Week 7: Databases

02807 Computational Tools for Data Science

Exercises

1 Your first SQLite database

- Install the SQLite CLI on your machine using apt/brew install sqlite3
- Create a new database using the sqlite3 command line program.
- Create a new table "measurements" in the database with the columns "sensor_id", "time", and "value" with types integer, integer, and real.
- Insert at least 5 rows of data into the table using SQL.
- Create a SELECT query to retrieve all the measurements.
- Create a SELECT query that returns the all the measurements from a given sensor_id.
- Use an UPDATE statement to change the value of one of the measurements.
- Use a DELETE statement to delete one of the measurements.

2 Python and SQLite Repeat the steps from the above exercise, but this time use Python to perform all the queries. See `https://docs.python.org/3.6/library/sqlite3.html` for the sqlite3 package. You should use Python to nicely print the results of the SELECT queries.

3 Indexing experiments Perform the following steps using Python:

- Insert rows in the measurements table such that it has a least 1.000.000 rows.
- Construct a SELECT query that returns all measurements for a given sensor_id in a given time period (ie. where time is between a given stant and end value). Ensure you query returns around 100 rows by picking appropriate values for sensor_id, start and end.
- Measure how long time it takes to run the query (if it is too fast to measure, insert extra rows in the table)
- Create an index on the column (sensor_id) and see if it speeds up the query.
- Play around with different indexes and see how it changes the performance. Remember to delete the previous index before testing a new one (otherwise it is difficult to know which one is being used by the database). You could try to create an index on: time, the combination of sensor_id and time.

4 Tryout DB browser for SQLite (or similar) Try out a visual program to inspect your database - for instance: `http://sqlitebrowser.org/`. It is often easier to get a good overview of a database in this way. Make sure to also try inserting/editing data and creating a new table this way.

5 SQL Challenges We recommend you do the following using Python to interface the database such that you have an easy way to save your queries.

- Create a query that returns all measurements where a given sensor_id has a value above a given threshold.
- Create a query the returns the total number of measurements and the average of all values.

- Create a query the returns the minimum, average and maximum value of measurements for each sensor_id.
- Add the possibility of specifying a time range for the above query.
- Make a query that returns all sensor_ids for which the maximum value has been over a given threshold.
- Before you can solve the following challenges, you should create an extra table with the name sensors that has columns id of type integer, and name of type text. Insert names for the different sensor_id you have used in the measurements table.
- Create a query that return the last 1000 measurements. Extend the query to report the name of the sensor for each of the measurements (Use joining).
- Create a query that returns all measurements for a sensor with a given name. Extend the query such that it is possible to specify a time range.
- Create a query that returns how many measurements each sensor have.
- If you want more practice, you can find many exercises on `https://www.w3resource.com/sql-exercises/`.

6 Indexing challenges Consider how well an index on (sensor_id, time) will speed up the following queries (if at all). If the proposed index is not optimal for the query, suggest alternative indexes. You can take an experimental and/or a theoretic approach to this exercise. Discuss your answers with a teacher/TA:

- SELECT * FROM measurements WHERE sensor_id = 100
- SELECT * FROM measurements WHERE time > 100 AND time < 200
- SELECT * FROM measurements WHERE time > 100 AND time < 200 AND sensor_id = 100
- SELECT * FROM measurements WHERE sensor_id = 100 OR time > 200
- SELECT * FROM measurements WHERE sensor_id = 100 AND time > 100 AND time < 101 AND value > 100
- SELECT * FROM measurements WHERE sensor_id > 100 AND time > 200