|  |  |
| --- | --- |
|  | Define transaction. Write its properties.  A transaction is a collection of operations that performs a single logical function in a database application. properties of the transactions are:  • **Atomicity**- Either all operations of the transaction are reflected properly in the database, or none are.  **• Consistency**- Execution of a transaction in isolation (that is, with no other transaction executing concurrently) preserves the consistency of the database.  **• Isolation**- Even though multiple transactions may execute concurrently, the system guarantees that, for every pair of transactions Ti and Tj , it appears to Ti that either Tj finished execution before Ti started or Tj started execution after Ti finished. Thus, each transaction is unaware of other transactions executing concurrently in the system.  • **Durability.** After a transaction completes successfully, the changes it has made to the database persist, even if there are system failures. |
|  | Brief about the modes in which data item may be locked.  A lock is a mechanism to control concurrent access to a data item  Data items can be locked in two modes :  1. exclusive (X) mode. Data item can be both read as well as written. X-lock is requested using lock-X instruction.  2. shared (S) mode. Data item can only be read. S-lock is requested using lock-S instruction. |
|  | When are two schedules conflict equivalent?  The schedules S1 and S2 are said to be conflict-equivalent if the following two conditions are satisfied:   1. Both schedules S1 and S2 involve the same set of transactions (including the ordering of actions within each transaction). 2. Both schedules have the same set of conflicting operations. |
|  | What is serializability? How it is tested?  The Serializability of a schedule is tested using a Serialization graph.  Assume a schedule S1. For S1, a graph called Precedence Graph is constructed. This graph consists of a pair G = (V, E), where E is a set of the edges, and V is a set of all vertices.  ll the transactions participating in the schedule are stored in the vertices. The set of edges is used to contain all edges Ti ->Tj for which one of the three conditions holds:   1. Create a node Ti → Tj if Ti executes write (Q) before Tj executes read (Q). 2. Create a node Ti → Tj if Ti executes read (Q) before Tj executes write (Q). 3. Create a node Ti → Tj if Ti executes write (Q) before Tj executes write (Q). |
|  | What is meant by concurrency control?  In a database management system (DBMS), concurrency control manages simultaneous access to a database. It prevents two users from editing the same record at the same time and also serializes transactions for backup and recovery. |
|  | What is the need for concurrency?  If many transactions try to access the same data, then inconsistency arises. Concurrency control required to maintain consistency data.  For example, if we take ATM machines and do not use concurrency, multiple persons cannot draw money at a time in different places. This is where we need concurrency. |
|  | What are the types of locks?  A lock is a mechanism to control concurrent access to a data item  There are two types of Locks   1. Shared lock 2. Exclusive lock   **Shared lock:**  Shared locks are placed on resources whenever a read operation (select) is performed. Multiple shared locks can be simultaneously set on a resource.  **Exclusive lock:**  Exclusive locks are placed on resources whenever a write operation (INSERT, UPDATE And DELETE) are performed. Only one exclusive lock can be placed on a resource at a time. i.e. the first user who acquires an exclusive lock will continue to have the sole ownership of the resource, and no other user can acquire an exclusive lock on that resource |
|  | What is upgrading and downgrading of locks?  It gives a mechanism for conversion from shared lock to exclusive lock is called as upgrade.  It gives a mechanism for conversion from exclusive lock to shared lock is called as downgrade. |
|  | Define – Lock.  A lock is a variable associated with a data item that describes the status of the item withrespect to possible operations that can be applied to it. Generally, there is one lock for each dataitem in the [database](https://ecomputernotes.com/fundamental/what-is-a-database/advantages-and-disadvantages-of-dbms). Locks are used as a means of synchronizing the access by concurrenttransactions to the database item. |
|  | What are the advantages of concurrency and write the disadvantages of not controlling concurrency?  The ability of a database system which handles simultaneously or a number of transactions by interleaving parts of the actions or the overlapping this is called concurrency of the system.  Advantages of concurrency  The good is to serve many users and provides better throughput by sharing resources.   * Reduced waiting time response time or turn around time. * Increased throughput or resource utilization * If we run only one transaction at a time than the acid property is sufficient but it is possible that when multiple transactions are executed concurrently than database may become inconsistent. * Overlapping with the input-output activity with CPU also makes the response time better. * But interleaving of instruction between transaction may also lead to many problems due to which concurrency control is required.   Problems due to concurrency  There are many which may occur due to concurrency,   1. Dirty read problem 2. Loss update problem/ write - write problem 3. Unrepeatable and phantom read problem 4. Incorrect summary problem |
|  | List the various concurrency control protocols.  The concurrency control protocols ensure the atomicity, consistency, isolation, durability and serializability of the concurrent execution of the database transactions. Therefore, these protocols are categorized as:   * Lock Based Concurrency Control Protocol * Time Stamp Concurrency Control Protocol * Validation Based Concurrency Control Protocol |
|  | Differentiate strict two phase locking protocol and rigorous two phase locking protocol.  the difference between Strict 2-PL and Rigorous 2-PL is that Rigorous is more restrictive, it requires both Exclusive and Shared locks to be held until after the Transaction commits and this is what makes the implementation of Rigorous 2-PL easier. |
|  | Define cascade less schedule. Give example.  If in a schedule, a transaction is not allowed to read a data item until the last transaction that has written it is committed or aborted, then such a schedule is called as a **Cascadeless Schedule**.   * Cascadeless schedule allows only committed read operations. * Therefore, it avoids cascading roll back and thus saves CPU time.   https://www.gatevidyalay.com/wp-content/uploads/2018/06/Cascadeless-Schedule-1.png |
|  | What are two pitfalls (problems) of lock-based protocols? . Define two-phase locking.  The two pitfalls (problems) of lock-based protocols are  **Starvation**  Starvation is the situation when a transaction needs to wait for an indefinite period to acquire a lock.  Following are the reasons for Starvation:   * When waiting scheme for locked items is not properly managed * In the case of resource leak * The same transaction is selected as a victim repeatedly   **Deadlock**  Deadlock refers to a specific situation where two or more processes are waiting for each other to release a resource or more than two processes are waiting for the resource in a circular chain.  **Two-phase locking.**  The protocol that ensures serializability is the two-phase locking protocol. This protocol requires that each transaction issue lock and unlock requests in two phases:  1. Growing phase. A transaction may obtain locks, but may not release any lock.  2. Shrinking phase. A transaction may release locks, but may not obtain any new locks. |
|  | What is dirty read? Give example.  **Dirty Reads** A dirty read occurs when a transaction reads data that has not yet been committed. For example, suppose transaction 1 updates a row. Transaction 2 reads the updated row before transaction 1 commits the update.  https://lh5.googleusercontent.com/APQXvpDANuzG91ijV73OTdida3bazngKOZc6ibEZ5FQQ6cdhvBhnrGJNf5SAus_tMAUUc1vtxtZ8h3CwuLDdev2ilJZtr5-P8v6me3U5q6CkJ8Dz1PiIHwHDdgi7U5KsTQTl-i-F |
|  | What are various states of a transaction?  https://media.geeksforgeeks.org/wp-content/uploads/20200501195048/Tt7.png |
|  | List the Desirable Properties of Transactions  https://lh4.googleusercontent.com/xbzNu-99-k2kgVka00-hp8kdBgFt9aDKiacPuFCozVVDCtiRkKu_1HGAxGq3nPaCQgjs3bkurOKn2oAPdTR0cQ1-XiVcRj0oEUaSyYosy3RE5VRwN1tegcuV8YhIs5WpNh1ZpNYx |
|  | What is commit point?  A transaction **T** is said to reach its commit point only when it completes all its operation that has actually accessed the database and have been executed successfully and the effect of all transaction operation on the database have been recorded in the log.  The transaction is actually said to be committed when it is beyond the commit point and its effect is actually assumed to be permanently recorded or captured in the database.  After this transaction basically writes a commit **record[commit, T]** into the log. |
|  | Define Log?  The log is a sequence of log records, recording all the update activities in the database. In a stable storage, logs for each transaction are maintained. Any operation which is performed on the database is recorded is on the log. |
|  | What are the different isolation levels?  There are four levels of isolations which are explained below −   1. **Read Uncommitted** − It is the lowest level of isolation. At this level; the dirty reads are allowed, which means one can read the uncommitted changes made by another. 2. **Read committed** − It allows no dirty reads, and clearly states that any uncommitted data is committed now it is read. 3. **Repeatable Read** − This is the most restricted level of isolation. The transaction holds read locks on all the rows it references and write locks over all the rows it updates/inserts/deletes. So, there is no chance of non-repeatable reads. 4. **Serializable** − The highest level of civilization. It determines that all concurrent transactions be executed serially. |
|  | Define Deadlock  A **deadlock** is a condition wherein two or more tasks are waiting for each other in order to be finished but none of the task is willing to give up the resources that other task needs. In this situation no task ever gets finished and is in waiting state forever.  Deadlock diagram |
|  | Name the schemes that prevent deadlock  There are two schemes to prevent deadlock called *wound-wait* and *wait-die*.  **Wait\_Die :** An older transaction is allowed to wait for a younger transaction, whereas a younger transaction requesting an item held by an older transaction is aborted and restarted.  **Wound\_Wait :** It is just the opposite of the Wait\_Die technique. Here, a younger transaction is allowed to wait for an older one, whereas if an older transaction requests an item held by the younger transaction, we preempt the younger transaction by aborting it. |
|  | When does the problem of starvation occur in lock?  *Starvation* occurs when one or more threads in your program are blocked from gaining access to a resource and, as a result, cannot make progress. |

PART B

* 1. **How is concurrency performed? Explain the protocol that is used to maintain the concurrency concept.**

**2.What is deadlock? How does it occur? How transactions be written to (i) Deadlock Prevention (ii) Deadlock Detection. (iii) Deadlock Recovery Illustrate with suitable examples.**