|  |  |
| --- | --- |
| **Cognizant Digital Nurture 4.0: Deep Skilling** | |
| Name: Rajatesh Paul | Superset ID: |
| Week: 02 | Advances SQL (Exercise: Stored procedure) |

**Exercise 1:** Create a Stored Procedure

Goal: Create a stored procedure to retrieve employee details by department.

Steps:

1. Define the stored procedure with a parameter for DepartmentID.

2. Write the SQL query to select employee details based on the DepartmentID.

3. Create a stored procedure named `sp\_InsertEmployee` with the following code:

CREATE PROCEDURE sp\_InsertEmployee @FirstName VARCHAR(50), @LastName VARCHAR(50), @DepartmentID INT, @Salary DECIMAL(10,2), @JoinDate DATE AS BEGIN INSERT INTO Employees (FirstName, LastName, DepartmentID, Salary, JoinDate) VALUES (@FirstName, @LastName, @DepartmentID, @Salary, @JoinDate); END;

**Code:**

IF OBJECT\_ID('sp\_GetEmployeesByDepartment', 'P') IS NOT NULL

DROP PROCEDURE sp\_GetEmployeesByDepartment;

GO

CREATE PROCEDURE sp\_GetEmployeesByDepartment

@DepartmentID INT

AS

BEGIN

SELECT EmployeeID, FirstName, LastName, Salary, JoinDate

FROM Employees

WHERE DepartmentID = @DepartmentID;

END;

GO

EXEC sp\_GetEmployeesByDepartment @DepartmentID = 2;

GO

**Output:**

****

**Exercise 2:** Modify a Stored Procedure

Goal: Modify the stored procedure to include employee salary in the result.

Steps:

1. Open the existing stored procedure.

2. Add the Salary column to the SELECT statement.

3. Save the changes by executing the Stored procedure content.

**Code:**

IF OBJECT\_ID('sp\_GetEmployeesByDepartment', 'P') IS NOT NULL

DROP PROCEDURE sp\_GetEmployeesByDepartment;

GO

-- Step 2: Create the initial version of the procedure (without Salary)

CREATE PROCEDURE sp\_GetEmployeesByDepartment

@DepartmentID INT

AS

BEGIN

SELECT EmployeeID, FirstName, LastName, JoinDate

FROM Employees

WHERE DepartmentID = @DepartmentID;

END;

GO

-- Optional: Test the initial version

EXEC sp\_GetEmployeesByDepartment @DepartmentID = 1;

GO

-- Step 3: Modify the procedure to include Salary in the SELECT

ALTER PROCEDURE sp\_GetEmployeesByDepartment

@DepartmentID INT

AS

BEGIN

SELECT EmployeeID, FirstName, LastName, Salary, JoinDate

FROM Employees

WHERE DepartmentID = @DepartmentID;

END;

GO

-- Step 4: Test the modified version

EXEC sp\_GetEmployeesByDepartment @DepartmentID = 2;

GO

**Output:**

****

**Exercise 3:** Delete a Stored Procedure

Goal: Delete the stored procedure created in Exercise 1.

Steps:

1. Write the SQL command to delete the stored procedure.

2. Execute the command.

**Code:**

-- Make sure you're in the correct database

USE EmployeeDB;

GO

-- Step: Drop the stored procedure if it exists

IF OBJECT\_ID('sp\_GetEmployeesByDepartment', 'P') IS NOT NULL

BEGIN

DROP PROCEDURE sp\_GetEmployeesByDepartment;

PRINT 'Stored procedure sp\_GetEmployeesByDepartment has been deleted.';

END

ELSE

BEGIN

PRINT 'Stored procedure sp\_GetEmployeesByDepartment does not exist.';

END;

GO

**Exercise 4:** Execute a Stored Procedure

Goal: Execute the stored procedure to retrieve employee details for a specific department.

Steps:

1. Write the SQL command to execute the stored procedure with a DepartmentID parameter.

2. Execute the command and review the results.

**Code:**

IF OBJECT\_ID('sp\_GetEmployeesByDepartment', 'P') IS NULL

BEGIN

EXEC('

CREATE PROCEDURE sp\_GetEmployeesByDepartment

@DepartmentID INT

AS

BEGIN

SELECT EmployeeID, FirstName, LastName, Salary, JoinDate

FROM Employees

WHERE DepartmentID = @DepartmentID;

END

');

PRINT 'Procedure recreated for testing.';

END

GO

EXEC sp\_GetEmployeesByDepartment @DepartmentID = 2;

GO

**Output:**



**Exercise 5:** Return Data from a Stored Procedure

Goal: Create a stored procedure that returns the total number of employees in a department.

Steps:

1. Define the stored procedure with a parameter for DepartmentID.

2. Write the SQL query to count the number of employees in the specified department.

3. Save the stored procedure by executing the Stored procedure content.

Make sure you're using the correct database

**Code:**

IF OBJECT\_ID('sp\_GetEmployeeCountByDepartment', 'P') IS NOT NULL

DROP PROCEDURE sp\_GetEmployeeCountByDepartment;

GO

CREATE PROCEDURE sp\_GetEmployeeCountByDepartment

@DepartmentID INT

AS

BEGIN

SELECT COUNT(\*) AS EmployeeCount

FROM Employees

WHERE DepartmentID = @DepartmentID;

END;

GO

EXEC sp\_GetEmployeeCountByDepartment @DepartmentID = 3;

GO

**Output:**

****

**Exercise 6:** Use Output Parameters in a Stored Procedure

Goal: Create a stored procedure that returns the total salary of employees in a department using an output parameter.

Steps:

1. Define the stored procedure with a parameter for DepartmentID and an output parameter for total salary.

2. Write the SQL query to calculate the total salary of employees in the specified department.

3. Save the stored procedure by executing the Stored procedure content.

**Code:**

USE EmployeeDB;

GO

IF OBJECT\_ID('sp\_GetTotalSalaryByDepartment', 'P') IS NOT NULL

DROP PROCEDURE sp\_GetTotalSalaryByDepartment;

GO

CREATE PROCEDURE sp\_GetTotalSalaryByDepartment

@DepartmentID INT,

@TotalSalary DECIMAL(18,2) OUTPUT

AS

BEGIN

SELECT @TotalSalary = SUM(Salary)

FROM Employees

WHERE DepartmentID = @DepartmentID;

END;

GO

DECLARE @SalaryTotal DECIMAL(18,2);

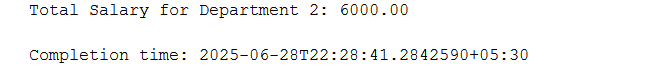
EXEC sp\_GetTotalSalaryByDepartment

@DepartmentID = 3,

@TotalSalary = @SalaryTotal OUTPUT;

PRINT 'Total Salary for Department 3: ' + CAST(@SalaryTotal AS VARCHAR);

GO

**Output:  
**