Name:	Danish singh sambyal
UID:	23BCS11172
Subject:	ADBMS
Section:	622-B

Ans 4.3:

Code:

```
-- ==========================================
-- SETUP: Create StudentEnrollments Table
-- ==========================================
DROP TABLE IF EXISTS StudentEnrollments;

CREATE TABLE StudentEnrollments (
    student_id INT PRIMARY KEY,
    student_name VARCHAR(100),
    course_id VARCHAR(10),
    enrollment_date DATE
);

-- Insert sample data
INSERT INTO StudentEnrollments VALUES
```

```sql
(1, 'Ashish', 'CSE101', '2024-06-01'),

(2, 'Smaran', 'CSE102', '2024-06-01'),

(3, 'Vaibhav', 'CSE103', '2024-06-01');


-- ================================================

-- PART A: Simulating a Deadlock

-- Run in two sessions (Session 1 and Session 2)

-- ================================================


-- ===== Session 1 =====

START TRANSACTION;

-- Lock row with student_id=1

UPDATE StudentEnrollments

SET enrollment_date = '2024-06-05'

WHERE student_id = 1;


-- Try to update row 2 (will wait because Session 2 locks it)

UPDATE StudentEnrollments

SET enrollment_date = '2024-06-06'

WHERE student_id = 2;

-- This will cause deadlock with Session 2


-- ===== Session 2 =====

START TRANSACTION;

-- Lock row with student_id=2

UPDATE StudentEnrollments
```

```sql
SET enrollment_date = '2024-06-07'

WHERE student_id = 2;


-- Try to update row 1 (will wait because Session 1 locks it)

UPDATE StudentEnrollments

SET enrollment_date = '2024-06-08'

WHERE student_id = 1;

-- One of the transactions will be aborted automatically

-- ERROR: Deadlock found when trying to get lock


-- =============================================

-- PART B: Applying MVCC (Snapshot Reads)

-- =============================================


TRUNCATE TABLE StudentEnrollments;


INSERT INTO StudentEnrollments VALUES

(1, 'Ashish', 'CSE101', '2024-06-01');


-- ===== Session 1 (Reader) =====

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;

START TRANSACTION;

SELECT * FROM StudentEnrollments WHERE student_id = 1;

-- Output: enrollment_date = 2024-06-01

-- Keep this transaction open, do not commit yet.
```

```sql
-- ===== Session 2 (Writer) =====

START TRANSACTION;

UPDATE StudentEnrollments

SET enrollment_date = '2024-07-10'

WHERE student_id = 1;

COMMIT;


-- ===== Back to Session 1 =====

-- Still sees old value due to MVCC snapshot

SELECT * FROM StudentEnrollments WHERE student_id = 1;

-- Output: enrollment_date = 2024-06-01

COMMIT;


-- After commit, check latest

SELECT * FROM StudentEnrollments WHERE student_id = 1;

-- Output: enrollment_date = 2024-07-10


-- ================================================

-- PART C: Comparing Behavior With and Without MVCC

-- ================================================


-- CASE 1: Without MVCC (Locking with SELECT FOR UPDATE)

-- ===== Session 1 =====

START TRANSACTION;

SELECT * FROM StudentEnrollments WHERE student_id = 1 FOR UPDATE;

-- This locks the row
```

```sql
-- ===== Session 2 =====

SELECT * FROM StudentEnrollments WHERE student_id = 1;

-- This will block until Session 1 commits


-- ===== Session 1 commits =====

COMMIT;


-- CASE 2: With MVCC (Normal SELECT Snapshot Reads)

-- ===== Session 1 =====

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;

START TRANSACTION;

SELECT * FROM StudentEnrollments WHERE student_id = 1;

-- Assume output: 2024-07-10


-- ===== Session 2 =====

START TRANSACTION;

UPDATE StudentEnrollments

SET enrollment_date = '2024-08-01'

WHERE student_id = 1;

COMMIT;


-- ===== Back to Session 1 =====

-- Still sees old snapshot

SELECT * FROM StudentEnrollments WHERE student_id = 1;

-- Output: 2024-07-10
```

COMMIT;

-- ===== Final check =====

SELECT * FROM StudentEnrollments WHERE student_id = 1;

-- Output: 2024-08-01

Output:

fieldCount	affectedRows	insertId	info	serverStatus	warningStatus	changedRows
0	0	0		10	0	0
0	0	0		10	0	0
0	3	0	Records: 3 Duplicates: 0 ...	10	0	0
0	0	0		16395	0	0
0	1	0	Rows matched: 1 Chang...	11	0	1
0	1	0	Rows matched: 1 Chang...	11	0	1
0	0	0		16395	0	0
0	1	0	Rows matched: 1 Chang...	11	0	1
0	1	0	Rows matched: 1 Chang...	11	0	1
0	0	0		16394	0	0
0	1	0		10	0	0
0	0	0		10	0	0
0	0	0		16395	0	0