M. M. Institute of Computer Technology & Business Management
Maharishi Markandeshwar (Deemed to be University)
Mullana – Ambala, Haryana (India) – 133207
(Established Under Section 3 of UGC Act. 1956)
(Accredited by NAAC with Grade 'A')

Subject Code: BCA-306 **Subject Title:** Database Management System Lab

---

Course Objective:

- Understand the basic concepts and the applications of database systems.
- Learn the basics of SQL and construct queries using MySQL.
- Understand the relational database design and different types of key constraints.
- Applying advanced query, joining tables using joins and using subqueries.

List of Practicals

1. Database Creation and Table Design

- Task: Create a database named "University_Records".
- Tables:
  - Students: Student_Id, Student_Name, Department, Year_Of_Study, Contact_No, Email, Address.
  - Courses: Course_Id, Course_Name, Credits, Department.
  - Enrollments: Enrollment_Id, Student_Id, Course_Id, Enrollment_Date.
  - Professors: Professor_Id, Professor_Name, Department, Contact_No, Email.

2. Data Manipulation

- Insert: Populate the Students table with at least 20 records, Courses with 10 records, and Enrollments with 30 records.
- Update: Change the department of students enrolled in specific courses.
- Delete: Remove the enrollment records of students who have graduated.
- Alter:
  - Modify the Contact_No data type in Students to VARCHAR(15).
  - Add a Grade column in the Enrollments table to record student grades.
- Drop the Professors table after confirming it's no longer needed.

3. Constraints and Keys

- Primary Key: Apply on Student_Id, Course_Id, Enrollment_Id, and Professor_Id.
- Unique Key: Ensure Email in Students and Professors is unique.
- Not Null: Apply on Student_Name, Course_Name, and Professor_Name.
- Foreign Key:
  - Link Student_Id in Enrollments to Student_Id in Students.
  - Link Course_Id in Enrollments to Course_Id in Courses.
- Check Constraint: Ensure that Credits in Courses are between 1 and 5.

4. Advanced Query Operations

- ***Order By:*** List all courses ordered by Course_Name.
- ***Group By:*** Calculate the average grade for each course.
- ***Having:*** Identify departments with an average course enrollment greater than 100 students.

5. Using SQL Operators

- ***IN:*** List students who belong to the "Engineering" or "Business" departments.
- ***BETWEEN:*** Find courses with credit hours between 2 and 4.
- ***Concatenation:*** Display the full names of students by combining their first and last names.
- ***LIKE:*** Find students whose names start with 'S' and are in their final year.

6. Views

- ***Create a View:***
  - Display student names, course names, and enrollment dates.
  - Filter the view to include only students in their final year.
- ***Update View:*** Modify the view to also include the student's GPA.
- ***Drop View:*** Remove the view when it is no longer needed.

7. Join Operations

- ***Inner Join****:* List all students along with the courses they are enrolled in.
- ***Left Outer Join:*** Display all courses and their enrolled students, including courses with no enrollments.
- ***Right Outer Join:*** List all professors and the courses they teach, including professors who currently teach no courses.
- ***Full Outer Join:*** Combine all students and their enrollment details, showing all records even if some information is missing.
- ***Self Join:*** Identify students who share the same address.

8. Aggregation and Subqueries

- ***Aggregation:***
  - Calculate the average grade for students enrolled in "Computer Science" courses.
  - List the names and contact details of students with the highest GPA.
- ***Subqueries:***
  a. Find the names of students enrolled in more than three courses in the current semester.
  b. List the courses that have fewer than five students enrolled.

9. PL/SQL Functions

- Create a Function to calculate the total credits a student is enrolled in.
- Create a Function that accepts Course_Id and returns the number of students enrolled in that course.
- Create a Function to find the top 3 highest grades in a specific course.

10. Transaction Management and Security

- ***Roles and Privileges:***
  - Create roles for "Student_Admin", "Course_Admin", and "Professor_Admin" with appropriate access permissions.
- ***Grant and Revoke:***
  - Implement commands to manage access for these roles on the University_Records database.
- ***Transaction Control:***
  - Demonstrate the use of Commit, Rollback, and Savepoint by simulating a scenario where a student's course enrollment needs to be reverted.

11. Complex queries

- Write a query to calculate the average grade of students in each department. Display the department name and average grade.

- Write a query to find the departments that offer the most number of courses. Display the department name and the number of courses.
- Write a query to display all courses along with the number of students enrolled in each course. Include courses that have no students enrolled.
- Write a query to find the student with the highest GPA in each year of study. Display the year, student name, and GPA.
- Write a query to find professors who teach the most number of courses. Display the professor's name and the count of courses they teach.

Teaching and Examination Scheme

Teaching Hours	Credits	ExaminationalMarks			Internal Break-up			
02	2.0	Internal	Practical	Total	Attendance	Performance of Practical	Lab. File	Viva Voce
		60	40	100	10*	30	10	10

***In proportion to the percentage of classes attended.**