NAME- Roushan kumar; UID-23BCC70013; SUB-ADBMS

EXP-10

➢ **AIM:** To demonstrate the ACID properties of database transactions (especially Atomicity and Consistency) by performing multiple inserts into the FeePayments table, handling failures using ROLLBACK, and ensuring the database remains in a consistent state.

➢ **THEORY:**

- Transactions in DBMS: A transaction is a sequence of SQL operations treated as a single unit. Either all operations succeed (COMMIT) or none (ROLLBACK).
- ACID Properties:
- Atomicity: Ensures all operations in a transaction are completed, or none are.
- Consistency: Database moves from one valid state to another.
- Isolation: Transactions do not interfere with each other.
- Durability: Once committed, changes are permanent.
- Use Case of Transactions:
- Insert multiple fee payment records.
- If any insert fails (e.g., duplicate payment_id or invalid data), the entire transaction is rolled back.
- SQL Commands Used:
- START TRANSACTION / BEGIN: Begin a transaction
- COMMIT: Save changes permanently
- ROLLBACK: Undo changes due to failure

➢ **CODES:**

- Part A: Insert Multiple Fee Payments (Successful Transaction)

```
-- Begin transaction
START TRANSACTION;

-- Insert multiple valid records
INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
VALUES (1, 'Ashish', 5000.00, '2024-06-01');

INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
VALUES (2, 'Smaran', 4500.00, '2024-06-02');

INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
VALUES (3, 'Vaibhav', 5500.00, '2024-06-03');

-- Commit transaction
COMMIT;

-- Verify inserted records
SELECT * FROM FeePayments;
DELIMITER;
```

- Part B: Failed Transaction with ROLLBACK
  ```
  -- Begin transaction
  START TRANSACTION;

  -- First insert valid
  INSERT INTO FeePayments (payment_id,
  student_name, amount, payment_date)
  VALUES (4, 'Kiran', 4800.00, '2024-06-04');

  -- Second insert invalid (duplicate ID)
  INSERT INTO FeePayments (payment_id,
  student_name, amount, payment_date)
  VALUES (1, 'Ashish', 5000.00, '2024-06-01');

  -- Transaction fails, rollback
  ROLLBACK;

  -- Verify table remains unchanged
  SELECT * FROM FeePayments;
  ```
- Part C: Partial Failure Demonstration
  ```
  START TRANSACTION;

  -- First insert valid
  INSERT INTO FeePayments (payment_id,
  student_name, amount, payment_date)
  VALUES (5, 'Rohit', 5000.00, '2024-06-05');

  -- Second insert invalid (NULL student_name)
  INSERT INTO FeePayments (payment_id,
  student_name, amount, payment_date)
  VALUES (6, NULL, 4700.00, '2024-06-06');
  ```

```
ROLLBACK;

SELECT * FROM FeePayments;
```

➢ **OUTPUTS:**

payment_id	student_name	amount	payment_date
1	Ashish	5000.00	2024-06-01
2	Smaran	4500.00	2024-06-02
3	Vaibhav	5500.00	2024-06-03

➢ **LEARNING OUTCOMES:**

1. Learned how to use **transactions** in SQL with START TRANSACTION, COMMIT, and ROLLBACK.
2. Understood **Atomicity**, ensuring all operations in a transaction succeed or none are applied.
3. Observed **Consistency**, maintaining valid database state even when transactions fail.
4. Gained experience handling **transaction failures** caused by constraint violations or duplicates.
5. Practiced **ACID principles** in action, reinforcing database reliability and integrity.