Experiment 10

Subject: ADBMS

Subject Code:23CSP-333

Date: 26th September 2025

Name: Aditya Sivam Kashyap

UID : 23BCC70036

Section: 23BCC-1

1. Aim:

Transactions and Concurrency Control

1. Part A: To demonstrate a successful transaction where multiple INSERT statements are treated as a single, atomic unit of work using BEGIN and COMMIT.
2. Part B: To simulate a transaction failure due to a constraint violation and use ROLLBACK to explicitly undo all changes, preserving atomicity and consistency.
3. Part C: To show how a transaction automatically enters a failed state after a single invalid statement, causing a COMMIT to fail and ensuring a consistent state.
4. Part D: To provide a comprehensive flow that verifies all four ACID properties—Atomicity, Consistency, Isolation, and Durability—in a series of operations.

2. Theory:

A database transaction is a sequence of operations performed as a single logical unit of work. For a transaction to be reliable, it must adhere to the four ACID properties. Atomicity ensures that the transaction is "all or nothing"—either all operations succeed, or none do. Consistency guarantees that the database remains in a valid state before and after the transaction. Isolation ensures that concurrent transactions do not interfere with one another. Finally, Durability guarantees that once a transaction is committed, its changes are permanent and will survive any subsequent system failure.

3. SQL Queries:

1. Part A: Successful Atomic Transaction
   - Create the initial table and insert multiple records in one transaction.

```
CREATE TABLE FeePayments (
    payment_id INT PRIMARY KEY,
    student_name VARCHAR(100),
    amount DECIMAL(10,2),
    payment_date DATE
);
```

```
BEGIN;


INSERT INTO FeePayments VALUES (1, 'Ashish', 5000.00,
'2024-06-01');
INSERT INTO FeePayments VALUES (2, 'Smaran', 4500.00,
'2024-06-02');
INSERT INTO FeePayments VALUES (3, 'Vaibhav', 5500.00,
'2024-06-03');

COMMIT;

SELECT * FROM feepayments;
```

2. Part B: Explicit ROLLBACK after Failure
   - Add a CHECK constraint to the table.

```
ALTER TABLE FeePayments
ADD CONSTRAINT chk_positive_amount CHECK (amount > 0);
```

   - Attempt a transaction with an invalid record and explicitly roll it back.

```
BEGIN;


INSERT INTO FeePayments (payment_id, student_name,
amount, payment_date)
VALUES (4, 'Kiran', 6000.00, '2024-06-04');

INSERT INTO FeePayments (payment_id, student_name,
amount, payment_date)
VALUES (5, 'Invalid', -2000.00, '2024-06-05');

ROLLBACK;

SELECT * FROM feepayments;
```

3. Part C: Automatic Rollback on Partial Failure
    - Add a NOT NULL constraint to the table.

```
ALTER TABLE FeePayments
ALTER COLUMN student_name SET NOT NULL;
```

    - Attempt a transaction where an invalid statement causes the COMMIT to fail.

```
BEGIN;

INSERT INTO FeePayments (payment_id, student_name,
amount, payment_date)
VALUES (4, 'Rohan', 6000.00, '2024-06-05');

INSERT INTO FeePayments (payment_id, student_name,
amount, payment_date)
VALUES (5, NULL, 5000.00, '2024-06-06');

COMMIT;


SELECT * FROM feepayments;
```

4. Part D: Verifying All ACID Properties
    - Run a sequence of transactions to demonstrate A, C, I, and D.

```
BEGIN;
INSERT INTO FeePayments (payment_id, student_name,
amount, payment_date)
VALUES (6, 'Anjali', 7000.00, '2024-06-08');

INSERT INTO FeePayments (payment_id, student_name,
amount, payment_date)
VALUES (1, 'Duplicate', 9999.00, '2024-06-09');
COMMIT;

BEGIN;
INSERT INTO FeePayments (payment_id, student_name,
amount, payment_date)
VALUES (7, 'Kavita', 6500.00, '2024-06-10');
```

```
COMMIT;

BEGIN;
UPDATE FeePayments
SET amount = 5250.00
WHERE payment_id = 1;
COMMIT;


SELECT * FROM FeePayments;
```

4. Result:

Part-A:

Query | Query History

```
1  create table FeePayments(payment_id int primary key,
2  student_name varchar(100),
3  amount decimal(10,2),payment_date date);
4
5
6  begin;
7  insert into FeePayments values(1,'Ashish',
8  5000.00,'2024-06-01');
9  insert into FeePayments values (2,'Smaran',4500.00,
10 '2024-06-02');
11 insert into FeePayments values(3,'Vaibhav',5500.00,
12 '2024-06-03');
13 commit;
14
15 select * from feepayments;
```

Data Output | Messages | Notifications

	payment_id [PK] integer	student_name character varying (100)	amount numeric (10,2)	payment_date date
1	1	Ashish	5000.00	2024-06-01
2	2	Smaran	4500.00	2024-06-02
3	3	Vaibhav	5500.00	2024-06-03

Part-B:

```
BEGIN;


INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
VALUES (4, 'Kiran', 6000.00, '2024-06-04');


INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
VALUES (1, 'Ashish', -2000.00, '2024-06-05');

ROLLBACK;

select * from feepayments;
```

Data Output | Messages | Notifications

	payment_id [PK] integer	student_name character varying (100)	amount numeric (10,2)	payment_date date
1	1	Ashish	5000.00	2024-06-01
2	2	Smaran	4500.00	2024-06-02
3	3	Vaibhav	5500.00	2024-06-03

Part-C:

Query | Query History

```
BEGIN;


INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
VALUES (4, 'Rohan', 6000.00, '2024-06-05');

INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
VALUES (5, NULL, 5000.00, '2024-06-06');

COMMIT;


select * from feepayments;
```

Data Output | Messages | Notifications

Showing rows

	payment_id [PK] integer	student_name character varying (100)	amount numeric (10,2)	payment_date date
1	1	Ashish	5000.00	2024-06-01
2	2	Smaran	4500.00	2024-06-02
3	3	Vaibhav	5500.00	2024-06-03

Part-D:

Query | Query History

```
82
83  INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
84  VALUES (6, 'Anjali', 7000.00, '2024-06-08');
85
86  INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
87  VALUES (1, 'Duplicate', 9999.00, '2024-06-09');
88
89  COMMIT;
90
91  BEGIN;
92
93  INSERT INTO FeePayments (payment_id, student_name, amount, payment_date)
94  VALUES (7, 'Kavita', 6500.00, '2024-06-10');
95
96  COMMIT;
97
98  BEGIN;
99
100 UPDATE FeePayments
101 SET amount = 5250.00
```

Data Output | Messages | Notifications

	payment_id [PK] integer	student_name character varying (100)	amount numeric (10,2)	payment_date date
1	2	Smaran	4500.00	2024-06-02
2	3	Vaibhav	5500.00	2024-06-03
3	7	Kavita	6500.00	2024-06-10
4	1	Ashish	5250.00	2024-06-01