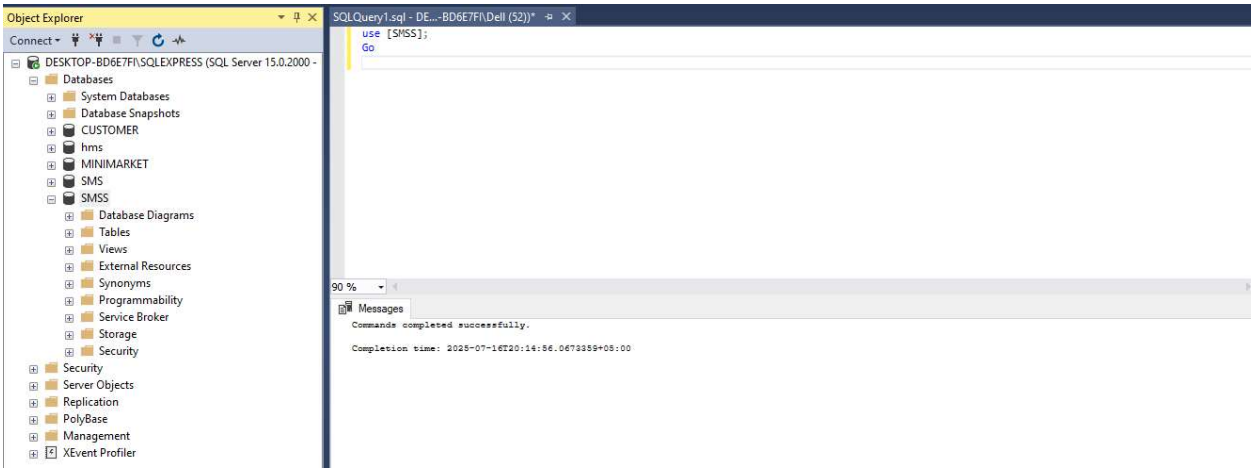
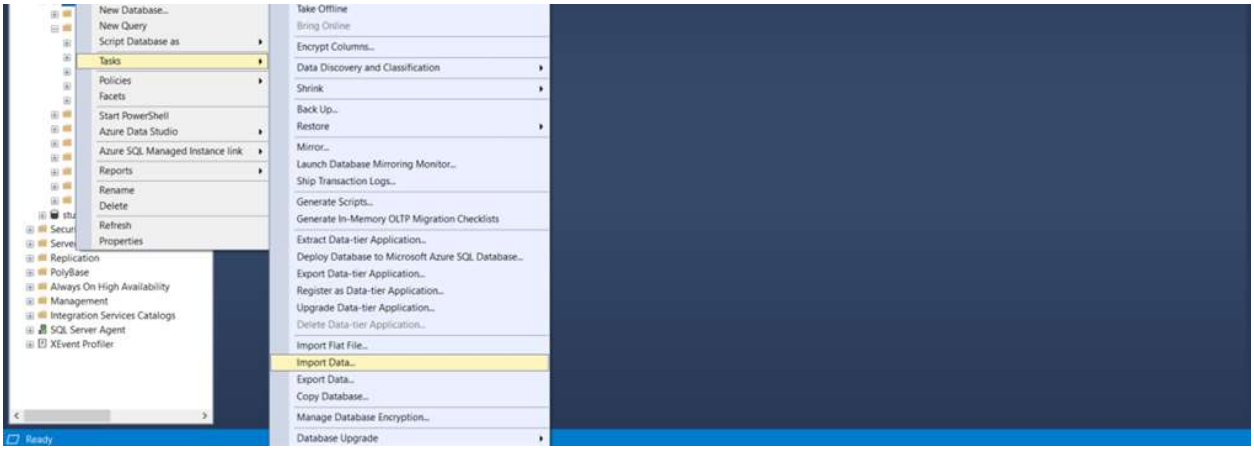
Database Project

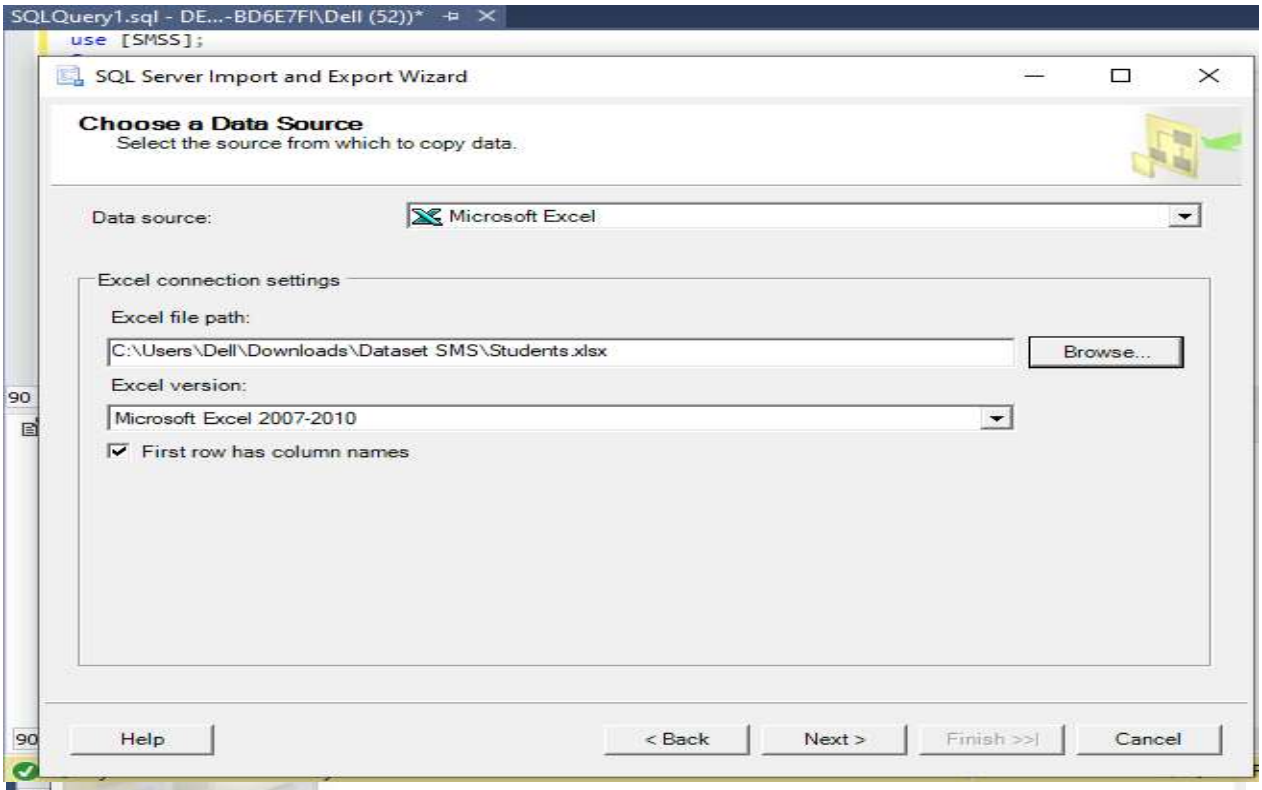
STUDENT MANAGEMENT SYSTEM

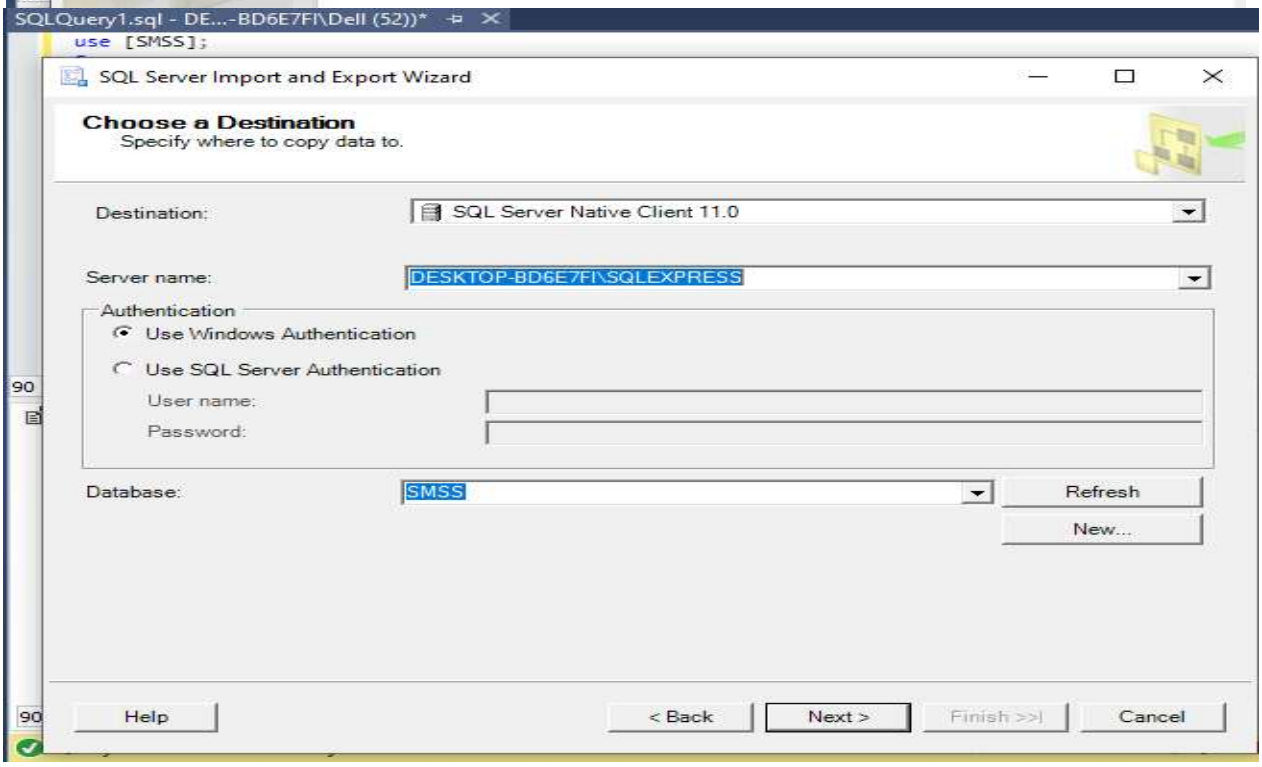
Creating Database

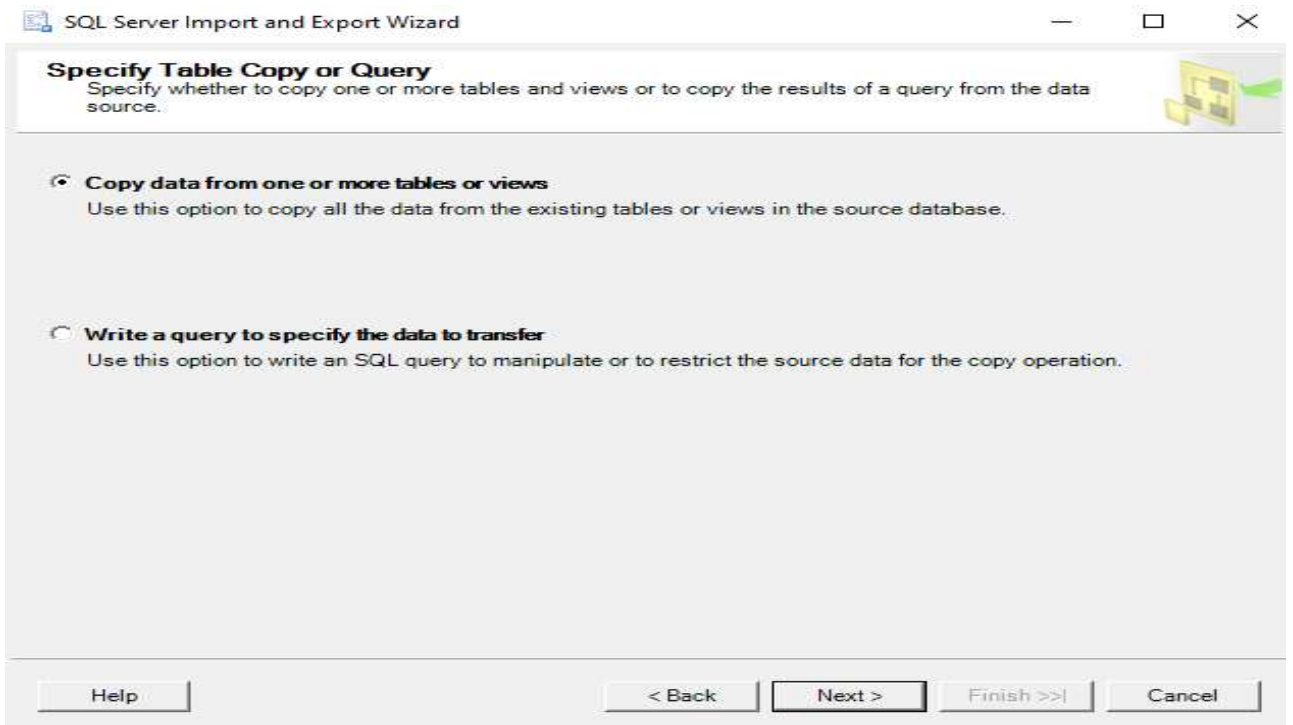


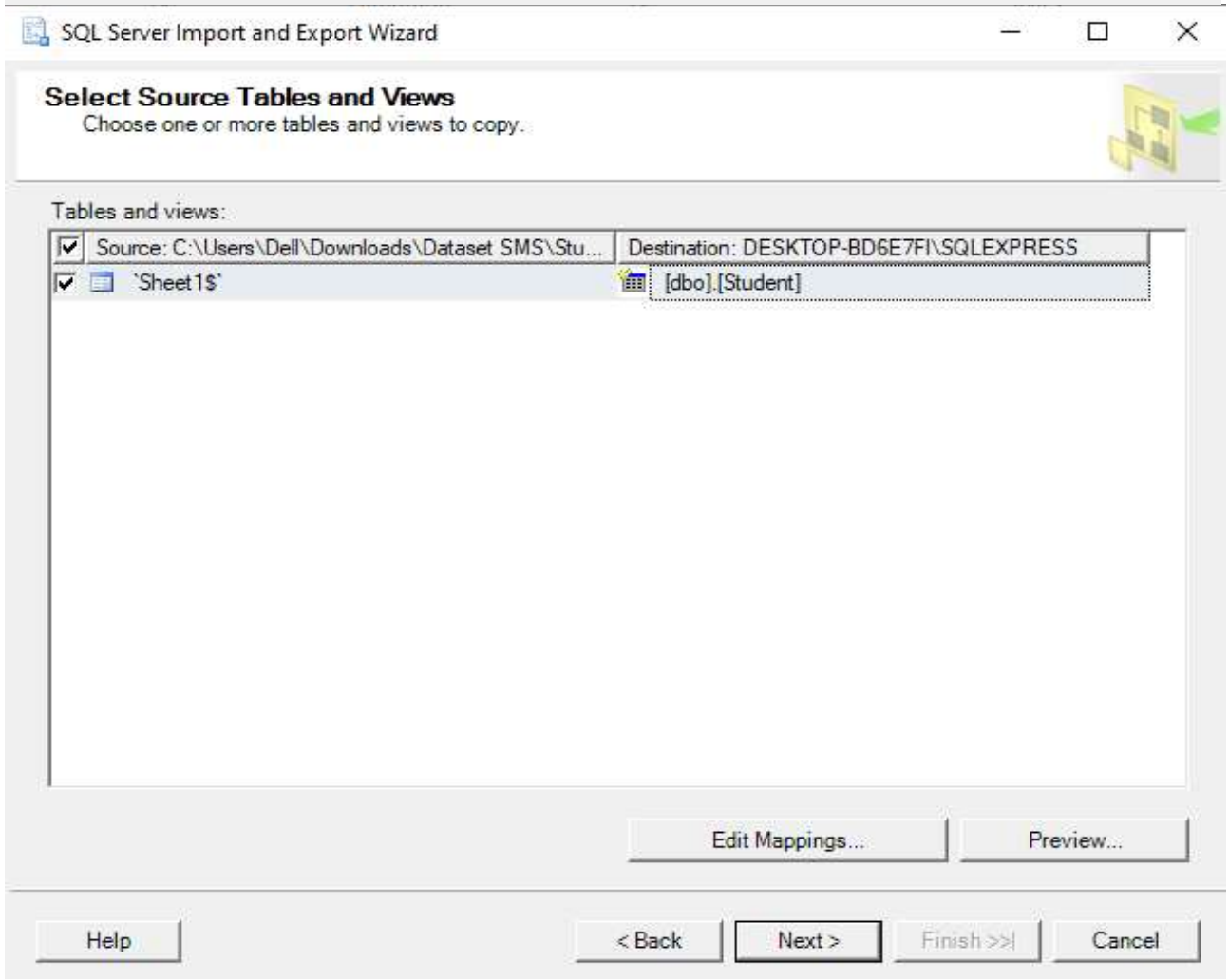
Import File Data

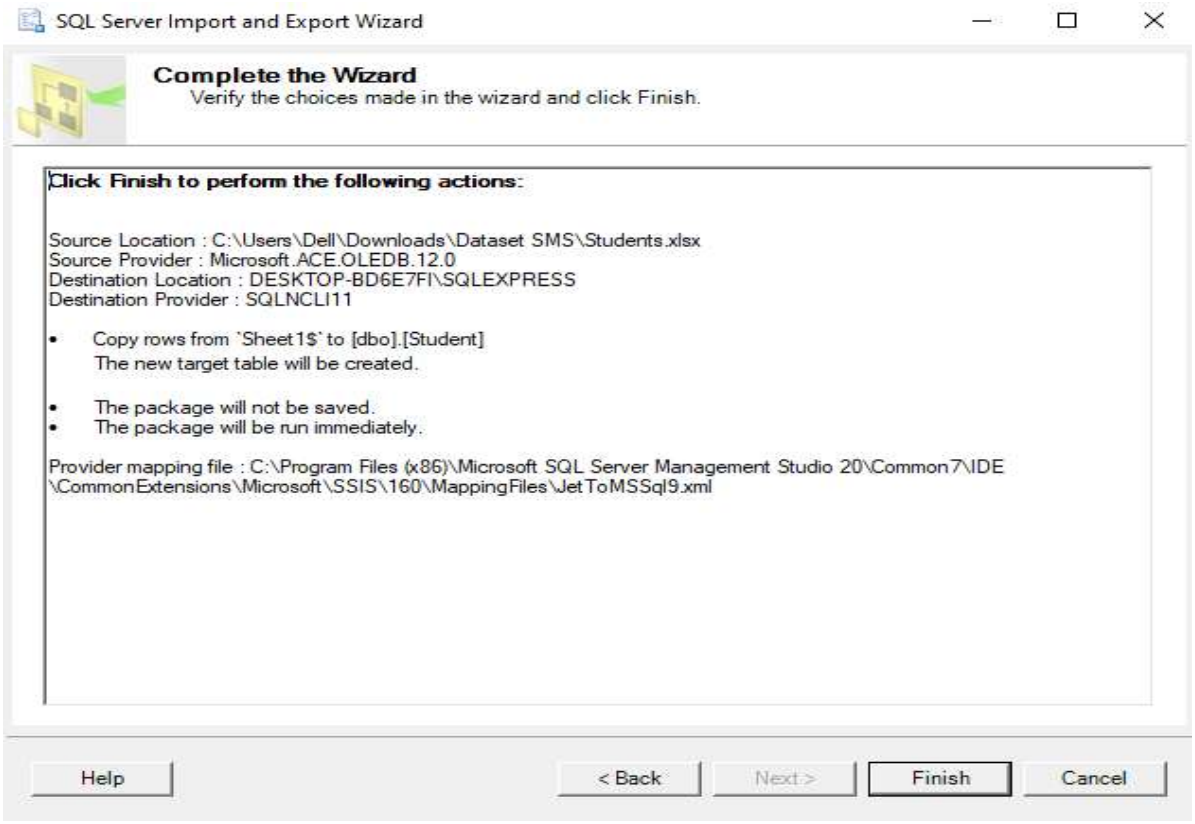


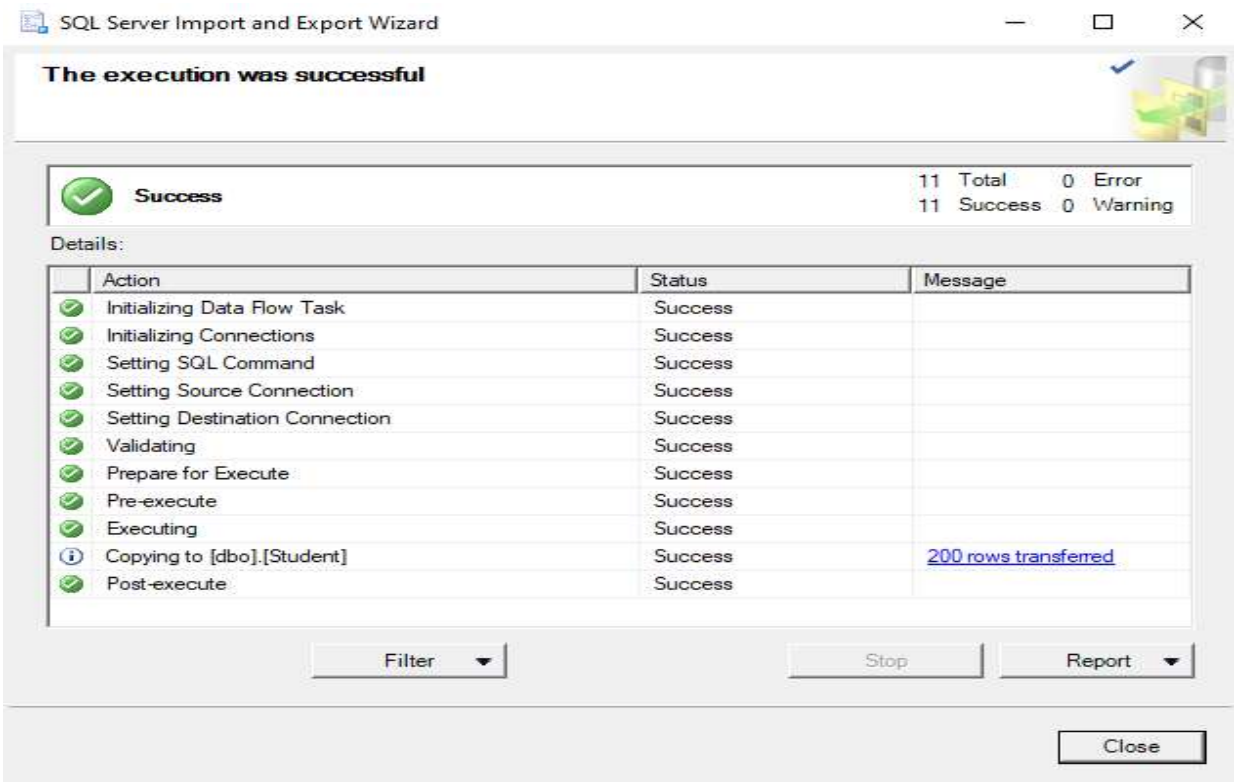

SQLQuery1.sql - DE...-BD6E7FI\Dell (52))*
use [SMSS];
SQL Server Import and Export Wizard
Choose a Data Source
Select the source from which to copy data.
Data source:
Microsoft Excel
Excel connection settings
Excel file path:
C:\Users\Dell\Downloads\Dataset SMS\Students.xlsx
Browse...
Excel version:
Microsoft Excel 2007-2010
First row has column names
Help
< Back
Next >
Finish >>|
Cancel


SQLQuery1.sql - DE...-BD6E7FI\Dell (52))*
use [SMSS];
SQL Server Import and Export Wizard
Choose a Destination
Specify where to copy data to.
Destination:
SQL Server Native Client 11.0
Server name:
DESKTOP-BD6E7FI\SQLEXPRESS
Authentication
Use Windows Authentication
Use SQL Server Authentication
User name:
Password:
Database:
SMSS
Refresh
New...
Help
< Back
Next >
Finish >>|
Cancel


SQL Server Import and Export Wizard
Specify Table Copy or Query
Specify whether to copy one or more tables and views or to copy the results of a query from the data source.
Copy data from one or more tables or views
Use this option to copy all the data from the existing tables or views in the source database.
Write a query to specify the data to transfer
Use this option to write an SQL query to manipulate or to restrict the source data for the copy operation.
Help
< Back
Next >
Finish >>|
Cancel

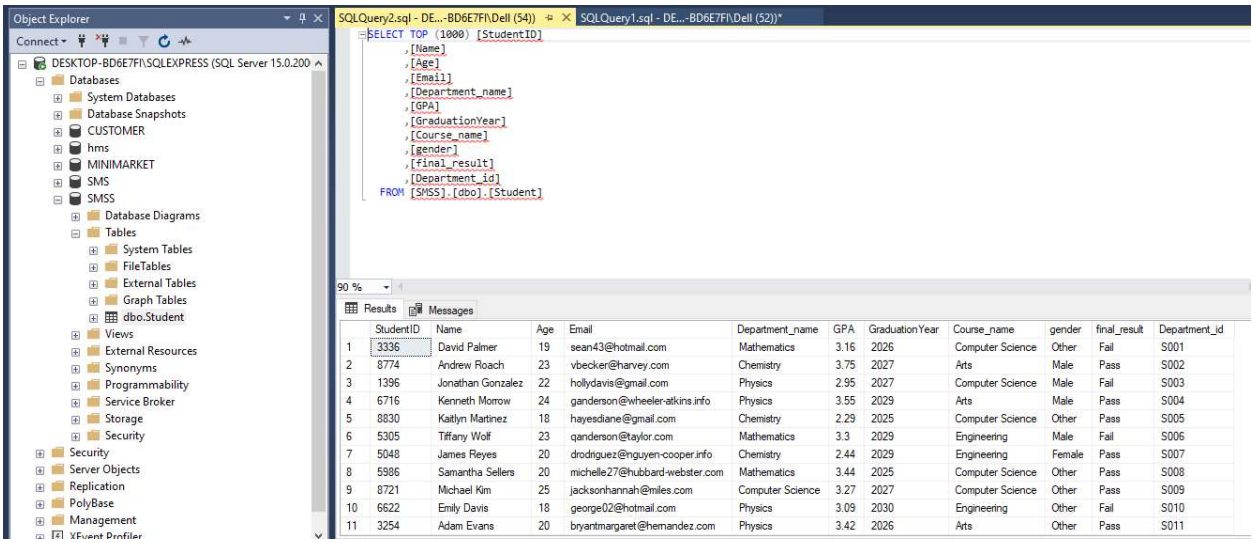

SQL Server Import and Export Wizard
Select Source Tables and Views
Choose one or more tables and views to copy.
Tables and views:
Source: C:\Users\Dell\Downloads\Dataset SMS\Stu...
Destination: DESKTOP-BD6E7FI\SQLEXPRESS
`Sheet1$`
[dbo].[Student]
Edit Mappings...
Preview...
Help
< Back
Next >
Finish >>|
Cancel





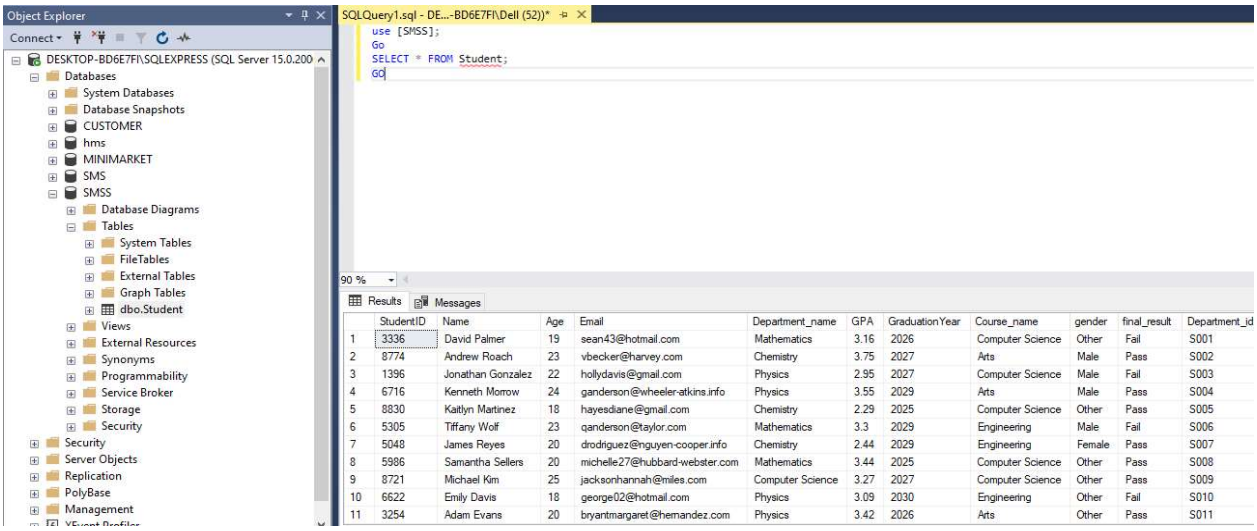
The method of importing all the other tables from excel to SQL is same as above.

Table view



Functional Dependencies

For Student Table:



Checking one by one the repeating values in a column with the query as follows:

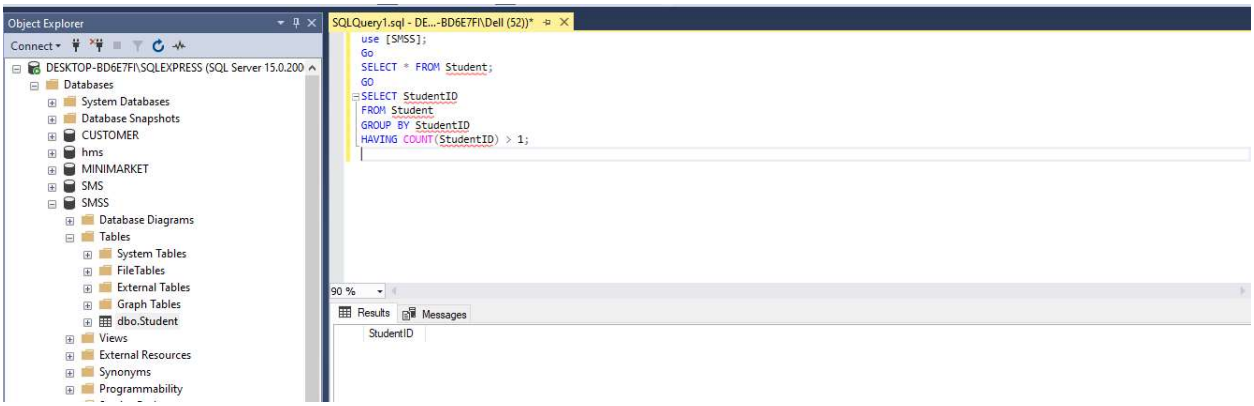
SELECT Column_name

FROM Table_name

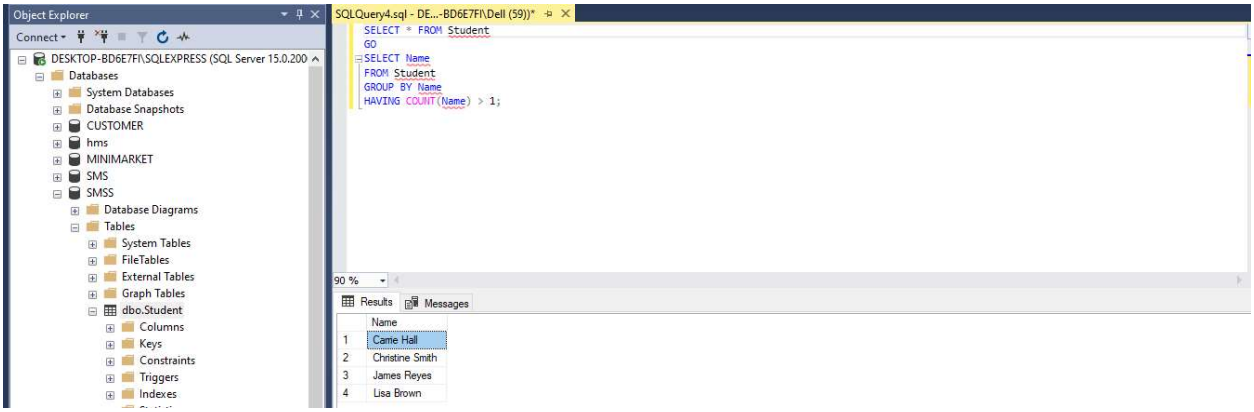
GROUP BY Column_name

HAVING COUNT(Column_name) > 1;

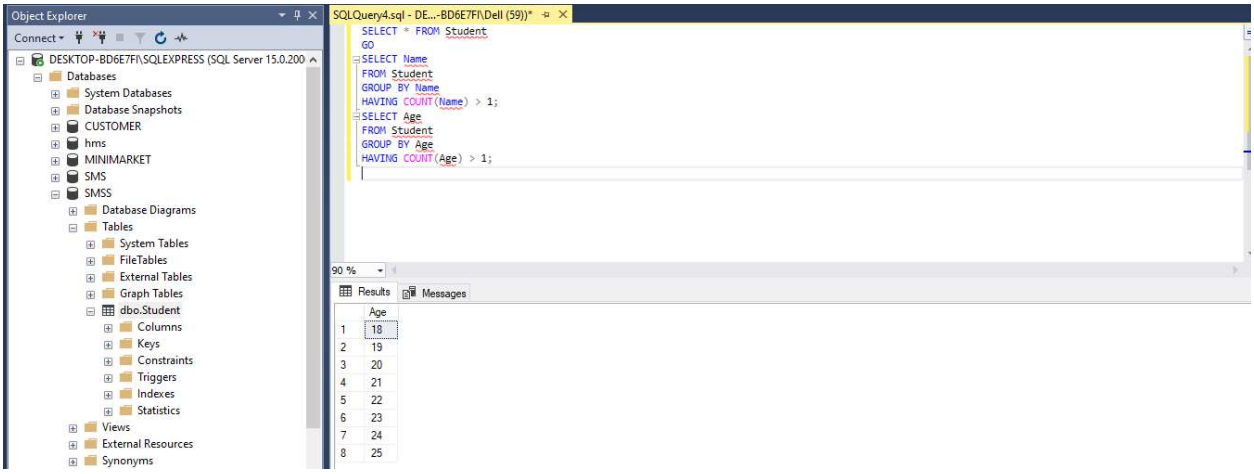
[Attribute: StudentID]



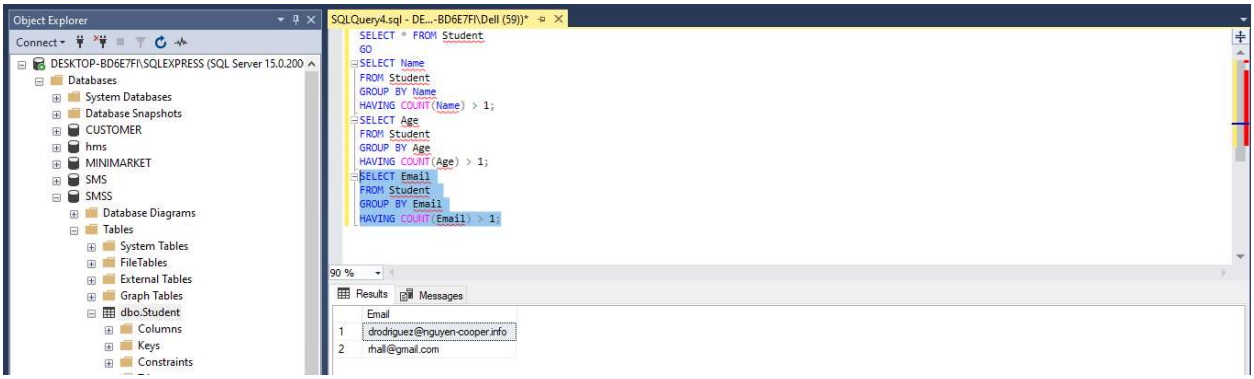
[Attribute: Name]



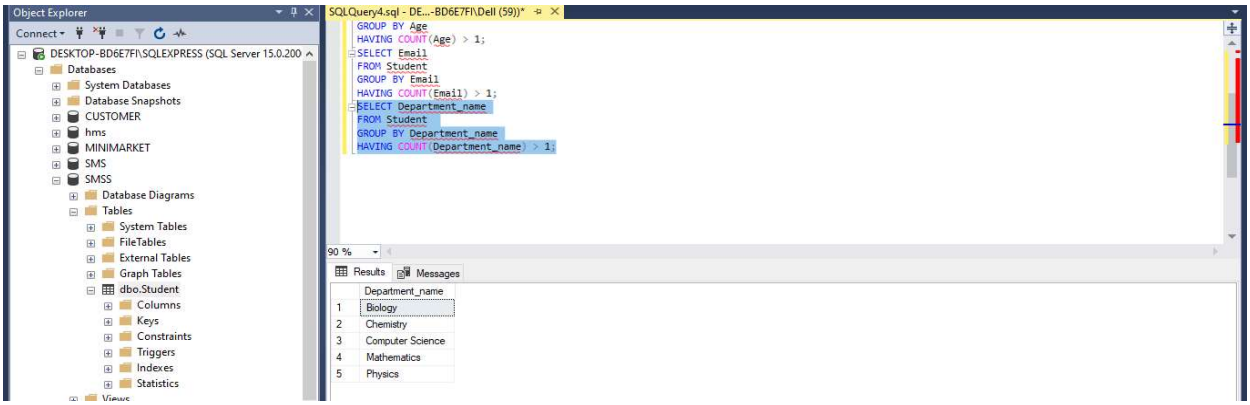
[Attribute: Age]



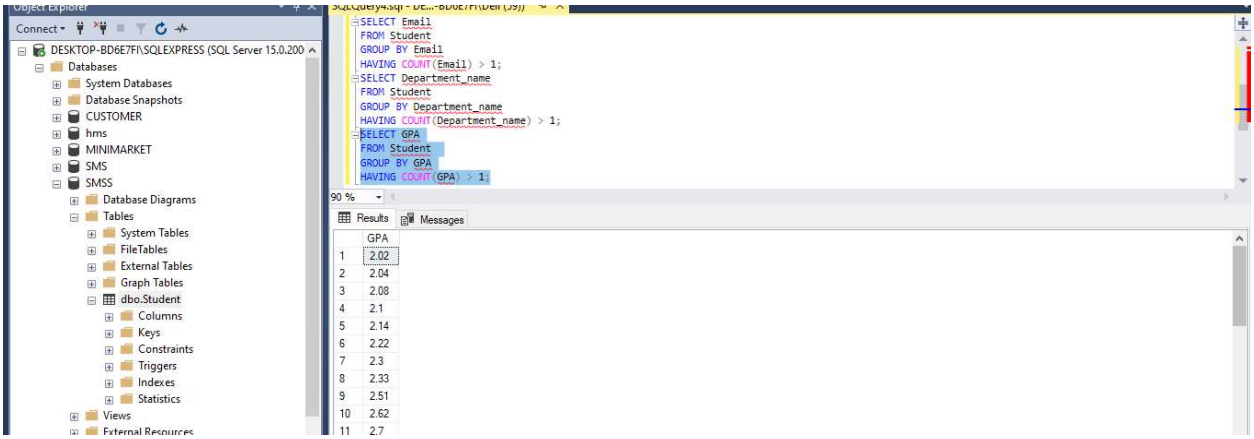
[Attribute: Email]



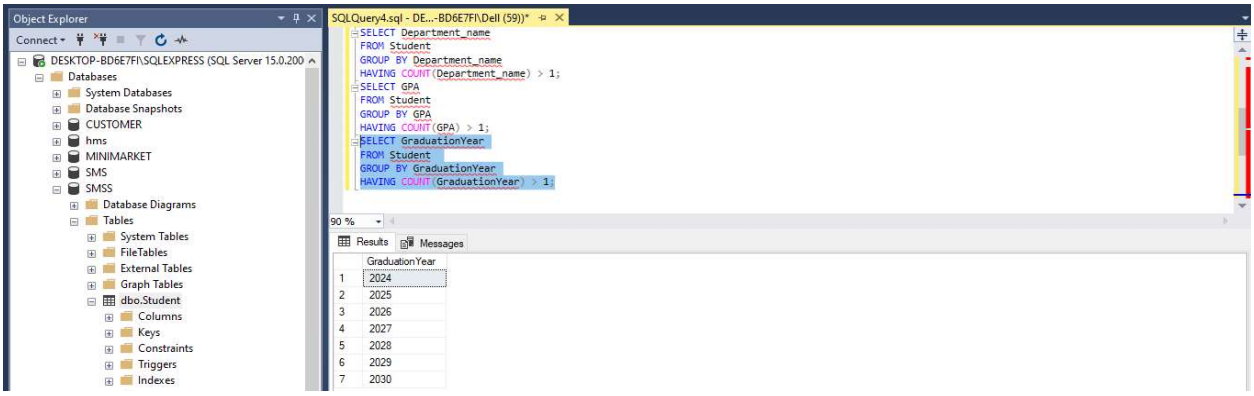
[Attribute: Department_name]



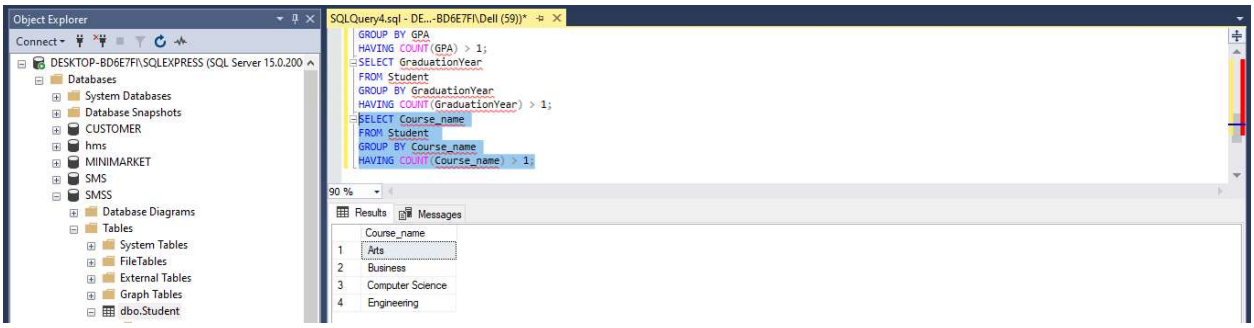
[Attribute: GPA]



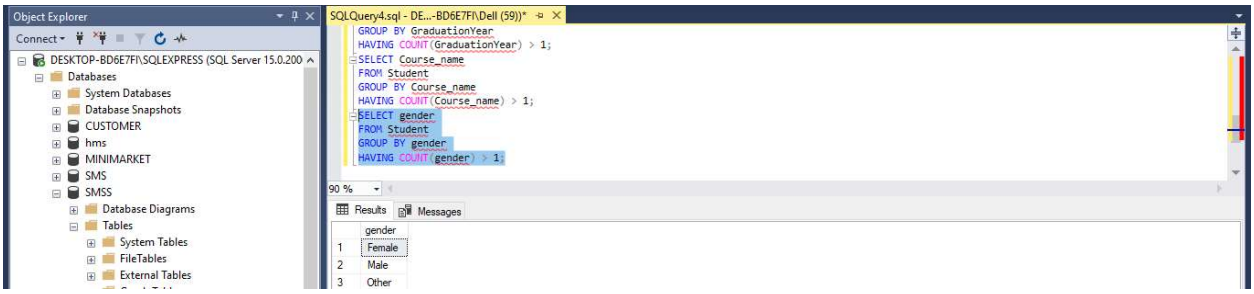
[Attribute: GraduationYear]



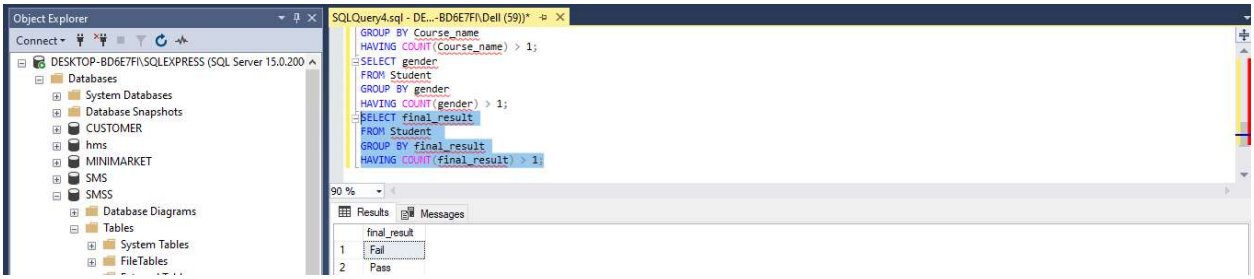
[Attribute: Course_name]



[Attribute: gender]



[Attribute: final_result]

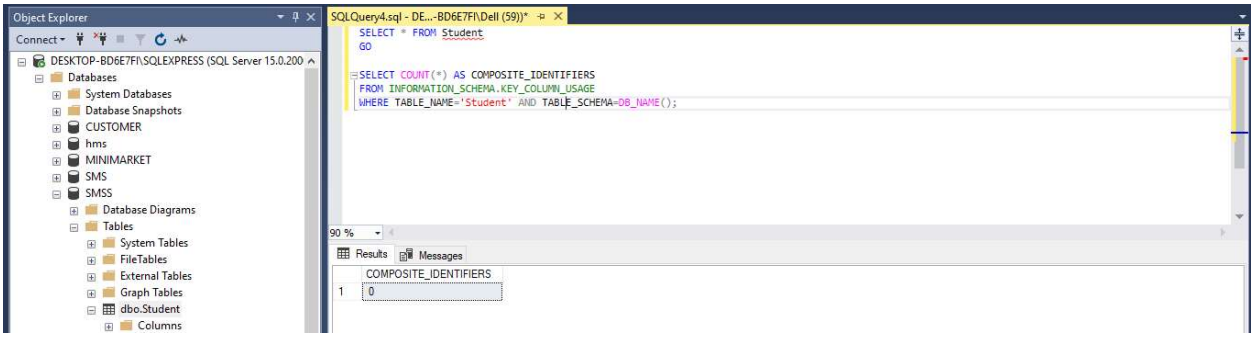


we get to know that **StudentID** does not have repetition and it can determine all other attributes. The other columns have repeating values so we cannot consider them as unique identifers.

So, **StudentID** will be determinant and all other attributes will be dependent. Now we have to check if there are any composite keys that can uniquely identify records by executing the following query:

Composite Key Identifier

```
SELECT COUNT(*) AS COMPOSITE_IDENTIFIERS
FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE
WHERE TABLE_NAME='Student' AND TABLE_SCHEMA=DB_NAME();
```



This query tells us if there are any composite keys but as the result of the query is zero we get to know that there are no composite keys which can uniquely identify records. so, Reg-no is the uniquely identifying key(PK).

Functional Dependency Analysis:

StudentID → Name

StudentID → Age

StudentID → Email

StudentID → Department_name

StudentID → GPA

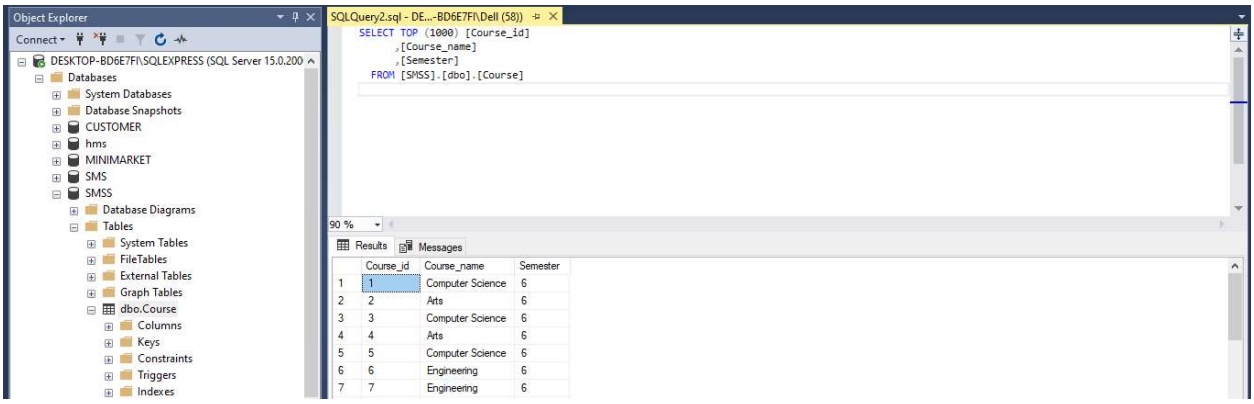
StudentID → GraduationYear

StudentID → Course_name

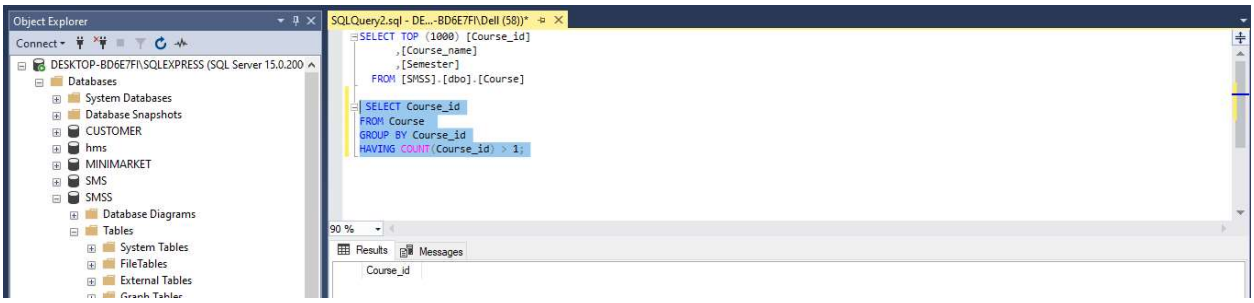
StudentID → gender

StudentID → final_result

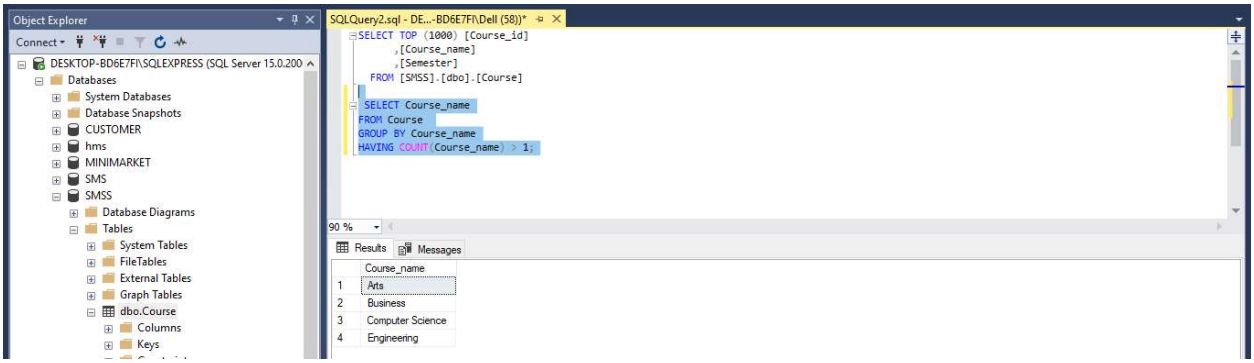
From Course table:



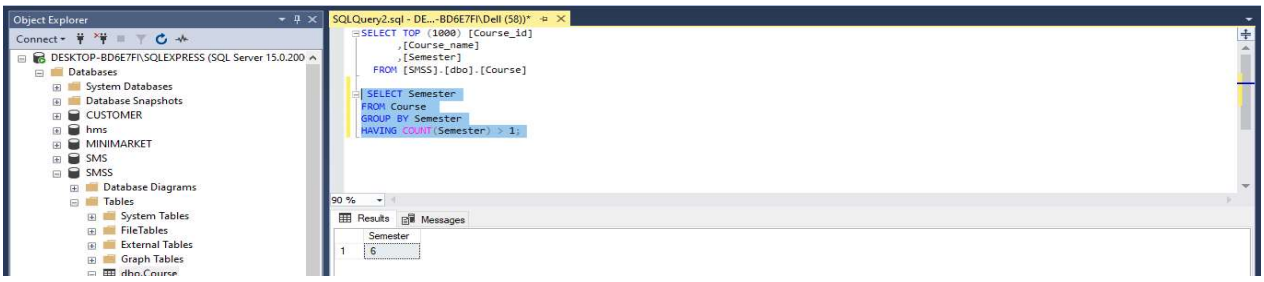
[Attribute: Course_id]



[Attribute: Course_name]



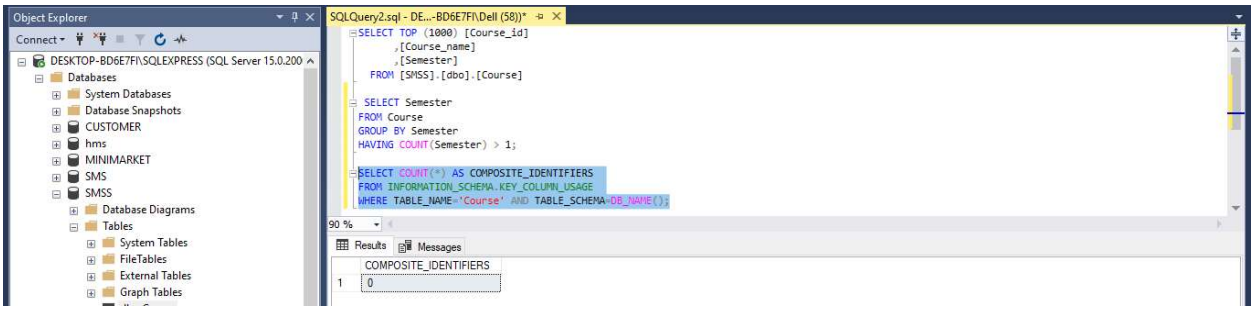
[Semester]



Course_name and Semester are constantly repeating as we can see in the table so they cannot be considered as unique identifier alone. The unique identifier in the course table are [course_id] The other columns have repeating values so we cannot consider them as unique identifers.

Composite Key Identifier

SELECT COUNT(*) AS COMPOSITE_IDENTIFIERS FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE WHERE TABLE_NAME='Customer' AND TABLE_SCHEMA=DB_NAME();



So functional dependency that exist in Course table is:

PARTIAL FUNCTIONAL DEPENDENCY:

[course_id] → [course_name]

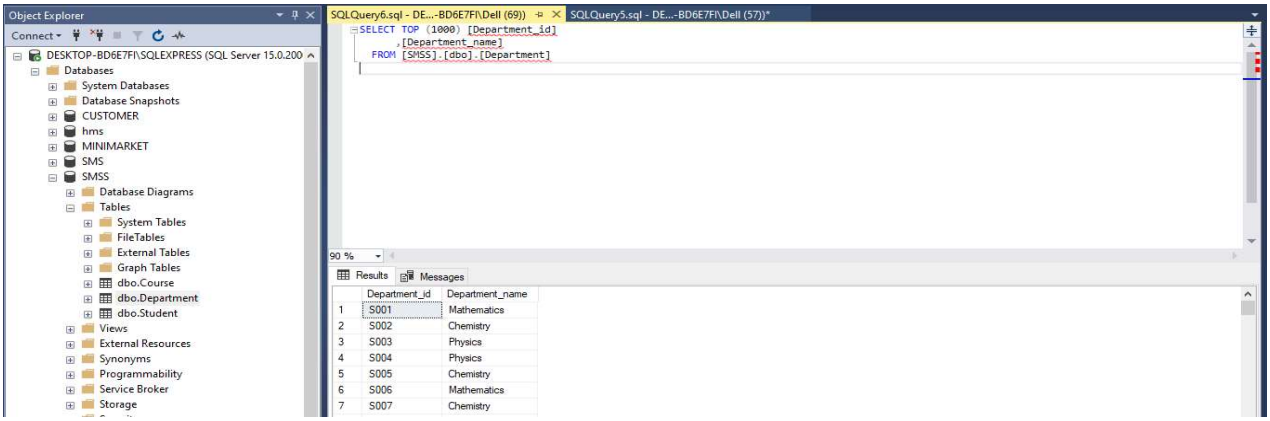
[course_id] → [Semester]

This means that in Course table [course_id] determines {[course_name],[Semester]}

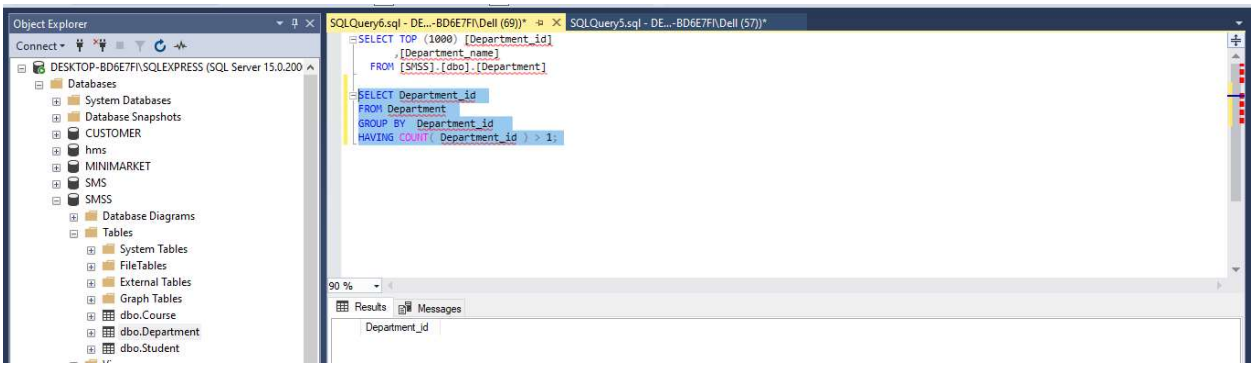
Determinants: [course_id]

Dependents: [course_name], [Semester]

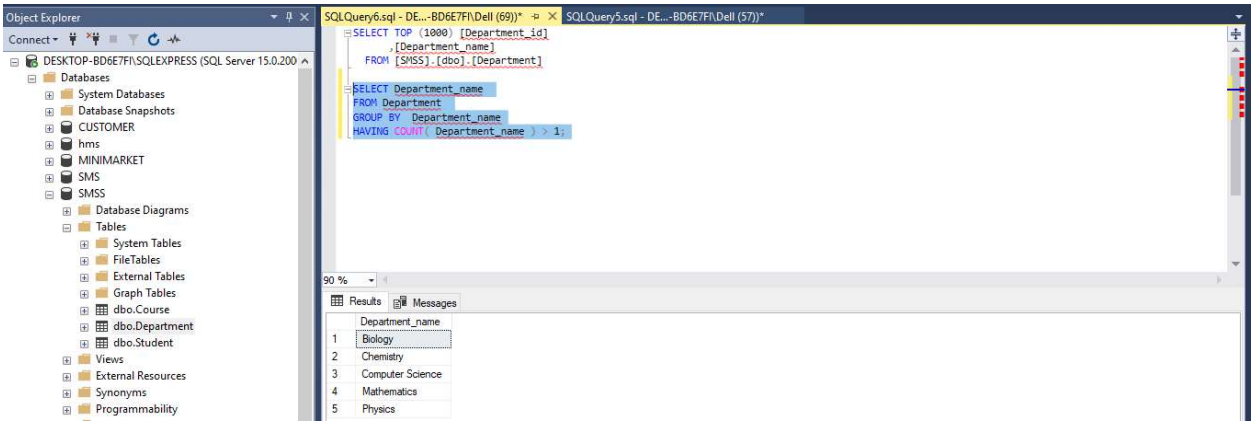
For Department table:



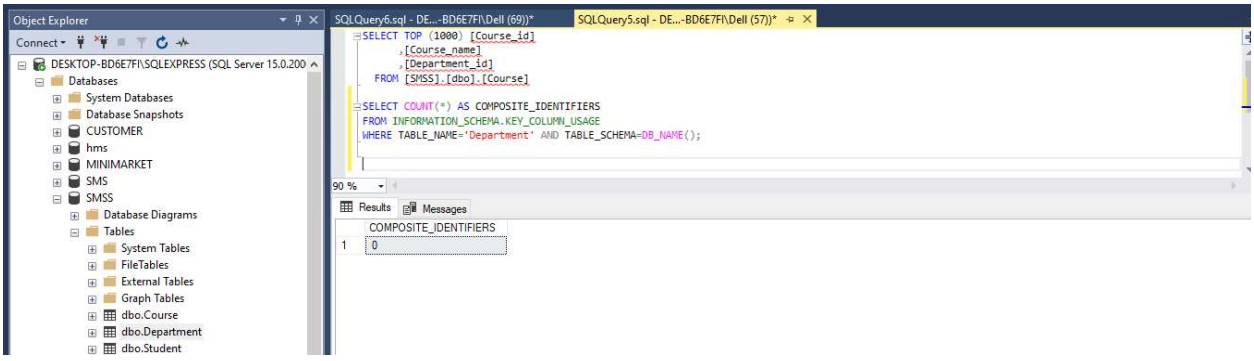
[Attribute: Department_id]



[Attribute: Department_name]



Composite key identifier



Given that dep-id and dep-name both uniquely identify a department, we can establish the following functional dependencies:

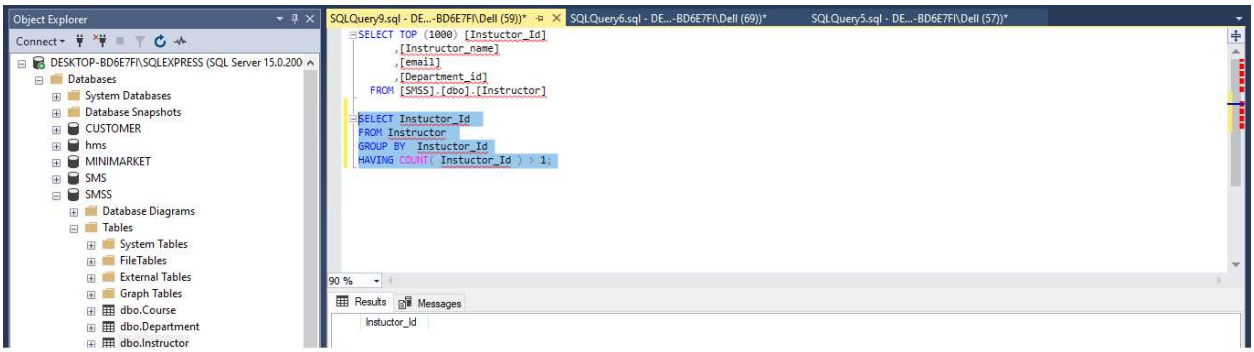
Department_id uniquely identifies the department's name : dep-id → dep-name

Determinant: [dep-id], [dep-name]

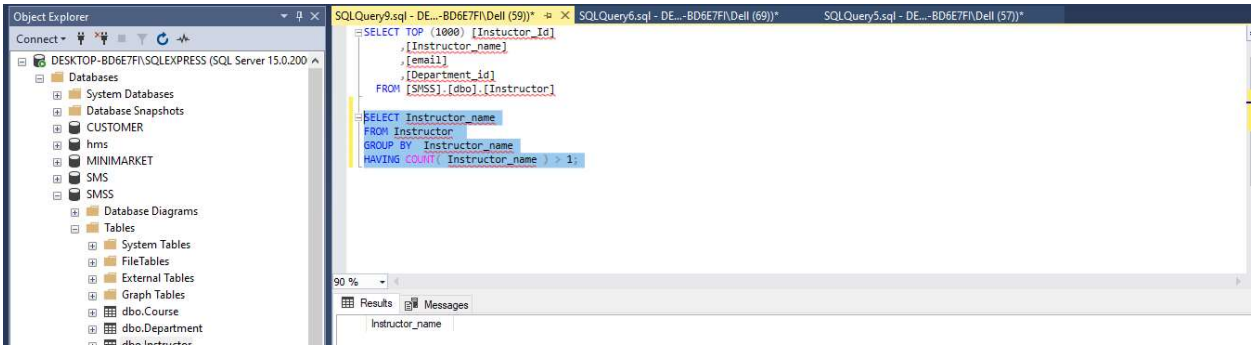
Dependents: [dep-id]

For Instructor Table:

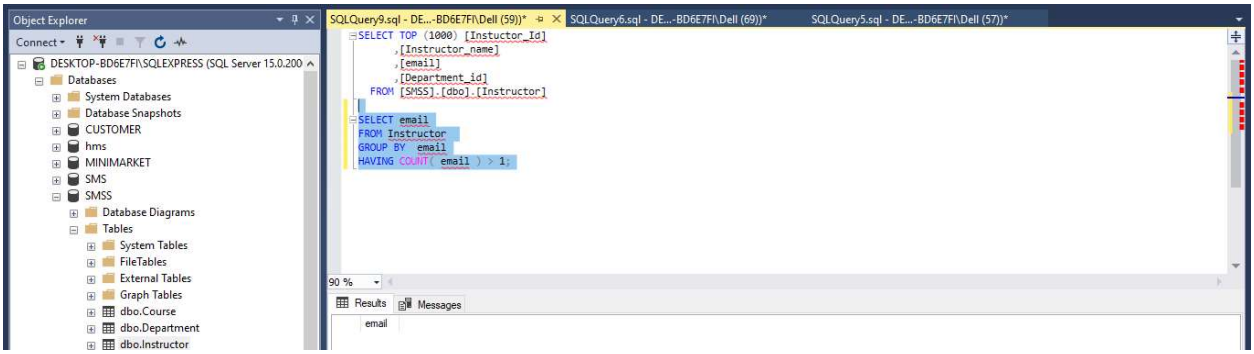
[Attributes: Instructor_id]



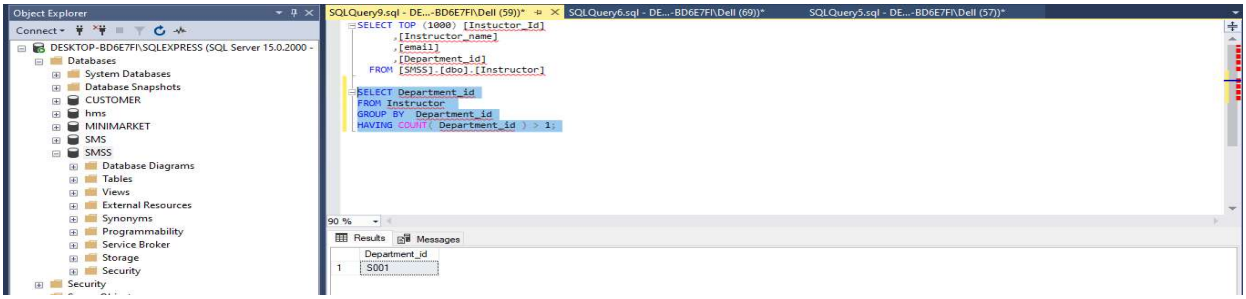
[Attributes: Instructor_name]



[Attribute: Email]



[Attribute: department_id]



Since we can see that dep-id is not unique it can become foreign key in this table.

department-id are constantly repeating as we can see in the table so they cannot be considered as unique identifier alone. The unique identifier in the Instructor table are [Instructor_id] and [Instructor_Email],[Instructor_Name].

Composite keys Identifier

So functional dependency that exist in Instructor table is:

[Instructor_Id], [Instructor_Name], [Instructor_Email] → [dep-id]

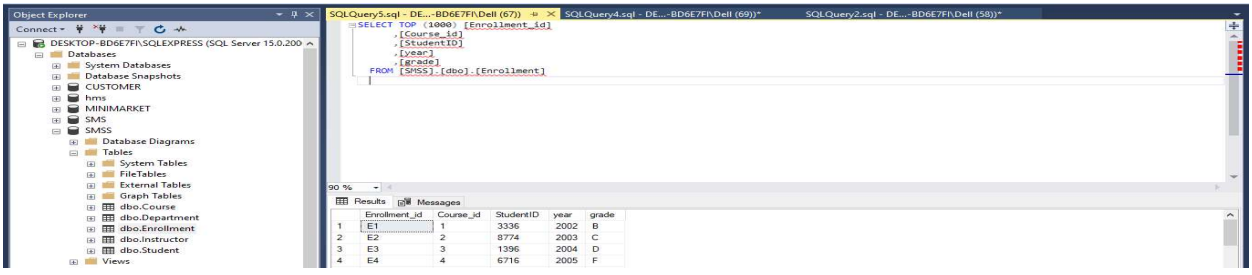
This suggests that the combination of Instructor_Id, Instructor_Name, and Instructor_Email

determines dep-id.

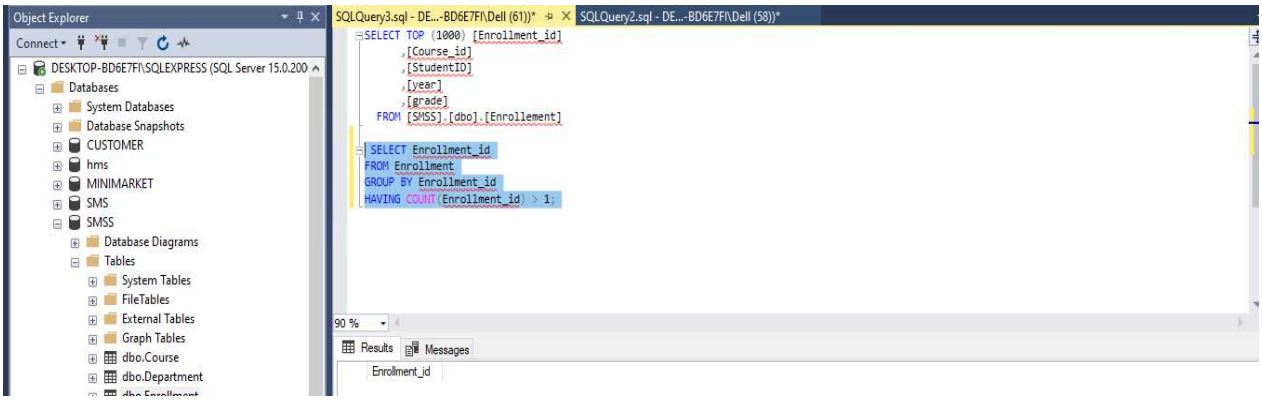
Determinants: [Instructor_Id], [Instructor_Name] ,[Instructor_Email]

Dependents: [dep-id]

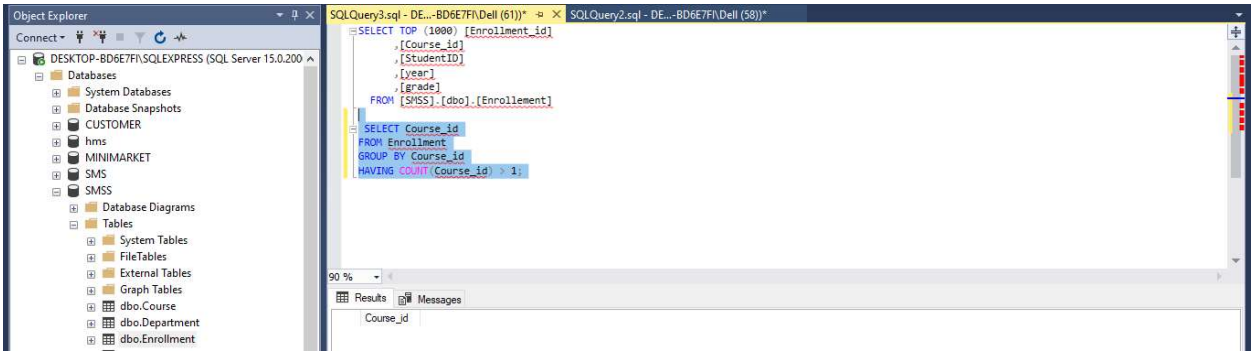
For Enrollment Table:



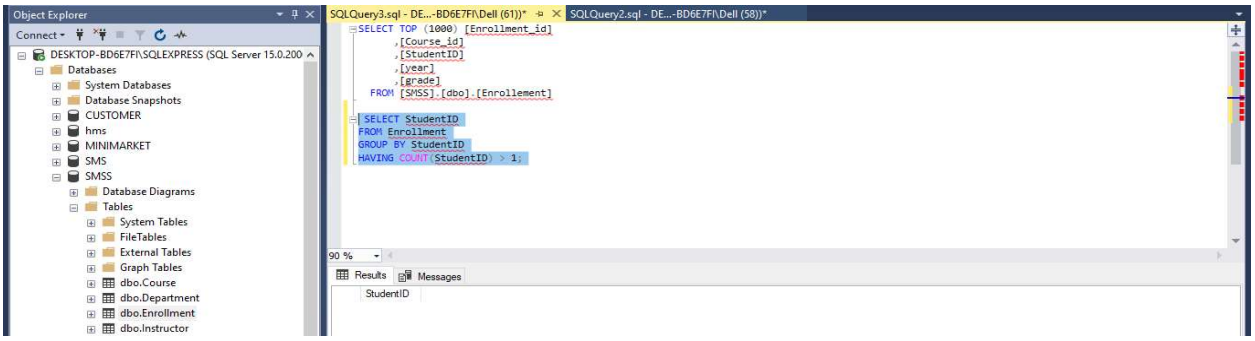
[Attribute: Enrollment_id]



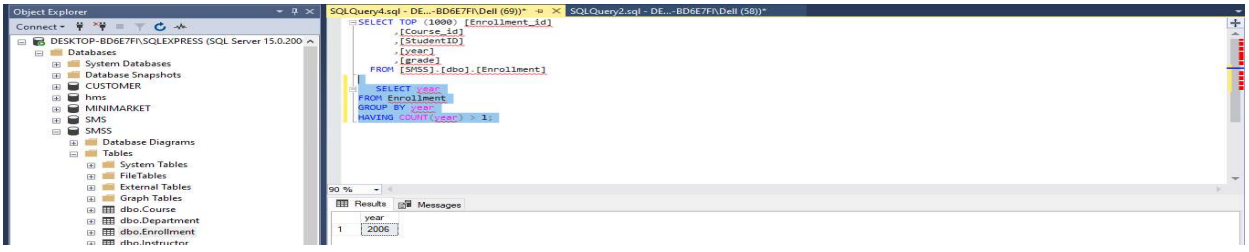
[Course_id]



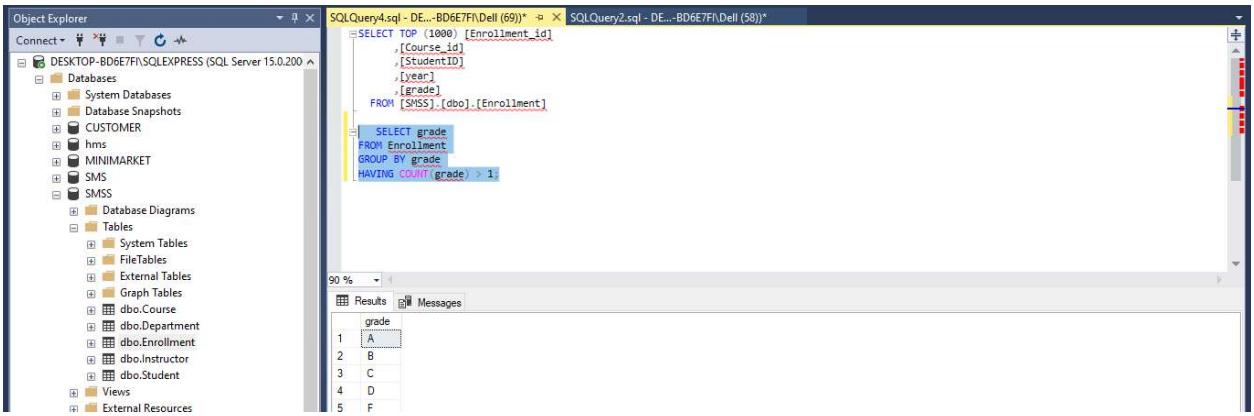
[student_id]



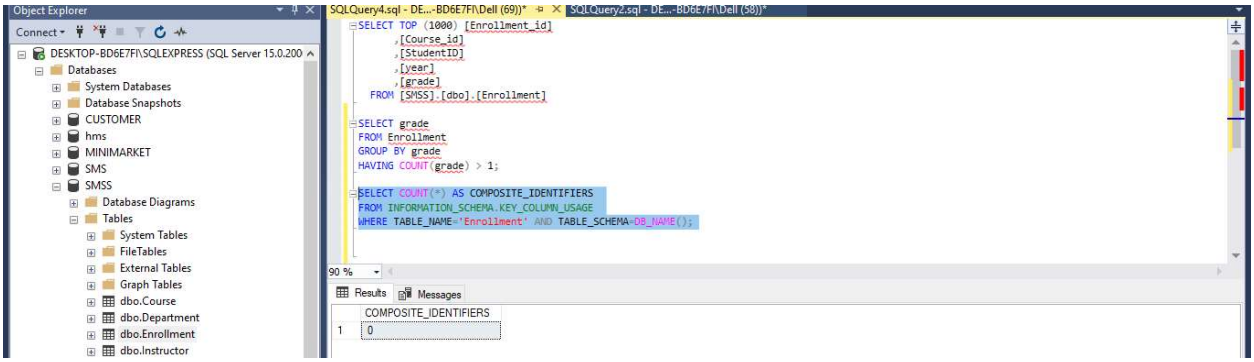
[year]



[grade]



Composite key identifier



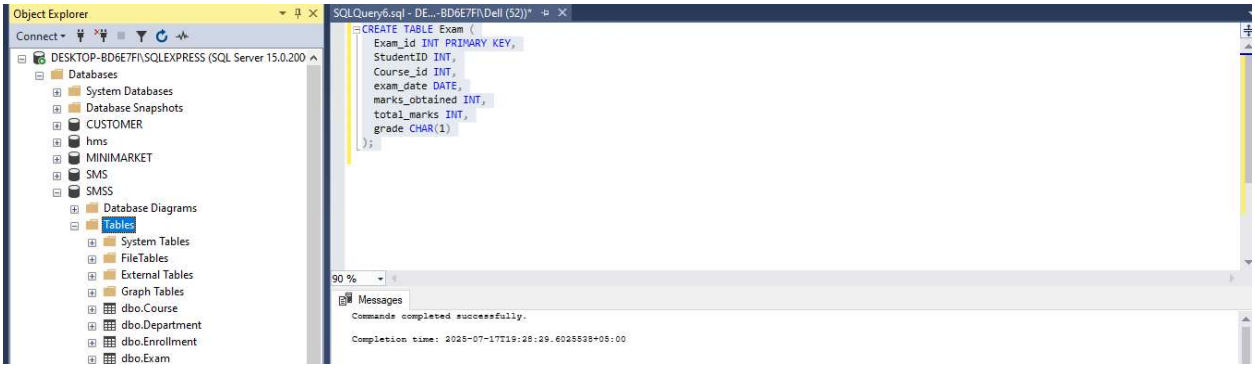
If we have defined only Enrollment_id as PK, no composite PK will appear. If we define (StudentID, Course_id) as composite PK, both will appear under the PK constraint.

Using Query

Step-wise implementation for Exam table

1. **Create Exam Table**

```
CREATE TABLE Exam (
  Exam_id INT PRIMARY KEY,
  StudentID INT,
  Course_id INT,
  exam_date DATE,
  marks_obtained INT,
  total_marks INT,
  grade CHAR(1)
);
```

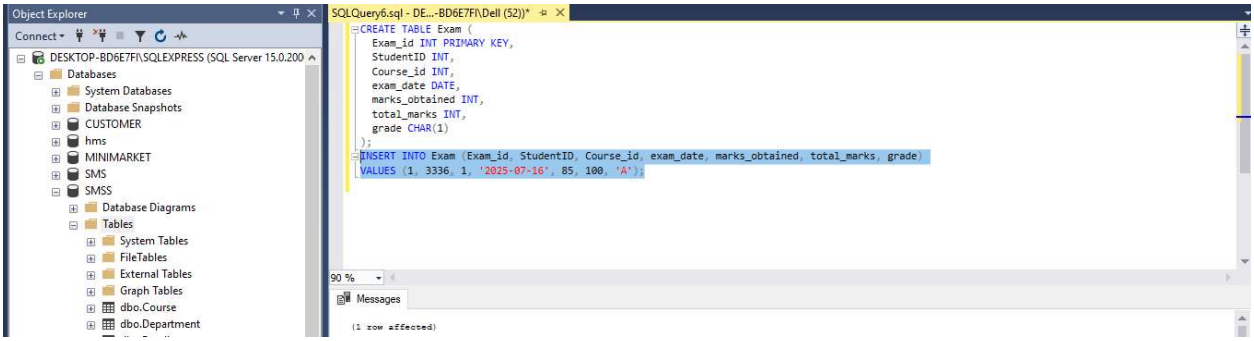


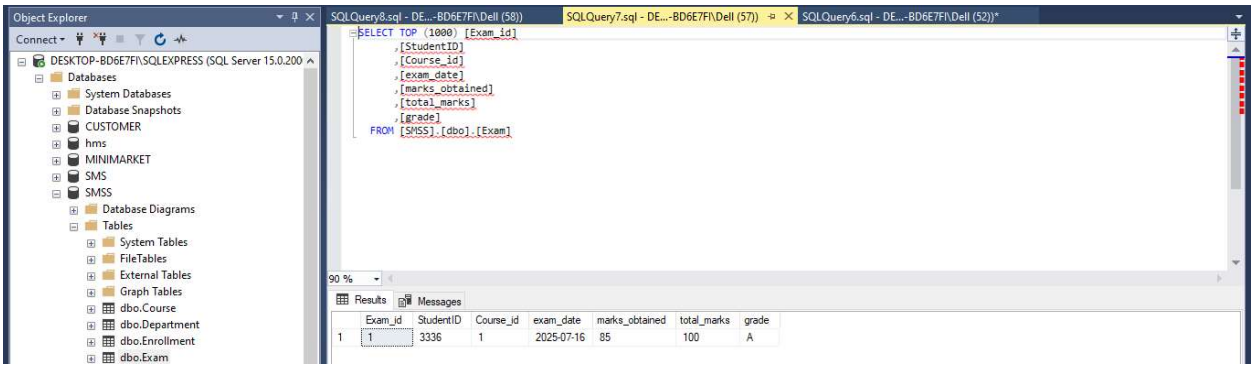
2. Insert Data into Exam Table

Insert single record:

INSERT INTO Exam (Exam_id, StudentID, Course_id, exam_date, marks_obtained, total_marks, grade)

VALUES (1, 3336, 1, '2025-07-16', 85, 100, 'A');





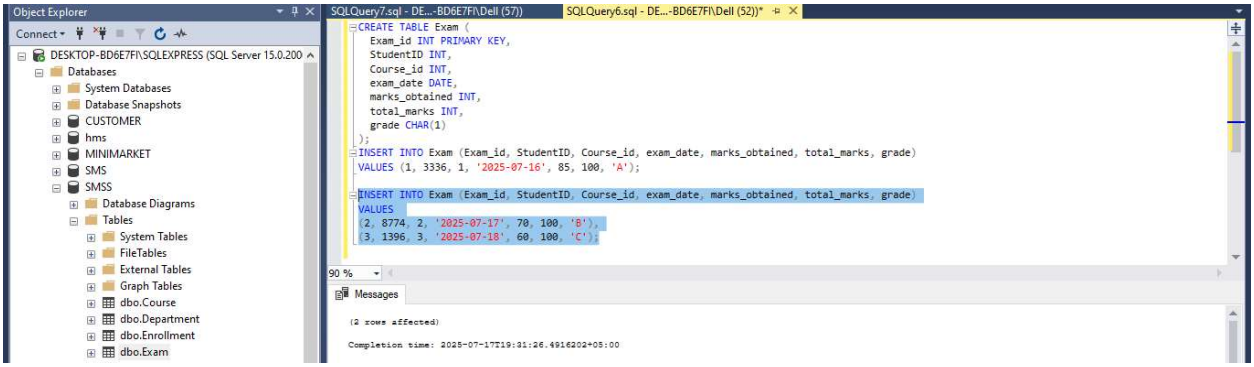
Insert multiple records:

INSERT INTO Exam (Exam_id, StudentID, Course_id, exam_date, marks_obtained, total_marks, grade)

VALUES

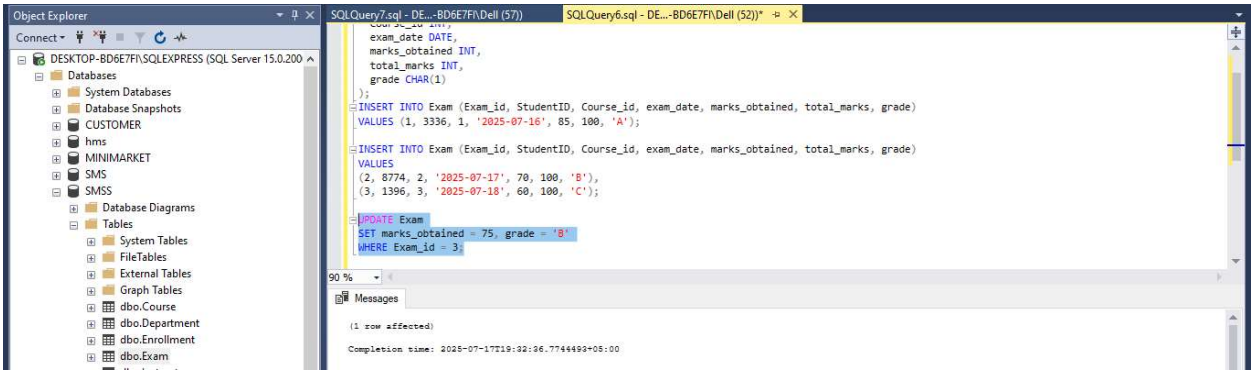
(2, 8774, 2, '2025-07-17', 70, 100, 'B'),

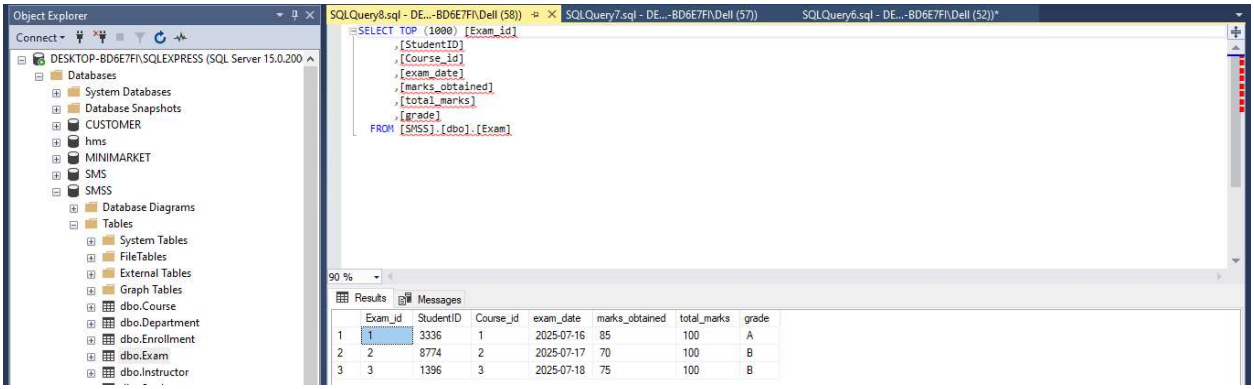
(3, 1396, 3, '2025-07-18', 60, 100, 'C');



3. Update Data in Exam Table

Update marks and grade for a specific exam:



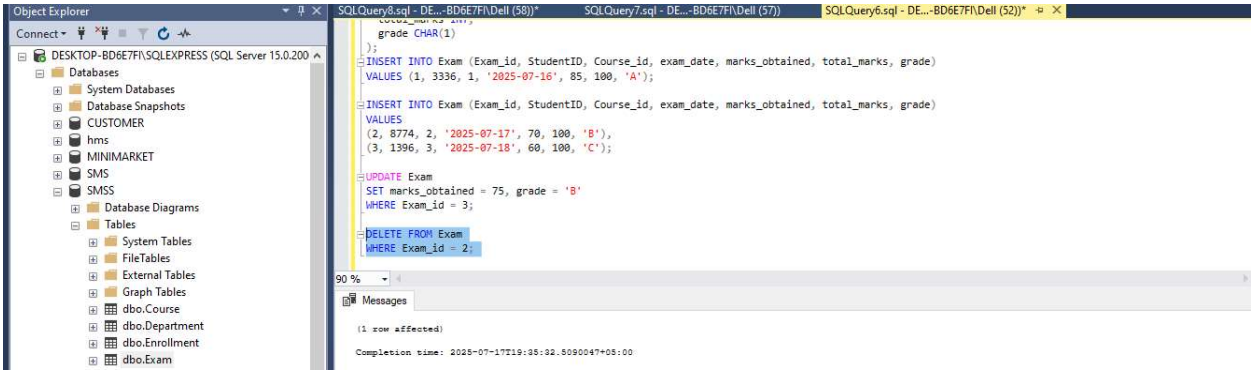


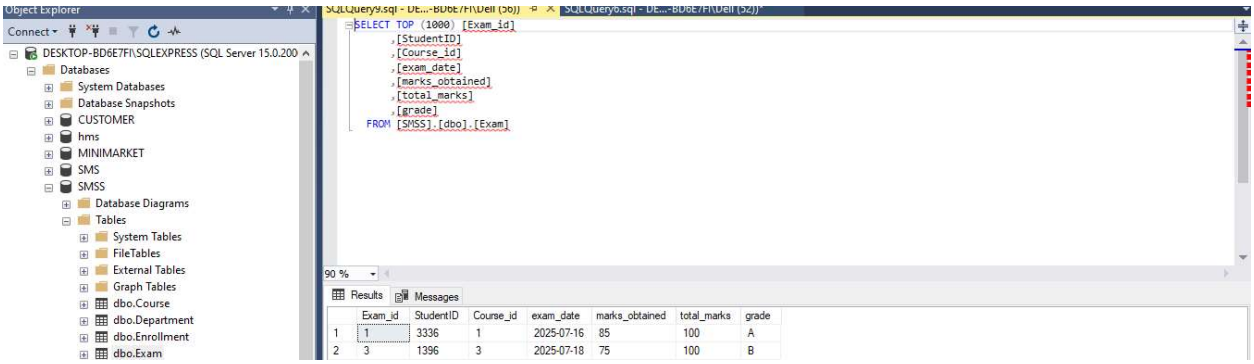
4. Delete Data from Exam Table

Delete a specific record:

DELETE FROM Exam

WHERE Exam_id = 2;

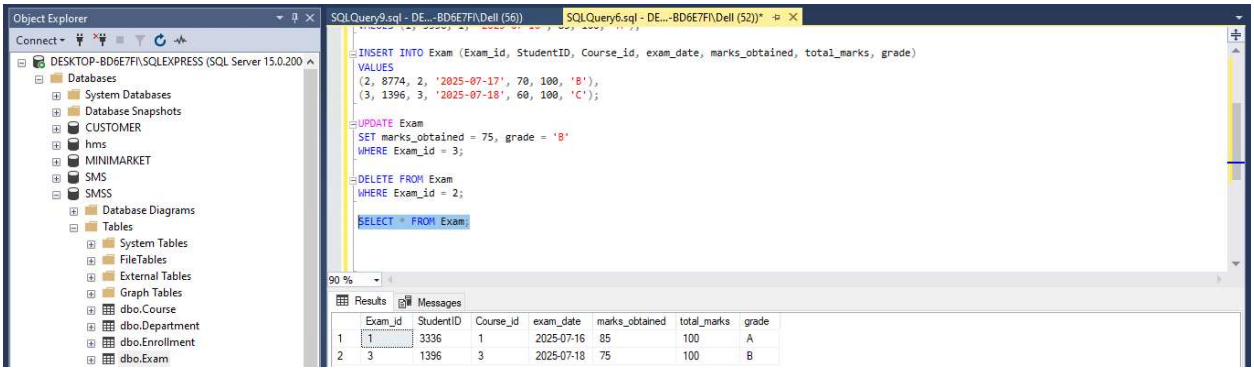




5. Basic Retrieval Queries

Retrieve all records:

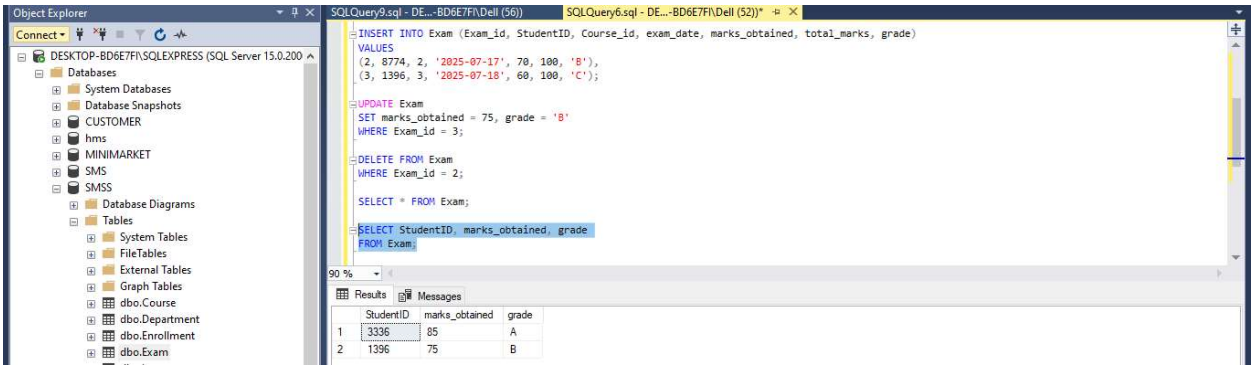
SELECT * FROM Exam;



Retrieve specific columns:

SELECT StudentID, marks_obtained, grade

FROM Exam**;**

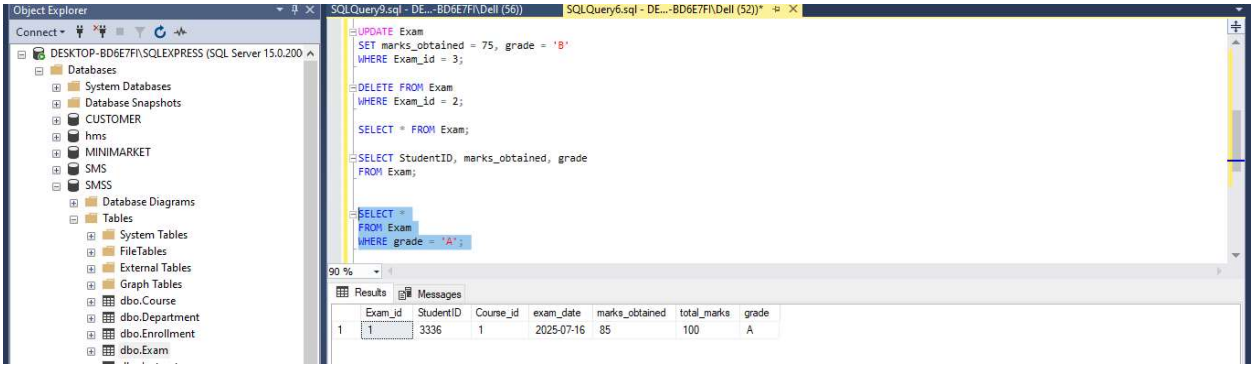


Retrieve data with condition (WHERE clause):

SELECT *

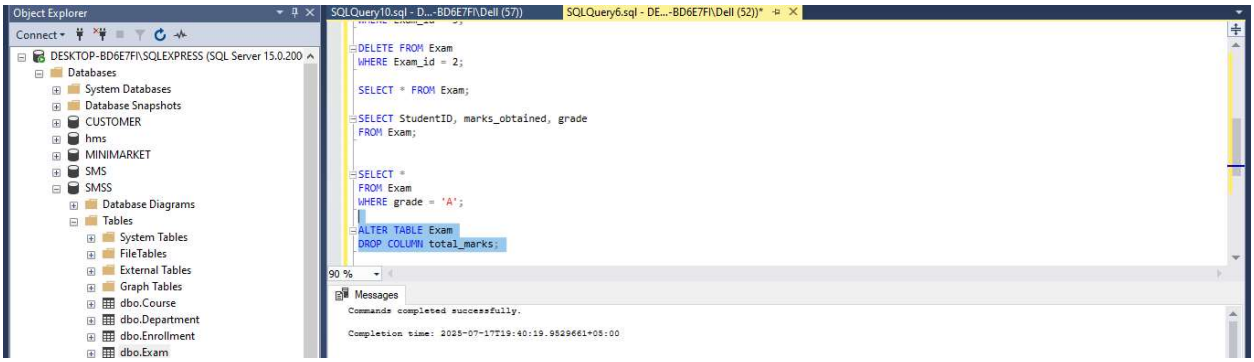
FROM Exam

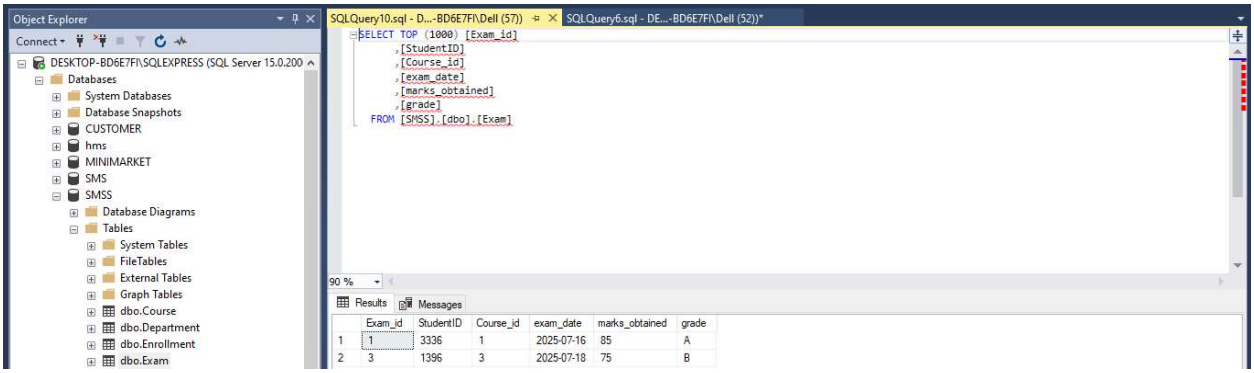
WHERE grade = 'A';



6. Delete Specific Column

Delete a column (e.g. total_marks):



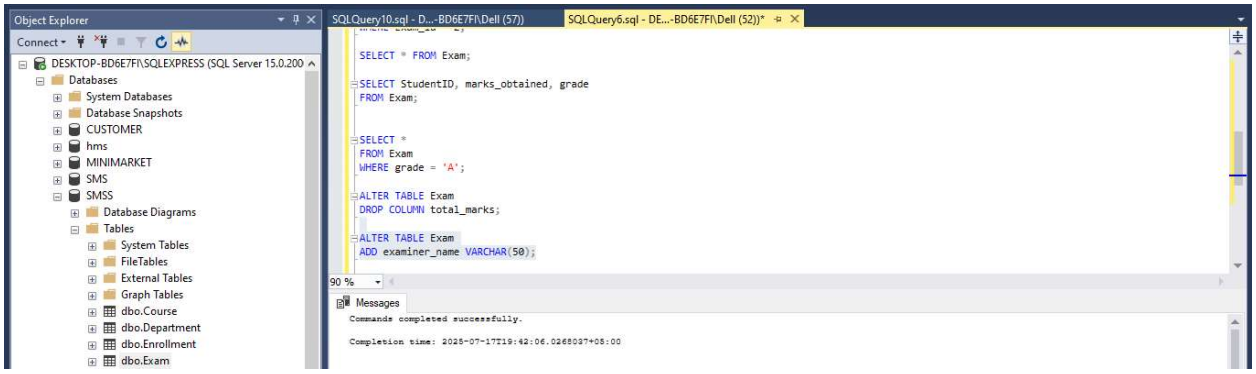


7. Alter Table: Add Columns

Add a single column (e.g. examiner_name):

ALTER TABLE Exam

ADD examiner_name VARCHAR(50);



Add multiple columns (e.g. exam_room and duration):

ALTER TABLE Exam

ADD exam_room VARCHAR(20),

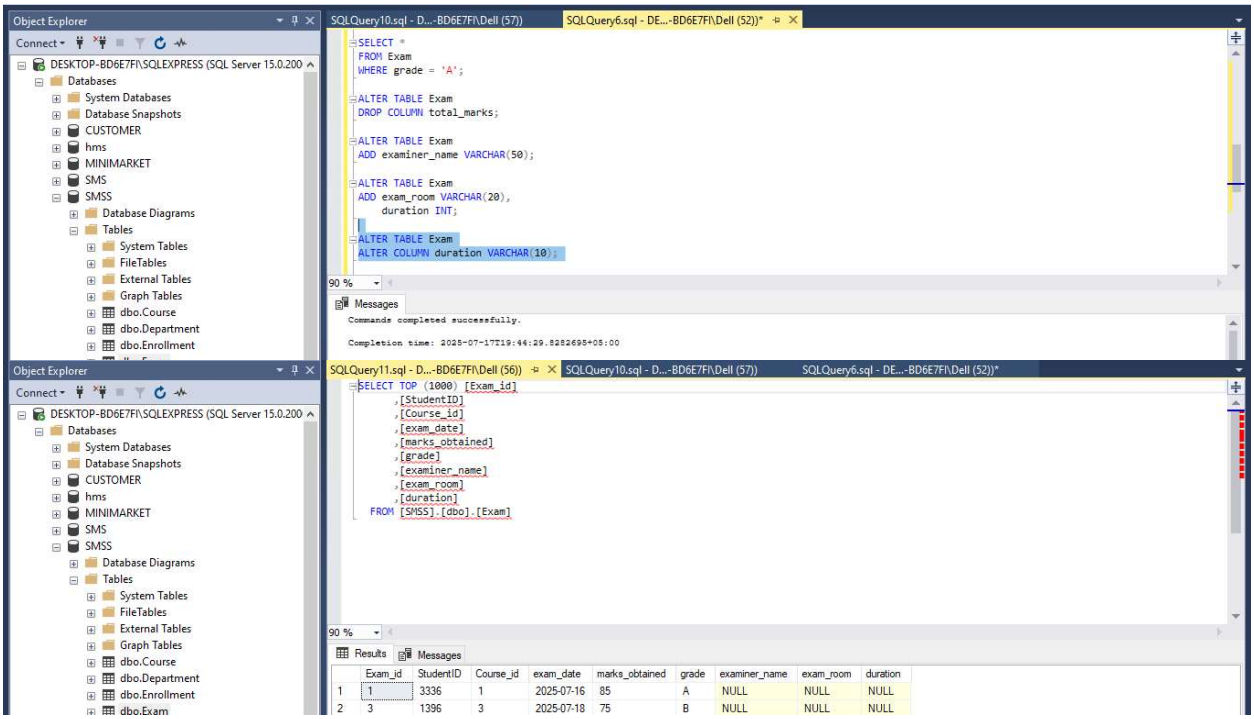
duration INT;

8. Modify Column

Change data type of duration column:

ALTER TABLE Exam

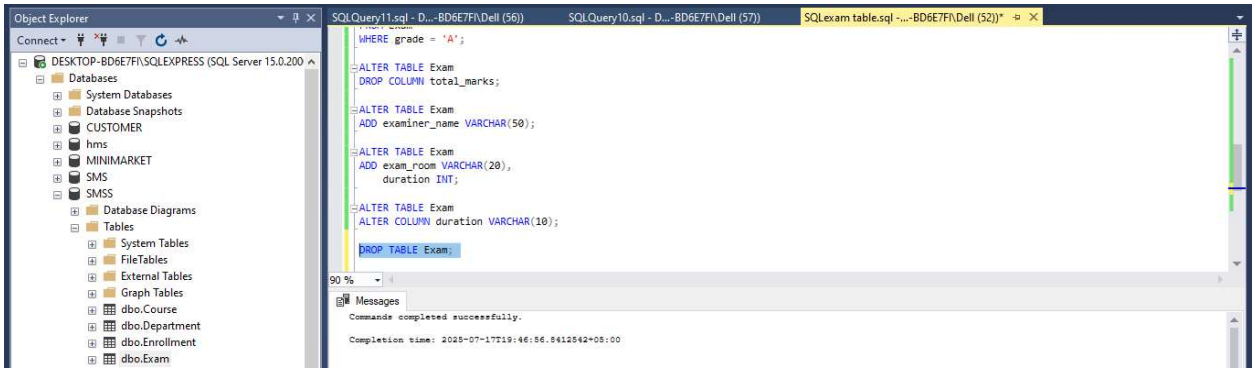
ALTER COLUMN duration VARCHAR(10);



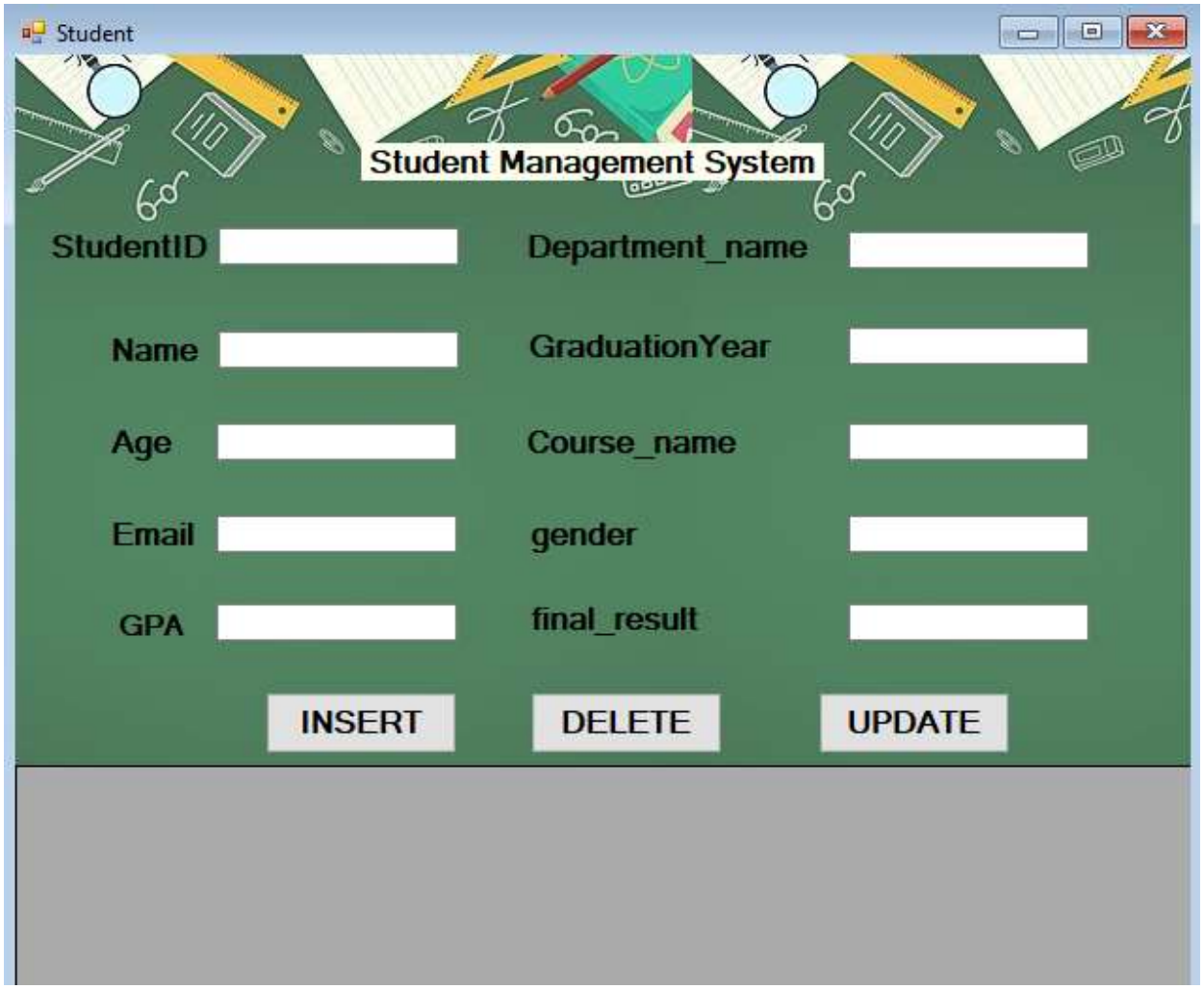
9. Delete Entire Exam Table

Drop table:

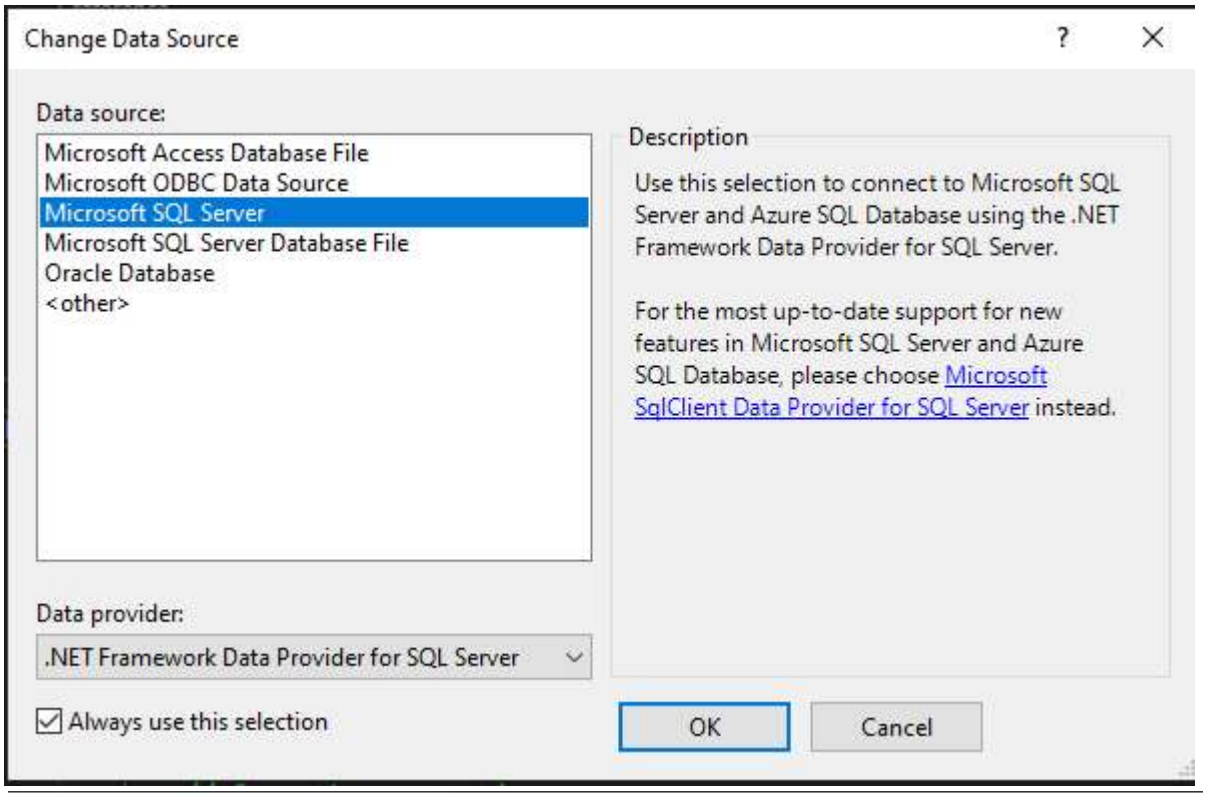
DROP TABLE Exam;

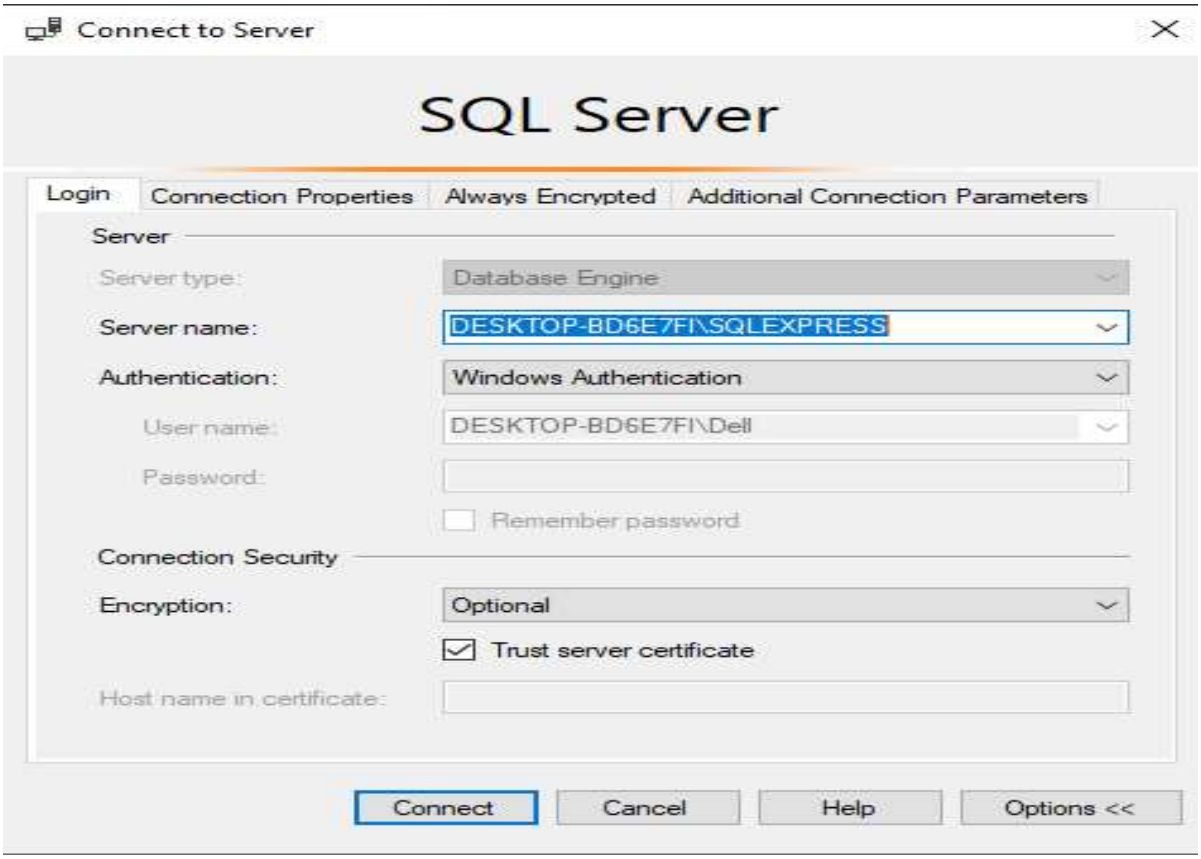


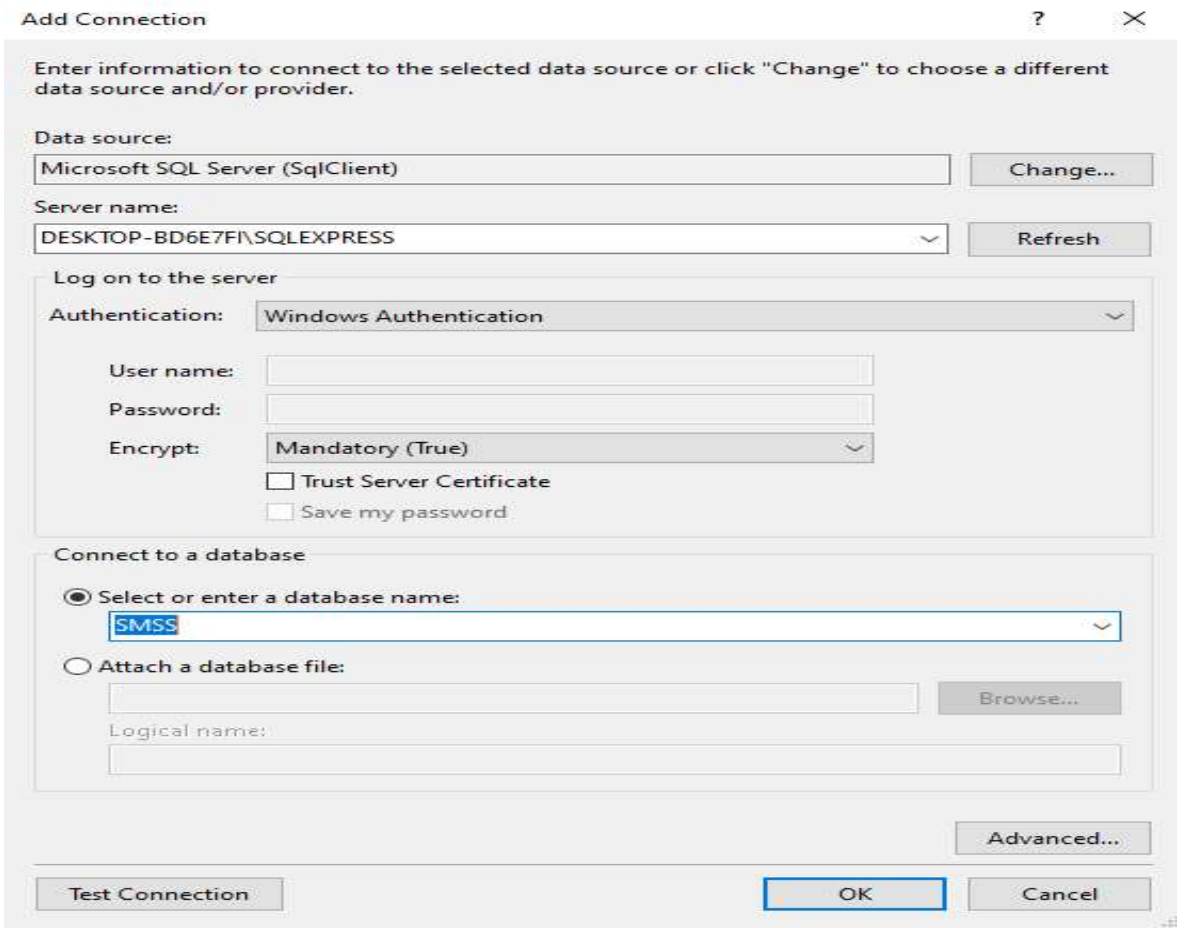
Frontend Form using Visual Studio Code

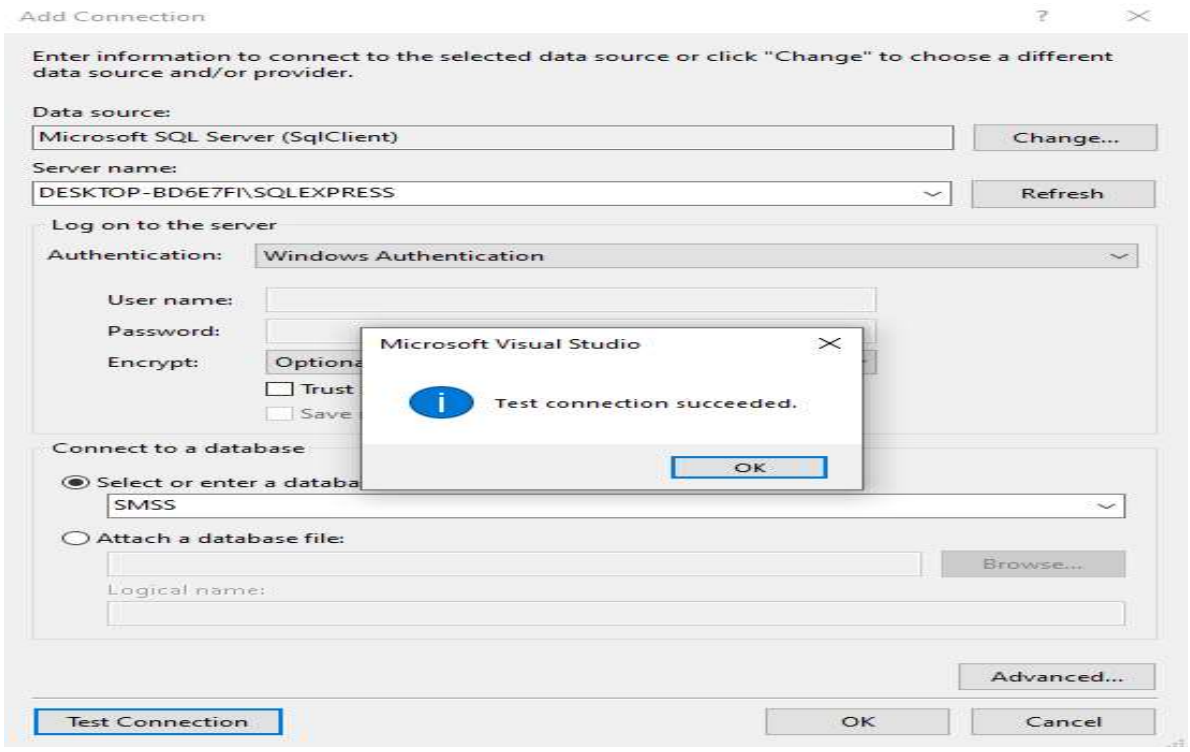


Connection With DB





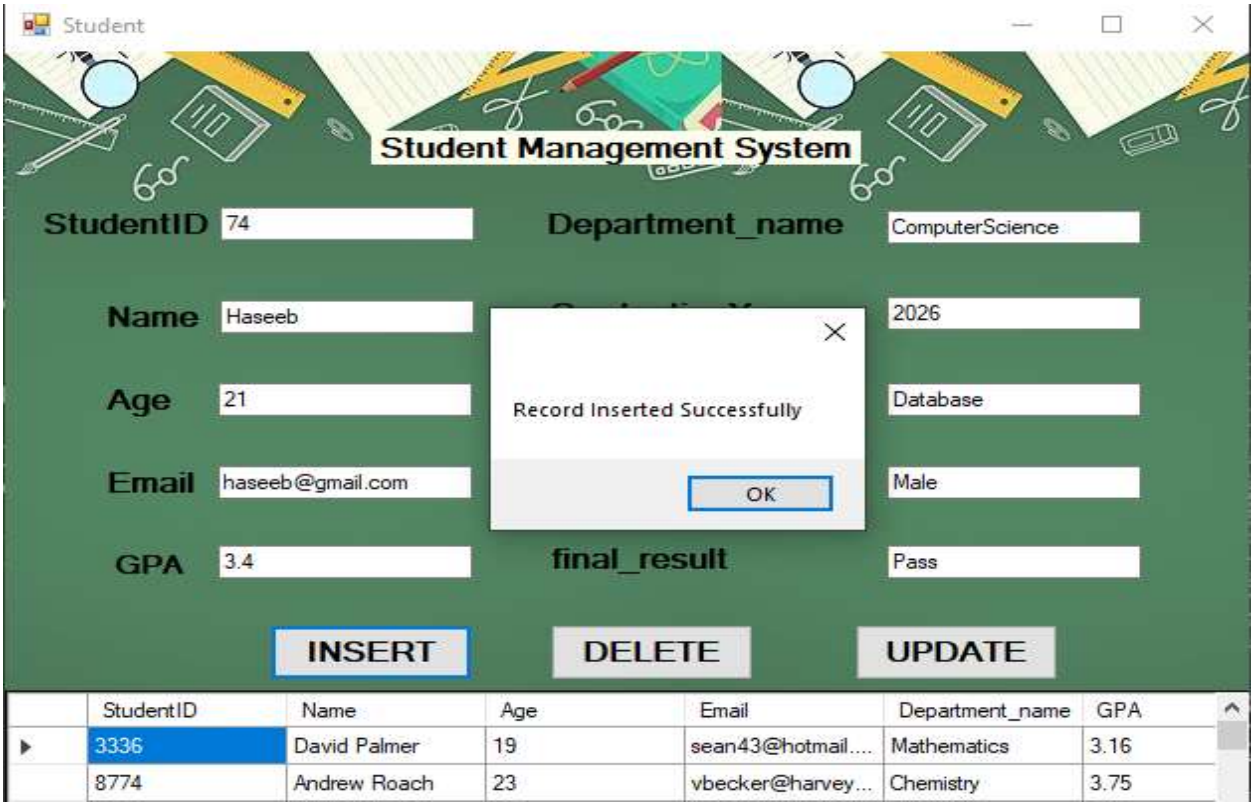

Add Connection
Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.
Data source:
Microsoft SQL Server (SqlClient)
Change...
Server name:
DESKTOP-BD6E7FI\SQLEXPRESS
Refresh
Log on to the server
Authentication:
Windows Authentication
User name:
Password:
Encrypt:
Mandatory (True)
Trust Server Certificate
Save my password
Connect to a database
Select or enter a database name:
SMSS
Attach a database file:
Browse...
Logical name:
Advanced...
Test Connection
OK
Cancel


Add Connection
Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.
Data source:
Microsoft SQL Server (SqlClient)
Change...
Server name:
DESKTOP-BD6E7FI\SQLEXPRESS
Refresh
Log on to the server
Authentication:
Windows Authentication
User name:
Password:
Encrypt:
Microsoft Visual Studio
Test connection succeeded.
OK
Connect to a database
SMSS
Attach a database file:
Browse...
Logical name:
Advanced...
Test Connection
OK
Cancel

INSERT Button

```
private void button1_Click(object sender, EventArgs e)
{
    using (SqlConnection con = new SqlConnection("Data Source=DESKTOP-BD6E7FI\\SQLEXPRESS;Initial Catalog=SMSS;Integrated Security=True;Encrypt=False"))
    {
        string Query = @"INSERT INTO Student
(StudentID, Name, Age, Email, Department_name, GPA, GraduationYear, Course_name, gender, final_result)
VALUES
(@StudentID, @Name, @Age, @Email, @Department_name, @GPA, @GraduationYear, @Course_name, @gender, @final_result)";
        using (SqlCommand cmd = new SqlCommand(Query, con))
        {
            cmd.Parameters.AddWithValue("@StudentID", txtID.Text);
            cmd.Parameters.AddWithValue("@Name", txtname.Text);
            cmd.Parameters.AddWithValue("@Age", txtage.Text);
            cmd.Parameters.AddWithValue("@Email", txtemail.Text);
            cmd.Parameters.AddWithValue("@GPA", txtgpa.Text);
            cmd.Parameters.AddWithValue("@Department_name", txtdepartment_name.Text);
            cmd.Parameters.AddWithValue("@GraduationYear", txtgraduationyear.Text);
            cmd.Parameters.AddWithValue("@Course_name", txtcoursename.Text);
            cmd.Parameters.AddWithValue("@gender", txt_gnder.Text);
            cmd.Parameters.AddWithValue("@final_result", txtfinalresult.Text);
            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
```

```
MessageBox.Show("Record Inserted Successfully");

ClearFields(); // Clear fields after insertion

GetStudentRecord(); // refresh DataGridView to show new data }}}
```

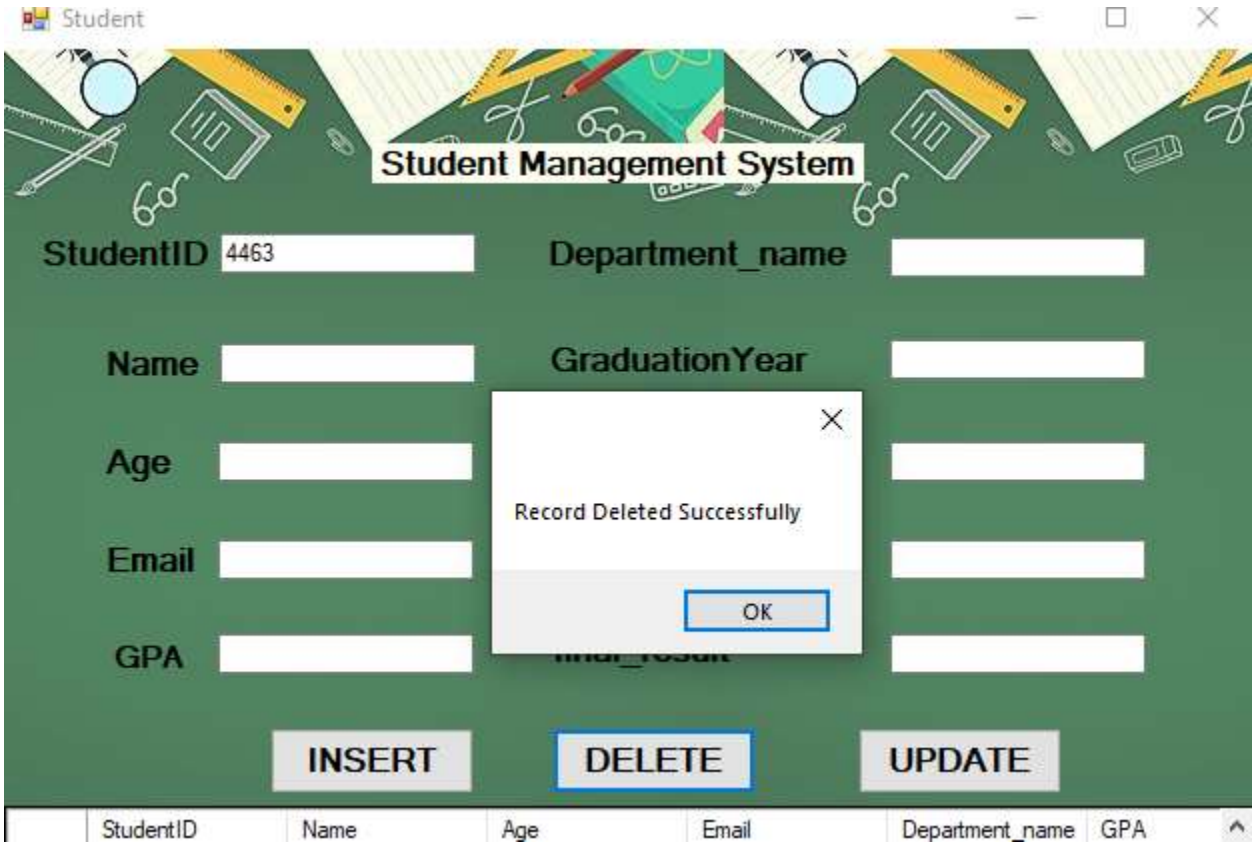


Output

StudentID	Name	Age	Email	Department_name	GPA
9179	Theresa Hughes	21	mstanley@yahoo...	Mathematics	3.23
4463	Raza	22	raza@gmail.com	SoftwareEnginee...	2.9
74	Haseeb	21	haseeb@gmail.c...	ComputerScience	3.4

DELETE button

```
using (SqlConnection con = new SqlConnection("Data Source=DESKTOP-
BD6E7FI\\SQLEXPRESS;Initial Catalog=SMSS;Integrated
Security=True;TrustServerCertificate=True"))

{

   string query = "DELETE FROM Student WHERE StudentID = @StudentID";

   using (SqlCommand cmd = new SqlCommand(query, con))

   {

      cmd.Parameters.AddWithValue("@StudentID", txtID.Text);

      con.Open();

      cmd.ExecuteNonQuery();

      con.Close();

      MessageBox.Show("Record Deleted Successfully");

      ClearFields(); // clear input fields after deletion

      GetStudentRecord(); // refresh DataGridView to show updated records

}}
```



Output

StudentID	Name	Age	Email	Department_name	GPA
3274	Catherine Velasq...	22	fsutton@gmail.com	Physics	2.71
3609	Joyce Thomas	19	zanderson@barr...	Chemistry	2.22
9179	Theresa Hughes	21	mstanley@yahoo...	Mathematics	3.23
74	Haseeb	21	haseeb@gmail.c...	ComputerScience	3.4

UPDATE Button

```
 private void button4_Click(object sender, EventArgs e)

 {

    using (SqlConnection con = new SqlConnection("Data Source=DESKTOP-
BD6E7FI\\SQLEXPRESS;Initial Catalog=SMSS;Integrated
Security=True;TrustServerCertificate=True"))

    {

       string query = @"UPDATE Student

 SET Name = @Name,

    Age = @Age,

    Email = @Email,

    Department_name = @Department_name,

    GPA = @GPA,

    GraduationYear = @GraduationYear,

    Course_name = @Course_name,

    gender = @gender,

    final_result = @final_result

 WHERE StudentID = @StudentID";

       using (SqlCommand cmd = new SqlCommand(query, con))

       {

          cmd.Parameters.AddWithValue("@StudentID", txtID.Text);

          cmd.Parameters.AddWithValue("@Name", txtname.Text);

          cmd.Parameters.AddWithValue("@Age", txtage.Text);

          cmd.Parameters.AddWithValue("@Email", txtemail.Text);
```

```
cmd.Parameters.AddWithValue("@Department_name", txtdepartment_name.Text);

cmd.Parameters.AddWithValue("@GPA", txtgpa.Text);

cmd.Parameters.AddWithValue("@GraduationYear", txtgraduationyear.Text);

cmd.Parameters.AddWithValue("@Course_name", txtcoursename.Text);

cmd.Parameters.AddWithValue("@gender", txt_gnder.Text);

cmd.Parameters.AddWithValue("@final_result", txtfinalresult.Text);

con.Open();

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Record Updated Successfully");

ClearFields(); // clear input fields after updating

GetStudentRecord(); // refresh DataGridView to show updated data}}}
```

StudentID	Name	Age	Email	Department_name	GPA
3274	Catherine Velasq...	22	fsutton@gmail.com	Physics	2.71
3609	Joyce Thomas	19	zanderson@barr...	Chemistry	2.22
9179	Taqi	30	Taqi@gmail.com	SoftwareEnginee...	3

Code Form1.cs

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

using System.Xml.Linq;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.ListView;

using System.Text.RegularExpressions;

namespace Studentform

{

   public partial class Form1 : Form

   {

      private void ClearFields()

      {

         txtID.Clear();

         txtname.Clear();

         txtage.Clear();
```

```
            txtemail.Clear();
            txtdepartment_name.Clear();
            txtgpa.Clear();
            txtgraduationyear.Clear();
            txtcoursename.Clear();
            txt_gnder.Clear();
            txtfinalresult.Clear();
        }
        public Form1()
        {
            InitializeComponent();
            GetStudentRecord();
        }
        private void GetStudentRecord()
        {
            // Define the connection string
            SqlConnection con = new SqlConnection("Data Source=DESKTOP-BD6E7FI\\SQLEXPRESS;Initial
Catalog=SMSS;Integrated Security=True;Encrypt=False");
            // Define the SQL query
            SqlCommand cmd = new SqlCommand("SELECT * FROM Student", con);
            // Initialize a DataTable to hold the query results
            DataTable dt = new DataTable();

            try
```

```
    {
        // Open the connection
        con.Open();
        // Execute the query and load the results into the DataTable
        SqlDataReader sdr = cmd.ExecuteReader();
        dt.Load(sdr);
    }
    catch (Exception ex)
    {
        // Handle any errors that may have occurred
        MessageBox.Show("An error occurred: " + ex.Message);
    }
    finally
    {
        // Close the connection
        con.Close();
    }
    // Bind the DataTable to the DataGridView
    StudentdataGridView1.DataSource = dt;
}
private void textBox1_TextChanged(object sender, EventArgs e)
{
        }
private void textBox6_TextChanged(object sender, EventArgs e)
```

```
        {

        }

        private void label9_Click(object sender, EventArgs e)

        {

        }

        private void textBox5_TextChanged(object sender, EventArgs e)

        {

        }

        private void textBox7_TextChanged(object sender, EventArgs e)

        {

        }

        private void button3_Click(object sender, EventArgs e)

        {

         }

        private void button1_Click(object sender, EventArgs e)

        {

            using (SqlConnection con = new SqlConnection("Data Source=DESKTOP-
BD6E7FI\\SQLEXPRESS;Initial Catalog=SMSS;Integrated Security=True;Encrypt=False"))

            {

                string Query = @"INSERT INTO Student

(StudentID, Name, Age, Email, Department_name, GPA, GraduationYear, Course_name, gender,
final_result)

VALUES

(@StudentID, @Name, @Age, @Email, @Department_name, @GPA, @GraduationYear,
@Course_name, @gender, @final_result)";
```

```
        using (SqlCommand cmd = new SqlCommand(Query, con))
        {
            cmd.Parameters.AddWithValue("@StudentID", txtID.Text);
            cmd.Parameters.AddWithValue("@Name", txtname.Text);
            cmd.Parameters.AddWithValue("@Age", txtage.Text);
            cmd.Parameters.AddWithValue("@Email", txtemail.Text);
            cmd.Parameters.AddWithValue("@GPA", txtgpa.Text);
            cmd.Parameters.AddWithValue("@Department_name", txtdepartment_name.Text);
            cmd.Parameters.AddWithValue("@GraduationYear", txtgraduationyear.Text);
            cmd.Parameters.AddWithValue("@Course_name", txtcoursename.Text);
            cmd.Parameters.AddWithValue("@gender", txt_gnder.Text);
            cmd.Parameters.AddWithValue("@final_result", txtfinalresult.Text);
            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("Record Inserted Successfully");
            ClearFields(); // Clear fields after insertion
            GetStudentRecord(); // refresh DataGridView to show new data
        }} }
            private void Form1_Load(object sender, EventArgs e)
    {
    }
    private void button2_Click(object sender, EventArgs e)
    {
```

```
        using (SqlConnection con = new SqlConnection("Data Source=DESKTOP-
BD6E7FI\\SQLEXPRESS;Initial Catalog=SMSS;Integrated Security=True;TrustServerCertificate=True"))

        {

           string query = "DELETE FROM Student WHERE StudentID = @StudentID";

           using (SqlCommand cmd = new SqlCommand(query, con))

           {

              cmd.Parameters.AddWithValue("@StudentID", txtID.Text);

              con.Open();

              cmd.ExecuteNonQuery();

              con.Close();

              MessageBox.Show("Record Deleted Successfully");

              ClearFields(); // clear input fields after deletion

              GetStudentRecord(); // refresh DataGridView to show updated records

}}}

               private void button4_Click(object sender, EventArgs e)

     {

        using (SqlConnection con = new SqlConnection("Data Source=DESKTOP-
BD6E7FI\\SQLEXPRESS;Initial Catalog=SMSS;Integrated Security=True;TrustServerCertificate=True"))

        {

           string query = @"UPDATE Student

      SET Name = @Name,

        Age = @Age,

        Email = @Email,

        Department_name = @Department_name,

        GPA = @GPA,
```

```
    GraduationYear = @GraduationYear,
    Course_name = @Course_name,
    gender = @gender,
    final_result = @final_result
WHERE StudentID = @StudentID";
        using (SqlCommand cmd = new SqlCommand(query, con))
        {
            cmd.Parameters.AddWithValue("@StudentID", txtID.Text);
            cmd.Parameters.AddWithValue("@Name", txtname.Text);
            cmd.Parameters.AddWithValue("@Age", txtage.Text);
            cmd.Parameters.AddWithValue("@Email", txtemail.Text);
            cmd.Parameters.AddWithValue("@Department_name", txtdepartment_name.Text);
            cmd.Parameters.AddWithValue("@GPA", txtgpa.Text);
            cmd.Parameters.AddWithValue("@GraduationYear", txtgraduationyear.Text);
            cmd.Parameters.AddWithValue("@Course_name", txtcoursename.Text);
            cmd.Parameters.AddWithValue("@gender", txt_gnder.Text);
            cmd.Parameters.AddWithValue("@final_result", txtfinalresult.Text);
            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("Record Updated Successfully");
            ClearFields(); // clear input fields after updating
            GetStudentRecord(); // refresh DataGridView to show updated data
        }}}}}
```