

## Project 3

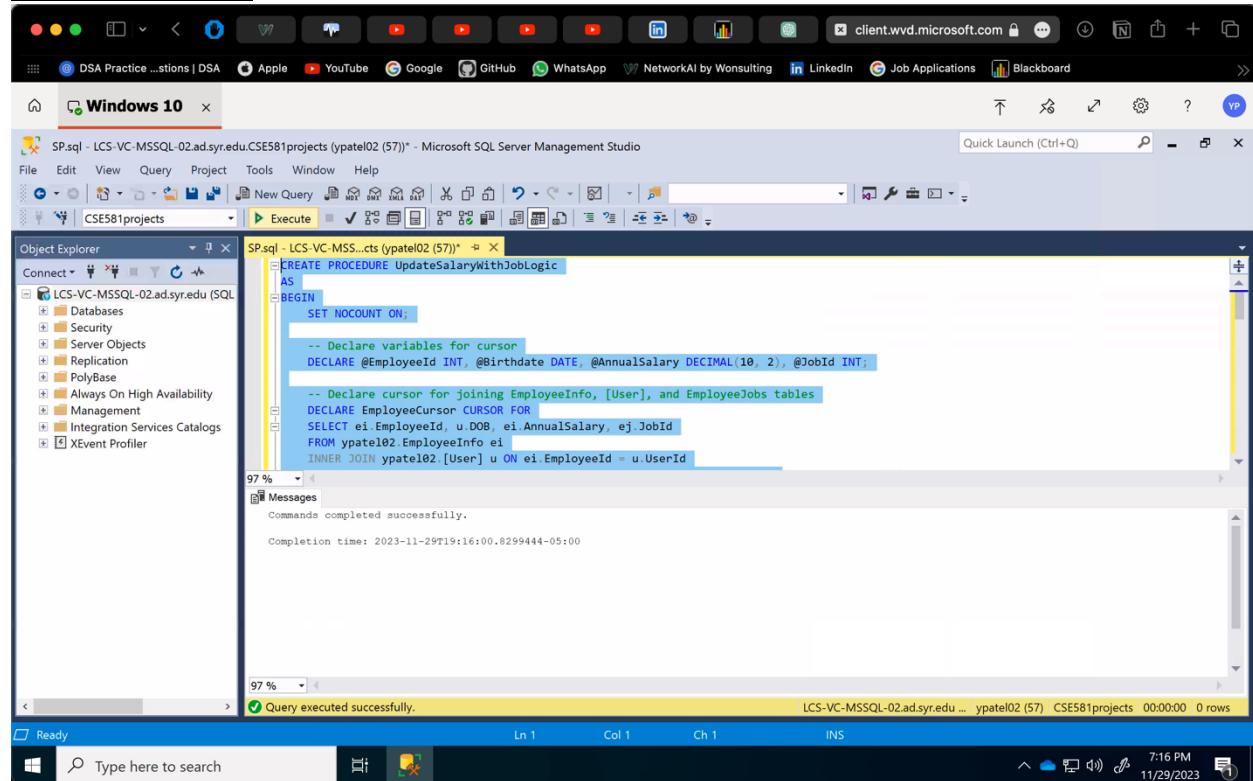
**Submission By – Yash Rajeshkumar Patel | SUID - 517958851**

- **(10% points) SP to use cursor(s)**

**Stored Procedure Name: UpdateSalaryWithJobLogic**

**Purpose:** The UpdateSalaryWithJobLogic stored procedure is designed to intelligently adjust the annual salary of employees based on a set of business rules. This stored procedure utilizes a cursor to iterate through a result set obtained by joining three tables: EmployeeInfo, User, and EmployeeJobs. The primary focus is to identify employees aged 25 or older and, for a specific set of job roles (JobIds 5, 7, 9, 11, 13), grant a 7% increase in their annual salary. For other job roles, a 5% increase is applied.

**Creation of SP 1**



The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SP.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (57))". The main window displays the T-SQL code for the stored procedure:

```

CREATE PROCEDURE UpdateSalaryWithJobLogic
AS
BEGIN
    SET NOCOUNT ON;

    -- Declare variables for cursor
    DECLARE @EmployeeId INT, @Birthdate DATE, @AnnualSalary DECIMAL(10, 2), @JobId INT;

    -- Declare cursor for joining EmployeeInfo, [User], and EmployeeJobs tables
    DECLARE EmployeeCursor CURSOR FOR
        SELECT ei.EmployeeId, u.DOB, ei.AnnualSalary, ej.JobId
        FROM ypatel02.EmployeeInfo ei
        INNER JOIN ypatel02.[User] u ON ei.EmployeeId = u.UserId

```

The code is highlighted in blue and yellow. The status bar at the bottom indicates "Commands completed successfully." and "Completion time: 2023-11-29T19:16:00.8299444-05:00".

## Data before Execution SP1

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer pane on the left shows the database connection to 'LCS-VC-MSSQL-02.ad.syr.edu'. The main pane displays a T-SQL script for a stored procedure named 'yplate02.UpdateSalaryWithJobLogic'. The script includes a PRINT statement, a comment '-- Before Updating', and a SELECT query that joins four tables: EmployeeInfo (ei), User (u), EmployeeJobs (ej), and Employee (eo). Below the script, the 'Results' tab is selected, showing a table with 10 rows of employee data. The table has columns: EmployeeId, DOB, AnnualSalary, and JobId. The data is as follows:

EmployeeId	DOB	AnnualSalary	JobId
1	1995-06-15	8000.00	5
2	1996-08-20	8100.00	6
3	1997-02-10	8200.00	7
4	1998-11-30	8300.00	8
5	1999-07-25	8400.00	9
6	2000-04-14	8500.00	10
7	2001-01-03	8600.00	11
8	1997-09-12	8700.00	12
9	1998-06-05	8800.00	13
10	1999-03-18	8900.00	14

At the bottom of the results pane, a message indicates 'Query executed successfully.' The status bar at the bottom right shows the date and time: 11/29/2023 7:14 PM.

## Executing SP1

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer pane on the left shows the database connection to 'LCS-VC-MSSQL-02.ad.syr.edu'. The main pane displays the same T-SQL script for the stored procedure 'yplate02.UpdateSalaryWithJobLogic'. The script now includes an EXEC statement to run the procedure. Below the script, the 'Messages' tab is selected, showing the output of the procedure execution. The message reads 'Salary update with job logic completed.' followed by the completion time: 'Completion time: 2023-11-29T19:19:15.2933453-05:00'. At the bottom of the messages, a message indicates 'Query executed successfully.' The status bar at the bottom right shows the date and time: 11/29/2023 7:19 PM.

## Results after Executing SP1

The screenshot shows a Microsoft SQL Server Management Studio (SSMS) window titled "SP.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (57))". The query pane displays the following T-SQL code:

```
-- Executing Stored Procedure
EXEC ypatel02.UpdateSalaryWithJobLogic;
-- After
SELECT ei.EmployeeId, u.DOB, ei.AnnualSalary, ej.JobId
FROM ypatel02.EmployeeInfo ei
INNER JOIN ypatel02.[User] u ON ei.EmployeeId = u.UserId
INNER JOIN ypatel02.EmployeeJobs ej ON ei.EmployeeId = ej.EmployeeId;
```

The results pane shows a table with the following data:

	EmployeeId	DOB	AnnualSalary	JobId
1	24	1995-06-15	8560.00	5
2	25	1996-08-20	8505.00	6
3	26	1997-02-10	8774.00	7
4	27	1998-11-30	8715.00	8
5	28	1999-07-25	8400.00	9
6	29	2000-04-14	8500.00	10
7	30	2001-01-03	8600.00	11
8	31	1997-09-12	9135.00	12
9	32	1998-06-05	9416.00	13
10	33	1999-03-18	8900.00	14

At the bottom of the results pane, a message states "Query executed successfully." and "LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (57) CSE581projects 00:00:00 10 rows". The status bar at the bottom of the screen shows "Ready", "Type here to search", "7:19 PM", and "11/29/2023".

- **(10% points) SP to update data in a table (perform validation)**

### Stored Procedure Name: UpdateEmployeeDataWithValidation

Description : This stored procedure is designed to update employee data with validation checks. It takes input parameters such as @EmployeeId (for identifying the employee), @NewSalary (the updated annual salary), and @NewJobTitle (the updated job title).

#### Validations :

- Verifies whether the specified employee exists in the **EmployeeInfo** table.
- Validates that the new salary provided (**@NewSalary**) is not negative.
- Validates that the new job title provided (**@NewJobTitle**) exists in the **JobInfo** table.

#### Creation of SP2

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer pane on the left shows a connection to 'LCS-VC-MSSQL-02.ad.syr.edu'. The 'CSE581projects' database is selected. The 'Scripting' tab is active in the toolbar. A query window titled 'SP2.sql - LCS-VC-MS...cts (ypatel02 (54))' contains the following T-SQL code:

```

CREATE PROCEDURE UpdateEmployeeDataWithValidation
    @EmployeeId INT,
    @NewSalary DECIMAL(10, 2),
    @NewJobTitle NVARCHAR(50)
AS
BEGIN
    SET NOCOUNT ON;

    -- Check if employee exists
    IF NOT EXISTS (SELECT 1 FROM ypatel02.EmployeeInfo WHERE EmployeeId = @EmployeeId)
    BEGIN
        RAISERROR('Employee does not exist.', 16, 1);
        RETURN;
    END

```

The 'Messages' pane at the bottom shows the output of the command:

```

Commands completed successfully.

Completion time: 2023-11-29T22:25:26.1216453-05:00

```

The status bar at the bottom right indicates 'Query executed successfully.' and shows the date and time: '10:25 PM 11/29/2023'.

## Data Before Executing SP2

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the CSE581projects database. The central pane displays a query results grid for a SELECT statement. The results show 10 rows of employee data:

EmployeeId	JobId	JobTitle	AnnualSalary	
1	24	5	Software Engineer	8500.00
2	25	6	Mechanical Engineer	8500.00
3	26	7	Business Analyst	3774.00
4	27	8	Biologist	8715.00
5	28	9	Historian	8400.00
6	29	10	Data Scientist	8500.00
7	30	11	Electrical Engineer	8600.00
8	31	12	Marketing Manager	9125.00
9	32	13	Chemist	9416.00
10	33	14	Archivist	8900.00

At the bottom, a message indicates "Query executed successfully."

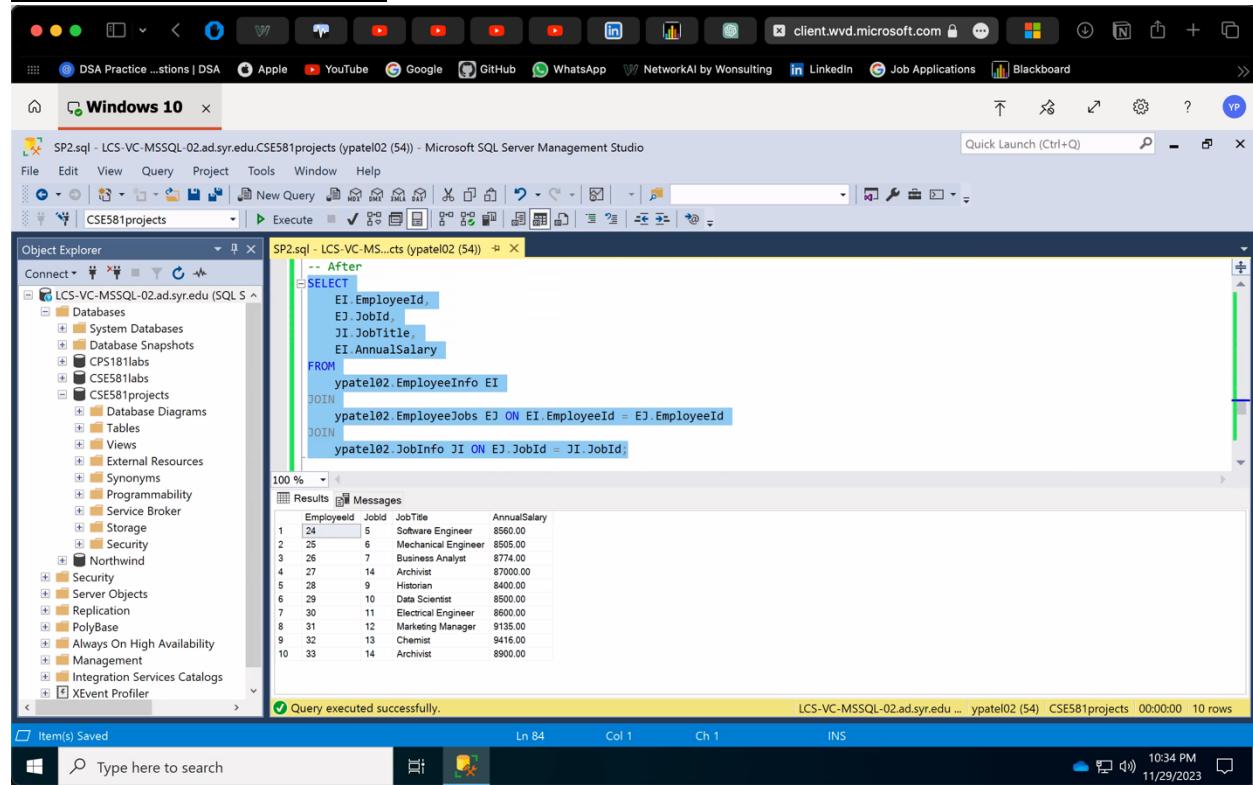
## Executing SP2 (Valid)

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the CSE581projects database. The central pane displays a query results grid for a SELECT statement. The results show 1 row of employee data:

EmployeeId	JobId	JobTitle	AnnualSalary
27	14	Archivist	8900.00

At the bottom, a message indicates "Employee data updated successfully."

## Data After Executing SP2

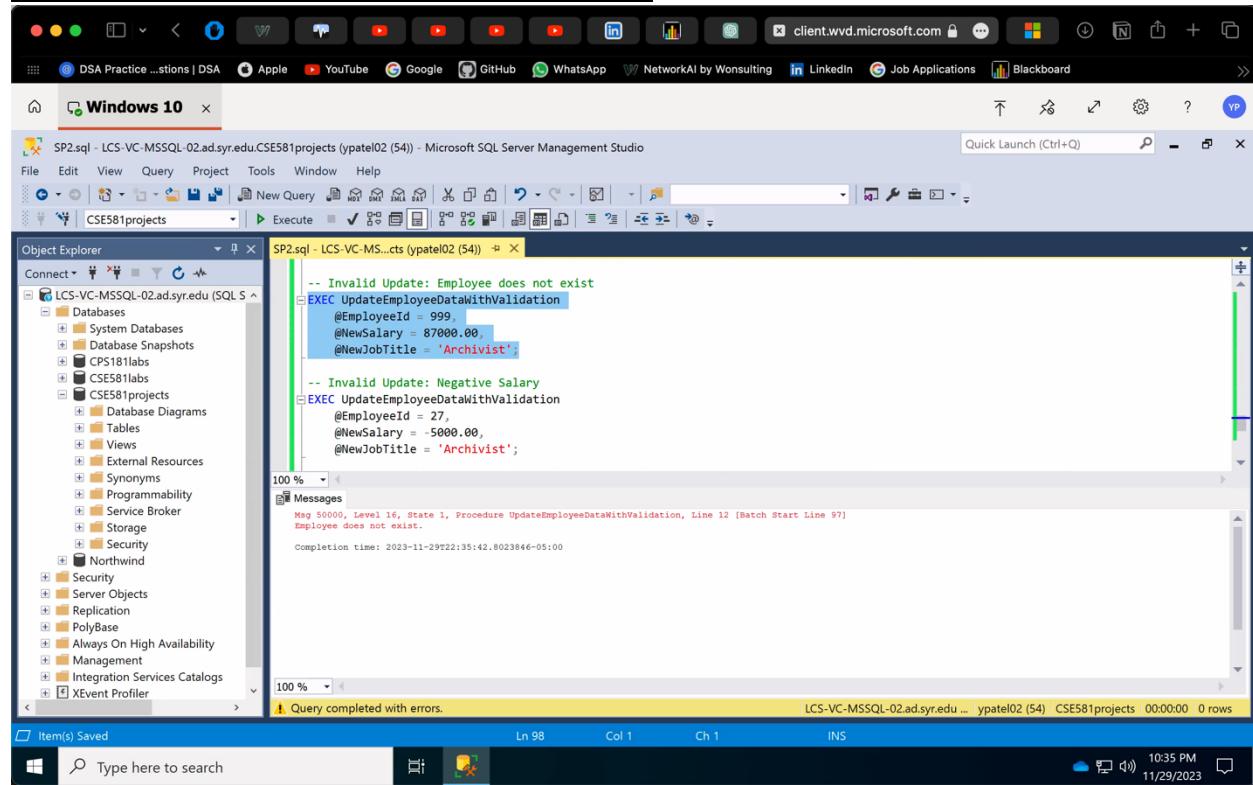


```
-- After
SELECT
    EI.EmployeeId,
    EJ.JobId,
    JI.JobTitle,
    EI.AnnualSalary
FROM
    ypatel02.EmployeeInfo EI
JOIN
    ypatel02.EmployeeJobs EJ ON EI.EmployeeId = EJ.EmployeeId
JOIN
    ypatel02.JobInfo JI ON EJ.JobId = JI.JobId;
```

EmployeeId	JobId	JobTitle	AnnualSalary	
1	24	5	Software Engineer	8500.00
2	25	6	Mechanical Engineer	8500.00
3	26	7	Business Analyst	3774.00
4	27	14	Archivist	8700.00
5	28	9	Historian	8400.00
6	29	10	Data Scientist	8500.00
7	30	11	Electrical Engineer	8600.00
8	31	12	Marketing Manager	9125.00
9	32	13	Chemist	8416.00
10	33	14	Archivist	8900.00

Query executed successfully.

## Executing when Employee does not exists.



```
-- Invalid Update: Employee does not exist
EXEC UpdateEmployeeDataWithValidation
    @EmployeeId = 999,
    @NewSalary = 87000.00,
    @NewJobTitle = 'Archivist';

-- Invalid Update: Negative Salary
EXEC UpdateEmployeeDataWithValidation
    @EmployeeId = 27,
    @NewSalary = -5000.00,
    @NewJobTitle = 'Archivist';
```

Completion time: 2023-11-29T22:35:42.8023846-05:00

Query completed with errors.

## Executing when Negative Salary is Entered

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a connection to 'LCS-VC-MSSQL-02.ad.syr.edu'. The 'CSE581projects' database is selected. In the center, the 'SP2.sql' file is open, containing a stored procedure definition. A portion of the code is highlighted in blue:

```
-- Invalid Update: Negative Salary
EXEC UpdateEmployeeDataWithValidation
    @EmployeeId = 27,
    @NewSalary = -5000.00,
    @NewJobTitle = 'Archivist';
```

Execution results are shown in the 'Messages' pane:

```
Msg 50000, Level 16, State 2, Procedure UpdateEmployeeDataWithValidation, Line 19 [Batch Start Line 100]
Invalid salary. Salary cannot be negative.
```

The status bar at the bottom indicates 'Query completed with errors.'

## Executing when Invalid Job Title Entered

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a connection to 'LCS-VC-MSSQL-02.ad.syr.edu'. The 'CSE581projects' database is selected. In the center, the 'SP2.sql' file is open, containing a stored procedure definition. A portion of the code is highlighted in blue:

```
-- Invalid Update: Invalid Job Title
EXEC UpdateEmployeeDataWithValidation
    @EmployeeId = 27,
    @NewSalary = 87000.00,
    @NewJobTitle = 'InvalidJobTitle';
```

Execution results are shown in the 'Messages' pane:

```
Msg 50000, Level 16, State 3, Procedure UpdateEmployeeDataWithValidation, Line 26 [Batch Start Line 109]
Invalid job title.
```

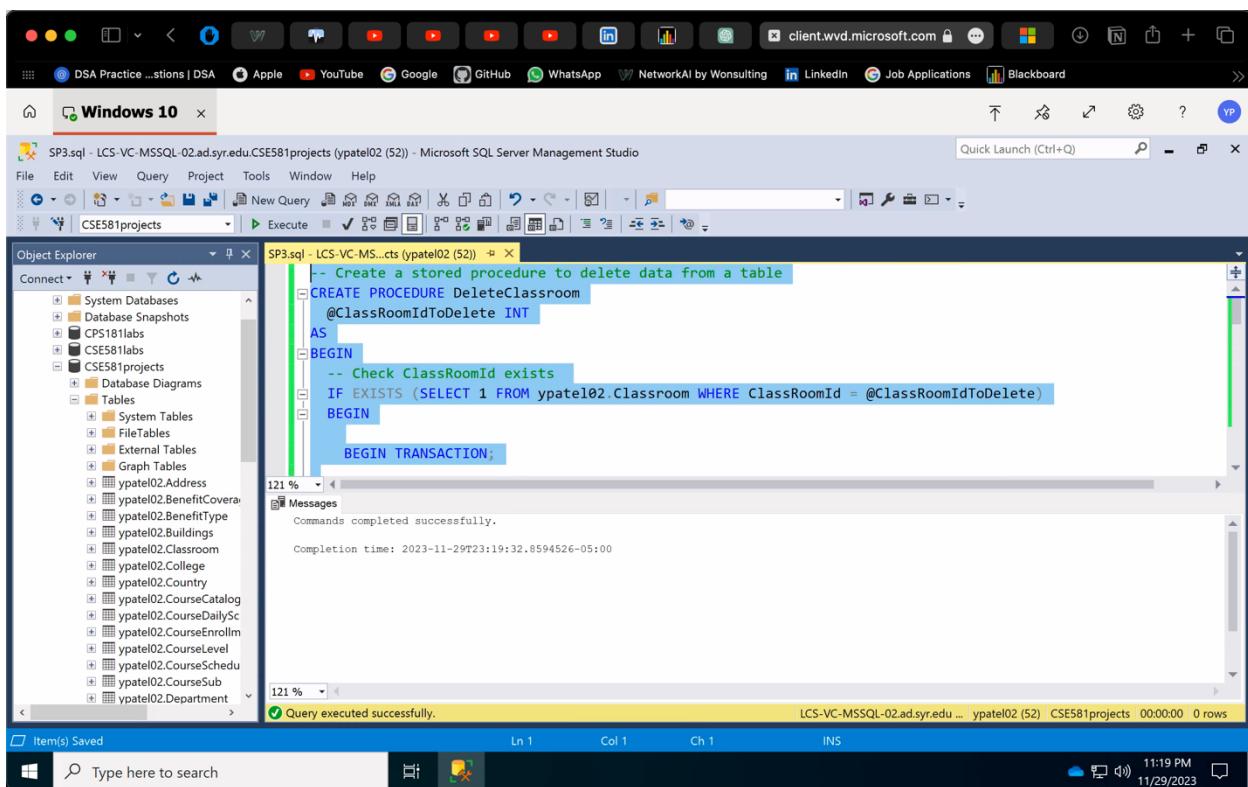
The status bar at the bottom indicates 'Query completed with errors.'

- **(10% points) SP to delete data from a table**

### Stored Procedure Name: DeleteClassroom

Description: The stored procedure DeleteClassroom is designed to facilitate the secure deletion of a classroom record from the ypatel02.Classroom table in our project database. Ensure number of classrooms available in a building gets updated accordingly and sets classroom of classes to be null if that classroom was removed.

### Creation of SP3



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The title bar indicates the connection is to 'SP3.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (52))'. The Object Explorer on the left shows the database structure, including tables like Address, BenefitCoverage, BenefitType, Buildings, Classroom, College, Country, CourseCatalog, CourseDailySc, CourseEnrollm, CourseLevel, CourseSchedu, CourseSub, and Department. The main query editor window contains the following T-SQL code:

```
-- Create a stored procedure to delete data from a table
CREATE PROCEDURE DeleteClassroom
    @ClassRoomIdToDelete INT
AS
BEGIN
    -- Check ClassRoomId exists
    IF EXISTS (SELECT 1 FROM ypatel02.Classroom WHERE ClassRoomId = @ClassRoomIdToDelete)
    BEGIN
        BEGIN TRANSACTION;
        -- Your delete logic here
        COMMIT TRANSACTION;
    END
END
```

The 'Messages' pane at the bottom shows the command completed successfully with a completion time of 2023-11-29T23:19:32.8594526-05:00. The status bar at the bottom right shows the session details: LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) CSE581projects 00:00:00 0 rows.

## Data Before Executing SP3

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases and projects, including 'CSE581projects'. The central pane displays a query window titled 'SP3.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (52))'. The query is:

```
-- Before
SELECT
    C.ClassRoomID,
    B.BuildingID,
    B.BuildingName,
    B.NumberOfClasses AS NumberOfClassesInBuilding
FROM
    Classroom C
JOIN
    Buildings B ON C.BuildingID = B.BuildingID
ORDER BY
    C.ClassRoomID
```

The 'Results' tab shows the output of the query:

ClassRoomID	BuildingID	BuildingName	NumberOfClassesInBuilding
1	2	Engineering Building	12
2	3	Science Hall	10
3	4	Science Hall	10
4	5	Business Center	12
5	6	Business Center	12
6	7	Liberal Arts Building	8
7	8	Liberal Arts Building	8
8	9	History Tower	6
9	10	History Tower	6

The status bar at the bottom indicates 'Query executed successfully.' and 'LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) CSE581projects 00:00:00 9 rows'.

## Executing SP3

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases and projects, including 'CSE581projects'. The central pane displays a query window titled 'SP3.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (52))'. The query is:

```
-- Before
SELECT
    C.ClassRoomID,
    B.BuildingID,
    B.BuildingName,
    B.NumberOfClasses AS NumberOfClassesInBuilding
FROM
    Classroom C
JOIN
    Buildings B ON C.BuildingID = B.BuildingID
ORDER BY
    C.ClassRoomID

-- Execute
EXEC ypatel02.DeleteClassroom @ClassRoomIdToDelete = 2

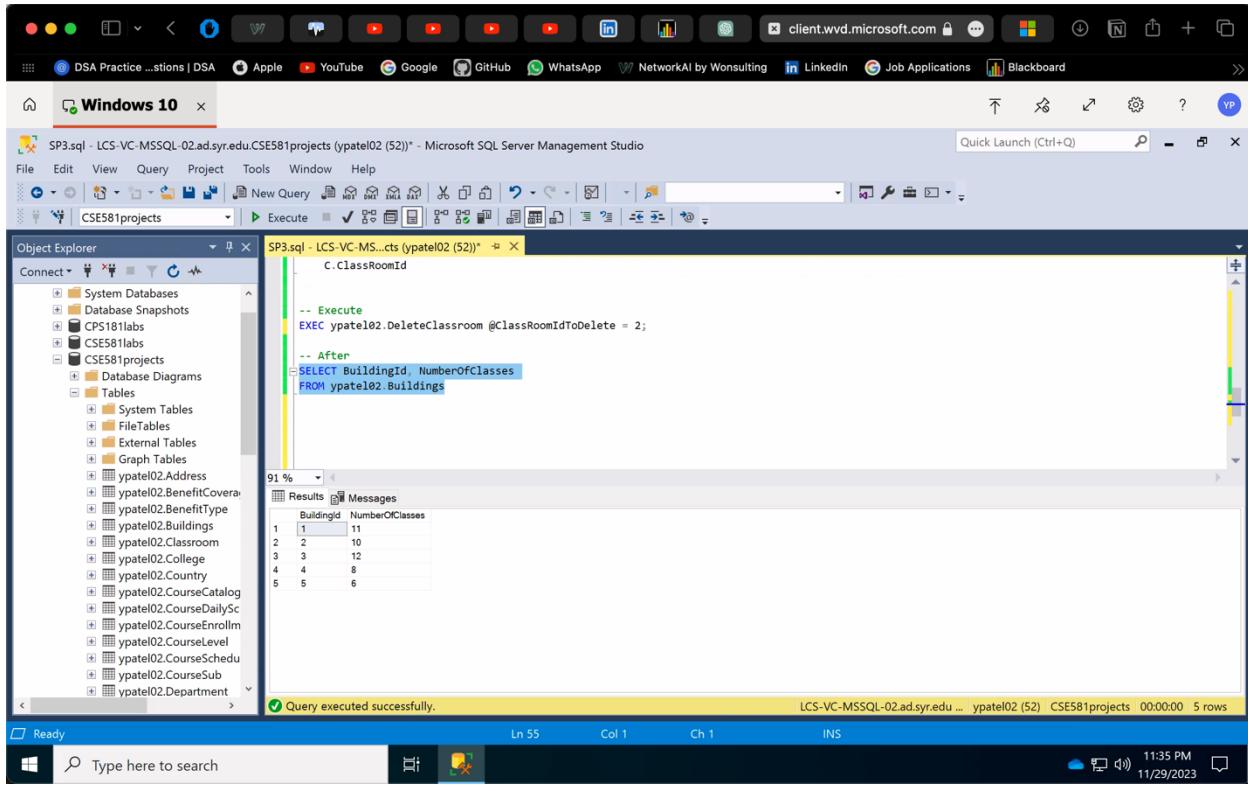
-- After
```

The 'Messages' tab shows the execution results:

```
(1 row affected)
(1 row affected)
(1 row affected)
```

The status bar at the bottom indicates 'Query executed successfully.' and 'LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) CSE581projects 00:00:00 0 rows'.

## Data After Executing SP3



The screenshot shows the Microsoft SQL Server Management Studio interface running on a Windows 10 desktop. The title bar reads "SP3.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (52))". The Object Explorer sidebar lists various database objects. In the center, a query window titled "C.ClassroomId" contains the following SQL code:

```
-- Execute  
EXEC ypatel02.DeleteClassroom @ClassRoomIdToDelete = 2;  
  
-- After  
SELECT BuildingId, NumberOfClasses  
FROM ypatel02Buildings
```

The results pane displays the following data:

	BuildingId	NumberOfClasses
1	1	11
2	2	10
3	3	12
4	4	8
5	5	6

A status bar at the bottom indicates "Query executed successfully." and "LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) CSE581projects 00:00:00 5 rows". The system tray shows the date and time as "11/29/2023 11:35 PM".

- **(10%+5% bonus points) 1 SP of your own choice, performing a business action<sup>1</sup>.**

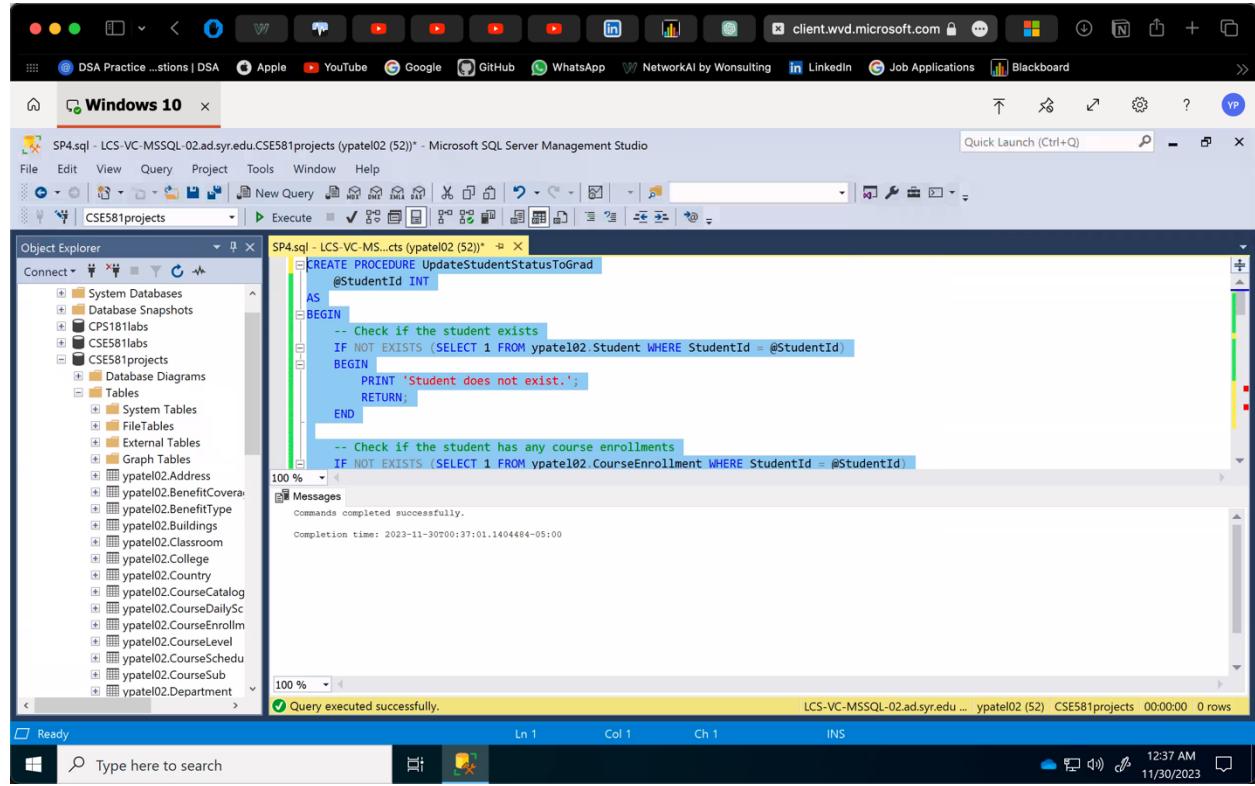
### Stored Procedure Name : UpdateStudentStatusToGrad

Description : The stored procedure UpdateStudentStatusToGrad automates the process of updating a student's graduation status within the educational system. This checks Grades obtained in courses enrolled during his study and will not allow to graduate if he has failed one or more subjects.

#### Validations :

- Check if Student Exists
- Check if Student Failed one or more Courses.
- Checks if Student already Graduated.

#### Creation of SP4



The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SP4.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (52)) - Microsoft SQL Server Management Studio". The main window displays the following T-SQL code:

```

CREATE PROCEDURE UpdateStudentStatusToGrad
    @StudentId INT
AS
BEGIN
    -- Check if the student exists
    IF NOT EXISTS (SELECT 1 FROM ypatel02.Student WHERE StudentId = @StudentId)
    BEGIN
        PRINT 'Student does not exist.';
        RETURN;
    END

    -- Check if the student has any course enrollments
    IF NOT EXISTS (SELECT 1 FROM ypatel02.CourseEnrollment WHERE StudentId = @StudentId)
    BEGIN
        PRINT 'Student has no courses enrolled.';
        RETURN;
    END
END

```

The code is highlighted in blue and green, indicating syntax. The status bar at the bottom right shows "Commands completed successfully." and "Completion time: 2023-11-30T00:37:01.1404484-05:00". The status bar also indicates "Query executed successfully." and "LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) CSE581projects 00:00:00 0 rows".

---

<sup>1</sup> You will get 5% bonus if this is a valid action within the scope of the business problem, and the SP is reasonably complex. In other words, a 4 line single-table select SP will get you nothing.

## Data Before Executing SP4

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including System Databases, Database Snapshots, CPS181labs, CSE581labs, and CSE581projects. The CSE581projects database is selected. The Results pane on the right displays the output of a query:

```
--Before
SELECT C.StudentId, C.CRN, C.EnrollmentStatusId, S.StudentStatusId, S.IsGraduate
FROM CourseEnrollment C
LEFT JOIN Student S ON C.StudentId = S.StudentId;
```

The results show 10 rows of data:

StudentId	CRN	EnrollmentStatusId	StudentStatusId	IsGraduate
1	24	1	1	0
2	25	2	1	0
3	26	3	3	1
4	27	4	2	0
5	28	5	1	0
6	29	6	1	0
7	30	7	2	0
8	31	8	3	1
9	32	9	4	0
10	33	10	5	1

Message bar: Query executed successfully.

## Executing SP4

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including System Databases, Database Snapshots, CPS181labs, CSE581labs, and CSE581projects. The CSE581projects database is selected. The Results pane on the right displays the output of a query:

```
--Before
SELECT C.StudentId, C.CRN, C.EnrollmentStatusId, S.StudentStatusId, S.IsGraduate
FROM CourseEnrollment C
LEFT JOIN Student S ON C.StudentId = S.StudentId;

-- Valid Execution
EXEC UpdateStudentStatusToGrad 24;

-- After
SELECT C.StudentId, C.CRN, C.EnrollmentStatusId, S.StudentStatusId, S.IsGraduate
FROM CourseEnrollment C
LEFT JOIN Student S ON C.StudentId = S.StudentId;
```

The results show 1 row affected, indicating the update was successful. The message bar at the bottom states "StudentStatus updated successfully".

Message bar: Query executed successfully.

## Data After Executing SP4

```
--Before
SELECT C.StudentId, C.CRN, C.EnrollmentStatusId, S.StudentStatusId, S.IsGraduate
FROM CourseEnrollment C
LEFT JOIN Student S ON C.StudentId = S.StudentId;

-- Valid Execution
EXEC UpdateStudentStatusToGrad 24;

-- After
SELECT C.StudentId, C.CRN, C.EnrollmentStatusId, S.StudentStatusId, S.IsGraduate
FROM CourseEnrollment C
LEFT JOIN Student S ON C.StudentId = S.StudentId;
```

StudentId	CRN	EnrollmentStatusId	StudentStatusId	IsGraduate
1	24	1	3	1
2	25	2	1	0
3	26	3	3	1
4	27	4	2	0
5	28	5	1	0
6	29	6	1	0
7	30	7	2	0
8	31	8	3	0
9	32	9	4	0
10	33	10	5	1

Query executed successfully.

## Executing SP4 on Invalid Student (does not exists)

```
-- After
SELECT C.StudentId, C.CRN, C.EnrollmentStatusId, S.StudentStatusId, S.IsGraduate
FROM CourseEnrollment C
LEFT JOIN Student S ON C.StudentId = S.StudentId;

-- Invalid Execution (Student does not exist)
EXEC UpdateStudentStatusToGrad 100;

-- Invalid Execution (Some enrollments have EnrollmentStatusId <> 5) i.e. Student has Failed one or more subjects
EXEC UpdateStudentStatusToGrad 28;

-- Invalid Execution (Student is already in StudentStatusId = 3) i.e. Already a Graduated Student
EXEC UpdateStudentStatusComplex 26;
```

Student does not exist.

Completion time: 2023-11-30T00:42:00.5020514-05:00

Query executed successfully.

## Executing SP4 on Student who Failed one or more Courses

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SP4.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (52)) - Microsoft SQL Server Management Studio". The Object Explorer on the left shows the database structure, including System Databases, Database Snapshots, CPS181labs, CSE581labs, and CSE581projects. The CSE581projects folder contains Database Diagrams, Tables, System Tables, FileTables, External Tables, Graph Tables, and several tables starting with ypatel02\_. The main pane displays the stored procedure code:

```
-- After
SELECT C.StudentId, C.CRN, C.EnrollmentStatusId, S.StudentStatusId, S.IsGraduate
FROM CourseEnrollment C
LEFT JOIN Student S ON C.StudentId = S.StudentId;

-- Invalid Execution (Student does not exist)
EXEC UpdateStudentStatusToGrad 100;

-- Invalid Execution (Some enrollments have EnrollmentStatusId <> 5) i.e. Student has Failed one or more subjects
EXEC UpdateStudentStatusToGrad 28;

-- Invalid Execution (Student is already in StudentStatusId = 3) i.e. Already a Graduated Student
EXEC UpdateStudentStatusComplex 26;
```

The Messages pane shows the following error message:

Cannot update StudentStatus. Some enrollments have EnrollmentStatusId <> 5.  
Completion time: 2023-11-30T00:42:32.7844121-05:00

The status bar at the bottom indicates "Query executed successfully.", "LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) CSE581projects 00:00:00 0 rows", and the current date and time as "11/30/2023 12:42 AM".

## Executing SP4 A student who is already Graduated

The screenshot shows the Microsoft SQL Server Management Studio interface, identical to the previous one but with a different query. The title bar reads "SP4.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (52))\* - Microsoft SQL Server Management Studio". The main pane displays the same stored procedure code as before:

```
-- After
SELECT C.StudentId, C.CRN, C.EnrollmentStatusId, S.StudentStatusId, S.IsGraduate
FROM CourseEnrollment C
LEFT JOIN Student S ON C.StudentId = S.StudentId;

-- Invalid Execution (Student does not exist)
EXEC UpdateStudentStatusToGrad 100;

-- Invalid Execution (Some enrollments have EnrollmentStatusId <> 5) i.e. Student has Failed one or more subjects
EXEC UpdateStudentStatusToGrad 28;

-- Invalid Execution (Student is already in StudentStatusId = 3) i.e. Already a Graduated Student
EXEC UpdateStudentStatusToGrad 26;
```

The Messages pane shows the following error message:

Student is already in StudentStatusId = 3.  
Completion time: 2023-11-30T00:43:35.8006516-05:00

The status bar at the bottom indicates "Query executed successfully.", "LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) CSE581projects 00:00:00 0 rows", and the current date and time as "11/30/2023 12:43 AM".

**Create 1 Function (10% points) which can be executed by “Graders”. Not the function you did for the lab.**

**Function Name : FindAvailableClassrooms**

**Description :** This function, ypatel02.FindAvailableClassrooms, is designed to find available classrooms for a specified day of the week and time range. It utilizes information from the ypatel02.Classroom, ypatel02.CourseDailySchedule, and ypatel02.CourseSchedule tables. The function identifies classrooms that are not scheduled for courses during the specified period, considering both the end and start times of the courses. Additionally, it handles cases where ClassRoomId is NULL in the CourseSchedule table. The result is a list of distinct available ClassRoomIds meeting the given criteria.

**Creation of Function**

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The title bar reads "Function.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (52)) - Microsoft SQL Server Management Studio". The main window displays the following T-SQL code:

```
CREATE FUNCTION ypatel02.FindAvailableClassrooms (
    @dayofweek NVARCHAR(50),
    @reqStartHour INT,
    @reqStartMin INT,
    @reqEndHour INT,
    @reqEndMin INT
)
RETURNS TABLE
AS
BEGIN
    RETURN (
        SELECT DISTINCT c.ClassRoomId
        FROM ypatel02.Classroom c
        WHERE c.ClassRoomId NOT IN (
            SELECT cs.ClassRoomId
            FROM ypatel02.CourseDailySchedule cs
            WHERE cs.DayOfTheWeek = @dayofweek
                AND cs.StartTime <= DATEADD(minute, (@reqStartHour * 60) + @reqStartMin, GETDATE())
                AND cs.EndTime >= DATEADD(minute, (@reqEndHour * 60) + @reqEndMin, GETDATE())
        )
    );
END
```

The status bar at the bottom indicates "Query executed successfully." and "Completion time: 2023-11-30T21:02:38.2999125-05:00". The bottom right corner shows the date and time as "11/30/2023 9:02 PM".

## CourseSchedule Table Before Executing Function

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a connection to 'LCS-VC-MSSQL-02.ad.syr.edu'. The 'CSE581projects' database is selected. The 'Function.sql' file is open in the main pane. The code includes a function definition and several SELECT statements. The results pane displays the 'CourseSchedule' table with 10 rows. The status bar at the bottom right shows the query was executed successfully at 8:49 PM on 11/30/2023.

CRN	CourseCode	CourseNumber	Section	SemesterId	ClassRoomId
1	CSCI	101	001	1	NULL
2	CSCI	201	001	2	NULL
3	MATH	101	001	1	3
4	CHEM	101	001	2	4
5	PHYS	101	001	1	5
6	HIST	101	001	2	6
7	CSCI	301	001	1	7
8	MATH	201	001	2	8
9	CHEM	201	001	1	9
10	PHYS	201	001	2	10

## CourseDailySchedule before Executing Function

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a connection to 'LCS-VC-MSSQL-02.ad.syr.edu'. The 'CSE581projects' database is selected. The 'Function.sql' file is open in the main pane. The code includes a function definition and several SELECT statements. The results pane displays the 'CourseDailySchedule' table with 10 rows. The status bar at the bottom right shows the query was executed successfully at 8:49 PM on 11/30/2023.

DailyId	CRN	DayOfWeek	StartTime	StartMinute	EndHour	EndMinute
1	1	Monday	9	0	10	30
2	1	Wednesday	9	0	10	30
3	2	Tuesday	13	30	15	0
4	2	Thursday	13	30	15	0
5	3	Monday	10	30	12	0
6	3	Wednesday	10	30	12	0
7	4	Tuesday	15	0	16	30
8	4	Thursday	15	0	16	30
9	5	Monday	8	0	9	30
10	5	Wednesday	8	0	9	30

## Classroom Table Before executing Function

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "Function.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (S2)) - Microsoft SQL Server Management Studio". The Object Explorer on the left shows the database structure. The main pane displays a query window with the following SQL code:

```
        )
    );
GO;

--SELECT * FROM ypatel02.CourseSchedule;
--SELECT * FROM ypatel02.CourseDailySchedule;
--SELECT DISTINCT ClassroomId FROM Classroom;
--SELECT * FROM Classroom;

-- Checking when occupied by classroom 1 (9-10:30), 4(8-9:30)
SELECT * FROM ypatel02.FindAvailableClassrooms(N'Monday', 9, 10, 20, 10);
```

The Results pane shows the output of the last SELECT statement:

ClassRoomId	BuildingId	Level	RoomNumber	ProjectorId	Capacity
1	3	2	101	3	60
2	4	2	201	4	30
3	5	3	101	5	70
4	6	3	201	6	35
5	7	4	101	7	55
6	8	4	201	8	45
7	9	5	101	9	25
8	10	5	201	10	20

A status bar at the bottom indicates "Query executed successfully." and "LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (S2) CSE581projects 00:00:00 8 rows".

## Function Execution Case 1 : Checking when occupied by classroom 1 (9-10:30), 5(8-9:30)

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

```
GO;

-- Tables we should know
SELECT * FROM ypatel02.CourseSchedule;
SELECT * FROM ypatel02.CourseDailySchedule;
-- Distinct Classrooms are 3,4,5,6,7,8,9,10 as classroom 1 and 2 are deleted
SELECT DISTINCT ClassroomId FROM Classroom;
SELECT * FROM Classroom;

-- Executing Function
-- Checking when occupied by classroom 1 (9-10:30), 5(8-9:30)
SELECT * FROM ypatel02.FindAvailableClassrooms('Monday', 9, 10, 10, 10);
-- Checking when all classrooms are free
```

The results pane shows a table with one row:

ClassRoomId
3

Message bar: Query executed successfully.

## Function Execution Case 2 : Checking when all classrooms are free.

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

```
-- Tables we should know
SELECT * FROM ypatel02.CourseSchedule;
SELECT * FROM ypatel02.CourseDailySchedule;
-- Distinct Classrooms are 3,4,5,6,7,8,9,10 as classroom 1 and 2 are deleted
SELECT DISTINCT ClassroomId FROM Classroom;
SELECT * FROM Classroom;

-- Executing Function
-- Checking when occupied by classroom 1 (9-10:30), 5(8-9:30)
SELECT * FROM ypatel02.FindAvailableClassrooms('Monday', 9, 10, 10, 10);
-- Checking when all classrooms are free
SELECT * FROM ypatel02.FindAvailableClassrooms('Monday', 14, 10, 15, 30);
-- Checking when occupied by Classroom 4 (15-16:30)
```

The results pane shows a table with eight rows:

ClassRoomId
3
4
5
6
7
8
9
10

Message bar: Query executed successfully.

### Function Execution Case 3 : Checking when occupied by classroom 4 (15-16:30)

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

```
SELECT * FROM ypatel02.CourseDailySchedule;
-- Distict Classrooms are 3,4,5,6,7,8,9,10 as classroom 1 and 2 are deleted
SELECT DISTINCT ClassRoomId FROM Classroom;
SELECT * FROM Classroom;

--- Executing Function
-- Checking when occupied by classroom 1 (9-10:30), 5(8-9:30)
SELECT * FROM ypatel02.FindAvailableClassrooms(N'Monday', 9, 10, 10, 10);
-- Checking when all classrooms are free
SELECT * FROM ypatel02.FindAvailableClassrooms(N'Monday', 14, 10, 15, 30);
-- Checking when occupied by classroom 4 (15-16:30)
SELECT * FROM ypatel02.FindAvailableClassrooms(N'Tuesday', 14, 10, 15, 15);
-- Checking when all classrooms are free
```

The results pane shows a table with the following data:

ClassRoomId
1
2
3
4
5
6
7
9
10

A status bar at the bottom indicates "Query executed successfully." and "LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) CSE581projects 00:00:00 7 rows".

### Function Execution Case 4 : Checking when all classrooms are free.

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

```
SELECT DISTINCT ClassRoomId FROM Classroom;
SELECT * FROM Classroom;

--- Executing Function
-- Checking when occupied by classroom 1 (9-10:30), 5(8-9:30)
SELECT * FROM ypatel02.FindAvailableClassrooms(N'Monday', 9, 10, 10, 10);
-- Checking when all classrooms are free
SELECT * FROM ypatel02.FindAvailableClassrooms(N'Monday', 14, 10, 15, 30);
-- Checking when occupied by classroom 4 (15-16:30)
SELECT * FROM ypatel02.FindAvailableClassrooms(N'Tuesday', 14, 10, 15, 15);
-- Checking when all classrooms are free
SELECT * FROM ypatel02.FindAvailableClassrooms(N'Tuesday', 9, 10, 10, 10);
```

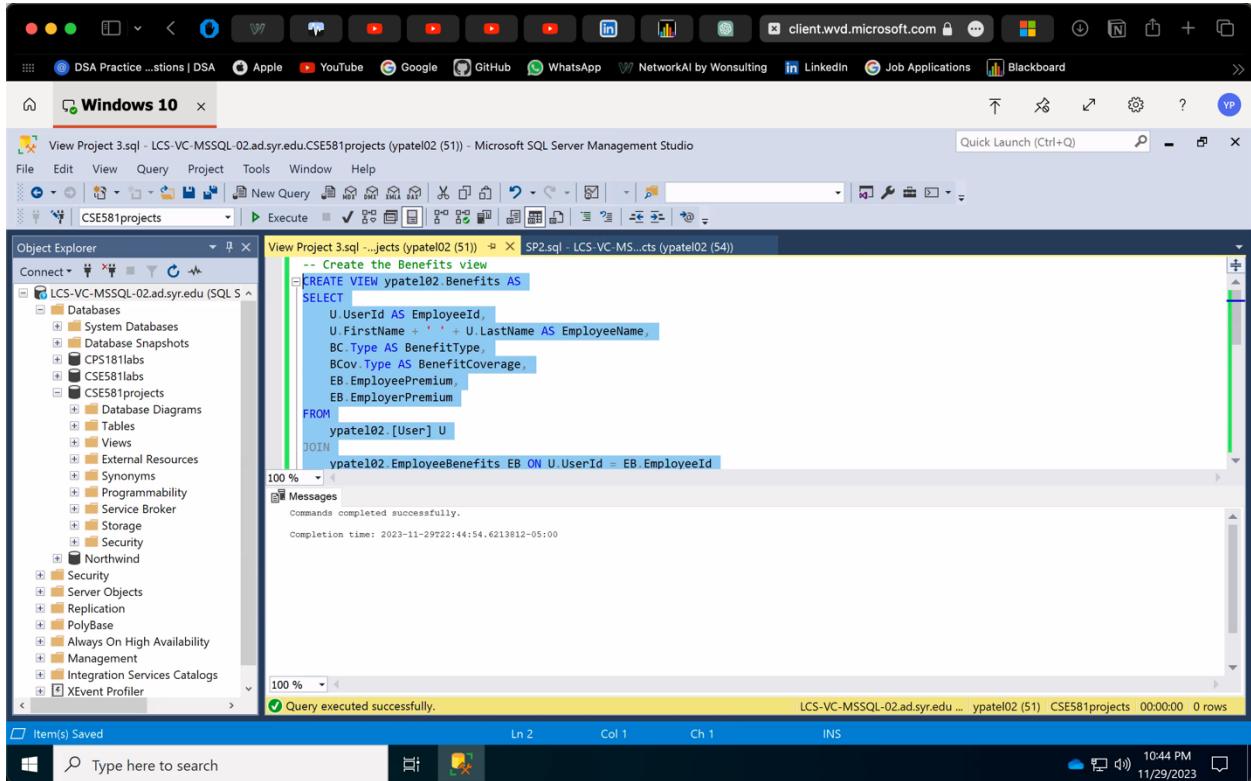
The results pane shows a table with the following data:

ClassRoomId
1
2
3
4
5
6
7
8
9
10

A status bar at the bottom indicates "Query executed successfully." and "LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) CSE581projects 00:00:00 8 rows".

You will also *create a view* (named as “Benefits”) (10% points) which can be viewed by “Graders” (GRANT SELECT ON SCHEMA::[yourSchema] TO Graders;) that shows every employee’s name, ID, benefit’s type, benefit coverage, employee premium and employer premium.

## Creating View as Described



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases, tables, and other objects. The central pane displays a T-SQL script for creating a view:

```
-- Create the Benefits view
CREATE VIEW ypatel02.Benefits AS
SELECT
    U.UserId AS EmployeeId,
    U.FirstName + ' ' + U.LastName AS EmployeeName,
    BC.Type AS BenefitType,
    BCov.Type AS BenefitCoverage,
    EB.EmployeePremium,
    EB.EmployerPremium
FROM
    ypatel02.[User] U
JOIN
    ypatel02.EmployeeBenefits EB ON U.UserId = EB.EmployeeId
```

The status bar at the bottom indicates "Query executed successfully." and provides details about the execution time and rows affected.

## Execution of Benefits View

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface running on a Windows 10 desktop. The title bar reads "Windows 10" and "View Project 3.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (51)) - Microsoft SQL Server Management Studio". The main window displays a query results grid titled "Messages".

The query executed was:

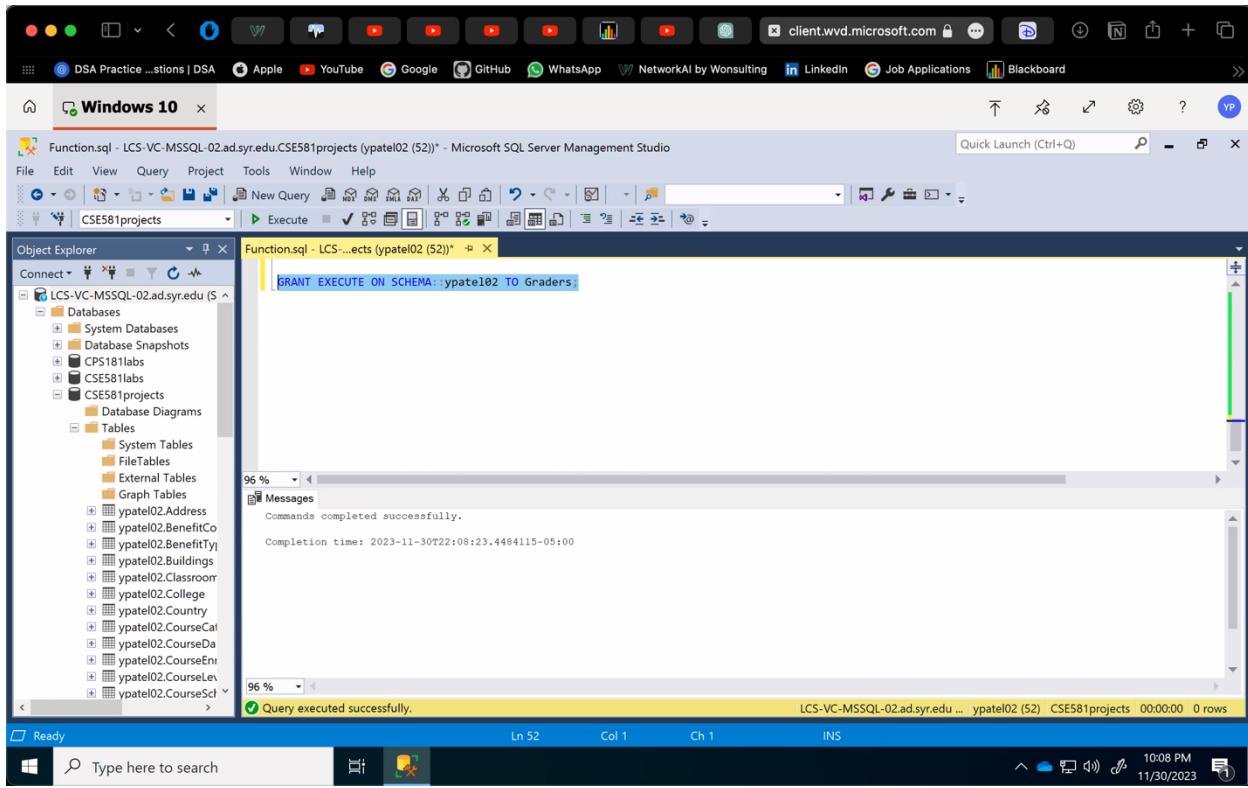
```
SELECT * FROM ypatel02.Benefits;
```

The results show 10 rows of data:

	EmployeeID	EmployeeName	BenefitType	BenefitCoverage	EmployeePremium	EmployerPremium
1	24	Yash Patel	Medical	100.00	400.00	
2	25	Anant Rajpurohit	Dental	120.00	410.00	
3	26	Shivam Patel	Vision	90.00	350.00	
4	27	Kartik Shah	Life	90.00	380.00	
5	28	Preet Shah	Disability	110.00	420.00	
6	29	Sohan Thakur	Retirement	95.00	400.00	
7	30	Shubham Desai	Paid Time Off	75.00	320.00	
8	31	Ruthi Shah	Flexible Spending Account	85.00	340.00	
9	32	Jay Ganatra	Health Savings Account	65.00	280.00	
10	33	Mirin Sharma	Education Assistance	70.00	290.00	

Message bar at the bottom: "Query executed successfully."

## GRANTING ACCESS TO GRADERS



Function.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (52)) - Microsoft SQL Server Management Studio

```
GRANT EXECUTE ON SCHEMA::ypatel02 TO Graders;
```

Messages

Commands completed successfully.

Completion time: 2023-11-30T22:08:23.4484115-05:00

Query executed successfully.

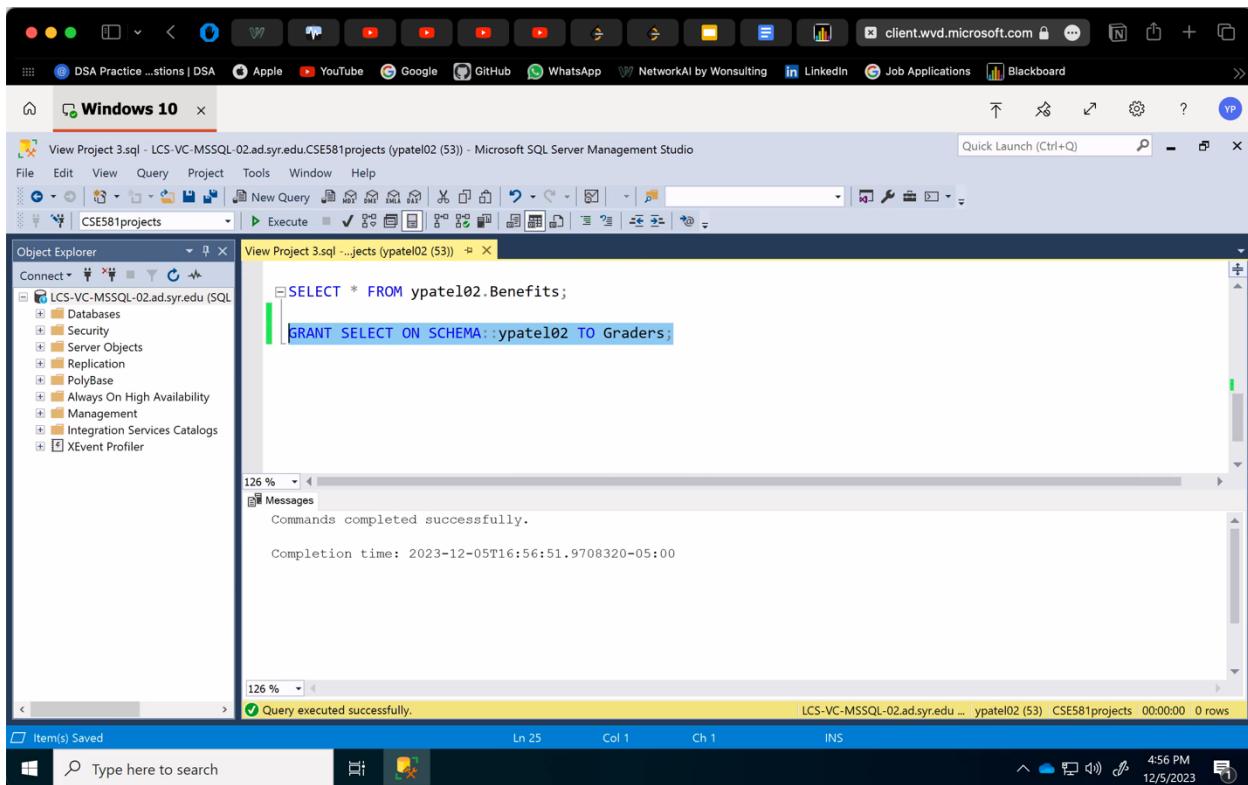
LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (52) | CSE581projects | 00:00:00 | 0 rows

Ready

Type here to search

10:08 PM 11/30/2023

This screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a database named 'CSE581projects' containing several tables under the 'Tables' node. In the center, a query window displays the command 'GRANT EXECUTE ON SCHEMA::ypatel02 TO Graders;'. The status bar at the bottom indicates the command was completed successfully at 10:08 PM on 11/30/2023.



View Project 3.sql - LCS-VC-MSSQL-02.ad.syr.edu.CSE581projects (ypatel02 (53)) - Microsoft SQL Server Management Studio

```
SELECT * FROM ypatel02.Benefits;
```

```
GRANT SELECT ON SCHEMA::ypatel02 TO Graders;
```

Messages

Commands completed successfully.

Completion time: 2023-12-05T16:56:51.9708320-05:00

Query executed successfully.

LCS-VC-MSSQL-02.ad.syr.edu ... ypatel02 (53) | CSE581projects | 00:00:00 | 0 rows

item(s) Saved

Type here to search

4:56 PM 12/05/2023

This screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a database named 'CSE581projects' containing various objects like 'Databases', 'Security', and 'Management'. In the center, a query window displays the command 'GRANT SELECT ON SCHEMA::ypatel02 TO Graders;'. The status bar at the bottom indicates the command was completed successfully at 4:56 PM on 12/05/2023.