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

Problem Set 5

This exercise is actually the first part of Programming Assignment 3. We want you to get started on this work during the week of May 27, so we are giving this to you as a Problem Set.

Programming Assignment 3 will be significantly shorter and easier than Programming Assignment 2.

Rick (that is, me) had hoped to include working with Amazon Web Services (AWS) as part of Programming Assignment 3, but I feel that there is not enough time left in the quarter. I will be presenting some material about AWS in a lecture. Also, if you want to get some experience with AWS, then I can guide you through some basic steps, including setting up an account, getting DocumentDB set up (DocumentDB is the AWS version of MongoDB), and enabling you to work with it. Please send me an email if you are interested in that.

As with Programming Assignment 2, you may work in teams of size up to 3 people on Programming Assignment 3. (You may also work solo or in a team of size 2.) You can work with your teammates as you complete this Problem Set 5.

This problem set is is worth 2 points, and is due by 11:59PM on Sunday, June 2, 2024. (We suggest that you complete this earlier, because the full Programming Assignment 3 will be posted on or before Friday, May 31, and the remainder of Programming Assignment 3 may take a bit longer than this Problem Set 5.)

For this exercise, please use the file `Loading Local MongoDB with Listings & Reviews-vXX.ipynb` in the PROBLEM-SET-5 folder in Canvas.

As you will see, the notebook illustrates how you can use `pymongo` and PostgreSQL to combine the `listings` and `reviews` data to make a large and somewhat interesting collection in MongoDB.

For this exercise you are to:

1. Create a notebook that populates the MongoDB on your laptop with the full set of listing objects as illustrated in `Loading Local MongoDB with Listings & Reviews-vXX.ipynb`.

2. Once you have MongoDB populated, you are to add 4 cells to your notebook with `pymongo` scripts/queries as follows:

   - Query 1: Output is the number of listings that have `last_review` between February 1, 2021, and March 15, 2023, inclusive.
   - Query 2: Output is the number of listings that have an array of reviews with length at least 50. (You may want to take inspiration from https://stackoverflow.com/questions/41918605/mongodb-find-array-length-greater-than-specified-size.)
   - Query 3: Output is the number of listings that have a review containing the word "awesome" (case sensitive) OR a review containing the word "amazing" (case sensitive).
   - Query 4: Same as Query 3, but ignoring case.

   Finally, **each team member should run the notebook on their own machine.** (This is a slight revision of the instructions from a previous version of this problem set.)

   Submission instructions: Each team member should prepare a zip file named **prob_set_05.zip** which contains three files, as follows.

   (a) A very short pdf document that lists the members of the team you worked with (including your own name). This file should be named **prob_set_05.pdf**.

(b) The jupyter notebook that you used to populate your MongoDB and produce the results for the 4 queries. Please name this file **prob_set_05.ipynb**.

(c) Please prepare a csv file and name it as **prob_set_5.csv**. It should have the following properties:

   i. The csv file has a header row.

   ii. Column 1 is labeled "Query Number"

   iii. Column 2 is labeled "Count"

   iv. There are 4 rows, corresponding to the 4 queries, in that order.

   v. The entries of the first column are simply 1,2,3 and 4.

   vi. The entries of the second column should be the counts that you obtained for each of the 4 queries by running your notebook on your machine.

   Here is an example of the format for the csv file that you are to use. The Count values here are completely made up.

Query number	**Count**
1	10
2	150
3	23
4	104

Table 1: Example structure of table in the csv file to be submitted

   (You can create the csv file by hand, i.e., you do not need to have your notebook produce it.)

Note: We have learned that When you submit your zip file into Canvas, then Canvas automatically generates a filename which includes an encoding of your name. That's why we are not asking you to include your name in the filename of the zip file that you submit.