Experiment 3

Subject: ADBMS | Name: Jayanaath S

Subject Code:23CSP-333 | UID : 23BCC70022

Date: 22th August 2025 | Section: 23BCC-1

➢ Aim:

Transaction Management and Save point Simulation in Student Enrollments

1. To create 3 tables – Students, Courses, Enrollments
2. To insert values into Students, Courses and Enrollments, and display the tables
3. To start transaction and insert first Enrollment
4. To set Savepoint before second Enrollment
5. Display Student Enrollments with course and grade details

➢ Theory:

TCL in SQL is used to manage transactions in a database. A **transaction** is a sequence of operations performed as a single logical unit of work. TCL ensures the database remains consistent and reliable.

The main TCL commands are:

- **COMMIT** – Saves all changes made by the current transaction permanently in the database.
- **ROLLBACK** – Undoes changes made by the current transaction, returning the database to the last committed state.
- **SAVEPOINT** – Sets a checkpoint within a transaction, allowing partial rollback to that point instead of the whole transaction.
- **SET TRANSACTION** – Defines properties of a transaction, such as read/write access.

TCL helps maintain **data integrity** by controlling how and when changes are applied or undone.

➢ SQL Queries:

1. To create 3 tables – Students, Courses, Enrollments:

```
create table students(student_id int primary key, name
varchar(100),dob date);
create table courses(course_id int primary key,title
varchar(100));
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));

desc students;
desc courses;
desc enrollments;
```

2. To insert values into Students, Courses and Enrollments, and display the tables:

```
insert into students values(1, 'Ashish', '2002-03-
14'),(2, 'Smaran', '2001-08-22'),(3, 'Vaibhav', '2003-01-
05');
insert into courses values(101, 'DBMS'),(102, 'Operating
Systems'),(103, 'Computer Networks');
insert into enrollments values(1, 1, 101, 'A'),(2, 1,
102, 'B+');

select * from students;
select * from courses;
select * from enrollments;
```

3. To start transaction and insert first Enrollment:

```
begin;
insert into enrollments values (3,1,103,'A');
select * from enrollments;
```

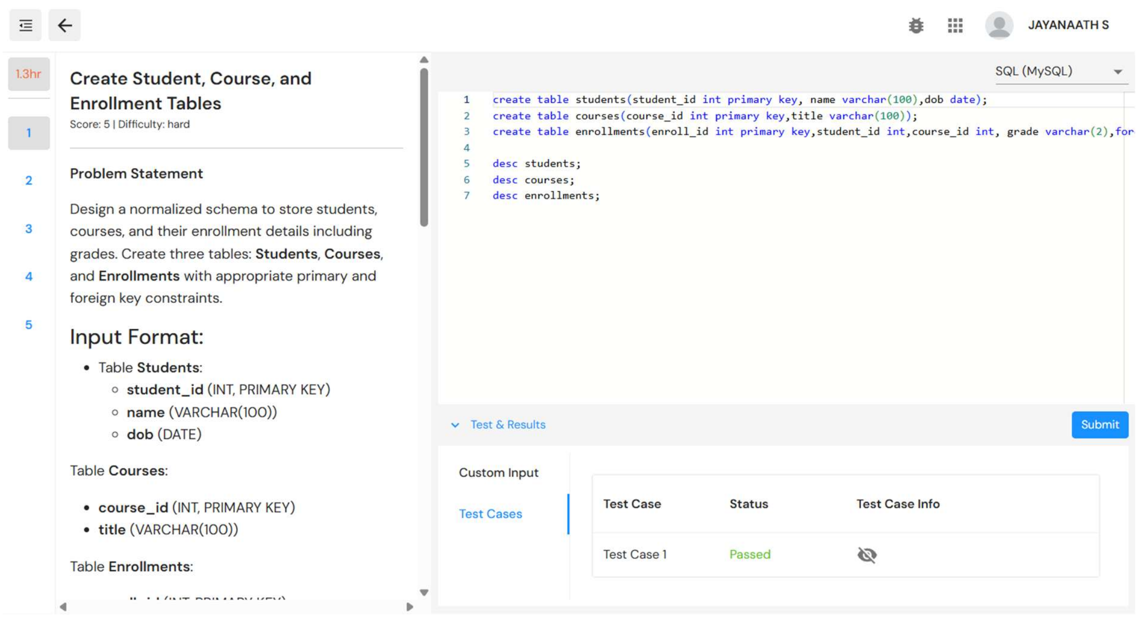
4. To set Savepoint before second Enrollment

```
savepoint before_faulty;
select * from enrollments;
commit;
```

5. Display Student Enrollments with course and grade details

```
select t1.name as student_name,t2.title as
course_title,t3.grade from students t1
inner join enrollments t3 on t1.student_id=t3.student_id
inner join courses t2 on t2.course_id=t3.course_id;
```

➢ <u>Result:</u>



Create Student, Course, and Enrollment Tables

Score: 5 | Difficulty: hard

Problem Statement

Design a normalized schema to store students, courses, and their enrollment details including grades. Create three tables: **Students**, **Courses**, and **Enrollments** with appropriate primary and foreign key constraints.

Input Format:

- Table **Students**:
  - **student_id** (INT, PRIMARY KEY)
  - **name** (VARCHAR(100))
  - **dob** (DATE)

Table **Courses**:

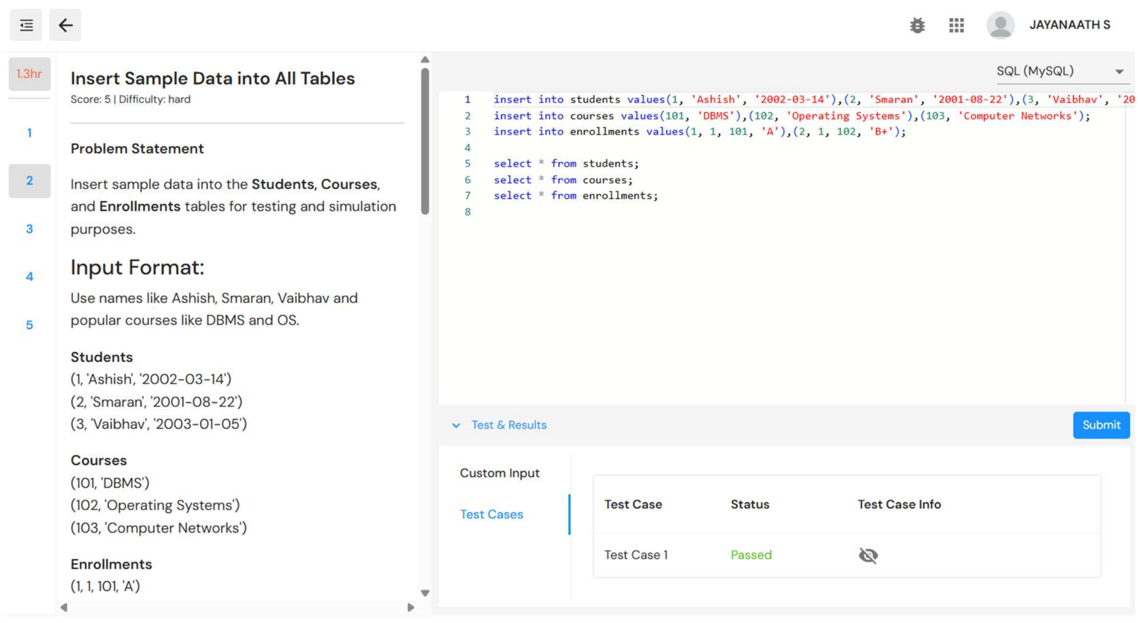
- **course_id** (INT, PRIMARY KEY)
- **title** (VARCHAR(100))

Table **Enrollments**:

```
create table students(student_id int primary key, name varchar(100),dob date);
create table courses(course_id int primary key,title varchar(100));
create table enrollments(enroll_id int primary key,student_id int,course_id int, grade varchar(2),for

desc students;
desc courses;
desc enrollments;
```

Test Case	Status	Test Case Info
Test Case 1	Passed	



Insert Sample Data into All Tables

Score: 5 | Difficulty: hard

Problem Statement

Insert sample data into the **Students, Courses**, and **Enrollments** tables for testing and simulation purposes.

Input Format:

Use names like Ashish, Smaran, Vaibhav and popular courses like DBMS and OS.

Students
(1, 'Ashish', '2002-03-14')
(2, 'Smaran', '2001-08-22')
(3, 'Vaibhav', '2003-01-05')

Courses
(101, 'DBMS')
(102, 'Operating Systems')
(103, 'Computer Networks')

Enrollments
(1, 1, 101, 'A')

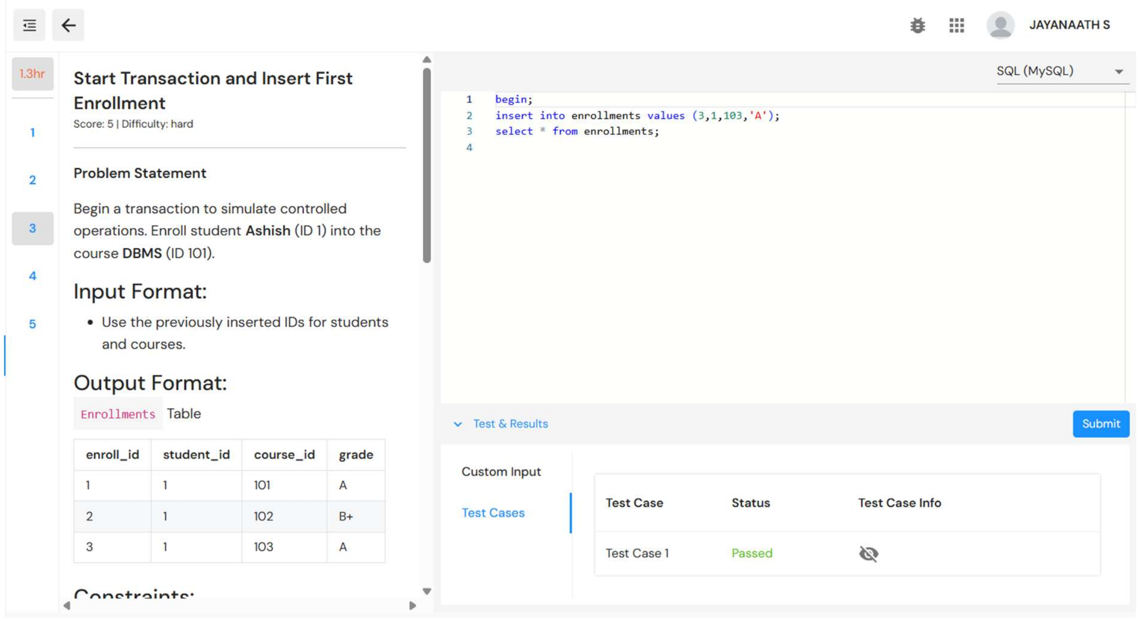
SQL (MySQL)

```sql
insert into students values(1, 'Ashish', '2002-03-14'),(2, 'Smaran', '2001-08-22'),(3, 'Vaibhav', '20
insert into courses values(101, 'DBMS'),(102, 'Operating Systems'),(103, 'Computer Networks');
insert into enrollments values(1, 1, 101, 'A'),(2, 1, 102, 'B+');

select * from students;
select * from courses;
select * from enrollments;

```

Test Case	Status	Test Case Info
Test Case 1	Passed	



Start Transaction and Insert First Enrollment

Score: 5 | Difficulty: hard

Problem Statement

Begin a transaction to simulate controlled operations. Enroll student **Ashish** (ID 1) into the course **DBMS** (ID 101).

Input Format:

- Use the previously inserted IDs for students and courses.

Output Format:

`Enrollments` Table

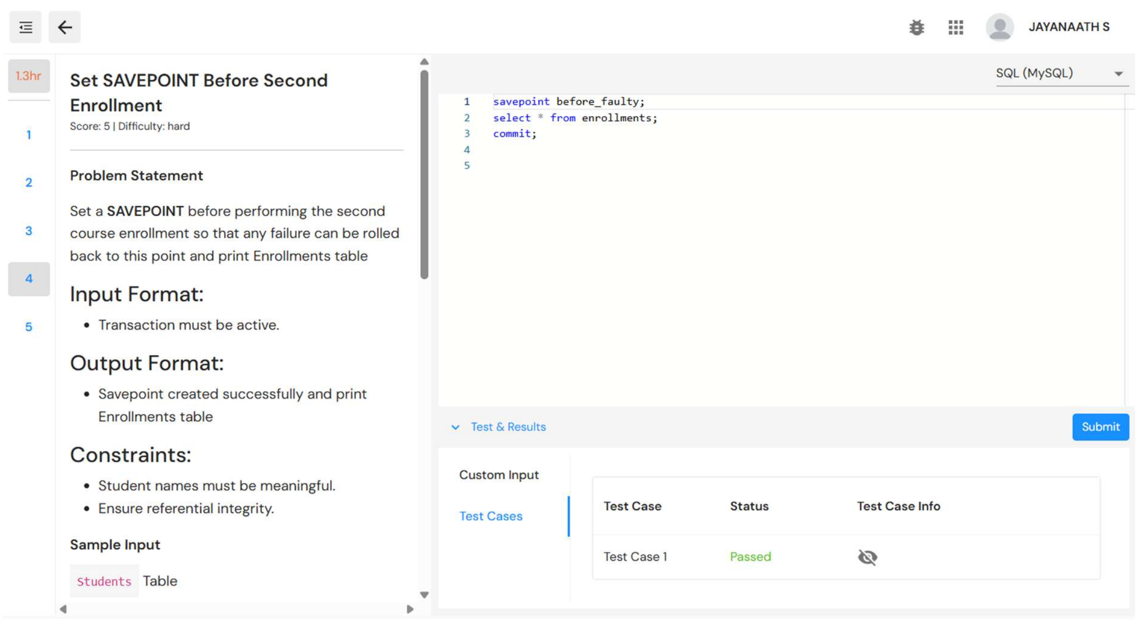
enroll_id	student_id	course_id	grade
1	1	101	A
2	1	102	B+
3	1	103	A

SQL (MySQL)

```sql
begin;
insert into enrollments values (3,1,103,'A');
select * from enrollments;

```

Test Case	Status	Test Case Info
Test Case 1	Passed	



Set SAVEPOINT Before Second Enrollment

Score: 5 | Difficulty: hard

Problem Statement

Set a **SAVEPOINT** before performing the second course enrollment so that any failure can be rolled back to this point and print Enrollments table

Input Format:

- Transaction must be active.

Output Format:

- Savepoint created successfully and print Enrollments table

Constraints:

- Student names must be meaningful.
- Ensure referential integrity.

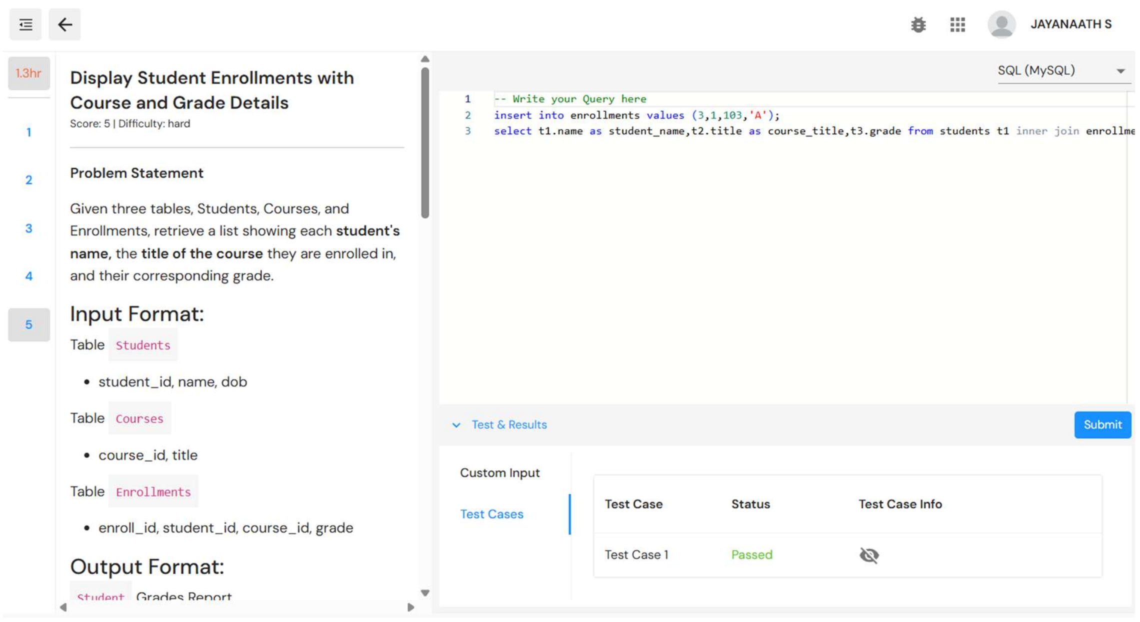
Sample Input

`Students` Table

SQL (MySQL)

```
savepoint before_faulty;
select * from enrollments;
commit;
```

Test Case	Status	Test Case Info
Test Case 1	Passed	



Display Student Enrollments with Course and Grade Details

Score: 5 | Difficulty: hard

Problem Statement

Given three tables, Students, Courses, and Enrollments, retrieve a list showing each **student's name**, the **title of the course** they are enrolled in, and their corresponding grade.

Input Format:

Table `Students`

- student_id, name, dob

Table `Courses`

- course_id, title

Table `Enrollments`

- enroll_id, student_id, course_id, grade

Output Format:

`Student` Grades Report

SQL (MySQL)

```
-- Write your Query here
insert into enrollments values (3,1,103,'A');
select t1.name as student_name,t2.title as course_title,t3.grade from students t1 inner join enrollme
```

Test Case	Status	Test Case Info
Test Case 1	Passed	