Lab Report 4: Rollback and Recovery

Quick Summary of Lab 4: Lab 4 focused on the log based rollbacks and recovery. It changes the buffer management policy to a NOFORCE. This was because then the dirty pages wouldn't be forced to flush to the disk after a commit. This allowed SimpleDB to avoid unused disk access and make it more efficient. Now in order to allow for rollback and recovery, the lab stated to implement logging in BufferPool in order to have all the proper entries.

Implementation:
Undo: For Undo, I iterated over the logs that were set previously. This updates the pages to a prior updated state. Also for undo, this was necessary by both rollback and the recovery methods. This allowed multiple transactions to undo in recovery and also single for rollback.
Redo: Redo pretty much iterated through the logs and updated their statuses.
BufferPool: I implemented the buffer pool to a NOFORCE policy. I also updated evictPage() by removing the check for dirty pages before a given flush. I also updated transactionComplete() which added logging to it.
Rollback: For Rollback, I set the relevant tids into the log and compiled the changes by undoing their current states.
Recovery: This method went back to the commit states of the database. I used redo to redo the committed pages and undo the uncommitted pages.

I found the recovery method rather difficult mainly because I had to handle a lot of the cases presented I ended up using cases for each case: Begin_Record, Checkpoint_Record, Commit_Record, Update_Record, and Abort_Record. I also found the BufferPool change to a NOFORCE policy to be rather difficult to implement. I had to adjust some of my implementations because it wasn't working for the log file-based transactions portion of the lab. This added some extra time for me to figure it out.

Changes made outside of LogFile.java: I made no changes outside of the BufferPool and LogFile Class.
Unit Test: One unit test that I would include to improve the testing suite would be to have different tests that handle each type of entry in the Recovery method. By adding this test or these tests, students would be able to check if they were implementing the right entry correctly. This would be to check to begin, check, commit, update, and abort.

Additional Feedback: None