University of California, Los Angeles
Department of Computer Science

Computer Science 143 **Prof. Ryan Rosario**

Homework 4

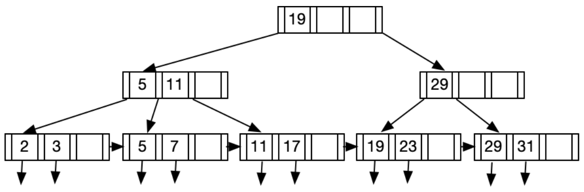
Due Wednesday, May 19, 2021 at 11:59pm ***via CCLE***

Part 1: Indices and B+ Trees

For any exercises that involve drawing, you can draw on paper and scan, take a photo etc., use a diagram program, etc. and submit the file in your homework writeup.

Exercises.

1. Using the B+ Tree below, built on top of a primary index, perform the following inserts. To match our solution, when you split, you should keep the first $\lceil \frac{n}{2} \rceil$ keys of the original node in the original node and move the rest over to the newly allocated node. This matches the book and lecture. (Remember that other resources may do this differently)



   (a) Insert 9, 10, and 8, in that order and show the resulting tree.

   (b) Using the original tree above, draw the path through the tree we would need to follow to find the **record** associated with key-value 11. Be careful! Remember where records are stored. Assume we are searching on a key.

   (c) Using the original tree above, draw the path through the tree we would need to follow to find the **records** associated with keys 7 to 17 inclusive. Be careful! Remember where records are stored. Again, assume we are searching on a key.

   (d) Repeat the previous part assuming the underlying index is a secondary/non-clustering index and we are searching on a key.

   For your studying, you should repeat this problem assuming we are searching on a non-key.

2. Explain why it is better to use a clustering B+ Tree to perform a range query instead of a hash index. You will probably need to use some cost calculations to prove your point.

3. Suppose we have a secondary B+ Tree index, entirely on disk, for relation R and we want to retrieve records with keys in (a, b) exclusive. Suppose that there are keys in this tree that are $\leq a$ and that there are keys in this tree that are $\geq b$. Write an expression for the total worst case execution time spent on disk I/O. Let's say this B+ Tree is stored on a hard disk that has a seek time of 90 microseconds and block transfer time of 2 microseconds. Assume that the seek time already takes the rotational latency induced by a 7200rpm disk into account. What is your time estimate? You *may* need a variable b denoting the number of blocks read, or n the number of records processed. *See the lecture slides for an example.* Assume that keys are unique. **For your studying**, you should repeat this problem with a primary index.

Part 2: Join Algorithms Exercises.

4. Let R and S be two relations and assume neither fits entirely in the memory buffer. R contains three attributes A, B, C and S contains three attributes C, D, E. R contains 20,000 tuples, S has 50,000 tuples. 50 tuples of R fit in a block, and 20 tuples of S fit in a block. Compute an estimate for the total number of block transfers and seeks using the **block nested-join loop** to compute $\bowtie_\theta$ assuming θ is a simple equijoin. Make the same assumption as in lecture that one block from R and one block from S fit into the buffer.

   (a) Consider the case where R is the outer relation.

   (b) Repeat the previous calculation with R as the inner relation.

   For your studying (but not this homework), you should repeat this exercise for each join algorithm we discussed in class, excluding hash join.

5. We can implement accessory joins using some of the algorithms discussed in class. Suppose we want to implement the *antijoin*. $R \triangleright S$ gives us all tuples in R that **do not** have a match in S. Which join algorithm do you believe the optimizer would pick and why? *For this problem, just assume an antijoin is an equijoin.*