CSC 370 – Spring 2016 – Assignment 4
Due date: March 31, 2016, 11:55pm
(e-submit [preferred, scans accepted], or slip it under my door, ECS 556)

Ex. 1 (7 pts)

Consider a disk with average seek time of 10 ms, average rotational latency of 5 ms, and a transfer time of 1 ms for a 4KB block. The cost of reading/writing a block is the sum of these values (i.e. 16 ms). We are asked to sort a large relation consisting of 10,000,000 blocks of 4KB each. For this, we use a computer on which the main memory available for buffering is 320 blocks (*somewhat small memory*). We begin as usual by creating sorted runs of 320 blocks each in phase 1. Then, we do 319-way merges. Determine the number of phases needed, and evaluate the cost of the Multi Phase Multiway Merge Sort.

Ex. 2 (7 pts)

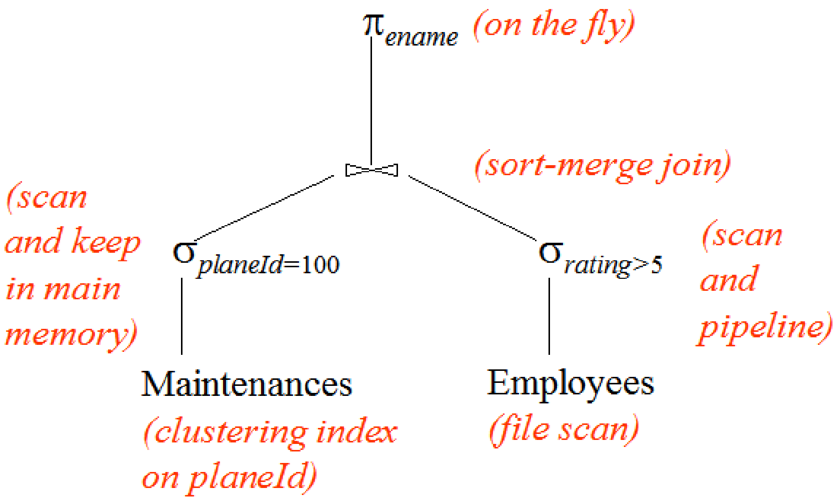
Build a B+ tree index with n=3 using the following sequence of keys:

2, 15, 31, 32, 6, 18, 19, 20, 3, 4, 5, 40, 41, 42

Redraw the tree each time an insertion is done.

Ex. 3 (5 pts)

Consider the following query plan.



What is the cost in terms of number of I/Os for this plan?
Notes. The result of the left selection, being small, is kept in main memory, where it is sorted. The result of the right selection is pipelined to the join operator, i.e. the generation of the sorted sublists for the first phase of sort is done on the fly.
Do not count the I/Os for writing the final results (after projection).
Consult *queryeval.pdf* for the table statistics.

Ex. 4 (12 pts)

For each of the schedules of transactions T1, T2, and T3 below:

1. r1(A); r2(B); r3(C); r1(B); r2(C); r3(D); w1(A); w2(B); w3(C);
2. r1(A); r2(B); r3(C); r1(B); r2(C); r3(A); w1(A); w2(B); w3(C);

do each of the following:

i. Insert shared and exclusive locks, and insert unlock actions. Place a shared lock immediately in front of each read action that is not followed by a write action of the same element by the same transaction. Place an exclusive lock in front of every other read or write action. Place the necessary unlocks at the end of every transaction.

Tell what happens when each schedule is run by a scheduler that supports shared and exclusive locks.

ii) Insert shared and exclusive locks in a way that allows upgrading. Place a shared lock in front of every read, an exclusive lock in front of every write, and place the necessary unlocks at the ends of the transactions.

Tell what happens when each schedule is run by a scheduler that supports shared locks, exclusive locks, and upgrading.

iii) Insert shared, exclusive, and update locks, along with unlock actions. Place a shared lock in front of every read action that is not going to be upgraded, place an update lock in front of every read action that will be upgraded, and place an exclusive lock in front of every write action. Place unlocks at the ends of transactions, as usual.

Tell what happens when each schedule is run by a scheduler that supports shared, exclusive, and update locks.

Ex 5. (5 pts)
Consider the following sequence of UNDO log records:
<START S>
<S,A,60>
<COMMIT S>
<START T>
<T,A,10>
<START U>
<U,B,20>
<T,C,30>
<START V>
<U,D,40>
<V,F,70>
<COMMIT U>

<T,E,50>
<COMMIT T>
<V,B,80>
<COMMIT V>

Suppose that we begin a nonquiscent checkpoint immediately after one of the following log records has written (in memory):

a) <S,A,60>
b) <T,A,10>
c) <U,B,20>
d) <U,D,40>
e) <T,E,50>

For each, tell:

i) When the <END CKPT> record is written, and
ii) For each possible point at which a crash could occur, how far back in the log we must look to find all the possible incomplete transactions.

Ex 6. (5 pts)
Consider the previous exercise, but interpret the log as a REDO log.

For each (a,b,c,d,f), tell:

i) At what points could the <END CKPT> record be written, and
ii) For each possible point at which a crash could occur, how far back in the log we must look to find all the possible incomplete transactions. Consider both the case that the <END CKPT> record was or was not written prior to the crash.

Ex. 7. (6 pts)
For each of the following relation schemas and sets of FD's:

a) R(A,B,C,D) with FD's AB→C, B→D, CD→A, AD→B.
b) R(A,B,C,D) with FD's A→B, B→C, C→D, D→A.

Indicate the BCNF violations (if any). Decompose the relations, as necessary, into relations that are in BCNF.