## **Assignment No: 4**

Name: Pratik Wani

#### SQL Queries

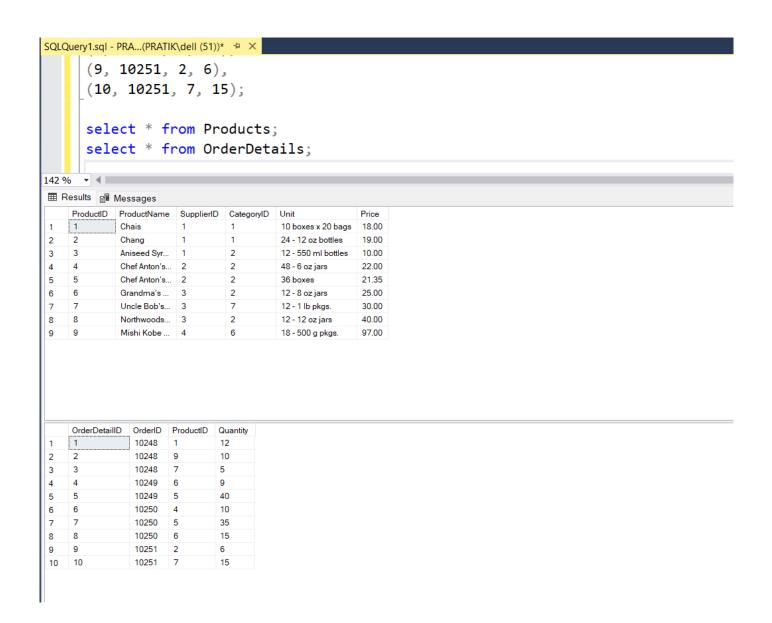
- Here I created a database named as 'Practice' and table named as 'Products' and 'OrderDetails' performed all the Join types.
- Create database, Create tables and Inserted Data:
  - Create is a DDL command
  - Creating a database doesn't automatically set it as the active database, so we need USE command
  - While Creating Table we need to give all the column names there data types and constraints

```
SQLQuery1.sql - PRA...(PRATIK\dell (51))* 😕 🔀
    ----Assignment 3
    use practice;
    -- Create the Products table
   □CREATE TABLE Products (
        ProductID INT PRIMARY KEY,
        ProductName NVARCHAR(255),
        SupplierID INT,
        CategoryID INT,
        Unit NVARCHAR(50),
        Price DECIMAL(10, 2)
    );
    -- Insert data into the Products table
   垣 INSERT INTO Products (ProductID, ProductName, SupplierID, CategoryID, Unit, Price)
    VALUES
    (1, 'Chais', 1, 1, '10 boxes x 20 bags', 18),
    (2, 'Chang', 1, 1, '24 - 12 oz bottles', 19),
    (3, 'Aniseed Syrup', 1, 2, '12 - 550 ml bottles', 10),
    (4, 'Chef Anton''s Cajun Seasoning', 2, 2, '48 - 6 oz jars', 22),
    (5, 'Chef Anton''s Gumbo Mix', 2, 2, '36 boxes', 21.35),
    (6, 'Grandma''s Boysenberry Spread', 3, 2, '12 - 8 oz jars', 25),
    (7, 'Uncle Bob''s Organic Dried Pears', 3, 7, '12 - 1 lb pkgs.', 30),
     (8, 'Northwoods Cranberry Sauce', 3, 2, '12 - 12 oz jars', 40),
     (9, 'Mishi Kobe Niku', 4, 6, '18 - 500 g pkgs.', 97);
```

```
SQLQuery1.sql - PRA...(PRATIK\dell (51))* 垣 🗶
    -- Create the OrderDetails table
   OrderDetailID INT PRIMARY KEY,
        OrderID INT,
        ProductID INT references products(productid),
        Quantity INT
    );
    -- Insert data into the OrderDetails table
   □INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity)
    VALUES
    (1, 10248, 1, 12),
    (2, 10248, 9, 10),
    (3, 10248, 7, 5),
    (4, 10249, 6, 9),
    (5, 10249, 5, 40),
    (6, 10250, 4, 10),
    (7, 10250, 5, 35),
    (8, 10250, 6, 15),
    (9, 10251, 2, 6),
    (10, 10251, 7, 15);
```

#### Select:

Select \* gives all records



### Exists:

- The EXISTS operator is used to test for the existence of any record in a subquery.
- The EXISTS operator returns TRUE if the subquery returns one or more records

```
SQLassg4 sql.sql -...e (PRATIK\dell (84))* + ×
      (10, 10251, 7, 15);
      select * from Products;
      select * from OrderDetails;
      --Exists
    SELECT ProductName FROM Products
      WHERE exists (SELECT ProductID FROM OrderDetails WHERE Quantity=15);
ProductName
   Chang
   Aniseed Syrup
    Chef Anton's Cajun Seasoning
   Chef Anton's Gumbo Mix
   Grandma's Boysenberry Spread
   Uncle Bob's Organic Dried Pears
    Northwoods Cranberry Sauce
   Mishi Kobe Niku
```

### ■ Any:

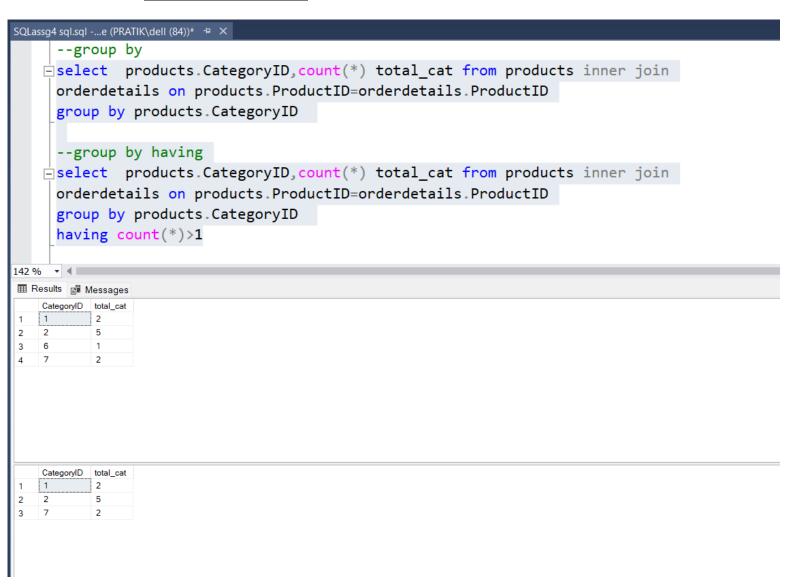
- o returns a boolean value as a result
- o returns TRUE if ANY of the subquery values meet the condition

```
SQLassg4 sql.sql -...e (PRATIK\dell (84))* 🕆 🗶
    --1)
    SELECT ProductName, productid
     FROM Products WHERE ProductID = ANY (SELECT ProductID FROM OrderDetails WHERE Quantity=15);
      --2)
    SELECT ProductName, ProductID
     FROM Products
     WHERE ProductID=ANY(SELECT ProductID FROM OrderDetails WHERE Quantity>12);
     --3)
    SELECT ProductName
     FROM Products
     WHERE ProductID=ANY(SELECT ProductID FROM OrderDetails WHERE Quantity > 1000);
142 %
ProductName
                      productid
   Grandma's Boysenberry Spread 6
   Uncle Bob's Organic Dried Pears 7
   ProductName
                      ProductID
   Chef Anton's Gumbo Mix
    Grandma's Boysenberry Spread
   Uncle Bob's Organic Dried Pears 7
   ProductName
```

### All:

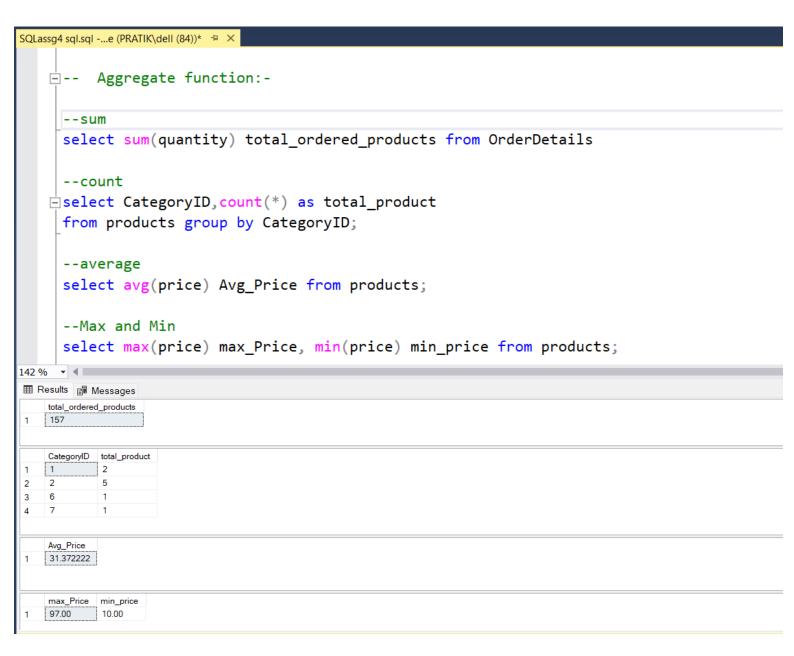
- O Returns a boolean value as a result
- O Returns TRUE if ALL of the subquery values meet the condition
- o Is used with SELECT, WHERE and HAVING statements

• Group By and Having:



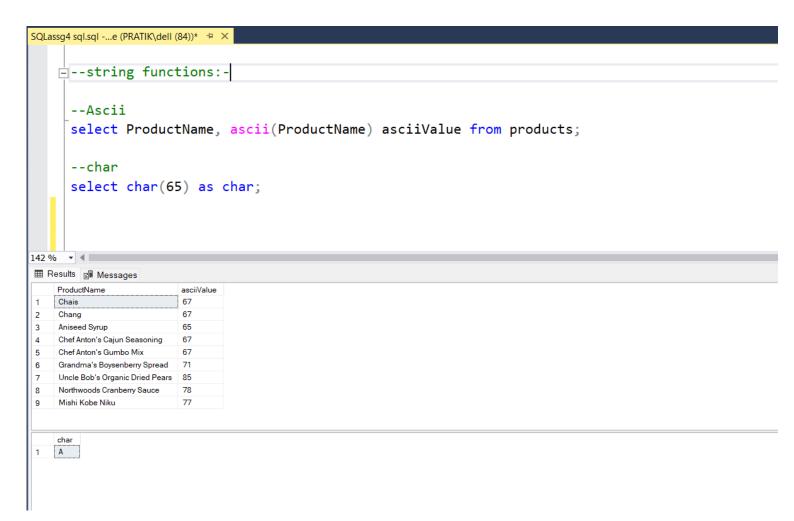
### Aggregate functions:

- Sum: Returns the Addition of the values
- o Count: Returns the Total count used with the group by statement
- o Avg: Returns the Average value of the values
- Max: Returns the Max values
- o Min: Return the Min values

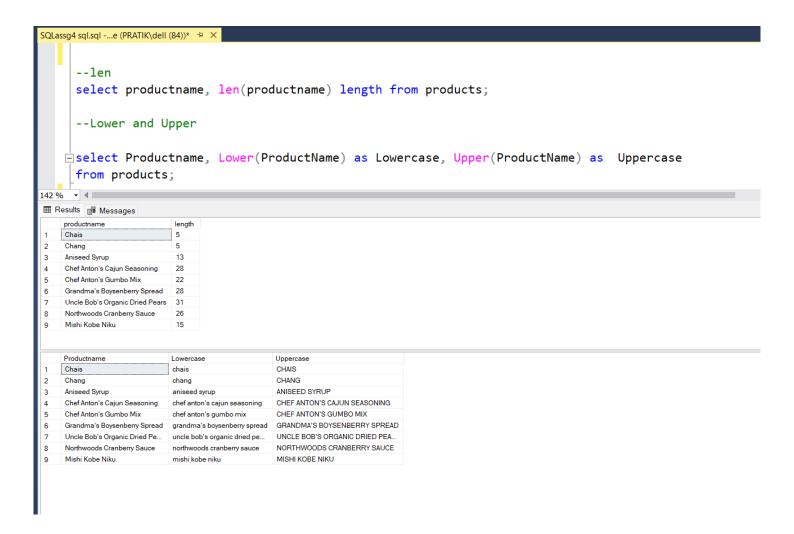


# String functions:

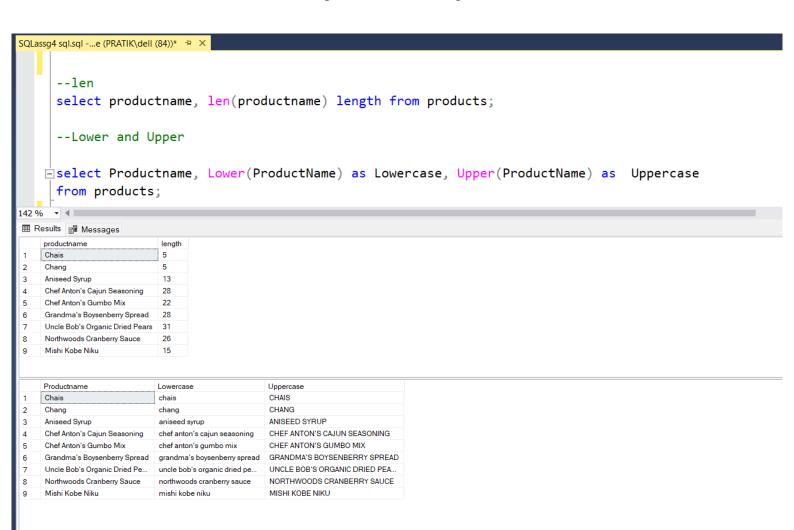
- o Ascii: Returns the Ascii value for the column or value which we pass.
- Char: Char is just opposite of Ascii. It returns the char associated with the ascii
   number



- Len: Returns the length of strings
- Lower: Convert the string into lower case
- Upper: Converts the string into upper case

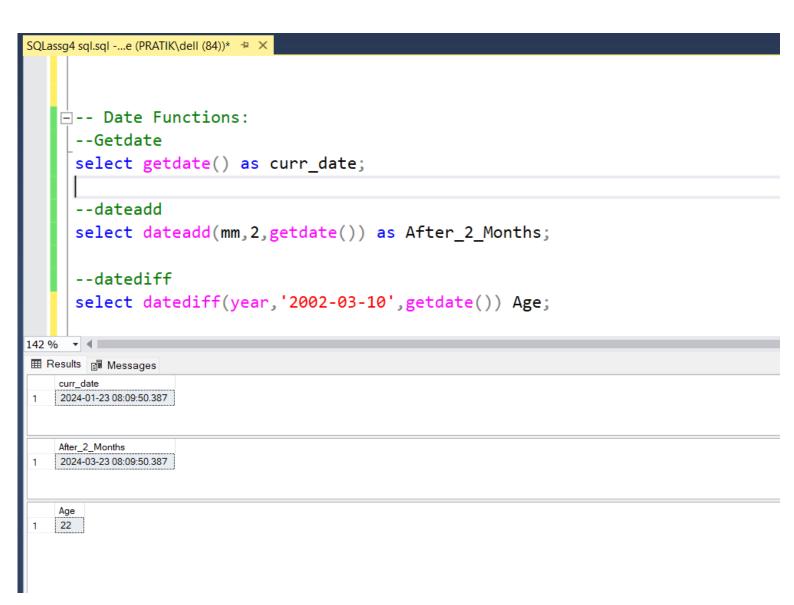


- o Reverse: Reverse the strings chars
- Str: Convert the integer values into strings

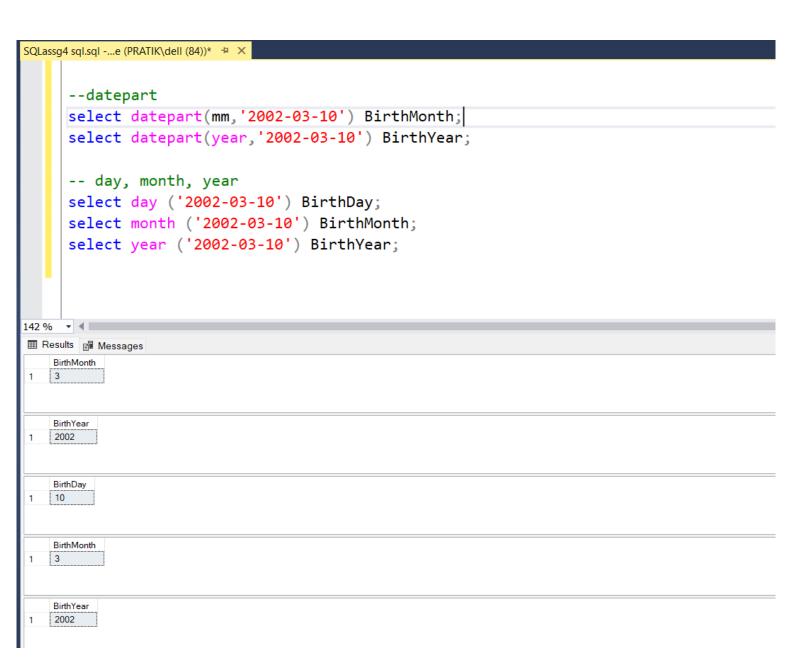


### Date Functions:

- o Getdate: Returns the current date
- o Dateadd: Add months to existed date
- o Datediff: It will return the difference of date, months, years.

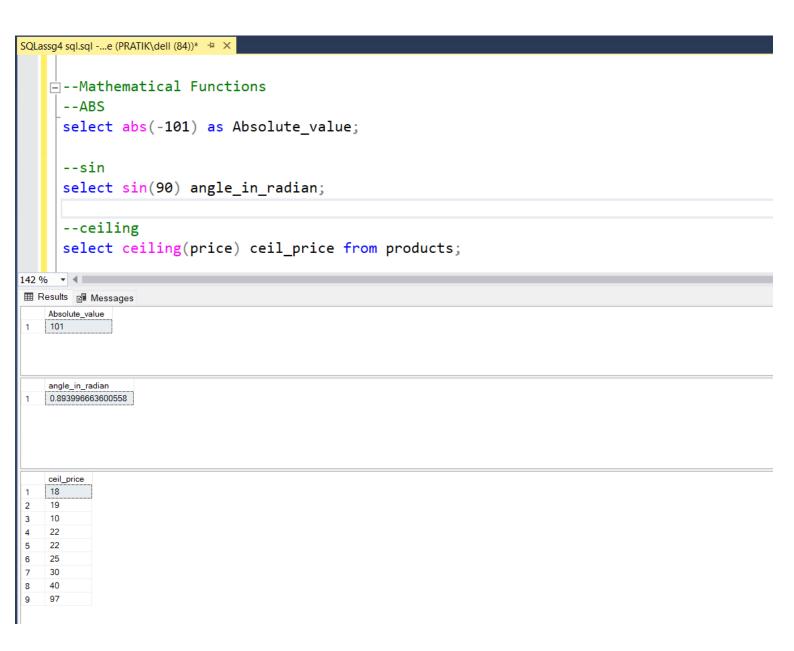


- o Datepart: Return the Specific part of the date
- o Month: Return the month from date
- o Year: Return the year from date
- o Day: Return the day from date

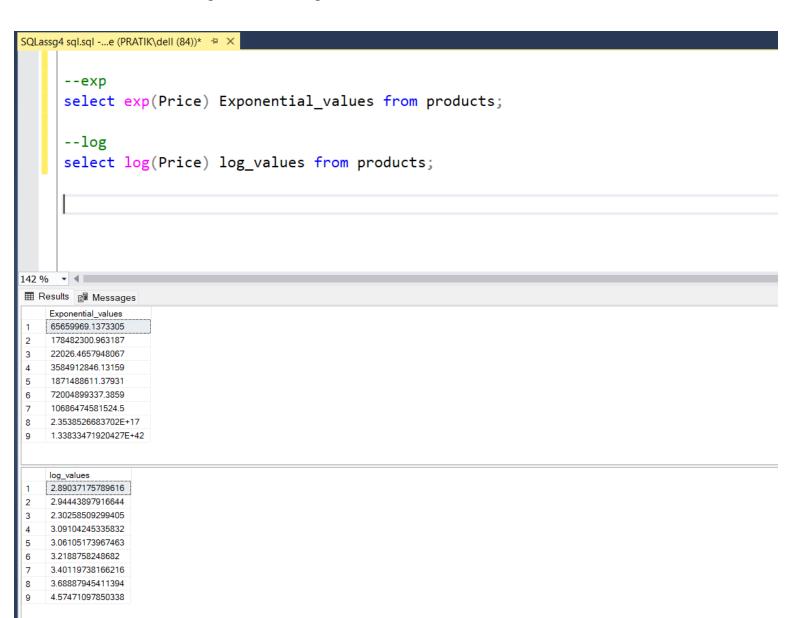


## Mathematical Functions:

- O ABS: Returns the absolute values
- O Sin: Returns the value of angle in radian
- O Ceiling: Returns the greatest to the specified value

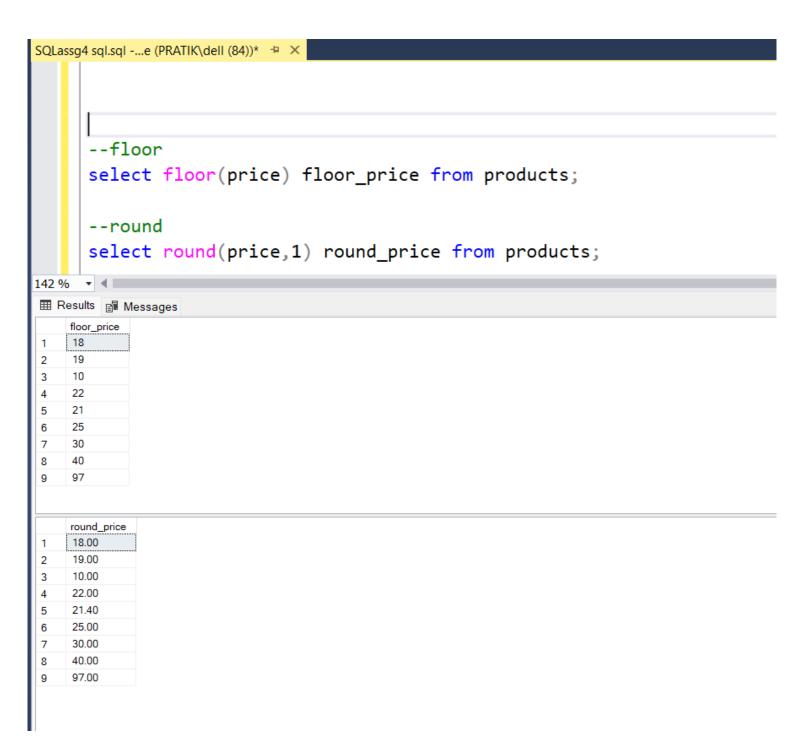


- o Exp: Returns the Exponential values
- o Log: Returns the logarithmic values



o Floor: Returns the smallest to the specified value

o Round: Round of the values



# Nested Subquery:

- o Query inside Query means subquery
- A subquery can be nested inside other subqueries which is called nested subquery.

