**Exercise 1: Ranking and Window Functions**

Goal: Use ROW\_NUMBER(), RANK(), DENSE\_RANK(), OVER(), and PARTITION BY.

Scenario:

Find the top 3 most expensive products in each category using different ranking functions.

Steps:

1. Use ROW\_NUMBER() to assign a unique rank within each category.

SELECT \*

FROM (

SELECT ProductID, ProductName, Category, Price,

ROW\_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS row\_num

FROM Products

) ranked

WHERE row\_num <= 3;

2. Use RANK() and DENSE\_RANK() to compare how ties are handled.

SELECT \*

FROM (

SELECT ProductID, ProductName, Category, Price,

RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS price\_rank

FROM Products

) ranked

WHERE price\_rank <= 3;

3. Use PARTITION BY Category and ORDER BY Price DESC.

SELECT \*

FROM (

SELECT ProductID, ProductName, Category, Price,

DENSE\_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS dense\_price\_rank

FROM Products

) ranked

WHERE dense\_price\_rank <= 3;

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.

CREATE PROCEDURE sp\_GetEmployeesByDepartment

@DepartmentID INT

AS

BEGIN

SELECT

EmployeeID,

FirstName,

LastName,

DepartmentID,

Salary,

JoinDate

FROM Employees

WHERE DepartmentID = @DepartmentID;

END;

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

SELECT

EmployeeID,

FirstName,

LastName,

DepartmentID,

Salary,

JoinDate

FROM Employees

WHERE DepartmentID = @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;

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.

CREATE PROCEDURE sp\_CountEmployeesByDepartment

@DepartmentID INT

AS

BEGIN

SELECT COUNT(\*) AS TotalEmployees

FROM Employees

WHERE DepartmentID = @DepartmentID;

END;

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

SELECT COUNT(\*) AS TotalEmployees

FROM Employees

WHERE DepartmentID = @DepartmentID;

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

CREATE PROCEDURE sp\_CountEmployeesByDepartment

@DepartmentID INT

AS

BEGIN

SELECT COUNT(\*) AS TotalEmployees

FROM Employees

WHERE DepartmentID = @DepartmentID;

END;