Lab 3 Handout: Objects and Databases

Due: 1 November 2018

1 Objectives

- Use SQLite in Python for permanently storing information.

2 Task description

Instructor - Course - Student - SQLite. This lab is a follow-up to Lab 2, where you implemented three classes: Instructor, Course and Student. A student derived from Student can select or drop courses. An instructor derived from Instructor can teach courses. A course derived from Course can enroll and remove students. In Lab 2, all data are stored in memory. In this lab, we want to store them in a database, which we can view, edit and analyze later. To this end, we use the `sqlite3` library in Python, which includes methods that can help us interact with SQLite databases.

We can import the `sqlite3` library in the following way: `import sqlite3 as sqlite`. SQLite is a lightweight, portable relational database management system. Refer to SQLite Tutorial to get started and The Python Standard Library for more details. In particular, the SQL statements such as INSERT, INSERT OR REPLACE, INSERT OR IGNORE, SELECT and JOIN are useful for this lab.

You decide which tables to include in your SQLite database and design columns for each table. Creating an object adds a record to one of these tables, and calling a class method (such as `select`, `drop`, `enroll`, `remove`, and `teach`) updates a table.

The special method `__str__` in each class retrieves information from relevant database tables and return it as a string. For example, `print(s)` displays the student name (student number) followed by a list of courses he has selected. Each course contains its course number, course name, the semester in which this course is offered, and the instructor's name. `print(c)` displays the course number, course name, semester and instructor's name followed by a list of enrolled students. `print(t)` displays the instructor's name, followed by a list courses under his name. Include its course number, course name, semester, and time and location for each course when calling `print(t)`.

You should prevent the database tables from adding duplicated records.

You can manually create a database and then a few tables with DB Browser for SQLite.

The starter code for this lab can be downloaded from the course homepage.

3 Requirements

- Do the lab in a group with a maximum of 2 students.
- Make sure the commented statements below `if __name__ == '__main__'` in the provided starter code produce reasonable results.
- I only accept object-oriented implementation.
- The partial work is due immediately after the lab. The completed work is due on November 1.
- Submit your work with the following email subject line: `[OO Lab3] STATUS Name(s) and student number(s)`, where `STATUS` can be `PARTIAL`, `COMPLETE`, or `PARTIAL&COMPLETE`.
- Submit by the due date a working python source file, namely, `Lab3.py`.
- Submit by the due date a SQLite database, namely, `school.sqlite3`, containing the result after the program finished executing the statements after `if __name__ == '__main__'`.
- Submit by the due date a text file containing the console output from your python program.

- Question: how did you avoid duplicated deletion from or duplicated addition to database tables? For example, the call `c1.enroll(s1)` following `s1.select(c1)` would cause duplicated operations on a table. Please append your answer in the above text file.