CS127 Quiz #2

Date: November 13th, 2019 at 3:00 pm

Your BannerID

BannerID: _____

Write your solutions on this piece of paper and hand it in.

Question #1 General Topics

30 points

A. Unlike ISAM, _____, and _____ dynamically adjust to inserts and deletes.

a. clustered indexes, non-clustered indexes
b. dense indexes, sparse indexes
c. sequential indexes, static hash indexes
d. B+-Trees, extendible hashing indexes
e. slotted page structures, multitable clusterings

B. _____ allow for better compression of data in memory than _____.

a. B-Trees, sequential indexes
b. multitable clusterings, sequential indexes
c. queries that involve joins, queries that do not involve joins
d. B+-Trees, B-Trees
e. columnstores, rowstores

C. The buffer manager reads blocks into the buffer. When a transaction accesses a block, the buffer manager _____ the block in memory to prevent the block from being written back to disc.

a. sorts
b. deletes
c. pins
d. flushes
e. encrypts

D. _____ enhance performance of queries that involve _____, and decrease performance of queries that involve _____.

a. multitable clusterings, joins, single relations

b. free lists, deletions, insertions

c. hash indexes, ranges, returning a single tuple

d. B+-Trees, returning a single tuple, ranges

e. dense indexes, joins, single relations

E. Given a candidate key attribute, _____ on the attribute may contain exactly one or two copies of the same search key, while _____ on the attribute may contain only one copy of the same search key.

a. dense indexes, sparse indexes

b. non-clustered indexes, clustered indexes

c. extendible hashing indexes, linear hashing indexes

d. B-Trees, dense indexes

e. B+-Trees, B-Trees

F. _____ are well-suited for fixed length tuples, and _____ are well-suited for variable-length tuples.

a. sequential storage structures, slotted page storage structures

b. hash indexes, B+-Tree indexes

c. dense indexes, sparse indexes

d. primary indexes, secondary indexes

e. merge joins, indexed nested loop joins

Question #2 Dense and Sparse Indices

10 points
A sequential file (sorted by the key field) consists of 10,000 records. Blocks are 1000 bytes long; there is no need for a block header. Records are 100 bytes long, of which 12 bytes are the key field. Pointers take 8 bytes.

A. Assuming there is an infinite amount of main memory, the number of blocks required for a sequential dense index on a primary key in this file is (Hint: Don't think B-tree)

a 80

b 120

c 200

d 800

e 1200

B. The number of blocks required for a block-oriented, sparse index on this file is

a 8

b 12

c 20

d 80

e 120

Question #3 B+-Trees

10 points

A. A B+-tree of order 4 and of height 3 will have a maximum of _____ leaves.

a. 64
b. 128
c. 186
d. 256
e. 512

B. Consider a B+-Tree of order 3 (each node has 3 children, and 2 keys), the tree here shows the **keys** of the tree:

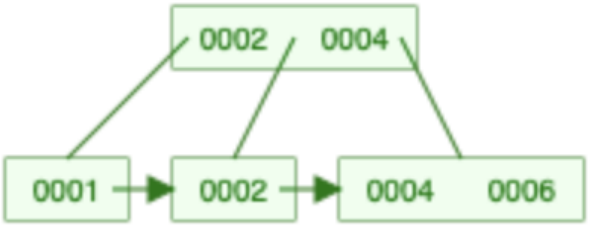

Figure 1: B+-Tree Insertion Problem

What is the height of the tree after inserting 3 then 7 (in that order)?

a. 0
b. 1
c. 2
d. 3
e. 4

Question #4 Extendable Hashing

15 points
With x as the employee name, hash h(x) returns the following 10-bit values:

x	h(x)
Pete	1001111010
Mary	0100000000
Jane	1100011110
Bill	0110000000
John	0001101001
Vince	1101110101
Karen	0000110111

Using the above values, and a bucket size of 2, we draw a picture of the state of an extendable hash table after records with the following keys are inserted in the given order:
Pete, Karen, Mary, Vince, Jane, John, Bill

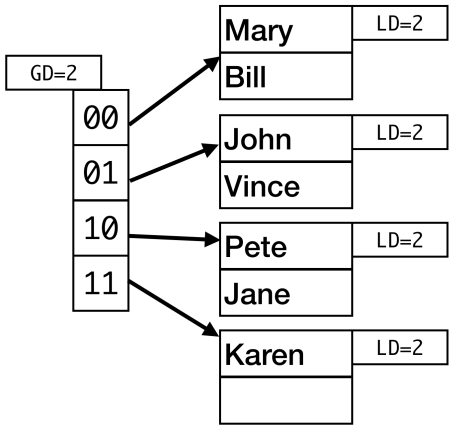

Figure 2: Extendable Hashing problem

A. Does this picture change if we insert the keys in the opposite order?

B. Draw an altered picture to show what happens when we add a record for Sol to the extendable hash table in Figure 2 if h(Sol) = 0001010001.

Question #5 Linear Hashing

15 points
Your linear hash table uses a hash function h(x) = x. Each bucket in the table has a size = 2. The table starts with 2 buckets. After hashing 5, 4, 6, and 3, it looks like this. Note the split pointer is pointing at bucket 0.

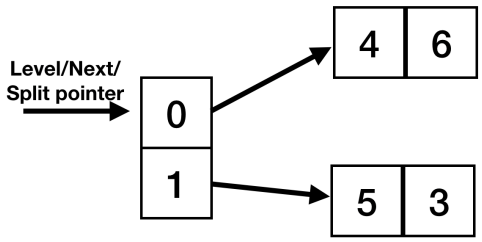

Figure 3: Linear Hashing problem

A. Draw the result after inserting 2. Your drawing should show the final location of the split pointer after the operation is completed.

B. Draw the result after inserting 10. Your drawing should show the final location of the split pointer after the operation is completed.

Question #6 Query Processing

20 points
Consider

- Relations R(A, B) and S(A, C) with the b_r and b_s number of blocks, and n_r and n_s numbers of records, respectively.
- Neither R nor S fit in memory

A. Block nested loop join for relations R and S has two for loops: an inner and outer. If main memory could fit M blocks, and there are no indexes, how many iterations does the outer loop go through?

B. R has a clustered index on A (implying that the relation's tuples are sorted on A). If we wanted to use the index to satisfy the query SELECT * from R where A $>$ 10, how many times do we need to traverse the index?

C. If S also has a clustered index on A, what is the lowest-cost way of joining R and S on A?