**Week2:**

**Advanced SQL:**

**1.SQL Exercises - Advanced Concepts:**

**1.Exercise 1: Rankings and Window Functions:**

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

**Query:**

SELECT

Category,

ProductName,

Price,

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

FROM Products;

**Output:**

| Category | ProductName | Price | RowNum |
| --- | --- | --- | --- |
| **Clothing** | **Jacket** | **100** | **1** |
| **Clothing** | **Jeans** | **40** | **2** |
| **Clothing** | **T-Shirt** | **20** | **3** |
| **Electronics** | **Laptop** | **1200** | **1** |
| **Electronics** | **Phone** | **800** | **2** |
| **Electronics** | **Tablet** | **600** | **3** |
| **Furniture** | **Desk** | **300** | **1** |
| **Furniture** | **Shelf** | **150** | **2** |
| **Furniture** | **Chair** | **120** | **3** |

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

**Query:**

SELECT

Category,

ProductName,

Price,

RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS RankNum,

DENSE\_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS DenseRankNum

FROM Products;

**Output:**

| Category | ProductName | Price | RankNum | DenseRankNum |
| --- | --- | --- | --- | --- |
| **Clothing** | **Jacket** | **100** | **1** | **1** |
| **Clothing** | **Jeans** | **40** | **2** | **2** |
| **Clothing** | **T-Shirt** | **20** | **3** | **3** |
| **Electronics** | **Laptop** | **1200** | **1** | **1** |
| **Electronics** | **Phone** | **800** | **2** | **2** |
| **Electronics** | **Tablet** | **600** | **3** | **3** |
| **Furniture** | **Desk** | **300** | **1** | **1** |
| **Furniture** | **Shelf** | **150** | **2** | **2** |
| **Furniture** | **Chair** | **120** | **3** | **3** |

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

**A. Using ROW\_NUMBER():**

**Query:**

SELECT \*

FROM (

SELECT

Category,

ProductName,

Price,

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

FROM Products

) ranked

WHERE RowNum <= 3;

**Output:**

| Category | ProductName | Price | RowNum |
| --- | --- | --- | --- |
| **Clothing** | **Jacket** | **100** | **1** |
| **Clothing** | **Jeans** | **40** | **2** |
| **Clothing** | **T-Shirt** | **20** | **3** |
| **Electronics** | **Laptop** | **1200** | **1** |
| **Electronics** | **Phone** | **800** | **2** |
| **Electronics** | **Tablet** | **600** | **3** |
| **Furniture** | **Desk** | **300** | **1** |
| **Furniture** | **Shelf** | **150** | **2** |
| **Furniture** | **Chair** | **120** | **3** |

**B. Using RANK():**

**Query:**

SELECT \*

FROM (

SELECT

Category,

ProductName,

Price,

RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS RankNum

FROM Products

) ranked

WHERE RankNum <= 3;

**Output:**

| Category | ProductName | Price | RankNum |
| --- | --- | --- | --- |
| **Clothing** | **Jacket** | **100** | **1** |
| **Clothing** | **Jeans** | **40** | **2** |
| **Clothing** | **T-Shirt** | **20** | **3** |
| **Electronics** | **Laptop** | **1200** | **1** |
| **Electronics** | **Phone** | **800** | **2** |
| **Electronics** | **Tablet** | **600** | **3** |
| **Furniture** | **Desk** | **300** | **1** |
| **Furniture** | **Shelf** | **150** | **2** |
| **Furniture** | **Chair** | **120** | **3** |

**C. Using DENSE\_RANK():**

**Query:**

SELECT \*

FROM (

SELECT

Category,

ProductName,

Price,

DENSE\_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS DenseRankNum

FROM Products

) ranked

WHERE DenseRankNum <= 3;

**Output:**

| Category | ProductName | Price | DenseRankNum |
| --- | --- | --- | --- |
| **Clothing** | **Jacket** | **100** | **1** |
| **Clothing** | **Jeans** | **40** | **2** |
| **Clothing** | **T-Shirt** | **20** | **3** |
| **Electronics** | **Laptop** | **1200** | **1** |
| **Electronics** | **Phone** | **800** | **2** |
| **Electronics** | **Tablet** | **600** | **3** |
| **Furniture** | **Desk** | **300** | **1** |
| **Furniture** | **Shelf** | **150** | **2** |
| **Furniture** | **Chair** | **120** | **3** |

**2.SQL Exercise: Stored Procedures:**

**Exercise 1: Create Stored Procedures:**

**1. Stored Procedure to Retrieve Employees by Department:**

**Query:**

DELIMITER $$

CREATE PROCEDURE sp\_GetEmployeesByDepartment(IN dept\_id INT)

BEGIN

SELECT

EmployeeID,

FirstName,

LastName,

DepartmentID,

Salary,

JoinDate

FROM Employees

WHERE DepartmentID = dept\_id;

END$$

DELIMITER ;

###CALL sp\_GetEmployeesByDepartment(3);

**Output:**

| **EmployeeID** | **FirstName** | **LastName** | **DepartmentID** | **Salary** | **JoinDate** |
| --- | --- | --- | --- | --- | --- |
| **3** | **Michael** | **Johnson** | **3** | **7000.00** | **2018-07-30** |
| **5\*** | **Sarah** | **Wilson** | **3** | **6500.00** | **2023-08-10** |

**2. Create a Stored Procedure to Insert a New Employee:**

**Query:**

DELIMITER $$

CREATE PROCEDURE sp\_InsertEmployee(

IN FirstName VARCHAR(50),

IN LastName VARCHAR(50),

IN DepartmentID INT,

IN Salary DECIMAL(10,2),

IN JoinDate DATE

)

BEGIN

INSERT INTO Employees (FirstName, LastName, DepartmentID, Salary, JoinDate)

VALUES (FirstName, LastName, DepartmentID, Salary, JoinDate);

END$$

DELIMITER ;

###CALL sp\_InsertEmployee('Sarah', 'Wilson', 3, 6500.00, '2023-08-10');

**Verify:**

SELECT \* FROM Employees WHERE LastName = 'Wilson';

**Output:**

| **EmployeeID** | **FirstName** | **LastName** | **DepartmentID** | **Salary** | **JoinDate** |
| --- | --- | --- | --- | --- | --- |
| **5** | **Sarah** | **Wilson** | **3** | **6500.00** | **2023-08-10** |

**Exercise 5:Return data from a stored procedure:**

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

**Query:**

DELIMITER $$

CREATE PROCEDURE sp\_CountEmployeesByDepartment(IN dept\_id INT)

BEGIN

SELECT COUNT(\*) AS EmployeeCount

FROM Employees

WHERE DepartmentID = dept\_id;

END$$

DELIMITER ;

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

**Query:**

SELECT COUNT(\*) AS EmployeeCount

FROM Employees

WHERE DepartmentID = dept\_id;

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

**Query:**

CALL sp\_CountEmployeesByDepartment(3);

**Output:**

| **EmployeeCount** |
| --- |
| 1 |