Nick Alvarez
CS 657
D. Zhao
4 May 2021

Programming Assignment 4
Design Specification

This program's purpose is to continue the design of a SQL type database in Python. Basic commands were implemented in PA1, including those to create or delete databases and tables, as well as to query and modify the tables. In PA2, table contents could be updated and queried more effectively. In PA3, table joins were implemented. Finally, in this revision, transaction functionality was added to the program.

The only task in this programming assignment was to add transaction functionality. This was a multi-step process. First, the program defaults to no locks being in place made by the user. With each command execution (more useful for CLI mode), the program checks if any file locks have been made. If they have, set the "isLocked" flag to true. However, if the user created these locks by beginning a transaction, leave it set to false.
On this topic, when a transaction is begun, the program checks if there are any locks in place. If not, locks are created for every table in the database and the "userMadeLocks" flag is set to True. Since this flag is true, the "isLocked" flag will never be true, and a commit will be successful. When committing, the program checks if there is a lock made by the user, and if there is, it will release the locks and run any pending commands. If there was no lock made by the user, the transaction is aborted.
Transactions require that tables be modified in-memory without any changes to the database until committed. Table commands, like inserting, altering, or removing tuples, are affected by this. If a transaction is in progress, any changes made to the tables will be stored in a separate file named "<Table>.new.txt" and a command will be created to remove the existing table and replace it with the new one. These commands are placed into a list that are executed one by one when the user commits.

Rollback functionality was not included, however, addition would not be difficult. Simply make a system call to delete any file ending in .new.txt. Dropping of tables was not included either, but would follow a similar structure as anything relating to table modifications, wherein system commands would be saved and executed later. With the table modification commands, the user can only make one change per table before committing. This would need to be resolved with some kind of flag to see if a change had been made beforehand.

Execution Commands
There are two options for execution: the included command line interface or file input.
Command Line Interface:
In a Unix environment, run the command
```
python3 main.py
```
The user can type commands as they please.
File Input:

In a Unix environment, run the command

```
python3 main.py SomeFile.sql
```

Every command is executed in order in the file.

Python 3.7 or newer is required due to some of the syntax and functions used.

Ensure that all the modules are in the same folder (dbutils, tableutils, joinutils and queryutils).

Included is the original PA4_test file (for CLI use) and test files for each sequence of the testing (for file input).

The included PA4_test.sql file contains the same commands but is modified in the following ways:

- Flights table was represented as both “Flights” and “flights”. Changed to uppercase F only.