Report of Database Lab2
Name: Nayun Xu
Student ID: 515030910635

1.Design decisions

1.1 Eviction policy

I use LRU to manage page eviction. In the code, I use a ***LinkedList*** to store ***pageid***. And this list represents the order of the latest time that the corresponding page is visited. Each time a page is visited, I first remove its ***pageid*** from the list(if it exists), and then add it to the end of the list. When doing eviction, I scan the list from its beginning to find the first page that isn't dirty. If a clean page exists, then evict it. Otherwise, I choose the first one in the list, flush it to disk, and then evict it.

1.2 BTree insert

First locate the leaf page that the tuple should be placed. And try to insert the tuple into this leaf page. If it is already full, then split it, copy up an entry to its parent, and deal with the parent recursively. The ***insertTuple*** method in ***BTreefile*** is provided, so I just implement the split and copy up method.

1.3 BTree delete

First locate the leaf page that the tuple to be deleted is stored. And check the number of entries in this page. If after deletion, the page will be less than half full, then try to redistribute it by stealing from siblings. If stealing fails, merge it with siblings. And merging needs to delete the corresponding entry in the parent page, so deal with the parent recursively. The ***deleteTuple*** and ***handleMinOccupancyPage*** method is provided in ***BTreefile***, so I just implement the steal and merge method.

2.Changes on API

I add a Linkedlist in ***BufferPool*** to implement LRU.

3.Incomplete elements

I still don't know how to implement the lock.

4.Experience

I spent about 15 to 20 hours(not exact numbers) on this lab. The beginning was the hardest, because I had to go through the methods that were given already. And at first, there were some mistakes in my ***BufferPool***, and it still passed the corresponding test. So it resulted in the content in memory different from the disk(Modification not written to disk). And to locate this problem, I went through the ***BTreefile*** and ***BufferPool*** to understand the process that a page is written back to the disk. After understanding all the given methods, I found the work bacame easier. The debug was not easy, because the format of the test. I spent some time understanding the test files to help debugging.