1. This question is worth 60 points. Please find an in-memory implementation of red-black trees and hash structures. You will fill them each with data that we will provide. Then do timing experiments for the operations we will provide (i for insert, d for delete). Then show us your timings as well as code. For 20 points extra credit, find the minimum and maximum depth of your tree. For more information, please see http://www.cs.nyu.edu/lichtman/db-spring06/homework.html#hw3

2. This question is worth 20 points. Show the script in your database system of choice. Recall the schema from homework 1:

   Employee(Name, Salary, Manager, Department)
   Course(Student, Subj, Prof, Grade)

We will give you data to generate a large enough database so that the following queries require at least 5 seconds each on our test computer. Show the timings and the results.

   (a) Find the number of employees who earn at least 5000 more than their managers.
   (b) For those departments in which more than one person takes courses, find the average salary by department.
   (c) Find the average salary earned by people taking Roe’s course.

Now show how to add indexes in such a way that each query requires less than 1 second.

3. This question is worth 20 points. Perform the following experiment using a file of 10,000 rows we will provide.

   1. Establish the above database without indexes. Perform 10,000 inserts into the employee table Record the timing.
   2. Reestablish the above database but this time with indexes. Perform 10,000 inserts into the employee table Record the timings.

Discuss any conclusions you might draw.

4. This question is worth 30 points. Please populate the following tables from the TPC-h schema: part, supplier, partsupp, nation, region. The schema is on page 12 of http://www.tpc.org/tpch/spec/tpch2.3.0.pdf. Data can be generated by the TPC’s DBGEN tool you’ll find in the same site. Now consider the query:

   insert result
   select part.name, supplier.name, nation.name, region.name
   from part, supplier, nation, region, partsupp
   where part.partkey = partsupp.partkey
   and supplier.suppkey = partsupp.suppkey
and supplier.nationkey = nation.nationkey
and nation.regionkey = region.regionkey

a. (10 points) Find the join order that would generate the least number of intermediate rows. We don’t care about indices here, just intermediate result sizes. Show the intermediate row counts by running count queries.

b. (10 points) Did you need to try all the combinations of joins to find the order given in a? (Hint: we hope not.) If not, which combinations of joins did you try and why.

c. (10 points) Introduce a predicate (an extra equality selection on a single table in the where clause) to the full query join that changes the join order of item a. Explain the change using intermediate row counts.