B.TECH. COMPUTER SCIENCE AND ENGINEERING
CSLR51 - Database Management Systems Laboratory
#Session: 06 || Date: 05/09/2024

SHALU KUMARI
106122112

```
CREATE TABLE classroom (
 building VARCHAR(50),
 room_number VARCHAR(10),
 capacity INT,
 PRIMARY KEY (building, room_number)
);



CREATE TABLE department (
 dept_name VARCHAR(50),
 building VARCHAR(50),
 budget INT,
 PRIMARY KEY (dept_name)
);



 CREATE TABLE course (
 course_id VARCHAR(10),
 title VARCHAR(100),
 dept_name VARCHAR(50),
 credits INT,
 PRIMARY KEY (course_id),
 FOREIGN KEY (dept_name) REFERENCES department(dept_name) );



 CREATE TABLE professor (
 pID INT,
 name VARCHAR(100),
 dept_name VARCHAR(50),
 salary INT,
 PRIMARY KEY (pID),
 FOREIGN KEY (dept_name) REFERENCES department(dept_name) );



 CREATE TABLE section (
 course_id VARCHAR(10),
 sec_id VARCHAR(10),
 semester VARCHAR(10),
 year INT,
 building VARCHAR(50),
 room_number VARCHAR(10),
 time_slot_id INT,
 PRIMARY KEY (course_id, sec_id, semester, year),
 FOREIGN KEY (course_id) REFERENCES course(course_id),  FOREIGN KEY
(building, room_number) REFERENCES classroom(building, room_number) );

 CREATE TABLE teaches (
 pID INT,
 course_id VARCHAR(10),
 sec_id VARCHAR(10),
 semester VARCHAR(10),
 year INT,
 PRIMARY KEY (pID, course_id, sec_id, semester, year),
 FOREIGN KEY (pID) REFERENCES professor(pID),  FOREIGN KEY
(course_id, sec_id, semester, year) REFERENCES
section(course_id, sec_id, semester, year)
);


CREATE TABLE student (
 pID INT,
 name VARCHAR(100),
 dept_name VARCHAR(50),
 tot_cred INT,
 PRIMARY KEY (pID),
 FOREIGN KEY (dept_name) REFERENCES department(dept_name) );



CREATE TABLE takes (
 sID INT,
 course_id VARCHAR(10),
 sec_id VARCHAR(10),
 semester VARCHAR(10),
 year INT,
 grade CHAR(1),
 PRIMARY KEY (sID, course_id, sec_id, semester, year),
FOREIGN KEY (sID) REFERENCES student(pID),
 FOREIGN KEY (course_id, sec_id, semester, year) REFERENCES
section(course_id, sec_id, semester, year)
);



 CREATE TABLE guide (
 sID INT,
 pID INT,
 PRIMARY KEY (sID, pID),
 FOREIGN KEY (sID) REFERENCES student(pID),
 FOREIGN KEY (pID) REFERENCES professor(pID) );

CREATE TABLE time_slot (
 time_slot_id INT,
 day VARCHAR(10),
 start_time TIME,
 end_time TIME,
 PRIMARY KEY (time_slot_id) );
```

~~~
 CREATE TABLE prereq (
 course_id VARCHAR(10),
 prereq_id VARCHAR(10),
 PRIMARY KEY (course_id, prereq_id),
 FOREIGN KEY (course_id) REFERENCES course(course_id),
FOREIGN KEY (prereq_id) REFERENCES course(course_id) );
```

### Step 2: Insert Sample Data

```sql
-- Insert data into department
INSERT INTO department VALUES ('CSE', 'Building1',
50000); INSERT INTO department VALUES ('ECE',
'Building2', 60000);
INSERT   INTO   department   VALUES   ('ME',   'Building3',
70000);    INSERT    INTO    department    VALUES    ('CE',
'Building4',  80000);  INSERT  INTO  department  VALUES
('EE', 'Building5', 90000);

-- Insert data into course
INSERT INTO course VALUES ('CS-101', 'Data Structures', 'CSE', 3);
INSERT INTO course VALUES ('CS-102', 'Algorithms', 'CSE', 4);
INSERT INTO course VALUES ('CS-103', 'Operating Systems', 'CSE',
3); INSERT INTO course VALUES ('ECE-201', 'Digital Circuits',
'ECE', 3); INSERT INTO course VALUES ('ME-301', 'Thermodynamics',
'ME', 4);

-- Insert data into professor
INSERT INTO professor VALUES (1, 'Dr. John Doe', 'CSE', 120000);
INSERT INTO professor VALUES (2, 'Dr. Jane Smith', 'ECE', 110000);
INSERT INTO professor VALUES (3, 'Dr. Sam Wilson', 'ME', 130000);
INSERT INTO professor VALUES (4, 'Dr. Bruce Banner', 'CE',
140000); INSERT INTO professor VALUES (5, 'Dr. Tony Stark', 'EE',
150000);

-- Insert data into classroom
INSERT INTO classroom VALUES ('Building1', '101', 50);
INSERT INTO classroom VALUES ('Building2', '102', 60);
INSERT INTO classroom VALUES ('Building3', '103', 70);
INSERT INTO classroom VALUES ('Building4', '104', 80);
INSERT INTO classroom VALUES ('Building5', '105', 90);

-- Insert data into section
INSERT INTO section VALUES ('CS-101', '1', 'Fall', 2020,
'Building1', '101', 1);
INSERT INTO section VALUES ('CS-102', '1', 'Fall', 2020,
'Building1', '101', 2);
INSERT INTO section VALUES ('CS-103', '1', 'Spring', 2019,
'Building1', '101', 3);
INSERT INTO section VALUES ('ECE-201', '1', 'Spring', 2019,
'Building2', '102', 4);
INSERT INTO section VALUES ('ME-301', '1', 'Spring', 2019,
'Building3', '103', 5);
~~~

```
-- Insert data into student
INSERT INTO student VALUES (101, 'Alice', 'CSE', 15);
INSERT INTO student VALUES (102, 'Bob', 'ECE', 20);
INSERT INTO student VALUES (103, 'Charlie', 'ME', 30);
INSERT INTO student VALUES (104, 'David', 'CE', 25);
INSERT INTO student VALUES (105, 'Eve', 'EE', 40);

-- Insert data into takes
INSERT INTO takes VALUES (101, 'CS-101', '1', 'Fall', 2020, 'A');
INSERT INTO takes VALUES (102, 'CS-102', '1', 'Fall', 2020, 'B');
INSERT INTO takes VALUES (103, 'CS-103', '1', 'Spring', 2019, 'C');
INSERT INTO takes VALUES (104, 'ECE-201', '1', 'Spring', 2019,
'A'); INSERT INTO takes VALUES (105, 'ME-301', '1', 'Spring', 2019,
'B');

-- Insert data into guide
INSERT INTO guide VALUES (101, 1);
INSERT INTO guide VALUES (102, 2);
INSERT INTO guide VALUES (103, 3);
INSERT INTO guide VALUES (104, 4);
INSERT INTO guide VALUES (105, 5);
-- Insert data into time_slot
INSERT INTO time_slot VALUES (1, 'Monday', '09:00', '10:00');
INSERT INTO time_slot VALUES (2, 'Tuesday', '10:00', '11:00');
INSERT INTO time_slot VALUES (3, 'Wednesday', '11:00',
'12:00'); INSERT INTO time_slot VALUES (4, 'Thursday', '12:00',
'13:00'); INSERT INTO time_slot VALUES (5, 'Friday', '13:00',
'14:00'); ```
```

SQL Queries for the Questions

a. Find the titles of courses in the CSE department that have 3 credits. sql

```
SELECT title FROM course WHERE dept_name = 'CSE' AND credits = 3;
```

b. Find the highest salary of any professor.
sql

```
SELECT MAX(salary) AS highest_salary FROM professor;
```

c. Find all professors earning the highest salary.
sql

```
SELECT name FROM professor WHERE salary = (SELECT MAX(salary)
FROM professor);
```

d. Find the maximum enrollment, across all sections, in Fall 2020. sql

```
SELECT MAX(enrollment_count) FROM (
 SELECT COUNT(*) AS enrollment_count
 FROM takes
 WHERE semester = 'Fall' AND year = 2020
```

```
 GROUP BY course_id, sec_id
) AS enrollments;
```

e. Find the enrollment of each section that was offered in Spring 2019. sql

```
SELECT course_id, sec_id, COUNT(*) AS enrollment
FROM takes
WHERE semester = 'Spring' AND year = 2019
GROUP BY course_id, sec_id;
```

f. Find the IDs and names of all students who have not taken any course offering before Spring 2013.
sql

```
SELECT pID, name
FROM student
WHERE pID NOT IN (
 SELECT sID
 FROM takes
 WHERE (semester = 'Spring' AND year < 2013) OR year < 2013 );
```

g. Find the lowest, across all departments, of the per-department maximum salary.

```
SELECT MIN(max_salary)
FROM (
 SELECT MAX(salary) AS max_salary
 FROM professor
 GROUP BY dept_name
) AS dept_salaries;
```

h. Create a new course â€œCS-001â€, titled â€œWeekly Seminarâ€, with 1 credit.

```
INSERT INTO course VALUES ('CS-001', 'Weekly Seminar', 'CSE', 1);
```

i. Delete the course CS-001. What will happen if you run this delete statement without first deleting offerings (sections) of this course? sql

```
DELETE FROM course WHERE course_id = 'CS-001';
```