ECS 116 Databases for Non-Majors / Data Management for Data Science

Programming Assignment 2
Part 1

A. Prelude

1. The assignment is of 25 points.
2. Last date of submission for Part 1 + Part 2 of this assignment is May 17, 2024, Friday @ 11:59 pm. You do not have to submit anything for Part 1, but we recommend that you try to finish Part 1 by Tuesday, May 7, so that you have plenty of time to work on Part 2.
3. Late submissions will be graded according to the late policy. Specifically, 10% of grade is deducted if you are up to 24 hours late, 20% is deducted if you are 24 to 48 hours late, and no credit if turned in after 48 hours.
4. This assignment will be in teams of 3 (with one or two teams of 2 or 4 depending on class size).
5. Each team member should have a full installation of the data and the code in their laptop, so that they can run everything. (Teammates might develop different parts of the code, but in the end each teammate should be able to run everything and get essentially the same results). In particular, all teammates will be required to create a performance report based on execution on their own machine.
6. Plagiarism is strictly prohibited. You're free to discuss high-level concepts amongst your peers. However, cheating will result in no points on the assignment and reporting to OSSJA.

B. Step 1: Uploading the AirBnB data for New York City into PostgreSQL

1. Download the 4 csv files in the GoogleDocs at
https://drive.google.com/drive/folders/14gWh0ck3vzWxyakaWHHH38AgWY7UC-IQ?usp=sharing
2. In DBeaver create a new database **airbnb**.
3. Upload the csv files for calendar, listing and neighborhoods into PostgreSQL.
   - As with the africa_ac_db.csv file, after a first attempt to upload one of these csv files into DBeaver you may have to modify the data types of various columns, then delete the data, and reload the data directly into the table in DBeaver. In general, for an "id", "listing_id" or similar fields use a data type "varchar(32)" (or perhaps larger if necessary). For any date fields use data type "date", so that you can make queries based on date ranges.
4. As a sanity check, you can confirm that the number of records in your database tables correspond to the number of records in the csv files:
   - Calendar should have 14,299,870 records
   - Listing should have 39,202 records
   - Neighborhood should have 230 records
   - NOTE: you can check the number of lines in a csv file from a terminal window as follows:
     - For mac, in a terminal window type, e.g., `wc <filename>`. This gives the line count, the word count and the character count
     - For PC, in a powershell window type `type <filename> | find /v /c ""`. (See https://jordenthecoder.medium.co to-count-number-of-lines-of-large-data-files-in-windows-660ae8a3f4f7.)
5. Now work on uploading the file reviews.csv into PostgreSQL

- On the first attempt to upload using DBeaver you will get errors talking about "integer out of range". Use the "Skip All" choice so that DBeaver will create the relation schema for reviews
- Check on the table - how many tuples does it have in it? What are the columns and data types?
- Since all of the integer columns are some form of ID, let's convert all of them to varchar(32)
- Empty the table and reload the csv file
- Now you get an issue about character length of the "comments" attribute. Resolve this by
- Now DBeaver complains about a row that appears corrupted, in particular because it looks like part of a comment is in the listing ID. Note also that if you do "skip" then the entire batch holding the offending tuple will be dropped.
- At this point it probably makes sense to try a different approach for loading the csv file!
- Try using the sql command COPY that loads directly from a csv file.

```
COPY reviews(listing_id, id, date, reviewer_id, reviewer_name, comments)
FROM '<full path to csv file>'
DELIMITER ','
CSV HEADER;
```

- Notice the error that a comment is > 4096 characters. Make that datatype `varchar(10000)` and try again.
- Finally, examine the data type and values in the "date" columns of the calendar table and the reviews table. Change the data type in reviews to match the data type in calendar. This will enable selections joins on date fields.
- NOTE: In the above steps you used both DBeaver and the COPY command to upload data. Sometimes one works, sometimes the other, so we wanted you to have practice with both.

C. Step 2: Setting up test harness and related infrastructure

1. The goal of this step is to set up a "test harness", that is, an environment that will make it easy to run a range of difference performance tests (also called "benchmarking"). The testing will involve

   - A handful of queries and updates against the AirBnB data, and
   - A handful of indexes that can be added to (and dropped from) the AirBnB data.

   During the benchmarking you will be exploring the impact (positive or negative) of having some indexes in place, when executing both queries and updates.

2. To set up your test harness, you should follow the examples in the notebook Prog-ass-2–v04–mostly-without-util.ipynb, which is in the PROG-ASSMT-2 area of Canvas. The notebook is self-explanatory. Also, Aunsh has gone over the notebook in the Discussion of May 2, 2024.

D. Step 3: Doing performance testing & capturing performance results

(Will be filled in for Part 2.)

E. Step 4: Visualizing performance results

(Will be filled in for Part 2.)

F. Submission

(Will be filled in for Part 2.)