Conor Reisman

4/12/2013

CSE444

Lab1

## Design Decisions

Tuple and TupleDesc were both implemented as lists. This makes the class work in O(1) time for every method other than fieldNameToIndex (not including the ones that must be O(N)). I could make it O(1) with a  map, but it's not worth it.

Catalog is a pair of maps, one from names to ids and one from ids to relevant data (DbFile, primary key, and name). This is O(1) for all queries.

BufferPool is a map from PageId to Page to make that lookup O(1).

HeapPage has a map from Tuple to its index in the Tuple array. It also has a list of empty slots for inserting new Tuple. This makes lookup and insert O(1). These aren't really useful for lab1 stuff, but I accidentally did insert and delete, so they were helpful there.

HeapFile has a static counter to assign unique ids. These are cached by file name in a static map so HeapFiles with the same name won't get different ids. As such, I didn't follow the recommended bad implementation of hashing.

My HeapPage and HeapFile iterator classes have similar implementations. They are greedy in that they immediately find the next tuple after returning the last one. This is done under the assumption that if an iterator is being used they will probably scan all of the items and it better fits the idea that hasNext is not a mutator method.

## API Changes

Most API changes were very minor, such as throwing IllegalArgumentExceptions on invalid inputs. I also changed Catalog.addTable(DbFile file, String name) to pass null to the other Catalog.addTable method since the empty string is a valid field name and could accidentally create a primary key. Other changes made were adding implements Iterator/Iterable to different classes (Tuple, TupleDesc, and HeapPage). The most important change was to make the Page interface implement Iterable. This allows me to not have to cast Page to HeapPage in order to get an iterator in HeapFile. It also makes sense since all pages should have an iterator method that returns an Iterator<Tuple>.

## Incomplete/Missing Implementation

Nothing that was required for lab 1 was left unimplemented.

## Other

I spent around 16 hours implementing and reviewing this assignment.

A full list of my complaints about this assignment can be found in notes.txt.

My major complains are:

- The wrong exceptions are thrown constantly.
- The proposed implementation for HeapFile ids doesn't guarantee uniqueness. After it states EXPLICITLY that it MUST be unique.
- The Page class should have an iterator method so I don't have to cast.
- PageId shouldn't be an interface. It's basic; there's a table id and a page number! Because HeapFile requires a HeapPageId, but other classes store a PageId, I now have to create a new HeapPageId or cast the old one in order to use it.