DBMS Practical File

Practical – Implementation of Strict 2PL and Rigorous 2PL

Name: Anshuman Diwakar
Course: B.Tech (CSE)
Subject: Database Management Systems
Instructor: Karan Gupta
Date: September 9, 2025

Aim

To demonstrate **Strict 2PL** and **Rigorous 2PL** protocols using SQL transactions.

Theory

- **Strict 2PL**:
  - Shared (S) locks may be released after use.
  - Exclusive (X) locks are held until commit/abort.
  - Ensures conflict-serializable and cascadeless schedules.
- **Rigorous 2PL**:
  - Both shared and exclusive locks are held until commit.
  - Ensures conflict-serializable, cascadeless, and strict schedules.

Database Setup

```sql
CREATE TABLE Accounts (
    acc_no INT PRIMARY KEY,
    balance INT
);

INSERT INTO Accounts VALUES (1, 1000);
INSERT INTO Accounts VALUES (2, 2000);
```

Transaction 1 (T1)

```sql
BEGIN TRANSACTION;

-- T1 reads Account 1
SELECT balance FROM Accounts WHERE acc_no = 1;

-- T1 updates Account 2
UPDATE Accounts SET balance = balance + 500 WHERE acc_no = 2;

-- Commit releases locks
COMMIT;
```

Transaction 2 (T2)

```
BEGIN TRANSACTION;

-- T2 updates Account 1
UPDATE Accounts SET balance = balance - 200 WHERE acc_no = 1;

-- T2 reads Account 2
SELECT balance FROM Accounts WHERE acc_no = 2;

COMMIT;
```

Schedule under Strict 2PL

```
T1: S-lock(A1), read(A1), release S-lock(A1)
T2: X-lock(A1), write(A1) (waits until T1 releases)
T1: X-lock(A2), write(A2), commit (releases X-lock)
T2: read(A2), commit
```

Schedule under Rigorous 2PL

```
T1: S-lock(A1), read(A1) (lock not released)
T2: waits for X-lock(A1)
T1: X-lock(A2), write(A2)
T1: commit (all locks released)
T2: X-lock(A1), write(A1), S-lock(A2), read(A2), commit
```

Conclusion

- **Strict 2PL** holds exclusive locks till commit but may release shared locks earlier.
- **Rigorous 2PL** holds *all locks* (S and X) till commit.
- Rigorous 2PL gives the highest degree of isolation and avoids cascading aborts completely.