Data Retrieval from Multiple Tables Using JOINs

Objective: To understand and apply INNER JOIN operations to retrieve meaningful information from multiple related tables by correctly using primary–foreign key relationships and table aliases, without modifying the underlying database structure or stored data.

##

Before executing the JOIN queries in this lab, ensure that your tables contain **appropriate and sufficient data** to produce meaningful results.

If the required data for any question is not present in your tables, you must **either insert suitable data** (using realistic values consistent with the college database) **or modify the query conditions** so that they work correctly with the **existing data** in your database.

You are **not allowed to change the table structure**, but you may add records if necessary to satisfy the requirements of the questions.

##

Short Notes (Read Before Attempting the Questions)

1. Why JOIN is Needed

In a relational database, data is stored in **multiple related tables** to avoid redundancy. To get meaningful information (for example, student name with course name), data must be retrieved **from more than one table at the same time**. This is done using **JOIN operations**.

2. INNER JOIN

INNER JOIN retrieves only those records that have **matching values in both tables**.

Example situation:

- Student table has `StudentID`

- Enrollment table has `StudentID`
  Only students who are present in Enrollment will appear in the result.

Example code

```
SELECT Student.Name, Enrollment.CourseID
FROM Student
INNER JOIN Enrollment
ON Student.StudentID = Enrollment.StudentID;
```

3. JOIN Condition

A JOIN condition specifies **how two tables are related**, usually using:

- Primary key of one table

- Foreign key of another table

Example:
Student.StudentID = Enrollment.StudentID

4. Using Aliases with JOIN

Aliases make queries easier to read, especially when multiple tables are involved.

Example code:

```
SELECT S.Name, C.CourseName
FROM Student AS S
INNER JOIN Course AS C
ON S.DepartmentID = C.DepartmentID;
```

Aliases do not change table names permanently.

5. JOINing More Than Two Tables

More than two tables can be joined in a single query.

Example situation:
Student → Enrollment → Course

Example code:

```
SELECT S.Name, C.CourseName
FROM Student AS S
INNER JOIN Enrollment AS E
ON S.StudentID = E.StudentID
INNER JOIN Course AS C
ON E.CourseID = C.CourseID;
```

6. Read-Only Nature of JOIN Queries

JOIN queries **do not modify data**.
They only display combined information from multiple tables.

Lab Questions

Part A: Student–Enrollment Relationship

1. Display `StudentID` and `Name` from Student table along with `CourseID` from Enrollment table.

2. Display student `Name` and `Semester` for all enrolled students.

3. Display student `Name` and `Grade` obtained in each course.

Part B: Course–Faculty Relationship

4. Display `CourseName` and `Credits` along with the `Faculty Name` who teaches the course.

5. Display all courses taught by a faculty member whose `FacultyID` is `'F201'`.

6. Display faculty `Name` and `Designation` along with the `CourseName` they teach.

Part C: Student–Course–Enrollment Relationship

7. Display student `Name` along with `CourseName` for all enrollments.
8. Display student `Name`, `CourseName`, and `Semester`.
9. Display student `Name`, `CourseName`, and `Grade` for Semester 4 only.
10. Display all courses taken by a student whose `StudentID` is `'S401'`.

Part D: Department-Based JOIN Queries

11. Display student `Name` and `DepartmentID` along with `CourseName`.
12. Display department-wise list of courses using `DepartmentID`.
13. Display faculty `Name` along with department name they belong to.

Part E: JOIN with Filtering and Sorting

14. Display student `Name` and `CourseName` for students belonging to Department `'D101'`.
15. Display student `Name` and `CourseName` sorted by student name in ascending order.

16. Display faculty `Name` and `CourseName` sorted by course credits in descending order.

Part F: Analytical JOIN Queries

17. Display number of students enrolled in each course.
18. Display number of courses taught by each faculty member.
19. Display department-wise count of students.
20. Display course name along with maximum grade awarded in that course.

Final Reminder to Students

- Use **INNER JOIN only** for this lab.
- Use **table aliases** to improve readability.
- Ensure JOIN conditions are correct and meaningful.
- Do not modify any table or data.