answer – 36.1

1605023

Perform schema tuning for the above 2 applications.

Solution

Here, applications are –

a. Bank account verification system

b. National electronic health record system

(a.) Person – BAVS (NID, name, f-name, m-name, DOB, H-no, street, city, thana, district, division, income, profession, qualification, spouse – NID).

(b.) Person – NEHRS (NID, DOB, blood-group, height, weight, BMI, spouse – NID).

Ans.

answer – 36.2

a. Perform schema tuning for the above query by denormalization.

b. Explain the drawbacks.

c. How can these drawbacks be removed?

Solution

(a.) The normalized relations Student and Takes can be joined together to generate a denormalized relation Student-Takes with frequently accessed attributes from both normalized relations. The denormalized relation will, then, be stored separately and all the queries will be processed in this new relation.

b. More space is required in storage as the relations are denormalized. Also, extra operations are required to keep data consistency on updates across multiple relations.

c. Materialized views can be used to remove aforementioned drawbacks.

Ans.

answer—36.3

Explain how the join will be computed very efficiently in the above case.

Solution

Relations/tables that are joined frequently during query processing can be clustered together on the same disk block. Thus, number of seeks (disk head movement) gets reduced. Consequently, join operation takes less time and computation becomes very efficient.

Ans.

answer – 37.1

1605023

How can you improve the above queries?

Solution

The query is given as follows:

```
select sum(salary)
from instructor
where dept_name = ?
```

with

parameters {CSE}, {ME}, {CE}, --- from client side queries.

Here, each call from client to server for individual department has an overhead of network communication as well as query processing at server side. We can improve the performance of above queries by replacing them with the following one:

```
select dept_name, sum(salary)
from instructor
group by dept_name
```

Thus, client side gets the desired results from server side with just one call for all departments. Each department can fetch the required result from this query result at client side. So, number of calls from client to server is reduced to one. Consequently, overhead associated with network (bandwidth) load and database workload gets reduced.

Ans.

answer – 38.1

1605023

a. Lock space is exhausted in long update transactions. Explain.

b. Log space is exhausted in long update transactions. Explain.

c. Recovery time is increased in long update transactions. Explain.

Solution

(a.) In this long update transaction scenario, each data item (associated with one employee) requires individual x-lock, that is, lock-X(empN, TID). x-lock can not be released before the last commit in a specific transaction for a particular record in two-phase lock scheme. As a result, lock space gets exhausted in this scenario since lock manager grants lock for each data item.

(b.) In this long update transaction scenario, each data item has its own log record. As a result, huge amount of log records are stored during such a transaction. Also, they are not usually erased for recovery purpose. Thus, log space gets exhausted in this scenario.

(e.) In this long update transaction scenario, the whole thing is considered as one transaction with log records associated with individual employees. As a result, no checkpoint is set in-between any pair of log records. In fact, checkpoints are set at the beginning and at the end of this one gigantic transaction. In addition to that recovery manager takes recovery measures on the basis of these checkpoints. Thus, recovery time gets increased in this scenario.

Ans.