Writeup for lab1 **Yuwei Wu**

CS392 Database Management System ACM Class, Zhiyuan College, SJTU

Prof. **Feifei Li** Due Date: April 23rd, 2019

Submit Date: April 21, 2019

Design decisions about lab1

- To build the iterator which iterates over all the fields of this tuple, I first wrote an auxiliary class called FieldIterator and the fields() function just needs to return the instance of FieldIterator. So does the TDItemIterator, the PageTupleIterator and the HeapFileItertor.
- To implement the structure of Catalog, I wrote four ConcurrentHashMap: tableId2File, tableId2Name, name2Id and tableId2PField. Using ConcurrentHashMap is to ensure thread safety. And the HashMap name2Id is to handle name conflicts.
- The HeapFileItertor is designed as a itertor that use TransactionId and the heapfile id to access a HeapPageTupleItertor that has already been implemented with specific page number using BufferPool to allow iterate through all the tuples over all pages.
- Finally, the sequential scan access method SeqScan is to use a registered table id and then find the databasefile that carry the information of the table. And iterating through the tuples of the table can simply use the iterator of the databasefile.

API changes

- Add exception handle codes for the constructor for HeapPage as the line "this.td = Database.getCatalog().getTupleDesc(id.getTableId());" may throw exception for invalid tableId.

Missing or incomplete elements

- I haven't done the part of the eviction policy which means current code will confront problems when using large size of data which has larger size than the size of Bufferpool as input and the iterator cannot fetch data out of the size of Bufferpool.
- Just do a simple exception throw when lock to a page is held by another transaction. And further work can be done here to handle conflicts like this.

Time spent and difficulties

- I spent three whole days working on lab1.
- I found the part that needs to judge the whether the heapfile slot is empty is quite difficult. First, java use big endian to store the header and I spent quite a lot time to figure it out. Second, I am not quite familiar with the bitwise operations and that costs some time.