# Answer following questions

1. What is a result set?

A result set is a set of rows that are generated post executing a query.

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

Union performs a deduplication step before returning the final results, Union All retains all the duplicates and returns the full, concatenated results.

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

Intersect – All distinct rows selected by both queries

Minus – All distinct rows selected by the first query but not the second

1. What is the difference between Union and Join?

Joins combine data into new columns.

Unions combine data into new rows.

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

Inner Join returns only matching rows between both the tables, non-matching rows are eliminated.

Full Join returns all the rows from both the tables(left and right), including non-matching rows from both the tables.

1. What is difference between left join and outer join

No difference, both are the same.

1. What is cross join?

A cross join returns the Cartesian product of the sets of records from the two joined tables. Thus, it equates to an inner join where the join-condition always evaluates to true.

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

WHERE CLAUSE –

* Used to filter the records from the tables based on specific conditions.
* Can be used without the GROUP BY CLAUSE.
* Implements in row operations.
* Cannot contain aggregate function.
* Can be used with Select, Update, Delete statement.
* Used before the GROUP BY CLAUSE.
* Used with single row function like UPPER, LOWER, etc.

HAVING CLAUSE –

* Used to filter the records from the groups based on specific conditions.
* Cannot be used without GROUP BY CLAUSE.
* Implements in column operations.
* Can contain aggregate function.
* Can only be used with a SELECT statement.
* Used after the GROUP BY CLAUSE.
* Used with multiple row functions like SUM, COUNT, etc.

1. Can there be multiple group by columns?

Yes

# Write queries for following scenarios

1. 295
2. SELECT COUNT(ProductSubcategoryID) FROM Production.Product;
3. SELECT ProductSubcategoryID, COUNT(ProductSubcategoryID) 'CountedProducts' FROM Production.Product GROUP BY ProductSubcategoryID;
4. SELECT COUNT(ProductID) FROM Production.Product WHERE ProductSubcategoryID IS NULL;
5. SELECT SUM(Quantity) FROM Production.ProductInventory;
6. SELECT ProductID, SUM(Quantity) 'TheSum' FROM Production.ProductInventory WHERE LocationID = 40 GROUP BY ProductID HAVING SUM(Quantity) < 100;
7. SELECT ProductID, Shelf, SUM(Quantity) 'TheSum' FROM Production.ProductInventory WHERE LocationID = 40 GROUP BY ProductID, Shelf HAVING SUM(Quantity) < 100;
8. SELECT AVG(Quantity) 'Average' FROM Production.ProductInventory WHERE LocationID = 10;
9. SELECT ProductID, Shelf, AVG(Quantity) 'TheAvg' FROM Production.ProductInventory GROUP BY ProductID, Shelf;
10. SELECT ProductID, Shelf, AVG(Quantity) 'TheAvg' FROM Production.ProductInventory GROUP BY ProductID, Shelf HAVING Shelf != 'N/A';
11. SELECT Color, Class, COUNT(ProductID) 'TheCount', AVG(ListPrice) 'AvgPrice' FROM Production.Product GROUP BY Color, Class HAVING (Color IS NOT NULL AND Class IS NOT NULL);
12. SELECT c.Name as 'Country', s.Name as 'Province' FROM Person.CountryRegion c INNER JOIN Person.StateProvince s on c.CountryRegionCode = s.CountryRegionCode;
13. SELECT c.Name as 'Country', s.Name as 'Province' FROM Person.CountryRegion c INNER JOIN Person.StateProvince s on c.CountryRegionCode = s.CountryRegionCode WHERE c.Name = 'Germany' OR c.Name = 'Canada';
14. SELECT p.ProductName FROM Products p INNER JOIN [Order Details] od ON p.ProductID = od.ProductID INNER JOIN Orders o ON o.OrderID = od.OrderID WHERE p.UnitsOnOrder > 0 AND o.OrderDate < DATEADD(year, -25, GETDATE());

17. SELECT City, COUNT(CustomerID) 'NoOfCustomers' from Customers GROUP BY City;

18. SELECT City, COUNT(CustomerID) from Customers GROUP BY City HAVING COUNT(CustomerID) > 10;

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

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

22. SELECT c.ContactName, COUNT(od.Quantity) FROM Customers c LEFT JOIN Orders o ON c.CustomerID = o.CustomerID LEFT JOIN [Order Details] od ON o.OrderID = od.OrderID GROUP BY ContactName HAVING COUNT(od.Quantity) > 100;

23.