Programming Assignment 4

CS 457

Adam Lychuk

Implementation

Transactions and Locking

For transactions a couple of new variables are added to the data manager. A bool on whether a transaction has been started, a list of modified tables including their data, a integer describing if the transaction has been successful or not, and a list of table names the current data_manager can access.

Beginning the transaction simply sets the transaction variable to true signifying that the data manager has begun a transaction. Locks are implemented using add_lock and remove_lock which both add/remove a file with the specified table's name concated with _lock. The update function has been modified at the start. If a transaction has started it uses the function __check_for_lock to search the current directory for a lock. If it sees a locked file not on its access_list it aborts the transaction. Otherwise it makes the modification and appends the modified table to a list of changed tables. When commit is called those tables are written to memory instead of being written on the update function call. In addition after the commit the lock on the file is removed. Select is already read only so no locks are used on it.

Compilation

To compile and run the source code simply type

python3 driver.py <sql file name>

Or

python3 driver.py

For standard input. When using the standard input press enter after each individual command or series of commands delimited by semicolon. It also works for injecting a file by typing

python3 driver.py <PA3_test.sql

For this specific project the standard input is best as it allows changes to be made incrementally in each terminal. For testing purposes I copied and pasted sets and individual commands in, but the other options should also still be working.