

VIGNESH KUMAR KARTHIKEYAN RAJALAKSHMI

Name

20424508

CWID

Quiz

1

Oct 10, 2018
Due Oct 14, 11:59pm

Quiz 1: CS525 - Advanced Database Organization

Please leave this empty! 1.1

1.2

1.3

Sum

Instructions

- You have to hand in the assignment using your blackboard
- This is an individual and not a group assignment. Fraud will result in 0 points
- For your convenience the number of points for each part and questions are shown in parenthesis.
- There are 3 parts in this quiz
 1. Disk Organization
 2. SQL
 3. Index Structures

BY SUBMITTING THIS EXAM THROUGH THE ONLINE SYSTEM, I AFFIRM ON MY HONOR THAT I AM AWARE OF THE STUDENT DISCIPLINARY CODE, AND (I) HAVE NOT GIVEN NOR RECEIVED ANY UNAUTHORIZED AID TO/FROM ANY PERSON OR PERSONS, AND (II) HAVE NOT USED ANY UNAUTHORIZED MATERIALS IN COMPLETING MY ANSWERS TO THIS TAKE-HOME EXAMINATION.

P. R. Vignesh Kumar

Part 1.1 Disk Organization (Total: 20 Points)

Question 1.1.1 Disk Access (20 Points)

Consider a disk with a sector size of 512 bytes, 2000 tracks per surface, 50 sectors per track, five double-sided platters, and average seek time of 10 msec. Suppose that a block size of 1024 bytes is chosen. Suppose that a file containing 100,000 records of 100 bytes each is to be stored on such a disk and that no record is allowed to span two blocks.

1. How many records fit onto a block?

$$\frac{1024}{100} = 10 \text{ records} \quad \begin{array}{l} 1 \text{ Block} = 1024 \text{ bytes} \\ \text{each record of } 100 \text{ bytes} \end{array}$$
$$\therefore \frac{1024}{100} = 10 \text{ records.}$$

2. How many blocks are required to store the entire file?

10 records in 1 block

$$\therefore 100,000 \text{ records can be stored in } \frac{100,000}{10} = 10,000 \text{ blocks}$$

3. If the file is arranged sequentially on the disk, how many surfaces are needed?

25 blocks per track

$$\therefore 25 \times 2 = 10,000 \text{ (blocks)}$$

$\therefore 400 \text{ tracks}$

$$400/2000 = 1/5$$

Total surfaces = 10

$$\therefore \frac{1}{5} \times 10 = 2$$

$\therefore 2 \text{ surfaces.}$

4. How many records of 100 bytes each can be stored using this disk?

2000 cylinders with 250 blocks per cylinder

$$\therefore 500,000 \text{ blocks. } 1 \text{ block has } 10 \text{ records}$$
$$\therefore 500,000 \text{ blocks can have } 5,000,000 = 5 \times 10^6 \text{ records.}$$

5. If pages are stored sequentially on disk, with page 1 on block 1 of track 1, what page is stored on block 1 of track 1 on the next disk surface?

for 1 track 25 blocks

$$\therefore 25 \text{ pages can be stored in track 1}$$

26th page is stored on block 1 of track 1 on next disk surface.

6. How would your answer change if the disk were capable of reading and writing from all heads in parallel?

If the heads can read and write parallel then all 10 surfaces have page 1 on the block 1 of track 1.

\therefore it will be block 1 of track 1 on next surface has the same page 1.

Part 1.2 SQL (Total: 35 Points)

Consider the following relations:

- Suppliers(sid:integer, sname:string, address:string)
- Parts(pid:integer, pname:string, color:string)
- Catalog(sid:integer, pid:integer, cost:real)

The key fields are underlined, and the domain of each field is listed after the field name. Therefore sid is the key for Suppliers, pid is the key for Parts, and sid and pid together form the key for Catalog. The Catalog relation lists the prices charged for parts by Suppliers. Write the following queries in SQL statements and in relational algebra expression.

Question 1.2.1 (5 Points)

Find the names of suppliers who supply some red part

SQL: $\text{Select } S.\text{sname}$
 $\text{From Suppliers } S, \text{ parts } P, \text{ Catalog } C$
 $\text{where } P.\text{color} = 'red' \text{ AND } C.\text{pid} = P.\text{pid} \text{ AND } C.\text{sid} = S.\text{sid}$

Relational algebra:

$\pi_{\text{sname}} (\pi_{\text{sid}} ((\pi_{\text{pid}} (\sigma_{\text{color} = 'red'} \text{ Parts})) \bowtie \text{Catalog})) \bowtie \text{Suppliers}$

Question 1.2.2 (5 Points)

Find the sids of suppliers who supply some red or green part.

SQL: $\text{Select } C.\text{sid}$
 $\text{from Catalog } C, \text{ parts } P$
 $\text{where } (P.\text{color} = 'red' \text{ or } P.\text{color} = 'green')$
 $\text{AND } P.\text{pid} = C.\text{pid}$

Relational algebra:

$\pi_{\text{sid}} (\pi_{\text{pid}} (\sigma_{\text{color} = 'red' \vee \text{color} = 'green'} \text{ Parts})) \bowtie \text{Catalog})$

Question 1.2.3 (5 Points)

Find the sids of suppliers who supply some red part or are at 10 West 31st Street.

Sol: Select S.sid from Suppliers S
 where S.address = '10 west 31st street' OR
 S.sid in (Select C.sid from parts p, Catalog C where
 P.color = 'red' AND P.pid = C.pid)

$P(R, \pi_{sid}((\pi_{pid \ \& \ color='red'} Part) \bowtie Catalog))$
 $P(R, \pi_{sid}((\pi_{pid \ \& \ color} Part) \bowtie \pi_{sid \ \& \ address='10 \ west \ 31st \ street'} Supplier)))$
 $R_1 \cup R_2$

Question 1.2.4 (5 Points)

Find the sids of suppliers who supply some red part and some green part.

Select C.sid from Parts p, Catalog C where
 P.color = 'red' and P.pid = C.pid and exists (Select P₂.pid from
 Parts p₂, Catalog C₂ where p₂.color = 'green' and C₂.sid = C.sid and
 P₂.pid = C₂.pid)

$P(R_1, \pi_{sid}((\pi_{pid \ \& \ color='green'} Part) \bowtie Catalog))$
 $P(R_2, \pi_{sid}((\pi_{pid \ \& \ color='red'} Part) \bowtie Catalog))$
 $R_1 \cap R_2$

Question 1.2.5 (5 Points)

Find pairs of sids such that the supplier with the first sid charges more for some part than the supplier with the second sid.

Select C₁.sid C₂.sid from Catalog C₁ and Catalog C₂
 where C₁.pid = C₂.pid AND C₁.sid < > C₂.sid and C₁.cost > C₂.cost

$P(R_1, Catalog)$

$P(R_2, Catalog)$

$\pi_{R_1.sid, R_2.sid}(\sigma_{R_1.pid=R_2.pid \wedge R_1.sid \neq R_2.sid \wedge R_1.cost > R_2.cost} (R_1 \times R_2))$

Question 1.2.6 (5 Points)

Write only an SQL query that find the pids of parts supplied by at least two different suppliers.

Select C.pid from Catalog C where exists (Select C₁.sid from Catalog C₁ where C₁.pid = C.pid and C₁.sid <> C₂.sid)

$\rho(R_1, \text{Catalog}) \rho(R_2, \text{Catalog})$

$\pi(R_1.\text{pid} (\sigma_{R_1.\text{pid} = R_2.\text{pid} \wedge R_1.\text{sid} \neq R_2.\text{sid}} (R_1 \bowtie R_2)))$

Question 1.2.7 (5 Points)

Write only an SQL query that find the pids of the most expensive parts supplied by suppliers named Yosemite Sham.

Select C.pid from Catalog C, Supplier S where
S.sname = 'Yosemite Sham' and C.sid = S.sid AND

C.cost ≥ all (Select C₂.cost from Catalog C₂,
Supplier S₂ where S₂.sname = 'Yosemite Sham' and C₂.sid = S₂.sid)

$\rho(R_1, \pi_{\text{sid}} (\sigma_{\text{sname} = \text{'Yosemite Sham'}} \text{Suppliers}))$

$\rho(R_2, R_1 \bowtie \text{Catalog})$

$\rho(R_3, R_2)$

$\rho(R_4, (1 \rightarrow \text{sid}, 2 \rightarrow \text{pid}, 3 \rightarrow \text{cost}), \sigma_{R_3.\text{cost} > R_2.\text{cost}} (R_3 \bowtie R_2))$

$\pi_{\text{pid}} (R_2 - \pi_{\text{sid}, \text{pid}, \text{cost}} R_4)$

Part 1.3 Index Structures (Total: 65 Points)

Question 1.3.1 B⁺-tree Construction (15 Points)

Assume that you have the following table:

Item		
SSN	name	age
2	Pete	13
23	Bob	23
5	John	49
39	Joe	45
29	Alice	77
17	Lily	3
19	Manny	33
7	Gertrud	29
11	Heinz	14
3	Sammy	34

Create a B⁺-tree for table Item on key SSN. Assume that the tree is initially empty and values are added in ascending order. Construct B⁺-tree for the cases where the number of pointers that will fit in one node is as follows:

ascending order

a. Three

2, 3, 5, 7, 11, 17, 19, 23, 29, 39

b. Five

Write down the resulting B⁺-tree after each step and when splitting or merging nodes follow these conventions:

- **Leaf Split:** In case a leaf node needs to be split during insertion and n is even, the left node should get the extra key. E.g, if $n = 2$ and we insert a key 4 into a node [1,5], then the resulting nodes should be [1,4] and [5]. For odd values of n we can always evenly split the keys between the two nodes. In both cases the value inserted into the parent is the smallest value of the right node.
- **Non-Leaf Split:** In case a non-leaf node needs to be split and n is odd, we cannot split the node evenly (one of the new nodes will have one more key). In this case the "middle" value inserted into the parent should be taken from the right node. E.g., if $n = 3$ and we have to split a non-leaf node [1,3,4,5], the resulting nodes would be [1,3] and [5]. The value inserted into the parent would be 4.
- **Node Underflow:** In case of a node underflow you should first try to redistribute values from a sibling and only if this fails merge the node with one of its siblings. Both approaches should prefer the left sibling. E.g., if we can borrow values from both the left and right sibling, you should borrow from the left one.

a) Pointers = 3

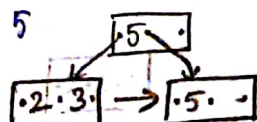
(i) insert 2



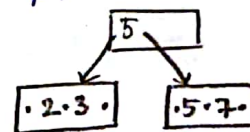
(ii) insert 3



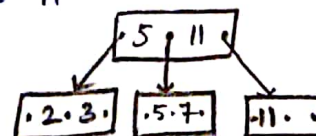
(iii) insert 5



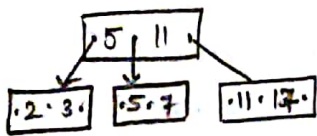
(iv) insert 7



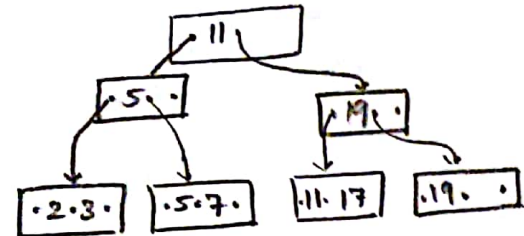
(v) insert 11



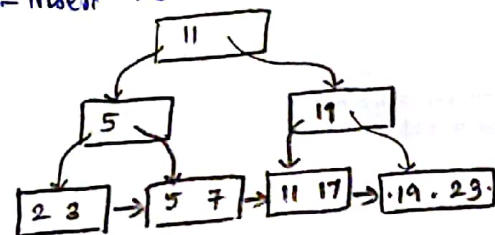
- insert (17):



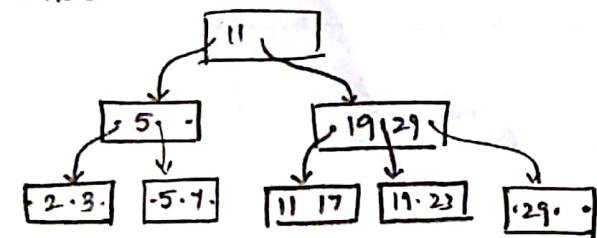
- insert 19:



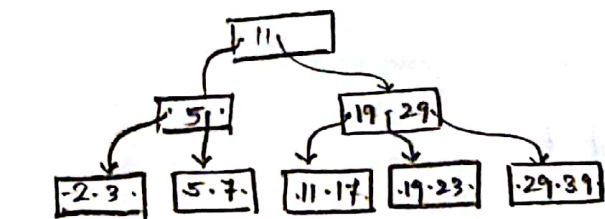
- insert 23



- insert 29

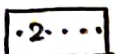


- insert 39

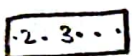


Pointer = 5:-

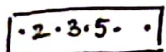
(i) insert 2



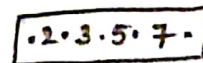
(ii) insert 3



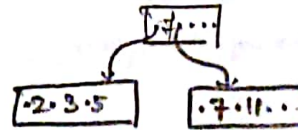
(iii) insert 5



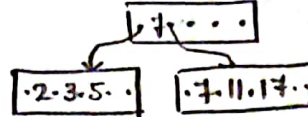
(iv) insert 7



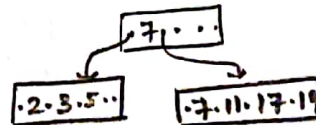
(v) insert 11



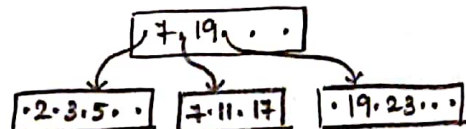
(vi) insert 17



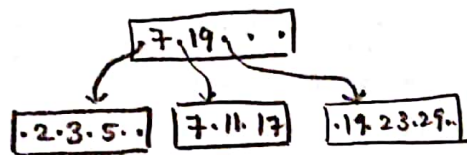
(vii) insert 19



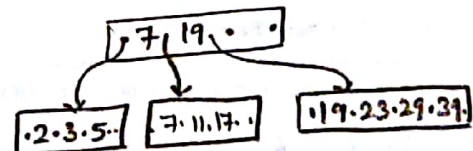
(viii) insert 23



(ix) insert 29



(x) insert 39



Question 1.3.2 Extensible Hashing-tree Construction (25 Points)

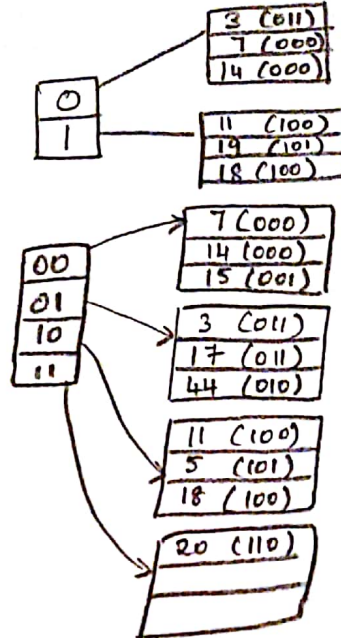
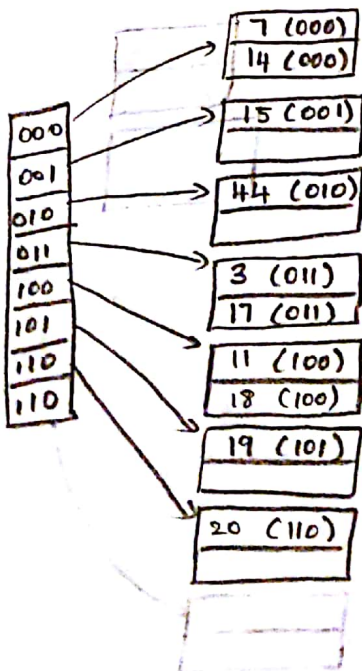
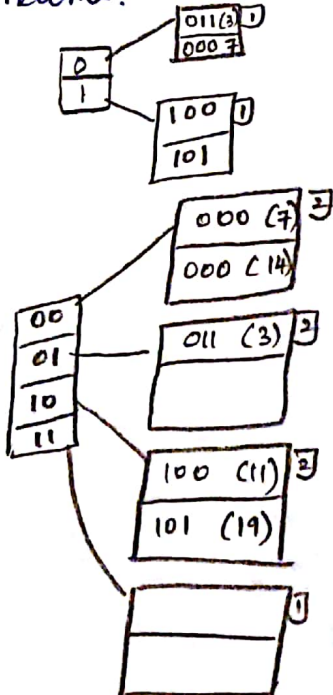
Suppose that we are using Extensible Hashing on a file that contains records with the following search key values: 3, 11, 7, 19, 14, 18, 15, 17, 20, 44, 33, 43

Show the Extensible Hash structure for this file if the hash function is $h(x) = x \bmod 7$ and buckets can hold

a. 2 records

b. 3 records

2 records:



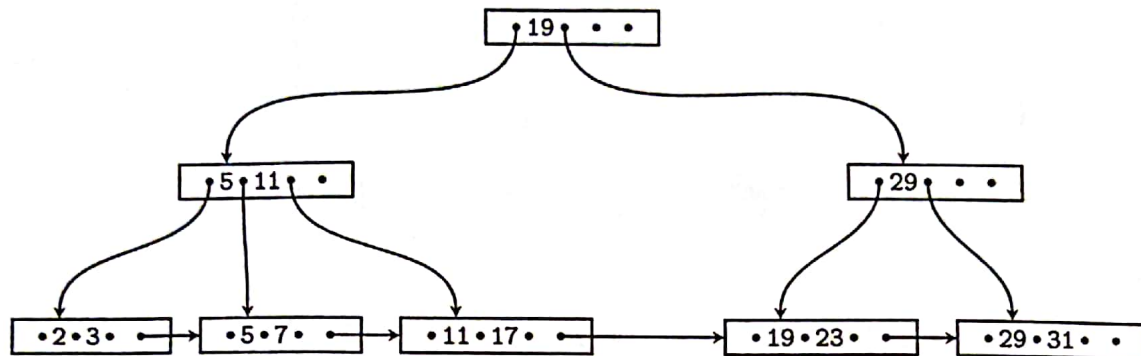
records = 3

Question 1.3.3 Operations (25 Points)

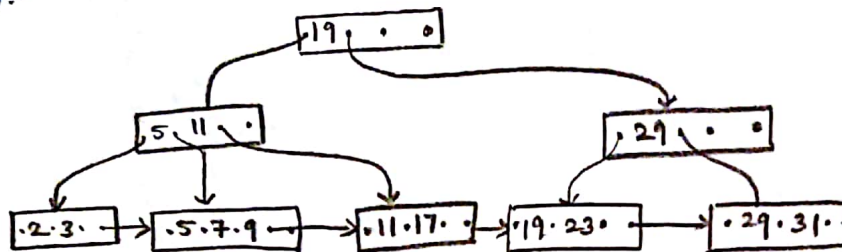
Given is the B⁺-tree shown below ($n = 3$). Execute the following operations and write down the resulting B⁺-tree after each operation:

insert(9), delete(23), insert(10), insert(8), delete(19), delete(11)

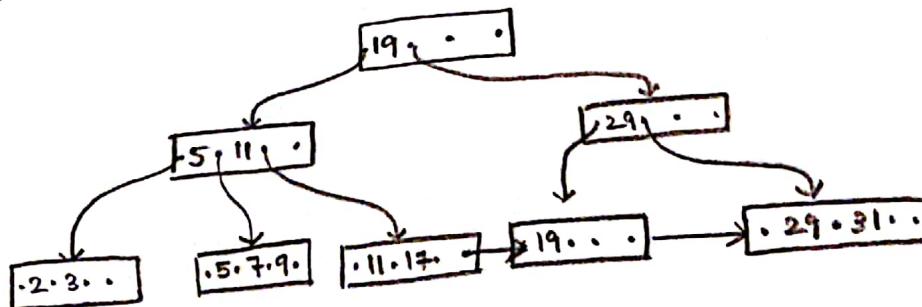
Use the conventions for splitting and merging introduced in the previous question.



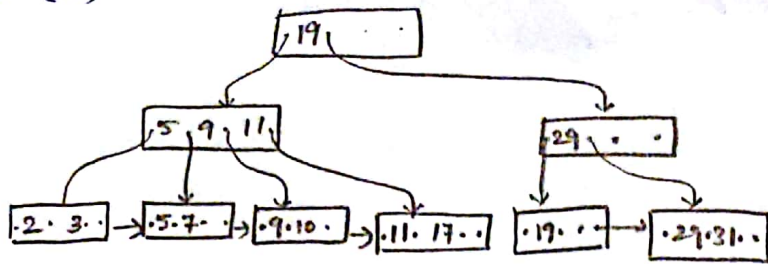
(i) insert (9):-



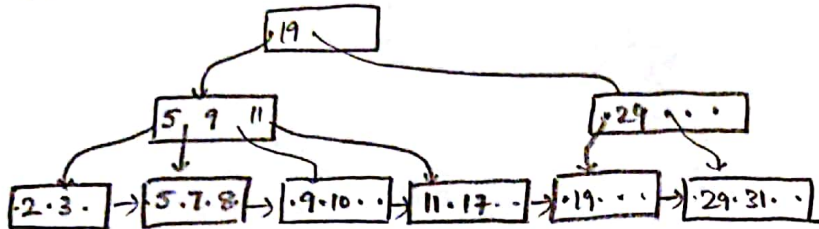
(ii) delete (23)



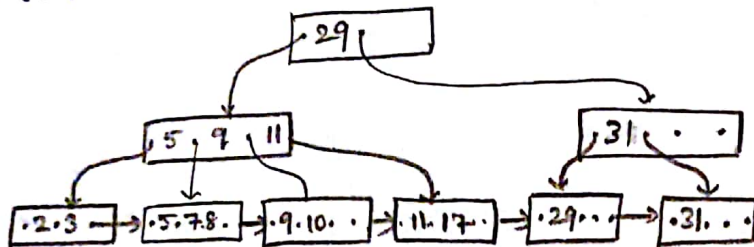
(iii) insert (10) :-



(iv) insert (8) :-



(v) delete (19)



(vi) delete (11) :

