Spring 2023 CSE 480
Project 5: DBMS - Cont.

April 3rd, 2023

---

1 A Word of Warning

Make sure you read through the submission instructions thoroughly before you submit your project. Failure to comply with the submission instructions will result in a zero for this project.

This project can build directly off of your Project 3 or Project 4. I also highly recommend that you comment your code thoroughly so you can remember how your code works in the future. As your project 6 can build off of this project.

We will not be releasing reference solutions to projects, so if you do not do a project, or do it poorly, you will have to finish that project before starting the next project. Because of this, we will offer a resubmission of this project one week after the initial deadline. See the class syllabus for how resubmissions will be graded. Additionally, we will give more help in office hours after the project deadline to help you finish a previous project that you might not have completed properly.

2 Project Due Date

This project is due before April 14th at 11:59pm. The second-chance deadline for this project is April 21st before 11:59pm. Submission instructions for this project can be found at the end of this document.

3 Project Overview

This project will emulate the behaviour of the built-in python module, "sqlite3". Your module will be able to execute SQL statements corresponding to: default values, views, parameterized queries, etc.

4 What You are Given

On D2L we have released some test cases, and a script to run the test cases. We have also released 'regression' tests, which you can use to determine if your updates for Project 5 have broken any previous functionality. You are responsible for creating your own test cases as well to test the robustness of your solution. There are edge cases that we did not release test cases for, and it is up to you to consider all of these and make sure your solution will work in all conditions.

Please note that this cli.py script is different than the previous scripts, so make sure you download the new cli.py for testing.

5 What You Need To Do

You need to build the functionality for Project 5 on top of your Project 3 or 4. I highly recommend that you make a copy of your previous project in case things go wrong.

All of the code for this project must be in 'project.py'. There is no need to write any database output to any files. Each test will be done on a fresh database.

Your program is allowed to use any built-in modules of python, with the exception of sqlite3. sqlite3 is the reference implementation for this project, your output will be compared against it for correctness. Importing the sqlite3 module in project.py will be considered academic dishonesty for these project, which will lead to an automatic failure of this course.

You cannot use any modules of python that are not built-in, meaning you cannot use things like Numpy or Pandas.

If you use any resources that did not come directly from the class, you must specify and cite them in your code file. At the top of the starter code there will be a section to paste links to anywhere you got code or ideas from. This is to make sure that you are not plagiarising.

6 Test Categories

6.1 Regression

These tests should pass if you completed Project 3. They should pass automatically at the start of the project. If these fail, you've made a regression (you have broken previously functional code).

6.2 DESC (Descending Order)

The DESC keyword indicates that the collation should be reversed, so that values are ordered in a descending manner instead of the default (ascending).

SELECT * FROM students ORDER BY name DESC, grade DESC;

6.3 Default Values

It is often useful to supply non-NULL default values for specific columns. For instance, if no grade is supplied for an insert statement, I may want to assume a grade of 0.0. The way you specify the default value for a column is in the CREATE TABLE statement:

CREATE TABLE students (name TEXT, grade REAL DEFAULT 0.0, id TEXT);

Additionally, if you want to insert a row with entirely default values, you can do so like:

INSERT INTO students DEFAULT VALUES;

Note that this requires that you've completed the insert-into columns functionality from project 3.

6.4 View

Views are read-only named SELECT statements. They act like a table, but if any of the tables the underlying SELECT statements it draws from changes, the results returned are changed (on subsequent queries). Example:

CREATE TABLE students (name TEXT, grade REAL);
CREATE VIEW stu_view AS SELECT * FROM students WHERE grade > 3.0 ORDER BY name;
SELECT name FROM stu_view ORDER BY grade;

For simplicity, none of the columns of the view's underlying tables will share names.

6.4.1 Parameterized Queries

Parameterized queries make it easy to reuse queries with wildcard (?) values pulled from the variables in the programming language interface. You need to implement a executemany method on your Connection class that accepts two arguments: a SQL statement with wildcard placeholders, and list of tuples with the values that should be slotted in. Example:

conn.execute("CREATE TABLE students (name TEXT, grade REAL, class INTEGER);")
conn.executemany("INSERT INTO students VALUES (?, ?, 480);", [('James', 3.5), ('Yaxin', 2.5), ('Li', 3.0)])

6.5 Aggregate Functions

You need to support the aggregate functions: min and max. Because we aren't implementing the GROUP BY clause, any SELECT statement using an aggregate function will only return a single row.

Example:

SELECT MAX(grade) FROM students ORDER BY grade;

You can assume that the MIN and MAX keywords will always be fully capitalized.

7 Testing Your Code

7.1 Testing Yourself

We've provided a few sample sql transactions as well as a python script to test your code.

To run your code with the testing files provided:

python3 cli.py <filename>

Where the second argument is the .sql file to test. Each .sql file is a set of sql commands that compose one transaction. We also provide a way for you to compare against the ground truth, which is python's sqlite3 module. To run the ground truth, use the following command:

python3 cli.py <filename> --sqlite

Your output should be identical to the output from the sqlite3 module. In the cli.py file we have a few print statements so you know if you are using your own code or the ground truth sqlite3 module. You can remove this code if you like, in our final testing we will not have these print statements, of course.

You can, and should create your own test cases as well.

7.2 Instructor Testing

When we test your code, we will run it in almost the exact same way, just with more test cases.

8 Assumptions and Guarantees

All tests will be legal SQL (no syntax errors, inserting into nonexistent tables or data type violations).
All select statements will have "FROM" and "ORDER BY" clauses
All table and columns names will start with a letter (or underscore) and be followed by 0 or more letters or numbers or underscores

9 Submission Instructions

You will submit a file named "project.py" to the handin (handin.cse.msu.edu). All of your code should be in the "project.py" file. If you used any external resources (something other than class material), make sure that you cite the resource in a comment at the top of your "project.py" submission file. The top of the starter code also has space to put your name, netid, PID, and an

estimation for how long the project took you to complete. Make sure you fill this out fully before submitting your project.