CMPSC 312
Database Systems
Fall 2020

Lab 3 Assignment:
Relational Data Modeling for Protein Data (using lab 2)
Submit deliverables through your assignment GitHib repository.
Place report document `writing/` directory

Objectives

To learn how to add more tables to a previously created database in SQLite3 to store downloaded data.

GitHub Starter Link

https://classroom.github.com/a/3Amrnni5

To use this link, please follow the steps below.

- Click on the link and accept the assignment
- Once the importing task has completed, click on the created assignment link which will take you to your newly created GitHub repository for this lab,
- Clone this repository (bearing your name) and work locally
- As you are working on your lab, you are to commit and push regularly. The commands are the following.
  - `git add -A`
  - `git commit -m ‘‘Your notes about commit here’’`
  - `git push`

Introduction

Often when you create a database, you will find it necessary to add extra tables to be able to enhance its use. In this lab, you will modify the SQL builder code from your previous lab (i.e., the file `proteinBase_build.txt`) to contain two additional tables, containing the downloaded protein data from `https://www.uniprot.org/` from searches for *m-protein* and *sars*. We note that these two additional tables will contain protein data which we believe to have connections to the original data from the original database.

Data Sets

Your database must be re-designed to hold new protein data from UniProt at `http://www.uniprot.org`. To obtain your additional data to be added to your original database, please open your browser and find the UniProt website and complete searches for *m-protein* and *sars*. Listed below, you will find the links for the datasets that are to be contained in your SQLite database, if you have trouble using the search feature on the Uniprot website.

Data References

Your database is to contain the data from the following sources.

- From last lab, *n-protein*: `https://www.uniprot.org/uniprot/?query=n-protein&sort=score`
- From last lab, *s-protein*: `https://www.uniprot.org/uniprot/?query=s-protein&sort=score`
- New table for *Sars*: `https://www.uniprot.org/uniprot/?query=sars&sort=score`
- New table for *m-protein*: `https://www.uniprot.org/uniprot/?query=m-protein&sort=score`

Downloading and Formatting

Please download the returned protein data from your searches at UniProt using the un-compressed, `tab-separated` options as shown in Figure 1. Please keep all your database building files in your GitHub classroom repository and not in the course *ClassDocs* repository.

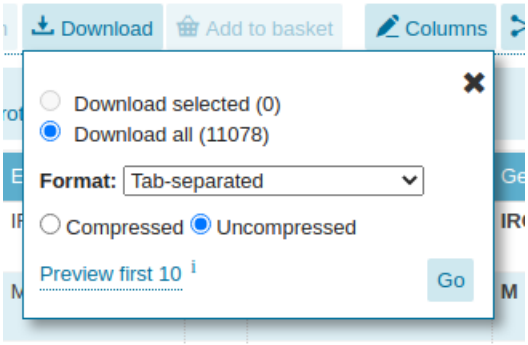

Figure 1: Download the data using the options for un-compressed and the tab-separated format. Please do not include these files in your submission repository as they are not necessary to evaluate your grade.

Trimming the File-Headers

As before, you will need to remove the first line of header information from each downloaded file. Your downloaded files have the column headers given on the first line which must be removed before you can use them to build your database. To do this, load each file in a text editor such as `Atom` and remove the top line of the file. This line will contain the terms; *Entry*, *Entry name*, *Protein names*, and etc., and were added in the file by Uniprot as column headers. Save your work as a text file.

Preparing and Compiling a Build-file to Create Your Database

Since you now have two additional tables to place in your database from last week's lab, you will have to modify your builder code (i.e., the file `proteinBase_build.txt` from your previous lab), to accommodate your new tables. This procedure is very similar to the one that you utilized in your previous lab. Please note, in this step, you are creating a new builder file that contains the same code from the previous lab, in addition to the new code for your new tables. Your data files from the previous lab will also have to be copied to over this lab's repository, along with the new files that you obtain for the proteins, *m-protein* and *sars*. Please make a directory in your GitHub Classroom repository called `data/` in which you will store these data files and will be used by your builder file to create the database. The full path of this directory in the repository will be `src/data/`. Be sure to update the `.import` lines of your builder file's code to import the four data files (i.e., two files from the previous lab, and the two new files from this lab.)

Your new build file will be called `proteinBase2_build.txt` and you will build your new database using the terminal command;

```
cat proteinBase2_build.txt | sqlite3 proteinDB2.sqlite3
```

Please note, you will not be using this command in the SQLite environment, this command is to be used at the TERMINAL (unix) prompt in your Docker container.

Querying the Database Tables

Now that you have built a successful database, please answer the following questions-in-blue. When grading, the instructor will often be concentrating on the structure of your query, since the results may change depending on the age of the data. Note, to access your database with SQLite3, you will use the bellow command at your TERMINAL prompt.

```
sqlite3 proteinDB2.sqlite3
```

The following questions assume that the names of the tables are arranged in the following ways.

- m-protein: mprot
- n-protein: nprot
- Sars proteins: sars
- s-protein: sprot

1. In your own words, write the questions that the following queries answer;
   - select count(sa.protID) from nprot n, sars sa where sa.protID == n.protID;
   - select count(distinct(sa.protID)) from nprot n, sars sa, mprot m where sa.protID == n.protID AND sa.protID == m.protID;
2. Please give an obvious reason why the two queries below have the same output.
   - select count(distinct(s.protID)) from sars s;
   - select count(s.protID) from sars s;
3. Write the query to list all distinct *Organisms* which are common to both the `sars` and `nprot` tables.
4. Write the query to count all distinct *Organisms* which are common to all tables: `sars`, `nprot`, `mprot` and `sprot`.
5. How many distinct protIDs are there in the `sars` where the organism == "Mus musculus (Mouse)"?
6. What is the average length of the proteins in the `sars` table for the organism "Mus musculus (Mouse)"
7. What is the largest length of the proteins in the `sprot` table for the organism "Mus musculus (Mouse)"
8. List the protIDs and their associated organisms from the `sars` table where the length of the protein is equal to 240.

9. List the protIDs and their associated organisms from the `nprot` table where the length of the protein is equal to 220.
10. Running queries can help you find patterns that you may not expect to find. What natural trend(s) did you notice from the result of your two previous queries above?
11. Write a question of your own that uses all four tables to answer, then provide the query to respond to your question involving all four tables.

Summary of the Required Deliverables

Please submit your work by pushing it to your GitHub Classroom repository.

1. **Query and Results document**: You will modify the file `writing/queries.md` to respond to the questions-in-blue, above.
2. **Database-building file**: You will submit your edited build file (`src/proteinBase2-build.txt`) to be used to build your database from your data files.

In adherence to the Honor Code, students should complete this assignment on an individual basis. While it is appropriate for students in this class to have high-level conversations about the assignment, it is necessary to distinguish carefully between the student who discusses the principles underlying a problem with others and the student who produces assignments that are identical to, or merely variations on, someone else's work. Deliverables that are nearly identical to the work of others will be taken as evidence of violating Allegheny College's Honor Code.