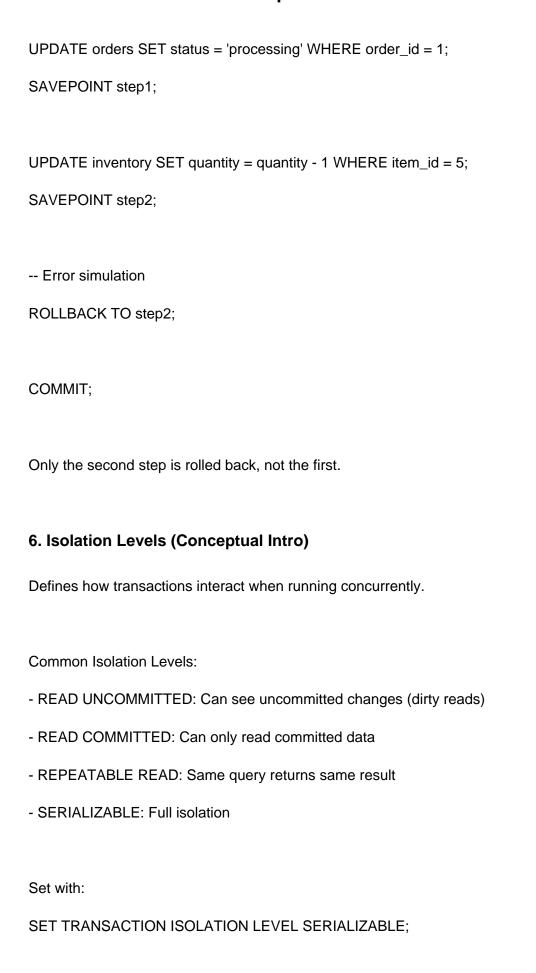
1. Why Use Transactions? Use Case: - Transfer money - Update multiple related tables - Ensure either all steps succeed or none happen Without transactions, partial updates could leave your database in an inconsistent state. 2. Real-World Scenario: Bank Transfer START TRANSACTION; -- Deduct from sender UPDATE accounts SET balance = balance - 1000 WHERE account_id = 101; -- Add to receiver UPDATE accounts SET balance = balance + 1000 WHERE account_id = 202; -- Log the transfer INSERT INTO transactions (from_id, to_id, amount) VALUES (101, 202, 1000);

All three steps are needed. If any fails, we ROLLBACK.

COMMIT;

3. What Happens Without COMMIT? START TRANSACTION; UPDATE products SET stock = stock - 1 WHERE id = 5; -- COMMIT is not called On disconnect, the change is lost. Always end with COMMIT or ROLLBACK. 4. Handling Errors with ROLLBACK START TRANSACTION; UPDATE accounts SET balance = balance - 1000 WHERE account_id = 1; -- Simulated error INSERT INTO trnasactions (from_id, to_id, amount) VALUES (1, 2, 1000); ROLLBACK; The previous update is undone because of the error. 5. Using SAVEPOINT and ROLLBACK TO START TRANSACTION;



Practice Exercises

- 1. Create a transaction that updates both customer and order tables.
- 2. Use SAVEPOINT to rollback part of a shopping cart checkout.
- 3. Simulate an error (e.g., product out of stock) and recover with ROLLBACK.