

PROJECT - 1

ABSTRACT:

College Database

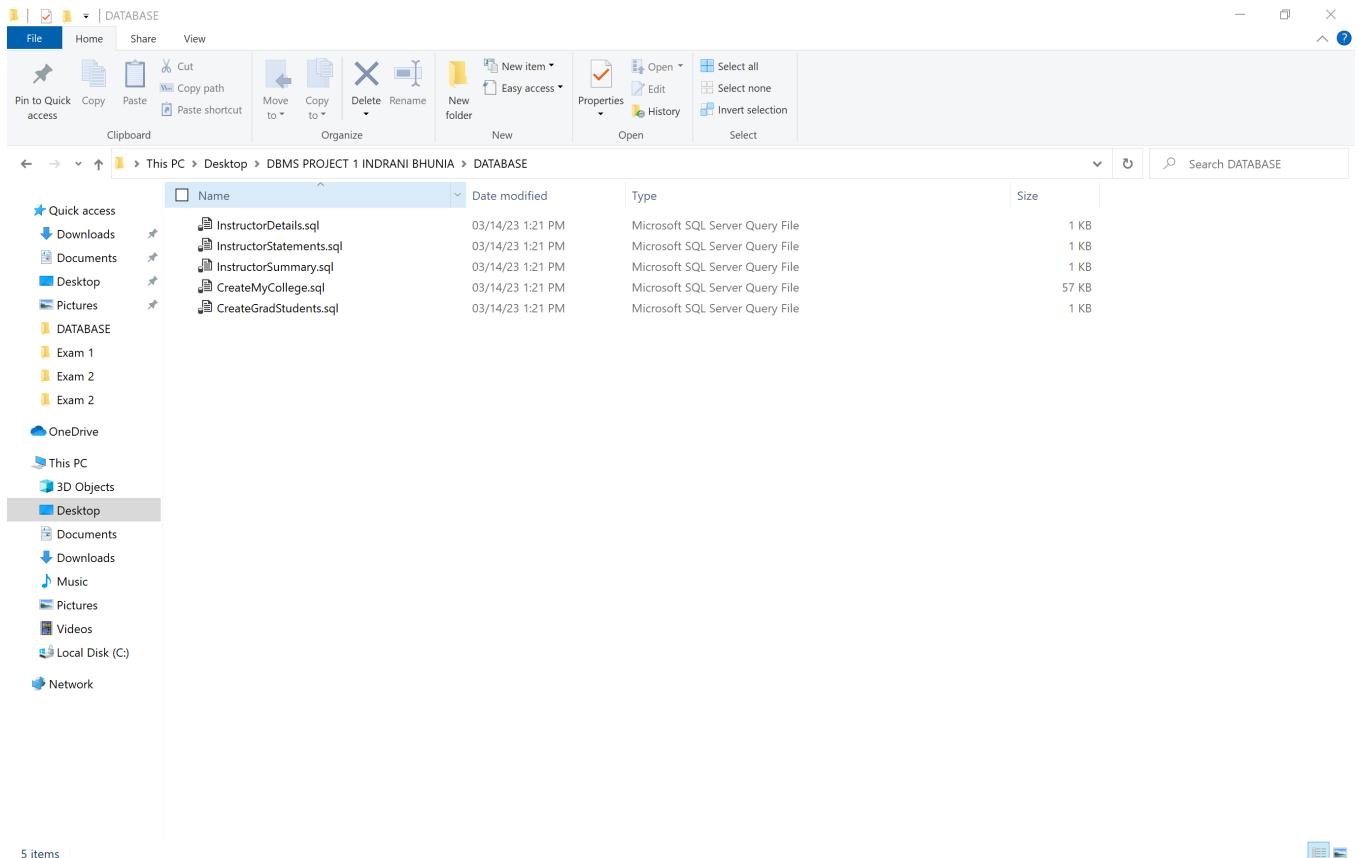
This project focuses on utilizing SQL Server Management Studio to run SQL queries in a college database's database management system. I ran all of the attached files in various query editors in the first section. SQL Joins, Union, Subqueries, and other operations are covered in the second section. In this part, simple DML queries like SELECT, UPDATE, INSERT, and DELETE are employed. It also includes several tasks that use DDL (Data Definition Language) queries. The questions in the following area are based on advanced SQL techniques and involve writing views, stored procedures, functions, and scripts. The final section, which deals with database design, contains two database designs that I created, one for a college database and the other for a website for a book. Additionally, this report includes comments on the project as well as some of the materials I used. I will establish the MyCollege database for this project, generate and enter SQL statements on it, then execute those statements.

TABLE OF CONTENTS:

SR No.	Description	PageNo.
1	Database Setup	3 - 16
2	Essential SQL skills	16 - 54
3	Advanced SQL skills	55-78
4	Database Design	79-82
5	Reference	83
6	Remarks	83

- **PART A, Database Setup -**

1. Download sql files from this project tab and open CreateMyCollege.sql in SQL server management studio. Execute the entire script and show the message in the Message tab, indicating the script is executed successfully.



Comment : Downloaded the SQL files required for the project

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a connection to 'DESKTOP-DU81IMG\SQLEXPRESS' (SQL Server 16.0.1000 - DESKTOP-DU81IMG). The 'master' database is selected. The central pane displays a script named 'CreateMyCollege.sql' with the following content:

```
USE master
GO

IF DB_ID('MyCollege') IS NOT NULL
    DROP DATABASE MyCollege

CREATE DATABASE MyCollege
GO

USE [MyCollege]
GO
/******** Object: Table [dbo].[Courses]    Script Date: 10/12/2022 10:15:00 AM *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Courses](
    [CourseID] [int] IDENTITY(1,1) NOT NULL,
    [CourseNumber] [int] NULL,
    [CourseDescription] [varchar](50) NOT NULL,
    [CourseUnits] [int] NOT NULL,
    [DepartmentID] [int] NOT NULL,
    [InstructorID] [int] NOT NULL,
PRIMARY KEY CLUSTERED
(
    [CourseID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
GO
/******** Object: Table [dbo].[Departments]   Script Date: 10/12/2022 10:15:00 AM *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Departments](
    [DepartmentID] [int] IDENTITY(1,1) NOT NULL,
    [DepartmentName] [varchar](40) NULL,
PRIMARY KEY CLUSTERED
```

Comment :CreateMyCollege.sql

CreateMyCollege.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (59)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

Quick Launch (Ctrl+Q)

Object Explorer

Connect ▾

DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000 - DESKTOP-DU81IMG\abhis) ▾

- atabases
- Security
- Server Objects
- Replication
- Management
- XEvent Profiler

CreateMyCollege... (Execute)

CREATE DATABASE MyCollege

USE master
GO

IF DB_ID('MyCollege') IS NOT NULL
DROP DATABASE MyCollege

CREATE DATABASE MyCollege
GO

USE [MyCollege]
GO

***** Object: Table [dbo].[Courses] Script Date: 10/12/2022 10:15:00 AM *****
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Courses](
[CourseID] [int] IDENTITY(1,1) NOT NULL,
[CourseNumber] [int] NULL,

Messages

(1 row affected)
(1 row affected)

100 %

Query executed successfully.

Comment : Entire script executed, with a message indicating successful script execution in theMessage tab.

2. Do Navigate through the database objects and view the column definitions for each table. Open a new Query Editor window. Show details in Courses table and Instructors table using SELECT statement.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases, tables, and other objects. The central area displays the 'Table Designer' for the 'Courses' table. The 'Columns' tab shows the following columns:

Column Name	Data Type	Allow Nulls
CourseID	int	<input type="checkbox"/>
CourseNumber	int	<input checked="" type="checkbox"/>
CourseDescription	varchar(50)	<input type="checkbox"/>
CourseUnits	int	<input type="checkbox"/>
DepartmentID	int	<input type="checkbox"/>
InstructorID	int	<input type="checkbox"/>

The 'Column Properties' pane on the right shows the properties for the 'CourseID' column under the '(General)' section:

- (Name) CourseID
- Allow Nulls No
- Data Type int
- Default Value or Binding

Comment :Column definition of table – ‘Courses’

DESKTOP-DU81IMG\SQLEXPRESS.MyCollege - dbo.Table_1* - Microsoft SQL Server Management Studio

File Edit View Project Table Designer Tools Window Help

New Query Execute

Object Explorer

Connect Databases MyCollege Tables System Tables FileTables External Tables Graph Tables dbo.Courses dbo.Departments Columns Keys Constraints Triggers Indexes Statistics dbo.GradStudents dbo.Instructors dbo.StudentCourses dbo.Students dbo.Tuition Views External Resources Synonyms Programmability Query Store Service Broker Storage Security MySchool ProductOrders Security

Column Name Data Type Allow Nulls

DepartmentID	int	<input type="checkbox"/>
DepartmentName	varchar(40)	<input checked="" type="checkbox"/>

Column Properties

(General) (Name) InstructorID Allow Nulls No Data Type int Default Value or Binding

Table Designer (General)

Comment : Column definition of table – ‘Departments’

DESKTOP-DU81IMG\SQLEXPRESS.MyCollege - dbo.Table_2* - Microsoft SQL Server Management Studio

File Edit View Project Table Designer Tools Window Help

New Query Execute

Object Explorer

Connect Databases MyCollege Tables System Tables FileTables External Tables Graph Tables dbo.Courses dbo.Departments dbo.GradStudents dbo.Instructors Columns Keys Constraints Triggers Indexes Statistics dbo.StudentCourses dbo.Students dbo.Tuition Views External Resources Synonyms Programmability Query Store Service Broker Storage Security MySchool ProductOrders Security

Column Name Data Type Allow Nulls

InstructorID	int	<input type="checkbox"/>
LastName	varchar(25)	<input type="checkbox"/>
FirstName	varchar(25)	<input checked="" type="checkbox"/>
Status	char(1)	<input type="checkbox"/>
DepartmentChairman	bit	<input type="checkbox"/>
HireDate	date	<input checked="" type="checkbox"/>
AnnualSalary	money	<input type="checkbox"/>
DepartmentID	int	<input type="checkbox"/>

Column Properties

(General) (Name) InstructorID Allow Nulls No Data Type int Default Value or Binding

Table Designer Collation <database default> (General)

Comment : Column definition of table – ‘Instructors’

DESKTOP-DU81IMG\SQLEXPRESS.MyCollege - dbo.Table_3* - Microsoft SQL Server Management Studio

File Edit View Project Table Designer Tools Window Help

New Query MDX DAX XML Spatial Indexes...

Execute

Object Explorer

Connect ▾

DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)

- Databases
 - System Databases
 - Database Snapshots
 - AP
 - Examples
 - MyCollege
- Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.Courses
 - dbo.Departments
 - dbo.GradStudents
 - dbo.Instructors
 - dbo.StudentCourses
 - dbo.Students
 - dbo.Tuition
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Query Store
 - Service Broker
 - Storage
 - Security
- MySchool
- ProductOrders
- Security
- Server Objects
- Replication
- Management
- XEvent Profiler

Ready

DESKTOP-DU81IMG\...ege - dbo.Table_3* SQLQuery1.sql D:\U81IMG\abhis (72)*

Column Name	Data Type	Allow Nulls
StudentID	int	<input type="checkbox"/>
CourseID	int	<input type="checkbox"/>

Column Properties

(General)

(Name) StudentID
Allow Nulls No
Data Type int
Default Value or Binding

Table Designer

Collation <database default>
Computed Column Specification

(General)

Comment :Column definition of table – ‘StudentCourses’

DESKTOP-DU81IMG\SQLEXPRESS.MyCollege - dbo.Table_1* - Microsoft SQL Server Management Studio

File Edit View Project Table Designer Tools Window Help

New Query MDX DAX XML Spatial Indexes...

Execute

Object Explorer

Connect ▾

DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)

- Databases
 - System Databases
 - Database Snapshots
 - AP
 - Examples
 - MyCollege
- Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.Courses
 - dbo.Departments
 - dbo.GradStudents
 - dbo.Instructors
 - dbo.StudentCourses
 - dbo.Students
 - dbo.Tuition
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Query Store
 - Service Broker
 - Storage
 - Security
- MySchool
- ProductOrders
- Security
- Server Objects
- Replication
- Management
- XEvent Profiler

Ready

DESKTOP-DU81IMG\...ege - dbo.Table_1* SQLQuery1.sql D:\U81IMG\abhis (72)*

Column Name	Data Type	Allow Nulls
StudentID	int	<input type="checkbox"/>
LastName	varchar(25)	<input type="checkbox"/>
FirstName	varchar(25)	<input type="checkbox"/>
EnrollmentDate	datetime2(7)	<input type="checkbox"/>
GraduationDate	date	<input type="checkbox"/>

Column Properties

(General)

(Name) StudentID
Allow Nulls No
Data Type int
Default Value or Binding

Table Designer

Collation <database default>
Computed Column Specification

(General)

Comment : Column definition of table – ‘Students’

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'MyCollege' database is selected. In the center pane, the 'Table Designer' is open for the 'Tuition' table. The table has three columns: 'PartTimeCost', 'FullTimeCost', and 'PerUnitCost', all of type 'smallmoney'. The 'Column Properties' window is open for the 'PartTimeCost' column, showing its name, data type ('smallmoney'), and other properties like 'Allow Nulls'.

Comment : Column definition of table – ‘Tuition’

The screenshot shows the Microsoft SQL Server Management Studio interface with a query window titled 'CreateMyCollege...'. The query is 'Select * From MyCollege.dbo.Courses;'. The results grid displays 25 rows of course data. The columns are CourseID, CourseNumber, CourseDescription, CourseUnits, DepartmentID, and InstructorID. The data includes various courses like Beginning Accounting, Abstract Algebra, Primary Education, Anatomy, Social Psychology, etc.

	CourseID	CourseNumber	CourseDescription	CourseUnits	DepartmentID	InstructorID
1	1	36598	Beginning Accounting	3	1	1
2	2	48926	Abstract Algebra	3	4	5
3	3	14862	Primary Education	3	2	8
4	4	54321	Anatomy	3	6	16
5	5	82754	Social Psychology	3	7	9
6	6	13524	Statistical Analysis	3	1	11
7	7	24653	Intro to Marketing	3	1	4
8	8	2579	Intro to Calculus	3	4	5
9	9	98765	Intermediate Accounting	3	1	1
10	10	96032	Smart Maths	3	1	4
11	11	58230	Physiology	3	6	16
12	12	81266	Intro to Management	3	1	7
13	13	64321	Secondary Education	3	2	8
14	14	32751	Business Writing	2	1	10
15	15	46972	Biology	4	6	3
16	16	15487	Music Theory	3	5	12
17	17	28177	Classic Literature	3	3	15
18	18	90908	Educational Theory	3	2	13
19	19	55783	Shakespeare	3	3	15
20	20	65324	Population and Demographics	3	7	9
21	21	74832	Creative Writing	3	3	2

Comment : Details in Courses table, table No of rows : 25

SQLQuery5.sql - DESKTOP-DU81IMG\SQLEXPRESS.master (DESKTOP-DU81IMG\abhis (53)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

master Execute

Object Explorer

SQLQuery5.sql - D...U81IMG\abhis (53)* CreateMyCollege...U81IMG\abhis (59)

```
Select *
From MyCollege.dbo.Instructors;
```

Results Messages

InstructorID	LastName	FirstName	Status	DepartmentChairman	HireDate	AnnualSalary	DepartmentID
1	Brown	Billy	F	1	2016-01-10	77500.00	1
2	Thomas	William	P	0	2016-03-30	38500.00	3
3	Amundsen	Rachel	F	1	2016-06-05	79000.00	6
4	Green	Gene	F	0	2016-08-02	75000.00	1
5	McGregor	NULL	F	1	2017-01-09	74000.00	4
6	Paxton	Arnold	P	0	2017-07-18	36000.00	5
7	Rogers	NULL	P	0	2017-10-22	38000.00	1
8	Smith	John	F	1	2018-05-05	73000.00	2
9	Connors	Daniel	F	1	2018-03-04	35000.00	7
10	Jones	Sam	F	1	2018-09-21	74000.00	3
11	Vilms	Jonathan	P	0	2018-11-18	35500.00	1
12	Thomas	Demick	P	0	2019-01-17	35500.00	5
13	Black	Bill	P	0	2019-04-20	34000.00	2
14	Warren	Angela	P	0	2019-07-14	33000.00	4
15	Drew	Daniel	F	0	2019-08-25	72000.00	3
16	Gallegos	Tomas	F	0	2020-03-23	64000.00	6

Query executed successfully.

DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... master 00:00:00 16 rows

Ready Col 2 Ch 31 INS

Comment :Details in Instructors, table No of rows : 16

3. Open another Query Editor window and then enter and run this statement:
SELECT COUNT(*) AS NumberOfInstructors FROM Instructors

Query –

```
SELECT COUNT(*) AS  
NumberOfInstructors FROM  
Instructors;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure of 'MyCollege'. The 'Tables' node under 'MyCollege' is expanded, showing various tables like 'Courses', 'Departments', 'Instructors', etc. In the center, a query window titled 'SQLQuery5.sql' contains the following SQL code:

```
Select COUNT(*) AS NumberOfInstructors  
From Instructors;
```

Below the query window, the results pane shows a single row of data:

1	16
	NumberofInstructors

At the bottom of the results pane, a message says 'Query executed successfully.' The status bar at the bottom of the screen shows 'Ready', 'Ln 2', 'Col 18', 'Ch 18', and 'INS'.

Comment : Query result output, showing the total number of instructors in Instructors table.
No of rows : 1

4. Open the script named InstructorDetails.sql. Note that this script contains just one SQL statement. Then, run the statement.

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "InstructorDetails.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The Object Explorer sidebar shows a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". The main pane displays the query "InstructorDetails.sql" with the following SQL code:

```
SELECT LastName, FirstName, HireDate, AnnualSalary
FROM Instructors
ORDER BY AnnualSalary;
```

The Results tab shows the output of the query, which is a table with 16 rows of instructor data:

	LastName	FirstName	HireDate	AnnualSalary
1	Warren	Angela	2019-07-14	33000.00
2	Black	Bill	2019-04-20	34000.00
3	Vilma	Jonathan	2018-11-18	35500.00
4	Thomas	Derrick	2019-01-17	35500.00
5	Paxton	Arnold	2017-07-15	36000.00
6	Rogers	NULL	2017-10-22	38000.00
7	Thomas	Villiam	2016-03-30	38500.00
8	Gallegos	Tomas	2020-03-23	64000.00
9	Connors	Daniel	2018-03-04	71500.00
10	Smith	John	2018-02-05	73000.00
11	McGregor	NULL	2017-01-03	74000.00
12	Green	Gene	2016-08-02	75000.00
13	Brown	Billy	2016-01-10	77500.00
14	Amundsen	Rachel	2016-06-05	79000.00
15	Drew	Daniel	2019-08-25	79200.00
16	Jones	Sally	2018-09-21	81400.00

The status bar at the bottom right indicates "DESKTOP-DU81IMG\SQLEXPRESS ... | DESKTOP-DU81IMG\abhis ... | MyCollege | 00:00:00 | 16 rows".

Comment : Query result output, showing the Annual Salary of Instructors & noted the script contains only one SQL statement. No of rows : 16

5. Open the script named InstructorSummary.sql. Note that this opens another QueryEditor window.

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "InstructorSummary.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The menu bar includes File, Edit, View, Query, Project, Tools, Window, and Help. The toolbar has various icons for file operations like New Query, Save, Print, and Execute. The Object Explorer on the left shows the database structure under "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". A query editor window titled "InstructorSummary....U81IMG\abhis (62)" contains the following SQL code:

```
SELECT COUNT(*) AS NumberOfInstructors,
       MAX(AnnualSalary) AS MaxSalary,
       AVG(AnnualSalary) AS AverageSalary
  FROM Instructors;
```

The Results pane below shows the output of the query:

	NumberOfInstructors	MaxSalary	AverageSalary
1	16	81400.00	57818.75

At the bottom, a status bar indicates "Query executed successfully." and "1 rows".

Comment : Query result output, showing the Maximum and Average Salary of Instructors.
No of rows : 1

6. Open the script named InstructorStatements.sql. Note that this script contains two SQL statements that end with semicolons.

```

InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
New Query Execute
Object Explorer
Connect ▾ MyCollege
Databases Security Server Objects Replication Management XEvent Profiler
InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62))
SELECT LastName, FirstName, HireDate, AnnualSalary
FROM Instructors
ORDER BY AnnualSalary;

SELECT COUNT(*) AS NumberOfInstructors,
       MAX(AnnualSalary) AS MaxSalary,
       AVG(AnnualSalary) AS AverageSalary
FROM Instructors;
    
```

	Last Name	First Name	Hire Date	Annual Salary
4	Thomas	Derrick	2019-01-17	35500.00
5	Paxton	Arnold	2017-07-15	36000.00
6	Rogers	NULL	2017-10-22	38000.00
7	Thomas	William	2016-03-30	38500.00
8	Gallegos	Tomas	2020-03-23	64000.00
9	Connor	Daniel	2018-03-04	71500.00
10	Smith	John	2018-02-05	73000.00
11	McGregor	NULL	2017-01-03	74000.00
12	Green	Gene	2016-08-02	75000.00
13	Brown	Billy	2016-01-10	77500.00
14	Amundsen	Rachel	2016-06-05	79000.00
15	Drew	Daniel	2019-08-25	79200.00
16	Jones	Sally	2018-09-21	81400.00

Number of Instructors	Max Salary	Average Salary
16	81400.00	57818.75

Query executed successfully.

Comment :Opened the script named InstructorStatement.sql and noted that the statements that end with semicolons, which indicated that they are 2 different SQL query statements. No of rows : 16 and for query2 No of rows : 1

7. Press the F5 key or click the Execute button to run both statements in this script. Note that this displays the results in two Results tabs. Make sure to view the results of both SELECT statements.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a connection to 'DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)' is selected, with the database 'MyCollege' chosen. A query window titled 'InstructorStatements..U81IMG\abhis (62) -' contains the following T-SQL script:

```

SELECT LastName, FirstName, HireDate, AnnualSalary
FROM Instructors
ORDER BY AnnualSalary;

SELECT COUNT(*) AS NumberOfInstructors,
       MAX(AnnualSalary) AS MaxSalary,
       AVG(AnnualSalary) AS AverageSalary
FROM Instructors;

```

Below the script, two Results tabs are displayed. The first tab, 'Results', shows the individual instructor records from the 'Instructors' table, with 17 rows of data. The second tab, 'Messages', shows the summary statistics: 16 instructors, a maximum salary of 81400.00, and an average salary of 57818.75.

Last Name	First Name	Hire Date	Annual Salary
Thomas	Derrick	2019-01-17	35500.00
Paxton	Arnold	2017-07-15	36000.00
Rogers	NULL	2017-10-22	38000.00
Thomas	William	2016-03-30	38500.00
Galligos	Tomas	2020-03-23	64000.00
Connors	Daniel	2018-03-04	71500.00
Smith	John	2018-02-05	73000.00
McGregor	NULL	2017-01-03	74000.00
Green	Gene	2016-08-02	75000.00
Brown	Billy	2016-01-10	77500.00
Amundsen	Rachel	2015-05-05	79000.00
Drew	Daniel	2019-08-25	79200.00
Jones	Sally	2018-09-21	81400.00

Number of Instructors	Max Salary	Average Salary
16	81400.00	57818.75

Comment : Executed both the statements in the script.

Noted the results in the two Results tabs, which showcase the Annual Salary, MaxSalary and AverageSalary of Instructors. No of rows : 17

8. Exit from SQL Server Management Studio.

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The menu bar includes File, Edit, View, Query, Project, Tools, Window, and Help. The toolbar has various icons for file operations like New Query, Save, and Execute. The Object Explorer on the left shows a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)" with nodes for Databases, Security, Server Objects, Replication, Management, and XEvent Profiler. The main pane displays a query window titled "InstructorStatements...U81IMG\abhis (62)". The query is:`SELECT LastName, FirstName, HireDate, AnnualSalary
FROM Instructors
ORDER BY AnnualSalary;

SELECT COUNT(*) AS NumberOfInstructors,
 MAX(AnnualSalary) AS MaxSalary,
 AVG(AnnualSalary) AS AverageSalary
FROM Instructors;`

Below the query window is a results grid titled "Results". It contains two tables. The first table lists 16 rows of instructor data with columns: LastName, FirstName, HireDate, and AnnualSalary. The second table summarizes the data with columns: NumberOfInstructors, MaxSalary, and AverageSalary. The status bar at the bottom right indicates "Query executed successfully.", "DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 17 rows", and "Ln 1 Col 1 Ch 1 INS".

Comment :Closing SQL Server Management Studio. No of rows : 1

- **PART B, Essential SQL Skills –**

1. Write a SELECT statement that returns all of the columns from the Courses table.
Then, run this statement to make sure it works correctly.

Query –

```
SELECT *
FROM Courses;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure for 'MyCollege'. In the center, the 'Query Editor' pane contains the following SQL code:

```
SELECT *
FROM MyCollege.dbo.Courses;
```

The results pane on the right shows the output of the query, which is a table with 25 rows of course data. The columns are: CourseID, CourseNumber, CourseDescription, CourseUnits, DepartmentID, and InstructorID. The data includes various courses like Beginning Accounting, Abstract Algebra, Primary Education, Anatomy, Social Psychology, etc.

CourseID	CourseNumber	CourseDescription	CourseUnits	DepartmentID	InstructorID
1	36598	Beginning Accounting	3	1	1
2	48926	Abstract Algebra	3	4	5
3	14862	Primary Education	3	2	8
4	54321	Anatomy	3	6	16
5	82754	Social Psychology	3	7	9
6	13524	Statistical Analysis	3	1	11
7	24653	Intro to Marketing	3	1	4
8	22679	Intro to Calculus	3	4	5
9	98765	Intermediate Accounting	3	1	1
10	96032	Social Media	3	1	4
11	58230	Physiology	3	6	16
12	81256	Intro to Management	3	1	7
13	64321	Secondary Education	3	2	8
14	32751	Business Writing	2	1	10
15	46972	Biology	4	6	3
16	15487	Music Theory	3	5	12
17	28177	Classic Literature	3	3	15
18	90900	Educational Theory	3	2	13
19	55783	Shakespeare	3	3	15
20	63284	Population and Demographics	3	7	9
21	74832	Creative Writing	3	3	2

At the bottom of the results pane, a message indicates: "Query executed successfully." Below the results pane, the status bar shows: "LN 2 Col 28 Ch 28 INS".

Comment : Executed the SELECT statement that returns all the columns of Courses table.
No of rows : 25

2. Write a SELECT statement that returns one column from the Students table named FullName that joins the LastName and FirstName columns. Format this column with the last name, a comma, a space, and the first name. Sort the result set by last name in ascending sequence. Return only the students whose last name begins with a letter from A to M.

Query –

```
SELECT LastName + ',' + ' ' + FirstName AS
FullName
FROM Students
WHERE LastName LIKE '[A-M]%'
ORDER BY LastName ASC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window contains the following SQL code:

```
SELECT LastName + ',' + ' ' + FirstName AS FullName
FROM Students
WHERE LastName LIKE '[A-M]%'
ORDER BY LastName ASC;
```

The results pane displays a table with 28 rows, each containing a student's full name in the format "LastName, FirstName". The rows are numbered 1 through 28. The data is as follows:

	FullName
1	Biden, Jonas
2	Bonwell, Brian
3	Butler, George
4	Cadden, James
5	Clement, Cal
6	Cramsdon, Walter
7	DeLorean, Cameron
8	Easton, Barney
9	Flores, Jesus
10	Franks, Karen
11	Gardner, Faye
12	Geary, Annette
13	George, Mona
14	Goodell, Conner
15	Griffin, Gerald
16	Hallowell, Jimmy
17	Hoffman, Wilma
18	Howard, Amber
19	Jackson, Floyd
20	Johnson, Timothy
21	Jones, Andrew

At the bottom of the results pane, a message indicates: "Query executed successfully." The status bar at the bottom right shows "DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 28 rows".

Comment :Successfully executed a SELECT Statement that returns a column 'FullName' inthe required format using Students table. No of rows : 28

3. Write a SELECT statement that returns these column names and data from theInstructors table:

LastName The LastName
column
FirstName The FirstName
column
HireDate The HireDate column

Return only the rows with a hire date that's in 2019. Sort the result set in ascending sequence by the HireDate column.

YEAR() – The YEAR() function returns the year part for a specified date.

Query –

```
SELECT LastName, FirstName,
HireDate
FROM Instructors
WHERE YEAR(HireDate) = 2019
ORDER BY HireDate ASC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The main area has a query window with the following content:

```
SELECT LastName, FirstName,
HireDate
FROM Instructors
WHERE YEAR(HireDate) = 2019
ORDER BY HireDate ASC;
```

Below the query window is a results grid titled "Results" showing the following data:

	LastName	FirstName	HireDate
1	Thomas	Derrick	2019-01-17
2	Black	Bill	2019-04-20
3	Warren	Angela	2019-07-14
4	Drew	Daniel	2019-08-25

A status bar at the bottom indicates "Query executed successfully." and "4 rows".

Comment :SELECT query result output, displaying the column names and data from Instructors table according to the requirements using YEAR function and ORDER BY.

Noof rows : 4

4. Write a SELECT statement that returns these column names and data from the Studentstable:

FirstName	The FirstName column
LastName	The LastName column
EnrollmentDate	The EnrollmentDate
columnCurrentDate	The current date
MonthsAttended	A column that's calculated by getting the difference betweenthe enrollment date and the current date Sort the result set in ascending sequence by the MonthsAttended column.

Query –

```
SELECT FirstName, LastName, EnrollmentDate, GETDATE() AS CurrentDate, DATEDIFF(day, EnrollmentDate, GETDATE()) AS MonthsAttended
FROM Students
ORDER BY MonthsAttended ASC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with a connection to 'DESKTOP-DU81IMG\SQLEXPRESS'. The right pane contains a query window with the following content:

```
SELECT FirstName, LastName, EnrollmentDate, GETDATE() AS CurrentDate,
DATEDIFF(day, EnrollmentDate, GETDATE()) AS MonthsAttended
FROM Students
ORDER BY MonthsAttended ASC;
```

Below the query window is the Results pane, which shows the output of the query. The results are presented in a table with the following columns: FirstName, LastName, EnrollmentDate, CurrentDate, and MonthsAttended. The data consists of 43 rows, each representing a student's information and their calculated months attended. The 'CurrentDate' column shows the current date as 2023-03-15. The 'MonthsAttended' column shows the calculated difference in days, divided by 30, ranging from 1165 down to 1912.

	FirstName	LastName	EnrollmentDate	CurrentDate	MonthsAttended
1	Jonas	Biden	2020-01-05 13:47:21	2023-03-15 11:27:22.683	1165
2	Stanley	Silver	2020-01-04 11:16:19	2023-03-15 11:27:22.683	1166
3	Mona	George	2019-12-22 15:29:44	2023-03-15 11:27:22.683	1179
4	Lisa	Lile	2019-12-17 11:42:28	2023-03-15 11:27:22.683	1184
5	Walter	Crammeder	2019-09-05 13:48:37	2023-03-15 11:27:22.683	1191
6	Andrew	Wade	2019-09-05 13:48:37	2023-03-15 11:27:22.683	1191
7	Timothy	Johnson	2018-08-24 09:01:04	2023-03-15 11:27:22.683	1319
8	Kenny	Franks	2018-07-23 10:42:03	2023-03-15 11:27:22.683	1331
9	Faye	Gardner	2018-07-22 08:15:57	2023-03-15 11:27:22.683	1332
10	Conner	Goodell	2018-01-02 14:21:58	2023-03-15 11:27:22.683	1533
11	Gerald	Griffin	2018-01-02 16:04:04	2023-03-15 11:27:22.683	1533
12	Vincent	Manning	2018-12-14 15:37:43	2023-03-15 11:27:22.683	1552
13	Lettie	Osborne	2018-12-12 17:14:41	2023-03-15 11:27:22.683	1554
14	Floyd	Jackson	2018-07-28 09:27:53	2023-03-15 11:27:22.683	1694
15	Andrew	Jones	2018-07-24 10:53:26	2023-03-15 11:27:22.683	1695
16	Tanya	Sommers	2018-07-22 15:41:12	2023-03-15 11:27:22.683	1697
17	Annette	Geary	2018-07-12 09:33:47	2023-03-15 11:27:22.683	1707
18	James	Camden	2018-01-04 11:12:31	2023-03-15 11:27:22.683	1896
19	Barney	Easton	2018-01-04 14:14:01	2023-03-15 11:27:22.683	1896
20	Jesus	Florès	2018-01-03 16:23:47	2023-03-15 11:27:22.683	1897
21	Jimmy	Hallowell	2017-12-19 13:44:25	2023-03-15 11:27:22.683	1912

At the bottom of the results pane, it says 'Query executed successfully.'

Comment :Successfully executed a SELECT statement that returns the required column names and data using GETDATE() and DATEDIFF() function; ordered them in ascendingorder using ORDER BY. No of rows : 43

5. Write a SELECT statement that returns these column names and data from theInstructors table:

FirstName	The FirstName column
LastName	The LastName column
AnnualSalary	The AnnualSalary column

Return only the top 20 percent of instructors based on annual salary.

TOP clause – The TOP clause is used to specify the number of records to return in both numeric and percentage format.

Query –

```
SELECT TOP 20 PERCENT FirstName AS 'The FirstName column',
LastName AS 'The LastName column', AnnualSalary AS 'The
AnnualSalary column'
FROM Instructors
ORDER BY AnnualSalary DESC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases, including MyCollege, which contains tables like AP, Examples, and Instructors. The central pane shows three tabs: SQLQuery5.sql, SQLQuery4.sql, and SQLQuery3.sql. The SQLQuery3.sql tab contains the executed query:

```
SELECT TOP 20 PERCENT FirstName AS 'The FirstName column',
LastName AS 'The LastName column', AnnualSalary AS 'The
AnnualSalary column'
FROM Instructors
ORDER BY AnnualSalary;
```

The Results pane below shows the output:

	The FirstName column	The LastName column	The AnnualSalary column
1	Angela	Warren	33000.00
2	Bill	Black	34000.00
3	Jonathan	Vilma	35500.00
4	Derrick	Thomas	35500.00

At the bottom, a message bar indicates "Query executed successfully." and shows statistics: Ln 4, Col 23, Ch 23, INS.

Comment :Successfully executed a SELECT statement that returns the required column names and data using TOP clause and ordered them based on annual salary using ORDER BY. No of rows : 4

6. Write a SELECT statement that returns these columns and data from the Tuition table, along with a constant value and two calculated values:

FullTimeCost	The FullTimeCost
columnPerUnitCost	The PerUnitCost column
Units	12
TotalPerUnitCost	A column that's calculated by multiplying the per unit cost by the units
TotalTuition	A column that's calculated by adding the full time cost to the total per unit cost

Query –

```
SELECT FullTimeCost, PerUnitCost, Units = 12,
       PerUnitCost*12 AS TotalPerUnitCost,
       FullTimeCost + (PerUnitCost*12) AS
       TotalTuition
FROM Tuition;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The menu bar includes File, Edit, View, Query, Project, Tools, Window, Help. The toolbar has various icons for file operations. The Object Explorer on the left shows a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)" with nodes for Databases, Security, Server Objects, Replication, Management, and XEvent Profiler. The main window contains a query editor tab titled "InstructorStatement...U81IMG\abhis (62)*" with the following SQL code:

```
SELECT FullTimeCost, PerUnitCost, Units = 12,
       PerUnitCost*12 AS TotalPerUnitCost,
       FullTimeCost + (PerUnitCost*12) AS
       TotalTuition
FROM Tuition;
```

Below the query editor is a results grid titled "Results" showing one row of data:

	FullTimeCost	PerUnitCost	Units	TotalPerUnitCost	TotalTuition
1	1250.00	62.50	12	750.00	2000.00

At the bottom of the results grid, a message says "Query executed successfully." and "1 rows". The status bar at the bottom of the screen shows "Ready", "Ln 4", "Col 14", "Ch 14", and "INS".

Comment :Successfully executed a SELECT statement that returns the required columns and data from Tuition table using SQL arithmetic operators. No of rows : 1

7. Write a SELECT statement that joins the Courses table to the Departments table and returns these columns: CourseNumber, CourseDescription, DepartmentName. Sort the result set by DepartmentName and then by CourseNumber in ascending order.

Query –

```
SELECT CourseNumber, CourseDescription,
DepartmentName FROM Courses JOIN Departments
ON Courses.DepartmentID =
Departments.DepartmentID ORDER BY
DepartmentName, CourseNumber ASC
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window contains the following SQL code:

```
SELECT CourseNumber, CourseDescription, DepartmentName
FROM Courses JOIN Departments
ON Courses.DepartmentID = Departments.DepartmentID
ORDER BY DepartmentName, CourseNumber ASC
```

The results pane displays a table with 25 rows, showing the joined data from the Courses and Departments tables. The columns are CourseNumber, CourseDescription, and DepartmentName. The data includes various course numbers like 13524, 24653, etc., descriptions like 'Statistical Analysis', 'Intro to Marketing', etc., and department names like 'Business', 'English', etc.

	CourseNumber	CourseDescription	DepartmentName
1	13524	Statistical Analysis	Business
2	24653	Intro to Marketing	Business
3	32751	Business Writing	Business
4	36598	Beginning Accounting	Business
5	81256	Intro to Management	Business
6	96032	Social Media	Business
7	98765	Intermediate Accounting	Business
8	14862	Primary Education	Education
9	64321	Secondary Education	Education
10	90908	Educational Theory	Education
11	28177	Classic Literature	English
12	37645	Composition	English
13	55783	Shakespeare	English
14	74832	Creative Writing	English
15	22679	Intro to Calculus	Mathematics
16	44386	Trigonometry	Mathematics
17	48926	Abstract Algebra	Mathematics
18	15487	Music Theory	Music
19	33218	Marching Band	Music
20	46972	Biology	Science
21	54321	Anatomy	Science

Comment :Successfully executed the SELECT statement that returns the 3 required columns after applying JOIN operation between Courses and Departments table; ordered them in ascending order using ORDER BY. No of rows : 25

8. Write a SELECT statement that joins the Instructors table to the Courses table and returns these columns: LastName, FirstName, CourseNumber, CourseDescription.

Return all courses for each instructor with a status of "P" (part time). Sort the result set by LastName and then by FirstName in ascending order.

Query –

```
SELECT LastName, FirstName, CourseNumber, CourseDescription,
Status
FROM Instructors JOIN Courses
ON Instructors.InstructorID =
Courses.InstructorID
WHERE Status = 'P'
ORDER BY LastName, FirstName ASC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The toolbar has various icons for file operations, queries, and database management. The left pane is the Object Explorer, showing the connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)" and its databases, security, server objects, replication, and management. The main pane contains a query window titled "InstructorStatement...J81IMG\abhis (62)*" with the following SQL code:

```
SELECT LastName, FirstName, CourseNumber, CourseDescription, Status
FROM Instructors JOIN Courses
ON Instructors.InstructorID = Courses.InstructorID
WHERE Status = 'P'
ORDER BY LastName, FirstName ASC;
```

Below the query window is a results grid titled "Results" showing the output of the query. The columns are LastName, FirstName, CourseNumber, CourseDescription, and Status. The data is as follows:

	LastName	FirstName	CourseNumber	CourseDescription	Status
1	Black	Bill	90908	Educational Theory	P
2	Rogers	NULL	81256	Intro to Management	P
3	Thomas	Derrick	15487	Music Theory	P
4	Thomas	Derrick	33218	Marching Band	P
5	Thomas	William	74832	Creative Writing	P
6	Vilma	Jonathan	13524	Statistical Analysis	P
7	Warren	Angela	44386	Trigonometry	P

At the bottom of the results grid, a message says "Query executed successfully." The status bar at the bottom right shows "DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 7 rows".

Comment :Successfully executed a SELECT statement that returns only the required rows of data with status "P" using JOIN operation between Instructors and Courses table; ordered them in the necessary manner using ORDER BY. No of rows : 7

9. Use the UNION operator to generate a result set consisting of five columns from theStudents table:

Status	A calculated column that contains a value of UNDERGRAD or GRADUATED
FirstName	The FirstName column
LastName	The LastName column
EnrollmentDate	The EnrollmentDate column
GraduationDate	The GraduationDate column

If the student doesn't have a value in the GraduationDate column, the Status column should contain a value of UNDERGRAD. Otherwise, it should contain a value of GRADUATED.

Sort the final result set by

EnrollmentDate.Query –

```
SELECT 'UNDERGRAD' AS Status, FirstName, LastName,  
EnrollmentDate, GraduationDate  
FROM Students  
WHERE GraduationDate IS  
NULLUNION  
SELECT 'GRADUATED' AS Status, FirstName, LastName,  
EnrollmentDate, GraduationDate  
FROM Students  
WHERE GraduationDate IS NOT  
NULLORDER BY EnrollmentDate;
```

InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

New Query

MyCollege

Object Explorer

```

SELECT 'UNDERGRAD' AS Status, FirstName, LastName, EnrollmentDate,
GraduationDate
FROM Students
WHERE GraduationDate IS NULL
UNION
SELECT 'GRADUATED' AS Status, FirstName, LastName, EnrollmentDate,
GraduationDate
FROM Students
WHERE GraduationDate IS NOT NULL
ORDER BY EnrollmentDate;

```

Results Messages

Status	FirstName	LastName	EnrollmentDate	GraduationDate
23 UNDERGRAD	Jimmy	Hallowell	2017-12-19 13:44:25	NULL
24 UNDERGRAD	Jesus	Flores	2018-01-03 16:23:47	NULL
25 UNDERGRAD	James	Camden	2018-01-04 11:12:31	NULL
26 UNDERGRAD	Barney	Easton	2018-01-04 14:14:02	NULL
27 UNDERGRAD	Annette	Geary	2018-01-12 09:33:47	NULL
28 UNDERGRAD	Tanya	Sommer	2018-07-22 15:41:12	NULL
29 UNDERGRAD	Andrew	Jones	2018-07-24 10:53:26	NULL
30 UNDERGRAD	Floyd	Jackson	2018-07-25 09:27:53	NULL
31 UNDERGRAD	Lettie	Osborne	2018-12-12 17:14:22	NULL
32 UNDERGRAD	Vincent	Manning	2018-12-14 15:37:45	NULL
33 UNDERGRAD	Conner	Goodell	2019-01-02 14:21:58	NULL
34 UNDERGRAD	Gerald	Griffin	2019-01-02 16:04:04	NULL
35 UNDERGRAD	Faye	Gardner	2019-07-22 08:15:57	NULL
36 UNDERGRAD	Karen	Frank	2019-07-23 10:42:03	NULL
37 UNDERGRAD	Timothy	Johnson	2019-08-04 09:01:04	NULL
38 UNDERGRAD	Andrew	Walker	2019-08-05 13:48:26	NULL
39 UNDERGRAD	Walter	Cramdell	2019-12-15 10:18:37	NULL
40 UNDERGRAD	Lisa	Lisle	2019-12-17 11:42:28	NULL
41 UNDERGRAD	Mona	George	2019-12-22 15:29:44	NULL
42 UNDERGRAD	Stanley	Silver	2020-01-04 11:16:19	NULL
43 UNDERGRAD	Jonas	Biden	2020-01-05 13:47:21	NULL

Query executed successfully.

DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 43 rows

Ready

Comment :Successfully executed the query that returns the required (data) rows of UNDERGRAD and GRADUATED students using UNION and SQL NULL values. No ofrows : 43

10. Write a SELECT statement that returns these columns:
InstructorDept The DepartmentName column from the Departments
table for a related instructor

LastName The LastName column from the Instructors table
FirstName The FirstName column from the Instructors table
CourseDescription The CourseDescription column from the Courses table
CourseDept The DepartmentName column from the Departments table
for a related instructor

Return one row for each course that's in a different department than the department of the instructor assigned to teach that course.

(Hint: You will need to join the Departments table to both the Instructors table and the Courses table, which will require you to use table aliases to distinguish the two tables.)

Query –

```
SELECT ID.DepartmentName AS
    InstructorDept, LastName,
    FirstName,
    CourseDescriptio
    n,
    CD.DepartmentName AS
CourseDept FROM Courses AS C JOIN
Departments AS CD
    ON C.DepartmentID =
    CD.DepartmentID JOIN Instructors
    AS I
    ON C.InstructorID =
    I.InstructorID JOIN Departments
    AS ID
    ON I.DepartmentID = ID.DepartmentID
WHERE CD.DepartmentName <>
ID.DepartmentName;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The left pane is the Object Explorer, showing a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". The right pane contains a query window titled "InstructorStatements..U81IMG\abhis (62)*" with the following SQL code:

```
SELECT ID.DepartmentName AS InstructorDept,
       LastName,
       FirstName,
       CourseDescription,
       CD.DepartmentName AS CourseDept
  FROM Courses AS C JOIN Departments AS CD
    ON C.DepartmentID = CD.DepartmentID
   JOIN Instructors AS I
    ON C.InstructorID = I.InstructorID
   JOIN Departments AS ID
    ON I.DepartmentID = ID.DepartmentID
 WHERE CD.DepartmentName <> ID.DepartmentName;
```

Below the query window is a results grid titled "Results" with one row of data:

	InstructorDept	LastName	FirstName	CourseDescription	CourseDept
1	English	Jones	Sally	Business Writing	Business

The status bar at the bottom indicates "Query executed successfully." and "1 rows".

Comment :Successfully executed a query that returned one row for each course that's in a different department than the department of the instructor assigned to teach that course using JOIN that joined three tables i.e Courses, Department and Instructors. No of rows : 1

11. Write a SELECT statement that returns one row for each instructor that has courses with these columns:

The instructor first and last names from the Instructors table in this format: John Doe (Note: If the instructor first name has a null value, the concatenation of the first and last name will result in a null value.)

A count of the number of courses in the Courses

table The sum of the course units in the Courses table

(Hint: You will need to concatenate the instructor first and last names again in the GROUP BY clause.)

Sort the result set in descending sequence by the total course units for each instructor.

Query –

```
SELECT (FirstName + ' ' + LastName) AS  
    FullName, COUNT(CourseID) AS  
    NumberofCourses, SUM(CourseUnits) AS  
    SumofCourseUnits  
FROM Instructors left JOIN Courses  
ON Instructors.InstructorID =  
Courses.InstructorID GROUP BY (FirstName + '  
+ LastName)  
ORDER BY SumofCourseUnits DESC;
```

COUNT() Function - The COUNT() function returns the number of rows that matches a specified criterion.

SUM() Function - The SUM() function returns the total sum of a numeric column.

```

SELECT (FirstName + ''+LastName) AS FullName,
       COUNT(CourseID) AS NumberOfCourses,
       SUM(CourseUnits) AS SumofCourseUnits
  FROM Instructors left JOIN Courses
    ON Instructors.InstructorID = Courses.InstructorID
 GROUP BY (FirstName + ''+LastName)
 ORDER BY SumofCourseUnits DESC;

```

	FullName	NumberOfCourses	SumofCourseUnits
1	NULL	3	9
2	RachaelAmundsen	2	8
3	TomasGalllegos	2	6
4	GeneGreen	2	6
5	JohnSmith	2	6
6	BillyBrown	2	6
7	DanielConnors	2	6
8	DanielDrew	2	6
9	DerrickThomas	2	5
10	SallyJones	2	5
11	WilliamThomas	1	3
12	BillBlack	1	3
13	JonathanVilma	1	3
14	AngelaWarren	1	3
15	ArnoldPaxton	0	NULL

Query executed successfully.

Comment :Successfully executed a SELECT statement that returns one row for each instructor that has courses using LEFT JOIN, GROUP BY, ORDER BY along with COUNT() and SUM() functions. No of rows : 15 , As, one of the instructor's first name had a null value, the concatenation of the first and last name will resulted in a null value.

12. Write a SELECT statement that answers this question: What is the total number of courses taught by parttime instructors? Return these columns:

The instructor last name and first name from the Instructors table in this format: Doe, John (Note: If the instructor first name has a null value, the concatenation of the first and last name will result in a null value.)

The total number of courses taught for each instructor in the Courses table Use the ROLLUP operator to include a row that gives the grand total.

Query –

```
SELECT LastName + ',' + ' ' + FirstName AS
      FullName, SUM(CourseID) AS TotalCourses
   FROM Instructors JOIN Courses
     ON Instructors.InstructorID =
    Courses.InstructorID WHERE Instructors.Status
= 'P'
 GROUP BY (LastName + ',' + ' ' + FirstName) WITH ROLLUP;
```

ROLLUP operator – Rollup operator helps in providing a summary row for each grouping level. It is an extension of GROUP BY Clause.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the 'MyCollege' database structure, including tables like Instructors, Courses, and Students. The central pane contains a query window with the following SQL code:

```
SELECT LastName + ',' + ' ' + FirstName AS FullName,
      SUM(CourseID) AS TotalCourses
   FROM Instructors JOIN Courses
     ON Instructors.InstructorID =
    Courses.InstructorID WHERE Instructors.Status = 'P'
 GROUP BY (LastName + ',' + ' ' + FirstName) WITH ROLLUP;
```

The Results pane below shows the output of the query. It includes a header row and seven data rows, with the last row being a summary row (Grand Total). The data is as follows:

	FullName	TotalCourses
1	NULL	12
2	Black,Bill	18
3	Thomas,Derrick	38
4	Thomas,William	21
5	Vilma,Jonathan	6
6	Warren,Angela	25
7	NULL	120

At the bottom of the results pane, a message says "Query executed successfully." The status bar at the bottom of the screen shows "Ln 2 Col 38 Ch 38 INS".

Comment :Successfully executed a displayed the total number of courses taught by part-time instructors as 120, using ROLLUP, GROUPBY, SUM() and JOIN. No of rows : 7

13. Write a SELECT statement that returns the same result set as this SELECT statement, but don't use a join. Instead, use a subquery in a WHERE clause that uses the IN keyword.

```
SELECT DISTINCT LastName,  
FirstName FROM Instructors i JOIN  
Courses c  
ON i.InstructorID = c.InstructorID  
ORDER BY LastName, FirstName
```

Query –

```
SELECT DISTINCT LastName,  
FirstName FROM Instructors  
WHERE InstructorID IN  
(SELECT InstructorID  
FROM Courses)  
ORDER BY LastName, FirstName;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The top bar displays the title "InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The left pane is the Object Explorer, showing a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". The right pane contains a query window with the following code:

```
SELECT DISTINCT LastName, FirstName  
FROM Instructors  
WHERE InstructorID IN  
(SELECT InstructorID  
FROM Courses)  
ORDER BY LastName, FirstName;
```

Below the query window is a results grid titled "Results" showing the output of the query:

	LastName	FirstName
1	Amundsen	Rachel
2	Black	Bill
3	Brown	Billy
4	Connors	Daniel
5	Drew	Daniel
6	Gallegos	Tomas
7	Green	Gene
8	Jones	Sally
9	McGregor	NULL
10	Rogers	NULL
11	Smith	John
12	Thomas	Derrick
13	Thomas	William
14	Vilma	Jonathan
15	Warren	Angela

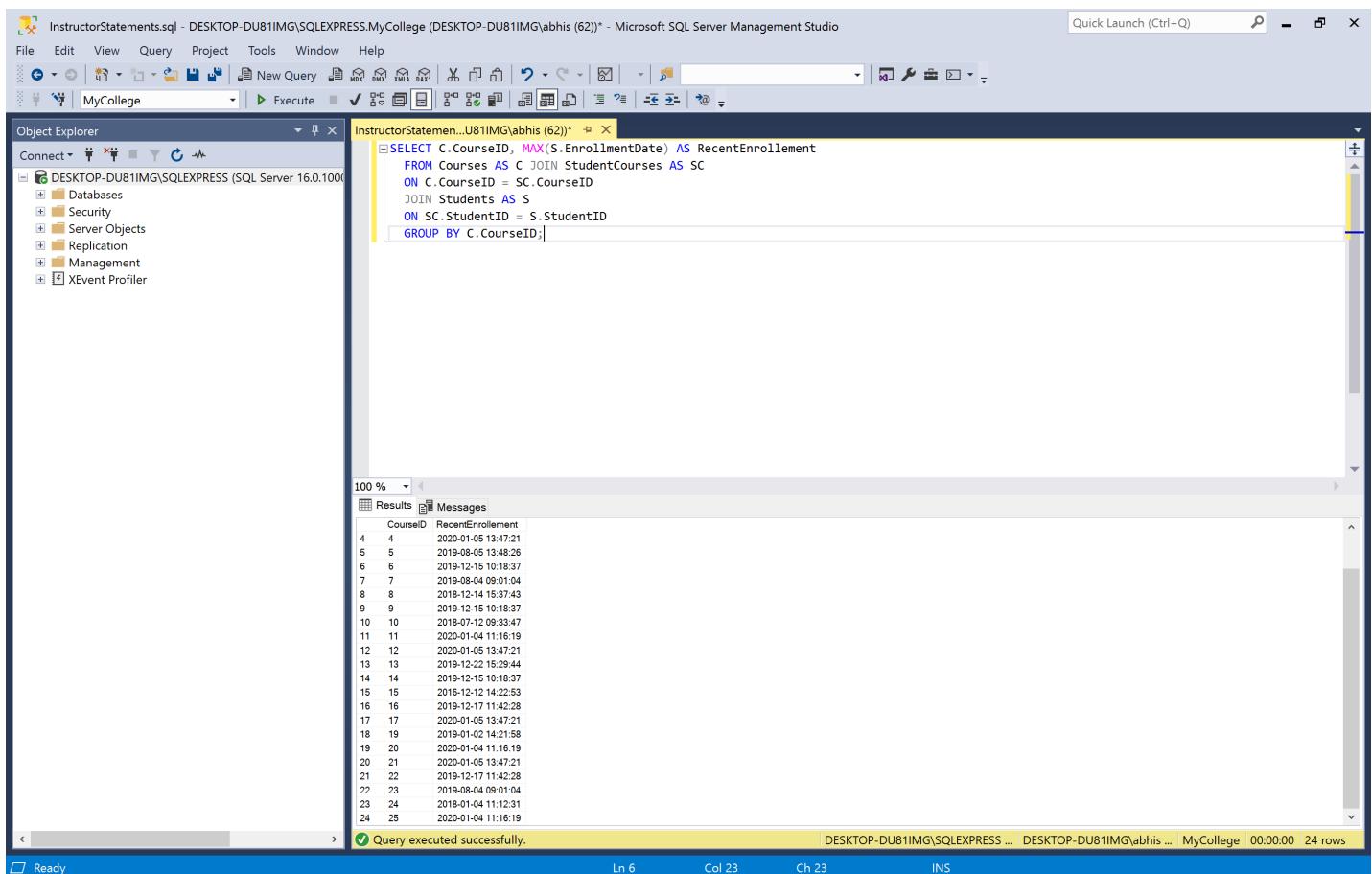
The status bar at the bottom indicates "Query executed successfully." and "15 rows".

Comment : Successfully executed the SELECT without using JOIN, using logic of Subquery via 'IN', No of rows : 15

14. Write a SELECT statement that returns one row for each course with these columns: The CourseID column from the Courses table
 The most recent enrollment date for that course from the Students table
 Change the SELECT statement to a CTE. Then, write a SELECT statement that returns one row per course that shows the CourseDescription for the course and the LastName, FirstName, and EnrollmentDate for the student with the most recent enrollment data.

SELECT Query –

```
SELECT C.CourseID, MAX(S.EnrollmentDate) AS RecentEnrollement
FROM Courses AS C JOIN StudentCourses
AS SC
ON C.CourseID =
SC.CourseIDJOIN Students
AS S
ON SC.StudentID =
S.StudentIDGROUP BY
C.CourseID;
```



The screenshot shows the Microsoft SQL Server Management Studio interface. The query window contains the provided SQL code. The results pane shows a table with two columns: CourseID and RecentEnrollement. The data consists of 24 rows, each mapping a course ID to its latest enrollment date. The results are as follows:

CourseID	RecentEnrollement
4	2020-01-05 13:47:21
5	2019-08-05 13:48:26
6	2019-12-15 10:18:37
7	2019-08-04 09:01:04
8	2018-12-14 15:37:43
9	2019-12-15 10:18:37
10	2018-07-12 09:33:47
11	2020-01-04 11:16:19
12	2020-01-05 13:47:21
13	2019-12-22 15:29:44
14	2019-12-15 10:18:37
15	2016-12-12 14:22:53
16	2019-12-17 11:42:28
17	2020-01-05 13:47:21
18	2019-01-02 14:21:58
19	2020-01-04 11:16:19
20	2020-01-05 13:47:21
21	2019-12-17 11:42:28
22	2019-08-04 09:01:04
23	2018-01-04 11:12:31
24	2020-01-04 11:16:19

Query executed successfully.

Comment : SELECT statement that returns one row for each course. No of rows : 24

CTE Query –

```

WITH CourseSummary
AS (
SELECT C.CourseID, MAX(S.EnrollmentDate) AS RecentEnrollement
FROM Courses AS C JOIN StudentCourses AS SC
ON C.CourseID = SC.CourseID
JOIN Students AS S
ON SC.StudentID = S.StudentID
GROUP BY C.CourseID
)
SELECT CourseDescription, LastName, FirstName,
EnrollmentDate
FROM Courses AS C JOIN StudentCourses AS SC
ON C.CourseID = SC.CourseID
JOIN Students AS S
ON SC.StudentID = S.StudentID
JOIN CourseSummary AS CS
ON C.CourseID = CS.CourseID
AND S.EnrollmentDate = CS.RecentEnrollement;

```

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with a connection to 'DESKTOP-DU81IMG\SQLEXPRESS.MyCollege'. The right pane contains the query window with the CTE code. Below the code, the 'Results' tab is selected, showing the output of the query. The output table has columns: CourseDescription, LastName, FirstName, and EnrollmentDate. The data consists of 24 rows, each representing a course and its most recent enrollment record.

CourseDescription	LastName	FirstName	EnrollmentDate
Intro to Calculus	Manning	Vincent	2018-12-14 15:37:43
Shakespeare	Goodell	Conner	2019-01-02 14:21:58
Beginning Accounting	Johnson	Timothy	2019-08-04 09:01:04
Intro to Marketing	Johnson	Timothy	2019-08-04 09:01:04
Composition	Johnson	Timothy	2019-08-04 09:01:04
Primary Education	Walker	Andrew	2019-08-05 13:48:26
Social Psychology	Walker	Andrew	2019-08-05 13:48:26
Statistical Analysis	Cramden	Walter	2019-12-15 10:18:37
Intermediate Accounting	Cramden	Walter	2019-12-15 10:18:37
Business Writing	Cramden	Walter	2019-12-15 10:18:37
Music Theory	Lisle	Lisa	2019-12-17 11:42:28
Marching Band	Lisle	Lisa	2019-12-17 11:42:28
Abstract Algebra	George	Mona	2019-12-22 15:29:44
Secondary Education	George	Mona	2019-12-22 15:29:44
Human Physiology	Silver	Stanley	2020-01-04 11:16:19
Population and Demographics	Silver	Stanley	2020-01-04 11:16:19
Trigonometry	Silver	Stanley	2020-01-04 11:16:19
Anatomy	Biden	Jones	2020-01-05 13:47:21
Intro to Management	Biden	Jones	2020-01-05 13:47:21
Classic Literature	Biden	Jones	2020-01-05 13:47:21
Creative Writing	Biden	Jones	2020-01-05 13:47:21

Comment : CTE _ STATEMENT that returns one row for each course. No of rows : 24

15. Write an INSERT statement that adds this row to the Departments table:
DepartmentName: History
Code the INSERT statement so SQL Server automatically generates the value for the DepartmentID column.

Query –

```
INSERT Departments
(DepartmentName)
VALUES ('History');
```

```
SELECT *
FROM Departments;
```

InstructorID		The next automatically generated ID
:LastName:		IDBenedict
FirstName:		Susa
Status:		nP
DepartmentChairman	0	
:HireDate:		Today's date
AnnualSalary:		34000.00
DepartmentID:		9
InstructorID:		The next automatically generated ID
Lastname:		Adams
FirstName:		
Status:		nul
DepartmentChairman		IF
:HireDate:	1	
AnnualSalary:		Today's

Write this statement without using a column list.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a database named 'MyCollege' is selected. A query window titled 'InstructorStatements...U81IMG\abhis (62)*' displays the following T-SQL code:

```
INSERT Departments
(DepartmentName)
VALUES ('History');
SELECT *
FROM Departments;
```

The results pane shows a table with two columns: 'DepartmentID' and 'DepartmentName'. The data is as follows:

DepartmentID	DepartmentName
1	Business
2	Education
3	English
4	History
5	Mathematics
6	Music
7	Political Science
8	Science
9	Sociology

At the bottom of the results pane, a message states 'Query executed successfully.' and provides details: DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 9 rows.

Comment :Successfully Inserted DepartmentName value 'History' into Departments table using INSERT statement. No of rows : 9

Comments - SQL Server automatically generates a new value for DepartmentID every time you insert a new value into the DepartmentName column of the Departments table.

16) Write a single INSERT statement that adds these rows to the Instructors table:

Query :

INSERT INTO Instructors VALUES

```
('Benedict','Susan','P',0,GETDATE(),34000.00,10),
```

```
('Adams',NULL,'F',1, GETDATE(),66000.00,10);
```

SELECT * FROM Instructors;

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the MyCollege database with its tables like Instructors, Courses, and Departments. The central pane displays the SQL query:

```
INSERT INTO Instructors VALUES
('Benedict', 'Susan', 'P', 0, GETDATE(), 34000.00, 9),
('Adams', NULL, 'F', 1, GETDATE(), 66000.00, 9);
```

The Messages pane at the bottom right shows the results of the query execution:

```
(2 rows affected)
Completion time: 2023-03-14T16:25:04.5421160+05:30
```

A status bar at the bottom indicates "Query executed successfully." and provides details about the session.

Comment :Wrote a query without using column list. 2 rows affected.

SQLQuery5.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (53))* - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

New Query MDI SSWA DTS Execute

Object Explorer

Connect Databases System Databases Database Snapshots AP Examples MyCollege Database Diagrams Tables System Tables FileTables External Tables Graph Tables dbo.Courses dbo.Departments dbo.Instructors dbo.StudentCourses dbo.Students dbo.Tuition Views External Resources Synonyms Programmability Query Store Service Broker Storage Security ProductOrders Security Server Objects Replication Management XEvent Profiler

SQLQuery5.sql - D...U81IMG\abhis (53)* CreateMyCollege.s...U81IMG\abhis (59)

```
INSERT INTO Instructors VALUES
('Benedict', 'Susan', 'P', 0, GETDATE(), 34000.00, 9),
('Adams', NULL, 'F', 1, GETDATE(), 66000.00, 9);

SELECT * FROM Instructors;
```

Results Messages

InstructorID	LastName	FirstName	Status	DepartmentChairman	HireDate	AnnualSalary	DepartmentID
1	Brown	Billy	F	1	2016-01-10	77500.00	1
2	Thomas	William	P	0	2016-03-30	38500.00	3
3	Amundsen	Rachel	F	1	2016-06-05	79000.00	6
4	Green	Gene	F	0	2016-08-02	75000.00	1
5	McGregor	NULL	F	1	2017-01-03	74000.00	4
6	Paxton	Arnold	P	0	2017-07-15	36000.00	5
7	Rogers	NULL	P	0	2017-10-22	38000.00	1
8	Smith	John	F	1	2018-02-05	73000.00	2
9	Connors	Daniel	F	1	2018-03-04	71500.00	7
10	Jones	Sally	F	1	2018-09-21	74000.00	3
11	Vilma	Jonathan	P	0	2018-11-18	35500.00	1
12	Thomas	Derrick	P	0	2019-01-17	35500.00	5
13	Black	Bill	P	0	2019-04-20	34000.00	2
14	Warren	Angela	P	0	2019-07-14	33000.00	4
15	Drew	Daniel	F	0	2019-08-25	72000.00	3
16	Gallegos	Tomas	F	0	2020-03-23	64000.00	6
17	Benedict	Susan	P	0	2023-03-14	34000.00	9
18	Adams	NULL	F	1	2023-03-14	66000.00	9
19	Benedict	Susan	P	0	2023-03-14	34000.00	9
20	Adams	NULL	F	1	2023-03-14	66000.00	9

Query executed successfully.

DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 | 20 rows

Ready Ln 6 Col 27 Ch 27 INS

Comments : Successfully inserted the 2 rows of data in Instructions table which are numbered 34 and 35 in the InstructorID column.

17. Write an UPDATE statement that modifies the first instructor you added in the above question. This statement should change the AnnualSalary column from 34,000 to 35,000, and it should use the InstructorID column to identify the row.

Query :

UPDATE Instructors

SET AnnualSalary =

350000 WHERE InstructorID

= 20;

SELECT * FROM

Instructors WHERE

InstructorID = 20;

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery5.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (53)) - Microsoft SQL Server Management Studio". The main window has two panes: "Object Explorer" on the left and "SQL Query Editor" on the right. The Object Explorer shows the database structure for "MyCollege", including tables like "Courses", "Departments", "Instructors", etc. The SQL Query Editor contains the following code:

```
UPDATE Instructors
SET AnnualSalary = 350000
WHERE InstructorID = 19;
```

The results pane below the query editor shows the output:

```
(1 row affected)
Completion time: 2023-03-14T16:27:14.3941305+05:30
```

At the bottom of the screen, a status bar indicates "Ready", "Ln 3", "Col 17", "Ch 17", and "INS". A yellow message bar at the bottom says "Query executed successfully."

Comments : Successfully updated the Instructors table with the required value, using UPDATE statement.

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery5.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (53)) - Microsoft SQL Server Management Studio". The menu bar includes File, Edit, View, Query, Project, Tools, Window, and Help. The toolbar has various icons for file operations like New Query, Save, Print, and Execute. The Object Explorer on the left shows the database structure under "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". The "MyCollege" database is expanded, showing tables like "Instructors", "Courses", "Departments", "Students", and "Tuition". The "Results" tab in the center displays the output of a query:

```
SELECT * FROM Instructors  
WHERE InstructorID = 19;
```

InstructorID	LastName	FirstName	Status	DepartmentChairman	HireDate	AnnualSalary	DepartmentID
19	Benedict	Susan	P	0	2023-03-14	350000.00	9

Below the results, a message bar says "Query executed successfully." and shows the session details: DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 1 rows.

Comments : The updated value, which was raised from 34,000 to 35,0000, should be confirmed.
No. of rows 1

18) Write a DELETE statement that deletes the second instructor you added in question.21 This statement should use the InstructorID column to identify the row.

Query :

```
DELETE FROM  
Instructors WHERE  
InstructorID = 20;
```

```
SELECT * FROM  
Instructors WHERE  
InstructorID = 20;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure of 'MyCollege'. It includes nodes for Databases, Tables (with sub-nodes like System Tables, FileTables, External Tables, Graph Tables, and various dbo.* tables), Views, and other system objects. In the center, a query window titled 'SQLQuery5.sql' contains two queries. The first query is a DELETE statement targeting the 'Instructors' table where 'InstructorID = 20'. The second query is a SELECT statement from the same table with the same WHERE clause. Below the queries, the 'Messages' pane shows the execution results: '(0 rows affected)' and 'Completion time: 2023-03-14T16:31:05.2805814+05:30'. At the bottom, a status bar indicates 'Ready', 'Ln 3', 'Col 1', 'Ch 1', and 'INS'.

Comments : Successfully deleted the second instructor, using DELETE statement.

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure of 'MyCollege'. In the center, the 'SQLQuery5.sql' query editor window contains two queries:

```
DELETE FROM Instructors  
WHERE InstructorID = 20;  
  
SELECT * FROM Instructors  
WHERE InstructorID = 20;
```

The status bar at the bottom indicates "Query executed successfully." and "0 rows".

Comments : Instructors table not showing any value, indicating successful deletion of second instructor from Instructors table.

19) Write a DELETE statement that deletes the row in the Departments table that has anID of 9. When you execute this statement, it will produce an error since the department has related rows in the Instructors table. To fix that, precede the DELETE statement with another DELETE statement that deletes all instructors in this department.

Query :

```
DELETE FROM Instructors
WHERE DepartmentID = 9;
```

```
DELETE FROM
Departments WHERE
DepartmentID = 9;
```

SQLQuery5.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (53))* - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

Object Explorer

Connect Databases System Databases Database Snapshots AP Examples MyCollege Database Diagrams Tables System Tables External Tables Graph Tables dbo.Courses dbo.Departments dbo.Instructors dbo.StudentCourses dbo.Students dbo.Tuition Views External Resources Synonyms Programmability Query Store Service Broker Storage Security ProductOrders Security Server Objects Replication Management XEvent Profiler

SQLQuery5.sql - D...U81IMG\abhis (53)* CreateMyCollege.s...U81IMG\abhis (59)

DELETE FROM Departments
WHERE DepartmentID = 9;

Messages

Msg 547, Level 16, State 0, Line 1
The DELETE statement conflicted with the REFERENCE constraint "FK_Instructor_DepartmentID". The conflict occurred in database "MyCollege", table "dbo.Instructors", column 'DepartmentID'.
The statement has been terminated.
Completion time: 2023-03-14T16:37:47.3420898+05:30

Query completed with errors.

Ln 2 Col 24 Ch 24 INS

Comments : Error statement since, the department has related rows in the Instructors table. Due to the department's linked rows in the Instructors table, we are experiencing an error. Prior to deleting them from the Departments dataset, we must first delete those identical needed values from the Instructors table. This is because the Departments and Instructors table and the DepartmentID column have a foreign key-primary key connection.

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery5.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (53)) - Microsoft SQL Server Management Studio". The main window has two tabs: "SQLQuery5.sql - D...U81IMG\abhis (53)*" and "CreateMyCollege.s...U81IMG\abhis (59)". The left pane is the Object Explorer, showing the database structure for "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". The right pane contains the following SQL code:

```
DELETE FROM Instructors
WHERE DepartmentID = 9;

DELETE FROM Departments
WHERE DepartmentID = 9;
```

The "Messages" pane at the bottom shows the execution results:

```
(3 rows affected)
(1 row affected)

Completion time: 2023-03-14T16:38:56.1484141+05:30
```

The status bar at the bottom indicates "Query executed successfully.", "Ln 5", "Col 12", "Ch 12", and "INS".

Comments : Successfully deleted the rows from the Departments table with DepartmentID = '9', using DELETE statement.

20) Write an UPDATE statement that increases the annual salary for all instructors in the Education department by 5%. To do that, join the Departments and Instructors tables and then filter the rows by the department name

Query :

UPDATE Instructors

```
SET AnnualSalary = (AnnualSalary + (0.05*AnnualSalary))
```

```
FROM Instructors JOIN Departments
```

```
ON Instructors.InstructorID =
```

```
Departments.DepartmentID WHERE DepartmentName =
```

```
'Education';
```

```
SELECT *
```

```
FROM Instructors JOIN Departments
```

```
ON Instructors.InstructorID =
```

```
Departments.DepartmentID WHERE DepartmentName =
```

```
'Education';
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including databases, tables, and other objects under the 'MyCollege' database. The central pane displays a query window with the following T-SQL code:

```
UPDATE Instructors
SET AnnualSalary = (AnnualSalary + (0.05*AnnualSalary))
FROM Instructors JOIN Departments
ON Instructors.InstructorID = Departments.DepartmentID
WHERE DepartmentName = 'Education';
```

The status bar at the bottom indicates "Query executed successfully." and shows statistics: Ln 5, Col 36, Ch 36, INS.

Comments : Successfully increased the annual salary for all instructors in the Education department by 5%, using JOIN and UPDATE.

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The menu bar includes File, Edit, View, Query, Project, Tools, Window, and Help. The toolbar has various icons for connecting, executing queries, and managing objects. The Object Explorer on the left shows a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)" with nodes for Databases, Security, Server Objects, Replication, Management, and XEvent Profiler. A query window titled "InstructorStatements...U81IMG\abhis (62)*" contains the following SQL code:

```
SELECT *
FROM Instructors JOIN Departments
ON Instructors.InstructorID = Departments.DepartmentID
WHERE DepartmentName = 'Education';
```

The Results pane displays the output of the query:

InstructorID	LastName	FirstName	Status	DepartmentChairman	HireDate	AnnualSalary	DepartmentID	DepartmentID	DepartmentName	
1	2	Thomas	William	P	0	2016-03-30	40425.00	3	2	Education

A status bar at the bottom indicates "Query executed successfully." and provides details like the server name, session ID, and execution time.

Comments : The updated annual salary of instructors from Education department.

21) Write a DELETE statement that deletes instructors that aren't teaching any courses.
To do that, use a subquery in the WHERE clause.

Query :

```
DELETE FROM Instructors
WHERE InstructorID NOT
IN
(SELECT InstructorID FROM Courses);

SELECT * FROM
Instructors WHERE
InstructorID NOT IN
(SELECT InstructorID FROM Courses);
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "InstructorStatements.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The main window has a toolbar at the top and a menu bar below it. On the left is the Object Explorer pane, which shows a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". Under this, there are nodes for Databases, Security, Server Objects, Replication, Management, and XEvent Profiler. The central pane contains a query editor window titled "InstructorStatements...U81IMG\abhis (62)*". It displays the following T-SQL code:

```
DELETE FROM Instructors
WHERE InstructorID NOT IN
(SELECT InstructorID FROM Courses);
```

Below the query editor is the Messages pane, which shows the output of the query execution:

```
(1 row affected)
Completion time: 2023-03-15T11:51:14.7696631+05:30
```

At the bottom of the screen, the status bar indicates "Ready", "Ln 3", "Col 36", "Ch 36", and "INS". A yellow status bar at the very bottom also says "Query executed successfully." and "0 rows".

Comments : Successfully deleted 3 rows from Instructors

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a database named 'MyCollege' is selected. A query window titled 'InstructorStatements...U81IMG\abhis (62)*' contains the following SQL code:

```
SELECT * FROM Instructors  
WHERE InstructorID NOT IN  
(SELECT InstructorID FROM Courses);
```

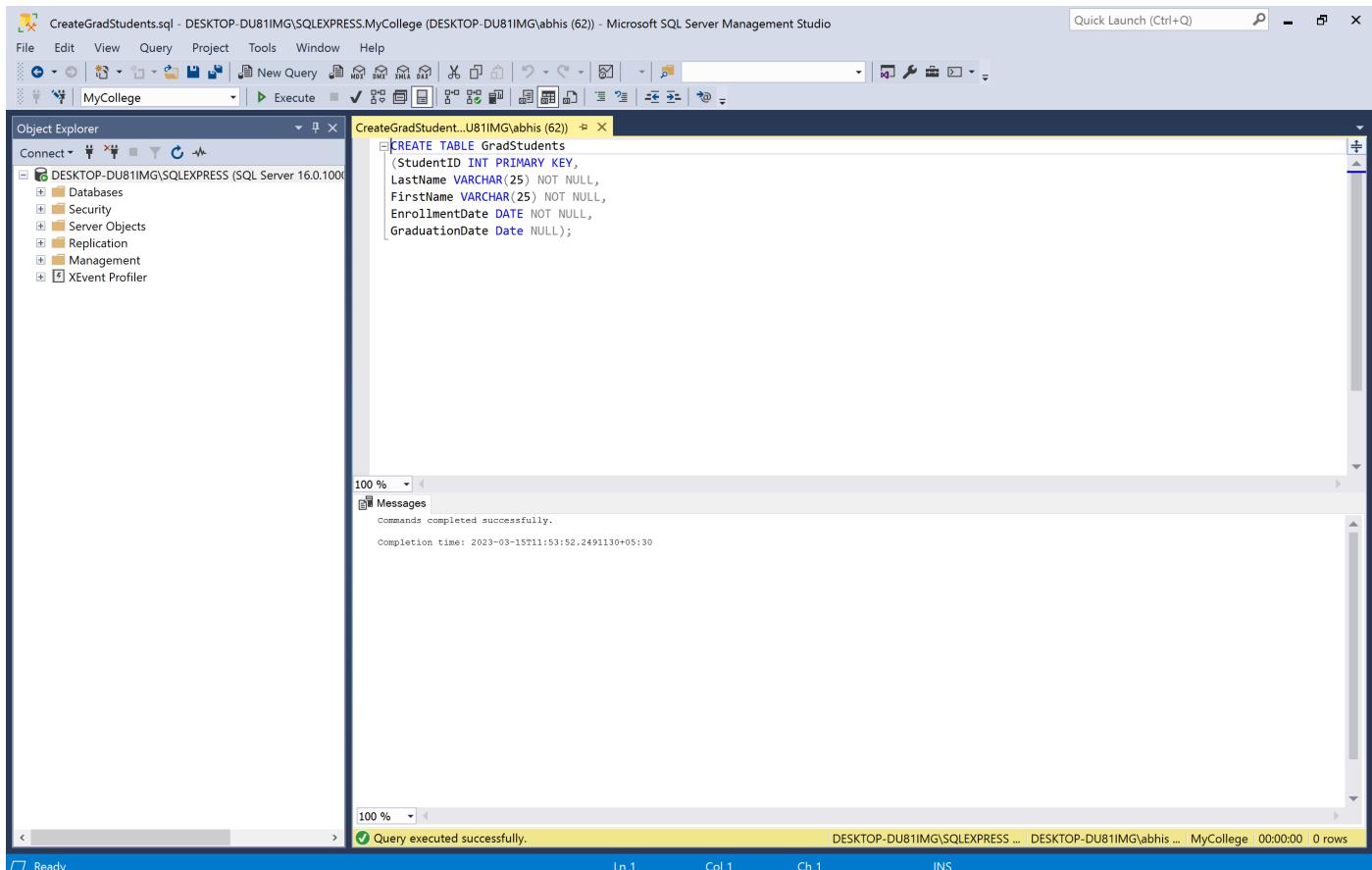
The results pane shows a table with columns: InstructorID, LastName, FirstName, Status, DepartmentChairman, HireDate, AnnualSalary, and DepartmentID. Below the results, a message indicates: "Query executed successfully." The status bar at the bottom right shows: DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 0 rows.

Comments : Successfully deleted instructors that aren't teaching any courses, using DELETE and subquery in the WHERE clause.

22) Open the script named CreateGradStudents.sql that's attached above. Run this file to create a table named GradStudents. This table has the same columns as the Students table, but the StudentID column isn't defined as an identity column.

Query :

```
CREATE TABLE GradStudents
(StudentID INT PRIMARY KEY,
LastName VARCHAR(25) NOT NULL,
FirstName VARCHAR(25) NOT NULL,
EnrollmentDate DATE NOT NULL,
GraduationDate DATE NULL;
```



Comments : Successfully created a table named GradStudents, using the attached file given for this project.

23) Write an INSERT statement that inserts rows from the Students table into the GradStudents table. Include only the rows for students that have graduated, and don't use a column list.

Query :

```
INSERT INTO GradStudents  
SELECT * FROM Students  
WHERE GraduationDate IS NOT NULL;
```

```
SELECT * FROM GradStudents;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "CreateGradStudents.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The menu bar includes File, Edit, View, Query, Project, Tools, Window, Help. The toolbar has various icons for database management. The Object Explorer on the left shows a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". The main query window contains the following SQL code:

```
INSERT INTO GradStudents  
SELECT * FROM Students  
WHERE GraduationDate IS NOT NULL;
```

Below the query window, the Messages pane displays the results of the execution:

```
(7 rows affected)  
Completion time: 2023-03-15T11:56:16.1964832+05:30
```

At the bottom of the screen, the status bar shows "Ready", "Ln 4", "Col 1", "Ch 1", and "INS".

Comments : Successfully inserted rows into the GradStudents table of only students who have graduated, using INSERT statement.

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "CreateGradStudents.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (62)) - Microsoft SQL Server Management Studio". The Object Explorer sidebar shows a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". A query window titled "CreateGradStudent...U81IMG\abhis (62)*" contains the SQL command "SELECT * FROM GradStudents;". The results pane displays a table with 7 rows of data:

StudentID	LastName	FirstName	EnrollmentDate	GraduationDate
1	Howard	Amber	2015-12-18	2019-12-14
2	White	George	2015-12-20	2019-12-14
3	MacNamara	Tony	2015-12-21	2019-05-07
4	Welch	Jonathan	2015-12-21	2019-12-14
5	Price	Rose	2016-01-02	2019-12-14
6	Rodriguez	Jesse	2016-01-03	2019-05-07
7	Smith	Roberta	2016-07-22	2019-12-14

The status bar at the bottom indicates "Ready", "Ln 1", "Col 28", "Ch 28", and "INS". A message bar says "Query executed successfully.".

Comments : Verifying the result, No of rows 7

24) Open the script named CreateMyCollege.sql. Then, run this script. That should restore

the data that's in the database. If an error message is displayed indicating that the database is in use, you'll need to close and restart the Management Studio and then run the script again.

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "CreateMyCollege.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (63)) - Microsoft SQL Server Management Studio". The Object Explorer on the left shows a connection to "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". The main pane displays the following T-SQL script:

```
USE master
GO

IF DB_ID('MyCollege') IS NOT NULL
    DROP DATABASE MyCollege

CREATE DATABASE MyCollege
GO

USE [MyCollege]
GO
/******** Object: Table [dbo].[Courses] Script Date: 10/12/2022 10:15:00 AM *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Courses](
    [CourseID] [int] IDENTITY(1,1) NOT NULL,
    [CourseNumber] [int] NULL,
```

The Messages pane at the bottom shows the results of the execution:

```
(1 row affected)
```

A status bar at the bottom right indicates "Query executed successfully.", "DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:02 0 rows", and "Ln 1 Col 1 Ch 1 INS".

Comments : Successfully restored the data in the database named 'MyCollege' by running the given script.

25) Write a SELECT statement that returns these columns from the Students table:

A column that uses the CONVERT function to return the EnrollmentDate column in this format: MM/DD/YYYY. In other words, use 2-digit months and days and a 4-digit year, and separate each date component with slashes.

A column that uses the CONVERT function to return the EnrollmentDate column with the date, and the hours and minutes on a 12-hour clock with an am/pm indicator.

A column that uses the CONVERT function to return the EnrollmentDate column with just the time in a 24-hour format, including the milliseconds.

A column that uses the CONVERT function to return the EnrollmentDate column with just the month and day.

Query :

```
SELECT  
CONVERT(VARCHAR, EnrollmentDate, 101) AS [Column 1],  
CONVERT(VARCHAR, EnrollmentDate, 100) AS [Column 2],  
CONVERT(VARCHAR, EnrollmentDate, 114) AS [Column 3],  
CONVERT(VARCHAR(5), EnrollmentDate, 101) AS [Column 4]  
FROM Students;
```

The screenshot shows two Microsoft SQL Server Management Studio (SSMS) sessions side-by-side.

Top Window:

- Object Explorer:** Shows the database structure for "DESKTOP-DU81IMG\SQLEXPRESS.MyCollege".
- SQL Query Window:** Contains the following T-SQL code:


```
SELECT
    CONVERT(VARCHAR, EnrollmentDate, 101) AS [Column 1],
    CONVERT(VARCHAR, EnrollmentDate, 100) AS [Column 2],
    CONVERT(VARCHAR, EnrollmentDate, 114) AS [Column 3],
    CONVERT(VARCHAR, 5, EnrollmentDate, 101) AS [Column 4]
FROM Students;
```
- Messages Window:** Displays the execution message: "Commands completed successfully. Completion time: 2023-03-14T17:10:22.9833064+05:30".
- Status Bar:** Shows "Query executed successfully.", "Ln 3", "Col 73", "Ch 73", and "INS".

Bottom Window:

- Object Explorer:** Shows the database structure for "DESKTOP-DU81IMG\abhis (67)".
- SQL Query Window:** Contains the following T-SQL code:


```
CREATE VIEW DepartmentInstructors
AS SELECT d.DepartmentName, i.LastName, i.FirstName, i.Status, i.AnnualSalary
FROM Departments d, Instructors i WHERE d.DepartmentID = i.DepartmentID;
```
- Messages Window:** Displays the execution message: "Commands completed successfully. Completion time: 2023-03-14T17:10:22.9833064+05:30".
- Status Bar:** Shows "Query executed successfully.", "Ln 3", "Col 73", "Ch 73", and "INS".

Comments : Successfully returned 4 columns in the required format as stated in the question statement, using CONVERT() function. No of rows 43

PART C, Advanced SQL Skills (views/stores procedures/ functions /scripts) –

Query :

```
CREATE VIEW DepartmentInstructors
AS SELECT d.DepartmentName, i.LastName, i.FirstName,
i.Status ,i.AnnualSalaryFROM Departments d, Instructors i WHERE
d.DepartmentID= i.DepartmentID;
```

Comments : Successfully created a view named DepartmentInstructors returning the three required columns, using CREATE VIEW statement.

- 2) Write a SELECT statement that returns all the columns from the DepartmentInstructors view that you created in exercise 1.

Query : `SELECT *FROM DepartmentInstructors;`

Comments : Successfully displayed query result output, showing all the columns from the DepartmentInstructors view, using SELECT statement.

SQLQuery2.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (67)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

New Query DMO XML XMLE Execute

Object Explorer

Connect Databases AP Examples Database Snapshots Database Diagrams Tables Views External Resources Synonyms Programmability Query Store Service Broker Storage Security MyCollege ProductOrders Security Server Objects Replication Management XEvent Profiler

SQLQuery2.sql - D...U81IMG\abhis (67)* SQLQuery1.sql - D...U81IMG\abhis (52)*

```
SELECT * FROM DepartmentInstructors
```

Results Messages

	DepartmentName	LastName	FirstName	Status	AnnualSalary
1	Business	Brown	Billy	F	77500.00
2	English	Thomas	William	P	38500.00
3	Science	Amundsen	Rachel	F	79000.00
4	Business	Green	Gene	F	75000.00
5	Mathematics	McGregor	NULL	F	74000.00
6	Music	Paxton	Arnold	P	36000.00
7	Business	Rogers	NULL	P	38000.00
8	Education	Smith	John	F	73000.00
9	Sociology	Connors	Daniel	F	71500.00
10	English	Jones	Sally	F	74000.00
11	Business	Vilma	Jonathan	P	35500.00
12	Music	Thomas	Derrick	P	35500.00
13	Education	Black	Bill	P	34000.00
14	Mathematics	Warren	Angela	P	33000.00
15	English	Drew	Daniel	F	72000.00
16	Science	Gallegos	Tomas	F	64000.00

Query executed successfully.

DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 16 rows

Ready Ln 1 Col 36 Ch 36 INS

3) Return one row for each fulltime instructor in the English department.

Query :

```
SELECT *
FROM DepartmentInstructors
WHERE DepartmentName =
'English' AND Status = 'F';
```

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure, including the MyCollege database which contains tables like ProductOrders, Security, and Replication. In the center, the Results pane shows the output of the executed query:

	DepartmentName	LastName	FirstName	Status	AnnualSalary
1	English	Jones	Sally	F	81400.00
2	English	Drew	Daniel	F	78200.00

At the bottom of the Results pane, a message indicates "Query executed successfully." The status bar at the bottom right shows "Ready", "Ln 4", "Col 18", "Ch 18", and "INS".

Comments: Successfully executed a SELECT statement that return one row for each full-time instructor in English department, total 9 rows of data.

4) Write an UPDATE statement that updates the DepartmentInstructors view you created in exercise 1 so it increases the annual salary for each fulltime instructor inthe English department by 10%. Then, run the SELECT statement you wrote in exercise 2 to be sure this worked correctly.

Query :

```
UPDATE DepartmentInstructors SET AnnualSalary = (AnnualSalary)*(1.1) WHERE
Status = 'F' AND DepartmentName = 'English';
SELECT * FROM DepartmentInstructors
```

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer displays the database structure for 'MyCollege'. In the center, the Results pane shows the output of the executed query. The query updates the 'AnnualSalary' column for full-time instructors ('Status = 'F'') in the 'English' department ('DepartmentName = 'English''). The results show 16 rows affected, with the salary increased by 10%.

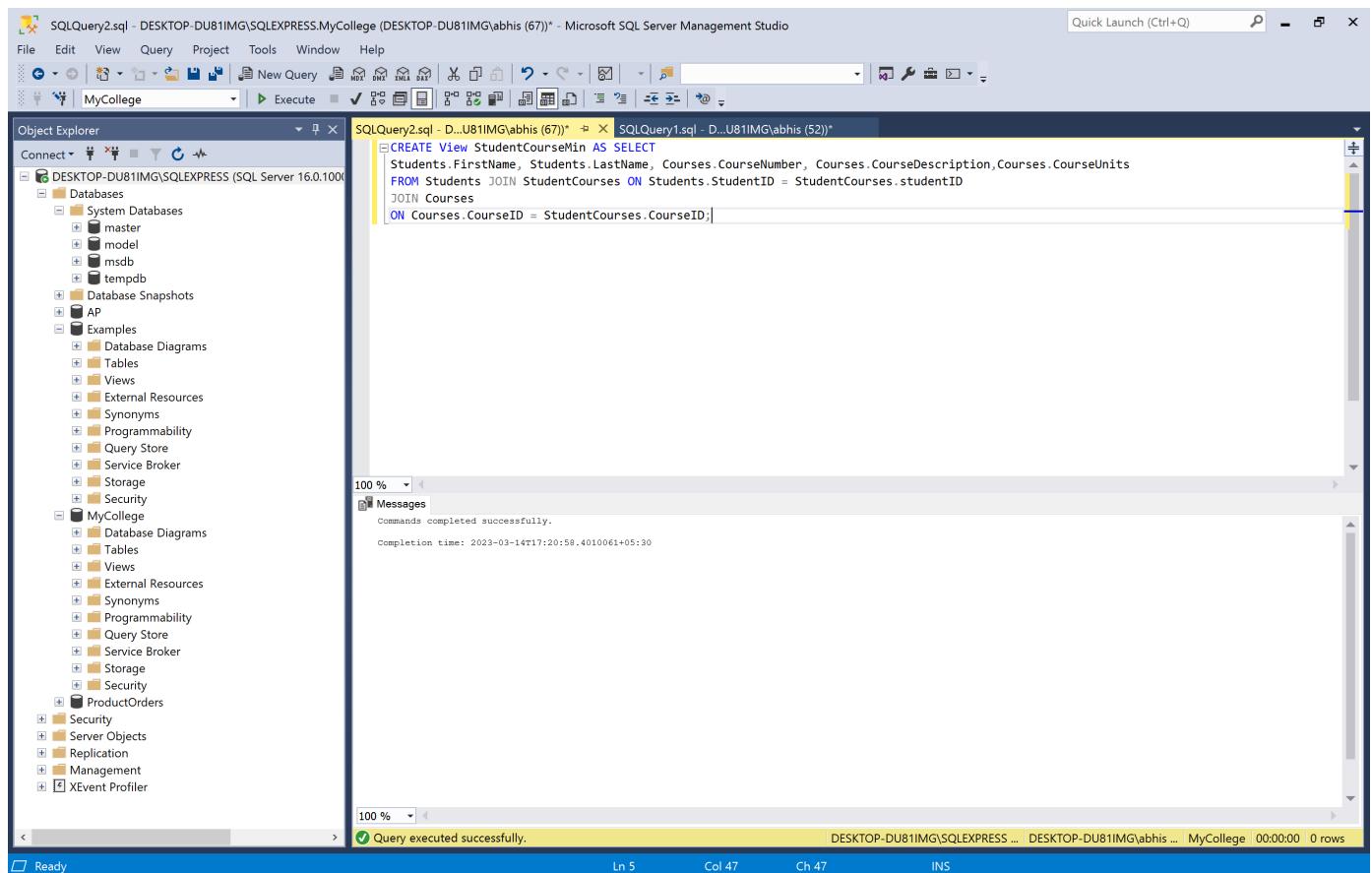
	DepartmentName	LastName	FirstName	Status	AnnualSalary
1	Business	Brown	Billy	F	77500.00
2	English	Thomas	William	P	38500.00
3	Science	Amundsen	Rachel	F	79000.00
4	Business	Creen	Gene	F	75000.00
5	Mathematics	McGregor	NULL	F	74000.00
6	Music	Paxton	Arnold	P	36000.00
7	Business	Rogers	NULL	P	38000.00
8	Education	Smith	John	F	73000.00
9	Sociology	Connors	Daniel	F	71500.00
10	English	Jones	Sally	F	81400.00
11	Business	Vilma	Jonathan	P	35500.00
12	Music	Thomas	Derrick	P	35500.00
13	Education	Black	Bill	P	34000.00
14	Mathematics	Warren	Angela	P	33000.00
15	English	Drew	Daniel	F	79200.00
16	Science	Gallegos	Tomas	F	64000.00

Comments: Successfully updated the DepartmentInstructors view so as to increase the annual salary for each full time instructor in the English department by 10%.

5) Create a view named StudentCoursesMin that returns these columns: the FirstName and LastName from the Students table and the CourseNumber,CourseDescription, and CourseUnits from the Courses

Query :

```
CREATE View StudentCoursesMin AS SELECT
Students.FirstName, Students.LastName, Courses.CourseNumber, Courses.CourseDescri
ption, Courses.CourseUnits
FROM Students JOIN StudentCourses ON Students.StudentID =
StudentCourses.StudentIDJOIN Courses
ON Courses.CourseID = StudentCourses.CourseID
```



Comments: Successfully created a view named StudentCoursesMin with the required columns, using CREATE VIEW statement.

SQLQuery2.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (67)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

New Query New Object Explorer Task List Object Explorer

MyCollege Execute

SQLQuery2.sql - D...U81IMG\abhis (67)* SQLQuery1.sql - D...U81IMG\abhis (52)*

```
SELECT * FROM StudentCourseMin;
```

Object Explorer

Connect DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000) MyCollege

- Databases
 - System Databases
 - master
 - model
 - msdb
 - tempdb
 - Database Snapshots
 - AP
 - Examples
 - Database Diagrams
 - Tables
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Query Store
 - Service Broker
 - Storage
 - Security
 - MyCollege
 - Database Diagrams
 - Tables
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Query Store
 - Service Broker
 - Storage
 - Security
 - ProductOrders
 - Security
 - Server Objects
 - Replication
 - Management
 - XEvent Profiler

Results Messages

	FirstName	LastName	CourseNumber	CourseDescription	CourseUnits
1	Donna	Taylor	96032	Social Media	3
2	Donna	Taylor	81256	Intro to Management	3
3	Donna	Taylor	46972	Biology	4
4	Donna	Taylor	74832	Creative Writing	3
5	Bonnie	Williams	36598	Beginning Accounting	3
6	Bonnie	Williams	24653	Intro to Marketing	3
7	Bonnie	Williams	55783	Shakespeare	3
8	Thomas	Kent	13524	Statistical Analysis	3
9	Thomas	Kent	22679	Intro to Calculus	3
10	Thomas	Kent	96032	Social Media	3
11	Maggie	Kramer	48926	Abstract Algebra	3
12	Maggie	Kramer	84937	Microbiology	4
13	Cameron	Delorean	24653	Intro to Marketing	3
14	Cameron	Delorean	64321	Secondary Education	3
15	Cameron	Delorean	32751	Business Writing	2
16	Roberta	Smith	36598	Beginning Accounting	3
17	Roberta	Smith	24653	Intro to Marketing	3
18	Roberta	Smith	81256	Intro to Management	3
19	Brian	Bonwell	13524	Statistical Analysis	3
20	Brian	Bonwell	22679	Intro to Calculus	3
21	Brian	Bonwell	46972	Biology	4

Query executed successfully.

DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 | 100 rows

Ready

Ln 1 Col 32 Ch 32 INS

Comments: Verifying the StudentCoursesMin and 100 rows affected.

6) Write a script that declares a variable and sets it to the count of all students in the Students table that haven't graduated. If the count is greater than or equal to 100, the script should display a message that says, "The number of undergrad students is greater than or equal to 100". Otherwise, it should say, "The number of undergrad students is less than 100".

Query :

```
DECLARE @DBStatus varchar(20)
set @DBStatus=(SELECT COUNT(*) AS UnderGrad FROM Students WHERE
GraduationDate ISNULL)
if @DBStatus>=100
Print 'The number of undergrad students is greater than or equal
to 100'else
Print 'The number of undergrad students is less than 100'
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the MyCollege database which contains tables like Students, ProductOrders, and others. The central pane displays a query window with the following SQL code:

```
SELECT COUNT(*) AS [Number of UnderGra Students]
FROM Students
WHERE GraduationDate IS NULL;
```

The results pane below shows the output of the query:

Number of UnderGra Students
36

A status bar at the bottom indicates "Query executed successfully." and provides details about the session.

Comments: Number of undergrad students are 36

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery2.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (67)) - Microsoft SQL Server Management Studio". The menu bar includes File, Edit, View, Query, Project, Tools, Window, Help. The toolbar has various icons for file operations like New Query, Save, Print, etc. The Object Explorer on the left shows the database structure under "DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)". The main pane contains two queries: "SQLQuery2.sql" and "SQLQuery1.sql". The code in "SQLQuery2.sql" is:

```
DECLARE @DBStatus varchar(20)
set @DBStatus= (SELECT COUNT(*) AS UnderGrad FROM Students WHERE GraduationDate IS NULL)
if @DBStatus>100
Print 'The number of undergrad students is greater than or equal to 100'
else
Print 'The number of undergrad students is less than 100'
```

The code in "SQLQuery1.sql" is:

```
100 % < > Messages
The number of undergrad students is less than 100
Completion time: 2023-03-14T17:39:01.1824283+05:30
```

The status bar at the bottom indicates "Ready", "Ln 6", "Col 58", "Ch 58", and "INS".

Comments : The number of undergrad students is less than 100

7) Write a script that uses two variables to store (1) the count of all of the instructors in the Instructors table and (2) the average annual salary for those instructors. If the instructor count is greater than or equal to 10, the script should print a message that displays the values of both variables. Otherwise, the script should print a message that says, "The number of fulltime instructors is less than 10".

Query :

```
DECLARE @count
varchar(20) DECLARE @avg
varchar(20)
set @count=(SELECT COUNT(*) FROM Instructors)
set @avg=(SELECT avg(AnnualSalary) FROM
Instructors) if @count>=10
Print CONCAT('The number of Instructors is:', @count, CHAR(10), 'and their
average salary is:', @avg)
else
Print 'The number of fulltime instructors is less than 10'
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases such as DESKTOP-DU81IMG\SQLEXPRESS and MyCollege. The central SQL Query window contains the T-SQL script provided in the question. When executed, the script prints the count of instructors (16) and their average salary (57818.75). The status bar at the bottom of the window indicates 'Query executed successfully.'

Comments : The number of Instructors is 16 and their average salary is 57818.75

8) Write a script that attempts to delete the department with the name 'Sociology' from the Departments table. If the delete is successful, the script should display this message:

SUCCESS: Record was deleted.

If the delete is unsuccessful, the script should display a message something like this:

FAILURE: Record was not deleted.

Error 547: The DELETE statement conflicted with the REFERENCE constraint "FK__Instructors__DepartmentID". The conflict occurred in database "MurachCollege", table "dbo.Instructors", column 'DepartmentID'.

Query :

```
BEGIN TRY
DELETE FROM Departments WHERE DepartmentName='Sociology';
PRINT 'SUCCESS: Record was deleted.';
END TRY
BEGIN CATCH
PRINT 'FAILURE: Record was not deleted.';
PRINT 'Error ' + CONVERT(varchar, ERROR_NUMBER(), 1)
+ ':' +
ERROR_MESSAGE(); END
CATCH;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, there are two databases listed: 'DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)' and 'MyCollege'. The 'MyCollege' database is expanded, showing its tables, views, and other objects. In the center pane, there is a 'SQLQuery2.sql - D...U81IMG\abhis (67)*' window containing the SQL script provided above. Below it is another window titled 'SQLQuery1.sql - D...U81IMG\abhis (52)*' which contains a different set of SQL statements. At the bottom of the screen, the status bar displays 'Query executed successfully.' and other system information.

Comments: Sociology department name was successfully deleted with the help of aforementioned query.

9) Write a script that creates and calls a function named fnStudentUnits that calculates the total course units of a student in the StudentCourses table. To do that, this function should accept one parameter for the student ID, and it should return the sum of the course units for the student.

Query :

```

CREATE FUNCTION fnStudentUnits (@StudentID
INT)
RETURNS
INTAS

BEGIN
DECLARE @SumOfCourseUnits
INTSET @SumOfCourseUnits =
0

SELECT @SumOfCourseUnits =
SUM(CourseUnits) FROM StudentCourses AS
SC
JOIN Students AS S
ON S.StudentID =
SC.StudentID JOIN Courses C
ON C.CourseID = SC.CourseID

RETURN
@SumOfCourseUnitsEND
GO

```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases like master, model, msdb, tempdb, and MyCollege. The center pane displays the T-SQL code for creating the function:

```

CREATE FUNCTION fnStudentUnits
(@StudentID INT)
RETURNS INT
AS
BEGIN
DECLARE @SumOfCourseUnits INT
SET @SumOfCourseUnits = 0

SELECT @SumOfCourseUnits = SUM(CourseUnits)
FROM StudentCourses AS SC
JOIN Students AS S
ON S.StudentID = SC.StudentID
JOIN Courses C
ON C.CourseID = SC.CourseID

RETURN @SumOfCourseUnits
END
GO

```

The status bar at the bottom indicates "Query executed successfully." and "Completion time: 2023-03-14T17:46:47.1204471+05:30".

Comments : Successfully wrote the query calls a function named fnStudentUnits that calculates the total course units of a student in the StudentCourses table. To do that, this function should accept one parameter for the student ID, and it should return the sum of the course units for the student.

10)The SELECT statement that calls the function should return the StudentID from theStudentCourses table, the CourseNumber and CourseUnits from the Courses table, and the value return by the fnStudentUnits function for that student.

Query:

```
SELECT SC.StudentID, C.CourseID, C.CourseUnits,
dbo.fnStudentUnits(StudentID) AS 'TotalUnits'
FROM StudentCourses AS SC
JOIN Courses AS C
ON C.CourseID = SC.CourseID
```

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure, including the 'MyCollege' database and its tables like 'StudentCourses' and 'Courses'. In the center, the 'SQLQuery1.sql' window contains the provided T-SQL query. Below it, the 'Results' pane shows the output of the query, which is a table with four columns: StudentID, CourseID, CourseUnits, and TotalUnits. The data consists of 13 rows, with the first row highlighted. At the bottom of the results pane, a message indicates 'Query executed successfully.' The status bar at the bottom right shows the session details: DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 | 100 rows.

StudentID	CourseID	CourseUnits	TotalUnits
1	5	10	298
2	5	12	298
3	5	15	298
4	5	21	298
5	8	1	298
6	8	7	298
7	8	19	298
8	9	6	298
9	9	8	298
10	9	10	298
11	10	2	298
12	10	24	298
13	11	7	298

Comments: Successfully wrote the query that calls the function to return the StudentID from the StudentCourses table, the CourseNumber and CourseUnits from the Courses table, and the value return by the fnStudentUnits function for that student.

11. Write a script that creates and calls a stored procedure named spUpdateInstructor that updates the AnnualSalary column in the Instructors table. This procedure should have one parameter for the instructor ID and another for the annual salary. If the value for the AnnualSalary column is a negative number, the stored procedure should raise an error that indicates that the value for this column must be a positive number. Code at least two EXEC statements that test this procedure.

Query:

```
CREATE PROCEDURE spUpdateInstructor
    @InstructorID INT,
    @AnnualSalary DECIMAL(10, 2)
AS
BEGIN
    -- Check if AnnualSalary is a positive number
    IF @AnnualSalary < 0
        BEGIN
            -- Raise an error if AnnualSalary is negative
            RAISERROR('The value for AnnualSalary must be a positive number.', 16, 1)
            RETURN
        END
    -- Update the AnnualSalary for the specified InstructorID
    UPDATE Instructors SET AnnualSalary = @AnnualSalary WHERE InstructorID = @InstructorID
END
GO

-- Call the stored procedure with a valid AnnualSalary value
EXEC spUpdateInstructor @InstructorID = 1, @AnnualSalary = 50000.00
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases, including 'MyCollege'. The central pane displays a script named 'SQLQuery1.sql' containing a stored procedure definition:

```
CREATE PROCEDURE spUpdateInstructor
    @InstructorID INT,
    @AnnualSalary DECIMAL(10, 2)
AS
BEGIN
    -- Check if AnnualSalary is a positive number
    IF @AnnualSalary < 0
        BEGIN
            -- Raise an error if AnnualSalary is negative
            RAISERROR('The value for AnnualSalary must be a positive number.', 16, 1)
            RETURN
        END
    -- Update the AnnualSalary for the specified InstructorID
    UPDATE Instructors SET AnnualSalary = @AnnualSalary WHERE InstructorID = @InstructorID
END
GO
-- Call the stored procedure with a valid AnnualSalary value
EXEC spUpdateInstructor @InstructorID = 1, @AnnualSalary = 50000.
-- Call the stored procedure with a valid AnnualSalary value
EXEC spUpdateInstructor @InstructorID = 2, @AnnualSalary = -10000.00
```

The status bar at the bottom indicates 'Query executed successfully.' and shows the completion time as 2023-04-09T05:58:19.0293619+05:30.

Comments : For the first exec call it's valid and update the record

-- Call the stored procedure with an invalid AnnualSalary value
EXEC spUpdateInstructor @InstructorID = 2, @AnnualSalary = -10000.00

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases, tables, and other objects under the 'MyCollege' database. The central pane displays a script named 'SQLQuery1.sql' containing the creation of a stored procedure 'spUpdateInstructor' and its execution with invalid parameters. The 'Messages' pane at the bottom shows an error message: 'Msg 50000, Level 16, State 1, Procedure spUpdateInstructor, Line 10 [Batch Start Line 18] The value for AnnualSalary must be a positive number.' Below the messages, it says 'Completion time: 2023-04-09T06:02:29.217297+05:30'. The status bar at the bottom indicates 'Ready', 'Ln 20', 'Col 69', 'Ch 69', and 'INS'.

```
CREATE PROCEDURE spUpdateInstructor
@InstructorID INT,
@AnnualSalary DECIMAL(10, 2)
AS
BEGIN
    -- Check if AnnualSalary is a positive number
    IF @AnnualSalary < 0
    BEGIN
        -- Raise an error if AnnualSalary is negative
        RAISERROR('The value for AnnualSalary must be a positive number.', 16, 1)
        RETURN
    END
    -- Update the AnnualSalary for the specified InstructorID
    UPDATE Instructors SET AnnualSalary = @AnnualSalary WHERE InstructorID = @InstructorID
END
GO
-- Call the stored procedure with a valid AnnualSalary value
EXEC spUpdateInstructor @InstructorID = 1, @AnnualSalary = 50000.
-- Call the stored procedure with a valid AnnualSalary value
EXEC spUpdateInstructor @InstructorID = 2, @AnnualSalary = -10000.00
```

Comments : Throws error as salary is less than 0

12. Create a trigger named Instructors_UPDATE that checks the new value for the AnnualSalary column of the Instructors table. This trigger should raise an appropriate error if the annual salary is greater than 120,000 or less than 0. If the new annual salary is between 0 and 12,000, this trigger should modify the new annual salary by multiplying it by 12. That way, a monthly salary of 5,000 becomes an annual salary of 60,000. Test this trigger with an appropriate UPDATE statement.

Query:

```
CREATE TRIGGER Instructors_UPDATE ON Instructors
AFTER UPDATE
AS
BEGIN
    DECLARE @InstructorID INT
    DECLARE @NewAnnualSalary DECIMAL(10, 2)

    SELECT @InstructorID = InstructorID, @NewAnnualSalary = AnnualSalary
    FROM inserted

    IF @NewAnnualSalary > 120000
    BEGIN
        RAISERROR('Annual salary cannot be greater than 120,000.', 16, 1)
        ROLLBACK
    END

    IF @NewAnnualSalary < 0
    BEGIN
        RAISERROR('Annual salary cannot be less than 0.', 16, 1)
        ROLLBACK
    END

    IF @NewAnnualSalary > 0 AND @NewAnnualSalary < 12000
    BEGIN
        UPDATE Instructors
        SET AnnualSalary = @NewAnnualSalary * 12
        WHERE InstructorID = @InstructorID
    END
END
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases like DESKTOP-DU81IMG\SQLEXPRESS, MyCollege, and MySchool. The central pane displays a T-SQL script named SQLQuery2.sql, which contains a stored procedure. The script includes logic to validate salary values and update the salary of instructors. The status bar at the bottom indicates the command was executed successfully.

```
SQLQuery2.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (61))* - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Quick Launch (Ctrl+Q) X
Object Explorer
Connect ▾
DESKTOP-DU81IMG\SQLEXPRESS (SQL Server 16.0.1000)
Databases
System Databases
Database Snapshots
AP
Examples
MyCollege
Database Diagrams
Tables
Views
External Resources
Synonyms
Programmability
Query Store
Service Broker
Storage
Security
MySchool
ProductOrders
Security
Server Objects
Replication
Management
XEvent Profiler
SQLQuery1.sql - D...U81IMG\abhis (56)*
DECLARE @NewAnnualSalary DECIMAL(10, 2)
SELECT @InstructorID = InstructorID, @NewAnnualSalary = AnnualSalary
FROM inserted
IF @NewAnnualSalary > 120000
BEGIN
    RAISERROR('Annual salary cannot be greater than 120,000.', 16, 1)
    ROLLBACK
END
IF @NewAnnualSalary < 0
BEGIN
    RAISERROR('Annual salary cannot be less than 0.', 16, 1)
    ROLLBACK
END
IF @NewAnnualSalary > 0 AND @NewAnnualSalary < 12000
BEGIN
    UPDATE Instructors
    SET AnnualSalary = @NewAnnualSalary * 12
    WHERE InstructorID = @InstructorID
END
END
100 %
Messages
Commands completed successfully.
Completion time: 2023-04-09T06:17:01.5781886+05:30
100 %
> ✓ Query executed successfully.
DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 0 rows
Ln 17 Col 13 Ch 10 INS
Ready
```

Comments : Commands completed successfully

Test the trigger by update statements,

-- update statement to test ---

-- statement 1 ---

UPDATE Instructors

SET AnnualSalary = 5000

WHERE InstructorID = 1

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure, including the MyCollege database which contains tables like Instructors, Students, and Courses. The central pane shows two queries. The top query is a trigger definition for the Instructors table:

```
CREATE TRIGGER trg_Instructor_Salary
ON Instructors
FOR UPDATE
AS
BEGIN
    IF @NewAnnualSalary > 120000
        RAISERROR('Annual salary cannot be greater than 120,000.', 16, 1)
        ROLLBACK
    ELSE IF @NewAnnualSalary < 0
        RAISERROR('Annual salary cannot be less than 0.', 16, 1)
        ROLLBACK
    ELSE IF @NewAnnualSalary > 0 AND @NewAnnualSalary < 12000
        BEGIN
            UPDATE Instructors
                SET AnnualSalary = @NewAnnualSalary * 12
            WHERE InstructorID = @InstructorID
        END
END
```

The bottom query is the test update statement:

```
-- update statement to test ---
-- statement 1 ---
UPDATE Instructors
SET AnnualSalary = 5000
WHERE InstructorID = 1
```

The Messages pane at the bottom shows the execution results:

- (1 row affected)
- (1 row affected)

Completion time: 2023-04-09T06:20:32.1903153+05:30

Comments : This is valid update so it will perform the update operation

-- statement 2 ---

UPDATE Instructors

SET AnnualSalary = 500000

WHERE InstructorID = 1

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure of 'MyCollege'. The main window contains two queries:

```
RAISERROR('Annual salary cannot be less than 0.', 16, 1)
ROLLBACK
END
IF @NewAnnualSalary > 0 AND @NewAnnualSalary < 12000
BEGIN
    UPDATE Instructors
        SET AnnualSalary = @NewAnnualSalary * 12
        WHERE InstructorID = @InstructorID
    END
END
-- update statement to test ---
-- statement 1 ---
UPDATE Instructors
SET AnnualSalary = 5000
WHERE InstructorID = 1

-- statement 2 ---
UPDATE Instructors
SET AnnualSalary = 500000
WHERE InstructorID = 1
```

The second update statement is highlighted in blue. The Messages pane at the bottom shows the following error output:

```
Msg 50000, Level 16, State 1, Procedure Instructors_UPDATE, Line 11 [Batch Start Line 31]
Annual salary cannot be greater than 120,000.
Msg 3609, Level 16, State 1, Line 33
The transaction ended in the trigger. The batch has been aborted.
```

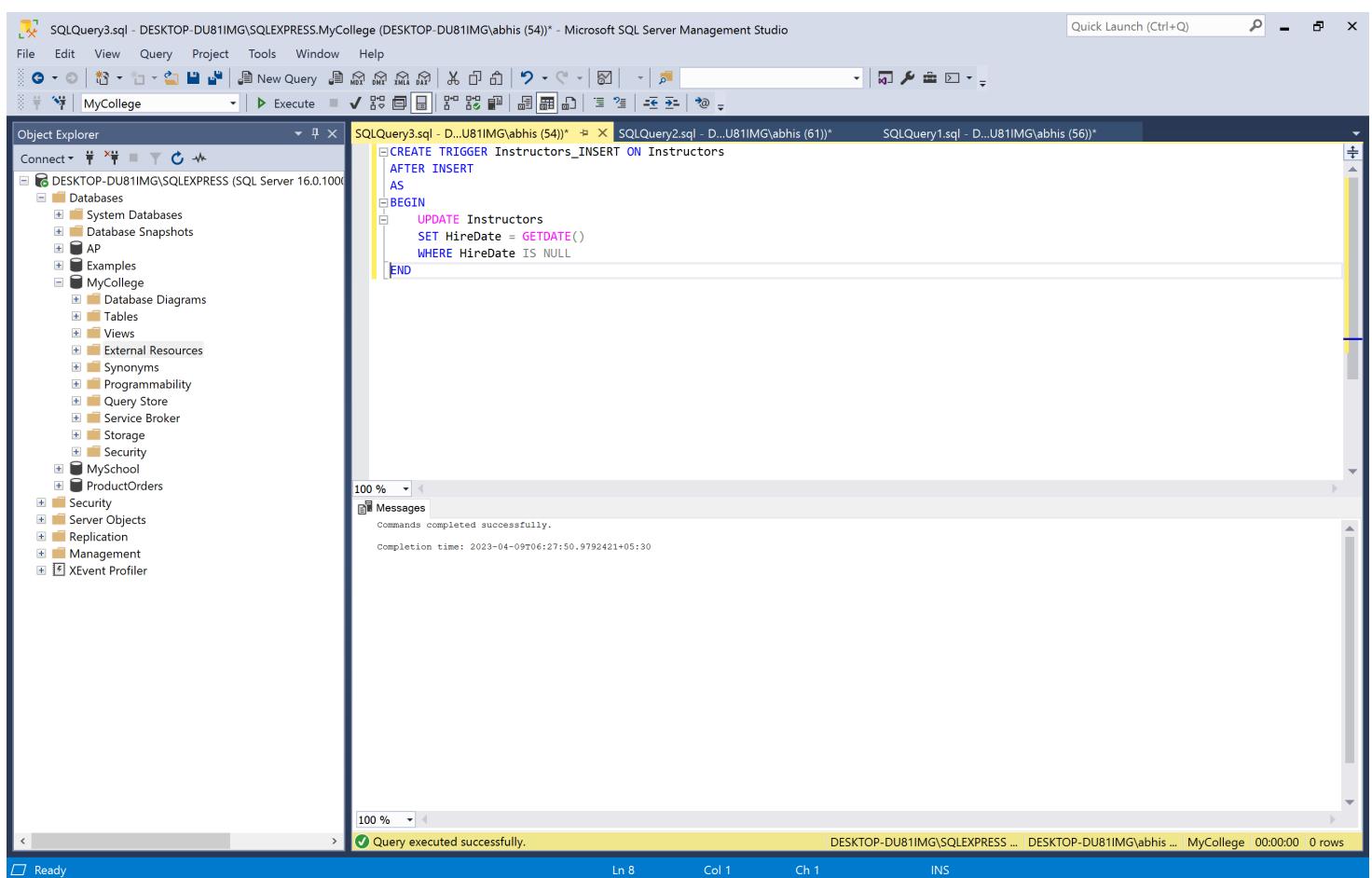
The status bar at the bottom indicates "Query completed with errors." and "0 rows".

Comment : As the Annual Salary is more than 120,000 it throws error and rollbacks the transaction

13.Create a trigger named Instructors_INSERT that inserts the current date for the HireDate column of the Instructors table if the value for that column is null. Test this trigger with an appropriate INSERT statement.

Query:

```
CREATE TRIGGER Instructors_INSERT ON Instructors
AFTER INSERT
AS
BEGIN
    UPDATE Instructors
    SET HireDate = GETDATE()
    WHERE HireDate IS NULL
END
```



Comments : Trigger created successfully

Test the Trigger:

--Testing for the trigger--

-- Statement 1 with empty date ---

```
INSERT INTO Instructors (FirstName, LastName, AnnualSalary, DepartmentID, Status, DepartmentChairman)
VALUES ('Indrani', 'Bhunia', 60000.00, 3, 'F', 1)
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including the MyCollege database which contains tables like Instructors, AP, Examples, and Security. The central pane displays three open queries: SQLQuery3.sql, SQLQuery2.sql, and SQLQuery1.sql. SQLQuery3.sql contains the creation of a trigger named Instructors_INSERT that updates the HireDate field to the current date for any new rows inserted into the Instructors table where the HireDate is null. Below the trigger definition, a comment block -- Testing for the trigger-- is followed by a statement -- Statement 1 with empty date --- and an INSERT INTO command with specific values. The Messages pane at the bottom shows the execution results: (1 row affected) and (1 row affected), indicating the trigger fired once for each row inserted. The status bar at the bottom right shows the query executed successfully, the completion time, and the number of rows affected.

Comments : Inserting Row

SQLQuery3.sql - DESKTOP-DU81IMG\SQLEXPRESS.MyCollege (DESKTOP-DU81IMG\abhis (54)) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

Quick Launch (Ctrl+Q)

Object Explorer

Connect ▾ MyCollege Execute

SQLQuery3.sql - D...U81IMG\abhis (54)* | X SQLQuery2.sql - D...U81IMG\abhis (61)* | SQLQuery1.sql - D...U81IMG\abhis (56)*

```

CREATE TRIGGER Instructors_INSERT ON Instructors
AFTER INSERT
AS
BEGIN
    UPDATE Instructors
    SET HireDate = GETDATE()
    WHERE HireDate IS NULL
END

--Testing for the trigger--
-- Statement 1 with empty date ---
INSERT INTO Instructors (FirstName, LastName, AnnualSalary, DepartmentID, Status, DepartmentChairman)
VALUES ('Indrani', 'Bhunia', 60000.00, 3, 'f', 1)

--Check statement 1--
SELECT *FROM Instructors where DepartmentID=3;

```

Results Messages

InstructorID	LastName	FirstName	Status	DepartmentChairman	HireDate	AnnualSalary	DepartmentID	
1	2	Thomas	William	P	0	2016-03-30	38500.00	3
2	10	Jones	Sally	F	1	2018-09-21	74000.00	3
3	15	Drew	Daniel	F	0	2019-08-25	72000.00	3
4	18	Bhunia	Indrani	f	1	2023-04-09	60000.00	3

Query executed successfully.

DESKTOP-DU81IMG\SQLEXPRESS ... DESKTOP-DU81IMG\abhis ... MyCollege 00:00:00 4 rows

Ready Ln 17 Col 47 Ch 47 INS

Comments :The select query checks for Indrani Bhunia whose hiredate is current date as it wasn't set during insertion

-- Statement 2 with non empty date ---

```
INSERT INTO Instructors (FirstName, LastName, AnnualSalary, DepartmentID, Status, DepartmentChairman, HireDate)
VALUES ('Indrani2', 'Bhunia2', 60000.00, 3, 'F', 1, '2016-03-30')
```

-- Check statement 2 ---

```
select * from Instructors where DepartmentID=3 and FirstName='Indrani2';
```

```
-- Check statement 1 --
SELECT * FROM Instructors where DepartmentID=3;

-- Insert statement
INSERT INTO Instructors (FirstName, LastName, AnnualSalary, DepartmentID, Status, DepartmentChairman, HireDate)
VALUES ('Indrani2', 'Bhunia2', 60000.00, 3, 'F', 1, '2016-03-30');

-- Statement 2 with non empty date ---
-- Insert statement
INSERT INTO Instructors (FirstName, LastName, AnnualSalary, DepartmentID, Status, DepartmentChairman, HireDate)
VALUES ('Indrani2', 'Bhunia2', 60000.00, 3, 'F', 1, '2016-03-30');
-- Check statement 2 ---
select * from Instructors where DepartmentID=3 and FirstName='Indrani2';
```

InstructorID	LastName	FirstName	Status	DepartmentChairman	HireDate	AnnualSalary	DepartmentID
1	Bhunia2	Indrani2	F	1	2016-03-30	60000.00	3

Query executed successfully.

Comments : Testing statement2 where the HireDate is given while insertion

14. Write a script that includes two SQL statements coded as a transaction to delete the row with a student ID of 10 from the Students table. To do this, you must first delete all courses for that student from the StudentCourses table. If these statements execute successfully, commit the changes. Otherwise, roll back the changes.

Query:

```
BEGIN TRANSACTION  
BEGIN TRY  
    DELETE FROM StudentCourses WHERE StudentID = 10  
    DELETE FROM Students WHERE StudentID = 10  
    COMMIT TRANSACTION  
END TRY  
BEGIN CATCH  
    ROLLBACK TRANSACTION  
END CATCH
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including the MyCollege database which contains tables like Students and StudentCourses. The central pane displays the SQL code for the transaction script. The Messages pane at the bottom shows the execution results: '(2 rows affected)' for the first delete statement and '(1 row affected)' for the second. The status bar at the bottom indicates 'Query executed successfully.' and provides other session details.

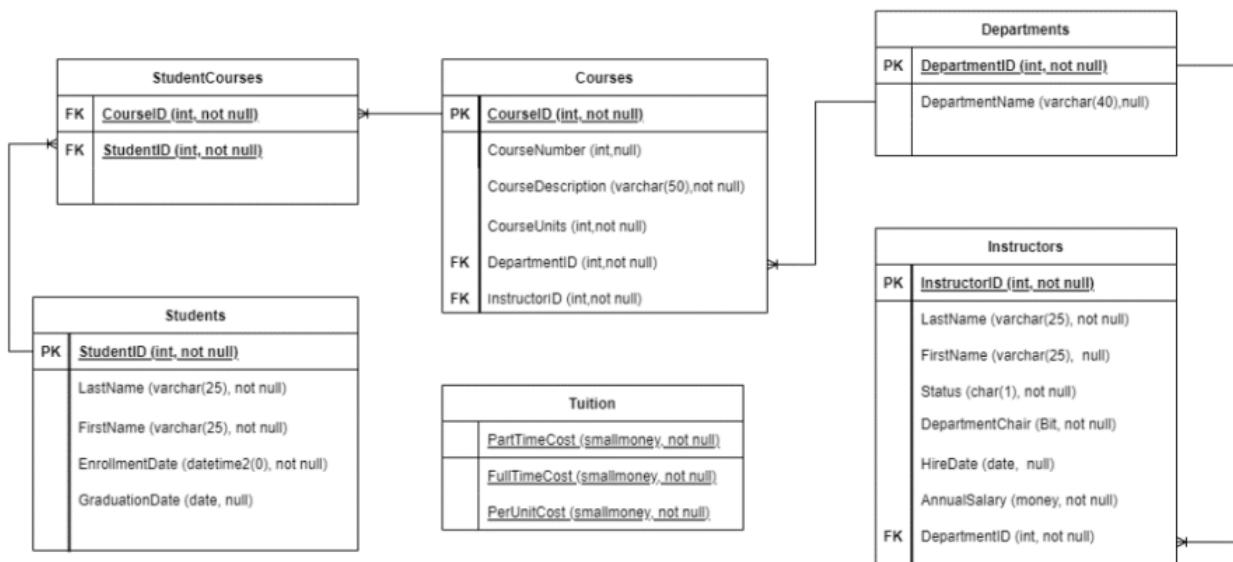
```
BEGIN TRANSACTION  
BEGIN TRY  
    DELETE FROM StudentCourses WHERE StudentID = 10  
    DELETE FROM Students WHERE StudentID = 10  
    COMMIT TRANSACTION  
END TRY  
BEGIN CATCH  
    ROLLBACK TRANSACTION  
END CATCH
```

Comments : Executed transaction successfully

Part D :

Database Design

1.Create a database diagram that shows the relationships between the tables in the MyCollege database. You can use any digital tool to draw your diagrams. "Resources and Links" tab on this Blackboard page lists several tools for drawing E/R diagrams.

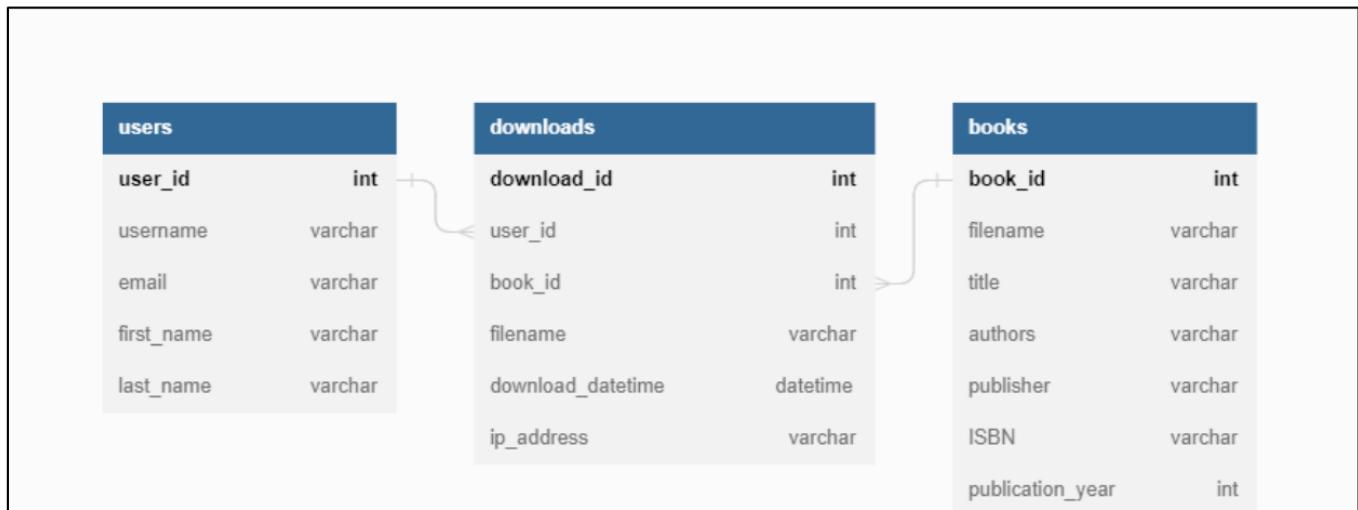


Comment : ERD for college database

Explanation : The MyCollege Database's schema includes six tables: Departments, Courses, StudentCourses, Instructors, Students, and Tuition. Table StudentCourses has two foreign keys: CourseID and StudentID. Table Courses has a primary key of CourseID and two foreign keys: DepartmentID and InstructorID. Table Instructors has a primary key of InstructorID and a foreign key of DepartmentID. Table Students has a primary key of StudentID. The StudentCourses table acts as a linking table because there is a many-to-many relationship between Students and Courses. The Tuition table is an independent object with no relationships to other tables. The Departments, Instructors, and Courses tables are connected with one-to-many relationships. There is a one-to-many relationship between Departments and Instructors. The above diagram illustrates these connections.

2. Design a database diagram for a database that stores information about the downloads that users make from a book website.

- 1. Each user must have a user name, email address, first name, and last name.**
- 2. Each user can have zero or more downloads.**
- 3. Each download must have a filename, download date/time, the IP of the downloading site.**
- 4. Each book can be related to zero or more downloads.**
- 5. Each book must have a filename, title, authors, publisher, ISBN, publisher, year.**



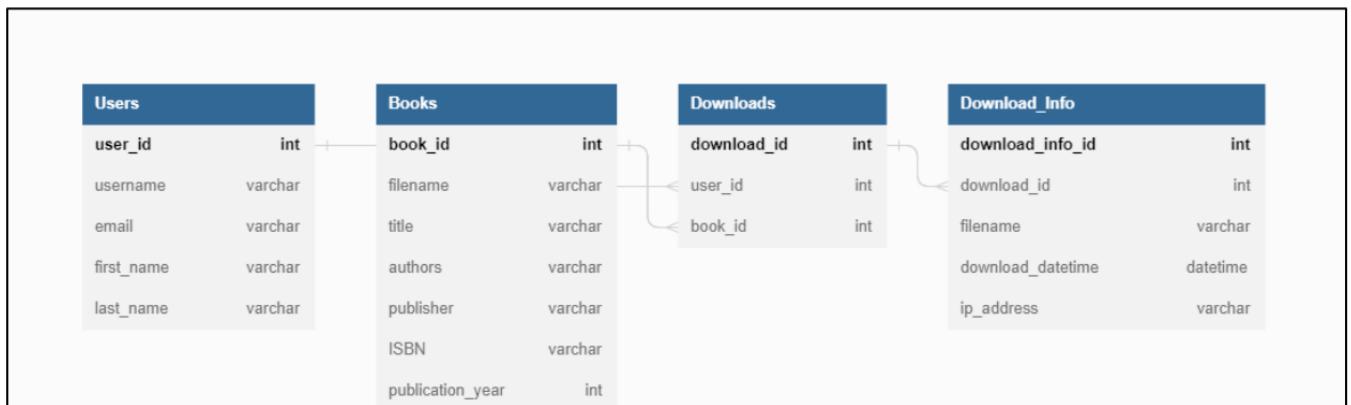
Explanation: This diagram includes three tables: Book, Download, and User. The Book table includes the information about the book, with a unique BookID, filename, title, authors, publisher, ISBN, and year. The User table includes information about the user, with a unique UserID, username, email address, first name, and last name. The Download table is a many-to-many relationship table that connects the Book and User tables. It includes a unique DownloadID, the Book filename that was downloaded (foreign key to Book table), the UserID of the user who downloaded the book (foreign key to User table), the date and time of the download, and the IP address of the site where the download originated. This database diagram can store information about multiple downloads made by different users for different books. It allows for easy tracking of the books that are downloaded, when they were downloaded, and by whom.

3. Analyze your diagram and normalize it to its third-normal form.

The before schema has three tables: Users, Downloads, and Books. The Users table contains information about users such as user_id, username, email, first_name, and last_name. The Downloads table contains download information such as download_id, user_id, book_id, filename, download_datetime, and ip_address. The Books table contains book information such as book_id, filename, title, authors, publisher, ISBN, and publication_year.

This schema is not in 3NF as there are dependencies between non-key attributes and partial dependencies. The Downloads table has a partial dependency on user_id and book_id, meaning that the attributes of the table are not functionally dependent on the entire primary key.

After normalizing the schema, we have four tables: Users, Books, Downloads, and Download_Info. The Users table remains the same as the original schema. The Books table also remains the same. However, we have split the Downloads table into two tables: Downloads and Download_Info. The Downloads table contains only the download_id, user_id, and book_id columns, which form the primary key. The Download_Info table contains the remaining columns from the original Downloads table: filename, download_datetime, and ip_address. This separation eliminates the partial dependency and creates two tables that are both in 3NF.



Explanation:

The normalized database schema includes four tables: Users, Books, Downloads, and Download_Info.

The Users table remains the same as in the original schema and includes columns for user_id, username, email, first_name, and last_name.

The Books table also remains the same as in the original schema and includes columns for book_id, filename, title, authors, publisher, ISBN, and publication_year.

The Downloads table has been split into two tables: Downloads and Download_Info. The Downloads table includes only the columns download_id, user_id, and book_id, which form the primary key of the table. This separation helps to eliminate the partial dependency and creates two tables that are both in 3NF.

The Download_Info table contains the remaining columns from the original Downloads table, including filename, download_datetime, and ip_address. This table is related to the Downloads table by the download_id column.

References

1. Lucidchart: A web-based diagramming tool that offers various templates, including database diagrams. <https://www.lucidchart.com/>
2. Draw.io: A free online diagramming tool that allows you to create a variety of diagrams, including ER diagrams. <https://app.diagrams.net/>
3. SQL tutorials - w3 schools, <https://www.w3schools.com/sql>

Remarks:

Project 1 provided a comprehensive understanding of DBMS concepts, including Joins, Unions, Aggregate functions, Subqueries, Triggers, and Procedures. It also provided experience in advanced queries and database design models, with practical applications of procedures and views. Through the project, I gained proficiency in executing DDL and DML applications, drawing logical models, and mapping them. Additionally, I learned about SQL data types, constraints, and statement processing, as well as different types of operators and case statements. Overall, the project was a valuable experience for developing a deeper understanding of basic to advanced DBMS concepts.

