**TASK 1**

**Project Title: Academic Management System**

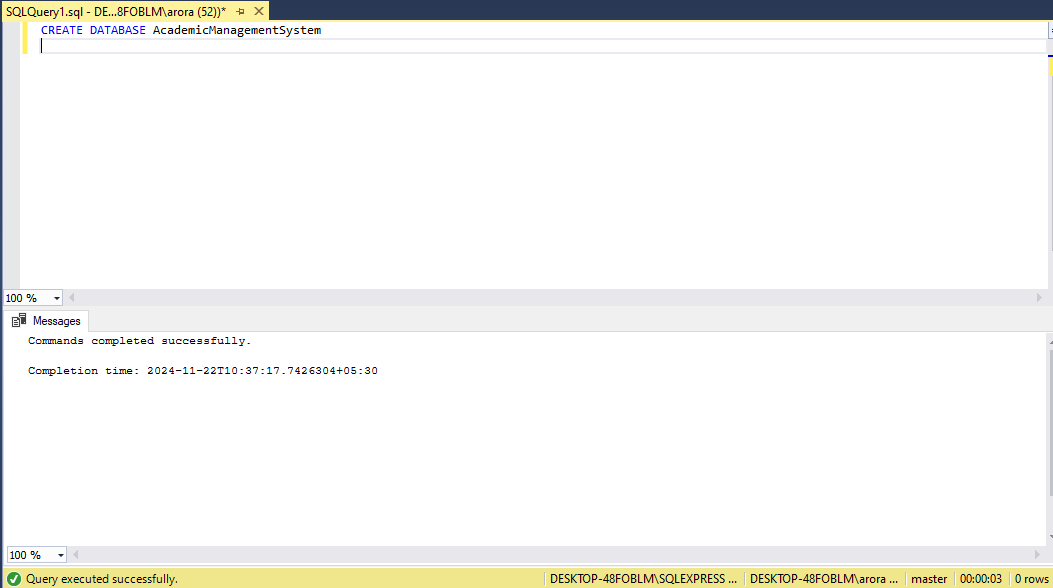
**Project Description:**

Design and develop an Academic Management System using SQL. The projects should involve three tables 1.StudentInfo 2. CoursesInfo 3.EnrollmentInfo. The Aim is to create a system that allows for managing student information and course enrollment.

**Github Link-** [**https://github.com/Dishikamehndiratta/Academic-Management-System/blob/main/README.md**](https://github.com/Dishikamehndiratta/Academic-Management-System/blob/main/README.md)

1. **Database creation  
   Code-** CREATE DATABASE AcademicManagementSystem;

USE AcademicManagementSystem;



1. **Creation of Tables**

**Code-** CREATE TABLE StudentInfo (

STU\_ID INT PRIMARY KEY IDENTITY(1,1),

STU\_NAME VARCHAR(100) NOT NULL,

DOB DATE NOT NULL,

PHONE\_NO VARCHAR(15) NOT NULL,

EMAIL\_ID VARCHAR(100) NOT NULL UNIQUE,

ADDRESS VARCHAR(255)

);

CREATE TABLE CoursesInfo (

COURSE\_ID INT PRIMARY KEY IDENTITY(1,1),

COURSE\_NAME VARCHAR(100) NOT NULL,

COURSE\_INSTRUCTOR\_NAME VARCHAR(100) NOT NULL

);

CREATE TABLE EnrollmentInfo (

ENROLLMENT\_ID INT PRIMARY KEY IDENTITY(1,1),

STU\_ID INT NOT NULL,

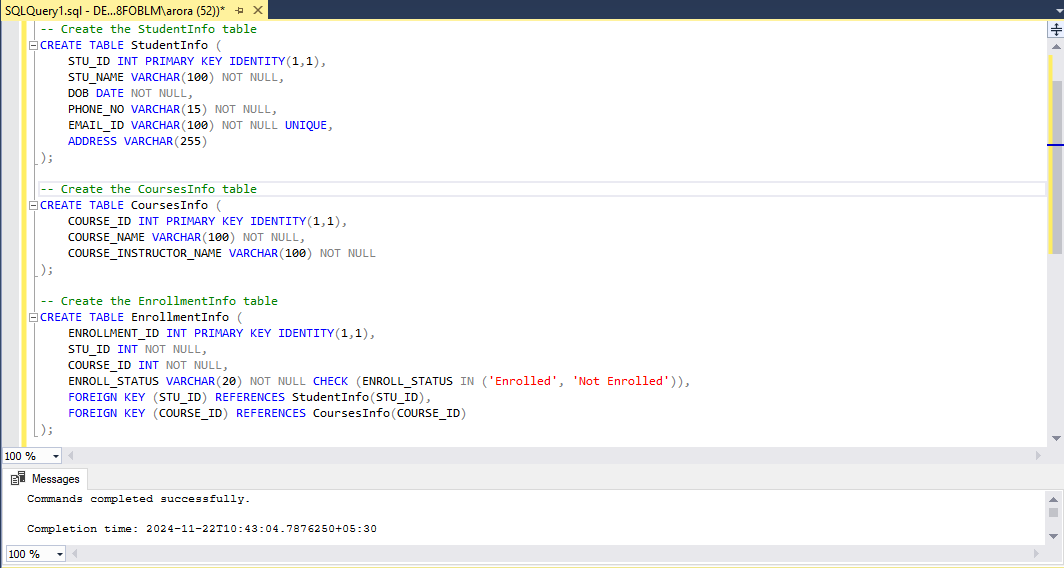
COURSE\_ID INT NOT NULL,

ENROLL\_STATUS VARCHAR(20) NOT NULL CHECK (ENROLL\_STATUS IN ('Enrolled', 'Not Enrolled')),

FOREIGN KEY (STU\_ID) REFERENCES StudentInfo(STU\_ID),

FOREIGN KEY (COURSE\_ID) REFERENCES CoursesInfo(COURSE\_ID)

);



1. **Inserting Values**

**Code-** INSERT INTO StudentInfo (STU\_NAME, DOB, PHONE\_NO, EMAIL\_ID, ADDRESS)

VALUES

('John Doe', '2000-05-15', '1234567890', 'johndoe@example.com', '123 Main Street, City A'),('Jane Smith', '1998-08-22', '9876543210', 'janesmith@example.com', '456 Elm Street, City B'),('Alice Johnson', '2001-03-10', '5551234567', 'alicej@example.com', '789 Oak Street, City C');

INSERT INTO CoursesInfo (COURSE\_NAME, COURSE\_INSTRUCTOR\_NAME)

VALUES

('Database Systems', 'Dr. James Carter'),

('Data Structures', 'Dr. Emily Davis'),

('Software Engineering', 'Dr. Robert Brown');

INSERT INTO EnrollmentInfo (STU\_ID, COURSE\_ID, ENROLL\_STATUS)

VALUES

(1, 1, 'Enrolled'),

(2, 2, 'Enrolled'),

(3, 3, 'Not Enrolled'),

(1, 3, 'Enrolled'); (1, 3, 'Enrolled');



1. **Retrieve the Student Information**

**a) Write a query to retrieve student details, such as student name, contact informations, and Enrollment status.**

**CODE-** SELECT s.STU\_NAME AS StudentName,

s.PHONE\_NO AS ContactNumber,

s.EMAIL\_ID AS Email,

e.ENROLL\_STATUS AS EnrollmentStatus

FROM

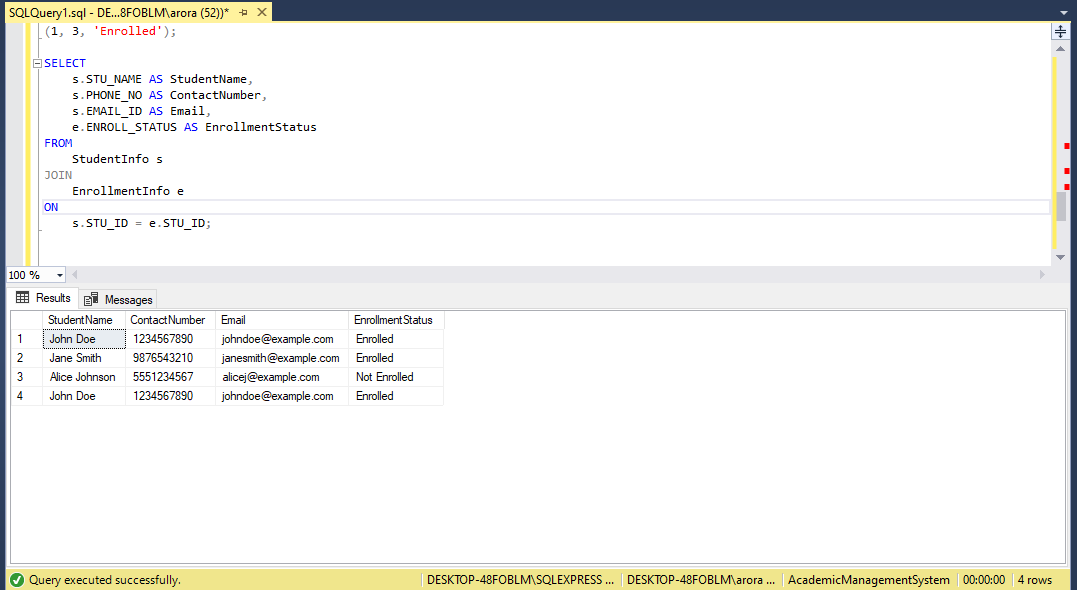
StudentInfo s

JOIN

EnrollmentInfo e

ON

s.STU\_ID = e.STU\_ID;



**b) Write a query to retrieve a list of courses in which a specific student is enrolled.**

**Code-**  c.COURSE\_NAME AS CourseName,

c.COURSE\_INSTRUCTOR\_NAME AS InstructorName,

e.ENROLL\_STATUS AS EnrollmentStatus

FROM

CoursesInfo c

JOIN

EnrollmentInfo e

ON

c.COURSE\_ID = e.COURSE\_ID

JOIN

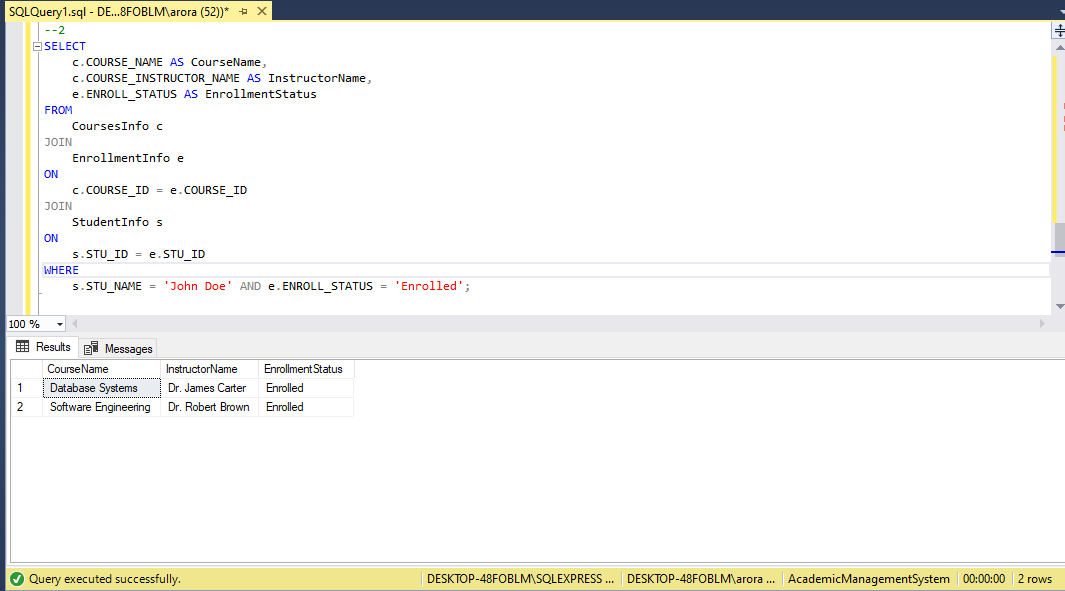
StudentInfo s

ON

s.STU\_ID = e.STU\_ID

WHERE

s.STU\_NAME = 'John Doe' AND e.ENROLL\_STATUS = 'Enrolled';



**c) Write a query to retrieve course information, including course name, instructor information.**

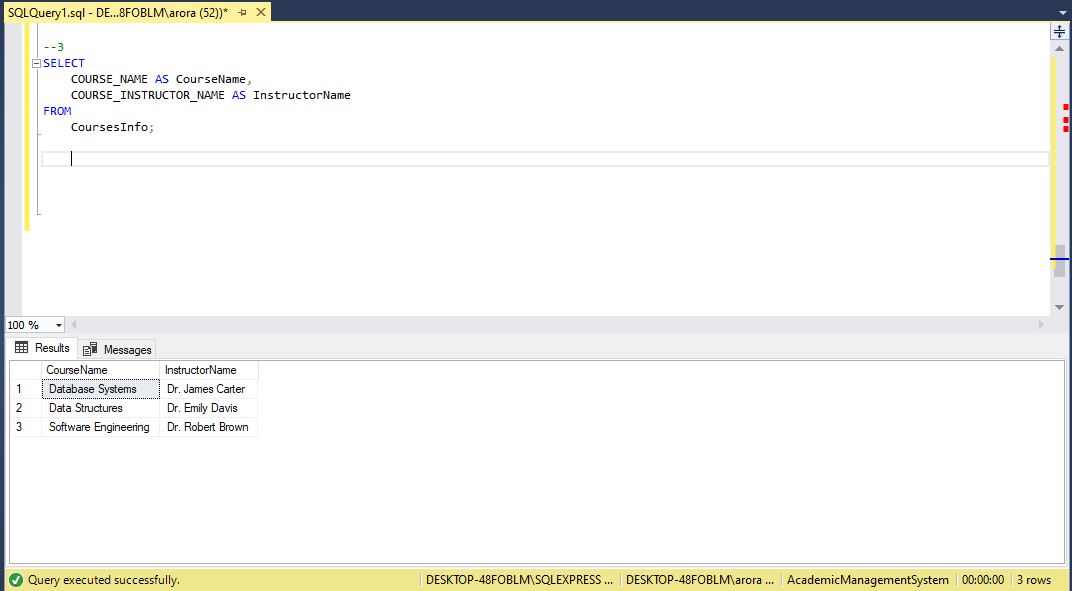
**Code-** SELECT

COURSE\_NAME AS Course\_Name,

COURSE\_INSTRUCTOR\_NAME AS Instructor\_Name

FROM

CoursesInfo;



**d) Write a query to retrieve course information for a specific course .**

**Code-** SELECT

COURSE\_NAME AS CourseName,

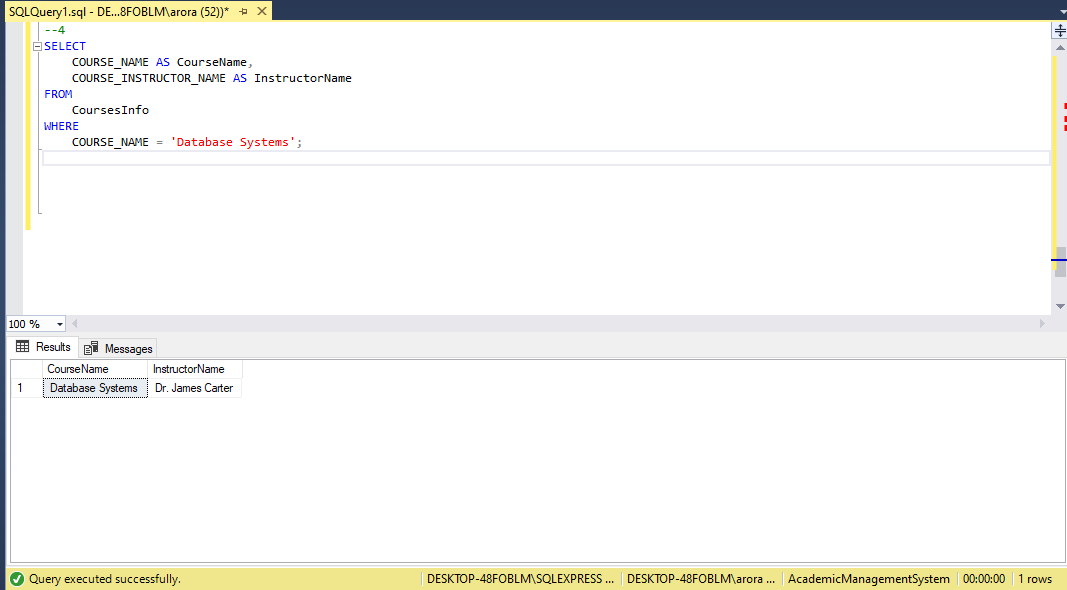
COURSE\_INSTRUCTOR\_NAME AS InstructorName

FROM

CoursesInfo

WHERE

COURSE\_NAME = 'Database Systems';



**e) Write a query to retrieve course information for multiple courses.**

**Code-** SELECT

COURSE\_NAME AS CourseName,

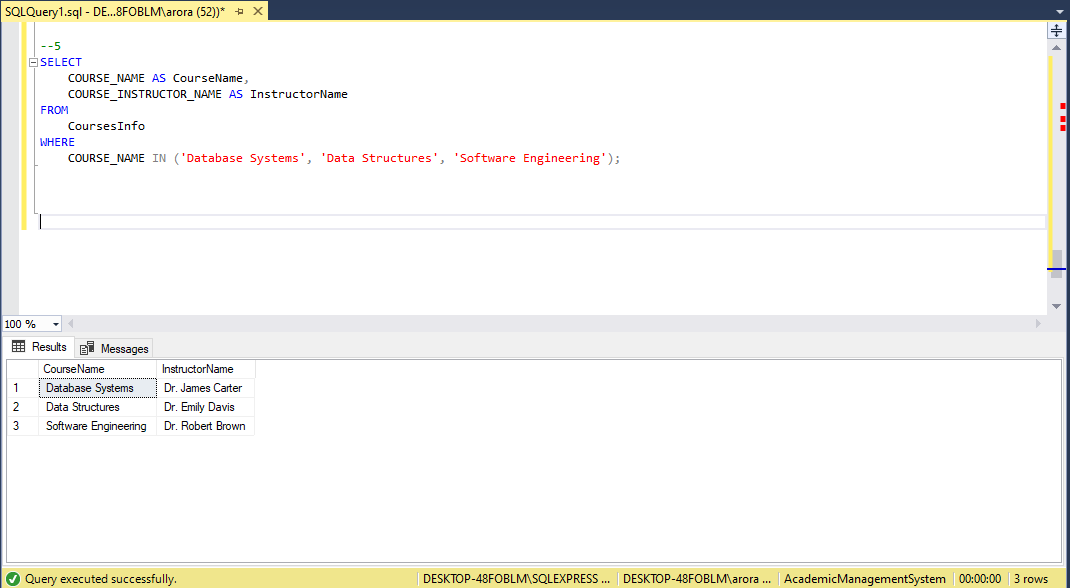
COURSE\_INSTRUCTOR\_NAME AS InstructorName

FROM

CoursesInfo

WHERE

COURSE\_NAME IN ('Database Systems', 'Data Structures', 'Software Engineering');



**5.Reporting and Analytics (Using joining queries)**

**a) Write a query to retrieve the number of students enrolled in each course**

**Code-** SELECT

c.COURSE\_NAME AS CourseName,

COUNT(e.STU\_ID) AS EnrolledStudents

FROM

CoursesInfo c

LEFT JOIN

EnrollmentInfo e

ON

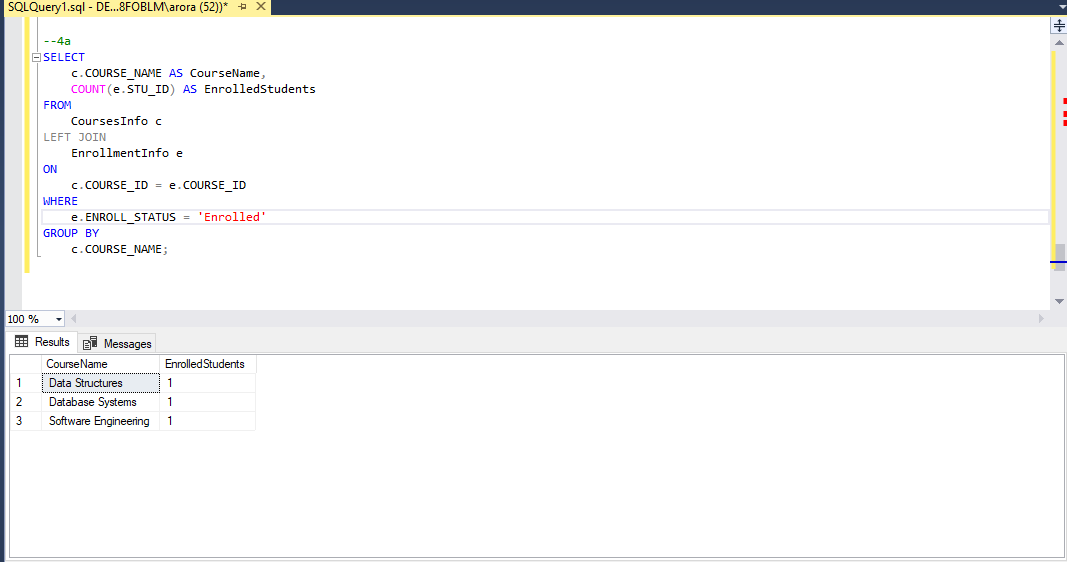
c.COURSE\_ID = e.COURSE\_ID

WHERE

e.ENROLL\_STATUS = 'Enrolled'

GROUP BY

c.COURSE\_NAME;



**b) Write a query to retrieve the list of students enrolled in a specific course**

**Code-** SELECT

s.STU\_NAME AS StudentName,

s.PHONE\_NO AS ContactNumber,

s.EMAIL\_ID AS Email

FROM

StudentInfo s

JOIN

EnrollmentInfo e

ON

s.STU\_ID = e.STU\_ID

JOIN

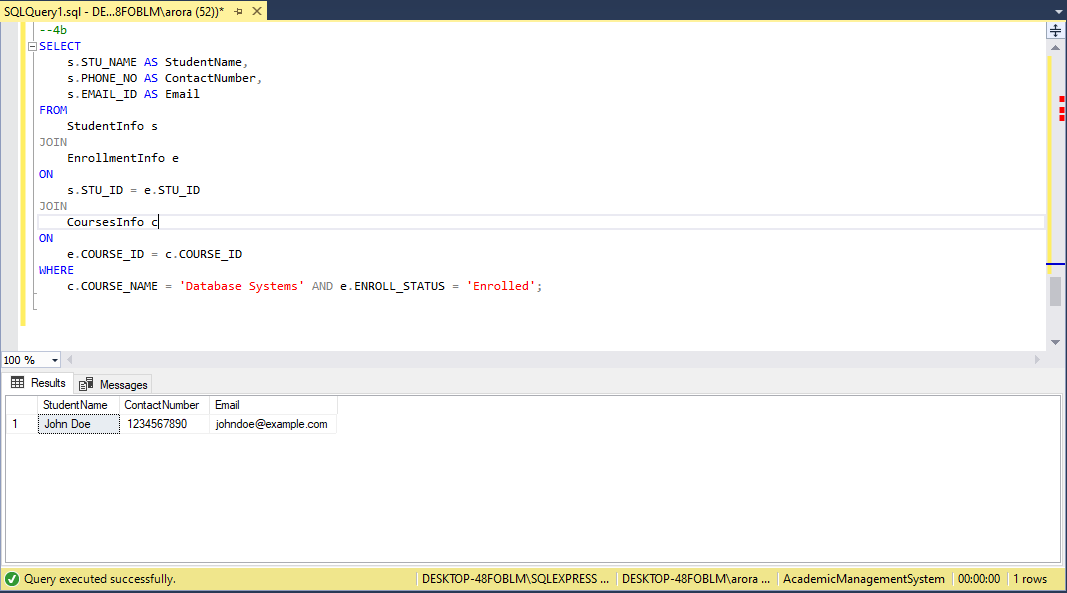
CoursesInfo c

ON

e.COURSE\_ID = c.COURSE\_ID

WHERE

c.COURSE\_NAME = 'Database Systems' AND e.ENROLL\_STATUS = 'Enrolled';



**c) Write a query to retrieve the count of enrolled students for each instructor.**

**Code-** SELECT

c.COURSE\_INSTRUCTOR\_NAME AS InstructorName,

COUNT(e.STU\_ID) AS EnrolledStudents

FROM

CoursesInfo c

JOIN

EnrollmentInfo e

ON

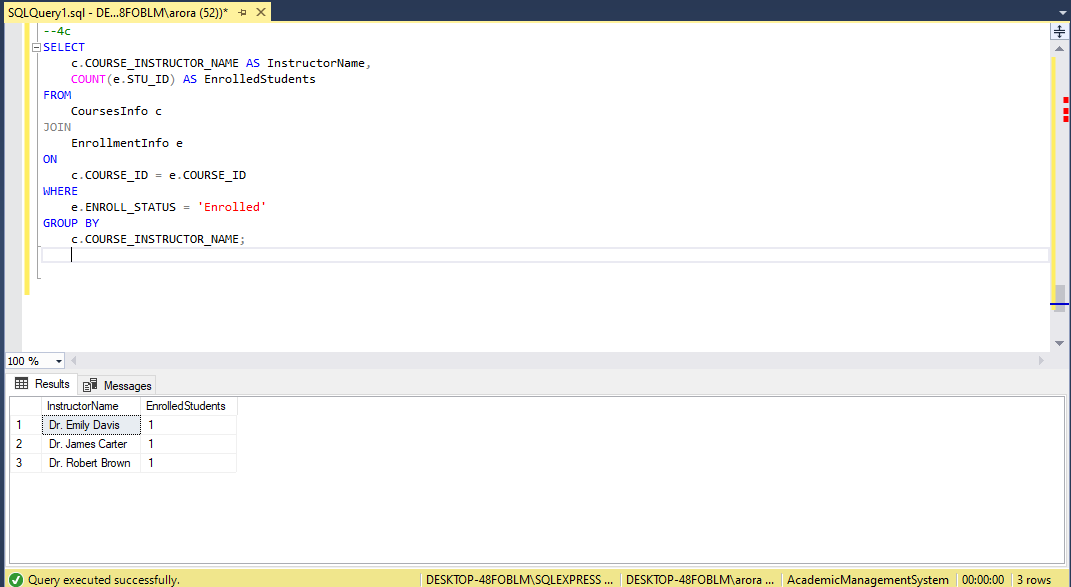
c.COURSE\_ID = e.COURSE\_ID

WHERE

e.ENROLL\_STATUS = 'Enrolled'

GROUP BY

c.COURSE\_INSTRUCTOR\_NAME;



**d) Write a query to retrieve the list of students who are enrolled in multiple courses**

**Code**- SELECT

s.STU\_NAME AS StudentName,

s.PHONE\_NO AS ContactNumber,

s.EMAIL\_ID AS Email,

COUNT(e.COURSE\_ID) AS NumberOfCourses

FROM

StudentInfo s

JOIN

EnrollmentInfo e

ON

s.STU\_ID = e.STU\_ID

WHERE

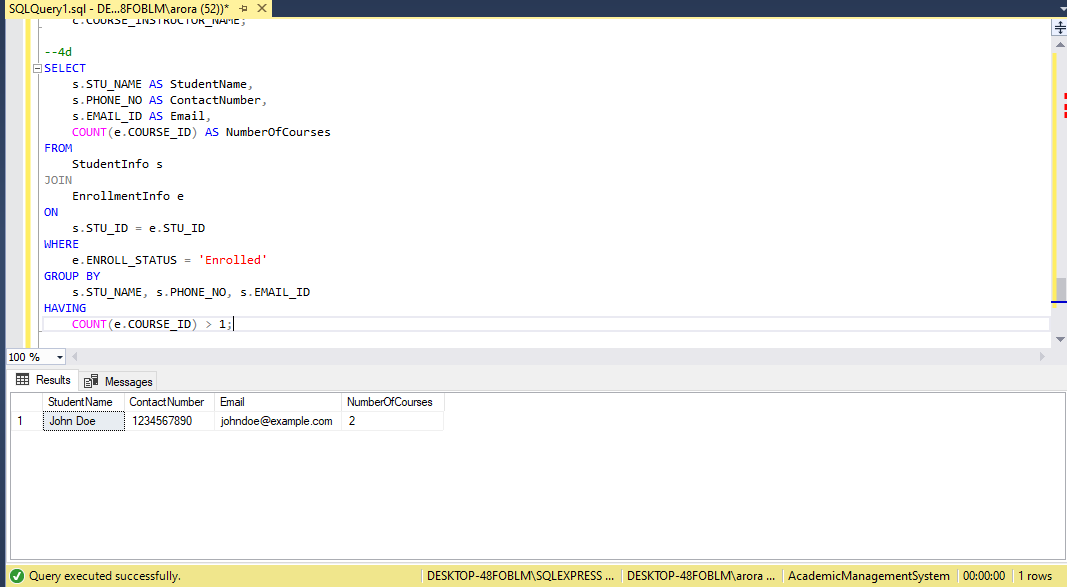
e.ENROLL\_STATUS = 'Enrolled'

GROUP BY

s.STU\_NAME, s.PHONE\_NO, s.EMAIL\_ID

HAVING

COUNT(e.COURSE\_ID) > 1;



**e) Write a query to retrieve the courses that have the highest number of enrolled students (arranging from highest to lowest)**

**Code-** SELECT

c.COURSE\_NAME AS CourseName,

c.COURSE\_INSTRUCTOR\_NAME AS InstructorName,

COUNT(e.STU\_ID) AS EnrolledStudents

FROM

CoursesInfo c

JOIN

EnrollmentInfo e

ON

c.COURSE\_ID = e.COURSE\_ID

WHERE

e.ENROLL\_STATUS = 'Enrolled'

GROUP BY

c.COURSE\_NAME, c.COURSE\_INSTRUCTOR\_NAME

ORDER BY

EnrolledStudents DESC;

