Written Assignment #3
CAS CS 460: Introduction to Database Systems
Summer I 2017

Due: Tuesday, 22, 2017, in class.

Problem 1. [20pts]

Consider a disk with sector size of 1024 bytes, 1000 tracks per surface, 50 sectors per track, 4 double-sided platters. Suppose that the disk platters rotate at 7400 rpm (revolutions per minute) and the average seek time is 7 msec.

Suppose that a block (page) of 4096 bytes is chosen. Now, consider a file with 1,000,000 records of 190 bytes each that is to be stored on such a disk and that no record is allowed to span two blocks. Also, no block can span two tracks.

1. How many records fit onto a block?
2. How many blocks are required to store the entire file? If the file is arranged sequentially on the disk, how many cylinders are needed?
3. How many records of 100 bytes each can be stored using this disk?
4. What time is required to read a file containing 100,000 records of 100 bytes each sequentially?
5. What is the time required to read a file containing 100,000 records of 100 bytes each in a random order? To read a record, the block containing the record has to be fetched from disk. Assume that each block request incurs the average seek time and rotational delay.

Problem 2. [20 pts]

Consider a B+-tree of order two (d=2). Thus, the maximum number of pointers per node is 5 and the maximum number of entries is 4.

1. Show the results of entering one by one the keys:
   (21, 12, 13, 45, 23, 67, 89, 79, 95, 11, 10, 14, 17, 33, 46, 28)
   (in that order) to an initially empty B+-tree. Show the state of the tree after every 4 insertions.
2. What is the utilization of the tree? The utilization of the tree is defined as the total number of entries in all the nodes of the tree (both leaf and non-leaf nodes) over the maximum number of entries that the same nodes can store.
3. Can you give a different insertion order of the same values that can create a tree with different height? If yes, provide the order. If not, explain why.

Problem 3. [20 pts]

Suppose that we are using extensible hashing on a file that contains records with the following search-key values:

(2369, 2428, 4750, 6975, 9208, 1821, 4692, 3943, 1620, 7115, 4871, 5659, 1074, 4981, 1453)

Load these records into a file in the given order using extensible hashing. Assume that every block (bucket) can store up to four (4) values.

Show the structure of the directory after every 3 insertions, and the global and local depths. Use the hash function: $h(K) = K \ \ mod \ \ 128$ and then apply the extensible hashing technique. Using this function,

every number is mapped first to a number between 0 and 127 and then we take its binary representation. Then, the extensible hashing technique is applied on the binary representation. Furthermore, initially, you start with a single bucket and a single pointer and the global and local depths are zero (0).

Problem 4. [20 pts]

Suppose that blocks (pages) on the disk can hold either ten (10) records or 100 keys and 101 pointers. So, the order of the B+-tree is d=50. Also assume that the average B+-tree node is 70% full, i.e. it will have 70 keys and 71 pointers. We can use B+-trees as part of several different structures. For each structure described below, determine (i) the total number of blocks needed for a 1,000,000-record file, (ii) the average number of disk I/O's to retrieve a record given its search key, (iii) the average number of disk I/O's to retrieve records for a range query that is matched by 1000 records. You may assume that nothing is in memory initially, and the search key is the primary key for the records:

1. We use Alternative 1, to store the data and the actual records are stored in the leaf nodes of the tree. The B+-tree is a clustered index.
2. We use Alternative 2 for the B+-tree index abd the B+-tree is a primary index. The data file that stores the actual records is organized as a sorted file on the search attribute and each page of the file contains 10 records. This is a clustered index.
3. The same as (2), but the data file consists of records in no particular order (heap file), packed 10 to a page. This is an unclustered index.

Problem 5. [20 pts]

Consider the join $R \bowtie S$ where the join predicate is $R.a = S.b$, given the following metadata about R and S:

- Relation R contains 20,000 tuples and has 10 tuples per block
- Relation S contains 5,000 tuples and has 10 tuples per block
- Attribute b of relation S is the primary key for S, and every tuple in S matches 3 tuples in R
- There exists a unclustered (secondary) index on $R.a$ with height 3
- There exists a clustered (primary) index on $S.b$ with height 2
- The main memory buffer can hold 5 blocks (B=5)

Answer the following questions:

1. If $R \bowtie S$ is evaluated with a block nested loop join, which relation should be the outer relation? Justify your answer. What is the cost of the join in number of I/O's?
2. If $R \bowtie S$ is evaluated with an index nested loop join, what will be the cost of the join in number of I/O's? Show your cost analysis.
3. What is the cost of a plan that evaluates this query using sort-merge join. Show the details of your cost analysis.
4. Evaluate the cost of computing the $R \bowtie S$ using hash join assuming: i) The main memory buffer can hold 202 blocks, ii) The main memory buffer can hold 11 blocks.