NAME-AVIRAL SRIVASTAVA; UID-23BCC70028; SEC-23BCC-1(A)

EXPERIMENT-03

• <u>AIM: -</u> To design a normalized relational database schema using SQL to manage students, courses, and their enrollments with proper integrity constraints; perform data insertion, transactions using SAVEPOINTS and ROLLBACK, simulate error handling with faulty records, and retrieve meaningful reports using joins to display students' enrollment and grade details.

• THEORY: -

1. **Relational Database Design & Normalization**: -A relational database stores data in structured tables (relations) where each table represents an entity (e.g., Students, Courses). To avoid redundancy and maintain data integrity, databases are normalized — i.e., broken down into multiple related tables that satisfy normal forms (up to 3NF in this case). The key entities in this schema are:

Students (student id, name, dob)

Courses (course_id, title)

Enrollments (enroll_id, student_id, course_id, grade) Here, Enrollments act as a junction table to manage many-to-many relationships between Students and Courses.

2. Primary and Foreign Key Constraints

Primary Keys (PK) uniquely identify each record in a table (e.g., student_id, course_id, enroll_id).

Foreign Keys (FK) maintain referential integrity, ensuring that data in one table corresponds to valid data in another (e.g., student_id in Enrollments must exist in Students).

3. SQL Transactions

A transaction is a sequence of one or more SQL operations treated as a single logical unit of work. Transactions are governed by ACID properties (Atomicity, Consistency, Isolation, Durability).

Using START TRANSACTION, SAVEPOINT, and ROLLBACK, we can:

Perform operations safely,

Set a SAVEPOINT to mark a specific state,

ROLLBACK TO SAVEPOINT if a part of the transaction fails (e.g., duplicate primary key insertion), preventing data corruption.

4. Error Simulation and Recovery

To test database robustness, simulated errors such as inserting duplicate enroll_id values or invalid student id (violating FK constraints) help demonstrate:

How constraints enforce data validity,

How ROLLBACK TO SAVEPOINT can undo partial changes, leaving valid records intact.

5. Data Retrieval with JOINs

Using JOIN operations, we can combine related data across multiple tables. For example:

A JOIN between Students, Courses, and Enrollments allows us to fetch each student's name, course title, and grade. This is essential for generating real-world reports and dashboards.

6. **Real-World Applications:** This schema is similar to those used in:

College/University management systems,

Online learning platforms (e.g., Coursera, edX),

Training tracking systems in corporations.

• CODES: -

```
-- Create Students table

CREATE TABLE Students (
    student_id INT PRIMARY KEY,
    name VARCHAR(100),
    dob DATE
);

-- Create Courses table

CREATE TABLE Courses (
    course_id INT PRIMARY KEY,
    title VARCHAR(100)
);

-- Create Enrollments table with foreign key constraints

CREATE TABLE Enrollments (
```

```
enroll id INT PRIMARY KEY,
  student id INT,
  course id INT,
  grade VARCHAR(2),
  FOREIGN KEY (student id) REFERENCES Students(student id),
  FOREIGN KEY (course id) REFERENCES Courses (course id)
);
-- Insert into Students
INSERT INTO Students VALUES
(1, 'Ashish', '2002-03-14'),
(2, 'Smaran', '2001-08-22'),
(3, 'Vaibhav', '2003-01-05');
-- Insert into Courses
INSERT INTO Courses VALUES
(101, 'DBMS'),
(102, 'Operating Systems'),
(103, 'Computer Networks');
-- Begin Transaction
START TRANSACTION;
-- First Enrollment: Ashish into DBMS
INSERT INTO Enrollments VALUES (1, 1, 101, 'A');
-- Create SAVEPOINT
SAVEPOINT sp1;
-- Second Enrollment: Ashish into Operating Systems
INSERT INTO Enrollments VALUES (2, 1, 102, 'B+');
-- Third Enrollment: Ashish into Computer Networks
INSERT INTO Enrollments VALUES (3, 1, 103, 'A');
-- Simulate an error: Duplicate enroll id
-- This will fail because enroll id 2 already exists
INSERT INTO Enrollments VALUES (2, 1, 101, 'C');
```

-- Rollback to savepoint to undo the faulty insert

ROLLBACK TO SAVEPOINT sp1;

- -- Commit the transaction COMMIT;
- -- Final query to display student name, course title, and grade SELECT

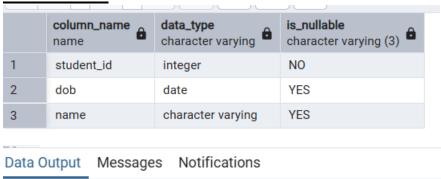
s.name AS student_name, c.title AS course_title, e.grade

FROM Enrollments e

JOIN Students s ON e.student id = s.student id

JOIN Courses c ON e.course id = c.course id;

• OUTPUTS: -





• LEARNING OUTCOMES: -

- ➤ Understanding of Database Schema Design: Ability to create normalized relational tables using SQL.
- ➤ Understanding how to define **Primary Keys** and **Foreign Keys** to enforce entity integrity and referential integrity.
- ➤ Application of Data Integrity Constraints: Hands-on experience in applying constraints to prevent invalid or duplicate data entries (e.g., primary key violations, foreign key constraints).
- > Transaction Management in SQL:Practical understanding of ACID properties in transactions.

- ➤ Use of START TRANSACTION, SAVEPOINT, ROLLBACK TO SAVEPOINT, and COMMIT for safe and controlled data operations.
- ➤ Error Simulation and Recovery: Ability to simulate errors (e.g., inserting duplicate keys) and recover using ROLLBACK techniques without losing valid data.