Lab 03. Spatial Data

**Due date:** Thursday, Feb 13 submitted as Word document to Canvas ***Lab03*** link. This lab counts 9 % toward your total grade.

**Objectives:**

1. Understand and apply logical operators to filter datasets.

2. Utilize Data Query Language (DQL) to retrieve and summarize data.

3. Perform Data Manipulation Language (DML) operations to update and modify records.

4. Use Data Definition Language (DDL) to alter table structures.

5. Implement subqueries to perform complex queries.

**Format of answer:** Submit your answers as a **Word document or pdf** with graphs (screenshot of your result table) and answer in SQL.

**Notice:** All SQL commands are in blue color

**In-Class Exercise (3pts)**

**Recap:**

**A white background with black text

Description automatically generated**

**SELECT**

**stusab,statea,**

**100.0 \* (Sum(associate) + Sum(master)) /Sum(total\_pop) AS asso\_mastr\_pct**

**FROM ch04.us\_attribute**

**GROUP BY statea,stusab**

**ORDER BY asso\_mastr\_pct DESC;**

**Or**

**SELECT**

**stusab,statea,**

**100 \* (Sum(associate) + Sum(master))::NUMERIC /Sum(total\_pop) AS asso\_mastr\_pct**

**FROM ch04.us\_attribute**

**GROUP BY statea,stusab**

**ORDER BY asso\_mastr\_pct DESC;**

**Import** nyc\_census\_blocks  **from psql**

**Psql is the command-line interface (CLI) for PostgreSQL, allowing users to interact with a PostgreSQL database directly from the terminal.**

**Cmd is the CLI for windows.**

**1. search cmd and run as administrator, change and paste the following command to check the coordinate system of the data**

ogrinfo -al -so "C:\path\to\your\shapefile.shp"

Tips:

* Ogrinfo is a command-line tool providing detailed metadata and structure information about **geospatial datasets**, such as Shapefiles, GeoJSON, KML, PostGIS, etc.

Task 1. What is the Projected CRS for this data?

**Projected CRS: 26918(NAD83/UTM Zone 18N)**

**2. In cmd, change and run the following command lines to import .shp file (the password is needed after hitting enter)**

**First, navigate to PostgreSQL’s bin directory:**

cd C:\Program Files\PostgreSQL\17\bin

**Second, run the shp2pgsql command from there:**

shp2pgsql -I -s <SRID> "C:\path\to\your\shapefile.shp" my\_schema.my\_table | psql -U your\_user -d your\_database

Tips:

* -I → **Creates a spatial index** for faster spatial queries.
* -s <SRID> → Sets the **SRID (e.g., Spatial Reference System - EPSG:4326 for WGS 84)**.
* "C:\path\to\your\shapefile.shp" → **Full path** to your shapefile.
* My\_schema.my\_table → The **PostgreSQL schema and table name**.
* | psql -U your\_user -d your\_database → Pipes the SQL output directly into **PostgreSQL**.
* shp2pgsql is a command-line tool that converts **ESRI Shapefiles (.shp)** into SQL statements.

Task 2. please paste your command line and make a screenshot to display thenyc\_census\_blockstable in pgAdmin.

**You can now type the following line to connect to PostgreSQL**

psql -U postgres -d your\_database

Tips:

* \l → List databases
* \c db\_name → Connect to a database
* \dt → List tables
* \d table\_name → Describe a table
* \du → List users
* SELECT \* FROM table\_name; → Retrieve data

Using the nyc\_census\_blocks table, answer the following questions.

Here is some helpful information to get started.

|  |  |
| --- | --- |
| **blkid** | A 15-digit code that uniquely identifies every census **block**. (“360050001009000”) |
| **popn\_total** | Total number of people in the census block |
| **popn\_white** | Number of people self-identifying as “white” in the block |
| **popn\_black** | Number of people self-identifying as “black” in the block |
| **popn\_nativ** | Number of people self-identifying as “native american” in the block |
| **popn\_asian** | Number of people self-identifying as “asias” in the block |
| **popn\_other** | Number of people self-identifying with other categories in the block |
| **hous\_total** | Number of housing units in the block |
| **hous\_own** | Number of owner-occupied housing units in the block |
| **hous\_rent** | Number of renter-occupied housing units in the block |
| **boroname** | Name of the New York borough. Manhattan, The Bronx, Brooklyn, Staten Island, Queens |
| **geom** | Polygon boundary of the block |

Task 3. Logical Operators

A screenshot of a computer

Description automatically generated

Question: Retrieve all census blocks where the total population exceeds 1000 and the number of owner-occupied housing units is greater than renter-occupied ones.

Task 4. Data Query Language (DQL)

A white text with black text

Description automatically generated

Question: List the total population and total housing units for each borough. Aggregate popn\_total and house\_total to each borough

Task 5. Data Manipulation Language (DML)

Question: Increase the number of renter-occupied housing units by 10% in all blocks where the borough is 'Manhattan'.

A screenshot of a computer

Description automatically generated

Task 6. Data Definition Language (DDL)

A white background with black text

Description automatically generated A screenshot of a computer

Description automatically generated

Question: Add a new column to the table to store the percentage of white population in each block. The output should display the attribute table with new columns.

**Own Your Own (6pts)**

Task 1. Data Manipulation Language (DML)

Question: Populate the new column 'perc\_white' with the percentage of the white population relative to the total population.

Task 2. Advanced Logical Operators

Question: Retrieve blocks where the total population is less than 500 or the percentage of the white population exceeds 70%.

Task 3. Combining DQL and Spatial Queries

Question: Find all census blocks in 'Brooklyn' with a total population density greater than 10,000 people per square kilometer.

Hint: Using ST\_Area(geom) to calculate the Area of each census tract

Task 4. Creating a New Table (DDL)

Question: Create a new table ‘borough\_summary’ to store summarized population data by borough.

|  |  |
| --- | --- |
| Column name | Data Type |
| boroname | VARCHAR(50) |
| population | INTEGER |
| housing | INTEGER |

Task 5. Inserting Data (DML)

Question: Insert data into the new ' borough\_summary ' table.

Task 6. Deleting Records (DML)

Question: Delete records from the 'borough\_population\_summary' table where the total population is less than 500,000.

Bonus (0.5pts): Using Subqueries

Question: Retrieve blocks where the total population is above the average population of all blocks.

Hint: To solve this, think about how to calculate the average population first using an aggregate function. Then, use a subquery to embed this average into the main query's WHERE clause to compare each block's population against it.

Hint: subquery syntax:

SELECT column\_name  
FROM table\_name  
WHERE column\_name expression operator   
 (SELECT column\_name FROM table\_name WHERE ...);