

# SQL Project: Student Course Management System

## 1. Introduction

The "Student Course Management System" project is designed to develop a database that manages student information, course details, enrollments, and instructors. This report documents the entire process, from database creation to executing SQL queries that handle data manipulation and retrieval. The report also includes visual outputs for each query to demonstrate the results.

## 2. Project Overview

### Project Overview:

This project involves creating a database for managing student courses. The database includes tables for students, courses, enrollments, and instructors, and covers SQL topics such as selection, filtering, aggregation, joins, and subqueries.

### Project Requirements:

- **Database Setup:**  
Create a database named Student Course Management.
- **Table Creation:**  
Create the following tables with appropriate data types and constraints:
  - **Students:** student\_id, first\_name, last\_name, email, date\_of\_birth
  - **Courses:** course\_id, course\_name, course\_description
  - **Instructors:** instructor\_id, first\_name, last\_name, email
  - **Enrollments:** enrollment\_id, student\_id, course\_id, enrollment\_date
- **Insert Sample Data:**  
Insert at least 10 students, 5 courses, 3 instructors, and 15 enrollments.
- **Basic Queries:**  
Select all students, courses, and enrollments with student and course names.

- **Advanced Queries:**

Perform specific selections, updates, and deletions, calculate averages, and group results.

- **Join Queries:**

Execute joins to retrieve related data from multiple tables.

- **Subqueries and Set Operations:**

Perform subqueries, find specific data using conditions, and combine results.

- **Functions and Stored Procedures:**

Create stored procedures and functions to manage database operations.

- **Aggregate Functions and Grouping:**

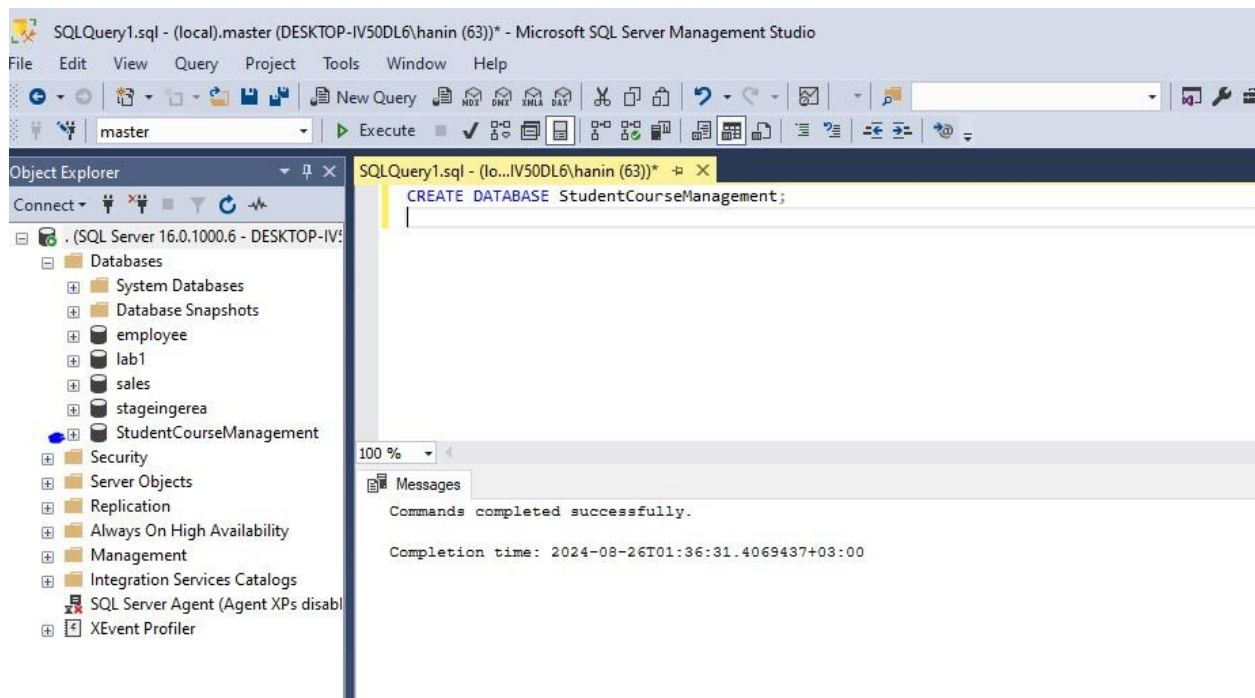
Calculate totals, averages, and other aggregations.

- **Additional Tasks:**

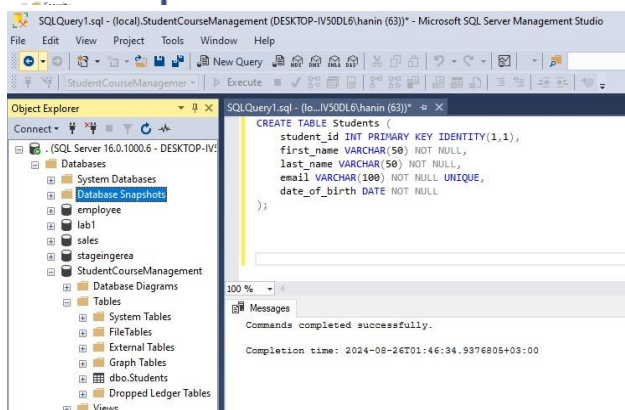
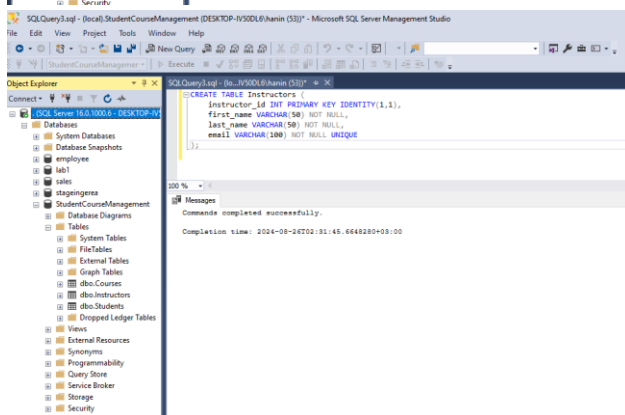
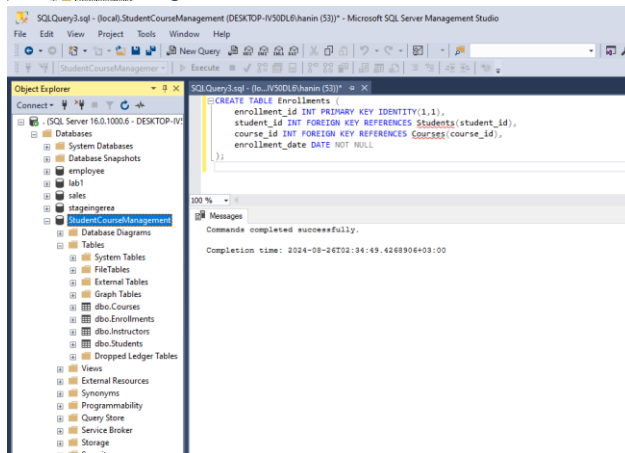
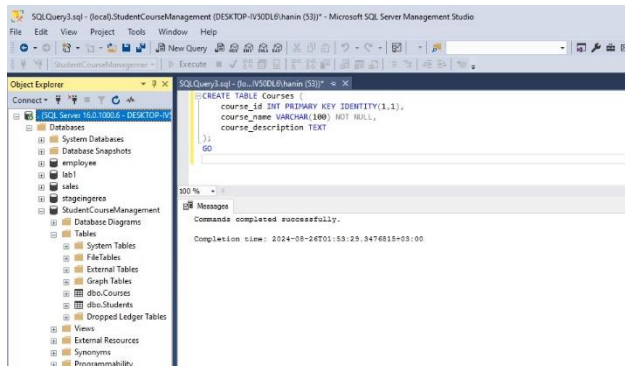
Use aliases, CASE, EXISTS, and comments to enhance SQL readability and functionality.

## \* Database Setup and Table Creation

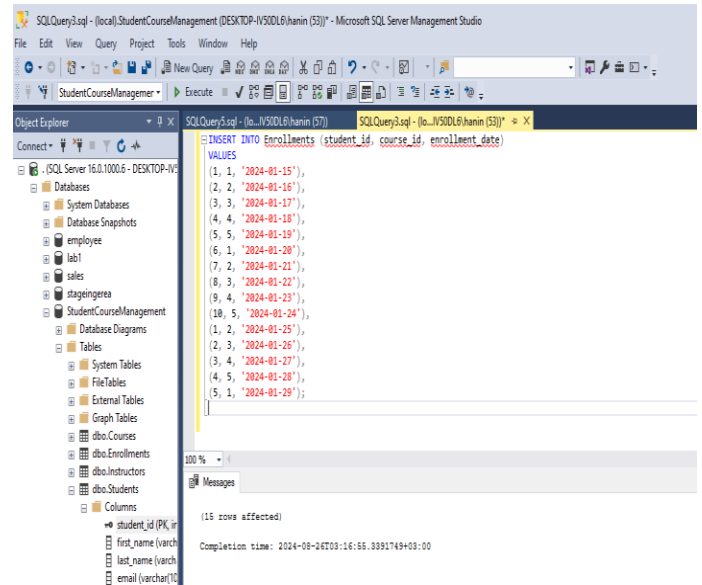
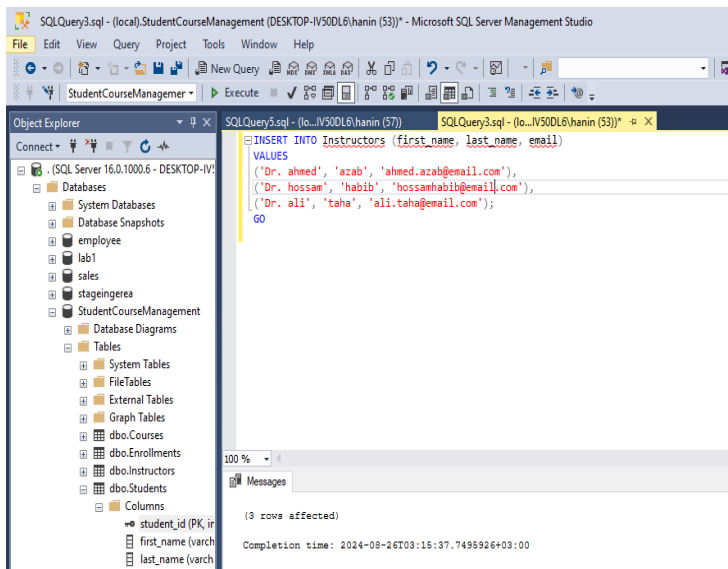
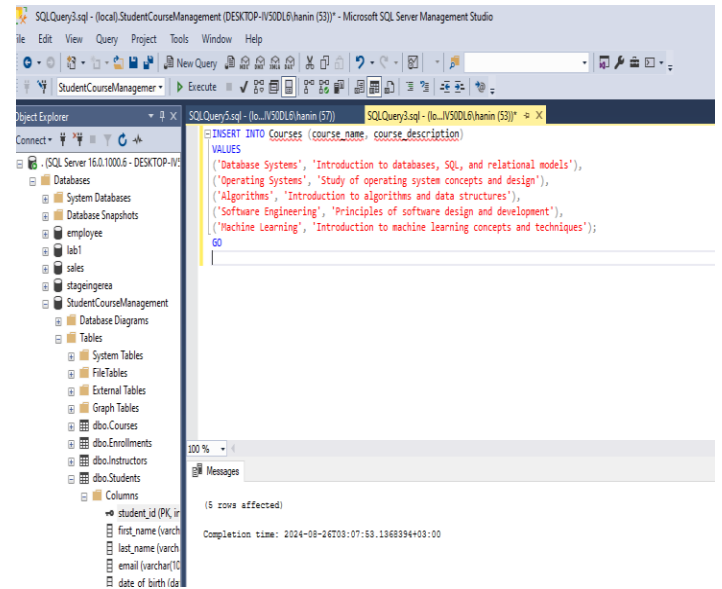
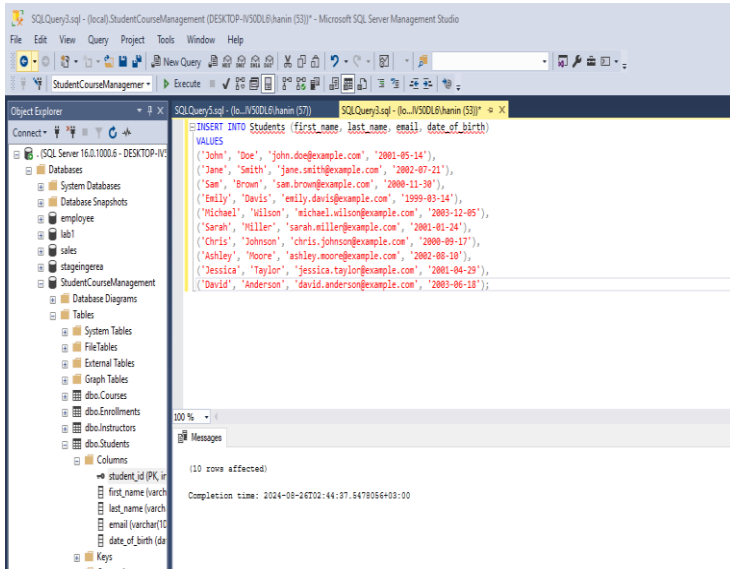
### Database Setup



## Tables Creation



## \*Inserting Data to tables



## \* Executing Basic Queries

- Select all students.

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement'. The central query window contains the SQL query: `SELECT * FROM Students;`. The Results pane at the bottom displays the output of the query, which is a table with 10 rows and 5 columns: `student_id`, `first_name`, `last_name`, `email`, and `date_of_birth`.

student_id	first_name	last_name	email	date_of_birth
1	hanin	hossam	john.doe@email.com	2001-05-14
2	hossam	Same	jane.smith@email.com	2002-07-21
3	habib	Brown	sam.brown@email.com	2000-11-30
4	emea	Davis	emily.davis@email.com	1999-03-14
5	Michael	Wilson	michael.wilson@email.com	2003-12-05
6	Sarah	Miller	sarah.miller@email.com	2001-01-24
7	naser	Johnson	chris.johnson@email.com	2000-09-17
8	mahamed	Moore	ashley.moore@email.com	2002-08-10
9	ahmed	Taylor	jessica.taylor@email.com	2001-04-29
10	aner	Anderson	david.anderson@email.com	2003-06-18

- Select all courses.

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement'. The central query window contains the SQL query: `SELECT * FROM Courses;`. The Results pane at the bottom displays the output of the query, which is a table with 5 rows and 3 columns: `course_id`, `course_name`, and `course_description`.

course_id	course_name	course_description
1	Database Systems	Introduction to databases, SQL, and relational mo...
2	Operating Systems	Study of operating system concepts and design
3	Algorithms	Introduction to algorithms and data structures
4	Software Engineering	Principles of software design and development
5	Machine Learning	Introduction to machine learning concepts and te...

- **Select all enrollments with student names and course names.**

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement'. The query editor in the center contains the following SQL query:

```
SELECT
    e.enrollment_id,
    s.first_name + ' ' + s.last_name AS student_name,
    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;
```

The Results pane at the bottom shows the output of the query, displaying 15 rows of enrollment data. The columns are enrollment\_id, student\_name, course\_name, and enrollment\_date.

enrollment_id	student_name	course_name	enrollment_date
1	hanin hossam	Database Systems	2024-01-15
2	hossam Same	Operating Systems	2024-01-16
3	habib Brown	Algorithms	2024-01-17
4	emea Davis	Software Engineering	2024-01-18
5	Michael Wilson	Machine Learning	2024-01-19
6	Sarah Miller	Database Systems	2024-01-20
7	naser Johnson	Operating Systems	2024-01-21
8	mahamed Moore	Algorithms	2024-01-22
9	ahmed Taylor	Software Engineering	2024-01-23
10	amer Anderson	Machine Learning	2024-01-24
11	hanin hossam	Operating Systems	2024-01-25
12	hossam Same	Algorithms	2024-01-26
13	habib Brown	Software Engineering	2024-01-27
14	emea Davis	Machine Learning	2024-01-28
15	Michael Wilson	Database Systems	2024-01-29

## \*Executing Advanced Queries

- **Select students who enrolled in a specific course.**

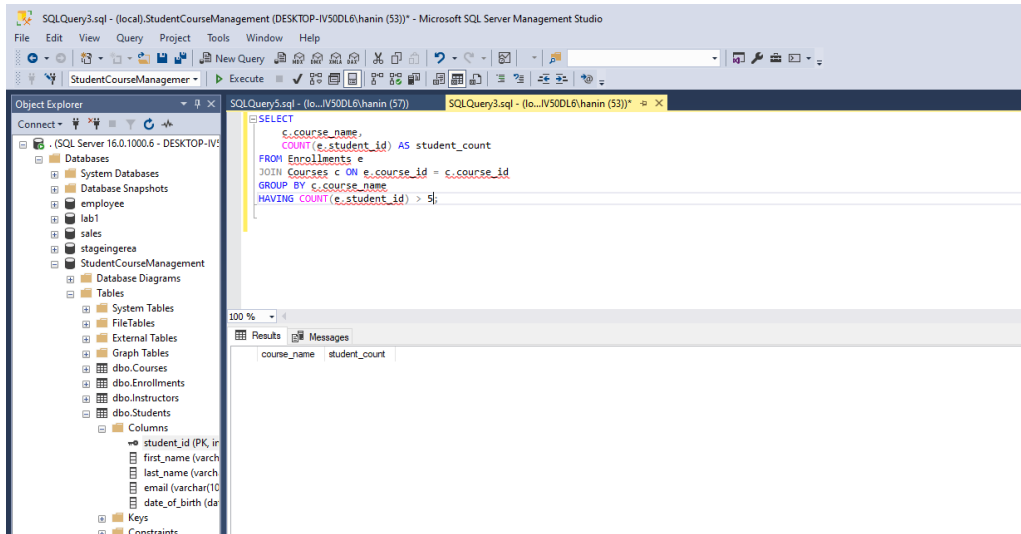
The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement'. The query editor in the center contains the following SQL query:

```
SELECT
    s.first_name,
    s.last_name
FROM Students s
JOIN Enrollments e ON s.student_id = e.student_id
WHERE e.course_id = 1;
```

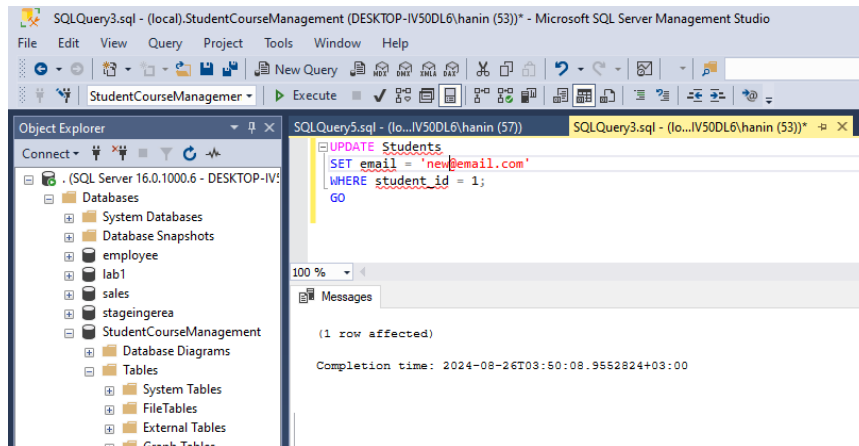
The Results pane at the bottom shows the output of the query, displaying 3 rows of student data. The columns are first\_name and last\_name.

first_name	last_name
hanin	hossam
Sarah	Miller
Michael	Wilson

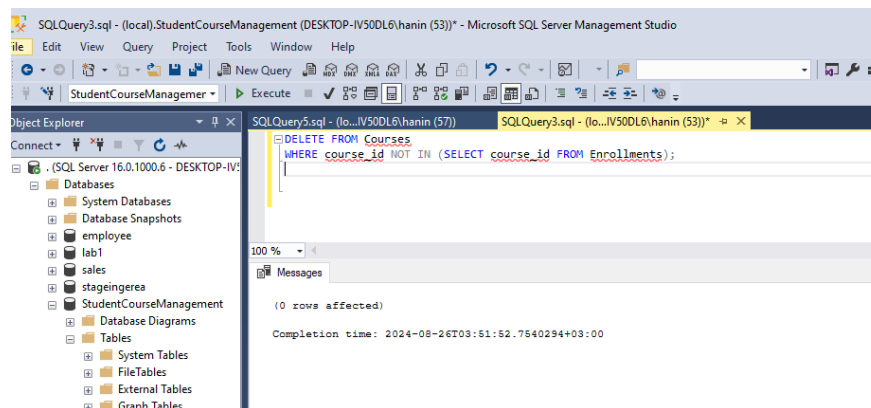
- Select courses with more than 5 students.



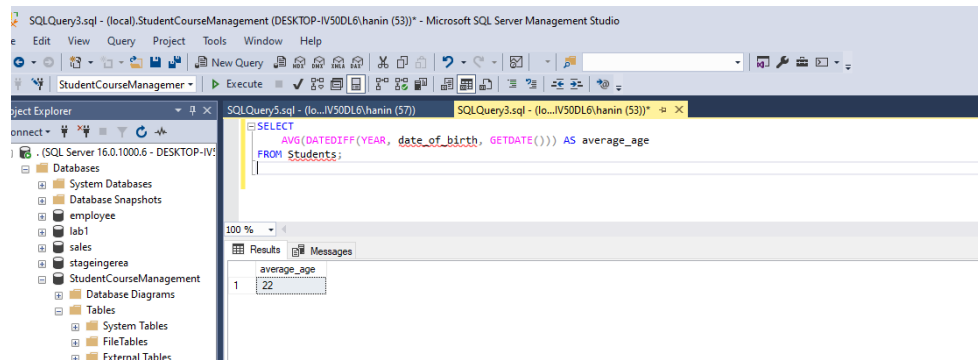
- Update a student's email.



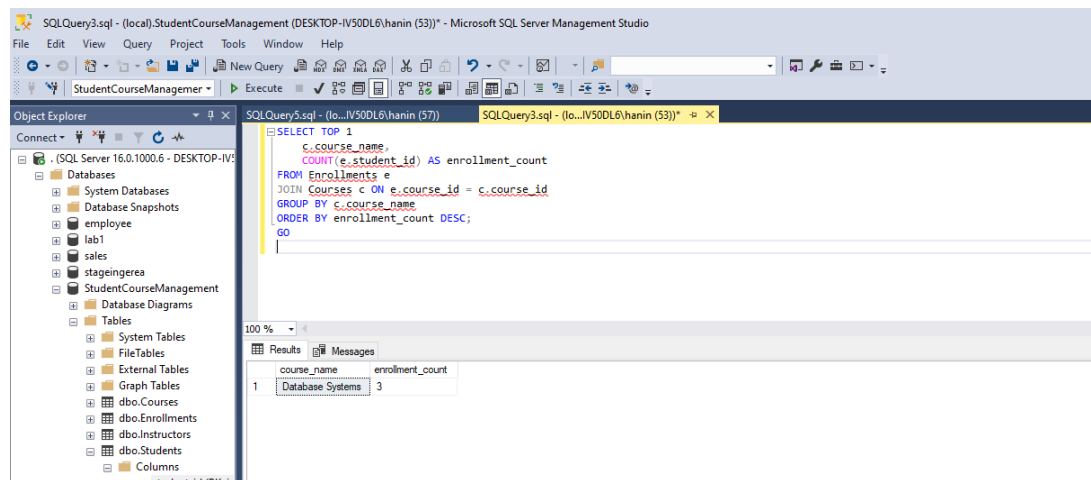
- Delete a course that no students are enrolled in.



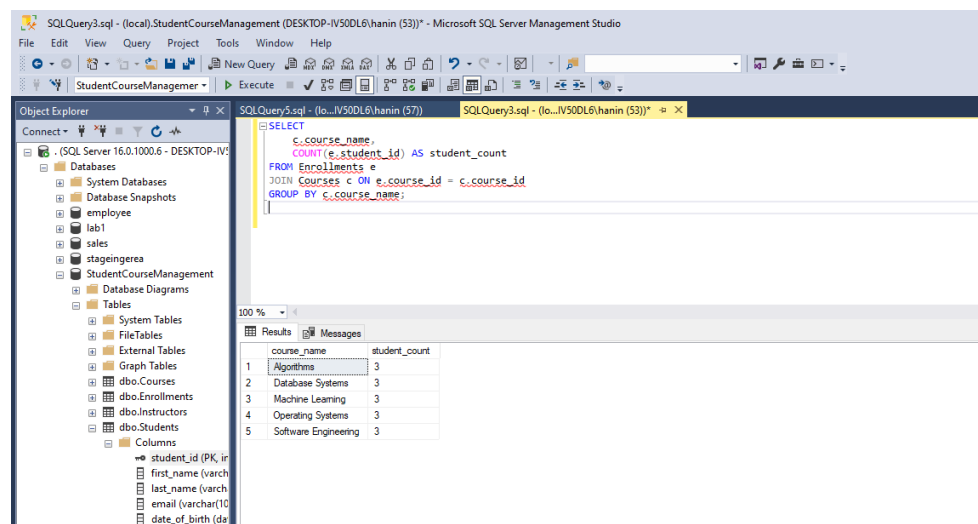
- Calculate the average age of students.



- Find the course with the maximum enrollments.



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





## \*Join Queries

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement'. The main query window contains the following SQL code:

```
SELECT
    s.first_name + ' ' + s.last_name AS student_name,
    c.course_name
FROM Students s
JOIN Enrollments e ON s.student_id = e.student_id
JOIN Courses c ON e.course_id = c.course_id;
```

The Results pane shows the output of the query, displaying 15 rows of student names and course names.

student_name	course_name
hanin hassam	Database Systems
hossam Same	Operating Systems
habib Brown	Algorithms
emea Davis	Software Engineering
Michael Wilson	Machine Learning
Sarah Miller	Database Systems
naser Johnson	Operating Systems
mahamed Moore	Algorithms
ahmed Taylor	Software Engineering
amer Anderson	Machine Learning
hanin hassam	Operating Systems
hossam Same	Algorithms
habib Brown	Software Engineering
emea Davis	Machine Learning
Michael Wilson	Database Systems

- List all instructors and their courses.

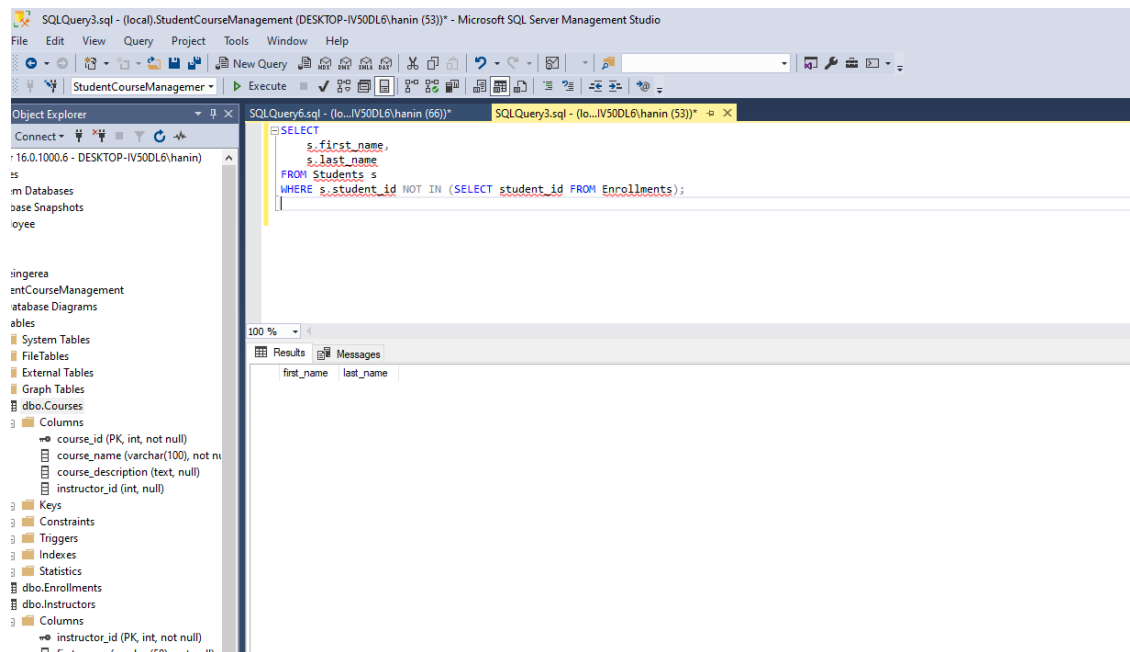
The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement'. The main query window contains the following SQL code:

```
SELECT
    i.first_name + ' ' + i.last_name AS instructor_name,
    c.course_name
FROM Instructors i
JOIN Courses c ON i.instructor_id = c.instructor_id;
```

The Results pane shows the output of the query, displaying 5 rows of instructor names and course names.

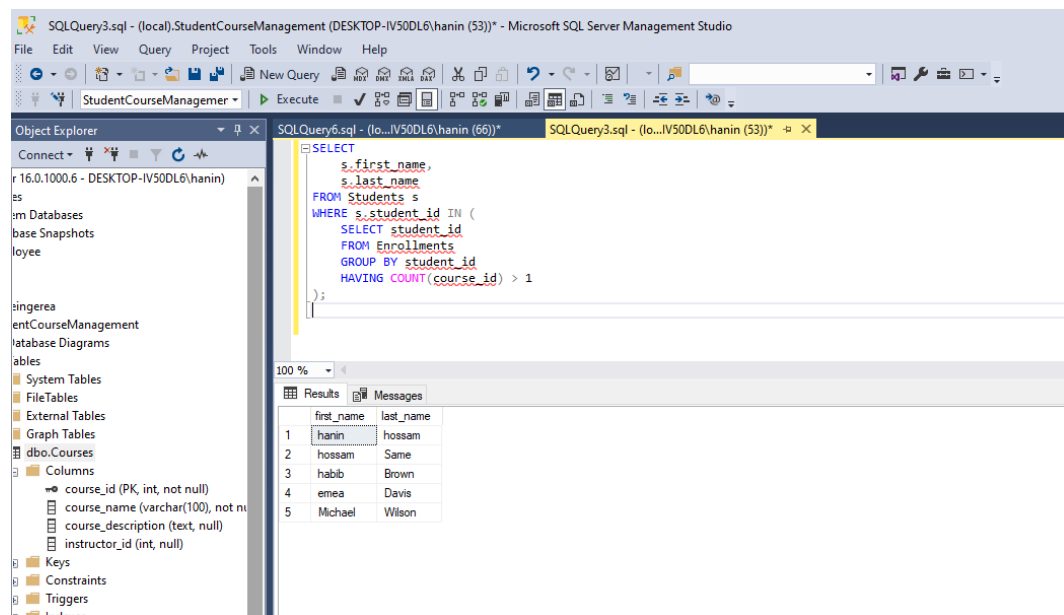
instructor_name	course_name
Dr. ahmed azab	Database Systems
Dr. ali taha	Operating Systems
Dr. ali taha	Algorithms
Dr. hossam habib	Software Engineering
Dr. ahmed azab	Machine Learning

- Find students who are not enrolled in any course.



## Subqueries and Set Operations

- Select students enrolled in more than one course.



- Find courses taught by a specific instructor.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement', including tables like 'Courses' and 'Instructors'. The query window on the right contains the following SQL query:

```
SELECT
    course_name
FROM
    Courses
WHERE
    instructor_id = (
        SELECT
            instructor_id
        FROM
            Instructors
        WHERE
            first_name = 'Dr. ali'
    );
```

The Results pane at the bottom shows the output of the query:

course_name
Operating Systems
Algorithms

- Select the top 3 students with the most enrollments.

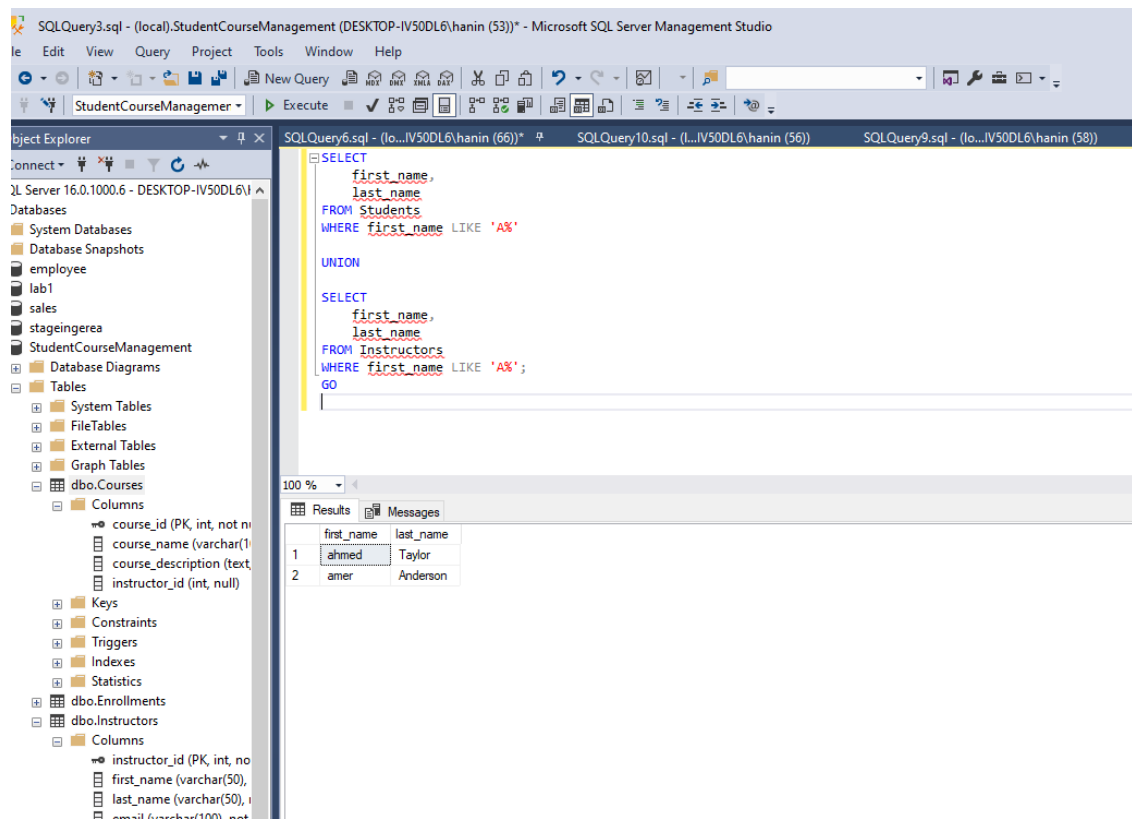
The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement', including tables like 'Students', 'Enrollments', and 'Instructors'. The query window on the right contains the following SQL query:

```
SELECT TOP 3
    s.first_name,
    s.last_name,
    (SELECT COUNT(*)
     FROM Enrollments e
     WHERE e.student_id = s.student_id) AS enrollment_count
FROM
    Students s
ORDER BY
    enrollment_count DESC;
```

The Results pane at the bottom shows the output of the query:

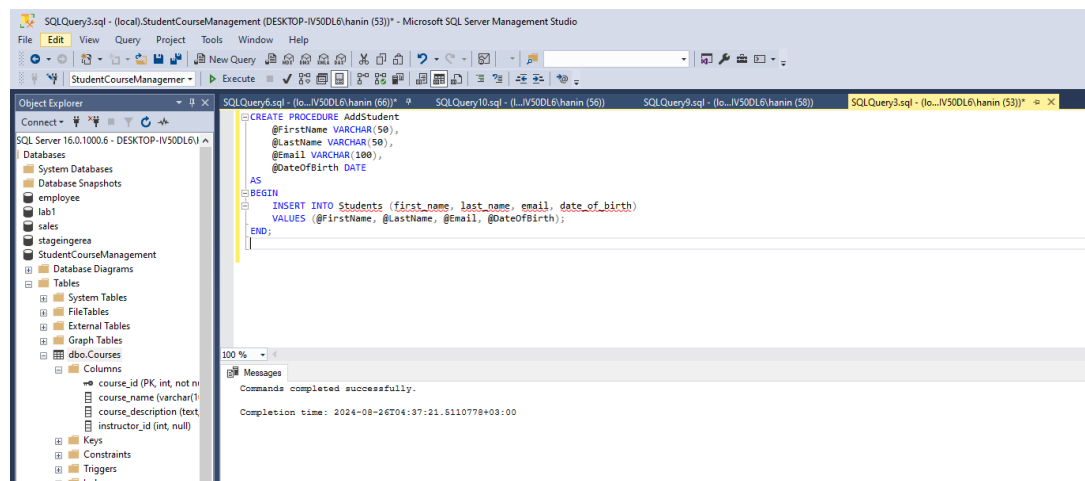
first_name	last_name	enrollment_count
emea	Davis	2
habib	Brown	2
hossam	Same	2

- Use UNION to combine results of two different SELECT queries.

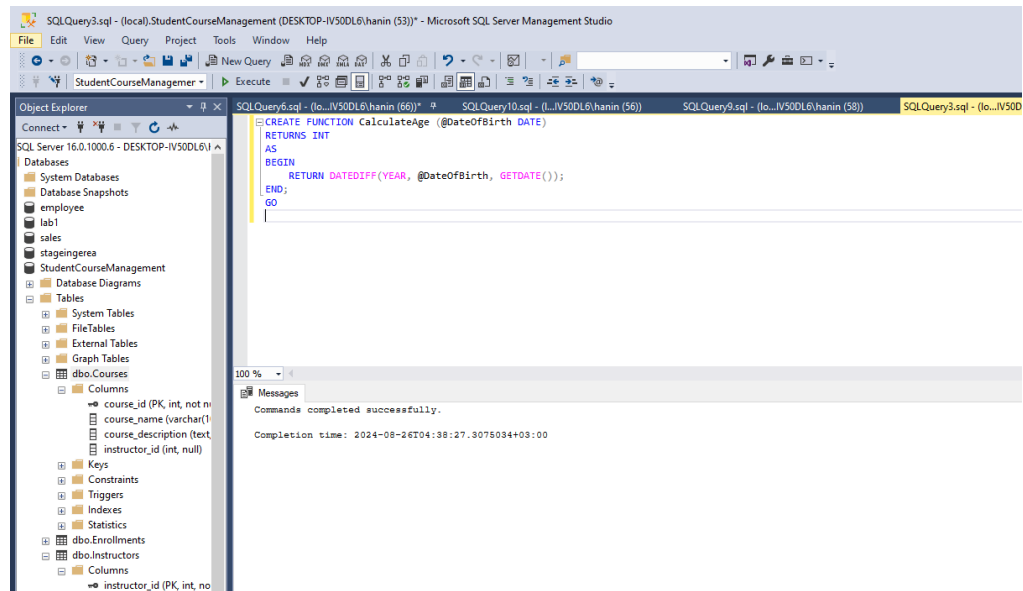


## \*Functions and Stored Procedures

- Create a stored procedure to add a new student.

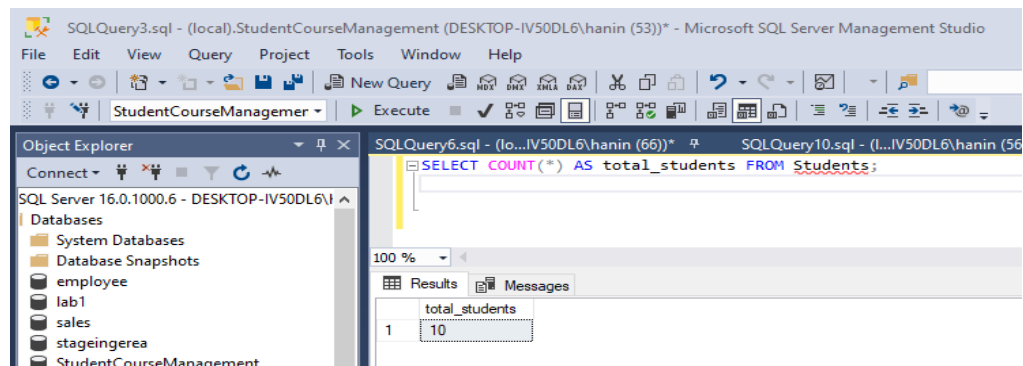


- **Create a function to calculate the age of a student based on their date of birth.**



## **\*Aggregate Functions and Grouping**

- **Calculate the total number of students.**



- **Calculate the average, minimum, and maximum number of enrollments per course.**

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement'. The query window on the right contains the following SQL code:

```
SELECT
    AVG(enrollment_count) AS avg_enrollments,
    MIN(enrollment_count) AS min_enrollments,
    MAX(enrollment_count) AS max_enrollments
FROM (
    SELECT COUNT(student_id) AS enrollment_count
    FROM Enrollments
    GROUP BY course_id
) AS EnrollmentStats;
GO
```

The Results pane at the bottom shows the output of the query:

	avg_enrollments	min_enrollments	max_enrollments
1	3	3	3

## \*Additional Tasks

- **Create aliases for complex column names.**

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for 'StudentCourseManagement'. The query window on the right contains the following SQL code:

```
SELECT
    first_name + ' ' + last_name AS student_name
FROM Students
```

The Results pane at the bottom shows the output of the query:

	student_name
1	hanin hossam
2	hossam Same
3	habib Brown
4	emea Davis
5	Michael Wilson
6	Sarah Miller
7	naser Johnson
8	mahamed Moore
9	ahmed Taylor
10	amer Anderson

- **Use CASE to categorize students based on their age.**

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the 'StudentCourseManagement' database selected. The central query editor contains the following SQL code:

```
SELECT
    first_name,
    last_name,
    CASE
        WHEN DATEDIFF(YEAR, date_of_birth, GETDATE()) < 20 THEN 'Teenager'
        WHEN DATEDIFF(YEAR, date_of_birth, GETDATE()) BETWEEN 20 AND 30 THEN 'Young Adult'
        ELSE 'Adult'
    END AS age_category
FROM Students;
```

The bottom pane shows the 'Results' tab with the following data:

	first_name	last_name	age_category
1	hanin	hossam	Young Adult
2	hossam	Same	Young Adult
3	habib	Brown	Young Adult
4	emea	Davis	Young Adult
5	Michael	Wilson	Young Adult
6	Sarah	Miller	Young Adult
7	naser	Johnson	Young Adult
8	mahamed	Moore	Young Adult
9	ahmed	Taylor	Young Adult
10	amer	Anderson	Young Adult

- **Use EXISTS to find courses with at least one enrolled student.**

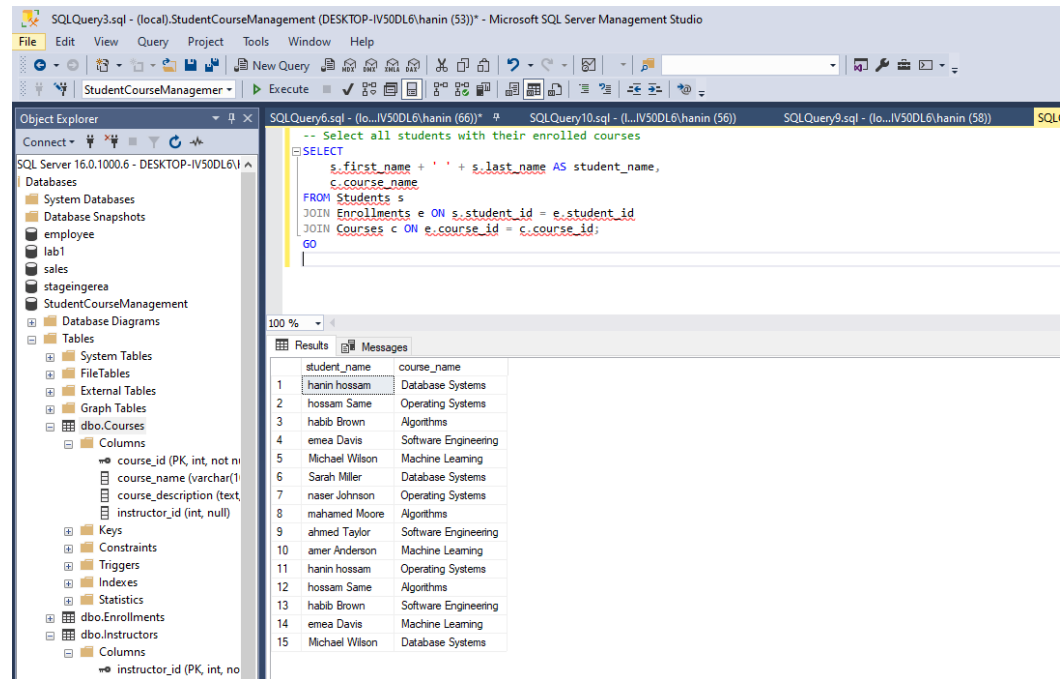
The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the 'StudentCourseManagement' database selected. The central query editor contains the following SQL code:

```
SELECT
    course_name
FROM Courses c
WHERE EXISTS (
    SELECT 1
    FROM Enrollments e
    WHERE e.course_id = c.course_id
);
GO
```

The bottom pane shows the 'Results' tab with the following data:

	course_name
1	Database Systems
2	Operating Systems
3	Algorithms
4	Software Engineering
5	Machine Learning

- **Create comments in SQL for clarity.**



## 4. Conclusion

**This project provided a practical application of SQL in managing a student course database. By designing the database, creating tables, and writing various SQL queries**