John Kennedy Hancock
October 2021 Google Data Analytics Capstone

Case Study

BUSINESS TASK

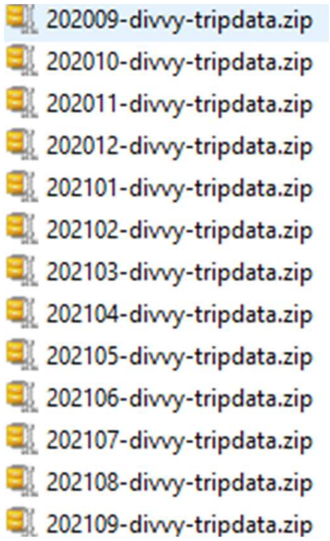
Cyclistic built its successful bike sharing program by offering two methods of customer engagements, casual riders and members. Casual riders are defined as those customers who purchase rides on short term basis, one trip or a day pass. Full members purchase annual memberships. Our financial analyst team believe that we can be more profitable if we increase annual memberships.

In order to increase our memberships, we are tasked with the need to understand our members. We need to understand the differences between annual and casual members. We need to understand how each customer group engages with our service. We need to get an insight into how often each group uses our bikes.

PREPARE

Cyclistic provided 36 trip datasets. (See the datasets at this link.) This data publicly provided by Motivate International Inc. under the licensing protocols available at this link. Motivate maintains and culls the data of personally identifiable information such as gender and credit card information.

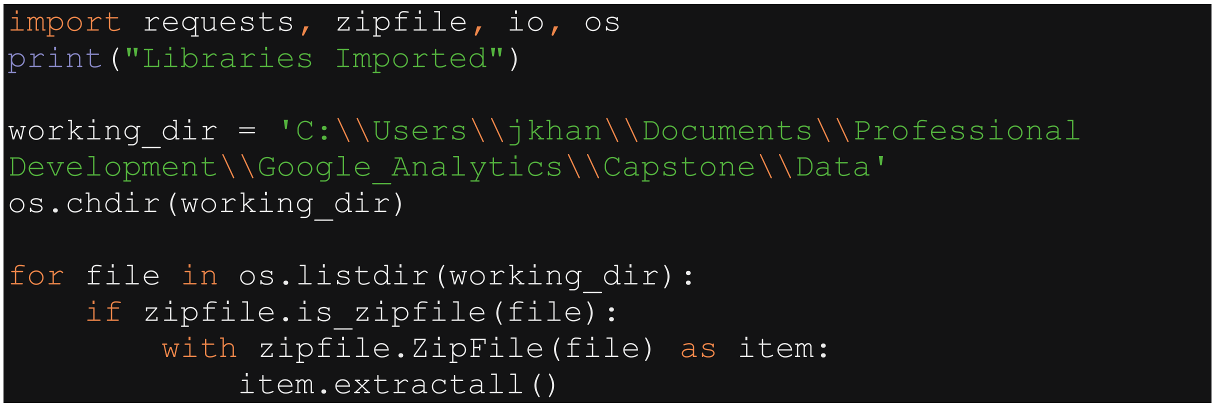
For this study, I downloaded 12 months of data from September 2020 through September 2021.



202009-divvy-tripdata.zip
202010-divvy-tripdata.zip
202011-divvy-tripdata.zip
202012-divvy-tripdata.zip
202101-divvy-tripdata.zip
202102-divvy-tripdata.zip
202103-divvy-tripdata.zip
202104-divvy-tripdata.zip
202105-divvy-tripdata.zip
202106-divvy-tripdata.zip
202107-divvy-tripdata.zip
202108-divvy-tripdata.zip
202109-divvy-tripdata.zip

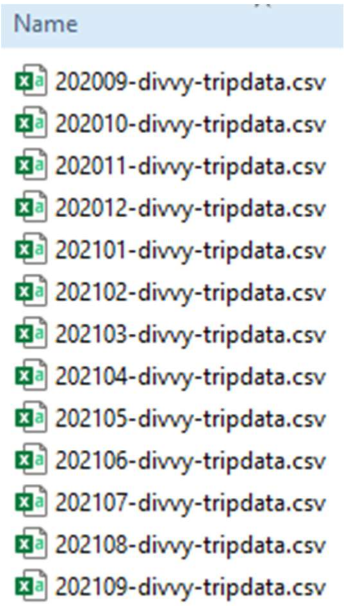
Using this Python code snippet, I unzipped all of the files.

Unzip all files:



```
import requests, zipfile, io, os
print("Libraries Imported")

working_dir = 'C:\\Users\\jkhan\\Documents\\Professional
Development\\Google_Analytics\\Capstone\\Data'
os.chdir(working_dir)

for file in os.listdir(working_dir):
    if zipfile.is_zipfile(file):
        with zipfile.ZipFile(file) as item:
            item.extractall()
```

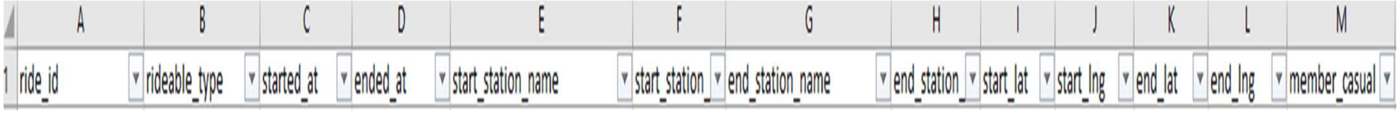
After unzipping, we see that there are 13 csv files for each month.



Name

202009-divvy-tripdata.csv
202010-divvy-tripdata.csv
202011-divvy-tripdata.csv
202012-divvy-tripdata.csv
202101-divvy-tripdata.csv
202102-divvy-tripdata.csv
202103-divvy-tripdata.csv
202104-divvy-tripdata.csv
202105-divvy-tripdata.csv
202106-divvy-tripdata.csv
202107-divvy-tripdata.csv
202108-divvy-tripdata.csv
202109-divvy-tripdata.csv

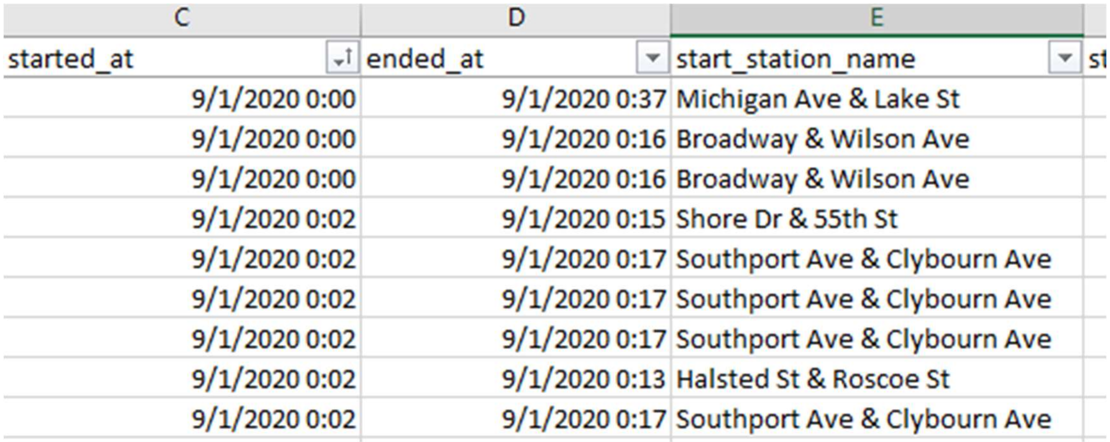
Looking at the csv files in excel, we see that there 13 fields for each data observation.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	ride_id	rideable_type	started_at	ended_at	start_station_name	start_station_	end_station_name	end_station_	start_lat	start_lng	end_lat	end_lng	member_casual

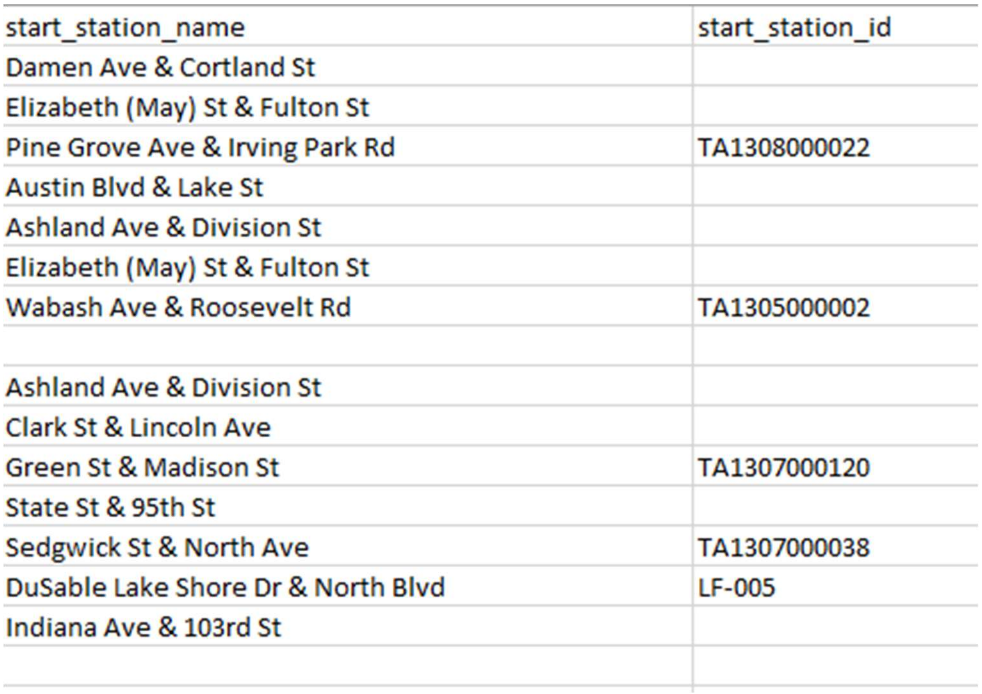
1. ride_id – A unique, alphanumeric identifier for each ride
2. rideable_type – A categorical type docked_bike or electric_bike
3. started_at - The date/time without timestamp where the ride started.
4. ended_at - - The date/time without timestamp where the ride ended.
5. start_station_name - The name of the station where the ride started.
6. start_station_id – Alphanumeric identifier for the starting station
7. end_station_name - - The name of the station where the ride ended.
8. end_station_id - Alphanumeric identifier for the ending station
9. start_lat – Latitude for the starting station
10. start_lng – Longitude for the starting station
11. end_lat – Latitude for the ending station
12. end_lng– Longitude for the ending station
13. member_casual – Categorical variable for member type.

Sorting the data by “started_at” shows the earliest date of the data:



C	D	E	
started_at	ended_at	start_station_name	st
9/1/2020 0:00	9/1/2020 0:37	Michigan Ave & Lake St	
9/1/2020 0:00	9/1/2020 0:16	Broadway & Wilson Ave	
9/1/2020 0:00	9/1/2020 0:16	Broadway & Wilson Ave	
9/1/2020 0:02	9/1/2020 0:15	Shore Dr & 55th St	
9/1/2020 0:02	9/1/2020 0:17	Southport Ave & Clybourn Ave	
9/1/2020 0:02	9/1/2020 0:17	Southport Ave & Clybourn Ave	
9/1/2020 0:02	9/1/2020 0:17	Southport Ave & Clybourn Ave	
9/1/2020 0:02	9/1/2020 0:13	Halsted St & Roscoe St	
9/1/2020 0:02	9/1/2020 0:17	Southport Ave & Clybourn Ave	

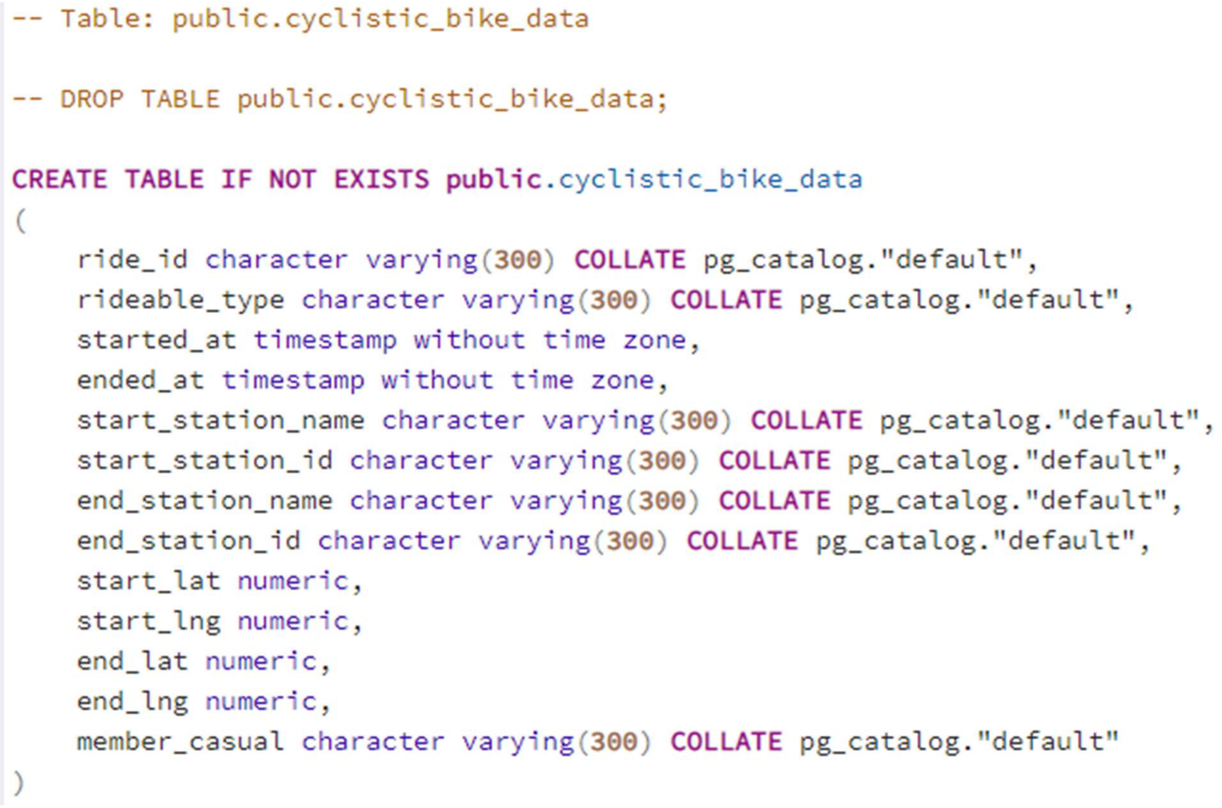
Also, there is missing data as well



start_station_name	start_station_id
Damen Ave & Cortland St	
Elizabeth (May) St & Fulton St	
Pine Grove Ave & Irving Park Rd	TA1308000022
Austin Blvd & Lake St	
Ashland Ave & Division St	
Elizabeth (May) St & Fulton St	
Wabash Ave & Roosevelt Rd	TA1305000002
Ashland Ave & Division St	
Clark St & Lincoln Ave	
Green St & Madison St	TA1307000120
State St & 95th St	
Sedgwick St & North Ave	TA1307000038
DuSable Lake Shore Dr & North Blvd	LF-005
Indiana Ave & 103rd St	

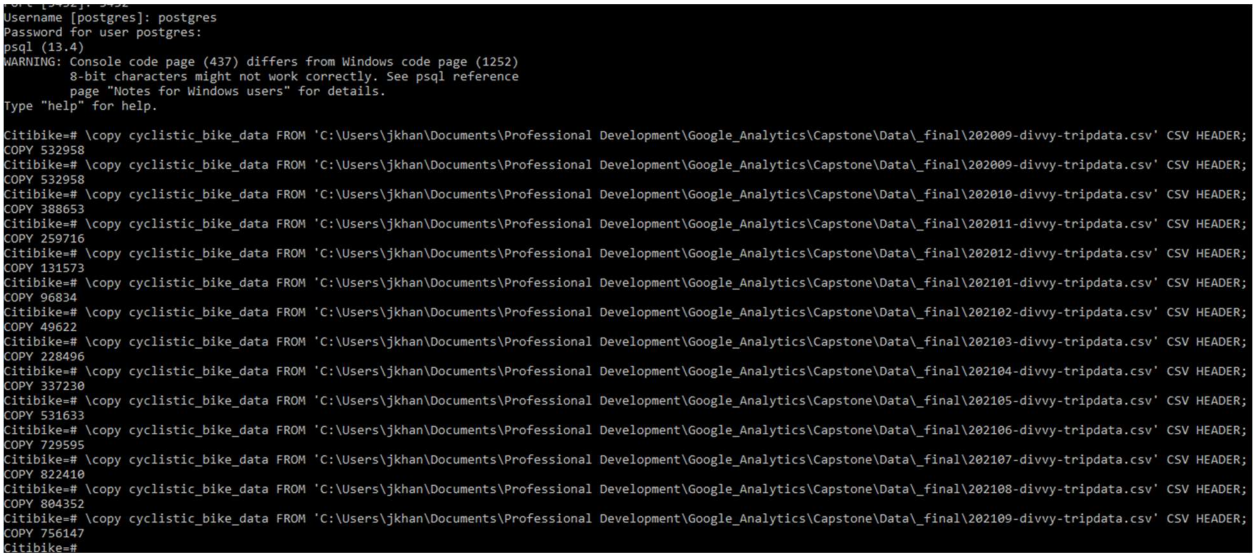
PROCESS

For processing, I created a PostgreSQL database and a created a table for the data:



```sql
-- Table: public.cyclistic_bike_data

-- DROP TABLE public.cyclistic_bike_data;

CREATE TABLE IF NOT EXISTS public.cyclistic_bike_data
(
    ride_id character varying(300) COLLATE pg_catalog."default",
    rideable_type character varying(300) COLLATE pg_catalog."default",
    started_at timestamp without time zone,
    ended_at timestamp without time zone,
    start_station_name character varying(300) COLLATE pg_catalog."default",
    start_station_id character varying(300) COLLATE pg_catalog."default",
    end_station_name character varying(300) COLLATE pg_catalog."default",
    end_station_id character varying(300) COLLATE pg_catalog."default",
    start_lat numeric,
    start_lng numeric,
    end_lat numeric,
    end_lng numeric,
    member_casual character varying(300) COLLATE pg_catalog."default"
)
```

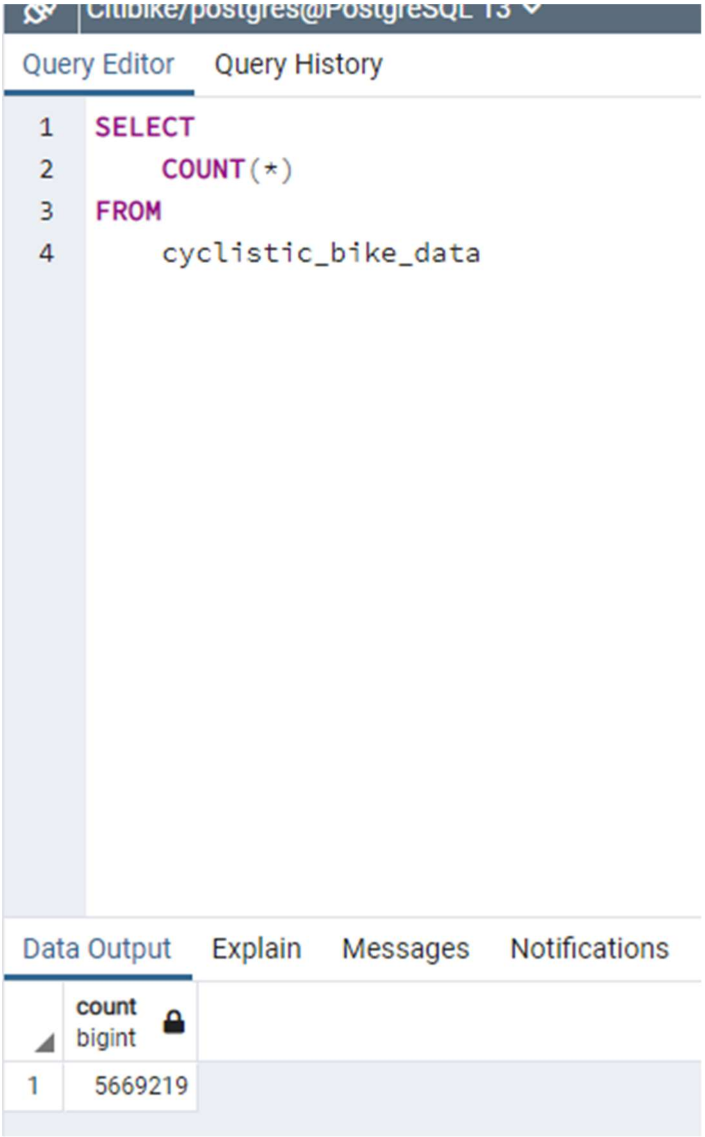
Next, I created a script to load the 13 csv files into the database:



```
Port [5432]: 5432
Username [postgres]: postgres
Password for user postgres:
psql (13.4)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202009-divvy-tripdata.csv' CSV HEADER;
COPY 532958
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202009-divvy-tripdata.csv' CSV HEADER;
COPY 532958
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202010-divvy-tripdata.csv' CSV HEADER;
COPY 388653
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202011-divvy-tripdata.csv' CSV HEADER;
COPY 259716
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202012-divvy-tripdata.csv' CSV HEADER;
COPY 131573
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202101-divvy-tripdata.csv' CSV HEADER;
COPY 96834
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202102-divvy-tripdata.csv' CSV HEADER;
COPY 49622
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202103-divvy-tripdata.csv' CSV HEADER;
COPY 228496
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202104-divvy-tripdata.csv' CSV HEADER;
COPY 337230
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202105-divvy-tripdata.csv' CSV HEADER;
COPY 531633
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202106-divvy-tripdata.csv' CSV HEADER;
COPY 729595
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202107-divvy-tripdata.csv' CSV HEADER;
COPY 822410
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202108-divvy-tripdata.csv' CSV HEADER;
COPY 804352
Citibike=# \copy cyclistic_bike_data FROM 'C:\Users\jkhan\Documents\Professional Development\Google_Analytics\Capstone\Data\_final\202109-divvy-tripdata.csv' CSV HEADER;
COPY 756147
Citibike=#
```

After import

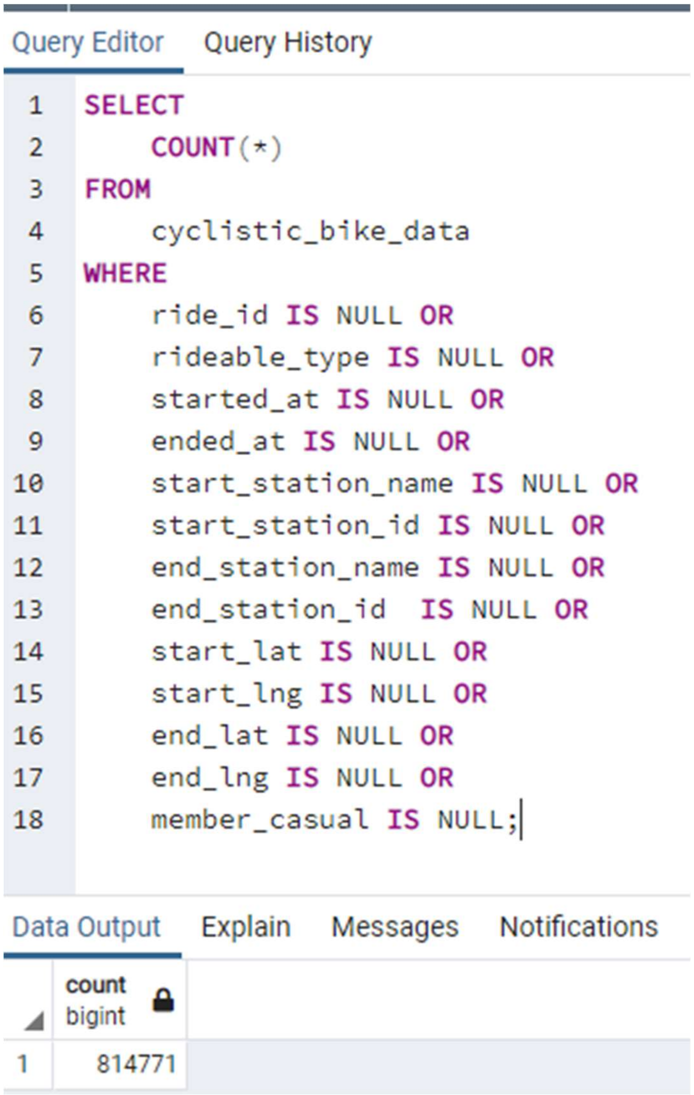
1. The total number of rides for the past 12 months is **5,669,219**.



```
SELECT
    COUNT(*)
FROM
    cyclistic_bike_data
```

```
   count
   bigint
1  5669219
```

2. A count of NULL values shows that there are **814,771** records with at least one null value.



```sql
SELECT
    COUNT(*)
FROM
    cyclistic_bike_data
WHERE
    ride_id IS NULL OR
    rideable_type IS NULL OR
    started_at IS NULL OR
    ended_at IS NULL OR
    start_station_name IS NULL OR
    start_station_id IS NULL OR
    end_station_name IS NULL OR
    end_station_id  IS NULL OR
    start_lat IS NULL OR
    start_lng IS NULL OR
    end_lat IS NULL OR
    end_lng IS NULL OR
    member_casual IS NULL;
```

```
   count
   bigint
1  814771
```

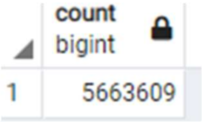
The null values are in features:

start_station_name
start_station_id
end_station_name
en d_station_id
start_lat
start_lng
end_lat
end_lng

CLEANING – Part One

KEY DECISION

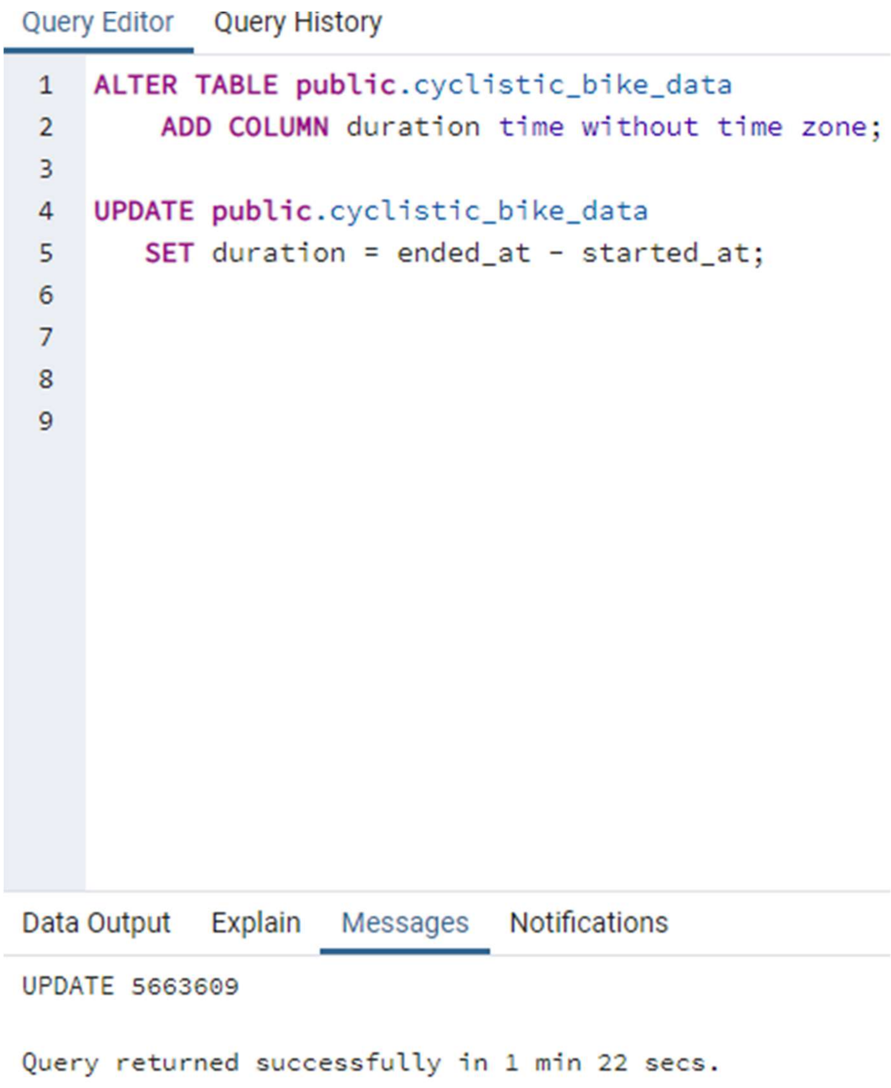
For the 5,610 rides with no ending station name, id, or geographical information, I removed these 5,610 records from the database leaving a total count **of 5,663,609**.



	count bigint
1	5663609

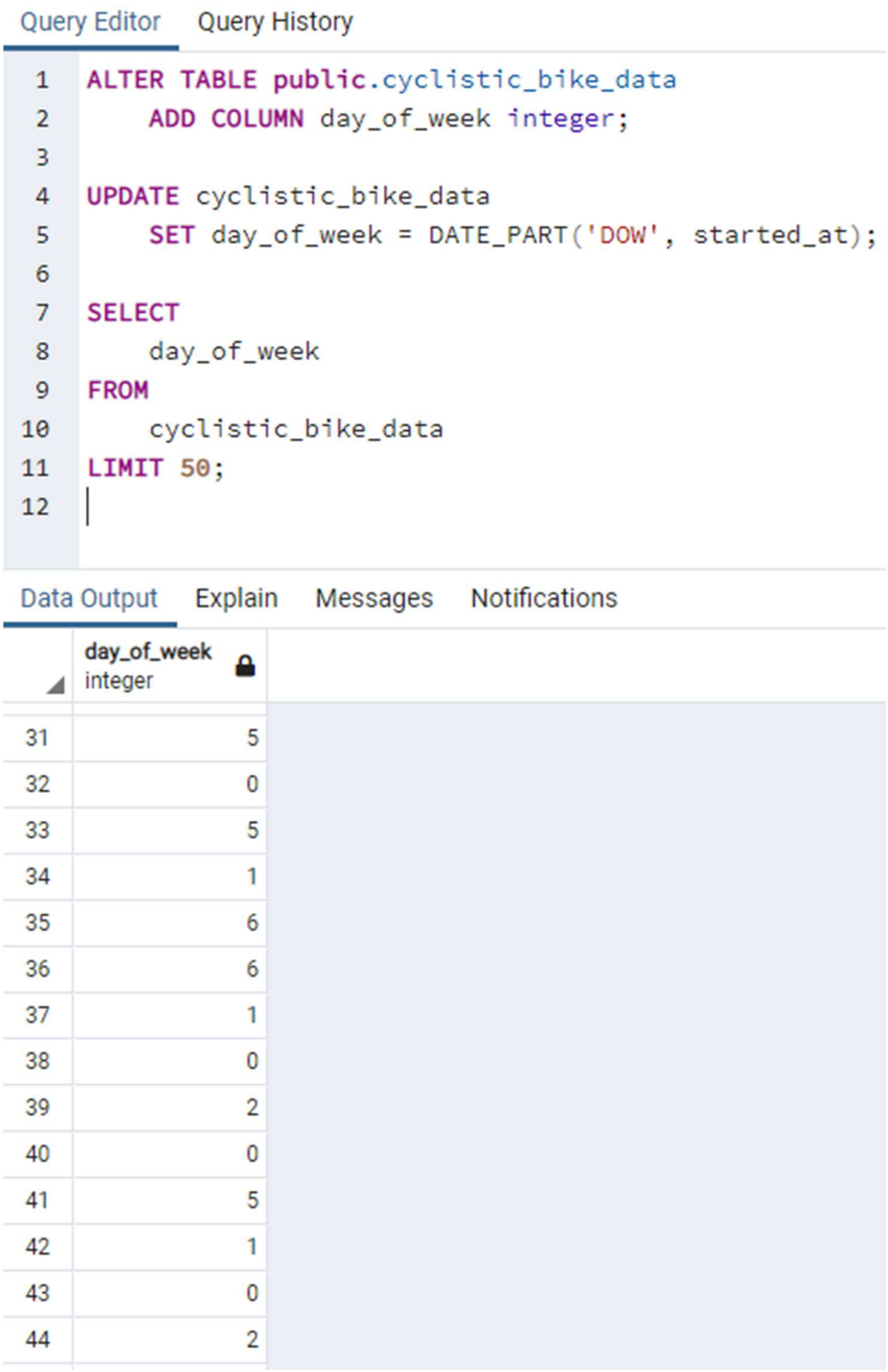
For the other null values, I will clean them in R.

3. Adding features to the table:
   duration – the total time of each ride
   day_of_week – the day of the week for each ride



```sql
ALTER TABLE public.cyclistic_bike_data
     ADD COLUMN duration time without time zone;

UPDATE public.cyclistic_bike_data
   SET duration = ended_at - started_at;
```

```
UPDATE 5663609

Query returned successfully in 1 min 22 secs.
```

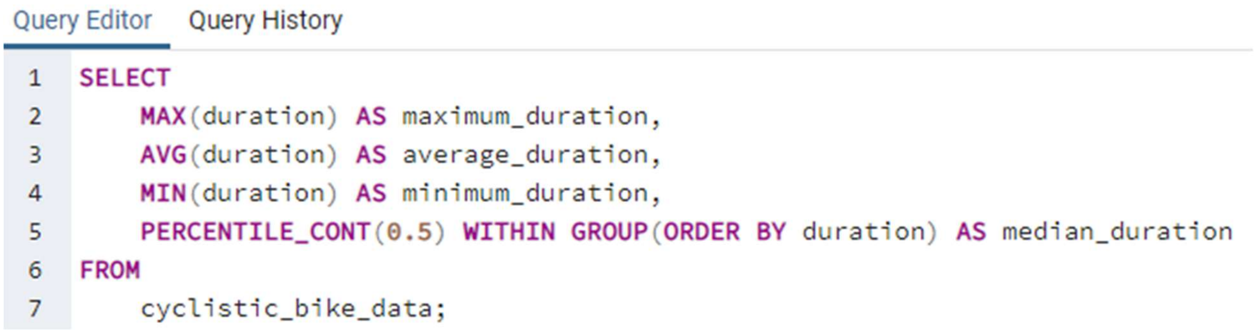


```sql
ALTER TABLE public.cyclistic_bike_data
    ADD COLUMN day_of_week integer;

UPDATE cyclistic_bike_data
    SET day_of_week = DATE_PART('DOW', started_at);

SELECT
    day_of_week
FROM
    cyclistic_bike_data
LIMIT 50;

```

	day_of_week integer
31	5
32	0
33	5
34	1
35	6
36	6
37	1
38	0
39	2
40	0
41	5
42	1
43	0
44	2

4. Basic Metrics

A. Max, Min, Avg, and Median



```
SELECT
    MAX(duration) AS maximum_duration,
    AVG(duration) AS average_duration,
    MIN(duration) AS minimum_duration,
    PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY duration) AS median_duration
FROM
    cyclistic_bike_data;
```

The results from the query above show that the

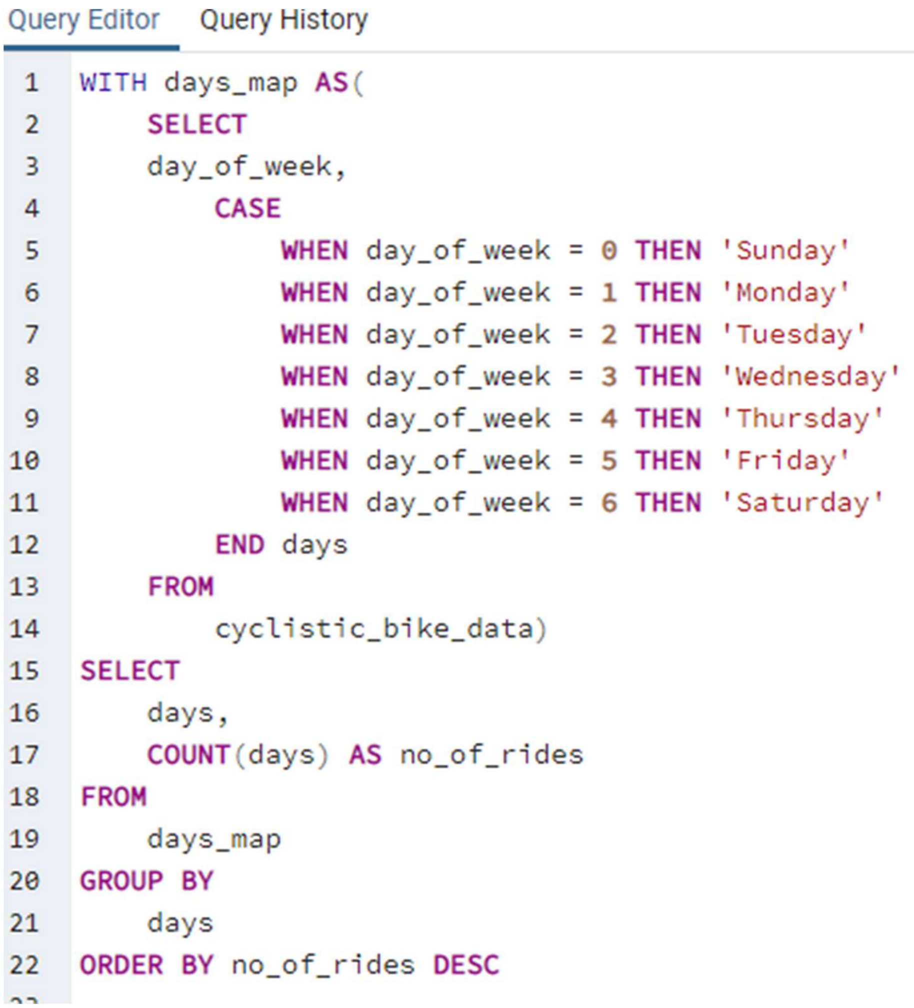
Maximum duration for a ride is 23:59:59

Minimum duration for a ride is 00:00:00

Average duration is 00:21:52

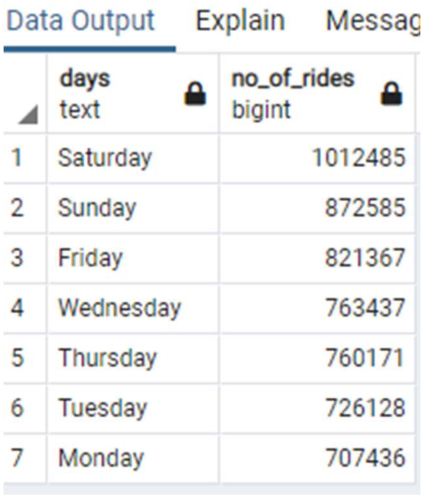
Median duration is 00:12:44

B. Days of the Week



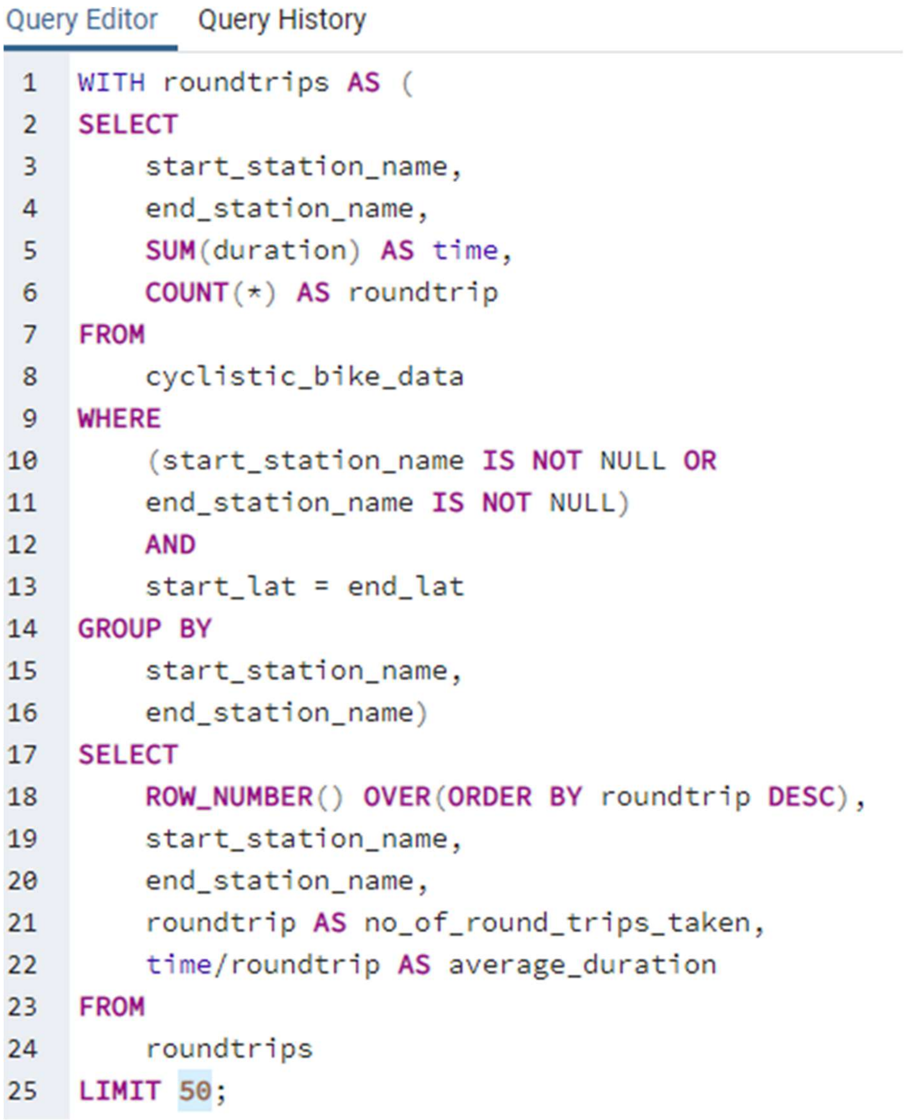
```
WITH days_map AS(
    SELECT
    day_of_week,
        CASE
            WHEN day_of_week = 0 THEN 'Sunday'
            WHEN day_of_week = 1 THEN 'Monday'
            WHEN day_of_week = 2 THEN 'Tuesday'
            WHEN day_of_week = 3 THEN 'Wednesday'
            WHEN day_of_week = 4 THEN 'Thursday'
            WHEN day_of_week = 5 THEN 'Friday'
            WHEN day_of_week = 6 THEN 'Saturday'
        END days
    FROM
        cyclistic_bike_data)
SELECT
    days,
    COUNT(days) AS no_of_rides
FROM
    days_map
GROUP BY
    days
ORDER BY no_of_rides DESC
```

This query returns the days of the and the number of rides for each day. Saturdays, Sundays, and Fridays are the most popular days:



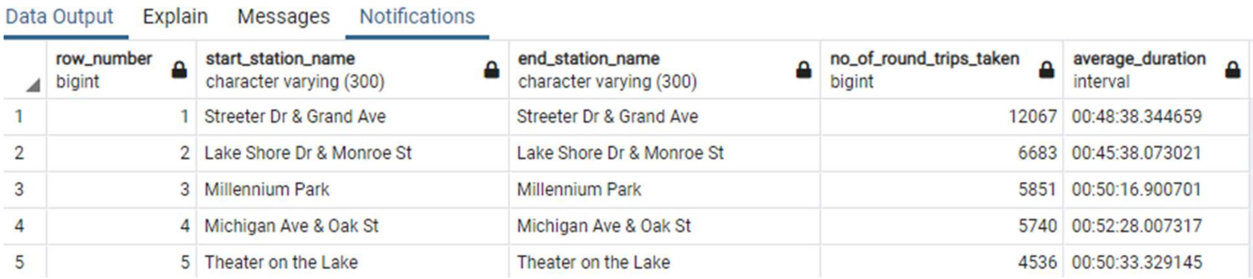
	days text	no_of_rides bigint
1	Saturday	1012485
2	Sunday	872585
3	Friday	821367
4	Wednesday	763437
5	Thursday	760171
6	Tuesday	726128
7	Monday	707436

C. Roundtrips



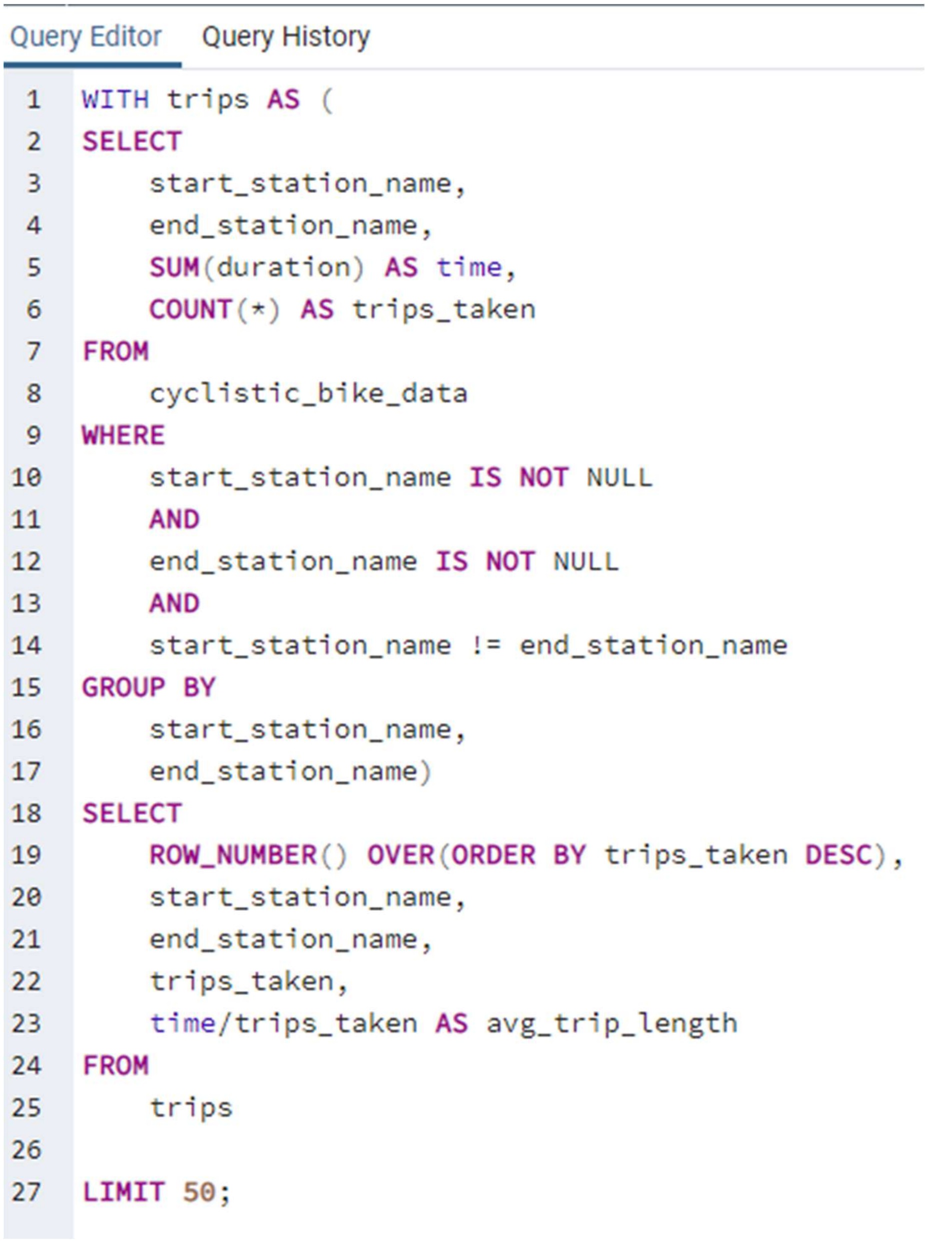
```
WITH roundtrips AS (
SELECT
    start_station_name,
    end_station_name,
    SUM(duration) AS time,
    COUNT(*) AS roundtrip
FROM
    cyclistic_bike_data
WHERE
    (start_station_name IS NOT NULL OR
    end_station_name IS NOT NULL)
    AND
    start_lat = end_lat
GROUP BY
    start_station_name,
    end_station_name)
SELECT
    ROW_NUMBER() OVER(ORDER BY roundtrip DESC),
    start_station_name,
    end_station_name,
    roundtrip AS no_of_round_trips_taken,
    time/roundtrip AS average_duration
FROM
    roundtrips
LIMIT 50;
```

The above query calculates the top 50 most popular round trips where the start and ending stations are the same. The Streeter Drive and Grand Ave round trip is almost twice as popular as the next round trip.

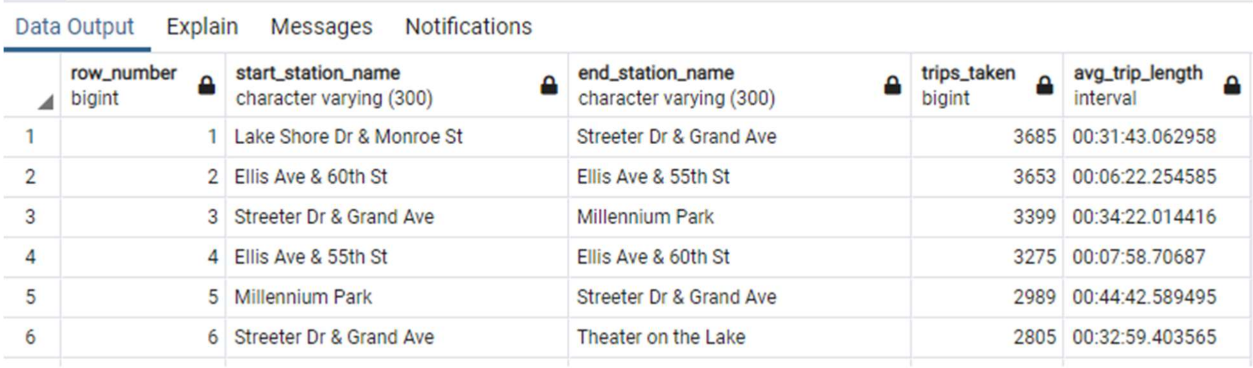


	row_number bigint	start_station_name character varying (300)	end_station_name character varying (300)	no_of_round_trips_taken bigint	average_duration interval
1	1	Streeter Dr & Grand Ave	Streeter Dr & Grand Ave	12067	00:48:38.344659
2	2	Lake Shore Dr & Monroe St	Lake Shore Dr & Monroe St	6683	00:45:38.073021
3	3	Millennium Park	Millennium Park	5851	00:50:16.900701
4	4	Michigan Ave & Oak St	Michigan Ave & Oak St	5740	00:52:28.007317
5	5	Theater on the Lake	Theater on the Lake	4536	00:50:33.329145

D. Single Trips



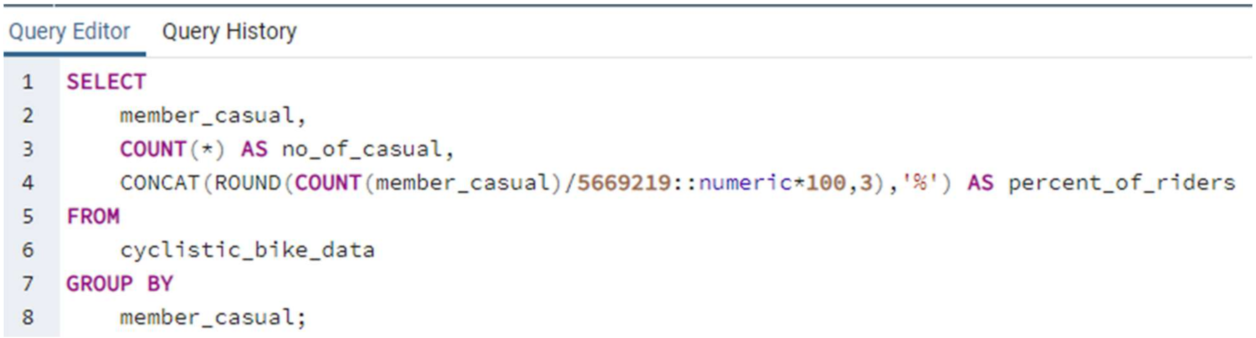
```sql
WITH trips AS (
SELECT
    start_station_name,
    end_station_name,
    SUM(duration) AS time,
    COUNT(*) AS trips_taken
FROM
    cyclistic_bike_data
WHERE
    start_station_name IS NOT NULL
    AND
    end_station_name IS NOT NULL
    AND
    start_station_name != end_station_name
GROUP BY
    start_station_name,
    end_station_name)
SELECT
    ROW_NUMBER() OVER(ORDER BY trips_taken DESC),
    start_station_name,
    end_station_name,
    trips_taken,
    time/trips_taken AS avg_trip_length
FROM
    trips

LIMIT 50;
```



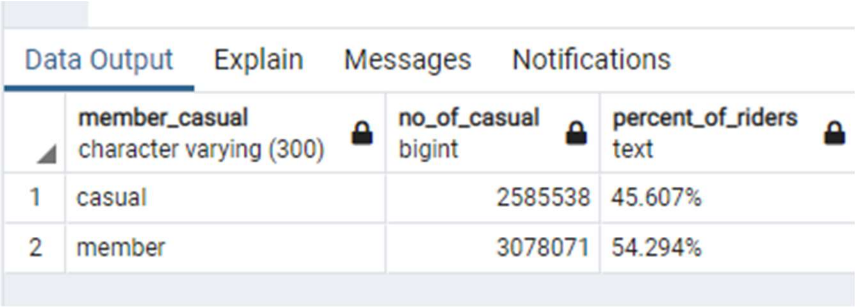
	row_number bigint	start_station_name character varying (300)	end_station_name character varying (300)	trips_taken bigint	avg_trip_length interval
1	1	Lake Shore Dr & Monroe St	Streeter Dr & Grand Ave	3685	00:31:43.062958
2	2	Ellis Ave & 60th St	Ellis Ave & 55th St	3653	00:06:22.254585
3	3	Streeter Dr & Grand Ave	Millennium Park	3399	00:34:22.014416
4	4	Ellis Ave & 55th St	Ellis Ave & 60th St	3275	00:07:58.70687
5	5	Millennium Park	Streeter Dr & Grand Ave	2989	00:44:42.589495
6	6	Streeter Dr & Grand Ave	Theater on the Lake	2805	00:32:59.403565

The above query calculates the top 50 most popular single trips where the start and ending stations are not the same. The most popular are Lake Shore Drive & Monroe to Streeter Dr & Grand Ave, Ellis Ave & 60th Street and Ellis Ave & 55th Street.

E. Casual vs. Members



```
SELECT
    member_casual,
    COUNT(*) AS no_of_casual,
    CONCAT(ROUND(COUNT(member_casual)/5669219::numeric*100,3),'%') AS percent_of_riders
FROM
    cyclistic_bike_data
GROUP BY
    member_casual;
```



	member_casual character varying (300)	no_of_casual bigint	percent_of_riders text
1	casual	2585538	45.607%
2	member	3078071	54.294%

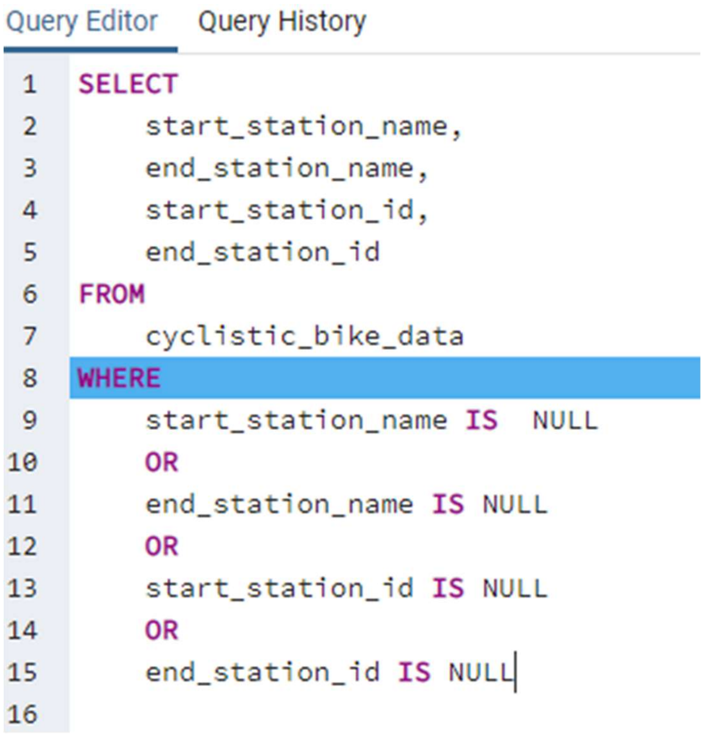
The results of the above query which counts the number of casual and member riders shows that there are 9% more members than casual riders.

CLEANING – Part Two

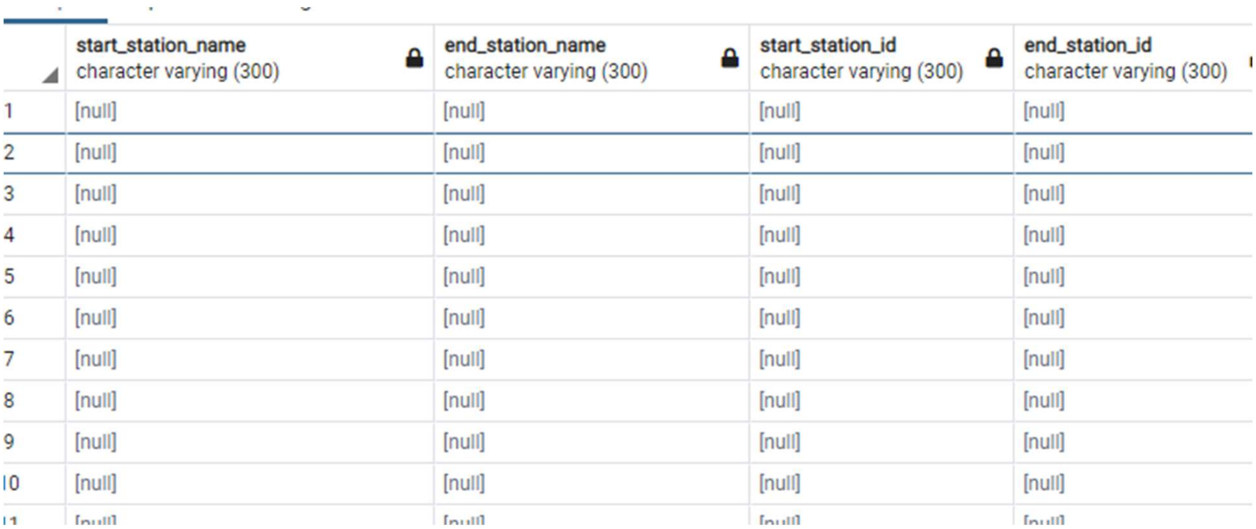
KEY DECISION

There are 809,161 records that have NULL values in the station name or id. These records are incomplete and including them in the analysis may lead to incorrect recommendations. There is no way to impute the null values due to mis-matching values in the latitude and longitude features in the dataset.

As a result, the only way to handle these 809,161 will be to delete them from further analysis.



```
SELECT
    start_station_name,
    end_station_name,
    start_station_id,
    end_station_id
FROM
    cyclistic_bike_data
WHERE
    start_station_name IS  NULL
    OR
    end_station_name IS NULL
    OR
    start_station_id IS NULL
    OR
    end_station_id IS NULL

```

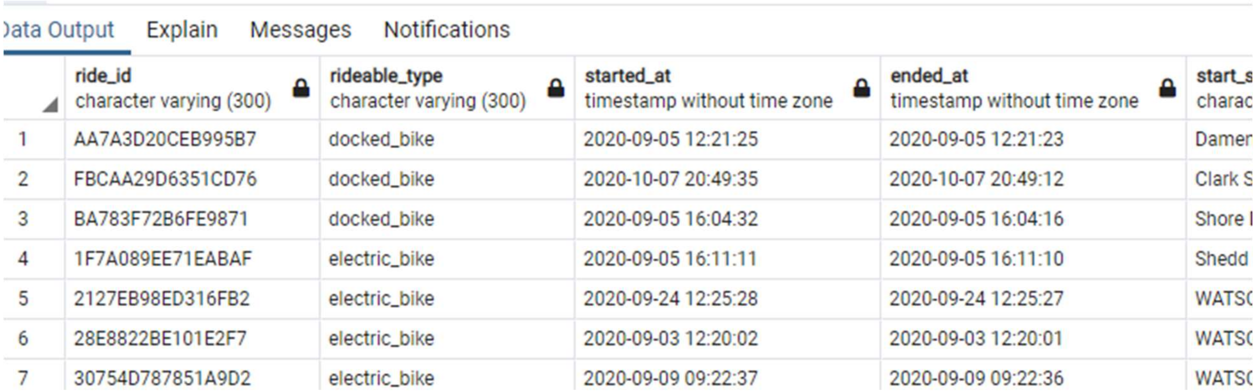


	start_station_name character varying (300)	end_station_name character varying (300)	start_station_id character varying (300)	end_station_id character varying (300)
1	[null]	[null]	[null]	[null]
2	[null]	[null]	[null]	[null]
3	[null]	[null]	[null]	[null]
4	[null]	[null]	[null]	[null]
5	[null]	[null]	[null]	[null]
6	[null]	[null]	[null]	[null]
7	[null]	[null]	[null]	[null]
8	[null]	[null]	[null]	[null]
9	[null]	[null]	[null]	[null]
10	[null]	[null]	[null]	[null]
11	[null]	[null]	[null]	[null]

CLEANING – Part Three

KEY DECISION

There are 5,333 records where the ended_at time is less than the started_at time. These records will have to be removed in order to preserve the integrity of the data:



	ride_id	rideable_type	started_at	ended_at	start_s
	character varying (300)	character varying (300)	timestamp without time zone	timestamp without time zone	charac
1	AA7A3D20CEB995B7	docked_bike	2020-09-05 12:21:25	2020-09-05 12:21:23	Damer
2	FBCAA29D6351CD76	docked_bike	2020-10-07 20:49:35	2020-10-07 20:49:12	Clark S
3	BA783F72B6FE9871	docked_bike	2020-09-05 16:04:32	2020-09-05 16:04:16	Shore I
4	1F7A089EE71EABAF	electric_bike	2020-09-05 16:11:11	2020-09-05 16:11:10	Shedd
5	2127EB98ED316FB2	electric_bike	2020-09-24 12:25:28	2020-09-24 12:25:27	WATS(
6	28E8822BE101E2F7	electric_bike	2020-09-03 12:20:02	2020-09-03 12:20:01	WATS(
7	30754D787851A9D2	electric_bike	2020-09-09 09:22:37	2020-09-09 09:22:36	WATS(

After removing these records, there are 4,849,115 records which will be analyzed.

ANALYZE

(See RMD Notebook File)

Key Findings

- Casual riders, who are less than half of all riders, ride mostly on the weekends
- Casual riders ride for less time during the day than members
- The starting points and routes for the casual rider occur near tourist destinations

Recommendations

- Create a new class of memberships by offering weekend only passes
- The benefit to the casual rider will be increased time on the bike without penalty
- Prices for the weekend passes should be priced at a discount for two day passes
- Prices for the day and hour passes on the weekend should be raised in order to boost the sales of the weekend passes