# Improving Pittsburgh Bus Transit

Exploring Pittsburgh's Bus Transit to Reduce Traffic Congestion and Increase Ridership

Rami Ariss
Civil Engineering
Carnegie Mellon University
Pittsburgh, PA USA
rariss@andrew.cmu.edu

Rhea Upadhyay
Civil Engineering
Carnegie Mellon University
Pittsburgh, PA USA
rheau@andrew.cmu.edu

Chirag Sachdeva
Civil Engineering
Carnegie Mellon University
Pittsburgh, PA USA
csachdev@andrew.cmu.edu

## MOTIVATION

As urbanization establishes dense population centers in cities, public transportation systems are critical to mobilizing citizens, factoring into the quality of life, urban sprawl, and environmental impact. Most students depend on low cost public transportation, and as students we are inspired to question ways in which the built environment around us can be improved. Living in Pittsburgh, we want to explore ways in which we can utilize available data to make recommendations on how to improve Pittsburgh's Bus Transit system. Generating a set of insights can inform policy or transportation infrastructure investments to target segments of the bus transportation system that can reduce traffic congestion and increase ridership, improving the standard of living and reducing the environmental impact of Pittsburghers.

## DATA SOURCES

Rich, publicly available data sources are provided by Allegheny County to profile socio-demographics, bus transit stops and usage performance metrics, and vehicle traffic counts. The primary data sources we will be using are:

1. (Pittsburgh SNAP Census Data, 2010): Provides census data for Pittsburgh, including population, income, education, and demographics by neighborhoods.
2. (Port Authority Transit Stop Usage, 2019): Port Authority bus transit stops with latitude/longitude and average boardings and alightings.
3. (Port Authority Monthly Average Ridership by Route, 2019): Port Authority bus routes with average weekday, Saturday, and Sunday ridership per month.
4. (Port Authority Monthly On-Time Performance by Route, 2019): Port Authority bus routes with on-time performance (OTP) by weekday, Saturday, and Sunday per month.
5. (Traffic Counts 2012-2014 Data, 2014): 1,200 traffic sensors deployed across Allegheny County providing aggregate hourly average vehicle counts. Data collected from 2012-2014 by CMU's Traffic21 Institute.

## OBJECTIVES

We aim to propose ways in which Pittsburgh's bus transit system can be improved to reduce vehicle traffic congestion in Allegheny county and to increase bus transit ridership. This objective will be achieved by:

1. Designing a database that can store disparate data sources, including census data, traffic count data, and bus stop data, to extract useful bus transit insights;
2. Use geospatial analytics to create relations between neighborhoods, bus stops, and traffic sensors;
3. Utilize SQL queries to identify bus stops whose improvement can most benefit Pittsburgh by improving ridership and/or reducing traffic;
4. Generate observations and visualize trends that provide insight into the bus stops that should be prioritized for improvement given sociodemographic information;
5. Demonstrate a mySQL database front-end interface using Tableau that allows users to drill into further insights on why bus stops may be underutilized, or underperforming given sociodemographic data and traffic count information.

## SUMMARY OF TASKS PERFORMED

All objectives were completed and achieved by the following ordered tasks:

**1) Data Collection:** We identified and consolidated all raw data sources from the Western Pennsylvania Regional Data Center, including data from the Pittsburgh Port Authority, Pittsburgh SNAP census, and CMU's Traffic21 Traffic Counts study. These data sources are all disjointed and unconnected on the Western Pennsylvania Regional Data Center website.

**2) Entity-Relationship Diagram (ERD):** After reviewing the available data and data dictionaries from the raw data, we generated an ERD that primarily focused on defining the key entities and relationship associations, as well as key attribute

groups we desired from the raw data sources. In the ERD design, we considered which data sources may be repeated year over year.

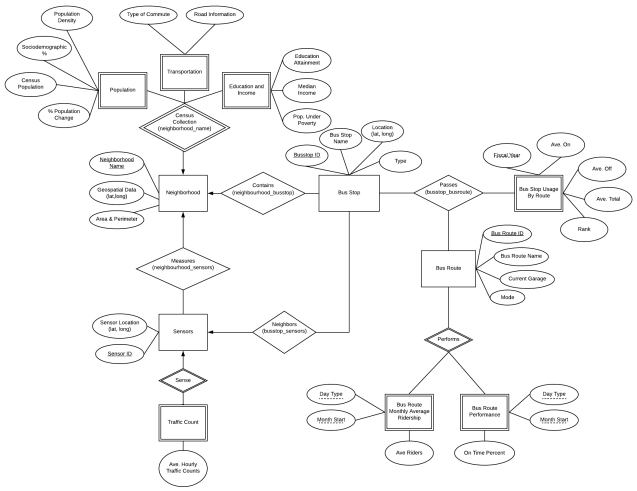



**Figure 1 - ERD associating SNAP neighborhood census, Traffic21 traffic count study, and Port Authority bus stop and bus route metadata and performance data.**

In this design, the SNAP census data is intended to hold just the 2010 census results because a future census may have different data collected and thus warrant different entities. Likewise, the traffic counts study was likely a onetime study. These two cases justify the one-to-many relationships between the respective weak-entity relationships.

On the other hand, the Port Authority regularly collects annual data on bus stop usage and performance. Additionally, bus stops and bus routes change regularly over time. This justifies the many-to-many relationships and conjugate keys for weak entities.

**3) Geospatial Analysis:** Using geopandas in python, we mapped relations between neighborhoods, bus stops, and traffic sensors. The desired maps are: bus stop IDs to traffic count sensor IDs; traffic count sensor IDs to neighborhood names; and bus stop IDs to neighborhood names. This was accomplished by:

- Importing bus route, traffic sensor, and neighborhood data through geopandas using coordinate reference system (CRS) EPSG:4326 (spherical; degrees).
- Transforming bus stop and traffic sensor geospatial data from the spherical projection to a mercator projection CEA to calculate Euclidian distances in meters.
- Creating a function that iterates through bus stop IDs, calculates Euclidian distances from each traffic sensor location, determines the traffic sensor with the shortest Euclidian distance, and returns the nearest traffic count sensor ID. This mapped all bus stop IDs to traffic count sensor IDs, as well as generates a Linestring geometry connecting the bus stop to traffic sensor location for visualization.
- Creating a function that iterates through given lat, long points (i.e. bus stop and traffic count sensor locations) and determines which polygon the location is within (i.e. neighborhood shapes). This mapped traffic sensor IDs to neighborhood names, and bus stop IDs to neighborhood names.
- Outputting all results into CSVs ready for importing into a mySQL database.
- Visualizing the resulting CSVs in QGIS to validate results (see Figure 2).

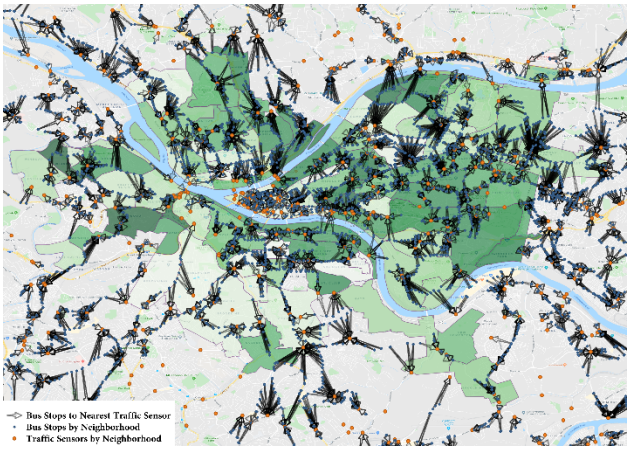



**Figure 2 – Interactive QGIS map of bus stops, traffic sensors, neighborhoods (green polygons), and arrows pointing from bus stops to their nearest identified traffic sensor used to validate Python geospatial analysis results.**

**4) mySQL Database Creation:** We formulated a single SQL script that creates the `Pittsburgh_bus_transit` database with tables satisfying all entities and relationships of the ERD in Figure 1 (see Appendix (H)). This script populates all desired data using the LOAD DATA SQL command with the LOCAL INFILE command to load data locally from a user's local directory to the localhost served database. This script takes only ~2 seconds to run and populate data from the multiple data sources (see Appendix (A), Figure 9)!

This database stores geometries and spatial data for use in visualization and geospatial SQL queries compatible with mySQL. The order of table creation was critical in generating the database due to the multiple weak entities and foreign key references. The order for creating the tables is as follows: *neighborhood > population > educational_income > transportation > sensors > traffic_count > neighbour_sensors > bus_stop > bus_route > neighbour_busstop > busstop_sensors > busstop_busroute > busstopusage_busroute > ridershipby_busroute > performance_busroute.*

The resulting database UML demonstrating all tables, relationships, and attributes is shown in Figure 3:

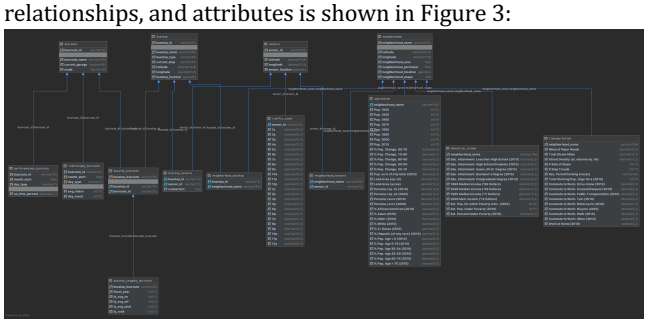


**Figure 3 - UML generated by DataGrip demonstrating `pittsburgh_bus_transit` database with all tables, relationships, keys, and attributes.**

**5) Preliminary Results with SQL Queries:** With the created database, we ran SQL queries to generate preliminary results to demonstrate the following:

- The need for using a relational geospatial database to associate the multiple data sources for complex queries to conduct the desired bus transit analysis.
- Demonstrate the database flexibility and test all relationships between neighborhood, traffic count, bus stop, and bus route data with INNER JOINS.
- Explore the SNAP, traffic count, and bus performance data to generate preliminary results on bus stops whose improvement can most benefit Pittsburgh.

**6) Tableau:** We created a front-end interface the database by connecting directly to the localhost mySQL database using Tableau (see Figure 4). Tableau allows us and potential users to implement complex data queries and explore results that can strengthen conclusions on how Pittsburgh bus transit can be improved.

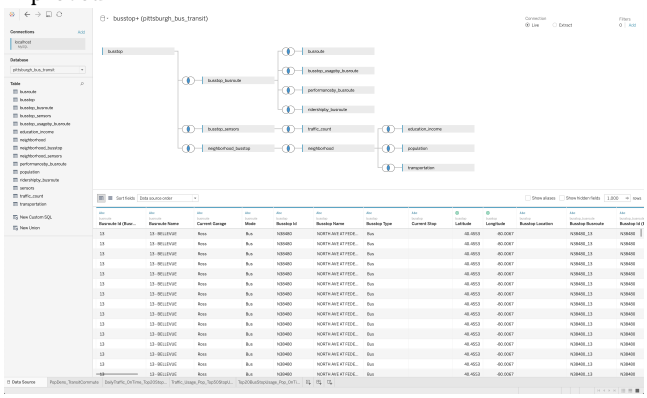
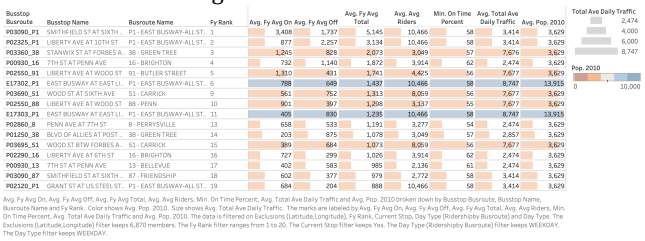


**Figure 4 - Connecting to the mySQL database in Tableau and creating a data source with inner joins from bus stop, traffic count, and neighborhood data.**

## RESULTS AND CONCLUSIONS

We demonstrate some early exploratory analysis using SQL queries directly in the database. In the SQL query in Appendix (B), we show that the top two neighborhoods using public transit are Shadyside and Squirrel Hill South. These are likely driven largely by the university populations in and around those neighborhoods. This of course ties back to our initial motivation as university students.

It can be observed that a large number of people use the bus stops in the Central Business District and East Liberty neighborhoods though these neighborhoods are the most highly populated. The reason for this is that both these neighborhoods are likely top workplace locations. See Figure 5 and Figure 6 below:



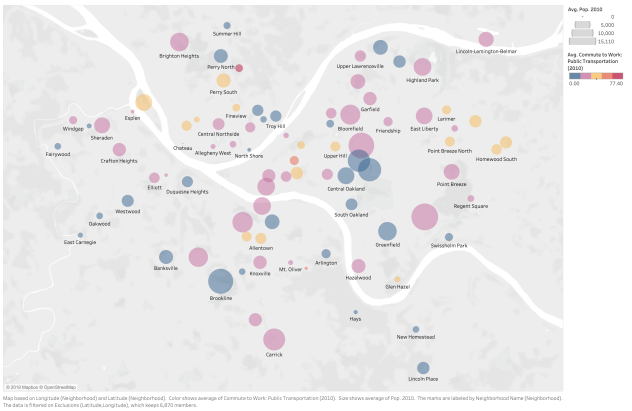
| Busstop Busroute | Busstop Name | Busroute Name | Fy Rank | Avg. Fy Avg On | Avg. Fy Avg Off | Avg. Fy Avg Total | Avg. Avg Riders | Min. On Time Percent | Avg. Total Ave Daily Traffic | Avg. Pop. 2010 |
|---|---|---|---|---|---|---|---|---|---|---|
| P03090_P1 | SMITHFIELD ST AT SIXTH ... | P1 - EAST BUSWAY-ALL ST... | 1 | 1,408 | 1,737 | 5,145 | 10,466 | 58 | 3,414 | 3,629 |
| P02325_P1 | LIBERTY AVE AT 10TH ST | P1 - EAST BUSWAY-ALL ST... | 2 | 877 | 2,257 | 3,134 | 10,466 | 58 | 3,414 | 3,629 |
| P03360_38 | STANWIX ST AT FORBES A... | 38 - GREEN TREE | 3 | 1,245 | 828 | 2,073 | 3,049 | 57 | 7,676 | 3,629 |
| P00930_16 | 7TH ST AT PENN AVE | 16 - BRIGHTON | 4 | 732 | 1,140 | 1,872 | 3,914 | 62 | 2,474 | 3,629 |
| P02550_91 | LIBERTY AVE AT WOOD ST | 91 - BUTLER STREET | 5 | 1,310 | 431 | 1,741 | 4,425 | 56 | 7,677 | 3,629 |
| E17302_P1 | EAST BUSWAY AT EAST LI... | P1 - EAST BUSWAY-ALL ST... | 6 | 788 | 649 | 1,437 | 10,466 | 58 | 8,747 | 13,915 |
| P03690_51 | WOOD ST AT SIXTH AVE | 51 - CARRICK | 9 | 561 | 752 | 1,313 | 8,059 | 56 | 7,677 | 3,629 |
| P02550_88 | LIBERTY AVE AT WOOD ST | 88 - PENN | 10 | 901 | 397 | 1,298 | 3,137 | 55 | 7,677 | 3,629 |
| E17303_P1 | EAST BUSWAY AT EAST LI... | P1 - EAST BUSWAY-ALL ST... | 11 | 405 | 830 | 1,235 | 10,466 | 58 | 8,747 | 13,915 |
| P02860_8 | PENN AVE AT 7TH ST | 8 - PERRYSVILLE | 13 | 658 | 533 | 1,191 | 3,277 | 54 | 2,474 | 3,629 |
| P01250_38 | BLVD OF ALLIES AT POST ... | 38 - GREEN TREE | 14 | 203 | 875 | 1,078 | 3,049 | 57 | 2,857 | 3,629 |
| P03695_51 | WOOD ST BTW FORBES A... | 51 - CARRICK | 15 | 389 | 684 | 1,073 | 8,059 | 56 | 7,677 | 3,629 |
| P02290_16 | LIBERTY AVE AT 6TH ST | 16 - BRIGHTON | 16 | 727 | 299 | 1,026 | 3,914 | 62 | 2,474 | 3,629 |
| P00930_13 | 7TH ST AT PENN AVE | 13 - BELLEVUE | 17 | 402 | 583 | 985 | 2,136 | 61 | 2,474 | 3,629 |
| P03090_87 | SMITHFIELD ST AT SIXTH ... | 87 - FRIENDSHIP | 18 | 602 | 377 | 979 | 2,772 | 58 | 3,414 | 3,629 |
| P02120_P1 | GRANT ST AT US STEEL ST... | P1 - EAST BUSWAY-ALL ST... | 19 | 684 | 204 | 888 | 10,466 | 58 | 3,414 | 3,629 |

**Figure 5 – Top 20 Bus Stops by FY2019 usage are all in Downtown Pittsburgh and East Liberty locations. East Liberty stops and EAST BUSWAY route has highest avg. weekday ridership, and lowest minimum monthly on-time percent. East Liberty has a higher population that Downtown stops. Recommendation is to focus on improving EAST BUSWAT AT EAST LIBERTY bus stop and P1 EAST BUSWAY route.**


**Figure 6 – Downtown and East Liberty bus stops are the most heavily used bus stops in 2019. East Liberty has a high population but low on-time percentage and should be considered for bus transit improvements.**

It is observed that a large number of people live around the neighborhoods of Oakland, Squirrel Hill, Brookline, and Bloomfield (see Figure 7). However, the neighborhood population does not seem to directly correlate with the number of commuters from that neighborhood. Rather, neighborhood location seems to be a dominating factor in influencing the number of people using public transport for work. People living

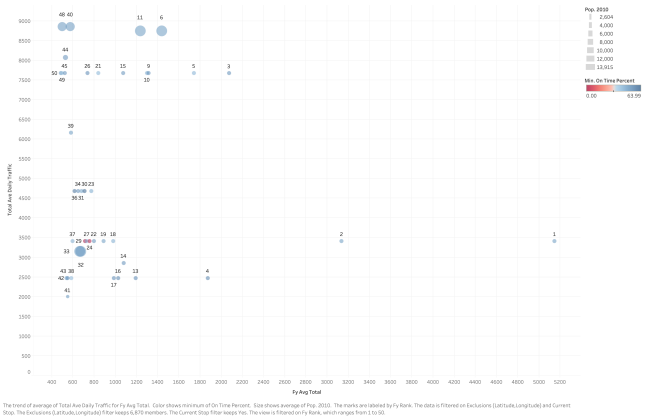
very close or very far from the workplace likely prefer not to use public transport.




Map based on Longitude (Neighborhood) and Latitude (Neighborhood). Color shows average of Commute to Work: Public Transportation (2010). Size shows average of Pop. 2010. The marks are labeled by Neighborhood Name (Neighborhood). The data is filtered on Exclusions (Latitude,Longitude), which keeps 6,870 members.

**Figure 7 – Neighborhoods with high percentages of the population using public transit to commute to work tend to have lower total populations.**

It is observed that there isn't a correlation between the daily traffic and bus stop usage (see Figure 8). Some of the least used bus stops are shown to have a lot of daily traffic. Additionally, bus stops (sensor data near the stops) that are heavily used (in the Central Business District and East Liberty) are not the most heavily trafficked during the peak commute hours of morning and evening (see Appendix (G)). It shows that people prefer taking public transport in these neighborhoods and not necessarily driving in comparison to other neighborhoods. This may be indicative of a good public transport system.




The trend of average of Total Ave Daily Traffic for Fy Avg Total. Color shows minimum of On Time Percent. Size shows average of Pop. 2010. The marks are labeled by Fy Rank. The data is filtered on Exclusions (Latitude,Longitude) and Current Stop. The Exclusions (Latitude,Longitude) filter keeps 6,870 members. The Current Stop filter keeps Yes. The view is filtered on Fy Rank, which ranges from 1 to 50.

**Figure 8 – No strong correlations are observed between bus stops with high traffic counts and high bus stop usage**

**Recommendation: Our preliminary conclusion is for the Port Authority to focus on the East Liberty bus stops and EAST BUSWAY route P1 to improve bus transit.** The top two East Liberty bus stops are rank 6 and 11 in utilization. East Liberty is a growing neighborhood (given population and traffic), and improving the bus stops there with more routes, buses, stations, etc. can accommodate growing demand, potentially decrease traffic and improve livelihoods for a large population of Pittsburghers.

**Database Design:** Pros of our database design include the consolidation and association of multiple unconnected data sources to analyze census data, traffic count data, and bus stop data with geospatial analytics. Our database is also dynamic enough to store annual updated Port Authority reports on bus stop and route performance. This ultimately provides the ability to study neighborhood public transport usage, bus stop usage trends and traffic conditions to identify potential bus stops that would need improvements where the original Western Pennsylvania data sources could not.

Cons of our database design are inflexibility with updating our traffic count census reports, although these happen infrequently. We heavily depend on the Euclidian distance calculations to associate bus stops with the nearest traffic sensor. This is relatively inaccurate since we can see some bus stops linked to traffic sensors across the river. This leads to inaccurate results using the sensors near a bus stops to estimate the traffic conditions for the bus stop. Additionally, we assume that bus routes along a congested sensor can relieve traffic congestion given more ridership.

## TECHNICAL COURSE CONTENT

We utilized the following technical content covered in the Data Management course:

1. Utilized Entity-Relationship Diagrams to design a database integrating numerous, disparate data sources;
2. Generated and created a SQL database centralizing all data sources to be used for the bus transit analysis;
3. Leveraged relational SQL queries to generate insights and extract joined and filtered data across multiple data sources to achieve the objectives.

## REFERENCES

*Pittsburgh SNAP Census Data*. (2010). Retrieved from Western Pennsylvania Regional Data Center: https://data.wprdc.org/dataset/pgh

*Port Authority Monthly Average Ridership by Route*. (2019). Retrieved from Western Pennsylvania Regional Data Center: https://data.wprdc.org/dataset/port-authority-monthly-average-ridership-by-route

*Port Authority Monthly On-Time Performance by Route*. (2019). Retrieved from Western Pennsylvania Regional Data Center: https://data.wprdc.org/dataset/port-authority-monthly-average-on-time-performance-by-route

*Port Authority Transit Stop Usage*. (2019). Retrieved from Western Pennsylvania Regional Data Center:

https://data.wprdc.org/dataset/port-authority-transit-stop-usage

*Traffic Counts 2012-2014 Data*. (2014). Retrieved from Western Pennsylvania Regional Data Center: https://data.wprdc.org/dataset/allegheny-county-traffic-counts/resource/8edd8a76-8607-4ed3-960f-dcae914fd937

## APPENDIX

### A) Running SQL script to create `pittsburgh_bus_transit` database

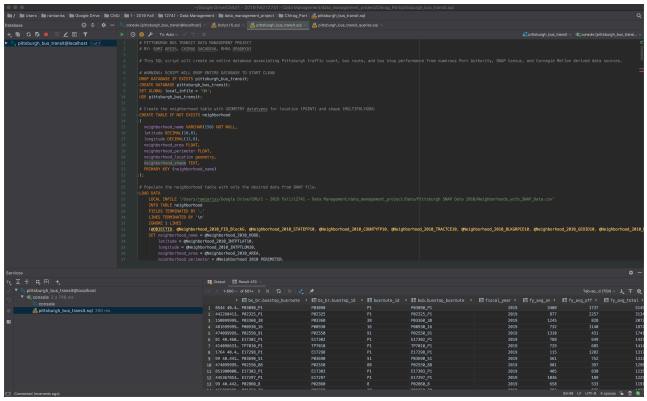


**Figure 9 – Demonstration of running SQL script in DataGrip in ~2 seconds to create database, data tables, and load all data into `pittsburgh_bus_transit` database.**

### B) QUERY : Top 5 Neighborhoods with most people taking public transportation (2010)

```
SELECT n.neighborhood_name, `Pop. 2010` * `Commute to
    Work: Public Transportation (2010)` as `People taking
    Public Transportation` FROM neighborhood n
    INNER JOIN population p ON n.neighborhood_name =
    p.neighborhood_name
    INNER JOIN transportation t ON n.neighborhood_name =
    t.neighborhood_name
    INNER JOIN education_income ei on n.neighborhood_name
    = ei.neighborhood_name
ORDER BY `People taking Public Transportation`DESC
LIMIT 5;
```

| neighborhood_name | People taking Public Transportation |
|---|---|
| **Shadyside** | 374313.5 |
| **Squirrel Hill South** | 342997 |
| **Marshall-Shadeland** | 207274.9 |
| **Carrick** | 198214.8 |
| **Beechview** | 169048.8 |

### C) QUERY : Top 5 BusStops with most users per month (2019)

```
SELECT busstop_name, fy_avg_total FROM busstop bs
    INNER JOIN busstop_sensors bs_s ON bs.busstop_id =
    bs_s.busstop_id
    INNER JOIN busstop_busroute bs_br on bs_br.busstop_id =
    bs.busstop_id
    INNER JOIN busstop_usageby_busroute bub on
    bs_br.busstop_busroute = bub.busstop_busroute
    INNER JOIN traffic_count t ON t.sensor_id = bs_s.sensor_id
ORDER BY fy_rank
LIMIT 5;
```

| busstop_name | fy_avg_total |
|---|---|
| **SMITHFIELD ST AT SIXTH AVE** | 5145 |
| **LIBERTY AVE AT 10TH ST** | 3134 |
| **STANWIX ST AT FORBES AVE** | 2073 |
| **7TH ST AT PENN AVE** | 1872 |
| **LIBERTY AVE AT WOOD ST** | 1741 |

### D) QUERY : Top 5 BusStops with most users getting ON per month (2019)

```
SELECT busstop_name, fy_avg_on FROM busstop bs
    INNER JOIN busstop_sensors bs_s ON bs.busstop_id =
      bs_s.busstop_id
    INNER JOIN busstop_busroute bs_br on bs_br.busstop_id
      = bs.busstop_id
    INNER JOIN busstop_usageby_busroute bub on
      bs_br.busstop_busroute = bub.busstop_busroute
    INNER JOIN traffic_count t ON t.sensor_id =
      bs_s.sensor_id
ORDER BY fy_avg_on DESC;
```

| busstop_name | fy_avg_on |
|---|---|
| **SMITHFIELD ST AT SIXTH AVE** | 3408 |
| **LIBERTY AVE AT WOOD ST** | 1310 |
| **STANWIX ST AT FORBES AVE** | 1245 |
| **EAST BUSWAY AT WILKINSBURG S** | 1036 |
| **LIBERTY AVE AT WOOD ST** | 901 |

## E) QUERY : Top 5 BusStops with most users getting OFF per month (2019)

```
SELECT busstop_name, fy_avg_off FROM busstop bs
  INNER JOIN busstop_sensors bs_s ON bs.busstop_id =
    bs_s.busstop_id
  INNER JOIN busstop_busroute bs_br on bs_br.busstop_id =
    bs.busstop_id
  INNER JOIN busstop_usageby_busroute bub on
    bs_br.busstop_busroute = bub.busstop_busroute
  INNER JOIN traffic_count t ON t.sensor_id = bs_s.sensor_id
ORDER BY fy_avg_off DESC
LIMIT 5;
```

| busstop_name | fy_avg_off |
|---|---|
| **LIBERTY AVE AT 10TH ST** | 2257 |
| **SMITHFIELD ST AT SIXTH AVE** | 1737 |
| **EAST BUSWAY AT WILKINSBURG S** | 1202 |
| **7TH ST AT PENN AVE** | 1140 |
| **BLVD OF ALLIES AT POST GAZET** | 875 |

## F) QUERY: Max Morning and Evening traffic count for the nearest traffic sensors to Bus Stops

```
SELECT MAX(6a+7a+8a+9a) as `Max Morning commute`,
MAX(4p+5p+6p+7p) as `Max Evening commute` FROM busstop
bs

  INNER JOIN busstop_sensors bs_s ON bs.busstop_id =
bs_s.busstop_id

  INNER JOIN neighborhood_busstop nh_bs ON bs.busstop_id =
nh_bs.busstop_id

  INNER JOIN busstop_busroute bs_br on bs_br.busstop_id =
bs.busstop_id

  INNER JOIN busstop_usageby_busroute bub on
bs_br.busstop_busroute = bub.busstop_busroute

  INNER JOIN traffic_count t ON t.sensor_id = bs_s.sensor_id

;
```

## G) QUERY: Percent of max traffic count for morning and evening commute for top 5 bus stops and rank 6 and 11 East Liberty bus stops by bus usage

```
SELECT neighborhood_name, busstop_name, if(6a+7a+8a+9a<0,
"No data", (6a+7a+8a+9a)/3279.00) as `Morning commute`,
if(4p+5p+6p+7p<0, "No data", (4p+5p+6p+7p)/4747.50) as
`Evening commute` FROM busstop bs

  INNER JOIN busstop_sensors bs_s ON bs.busstop_id =
bs_s.busstop_id

  INNER JOIN neighborhood_busstop nh_bs ON bs.busstop_id =
nh_bs.busstop_id

  INNER JOIN busstop_busroute bs_br on bs_br.busstop_id =
bs.busstop_id

  INNER JOIN busstop_usageby_busroute bub on
bs_br.busstop_busroute = bub.busstop_busroute

  INNER JOIN traffic_count t ON t.sensor_id = bs_s.sensor_id

where busstop_name in ("SMITHFIELD ST AT SIXTH AVE",
"LIBERTY AVE AT 10TH ST","STANWIX ST AT FORBES
AVE","7TH ST AT PENN AVE","LIBERTY AVE AT WOOD
ST")

group by  busstop_name;

;
```

| neighborhood_name | busstop_name | Morning commute | Evening commute |
|---|---|---|---|
| **Central Business District** | 7TH ST AT PENN AVE | NO DATA | 0.301001 |
| **Central Business District** | STANWIX ST AT FORBES AVE | 0.413236 | 0.533333 |
| **Central Business District** | LIBERTY AVE AT WOOD ST | 0.409271 | 0.443391 |
| **Central Business District** | SMITHFIELD ST AT SIXTH AVE | NO DATA | 0.436124 |
| **Central Business District** | LIBERTY AVE AT 10TH ST | NO DATA | 0.436124 |

## H) SQL script for database creation

```
# PITTSBURGH BUS TRANSIT DATA MANAGEMENT PROJECT

# BY: RAMI ARISS, CHIRAG SACHDEVA, RHEA UPADHYAY


# This SQL script will create an entire database associating Pittsburgh traffic count, bus route, and bus stop performance from numerous Port Authority, SNAP Census, and Carnegie Mellon derived data sources.


# NOTE: Find and replace root directory "/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project" with local root directory before running script.


# WARNING: SCRIPT WILL DROP ENTIRE DATABASE TO START CLEAN

DROP DATABASE IF EXISTS pittsburgh_bus_transit;

CREATE DATABASE pittsburgh_bus_transit;

SET GLOBAL local_infile = 'ON';

USE pittsburgh_bus_transit;


# Create the neighborhood table with GEOMETRY datatypes for location (POINT) and shape (MULTIPOLYGON)

CREATE TABLE IF NOT EXISTS neighborhood
(
 neighborhood_name VARCHAR(150) NOT NULL,
 latitude DECIMAL(10,8),
 longitude DECIMAL(11,8),
 neighborhood_area FLOAT,
 neighborhood_perimeter FLOAT,
 neighborhood_location geometry,
 neighborhood_shape TEXT,
 PRIMARY KEY (neighborhood_name)
);


# Populate the neighborhood table with only the desired data from SNAP file.

LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Pittsburgh SNAP Data 2010/Neighborhoods_with_SNAP_Data.csv'
  INTO TABLE neighborhood
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@OBJECTID, @Neighborhood_2010_FID_BlockG,
@Neighborhood_2010_STATEFP10,
@Neighborhood_2010_COUNTYFP10,
@Neighborhood_2010_TRACTCE10,
@Neighborhood_2010_BLKGRPCE10,
@Neighborhood_2010_GEOID10,
@Neighborhood_2010_NAMELSAD10,
@Neighborhood_2010_MTFCC10,
@Neighborhood_2010_FUNCSTAT10,
@Neighborhood_2010_ALAND10,
@Neighborhood_2010_AWATER10,
@Neighborhood_2010_INTPTLAT10,
@Neighborhood_2010_INTPTLON10,
@Neighborhood_2010_Shape_Leng,
@Neighborhood_2010_FID_Neighb,
@Neighborhood_2010_AREA,
@Neighborhood_2010_PERIMETER,
@Neighborhood_2010_NEIGHBOR_,
@Neighborhood_2010_NEIGHBOR_I,
@Neighborhood_2010_HOOD,
@Neighborhood_2010_HOOD_NO,
@Neighborhood_2010_ACRES,
@Neighborhood_2010_SQMILES,
```

@Neighborhood_2010_DPWDIV,
@Neighborhood_2010_UNIQUE_ID,
@Neighborhood_2010_SECTORS,
@Neighborhood_2010_Shape_Le_1,
@Neighborhood_2010_Shape_Ar_1,
@Neighborhood_2010_Page_Number,
@SNAP_All_csv_Neighborhood, @SNAP_All_csv_Sector__,
@Pop__1940, @Pop__1950, @Pop__1960, @Pop__1970,
@Pop__1980, @Pop__1990, @Pop__2000, @Pop__2010,
@F__Pop__Change__60_70, @F__Pop__Change__70_80,
@F__Pop__Change__80_90, @F__Pop__Change__90_00,
@F__Pop__Change__00_10, @Pop__as___of_City_total__2010_,
@Land_Area__sq__mi_, @SNAP_All_csv_Land_Area__acres_,
@Persons___sq__mi__2010_, @Persons___sq__mi__2000_,
@SNAP_All_csv_Persons___acre__20,
@SNAP_All_csv_Persons___acre__21,
@SNAP_All_csv___African_American,
@SNAP_All_csv___Asian__2010_,
@SNAP_All_csv___Other__2010_,
@SNAP_All_csv___White__2010_,
@SNAP_All_csv___2__Races__2010_,
@SNAP_All_csv___Hispanic__of_any, @F__Pop__Age___5__2010_,
@F__Pop__Age_5_19__2010_, @F__Pop__Age_20_34__2010_,
@F__Pop__Age_35_59__2010_, @F__Pop__Age_60_74__2010_,
@F__Pop__Age___75__2010_,
@SNAP_All_csv_Total___Units__200,
@SNAP_All_csv_Total___Units__201,
@SNAP_All_csv___Occupied_Units__,
@SNAP_All_csv___Vacant_Units__20,
@SNAP_All_csv___Occupied_Units_1,
@SNAP_All_csv___Owner_Occupied_U,
@SNAP_All_csv___Renter_Occupied_,
@Est__Avg__Yrs__of_Residence__20,
@SNAP_All_csv___Living_in_Househ,
@SNAP_All_csv___Living_in_Group_,
@SNAP_All_csv___Units_Built_00_0,
@SNAP_All_csv___Units_Built_90_9,
@SNAP_All_csv___Units_Built_80_8,
@SNAP_All_csv___Units_Built_60_7,
@SNAP_All_csv___Units_Built_40_5,
@SNAP_All_csv___Units_Built_befo,
@SNAP_All_csv_Median_Home_Value_,
@Med__Val____00_in__10_Dollars_,
@SNAP_All_csv_Median_Home__Value,
@SNAP_All_csv___Change_Real_Valu,
@SNAP_All_csv_Median_Sale_Price_,
@SNAP_All_csv___Sales_Counted__2,
@SNAP_All_csv_Foreclosures__2008,
@SNAP_All_csv_Foreclosures__2010,
@SNAP_All_csv___of_all_Housing_U,
@Total_Age_16__N_hood_Residents_,
@SNAP_All_csv_Resident_Jobs__Con,
@SNAP_All_csv_Resident_Jobs__Man,
@SNAP_All_csv_Resident_Jobs__Ret,
@SNAP_All_csv_Resident_Jobs__Tra,
@SNAP_All_csv_Resident_Jobs__Inf,
@SNAP_All_csv_Resident_Jobs__Fin,
@SNAP_All_csv_Resident_Jobs__Pro,
@SNAP_All_csv_Resident_Jobs__Edu,
@SNAP_All_csv_Resident_Jobs__Art,
@SNAP_All_csv_Resident_Jobs__Pub,
@SNAP_All_csv_Resident_Jobs__Oth,
@Total___Jobs_Located_in_N_hood_,
@SNAP_All_csv_Jobs_in__Hood__Con,
@SNAP_All_csv_Jobs_in__Hood__Man,
@SNAP_All_csv_Jobs_in__Hood__Ret,
@SNAP_All_csv_Jobs_in__Hood__Tra,
@SNAP_All_csv_Jobs_in__Hood__Inf,
@SNAP_All_csv_Jobs_in__Hood__Fin,
@SNAP_All_csv_Jobs_in__Hood__Pro,
@SNAP_All_csv_Jobs_in__Hood__Edu,
@SNAP_All_csv_Jobs_in__Hood__Art,
@SNAP_All_csv_Jobs_in__Hood__Pub,
@SNAP_All_csv_Jobs_in__Hood__Oth,
@SNAP_All_csv_Total_Pop__25_and_,
@Edu__Attainment__Less_than_High,
@Edu__Attainment__High_School_Gr,
@Edu__Attainment__Assoc__Prof__D,
@Edu__Attainment__Bachelor_s_Deg,
@Edu__Attainment__Postgraduate_D,
@SNAP_All_csv_1999_Median_Income,
@SNAP_All_csv_2009_Median_Income,
@SNAP_All_csv_1999_Median_Inco_1,
@F2009_Med__Income___13_Dollars_,
@Est__Pop__for_which_Poverty_Cal,
@Est__Pop__Under_Poverty__2010_,
@Est__Percent_Under_Poverty__201,
@SNAP_All_csv__Part_1__Major_Cri,
@SNAP_All_csv__Part_2_Reports__2,
@SNAP_All_csv__Other_Police_Repo,
@SNAP_All_csv_Part_1_Crime_per_1,
@SNAP_All_csv_Part_2_Crime_per_1,
@SNAP_All_csv__Murder__2010_,
@SNAP_All_csv__Rape__2010_,
@SNAP_All_csv__Robbery__2010_, @F_Agr__Assault__2010_,
@SNAP_All_csv__Burglary__2010_,
@SNAP_All_csv__Auto_Theft__2010_,
@SNAP_All_csv__Drug_Violations__,
@SNAP_All_csv_Land_Area__acres1,
@Approx__Total___Parcels__2010_,
@Approx__Total___Taxable_Parcels,
@Approx____of_Structures__2010_,
@Approx____Unoccupied_Parcels__2,
@SNAP_All_csv___Good___Excellent,
@SNAP_All_csv___Average_Conditio,
@SNAP_All_csv___Poor___Derelict_,
@F__Residential_Bldg__Permits__2,
@F__Residential_Bldg__Permits__3,

```
@F__Commercial_Bldg__Permits__20,
@F__Commercial_Bldg__Permits__21,
@SNAP_All_csv___Code_Violations_,
@F__of_all_Bldgs__w__Code_Violat,
@SNAP_All_csv___Condemned_Struct,
@F__of_all_Bldgs__Condemned__201,
@SNAP_All_csv___Demolitions__201,
@F__Tax_Delinquent_Prop___2__yrs,
@F__of_Taxable_Prop__Delinquent_,
@SNAP_All_csv_Landslide_Prone___,
@SNAP_All_csv_Undermined____land,
@SNAP_All_csv_Flood_Plain____lan,
@SNAP_All_csv___Street_Trees,
@SNAP_All_csv_Park_Space__acres_,
@SNAP_All_csv_Park_Space____of_l,
@Park_Space__acres_1000_pers__,
@SNAP_All_csv_Greenway____of_lan,
@SNAP_All_csv_Woodland____of_lan,
@SNAP_All_csv_Cemetery____of_lan,
@SNAP_All_csv_Residential,
@SNAP_All_csv_Mixed_Use___Commer,
@SNAP_All_csv_Mixed_Use___Indust,
@Institutional___Edu____Med_,      @SNAP_All_csv_Open_Space,
@SNAP_All_csv_Hillside,      @SNAP_All_csv_Special_Land_Use,
@SNAP_All_csv_Miles_of_Major_Roa,
@SNAP_All_csv_Total_Street_Miles,
@Street_Density__st__mi_area_sq_,
@SNAP_All_csv___Sets_of_Steps,
@SNAP_All_csv___Step_Treads,
@Res__Permit_Parking_Area_s_,
@Total_Working_Pop___Age_16____2,
@SNAP_All_csv_Commute_to_Work__D,
@SNAP_All_csv_Commute_to_Work__C,
@SNAP_All_csv_Commute_to_Work__P,
@SNAP_All_csv_Commute_to_Work__T,
@SNAP_All_csv_Commute_to_Work__M,
@SNAP_All_csv_Commute_to_Work__B,
@SNAP_All_csv_Commute_to_Work__W,
@SNAP_All_csv_Commute_to_Work__O,
@SNAP_All_csv_Work_at_Home__2010)

  SET neighborhood_name = @Neighborhood_2010_HOOD,

    latitude = @Neighborhood_2010_INTPTLAT10,

    longitude = @Neighborhood_2010_INTPTLON10,

    neighborhood_area = @Neighborhood_2010_AREA,

    neighborhood_perimeter =
@Neighborhood_2010_PERIMETER,

    neighborhood_location =
POINT(@Neighborhood_2010_INTPTLAT10,
@Neighborhood_2010_INTPTLON10)

;
```

```
# Create a temporary neighborhood_temp table to load shape (MULTIPOLYGON) and update the desired neighborhood table

CREATE TABLE IF NOT EXISTS neighborhood_temp LIKE neighborhood;


# Populate the neighborhood_temp table with only the desired fields from the CSV shape file.

LOAD DATA

  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Python/neighborhoods_shp.csv'

  INTO TABLE neighborhood_temp

  FIELDS TERMINATED BY '\t'

  LINES TERMINATED BY '\n'

  IGNORE 1 LINES

  (@WKT, @Neighbor19)

  SET neighborhood_name = @Neighbor19,

    neighborhood_shape = @WKT

;


# Update the neighbor table with the shape GEOMETRY from neighbor_temp on neighborhood_name PRIMARY KEY

UPDATE neighborhood n

INNER JOIN neighborhood_temp nt on n.neighborhood_name = nt.neighborhood_name

SET n.neighborhood_shape = nt.neighborhood_shape;


# DROP the temporary neighbor table

DROP TABLE neighborhood_temp;


# Create the population table to store desired data from the SNAP Population and Density.

CREATE TABLE IF NOT EXISTS population

(

 neighborhood_name VARCHAR(150) NOT NULL,

 `Pop. 1940` INT NULL,

 `Pop. 1950` INT NULL,
```

```
`Pop. 1960` INT NULL,
`Pop. 1970` INT NULL,
`Pop. 1980` INT NULL,
`Pop. 1990` INT NULL,
`Pop. 2000` INT NULL,
`Pop. 2010` INT NULL,
`% Pop. Change, 60-70` DECIMAL(5,2) NULL,
`% Pop. Change, 70-80` DECIMAL(5,2) NULL,
`% Pop. Change, 80-90` DECIMAL(5,2) NULL,
`% Pop. Change, 90-00` DECIMAL(5,2) NULL,
`% Pop. Change, 00-10` DECIMAL(5,2) NULL,
`Pop. as % of City total (2010)` DECIMAL(5,2) NULL,
`Land Area (sq. mi)` DECIMAL(10,2) NULL,
`Land Area (acres)` DECIMAL(10,2) NULL,
`Persons / sq. mi (2010)` DECIMAL(10,2) NULL,
`Persons / sq. mi (2000)` DECIMAL(10,2) NULL,
`Persons / acre (2010)` DECIMAL(10,2) NULL,
`Persons / acre (2000)` DECIMAL(10,2) NULL,
`% African American (2010)` DECIMAL(5,2) NULL,
`% Asian (2010)` DECIMAL(5,2) NULL,
`% Other (2010)` DECIMAL(5,2) NULL,
`% White (2010)` DECIMAL(5,2) NULL,
`% 2+ Races (2010)` DECIMAL(5,2) NULL,
`% Hispanic (of any race) (2010)` DECIMAL(5,2) NULL,
`% Pop. Age < 5 (2010)` DECIMAL(5,2) NULL,
`% Pop. Age 5-19 (2010)` DECIMAL(5,2) NULL,
`% Pop. Age 20-34 (2010)` DECIMAL(5,2) NULL,
`% Pop. Age 35-59 (2010)` DECIMAL(5,2) NULL,
`% Pop. Age 60-74 (2010)` DECIMAL(5,2) NULL,
`% Pop. Age > 75 (2010)` DECIMAL(5,2) NULL,
 CONSTRAINT FOREIGN KEY (neighborhood_name) REFERENCES neighborhood(neighborhood_name)
);

# Populate the population table with the desired fields.
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Pittsburgh SNAP Data 2010/opendata-pghsnap-neighborhood-census-data_populationanddensity.csv'
  INTO TABLE population
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@`Neighborhood`,@`Sector #`,`Pop. 1940`,`Pop. 1950`,`Pop. 1960`,`Pop. 1970`,`Pop. 1980`,`Pop. 1990`,`Pop. 2000`,`Pop. 2010`,`% Pop. Change, 60-70`,`% Pop. Change, 70-80`,`% Pop. Change, 80-90`,`% Pop. Change, 90-00`,`% Pop. Change, 00-10`,`Pop. as % of City total (2010)`,`Land Area (sq. mi)`,`Land Area (acres)`,`Persons / sq. mi (2010)`,`Persons / sq. mi (2000)`,`Persons / acre (2010)`,`Persons / acre (2000)`,`% African American (2010)`,`% Asian (2010)`,`% Other (2010)`,`% White (2010)`,`% 2+ Races (2010)`,`% Hispanic (of any race) (2010)`,`% Pop. Age < 5 (2010)`,`% Pop. Age 5-19 (2010)`,`% Pop. Age 20-34 (2010)`,`% Pop. Age 35-59 (2010)`,`% Pop. Age 60-74 (2010)`,`% Pop. Age > 75 (2010)`)
  SET neighborhood_name = @Neighborhood
;

# Create the education_income table to store desired data from the SNAP Education and Income.
CREATE TABLE IF NOT EXISTS education_income
(
 neighborhood_name VARCHAR(150) NOT NULL,
 `Edu. Attainment: Less than High School (2010)` DECIMAL(5,2) NULL,
 `Edu. Attainment: High School Graduate (2010)` DECIMAL(5,2) NULL,
 `Edu. Attainment: Assoc./Prof. Degree (2010)` DECIMAL(5,2) NULL,
 `Edu. Attainment: Bachelor's Degree (2010)` DECIMAL(5,2) NULL,
 `Edu. Attainment: Postgraduate Degree (2010)` DECIMAL(5,2) NULL,
 `1999 Median Income ('99 Dollars)` DECIMAL(12,2) NULL,
 `2009 Median Income ('09 Dollars)` DECIMAL(12,2) NULL,
 `1999 Median Income ('11 Dollars)` DECIMAL(12,2) NULL,
 `2009 Med. Income ('13 Dollars)` DECIMAL(12,2) NULL,
```

```
 `Est. Pop. for which Poverty Calc. (2010)` INT NULL,
 `Est. Pop. Under Poverty (2010)` INT NULL,
 `Est. Percent Under Poverty (2010)` DECIMAL(5,2) NULL,
 CONSTRAINT FOREIGN KEY (neighborhood_name) REFERENCES neighborhood(neighborhood_name)
);

# Populate the education_income table from desired data
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Pittsburgh SNAP Data 2010/opendata-pghsnap-neighborhood-census-data_employment_educationandincome.csv'
  INTO TABLE education_income
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@`Neighborhood`,@`Sector #`,@`Population (2010)`,@`Total Pop, 25 and older (2010)`,`Edu. Attainment: Less than High School (2010)`,`Edu. Attainment: High School Graduate (2010)`,`Edu. Attainment: Assoc./Prof. Degree (2010)`,`Edu. Attainment: Bachelor's Degree (2010)`,`Edu. Attainment: Postgraduate Degree (2010)`,`1999 Median Income ('99 Dollars)`,`2009 Median Income ('09 Dollars)`,`1999 Median Income ('11 Dollars)`,`2009 Med. Income ('13 Dollars)`,`Est. Pop. for which Poverty Calc. (2010)`,`Est. Pop. Under Poverty (2010)`,`Est. Percent Under Poverty (2010)`)
  SET neighborhood_name = @Neighborhood
;

# Create the transportation table to store desired data from the SNAP Transportation.
CREATE TABLE IF NOT EXISTS transportation
(
 neighborhood_name VARCHAR(150) NOT NULL,
 `Miles of Major Roads` DECIMAL(10,2) NULL,
 `Total Street Miles` DECIMAL(10,2) NULL,
 `Street Density (st. mi/area sq. mi)` DECIMAL(10,2) NULL,
 `# Sets of Steps` INT NULL,
 `# Step Treads` INT NULL,
 `Res. Permit Parking Area(s)` VARCHAR(45) NULL,
 `Total Working Pop. (Age 16+) (2010)` INT NULL,
 `Commute to Work: Drive Alone (2010)` DECIMAL(5,2) NULL,
 `Commute to Work: Carpool/Vanpool (2010)` DECIMAL(5,2) NULL,
 `Commute to Work: Public Transportation (2010)` DECIMAL(5,2) NULL,
 `Commute to Work: Taxi (2010)` DECIMAL(5,2) NULL,
 `Commute to Work: Motorcycle (2010)` DECIMAL(5,2) NULL,
 `Commute to Work: Bicycle (2010)` DECIMAL(5,2) NULL,
 `Commute to Work: Walk (2010)` DECIMAL(5,2) NULL,
 `Commute to Work: Other (2010)`DECIMAL(5,2) NULL,
 `Work at Home (2010)` DECIMAL(5,2) NULL,
 CONSTRAINT FOREIGN KEY (neighborhood_name) REFERENCES neighborhood(neighborhood_name)
);

# Populate the transportation table with desired data.
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Pittsburgh SNAP Data 2010/opendata-pghsnap-neighborhood-census-data_employment_transportation.csv'
  INTO TABLE transportation
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@`Neighborhood`,@`Sector #`,@`Population (2010)`,`Miles of Major Roads`,`Total Street Miles`,`Street Density (st. mi/area sq. mi)`,`# Sets of Steps`,`# Step Treads`,`Res. Permit Parking Area(s)`,`Total Working Pop. (Age 16+) (2010)`,`Commute to Work: Drive Alone (2010)`,`Commute to Work: Carpool/Vanpool (2010)`,`Commute to Work: Public Transportation (2010)`,`Commute to Work: Taxi (2010)`,`Commute to Work: Motorcycle (2010)`,`Commute to Work: Bicycle (2010)`,`Commute to Work: Walk (2010)`,`Commute to Work: Other (2010)`,`Work at Home (2010)`)
  SET neighborhood_name = @Neighborhood
;
```

```
# Create the sensors table storing sensor information and location from traffic_counts data.
CREATE TABLE IF NOT EXISTS sensors
(
 sensor_id VARCHAR(12) NOT NULL,
 latitude DECIMAL(10,8),
 longitude DECIMAL(11,8),
 sensor_location geometry,
 PRIMARY KEY (sensor_id)
);


# Populate the sensors table with desired data.
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Traffic Counts /8edd8a76-8607-4ed3-960f-dcae914fd937.csv'
  INTO TABLE sensors
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@sensor_ID, @Longitude, @Latitude,@1a,@2a,@3a,@4a,@5a,@6a,@7a,@8a,@9a,@10a, @11a,@12p,@1p,@2p,@3p,@4p,@5p,@6p,@7p,@8p,@9p,@10p,@11p,@12a)
  SET sensor_id = @sensor_ID,
     latitude = @Latitude,
     longitude = @Longitude,
     sensor_location = POINT(@Latitude, @Longitude)
;


# Create the traffic_count table
CREATE TABLE IF NOT EXISTS traffic_count
(
 sensor_id VARCHAR(12) NOT NULL,
 1a DECIMAL(10, 2) NULL,
 2a DECIMAL(10, 2) NULL,
 3a DECIMAL(10, 2) NULL,
 4a DECIMAL(10, 2) NULL,
 5a DECIMAL(10, 2) NULL,
 6a DECIMAL(10, 2) NULL,
 7a DECIMAL(10, 2) NULL,
 8a DECIMAL(10, 2) NULL,
 9a DECIMAL(10, 2) NULL,
 10a DECIMAL(10, 2) NULL,
 11a DECIMAL(10, 2) NULL,
 12p DECIMAL(10, 2) NULL,
 1p DECIMAL(10, 2) NULL,
 2p DECIMAL(10, 2) NULL,
 3p DECIMAL(10, 2) NULL,
 4p DECIMAL(10, 2) NULL,
 5p DECIMAL(10, 2) NULL,
 6p DECIMAL(10, 2) NULL,
 7p DECIMAL(10, 2) NULL,
 8p DECIMAL(10, 2) NULL,
 9p DECIMAL(10, 2) NULL,
 10p DECIMAL(10, 2) NULL,
 11p DECIMAL(10, 2) NULL,
 12a DECIMAL(10, 2) NULL,
 CONSTRAINT FOREIGN KEY (sensor_id) REFERENCES sensors(sensor_id)
);


# Load data into traffic_count table
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Traffic Counts /8edd8a76-8607-4ed3-960f-dcae914fd937.csv'
  INTO TABLE traffic_count
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
```

```
(@sensor_ID, @Longitude, @Latitude,1a,2a,3a,4a,5a,6a,7a,8a,9a,10a,11a,12p,1p,2p,3p,4p,5p,6p,7p,8p,9p,10p,11p,12a)
SET sensor_id = @sensor_ID
;

# Create neighborhood_sensors table to spatially associate sensors to neighborhoods.
CREATE TABLE IF NOT EXISTS neighborhood_sensors
(
neighborhood_name VARCHAR(150) NOT NULL,
sensor_id VARCHAR(12) NOT NULL,
CONSTRAINT FOREIGN KEY (neighborhood_name) REFERENCES neighborhood(neighborhood_name),
CONSTRAINT FOREIGN KEY (sensor_id) REFERENCES sensors(sensor_id)
);

# Load data into measures table from nearest neighbor by euclidian distance results completed in python.
LOAD DATA
LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Python/traffic_count_sensors_to_neighborhoods_clean.csv'
INTO TABLE neighborhood_sensors
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(@node, @polygon)
SET sensor_id = @node,
neighborhood_name = @polygon
;

# Create busstop table for all busstop metadata.
CREATE TABLE IF NOT EXISTS busstop
(
busstop_id VARCHAR(10) NOT NULL,
busstop_name VARCHAR(150) NOT NULL,
busstop_type VARCHAR(45),
current_stop VARCHAR(10),
latitude DECIMAL(10,8),
longitude DECIMAL(11,8),
busstop_location geometry,
PRIMARY KEY(busstop_id)
);

# Load data into busstop table with desired data from busstopusagebyroute. Use "IGNORE" to populate just first instance of the stop as each row is a stop and route
LOAD DATA
LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Port Authority Transit Stop Usage/busstopusagebyroute_clean.csv'
IGNORE
INTO TABLE busstop
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(@STOP_ID, @ROUTE, @STOP_ROUTE, @STOP_NAME, @CLEVER_ID, @LATITUDE, @LONGITUDE, @ALL_ROUTES, @SHELTER, @STOP_TYPE, @CURRENT_STOP, @FY19_AVG_ON, @FY19_AVG_OFF, @FY19_AVG_TOTAL, @FY19_RANK, @1806_ON, @1806_OFF, @1809_ON, @1809_OFF, @1811_ON, @1811_OFF, @1903_ON, @1903_OFF, @1906_ON, @1906_OFF)
SET busstop_id = @STOP_ID,
busstop_name = @STOP_NAME,
busstop_type = @STOP_TYPE,
current_stop = @CURRENT_STOP,
latitude = @LATITUDE,
longitude = @LONGITUDE,
busstop_location = POINT(@LATITUDE, @LONGITUDE)
;

# Create busroute table for all busroute metadata.
CREATE TABLE IF NOT EXISTS busroute
(
```

```
  busroute_id VARCHAR(10) NOT NULL,
  busroute_name VARCHAR(150) NOT NULL,
  current_garage VARCHAR(45),
  mode VARCHAR(10),
  PRIMARY KEY(busroute_id)
);
```

# Load data into busroute table with desired data from monthly average ridership by route. Use "IGNORE" to populate just first instance of the route.

```
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Port Authority Monthly Average Ridership by Route/12bb84ed-397e-435c-8d1b-8ce543108698.csv'
  IGNORE
  INTO TABLE busroute
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@route, @ridership_route_code, @route_full_name, @current_garage, @mode, @month_start, @year_month, @day_type, @avg_riders, @day_count)
  SET busroute_id = @route,
    busroute_name = @route_full_name,
    current_garage = @current_garage,
    mode = @mode
;
```

# Create neighborhood_busstop table to associate busstops with the neighborhoods they are in. These were determined using geopandas CONTAINS query in python.

```
CREATE TABLE IF NOT EXISTS neighborhood_busstop
(
  busstop_id VARCHAR(10) NOT NULL,
  neighborhood_name VARCHAR(150) NOT NULL,
  CONSTRAINT FOREIGN KEY (busstop_id) REFERENCES busstop(busstop_id),
  CONSTRAINT FOREIGN KEY (neighborhood_name) REFERENCES neighborhood(neighborhood_name)
);
```

# Load data into neighborhood_busstop table from geopandas CONTAINS query results completed in python.

```
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Python/bus_stops_to_neighborhoods_clean.csv'
  INTO TABLE neighborhood_busstop
  FIELDS TERMINATED BY ';'
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@node, @polygon)
  SET busstop_id = @node,
    neighborhood_name = @polygon
;
```

# Create busstop_sensors table to associate busstops with the nearest traffic sensor determined by euclidian distances using geopandas distance calculations in python.

```
CREATE TABLE IF NOT EXISTS busstop_sensors
(
  busstop_id VARCHAR(10) NOT NULL,
  sensor_id VARCHAR(12) NOT NULL,
  connection TEXT,
  CONSTRAINT FOREIGN KEY (busstop_id) REFERENCES busstop(busstop_id),
  CONSTRAINT FOREIGN KEY (sensor_id) REFERENCES sensors(sensor_id)
);
```

# Load data into neighborhood_busstop table from geopandas CONTAINS query results completed in python.

```
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data
```

```
Management/data_management_project/Python/bus_stops_to_traffic_count_sensors_clean.csv'
  INTO TABLE busstop_sensors
  FIELDS TERMINATED BY ';'
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@STOP_ID, @sensor_ID, @geometry)
  SET busstop_id = @STOP_ID,
    sensor_id = @sensor_ID,
    connection = @geometry
;


# Create busstop_busroute table to associate bus stops with the bus routes that pass through them.
CREATE TABLE IF NOT EXISTS busstop_busroute
(
  busstop_busroute VARCHAR(20) NOT NULL,
  busstop_id VARCHAR(10) NOT NULL,
  busroute_id VARCHAR(10) NOT NULL,
  PRIMARY KEY (busstop_busroute),
  CONSTRAINT FOREIGN KEY (busstop_id) REFERENCES busstop(busstop_id),
  CONSTRAINT FOREIGN KEY (busroute_id) REFERENCES busroute(busroute_id)
);


# Load data into busstop_busroute table from desired data in busstopusagebyroute.
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Port Authority Transit Stop Usage/busstopusagebyroute_clean.csv'
#   IGNORE
  INTO TABLE busstop_busroute
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@STOP_ID, @ROUTE, @STOP_ROUTE, @STOP_NAME, @CLEVER_ID, @LATITUDE, @LONGITUDE, @ALL_ROUTES, @SHELTER, @STOP_TYPE, @CURRENT_STOP, @FY19_AVG_ON, @FY19_AVG_OFF, @FY19_AVG_TOTAL, @FY19_RANK, @1806_ON, @1806_OFF, @1809_ON, @1809_OFF, @1811_ON, @1811_OFF, @1903_ON, @1903_OFF, @1906_ON, @1906_OFF)
  SET busstop_busroute = @STOP_ROUTE,
    busstop_id = @STOP_ID,
    busroute_id = @ROUTE
;


# Create busstop_usageby_busroute table to store the bus stop usage (ridership) by bus route. We use a combined busstop_busroute key.
CREATE TABLE IF NOT EXISTS busstop_usageby_busroute
(
  busstop_busroute VARCHAR(20) NOT NULL,
  fiscal_year INT NOT NULL,
  fy_avg_on INT NULL,
  fy_avg_off INT NULL,
  fy_avg_total INT NULL,
  fy_rank INT NULL,
  CONSTRAINT FOREIGN KEY (busstop_busroute) REFERENCES busstop_busroute(busstop_busroute)
);


# Load data into busstop_busroute table from desired data in busstopusagebyroute.
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Port Authority Transit Stop Usage/busstopusagebyroute_clean.csv'
#   IGNORE
  INTO TABLE busstop_usageby_busroute
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@STOP_ID, @ROUTE, @STOP_ROUTE, @STOP_NAME, @CLEVER_ID, @LATITUDE, @LONGITUDE, @ALL_ROUTES,
```

```
@SHELTER, @STOP_TYPE, @CURRENT_STOP, @FY19_AVG_ON, @FY19_AVG_OFF, @FY19_AVG_TOTAL, @FY19_RANK, @1806_ON, @1806_OFF, @1809_ON, @1809_OFF, @1811_ON, @1811_OFF, @1903_ON, @1903_OFF, @1906_ON, @1906_OFF)
  SET busstop_busroute = @STOP_ROUTE,
    fiscal_year = '2019',
    fy_avg_on = @FY19_AVG_ON,
    fy_avg_off = @FY19_AVG_OFF,
    fy_avg_total = @FY19_AVG_TOTAL,
    fy_rank = @FY19_RANK
;

# Create ridershipby_busroute table for monthly average ridership by bus routes. NOTE the conjugate primary key of busroute_id (FK) and month_start.
CREATE TABLE IF NOT EXISTS ridershipby_busroute
(
  busroute_id VARCHAR(10) NOT NULL,
  month_start DATE NOT NULL,
  day_type VARCHAR(7) NOT NULL,
  avg_riders INT NULL,
  day_count INT NULL,
  PRIMARY KEY(busroute_id, month_start, day_type),
  CONSTRAINT FOREIGN KEY(busroute_id) REFERENCES busroute(busroute_id)
);

# Load data into busroute table with desired data from monthly average ridership by route. Use "IGNORE" to populate just first instance of the route.
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Port Authority Monthly Average Ridership by Route/12bb84ed-397e-435c-8d1b-8ce543108698.csv'
#   IGNORE
  INTO TABLE ridershipby_busroute
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@route, @ridership_route_code, @route_full_name, @current_garage, @mode, month_start, @year_month, day_type, avg_riders, day_count)
  SET busroute_id = @route
;

# Create performanceby_busroute table for monthly on time performance by bus routes. NOTE the conjugate primary key of busroute_id (FK) and month_start.
CREATE TABLE IF NOT EXISTS performanceby_busroute
(
  busroute_id VARCHAR(10) NOT NULL,
  month_start DATE NOT NULL,
  day_type VARCHAR(7) NOT NULL,
  on_time_percent DECIMAL(5,2) NULL,
  PRIMARY KEY(busroute_id, month_start, day_type),
  CONSTRAINT FOREIGN KEY(busroute_id) REFERENCES busroute(busroute_id)
);

# Load data into busroute table with desired data from monthly average ridership by route. Use "IGNORE" to populate just first instance of the route.
LOAD DATA
  LOCAL INFILE '/Users/ramiariss/Google Drive/CMU/1 - 2019 Fall/12741 - Data Management/data_management_project/Data/Port Authority Monthly On Time Performance by Route/00eb9600-69b5-4f11-b20a-8c8ddd8cfe7a.csv'
#   IGNORE
  INTO TABLE performanceby_busroute
  FIELDS TERMINATED BY ','
  LINES TERMINATED BY '\n'
  IGNORE 1 LINES
  (@route, @ridership_route_code, @route_full_name, @current_garage, @mode, month_start, @year_month, day_type, @on_time_percent, @data_source)
  SET busroute_id = @route,
```

```
    on_time_percent = @on_time_percent*100
;
```

## I) Additional resources available upon request

The following resources generated during this project can be provided upon request:

- Any raw data sources utilized in the project;
- iPython Notebook for Geospatial Analysis;
- QGIS file for geospatial analysis validation;
- Tableau file for mySQL database connection and interface;
- `pittsburgh_bus_transit` database creation SQL script;
- SQL script of demonstrated SQL queries used in results and analysis.