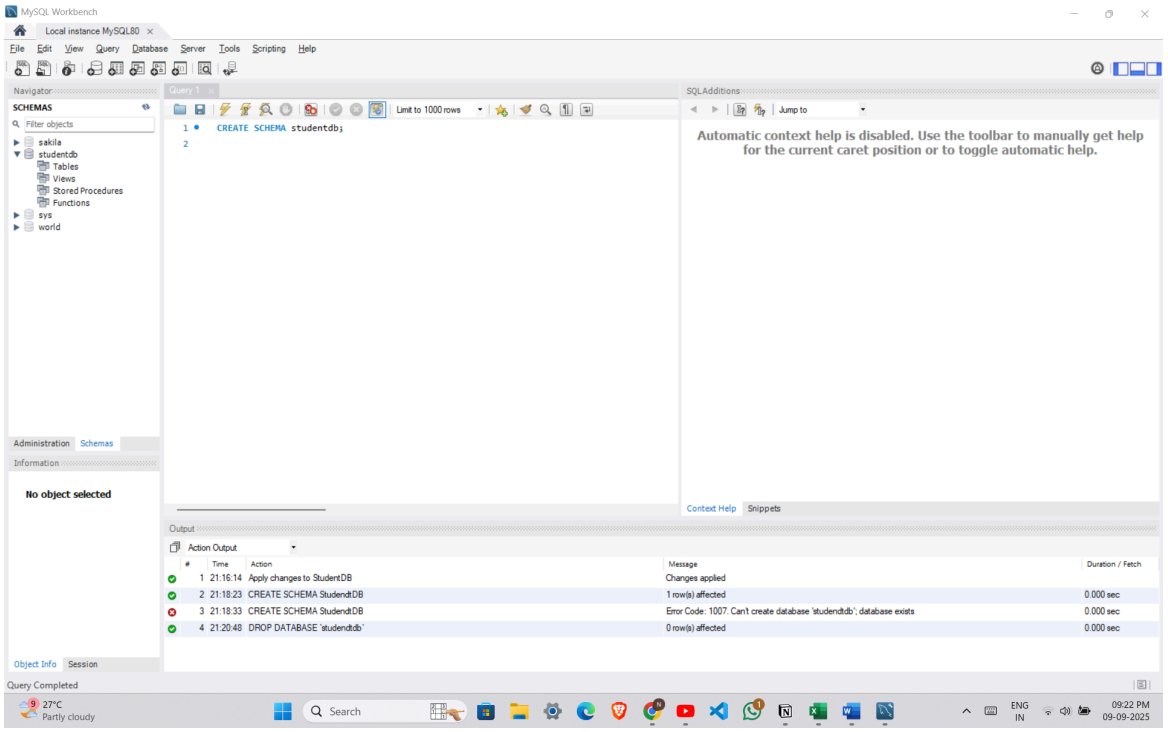
WEEK-1 ASSIGNMENTS

▼ Assignment 1

```
CREATE SCHEMA studentdb;
```



assignment1.sql

▼ Assignment 2

- Select the Database

```
USE studentdb;
```

- Create Students Table

```
CREATE TABLE students(
student_id INT PRIMARY KEY AUTO_INCREMENT,
name VARCHAR(100),
age INT,
gender ENUM('Male', 'Female'),
course_id INT
);
```

- `Create Courses Table`

```
CREATE TABLE courses (
course_id INT PRIMARY KEY AUTO_INCREMENT,
course_name VARCHAR(100),
duration VARCHAR(50)
);
```

- `Create Marks Table`

```
CREATE TABLE marks (
mark_id INT PRIMARY KEY AUTO_INCREMENT,
student_id INT,
subject VARCHAR(100),
score DECIMAL(5,2)
);
```

- `Modify Students table to add a new column email`

```
ALTER TABLE Students ADD COLUMN email VARCHAR(100);
```

- `Drop the Marks table and recreate it with the same structure.`

```
DROP TABLE IF EXISTS Marks;
```

```
CREATE TABLE marks (
mark_id INT PRIMARY KEY AUTO_INCREMENT,
```

```
student_id INT,
subject VARCHAR(100),
score DECIMAL(5,2)
);
```

assignment2.sql

▼ Assignment 3

1. Insert 5 Rows per Table

```
-- Insert into Courses
INSERT INTO courses (course_name, duration) VALUES
('DBMS', '6 months'),
('OS', '1 year'),
('Python', '1 year'),
('Java', '6 months'),
('C++', '6 months');

-- Insert into Students
INSERT INTO Students (name, age, gender, course_id, email) VALUES
('Nisharg Soni', 21, 'Male', 1, 'nisharg@gmail.com'),
('Dakshil Gorasiya', 19, 'Male', 3, 'dakshil@gmail.com'),
('Diya Mehta', 22, 'Female', 2, 'diya@gmail.com'),
('Manish Patel', 20, 'Male', NULL, 'manish@gmail.com'),
('Krisha Shah', 23, 'Female', 1, 'krisha@gmail.com');

-- Insert into Marks
INSERT INTO Marks (student_id, subject, score) VALUES
(1, 'DBMS', 88.5),
(2, 'Python', 92.0),
(3, 'OS', 85.0),
(4, 'DLD', 78.0),
(5, 'DBMS', 91.5);
```

2. Update one student’s course.

```
UPDATE Students SET course_id = 4 WHERE student_id = 2;
```

3. Delete a student record.

```
DELETE FROM Students WHERE student_id = 4;
```

assignment3.sql

▼ Assignment 4

1. Students Above Age 20

```
SELECT * FROM students WHERE age > 20;
```

2. Students Ordered Alphabetically

```
SELECT * FROM students ORDER BY name ASC;
```

3. Total Students per Course

```
SELECT course_id, COUNT(*) AS total_students
FROM students
GROUP BY course_id;
```

4. Courses with More Than 2 Students

```
SELECT course_id, COUNT(*) AS student_count
FROM students
GROUP BY course_id
HAVING student_count > 2;
```

assignment4.sql

▼ Assignment 5

1. Display students with their enrolled course names using INNER JOIN.

```
SELECT s.student_id, s.name, c.course_id, c.course_name
FROM Students s
INNER JOIN Courses c ON s.course_id = c.course_id;
```

2. Display all students even if they are not enrolled in any course (LEFT JOIN).

```
SELECT s.student_id, s.name, c.course_id, c.course_name
FROM students s
LEFT JOIN courses c ON s.course_id = c.course_id;
```

3. Display all courses and their students (RIGHT JOIN).

```
SELECT s.student_id, s.name, c.course_id, c.course_name
FROM students s
RIGHT JOIN courses c ON s.course_id = c.course_id;
```

4. Find highest, lowest, and average marks per subject.

```
SELECT subject,
    MAX(score) AS max_score,
    MIN(score) AS min_score,
    AVG(score) AS avg_score
FROM marks
GROUP BY subject;
```

5. Count how many male and female students exist.

```
SELECT gender, COUNT(*) AS total_count
FROM students
GROUP BY gender;
```

assignment5.sql

▼ LibraryDB