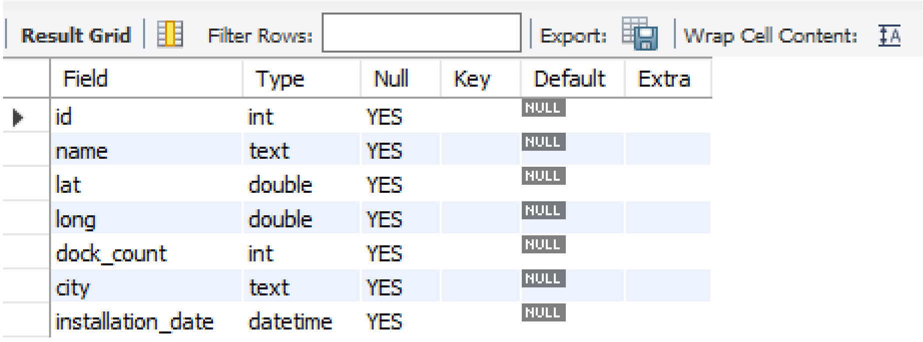
Contents

Importing Data from CSV to SQL

- Tables were available in csv format and were imported to MySQL for analysis.
- Columns which were having dates while importing were stored as datetime format so as to correctly represent the data type in MySQL for dates.
- STR_TO_DATE (time, '%Y/%m/%d %H:%i:%s') was applied on columns containing date as text format and then stored as datetime data format in my MySQL tables after data was imported from csv.

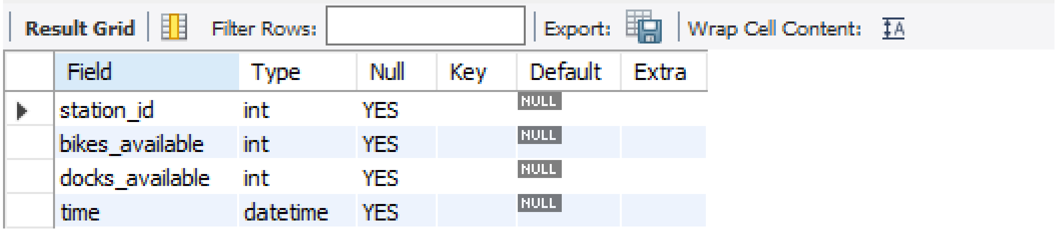
Schema for different tables is as follows after importing the data:

Station Table Schema: Contains information about each station like in which city it is located, station name and how many docks are present for the bikes.



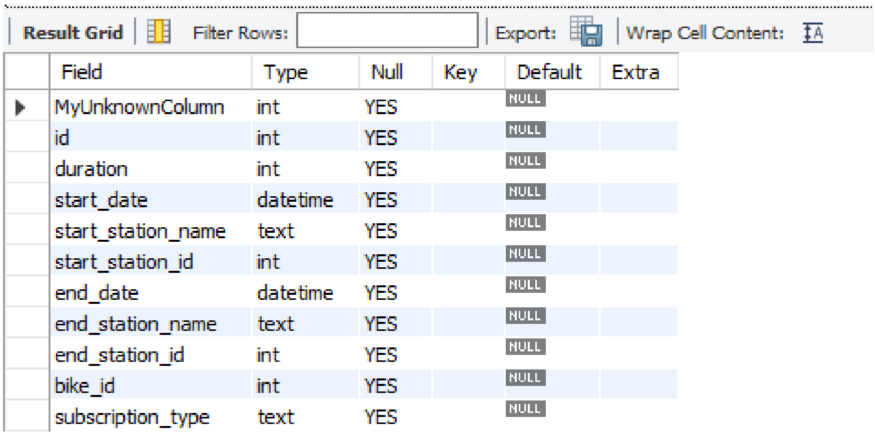
Field	Type	Null	Key	Default	Extra
id	int	YES		NULL	
name	text	YES		NULL	
lat	double	YES		NULL	
long	double	YES		NULL	
dock_count	int	YES		NULL	
city	text	YES		NULL	
installation_date	datetime	YES		NULL	

Status Table Schema: Bikes and docs available during different times of day.



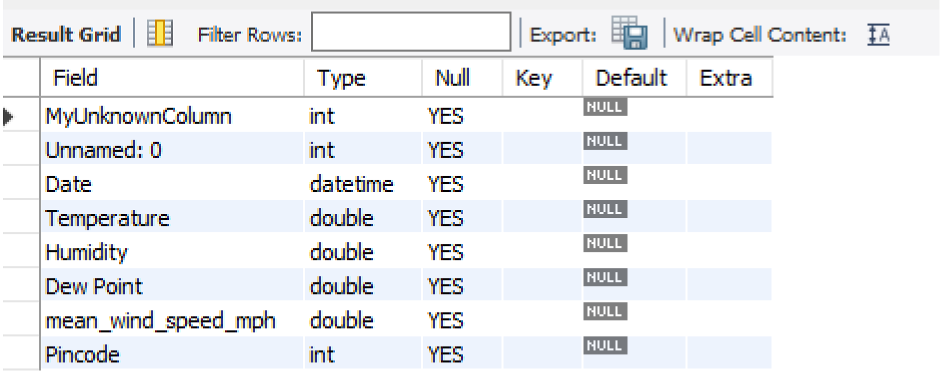
Field	Type	Null	Key	Default	Extra
station_id	int	YES		NULL	
bikes_available	int	YES		NULL	
docks_available	int	YES		NULL	
time	datetime	YES		NULL	

Trip Table Schema: User trip information



Field	Type	Null	Key	Default	Extra
MyUnknownColumn	int	YES		NULL	
id	int	YES		NULL	
duration	int	YES		NULL	
start_date	datetime	YES		NULL	
start_station_name	text	YES		NULL	
start_station_id	int	YES		NULL	
end_date	datetime	YES		NULL	
end_station_name	text	YES		NULL	
end_station_id	int	YES		NULL	
bike_id	int	YES		NULL	
subscription_type	text	YES		NULL	

Weather Table schema: Weather related attributes



Field	Type	Null	Key	Default	Extra
MyUnknownColumn	int	YES		NULL	
Unnamed: 0	int	YES		NULL	
Date	datetime	YES		NULL	
Temperature	double	YES		NULL	
Humidity	double	YES		NULL	
Dew Point	double	YES		NULL	
mean_wind_speed_mph	double	YES		NULL	
Pincode	int	YES		NULL	

Understanding the Data

1. What are total numbers of

a. Bike stations?

/*

CONTEXT:

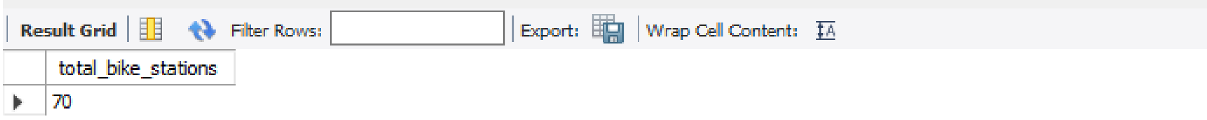
- Id in station table is unique for each station, so the count of rows in station table would give the number of bike stations.

RESULT EXPECTATION:

- Total number of bike stations

*/

SELECT COUNT(*) as total_bike_stations FROM station;



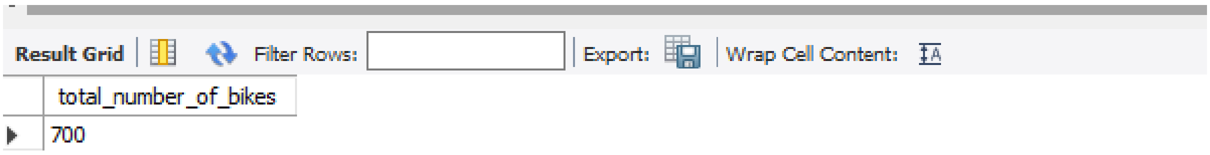
total_bike_stations
70

b. Bikes?

/*
CONTEXT:
- Bike ID is present in trip table.Count of distinct bike id's in trip table would be the number of bikes.
-As in trip table there can be rows with multiple bike id's so distinct is used for getting the unique bikes.
RESULT EXPECTATION:

- Total number of bikes
*/

```
SELECT COUNT(DISTINCT(bike_id)) as total_number_of_bikes
FROM trip;
```

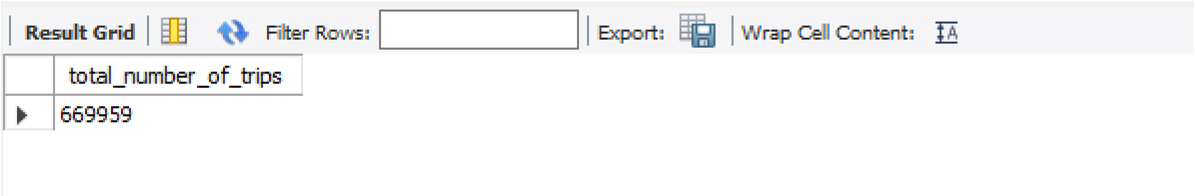


```
total_number_of_bikes
700
```

c. Trips?

/*
CONTEXT:
- Trip ID is present in trip table as id column. Count of distinct id's in trip table would be the number of trips.
RESULT EXPECTATION:
- Total number of trips
*/

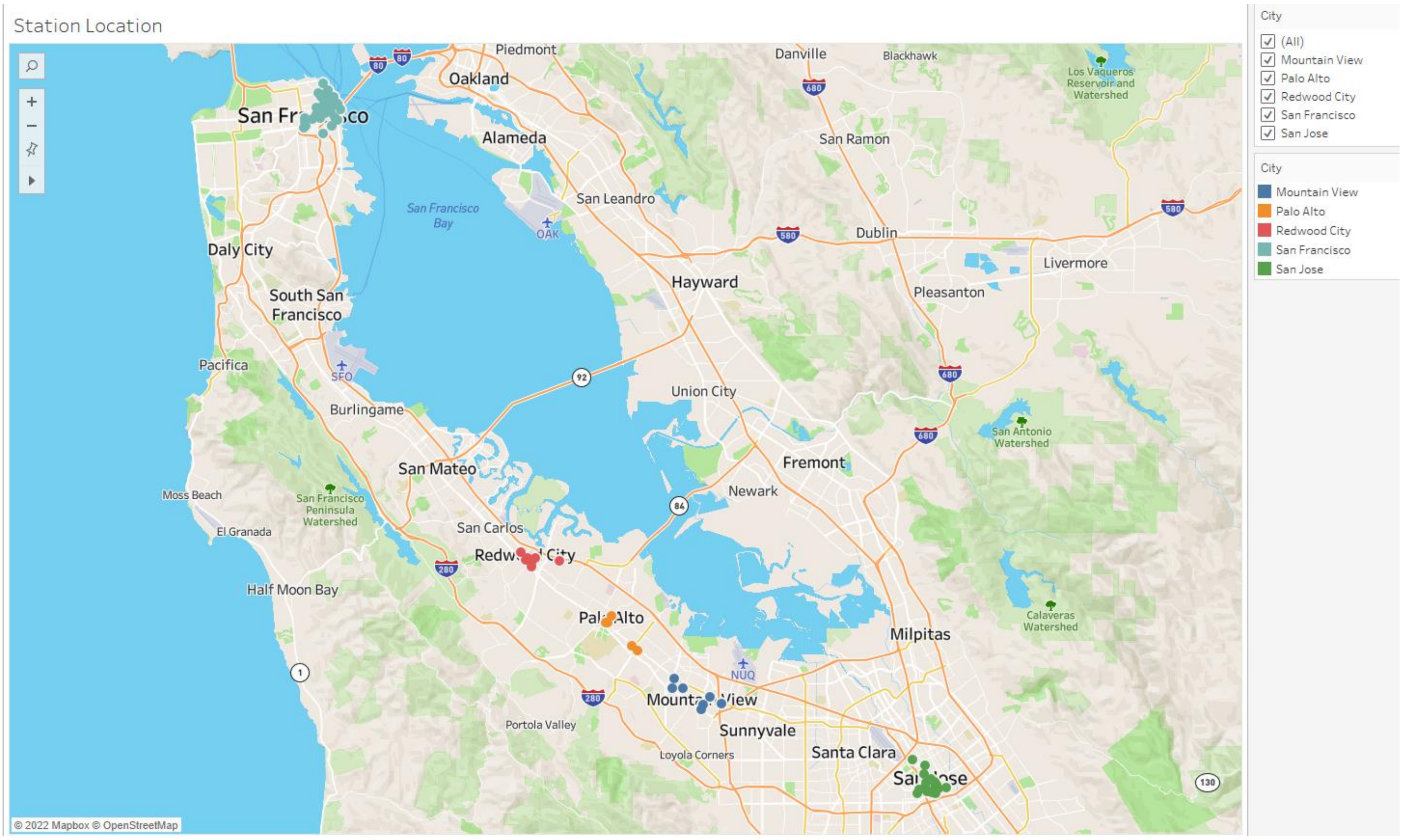
```
SELECT COUNT(DISTINCT(id)) as total_number_of_trips
FROM trip;
```



```
total_number_of_trips
669959
```

2. Constructing a geographical plot to show the location of each bike station using the latitude and longitude provided under the Station table to get an idea how is the data spread across different stations.
   Tableau public plot link:

   https://public.tableau.com/app/profile/har.shobhit.dayal/viz/StationLocation_16548405644840/Sheet1?publish=yes



Station Location

City

☑ (All)

☑ Mountain View

☑ Palo Alto

☑ Redwood City

☑ San Francisco

☑ San Jose

City

🟦 Mountain View

🟧 Palo Alto

🟥 Redwood City

🟩 San Francisco

🟩 San Jose

3. Exploring the relationship between the following columns (one to one, many to one, many to many)?
    a. bike_id (Trip table) and start_station_id (Trip table)
    Many to Many relationship

    b. pincode (Weather table) and station location (latitude and longitude in Station table)
    - **There does not exist a direct relationship between the station and weather table as data cannot be joined due to foreign key missing.**
    - **But if data was properly structured then it would have One(pincode) to many (latitude and longitude in Station table) relationship.**

    c. 8/29/2013 (date column in Weather table) and mean wind speed (Weather table)
    Many to Many relationship

4. Calculating the first and the last trip in the data.

/*

CONTEXT:

- First trip can be calculated by ordering data in trip table by start date in ascending order and taking first record using limit.

RESULT EXPECTATION:

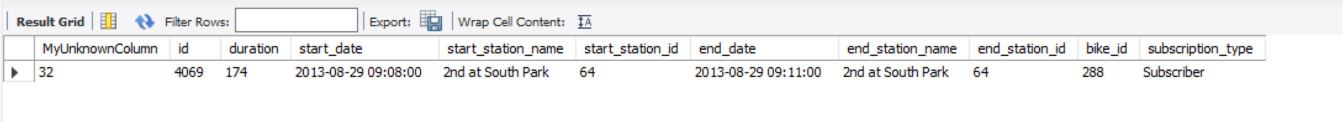
- First trip taken information

*/

SELECT * FROM trip

ORDER BY start_date

LIMIT 1;



MyUnknownColumn	id	duration	start_date	start_station_name	start_station_id	end_date	end_station_name	end_station_id	bike_id	subscription_type
32	4069	174	2013-08-29 09:08:00	2nd at South Park	64	2013-08-29 09:11:00	2nd at South Park	64	288	Subscriber

/*

CONTEXT:

- Last trip can be calculated by ordering data in trip table by start date in descending order and taking

first row using limit

RESULT EXPECTATION:

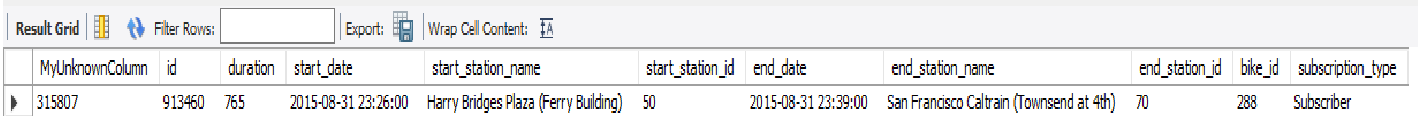
- Last trip taken information

*/

SELECT * FROM trip

ORDER BY start_date DESC

LIMIT 1;



MyUnknownColumn	id	duration	start_date	start_station_name	start_station_id	end_date	end_station_name	end_station_id	bike_id	subscription_type
315807	913460	765	2015-08-31 23:26:00	Harry Bridges Plaza (Ferry Building)	50	2015-08-31 23:39:00	San Francisco Caltrain (Townsend at 4th)	70	288	Subscriber

5. Getting to know the average duration

a. Of all the trips?

/*

CONTEXT:

- Average duration of all the trips can be calculated by making use of average function on duration column in trip table.

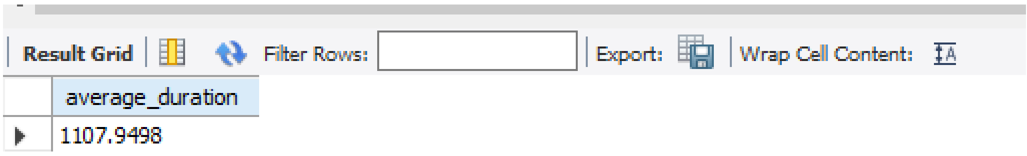
RESULT EXPECTATION:

- Average duration of all trips taken by customers

*/

SELECT AVG(duration) AS average_duration

FROM trip;



average_duration
1107.9498

b. Of trips on which customers are ending their rides at the same station from where they started?

/*

CONTEXT:

- Average duration of trips where customer started their rides and ended on the same station can be calculated by making use of average function on duration column and taking those records where start station name and end station name is same.

RESULT EXPECTATION:

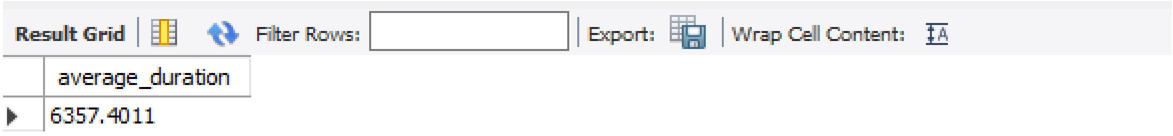
- Average duration of all trips where customers ended their rides at same station where they started

*/

SELECT AVG(duration) AS average_duration

FROM trip

WHERE start_station_name = end_station_name;



average_duration
6357.4011

6. Which bike has been used the most in terms of duration?

/*

CONTEXT:

- In order to calculate which bike has been used in terms of duration,records need to be grouped by bike

ids and aggregation of sum on duration should be computed.This would ensure that for a given bike id if there are

multiple records we take the sum of duration for each record.

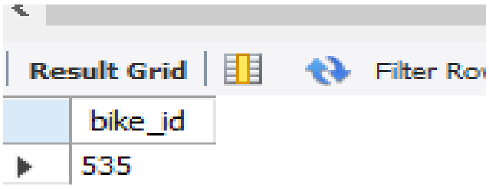
RESULT EXPECTATION:

- Bike which has been used most in terms of duration.

*/

SELECT bike_id

```
FROM trip

GROUP by bike_id

ORDER by SUM(duration) DESC limit 1;
```

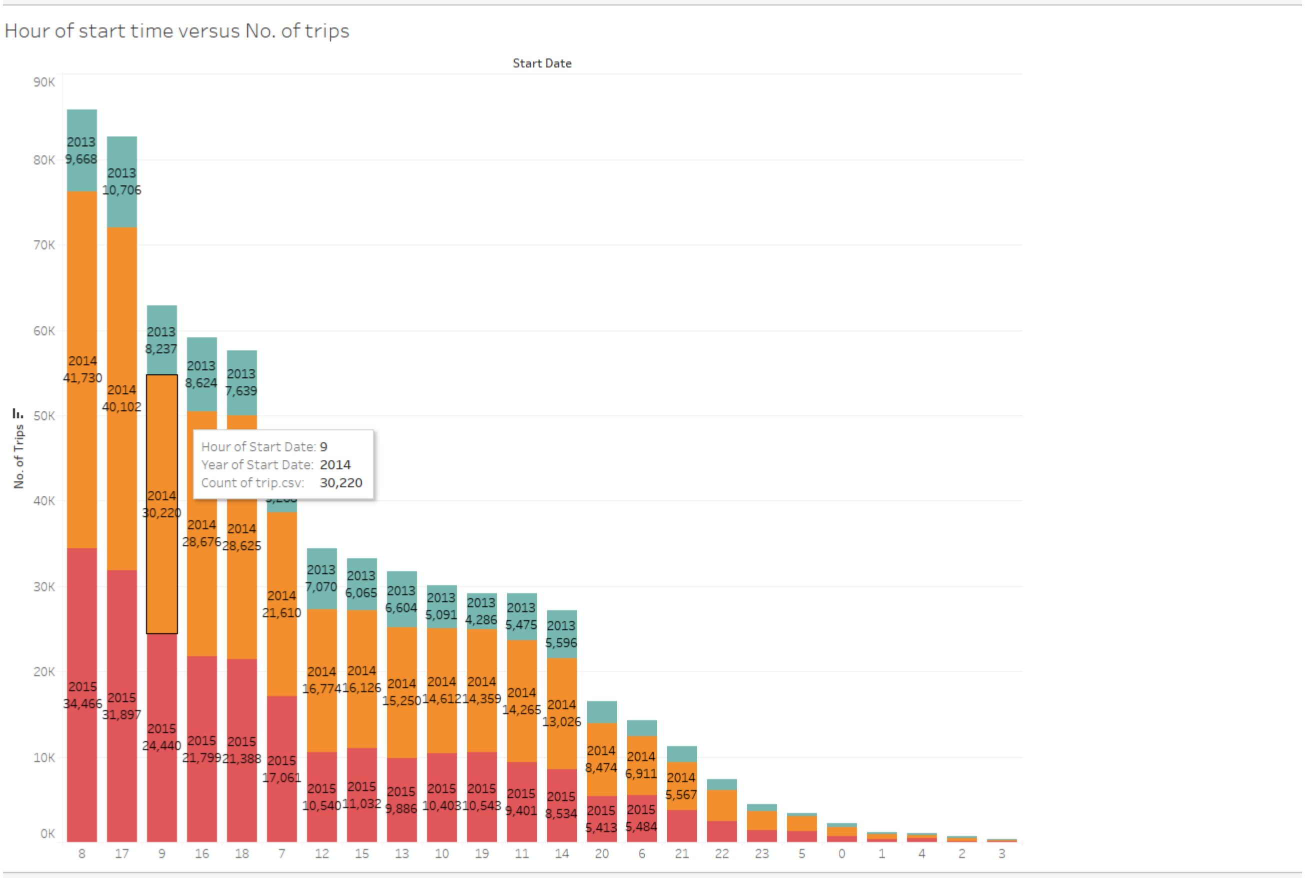


bike_id
535

7. Visualizing data for

   a. Hour of start time versus No. of trips

      Tableau public Link:
      https://public.tableau.com/app/profile/har.shobhit.dayal/viz/HourofstarttimeversusNo_oftrips_assignment/HourofstarttimeversusNo_oftrips?publish=yes

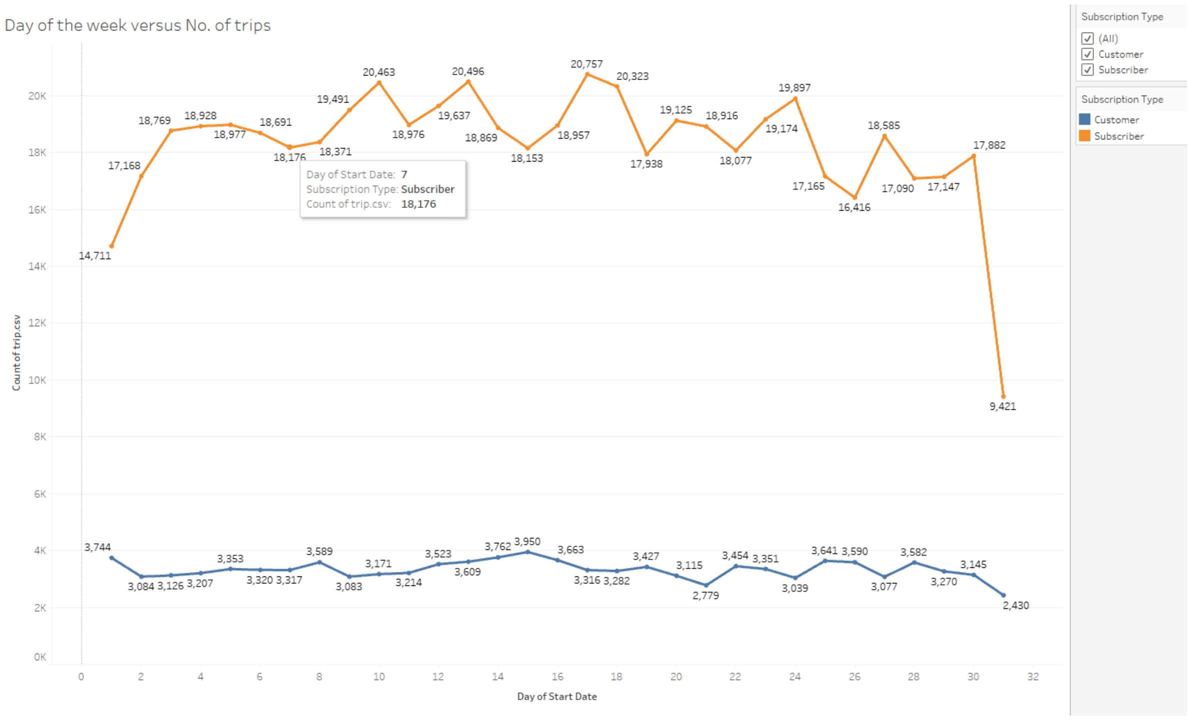


Hour of Start Date vs Number of Trips for that hour (Hour in 24hr format)

   b. Day of the week versus No. of trips also denote subscribers and customers with different colors.

      Tableau Public Link:

https://public.tableau.com/app/profile/har.shobhit.dayal/viz/DayoftheweekversusNo_oftrips_assignment/DayoftheweekversusNo_oftrips?publish=yes



Day of Week vs Number of Trips for that Day

Demand Prediction

1. What are the top 10 least popular stations?

/*

CONTEXT:

- Least popular stations would be those where few customers take the bikes from the station.

- This can be calculated by counting the number of times a station name comes in start station name column

- Limit of 10 can be used to get top 10 least popular stations.

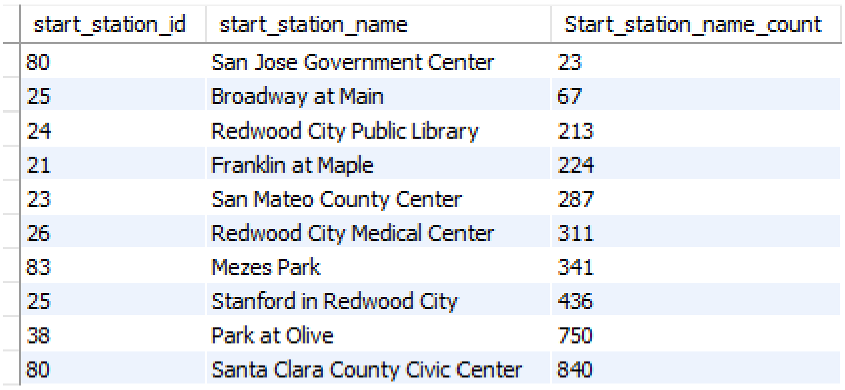
RESULT EXPECTATION:

- Top 10 least popular station information

*/

```
SELECT start_station_id,start_station_name,COUNT(start_station_name) AS Start_station_name_count

FROM trip

GROUP BY start_station_name
```

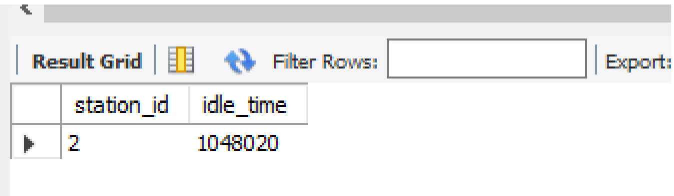
```
ORDER BY count(start_station_name)

LIMIT 10;
```



start_station_id	start_station_name	Start_station_name_count
80	San Jose Government Center	23
25	Broadway at Main	67
24	Redwood City Public Library	213
21	Franklin at Maple	224
23	San Mateo County Center	287
26	Redwood City Medical Center	311
83	Mezes Park	341
25	Stanford in Redwood City	436
38	Park at Olive	750
80	Santa Clara County Civic Center	840

2. Idle time is the duration for which a station remains inactive. Idle time needs to be calculated for each station which would help in seeing at what time of day the station does not require bikes(supply-demand)

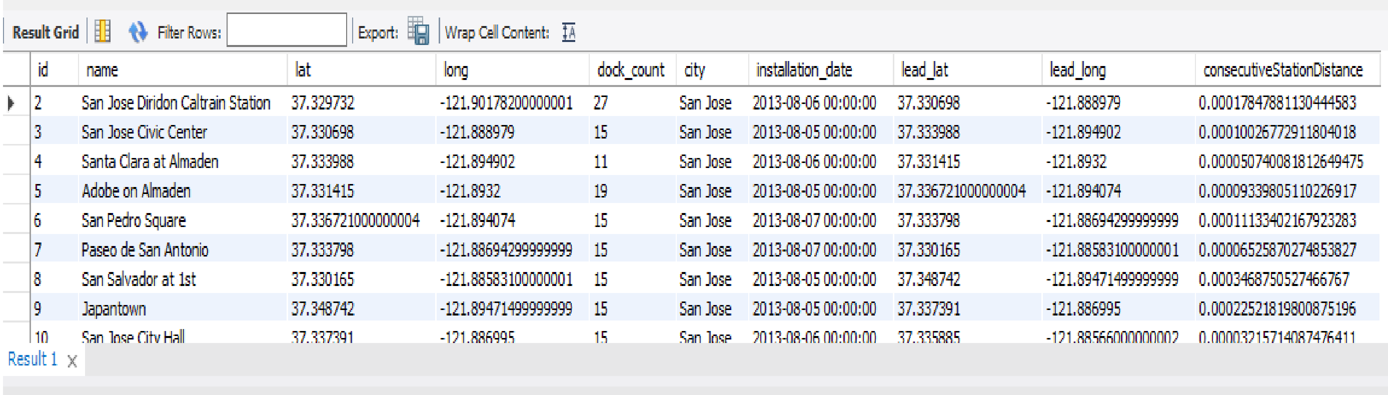
```
 /*

CONTEXT:

- Idle Time is the duration for which a station has more than 3 bikes available.

- This can be calculated by first partitioning the data according to station id
    and then ordering data by time.

- For all the rows below logic needs to be implemented:

            total_idle_time_station = 0

            if bikes_available > 3:

                                total_idle_time_station = total_idle_time_station +
  (next_row_time - current_row_time)

RESULT EXPECTATION:

- Idle Time for the stations

*/

 SELECT station_id,
 sum(TIMESTAMPDIFF(SECOND,st.time,st.lead_bikes_available_time)) AS
 idle_time

 -- Partition stations by station id so as to get data for station 1 then station 2

 -- LEAD would help us in getting the time for next row after partition is done

  FROM(

  SELECT *,
 LEAD(time) OVER(PARTITION BY station_id ORDER BY time) AS
lead_bikes_available_time

 FROM status

 ) AS st

 WHERE bikes_available > 3

 ORDER BY idle_time;
```



station_id	idle_time
2	1048020

3. In case two stations are nearby, it might be possible to shut one down. Need to find the distance between consecutive stations (between Stations 1 and 2, Stations 2 and 3, and so on)

```
/*

CONTEXT:

- Haversine distance is calculated for the consecutive stations

- To get data for consecutive station lead is used and then haversine distance is
calculated with lat,long of current row.

RESULT EXPECTATION:

- Consecutive Station haversine distance

*/

SELECT *,

acos(

cos(radians( st.lat ))

* cos(radians( st.lead_lat ))

* cos(radians( st.long ) - radians( st.lead_long ))

+ sin(radians( st.lat ))

* sin(radians( st.lead_lat ))

) AS consecutiveStationDistance FROM (SELECT *,

LEAD(station.lat) OVER(ORDER BY station.id) AS lead_lat,

LEAD(station.long) OVER(ORDER BY station.id ) AS lead_long
FROM station
) AS st;
```



id	name	lat	long	dock_count	city	installation_date	lead_lat	lead_long	consecutiveStationDistance
2	San Jose Diridon Caltrain Station	37.329732	-121.90178200000001	27	San Jose	2013-08-06 00:00:00	37.330698	-121.888979	0.00017847881130444583
3	San Jose Civic Center	37.330698	-121.888979	15	San Jose	2013-08-05 00:00:00	37.333988	-121.894902	0.00010026772911804018
4	Santa Clara at Almaden	37.333988	-121.894902	11	San Jose	2013-08-06 00:00:00	37.331415	-121.8932	0.000050740081812649475
5	Adobe on Almaden	37.331415	-121.8932	19	San Jose	2013-08-05 00:00:00	37.336721000000004	-121.894074	0.0000933980511022691 7
6	San Pedro Square	37.336721000000004	-121.894074	15	San Jose	2013-08-07 00:00:00	37.333798	-121.88694299999999	0.00011133402167923283
7	Paseo de San Antonio	37.333798	-121.88694299999999	15	San Jose	2013-08-07 00:00:00	37.330165	-121.88583100000001	0.00006525870274853827
8	San Salvador at 1st	37.330165	-121.88583100000001	15	San Jose	2013-08-05 00:00:00	37.348742	-121.89471499999999	0.0003468750527466767
9	Japantown	37.348742	-121.89471499999999	15	San Jose	2013-08-05 00:00:00	37.337391	-121.886995	0.00022521819800875196
10	San Jose City Hall	37.337391	-121.886995	15	San Jose	2013-08-06 00:00:00	37.335885	-121.88566000000002	0.00003215714087476411

4. Using the findings above to recommend three stations that can be shut. For example, if the Japantown and Ryland stations are nearby, and the Japantown is not as popular as the Ryland station, then it can be recommended to shut.

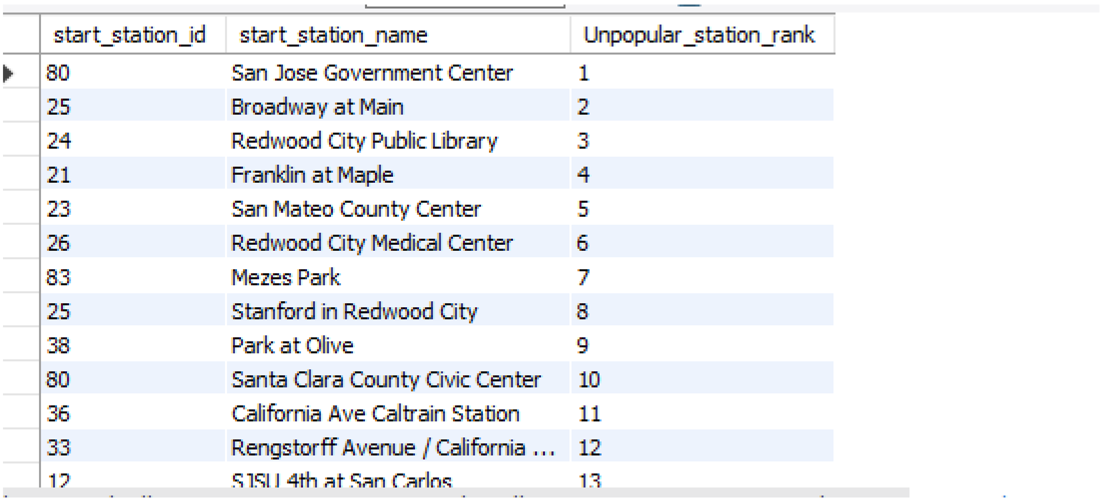
So, in order to solve which three stations to shut down following factors are taken:

- Stations which are least popular (having less frequency of trips).
- Idle time of all the stations.
- Consecutive distance between stations calculated using haversine distance.

i. Unpopular stations from the trip data are computed by taking into account the number of trips for each station and stored in view unpopular_stations.

```
/*
CONTEXT:
- Least popular stations would be those where few customers take the bikes from the
station.
- This can be calculated by counting the number of times a station name comes in start
station name column
- Rank is assigned for each station i.e station with rank 1 is least popular and so on.
RESULT EXPECTATION:
- A view is created which would have a table ranked according to popularity
*/
CREATE VIEW unpopular_stations AS
SELECT start_station_id,start_station_name,
RANK() OVER(ORDER BY COUNT(start_station_name)) AS Unpopular_station_rank
FROM trip
GROUP BY start_station_name;
```

```
SELECT * FROM unpopular_stations;
```



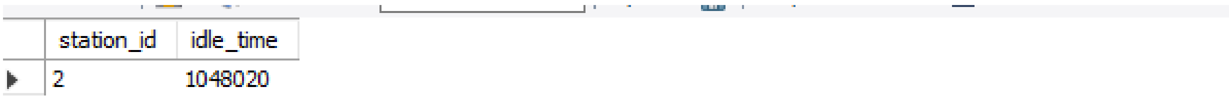
start_station_id	start_station_name	Unpopular_station_rank
80	San Jose Government Center	1
25	Broadway at Main	2
24	Redwood City Public Library	3
21	Franklin at Maple	4
23	San Mateo County Center	5
26	Redwood City Medical Center	6
83	Mezes Park	7
25	Stanford in Redwood City	8
38	Park at Olive	9
80	Santa Clara County Civic Center	10
36	California Ave Caltrain Station	11
33	Rengstorff Avenue / California ...	12
12	SJSU 4th at San Carlos	13

ii. Idle time for stations is calculated and stored in view idle_time_stations. On running the below query, only station 2 has idle time >0 and for all the other stations there is no entry is present in status table. So, considering for all other stations they have 0 or Null idle time.

Idle_time is calculated in seconds

```
/*

CONTEXT:

- Idle Time is the duration for which a station has more than 3 bikes available.

- This can be calculated by first partitioning the data according to station id and
then ordering data by time.

- For all the rows below logic needs to be implemented:

          total_idle_time_station = 0

          if bikes_available > 3:

                    total_idle_time_station = total_idle_time_station +
(next_row_time - current_row_time)

RESULT EXPECTATION:

- A view is created which would contain idle time for the stations

*/


CREATE VIEW  idle_time_stations AS

SELECT station_id,
sum(TIMESTAMPDIFF(SECOND,st.time,st.lead_bikes_available_time)) AS
idle_time
```

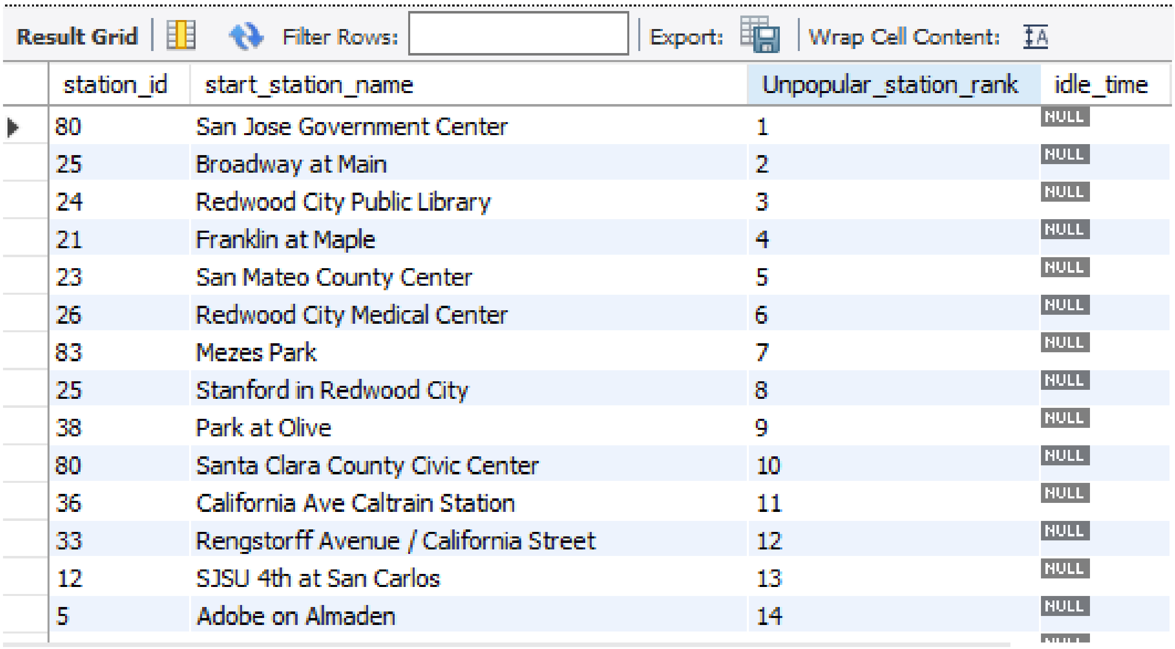
```
-- Partition stations by station id so as to get data for station 1 then station 2

-- LEAD would help us in getting the time for next row after partition is done

FROM(

SELECT *,

LEAD(time) OVER(PARTITION BY station_id ORDER BY time) AS
lead_bikes_available_time

FROM status

) AS st

WHERE bikes_available > 3

ORDER BY idle_time DESC;


SELECT * FROM idle_time_stations;
```



station_id	idle_time
2	1048020

iii. Left join of unpopular_stations and idle_time_stations views is computed to get a combined view for each station and store it in a view unpopular_idle_time_stations

```
/*
CONTEXT:
- Inner join of unpopular_stations and idle_time_stations to get a combined view.
RESULT EXPECTATION:
- A view is created which would contain idle time for the stations and unpopular
station rank.
*/
CREATE VIEW unpopular_idle_time_stations AS
SELECT start_station_id AS station_id, start_station_name, Unpopular_station_rank,
idle_time
FROM unpopular_stations
LEFT JOIN idle_time_stations
ON station_id = start_station_id;


SELECT * FROM unpopular_idle_time_stations;
```

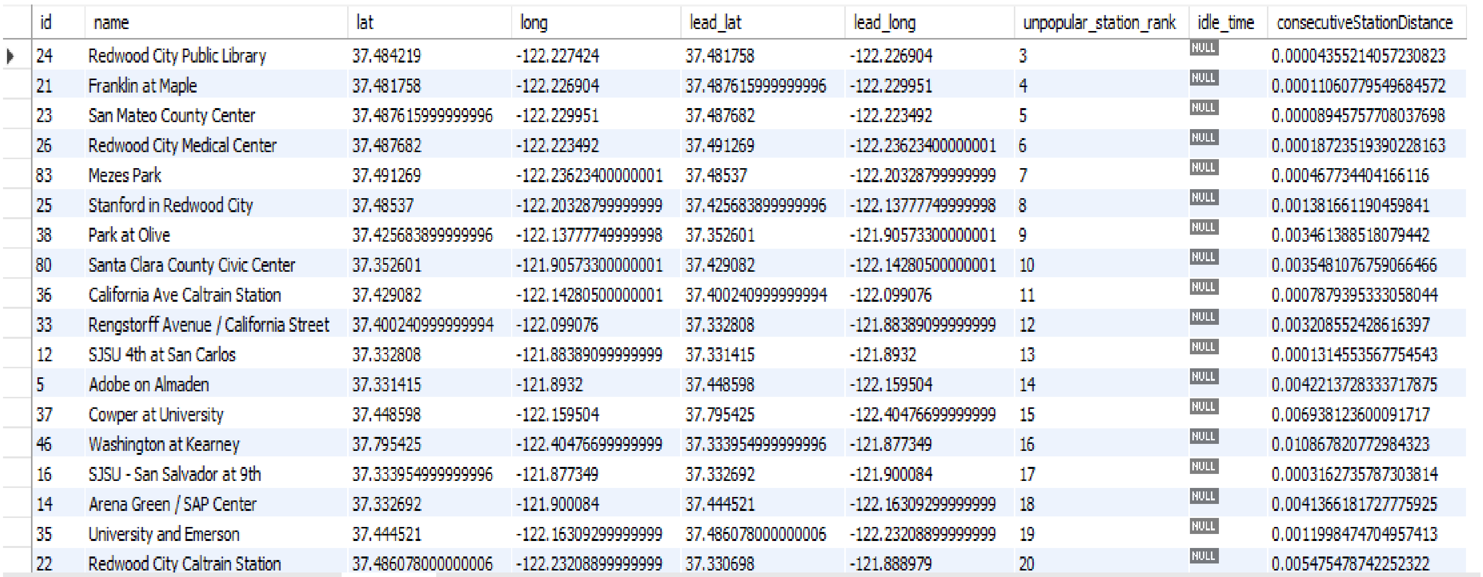


station_id	start_station_name	Unpopular_station_rank	idle_time
80	San Jose Government Center	1	NULL
25	Broadway at Main	2	NULL
24	Redwood City Public Library	3	NULL
21	Franklin at Maple	4	NULL
23	San Mateo County Center	5	NULL
26	Redwood City Medical Center	6	NULL
83	Mezes Park	7	NULL
25	Stanford in Redwood City	8	NULL
38	Park at Olive	9	NULL
80	Santa Clara County Civic Center	10	NULL
36	California Ave Caltrain Station	11	NULL
33	Rengstorff Avenue / California Street	12	NULL
12	SJSU 4th at San Carlos	13	NULL
5	Adobe on Almaden	14	NULL

iv. A subquery is further created to calculate the haversine distance for consecutive stations in unpopular_idle_time_stations view. This process ensures that unpopular stations, idle time and consecutive station distance factors are taken into account for considering which stations to shut down

/*
CONTEXT:
- Consecutive station haversine distance is computed for records in unpopular_idle_time_stations by taking lat,long data
from station table.
RESULT EXPECTATION:
- Consecutive Station haversine distance for unpopular_idle_time_stations
*/

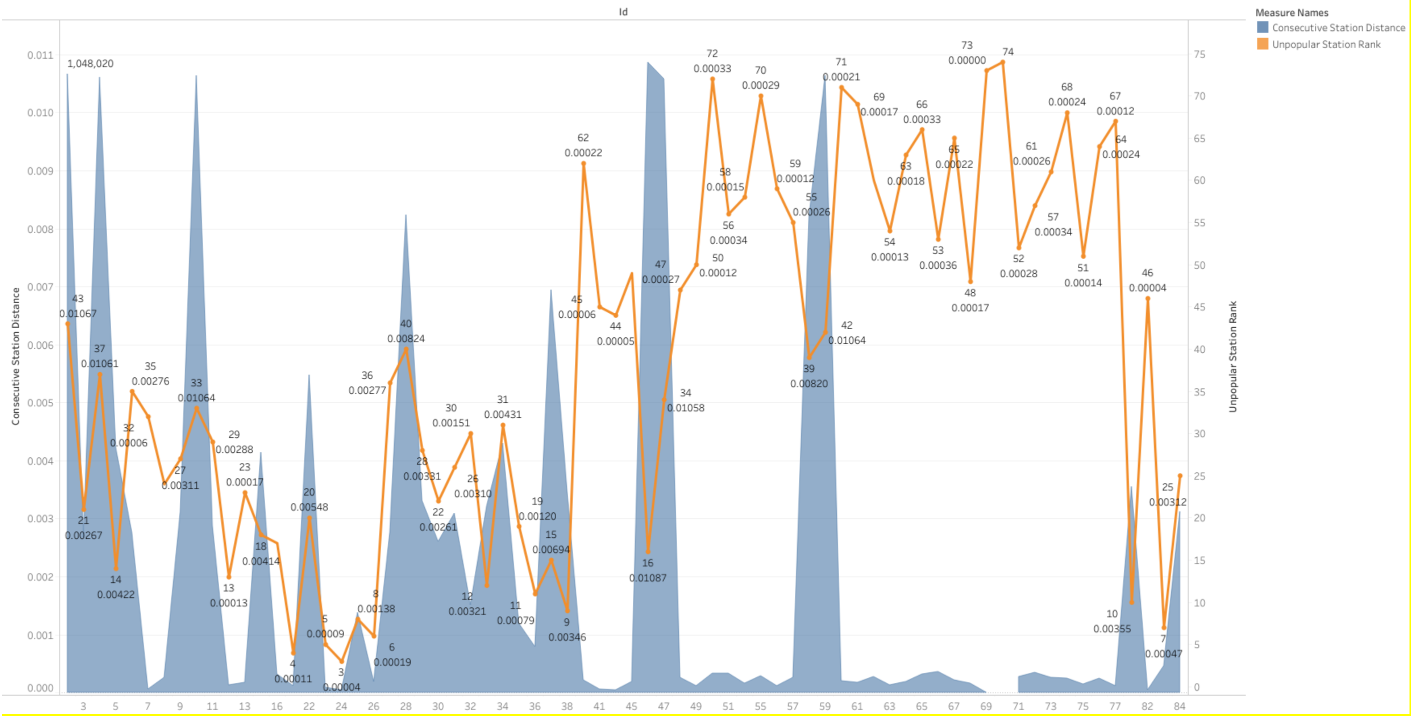
```
SELECT id,name,lat,st.long,lead_lat,lead_long,unpopular_station_rank,idle_time,
acos(
cos(radians( st.lat ))
* cos(radians( st.lead_lat ))
* cos(radians( st.long ) - radians( st.lead_long ))
+ sin(radians( st.lat ))
* sin(radians( st.lead_lat ))
) AS consecutiveStationDistance FROM (SELECT *,
LEAD(unp_st.lat) OVER() as lead_lat,
LEAD(unp_st.long) OVER( ) as lead_long
FROM
(SELECT * FROM station
INNER JOIN unpopular_idle_time_stations
ON station.name = unpopular_idle_time_stations.start_station_name
) AS unp_st
) AS st;
```



id	name	lat	long	lead_lat	lead_long	unpopular_station_rank	idle_time	consecutiveStationDistance
24	Redwood City Public Library	37.484219	-122.227424	37.481758	-122.226904	3	NULL	0.00004355214057230823
21	Franklin at Maple	37.481758	-122.226904	37.487615999999996	-122.229951	4	NULL	0.00011060779549684572
23	San Mateo County Center	37.487615999999996	-122.229951	37.487682	-122.223492	5	NULL	0.00008945757708037698
26	Redwood City Medical Center	37.487682	-122.223492	37.491269	-122.23623400000001	6	NULL	0.00018723519390228163
83	Mezes Park	37.491269	-122.23623400000001	37.48537	-122.20328799999999	7	NULL	0.000467734404166116
25	Stanford in Redwood City	37.48537	-122.20328799999999	37.425683899999996	-122.13777749999998	8	NULL	0.001381661190459841
38	Park at Olive	37.425683899999996	-122.13777749999998	37.352601	-121.90573300000001	9	NULL	0.003461388518079442
80	Santa Clara County Civic Center	37.352601	-121.90573300000001	37.429082	-122.14280500000001	10	NULL	0.0035481076759066466
36	California Ave Caltrain Station	37.429082	-122.14280500000001	37.400240999999994	-122.099076	11	NULL	0.000787939533058044
33	Rengstorff Avenue / California Street	37.400240999999994	-122.099076	37.332808	-121.88389099999999	12	NULL	0.003208552428616397
12	SJSU 4th at San Carlos	37.332808	-121.88389099999999	37.331415	-121.8932	13	NULL	0.0001314553567754543
5	Adobe on Almaden	37.331415	-121.8932	37.448598	-122.159504	14	NULL	0.0042213728333717875
37	Cowper at University	37.448598	-122.159504	37.795425	-122.40476699999999	15	NULL	0.006938123600091717
46	Washington at Kearney	37.795425	-122.40476699999999	37.333954999999996	-121.877349	16	NULL	0.010867820772984323
16	SJSU - San Salvador at 9th	37.333954999999996	-121.877349	37.332692	-121.900084	17	NULL	0.0003162735787303814
14	Arena Green / SAP Center	37.332692	-121.900084	37.444521	-122.16309299999999	18	NULL	0.0041366181727775925
35	University and Emerson	37.444521	-122.16309299999999	37.486078000000006	-122.23208899999999	19	NULL	0.0011998474704957413
22	Redwood City Caltrain Station	37.486078000000006	-122.23208899999999	37.330698	-121.888979	20	NULL	0.005475478742252322

- Result from the above query was exported into an csv file and was visualized using tableau to plot a curve of unpopular station rank vs consecutive station distance.
- Idle time is not in the plot as it is available only for station 2 which in fact comes in popular stations list.

Consecutive Station Distance vs Unpopular Station Rank



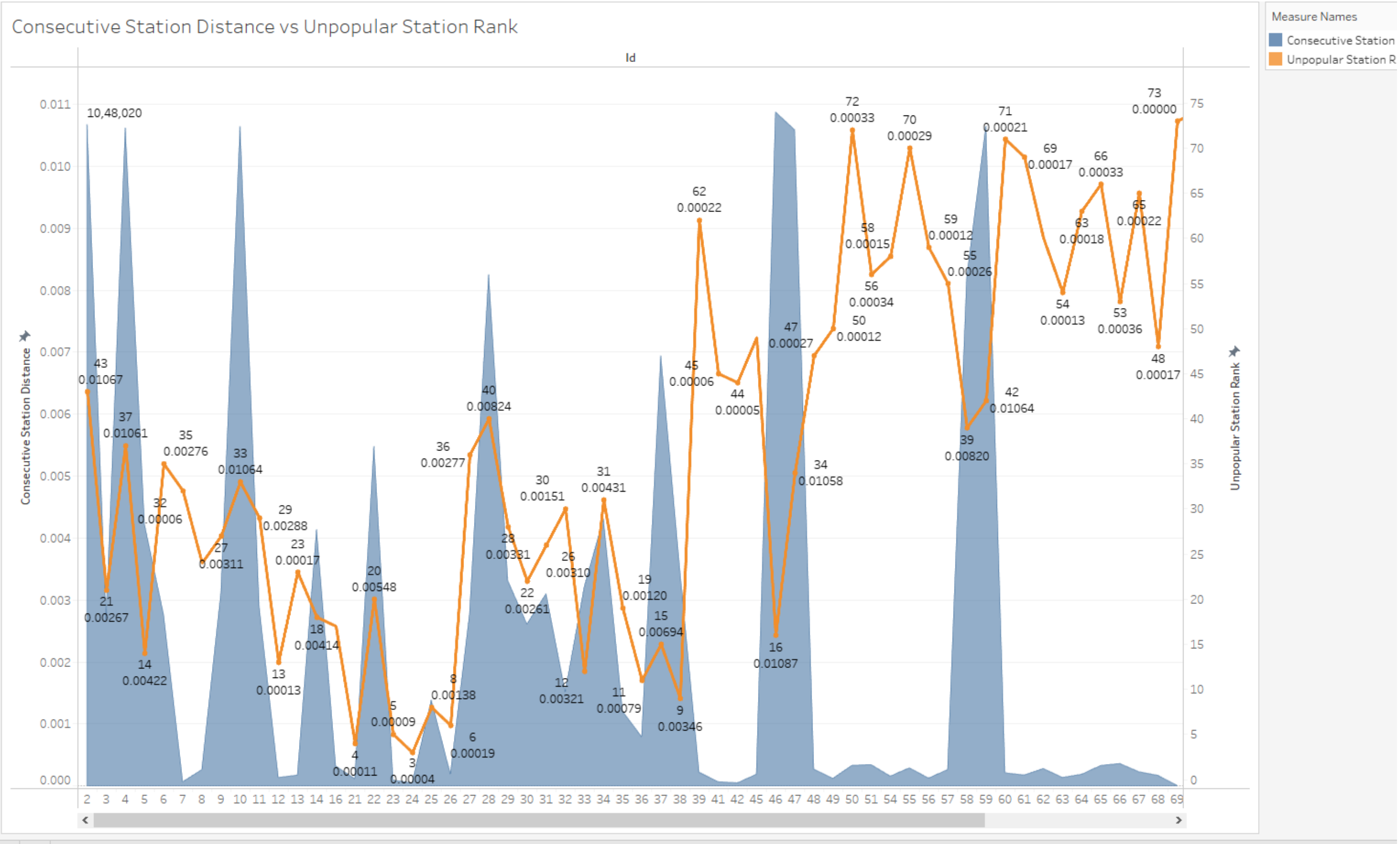


Tableau Public Link

https://public.tableau.com/app/profile/har.shobhit.dayal/viz/ConsecutiveStationDistancevsUnpopularStationRank/ConsecutiveStationDistancevsUnpopularStationRank

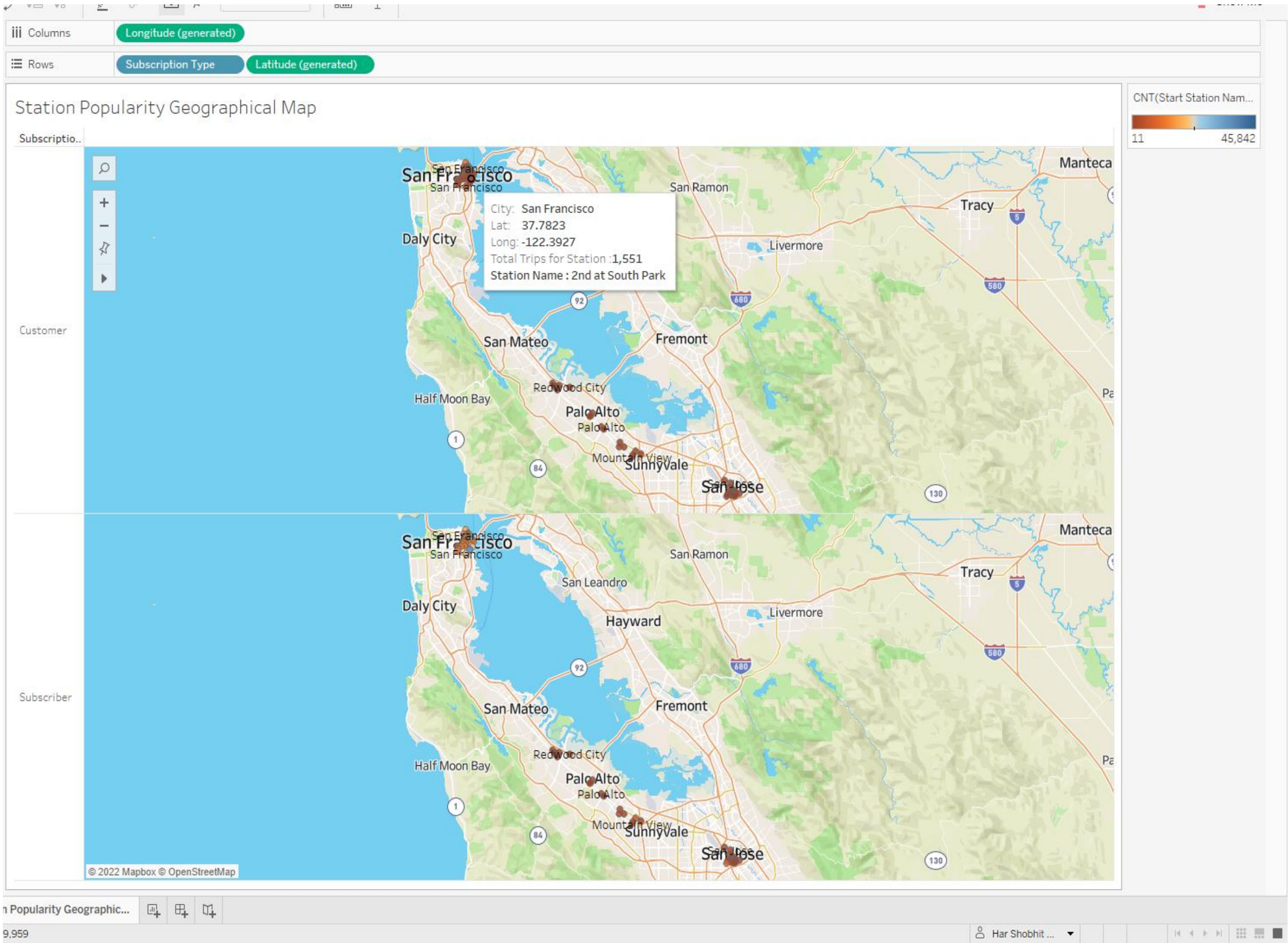
- From the above plots it can be observed that station 21,23,24 are some of the least popular stations and their consecutive station distance is also very less.
- So, as per the above analysis I would recommend to shut down station 21,23 and 24 respectively.

Optimizing Operations

1. Plotting the popularity of each station on a map for subscribers and customers

Tableau Public Link

https://public.tableau.com/app/profile/har.shobhit.dayal/viz/StationPopularityGeographicalMap/StationPopularityGeographicalMap?publish=yes



Columns: Longitude (generated)

Rows: Subscription Type, Latitude (generated)

Station Popularity Geographical Map

Subscriptio..

Customer

City: San Francisco
Lat: 37.7823
Long: -122.3927
Total Trips for Station :1,551
Station Name : 2nd at South Park

Subscriber

CNT(Start Station Nam...
11 45,842

© 2022 Mapbox © OpenStreetMap

n Popularity Geographic...

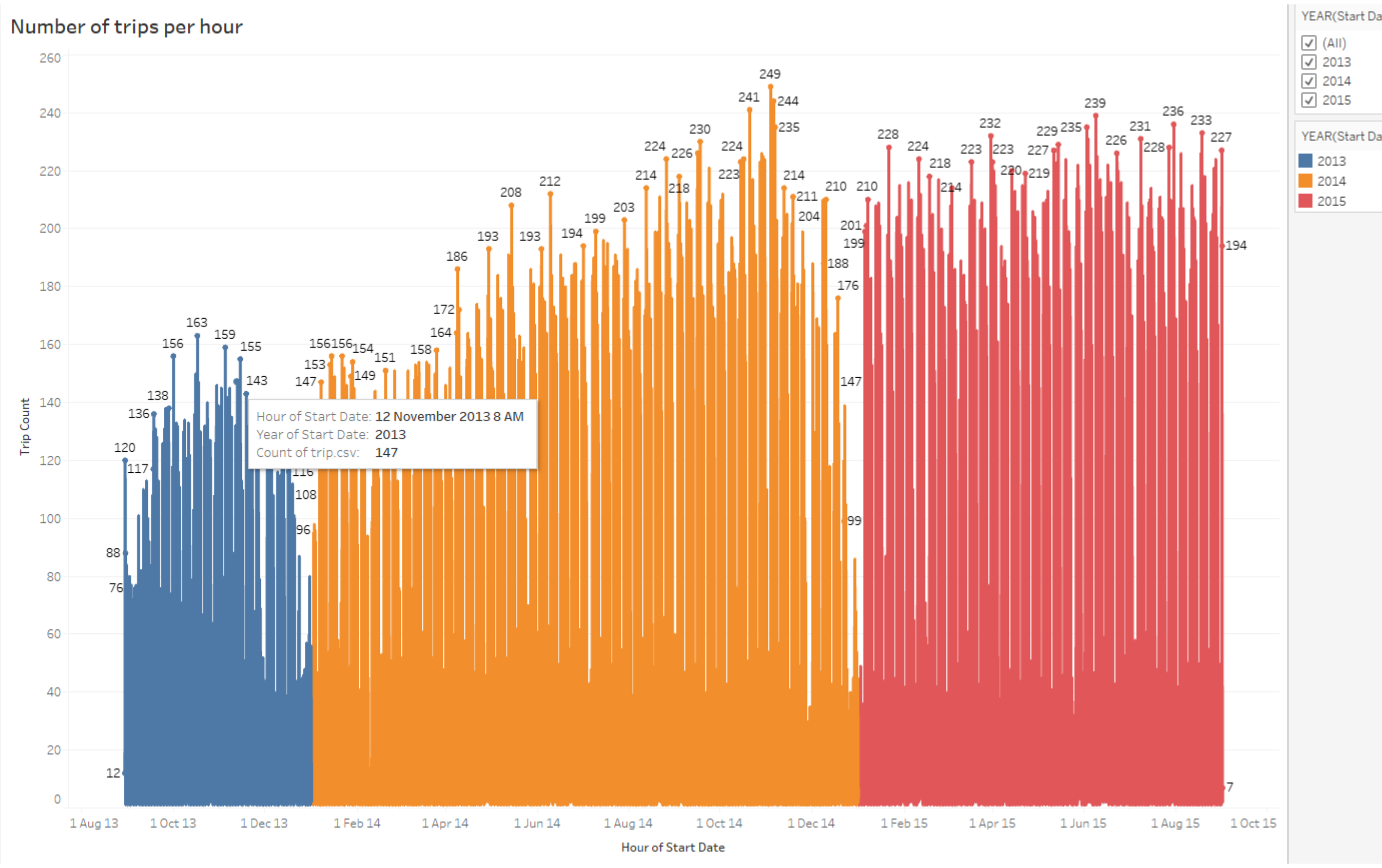
9,959

Har Shobhit ...

2. Plotting the number of trips per hour for all the data provided in the Trip table

Tableau Public Link

https://public.tableau.com/app/profile/har.shobhit.dayal/viz/Numberoftripsperhour_16546247039830/Numberoftripsperhour?publish=yes



Hour of Start Date: 12 November 2013 8 AM
Year of Start Date: 2013
Count of trip.csv: 147

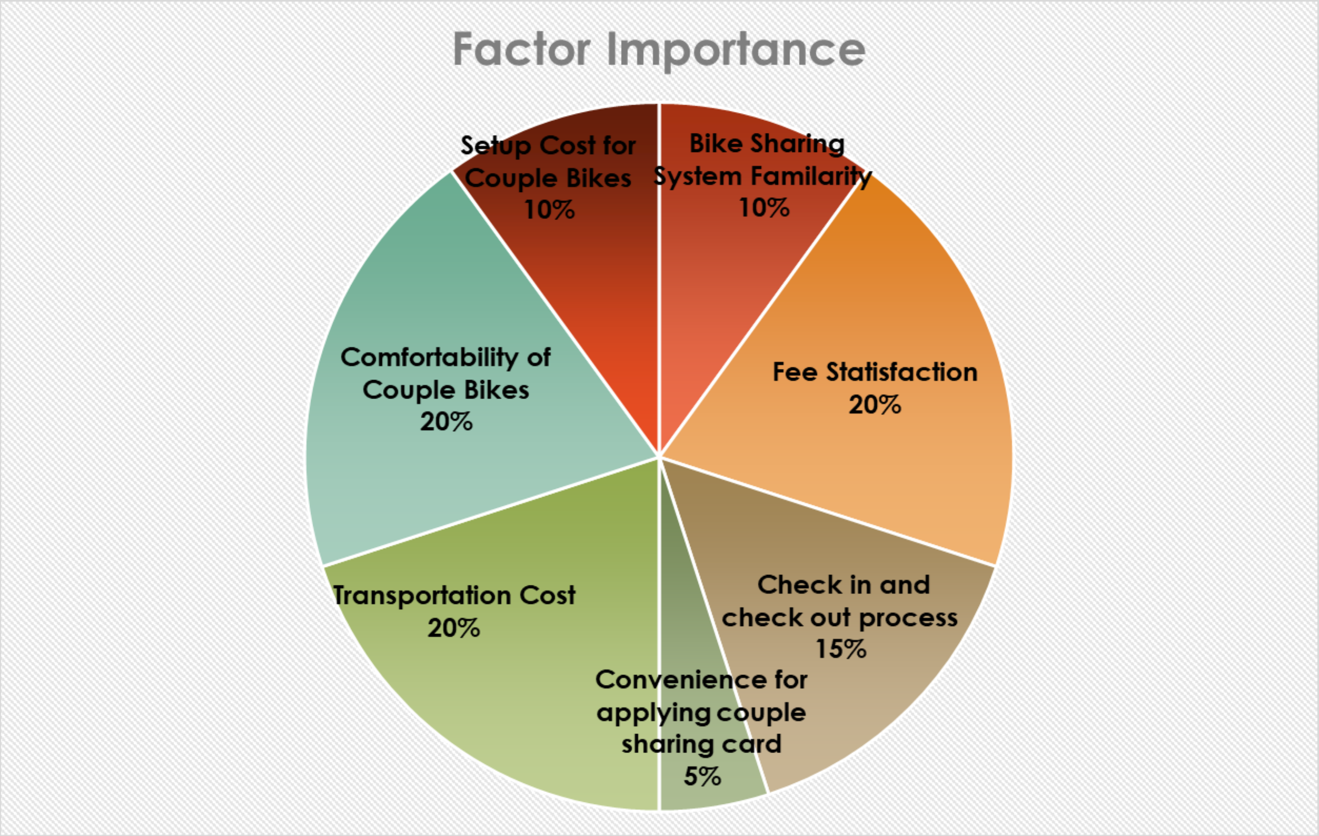
Hour of Start Date vs Number of Trips

3. Use the findings above to provide insights on how to optimize operations.

- In order to optimize operations, following factors are important:
  - Bike rebalancing policy from one station to other which would guarantee:
    - Every customer would find a bike at the origin station.
    - Customer would find a free parking spot at the final station.
- In order to meet the demand for bikes, rebalancing of bikes from less popular station to more popular station is required
  - In order to decide which stations are more popular than others popularity plot (Plotting the popularity of each station on a map for subscribers and customers) can be used and rank the stations according to their popularity.
    - For e.g. for station Id 70 it is one of the most popular stations so demand of bikes would be high here during the peak hours.
    - So, in order to meet the demand, bikes can be rebalanced by taking some bikes from less popular stations like 24 to where demand is not that high to station 70.
- From the plot (Plotting the number of trips per hour for all the data provided in the Trip table) it can be observed for which hour during the day maximum number of trips are there and for which minimum.
  - It was seen on average that peek time is between 7-10 am for most of the stations and between 3-6 pm in the evening.
  - This insight would help us to decide during which time of day bikes can be rebalanced from one station to other.

        - In order to rebalance the bikes a transportation system is required which would transfer the bikes to the required station.
        - The rebalancing of bikes should be done outside of peak hours window (7-10 am, 3-6 pm) so that bikes are available for customers when rush is there at the stations.
- In order to approach the effective solution for rebalancing the bikes some terms would be used as:
    - Demand gap -> Difference of bikes taken from station (initial bikes at start of day) - Number of bikes returned to station at end of day.
        - For eg for a day station 1 to station 2 18 bikes were rented and 12 bikes were used from station 2 to station 1. Then demand gap for station 1 is 18-12 = +6 and for station 2 demand gap is 12-18 = -6 i.e it means station 2 needs to have at least 6 docs available to meet customer demand.
    - Problem station -> When the number of absolute (bikes available + demand gap) goes above the station's dock capacity
    - There are 2 sub categories of problem station:
        - Loading station -> Station which does not have enough bikes to fulfil the customer demands.
        - Unloading station -> Station which does not have enough docks to store the bikes when customer return the bikes at their final station.
    - Normal Station-> When absolute (number of bikes available + demand gap) is within the station dock's capacity it is called a normal station where rebalancing is not required.
- Different stations can be classified as problem or normal station by calculating the demand gap and average bikes available (Calculating the average number of bikes and docks available for Station 2., dock count available from the historical data available in status table and trip table.
- An approach as below for rebalancing can be followed:
    - Transfer the bikes from unloading to loading station so as to solve the problem for one problem station then move on to another next loading station by transferring bikes from an unloading station till all customer demands are met.
    - Here the popularity of station would be also an important factor that can be used to decide from which loading or unloading stations bikes have to be picked up and dropped to.
    - The bikes need to be transported from one station to other before the peak hours are there which can be calculated from the trips per hour analysis that was done earlier.
        - This approach will help of making sure the customer demands are met for loading stations so that bikes are available.
        - This would also help to ensure the docs are available at each station so that customer can drop the bike and not place anywhere else.

Zulip has decided to start a new product line called Couple Bikes. This will enable two persons to travel from one station to another at the same time. What can be some of the factors that can be considered while validating the idea of couple bikes?



- Factors to consider are:
    - Familiarity of Bike Sharing System
        - The above factor talks about how much people are aware of the bike sharing system.
        - If people are not aware that such a system exists then the chance of people using this would be very low.
    - Satisfaction with Fee for Couple Bikes
        - Price is an important factor for deciding to bring couple bikes or not.
        - It would be of utmost importance to see if the customers are satisfied with the fee charged for the couple bikes.
    - Check in and check out process
        - People would prefer couple bikes if check in and check out process from one station to other is easy to use and understand.
    - Convenience for applying couple sharing card
        - The above factor would indicate how smooth is the process for couples to get their couple card issued.
    - Transportation Cost for Couple Bikes

    - As it was observed in optimizing operations in order to meet the customer demands bikes need to be transported from one station to other.
    - Cost for transporting these couple bikes also play a major role in deciding whether couple bikes should be introduced or not.
  - Comfortability of Couple Bikes
    - Customers should feel comfortable while using the couple bikes so that they use it more frequently.
  - Setup Cost for Couple Bikes
    - A cost evaluation needs to be performed on how much resources, cost and time would be required for producing couple bikes.