NAME:Siddarth Sakthi M

REG.NO:230701314

ASSIGNMENT5

ACID is an acronym that represents a set of properties that guarantee that database transactions are processed reliably. The ACID properties are crucial for ensuring data integrity and consistency in database systems, especially in environments where multiple transactions may occur concurrently. Here's a detailed explanation of each property:

1. Atomicity

- **Definition: Atomicity ensures that a transaction is treated as a single, indivisible unit of work. This means that either all operations within the transaction are completed successfully, or none of them are applied at all.**
- **Implication: If a transaction fails at any point, any changes made during that transaction are rolled back, leaving the database in its previous state. This prevents partial updates that could lead to data inconsistency.**
- **Example: In a banking application, if a transaction involves transferring money from Account A to Account B, both the debit from Account A and the credit to Account B must either complete successfully or not happen at all. If the debit succeeds but the credit fails, the transaction should be rolled back to maintain consistency.**

2. Consistency

- **Definition: Consistency ensures that a transaction brings the database from one valid state to another valid state. The integrity constraints defined in the database must be maintained before and after the transaction.**
- **Implication: Any data written to the database must be valid according to all defined rules, including constraints, cascades, and triggers. If a transaction violates any integrity constraint, it should not be allowed to complete.**

- **Example: If a database has a constraint that prevents negative balances in bank accounts, a transaction attempting to withdraw more money than is available should be rejected, ensuring that the database remains in a consistent state.**

3. Isolation

- **Definition: Isolation ensures that concurrently executing transactions do not interfere with each other. Each transaction should appear to be executed in isolation, even if they are running at the same time.**
- **Implication: The results of a transaction should not be visible to other transactions until the transaction is committed. This prevents issues like dirty reads, non-repeatable reads, and phantom reads.**
- **Example: If two transactions are trying to update the same record simultaneously, isolation ensures that one transaction completes before the other can access the record, preventing conflicts and ensuring that each transaction operates on a consistent view of the data.**

4. Durability

- **Definition: Durability guarantees that once a transaction has been committed, its changes will persist in the database, even in the event of a system failure, crash, or power loss.**
- **Implication: After a transaction is successfully completed, the changes made by that transaction are saved permanently, typically through techniques like write-ahead logging or database snapshots.**
- **Example: If a user transfers funds and the transaction is confirmed, the changes to the accounts must remain intact even if the database crashes immediately afterward. The system must ensure that the committed transaction is recoverable.**

Summary

The ACID properties are foundational principles that help maintain the integrity and reliability of database transactions. They ensure that:

- **Atomicity prevents partial transactions,**
- **Consistency maintains data integrity,**

- **Isolation allows concurrent transactions to operate without interference, and**
- **Durability guarantees that committed transactions are saved permanently.**

These properties are especially important in applications where data integrity is critical, such as banking systems, e-commerce platforms, and any scenario involving sensitive or critical data. By adhering to ACID principles, database systems can provide a reliable environment for processing transactions.