Exercise 1: Ranking and Window Functions

Goal: Use ROW_NUMBER(), RANK(), DENSE_RANK(), OVER(), and PARTITION BY

Solution

```
CREATE TABLE products (
    product_id INT PRIMARY KEY,
    product_name VARCHAR(100),
    category VARCHAR(50),
    price DECIMAL(10, 2)
);
```

```
INSERT INTO products (product_id, product_name, category, price) VALUES
(1, 'Laptop A', 'Electronics', 1000.00),
(2, 'Laptop B', 'Electronics', 950.00),
(3, 'Laptop C', 'Electronics', 1000.00),
(4, 'Phone A', 'Electronics', 500.00),
(5, 'T-shirt A', 'Clothing', 30.00),
(6, 'T-shirt B', 'Clothing', 25.00),
(7, 'T-shirt C', 'Clothing', 30.00),
(8, 'Shoes A', 'Clothing', 60.00),
(9, 'Chair A', 'Furniture', 150.00),
(10, 'Table A', 'Furniture', 300.00),
(11, 'Sofa A', 'Furniture', 300.00),
(12, 'Lamp A', 'Furniture', 100.00);
```

Results | Messages

	product_id	product_name	category	price
1	1	Laptop A	Electronics	1000.00
2	2	Laptop B	Electronics	950.00
3	3	Laptop C	Electronics	1000.00
4	4	Phone A	Electronics	500.00
5	5	T-shirt A	Clothing	30.00
6	6	T-shirt B	Clothing	25.00
7	7	T-shirt C	Clothing	30.00
8	8	Shoes A	Clothing	60.00
9	9	Chair A	Furniture	150.00
10	10	Table A	Furniture	300.00
11	11	Sofa A	Furniture	300.00
12	12	Lamp A	Furniture	100.00

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 *,
        ROW_NUMBER() OVER (PARTITION BY category ORDER BY price DESC) AS
row_num
   FROM products
) AS ranked
WHERE row_num <= 3;
```

Results Messages

	product_id	product_name	category	price	row_num
1	8	Shoes A	Clothing	60.00	1
2	5	T-shirt A	Clothing	30.00	2
3	7	T-shirt C	Clothing	30.00	3
4	1	Laptop A	Electronics	1000.00	1
5	3	Laptop C	Electronics	1000.00	2
6	2	Laptop B	Electronics	950.00	3
7	10	Table A	Furniture	300.00	1
8	11	Sofa A	Furniture	300.00	2
9	9	Chair A	Furniture	150.00	3

2.Use RANK() and DENSE_RANK() to compare how ties are handled

```
SELECT *
FROM (
   SELECT *,
        RANK() OVER (PARTITION BY category ORDER BY price DESC) AS rank_num
   FROM products
) AS ranked
WHERE rank_num <= 3;
```

Results Messages

	product_id	product_name	category	price	rank_num
1	8	Shoes A	Clothing	60.00	1
2	5	T-shirt A	Clothing	30.00	2
3	7	T-shirt C	Clothing	30.00	2
4	1	Laptop A	Electronics	1000.00	1
5	3	Laptop C	Electronics	1000.00	1
6	2	Laptop B	Electronics	950.00	3
7	10	Table A	Furniture	300.00	1
8	11	Sofa A	Furniture	300.00	1
9	9	Chair A	Furniture	150.00	3

3.

```
SELECT *
FROM (
   SELECT *,
        DENSE_RANK() OVER (PARTITION BY category ORDER BY price DESC) AS
dense_rank_num
   FROM products
) AS ranked
WHERE dense_rank_num <= 3;
```

Results Messages

	product_id	product_name	category	price	dense_rank_num
1	8	Shoes A	Clothing	60.00	1
2	5	T-shirt A	Clothing	30.00	2
3	7	T-shirt C	Clothing	30.00	2
4	6	T-shirt B	Clothing	25.00	3
5	1	Laptop A	Electronics	1000.00	1
6	3	Laptop C	Electronics	1000.00	1
7	2	Laptop B	Electronics	950.00	2
8	4	Phone A	Electronics	500.00	3
9	10	Table A	Furniture	300.00	1
10	11	Sofa A	Furniture	300.00	1
11	9	Chair A	Furniture	150.00	2
12	12	Lamp A	Furniture	100.00	3

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

Solution

```
CREATE TABLE Departments (
    DepartmentID INT PRIMARY KEY,
    DepartmentName VARCHAR(100)
);
```

```
CREATE TABLE Employees (
    EmployeeID INT PRIMARY KEY IDENTITY(1,1),  -- Auto-increment ID
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    DepartmentID INT FOREIGN KEY REFERENCES Departments(DepartmentID),
    Salary DECIMAL(10,2),
    JoinDate DATE
);
```

```
INSERT INTO Departments (DepartmentID, DepartmentName) VALUES
(1, 'HR'),
(2, 'Finance'),
(3, 'IT'),
(4, 'Marketing');
```

	DepartmentID	DepartmentName
1	1	HR
2	2	Finance
3	3	IT
4	4	Marketing

```
INSERT INTO Employees (FirstName, LastName, DepartmentID, Salary, JoinDate) VALUES
('John', 'Doe', 1, 5000.00, '2020-01-15'),
('Jane', 'Smith', 2, 6000.00, '2019-03-22'),
('Michael', 'Johnson', 3, 7000.00, '2018-07-30'),
('Emily', 'Davis', 4, 5500.00, '2021-11-05');
```

	EmployeeID	FirstName	LastName	DepartmentID	Salary	JoinDate
1	1	John	Doe	1	5000.00	2020-01-15
2	2	Jane	Smith	2	6000.00	2019-03-22
3	3	Michael	Johnson	3	7000.00	2018-07-30
4	4	Emily	Davis	4	5500.00	2021-11-05

```
CREATE PROCEDURE sp_GetEmployeesByDepartment
   @DeptID INT
AS
BEGIN
   SELECT
      e.EmployeeID,
      e.FirstName,
      e.LastName,
      d.DepartmentName,
      e.Salary,
      e.JoinDate
   FROM Employees e
   INNER JOIN Departments d ON e.DepartmentID = d.DepartmentID
   WHERE e.DepartmentID = @DeptID;
END;


EXEC sp_GetEmployeesByDepartment @DeptID = 1;
```

	EmployeeID	FirstName	LastName	DepartmentName	Salary	JoinDate
1	1	John	Doe	HR	5000.00	2020-01-15

```
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;
```

```
EXEC sp_InsertEmployee
    @FirstName = 'Robert',
    @LastName = 'King',
    @DepartmentID = 2,
    @Salary = 6200.00,
    @JoinDate = '2022-06-15';
```

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.

```
CREATE PROCEDURE sp_GetEmployeeCountByDepartment
    @DepartmentID INT
AS
BEGIN
    SELECT
        COUNT(*) AS EmployeeCount
    FROM
        Employees
    WHERE
        DepartmentID = @DepartmentID;
END;
```

```
EXEC sp_GetEmployeeCountByDepartment @DepartmentID = 3;
```