**Writeup for lab2** **Yuwei Wu**

**CS392 Database Management System** ACM Class, Zhiyuan College, SJTU

Prof. **Feifei Li** Due Date: May 56h, 2019

Submit Date: April 30, 2019

**Design decisions about lab2**

- To implement the evictPage(), I just simply choose a random evict policy. Each time, I will record a list of clean pages and randomly flash one of these pages. I think only evicting clean pages will save time in reading and writing from disk.
- To implement the insertion part, it's about spliting leaf page and spliting internal page. For the spliting leaf page part, I just create a new leaf page and move first half of the tuples in the original leaf page to this new leaf page and then update the related sibling pointers and then update the children pointer of the parent. For the spliting internal page part, things goes almost same as the spliting leaf page part. The only difference is that the two split internal pages need to update their children pointers. The returned page is returned according to the comparison between field of middle key entry and the inserted field.
- To implement the deletion part, it's about stealing and merging. In stealing from leaf page, fist I move enough number of tuples to the needed page from the other page and update the entry of the parent of two leaf pages. And in stealing from internal page, first move the parent entry down to the needed page and then move enough entries to the needed page from the sibling internal page. After doing this, just update the parent entry using middle key filed and update the child pointers of parent page and the two redistributed internal pages.
- For the merging part, the merging leaf pages part is just to move tuples from the sibling page to the desired leaf page and update the sibling pointers of these leaf pages. The merging internal pages part is to first move the parent entry down to the desired page and then move all the entries of the sibling internal page to the desired page. After finishing this, just update children pointers and delete parent entry and parent page.

**API changes**

- No API changes.

**Missing or incomplete elements**

- I didn't pay much attention to the part of dirty pages. In my opinion, after inserting or deleting tuples in dbFile, the pages in returned page list are all dirty and I simply mark all this pages dirty and update my pageMape in bufferpool. I am not sure whether problems will occur in the following labs.
- Also I keep an pageId2transactionId map in bufferpool to record transaction id and prevent possible conflicts with more than one transaction request but the way I process this is too simple and I didn't take lock and different permissions into consideration.

**Time spent and difficulties**

- I spent two whole days working on lab2.
- I found the part that needs to implement splitLeafPage difficult as I forgot to modify sibling pointer of the sibling's neighbor, which cost a lot of time debugging. In deed, I found it difficult to debug through the BTreeFile as it is impossible to print the whole tree for large data size and if the test data size is small, the problem may not occur.