1.

SELECT FullName, FaxNumber, PhoneNumber, CompanyFaxNumber, CompanyPhoneNumber

FROM Application.People p0

LEFT JOIN (

SELECT p.PersonID, s1.FaxNumber AS CompanyFaxNumber, s1.PhoneNumber AS CompanyPhoneNumber

FROM Application.People p

JOIN Sales.Customers s1 ON p.PersonID = s1.PrimaryContactPersonID

UNION

SELECT p.PersonID, s2.FaxNumber, s2.PhoneNumber

FROM Application.People p

JOIN Sales.Customers s2 ON p.PersonID = s2.AlternateContactPersonID) a

ON p0.PersonID = a.PersonID

2.

SELECT CustomerName AS CompanyName

FROM Sales.Customers c

JOIN Application.People p ON c.PrimaryContactPersonID = p.PersonID

AND c.PhoneNumber = p.PhoneNumber

3.

SELECT DISTINCT CustomerID

FROM Sales.Orders

WHERE YEAR(OrderDate) < 2016

AND CustomerID NOT IN (

SELECT DISTINCT CustomerID

FROM Sales.Orders

WHERE YEAR(OrderDate) >= 2016)

4.

SELECT ol.StockItemID, COUNT(\*) AS Quantity

FROM Purchasing.PurchaseOrderLines ol

JOIN Purchasing.PurchaseOrders o ON ol.PurchaseOrderID = o.PurchaseOrderID

WHERE YEAR(o.OrderDate) = 2013

GROUP BY ol.StockItemID

5.

SELECT StockItemName

FROM Warehouse.StockItems

WHERE LEN(StockItemName) > 10

6.

WITH cte AS(

SELECT ol.StockItemID, s.StateProvinceName

FROM Sales.Customers c

JOIN Sales.Orders o ON c.CustomerID = o.CustomerID ON YEAR(o.OrderDate) = 2014

JOIN Sales.OrderLines ol ON o.OrderID = ol.OrderID

JOIN Application.Cities ci ON c.DeliveryCityID = ci.CityID

JOIN Application.StateProvinces s ON ci.StateProvinceID = s.StateProvinceID

)

SELECT DISTINCT StockItemID

FROM cte

WHERE StockItemID NOT IN (SELECT StockItemID FROM cte WHERE StateProvinceName IN ('Alabama', 'Georgia'))

7.

SELECT s.StateProvinceName, AVG(DATEDIFF(d, o.OrderDate, i.ConfirmedDeliveryTime)) AS AvgProDate

FROM Sales.Orders o

JOIN Sales.Invoices i ON o.OrderID = i.OrderID

JOIN Sales.Customers c ON c.CustomerID = o.CustomerID

JOIN Application.Cities ci ON ci.CityID = c.DeliveryCityID

JOIN Application.StateProvinces s ON ci.StateProvinceID = s.StateProvinceID

GROUP BY s.StateProvinceName

8.

SELECT s.StateProvinceName, MONTH(o.OrderDate) AS month,

AVG(DATEDIFF(d, o.OrderDate, i.ConfirmedDeliveryTime)) AS AvgProcessDay

FROM Sales.Orders o

JOIN Sales.Invoices i ON o.OrderID = i.OrderID

JOIN Sales.Customers c ON c.CustomerID = o.CustomerID

JOIN Application.Cities ci ON ci.CityID = c.DeliveryCityID

JOIN Application.StateProvinces s ON ci.StateProvinceID = s.StateProvinceID

GROUP BY s.StateProvinceName, MONTH(o.OrderDate)

9.

WITH cte0 AS (

SELECT ol.StockItemID, SUM(ol.OrderedOuters) AS PurchaseQuantity

FROM Purchasing.PurchaseOrders o

JOIN Purchasing.PurchaseOrderLines ol ON o.PurchaseOrderID = ol.PurchaseOrderID

WHERE YEAR(o.OrderDate) = 2015

GROUP BY ol.StockItemID

),

cte1 AS (

SELECT ol.StockItemID, SUM(ol.Quantity) AS SaleQuantity

FROM Sales.Orders o

JOIN Sales.OrderLines ol ON o.OrderID = ol.OrderID

WHERE YEAR(o.OrderDate) = 2015

GROUP BY ol.StockItemID)

SELECT c0.StockItemID

FROM cte0 c0

LEFT JOIN cte1 c1 ON c0.StockItemID = c1.StockItemID

WHERE c1.SaleQuantity IS NULL

OR c0.PurchaseQuantity > c1.SaleQuantity

10.

WITH cte AS (

SELECT c.CustomerID

FROM Sales.Customers c

JOIN sales.Orders o ON c.CustomerID = o.CustomerID

JOIN sales.OrderLines ol ON ol.OrderID = o.OrderID

JOIN Warehouse.StockItemStockGroups s ON s.StockItemID = ol.StockItemID

JOIN Warehouse.StockGroups sg ON sg.StockGroupID = s.StockGroupID

WHERE YEAR(o.OrderDate) = 2016

AND sg.StockGroupName = 'Mugs'

GROUP BY c.CustomerID

HAVING COUNT(ol.Quantity) <= 10

)

SELECT c.CustomerName, c.PhoneNumber, p.FullName

FROM sales.Customers c

JOIN Application.People p ON c.PrimaryContactPersonID = p.PersonID

WHERE c.CustomerID IN (SELECT CustomerID

FROM cte)

11.

SELECT CityName

FROM Application.Cities

WHERE ValidFrom > '2015-01-01'

12.

SELECT s.StockItemName, CONCAT(c.DeliveryAddressLine2, ' ', c.DeliveryAddressLine1) AS DeliveryAdress,

st.StateProvinceName, ci.CityName, co.CountryName, c.CustomerName, p.FullName, c.PhoneNumber, ol.Quantity

FROM Sales.OrderLines ol

JOIN Sales.Orders o ON ol.OrderID = o.OrderID

JOIN Sales.Customers

FOR SYSTEM\_TIME AS OF '2014-07-01' c ON c.CustomerID = o.CustomerID

JOIN Application.People

FOR SYSTEM\_TIME AS OF '2014-07-01' p ON c.PrimaryContactPersonID = p.PersonID

JOIN Warehouse.StockItems

FOR SYSTEM\_TIME AS OF '2014-07-01' s ON s.StockItemID = ol.StockItemID

JOIN Application.Cities

FOR SYSTEM\_TIME AS OF '2014-07-01' ci ON c.DeliveryCityID = ci.CityID

JOIN Application.StateProvinces

FOR SYSTEM\_TIME AS OF '2014-07-01' st ON ci.StateProvinceID = st.StateProvinceID

JOIN Application.Countries

FOR SYSTEM\_TIME AS OF '2014-07-01' co ON st.CountryID = co.CountryID

13.

WITH cte0 AS (

SELECT s.StockGroupID, SUM(p.OrderedOuters) AS PurchaseQuantity

FROM Purchasing.PurchaseOrderLines p

JOIN Warehouse.StockItemStockGroups s ON p.StockItemID = s.StockItemID

GROUP BY s.StockGroupID

),

cte1 AS (

SELECT s.StockGroupID, SUM(o.Quantity) AS SaleQuantity

FROM Sales.OrderLines o

JOIN Warehouse.StockItemStockGroups s ON o.StockItemID = s.StockItemID

GROUP BY s.StockGroupID)

SELECT s.StockGroupName, ISNULL(c0.PurchaseQuantity, 0) AS PurchaseQuantity, ISNULL(c1.SaleQuantity, 0) AS SaleQuantity,

ISNULL(c0.PurchaseQuantity, 0) - ISNULL(c1.SaleQuantity, 0) AS RemainingQuantity

FROM Warehouse.StockGroups s

LEFT JOIN cte0 c0 ON s.StockGroupID = c0.StockGroupID

LEFT JOIN cte1 c1 ON s.StockGroupID = c1.StockGroupID

14.

WITH cte0 AS (

SELECT ol.StockItemID, c.DeliveryCityID, COUNT(\*) AS Delivery

FROM Sales.OrderLines ol

JOIN Sales.Orders o ON o.OrderID = ol.OrderID

JOIN sales.Customers c ON o.CustomerID = c.CustomerID

WHERE YEAR(o.OrderDate) = 2016

GROUP BY ol.StockItemID, c.DeliveryCityID),

cte1 AS(

SELECT StockItemID, DeliveryCityID

FROM (

SELECT StockItemID, DeliveryCityID,

DENSE\_RANK() OVER(PARTITION BY DeliveryCityId ORDER BY Delivery DESC) AS rnk

FROM cte0) a

WHERE rnk = 1

)

SELECT c.CityName, ISNULL(s.StockItemName, 'No Sale') AS MostDelivery

FROM cte1 c1 JOIN Warehouse.StockItems s ON c1.StockItemID = s.StockItemID

RIGHT JOIN Application.Cities c ON c1.DeliveryCityID = c.CityID

15.

SELECT OrderID

FROM Sales.Invoices

WHERE JSON\_VALUE(ReturnedDeliveryData, '$.Events[1].Comment') IS NOT NULL

16.

SELECT StockItemID, StockItemName

FROM Warehouse.StockItems

WHERE JSON\_VALUE(CustomFields, '$.CountryOfManufacture') = 'China'

17.

SELECT JSON\_VALUE(s.CustomFields, '$.CountryOfManufacture') AS Country, SUM(ol.Quantity) AS Quantity

FROM sales.Orders o

JOIN Sales.OrderLines ol ON o.OrderID = ol.OrderID

JOIN Warehouse.StockItems s ON ol.StockItemID = s.StockItemID

WHERE YEAR(o.OrderDate) = 2015

GROUP BY JSON\_VALUE(s.CustomFields, '$.CountryOfManufacture')

18.

CREATE VIEW Sales.StockItemByYear AS

WITH cte0 AS (

SELECT StockGroupName, 2013 AS [Year]

FROM Warehouse.StockGroups

UNION ALL

SELECT StockGroupName, [Year] + 1

FROM cte0

WHERE [Year] < 2017

),

cte1 AS (

SELECT YEAR(o.OrderDate) AS [Year], sg.StockGroupName, SUM(ol.Quantity) AS Quantity

FROM Sales.Orders o

JOIN Sales.OrderLines ol ON o.OrderID = ol.OrderID

JOIN Warehouse.StockItems s ON ol.StockItemID =s.StockItemID

JOIN Warehouse.StockItemStockGroups g ON g.StockItemID = s.StockItemID

JOIN Warehouse.StockGroups sg ON g.StockGroupID = sg.StockGroupID

WHERE YEAR(o.OrderDate) BETWEEN 2013 AND 2017

GROUP BY YEAR(o.OrderDate), sg.StockGroupName

),

cte2 AS (

SELECT c0.StockGroupName, c0.[Year], ISNULL(c1.Quantity, 0) AS Quantity

FROM cte0 c0

LEFT JOIN cte1 c1 ON c0.[Year] = c1.[Year]

AND c0.StockGroupName = c1.StockGroupName

)

SELECT StockGroupName, [2013], [2014], [2015], [2016], [2017]

FROM cte2

PIVOT

(

MIN(Quantity) FOR

Year IN ([2013], [2014], [2015], [2016], [2017])

) TBL

19.

CREATE VIEW Sales.StockItemByName AS

WITH cte0 AS (

SELECT StockGroupName, 2013 AS [Year]

FROM Warehouse.StockGroups

UNION ALL

SELECT StockGroupName, [Year] + 1

FROM cte0

WHERE [Year] < 2017

),

cte1 AS (

SELECT YEAR(o.OrderDate) AS [Year], sg.StockGroupName, SUM(ol.Quantity) AS Quantity

FROM Sales.Orders o

JOIN Sales.OrderLines ol ON o.OrderID = ol.OrderID

JOIN Warehouse.StockItems s ON ol.StockItemID =s.StockItemID

JOIN Warehouse.StockItemStockGroups g ON g.StockItemID = s.StockItemID

JOIN Warehouse.StockGroups sg ON g.StockGroupID = sg.StockGroupID

WHERE YEAR(o.OrderDate) BETWEEN 2013 AND 2017

GROUP BY YEAR(o.OrderDate), sg.StockGroupName

),

cte2 AS (

SELECT c0.StockGroupName, c0.[Year], ISNULL(c1.Quantity, 0) AS Quantity

FROM cte0 c0

LEFT JOIN cte1 c1 ON c0.[Year] = c1.[Year]

AND c0.StockGroupName = c1.StockGroupName

)

SELECT [Year], [Novelty Items], [Clothing], [Mugs], [T-Shirts],

[Airline Novelties], [Computing Novelties], [USB Novelties], [Furry Footwear], [Toys], [Packaging Materials]

FROM cte2

PIVOT

(

SUM(Quantity) FOR

StockGroupName IN ([Novelty Items], [Clothing], [Mugs], [T-Shirts],

[Airline Novelties], [Computing Novelties], [USB Novelties], [Furry Footwear], [Toys], [Packaging Materials])

) TBL

20.

CREATE FUNCTION Sales.OrderTotal (@orderid INT)

RETURNS TABLE

AS

RETURN (

SELECT OrderID, SUM(Quantity \* UnitPrice) AS Total

FROM Sales.OrderLines

WHERE OrderID = @orderid

GROUP BY OrderID

)

SELECT \*

FROM Sales.Invoices i

CROSS APPLY Sales.OrderTotal(OrderID) f

21.

CREATE SCHEMA ods

CREATE TABLE ods.Orders

(OrderID INT PRIMARY KEY,

OrderDate DATE,

OrderTotal DECIMAL(18, 2),

CustomerID INT)

CREATE PROCEDURE ods.OrderTotalOfDate

@OrderDate DATE

AS

IF EXISTS (SELECT 1 FROM ods.Orders WHERE OrderDate = @OrderDate)

BEGIN

RAISERROR('Date Exists ', 16, 1)

END

ELSE

BEGIN

BEGIN TRANSACTION

INSERT INTO ods.Orders

SELECT o.OrderID, o.OrderDate, f.Total, o.CustomerID

FROM Sales.Orders o

CROSS APPLY Sales.OrderTotal(OrderID) f

WHERE o.OrderDate = @OrderDate

COMMIT

END

22.

CREATE TABLE ods.StockItems(

StockItemID INT PRIMARY KEY,

StockItemName NVARCHAR(100) NOT NULL,

SupplierID INT NOT NULL,

ColorID INT NULL,

UnitPackageID INT NOT NULL,

OuterPackageID INT NOT NULL,

Brand NVARCHAR(50) NULL,

Size NVARCHAR(20) NULL,

LeadTimeDays INT NOT NULL,

QuantityPerOuter INT NOT NULL,

IsChillerStock BIT NOT NULL,

Barcode NVARCHAR(50) NULL,

TaxRate DECIMAL(18, 3) NOT NULL,

UnitPrice DECIMAL(18, 2) NOT NULL,

RecommendedRetailPrice DECIMAL(18, 2) NULL,

TypicalWeightPerUnit DECIMAL(18, 3) NOT NULL,

MarketingComments NVARCHAR(MAX) NULL,

InternalComments NVARCHAR(MAX) NULL,

CountryOfManufacture NVARCHAR(20) NULL,

[Range] NVARCHAR(20) NULL,

Shelflife NVARCHAR(20) NULL

)

MERGE INTO ods.StockItems AS T

USING Warehouse.StockItems AS R

ON T.StockItemID = R.StockItemID

WHEN NOT MATCHED

THEN INSERT VALUES (R.StockItemID, R.StockItemName, R.SupplierID, R.ColorID,

R.UnitPackageID, R.OuterPackageID, R.Brand, R.Size, R.LeadTimeDays,

R.QuantityPerOuter, R.IsChillerStock, R.Barcode, R.TaxRate, R.UnitPrice,

R.RecommendedRetailPrice, R.TypicalWeightPerUnit, R.MarketingComments,

R.InternalComments, JSON\_VALUE(R.CustomFields, '$.CountryOfManufacture'),

JSON\_VALUE(R.CustomFields, '$.Range'), JSON\_VALUE(R.CustomFields, '$.ShelfLife'));

23.

CREATE PROCEDURE ods.NewOrderTotalOfDate

@OrderDate DATE

AS

BEGIN TRANSACTION

DELETE FROM ods.Orders

WHERE OrderDate < @OrderDate

COMMIT

BEGIN TRANSACTION

MERGE ods.Orders T

USING (

SELECT o.OrderID, o.OrderDate, f.Total, o.CustomerID

FROM Sales.Orders o

CROSS APPLY Sales.OrderTotal(OrderID) f

WHERE DATEDIFF(d, @OrderDate, OrderDate) BETWEEN 1 AND 7

) R

ON T.OrderID = R.OrderID

WHEN NOT MATCHED

THEN INSERT VALUES (R.OrderID, R.OrderDate, R.Total, R.CustomerID);

COMMIT

24.

DECLARE @json NVARCHAR(MAX) = N'{

"PurchaseOrders":[

{

"StockItemName":"Panzer Video Game",

"Supplier":"7",

"UnitPackageId":"1",

"OuterPackageId":[

6,

7

],

"Brand":"EA Sports",

"LeadTimeDays":"5",

"QuantityPerOuter":"1",

"TaxRate":"6",

"UnitPrice":"59.99",

"RecommendedRetailPrice":"69.99",

"TypicalWeightPerUnit":"0.5",

"CountryOfManufacture":"Canada",

"Range":"Adult",

"OrderDate":"2018-01-01",

"DeliveryMethod":"Post",

"ExpectedDeliveryDate":"2018-02-02",

"SupplierReference":"WWI2308"

},

{

"StockItemName":"Panzer Video Game",

"Supplier":"5",

"UnitPackageId":"1",

"OuterPackageId":"7",

"Brand":"EA Sports",

"LeadTimeDays":"5",

"QuantityPerOuter":"1",

"TaxRate":"6",

"UnitPrice":"59.99",

"RecommendedRetailPrice":"69.99",

"TypicalWeightPerUnit":"0.5",

"CountryOfManufacture":"Canada",

"Range":"Adult",

"OrderDate":"2018-01-025",

"DeliveryMethod":"Post",

"ExpectedDeliveryDate":"2018-02-02",

"SupplierReference":"269622390"

}

]

}'

SELECT \*

FROM OPENJSON(@json, '$.PurchaseOrders')

WITH (

StockName NVARCHAR(50) '$.StockItemName',

SupplierID INT '$.Supplier',

ColorID INT '$.Color',

UnitPackageID INT '$.UnitPackageId',

OuterPackageID NVARCHAR(MAX) '$.OuterPackageId',

Brand NVARCHAR(20) '$.Brand',

Size NVARCHAR(20) '$.Size',

LeadTimeDays INT '$.LeadTimeDays',

QuantityPerOuter INT '$.QuantityPerOuter',

IsChillerStock BIT '$.IsChillerStock',

Barcode NVARCHAR(50) '$.Barcode',

TaxRate DECIMAL(18, 3) '$.TaxRate',

UnitPrice DECIMAL(18, 2) '$.UnitPrice',

RecommendedRetailPrice DECIMAL(18, 2) '$.RecommendedRetailPrice'

)

25.

SELECT Year AS Year,

[Novelty Items] AS 'StockGroup.Novelty Items',

[Clothing] AS 'StockGroup.Clothing',

[Mugs] AS 'StockGroup.Mugs',

[T-Shirts] AS 'StockGroup.T-Shirts',

[Airline Novelties] AS 'StockGroup.Airline Novelties',

[Computing Novelties] AS 'StockGroup.Computing Novelties',

[USB Novelties] AS 'StockGroup.USB Novelties',

[Furry Footwear] AS 'StockGroup.Furry Footwear',

[Toys] AS 'StockGroup.Toys',

[Packaging Materials] AS 'StockGroup.Packaging Materials'

FROM Sales.StockItemByName

FOR JSON PATH

26.

SELECT Year AS '@Year',

[Novelty Items] AS NoveltyItems,

[Clothing],

[Mugs],

[T-Shirts],

[Airline Novelties] AS AirlineNovelties,

[Computing Novelties] AS ComputingNovelties,

[USB Novelties] AS USBNovelties,

[Furry Footwear] AS FurryFootwear,

[Toys],

[Packaging Materials] AS PackagingMaterials

FROM Sales.StockItemByName

FOR XML PATH('StockItems')

27.

CREATE TABLE ods.ConfirmedDeviveryJson

(ID INT PRIMARY KEY,

Date DATE,

Value NVARCHAR(MAX))

CREATE PROCEDURE ods.InsertInvoicesOfDate

@Date DATE

AS

BEGIN

DECLARE @i NVARCHAR(MAX) = '';

DECLARE @oi NVARCHAR(MAX) = '';

DECLARE @column NVARCHAR(MAX);

DECLARE @query NVARCHAR(MAX);

DECLARE @json NVARCHAR(MAX);

SELECT @i = @i + 'i.' + COLUMN\_NAME + ', '

FROM (SELECT COLUMN\_NAME

FROM(

SELECT COLUMN\_NAME

FROM INFORMATION\_SCHEMA.COLUMNS

WHERE TABLE\_NAME = N'Invoices') a

WHERE COLUMN\_NAME != 'InvoiceDate'

) b

SELECT @oi = @oi + 'oi.' + COLUMN\_NAME + ', '

FROM (

SELECT COLUMN\_NAME

FROM(

SELECT COLUMN\_NAME

FROM INFORMATION\_SCHEMA.COLUMNS

WHERE TABLE\_NAME = 'InvoiceLines') a

WHERE COLUMN\_NAME NOT IN ('InvoiceID', 'LastEditedBy', 'LastEditedWhen')

) b

SET @column = @i + SUBSTRING(@oi, 0, LEN(@oi))

SET @query = 'SET @json = (SELECT ID = oi1.InvoiceLineID, Date = i1.InvoiceDate,

Value = (

SELECT ' + @column + '

FROM Sales.Invoices i

JOIN Sales.InvoiceLines oi ON i.InvoiceID = oi.InvoiceID

WHERE oi.InvoiceLineID = oi1.InvoiceLineID

FOR JSON PATH, ROOT('Value'), INCLUDE\_NULL\_VALUES)

FROM Sales.Invoices i1

JOIN Sales.InvoiceLines oi1 ON i1.InvoiceID = oi1.InvoiceID

WHERE i1.InvoiceDate = @Date

FOR JSON PATH, INCLUDE\_NULL\_VALUES)'

EXEC SP\_EXECUTESQL @query, N'@Date DATE, @json NVARCHAR(MAX) OUT', @Date, @json OUT

INSERT INTO ods.ConfirmedDeviveryJson

SELECT \*

FROM OPENJSON(@json)

WITH(

ID INT '$.ID',

Date DATE '$.Date',

Value NVARCHAR(MAX) '$.Value' AS JSON)

END

DECLARE @loop NVARCHAR(MAX) = ''

SELECT @loop = @loop + N'EXEC ods.InsertInvoicesOfDate ''' + CAST(InvoiceDate AS NVARCHAR)+ '''; '

FROM (SELECT DISTINCT InvoiceDate

FROM Sales.Invoices

WHERE CustomerID = 1) a

EXEC SP\_EXECUTESQL @loop

SELECT \*

FROM ods.ConfirmedDeviveryJson

WHERE JSON\_VALUE(Value, '$.Value[0].CustomerID') = 1