What Labs About and Some Design Descisions

For this assignment we are trying to build a simple database and I learned a lot about about the structure that database should have.

First of all I implemented the Tuple and TupleDesc. The TupleDesc specify the schema of tuples, it contains information of type and name of each field. A Tuple is a single row of the tale that contains value of each field follow the format of TupleDesc. I made the TupleDesc a HashMap with id of field as key and TDItem (represent of a field schema including the type and name of field) as value. I also make tuples a HashMap. HashMap's put and get method are O(1) , which makes searching and adding field schema and field value super fast in case we had a large number of attributes of table.

Second, I implemented Catalog, which is a menu of table. It contains id, name, publicKey, file that stores contents of the table, and table schema (TupleDesc). I made three HashMaps. First two are a map with name as key and table id as value and a map with id as key and name as value. Because each table will have its unique tableId and unique name, we could create this two maps to represent bi-directional idea and to speed up search for name using id and search id using name. The other map's key is tableId and value is a three element array stores primary key, schema, and file for the table. There are method required to fetch these three information by table id and thus HashMap would make fetch easier.

Third, I implemented BufferPool. The class caches pages HeapFile fetch from the disk. BufferPool is also a Hash Map (concurrent for later labs) to speed up fetch of pages in the buffer by PageID. When getPage() is called, bufferPool will first look for the page in it's buffer. If page exist then return the page thus we don't have to make redundent fetches from disk. If not we will fetch the whole page needed from disk by calling HeapFile's readPage().

Fourth, we implemented HeapPage and HeapFile. HeapPage represents the instance of pages stored in HeapFile that is fetched by HeapFile from disk and cached in BufferPool. HeapPage contains slots of tuples and has page id which is a combination of table id and page number. It has an byte array header (where each bit of the byte means if the corresponding slot is valid or not) and a tuple array stores tuples. There isn't a lot freedom in design data structure for both HeapPage and HeapFile but I used bit mask to quickly get bit in the header to determin if slot is valid or not. The HeapFile fetch pages from disk and provies tuple iterator for BufferPool cached pages. I initialize iterator in the constructor and store iterator as a public field rather than create it

when return it in iterator() method because the open() will change its and we would like to store its change and only reflect it during method call. I calculated total number of tuples and if the tuple loop count has reached this number then there would be no next tuple (hasNext return false). The next() method will fetch page from buffer pool by PageID. For readingPage() method of HeapFile, it fetches page from disk using RandomAccessFile method and read byte by byte. I calculated the offset of reading on different pages and it is the page number that need to read times the size of a page specified by BufferPool.

Last, I implemented SeqScan, it is an operator and an iterator that scans tuples from Heapfile. It's open() method also opens up the iterator of Heapfile. It's hasNext() method calls heapfile iterator's hasNext() to see if there is any more tuples need to iterate. It's next() method calls Heapfile's next() method fetch the next tuple in the HeapFile.

One Unit Test To Add

The testIteratorBasic of HeapFileReadTest() only consist testing iteration functionality of a single page heapfile. The testIteratorClose() method has using two paged heapfile but it is not testing open() and then next() routine. I would have more paged heapfile for the testIteratorBasic to test iterator over tuples on different page. Also, I would test iterator to fetch tuples over invalid slot.

Other Declarations

I didn't change any of API. I didn't miss any implementation other than methods with "not necessary for lab1". I don' t have any feedback. I didn't collaborate with anyone.