**CMPUT 291**

**Project 2**

Kathleen Baker

Steven Cherfan

Brandon Smolley

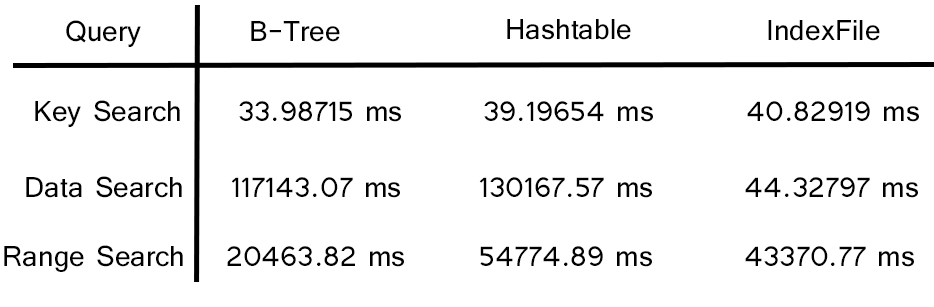
**Implementation**

The first step to using our program is to set the user permission, this can be achieved by running “chmod u + x mydbtest” from the terminal. Upon running the permission command the program can be run by entering “./mydbtest x” where ‘x’ is either ‘btree,’ ‘hash,’ or ‘indexfile.’ The btree option creates a B-Tree file structure, the hash option creates a hash table file structure and the indexfile generates our improved file structure.

The IndexFile is a combination of two separate database files that have the same data. They both use the B-Tree structure as opposed to the hash table structure due to the performance gains that B-Tree offer when doing key searches, as opposed to hash tables which have a much slower key search. The first file constructs the database normally, generating all keys and their associated data values. The second file is generated at the same time, but reverses the key and the data values. This allows the data values to act as keys when searching the database, which allows us to significantly decrease the time required to search for data in the database.

**Experimental Results**

Data



The above graph showcases our results after running each test four times and recording the time taken to complete each test in micro seconds. The averages of these times were then calculated and placed in this table to illustrate the difference in speed between the database structures.

**Analysis**

From the experimental results we can analyze the effect of the structure on key and data retrieval time. The B-Tree, hash table and Index file structures all show similar time taken to find a key in their database, with the Index file being the slowest of the three by a small amount. Due to the nature of the searches the difference in key retrieval time doesn’t appear to be significantly different from the B-Tree and hash table formats, due to the different keys that were searched for.

The data search shows slow retrieval times in both the B-tree and hash tables formats, this is most likely due to the linear search that is done in order to find the data as they cannot jump to keys in order to get the needed value. The data search is where the advantage of the index file structure is seen, as it is essentially a copy of the original database but with reversed key and data values. This allows any search done using index files to have retrieval times similar to that of a key search.

The range search for all of the structures showed results consistent with the previous tests. All of the range searches were done using keys they should all have similar retrieval times. Since our index file uses the B-Tree structure it has a similar time to finish that the of the original B-Tree, any differences would be a result of the data used to do the range search and the number of records that needed to be found to complete the search. The hash table had the slowest time to finish out of the three, however due to the keys used to perform the search any significant differences may be attributed to the number of records that were searched in comparison to the B-tree and Index file range searches.