Erik Kim

GROUP 4 – PROJECT 1

CSCI331 10:45 Professor Heller

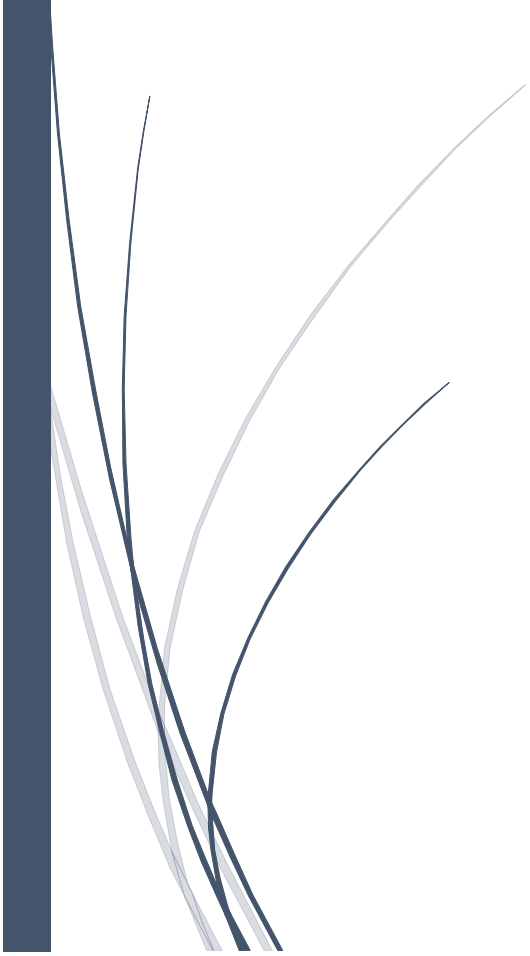


Table of Contents

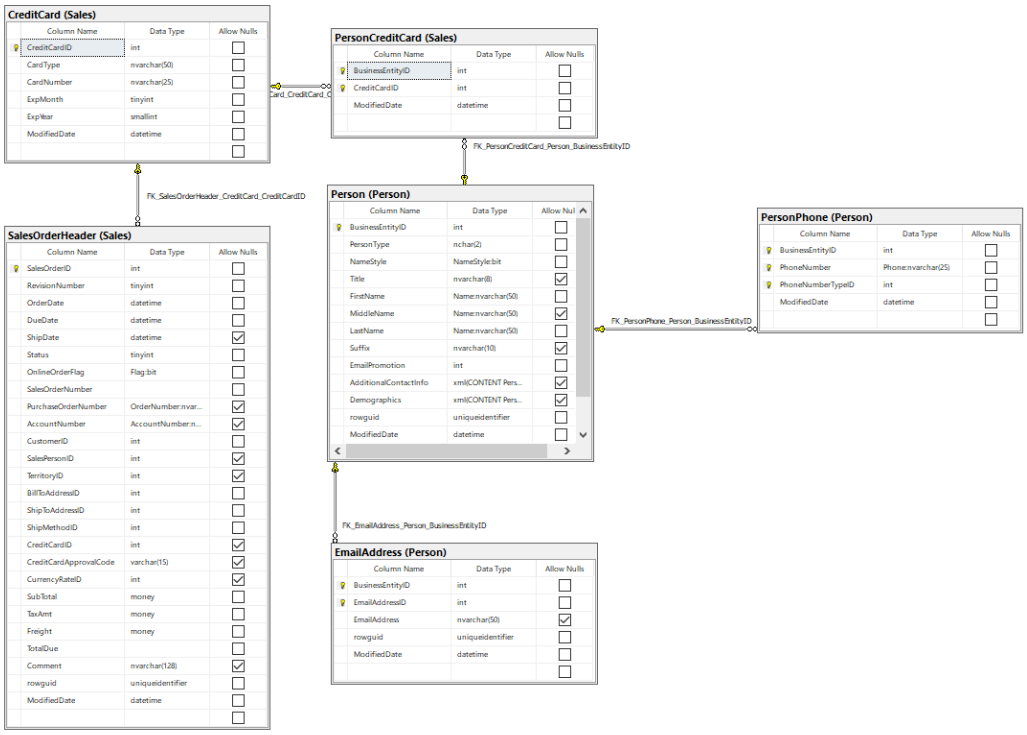
All information is formatted as such:

Proposition - Standard/Key View – Tables - Relational and JSON Output

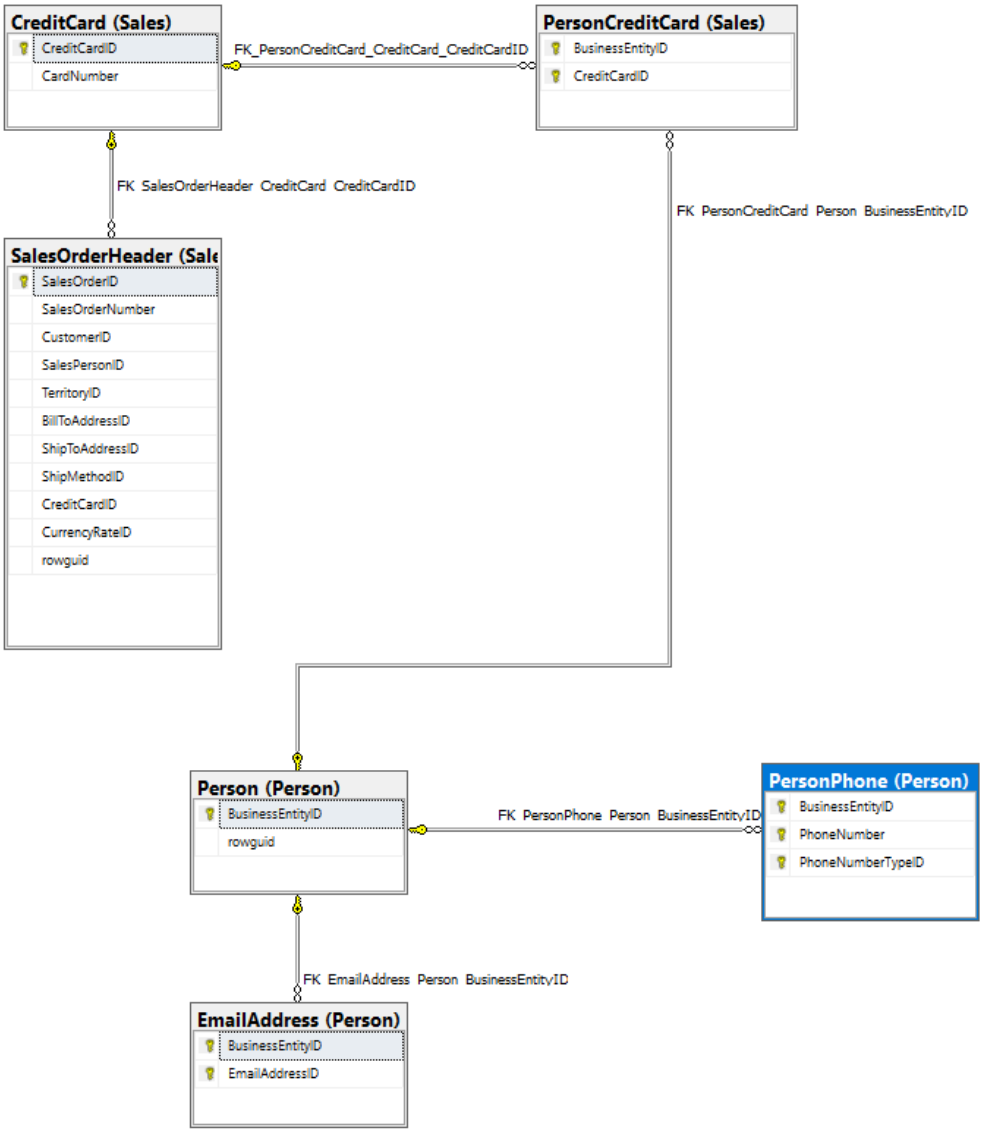
Subsystem Diagrams

Person

Standard View:

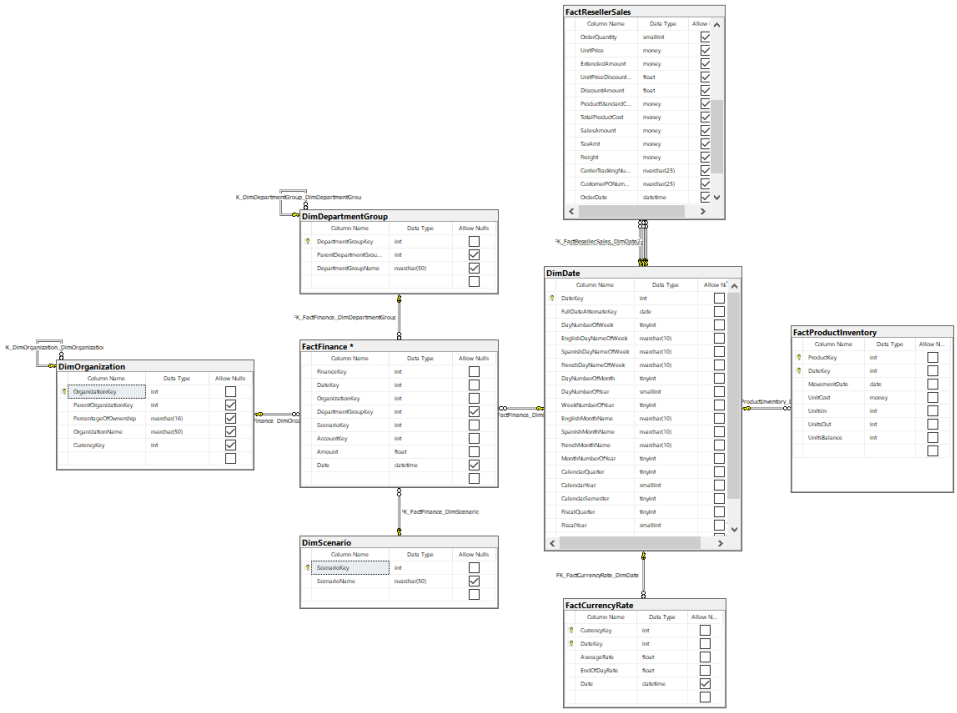


Key View:

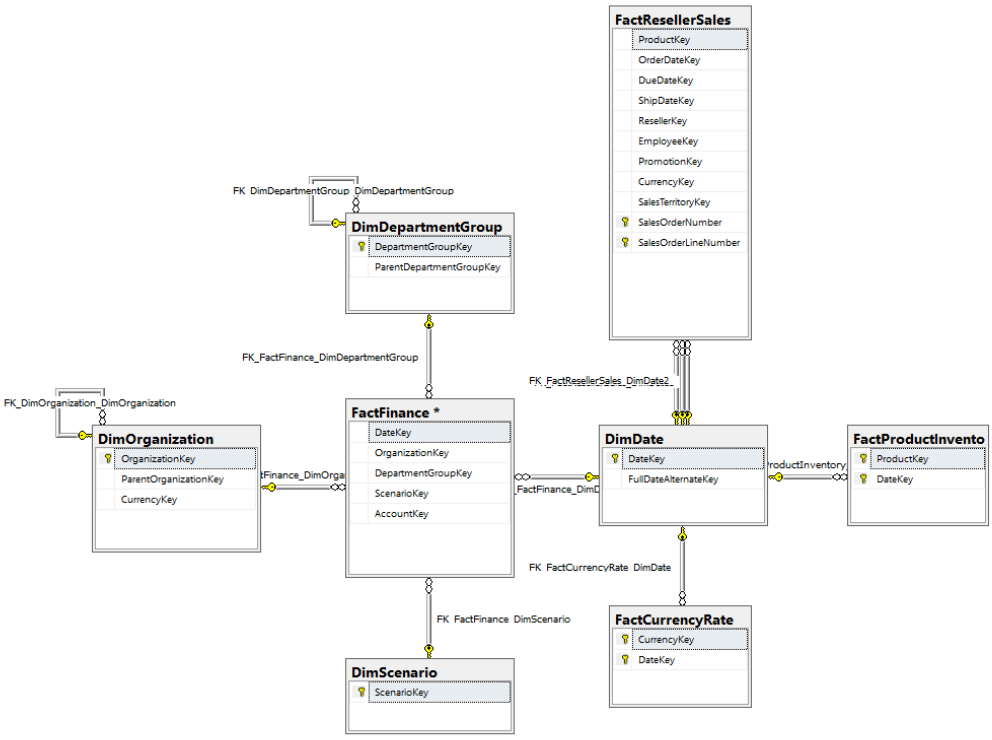

CreditCard (Sales)
CreditCardID
CardNumber
FK_PersonCreditCard_CreditCard_CreditCardID
PersonCreditCard (Sales)
BusinessEntityID
CreditCardID
FK SalesOrderHeader CreditCard CreditCardID
FK PersonCreditCard Person BusinessEntityID
SalesOrderHeader (Sale
SalesOrderID
SalesOrderNumber
CustomerID
SalesPersonID
TerritoryID
BillToAddressID
ShipToAddressID
ShipMethodID
CreditCardID
CurrencyRateID
rowguid
Person (Person)
BusinessEntityID
rowguid
FK PersonPhone Person BusinessEntityID
PersonPhone (Person)
BusinessEntityID
PhoneNumber
PhoneNumberTypeID
FK EmailAddress Person BusinessEntityID
EmailAddress (Person)
BusinessEntityID
EmailAddressID

Dimdate

Standard View:

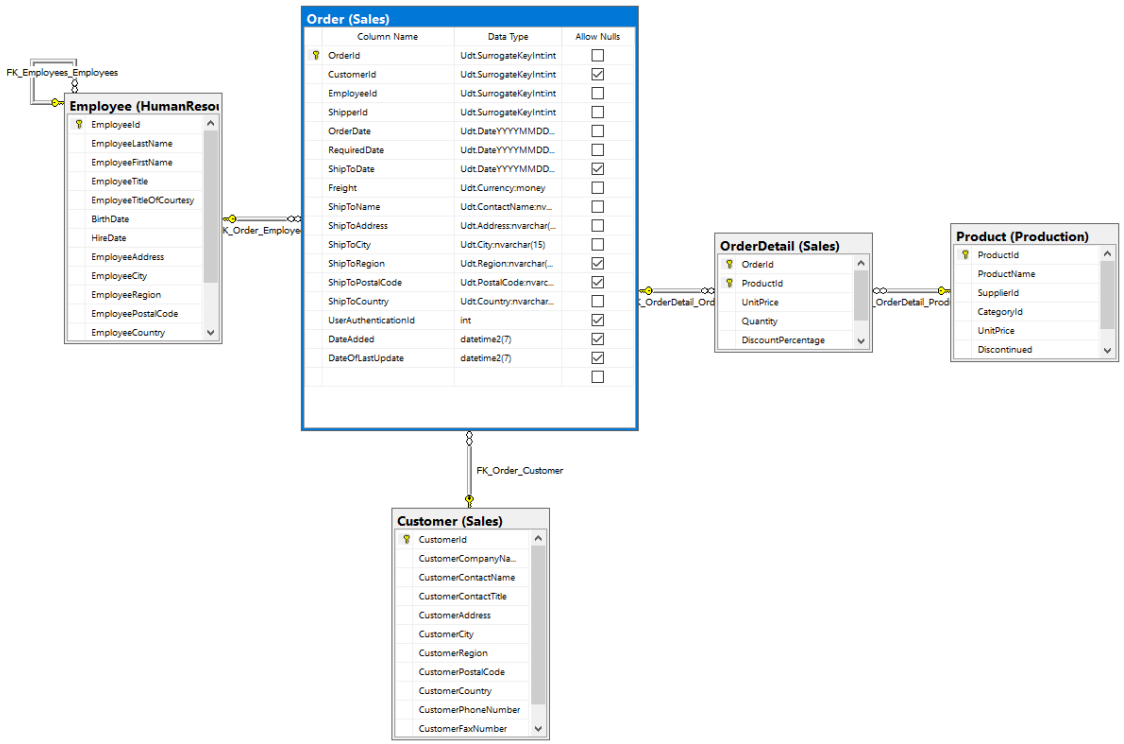


Key View:

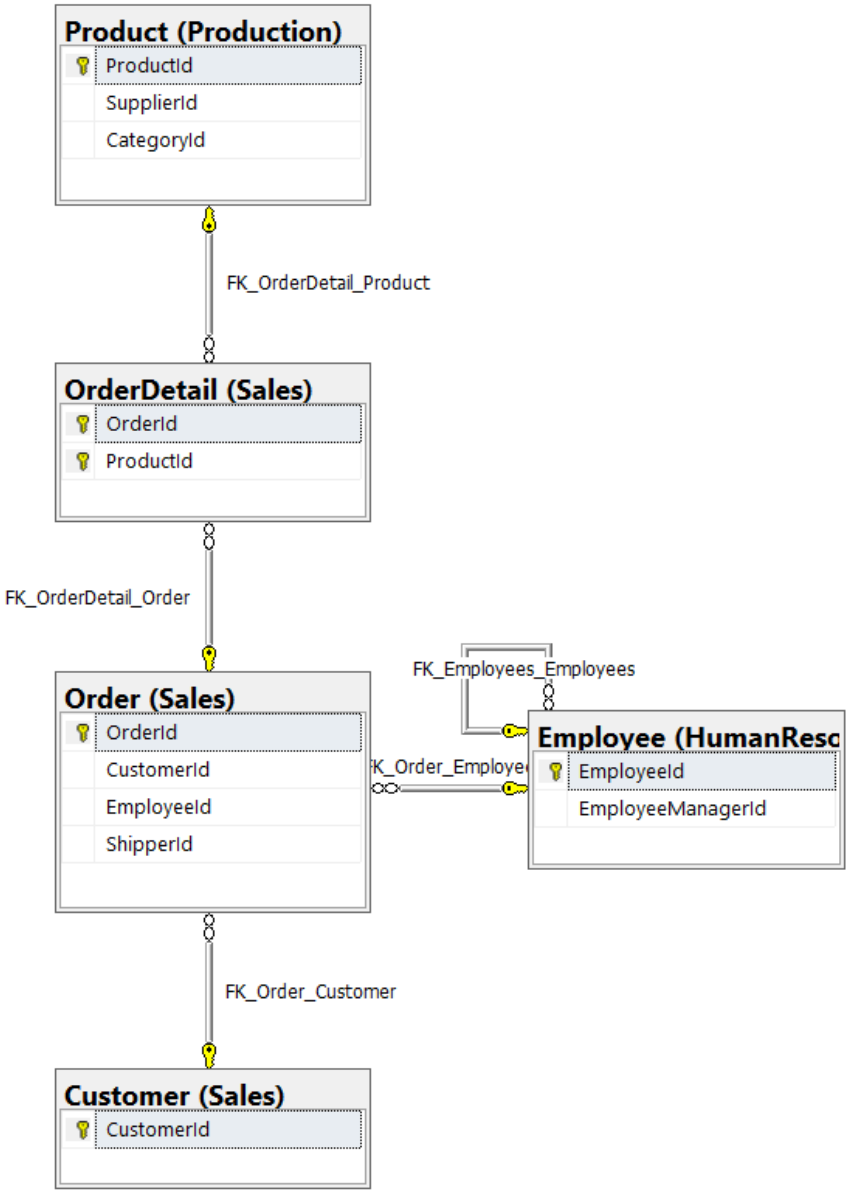

FactResellerSales
ProductKey
OrderDateKey
DueDateKey
ShipDateKey
ResellerKey
EmployeeKey
PromotionKey
CurrencyKey
SalesTerritoryKey
SalesOrderNumber
SalesOrderLineNumber
FK_DimDepartmentGroup_DimDepartmentGroup
DimDepartmentGroup
DepartmentGroupKey
ParentDepartmentGroupKey
FK_FactFinance_DimDepartmentGroup
FK_FactResellerSales_DimDate2
FK_DimOrganization_DimOrganization
DimOrganization
OrganizationKey
ParentOrganizationKey
CurrencyKey
FactFinance *
DateKey
OrganizationKey
DepartmentGroupKey
ScenarioKey
AccountKey
DimDate
DateKey
FullDateAlternateKey
FactProductInvento
ProductKey
DateKey
FK_FactCurrencyRate_DimDate
FK_FactFinance_DimScenario
FactCurrencyRate
CurrencyKey
DateKey
DimScenario
ScenarioKey

Orders

Standard View:



Key View:


Product (Production)
ProductId
SupplierId
CategoryId
FK_OrderDetail_Product
OrderDetail (Sales)
OrderId
ProductId
FK_OrderDetail_Order
FK_Employees_Employees
Order (Sales)
OrderId
CustomerId
EmployeeId
ShipperId
Employee (HumanResc
EmployeeId
EmployeeManagerId
FK_Order_Employe
FK_Order_Customer
Customer (Sales)
CustomerId

Simple Queries

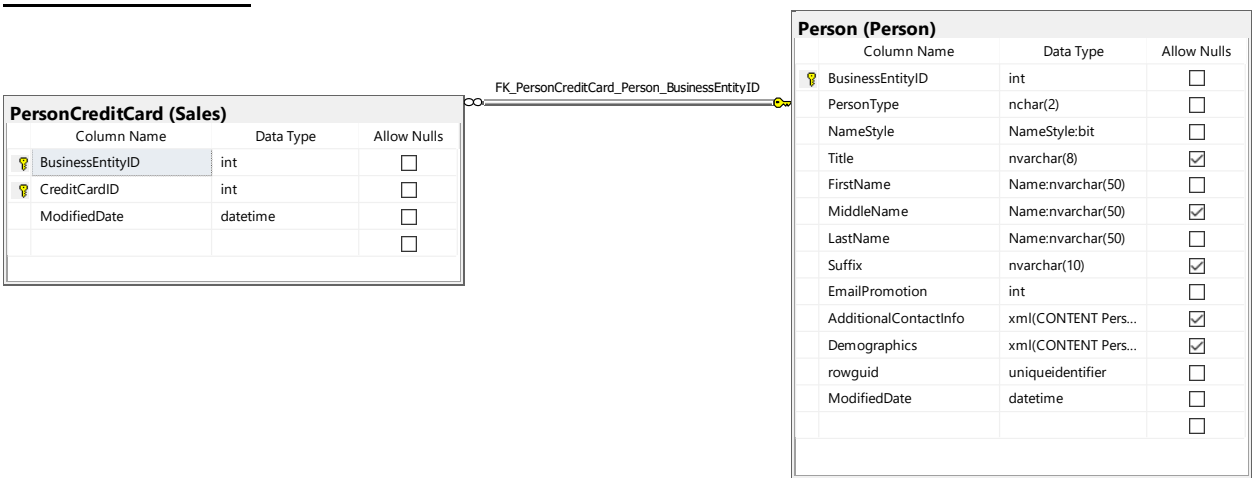
- All simple queries use the AdventureWorks2017 database
- All simple queries use the Person subsystem

Simple Query Propositions:

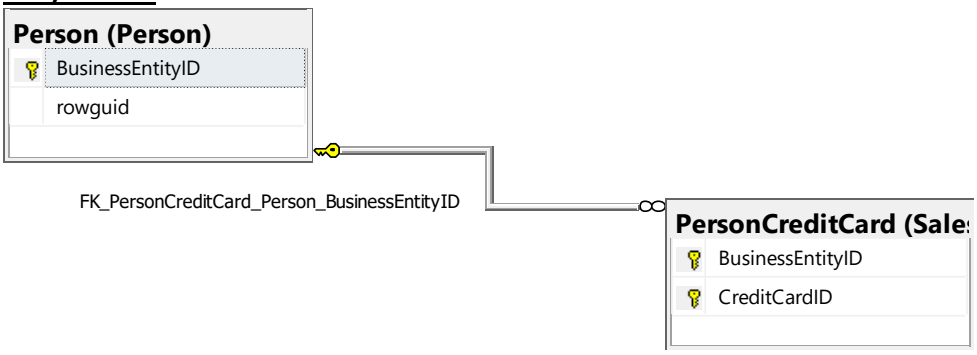
Simple Query 1:

```
--Find credit card IDs of all business entities that have used credit cards, using
AdventureWorks2017.
```

Standard view:



Key view:



Columns from tables

Table Name	Column Names
PersonCreditCard	CreditCardID
Person	BusinessEntityID

Order By

Table Name	Column Name	Sort Order
PersonCreditCard	CreditCardID	ASC

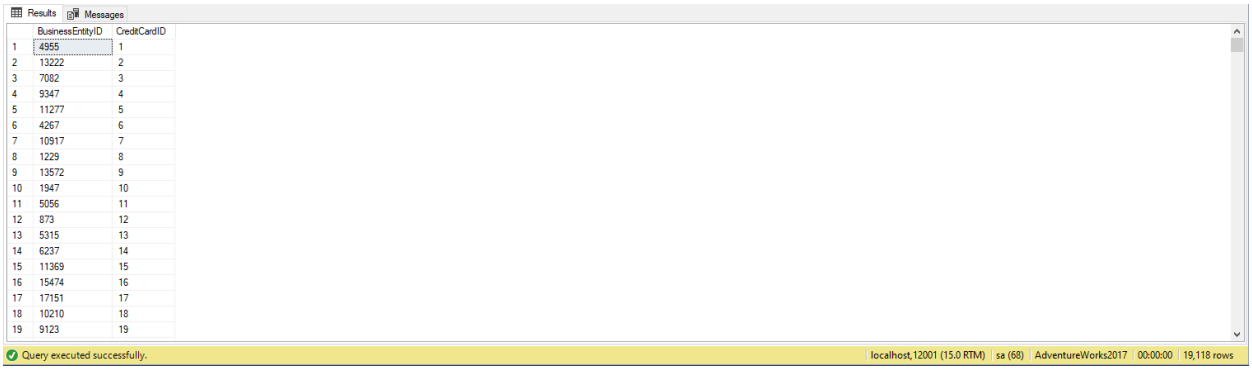
Without JSON:

```
SELECT DISTINCT
       p.BusinessEntityID,
       pc.CreditCardID
FROM Person.Person AS p
    INNER JOIN Sales.PersonCreditCard AS pc
        ON p.BusinessEntityID = pc.BusinessEntityID
ORDER BY pc.CreditCardID;
--FOR JSON PATH, ROOT ('BusinessCCID'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT DISTINCT
       p.BusinessEntityID,
       pc.CreditCardID
FROM Person.Person AS p
    INNER JOIN Sales.PersonCreditCard AS pc
        ON p.BusinessEntityID = pc.BusinessEntityID
ORDER BY pc.CreditCardID
FOR JSON PATH, ROOT ('BusinessCCID'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (19118)

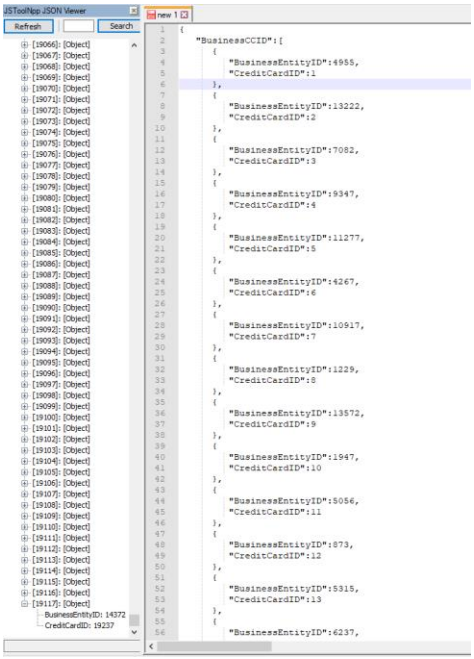


Results | Messages

	BusinessEntityID	CreditCardID
1	4955	1
2	13222	2
3	7082	3
4	9347	4
5	11277	5
6	4267	6
7	10917	7
8	1229	8
9	13572	9
10	1947	10
11	5056	11
12	873	12
13	5315	13
14	6237	14
15	11369	15
16	15474	16
17	17151	17
18	10210	18
19	9123	19

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | AdventureWorks2017 | 00:00:00 | 19,118 rows

Sample JSON Output with total number of rows returned (19118)



JSTool/Npp JSON Viewer — Refresh | Search

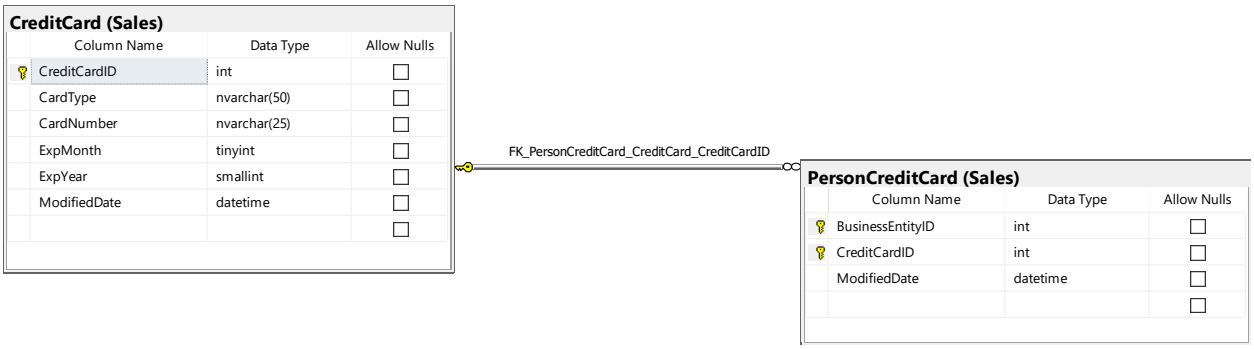
new 1


```
{
  "BusinessCCID":[
      {
         "BusinessEntityID":4955,
         "CreditCardID":1
      },
      {
         "BusinessEntityID":13222,
         "CreditCardID":2
      },
      {
         "BusinessEntityID":7082,
         "CreditCardID":3
      },
      {
         "BusinessEntityID":9347,
         "CreditCardID":4
      },
      {
         "BusinessEntityID":11277,
         "CreditCardID":5
      },
      {
         "BusinessEntityID":4267,
         "CreditCardID":6
      },
      {
         "BusinessEntityID":10917,
         "CreditCardID":7
      },
      {
         "BusinessEntityID":1229,
         "CreditCardID":8
      },
      {
         "BusinessEntityID":13572,
         "CreditCardID":9
      },
      {
         "BusinessEntityID":1947,
         "CreditCardID":10
      },
      {
         "BusinessEntityID":5056,
         "CreditCardID":11
      },
      {
         "BusinessEntityID":873,
         "CreditCardID":12
      },
      {
         "BusinessEntityID":5315,
         "CreditCardID":13
      },
      {
         "BusinessEntityID":6237,
```

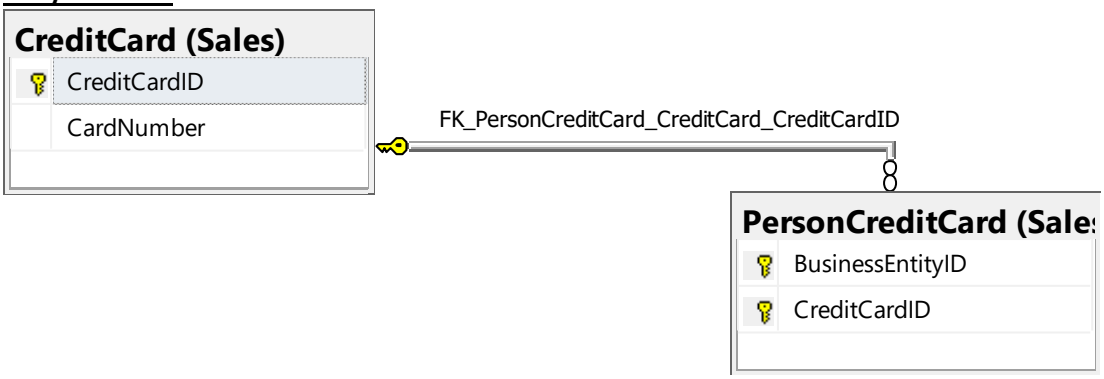
Simple Query 2:

```
--Find credit card information of all business entities that have used credit cards using
their credit card IDs, using AdventureWorks2017.
```

Standard view:



Key View:



Columns from tables

Table Name	Column Names
CreditCard	CardType CardNumber ExpMonth ExpYear ModifiedDate
PersonCreditCard	BusinessEntityID CreditCardID

Order By

Table Name	Column Name	Sort Order
PersonCreditCard	CreditCardID	ASC

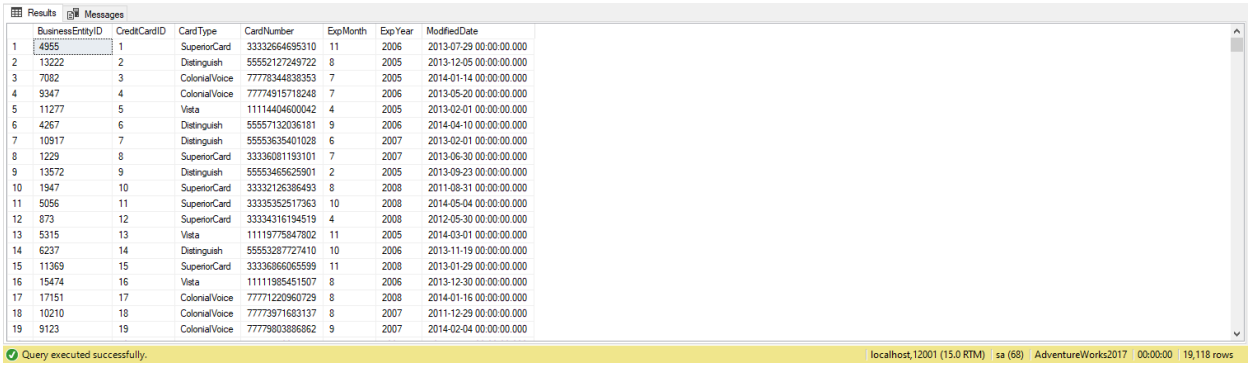
Without JSON:

```
SELECT DISTINCT
       pc.BusinessEntityID,
          pc.CreditCardID,
          cc.CardType,
          cc.CardNumber,
          cc.ExpMonth,
          cc.ExpYear,
          cc.ModifiedDate
FROM Sales.CreditCard AS cc
    INNER JOIN Sales.PersonCreditCard AS pc
        ON cc.CreditCardID = pc.CreditCardID
ORDER BY pc.CreditCardID;
--FOR JSON PATH, ROOT ('BusinessCCInfo'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT DISTINCT
       pc.BusinessEntityID,
          pc.CreditCardID,
          cc.CardType,
          cc.CardNumber,
          cc.ExpMonth,
          cc.ExpYear,
          cc.ModifiedDate
FROM Sales.CreditCard AS cc
    INNER JOIN Sales.PersonCreditCard AS pc
        ON cc.CreditCardID = pc.CreditCardID
ORDER BY pc.CreditCardID;
FOR JSON PATH, ROOT ('BusinessCCInfo'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (19118)

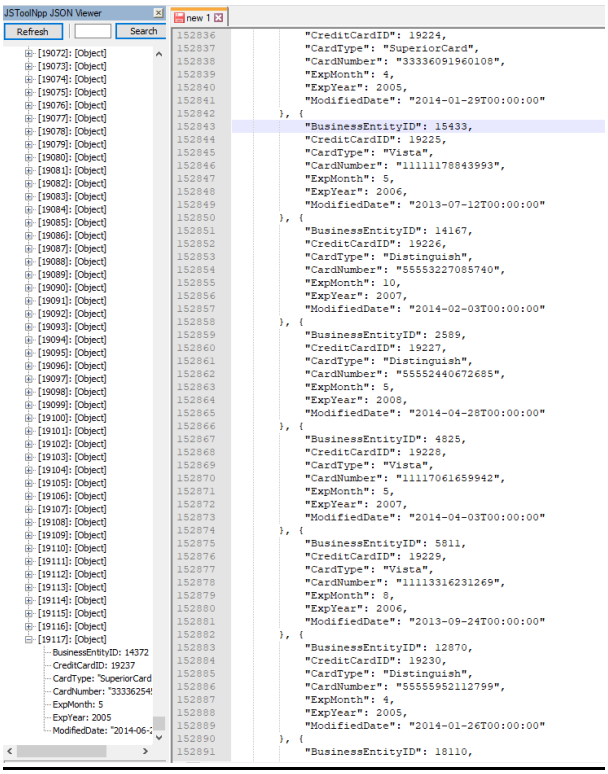


Results | Messages

	BusinessEntityID	CreditCardID	CardType	CardNumber	ExpMonth	ExpYear	ModifiedDate
1	4955	1	SuperiorCard	33332664695310	11	2006	2013-07-29 00:00:00.000
2	13222	2	Distinguish	55552127249722	8	2005	2013-12-05 00:00:00.000
3	7082	3	ColonialVoice	77778344838353	7	2005	2014-01-14 00:00:00.000
4	9347	4	ColonialVoice	77774915718248	7	2006	2013-05-20 00:00:00.000
5	11277	5	Vista	11114404600042	4	2005	2013-02-01 00:00:00.000
6	4267	6	Distinguish	55557132036181	9	2006	2014-04-10 00:00:00.000
7	10917	7	Distinguish	55553635401028	6	2007	2013-02-01 00:00:00.000
8	1229	8	SuperiorCard	33336081193101	7	2007	2013-06-30 00:00:00.000
9	13572	9	Distinguish	55553465625901	2	2005	2013-09-23 00:00:00.000
10	1947	10	SuperiorCard	33332126386493	8	2008	2011-08-31 00:00:00.000
11	5056	11	SuperiorCard	33335352517363	10	2008	2014-05-04 00:00:00.000
12	873	12	SuperiorCard	33334316194519	4	2008	2012-05-30 00:00:00.000
13	5315	13	Vista	11119775847802	11	2005	2014-03-01 00:00:00.000
14	6237	14	Distinguish	55553287727410	10	2006	2013-11-19 00:00:00.000
15	11369	15	SuperiorCard	33336866065599	11	2008	2013-01-29 00:00:00.000
16	15474	16	Vista	11111985451507	8	2006	2013-12-30 00:00:00.000
17	17151	17	ColonialVoice	77771220960729	8	2008	2014-01-16 00:00:00.000
18	10210	18	ColonialVoice	77773971683137	8	2007	2011-12-29 00:00:00.000
19	9123	19	ColonialVoice	77779803886862	9	2007	2014-02-04 00:00:00.000

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | AdventureWorks2017 | 00:00:00 | 19,118 rows

Sample JSON Output with total number of rows returned (19118)



JSToolNpp JSON Viewer

Refresh | Search

[19072]: [Object]
[19073]: [Object]
[19074]: [Object]
[19075]: [Object]
[19076]: [Object]
[19077]: [Object]
[19078]: [Object]
[19079]: [Object]
[19080]: [Object]
[19081]: [Object]
[19082]: [Object]
[19083]: [Object]
[19084]: [Object]
[19085]: [Object]
[19086]: [Object]
[19087]: [Object]
[19088]: [Object]
[19089]: [Object]
[19090]: [Object]
[19091]: [Object]
[19092]: [Object]
[19093]: [Object]
[19094]: [Object]
[19095]: [Object]
[19096]: [Object]
[19097]: [Object]
[19098]: [Object]
[19099]: [Object]
[19100]: [Object]
[19101]: [Object]
[19102]: [Object]
[19103]: [Object]
[19104]: [Object]
[19105]: [Object]
[19106]: [Object]
[19107]: [Object]
[19108]: [Object]
[19109]: [Object]
[19110]: [Object]
[19111]: [Object]
[19112]: [Object]
[19113]: [Object]
[19114]: [Object]
[19115]: [Object]
[19116]: [Object]
[19117]: [Object]
BusinessEntityID: 14372
CreditCardID: 19237
CardType: "SuperiorCard
CardNumber: "33336254
ExpMonth: 5
ExpYear: 2005
ModifiedDate: "2014-06-2

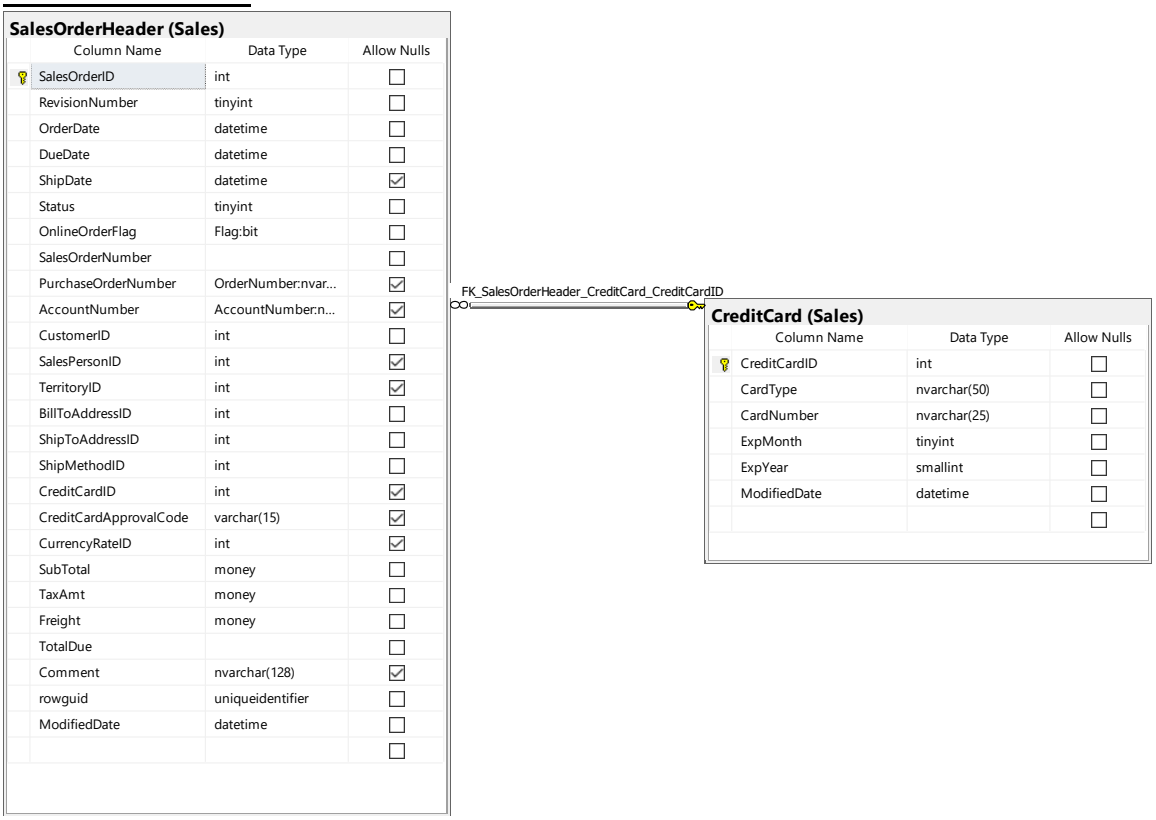
new 1


```
            "CreditCardID": 19224,
            "CardType": "SuperiorCard",
            "CardNumber": "33336091960108",
            "ExpMonth": 4,
            "ExpYear": 2005,
            "ModifiedDate": "2014-01-29T00:00:00"
        }, {
            "BusinessEntityID": 15433,
            "CreditCardID": 19225,
            "CardType": "Vista",
            "CardNumber": "11111178843993",
            "ExpMonth": 5,
            "ExpYear": 2006,
            "ModifiedDate": "2013-07-12T00:00:00"
        }, {
            "BusinessEntityID": 14167,
            "CreditCardID": 19226,
            "CardType": "Distinguish",
            "CardNumber": "55553227085740",
            "ExpMonth": 10,
            "ExpYear": 2007,
            "ModifiedDate": "2014-02-03T00:00:00"
        }, {
            "BusinessEntityID": 2589,
            "CreditCardID": 19227,
            "CardType": "Distinguish",
            "CardNumber": "55552440672685",
            "ExpMonth": 5,
            "ExpYear": 2008,
            "ModifiedDate": "2014-04-28T00:00:00"
        }, {
            "BusinessEntityID": 4825,
            "CreditCardID": 19228,
            "CardType": "Vista",
            "CardNumber": "11117061659942",
            "ExpMonth": 5,
            "ExpYear": 2007,
            "ModifiedDate": "2014-04-03T00:00:00"
        }, {
            "BusinessEntityID": 5811,
            "CreditCardID": 19229,
            "CardType": "Vista",
            "CardNumber": "11113316231269",
            "ExpMonth": 8,
            "ExpYear": 2006,
            "ModifiedDate": "2013-09-24T00:00:00"
        }, {
            "BusinessEntityID": 12870,
            "CreditCardID": 19230,
            "CardType": "Distinguish",
            "CardNumber": "55555952112799",
            "ExpMonth": 4,
            "ExpYear": 2005,
            "ModifiedDate": "2014-01-26T00:00:00"
        }, {
            "BusinessEntityID": 18110,
```

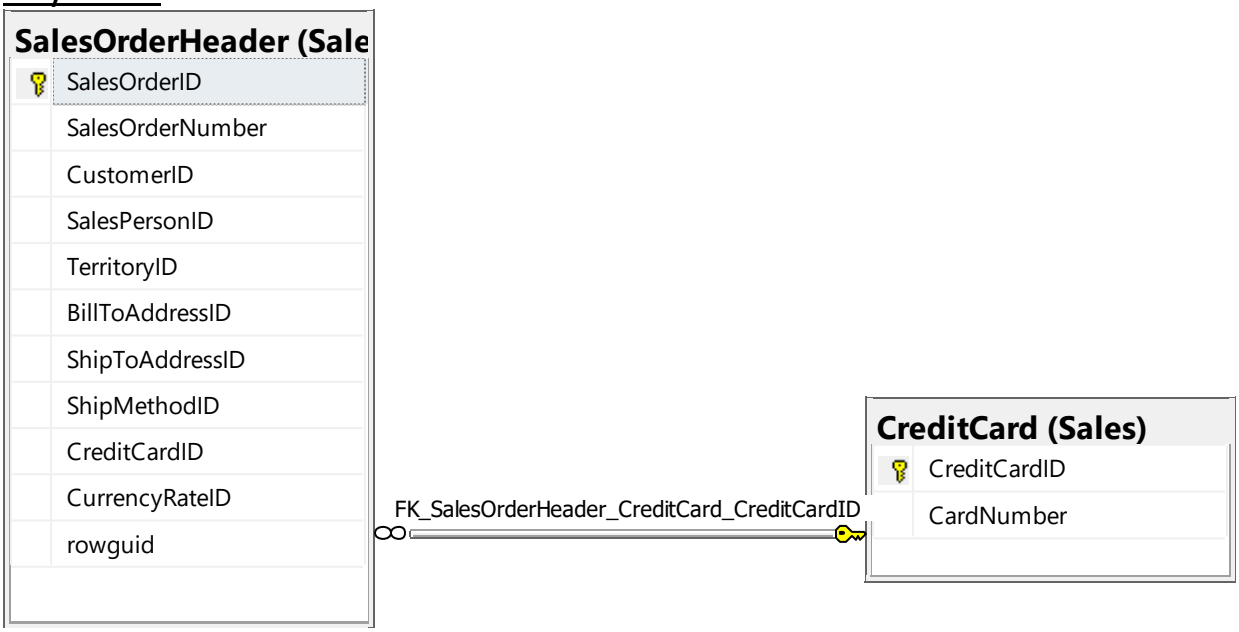
Simple Query 3:

```
--Using credit card ID, find OrderID, OrderDate, DueDate, ShipDate and Status, using AdventureWorks2017.
```

Standard view:



Key View:



Columns from tables

Table Name	Column Names
CreditCard	CreditCardID
SalesOrderHeader	OrderDate DueDate ShipDate

Order By

Table Name	Column Name	Sort Order
SalesOrderHeader	CreditCardID OrderDate	ASC ASC

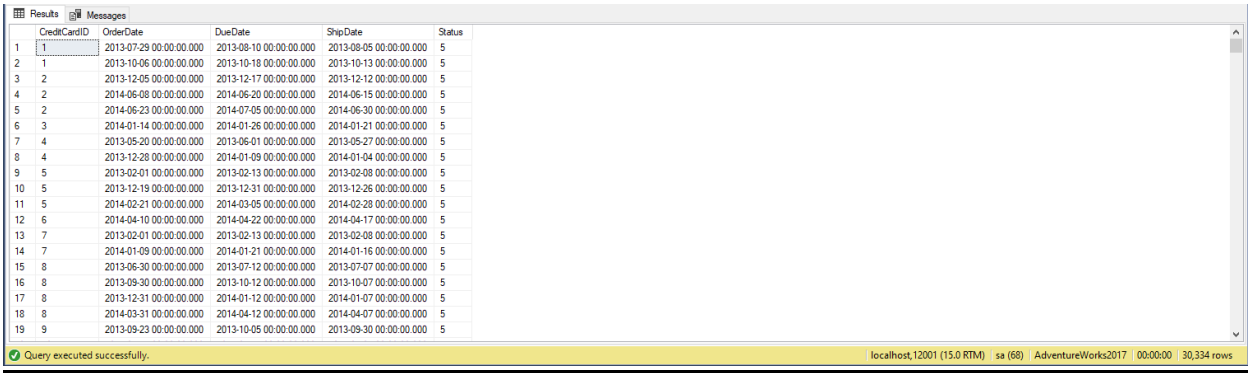
Without JSON:

```
SELECT cc.CreditCardID,
       oh.OrderDate,
       oh.DueDate,
       oh.ShipDate,
       oh.[Status]
FROM Sales.CreditCard AS cc
    INNER JOIN Sales.SalesOrderHeader AS oh
        ON cc.CreditCardID = oh.CreditCardID
ORDER BY oh.CreditCardID, oh.OrderDate;
--FOR JSON PATH, ROOT ('CCOrderInfo'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT cc.CreditCardID,
       oh.OrderDate,
       oh.DueDate,
       oh.ShipDate,
       oh.[Status]
FROM Sales.CreditCard AS cc
    INNER JOIN Sales.SalesOrderHeader AS oh
        ON cc.CreditCardID = oh.CreditCardID
ORDER BY oh.CreditCardID, oh.OrderDate
FOR JSON PATH, ROOT ('CCOrderInfo'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (30344)

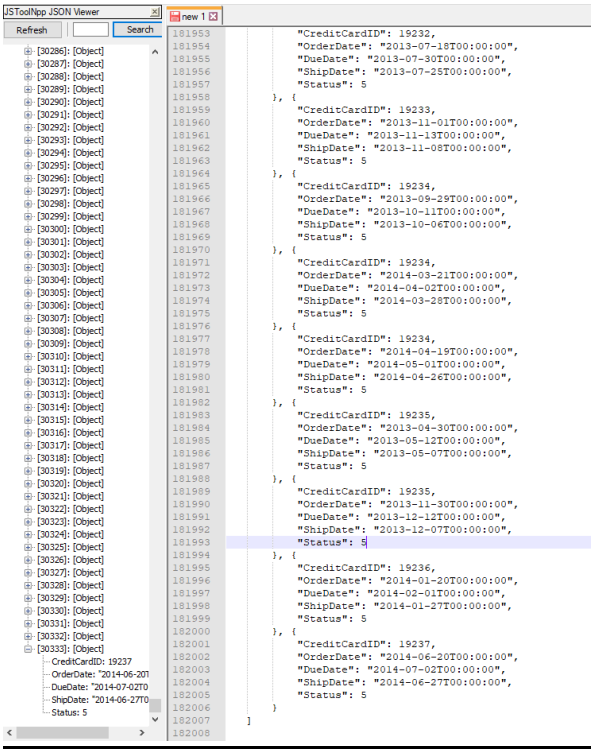


Results | Messages

	CreditCardID	OrderDate	DueDate	ShipDate	Status
1	1	2013-07-29 00:00:00.000	2013-08-10 00:00:00.000	2013-08-05 00:00:00.000	5
2	1	2013-10-06 00:00:00.000	2013-10-18 00:00:00.000	2013-10-13 00:00:00.000	5
3	2	2013-12-05 00:00:00.000	2013-12-17 00:00:00.000	2013-12-12 00:00:00.000	5
4	2	2014-06-08 00:00:00.000	2014-06-20 00:00:00.000	2014-06-15 00:00:00.000	5
5	2	2014-06-23 00:00:00.000	2014-07-05 00:00:00.000	2014-06-30 00:00:00.000	5
6	3	2014-01-14 00:00:00.000	2014-01-26 00:00:00.000	2014-01-21 00:00:00.000	5
7	4	2013-05-20 00:00:00.000	2013-06-01 00:00:00.000	2013-05-27 00:00:00.000	5
8	4	2013-12-28 00:00:00.000	2014-01-09 00:00:00.000	2014-01-04 00:00:00.000	5
9	5	2013-02-01 00:00:00.000	2013-02-13 00:00:00.000	2013-02-08 00:00:00.000	5
10	5	2013-12-19 00:00:00.000	2013-12-31 00:00:00.000	2013-12-26 00:00:00.000	5
11	5	2014-02-21 00:00:00.000	2014-03-05 00:00:00.000	2014-02-28 00:00:00.000	5
12	6	2014-04-10 00:00:00.000	2014-04-22 00:00:00.000	2014-04-17 00:00:00.000	5
13	7	2013-02-01 00:00:00.000	2013-02-13 00:00:00.000	2013-02-08 00:00:00.000	5
14	7	2014-01-09 00:00:00.000	2014-01-21 00:00:00.000	2014-01-16 00:00:00.000	5
15	8	2013-06-30 00:00:00.000	2013-07-12 00:00:00.000	2013-07-07 00:00:00.000	5
16	8	2013-09-30 00:00:00.000	2013-10-12 00:00:00.000	2013-10-07 00:00:00.000	5
17	8	2013-12-31 00:00:00.000	2014-01-12 00:00:00.000	2014-01-07 00:00:00.000	5
18	8	2014-03-31 00:00:00.000	2014-04-12 00:00:00.000	2014-04-07 00:00:00.000	5
19	9	2013-09-23 00:00:00.000	2013-10-05 00:00:00.000	2013-09-30 00:00:00.000	5

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | AdventureWorks2017 | 00:00:00 | 30,334 rows

Sample JSON Output with total number of rows returned (30344)



JSToolNpp JSON Viewer

Refresh | Search

[30286]: [Object]
[30287]: [Object]
[30288]: [Object]
[30289]: [Object]
[30290]: [Object]
[30291]: [Object]
[30292]: [Object]
[30293]: [Object]
[30294]: [Object]
[30295]: [Object]
[30296]: [Object]
[30297]: [Object]
[30298]: [Object]
[30299]: [Object]
[30300]: [Object]
[30301]: [Object]
[30302]: [Object]
[30303]: [Object]
[30304]: [Object]
[30305]: [Object]
[30306]: [Object]
[30307]: [Object]
[30308]: [Object]
[30309]: [Object]
[30310]: [Object]
[30311]: [Object]
[30312]: [Object]
[30313]: [Object]
[30314]: [Object]
[30315]: [Object]
[30316]: [Object]
[30317]: [Object]
[30318]: [Object]
[30319]: [Object]
[30320]: [Object]
[30321]: [Object]
[30322]: [Object]
[30323]: [Object]
[30324]: [Object]
[30325]: [Object]
[30326]: [Object]
[30327]: [Object]
[30328]: [Object]
[30329]: [Object]
[30330]: [Object]
[30331]: [Object]
[30332]: [Object]
[30333]: [Object]
CreditCardID: 19237
OrderDate: "2014-06-20T
DueDate: "2014-07-02T0
ShipDate: "2014-06-27T0
Status: 5

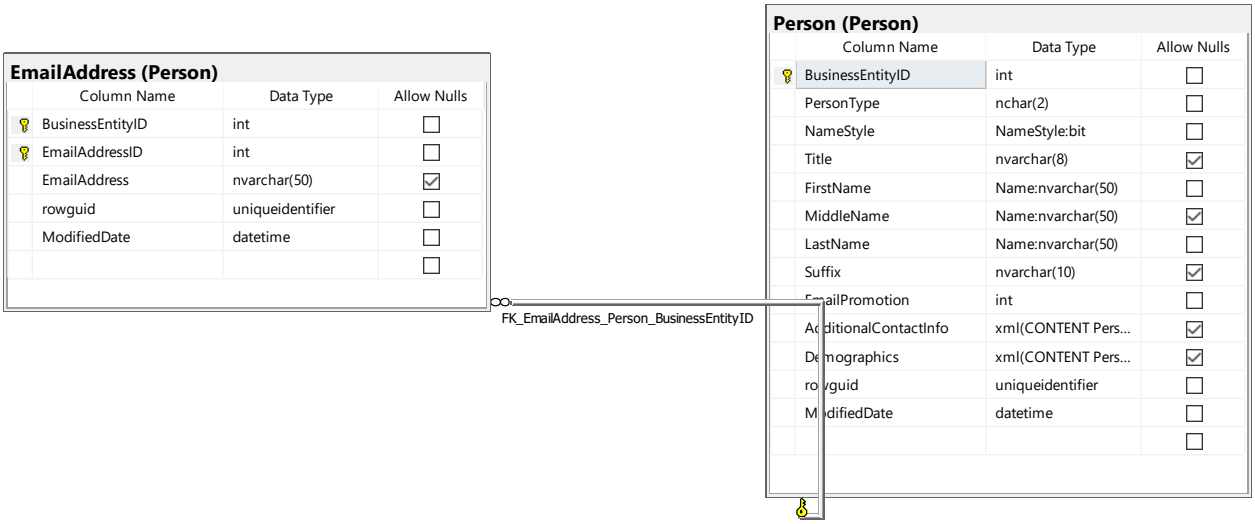
new 1


```
        "CreditCardID": 19232,
        "OrderDate": "2013-07-18T00:00:00",
        "DueDate": "2013-07-30T00:00:00",
        "ShipDate": "2013-07-25T00:00:00",
        "Status": 5
    }, {
        "CreditCardID": 19233,
        "OrderDate": "2013-11-01T00:00:00",
        "DueDate": "2013-11-13T00:00:00",
        "ShipDate": "2013-11-08T00:00:00",
        "Status": 5
    }, {
        "CreditCardID": 19234,
        "OrderDate": "2013-09-29T00:00:00",
        "DueDate": "2013-10-11T00:00:00",
        "ShipDate": "2013-10-06T00:00:00",
        "Status": 5
    }, {
        "CreditCardID": 19234,
        "OrderDate": "2014-03-21T00:00:00",
        "DueDate": "2014-04-02T00:00:00",
        "ShipDate": "2014-03-28T00:00:00",
        "Status": 5
    }, {
        "CreditCardID": 19234,
        "OrderDate": "2014-04-19T00:00:00",
        "DueDate": "2014-05-01T00:00:00",
        "ShipDate": "2014-04-26T00:00:00",
        "Status": 5
    }, {
        "CreditCardID": 19235,
        "OrderDate": "2013-04-30T00:00:00",
        "DueDate": "2013-05-12T00:00:00",
        "ShipDate": "2013-05-07T00:00:00",
        "Status": 5
    }, {
        "CreditCardID": 19235,
        "OrderDate": "2013-11-30T00:00:00",
        "DueDate": "2013-12-12T00:00:00",
        "ShipDate": "2013-12-07T00:00:00",
        "Status": 5
    }, {
        "CreditCardID": 19236,
        "OrderDate": "2014-01-20T00:00:00",
        "DueDate": "2014-02-01T00:00:00",
        "ShipDate": "2014-01-27T00:00:00",
        "Status": 5
    }, {
        "CreditCardID": 19237,
        "OrderDate": "2014-06-20T00:00:00",
        "DueDate": "2014-07-02T00:00:00",
        "ShipDate": "2014-06-27T00:00:00",
        "Status": 5
    }
]
```

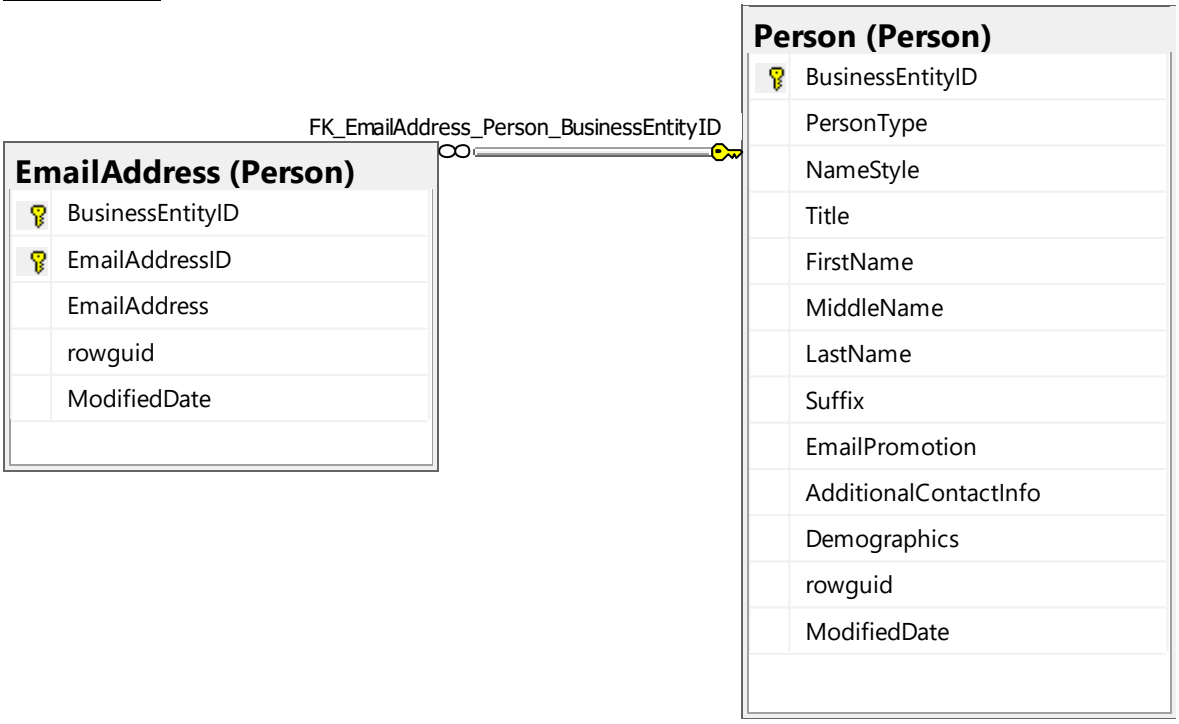
Simple Query 4:

```
--Find all respective email addresses for each business using BusinessID, using
AdventureWorks2017.
```

Standard View:



Key View:



Columns from tables

Table Name	Column Names
Person	BusinessEntityID
EmailAddress	EmailAddress

Order By

Table Name	Column Name	Sort Order
Person	BusinessEntityID	ASC

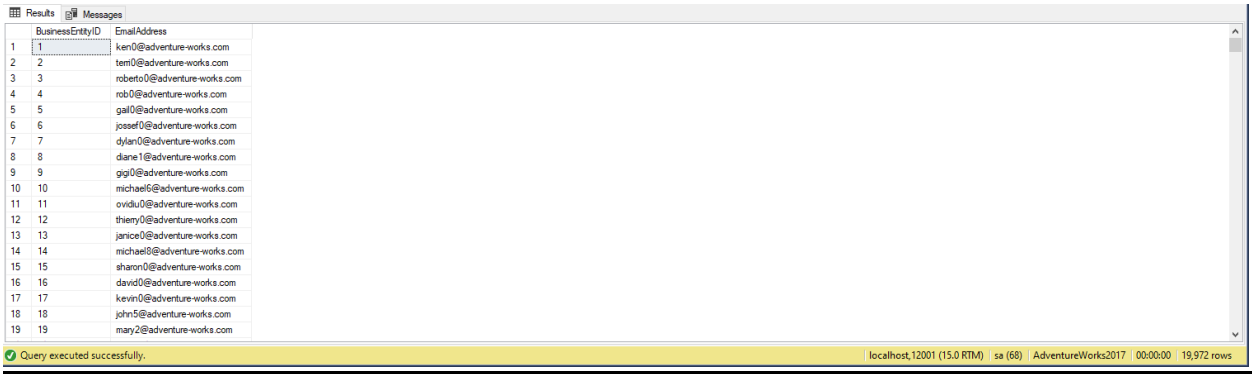
Without JSON:

```
SELECT p.BusinessEntityID,
       e.EmailAddress
FROM Person.Person AS p
    INNER JOIN Person.EmailAddress AS e
        ON e.BusinessEntityID = p.BusinessEntityID
ORDER BY p.BusinessEntityID;
--FOR JSON PATH, ROOT ('BusinessEmail'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT p.BusinessEntityID,
       e.EmailAddress
FROM Person.Person AS p
    INNER JOIN Person.EmailAddress AS e
        ON e.BusinessEntityID = p.BusinessEntityID
ORDER BY p.BusinessEntityID
FOR JSON PATH, ROOT ('BusinessEmail'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (19972)

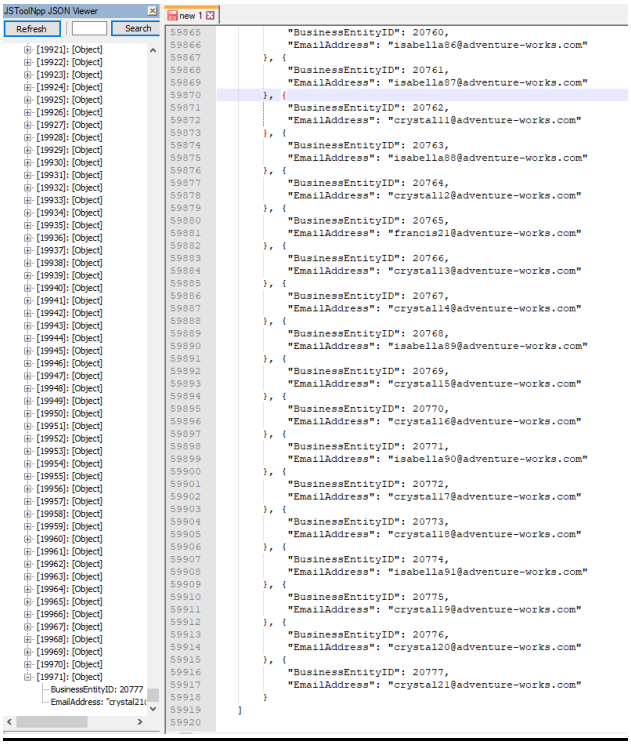


Results | Messages

	BusinessEntityID	EmailAddress
1	1	ken0@adventure-works.com
2	2	terri0@adventure-works.com
3	3	roberto0@adventure-works.com
4	4	rob0@adventure-works.com
5	5	gail0@adventure-works.com
6	6	jossef0@adventure-works.com
7	7	dylan0@adventure-works.com
8	8	diane1@adventure-works.com
9	9	gigi0@adventure-works.com
10	10	michael6@adventure-works.com
11	11	ovidiu0@adventure-works.com
12	12	thierry0@adventure-works.com
13	13	janice0@adventure-works.com
14	14	michael8@adventure-works.com
15	15	sharon0@adventure-works.com
16	16	david0@adventure-works.com
17	17	kevin0@adventure-works.com
18	18	john5@adventure-works.com
19	19	mary2@adventure-works.com

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | AdventureWorks2017 | 00:00:00 | 19,972 rows

Sample JSON Output with total number of rows returned (19972)



JSToolNpp JSON Viewer

Refresh | Search

[19921]: [Object]
[19922]: [Object]
[19923]: [Object]
[19924]: [Object]
[19925]: [Object]
[19926]: [Object]
[19927]: [Object]
[19928]: [Object]
[19929]: [Object]
[19930]: [Object]
[19931]: [Object]
[19932]: [Object]
[19933]: [Object]
[19934]: [Object]
[19935]: [Object]
[19936]: [Object]
[19937]: [Object]
[19938]: [Object]
[19939]: [Object]
[19940]: [Object]
[19941]: [Object]
[19942]: [Object]
[19943]: [Object]
[19944]: [Object]
[19945]: [Object]
[19946]: [Object]
[19947]: [Object]
[19948]: [Object]
[19949]: [Object]
[19950]: [Object]
[19951]: [Object]
[19952]: [Object]
[19953]: [Object]
[19954]: [Object]
[19955]: [Object]
[19956]: [Object]
[19957]: [Object]
[19958]: [Object]
[19959]: [Object]
[19960]: [Object]
[19961]: [Object]
[19962]: [Object]
[19963]: [Object]
[19964]: [Object]
[19965]: [Object]
[19966]: [Object]
[19967]: [Object]
[19968]: [Object]
[19969]: [Object]
[19970]: [Object]
[19971]: [Object]
BusinessEntityID: 20777
EmailAddress: "crystal21(

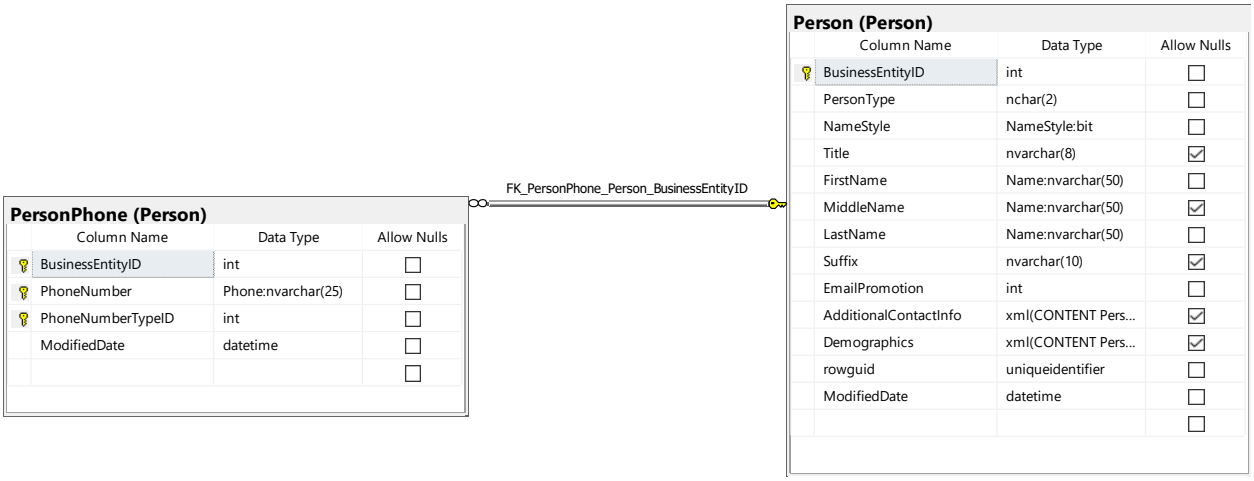
new 1


```
        "BusinessEntityID": 20760,
        "EmailAddress": "isabella86@adventure-works.com"
    }, {
        "BusinessEntityID": 20761,
        "EmailAddress": "isabella87@adventure-works.com"
    }, {
        "BusinessEntityID": 20762,
        "EmailAddress": "crystal11@adventure-works.com"
    }, {
        "BusinessEntityID": 20763,
        "EmailAddress": "isabella88@adventure-works.com"
    }, {
        "BusinessEntityID": 20764,
        "EmailAddress": "crystal12@adventure-works.com"
    }, {
        "BusinessEntityID": 20765,
        "EmailAddress": "francis21@adventure-works.com"
    }, {
        "BusinessEntityID": 20766,
        "EmailAddress": "crystal13@adventure-works.com"
    }, {
        "BusinessEntityID": 20767,
        "EmailAddress": "crystal14@adventure-works.com"
    }, {
        "BusinessEntityID": 20768,
        "EmailAddress": "isabella89@adventure-works.com"
    }, {
        "BusinessEntityID": 20769,
        "EmailAddress": "crystal15@adventure-works.com"
    }, {
        "BusinessEntityID": 20770,
        "EmailAddress": "crystal16@adventure-works.com"
    }, {
        "BusinessEntityID": 20771,
        "EmailAddress": "isabella90@adventure-works.com"
    }, {
        "BusinessEntityID": 20772,
        "EmailAddress": "crystal17@adventure-works.com"
    }, {
        "BusinessEntityID": 20773,
        "EmailAddress": "crystal18@adventure-works.com"
    }, {
        "BusinessEntityID": 20774,
        "EmailAddress": "isabella91@adventure-works.com"
    }, {
        "BusinessEntityID": 20775,
        "EmailAddress": "crystal19@adventure-works.com"
    }, {
        "BusinessEntityID": 20776,
        "EmailAddress": "crystal20@adventure-works.com"
    }, {
        "BusinessEntityID": 20777,
        "EmailAddress": "crystal21@adventure-works.com"
    }
]
```

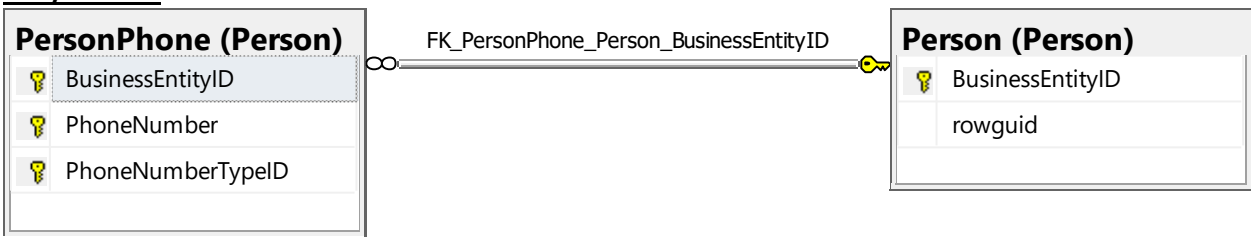
Simple Query 5:

```
--Find all respective phone numbers for each business using BusinessID, using
AdventureWorks2017.
```

Standard View:



Key View:



Columns from tables

Table Name	Column Names
Person	BusinessEntityID
PersonPhone	PhoneNumber

Order By

Table Name	Column Name	Sort Order
Person	BusinessEntityID	ASC

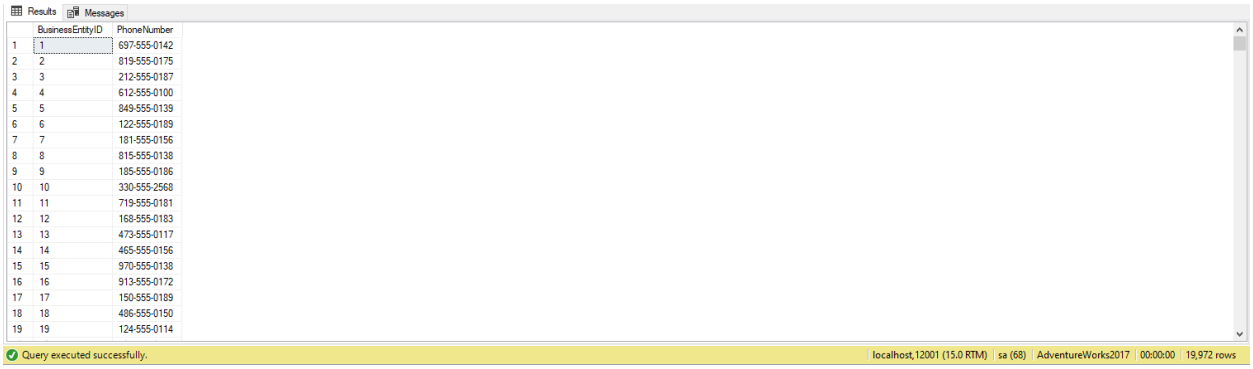
Without JSON:

```
SELECT p.BusinessEntityID,
       pp.PhoneNumber
FROM Person.Person AS p
    INNER JOIN Person.PersonPhone AS pp
        ON p.BusinessEntityID = pp.BusinessEntityID
ORDER BY p.BusinessEntityID;
--FOR JSON PATH, ROOT ('BusinessPhone'), INCLUDE_NULL_VALUES;
```

Without JSON:

```
SELECT p.BusinessEntityID,
       pp.PhoneNumber
FROM Person.Person AS p
    INNER JOIN Person.PersonPhone AS pp
        ON p.BusinessEntityID = pp.BusinessEntityID
ORDER BY p.BusinessEntityID
FOR JSON PATH, ROOT ('BusinessPhone'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (19972)

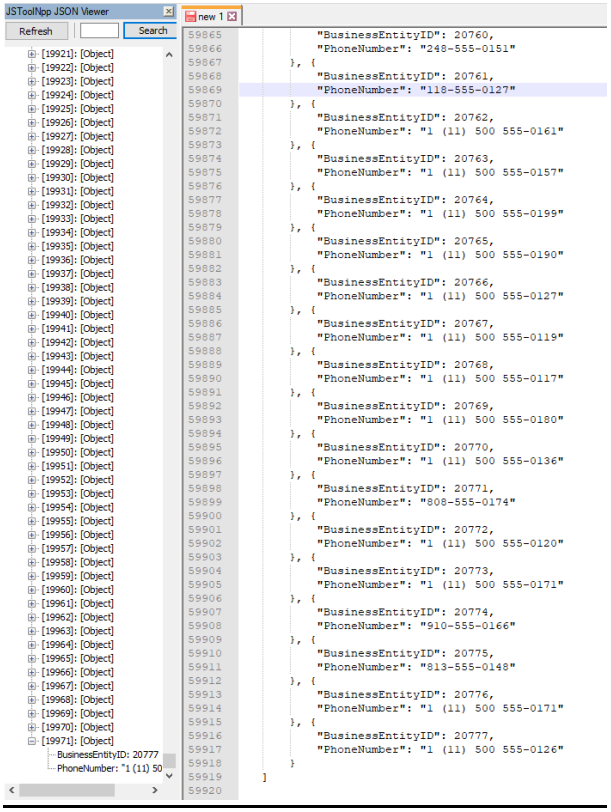


Results | Messages

	BusinessEntityID	PhoneNumber
1	1	697-555-0142
2	2	819-555-0175
3	3	212-555-0187
4	4	612-555-0100
5	5	849-555-0139
6	6	122-555-0189
7	7	181-555-0156
8	8	815-555-0138
9	9	185-555-0186
10	10	330-555-2568
11	11	719-555-0181
12	12	168-555-0183
13	13	473-555-0117
14	14	465-555-0156
15	15	970-555-0138
16	16	913-555-0172
17	17	150-555-0189
18	18	486-555-0150
19	19	124-555-0114

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | AdventureWorks2017 | 00:00:00 | 19,972 rows

Sample JSON Output with total number of rows returned (19972)



JSToolNpp JSON Viewer

Refresh | Search

[19921]: [Object]
[19922]: [Object]
[19923]: [Object]
[19924]: [Object]
[19925]: [Object]
[19926]: [Object]
[19927]: [Object]
[19928]: [Object]
[19929]: [Object]
[19930]: [Object]
[19931]: [Object]
[19932]: [Object]
[19933]: [Object]
[19934]: [Object]
[19935]: [Object]
[19936]: [Object]
[19937]: [Object]
[19938]: [Object]
[19939]: [Object]
[19940]: [Object]
[19941]: [Object]
[19942]: [Object]
[19943]: [Object]
[19944]: [Object]
[19945]: [Object]
[19946]: [Object]
[19947]: [Object]
[19948]: [Object]
[19949]: [Object]
[19950]: [Object]
[19951]: [Object]
[19952]: [Object]
[19953]: [Object]
[19954]: [Object]
[19955]: [Object]
[19956]: [Object]
[19957]: [Object]
[19958]: [Object]
[19959]: [Object]
[19960]: [Object]
[19961]: [Object]
[19962]: [Object]
[19963]: [Object]
[19964]: [Object]
[19965]: [Object]
[19966]: [Object]
[19967]: [Object]
[19968]: [Object]
[19969]: [Object]
[19970]: [Object]
[19971]: [Object]
BusinessEntityID: 20777
PhoneNumber: "1 (11) 50

new 1


```
        {
            "BusinessEntityID": 20760,
            "PhoneNumber": "248-555-0151"
        }, {
            "BusinessEntityID": 20761,
            "PhoneNumber": "118-555-0127"
        }, {
            "BusinessEntityID": 20762,
            "PhoneNumber": "1 (11) 500 555-0161"
        }, {
            "BusinessEntityID": 20763,
            "PhoneNumber": "1 (11) 500 555-0157"
        }, {
            "BusinessEntityID": 20764,
            "PhoneNumber": "1 (11) 500 555-0199"
        }, {
            "BusinessEntityID": 20765,
            "PhoneNumber": "1 (11) 500 555-0190"
        }, {
            "BusinessEntityID": 20766,
            "PhoneNumber": "1 (11) 500 555-0127"
        }, {
            "BusinessEntityID": 20767,
            "PhoneNumber": "1 (11) 500 555-0119"
        }, {
            "BusinessEntityID": 20768,
            "PhoneNumber": "1 (11) 500 555-0117"
        }, {
            "BusinessEntityID": 20769,
            "PhoneNumber": "1 (11) 500 555-0180"
        }, {
            "BusinessEntityID": 20770,
            "PhoneNumber": "1 (11) 500 555-0136"
        }, {
            "BusinessEntityID": 20771,
            "PhoneNumber": "808-555-0174"
        }, {
            "BusinessEntityID": 20772,
            "PhoneNumber": "1 (11) 500 555-0120"
        }, {
            "BusinessEntityID": 20773,
            "PhoneNumber": "1 (11) 500 555-0171"
        }, {
            "BusinessEntityID": 20774,
            "PhoneNumber": "910-555-0166"
        }, {
            "BusinessEntityID": 20775,
            "PhoneNumber": "813-555-0148"
        }, {
            "BusinessEntityID": 20776,
            "PhoneNumber": "1 (11) 500 555-0171"
        }, {
            "BusinessEntityID": 20777,
            "PhoneNumber": "1 (11) 500 555-0126"
        }
    ]
```

Medium Queries

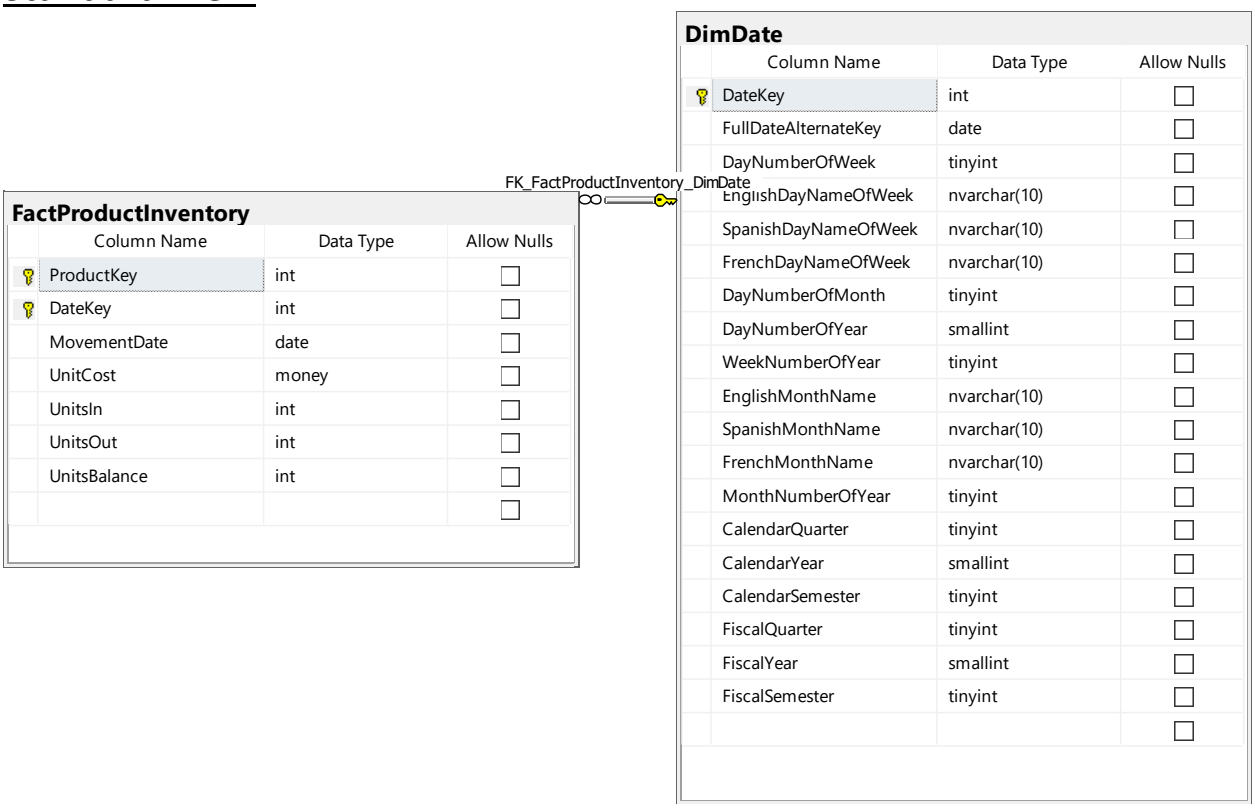
- All medium queries use the AdventureWorksDW2017 database
- All medium queries use the DimDate subsystem

Medium Query Propositions:

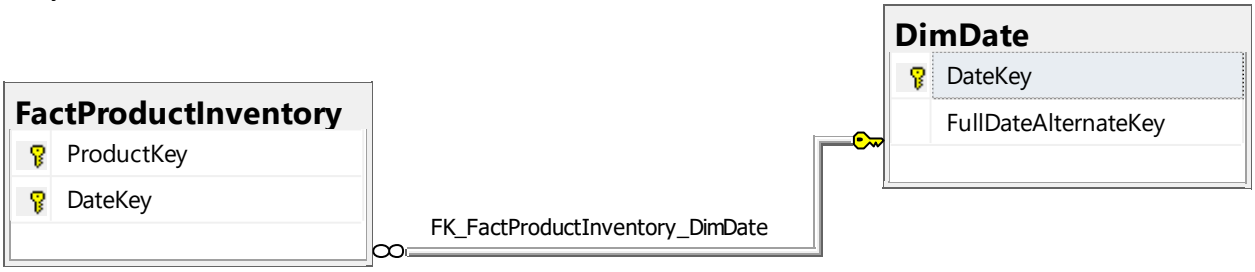
Medium Query 6:

```
--Find the sum of unitsin in FactProductInventory in each calendar year using the datekey
from FactProductInventory and DimDate, using AdventureWorksDW2017.
```

Standard View:



Key view:



Columns from tables

Table Name	Column Names
DimDate	CalendarYear
FactProductInventory	UnitsIn

Order By

Table Name	Column Name	Sort Order
DimDate	CalendarYear	ASC

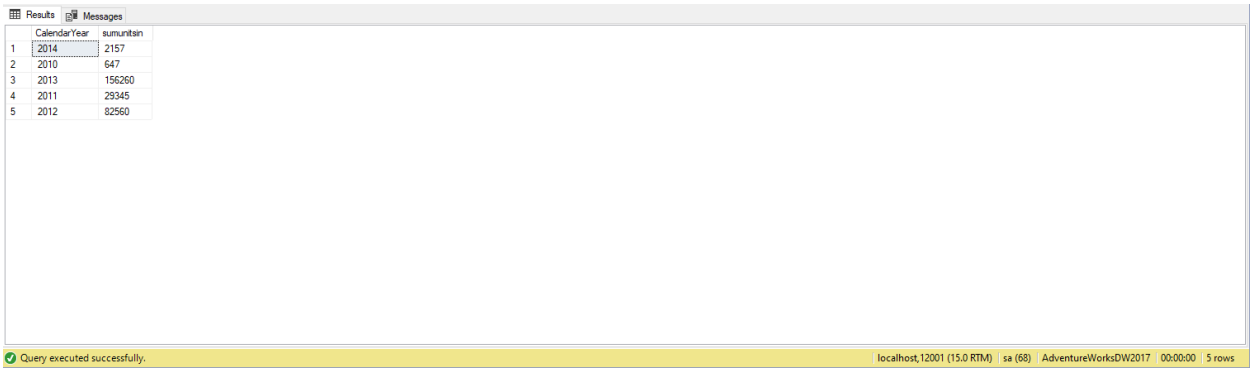
Without JSON:

```
SELECT dd.CalendarYear,
       SUM(fpi.UnitsIn) AS sumunitsin
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactProductInventory AS fpi
        ON dd.DateKey = fpi.DateKey
GROUP BY dd.CalendarYear;
--FOR JSON PATH, ROOT ('SumUnits'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT dd.CalendarYear,
       SUM(fpi.UnitsIn) AS sumunitsin
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactProductInventory AS fpi
        ON dd.DateKey = fpi.DateKey
GROUP BY dd.CalendarYear
FOR JSON PATH, ROOT ('SumUnits'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (5)

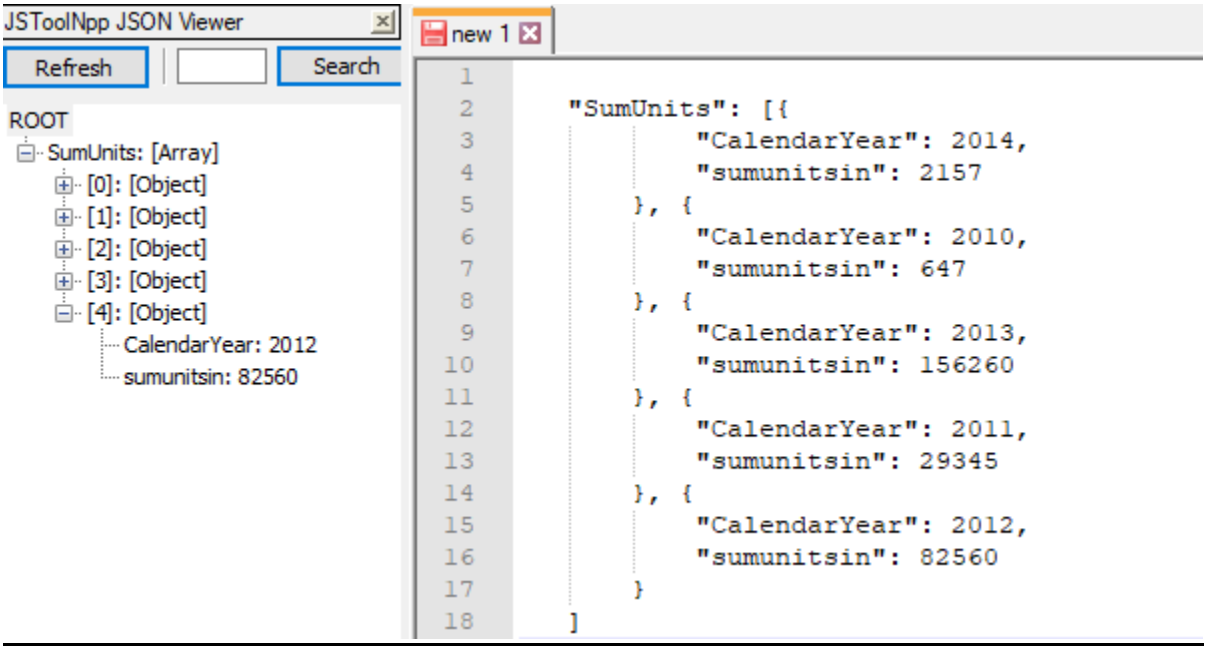


Results | Messages

	CalendarYear	sumunitsin
1	2014	2157
2	2010	647
3	2013	156260
4	2011	29345
5	2012	82560

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | AdventureWorksDW2017 | 00:00:00 | 5 rows

Sample JSON Output with total number of rows returned (5)



JSToolNpp JSON Viewer

Refresh | Search

ROOT

- SumUnits: [Array]
  - [0]: [Object]
  - [1]: [Object]
  - [2]: [Object]
  - [3]: [Object]
  - [4]: [Object]
    - CalendarYear: 2012
    - sumunitsin: 82560

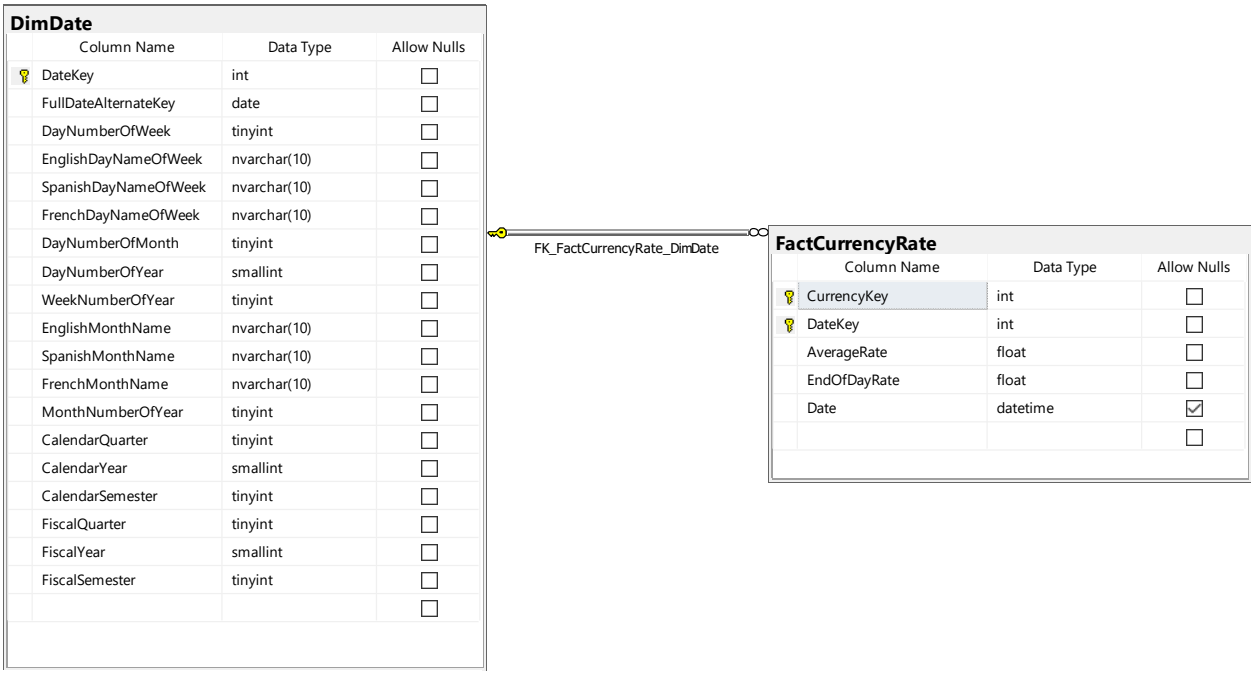
new 1

```
1
2     "SumUnits": [{
3             "CalendarYear": 2014,
4             "sumunitsin": 2157
5         }, {
6             "CalendarYear": 2010,
7             "sumunitsin": 647
8         }, {
9             "CalendarYear": 2013,
10            "sumunitsin": 156260
11        }, {
12            "CalendarYear": 2011,
13            "sumunitsin": 29345
14        }, {
15            "CalendarYear": 2012,
16            "sumunitsin": 82560
17        }
18    ]
```

Medium Query 7:

```
--Using the datekey from FactCurrencyRate, find the spanish version of the date from
DimDate, using AdventureWorks2017.
```

Standard view:



Key view:

Columns from tables

Table Name	Column Names
DimDate	SpanishMonthName SpanishDayNameofWeek CalendarYear
FactCurrencyRate	DateKey

Order By

Table Name	Column Name	Sort Order
FactCurrencyRate	DateKey	ASC

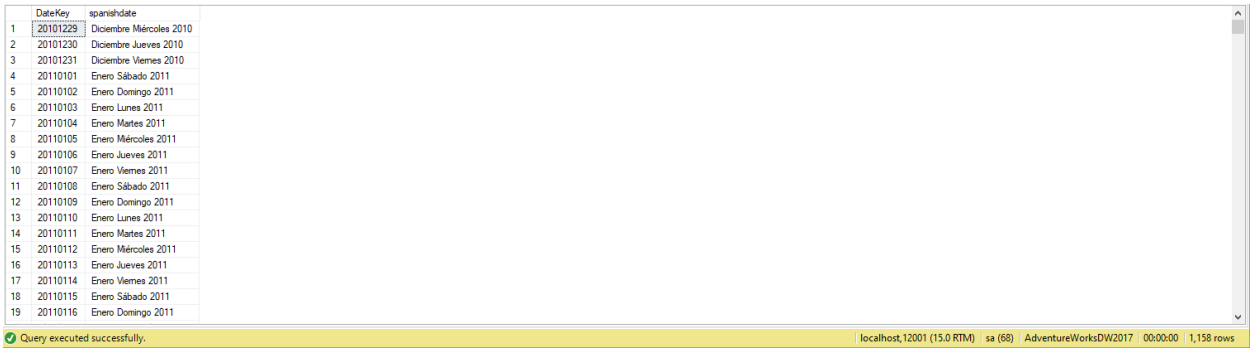
Without JSON:

```
SELECT fcr.DateKey,
       CONCAT(dd.SpanishMonthName, ' ', dd.SpanishDayNameOfWeek, ' ', dd.CalendarYear) AS
spanishdate
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactCurrencyRate AS fcr
        ON dd.DateKey = fcr.DateKey
GROUP BY fcr.DateKey,
         dd.SpanishMonthName,
         dd.SpanishDayNameOfWeek,
         dd.CalendarYear
ORDER BY fcr.DateKey;
--FOR JSON PATH, ROOT ('SpanishDate'), INCLUDE_NULL_VALUES;
```

With JSON:

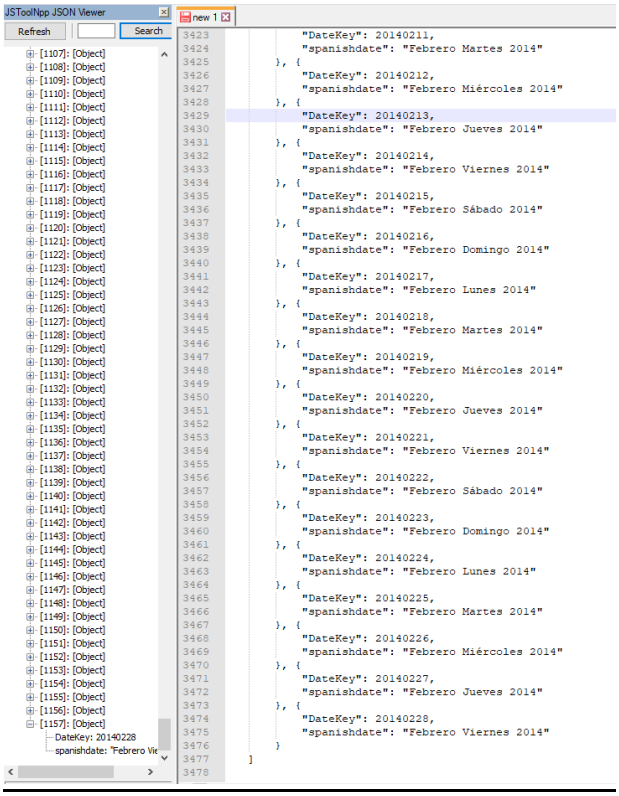
```
SELECT fcr.DateKey,
       CONCAT(dd.SpanishMonthName, ' ', dd.SpanishDayNameOfWeek, ' ', dd.CalendarYear) AS
spanishdate
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactCurrencyRate AS fcr
        ON dd.DateKey = fcr.DateKey
GROUP BY fcr.DateKey,
         dd.SpanishMonthName,
         dd.SpanishDayNameOfWeek,
         dd.CalendarYear
ORDER BY fcr.DateKey
FOR JSON PATH, ROOT ('SpanishDate'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (1158)



	DateKey	spanishdate
1	20101229	Diciembre Miércoles 2010
2	20101230	Diciembre Jueves 2010
3	20101231	Diciembre Viernes 2010
4	20110101	Enero Sábado 2011
5	20110102	Enero Domingo 2011
6	20110103	Enero Lunes 2011
7	20110104	Enero Martes 2011
8	20110105	Enero Miércoles 2011
9	20110106	Enero Jueves 2011
10	20110107	Enero Viernes 2011
11	20110108	Enero Sábado 2011
12	20110109	Enero Domingo 2011
13	20110110	Enero Lunes 2011
14	20110111	Enero Martes 2011
15	20110112	Enero Miércoles 2011
16	20110113	Enero Jueves 2011
17	20110114	Enero Viernes 2011
18	20110115	Enero Sábado 2011
19	20110116	Enero Domingo 2011

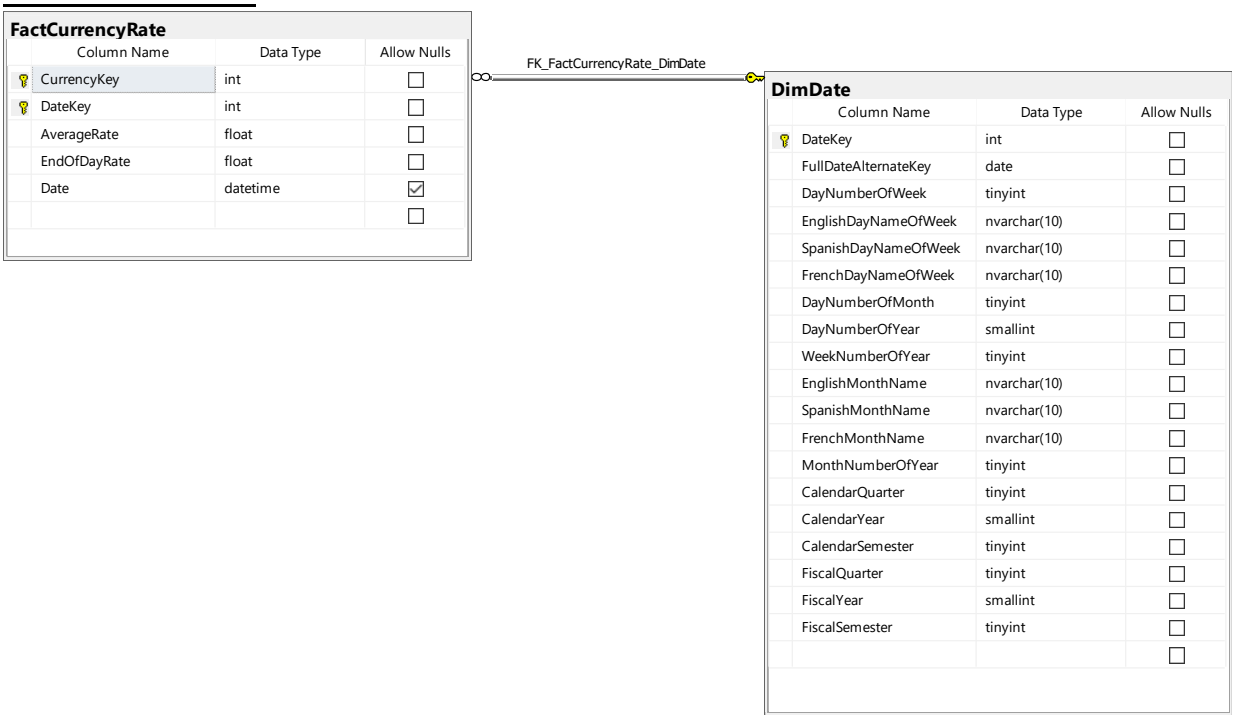
Sample JSON Output with total number of rows returned (1158)



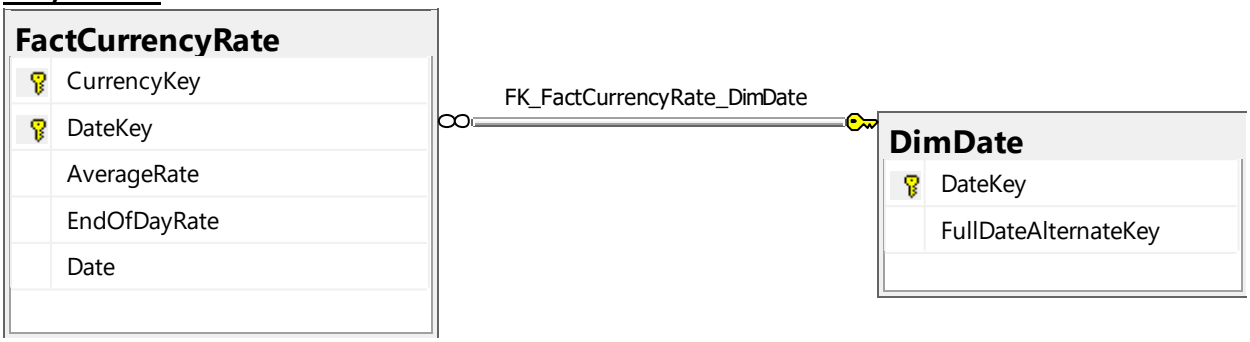
Medium Query 8:

```
--Find the minimum average rate of each fiscal year from FactCurrencyRate using DateKey
from DimDate, using AdventureWorksDW2017.
```

Standard View:



Key View:



Columns from tables

Table Name	Column Names
DimDate	FiscalYear
FactCurrencyRate	AverageRate

Order By

Table Name	Column Name	Sort Order
DimDate	FiscalYear	ASC

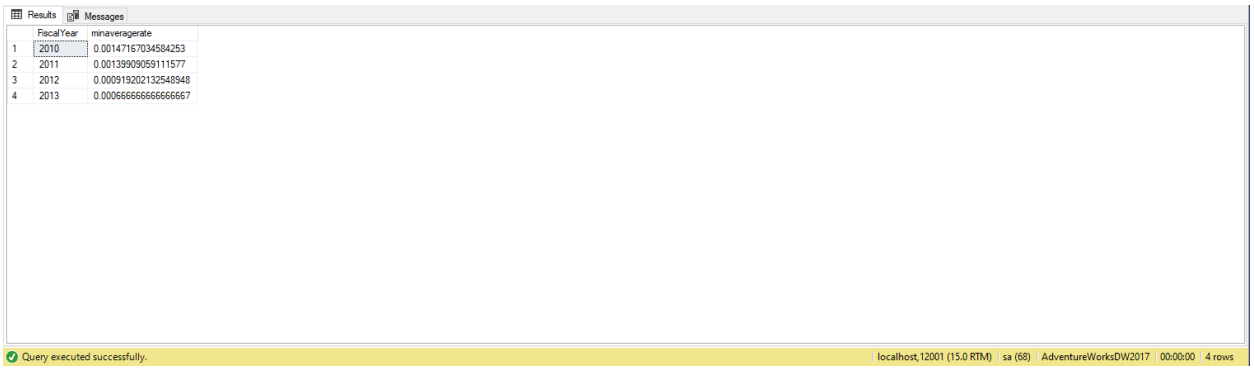
Without JSON:

```
SELECT dd.FiscalYear,
       MIN(fcr.AverageRate) AS minaveragerate
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactCurrencyRate AS fcr
        ON dd.DateKey = fcr.DateKey
GROUP BY dd.FiscalYear
ORDER BY dd.FiscalYear;
--FOR JSON PATH, ROOT ('FiscalMinAvg'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT dd.FiscalYear,
       MIN(fcr.AverageRate) AS minaveragerate
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactCurrencyRate AS fcr
        ON dd.DateKey = fcr.DateKey
GROUP BY dd.FiscalYear
ORDER BY dd.FiscalYear
FOR JSON PATH, ROOT ('FiscalMinAvg'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (4)

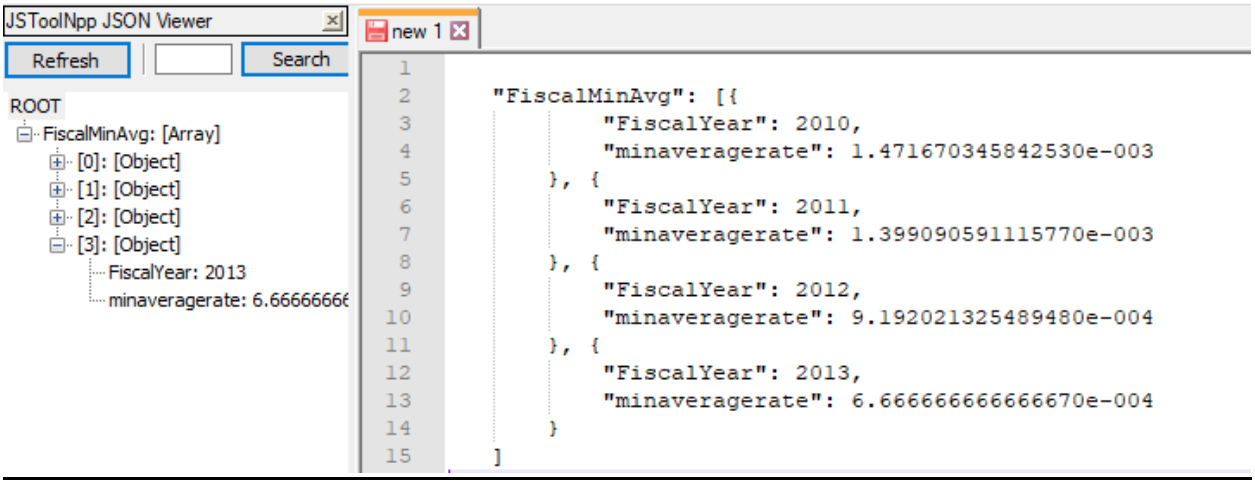


Results | Messages

	FiscalYear	minaveragerate
1	2010	0.00147167034584253
2	2011	0.00139909059111577
3	2012	0.000919202132548948
4	2013	0.000666666666666667

Query executed successfully. localhost,12001 (15.0 RTM) | sa (68) | AdventureWorksDW2017 | 00:00:00 | 4 rows

Sample JSON Output with total number of rows returned (4)



JSToolNpp JSON Viewer

Refresh | Search

ROOT

- FiscalMinAvg: [Array]
  - [0]: [Object]
  - [1]: [Object]
  - [2]: [Object]
  - [3]: [Object]
    - FiscalYear: 2013
    - minaveragerate: 6.66666666

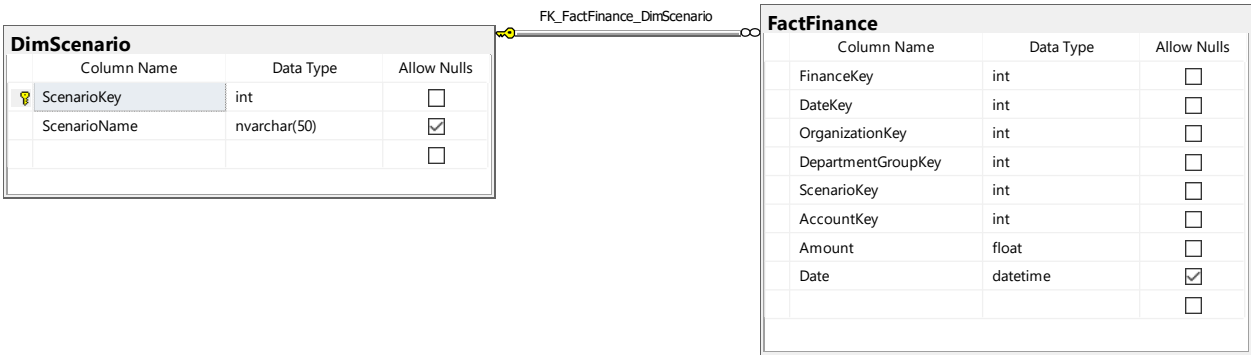
new 1

```
{
    "FiscalMinAvg": [{
            "FiscalYear": 2010,
            "minaveragerate": 1.471670345842530e-003
        }, {
            "FiscalYear": 2011,
            "minaveragerate": 1.399090591115770e-003
        }, {
            "FiscalYear": 2012,
            "minaveragerate": 9.192021325489480e-004
        }, {
            "FiscalYear": 2013,
            "minaveragerate": 6.666666666666670e-004
        }
    ]
```

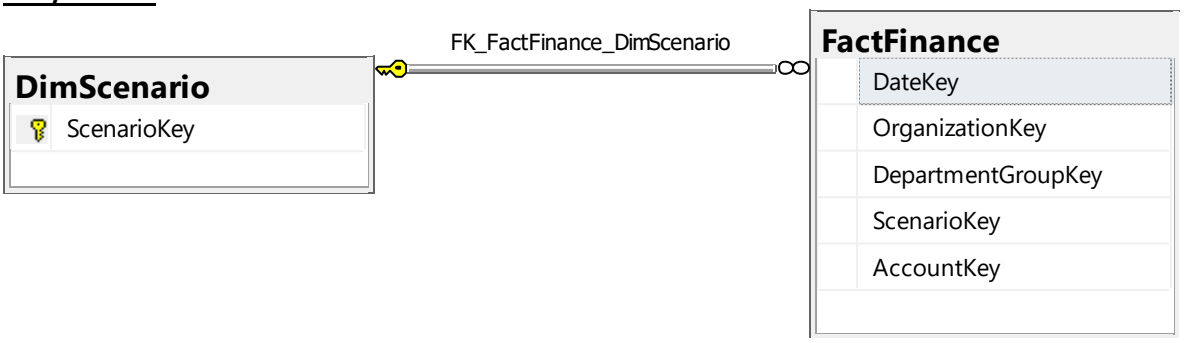
Medium Query 9:

```
--Find the number of scenario keys and display scenario names for each key, using
AdventureWorksDW2017.
```

Standard view:



Key view:



Columns from tables

Table Name	Column Names
FactFinance	ScenarioKey
DimScenario	ScenarioName

Order By

Table Name	Column Name	Sort Order
FactFinance	ScenarioKey	ASC

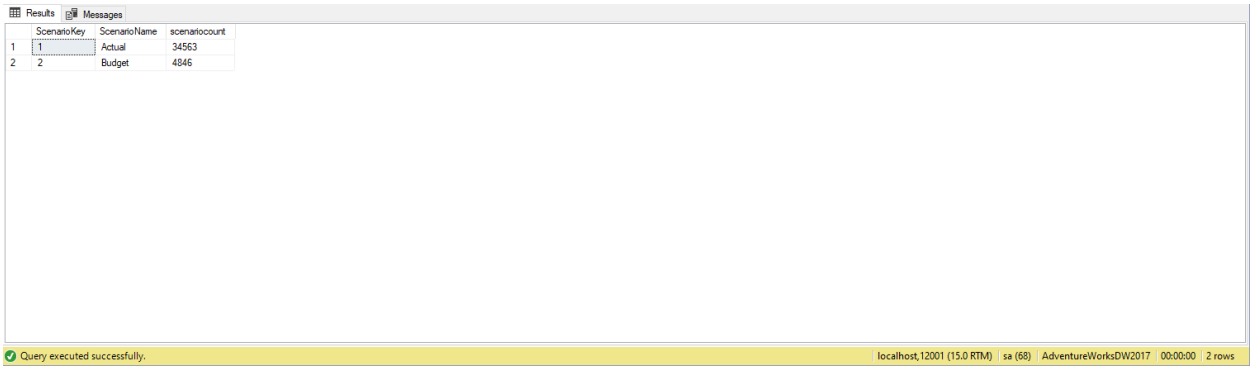
Without JSON:

```
SELECT ff.ScenarioKey,
       ds.ScenarioName,
       COUNT(ff.ScenarioKey)
FROM dbo.FactFinance AS ff
    INNER JOIN dbo.DimScenario AS ds
        ON ff.ScenarioKey = ds.ScenarioKey
GROUP BY ff.ScenarioKey,
         ds.ScenarioName
ORDER BY ff.ScenarioKey;
--FOR JSON PATH, ROOT ('ScenarioKeyCountAndName'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT dd.FiscalYear,
       MIN(fcr.AverageRate) AS minaveragerate
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactCurrencyRate AS fcr
        ON dd.DateKey = fcr.DateKey
GROUP BY dd.FiscalYear
ORDER BY dd.FiscalYear
FOR JSON PATH, ROOT ('FiscalMinAvg'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (2)

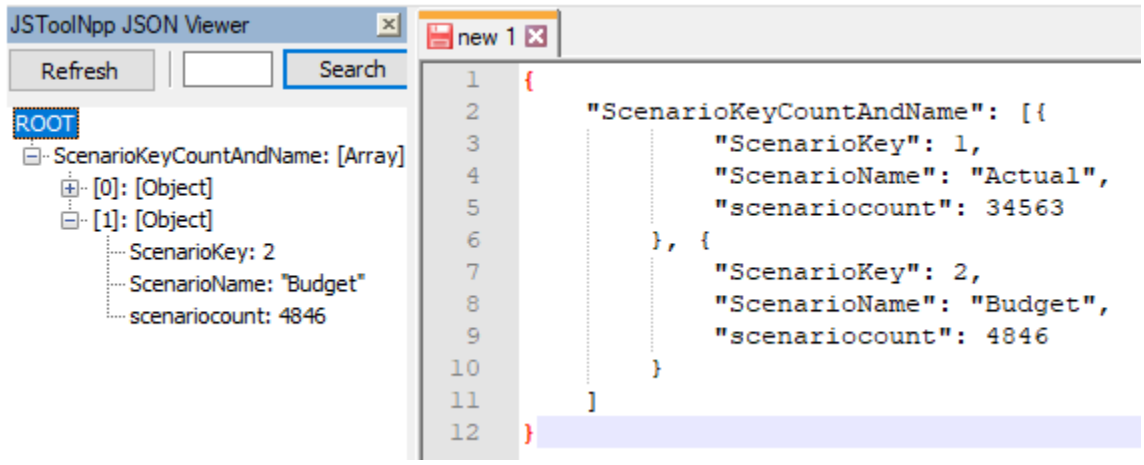


Results | Messages

	ScenarioKey	ScenarioName	scenariocount
1	1	Actual	34563
2	2	Budget	4846

Query executed successfully. localhost,12001 (15.0 RTM) | sa (68) | AdventureWorksDW2017 | 00:00:00 | 2 rows

Sample JSON Output with total number of rows returned (2)



JSToolNpp JSON Viewer

Refresh | Search

ROOT

- ScenarioKeyCountAndName: [Array]
  - [0]: [Object]
  - [1]: [Object]
    - ScenarioKey: 2
    - ScenarioName: "Budget"
    - scenariocount: 4846

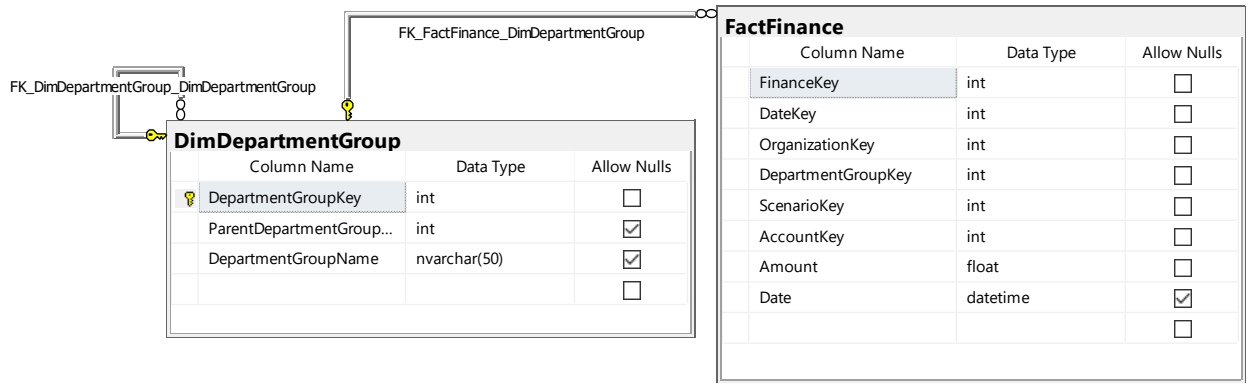
new 1

```
{
    "ScenarioKeyCountAndName": [{
            "ScenarioKey": 1,
            "ScenarioName": "Actual",
            "scenariocount": 34563
        }, {
            "ScenarioKey": 2,
            "ScenarioName": "Budget",
            "scenariocount": 4846
        }
    ]
}
```

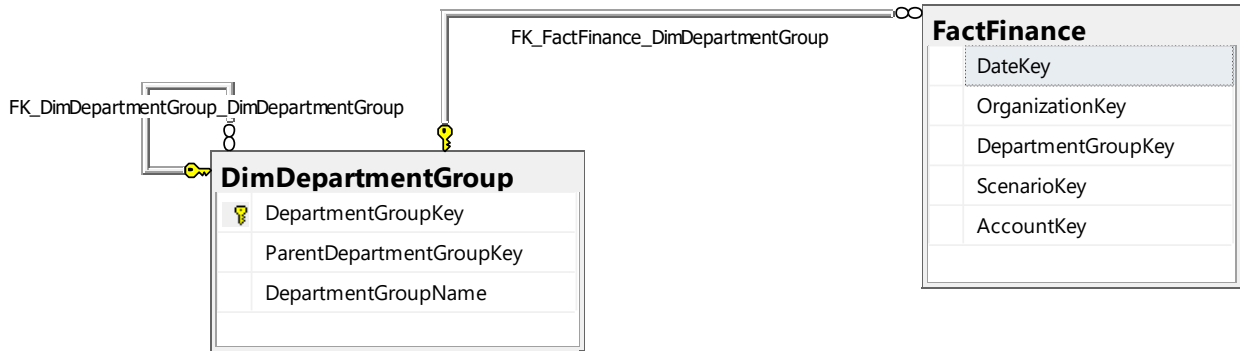
Medium Query 10:

```
--Use the department key in FactFinance to figure out if any ParentDepartmentGroupKey in
DimDepartmentGroup are NULL, then coalesce them to '0', using AdventureWorksDW2017.
```

Standard view:



Key view:



Columns from tables

Table Name	Column Names
FactFinance	DepartmentGroupKey
DimDepartmentGroup	ParentDepartmentGroupKey

Order By

Table Name	Column Name	Sort Order
FactFinance	DepartmentGroupKey	ASC

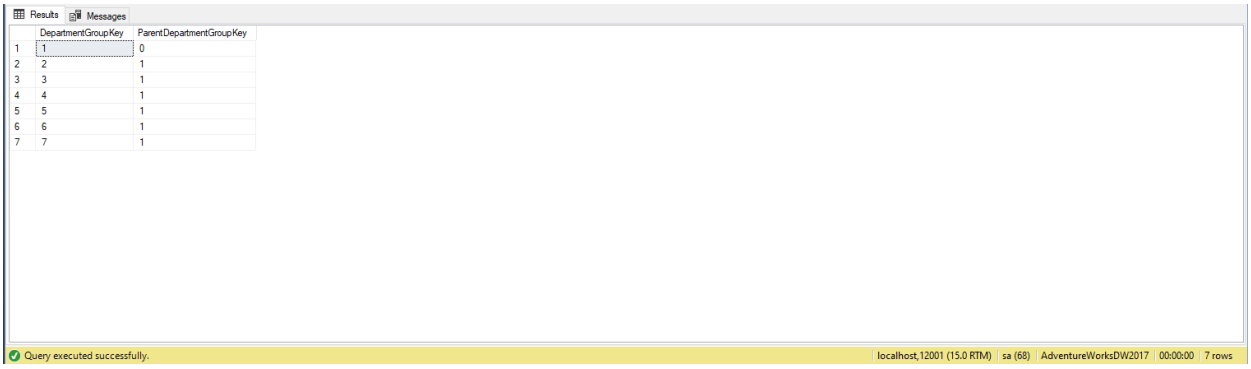
Without JSON:

```
SELECT ff.DepartmentGroupKey,
       COALESCE(ddg.ParentDepartmentGroupKey, 0) AS ParentDepartmentGroupKey
FROM dbo.FactFinance AS ff
    INNER JOIN dbo.DimDepartmentGroup AS ddg
        ON ff.DepartmentGroupKey = ddg.DepartmentGroupKey
GROUP BY ff.DepartmentGroupKey,
         ddg.ParentDepartmentGroupKey
ORDER BY ff.DepartmentGroupKey;
--FOR JSON PATH, ROOT ('CheckKeyForNull'), INCLUDE_NULL_VALUES;
```

With JSON:

```
USE AdventureWorksDW2017;
SELECT ff.DepartmentGroupKey,
       COALESCE(ddg.ParentDepartmentGroupKey, 0) AS ParentDepartmentGroupKey
FROM dbo.FactFinance AS ff
    INNER JOIN dbo.DimDepartmentGroup AS ddg
        ON ff.DepartmentGroupKey = ddg.DepartmentGroupKey
GROUP BY ff.DepartmentGroupKey,
         ddg.ParentDepartmentGroupKey
ORDER BY ff.DepartmentGroupKey
FOR JSON PATH, ROOT ('CheckKeyForNull'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (7)

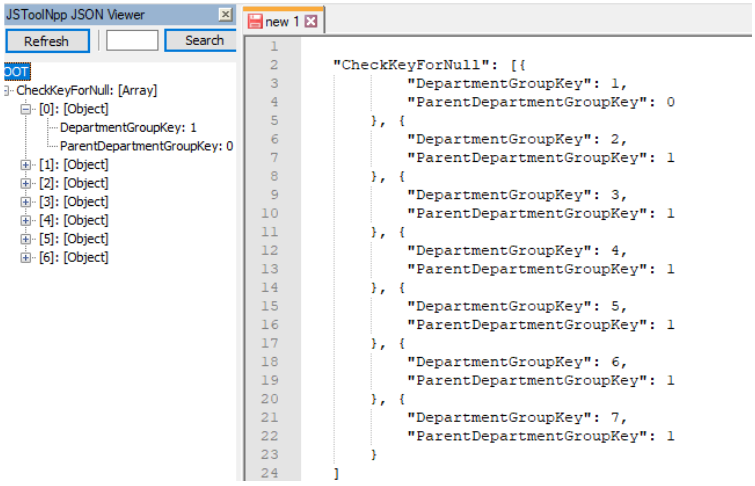


Results Messages

	DepartmentGroupKey	ParentDepartmentGroupKey
1	1	0
2	2	1
3	3	1
4	4	1
5	5	1
6	6	1
7	7	1

Query executed successfully. localhost,12001 (15.0 RTM) | sa (68) | AdventureWorksDW2017 | 00:00:00 | 7 rows

Sample JSON Output with total number of rows returned (7)



JSToolNpp JSON Viewer

Refresh Search

ROOT

- CheckKeyForNull: [Array]
  - [0]: [Object]
    - DepartmentGroupKey: 1
    - ParentDepartmentGroupKey: 0
  - [1]: [Object]
  - [2]: [Object]
  - [3]: [Object]
  - [4]: [Object]
  - [5]: [Object]
  - [6]: [Object]

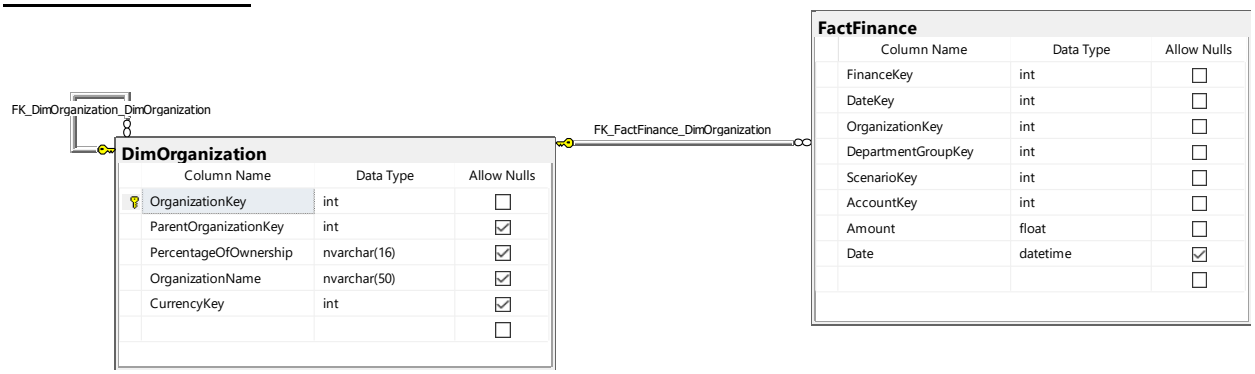
new 1

```
"CheckKeyForNull": [{
        "DepartmentGroupKey": 1,
        "ParentDepartmentGroupKey": 0
    }, {
        "DepartmentGroupKey": 2,
        "ParentDepartmentGroupKey": 1
    }, {
        "DepartmentGroupKey": 3,
        "ParentDepartmentGroupKey": 1
    }, {
        "DepartmentGroupKey": 4,
        "ParentDepartmentGroupKey": 1
    }, {
        "DepartmentGroupKey": 5,
        "ParentDepartmentGroupKey": 1
    }, {
        "DepartmentGroupKey": 6,
        "ParentDepartmentGroupKey": 1
    }, {
        "DepartmentGroupKey": 7,
        "ParentDepartmentGroupKey": 1
    }
]
```

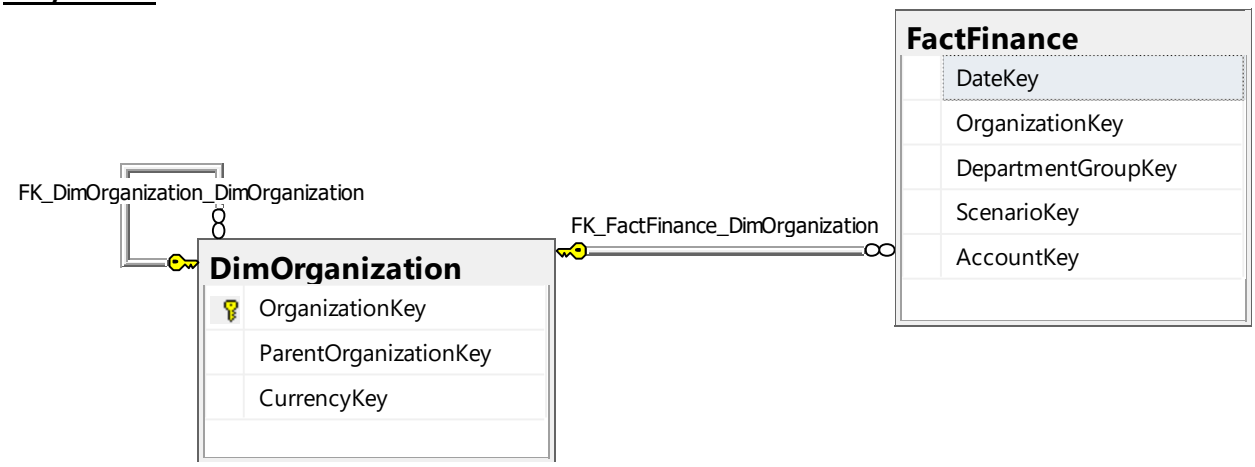
Medium Query 11:

```
--Convert the percentage of ownership out of all organizations with their keys from
DimOrganization using the OrganizationKey from FactFinance to decimal, using
AdventureWorksDW2017.
```

Standard view:



Key view:



Columns from tables

Table Name	Column Names
DimOrganization	PercentageofOwnership
FactFinance	OrganizationKey

Order By

Table Name	Column Name	Sort Order
FactFinance	OrganizationKey	ASC

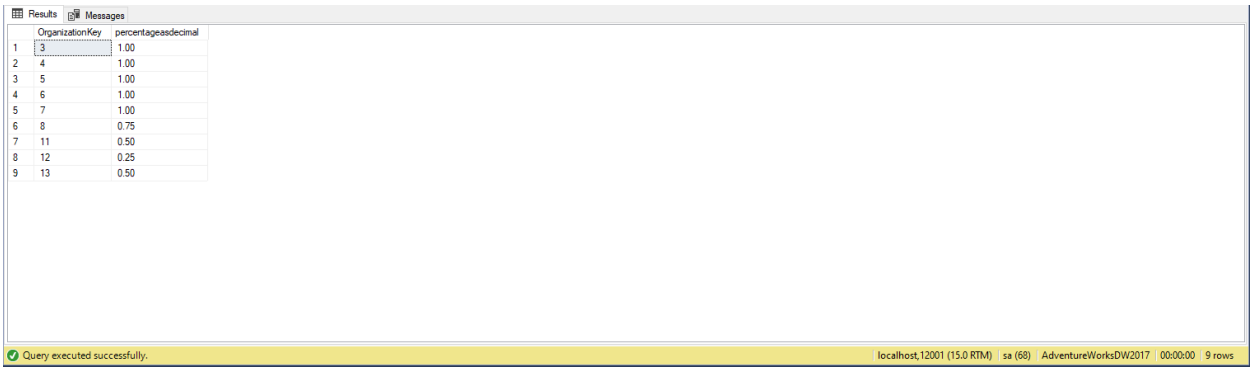
Without JSON:

```
SELECT ff.OrganizationKey,
       (CONVERT(DECIMAL(5, 2), do.PercentageOfOwnership)) AS percentageasdecimal
FROM dbo.DimOrganization AS do
    INNER JOIN dbo.FactFinance AS ff
        ON ff.OrganizationKey = do.OrganizationKey
GROUP BY ff.OrganizationKey,
         do.PercentageOfOwnership
ORDER BY ff.OrganizationKey;
--FOR JSON PATH, ROOT ('ConvertToDecimal'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT ff.OrganizationKey,
       (CONVERT(DECIMAL(5, 2), do.PercentageOfOwnership)) AS percentageasdecimal
FROM dbo.DimOrganization AS do
    INNER JOIN dbo.FactFinance AS ff
        ON ff.OrganizationKey = do.OrganizationKey
GROUP BY ff.OrganizationKey,
         do.PercentageOfOwnership
ORDER BY ff.OrganizationKey
FOR JSON PATH, ROOT ('ConvertToDecimal'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (9)

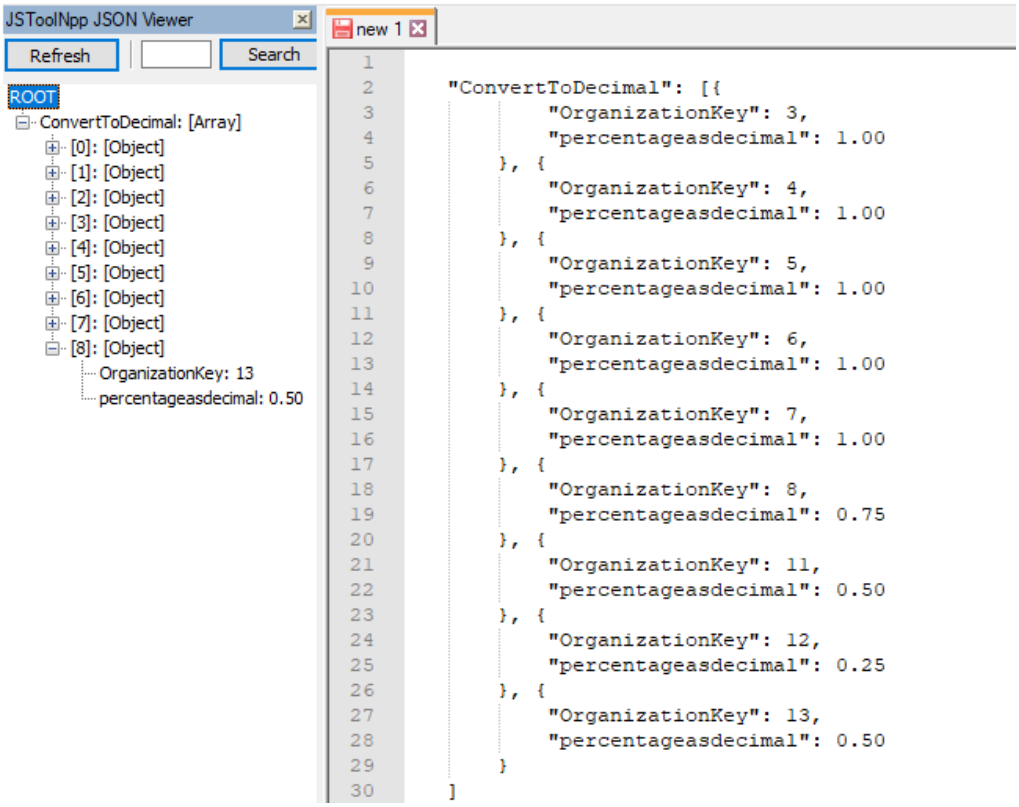


Results | Messages

	OrganizationKey	percentageasdecimal
1	3	1.00
2	4	1.00
3	5	1.00
4	6	1.00
5	7	1.00
6	8	0.75
7	11	0.50
8	12	0.25
9	13	0.50

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | AdventureWorksDW2017 | 00:00:00 | 9 rows

Sample JSON Output with total number of rows returned (9)



JSToolNpp JSON Viewer

Refresh | Search

ROOT

- ConvertToDecimal: [Array]
  - [0]: [Object]
  - [1]: [Object]
  - [2]: [Object]
  - [3]: [Object]
  - [4]: [Object]
  - [5]: [Object]
  - [6]: [Object]
  - [7]: [Object]
  - [8]: [Object]
    - OrganizationKey: 13
    - percentageasdecimal: 0.50

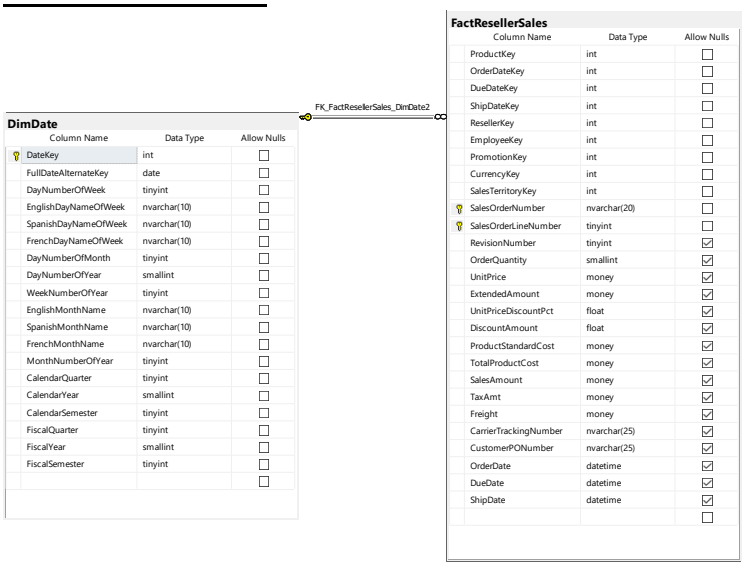
new 1

```
"ConvertToDecimal": [{
        "OrganizationKey": 3,
        "percentageasdecimal": 1.00
    }, {
        "OrganizationKey": 4,
        "percentageasdecimal": 1.00
    }, {
        "OrganizationKey": 5,
        "percentageasdecimal": 1.00
    }, {
        "OrganizationKey": 6,
        "percentageasdecimal": 1.00
    }, {
        "OrganizationKey": 7,
        "percentageasdecimal": 1.00
    }, {
        "OrganizationKey": 8,
        "percentageasdecimal": 0.75
    }, {
        "OrganizationKey": 11,
        "percentageasdecimal": 0.50
    }, {
        "OrganizationKey": 12,
        "percentageasdecimal": 0.25
    }, {
        "OrganizationKey": 13,
        "percentageasdecimal": 0.50
    }
]
```

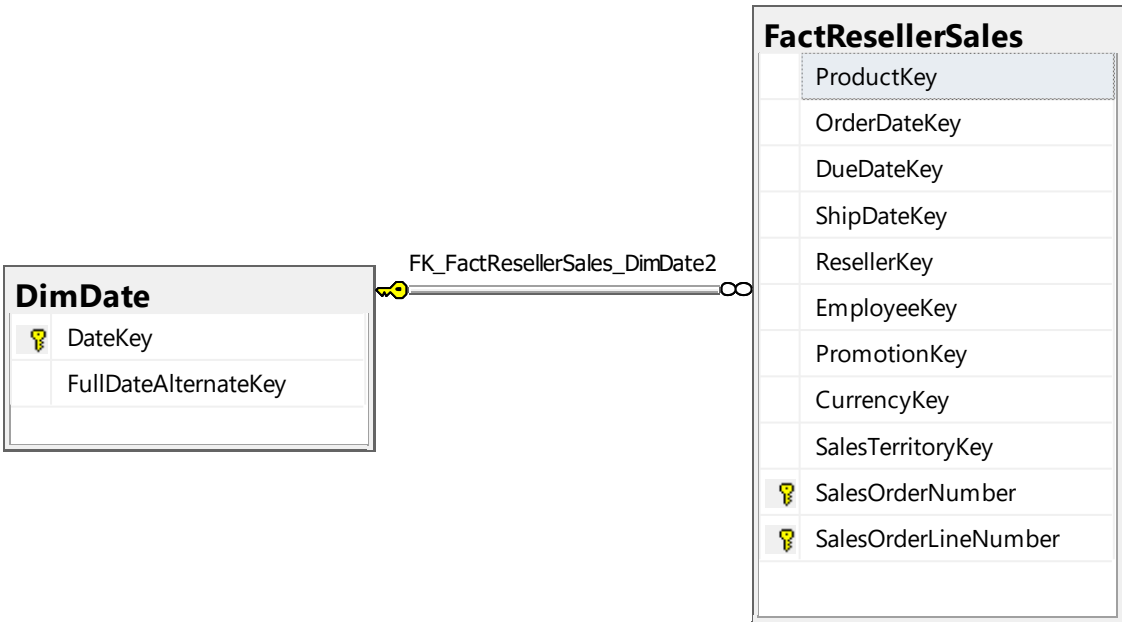
Medium Query 12:

```
--Using the DateKey from DimDate, find the difference between the DueDate and OrderDate
from FactResellerSales, using AdventureWorksDW2017.
```

Standard View:



Key View:



Columns from tables

Table Name	Column Names
DimDate	DateKey
FactResellerSales	OrderDate DueDate

Order By

Table Name	Column Name	Sort Order
DimDate	DateKey	ASC

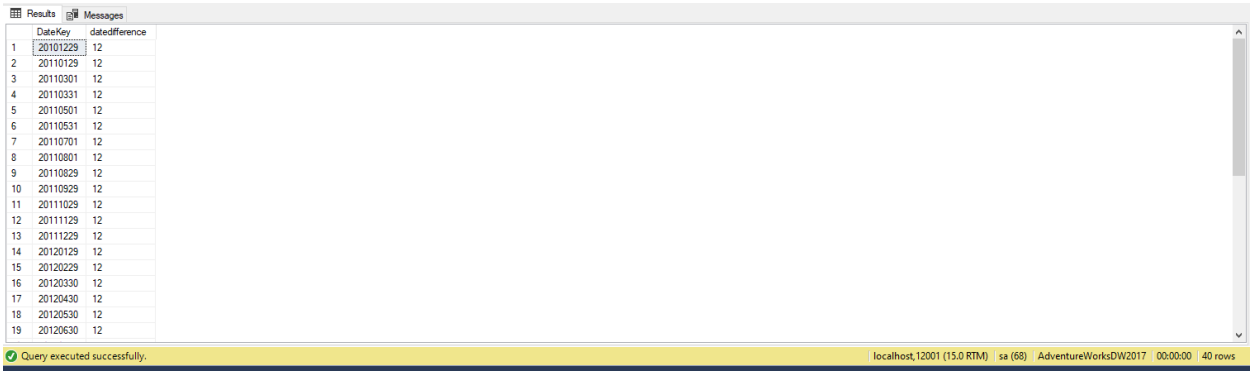
Without JSON:

```
SELECT dd.DateKey,
       DATEDIFF(DAY, frs.OrderDate, frs.DueDate) AS datedifference
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactResellerSales AS frs
        ON dd.DateKey = frs.OrderDateKey
GROUP BY dd.DateKey,
         frs.DueDate,
         frs.OrderDate
ORDER BY dd.DateKey;
--FOR JSON PATH, ROOT ('ShipTime'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT dd.DateKey,
       DATEDIFF(DAY, frs.OrderDate, frs.DueDate) AS datedifference
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactResellerSales AS frs
        ON dd.DateKey = frs.OrderDateKey
GROUP BY dd.DateKey,
         frs.DueDate,
         frs.OrderDate
ORDER BY dd.DateKey
FOR JSON PATH, ROOT ('ShipTime'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (40)

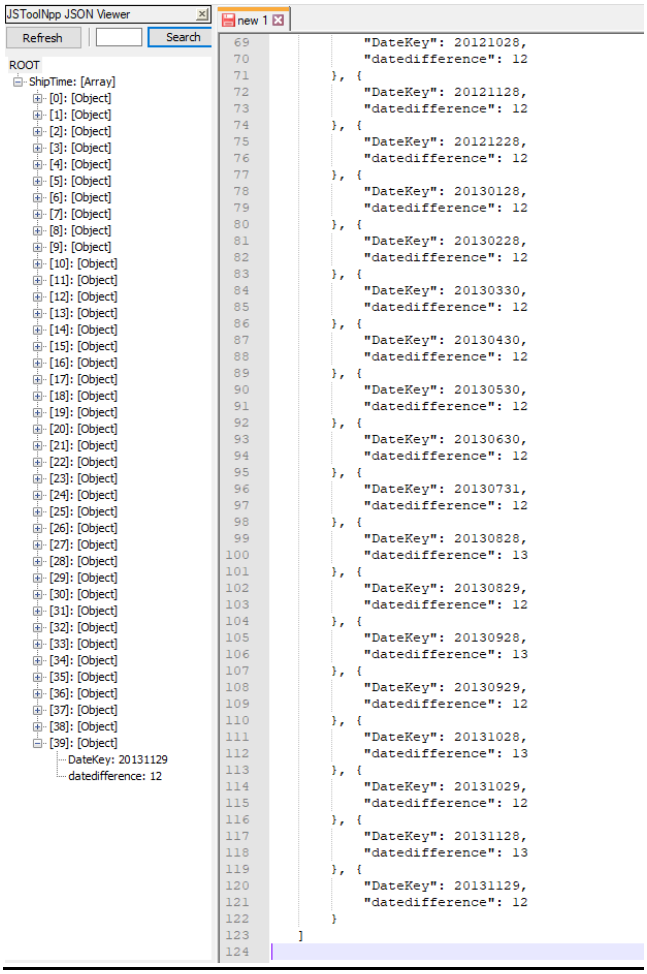


Results | Messages

	DateKey	datedifference
1	20101229	12
2	20110129	12
3	20110301	12
4	20110331	12
5	20110501	12
6	20110531	12
7	20110701	12
8	20110801	12
9	20110829	12
10	20110929	12
11	20111029	12
12	20111129	12
13	20111229	12
14	20120129	12
15	20120229	12
16	20120330	12
17	20120430	12
18	20120530	12
19	20120630	12

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | AdventureWorksDW2017 | 00:00:00 | 40 rows

Sample JSON Output with total number of rows returned (40)



JSToolNpp JSON Viewer

Refresh | Search

ROOT

- ShipTime: [Array]
  - [0]: [Object]
  - [1]: [Object]
  - [2]: [Object]
  - [3]: [Object]
  - [4]: [Object]
  - [5]: [Object]
  - [6]: [Object]
  - [7]: [Object]
  - [8]: [Object]
  - [9]: [Object]
  - [10]: [Object]
  - [11]: [Object]
  - [12]: [Object]
  - [13]: [Object]
  - [14]: [Object]
  - [15]: [Object]
  - [16]: [Object]
  - [17]: [Object]
  - [18]: [Object]
  - [19]: [Object]
  - [20]: [Object]
  - [21]: [Object]
  - [22]: [Object]
  - [23]: [Object]
  - [24]: [Object]
  - [25]: [Object]
  - [26]: [Object]
  - [27]: [Object]
  - [28]: [Object]
  - [29]: [Object]
  - [30]: [Object]
  - [31]: [Object]
  - [32]: [Object]
  - [33]: [Object]
  - [34]: [Object]
  - [35]: [Object]
  - [36]: [Object]
  - [37]: [Object]
  - [38]: [Object]
  - [39]: [Object]
    - DateKey: 20131129
    - datedifference: 12

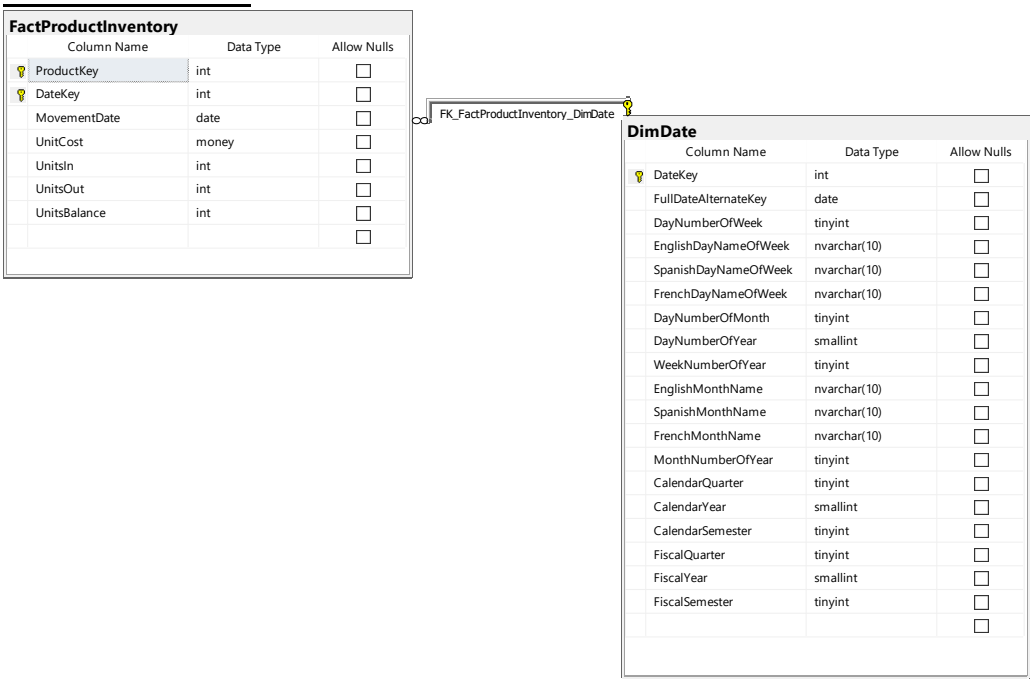
new 1


```
            "DateKey": 20121028,
            "datedifference": 12
        }, {
            "DateKey": 20121128,
            "datedifference": 12
        }, {
            "DateKey": 20121228,
            "datedifference": 12
        }, {
            "DateKey": 20130128,
            "datedifference": 12
        }, {
            "DateKey": 20130228,
            "datedifference": 12
        }, {
            "DateKey": 20130330,
            "datedifference": 12
        }, {
            "DateKey": 20130430,
            "datedifference": 12
        }, {
            "DateKey": 20130530,
            "datedifference": 12
        }, {
            "DateKey": 20130630,
            "datedifference": 12
        }, {
            "DateKey": 20130731,
            "datedifference": 12
        }, {
            "DateKey": 20130828,
            "datedifference": 13
        }, {
            "DateKey": 20130829,
            "datedifference": 12
        }, {
            "DateKey": 20130928,
            "datedifference": 13
        }, {
            "DateKey": 20130929,
            "datedifference": 12
        }, {
            "DateKey": 20131028,
            "datedifference": 13
        }, {
            "DateKey": 20131029,
            "datedifference": 12
        }, {
            "DateKey": 20131128,
            "datedifference": 13
        }, {
            "DateKey": 20131129,
            "datedifference": 12
        }
    ]
```

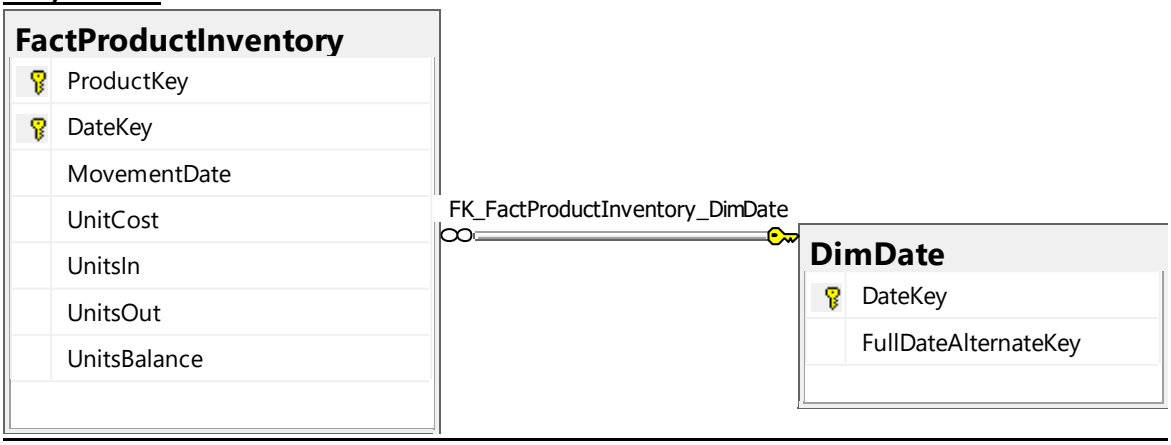
Medium Query 13:

```
--Using the DateKey from DimDate, find the average amount of unitbalance from
FactProductInventory for each calendar year, using AdventureWorksDW2017.
```

Standard view:



Key View:



Columns from tables

Table Name	Column Names
DimDate	CalendarYear
FactProductInventory	UnitsBalance

Order By

Table Name	Column Name	Sort Order
DimDate	CalendarYear	ASC

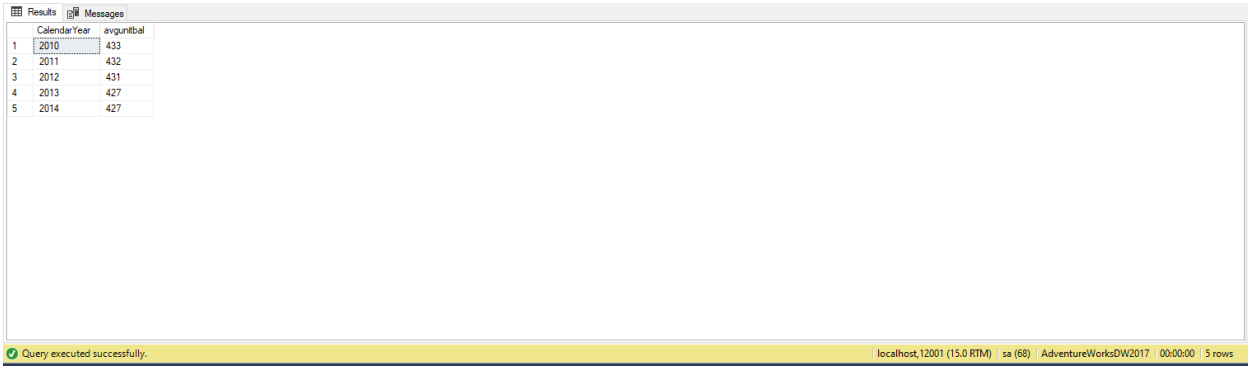
Without JSON:

```
SELECT dd.CalendarYear,
       AVG(fpi.UnitsBalance) AS avgunitbal
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactProductInventory AS fpi
        ON dd.DateKey = fpi.DateKey
GROUP BY dd.CalendarYear
ORDER BY dd.CalendarYear;
--FOR JSON PATH, ROOT ('AVGUnitsBalance'), INCLUDE_NULL_VALUES;
```

With JSON:

```
SELECT dd.CalendarYear,
       AVG(fpi.UnitsBalance) AS avgunitbal
FROM dbo.DimDate AS dd
    INNER JOIN dbo.FactProductInventory AS fpi
        ON dd.DateKey = fpi.DateKey
GROUP BY dd.CalendarYear
ORDER BY dd.CalendarYear
FOR JSON PATH, ROOT ('AVGUnitsBalance'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (5)

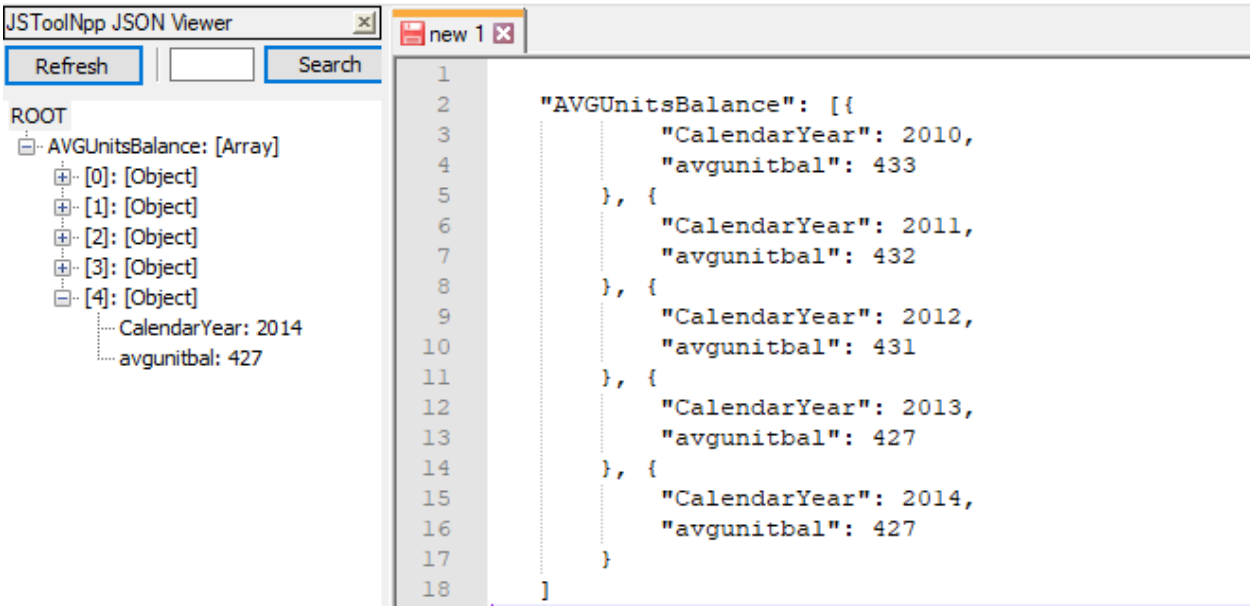


Results | Messages

	CalendarYear	avgunitbal
1	2010	433
2	2011	432
3	2012	431
4	2013	427
5	2014	427

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | AdventureWorksDW2017 | 00:00:00 | 5 rows

Sample JSON Output with total number of rows returned (5)



JSToolNpp JSON Viewer

Refresh | Search

ROOT

- AVGUnitsBalance: [Array]
  - [0]: [Object]
  - [1]: [Object]
  - [2]: [Object]
  - [3]: [Object]
  - [4]: [Object]
    - CalendarYear: 2014
    - avgunitbal: 427

new 1

```
{
    "AVGUnitsBalance": [{
            "CalendarYear": 2010,
            "avgunitbal": 433
        }, {
            "CalendarYear": 2011,
            "avgunitbal": 432
        }, {
            "CalendarYear": 2012,
            "avgunitbal": 431
        }, {
            "CalendarYear": 2013,
            "avgunitbal": 427
        }, {
            "CalendarYear": 2014,
            "avgunitbal": 427
        }
    ]
```

Complex Queries

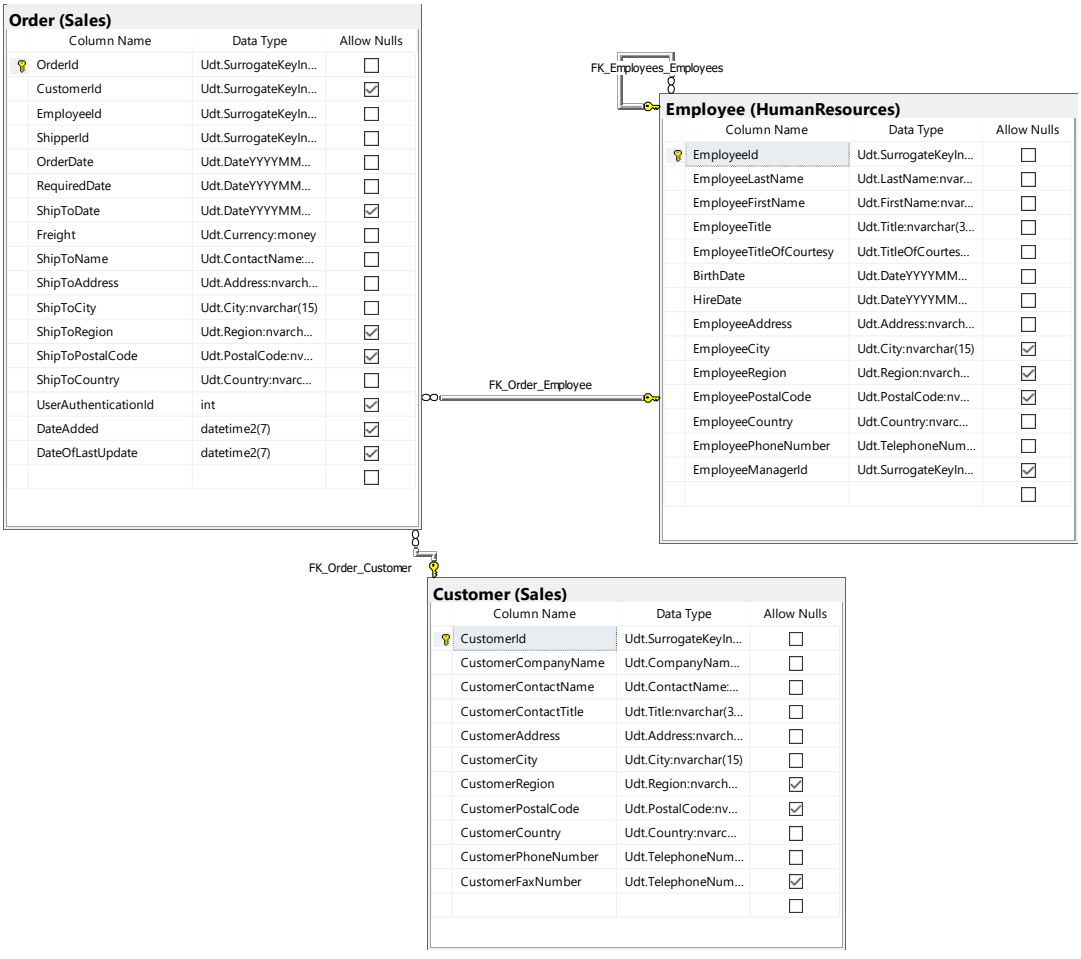
- All complex queries use the Northwinds2020TSQLV6 database
- All complex queries use the Orders subsystem

Complex Query Propositions:

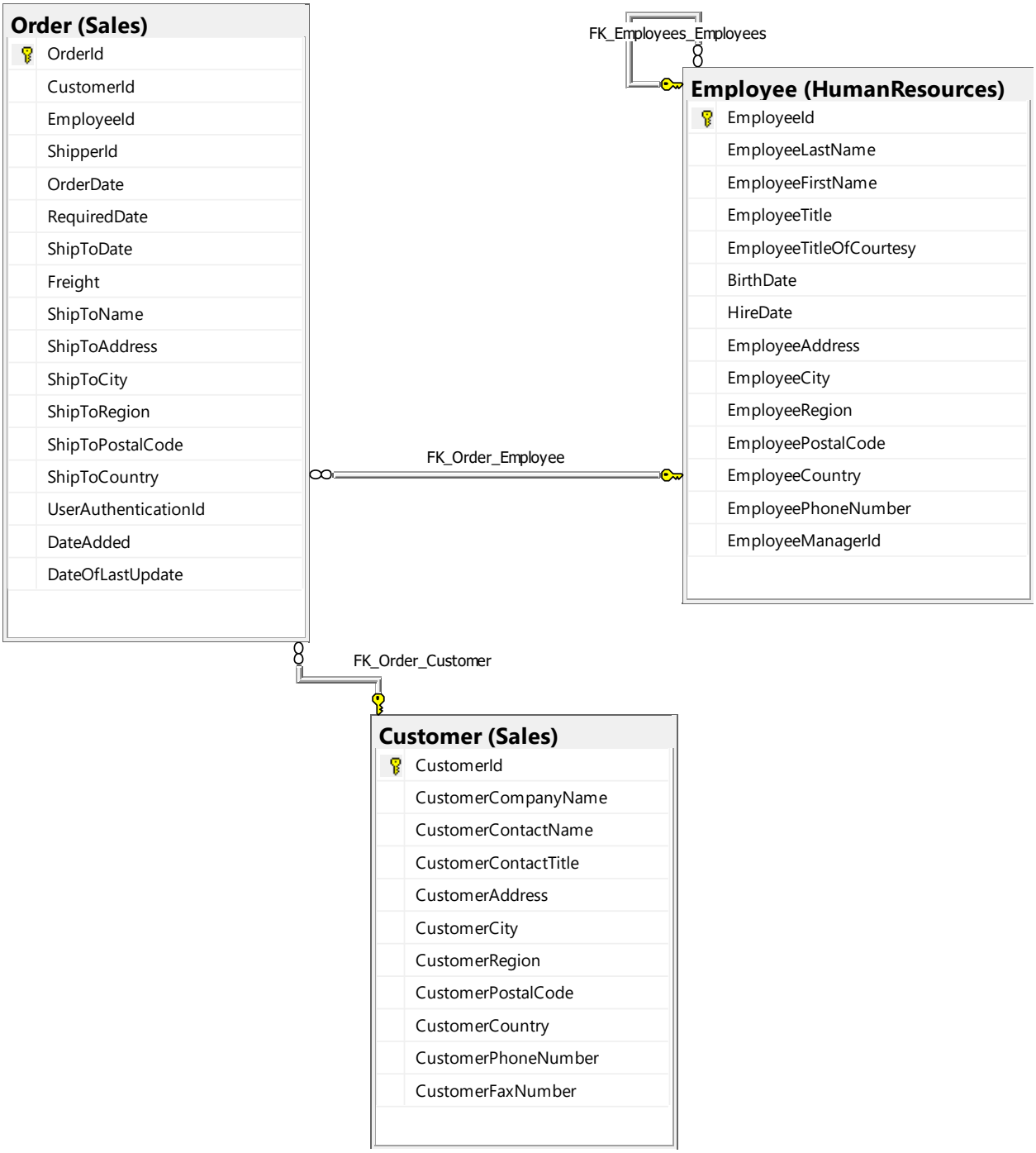
Complex Query 14:

```
--Create a scalar function to combine the first name and last name of each employee from
HumanResources.Employee and the contact name and title in Sales.Customer using the
OrderID from Sales.[Order], using Northwinds2020TSQLV6.
```

Standard view:



Key view:


Order (Sales)
OrderId
CustomerId
EmployeeId
ShipperId
OrderDate
RequiredDate
ShipToDate
Freight
ShipToName
ShipToAddress
ShipToCity
ShipToRegion
ShipToPostalCode
ShipToCountry
UserAuthenticationId
DateAdded
DateOfLastUpdate
FK_Employees_Employees
Employee (HumanResources)
EmployeeId
EmployeeLastName
EmployeeFirstName
EmployeeTitle
EmployeeTitleOfCourtesy
BirthDate
HireDate
EmployeeAddress
EmployeeCity
EmployeeRegion
EmployeePostalCode
EmployeeCountry
EmployeePhoneNumber
EmployeeManagerId
FK_Order_Employee
FK_Order_Customer
Customer (Sales)
CustomerId
CustomerCompanyName
CustomerContactName
CustomerContactTitle
CustomerAddress
CustomerCity
CustomerRegion
CustomerPostalCode
CustomerCountry
CustomerPhoneNumber
CustomerFaxNumber

Columns from tables

Table Name	Column Names
Order	OrderId
Employee	EmployeeFirstName EmployeeLastName
Customer	CustomerContactName CustomerContactTitle

Order By

Table Name	Column Name	Sort Order
Order	OrderId	ASC

Without JSON:

```
USE Northwinds2020TSQLV6;
GO

CREATE OR ALTER FUNCTION Sales.udf_FullName
(
    @firstname NVARCHAR(100),
    @lastname NVARCHAR(100)
)
RETURNS NVARCHAR(100)
AS
BEGIN
    RETURN @firstname + ' ' + @lastname;
END;
GO

USE Northwinds2020TSQLV6;
SELECT so.OrderId,
       Sales.udf_FullName(hr.EmployeeFirstName, hr.EmployeeLastName) AS employeefullname,
       Sales.udf_FullName(sc.CustomerContactName, sc.CustomerContactTitle) AS
customerfullname
FROM Sales.[Order] AS so
    INNER JOIN HumanResources.Employee AS hr
        ON so.EmployeeId = hr.EmployeeId
    INNER JOIN Sales.Customer AS sc
        ON so.CustomerId = sc.CustomerId
GROUP BY so.OrderId,
         hr.EmployeeFirstName,
         hr.EmployeeLastName,
         sc.CustomerContactName,
         sc.CustomerContactTitle
ORDER BY so.OrderId;
--FOR JSON PATH, ROOT ('FullName'), INCLUDE_NULL_VALUES;
```

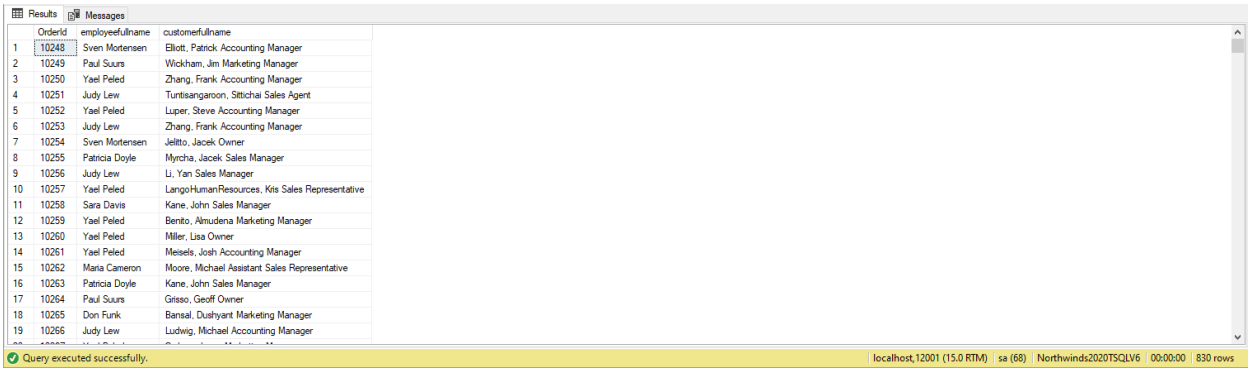
With JSON:

```
USE Northwinds2020TSQLV6;
GO

CREATE OR ALTER FUNCTION Sales.udf_FullName
(
    @firstname NVARCHAR(100),
    @lastname NVARCHAR(100)
)
RETURNS NVARCHAR(100)
AS
BEGIN
    RETURN @firstname + ' ' + @lastname;
END;
GO

USE Northwinds2020TSQLV6;
SELECT so.OrderId,
       Sales.udf_FullName(hr.EmployeeFirstName, hr.EmployeeLastName) AS employeefullname,
       Sales.udf_FullName(sc.CustomerContactName, sc.CustomerContactTitle) AS
customerfullname
FROM Sales.[Order] AS so
    INNER JOIN HumanResources.Employee AS hr
        ON so.EmployeeId = hr.EmployeeId
    INNER JOIN Sales.Customer AS sc
        ON so.CustomerId = sc.CustomerId
GROUP BY so.OrderId,
         hr.EmployeeFirstName,
         hr.EmployeeLastName,
         sc.CustomerContactName,
         sc.CustomerContactTitle
ORDER BY so.OrderId
FOR JSON PATH, ROOT ('FullName'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (830)

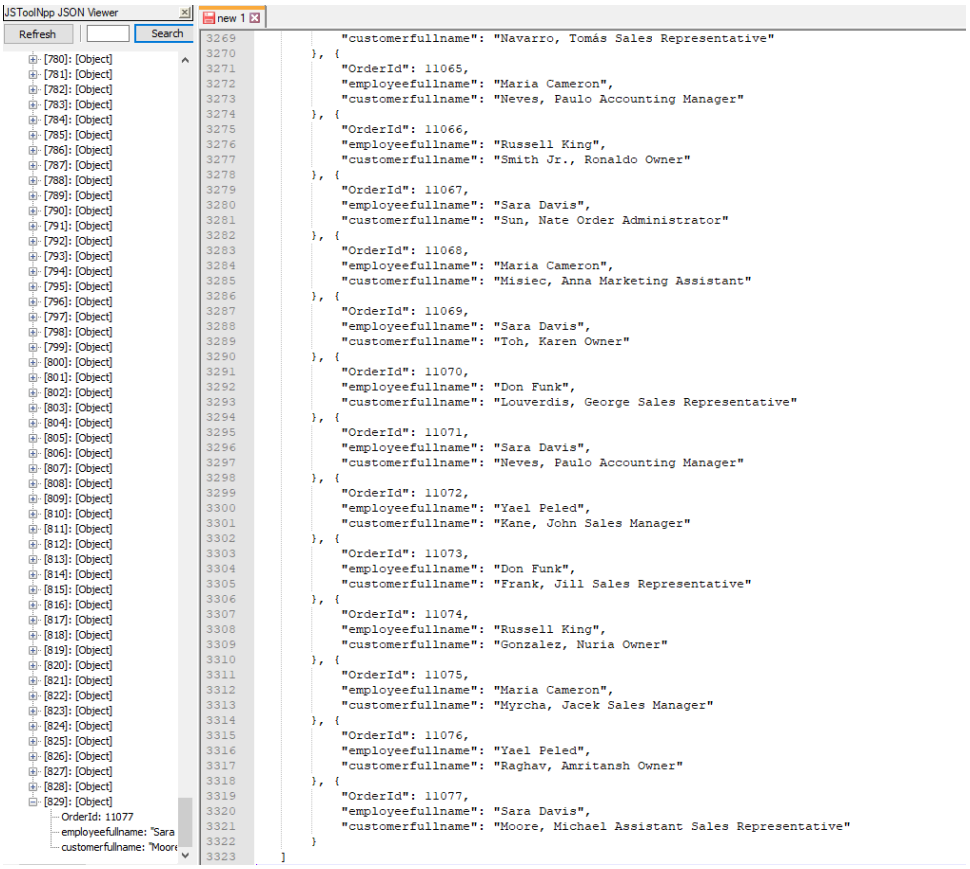


Results | Messages

	OrderId	employeefullname	customerfullname
1	10248	Sven Mortensen	Elliott, Patrick Accounting Manager
2	10249	Paul Suurs	Wickham, Jim Marketing Manager
3	10250	Yael Peled	Zhang, Frank Accounting Manager
4	10251	Judy Lew	Tuntisangaroon, Sittichai Sales Agent
5	10252	Yael Peled	Luper, Steve Accounting Manager
6	10253	Judy Lew	Zhang, Frank Accounting Manager
7	10254	Sven Mortensen	Jelitto, Jacek Owner
8	10255	Patricia Doyle	Myrcha, Jacek Sales Manager
9	10256	Judy Lew	Li, Yan Sales Manager
10	10257	Yael Peled	LangoHumanResources, Kris Sales Representative
11	10258	Sara Davis	Kane, John Sales Manager
12	10259	Yael Peled	Benito, Almudena Marketing Manager
13	10260	Yael Peled	Miller, Lisa Owner
14	10261	Yael Peled	Meisels, Josh Accounting Manager
15	10262	Maria Cameron	Moore, Michael Assistant Sales Representative
16	10263	Patricia Doyle	Kane, John Sales Manager
17	10264	Paul Suurs	Grisso, Geoff Owner
18	10265	Don Funk	Bansal, Dushyant Marketing Manager
19	10266	Judy Lew	Ludwig, Michael Accounting Manager

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | Northwinds2020TSQLV6 | 00:00:00 | 830 rows

Sample JSON Output with total number of rows returned (830)



JSToolNpp JSON Viewer

Refresh | Search

[780]: [Object]
[781]: [Object]
[782]: [Object]
[783]: [Object]
[784]: [Object]
[785]: [Object]
[786]: [Object]
[787]: [Object]
[788]: [Object]
[789]: [Object]
[790]: [Object]
[791]: [Object]
[792]: [Object]
[793]: [Object]
[794]: [Object]
[795]: [Object]
[796]: [Object]
[797]: [Object]
[798]: [Object]
[799]: [Object]
[800]: [Object]
[801]: [Object]
[802]: [Object]
[803]: [Object]
[804]: [Object]
[805]: [Object]
[806]: [Object]
[807]: [Object]
[808]: [Object]
[809]: [Object]
[810]: [Object]
[811]: [Object]
[812]: [Object]
[813]: [Object]
[814]: [Object]
[815]: [Object]
[816]: [Object]
[817]: [Object]
[818]: [Object]
[819]: [Object]
[820]: [Object]
[821]: [Object]
[822]: [Object]
[823]: [Object]
[824]: [Object]
[825]: [Object]
[826]: [Object]
[827]: [Object]
[828]: [Object]
[829]: [Object]
OrderId: 11077
employeefullname: "Sara
customerfullname: "Moore

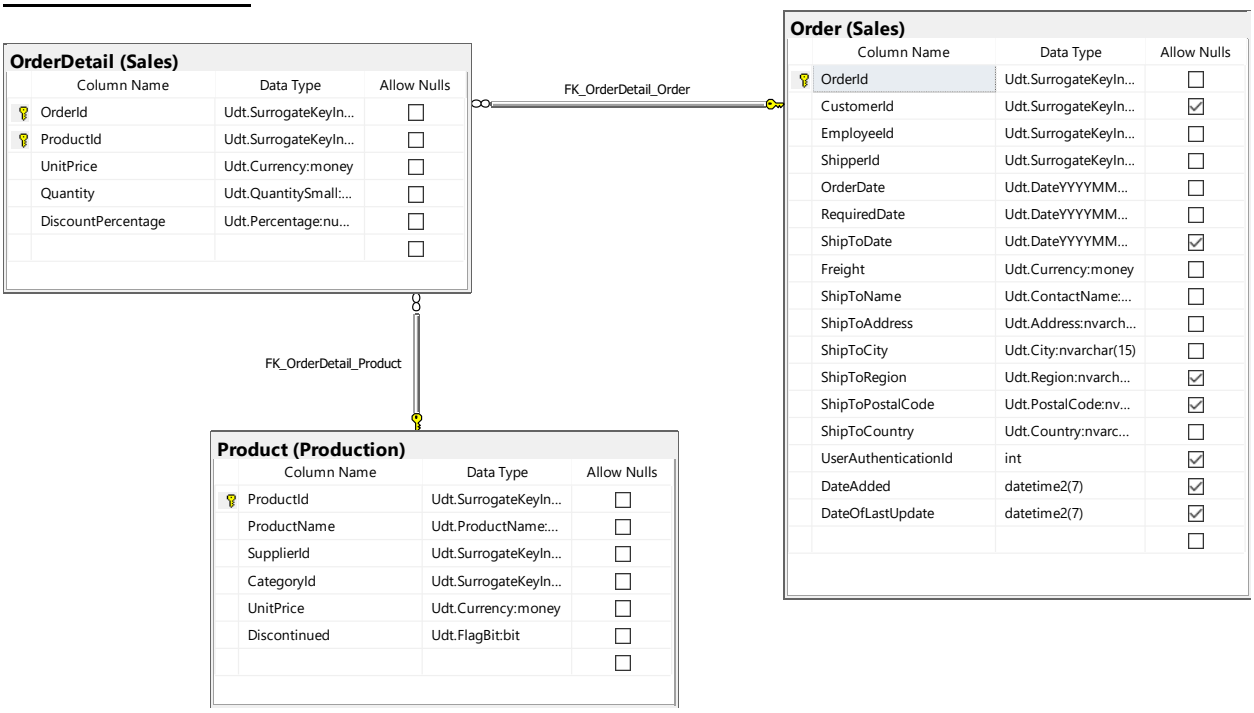
new 1


```
        "customerfullname": "Navarro, Tomás Sales Representative"
    }, {
        "OrderId": 11065,
        "employeefullname": "Maria Cameron",
        "customerfullname": "Neves, Paulo Accounting Manager"
    }, {
        "OrderId": 11066,
        "employeefullname": "Russell King",
        "customerfullname": "Smith Jr., Ronaldo Owner"
    }, {
        "OrderId": 11067,
        "employeefullname": "Sara Davis",
        "customerfullname": "Sun, Nate Order Administrator"
    }, {
        "OrderId": 11068,
        "employeefullname": "Maria Cameron",
        "customerfullname": "Misiec, Anna Marketing Assistant"
    }, {
        "OrderId": 11069,
        "employeefullname": "Sara Davis",
        "customerfullname": "Toh, Karen Owner"
    }, {
        "OrderId": 11070,
        "employeefullname": "Don Funk",
        "customerfullname": "Louverdis, George Sales Representative"
    }, {
        "OrderId": 11071,
        "employeefullname": "Sara Davis",
        "customerfullname": "Neves, Paulo Accounting Manager"
    }, {
        "OrderId": 11072,
        "employeefullname": "Yael Peled",
        "customerfullname": "Kane, John Sales Manager"
    }, {
        "OrderId": 11073,
        "employeefullname": "Don Funk",
        "customerfullname": "Frank, Jill Sales Representative"
    }, {
        "OrderId": 11074,
        "employeefullname": "Russell King",
        "customerfullname": "Gonzalez, Nuria Owner"
    }, {
        "OrderId": 11075,
        "employeefullname": "Maria Cameron",
        "customerfullname": "Myrcha, Jacek Sales Manager"
    }, {
        "OrderId": 11076,
        "employeefullname": "Yael Peled",
        "customerfullname": "Raghav, Amritansh Owner"
    }, {
        "OrderId": 11077,
        "employeefullname": "Sara Davis",
        "customerfullname": "Moore, Michael Assistant Sales Representative"
    }
]
```

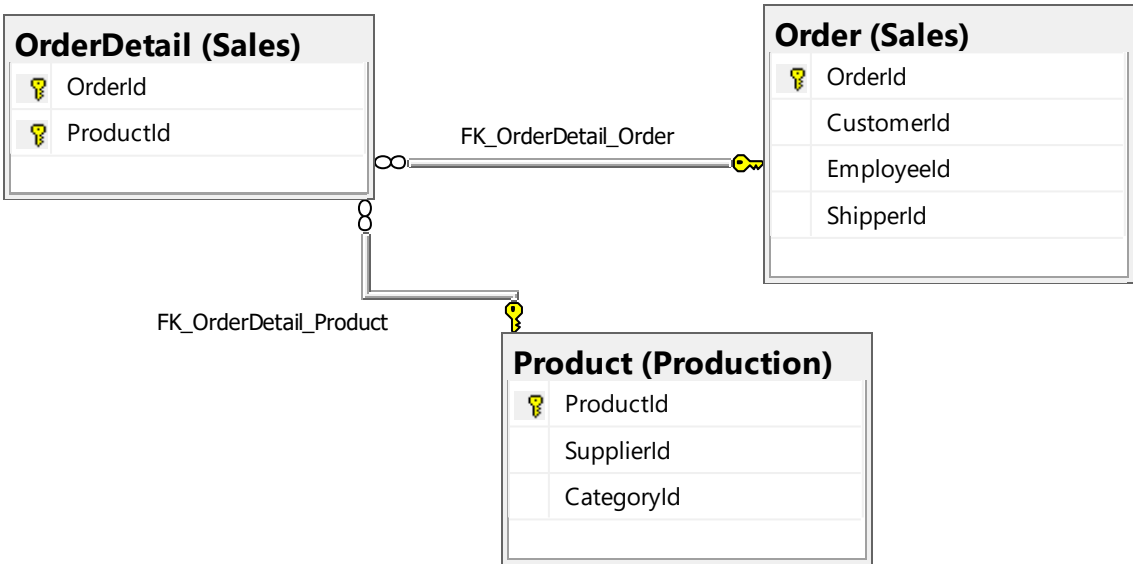
Complex Query 15:

```
--Create a scalar function that returns average unit price from Production.Product and
the ship date from Sales.[Order] using the OrderId in Sales.OrderDetail, using
Northwinds2020TSQLV6.
```

Standard view:



Key view:



Columns from tables

Table Name	Column Names
Order	ShipToDate
OrderDetail	OrderId
Product	UnitPrice

Order By

Table Name	Column Name	Sort Order
OrderDetail	OrderId	ASC

Without JSON:

```
USE Northwinds2020TSQLV6;
GO

CREATE OR ALTER FUNCTION Sales.udf_AverageUnit
(
    @unitprice MONEY
)
RETURNS MONEY
BEGIN
    DECLARE @average MONEY;
    SELECT @average = (SUM(@unitprice) / COUNT(@unitprice));
    RETURN @average;
END;
GO

USE Northwinds2020TSQLV6;
SELECT od.OrderId,
       Sales.udf_AverageUnit(pp.UnitPrice) AS avgunitprice,
       o.ShipToDate AS shiptodate
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.orderid
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY od.OrderId,
         pp.UnitPrice,
         o.ShipToDate
ORDER BY od.OrderId;
--FOR JSON PATH, ROOT ('AverageUnit'), INCLUDE_NULL_VALUES;
```

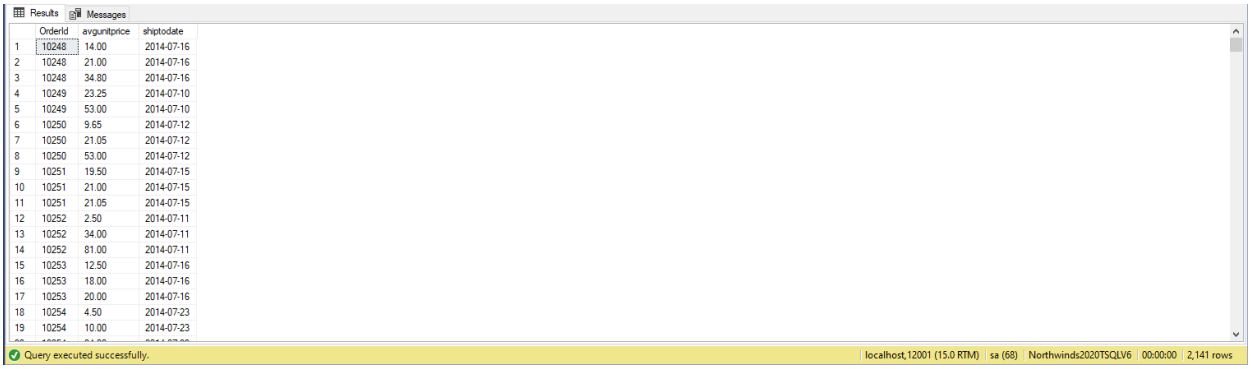
With JSON:

```
USE Northwinds2020TSQLV6;
GO

CREATE OR ALTER FUNCTION Sales.udf_AverageUnit
(
    @unitprice MONEY
)
RETURNS MONEY
BEGIN
    DECLARE @average MONEY;
    SELECT @average = (SUM(@unitprice) / COUNT(@unitprice));
    RETURN @average;
END;
GO

USE Northwinds2020TSQLV6;
SELECT od.OrderId,
       pp.UnitPrice AS avgunitprice,
       o.ShipToDate AS shiptodate
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.orderid
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY od.OrderId,
         pp.UnitPrice,
         o.ShipToDate
ORDER BY od.OrderId
FOR JSON PATH, ROOT ('AverageUnit'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (2141)

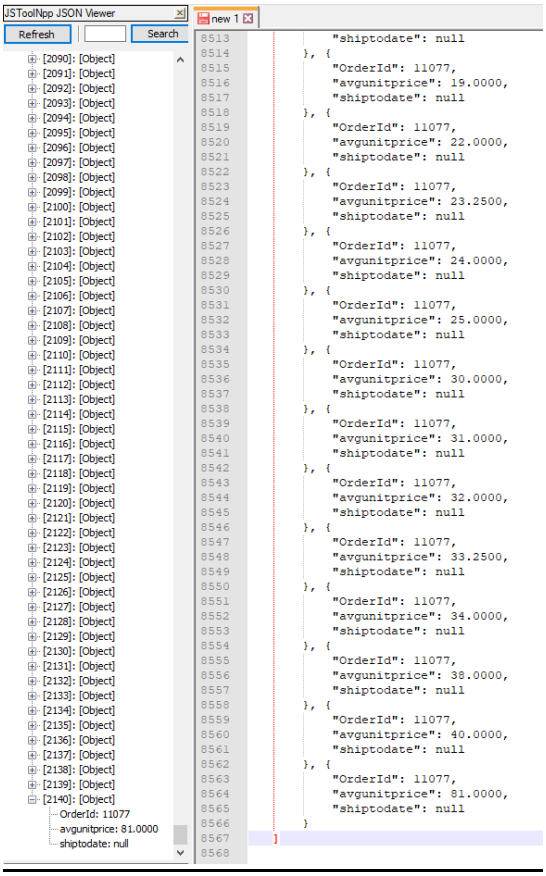


Results | Messages

	OrderId	avgunitprice	shiptodate
1	10248	14.00	2014-07-16
2	10248	21.00	2014-07-16
3	10248	34.80	2014-07-16
4	10249	23.25	2014-07-10
5	10249	53.00	2014-07-10
6	10250	9.65	2014-07-12
7	10250	21.05	2014-07-12
8	10250	53.00	2014-07-12
9	10251	19.50	2014-07-15
10	10251	21.00	2014-07-15
11	10251	21.05	2014-07-15
12	10252	2.50	2014-07-11
13	10252	34.00	2014-07-11
14	10252	81.00	2014-07-11
15	10253	12.50	2014-07-16
16	10253	18.00	2014-07-16
17	10253	20.00	2014-07-16
18	10254	4.50	2014-07-23
19	10254	10.00	2014-07-23

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | Northwinds2020TSQLV6 | 00:00:00 | 2,141 rows

Sample JSON Output with total number of rows returned (2141)



JSToolNpp JSON Viewer

Refresh | Search

[2090]: [Object]
[2091]: [Object]
[2092]: [Object]
[2093]: [Object]
[2094]: [Object]
[2095]: [Object]
[2096]: [Object]
[2097]: [Object]
[2098]: [Object]
[2099]: [Object]
[2100]: [Object]
[2101]: [Object]
[2102]: [Object]
[2103]: [Object]
[2104]: [Object]
[2105]: [Object]
[2106]: [Object]
[2107]: [Object]
[2108]: [Object]
[2109]: [Object]
[2110]: [Object]
[2111]: [Object]
[2112]: [Object]
[2113]: [Object]
[2114]: [Object]
[2115]: [Object]
[2116]: [Object]
[2117]: [Object]
[2118]: [Object]
[2119]: [Object]
[2120]: [Object]
[2121]: [Object]
[2122]: [Object]
[2123]: [Object]
[2124]: [Object]
[2125]: [Object]
[2126]: [Object]
[2127]: [Object]
[2128]: [Object]
[2129]: [Object]
[2130]: [Object]
[2131]: [Object]
[2132]: [Object]
[2133]: [Object]
[2134]: [Object]
[2135]: [Object]
[2136]: [Object]
[2137]: [Object]
[2138]: [Object]
[2139]: [Object]
[2140]: [Object]
OrderId: 11077
avgunitprice: 81.0000
shiptodate: null

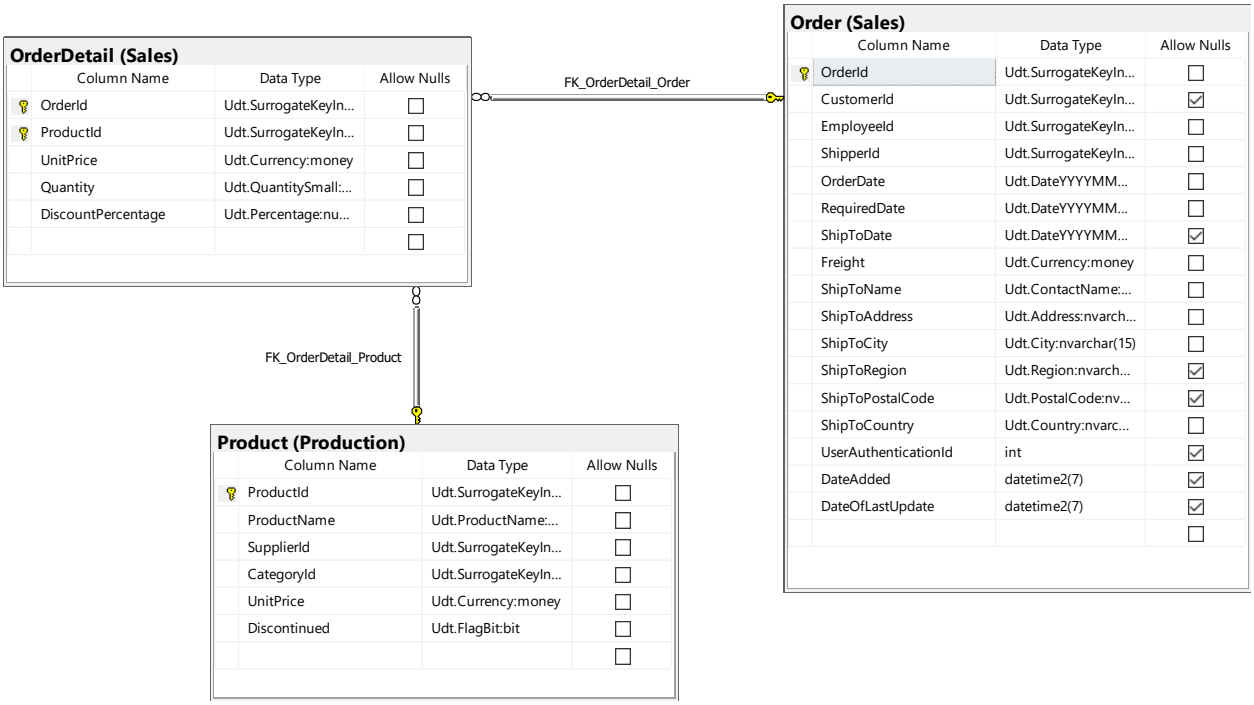
new 1


```
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 19.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 22.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 23.2500,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 24.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 25.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 30.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 31.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 32.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 33.2500,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 34.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 38.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 40.0000,
        "shiptodate": null
    }, {
        "OrderId": 11077,
        "avgunitprice": 81.0000,
        "shiptodate": null
    }
]
```

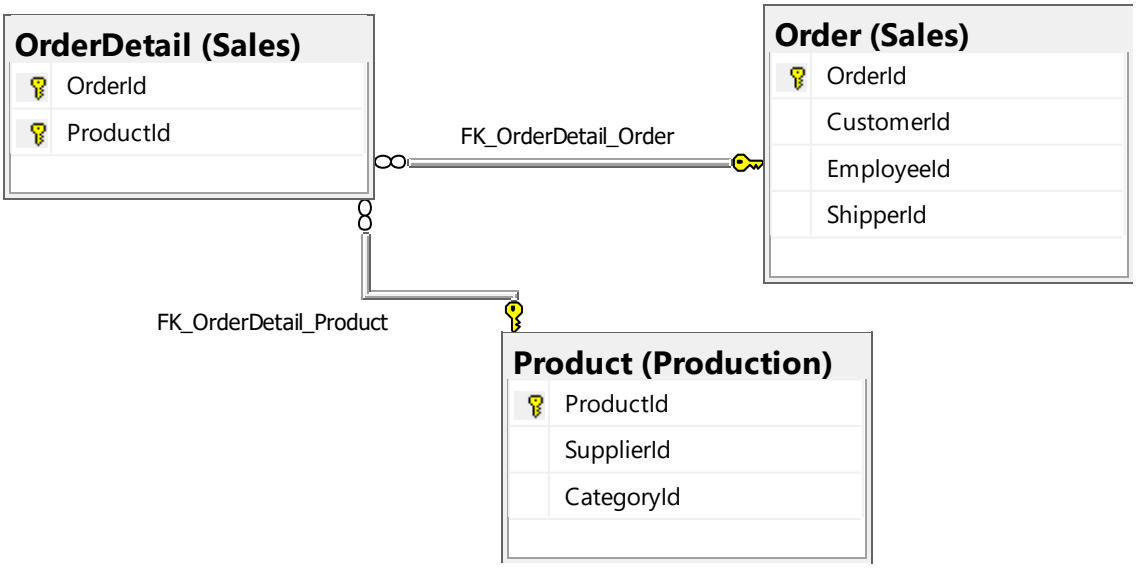
Complex Query 16:

```
--Create a column for the nextorderid in Sales.[Order], display ProductId from
Sales.OrderDetail and ProductName from Production.Product, using Northwinds2020TSQLV6.
```

Standard view:



Key view:



Columns from tables

Table Name	Column Names
Order	OrderId
OrderDetail	ProductId
Product	ProductName

Order By

Table Name	Column Name	Sort Order
Order	OrderId	ASC

Without JSON:

```
USE Northwinds2020TSQLV6;
GO

CREATE OR ALTER FUNCTION Sales.udf_NextOrder
(
    @currentorder INT
)
RETURNS INT
BEGIN
    DECLARE @nextorder INT;
    SELECT @nextorder = @currentorder + 1;
    RETURN @nextorder;
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
       Sales.udf_NextOrder(o.OrderId) AS nextorderid,
       od.ProductId,
       pp.ProductName
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.OrderId
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY o.OrderId,
         od.ProductId,
         pp.ProductName
ORDER BY o.OrderId;
--FOR JSON PATH, ROOT ('NextOrder'), INCLUDE_NULL_VALUES;
```

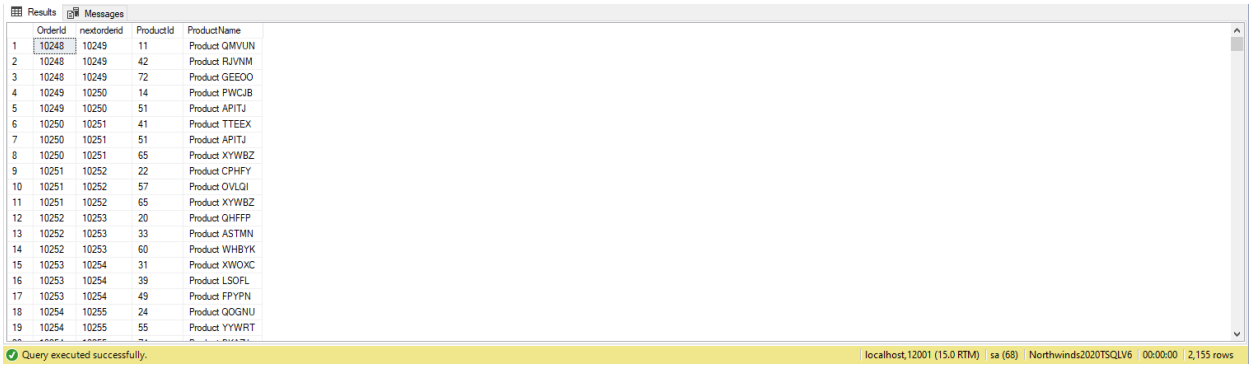
With JSON:

```
USE Northwinds2020TSQLV6;
GO

CREATE OR ALTER FUNCTION Sales.udf_NextOrder
(
    @currentorder INT
)
RETURNS INT
BEGIN
    DECLARE @nextorder INT;
    SELECT @nextorder = @currentorder + 1;
    RETURN @nextorder;
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
       Sales.udf_NextOrder(o.OrderId) AS nextorderid,
       od.ProductId,
       pp.ProductName
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.OrderId
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY o.OrderId,
         od.ProductId,
         pp.ProductName
ORDER BY o.OrderId
FOR JSON PATH, ROOT ('NextOrder'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (2155)

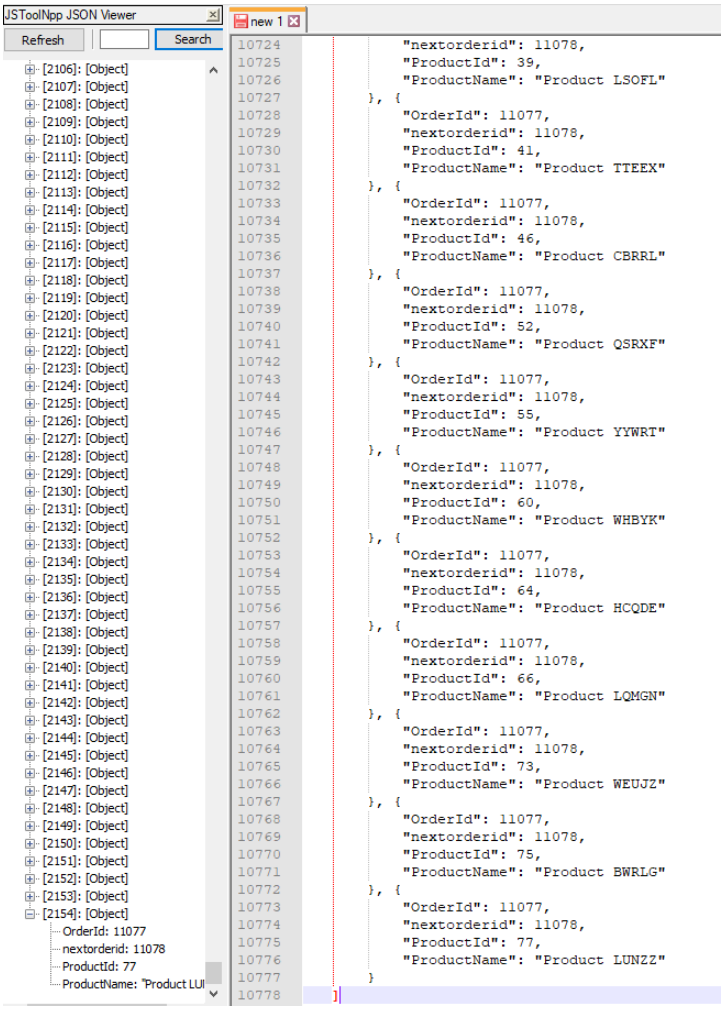


Results | Messages

	OrderId	nextorderid	ProductId	ProductName
1	10248	10249	11	Product QMVUN
2	10248	10249	42	Product RJVNM
3	10248	10249	72	Product GEEOO
4	10249	10250	14	Product PWCJB
5	10249	10250	51	Product APITJ
6	10250	10251	41	Product TTEEX
7	10250	10251	51	Product APITJ
8	10250	10251	65	Product XYWBZ
9	10251	10252	22	Product CPHFY
10	10251	10252	57	Product OVLQI
11	10251	10252	65	Product XYWBZ
12	10252	10253	20	Product QHFFP
13	10252	10253	33	Product ASTMN
14	10252	10253	60	Product WHBYK
15	10253	10254	31	Product XWOXC
16	10253	10254	39	Product LSOFL
17	10253	10254	49	Product FPYPN
18	10254	10255	24	Product QOGNU
19	10254	10255	55	Product YYWRT

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | Northwinds2020TSQLV6 | 00:00:00 | 2,155 rows

Sample JSON Output with total number of rows returned (2155)



JSToolNpp JSON Viewer

Refresh | Search

[2106]: [Object]
[2107]: [Object]
[2108]: [Object]
[2109]: [Object]
[2110]: [Object]
[2111]: [Object]
[2112]: [Object]
[2113]: [Object]
[2114]: [Object]
[2115]: [Object]
[2116]: [Object]
[2117]: [Object]
[2118]: [Object]
[2119]: [Object]
[2120]: [Object]
[2121]: [Object]
[2122]: [Object]
[2123]: [Object]
[2124]: [Object]
[2125]: [Object]
[2126]: [Object]
[2127]: [Object]
[2128]: [Object]
[2129]: [Object]
[2130]: [Object]
[2131]: [Object]
[2132]: [Object]
[2133]: [Object]
[2134]: [Object]
[2135]: [Object]
[2136]: [Object]
[2137]: [Object]
[2138]: [Object]
[2139]: [Object]
[2140]: [Object]
[2141]: [Object]
[2142]: [Object]
[2143]: [Object]
[2144]: [Object]
[2145]: [Object]
[2146]: [Object]
[2147]: [Object]
[2148]: [Object]
[2149]: [Object]
[2150]: [Object]
[2151]: [Object]
[2152]: [Object]
[2153]: [Object]
[2154]: [Object]
OrderId: 11077
nextorderid: 11078
ProductId: 77
ProductName: "Product LU

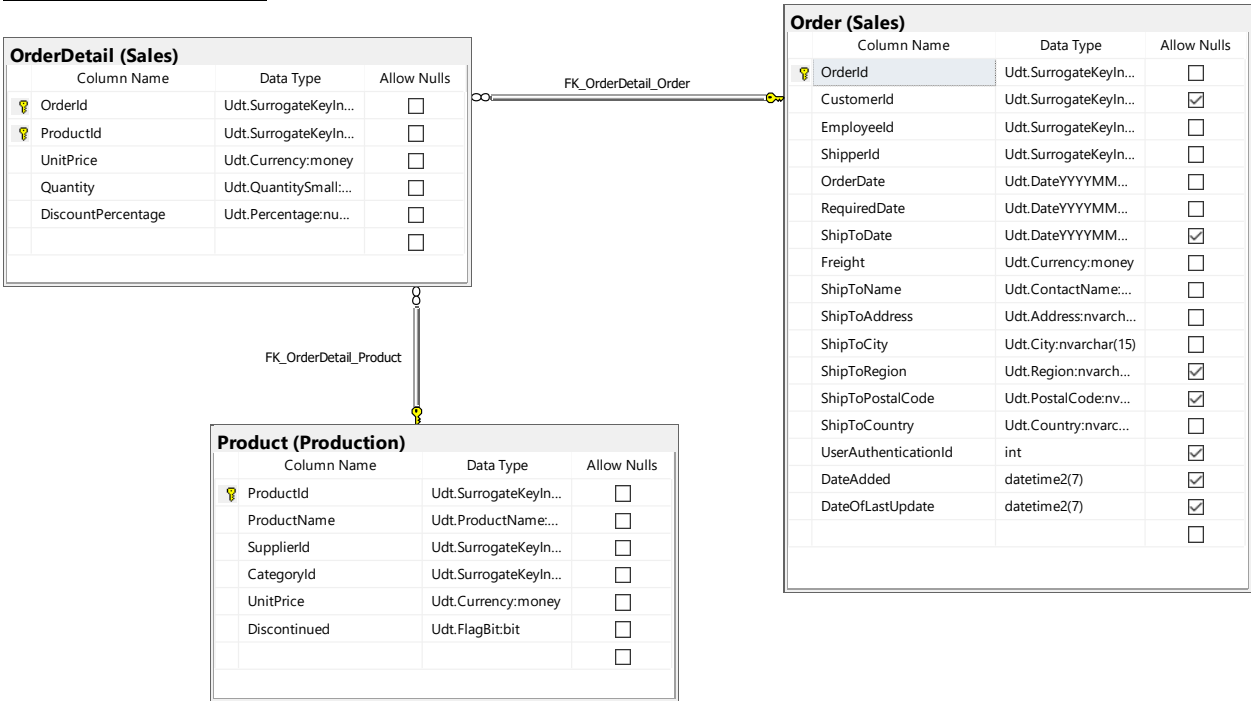
new 1


```
            "nextorderid": 11078,
            "ProductId": 39,
            "ProductName": "Product LSOFL"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 41,
            "ProductName": "Product TTEEX"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 46,
            "ProductName": "Product CBRRL"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 52,
            "ProductName": "Product QSRXF"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 55,
            "ProductName": "Product YYWRT"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 60,
            "ProductName": "Product WHBYK"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 64,
            "ProductName": "Product HCQDE"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 66,
            "ProductName": "Product LQMGN"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 73,
            "ProductName": "Product WEUJZ"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 75,
            "ProductName": "Product BWRLG"
        }, {
            "OrderId": 11077,
            "nextorderid": 11078,
            "ProductId": 77,
            "ProductName": "Product LUNZZ"
        }
    ]
```

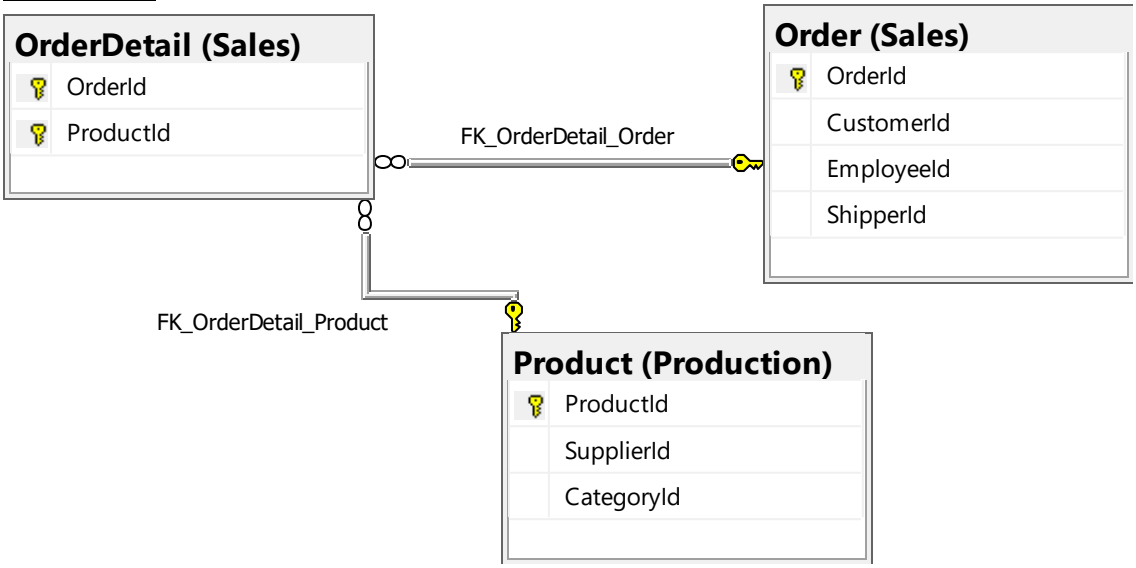
Complex Query 17:

```
--Find the calendar year quarter of the OrderDate from Sales.Order, the ProductID from
Sales.OrderDetail and the ProductName from Production.Product, using
Northwinds2020TSQLV6.
```

Standard View:



Key view:



Columns from tables

Table Name	Column Names
Order	OrderID OrderDate
OrderDetail	ProductID
Product	ProductName

Order By

Table Name	Column Name	Sort Order
Order	OrderId	ASC

Without JSON:

```
USE Northwinds2020TSQLV6;

GO
CREATE OR ALTER FUNCTION Sales.udf_CalendarYearQuarter
(
    @orderdate DATE
)
RETURNS NVARCHAR(20)
BEGIN
    DECLARE @CalendarDate VARCHAR(20);

    SELECT @CalendarDate = CASE
                               WHEN MONTH(@orderdate) >= 1
                                    AND MONTH(@orderdate) <= 3 THEN
                                   'Q1'
                               WHEN MONTH(@orderdate) >= 4
                                    AND MONTH(@orderdate) <= 6 THEN
                                   'Q2'
                               WHEN MONTH(@orderdate) >= 7
                                    AND MONTH(@orderdate) <= 9 THEN
                                   'Q3'
                               WHEN MONTH(@orderdate) >= 10
                                    AND MONTH(@orderdate) <= 12 THEN
                                   'Q4'
                               ELSE
                                   'N/A'
                           END;
    RETURN @CalendarDate;
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
          YEAR(o.OrderDate) AS orderyear,
       Sales.udf_CalendarYearQuarter(o.OrderDate) AS calendaryearquarter,
       od.ProductId,
       pp.ProductName
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.OrderId
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY o.OrderId,
            o.OrderDate,
         od.ProductId,
         pp.ProductName
ORDER BY o.OrderId;
--FOR JSON PATH, ROOT ('CalendarYearQuarter'), INCLUDE_NULL_VALUES;
```

With JSON:

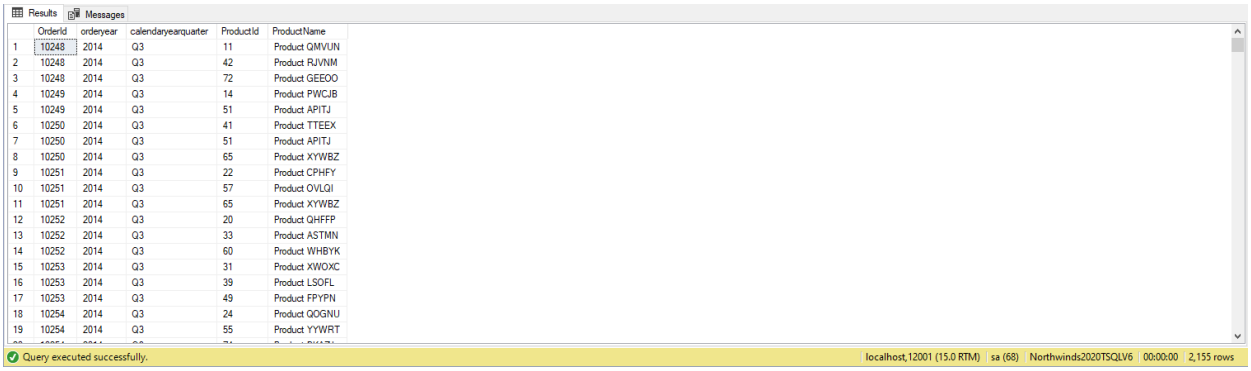
```
USE Northwinds2020TSQLV6;

GO
CREATE OR ALTER FUNCTION Sales.udf_CalendarYearQuarter
(
    @orderdate DATE
)
RETURNS NVARCHAR(20)
BEGIN
    DECLARE @CalendarDate VARCHAR(20);

    SELECT @CalendarDate = CASE
                               WHEN MONTH(@orderdate) >= 1
                                    AND MONTH(@orderdate) <= 3 THEN
                                   'Q1'
                               WHEN MONTH(@orderdate) >= 4
                                    AND MONTH(@orderdate) <= 6 THEN
                                   'Q2'
                               WHEN MONTH(@orderdate) >= 7
                                    AND MONTH(@orderdate) <= 9 THEN
                                   'Q3'
                               WHEN MONTH(@orderdate) >= 10
                                    AND MONTH(@orderdate) <= 12 THEN
                                   'Q4'
                               ELSE
                                   'N/A'
                           END;
    RETURN @CalendarDate;
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
          YEAR(o.OrderDate) AS orderyear,
       Sales.udf_CalendarYearQuarter(o.OrderDate) AS calendaryearquarter,
       od.ProductId,
       pp.ProductName
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.OrderId
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY o.OrderId,
            o.OrderDate,
         od.ProductId,
         pp.ProductName
ORDER BY o.OrderId
FOR JSON PATH, ROOT ('CalendarYearQuarter'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (2155)

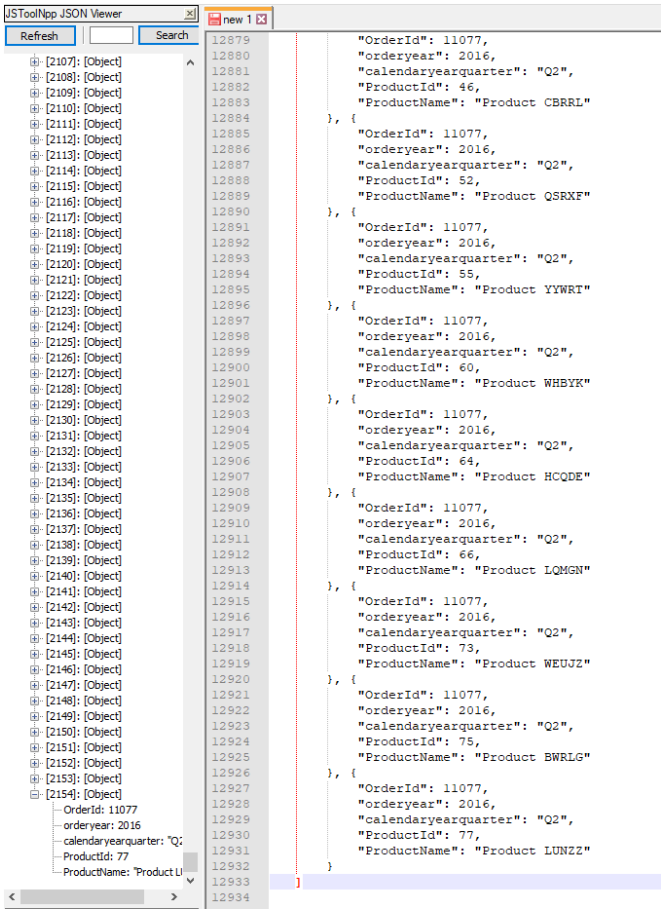


Results | Messages

	OrderId	orderyear	calendaryearquarter	ProductId	ProductName
1	10248	2014	Q3	11	Product QMVUN
2	10248	2014	Q3	42	Product RJVNM
3	10248	2014	Q3	72	Product GEEOO
4	10249	2014	Q3	14	Product PWCJB
5	10249	2014	Q3	51	Product APITJ
6	10250	2014	Q3	41	Product TTEEX
7	10250	2014	Q3	51	Product APITJ
8	10250	2014	Q3	65	Product XYWBZ
9	10251	2014	Q3	22	Product CPHFY
10	10251	2014	Q3	57	Product OVLQI
11	10251	2014	Q3	65	Product XYWBZ
12	10252	2014	Q3	20	Product QHFFP
13	10252	2014	Q3	33	Product ASTMN
14	10252	2014	Q3	60	Product WHBYK
15	10253	2014	Q3	31	Product XWOXC
16	10253	2014	Q3	39	Product LSOFL
17	10253	2014	Q3	49	Product FPYPN
18	10254	2014	Q3	24	Product QOGNU
19	10254	2014	Q3	55	Product YYWRT

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | Northwinds2020TSQLV6 | 00:00:00 | 2,155 rows

Sample JSON Output with total number of rows returned (2155)



JSToolNpp JSON Viewer

Refresh | Search

[2107]: [Object]
[2108]: [Object]
[2109]: [Object]
[2110]: [Object]
[2111]: [Object]
[2112]: [Object]
[2113]: [Object]
[2114]: [Object]
[2115]: [Object]
[2116]: [Object]
[2117]: [Object]
[2118]: [Object]
[2119]: [Object]
[2120]: [Object]
[2121]: [Object]
[2122]: [Object]
[2123]: [Object]
[2124]: [Object]
[2125]: [Object]
[2126]: [Object]
[2127]: [Object]
[2128]: [Object]
[2129]: [Object]
[2130]: [Object]
[2131]: [Object]
[2132]: [Object]
[2133]: [Object]
[2134]: [Object]
[2135]: [Object]
[2136]: [Object]
[2137]: [Object]
[2138]: [Object]
[2139]: [Object]
[2140]: [Object]
[2141]: [Object]
[2142]: [Object]
[2143]: [Object]
[2144]: [Object]
[2145]: [Object]
[2146]: [Object]
[2147]: [Object]
[2148]: [Object]
[2149]: [Object]
[2150]: [Object]
[2151]: [Object]
[2152]: [Object]
[2153]: [Object]
[2154]: [Object]
OrderId: 11077
orderyear: 2016
calendaryearquarter: "Q2"
ProductId: 77
ProductName: "Product LUNZZ"

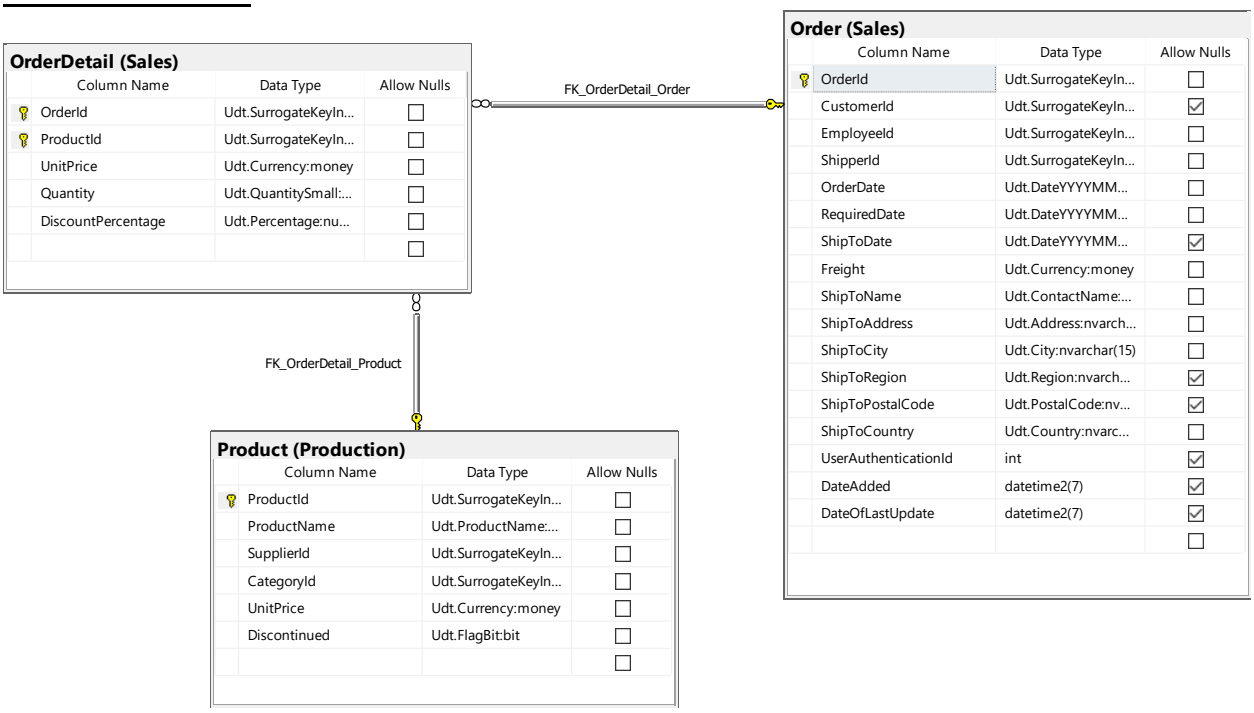
new 1


```
        "OrderId": 11077,
        "orderyear": 2016,
        "calendaryearquarter": "Q2",
        "ProductId": 46,
        "ProductName": "Product CBRRL"
    }, {
        "OrderId": 11077,
        "orderyear": 2016,
        "calendaryearquarter": "Q2",
        "ProductId": 52,
        "ProductName": "Product QSRXF"
    }, {
        "OrderId": 11077,
        "orderyear": 2016,
        "calendaryearquarter": "Q2",
        "ProductId": 55,
        "ProductName": "Product YYWRT"
    }, {
        "OrderId": 11077,
        "orderyear": 2016,
        "calendaryearquarter": "Q2",
        "ProductId": 60,
        "ProductName": "Product WHBYK"
    }, {
        "OrderId": 11077,
        "orderyear": 2016,
        "calendaryearquarter": "Q2",
        "ProductId": 64,
        "ProductName": "Product HCQDE"
    }, {
        "OrderId": 11077,
        "orderyear": 2016,
        "calendaryearquarter": "Q2",
        "ProductId": 66,
        "ProductName": "Product LQMGN"
    }, {
        "OrderId": 11077,
        "orderyear": 2016,
        "calendaryearquarter": "Q2",
        "ProductId": 73,
        "ProductName": "Product WEUJZ"
    }, {
        "OrderId": 11077,
        "orderyear": 2016,
        "calendaryearquarter": "Q2",
        "ProductId": 75,
        "ProductName": "Product BWRLG"
    }, {
        "OrderId": 11077,
        "orderyear": 2016,
        "calendaryearquarter": "Q2",
        "ProductId": 77,
        "ProductName": "Product LUNZZ"
    }
]
```

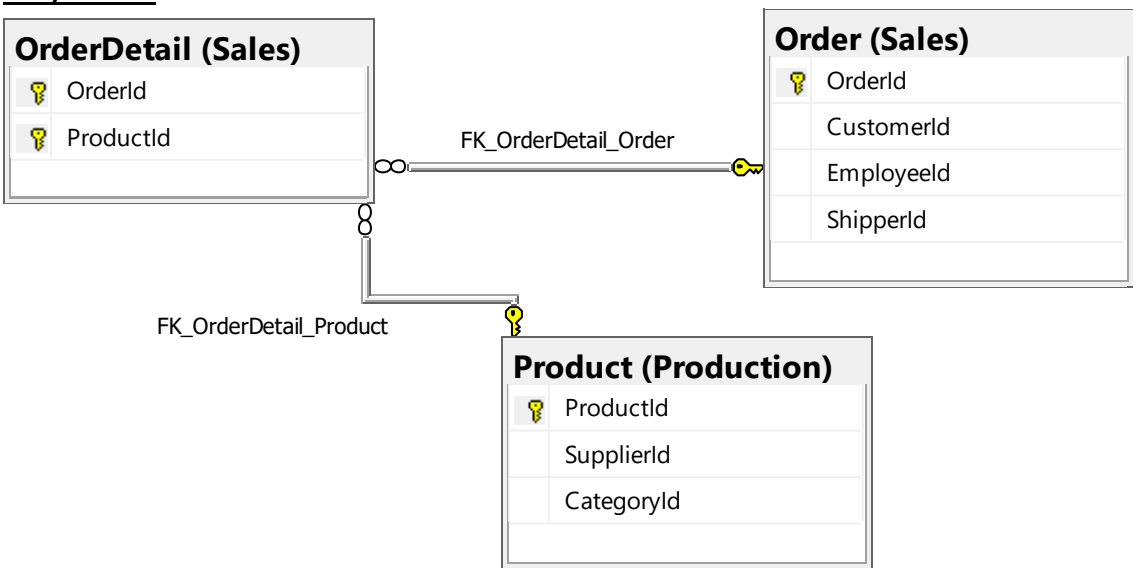
Complex Query 18:

```
--Get total unit price and name of each product from Production.Product, using the
ProductID and quantity from Sales.OrderDetail, OrderId from Sales.[Order], using
Northwinds2020TSQLV6.
```

Standard view:



Key view:



Columns from tables

Table Name	Column Names
Order	OrderId
OrderDetail	ProductId Quantity DiscountPercentage
Product	ProductName UnitPrice

Order By

Table Name	Column Name	Sort Order
Order	OrderId	ASC

Without JSON:

```
USE Northwinds2020TSQLV6;

GO
CREATE OR ALTER FUNCTION Sales.udf_TotalUnitPrice
(
    @quantity INT,
    @unitprice MONEY,
    @discountpercentage NUMERIC(4, 3)
)
RETURNS MONEY
AS
BEGIN
    RETURN (@quantity * @unitprice) * (1. - @discountpercentage);
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
       od.ProductId,
       pp.ProductName,
       Sales.udf_TotalUnitPrice(od.Quantity, pp.UnitPrice, od.DiscountPercentage) AS
totalunitprice
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.OrderId
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY o.OrderId,
         od.ProductId,
         pp.ProductName,
         od.Quantity,
         pp.UnitPrice,
         od.DiscountPercentage
ORDER BY o.OrderId;
--FOR JSON PATH, ROOT ('TotalUnitPrice'), INCLUDE_NULL_VALUES;
```

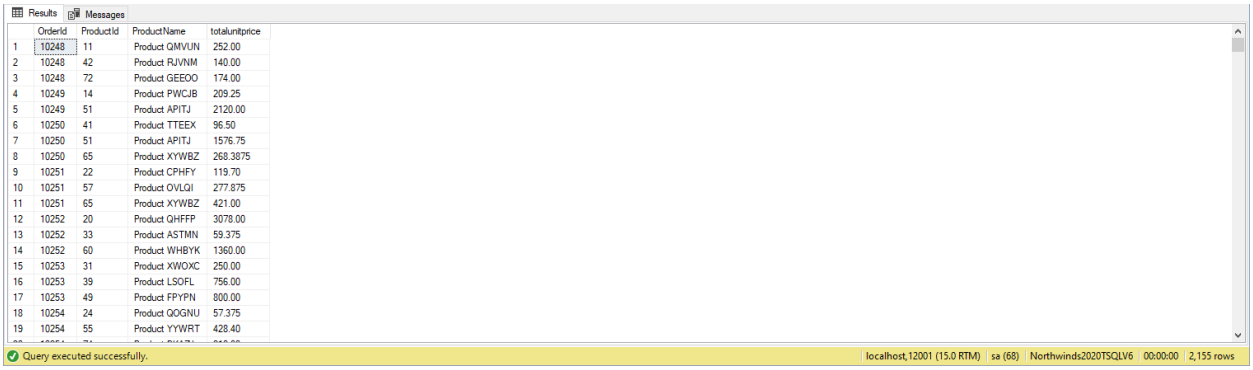
With JSON:

```
USE Northwinds2020TSQLV6;

GO
CREATE OR ALTER FUNCTION Sales.udf_TotalUnitPrice
(
    @quantity INT,
    @unitprice MONEY,
    @discountpercentage NUMERIC(4, 3)
)
RETURNS MONEY
AS
BEGIN
    RETURN (@quantity * @unitprice) * (1. - @discountpercentage);
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
       od.ProductId,
       pp.ProductName,
       Sales.udf_TotalUnitPrice(od.Quantity, pp.UnitPrice, od.DiscountPercentage) AS
totalunitprice
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.OrderId
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY o.OrderId,
         od.ProductId,
         pp.ProductName,
         od.Quantity,
         pp.UnitPrice,
         od.DiscountPercentage
ORDER BY o.OrderId
FOR JSON PATH, ROOT ('TotalUnitPrice'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (2155)

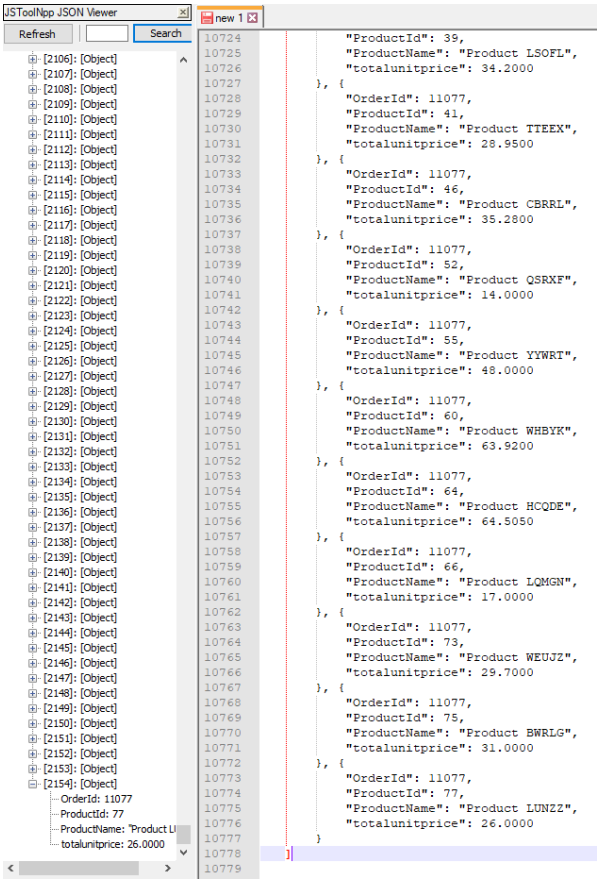


Results | Messages

	OrderId	ProductId	ProductName	totalunitprice
1	10248	11	Product QMVUN	252.00
2	10248	42	Product RJVNM	140.00
3	10248	72	Product GEEOO	174.00
4	10249	14	Product PWCJB	209.25
5	10249	51	Product APITJ	2120.00
6	10250	41	Product TTEEX	96.50
7	10250	51	Product APITJ	1576.75
8	10250	65	Product XYWBZ	268.3875
9	10251	22	Product CPHFY	119.70
10	10251	57	Product OVLQI	277.875
11	10251	65	Product XYWBZ	421.00
12	10252	20	Product QHFFP	3078.00
13	10252	33	Product ASTMN	59.375
14	10252	60	Product WHBYK	1360.00
15	10253	31	Product XWOXC	250.00
16	10253	39	Product LSOFL	756.00
17	10253	49	Product FPYPN	800.00
18	10254	24	Product QOGNU	57.375
19	10254	55	Product YYWRT	428.40

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | Northwinds2020TSQLV6 | 00:00:00 | 2,155 rows

Sample JSON Output with total number of rows returned (2155)



JSToolNpp JSON Viewer

Refresh | Search

[2106]: [Object]
[2107]: [Object]
[2108]: [Object]
[2109]: [Object]
[2110]: [Object]
[2111]: [Object]
[2112]: [Object]
[2113]: [Object]
[2114]: [Object]
[2115]: [Object]
[2116]: [Object]
[2117]: [Object]
[2118]: [Object]
[2119]: [Object]
[2120]: [Object]
[2121]: [Object]
[2122]: [Object]
[2123]: [Object]
[2124]: [Object]
[2125]: [Object]
[2126]: [Object]
[2127]: [Object]
[2128]: [Object]
[2129]: [Object]
[2130]: [Object]
[2131]: [Object]
[2132]: [Object]
[2133]: [Object]
[2134]: [Object]
[2135]: [Object]
[2136]: [Object]
[2137]: [Object]
[2138]: [Object]
[2139]: [Object]
[2140]: [Object]
[2141]: [Object]
[2142]: [Object]
[2143]: [Object]
[2144]: [Object]
[2145]: [Object]
[2146]: [Object]
[2147]: [Object]
[2148]: [Object]
[2149]: [Object]
[2150]: [Object]
[2151]: [Object]
[2152]: [Object]
[2153]: [Object]
[2154]: [Object]
OrderId: 11077
ProductId: 77
ProductName: "Product LU
totalunitprice: 26.0000

new 1

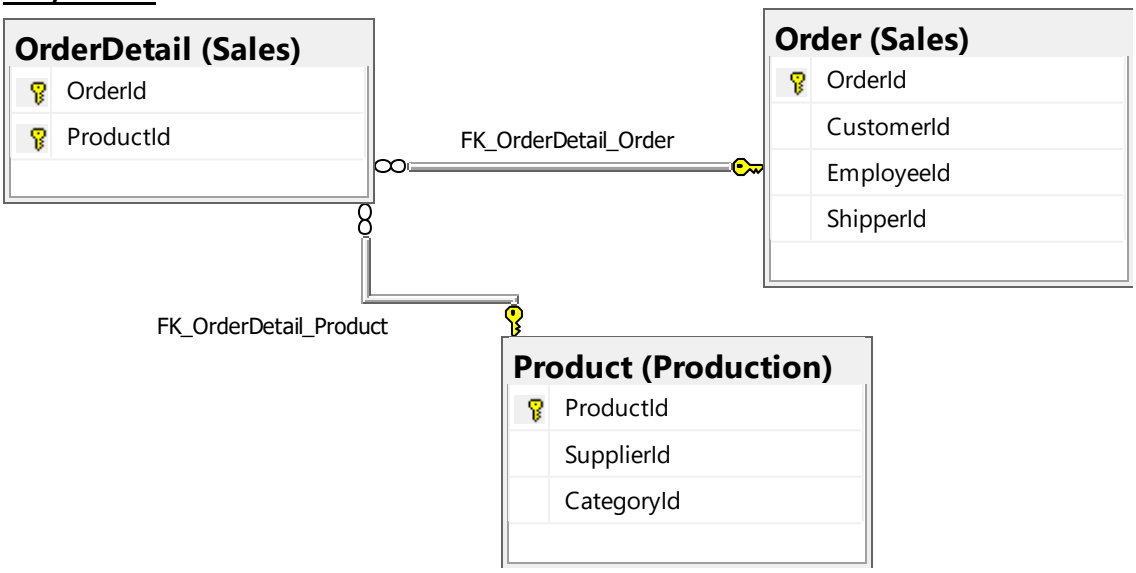

```
            "ProductId": 39,
            "ProductName": "Product LSOFL",
            "totalunitprice": 34.2000
        }, {
            "OrderId": 11077,
            "ProductId": 41,
            "ProductName": "Product TTEEX",
            "totalunitprice": 28.9500
        }, {
            "OrderId": 11077,
            "ProductId": 46,
            "ProductName": "Product CBRRL",
            "totalunitprice": 35.2800
        }, {
            "OrderId": 11077,
            "ProductId": 52,
            "ProductName": "Product QSRXF",
            "totalunitprice": 14.0000
        }, {
            "OrderId": 11077,
            "ProductId": 55,
            "ProductName": "Product YYWRT",
            "totalunitprice": 48.0000
        }, {
            "OrderId": 11077,
            "ProductId": 60,
            "ProductName": "Product WHBYK",
            "totalunitprice": 63.9200
        }, {
            "OrderId": 11077,
            "ProductId": 64,
            "ProductName": "Product HCQDE",
            "totalunitprice": 64.5050
        }, {
            "OrderId": 11077,
            "ProductId": 66,
            "ProductName": "Product LQMGN",
            "totalunitprice": 17.0000
        }, {
            "OrderId": 11077,
            "ProductId": 73,
            "ProductName": "Product WEUJZ",
            "totalunitprice": 29.7000
        }, {
            "OrderId": 11077,
            "ProductId": 75,
            "ProductName": "Product BWRLG",
            "totalunitprice": 31.0000
        }, {
            "OrderId": 11077,
            "ProductId": 77,
            "ProductName": "Product LUNZZ",
            "totalunitprice": 26.0000
        }
    }
```

Complex Query 19:

```
--Create a column to determine if an order was placed at the end of the month from the
OrderDates in Sales.[Order], show ProductName from Production.Product and ProductId from
Sales.OrderDetails using Northwinds2020TSQLV6.
```

Standard view:

Key view:



Columns from tables

Table Name	Column Names
Order	OrderId OrderDate
OrderDetail	ProductId
Product	ProductName

Order By

Table Name	Column Name	Sort Order
Order	OrderI	ASC

Without JSON:

```
USE Northwinds2020TSQLV6;

GO
CREATE OR ALTER FUNCTION Sales.udf_EomonthCheck
(
    @date DATE
)
RETURNS NVARCHAR(3)
AS
BEGIN
    DECLARE @result NVARCHAR(3);
    DECLARE @endofmonth DATE = EOMONTH(@date);
    SELECT @result = CASE
                        WHEN @endofmonth = @date THEN
                            'YES'
                        ELSE
                            'NO'
                    END;
    RETURN @result;
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
       Sales.udf_EomonthCheck(o.OrderDate) AS endofmonth,
       pp.ProductName,
       od.ProductId
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON  o.OrderId = od.OrderId
    INNER JOIN Production.Product AS pp
        ON pp.ProductId = od.ProductId
GROUP BY o.OrderId,
             o.OrderDate,
         pp.ProductName,
         od.ProductId
ORDER BY o.OrderId;
--FOR JSON PATH, ROOT ('EndOfMonth'), INCLUDE_NULL_VALUES;
```

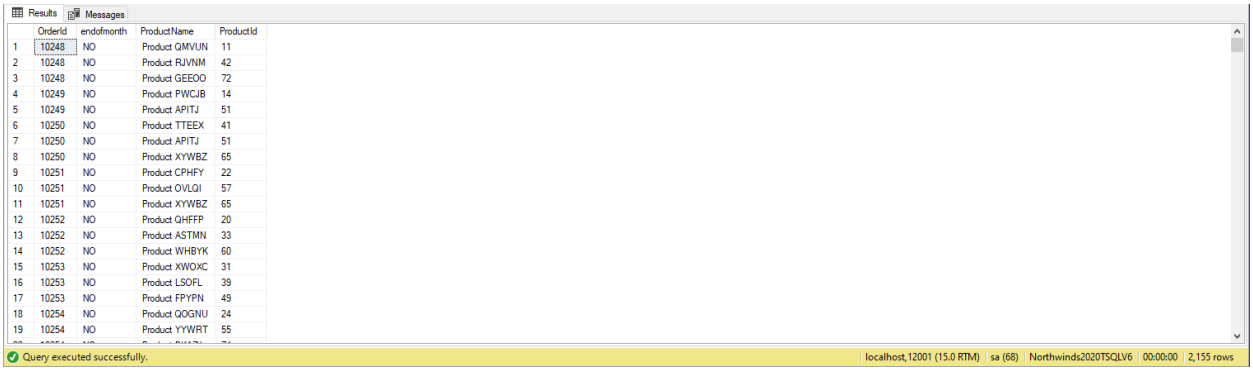
With JSON:

```
USE Northwinds2020TSQLV6;

GO
CREATE OR ALTER FUNCTION Sales.udf_EomonthCheck
(
    @date DATE
)
RETURNS NVARCHAR(3)
AS
BEGIN
    DECLARE @result NVARCHAR(3);
    DECLARE @endofmonth DATE = EOMONTH(@date);
    SELECT @result = CASE
                        WHEN @endofmonth = @date THEN
                            'YES'
                        ELSE
                            'NO'
                    END;
    RETURN @result;
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
       Sales.udf_EomonthCheck(o.OrderDate) AS endofmonth,
       pp.ProductName,
       od.ProductId
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON  o.OrderId = od.OrderId
    INNER JOIN Production.Product AS pp
        ON pp.ProductId = od.ProductId
GROUP BY o.OrderId,
              o.OrderDate,
         pp.ProductName,
         od.ProductId
ORDER BY o.OrderId
FOR JSON PATH, ROOT ('EndOfMonth'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (2155)

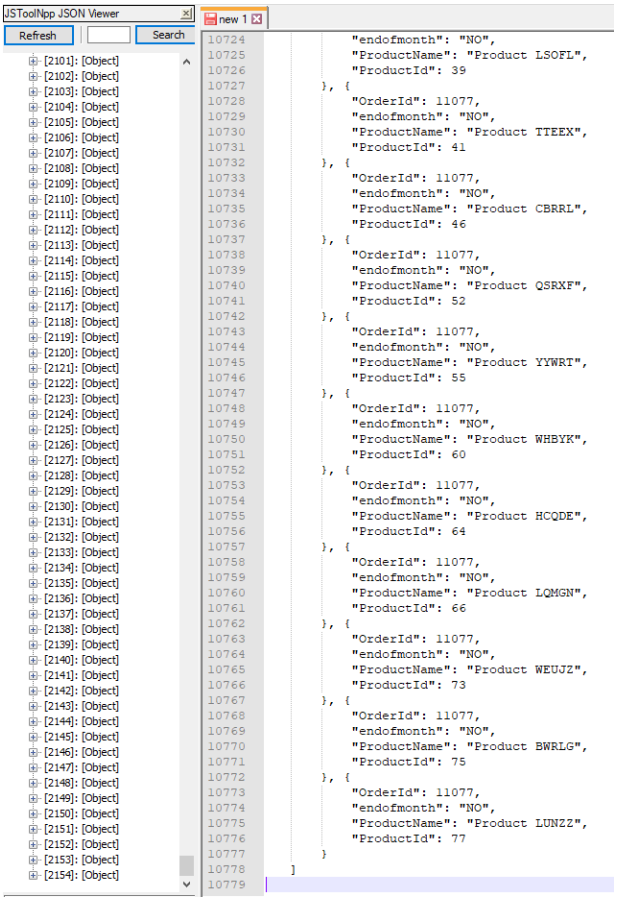


Results | Messages

	OrderId	endofmonth	ProductName	ProductId
1	10248	NO	Product QMVUN	11
2	10248	NO	Product RJVNM	42
3	10248	NO	Product GEEOO	72
4	10249	NO	Product PWCJB	14
5	10249	NO	Product APITJ	51
6	10250	NO	Product TTEEX	41
7	10250	NO	Product APITJ	51
8	10250	NO	Product XYWBZ	65
9	10251	NO	Product CPHFY	22
10	10251	NO	Product OVLQI	57
11	10251	NO	Product XYWBZ	65
12	10252	NO	Product QHFFP	20
13	10252	NO	Product ASTMN	33
14	10252	NO	Product WHBYK	60
15	10253	NO	Product XWOXC	31
16	10253	NO	Product LSOFL	39
17	10253	NO	Product FPYPN	49
18	10254	NO	Product QOGNU	24
19	10254	NO	Product YYWRT	55

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | Northwinds2020TSQLV6 | 00:00:00 | 2,155 rows

Sample JSON Output with total number of rows returned (2155)



JSToolNpp JSON Viewer

Refresh | Search

[2101]: [Object]
[2102]: [Object]
[2103]: [Object]
[2104]: [Object]
[2105]: [Object]
[2106]: [Object]
[2107]: [Object]
[2108]: [Object]
[2109]: [Object]
[2110]: [Object]
[2111]: [Object]
[2112]: [Object]
[2113]: [Object]
[2114]: [Object]
[2115]: [Object]
[2116]: [Object]
[2117]: [Object]
[2118]: [Object]
[2119]: [Object]
[2120]: [Object]
[2121]: [Object]
[2122]: [Object]
[2123]: [Object]
[2124]: [Object]
[2125]: [Object]
[2126]: [Object]
[2127]: [Object]
[2128]: [Object]
[2129]: [Object]
[2130]: [Object]
[2131]: [Object]
[2132]: [Object]
[2133]: [Object]
[2134]: [Object]
[2135]: [Object]
[2136]: [Object]
[2137]: [Object]
[2138]: [Object]
[2139]: [Object]
[2140]: [Object]
[2141]: [Object]
[2142]: [Object]
[2143]: [Object]
[2144]: [Object]
[2145]: [Object]
[2146]: [Object]
[2147]: [Object]
[2148]: [Object]
[2149]: [Object]
[2150]: [Object]
[2151]: [Object]
[2152]: [Object]
[2153]: [Object]
[2154]: [Object]

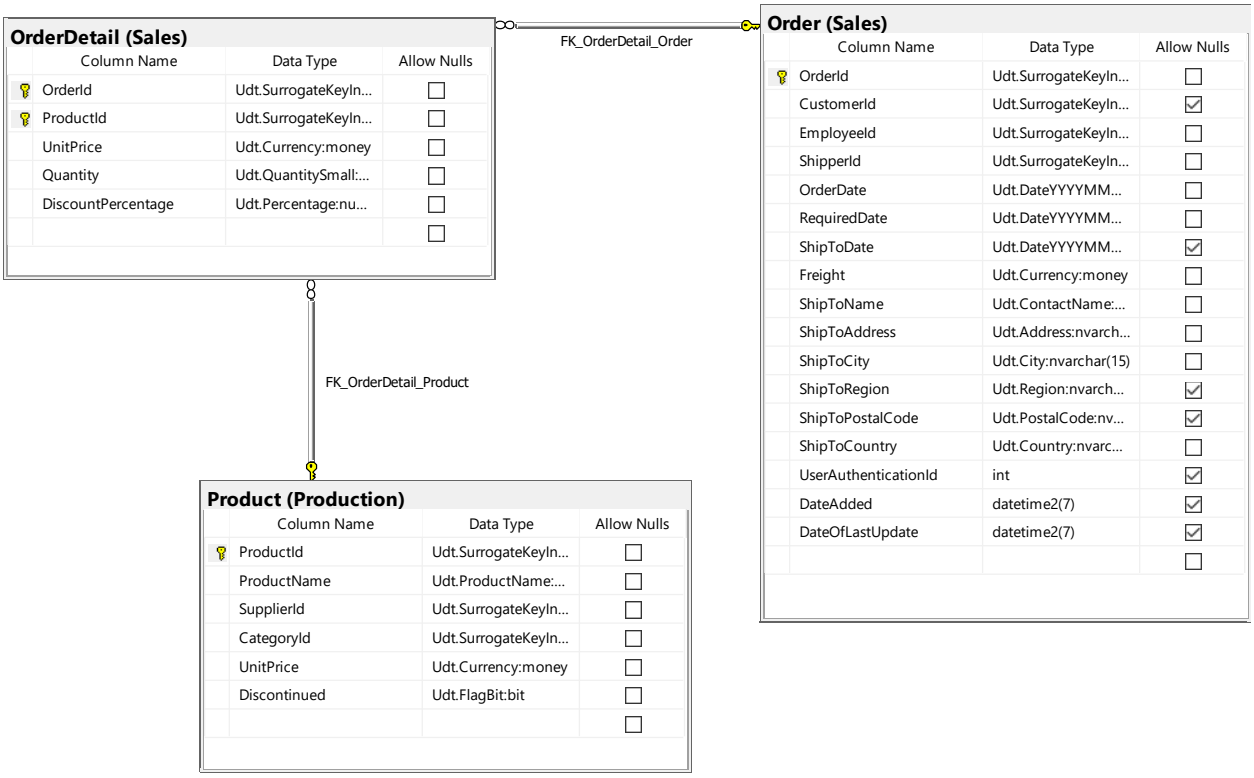
new 1


```
            "endofmonth": "NO",
            "ProductName": "Product LSOFL",
            "ProductId": 39
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product TTEEX",
            "ProductId": 41
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product CBRRL",
            "ProductId": 46
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product QSRXF",
            "ProductId": 52
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product YYWRT",
            "ProductId": 55
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product WHBYK",
            "ProductId": 60
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product HCQDE",
            "ProductId": 64
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product LQMGN",
            "ProductId": 66
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product WEUJZ",
            "ProductId": 73
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product BWRLG",
            "ProductId": 75
        }, {
            "OrderId": 11077,
            "endofmonth": "NO",
            "ProductName": "Product LUNZZ",
            "ProductId": 77
        }
    ]
```

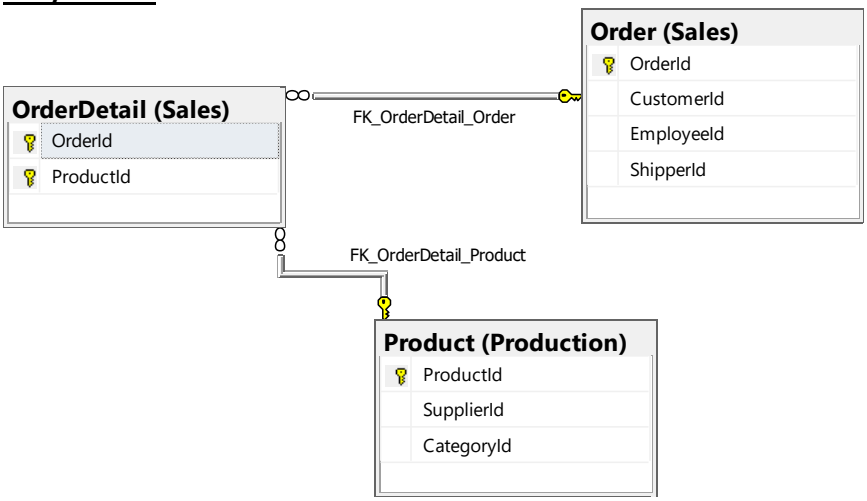
Complex Query 20:

```
--Requires scalar function TotalUnitPrice from Query 18.
--Create a scalar function that categorizes totalunitprice as cheap or expensive from
Production.Product, showing OrderID from Sales.[Order] and ProductID from
Sales.OrderDetail, using Northwinds2020TSQLV6.
```

Standard view:



Key view:



Columns from tables

Table Name	Column Names
Order	OrderId
OrderDetail	ProductId Quantity DiscountPercentage
Product	ProductName UnitPrice

Order By

Table Name	Column Name	Sort Order
Order	OrderId	ASC

Without JSON:

```
USE Northwinds2020TSQLV6;

GO

CREATE OR ALTER FUNCTION Sales.udf_PriceCheck
(
    @totalunitprice MONEY
)
RETURNS NVARCHAR(20)
AS
BEGIN
    DECLARE @result NVARCHAR(20);
    SELECT @result = CASE
                         WHEN @totalunitprice < 1 THEN
                             'Cheapest'
                         WHEN @totalunitprice > 1 THEN
                             'Expensive'
                         ELSE
                             'N/A'
                     END;
    RETURN @result;
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
       od.ProductId,
       pp.ProductName,
       Sales.udf_PriceCheck((Sales.udf_TotalUnitPrice(od.Quantity, pp.UnitPrice,
od.DiscountPercentage))) AS PriceCheck
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.OrderId
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY o.OrderId,
         od.ProductId,
         pp.ProductName,
         od.Quantity,
         pp.UnitPrice,
         od.DiscountPercentage
ORDER BY o.OrderId;
--FOR JSON PATH, ROOT ('PriceCheck'), INCLUDE_NULL_VALUES;
```

With JSON:

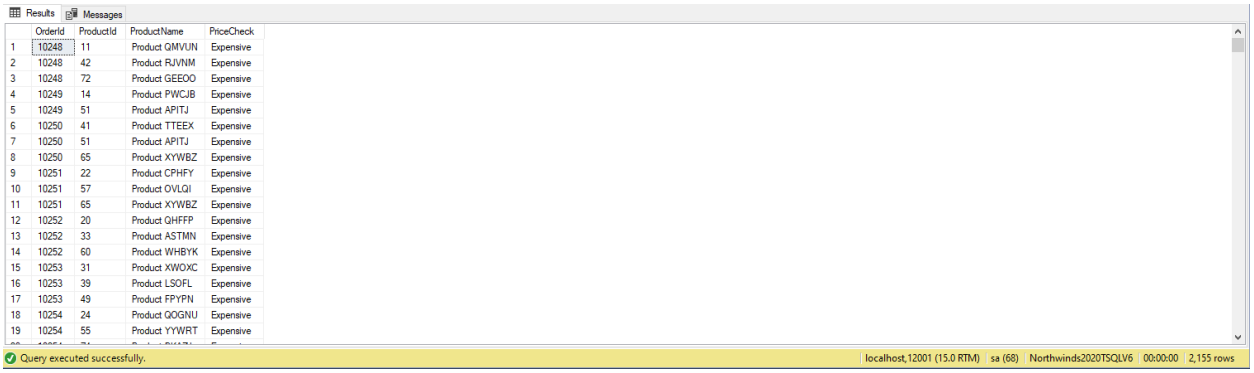
```
USE Northwinds2020TSQLV6;

GO

CREATE OR ALTER FUNCTION Sales.udf_PriceCheck
(
    @totalunitprice MONEY
)
RETURNS NVARCHAR(20)
AS
BEGIN
    DECLARE @result NVARCHAR(20);
    SELECT @result = CASE
                         WHEN @totalunitprice < 1 THEN
                             'Cheapest'
                         WHEN @totalunitprice > 1 THEN
                             'Expensive'
                         ELSE
                             'N/A'
                     END;
    RETURN @result;
END;
GO

USE Northwinds2020TSQLV6;
SELECT o.OrderId,
       od.ProductId,
       pp.ProductName,
       Sales.udf_PriceCheck((Sales.udf_TotalUnitPrice(od.Quantity, pp.UnitPrice,
od.DiscountPercentage))) AS PriceCheck
FROM Sales.[Order] AS o
    INNER JOIN Sales.OrderDetail AS od
        ON od.OrderId = o.OrderId
    INNER JOIN Production.Product AS pp
        ON od.ProductId = pp.ProductId
GROUP BY o.OrderId,
         od.ProductId,
         pp.ProductName,
         od.Quantity,
         pp.UnitPrice,
         od.DiscountPercentage
ORDER BY o.OrderId
FOR JSON PATH, ROOT ('PriceCheck'), INCLUDE_NULL_VALUES;
```

Sample Relational Output with total number of rows returned (2155)

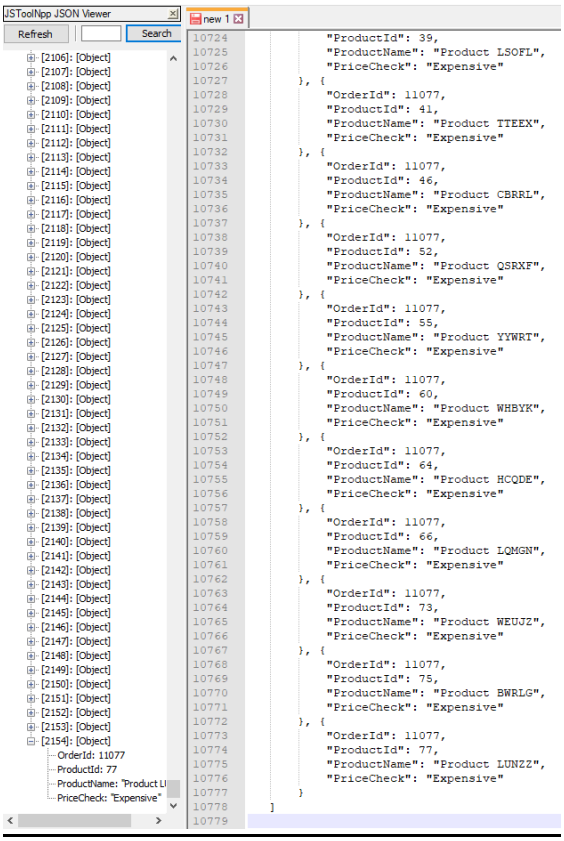


Results | Messages

	OrderId	ProductId	ProductName	PriceCheck
1	10248	11	Product QMVUN	Expensive
2	10248	42	Product RJVNM	Expensive
3	10248	72	Product GEEOO	Expensive
4	10249	14	Product PWCJB	Expensive
5	10249	51	Product APITJ	Expensive
6	10250	41	Product TTEEX	Expensive
7	10250	51	Product APITJ	Expensive
8	10250	65	Product XYWBZ	Expensive
9	10251	22	Product CPHFY	Expensive
10	10251	57	Product OVLQI	Expensive
11	10251	65	Product XYWBZ	Expensive
12	10252	20	Product QHFFP	Expensive
13	10252	33	Product ASTMN	Expensive
14	10252	60	Product WHBYK	Expensive
15	10253	31	Product XWOXC	Expensive
16	10253	39	Product LSOFL	Expensive
17	10253	49	Product FPYPN	Expensive
18	10254	24	Product QOGNU	Expensive
19	10254	55	Product YYWRT	Expensive

Query executed successfully. | localhost,12001 (15.0 RTM) | sa (68) | Northwinds2020TSQLV6 | 00:00:00 | 2,155 rows

Sample JSON Output with total number of rows returned (2155)



JSToolNpp JSON Viewer

Refresh | Search

[2106]: [Object]
[2107]: [Object]
[2108]: [Object]
[2109]: [Object]
[2110]: [Object]
[2111]: [Object]
[2112]: [Object]
[2113]: [Object]
[2114]: [Object]
[2115]: [Object]
[2116]: [Object]
[2117]: [Object]
[2118]: [Object]
[2119]: [Object]
[2120]: [Object]
[2121]: [Object]
[2122]: [Object]
[2123]: [Object]
[2124]: [Object]
[2125]: [Object]
[2126]: [Object]
[2127]: [Object]
[2128]: [Object]
[2129]: [Object]
[2130]: [Object]
[2131]: [Object]
[2132]: [Object]
[2133]: [Object]
[2134]: [Object]
[2135]: [Object]
[2136]: [Object]
[2137]: [Object]
[2138]: [Object]
[2139]: [Object]
[2140]: [Object]
[2141]: [Object]
[2142]: [Object]
[2143]: [Object]
[2144]: [Object]
[2145]: [Object]
[2146]: [Object]
[2147]: [Object]
[2148]: [Object]
[2149]: [Object]
[2150]: [Object]
[2151]: [Object]
[2152]: [Object]
[2153]: [Object]
[2154]: [Object]
OrderId: 11077
ProductId: 77
ProductName: "Product LU
PriceCheck: "Expensive"

new 1

```
            "ProductId": 39,
            "ProductName": "Product LSOFL",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 41,
            "ProductName": "Product TTEEX",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 46,
            "ProductName": "Product CBRRL",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 52,
            "ProductName": "Product QSRXF",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 55,
            "ProductName": "Product YYWRT",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 60,
            "ProductName": "Product WHBYK",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 64,
            "ProductName": "Product HCQDE",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 66,
            "ProductName": "Product LQMGN",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 73,
            "ProductName": "Product WEUJZ",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 75,
            "ProductName": "Product BWRLG",
            "PriceCheck": "Expensive"
        }, {
            "OrderId": 11077,
            "ProductId": 77,
            "ProductName": "Product LUNZZ",
            "PriceCheck": "Expensive"
        }
    ]
```