Review-5

TABLE LOAN1: insert into loan1 values(1234567890,'Business loan','4563896467',1000000,10); insert into loan1 values( 1256844553,'Home loan','5009657657',1550000,15); insert into loan1 values( 1328944889,'Car loan','9834467345',5050000,5); insert into loan1 values(1982465830,'Car loan','6543217890',50000,12); insert into loan1 values( 2134686921,'Home loan','1234678766',550000,12); can you prepare two code for concurrency control and transaction in serially and concurrent

* We start a transaction using **BEGIN TRANSACTION;**.
* We update a loan's amount in the **loan1** table.
* We then select some rows from the **loan1** table.
* Finally, we commit the transaction using **COMMIT;**, which makes all changes permanent.

**Concurrent Execution Example:**

In a concurrent execution approach, multiple transactions can run simultaneously. To handle concurrency and prevent issues like dirty reads or lost updates, we can use locking mechanisms.

1. **Concurrent Transaction 1**

-- Concurrent Transaction 1

START TRANSACTION;

UPDATE loan1 SET amount = amount - 1854 WHERE loan\_id = 1234567890;

INSERT INTO transactions VALUES (127865, 1234567890, 'DEBIT CARD', 1854, NOW());

COMMIT;

Before Transaction:

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1234567890 Business loan 4563896467 1000000 10

After Transaction:

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1234567890 Business loan 4563896467 999146 10

1. **Serial Transaction 1**

-- Serial Transaction 1

START TRANSACTION;

UPDATE loan1 SET amount = amount - 10080 WHERE loan\_id = 1234567890;

INSERT INTO transactions VALUES (167843, 1234567890, 'CREDIT CARD', 10080, NOW());

COMMIT;

Before Transaction

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1234567890 Business loan 4563896467 999146 10

After Transaction:

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1234567890 Business loan 4563896467 989066 10

1. **Concurrent Transaction 2**

-- Concurrent Transaction 2

START TRANSACTION;

UPDATE loan1 SET amount = amount + 98234 WHERE loan\_id = 1256844553;

INSERT INTO transactions VALUES (195648, 1256844553, 'UPI', 98234, NOW());

COMMIT;

Before Transaction:

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1256844553 Home loan 5009657657 1550000 15

After Transaction:

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1256844553 Home loan 5009657657 1648234 15

1. **Serial Transaction 2**

-- Serial Transaction 2

START TRANSACTION;

UPDATE loan1 SET amount = amount + 100230 WHERE loan\_id = 1328944889;

INSERT INTO transactions VALUES (178547, 1328944889, 'UPI', 100230, NOW());

COMMIT;

Before Transaction:

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1328944889 Car loan 9834467345 5050000 5

after Transaction:

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1328944889 Car loan 9834467345 5150230 5

1. **Concurrent Transaction 3**

-- Concurrent Transaction 3

START TRANSACTION;

UPDATE loan1 SET amount = amount + 10000 WHERE loan\_id = 2134686921;

INSERT INTO transactions VALUES (176599, 2134686921, 'NEFT', 10000, NOW());

COMMIT;

Before Transaction:

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

2134686921 Home loan 1234678766 550000 12

After Transaction

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

2134686921 Home loan 1234678766 560000 12

1. **Serial Transaction 3**

-- Serial Transaction 3

START TRANSACTION;

UPDATE loan1 SET amount = amount + 1003 WHERE loan\_id = 1982465830;

INSERT INTO transactions VALUES (123456, 1982465830, 'NEFT', 1003, NOW());

COMMIT;

Before Transaction

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1982465830 Car loan 6543217890 50000 12

After Transaction:

loan\_id loan\_type customer\_id amount duration

-----------------------------------------------------------

1982465830 Car loan 6543217890 51003 12

Therefore, **duration** refers to the duration of the loan specified in the **loan1** table, representing the period for which the loan is intended to be repaid or managed. It's an essential attribute when managing and tracking loans within a financial system