CSCA20 Assignment 2

Due: August 3, 2014. 5:00pm

Some Notes Before We Begin

By this point you should be familiar with this little memo. There will be no pre-marks, and it's your responsibility to ensure that your functions work exactly as they should, names, etc. Also, as with the previous assignments, documentation and style will contribute heavily to your mark.

Your assignment

The work you did in A1 was very nice, but now USTC[1] wants an update. They find that reading the files every time they want to boot up SORI makes everything too slow. And the dictionaries become memory hogs after a while. So they want you to improve the system by using databases.

As before, they have provided you with two files for this task, `enrolment.csv` and `grades.csv`. This time they also have two more files: `faculty.csv` and `teaches.csv`. They assure you that the files are properly formatted `csv` files, so you don't have to worry about getting bad data.

The Database

You can design the database in any way you see fit, but a substantial part of your mark will be assigned to having a sensible design. If you store everything in one massive table, your mark will suffer for it.

Functions

In order to complete this assignment, you must complete the following functions. You may use helper functions if you wish, but the following functions must be present. There will be more than one way to complete each of these functions. In particular, you could do almost all of the work in your SQL queries, or almost all of the work in Python. In general, letting SQL do the work for you will result in the best marks (remember, we're lazy computer scientists).

read_csv_data

`read_csv_data` takes four files as parameters[2]. The files will be `csv` files formatted exactly as `enrolment.csv`, `grades.csv`, `faculty.csv` and `teaches.csv` (in that order). The function should read these files, and create/populate a database with appropriate tables to hold all of the files' information. The function should return a cursor for this database.

add_record

`add_record` takes three parameters, an sql cursor, a string representing the table to which we'd like to add information (one of: {"ENROLMENT", "GRADES", "FACULTY" or "TEACHES"}), and a tuple of strings representing the data that we'd like to add, in the same order as given in the respective csv files. The function then adds that record to the database. You may assume that all data is reasonable and correctly formatted.

[1] University of Scarborough, Toronto Campus

[2] As in the exercises, remember that these are file handles, not the names of the files. The files will be opened and closed by the `interview` function, or the automarker, this function should not open or close any files

find_my_dgpa

`find_my_dgpa` takes as its parameters an sql cursor and a string representing a student number. It returns the dgpa (see A0 for definition) for that student as a float.

find_lecturers

`find_lecturers` takes two parameters, an sql cursor and a string representing a student number. The function returns a set of strings representing the names of all of the faculty members who have lectured courses taken by the student with the provided student number.

lecturer_avg

`lecturer_avg` takes an sql cursor and a string representing the name of a lecturer as its parameters, returns a float representing the average mark of any student taking a course taught by that lecturer.

interview

Similarly to your previous assignments, this function acts as your interface into the system and should allow users to do things like add records, run queries, etc. This function is the only place in your code where you should use `print`, `input`, `open` or `close`. Your goal is to make this function as user-friendly as possible. You will be marked on your design, and so even if you can't get the other functions to work, you should still write the code to show how this function *would* work, even if it's printing dummy data.

What to hand in

All of your code should be in a file called `a2.py`. The code should not print anything or ask for any input when imported (any testing code should be inside an `if(__name__ == "__main__")` block). The only modules you may import are `csv` and `sqlite3`.