Case Study of Cyclistic Bike Sharing Company

Technical Document

Stephen Hartley

Introduction

This case study is the Google Data Analytics Capstone Project that can be found on coursera.org and is part of the Google Data Analytics Professional Certificate course. This case study is for the fictitious Bike Sharing company called Cyclistic based on the Chicago Divvy Bike sharing company. The data analysis will follow the 6-step process of Ask, Prepare, Process, Analyze, Share, and Act. Along the way, I will be referring to the Case Study Roadmap guiding questions and key tasks.

Background

In 2016, Cyclistic launched a successful bike-share offering. Since then, the program has grown to a fleet of over 6,000 bicycles that are geo-tracked and locked into a network of over 1500 stations across Chicago. The bikes can be unlocked from one station and returned to any other station in the system anytime. Customers who purchase single-ride or full-day passes are referred to as casual riders. Customers who purchase annual memberships are Cyclistic members.

Software tools

Excel 2021
RStudio 2023.09.0 build 463
R 4.3.1
Tidyverse 2.0.0
Dplyr 1.1.1 Forcats 1.0.0 Ggplot2 3.4.2 Lubridate 1.9.2
Purrr 1.0.1 Readr 2.1.4 Stringr 1.5.0 Tibble 3.2.1 Tidyr 1.3.0

The Data

The data for this case study has been provided by the Motivate International Inc. under license from the Divvy Bikes and Scooters, LLC. Here are the links to the data and license. I am using the data for the last 12 months - July 2022 to June 2023. For privacy all customer data was never included in the date files.

Downloaded the 12 monthly zip files and unzipped the CSV files into a separate folder, these files when unzipped ranged in size from 36MB to 141MB or 181 thousand rows to over 800 thousand rows. Each file was named with year-month-tripdata.csv. Each row in the file described a rental trip by either a member or casual customer with a starting date timestamp and ending date timestamp, along with station names and/or geo tracking Latitude and Longitude. At first look these files they were too large to combine into one spreadsheet with some missing data, like station names and geo data.

Starting with Excel

Loaded each file into Excel separately and did the following steps

- Removed columns ride_id, start_station_id, and end_station_id.
- Formatted the Geo data start_lat, start_lng, end_lat, and end_lng to only have 5 decimal places.
- Saved the file adding a _5 at the end of it and reloaded it. This makes the formatting permanent.
- Combined the start_lat and start_lng columns into one column start_lat_lng using the =CONCAT(K2,",",L2) function. Then did the same for the end_lat and end_lng.
- Remove the columns start_lat, start_lng, end_lat, and end_lng.

- Created a separate spreadsheet with the start_station_name and the start_lat_lng columns, this is the beginning of our master station name spreadsheet. Will be adding several other months' worth of station into to create a full master file.
- Filter the end_station_name and end_lat_lng rows to find where both are empty, if both are empty then there is no way fill in the missing data and remove these rows. Repeat for start_lat_lng but I did not find any missing start_lat_lng rows.
- Saving the file with a _5_temp suffix (ex 202212-divvy-tripdata_5_temp.csv)

Repeated these steps on all the months data files

Now it's time to fill in the missing station names using our new master station file.

First need to remove the duplicate entries.

Using the Filter command to remove duplicate rows and sort ascending.

Going back to the individual month's files use the following lookup command to populate the missing station name fields

Filter the start_station_name column to show empty rows then copy in the VLookup command adjusting the cell numbers and start_lat_lng columns to get started.

```
=vlookup($P3,'file:///c:/Users/Steve/Documents/Case Study 1 - Cyclistic/stati
on_names.csv'#$station_names.$a$2:$C$165196,3,0)
```

Then repeat for end_station_name.

To finish preparing the spreadsheets

- Remove the start_lat_lng and end_lat_lng columns.
- Add columns ride_length, ride_length_minutes, nday_of_week, month, time, hour
- ride_length = end_at – start_at =C2-B2
- ride_length_minutes = ride_length * 1440 =G2*1440
- nday_of_week = WEEKDAY(B2,1)
- month = number of the file
- time = RIGHT(C2,8) and format it as time HH:MM:SS
- hour = LEFT(K2,2)

Save file without divvy and _5_temp suffix (ex 202212-tripdata.csv)

Repeat for the other 11 files.

Switch to using RStudio to prepare and analyze the data

Load needed Library files

```
library("tidyverse")
```

```
## ── Attaching core tidyverse packages ──────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.1     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.2     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
```

```
options(dplyr.summarise.inform = FALSE)
```

Reading individual data files

```
d01_trips <- read_csv("raw_data/202301-tripdata.csv", show_col_types = FALSE)
d02_trips <- read_csv("raw_data/202302-tripdata.csv", show_col_types = FALSE)
d03_trips <- read_csv("raw_data/202303-tripdata.csv", show_col_types = FALSE)
d04_trips <- read_csv("raw_data/202304-tripdata.csv", show_col_types = FALSE)
d05_trips <- read_csv("raw_data/202305-tripdata.csv", show_col_types = FALSE)
d06_trips <- read_csv("raw_data/202306-tripdata.csv", show_col_types = FALSE)
d07_trips <- read_csv("raw_data/202207-tripdata.csv", show_col_types = FALSE)
d08_trips <- read_csv("raw_data/202208-tripdata.csv", show_col_types = FALSE)
d09_trips <- read_csv("raw_data/202209-tripdata.csv", show_col_types = FALSE)
d10_trips <- read_csv("raw_data/202210-tripdata.csv", show_col_types = FALSE)
d11_trips <- read_csv("raw_data/202211-tripdata.csv", show_col_types = FALSE)
d12_trips <- read_csv("raw_data/202212-tripdata.csv", show_col_types = FALSE)
```

Combine data into Quarterly data frames

```
q1_trips <- bind_rows(d01_trips, d02_trips, d03_trips)
q2_trips <- bind_rows(d04_trips, d05_trips, d06_trips)
q3_trips <- bind_rows(d07_trips, d08_trips, d09_trips)
q4_trips <- bind_rows(d10_trips, d11_trips, d12_trips)
```

Combine data into yearly data frame

```
all_trips <- bind_rows(q1_trips, q2_trips, q3_trips, q4_trips)
summary(all_trips)
```

```
##  rideable_type       started_at          ended_at         start_station_na
me
##  Length:5696059     Length:5696059     Length:5696059     Length:5696059
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##  end_station_name   member_casual       ride_length         nday_of_week
##  Length:5696059     Length:5696059     Min.   : 0.000012   Min.   :1.000
##  Class :character   Class :character   1st Qu.: 0.004167   1st Qu.:2.000
##  Mode  :character   Mode  :character   Median : 0.006944   Median :4.000
##                                        Mean   : 0.010788   Mean   :4.153
##                                        3rd Qu.: 0.012014   3rd Qu.:6.000
##                                        Max.   :22.246528   Max.   :7.000
##      month        ride_length_minutes     time               hour
##  Min.   : 1.000   Min.   :    1.00    Length:5696059    Min.   : 0.00
##  1st Qu.: 5.000   1st Qu.:    6.00    Class1:hms        1st Qu.:11.00
##  Median : 7.000   Median :   10.00    Class2:difftime   Median :15.00
##  Mean   : 6.968   Mean   :   15.54    Mode  :numeric    Mean   :14.15
##  3rd Qu.: 9.000   3rd Qu.:   17.00                      3rd Qu.:18.00
##  Max.   :12.000   Max.   :32035.00                      Max.   :23.00
```

Some quick calculations on the data

Average Ride Length for Casual and Member customers

```
aggregate(all_trips$ride_length_minutes ~ all_trips$member_casual, FUN = mean
)
```

```
##   all_trips$member_casual all_trips$ride_length_minutes
## 1                  casual                       20.73134
## 2                  member                       12.24034
```

Create The grand total mean, median, main, and max number of trips

```
master_df <- all_trips %>%
  summarise(number_of_ride = n(),
            mean_rides = mean(ride_length_minutes),
            median_rides = median(ride_length_minutes),
            max_rides = max(ride_length_minutes),
            min_rides = min(ride_length_minutes),
  .groups = 'drop') %>%
  as.data.frame()

# Save data frame
write_csv(master_df,"reports/master_results.csv")
```

Average ride time by each day for members vs casual users

```
aggregate(all_trips$ride_length_minutes ~ all_trips$member_casual + all_trips
$nday_of_week, FUN = mean)
```

```
##    all_trips$member_casual all_trips$nday_of_week all_trips$ride_length_mi
nutes
## 1                   casual                       1                 23.80078
## 2                   member                       1                 13.52368
```

```
## 3                casual                2              20.65447
## 4                member                2              11.66975
## 5                casual                3              18.38439
## 6                member                3              11.66924
## 7                casual                4              17.90374
## 8                member                4              11.72555
## 9                casual                5              18.22342
## 10               member                5              11.78636
## 11               casual                6              20.00501
## 12               member                6              12.13942
## 13               casual                7              23.62152
## 14               member                7              13.73047
```

Create new column for day_of_week from the Number day of week

```
all_trips$day_of_week <- as.numeric(all_trips$nday_of_week)
```

Add the words to the column day_of_week

```
all_trips$day_of_week[all_trips$nday_of_week == '1'] <- 'Sunday'
all_trips$day_of_week[all_trips$nday_of_week == '2'] <- 'Monday'
all_trips$day_of_week[all_trips$nday_of_week == '3'] <- 'Tuesday'
all_trips$day_of_week[all_trips$nday_of_week == '4'] <- 'Wednesday'
all_trips$day_of_week[all_trips$nday_of_week == '5'] <- 'Thursday'
all_trips$day_of_week[all_trips$nday_of_week == '6'] <- 'Friday'
all_trips$day_of_week[all_trips$nday_of_week == '7'] <- 'Saturday'
```

Average ride time by each day for members vs casual users

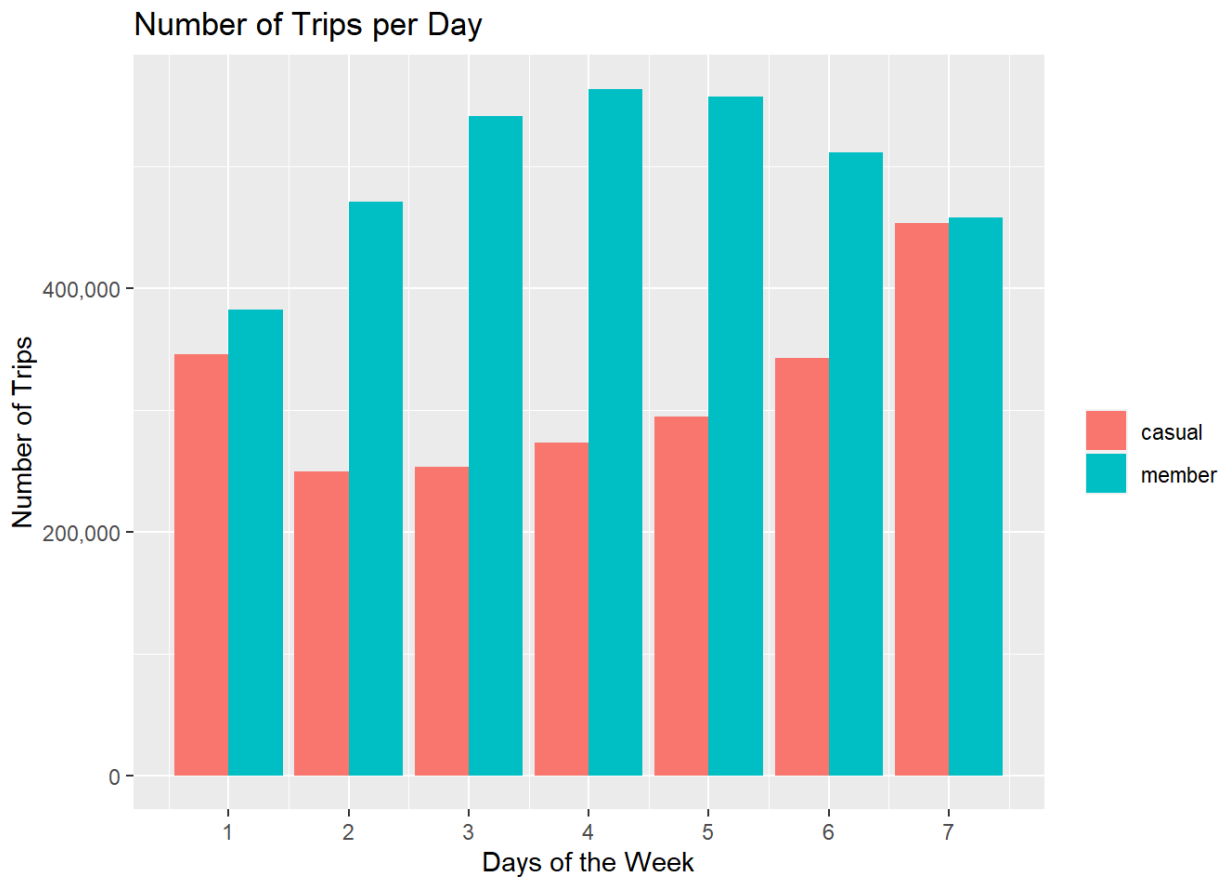
```
# Now, let's run the average ride time by each day for members vs casual user
s

aggregate(all_trips$ride_length_minutes ~ all_trips$member_casual + all_trips
$day_of_week, FUN = mean)
```

```
##    all_trips$member_casual all_trips$day_of_week all_trips$ride_length_min
utes
## 1                  casual                Friday                 20.00501
## 2                  member                Friday                 12.13942
## 3                  casual                Monday                 20.65447
## 4                  member                Monday                 11.66975
## 5                  casual              Saturday                 23.62152
## 6                  member              Saturday                 13.73047
## 7                  casual                Sunday                 23.80078
## 8                  member                Sunday                 13.52368
## 9                  casual              Thursday                 18.22342
## 10                 member              Thursday                 11.78636
## 11                 casual               Tuesday                 18.38439
## 12                 member               Tuesday                 11.66924
## 13                 casual             Wednesday                 17.90374
## 14                 member             Wednesday                 11.72555
```

Create bar chart for number of rides per day of the week

```
# Let's visualize the number of rides by rider type
all_trips %>%
  group_by(member_casual, nday_of_week) %>%
  summarise(number_of_rides = n()
            ,average_duration_minutes = mean(ride_length_minutes)) %>%
  arrange(member_casual, nday_of_week)  %>%
  ggplot(aes(x = nday_of_week, y = number_of_rides, fill = member_casual)) +
  geom_col(position = "dodge") + scale_x_continuous(breaks = c(1,2,3,4,5,6,7)
) +
  scale_y_continuous(labels = scales::comma) +
      labs(title = "Number of Trips per Day", fill = "") +
      xlab("Days of the Week") +
      ylab("Number of Trips")
```

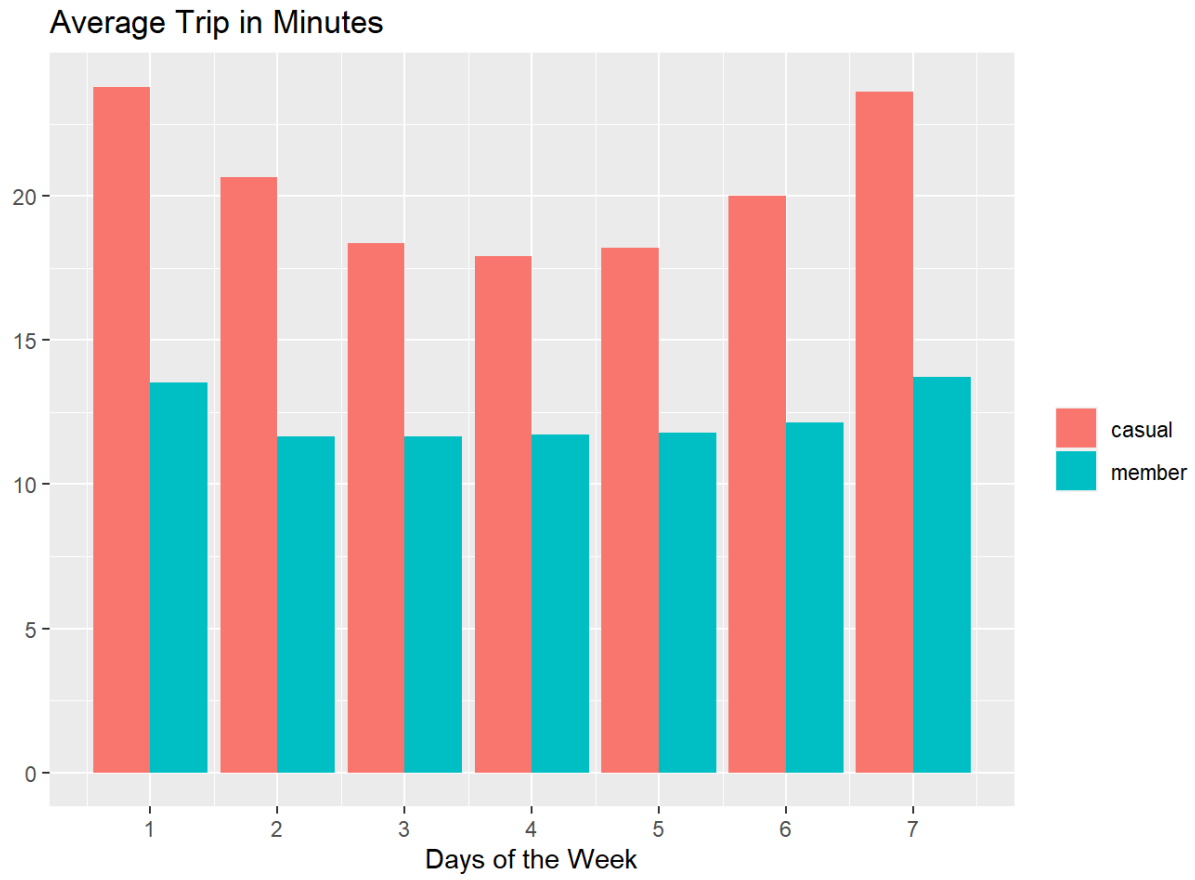



```
ggsave("reports/number_of_trips_per_day.png")
```

Create bar chart for Average Trip in Minutes per day of the week

```
# Let's create a visualization for average duration
all_trips %>%
  group_by(member_casual, nday_of_week, day_of_week) %>%
  summarise(number_of_rides = n(),
  average_duration_minutes = mean(ride_length_minutes)) %>%
  arrange(member_casual, nday_of_week)  %>%
  ggplot(aes(x = nday_of_week, y = average_duration_minutes,
  fill = member_casual)) +
  geom_col(position = "dodge") +
  scale_x_continuous(breaks = c(1,2,3,4,5,6,7)) +
  scale_y_continuous(labels = scales::comma) +
```

```
  labs(title = "Average Trip in Minutes", x = "Days of the Week", y = "",
  fill = "")
```




```
ggsave("reports/average_trip_duration.png")
```

Summary data analysis by month and day of week

```
day_week_df <- all_trips %>% group_by(month, nday_of_week, member_casual) %>%
  summarise(number_of_ride = n(),
            mean_rides = mean(ride_length_minutes),
            median_rides = median(ride_length_minutes),
            max_rides = max(ride_length_minutes),
            min_rides = min(ride_length_minutes),
  .groups = 'drop') %>%
  as.data.frame()
```

```
# Save data frame
write_csv(day_week_df,"reports/sum_memcas_day_week_df.csv")
```

Calculate the number of rides per hour per day of the week for member and/or casual

```
hour_day_df <- all_trips %>% group_by(hour, nday_of_week, member_casual) %>%
  summarise(number_of_ride = n(),
  .groups = 'drop') %>%
  as.data.frame()

write_csv(hour_day_df,"reports/hour_day_df_results.csv")
```

Create data frames that show number of rides by start/ending station and day of the week

```
ride_start_station_name_df <- all_trips %>%
  group_by(member_casual, start_station_name, ) %>%
  summarise(number_of_ride = n(),   .groups = 'drop') %>%
  as.data.frame()

# Save data frame
write_csv(ride_start_station_name_df,"reports/ride_start_station_name_df.csv"
)

ride_end_station_name_df <- all_trips %>%
  group_by(member_casual, end_station_name) %>%
  summarise(number_of_ride = n(), .groups = 'drop') %>%
  as.data.frame()

# Save data frame
```

```
write_csv(ride_end_station_name_df,"reports/ride_end_station_name_df.csv")
```

Save the complete data frame as a file

```
write_csv(all_trips,"all_trips.csv")
```

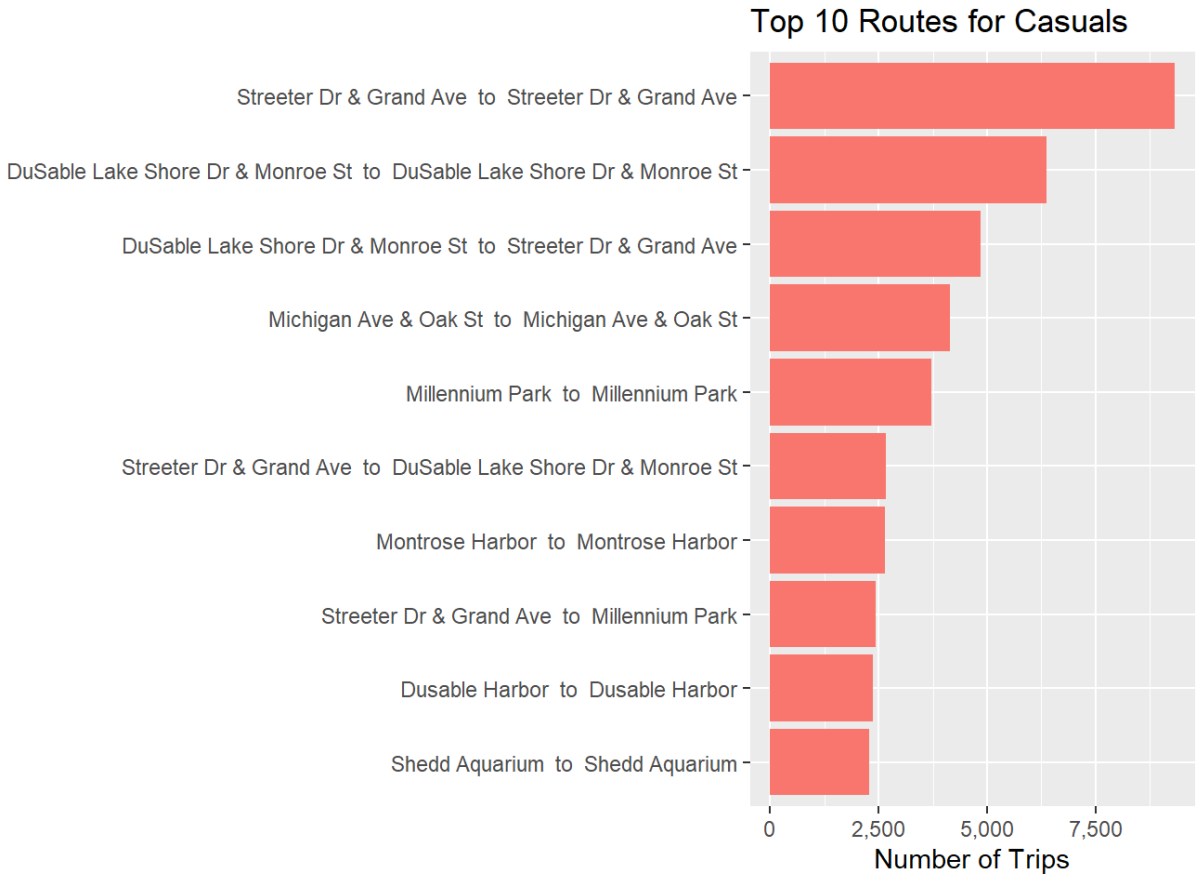
Create another data frame without the #N/A station names - not needed for top ten charts

```
all_trips_v2 <- all_trips[!(all_trips$start_station_name == "#N/A" | all_trips$end_sta
tion_name == "#N/A") ,]
all_trips_v2
```

```
## # A tibble: 4,785,847 × 13
##    rideable_type started_at    ended_at      start_station_name end_station_name
##    <chr>         <chr>         <chr>         <chr>              <chr>
##  1 electric_bike 1/1/2023 0:01 1/1/2023 0:02 Hyde Park Blvd & … Hyde Park Blvd …
##  2 classic_bike  1/1/2023 0:02 1/1/2023 0:29 Fairbanks Ct & Gr… New St & Illino…
##  3 electric_bike 1/1/2023 0:03 1/1/2023 0:07 Sheridan Rd & Loy… Sheridan Rd & L…
##  4 classic_bike  1/1/2023 0:04 1/1/2023 0:13 Leavitt St & Lawr… Broadway & Argy…
##  5 electric_bike 1/1/2023 0:04 1/1/2023 0:16 Clark St & Montro… Clark St & Mont…
##  6 classic_bike  1/1/2023 0:04 1/1/2023 0:31 State St & Randol… Indiana Ave & R…
##  7 electric_bike 1/1/2023 0:05 1/1/2023 0:21 Wabash Ave & Gran… Streeter Dr & G…
##  8 classic_bike  1/1/2023 0:06 1/1/2023 0:29 Fairbanks Ct & Gr… New St & Illino…
##  9 electric_bike 1/1/2023 0:07 1/1/2023 0:13 Streeter Dr & Gra… Streeter Dr & G…
## 10 electric_bike 1/1/2023 0:09 1/1/2023 0:14 Broadway & Belmon… Pine Grove Ave …
## # ℹ 4,785,837 more rows
## # ℹ 8 more variables: member_casual <chr>, ride_length <dbl>,
## #   nday_of_week <dbl>, month <dbl>, ride_length_minutes <dbl>, time <time>,
## #   hour <dbl>, day_of_week <chr>
```

Create Top 10 Routes for Casual Customers

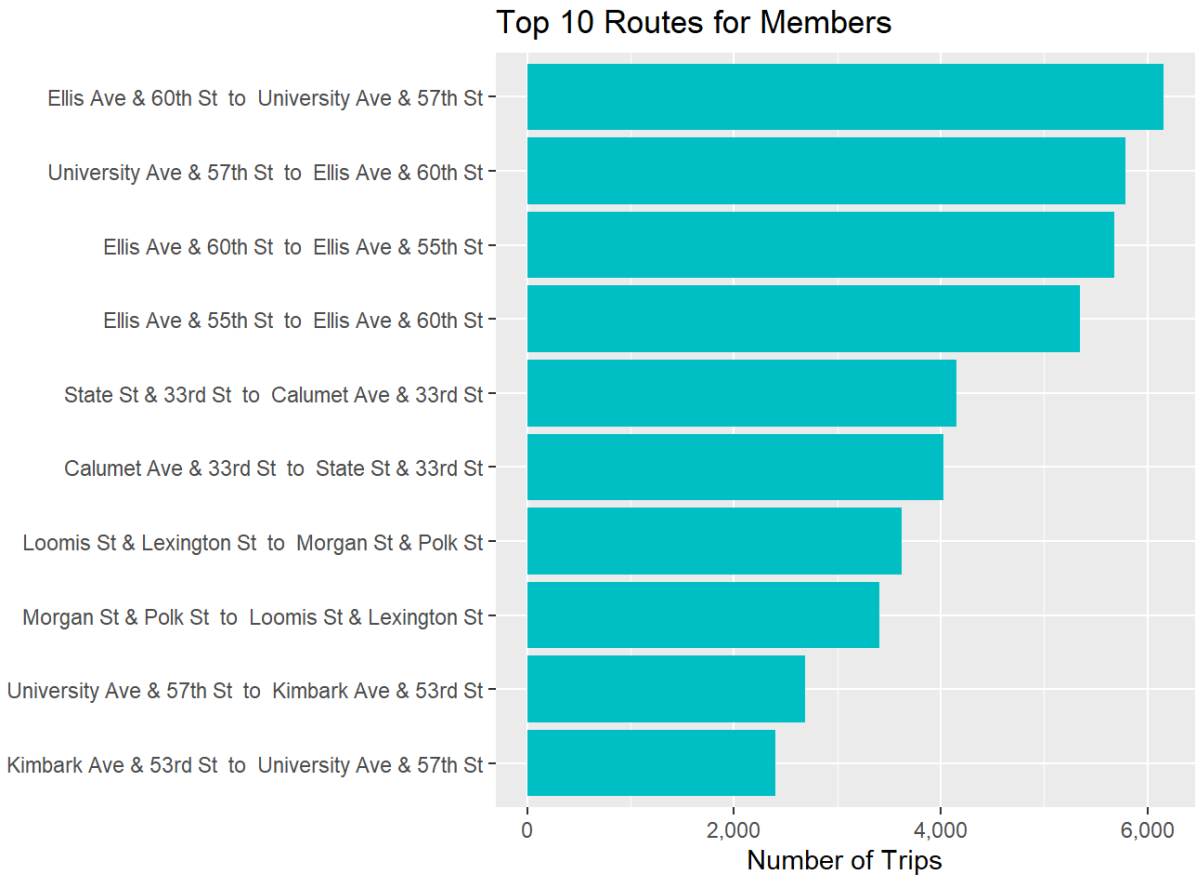
```
all_trips_v2 %>%
 mutate(route = paste(start_station_name, " to ", end_station_name)) %>%
 group_by(member_casual, route) %>%
 summarize(number_of_rides = n()) %>%
 arrange(member_casual, desc(number_of_rides)) %>%   top_n(10) %>%
 filter(member_casual == "casual") %>%
 ggplot(mapping = aes( x = number_of_rides, y = reorder(route, number_of_rides))) +
 geom_col(position = "dodge",
 fill = "#F8766D") + scale_x_continuous(labels = scales::comma) +
 labs(title = "Top 10 Routes for Casuals", x = "Number of Trips", y = "")
```




```
# Save data frame
ggsave("reports/top_routes_casuals.png")
```

Create Top 10 Routes for Member Customers

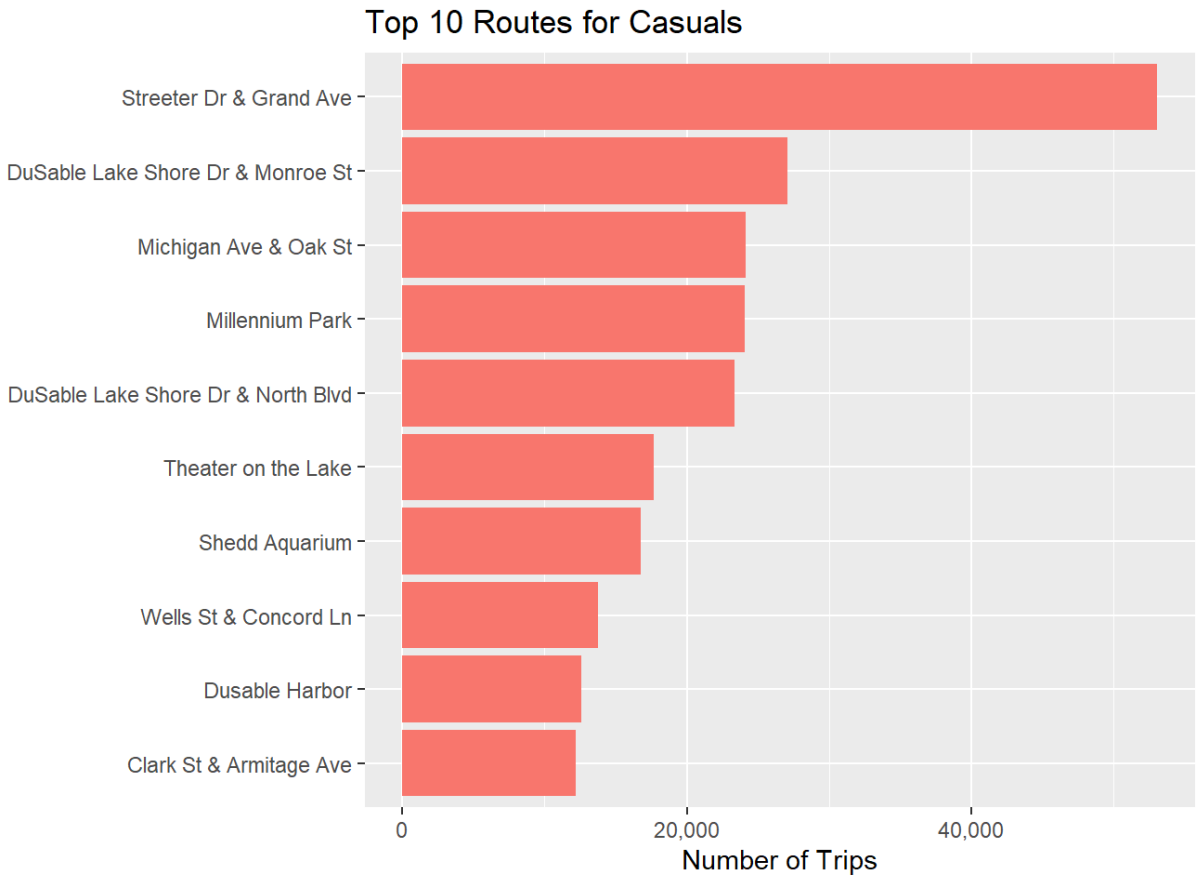
```
all_trips_v2 %>%
  mutate(route = paste(start_station_name, " to ", end_station_name)) %>%
  group_by(member_casual, route) %>%
  summarize(number_of_rides = n()) %>%
  arrange(member_casual, desc(number_of_rides)) %>%   top_n(10) %>%
  filter(member_casual == "member") %>%
  ggplot(mapping = aes( x = number_of_rides, y = reorder(route, number_of_rides))) +
  geom_col(position = "dodge",
  fill = "#00BFC4") + scale_x_continuous(labels = scales::comma) +
  labs(title = "Top 10 Routes for Members", x = "Number of Trips", y = "")
```




```
# Save data frame
ggsave("reports/top_routes_members.png")
```

Create Top 10 Ending Station for Casuals

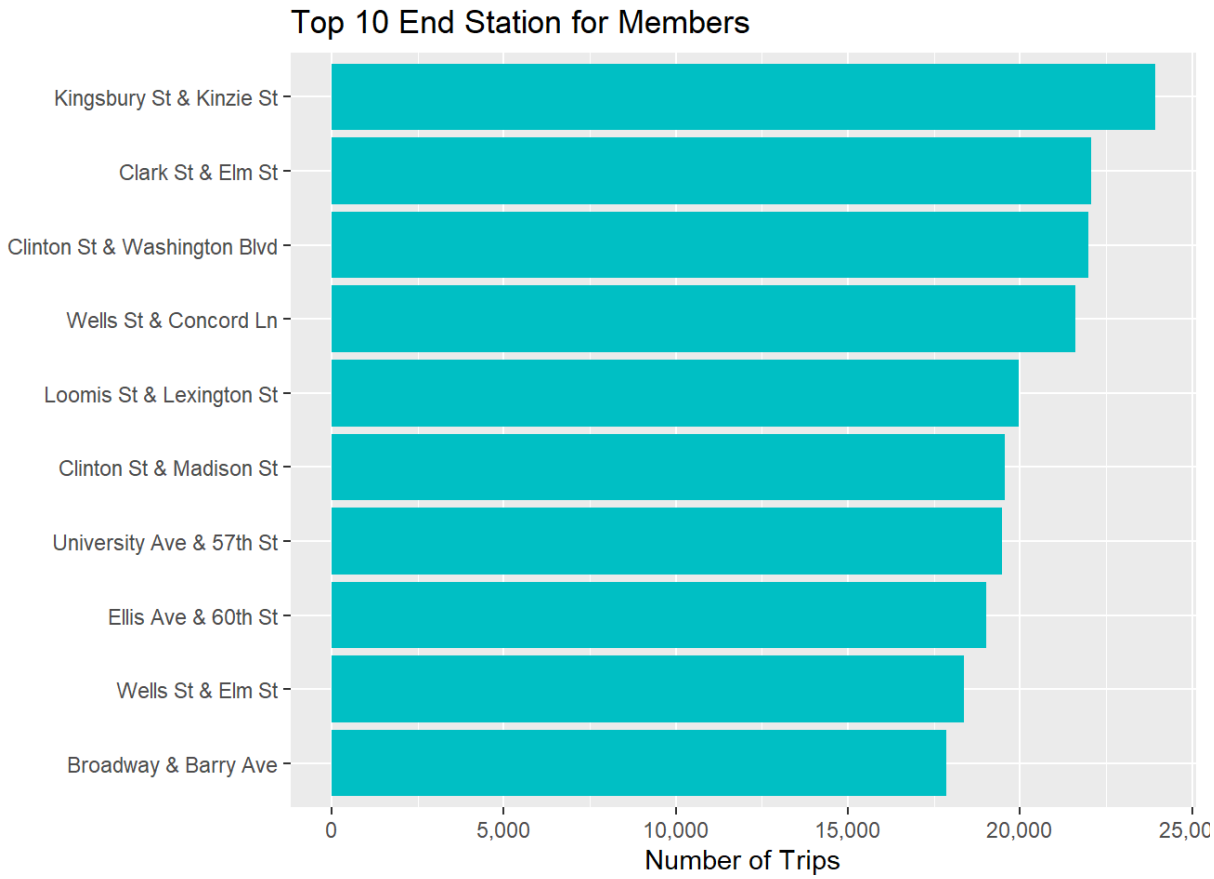
```
all_trips_v2 %>%
  group_by(member_casual, end_station_name) %>%
  summarize(number_of_rides = n()) %>%
  arrange(member_casual, desc(number_of_rides)) %>%   top_n(10) %>%
  filter(member_casual == "casual") %>%
  ggplot(mapping = aes( x = number_of_rides,
  y = reorder(end_station_name, number_of_rides))) +
  geom_col(position = "dodge",
  fill = "#F8766D") + scale_x_continuous(labels = scales::comma) +
  labs(title = "Top 10 Routes for Casuals", x = "Number of Trips", y = "")
```




```
# Save data frame
ggsave("reports/top_end_routes_casuals.png")
```

Create Top 10 Ending Station for Members

```
all_trips_v2 %>%
  group_by(member_casual, end_station_name) %>%
  summarize(number_of_rides = n()) %>%
  arrange(member_casual, desc(number_of_rides)) %>%   top_n(10) %>%
  filter(member_casual == "member") %>%
  ggplot(mapping = aes( x = number_of_rides,
  y = reorder(end_station_name, number_of_rides))) +
  geom_col(position = "dodge",
  fill = "#00BFC4") + scale_x_continuous(labels = scales::comma) +
  labs(title = "Top 10 End Station for Members", x = "Number of Trips", y = "")
```




```
# Save data frame
ggsave("reports/top_end_routes_members.png")
```

Create Joy ride statistics

```
joy_ride_totals_df <- all_trips_v2 %>%
    filter(start_station_name == end_station_name) %>%
    group_by(member_casual,  ) %>%
    summarise(number_of_ride = n(),
    mean_ride_length_minutes = mean(ride_length_minutes),
    .groups = 'drop') %>%
    as.data.frame()

# Save data frame
write_csv(joy_ride_totals_df,"reports/joy_ride_totals_df.csv")

joy_ride_grand_totals_df <- all_trips_v2 %>%
  filter(start_station_name == end_station_name) %>%
  summarise(number_of_ride = n(),
  mean_ride_length_minutes = mean(ride_length_minutes),
  .groups = 'drop') %>%
  as.data.frame()

# Save data frame
write_csv(joy_ride_grand_totals_df,"reports/joy_ride_grand_totals_df.csv")

joy_station_name_df <- all_trips_v2 %>%
    filter(start_station_name == end_station_name) %>%
    group_by(member_casual, start_station_name, ) %>%
    summarise(number_of_ride = n(),
    mean_ride_length_minutes = mean(ride_length_minutes),
    .groups = 'drop') %>%
    as.data.frame()

# Save data frame
write_csv(joy_station_name_df,"reports/joy_station_name_df.csv")
```

All done - Clean up Remove data frames

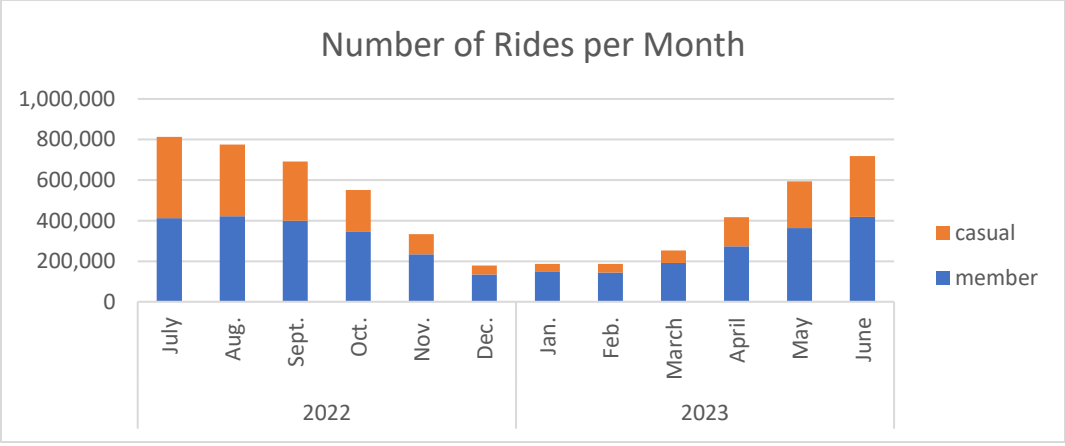
```
rm(all_trips)
rm(all_trips_v2)
rm(day_week_df)
rm(joy_ride_grand_totals_df)
rm(joy_ride_totals_df)
rm(joy_station_name_df)
rm(hour_day_df)
rm(ride_start_station_name_df)
rm(ride_end_station_name_df)
```

- -

Switching back to Excel to create pivot tables and charts

To create the summary table of monthly rides

Loaded the “hour_day_df_results.csv” into Excel added column for year and then made a pivot table with bar chart.



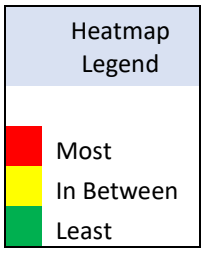
Rides per Month			
	Casual	Member	Total
2022	**1395804**	**1945634**	**3341438**
July	400381	411877	812258
Aug.	353684	421225	774909
Sept.	292561	399359	691920
Oct.	205876	344738	550614
Nov.	99188	233761	332949
Dec.	44114	134674	178788
2023	**816356**	**1538265**	**2354621**
Jan.	39206	147157	186363
Feb.	42177	144370	186547
March	60861	191871	252732
April	144017	272902	416919
May	229677	363717	593394
June	300418	418248	718666
Grand Total	**2212160**	**3483899**	**5696059**

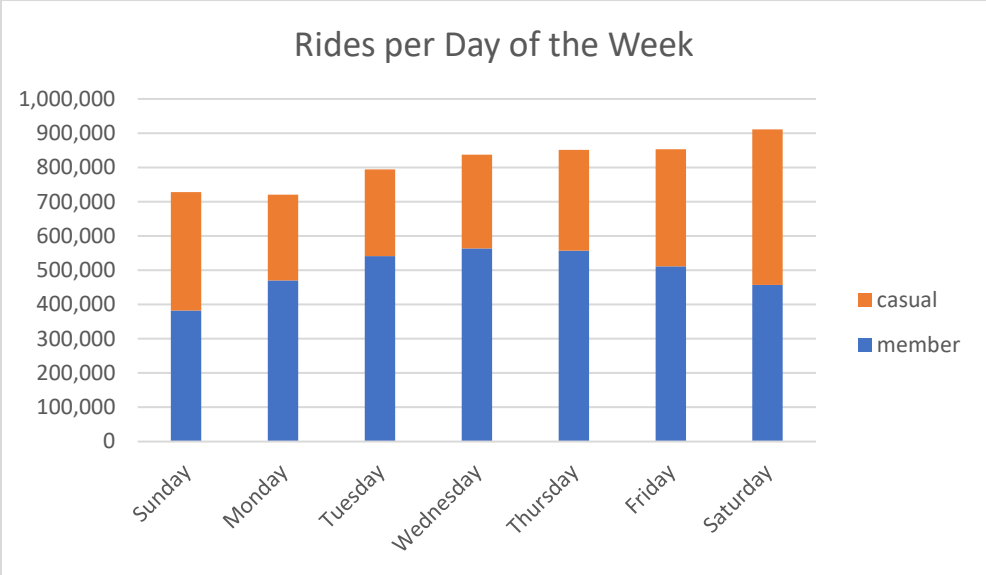
To create the summary table of Daily rides

Created another pivot table and bar chart to show the number of trips taken on the different days of the week.

To create the heatmap effect I used Excel's conditional formatting rule (3-Color-Scale) for the Casual column and a second conditional formatting rule for the member column.

Rides per Day of the Week			
	Casual	**Member**	**Totals**
Sunday	346057	382167	728224
Monday	249541	470690	720231
Tuesday	253302	540953	794255
Wednesday	273294	563678	836972
Thursday	294188	557418	851606
Friday	342291	511378	853669
Saturday	453487	457615	911102
Grand Total	**2212160**	**3483899**	**5696059**





To create the Daily Rides broken down by Hour of the day table

Loaded the "hour_day_df_results.csv" into Excel made a pivot table. Selected only the Casual users and copied this table to a new sheet and rotated it.

To create the heatmap effect I used Excel's conditional formatting rule (3-Color-Scale) all the hours and then a separate conditional formatting rule (3-Color-Scale) for the Total row on the bottom of the table.

Casual Rides per Hour

	Sun.	Mon.	Tues.	Wed.	Thurs.	Fri.	Sat.
0am	13527	3454	2597	2789	3522	5228	11349
1am	8945	2021	1509	1554	1942	3221	8541
2am	5949	1355	891	925	1186	1861	4769
3am	3124	1013	646	665	802	1180	2407
4am	1813	915	720	729	674	841	1163
5am	1663	1693	1862	1958	1798	1864	1322
6am	2499	4127	5352	5254	5363	6334	2476
7am	3960	6863	9675	9873	9808	8698	4568
8am	6647	9045	11912	12475	12700	10073	9121
9am	12263	8340	8386	8813	9122	9798	15369
10am	18622	9804	8716	8629	9563	11833	22872
11am	24069	12475	10719	11361	11898	15763	29385
12pm	27075	15485	13179	13205	14659	19863	33952

1pm	28436	16270	13515	13680	15239	21058	35911
2pm	29820	17104	14356	15004	16720	23131	36872
3pm	31256	18916	17420	17949	20213	27362	37926
4pm	29213	21908	23185	24618	26039	30578	37221
5pm	25160	25337	29033	31821	32963	33216	34581
6pm	21173	22247	24698	27135	29564	30055	31020
7pm	16001	17012	18050	20676	21775	22568	23948
8pm	11484	12395	12744	14970	15428	16346	18344
9pm	9671	9654	11144	13247	13321	13803	16955
10pm	8208	7322	8198	9947	12243	14345	17235
11pm	5479	4786	4795	6017	7646	13272	16180
Total	346057	249541	253302	273294	294188	342291	453487

Heatmap Legend Most In Between Least

To create the busiest station table

Loaded the “ride_start_station_name_df.csv” and “ride_end_station_name_df.csv” into Excel and created a sheet with the data combined to made a pivot table. That allowed me to selected both start and end along with both casual and member users or any combination of the 4 variables. To create the top ten table of busiest stations I sorted by number of rides to cut and paste into another table for formatting with pretty colors. To identify which locations were near the Lake Michigan I manually did a google maps search for each of the ten stations and noticed many were along the waterfront.

Top Ten Locations	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Grand Total
Grand Total	**1330675**	**1326507**	**1464228**	**1538439**	**1560678**	**1555840**	**1658209**	**10434576**
Streeter Dr & Grand Ave*	24714	16428	14418	14697	15969	20365	33686	140277
DuSable Lake Shore Dr & Monroe St*	15258	8869	7382	7572	8427	11238	20699	79445
DuSable Lake Shore Dr & North Blvd*	13294	9533	8833	9915	9982	9616	18124	79297
Michigan Ave & Oak St*	13593	8707	8560	8935	9671	11354	17472	78292
Wells St & Concord Ln*	10263	8804	9100	10063	10543	11264	14926	74963
Clark St & Elm St	9423	9126	10075	10420	10286	10288	10711	70329
Kingsbury St & Kinzie St	6504	10472	11455	11206	10716	10021	7773	68147
Millennium Park*	11164	8521	7130	7037	9040	10246	14221	67359
Theater on the Lake*	11944	7381	6763	6827	7208	8187	15383	63693
Halsted St & Fulton St	7031	7001	8521	9631	10343	9859	9255	61641

To answer the question “Where is everyone going?” I repeated the process for only the ending stations and for each of the Casual and Member users, to create the tables.

Top Ten Casual Ending Locations	
Streeter Dr & Grand Ave*	54925
DuSable Lake Shore Dr & Monroe St*	28132
Michigan Ave & Oak St*	25122
Millennium Park*	24959
DuSable Lake Shore Dr & North Blvd*	23994
Theater on the Lake*	18319
Shedd Aquarium*	17262
Wells St & Concord Ln*	14592
Dusable Harbor*	13116
Clark St & Armitage Ave*	12683

To create the Joy ride breakdown table

Loaded the “joy_ride_totals_df.csv” into Excel and created the table.

Breakdown of Joy Rides				
		Number of Joy Ride	Percentage of Total Rides	Average Ride Length in Minutes
casual		163,856	3%	34.7
member		110,859	2%	15.3
Totals	5,696,059	274,715	5%	26.9

To create the Joy ride top ten table

Loaded the “joy_ride_station_name_df.csv” into Excel and sorted the data by number of rides and created the table.

Top Ten Locations for Casual Joy Rides		
	Number of Ride	Average Ride Length in Minutes
Streeter Dr & Grand Ave*	9324	41.2
DuSable Lake Shore Dr & Monroe St*	6360	34.6
Michigan Ave & Oak St*	4148	46.1
Millennium Park*	3725	39.2
Montrose Harbor*	2640	49.6
Dusable Harbor*	2369	35.3
Shedd Aquarium*	2284	22.2
DuSable Lake Shore Dr & North Blvd*	2146	38.8
Theater on the Lake*	1949	42.7
Adler Planetarium*	1803	36.0

This completes the “Cyclistic Bike Sharing - Technical Document.docx”, for the complete analysis, please refer to separate document “Cyclistic Bike Sharing – Final Report.docx”