CS 143: Summary of lab 1

Zhehao Wang 404380075

Jan 2015

1 Design decisions

Most of decisions in lab 1 are implementation decisions: mainly choosing data structures for members of certain classes.

For example, in *Catalog* we introduced the class *Table* to hold the entries of *tableId*, *tableName*, *dbFile*, and *pKeyField*. *Catalog* needs to store a series of tables, and for that we chose *ConcurrentHashMap<int, Table>* which maps *tableId* to the *Table* entry. This is because *tableId* should be unique, and most of the access methods refer to values in the entry by the *tableId*, *HashMap* lookup would be faster than linear look up, and concurrent access could be a future concern. One method *getTableId* gets the ID from *tableName*, for this kind of access we created another mapping from table names to IDs, since one table name cannot map to two table IDs.

Similar decision exists for choosing the data structure to hold *TDItems* in *TupleDesc*; for that we choose an *ArrayList* since its access pattern is most like a plain list. Here we don't go into detail about other similar choices.

Another minor decision is whether *HeapFile.readPage* should be throw *IOException* and *FileNotFoundException* as it reads from a RandomAccessFile; we chose not to throw these errors and instead handle them within *readPage* method, becaus *readPage* is able to handle them, and throwing them would require much change to the interface.

A final minor decision is the implementation of *HeapFileIterator* class which implements *DbFileIterator* interface. Such an implementation is needed, as we cannot instantiate an interface when accessing *HeapFile.iterator()*, so instead we implement the interface, and hook it up with *BufferPool.getPage()* to provide the functions it needs.

2 Changes you made to the API

Changes to the API were minimal in this project, with the only ones being adding the NoSuchElementException to these function signatures. (Java uses a type system enhanced with explicit exception declaration, so adding types of Exception counts as change to the API)

```
String Catalog.getPrimaryKey(int tableid)
String Catalog.getTableName(int tableid)
```

As in the handin, these become

```
String Catalog.getPrimaryKey(int tableid) throws NoSuchElementException
String Catalog.getTableName(int tableid) throws NoSuchElementException
```

Reason being that other table attribute getters which take tableId throw such an exception if a table with matching ID does not exist, and similar behavior could be expected from these two.

There are a few behavior changes to the code, which seems rather trivial for this project. (For example, Catalog.toString() prints "Name(Type)" instead of "Type(Name)" as given in spec, since the former seems more conventional); Also a few trivial behavior definitions. (For example, SeqScan.rewind() calls open() automatically after rewinding, as the spec didn't define whether rewind() should provide an opened iterator.)

3 Missing elements

Other than the functions mentioned as "not needed for lab 1", the implementation does not have the full *BufferPool.getPage()* implementation (for a start, missing eviction call even though eviction is not implemented). Implementing these without an actual eviction implementation would not change the behavior in lab 1's tests, so these functions are not fully implemented.

4 Summary

The project took about 24 hours in total. Getting started with simpledb, learning what's implemented and what's needed, as well as the test framework took a considerable amount of time. The actual implementation is rather clear once the relationship between each module, and the utilities of each class are figured out. Testing took a large amount of time, since issues such as NullPointerException and ClassCastException handling was not done correctly in the initial implementation.

One interesting note while testing: the author created his own table file, and tried the sample scan code (src folder IteratorQueryTest.java) on the custom dat file, however the final line is always missing. The problem is resolved using hexdump on the dat file, and finding that the last bit in page header is not set as expected. The problem ended up being the source txt file (copied directly from the example one in html) has one white space before a line break and after the last integer, which causes the last integer and the line it belongs to get omitted during conversion.