Cyclistic2019AnnualReport

Diego Martinez

2022-06-24

Cyclistic Case Study

STEP 1: EXCEL PREPARATION AND LOADING LIBRARIES/DATA IN R

Thd following section describes the changes I performed on Excel. The main objective of this section is to show my proficiency in Excel, for all other purposes, this section may be skipped.

As secondary objective, yet just as important, it is to ensure the documentation of processes.

First, I opened the data containing all bicycle trips on 2019 by our fictional company Cyclistic. In this case, it is on four .cvs files, dividing the year in four quarters. As my first tool, I use Excel to open this files. Even before interacting with the data, I make sure to load each column on the correct data type. After a swift check for any major discrepancies, I did data validation on all columns: check for duplicates, made sure all values were valid and the correct type.

Aside from some null cells, the only column that needed significant changes was the trip duration column, which in some of the quarters, is less than zero. After some light research, I found out these were test drives made by the real company "Cyclist" gets its data from. We will deal with this later on.

The first quarter has a trip duration column which is pretty useful but absent on the subsequent quarters, so I took it off for consistency.I calculate trip duration by subtracting the end and start times and rename it to ride_length.

I will perform the same calculation later on in R because this case study is to show off some of skills after all. The first quarter also had a different date format so I took note of that for future calculations. I also dropped the ride_length column because it is in seconds and I plan to calculate ride_length in hours:minutes:seconds format; having another column for seconds would be redundant

In quarter 2 and 4, the data was so large it wouldn't fit the sheet and as a result data was missing. I imported the missing columns in a new spreadsheet to investigate. Q2 YMD So big it might not fit spreadsheet, show how you avoid that. Imported on a new spreadsheet, it only had one more trip on 6-27. Normally, I want to use all the valid data available to make the best prediction possible, but since this is a fictional company, and I already checked for duplicate, I will leave the last trip out of the Excel analysis part.

Quarter 3 was even less eventful and had less than minor modifications

Now in R

Let's set our directory

Installing all necessary libraries:

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
```

```
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
## [1] "/Users/diegofarela/Desktop/Cyclystic_2019byQuarter"
```

Loading all our data

```
q1_2019 <- read_csv("C_2019_Q1.csv")
```

```
## Rows: 365069 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): start_time, end_time, ride_length, from_station_name, to_station_na...
## dbl (6): trip_id, bikeid, day_of_week, from_station_id, to_station_id, birth...
##
## 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.
```

```
q2_2019 <- read_csv("C_2019_Q2.csv")
```

```
## Rows: 1048575 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): 01 - Rental Details Local Start Time, 01 - Rental Details Local End...
## dbl (6): 01 - Rental Details Rental ID, day_of_week, 01 - Rental Details Bik...
##
## 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.
```

```
q3_2019 <- read_csv("C_2019_Q3.csv")
```

```
## Rows: 1048575 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): start_time, end_time, ride_length, from_station_name, to_station_na...
## dbl (6): trip_id, bikeid, day_of_week, from_station_id, to_station_id, birth...
##
## 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.
```

```
q4_2019 <- read_csv("C_2019_Q4.csv")
```

```
## Rows: 704054 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): start_time, end_time, ride_length, from_station_name, to_station_na...
## dbl (6): trip_id, bikeid, day_of_week, from_station_id, to_station_id, birth...
##
```

```
## 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"            "ride_length"       "day_of_week"
##  [7] "from_station_id"   "from_station_name" "to_station_id"
## [10] "to_station_name"   "usertype"          "gender"
## [13] "birthyear"
```

```
colnames(q2_2019)
```

```
##  [1] "01 - Rental Details Rental ID"
##  [2] "01 - Rental Details Local Start Time"
##  [3] "01 - Rental Details Local End Time"
##  [4] "ride_length"
##  [5] "day_of_week"
##  [6] "01 - Rental Details Bike ID"
##  [7] "03 - Rental Start Station ID"
##  [8] "03 - Rental Start Station Name"
##  [9] "02 - Rental End Station ID"
## [10] "02 - Rental End Station Name"
## [11] "User Type"
## [12] "Member Gender"
## [13] "05 - Member Details Member Birthday Year"
```

```
colnames(q3_2019)
```

```
##  [1] "trip_id"           "start_time"        "end_time"
##  [4] "bikeid"            "ride_length"       "day_of_week"
##  [7] "from_station_id"   "from_station_name" "to_station_id"
## [10] "to_station_name"   "usertype"          "gender"
## [13] "birthyear"
```

```
colnames(q4_2019)
```

```
##  [1] "trip_id"           "start_time"        "end_time"
##  [4] "bikeid"            "ride_length"       "day_of_week"
##  [7] "from_station_id"   "from_station_name" "to_station_id"
## [10] "to_station_name"   "usertype"          "gender"
## [13] "birthyear"
```

Let's rename our columns/variables to make them consistent

```
(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
                  ,gender = gender
                  ,birthyear = birthyear))
```

```
## # A tibble: 704,054 x 13
##     ride_id started_at   ended_at     rideable_type ride_length day_of_week
##       <dbl> <chr>        <chr>                <dbl> <chr>              <dbl>
##  1 25223640 10/1/19 0:01 10/1/19 0:17          2215 0:15:41                3
##  2 25223641 10/1/19 0:02 10/1/19 0:06          6328 1:15:41                3
##  3 25223642 10/1/19 0:04 10/1/19 0:18          3003 2:15:41                3
##  4 25223643 10/1/19 0:04 10/1/19 0:43          3275 3:15:41                3
##  5 25223644 10/1/19 0:04 10/1/19 0:35          5294 4:15:41                3
##  6 25223645 10/1/19 0:04 10/1/19 0:10          1891 5:15:41                3
##  7 25223646 10/1/19 0:04 10/1/19 0:22          1061 6:15:41                3
##  8 25223647 10/1/19 0:04 10/1/19 0:29          1274 7:15:41                3
##  9 25223648 10/1/19 0:05 10/1/19 0:29          6011 8:15:41                3
## 10 25223649 10/1/19 0:05 10/1/19 2:23          2957 9:15:41                3
## # ... with 704,044 more rows, and 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>
```

```
(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
                  ,gender = gender
                  ,birthyear = birthyear))
```

```
## # A tibble: 1,048,575 x 13
##     ride_id started_at  ended_at    rideable_type ride_length day_of_week
##       <dbl> <chr>       <chr>               <dbl> <chr>              <dbl>
##  1 23479388 7/1/19 0:00 7/1/19 0:20          3591 0:20:14                2
##  2 23479389 7/1/19 0:01 7/1/19 0:18          5353 0:17:28                2
##  3 23479390 7/1/19 0:01 7/1/19 0:27          6180 0:25:54                2
##  4 23479391 7/1/19 0:02 7/1/19 0:27          5540 0:25:03                2
##  5 23479392 7/1/19 0:02 7/1/19 0:22          6014 0:20:13                2
##  6 23479393 7/1/19 0:02 7/1/19 0:07          4941 0:05:10                2
##  7 23479394 7/1/19 0:02 7/1/19 0:23          3770 0:20:48                2
##  8 23479395 7/1/19 0:02 7/1/19 0:28          5442 0:25:50                2
##  9 23479396 7/1/19 0:02 7/1/19 0:28          2957 0:26:23                2
## 10 23479397 7/1/19 0:02 7/1/19 0:29          6091 0:26:29                2
## # ... with 1,048,565 more rows, and 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>
```

```
(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"
                  ,gender = "Member Gender"
                  ,birthyear = "05 - Member Details Member Birthday Year"))
```

```
## # A tibble: 1,048,575 x 13
##     ride_id started_at  ended_at    ride_length day_of_week rideable_type
##       <dbl> <chr>       <chr>       <chr>              <dbl>         <dbl>
##  1 22178529 4/1/19 0:02 4/1/19 0:09 0:07:26                2          6251
##  2 22178530 4/1/19 0:03 4/1/19 0:20 0:17:28                2          6226
##  3 22178531 4/1/19 0:11 4/1/19 0:15 0:04:12                2          5649
##  4 22178532 4/1/19 0:13 4/1/19 0:18 0:05:57                2          4151
##  5 22178533 4/1/19 0:19 4/1/19 0:36 0:16:47                2          3270
##  6 22178534 4/1/19 0:19 4/1/19 0:23 0:04:17                2          3123
##  7 22178535 4/1/19 0:26 4/1/19 0:35 0:09:08                2          6418
##  8 22178536 4/1/19 0:29 4/1/19 0:36 0:06:23                2          4513
##  9 22178537 4/1/19 0:32 4/1/19 1:07 0:35:37                2          3280
## 10 22178538 4/1/19 0:32 4/1/19 1:07 0:35:20                2          5534
## # ... with 1,048,565 more rows, and 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>
```

```
(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
                  ,gender = gender
                  ,birthyear = birthyear))
```

```
## # A tibble: 365,069 x 13
##     ride_id started_at  ended_at    rideable_type ride_length day_of_week
##       <dbl> <chr>       <chr>               <dbl> <chr>              <dbl>
##  1 21742443 1/1/19 0:04 1/1/19 0:11          2167 0:06:30                3
##  2 21742444 1/1/19 0:08 1/1/19 0:15          4386 0:07:21                3
##  3 21742445 1/1/19 0:13 1/1/19 0:27          1524 0:13:49                3
##  4 21742446 1/1/19 0:13 1/1/19 0:43           252 0:29:43                3
##  5 21742447 1/1/19 0:14 1/1/19 0:20          1170 0:06:04                3
##  6 21742448 1/1/19 0:15 1/1/19 0:19          2437 0:03:36                3
##  7 21742449 1/1/19 0:16 1/1/19 0:19          2708 0:02:57                3
##  8 21742450 1/1/19 0:18 1/1/19 0:20          2796 0:01:40                3
```

```
##  9 21742451 1/1/19 0:18 1/1/19 0:47          6205 0:28:47               3
## 10 21742452 1/1/19 0:19 1/1/19 0:24          3939 0:05:36               3
## # ... with 365,059 more rows, and 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>
```

Now we convert ride_id and rideable_type to character so that they can stack correctly and we take the opportunity to turn our char-type dates into the R classdate so we can perform calculations

```
q4_2019 <-  mutate(q4_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))
q2_2019 <-  mutate(q2_2019, ride_id = as.character(ride_id)
                   ,rideable_type = as.character(rideable_type))
q1_2019 <-  mutate(q1_2019, ride_id = as.character(ride_id)
                   ,rideable_type = as.character(rideable_type))
```

Inspect the dataframes and look for incongruencies

```
str(q1_2019)
```

```
## tibble [365,069 x 13] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:365069] "21742443" "21742444" "21742445" "21742446" ...
##  $ started_at        : chr [1:365069] "1/1/19 0:04" "1/1/19 0:08" "1/1/19 0:13" "1/1/19 0:13" ...
##  $ ended_at          : chr [1:365069] "1/1/19 0:11" "1/1/19 0:15" "1/1/19 0:27" "1/1/19 0:43" ...
##  $ rideable_type     : chr [1:365069] "2167" "4386" "1524" "252" ...
##  $ ride_length       : chr [1:365069] "0:06:30" "0:07:21" "0:13:49" "0:29:43" ...
##  $ day_of_week       : num [1:365069] 3 3 3 3 3 3 3 3 3 3 ...
##  $ 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 &
##  $ 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 (*)" "W
##  $ 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 ...
```

```
str(q2_2019)
```

```
## tibble [1,048,575 x 13] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:1048575] "22178529" "22178530" "22178531" "22178532" ...
##  $ started_at        : chr [1:1048575] "4/1/19 0:02" "4/1/19 0:03" "4/1/19 0:11" "4/1/19 0:13" ...
##  $ ended_at          : chr [1:1048575] "4/1/19 0:09" "4/1/19 0:20" "4/1/19 0:15" "4/1/19 0:18" ...
##  $ ride_length       : chr [1:1048575] "0:07:26" "0:17:28" "0:04:12" "0:05:57" ...
##  $ day_of_week       : num [1:1048575] 2 2 2 2 2 2 2 2 2 2 ...
##  $ rideable_type     : chr [1:1048575] "6251" "6226" "5649" "4151" ...
##  $ start_station_id  : num [1:1048575] 81 317 283 26 202 420 503 260 211 211 ...
##  $ start_station_name: chr [1:1048575] "Daley Center Plaza" "Wood St & Taylor St" "LaSalle St & Jacks
##  $ end_station_id    : num [1:1048575] 56 59 174 133 129 426 500 499 211 211 ...
##  $ end_station_name  : chr [1:1048575] "Desplaines St & Kinzie St" "Wabash Ave & Roosevelt Rd" "Canal
##  $ member_casual     : chr [1:1048575] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
##  $ gender            : chr [1:1048575] "Male" "Female" "Male" "Male" ...
##  $ birthyear         : num [1:1048575] 1975 1984 1990 1993 1992 ...
```

```
str(q3_2019)
```

```
## tibble [1,048,575 x 13] (S3: tbl_df/tbl/data.frame)
```

```
##  $ ride_id           : chr [1:1048575] "23479388" "23479389" "23479390" "23479391" ...
##  $ started_at        : chr [1:1048575] "7/1/19 0:00" "7/1/19 0:01" "7/1/19 0:01" "7/1/19 0:02" ...
##  $ ended_at          : chr [1:1048575] "7/1/19 0:20" "7/1/19 0:18" "7/1/19 0:27" "7/1/19 0:27" ...
##  $ rideable_type     : chr [1:1048575] "3591" "5353" "6180" "5540" ...
##  $ ride_length       : chr [1:1048575] "0:20:14" "0:17:28" "0:25:54" "0:25:03" ...
##  $ day_of_week       : num [1:1048575] 2 2 2 2 2 2 2 2 2 2 ...
##  $ start_station_id  : num [1:1048575] 117 381 313 313 168 300 168 313 43 43 ...
##  $ start_station_name: chr [1:1048575] "Wilton Ave & Belmont Ave" "Western Ave & Monroe St" "Lakevie
##  $ end_station_id    : num [1:1048575] 497 203 144 144 62 232 62 144 195 195 ...
##  $ end_station_name  : chr [1:1048575] "Kimball Ave & Belmont Ave" "Western Ave & 21st St" "Larrabee
##  $ member_casual     : chr [1:1048575] "Subscriber" "Customer" "Customer" "Customer" ...
##  $ gender            : chr [1:1048575] "Male" NA NA NA ...
##  $ birthyear         : num [1:1048575] 1992 NA NA NA NA ...
```

```
str(q4_2019)
```

```
## tibble [704,054 x 13] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:704054] "25223640" "25223641" "25223642" "25223643" ...
##  $ started_at        : chr [1:704054] "10/1/19 0:01" "10/1/19 0:02" "10/1/19 0:04" "10/1/19 0:04" ..
##  $ ended_at          : chr [1:704054] "10/1/19 0:17" "10/1/19 0:06" "10/1/19 0:18" "10/1/19 0:43" ..
##  $ rideable_type     : chr [1:704054] "2215" "6328" "3003" "3275" ...
##  $ ride_length       : chr [1:704054] "0:15:41" "1:15:41" "2:15:41" "3:15:41" ...
##  $ day_of_week       : num [1:704054] 3 3 3 3 3 3 3 3 3 3 ...
##  $ 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
##  $ 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 &
##  $ 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 ...
```

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

```
all_trips <- bind_rows(q1_2019, q2_2019, q3_2019, q4_2019)
```

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"     "ride_length"        "day_of_week"
##  [7] "start_station_id"  "start_station_name" "end_station_id"
## [10] "end_station_name"  "member_casual"      "gender"
## [13] "birthyear"
```

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

```
## [1] 3166273
```

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

```
## [1] 3166273      13
```

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

```
## # A tibble: 6 x 13
##   ride_id  started_at  ended_at    rideable_type ride_length day_of_week
##   <chr>    <chr>       <chr>       <chr>         <chr>             <dbl>
## 1 21742443 1/1/19 0:04 1/1/19 0:11 2167          0:06:30               3
## 2 21742444 1/1/19 0:08 1/1/19 0:15 4386          0:07:21               3
## 3 21742445 1/1/19 0:13 1/1/19 0:27 1524          0:13:49               3
## 4 21742446 1/1/19 0:13 1/1/19 0:43 252           0:29:43               3
## 5 21742447 1/1/19 0:14 1/1/19 0:20 1170          0:06:04               3
## 6 21742448 1/1/19 0:15 1/1/19 0:19 2437          0:03:36               3
## # ... with 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>
```

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

```
## tibble [3,166,273 x 13] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:3166273] "21742443" "21742444" "21742445" "21742446" ...
##  $ started_at        : chr [1:3166273] "1/1/19 0:04" "1/1/19 0:08" "1/1/19 0:13" "1/1/19 0:13" ...
##  $ ended_at          : chr [1:3166273] "1/1/19 0:11" "1/1/19 0:15" "1/1/19 0:27" "1/1/19 0:43" ...
##  $ rideable_type     : chr [1:3166273] "2167" "4386" "1524" "252" ...
##  $ ride_length       : chr [1:3166273] "0:06:30" "0:07:21" "0:13:49" "0:29:43" ...
##  $ day_of_week       : num [1:3166273] 3 3 3 3 3 3 3 3 3 3 ...
##  $ start_station_id  : num [1:3166273] 199 44 15 123 173 98 98 211 150 268 ...
##  $ start_station_name: chr [1:3166273] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave
##  $ end_station_id    : num [1:3166273] 84 624 644 176 35 49 49 142 148 141 ...
##  $ end_station_name  : chr [1:3166273] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)"
##  $ member_casual     : chr [1:3166273] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
##  $ gender            : chr [1:3166273] "Male" "Female" "Female" "Male" ...
##  $ birthyear         : num [1:3166273] 1989 1990 1994 1993 1994 ...
```

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

```
##    ride_id           started_at          ended_at         rideable_type
##  Length:3166273     Length:3166273     Length:3166273     Length:3166273
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  ride_length         day_of_week   start_station_id start_station_name
##  Length:3166273     Min.   :1.00   Min.   :  1.0    Length:3166273
##  Class :character   1st Qu.:2.00   1st Qu.: 77.0    Class :character
##  Mode  :character   Median :4.00   Median :174.0    Mode  :character
##                     Mean   :4.04   Mean   :201.3
##                     3rd Qu.:6.00   3rd Qu.:289.0
##                     Max.   :7.00   Max.   :673.0
##
##  end_station_id  end_station_name   member_casual         gender
##  Min.   :  1.0   Length:3166273     Length:3166273     Length:3166273
##  1st Qu.: 77.0   Class :character   Class :character   Class :character
##  Median :174.0   Mode  :character   Mode  :character   Mode  :character
##  Mean   :202.1
```

```
##  3rd Qu.:290.0
##  Max.   :673.0
##
##    birthyear
##  Min.   :1759
##  1st Qu.:1979
##  Median :1987
##  Mean   :1984
##  3rd Qu.:1992
##  Max.   :2014
##  NA's   :434079
```

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)
```

```
##
##   Customer Subscriber
##     700318    2465955
```

Reassign to the desired values (we will go 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
##  700318 2465955
```

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

```
all_trips$date <- as.Date(all_trips$started_at, format = "%m/%d/%Y %H:%M") #The default format is yyyy-
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")
```

Also, turn our dates into the R Date-Time Class

```
all_trips$started_at <-as.POSIXct(all_trips$started_at,format="%m/%d/%Y %H:%M",tz='UTC')
all_trips$ended_at <- as.POSIXct(all_trips$ended_at,format="%m/%d/%Y %H:%M",tz='UTC')

str(all_trips)
```

```
## tibble [3,166,273 x 17] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:3166273] "21742443" "21742444" "21742445" "21742446" ...
##  $ started_at        : POSIXct[1:3166273], format: "0019-01-01 00:04:00" "0019-01-01 00:08:00" ...
##  $ ended_at          : POSIXct[1:3166273], format: "0019-01-01 00:11:00" "0019-01-01 00:15:00" ...
##  $ rideable_type     : chr [1:3166273] "2167" "4386" "1524" "252" ...
##  $ ride_length       : chr [1:3166273] "0:06:30" "0:07:21" "0:13:49" "0:29:43" ...
##  $ day_of_week       : chr [1:3166273] "Tuesday" "Tuesday" "Tuesday" "Tuesday" ...
##  $ start_station_id  : num [1:3166273] 199 44 15 123 173 98 98 211 150 268 ...
##  $ start_station_name: chr [1:3166273] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave
##  $ end_station_id    : num [1:3166273] 84 624 644 176 35 49 49 142 148 141 ...
##  $ end_station_name  : chr [1:3166273] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)"
##  $ member_casual     : chr [1:3166273] "member" "member" "member" "member" ...
##  $ gender            : chr [1:3166273] "Male" "Female" "Female" "Male" ...
##  $ birthyear         : num [1:3166273] 1989 1990 1994 1993 1994 ...
##  $ date              : Date[1:3166273], format: "0019-01-01" "0019-01-01" ...
##  $ month             : chr [1:3166273] "01" "01" "01" "01" ...
##  $ day               : chr [1:3166273] "01" "01" "01" "01" ...
##  $ year              : chr [1:3166273] "0019" "0019" "0019" "0019" ...
```

Add a "ride_length" calculation to all_trips (in seconds), we already did this on Excel, but I like to remind myself how I would do it if I did everything in R. This is a useful library if you find the code below confusing 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, units = c("secs"))
```

Inspect the structure of the columns

```
str(all_trips)
```

```
## tibble [3,166,273 x 17] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:3166273] "21742443" "21742444" "21742445" "21742446" ...
##  $ started_at        : POSIXct[1:3166273], format: "0019-01-01 00:04:00" "0019-01-01 00:08:00" ...
##  $ ended_at          : POSIXct[1:3166273], format: "0019-01-01 00:11:00" "0019-01-01 00:15:00" ...
##  $ rideable_type     : chr [1:3166273] "2167" "4386" "1524" "252" ...
##  $ ride_length       : 'difftime' num [1:3166273] 420 420 840 1800 ...
##   ..- attr(*, "units")= chr "secs"
##  $ day_of_week       : chr [1:3166273] "Tuesday" "Tuesday" "Tuesday" "Tuesday" ...
##  $ start_station_id  : num [1:3166273] 199 44 15 123 173 98 98 211 150 268 ...
##  $ start_station_name: chr [1:3166273] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave
##  $ end_station_id    : num [1:3166273] 84 624 644 176 35 49 49 142 148 141 ...
##  $ end_station_name  : chr [1:3166273] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)"
##  $ member_casual     : chr [1:3166273] "member" "member" "member" "member" ...
##  $ gender            : chr [1:3166273] "Male" "Female" "Female" "Male" ...
##  $ birthyear         : num [1:3166273] 1989 1990 1994 1993 1994 ...
##  $ date              : Date[1:3166273], format: "0019-01-01" "0019-01-01" ...
##  $ month             : chr [1:3166273] "01" "01" "01" "01" ...
##  $ day               : chr [1:3166273] "01" "01" "01" "01" ...
##  $ year              : chr [1:3166273] "0019" "0019" "0019" "0019" ...
```

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$ride_length <- as.numeric(all_trips$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, na.rm = TRUE) #straight average (total ride length / rides)
```

```
## [1] 1441.017
```

```
median(all_trips_v2$ride_length, na.rm = TRUE) #midpoint number in the ascending array of ride lengths
```

```
## [1] 720
```

```
max(all_trips_v2$ride_length, na.rm = TRUE) #longest ride
```

```
## [1] 10632000
```

```
min(all_trips_v2$ride_length, na.rm = TRUE) #shortest ride
```

```
## [1] 60
```

You can condense the four lines above to one line using summary() on the specific attribute

```
summary(all_trips_v2$ride_length, na.rm = TRUE)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##       60      420      720     1441     1260 10632000
```

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                3506.5353
## 2                     member                 854.4247
```

```
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                      1560
## 2                     member                       600
```

```
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                  10632000
```

```
## 2                     member                   9056640
```

```
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                       60
## 2                     member                       60
```

In this part, we have found that, to my happy surprise, casual member rides are generally longer than the member rides. Now we know there is a financial opportunity to turn casual riders into members For our next step we need our days in order, currently they are not so let's change that

```
all_trips_v2$day_of_week <- ordered(all_trips_v2$day_of_week, levels=c("Sunday", "Monday", "Tuesday", "
```

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                   Sunday                3390.4778
## 2                      member                   Sunday                 923.0055
## 3                      casual                   Monday                3341.0472
## 4                      member                   Monday                 847.1972
## 5                      casual                  Tuesday                3540.2557
## 6                      member                  Tuesday                 843.8204
## 7                      casual                Wednesday                3809.8915
## 8                      member                Wednesday                 813.2484
## 9                      casual                 Thursday                3725.5807
## 10                     member                 Thursday                 831.3104
## 11                     casual                   Friday                3766.9274
## 12                     member                   Friday                 830.8192
## 13                     casual                 Saturday                3264.6637
## 14                     member                 Saturday                 974.0835
```

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
            ,average_duration = mean(ride_length)) %>%           # average duration
  arrange(member_casual, weekday)                                          # sorts
```

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

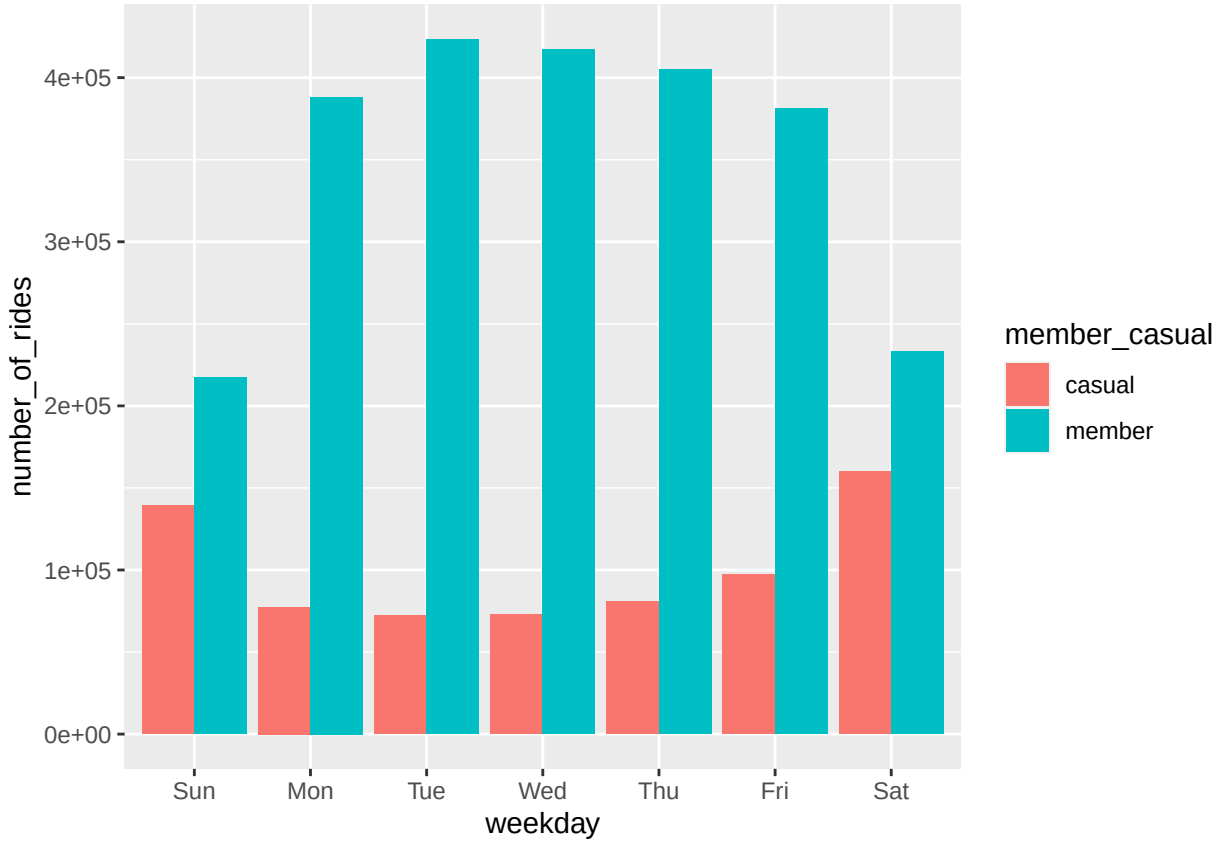
```
## # A tibble: 14 x 4
## # Groups:   member_casual [2]
##    member_casual weekday number_of_rides average_duration
##    <chr>         <ord>             <int>            <dbl>
##  1 casual        Sun              139266            3390.
##  2 casual        Mon               77444            3341.
##  3 casual        Tue               72260            3540.
##  4 casual        Wed               73026            3810.
##  5 casual        Thu               80882            3726.
##  6 casual        Fri               97436            3767.
##  7 casual        Sat              159998            3265.
```

```
##  8 member        Sun             217524          923.
##  9 member        Mon             388161          847.
## 10 member        Tue             423379          844.
## 11 member        Wed             417095          813.
## 12 member        Thu             405072          831.
## 13 member        Fri             381406          831.
## 14 member        Sat             233311          974.
```

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.
```



Most members, rent bikes within the week (Mon-Fri) and most casuals, rent bikes in the weekend. The casual to member ratio goes up on the weekend. I would recommend to promote this days more since there is a high opportunity of converting casual into members in this day.

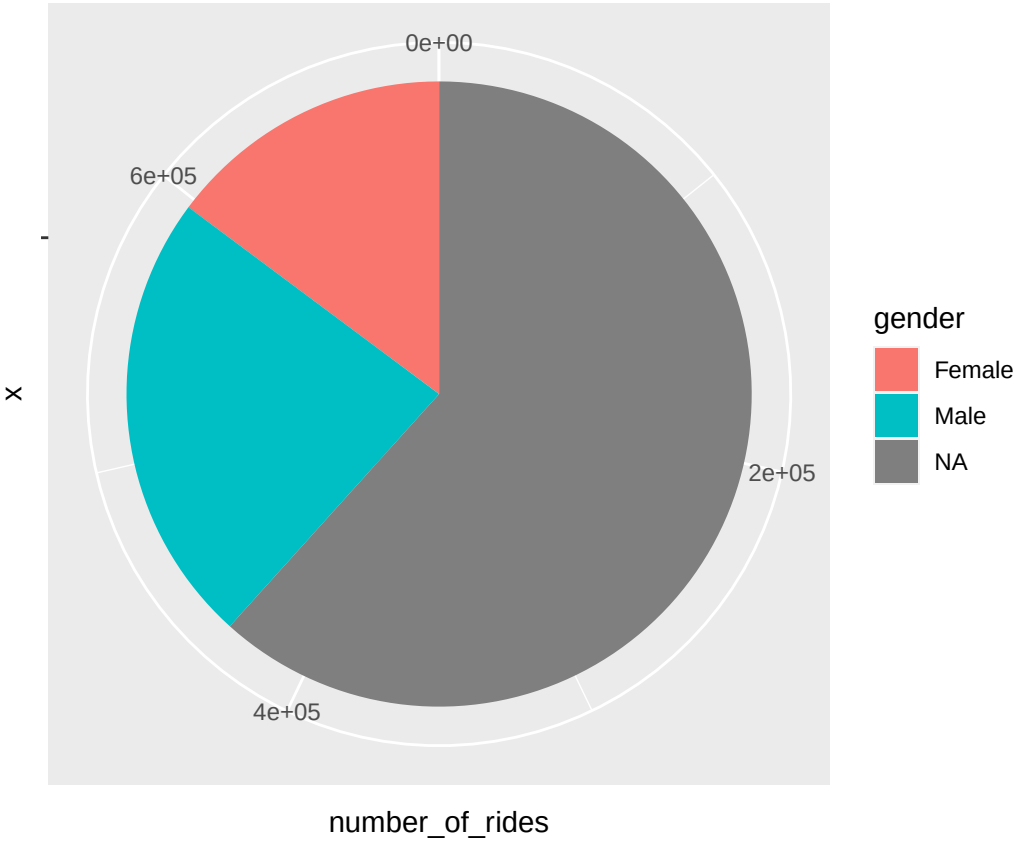
Maybe a weekly activity like hosting bike-compatible events on weekends (races, search&find, collecting tokens at locations).

Another useful action to take is comparing the price of four day rides to a member payment in a favorable light. Let's visualize the gender ratio between casual and members

Gender ratio in casual rides

```
all_trips_v2 %>%
  group_by(member_casual, gender) %>%
  summarise(number_of_rides = n()) %>%
  filter(member_casual == "casual",.preserve = FALSE) %>%
  ggplot(aes(x = "", y = number_of_rides, fill = gender)) +
  geom_bar(stat="identity", width=1) +
  coord_polar("y", start=0)
```

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

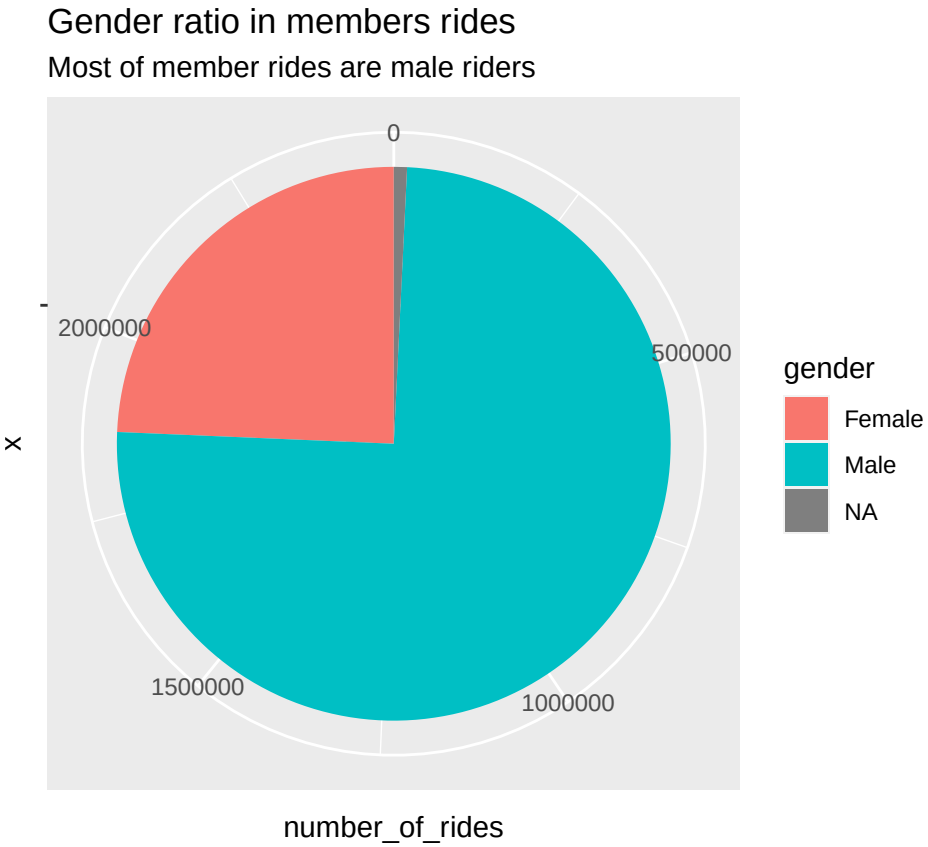


Most people who ride casually rarely select a gender so we can't get much from this Gender ratio in members rides

```
all_trips_v2 %>%
  group_by(member_casual, gender) %>%
  filter(member_casual == "member") %>%
  summarise(number_of_rides = n()) %>%
  ggplot(aes(x = "", y = number_of_rides, fill = gender)) +
  geom_bar(stat = "identity", width=1) +
  coord_polar("y", start=0) +
  ggtitle(label = 'Gender ratio in members rides', subtitle = 'Most of member rides are male riders')
```

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

```
## `.groups` argument.
```

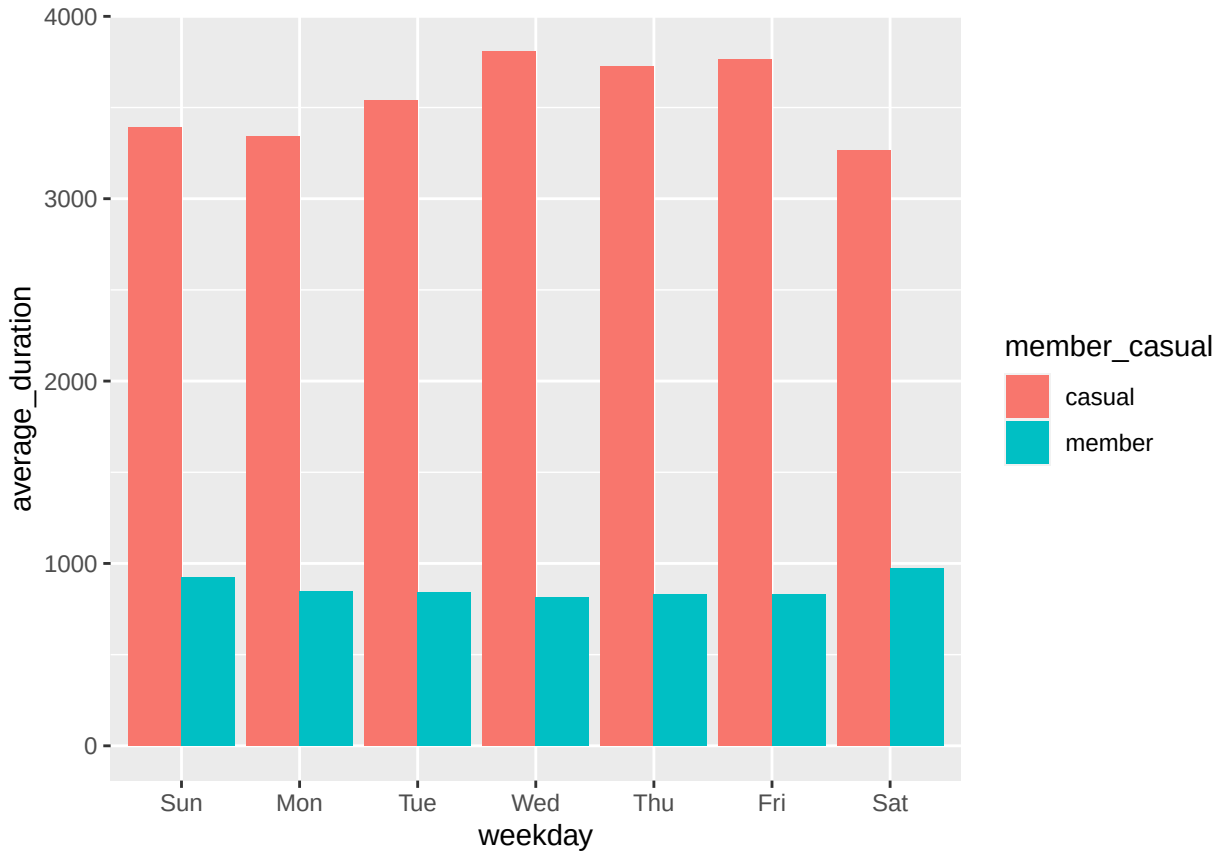


Member rides are overwhelmingly male, we would expect a slight male preference but this is very drastic. The company is getting over 75% of their costumers from roughly 50% of population. Advertise more to women!

Still, advertising might not be enough, maybe this is a reflection of a bigger problem where women don't feel safe riding bikes on the city, this is a subject worth spending more research on 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.
```



Member and casual ride-durations stay consistent throughout the week, casual member rides are significantly longer, there is potential to take advantage of this but all of them have certain drawback Let's make a list of the most used start station by members and casuals This is the top most used by members

```
all_trips_v2 %>%
  group_by(member_casual, start_station_name) %>%
  mutate(station_usage = n()) %>%
  select(member_casual, start_station_name, station_usage) %>%
  arrange(desc(station_usage)) %>%
  filter(member_casual == "member") %>%
  distinct() %>%
  print()
```

```
## # A tibble: 637 x 3
## # Groups:   member_casual, start_station_name [637]
##    member_casual start_station_name           station_usage
##    <chr>         <chr>                                <int>
##  1 member        Canal St & Adams St                  43032
##  2 member        Clinton St & Washington Blvd         39036
##  3 member        Clinton St & Madison St              38925
##  4 member        Columbus Dr & Randolph St            27008
##  5 member        Kingsbury St & Kinzie St             26078
##  6 member        Franklin St & Monroe St              25945
##  7 member        Daley Center Plaza                   25075
##  8 member        Canal St & Madison St                23305
##  9 member        Michigan Ave & Washington St         21373
```

```
## 10 member        LaSalle St & Jackson Blvd            19425
## # ... with 627 more rows
```

Let's make a list of the most used destination station by members and casuals

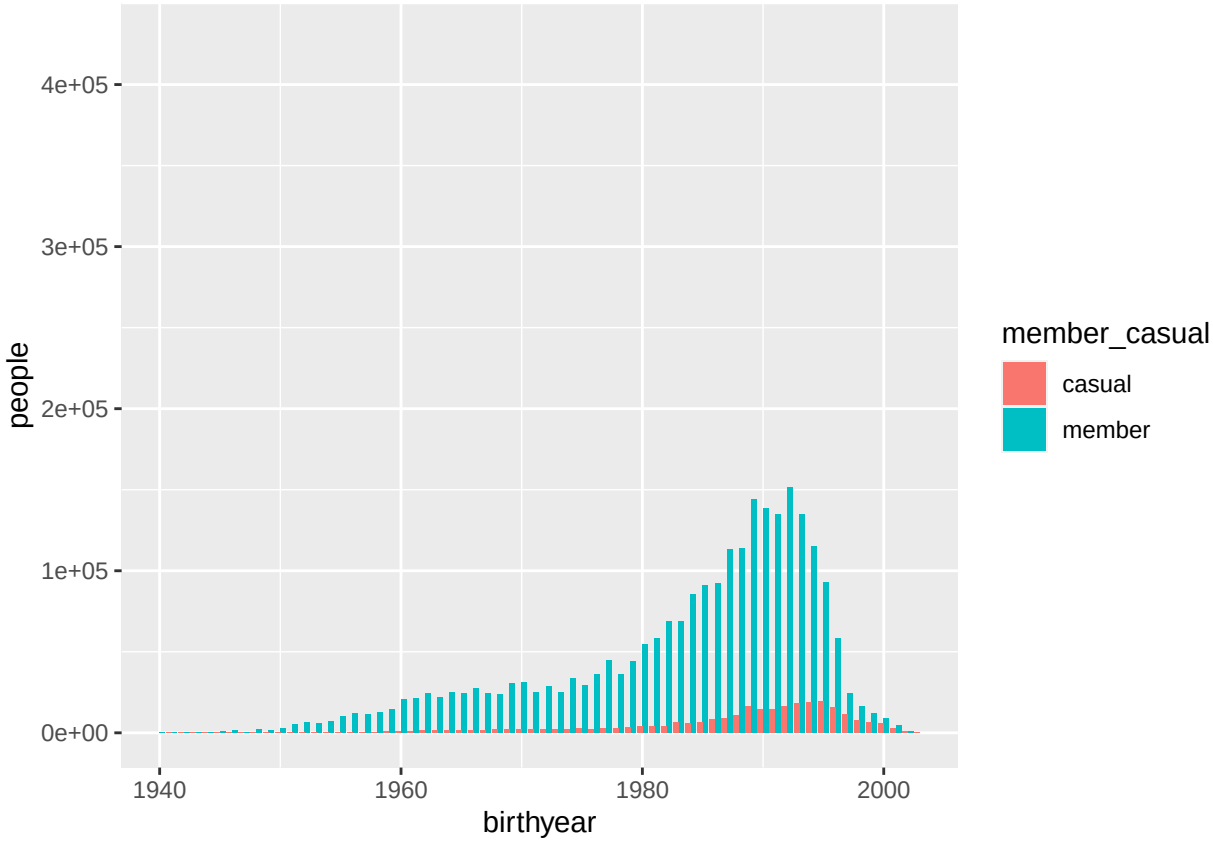
```
all_trips_v2 %>%
  group_by(member_casual, end_station_name) %>%
  mutate(station_usage = n()) %>%
  select(member_casual, end_station_name, station_usage) %>%
  arrange(desc(station_usage)) %>%
  filter(member_casual == "casual") %>%
  distinct() %>%
  print()
```

```
## # A tibble: 641 x 3
## # Groups:   member_casual, end_station_name [641]
##    member_casual end_station_name             station_usage
##    <chr>         <chr>                                <int>
##  1 casual        Streeter Dr & Grand Ave              53797
##  2 casual        Lake Shore Dr & Monroe St            25218
##  3 casual        Millennium Park                      19972
##  4 casual        Michigan Ave & Oak St                18830
##  5 casual        Lake Shore Dr & North Blvd           18809
##  6 casual        Theater on the Lake                  15238
##  7 casual        Shedd Aquarium                       13280
##  8 casual        Michigan Ave & Washington St         10468
##  9 casual        Adler Planetarium                     8652
## 10 casual        Dusable Harbor                        7782
## # ... with 631 more rows
```

We can't do analysis on this part yet! I will show my stakeholders a map with the most used stations, which is achievable in R but I want to make use of a unique Tableau tool for my analysis and visualization Let's see if there are significant age differences among members and casual rides

```
all_trips_v2 %>%
  group_by(member_casual, birthyear) %>%
  mutate(people = n()) %>%
  select(member_casual, birthyear, people) %>%
  arrange(desc(people)) %>%
  distinct() %>%
  ggplot(aes(x = birthyear, y = people, fill = member_casual)) +
  geom_col(position = "dodge")+
  xlim(1940, 2003)
```

```
## Warning: Removed 39 rows containing missing values (geom_col).
```



As expected, the majority of rides are done by the youngest adults and young adults in general should be the main age target

STEP 5: EXPORT SUMMARY FILE FOR FURTHER ANALYSIS

Create a csv file that we will visualize in Excel, Tableau, or my presentation software N.B.: This file location is for a Mac. If you are working on a PC, change the file location accordingly

```
counts <- aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FU
write.csv(counts, file = '~/Desktop/Cyclystic_2019byQuarter/avg_ride_length.csv')
```

We are done!