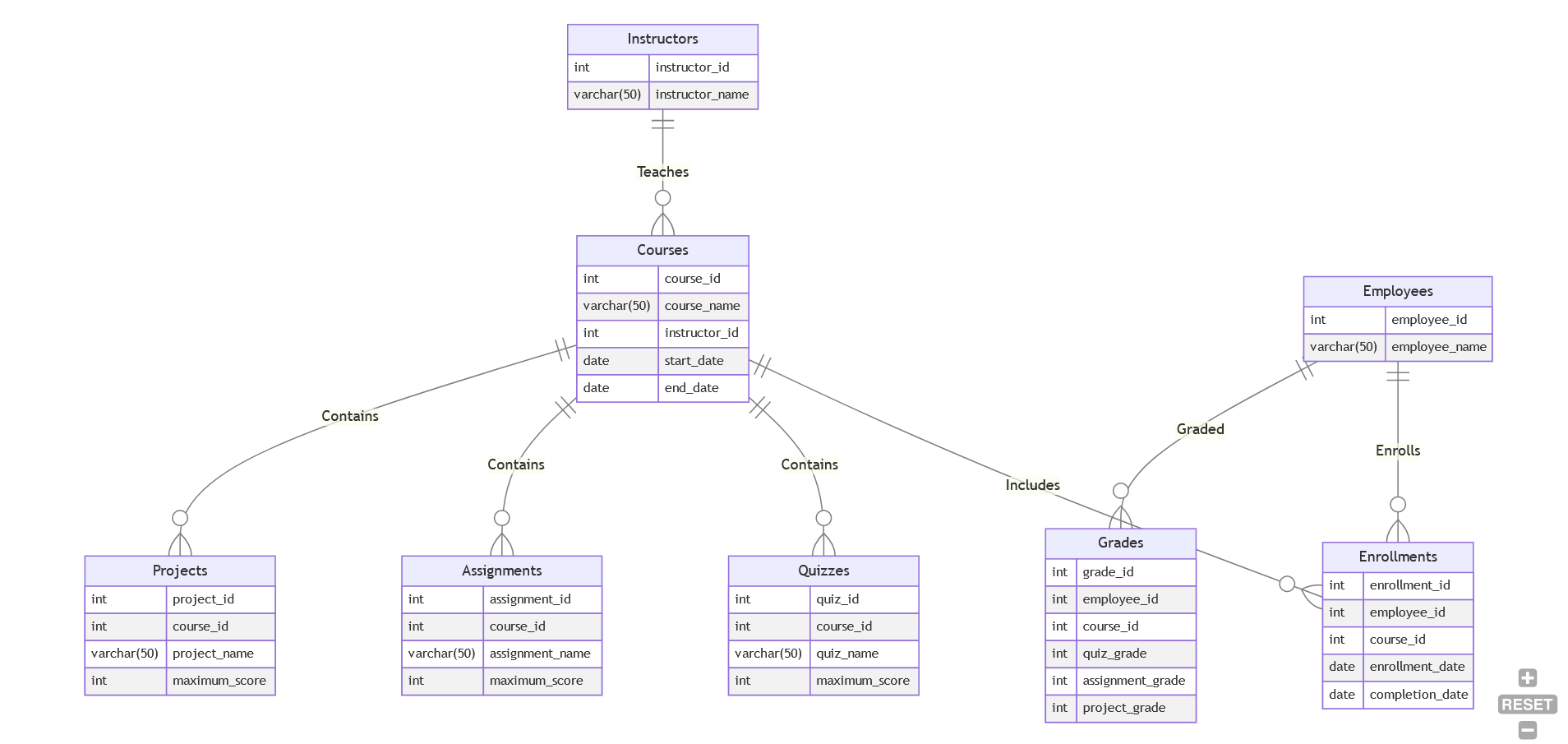
**PROJECT**

**SUBMITTED TO: MA’AM HUMAIRA JABEEN**

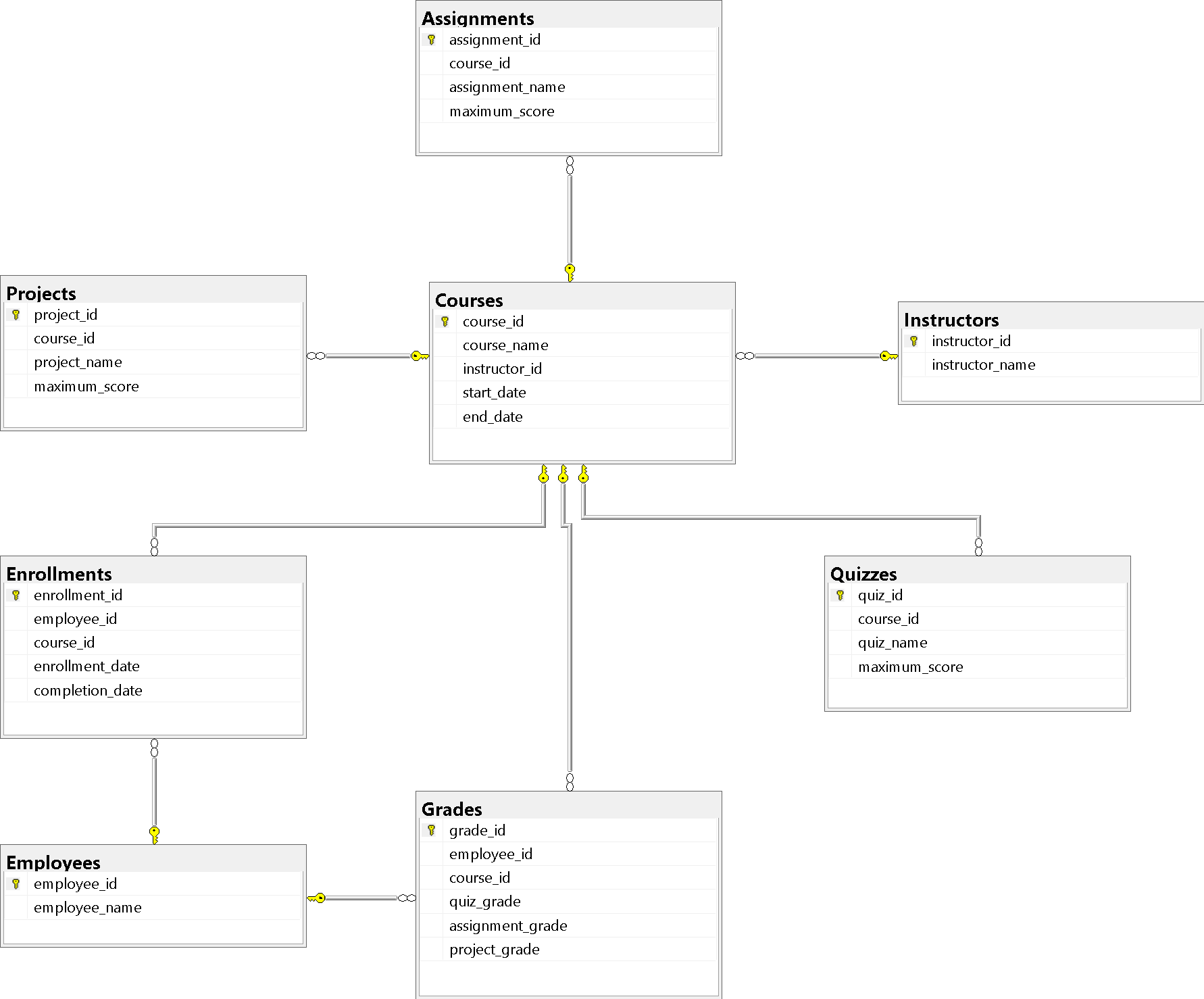
DATABASE SYSTEM 1

SYED SHAH HUSSIAN (FA21-bse-172)  
EISSA MASOOD (FA21-bse-086)  
EBADAT NISSA (FA21-bse-085)



**ERD**  
**CARDINALITY:**

**PRIMARY KEY + FOREIGY KEYS:**



-- Create the Instructors table

CREATE TABLE Instructors (

instructor\_id INT PRIMARY KEY,

instructor\_name VARCHAR(50)

);

INSERT INTO Instructors (instructor\_id, instructor\_name)

VALUES

(1, 'Ali Khan'),

(2, 'Fatima Ahmed'),

(3, 'Mohammad Khan'),

(4, 'Sara Malik'),

(5, 'Ahmed Ali'),

(6, 'Ayesha Khan'),

(7, 'Usman Ahmed'),

(8, 'Farah Hussain'),

(9, 'Abdullah Malik'),

(10, 'Sadia Ahmed');

-- Create the Employees table

CREATE TABLE Employees (

employee\_id INT PRIMARY KEY,

employee\_name VARCHAR(50)

);

INSERT INTO Employees (employee\_id, employee\_name)

VALUES

(1, 'Mohammad Khan'),

(2, 'Fatima Ali'),

(3, 'Ahmed Malik'),

(4, 'Ayesha Ahmed'),

(5, 'Ali Khan'),

(6, 'Sara Malik'),

(7, 'Usman Ahmed'),

(8, 'Farah Khan'),

(9, 'Abdullah Malik'),

(10, 'Sadia Ahmed');

-- Create the Courses table

CREATE TABLE Courses (

course\_id INT PRIMARY KEY,

course\_name VARCHAR(50),

instructor\_id INT,

start\_date DATE,

end\_date DATE,

FOREIGN KEY (instructor\_id) REFERENCES Instructors(instructor\_id)

);

INSERT INTO Courses (course\_id, course\_name, instructor\_id, start\_date, end\_date)

VALUES

(1, 'Course 1', 1, '2023-01-01', '2023-02-01'),

(2, 'Course 2', 1, '2023-02-01', '2023-03-01'),

(3, 'Course 3', 3, '2023-03-01', '2023-04-01'),

(4, 'Course 4', 3, '2023-04-01', '2023-05-01'),

(5, 'Course 5', 5, '2023-05-01', '2023-06-01'),

(6, 'Course 6', 5, '2023-06-01', '2023-07-01'),

(7, 'Course 7', 6, '2023-07-01', '2023-08-01'),

(8, 'Course 8', 7, '2023-08-01', '2023-09-01'),

(9, 'Course 9', 7, '2023-09-01', '2023-10-01'),

(10, 'Course 10', 10, '2023-10-01', '2023-11-01'),

(11, 'Course 11', 2, '2023-10-01', '2023-11-01'),

(12, 'Course 12', 4, '2023-10-01', '2023-11-01'),

(13, 'Course 13', 8, '2023-10-01', '2023-11-01'),

(14, 'Course 14', 9, '2023-10-01', '2023-11-01');

-- Create the Enrollments table

CREATE TABLE Enrollments (

enrollment\_id INT PRIMARY KEY,

employee\_id INT,

course\_id INT,

enrollment\_date DATE,

completion\_date DATE,

FOREIGN KEY (employee\_id) REFERENCES Employees(employee\_id),

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

);

INSERT INTO Enrollments (enrollment\_id, employee\_id, course\_id, enrollment\_date, completion\_date)

VALUES

(1, 1, 1, '2023-01-01', '2023-01-15'),

(2, 2, 2, '2023-02-01', '2023-02-15'),

(3, 3, 3, '2023-03-01', '2023-03-15'),

(4, 4, 4, '2023-04-01', '2023-04-15'),

(5, 5, 5, '2023-05-01', '2023-05-15'),

(6, 6, 6, '2023-06-01', '2023-06-15'),

(7, 7, 7, '2023-07-01', '2023-07-15'),

(8, 8, 8, '2023-08-01', '2023-08-15'),

(9, 9, 9, '2023-09-01', '2023-09-15'),

(10, 10, 10, '2023-10-01', '2023-10-15');

-- Create the Quizzes table

CREATE TABLE Quizzes (

quiz\_id INT PRIMARY KEY,

course\_id INT,

quiz\_name VARCHAR(50),

maximum\_score INT,

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

);

INSERT INTO Quizzes (quiz\_id, course\_id, quiz\_name, maximum\_score)

VALUES

(1, 1, 'Quiz 1', 10),

(2, 2, 'Quiz 2', 10),

(3, 3, 'Quiz 3', 10),

(4, 4, 'Quiz 4', 10),

(5, 5, 'Quiz 5', 10),

(6, 6, 'Quiz 6', 10),

(7, 7, 'Quiz 7', 10),

(8, 8, 'Quiz 8', 10),

(9, 9, 'Quiz 9', 10),

(10, 10, 'Quiz 10', 10);

-- Create the Assignments table

CREATE TABLE Assignments (

assignment\_id INT PRIMARY KEY,

course\_id INT,

assignment\_name VARCHAR(50),

maximum\_score INT,

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

);

INSERT INTO Assignments (assignment\_id, course\_id, assignment\_name, maximum\_score)

VALUES

(1, 1, 'Assignment 1', 10),

(2, 2, 'Assignment 2', 10),

(3, 3, 'Assignment 3', 10),

(4, 4, 'Assignment 4', 10),

(5, 5, 'Assignment 5', 10),

(6, 6, 'Assignment 6', 10),

(7, 7, 'Assignment 7', 10),

(8, 8, 'Assignment 8', 10),

(9, 9, 'Assignment 9', 10),

(10, 10, 'Assignment 10', 10);

-- Create the Projects table

CREATE TABLE Projects (

project\_id INT PRIMARY KEY,

course\_id INT,

project\_name VARCHAR(50),

maximum\_score INT,

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

);

INSERT INTO Projects (project\_id, course\_id, project\_name, maximum\_score)

VALUES

(1, 1, 'Project 1', 100),

(2, 2, 'Project 2', 100),

(3, 3, 'Project 3', 100),

(4, 4, 'Project 4', 100),

(5, 5, 'Project 5', 100),

(6, 6, 'Project 6', 100),

(7, 7, 'Project 7', 100),

(8, 8, 'Project 8', 100),

(9, 9, 'Project 9', 100),

(10, 10, 'Project 10', 100);

-- Create the Grades table

CREATE TABLE Grades (

grade\_id INT PRIMARY KEY,

employee\_id INT,

course\_id INT,

quiz\_grade INT,

assignment\_grade INT,

project\_grade INT,

FOREIGN KEY (employee\_id) REFERENCES Employees(employee\_id),

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

);

INSERT INTO Grades (grade\_id, employee\_id, course\_id, quiz\_grade, assignment\_grade, project\_grade)

VALUES

(1, 1, 1, 8, 9, 95),

(2, 2, 2, 9, 9, 85),

(3, 3, 3, 10, 10, 90),

(4, 4, 4, 7, 8, 80),

(5, 5, 5, 8, 5, 75),

(6, 6, 6, 9, 9, 70),

(7, 7, 7, 10, 5, 85),

(8, 8, 8, 7, 8, 90),

(9, 9, 9, 8, 8, 95),

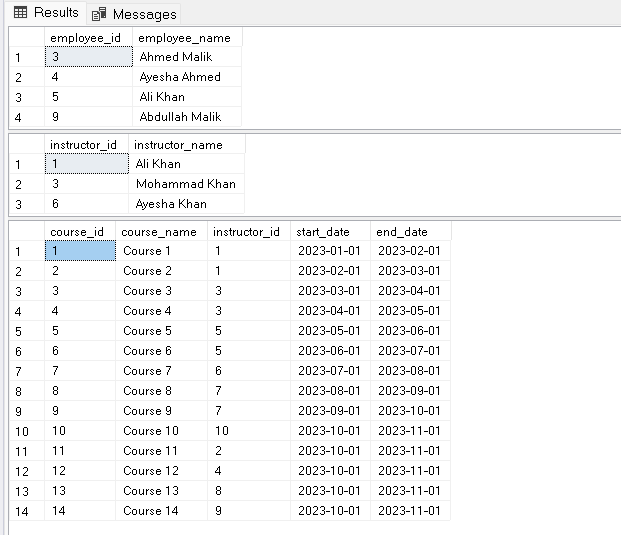
(10, 10, 10, 9, 90, 100);

**--a**

SELECT \* FROM Employees WHERE employee\_name LIKE 'A%';

SELECT \* FROM Instructors WHERE instructor\_name LIKE '%Khan';

SELECT \* FROM Courses WHERE course\_name LIKE '%Course%';



**--b**

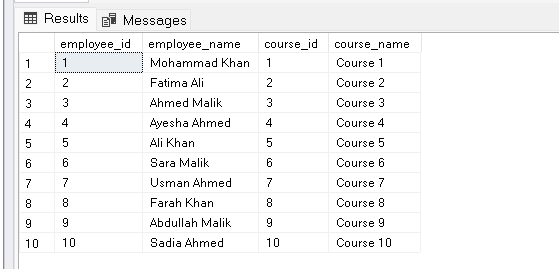
--Query to retrieve the enrollment details of employees along with their corresponding course information:

SELECT E.employee\_id, E.employee\_name, C.course\_id, C.course\_name

FROM Employees E

INNER JOIN Enrollments EN ON E.employee\_id = EN.employee\_id

INNER JOIN Courses C ON EN.course\_id = C.course\_id;

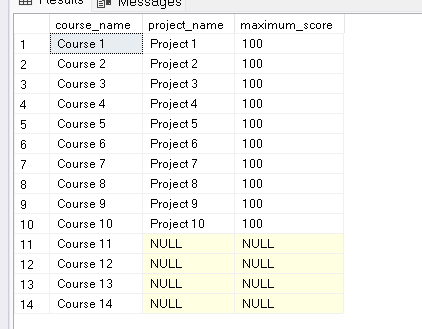


--Query to retrieve the project names and maximum scores of projects associated with each course:

SELECT C.course\_name, P.project\_name, P.maximum\_score

FROM Courses C

LEFT JOIN Projects P ON C.course\_id = P.course\_id;



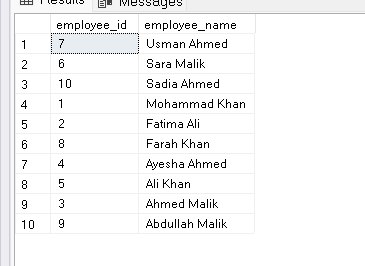
**--c**

--Get employees sorted by their names in descending order:

SELECT employee\_id, employee\_name

FROM Employees

ORDER BY employee\_name DESC;

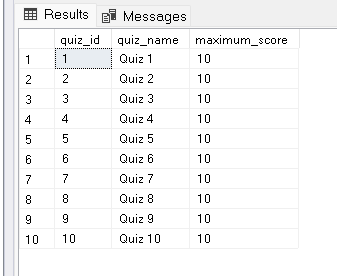


--Find quizzes sorted by the maximum score in descending order:

SELECT quiz\_id, quiz\_name, maximum\_score

FROM Quizzes

ORDER BY maximum\_score DESC;



**--d**

--Query to calculate the total number of employees:

SELECT COUNT(\*) AS total\_employees

FROM Employees;

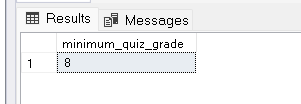


--Query to find the minimum quiz grade in a specific course:

SELECT MIN(quiz\_grade) AS minimum\_quiz\_grade

FROM Grades

WHERE course\_id = 1;

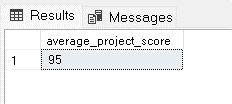


--Query to calculate the average project score for a given employee:

SELECT AVG(project\_grade) AS average\_project\_score

FROM Grades

WHERE employee\_id = 1;



**--e**

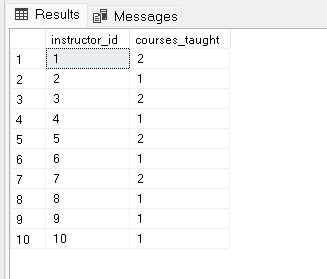
--Query to find instructors who have taught at least 1 courses:

SELECT instructor\_id, COUNT(\*) AS courses\_taught

FROM Courses

GROUP BY instructor\_id

HAVING COUNT(\*) >= 1;



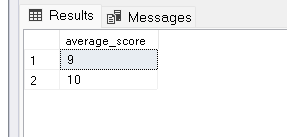
--Query to find quizzes with an average score higher than 8:

SELECT AVG(quiz\_grade) AS average\_score

FROM Grades

GROUP BY quiz\_grade

HAVING AVG(quiz\_grade) > 8;



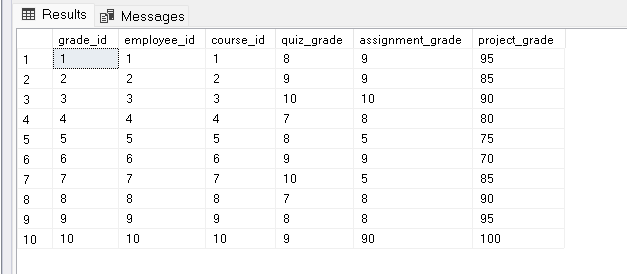
**--f**

--Query to retrieve employees who have a Project grade greater than or equal to all quiz grades:

SELECT \*

FROM Grades

WHERE project\_grade >= ALL (SELECT quiz\_grade FROM Grades);

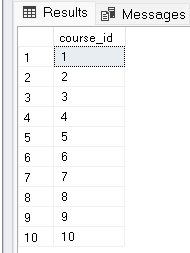


--Query to retrieve courses where the maximum project score is lesser than any quiz maximum score:

SELECT course\_id

FROM Quizzes

WHERE maximum\_score < ANY (SELECT maximum\_score FROM Projects);



**--g**

--Query to retrieve employees who are enrolled in a course taught by instructor 'Ali Khan':

SELECT employee\_id, employee\_name

FROM Employees

WHERE employee\_id IN (

SELECT employee\_id

FROM Enrollments

WHERE course\_id IN (

SELECT course\_id

FROM Courses

WHERE instructor\_id = (

SELECT instructor\_id

FROM Instructors

WHERE instructor\_name = 'Ali Khan'

)

)

);



--Query to retrieve courses in which an employee achieved a perfect score (100) in the project:

SELECT course\_id, course\_name

FROM Courses

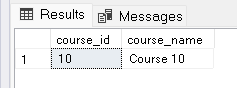
WHERE course\_id IN (

SELECT course\_id

FROM Grades

WHERE project\_grade = 100

);



--Query to retrieve courses that have at least one employee with a grade higher than 9 in any quiz:

SELECT course\_id, course\_name

FROM Courses

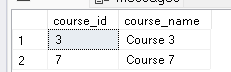
WHERE course\_id IN (

SELECT course\_id

FROM Grades

WHERE quiz\_grade > 9

);



**--h**

**--SIMPLE JOIN**

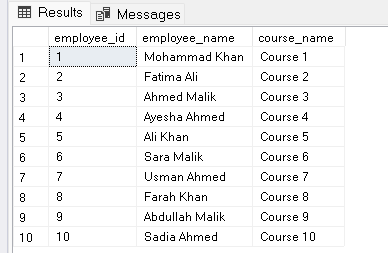
--Retrieve the employees and their corresponding course names:

SELECT E.employee\_id, E.employee\_name, C.course\_name

FROM Employees E

JOIN Enrollments EN ON E.employee\_id = EN.employee\_id

JOIN Courses C ON EN.course\_id = C.course\_id;

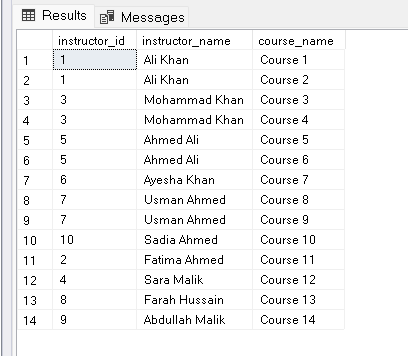


--Retrieve the instructors and the courses they teach:

SELECT I.instructor\_id, I.instructor\_name, C.course\_name

FROM Instructors I

JOIN Courses C ON I.instructor\_id = C.instructor\_id;



**--SORTING A JOIN**

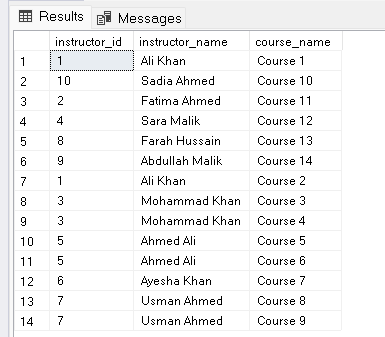
--Retrieve the instructors and the courses they teach, sorted by the course names in ascending order:

SELECT I.instructor\_id, I.instructor\_name, C.course\_name

FROM Instructors I

JOIN Courses C ON I.instructor\_id = C.instructor\_id

ORDER BY C.course\_name ASC;



--Retrieve the employees and their corresponding course names, sorted by the employee names in descending order:

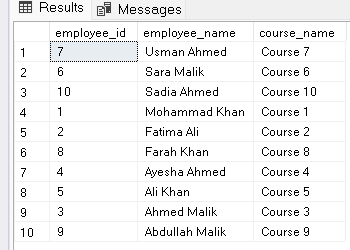
SELECT E.employee\_id, E.employee\_name, C.course\_name

FROM Employees E

JOIN Enrollments EN ON E.employee\_id = EN.employee\_id

JOIN Courses C ON EN.course\_id = C.course\_id

ORDER BY E.employee\_name DESC;



**--THREE TABLE JOIN**

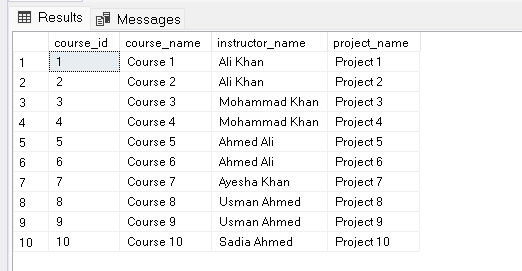
--Retrieve the courses, their corresponding instructor names, and the project names associated with each course:

SELECT C.course\_id, C.course\_name, I.instructor\_name, P.project\_name

FROM Courses C

JOIN Instructors I ON C.instructor\_id = I.instructor\_id

JOIN Projects P ON C.course\_id = P.course\_id;



**--OUTER JOIN**

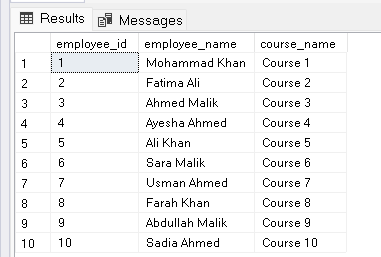
--Retrieve all employees and their corresponding course names, including employees who haven't enrolled in any course:

SELECT E.employee\_id, E.employee\_name, C.course\_name

FROM Employees E

LEFT JOIN Enrollments EN ON E.employee\_id = EN.employee\_id

LEFT JOIN Courses C ON EN.course\_id = C.course\_id;

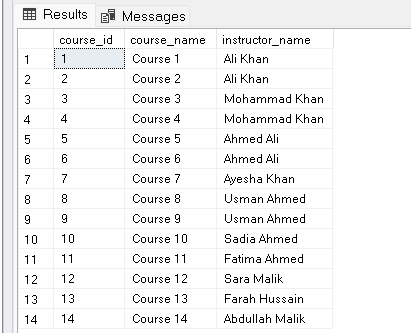
****

--Retrieve all courses and their corresponding instructor names, including courses without assigned instructors:

SELECT C.course\_id, C.course\_name, I.instructor\_name

FROM Courses C

LEFT JOIN Instructors I ON C.instructor\_id = I.instructor\_id;

****

**--i CREATING UPDATABLE VIEWS**

CREATE VIEW UpdatableCoursesView AS

SELECT course\_id, course\_name

FROM Courses;

**--j MODIFYING VIEWS**

ALTER VIEW UpdatableCoursesView AS

SELECT course\_id, course\_name, instructor\_id, start\_date, end\_date

FROM Courses;

SQL Server does not support directly modifying the structure of a view using ALTER VIEW instead, we need to drop and recreate the view with the desired changes.

================================= **END ☺** ==================================