There are seven charts included in this submission. IndexedSelectTimes.png,Non-indexedSelectTimes.png, and RangeQueryTimes.png are fairly self explanatory: they chart the average performance time of our implementation of the select operation both when using an index and not using an index to perform the select. The selects were performed 100 times for each data structure as well as they were performed on tables with a range of sizes from 100 records to 5000 records.

For the join operations things are decidedly more complicated. JoinTimes5000A plots the average time to join two tables for both EquiJoin and NaturalJoins using various different index data structures. Due to the poor performance of our LinHashMap implementation, however, this data is mostly unillustrative. Thus, the JoinTimes5000B chart plots the same data but excludes the performance results using the LinHashMap index data structure. The performance of our implementation using the BpTreeMap vs. the Oracle provided TreeMap data structures is more obvious in this chart, and it is clear that the average join time of our implementation benefits from using our implementation of the BpTreeMap vs the stock TreeMap data structure. There are two more charts plotting join times as well: JoinTimes2000A and JoinTimes2000B. These plot the average performance time of our implementation of the NaturalJoin and EquiJoin operations for tables with size from 100 records to 2000 records and using all three data structures as well as a nested loop implementation for the join operation. You can clearly see in JoinTimes2000A that the average time performance of the nested loop implementation is exponential in the size of the tables. Compared to a near constant asymptotic performance for the data structure implementations. The latter of the two charts excludes the nested loop performance in order to more clearly compare the performance for join operations when separately utilizing one of the three index data structures.

The work for this project was divided as follows:

Teresa and Jason fixed errors from the BpTreeMap implementation from Project2. Sharon and Trent fixed errors from the LinHashMap implementation from Project2. I integrated Table.java into the TestTupleGenerator class, ran the simulations to measure performance time, and charted the performance through the use of Excel spreadsheet tools. Thanks!