Database Structure

1. `students`

Stores data about each student.

Column	Data Type	Description
`student_id`	INT (PK)	Unique ID for each student
`name`	VARCHAR	Full name of the student
`gender`	VARCHAR	Gender of the student
`department_id`	INT (FK)	Links student to their department

---

2. `departments`

List of academic departments.

Column	Data Type	Description
`department_id`	INT (PK)	Unique ID for each department
`department_name`	VARCHAR	Name of the department

---

3. `courses`

Holds information about the available courses.

Column	Data Type	Description
`course_id`	INT (PK)	Unique ID for each course
`course_name`	VARCHAR	Name of the course

`department_id`	INT (FK)	Department offering the course
`instructor_id`	INT (FK)	Instructor assigned to the course

4. `instructors`

Contains data about instructors.

Column	Data Type	Description
`instructor_id`	INT (PK)	Unique ID for each instructor
`name`	VARCHAR	Name of the instructor

5. `enrollments`

Tracks student enrollments in courses.

Column	Data Type	Description
`enrollment_id`	INT (PK)	Unique ID for each enrollment
`student_id`	INT (FK)	The student enrolled
`course_id`	INT (FK)	The course the student enrolled in

Entity Relationships

```
students.department_id    → departments.department_id
students.student_id       → enrollments.student_id
courses.course_id         → enrollments.course_id
courses.department_id     → departments.department_id
courses.instructor_id     → instructors.instructor_id
```

Query Logic Documentation

Student & Enrollment Reports

Q1: How many students are currently enrolled in each course?

```
SELECT c.course_name AS course, COUNT(e.student_id) AS no_of_students
FROM enrollments AS e
JOIN courses AS c ON e.course_id = c.course_id
GROUP BY course;
```

- **Goal:** Count how many students are enrolled per course.

First select the course name column to list all courses and count the number of students through their id in the enrollment table to be sure you're getting all enrolled students. Join the two tables and of course, group the aggregate column by course.

Q2: Which students are enrolled in multiple courses, and which courses are they taking?

```
SELECT s.name AS students, COUNT(e.course_id) AS
no_of_enrolled_courses, c.course_name AS courses
FROM students AS s
JOIN courses AS c ON s.department_id = c.department_id
JOIN enrollments AS e ON s.student_id = e.student_id
GROUP BY students, courses
HAVING COUNT(e.course_id) > 1;
```

- **Goal:** Show students enrolled in **more than one course**, along with each course.

Select the students' name, count the number of courses they are enrolled in through their course id in the enrollments table, and the name of the course as well. Join these three tables on the appropriate columns and group the aggregate column by the students' name and courses(I’m using MySQL), also filter for course count that’s more than one.

Q3: Total number of students per department across all courses

```
SELECT department_name AS department , COUNT(DISTINCT e.student_id) AS
no_of_students
FROM courses AS c
JOIN departments AS d ON c.department_id = d.department_id
JOIN enrollments AS e ON c.course_id = e.course_id
GROUP BY department;
```

- **Goal:** For each department, count unique students across all their courses.

Select the department name column and count the unique students through their student id in the enrollments table to be sure you’re counting enrolled students. Join the two tables appropriately and group the aggregate column by departments.

Course & Instructor Analysis

Q1: Which courses have the highest number of enrollments?

```
SELECT course_name AS courses, COUNT(DISTINCT e.student_id) AS
no_of_enrollment
FROM courses AS c
JOIN enrollments AS e ON c.course_id = e.course_id
GROUP BY courses
ORDER BY no_of_enrollment DESC
LIMIT 2;
```

- **Goal:** Identify top courses with most enrolled students.

Select the course name columns and count the number of unique students enrolled. Join the two tables appropriately and group the aggregate column by the courses. Order the table by the number of enrollment descending to sort from the highest to lowest and use limit to show just the top courses. In this case, there were 2 top courses.

Q2: Which department has the least number of students?

```
SELECT department_name AS departments, COUNT(s.department_id) AS
no_of_students
FROM departments AS d
JOIN students AS s ON d.department_id = s.department_id
GROUP BY departments
ORDER BY no_of_students
LIMIT 1;
```

- **Goal:** Find the department with the lowest student count.

Select the department name column and count the number of students through their department id in the students table. Join the two tables and group the aggregate column by the department name. Order by the number of students to sort from lowest to highest then limit to one to return only the first row.

Data Integrity & Operational Insights

Q1: Are there any students not enrolled in any course?

```
SELECT s.student_id, s.name AS not_enrolled_student
FROM students AS s
LEFT JOIN enrollments AS e ON s.student_id = e.student_id
WHERE enrollment_id IS NULL;
```

- **Goal:** Find students who don't appear in `enrollments`.

Select the student id and the name of the student column. Left join to the enrollment table since we are looking for not enrolled students and filter for enrollment ids that are NULL values meaning not enrolled.

Q2: How many courses does each student take on average?

```
SELECT ROUND((COUNT(*)/COUNT(DISTINCT student_id)), 1) AS
ave_course_per_student
FROM enrollments;
```

- **Goal:** Get the average number of courses per student.

Basically Total enrollments ÷ unique students and rounded to 1 decimal point.

Q3: Gender distribution across courses and instructors

```
SELECT c.course_name AS course, i.name as instructors,
 SUM(CASE WHEN s.gender = 'Male' THEN 1 ELSE 0 END) AS Male_students,
 SUM(CASE WHEN s.gender = 'Female' THEN 1 ELSE 0 END) AS
Female_students
FROM courses AS c
JOIN students AS s ON c.department_id = s.department_id
JOIN instructors AS i ON c.instructor_id = i.instructor_id
GROUP BY c.course_name, i.name;
```

- **Goal:** Show gender count by course and instructor.

Select course name column, instructor name column, and use sum of case statements to add up the number of male and female students separately. Join the appropriate tables and group the aggregate columns by the courses' name and the instructors' name (I'm using MySQL).

Q4: Course with highest number of male or female students

```
SELECT c.course_name AS course,
 SUM(CASE WHEN s.gender = 'Male' THEN 1 ELSE 0 END) AS Male_students,
 SUM(CASE WHEN s.gender = 'Female' THEN 1 ELSE 0 END) AS
Female_students
FROM courses AS c
JOIN students AS s ON c.department_id = s.department_id
GROUP BY c.course_name
ORDER BY Male_students DESC, Female_students DESC
LIMIT 2;
```

- **Goal:** Find the top courses based on gender-specific enrollments.

Similar to the last query, select the courses' name column and the previous aggregate columns, Join the tables and Group by the course name column. Order by both male and female students descending to sort from highest to lowest and limit to get only the rows with the top courses by gender. In this case, there's 2.