Divvy_Analysis

Sahebrao Mekal

2022-12-22

INTRODUCTION

This capstone project is the final project in my Google Data Analytics Professional Certificate course.

In this case study I will be analyzing a public dataset for a fictional company provided by the course.

I will be using R programming language for this analysis because of its fast big data processing, easy cleanup,

statistical analytics tools and data visualizations.

The following data analysis steps will be followed:

Ask,

Prepare,

Process,

Analyze,

Share,

Act.

The case study roadmap below will be followed on each step:

Code, when needed.

> Key tasks.

> Deliverables.

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.

*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?

Lily Moreno (the director of marketing and my manager) has assigned you the first question to answer:

“1. How do annual members and casual riders use Cyclistic bikes differently?”

1. A clear statement of the business task

Key tasks

1. Identify the business task

Converting casual riders to annual members by understanding how they differ.

Average time Cyclistic bikes used by casual riders and annual members by average ride time/ day of the week .

2. Consider key stakeholders

Director of Marketing (Lily Moreno), Marketing Analytics team, Executive team.

3. Deliverable

A clear statement of the business task

To find the differences between the casual riders and annual members.

*PREPARE

I will use Cyclistic's historical trip data to analyze and identify trends.

The data has been made available by Motivate International Inc. under this license. Datasets are available here link.

Key tasks

Download data and store it appropriately.

Data has been downloaded and copies have been stored securely on my computer.

2. Identify how it's organized.

The data is in CSV (comma-separated values) format, and there are a total of 24 relevant columns.

3. Sort and filter the data.

For this analysis, I will be using data from

(Divvy_Trips_2019_Q1

Divvy_Trips_2019_Q2

Divvy_Trips_2019_Q3

Divvy_Trips_2019_Q4

Divvy_Trips_2020_Q1).

4. Determine the credibility of the data.

For the purpose of this case study, the datasets are appropriate and will enable me to answer the business questions.

But data-privacy issues will prohibit me from using rider's personally identifiable information and this will prevent me from determining

if riders have purchased multiple single passes. All ride_ids are unique.

Deliverable

A description of all data sources used

Main source of data provided by the Cylistic company.

The data has been made available by Motivate International Inc. This is public data.

#==========

Divvy_Analysis

This analysis is based on the Divvy case study "'Sophisticated, Clear, and Polished': Divvy and Data Visualization" written by Sahebrao Mekal.

The purpose of this script is to consolidate downloaded Divvy data into a single dataframe and then conduct simple analysis to help answer

The key question: “In what ways do members and casual riders use Divvy bikes differently?”

# #

Install required packages

tidyverse for data import and wrangling

lubridate for date functions

ggplot for visualization

# #

#=====================

STEP 1: COLLECT DATA

#=====================

2. A description of all data sources used

Upload Divvy datasets (csv files) here by using the import dataset function in the File menu.

or

Upload Divvy datasets (csv files) here

```
Divvy_Trips_2019_Q1 <- read_csv("Divvy_Trips_2019_Q1.csv")
```

```
## Rows: 365069 Columns: 12
## ── Column specification
──────────────────────────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (4): from_station_name, to_station_name, usertype, gender
## dbl  (5): trip_id, bikeid, from_station_id, to_station_id, birthyear
## num  (1): tripduration
## 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 this message.
```

```
Divvy_Trips_2019_Q2 <- read_csv("Divvy_Trips_2019_Q2.csv")
```

```
## Rows: 1108163 Columns: 12
## ── Column specification ──────────────────────────────────────────────────────
## Delimiter: ","
## chr  (4): 03 - Rental Start Station Name, 02 - Rental End Station Name, User...
## dbl  (5): 01 - Rental Details Rental ID, 01 - Rental Details Bike ID, 03 - R...
## num  (1): 01 - Rental Details Duration In Seconds Uncapped
## dttm (2): 01 - Rental Details Local Start Time, 01 - Rental Details Local 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 this message.

Divvy_Trips_2019_Q3 <- read_csv("Divvy_Trips_2019_Q3.csv")

## Rows: 1640718 Columns: 12
## ── Column specification ──────────────────────────────────────────────────────
## Delimiter: ","
## chr  (4): from_station_name, to_station_name, usertype, gender
## dbl  (5): trip_id, bikeid, from_station_id, to_station_id, birthyear
## num  (1): tripduration
## 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 this message.

Divvy_Trips_2019_Q4 <- read_csv("Divvy_Trips_2019_Q4.csv")

## Rows: 704054 Columns: 12
## ── Column specification ──────────────────────────────────────────────────────
## Delimiter: ","
## chr  (4): from_station_name, to_station_name, usertype, gender
## dbl  (5): trip_id, bikeid, from_station_id, to_station_id, birthyear
## num  (1): tripduration
## 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 this message.
```

```
Divvy_Trips_2020_Q1 <- read_csv("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_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.
```

```
#=====================================================
# STEP 2: WRANGLE DATA AND COMBINE INTO A SINGLE FILE
#=====================================================
```

Compare column names each of the files

use a command to join them into one file

```
colnames(Divvy_Trips_2019_Q1)

##  [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(Divvy_Trips_2019_Q2)

##  [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(Divvy_Trips_2019_Q3)
```

```
##  [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(Divvy_Trips_2019_Q4)
```

```
##  [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(Divvy_Trips_2020_Q1)
```

```
##  [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"
```

Rename columns to make them consistent (as this will be the supposed going-forward table design for Divvy)

```
(Divvy_Trips_2019_Q4 <- rename(Divvy_Trips_2019_Q4
                ,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_t…¹ tripd…² start…³
```

```
##       <dbl> <dttm>              <dttm>                <dbl>   <dbl>   <dbl>
##  1 25223640 2019-10-01 00:01:39 2019-10-01 00:17:20        2215     940      20
##  2 25223641 2019-10-01 00:02:16 2019-10-01 00:06:34        6328     258      19
##  3 25223642 2019-10-01 00:04:32 2019-10-01 00:18:43        3003     850      84
##  4 25223643 2019-10-01 00:04:32 2019-10-01 00:43:43        3275    2350     313
##  5 25223644 2019-10-01 00:04:34 2019-10-01 00:35:42        5294    1867     210
##  6 25223645 2019-10-01 00:04:38 2019-10-01 00:10:51        1891     373     156
##  7 25223646 2019-10-01 00:04:52 2019-10-01 00:22:45        1061    1072      84
##  8 25223647 2019-10-01 00:04:57 2019-10-01 00:29:16        1274    1458     156
##  9 25223648 2019-10-01 00:05:20 2019-10-01 00:29:18        6011    1437     156
## 10 25223649 2019-10-01 00:05:20 2019-10-01 02:23:46        2957    8306     336
## # … with 704,044 more rows, 6 more variables: start_station_name <chr>,
## #   end_station_id <dbl>, end_station_name <chr>, member_casual <chr>,
## #   gender <chr>, birthyear <dbl>, and abbreviated variable names
## #   ¹rideable_type, ²tripduration, ³start_station_id
```

```
(Divvy_Trips_2019_Q3 <- rename(Divvy_Trips_2019_Q3
                ,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: 1,640,718 × 12
##     ride_id started_at          ended_at            rideable_t…¹ tripd…² start…³
##       <dbl> <dttm>              <dttm>                <dbl>   <dbl>   <dbl>
##  1 23479388 2019-07-01 00:00:27 2019-07-01 00:20:41        3591    1214     117
##  2 23479389 2019-07-01 00:01:16 2019-07-01 00:18:44        5353    1048     381
##  3 23479390 2019-07-01 00:01:48 2019-07-01 00:27:42        6180    1554     313
##  4 23479391 2019-07-01 00:02:07 2019-07-01 00:27:10        5540    1503     313
##  5 23479392 2019-07-01 00:02:13 2019-07-01 00:22:26        6014    1213     168
##  6 23479393 2019-07-01 00:02:21 2019-07-01 00:07:31        4941     310     300
##  7 23479394 2019-07-01 00:02:24 2019-07-01 00:23:12        3770    1248     168
##  8 23479395 2019-07-01 00:02:26 2019-07-01 00:28:16        5442    1550     313
##  9 23479396 2019-07-01 00:02:34 2019-07-01 00:28:57        2957    1583      43
## 10 23479397 2019-07-01 00:02:45 2019-07-01 00:29:14        6091    1589      43
## # … with 1,640,708 more rows, 6 more variables: start_station_name <chr>,
```

```
## #   end_station_id <dbl>, end_station_name <chr>, member_casual <chr>,
## #   gender <chr>, birthyear <dbl>, and abbreviated variable names
## #   ¹rideable_type, ²tripduration, ³start_station_id
```

```
(Divvy_Trips_2019_Q2 <- rename(Divvy_Trips_2019_Q2
                ,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: 1,108,163 × 12
##     ride_id started_at          ended_at            rideable_t…¹ 01 - …² start…³
##       <dbl> <dttm>              <dttm>                     <dbl>   <dbl>   <dbl>
##  1 22178529 2019-04-01 00:02:22 2019-04-01 00:09:48         6251     446      81
##  2 22178530 2019-04-01 00:03:02 2019-04-01 00:20:30         6226    1048     317
##  3 22178531 2019-04-01 00:11:07 2019-04-01 00:15:19         5649     252     283
##  4 22178532 2019-04-01 00:13:01 2019-04-01 00:18:58         4151     357      26
##  5 22178533 2019-04-01 00:19:26 2019-04-01 00:36:13         3270    1007     202
##  6 22178534 2019-04-01 00:19:39 2019-04-01 00:23:56         3123     257     420
##  7 22178535 2019-04-01 00:26:33 2019-04-01 00:35:41         6418     548     503
##  8 22178536 2019-04-01 00:29:48 2019-04-01 00:36:11         4513     383     260
##  9 22178537 2019-04-01 00:32:07 2019-04-01 01:07:44         3280    2137     211
## 10 22178538 2019-04-01 00:32:19 2019-04-01 01:07:39         5534    2120     211
## # … with 1,108,153 more rows, 6 more variables: start_station_name <chr>,
## #   end_station_id <dbl>, end_station_name <chr>, member_casual <chr>,
## #   `Member Gender` <chr>, `05 - Member Details Member Birthday Year` <dbl>,
## #   and abbreviated variable names ¹rideable_type,
## #   ²`01 - Rental Details Duration In Seconds Uncapped`, ³start_station_id
```

Inspect the dataframes and look for incongruencies

```
str(Divvy_Trips_2020_Q1)
```

```
## spc_tbl_ [426,887 × 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:426887] "EACB19130B0CDA4A" "8FED874C809DC021" "789F3C21E472CA96"
```

```
"C9A388DAC6ABF313" ...
##  $ rideable_type     : chr [1:426887] "docked_bike" "docked_bike" "docked_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 38 ...
##  $ end_station_name  : chr [1:426887] "Clark St & Leland Ave" "Southport Ave & Irving Park Rd" "Wilton
Ave & Belmont Ave" "Fairbanks Ct & Grand Ave" ...
##  $ 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" "member" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   ride_id = col_character(),
##   ..   rideable_type = col_character(),
##   ..   started_at = col_datetime(format = ""),
##   ..   ended_at = col_datetime(format = ""),
##   ..   start_station_name = col_character(),
##   ..   start_station_id = col_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>

str(Divvy_Trips_2019_Q4)

## spc_tbl_ [704,054 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : num [1:704054] 25223640 25223641 25223642 25223643 25223644 ...
##  $ 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 3275 5294 ...
##  $ tripduration      : num [1:704054] 940 258 850 2350 1867 ...
```

```
##  $ start_station_id  : num [1:704054] 20 19 84 313 210 156 84 156 156 336 ...
##  $ start_station_name: chr [1:704054] "Sheffield Ave & Kingsbury St" "Throop (Loomis) St & Taylor St"
"Milwaukee Ave & Grand Ave" "Lakeview Ave & Fullerton Pkwy" ...
##  $ end_station_id    : num [1:704054] 309 241 199 290 382 226 142 463 463 336 ...
##  $ end_station_name  : chr [1:704054] "Leavitt St & Armitage Ave" "Morgan St & Polk St" "Wabash Ave &
Grand Ave" "Kedzie Ave & Palmer Ct" ...
##  $ member_casual     : chr [1:704054] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
##  $ gender            : chr [1:704054] "Male" "Male" "Female" "Male" ...
##  $ birthyear         : num [1:704054] 1987 1998 1991 1990 1987 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   trip_id = col_double(),
##   ..   start_time = col_datetime(format = ""),
##   ..   end_time = col_datetime(format = ""),
##   ..   bikeid = col_double(),
##   ..   tripduration = col_number(),
##   ..   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(Divvy_Trips_2019_Q3)
```

```
## spc_tbl_ [1,640,718 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : num [1:1640718] 23479388 23479389 23479390 23479391 23479392 ...
##  $ started_at        : POSIXct[1:1640718], format: "2019-07-01 00:00:27" "2019-07-01 00:01:16" ...
##  $ ended_at          : POSIXct[1:1640718], format: "2019-07-01 00:20:41" "2019-07-01 00:18:44" ...
##  $ rideable_type     : num [1:1640718] 3591 5353 6180 5540 6014 ...
##  $ tripduration      : num [1:1640718] 1214 1048 1554 1503 1213 ...
##  $ start_station_id  : num [1:1640718] 117 381 313 313 168 300 168 313 43 43 ...
##  $ start_station_name: chr [1:1640718] "Wilton Ave & Belmont Ave" "Western Ave & Monroe St"
"Lakeview Ave & Fullerton Pkwy" "Lakeview Ave & Fullerton Pkwy" ...
##  $ end_station_id    : num [1:1640718] 497 203 144 144 62 232 62 144 195 195 ...
##  $ end_station_name  : chr [1:1640718] "Kimball Ave & Belmont Ave" "Western Ave & 21st St" "Larrabee
St & Webster Ave" "Larrabee St & Webster Ave" ...
##  $ member_casual     : chr [1:1640718] "Subscriber" "Customer" "Customer" "Customer" ...
```

```
##  $ gender            : chr [1:1640718] "Male" NA NA NA ...
##  $ birthyear         : num [1:1640718] 1992 NA NA NA NA ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   trip_id = col_double(),
##   ..   start_time = col_datetime(format = ""),
##   ..   end_time = col_datetime(format = ""),
##   ..   bikeid = col_double(),
##   ..   tripduration = col_number(),
##   ..   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(Divvy_Trips_2019_Q2)
```

```
## spc_tbl_ [1,108,163 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id                                  : num [1:1108163] 22178529 22178530 22178531 22178532
22178533 ...
##  $ started_at                               : POSIXct[1:1108163], format: "2019-04-01 00:02:22" "2019-04-01
00:03:02" ...
##  $ ended_at                                 : POSIXct[1:1108163], format: "2019-04-01 00:09:48" "2019-04-01
00:20:30" ...
##  $ rideable_type                            : num [1:1108163] 6251 6226 5649 4151 3270 ...
##  $ 01 - Rental Details Duration In Seconds Uncapped: num [1:1108163] 446 1048 252 357 1007 ...
##  $ start_station_id                         : num [1:1108163] 81 317 283 26 202 420 503 260 211 211 ...
##  $ start_station_name                       : chr [1:1108163] "Daley Center Plaza" "Wood St & Taylor St"
"LaSalle St & Jackson Blvd" "McClurg Ct & Illinois St" ...
##  $ end_station_id                           : num [1:1108163] 56 59 174 133 129 426 500 499 211 211 ...
##  $ end_station_name                         : chr [1:1108163] "Desplaines St & Kinzie St" "Wabash Ave &
Roosevelt Rd" "Canal St & Madison St" "Kingsbury St & Kinzie St" ...
##  $ member_casual                            : chr [1:1108163] "Subscriber" "Subscriber" "Subscriber"
"Subscriber" ...
##  $ Member Gender                            : chr [1:1108163] "Male" "Female" "Male" "Male" ...
##  $ 05 - Member Details Member Birthday Year : num [1:1108163] 1975 1984 1990 1993 1992 ...
##  - attr(*, "spec")=
```

```
##   .. cols(
##   ..   `01 - Rental Details Rental ID` = col_double(),
##   ..   `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_number(),
##   ..   `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(Divvy_Trips_2019_Q1)

## spc_tbl_ [365,069 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ trip_id          : num [1:365069] 21742443 21742444 21742445 21742446 21742447 ...
##  $ start_time       : POSIXct[1:365069], format: "2019-01-01 00:04:37" "2019-01-01 00:08:13" ...
##  $ end_time         : POSIXct[1:365069], format: "2019-01-01 00:11:07" "2019-01-01 00:15:34" ...
##  $ bikeid           : num [1:365069] 2167 4386 1524 252 1170 ...
##  $ tripduration     : num [1:365069] 390 441 829 1783 364 ...
##  $ from_station_id  : num [1:365069] 199 44 15 123 173 98 98 211 150 268 ...
##  $ from_station_name: chr [1:365069] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave
& 18th St" "California Ave & Milwaukee Ave" ...
##  $ to_station_id    : num [1:365069] 84 624 644 176 35 49 49 142 148 141 ...
##  $ to_station_name  : chr [1:365069] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)"
"Western Ave & Fillmore St (*)" "Clark St & Elm St" ...
##  $ usertype         : chr [1:365069] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
##  $ gender           : chr [1:365069] "Male" "Female" "Female" "Male" ...
##  $ birthyear        : num [1:365069] 1989 1990 1994 1993 1994 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   trip_id = col_double(),
##   ..   start_time = col_datetime(format = ""),
##   ..   end_time = col_datetime(format = ""),
##   ..   bikeid = col_double(),
##   ..   tripduration = col_number(),
##   ..   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>
```

Convert ride_id and rideable_type to character so that they can stack correctly

```
Divvy_Trips_2019_Q4 <-  mutate(Divvy_Trips_2019_Q4, ride_id = as.character(ride_id)
                   ,rideable_type = as.character(rideable_type))
Divvy_Trips_2019_Q3 <-  mutate(Divvy_Trips_2019_Q3, ride_id = as.character(ride_id)
                   ,rideable_type = as.character(rideable_type))
Divvy_Trips_2019_Q2 <-  mutate(Divvy_Trips_2019_Q2, ride_id = as.character(ride_id)
                   ,rideable_type = as.character(rideable_type))
```

Stack individual quarter's data frames into one big data frame.

```
all_trips <- bind_rows(Divvy_Trips_2019_Q1, Divvy_Trips_2019_Q2, Divvy_Trips_2019_Q3,
Divvy_Trips_2019_Q4, Divvy_Trips_2020_Q1)
```

Remove lat, long, birthyear, and gender fields as this data was dropped beginning in 2020.

```
all_trips <- all_trips %>%
  select(-c(start_lat, start_lng, end_lat, end_lng, birthyear, gender, "01 - Rental Details Duration In Seconds
Uncapped", "05 - Member Details Member Birthday Year", "Member Gender", "tripduration"))
```

*PROCESS

Cleaning and preparing data for analysis

Key tasks

Check the data for errors.

Choose your tools.

Transform the data so you can work with it effectively.

Document the cleaning process.

Deliverable

Documentation of any cleaning or manipulation of data

3. Documentation of any cleaning or manipulation of data

#===

STEP 3: CLEAN UP AND ADD DATA TO PREPARE FOR ANALYSIS

#===

Inspect the new table that has been created

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

```
##  [1] "trip_id"            "start_time"         "end_time"
##  [4] "bikeid"             "from_station_id"    "from_station_name"
##  [7] "to_station_id"      "to_station_name"    "usertype"
## [10] "ride_id"            "started_at"         "ended_at"
## [13] "rideable_type"      "start_station_id"   "start_station_name"
## [16] "end_station_id"     "end_station_name"   "member_casual"
```

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

```
## [1] 4244891
```

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

```
## [1] 4244891      18
```

```
head(all_trips)  #See the first 6 rows of data frame.  Also tail(all_trips)
```

```
## # A tibble: 6 × 18
##   trip_id start_time          end_time            bikeid from_…¹ from_…² to_st…³
```

```
##     <dbl> <dttm>              <dttm>              <dbl>   <dbl> <chr>     <dbl>
## 1  2.17e7 2019-01-01 00:04:37 2019-01-01 00:11:07   2167     199 Wabash…      84
## 2  2.17e7 2019-01-01 00:08:13 2019-01-01 00:15:34   4386      44 State …     624
## 3  2.17e7 2019-01-01 00:13:23 2019-01-01 00:27:12   1524      15 Racine…     644
## 4  2.17e7 2019-01-01 00:13:45 2019-01-01 00:43:28    252     123 Califo…     176
## 5  2.17e7 2019-01-01 00:14:52 2019-01-01 00:20:56   1170     173 Mies v…      35
## 6  2.17e7 2019-01-01 00:15:33 2019-01-01 00:19:09   2437      98 LaSall…      49
## # … with 11 more variables: to_station_name <chr>, usertype <chr>,
## #   ride_id <chr>, started_at <dttm>, ended_at <dttm>, rideable_type <chr>,
## #   start_station_id <dbl>, start_station_name <chr>, end_station_id <dbl>,
## #   end_station_name <chr>, member_casual <chr>, and abbreviated variable names
## #   ¹from_station_id, ²from_station_name, ³to_station_id
```

```
str(all_trips)  #See list of columns and data types (numeric, character, etc)
```

```
## tibble [4,244,891 × 18] (S3: tbl_df/tbl/data.frame)
##  $ trip_id           : num [1:4244891] 21742443 21742444 21742445 21742446 21742447 ...
##  $ start_time        : POSIXct[1:4244891], format: "2019-01-01 00:04:37" "2019-01-01 00:08:13" ...
##  $ end_time          : POSIXct[1:4244891], format: "2019-01-01 00:11:07" "2019-01-01 00:15:34" ...
##  $ bikeid            : num [1:4244891] 2167 4386 1524 252 1170 ...
##  $ from_station_id   : num [1:4244891] 199 44 15 123 173 98 98 211 150 268 ...
##  $ from_station_name : chr [1:4244891] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine
Ave & 18th St" "California Ave & Milwaukee Ave" ...
##  $ to_station_id     : num [1:4244891] 84 624 644 176 35 49 49 142 148 141 ...
##  $ to_station_name   : chr [1:4244891] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)"
"Western Ave & Fillmore St (*)" "Clark St & Elm St" ...
##  $ usertype          : chr [1:4244891] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
##  $ ride_id           : chr [1:4244891] NA NA NA NA ...
##  $ started_at        : POSIXct[1:4244891], format: NA NA ...
##  $ ended_at          : POSIXct[1:4244891], format: NA NA ...
##  $ rideable_type     : chr [1:4244891] NA NA NA NA ...
##  $ start_station_id  : num [1:4244891] NA NA NA NA NA NA NA NA NA NA ...
##  $ start_station_name: chr [1:4244891] NA NA NA NA ...
##  $ end_station_id    : num [1:4244891] NA NA NA NA NA NA NA NA NA NA ...
##  $ end_station_name  : chr [1:4244891] NA NA NA NA ...
##  $ member_casual     : chr [1:4244891] NA NA NA NA ...
```

```
summary(all_trips)  #Statistical summary of data. Mainly for numerics
```

```
##     trip_id           start_time
##  Min.   :21742443   Min.   :2019-01-01 00:04:37.00
```

```
##  1st Qu.:21848765   1st Qu.:2019-01-23 05:26:54.00
##  Median :21961829   Median :2019-02-25 07:52:56.00
##  Mean   :21960872   Mean   :2019-02-19 21:43:15.42
##  3rd Qu.:22071823   3rd Qu.:2019-03-17 16:52:47.00
##  Max.   :22178528   Max.   :2019-03-31 23:53:48.00
##  NA's   :3879822    NA's   :3879822
##     end_time                       bikeid        from_station_id
##  Min.   :2019-01-01 00:11:07.00   Min.   :   1      Min.   :  2
##  1st Qu.:2019-01-23 05:49:40.00   1st Qu.:1777      1st Qu.: 76
##  Median :2019-02-25 08:03:50.00   Median :3489      Median :170
##  Mean   :2019-02-19 22:00:11.91   Mean   :3429      Mean   :198
##  3rd Qu.:2019-03-17 17:16:16.00   3rd Qu.:5157      3rd Qu.:287
##  Max.   :2019-06-17 16:04:35.00   Max.   :6471      Max.   :665
##  NA's   :3879822                  NA's   :3879822   NA's   :3879822
##  from_station_name  to_station_id     to_station_name      usertype
##  Length:4244891     Min.   :  2       Length:4244891     Length:4244891
##  Class :character   1st Qu.: 76       Class :character   Class :character
##  Mode  :character   Median :168       Mode  :character   Mode  :character
##                     Mean   :199
##                     3rd Qu.:287
##                     Max.   :665
##                     NA's   :3879822
##    ride_id            started_at
##  Length:4244891     Min.   :2019-04-01 00:02:22.00
##  Class :character   1st Qu.:2019-06-23 07:49:09.25
##  Mode  :character   Median :2019-08-14 17:43:38.00
##                     Mean   :2019-08-26 00:49:59.38
##                     3rd Qu.:2019-10-12 12:10:21.00
##                     Max.   :2020-03-31 23:51:34.00
##                     NA's   :365069
##     ended_at                      rideable_type       start_station_id
##  Min.   :2019-04-01 00:09:48.00   Length:4244891     Min.   :  1.0
##  1st Qu.:2019-06-23 08:20:27.75   Class :character   1st Qu.: 77.0
##  Median :2019-08-14 18:02:04.00   Mode  :character   Median :174.0
##  Mean   :2019-08-26 01:14:37.06                      Mean   :202.9
##  3rd Qu.:2019-10-12 12:36:16.75                      3rd Qu.:291.0
##  Max.   :2020-05-19 20:10:34.00                      Max.   :675.0
##  NA's   :365069                                      NA's   :365069
##  start_station_name end_station_id   end_station_name   member_casual
##  Length:4244891     Min.   :  1.0    Length:4244891     Length:4244891
##  Class :character   1st Qu.: 77.0    Class :character   Class :character
```

```
##  Mode  :character   Median :174.0    Mode  :character   Mode  :character
##                     Mean   :203.8
##                     3rd Qu.:291.0
##                     Max.   :675.0
##                     NA's   :365070
```

There are a few problems we will need to fix:

(1) In the “member_casual” column, there are two names for members (“member” and “Subscriber”) and two names for casual riders (“Customer” and “casual”). Consolidate that from four to two labels.

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

(3) Add a calculated field for length of ride since the 2020Q1, data did not have the “tripduration” column. Add “ride_length” to the entire dataframe for consistency.

(4) There are some rides where tripduration shows up as negative, including several hundred rides where Divvy took bikes out of circulation for Quality Control reasons. So delete these rides.

In the “member_casual” column, replace “Subscriber” with “member” and “Customer” with “casual”

Before 2020, Divvy used different labels for these two types of riders …Make our data frame consistent with their current nomenclature

N.B.: “Level” is a special property of a column that is retained even if a subset does not contain any values from a specific level

See how many observations fall under each usertype

```
table(all_trips$member_casual)
```

```
##
##    casual   Customer     member Subscriber
##     48480     857474     378407    2595461
```

Reassign to the desired values (Going with the current 2020 labels)

```
all_trips <-  all_trips %>%
  mutate(member_casual = recode(member_casual
                                ,"Subscriber" = "member"
                                ,"Customer" = "casual"))
```

Check to make sure the proper number of observations were reassigned

```
table(all_trips$member_casual)
```

```
##
##  casual  member
##  905954 2973868
```

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

This will aggregate ride data for each month, day, or year … before completing these operations we could only aggregate at the ride level

https://www.statmethods.net/input/dates.html more on date formats in R found at that link

```
all_trips$date <- as.Date(all_trips$started_at) #The default format is yyyy-mm-dd
all_trips$month <- format(as.Date(all_trips$date), "%m")
all_trips$day <- format(as.Date(all_trips$date), "%d")
all_trips$year <- format(as.Date(all_trips$date), "%Y")
all_trips$day_of_week <- format(as.Date(all_trips$date), "%A")
```

Add a “ride_length” calculation to all_trips (in seconds)

https://stat.ethz.ch/R-manual/R-devel/library/base/html/difftime.html

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

Inspect the structure of the columns

```
str(all_trips)
```

```
## tibble [4,244,891 × 24] (S3: tbl_df/tbl/data.frame)
##  $ trip_id          : num [1:4244891] 21742443 21742444 21742445 21742446 21742447 ...
##  $ start_time       : POSIXct[1:4244891], format: "2019-01-01 00:04:37" "2019-01-01 00:08:13" ...
##  $ end_time         : POSIXct[1:4244891], format: "2019-01-01 00:11:07" "2019-01-01 00:15:34" ...
##  $ bikeid           : num [1:4244891] 2167 4386 1524 252 1170 ...
##  $ from_station_id  : num [1:4244891] 199 44 15 123 173 98 98 211 150 268 ...
##  $ from_station_name : chr [1:4244891] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine
Ave & 18th St" "California Ave & Milwaukee Ave" ...
##  $ to_station_id    : num [1:4244891] 84 624 644 176 35 49 49 142 148 141 ...
##  $ to_station_name  : chr [1:4244891] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)"
"Western Ave & Fillmore St (*)" "Clark St & Elm St" ...
##  $ usertype         : chr [1:4244891] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
##  $ ride_id          : chr [1:4244891] NA NA NA NA ...
##  $ started_at       : POSIXct[1:4244891], format: NA NA ...
##  $ ended_at         : POSIXct[1:4244891], format: NA NA ...
##  $ rideable_type    : chr [1:4244891] NA NA NA NA ...
##  $ start_station_id  : num [1:4244891] NA NA NA NA NA NA NA NA NA NA ...
##  $ start_station_name: chr [1:4244891] NA NA NA NA ...
##  $ end_station_id   : num [1:4244891] NA NA NA NA NA NA NA NA NA NA ...
##  $ end_station_name  : chr [1:4244891] NA NA NA NA ...
##  $ member_casual    : chr [1:4244891] NA NA NA NA ...
##  $ date             : Date[1:4244891], format: NA NA ...
##  $ month            : chr [1:4244891] NA NA NA NA ...
##  $ day              : chr [1:4244891] NA NA NA NA ...
##  $ year             : chr [1:4244891] NA NA NA NA ...
##  $ day_of_week      : chr [1:4244891] NA NA NA NA ...
##  $ ride_length      : 'difftime' num [1:4244891] NA NA NA NA ...
##   ..- attr(*, "units")= chr "secs"
```

Convert “ride_length” from Factor to numeric then run calculations on the data

```
is.factor(all_trips$ride_length)
```

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

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

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

Remove “bad” data

The data frame includes a few hundred entries when bikes were taken out of docks and checked for quality by Divvy or ride_length was negative

Create a new version of the dataframe (v2) since data is being removed

https://www.datasciencemadesimple.com/delete-or-drop-rows-in-r-with-conditions-2/

```
all_trips_v2 <- all_trips[!(all_trips$start_station_name == "HQ QR" | all_trips$ride_length<0),]
```

*ANALYZE

All the required information is now in one place and ready for exploration.

Key tasks

Aggregate your data so it’s useful and accessible.

Organize and format your data.

Perform calculations.

Identify trends and relationships.

Deliverable

A summary of the analysis.

4. A summary of your analysis

```
#=====================================
# STEP 4: CONDUCT DESCRIPTIVE ANALYSIS
#=====================================
```

```
mean(all_trips_v2$ride_length) #straight average (total ride length / rides)
```

```
## [1] NA
```

```
median(all_trips_v2$ride_length) #midpoint number in the ascending array of ride lengths
```

```
## [1] NA
```

```
max(all_trips_v2$ride_length) #longest ride
```

```
## [1] NA
```

```
min(all_trips_v2$ride_length) #shortest ride
```

```
## [1] NA
```

Condense the four lines above to one line using summary() on the specific attribute

```
summary(all_trips_v2$ride_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##       1     412     712    1479    1289 9387024  365069
```

Compare members and casual users

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = mean)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                3552.7502
## 2                     member                 850.0662
```

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = median)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                    1546
## 2                     member                     589
```

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = max)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                 9387024
## 2                     member                 9056634
```

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = min)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                       2
## 2                     member                       1
```

See the average ride time by each day for members vs casual users

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FUN = mean)
```

```
##    all_trips_v2$member_casual all_trips_v2$day_of_week all_trips_v2$ride_length
## 1                      casual                   Friday                3773.8351
## 2                      member                   Friday                 824.5305
## 3                      casual                   Monday                3372.2869
## 4                      member                   Monday                 842.5726
## 5                      casual                 Saturday                3331.9138
## 6                      member                 Saturday                 968.9337
## 7                      casual                   Sunday                3581.4054
## 8                      member                   Sunday                 919.9746
## 9                      casual                 Thursday                3682.9847
## 10                     member                 Thursday                 823.9278
## 11                     casual                  Tuesday                3596.3599
## 12                     member                  Tuesday                 826.1427
## 13                     casual                Wednesday                3718.6619
## 14                     member                Wednesday                 823.9996
```

Notice that the days of the week are out of order. Let's fix that.

```
all_trips_v2$day_of_week <- ordered(all_trips_v2$day_of_week, levels=c("Sunday", "Monday", "Tuesday",
"Wednesday", "Thursday", "Friday", "Saturday"))
```

Now, let's run the average ride time by each day for members vs casual users

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FUN =
mean)
```

```
##    all_trips_v2$member_casual all_trips_v2$day_of_week all_trips_v2$ride_length
## 1                      casual                   Sunday                3581.4054
## 2                      member                   Sunday                 919.9746
## 3                      casual                   Monday                3372.2869
## 4                      member                   Monday                 842.5726
## 5                      casual                  Tuesday                3596.3599
## 6                      member                  Tuesday                 826.1427
## 7                      casual                Wednesday                3718.6619
## 8                      member                Wednesday                 823.9996
## 9                      casual                 Thursday                3682.9847
## 10                     member                 Thursday                 823.9278
## 11                     casual                   Friday                3773.8351
## 12                     member                   Friday                 824.5305
## 13                     casual                 Saturday                3331.9138
## 14                     member                 Saturday                 968.9337
```

analyze ridership data by type and weekday

```
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%  #creates weekday field using wday()
  group_by(member_casual, weekday) %>%  #groups by usertype and weekday
  summarise(number_of_rides = n()                       #calculates the number of rides and average duration
            ,average_duration = mean(ride_length)) %>%      # calculates the average duration
  arrange(member_casual, weekday)                           # sorts
```

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

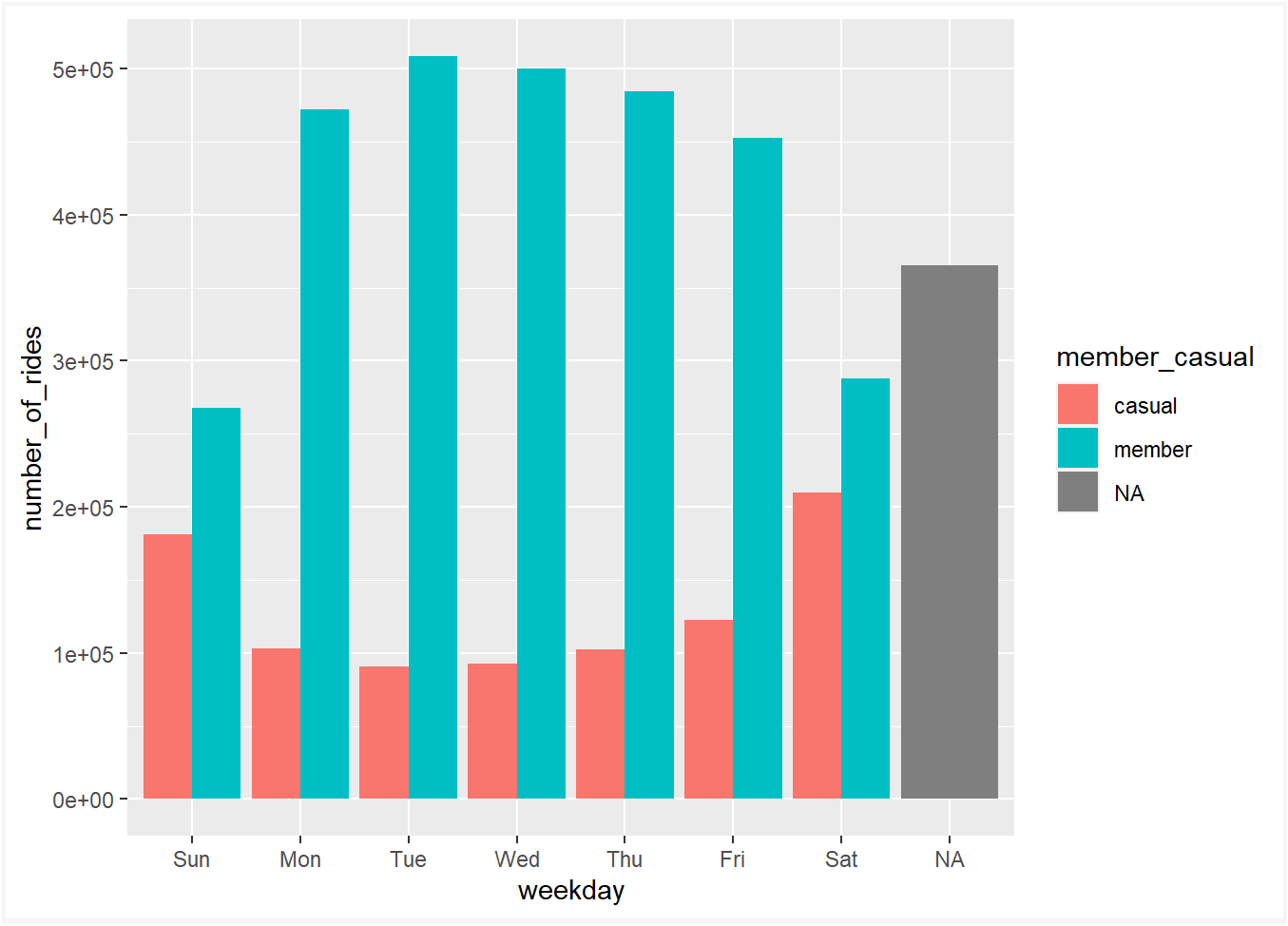
```
## # A tibble: 15 × 4
```

```
## # Groups:   member_casual [3]
##   member_casual weekday number_of_rides average_duration
##   <chr>         <ord>           <int>            <dbl>
##  1 casual       Sun            181293            3581.
##  2 casual       Mon            103296            3372.
##  3 casual       Tue             90510            3596.
##  4 casual       Wed             92457            3719.
##  5 casual       Thu            102679            3683.
##  6 casual       Fri            122404            3774.
##  7 casual       Sat            209543            3332.
##  8 member       Sun            267965             920.
##  9 member       Mon            472196             843.
## 10 member       Tue            508445             826.
## 11 member       Wed            500329             824.
## 12 member       Thu            484177             824.
## 13 member       Fri            452790             825.
## 14 member       Sat            287958             969.
## 15 <NA>         <NA>           365069              NA
```

Let’s visualize the number of rides by rider type

```
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = weekday, y = number_of_rides, fill = member_casual)) +
  geom_col(position = "dodge")
```

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

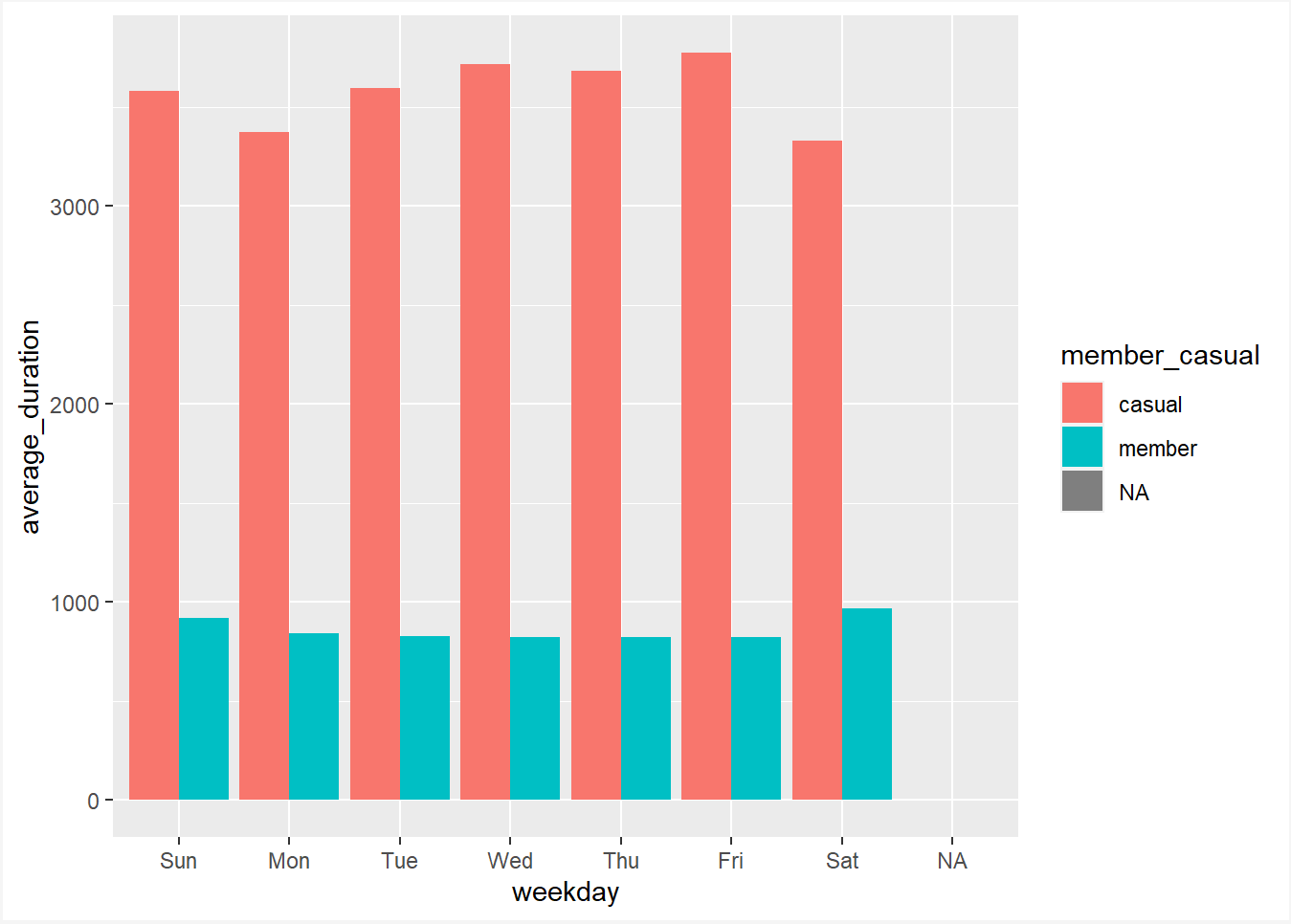


Let’s create a visualization for average duration

```
all_trips_v2 %>%
 mutate(weekday = wday(started_at, label = TRUE)) %>%
 group_by(member_casual, weekday) %>%
 summarise(number_of_rides = n()
           ,average_duration = mean(ride_length)) %>%
 arrange(member_casual, weekday)  %>%
 ggplot(aes(x = weekday, y = average_duration, fill = member_casual)) +
 geom_col(position = "dodge")
```

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

Warning: Removed 1 rows containing missing values (`geom_col()`).



– From the above graph, we can observe that casual riders went a longer distance compared to members.

#==========

STEP 5: EXPORT SUMMARY FILE FOR FURTHER ANALYSIS

#==========

Create a csv file that we will visualize in Tableau.

```
counts <- aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FUN = mean)

write.csv(counts, file = "avg_ride_length.csv")
```

*SHARE

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

5. Supporting visualizations and key findings

Question: How do annual members and casual riders use Cyclistic bikes differently?

—> Conclusions:

- Casual riders go farther in terms of distance.

- Casual riders travel for a longer time period.

- Members ride less on the weekend compared to casual riders.

*ACT

This phase will be carried out by the executive team, Director of Marketing (Lily Moreno) and the Marketing Analytics team

based on my analysis, Conclusions.

6. Your top three recommendations based on your analysis

- Host fun biking competitions with prizes for members only on weekends. Since casual riders will be attracted by competition prizes.

- By combining Membership benefits with a limited time discount upon signing up. Or get extra 3 months upon signing up within 3 weeks.

- Advertise by cycling you can burn your calories faster. “Cyclistic Bikes: The Right way to Diet”.

THANK YOU! FOR READING, PLEASE PROVIDE YOUR VALUABLE FEEDBACK.

#==========