PA1 Write-up

Team members: Wei-Hsiang Lin and Alex Wong

1. Describe any design decisions you made. These may be minimal for pa1.

1.1 Tuple, TupleDesc

- In TupleDesc and Tuple, we used ArrayList and ArrayList to contain fieldTypes, fieldNames and tupleFields because it can grow as needed (not static in length).
- We also had to write TupleDesc's equal method to allow for it to make comparisons.

1.2 Catalog

- Here, we used a HashMap to store a mapping between file id and another class called TableInfo, which contains information about a table, such as table name, primary key field and dbFile.
- Sometimes, we have a table name and want to find the table id. Instead of looking it up, we decided to store this information in a HashMap in order to provide constant time access.

1.3 HeapFile, HeapPage, HeapPageId

- HeapFile is an implementation of DbFile that stores a collection of pages, which stores a collection of tuples.
    - It stores a File class, which is used to access a particular page's data.
    - The iterator method also returns a DbFileIterator to iterate over all the pages in a given HeapFile.
- HeapPage stores data for one page of HeapFiles.
    - Using a byte array, we use 1 bit per tuple to store information about whether that tuple (in the page) is valid or invalid (been deleted or never initialized). When checking this information, we use bit-wise manipulation to optimize performance
    - Since it contains data for tuples, it contains a Tuples and TuplesDesc array as well.
    - Lastly, a HeapPage's unique identifier contains table id and page number. This is represented by a HeapPageId class.

1.4 BufferPool

- The purpose of a BufferPool is to cache pages in memory that have been recently read from disk.
    - Here, we maintain a HashMap of PageId and Page to provide constant time access if it is stored in the BufferPool via the page id. However, if it is not, we need to retrieve the HeapFile for that particular table and perform readPage on the HeapFile to retrieve the page.

1.5 SeqScan

- In short, given a table id, this class utilizes the Catalog class to access the HeapFile associated to the table id. Then, it gets the iterator from the HeapFile. Following, most of the operations in this class uses the iterator.

2. Discuss and justify any changes you made to the API.

None

3. Describe any missing or incomplete elements of your code.

We completed all parts required to perform both **ant test** and **ant systemtest**.

4. Describe how long you spent on the assignment, and whether there was anything you found particularly difficult or confusing.

- As a team, we spent around 2 days to complete.
- The most challenging task was getting lost in the details. We were able to figure it out after we drew it out visually.