

--1 . List of all customers

```
SELECT * FROM Sales.Customer;
```

--2.List of all customers where company name ends with 'N':

```
SELECT *  
FROM Sales.Store s  
JOIN Sales.Customer c ON s.BusinessEntityID = c.StoreID  
WHERE s.Name LIKE '%N';
```

--3-List of all customers who live in Berlin or London:

```
SELECT c.*  
FROM Sales.Customer c  
JOIN Person.BusinessEntityAddress bea ON c.CustomerID = bea.BusinessEntityID  
JOIN Person.Address a ON bea.AddressID = a.AddressID  
WHERE a.City IN ('Berlin', 'London');
```

--4-List of all customers who live in UK or USA

```
SELECT c.*  
FROM Sales.Customer c  
JOIN Person.BusinessEntityAddress bea ON c.CustomerID = bea.BusinessEntityID  
JOIN Person.Address a ON bea.AddressID = a.AddressID  
JOIN Person.StateProvince sp ON a.StateProvinceID = sp.StateProvinceID  
WHERE sp.CountryRegionCode IN ('UK', 'US');
```

--5-List of all products sorted by product name

```
SELECT Name  
FROM Production.Product  
ORDER BY Name;
```

--6-List of all products where product name starts with an A

```
SELECT ProductID, Name, ProductNumber, MakeFlag, FinishedGoodsFlag, Color,  
SafetyStockLevel, ReorderPoint, StandardCost, ListPrice, Size, SizeUnitMeasureCode,  
WeightUnitMeasureCode, Weight, DaysToManufacture, ProductLine, Class, Style,  
ProductSubcategoryID, ProductModelID, SellStartDate, SellEndDate, DiscontinuedDate,  
rowguid, ModifiedDate  
FROM Production.Product  
WHERE Name LIKE 'A%'  
ORDER BY Name;
```

--7-List of customers who ever placed an order

```
SELECT DISTINCT c.*
FROM Sales.Customer c
JOIN Sales.SalesOrderHeader soh ON c.CustomerID = soh.CustomerID;
```

--8-List of customers who live in london and have bought chai

```
SELECT DISTINCT c.*
FROM Sales.Customer c
JOIN Sales.SalesOrderHeader soh ON c.CustomerID = soh.CustomerID
JOIN Sales.SalesOrderDetail sod ON soh.SalesOrderID = sod.SalesOrderID
JOIN Production.Product p ON sod.ProductID = p.ProductID
JOIN Person.BusinessEntityAddress bea ON soh.ShipToAddressID = bea.BusinessEntityID
JOIN Person.Address a ON bea.AddressID = a.AddressID
WHERE a.City = 'London' AND p.Name = 'Chai';
```

--9-List of customer who never placed an order

```
SELECT * FROM Sales.Customer c
WHERE NOT EXISTS (
    SELECT 1
    FROM Sales.SalesOrderHeader soh
    WHERE c.CustomerID = soh.CustomerID
);
```

--10-List of customers who ordered a Tofu

```
SELECT DISTINCT c.*
FROM Sales.Customer c
JOIN Sales.SalesOrderHeader soh ON c.CustomerID = soh.CustomerID
JOIN Sales.SalesOrderDetail sod ON soh.SalesOrderID = sod.SalesOrderID
JOIN Production.Product p ON sod.ProductID = p.ProductID
WHERE p.Name = 'Tofu';
```

--11-Details of first order of the system

```
SELECT TOP 1 *
FROM Sales.SalesOrderHeader
ORDER BY OrderDate ASC;
```

--12-Find the details of most expensive order date

```
WITH OrderTotals AS (
    SELECT
        SalesOrderID,
        SUM(LineTotal) AS TotalOrderCost
    FROM
        Sales.SalesOrderDetail
```

```
GROUP BY
    SalesOrderID
)
SELECT
    soh.SalesOrderID,
    soh.OrderDate,
    soh.DueDate,
    soh.ShipDate,
    soh.Status,
    soh.OnlineOrderFlag,
    soh.SalesOrderNumber,
    soh.PurchaseOrderNumber,
    soh.AccountNumber,
    soh.CustomerID,
    soh.SalesPersonID,
    soh.TerritoryID,
    soh.BillToAddressID,
    soh.ShipToAddressID,
    soh.ShipMethodID,
    soh.CreditCardID,
    soh.CreditCardApprovalCode,
    soh.CurrencyRateID,
    soh.SubTotal,
    soh.TaxAmt,
    soh.Freight,
    soh.TotalDue,
    soh.Comment,
    soh.rowguid,
    soh.ModifiedDate,
    ot.TotalOrderCost
FROM
    Sales.SalesOrderHeader soh
JOIN
    OrderTotals ot ON soh.SalesOrderID = ot.SalesOrderID
ORDER BY
    ot.TotalOrderCost DESC
```

--13- for each order get theorderid and average quantity of items in that order

```
SELECT
    SalesOrderID,
    AVG(OrderQty) AS AverageQuantity
FROM
    Sales.SalesOrderDetail
GROUP BY
    SalesOrderID;
```

--14-For each order get the OrderID,minimum quantity and maximum quantity for that order



```
SELECT
    SalesOrderID,
    MIN(OrderQty) AS MinimumQuantity,
    MAX(OrderQty) AS MaximumQuantity
FROM
    Sales.SalesOrderDetail
GROUP BY
    SalesOrderID;
```

--15-Get a list of all managers and total no of employees who report to them.

USE AdventureWork;

```
SELECT
    E.BusinessEntityID AS ManagerID,
    P.FirstName + ' ' + P.LastName AS ManagerName,
    COUNT(RE.BusinessEntityID) AS TotalEmployees
FROM
    HumanResources.Employee AS E
INNER JOIN
    Person.Person AS P ON E.BusinessEntityID = P.BusinessEntityID
LEFT JOIN
    HumanResources.Employee AS RE ON RE.OrganizationNode.GetAncestor(1) =
        E.OrganizationNode
GROUP BY
    E.BusinessEntityID, P.FirstName, P.LastName
ORDER BY
    TotalEmployees DESC;
```

--16-Get the orderID and total quantity for each order that has a total quantity of greater than 300.

```
SELECT
    SalesOrderID,
    SUM(OrderQty) AS TotalQuantity
FROM
    Sales.SalesOrderDetail
GROUP BY
    SalesOrderID
HAVING
    SUM(OrderQty) > 300;
```

--17-List of all the orders placed on after 1996/12/31.

```
SELECT *
FROM Sales.SalesOrderHeader
WHERE OrderDate >= '1996-12-31';
```

--18-List of all orders shipped to Canada.

```
SELECT
    soh.SalesOrderID,
```

```
    soh.OrderDate,
    soh.ShipDate,
    soh.Status,
    soh.TotalDue,
    a.AddressLine1,
    a.AddressLine2,
    a.City,
    sp.Name AS StateProvince,
    cr.Name AS CountryRegion
FROM
    Sales.SalesOrderHeader soh
JOIN
    Person.Address a ON soh.ShipToAddressID = a.AddressID
JOIN
    Person.StateProvince sp ON a.StateProvinceID = sp.StateProvinceID
JOIN
    Person.CountryRegion cr ON sp.CountryRegionCode = cr.CountryRegionCode
WHERE
    cr.Name = 'Canada';
```

--19-List of all orders with order total > 200 .

```
SELECT *
FROM Sales.SalesOrderHeader
WHERE TotalDue > 200;
```

--20-List of all countries and sales made in each country .

```
SELECT
    cr.Name AS Country,
    SUM(soh.TotalDue) AS TotalSales
FROM
    Sales.SalesOrderHeader soh
JOIN
    Person.Address a ON soh.ShipToAddressID = a.AddressID
JOIN
    Person.StateProvince sp ON a.StateProvinceID = sp.StateProvinceID
JOIN
    Person.CountryRegion cr ON sp.CountryRegionCode = cr.CountryRegionCode
GROUP BY
    cr.Name
ORDER BY
    TotalSales DESC;
```

--21-List of customers Contact name and numbers of orders they placed .

```
SELECT
    p.FirstName + ' ' + p.LastName AS ContactName,
    COUNT(soh.SalesOrderID) AS NumberOfOrders
FROM
    Sales.SalesOrderHeader soh
```

```
JOIN
    Sales.Customer c ON soh.CustomerID = c.CustomerID
JOIN
    Person.Person p ON c.PersonID = p.BusinessEntityID
GROUP BY
    p.FirstName, p.LastName
ORDER BY
    NumberOfOrders DESC;
```

--22-List of customers contactnames who have placed more than 3 orders.

USE AdventureWork;

```
SELECT
    P.FirstName + ' ' + P.LastName AS ContactName,
    COUNT(SOH.SalesOrderID) AS NumberOfOrders
FROM
    Sales.Customer AS C
INNER JOIN
    Sales.SalesOrderHeader AS SOH ON C.CustomerID = SOH.CustomerID
INNER JOIN
    Person.Person AS P ON C.PersonID = P.BusinessEntityID
GROUP BY
    P.FirstName, P.LastName
HAVING
    COUNT(SOH.SalesOrderID) > 3
ORDER BY
    NumberOfOrders DESC;
```

--23-List of discontinued products which were ordered between 1/1/1997 and 1/1/1998

USE AdventureWork;

```
SELECT
    P.ProductID,
    P.Name AS ProductName,
    P.DiscontinuedDate
FROM
    Sales.SalesOrderDetail AS SOD
INNER JOIN
    Sales.SalesOrderHeader AS SOH ON SOD.SalesOrderID = SOH.SalesOrderID
INNER JOIN
    Production.Product AS P ON SOD.ProductID = P.ProductID
WHERE
    SOH.OrderDate BETWEEN '1997-01-01' AND '1998-01-01'
    AND P.DiscontinuedDate IS NOT NULL
    AND P.DiscontinuedDate <= '1998-01-01'
GROUP BY
    P.ProductID, P.Name, P.DiscontinuedDate
ORDER BY
```

--24-List of employees firstname,lastname,superviser Firstname,Lastname
USE AdventureWork;

```
SELECT
    E1.FirstName AS EmployeeFirstName,
    E1.LastName AS EmployeeLastName,
    E2.FirstName AS SupervisorFirstName,
    E2.LastName AS SupervisorLastName
FROM
    HumanResources.Employee AS Emp1
INNER JOIN
    Person.Person AS E1 ON Emp1.BusinessEntityID = E1.BusinessEntityID
LEFT JOIN
    HumanResources.Employee AS Emp2 ON Emp1.OrganizationNode.GetAncestor(1) =
        Emp2.OrganizationNode
LEFT JOIN
    Person.Person AS E2 ON Emp2.BusinessEntityID = E2.BusinessEntityID
ORDER BY
    EmployeeLastName, EmployeeFirstName;
```

--25-List of employees ID and total sale conducted by employee.
USE AdventureWork;

```
SELECT
    E.BusinessEntityID AS EmployeeID,
    SUM(SOH.TotalDue) AS TotalSales
FROM
    Sales.SalesOrderHeader AS SOH
INNER JOIN
    Sales.SalesPerson AS SP ON SOH.SalesPersonID = SP.BusinessEntityID
INNER JOIN
    HumanResources.Employee AS E ON SP.BusinessEntityID = E.BusinessEntityID
GROUP BY
    E.BusinessEntityID
ORDER BY
    TotalSales DESC;
```

--26-List of employees whose Firstname contains character a.

```
USE AdventureWork;
SELECT FirstName, LastName
FROM HumanResources.Employee AS e
JOIN Person.Person AS p ON e.BusinessEntityID = p.BusinessEntityID
WHERE p.FirstName LIKE '%a%';
```

--27-List of managers who have more than four people reporting to them.

USE AdventureWork;

SELECT

ManagerPerson.FirstName,
ManagerPerson.LastName,
COUNT(Employee.BusinessEntityID) AS ReportCount

FROM

HumanResources.Employee AS Employee

JOIN

HumanResources.Employee AS Manager ON Employee.OrganizationNode.GetAncestor(1) =
Manager.OrganizationNode

JOIN

Person.Person AS ManagerPerson ON Manager.BusinessEntityID =
ManagerPerson.BusinessEntityID

GROUP BY

ManagerPerson.FirstName,
ManagerPerson.LastName

HAVING

COUNT(Employee.BusinessEntityID) > 4;

--28-List of orders and ProductNames.

USE AdventureWork;

SELECT

soh.SalesOrderID,
soh.OrderDate,
p.Name AS ProductName

FROM

Sales.SalesOrderHeader soh

JOIN

Sales.SalesOrderDetail sod ON soh.SalesOrderID = sod.SalesOrderID

JOIN

Production.Product p ON sod.ProductID = p.ProductID;

--29-List of orders placed by best customer.

--identify the best customer.

USE AdventureWork;

WITH CustomerTotal AS (

SELECT

soh.CustomerID,
SUM(soh.TotalDue) AS TotalSpent

FROM

Sales.SalesOrderHeader soh

GROUP BY

soh.CustomerID


```
)
SELECT TOP 1
    CustomerID,
    TotalSpent
FROM
    CustomerTotal
ORDER BY
    TotalSpent DESC;

--order placed by the best customer.
USE AdventureWork;
SELECT
    soh.SalesOrderID,
    soh.OrderDate,
    soh.TotalDue
FROM
    Sales.SalesOrderHeader soh
WHERE
    soh.CustomerID = (
        SELECT TOP 1
            CustomerID
        FROM
            (
                SELECT
                    soh.CustomerID,
                    SUM(soh.TotalDue) AS TotalSpent
                FROM
                    Sales.SalesOrderHeader soh
                GROUP BY
                    soh.CustomerID
            ) AS CustomerTotal
        ORDER BY
            TotalSpent DESC
    );

--30-List of orders placed by customer who do not have a fax number.
USE AdventureWork;

SELECT
    soh.SalesOrderID,
    soh.OrderDate,
    soh.TotalDue,
    c.CustomerID,
    p.FirstName,
    p.LastName
FROM
    Sales.SalesOrderHeader soh
JOIN
    Sales.Customer c ON soh.CustomerID = c.CustomerID
JOIN
```

```
Person.Person p ON c.PersonID = p.BusinessEntityID
LEFT JOIN
    Person.PersonPhone pp ON p.BusinessEntityID = pp.BusinessEntityID AND
        pp.PhoneNumberTypeID = (SELECT PhoneNumberTypeID FROM Person.PhoneNumberType
                                WHERE Name = 'Fax')
WHERE
    pp.PhoneNumber IS NULL;
```

--31-List of postal codes where the product tofu was shipped.

USE AdventureWork;

```
SELECT DISTINCT
    a.PostalCode
FROM
    Sales.SalesOrderDetail sod
JOIN
    Sales.SalesOrderHeader soh ON sod.SalesOrderID = soh.SalesOrderID
JOIN
    Production.Product p ON sod.ProductID = p.ProductID
JOIN
    Person.Address a ON soh.ShipToAddressID = a.AddressID
WHERE
    p.Name = 'Tofu';
```

--32-List of product Names that were shipped to France.

USE AdventureWork;

```
SELECT DISTINCT
    p.Name AS ProductName
FROM
    Sales.SalesOrderDetail sod
JOIN
    Sales.SalesOrderHeader soh ON sod.SalesOrderID = soh.SalesOrderID
JOIN
    Production.Product p ON sod.ProductID = p.ProductID
JOIN
    Person.Address a ON soh.ShipToAddressID = a.AddressID
JOIN
    Person.StateProvince sp ON a.StateProvinceID = sp.StateProvinceID
JOIN
    Person.CountryRegion cr ON sp.CountryRegionCode = cr.CountryRegionCode
WHERE
    cr.Name = 'France';
```

--33-List of ProductNames and Categories for the supplier 'Speciality Biscuits, Ltd.

USE AdventureWork;

```
SELECT
    p.Name AS ProductName,
    pc.Name AS CategoryName
FROM
    Production.Product p
JOIN
    Production.ProductSubcategory psc ON p.ProductSubcategoryID =
        psc.ProductSubcategoryID
JOIN
    Production.ProductCategory pc ON psc.ProductCategoryID = pc.ProductCategoryID
JOIN
    Purchasing.ProductVendor pv ON p.ProductID = pv.ProductID
JOIN
    Purchasing.Vendor v ON pv.BusinessEntityID = v.BusinessEntityID
WHERE
    v.Name = 'Speciality Biscuits, Ltd.';
```

--34-List of products that were never ordered.

USE AdventureWork;

```
SELECT
    p.ProductID,
    p.Name AS ProductName
FROM
    Production.Product p
LEFT JOIN
    Sales.SalesOrderDetail sod ON p.ProductID = sod.ProductID
WHERE
    sod.ProductID IS NULL;
```

--35--List of products where units in stock is less than 10 and units on order are 0.

USE AdventureWork;

```
SELECT
    p.ProductID,
    p.Name AS ProductName,
    pi.Quantity AS UnitsInStock
FROM
    Production.Product p
JOIN
    Production.ProductInventory pi ON p.ProductID = pi.ProductID
LEFT JOIN
    (SELECT ProductID, SUM(OrderQty) AS TotalOrdered
     FROM Purchasing.PurchaseOrderDetail
     GROUP BY ProductID) pod ON p.ProductID = pod.ProductID
WHERE
    pi.Quantity < 10
```

```
AND (pod.TotalOrdered IS NULL OR pod.TotalOrdered = 0);
```

```
--36-List of top 10 countries by sales.
```

```
USE AdventureWork;
```

```
SELECT TOP 10
```

```
    cr.Name AS Country,
```

```
    SUM(soh.TotalDue) AS TotalSales
```

```
FROM
```

```
    Sales.SalesOrderHeader soh
```

```
JOIN
```

```
    Person.Address a ON soh.ShipToAddressID = a.AddressID
```

```
JOIN
```

```
    Person.StateProvince sp ON a.StateProvinceID = sp.StateProvinceID
```

```
JOIN
```

```
    Person.CountryRegion cr ON sp.CountryRegionCode = cr.CountryRegionCode
```

```
GROUP BY
```

```
    cr.Name
```

```
ORDER BY
```

```
    TotalSales DESC;
```

```
--37-Number of orders each employee has taken for customers with CustomerIDs between A and AO.
```

```
USE AdventureWork;
```

```
SELECT
```

```
    e.BusinessEntityID AS EmployeeID,
```

```
    COUNT(soh.SalesOrderID) AS OrderCount
```

```
FROM
```

```
    Sales.SalesOrderHeader soh
```

```
JOIN
```

```
    Sales.Customer c ON soh.CustomerID = c.CustomerID
```

```
JOIN
```

```
    HumanResources.Employee e ON soh.SalesPersonID = e.BusinessEntityID
```

```
WHERE
```

```
    c.AccountNumber BETWEEN 'A' AND 'AO'
```

```
GROUP BY
```

```
    e.BusinessEntityID;
```

```
--38-Orderdate of most expensive order.
```

```
USE AdventureWork;
```

```
SELECT
```

```
    OrderDate
```

```
FROM
```

```
    Sales.SalesOrderHeader
```

WHERE

TotalDue = (SELECT MAX(TotalDue) FROM Sales.SalesOrderHeader);

--39-Product name and total revenue from that product.

USE AdventureWork;

SELECT

p.Name AS ProductName,
SUM(sod.LineTotal) AS TotalRevenue

FROM

Sales.SalesOrderDetail sod

JOIN

Sales.SalesOrderHeader soh ON sod.SalesOrderID = soh.SalesOrderID

JOIN

Production.Product p ON sod.ProductID = p.ProductID

GROUP BY

p.Name

ORDER BY

TotalRevenue DESC;

--40-Supplierid and number of products offered.

USE AdventureWork;

SELECT

pv.BusinessEntityID AS SupplierID,
COUNT(pv.ProductID) AS NumberOfProducts

FROM

Purchasing.ProductVendor pv

GROUP BY

pv.BusinessEntityID

ORDER BY

NumberOfProducts DESC;

--41-Top ten customers based on their business.

USE AdventureWork;

SELECT

c.CustomerID,
p.FirstName,
p.LastName,
SUM(soh.TotalDue) AS TotalBusiness

FROM

Sales.SalesOrderHeader soh

JOIN

Sales.Customer c ON soh.CustomerID = c.CustomerID

JOIN

```
    Person.Person p ON c.PersonID = p.BusinessEntityID
GROUP BY
    c.CustomerID,
    p.FirstName,
    p.LastName
ORDER BY
    TotalBusiness DESC
OFFSET 0 ROWS
FETCH NEXT 10 ROWS ONLY;
```

--42-what is the total revenue of that company.

USE AdventureWork;

```
SELECT
    SUM(TotalDue) AS TotalRevenue
FROM
    Sales.SalesOrderHeader;
```