SQL Queries (1)
Querying Single Table

Akitaka Matsuo

Department of Government

Simple SQL Query

- Typically the query is like:

  `SELECT` … **`FROM`** … (**`WHERE`**…)**`;`**

  - **`SELECT`** chooses fields or types of computation (sum, count)
  - **`FROM`** determines which table to look up
  - **`WHERE`** is about data slicing with some conditions

Example queries

- `SELECT * FROM covid;`
  - Selecting all columns (*) and rows from covid table
    = Get the entire table as the result
- `SELECT fips, cases, date FROM covid LIMIT 5;`
  - Selecting three columns from covid table, but limit query result to 5 rows (i.e. df.head(5))

Notes on query language

- SQL commands are not case sensitive
  - e.g. SELECT, Select, select
  - Some prefer capital
- Fields are case sensitive
- Use single quotes

WHERE

- A conditional statement will be placed after where
  - `SELECT * FROM covid WHERE state = 'Alabama';`
- Operators
  - ‘=‘
  - ‘>’, ‘<‘, ‘>=‘, ‘<=‘
  - BETWEEN: Between a certain range (**`BETWEEN x AND y`**)
  - LIKE: Pattern matching
    - Use % for wild card (e.g. **`WHERE county LIKE ‘%lake'`**)
  - IN: Set evaluation

How we connect database

- The demo uses an SQLite database connected from Python

- Steps
  1. Generate connection object using **sqlite3** package
  2. Run the query with pandas **pd.read_sql_query()**
     - With connection object as one of arguments
     - Returning object is a pandas DataFrame

SQLite

- What is it?
  - A file-based RDBMS
  - Widely used in mobile applications
- Pros and cons (against MySQL etc)
  - Pros
    - No setup needed
    - Very portable
    - (Sort of) low latency
  - Cons
    - Lack of multi-connection capability
    - Slow to update tables

DISTINCT

- Returns distinct values in a column from a table
- Example:

```
SELECT DISTINCT state FROM covid;
```

ORDER BY

- Sort the results by the value of specific column(s)
- Could be ascending or descending order
  - Default ascending (**`ASC`**)
- Example:

```
SELECT *
FROM covid
WHERE date = '2020-09-01'
ORDER BY deaths DESC;
```

Computation

- Functions:
  - e.g. **`COUNT, AVG, SUM, MIN, MAX`**
  - Example:

```
SELECT MAX(cases)
FROM covid
WHERE state = 'New York';
```

- **`COUNT`**: Count the number of rows. Working with * or column names
- Working well with **`GROUP BY`** (see next)

GROUP BY

- It will group rows by the value of specific column
- Aggregate function works with it
- Example:

```
SELECT state, SUM(cases)
FROM covid
GROUP BY state;
```

Computing where?

- With pandas and SQL, we can execute the computation either in python or SQL
  - In SQL: run query to run computation
  - In python: get the data first, then do computation in pandas/NumPy
- Which is better?
  - Depends.
    1. Size of the data (Computation cost)
    2. Location of the data (Network cost)
    3. Optimisation of the databases