Midterm 1

CS386D Database Systems
Date: 3/2/2020
Version: 1.0

Last Initial:	Name:

1. **Definitions** – Define the following terms (50)

a) Dense Index

Every record's key value is stored in the index

Note: *The text emphasizes that, as a secondary index, a B+ tree will be dense, and emphasizes that that means pointers to every data record, and thus a dense index requires more space. I emphasize that all the values in the indexed column are replicated. A consequence of that is that certain queries can be processed without accessing the data file.*
Answers that spoke to the textbooks emphasis, but did not mention the replication of key values in the dense index were docked 1 point.

b) Access Path
A method for fetching data from disk.
or
The sequence of blocks/pages required to locate and retrieve data

c) Join as in R ⋈ S, a succinct answer in algebraic notation is prefered

$R \bowtie S = \sigma_{\Theta} (R \times S)$
where $\Theta = R.ai = S.bj$ *for all attributes ai and bj that have the same name*

d) Right semi-join. R. ⋉ S a succinct answer in algebraic notation is prefered.

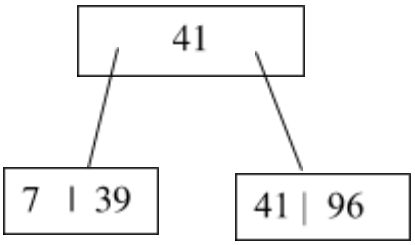
$R \ltimes S = \pi_{attr(S)} (R \bowtie S)$

e) Horizontal Hash Partition (on attribute A)

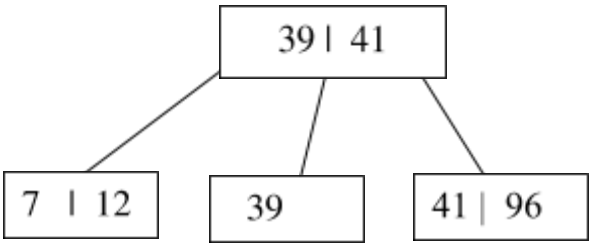
Let Pi be a server or compute node labeled with integer i, for P servers
Then a tuple, or row, t is assigned to Pi iff hash(t.A) = i

2) B+ Trees (40)

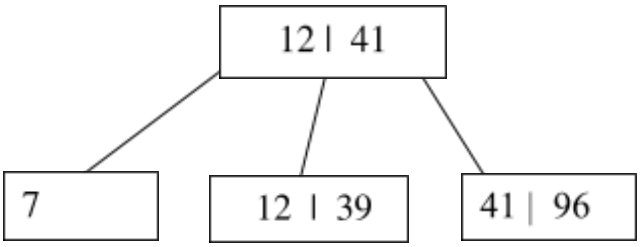
a) Consider the B+-tree illustrated below. A maximum of 2 key values and their corresponding surrogates are allowed in a leaf and 2 split values in an interior node.



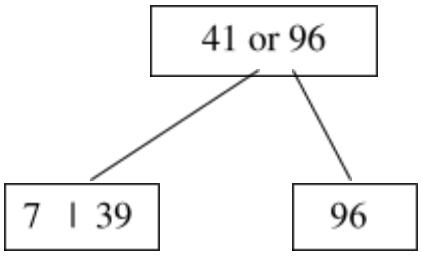
i. Draw the tree resulting from the insert of a record with search key value 12.



OR



ii. Draw the tree resulting from the removal, from the original tree (i.e. 12 was not inserted) of the record with search key value 41.



b) Consider an insert into a B+-tree that caused the root to split. Let each leaf contain a maximum of s surrogates pointing to records in data pages. Similarly, interior nodes contain a maximum of t split values, the height of the tree before the split is h. s,t and h are even numbers. For each of these questions indicate true or false and provide a short explanation.

i. The root will contain t/2 split values

False. The new root will contain a single value.

ii. Before the split, the minimum occupancy of every leaf node is greater than 50%

Intended Answer, but the question should have been worded "strictly greater than"

False. Btree maintains that all leaves have floor((s+1)/2) which is exactly 50% since s is even. (This was a very picky question, you were to detect that the question spoke to "strictly greater than".)

So this question is being struck from the test.

If you did not get full credit for this question, email a scan of this page to Geetali, the subject line should be precisely "midterm question 2b" and she will add points.

3) Bit-mapped Indexes (40)

a) Consider the uncompressed bit map index entry, 01000100100. What is the compressed representation of that entry (using the run-length compression technique covered in class).

01101100

b) Consider the compressed bit map index entry, 101 001010. What is the uncompressed representation of that entry (using the run-length compression technique covered in class).

problem was changed at the exam.
I do not have any exam papers with me, so answer is pending…

Note: *Both padded and unpadded answers are correct.*

Consider a table with the following schema, Grades(EID, Name, Course, Grade). Suppose the university offers 1024 different courses, and the grades assigned are from the set {A,B,C,D, F}. Make the usual random and uniform simplifying assumptions. The Grades table contains n rows and a bitmap index is built using a search key on Course.

c) If an uncompressed bitmap index is built for the column Course, how many *bits* are required to store the bitmaps?

This question is, in part, the kind of question that provides too much information, contributing to its difficulty. In this regard, the intention was for you to notice the question asked about an index on column Course. Thus, the column Grade and it's values are not part of the answer.

Given there are 1024 courses, and it was stipulated that you may simplify the problem by assuming uniform distribution of the data on average, the runs will contain 1023, zeros. There will be 1024 bitmaps, 1 for each course. For n rows, there will be, on average, n/1024 runs.

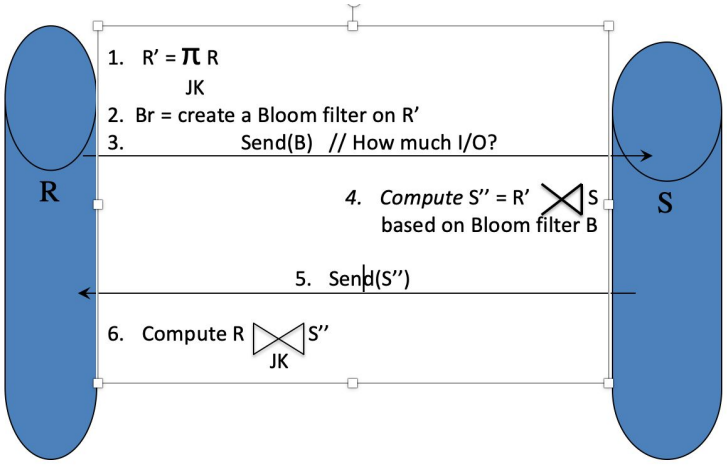
*Thus, 1024 bits per run * (n/1024) runs * 1024 bitmaps = 1024 n*
Note: *Binary marking for solutions with no explanation.*

d If run length encoding is used as a compression method, *approximately* how many bits are required to store the bitmap index on Course? (assume every 1024 bit is set).

Given there are 1024 courses, and it was stipulated that you may simplify the problem by assuming uniform distribution of the data on average, the runs will contain 1023, zeros. There will be 1024 bitmaps, 1 for each course. For n rows, there will be, on average, n/1024 runs.
With compression:
*Thus, 20 bits per run * (n/1024) runs * 1024 bitmaps = 20n*
Note: *Binary marking for solutions with no explanation.*

4) Bloom Filter and Semi-join Reduction (40)

Consider the use of a Bloom filter to approximate a semi-join reduction per the use of semi-join reduction to reduce network communication for a join between two tables, each table residing on a different server. i.e.R ⋈ S The relevant slide from the lecture is replicated below.
R.jk = S.jk



Note, for reasons that will become clear, the Bloom filter previously labeled B, is now labeled Br.

Circle the number of the correct answer. (choices are labeled by small roman numerals, e.g. i, ii.)

A. For a given false positive rate, the size of the Bloom Filter is dependent on

   i. The number of rows in R

   ii. The number of distinct values in column R.jk

B. Assume that the server storing R has no information, whatsoever, about table S, and vice versa. Further R has just 16 rows and column R.jk is also the primary key. One might consider creating a bitmap index on R.jk and communicating that index across the network to the server with S, (in lieu of creating and communicating a Bloom filter). For a Bloom filter if we assume each row of R has a unique primary key and there will be 10 bits used for every row/value, then we will have a false positive rate of around . In comparison the longest bitmap in the bitmap index will be 15, zero's, followed by a 1. Thus, compressed, that bitmap will require just 8 bits, and the representation is exact. Thus,

   i. One should send bitmaps that represent R.jk to the server storing S rather than a Bloom Filter

   ii. One should still send a Bloom Filter, per the algorithm described in class.

The bitmaps are only useful if you also know the value that corresponds to each map. So, if there is some advantage to using bit maps, one would also have to consider the additional I/O of transmitting the values.

C. Suppose the problem is extended to three relations. That is rather than executing a join between R and S, where R and S are stored on two different servers, we now consider joining 3 tables, R, S and T where R,S and T are each stored on a different server. For clarity, the SQL query follows, and none of the arguments (*.jk) is a primary key

```
Select *
From R, S, T
Where R.jk = S.jk AND S.jk = T.jk
```

Notice, by transitivity, R.jk = T.jk, but that operation is not represented in the query and need not be explicitly computed during query execution.

To extend the algorithm to three tables, S'' in the figure above can be represented by a Bloom filter, Bs''populated by the values in S''.jk. In turn, that Bloom Filter, Bs'' can be sent to the third server and used to compute an approximation of S'' ⋈ T''. T'' is lossless wrt the query above the same way S'' is lossless for the two table version, and the final query result can be computed using T''. T'' may contain far fewer rows than T.

There are two ways to compute Bs'',

One, given S'' compute Bs''

Two, arrange, apriori, that there is a Bloom Filter, Bs, representing the contents of S.jk. Then Bs'' can be computed by ANDing Br and Bs

Suppose we choose number two, a priori, have Bloom filters Br and Bs and be able to AND them together. Finally, a maximum false positive rate is given for Bs''. The size in bits of the Bs'' Bloom Filter is determined by (circle the best answer)

i. The number of rows in S

ii. The number of unique values in S.jk

iii. The number of rows in R, plus the number of rows in S

iv. The number of unique values in R.jk, plus the number of unique values in S.jk

v. The greater of the number of rows in R and the number of rows in S

vi. The greater of the number of unique values in R.jk and the number of unique values in S.jk

To size a filter for a given false positive rate one must know the number of unique values that will be stored in the filter. Since the problem states a maximum false positive rate, than we must consider the maximum number of different values that may be represented in the Bs'' filter. The maximum number of actual values will be no more than the greater of the number of values stored in Br and Bs. Although Bs'' is the result of ANDing Br and Bs, although the probability may be infinitesimal, due to false positives, there is no assurance that no set bits in the more densely populated Bloom Filter will, by virtue of the AND, be reset to 0. So, the greater of the number of unique values in R.jk and the number of unique values in S.jk

D. In summary, the three table problem will entail 3 Bloom Filters, Br, a filter on the jk column of R, similarly Bs, and Bs''. The best implementation comprises

   i. The three Bloom filters having the same number of bits, but the number of hash functions is optimized for each of the three filters and their anticipated contents

   ii. The three Bloom filters having the same number of bits and the same number and definition of hash functions used.

The contents of two Bloom filters are only comparable if the filters have the identical number of bits, and the bits are set using identical hash functions.

Thus to AND Br and Bs to produce, Bs'' ii must be true.
If Br and Bs are not comparable, then, like the two table version of the algorithm, the actual contents of the S.jk column must be tested for membership in Br. One may create Bs'' from the result of that computation, but if so, Bs served no purpose.