answer–4.1

1605023

- Round-robin technique for tuples distribution in nodes follows the following formula:

i-th tuple goes to (i mod n)th node where

n = total number of nodes.

so, if we have tuples t0, t1, ----, t39, t40, ----

and n = 40,

then t0 goes to (0 % 40) = 0 th node

t1 goes to (1 % 40) = 1 th node

and t39 goes to (39 % 40) = 39 th node.

Date :

tho tuple again returns to 0th node and so on.

Ans.

answer – 4·2

1605023

Here, nodes are n0, n1, ---, n39 in a data center. and tuples with attributes are t0, t1, ---, t40, ----.

So, n = 40 nodes.

i. if NID alone is used as partitioning attribute in hashing, then a hash function designed will generate a value ranging from 0 to 39 based on provided NID. That tuple goes to the hash_value-th node.

ii. using street, city, district (combined character count, location code, etc.) as partitioning attribute to do hashing, the partitioning will work similarly as case – i.

Ans.

answer-4.3

1605023

in a data center,
nodes are $n_0, ---, n_9, n_{10}, ----, n_{19}$; so, $n = 20$ nodes.
tuples are $t_0, t_1, -----, t_{399}$.
ID ranges from 202105001 to 202105400.
So, considering last 3 varying digits.

a. a partition vector based on ID is given:

$V = [v_0, v_1, ---, v_{18}]$. where $v_0 = 21, v_1 = 41, v_2 = 61,$

$----, v_{17} = 361, v_{18} = 381.$

this partition vector is derived by grouping each partition with $\frac{400}{20} = 20$ tuples.

b. here, P0 contains ID from 1 to 20 as $v_0 = 21$.
P1 contains ID from 21 to 40 as $v_0 \leq v < v_1$
or $21 \leq v < 41$.

and P19 contains all IDs greater than 380
as $v_{18} = 381$ and $381 \leq v \leq 400$.

Ans.

answer-4.4

1605023

considering the given schema Person and Parents, we need to process the query:

SELECT * FROM person r, parents s WHERE r.age= s.f-age

a. given partitions are: P1, P2, P3, P4 for both relations. node-i contains partition-i from both Person and Parents tables. Also, query processing is done parallely in 4 nodes.
Here, both relations Person and Parents are partitioned based on age (f-age for parents), a node contains/joins the partitions for same age range from both relation.
So, above query where age equality is checked will be executed by 4 node parallely. If the query specifically asks for an age range, then that query may be executed by one or few nodes leaving other nodes available for other queries. If only one node is needed for processing the query, then the processing will be reduced to $\frac{1}{4}$th of that of a normal case.

b. speed-up: if the query is processed in more than one node, then a linear or sublinear speed up will be observed as processing / task is devided among nodes.

scale-up: new nodes need to be introduced to the system to maintain a linear/ sublinear scale-up. if problem size or query range is increased.

Ans.

answer-5.1

1605023

- Round-robin partitioning
- skewing হতে পারে
chance কম।

Here, we need to explain the performance of Round-Robin partitioning technique for the following scenario:

a. Scanning the entire relation

round-robin is best suited for sequential scan of entire relation on a query as all nodes have almost equal number of tuples.

b. point query

if we know which tuple to look for beforehand, then we can directly retrieve from a specific node.
Else, a sequential scan is required which takes time.

c. range query

In this case, the performance will not be good as now the query needs to be processed at all nodes. It will take a lot of time and no node can process other query then.

Ans.

1605023

answer – 5.2

Here, we need to explain the performance of hash partitioning technique for the following scenario: Date :

a. scanning the entire relation

Hash partitioning is good for sequential access given that hash function is good and partitioning is done on key attributes. Then, tuples are equally distributed among nodes.

b. point query

It performs well in partitioning attribute query. Single node is looked up and other nodes are available for processing other queries.

c. range query

It is inefficient when it comes to range queries because all nodes must be processing the query then. So, parallelism, here, can not be leveraged.

Ans.

answer-5.3

1605023

Here, we need to explain the performance of range partitioning technique for the following scenario:

a. scanning the entire relation

range partitioning provides data clustering by partitioning attribute value. So, it performs good in sequential scanning of tuples.

b. Point query

It performs well in point query on partitioning attribute because only one node needs to be accessed.

c. range query: for range queries, one or few nodes may need to be accessed (query on partitioning attribute). Then, other nodes are available for other queries. So, it performs well if required tuples are from one to a few blocks. Otherwise, potential parallelism in disk access is wasted. (example of execution skew) Ans.

1605023

answer - 6.1

a. Hash partitioning

- certain part. value may repeat in many tuples, so attribute-value, thus, data-distribution skew occurs.

b. range partitioning

- tuples are grouped based on a partitioning value. So, attr.-value skew, thus, data-distribution skew occurs.

## a. Hash Partitioning

- partition skew is less likely to occur as partitioning is based on a hash function.
- execution skew is less likely unless a tuple with a specific value is added multiple times to same node by hashing.

## b. range partitioning

- partition skew is very likely in the case of a badly chosen range-part. vector.
- execution skew may occur if tuples are partitioned based on an attribute like date. Then, all recent tuples will end up in the same node.

Ans.

answer-6.2

1605023

Date

(a.)

The given histogram is an equi-width histogram.

(b.)

definition of an appx. partition vector:

$V = [v_1, v_2, v_3]$ for nodes N0, N1, N2, N3.

$\Rightarrow V = [3, 8, 17]$.

This vector is obtained by the following calculation-

total tuple = $45 \times 5.0 + 35 \times 5.0 + 25 \times 5.0$

$+ 50 \times 5.0 + 15 \times 5.0$

$\Rightarrow$ total tuple = 850

so, each node may have approximately = $\frac{850}{4} = 212-213$ tuples

So, $v1 = 3$ and N1 contains 2 tuples $(1 \le v < 3)$

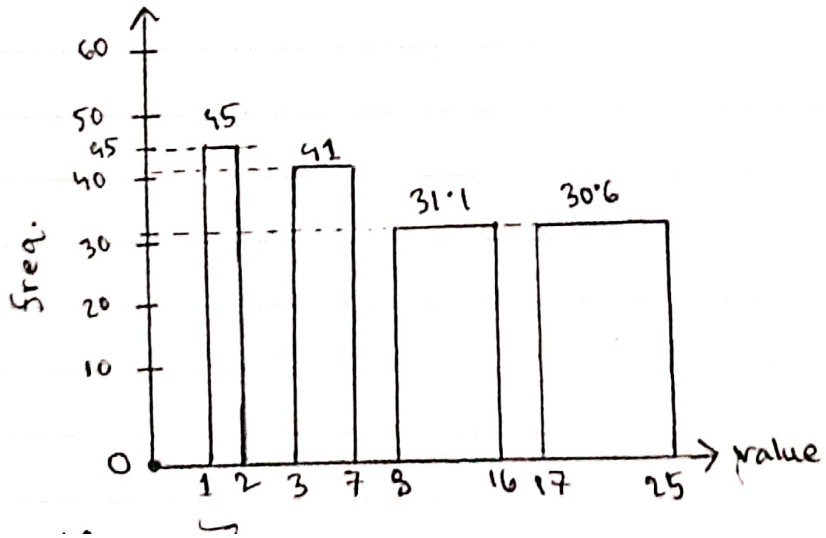
$v2 = 8$ and N2 contains 5 tuples $(3 \le v < 8)$

$v3 = 17$ and N3 contains 9 tuples $(8 \le v < 17)$

and thus, N4 contains other 9 tuples from value 17 to 25. $(17 \le v \le 25)$.

Here, uniform dist. within each range of histogram is assumed.

(c.)




Ans.