Bike Share Capstone Project

Mohamed Elsayed

March 1st, 2022

I. Introduction

Cyclistic is a bike-share program that features more than 5,800 bicycles and 600 docking stations. Cyclistic sets itself apart by also offering reclining bikes, hand tricycles, and cargo bikes, making bike-share more inclusive to people with disabilities and riders who can't use a standard two-wheeled bike. The majority of riders opt for traditional bikes; about 8% of riders use the assistive options. Cyclistic users are more likely to ride for leisure, but about 30% use them to commute to work each day

The director of marketing believes the company's future success depends on maximizing the number of annual memberships. Therefore, The objective of this project is to identify the main differences between member & casual user patterns when using Cyclistic.

Identifying & highlighting the most prominent differences between those two user groups will be done by exploring data from the year 2021. The data has been made available for download by Motivate International Inc. here **https://divvy-tripdata.s3.amazonaws.com/index.html** under this license **https://ride.divvybikes.com/data-license-agreement**

II. Outline

This project will be focusing on two main tools to achieve the above objective, SQL (BigQuery) for data manipulation & Tableau for data visualization. The steps below outline the process followed:

1. Download the monthly ".csv" data files for the year 2021.
2. Upload files into BigQuery to be combined into one table for all months from that year.
3. Start data exploration by filtering, sorting & manipulating data as needed to get some data insights.
4. Export the full dataset to Tableau for additional insights using visualization.
5. Propose recommendations based on the results of the above steps.

III. Data Preparation & Manipulation

In order to upload ".csv" files into BigQuery, each individual file needs to be less than 100 Mb in size. After examining the monthly ".csv" file sizes, it was found that months June through October all exceed that size limit. So in order to get around that issue, "R" will be used to split each one of those months into 2 files, that way files can be successfully uploaded to BigQuery without any issues.

Below is the code that was used to import, split & export those ".csv" files using "R"

Please Note: The below chunk shows only the code for splitting the month of June, other months were not included in this report for presentation purposes & to save space. This is an example to show the method used, ALL other months were split using the exact same method. Details are in the Rmd file.

```
#- Splitting June
Jun_2021<- read_csv("./Google Data Analytics/
         8. Google Data Analytics Capstone/Case Study - 1/202106-divvy-tripdata.csv")

Jun_2021_1<- Jun_2021[1:375000,]
Jun_2021_2<- Jun_2021[375001:729595,]
write_csv(Jun_2021_1, "Jun_2021_1.csv")
write_csv(Jun_2021_2, "Jun_2021_2.csv")
write_csv(Jun_2021_1, "202106-divvy-tripdata_1.csv")
write_csv(Jun_2021_2, "202106-divvy-tripdata_2.csv")
```

After ALL files are of the allowable sizes for upload, they are uploaded into BigQuery using Auto-Detect schema to be combined into one dataset representing the riders data for the full year of 2021. After importing a few files to BigQuery it was realized that the “end_lat” & “end_lng” variables are being imported as strings not as integers or “FLOAT64” which gave an error when executing the query to combine all months in one table. The “`CAST`” function was added to the query to convert those variables into integers to overcome this issue.

Below is the SQL code used to manipulate & combine the data as needed to create the full dataset for one year.

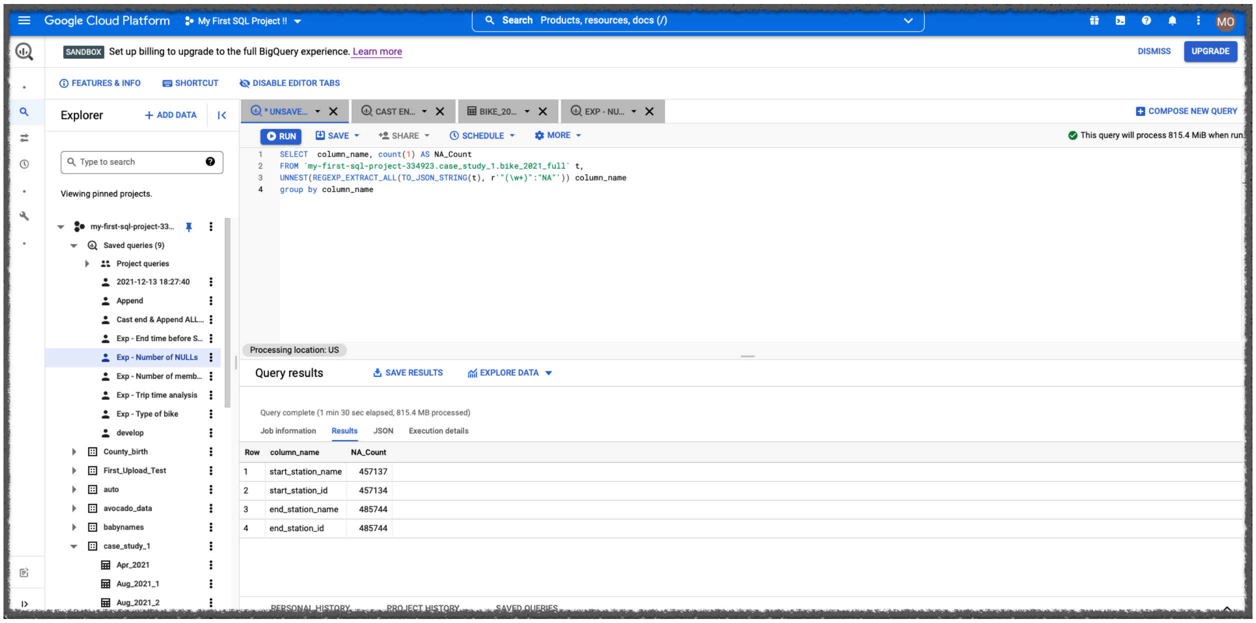
```
# Change Latitude data type to match ALL imported tables
# Then Append ALL months to create a full year database

SELECT *
FROM ‘my-first-sql-project-334923.case_study_1.Jan_2021‘
union all
select *
FROM ‘my-first-sql-project-334923.case_study_1.Feb_2021‘
union all
select *
FROM ‘my-first-sql-project-334923.case_study_1.Mar_2021‘
union all
select *
FROM ‘my-first-sql-project-334923.case_study_1.Apr_2021‘
union all
select *
FROM ‘my-first-sql-project-334923.case_study_1.May_2021‘
union all
SELECT * replace(
safe_cast(end_lat as FLOAT64) as end_lat,
safe_cast(end_lng as FLOAT64) as end_lng)
FROM ‘my-first-sql-project-334923.case_study_1.Jun_2021_1‘
union all
select * replace(
safe_cast(end_lat as FLOAT64) as end_lat,
safe_cast(end_lng as FLOAT64) as end_lng)
from ‘my-first-sql-project-334923.case_study_1.Jun_2021_2‘
union all
SELECT * replace(
safe_cast(end_lat as FLOAT64) as end_lat,
safe_cast(end_lng as FLOAT64) as end_lng)
FROM ‘my-first-sql-project-334923.case_study_1.Jul_2021_1‘
union all
select *
from ‘my-first-sql-project-334923.case_study_1.Jul_2021_2‘
```

```
union all
SELECT * replace(
safe_cast(end_lat as FLOAT64) as end_lat,
safe_cast(end_lng as FLOAT64) as end_lng)
FROM ‘my-first-sql-project-334923.case_study_1.Aug_2021_1‘
union all
select *
from ‘my-first-sql-project-334923.case_study_1.Aug_2021_2‘
union all
SELECT * replace(
safe_cast(end_lat as FLOAT64) as end_lat,
safe_cast(end_lng as FLOAT64) as end_lng)
FROM ‘my-first-sql-project-334923.case_study_1.Sep_2021_1‘
union all
select *
from ‘my-first-sql-project-334923.case_study_1.Sep_2021_2‘
union all
SELECT * replace(
safe_cast(end_lat as FLOAT64) as end_lat,
safe_cast(end_lng as FLOAT64) as end_lng)
FROM ‘my-first-sql-project-334923.case_study_1.Oct_2021_1‘
union all
select *
from ‘my-first-sql-project-334923.case_study_1.Oct_2021_2‘
union all
select *
FROM ‘my-first-sql-project-334923.case_study_1.Nov_2021‘
union all
select *
FROM ‘my-first-sql-project-334923.case_study_1.Dec_2021‘
```

IV. Data Exploration using SQL

After combining the 12 months of the year 2021, exploring the full dataset using SQL shows that the table has **5,593,052** observations & 13 variables.

Below is the SQL code and the results of checking the dataset for missing values. The result of the query shows that the database has almost 10% of it's values missing for the start station name & end station name variables. This is significant but there's no way those missing values can be filled accurately. Additionally, those missing values won't affect calculations for achieving the main objective of this project which explores the differences between members & casual users.



```sql
SELECT  column_name, count(1) AS NA_Count
FROM `my-first-sql-project-334923.case_study_1.bike_2021_full` t,
UNNEST(REGEXP_EXTRACT_ALL(TO_JSON_STRING(t), r'"(\w+)":"NA"')) column_name
group by column_name
```

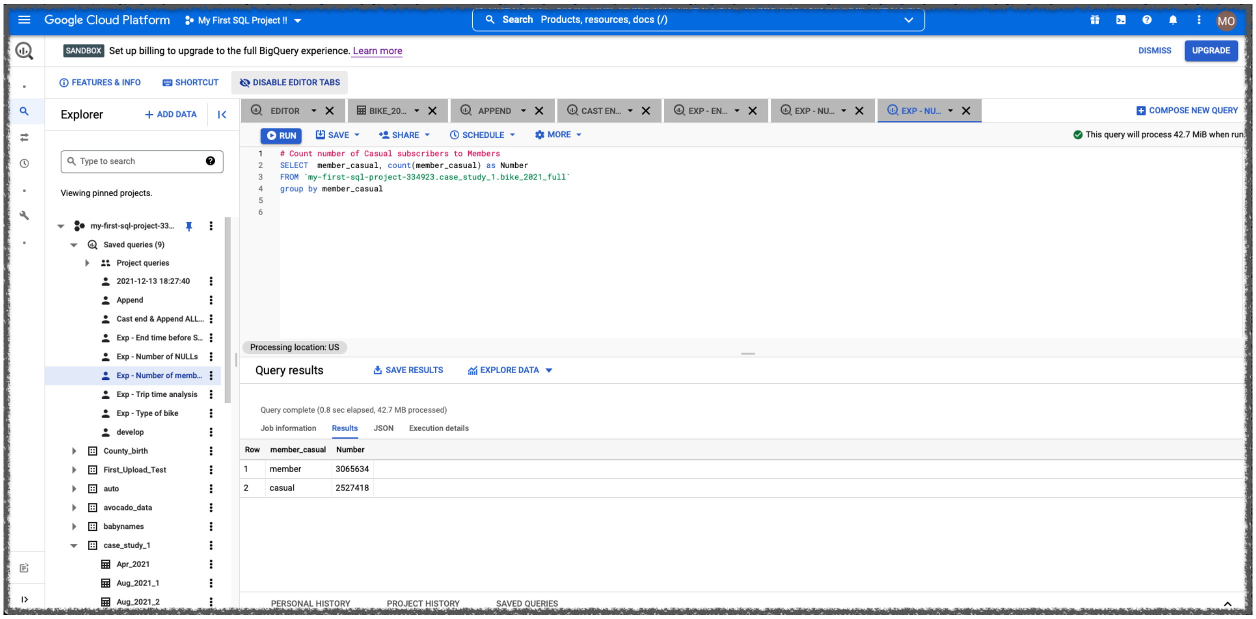
Processing location: US

Query results

Query complete (1 min 30 sec elapsed, 815.4 MB processed)

Row	column_name	NA_Count
1	start_station_name	457137
2	start_station_id	457134
3	end_station_name	485744
4	end_station_id	485744

The next step is exploring the number of members vs casual users. The query results below show that there are around 3 million members, while there are around 2.5 million casual users.



```sql
# Count number of Casual subscribers to Members
SELECT  member_casual, count(member_casual) as Number
FROM `my-first-sql-project-334923.case_study_1.bike_2021_full`
group by member_casual
```

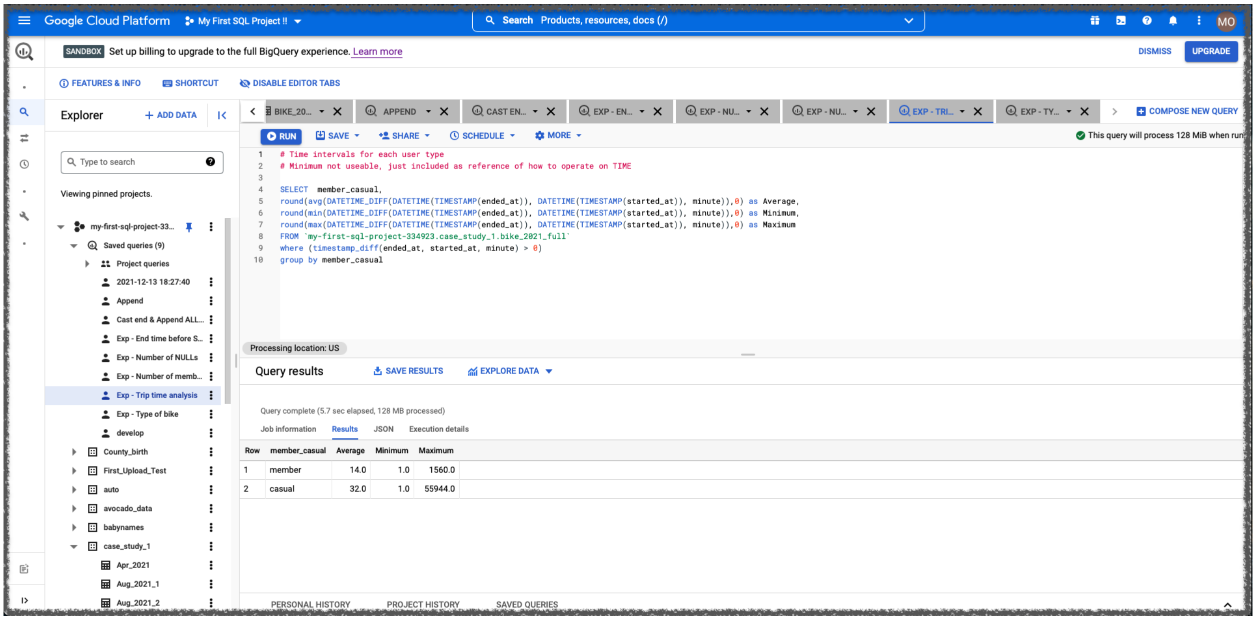
Processing location: US

Query results

Query complete (0.8 sec elapsed, 42.7 MB processed)

Row	member_casual	Number
1	member	3065634
2	casual	2527418

When exploring the average & the maximum trip times for each user group in minutes, the query results below are very interesting.. it seems that casual users are using the bikes for longer times than members. With the average trip time for members being 14 minutes while the average trip time for casual users is 32 minutes, That's more than double.

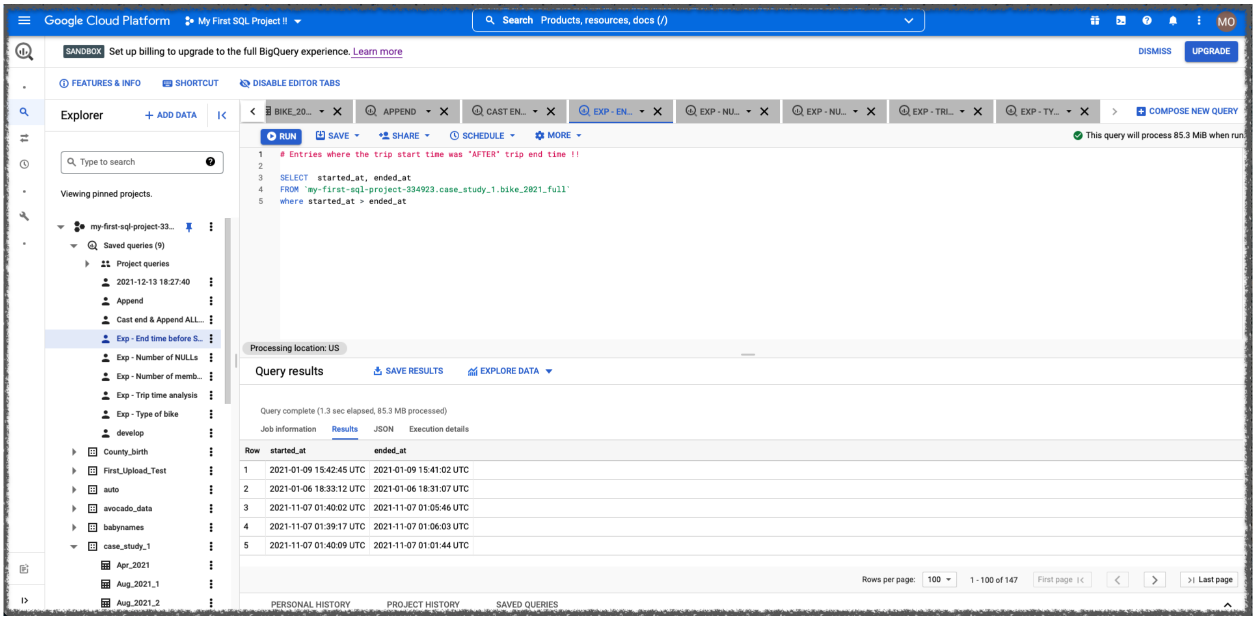


```
# Time intervals for each user type
# Minimum not useable, just included as reference of how to operate on TIME

SELECT  member_casual,
round(avg(DATETIME_DIFF(DATETIME(TIMESTAMP(ended_at)), DATETIME(TIMESTAMP(started_at)), minute)),0) as Average,
round(min(DATETIME_DIFF(DATETIME(TIMESTAMP(ended_at)), DATETIME(TIMESTAMP(started_at)), minute)),0) as Minimum,
round(max(DATETIME_DIFF(DATETIME(TIMESTAMP(ended_at)), DATETIME(TIMESTAMP(started_at)), minute)),0) as Maximum
FROM `my-first-sql-project-334923.case_study_1.bike_2021_full`
where (timestamp_diff(ended_at, started_at, minute) > 0)
group by member_casual
```

Query complete (5.7 sec elapsed, 128 MB processed)

Row	member_casual	Average	Minimum	Maximum
1	member	14.0	1.0	1560.0
2	casual	32.0	1.0	55944.0

The results above show that there's an issue with the minimum trip times in the dataset. In order to try to understand the reason for that, the query below checks the entries in which the trip end time was "BEFORE" the trip start time. Obviously those entries were due to an error but the results of the query show that there are 147 entries in the whole dataset that meet this condition. This is why the minimum ride times are not accurate.



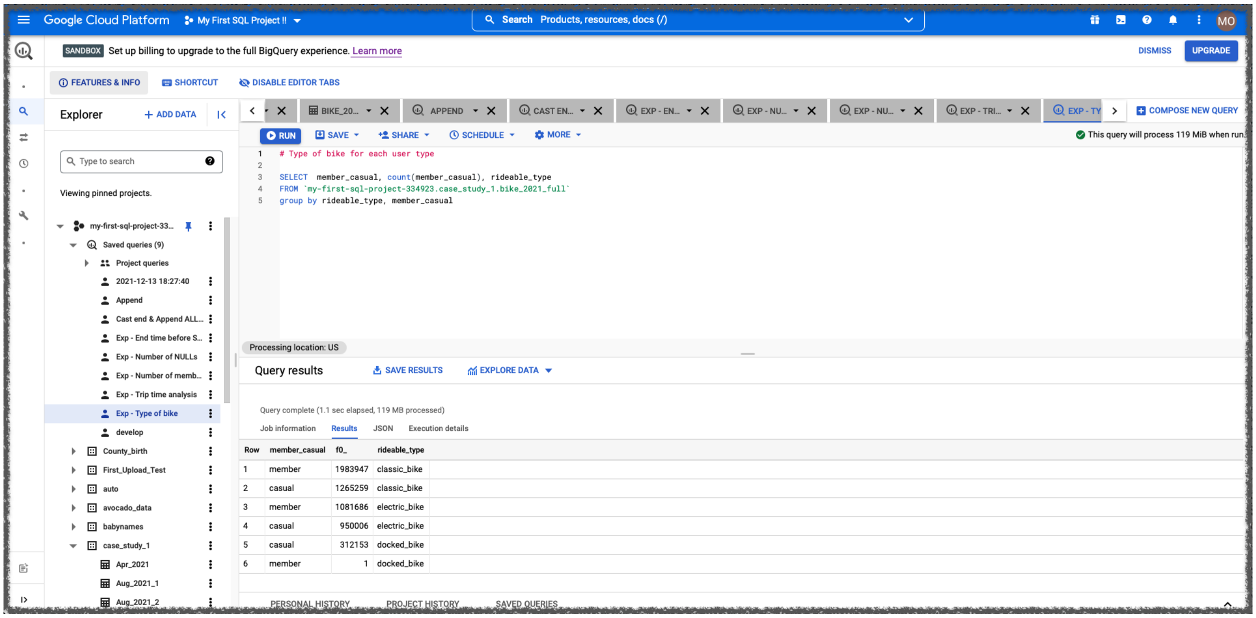
```
# Entries where the trip start time was "AFTER" trip end time !!

SELECT  started_at, ended_at
FROM `my-first-sql-project-334923.case_study_1.bike_2021_full`
where started_at > ended_at
```

Query complete (1.3 sec elapsed, 85.3 MB processed)

Row	started_at	ended_at
1	2021-01-09 15:42:45 UTC	2021-01-09 15:41:02 UTC
2	2021-01-06 18:33:12 UTC	2021-01-06 18:31:07 UTC
3	2021-11-07 01:40:02 UTC	2021-11-07 01:05:46 UTC
4	2021-11-07 01:39:17 UTC	2021-11-07 01:06:03 UTC
5	2021-11-07 01:40:09 UTC	2021-11-07 01:01:44 UTC

Rows per page: 100 1 - 100 of 147

Finally, exploring the types of bikes used by different user groups show that in general classic bikes are the most popular choice followed by electric bikes. Also, docked bikes are almost exclusively used by casual members. Query & results are below



```
# Type of bike for each user type

SELECT  member_casual, count(member_casual), rideable_type
FROM `my-first-sql-project-334923.case_study_1.bike_2021_full`
group by rideable_type, member_casual
```

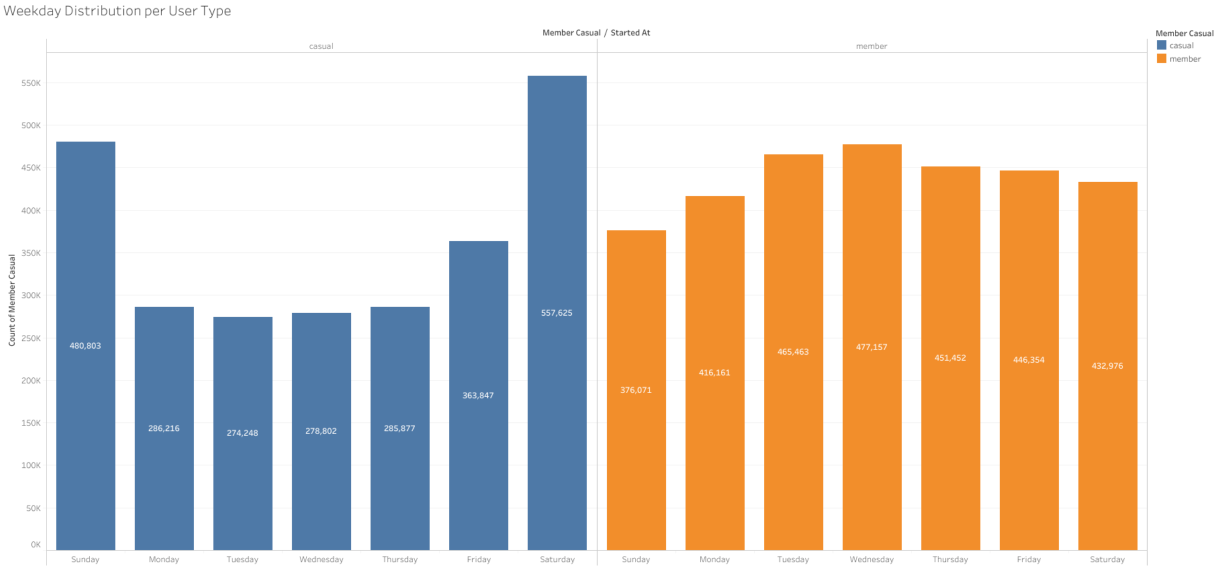
Query results

Query complete (1.1 sec elapsed, 119 MB processed)

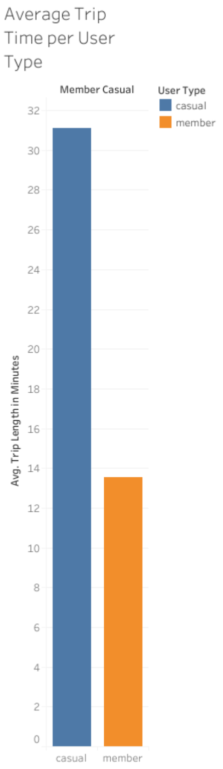
Row	member_casual	f0_	rideable_type
1	member	1983947	classic_bike
2	casual	1265259	classic_bike
3	member	1081686	electric_bike
4	casual	950006	electric_bike
5	casual	312153	docked_bike
6	member	1	docked_bike

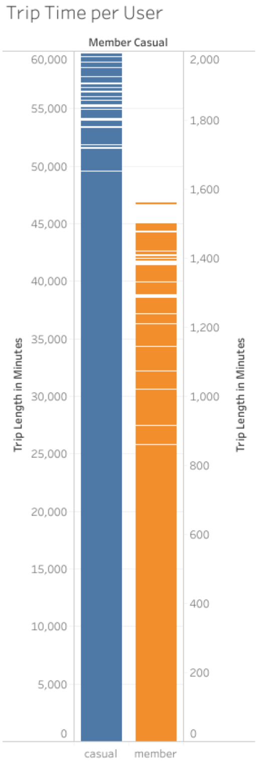
V. Data Exploration using Tableau

For visualization of the results of the queries above & for additional insights, data was exported to Tableau. The first plot below will illustrate the weekday distribution for each user group. The results show that for casual users, weekends are the busiest days of the week while for members Tuesday & Wednesday are the busiest.



The next plot is a combination of 2 graphs. The one on the left is a bar graph that shows the average trip time for casual users vs. members. And the one on the right is a Gantt view the distribution of ride time lengths per user type. This shows that in fact casual users trip times are significantly longer than members.



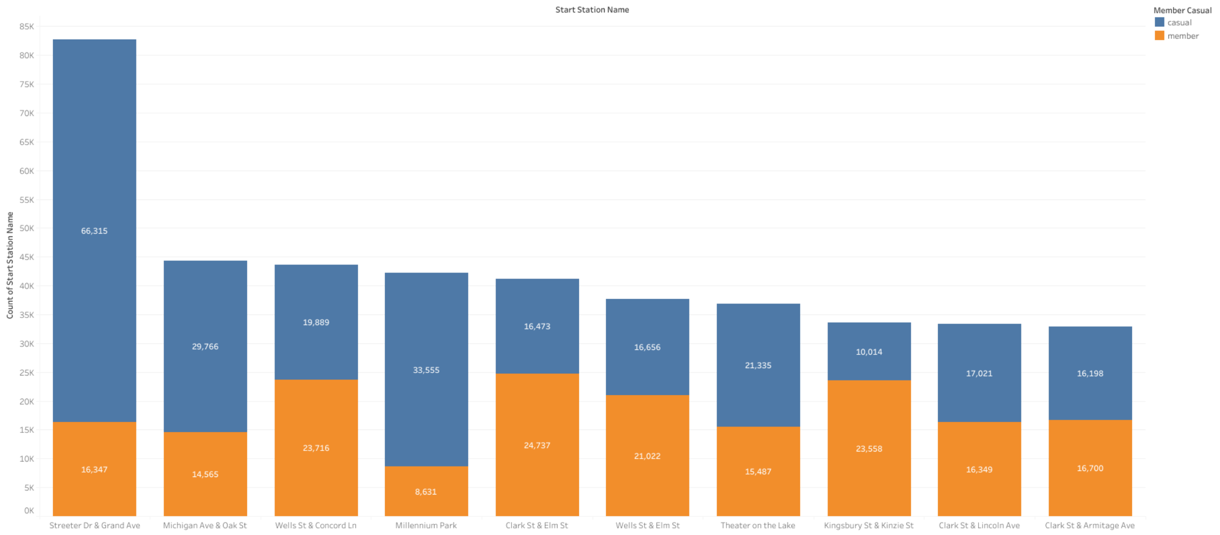


Now turning to the user distribution for the most popular stations from which trips start, insights are illustrated using two methods.

1. A stacked bar graph that shows the number of each user type per station for the busiest 10 stations

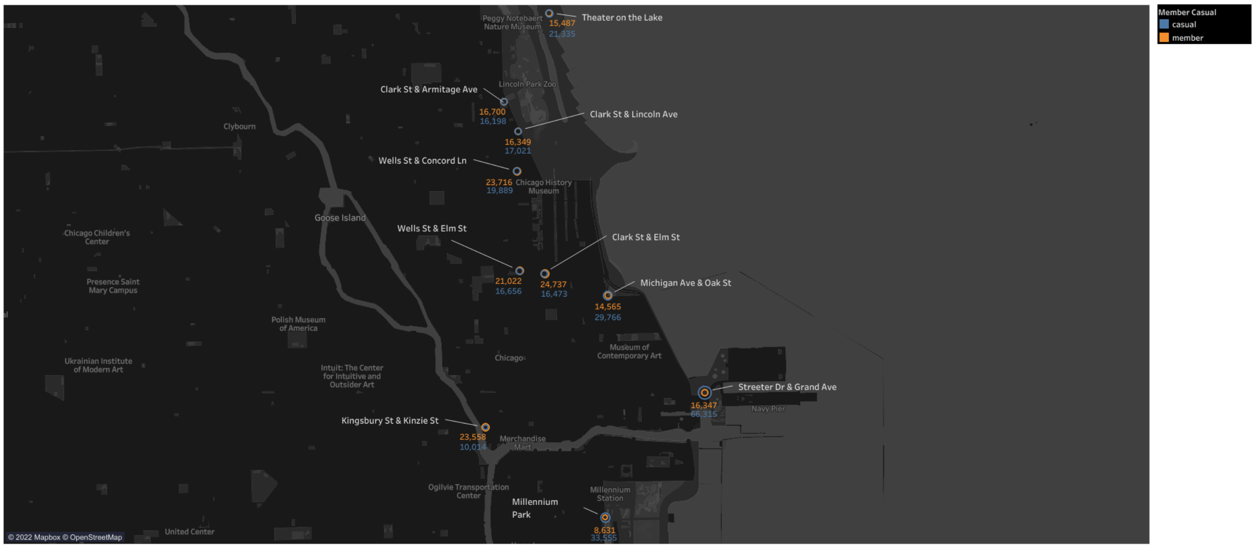
2. A map that shows the locations of those stations with the corresponding user types in Chicago & the proximity to points of interest

Top Ten Busiest Stations per User Type



Note: The ring size on each location represents the number of trips started from that location. Each user type is represented by a different color illustrated by the legend. As illustrated on the map below, there are two locations in which casual users are significantly more than members.

User Type per Station Name (Top 10)

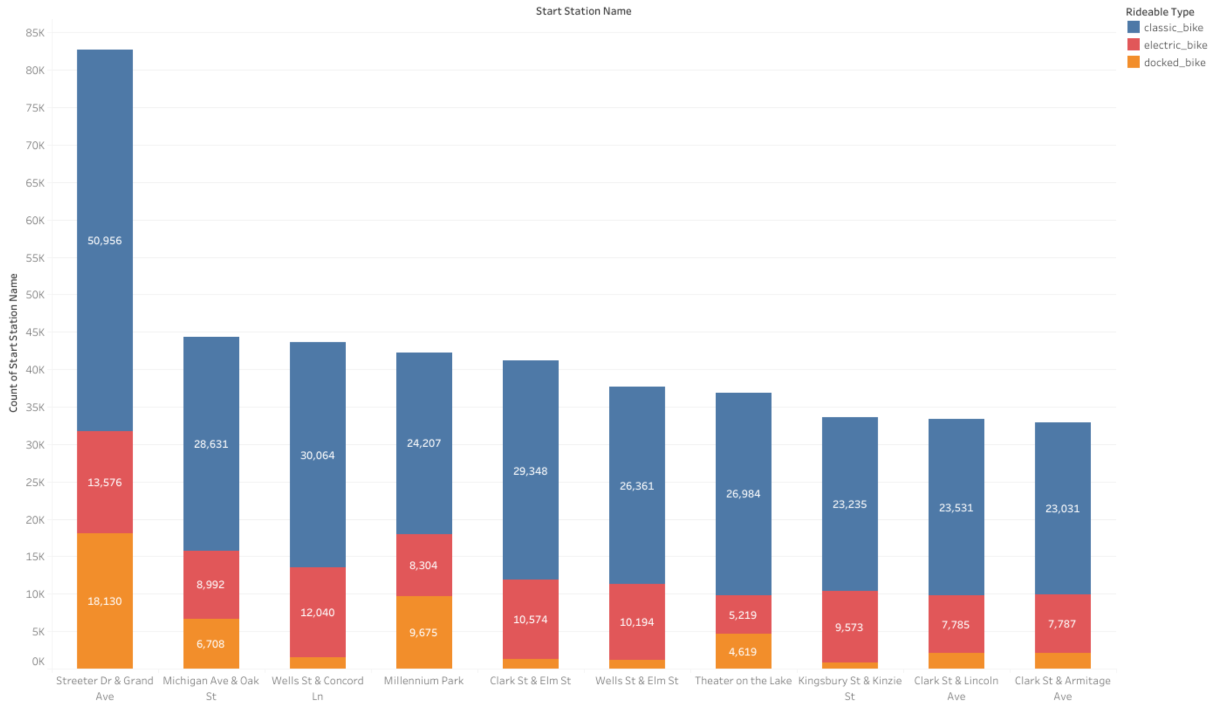


Following the same methods of visualization, the most popular bike types for the top 10 busiest stations are explored using the two methods below.

1. A stacked bar graph that shows the number of each bike type rented per station for the busiest 10 stations

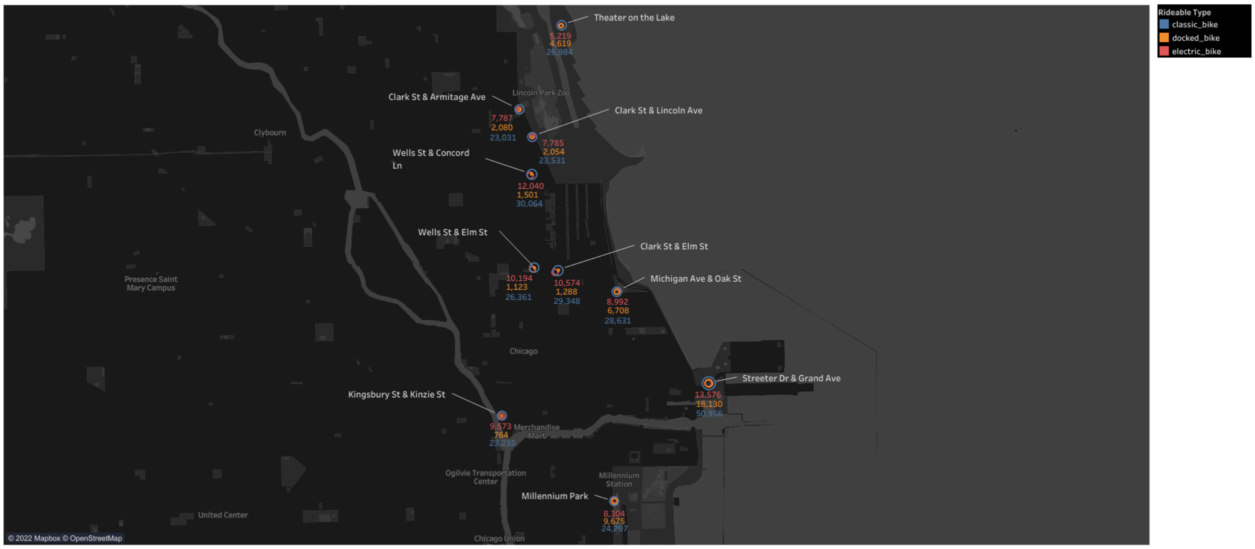
2. A map that shows the locations of those stations with the corresponding bike type usage in Chicago & the proximity to points of interest

Bike Type Distribution for the Top 10 Stations

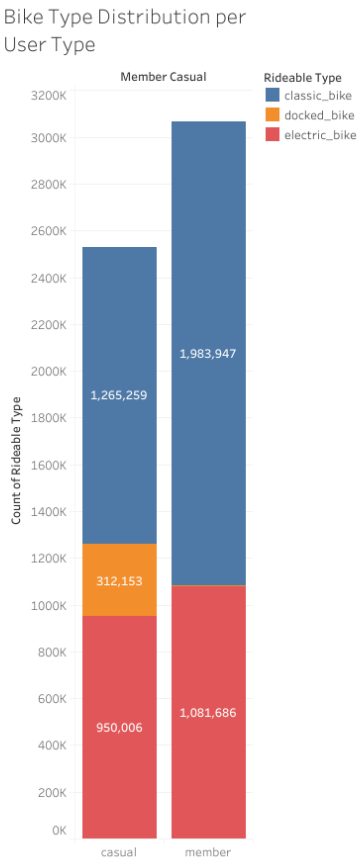


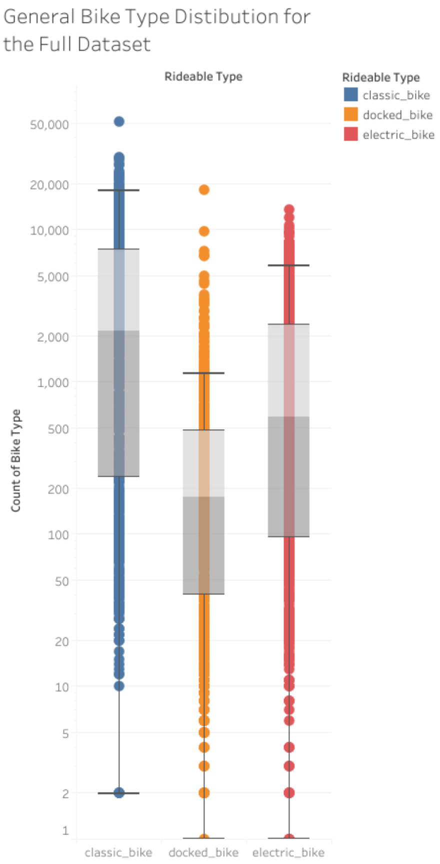
Note: The ring size on each location represents the number of trips started from that location. Each bike type is represented by a different color illustrated by the legend.

Bike Type per Station Name (Top 10)



The next plot is a combination of 2 graphs. The one on the left is a stacked bar graph that illustrates the distribution of different bike types used per each user group. The one on the right is a box plot that shows the distribution of different bike types in the whole dataset. A logarithmic scale is used to better represent the averages and the lower values. The results show that the classic bikes are the most popular, followed by electric bikes. The results match the outcome of the SQL queries shown in the data exploration using SQL section.





VI. Insights summary & recommendations

Several important insights were discovered through the data exploration performed above. Those insights were key in determining the right recommendations to achieve the objective of this project. The most important insights were as follows:

1. The number of casual users almost matches the number of members. This indicates that there's a big target audience for the marketing campaigns & there's a possibility to potentially double the number of current memberships.
2. Casual members trip times are almost double the trip times for members.
3. Casual members usage of the service almost doubles on weekends.
4. Docked bikes are almost exclusively used by casual users.
5. Casual users outnumber members in three out of the busiest ten stations.

After studying those insights, my recommendations for increasing the number of memberships by targeting casual users with campaigns to switch into members are as follows:

1. Encourage casual users to apply for memberships by creating special offers for rides on weekends & for ride times longer than 15 minutes.
2. Focus the marketing campaigns on the following three stations in which casual users are significantly more than members
   A. Streeter Dr & Grand Ave
   B. Michigan Ave & Oak St
   C. Millennium Park
3. Encourage casual members to apply for memberships by creating special offers on docked bikes.