Project Title - Northeastern Transportation System

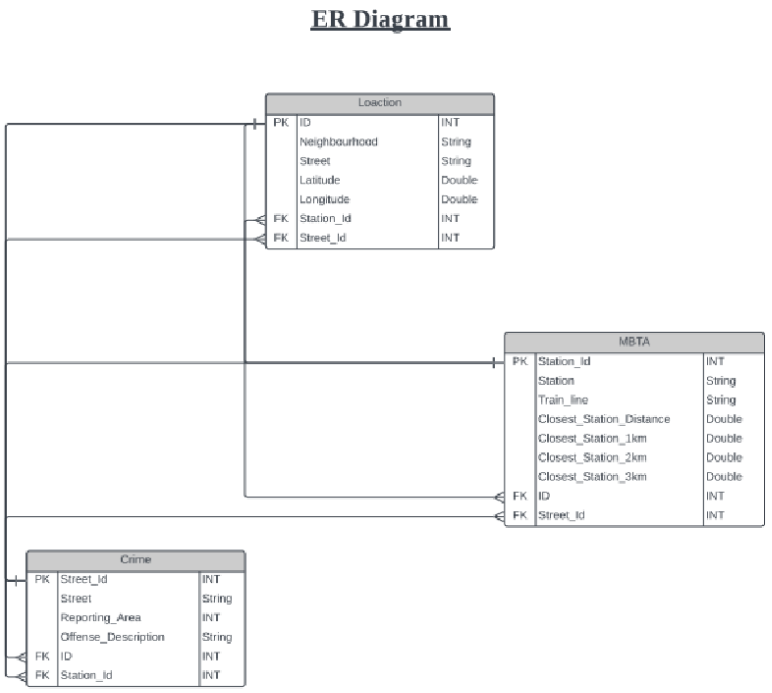
Project Scope:

The project intends to serve the Northeastern community by assisting students, faculty, and other community members to reach their destinations promptly and safely regardless of the time using different modes of transport viz. cabs, buses, trams, driving.

Project Description:

We are developing a transportation database system, specifically for the Northeastern community, where the source will be Boston city for the traveller to go to his desired location which will comprise (Station Names, Street Names, Crime Rate and Offense Description). Once the location is determined, he will be able to see the mode of transportation, closest station of T-line (He will also get to know A particular station has how many lines and also get to know the type of lines eg-Red , Orange , Green). According to neighbourhood he will be also able to get to know the station within the range of 3miles in the database which will also include the street name and the percentage of crime rate on that street and also the type of offense that has been occurred. This will be very helpful for commuters who are uncertain of which streets are safe or not, to reach their destination so the safety parameter is distinct and the crime rate of a particular street. The user then can choose his or her route to the destination in accordance with it.

ER Diagram:



Abbreviation:

PK-Primary Key

FK-Foreign Key

ER Diagram SQL Code:

```
CREATE TABLE `MBTA` (
  `Station_Id` INT,
  `Station` String,
  `Train_line` String,
  `Closest_Station_Distance` Double,
  `Closest_Station_1km` Double,
  `Closest_Station_2km` Double,
  `Closest_Station_3km` Double,
  `ID` INT,
  `Street_Id` INT,
  PRIMARY KEY (`Station_Id`),
  FOREIGN KEY (`Street_Id`) REFERENCES `Crime`(`Street_Id`),
  FOREIGN KEY (`ID`) REFERENCES `Loaction`(`ID`)
);

CREATE TABLE `Crime` (
  `Street_Id` INT,
  `Street` String,
  `Reporting_Area` INT,
  `Offense_Description` String,
  `ID` INT,
  `Station_Id` INT,
  PRIMARY KEY (`Street_Id`),
  FOREIGN KEY (`Station_Id`) REFERENCES `MBTA`(`Station_Id`),
  FOREIGN KEY (`ID`) REFERENCES `Loaction`(`ID`)
);

CREATE TABLE `Loaction` (
```

```
  `ID` INT,
  `Neighbourhood` String,
  `Street` String,
  `Latitude` Double,
  `Longitude` Double,
  `Station_Id` INT,
  `Street_Id` INT,
  PRIMARY KEY (`ID`),
  FOREIGN KEY (`Street_Id`) REFERENCES `Crime`(`Street_Id`),
  FOREIGN KEY (`Station_Id`) REFERENCES `MBTA`(`Station_Id`)
);
```

Sample Table:

Table 1: Location

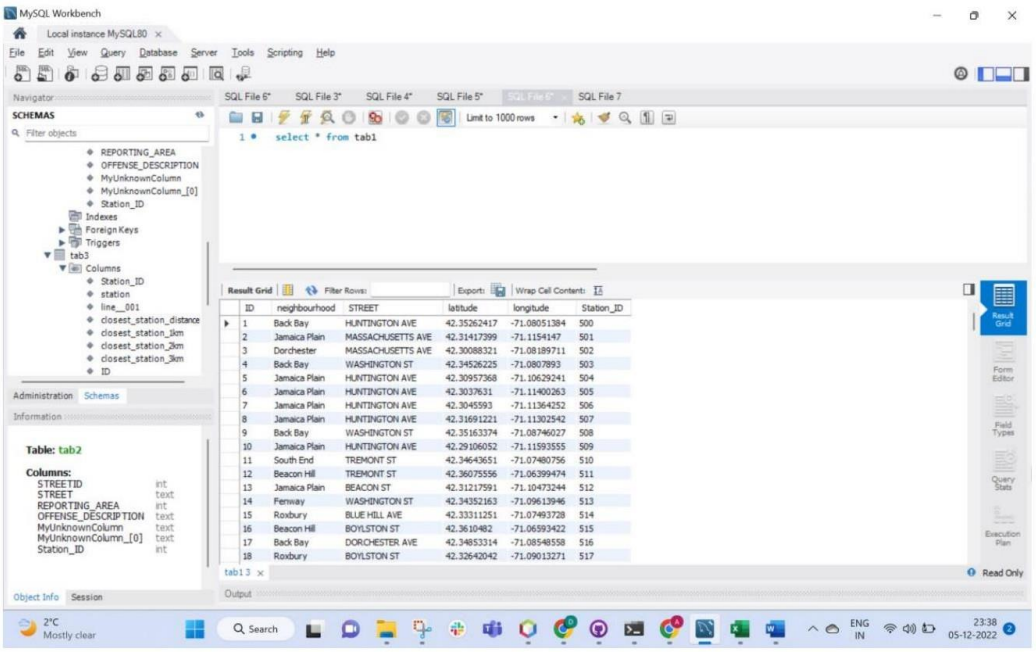


Table 2: Crime

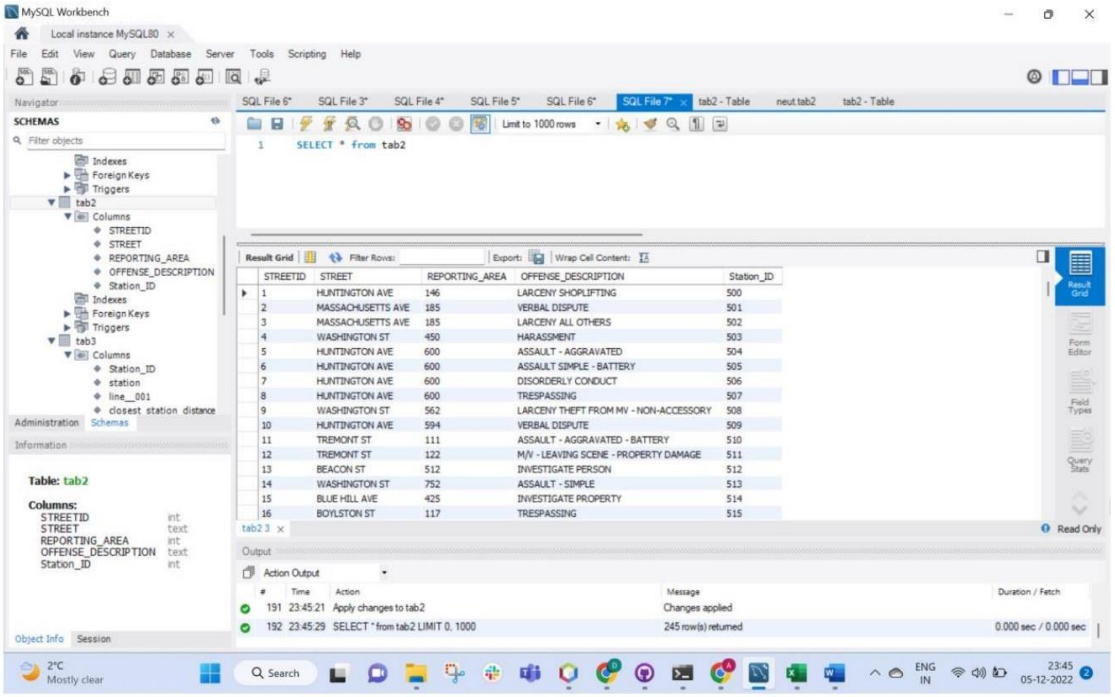
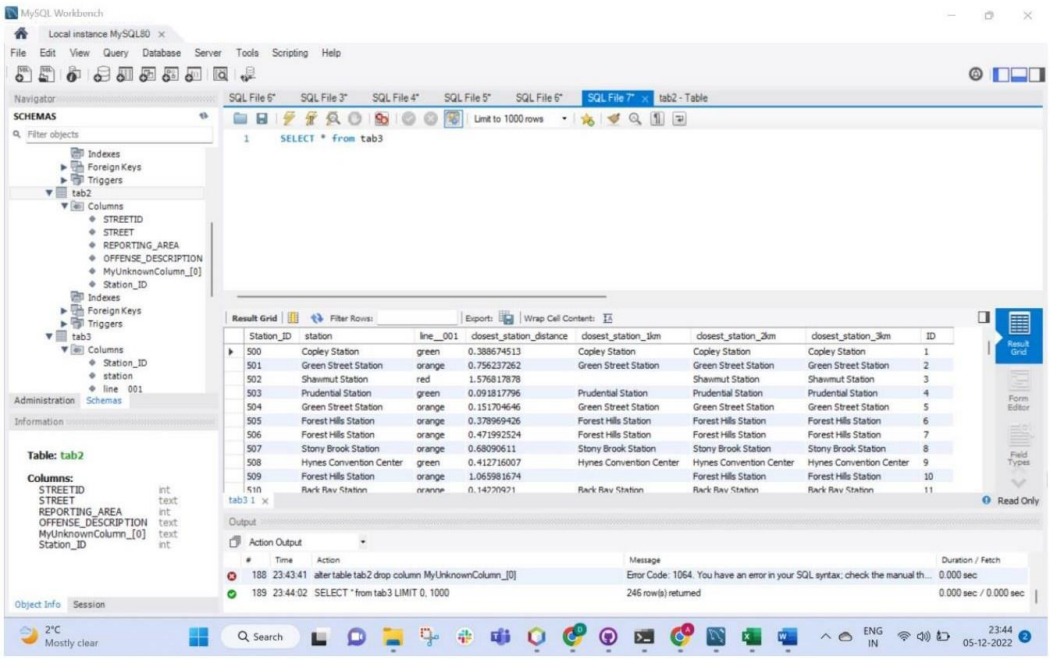
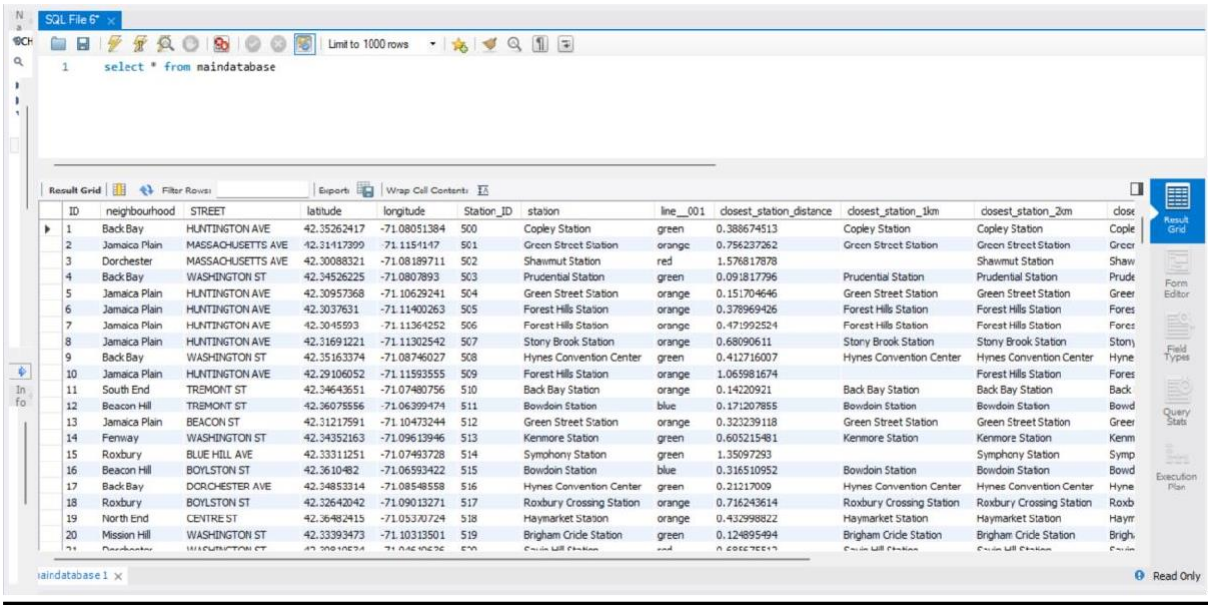


Table 3: MBTA



Main Database:

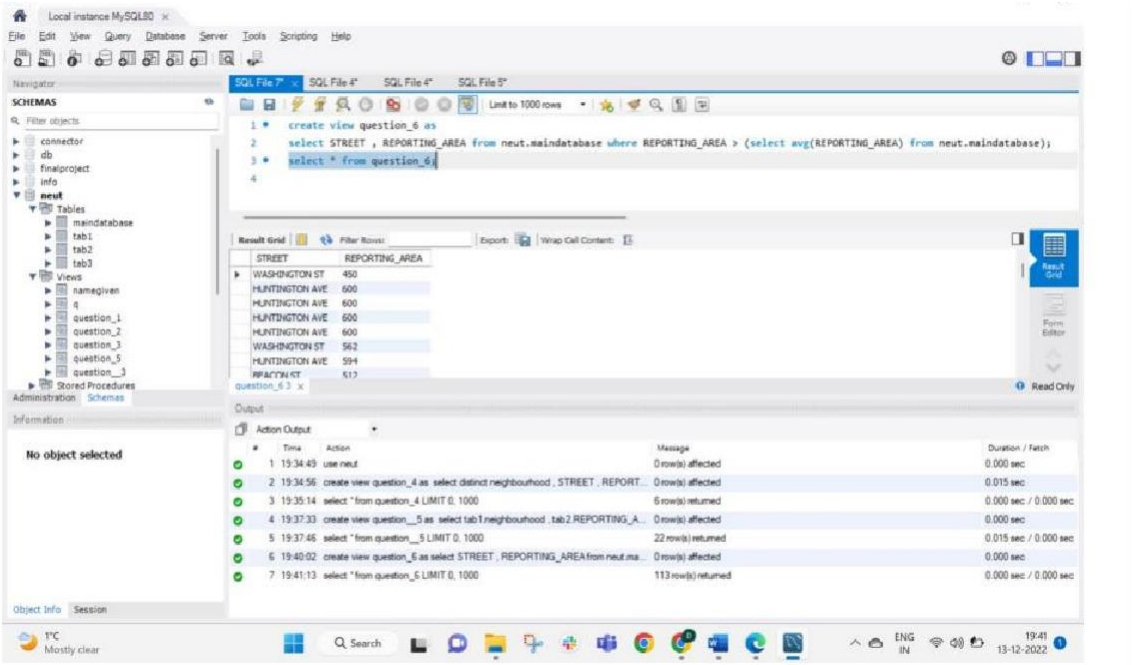


ID	neighbourhood	STREET	latitude	longitude	Station_ID	station	line_001	closest_station_distance	closest_station_1km	closest_station_2km	close
1	Back Bay	HUNTINGTON AVE	42.35262417	-71.08051384	500	Copley Station	green	0.388674513	Copley Station	Copley Station	Copl
2	Jamaica Plain	MASSACHUSETTS AVE	42.31417399	-71.1151417	501	Green Street Station	orange	0.756237262	Green Street Station	Green Street Station	Gree
3	Dorchester	MASSACHUSETTS AVE	42.30088321	-71.08189711	502	Shawmut Station	red	1.576817878		Shawmut Station	Shaw
4	Back Bay	WASHINGTON ST	42.34526225	-71.0807893	503	Prudential Station	green	0.091817796	Prudential Station	Prudential Station	Prud
5	Jamaica Plain	HUNTINGTON AVE	42.30957368	-71.10629241	504	Green Street Station	orange	0.151704646	Green Street Station	Green Street Station	Gree
6	Jamaica Plain	HUNTINGTON AVE	42.3037631	-71.11400263	505	Forest Hills Station	orange	0.378969426	Forest Hills Station	Forest Hills Station	Fores
7	Jamaica Plain	HUNTINGTON AVE	42.3045593	-71.11364252	506	Forest Hills Station	orange	0.471992524	Forest Hills Station	Forest Hills Station	Fores
8	Jamaica Plain	HUNTINGTON AVE	42.31691221	-71.11302542	507	Stony Brook Station	orange	0.68090611	Stony Brook Station	Stony Brook Station	Stony
9	Back Bay	WASHINGTON ST	42.35163374	-71.08746027	508	Hynes Convention Center	green	0.412716007	Hynes Convention Center	Hynes Convention Center	Hyne
10	Jamaica Plain	HUNTINGTON AVE	42.29106052	-71.11593555	509	Forest Hills Station	orange	1.065981674		Forest Hills Station	Fores
11	South End	TREMONT ST	42.34643651	-71.07480756	510	Back Bay Station	orange	0.14220921	Back Bay Station	Back Bay Station	Back
12	Beacon Hill	TREMONT ST	42.36075556	-71.06399474	511	Bowdoin Station	blue	0.171207855	Bowdoin Station	Bowdoin Station	Bowd
13	Jamaica Plain	BEACON ST	42.31217591	-71.10473244	512	Green Street Station	orange	0.323239118	Green Street Station	Green Street Station	Gree
14	Fenway	WASHINGTON ST	42.34352163	-71.09613946	513	Kenmore Station	green	0.605215481	Kenmore Station	Kenmore Station	Kenm
15	Roxbury	BLUE HILL AVE	42.33311251	-71.07493728	514	Symphony Station	green	1.35097293		Symphony Station	Symp
16	Beacon Hill	BOYLSTON ST	42.3610482	-71.06593422	515	Bowdoin Station	blue	0.316510952	Bowdoin Station	Bowdoin Station	Bowd
17	Back Bay	DORCHESTER AVE	42.34853314	-71.08548558	516	Hynes Convention Center	green	0.21217009	Hynes Convention Center	Hynes Convention Center	Hyne
18	Roxbury	BOYLSTON ST	42.32642042	-71.09013271	517	Roxbury Crossing Station	orange	0.716243614	Roxbury Crossing Station	Roxbury Crossing Station	Roxb
19	North End	CENTRE ST	42.36482415	-71.05370724	518	Haymarket Station	orange	0.432998822	Haymarket Station	Haymarket Station	Haym
20	Mission Hill	WASHINGTON ST	42.33393473	-71.10313501	519	Brigham Circle Station	green	0.124895494	Brigham Circle Station	Brigham Circle Station	Brigh

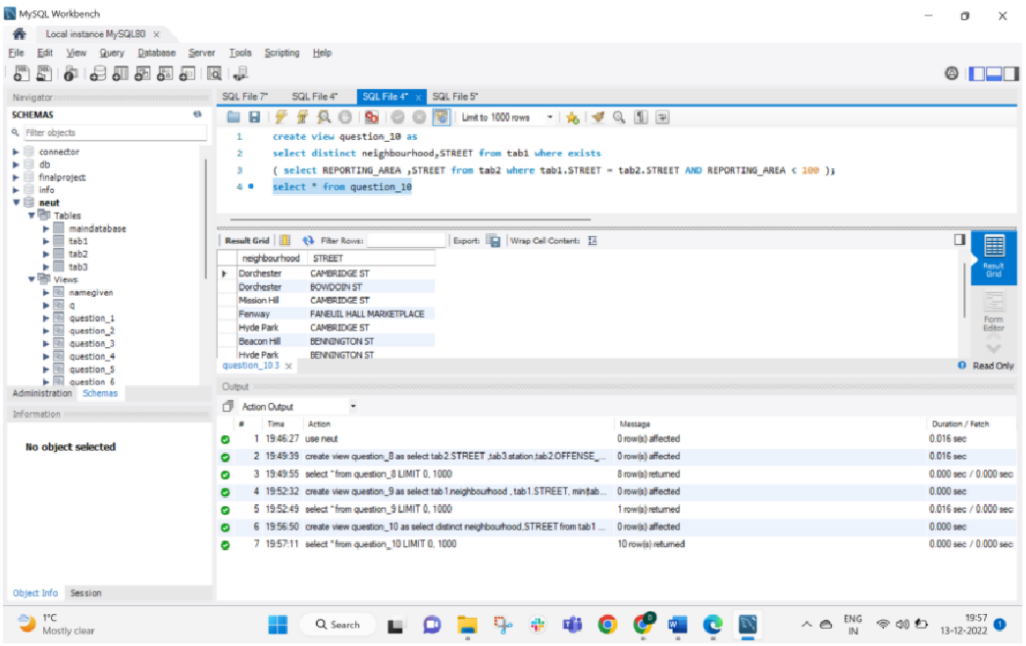
closest_station_3km	REPORTING_AREA	OFFENSE_DESCRIPTION
Copley Station	146	LARCENY SHOPLIFTING
Green Street Station	185	VERBAL DISPUTE
Shawmut Station	185	LARCENY ALL OTHERS
Prudential Station	450	HARASSMENT
Green Street Station	600	ASSAULT - AGGRAVATED
Forest Hills Station	600	ASSAULT SIMPLE - BATTERY
Forest Hills Station	600	DISORDERLY CONDUCT
Stony Brook Station	600	TRESPASSING
Hynes Convention Center	562	LARCENY THEFT FROM MV - NON-ACCESSORY
Forest Hills Station	594	VERBAL DISPUTE
Back Bay Station	111	ASSAULT - AGGRAVATED - BATTERY
Bowdoin Station	122	M/V - LEAVING SCENE - PROPERTY DAMAGE
Green Street Station	512	INVESTIGATE PERSON
Kenmore Station	752	ASSAULT - SIMPLE
Symphony Station	425	INVESTIGATE PROPERTY
Bowdoin Station	117	TRESPASSING
Hynes Convention Center	348	ASSAULT - AGGRAVATED - BATTERY
Roxbury Crossing Station	138	INVESTIGATE PERSON
Haymarket Station	291	ASSAULT SIMPLE - BATTERY
Brigham Circle Station	778	PROPERTY - FOUND
Savin Hill Station	350	ASSAULT SIMPLE - BATTERY

Output of Views:

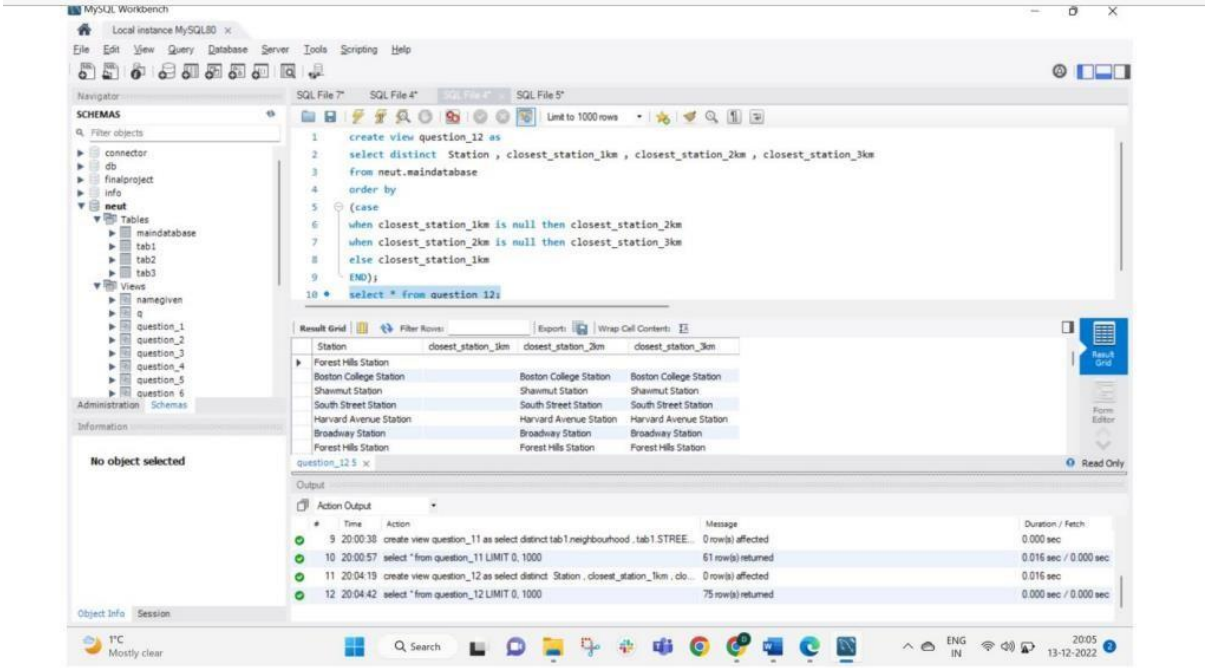
Q.1 To display street name where Reporting crime count more than the average reporting count.



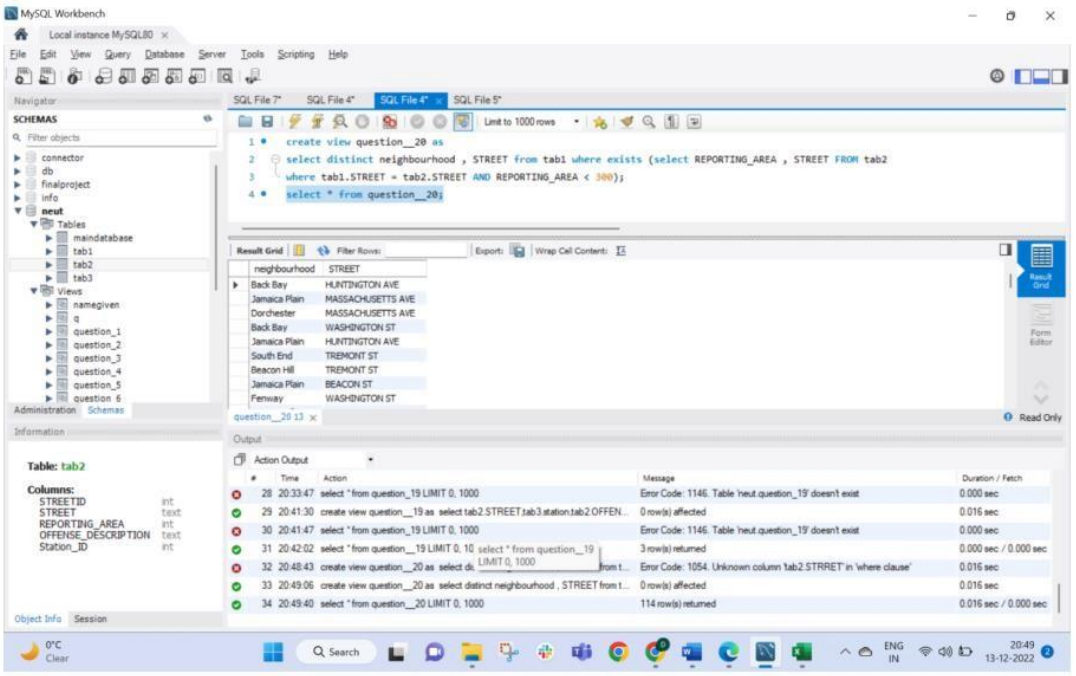
Q.2 To find the neighbourhood name and street name who has reported crime less than 100



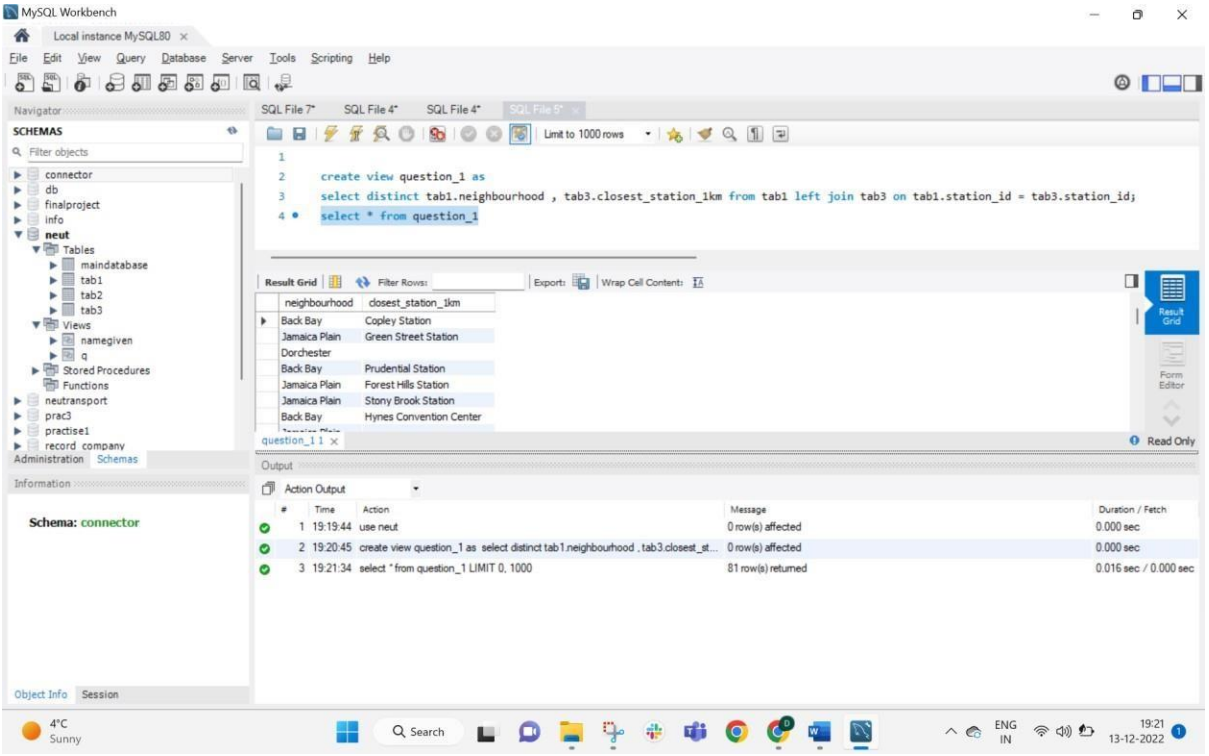
Q.3 Check for the closest station from 1,2,3km using case expression



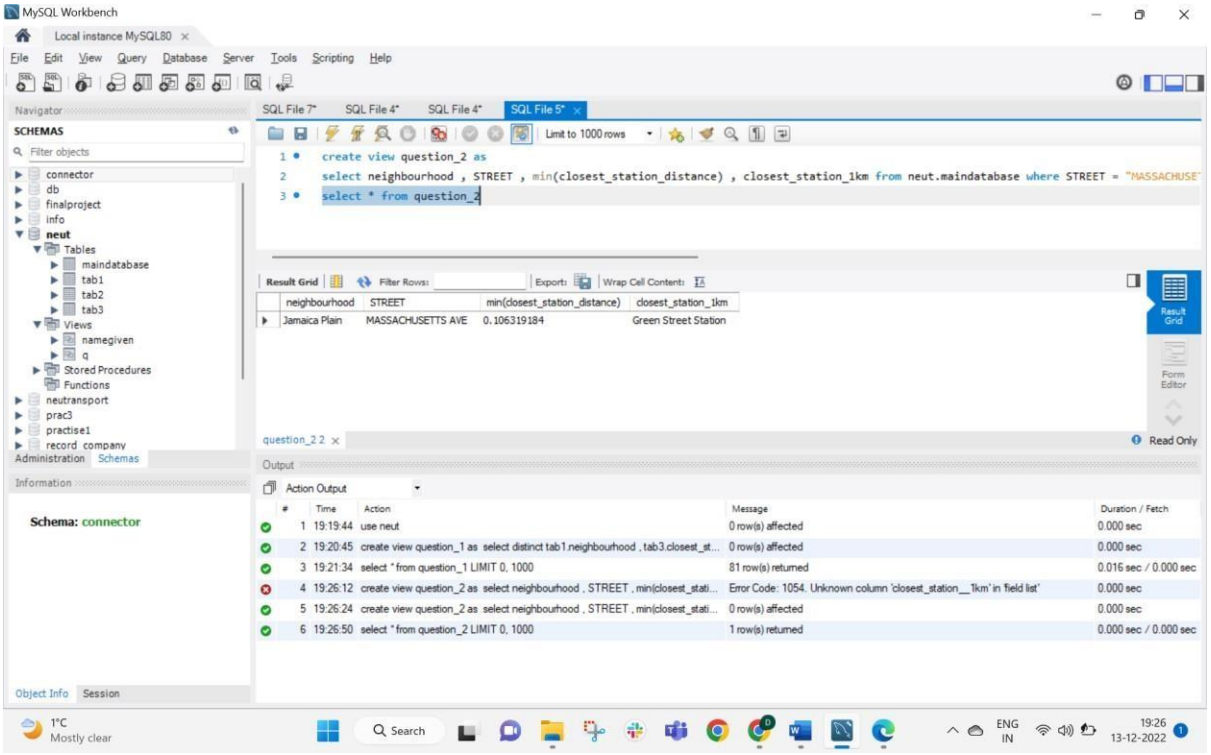
Q.4 To find the neighbourhood's name and street name who has reported crime less than 300



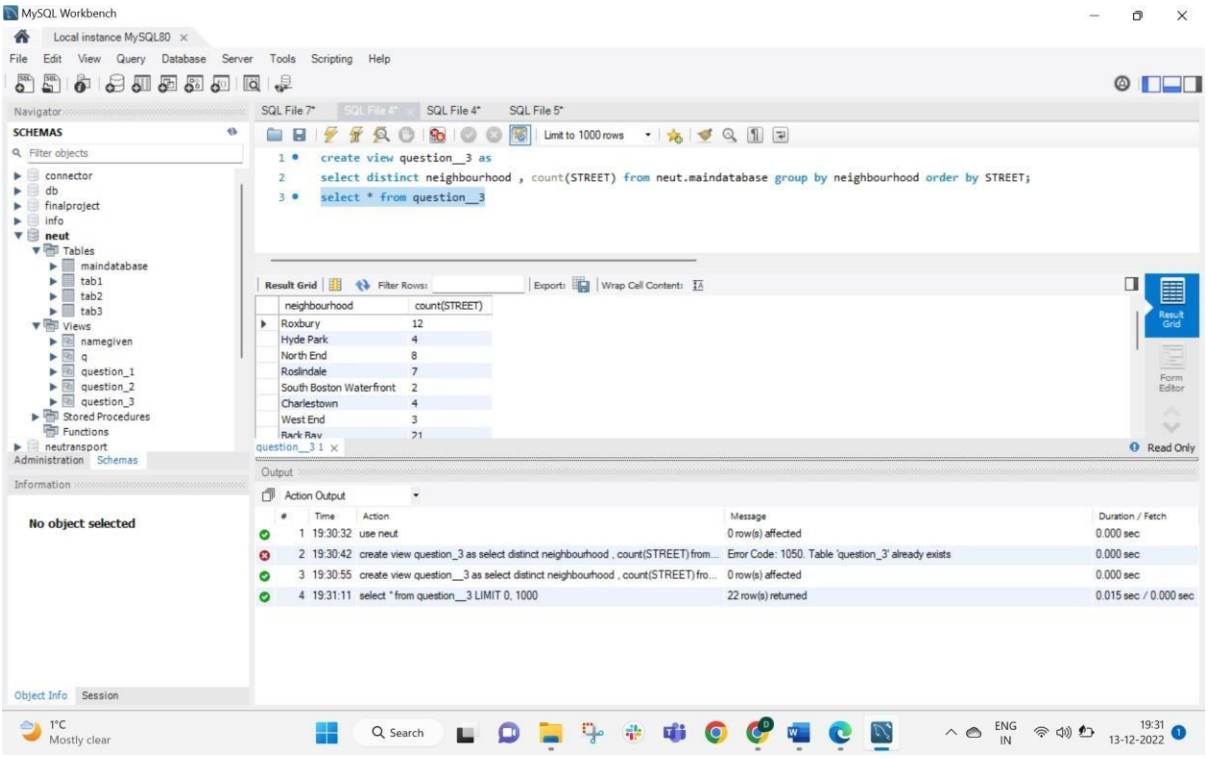
Q5. Which are the stations near to particular neighbourhood ?



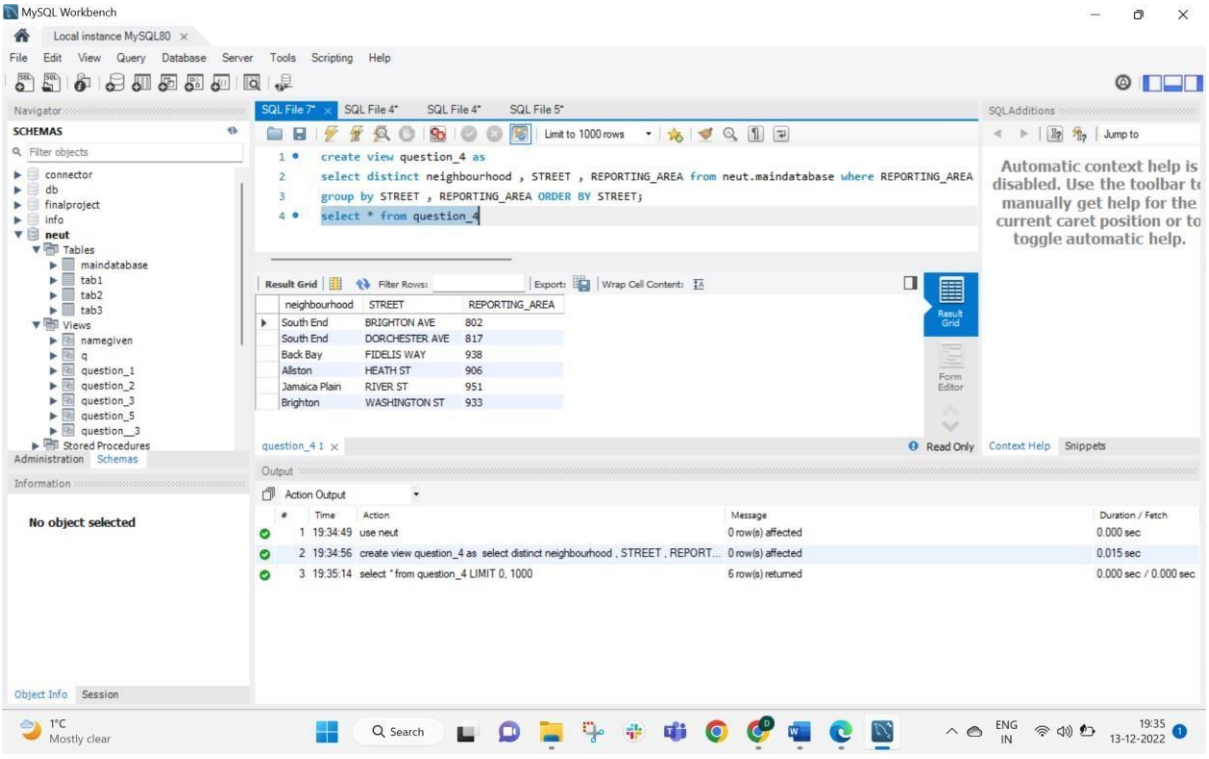
Q6. To find the nearest station , distance , neighbourhood from the street name MASSACHUSETTS AVE



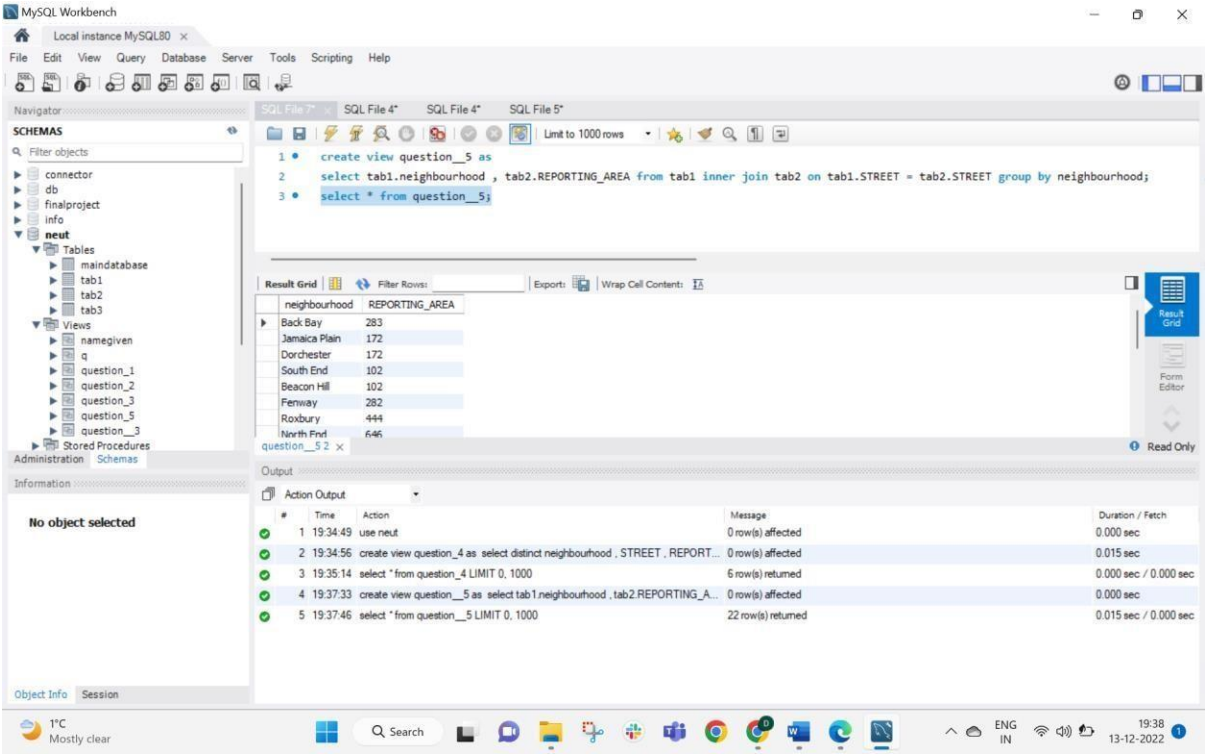
Q7. To count the number of street in each neighbourhood



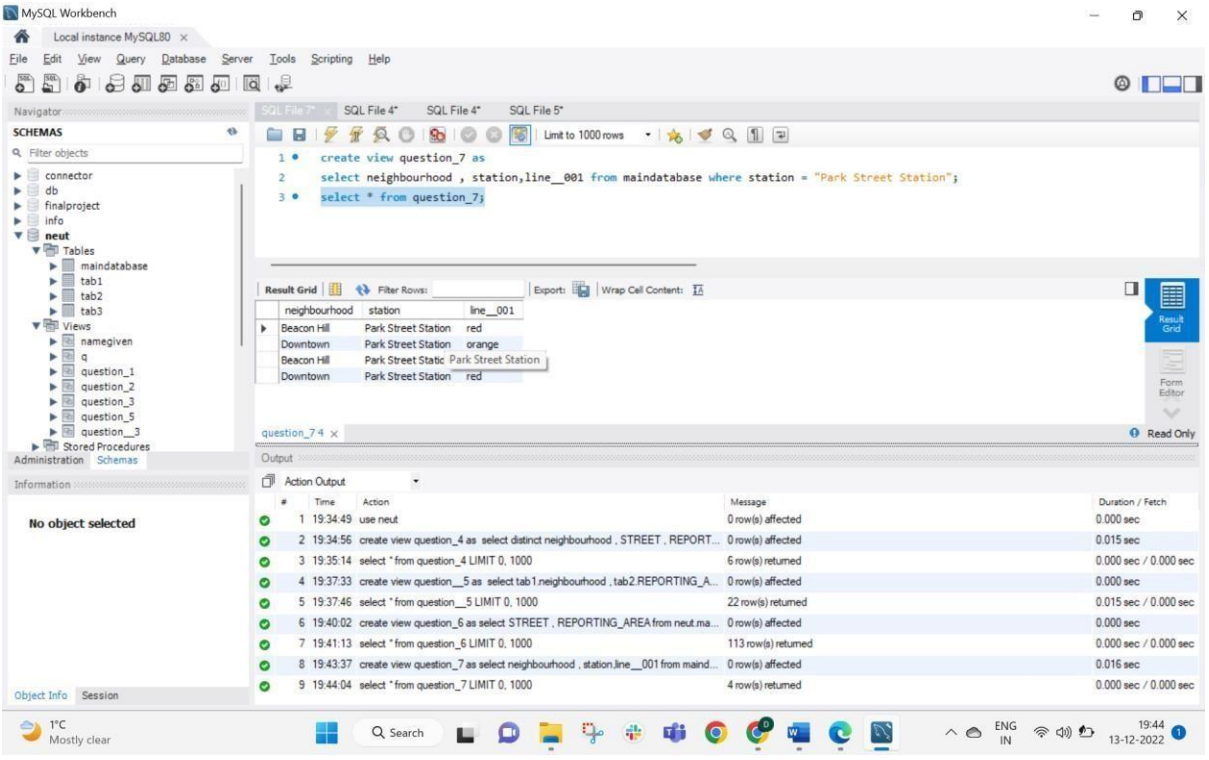
Q8. To find the street, neighbourhood name where reporting crime counts are more than 800



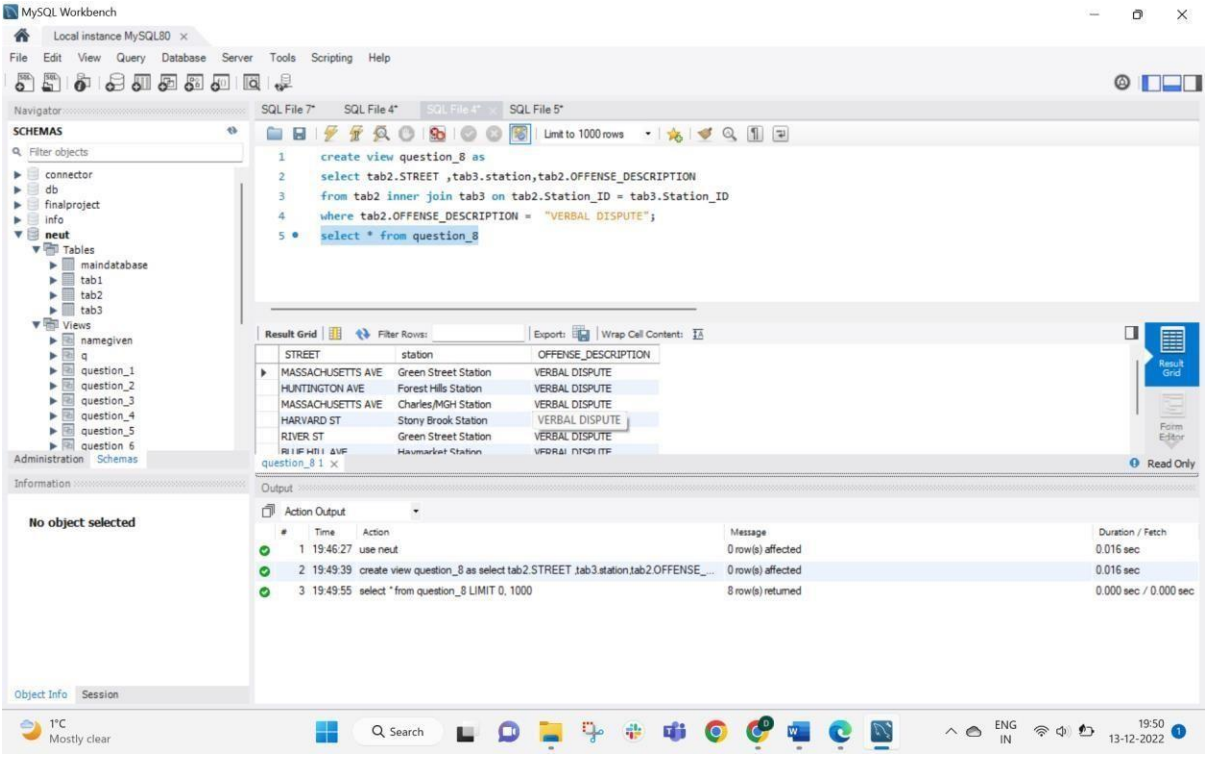
Q9. What is the count of reporting crime of a particular neighbourhood?



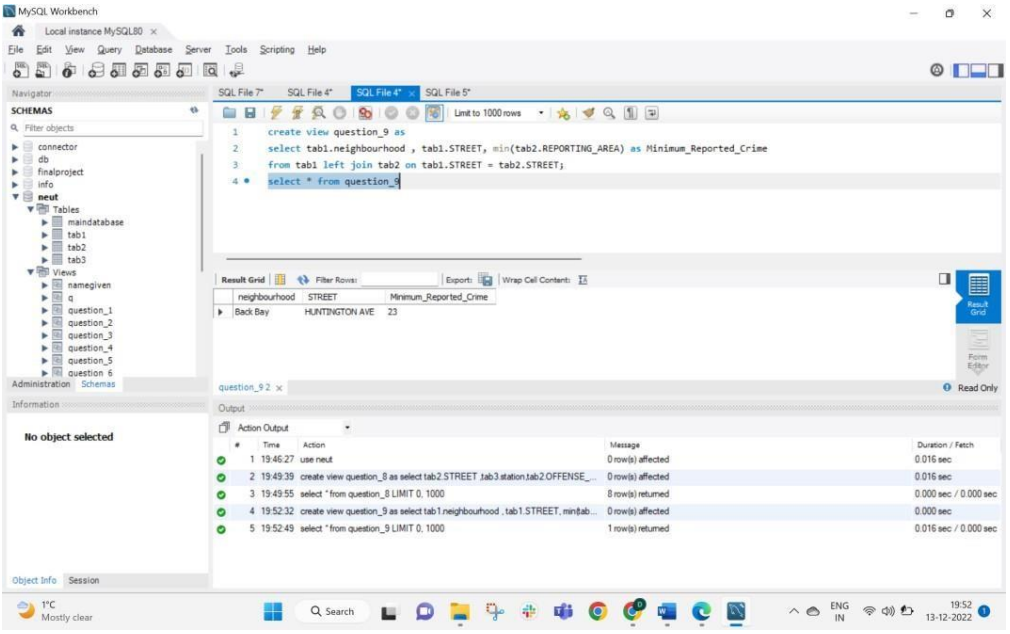
Q10. To select the station and neighbourhood name which has more than one train connection



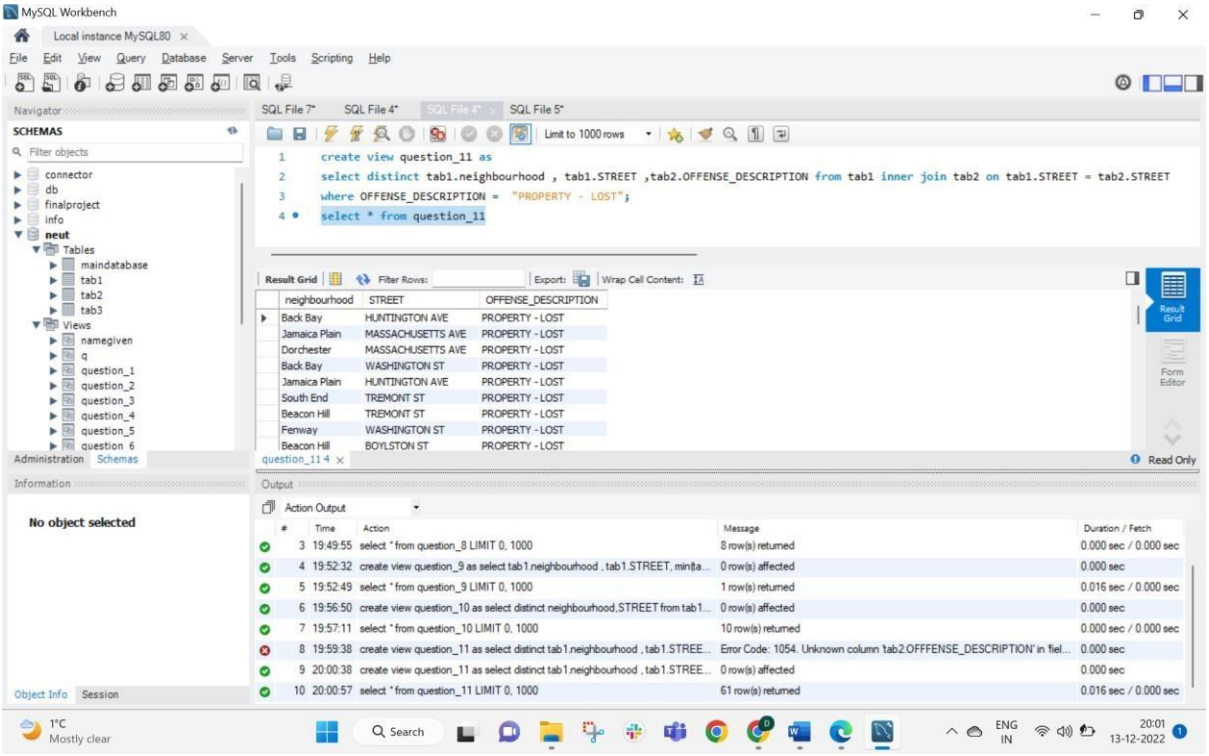
Q11. To select the street name and station who has committed the offence of Verbal Dispute



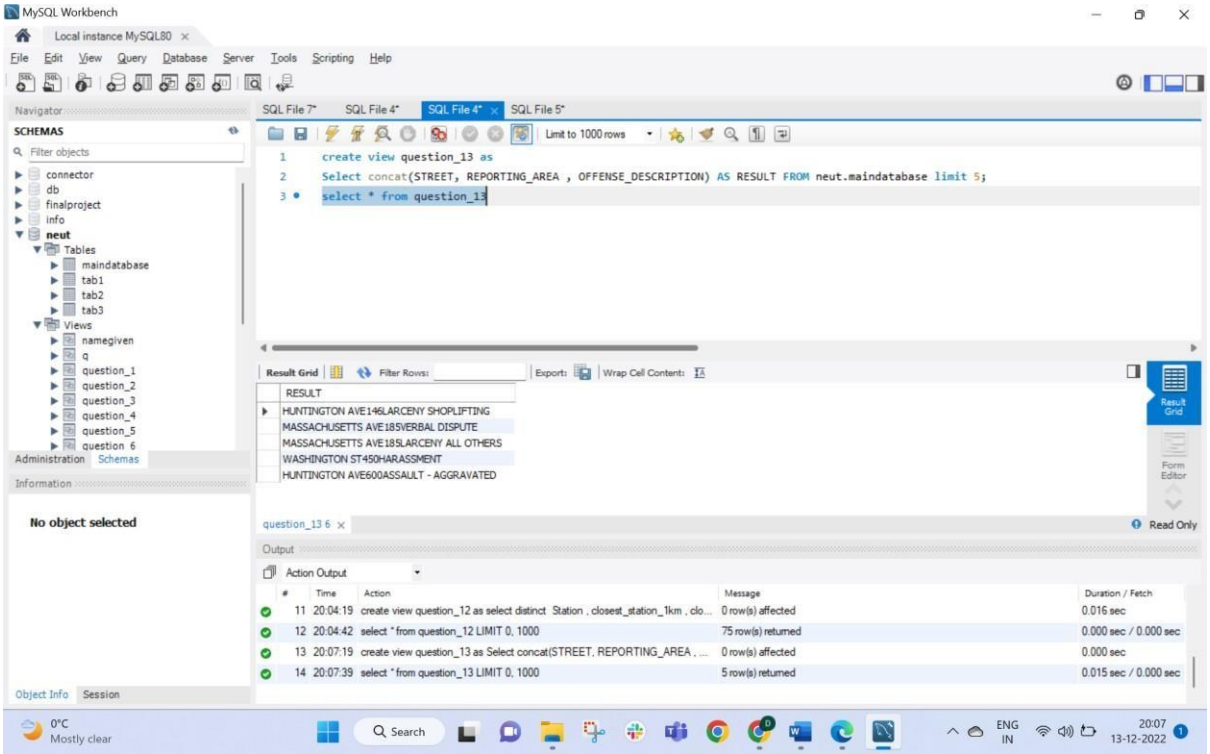
Q12. To select the neighbourhood, street and reported crime area which has the most minimum reported crime



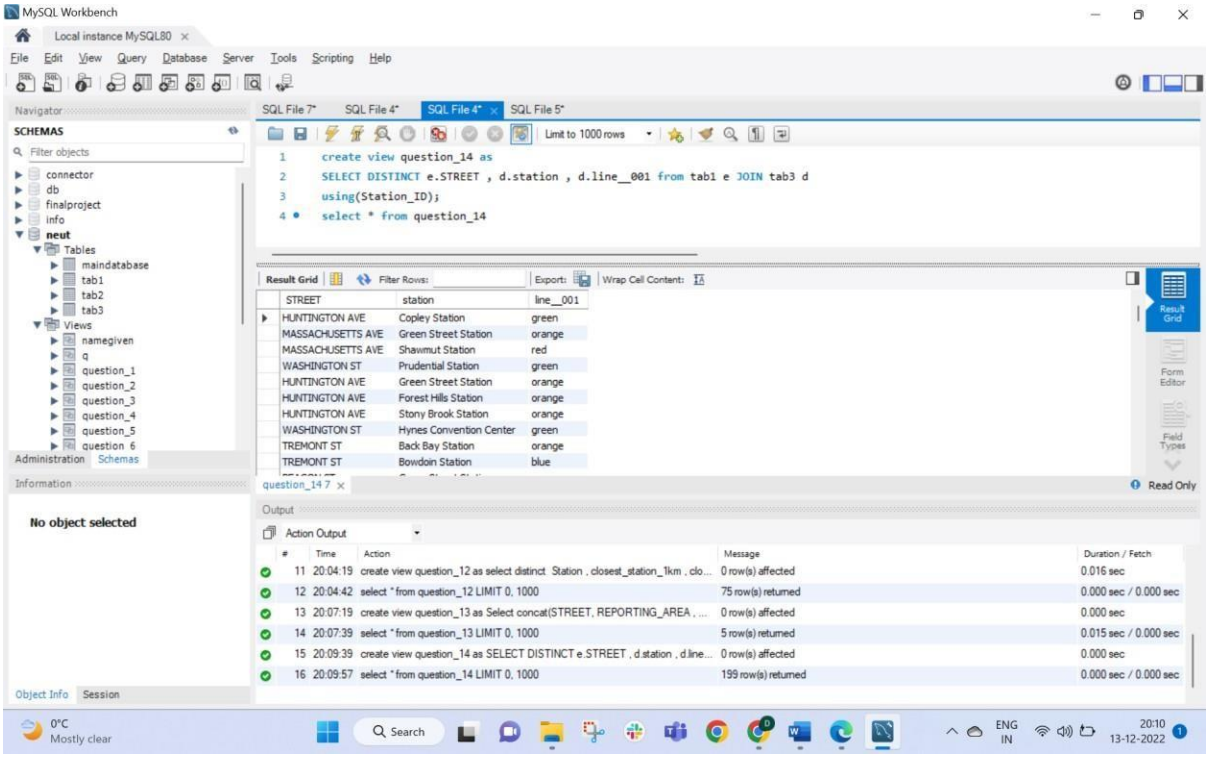
Q13. To find the Neighbourhood name and street name and who has reported the offense of property lost



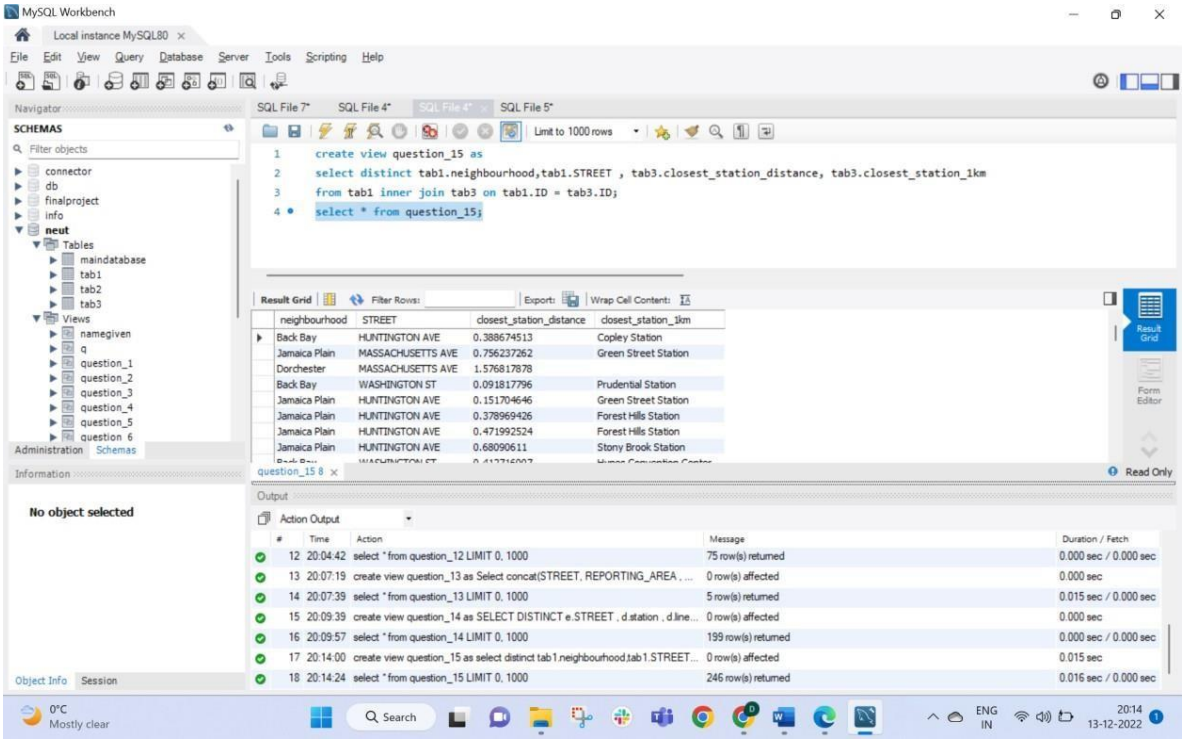
Q14. To find the recent 5 offense, reported crime of the streets



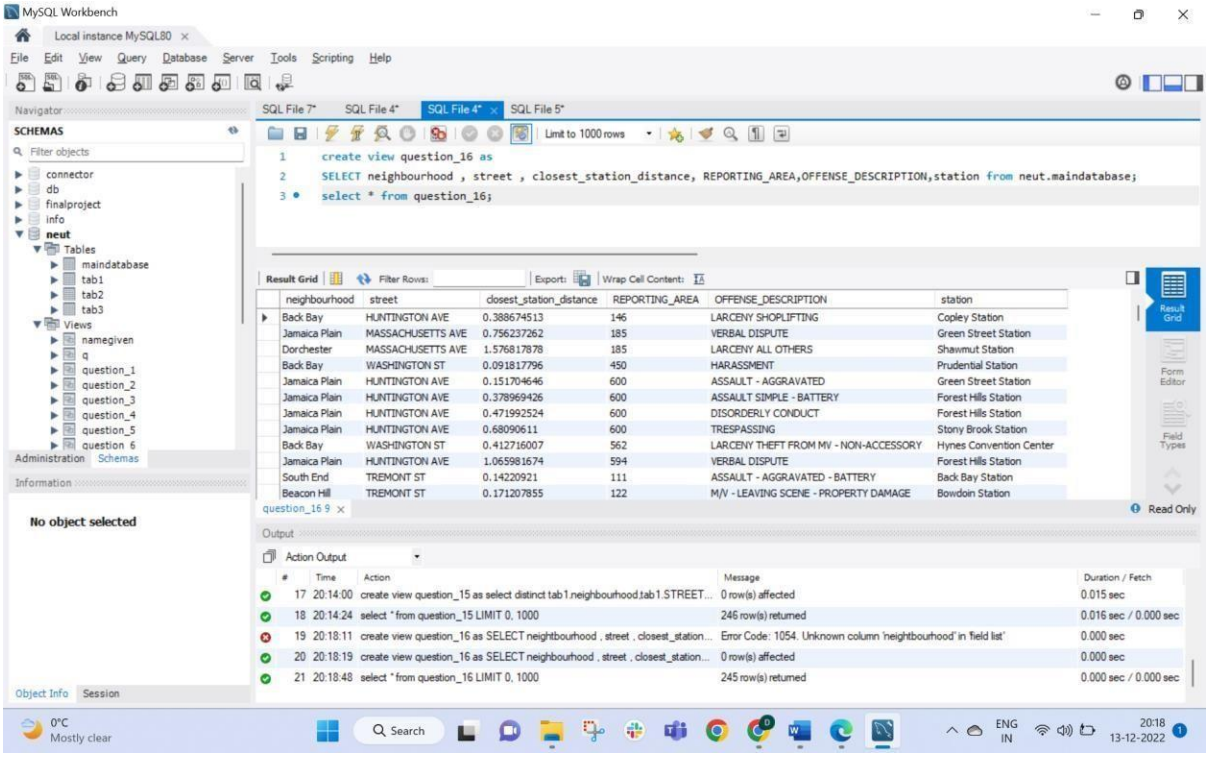
Q15. To find the station, the train line of the street using “USING Clause”



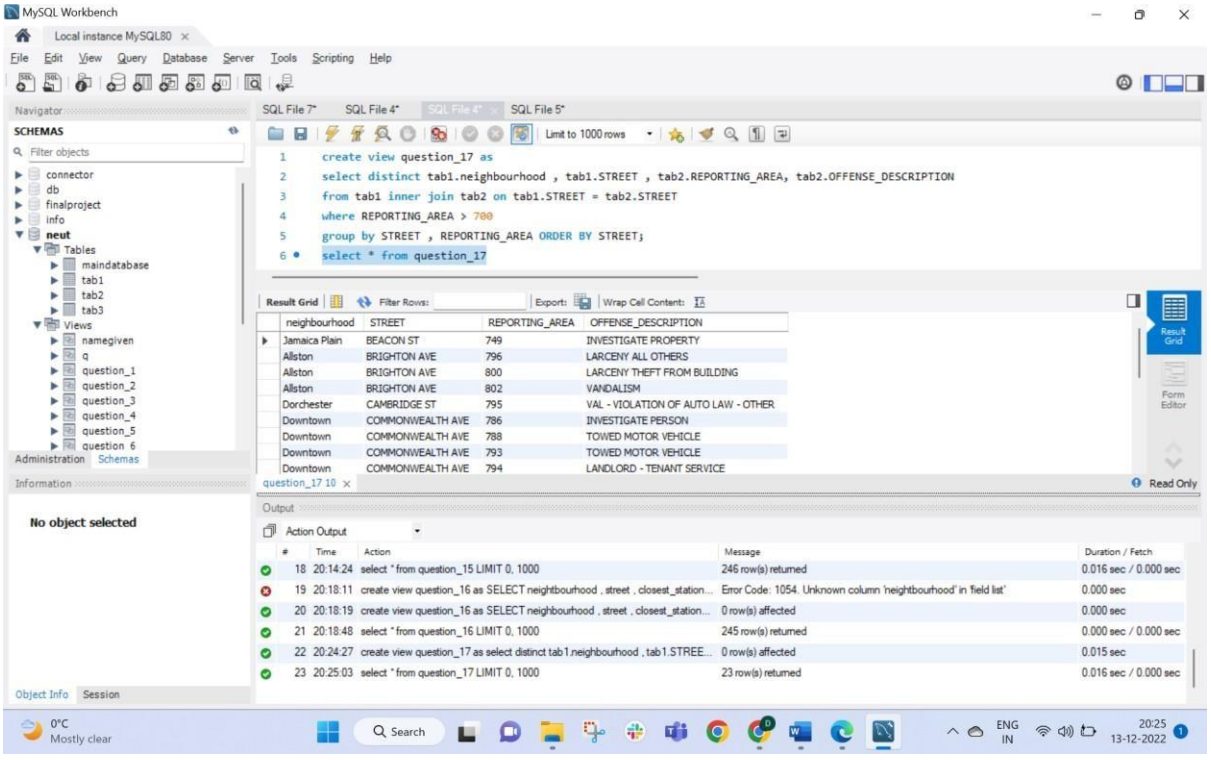
Q16 To find the closest station, distance from a particular neighbourhood and street.



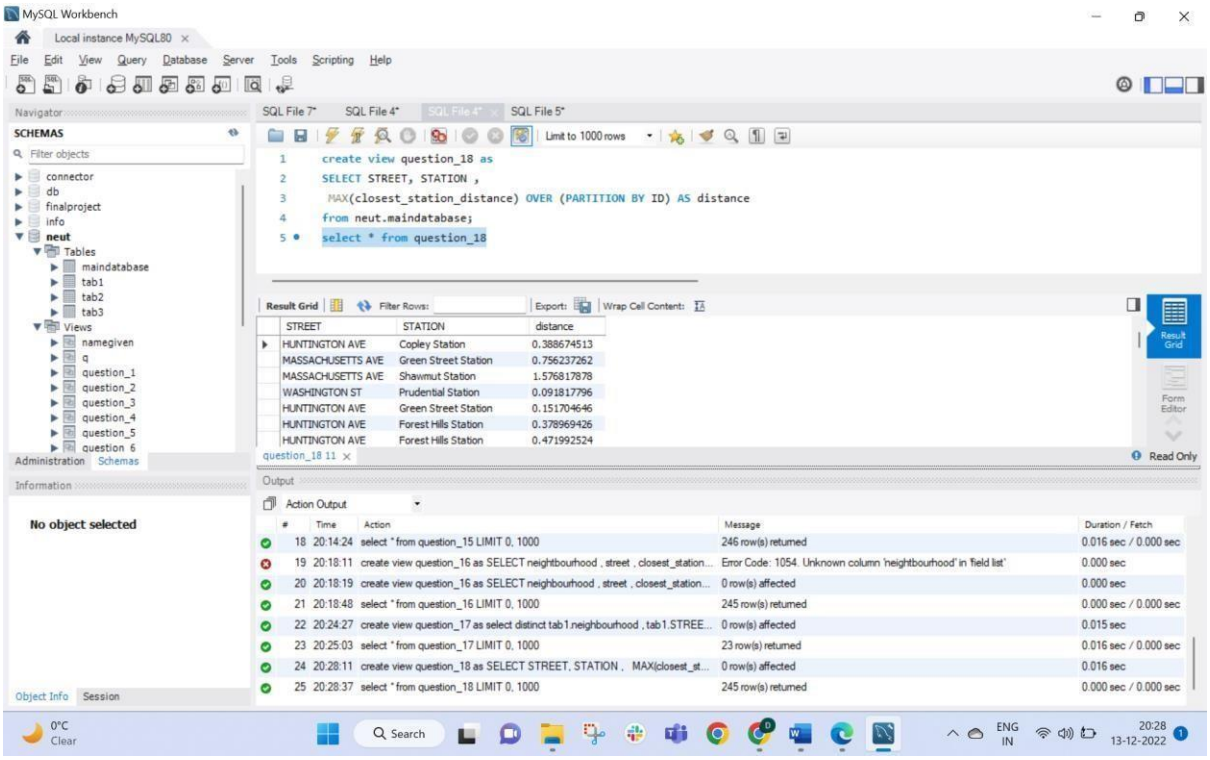
Q17 . We have created a view and given the name (namegiven)



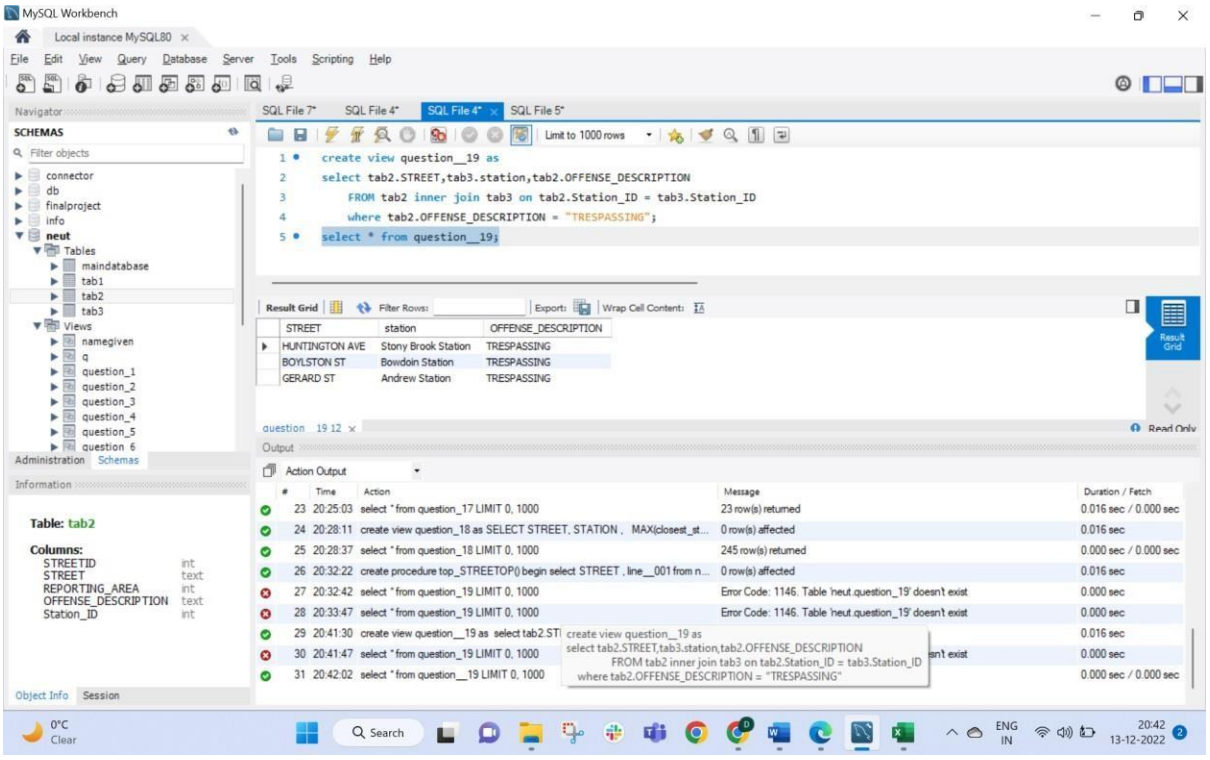
Q18. To find street name, neighbourhood and offense description on the basis on reporting crime



Q19. To find the station which is farest from the street along with the distance using “PARTITION BY” clause

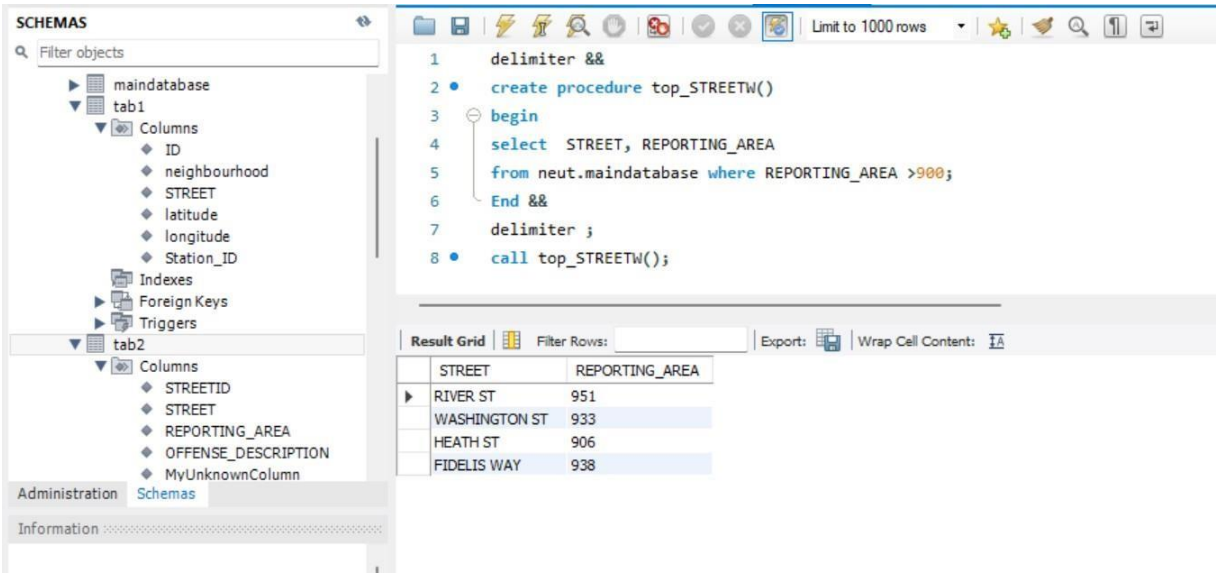


Q20. To find the Neighbourhood name and street name and who has reported the offense of TRESPASSING

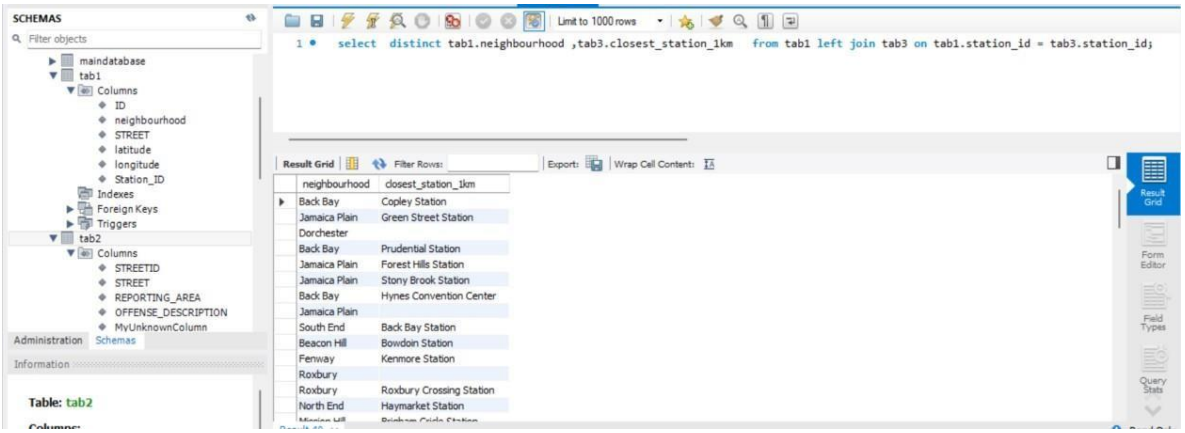


Use Cases:

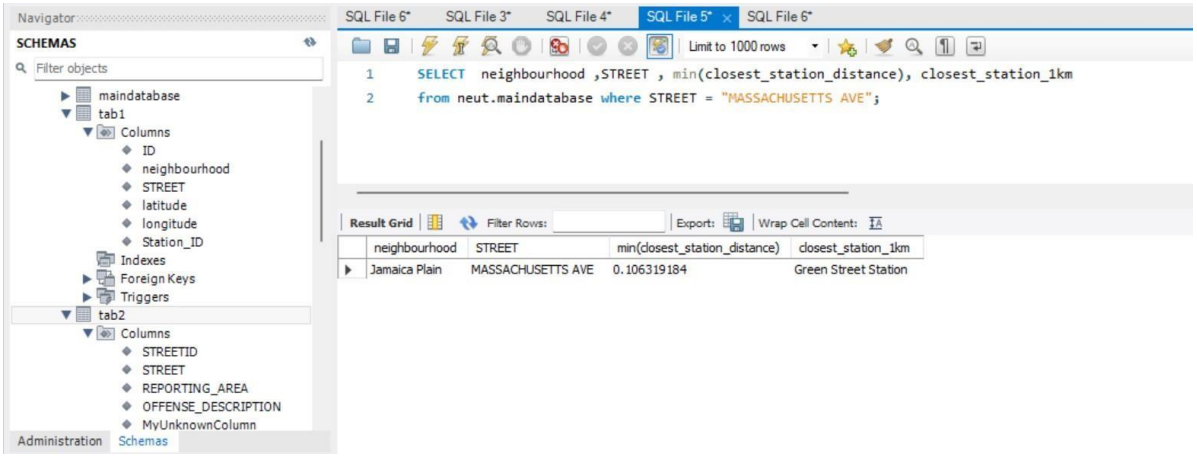
1. Created a store procedure for the crime reports more than 900



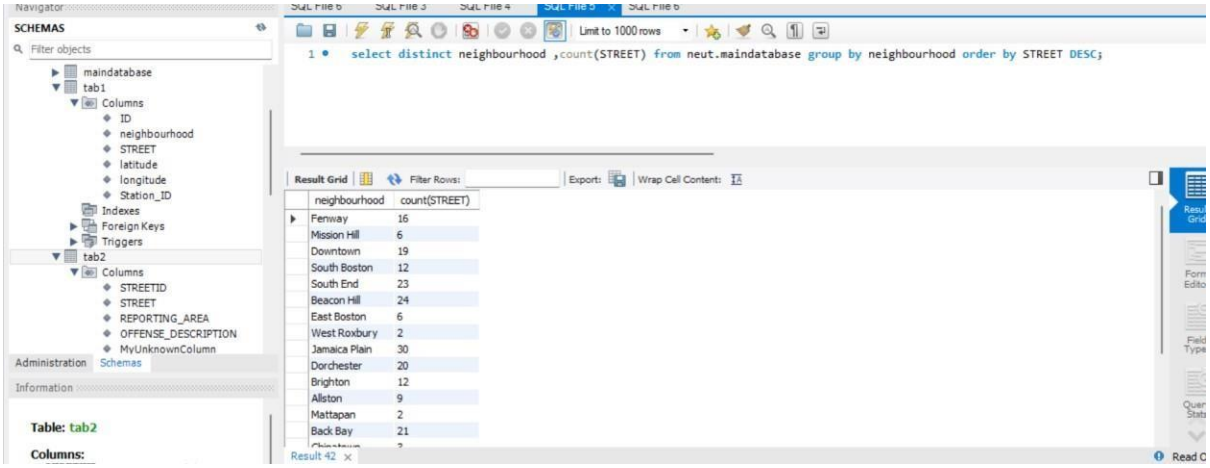
2. Which are the stations near to particular neighbourhood ?



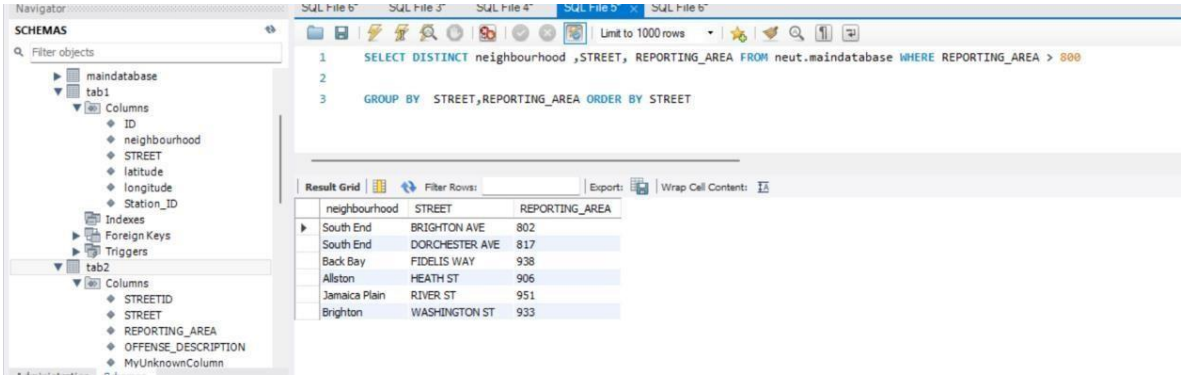
3. To find the nearest station , distance , neighbourhood from the street name MASSACHUSETTS AVE



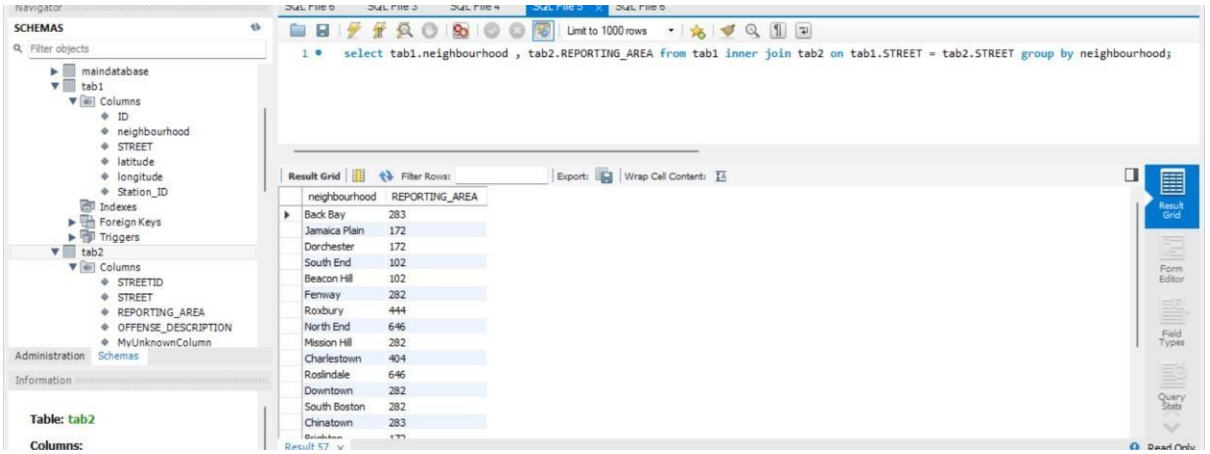
4. To count the number of street in each neighbourhood



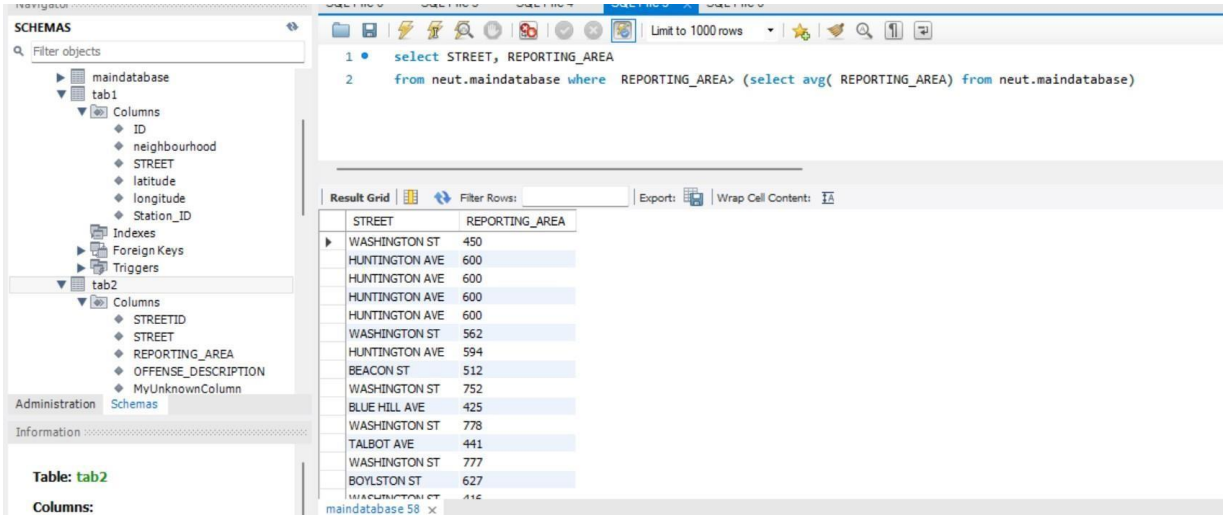
5. To find the street, neighbourhood name where reporting crime counts are more than 800



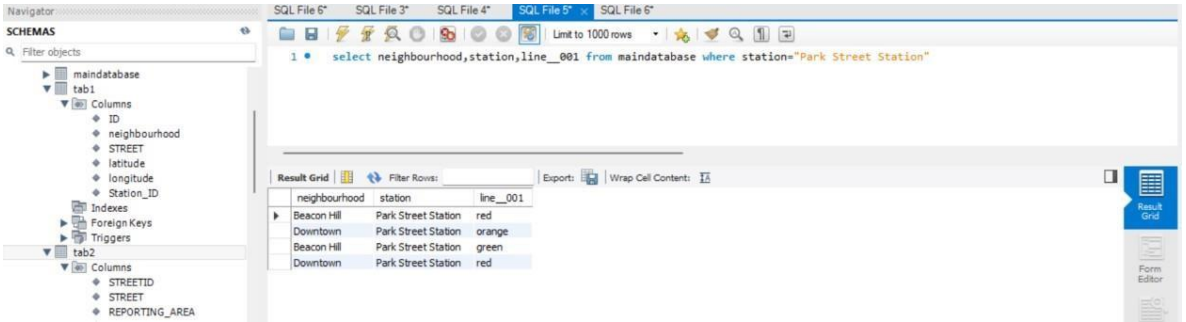
6. What is the count of reporting crime of a particular neighbourhood ?



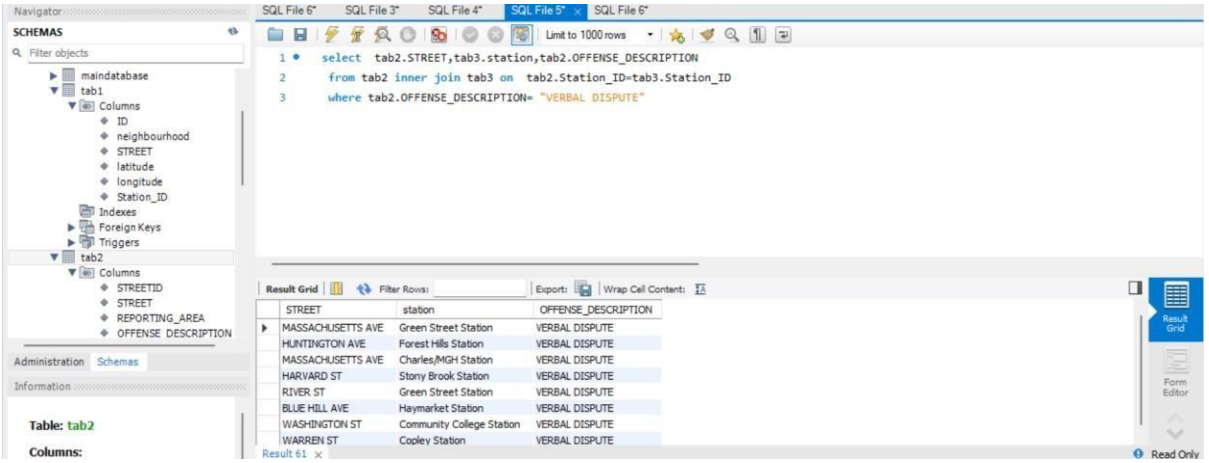
7. To display street name where Reporting crime count more than the avg reporting count



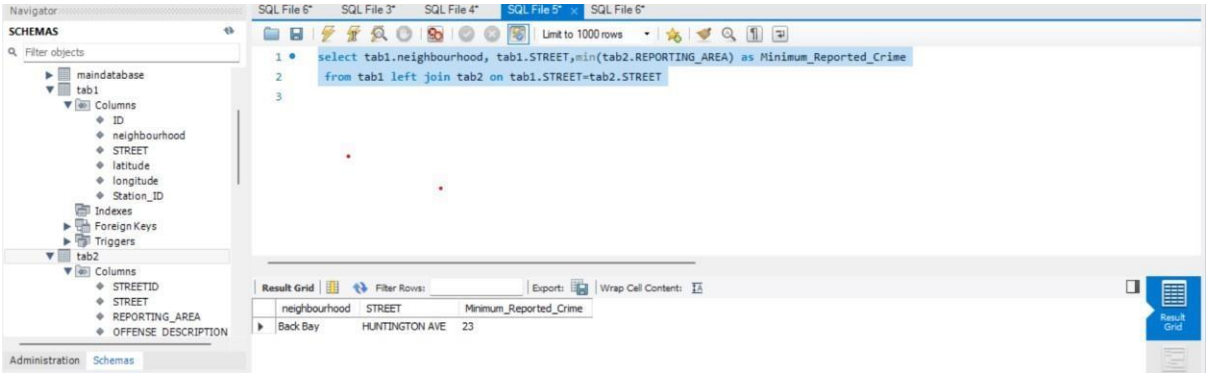
8. To select the station and neighbourhood name which has more than one train connection



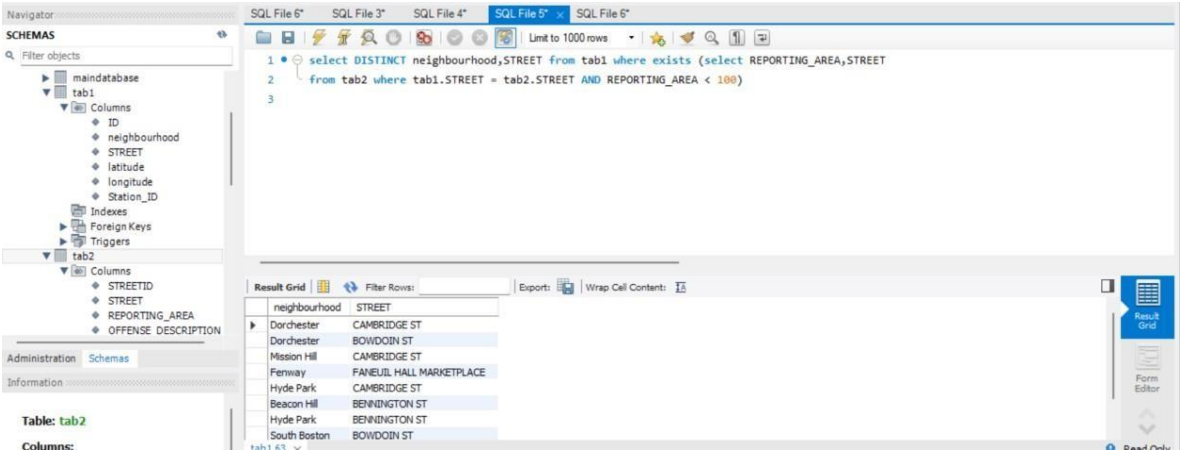
9. To select the street name and station who has committed the offence of Verbal Dispute



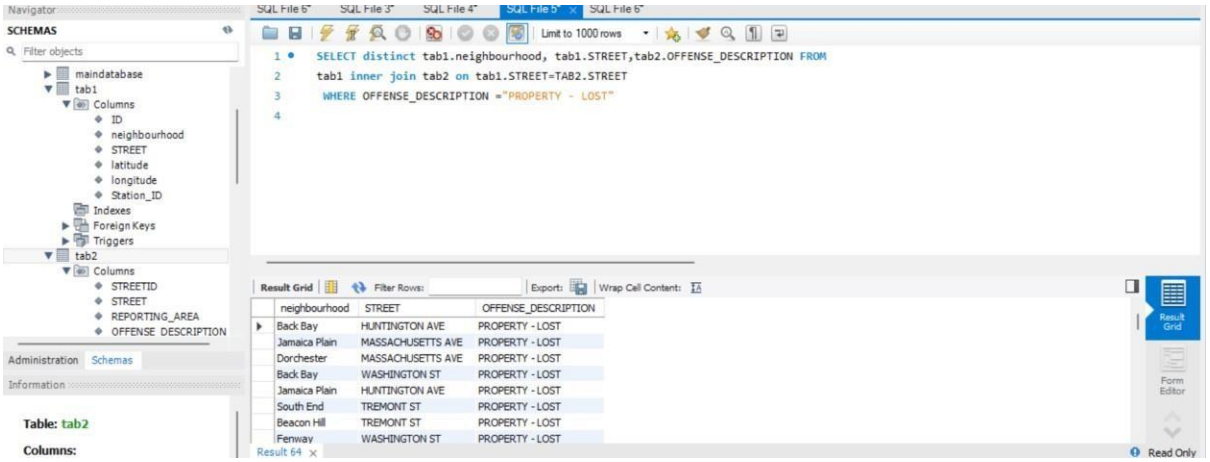
10. To select the neighbourhood, street and reported crime area which has the most minimum reported crime



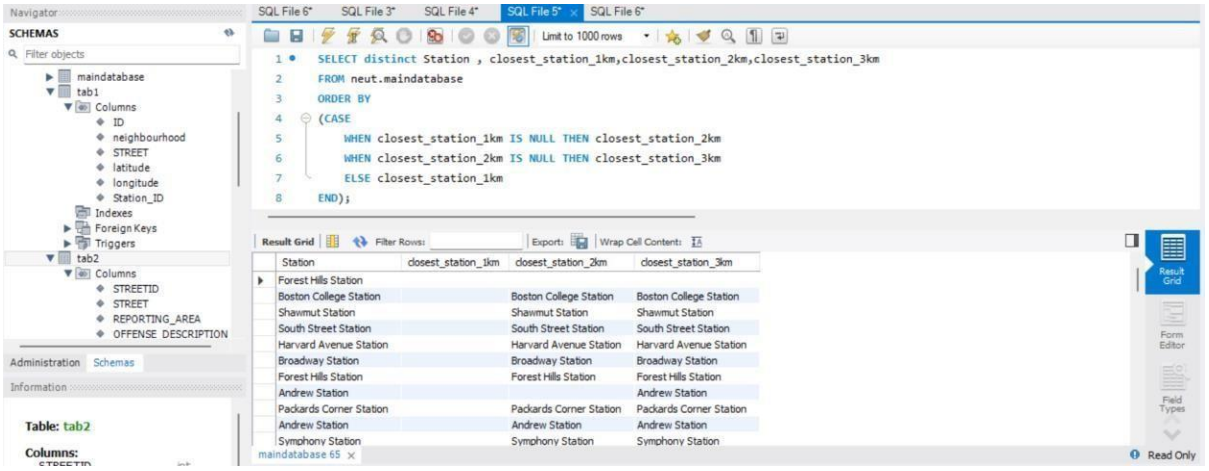
11. To find the neighbourhood name and street name who has reported crime less than 100



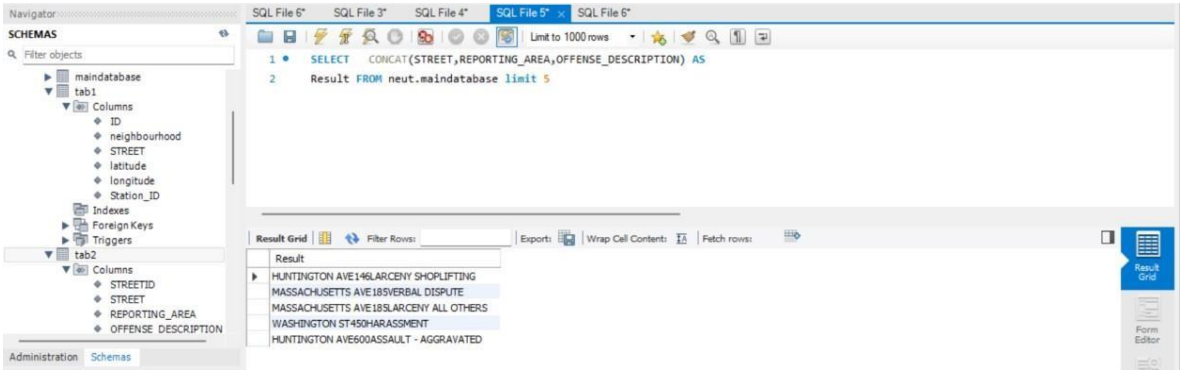
12. To find the Neighbourhood name and street name and who has reported the offense of property lost



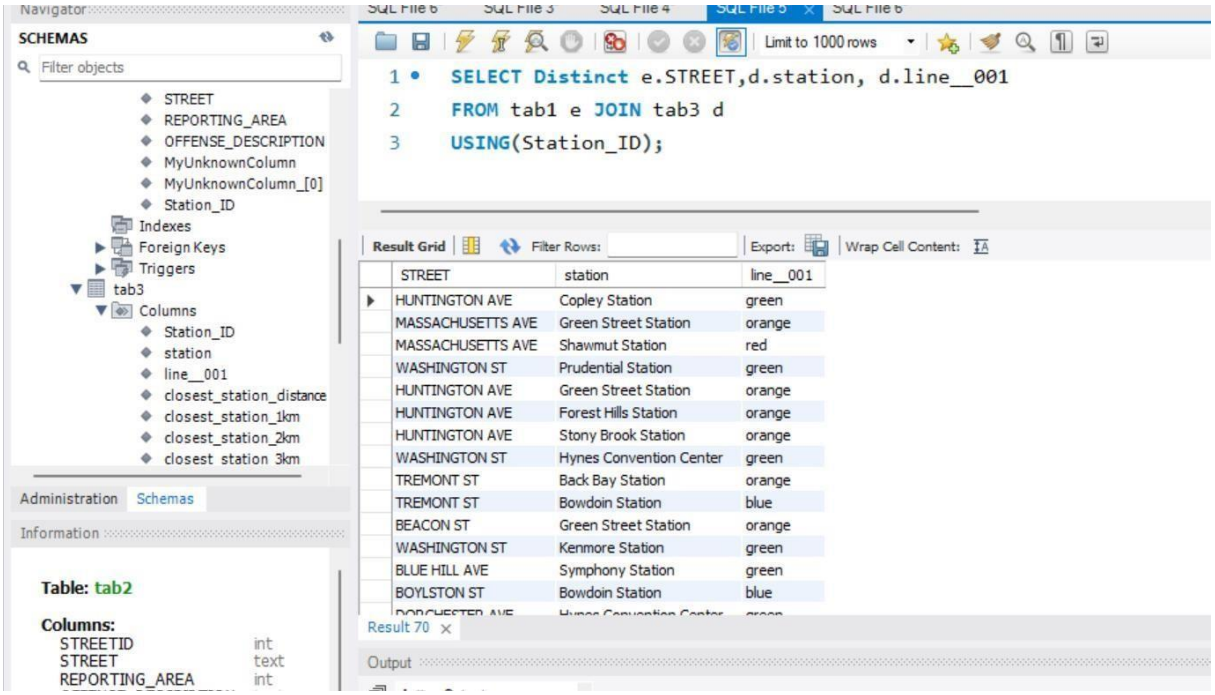
13. Check for the closest station from 1,2,3km using case expression



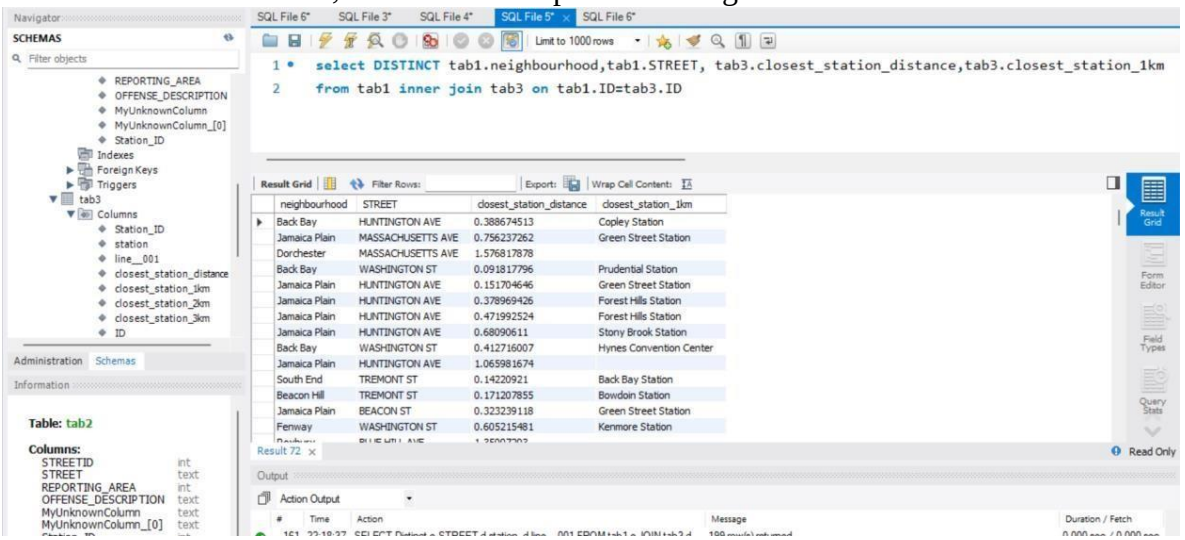
14. To find the recent 5 offense, reported crime of the streets



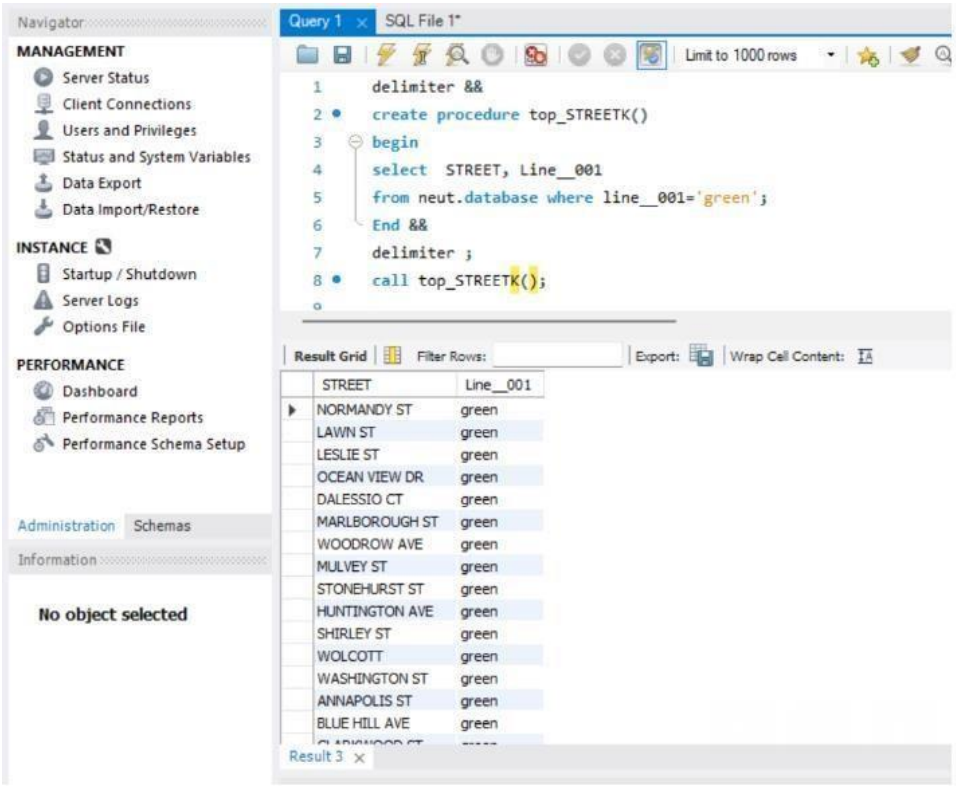
15. To find the station, the train line of the street using “USING Clause”



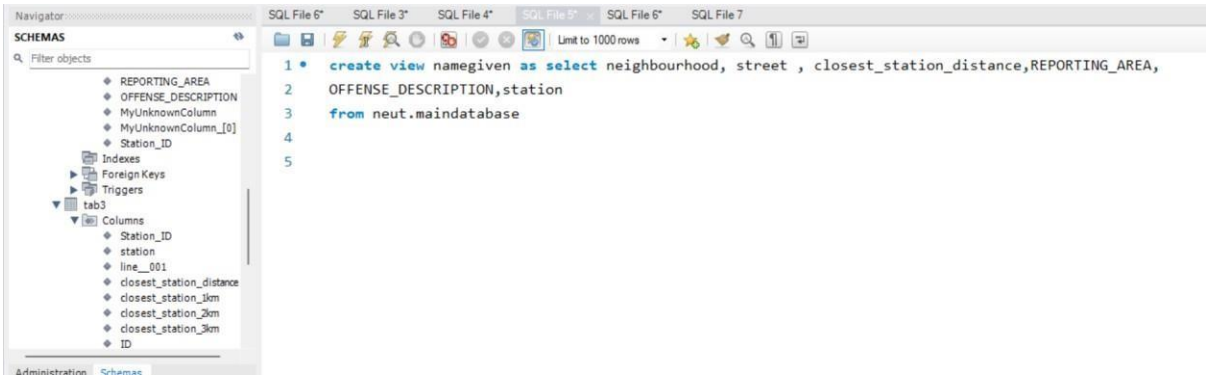
16. To find the closest station, distance from a particular neighbourhood and street.



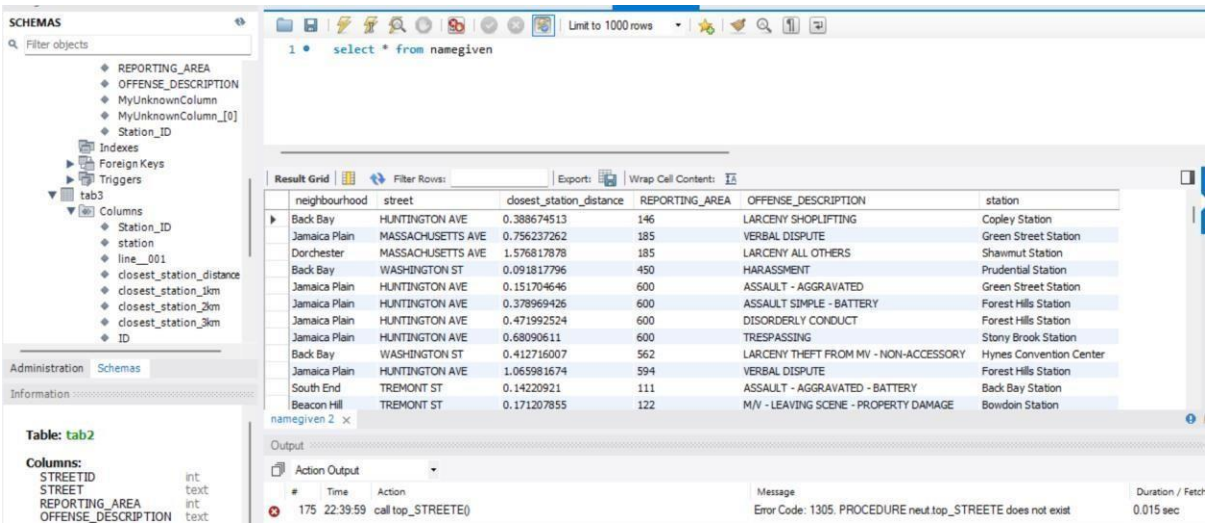
17.To create a stored procedure to show the street which are connected green line



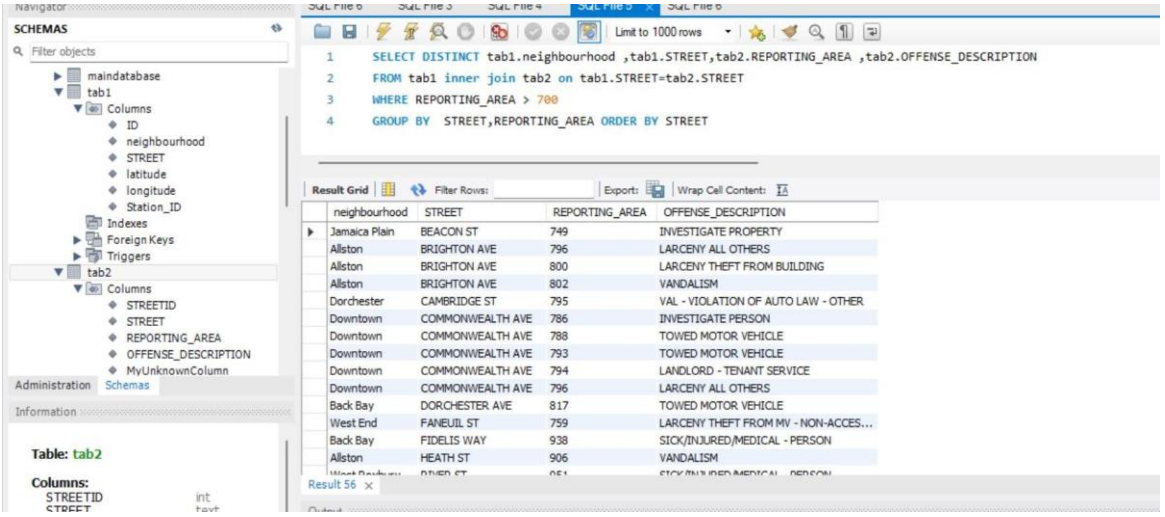
18.We have created a view and given the name (namegiven)



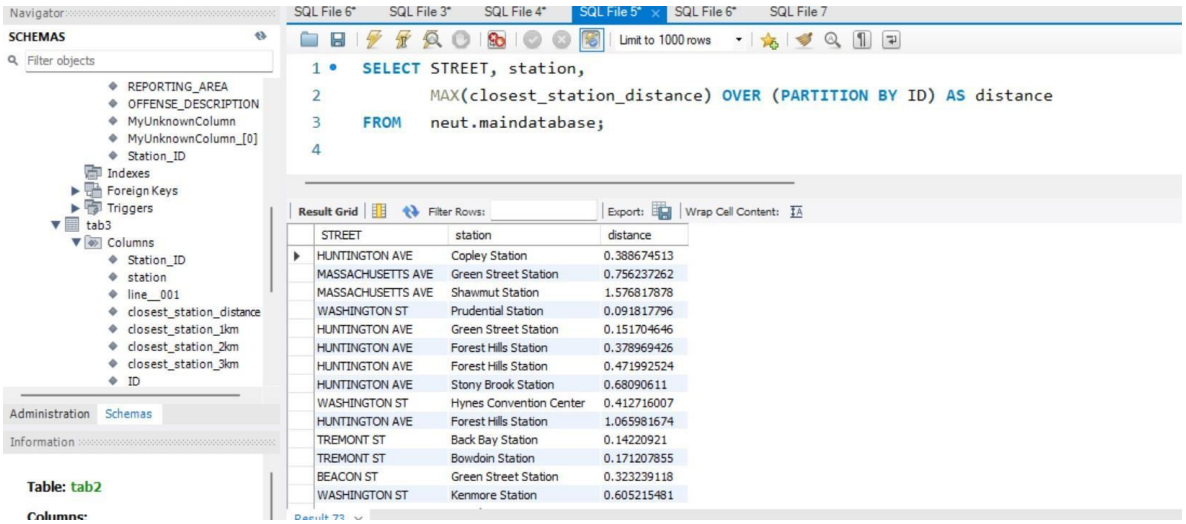
Now we have called the view which is to find the neighbourhood , street , closest station , reporting area offense description and station



19. To find street name, neighbourhood and offense description on the basis on reporting crime



20.To find the station which is farest from the street along with the distance using “PARTITION BY” clause



Steps from choosing dataset to getting the final DB:

Step 1: Searched for datasets related to the topic and finalized 2 datasets and web scraped using Browse AI.

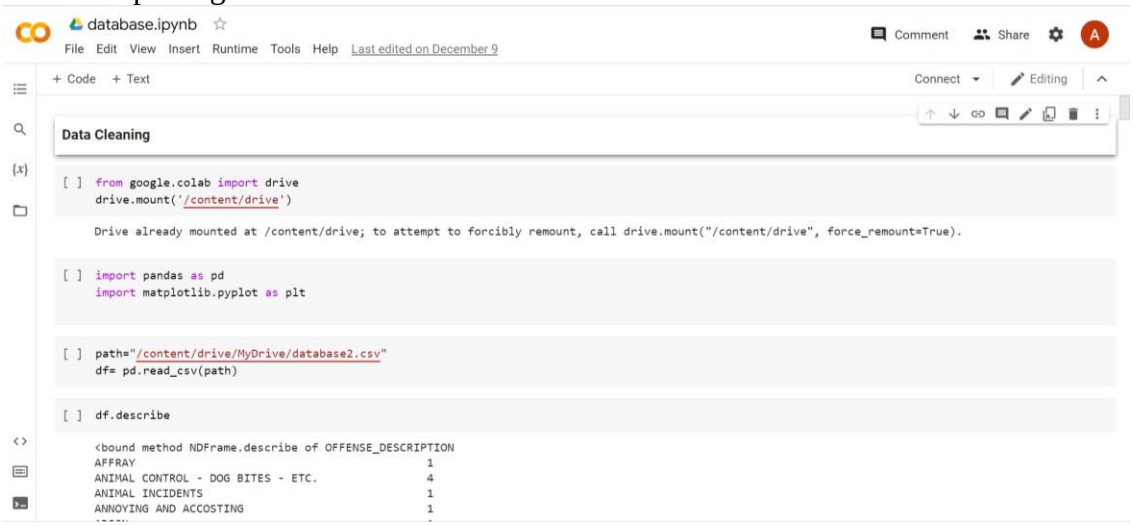
Links for the Datasets are:

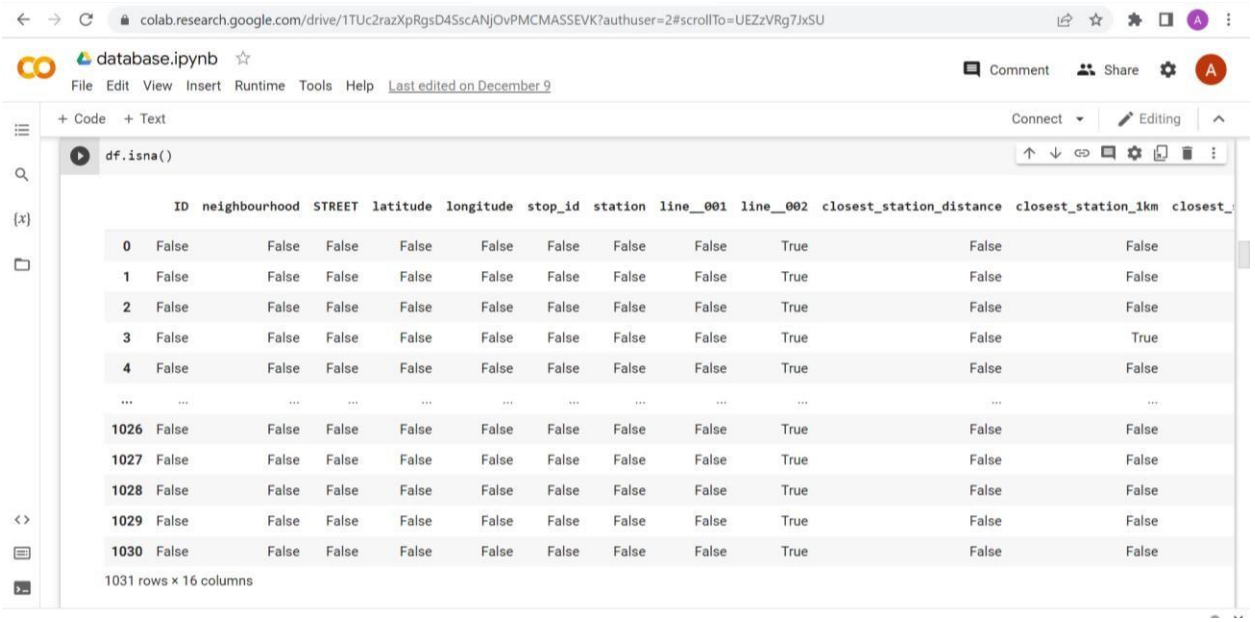
- https://github.com/kanguyn/Interactive-Data-Visualization-MBTA-airbnb/blob/master/data/mbta_metadata.csv
- https://www.kaggle.com/datasets/AnalyzeBoston/crimes-in-boston

Step 2: The datasets were cleaned as it includes null values and duplicate values and they needed to clean for further steps. These steps were done on Google collab.

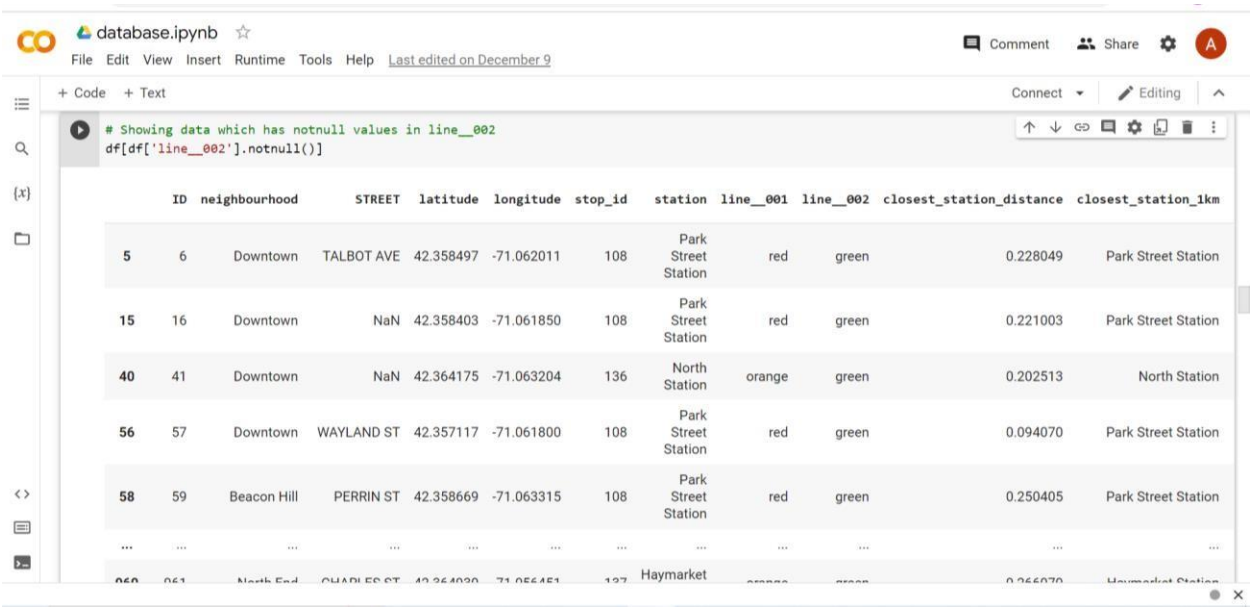
Data Cleaning:

1. Importing the CSV file





2. Finding the data which has no null value in the column line__002



database.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

Comment Share

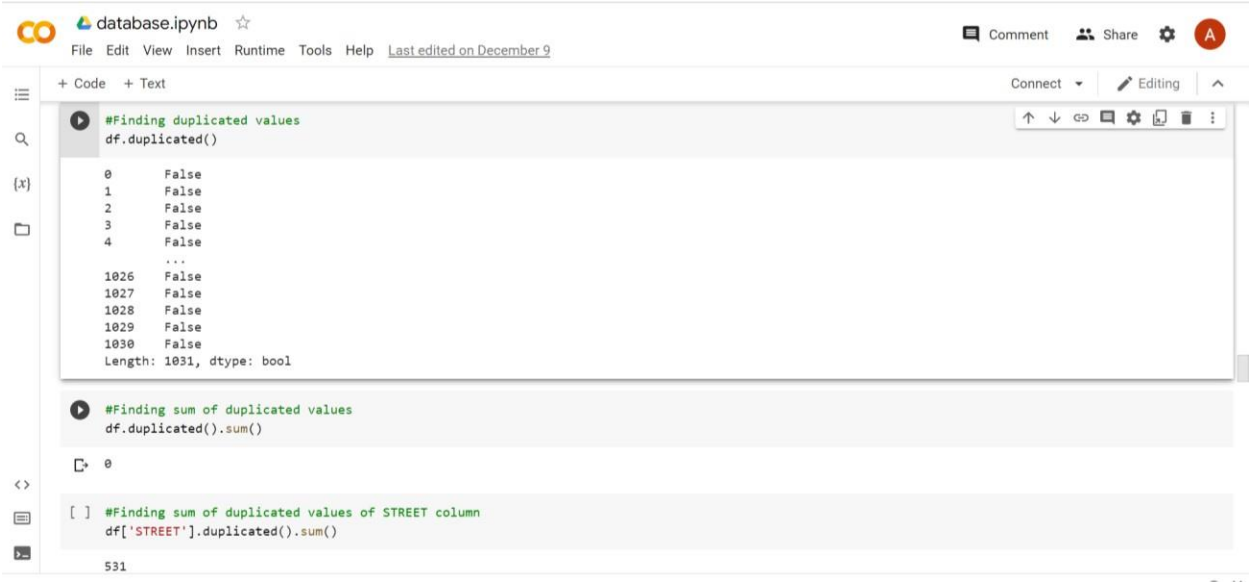
+ Code + Text

Connect Editing

```
# Showing data which has notnull values in line__002
df[df['line__002'].notnull()]
```

	ID	neighbourhood	STREET	latitude	longitude	stop_id	station	line__001	line__002	closest_station_distance	closest_station_1km
5	6	Downtown	TALBOT AVE	42.358497	-71.062011	108	Park Street Station	red	green	0.228049	Park Street Station
15	16	Downtown	NaN	42.358403	-71.061850	108	Park Street Station	red	green	0.221003	Park Street Station
40	41	Downtown	NaN	42.364175	-71.063204	136	North Station	orange	green	0.202513	North Station
56	57	Downtown	WAYLAND ST	42.357117	-71.061800	108	Park Street Station	red	green	0.094070	Park Street Station
58	59	Beacon Hill	PERRIN ST	42.358669	-71.063315	108	Park Street Station	red	green	0.250405	Park Street Station
...	...	...	...	...	...	...	...	...	...	...	...
960	961	North End	CHARLES ST	42.364030	-71.056451	137	Haymarket	orange	green	0.266070	Haymarket Station

3. Finding the duplicated values in the dataset



database.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

Comment Share

+ Code + Text

Connect Editing

```
#Finding duplicated values
df.duplicated()
```

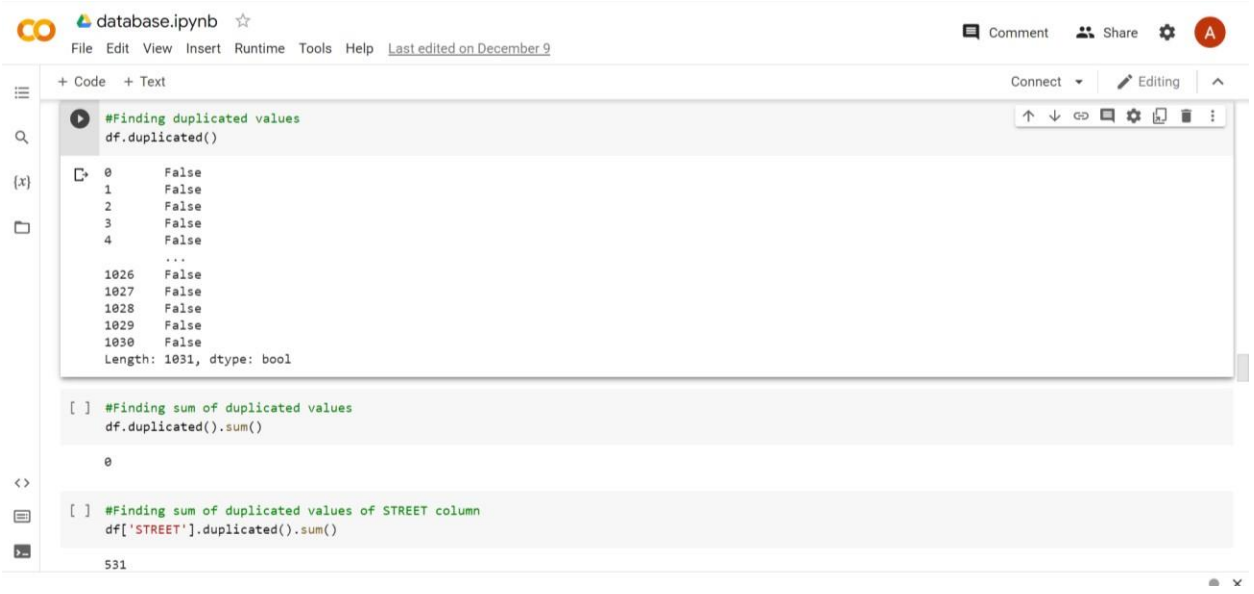
```
0       False
1       False
2       False
3       False
4       False
        ...
1026    False
1027    False
1028    False
1029    False
1030    False
Length: 1031, dtype: bool
```

```
#Finding sum of duplicated values
df.duplicated().sum()
```

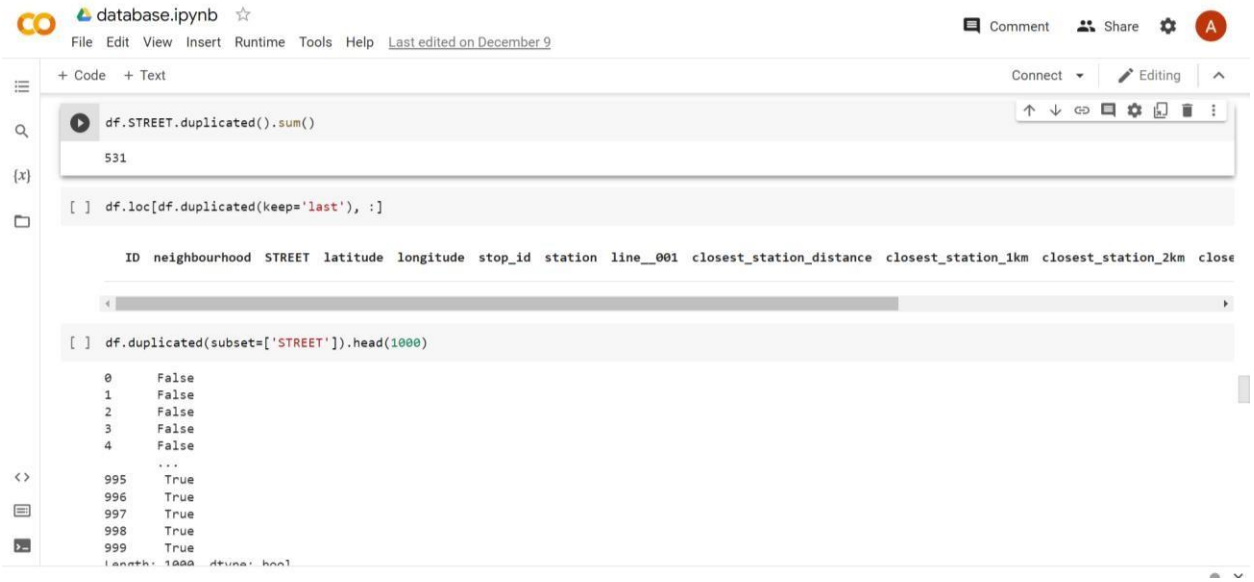
```
0
```

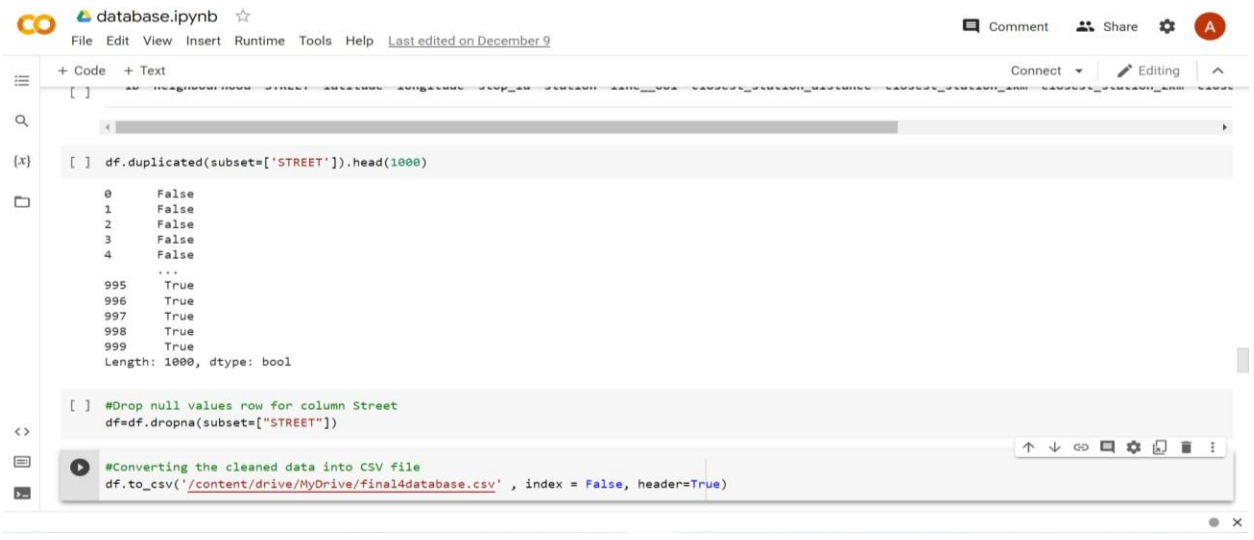
```
[ ] #Finding sum of duplicated values of STREET column
df['STREET'].duplicated().sum()
```

```
531
```



4. Finding the duplicated values of column STREET and removing the duplicated value





Step 3: The cleaned dataset was then further converted into a CSV file named “maindatabase”.

Step 4: According to the dataset and creation of tables, ER diagram was prepared

Step 5: Further the cleaned data CSV file was imported into SQL Workbench

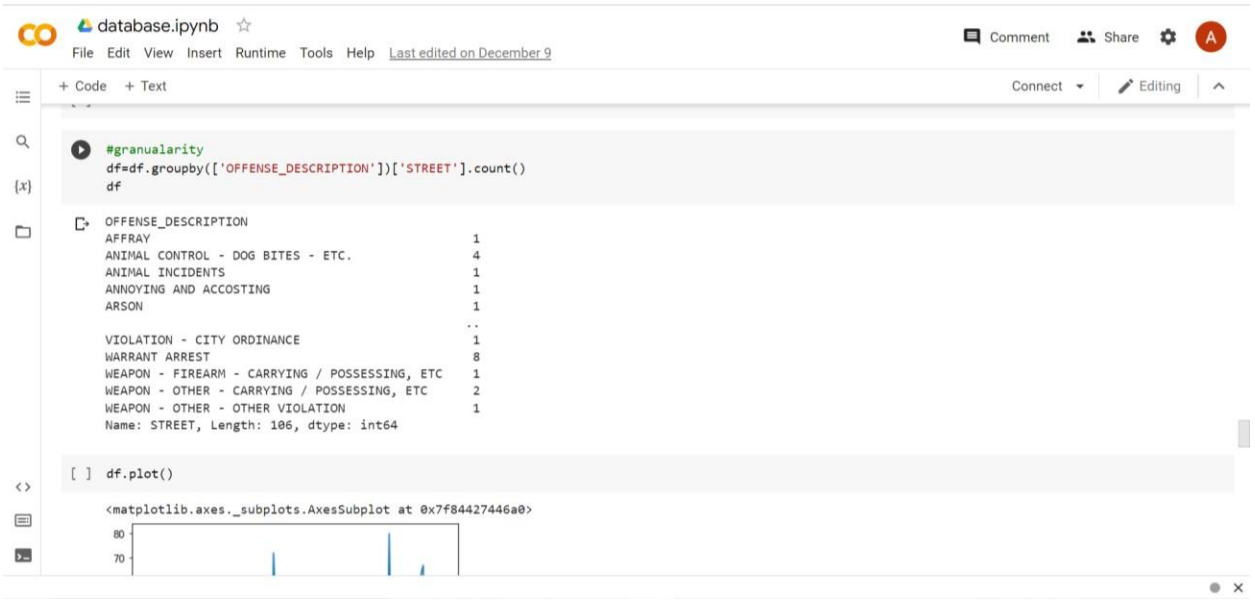
Step 6: The 20 sub-queries were implemented on SQL Workbench

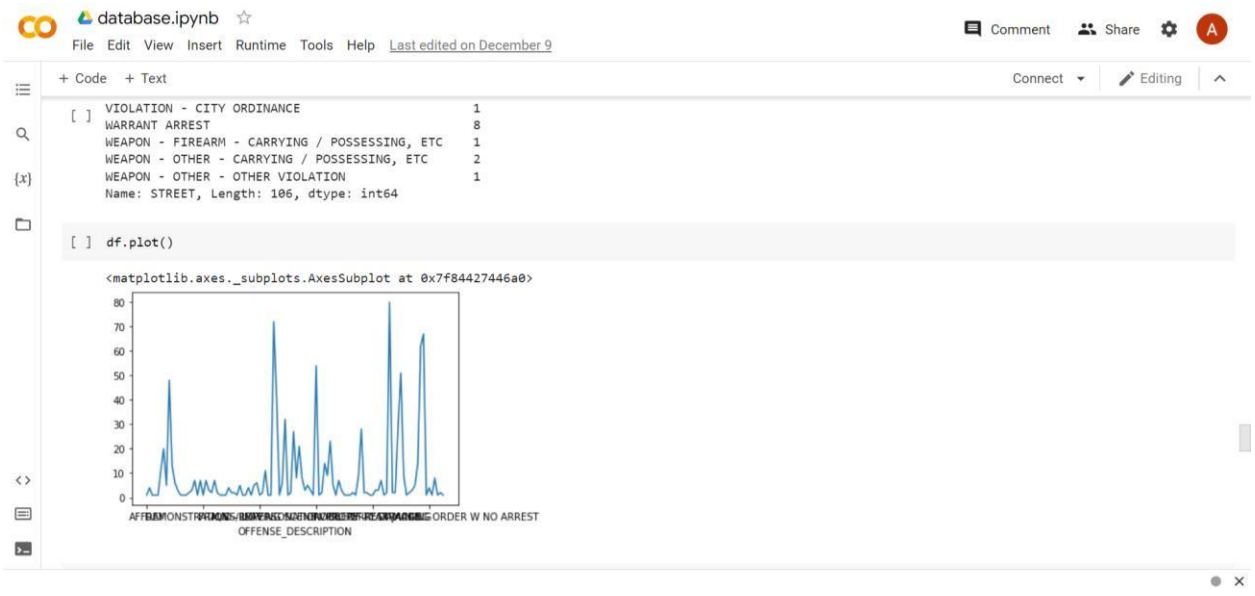
Step 7: After completing the queries, data visualization was implemented on the final dataset on Google Collab

Step 8: Granularity, Bar graphs, and Histograms have been produced on the basis of the final dataset

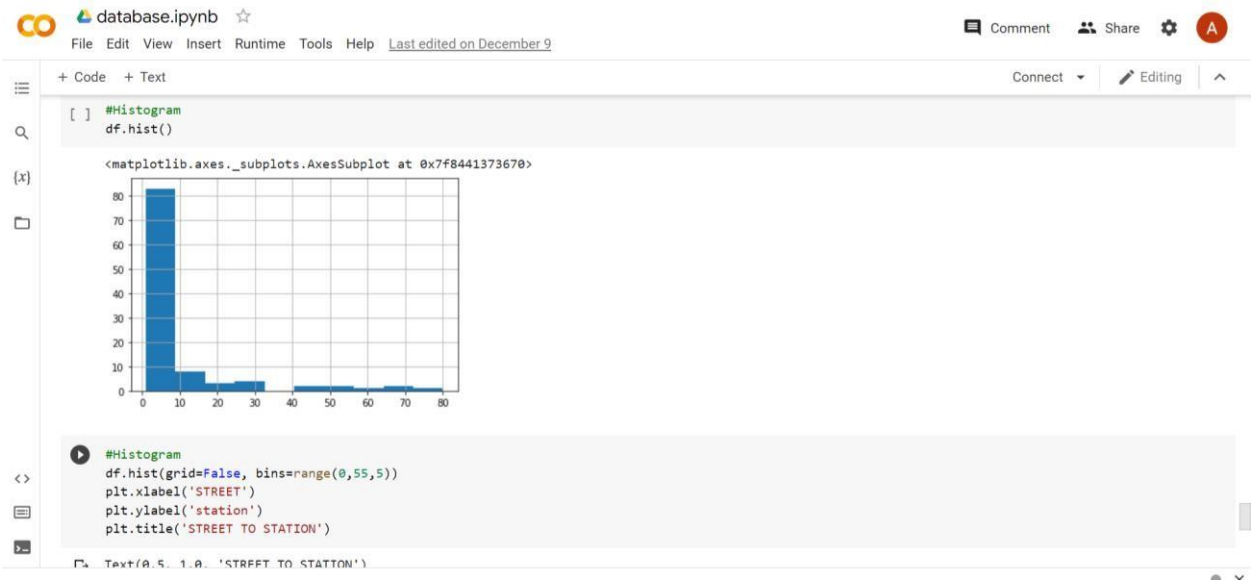
Data Visualization:

1. **Granuality**





2. Histogram



3. Histogram

Step 9: After data visualization, we have performed Data Normalization.

Data Normalization:

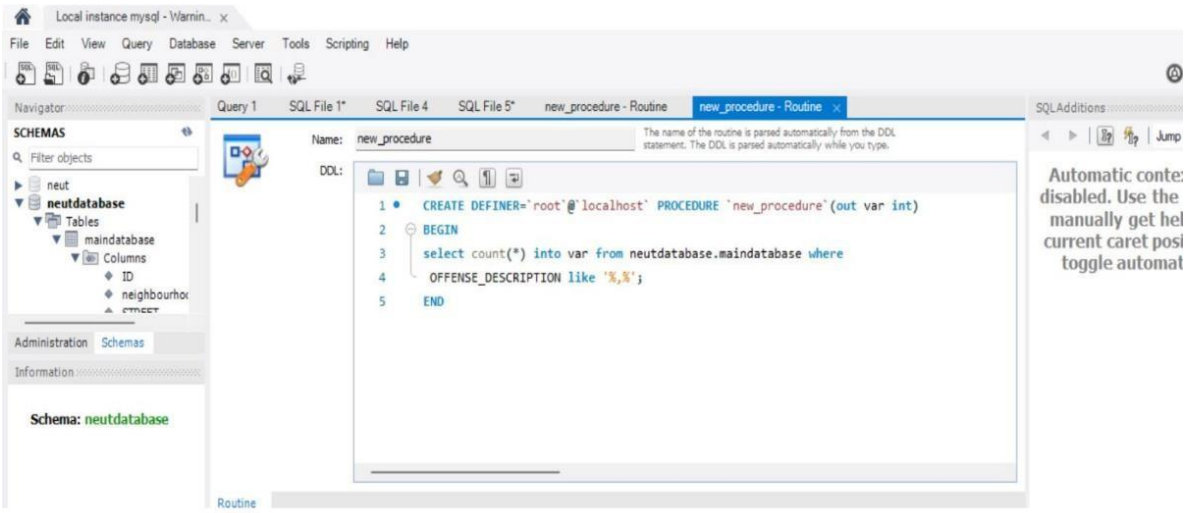
In the below process, we have checked whether our tables are in 1NF, 2NF and 3NF.

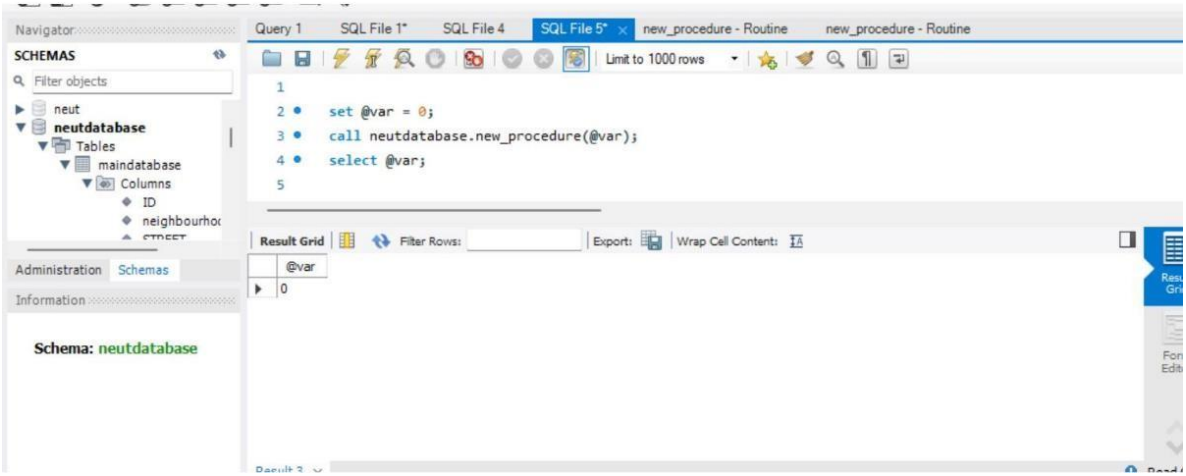
1NF:

Here, in order to check whether our table is in 1NF or not we have perfomed the following steps –

Step 1 – We have created a stored procedure named “new

_procedure” in order to check whether the values in each column are atomic and there is no repeating groups.

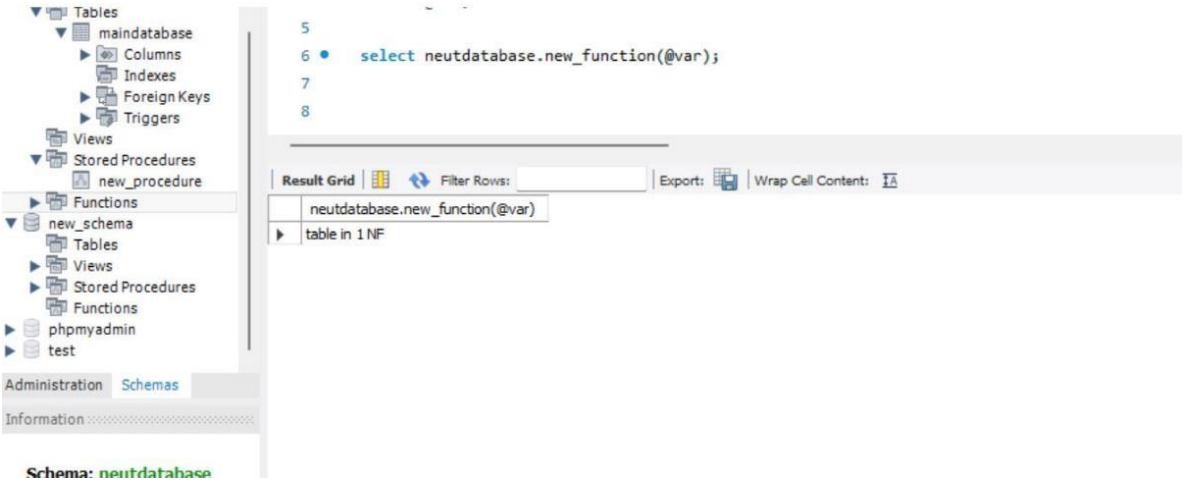




Step 2- Then we have created a function named “new_function” to check and display the result .

DDL:

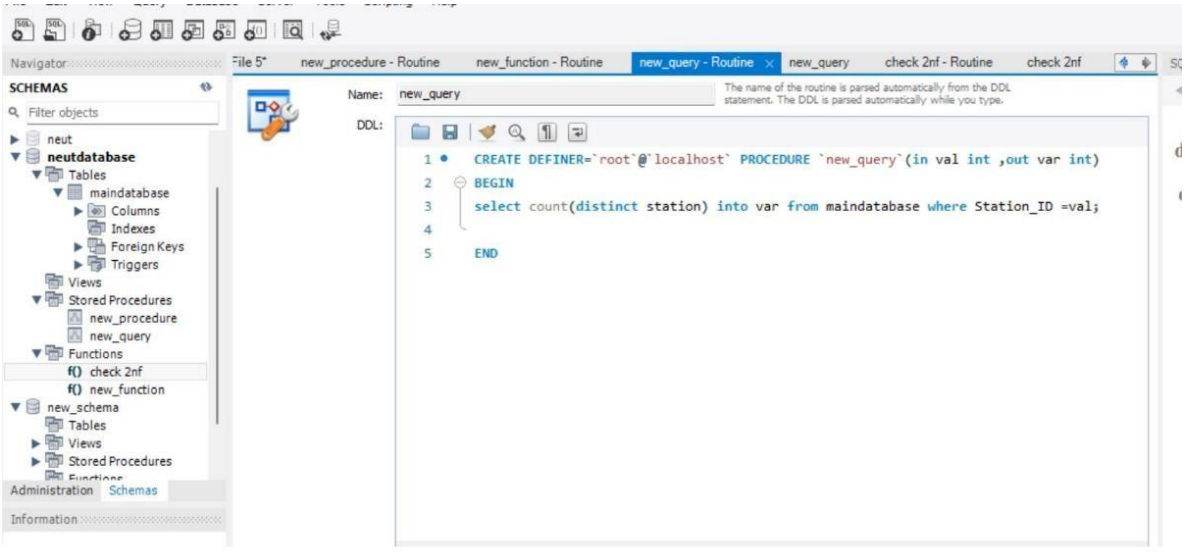
```
CREATE DEFINER=`root`@`localhost` FUNCTION `new_function`(a int) RETURNS varchar(20) C
    READS SQL DATA
    DETERMINISTIC
BEGIN
    declare result varchar(20);
    SET global log_bin_trust_function_creators=1;
    IF a>0 THEN
        SET result = "table is not in 1NF";
    ELSE
        SET result = 'table in 1 NF';
    END IF;
    RETURN result;
END
```

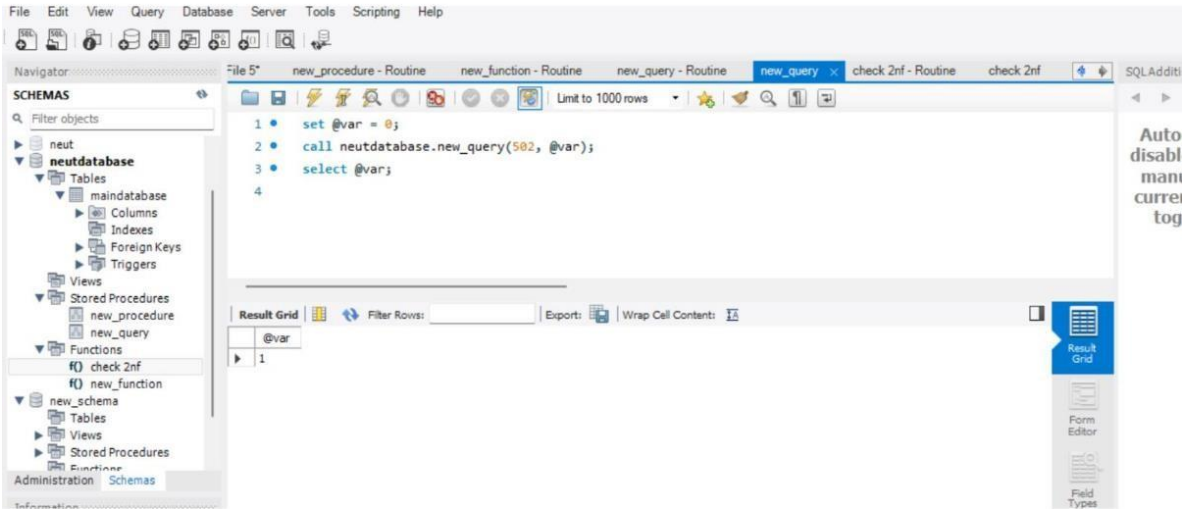


2NF:

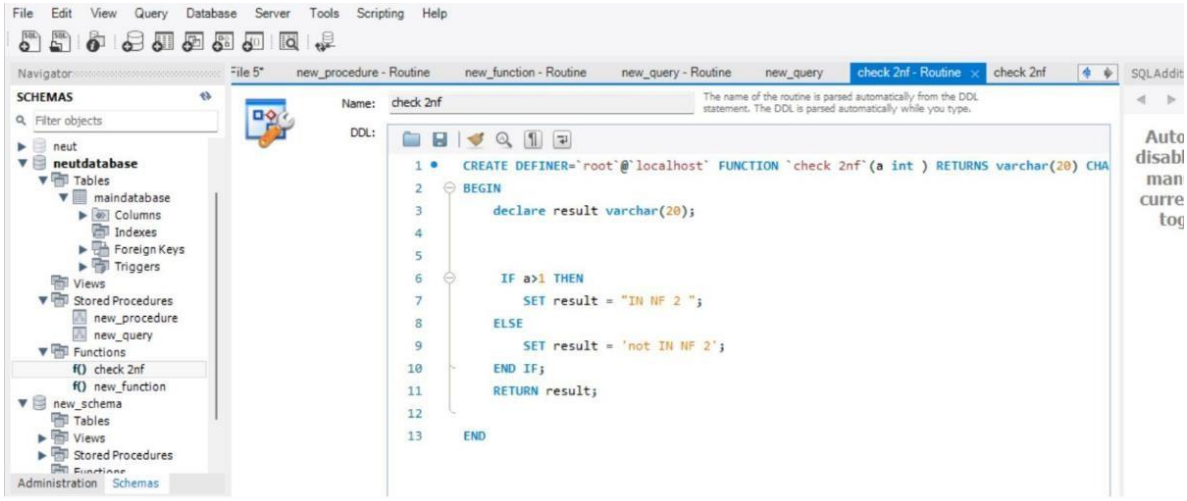
Here, in order to check whether our table is in 2NF or not we have perfomed the following steps –

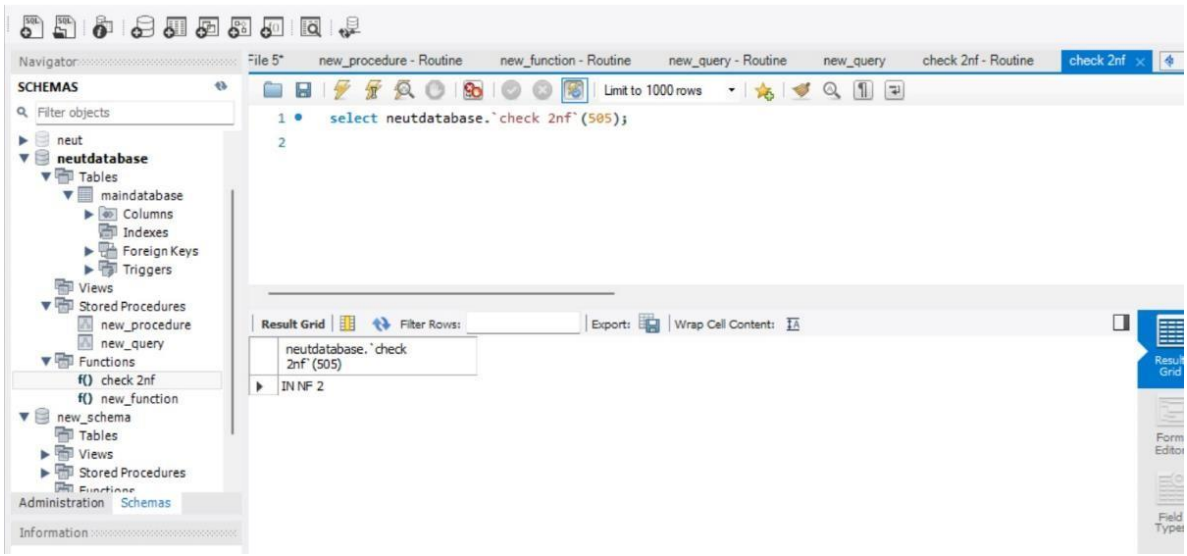
Step 1 – We have created a stored procedure named “new_query” in order to check that there is no partial dependencies.





Step 2- Then we have created a function named “check 2nf” to check and display the result .

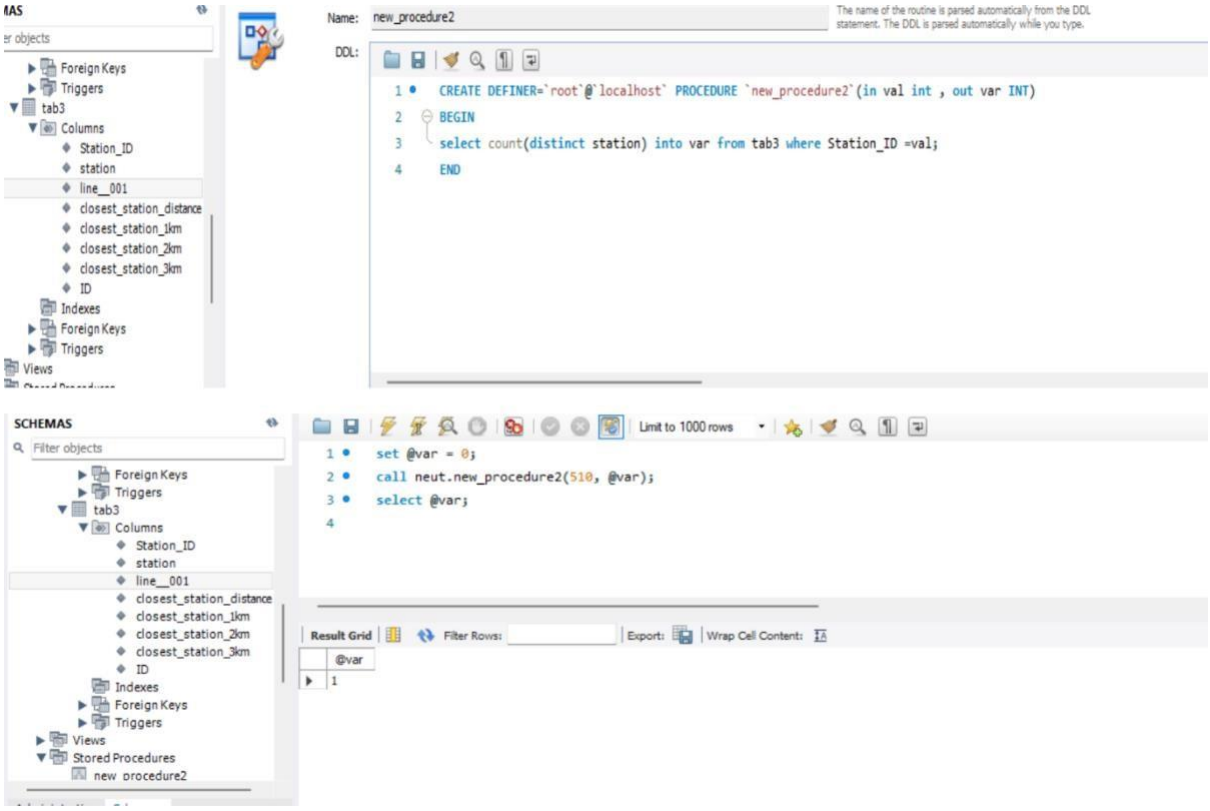




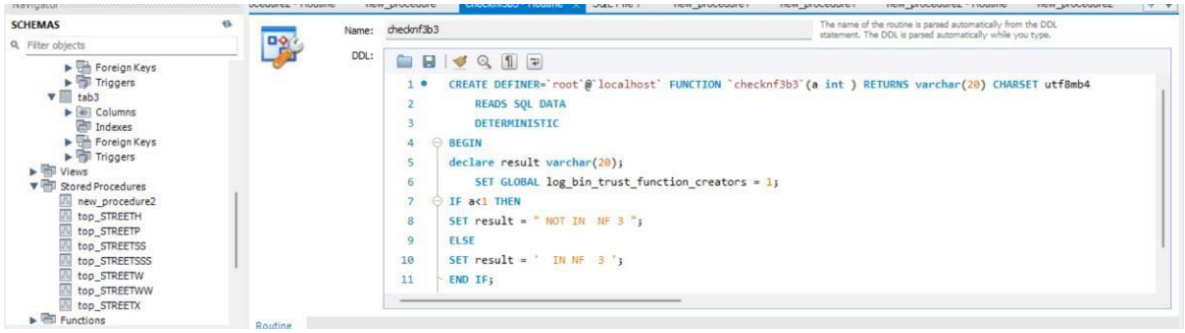
3NF:

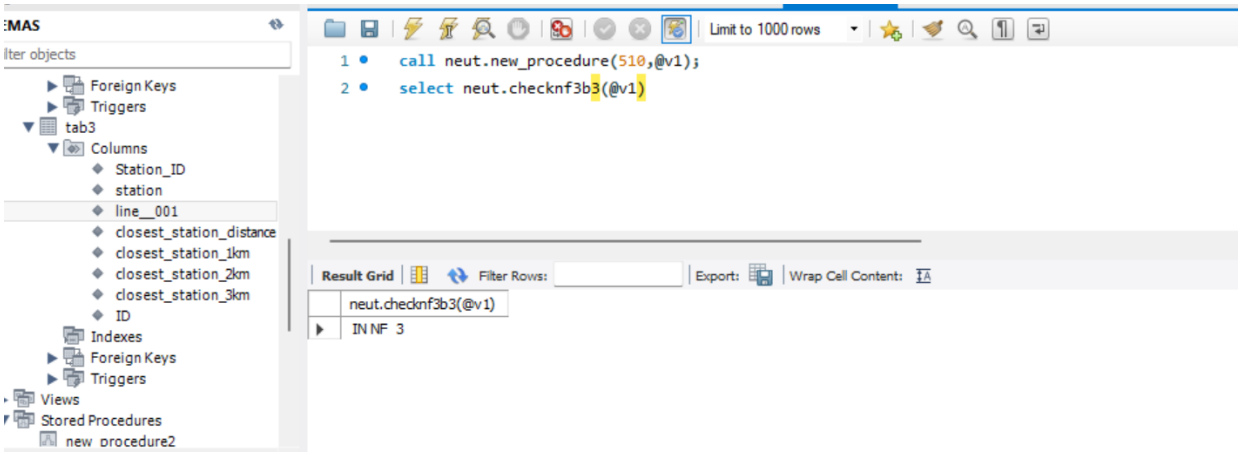
Here, in order to check whether our table is in 3NF or not we have perfomed the following steps –

Step 1 – We have created a stored procedure named “new _procedure2” in order to check that there is no transitive dependencies.



Step 2- Then we have created a function named “checknf3b3” to check and display the result .





Team Number- 9

Team Members Description:

1. Name: Deepak Sawalka

NU-ID: 002910793

NU Email-ID: sawalka.d@northeastern.edu

GitHub-ID: DeepakSawalka

2. Name: Krishana Kapadia

NU-ID: 002762329

NU Email-ID: kapadia.kr@northeastern.edu

GitHub-ID: krishnakapadia29

3. Name: Janhvi Shah

NU-ID: 002783375

NU Email-ID: shah.janh@northeastern.edu

GitHub-ID: janhvishah

4. Name: Anjali Kabra

NU-ID: 002762324

NU Email-ID: kabra.an@northeastern.edu

GitHub-ID: anjalikabra18