Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
(Deemed to be University Estd. u/s 3 of UGC Act, 1956)

WEEKLY LESSON PLAN

Department: Computer Science and Engineering

Year / Semester: 2025-26 / Winter

Course Code / Course Name: 10213CS224 / Full Stack Application Development

Unit I: Database Management and Web Development basics
Course Outcome: • Understand and use database queries and web technologies to build interactive applications.

Week	Session	Topics to be Covered	Problems to Practice (Hands-On)	Resources
Week 1	Session 1	Git	1. Practice basic Git Commands 2. Upload single document to github using bash. 3. Upload and update a folder in github. 4. Create branch and push/commit one folder in Eclipse. 5. Create branch and push/commit one folder in VS Code.	https://youtu.be/vA5TTz6BXhY
	Session 2	SQL	Install SQL tool, MySQL and VS Code. 1. Create a database. 2. Create at least two tables: * Student (VTU Number, Name, Email, Phone, Department) * Course (Course Code, Course Name, Faculty Id, Student ID, Faculty Email)	https://www.programiz.com/sql/online-compiler https://www.w3schools.com

			3. Insert minimum 5 records into each table. 4. Select records using different CLAUSE.	
	Session 3	SQL	1. Write SELECT queries to display all records. 2. Write queries using aggregate functions. 3. Sort data and display based in ascending/descending order of the VTU number. 4. Display student records belonging to a particular department. 5. Map VTU no. with course name and faculty using joins.	https://www.w3schools.com/
	Session 4	SQL	1. Select set of students based on course and count the number of students. 2. Select set of courses a faculty is handling and insert into the new table. 3. Select top 5 students from same department and update their phone no. with country code.	https://youtu.be/EQbhKjBmW88
	Session 5	SQL	1. Initiate and commit a transaction. 2. Set a savepoint, perform transactions and rollback to the save point. 3. Create a user-defined function to perform simple mathematical calculations. 4. Create a user-defined function to execute a select query to return a value.	https://www.w3schools.com/mysql/default.asp

Session 2:

Install SQL tool, MySQL and VS Code.

1. Create a database.

```
CREATE DATABASE fullstack;
USE fullstack;
```

2. Create at least two tables:

Table 1: Student (VTU Number, Name, Email, Phone, Department)

```
CREATE TABLE Student (
 VTU_Number VARCHAR(20) PRIMARY KEY,
 Name VARCHAR(100),
 Email VARCHAR(100),
 Phone VARCHAR(15),
 Department VARCHAR(50)
);
```

Table 2: Course (Course Code, Course Name, Faculty Id, Student ID, Faculty Email)

```
CREATE TABLE Course (
Course_Code VARCHAR(10) PRIMARY KEY,
Course_Name VARCHAR(100),
Faculty_Id INT,
Student_ID VARCHAR(20),
Faculty_Email VARCHAR(100),
FOREIGN KEY (Student_ID) REFERENCES Student(VTU_Number)
);
```

3. Insert minimum 5 records into each table.

```
-- Inserting into Student Table
INSERT INTO Student VALUES
```

```
('1MS23CS001', 'Arjun Kumar', 'arjun@mail.com', '9845011223', 'CSE'),
('1MS23CS002', 'Sneha Rao', 'sneha@mail.com', '9845022334', 'IT'),
('1MS23CS003', 'Rahul Nair', 'rahul@mail.com', '9845033445', 'CSE'),
('1MS23CS004', 'Priya Das', 'priya@mail.com', '9845044556', 'ECE'),
('1MS23CS005', 'Amit Singh', 'amit@mail.com', '9845055667', 'IT');
-- Inserting into Course Table
INSERT INTO Course VALUES
('CS101', 'Database Management', 101, '1MS23CS001', 'prof_smith@vtu.edu'),
('IS202', 'Data Structures', 102, '1MS23CS002', 'prof_jones@vtu.edu'),
('CS103', 'Operating Systems', 101, '1MS23CS003', 'prof_smith@vtu.edu'),
('EC301', 'Digital Electronics', 103, '1MS23CS004', 'prof_kumar@vtu.edu'),
('IS204', 'Web Programming', 104, '1MS23CS005', 'prof_leila@vtu.edu');
```

4. Select records using different CLAUSE.

   i. **WHERE clause:**

   ```
   SELECT * FROM Student WHERE Department = 'CSE';
   ```

   ii. **ORDER BY Clause (Sorting):**

   ```
   SELECT * FROM Course ORDER BY Course_Name ASC;
   ```

   iii. **LIKE Clause (Pattern Matching):**

   ```
   SELECT Name, Email FROM Student WHERE Email LIKE '%@mail.com';
   ```

   iv. **LIMIT Clause (Restricting Results):**

   ```
   SELECT * FROM Student LIMIT 3;
   ```

v. **GROUP BY & HAVING Clause (Aggregating):**

```
SELECT Department, COUNT(*) as Total_Students
FROM Student
GROUP BY Department
HAVING COUNT(*) > 1;
```

Session 3:

1. Write SELECT queries to display all records.

```
-- Display all students
SELECT * FROM Student;
-- Display all courses
SELECT * FROM Course;
```

2. Write queries using aggregate functions.

```
-- Count total number of students
SELECT COUNT(*) AS Total_Students FROM Student;
-- Count unique departments
SELECT COUNT(DISTINCT Department) AS Department_Count FROM Student;
-- Find the highest Faculty ID assigned
SELECT MAX(Faculty_Id) AS Highest_Faculty_ID FROM Course;
```

3. Sort data and display based in ascending/descending order of the VTU number.

```
-- Sort - Ascending order (Default)
SELECT * FROM Student
ORDER BY VTU_Number ASC;
-- Sort - in Descending order
SELECT * FROM Student
ORDER BY VTU_Number DESC;
```

4. Display student records belonging to a particular department.

```
SELECT * FROM Student
WHERE Department = 'CSE';
```

5. Map VTU no. with course name and faculty using joins.

Joins Summary:

Join Type	Records Included
Inner	Only matches found in both tables.
Left	Everything from Left + matches from Right.
Right	Everything from Right + matches from Left.
Full	Everything from both tables.
Cross	Every possible combination of both tables.

Inner Join:

```
SELECT
   s.VTU_Number,
   s.Name AS Student_Name,
   c.Course_Name,
   c.Faculty_Id,
   c.Faculty_Email
FROM Student s
INNER JOIN Course c ON s.VTU_Number = c.Student_ID;
```

Left Join:

```
SELECT
   s.VTU_Number,
   s.Name,
   c.Course_Name
FROM Student s
LEFT JOIN Course c ON s.VTU_Number = c.Student_ID;
```

Right Join:

```
SELECT
   s.Name,
   c.Course_Code,
   c.Course_Name
FROM Student s
RIGHT JOIN Course c ON s.VTU_Number = c.Student_ID;
```

FULL JOIN (Full Outer Join):

```
SELECT s.Name, c.Course_Name FROM Student s
LEFT JOIN Course c ON s.VTU_Number = c.Student_ID
UNION
SELECT s.Name, c.Course_Name FROM Student s
RIGHT JOIN Course c ON s.VTU_Number = c.Student_ID;
```

Cross Join(Cartesian Product):

```
SELECT s.Name, c.Course_Name
FROM Student s
CROSS JOIN Course c;
```