CSEP 545: Milestone Report 1

Team members:

Mihnea Olteanu ([mihneao@gmail.com](mailto:mihneao@gmail.com))

Vincent T.C. Lung ([Vincent.lung@gmail.com](mailto:Vincent.lung@gmail.com))

# Summary

We are working on steps 1 to 3 of the project.

For step 1 and 2, atomicity and durability, we are implementing the shadowing with page level locking. We are currently planning to use a page size of 4KB for our data file with that assumption that that will allow us to write each page atomically to disk.

For step 3, we modified the LockManager to support lock conversion and we created a dozen or so unit test cases to verify its correctness. One subtle but interesting thing that we noticed is the power of two-phase locking in lock conversion. In one of our unit tests, we created 2 threads that compete for a resource: Thread 1 acquires a read lock on a resource and thread 2 attempts to acquire a write lock on the same resource. When thread 1 wants to upgrade its read lock to a write lock, our code uses two-phase locking to acquire a write lock for thread 1 before releasing its read lock. Our unit test verifies this will in fact prevent thread 2 from “stealing” the resource from thread 1.

Overall, we know there is still a lot of work to be done but we feel pretty good about our progress.

# Detailed Design

## Durable Storage

The implementation of our durable storage data file has the following components:

* **Page Table:** each data file has one instance of a page table used to translate logical record addresses into physical addresses corresponding to the location of the record in the actual data file
* **Resource Index:** each data file has one instance of the resource index used to store a mapping of resource id (RID or Client) to logical address in the page
* **Free Page List:** each data file has one instance of a free page list used to recycle pages that have been freed up by the shadowing algorithm’s moving of data to new physical location
* **DB Root:** each data file will have a DB root instance stored on page 0 of the data file containing the addresses of the page table, resource index, and free page list

The data file will support variable length record stored on fixed sized pages. This allows us the flexibility to store all types of resource records and the data file meta data information in one physical disk file.

**Design Limitations/Assumptions:**

* 4kB of data written to a file will be flushed as one atomic disk operation to durable storage. (We reserve the right to bump the page size to 8kB if we run into issues due to the page size being too small.)
* We do not support records whose size is larger than the size of one page (right now that means we do not support records that would take more than 4kB to store on disk)
* We are making the assumption that we will be able to load the page table, resource index, and free page list into memory in their entirety for read/write operations to those structures. This requirement will limit the number of resources our data file can store but should give us adequate capacity for this project at a great saving in code complexity.
* The most granular locking instance we will support is Page level locking. Allowing for record/resource level locking will be out of scope of our implementation.

# Source code directory structure

You can find the source code of our project under the various folders in our submitted zip file. Here is the directory structure of our project (underlined green indicates folders we have worked on thus far):

* CSEP545/ - Main executable for the project.
* CSEP545/UnitTestProject – Contains the unit test cases we have created for the lock manager.
* MyRM/ - Contains the lock manager, resource manager and storage manager.
* MyRMTests/ - Contains unit tests we created for classes in the MyRM folder.
* MyTM/ - Transaction manager
* MyWC/ - Workflow coordinator
* TP/ - Transaction Processor