

1.Experimental procedure:

Exercise 1: Transactions, Locking, and Concurrency Control

I defined a `LockManager` class to encapsulate synchronization methods related to locks, avoiding interference with other methods in the `BufferPool` class.

Exercise 2: Lock Lifetime

The design for acquiring locks is simple, but implementation is tricky. I followed the guidelines to implement locking at the page level. An important point to consider is ensuring synchronization blocks when flushing dirty pages to disk.

Exercise 3: Implementing NO STEAL

Transactions only write modifications to disk upon commit. This means that transactions can be aborted by discarding dirty pages and re-reading them from the disk. Therefore, we cannot evict dirty pages. If all pages are dirty, a `DbException` is raised. Thus, only slight modifications are needed for the eviction strategy regarding writes.

Exercise 4: Transactions

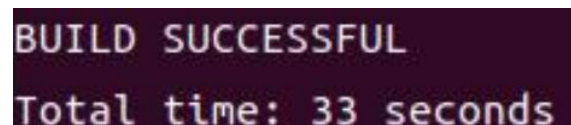
At the start of each query, a `TransactionId` object is created. This object is passed to every operator involved in the query. After the query is completed, the `BufferPool` method `transactionComplete()` is called. If the transaction successfully completes, all dirty pages in the disk must be flushed to disk; if the transaction fails, then a rollback is necessary: reversing the changes from the disk back to the BufferPool. Finally, all locks held by the transaction are released.

Exercise 5: Deadlocks and Aborts

A deadlock is a situation where multiple concurrent processes are waiting on each other due to competition for system resources. Since implementing timeout waits is relatively simple and suitable for this lab, we adopted timeout waits: if a resource is not acquired within a certain period, the process automatically relinquishes any resources it holds.

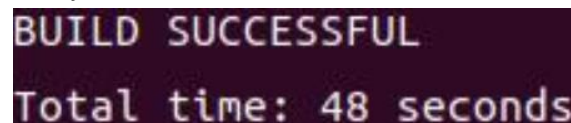
2.Run Result:

ant test:



```
BUILD SUCCESSFUL
Total time: 33 seconds
```

ant systemtest:



```
BUILD SUCCESSFUL
Total time: 48 seconds
```

3.Time consumed:

This lab is related to lock. Because we had taken the Operating Systems course, it is easy for us to understand. However, it is difficult to implement codes. It took me 4 days totally to complete this lab.