Writeup Lab5

Maintaining state about locks

I use a class `PageLock` to manage locks for different objects (use page granularity lock as the README suggests). It contains the set of locks (Shared Locks and Exclusive Lock) which the page holds, and can acquire/release lock for a specific transaction.

As for `releasePage()` and `holdsLock()` in exe 1, just do the `releaseLock` and `isHolding` method that are already implemented in `PageLock` class, note the operations must be done in `synchronized` block.

Check for deadlock

I use a class `DependencyGraph` to maintain the current Dependency Graph for transactions. When I need to retrieve a specified page, I try to get the lock with needed permission first. If the acquire operation doesn't success, add the `(tid, pid)` to the `DG` and judge if there is a deadlock, if so, just abort the transaction and throw an exception.

Page eviction

I change the eviction logic ——— dirty pages now cannot be evicted because we use NO STEAL policy.

Transaction Complete

If a transaction completes (commit or abort), first remove it from the Xact table (`tid2Pid`). Then for the locks that the transaction holds, don't forget to release them. Finally check if the transaction is committed, if committed, flush the dirty pages dirtied by it to disk, if aborted, reset the pages to the BeforeImage.