|  |  |  |
| --- | --- | --- |
|  | Pimpri Chinchwad Education Trust’s  **Pimpri Chinchwad College of Engineering** An Autonomous Institute  (Permanently affiliated to Savitribai Phule Pune University) |  |
| SEMESTER-IV |
| Assignment 7 | | |

**Assignment 7 : Implement SQL queries to demonstrate the nested queries in SQL using MySql.** Consider the following database schema :

**Tables:**

**Students Table:**

CREATE TABLE Students (

StudentID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

Age INT CHECK (Age >= 18), -- Students must be at least 18

DepartmentID INT,

CGPA DECIMAL(3,2) CHECK (CGPA BETWEEN 0 AND 10), -- CGPA should be between 0 and 10 FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID) ON DELETE SET NULL

);

**Departments Table:**

CREATE TABLE Departments (

DepartmentID INT PRIMARY KEY AUTO\_INCREMENT,

DeptName VARCHAR(100) UNIQUE NOT NULL,

HOD VARCHAR(100) NOT NULL -- Name of the Head of Department

);

**Courses Table:**

CREATE TABLE Courses (

CourseID VARCHAR(10) PRIMARY KEY, -- Course codes like CSE101, ECE201

CourseName VARCHAR(100) NOT NULL,

DepartmentID INT,

FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID) ON DELETE CASCADE

);

**Enrollments Table:**

CREATE TABLE Enrollments (

EnrollmentID INT PRIMARY KEY AUTO\_INCREMENT,

StudentID INT,

CourseID VARCHAR(10),

Marks INT CHECK (Marks BETWEEN 0 AND 100), -- Marks should be between 0 and 100

FOREIGN KEY (StudentID) REFERENCES Students(StudentID),

FOREIGN KEY (CourseID) REFERENCES Courses(CourseID),

UNIQUE (StudentID, CourseID) -- A student cannot enroll in the same course more than once

);

SOLUTION:-

CREATE TABLE Departments180 (

DepartmentID INT PRIMARY KEY AUTO\_INCREMENT,

DeptName VARCHAR(100) UNIQUE NOT NULL,

HOD VARCHAR(100) NOT NULL

);

CREATE TABLE Students180 (

StudentID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

Age INT,

DepartmentID INT,

CGPA DECIMAL(3,2),

FOREIGN KEY (DepartmentID) REFERENCES Departments180(DepartmentID) ON DELETE SET NULL,

CONSTRAINT Age\_Check CHECK (Age >= 18),

CONSTRAINT CGPA\_Check CHECK (CGPA BETWEEN 0 AND 10)

);

CREATE TABLE Courses180(

CourseID VARCHAR(10) PRIMARY KEY, -- Course codes like CSE101, ECE201

CourseName VARCHAR(100) NOT NULL,

DepartmentID INT,

FOREIGN KEY (DepartmentID) REFERENCES Departments180(DepartmentID) ON DELETE CASCADE

);

CREATE TABLE Enrollments180 (

EnrollmentID INT PRIMARY KEY AUTO\_INCREMENT,

StudentID INT,

CourseID VARCHAR(10),

Marks INT CHECK (Marks BETWEEN 0 AND 100), -- Marks should be between 0 and 100

FOREIGN KEY (StudentID) REFERENCES Students180(StudentID),

FOREIGN KEY (CourseID) REFERENCES Courses180(CourseID),

UNIQUE (StudentID, CourseID) -- A student cannot enroll in the same course more than once

);

-- Insert data into Departments180

INSERT INTO Departments180 (DeptName, HOD) VALUES

('Computer Science', 'Dr. Smith'),

('Electronics', 'Dr. Johnson'),

('Mechanical', 'Dr. Brown'),

('Civil', 'Dr. White'),

('Chemical', 'Dr. Davis'),

('Biotechnology', 'Dr. Miller'),

('Aerospace', 'Dr. Wilson'),

('Automobile', 'Dr. Taylor'),

('IT', 'Dr. Anderson'),

('Artificial Intelligence', 'Dr. Thomas'),

('Data Science', 'Dr. Moore'),

('Robotics', 'Dr. Jackson'),

('Cyber Security', 'Dr. Harris'),

('Electrical', 'Dr. Martin'),

('Environmental', 'Dr. Lewis');

-- Insert data into Students180

INSERT INTO Students180 (Name, Age, DepartmentID, CGPA) VALUES

('Alice Johnson', 20, 1, 8.5),

('Bob Smith', 22, 2, 7.8),

('Charlie Brown', 21, 3, 9.2),

('David Wilson', 23, 4, 6.5),

('Emma Davis', 24, 5, 7.9),

('Frank Miller', 20, 6, 8.1),

('Grace Hall', 22, 7, 7.2),

('Hannah White', 19, 8, 9.5),

('Isaac King', 21, 9, 6.8),

('Jack Turner', 22, 10, 9.1),

('Kevin Adams', 24, 11, 5.9),

('Lily Baker', 23, 12, 8.9),

('Michael Scott', 21, 13, 7.0),

('Nancy Green', 20, NULL, 8.0), -- No department (For testing ON DELETE SET NULL)

('Oliver Wright', 23, 14, 9.7);

-- Insert data into Courses180

INSERT INTO Courses180 (CourseID, CourseName, DepartmentID) VALUES

('CSE101', 'Data Structures', 1),

('CSE102', 'Algorithms', 1),

('ECE201', 'Digital Circuits', 2),

('ME301', 'Thermodynamics', 3),

('CIV401', 'Structural Analysis', 4),

('CH501', 'Chemical Reactions', 5),

('BT601', 'Genetic Engineering', 6),

('AE701', 'Aerodynamics', 7),

('AU801', 'Automobile Engines', 8),

('IT901', 'Operating Systems', 9),

('AI101', 'Machine Learning', 10),

('DS201', 'Big Data Analytics', 11),

('ROB301', 'Robotic Vision', 12),

('CYB401', 'Network Security', 13),

('EE501', 'Power Systems', 14);

-- Insert data into Enrollments180

INSERT INTO Enrollments180 (StudentID, CourseID, Marks) VALUES

(1, 'CSE101', 90),

(1, 'CSE102', 85),

(2, 'ECE201', 78),

(3, 'ME301', 88),

(4, 'CIV401', 67),

(5, 'CH501', 45),

(6, 'BT601', 92), -- Highest marks in a course

(7, 'AE701', 81),

(8, 'AU801', 55),

(9, 'IT901', 61),

(10, 'AI101', 77),

(11, 'DS201', 83),

(12, 'ROB301', 49),

(13, 'CYB401', 39), -- Failed student (Marks < 40)

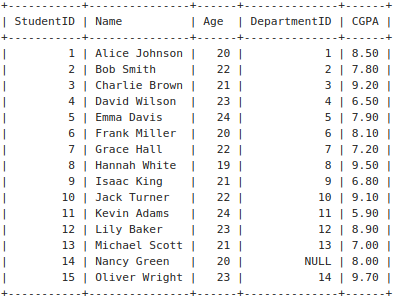
(14, 'EE501', 96); -- High score for highest marks query

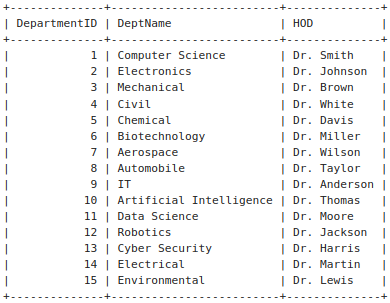
SELECT \* FROM Students180;

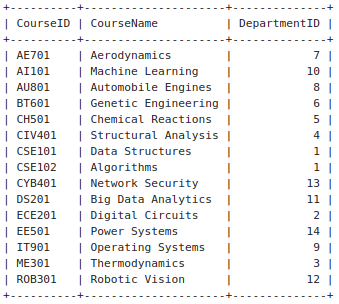
SELECT \* FROM Departments180;

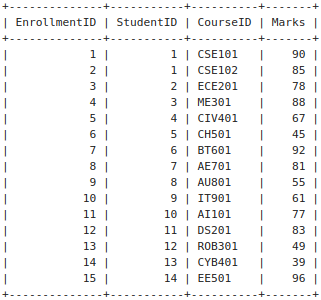
SELECT \* FROM Courses180;

SELECT \* FROM Enrollments180;









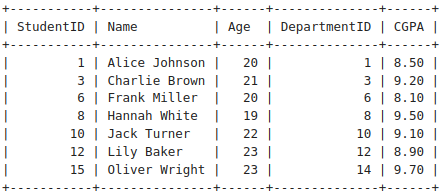
**Solve following SQL Queries:**

1. Find students who have a CGPA higher than the university average CGPA

SELECT \* FROM Students180

WHERE CGPA > (SELECT AVG(CGPA)

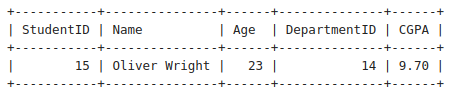
FROM Students180);



2. Find students who have never enrolled in any course

SELECT \* FROM Students180

WHERE StudentID NOT IN (SELECT DISTINCT StudentID FROM Enrollments180);



3. Get courses that have no students enrolled

SELECT \* FROM Courses180

WHERE CourseID NOT IN (SELECT DISTINCT CourseID

FROM Enrollments180);

* 🡪 Empty Set

4. Get students enrolled in courses offered by the 'Computer Sci' department

SELECT DISTINCT s.\*

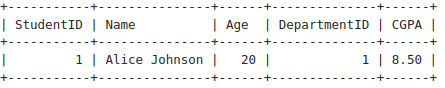
FROM Students180 s

JOIN Enrollments180 e ON s.StudentID = e.StudentID

JOIN Courses180 c ON e.CourseID = c.CourseID

JOIN Departments180 d ON c.DepartmentID = d.DepartmentID

WHERE d.DeptName = 'Computer Science';

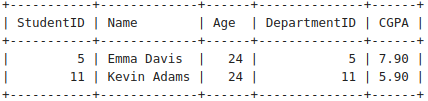


5. Find students who have the same age as the oldest student

SELECT \* FROM Students180

WHERE Age = (SELECT MAX(Age)

FROM Students180);

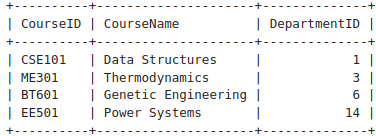


6. Retrieve courses where at least one student scored more than 85 marks

SELECT DISTINCT c.\* FROM Courses180 c

JOIN Enrollments180 e ON c.CourseID = e.CourseID

WHERE e.Marks > 85;

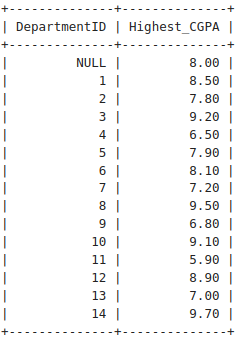


7. Find the highest CGPA in each department

SELECT DepartmentID, MAX(CGPA) AS Highest\_CGPA

FROM Students180

GROUP BY DepartmentID;



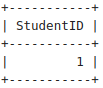
8. Get students who have enrolled in more than one course

SELECT StudentID

FROM Enrollments180

GROUP BY StudentID

HAVING COUNT(CourseID) > 1;



9. Get the department with the most students enrolled

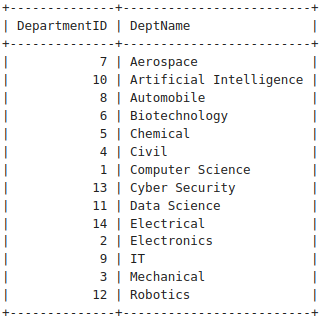
SELECT d.DepartmentID, d.DeptName

FROM Departments180 d

JOIN Students180 s ON d.DepartmentID = s.DepartmentID

GROUP BY d.DepartmentID, d.DeptName

ORDER BY COUNT(s.StudentID) DESC;

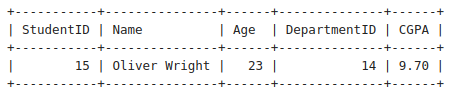


10. Find students who are not enrolled in any course

SELECT \* FROM Students180

WHERE StudentID NOT IN (SELECT DISTINCT StudentID

FROM Enrollments180);



11. Retrieve students who scored below the average marks in their courses

SELECT e.StudentID, e.CourseID, e.Marks

FROM Enrollments180 e

WHERE e.Marks < (SELECT AVG(Marks)

FROM Enrollments180

WHERE CourseID = e.CourseID);

* Empty Set

12. Find the department where the highest CGPA student belongs

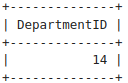
SELECT DepartmentID

FROM Students180

WHERE

CGPA = (SELECT MAX(CGPA)

FROM Students180);



13. Find the department with the highest number of students enrolled

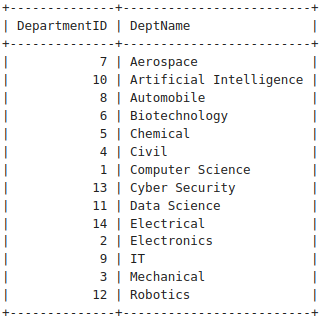
SELECT d.DepartmentID, d.DeptName

FROM Departments180 d

JOIN Students180 s ON d.DepartmentID = s.DepartmentID

GROUP BY d.DepartmentID, d.DeptName

ORDER BY COUNT(s.StudentID) DESC;



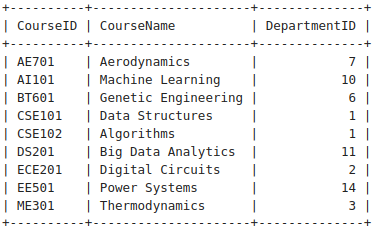
14. Get courses where the lowest score is higher than 70

SELECT c.\* FROM Courses180 c

WHERE 70 < (SELECT MIN(Marks)

FROM Enrollments180

WHERE CourseID = c.CourseID);

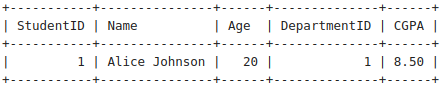


15. Retrieve students who have the same CGPA as 'Alice'

SELECT \* FROM Students180

WHERE CGPA = (SELECT CGPA FROM Students180

WHERE Name = 'Alice Johnson');



16. Get the course where the maximum number of students have enrolled

SELECT CourseID

FROM Enrollments180

GROUP BY CourseID

ORDER BY COUNT(StudentID)

DESC LIMIT 1;



17. Find students who are enrolled in more than two courses

SELECT StudentID

FROM Enrollments180

GROUP BY StudentID

HAVING COUNT(CourseID) > 2;

* 🡪 Empty Set

18. Retrieve students who have scored the highest marks in any course

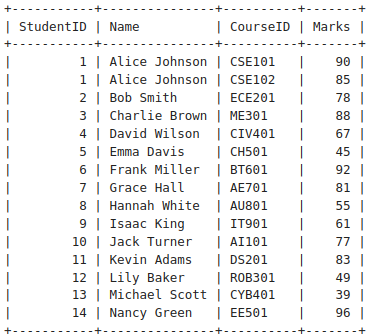
(Hint: Use Join along with nested quesries).

SELECT s.StudentID, s.Name, e.CourseID, e.Marks

FROM Students180 s

JOIN Enrollments180 e ON s.StudentID = e.StudentID

WHERE e.Marks = ( SELECT MAX(Marks) FROM Enrollments180 WHERE CourseID = e.CourseID);



19. Get the department with the lowest average CGPA

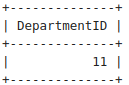
SELECT DepartmentID

FROM Students180

GROUP BY DepartmentID

ORDER BY AVG(CGPA)

ASC LIMIT 1;



20. Find the course in which the maximum number of students have failed (Marks < 40)

SELECT CourseID

FROM Enrollments180

WHERE Marks < 40

GROUP BY CourseID

ORDER BY COUNT(\*)

DESC LIMIT 1;

