جامعة ٦ أكتوبر
October 6 University

علوم الحاسب ونظم المعلومات
Info Sys & Comp Science

Advanced Database Project

“Faculty Management System”

Course instructor:

DR. Randa Alwakeel

Supervision of:

Eng. Doaa Bliedy

Prepared by

Name	ID	Grade
Moataz Ibrahim Gaber	202124938	
Ali Farouk Ali	202116264	
Mario Emad Edward	202116342	
Omar Mohamed Ahmed	202122997	
Nancy Khaled Sayed	202124689	

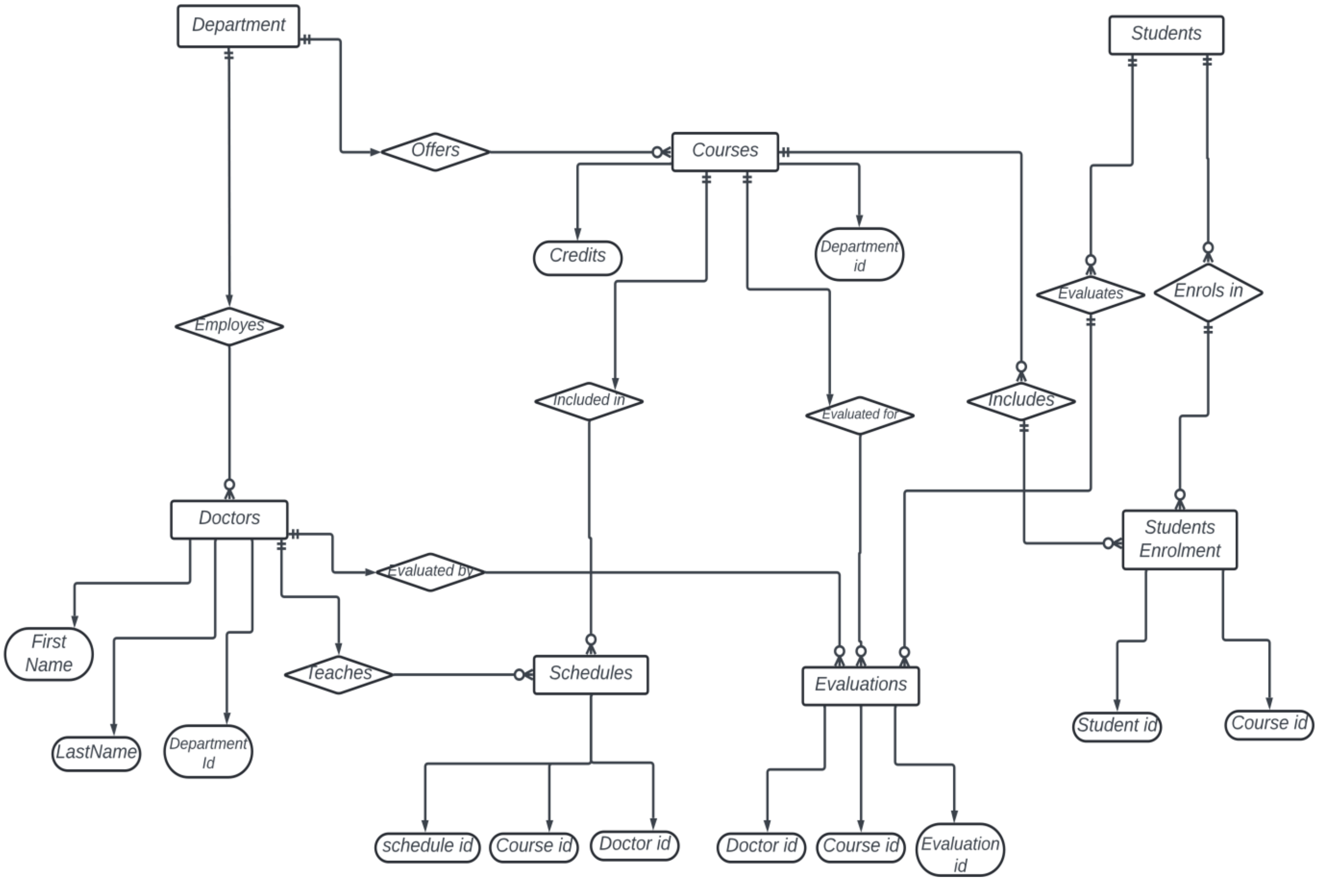
Table of contents:

INTRODUCTION

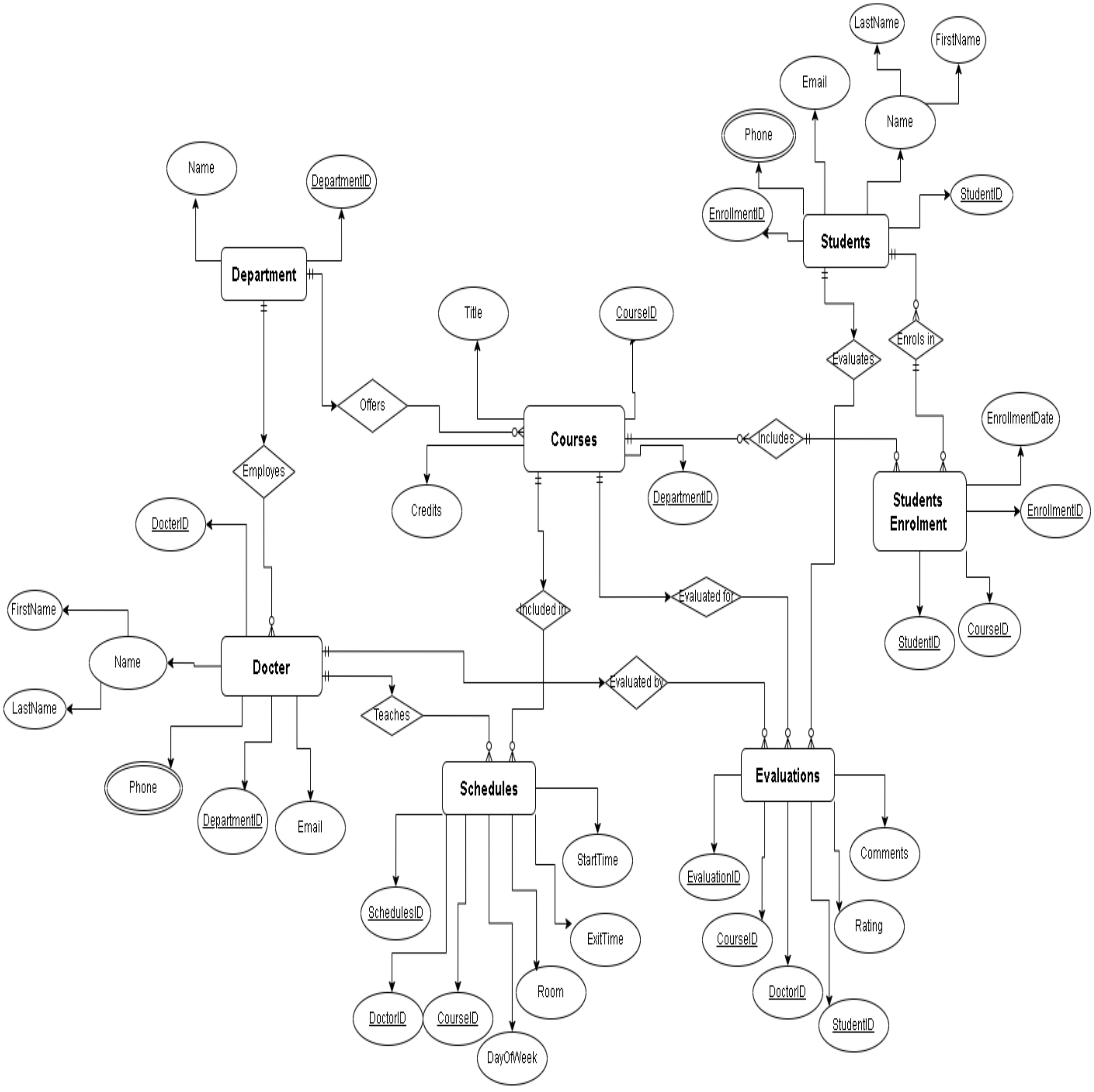
- The Faculty Management System is a software solution designed to simplify and streamline administrative tasks within educational institutions. It provides a centralized platform for managing faculty information, course details, attendance records, grading, communication, resource allocation, and reporting. By automating these processes, the system enhances efficiency, communication, and accountability, ultimately improving the overall management of academic operations.

- The Faculty Management System is user-friendly software designed to handle the day-to-day administrative tasks within educational institutions. It helps manage faculty information, course details, attendance records, grading, and communication channels efficiently. With its intuitive interface and streamlined processes, the system aims to enhance organization, communication, and overall effectiveness in managing faculty-related operations.

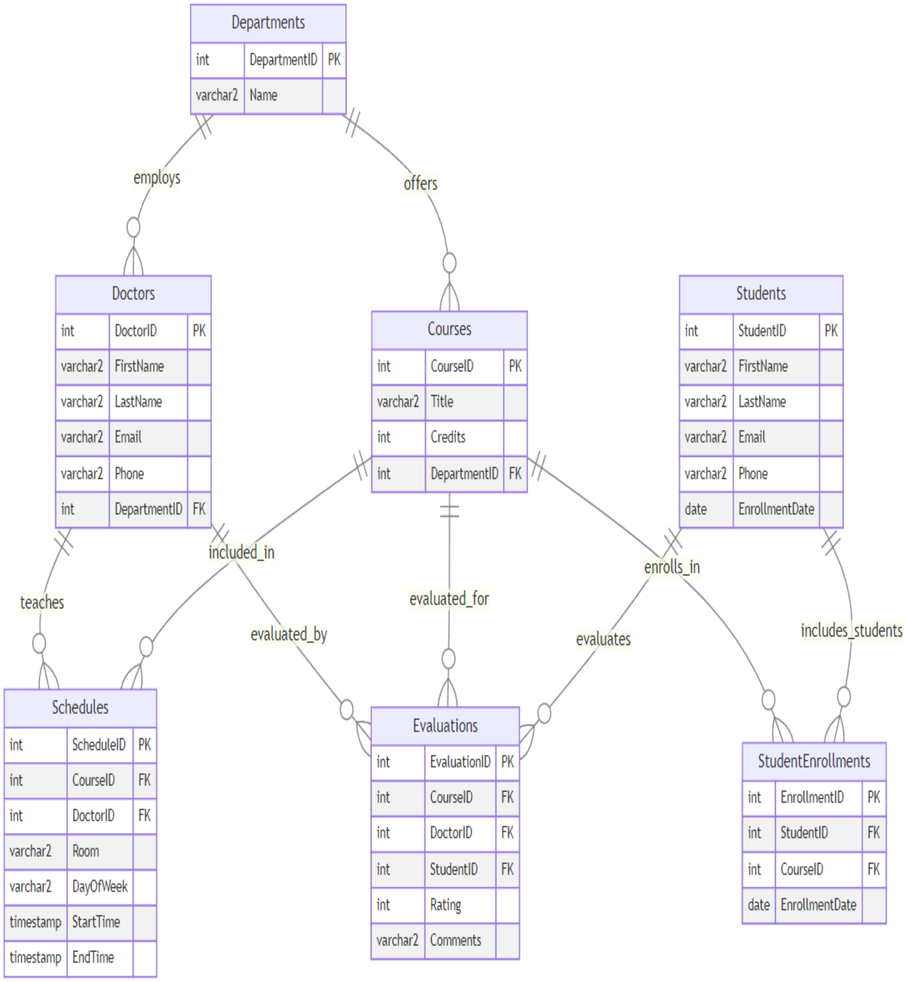
ERD


Department
Students
Offers
Courses
Credits
Department id
Evaluates
Enrols in
Employes
Included in
Evaluated for
Includes
Doctors
Students Enrolment
Evaluated by
First Name
Teaches
Schedules
Evaluations
LastName
Department Id
Student id
Course id
schedule id
Course id
Doctor id
Doctor id
Course id
Evaluation id

Enhanced ERD



RDM



Departments		
int	DepartmentID	PK
varchar2	Name	

Doctors		
int	DoctorID	PK
varchar2	FirstName	
varchar2	LastName	
varchar2	Email	
varchar2	Phone	
int	DepartmentID	FK

Courses		
int	CourseID	PK
varchar2	Title	
int	Credits	
int	DepartmentID	FK

Students		
int	StudentID	PK
varchar2	FirstName	
varchar2	LastName	
varchar2	Email	
varchar2	Phone	
date	EnrollmentDate	

Schedules		
int	ScheduleID	PK
int	CourseID	FK
int	DoctorID	FK
varchar2	Room	
varchar2	DayOfWeek	
timestamp	StartTime	
timestamp	EndTime	

Evaluations		
int	EvaluationID	PK
int	CourseID	FK
int	DoctorID	FK
int	StudentID	FK
int	Rating	
varchar2	Comments	

StudentEnrollments		
int	EnrollmentID	PK
int	StudentID	FK
int	CourseID	FK
date	EnrollmentDate	

Code:

- ## DDL (Create):

```
CREATE TABLE Departments_ (
    DepartmentID INT PRIMARY KEY,
    Name VARCHAR2(100) NOT NULL
);
CREATE TABLE Doctors (
    DoctorID INT PRIMARY KEY,
    FirstName VARCHAR2(50),
    LastName VARCHAR2(50),
    Email VARCHAR2(100),
    Phone VARCHAR2(15),
    DepartmentID INT,
    FOREIGN KEY (DepartmentID) REFERENCES Departments_(DepartmentID)
);
CREATE TABLE Courses (
    CourseID INT PRIMARY KEY,
    Title VARCHAR2(100) NOT NULL,
    Credits INT,
    DepartmentID INT,
    FOREIGN KEY (DepartmentID) REFERENCES Departments_(DepartmentID)
);
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    FirstName VARCHAR2(50),
    LastName VARCHAR2(50),
    Email VARCHAR2(100),
    Phone VARCHAR2(15),
    EnrollmentDate DATE
);
CREATE TABLE Schedules (
    ScheduleID INT PRIMARY KEY,
    CourseID INT,
    DoctorID INT,
    Room VARCHAR2(50),
    DayOfWeek VARCHAR2(10),
    StartTime TIMESTAMP,
    EndTime TIMESTAMP,
    FOREIGN KEY (CourseID) REFERENCES Courses(CourseID),
    FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID)
);
CREATE TABLE Evaluations (
    EvaluationID INT PRIMARY KEY,
    CourseID INT,
    DoctorID INT,
    StudentID INT,
    Rating INT CHECK (Rating >= 1 AND Rating <= 5),
    Comments VARCHAR2(500),
    FOREIGN KEY (CourseID) REFERENCES Courses(CourseID),
    FOREIGN KEY (DoctorID) REFERENCES Doctors(DoctorID),
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID)
);
CREATE TABLE StudentEnrollments (
    EnrollmentID INT PRIMARY KEY,
    StudentID INT,
    CourseID INT,
    EnrollmentDate DATE,
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
);
```

- Insert Values:

```
➢ INSERT ALL
INTO Departments_ (DepartmentID, Name) VALUES (1, 'Computer Science')
INTO Departments_ (DepartmentID, Name) VALUES (2, 'Medicine')
INTO Departments_ (DepartmentID, Name) VALUES (3, 'Business Administration')
INTO Departments_ (DepartmentID, Name) VALUES (4, 'Engineering')
INTO Departments_ (DepartmentID, Name) VALUES (5, 'English')
INTO Departments_ (DepartmentID, Name) VALUES (6, 'Mathematics')
INTO Departments_ (DepartmentID, Name) VALUES (7, 'Psychology')
INTO Departments_ (DepartmentID, Name) VALUES (8, 'History')
INTO Departments_ (DepartmentID, Name) VALUES (9, 'Art')
INTO Departments_ (DepartmentID, Name) VALUES (10, 'Biology')
SELECT * FROM dual;
```

```
➢ INSERT ALL
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (1, 'John', 'Smith',
'john.smith@example.com', '123-456-7890', 2)
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (2, 'Emily', 'Johnson',
'emily.johnson@example.com', '987-654-3210', 2)
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (3, 'Michael', 'Williams',
'michael.williams@example.com', '555-555-5555', 2)
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (4, 'Jessica', 'Brown',
'jessica.brown@example.com', '111-222-3333', 2)
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (5, 'David', 'Jones',
'david.jones@example.com', '444-444-4444', 3)
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (6, 'Sarah', 'Davis',
'sarah.davis@example.com', '666-666-6666', 3)
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (7, 'Daniel', 'Miller',
'daniel.miller@example.com', '777-777-7777', 1)
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (8, 'Lisa', 'Wilson',
'lisa.wilson@example.com', '888-888-8888', 1)
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (9, 'Robert', 'Taylor',
'robert.taylor@example.com', '999-999-9999', 3)
INTO Doctors (DoctorID, FirstName, LastName, Email, Phone, DepartmentID) VALUES (10, 'Jennifer', 'Martinez',
'jennifer.martinez@example.com', '000-000-0000', 4)
SELECT * FROM dual;
```

```
➢ INSERT ALL
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (1, 'Introduction to Programming', 3, 1)
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (2, 'Anatomy and Physiology', 4, 2)
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (3, 'Marketing Principles', 3, 3)
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (4, 'Mechanical Engineering Fundamentals', 4, 4)
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (5, 'English Composition', 3, 5)
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (6, 'Calculus I', 4, 6)
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (7, 'Introduction to Psychology', 3, 7)
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (8, 'World History', 3, 8)
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (9, 'Art History', 3, 9)
INTO Courses (CourseID, Title, Credits, DepartmentID) VALUES (10, 'Cell Biology', 4, 10)
SELECT * FROM dual;
```

- INSERT ALL

```
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (1, 'Emma', 'Johnson',
'emma.johnson@example.com', '123-456-7890', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (2, 'Matthew', 'Williams',
'matthew.williams@example.com', '987-654-3210', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (3, 'Olivia', 'Brown',
'olivia.brown@example.com', '555-555-5555', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (4, 'Daniel', 'Smith',
'daniel.smith@example.com', '111-222-3333', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (5, 'Sophia', 'Jones',
'sophia.jones@example.com', '444-444-4444', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (6, 'Alexander', 'Davis',
'alexander.davis@example.com', '666-666-6666', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (7, 'Mia', 'Miller',
'mia.miller@example.com', '777-777-7777', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (8, 'Ethan', 'Wilson',
'ethan.wilson@example.com', '888-888-8888', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (9, 'Charlotte', 'Taylor',
'charlotte.taylor@example.com', '999-999-9999', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO Students (StudentID, FirstName, LastName, Email, Phone, EnrollmentDate) VALUES (10, 'Liam', 'Martinez',
'liam.martinez@example.com', '000-000-0000', TO_DATE('2023-09-01', 'YYYY-MM-DD'))
SELECT * FROM dual;
```

- INSERT ALL

```
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (1, 1, 1, 'Room A', 'Monday',
TO_TIMESTAMP('08:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('10:00:00', 'HH24:MI:SS'))
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (2, 2, 2, 'Room B', 'Tuesday',
TO_TIMESTAMP('09:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('11:00:00', 'HH24:MI:SS'))
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (3, 3, 3, 'Room C',
'Wednesday', TO_TIMESTAMP('10:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('12:00:00', 'HH24:MI:SS'))
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (4, 4, 4, 'Room D', 'Thursday',
TO_TIMESTAMP('11:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('13:00:00', 'HH24:MI:SS'))
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (5, 5, 5, 'Room E', 'Friday',
TO_TIMESTAMP('12:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('14:00:00', 'HH24:MI:SS'))
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (6, 6, 6, 'Room F', 'Monday',
TO_TIMESTAMP('13:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('15:00:00', 'HH24:MI:SS'))
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (7, 7, 7, 'Room G', 'Tuesday',
TO_TIMESTAMP('14:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('16:00:00', 'HH24:MI:SS'))
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (8, 8, 8, 'Room H',
'Wednesday', TO_TIMESTAMP('15:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('17:00:00', 'HH24:MI:SS'))
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (9, 9, 9, 'Room I', 'Thursday',
TO_TIMESTAMP('16:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('18:00:00', 'HH24:MI:SS'))
INTO Schedules (ScheduleID, CourseID, DoctorID, Room, DayOfWeek, StartTime, EndTime) VALUES (10, 10, 10, 'Room J', 'Friday',
TO_TIMESTAMP('17:00:00', 'HH24:MI:SS'), TO_TIMESTAMP('19:00:00', 'HH24:MI:SS'))
SELECT * FROM dual;
```

- INSERT ALL

```
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (1, 1, 1, 1, 4, 'Great course overall.')
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (2, 2, 2, 2, 5, 'The instructor was very knowledgeable.')
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (3, 3, 3, 3, 3, 'The course material could be more engaging.')
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (4, 4, 4, 4, 4, 'Enjoyed the hands-on projects.')
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (5, 5, 5, 5, 4, 'Challenging but rewarding.')
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (6, 6, 6, 6, 5, 'Excellent teaching style.')
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (7, 7, 7, 7, 3, 'Expected more interaction in class.')
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (8, 8, 8, 8, 4, 'Interesting content.')
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (9, 9, 9, 9, 5, 'Instructor provided valuable insights.')
INTO Evaluations (EvaluationID, CourseID, DoctorID, StudentID, Rating, Comments) VALUES (10, 10, 10, 10, 4, 'Would recommend to others.')
SELECT * FROM dual;
```

- INSERT ALL

```
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (1, 1, 1, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (2, 2, 2, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (3, 3, 3, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (4, 4, 4, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (5, 5, 5, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (6, 6, 6, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (7, 7, 7, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (8, 8, 8, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (9, 9, 9, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
INTO StudentEnrollments (EnrollmentID, StudentID, CourseID, EnrollmentDate) VALUES (10, 10, 10, TO_DATE('2023-09-01', 'YYYY-MM-DD'))
SELECT * FROM dual;
```

• Queries.

1. Retrieve all departments along with the total number of doctors in each department:

```
SELECT Departments_.*, TotalDoctors.Total
FROM Departments_
LEFT JOIN (
   SELECT DepartmentID, COUNT(DoctorID) AS Total
   FROM Doctors
   GROUP BY DepartmentID
) TotalDoctors ON Departments_.DepartmentID = TotalDoctors.DepartmentID;
```

2. Retrieve all courses along with the department name for each course:

```
SELECT Courses.*, Departments_.Name AS DepartmentName
FROM Courses
JOIN Departments_ ON Courses.DepartmentID = Departments_.DepartmentID;
```

3. Retrieve the courses with the highest number of enrolled students along with the count of enrolled students:

```
SELECT CourseID, COUNT(StudentID) AS EnrolledStudents
FROM StudentEnrollments
GROUP BY CourseID
ORDER BY EnrolledStudents DESC;
```

4. Retrieve all evaluations along with the doctor's full name and the course title:

```
SELECT Evaluations.*, Doctors.FirstName || ' ' || Doctors.LastName AS DoctorFullName, Courses.Title AS CourseTitle
FROM Evaluations
JOIN Doctors ON Evaluations.DoctorID = Doctors.DoctorID
JOIN Courses ON Evaluations.CourseID = Courses.CourseID;
```

5. Retrieve all students who haven't enrolled in any course:

```
SELECT Students.*
FROM Students
WHERE StudentID NOT IN (SELECT StudentID FROM StudentEnrollments);
```

6. Retrieve the courses offered by the Medicine department along with the number of enrolled students for each course:

```
SELECT Courses.*, COUNT(StudentEnrollments.StudentID) AS EnrolledStudents
FROM Courses
LEFT JOIN StudentEnrollments ON Courses.CourseID = StudentEnrollments.CourseID
WHERE Courses.DepartmentID = 2
GROUP BY Courses.CourseID, Courses.Title, Courses.Credits, Courses.DepartmentID;
```

7. Retrieve all evaluations with ratings above 3 along with the course title and the student's full name:

```
SELECT Evaluations.*, Courses.Title AS CourseTitle, Students.FirstName || ' ' || Students.LastName AS StudentFullName
FROM Evaluations
JOIN Courses ON Evaluations.CourseID = Courses.CourseID
JOIN Students ON Evaluations.StudentID = Students.StudentID
WHERE Evaluations.Rating > 3;
```

8. Retrieve the average rating of each doctor along with their full name:

```
SELECT Doctors.DoctorID, Doctors.FirstName || ' ' || Doctors.LastName AS DoctorFullName, AVG(Evaluations.Rating) AS
AvgRating
FROM Doctors
```

```
LEFT JOIN Evaluations ON Doctors.DoctorID = Evaluations.DoctorID
GROUP BY Doctors.DoctorID, Doctors.FirstName || ' ' || Doctors.LastName;
```

9. Retrieve all schedules for courses taught by doctors in the Psychology department:

```
SELECT Schedules.*, Courses.Title AS CourseTitle, Doctors.FirstName || ' ' || Doctors.LastName AS DoctorFullName
FROM Schedules
JOIN Courses ON Schedules.CourseID = Courses.CourseID
JOIN Doctors ON Schedules.DoctorID = Doctors.DoctorID
WHERE Doctors.DepartmentID = 7;
```

10. Retrieve all courses with no evaluations along with the department name for each course:

```
SELECT Courses.*, Departments_.Name AS DepartmentName
FROM Courses
JOIN Departments_ ON Courses.DepartmentID = Departments_.DepartmentID
WHERE Courses.CourseID NOT IN (SELECT CourseID FROM Evaluations);
```
