

MCA E23: Advanced Database Management Systems**[Part – I : Short/Medium Answer Type Questions]****UNIT-I**

- 1.** List the ACID properties. Explain the usefulness of each.
- 2.** Suppose that there is a database system that never fails. Is a recovery manager required for this system?
- 3.** Consider a file system such as the one on your favorite operating system.
 - a.** What are the steps involved in creation and deletion of files, and in writing data to a file?
 - b.** Explain how the issues of atomicity and durability are relevant to the creation and deletion of files, and to writing data to files.
- 4.** Database-system implementers have paid much more attention to the ACID properties than have file-system implementers. Why might this be the case?
- 5.** During its execution, a transaction passes through several states, until it finally commits or aborts. List all possible sequences of states through which a transaction may pass. Explain why each state transition may occur.
- 6.** Justify the following statement: Concurrent execution of transactions is more important when data must be fetched from (slow) disk or when transactions are long, and is less important when data is in memory and transactions are very short.
- 7.** Explain the distinction between the terms serial schedule and serializable schedule.
- 8.** Since every conflict-serializable schedule is view serializable, why do we emphasize conflict serializability rather than view serializability?
- 9.** What is a cascadeless schedule? Why is cascadelessness of schedules desirable? Are there any circumstances under which it would be desirable to allow noncascadeless schedules? Explain your answer.
- 10.** In multiple-granularity locking, what is the difference between implicit and explicit locking?

UNIT-II

1. Show that the two-phase locking protocol ensures conflict serializability, and that transactions can be serialized according to their lock points.

2. Consider the following two transactions:

```
T31: read(A);  
read(B);  
if A = 0 then B := B + 1;  
write(B).  
T32: read(B);  
read(A);  
if B = 0 then A := A + 1;  
write(A).
```

Add lock and unlock instructions to transactions T31 and T32, so that they observe the two-phase locking protocol. Can the execution of these transactions result in a deadlock?

3. What benefit does strict two-phase locking provide? What disadvantages result?

4. What benefit does rigorous two-phase locking provide? How does it compare with other forms of two-phase locking?

5. Most implementations of database systems use strict two-phase locking. Suggest three reasons for the popularity of this protocol.

6. Consider a database organized in the form of a rooted tree. Suppose that we insert a dummy vertex between each pair of vertices. Show that, if we follow the tree protocol on the new tree, we get better concurrency than if we follow the tree protocol on the original tree.

7. Show by example that there are schedules possible under the tree protocol that are not possible under the two-phase locking protocol, and vice versa.

8. Consider the following extension to the tree-locking protocol, which allows both shared and exclusive locks:

- A transaction can be either a read-only transaction, in which case it can request only shared locks, or an update transaction, in which case it can request only exclusive locks.
- Each transaction must follow the rules of the tree protocol. Read-only transactions may lock any data item first, whereas update transactions must lock the root first.

Show that the protocol ensures serializability and deadlock freedom.

9. Consider the following graph-based locking protocol, which allows only exclusive lock modes, and which operates on data graphs that are in the form of a rooted directed acyclic graph.

- A transaction can lock any vertex first.
- To lock any other vertex, the transaction must be holding a lock on the

majority of the parents of that vertex.

Show that the protocol ensures serializability and deadlock freedom.

UNIT-III

- 1.** Discuss the relative advantages of centralized and distributed databases.
- 2.** Explain how the following differ: fragmentation transparency, replication transparency, and location transparency.
- 3.** How might a distributed database designed for a local-area network differ from one designed for a wide-area network?
- 4.** When is it useful to have replication or fragmentation of data? Explain your answer.
- 5.** Explain the notions of transparency and autonomy. Why are these notions desirable from a human-factors standpoint?
- 6.** To build a highly available distributed system, you must know what kinds of failures can occur.
 - a.** List possible types of failure in a distributed system.
 - b.** Which items in your list from part a are also applicable to a centralized system?
- 7.** Consider a failure that occurs during 2PC for a transaction. For each possible failure that you listed in Exercise 19.6a, explain how 2PC ensures transaction atomicity despite the failure.
- 8.** Consider a distributed system with two sites, A and B. Can site A distinguish among the following?
 - B goes down.
 - The link between A and B goes down.
 - B is extremely overloaded and response time is 100 times longer than normal.

What implications does your answer have for recovery in distributed systems?

- 9.** The persistent messaging scheme described in this chapter depends on timestamps combined with discarding of received messages if they are too old. Suggest an alternative scheme based on sequence numbers instead of timestamps.
- 10.** Give an example where the read one, write all available approach leads to an erroneous state.
- 11.** Under what conditions is it less expensive to avoid deadlock than to allow deadlocks to occur and then to detect them?
- 12.** If deadlock is avoided by deadlock avoidance schemes, is starvation still possible? Explain your answer.

UNIT-IV

1. If we apply a distributed version of the multiple-granularity protocol to a distributed database, the site responsible for the root of the DAG may become a bottleneck. Suppose we modify that protocol as follows:

- Only intention-mode locks are allowed on the root.
- All transactions are given all possible intention-mode locks on the root automatically.

Show that these modifications alleviate this problem without allowing any nonserializable schedules.

2. Explain the difference between data replication in a distributed system and the maintenance of a remote backup site.

3. Give an example where lazy replication can lead to an inconsistent database state even when updates get an exclusive lock on the primary (master) copy.

4. Study and summarize the facilities that the database system you are using provides for dealing with inconsistent states that can be reached with lazy propagation of updates.

5. Discuss the advantages and disadvantages of the two methods for generating globally unique timestamps.

6. Consider a relation that is fragmented horizontally by plant-number:

employee (name, address, salary, plant-number)

Assume that each fragment has two replicas: one stored at the New York site and one stored locally at the plant site. Describe a good processing strategy for the following queries entered at the San Jose site.

- a. Find all employees at the Boca plant.
- b. Find the average salary of all employees.
- c. Find the highest-paid employee at each of the following sites: Toronto, Edmonton, Vancouver, Montreal.
- d. Find the lowest-paid employee in the company.

7. Consider the relations

employee (name, address, salary, plant-number)

machine (machine-number, type, plant-number)

Assume that the employee relation is fragmented horizontally by plant-number, and that each fragment is stored locally at its corresponding plant site. Assume that the machine relation is stored in its entirety at the Armonk site. Describe a good strategy for processing each of the following queries.

- a. Find all employees at the plant that contains machine number 1130.
- b. Find all employees at plants that contain machines whose type is "milling machine."
- c. Find all machines at the Almaden plant.
- d. Find employee $\bowtie$ machine.

8. For each of the strategies of Exercise 19.18, state how your choice of a strategy depends on:

- a. The site at which the query was entered
- b. The site at which the result is desired

- 9.** Is $r_i \text{ SJ } r_j$ necessarily equal to $r_j \text{ SJ } r_i$? Under what conditions does $r_i \text{ SJ } r_j = r_j \text{ SJ } r_i$ hold?
- 10.** Given that the LDAP functionality can be implemented on top of a database system, what is the need for the LDAP standard?
- 11.** Describe how LDAP can be used to provide multiple hierarchical views of data, without replicating the base level data.

UNIT-V

- 1.** Explain the difference between the three storage types—volatile, nonvolatile, and stable—in terms of I/O cost.
- 2.** Stable storage cannot be implemented.
 - a.** Explain why it cannot be.
 - b.** Explain how database systems deal with this problem.
- 3.** Compare the deferred- and immediate-modification versions of the log-based recovery scheme in terms of ease of implementation and overhead cost.
- 4.** Assume that immediate modification is used in a system. Show, by an example, how an inconsistent database state could result if log records for a transaction are not output to stable storage prior to data updated by the transaction being written to disk.
- 5.** Explain the purpose of the checkpoint mechanism. How often should checkpoints be performed? How does the frequency of checkpoints affect
 - System performance when no failure occurs
 - The time it takes to recover from a system crash
 - The time it takes to recover from a disk crash
- 6.** When the system recovers from a crash, it constructs an undo-list and a redo-list. Explain why log records for transactions on the undolist must be processed in reverse order, while those log records for transactions on the redo-list are processed in a forward direction.
- 7.** Compare the shadow-paging recovery scheme with the log-based recovery schemes in terms of ease of implementation and overhead cost.
- 8.** Consider a database consisting of 10 consecutive disk blocks (block 1, block 2, . . . , block 10). Show the buffer state and a possible physical ordering of the blocks after the following updates, assuming that shadow paging is used, that the buffer in main memory can hold only three blocks, and that a least recently used (LRU) strategy is used for buffer management.

read block 3

read block 7

read block 5

read block 3

read block 1

modify block 1

read block 10

modify block 5

[Part – II : Long Answer Type Questions]**UNIT-I**

1. Consider the following graph-based locking protocol that allows only exclusive lock modes, and that operates on data graphs that are in the form of a rooted directed acyclic graph.

- A transaction can lock any vertex first.
- To lock any other vertex, the transaction must have visited all the parents of that vertex, and must be holding a lock on one of the parents of the vertex.

Show that the protocol ensures serializability and deadlock freedom.

2. Consider a variant of the tree protocol called the forest protocol. The database is organized as a forest of rooted trees. Each transaction T_i must follow the following rules:

- The first lock in each tree may be on any data item.
- The second, and all subsequent, locks in a tree may be requested only if the parent of the requested node is currently locked.
- Data items may be unlocked at any time.
- A data item may not be relocked by T_i after it has been unlocked by T_i .

Show that the forest protocol does not ensure serializability.

3. Consider a database system that includes an atomic increment operation, in addition to the read and write operations. Let V be the value of data item X .

The operation

increment(X) by C

sets the value of X to $V + C$ in an atomic step. The value of X is not available to the transaction unless the latter executes a read(X). Figure 16.23 shows a lock compatibility matrix for three lock modes: share mode, exclusive mode, and incrementation mode.

a. Show that, if all transactions lock the data that they access in the corresponding mode, then two-phase locking ensures serializability.

b. Show that the inclusion of increment mode locks allows for increased concurrency.

4. In timestamp ordering, $W\text{-timestamp}(Q)$ denotes the largest timestamp of any transaction that executed write(Q) successfully. Suppose that, instead, we defined it to be the timestamp of the most recent transaction to execute write(Q) successfully. Would this change in wording make any difference? Explain your answer.

5. When a transaction is rolled back under timestamp ordering, it is assigned a new timestamp. Why can it not simply keep its old timestamp?

6. Although SIX mode is useful in multiple-granularity locking, an exclusive and intend-shared (XIS) mode is of no use. Why is it useless?

UNIT-II

- 1.** Use of multiple-granularity locking may require more or fewer locks than an equivalent system with a single lock granularity. Provide examples of both situations, and compare the relative amount of concurrency allowed.
- 2.** Consider the validation-based concurrency-control scheme. Show that by choosing $\text{Validation}(T_i)$, rather than $\text{Start}(T_i)$, as the timestamp of transaction T_i , we can expect better response time provided that conflict rates among transactions are indeed low.
- 3.** Show that there are schedules that are possible under the two-phase locking protocol, but are not possible under the timestamp protocol, and vice versa.
- 4.** For each of the following protocols, describe aspects of practical applications that would lead you to suggest using the protocol, and aspects that would suggest not using the protocol:
 - Two-phase locking
 - Two-phase locking with multiple-granularity locking
 - The tree protocol
 - Timestamp ordering
 - Validation
 - Multiversion timestamp ordering
 - Multiversion two-phase locking
- 5.** Under a modified version of the timestamp protocol, we require that a commit bit be tested to see whether a read request must wait. Explain how the commit bit can prevent cascading abort. Why is this test not necessary for write requests?
- 6.** Explain why the following technique for transaction execution may provide better performance than just using strict two-phase locking: First execute the transaction without acquiring any locks and without performing any writes to the database as in the validation based techniques, but unlike in the validation techniques do not perform either validation or perform writes on the database. Instead, rerun the transaction using strict two-phase locking. (Hint: Consider waits for disk I/O.)

UNIT III

- 1.** Consider the timestamp ordering protocol, and two transactions, one that writes two data items p and q, and another that reads the same two data items. Give a schedule whereby the timestamp test for a write operation fails and causes the first transaction to be restarted, in turn causing a cascading abort of the other transaction. Show how this could result in starvation of both transactions. (Such a situation, where two or more processes carry out actions, but are unable to complete their task because of interaction with the other processes, is called a livelock.)
- 2.** Explain the phantom phenomenon. Why may this phenomenon lead to an incorrect concurrent execution despite the use of the two-phase locking protocol?
- 3.** Devise a timestamp-based protocol that avoids the phantom phenomenon.
- 4.** Explain the reason for the use of degree-two consistency. What disadvantages does this approach have?
- 5.** Suppose that we use the tree protocol of Section 16.1.5 to manage concurrent access to a B+-tree. Since a split may occur on an insert that affects the root, it appears that an insert operation cannot release any locks until it has completed the entire operation. Under what circumstances is it possible to release a lock earlier?
- 6.** Give example schedules to show that if any of lookup, insert or delete do not lock the next key value, the phantom phenomenon could go undetected.

UNIT IV

- 1.** Explain how the buffer manager may cause the database to become inconsistent if some log records pertaining to a block are not output to stable storage before the block is output to disk.
- 2.** Explain the benefits of logical logging. Give examples of one situation where logical logging is preferable to physical logging and one situation where physical logging is preferable to logical logging.
- 3.** Explain the reasons why recovery of interactive transactions is more difficult to deal with than is recovery of batch transactions. Is there a simple way to deal with this difficulty? (Hint: Consider an automatic teller machine transaction in which cash is withdrawn.)
- 4.** Sometimes a transaction has to be undone after it has committed, because it was erroneously executed, for example because of erroneous input by a bank teller.
 - a.** Give an example to show that using the normal transaction undo mechanism to undo such a transaction could lead to an inconsistent state.
 - b.** One way to handle this situation is to bring the whole database to a state prior to the commit of the erroneous transaction (called point-in-time recovery).

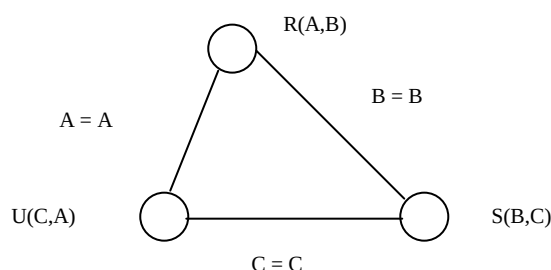
Transactions that committed later have their effects rolled back with this scheme.

Suggest a modification to the advanced recovery mechanism to implement point-in-time recovery.

- c.** Later non-erroneous transactions can be reexecuted logically, but cannot be reexecuted using their log records. Why?
- 5.** Logging of updates is not done explicitly in persistent programming languages. Describe how page access protections provided by modern operating systems can be used to create before and after images of pages that are updated.
- 6.** ARIES assumes there is space in each page for an LSN. When dealing with large objects that span multiple pages, such as operating system files, an entire page may be used by an object, leaving no space for the LSN. Suggest a technique to handle such a situation; your technique must support physical redos but need not support physiological redos.
- 7.** Explain the difference between a system crash and a "disaster."
- 8.** For each of the following requirements, identify the best choice of degree of durability in a remote backup system:
 - a.** Data loss must be avoided but some loss of availability may be tolerated.
 - b.** Transaction commit must be accomplished quickly, even at the cost of loss of some committed transactions in a disaster.
 - c.** A high degree of availability and durability is required, but a longer running time for the transaction commit protocol is acceptable.

UNIT V

1. A quorum-based commit protocol is implemented to deal with network partitioning of a four-site system. Sites 1 and 3 have the same importance. Sites 2 and 4 are more important than Sites 1 and 3. Site 2 is more important than Site 4. How do you decide on the smallest number of votes for V (total votes), V1, V2, V3, V4, Vc, and Va if we want the partitioned system to abort if and only if Sites 2 and 4 are in the same partition (but cannot abort if Sites 2 and 4 are in different partitions)?
2. An Employee table has the following relational scheme: "Employee (name, sal, loc, mgr)," where name is the primary key. The table has been horizontally fragmented into SP and MPLS fragments. SP has employees who work in St. Paul and MPLS contains all employees who work in Minneapolis. Each fragment is stored in the city where the employees are located. Assume transactions only enter the system in NY and there are no employees in NY. Write down the local schemas and global schema and indicate in which cities these schemas are located for the following three cases:
 - (A) The system does not provide for any transparencies.
 - (B) The system provides for location and replication transparencies.
 - (C) The system provides for location, replication, and fragmentation transparencies.
3. What is the condition for the optimistic concurrency control for transactions overlapping as (A) Ti completes its write phase before Tj starts its read phase, (B) Ti and Tj complete their validation phase at the same time, (C) Ti completes its validation phase before Tj completes its validation phase, (D) Ti completes its read phase after Tj completes its validation, and (E) both Ti and Tj complete their read phase exactly at the same time?
4. Assume the cost of sending a message from any site to any other site is C units of time for SI validation. Figure out the cost of each one of the phases of the following strategy. Also, assume that there are N transactions and, out of these, M will fail the validation.
 - Phase 1: Control site read-locks and reads all data that needs to be read.
 - Phase 2: Control site performs the validation.
 - Phase 3: Invalid transactions are rejected—locks are released for these transactions.
 - Phase 4: Control site calculates all the new values that need to be written.
 - Phase 5: Control site write-locks information at all the sites it needs (no ACK for locking is required).
 - Phase 6: Control site writes where it needs to write for successful transactions.
 - Phase 7: Control site unlocks at the sites that need to be unlocked.
5. Construct a reducer for R Using semi join programs having the join graph in the figure:



With:

R = (0,1), (3,4), (6,7), (7,1)
 S = (9,2), (4,5), (6,6), (7,7)

$U = (2,3), (5,0), (6,6), (7,7)$

Discuss the good and bad properties of your reducer program.

6. Consider the following two important applications, and disregard all other applications:

- (a)** A request about flight availability; at the terminal, all information about flights is shown.
- (b)** A request about reservations, which includes:
 - 1. Checking whether the passenger's data are already available, and if they are not, insert the passenger's data
 - 2. Checking whether there are seats available (assume no overbooking)
 - 3. Inserting the reservation description

What is required in order for application 2 to be correct?