Assignment 3
Explain the ACID properties of a transaction in your own words. Write SQL statements to simulate a transaction that includes locking and demonstrate different isolation levels to show concurrency control.

Ans:
The ACID properties of a transaction are essential characteristics that ensure the reliability and consistency of database operations. Here's a brief explanation of each:

1. **Atomicity**: Atomicity guarantees that a transaction is treated as a single unit of work. Either all of its operations are completed successfully, or none of them are. In other words, a transaction is indivisible. If any part of the transaction fails, the entire transaction is rolled back to its original state.

2. **Consistency**: Consistency ensures that the database remains in a valid state before and after the transaction. This means that all constraints, rules, and relationships defined in the database schema are enforced. Transactions must maintain the integrity of the data and the overall database structure.

3. **Isolation**: Isolation defines how transactions interact with each other when executed concurrently. Each transaction should operate independently of others, as if it is the only transaction running on the database. Isolation prevents interference between transactions, ensuring that the results of one transaction do not affect the results of another.

4. **Durability**: Durability guarantees that once a transaction is committed, its changes are permanent and will not be lost, even in the event of a system failure. The changes made by a committed transaction are saved to non-volatile storage (such as disk), ensuring their persistence.

Now, let's simulate a transaction using SQL statements and demonstrate different isolation levels for concurrency control. We'll use a simple scenario of transferring funds between two bank accounts.

```
-- Begin transaction

BEGIN TRANSACTION;



-- Deduct amount from account 1

UPDATE accounts

SET balance = balance - 100

WHERE account_number = '123';



-- Add amount to account 2

UPDATE accounts

SET balance = balance + 100

WHERE account_number = '456';



-- Commit transaction

COMMIT;
```

This SQL script represents a transaction where $100 is transferred from account '123' to account '456'. To demonstrate different isolation levels, let's use the `SET TRANSACTION` statement to set the isolation level before starting the transaction.

-- Set isolation level to Read Committed

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

BEGIN TRANSACTION;

-- SQL statements for transaction

COMMIT;

This sets the isolation level to `READ COMMITTED`, which ensures that transactions only see data committed before their transaction began. Other transactions can't see the uncommitted changes made by this transaction.

-- Set isolation level to Repeatable Read

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;

BEGIN TRANSACTION;

-- SQL statements for transaction

COMMIT;

With `REPEATABLE READ`, a transaction sees a consistent snapshot of the database as it was when the transaction started. This prevents other transactions from modifying the data that this transaction is reading.

-- Set isolation level to Serializable

```sql
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

BEGIN TRANSACTION;

-- SQL statements for transaction

COMMIT;
```