CASE STUDY

Cyclistic Bike-share

Analysis and Recommendations

Maximize the number of annual memberships for future growth

HUONG DAO QUYNH
03/2025

Table of Contents

01

Problem context

Overview of the company and the business task

Company overview

About	Cyclistic offers bike-share service with a fleet of 5824 bicycles and a network of 692 stations across Chicago.
Customer	• Broad segment • Cyclistic users are more likely to ride for **leisure**, but about 30% use the bikes to commute to **work** each day
Pricing plans	• **Casual riders**: Single-ride, Full-day • **Members**: Annual membership (more profitable)
Goal	Maximize the number of annual memberships for future growth
Business Task	Identify **key differences** between casual riders and annual members to tailor marketing strategies to **convert casual riders to annual members.**

02

Data Preparation

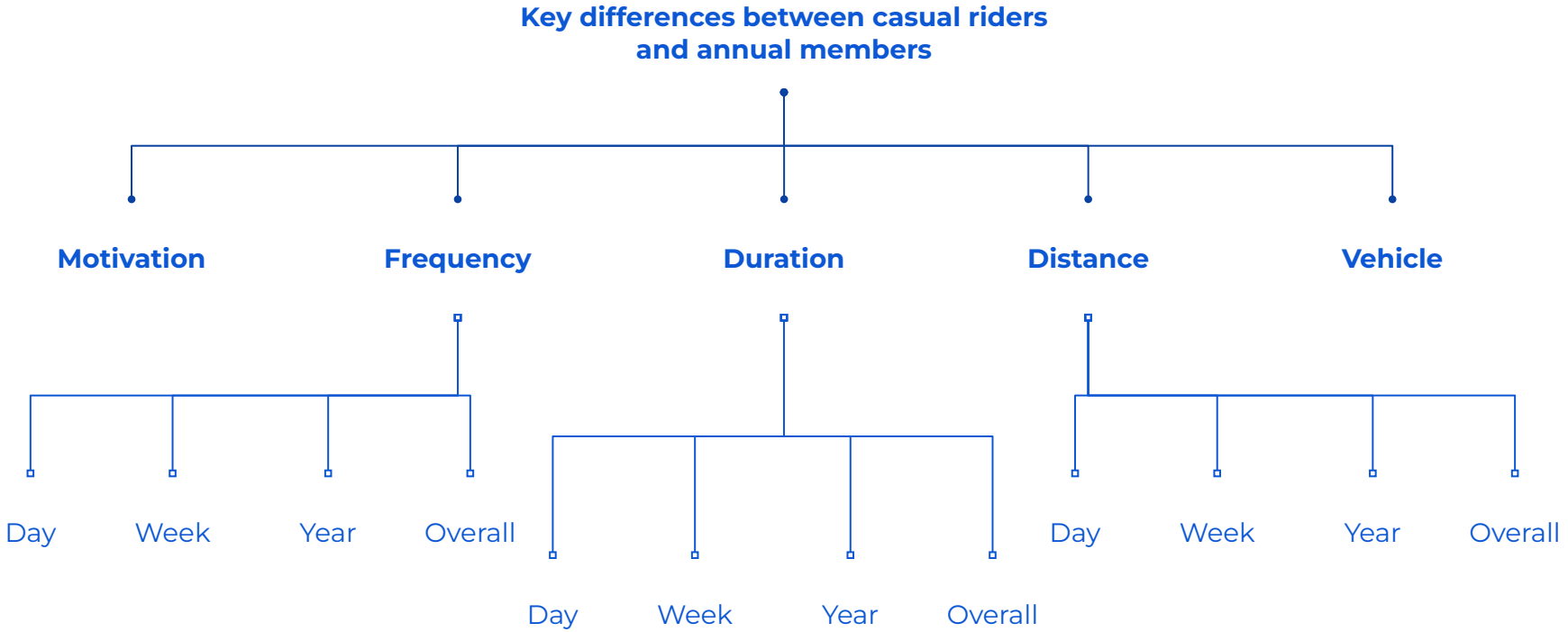
Data collection, cleaning, and structuring.

2.1. Data collection and validation

Data	06/2023 - 05/2024 Divvy Trip Data - The previous 12 months of Cyclistic data. The data before 06/2023 is not used.
Data provider	Cyclistic is a fictional company. The data has been made available byMotivate International Inc under this license.
Information	There are 12 csv files containing 13 variables and a total of nearly 6 million observations
Data Variables	• Ride identifier: ride_id • Vehicle type: rideable_type • Time: started_at, ended_at • Location: start_station_name, start_station_id, end_station_name, end_station_id, start_lat, start_lng, end_lat, end_lng • Membership type: number_casual

From the business task and the available data

Top-down, MECE Issue Tree



2.2. Data cleaning

Step 1

- Identify necessary data variables based on issue tree (slide 5).
- There are a total of 18 variables.

Step 2

- Define variable descriptions (slide 18) including name, description, type, format.
- Clean each data file using **Microsoft Excel.**
- Convert the excel files to csv files for better data manipulation.

Step 3

- Using **R** to join 12 cleaned csv data files vertically to form a dataset containing 5.6 million observations. (code in slide 23)
- Save the dataset into a csv file named “divvydata”.
- Load the dataset “divvydata” into **Power BI** for data visualization.

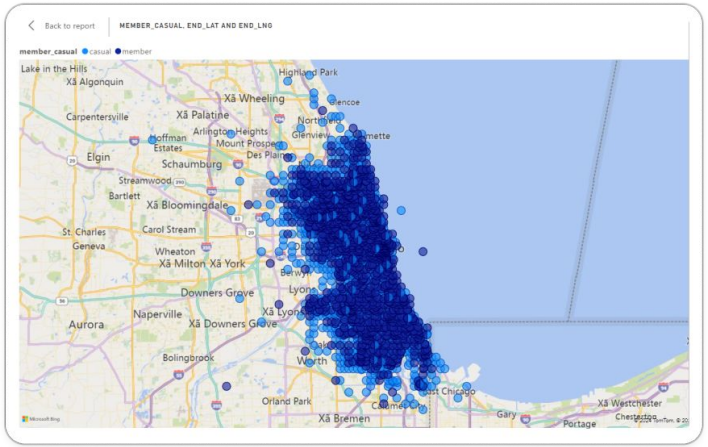
03

Data Analysis and Recommendations

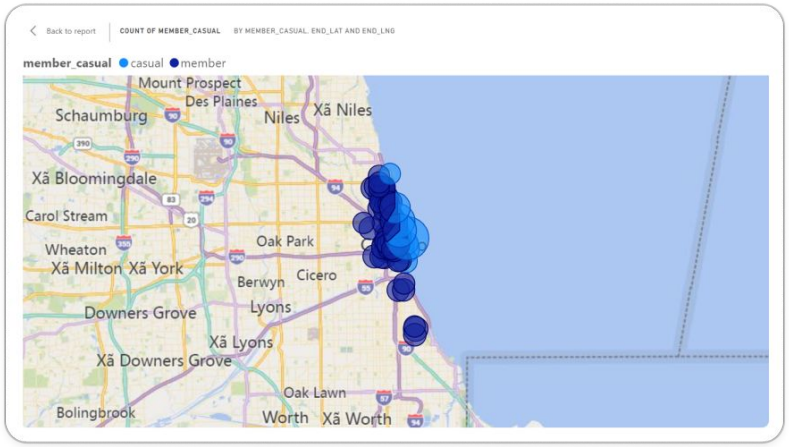
Data visualization, insights, and recommendations

3.1. Motivation differences

casual member



The distribution of all end locations



The distribution of end location that occur > 10000 times

Analysis

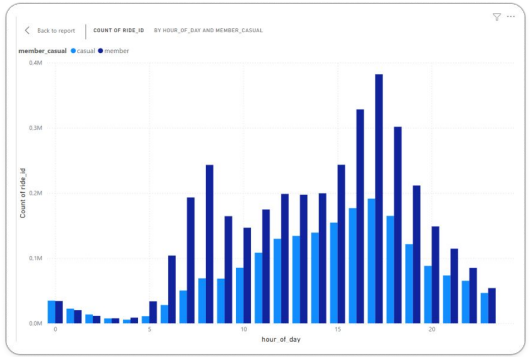
- Annual members visit schools, universities, station, parks, and residential areas most frequently.
- Casual riders visit parks, trails, and lakes the most.

Suggestions

- **Targeted Promotion:**
  - Advertise membership benefits at high-traffic annual member locations (schools, universities, stations).
  - Place temporary membership promo banners at parks, trails, and lakes to capture casual riders.
- **Geo-Targeted Ads:**
  - Push notifications for membership when a casual rider starts near a station or university.

3.2. Frequency differences

casual member

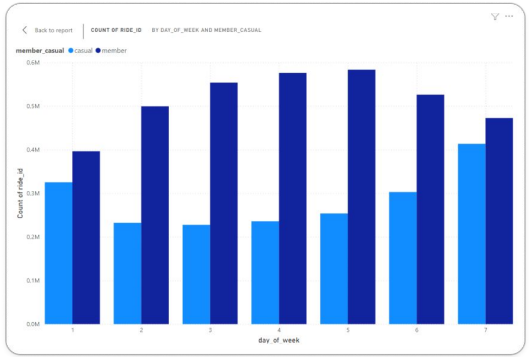


Count of ride started by hour of day

Analysis:

- Member start the most rides in the early morning (7:00 - 8:00) and in the late afternoon (16:00-17:00).
- Casual riders ride the most in the afternoon (16:00-17:00).

Suggestion: Timing-Based Messaging - Special promotions for annual members during 7:00 - 8:00 (convert casual user with similar usage pattern to annual) and 16:00-17:00 (create exclusive benefits for annual).



Count of ride started by day of week

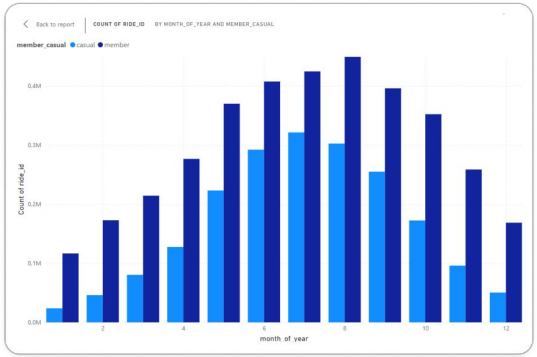
Analysis:

- Member start more ride during weekdays
- Casual riders start more in weekends.

Suggestion: Timing-Based Messaging - Weekday campaigns to highlight benefits for weekday commutes (convert casual user with annual usage pattern).

3.2. Frequency differences

casual member



Count of ride started by month of year

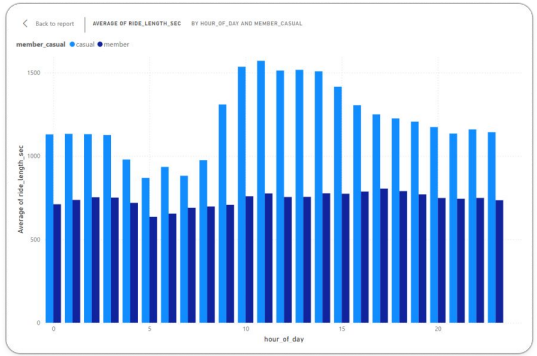
Analysis:

- Annual members and casual rider share similar pattern of reaching the highest number in summer (June - August).

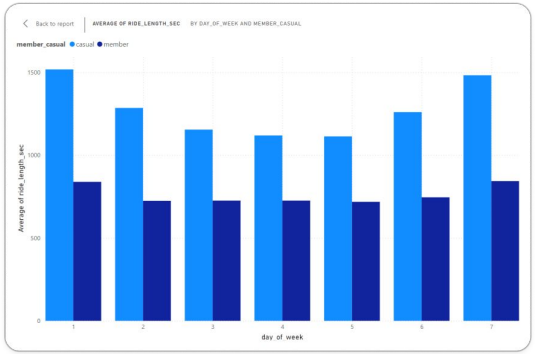
Suggestion: Seasonal Conversions - Launch a summer membership discount for casual riders who already ride frequently in peak season.

3.3. Duration differences

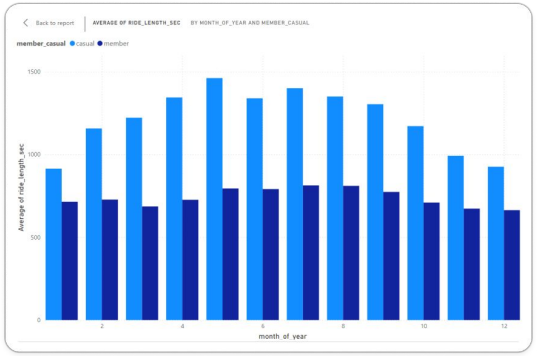
casual member



Mean duration by hour of day



Mean duration by day of week



Mean duration by month of year

Analysis:

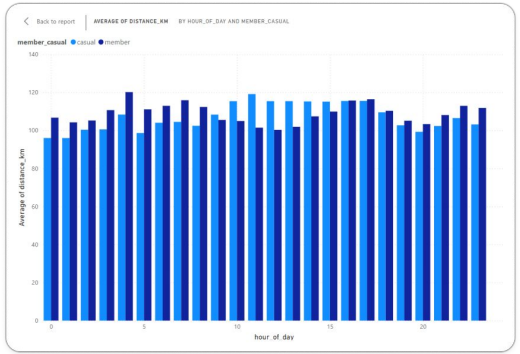
- Annual riders ride the similar duration around the day, the week and the year.
- Casual riders ride 1,7 times longer than annual riders, with different pattern.
  - In a day: they ride longest rides in the afternoon (from 10:00-16:00)
  - In a week: they ride longest rides on weekends
  - In a year: they ride longest rides in summer (May - July)

Suggestion:

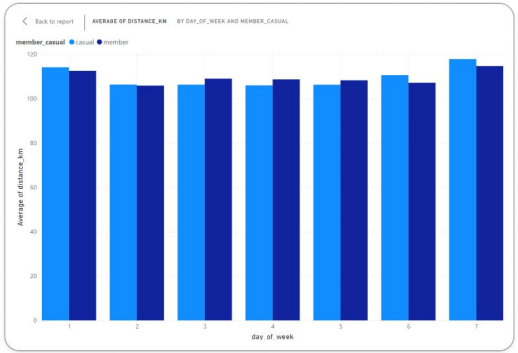
- **Membership for Long-time Rides in appropriate time:** time-based pricing nudge: “Riding more than 30 minutes? Save money with a membership!” in the afternoon, more on weekends, and summer.
- Highlight **those membership benefits for afternoon, weekends, and summer trips**, since casual riders prefer these times.

3.4. Distance differences

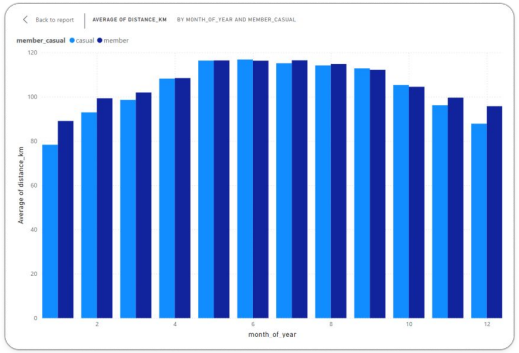
casual member



Mean distance by hour of day



Mean distance by day of week



Mean distance by month of year

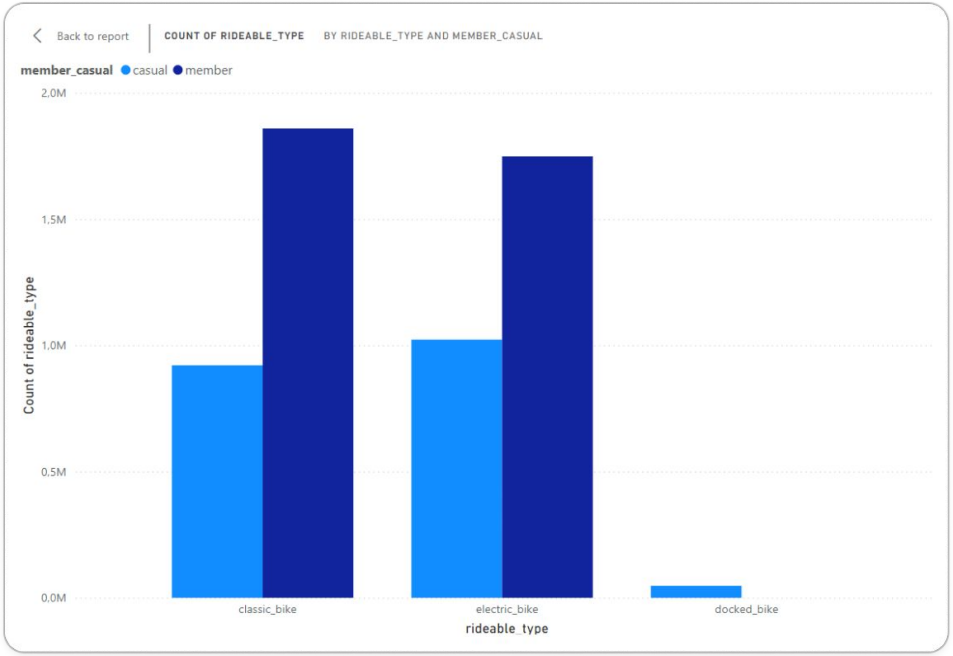
Analysis: Casual riders and annual members have similar riding distance throughout the day, the week and the year.

Suggestion:

- Members and casual riders are **charged differently on distance** (lower charge for annual members)
- The marketing strategy should emphasize the **similarity in distance**, and how the annual membership bring better benefits in the long run.

3.5. Bike types differences

casual member



Count of different bike types used by member and casual

Analysis:

- Annual member prefer classic bikes.
- Casual riders prefer electric bikes. Only casual riders use docked bikes.

Suggestion:

- **E-Bike Incentives for Membership:** Introduce a "Member-Only E-Bike Discount" to entice casual riders who prefer e-bikes
- **Docked Bike Promotions:** Create a dock-based rewards program for casual riders who use docked bikes.

3.6. Assumption: Residential differences

Assumption: Consider the using patterns above, I make an assumption that:

- Casual riders are **tourists**
- Members are **local residents**

Suggestion:

1. **Convert Tourists (Casual Riders) into Short-Term Members**
    - **Tourist Passes:** Offer 3-day, weekly, or monthly tourist memberships for visitors who stay longer.
    - **Hotel & Travel Partnerships:** Promote short-term passes through hotels, Airbnb hosts, and travel agencies.
    - **City Exploration Discounts:** Bundle bike access with museum, park, or attraction entry.
2. **Strengthen Local Resident (Annual Member) Benefits**
    - **Commuter Incentives:** Highlight cost savings over public transport or driving for daily commutes.
    - **Employer Partnerships:** Offer company-sponsored annual memberships as a commuter benefit.

04

Appendix

Additional informations: assumptions, data description, data cleaning records, code for joining data

4.1. Assumptions

- ride_id is a string of numbers and letters that has len equal 16.
- Rides that has ride_length under 60 seconds are errors.
  - Human error: sudden change in intention, order a ride by mistake, etc.
  - Vehicle error: the bike is broken, monthly vehicle check, etc.
  - System error
- Rides that has latitude and longitude values equal NULL are errors.
  - Human error: thief, broken
  - Vehicle error: bike taken for repair
  - System error: do not record bike return
- Rides that has distance_km results in #NUM! in the spreadsheet are rides that has distance_km approaching 0. Replace the all #NUM! with 0.

4.2. Variables descriptions

variable	description	type	format	additional info
ride_id	id of a ride	string	random string of numbers and letters	len =16
rideable_type	type of bike used	string	categories: classic_bike, electric_bike, docked_bike	
started_at	the time and date the ride started	datetime	dd/mm/yyyy hh:mm:ss	
ended_at	the time and date the ride ended	datetime	dd/mm/yyyy hh:mm:ss	must happen 60 seconds after the start time
start_station_name	the name of the station the ride started	string		
start_station_id	the id of the station the ride started	string		

4.2. Variables descriptions

variable	description	type	format	additional info
end_station_name	the id of the station the ride ended	string		
end_station_id	the id of the station the ride ended	string		
start_lat	the latitude of the station the ride started	float	round to 5 decimals	
start_lng	the longitude of the station the ride started	float	round to 5 decimals	
end_lat	the latitude of the station the ride ended	float	round to 5 decimals	
end_lng	the longitude of the station the ride ended	float	round to 5 decimals	
number_casual	type of membership of the rider	string	categories: casual, member	

4.2. Variables descriptions

variable	description	type	format	additional info
ride_length_sec	the length of the ride in seconds	float		assume that rides that less than 60 seconds are errors
distance_km	the distance between two points on the map by kilometers	float	round to 5 decimals	
hour_of_day	the hour of the day that a ride start	int		
day_of_week	the day of the week that a ride start	int		
month_of_year	the month of the year that a ride start	int		

4.3. Data cleaning records in Excel

#	description
1	Format all ride_id to text. Use LEN() to check if they are all 16.
2	Check if rideable_type contains only classic_bike and electric_bike. Use filter.
3	Format started_at and ended_at as “dd/mm/yyyy hh:mm:ss”
4	Add a column named “ride_length_sec” to calculate the trip duration. Subtract ended_at and started_at, multiply the results by 86400 to get the ride length in seconds.
5	Move all observations that has ride_length lower than 60 seconds to another sheet named tripdata_errors.
6	Convert all latitude and longitude values to number using VALUE(). Using 5 decimals.

4.3. Data cleaning records in Excel

#	description
7	Delete all observations that has NULL values in latitude and longitude columns.
8	Add a column named “distance_km”. Use the latitude and longitude, and the formula “=acos(sin(lat1)*sin(lat2)+cos(lat1)*cos(lat2)*cos(lon2-lon1))*6371 (6371 is Earth radius in km.)” to calculate the diagonal distance between the two points in km. Round to 5 decimals.
9	Check if member_casual only contain member and casual using filter.
10	Add a new column named “hour_of_day”. Use HOUR()
11	Add a new column named “day_of_week”. Use WEEKDAY(), return type 1.
12	Add a column named “month_of_year”. Use MONTH()

4.4. Code for joining data in R

```
install.packages('tidyverse')
library(tidyverse)

#=====================
# 1.COLLECT DATA
#=====================
# Upload cleaned divvy data sets (csv files) here

divvydata_202405 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202405-divvy-tripdata.csv")
divvydata_202404 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202404-divvy-tripdata.csv")
divvydata_202403 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202403-divvy-tripdata.csv")
divvydata_202402 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202402-divvy-tripdata.csv")
divvydata_202401 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202401-divvy-tripdata.csv")
divvydata_202312 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202312-divvy-tripdata.csv")
divvydata_202311 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202311-divvy-tripdata.csv")
divvydata_202310 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202310-divvy-tripdata.csv")
divvydata_202309 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202309-divvy-tripdata.csv")
divvydata_202308 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202308-divvy-tripdata.csv")
divvydata_202307 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202307-divvy-tripdata.csv")
divvydata_202306 <- read_csv("…/Capstone Project/data/xlsx/cleaned csv/202306-divvy-tripdata.csv")
```

4.4. Code for joining data in R

```
#=====================
# 2. MERGE DATA
#=====================
# As we've already clean the data and rename the columns in the cleaning steps. We can merge the data right away.

divvydata <- bind_rows(divvydata_202306, divvydata_202307, divvydata_202308, divvydata_202309, divvydata_202310,
divvydata_202311, divvydata_202312, divvydata_202401, divvydata_202402, divvydata_202403, divvydata_202404, divvydata_202405)

# The started_at and ended_at column appear as string because they were not originally presented by the standard datetime
format of R, which is yyyy-mm-dd

#=====================
# 3. CHECK THE DATA FOR ANALYSIS
#=====================
str(divvydata)
summary(divvydata)

# Check the options for each columns
table(divvydata$rideable_type)
table(divvydata$member_casual)
table(divvydata$day_of_week)
table(divvydata$hour_of_day)
table(divvydata$month_of_year)

# export the csv file of the joined data
write.csv(divvydata,"D:/University/COURSES/Google Data Analyst Professional Certificate/Capstone Project/data/xlsx/cleaned
csv/divvydata.csv", row.names = FALSE)
```

Thanks

If you have any questions regarding this project,

please contact me (Dao Quynh Huong) through:

- dqhuong192@gmail.com
- (+84) 366 275 295
- linkedin.com/dqhuong192