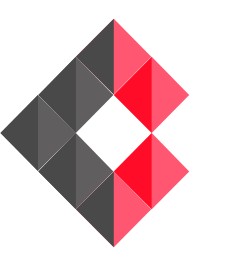
**Internship Task** - **RDBMS and SQL Task #1**



SQL case-based assignment using a **University Database** schema. This assignment will involve queries related to students, courses, departments, professors, and enrollments. I'll walk through the case, describe the database schema, and then provide 10 SQL queries related to university data analysis.

# Database Schema

**Students Table:**

**Column Name Data Type Description**

|  |  |
| --- | --- |
| student\_id | INT Primary key |
| first\_name | VARCHAR(100) Student's first name |
| last\_name | VARCHAR(100) Student's last name |
| email | VARCHAR(100) Student's email address |
| phone | VARCHAR(20) Student's phone number |
| date\_of\_birth | DATE Student's date of birth |

enrollment\_date DATE Date the student enrolled department\_id INT Foreign key (references Departments)

**Courses Table:**

**Column Name Data Type Description** course\_id INT Primary key

course\_name VARCHAR(100) Course name

|  |  |  |
| --- | --- | --- |
| department\_id | INT | Foreign key (references Departments) |
| professor\_id | INT | Foreign key (references Professors) |
| credits | INT | Number of credits for the course |

**Departments Table:**

**Column Name Data Type Description** department\_id INT Primary key

department\_name VARCHAR(100) Department name

**Professors Table:**

**Column Name Data Type Description** professor\_id INT Primary key

first\_name VARCHAR(100) Professor's first name last\_name VARCHAR(100) Professor's last name email VARCHAR(100) Professor's email address phone VARCHAR(20) Professor's phone number

**Enrollments Table:**

**Column Name Data Type Description** enrollment\_id INT Primary key student\_id INT Foreign key (references Students) **Column Name Data Type Description**

course\_id INT Foreign key (references Courses) enrollment\_date DATE Date the student enrolled in the course grade VARCHAR(5) Grade received in the course

# Case Study: University Data Analysis

**Background:**

You are a database analyst for a university. The university wants to generate several reports based on student enrollment, courses, professors, departments, and performance analysis.

# SQL Queries for the Case Study

1. **Find the Total Number of Students in Each Department**

**Ans :**

SELECT d.department\_name, COUNT(s.student\_id) AS total\_students

FROM Students s

JOIN Departments d ON s.department\_id = d.department\_id

GROUP BY d.department\_name;

1. **List All Courses Taught by a Specific Professor**

**Ans :**

SELECT c.course\_name

FROM Courses c

WHERE c.professor\_id = 1; // Replace 1 with the professor\_id

1. **Find the Average Grade of Students in Each Course**

**Ans :**

SELECT c.course\_name, AVG(e.grade) AS average\_grade

FROM Enrollments e

JOIN Courses c ON e.course\_id = c.course\_id

GROUP BY c.course\_name;

1. **List All Students Who Have Not Enrolled in Any Courses**

**Ans :**

SELECT s.first\_name, s.last\_name

FROM Students s

WHERE NOT EXISTS (

SELECT 1

FROM Enrollments e

WHERE s.student\_id = e.student\_id

);

1. **Find the Number of Courses Offered by Each Department**

**Ans :**

SELECT d.department\_name, COUNT(c.course\_id) AS num\_courses

FROM Departments d

LEFT JOIN Courses c ON d.department\_id = c.department\_id

GROUP BY d.department\_name;

1. **List All Students Who Have Taken a Specific Course (e.g., 'Database Systems')**

**Ans :**

SELECT s.first\_name, s.last\_name

FROM Students s

JOIN Enrollments e ON s.student\_id = e.student\_id

JOIN Courses c ON e.course\_id = c.course\_id

WHERE c.course\_name = 'Database Systems';

1. **Find the Most Popular Course Based on Enrollment Numbers**

**Ans :**

SELECT c.course\_name, COUNT(e.student\_id) AS enrollment\_count

FROM Courses c

JOIN Enrollments e ON c.course\_id = e.course\_id

GROUP BY c.course\_name

ORDER BY enrollment\_count DESC

LIMIT 1;

1. **Find the Average Number of Credits Per Student in a Department**

**Ans :**

SELECT AVG(credits\_per\_student) AS average\_credits

FROM (

SELECT s.student\_id, SUM(c.credits) AS credits\_per\_student

FROM Students s

JOIN Enrollments e ON s.student\_id = e.student\_id

JOIN Courses c ON e.course\_id = c.course\_id

WHERE s.department\_id = 1 -- Replace 1 with the department ID

GROUP BY s.student\_id

) AS student\_credits;

1. **List All Professors Who Teach in More Than One Department**

**Ans :**

SELECT p.first\_name, p.last\_name

FROM Professors p

WHERE (

SELECT COUNT(DISTINCT c.department\_id)

FROM Courses c

WHERE c.professor\_id = p.professor\_id

) > 1;

1. **Get the Highest and Lowest Grade in a Specific Course (e.g., 'Operating Systems')**

**Ans :**

SELECT MAX(e.grade) AS highest\_grade, MIN(e.grade) AS lowest\_grade

FROM Enrollments e

JOIN Courses c ON e.course\_id = c.course\_id

WHERE c.course\_name = 'Operating Systems';

**Task Summary:**

This SQL case study simulates the analysis of university-related data, with queries that focus on **students**, **courses**, **professors**, **departments**, and **enrollments**. The queries are designed to answer various questions such as finding the total number of students in a department, identifying the most popular courses, calculating average grades, and analyzing professor workloads across departments.

Each query uses SQL concepts such as JOIN, GROUP BY, COUNT(), AVG(), MAX(), MIN(), and filtering with WHERE. The data analysis could be useful for academic planning, performance tracking, and resource allocation within the university.