* Database type
  + Relational
    - Postgres (Open source)
    - SQL Server (Commercial)
    - SQLite (File-based)
  + Non-relational
    - MongoDB (JSON like)
* Postgres/PostGIS
  + Server configurations
  + Carto
  + Select all
    - SELECT \* FROM dcptransportation.citibike\_skill
  + Select columns
    - SELECT cartodb\_id, the\_geom, the\_geom\_webmercator, data\_stations\_lon, TO\_CHAR(data\_stations\_lon, '99D9') AS lon1, ROUND(data\_stations\_lon) AS lon2 FROM citibike\_skill
  + Filter
    - SELECT \* FROM citibike\_skill WHERE data\_stations\_capacity >= 20 AND boro IN ('Bk','Bx') AND (data\_stations\_name ILIKE '%ave%' OR data\_stations\_name LIKE 'Old Fulton S\_')
  + Sort
    - SELECT \* FROM citibike\_skill ORDER BY data\_stations\_capacity DESC, boro ASC
  + Join
    - SELECT station.\*, ridership.\_07\_06\_07\_1 AS citibikeridership FROM citibike\_skill AS station, citibike\_ridership AS ridership WHERE station.data\_stations\_station\_id = ridership.id
    - SELECT station.\*, ridership.\_07\_06\_07\_1 AS citibikeridership FROM citibike\_skill AS station LEFT JOIN citibike\_ridership AS ridership ON station.data\_stations\_station\_id = ridership.id
  + Aggregation
    - SELECT boro, MIN(cartodb\_id), AVG(data\_stations\_station\_id), SUM(data\_stations\_capacity), COUNT(data\_stations\_name) FROM citibike\_skill GROUP BY boro
  + DATE
  + With statement/ Common Table Expressions
  + Case when
  + Spatial
  + Insert/Delete/Update
    - INSERT INTO citibike\_skill (cartodb\_id, data\_stations\_short\_name) VALUES (10, 'abcd')
    - DELETE FROM citibike\_skill WHERE cartodb\_id = 10
    - UPDATE citibike\_skill SET data\_stations\_short\_name = 'abc' WHERE cartodb\_id = 1
  + Resources:
    - <https://carto.com/help/tutorials/using-sql/>
    - <https://carto.com/help/working-with-data/tips-for-geospatial-analysis/>
    - <https://www.postgresqltutorial.com/postgresql-joins/>
* SQLite/Spatialite
  + DBeaver
  + QGIS
* R