The following is an example of a recent data project I completed. I used SQL to extract data from a server, Excel to clean, analyse the present clear and concise charting. I investigated the differences between two bike sharing companies in the US. Click the presentation link to see my short PowerPoint presentation of the work. This document contains example SQL queries, Excel charts and important points that are summarized in the PPT presentation.

Project 2: Bike Share Data Analysis – Trevor Alback

Presentation Link: https://youtu.be/RIBOYMDOZ2E

I've chosen to explore data on two companies: Bluebikes (Boston) and Divvy (Chicago). I chose these because both have data covering years 2016-2019 as well as having gender and age user information. This will allow good comparison.

Question 1 – How many trips were there in each month of each year?

An SQL query for both companies can be run to generate a list showing counts of all trips ordered by year and month from 2016 to 2019. The queries and results for both companies are below.

Query (Bluebikes)

```
WITH

full_data (bike_id, start_time, end_time, start_station_id, end_station_id, user_type,
user_birth_year, user_gender) AS

(SELECT *

FROM bluebikes_2016

UNION ALL

SELECT *

FROM bluebikes_2017

UNION ALL

SELECT *

FROM bluebikes_2018

UNION ALL

SELECT *

FROM bluebikes_2019)


SELECT COUNT(*), DATE_PART('month', start_time) AS month_sep, DATE_PART('year', start_time)
AS year_sep

FROM full_data

GROUP BY month_sep, year_sep
```

ORDER BY year_sep, month_sep

Result (cont. across all months and years)

Count	Month	Year
12055	1	2016
14643	2	2016
41277	3	2016

Query (Divvy)

```
WITH

full_data_div (trip_id, bike_id, start_time, end_time, start_station_id, end_station_id, user_type, gender, birthyear) AS

(SELECT *

FROM divvybikes_2016

UNION ALL

SELECT *

FROM divvybikes_2017

UNION ALL

SELECT *

FROM divvybikes_2018

UNION ALL

SELECT *

FROM divvybikes_2019)


SELECT COUNT(*), DATE_PART('month', start_time) AS month_sep, DATE_PART('year', start_time) AS year_sep

FROM full_data_div

GROUP BY month_sep, year_sep

ORDER BY year_sep, month_sep
```

Result

COUNT	Month	Year
92839	1	2016
118120	2	2016

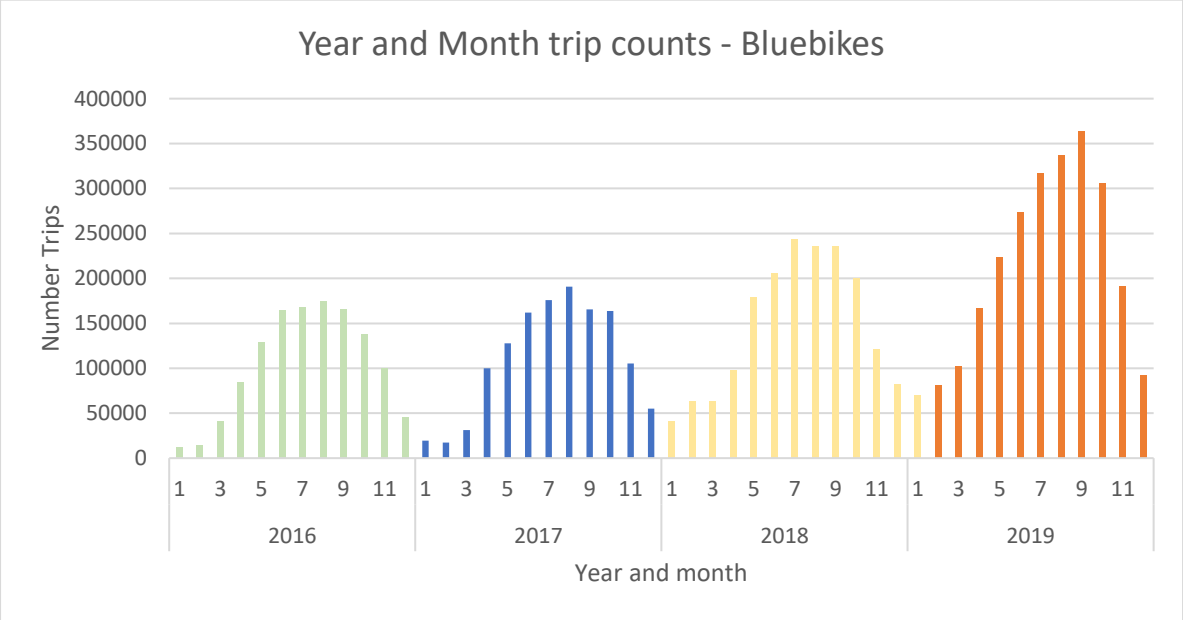
185954 3 2016

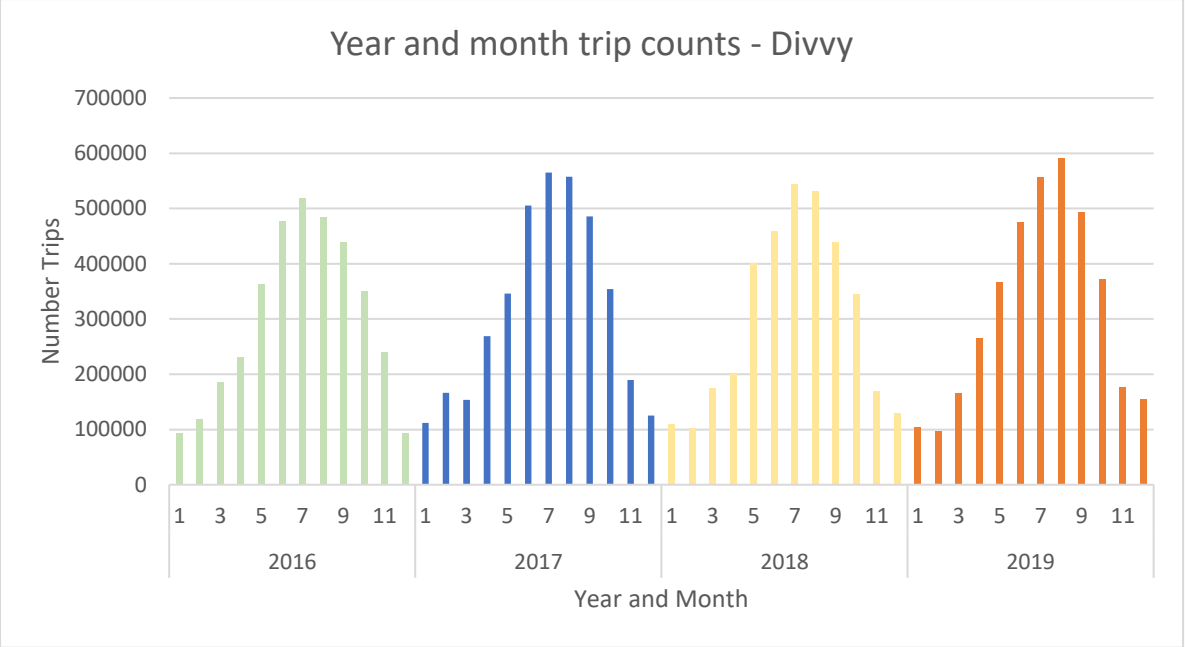
Comments

Months are 1-12 for Jan-Dec respectively. Counts of distinct bike_id's was not used because one bike_id can have multiple trips. Count (*) shows distinct (or unique) rows which will refer to one trip.

Visualization

The two charts below show the month and year break down for both companies. Both companies show yearly trends. Winter (in the US) has a reduction in trips, and summers (mid-year) show peaks. Bluebikes shows some visual evidence of year on year total trip increase (as bars increasingly get larger), but it will be better to investigate this quantitatively (see next question).

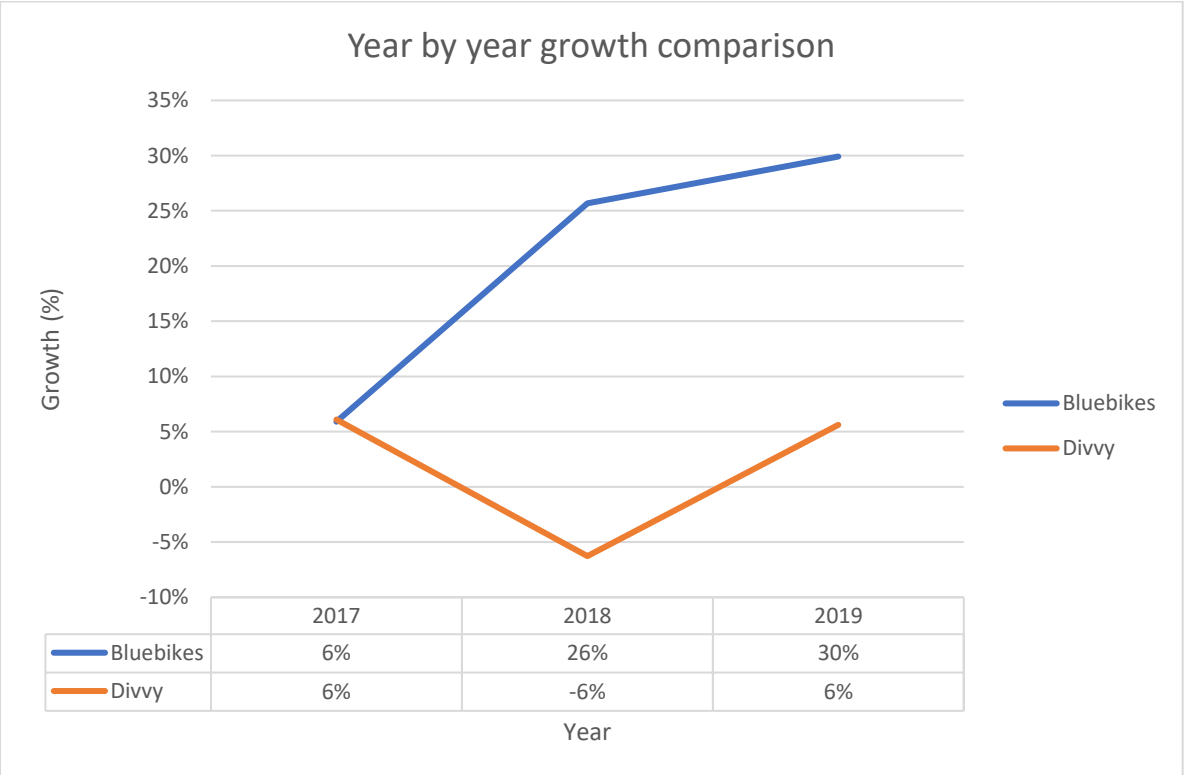




Question 2 – Which organization is showing the most growth in bike rentals?

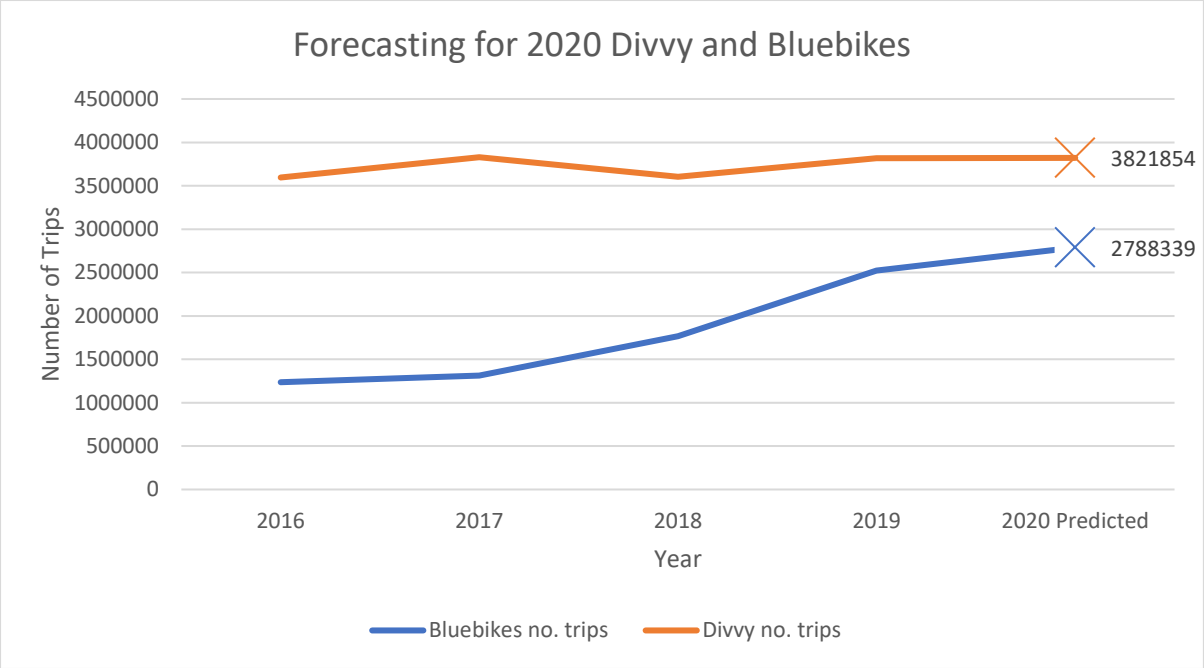
Below is a chart showing the growth rates year on year for both Bluebikes and Divvy. 2016 cannot be calculated as previous year data (2015) is not available.

Bluebikes has grown year on year. Divvy on the other hand has stayed about the same over the time period (with a dip in 2018). This displays that Bluebikes has shown more growth than Divvy.

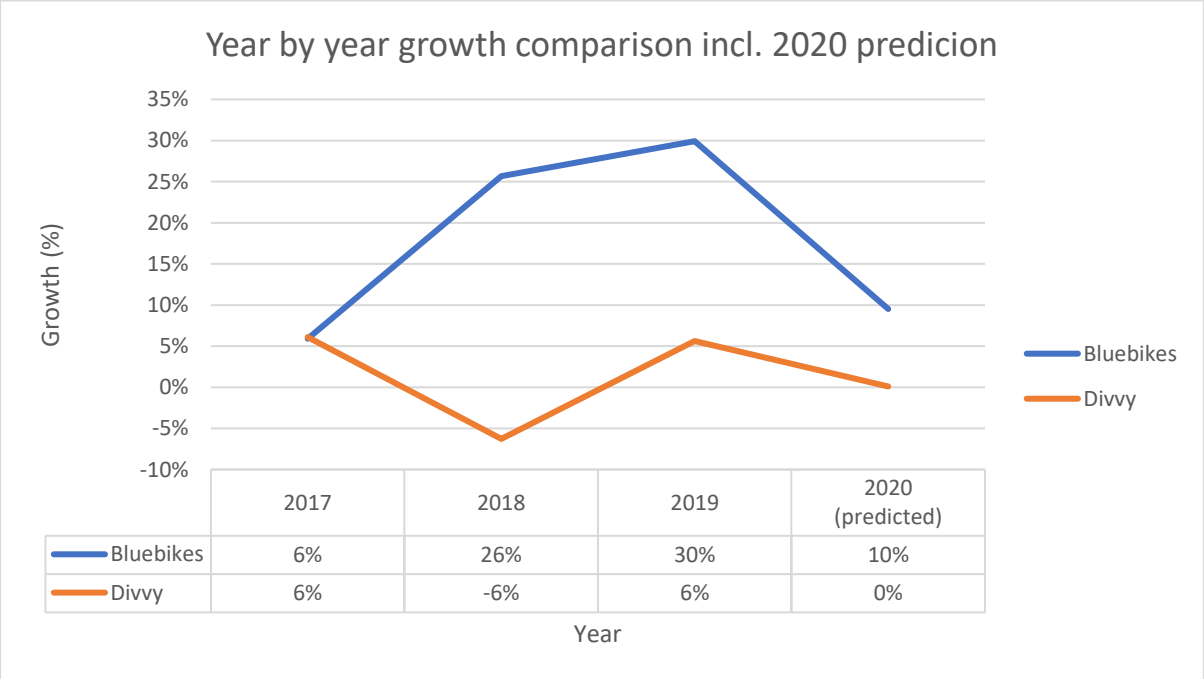


Question 3 – What would you expect to see in 2020 (assuming a COVID free 2020)?

The chart below shows the year by year total number of trips for both companies. I've predicted the 2020 number of trips based on linear regression. Note that this is an assumption and that there are only a couple of data points to make this prediction and therefore it should be treated as so. The second chart uses these predicted 2020 results to calculate the estimated growth rate for 2020 for both companies.



Using this 2020 predicted result, the expected growth including 2020 is below. Even though growth for Bluebikes will continue, it's not expected to grow as much as it did from 2018-2019 where there was a large uptake of trips. Divvy shouldn't expect to see significant growth based on these predictions which follow the previous year trends.



Question 4 – Longest trip

Using the coordinate points (latitude and longitude) for both start and end station, it was possible to run a query to calculate the distance in kilometres between stations. I limited the query to 100,000 rows due to its slow computation. Ordering also slowed the query significantly, so I ran it unordered, export to excel and continue analysis there.

Query (Bluebikes 2019, 100000 subset)

```
WITH
start_station AS
(SELECT bb.start_station_id,
                bb_s.latitude, bb_s.longtitude
        FROM bluebikes_2019 AS bb
        INNER JOIN bluebikes_stations AS bb_s
        ON bb.start_station_id = bb_s.id
),
end_station  AS
(       SELECT  bb1.end_station_id,
        bb_e.latitude, bb_e.longtitude
        FROM bluebikes_2019 AS bb1
        INNER JOIN bluebikes_stations AS bb_e
        ON bb1.end_station_id = bb_e.id
        )
SELECT calculate_distance(a.latitude, a.longtitude,
                                                b.latitude, b.longtitude,
                                                'K') AS dist
FROM start_station a,
        end_station b
WHERE a.start_station_id != b.end_station_id
LIMIT 100000
```

Result (first 5 rows)

Distance
1.37950086620763
4.18504515174988

0.830818297571687

1.24132917904525

4.84590985184304

A max function in excel on the exported data shows the longest trip in my subset of data for Bluebikes in 2019 was **11.4km**.

Query (Divvy 2019, 100,000 subset)

```
WITH
start_station AS
(SELECT div.start_station_id,
                div_s.latitude, div_s.longitude
        FROM divvybikes_2019 AS div
        INNER JOIN divvy_stations AS div_s
        ON div.start_station_id = div_s.id
),
end_station  AS
(       SELECT  div1.end_station_id,
        div_e.latitude, div_e.longitude
        FROM divvybikes_2019 AS div1
        INNER JOIN divvy_stations AS div_e
        ON div1.end_station_id = div_e.id
        )
SELECT calculate_distance(a.latitude, a.longitude,
                                                b.latitude, b.longitude,
                                                'K') AS dist
FROM start_station a,
        end_station b
WHERE a.start_station_id != b.end_station_id
LIMIT 100000
```

Result

Distance

4.84094150490896

9.94771597543802

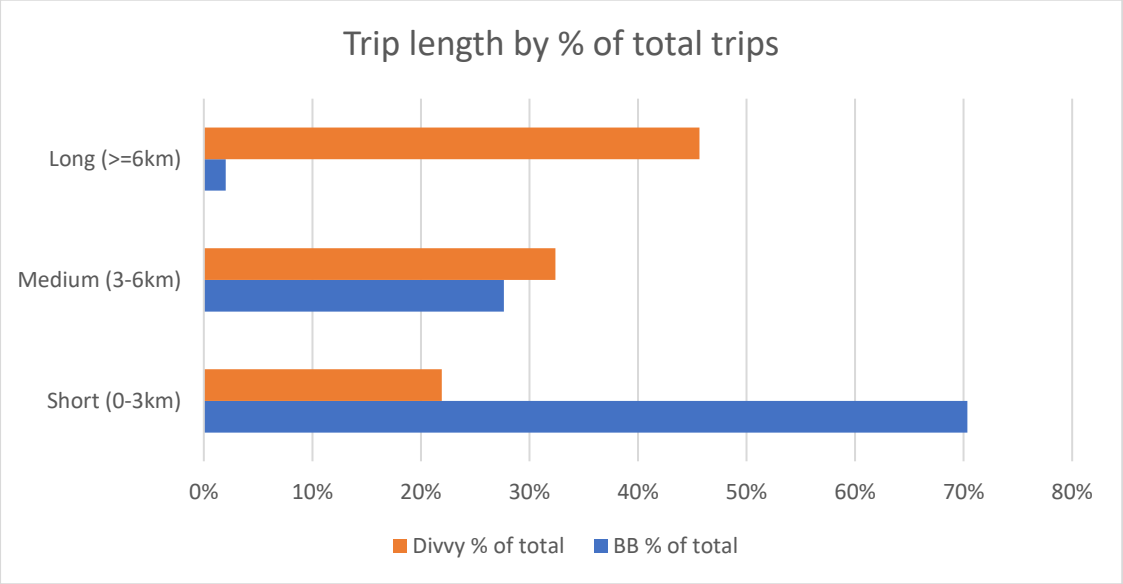
2.16992339954214

2.16992339954214

10.2720051847899

Output exported to excel to find max trip distance of: **23.3km**

The chart below shows a comparison of trip lengths by length group. Bluebikes have a large % of short trips and Divvy have a large % or long trips.



Question 5 – How often are bikes relocated?

Below I've shown queries that lag the end_station by 1 row partitioned by bike_id. This allows a comparison of end_station and start_station.

Query (Bluebikes 2019)

```
WITH

lag_end

AS (SELECT

        bike_id, start_station_id, end_station_id, start_time, LAG(end_station_id,1) OVER (PARTITION BY bike_id ORDER BY start_time) AS end_station_lag

FROM bluebikes_2019

)


SELECT bike_id, COUNT(end_station_lag) AS bike_relocations

FROM lag_end
```

```
WHERE end_station_lag != start_station_id
GROUP BY bike_id
ORDER BY bike_relocations DESC
```

Results

Bike_id	count of bike relocations
3363	102
4462	100
4345	99
3787	97
3797	97

Query (Divvy 2019)

```
WITH
lag_end
AS (SELECT
    bikeid, start_station_id, end_station_id, start_time, LAG(end_station_id,1) OVER (PARTITION BY bikeid ORDER BY start_time) AS end_station_lag
FROM divvybikes_2019)

SELECT bikeid, COUNT(end_station_lag) AS bike_relocations
FROM lag_end
WHERE end_station_lag != start_station_id
GROUP BY bikeid
ORDER BY bike_relocations DESC
```

Result

Bike_id	count of bike relocations
3790	87
635	86
4230	81
5985	80
6122	79

Comments

Each bike_id will have one null entry for the first trip (no previous end_station_id). This is not included in the count function.

By grouping by bike_id and ordering by bike_relocations DESC, it's possible to see which bikes where relocated the most. Further analysis could be done on these bikes_ids to see if they operate on popular routes or end at stations where they need to be relocated more often. This may give insights into staffing and operation costs which may ultimately impact profits.

A sum of these in excel gives total counts per company as:

Bluebikes 2019: **173963**

Bluebikes total 2019: **2522537**

Divvy 2019: **266375**

Divvy total 2019: **3818004**

Through % of total, it's possible to see how many of the trips required bike relocation in 2019 and hence the rate of bike relocation over that year.

Bluebikes: **10.6%**

Divvy: **4.6%**

Approximately twice as many bike relocations were required by Bluebikes compared to Divvy in 2019.

Question 6 – How far is a typical journey compared by categories

Bluebikes state that their genders are 0 = unknown, 1 = male, 2 = female (https://www.bluebikes.com/system-data). Divvy state the gender as 'male' or 'female' within the data set. I've chosen to extract 100,000 subsets of male and female genders. I've included user_type to see if this category can also be investigated with respect to typical journey distances.

Query (Bluebikes 2019, 100,000 subset on gender = 1)

```
WITH

start_station AS

(SELECT bb.start_station_id, bb.user_gender, bb.user_type,

                bb_s.latitude, bb_s.longtitude

        FROM bluebikes_2019 AS bb

        INNER JOIN bluebikes_stations AS bb_s

        ON bb.start_station_id = bb_s.id

),

end_station  AS
```

```
(       SELECT  bb1.end_station_id,
        bb_e.latitude, bb_e.longtitude
        FROM bluebikes_2019 AS bb1
        INNER JOIN bluebikes_stations AS bb_e
        ON bb1.end_station_id = bb_e.id
        )
SELECT calculate_distance(a.latitude, a.longtitude,
                                        b.latitude, b.longtitude,
                                        'K') AS dist, a.user_gender, a.user_type
FROM start_station a,
        end_station b
WHERE a.start_station_id != b.end_station_id AND a.user_gender = 1
LIMIT 100000
```

Result

Distance	gender	user_type
1.37950086620763	1	"Subscriber"
1.35504499465761	1	"Subscriber"
0.62892496698094	1	"Subscriber"
4.09713683121008	1	"Subscriber"
1.68473125619126	1	"Subscriber"
2.12397066034696	1	"Subscriber"
1.08548922447279	1	"Customer"
1.70077669244531	1	"Customer"

Changing the WHERE filter to a.user_gender = 2 gives the distances output but by female gender.

Query (Divvy 2019, 100,000 subset on ‘Male’)

```
WITH
start_station AS
(SELECT div.start_station_id, div.gender, div.user_type,
                div_s.latitude, div_s.longitude
        FROM divvybikes_2019 AS div
```

```
        INNER JOIN divvy_stations AS div_s
        ON div.start_station_id = div_s.id
),
end_station  AS
(       SELECT  div1.end_station_id,
        div_e.latitude, div_e.longitude
        FROM divvybikes_2019 AS div1
        INNER JOIN divvy_stations AS div_e
        ON div1.end_station_id = div_e.id
        )
SELECT calculate_distance(a.latitude, a.longitude,
                                                b.latitude, b.longitude,
                                                'K') AS dist, a.gender, a.user_type
FROM start_station a,
        end_station b
WHERE a.start_station_id != b.end_station_id AND a.gender LIKE 'Male'
LIMIT 100000
```

Result

Distance	gender	user_type
4.84094150490896	"Male"	"Subscriber"
5.58319163115697	"Male"	"Subscriber"
9.78629047947665	"Male"	"Subscriber"
3.56925653746274	"Male"	"Subscriber"
3.17747454162246	"Male"	"Subscriber"
5.0066514738906	"Male"	"Customer"

Changing WHERE a.gender LIKE ‘Female’ brings distances by female user

The two charts below show the typical or average trip distances by gender and user type. As already alluded to in Q4 Divvy has more longer trips compared to Bluebikes. This is reinforced below with the average trip distances being longer with Divvy compared to Bluebikes. For both Bluebikes and Divvy, Subscriber trips are shorter than Customer trips. Also, female customers make slightly longer

trips on average for both companies. Female subscribers on the other hand make shorter trips compared to males across both companies.