**ASSIGNMENT-4**

**Q1-** List the ACID properties. Explain the usefulness of each.

Answer:

ACID properties are categorized as Atomicity, Consistency, Isolation and Durability.

Atomicity: This factor ensures that all the operations of transaction are well reflected in the database, or none are accordingly. In a state of partial execution of a transaction, a failure will not be able to leave the database. A lack in atomicity will result in inconsistencies in the database. This is handled by a component known as recovery system.

Consistency: This factor maintains the database consistency. Usually the application programmer runs the consistency rate by coding transactions. This completely ensures that the transaction execution rate is at a consistent mode.

Isolation: The isolation factor ensures that there is no interference caused during the simultaneous execution of multiple transactions. The isolation process is controlled by the concurrency control system.

Durability: It shows the commitment of the transaction. This unit ensures that no system failure should result in loss of data related to the corresponding transfer funds after a successful execution of a transaction. This is achieved by persistently updating the database. The recovery system of the database handles the durability.

**Q2-** Consider the following two transactions:

T13: read (A);  
 read (B);  
 if A = 1 then B: = B - 1;  
 write (B)

T14: read(B);  
 read(A);  
if B = 1 then A := A - 1;  
 write(A)  
 Let the consistency requirement be A = 1 or B = 1, with A = 1 and B = 1 as the initial values.

1. Show that every serial execution involving these two transactions preserves the consistency of the database.
2. Show a concurrent execution of T13 and T14 that produces a non-serializable schedule.
3. Is there a concurrent execution of T13 and T14 that produces a serializable schedule?

Answer (a):

There are two possible cases in the order of the transaction which are as:

Case 1: T13 🡪 T14.

Case 2: T14 🡪 T13.

Case 1 [T13 🡪 T14] substitution; to meet the consistency of A as a value of 1:

|  |  |  |
| --- | --- | --- |
| Period values | A | B |
| Initial | 1 | 1 |
| After transaction T13 | 1 | 0 |
| After transaction T14 | 0 | 1 |

Consistency value met as A = 1

Case 2 [T14 🡪 T13] substitution; to meet the consistency of B as a value of 1:

|  |  |  |
| --- | --- | --- |
| Period values | A | B |
| Initial | 1 | 1 |
| After transaction T14 | 0 | 1 |
| After transaction T13 | 1 | 0 |

Consistency value met as B = 1

Hence, the required values of A and B met proves that every serial execution involving these two transactions preserves the consistency of the database

Answer (b):

For a concurrent execution of T13 and T14 that produces a non-serializable schedule, the system should execute transactions simultaneously in a parallel manner. It is represented as:

|  |  |
| --- | --- |
| Transaction T13 | Transaction T14 |
| Read (A);  Read (B);  If A = 1, then B: = B – 1  Write (B) | Read (B);  Read (A);  If B = 1, then A: = A – 1  Write (A) |

Answer (c):

There is no parallel execution resulting in a serializable schedule.

**Q3-** What is the two-phase locking protocol? What is the strict two-phase locking protocol? What is the rigorous two-phase locking protocol? What benefit does the strict two-phase locking protocol provide? What benefit does the rigorous two-phase locking protocol provide?

Answer:

**The two-phase locking protocol** ensures the serializability. This protocol requires that each transaction issue lock and unlock requests are categorized in two phases; growing phase and shrinking phase.   
The modified version of a two-phase locking in order to avoid cascading rollbacks is known as **a strict two-phase locking protocol**. This protocol requires that all exclusive-mode locks are performed by a transaction that is to be held until the commitment of that transaction. It also requires fulfilling the job done by two-phase locking protocol.   
**The rigorous two-phase locking protocol** is another variant of the two-phase locking protocol, which requires that all locks be held until the commit of the transaction. This protocol can be used to serialize the transactions in an order to which they are committed.

The strict two-phase locking protocol provides simple and quick recovery process as it produces only cascade-less schedules. It is simple to implement and also imposes low rollback overhead. This protocol allows an acceptable concurrency level.

The rigorous two-phase locking protocol has a beneficial feature to operate for two simultaneous transactions where their commitment order is directly proportional to the serializability order. In addition to that it has the benefits of the strict two-phase locking protocol as well.

**Q4-** Consider the log in Figure 16.5 (page 738). Suppose there is a crash just before the < T0 abort> log record is written out. Explain what would happen during recovery.

Answer:

After a crash, the recovery takes place in two phases which are as redo phase and undo phase. During the recovery in the redo phase, the whole system undergoes a redo mechanism for all the related operations after the record of the last checkpoint. At this point the C is valued at 600. Within this phase, the undo-list contains T0 and T1; after the commit log record is identified, T1 is removed first whilst on the other hand T2 is added after the start log record is identified. After the identification of abort log record, the transaction T0 is removed from the undo-list. This leaves the undo-list with only T2. A and B is valued at 400 and 2000 respectively here.

During the undo phase, the system scans the log backwards from the end (rewind). While scanning if the system identifies a log record of the transaction T2 updating A, the previous value of A will be reset which makes a redo-only log record to write it into the log (A=500; output the redo-only record<T2, A, 500>). After the identification of the state record of T2, an abort record is added to the transaction T2. As the undo-list does not keep any more transactions in its record, the undo phase terminates the whole process in order to complete the recovery. After the overall recovery process, the system states the variables as; A=500, B=2000, C=600. The added log records during recovery process includes the following:

< T2,A,500 >; < T2 abort >; < T0,B,2000 >; < T0, abort >