```r
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   1.5.0
## ✓ ggplot2   3.5.0     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ 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
```

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

# Set dplyr::filter and dplyr::lag as the default choices
conflict_prefer("filter", "dplyr")
```

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

```r
conflict_prefer("lag", "dplyr")
```

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

```r
#=====================
# STEP 1: COLLECT DATA
#=====================
# # Upload Divvy datasets (csv files) here
q1_2019 <- read_csv("/Users/yashds/Downloads/CASE STUDY 1 R
Code/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.
```

```r
q1_2020 <- read_csv("/Users/yashds/Downloads/CASE STUDY 1 R
Code/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
# 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(q1_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"

# Rename columns  to make them consistent with q1_2020 (as this will be the
supposed going-forward table design for Divvy)

(q1_2019 <- rename(q1_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: 365,069 × 12
##     ride_id started_at          ended_at            rideable_type
tripduration
##       <dbl> <dttm>              <dttm>                      <dbl>
<dbl>
##  1 21742443 2019-01-01 00:04:37 2019-01-01 00:11:07          2167
390
##  2 21742444 2019-01-01 00:08:13 2019-01-01 00:15:34          4386
441
##  3 21742445 2019-01-01 00:13:23 2019-01-01 00:27:12          1524
829
##  4 21742446 2019-01-01 00:13:45 2019-01-01 00:43:28           252
1783
##  5 21742447 2019-01-01 00:14:52 2019-01-01 00:20:56          1170
364
##  6 21742448 2019-01-01 00:15:33 2019-01-01 00:19:09          2437
216
##  7 21742449 2019-01-01 00:16:06 2019-01-01 00:19:03          2708
177
##  8 21742450 2019-01-01 00:18:41 2019-01-01 00:20:21          2796
100
##  9 21742451 2019-01-01 00:18:43 2019-01-01 00:47:30          6205
1727
## 10 21742452 2019-01-01 00:19:18 2019-01-01 00:24:54          3939
336
## # ℹ 365,059 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>
```

```
# Inspect the dataframes and look for incongruencies
str(q1_2019)
```

```
## spc_tbl_ [365,069 × 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : num [1:365069] 21742443 21742444 21742445 21742446
21742447 ...
##  $ started_at        : POSIXct[1:365069], format: "2019-01-01 00:04:37"
"2019-01-01 00:08:13" ...
##  $ ended_at          : POSIXct[1:365069], format: "2019-01-01 00:11:07"
"2019-01-01 00:15:34" ...
##  $ rideable_type     : num [1:365069] 2167 4386 1524 252 1170 ...
##  $ tripduration      : num [1:365069] 390 441 829 1783 364 ...
##  $ start_station_id  : num [1:365069] 199 44 15 123 173 98 98 211 150 268
...
##  $ start_station_name: chr [1:365069] "Wabash Ave & Grand Ave" "State St &
Randolph St" "Racine Ave & 18th St" "California Ave & Milwaukee Ave" ...
##  $ end_station_id    : num [1:365069] 84 624 644 176 35 49 49 142 148 141
...
##  $ end_station_name  : chr [1:365069] "Milwaukee Ave & Grand Ave"
"Dearborn St & Van Buren St (*)" "Western Ave & Fillmore St (*)" "Clark St &
```

```
Elm St" ...
##  $ member_casual     : 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>
```

```
str(q1_2020)
```

```
## 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>

# Convert ride_id and rideable_type to character so that they can stack
correctly
q1_2019 <-  mutate(q1_2019, 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(q1_2019, q1_2020)#, q3_2019)#, q4_2019, q1_2020)

# 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,
"tripduration"))

#=====================================================
# 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] "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_trips)  #How many rows are in data frame?

## [1] 791956

dim(all_trips)  #Dimensions of the data frame?

## [1] 791956      9

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

## # A tibble: 6 × 9
##   ride_id started_at          ended_at            rideable_type
```

```
start_station_id
##   <chr>   <dttm>              <dttm>              <chr>
<dbl>
## 1 217424… 2019-01-01 00:04:37 2019-01-01 00:11:07 2167
199
## 2 217424… 2019-01-01 00:08:13 2019-01-01 00:15:34 4386
44
## 3 217424… 2019-01-01 00:13:23 2019-01-01 00:27:12 1524
15
## 4 217424… 2019-01-01 00:13:45 2019-01-01 00:43:28 252
123
## 5 217424… 2019-01-01 00:14:52 2019-01-01 00:20:56 1170
173
## 6 217424… 2019-01-01 00:15:33 2019-01-01 00:19:09 2437
98
## # ℹ 4 more variables: start_station_name <chr>, end_station_id <dbl>,
## #   end_station_name <chr>, member_casual <chr>
```

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

```
## tibble [791,956 × 9] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:791956] "21742443" "21742444" "21742445"
"21742446" ...
##  $ started_at        : POSIXct[1:791956], format: "2019-01-01 00:04:37"
"2019-01-01 00:08:13" ...
##  $ ended_at          : POSIXct[1:791956], format: "2019-01-01 00:11:07"
"2019-01-01 00:15:34" ...
##  $ rideable_type     : chr [1:791956] "2167" "4386" "1524" "252" ...
##  $ start_station_id  : num [1:791956] 199 44 15 123 173 98 98 211 150 268
...
##  $ start_station_name: chr [1:791956] "Wabash Ave & Grand Ave" "State St &
Randolph St" "Racine Ave & 18th St" "California Ave & Milwaukee Ave" ...
##  $ end_station_id    : num [1:791956] 84 624 644 176 35 49 49 142 148 141
...
##  $ end_station_name  : chr [1:791956] "Milwaukee Ave & Grand Ave"
"Dearborn St & Van Buren St (*)" "Western Ave & Fillmore St (*)" "Clark St &
Elm St" ...
##  $ member_casual     : chr [1:791956] "Subscriber" "Subscriber"
"Subscriber" "Subscriber" ...
```

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

```
##    ride_id            started_at
##  Length:791956      Min.   :2019-01-01 00:04:37.00
##  Class :character   1st Qu.:2019-02-28 17:04:04.75
##  Mode  :character   Median :2020-01-07 12:48:50.50
##                     Mean   :2019-09-01 11:58:08.35
##                     3rd Qu.:2020-02-19 19:31:54.75
##                     Max.   :2020-03-31 23:51:34.00
##
##     ended_at                      rideable_type      start_station_id
```

```
##  Min.   :2019-01-01 00:11:07.00   Length:791956      Min.   :  2.0
##  1st Qu.:2019-02-28 17:15:58.75   Class :character   1st Qu.: 77.0
##  Median :2020-01-07 13:02:50.00   Mode  :character   Median :174.0
##  Mean   :2019-09-01 12:17:52.17                      Mean   :204.4
##  3rd Qu.:2020-02-19 19:51:54.50                      3rd Qu.:291.0
##  Max.   :2020-05-19 20:10:34.00                      Max.   :675.0
##
##  start_station_name end_station_id  end_station_name   member_casual
##  Length:791956      Min.   :  2.0   Length:791956      Length:791956
##  Class :character   1st Qu.: 77.0   Class :character   Class :character
##  Mode  :character   Median :174.0   Mode  :character   Mode  :character
##                     Mean   :204.4
##                     3rd Qu.:291.0
##                     Max.   :675.0
##                     NA's   :1
```

```r
# 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"). We will need to consolidate that from four to two labels.
# (2) The data can only be aggregated at the ride-level, which is too
granular. We will want to add some additional columns of data -- such as day,
month, year -- that provide additional opportunities to aggregate the data.
# (3) We will want to add a calculated field for length of ride since the
2020Q1 data did not have the "tripduration" column. We will 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. We will want to 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 ...
we will want to make our dataframe 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
# Begin by seeing how many observations fall under each usertype
table(all_trips$member_casual)
```

```
##
##     casual   Customer     member Subscriber
##      48480      23163     378407     341906
```

```r
library(dplyr)

# Reassign to the desired values (we will go with the current 2020 labels)
all_trips <- all_trips %>%
  mutate(member_casual = dplyr::recode(member_casual,
                                       "Subscriber" = "member",
                                       "Customer" = "casual"))
```

```r
# Check to make sure the proper number of observations were reassigned
table(all_trips$member_casual)

##
## casual member
##  71643 720313

# Add columns that list the date, month, day, and year of each ride
# This will allow us to 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 [791,956 × 15] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:791956] "21742443" "21742444" "21742445" "21742446" ...
##  $ started_at        : POSIXct[1:791956], format: "2019-01-01 00:04:37" "2019-01-01 00:08:13" ...
##  $ ended_at          : POSIXct[1:791956], format: "2019-01-01 00:11:07" "2019-01-01 00:15:34" ...
##  $ rideable_type     : chr [1:791956] "2167" "4386" "1524" "252" ...
##  $ start_station_id  : num [1:791956] 199 44 15 123 173 98 98 211 150 268 ...
##  $ start_station_name: chr [1:791956] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave & 18th St" "California Ave & Milwaukee Ave" ...
##  $ end_station_id    : num [1:791956] 84 624 644 176 35 49 49 142 148 141 ...
##  $ end_station_name  : chr [1:791956] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)" "Western Ave & Fillmore St (*)" "Clark St & Elm St" ...
##  $ member_casual     : chr [1:791956] "member" "member" "member" "member" ...
##  $ date              : Date[1:791956], format: "2019-01-01" "2019-01-01" ...
##  $ month             : chr [1:791956] "01" "01" "01" "01" ...
##  $ day               : chr [1:791956] "01" "01" "01" "01" ...
##  $ year              : chr [1:791956] "2019" "2019" "2019" "2019" ...
##  $ day_of_week       : chr [1:791956] "Tuesday" "Tuesday" "Tuesday"
```

```
"Tuesday" ...
##  $ ride_length       : 'difftime' num [1:791956] 390 441 829 1783 ...
##   ..- attr(*, "units")= chr "secs"

# Convert "ride_length" from Factor to numeric so we can 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 dataframe includes a few hundred entries when bikes were taken out of
docks and checked for quality by Divvy or ride_length was negative
# We will 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),]

#=====================================
# STEP 4: CONDUCT DESCRIPTIVE ANALYSIS
#=====================================
# Descriptive analysis on ride_length (all figures in seconds)
mean(all_trips_v2$ride_length) #straight average (total ride length / rides)

## [1] 1189.459

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

## [1] 539

max(all_trips_v2$ride_length) #longest ride

## [1] 10632022

min(all_trips_v2$ride_length) #shortest ride

## [1] 1

# You can 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.
##        1      331      539     1189      912 10632022
```

```r
# 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                5372.7839
## 2                     member                 795.2523
```

```r
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                      1393
## 2                     member                       508
```

```r
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                10632022
## 2                     member                 6096428
```

```r
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
```

```r
# 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
6090.7373
## 2                      member                   Friday
796.7338
## 3                      casual                   Monday
4752.0504
## 4                      member                   Monday
822.3112
## 5                      casual                 Saturday
4950.7708
## 6                      member                 Saturday
974.0730
## 7                      casual                   Sunday
5061.3044
## 8                      member                   Sunday
972.9383
## 9                      casual                 Thursday
8451.6669
## 10                     member                 Thursday
707.2093
```

```
## 11                      casual                    Tuesday
4561.8039
## 12                      member                    Tuesday
769.4416
## 13                      casual                  Wednesday
4480.3724
## 14                      member                  Wednesday
711.9838
```

```r
# 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
5061.3044
## 2                      member                    Sunday
972.9383
## 3                      casual                    Monday
4752.0504
## 4                      member                    Monday
822.3112
## 5                      casual                   Tuesday
4561.8039
## 6                      member                   Tuesday
769.4416
## 7                      casual                 Wednesday
4480.3724
## 8                      member                 Wednesday
711.9838
## 9                      casual                  Thursday
8451.6669
## 10                     member                  Thursday
707.2093
## 11                     casual                    Friday
6090.7373
## 12                     member                    Friday
796.7338
## 13                     casual                  Saturday
4950.7708
## 14                     member                  Saturday
974.0730
```

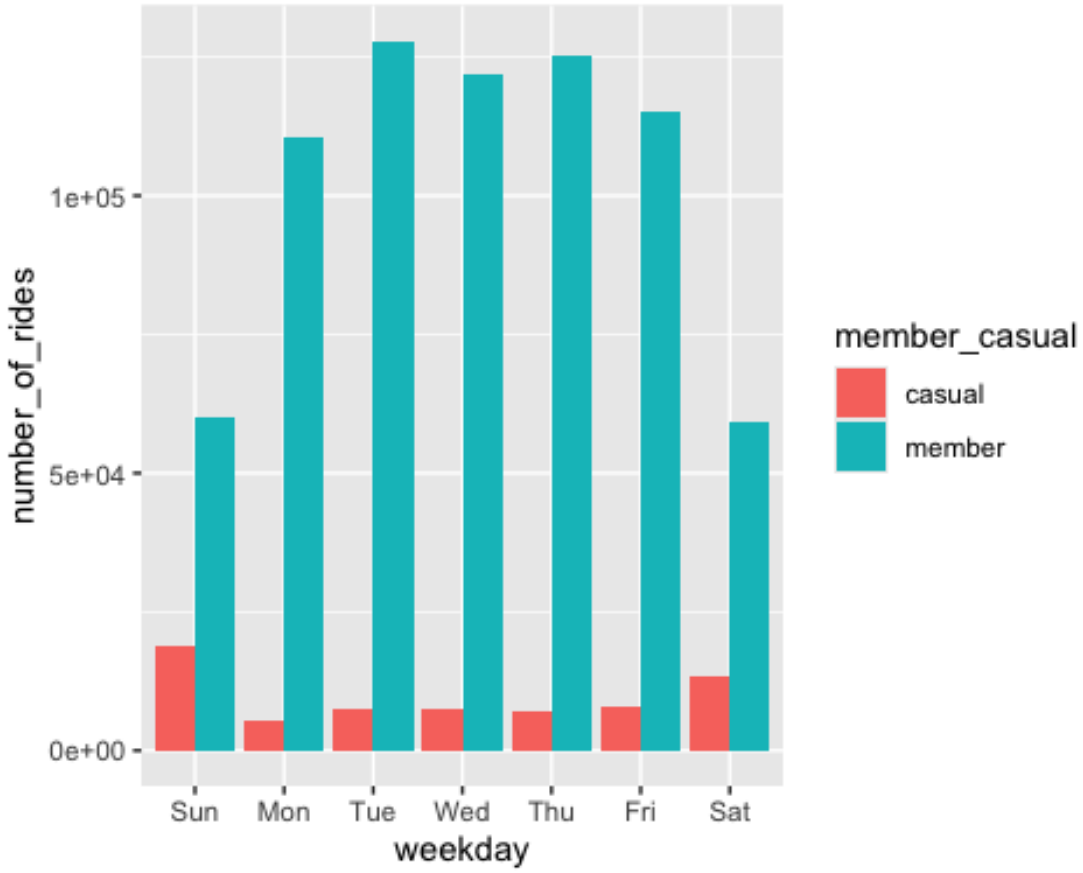
```r
# analyze ridership data by type and weekday
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%  #creates weekday
field using wday()
  group_by(member_casual, weekday) %>%  #groups by usertype and weekday
  summarise(number_of_rides = n()                              #calculates 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: 14 × 4
## # Groups:   member_casual [2]
##    member_casual weekday number_of_rides average_duration
##    <chr>         <ord>             <int>            <dbl>
##  1 casual        Sun               18652            5061.
##  2 casual        Mon                5591            4752.
##  3 casual        Tue                7311            4562.
##  4 casual        Wed                7690            4480.
##  5 casual        Thu                7147            8452.
##  6 casual        Fri                8013            6091.
##  7 casual        Sat               13473            4951.
##  8 member        Sun               60197             973.
##  9 member        Mon              110430             822.
## 10 member        Tue              127974             769.
## 11 member        Wed              121902             712.
## 12 member        Thu              125228             707.
## 13 member        Fri              115168             797.
## 14 member        Sat               59413             974.
```

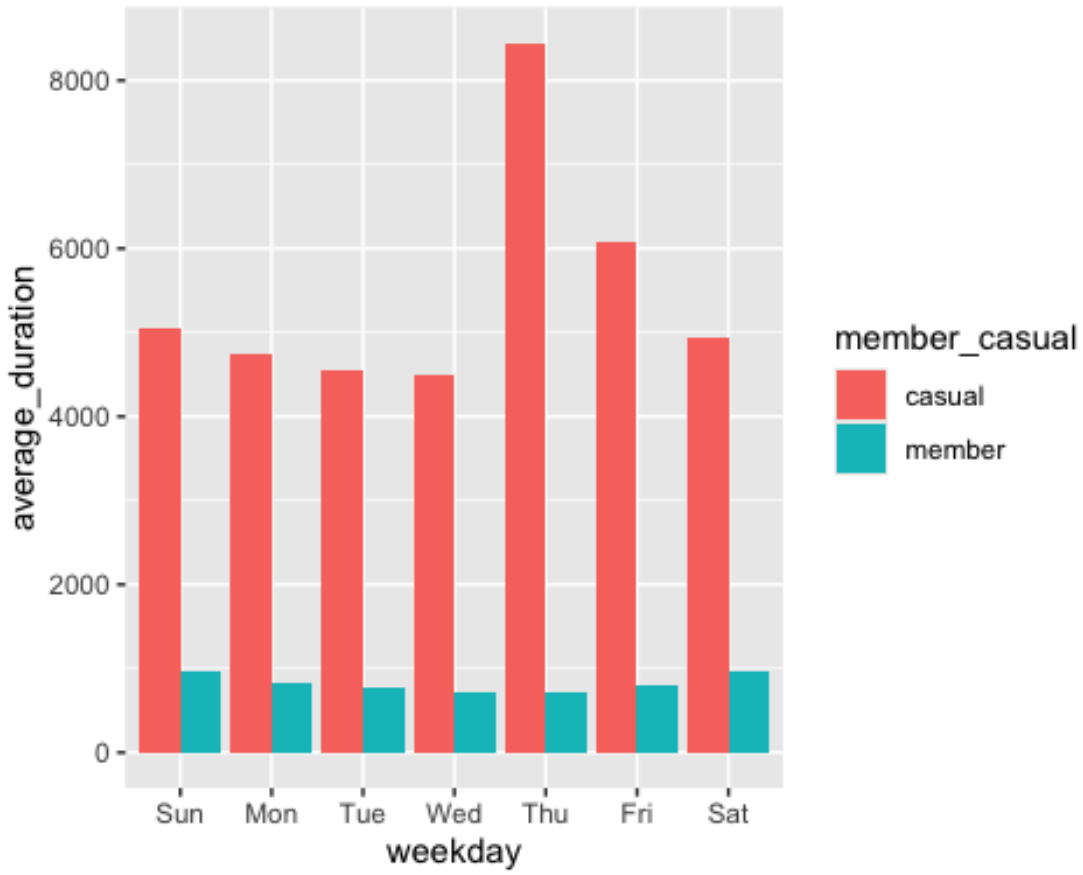
```r
# Let's visualize the number of rides by rider type
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = 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.
```




```
#=============================================
# STEP 5: EXPORT SUMMARY FILE FOR FURTHER ANALYSIS
#=============================================

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

counts

##    all_trips_v2$member_casual all_trips_v2$day_of_week
all_trips_v2$ride_length
## 1                      casual                   Sunday
5061.3044
## 2                      member                   Sunday
972.9383
## 3                      casual                   Monday
4752.0504
## 4                      member                   Monday
822.3112
## 5                      casual                  Tuesday
4561.8039
## 6                      member                  Tuesday
769.4416
```

```
## 7                      casual               Wednesday
4480.3724
## 8                      member               Wednesday
711.9838
## 9                      casual                Thursday
8451.6669
## 10                     member                Thursday
707.2093
## 11                     casual                  Friday
6090.7373
## 12                     member                  Friday
796.7338
## 13                     casual                Saturday
4950.7708
## 14                     member                Saturday
974.0730
```