

NAME- Piyush Kakkar; UID-23BCC70018; SUB-
ADBMS

EXP-10

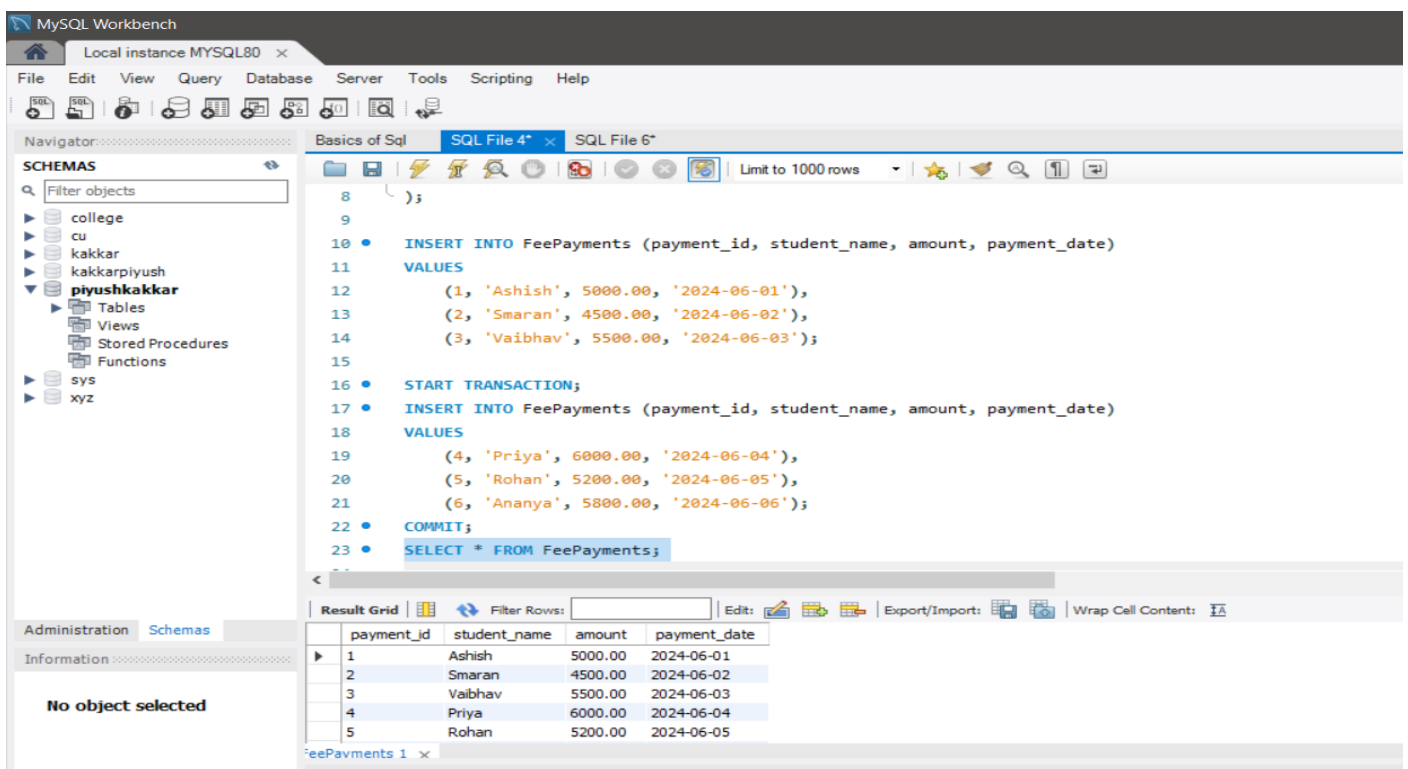
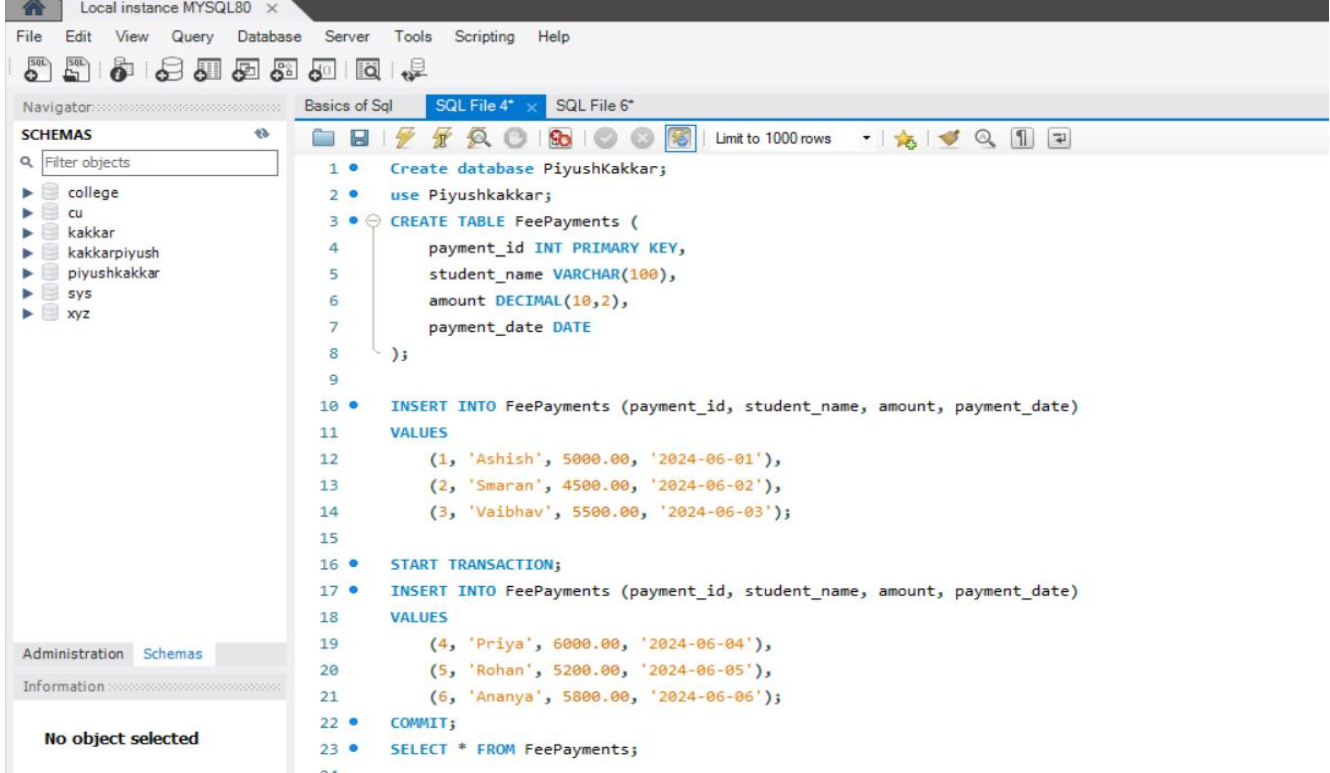
➤ **AIM:** To demonstrate the ACID properties of database transactions (especially Atomicity and Consistency) by performing multiple inserts into the FeePayments table, handling failures using ROLLBACK, and ensuring the database remains in a consistent state.

➤ **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)



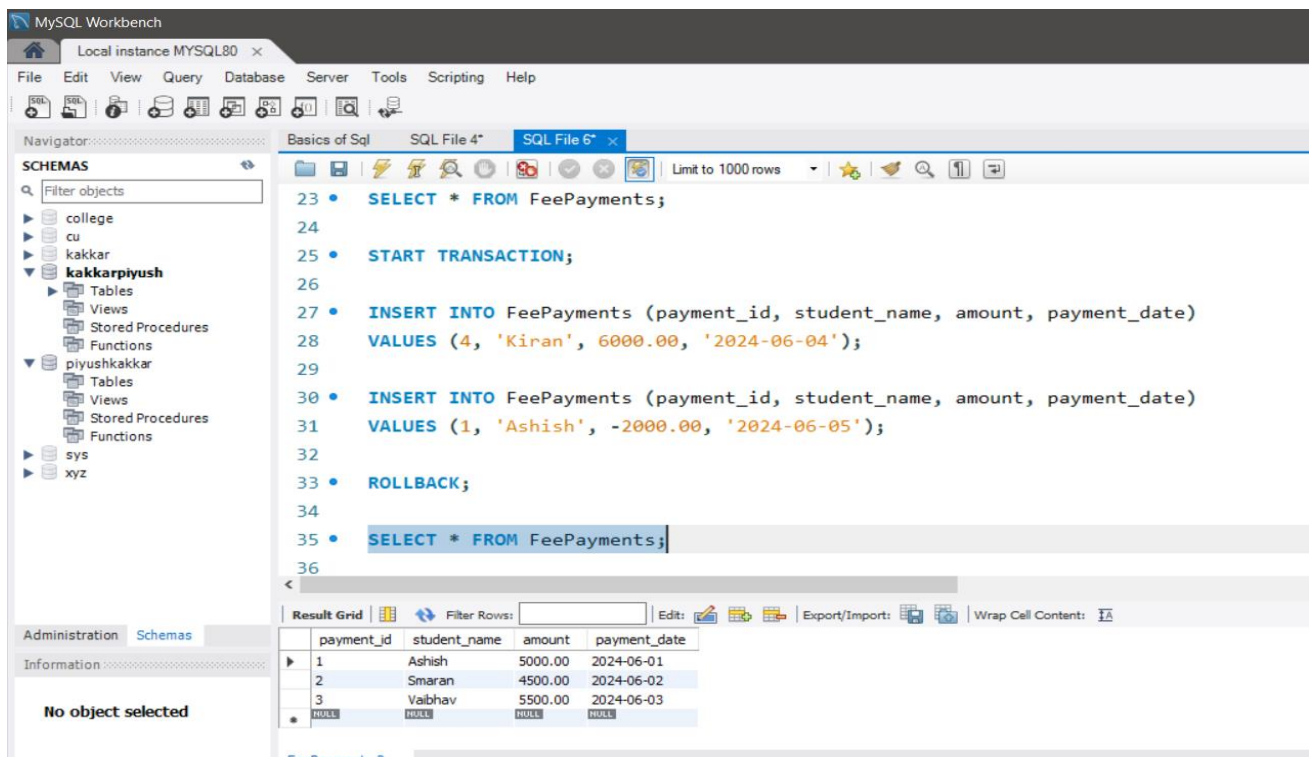
- Part B: Failed Transaction with ROLLBACK

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with a filter 'Filter objects'. The 'college' database is selected, showing its contents: 'cu', 'kakkar', 'kakkarpiyush', 'piyushkakkar' (containing 'Tables', 'Views', 'Stored Procedures', and 'Functions'), 'sys', and 'xyz'. The main editor window shows a SQL script in 'SQL File 6*'. The script creates a database 'kakkarpiyush', uses it, and creates a table 'FeePayments' with columns 'payment_id' (INT PRIMARY KEY), 'student_name' (VARCHAR(100)), 'amount' (DECIMAL(10,2) with a CHECK constraint for positive values), and 'payment_date' (DATE). It then starts a transaction, inserts three rows of data, commits the transaction, and selects all data from the 'FeePayments' table.

```
1 • create database kakkarpiyush;
2 • use kakkarpiyush;
3 • CREATE TABLE FeePayments (
4 •     payment_id INT PRIMARY KEY,
5 •     student_name VARCHAR(100),
6 •     amount DECIMAL(10,2) CHECK (amount > 0), -- Ensures positive amounts
7 •     payment_date DATE
8 • );
9
10 • START TRANSACTION;
11
12 • INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
13 • VALUES (1, 'Ashish', 5000.00, '2024-06-01');
14
15 • INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
16 • VALUES (2, 'Smaran', 4500.00, '2024-06-02');
17
18 • INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
19 • VALUES (3, 'Vaibhav', 5500.00, '2024-06-03');
20
21 • COMMIT;
22
23 • SELECT * FROM FeePayments;
24
25 • START TRANSACTION;
26
```

The screenshot shows the MySQL Workbench interface with the same schema tree as the previous image. The main editor window shows a SQL script in 'SQL File 6*'. The script continues from the previous one, inserting a fourth row, committing, and selecting all data. It then starts a new transaction, inserts a fifth row with a negative amount, and rolls back the transaction. Finally, it selects all data from the 'FeePayments' table.

```
18 • INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
19 • VALUES (3, 'Vaibhav', 5500.00, '2024-06-03');
20
21 • COMMIT;
22
23 • SELECT * FROM FeePayments;
24
25 • START TRANSACTION;
26
27 • INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
28 • VALUES (4, 'Kiran', 6000.00, '2024-06-04');
29
30 • INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
31 • VALUES (1, 'Ashish', -2000.00, '2024-06-05');
32
33 • ROLLBACK;
34
35 • SELECT * FROM FeePayments;
36
```



- 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.