Task-1

create database task1

1.Department table

CREATE TABLE Departments (

department\_id INT PRIMARY KEY,

department\_name VARCHAR(100)

);

INSERT INTO Departments (department\_id, department\_name)

VALUES

(1, 'Computer Science'),

(2, 'Mechanical Engineering'),

(3, 'Electrical Engineering'),

(4, 'Civil Engineering'),

(5, 'Biotechnology'),

(6, 'Chemical Engineering'),

(7, 'Aerospace Engineering'),

(8, 'Civil and Structural Engineering'),

(9, 'Environmental Science'),

(10, 'Physics');

===========================================================================================

2.Professors table

CREATE TABLE Professors (

professor\_id INT PRIMARY KEY,

first\_name VARCHAR(100),

last\_name VARCHAR(100),

email VARCHAR(100),

phone VARCHAR(20)

);

INSERT INTO Professors (professor\_id, first\_name, last\_name, email, phone)

VALUES

(1, 'Amit', 'Sharma', 'amit.sharma@university.com', '9876543210'),

(2, 'Rita', 'Patel', 'rita.patel@university.com', '9876543211'),

(3, 'Suresh', 'Verma', 'suresh.verma@university.com', '9876543212'),

(4, 'Pooja', 'Reddy', 'pooja.reddy@university.com', '9876543213'),

(5, 'Vikram', 'Singh', 'vikram.singh@university.com', '9876543214'),

(6, 'Anjali', 'Gupta', 'anjali.gupta@university.com', '9876543215'),

(7, 'Rajesh', 'Kumar', 'rajesh.kumar@university.com', '9876543216'),

(8, 'Sunita', 'Joshi', 'sunita.joshi@university.com', '9876543217'),

(9, 'Arun', 'Nair', 'arun.nair@university.com', '9876543218'),

(10, 'Neha', 'Chawla', 'neha.chawla@university.com', '9876543219');

===========================================================================================

3.student table

CREATE TABLE Students (

student\_id INT PRIMARY KEY,

first\_name VARCHAR(100),

last\_name VARCHAR(100),

email VARCHAR(100),

phone VARCHAR(20),

date\_of\_birth DATE,

enrollment\_date DATE,

department\_id INT,

FOREIGN KEY (department\_id) REFERENCES Departments(department\_id)

);

INSERT INTO Students (student\_id, first\_name, last\_name, email, phone, date\_of\_birth, enrollment\_date, department\_id)

VALUES

(1, 'Aarav', 'Sharma', 'aarav.sharma@student.com', '9876543210', '2000-05-15', '2023-08-01', 1),

(2, 'Priya', 'Patel', 'priya.patel@student.com', '9876543211', '1999-09-20', '2022-08-15', 2),

(3, 'Rahul', 'Kumar', 'rahul.kumar@student.com', '9876543212', '1998-12-10', '2021-08-01', 3),

(4, 'Ananya', 'Verma', 'ananya.verma@student.com', '9876543213', '2001-03-25', '2024-01-10', 4),

(5, 'Vikram', 'Singh', 'vikram.singh@student.com', '9876543214', '1997-07-08', '2020-07-25', 5),

(6, 'Neha', 'Gupta', 'neha.gupta@student.com', '9876543215', '2000-11-22', '2023-06-15', 1),

(7, 'Arjun', 'Reddy', 'arjun.reddy@student.com', '9876543216', '1998-04-05', '2021-09-30', 2),

(8, 'Isha', 'Malhotra', 'isha.malhotra@student.com', '9876543217', '1999-02-14', '2022-03-25', 3),

(9, 'Karan', 'Jadhav', 'karan.jadhav@student.com', '9876543218', '2002-01-19', '2024-02-05', 4),

(10, 'Meera', 'Nair', 'meera.nair@student.com', '9876543219', '2001-08-30', '2023-07-22', 5);

============================================================================================================

4.courses table

CREATE TABLE Courses (

course\_id INT PRIMARY KEY,

course\_name VARCHAR(100),

department\_id INT,

professor\_id INT,

credits INT,

FOREIGN KEY (department\_id) REFERENCES Departments(department\_id),

FOREIGN KEY (professor\_id) REFERENCES Professors(professor\_id)

);

INSERT INTO Courses (course\_id, course\_name, department\_id, professor\_id, credits)

VALUES

(1, 'Data Structures', 1, 1, 3),

(2, 'Thermodynamics', 2, 2, 4),

(3, 'Circuit Analysis', 3, 3, 3),

(4, 'Building Construction', 4, 4, 3),

(5, 'Genetic Engineering', 5, 5, 4),

(6, 'Operating Systems', 1, 6, 3),

(7, 'Fluid Mechanics', 2, 7, 4),

(8, 'Power Systems', 3, 8, 3),

(9, 'Environmental Engineering', 4, 9, 3),

(10, 'Microbiology', 5, 10, 4);

====================================================================================================================

5.Enrollments table

CREATE TABLE Enrollments (

enrollment\_id INT PRIMARY KEY,

student\_id INT,

course\_id INT,

enrollment\_date DATE,

grade VARCHAR(5),

FOREIGN KEY (student\_id) REFERENCES Students(student\_id),

FOREIGN KEY (course\_id) REFERENCES Courses(course\_id)

);

INSERT INTO Enrollments (enrollment\_id, student\_id, course\_id, enrollment\_date, grade)

VALUES

(1, 1, 1, '2023-08-01', 'A'),

(2, 2, 2, '2022-08-15', 'B'),

(3, 3, 3, '2021-08-01', 'C'),

(4, 4, 4, '2024-01-10', 'A'),

(5, 5, 5, '2020-07-25', 'B'),

(6, 6, 1, '2023-06-15', 'A'),

(7, 7, 2, '2021-09-30', 'B'),

(8, 8, 3, '2022-03-25', 'A'),

(9, 9, 4, '2024-02-05', 'C'),

(10, 10, 5, '2023-07-22', 'A');

=============================================================================

1. Find the Total Number of Students in Each Department

SELECT DepartmentID, COUNT(\*) AS TotalStudents

FROM Students

GROUP BY DepartmentID;

============================================================

2. List All Courses Taught by a Specific Professor

SELECT c.course\_id,c.course\_name, d.department\_name, c.credits FROM Courses c JOIN Departments d ON C.department\_id = d.department\_id

WHERE c.professor\_id = professor\_id;

=================================================

3. Find the Average Grade of Students in Each Course

SELECT

c.course\_name,

AVG(CASE

WHEN e.grade = 'A' THEN 4.0

WHEN e.grade = 'B' THEN 3.0

WHEN e.grade = 'C' THEN 2.0

WHEN e.grade = 'D' THEN 1.0

WHEN e.grade = 'F' THEN 0.0

ELSE NULL

END) AS average\_grade

FROM

enrollments e

JOIN

courses c ON e.course\_id = c.course\_id

GROUP BY

c.course\_name;

==========================================================================

4. List All Students Who Have Not Enrolled in Any Courses

select s.student\_id,

s.first\_name,

s.last\_name,

s.email,

s.phone,

s.date\_of\_birth,

s.enrollment\_date,

d.department\_name

FROM

students s

LEFT JOIN

enrollments e ON s.student\_id = e.student\_id

LEFT JOIN

Department d ON s.department\_id = d.department\_id

WHERE

e.enrollments\_id IS NULL;

====================================================================================

5. 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;

================================================================================

6. 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 = 'Database Systems';

==================================================================================

7. Find the Most Popular Course Based on Enrollment Numbers

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;

====================================================================================

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

SELECT d.department\_name, AVG(c.credits) AS avg\_credits\_per\_student

FROM Students s

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

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

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

GROUP BY d.department\_name;

==================================================================================

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

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

HAVING COUNT(DISTINCT c.department\_id) > 1;

================================================================================================

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

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';

=================================================================================