Create Database Schema ;

1. CREATE TABLE Students ( student\_id Integer PRIMARY KEY, first\_name Text(100), last\_name Text(100), email Text(100), phone Text(20), date\_of\_birth DATE, enrolment\_date DATE, department\_id Integer, FOREIGN KEY (department\_id) REFERENCES Departments(department\_id) );

2.CREATE TABLE Courses ( course\_id Integer PRIMARY KEY, course\_name Text(100), department\_id Integer, professor\_id Integer, credits Integer, FOREIGN KEY (department\_id) REFERENCES Departments(department\_id), FOREIGN KEY (professor\_id) REFERENCES Professors(professor\_id) );

1. CREATE TABLE departments ( department\_id Integer PRIMARY KEY, department\_name Text(100) );

4.CREATE TABLE professors ( professor\_id Integer PRIMARY KEY, first\_name Text(100), last\_name Text(100), email Text(100), phone Text(20) );

5.CREATE TABLE enrollments ( enrollment\_id Integer PRIMARY KEY, student\_id Integer, course\_id Integer, enrollment\_date DATE, grade Text(5), FOREIGN KEY (student\_id) REFERENCES students(student\_id), FOREIGN KEY (course\_id) REFERENCES courses(course\_id) );

1.INSERT INTO students (student\_id, first\_name, last\_name, email, phone, date\_of\_birth, enrollment\_date, department\_id) VALUES (1, 'Ishita', 'Khandelwal', '[ishita.khandelwal@gmail.com](mailto:ishita.khandelwal@gmail.com)', '9876511111', '2002-03-18', '2020-07-15', 1), (2, 'Lakshay', 'Bansal', '[lakshay.bansal@gmail.com](mailto:lakshay.bansal@gmail.com)', '9876511112', '2000-11-25', '2019-07-20', 2), (3, 'Meera', 'Nair', '[meera.nair@gmail.com](mailto:meera.nair@gmail.com)', '9876511113', '2003-05-30', '2021-08-05', 3), (4, 'Harsh', 'Kapoor', '[harsh.kapoor@gmail.com](mailto:harsh.kapoor@gmail.com)', '9876511114', '2001-10-10', '2020-08-10', 4), (5, 'Tanya', 'Chatterjee', '[tanya.chatterjee@gmail.com](mailto:tanya.chatterjee@gmail.com)', '9876511115', '2000-12-05', '2019-07-25', 5), (6, 'Rohit', 'Shetty', '[rohit.shetty@gmail.com](mailto:rohit.shetty@gmail.com)', '9876511116', '2001-02-14', '2020-07-30', 6), (7, 'Simran', 'Kaur', '[simran.kaur@gmail.com](mailto:simran.kaur@gmail.com)', '9876511117', '1999-09-22', '2018-08-10', 7), (8, 'Aniket', 'Desai', '[aniket.desai@gmail.com](mailto:aniket.desai@gmail.com)', '9876511118', '2003-01-08', '2021-07-28', 8), (9, 'Nikita', 'Rana', '[nikita.rana@gmail.com](mailto:nikita.rana@gmail.com)', '9876511119', '2000-08-16', '2019-07-18', 9), (10, 'Kabir', 'Saxena', '[kabir.saxena@gmail.com](mailto:kabir.saxena@gmail.com)', '9876511120', '2002-06-12', '2020-07-22', 10);

2.INSERT INTO courses (course\_id, course\_name, department\_id, professor\_id, credits) VALUES (1, 'Data Structures', 1, 1, 4), (2, 'Operating Systems', 1, 1, 3), (3, 'Thermodynamics', 2, 2, 4), (4, 'Fluid Mechanics', 2, 2, 3), (5, 'Digital Circuits', 3, 3, 4), (6, 'Microprocessors', 3, 3, 3), (7, 'Machine Learning', 1, 4, 4), (8, 'Robotics', 2, 4, 3), (9, 'Power Systems', 3, 5, 3), (10, 'Embedded Systems', 3, 5, 3);

3.INSERT INTO departments (department\_id, department\_name) VALUES (1, 'Computer Science'), (2, 'Mechanical Engineering'), (3, 'Electrical Engineering');

4.INSERT INTO professors (professor\_id, first\_name, last\_name, email, phone) VALUES (1, 'Raj', 'Verma', '[raj.verma@gmail.com](mailto:rajesh.verma@gmail.com)', '9876543210'), (2, 'Neha', 'Patel', 'n[eha.patel@gmail.com](mailto:sneha.patel@gmail.com)', '9876543211'), (3, 'Aman', 'Sharma', '[aman.sharma@gmail.com](mailto:amit.sharma@gmail.com)', '9876543212');

5.INSERT INTO enrollments (enrollment\_id, student\_id, course\_id, enrollment\_date, grade) VALUES (1, 1, 1, '2020-07-15', 'A'), (2, 2, 2, '2019-07-20', 'B'), (3, 3, 4, '2021-08-05', 'A'), (4, 4, 5, '2020-08-10', 'B+'), (5, 5, 5, '2019-07-25', 'A-'), (6, 6, 4, '2020-07-30', 'A'), (7, 7, 8, '2018-08-10', 'B'), (8, 8, 9, '2021-07-28', 'B+'), (9, 9, 4, '2019-08-18', 'A-'), (10, 10, 1, '2020-07-22', 'B+');

# **SQL quires question**

1. Find the Total Number of Students in Each Department

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

SELECT c.course\_id, c.course\_name, d.department\_name FROM courses c JOIN departments d ON c.department\_id = d.department\_id WHERE c.professor\_id = 1;

1. Find the Average Grade of Students in Each Course

SELECT c.course\_name, AVG( CASE e.grade WHEN 'A+' THEN 10 WHEN 'A' THEN 9 WHEN 'A-' THEN 8 WHEN 'B+' THEN 7 WHEN 'B' THEN 6 WHEN 'C' THEN 5 ELSE 0 END ) 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

SELECT s.student\_id, s.first\_name, s.last\_name, s.email FROM students s LEFT JOIN enrollments e ON s.student\_id = e.student\_id WHERE e.enrollment\_id IS NULL;

1. Find the Number of Courses Offered by Each Department

SELECT d.department\_name, COUNT(c.course\_id) AS total\_courses FROM courses c JOIN departments d ON c.department\_id = d.department\_id GROUP BY d.department\_name;

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

SELECT s.student\_id, s.first\_name, s.last\_name, s.email 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 = 'Data Structures';

1. Find the Most Popular Course Based on Enrollment Numbers

SELECT c.course\_id, c.course\_name, COUNT(e.enrollment\_id) AS total\_enrollments FROM enrollments e JOIN courses c ON e.course\_id = c.course\_id GROUP BY c.course\_id, c.course\_name ORDER BY total\_enrollments DESC LIMIT 3;

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

SELECT d.department\_name, AVG(student\_credits.total\_credits) AS avg\_credits FROM ( SELECT s.student\_id, s.department\_id, SUM(c.credits) AS total\_credits FROM students s JOIN enrollments e ON s.student\_id = e.student\_id JOIN courses c ON e.course\_id = c.course\_id GROUP BY s.student\_id, s.department\_id ) AS student\_credits JOIN departments d ON student\_credits.department\_id = d.department\_id GROUP BY d.department\_name;

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

SELECT p.professor\_id, p.first\_name, p.last\_name, COUNT(DISTINCT c.department\_id) AS department\_count FROM professors p JOIN courses c ON p.professor\_id = c.professor\_id GROUP BY p.professor\_id, p.first\_name, p.last\_name HAVING COUNT(DISTINCT c.department\_id) > 1;

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

SELECT MAX( CASE e.grade WHEN 'A+' THEN 10 WHEN 'A' THEN 9 WHEN 'A-' THEN 8 WHEN 'B+' THEN 7 WHEN 'B' THEN 6 WHEN 'C' THEN 5 ELSE 0 END ) AS highest\_grade\_point, MIN( CASE e.grade WHEN 'A+' THEN 10 WHEN 'A' THEN 9 WHEN 'A-' THEN 8 WHEN 'B+' THEN 7 WHEN 'B' THEN 6 WHEN 'C' THEN 5 ELSE 0 END ) AS lowest\_grade\_point FROM enrollments e JOIN courses c ON e.course\_id = c.course\_id WHERE c.course\_name = 'Data Structures';