Nathaniel Daniel

CS 457

Design Document

Overview

This project implements SQL transactions while building on top of the past projects. A demo is available in the top level of the project as “demo4.mp4” in the case of build issues. The compile times were trimmed from the previous project by removing the “testlib” dependency, which only tested the parser from PA1 and was not extended for future parts.

Building and Executing

This project uses a standard cmake build system. It may be built by first creating a directory called “build” in the source directory. After entering the directory, running “cmake ..” and “make” will build a project. The executable’s name is “BasicSqlCli”. The cli does not support loading commands from a file. Commands must be fed from the standard input.

Functionality and Implementation

This project implements transactions on top of PA3. I started by adding the begin and commit statements to the parser. Afterwards, I designed the table locks. A table lock in my project is an empty file with the same name as a table, but with a “.lock” extension. The presence of this file signifies that a table is locked. While in a transaction, the program will now check for locks while running statements. If a lock does not exist, it acquires one and performs its action. Only update statements are limited by transactions so far, as that was all that was needed to pass the test script.

After adding the table locks logic, I added “buffering” for the update statement execution while in a transaction. Instead of directly committing changes, it now buffers changes until the user commits. This is bug-prone in the presence of inserts, and buffered changes will also not be reflected in select statements. However, this naive implementation is competent enough to pass the test script.