Spring 2023 CSE 480
Project 6: Persistence

April 17th, 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, Project 4, or Project 5.

2 Project Due Date

This project is due before April 28th at 11:59pm. There is no second-chance submission for this project. 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 handle database persistence as well as multiple databases at the same time. Please note that in addition to the sqlite3 module not being allowed, the pickle module is also not allowed for this project.

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 6 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 6 on top of your Project 3, 4 or 5. 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'.

Your program is allowed to use any built-in modules of python, with the exception of sqlite3 and pickle. sqlite3 is the reference implementation for this project, your output will be compared against it for correctness. Importing the sqlite3 or pickle 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 Functions

You will need to implement the 'close' function on the Connection class. When the 'close' function is called, you should write your current database into a file on disk based on the filename sent in when the connection was started. I would recommend putting your database into the json, xml, or csv format, however you can choose how you want to save your database, with the exception of pickling it.

When 'close' is called, your database should be written to a file in a format of your choosing and the database should be closed. When a connection is started, you should check to see if a database with the given filename has been created, then you should read in the file from disk and recreate the database.

7 Test Categories

7.1 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).

7.2 Create Database

In these test cases, a database will be created, some tables will be created, and some queries will be performed. At the end of the case the 'close' method will be called on the connection. This means you should write the database to a file on disk and close the database as it won't be used again in this test.

At the beginning of the create database test any existing DB file of the same name will be deleted.

To pass these tests you have to return the proper output from the queries. These tests should pass in general if your project 3 functionality is complete. Therefore, these tests are worth significantly less than the read database tests.

The first line of this test case will be: 'FILENAME: <filename.db>' and will specify the name of the database. The last line of this test will be '<connectionid>: CLOSE' and will tell the connection to close. Your code will not see these first and last lines, similar to the parameters test cases from project 5, they will be used for our script to create or close a database.

7.3 Read Database

In the read database tests, a database will have already been created from a 'create database' test. You will read in the database from the file and perform any querying on the database.

You need to check whenever a connection object is created whether a database of that filename exists already. If it does, then you should read in the file and create your database.

The first line of this test case will be: 'OPEN: <filename.db>' and will specify the name of the database. The last line of this test will be '<connectionid>: ENDTEST' and will tell the script to end and delete the database file on disk.

7.4 Create Multiple Database

In these test cases, multiple databases will be created and used in different connections. There is no need to implement locking, as multiple connections will only modify different databases.

The first filename specified in the test will be used in the first connection, and the second filename specified will be used in the second connection. You can assume that when a new filename is specified then the next query will always be from a new connection.

These test cases will always end with both connections calling the 'close' function.

7.5 Read Multiple Database

In these test cases, you will read from one of the databases created in the Create Multiple Database test case. However, you will only read from one database at a time.

8 Testing Your Code

8.1 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.

For testing you should run the sql files in pairs (based on test number). For example, test.create_database.01.sql will create a database of a particular filename, and test.read_database.01.sql will read the database that was created in the previous case. This also holds for the multiple database test cases, however in the read cases they will each read from one of the databases from the create case. For example: test.create_multiple_database.01.sql will create two databases, and test.read_multiple_database.01.a.sql will read from one of the databases and test.read_multiple_database.01.b.sql will read from the second database.

In general you won't be able to test the read database tests until your create database tests have been completed.

8.2 Instructor Testing

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

9 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.

10 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.