```sql
-- ==================================================
-- Create Table with Constraints
-- ==================================================
CREATE TABLE StudentEnrollments (
   enrollment_id INT PRIMARY KEY,
   student_name VARCHAR(100) NOT NULL,
   course_id VARCHAR(10) NOT NULL,
   enrollment_date DATE NOT NULL,
   CONSTRAINT unique_student_course UNIQUE (student_name, course_id)
);
-- ==================================================
-- Insert Initial Data
-- ==================================================
INSERT INTO StudentEnrollments (enrollment_id, student_name, course_id, enrollment_date)
VALUES
(1, 'Ashish', 'CSE101', '2024-07-01'),
(2, 'Smaran', 'CSE102', '2024-07-01'),
(3, 'Vaibhav', 'CSE101', '2024-07-01');
SELECT * FROM StudentEnrollments;
-- ==================================================
-- Part A: Prevent Duplicate Enrollments
-- ============================================

-- User A transaction

START TRANSACTION;

INSERT INTO StudentEnrollments (enrollment_id, student_name,
course_id, enrollment_date)

VALUES (4, 'Ashish', 'CSE101', '2024-07-02');

-- This will FAIL (duplicate pair)

-- Rollback since duplicate attempt

ROLLBACK;

-- Only unique combinations exist

SELECT * FROM StudentEnrollments;

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

-- Part B: Row Locking with SELECT FOR UPDATE

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

-- User A

START TRANSACTION;

SELECT * FROM StudentEnrollments

WHERE student_name = 'Ashish' AND course_id = 'CSE101'

FOR UPDATE;

 -- Locks this row until commit/rollback

-- (At this point, User A keeps transaction open...)


-- User B (in another session) tries to update:
```

```sql
-- This query will BLOCK until User A commits/rolls back
UPDATE StudentEnrollments
SET enrollment_date = '2024-07-05'
WHERE student_name = 'Ashish' AND course_id = 'CSE101';

-- Once User A executes COMMIT or ROLLBACK,
-- User B’s update will proceed.

-- ==============================================
-- Part C: Demonstrate Locking Preserves Consistency
-- ==============================================
-- Suppose both User A and User B try to update the same row.
-- User A
START TRANSACTION;
SELECT * FROM StudentEnrollments
WHERE student_name = 'Ashish' AND course_id = 'CSE101'
FOR UPDATE;
-- Locks the row
UPDATE StudentEnrollments
SET enrollment_date = '2024-07-10'
WHERE student_name = 'Ashish' AND course_id = 'CSE101';
COMMIT;
 -- Unlocks row
-- User B (runs after A commits)
START TRANSACTION;
SELECT * FROM StudentEnrollments
WHERE student_name = 'Ashish' AND course_id = 'CSE101'
FOR UPDATE;
-- Now it can lock safely

UPDATE StudentEnrollments
SET enrollment_date = '2024-07-15'
WHERE student_name = 'Ashish' AND course_id = 'CSE101';
COMMIT;
-- Final state: enrollment_date = 2024-07-15 (last committed
update)
SELECT * FROM StudentEnrollments;
```