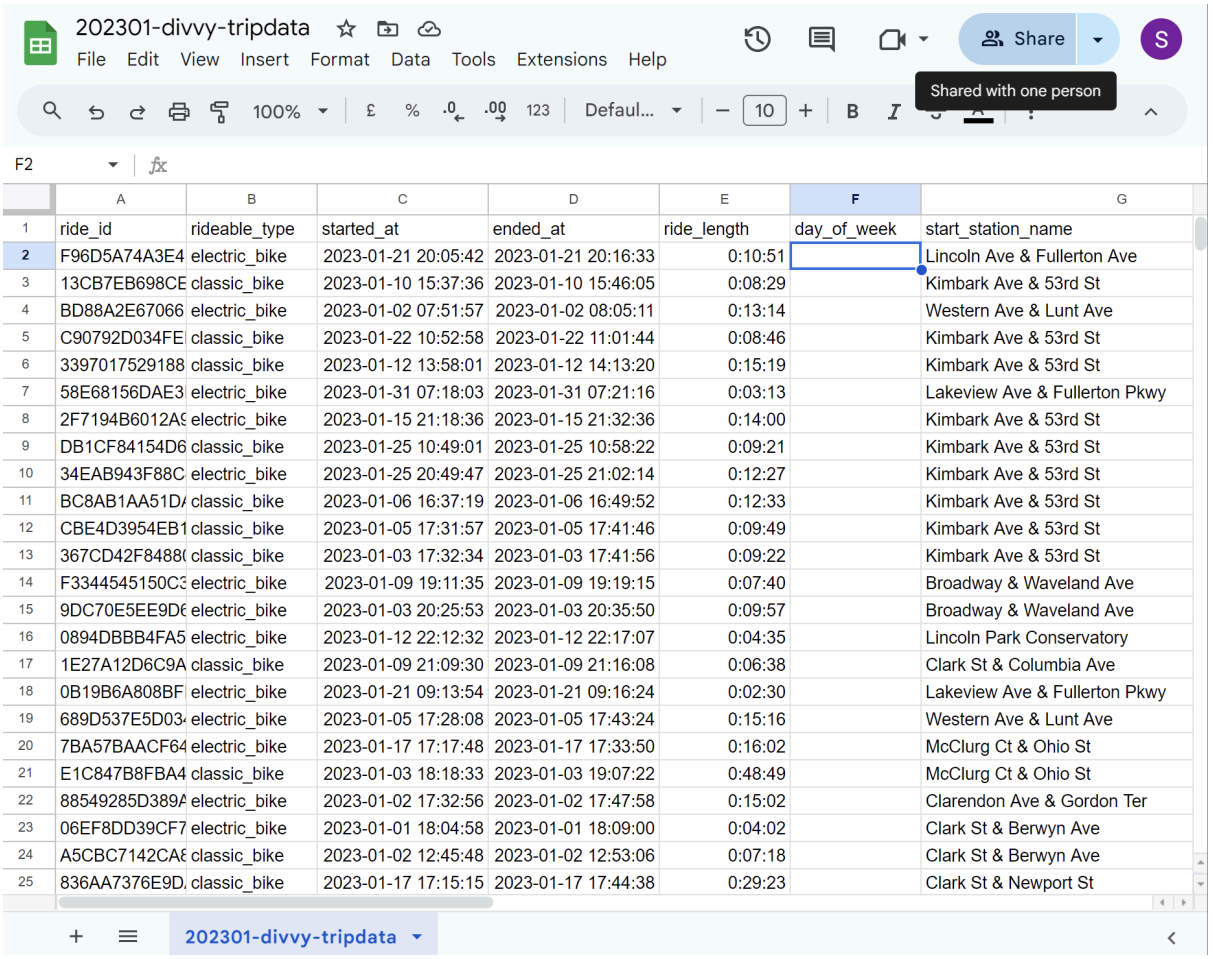
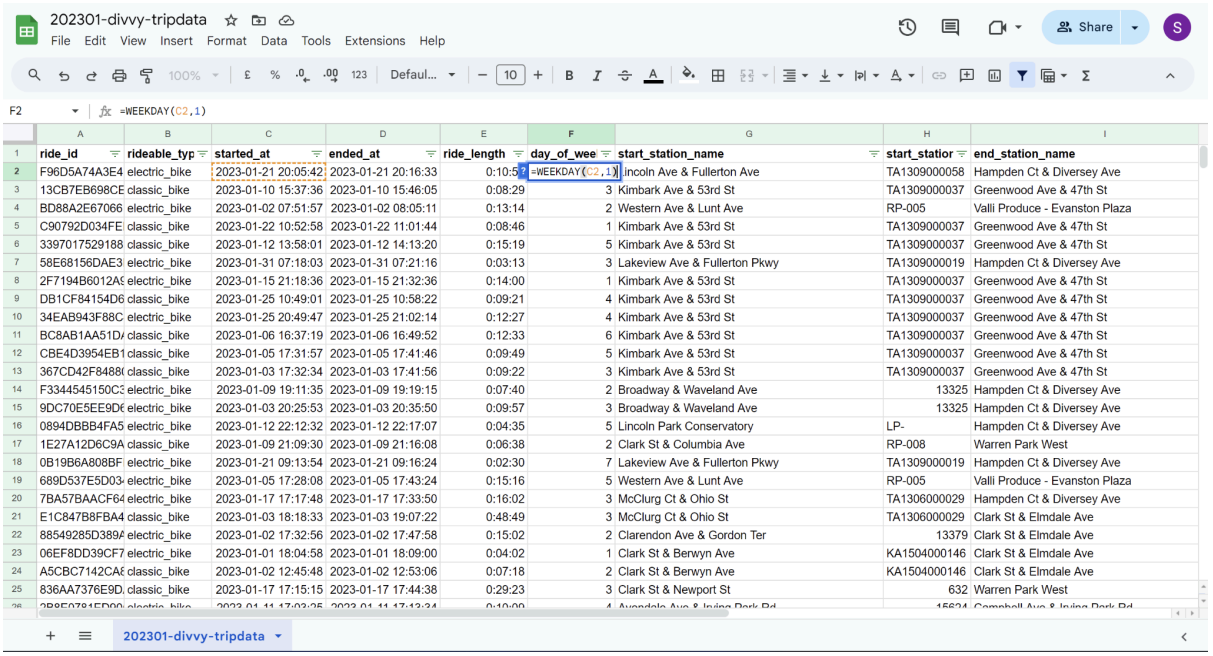
Analysis of Cyclistic customer behaviour (member vs casual riders)

Cyclistic is a bike-share company in Chicago. The director of marketing believes the company's future success depends on maximising the number of annual memberships. Thus, the team wants to better understand how casual riders and annual members use the bikes offered by Cyclistic differently from each other. The insights generated will allow the team to design a new marketing strategy, in order to convert casual riders into annual members.

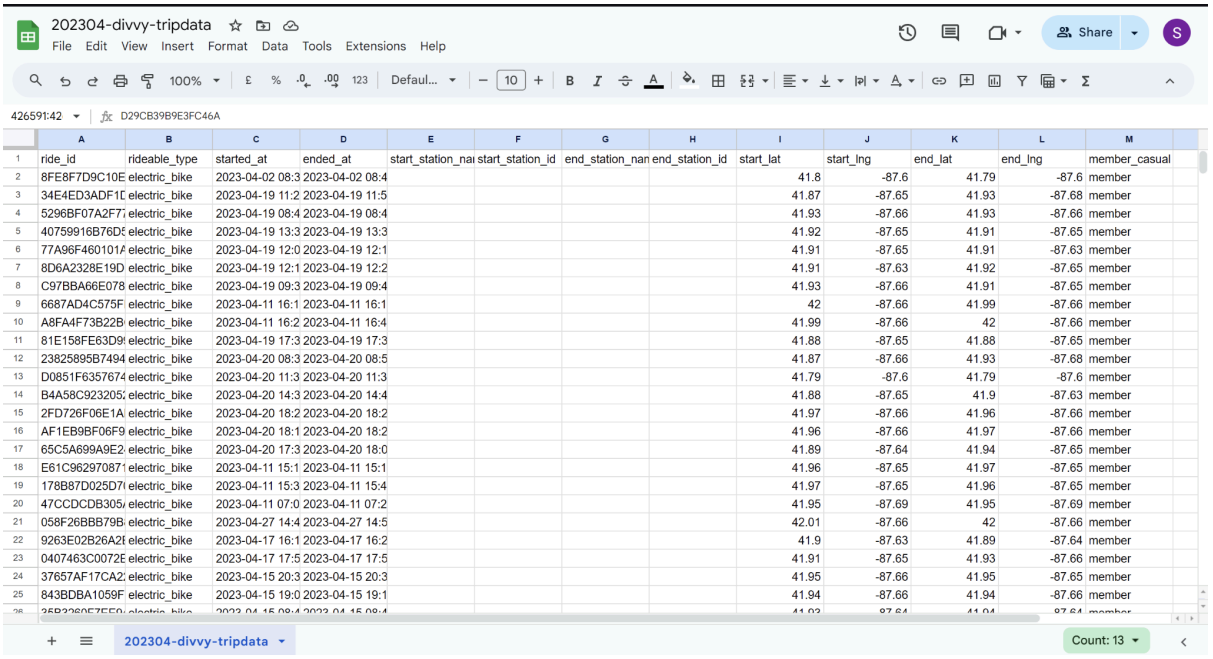


	A	B	C	D	E	F	G
1	ride_id	rideable_type	started_at	ended_at	ride_length	day_of_week	start_station_name
2	F96D5A74A3E4	electric_bike	2023-01-21 20:05:42	2023-01-21 20:16:33	0:10:51		Lincoln Ave & Fullerton Ave
3	13CB7EB698CE	classic_bike	2023-01-10 15:37:36	2023-01-10 15:46:05	0:08:29		Kimbark Ave & 53rd St
4	BD88A2E67066	electric_bike	2023-01-02 07:51:57	2023-01-02 08:05:11	0:13:14		Western Ave & Lunt Ave
5	C90792D034FE	classic_bike	2023-01-22 10:52:58	2023-01-22 11:01:44	0:08:46		Kimbark Ave & 53rd St
6	3397017529188	classic_bike	2023-01-12 13:58:01	2023-01-12 14:13:20	0:15:19		Kimbark Ave & 53rd St
7	58E68156DAE3	electric_bike	2023-01-31 07:18:03	2023-01-31 07:21:16	0:03:13		Lakeview Ave & Fullerton Pkwy
8	2F7194B6012A	electric_bike	2023-01-15 21:18:36	2023-01-15 21:32:36	0:14:00		Kimbark Ave & 53rd St
9	DB1CF84154D6	classic_bike	2023-01-25 10:49:01	2023-01-25 10:58:22	0:09:21		Kimbark Ave & 53rd St
10	34EAB943F88C	electric_bike	2023-01-25 20:49:47	2023-01-25 21:02:14	0:12:27		Kimbark Ave & 53rd St
11	BC8AB1AA51D/	classic_bike	2023-01-06 16:37:19	2023-01-06 16:49:52	0:12:33		Kimbark Ave & 53rd St
12	CBE4D3954EB1	classic_bike	2023-01-05 17:31:57	2023-01-05 17:41:46	0:09:49		Kimbark Ave & 53rd St
13	367CD42F8488(	classic_bike	2023-01-03 17:32:34	2023-01-03 17:41:56	0:09:22		Kimbark Ave & 53rd St
14	F3344545150C3	electric_bike	2023-01-09 19:11:35	2023-01-09 19:19:15	0:07:40		Broadway & Waveland Ave
15	9DC70E5EE9D(	electric_bike	2023-01-03 20:25:53	2023-01-03 20:35:50	0:09:57		Broadway & Waveland Ave
16	0894DBBB4FA5	electric_bike	2023-01-12 22:12:32	2023-01-12 22:17:07	0:04:35		Lincoln Park Conservatory
17	1E27A12D6C9A	classic_bike	2023-01-09 21:09:30	2023-01-09 21:16:08	0:06:38		Clark St & Columbia Ave
18	0B19B6A808BF	electric_bike	2023-01-21 09:13:54	2023-01-21 09:16:24	0:02:30		Lakeview Ave & Fullerton Pkwy
19	689D537E5D03	electric_bike	2023-01-05 17:28:08	2023-01-05 17:43:24	0:15:16		Western Ave & Lunt Ave
20	7BA57BAACF64	electric_bike	2023-01-17 17:17:48	2023-01-17 17:33:50	0:16:02		McClurg Ct & Ohio St
21	E1C847B8FBA4	classic_bike	2023-01-03 18:18:33	2023-01-03 19:07:22	0:48:49		McClurg Ct & Ohio St
22	88549285D389A	electric_bike	2023-01-02 17:32:56	2023-01-02 17:47:58	0:15:02		Clarendon Ave & Gordon Ter
23	06EF8DD39CF7	electric_bike	2023-01-01 18:04:58	2023-01-01 18:09:00	0:04:02		Clark St & Berwyn Ave
24	A5CBC7142CA8	classic_bike	2023-01-02 12:45:48	2023-01-02 12:53:06	0:07:18		Clark St & Berwyn Ave
25	836AA7376E9D	classic_bike	2023-01-17 17:15:15	2023-01-17 17:44:38	0:29:23		Clark St & Newport St

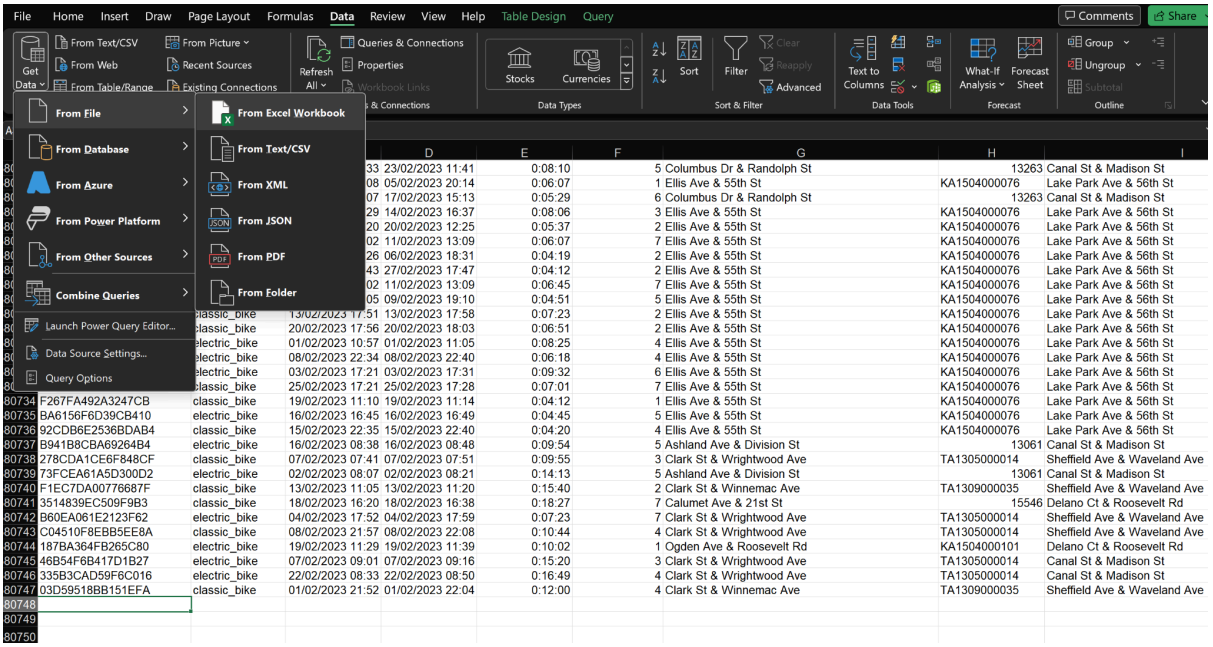
202301-divvy-tripdata

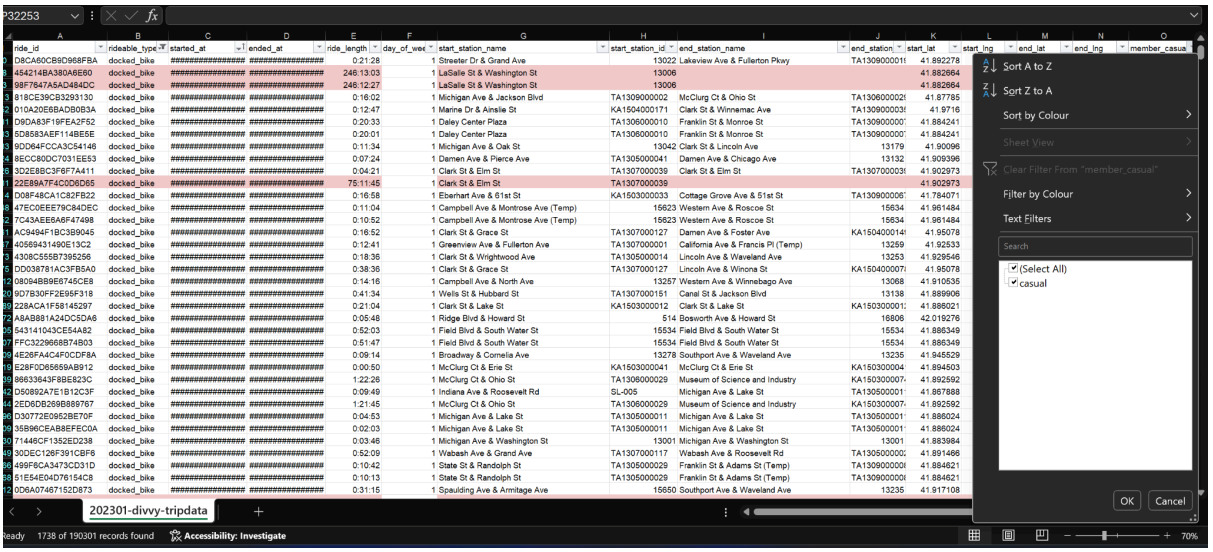


In the “day_of_week” column, I calculated the day of the week that each ride started using the ‘WEEKDAY’ function, as can be seen above. I then formatted the cells to be numbers without decimals, noting that 1 = Sunday and 7 = Saturday. I proceeded to do this for all the spreadsheets for each month.

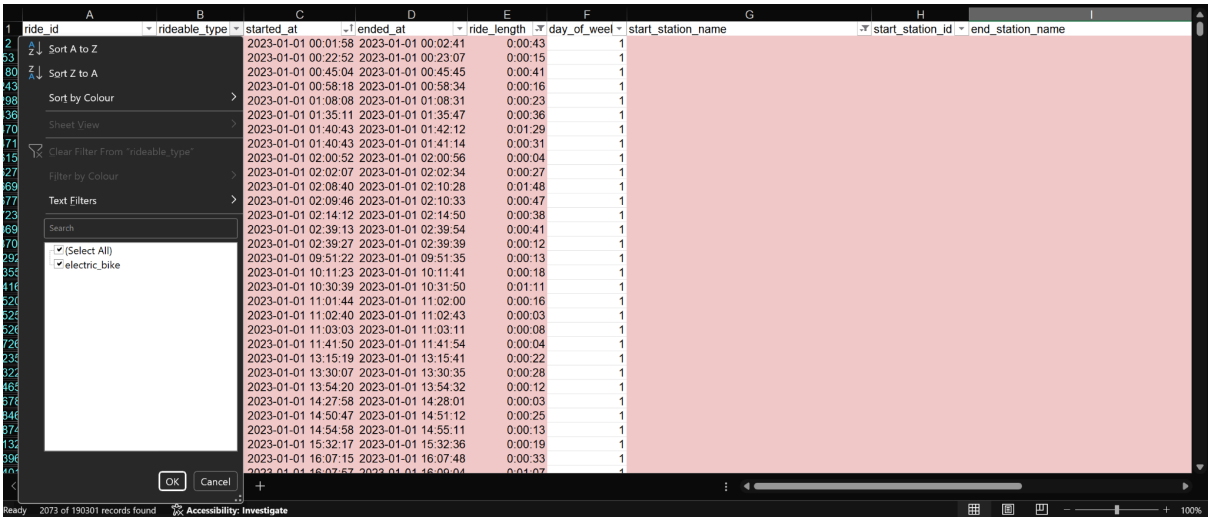


Upon opening the .csv file for the month of April, we can see there are blank fields in the data. This will need to be filled later by checking the matching longitude and latitude of each station. All months of the data will need to be filtered for blank fields and cleaned later when the spreadsheets are merged.





For the docked bike type, the only type of members using them in the month of January are casual members, as can be seen above in the screenshot of the filter on member types.



Some data will need to be deleted as well, such as those with ride lengths that are too short (e.g. less than 2 minutes) or rides that have no start and end stations, as these may have been taken out by Cyclistic for testing purposes and cannot be used for analysis. Majority of the rides that lasted 2 minutes or less are either missing a station, or they start and end at the same station. Also, when filtering for the rideable type, they are all electric bikes.

The same process will be carried out with every month of data.

I tried to delete all blank fields in the spreadsheet for January (which is smaller in size compared to the rest of the months), however, the spreadsheet would crash every time I tried. Instead I will have to do everything using R studio.

R studio cloud only had 1GB of RAM that I could use, which would not allow all files to be loaded as the .csv files altogether were too large. Therefore, I had to download R Studio onto my laptop.

Preparing the data

I had to first install and load the packages that I would be using for preparing and analysing the data.

Libraries

```
> install.packages("tidyverse")
WARNING: Rtools is required to build R packages but is not currently installed. Please downl
oad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/noora/AppData/Local/R/win-library/4.3'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/tidyverse_2.0.0.zip'
Content type 'application/zip' length 430880 bytes (420 KB)
downloaded 420 KB

package 'tidyverse' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\noora\AppData\Local\Temp\RtmpoVSaIw\downloaded_packages
> #loading libraries
> library(tidyverse)
── Attaching core tidyverse packages ──────────────────────────────────── tidyverse 2.0.0 ──
✓ dplyr     1.1.4     ✓ readr     2.1.5
✓ forcats   1.0.0     ✓ stringr   1.5.1
✓ ggplot2   3.4.4     ✓ tibble    3.2.1
✓ lubridate 1.9.3     ✓ tidyr     1.3.1
✓ purrr     1.0.2
── Conflicts ────────────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package to force all conflicts to become errors
> library(lubridate)
> library(hms) #time

Attaching package: 'hms'

The following object is masked from 'package:lubridate':

    hms

> library(data.table)
data.table 1.15.0 using 8 threads (see ?getDTthreads).  Latest news: r-datatable.com

Attaching package: 'data.table'

The following objects are masked from 'package:lubridate':

    hour, isoweek, mday, minute, month, quarter, second, wday, week, yday, year

The following objects are masked from 'package:dplyr':

    between, first, last

The following object is masked from 'package:purrr':

    transpose

> library(plyr)
------------------------------------------------------------------------------
You have loaded plyr after dplyr - this is likely to cause problems.
If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
library(plyr); library(dplyr)
------------------------------------------------------------------------------

Attaching package: 'plyr'

The following objects are masked from 'package:dplyr':

    arrange, count, desc, failwith, id, mutate, rename, summarise, summarize

The following object is masked from 'package:purrr':

    compact

> library(ggplot2)
```

Loading all 12 months of data

Next, I had to set the working directory where I saved all of the .csv files for the individual months using the following line of code:

```
setwd("~/R/Cyclistic Capstone Project")
```

The original .csv files/data frames were then loaded in order to merge them all for a full year view. Each individual month of data was loaded using the code below:

```
> jan2023_data <- read.csv("202301-divvy-tripdata.csv")

> feb2023_data <- read.csv("202302-divvy-tripdata.csv")

> mar2023_data <- read.csv("202303-divvy-tripdata.csv")

> apr2023_data <- read.csv("202304-divvy-tripdata.csv")

> may2023_data <- read.csv("202305-divvy-tripdata.csv")

> jun2023_data <- read.csv("202306-divvy-tripdata.csv")

> jul2023_data <- read.csv("202307-divvy-tripdata.csv")

> aug2023_data <- read.csv("202308-divvy-tripdata.csv")

> sep2023_data <- read.csv("202309-divvy-tripdata.csv")

> oct2023_data <- read.csv("202310-divvy-tripdata.csv")

> nov2023_data <- read.csv("202311-divvy-tripdata.csv")

> dec2023_data <- read.csv("202312-divvy-tripdata.csv")
```

Processing the data

Combining all 12 months of data into a single data frame

All data frames for every month were then merged together using the **rbind()** function.

```
> cyclistic_df <- rbind(jan2023_data, feb2023_data, mar2023_data,
apr2023_data, may2023_data, jun2023_data, jul2023_data, aug2023_data,
sep2023_data, oct2023_data, nov2023_data, dec2023_data)
```

In order to free up space in the environment, all individual data frames for each month were removed.

```
> remove(jan2023_data, feb2023_data, mar2023_data, apr2023_data,
may2023_data, jun2023_data, jul2023_data, aug2023_data, sep2023_data,
oct2023_data, nov2023_data, dec2023_data)
```

Adding ride length column

I ran into a lot of issues while trying to calculate the ride_length for each ride, as there were a lot of durations that lasted 0 minutes, which would not make any sense. The reason why it wouldn't make any sense, is because there were too many ride lengths that lasted 0 minutes, meanwhile there would definitely be a time difference. For example, a ride would start at 20:05:42 and end at 20:16:33, which would be a ride length of around 10 minutes. However, the ride_length column would show 0 minutes.

To fix this, I used the lubridate function to specify the format of the data in the **started_at** and **ended_at** columns. The lines of code are as follows:

```
> cyclistic_df2$started_at <- lubridate::ymd_hms(cyclistic_df2$started_at)

> cyclistic_df2$ended_at <- lubridate::ymd_hms(cyclistic_df2$ended_at)

> cyclistic_df2$ride_length <- difftime(cyclistic_df2$ended_at,
cyclistic_df2$started_at, units = "mins")
```

Adding columns for: day of the week, month, day, year, time, hour

The next columns to add to the data frame will be the day_of_week, month, day, year, time, and hour columns for analysis later.

```
> cyclistic_df2$day_of_week <- wday(cyclistic_df2$started_at) #calculate
the day of the week

> cyclistic_df2$day_of_week <- format(as.Date(cyclistic_df2$date), "%A")
#create column for day of week

> cyclistic_date$date <- as.Date(cyclistic_date$started_at) #default format
is yyyy-mm-dd, use start date

> cyclistic_df2$date <- as.Date(cyclistic_df2$started_at) #default format
is yyyy-mm-dd, use start date

> cyclistic_df2$day_of_week <- format(as.Date(cyclistic_df2$date), "%A")
#create column for day of week

> cyclistic_df2$month <- format(as.Date(cyclistic_df2$date), "%m")#create
column for month

> cyclistic_df2$day <- format(as.Date(cyclistic_df2$date), "%d") #create
column for day

> cyclistic_df2$year <- format(as.Date(cyclistic_df2$date), "%Y") #create
column for year

> cyclistic_df2$time <- format(as.Date(cyclistic_df2$date), "%H:%M:%S")
#format time as HH:MM:SS
```

```
> cyclistic_df2$time <- as_hms((cyclistic_df2$started_at)) #create new
column for time

> cyclistic_df2$hour <- hour(cyclistic_df2$time) #create new column for
hour
```

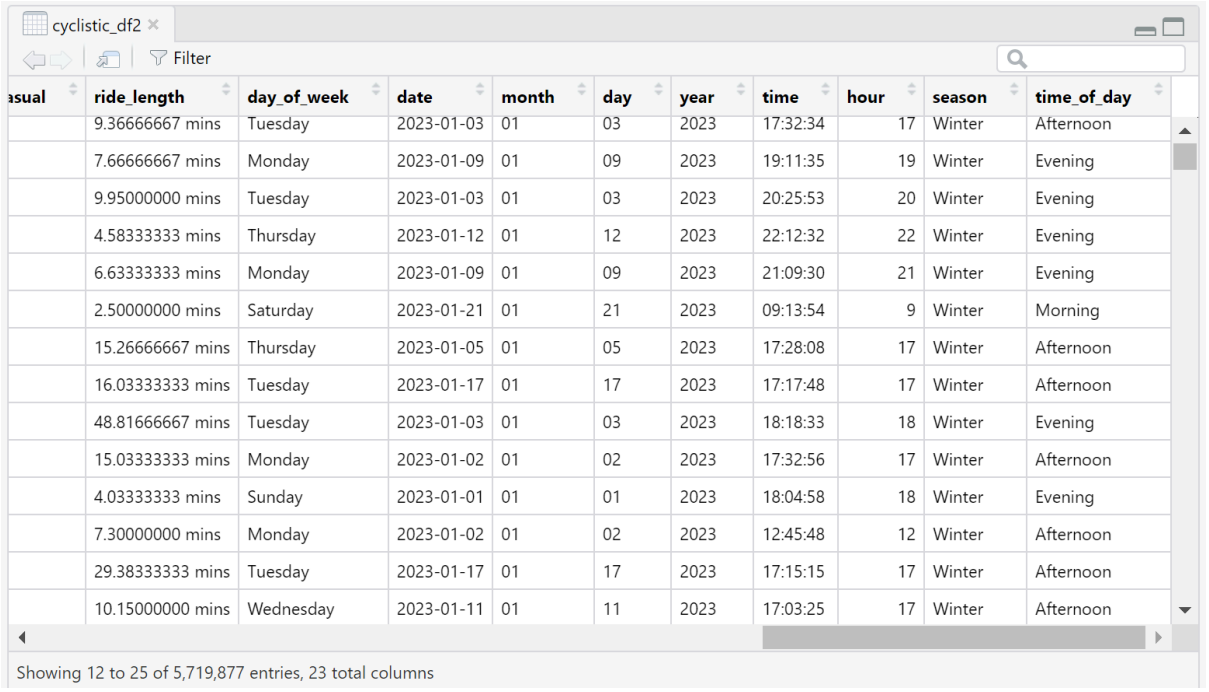
Creating columns for different seasons of the year for further analysis

```
> cyclistic_df2 <-cyclistic_df2 %>% mutate(season = case_when(month == "03"
~ "Spring",
      month == "04" ~ "Spring",
      month == "05" ~ "Spring",
      month == "06"  ~ "Summer",
      month == "07"  ~ "Summer",
      month == "08"  ~ "Summer",
      month == "09" ~ "Fall",
      month == "10" ~ "Fall",
      month == "11" ~ "Fall",
      month == "12" ~ "Winter",
      month == "01" ~ "Winter",
      month == "02" ~ "Winter")
)
```

Creating columns for different times of the day

```
> cyclistic_df2 <-cyclistic_df2 %>% mutate(time_of_day = case_when(hour ==
"0" ~ "Night",
      hour == "1" ~ "Night",
      hour == "2" ~ "Night",
      hour == "3" ~ "Night",
      hour == "4" ~ "Night",
      hour == "5" ~ "Night",
      hour == "6" ~ "Morning",
      hour == "7" ~ "Morning",
      hour == "8" ~ "Morning",
      hour == "9" ~ "Morning",
      hour == "10" ~ "Morning",
      hour == "11" ~ "Morning",
      hour == "12" ~ "Afternoon",
      hour == "13" ~ "Afternoon",
      hour == "14" ~ "Afternoon",
      hour == "15" ~ "Afternoon",
      hour == "16" ~ "Afternoon",
      hour == "17" ~ "Afternoon",
      hour == "18" ~ "Evening",
      hour == "19" ~ "Evening",
      hour == "20" ~ "Evening",
      hour == "21" ~ "Evening",
      hour == "22" ~ "Evening",
      hour == "23" ~ "Evening")
)
```

Removing rows with NA values



cyclistic_df2

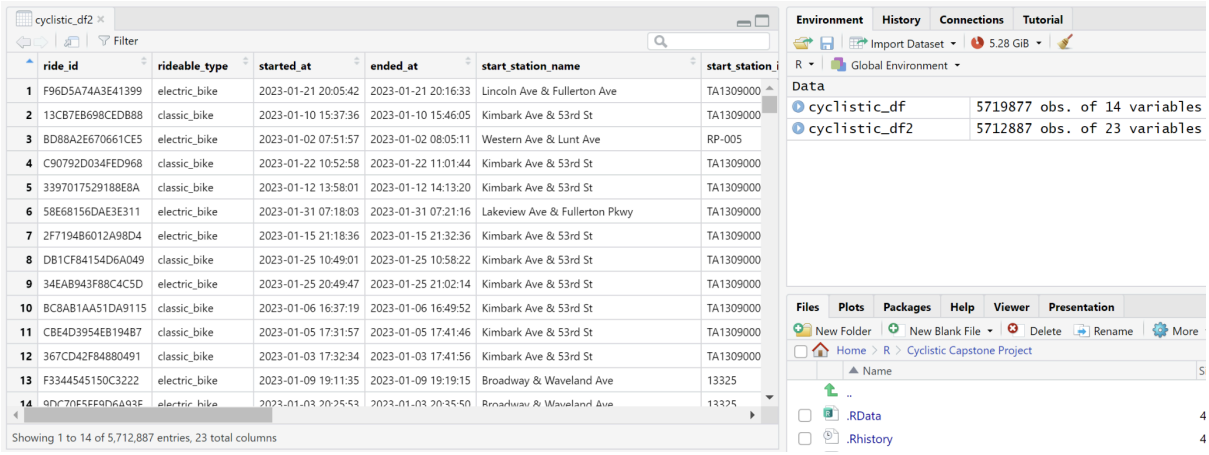
Filter

asual	ride_length	day_of_week	date	month	day	year	time	hour	season	time_of_day
	9.36666667 mins	Tuesday	2023-01-03	01	03	2023	17:32:34	17	Winter	Afternoon
	7.66666667 mins	Monday	2023-01-09	01	09	2023	19:11:35	19	Winter	Evening
	9.95000000 mins	Tuesday	2023-01-03	01	03	2023	20:25:53	20	Winter	Evening
	4.58333333 mins	Thursday	2023-01-12	01	12	2023	22:12:32	22	Winter	Evening
	6.63333333 mins	Monday	2023-01-09	01	09	2023	21:09:30	21	Winter	Evening
	2.50000000 mins	Saturday	2023-01-21	01	21	2023	09:13:54	9	Winter	Morning
	15.26666667 mins	Thursday	2023-01-05	01	05	2023	17:28:08	17	Winter	Afternoon
	16.03333333 mins	Tuesday	2023-01-17	01	17	2023	17:17:48	17	Winter	Afternoon
	48.81666667 mins	Tuesday	2023-01-03	01	03	2023	18:18:33	18	Winter	Evening
	15.03333333 mins	Monday	2023-01-02	01	02	2023	17:32:56	17	Winter	Afternoon
	4.03333333 mins	Sunday	2023-01-01	01	01	2023	18:04:58	18	Winter	Evening
	7.30000000 mins	Monday	2023-01-02	01	02	2023	12:45:48	12	Winter	Afternoon
	29.38333333 mins	Tuesday	2023-01-17	01	17	2023	17:15:15	17	Winter	Afternoon
	10.15000000 mins	Wednesday	2023-01-11	01	11	2023	17:03:25	17	Winter	Afternoon

Showing 12 to 25 of 5,719,877 entries, 23 total columns

In the screenshot above, we can see the number of rows including NA values is **5,719,877** rows. In order to clean the data and omit the NA values, the following line of code was used:

```
> cyclistic_df2 <- na.omit(cyclistic_df2)
```



cyclistic_df2

Filter

	ride_id	rideable_type	started_at	ended_at	start_station_name	start_station_i
1	F96D5A74A3E41399	electric_bike	2023-01-21 20:05:42	2023-01-21 20:16:33	Lincoln Ave & Fullerton Ave	TA1309000
2	13CB7EB698CEDB88	classic_bike	2023-01-10 15:37:36	2023-01-10 15:46:05	Kimbark Ave & 53rd St	TA1309000
3	BD88A2E670661CE5	electric_bike	2023-01-02 07:51:57	2023-01-02 08:05:11	Western Ave & Lunt Ave	RP-005
4	C90792D034FED968	classic_bike	2023-01-22 10:52:58	2023-01-22 11:01:44	Kimbark Ave & 53rd St	TA1309000
5	3397017529188E8A	classic_bike	2023-01-12 13:58:01	2023-01-12 14:13:20	Kimbark Ave & 53rd St	TA1309000
6	58E68156DAE3E311	electric_bike	2023-01-31 07:18:03	2023-01-31 07:21:16	Lakeview Ave & Fullerton Pkwy	TA1309000
7	2F7194B6012A98D4	electric_bike	2023-01-15 21:18:36	2023-01-15 21:32:36	Kimbark Ave & 53rd St	TA1309000
8	DB1CF84154D6A049	classic_bike	2023-01-25 10:49:01	2023-01-25 10:58:22	Kimbark Ave & 53rd St	TA1309000
9	34EAB943F88C4C5D	electric_bike	2023-01-25 20:49:47	2023-01-25 21:02:14	Kimbark Ave & 53rd St	TA1309000
10	BC8AB1AA51DA9115	classic_bike	2023-01-06 16:37:19	2023-01-06 16:49:52	Kimbark Ave & 53rd St	TA1309000
11	CBE4D3954EB194B7	classic_bike	2023-01-05 17:31:57	2023-01-05 17:41:46	Kimbark Ave & 53rd St	TA1309000
12	367CD42F84880491	classic_bike	2023-01-03 17:32:34	2023-01-03 17:41:56	Kimbark Ave & 53rd St	TA1309000
13	F3344545150C3222	electric_bike	2023-01-09 19:11:35	2023-01-09 19:19:15	Broadway & Waveland Ave	13325
14	9DC70F5FE9D6A93F	electric_bike	2023-01-03 20:25:53	2023-01-03 20:35:50	Broadway & Waveland Ave	13325

Showing 1 to 14 of 5,712,887 entries, 23 total columns

Environment | History | Connections | Tutorial

Import Dataset | 5.28 GiB

R | Global Environment

Data

cyclistic_df	5719877 obs. of 14 variables
cyclistic_df2	5712887 obs. of 23 variables

Files | Plots | Packages | Help | Viewer | Presentation

New Folder | New Blank File | Delete | Rename | More

Home > R > Cyclistic Capstone Project

Name

..

.RData 4

.Rhistory 4

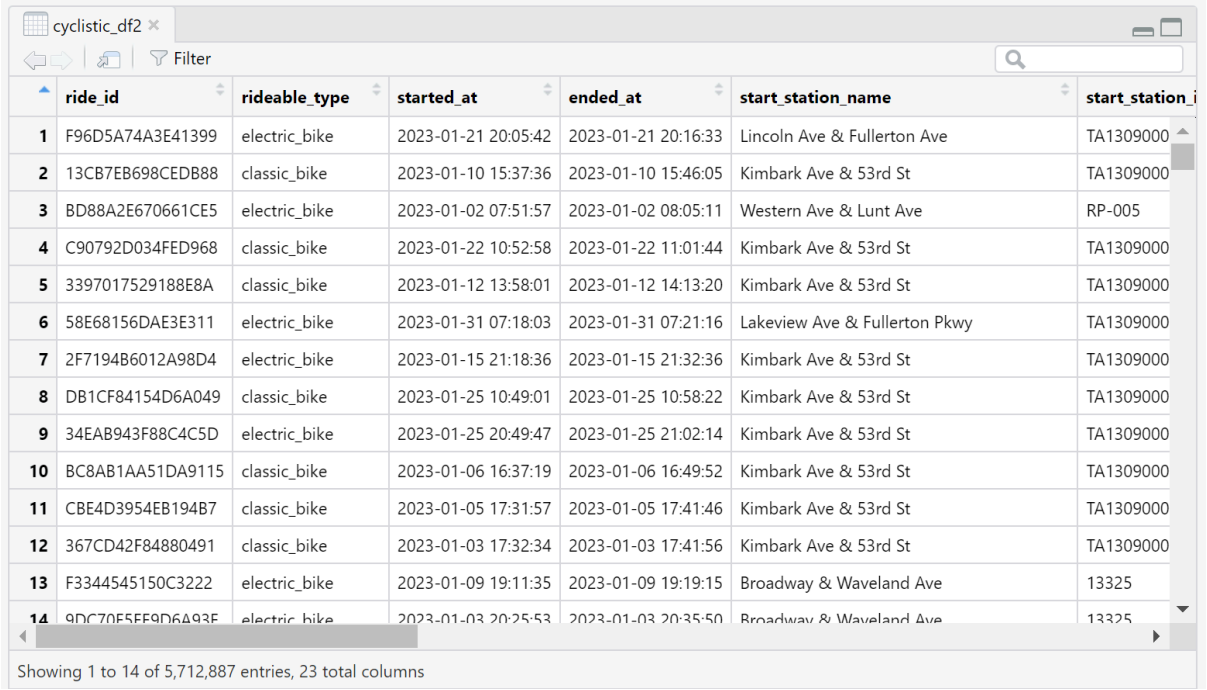
Upon removing NA values within the data frame, the number of rows have been reduced from **5,719,877** to **5,712,887**, a total of 6990 rows with NA values were removed from the data frame.

Removing duplicate rows from the data frame

Next, all duplicate rows were removed from the data frame. Before doing this, the number of rows within the data frame was **5,712,887**. To remove the duplicate rows, the following line of code was implemented:

```
> cyclistic_df2 <- distinct(cyclistic_df2)
```

After running the code, no rows were removed, as the total number of rows remained the same, which can be seen at the bottom of the screenshot below.



cyclistic_df2

Filter

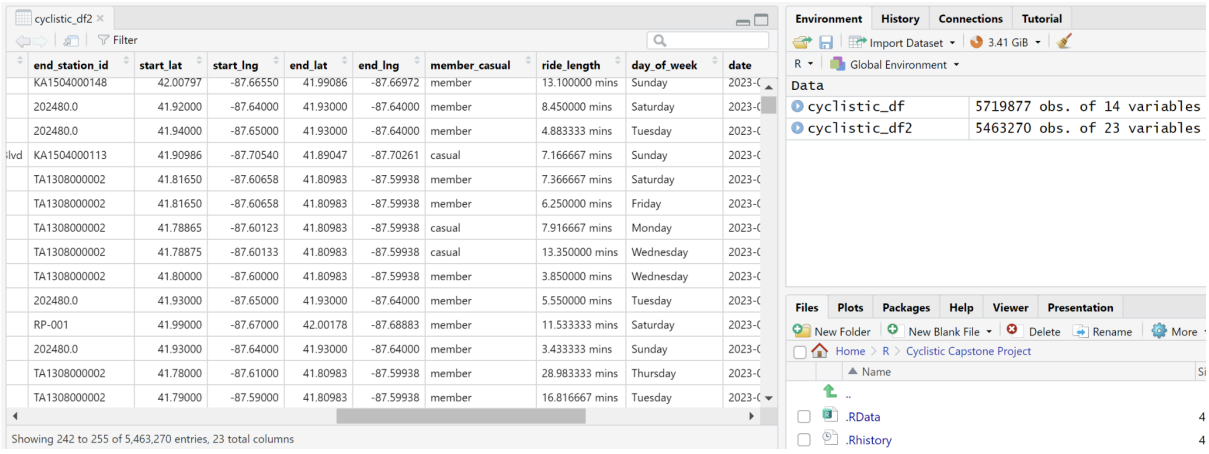
	ride_id	rideable_type	started_at	ended_at	start_station_name	start_station_i
1	F96D5A74A3E41399	electric_bike	2023-01-21 20:05:42	2023-01-21 20:16:33	Lincoln Ave & Fullerton Ave	TA1309000
2	13CB7EB698CEDB88	classic_bike	2023-01-10 15:37:36	2023-01-10 15:46:05	Kimbark Ave & 53rd St	TA1309000
3	BD88A2E670661CE5	electric_bike	2023-01-02 07:51:57	2023-01-02 08:05:11	Western Ave & Lunt Ave	RP-005
4	C90792D034FED968	classic_bike	2023-01-22 10:52:58	2023-01-22 11:01:44	Kimbark Ave & 53rd St	TA1309000
5	3397017529188E8A	classic_bike	2023-01-12 13:58:01	2023-01-12 14:13:20	Kimbark Ave & 53rd St	TA1309000
6	58E68156DAE3E311	electric_bike	2023-01-31 07:18:03	2023-01-31 07:21:16	Lakeview Ave & Fullerton Pkwy	TA1309000
7	2F7194B6012A98D4	electric_bike	2023-01-15 21:18:36	2023-01-15 21:32:36	Kimbark Ave & 53rd St	TA1309000
8	DB1CF84154D6A049	classic_bike	2023-01-25 10:49:01	2023-01-25 10:58:22	Kimbark Ave & 53rd St	TA1309000
9	34EAB943F88C4C5D	electric_bike	2023-01-25 20:49:47	2023-01-25 21:02:14	Kimbark Ave & 53rd St	TA1309000
10	BC8AB1AA51DA9115	classic_bike	2023-01-06 16:37:19	2023-01-06 16:49:52	Kimbark Ave & 53rd St	TA1309000
11	CBE4D3954EB194B7	classic_bike	2023-01-05 17:31:57	2023-01-05 17:41:46	Kimbark Ave & 53rd St	TA1309000
12	367CD42F84880491	classic_bike	2023-01-03 17:32:34	2023-01-03 17:41:56	Kimbark Ave & 53rd St	TA1309000
13	F3344545150C3222	electric_bike	2023-01-09 19:11:35	2023-01-09 19:19:15	Broadway & Waveland Ave	13325
14	9DC70F5FF9D6A93F	electric_bike	2023-01-03 20:25:53	2023-01-03 20:35:50	Broadway & Waveland Ave	13325

Showing 1 to 14 of 5,712,887 entries, 23 total columns

Removing rows with ride lengths less than 2 minutes

Sticking to the original plan, all rides that lasted less than 2 minutes were removed from the data frame, as I deemed them to be invalid rides. Rides that lasted less than 2 minutes started and ended at the same station. This means that they were either accidental or they were used for testing, which would be unfair and cause problems and bias in the data.

```
> cyclistic_df2 <- cyclistic_df2[!(cyclistic_df2$ride_length <=1.9),]
```



cyclistic_df2

Filter

end_station_id	start_lat	start_lng	end_lat	end_lng	member_casual	ride_length	day_of_week	date
KA1504000148	42.00797	-87.66550	41.99086	-87.66972	member	13.100000 mins	Sunday	2023-(
202480.0	41.92000	-87.64000	41.93000	-87.64000	member	8.450000 mins	Saturday	2023-(
202480.0	41.94000	-87.65000	41.93000	-87.64000	member	4.883333 mins	Tuesday	2023-(
KA1504000113	41.90986	-87.70540	41.89047	-87.70261	casual	7.166667 mins	Sunday	2023-(
TA1308000002	41.81650	-87.60658	41.80983	-87.59938	member	7.366667 mins	Saturday	2023-(
TA1308000002	41.81650	-87.60658	41.80983	-87.59938	member	6.250000 mins	Friday	2023-(
TA1308000002	41.78865	-87.60123	41.80983	-87.59938	casual	7.916667 mins	Monday	2023-(
TA1308000002	41.78875	-87.60133	41.80983	-87.59938	casual	13.350000 mins	Wednesday	2023-(
TA1308000002	41.80000	-87.60000	41.80983	-87.59938	member	3.850000 mins	Wednesday	2023-(
202480.0	41.93000	-87.65000	41.93000	-87.64000	member	5.550000 mins	Tuesday	2023-(
RP-001	41.99000	-87.67000	42.00178	-87.68883	member	11.533333 mins	Saturday	2023-(
202480.0	41.93000	-87.64000	41.93000	-87.64000	member	3.433333 mins	Sunday	2023-(
TA1308000002	41.78000	-87.61000	41.80983	-87.59938	member	28.983333 mins	Thursday	2023-(
TA1308000002	41.79000	-87.59000	41.80983	-87.59938	member	16.816667 mins	Tuesday	2023-(

Showing 242 to 255 of 5,463,270 entries, 23 total columns

Environment | History | Connections | Tutorial

Import Dataset | 3.41 GiB

R | Global Environment

Data

cyclistic_df	5719877 obs. of 14 variables
cyclistic_df2	5463270 obs. of 23 variables

Files | Plots | Packages | Help | Viewer | Presentation

New Folder | New Blank File | Delete | Rename | More

Home > R > Cyclistic Capstone Project

Name

.RData

.Rhistory

After running the code above, the number of rows were reduced from **5,712,887** to **5,463,270**. A total of **279,617** extra rows were removed from the data frame.

```
> cyclistic_df2 <- cyclistic_df2 %>%
+    select(-c(ride_id, start_station_id,
end_station_id,start_lat,start_lng,end_lat,end_lng))
```

Rounding the ride length in minutes to 1 decimal place

To further clean the data, I rounded the ride length to 1 decimal place, as there were too many values after the decimals. To do this, I used the following line of code:

```
> cyclistic_df2$ride_length <- round(cyclistic_df2$ride_length, digits = 1)
```

member_casual	ride_length	day_of_week
member	13.100000 mins	Sunday
member	8.450000 mins	Saturday
member	4.883333 mins	Tuesday
casual	7.166667 mins	Sunday
member	7.366667 mins	Saturday
member	6.250000 mins	Friday
casual	7.916667 mins	Monday
casual	13.350000 mins	Wednesday
member	3.850000 mins	Wednesday
member	5.550000 mins	Tuesday
member	11.533333 mins	Saturday
member	3.433333 mins	Sunday
member	28.983333 mins	Thursday
member	16.816667 mins	Tuesday

member_casual	ride_length	day_of_week
member	10.8 mins	Saturday
member	8.5 mins	Tuesday
casual	13.2 mins	Monday
member	8.8 mins	Sunday
member	15.3 mins	Thursday
member	3.2 mins	Tuesday
member	14.0 mins	Sunday
member	9.3 mins	Wednesday
member	12.4 mins	Wednesday
member	12.6 mins	Friday
member	9.8 mins	Thursday
member	9.4 mins	Tuesday
member	7.7 mins	Monday
casual	9.9 mins	Tuesday

In the screenshots above, the difference can be seen after cleaning the data. The screenshot on the left is before running the line of code, and the screenshot on the right is the result after running the code and rounding the ride length in minutes to 1 decimal place.

Analysing the data

After the data was cleaned, the next step was to analyse it. To analyse the data, I could have either continued to create visualisations in R Studio, or I could have created visualisations and a dashboard using Tableau.

```
> fwrite(cyclistic_df2,"cyclistic_df2.csv")
```

Grouping by member type

```
> cyclistic_df2 %>% group_by(member_casual) %>% count(member_casual)
# A tibble: 2 × 2
# Groups:   member_casual [2]
  member_casual       n
  <chr>           <int>
1 casual        1973009
2 member        3490261
```

By running the above line of code, I grouped the member types by casual and cyclistic members. The line of code also asks to return the count of each member type. As a result, we can see that the majority of the users are members.

Total number of rides by each type of member

```
> cyclistic_df2 %>%
+     group_by(member_casual, rideable_type) %>%
+     count(rideable_type)
# A tibble: 5 × 3
# Groups:   member_casual, rideable_type [5]
  member_casual rideable_type       n
  <chr>         <chr>           <int>
1 casual        classic_bike   852608
2 casual        docked_bike     75073
3 casual        electric_bike 1045328
4 member        classic_bike  1757166
5 member        electric_bike 1733095
```

From the above screenshot, we can see that electric bikes are the most common type of bike used by members, and since members make for the most users overall, it means that electric bikes are the most commonly used type of bike, with **1,733,095** rides in 2023. We can also see that no members used any docked bikes.

Total number of rides for each type of bike

```
> cyclistic_df2 %>%
+     group_by(rideable_type) %>%
+     count(rideable_type)
# A tibble: 3 × 2
# Groups:   rideable_type [3]
  rideable_type       n
  <chr>           <int>
1 classic_bike  2609774
2 docked_bike     75073
3 electric_bike 2778423
```

By running another line of code/calculation, we can see that the most commonly used type of bike is an electric bike.

Count of rides by casual users

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     count(hour) %>%
+     print(n = 48)
# A tibble: 48 × 3
# Groups:   member_casual [2]
   member_casual  hour      n
   <chr>         <int>  <int>
 1 casual            0  35004
 2 casual            1  22669
 3 casual            2  13680
 4 casual            3   7518
 5 casual            4   5679
 6 casual            5  10891
 7 casual            6  28878
 8 casual            7  50990
 9 casual            8  68047
10 casual            9  67193
11 casual           10  83299
12 casual           11 106086
13 casual           12 125560
14 casual           13 130980
15 casual           14 136678
16 casual           15 152484
17 casual           16 174997
18 casual           17 191178
19 casual           18 165119
20 casual           19 121873
21 casual           20  88109
22 casual           21  73855
23 casual           22  65347
24 casual           23  46895
```

Count of rides by members

25	member	0	33606
26	member	1	19942
27	member	2	11543
28	member	3	7500
29	member	4	8225
30	member	5	32321
31	member	6	100084
32	member	7	186151
33	member	8	233717
34	member	9	156896
35	member	10	141399
36	member	11	167927
37	member	12	189974
38	member	13	188771
39	member	14	191826
40	member	15	234685
41	member	16	316822
42	member	17	371634
43	member	18	294525
44	member	19	208289
45	member	20	144820
46	member	21	112311
47	member	22	83786
48	member	23	53507

Number of rides per hour

```
> cyclistic_df2 %>% count(hour) %>% head(n = 24)
   hour      n
1     0  68610
2     1  42611
3     2  25223
4     3  15018
5     4  13904
6     5  43212
7     6 128962
8     7 237141
9     8 301764
10    9 224089
11   10 224698
12   11 274013
13   12 315534
14   13 319751
15   14 328504
16   15 387169
17   16 491819
18   17 562812
19   18 459644
20   19 330162
21   20 232929
22   21 186166
23   22 149133
24   23 100402
```

After running the above line of code, we can see that the times when most rides take place are between 12pm and 7pm (i.e. the afternoon).

Count of rides by members and casual riders in the morning

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(time_of_day == "Morning") %>%
+     count(time_of_day)
# A tibble: 2 × 3
# Groups:   member_casual [2]
  member_casual time_of_day      n
  <chr>         <chr>        <int>
1 casual        Morning     404493
2 member        Morning     986174
```

Count of total rides in the morning

```
> cyclistic_df2 %>%
+     filter(time_of_day == "Morning") %>%
+     count(time_of_day)
  time_of_day       n
1     Morning 1390667
```

Count of rides by members and casual riders in the afternoon

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(time_of_day == "Afternoon") %>%
+     count(time_of_day)
# A tibble: 2 × 3
# Groups:   member_casual [2]
  member_casual time_of_day       n
  <chr>         <chr>         <int>
1 casual        Afternoon    911877
2 member        Afternoon   1493712
```

Count of total rides in the afternoon

```
> cyclistic_df2 %>%
+     filter(time_of_day == "Afternoon") %>%
+     count(time_of_day)
  time_of_day       n
1   Afternoon 2405589
```

Count of rides by members and casual riders in the evening

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(time_of_day == "Evening") %>%
+     count(time_of_day)
# A tibble: 2 × 3
# Groups:   member_casual [2]
  member_casual time_of_day      n
  <chr>         <chr>        <int>
1 casual        Evening     561198
2 member        Evening     897238
```

Count of total rides in the evening

```
> cyclistic_df2 %>%
+     filter(time_of_day == "Evening") %>%
+     count(time_of_day)
  time_of_day       n
1     Evening 1458436
```

Count of rides by members and casual riders at night

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(time_of_day == "Night") %>%
+     count(time_of_day)
# A tibble: 2 × 3
# Groups:   member_casual [2]
  member_casual time_of_day      n
  <chr>         <chr>        <int>
1 casual        Night        95441
2 member        Night       113137
```

0

Count of total rides at night

```
> cyclistic_df2 %>%
+     filter(time_of_day == "Night") %>%
+     count(time_of_day)#
  time_of_day      n
1       Night 208578
```

Total rides by members of casual riders at all times of day

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     count(time_of_day)
# A tibble: 8 × 3
# Groups:   member_casual [2]
  member_casual time_of_day       n
  <chr>         <chr>         <int>
1 casual        Afternoon    911877
2 casual        Evening      561198
3 casual        Morning      404493
4 casual        Night         95441
5 member        Afternoon   1493712
6 member        Evening      897238
7 member        Morning      986174
8 member        Night        113137
```

After running the above code, we can see that most rides took place in the afternoon.

Total rides each day of the week by members and casual riders

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     count(day_of_week)
# A tibble: 14 x 3
# Groups:   member_casual [2]
   member_casual day_of_week     n
   <chr>         <chr>       <int>
 1 casual        Friday     298828
 2 casual        Monday     225226
 3 casual        Saturday   393214
 4 casual        Sunday     321273
 5 casual        Thursday   259479
 6 casual        Tuesday    236050
 7 casual        Wednesday  238939
 8 member        Friday     506067
 9 member        Monday     471514
10 member        Saturday   451060
11 member        Sunday     389879
12 member        Thursday   561737
13 member        Tuesday    550197
14 member        Wednesday  559807
```

Total number of rides each day of the week

```
> cyclistic_df2 %>%
+     count(day_of_week)
  day_of_week      n
1      Friday 804895
2      Monday 696740
3    Saturday 844274
4      Sunday 711152
5    Thursday 821216
6     Tuesday 786247
7   Wednesday 798746
```

Count of rides per day of the month by members and casual riders

```
> View(cyclistic_df2)
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     count(day) %>%
+     print(n = 62)
# A tibble: 62 × 3
# Groups:   member_casual [2]
   member_casual day       n
   <chr>         <chr>   <int>
 1 casual        01      62665
 2 casual        02      63585
 3 casual        03      75694
 4 casual        04      82788
 5 casual        05      57810
 6 casual        06      60169
 7 casual        07      68692
 8 casual        08      65683
 9 casual        09      76283
10 casual        10      74754
11 casual        11      57325
12 casual        12      61651
13 casual        13      56862
14 casual        14      56247
15 casual        15      77396
16 casual        16      64787
17 casual        17      58485
18 casual        18      64261
19 casual        19      63774
20 casual        20      69905
21 casual        21      70990
22 casual        22      63114
23 casual        23      69970
24 casual        24      66203
25 casual        25      54095
26 casual        26      53623
27 casual        27      60218
28 casual        28      58442
29 casual        29      63739
30 casual        30      64102
31 casual        31      29697
31 casual        31      29697
32 member        01     105926
33 member        02     109432
34 member        03     115297
35 member        04     117513
36 member        05     109808
37 member        06     118788
38 member        07     125954
39 member        08     116282
40 member        09     123630
41 member        10     126937
42 member        11     113157
43 member        12     115162
44 member        13     115452
45 member        14     113364
46 member        15     127056
47 member        16     110140
48 member        17     109937
49 member        18     120938
50 member        19     112374
51 member        20     125513
52 member        21     124076
53 member        22     108368
54 member        23     112064
55 member        24     114453
56 member        25     100455
57 member        26     103745
58 member        27     113086
59 member        28     110261
60 member        29      99596
61 member        30     108700
62 member        31      62797
```

Total rides per day of the month

```
> cyclistic_df2 %>%
+     count(day) %>%
+     head(n = 31)
   day      n
1   01 168591
2   02 173017
3   03 190991
4   04 200301
5   05 167618
6   06 178957
7   07 194646
8   08 181965
9   09 199913
10  10 201691
11  11 170482
12  12 176813
13  13 172314
14  14 169611
15  15 204452
16  16 174927
17  17 168422
18  18 185199
19  19 176148
20  20 195418
21  21 195066
22  22 171482
23  23 182034
24  24 180656
25  25 154550
26  26 157368
27  27 173304
28  28 168703
29  29 163335
30  30 172802
31  31  92494
```

Count of rides per month by members and casual riders

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     count(month) %>%
+     print(n = 24)
# A tibble: 24 × 3
# Groups:   member_casual [2]
   member_casual month      n
   <chr>         <chr>  <int>
 1 casual        01     38091
 2 casual        02     41149
 3 casual        03     59243
 4 casual        04    140129
 5 casual        05    223523
 6 casual        06    288357
 7 casual        07    317158
 8 casual        08    298771
 9 casual        09    251918
10 casual        10    170239
11 casual        11     94764
12 casual        12     49667
13 member        01    141469
14 member        02    138816
15 member        03    184321
16 member        04    262936
17 member        05    352215
18 member        06    399997
19 member        07    416740
20 member        08    441427
21 member        09    389390
22 member        10    345423
23 member        11    253042
24 member        12    164485
```

Total rides per month

```
> cyclistic_df2 %>%
+     count(month)
   month      n
1     01 179560
2     02 179965
3     03 243564
4     04 403065
5     05 575738
6     06 688354
7     07 733898
8     08 740198
9     09 641308
10    10 515662
11    11 347806
12    12 214152
```

Analysing seasons

Count of rides by members and casual riders in Spring

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(season == "Spring") %>%
+     count(season)
# A tibble: 2 × 3
# Groups:   member_casual [2]
  member_casual season      n
  <chr>         <chr>   <int>
1 casual        Spring 422895
2 member        Spring 799472
```

Total rides during Spring

```
> cyclistic_df2 %>%
+     filter(season == "Spring") %>%
+     count(season)
  season       n
1 Spring 1222367
```

Count of rides by members and casual riders in Summer

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(season == "Summer") %>%
+     count(season)
# A tibble: 2 × 3
# Groups:   member_casual [2]
  member_casual season       n
  <chr>         <chr>    <int>
1 casual        Summer  904286
2 member        Summer 1258164
```

Total rides during Summer

```
> cyclistic_df2 %>%
+     filter(season == "Summer") %>%
+     count(season)
  season       n
1 Summer 2162450
```

Count of rides by members and casual riders in Fall

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(season == "Fall") %>%
+     count(season)
# A tibble: 2 × 3
# Groups:   member_casual [2]
  member_casual season      n
  <chr>         <chr>   <int>
1 casual        Fall   516921
2 member        Fall   987855
```

Total rides during Fall

```
> cyclistic_df2 %>%
+     filter(season == "Fall") %>%
+     count(season)
  season       n
1   Fall 1504776
```

Count of rides by members and casual riders in Winter

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(season == "Winter") %>%
+     count(season)
# A tibble: 2 × 3
# Groups:   member_casual [2]
  member_casual season      n
  <chr>         <chr>   <int>
1 casual        Winter 128907
2 member        Winter 444770
```

Total rides during Winter

```
> cyclistic_df2 %>%
+     filter(season == "Winter") %>%
+     count(season)
  season      n
1 Winter 573677
```

Count of rides per season by members and casual riders

```
> cyclistic_df2 %>%
+     group_by(season, member_casual) %>%
+     count(season)
# A tibble: 8 × 3
# Groups:   season, member_casual [8]
  season member_casual      n
  <chr>  <chr>          <int>
1 Fall   casual         516921
2 Fall   member         987855
3 Spring casual         422895
4 Spring member         799472
5 Summer casual         904286
6 Summer member        1258164
7 Winter casual         128907
8 Winter member         444770
```

Total number of rides each season

```
> cyclistic_df2 %>%
+     group_by(season) %>%
+     count(season)
# A tibble: 4 × 2
# Groups:   season [4]
  season       n
  <chr>    <int>
1 Fall   1504776
2 Spring 1222367
3 Summer 2162450
4 Winter  573677
```

Total average ride length

```
> cyclistic_avgRide <- mean(cyclistic_df2$ride_length)
> print(cyclistic_avgRide)
Time difference of 15.81293 mins
```

Average ride length for members and casual riders

```
> cyclistic_df2 %>% group_by( member_casual) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 2 × 2
  member_casual time
  <chr>         <drtn>
1 casual        21.46765 mins
2 member        12.61637 mins
```

Analysing types of bikes

Average ride length for each bike type by members and casual riders

```
> cyclistic_df2 %>% group_by(member_casual, rideable_type) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 5 × 3
# Groups:   member_casual [2]
  member_casual rideable_type time
  <chr>         <chr>         <drtn>
1 casual        classic_bike  26.45269 mins
2 casual        docked_bike   54.72149 mins
3 casual        electric_bike 15.01346 mins
4 member        classic_bike  13.43945 mins
5 member        electric_bike 11.78185 mins
```

Total average ride length per bike type

```
> cyclistic_df2 %>% group_by(rideable_type) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 3 × 2
  rideable_type time
  <chr>         <drtn>
1 classic_bike  17.69085 mins
2 docked_bike   54.72149 mins
3 electric_bike 12.99768 mins
```

Analysing hours of the day

Average ride length each hour by members and casual riders

```
> cyclistic_df2 %>% group_by(hour, member_casual) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean)) %>%
+     print(n=48)
# A tibble: 48 × 3
# Groups:   hour [24]
    hour member_casual time
   <int> <chr>         <drtn>
 1     0 casual        19.13875 mins
 2     0 member        11.98508 mins
 3     1 casual        18.85789 mins
 4     1 member        12.51061 mins
 5     2 casual        18.88761 mins
 6     2 member        12.73723 mins
 7     3 casual        18.58003 mins
 8     3 member        13.04777 mins
 9     4 casual        16.25774 mins
10     4 member        12.44516 mins
11     5 casual        14.24889 mins
12     5 member        10.46397 mins
13     6 casual        15.20642 mins
14     6 member        10.92776 mins
15     7 casual        14.41322 mins
16     7 member        11.38422 mins
17     8 casual        16.05870 mins
18     8 member        11.53070 mins
19     9 casual        21.29758 mins
20     9 member        11.79104 mins
21    10 casual        24.87253 mins
22    10 member        12.66355 mins
23    11 casual        25.67395 mins
24    11 member        12.96501 mins
24    11 member        12.96501 mins
25    12 casual        25.23504 mins
26    12 member        12.63356 mins
27    13 casual        24.85580 mins
28    13 member        12.59952 mins
29    14 casual        24.90518 mins
30    14 member        13.03184 mins
31    15 casual        23.34990 mins
32    15 member        12.99302 mins
33    16 casual        21.43754 mins
34    16 member        13.11935 mins
35    17 casual        20.59108 mins
36    17 member        13.40698 mins
37    18 casual        20.27984 mins
38    18 member        13.21297 mins
39    19 casual        20.09026 mins
40    19 member        12.88165 mins
41    20 casual        19.48832 mins
42    20 member        12.57602 mins
43    21 casual        18.88500 mins
44    21 member        12.45812 mins
45    22 casual        19.21422 mins
46    22 member        12.46257 mins
47    23 casual        18.73035 mins
48    23 member        12.43387 mins
```

Total average ride length per hour

```
> cyclistic_df2 %>% group_by(hour) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean)) %>%
+     print(n=24)
# A tibble: 24 × 2
    hour time
   <int> <drtn>
 1     0 15.63479 mins
 2     1 15.88736 mins
 3     2 16.07296 mins
 4     3 15.81722 mins
 5     4 14.00238 mins
 6     5 11.41791 mins
 7     6 11.88586 mins
 8     7 12.03552 mins
 9     8 12.55175 mins
10     9 14.64158 mins
11    10 17.18961 mins
12    11 17.88536 mins
13    12 17.64804 mins
14    13 17.62007 mins
15    14 17.97189 mins
16    15 17.07201 mins
17    16 16.07909 mins
18    17 15.84730 mins
19    18 15.75162 mins
20    19 15.54257 mins
21    20 15.19071 mins
22    21 15.00776 mins
23    22 15.42100 mins
24    23 15.37478 mins
```

Analysing times of the day

Average ride length by members and casual riders in the morning

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(time_of_day == "Morning") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 2 × 2
  member_casual time
  <chr>         <drtn>
1 casual        20.99754 mins
2 member        11.88995 mins
```

Total average ride length in the morning

```
> cyclistic_df2 %>%
+     filter(time_of_day == "Morning") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
         time
1 14.539 mins
```

Average ride length by members and casual riders in the afternoon

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(time_of_day == "Afternoon") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 2 × 2
  member_casual time
  <chr>         <drtn>
1 casual        23.11350 mins
2 member        13.03235 mins
```

Total average ride length in the afternoon

```
> cyclistic_df2 %>%
+     filter(time_of_day == "Afternoon") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
           time
1 16.85377 mins
```

Average ride length by members and casual riders in the evening

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(time_of_day == "Evening") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 2 × 2
  member_casual time
  <chr>         <drtn>
1 casual        19.67727 mins
2 member        12.82222 mins
```

Total average ride length in the evening

```
> cyclistic_df2 %>%
+     filter(time_of_day == "Evening") %>%
+     summarise_at(vars(ride_length),
+                   list(time = mean))
           time
1 15.46001 mins
```

Average ride length by members and casual riders in the night

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(time_of_day == "Night") %>%
+     summarise_at(vars(ride_length),
+                    list(time = mean))
# A tibble: 2 × 2
  member_casual time
  <chr>         <drtn>
1 casual        18.26261 mins
2 member        11.82379 mins
```

Total average ride length in the night

```
> cyclistic_df2 %>%
+     filter(time_of_day == "Night") %>%
+     summarise_at(vars(ride_length),
+                    list(time = mean))
           time
1 14.77006 mins
```

Average ride length for each time of day by members and casual riders

```
> cyclistic_df2 %>%
+     group_by(time_of_day, member_casual) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 8 x 3
# Groups:   time_of_day [4]
  time_of_day member_casual time
  <chr>       <chr>         <drtn>
1 Afternoon   casual        23.11350 mins
2 Afternoon   member        13.03235 mins
3 Evening     casual        19.67727 mins
4 Evening     member        12.82222 mins
5 Morning     casual        20.99754 mins
6 Morning     member        11.88995 mins
7 Night       casual        18.26261 mins
8 Night       member        11.82379 mins
```

Total average ride length for each time of day

```
> cyclistic_df2 %>%
+     group_by(time_of_day) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 4 x 2
  time_of_day time
  <chr>       <drtn>
1 Afternoon   16.85377 mins
2 Evening     15.46001 mins
3 Morning     14.53900 mins
4 Night       14.77006 mins
```

Analysing days of the week

Average ride length of members and casual riders each day of the week

```
> cyclistic_df2 %>% group_by(member_casual, day_of_week) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 14 x 3
# Groups:   member_casual [2]
   member_casual day_of_week time
   <chr>         <chr>       <drtn>
 1 casual        Friday      20.83146 mins
 2 casual        Monday      21.10516 mins
 3 casual        Saturday    24.32470 mins
 4 casual        Sunday      24.99339 mins
 5 casual        Thursday    18.72808 mins
 6 casual        Tuesday     19.20280 mins
 7 casual        Wednesday   18.37516 mins
 8 member        Friday      12.56876 mins
 9 member        Monday      11.99733 mins
10 member        Saturday    14.02675 mins
11 member        Sunday      14.07150 mins
12 member        Thursday    12.11819 mins
13 member        Tuesday     12.10176 mins
14 member        Wednesday   12.03665 mins
```

Total average ride length per day of the week

```
> cyclistic_df2 %>% group_by(day_of_week) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 7 × 2
  day_of_week time
  <chr>       <drtn>
1 Friday      15.63640 mins
2 Monday      14.94150 mins
3 Saturday    18.82294 mins
4 Sunday      19.00562 mins
5 Thursday    14.20671 mins
6 Tuesday     14.23366 mins
7 Wednesday   13.93277 mins
```

Analysing days of the month

Average ride length per day of the month by members and casual riders

```
> cyclistic_df2 %>% group_by(day, member_casual) %>%
+     summarise_at(vars(ride_length),
+     list(time = mean)) %>%
+     print(n=62)
# A tibble: 62 × 3
# Groups:   day [31]
   day   member_casual time
   <chr> <chr>         <drtn>
 1 01    casual        22.06245 mins
 2 01    member        12.81178 mins
 3 02    casual        21.98759 mins
 4 02    member        12.68139 mins
 5 03    casual        23.28592 mins
 6 03    member        13.10064 mins
 7 04    casual        23.91053 mins
 8 04    member        13.59353 mins
 9 05    casual        20.15818 mins
10 05    member        12.27409 mins
11 06    casual        20.17311 mins
12 06    member        12.19005 mins
13 07    casual        21.06000 mins
14 07    member        12.43672 mins
15 08    casual        21.64307 mins
16 08    member        12.41732 mins
17 09    casual        23.17186 mins
18 09    member        12.93975 mins
19 10    casual        22.30559 mins
20 10    member        12.72333 mins
21 11    casual        20.10437 mins
22 11    member        12.70494 mins
23 12    casual        20.90695 mins
24 12    member        12.49472 mins
25 13    casual        20.32924 mins
26 13    member        12.38954 mins
27 14    casual        19.37088 mins
28 14    member        12.08226 mins
29 15    casual        22.05663 mins
30 15    member        12.92734 mins
31 16    casual        20.94494 mins
32 16    member        12.64349 mins
33 17    casual        20.34817 mins
34 17    member        12.30696 mins
35 18    casual        21.04938 mins
36 18    member        12.64109 mins
37 19    casual        21.67382 mins
38 19    member        12.67201 mins
39 20    casual        21.33366 mins
40 20    member        12.57437 mins
41 21    casual        21.59291 mins
42 21    member        12.66334 mins
43 22    casual        21.33216 mins
44 22    member        12.66701 mins
45 23    casual        22.26548 mins
46 23    member        12.82008 mins
47 24    casual        21.77890 mins
48 24    member        12.62678 mins
49 25    casual        20.11103 mins
50 25    member        12.36652 mins
51 26    casual        20.04584 mins
52 26    member        12.46503 mins
53 27    casual        20.96446 mins
54 27    member        12.37675 mins
55 28    casual        20.92927 mins
56 28    member        12.23526 mins
57 29    casual        22.70220 mins
58 29    member        13.05412 mins
59 30    casual        21.90294 mins
60 30    member        12.67831 mins
61 31    casual        20.37323 mins
62 31    member        12.36930 mins
```

Total average ride length per day of the month

```
> cyclistic_df2 %>% group_by(day) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean)) %>%
+     print(n=31)
# A tibble: 31 × 2
   day   time
   <chr> <drtn>
 1 01    16.25024 mins
 2 02    16.10148 mins
 3 03    17.13729 mins
 4 04    17.85773 mins
 5 05    14.99325 mins
 6 06    14.87412 mins
 7 07    15.47994 mins
 8 08    15.74749 mins
 9 09    16.84413 mins
10 10    16.27486 mins
11 11    15.19302 mins
12 12    15.42789 mins
13 13    15.00956 mins
14 14    14.49934 mins
15 15    16.38327 mins
16 16    15.71807 mins
17 17    15.09929 mins
18 18    15.55863 mins
19 19    15.93110 mins
20 20    15.70775 mins
21 21    15.91306 mins
22 22    15.85622 mins
23 23    16.45069 mins
24 24    15.98065 mins
25 25    15.07722 mins
26 26    15.04819 mins
27 27    15.36072 mins
28 28    15.24703 mins
29 29    16.81914 mins
30 30    16.10025 mins
31 31    14.93912 mins
```

Analysing months of the year

Average ride length per month for members and casual riders

```
> cyclistic_df2 %>% group_by(month, member_casual) %>%
+     summarise_at(vars(ride_length),
+     list(time = mean)) %>%
+     print(n=24)
# A tibble: 24 x 3
# Groups:   month [12]
   month member_casual time
   <chr> <chr>         <drtn>
 1 01    casual        14.20029 mins
 2 01    member        10.69920 mins
 3 02    casual        16.58870 mins
 4 02    member        11.08680 mins
 5 03    casual        15.87663 mins
 6 03    member        10.82263 mins
 7 04    casual        21.34509 mins
 8 04    member        12.12082 mins
 9 05    casual        22.96509 mins
10 05    member        13.16908 mins
11 06    casual        22.59161 mins
12 06    member        13.40329 mins
13 07    casual        23.62180 mins
14 07    member        13.78144 mins
15 08    casual        22.76371 mins
16 08    member        13.71924 mins
17 09    casual        21.97156 mins
18 09    member        13.08844 mins
19 10    casual        19.75830 mins
20 10    member        12.02571 mins
21 11    casual        16.72400 mins
22 11    member        11.42800 mins
23 12    casual        15.63554 mins
24 12    member        11.30065 mins
```

Total average ride length per month of the year

```
> cyclistic_df2 %>% group_by(month) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 12 x 2
   month time
   <chr> <drtn>
 1 01    11.44191 mins
 2 02    12.34481 mins
 3 03    12.05193 mins
 4 04    15.32771 mins
 5 05    16.97226 mins
 6 06    17.25235 mins
 7 07    18.03400 mins
 8 08    17.36991 mins
 9 09    16.57790 mins
10 10    14.57852 mins
11 11    12.87096 mins
12 12    12.30601 mins
```

Analysing the seasons

Average ride length for members and casual riders in spring

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(season == "Spring") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 2 × 2
  member_casual time
  <chr>         <drtn>
1 casual        21.43527 mins
2 member        12.28334 mins
```

Total average ride length in spring

```
> cyclistic_df2 %>%
+     filter(season == "Spring") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
           time
1 15.44958 mins
```

Average ride length for members and casual riders in summer

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(season == "Summer") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 2 × 2
  member_casual time
  <chr>         <drtn>
1 casual        23.00979 mins
2 member        13.63939 mins
```

Total average ride length in summer

```
> cyclistic_df2 %>%
+     filter(season == "Summer") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
           time
1 17.55787 mins
```

Average ride length for members and casual riders in fall

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(season == "Fall") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 2 × 2
  member_casual time
  <chr>         <drtn>
1 casual        20.28065 mins
2 member        12.29151 mins
```

Average ride length during fall

```
> cyclistic_df2 %>%
+     filter(season == "Fall") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
           time
1 15.03594 mins
```

Average ride length for members and casual riders in winter

```
> cyclistic_df2 %>%
+     group_by(member_casual) %>%
+     filter(season == "Winter") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 2 × 2
  member_casual time
  <chr>         <drtn>
1 casual        15.5157 mins
2 member        11.0426 mins
```

Average ride length during winter

```
> cyclistic_df2 %>%
+     filter(season == "Winter") %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
           time
1 12.04772 mins
```

Average ride length per season by members and casual riders

```
> cyclistic_df2 %>%
+     group_by(season, member_casual) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 8 x 3
# Groups:   season [4]
  season member_casual time
  <chr>  <chr>         <drtn>
1 Fall   casual        20.28065 mins
2 Fall   member        12.29151 mins
3 Spring casual        21.43527 mins
4 Spring member        12.28334 mins
5 Summer casual        23.00979 mins
6 Summer member        13.63939 mins
7 Winter casual        15.51570 mins
8 Winter member        11.04260 mins
```

Total ride length per season

```
> cyclistic_df2 %>%
+     group_by(season) %>%
+     summarise_at(vars(ride_length),
+                  list(time = mean))
# A tibble: 4 x 2
  season time
  <chr>  <drtn>
1 Fall   15.03594 mins
2 Spring 15.44958 mins
3 Summer 17.55787 mins
4 Winter 12.04772 mins
```

Analysing most popular start stations

Most popular start station for casual riders

```
> cyclistic_df2%>%
+     group_by(member_casual,start_station_name) %>%
+     dplyr::summarise(number_of_ride = n()) %>%
+     filter(start_station_name != "", "casual"== member_casual) %>%
+     arrange(-number_of_ride) %>%
+     head(n=30) %>%
+     select(-member_casual)
`summarise()` has grouped output by 'member_casual'. You can override using the
`.groups` argument.
Adding missing grouping variables: `member_casual`
# A tibble: 30 × 3
# Groups:   member_casual [1]
   member_casual start_station_name                number_of_ride
   <chr>         <chr>                                      <int>
 1 casual        Streeter Dr & Grand Ave                    44309
 2 casual        DuSable Lake Shore Dr & Monroe St          29364
 3 casual        Michigan Ave & Oak St                      21927
 4 casual        DuSable Lake Shore Dr & North Blvd         19668
 5 casual        Millennium Park                            19365
 6 casual        Shedd Aquarium                             17135
 7 casual        Theater on the Lake                        15837
 8 casual        Dusable Harbor                             14902
 9 casual        Wells St & Concord Ln                      11851
10 casual        Adler Planetarium                          11514
# ℹ 20 more rows
# ℹ Use `print(n = ...)` to see more rows
```

Most popular start station for members

```
> cyclistic_df2 %>%
+     group_by(member_casual,start_station_name) %>%
+     dplyr::summarise(number_of_ride = n()) %>%
+     filter(start_station_name != "", "member" == member_casual) %>%
+     arrange(-number_of_ride) %>%
+     head(n=30) %>%
+     select(-member_casual)
`summarise()` has grouped output by 'member_casual'. You can override using the
`.groups` argument.
Adding missing grouping variables: `member_casual`
# A tibble: 30 × 3
# Groups:   member_casual [1]
   member_casual start_station_name           number_of_ride
   <chr>         <chr>                                 <int>
 1 member        Kingsbury St & Kinzie St              25282
 2 member        Clinton St & Washington Blvd          25073
 3 member        Clark St & Elm St                     24288
 4 member        Wells St & Concord Ln                 20632
 5 member        Wells St & Elm St                     19658
 6 member        Clinton St & Madison St               19587
 7 member        University Ave & 57th St              18816
 8 member        Broadway & Barry Ave                  18419
 9 member        Loomis St & Lexington St              18218
10 member        State St & Chicago Ave                17352
# ℹ 20 more rows
# ℹ Use `print(n = ...)` to see more rows
```

Tableau Visualisations