ACID properties

Atomicity, Consistency, Isolation, Durability

Abhilasha
Lahigude

- Transactions should possess several properties, often called the **ACID** properties; they should be enforced by the concurrency control and recovery methods of the DBMS. The following are the ACID properties:
- **Atomicity:** A transaction is an atomic unit of processing; it should either be performed in its entirety or not performed at all.
- **Consistency preservation:** A transaction should be consistency preserving, meaning that if it is completely executed from beginning to end without interference from other transactions, it should take the database from one consistent state to another.

- **Isolation:** A transaction should appear as though it is being executed in isolation from other transactions, even though many transactions are executing concurrently. That is, the execution of a transaction should not be interfered with by any other transactions executing concurrently.
- **Durability or permanency:** The changes applied to the database by a committed transaction must persist in the database. These changes must not be lost because of any failure.

Atomicity

- The *atomicity property* requires that we execute a transaction to completion.
- It is the responsibility of the *transaction recovery subsystem* of a DBMS to ensure atomicity.
- If a transaction fails to complete for some reason, such as a system crash in the midst of transaction execution, the recovery technique must undo any effects of the transaction on the database.
- On the other hand, write operations of a committed transaction must be eventually written to disk.

Consistency preservation

- The preservation of *consistency* is generally considered to be the responsibility of the programmers who write the database programs or of the DBMS module that enforces integrity constraints.
- A **database state** is a collection of all the stored data items (values) in the database at a given point in time.
- A **consistent state** of the database satisfies the constraints specified in the schema as well as any other constraints on the database that should hold.
- A database program should be written in a way that guarantees that, if the database is in a consistent state before executing the transaction, it will be in a consistent state after the *complete* execution of the transaction,

Isolation

- The *isolation property* is enforced by the *concurrency control subsystem* of the DBMS.
- If every transaction does not make its updates (write operations) visible to other transactions until it is committed, one form of isolation is enforced that solves the temporary update problem and eliminates cascading rollbacks but does not eliminate all other problems.
- There have been attempts to define the **level of isolation** of a transaction.
- A transaction is said to have level 0 (zero) isolation if it does not overwrite the dirty reads of higher-level transactions.
- Level 1 (one) isolation has no lost updates, and level 2 isolation has no lost updates and no dirty reads. Finally, level 3 isolation (also called *true isolation*) has, in addition to level 2 properties, repeatable reads.

Durability or permanency

- the *durability property* is the responsibility of the *recovery subsystem* of the DBMS.
- Recovery systems take care of the changes made in database must not be lost because of any failure.

Thank you!