Analysis of San Francisco Bay Area Bike Share

Project 01

Database Foundations for Business Analytics

Professor Farzad Kamalzadeh

BUAN 6320

Analysis Done By

- Hajra Katlia (Hxk190028)
- Kushagra Rastogi (Kxr220031)
- Rana Udayveer Singh (RXX210005)
- Siddhesh Madhukar Koli (Sxk220091)

Dataset

We conduct analysis on the San Francisco Bay Area Bike Share dataset. The dataset weighs in at 2 GB. The dataset provides information on the reasonably priced, quick, and simple bike excursions in and around the San Francisco Bay Area. It includes details on the designated stations for picking up and dropping off bikes, the number of bikes and docks that are available, individual bike trips, and the weather forecast for a certain day for a given zip code.

Business Understanding

Data from the Bay Area Bike Share System is included in the collection. The San Francisco Municipal Transportation Agency (SFMTA) launched the Bay Area Bike Share System in 2013 to provide the general public with accessible and reasonably priced transportation. SFMTA gathered information on the trips made by users from various places in order to gain a better understanding of how the Bay Area Bike Share System is used.

In addition to understanding commute patterns in the San Francisco area, we are using this data to assess other Bay Area Bike Share system aspects. The average trip duration, number of bikes available at a specific location and time, average temperature, precipitation, humidity, and other weather-related variables on each day of the trip, as well as the latitude and longitude of the station locations, can all be determined using this data. It can also be used to track bike patterns depending on the day of the week and time.

After the study, different conclusions might be taken from this dataset. The Bay Area Bike Share system's effectiveness can be increased by determining the ideal time, place, and weather condition factors. For instance, if the weather is favorable and the peak usage hours have been recorded, the number of bikes provided to a certain station may be increased. It is possible to completely remove the bike-share program from the system if it is not being utilized as frequently at a specific station.

In order to increase the availability of bikes and docks at various stations, this analysis will look at popular routes, how the weather affects the use of bike-share services and demand patterns for excursions that may be made on two wheels throughout the day and week. The average travel time can be shortened, the number of bikes at busy stations can be increased, and the optimal temperature and time can be checked to modify the supply chain of bikes accordingly. These are all things that can be maximized after analyzing this data.

We intend to obtain comprehensive data regarding the ratio of subscribers to customers at a specific station at the conclusion of the analysis. We will investigate the best area to advertise in order to turn customers into subscribers. It would be necessary to increase the number of bikes and docks available at stations with strong demand. Which season of the year has the lowest demand and is best for carrying out maintenance work?

Understanding Data

Data from the San Francisco Bay Area Bike Share includes tables with information on the weather, station name and coordinates, trip, and trip status.

Station

Column Name	Datatype	Description
id	INT	Station id
name	VARCHAR	Name of the station
latitude	DOUBLE	Latitude of station
longitude	DOUBLE	Longitude of station
dock_count	INT	Number of docks available
city	VARCHAR	Station city
installation_date	DATETIME	Station installation date

Station id (id) is the primary key in the station table

Status

Column Name	Datatype	Description
station_id	INT	Unique station id
bikes_available	INT	Number of bikes available
docks_available	INT	Number of docs available
time	DATETIME	Time of the trip

Time of the trip (time) is the primary key in the status table

Trip

Column Name	Datatype	Description
id	INT	Station id
duration	DOUBLE	Duration of the trip
start_date	DATETIME	Start date of the trip
start_station_name	VARCHAR	Strt station name
start_station_id	INT	Station id where the trip starts
end_date	DATETIME	Date when the trip ends
end_station_id	INT	Station id where the trip ends
bike_id	INT	Bike id of the trip
subscription_type	VARCHAR	Subscription type of the trip
zip_code	VARCHAR	Zip codes

Station id (id) is the primary key in the trip table

Weather

Column Name	Datatype	Description
date	DATETIME	Date of the trip
min_temperature_f	INT	Minimum temperature on the day of the trip
max_temperature_f	INT	Maximum temperature on the day of the trip
min_humidity	INT	Minimum humidity on the day of the trip
max_humidity	INT	Maximum humidity on the day of the trip
min_sea_level_pressure	INT	Minimum sea level pressure on the day of the trip

max_sea_level_pressure	INT	Maximum sea level pressure on the day of the trip
min_visibility_miles	INT	Minimum visibility on the day of the trip
max_visibility_miles	INT	Maximum visibility on the day of the trip
min_wind_speed_mph	INT	Minimum wind speed on the day of the trip
precipitation_inches	INT	Precipitation inches
cloud_cover	INT	Cloud cover of the day
events	VARCHAR	Events on the day of the trip
wind_dr_degrees	INT	Wind in degrees on trip day
zip_code	INT	Zip code of the location

Date of the trip (date) is the primary key in the weather table

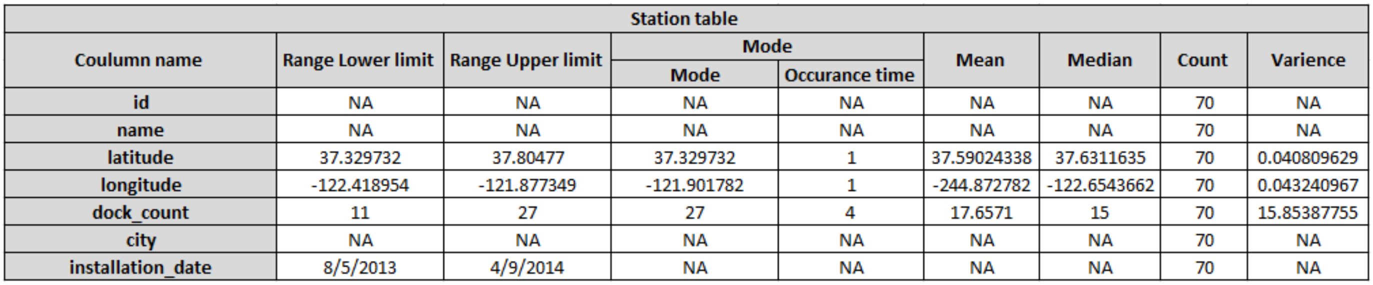
The date column in the dataset connects the trip table and the weather table. Through the start station id, the travel, status, and station are connected. The station ID serves as a link between a station and a station's coordinates. Therefore, they all are dependent on each other.

No column names were altered because they all seemed understandable and straightforward. There were no duplicate values found in either of the tables. However, there were some missing values within the tables.

- None of the columns in the station table had any missing values.
- None of the status table columns have any missing values.
- There are no empty columns in the trip table.
- The columns in the weather table has various missing value. Columns with the missing values are mentioned below:

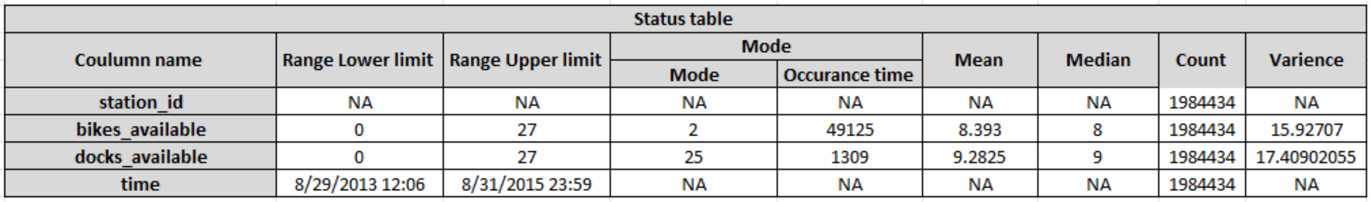
Weather Table	
Column Name	**Missing Values**
max_temperature_f	4
min_temperature_f	4
max_humidity	4
min_humidity	4
max_sea_level_pressure	4
min_sea_level_pressure	1
max_visibility_miles	13
min_visibility_miles	13
max_wind_speed_mph	1
precipitation_inches	1
wind_dr_degrees	1

Statistics of the data



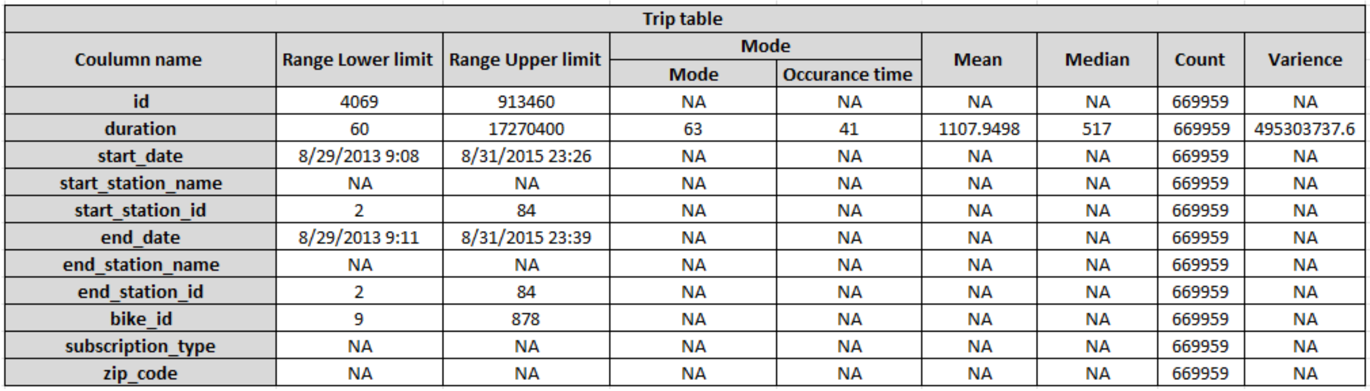
Station table

Coulumn name	Range Lower limit	Range Upper limit	Mode		Mean	Median	Count	Varience
			Mode	Occurance time				
id	NA	NA	NA	NA	NA	NA	70	NA
name	NA	NA	NA	NA	NA	NA	70	NA
latitude	37.329732	37.80477	37.329732	1	37.59024338	37.6311635	70	0.040809629
longitude	-122.418954	-121.877349	-121.901782	1	-244.872782	-122.6543662	70	0.043240967
dock_count	11	27	27	4	17.6571	15	70	15.85387755
city	NA	NA	NA	NA	NA	NA	70	NA
installation_date	8/5/2013	4/9/2014	NA	NA	NA	NA	70	NA



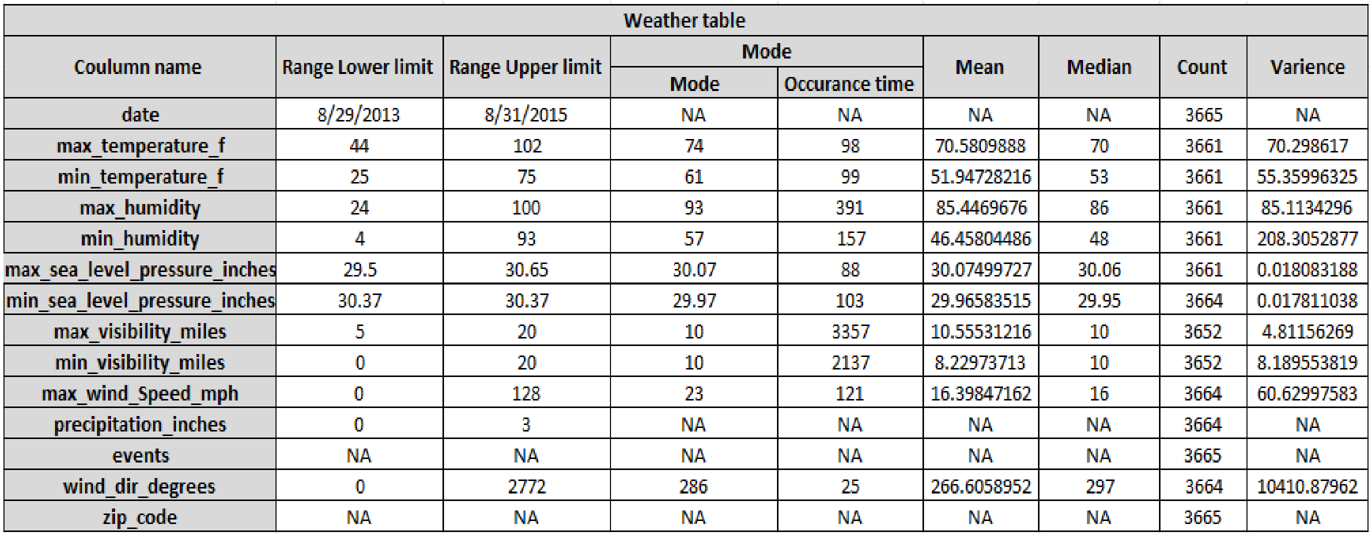
Status table

Coulumn name	Range Lower limit	Range Upper limit	Mode		Mean	Median	Count	Varience
			Mode	Occurance time				
station_id	NA	NA	NA	NA	NA	NA	1984434	NA
bikes_available	0	27	2	49125	8.393	8	1984434	15.92707
docks_available	0	27	25	1309	9.2825	9	1984434	17.40902055
time	8/29/2013 12:06	8/31/2015 23:59	NA	NA	NA	NA	1984434	NA



Trip table

Coulumn name	Range Lower limit	Range Upper limit	Mode		Mean	Median	Count	Varience
			Mode	Occurance time				
id	4069	913460	NA	NA	NA	NA	669959	NA
duration	60	17270400	63	41	1107.9498	517	669959	495303737.6
start_date	8/29/2013 9:08	8/31/2015 23:26	NA	NA	NA	NA	669959	NA
start_station_name	NA	NA	NA	NA	NA	NA	669959	NA
start_station_id	2	84	NA	NA	NA	NA	669959	NA
end_date	8/29/2013 9:11	8/31/2015 23:39	NA	NA	NA	NA	669959	NA
end_station_name	NA	NA	NA	NA	NA	NA	669959	NA
end_station_id	2	84	NA	NA	NA	NA	669959	NA
bike_id	9	878	NA	NA	NA	NA	669959	NA
subscription_type	NA	NA	NA	NA	NA	NA	669959	NA
zip_code	NA	NA	NA	NA	NA	NA	669959	NA

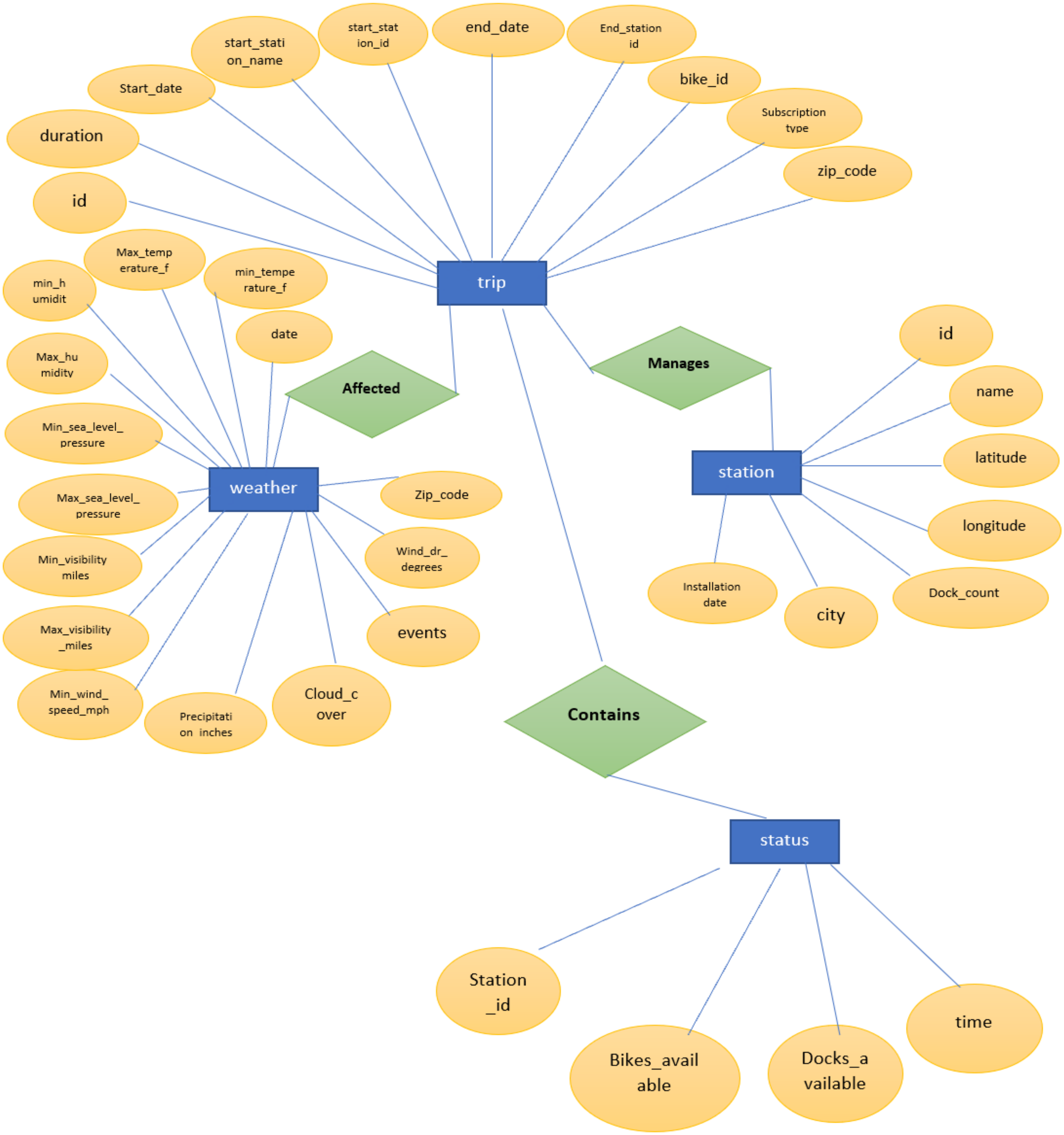


Weather table

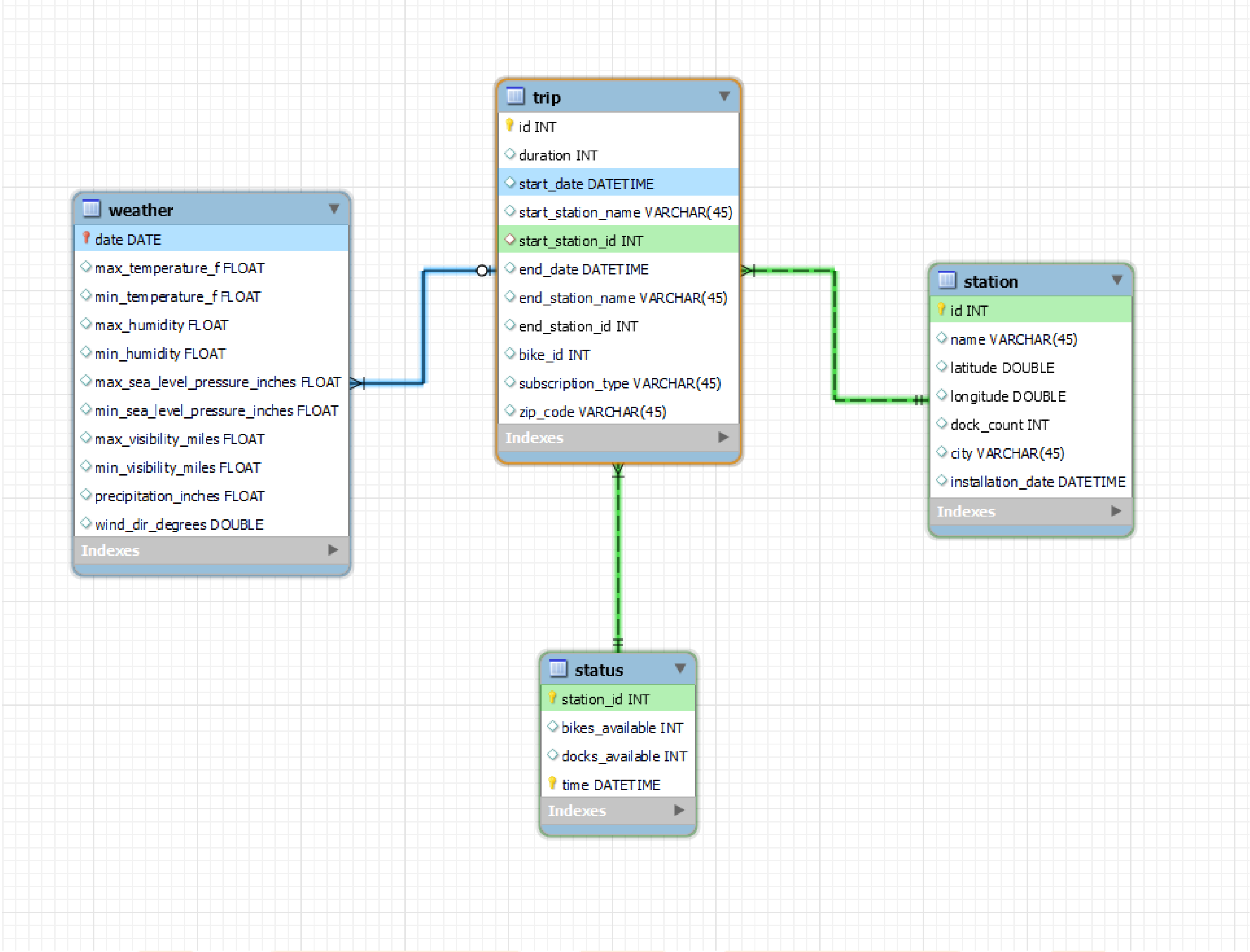
Coulumn name	Range Lower limit	Range Upper limit	Mode		Mean	Median	Count	Varience
			Mode	Occurance time				
date	8/29/2013	8/31/2015	NA	NA	NA	NA	3665	NA
max_temperature_f	44	102	74	98	70.5809888	70	3661	70.298617
min_temperature_f	25	75	61	99	51.94728216	53	3661	55.35996325
max_humidity	24	100	93	391	85.4469676	86	3661	85.1134296
min_humidity	4	93	57	157	46.45804486	48	3661	208.3052877
max_sea_level_pressure_inches	29.5	30.65	30.07	88	30.07499727	30.06	3661	0.018083188
min_sea_level_pressure_inches	30.37	30.37	29.97	103	29.96583515	29.95	3664	0.017811038
max_visibility_miles	5	20	10	3357	10.55531216	10	3652	4.81156269
min_visibility_miles	0	20	10	2137	8.22973713	10	3652	8.189553819
max_wind_Speed_mph	0	128	23	121	16.39847162	16	3664	60.62997583
precipitation_inches	0	3	NA	NA	NA	NA	3664	NA
events	NA	NA	NA	NA	NA	NA	3665	NA
wind_dir_degrees	0	2772	286	25	266.6058952	297	3664	10410.87962
zip_code	NA	NA	NA	NA	NA	NA	3665	NA

Database Design

The following schema design states the entities within the bike sharing data



Schema design Before Normalization



The table below shows the Primary key and Foreign key of the entities:

Table	Primary Key	Foreign Key
Station	id	NA
Status	(station_id, time)	station_id
Trip	id	start_station_id
Weather	date	NA

Functional Dependencies

Station Table

{id} ⟶ {Name, Dock Count} ✓

{id} ⟶ {Name, City, Longitude, Latitude} ⟶ **Violates BCNF (X)** ⟶ Decomposes into {id, Name}, {id, City, longitude, Latitude} ✓

Status Table

{station_id} ⟶ {Name, Dock Count} ✓

Trip Table

{id} ⟶ {start_station_id, end_station_id, start_station_name, end_station_name} ⟶ **Violates BCNF (X)** ⟶ Decomposes into {id, start_station_id, end_station_id}, {station_id, station_name} ✓

Weather Table

There is no functional dependency as every column in the weather table is calculated independently.

Normalizing Tables

After examining the first schema, we concluded that the tables still have some redundancy when viewed collectively. As a result, we chose to divide the tables, make new tables, and keep the information about the station's name and city in a different table that can also be utilized by trips.

Initial Functional Dependencies

{Station_Id} ⟶ {Name, Dock_count}

{Station_Id} ⟶ {bikes_available, docks_available}

{Trip_Id} ⟶ {start_station_id, end_station_id}

{Start_station_Id} ⟶ {city, longitude, latitude}

⟶

{Station_Id} ⟶ {Name, Dock_count, bikes_available, docks_available}

{Trip_Id} ⟶ {start_station_id, end_station_id}

{Start_station_Id} ⟶ {start_station_id (station_id), city, longitude, latitude}

Checking for any given FDs, BCNF Conditions

{Station_Id} → {Name, Dock_count} →	Station_id, Name and Dock_count are in the same table, and station_id is the key ✓
{Station_id} → {bikes_available, docks_available} →	Station_id, bikes_available and docks_available are in the same table, and station_id is the key ✓
{Trip_id} → {start_station_id, end_station} →	Trip_id, start_station_id, end_station_id are in the same table, and trip id is the key ✓
{start_station_id} → {city, longitude, latitude}	Start_station_id, city, longitude, and latitude are in the same table, and id is the key ✓
{station_id} → {Name, Dock_count, bikes_available, docks_available}	They aren't in the same table ✓

We can conclude that the supplied schema is in BCNF based on the conditions that were tested above.

Modified Schema Design

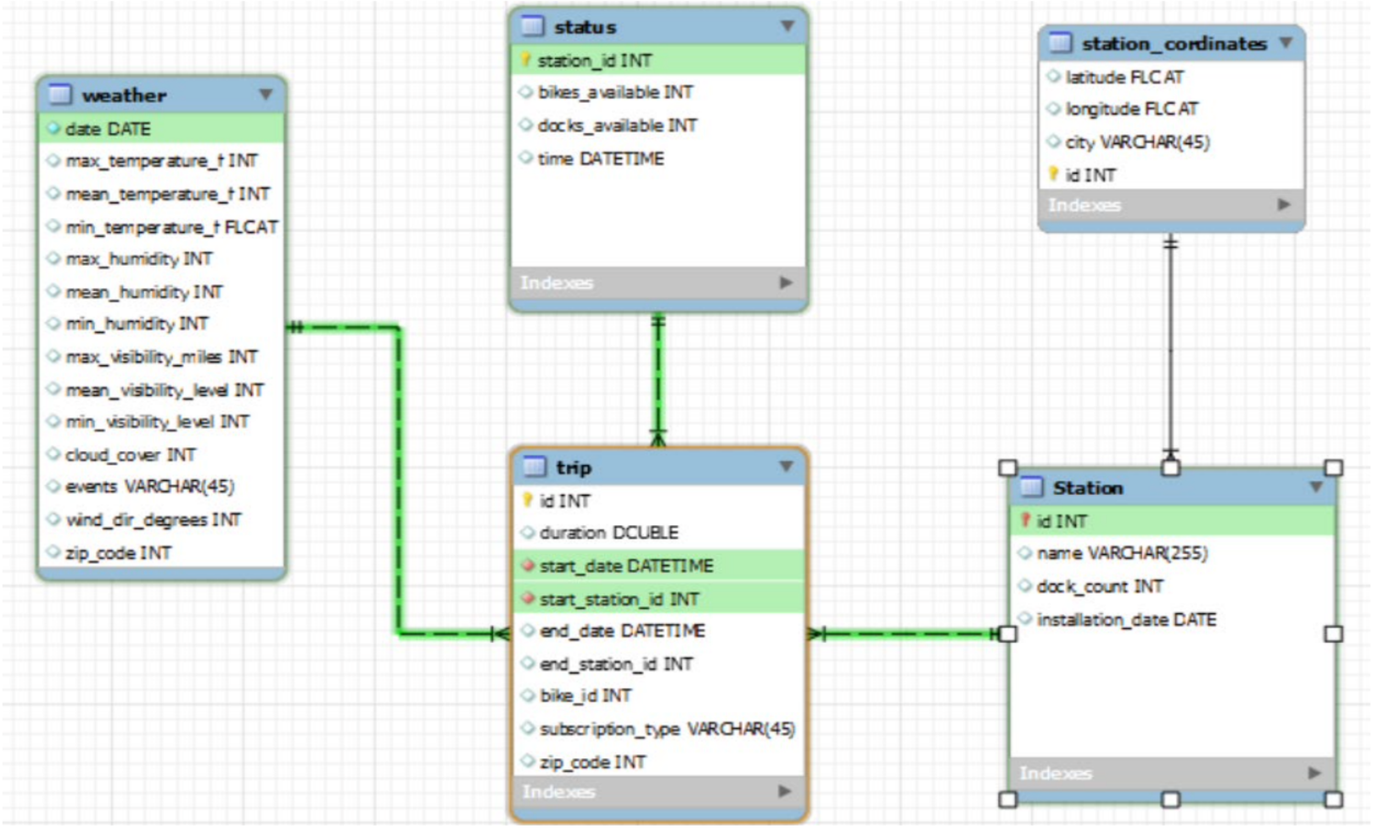
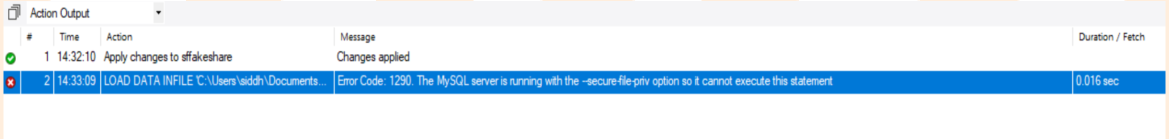


Table	Primary Key	Foreign Key
Trip	id	NA
Station	NA	start_station_id
Station_coordinates	id	Id (from station table)
Status	station_id	start_station_id
Weather	NA	start_date

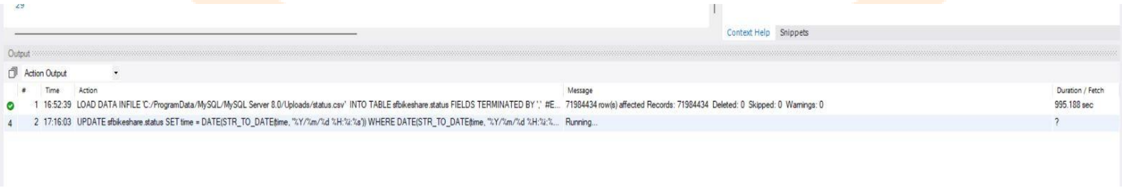
Data Cleaning

- **Importing Data**
  SQL displayed a security error stating that the current files were not located in a secure location when the file was being uploaded. We upload the file in a secure location first, then upload it in SQL to correct the problem. This fixed the issue, allowing us to upload the file in SQL.

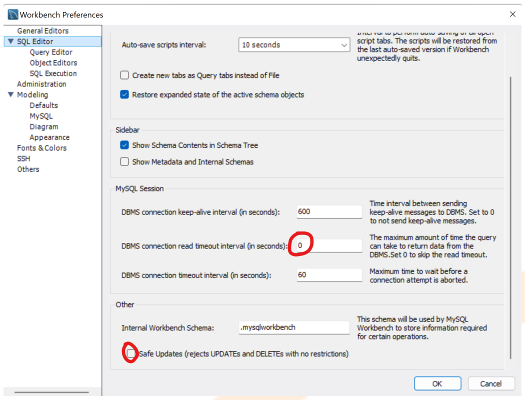


```
Action Output
#  Time      Action                                        Message                                                                                    Duration / Fetch
1  14:32:10  Apply changes to sffakeshare                  Changes applied
2  14:33:09  LOAD DATA INFILE 'C:\Users\siddh\Documents...  Error Code: 1290. The MySQL server is running with the --secure-file-priv option so it cannot execute this statement   0.016 sec
```

- **Formatting Issues**
  When importing the data, an error occurred because the date format was text and not YYYY-MM-DD. By altering the date format, this was resolved.
- **Query Running Time**
  Each inquiry had a readout time interval of just 30 seconds. However, each of our inquiries took 1300 seconds. As a result, we kept experiencing errors. We navigated to edit → preferences → SQL editor → MySQL session to fix the problem. DBMS timeout interval was set to 0 by command. We were able to perform our 1300-second query with ease because it didn't stop after 30 seconds.



```
Action Output
#  Time      Action                                                                                                          Message                                                                           Duration / Fetch
1  16:52:39  LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/status.csv' INTO TABLE sfbikeshare.status FIELDS TERMINATED BY ',' #E...  71984434 row(s) affected Records: 71984434  Deleted: 0  Skipped: 0  Warnings: 0   995.188 sec
2  17:16:03  UPDATE sfbikeshare.status SET time = DATE(STR_TO_DATE(time, '%Y/%m/%d %H:%i:%s')) WHERE DATE(STR_TO_DATE(time, '%Y/%m/%d %H:%i:%...  Running...                                                                        ?
```



Workbench Preferences

- General Editors
- ▼ SQL Editor
  - Query Editor
  - Object Editors
  - SQL Execution
- Administration
- ▼ Modeling
  - Defaults
  - MySQL
  - Diagram
  - Appearance
- Fonts & Colors
- SSH
- Others

Auto-save scripts interval: 10 seconds — Interval to perform auto-saving of all open script tabs. The scripts will be restored from the last auto-saved version if Workbench unexpectedly quits.

□ Create new tabs as Query tabs instead of File

☑ Restore expanded state of the active schema objects

Sidebar

☑ Show Schema Contents in Schema Tree

□ Show Metadata and Internal Schemas

MySQL Session

DBMS connection keep-alive interval (in seconds): 600 — Time interval between sending keep-alive messages to DBMS. Set to 0 to not send keep-alive messages.

DBMS connection read timeout interval (in seconds): 0 — The maximum amount of time the query can take to return data from the DBMS.Set 0 to skip the read timeout.

DBMS connection timeout interval (in seconds): 60 — Maximum time to wait before a connection attempt is aborted.

Other

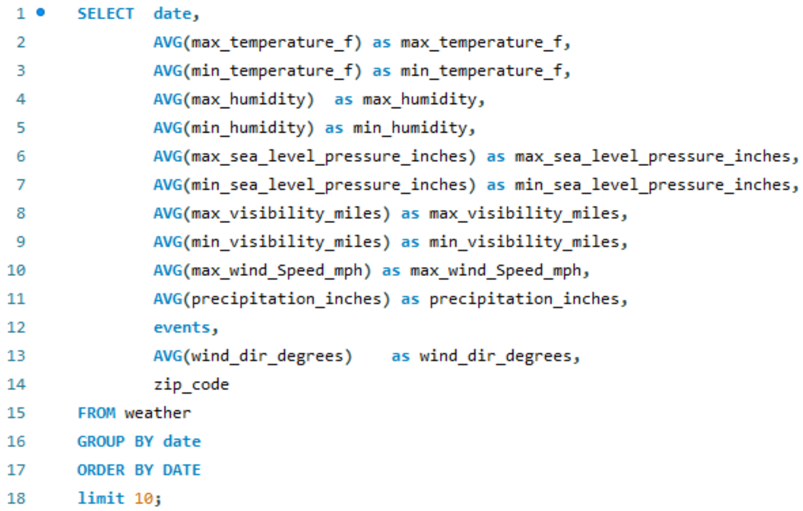
Internal Workbench Schema: .mysqlworkbench — This schema will be used by MySQL Workbench to store information required for certain operations.

□ Safe Updates (rejects UPDATEs and DELETEs with no restrictions)

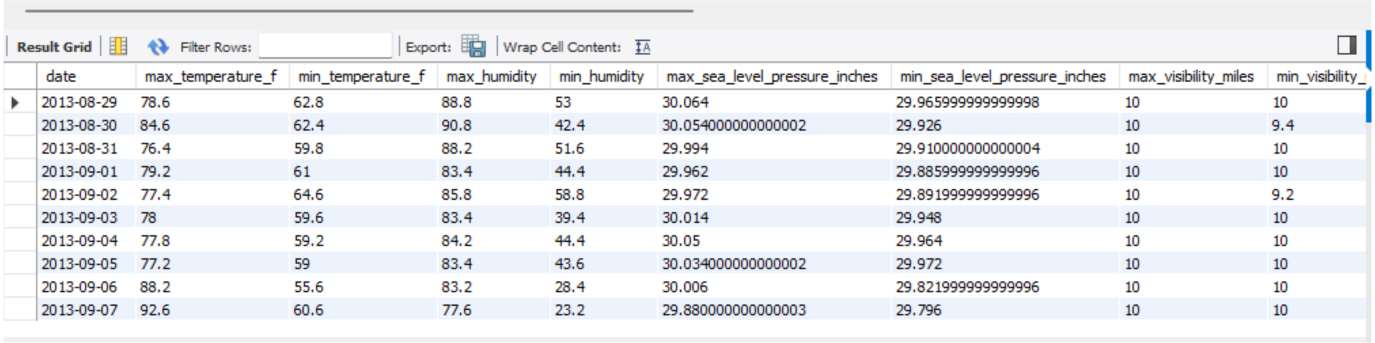
- **Non-Numerical Value**
  The zip codes contain a variety of non-numerical values, including rows with values typed as "nil," which prevents the data from being imported because the column is a numeric one. We resolved the problem by substituting blank values for "nil."

Data Testing

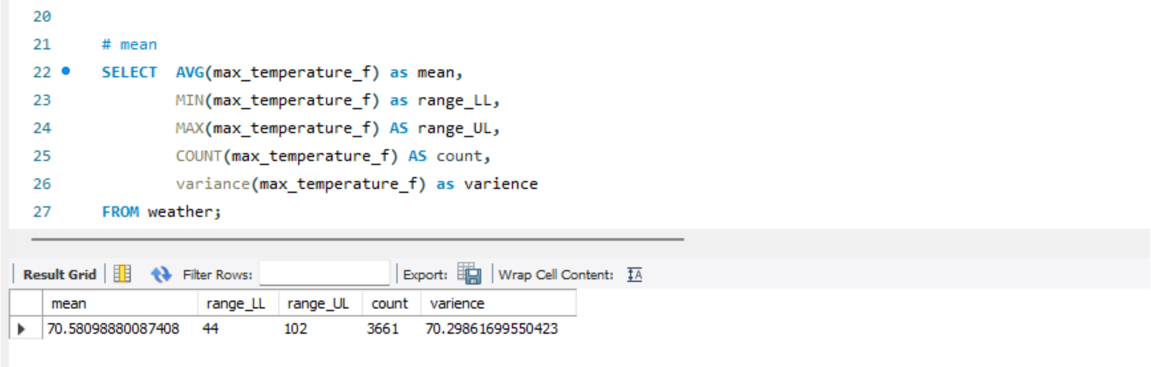
The statistics data was obtained for each table using the following query. Each table's column name was modified, but each table's query was the same. This is for the station table:



```sql
SELECT  date,
        AVG(max_temperature_f) as max_temperature_f,
        AVG(min_temperature_f) as min_temperature_f,
        AVG(max_humidity)  as max_humidity,
        AVG(min_humidity) as min_humidity,
        AVG(max_sea_level_pressure_inches) as max_sea_level_pressure_inches,
        AVG(min_sea_level_pressure_inches) as min_sea_level_pressure_inches,
        AVG(max_visibility_miles) as max_visibility_miles,
        AVG(min_visibility_miles) as min_visibility_miles,
        AVG(max_wind_Speed_mph) as max_wind_Speed_mph,
        AVG(precipitation_inches) as precipitation_inches,
        events,
        AVG(wind_dir_degrees)    as wind_dir_degrees,
        zip_code
FROM weather
GROUP BY date
ORDER BY DATE
limit 10;
```

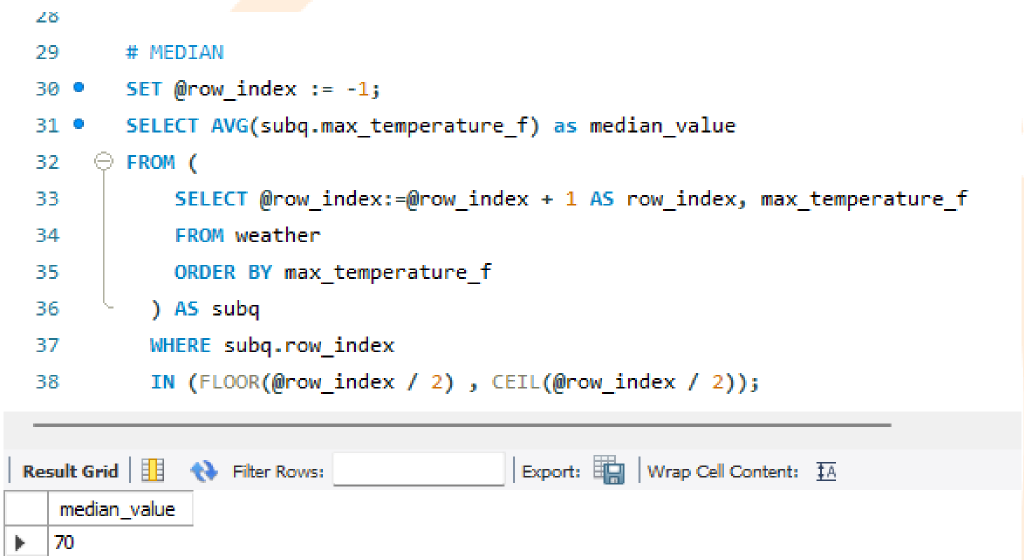


date	max_temperature_f	min_temperature_f	max_humidity	min_humidity	max_sea_level_pressure_inches	min_sea_level_pressure_inches	max_visibility_miles	min_visibility_
2013-08-29	78.6	62.8	88.8	53	30.064	29.965999999999998	10	10
2013-08-30	84.6	62.4	90.8	42.4	30.054000000000002	29.926	10	9.4
2013-08-31	76.4	59.8	88.2	51.6	29.994	29.910000000000004	10	10
2013-09-01	79.2	61	83.4	44.4	29.962	29.885999999999996	10	10
2013-09-02	77.4	64.6	85.8	58.8	29.972	29.891999999999996	10	9.2
2013-09-03	78	59.6	83.4	39.4	30.014	29.948	10	10
2013-09-04	77.8	59.2	84.2	44.4	30.05	29.964	10	10
2013-09-05	77.2	59	83.4	43.6	30.034000000000002	29.972	10	10
2013-09-06	88.2	55.6	83.2	28.4	30.006	29.821999999999996	10	10
2013-09-07	92.6	60.6	77.6	23.2	29.880000000000003	29.796	10	10



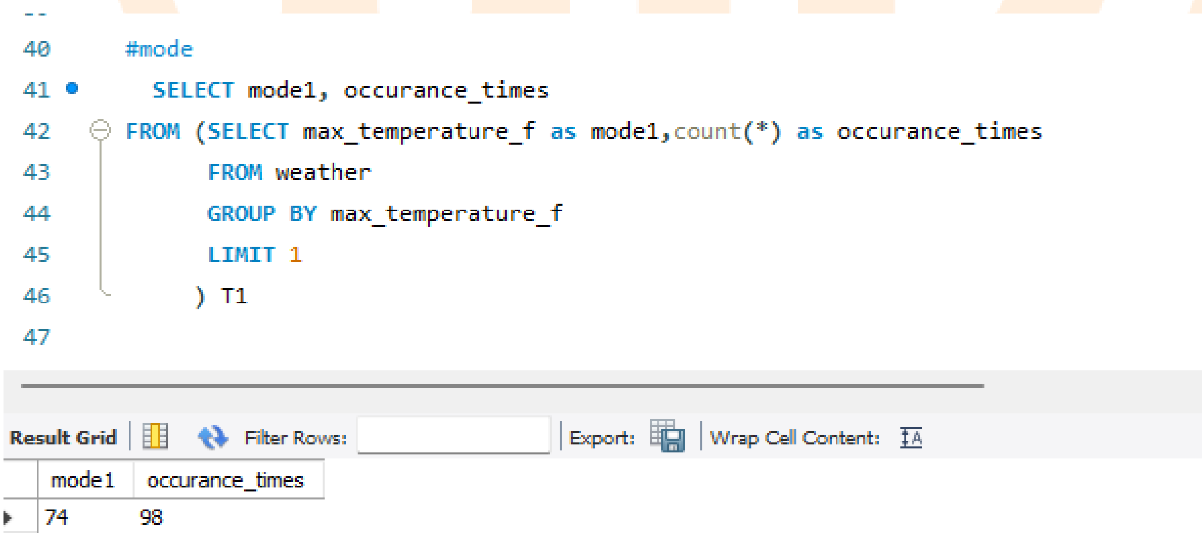
```
# mean
SELECT  AVG(max_temperature_f) as mean,
        MIN(max_temperature_f) as range_LL,
        MAX(max_temperature_f) AS range_UL,
        COUNT(max_temperature_f) AS count,
        variance(max_temperature_f) as varience
FROM weather;
```

mean	range_LL	range_UL	count	varience
70.58098880087408	44	102	3661	70.29861699550423



```
# MEDIAN
SET @row_index := -1;
SELECT AVG(subq.max_temperature_f) as median_value
FROM (
    SELECT @row_index:=@row_index + 1 AS row_index, max_temperature_f
    FROM weather
    ORDER BY max_temperature_f
  ) AS subq
  WHERE subq.row_index
  IN (FLOOR(@row_index / 2) , CEIL(@row_index / 2));
```

median_value
70



```
#mode
  SELECT mode1, occurance_times
FROM (SELECT max_temperature_f as mode1,count(*) as occurance_times
        FROM weather
        GROUP BY max_temperature_f
        LIMIT 1
      ) T1
```

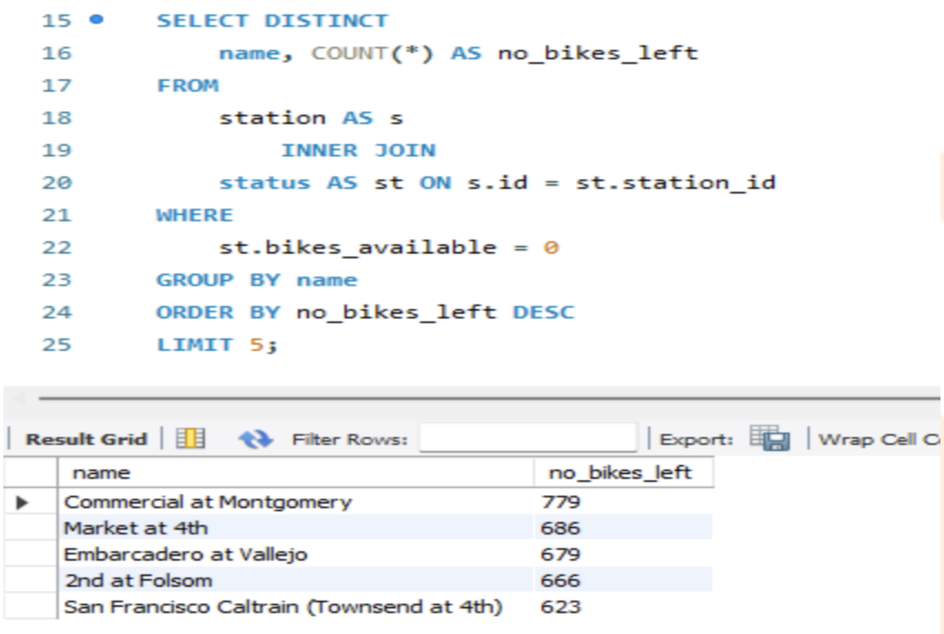
mode1	occurance_times
74	98

Refer to “**Data Understanding”** for the statistical data.

Analysis

In order to better organize the distribution of bikes and boost income by providing more rides at crowded stations, we chose to study the following parameters after learning about the industry.

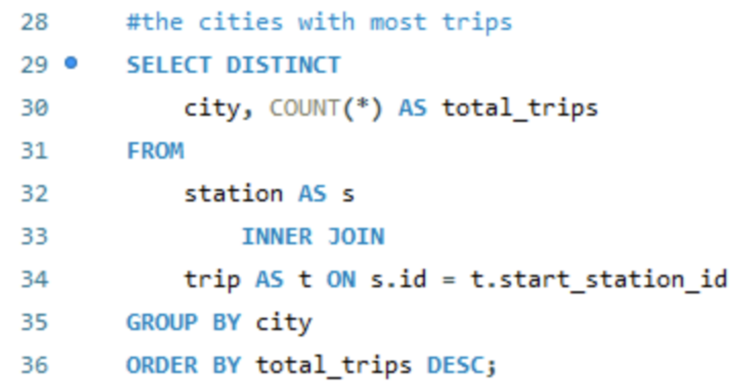
- Stations that often run out of bikes



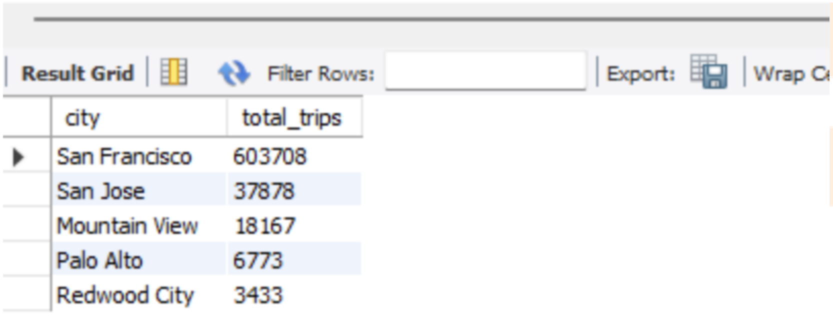
```
SELECT DISTINCT
    name, COUNT(*) AS no_bikes_left
FROM
    station AS s
        INNER JOIN
    status AS st ON s.id = st.station_id
WHERE
    st.bikes_available = 0
GROUP BY name
ORDER BY no_bikes_left DESC
LIMIT 5;
```

name	no_bikes_left
Commercial at Montgomery	779
Market at 4th	686
Embarcadero at Vallejo	679
2nd at Folsom	666
San Francisco Caltrain (Townsend at 4th)	623

- Cities with the most trips

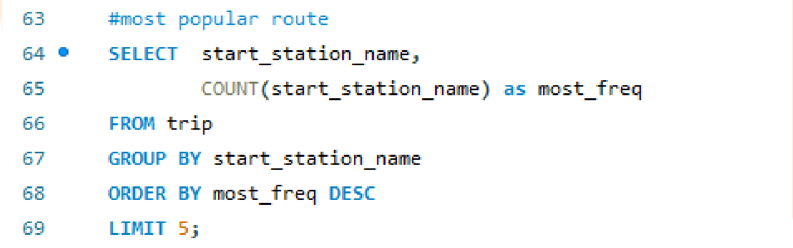


```sql
#the cities with most trips
SELECT DISTINCT
    city, COUNT(*) AS total_trips
FROM
    station AS s
        INNER JOIN
    trip AS t ON s.id = t.start_station_id
GROUP BY city
ORDER BY total_trips DESC;
```

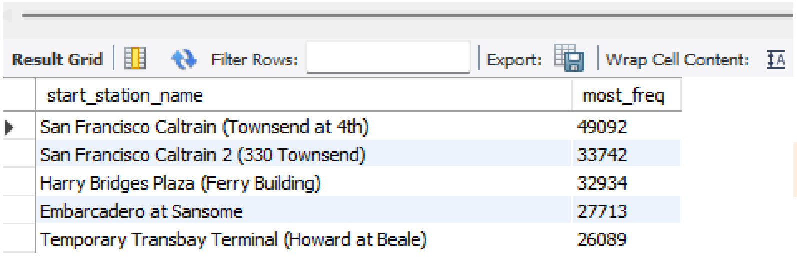


city	total_trips
San Francisco	603708
San Jose	37878
Mountain View	18167
Palo Alto	6773
Redwood City	3433

- Most popular route in San Francisco

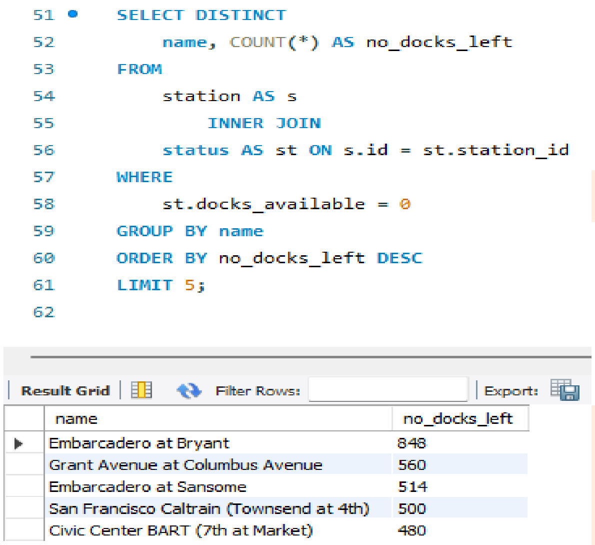


```sql
#most popular route
SELECT  start_station_name,
        COUNT(start_station_name) as most_freq
FROM trip
GROUP BY start_station_name
ORDER BY most_freq DESC
LIMIT 5;
```



start_station_name	most_freq
San Francisco Caltrain (Townsend at 4th)	49092
San Francisco Caltrain 2 (330 Townsend)	33742
Harry Bridges Plaza (Ferry Building)	32934
Embarcadero at Sansome	27713
Temporary Transbay Terminal (Howard at Beale)	26089

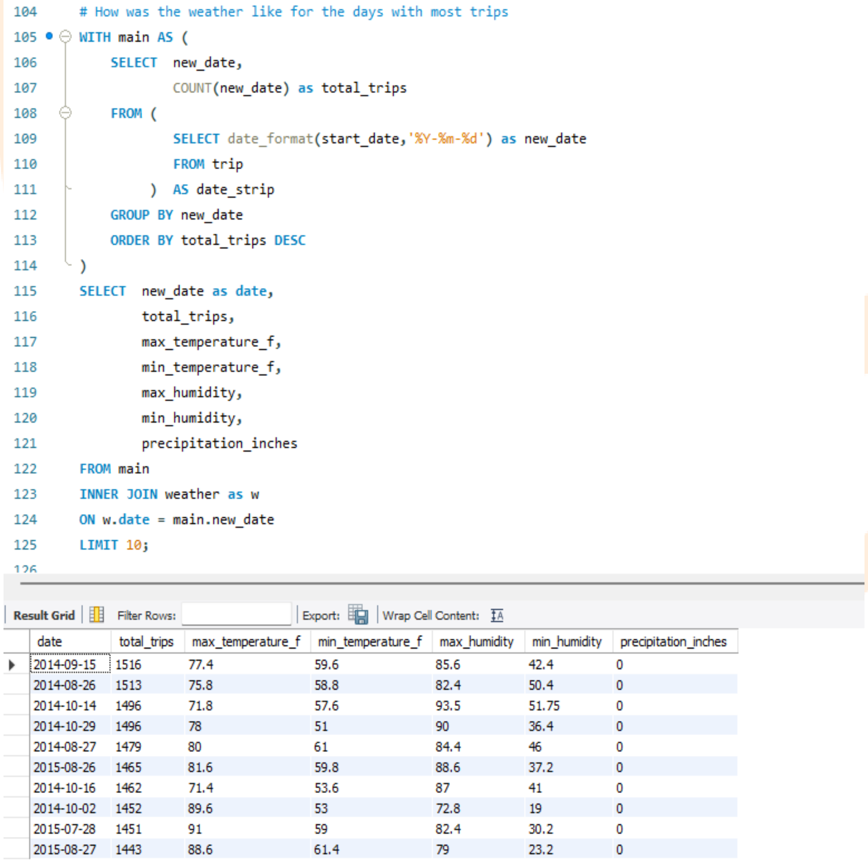
- Stations where docks ran out



```sql
SELECT DISTINCT
    name, COUNT(*) AS no_docks_left
FROM
    station AS s
        INNER JOIN
    status AS st ON s.id = st.station_id
WHERE
    st.docks_available = 0
GROUP BY name
ORDER BY no_docks_left DESC
LIMIT 5;
```

name	no_docks_left
Embarcadero at Bryant	848
Grant Avenue at Columbus Avenue	560
Embarcadero at Sansome	514
San Francisco Caltrain (Townsend at 4th)	500
Civic Center BART (7th at Market)	480

- How did the weather look like when we had the most customer trips(top 10)



```sql
# How was the weather like for the days with most trips
WITH main AS (
    SELECT  new_date,
            COUNT(new_date) as total_trips
    FROM (
            SELECT date_format(start_date,'%Y-%m-%d') as new_date
            FROM trip
         )  AS date_strip
    GROUP BY new_date
    ORDER BY total_trips DESC
)
SELECT  new_date as date,
        total_trips,
        max_temperature_f,
        min_temperature_f,
        max_humidity,
        min_humidity,
        precipitation_inches
FROM main
INNER JOIN weather as w
ON w.date = main.new_date
LIMIT 10;
```

date	total_trips	max_temperature_f	min_temperature_f	max_humidity	min_humidity	precipitation_inches
2014-09-15	1516	77.4	59.6	85.6	42.4	0
2014-08-26	1513	75.8	58.8	82.4	50.4	0
2014-10-14	1496	71.8	57.6	93.5	51.75	0
2014-10-29	1496	78	51	90	36.4	0
2014-08-27	1479	80	61	84.4	46	0
2015-08-26	1465	81.6	59.8	88.6	37.2	0
2014-10-16	1462	71.4	53.6	87	41	0
2014-10-02	1452	89.6	53	72.8	19	0
2015-07-28	1451	91	59	82.4	30.2	0
2015-08-27	1443	88.6	61.4	79	23.2	0

Improvements that could be made based on the present business model include:

- There should be more bikes at Commercial at Montgomery station because that is where we see the most significant bike shortages.
- Most trips are made in Redwood City, while the fewest are made in San Francisco. Thus, it is important to plan the number of bikes each station will receive.
- Embarcadero at Bryant is the station that ran out of docks; consequently, the number of docks needs to be expanded.
- The most frequented route in San Francisco was discovered to be from SF Caltrain to Townsend at 4th. As a result, we must always ensure that docks are available along this route.
- The fact that fewer excursions were taken below and above the average maximum temperature (70 degrees Fahrenheit) indicates that the weather did not significantly affect the trips.