Homework 4   Shruti Pansuriya

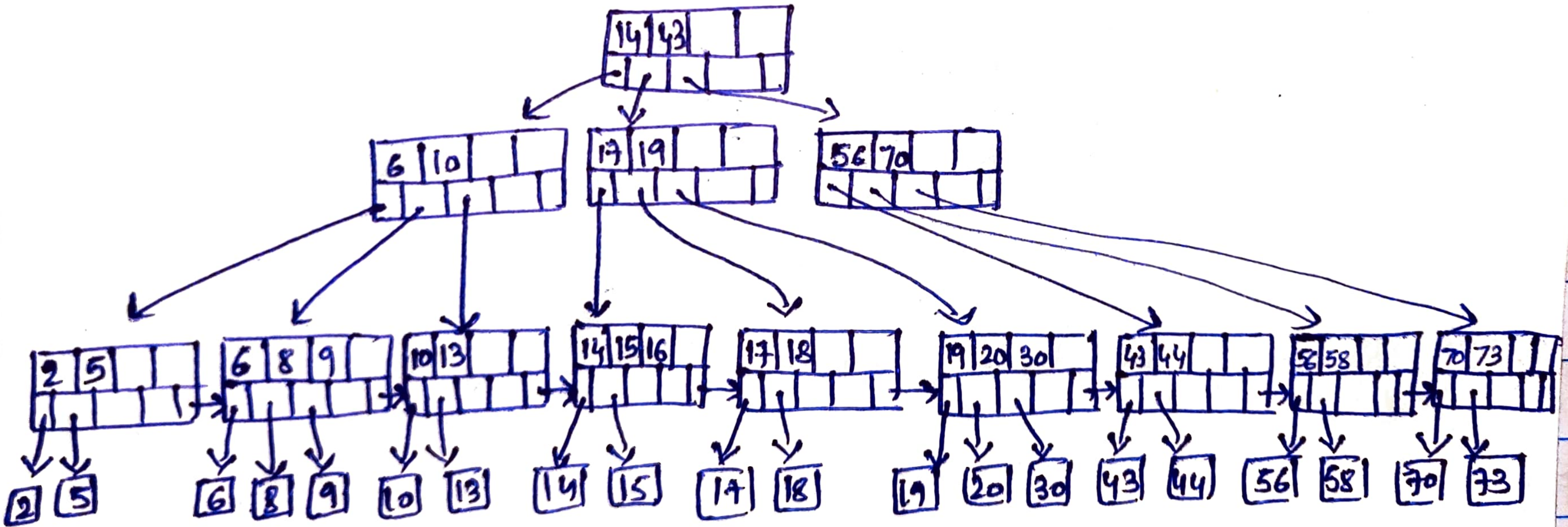
Q.1

Ans(a) Start at root, go to the left child, follow the last pointer (age>=19). i.e proceed down to leaves, now perform sequential traversal of leaves until the condn on age evaluates to false

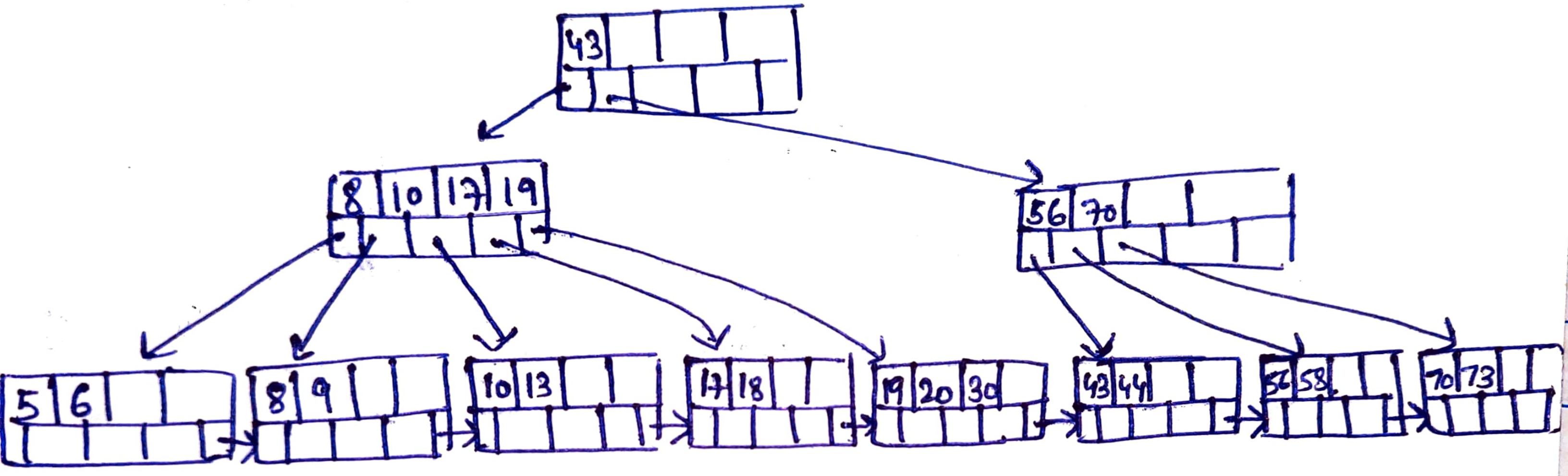
age >= 35 and age <= 65   age ∈ [35, 65]

6 I/O blocks are needed for the process.

Q.1 Ans (b)



Q.1 Ans (c)



After deleting 2, the B+ tree looks like the figure above.

Q.2

Ans (i) Natural joining tables R(a, b) & S(a, c)

(i) Given:-

$M = 102$

$B(R) = 5,000$

$B(S) = 20,000$

Assume o/p of join is given to the next operator in the query execution plan (instead of writing to the disk) and thus the cost of writing the output is ignored.

(a) (Block-based) nested loop join with R as outer relation

(i) steps :-

```
for each (M-2) blocks br of R do
    for each block bs of S do
        for each tuple r in br do
            for each tuple s in bs do
                if r and s join then
                    output(r, s)
```

Total no. of block I/o's needed

(ii) Total Cost :-

[one pass R, 50 passes through S]

$$B(R) + B(R)\,B(S) / (M-2)$$

$$= 5000 + \frac{(5000)(20,000)}{(100)} = 1,005,000$$

b. (Block - based) nested-loop join with S as outer relation

(i) Steps :-

```
for each (M-2) blocks of S do
    for each block br of R do
        for each tuple s in bs do
            for each tuple r in br do
                if s and r join then
                    output (r,s)
```

(ii) Total no. of block I/O's needed

Total cost :-

[one pass S, 200 passes through R]

$$B(S) + B(S)\,B(R)/(M-2)$$

$$\Rightarrow 20,000 + \frac{(20000)(5000)}{(100)} = 1,020,000$$

→ The block nested-loop join saves major block access for a situation where the buffer size is small enough to hold the entire relation into the memory.

(c) Sort merge join

$B(R) = 5000$

$B(S) = 20{,}000$

$M = 101$ (Assuming only 100 pages are used for sorting and 101 pages for merging).

$B(R) + B(S) \overset{x}{\leq} M^2$

$5000 + 20{,}000 > 101^2$ ∴ join cannot be done by using only a single merging pass.

Steps :-

① Pass 1 : sort R ⇒ 50 runs, 100 blocks/run

sort S ⇒ 200 runs, 100 blocks/run

Extra step : 200 runs ⟶ 2 runs 100 blocks/run

Cost for Pass 1 : Read write : 2B(R)

For S, read write : 4B(S)

② Pass 2 : B(R) + B(S)
(merge)

Total cost : 3B(R) + 5B(S)

= 3(5000) + 5(20,000)

= 115000

(d) Partitioned hash join

101 pages used in partitioning of relations and no hash table ip used to lookup in joining tuples.

Assumptions:

$\min(B(R), B(S)) <= M^2$

$\min(5000, 20000) <= M^2 \approx 10000$

Hash based algorithm for binary operations have a size requirement only on the smaller of two inp ut relation.

⇒ Steps:

Pass 1: hash R into 100 buckets, 50 blocks/bucket.

hash S into 100 buckets, 200 blocks/bucket

↓

Hash R into M-1 buckets send all buckets to disk — $2B(R)$

Hash S into M-1 buckets send all buckets to disk — $2B(S)$

Pass 2: join R_i with S_i → $B(R) + B(S)$

Total Cost: $3B(R) + 3B(S)$

⇒ $3(5000) + 3(20,000) = 75000$

Ans.

which algorithm is most efficient in terms of blocks I/O ?

Partitioned hash join