Case Study 1

Case Study: How Does a Bike-Share Navigate Speedy Success?

Introduction

Welcome to the Cyclistic bike-share analysis case study! In this case study, you will perform many real-world tasks of a junior data analyst. You will work for a fictional company, Cyclistic, and meet different characters and team members. In order to answer the key business questions, you will follow the steps of the data analysis process: **ask**, **prepare**, **process**, **analyze**, **share**, and **act**. Along the way, the **Case Study Roadmap** tables — including guiding questions and key tasks — will help you stay on the right path.

By the end of this lesson, you will have a portfolio-ready case study. Download the packet and reference the details of this case study anytime. Then, when you begin your job hunt, your case study will be a tangible way to demonstrate your knowledge and skills to potential employers.

Scenario

You are a junior data analyst working in 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 compelling data insights and professional data visualizations.

Characters and teams

- **Cyclistic:** A bike-share program that features more than 5,800 bicycles and 600 docking stations. Cyclistic sets itself apart by also offering reclining bikes, hand tricycles, and cargo bikes, 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 them to commute to work each day.
- **Lily Moreno:** The director of marketing and your manager. Moreno is responsible for the development of campaigns and initiatives to promote the bike-share program. 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 — as well 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.

About the company

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

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

Cyclistic's finance analysts have concluded that annual members are much more profitable than casual riders. Although the pricing flexibility 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 all-new customers, Moreno believes there is a very good chance to convert casual riders into members. She notes that casual riders are already aware of the Cyclistic program and have chosen Cyclistic for their mobility needs.

Moreno has set a clear goal: Design marketing strategies aimed at converting casual riders into annual members. In order to do that, however, the marketing analyst 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.

Ask

Three questions will guide the future marketing program:

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

Moreno has assigned you the first question to answer: How do annual members and casual riders use Cyclistic bikes differently?

You will produce a report with the following deliverables:

1. A clear statement of the business task
2. A description of all data sources used
3. Documentation of any cleaning or manipulation of data
4. A summary of your analysis
5. Supporting visualizations and key findings
6. Your top three recommendations based on your analysis

Use the following Case Study Roadmap as a guide. Note: Completing this case study within a week is a good goal.

Case Study Roadmap - Ask
Guiding questions ● What is the problem you are trying to solve? ● How can your insights drive business decisions?
Key tasks 1. Identify the business task 2. Consider key stakeholders
Deliverable ☐ A clear statement of the business task

Prepare

You will use Cyclistic's historical trip data to analyze and identify trends. Download the previous 12 months of Cyclistic trip data here. (Note: The datasets have a different name because Cyclistic is a fictional company. For the purposes of this case study, the datasets are appropriate and will enable you to answer the business questions. The data has been made available by Motivate International Inc. under this license.) This is public data that you can use to explore how different customer types are using Cyclistic bikes. But note that data-privacy issues prohibit you from using riders' personally identifiable information. This means that you won't be able to connect pass purchases to credit card numbers to determine if casual riders live in the Cyclistic service area or if they have purchased multiple single passes.

Now, prepare your data for analysis using the following Case Study Roadmap as a guide:

Case Study Roadmap - Prepare
Guiding questions • Where is your data located? • How is the data organized? • Are there issues with bias or credibility in this data? Does your data ROCCC? • How are you addressing licensing, privacy, security, and accessibility? • How did you verify the data's integrity? • How does it help you answer your question? • Are there any problems with the data?
Key tasks 1. Download data and store it appropriately. 2. Identify how it's organized. 3. Sort and filter the data.

4. Determine the credibility of the data.
Deliverable ☐ A description of all data sources used

Process

Then, process your data for analysis using the following Case Study Roadmap as a guide:

Case Study Roadmap - Process
Guiding questions ● What tools are you choosing and why? ● Have you ensured your data's integrity? ● What steps have you taken to ensure that your data is clean? ● How can you verify that your data is clean and ready to analyze? ● Have you documented your cleaning process so you can review and share those results?
Key tasks 1. Check the data for errors. 2. Choose your tools. 3. Transform the data so you can work with it effectively. 4. Document the cleaning process.
Deliverable ☐ Documentation of any cleaning or manipulation of data

Follow these steps:

1. Download the previous 12 months of Cyclistic trip data.
2. Unzip the files.
3. Create a folder on your desktop or Drive to house the files. Use appropriate file-naming conventions.
4. Create subfolders for the .CSV file and the .XLS or Sheets file so that you have a copy of the original data. Move the downloaded files to the appropriate subfolder.
5. Follow these instructions for either Excel (a) or Google Sheets (b):
    a. Launch Excel, open each file, and choose to Save As an Excel Workbook file. Put it in the subfolder you created for .XLS files.
    b. Open each .CSV file in Google Sheets and save it to the appropriate subfolder.
6. Open your spreadsheet and create a column called “ride_length.” Calculate the length of each ride by subtracting the column “started_at” from the column “ended_at” (for example, =D2-C2) and format as HH:MM:SS using Format > Cells > Time > 37:30:55.
7. Create a column called “day_of_week,” and calculate the day of the week that each ride started using the “WEEKDAY” command (for example, =WEEKDAY(C2,1)) in each file. Format as General or as a number with no decimals, noting that 1 = Sunday and 7 = Saturday.
8. Proceed to the analyze step.

If you like, continue working with the data to better familiarize yourself and perhaps even identify new approaches to answering the business questions.

Analyze

Now that your data is stored appropriately and has been prepared for analysis, start putting it to work. Use the following Case Study Roadmap as a guide:

Case Study Roadmap - Analyze

Guiding questions

- How should you organize your data to perform analysis on it?
- Has your data been properly formatted?
- What surprises did you discover in the data?
- What trends or relationships did you find in the data?
- How will these insights help answer your business questions?

Key tasks

1. Aggregate your data so it's useful and accessible.
2. Organize and format your data.
3. Perform calculations.
4. Identify trends and relationships.

Deliverable

- ☐ A summary of your analysis

Follow these steps for using spreadsheets

Open your spreadsheet application, then complete the following steps:

1. Where relevant, make columns consistent and combine them into a single worksheet.
2. Clean and transform your data to prepare for analysis.
3. Conduct descriptive analysis.
4. Run a few calculations in one file to get a better sense of the data layout. Options:
    - Calculate the mean of ride_length
    - Calculate the max ride_length
    - Calculate the mode of day_of_week
5. Create a pivot table to quickly calculate and visualize the data. Options:
    - Calculate the average ride_length for members and casual riders. Try rows = member_casual; Values = Average

of ride_length.
    - Calculate the average ride_length for users by day_of_week. Try columns = day_of_week; Rows = member_casual; Values = Average of ride_length.
    - Calculate the number of rides for users by day_of_week by adding Count of trip_id to Values.
6. Open another file and perform the same descriptive analysis steps. Explore different seasons to make some initial observations.
7. Once you have spent some time working with the individual spreadsheets, merge them into a full-year view. Do this with the tool you have chosen to use to perform your final analysis, either a spreadsheet, a database and SQL, or R Studio.
8. Export a summary file for further analysis.

Follow these steps for using SQL

Open your SQL tool of choice, then complete the following steps:

1. Import your data.
2. Explore your data, perhaps looking at the total number of rows, distinct values, maximum, minimum, or mean values.
3. Where relevant, use JOIN statements to combine your relevant data into one table.
4. Create summary statistics.
5. Investigate interesting trends and save that information to a table.

Follow these steps for using R

Open R Studio and use this script to complete the following steps:

1. Import your data.
2. Make columns consistent and merge them into a single dataframe.
3. Clean up and add data to prepare for analysis.
4. Conduct descriptive analysis.
5. Export a summary file for further analysis.

Share

Now that you have performed your analysis and gained some insights into your data, create visualizations to share your findings. Moreno has reminded you that they should be sophisticated and polished in order to effectively communicate to the executive team. Use the following Case Study Roadmap as a guide:

Case Study Roadmap - Share
Guiding questions ● Were you able to answer the question of how annual members and casual riders use Cyclistic bikes differently? ● What story does your data tell? ● How do your findings relate to your original question? ● Who is your audience? What is the best way to communicate with them? ● Can data visualization help you share your findings? ● Is your presentation accessible to your audience?
Key tasks 1. Determine the best way to share your findings. 2. Create effective data visualizations. 3. Present your findings. 4. Ensure your work is accessible.
Deliverable ☐ Supporting visualizations and key findings

Follow these steps:

1. Take out a piece of paper and a pen and sketch some ideas for how you will visualize the data.
2. Once you choose a visual form, open your tool of choice to create your visualization. Use a presentation software, such

as PowerPoint or Google Slides; your spreadsheet program; Tableau; or R.

3. Create your data visualization, remembering that contrast should be used to draw your audience's attention to the most important insights. Use artistic principles including size, color, and shape.
4. Ensure clear meaning through the proper use of common elements, such as headlines, subtitles, and labels.
5. Refine your data visualization by applying deep attention to detail.

Act

Now that you have finished creating your visualizations, act on your findings. Prepare the deliverables Morena asked you to create, including the three top recommendations based on your analysis. Use the following Case Study Roadmap as a guide:

Case Study Roadmap - Act
Guiding questions ● What is your final conclusion based on your analysis? ● How could your team and business apply your insights? ● What next steps would you or your stakeholders take based on your findings? ● Is there additional data you could use to expand on your findings?
Key tasks 1. Create your portfolio. 2. Add your case study. 3. Practice presenting your case study to a friend or family member.
Deliverable ☐ Your top three recommendations based on your analysis

Follow these steps:

1. If you do not have one already, create an online portfolio. (Use Creating an Interactive Portfolio with Google Sites or Build a Portfolio with Google Sites.)
2. Consider how you want to feature your case study in your portfolio.
3. Upload or link your case study findings to your portfolio.
4. Write a brief paragraph describing the case study, your process, and your discoveries.
5. Add the paragraph to introduce your case study in your portfolio.

Wrap-up

Congratulations on finishing the Cyclistic bike-share case study! If you like, complete one of the other case studies to continue growing your portfolio. Or, use the steps from the **ask**, **prepare**, **process**, **analyze**, **share**, and **act** Case Study Roadmap to create a new project all your own. Best of luck on your job search!

Case Study

Scenario

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**. Your team will design a new marketing strategy to **convert casual riders into annual members** from these insights.

ASK

A clear statement of the business task The financial analysts have concluded that annual membership is much more profitable than single-ride and full-day passes from their analysis. So to make people opt for the yearly membership, our marketing campaign should urge the casual riders to convert to annual riders. As a solution, we should understand why casual riders would convert to a yearly membership? Based on the insights from the above question, we can achieve the maximum required conversion rate from casual to annual riders.

PREPARE

Guiding questions

1)Where is your data located? Data is uploaded in RStudio cloud where I could use the R programming language for the analysis.

2)How is data organized? Data is segregated into quarters from the year 2013 to 2020 till the first quarter of the latter year. Each year having its CSV file.

3) Are there any issues with bias or credibility in this data? Does your data ROCCC? The data has been collected directly from the company's customers, that is, bike riders so there is no issue of bias and credibility for the same reason. It is also Reliable, Original, Comprehensive, Current, and Cited„ which satisfies ROCCC.

4) How are you accessing licensing, privacy, security, and accessibility? The data was collected by Motivate International Inc. under the following license https://www.divvybikes.com/data-license-agreement Also the dataset does not contain any personal information about its customers (or riders) to violate the privacy.

5) How did you verify the data's integrity? The qualities required to verify the data integrity are accuracy, completeness, consistency, and trustworthiness. The data is complete as it contains all the required components to measure the entity. The data is consistent across the years with every year having its CSV file which is organized in an equal number of columns and same data types. As the credibility was proven before, it is also trustworthy.

6) How does it help to answer your question? By creating new features from existing ones like rideable_type, started_at, and ended_at(which are date-timestamp variables), we can deduce relationship between annual members and casual riders. The relationship analyzed will be useful to answer the question, that is, convert casual riders to annual members

7) Are there any problems with the data? Yes, the data had a couple of problems. There are few rows with 'N/A' values which needs to be removed. Also, there are duplicates which have to be eliminated.

PROCESS

Guiding questions

1)What tools are you choosing and why? The entries in the trips tables from the years 2004 to 2020 are enormous. Since this is the case it is always easy and helpful to navigate through the data using either databases or R programming language. I will be using R language to deal with the data in this case study.

2) Have you ensured your data's integrity? The qualities required to verify the data integrity are accuracy, completeness, consistency, and trustworthiness. The data is complete as it contains all the required components to measure the entity. The data is consistent across the years with every year having its CSV file which is organized in an equal number of columns and same data types. As the credibility was proven before, it is also trustworthy.

3) What steps have you taken to ensure that your data is clean? a) I have concatenated all the CSV files of each year into a single data frame b) Removed all the empty rows and columns from the concatenated data frame. c) Checked the unique values in each variable using **count()** so that there is no misspelling anywhere. d) Omitted **N/A** values from the entire data frame. e) Removed duplicates

4) How can you verify that your data is clean and ready to analyze? After performing all the cleaning tasks mentioned above, I ran the below functions to verify: a) Used filter() to check if there were any missing values b) Used count() to check the unique values of each variable c) Used duplicated() to check for any duplicates present.

5) Have you documented your cleaning process so you can review and share those results? Yes, please find the below comments and snippets for the documentation.

Let's install and load all the required packages

```
install.packages("rmarkdown", repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/user/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)

## package 'rmarkdown' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##     C:\Users\user\AppData\Local\Temp\Rtmp44GrhA\downloaded_packages
```

```
install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/user/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)

## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##     C:\Users\user\AppData\Local\Temp\Rtmp44GrhA\downloaded_packages
```

```r
install.packages("janitor", repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/user/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)

## package 'janitor' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##     C:\Users\user\AppData\Local\Temp\Rtmp44GrhA\downloaded_packages
```

```r
install.packages("scales", repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/user/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)

## package 'scales' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##     C:\Users\user\AppData\Local\Temp\Rtmp44GrhA\downloaded_packages
```

```r
install.packages("knitr", repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/user/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)

## package 'knitr' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##     C:\Users\user\AppData\Local\Temp\Rtmp44GrhA\downloaded_packages
```

```r
library("tidyverse")
```

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

## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library("janitor")
```

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

```
library("scales")
```

```
##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
library("knitr")
```

1)Concatenating all the CSVs into a single data frame. Let's load all the individual CSVs and concatenate.

```
df1  <- read_csv("E:/Tridata/202004-divvy-tripdata.csv")
```

```
## Rows: 84776 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): ride_id, rideable_type, start_station_name, end_station_name, memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, e...
## 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.
```

```
df2  <- read_csv("E:/Tridata/202005-divvy-tripdata.csv")
```

```
## Rows: 200274 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): ride_id, rideable_type, start_station_name, end_station_name, memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, e...
## 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.
```

```
df3  <- read_csv("E:/Tridata/202006-divvy-tripdata.csv")
```

```
## Rows: 343005 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): ride_id, rideable_type, start_station_name, end_station_name, memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, e...
## 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.
```

```r
df4  <- read_csv("E:/Tridata/202007-divvy-tripdata.csv")
```

```
## Rows: 551480 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): ride_id, rideable_type, start_station_name, end_station_name, memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, e...
## 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.
```

```r
df5  <- read_csv("E:/Tridata/202008-divvy-tripdata.csv")
```

```
## Rows: 622361 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): ride_id, rideable_type, start_station_name, end_station_name, memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, e...
## 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.
```

```r
df6  <- read_csv("E:/Tridata/202009-divvy-tripdata.csv")
```

```
## Rows: 532958 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): ride_id, rideable_type, start_station_name, end_station_name, memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, e...
## 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.
```

```r
df7  <- read_csv("E:/Tridata/202010-divvy-tripdata.csv")
```

```
## Rows: 388653 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): ride_id, rideable_type, start_station_name, end_station_name, memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, e...
## 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.
```

```r
df8  <- read_csv("E:/Tridata/202011-divvy-tripdata.csv")
```

```
## Rows: 259716 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): ride_id, rideable_type, start_station_name, end_station_name, memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, e...
## 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.
```

```
df9  <- read_csv("E:/Tridata/202012-divvy-tripdata.csv")
```

```
## Rows: 131573 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.
```

```
df10 <- read_csv("E:/Tridata/202101-divvy-tripdata.csv")
```

```
## Rows: 96834 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.
```

```
df11 <- read_csv("E:/Tridata/202102-divvy-tripdata.csv")
```

```
## Rows: 49622 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.
```

```
df12 <- read_csv("E:/Tridata/202103-divvy-tripdata.csv")
```

```
## Rows: 228496 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.
```

```
binded_df <- rbind(df1, df2, df3, df4, df5, df6, df7, df8, df9, df10, df11, df12)
```

2)Removing any empty rows or columns present and checking for missing values (Check the missing values for each variable)

```
new_binded_df <- remove_empty(binded_df, which=c("rows", "cols"))
count(filter(new_binded_df, start_station_name==''),start_station_name, member_casual,sort=TRUE)
```

```
## # A tibble: 0 x 3
## # ... with 3 variables: start_station_name <chr>, member_casual <chr>, n <int>
```

3)count() returns unique values of the variable passed

```
binded_df %>%
  count(rideable_type)
```

```
## # A tibble: 3 x 2
##   rideable_type       n
##   <chr>           <int>
## 1 classic_bike   319873
## 2 docked_bike   2558469
## 3 electric_bike  611406
```

4)omitting NA values in the entire data frame

```
new_binded_df <- na.omit(binded_df)
```

5)Removing duplicates

```
new_binded_df_no_dups <- new_binded_df[!duplicated(new_binded_df$ride_id), ]
```

Analyze

Guiding Questions

1) How should you organize your data to perform analysis on it? Since the data sources contain separate CSV files for all the years and their respective quarters, after downloading them, I combined them into a single data frame. This combination was possible because all the CSV files had the same number and type of variables.

Also I created new features using the existing ones. Check them below: **a) riding_time**

```
clean_df <- new_binded_df_no_dups
clean_df <- clean_df %>%
  mutate(riding_time = as.numeric(ended_at-started_at)/60)
clean_df
```

```
## # A tibble: 3,294,483 x 14
##    ride_id       ridea~1 started_at          ended_at            start~2 start~3
##    <chr>         <chr>   <dttm>              <dttm>              <chr>   <chr>
##  1 A847FADBBC63~ docked~ 2020-04-26 17:45:14 2020-04-26 18:12:03 Eckhar~ 86
##  2 5405B80E996F~ docked~ 2020-04-17 17:08:54 2020-04-17 17:17:03 Drake ~ 503
##  3 5DD24A79A4E0~ docked~ 2020-04-01 17:54:13 2020-04-01 18:08:36 McClur~ 142
##  4 2A59BBDF5CDB~ docked~ 2020-04-07 12:50:19 2020-04-07 13:02:31 Califo~ 216
##  5 27AD306C119C~ docked~ 2020-04-18 10:22:59 2020-04-18 11:15:54 Rush S~ 125
##  6 356216E87513~ docked~ 2020-04-30 17:55:47 2020-04-30 18:01:11 Mies v~ 173
##  7 A2759CB06A81~ docked~ 2020-04-02 14:47:19 2020-04-02 14:52:32 Street~ 35
##  8 FC8BC2E2D54F~ docked~ 2020-04-07 12:22:20 2020-04-07 13:38:09 Ogden ~ 434
##  9 9EC5648678DE~ docked~ 2020-04-15 10:30:11 2020-04-15 10:35:55 LaSall~ 627
## 10 A8FFF89140C3~ docked~ 2020-04-04 15:02:28 2020-04-04 15:19:47 Kedzie~ 377
## # ... with 3,294,473 more rows, 8 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>, riding_time <dbl>, and abbreviated
## #   variable names 1: rideable_type, 2: start_station_name, 3: start_station_id
```

b) year_month

```
clean_df <- clean_df %>%
  mutate(year_month=paste(strftime(clean_df$started_at, "%Y"), "-",
                          strftime(clean_df$started_at, "%m"), "-",
                          strftime(clean_df$started_at, "%b")))
clean_df
```

```
## # A tibble: 3,294,483 x 15
##    ride_id       ridea~1 started_at          ended_at            start~2 start~3
##    <chr>         <chr>   <dttm>              <dttm>              <chr>   <chr>
##  1 A847FADBBC63~ docked~ 2020-04-26 17:45:14 2020-04-26 18:12:03 Eckhar~ 86
##  2 5405B80E996F~ docked~ 2020-04-17 17:08:54 2020-04-17 17:17:03 Drake ~ 503
##  3 5DD24A79A4E0~ docked~ 2020-04-01 17:54:13 2020-04-01 18:08:36 McClur~ 142
##  4 2A59BBDF5CDB~ docked~ 2020-04-07 12:50:19 2020-04-07 13:02:31 Califo~ 216
##  5 27AD306C119C~ docked~ 2020-04-18 10:22:59 2020-04-18 11:15:54 Rush S~ 125
##  6 356216E87513~ docked~ 2020-04-30 17:55:47 2020-04-30 18:01:11 Mies v~ 173
##  7 A2759CB06A81~ docked~ 2020-04-02 14:47:19 2020-04-02 14:52:32 Street~ 35
##  8 FC8BC2E2D54F~ docked~ 2020-04-07 12:22:20 2020-04-07 13:38:09 Ogden ~ 434
##  9 9EC5648678DE~ docked~ 2020-04-15 10:30:11 2020-04-15 10:35:55 LaSall~ 627
## 10 A8FFF89140C3~ docked~ 2020-04-04 15:02:28 2020-04-04 15:19:47 Kedzie~ 377
## # ... with 3,294,473 more rows, 9 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>, riding_time <dbl>, year_month <chr>,
## #   and abbreviated variable names 1: rideable_type, 2: start_station_name,
## #   3: start_station_id
```

Removing year_month with "2021 - 06 (Jun)" values from the data frame as June's month contains very few rows which are not helpful in our analysis.

```
clean_df <- filter(clean_df, year_month!="2021 - 06 (Jun)")
clean_df
```

```
## # A tibble: 3,294,483 x 15
##    ride_id       ridea~1 started_at          ended_at            start~2 start~3
##    <chr>         <chr>   <dttm>              <dttm>              <chr>   <chr>
##  1 A847FADBBC63~ docked~ 2020-04-26 17:45:14 2020-04-26 18:12:03 Eckhar~ 86
##  2 5405B80E996F~ docked~ 2020-04-17 17:08:54 2020-04-17 17:17:03 Drake ~ 503
##  3 5DD24A79A4E0~ docked~ 2020-04-01 17:54:13 2020-04-01 18:08:36 McClur~ 142
##  4 2A59BBDF5CDB~ docked~ 2020-04-07 12:50:19 2020-04-07 13:02:31 Califo~ 216
##  5 27AD306C119C~ docked~ 2020-04-18 10:22:59 2020-04-18 11:15:54 Rush S~ 125
##  6 356216E87513~ docked~ 2020-04-30 17:55:47 2020-04-30 18:01:11 Mies v~ 173
##  7 A2759CB06A81~ docked~ 2020-04-02 14:47:19 2020-04-02 14:52:32 Street~ 35
##  8 FC8BC2E2D54F~ docked~ 2020-04-07 12:22:20 2020-04-07 13:38:09 Ogden ~ 434
##  9 9EC5648678DE~ docked~ 2020-04-15 10:30:11 2020-04-15 10:35:55 LaSall~ 627
## 10 A8FFF89140C3~ docked~ 2020-04-04 15:02:28 2020-04-04 15:19:47 Kedzie~ 377
## # ... with 3,294,473 more rows, 9 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>, riding_time <dbl>, year_month <chr>,
## #   and abbreviated variable names 1: rideable_type, 2: start_station_name,
## #   3: start_station_id
```

c) Weekday

```
clean_df <- clean_df %>%
  mutate(weekday=strftime(clean_df$ended_at, "%a"))
clean_df
```

```
## # A tibble: 3,294,483 x 16
##    ride_id       ridea~1 started_at          ended_at            start~2 start~3
##    <chr>         <chr>   <dttm>              <dttm>              <chr>   <chr>
##  1 A847FADBBC63~ docked~ 2020-04-26 17:45:14 2020-04-26 18:12:03 Eckhar~ 86
##  2 5405B80E996F~ docked~ 2020-04-17 17:08:54 2020-04-17 17:17:03 Drake ~ 503
##  3 5DD24A79A4E0~ docked~ 2020-04-01 17:54:13 2020-04-01 18:08:36 McClur~ 142
##  4 2A59BBDF5CDB~ docked~ 2020-04-07 12:50:19 2020-04-07 13:02:31 Califo~ 216
##  5 27AD306C119C~ docked~ 2020-04-18 10:22:59 2020-04-18 11:15:54 Rush S~ 125
##  6 356216E87513~ docked~ 2020-04-30 17:55:47 2020-04-30 18:01:11 Mies v~ 173
##  7 A2759CB06A81~ docked~ 2020-04-02 14:47:19 2020-04-02 14:52:32 Street~ 35
##  8 FC8BC2E2D54F~ docked~ 2020-04-07 12:22:20 2020-04-07 13:38:09 Ogden ~ 434
##  9 9EC5648678DE~ docked~ 2020-04-15 10:30:11 2020-04-15 10:35:55 LaSall~ 627
## 10 A8FFF89140C3~ docked~ 2020-04-04 15:02:28 2020-04-04 15:19:47 Kedzie~ 377
## # ... with 3,294,473 more rows, 10 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>, riding_time <dbl>, year_month <chr>,
## #   weekday <chr>, and abbreviated variable names 1: rideable_type,
## #   2: start_station_name, 3: start_station_id
```

d) start_hour

```
clean_df <- clean_df %>%
  mutate(start_hour=strftime(clean_df$ended_at, format = "%H",tz = "UTC"))
clean_df
```

```
## # A tibble: 3,294,483 x 17
##    ride_id       ridea~1 started_at          ended_at            start~2 start~3
##    <chr>         <chr>   <dttm>              <dttm>              <chr>   <chr>
##  1 A847FADBBC63~ docked~ 2020-04-26 17:45:14 2020-04-26 18:12:03 Eckhar~ 86
##  2 5405B80E996F~ docked~ 2020-04-17 17:08:54 2020-04-17 17:17:03 Drake ~ 503
##  3 5DD24A79A4E0~ docked~ 2020-04-01 17:54:13 2020-04-01 18:08:36 McClur~ 142
##  4 2A59BBDF5CDB~ docked~ 2020-04-07 12:50:19 2020-04-07 13:02:31 Califo~ 216
##  5 27AD306C119C~ docked~ 2020-04-18 10:22:59 2020-04-18 11:15:54 Rush S~ 125
##  6 356216E87513~ docked~ 2020-04-30 17:55:47 2020-04-30 18:01:11 Mies v~ 173
##  7 A2759CB06A81~ docked~ 2020-04-02 14:47:19 2020-04-02 14:52:32 Street~ 35
##  8 FC8BC2E2D54F~ docked~ 2020-04-07 12:22:20 2020-04-07 13:38:09 Ogden ~ 434
##  9 9EC5648678DE~ docked~ 2020-04-15 10:30:11 2020-04-15 10:35:55 LaSall~ 627
## 10 A8FFF89140C3~ docked~ 2020-04-04 15:02:28 2020-04-04 15:19:47 Kedzie~ 377
## # ... with 3,294,473 more rows, 11 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>, riding_time <dbl>, year_month <chr>,
## #   weekday <chr>, start_hour <chr>, and abbreviated variable names
## #   1: rideable_type, 2: start_station_name, 3: start_station_id
```

2) Has your data been properly formatted? Yes, the data has been properly formatted with respective to their values.

3) What surprises did you discover in the data? Surprisingly, I found many NA values in the information about station names and ids (combined). Maybe while collecting the data, customers were not sure about the station's id numbers. However, the NA values in these fields will not affect our analysis.

4) What trends or relationships did you find in the data? Let's compare the member_casual feature with other newly created features to find any trends:

a) Let's start by comparing the number of members and casual riders

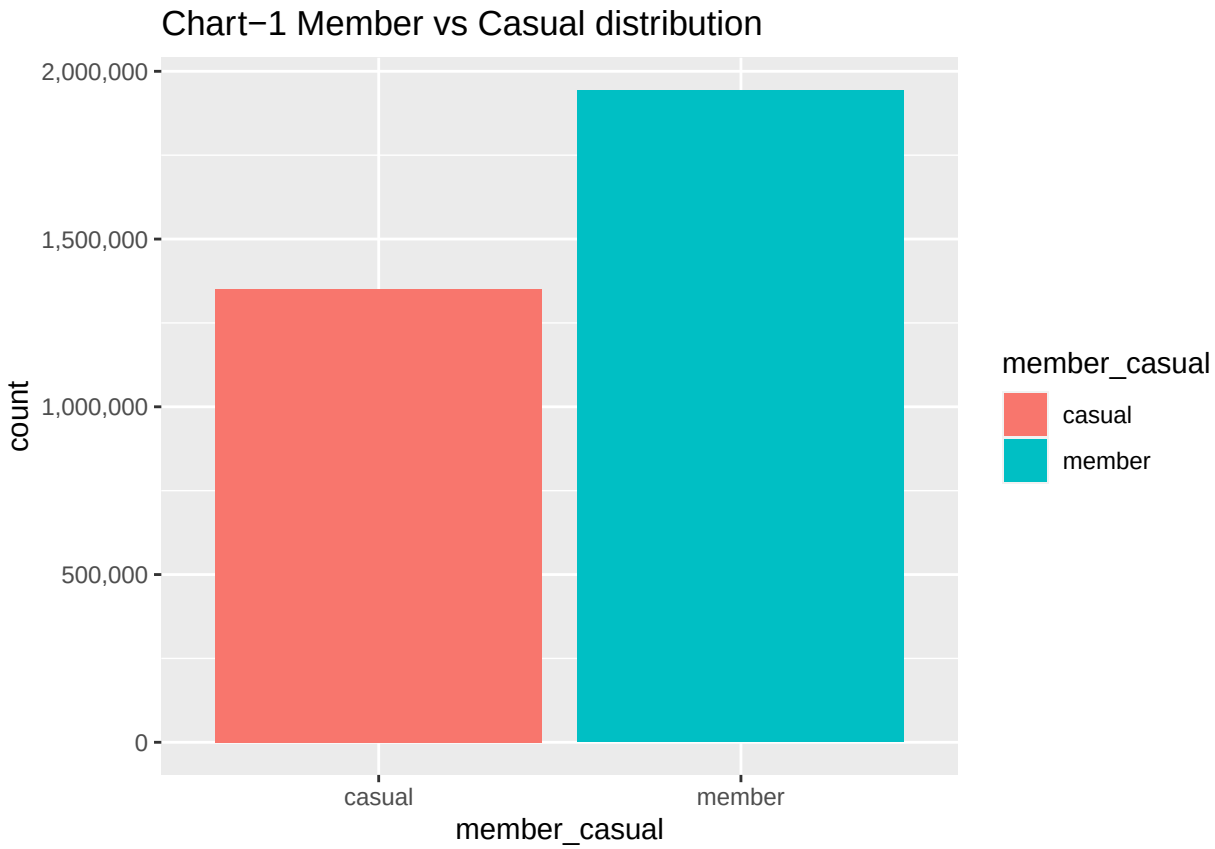
```
df <- clean_df
df %>%
  group_by(member_casual) %>%
  summarize(count=length(ride_id),
            percentage_of_total=(length(ride_id)/nrow(df))*100)
```

```
## # A tibble: 2 x 3
##   member_casual   count percentage_of_total
##   <chr>           <int>               <dbl>
## 1 casual        1351214                41.0
## 2 member        1943269                59.0
```

From above, it is known that 58% of the total riders in the last 12 months were annual members. The remaining(42%) are casual riders.

Let's plot the above table.

```
ggplot(df, aes(member_casual, fill=member_casual))+
  geom_bar()+
  labs(title="Chart-1 Member vs Casual distribution")+
  scale_y_continuous(labels=comma)
```



b) Let's check what percent of annual and casual riders ride every month

```
df %>%
  group_by(year_month) %>%
  summarize(count=length(ride_id),
            percentage_of_total=(length(ride_id)/nrow(df))*100,
            members_count=sum(member_casual=="member"),
            members_percent=(sum(member_casual=="member")/length(ride_id))*100,
            casual_count=sum(member_casual=="casual"),
            casual_percent=(sum(member_casual=="casual")/length(ride_id))*100) %>%
  arrange(year_month)
```

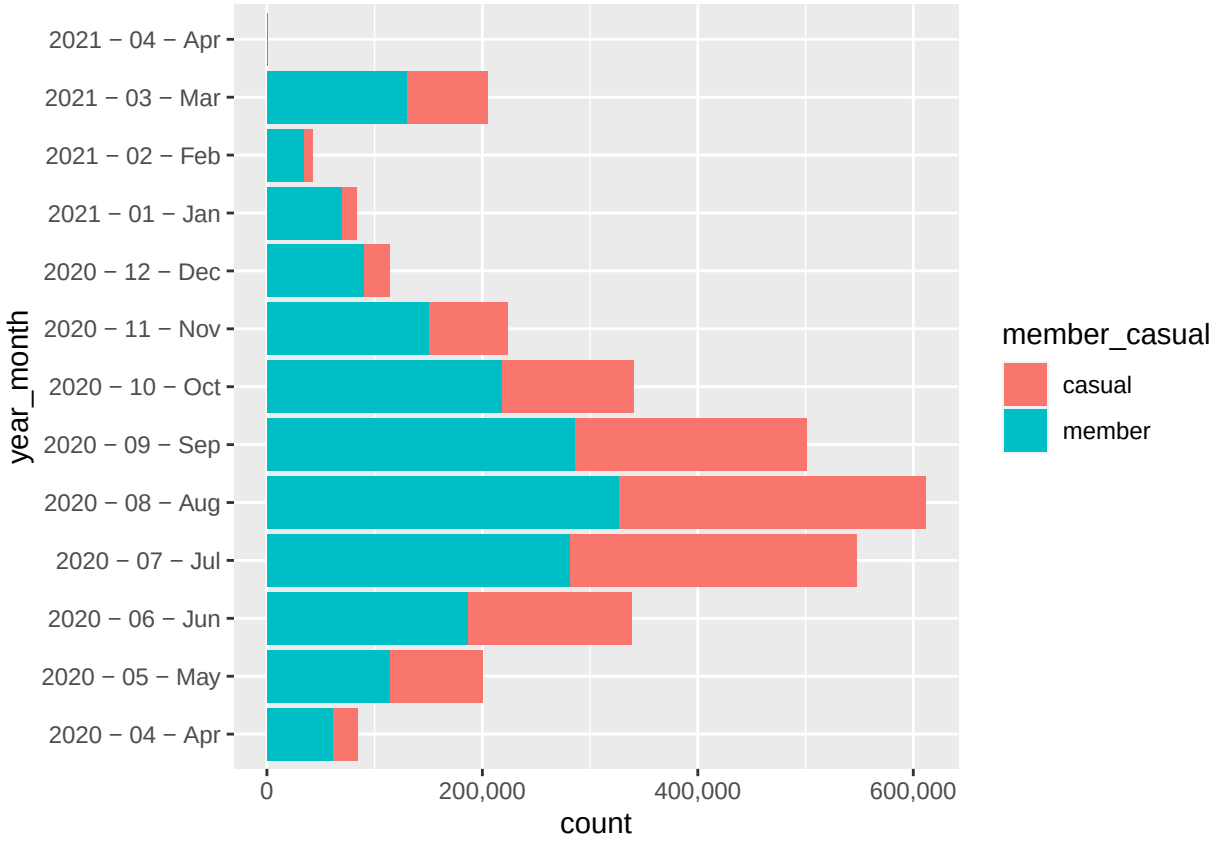
```
## # A tibble: 13 x 7
##    year_month       count percentage_of_total members_~1 membe~2 casua~3 casua~4
##    <chr>            <int>               <dbl>      <int>   <dbl>   <int>   <dbl>
##  1 2020 - 04 - Apr  83978              2.55        60570    72.1   23408    27.9
##  2 2020 - 05 - May 200652              6.09       113709    56.7   86943    43.3
##  3 2020 - 06 - Jun 338163             10.3        185869    55.0  152294    45.0
##  4 2020 - 07 - Jul 548013             16.6        281020    51.3  266993    48.7
##  5 2020 - 08 - Aug 611231             18.6        326042    53.3  285189    46.7
##  6 2020 - 09 - Sep 501353             15.2        285382    56.9  215971    43.1
##  7 2020 - 10 - Oct 340912             10.3        217622    63.8  123290    36.2
##  8 2020 - 11 - Nov 223732              6.79       150170    67.1   73562    32.9
##  9 2020 - 12 - Dec 113991              3.46        89441    78.5   24550    21.5
## 10 2021 - 01 - Jan  83696              2.54        68958    82.4   14738    17.6
## 11 2021 - 02 - Feb  42745              1.30        34212    80.0    8533    20.0
```

```
## 12 2021 - 03 - Mar 205002              6.22       129511    63.2   75491    36.8
## 13 2021 - 04 - Apr   1015              0.0308        763    75.2     252    24.8
## # ... with abbreviated variable names 1: members_count, 2: members_percent,
## #   3: casual_count, 4: casual_percent
```

As can be seen, August had a more number of riders than any other month. However, the percentage of annual members every month is more than the casual riders', which is a good thing. Our goal here would be to maximize the percent of members every month. Also, the number of riders started decreasing drastically in the peak winter months (November-February)

Let's plot the above table.

```
ggplot(df, aes(year_month, fill=member_casual))+
  geom_bar()+
  coord_flip()+
  scale_y_continuous(labels=comma)
```



c) Let's now check how riders ride in each hour of the day. Also, later we'll check how this varies per each day of the week

```
start_hour_df <- df %>%
  group_by(start_hour) %>%
  summarize(count=length(ride_id),
            percentage_of_total=(length(ride_id)/nrow(df))*100,
            members_count=sum(member_casual=="member"),
            members_percent=(sum(member_casual=="member")/length(ride_id))*100,
            casual_count=sum(member_casual=="casual"),
```

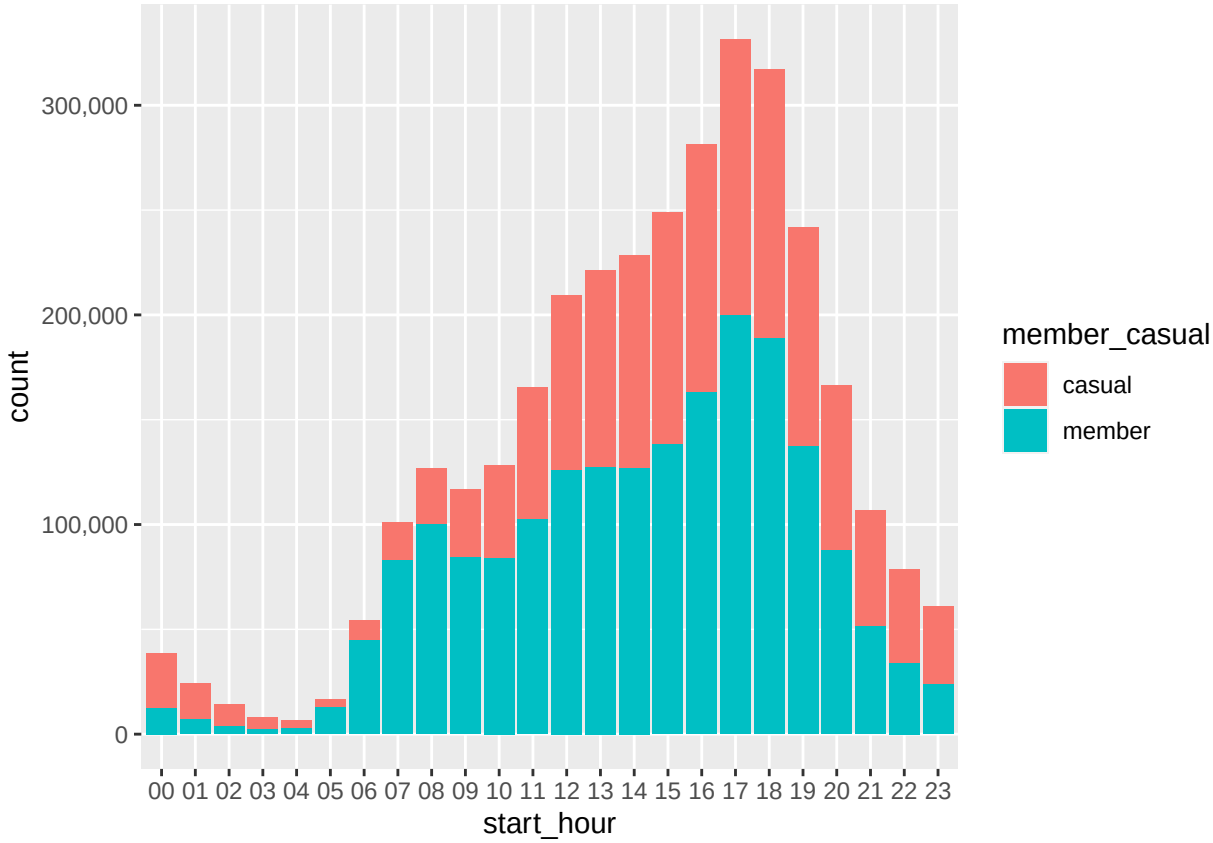
```
            casual_percent=(sum(member_casual=="casual")/length(ride_id))*100) %>%
  arrange(start_hour)
start_hour_df
```

```
## # A tibble: 24 x 7
##    start_hour  count percentage_of_total members_count members~1 casua~2 casua~3
##    <chr>       <int>               <dbl>         <int>     <dbl>   <int>   <dbl>
##  1 00          38744               1.18          12533      32.3   26211    67.7
##  2 01          24343               0.739          6932      28.5   17411    71.5
##  3 02          14205               0.431          3693      26.0   10512    74.0
##  4 03           8278               0.251          2159      26.1    6119    73.9
##  5 04           6732               0.204          2762      41.0    3970    59.0
##  6 05          16853               0.512         12810      76.0    4043    24.0
##  7 06          54503               1.65          44674      82.0    9829    18.0
##  8 07         101000               3.07          82744      81.9   18256    18.1
##  9 08         126904               3.85          99898      78.7   27006    21.3
## 10 09         116737               3.54          84393      72.3   32344    27.7
## # ... with 14 more rows, and abbreviated variable names 1: members_percent,
## #   2: casual_count, 3: casual_percent
```

The maximum number of riders is in the 17th hour, with 10% of the total riders. The number of member riders starts significantly increasing from the 5th hour and moderately decreases as the day passes. On the other hand, the number of casual riders peaks at midnight.

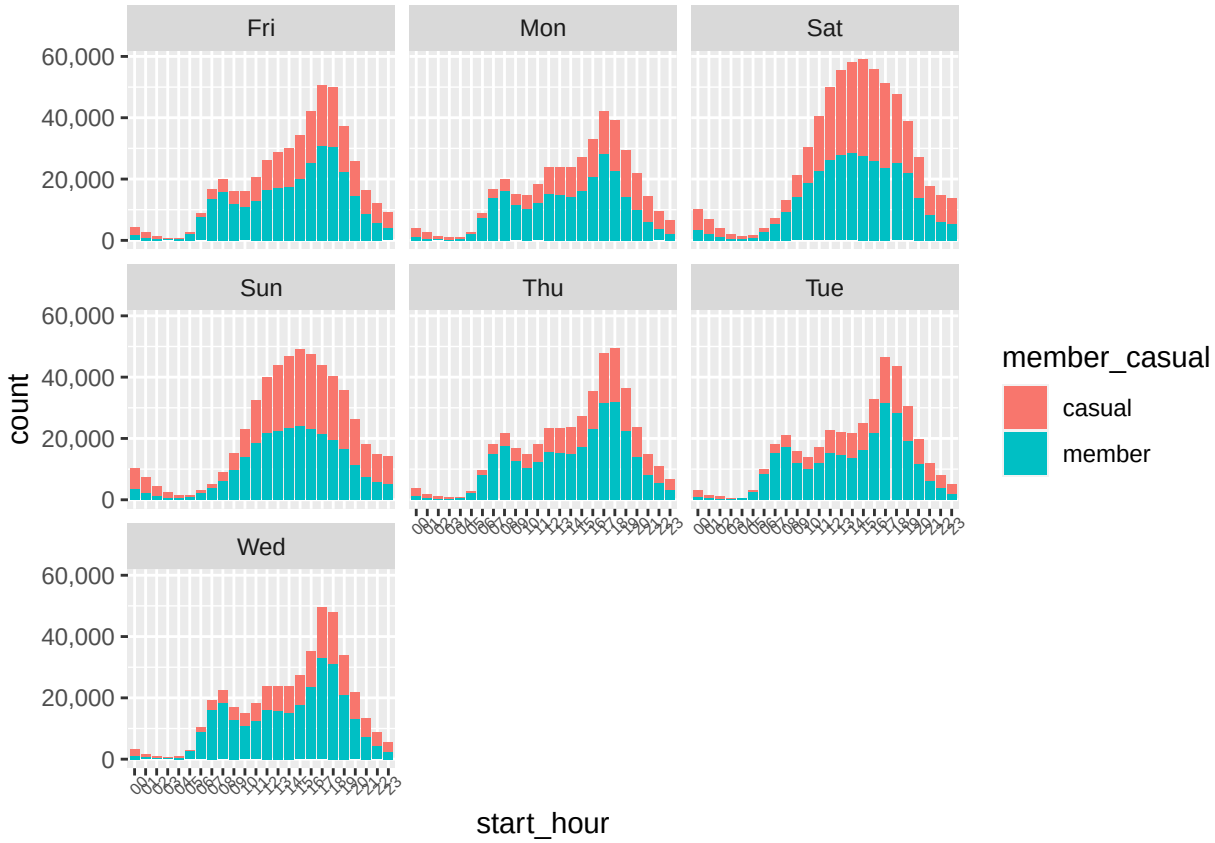
Let's plot the above table.

```
ggplot(df, aes(start_hour, fill=member_casual))+
  geom_bar()+
  scale_y_continuous(labels=comma)
```



Now let's plot the same comparison for each day of the week.

```
ggplot(df, aes(start_hour, fill=member_casual))+
  geom_bar()+
  facet_wrap(~weekday)+
  scale_y_continuous(labels=comma)+
  theme(axis.text.x = element_text(size=6, angle=45))
```



We can see that the number of casual riders is more on the weekends than on weekdays (where annual members are more).

To more comprehend the above analysis, let's divide the hours into morning, afternoon, and evening.

```
df <- mutate(df, hour_of_the_day=ifelse(df$start_hour<12, "Morning",
                                        ifelse(df$start_hour>=12 & df$start_hour<19, "Afternoon", "Even
```

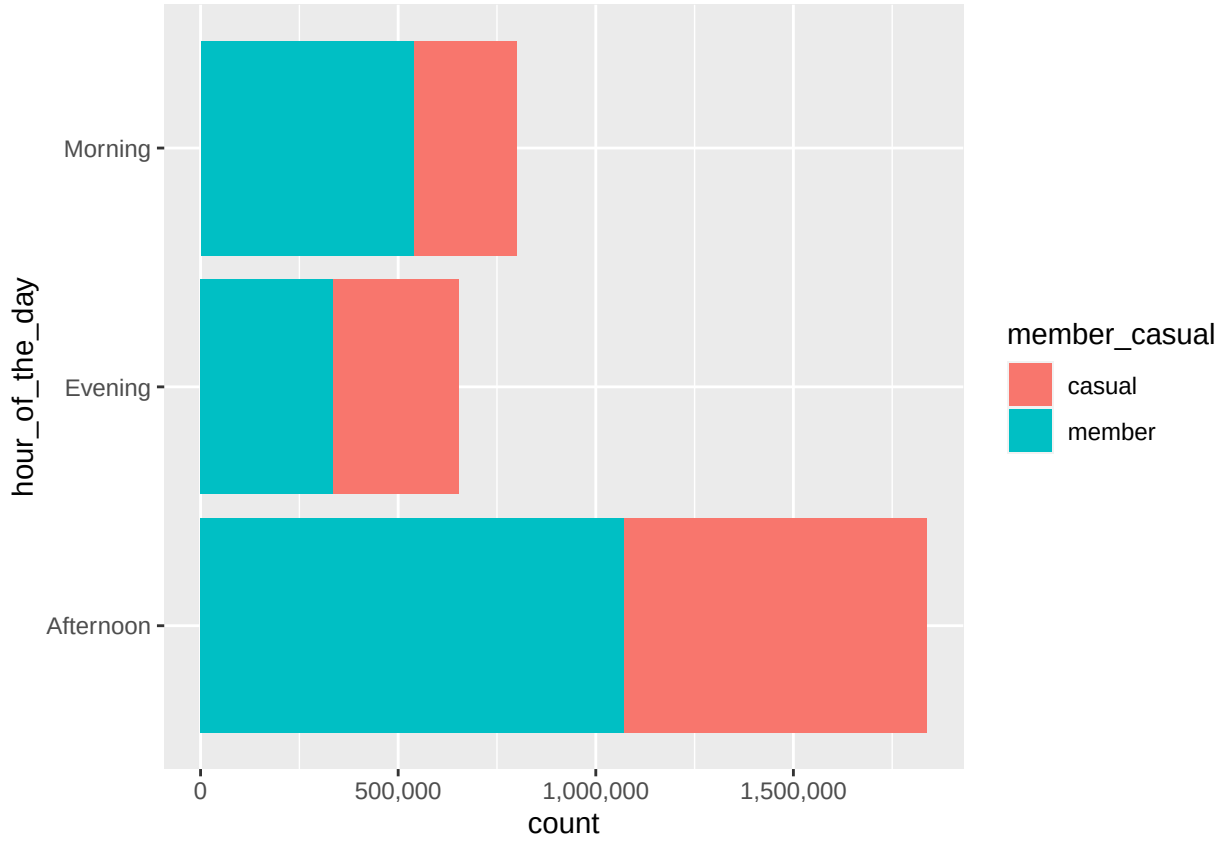
```
hour_type_df <- df %>%
  group_by(hour_of_the_day) %>%
  summarize(count=length(ride_id),
            percentage_of_total=(length(ride_id)/nrow(df))*100,
            members_count=sum(member_casual=="member"),
            members_percent=(sum(member_casual=="member")/length(ride_id))*100,
            casual_count=sum(member_casual=="casual"),
            casual_percent=(sum(member_casual=="casual")/length(ride_id))*100)
hour_type_df
```

```
## # A tibble: 3 x 7
##   hour_of_the_day   count percentage_of_total members_~1 membe~2 casua~3 casua~4
##   <chr>             <int>               <dbl>      <int>   <dbl>   <int>   <dbl>
## 1 Afternoon       1837960                55.8    1070358    58.2  767602    41.8
## 2 Evening          654796                19.9     334039    51.0  320757    49.0
## 3 Morning          801727                24.3     538872    67.2  262855    32.8
## # ... with abbreviated variable names 1: members_count, 2: members_percent,
## #   3: casual_count, 4: casual_percent
```

Mornings had more number of annual riders whereas evening has more number of casual riders. However, afternoon had more number of total riders compared to mornings or evenings.

Let's plot the above table

```
ggplot(df, aes(hour_of_the_day, fill=member_casual))+
  geom_bar()+
  #facet_wrap(~hour_of_the_day, scales = "free")+
  scale_y_continuous(labels=comma)+
  coord_flip()
```



d) Let's check how number of riders vary per each week of the day

```
df %>%
  group_by(weekday) %>%
  summarize(count=length(ride_id),
            percentage_of_total=(length(ride_id)/nrow(df))*100,
            members_count=sum(member_casual=="member"),
            members_percent=(sum(member_casual=="member")/length(ride_id))*100,
            casual_count=sum(member_casual=="casual"),
            casual_percent=(sum(member_casual=="casual")/length(ride_id))*100)
```

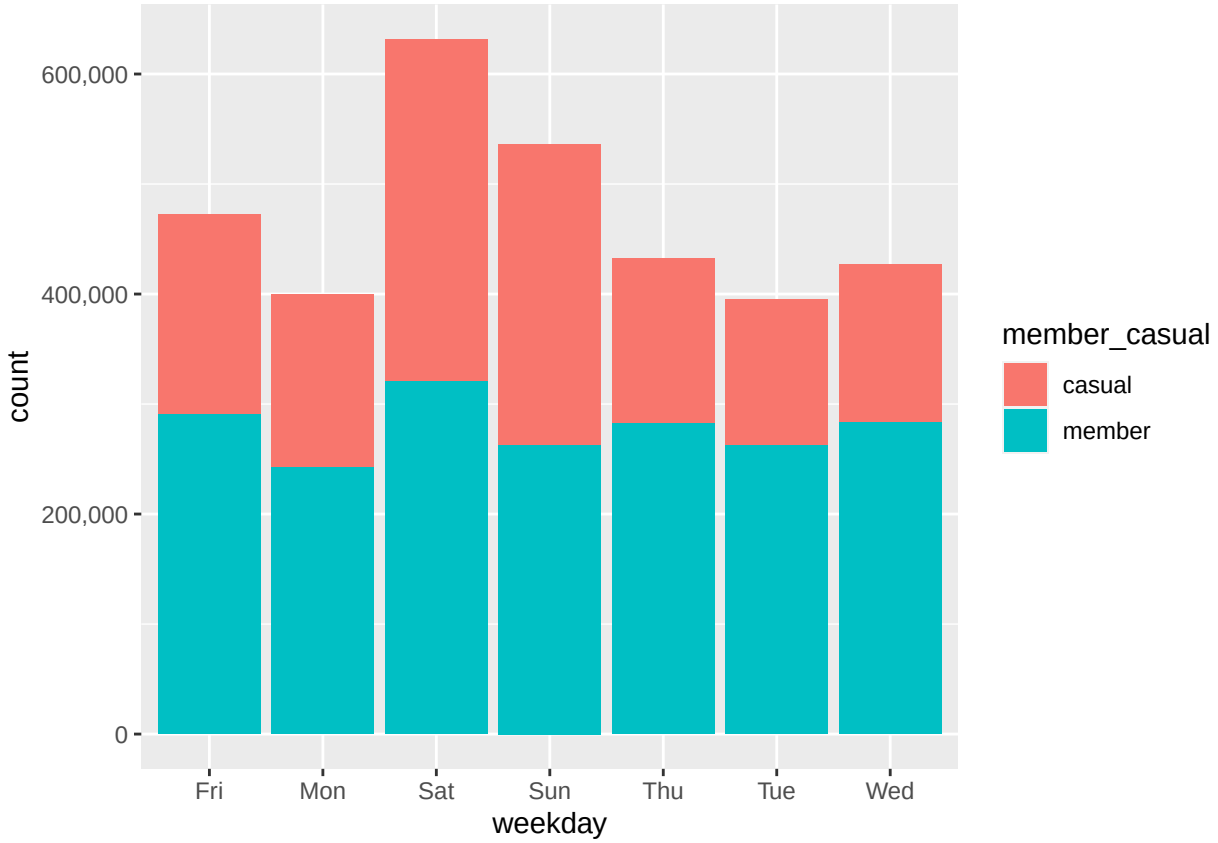
```
## # A tibble: 7 x 7
##   weekday  count percentage_of_total members_count members_per~1 casua~2 casua~3
##   <chr>    <int>               <dbl>         <int>         <dbl>   <int>   <dbl>
## 1 Fri     472476                14.3        290404          61.5  182072    38.5
```

```
## 2 Mon     399844                12.1        242419          60.6  157425    39.4
## 3 Sat     631591                19.2        320108          50.7  311483    49.3
## 4 Sun     535663                16.3        262828          49.1  272835    50.9
## 5 Thu     432672                13.1        282460          65.3  150212    34.7
## 6 Tue     395480                12.0        262104          66.3  133376    33.7
## 7 Wed     426757                13.0        282946          66.3  143811    33.7
## # ... with abbreviated variable names 1: members_percent, 2: casual_count,
## #   3: casual_percent
```

Saturdays and Sundays had more casual riders than annual members. Members usually ride on the weekdays due to work.

Let's plot the above table.

```
ggplot(df, aes(weekday, fill=member_casual))+
  geom_bar()+
  scale_y_continuous(labels=comma)
```



e) Let's check what types of bikes do riders usually ride

```
df %>%
  group_by(rideable_type) %>%
  summarize(count=length(ride_id),
            percentage_of_total=(length(ride_id)/nrow(df))*100,
            members_count=sum(member_casual=="member"),
            members_percent=(sum(member_casual=="member")/length(ride_id))*100,
```

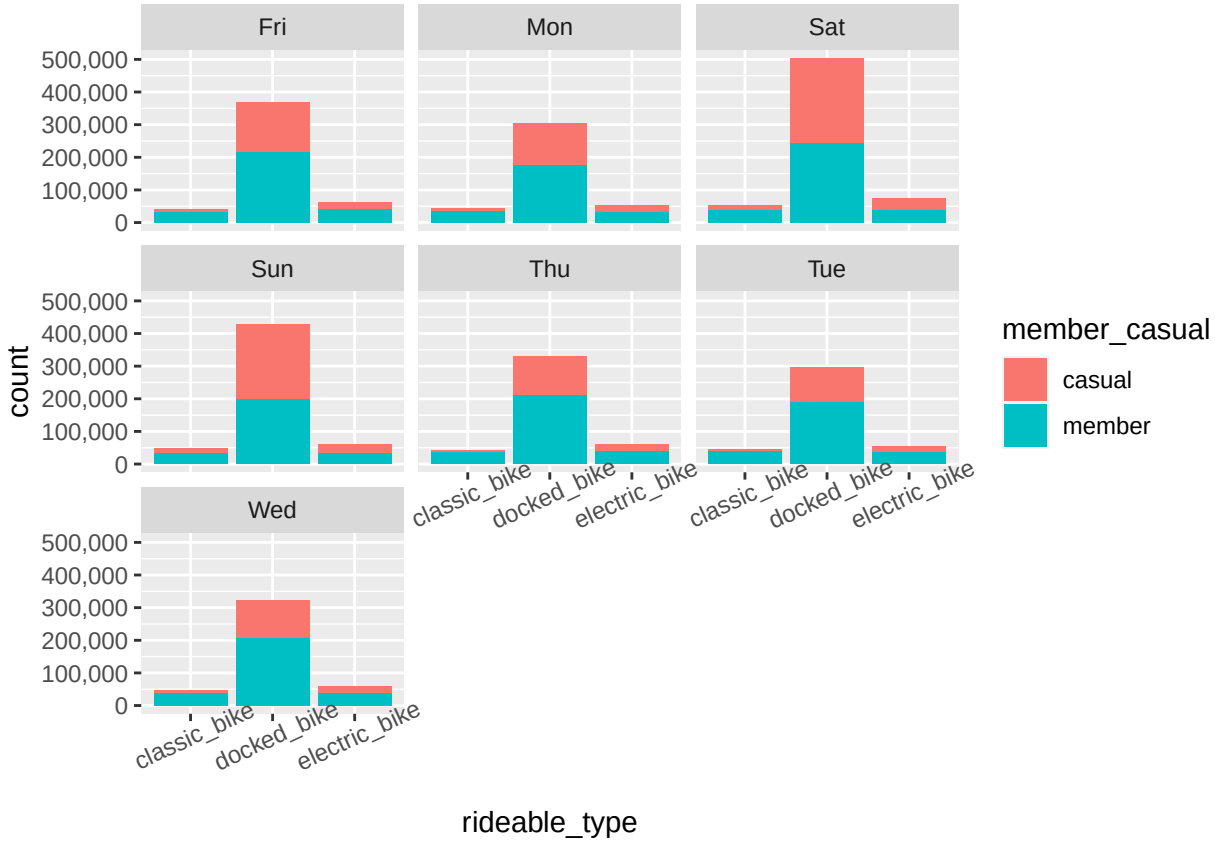
```
            casual_count=sum(member_casual=="casual"),
            casual_percent=(sum(member_casual=="casual")/length(ride_id))*100)
```

```
## # A tibble: 3 x 7
##   rideable_type   count percentage_of_total members_co~1 membe~2 casua~3 casua~4
##   <chr>           <int>               <dbl>        <int>   <dbl>   <int>   <dbl>
## 1 classic_bike   318614                9.67       248181    77.9   70433    22.1
## 2 docked_bike   2554083               77.5       1439492    56.4 1114591    43.6
## 3 electric_bike  421786               12.8        255596    60.6  166190    39.4
## # ... with abbreviated variable names 1: members_count, 2: members_percent,
## #   3: casual_count, 4: casual_percent
```

It seems docked bikes are more preferred over classic and electric bikes. However, riders have also chosen classic and electric too. Maybe the company has more docked bikes.

Let's plot the above table but we'll also how this trend works for each day.

```
ggplot(df, aes(rideable_type, fill=member_casual))+
  geom_bar()+
  scale_y_continuous(labels=comma)+
  facet_wrap(~weekday)+
  theme(axis.text.x = element_text(angle=25))
```



f) Let's consider riding_time feature now Let's print the summary of the riding_time variable to check if there are any anomalies.

```
summary(df$riding_time)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -29049.97      8.03     14.70     27.03     26.87  58720.03
```

As can be seen, there are outliers. The minimum riding time is negative, which is unusual as time can't be negative. The maximum also seems too large (that is, the rider has taken the bike for approximately 37 days). To confirm that this is an outlier, let's check each quantile value.

Printing the values in each quantiles with 5% difference

```
quantiles <- quantile(df$riding_time, seq(0,1,by=0.05))
quantiles
```

```
##            0%            5%           10%           15%           20%
## -29049.966667      3.200000      4.633333      5.816667      6.916667
##           25%           30%           35%           40%           45%
##      8.033333      9.183333     10.416667     11.700000     13.133333
##           50%           55%           60%           65%           70%
##     14.700000     16.483333     18.516667     20.883333     23.616667
##           75%           80%           85%           90%           95%
##     26.866667     30.816667     36.666667     46.450000     73.783333
##          100%
##  58720.033333
```

It is clear that the maximum value was an outlier and hence it is unworthy of consideration.

Considering only the values in the 5-95% interval

```
new_df_without_outliers <- df %>%
  filter(riding_time > as.numeric(quantiles['5%'])) %>%
  filter(riding_time < as.numeric(quantiles['95%']))
final_df <- new_df_without_outliers
```

Now let's compare the riding_time with all the other features used before

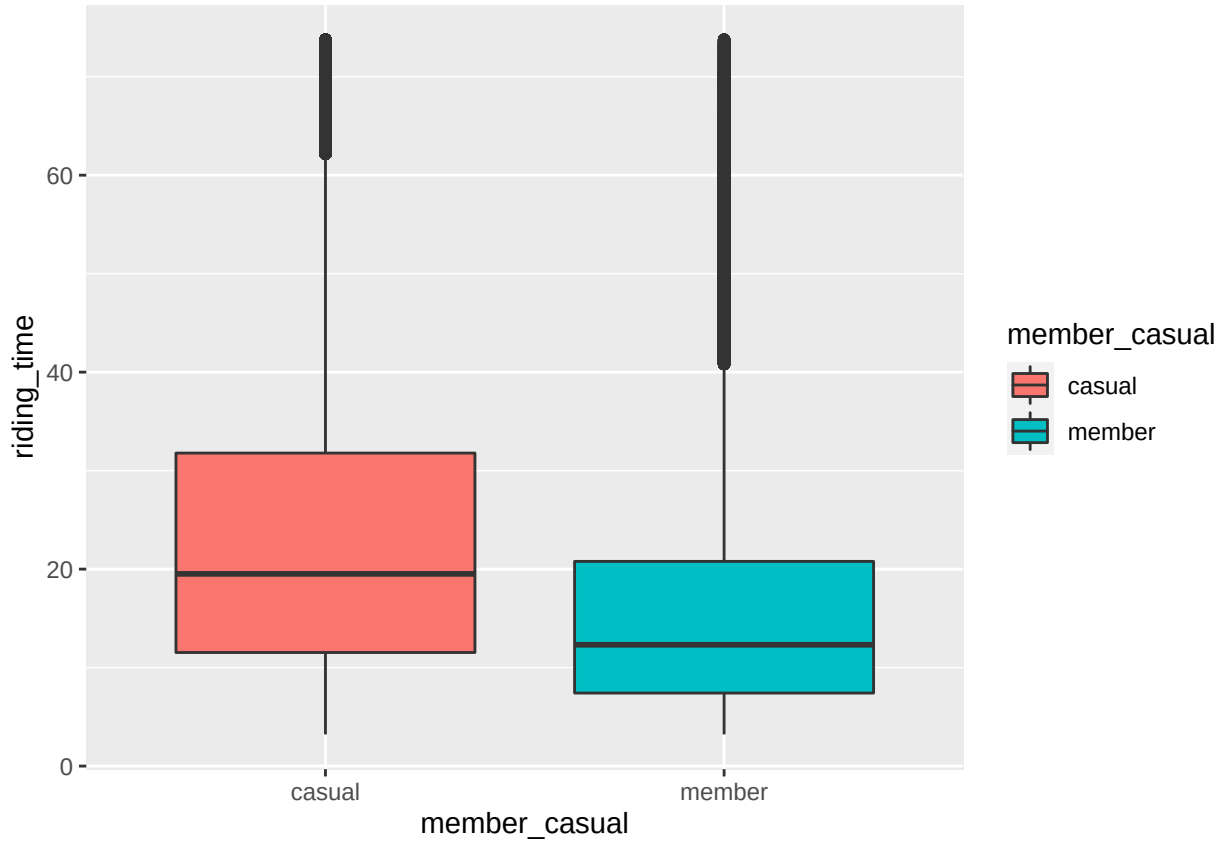
g) Let's start by checking the riding time of both members and casual riders

```
final_df %>%
  group_by(member_casual) %>%
  summarize(mean=mean(riding_time),
            first_quarter=quantile(riding_time, 0.25),
            median=median(riding_time),
            third_quarter=quantile(riding_time, 0.75),
            IQR = third_quarter-first_quarter)
```

```
## # A tibble: 2 x 6
##   member_casual  mean first_quarter median third_quarter   IQR
##   <chr>         <dbl>         <dbl>  <dbl>         <dbl> <dbl>
## 1 casual         24.0         11.5    19.5          31.8  20.2
## 2 member         15.6          7.42   12.3          20.8  13.4
```

Let's plot the same and check for any trends.

```
ggplot(final_df, aes(x=member_casual, y=riding_time, fill=member_casual))+
  geom_boxplot()
```



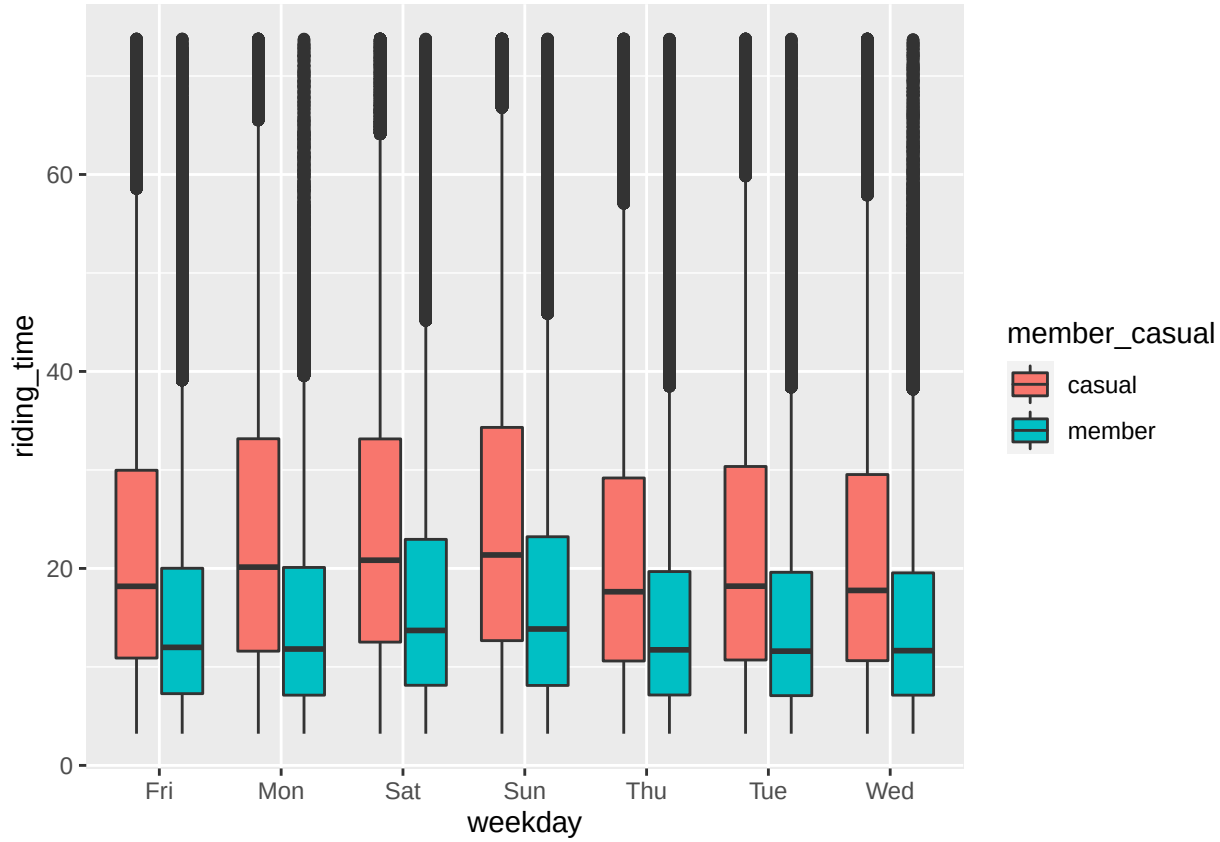
h) Let's next check riding time of both members and casual riders for each of the week Since the riding time is continuous and any feature compared to it would be discrete, we can go with box plots.

```
final_df %>%
  group_by(weekday) %>%
  summarize(mean=mean(riding_time),
            first_quarter=quantile(riding_time, 0.25),
            median=median(riding_time),
            third_quarter=quantile(riding_time, 0.75),
            IQR = third_quarter-first_quarter)
```

```
## # A tibble: 7 x 6
##   weekday  mean first_quarter median third_quarter   IQR
##   <chr>   <dbl>         <dbl>  <dbl>         <dbl> <dbl>
## 1 Fri      18.0          8.3    14.0          23.8  15.4
## 2 Mon      18.7          8.27   14.3          25.0  16.7
## 3 Sat      20.8          9.78   16.7          27.8  18.0
## 4 Sun      21.2          9.87   17.1          28.5  18.6
## 5 Thu      17.4          8.02   13.4          22.9  14.8
## 6 Tue      17.5          7.93   13.3          23.1  15.2
## 7 Wed      17.3          7.95   13.3          22.8  14.8
```

Let's plot the same and check for any trends.

```
ggplot(final_df, aes(x=weekday, y=riding_time, fill=member_casual))+
  geom_boxplot()
```



It can be clearly seen that the casual riders spend more time riding than annual members. Let's see why this is the case in the next steps.

i) Let's now check how these times vary for each month

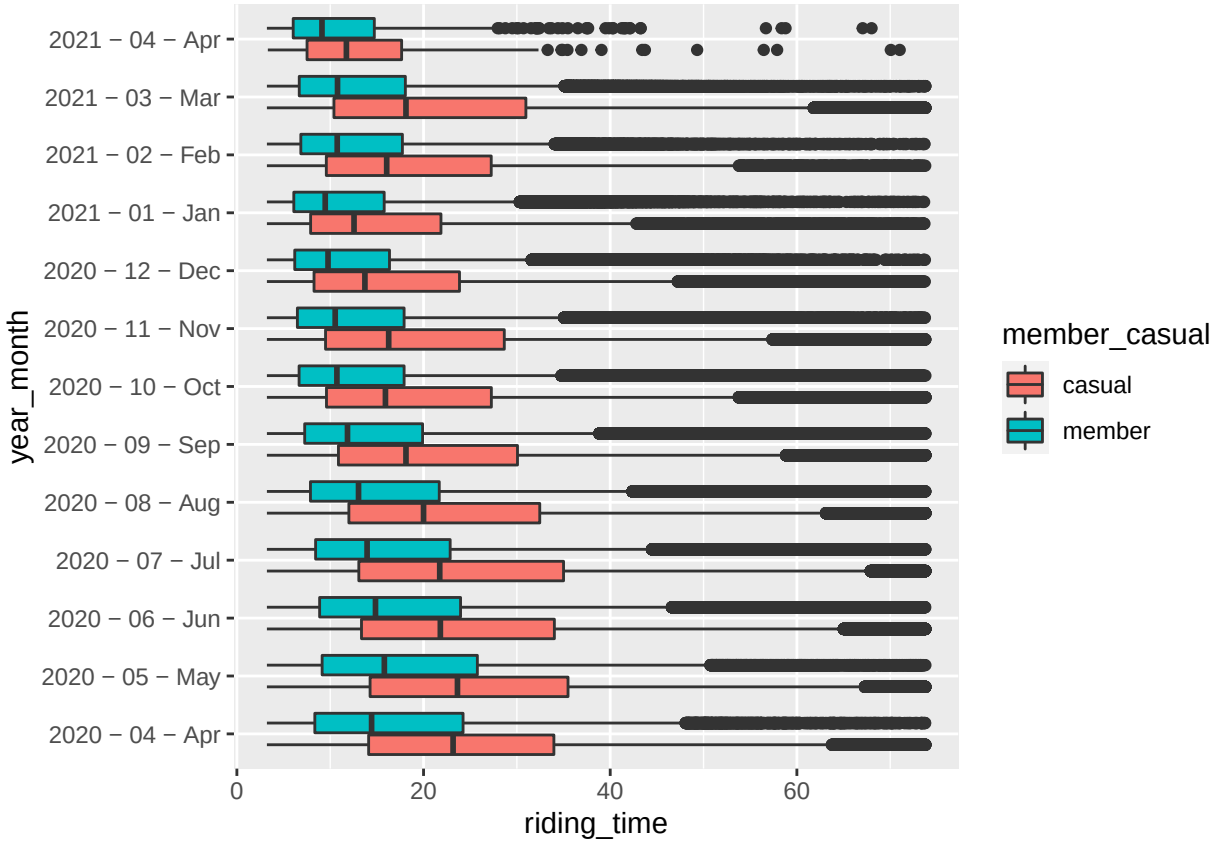
```
final_df %>%
  group_by(year_month) %>%
  summarize(mean=mean(riding_time),
            first_quarter=quantile(riding_time, 0.25),
            median=median(riding_time),
            third_quarter=quantile(riding_time, 0.75),
            IQR = third_quarter-first_quarter)
```

```
## # A tibble: 13 x 6
##    year_month       mean first_quarter median third_quarter   IQR
##    <chr>           <dbl>         <dbl>  <dbl>         <dbl> <dbl>
##  1 2020 - 04 - Apr  19.9          9.4   16.5           27.1 17.7
##  2 2020 - 05 - May  22.0         10.8   18.7           29.5 18.7
##  3 2020 - 06 - Jun  21.2         10.4   17.5           28.0 17.6
##  4 2020 - 07 - Jul  21.2         10.1   17.0           28.0 18.0
##  5 2020 - 08 - Aug  19.9          9.33  15.7           26.5 17.1
##  6 2020 - 09 - Sep  18.3          8.4   14.2           24.2 15.7
##  7 2020 - 10 - Oct  16.3          7.43  12.3           21.1 13.6
```

```
##  8 2020 - 11 - Nov  16.3          7.18  12.0          21.1 13.9
##  9 2020 - 12 - Dec  14.0          6.53  10.5          17.8 11.3
## 10 2021 - 01 - Jan  13.3          6.32   9.94         16.7 10.4
## 11 2021 - 02 - Feb  15.4          7.2   11.5          19.6 12.4
## 12 2021 - 03 - Mar  17.2          7.58  12.8          22.6 15.0
## 13 2021 - 04 - Apr  12.6          6.27   9.65         15.6  9.37
```

Let's plot the same and check for any trends.

```
ggplot(final_df, aes(x=year_month, y=riding_time, fill=member_casual))+
  geom_boxplot()+
  coord_flip()
```



As the number of riders in the winter months was less, the same reflects in the riding time.

Observations

Annual Members vs Casual Riders

According to the above analysis, let's see how members and casual riders differ:

1) The population of the annual members is more than the casual riders, with 58% of the total riders in the last 12 months.

2) The percentage of riders that own a bike is highest in July, August, and September. We can assume this rise due to the season (Summer to Fall transition)

3) We can also observe a trend with a similar reason (due to the season of the year) that is the number of bikes owned is few in the peak winter season, that is, the months of November, December, January, and February.

4) It is decisive that across all the months, the members were more in percent than casuals.

5) As an average in 12 months, annual members seem to start their journey from early morning 6 am and increase throughout the day to hit the peak at 5 pm. This trend might be because most of the members use their bikes to commute to their work. As the typical corporate day ends around 5 pm, there is a peak at that hour.

6) Also, as the day progresses, the casual riders start their journey for maybe recreational activities.

7) If we scrutinize the start hour per day of the week, we find that the annual members are not as active on the weekends as they are on the weekdays. In contrast, casual riders are more active on the weekends. This trend proves that members usually use their bikes to commute to work.

8) When later the hours of the day were classified into morning, afternoon, and evening, the visualization depicted that more members travel in the mornings and afternoons. In comparison, casual riders travel more in the afternoons and evenings.

9) When the riding time of casuals and members is compared, causal riders have higher riding time than members. This trend again proves that members use bikes to work and park, reducing their riding time.

10) Another proof that members have a fixed route and use bikes for the same reason throughout the weekdays is when we plot the riding time against each day.

11) The members' box in the boxplot remains almost constant for all the weekdays and slightly increases on the weekends. This trend could be maybe they use their bikes for recreational purposes.

12) Also, as the number of riders was less in the peak winter times, the same reflects on the riding time. There were fewer riders in these months, so was the riding time.

Changes required to convert casual riders to annual members:

1) Impose offers for annual members and not for casual riders.

2) Increase the price of the bikes on weekends for casual riders.

3) Place special offers for anyone who registers for the annual membership from November to February.

4) Reduce the limit on the time duration or distance a casual rider can travel.

5)Increase the surge for bikes for casual riders in the evenings. Annual members riding the bikes in the mid-night can avail themselves of free cafe or bar coupons.