RDBMS FUNDAMENTALS

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.

ACID PROPERTIES :

A transaction is a single logical unit of work which accesses and possibly modifies the contents of a database. Transactions access data using read and write operations.
In order to maintain consistency in a database, before and after the transaction, certain properties are followed. These are called **ACID** properties.

ATOMICITY :

Atomicity ensures that a transaction is treated as a single unit of work. Either all of the operations within the transaction are completed successfully, or none of them are. There is no partial completion. If any part of the transaction fails, the entire transaction is rolled back to its original state, ensuring data integrity.

CONSISTENCY :

Consistency ensures that the database remains in a valid state before and after the transaction. Even though individual operations within a transaction may change data, the overall database state remains consistent. Constraints, such as primary key constraints, foreign key constraints, and domain constraints, are maintained throughout the transaction.

ISOLATION :

Isolation ensures that the operations within a transaction are isolated from other transactions running concurrently. Each transaction should operate as if it is the only transaction running on the database. Isolation prevents interference between transactions and maintains data integrity. Isolation levels define the degree to which transactions are isolated from each other.

DURABILITY :

Durability ensures that once a transaction is committed, its changes are permanently stored in the database, even in the event of a system failure. The changes made by committed transactions should persist and should not be lost due to crashes or errors.

```
CREATE TABLE Accounts(
```

```
AccountId INT PRIMARY KEY,
Balance DECIMAL(10,2)
);

INSERT INTO Accounts VALUES( 1,5000);
INSERT INTO Accounts VALUES( 2,10000);
```

Start the transaction

```
START TRANSACTION;
```

Deduct amount from Account 1

```
UPDATE Accounts
SET Balance = Balance - 100
WHERE AccountID = 1
FOR UPDATE;
```

Add amount to Account 2

```
UPDATE Accounts
SET Balance = Balance + 100
WHERE AccountID = 2
FOR UPDATE;
```

Commit the transaction

```
COMMIT;
```

Demonstrating different isolation level:

Set isolation level to READ COMMITTED

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
```

START TRANSACTION;

SELECT * FROM Accounts;

COMMIT;

Set isolation level to REPEATABLE READ

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;

START TRANSACTION;

SELECT * FROM Accounts;

COMMIT;

Set isolation level to SERIALIZABLE

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

START TRANSACTION;

SELECT * FROM Accounts;

COMMIT;