# Assignment 4 hints

# Q1

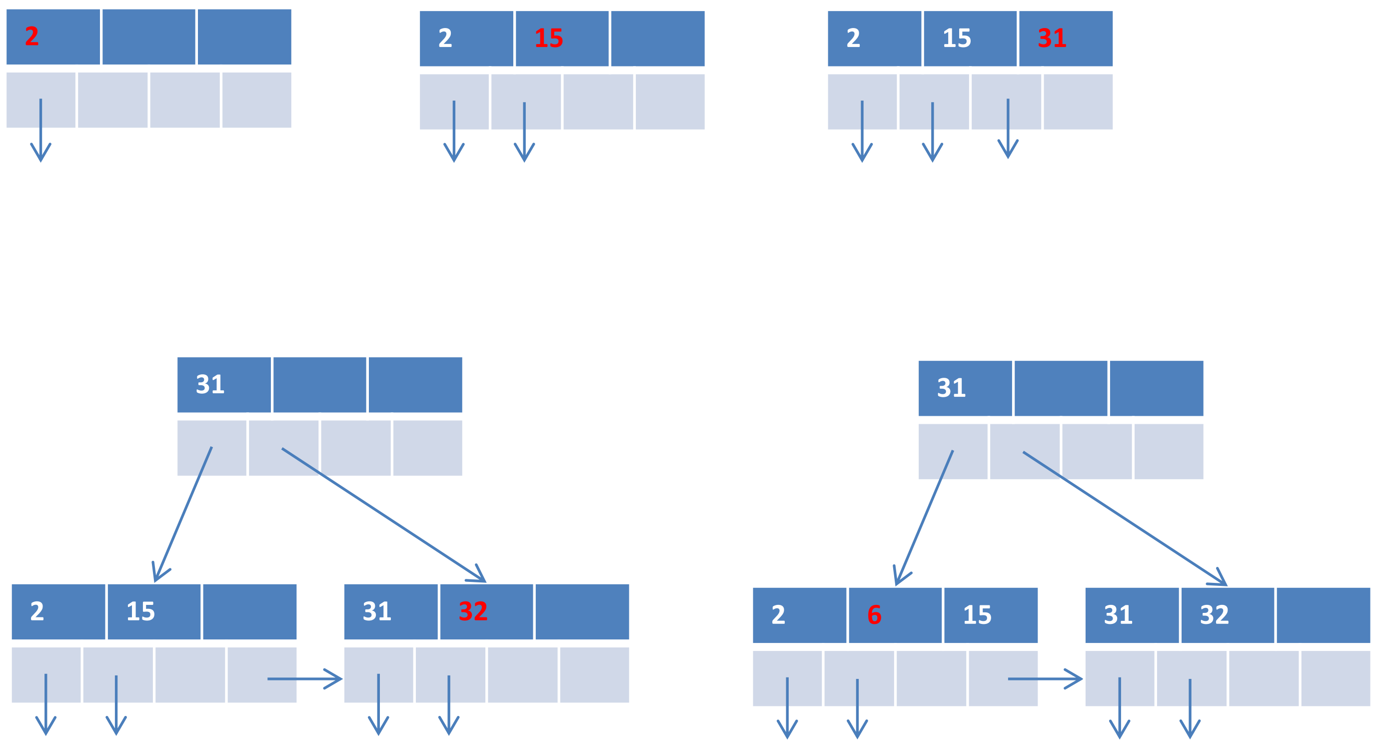
Consider a disk with average seek time of 10 ms, average rotational latency of 5 ms, and a transfer time of 1 ms for a 4KB block. The cost of reading/writing a block is the sum of these values (i.e. 16 ms). We are asked to sort a large relation consisting of 10,000,000 blocks of 4KB each. For this, we use a computer on which the main memory available for buffering is 320 blocks (*a bit 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.

We start by creating sorted sublists. We fill in the main memory (MM) with 320 blocks, sort them in MM and write the sorted sublist to disk (phase one).

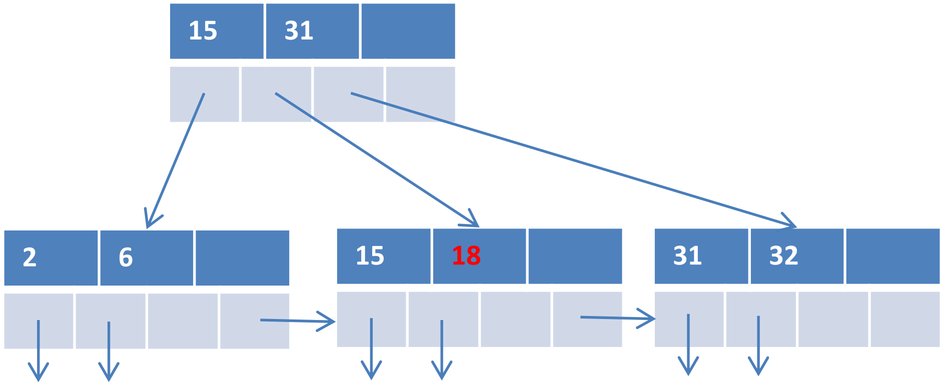
Next we need to do merge. However, the number of sorted sublists is too big to allow a merge with only one pass. Since we need an output buffer, we can only merge 319 sorted sublists at a time. Therefore, we end up again with sorted sublists, but fewer and bigger this time. These sublists need to be merged in turn. Find the number of sublists at each phase. Find the total number of I/Os. Find the time it takes to complete the full sort.

# Q2

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



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



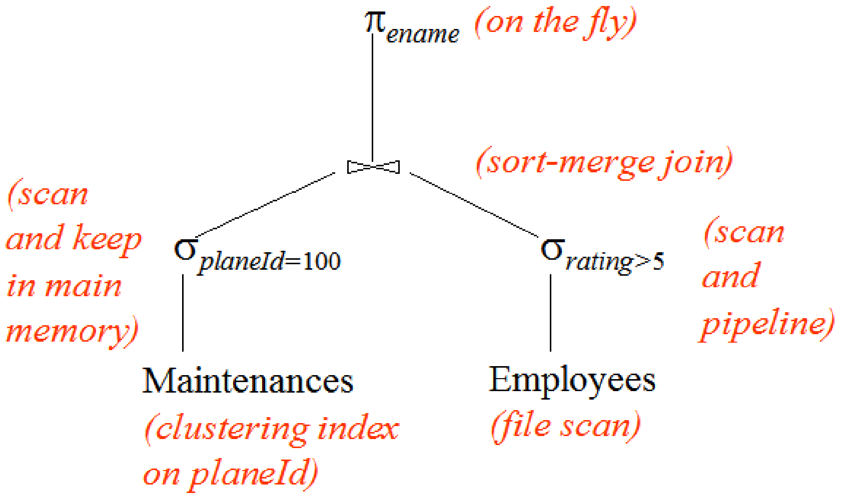

The tree will grow in height when you insert 4.
Be careful: each time, every branch of the tree should have the same height.
You can use the tree template here: copy paste the current tree and make changes/additions. (I am posting both PDF and Powerpoint file)

# Q3

Consider the following query plan.
What is the cost in term 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.

See slides for examples.



# Q4

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.

# Q4, i.1

$r_1(A); r_2(B); r_3(C); r_1(B); r_2(C); r_3(D); w_1(A); w_2(B); w_3(C);$

| $T_1$ | $T_2$ | $T_3$ |
|---|---|---|
| $xl_1(A); r_1(A)$ | | |
| | $xl_2(B); r_2(B)$ | |
| | | $xl_3(C); r_3(C)$ |
| $sl_1(B)$; denied | | |
| | $sl_2(C)$; denied | |
| | | $sl_3(D); r_3(D)$ |
| | | $w_3(C); u_3(C); u_3(D);$ |
| | $sl_2(C); r_2(C)$ | |
| | $w_2(B); u_2(B); u_2(C);$ | |
| $sl_1(B); r_1(B)$ | | |
| $w_1(A); u_1(A); u_1(B)$ | | |

# Q4, ii.1

$r_1(A); r_2(B); r_3(C); r_1(B); r_2(C); r_3(D); w_1(A); w_2(B); w_3(C);$

| $T_1$ | $T_2$ | $T_3$ |
|---|---|---|
| $sl_1(A); r_1(A)$ | | |
| | $sl_2(B); r_2(B)$ | |
| | | $sl_3(C); r_3(C)$ |
| $sl_1(B); r_1(B)$ | | |
| | $sl_2(C); r_2(C)$ | |
| | | $sl_3(D); r_3(D)$ |
| $xl_1(A); w_1(A); u_1(A); u_1(B)$ | | |
| | $xl_2(B); w_2(B); u_2(B); u_2(C);$ | |
| | | $xl_3(C); w_3(C); u_3(C); u_3(D);$ |

# Q5

For each of the following relation schemas and sets of FD's:

R(A,B,C,D) with FD's AB$\rightarrow$C, B$\rightarrow$D, CD$\rightarrow$A, AD$\rightarrow$B.

R(A,B,C,D) with FD's A$\rightarrow$B, B$\rightarrow$C, C$\rightarrow$D, D$\rightarrow$A.

Indicate all the BCNF violations. Decompose the relations, as necessary, into collections of relations that are in BCNF.

Easy in general. See slides for examples.