CSC 455: Database Processing for Large-Scale Analytics
Take-home Final

Due 11:00pm, Tuesday, March 15th , 2015

Be sure to report the running times when requested in the questions.

1. We will use a full day worth of tweets as an input (there are total of 4.4M tweets in this file): http://rasinsrv07.cstcis.cti.depaul.edu/CSC455/OneDayOfTweets.txt

    a. Create a 3rd table incorporating the Geo table (in addition to tweet and user tables that you already have) and extend your schema accordingly.
    You will need to generate an ID for the Geo table primary key (you may use any value or combination of values as long as it is unique) for that table and link it to the Tweet table (foreign key should be in the Tweet table because there can be multiple tweets sent from the same location). In addition to the primary key column, the geo table should have "type", "longitude" and "latitude" columns.

    b. Use python to download from the web and save to a local text file (not into database yet) at least 500,000 lines worth of tweets. Test your code with fewer rows first – you can reduce the number of tweets if your computer is running too slow to handle 500K tweets in a reasonable time. How long did it take to save?
    NOTE: Do NOT call read() or readlines() without any parameters. That command will attempt to read the entire file and you only need 500K rows.

    c. Repeat what you did in part-b, but instead of saving tweets to the file, populate the 3-table schema that you created in SQLite. Be sure to execute commit and verify that the data has been successfully loaded (report row counts for each of the 3 tables).
    If you use the posted example code be sure to turn off batching for this part. (i.e., batchRows set to 1). How long did this step take?

    d. Use your locally saved tweet file (created in part-b) to repeat the database population step from part-c. That is, load 500,000 tweets into the 3-table database using your saved file with tweets (do not use the URL to read twitter data). How does the runtime compare with part-c?

    e. Re-run the previous step with batching size of 500 (i.e. by inserting 500 rows at a time with executemany). You can adapt the posted example code. How does the runtime compare when batching is used?

2.

    a. Write and execute SQL queries to do the following. Don't forget to report the running times in each part – you do not need to report the output:

        i. Find tweets where tweet id_str contains "44" or "77" anywhere in the column

Be sure that your name and Assignment number appear at the top of your submitted files.
Don't forget to add the text of each question as a code comment in each of your files

ii. Find how many unique values are there in the "in_reply_to_user_id" column

iii. Find the tweet(s) with the longest text message

iv. Find the average longitude and latitude value for each user name.

v. Re-execute the query in part iv) 10 times and 100 times and measure the total runtime (just re-run the same exact query using a for-loop). Does the runtime scale linearly? (i.e., does it take 10X and 100X as much time?)

b. Write python code that is going to read the locally saved tweet data file from 1-b and perform the equivalent computation for parts 2-i and 2-ii only. How does the runtime compare to the SQL queries?
c. Extra-credit: Perform the python equivalent for 2-iii
d. Extra-credit: Perform the python equivalent for 2-iv

3.

a. Export the contents of the User table from a SQLite table into a sequence of INSERT statements within a file. This is very similar to what you did in Assignment 4. However, you have to add a unique ID column which has to be a string (you cannot use any numbers). Hint: one possibility is to replace digits with letters, e.g., chr(ord('a')+1) gives you a 'b' and chr(ord('a')+2) returns a 'c'

b. Create a similar collection of INSERT for the User table by reading/parsing data from the local tweet file that you have saved earlier. How do these compare in runtime? Which method was faster?

4. Export all three tables (Tweet, User and Geo tables) from the database into a |-separated text file. In this part, you do not have to modify the table within the database, just output file data (do not generate INSERT statements, just raw data)

a. For the Geo table, create a single default entry for the 'Unknown' location and round longitude and latitude to a maximum of 4 digits after the decimal.

b. For the Tweet table, replace NULLs by a reference to 'Unknown' entry (i.e., the foreign key column that references Geo table should refer to the "Unknown" entry you created in part-a. Report how many known/unknown locations there were in total (e.g., 10,000 known, 490,000 unknown, 2% locations are available)

c. For the User table file add a column (true/false) that specifies whether "screen_name" or "description" attribute contains within it the "name" attribute of the same user. That is, your output file should contain all of the columns from the User table, plus the new column. You do not have to modify the original User table.

Be sure that your name and Assignment number appear at the top of your submitted files.
Don't forget to add the text of each question as a code comment in each of your files