AdventureWorks Microsoft SQL Server Queries

```sql
--Database Overview
SELECT TABLE_SCHEMA as SchemaName, COUNT(TABLE_NAME) as ViewsPerSchema
FROM AdventureWorks2019.INFORMATION_SCHEMA.TABLES
GROUP BY TABLE_SCHEMA, TABLE_TYPE
HAVING TABLE_TYPE = 'VIEW';
SELECT COUNT(DISTINCT TABLE_NAME) as NumberOfBaseTables
FROM AdventureWorks2019.INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'BASE TABLE';
SELECT COUNT(DISTINCT TABLE_NAME) as NumberOfViews
FROM AdventureWorks2019.INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'VIEW';
SELECT COUNT(DISTINCT COLUMN_NAME) as NumberofUniqueColumns
FROM AdventureWorks2019.INFORMATION_SCHEMA.COLUMNS;

--Date range in dataset and total number of orders
SELECT MIN(CAST(OrderDate AS DATE)) AS DataStart, MAX(CAST(OrderDate AS Date)) AS DataEnd,
COUNT(SalesOrderID) AS NumberOfOrders
FROM Sales.SalesOrderHeader

--Customers Overview
--Customer Types
SELECT
        COUNT(DISTINCT CustomerID) AS NumberOfActiveCustomers,
        CASE
                WHEN OnlineOrderFlag = 0 THEN 'Reseller'
                WHEN OnlineOrderFlag = 1 THEN 'Retail'
        END AS CustomerType
FROM Sales.SalesOrderHeader
GROUP BY OnlineOrderFlag;
SELECT
        COUNT(DISTINCT SalesOrderID) AS NumberOfOrdersByCustomerType,
        CASE
                WHEN OnlineOrderFlag = 0 THEN 'Reseller'
                WHEN OnlineOrderFlag = 1 THEN 'Retail'
        END AS CustomerType
FROM Sales.SalesOrderHeader
GROUP BY OnlineOrderFlag;
SELECT COUNT(DISTINCT SalesOrderID) AS TotalNumberOfOrders
FROM Sales.SalesOrderHeader;
SELECT
        SUM(SubTotal) AS TotalSalesByCustomerType,
        CASE
                WHEN OnlineOrderFlag = 0 THEN 'Reseller'
                WHEN OnlineOrderFlag = 1 THEN 'Retail'
        END AS CustomerType
FROM Sales.SalesOrderHeader
GROUP BY OnlineOrderFlag;
SELECT SUM(SubTotal) AS TotalSales
FROM Sales.SalesOrderHeader;
--Customer Locations
--Resellers
SELECT
        COUNT(DISTINCT City) AS City,
        COUNT(DISTINCT StateProvinceName) AS StateProvince,
        COUNT(DISTINCT CountryRegionName) AS Country
FROM Sales.vStoreWithAddresses
--Retail
SELECT
        COUNT(DISTINCT City) AS City,
        COUNT(DISTINCT StateProvinceName) AS StateProvince,
        COUNT(DISTINCT CountryRegionName) AS Country
FROM Sales.vIndividualCustomer
--Countries Outside of the US
--Resellers
SELECT
```

```sql
        COUNT(DISTINCT City) AS City,
        COUNT(DISTINCT StateProvinceName) AS StateProvince
FROM Sales.vStoreWithAddresses
WHERE CountryRegionName != 'United States'
--Retail
SELECT
        COUNT(DISTINCT City) AS City,
        COUNT(DISTINCT StateProvinceName) AS StateProvince
FROM Sales.vIndividualCustomer
WHERE CountryRegionName != 'United States'

--Demographics
--Resellers
SELECT
        MIN(YearOpened) AS EarliestYearOpened,
        AVG(YearOpened) AS AverageYearOpened,
        MAX(YearOpened) AS LatestYearOpened
FROM Sales.vStoreWithDemographics
SELECT
        BusinessType,
        COUNT(DISTINCT BusinessEntityID) AS Count
FROM Sales.vStoreWithDemographics
GROUP BY BusinessType
SELECT
        Specialty,
        COUNT(DISTINCT BusinessEntityID) AS Count
FROM Sales.vStoreWithDemographics
GROUP BY Specialty;
--Retail
SELECT
        MIN(DATEDIFF(YEAR, BirthDate, '2014-06-30 00:00:00.000')) AS Youngest,
        AVG(DATEDIFF(YEAR, BirthDate, '2014-06-30 00:00:00.000')) AS AverageAge,
        MAX(DATEDIFF(YEAR, BirthDate, '2014-06-30 00:00:00.000')) AS Oldest
FROM Sales.vPersonDemographics;
-- above is a fictional scenario, below is real world scenario
SELECT
        MIN(DATEDIFF(YEAR, BirthDate, GETDATE())) AS Youngest,
        AVG(DATEDIFF(YEAR, BirthDate, GETDATE())) AS AverageAge,
        MAX(DATEDIFF(YEAR, BirthDate, GETDATE())) AS Oldest
FROM Sales.vPersonDemographics;
SELECT
        Gender,
        COUNT(DISTINCT BusinessEntityID) AS Count
FROM Sales.vPersonDemographics
GROUP BY Gender;
SELECT
        YearlyIncome,
        COUNT(DISTINCT BusinessEntityID) AS Count
FROM Sales.vPersonDemographics
GROUP BY YearlyIncome
ORDER BY YearlyIncome;

USE [AdventureWorks2019]
GO

--Use Table-valued Function to create dataset(s) for future reference to ease analysis
--All Customer
SELECT *
FROM dbo.ufnGetSalesbyCustomerType(0)
UNION ALL
SELECT *
FROM dbo.ufnGetSalesbyCustomerType(1);
--Resellers
SELECT *
FROM dbo.ufnGetSalesbyCustomerType(0);
--Retail
SELECT *
FROM dbo.ufnGetSalesbyCustomerType(1);
```

```
/****** Object:  UserDefinedFunction [dbo].[ufnGetSalesbyCustomerType]    Script Date: 4/4/2023 2:19:51 PM
******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO


CREATE FUNCTION [dbo].[ufnGetSalesbyCustomerType](@CustomerType int)
RETURNS @retGetSalesbyCustomerType TABLE
(
    -- Columns returned by the function
        [OrderDate] datetime NOT NULL,
        [OrderYear] int NOT NULL,
        [OrderMonth] int NOT NULL,
        [OrderDayofMonth] int NOT NULL,
        [OrderQuarter] int NOT NULL,
        [CustomerName] [nvarchar](50) NOT NULL,
        [CustomerType] [nvarchar](50) NOT NULL,
        [OrderQuantity] int NOT NULL,
        [SalesAmount] money NOT NULL,
        [ProductName] [nvarchar](50),
        [ProductCategoryName] [nvarchar](50) NOT NULL,
        [ProductSubcategoryName] [nvarchar](50) NOT NULL,
        [TerritoryID] int NOT NULL,
        [City] [nvarchar](30) NOT NULL,
        [StateProvinceName] [nvarchar](50) NOT NULL,
        [PostalCode] [nvarchar](15) NOT NULL,
        [CountryRegionName] [nvarchar](50) NOT NULL,
        [Age] int, -- if Reseller = years open at year of order / if Retail = age at year of order
        [CustomerSubcategory] [nvarchar](50) NULL, --if Reseller = BusinessType and Specialty / if Retail
= YearlyIncome
    [BusinessEntityID] int NOT NULL
)
AS
-- Returns sales, product, location details for the specified customer type (either Reseller = 0 or Retail
= 1).
-- Note: Many rows will be returned for based on all sales data available in Sale.SalesOrderHeader and
Sales.SalesOrderDetail
BEGIN
        IF @CustomerType IS NOT NULL
                BEGIN
                IF @CustomerType = 0 --Reseller
                        INSERT INTO @retGetSalesbyCustomerType
                                SELECT
                                        CAST(OrderDate AS DATE) AS OrderDate,
                                        DATEPART(YEAR, OrderDate) AS OrderYear,
                                        DATEPART(MONTH, OrderDate) AS OrderMonth,
                                        DATEPART(DAY, OrderDate) AS OrderDayofMonth,
                                        DATEPART(QUARTER, OrderDate) AS OrderQuarter,
                                        ss.Name AS CustomerName,
                                        'Reseller' AS CustomerType,
                                        sod.OrderQty AS OrderQuantity,
                                        sod.LineTotal AS SalesAmount,
                                        pp.Name AS ProductName,
                                        pc.Name AS ProductCategoryName,
                                        ps.Name AS ProductSubcategoryName,
                                        soh.TerritoryID AS TerritoryID,
                                        sa.City AS City,
                                        sa.StateProvinceName AS StateProvinceName,
                                        sa.PostalCode AS PostalCode,
                                        sa.CountryRegionName AS CountryRegionName,
                                        CAST(DATEPART(YEAR, OrderDate) AS int) - YearOpened AS Age,
                                        BusinessType AS CustomerSubcategory,
                                        ss.BusinessEntityID as BusinessEntityID
                                FROM Sales.SalesOrderHeader soh
                                JOIN Sales.SalesOrderDetail sod
                                ON soh.SalesOrderID = sod.SalesOrderID
                                JOIN Sales.Customer sc
                                ON sc.CustomerID = soh.CustomerID
                                JOIN Sales.Store ss
                                ON sc.StoreID = ss.BusinessEntityID
                                JOIN Sales.vStoreWithDemographics sd
                                ON ss.BusinessEntityID = sd.BusinessEntityID
                                JOIN Sales.vStoreWithAddresses sa
                                ON sd.BusinessEntityID = sa.BusinessEntityID
                                JOIN Production.Product pp
                                ON sod.ProductID = pp.ProductID
                                JOIN Production.ProductSubcategory ps
                                ON pp.ProductSubcategoryID = ps.ProductSubcategoryID
                                JOIN Production.ProductCategory pc
                                ON pc.ProductCategoryID = ps.ProductCategoryID;
                        ELSE --Retail
                            INSERT INTO @retGetSalesbyCustomerType
                                SELECT
                                        CAST(OrderDate AS DATE) AS OrderDate,
                                        DATEPART(YEAR, OrderDate) AS OrderYear,
                                        DATEPART(MONTH, OrderDate) AS OrderMonth,
                                        DATEPART(DAY, OrderDate) AS OrderDayofMonth,
                                        DATEPART(QUARTER, OrderDate) AS OrderQuarter,
                                        CONCAT(ic.LastName, ',', ic.FirstName) AS CustomerName,
                                        'Retail' AS CustomerType,
                                        sod.OrderQty AS OrderQuantity,
                                        sod.LineTotal AS SalesAmount,
                                        pp.Name AS ProductName,
                                        pc.Name AS ProductCategoryName,
                                        ps.Name AS ProductSubcategoryName,
                                        soh.TerritoryID AS TerritoryID,
                                        ic.City AS City,
                                        ic.StateProvinceName AS StateProvinceName,
                                        ic.PostalCode AS PostalCode,
                                        ic.CountryRegionName AS CountryRegionName,
                                        DATEDIFF(YEAR, BirthDate, OrderDate) AS Age,
                                                YearlyIncome AS CustomerSubcategory,
                                        ic.BusinessEntityID as BusinessEntityID
                                FROM Sales.SalesOrderHeader soh
                                JOIN Sales.SalesOrderDetail sod
                                ON soh.SalesOrderID = sod.SalesOrderID
                                JOIN Sales.Customer sc
                                ON sc.CustomerID = soh.CustomerID
                                JOIN Person.Person ppe
                                ON sc.PersonID = ppe.BusinessEntityID
                                JOIN Sales.vIndividualCustomer ic
                                ON ppe.BusinessEntityID = ic.BusinessEntityID
                                JOIN Sales.vPersonDemographics pd
                                ON ic.BusinessEntityID = pd.BusinessEntityID
                                JOIN Production.Product pp
                                ON sod.ProductID = pp.ProductID
                                JOIN Production.ProductSubcategory ps
                                ON pp.ProductSubcategoryID = ps.ProductSubcategoryID
                                JOIN Production.ProductCategory pc
                                ON pc.ProductCategoryID = ps.ProductCategoryID;
                        END

                    RETURN;
                END;

                GO

                EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Input parameter for the table value
                function ufnGetSalesbyCustomerType. Enter a valid CustomerType (0 = Reseller, 1 = Retail).' ,
                @level0type=N'SCHEMA',@level0name=N'dbo',
                @level1type=N'FUNCTION',@level1name=N'ufnGetSalesbyCustomerType',
                @level2type=N'PARAMETER',@level2name=N'@CustomerType'
                GO
```

```
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Table value function returning the
sales, product, customer details for either Reseller or Retail Customers.' ,
@level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'FUNCTION',@level1name=N'ufnGetSalesbyCustomerType'
GO



--Product Preferences
--Quantity
--Resellers
SELECT
        COUNT(OrderQuantity) AS QuantityByProductCategory,
        ProductCategoryName
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY ProductCategoryName
ORDER BY QuantityByProductCategory DESC;
SELECT
        TOP 5 COUNT(OrderQuantity) AS QuantityByProductSubcategory,
        ProductSubcategoryName
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY ProductSubcategoryName
ORDER BY QuantityByProductSubcategory DESC;
SELECT
        TOP 5 COUNT(OrderQuantity) AS QuantityByProduct,
        ProductName
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY ProductName
ORDER BY QuantityByProduct DESC;
--Retail
SELECT
        COUNT(OrderQuantity) AS QuantityByProductCategory,
        ProductCategoryName
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY ProductCategoryName
ORDER BY QuantityByProductCategory DESC;
SELECT
        TOP 5 COUNT(OrderQuantity) AS QuantityByProductSubcategory,
        ProductSubcategoryName
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY ProductSubcategoryName
ORDER BY QuantityByProductSubcategory DESC;
SELECT
        TOP 5 COUNT(OrderQuantity) AS QuantityByProduct,
        ProductName
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY ProductName
ORDER BY QuantityByProduct DESC;
--Sales Amount
--Resellers
SELECT
        SUM(SalesAmount) AS SalesByProductCategory,
        ProductCategoryName
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY ProductCategoryName
ORDER BY SalesByProductCategory DESC;
SELECT
        TOP 5 SUM(SalesAmount) AS SalesByProductSubcategory,
        ProductSubcategoryName
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY ProductSubcategoryName
ORDER BY SalesByProductSubcategory DESC;
SELECT
        TOP 5 SUM(SalesAmount) AS SalesByProduct,
        ProductName
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY ProductName
ORDER BY SalesByProduct DESC;
--Retail--
SELECT
```

```sql
        SUM(SalesAmount) AS SalesByProductCategory,
        ProductCategoryName
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY ProductCategoryName
ORDER BY SalesByProductCategory DESC;
SELECT
        TOP 5 SUM(SalesAmount) AS SalesByProductSubcategory,
        ProductSubcategoryName
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY ProductSubcategoryName
ORDER BY SalesByProductSubcategory DESC;
SELECT
        TOP 5 SUM(SalesAmount) AS SalesByProduct,
        ProductName
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY ProductName
ORDER BY SalesByProduct DESC;
--Non-Mountain-200 Products
--Reseller
SELECT
        TOP 5 SUM(SalesAmount) AS SalesByProduct,
        ProductName
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY ProductName
HAVING ProductName NOT LIKE 'Mountain-200%'
ORDER BY SalesByProduct DESC;
--Retail
SELECT
        TOP 5 SUM(SalesAmount) AS SalesByProduct,
        ProductName
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY ProductName
HAVING ProductName NOT LIKE 'Mountain-200%'
ORDER BY SalesByProduct DESC;

-- Time and Sales
--Quarter
---Resellers
SELECT DATEPART(QUARTER, OrderDate) AS 'Quarter', SUM(SalesAmount) AS TotalQuarterlySales
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY DATEPART(QUARTER, OrderDate)
ORDER BY TotalQuarterlySales DESC
---Retail
SELECT DATEPART(QUARTER, OrderDate) AS 'Quarter', SUM(SalesAmount) AS TotalQuarterlySales
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY DATEPART(QUARTER, OrderDate)
ORDER BY TotalQuarterlySales DESC
--Month
---Resellers
SELECT DATEPART(MONTH, OrderDate) AS 'Month', SUM(SalesAmount) AS TotalMonthlySales
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY DATEPART(MONTH, OrderDate)
ORDER BY TotalMonthlySales DESC
---Retail
SELECT DATEPART(MONTH, OrderDate) AS 'Month', SUM(SalesAmount) AS TotalMonthlySales
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY DATEPART(MONTH, OrderDate)
ORDER BY TotalMonthlySales DESC
--Day of Month
---Resellers
SELECT DATEPART(DAY, OrderDate) AS 'Day', SUM(SalesAmount) AS TotalDayofMonthSales
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY DATEPART(DAY, OrderDate)
ORDER BY TotalDayofMonthSales DESC
---Retail
SELECT DATEPART(DAY, OrderDate) AS 'Day', SUM(SalesAmount) AS TotalDayofMonthSales
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY DATEPART(DAY, OrderDate)
ORDER BY TotalDayofMonthSales DESC
```

```sql
--Day of Week
---Resellers
SET DATEFIRST 1; -- 1=MON 7=SUN
SELECT DATEPART(DW, OrderDate) AS 'DayOfWeek', SUM(SalesAmount) AS TotalDayofWeekSales
FROM dbo.ufnGetSalesbyCustomerType(0)
GROUP BY DATEPART(DW, OrderDate)
ORDER BY TotalDayofWeekSales DESC
---Retail
SET DATEFIRST 1; -- 1=MON 7=SUN
SELECT DATEPART(DW, OrderDate) AS 'DayOfWeek', SUM(SalesAmount) AS TotalDayofWeekSales
FROM dbo.ufnGetSalesbyCustomerType(1)
GROUP BY DATEPART(DW, OrderDate)
ORDER BY TotalDayofWeekSales DESC

--Opportunities
--Reaching Out to Inactive Customers
--Inactive Customers (no order in 12 months, but were active in 12 months prior to that)
--Reseller
SELECT pe.BusinessEntityID, EmailAddress, TerritoryID, StoreID, PersonType, FirstName, LastName,
PhoneNumber
FROM Person.EmailAddress pe
JOIN Person.Person pp
ON pe.BusinessEntityID = pp.BusinessEntityID
JOIN Person.PersonPhone ppp
ON pp.BusinessEntityID = ppp.BusinessEntityID
JOIN Sales.Customer sc
ON pp.BusinessEntityID = sc.PersonID
WHERE CustomerID NOT IN (
        SELECT soh.CustomerID
        FROM Sales.SalesOrderHeader soh
        WHERE OrderDate >= DATEADD(M,-12,GETDATE()) AND OrderDate >= DATEADD(M,-24,GETDATE()))
        AND PersonType = 'SC';
SELECT pe.BusinessEntityID, EmailAddress, TerritoryID, StoreID, PersonType, FirstName, LastName,
PhoneNumber
FROM Person.EmailAddress pe
JOIN Person.Person pp
ON pe.BusinessEntityID = pp.BusinessEntityID
JOIN Person.PersonPhone ppp
ON pp.BusinessEntityID = ppp.BusinessEntityID
JOIN Sales.Customer sc
ON pp.BusinessEntityID = sc.PersonID
WHERE CustomerID NOT IN (
        SELECT soh.CustomerID
        FROM Sales.SalesOrderHeader soh
        WHERE OrderDate >= DATEADD(M,-12,GETDATE()) AND OrderDate >= DATEADD(M,-24,GETDATE()))
        AND PersonType = 'IN';
--test as historical data, above query for fictional company, GETDATE = datetime of today
SELECT MAX(OrderDate)
FROM Sales.SalesOrderHeader
--use max date: 2014-06-30 00:00:00.000
--Resellers SC = Store Contact
SELECT pe.BusinessEntityID, EmailAddress, TerritoryID, StoreID, PersonType, FirstName, LastName,
PhoneNumber
FROM Person.EmailAddress pe
JOIN Person.Person pp
ON pe.BusinessEntityID = pp.BusinessEntityID
JOIN Person.PersonPhone ppp
ON pp.BusinessEntityID = ppp.BusinessEntityID
JOIN Sales.Customer sc
ON pp.BusinessEntityID = sc.PersonID
WHERE CustomerID NOT IN (
        SELECT soh.CustomerID
        FROM Sales.SalesOrderHeader soh
        WHERE OrderDate >= DATEADD(M,-12,'2014-06-30 00:00:00.000') AND OrderDate >= DATEADD(M,-24,'2014-
06-30 00:00:00.000'))
        AND PersonType = 'SC';
-- 146 resellers
SELECT pe.BusinessEntityID, EmailAddress, TerritoryID, StoreID, PersonType, FirstName, LastName,
PhoneNumber
```

```
FROM Person.EmailAddress pe
JOIN Person.Person pp
ON pe.BusinessEntityID = pp.BusinessEntityID
JOIN Person.PersonPhone ppp
ON pp.BusinessEntityID = ppp.BusinessEntityID
JOIN Sales.Customer sc
ON pp.BusinessEntityID = sc.PersonID
WHERE CustomerID NOT IN (
        SELECT soh.CustomerID
        FROM Sales.SalesOrderHeader soh
        WHERE OrderDate >= DATEADD(M,-12,'2014-06-30 00:00:00.000') AND OrderDate >= DATEADD(M,-24,'2014-
06-30 00:00:00.000'))
        AND PersonType = 'IN';
--870 retail customers, IN = Individual Customer
--Potential Gains Inactive to Active
--resellers
SELECT AVG(SubTotal) * 146
FROM Sales.SalesOrderHeader
WHERE OnlineOrderFlag = 0
--3087547.2348
--retail
SELECT AVG(SubTotal) * 870
FROM Sales.SalesOrderHeader
WHERE OnlineOrderFlag = 1
--923462.457

--Define stored procedure to allow for action
USE [AdventureWorks2019]
GO

/****** Object:  StoredProcedure [dbo].[uspGetInactiveCustomers]    Script Date: 4/5/2023 3:05:39 PM
******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

--obtain contact information for inactive customers who have not placed an order in the past 12 months,
--but who have placed an order in past 24 months
--@PersonType = IN for individual retail customers
--@PersonType = SC for stores reseller customers
CREATE PROCEDURE [dbo].[uspGetInactiveCustomers]
    @PersonType nvarchar(2)
AS
BEGIN
    SET NOCOUNT ON;
        IF @PersonType = 'SC'
                SELECT pe.BusinessEntityID, EmailAddress, TerritoryID, StoreID, PersonType, FirstName,
LastName, PhoneNumber
                FROM Person.EmailAddress pe
                JOIN Person.Person pp
                ON pe.BusinessEntityID = pp.BusinessEntityID
                JOIN Person.PersonPhone ppp
                ON pp.BusinessEntityID = ppp.BusinessEntityID
                JOIN Sales.Customer sc
                ON pp.BusinessEntityID = sc.PersonID
                WHERE CustomerID NOT IN (
                        SELECT soh.CustomerID
                        FROM Sales.SalesOrderHeader soh
                        WHERE OrderDate >= DATEADD(M,-12,GETDATE()) AND OrderDate <= DATEADD(M,-
24,GETDATE()))
                        AND PersonType = 'SC';
        IF @PersonType = 'IN'
                SELECT pe.BusinessEntityID, EmailAddress, TerritoryID, StoreID, PersonType, FirstName,
LastName, PhoneNumber
                FROM Person.EmailAddress pe
                JOIN Person.Person pp
                ON pe.BusinessEntityID = pp.BusinessEntityID
                JOIN Person.PersonPhone ppp
                ON pp.BusinessEntityID = ppp.BusinessEntityID
                JOIN Sales.Customer sc
                ON pp.BusinessEntityID = sc.PersonID
                WHERE CustomerID NOT IN (
                        SELECT soh.CustomerID
                        FROM Sales.SalesOrderHeader soh
                        WHERE OrderDate >= DATEADD(M,-12,GETDATE()) AND OrderDate <= DATEADD(M,-
24,GETDATE()))
                        AND PersonType = 'IN';
END;
GO

EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Stored procedure to return contact
information for inactive customers.' , @level0type=N'SCHEMA',@level0name=N'dbo',
@level1type=N'PROCEDURE',@level1name=N'uspGetInactiveCustomers'
GO

EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Input parameter for the stored procedure
uspGetInactiveCustomers. Enter a valid PersonType IN or SC.' , @level0type=N'SCHEMA',@level0name=N'dbo',
@level1type=N'PROCEDURE',@level1name=N'uspGetInactiveCustomers',
@level2type=N'PARAMETER',@level2name=N'@PersonType'
GO
--to use stored procedure
--@PersonType = 'SC' for resellers, 'IN' for retail
EXEC dbo.uspGetInactiveCustomers @PersonType = 'IN'
GO

EXEC dbo.uspGetInactiveCustomers @PersonType = 'SC'
GO

--Opportunities
--Balance Supply and Demand
--supply demand: by product is total manufactured less than total sold
-- identify production opportunities to meet customer needs
SELECT
        pw.ProductID,
        Name AS ProductName,
        SUM(StockedQty) as TotalManufactured,
        SUM(sod.OrderQty) AS TotalSold,
        SUM(StockedQty) - SUM(sod.OrderQty) AS MoreSupplyvDemand
FROM Production.WorkOrder pw
JOIN Production.Product pp
ON pw.ProductID = pp.ProductID
JOIN Sales.SalesOrderDetail sod
ON pp.ProductID = sod.ProductID
GROUP BY pw.ProductID, Name
ORDER BY MoreSupplyvDemand
-- identify over production of a product, areas to prevent loss, areas to promote sales
SELECT
        pw.ProductID,
        Name AS ProductName,
        SUM(StockedQty) as TotalManufactured,
        SUM(sod.OrderQty) AS TotalSold,
        SUM(StockedQty) - SUM(sod.OrderQty) AS MoreSupplyvDemand
FROM Production.WorkOrder pw
JOIN Production.Product pp
ON pw.ProductID = pp.ProductID
JOIN Sales.SalesOrderDetail sod
ON pp.ProductID = sod.ProductID
GROUP BY pw.ProductID, Name
ORDER BY MoreSupplyvDemand DESC

--Opportunities
--Address Sales Decreases
--Change in Sales YOY
--Reseller
SELECT
        DATEPART(YEAR, OrderDate) as YearSales,
```

```sql
        OnlineOrderFlag AS CustomerType,
        SUM(SubTotal) - LAG(SUM(SubTotal)) OVER (ORDER BY DATEPART(YEAR, OrderDate)) AS ChangeInSales
FROM Sales.SalesOrderHeader
GROUP BY DATEPART(YEAR, OrderDate), OnlineOrderFlag
HAVING OnlineOrderFlag = 0
---adjust for fewer months in 2014 and 2011
--EXCEL ChangeInSales/6 if 2014 else /12
--Retail
SELECT
        DATEPART(YEAR, OrderDate) as YearSales,
        OnlineOrderFlag AS CustomerType,
        SUM(SubTotal) - LAG(SUM(SubTotal)) OVER (ORDER BY DATEPART(YEAR, OrderDate)) AS ChangeInSales
FROM Sales.SalesOrderHeader
GROUP BY DATEPART(YEAR, OrderDate), OnlineOrderFlag
HAVING OnlineOrderFlag = 1
---adjust for fewer months in 2014 and 2011
--EXCEL ChangeInSales/6 if 2014 else /12

--Causes
--explore possible relationships
--Promotions: utilization and time
--reseller utilization
SELECT DATEPART(YEAR,OrderDate) AS DiscountYears, SUM(UnitPriceDiscount)*SUM(OrderQty) AS
TotalDiscountUsed
FROM Sales.SalesOrderHeader soh
JOIN Sales.SalesOrderDetail sod
ON soh.SalesOrderID = sod.SalesOrderID
GROUP BY DATEPART(YEAR, OrderDate), OnlineOrderFlag
HAVING OnlineOrderFlag = 0
--retail utilization
SELECT DATEPART(YEAR,OrderDate) AS DiscountYears, SUM(UnitPriceDiscount)*SUM(OrderQty) AS
TotalDiscountUsed
FROM Sales.SalesOrderHeader soh
JOIN Sales.SalesOrderDetail sod
ON soh.SalesOrderID = sod.SalesOrderID
GROUP BY DATEPART(YEAR, OrderDate), OnlineOrderFlag
HAVING OnlineOrderFlag = 1
--adjust for 2014
--count of promotions starting
SELECT COUNT(Description) AS NumberOfPromotionsStarted, DATEPART(YEAR, StartDate) AS StartYear
FROM Sales.SpecialOffer
GROUP BY DATEPART(YEAR, StartDate)
--count of promotions ending
SELECT COUNT(Description) AS NumberOfPromotionsEnding, DATEPART(YEAR, EndDate) AS EndYear
FROM Sales.SpecialOffer
GROUP BY DATEPART(YEAR, EndDate)
--Price Increases
--product categories
SELECT
        DATEPART(YEAR, StartDate) as YearPrice,
        DATEPART(MONTH, StartDate) as MonthPrice,
        SUM(plph.ListPrice) as ListPrice,
        LAG(SUM(plph.ListPrice)) OVER (ORDER BY DATEPART(YEAR, StartDate), DATEPART(MONTH, StartDate)) AS
PreviousPrice,
        SUM(plph.ListPrice) - LAG(SUM(plph.ListPrice)) OVER (ORDER BY DATEPART(YEAR, StartDate),
DATEPART(MONTH, StartDate)) AS ChangeInPrice
FROM Production.ProductListPriceHistory plph
JOIN Production.Product pp
ON plph.ProductID = pp.ProductID
JOIN Production.ProductSubcategory ps
ON pp.ProductSubcategoryID = ps.ProductSubcategoryID
JOIN Production.ProductCategory pc
ON ps.ProductCategoryID = pc.ProductCategoryID
GROUP BY DATEPART(YEAR, StartDate), DATEPART(MONTH, StartDate), pc.Name
HAVING pc.Name = 'Accessories'
--'Bikes' 'Clothing' 'Components'
```

```sql
--touring bikes
--Opportunities
SELECT TOP 3 SUM(LineTotal) TotalSales, pp.Name AS TopThreeTouringProductsByTotalSales
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
GROUP BY pp.Name
HAVING pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%'
-- HL Touring Frame - Yellow, 60, LL Touring Frame - Yellow, 62, HL Touring Frame - Yellow, 46
SELECT
        MIN(CAST(sod.ModifiedDate AS DATE)) AS FirstTouringProductOrder,
        MAX(CAST(sod.ModifiedDate AS DATE)) AS MostRecentTouringProductOrder
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
WHERE pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%'
--2013-05-30    2014-06-30

--touring and resellers
SELECT
        MIN(CAST(OrderDate AS DATE)) AS ResellerFirstTouringProductOrder
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 0
--2013-05-30
SELECT
        MAX(LineTotal) AS ResellerMaxTotalSales
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 0
--27893.619000
SELECT
        CAST(OrderDate AS DATE) AS ResellerDateOfMaxTotalSales
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 0 AND
LineTotal = 27893.619000
--2013-08-30
SELECT COUNT(OrderQty) AS ResellerTotalOrderQuantityOfTouringProducts
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 0
--6362
SELECT SUM(LineTotal) AS ResellerTotalSalesOfTouringProducts
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 0
--12,131,505.27

--touring and retail
SELECT
        MIN(CAST(OrderDate AS DATE)) AS RetailirstTouringProductOrder
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
```

```sql
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 1
--2013-05-30
SELECT
        MAX(LineTotal) AS RetailMaxTotalSales
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 1
--2384.070000
SELECT
        CAST(OrderDate AS DATE) AS RetailDateOfMaxTotalSales
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 1 AND
LineTotal = 2384.070000
--many dates
SELECT
        pp.Name AS RetailMaxTotalSalesProductOrdered
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 1 AND
LineTotal = 2384.070000
--Touring-1000 bikes
SELECT
        pp.Name AS RetailMaxTotalSalesProductOrdered,
        SUM(LineTotal) AS TotalTouringSalesByProduct
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
GROUP BY LineTotal, pp.Name, OnlineOrderFlag
HAVING (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 1
ORDER BY LineTotal DESC
--Yellow
SELECT COUNT(OrderQty) AS RetailTotalOrderQuantityOfTouringProducts
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 1
--4590
SELECT SUM(LineTotal) AS RetailTotalSalesOfTouringProducts
FROM Sales.SalesOrderDetail sod
JOIN Production.Product pp
ON sod.ProductID = pp.ProductID
JOIN Sales.SalesOrderHeader soh
ON sod.SalesOrderID = soh.SalesOrderID
WHERE (pp.Name LIKE 'Tour%' OR pp.Name LIKE '%tour%' OR pp.Name LIKE '%Tour%') AND OnlineOrderFlag = 1
--3879331.82
```