CMPSC 380
Principles of Database Systems
Fall 2016

Laboratory Assignment Three: Using the SQL Data Manipulation Language

Introduction

In our course this week, you were given Python code (filename: `02_campusQuery_selectSpecifics.py`) which demonstrates how to implement SQLite database queries using the Python library `sqlite3`. In today's lab, you will explore this code to be applied to the `campus_ii.sqlite3` database. This code will be modified to automatically query of all tables in the database. We have already discussed how to create a database from a text file. The database that we will use today may be re-created from `setupDB_ii.txt` (included with this lab). This lab has been divided into two parts. This part concerns the usage of Python code for automated queries and the second part concerns the usage of web pages code to display and modify the current database.

This is a team-based lab, you must work in teams of two (preferably) or three people. You may select your own team member(s).

The program code

Prepare and use a copy of the source code file `02_campusQuery_selectSpecifics.py` which has included in this lab (and also given out for discussion in class). This is the code partially discussed in class which provides examples of how to performs types of queries in Python. In this source code file, there are several variables defined near the top of the file (i.e., *sqlite_file, table_name, id_column, some_id, column_2 and column_3*). Each of these variables contain strings which are used by (*plugged-into*)the `execute` statements of the code. In the query statements below, we note that the variables *table_name* and *column_2* contain strings used to build the query string that is sent to the `execute` method.

```
c.execute('SELECT * FROM {tn} WHERE {cn}="Nelson"'.format(tn=table_name,
cn=column_2))
all_rows = c.fetchall()
print('1):', all_rows)
```

The variables are below which are plugged into the `SELECT` statement code, shown above.

- *table_name* = "*instructor*" and
- *column_2* = "*name*"

For each of your queries, your code should resemble the following type of programming.

```
table_name = 'instructor'   # the table to be queried
column_2 = 'name'           # the attribute to query

c.execute('SELECT * FROM {tn} WHERE {cn}="Nelson"'.format(tn=table_name,
cn=column_2))
all_rows = c.fetchall()
print('1):', all_rows)
```

Using a system of variables to plug-into query statements, we are able to make the query statements (themselves) very generic and simple because the execution string is always very similar across queries. The query themselves are different from each other when the when their substituted variables, containing tables and attribute information, are applied. In this lab, we will follow this theme of programming: our query statements will look very similar. The correct query is performed when the correct variables, containing the tables and variables, are set just above the query statements. Coding in this way will help you to keep your code simple, and also will help you to debug your query code much more easily when necessary.

Tables to Query

Coding helps to simplify repetitive work. The Python source code contains 8 basic examples of using Python to run queries over database tables. Your task is to apply each of 8 query examples to query each of your 6 tables from the *campus_iii.sqlite3* database.

1. `course`
2. `department`
3. `instructor`
4. `student`
5. `takes`
6. `teaches`

For this activity, you and your partner are to decide what information to query for each table and then write the code made up of `execute` or `update` commands. The actual information is not a factor for your instructor who is only looking for proof of mastery of skills to utilize Python for automatic querying.

Given Query Examples

The modules from the Python program take the following form.

1. SELECT * FROM instructor WHERE name = "Nelson";

2. SELECT name FROM instructor WHERE student =“S1”;
3. SELECT name, student FROM instructor WHERE name == “Watson”;
4. SELECT * FROM instructor WHERE ID like “101%” limit 3;
5. SELECT * FROM instructor WHERE Name like “__ll__”;
6. SELECT * FROM instructor WHERE Id =“10110”;
7. SELECT * FROM instructor WHERE Id =“007”;
8. UPDATE instructor SET salary == ”101” WHERE name == “Farber”;

Each of these examples must be applied to each of the above tables of your database. You are to create a python source code (as discussed in Section 1) in a single file that the instructor can run to test. The instructor will be looking for generic query strings which use variables to define tables and attributes.

Summary of the Required Deliverables

This assignment invites you to submit an electronic version of the following deliverables through Bitbucket. Please create a subdirectory in your “labs” directory called “lab4_part1’. Please also submit one lab response from each team. *Here, you will have to decide which member of the group is going to make this submission for your group.* Do not forget to add your names to this typed-up submission which is preferably, written in LaTeX. Submit the following materials.

1. One source code file which prints out all your queries. The instructor should be able to run this source code to view all the eight queries over each of the six tables.
2. Convincing evidence to demonstrate that your Python-driven query statements ran correctly. For instance, when changing some entry in a table, show the entry before and after an update.
3. A brief reflection on the challenges that you faced when completing this laboratory assignment.

You must place all of the required deliverable into a Bitbucket repository directory named `lab4_part1`. Please see the instructor if you have any questions about this assignment.