Nick Alvarez
CS 657
D. Zhao
23 March 2021

# Programming Assignment 2
## 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 this revision, table contents can be updated and queried more effectively.

The program builds off PA1 with the use of text file tables. Since each line in the file represents one tuple, creating, modifying, or deleting the tuples is very simple. As the projects get more complex, reading the files in, storing their values in a Table class, and writing them later could be a more efficient method, but the string and list based modification currently used is sufficient and quite modular.

The approach used in PA1, where commands are turned into a list and parsed, proved to still be useful. With commands like Select, where there can be any number of variables, parsing had to be adapted to trim the beginning and ends of that variable list based off known delimiters, like "select" and "from." Once all information is taken from the command, actual work can begin. Most of the functions implemented in this project follow the same process, wherein the relevant information is found in the table (in the form of list indexes) and modified accordingly.
Inserting a tuple is the simplest of all the functionality in this project, as once the command is parsed, the data is formatted and appended to the end of the table text file.
Deleting a tuple requires a loop to iterate through each line in the file, wherein the program searches for the named criteria, and, if a match is found, removes all data from that index in the temporary list. This is then written back to a now empty file (data cannot be modified in the middle of the file, so a copy must be made, modified, and written back in).
Modifying a tuple is similar to deletion, except instead of setting the list index to None, the relevant data is found and updated.
Querying specific data, as mentioned earlier, requires extra work due to the uncertainty of commands. Once all relevant indexes are found, searching through the list for the right fields can be done and a temporary list is created that stores each tuple as a string element in the list, which is printed.

Other unmentioned functionality is the irrelevance in which the column titles are ordered in the select command. If a table is set up as *pid* | *name* | *price* and the user selects *name, pid*, the output will still retain the correct table ordering.
Error handling for selection of columns not in the table has not been implemented, but an existence check similar to those in dbutils could remedy this in the future.

**Execution Commands**

In a Unix environment, run the command

```
python3 main.py
```

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

No arguments are used with the file, as standard input is used.

Ensure that all the modules are in the same folder (dbutils, tableutils, and queryutils). Future revisions may place them into their own folder if necessary.

The included PA2_test.sql file contains the same commands but is modified in the following ways:

- Removed multi-line commands to make parsing simpler.
- Fixed case-sensitivity issues with “Project”. They are now all uppercase P. SQLite3 (and Ubuntu) is case-sensitive, so my program is as well.