1-Creating the Tables:

```
SQLQuery9.sql -(...RS&H.mmheg (63))* * X -- Create the Students table :- CREATE TABLE Students ( student, id INT IDENTITY(1,1) PRIMARY KEY, first_name VARCHAR(50), last_name VARCHAR(50), email VARCHAR(100), date_of_birth DATE
     -- Create the Instructors table

CCREATE TABLE Instructors (
   instructor id INT IDENTITY(1,1) PRIMARY KEY,
   first_name VARCHAR(50),
   last_name VARCHAR(50),
   email VARCHAR(100)
);
      100 % -
 Messages
Commands completed successfully
     Completion time: 2024-09-02T00:58:33.1930821+03:00
 100 % -
```

2-inserting students:

```
□ INSERT INTO Students (first_name, last_name, email, date_of_birth) VALUES ('Ahmed', 'Mostafa', 'ahmed.mostafa@example.com', '2000-05-12'), ('Fatma', 'Hassan', 'fatma.hassan@example.com', '2001-08-23'), ('Omar', 'fatma.hassan@example.com', '2002-11-30'), ('Sara', 'Ali', 'sara.ali@example.com', '2003-07-19'), ('Mohamed', 'Khaled', 'mohamed.khaled@example.com', '2004-02-15'), ('Yasmin', 'Ibrahim', 'yasmin.ibrahim@example.com', '2004-09-08'), ('Khaled', 'Abdelrahman', 'khaled.abdelrahman@example.com', '2001-03-25'), ('Aya', 'Gamal', 'aya.gamal@example.com', '2002-06-14'), ('Hassan', 'Mahamoud', 'hassan.mahamoud@example.com', '2005-12-21'), ('Mona', 'Tarek', 'mona.tarek@example.com', '2004-10-10');
```

```
(10 rows affected)
Completion time: 2024-09-02T01:01:29.6650610+03:00
```

3-inserting instructors:

```
SQLQuerys.ql - (...IRSSLFAmmheg (63))* = ×

SINSERT INTO Instructors (first_name, last_name, email) VALUES

('Ahmed', 'Ezab', 'ahmed.ezab@gmail.com'),

('Mona', 'Shawky', 'mona.shawky@gmail.com'),

('Tarek', 'Adel', 'tarek.adel@gmail.com');
100 % - 4
      (3 rows affected)
      Completion time: 2024-09-02T01:00:05.8437132+03:00
```

4- inserting Courses:

```
EINSERT INTO Courses (course_name, course_description, instructor_id) VALUES

('Mathematics', 'An introductory course to Mathematics', 1), -- Assign to Ahmed

('Mathematics', 'An introductory course to Mathematics and The Course of Cour
100 % ▼ ◀ ■
Messages
                                (5 rows affected)
                                Completion time: 2024-09-02T01:00:50.2875895+03:00
```

5- inserting enrollments:

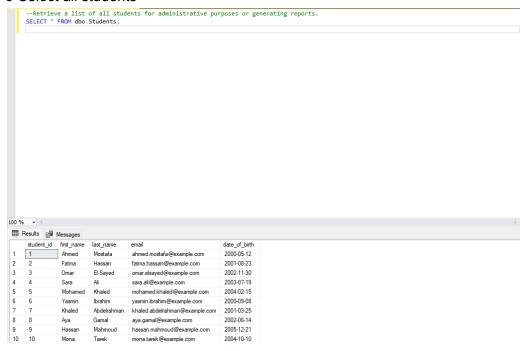
```
5- inserting enrollments:

| INSERT INTO Enrollments (student_id, course_id, enrollment_date) VALUES (1, 1, '2023-09-01'), (2, 1, '2023-09-02'), (3, 2, '2023-09-03'), (4, 3, '2023-09-03'), (4, 3, '2023-09-04'), (5, 4, '2023-09-06'), (7, 1, '2023-09-07'), (8, 2, '2023-09-08'), (9, 3, '2023-09-09'), (10, 4, '2023-09-11'), (1, 5, '2023-09-11'), (2, 3, '2023-09-12'), (3, 4, '2023-09-13'), (4, 5, '2023-09-14'), (5, 2, '2023-09-15');
 0 % +
Messages
```

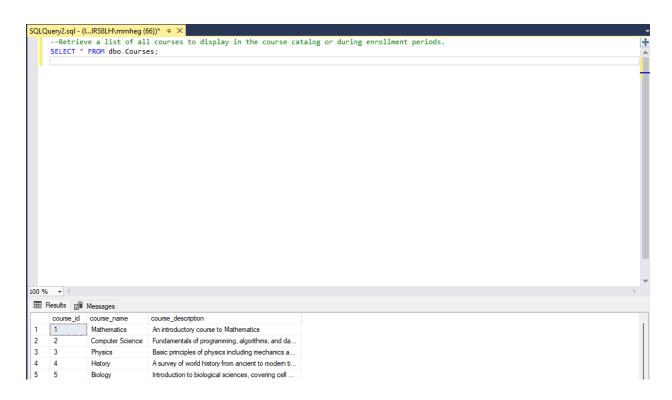
(15 rows affected)

Completion time: 2024-09-02T01:02:06.1797466+03:00

6-Select all students



7-Select all courses:



8-Select all enrollments with student names and course names:

```
--Generate a detailed report showing which students are enrolled in which courses.

□SELECT

e.enrollment_id,

s.first_name + ' ' + s.last_name AS student_name,

c.course_name

FROM

dbo.Enrollments e

JOIN dbo.Students s ON e.student_id = s.student_id

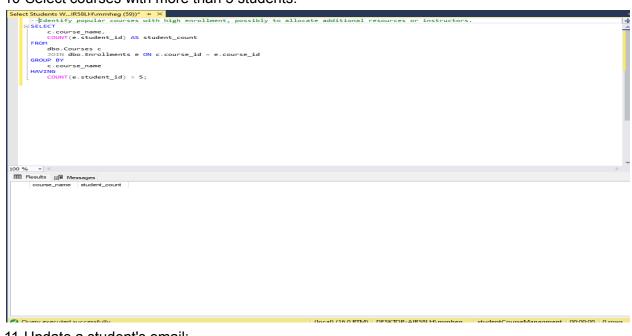
JOIN dbo.Courses c ON e.course_id = c.course_id;
```



9-Select students who enrolled in a specific course:



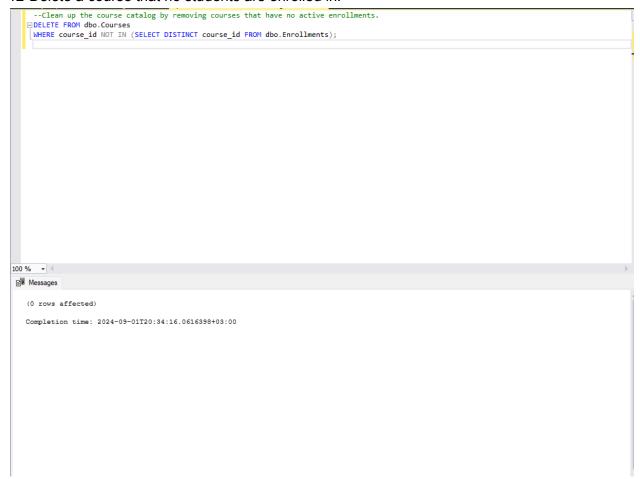
10-Select courses with more than 5 students:



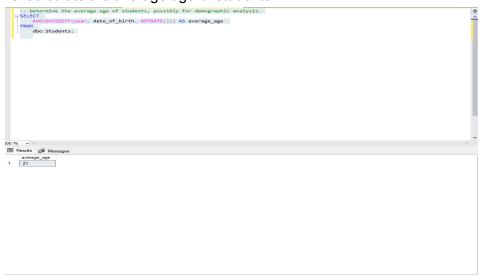
11-Update a student's email:



12-Delete a course that no students are enrolled in:



13-Calculate the average age of students:



14-Find the course with the maximum enrollments:

```
--Identify the most popular course, which might require additional resources or sections.

| SELECT TOP 1 | C. COURSE, name, | C. COURSE, name, | C. SOURCE, | C. SOUR
III Results Mussages

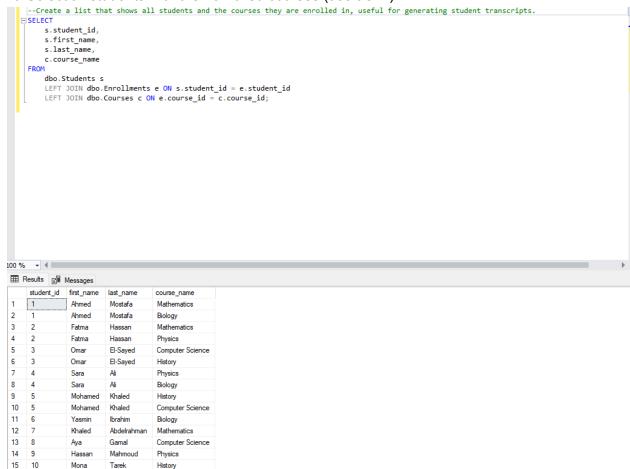
Course_name enrollment_count

Computer Science 3
```

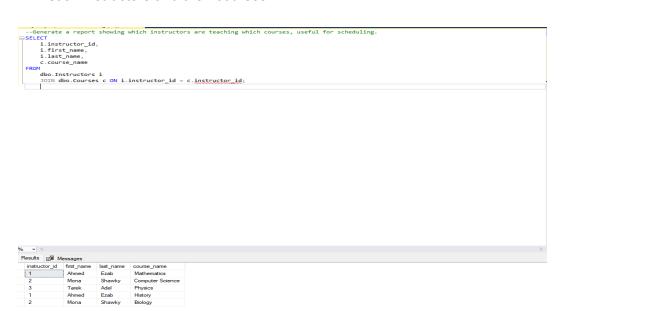
15-List courses along with the number of students enrolled (use GROUP BY):



16-Select all students with their enrolled courses (use JOIN):



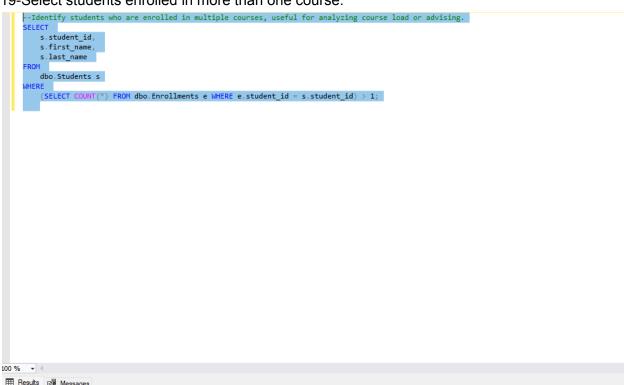
17-List all instructors and their courses:

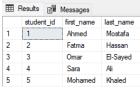


18- Find students who are not enrolled in any course:



19-Select students enrolled in more than one course:





20-Find courses taught by a specific instructor:

```
-- List all courses taught by a specific instructor, useful for planning and workload balancing.

BSELECT
FOO. Course_name

dbo. Course c

MHERE

c.instructor_id = (SELECT instructor_id FROM dbo.Instructors NHERE last_name = 'Ezabl');

00 % --

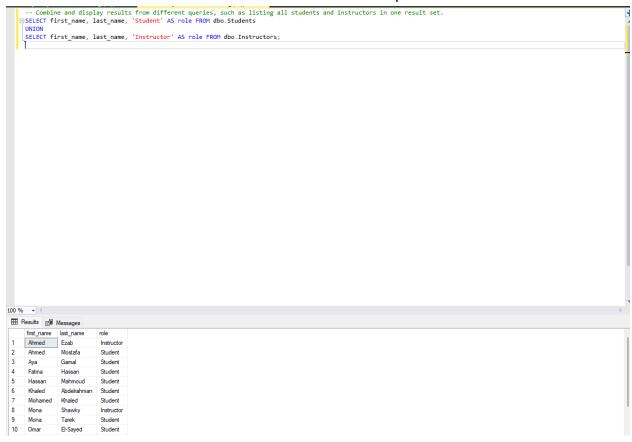
Results @ Messages

| course_name |
| Memendics |
| Memendics |
| Hendics |
| Hendic
```

21-Select the top 3 students with the most enrollments:



22-Use UNION to combine results of two different SELECT queries:



23-Create a stored procedure to add a new student:

```
--Automate the process of adding a new student to the database, ensuring consistency and reducing manual input.

(GREATE PROCEDURE AddStudent
(@FirstName VARCHAR(50))
(@LastName VARCHAR(50))
(@BoateOfBirth DATE

AS
 guateOfBirth DATE
AS
BESGIN
BIRSERT INTO dbo.Students (first_name, last_name, email, date_of_birth)
VALUES (@FirstName, @LastName, @Email, @OateOfBirth);
END;
Completion time: 2024-09-02T01:28:22.0398204+03:00
```

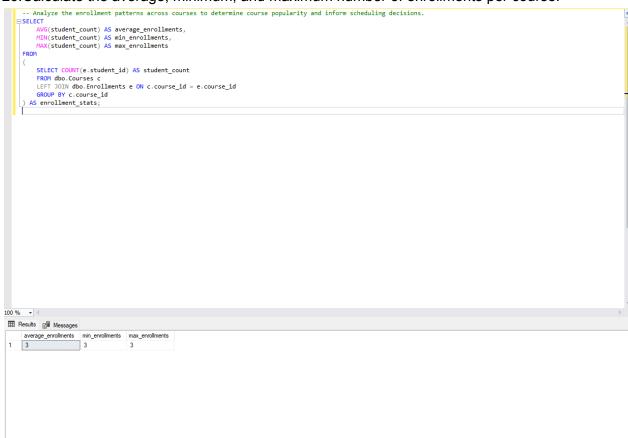
24-Create a function to calculate the age of a student based on their date of birth:



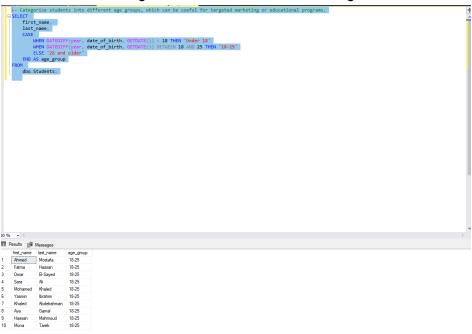
25-Calculate the total number of students:



26:Calculate the average, minimum, and maximum number of enrollments per course:



27-Use CASE to categorize students based on their age:



28-Use EXISTS to find courses with at least one enrolled student:



