# Answer following questions

1. What is a result set?

A result set is a set of data returned by a select statement.

1. What is the difference between Union and Union All?

Union removes duplicate rows whereas Union All does not.

1. What are the other Set Operators SQL Server has?

Intersect and Except.

1. What is the difference between Union and Join?

Union combines data vertically but Join combines data horizontally.

1. What is the difference between INNER JOIN and FULL JOIN?

A INNER JOIN only gives the intersection of two tables. A FULL JOIN returns all records in both tables.

1. What is difference between left join and outer join

A left join and a left outer join is the same.

1. What is cross join?

A cross join returns the cartesian product of two tables.

1. What is the difference between WHERE clause and HAVING clause?

Both are used to filter records. The HAVING clause can be used on aggregated results but the WHERE clause cannot.

1. Can there be multiple group by columns?

Yes. A group will contain the same values for the specified columns.

# Write queries for following scenarios

1. How many products can you find in the Production.Product table?

SELECT COUNT(\*) FROM Production.Product;

1. Write a query that retrieves the number of products in the Production.Product table that are included in a subcategory. The rows that have NULL in column ProductSubcategoryID are considered to not be a part of any subcategory.

SELECT COUNT(ProductID) FROM Production.Product WHERE ProductSubcategoryID IS NOT NULL;

1. How many Products reside in each SubCategory? Write a query to display the results with the following titles.

ProductSubcategoryID CountedProducts

-------------------- ---------------

SELECT ProductSubcategoryID, COUNT(ProductID) AS CountedProducts FROM Production.Product GROUP BY ProductSubcategoryID;

1. How many products that do not have a product subcategory.

SELECT COUNT(ProductID) FROM Production.Product WHERE ProductSubcategoryID IS NULL;

1. Write a query to list the sum of products quantity in the Production.ProductInventory table.

SELECT ProductID, SUM(Quantity) FROM Production.ProductInventory GROUP BY ProductID;

1. Write a query to list the sum of products in the Production.ProductInventory table and LocationID set to 40 and limit the result to include just summarized quantities less than 100.

ProductID TheSum

----------- ----------

SELECT ProductID, SUM(Quantity) AS TheSum FROM Production.ProductInventory WHERE LocationID = 40 GROUP BY ProductID HAVING SUM(Quantity) < 100;

1. Write a query to list the sum of products with the shelf information in the Production.ProductInventory table and LocationID set to 40 and limit the result to include just summarized quantities less than 100

Shelf ProductID TheSum

---------- ----------- -----------

SELECT Shelf, ProductID, SUM(Quantity) AS TheSum FROM Production.ProductInventory WHERE LocationID = 40 GROUP BY ProductID, Shelf HAVING SUM(Quantity) < 100;

1. Write the query to list the average quantity for products where column LocationID has the value of 10 from the table Production.ProductInventory table.

SELECT AVG(Quantity) FROM Production.ProductInventory WHERE LocationID = 10;

1. Write query to see the average quantity of products by shelf from the table Production.ProductInventory

ProductID Shelf TheAvg

----------- ---------- -----------

SELECT ProductID, Shelf, AVG(Quantity) AS TheAvg FROM Production.ProductInventory GROUP BY ProductID, Shelf;

1. Write query to see the average quantity of products by shelf excluding rows that has the value of N/A in the column Shelf from the table Production.ProductInventory

ProductID Shelf TheAvg

----------- ---------- -----------

SELECT ProductID, Shelf, AVG(Quantity) AS TheAvg FROM Production.ProductInventory GROUP BY ProductID, Shelf HAVING Shelf IS NOT NULL;

1. List the members (rows) and average list price in the Production.Product table. This should be grouped independently over the Color and the Class column. Exclude the rows where Color or Class are null.

Color Class TheCount AvgPrice

-------------- - ----- ----------- ---------------------

SELECT Color, Class, COUNT(ProductID) AS TheCount, AVG(ListPrice) AS AvgPrice FROM Production.Product WHERE Color IS NOT NULL AND Class IS NOT NULL GROUP BY Color, Class;

**Joins:**

1. Write a query that lists the country and province names from person. CountryRegion and person. StateProvince tables. Join them and produce a result set similar to the following.

Country Province

--------- ----------------------

SELECT c.Name AS Country, s.Name AS Province FROM Person.CountryRegion c JOIN Person.StateProvince s ON c.CountryRegionCode = s.CountryRegionCode;

1. Write a query that lists the country and province names from person. CountryRegion and person. StateProvince tables and list the countries filter them by Germany and Canada. Join them and produce a result set similar to the following.

Country Province

--------- ----------------------

SELECT c.Name AS Country, s.Name AS Province FROM Person.CountryRegion c JOIN Person.StateProvince s ON c.CountryRegionCode = s.CountryRegionCode WHERE c.Name IN ('Germany', 'Canada');

**Using Northwnd Database: (Use aliases for all the Joins)**

1. List all Products that has been sold at least once in last 25 years.

SELECT DISTINCT od.ProductID FROM Orders o JOIN [Order Details] od ON o.OrderID = od.OrderID WHERE YEAR(o.OrderDate) >= 1996;

1. List top 5 locations (Zip Code) where the products sold most.

SELECT TOP 5 o.ShipPostalCode FROM Orders o JOIN [Order Details] od ON o.OrderID = od.OrderID GROUP BY o.ShipPostalCode ORDER BY COUNT(od.ProductID) DESC;

1. List top 5 locations (Zip Code) where the products sold most in last 25 years.

SELECT TOP 5 o.ShipPostalCode FROM Orders o JOIN [Order Details] od ON o.OrderID = od.OrderID WHERE YEAR(o.OrderDate) >= 1996 GROUP BY o.ShipPostalCode ORDER BY COUNT(od.ProductID) DESC;

1. List all city names and number of customers in that city.

SELECT City, COUNT(CustomerID) FROM Customers GROUP BY City;

1. List city names which have more than 2 customers, and number of customers in that city

SELECT City, COUNT(CustomerID) FROM Customers GROUP BY City HAVING COUNT(CustomerID) > 2;

1. List the names of customers who placed orders after 1/1/98 with order date.

SELECT c.ContactName, o.OrderDate FROM Customers c JOIN Orders o ON c.CustomerID = o.CustomerID WHERE o.OrderDate >= CONVERT(datetime, '1998-01-01');

1. List the names of all customers with most recent order dates

SELECT c.ContactName, MAX(o.OrderDate) FROM Customers c JOIN Orders o ON c.CustomerID = o.CustomerID GROUP BY c.ContactName, c.CustomerID;

1. Display the names of all customers along with the count of products they bought

SELECT c.ContactName, SUM(od.Quantity) FROM Customers c JOIN Orders o ON c.CustomerID = o.CustomerID JOIN [Order Details] od ON o.OrderID = od.OrderID GROUP BY c.ContactName, c.CustomerID;

1. Display the customer ids who bought more than 100 Products with count of products.

SELECT c.ContactName, SUM(od.Quantity) FROM Customers c JOIN Orders o ON c.CustomerID = o.CustomerID JOIN [Order Details] od ON o.OrderID = od.OrderID GROUP BY c.ContactName, c.CustomerID;

1. List all of the possible ways that suppliers can ship their products. Display the results as below

Supplier Company Name Shipping Company Name

--------------------------------- ----------------------------------

SELECT su.CompanyName AS [Supplier Company Name], sh.CompanyName AS [Shipping Company Name] FROM Shippers sh CROSS JOIN Suppliers su;

1. Display the products order each day. Show Order date and Product Name.

SELECT o.OrderDate, p.ProductName FROM [Order Details] od JOIN Products p ON od.ProductID = p.ProductID JOIN Orders o ON od.OrderID = o.OrderID;

1. Displays pairs of employees who have the same job title.

SELECT e1.FirstName + ' ' + e1.LastName, e2.FirstName + ' ' + e2.LastName FROM Employees e1 CROSS JOIN Employees e2 WHERE e1.Title = e2.Title AND e1.EmployeeID != e2.EmployeeID;

1. Display all the Managers who have more than 2 employees reporting to them.

SELECT e3.FirstName + ' ' + e3.LastName FROM Employees e3 WHERE e3.EmployeeID IN (SELECT m.EmployeeID FROM Employees e INNER JOIN Employees m ON e.ReportsTo = m.EmployeeID GROUP BY m.EmployeeID HAVING COUNT(m.EmployeeID) > 2);

1. Display the customers and suppliers by city. The results should have the following columns

City

Name

Contact Name,

Type (Customer or Supplier)

SELECT City, CompanyName, ContactName, 'Customer' AS Type FROM Customers UNION SELECT City, CompanyName, ContactName, 'Supplier' AS Type FROM Suppliers;