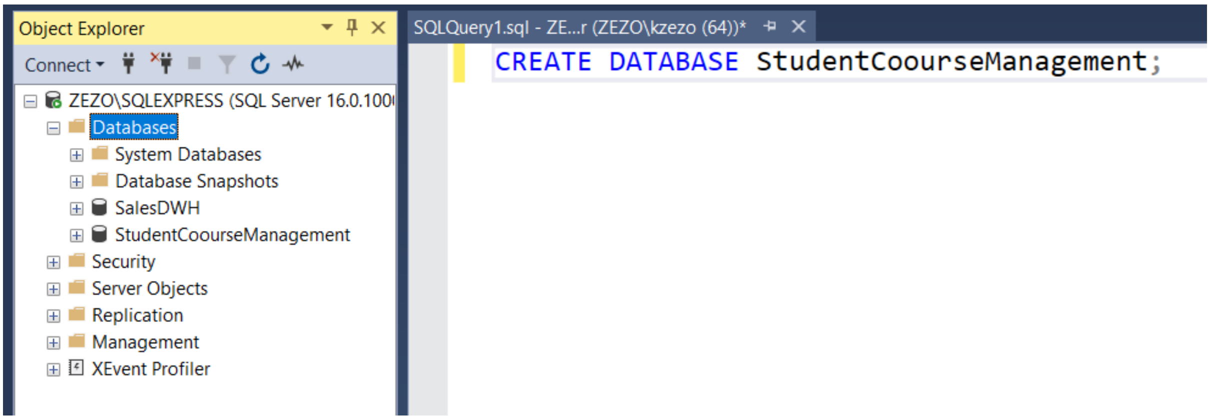
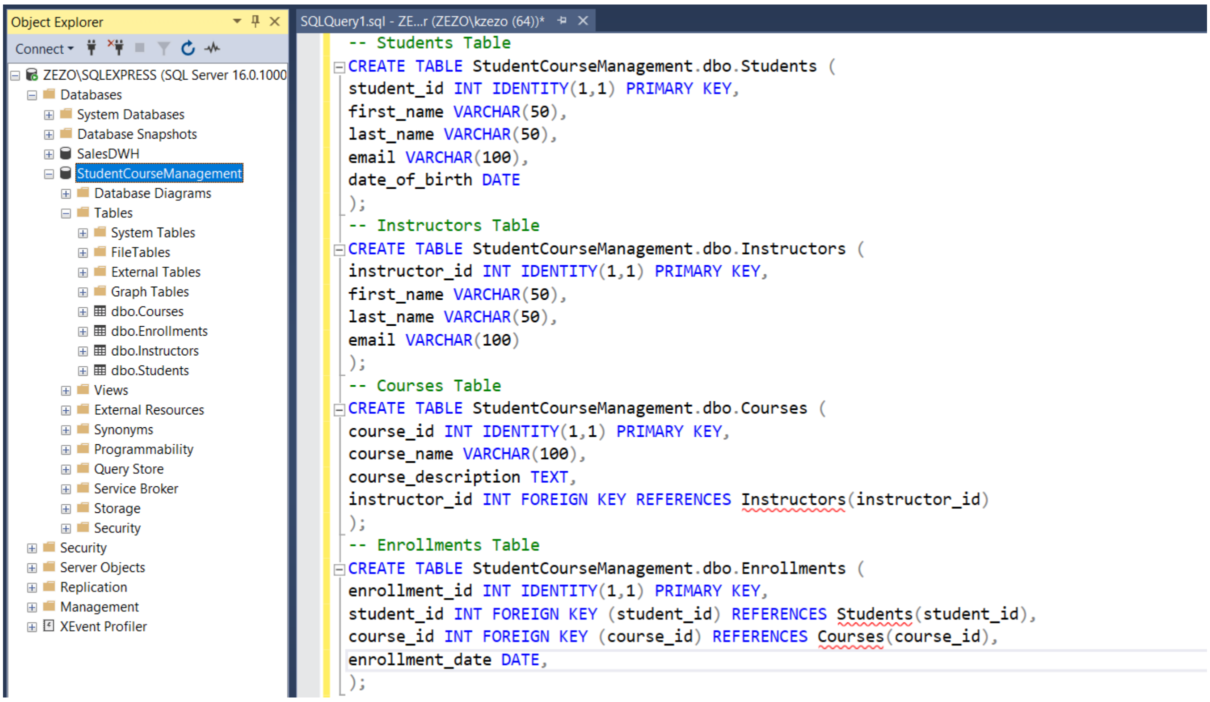
1) Database Setup:



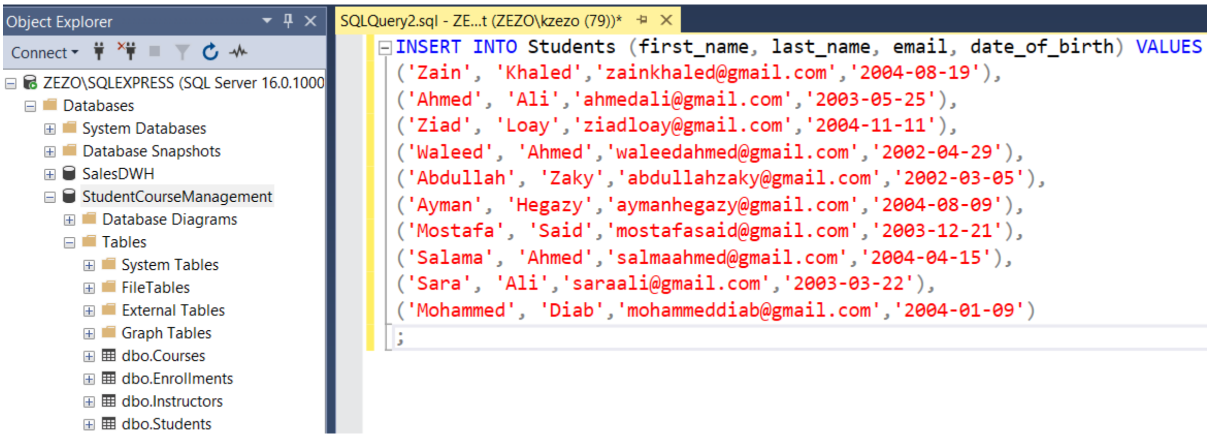
```sql
CREATE DATABASE StudentCoourseManagement;
```

2) Table Creation:

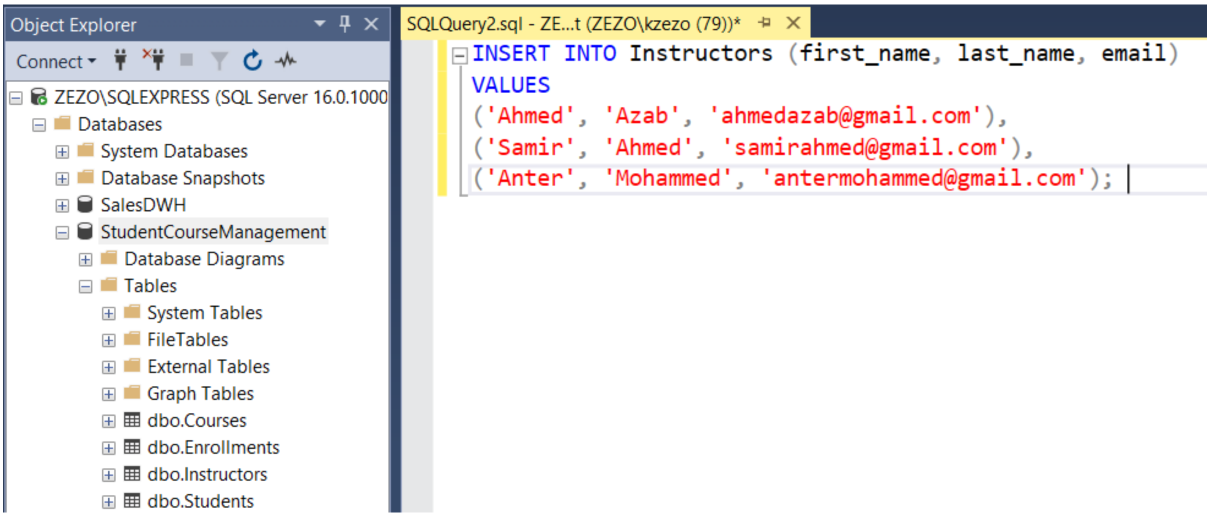


```sql
-- Students Table
CREATE TABLE StudentCourseManagement.dbo.Students (
student_id INT IDENTITY(1,1) PRIMARY KEY,
first_name VARCHAR(50),
last_name VARCHAR(50),
email VARCHAR(100),
date_of_birth DATE
);
-- Instructors Table
CREATE TABLE StudentCourseManagement.dbo.Instructors (
instructor_id INT IDENTITY(1,1) PRIMARY KEY,
first_name VARCHAR(50),
last_name VARCHAR(50),
email VARCHAR(100)
);
-- Courses Table
CREATE TABLE StudentCourseManagement.dbo.Courses (
course_id INT IDENTITY(1,1) PRIMARY KEY,
course_name VARCHAR(100),
course_description TEXT,
instructor_id INT FOREIGN KEY REFERENCES Instructors(instructor_id)
);
-- Enrollments Table
CREATE TABLE StudentCourseManagement.dbo.Enrollments (
enrollment_id INT IDENTITY(1,1) PRIMARY KEY,
student_id INT FOREIGN KEY (student_id) REFERENCES Students(student_id),
course_id INT FOREIGN KEY (course_id) REFERENCES Courses(course_id),
enrollment_date DATE,
);
```

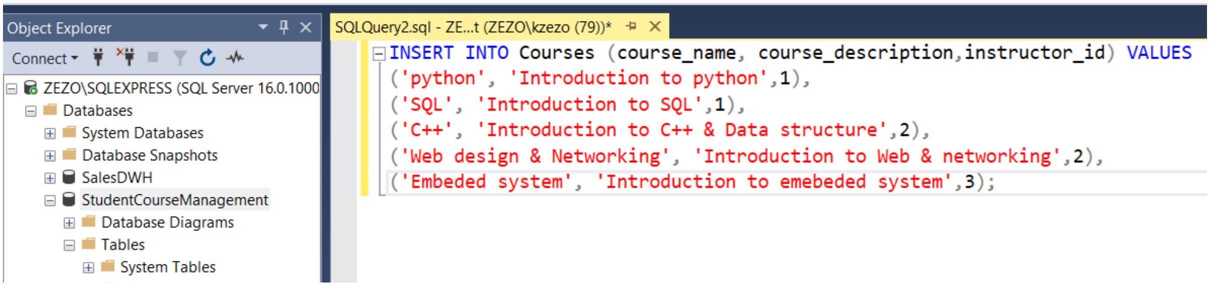
3) Insert Sample Data



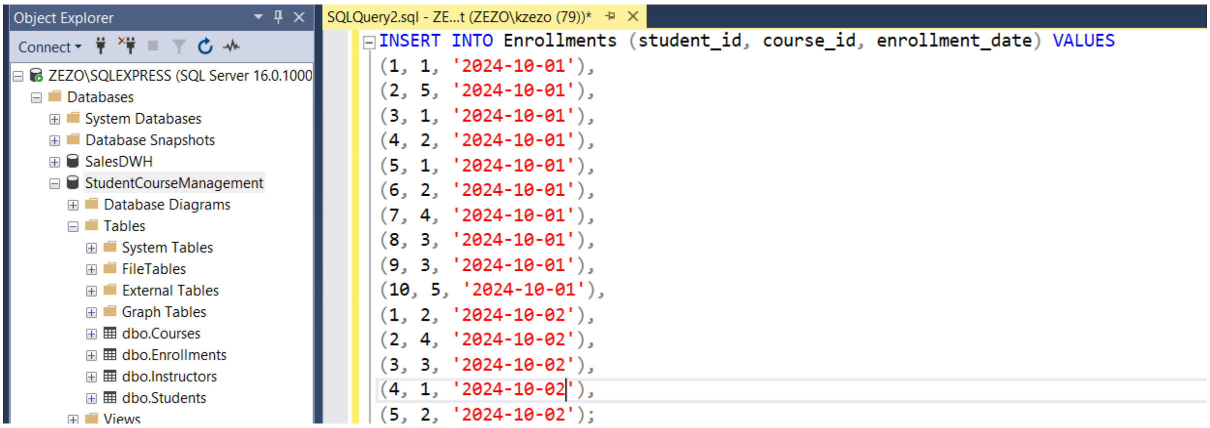
```sql
INSERT INTO Students (first_name, last_name, email, date_of_birth) VALUES
('Zain', 'Khaled','zainkhaled@gmail.com','2004-08-19'),
('Ahmed', 'Ali','ahmedali@gmail.com','2003-05-25'),
('Ziad', 'Loay','ziadloay@gmail.com','2004-11-11'),
('Waleed', 'Ahmed','waleedahmed@gmail.com','2002-04-29'),
('Abdullah', 'Zaky','abdullahzaky@gmail.com','2002-03-05'),
('Ayman', 'Hegazy','aymanhegazy@gmail.com','2004-08-09'),
('Mostafa', 'Said','mostafasaid@gmail.com','2003-12-21'),
('Salama', 'Ahmed','salmaahmed@gmail.com','2004-04-15'),
('Sara', 'Ali','saraali@gmail.com','2003-03-22'),
('Mohammed', 'Diab','mohammeddiab@gmail.com','2004-01-09')
;
```



```sql
INSERT INTO Instructors (first_name, last_name, email)
VALUES
('Ahmed', 'Azab', 'ahmedazab@gmail.com'),
('Samir', 'Ahmed', 'samirahmed@gmail.com'),
('Anter', 'Mohammed', 'antermohammed@gmail.com');
```



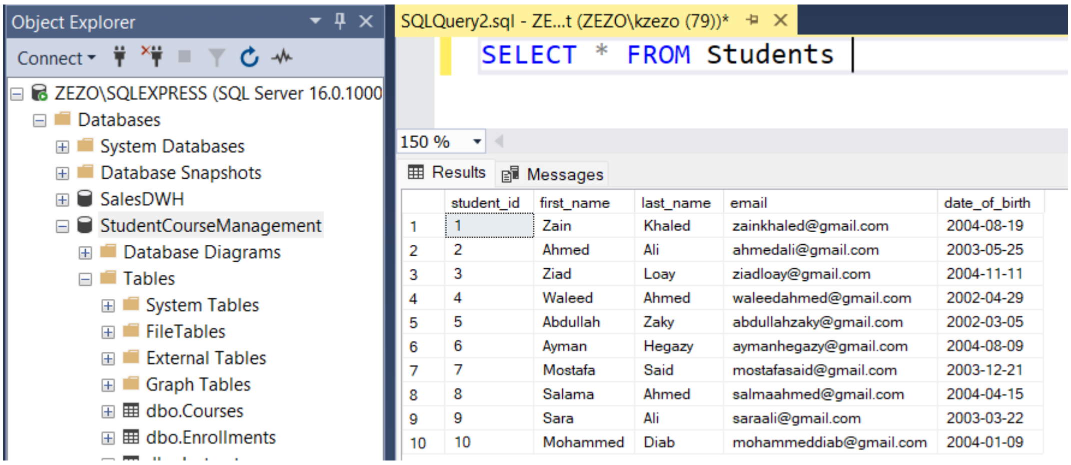
```sql
INSERT INTO Courses (course_name, course_description,instructor_id) VALUES
('python', 'Introduction to python',1),
('SQL', 'Introduction to SQL',1),
('C++', 'Introduction to C++ & Data structure',2),
('Web design & Networking', 'Introduction to Web & networking',2),
('Embeded system', 'Introduction to emebeded system',3);
```



```sql
INSERT INTO Enrollments (student_id, course_id, enrollment_date) VALUES
(1, 1, '2024-10-01'),
(2, 5, '2024-10-01'),
(3, 1, '2024-10-01'),
(4, 2, '2024-10-01'),
(5, 1, '2024-10-01'),
(6, 2, '2024-10-01'),
(7, 4, '2024-10-01'),
(8, 3, '2024-10-01'),
(9, 3, '2024-10-01'),
(10, 5, '2024-10-01'),
(1, 2, '2024-10-02'),
(2, 4, '2024-10-02'),
(3, 3, '2024-10-02'),
(4, 1, '2024-10-02'),
(5, 2, '2024-10-02');
```

4) Basic Queries:

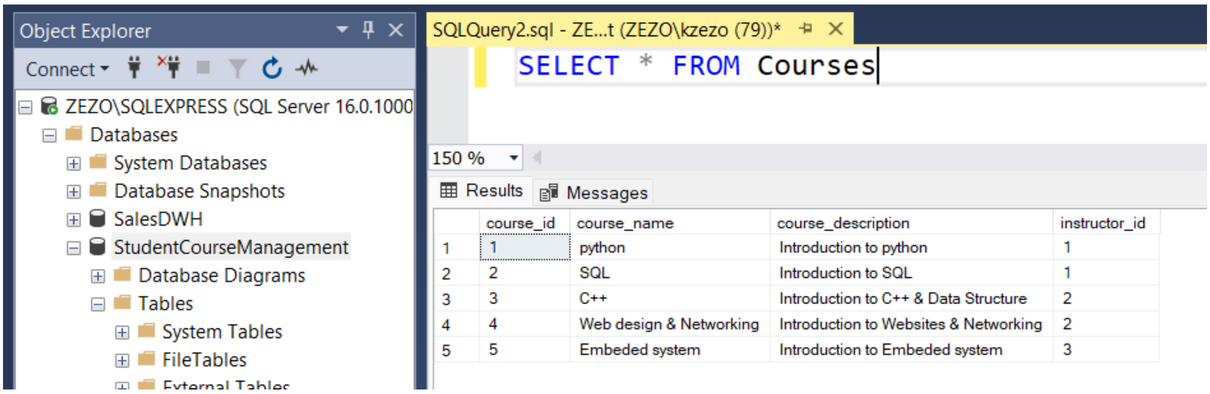
(i) Select all students



```sql
SELECT * FROM Students
```

	student_id	first_name	last_name	email	date_of_birth
1	1	Zain	Khaled	zainkhaled@gmail.com	2004-08-19
2	2	Ahmed	Ali	ahmedali@gmail.com	2003-05-25
3	3	Ziad	Loay	ziadloay@gmail.com	2004-11-11
4	4	Waleed	Ahmed	waleedahmed@gmail.com	2002-04-29
5	5	Abdullah	Zaky	abdullahzaky@gmail.com	2002-03-05
6	6	Ayman	Hegazy	aymanhegazy@gmail.com	2004-08-09
7	7	Mostafa	Said	mostafasaid@gmail.com	2003-12-21
8	8	Salama	Ahmed	salmaahmed@gmail.com	2004-04-15
9	9	Sara	Ali	saraali@gmail.com	2003-03-22
10	10	Mohammed	Diab	mohammeddiab@gmail.com	2004-01-09

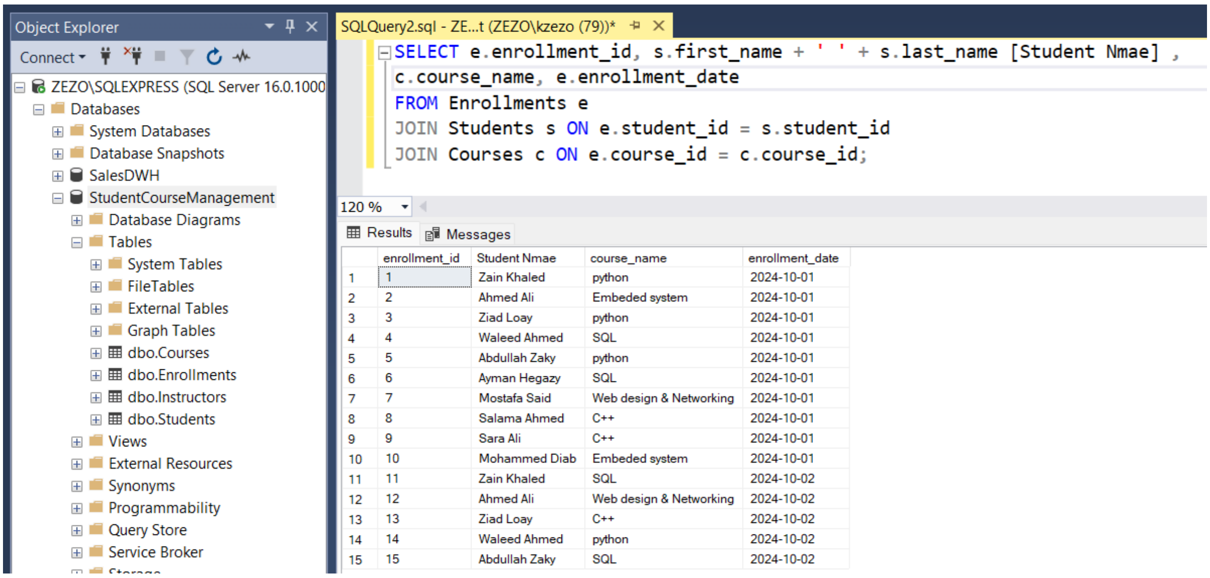
(ii) Select all courses



```sql
SELECT * FROM Courses
```

	course_id	course_name	course_description	instructor_id
1	1	python	Introduction to python	1
2	2	SQL	Introduction to SQL	1
3	3	C++	Introduction to C++ & Data Structure	2
4	4	Web design & Networking	Introduction to Websites & Networking	2
5	5	Embeded system	Introduction to Embeded system	3

(iii) Select all enrollments with student names and course names

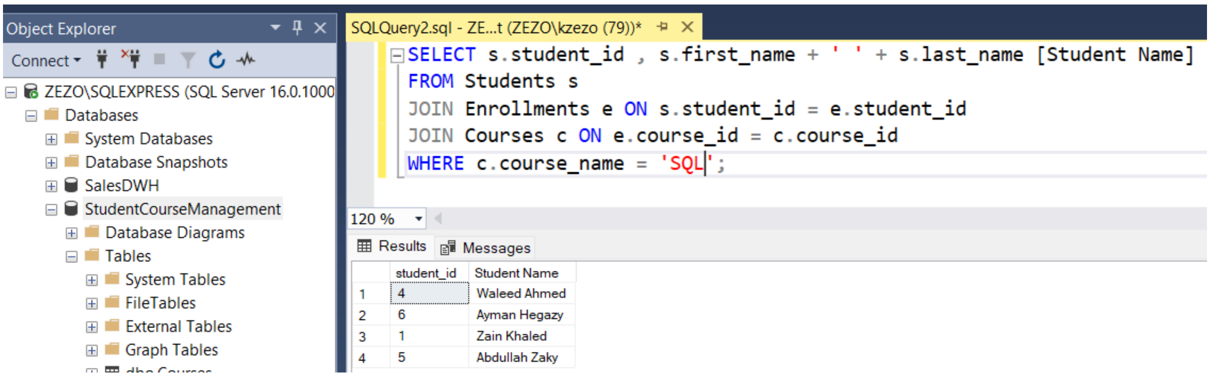


```sql
SELECT e.enrollment_id, s.first_name + ' ' + s.last_name [Student Nmae] ,
c.course_name, e.enrollment_date
FROM Enrollments e
JOIN Students s ON e.student_id = s.student_id
JOIN Courses c ON e.course_id = c.course_id;
```

	enrollment_id	Student Nmae	course_name	enrollment_date
1	1	Zain Khaled	python	2024-10-01
2	2	Ahmed Ali	Embeded system	2024-10-01
3	3	Ziad Loay	python	2024-10-01
4	4	Waleed Ahmed	SQL	2024-10-01
5	5	Abdullah Zaky	python	2024-10-01
6	6	Ayman Hegazy	SQL	2024-10-01
7	7	Mostafa Said	Web design & Networking	2024-10-01
8	8	Salama Ahmed	C++	2024-10-01
9	9	Sara Ali	C++	2024-10-01
10	10	Mohammed Diab	Embeded system	2024-10-01
11	11	Zain Khaled	SQL	2024-10-02
12	12	Ahmed Ali	Web design & Networking	2024-10-02
13	13	Ziad Loay	C++	2024-10-02
14	14	Waleed Ahmed	python	2024-10-02
15	15	Abdullah Zaky	SQL	2024-10-02

5) Advanced queries:

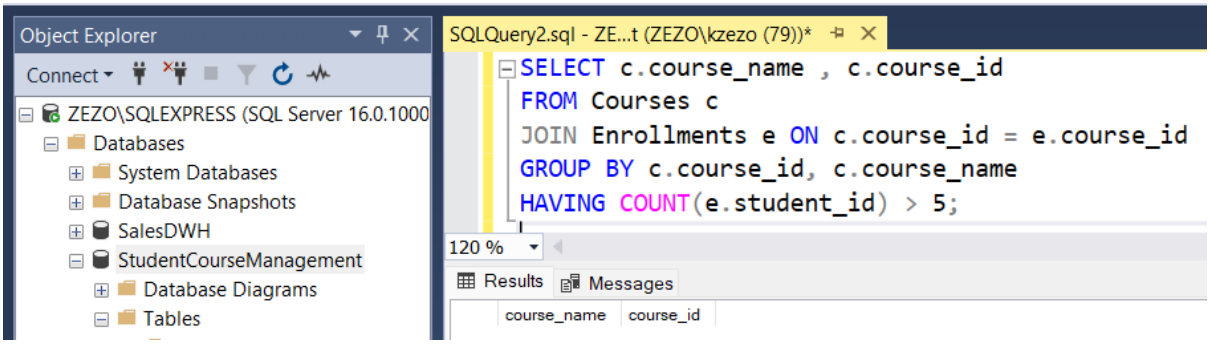
1-Select students who enrolled in a specific course



```sql
SELECT s.student_id , s.first_name + ' ' + s.last_name [Student Name]
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 = 'SQL';
```

	student_id	Student Name
1	4	Waleed Ahmed
2	6	Ayman Hegazy
3	1	Zain Khaled
4	5	Abdullah Zaky

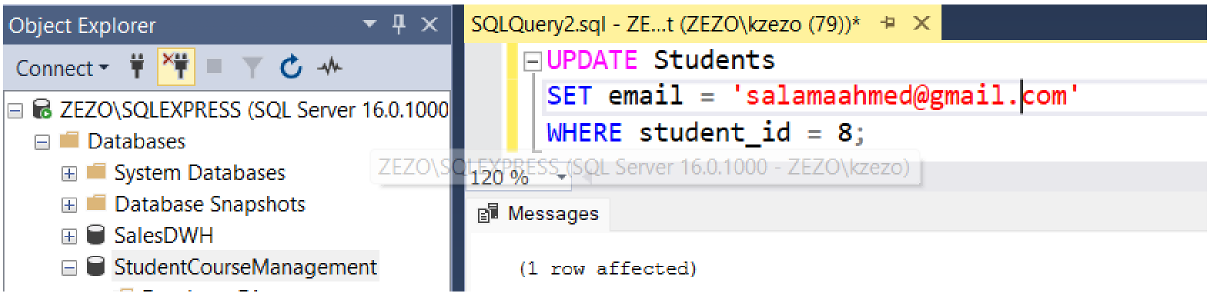
2-Select courses with more than 5 students



```sql
SELECT c.course_name , c.course_id
FROM Courses c
JOIN Enrollments e ON c.course_id = e.course_id
GROUP BY c.course_id, c.course_name
HAVING COUNT(e.student_id) > 5;
```

```
course_name   course_id
```

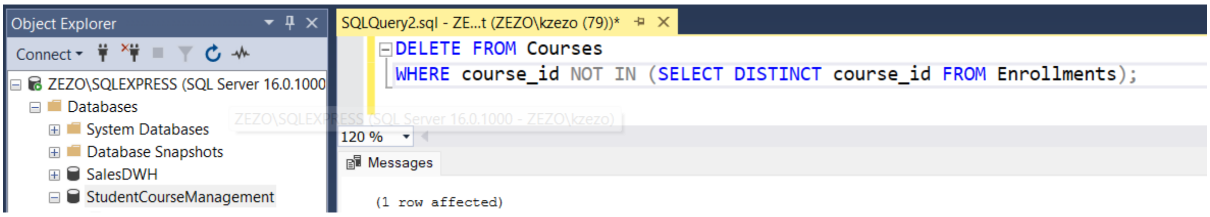
3-Update a student's email


Object Explorer
Connect
ZEZO\SQLEXPRESS (SQL Server 16.0.1000
Databases
System Databases
Database Snapshots
SalesDWH
StudentCourseManagement
SQLQuery2.sql - ZE...t (ZEZO\kzezo (79))*
ZEZO\SQLEXPRESS (SQL Server 16.0.1000 - ZEZO\kzezo)
120 %
Messages

```sql
UPDATE Students
SET email = 'salamaahmed@gmail.com'
WHERE student_id = 8;
```

```
(1 row affected)
```

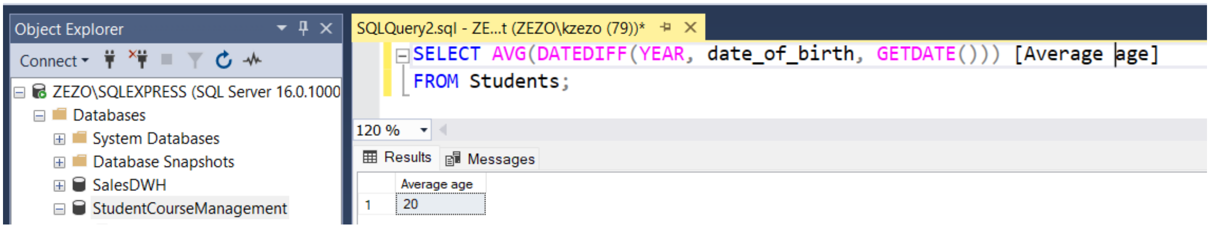
4- Delete a course that no students are enrolled in


Object Explorer
Connect
ZEZO\SQLEXPRESS (SQL Server 16.0.1000
Databases
System Databases
Database Snapshots
SalesDWH
StudentCourseManagement
SQLQuery2.sql - ZE...t (ZEZO\kzezo (79))*
ZEZO\SQLEXPRESS (SQL Server 16.0.1000 - ZEZO\kzezo)
120 %
Messages

```sql
DELETE FROM Courses
WHERE course_id NOT IN (SELECT DISTINCT course_id FROM Enrollments);
```

```
(1 row affected)
```

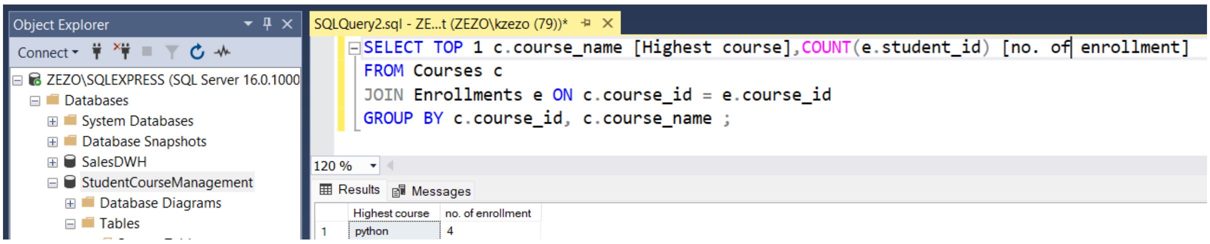
5-Calculate the average age of students



```sql
SELECT AVG(DATEDIFF(YEAR, date_of_birth, GETDATE())) [Average age]
FROM Students;
```

```
   Average age
1  20
```

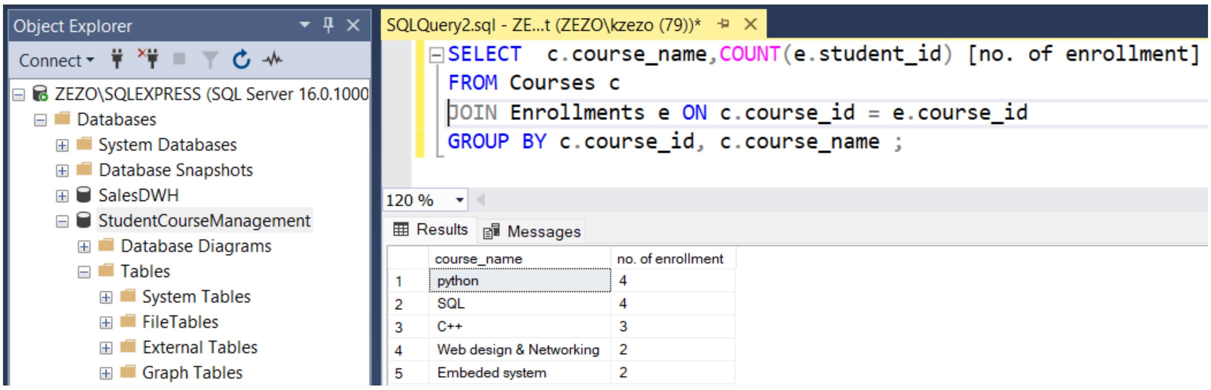
6-Find the course with the maximum enrollments



```sql
SELECT TOP 1 c.course_name [Highest course],COUNT(e.student_id) [no. of enrollment]
FROM Courses c
JOIN Enrollments e ON c.course_id = e.course_id
GROUP BY c.course_id, c.course_name ;
```

```
   Highest course   no. of enrollment
1  python           4
```

7- List courses along with the number of students enrolled (use GROUP BY).

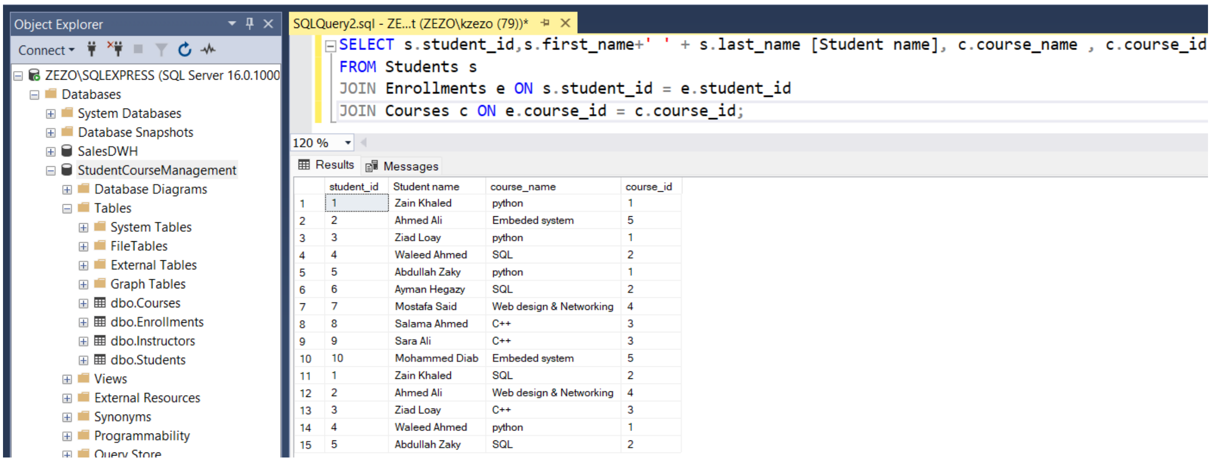


```sql
SELECT  c.course_name,COUNT(e.student_id) [no. of enrollment]
FROM Courses c
JOIN Enrollments e ON c.course_id = e.course_id
GROUP BY c.course_id, c.course_name ;
```

	course_name	no. of enrollment
1	python	4
2	SQL	4
3	C++	3
4	Web design & Networking	2
5	Embeded system	2

6) Join Queries:

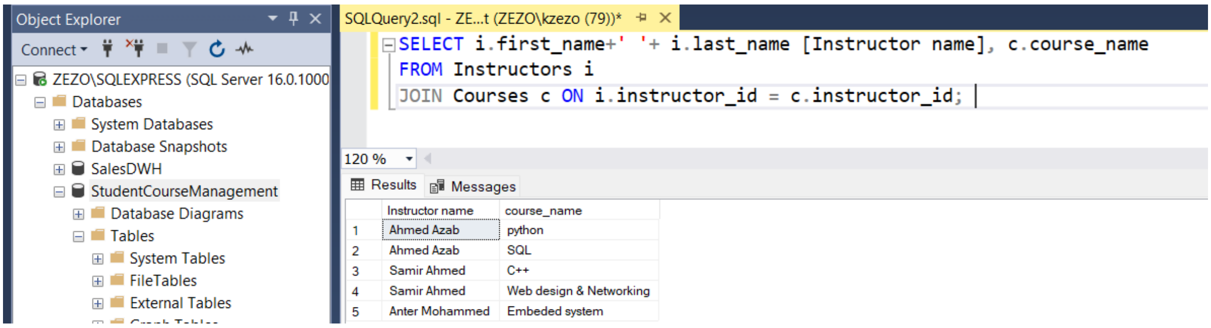
(i) Select all students with their enrolled courses (use JOIN)



```sql
SELECT s.student_id,s.first_name+' ' + s.last_name [Student name], c.course_name , c.course_id
FROM Students s
JOIN Enrollments e ON s.student_id = e.student_id
JOIN Courses c ON e.course_id = c.course_id;
```

	student_id	Student name	course_name	course_id
1	1	Zain Khaled	python	1
2	2	Ahmed Ali	Embeded system	5
3	3	Ziad Loay	python	1
4	4	Waleed Ahmed	SQL	2
5	5	Abdullah Zaky	python	1
6	6	Ayman Hegazy	SQL	2
7	7	Mostafa Said	Web design & Networking	4
8	8	Salama Ahmed	C++	3
9	9	Sara Ali	C++	3
10	10	Mohammed Diab	Embeded system	5
11	1	Zain Khaled	SQL	2
12	2	Ahmed Ali	Web design & Networking	4
13	3	Ziad Loay	C++	3
14	4	Waleed Ahmed	python	1
15	5	Abdullah Zaky	SQL	2

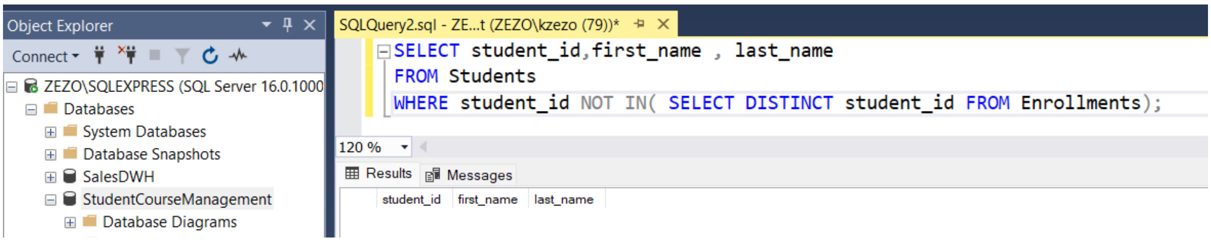
(ii) List all instructors and their courses



```sql
SELECT i.first_name+' '+ i.last_name [Instructor name], c.course_name
FROM Instructors i
JOIN Courses c ON i.instructor_id = c.instructor_id;
```

	Instructor name	course_name
1	Ahmed Azab	python
2	Ahmed Azab	SQL
3	Samir Ahmed	C++
4	Samir Ahmed	Web design & Networking
5	Anter Mohammed	Embeded system

(iii) Find students who are not enrolled in any course

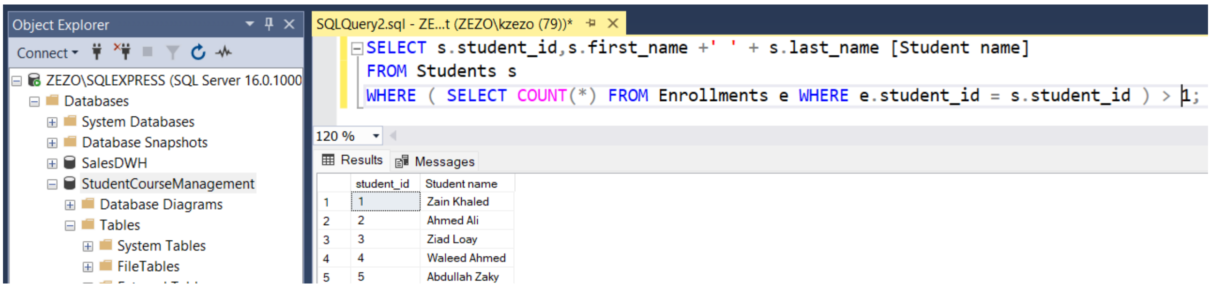


```sql
SELECT student_id,first_name , last_name
FROM Students
WHERE student_id NOT IN( SELECT DISTINCT student_id FROM Enrollments);
```

student_id	first_name	last_name

7) Subqueries and Set Operations:

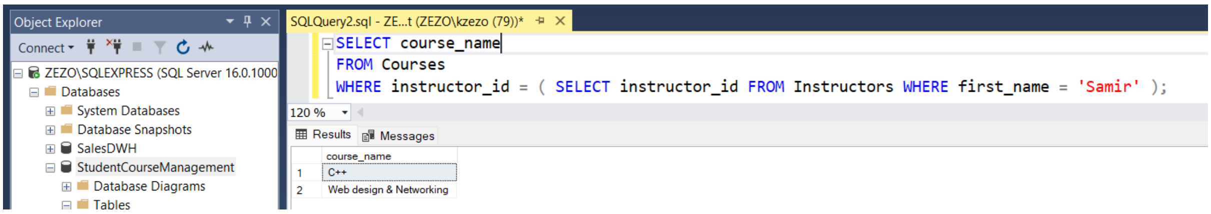
(i) Select students enrolled in more than one course



```sql
SELECT s.student_id,s.first_name +' ' + s.last_name [Student name]
FROM Students s
WHERE ( SELECT COUNT(*) FROM Enrollments e WHERE e.student_id = s.student_id ) > 1;
```

	student_id	Student name
1	1	Zain Khaled
2	2	Ahmed Ali
3	3	Ziad Loay
4	4	Waleed Ahmed
5	5	Abdullah Zaky

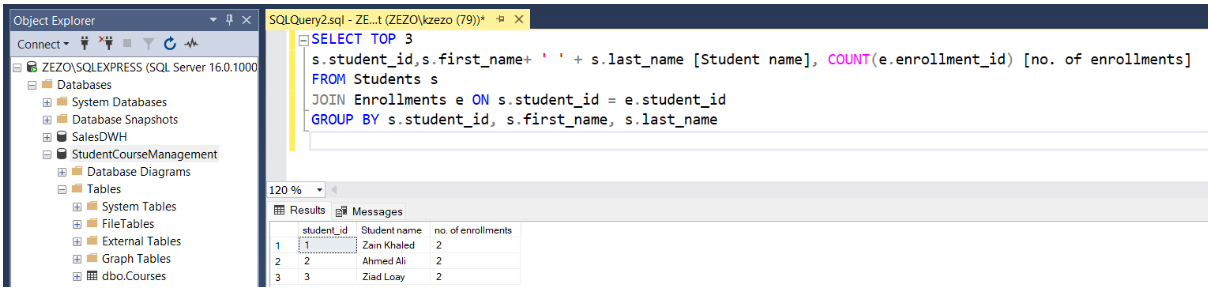
(ii) Find courses taught by a specific instructor



```sql
SELECT course_name
FROM Courses
WHERE instructor_id = ( SELECT instructor_id FROM Instructors WHERE first_name = 'Samir' );
```

	course_name
1	C++
2	Web design & Networking

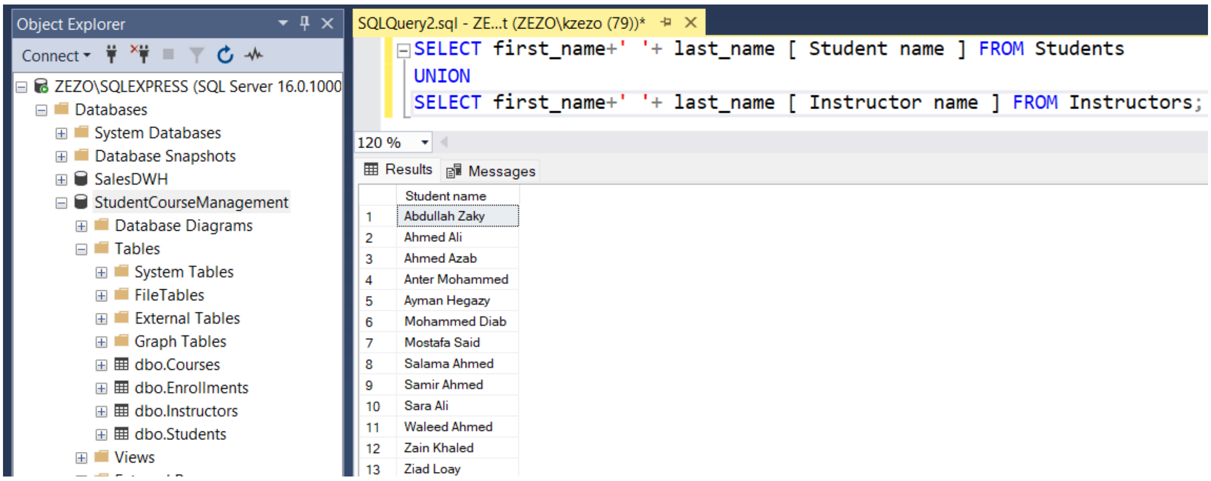
(iii) Select the top 3 students with the most enrollments



```sql
SELECT TOP 3
s.student_id,s.first_name+ ' ' + s.last_name [Student name], COUNT(e.enrollment_id) [no. of enrollments]
FROM Students s
JOIN Enrollments e ON s.student_id = e.student_id
GROUP BY s.student_id, s.first_name, s.last_name
```

	student_id	Student name	no. of enrollments
1	1	Zain Khaled	2
2	2	Ahmed Ali	2
3	3	Ziad Loay	2

(iv) Use UNION to combine results of two different SELECT queries

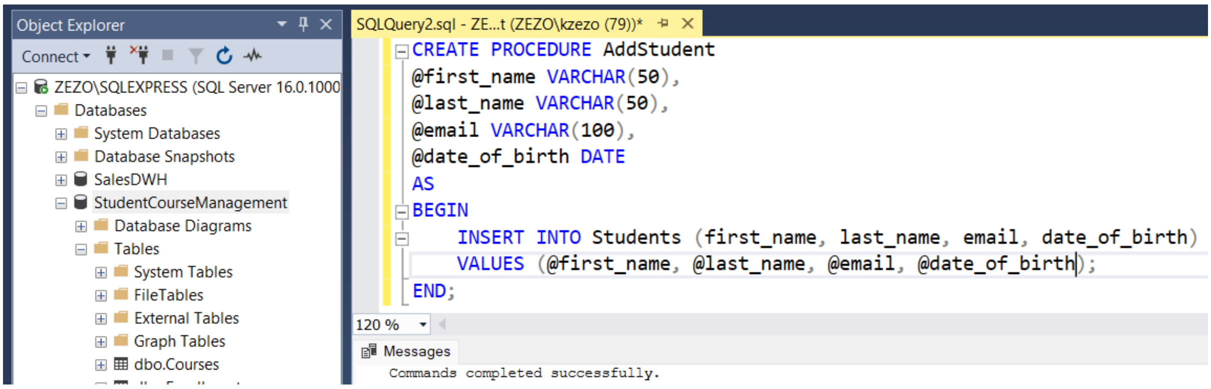


```sql
SELECT first_name+' '+ last_name [ Student name ] FROM Students
UNION
SELECT first_name+' '+ last_name [ Instructor name ] FROM Instructors;
```

	Student name
1	Abdullah Zaky
2	Ahmed Ali
3	Ahmed Azab
4	Anter Mohammed
5	Ayman Hegazy
6	Mohammed Diab
7	Mostafa Said
8	Salama Ahmed
9	Samir Ahmed
10	Sara Ali
11	Waleed Ahmed
12	Zain Khaled
13	Ziad Loay

8) Functions and Stored Procedures:

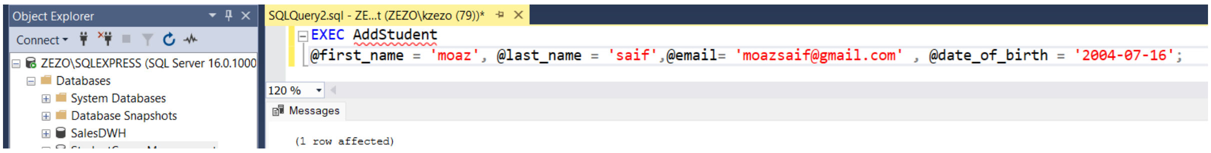
(i) Create a stored procedure to add a new student



```sql
CREATE PROCEDURE AddStudent
@first_name VARCHAR(50),
@last_name VARCHAR(50),
@email VARCHAR(100),
@date_of_birth DATE
AS
BEGIN
    INSERT INTO Students (first_name, last_name, email, date_of_birth)
    VALUES (@first_name, @last_name, @email, @date_of_birth);
END;
```

```
Commands completed successfully.
```

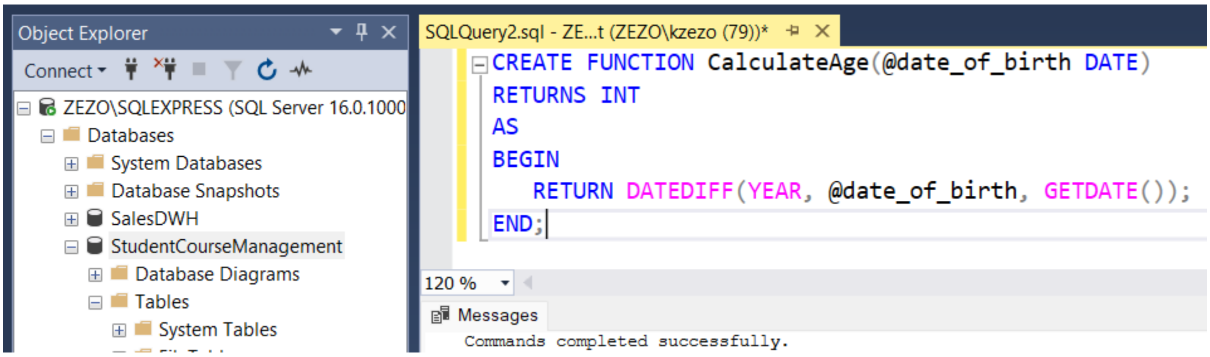
→ testing the procedure:



```sql
EXEC AddStudent
@first_name = 'moaz', @last_name = 'saif',@email= 'moazsaif@gmail.com' , @date_of_birth = '2004-07-16';
```

```
(1 row affected)
```

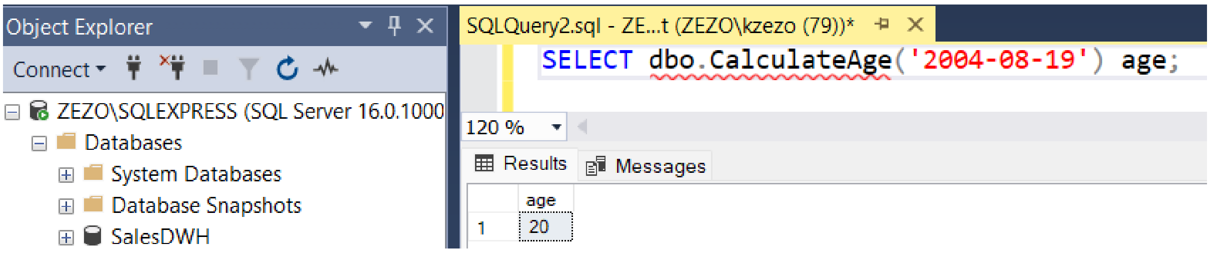
(ii) Create a function to calculate the age of a student based on their date of birth



```sql
CREATE FUNCTION CalculateAge(@date_of_birth DATE)
RETURNS INT
AS
BEGIN
    RETURN DATEDIFF(YEAR, @date_of_birth, GETDATE());
END;
```

```
Commands completed successfully.
```

→ Testing the function:

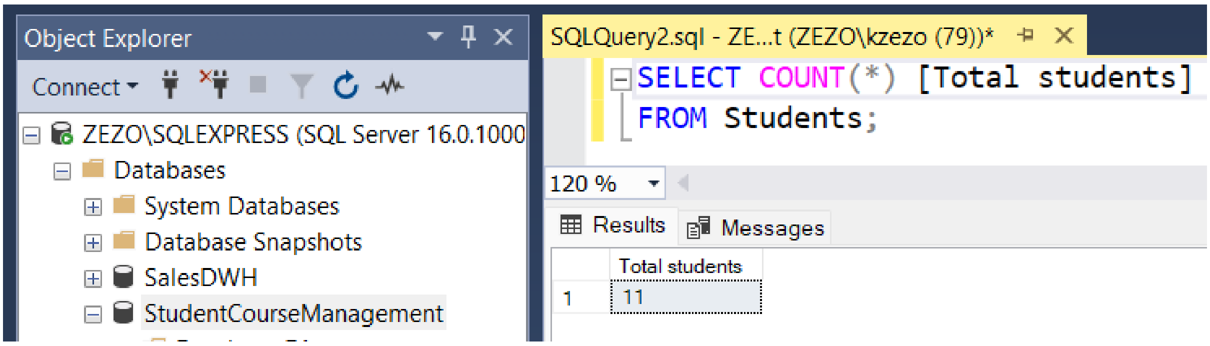


```sql
SELECT dbo.CalculateAge('2004-08-19') age;
```

	age
1	20

9) Aggregate Functions and Grouping:

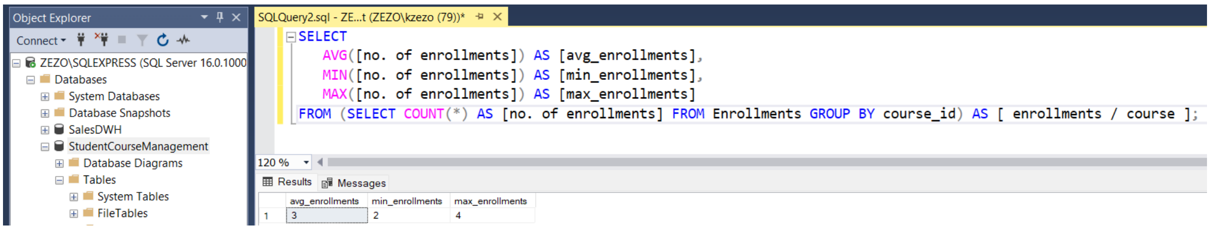
(i) Calculate the total number of students



```sql
SELECT COUNT(*) [Total students]
FROM Students;
```

	Total students
1	11

(ii) Calculate the average, minimum, and maximum number of enrollments per course

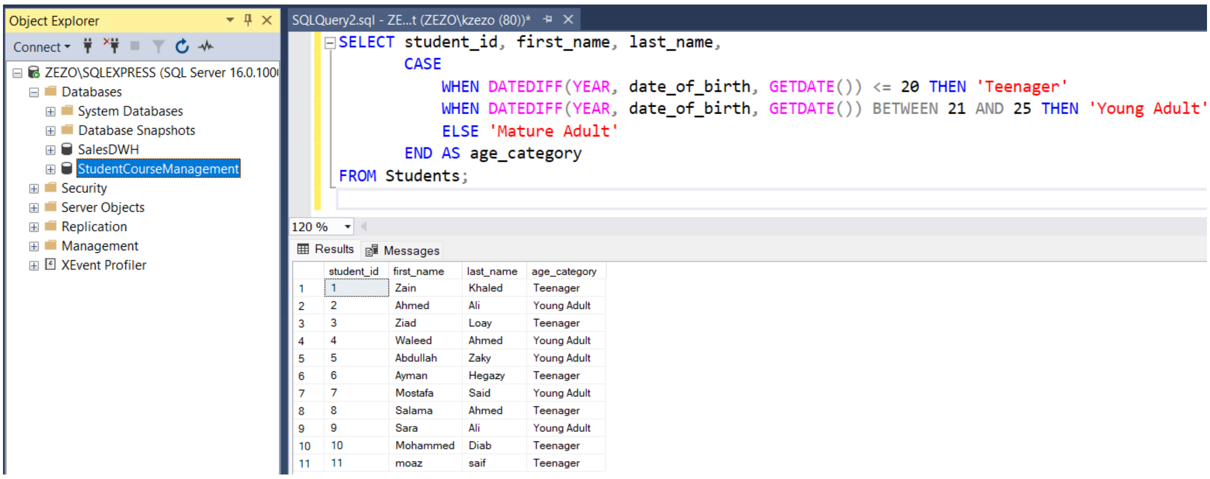


```sql
SELECT
    AVG([no. of enrollments]) AS [avg_enrollments],
    MIN([no. of enrollments]) AS [min_enrollments],
    MAX([no. of enrollments]) AS [max_enrollments]
FROM (SELECT COUNT(*) AS [no. of enrollments] FROM Enrollments GROUP BY course_id) AS [ enrollments / course ];
```

	avg_enrollments	min_enrollments	max_enrollments
1	3	2	4

10) Additional Tasks:

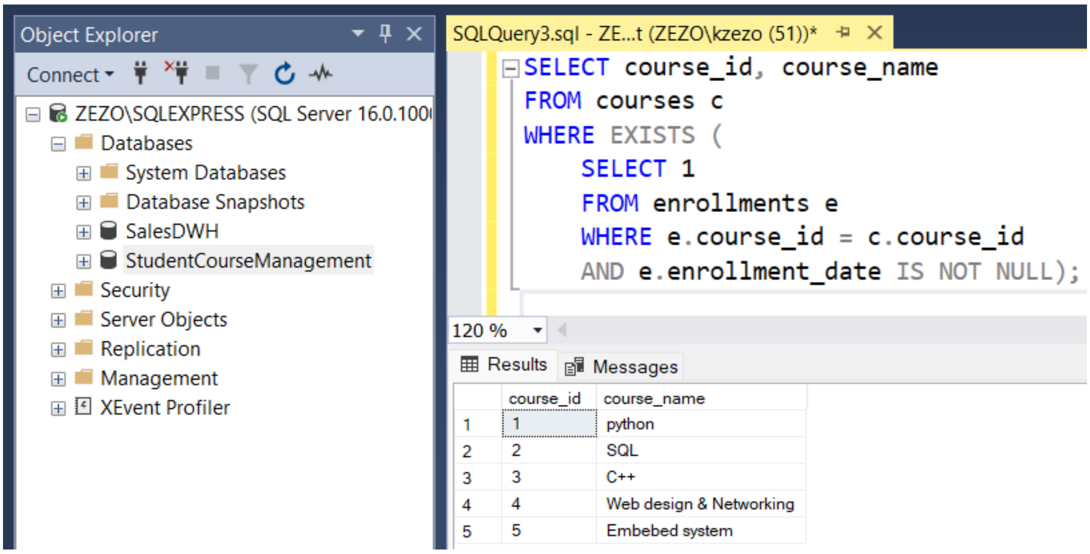
(i) Use CASE to categorize students based on their age.



```sql
SELECT student_id, first_name, last_name,
       CASE
           WHEN DATEDIFF(YEAR, date_of_birth, GETDATE()) <= 20 THEN 'Teenager'
           WHEN DATEDIFF(YEAR, date_of_birth, GETDATE()) BETWEEN 21 AND 25 THEN 'Young Adult'
           ELSE 'Mature Adult'
       END AS age_category
FROM Students;
```

	student_id	first_name	last_name	age_category
1	1	Zain	Khaled	Teenager
2	2	Ahmed	Ali	Young Adult
3	3	Ziad	Loay	Teenager
4	4	Waleed	Ahmed	Young Adult
5	5	Abdullah	Zaky	Young Adult
6	6	Ayman	Hegazy	Teenager
7	7	Mostafa	Said	Young Adult
8	8	Salama	Ahmed	Teenager
9	9	Sara	Ali	Young Adult
10	10	Mohammed	Diab	Teenager
11	11	moaz	saif	Teenager

(ii) Use EXISTS to find courses with at least one enrolled student.



```
SELECT course_id, course_name
FROM courses c
WHERE EXISTS (
    SELECT 1
    FROM enrollments e
    WHERE e.course_id = c.course_id
    AND e.enrollment_date IS NOT NULL);
```

	course_id	course_name
1	1	python
2	2	SQL
3	3	C++
4	4	Web design & Networking
5	5	Embebed system