

Name:	Arshdeep Singh
UID:	23BCS14221
Subject:	ADBMS
Section:	622-B

Ans 4.3:

Code:

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

INSERT INTO StudentEnrollments (student_id, student_name, course_id,
enrollment_date)
VALUES
(1, 'Ashish', 'CSE101', '2024-06-01'),
```

```
(2, 'Smaran', 'CSE102', '2024-06-01'),  
(3, 'Vaibhav', 'CSE103', '2024-06-01');
```

#Part A

```
START TRANSACTION;
```

```
-- Step 1: Lock row with student_id = 1
```

```
UPDATE StudentEnrollments
```

```
SET course_id = 'CSE201'
```

```
WHERE student_id = 1;
```

```
-- Step 2: Later tries to lock student_id = 2
```

```
UPDATE StudentEnrollments
```

```
SET course_id = 'CSE301'
```

```
WHERE student_id = 2;
```

#Part B

```
START TRANSACTION;
```

```
-- Step 1: Lock row with student_id = 2
```

```
UPDATE StudentEnrollments
```

```
SET course_id = 'CSE202'
```

```
WHERE student_id = 2;
```

```
-- Step 2: Later tries to lock student_id = 1
```

```
UPDATE StudentEnrollments
```

```
SET course_id = 'CSE302'
```

```
WHERE student_id = 1;
```

#Part B: Applying MVCC to Prevent Conflicts

--Transaction 1 (User A - Reader)

START TRANSACTION ISOLATION LEVEL REPEATABLE READ;

-- Reads snapshot data

SELECT student_id, student_name, course_id, enrollment_date

FROM StudentEnrollments

WHERE student_id = 1;

--**Transaction 2 (User B - Writer)**

START TRANSACTION;

-- Updates same row

UPDATE StudentEnrollments

SET enrollment_date = '2024-07-10'

WHERE student_id = 1;

COMMIT;

Part C: Comparing Locking vs MVCC

START TRANSACTION;

SELECT * FROM StudentEnrollments WHERE student_id = 1 FOR UPDATE;

UPDATE StudentEnrollments

```
SET course_id = 'CSE401'
WHERE student_id = 1;
START TRANSACTION;
SELECT * FROM StudentEnrollments WHERE student_id = 1;

-- This is BLOCKED until T1 commits
```

Scenario 2: MVCC (Snapshot Isolation)

Transaction 1 (Writer):

```
START TRANSACTION;
UPDATE StudentEnrollments
SET course_id = 'CSE402'
WHERE student_id = 1;
-- Not committed yet
```

Transaction 2 (Reader):

```
START TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SELECT * FROM StudentEnrollments WHERE student_id = 1;
```

Output:
(A)

20	19:00:36	UPDATE StudentEnrollments SET course_id = 'CSE301' WHERE student_id = 2	Error Code: 2013. Lost connection to MySQL server during query	30.016 sec
----	----------	---	--	------------

The error “Lost connection to MySQL server during query” occurs because Tab 1 tried to update a row that was already locked by Tab 2.

PART B:

22	19:56:03	SET SESSION TRANSACTION ISOLATION LEVEL REPEATABLE READ	0 row(s) affected	0.000 sec
23	19:56:03	START TRANSACTION	0 row(s) affected	0.000 sec
24	19:56:03	SELECT student_id, student_name, course_id, enrollment_date FROM StudentEnrollments WHERE student_id = 1 LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
25	19:56:43	SELECT student_id, student_name, course_id, enrollment_date FROM StudentEnrollments WHERE student_id = 1 LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
26	19:56:43	COMMIT	0 row(s) affected	0.000 sec

PART C:

Scenario 1

Session A (Transaction 1 - Locks row)

11	19:09:12	START TRANSACTION	0 row(s) affected	0.000 sec
4	12	19:09:12	SELECT * FROM StudentEnrollments WHERE student_id = 1 LIMIT 0, 1000 FOR UPDATE	Running... 7 / 7

12	19:09:12	SELECT * FROM StudentEnrollments WHERE student_id = 1 LIMIT 0, 1000 FOR UPDATE	Error Code: 2013. Lost connection to MySQL server during query	30.016 sec
----	----------	--	--	------------

Scenario 2

Session A (Writer)

31	19:12:58	START TRANSACTION	0 row(s) affected	0.016 sec
32	19:12:58	UPDATE StudentEnrollments SET course_id = 'CSE402' WHERE student_id = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
33	19:13:36	COMMIT	0 row(s) affected	0.000 sec

Session B (reader)

31	19:12:58	START TRANSACTION	0 row(s) affected	0.016 sec
32	19:12:58	UPDATE StudentEnrollments SET course_id = 'CSE402' WHERE student_id = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
33	19:13:36	COMMIT	0 row(s) affected	0.000 sec