Introduction

This case study, which is based on the analysis of a dataset from a fictional bike-sharing company called Cyclistic, is the capstone project of my Google Data Analytics Professional Certificate course.

In this project, I will be using the **R programming** language to conduct my analysis, complemented by the **RStudio IDE** for its robust ability to integrate effective statistical analysis tools with compelling data visualizations.

In order to answer the business questions of this case study, I will be following the data analysis process steps:

- **Ask**
- **Prepare**
- **Process**
- **Analyze**
- **Share**
- **Act**

In each of these steps, the analysis will be broken down into:

- **Key tasks**;
- A **deliverable**;
- The presentation and explanation of the **code** used, when applicable.

Scenario

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

Ask

Three questions will guide the future marketing program of the company:

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?**

KEY TASKS:

- **Identify the business task**:

Understand how casual riders and annual members use Cyclistic bikes differently.

- **The key stakeholders**:

Cyclistic Executive Team: Decision-makers who will approve the recommended marketing program.

Lily Moreno (Director of Marketing): Responsible for the development and implementation of marketing campaigns.

Cyclistic Marketing Analytics Team: Analysts who contribute to data collection, analysis, and reporting.

Cyclistic Users (Casual Riders and Annual Members): End-users whose behavior is under analysis for strategic marketing decisions.

DELIVERABLE:

- **A clear statement of the business task**:

Understand how casual riders and annual members use Cyclistic bikes differently, in order to design a new marketing strategy to drive future business success by maximizing the number of annual memberships, based on the analysis of the data.

Prepare

In this second step, I will prepare the data by establishing where is it located, how is it organized, if there are any problems with the data or issues with bias or credibility. I will also be addressing the issues of licensing, privacy and security of the data, and verify the data's integrity and credibility.

Finally, I will determine how can this data help answer the business questions.

I will use Cyclistic's own historical trip data to analyze and identify trends. This is public data, which has been made available by *Motivate International Inc.* under this license. The respective dataset is available here.

KEY TASKS:

- **Download data and store it appropriately**:

The data has been downloaded from this database, stored on my hard drive and the respective copies organized in this case study database folder, and made available on my RPubs and Kaggle profiles.

- **Identify how it's organized**:

The data is in comma-delimeted (.CSV) format. It is composed of four files, the "*Divvy_Trips_2019_Q2*" with 12 columns and 1.048.575 rows; the "*Divvy_Trips_2019_Q3*" with also 12 columns and 1.048.575 rows; the "*Divvy_Trips_2019_Q4*" with 12 columns and 704.054 rows; and the "*Divvy_Trips_2020_Q1*" with 13 columns and 426.887 rows.

- **Sort and filter the data**:

For this analysis I will work with the data from the second, third and fourth quarters of 2019 plus the 1st quarter of 2020, completing a 12-month period of data under this analysis.

- **Determine the credibility of the data**:

The credibility of the data can be assessed using the **ROCCC system**, which shows that:

- The data is **reliable**, since it has a large sample size, representative of the population in the analysis;
- The data is **original**, since we can identify its primary source;
- The data is **comprehensive**, since it's understandable and it is not flawed with missing critical information necessary to answer the business task and find the solution;
- The data is **current**, since it is up to date and relevant for this analysis;
- The data is **cited**, since its source has been well vetted.

The data's credibility and integrity is verifiable and, therefore, the data is adequate to provide reliable and comprehensive insights for this case study.

- **Determine the limitations of the data**:

This data has limitations regarding data privacy issues, since it is not allowed to use riders' personally identifiable information, such as gender and age. This means it won't be possible, in the scope of this case study, to connect pass purchases to credit card numbers and determine the respective identifiers, i.e., if casual riders live in areas with Cyclistic services available or if they have purchased multiple single passes.

DELIVERABLE:

- A description of all data sources used:

The data source is **primary**, provided by the Cyclistic company, and all data provided is available here.

Process

In this section, I will start to process the data for analysis by gathering it, cleaning it and prepping it.

I will conduct tests to confirm the data's integrity and I will also ensure that the data is clean and ready to analyze.

All of the process will be documented so that the results can be reviewed and shared.

KEY TASKS

- **Check the data for errors**.
- **Choose the tools to use**.
- **Transformation of the data so it can be worked with effectively**.
- **Document the cleaning process**.

The following R code chunks, presented in this section, will perform these tasks.

DELIVERABLE

- **Documentation of any cleaning or manipulation of data**

The following R code chunks, presented in this section, will provide this deliverable.

CODE:

Installation and loading of the packages we'll need:

```
## installation and loading of the packages

library(tidyverse)  # helps wrangle data

## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──

## ✓ dplyr     1.1.4     ✓ readr     2.1.4

## ✓ forcats   1.0.0     ✓ stringr   1.5.1

## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1

## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0

## ✓ purrr     1.0.2

## ── Conflicts ──────────────────────────────────────────── tidyverse_confl
icts() ──

## ✕ dplyr::filter() masks stats::filter()

## ✕ dplyr::lag()    masks stats::lag()

## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force
all conflicts to become errors

library(conflicted) # Use the conflicted package to manage conflicts

library (readr)     # readr function to load the csv files

# Set dplyr::filter and dplyr::lag as the default choices

conflict_prefer("filter", "dplyr")

## [conflicted] Will prefer dplyr::filter over any other package.

conflict_prefer("lag", "dplyr")

## [conflicted] Will prefer dplyr::lag over any other package.
```

Collecting and loading the datasets:

```
## uploading and collecting the Divvy datasets (csv files)

q2_2019 <- read_csv("/kaggle/input/my-google-data-analytics-capstone-proje
ct/Divvy_Trips_2019_Q2.csv")

## Warning: One or more parsing issues, call `problems()` on your data fra
me for details,

## e.g.:

##   dat <- vroom(...)

##   problems(dat)

## Rows: 1048575 Columns: 12

## ── Column specification ──────────────────────────────────────────────
─────────

## Delimiter: ","

## chr  (5): 01 - Rental Details Rental ID, 03 - Rental Start Station Name
, 02 ...

## dbl  (5): 01 - Rental Details Bike ID, 01 - Rental Details Duration In
Secon...

## dttm (2): 01 - Rental Details Local Start Time, 01 - Rental Details Loc
al En...

##

## 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 thi
s message.

q3_2019 <- read_csv("Divvy_Trips_2019_Q3.csv")

## Warning: One or more parsing issues, call `problems()` on your data fra
me for details,

## e.g.:

##   dat <- vroom(...)

##   problems(dat)

## Rows: 1048575 Columns: 12
## ── Column specification ──────────────────────────────────────────────────
────────

## Delimiter: ","

## chr  (5): trip_id, from_station_name, to_station_name, usertype, gender

## dbl  (5): bikeid, tripduration, from_station_id, to_station_id, birthye
ar

## dttm (2): start_time, end_time

##

## 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 thi
s message.

q4_2019 <- read_csv("/kaggle/input/my-google-data-analytics-capstone-proje
ct/Divvy_Trips_2019_Q4.csv")

## Warning: One or more parsing issues, call `problems()` on your data fra
me for details,

## e.g.:

##   dat <- vroom(...)

##   problems(dat)

## Rows: 704054 Columns: 12

## ── Column specification ──────────────────────────────────────────────────
────────

## Delimiter: ","

## chr  (5): trip_id, from_station_name, to_station_name, usertype, gender

## dbl  (5): bikeid, tripduration, from_station_id, to_station_id, birthye
ar

## dttm (2): start_time, end_time

##

## 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 thi
s message.

q1_2020 <- read_csv("/kaggle/input/my-google-data-analytics-capstone-proje
ct/Divvy_Trips_2020_Q1.csv")

## Rows: 426887 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_l
at, 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 thi
s message.
```

Wrangling the data into a single file:

```
# Compare column names each of the files

# While the names don't have to be in the same order, they DO need to match
perfectly before we can use a command to join them into one file

colnames(q2_2019)

##  [1] "01 - Rental Details Rental ID"

##  [2] "01 - Rental Details Local Start Time"

##  [3] "01 - Rental Details Local End Time"

##  [4] "01 - Rental Details Bike ID"

##  [5] "01 - Rental Details Duration In Seconds Uncapped"

##  [6] "03 - Rental Start Station ID"

##  [7] "03 - Rental Start Station Name"
##  [8] "02 - Rental End Station ID"

##  [9] "02 - Rental End Station Name"

## [10] "User Type"

## [11] "Member Gender"

## [12] "05 - Member Details Member Birthday Year"

colnames(q3_2019)

##  [1] "trip_id"           "start_time"        "end_time"

##  [4] "bikeid"            "tripduration"      "from_station_id"

##  [7] "from_station_name" "to_station_id"     "to_station_name"

## [10] "usertype"          "gender"            "birthyear"

colnames(q4_2019)

##  [1] "trip_id"           "start_time"        "end_time"

##  [4] "bikeid"            "tripduration"      "from_station_id"

##  [7] "from_station_name" "to_station_id"     "to_station_name"

## [10] "usertype"          "gender"            "birthyear"

colnames(q1_2020)

##  [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"
```

Finding: **Column names between the files are not the same nor consistent**. They need to match perfectly in order to be joined into a single file. It is, therefore, **necessary to rename the divergent columns for congruency**, which will be accomplished with the following code chunks:

```
# Rename columns of q2_2019, q3_2019 and q4_2019 to make them consistent wi
th q1_2020

(q2_2019 <- rename(q2_2019
                    ,ride_id = '01 - Rental Details Rental ID'

                    ,rideable_type = '01 - Rental Details Bike ID'

                    ,started_at = '01 - Rental Details Local Start Time'

                    ,ended_at = '01 - Rental Details Local End Time'

                    ,start_station_name = '03 - Rental Start Station Name'

                    ,start_station_id = '03 - Rental Start Station ID'

                    ,end_station_name = '02 - Rental End Station Name'

                    ,end_station_id = '02 - Rental End Station ID'

                    ,member_casual = 'User Type'))

## # A tibble: 1,048,575 × 12

##    ride_id                 started_at          ended_at            ride
able_type

##    <chr>                   <dttm>              <dttm>
<dbl>

##  1 "22178529"              2019-04-01 00:02:22 2019-04-01 00:09:48
6251

##  2 "22178530,2019-04-01 0… NA                  NA
NA

##  3 "22178531"              2019-04-01 00:11:07 2019-04-01 00:15:19
5649

##  4 "22178532"              2019-04-01 00:13:01 2019-04-01 00:18:58
4151

##  5 "22178533,2019-04-01 0… NA                  NA
NA

##  6 "22178534"              2019-04-01 00:19:39 2019-04-01 00:23:56
3123

##  7 "22178535"              2019-04-01 00:26:33 2019-04-01 00:35:41
6418

##  8 "22178536"              2019-04-01 00:29:48 2019-04-01 00:36:11
4513
```

```
##  9 "22178537,2019-04-01 0… NA                  NA
NA

## 10 "22178538,2019-04-01 0… NA                  NA
NA

## # ℹ 1,048,565 more rows

## # ℹ 8 more variables: `01 - Rental Details Duration In Seconds Uncapped
` <dbl>,

## #   start_station_id <dbl>, start_station_name <chr>, end_station_id <d
bl>,

## #   end_station_name <chr>, member_casual <chr>, `Member Gender` <chr>,

## #   `05 - Member Details Member Birthday Year` <dbl>

(q3_2019 <- rename(q3_2019

                   ,ride_id = trip_id

                   ,rideable_type = bikeid

                   ,started_at = start_time

                   ,ended_at = end_time

                   ,start_station_name = from_station_name

                   ,start_station_id = from_station_id

                   ,end_station_name = to_station_name

                   ,end_station_id = to_station_id

                   ,member_casual = usertype))

## # A tibble: 1,048,575 × 12

##    ride_id    started_at          ended_at            rideable_type tri
pduration

##    <chr>      <dttm>              <dttm>                      <dbl>
<dbl>

##  1 "23479388… NA                  NA                             NA
NA
```

```
##  2 "23479389… NA                  NA                            NA
NA
##  3 "23479390… NA                  NA                            NA
NA
##  4 "23479391… NA                  NA                            NA
NA
##  5 "23479392… NA                  NA                            NA
NA
##  6 "23479393" 2019-07-01 00:02:21 2019-07-01 00:07:31          4941
310
##  7 "23479394… NA                  NA                            NA
NA
##  8 "23479395… NA                  NA                            NA
NA
##  9 "23479396… NA                  NA                            NA
NA
## 10 "23479397… NA                  NA                            NA
NA
## # ℹ 1,048,565 more rows
## # ℹ 7 more variables: start_station_id <dbl>, start_station_name <chr>,
## #   end_station_id <dbl>, end_station_name <chr>, member_casual <chr>,
## #   gender <chr>, birthyear <dbl>
(q4_2019 <- rename(q4_2019
                   ,ride_id = trip_id
                   ,rideable_type = bikeid
                   ,started_at = start_time
                   ,ended_at = end_time
                   ,start_station_name = from_station_name
                   ,start_station_id = from_station_id
                   ,end_station_name = to_station_name

                   ,end_station_id = to_station_id

                   ,member_casual = usertype))

## # A tibble: 704,054 × 12

##    ride_id    started_at          ended_at            rideable_type tri
pduration

##    <chr>      <dttm>              <dttm>                      <dbl>
<dbl>

##  1 "25223640" 2019-10-01 00:01:39 2019-10-01 00:17:20          2215
940

##  2 "25223641" 2019-10-01 00:02:16 2019-10-01 00:06:34          6328
258

##  3 "25223642" 2019-10-01 00:04:32 2019-10-01 00:18:43          3003
850

##  4 "25223643… NA                  NA                            NA
NA

##  5 "25223644… NA                  NA                            NA
NA

##  6 "25223645" 2019-10-01 00:04:38 2019-10-01 00:10:51          1891
373

##  7 "25223646… NA                  NA                            NA
NA

##  8 "25223647… NA                  NA                            NA
NA

##  9 "25223648… NA                  NA                            NA
NA

## 10 "25223649… NA                  NA                            NA
NA

## # ℹ 704,044 more rows

## # ℹ 7 more variables: start_station_id <dbl>, start_station_name <chr>,

## #   end_station_id <dbl>, end_station_name <chr>, member_casual <chr>,
## #   gender <chr>, birthyear <dbl>
```

Let's inspect the dataframes and look for incongruencies

```
str(q2_2019)

## spc_tbl_ [1,048,575 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)

##  $ ride_id                                        : chr [1:1048575] "2
2178529" "22178530,2019-04-01 00:03:02,2019-04-01 00:20:30,6226,\"1,048.0\
",317,Wood St & Taylor St,59,Wabash Ave & Roose"| __truncated__ "22178531"
"22178532" ...

##  $ started_at                                     : POSIXct[1:1048575]
, format: "2019-04-01 00:02:22" NA ...

##  $ ended_at                                       : POSIXct[1:1048575]
, format: "2019-04-01 00:09:48" NA ...

##  $ rideable_type                                  : num [1:1048575] 62
51 NA 5649 4151 NA ...

##  $ 01 - Rental Details Duration In Seconds Uncapped: num [1:1048575] 44
6 NA 252 357 NA 257 548 383 NA NA ...

##  $ start_station_id                               : num [1:1048575] 81
NA 283 26 NA 420 503 260 NA NA ...

##  $ start_station_name                             : chr [1:1048575] "D
aley Center Plaza" NA "LaSalle St & Jackson Blvd" "McClurg Ct & Illinois S
t" ...

##  $ end_station_id                                 : num [1:1048575] 56
NA 174 133 NA 426 500 499 NA NA ...

##  $ end_station_name                               : chr [1:1048575] "D
esplaines St & Kinzie St" NA "Canal St & Madison St" "Kingsbury St & Kinzi
e St" ...

##  $ member_casual                                  : chr [1:1048575] "S
ubscriber" NA "Subscriber" "Subscriber" ...

##  $ Member Gender                                  : chr [1:1048575] "M
ale" NA "Male" "Male" ...

##  $ 05 - Member Details Member Birthday Year       : num [1:1048575] 19
75 NA 1990 1993 NA ...

##  - attr(*, "spec")=
##   .. cols(
##   ..   `01 - Rental Details Rental ID` = col_character(),
##   ..   `01 - Rental Details Local Start Time` = col_datetime(format = "
"),
##   ..   `01 - Rental Details Local End Time` = col_datetime(format = "")
,
##   ..   `01 - Rental Details Bike ID` = col_double(),
##   ..   `01 - Rental Details Duration In Seconds Uncapped` = col_double(
),
##   ..   `03 - Rental Start Station ID` = col_double(),
##   ..   `03 - Rental Start Station Name` = col_character(),
##   ..   `02 - Rental End Station ID` = col_double(),
##   ..   `02 - Rental End Station Name` = col_character(),
##   ..   `User Type` = col_character(),
##   ..   `Member Gender` = col_character(),
##   ..   `05 - Member Details Member Birthday Year` = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
str(q3_2019)
## spc_tbl_ [1,048,575 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:1048575] "23479388,2019-07-01 00:00:27,20
19-07-01 00:20:41,3591,\"1,214.0\",117,Wilton Ave & Belmont Ave,497,Kimbal
l Ave "| __truncated__ "23479389,2019-07-01 00:01:16,2019-07-01 00:18:44,5
353,\"1,048.0\",381,Western Ave & Monroe St,203,Western Ave &"| __truncate
d__ "23479390,2019-07-01 00:01:48,2019-07-01 00:27:42,6180,\"1,554.0\",313
,Lakeview Ave & Fullerton Pkwy,144,Larrabe"| __truncated__ "23479391,2019-
07-01 00:02:07,2019-07-01 00:27:10,5540,\"1,503.0\",313,Lakeview Ave & Ful
lerton Pkwy,144,Larrabe"| __truncated__ ...
##  $ started_at        : POSIXct[1:1048575], format: NA NA ...
##  $ ended_at          : POSIXct[1:1048575], format: NA NA ...
##  $ rideable_type     : num [1:1048575] NA NA NA NA NA ...

##  $ tripduration      : num [1:1048575] NA NA NA NA NA 310 NA NA NA NA .
..

##  $ start_station_id  : num [1:1048575] NA NA NA NA NA 300 NA NA NA NA .
..

##  $ start_station_name: chr [1:1048575] NA NA NA NA ...

##  $ end_station_id    : num [1:1048575] NA NA NA NA NA 232 NA NA NA NA .
..

##  $ end_station_name  : chr [1:1048575] NA NA NA NA ...

##  $ member_casual     : chr [1:1048575] NA NA NA NA ...

##  $ gender            : chr [1:1048575] NA NA NA NA ...

##  $ birthyear         : num [1:1048575] NA NA NA NA NA 1990 NA NA NA NA
...

##  - attr(*, "spec")=

##   .. cols(

##   ..   trip_id = col_character(),

##   ..   start_time = col_datetime(format = ""),

##   ..   end_time = col_datetime(format = ""),

##   ..   bikeid = col_double(),

##   ..   tripduration = col_double(),

##   ..   from_station_id = col_double(),

##   ..   from_station_name = col_character(),

##   ..   to_station_id = col_double(),

##   ..   to_station_name = col_character(),

##   ..   usertype = col_character(),

##   ..   gender = col_character(),

##   ..   birthyear = col_double()
##   .. )

##  - attr(*, "problems")=<externalptr>

str(q4_2019)

## spc_tbl_ [704,054 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)

##  $ ride_id           : chr [1:704054] "25223640" "25223641" "25223642"
"25223643,2019-10-01 00:04:32,2019-10-01 00:43:43,3275,\"2,350.0\",313,Lak
eview Ave & Fullerton Pkwy,290,Kedzie "| __truncated__ ...

##  $ started_at        : POSIXct[1:704054], format: "2019-10-01 00:01:39"
"2019-10-01 00:02:16" ...

##  $ ended_at          : POSIXct[1:704054], format: "2019-10-01 00:17:20"
"2019-10-01 00:06:34" ...

##  $ rideable_type     : num [1:704054] 2215 6328 3003 NA NA ...

##  $ tripduration      : num [1:704054] 940 258 850 NA NA 373 NA NA NA NA
...

##  $ start_station_id  : num [1:704054] 20 19 84 NA NA 156 NA NA NA NA ..
.

##  $ start_station_name: chr [1:704054] "Sheffield Ave & Kingsbury St" "T
hroop (Loomis) St & Taylor St" "Milwaukee Ave & Grand Ave" NA ...

##  $ end_station_id    : num [1:704054] 309 241 199 NA NA 226 NA NA NA NA
...

##  $ end_station_name  : chr [1:704054] "Leavitt St & Armitage Ave" "Morg
an St & Polk St" "Wabash Ave & Grand Ave" NA ...

##  $ member_casual     : chr [1:704054] "Subscriber" "Subscriber" "Subscr
iber" NA ...

##  $ gender            : chr [1:704054] "Male" "Male" "Female" NA ...

##  $ birthyear         : num [1:704054] 1987 1998 1991 NA NA ...

##  - attr(*, "spec")=

##   .. cols(

##   ..   trip_id = col_character(),

##   ..   start_time = col_datetime(format = ""),
##   ..   end_time = col_datetime(format = ""),

##   ..   bikeid = col_double(),

##   ..   tripduration = col_double(),

##   ..   from_station_id = col_double(),

##   ..   from_station_name = col_character(),

##   ..   to_station_id = col_double(),

##   ..   to_station_name = col_character(),

##   ..   usertype = col_character(),

##   ..   gender = col_character(),

##   ..   birthyear = col_double()

##   .. )

##  - attr(*, "problems")=<externalptr>

str(q1_2020)

## spc_tbl_ [426,887 × 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)

##  $ ride_id           : chr [1:426887] "EACB19130B0CDA4A" "8FED874C809DC
021" "789F3C21E472CA96" "C9A388DAC6ABF313" ...

##  $ rideable_type     : chr [1:426887] "docked_bike" "docked_bike" "dock
ed_bike" "docked_bike" ...

##  $ started_at        : POSIXct[1:426887], format: "2020-01-21 20:06:59"
"2020-01-30 14:22:39" ...

##  $ ended_at          : POSIXct[1:426887], format: "2020-01-21 20:14:30"
"2020-01-30 14:26:22" ...

##  $ start_station_name: chr [1:426887] "Western Ave & Leland Ave" "Clark
St & Montrose Ave" "Broadway & Belmont Ave" "Clark St & Randolph St" ...

##  $ start_station_id  : num [1:426887] 239 234 296 51 66 212 96 96 212 3
8 ...

##  $ end_station_name  : chr [1:426887] "Clark St & Leland Ave" "Southpor
t Ave & Irving Park Rd" "Wilton Ave & Belmont Ave" "Fairbanks Ct & Grand A
ve" ...
```

```
##  $ end_station_id    : num [1:426887] 326 318 117 24 212 96 212 212 96
100 ...

##  $ start_lat         : num [1:426887] 42 42 41.9 41.9 41.9 ...

##  $ start_lng         : num [1:426887] -87.7 -87.7 -87.6 -87.6 -87.6 ...

##  $ end_lat           : num [1:426887] 42 42 41.9 41.9 41.9 ...

##  $ end_lng           : num [1:426887] -87.7 -87.7 -87.7 -87.6 -87.6 ...

##  $ member_casual     : chr [1:426887] "member" "member" "member" "membe
r" ...

##  - 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_double(),

##   ..   end_station_name = col_character(),

##   ..   end_station_id = col_double(),

##   ..   start_lat = col_double(),

##   ..   start_lng = col_double(),

##   ..   end_lat = col_double(),

##   ..   end_lng = col_double(),

##   ..   member_casual = col_character()

##   .. )

##  - attr(*, "problems")=<externalptr>
```

Finding: The column values *ride_id* and *rideable_type* **need to be converted to character** so that they can **stack correctly**, which will be accomplished with the following code chunk:

```
# Converting ride_id and rideable_type to character

q2_2019 <- mutate(q2_2019, ride_id = as.character(ride_id)

                   ,rideable_type = as.character(rideable_type))



q3_2019 <- mutate(q3_2019, ride_id = as.character(ride_id)

                  ,rideable_type = as.character(rideable_type))



q4_2019 <- mutate(q4_2019, ride_id = as.character(ride_id)

                  ,rideable_type = as.character(rideable_type))
```

Now, we are going to stack all individual quarter's data frames into one big data frame:

```
all_rides <- bind_rows(q2_2019, q3_2019, q4_2019, q1_2020)
```

Finding: *Latitude values*, *longitude values*, *birthyear*, and *gender* fields **need to be removed from the dataset**, since they were dropped in the beginning of 2020. The following code chunk performs this action:

```
# Remove extra columns that were dropped in the most recent data of 2020

all_rides <- all_rides %>%

  select(-c('01 - Rental Details Duration In Seconds Uncapped','05 - Membe
r Details Member Birthday Year', 'Member Gender','05 - Member Details Memb
er Birthday Year', tripduration, start_lat, start_lng, end_lat, end_lng, b
irthyear, gender))
```

Now, let's start the process of cleaning up and adding data to prepare for analysis. - Let´s start by inspecting the new table we created:

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

## [1] "ride_id"            "started_at"         "ended_at"

## [4] "rideable_type"      "start_station_id"   "start_station_name"

## [7] "end_station_id"     "end_station_name"   "member_casual"

nrow(all_rides)       #How many rows are in data frame?

## [1] 3228091

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

```
## [1] 3228091       9

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

## # A tibble: 6 × 9

##   ride_id started_at          ended_at            rideable_type start_s
tation_id

##   <chr>   <dttm>              <dttm>              <chr>
<dbl>

## 1 "22178… 2019-04-01 00:02:22 2019-04-01 00:09:48 6251
81

## 2 "22178… NA                  NA                  <NA>
NA

## 3 "22178… 2019-04-01 00:11:07 2019-04-01 00:15:19 5649
283

## 4 "22178… 2019-04-01 00:13:01 2019-04-01 00:18:58 4151
26

## 5 "22178… NA                  NA                  <NA>
NA

## 6 "22178… 2019-04-01 00:19:39 2019-04-01 00:23:56 3123
420

## # ℹ 4 more variables: start_station_name <chr>, end_station_id <dbl>,

## #   end_station_name <chr>, member_casual <chr>

str(all_rides)        #See list of columns and data types

## tibble [3,228,091 × 9] (S3: tbl_df/tbl/data.frame)

##  $ ride_id           : chr [1:3228091] "22178529" "22178530,2019-04-01
00:03:02,2019-04-01 00:20:30,6226,\"1,048.0\",317,Wood St & Taylor St,59,W
abash Ave & Roose"| __truncated__ "22178531" "22178532" ...

##  $ started_at        : POSIXct[1:3228091], format: "2019-04-01 00:02:22
" NA ...

##  $ ended_at          : POSIXct[1:3228091], format: "2019-04-01 00:09:48
" NA ...

##  $ rideable_type     : chr [1:3228091] "6251" NA "5649" "4151" ...
##  $ start_station_id  : num [1:3228091] 81 NA 283 26 NA 420 503 260 NA N
A ...

##  $ start_station_name: chr [1:3228091] "Daley Center Plaza" NA "LaSalle
St & Jackson Blvd" "McClurg Ct & Illinois St" ...

##  $ end_station_id    : num [1:3228091] 56 NA 174 133 NA 426 500 499 NA
NA ...

##  $ end_station_name  : chr [1:3228091] "Desplaines St & Kinzie St" NA "
Canal St & Madison St" "Kingsbury St & Kinzie St" ...

##  $ member_casual     : chr [1:3228091] "Subscriber" NA "Subscriber" "Su
bscriber" ...

summary(all_rides)  #Statistical summary of data. Mainly for numerics

##    ride_id            started_at                       ended_at

##  Length:3228091     Min.   :2019-04-01 00:02:22   Min.   :2019-04-01 00
:09:48

##  Class :character   1st Qu.:2019-06-16 00:53:27   1st Qu.:2019-06-16 01
:05:40

##  Mode  :character   Median :2019-08-11 23:15:22   Median :2019-08-11 23
:24:24

##                     Mean   :2019-09-08 05:48:12   Mean   :2019-09-08 05
:59:27

##                     3rd Qu.:2019-11-27 08:57:45   3rd Qu.:2019-11-27 09
:05:32

##                     Max.   :2020-03-31 23:51:34   Max.   :2020-05-19 20
:10:34

##                     NA's   :1004094               NA's   :1004094

##  rideable_type      start_station_id  start_station_name end_station_id

##  Length:3228091     Min.   :  1.0     Length:3228091     Min.   :  1.0

##  Class :character   1st Qu.: 77.0     Class :character   1st Qu.: 77.0

##  Mode  :character   Median :174.0     Mode  :character   Median :174.0

##                     Mean   :203.5                        Mean   :203.4
##                     3rd Qu.:290.0                               3rd Qu.:290.0

##                     Max.   :675.0                               Max.   :675.0

##                     NA's   :1004094                             NA's   :100409
5

##  end_station_name   member_casual

##  Length:3228091     Length:3228091

##  Class :character   Class :character

##  Mode  :character   Mode  :character

##

##

##

##
```

Finding: From this quick inspection of the data, we can verify there are at least **4 problems that need to be fixed**:

1) In the *member_casual* column, there are two names for members ("member" and "subscriber") plus two names for casual riders ("customer" and "casual"). **This needs to be consolidate into just two labels**.

Let's start by assessing how many observations fall under each usertype:

```
table(all_rides$member_casual)

##

##     casual   Customer     member Subscriber

##      48480     189130     378407    1607980
```

Now, let's reassign to the desired values (we will go with the current 2020 labels):

```
all_rides <-  all_rides %>%

  mutate(member_casual = recode(member_casual

                               ,"Subscriber" = "member"

                               ,"Customer" = "casual"))
```

Finaly, Let's check to make sure the proper number of observations were reassigned:

```
table(all_rides$member_casual)

##

##  casual  member

##  237610 1986387
```

There, now we have just the two values to designate the type of user of the Cyclistic bike sharing service, *members* **and** *casual* **riders**.

2) The data can only be aggregated at the ride-level, which is too granular. **It's necessary to add some additional columns of data** – such as **day**, **month**, **year** – that provide additional opportunities to aggregate the data.

```
# Add columns that list the date, month, day, and year of each ride

all_rides$date <- as.Date(all_rides$started_at) #The default format is yyy
y-mm-dd

all_rides$month <- format(as.Date(all_rides$date), "%m")

all_rides$day <- format(as.Date(all_rides$date), "%d")

all_rides$year <- format(as.Date(all_rides$date), "%Y")

all_rides$day_of_week <- format(as.Date(all_rides$date), "%A")
```

3) We will also want to add a calculated field for *length_of_ride* since the 2020Q1 data didn't have the tripduration column. I will add *ride_length* to the entire dataframe for consistency.

```
# Add a "ride_length" calculation to all_rides (in seconds)

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

Let's have a look at the structure of the columns to verify the creation of *ride_length*

```
str(all_rides)

## tibble [3,228,091 × 15] (S3: tbl_df/tbl/data.frame)

##  $ ride_id          : chr [1:3228091] "22178529" "22178530,2019-04-01
00:03:02,2019-04-01 00:20:30,6226,\"1,048.0\",317,Wood St & Taylor St,59,W
abash Ave & Roose"| __truncated__ "22178531" "22178532" ...

##  $ started_at        : POSIXct[1:3228091], format: "2019-04-01 00:02:22
" NA ...

##  $ ended_at          : POSIXct[1:3228091], format: "2019-04-01 00:09:48
" NA ...
```

```
##  $ rideable_type     : chr [1:3228091] "6251" NA "5649" "4151" ...

##  $ start_station_id  : num [1:3228091] 81 NA 283 26 NA 420 503 260 NA N
A ...

##  $ start_station_name: chr [1:3228091] "Daley Center Plaza" NA "LaSalle
St & Jackson Blvd" "McClurg Ct & Illinois St" ...

##  $ end_station_id    : num [1:3228091] 56 NA 174 133 NA 426 500 499 NA
NA ...

##  $ end_station_name  : chr [1:3228091] "Desplaines St & Kinzie St" NA "
Canal St & Madison St" "Kingsbury St & Kinzie St" ...

##  $ member_casual     : chr [1:3228091] "member" NA "member" "member" ..
.

##  $ date              : Date[1:3228091], format: "2019-04-01" NA ...

##  $ month             : chr [1:3228091] "04" NA "04" "04" ...

##  $ day               : chr [1:3228091] "01" NA "01" "01" ...

##  $ year              : chr [1:3228091] "2019" NA "2019" "2019" ...

##  $ day_of_week       : chr [1:3228091] "Monday" NA "Monday" "Monday" ..
.

##  $ ride_length       : 'difftime' num [1:3228091] 446 NA 252 357 ...

##   ..- attr(*, "units")= chr "secs"
```

In order for us to run calculations on it, we need to convert *ride_length* from factor to numeric Let’s to that now:

```
is.factor(all_rides$ride_length)

## [1] FALSE

all_rides$ride_length <- as.numeric(as.character(all_rides$ride_length))

is.numeric(all_rides$ride_length)

## [1] TRUE
```

Let’s inspect the structure of the columns to make sure that *ride_length* is now numeric.

```
str(all_rides)

## tibble [3,228,091 × 15] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:3228091] "22178529" "22178530,2019-04-01
00:03:02,2019-04-01 00:20:30,6226,\"1,048.0\",317,Wood St & Taylor St,59,W
abash Ave & Roose"| __truncated__ "22178531" "22178532" ...

##  $ started_at        : POSIXct[1:3228091], format: "2019-04-01 00:02:22
" NA ...

##  $ ended_at          : POSIXct[1:3228091], format: "2019-04-01 00:09:48
" NA ...

##  $ rideable_type     : chr [1:3228091] "6251" NA "5649" "4151" ...

##  $ start_station_id  : num [1:3228091] 81 NA 283 26 NA 420 503 260 NA N
A ...

##  $ start_station_name: chr [1:3228091] "Daley Center Plaza" NA "LaSalle
St & Jackson Blvd" "McClurg Ct & Illinois St" ...

##  $ end_station_id    : num [1:3228091] 56 NA 174 133 NA 426 500 499 NA
NA ...

##  $ end_station_name  : chr [1:3228091] "Desplaines St & Kinzie St" NA "
Canal St & Madison St" "Kingsbury St & Kinzie St" ...

##  $ member_casual     : chr [1:3228091] "member" NA "member" "member" ..
.

##  $ date              : Date[1:3228091], format: "2019-04-01" NA ...

##  $ month             : chr [1:3228091] "04" NA "04" "04" ...

##  $ day               : chr [1:3228091] "01" NA "01" "01" ...

##  $ year              : chr [1:3228091] "2019" NA "2019" "2019" ...

##  $ day_of_week       : chr [1:3228091] "Monday" NA "Monday" "Monday" ..
.

##  $ ride_length       : num [1:3228091] 446 NA 252 357 NA 257 548 383 NA
NA ...
```

It is.

For the sake of clarity of the analysis and presentation of the results we will turn *ride_length* from seconds to minutes:

```
all_rides$ride_length <- all_rides$ride_length/60
```

Let’s have another quick inspection to verify the structure of the columns and confirm that *ride_length* now shows in minutes not seconds.

```
str(all_rides)

## tibble [3,228,091 × 15] (S3: tbl_df/tbl/data.frame)

##  $ ride_id          : chr [1:3228091] "22178529" "22178530,2019-04-01
00:03:02,2019-04-01 00:20:30,6226,\"1,048.0\",317,Wood St & Taylor St,59,W
abash Ave & Roose"| __truncated__ "22178531" "22178532" ...

##  $ started_at        : POSIXct[1:3228091], format: "2019-04-01 00:02:22
" NA ...

##  $ ended_at          : POSIXct[1:3228091], format: "2019-04-01 00:09:48
" NA ...

##  $ rideable_type     : chr [1:3228091] "6251" NA "5649" "4151" ...

##  $ start_station_id  : num [1:3228091] 81 NA 283 26 NA 420 503 260 NA N
A ...

##  $ start_station_name: chr [1:3228091] "Daley Center Plaza" NA "LaSalle
St & Jackson Blvd" "McClurg Ct & Illinois St" ...

##  $ end_station_id    : num [1:3228091] 56 NA 174 133 NA 426 500 499 NA
NA ...

##  $ end_station_name  : chr [1:3228091] "Desplaines St & Kinzie St" NA "
Canal St & Madison St" "Kingsbury St & Kinzie St" ...

##  $ member_casual     : chr [1:3228091] "member" NA "member" "member" ..
.

##  $ date              : Date[1:3228091], format: "2019-04-01" NA ...

##  $ month             : chr [1:3228091] "04" NA "04" "04" ...

##  $ day               : chr [1:3228091] "01" NA "01" "01" ...

##  $ year              : chr [1:3228091] "2019" NA "2019" "2019" ...

##  $ day_of_week       : chr [1:3228091] "Monday" NA "Monday" "Monday" ..
.

##  $ ride_length       : num [1:3228091] 7.43 NA 4.2 5.95 NA ...
```

And now it does.

4) Another issue with the data is that there are some rides where *tripduration* shows up as **negative**, including several hundred rides where bikes **were taken out of circulation for Quality Control reasons**. These rides will **have to be deleted**.

Let's do that right now:

```
all_rides_v2 <- all_rides[!(all_rides$start_station_name == "HQ QR" | all_
rides$ride_length<0),]
```

For the sake of clarity of the analysis and cleaning of the data, I will also remove the *nulls* from the dataset. We could use *drop_na() %>%* in each of the relevant commands, but in this case I opted to clean the dataset right off the bat.

```
all_rides_v2 <- na.omit(all_rides_v2)
```

Let's inspect the structure of our new dataframe to confirm that there are no null values in any of the columns:

```
summary(all_rides_v2)

##    ride_id             started_at                         ended_at

##  Length:2220220     Min.   :2019-04-01 00:02:22   Min.   :2019-04-01 00
:09:48

##  Class :character   1st Qu.:2019-06-15 18:28:41   1st Qu.:2019-06-15 18
:39:00

##  Mode  :character   Median :2019-08-11 17:39:55   Median :2019-08-11 17
:49:30

##                     Mean   :2019-09-07 22:21:36   Mean   :2019-09-07 22
:32:52

##                     3rd Qu.:2019-11-26 13:49:17   3rd Qu.:2019-11-26 13
:55:34

##                     Max.   :2020-03-31 23:51:34   Max.   :2020-05-19 20
:10:34

##  rideable_type      start_station_id start_station_name end_station_id

##  Length:2220220     Min.   :  1.0    Length:2220220     Min.   :  1.0

##  Class :character   1st Qu.: 77.0    Class :character   1st Qu.: 77.0

##  Mode  :character   Median :174.0    Mode  :character   Median :174.0

##                     Mean   :202.7                       Mean   :202.6

##                     3rd Qu.:289.0                       3rd Qu.:289.0

##                     Max.   :673.0                       Max.   :675.0
##  end_station_name   member_casual          date                month

##  Length:2220220     Length:2220220     Min.   :2019-04-01   Length:2220
220

##  Class :character   Class :character   1st Qu.:2019-06-15   Class :char
acter

##  Mode  :character   Mode  :character   Median :2019-08-11   Mode  :char
acter

##                                        Mean   :2019-09-07

##                                        3rd Qu.:2019-11-26

##                                        Max.   :2020-03-31

##      day                year            day_of_week         ride_length

##  Length:2220220     Length:2220220     Length:2220220     Min.   :
0.02

##  Class :character   Class :character   Class :character   1st Qu.:
5.50

##  Mode  :character   Mode  :character   Mode  :character   Median :
8.42

##                                                            Mean   :    1
1.26

##                                                            3rd Qu.:    1
2.10

##                                                            Max.   :15645
0.40
```

And now we have no null values littering our dataset.

Analyze

Now that the data is stored appropriately and has been prepared for analysis, it's time to start putting it to work.

In this section, I will organize and properly format the data in order to perform analysis on it, look for significant trends, relationships and generate insights from it, to help answer the business task.

KEY TASKS

- **Aggregate the data so it's useful and accessible.**

- **Organize and format the data.**
- **Perform calculations.**
- **Identify trends and relationships.**

The following R code chunks, presented in this section, will perform these tasks.

DELIVERABLE

- **A summary of the analysis.**

Let’s start by conducting our descriptive analysis.

First, we’ll have a look at the ride lengths to see if the data has anything interesting to tell us, in this aspect.

CODE

```
# Descriptive analysis on ride_length (all figures in minutes)

mean(all_rides_v2$ride_length)  #straight average (total ride length / ride
s)

## [1] 11.26118

median(all_rides_v2$ride_length) #midpoint in the ascending array of ride l
engths

## [1] 8.416667

max(all_rides_v2$ride_length) #longest ride

## [1] 156450.4

min(all_rides_v2$ride_length) #shortest ride

## [1] 0.01666667

# Or using summary() on the specific attribute to condense the above 4 line
in 1

summary(all_rides_v2$ride_length)

##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.

##      0.02      5.50      8.42     11.26     12.10 156450.40
```

Finding: From these quick calculations, it stands out, for instances, that the **average ride time for all types of riders** is of **11 minutes and 26 seconds**.

Now, let’s find out with the same metrics, if there is a big discrepancy or not in ride times between members vs casual users.

```
# Compare mean, median, max and min of ride_legth between members vs casual
users

aggregate(all_rides_v2$ride_length ~ all_rides_v2$member_casual, FUN = mea
n)

##   all_rides_v2$member_casual all_rides_v2$ride_length

## 1                     casual                 28.56247

## 2                     member                  9.22444

aggregate(all_rides_v2$ride_length ~ all_rides_v2$member_casual, FUN = med
ian)

##   all_rides_v2$member_casual all_rides_v2$ride_length

## 1                     casual                 11.81667

## 2                     member                  8.05000

aggregate(all_rides_v2$ride_length ~ all_rides_v2$member_casual, FUN = max
)

##   all_rides_v2$member_casual all_rides_v2$ride_length

## 1                     casual                156450.40

## 2                     member                 93793.52

aggregate(all_rides_v2$ride_length ~ all_rides_v2$member_casual, FUN = min
)

##   all_rides_v2$member_casual all_rides_v2$ride_length

## 1                     casual               0.03333333

## 2                     member               0.01666667
```

Finding: There is a **big difference between the ride times of members vs casual users**, where **casual user rides are much longer**, in terms of ride time, than member user rides.

Now, let’s find out if this varies in terms of the day of the week or if this difference in ride times between members and casual users holds regardless the day of the week.

```
# Let's assess the average ride time by each day for members vs casual user
s

aggregate(all_rides_v2$ride_length ~ all_rides_v2$member_casual + all_ride
s_v2$day_of_week, FUN = mean)
```

```
##    all_rides_v2$member_casual all_rides_v2$day_of_week all_rides_v2$rid
e_length

## 1                          casual                   Friday                2
8.151452

## 2                          member                   Friday
9.124590

## 3                          casual                   Monday                2
3.014277

## 4                          member                   Monday
9.270657

## 5                          casual                 Saturday                2
6.040054

## 6                          member                 Saturday
9.714433

## 7                          casual                   Sunday                3
6.457537

## 8                          member                   Sunday                1
0.011278

## 9                          casual                 Thursday                3
0.820628

## 10                         member                 Thursday
8.999737

## 11                         casual                  Tuesday                2
5.641588

## 12                         member                  Tuesday
9.025934

## 13                         casual                Wednesday                2
4.700810

## 14                         member                Wednesday
9.036617
```

- **Sunday**
- **Monday**
- **Tuesday**
- **Wednesday**

- **Thursday**
- **Friday**
- **Saturday**

  Finding: As we can confirm **this trend holds across all days of the week**, **casual user rides are in average always much longer in terms of ride time**, than member user rides.

So, we have confirmed that on average rides taken by casual users of Cyclistic are much longer than those taken by members.

Now, let's explore the number of rides – who takes more rides, casual users or members?

Let's find out.

```
all_rides_v2 %>%

  group_by(member_casual) %>%           # groups by type of user

  summarise(number_of_rides = n()) %>%  # calculates the number of rides

  arrange(member_casual)                # sorts it by type of user

## # A tibble: 2 × 2

##   member_casual number_of_rides

##   <chr>                   <int>

## 1 casual                 233840

## 2 member                1986380
```

Finding: As we can confirm, **in terms of the number of rides made**, **members make significantly more rides in total than casual riders**—precisely **89.47% more rides**.

Now, let's verify if this is a trend that holds throughout the days of the week, or if on some days there are more registered rides from casual users than from members, for example on weekends.

Let's check it out.

```
all_rides_v2 %>%

  mutate(weekday = wday(started_at, label = TRUE)) %>% # creates the weekd
ay field

  group_by(member_casual, weekday) %>% #groups by usertype and weekday

  summarise(number_of_rides = n()) %>%# calculates the number of rides

  arrange(member_casual, weekday) # sorts it
```

```
## `summarise()` has grouped output by 'member_casual'. You can override u
sing the

## `.groups` argument.

## # A tibble: 14 × 3

## # Groups:   member_casual [2]

##    member_casual weekday number_of_rides

##    <chr>         <ord>             <int>

##  1 casual        Sun               49154

##  2 casual        Mon               25719

##  3 casual        Tue               26391

##  4 casual        Wed               27666

##  5 casual        Thu               28303

##  6 casual        Fri               32142

##  7 casual        Sat               44465

##  8 member        Sun              172335

##  9 member        Mon              323335

## 10 member        Tue              350426

## 11 member        Wed              339217

## 12 member        Thu              323053

## 13 member        Fri              303302

## 14 member        Sat              174712
```

Finding: As we can confirm, **in terms of the number of rides made**, **members make significantly more rides everyday than casual riders.**

DELIVERABLE: To sum up, there is a **clear difference between casual users and members**, whereas **casual riders take fewer rides, but longer ones**, which can suggest a more recreational use of the service; **members have a much higher number of rides, but much shorter ones**, which can imply a much more functional use of the service.

This is clearly shown and supported by the table below .

```
all_rides_v2 %>%

  mutate(weekday = wday(started_at, label = TRUE)) %>%

  group_by(member_casual, weekday) %>%  #groups by usertype and weekday

  summarise(number_of_rides = n(), average_duration = mean(ride_length)) %
>%

  arrange(member_casual, weekday)

## `summarise()` has grouped output by 'member_casual'. You can override u
sing the

## `.groups` argument.

## # A tibble: 14 × 4

## # Groups:   member_casual [2]

##    member_casual weekday number_of_rides average_duration

##    <chr>         <ord>             <int>            <dbl>

##  1 casual        Sun               49154            36.5

##  2 casual        Mon               25719            23.0

##  3 casual        Tue               26391            25.6

##  4 casual        Wed               27666            24.7

##  5 casual        Thu               28303            30.8

##  6 casual        Fri               32142            28.2

##  7 casual        Sat               44465            26.0

##  8 member        Sun              172335            10.0

##  9 member        Mon              323335             9.27

## 10 member        Tue              350426             9.03

## 11 member        Wed              339217             9.04

## 12 member        Thu              323053             9.00

## 13 member        Fri              303302             9.12
## 14 member         Sat                174712              9.71
```

Share

Now that we have performed our analysis and gained significant insights into the data, it's time to create some effective visualizations and share our findings.

KEY TASKS

- **Determine the best way to share your findings.**
- **Create effective data visualizations.**
- **Present your findings.**
- **Ensure your work is accessible.**

DELIVERABLE

- **Supporting visualizations and key findings.**

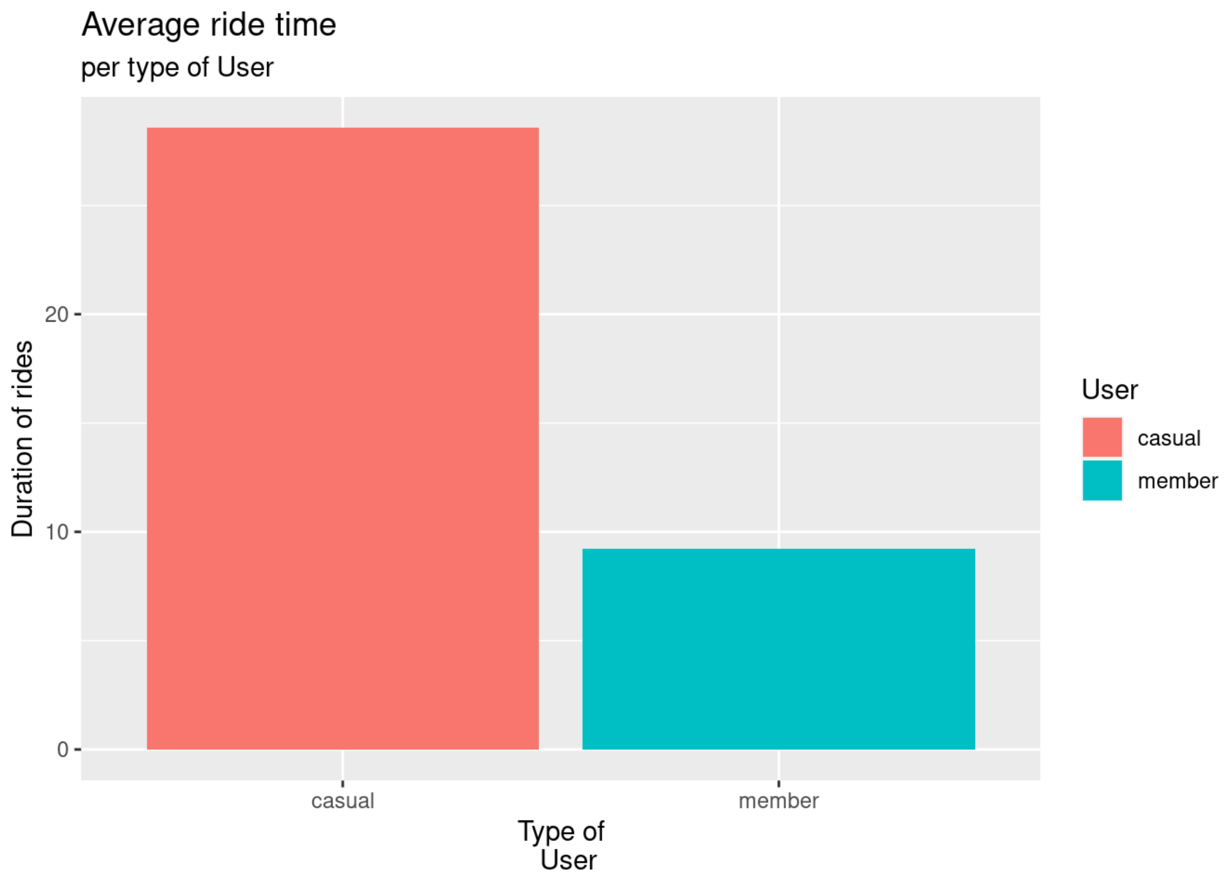
In this section, we will go straight to the **Deliverable**, since we will list our main findings supported by the respective visualization to drive each point home.

So let's get to it.

1) One of the main differences in how members and casual riders use the Cyclistic bike sharing service **is in relation to the time of the rides**. **Casual user rides are in average much longer** in comparison to those of the members.

- **Let's visualize this.**

```
all_rides_v2 %>%

  group_by(member_casual) %>%

  summarise(average_duration = mean(ride_length)) %>%

  arrange(member_casual) %>%

  ggplot(aes( x = member_casual, y = average_duration, fill = member_casua
l)) +

  geom_col(position = "dodge") +

  labs(title = "Average ride time", subtitle = "per type of User", x = "Ty
pe of

  User", y = "Duration of rides")+

  scale_fill_discrete(name = "User")
```



2) Another very important finding, regarding the average ride time, is that **this trend, of casual user riders being in average much longer than the member rides, holds across all days of the week**.

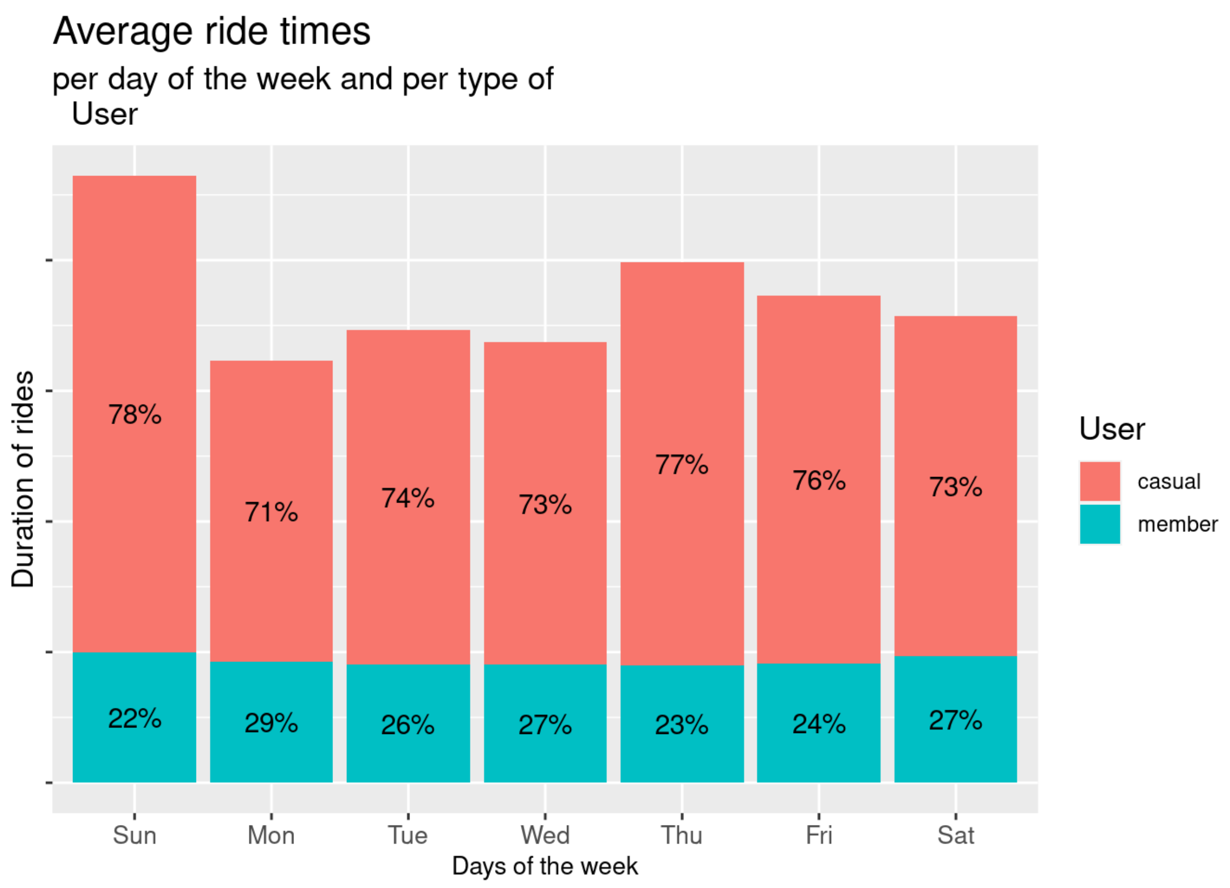
- So, casual user rides are in average, every day of the week, much longer in terms of ride time, than member user rides.

As a reminder, here are the corresponding days in english:

- **Sunday**
- **Monday**
- **Tuesday**
- **Wednesday**
- **Thursday**
- **Friday**
- **Saturday**

Let's go the visual

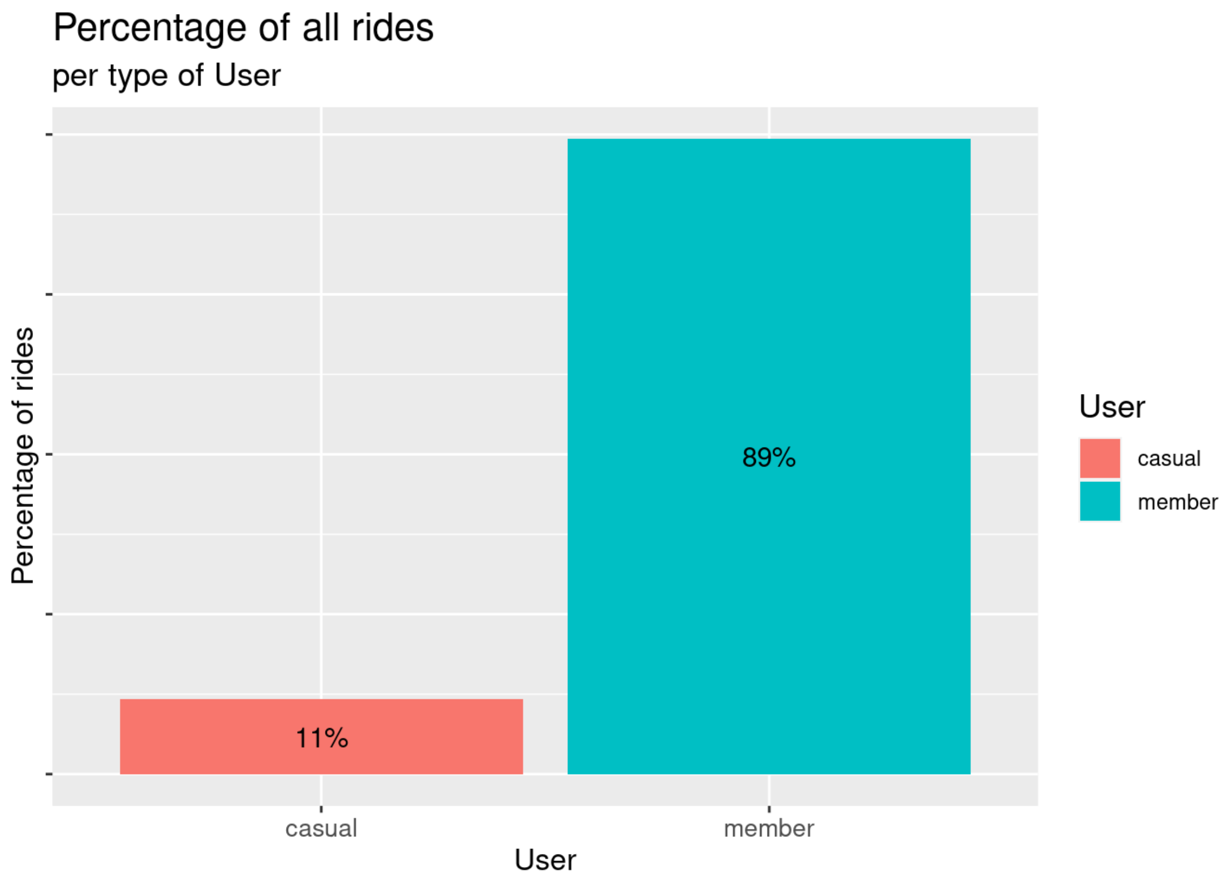
```
all_rides_v2 %>%

mutate(weekday = wday(started_at, label = TRUE)) %>%  #creates weekday fie
ld wday()

  group_by(member_casual,weekday) %>%
  summarise(number_of_rides=n(), average_duration = mean(ride_length)) %>%

  group_by(weekday) %>%

  mutate(percent_average_duration = average_duration/sum(average_duration)
) %>%

  mutate(percent_average_duration = round(percent_average_duration*100)) %
>%

  ggplot(aes(x = weekday, y = average_duration, fill=member_casual)) +

  geom_col(position="stack") +

  geom_text(aes(label = paste0(percent_average_duration, "%")),

            position = position_stack(vjust = 0.5)) +

  labs(title = "Average ride times", subtitle = "per day of the week and p
er type of

  User", x = "Days of the week", y = "Duration of rides")+

  scale_fill_discrete(name = "User") +

  theme(title = element_text(size = 13),

        axis.title.x = element_text(size = 10),

        axis.title.y = element_text(size = 12),

        axis.text.x = element_text(size = 10),

        axis.text.y = element_blank())

## `summarise()` has grouped output by 'member_casual'. You can override u
sing the

## `.groups` argument.
```



3) In terms of the number of rides, on the other hand, **members make significantly more rides in total** than casual riders—precisely **89.47% more rides**, like the data has shown.

Let's get a visual of it.

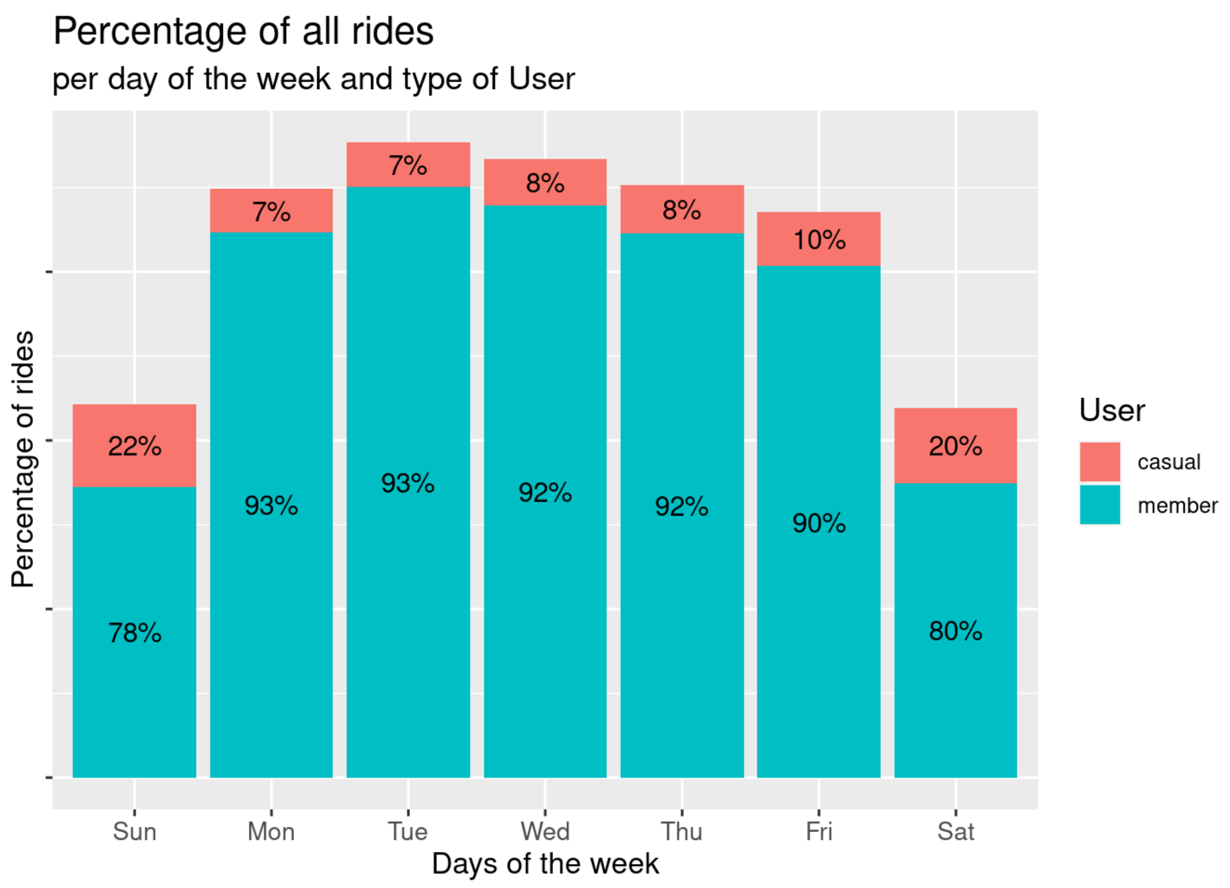
```
all_rides_v2 %>%

  group_by(member_casual) %>%           # groups by type of user

  summarise(number_of_rides = n()) %>%  # calculates the number of rides

  arrange(member_casual) %>%

  mutate(percent_number_of_rides = number_of_rides/sum(number_of_rides)) %
>% # calculates the percentages on the number of rides

  mutate(percent_number_of_rides = round(percent_number_of_rides*100)) %>%
# calculates the percentages on the number of rides

  ggplot(aes(x = member_casual,y = number_of_rides, fill = member_casual))
+

  geom_col(position="dodge")+
  geom_text(aes(label = paste0(percent_number_of_rides, "%")),
            position = position_stack(vjust = 0.5))+ # Adds percentages to
columns
  labs(title = "Percentage of all rides", subtitle = "per type of User", x
= "User", y = "Percentage of rides")+
  scale_fill_discrete(name = "User") + # Changes the legend title
  theme(title = element_text(size = 13),
        axis.title.x = element_text(size = 12),
        axis.title.y = element_text(size = 12),
        axis.text.x = element_text(size = 10),
        axis.text.y = element_blank())
```



4) Also, in terms of the number of rides made, **members make significantly more rides, regardless the day of the week**, than casual riders.

Let’s have a look.

```
all_rides_v2 %>%

  mutate(weekday = wday(started_at, label = TRUE)) %>%

  group_by(member_casual, weekday) %>%  #groups by usertype and weekday

  summarise(number_of_rides = n()) %>%      # calculates the number of rid
es

  group_by(weekday) %>%

  mutate(percent_number_of_rides = number_of_rides/sum(number_of_rides)) %
>%

  mutate(percent_number_of_rides = round(percent_number_of_rides*100)) %>%

  ggplot(aes(x = weekday, y = number_of_rides, fill= member_casual)) +

  geom_col(position="stack") + # transforms our standard column graph into
a stacked column graph

  geom_text(aes(label = paste0(percent_number_of_rides, "%")),

            position = position_stack(vjust = 0.5))+

  labs(title = "Percentage of all rides", subtitle = "per day of the week
and type of User", x = "User", y = "Percentage of rides")+

  xlab("Days of the week") +

  ylab("Percentage of rides") +

  scale_fill_discrete(name = "User") + # Changes the legend title

  theme(title = element_text(size = 13),

        axis.title.x = element_text(size = 12),

        axis.title.y = element_text(size = 12),

        axis.text.x = element_text(size = 10),

        axis.text.y = element_blank())

## `summarise()` has grouped output by 'member_casual'. You can override u
sing the

## `.groups` argument.
```



5) There is a **noticeable dip in the number of rides that members take on weekends**, although they still constitute a much larger percentage when compared with casual users. **This dip could offer an interesting clue revealed by the data**, helping us trace the profile of this user type. Combining this information** with what the data has already shown us — **that members, on average, take much shorter rides —** could strongly indicate that this user group tends to use the service more on workdays**, likely for their** daily commute**.

So, **the data has shown** there is a **clear difference between casual users and members**:

- **Casual riders take fewer rides, but longer ones**, which strongly **suggests a more recreational use of the service;**
- **Members have a much higher number of rides, but much shorter ones**, which can **imply a much more functional use of the service, such as a daily commute to work**.

Act

Now that the analysis is finished and we have obtained our results, it's time to present our conclusions and recommendations based on your analysis, to the Cyclistic Executive Team and the Director of Marketing.

So based on this analysis, **these are my recommendations**:

1) Since the casual riders have fewer number of rides, but a much higher average of ride time, of *28min:56sec* vs the *9min:22sec* of the members average ride time, I would recommend to change the casual ride plan to **include a “long ride fee”, for rides over 19 minutes, above which the ride fee would increase in 15%, at the 20min mark; in 20% at the 25min mark, and of 35% at the 30min mark.**

2) This condition, naturally, would not be applicable in the member plan subscription, which would have **no limits on the number of rides, nor on ride time**;

3) To make the member subscription plan even more attractive to the casual riders, I would recommend to **launch a campaign of 35% off the member subscription plan to casual riders that used the service more than 20 times in the last 2 months.** This would be attractive to riders who already use the service frequently and are loyal customers who just need a little “nudge” to go for the member plan, at that same time that prevents that someone, who just used the service once, could get access to this special price deal;

4) Lastly, I would recommend **a member-get-member type of deal**, where for **each friend a member gets to subscribe to member subscription plan, he would get a 15% cash back in a voucher for goods and services, and the friend a 15% off his or hers own member subscription plan.**