CSCI 331 PROJECT 1 – INDIVIDUAL PDF

PROFESSOR: PETER HELLER

SECTION: 9:15-10:30 AM

GROUP: G9-5

PDF OF: SHIVANIE KEMRAJ

THIS PDF CONTAINS THE 3 TOP, 3 WORST, AND WORST CORRECTED QUERIES

CONTENTS

TOP #1: A SIMPLE QUERY

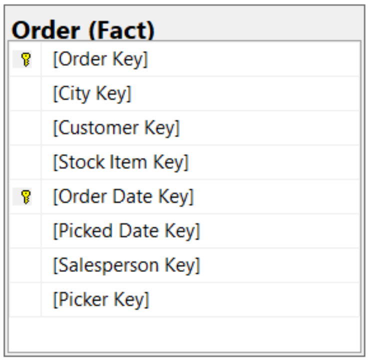
PROBLEM STATEMENT

Return order key with unit price using WideWorldImportersDW

REASON IT IS A TOP

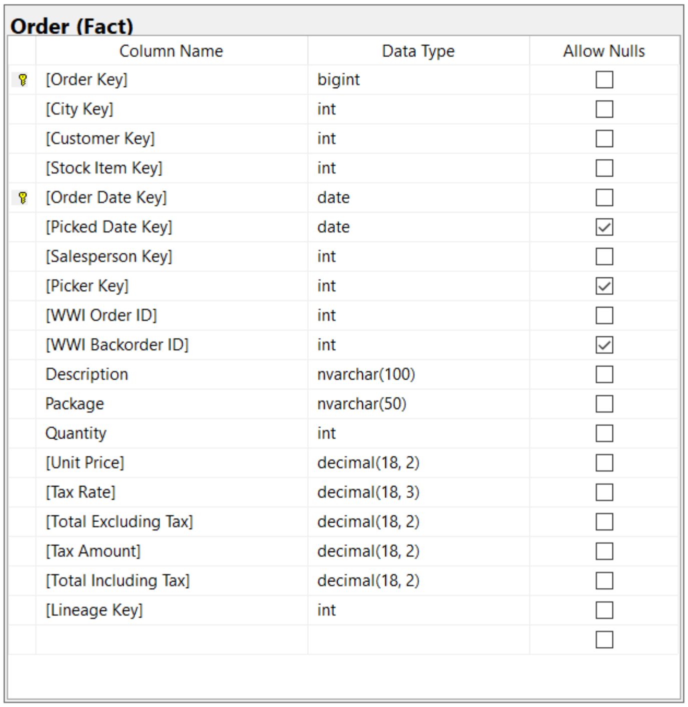
Sorted by order key

KEY AND STANDARD VIEW OF TABLES USED



Order (Fact)

- 🔑 [Order Key]
- [City Key]
- [Customer Key]
- [Stock Item Key]
- 🔑 [Order Date Key]
- [Picked Date Key]
- [Salesperson Key]
- [Picker Key]



Order (Fact)

	Column Name	Data Type	Allow Nulls
🔑	[Order Key]	bigint	☐
	[City Key]	int	☐
	[Customer Key]	int	☐
	[Stock Item Key]	int	☐
🔑	[Order Date Key]	date	☐
	[Picked Date Key]	date	☑
	[Salesperson Key]	int	☐
	[Picker Key]	int	☑
	[WWI Order ID]	int	☐
	[WWI Backorder ID]	int	☑
	Description	nvarchar(100)	☐
	Package	nvarchar(50)	☐
	Quantity	int	☐
	[Unit Price]	decimal(18, 2)	☐
	[Tax Rate]	decimal(18, 3)	☐
	[Total Excluding Tax]	decimal(18, 2)	☐
	[Tax Amount]	decimal(18, 2)	☐
	[Total Including Tax]	decimal(18, 2)	☐
	[Lineage Key]	int	☐
			☐

TABLE SHOWING COLUMNS PROJECTED IN THE END

Table Name	Column Name
Fact.Order	Order Key Unit Price

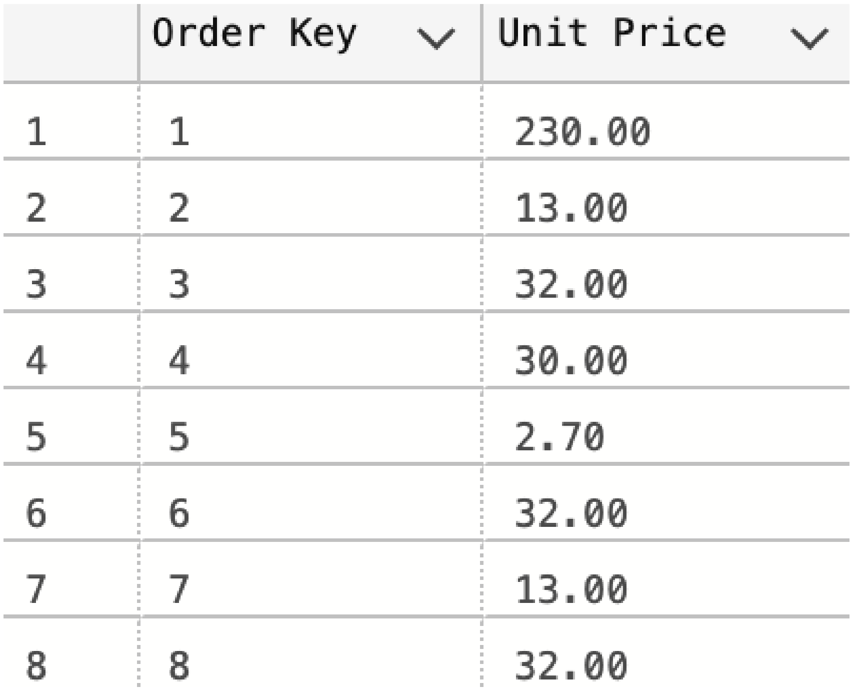
TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

Table Name	Column Name	Sort Order
Fact.Order	Order Key	ASC

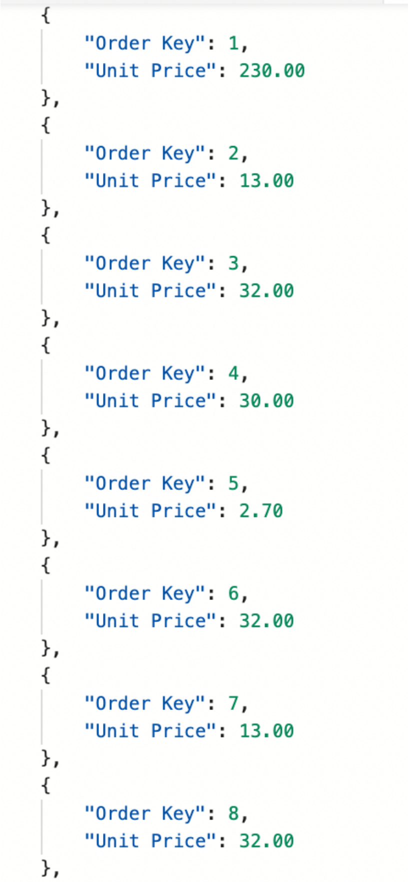
QUERY

```
USE WideWorldImportersDW;

SELECT [Order Key],
       [Unit Price]
FROM Fact.[Order]
ORDER BY [Order Key];
```

RELATIONAL AND JSON OUTPUT (231412 ROWS AFFECTED)



	Order Key	Unit Price
1	1	230.00
2	2	13.00
3	3	32.00
4	4	30.00
5	5	2.70
6	6	32.00
7	7	13.00
8	8	32.00



```
{
    "Order Key": 1,
    "Unit Price": 230.00
},
{
    "Order Key": 2,
    "Unit Price": 13.00
},
{
    "Order Key": 3,
    "Unit Price": 32.00
},
{
    "Order Key": 4,
    "Unit Price": 30.00
},
{
    "Order Key": 5,
    "Unit Price": 2.70
},
{
    "Order Key": 6,
    "Unit Price": 32.00
},
{
    "Order Key": 7,
    "Unit Price": 13.00
},
{
    "Order Key": 8,
    "Unit Price": 32.00
},
```

TOP #2: A SIMPLE QUERY

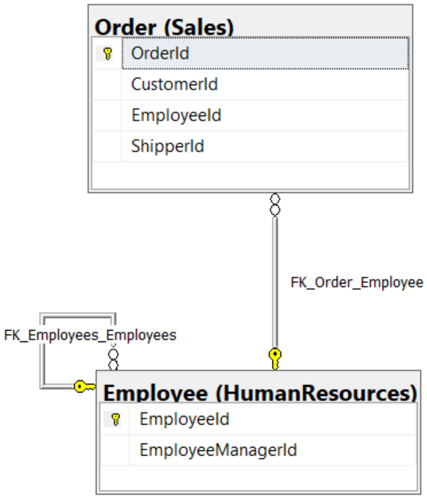
PROBLEM STATEMENT

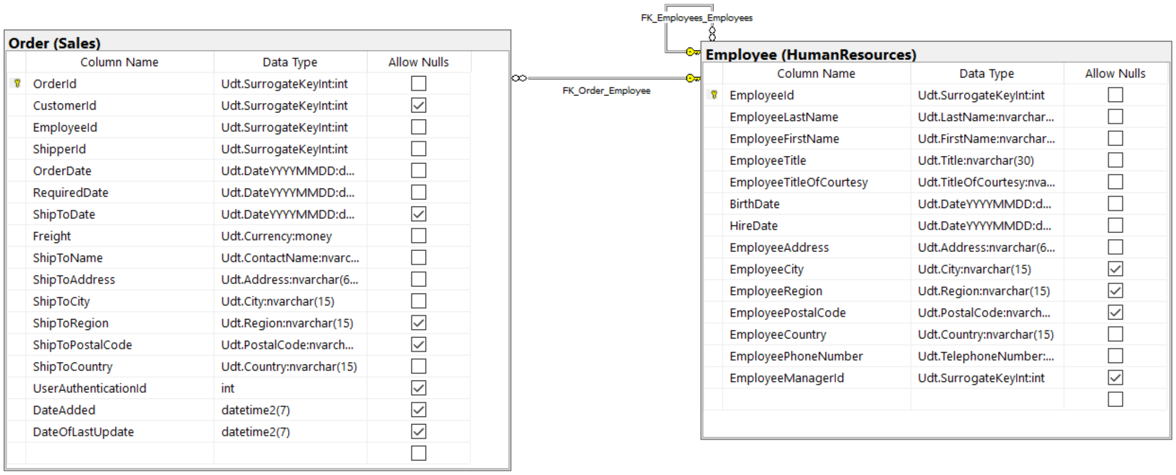
Returns concated employee's name with order id using Northwinds2022TSQLV7

REASON IT IS A TOP

Uses an inner join rather than cross join

KEY AND STANDARD VIEW OF TABLES USED





Order (Sales)

Column Name	Data Type	Allow Nulls
OrderId	Udt.SurrogateKeyInt:int	☐
CustomerId	Udt.SurrogateKeyInt:int	☑
EmployeeId	Udt.SurrogateKeyInt:int	☐
ShipperId	Udt.SurrogateKeyInt:int	☐
OrderDate	Udt.DateYYYYMMDD:d...	☐
RequiredDate	Udt.DateYYYYMMDD:d...	☐
ShipToDate	Udt.DateYYYYMMDD:d...	☑
Freight	Udt.Currency:money	☐
ShipToName	Udt.ContactName:nvarc...	☐
ShipToAddress	Udt.Address:nvarchar(6...	☐
ShipToCity	Udt.City:nvarchar(15)	☐
ShipToRegion	Udt.Region:nvarchar(15)	☑
ShipToPostalCode	Udt.PostalCode:nvarch...	☑
ShipToCountry	Udt.Country:nvarchar(15)	☐
UserAuthenticationId	int	☑
DateAdded	datetime2(7)	☑
DateOfLastUpdate	datetime2(7)	☑
		☐

Employee (HumanResources)

Column Name	Data Type	Allow Nulls
EmployeeId	Udt.SurrogateKeyInt:int	☐
EmployeeLastName	Udt.LastName:nvarchar...	☐
EmployeeFirstName	Udt.FirstName:nvarchar...	☐
EmployeeTitle	Udt.Title:nvarchar(30)	☐
EmployeeTitleOfCourtesy	Udt.TitleOfCourtesy:nva...	☐
BirthDate	Udt.DateYYYYMMDD:d...	☐
HireDate	Udt.DateYYYYMMDD:d...	☐
EmployeeAddress	Udt.Address:nvarchar(6...	☐
EmployeeCity	Udt.City:nvarchar(15)	☑
EmployeeRegion	Udt.Region:nvarchar(15)	☑
EmployeePostalCode	Udt.PostalCode:nvarch...	☑
EmployeeCountry	Udt.Country:nvarchar(15)	☐
EmployeePhoneNumber	Udt.TelephoneNumber:...	☐
EmployeeManagerId	Udt.SurrogateKeyInt:int	☑
		☐

TABLE SHOWING COLUMNS PROJECTED IN THE END

Table Name	Column Name
HumanResources.Employee	FullName (using EmployeeFirstName , EmployeeLastName)
Sales.Order	OrderID

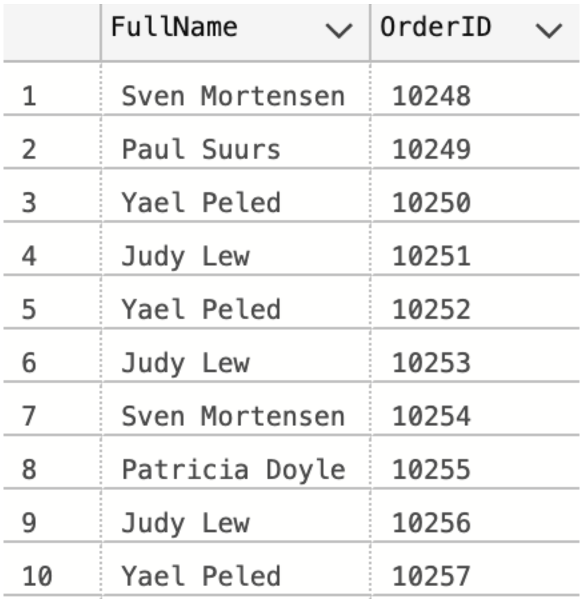
TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

Not applicable

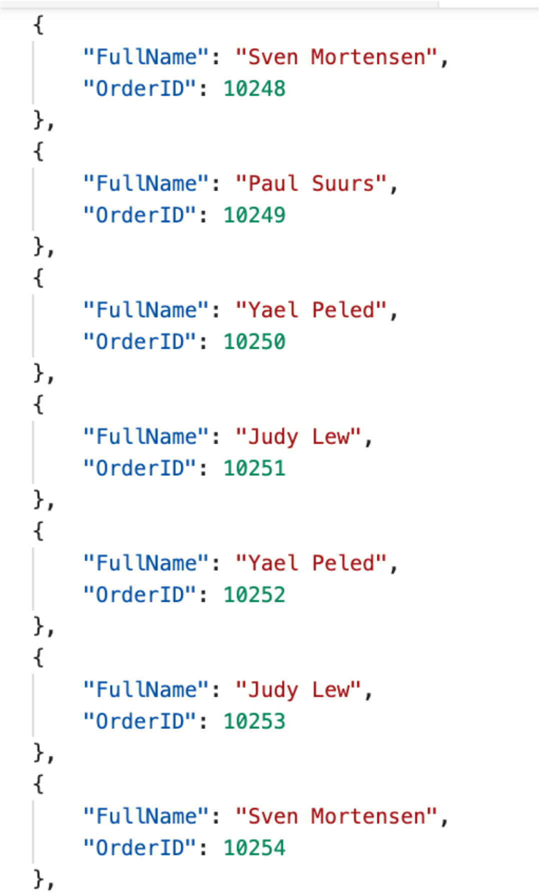
QUERY

```
USE Northwinds2022TSQLV7;

SELECT CONCAT(E.EmployeeFirstName, ' ', E.EmployeeLastName) AS FullName,
       O.OrderId
FROM HumanResources.Employee AS E
    INNER JOIN Sales.[Order] AS O
        ON E.EmployeeId =
        (
            SELECT O.EmployeeId
        )
GROUP BY O.OrderId,
         E.EmployeeFirstName,
         E.EmployeeLastName
```

RELATIONAL AND JSON OUTPUT (831 ROWS AFFECTED)



	FullName	OrderID
1	Sven Mortensen	10248
2	Paul Suurs	10249
3	Yael Peled	10250
4	Judy Lew	10251
5	Yael Peled	10252
6	Judy Lew	10253
7	Sven Mortensen	10254
8	Patricia Doyle	10255
9	Judy Lew	10256
10	Yael Peled	10257



```
{
    "FullName": "Sven Mortensen",
    "OrderID": 10248
},
{
    "FullName": "Paul Suurs",
    "OrderID": 10249
},
{
    "FullName": "Yael Peled",
    "OrderID": 10250
},
{
    "FullName": "Judy Lew",
    "OrderID": 10251
},
{
    "FullName": "Yael Peled",
    "OrderID": 10252
},
{
    "FullName": "Judy Lew",
    "OrderID": 10253
},
{
    "FullName": "Sven Mortensen",
    "OrderID": 10254
},
```

TOP #3: A COMPLEX QUERY

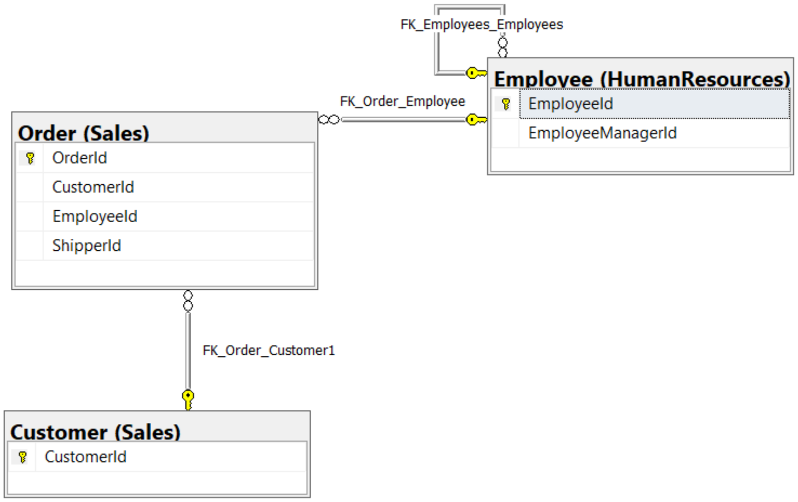
PROBLEM STATEMENT

Return custid, employed, ordered, and the season of the orders using Northwinds2022TSQLV7

REASON IT IS A TOP

Only scalar function that does something a built-in function could not

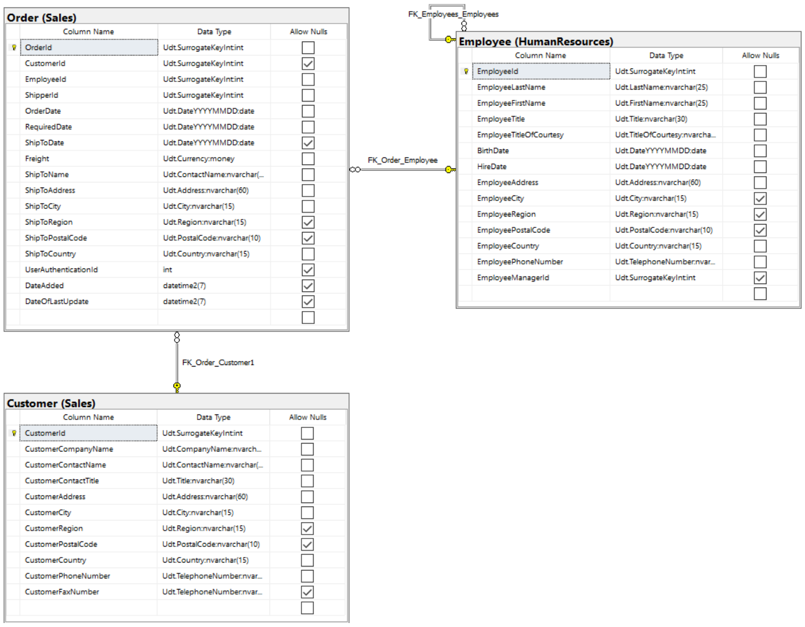
KEY AND STANDARD VIEW OF TABLES USED



Order (Sales)
OrderId
CustomerId
EmployeeId
ShipperId

Employee (HumanResources)
EmployeeId
EmployeeManagerId

Customer (Sales)
CustomerId



Order (Sales)

Column Name	Data Type	Allow Nulls
OrderId	Udt.SurrogateKeyInt:int	☐
CustomerId	Udt.SurrogateKeyInt:int	☑
EmployeeId	Udt.SurrogateKeyInt:int	☐
ShipperId	Udt.SurrogateKeyInt:int	☐
OrderDate	Udt.DateYYYYMMDD:date	☐
RequiredDate	Udt.DateYYYYMMDD:date	☐
ShipToDate	Udt.DateYYYYMMDD:date	☑
Freight	Udt.Currency:money	☐
ShipToName	Udt.ContactName:nvarchar(...	☐
ShipToAddress	Udt.Address:nvarchar(60)	☐
ShipToCity	Udt.City:nvarchar(15)	☐
ShipToRegion	Udt.Region:nvarchar(15)	☑
ShipToPostalCode	Udt.PostalCode:nvarchar(10)	☑
ShipToCountry	Udt.Country:nvarchar(15)	☐
UserAuthenticationId	int	☑
DateAdded	datetime2(7)	☑
DateOfLastUpdate	datetime2(7)	☑
		☐

Employee (HumanResources)

Column Name	Data Type	Allow Nulls
EmployeeId	Udt.SurrogateKeyInt:int	☐
EmployeeLastName	Udt.LastName:nvarchar(25)	☐
EmployeeFirstName	Udt.FirstName:nvarchar(25)	☐
EmployeeTitle	Udt.Title:nvarchar(30)	☐
EmployeeTitleOfCourtesy	Udt.TitleOfCourtesy:nvarcha...	☐
BirthDate	Udt.DateYYYYMMDD:date	☐
HireDate	Udt.DateYYYYMMDD:date	☐
EmployeeAddress	Udt.Address:nvarchar(60)	☐
EmployeeCity	Udt.City:nvarchar(15)	☑
EmployeeRegion	Udt.Region:nvarchar(15)	☑
EmployeePostalCode	Udt.PostalCode:nvarchar(10)	☑
EmployeeCountry	Udt.Country:nvarchar(15)	☐
EmployeePhoneNumber	Udt.TelephoneNumber:nvar...	☐
EmployeeManagerId	Udt.SurrogateKeyInt:int	☑
		☐

Customer (Sales)

Column Name	Data Type	Allow Nulls
CustomerId	Udt.SurrogateKeyInt:int	☐
CustomerCompanyName	Udt.CompanyName:nvarch...	☐
CustomerContactName	Udt.ContactName:nvarchar(...	☐
CustomerContactTitle	Udt.Title:nvarchar(30)	☐
CustomerAddress	Udt.Address:nvarchar(60)	☐
CustomerCity	Udt.City:nvarchar(15)	☐
CustomerRegion	Udt.Region:nvarchar(15)	☑
CustomerPostalCode	Udt.PostalCode:nvarchar(10)	☑
CustomerCountry	Udt.Country:nvarchar(15)	☐
CustomerPhoneNumber	Udt.TelephoneNumber:nvar...	☐
CustomerFaxNumber	Udt.TelephoneNumber:nvar...	☑
		☐

TABLE SHOWING COLUMNS PROJECTED IN THE END

Table Name	Column Name
Sales.Customer	CustomerId
HumanResources.Employee	EmployeeId
Sales.Order	OrderId Order Season (using OrderDate)

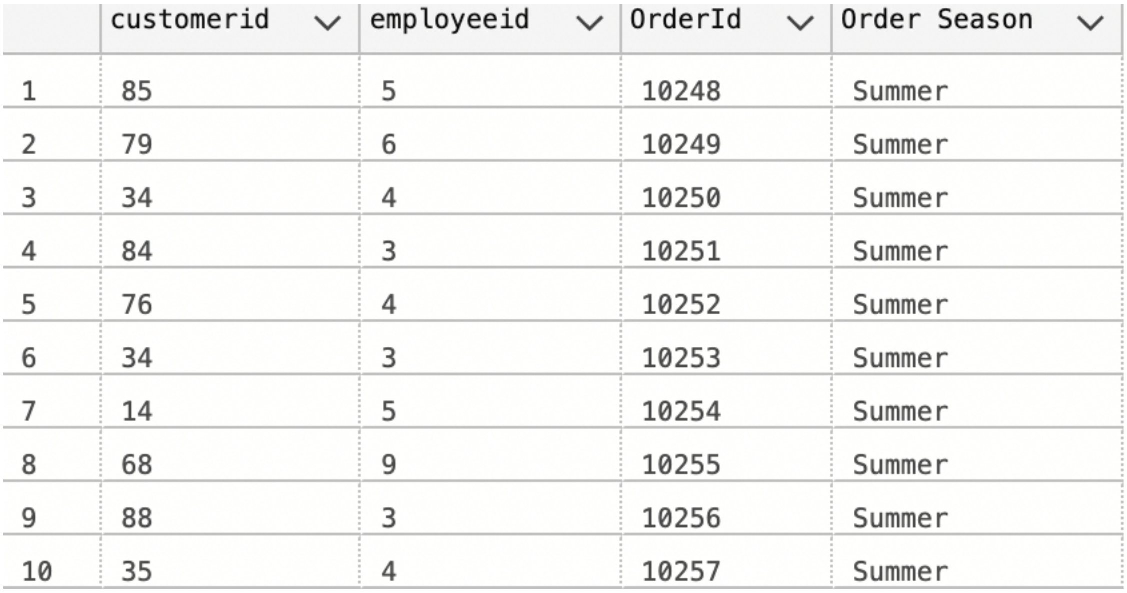
TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

Not applicable

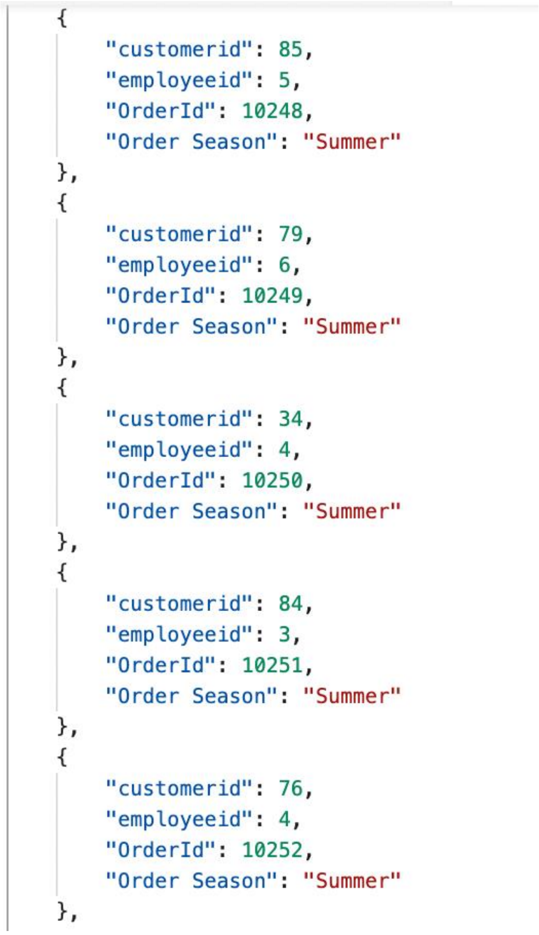
QUERY

```sql
USE Northwinds2022TSQLV7;

DROP FUNCTION IF EXISTS Sales.Season;
GO
CREATE FUNCTION Sales.Season
(
    @date DATE
)
RETURNS NVARCHAR(20)
AS
BEGIN
    DECLARE @Result NVARCHAR(20);
    DECLARE @Month INT = MONTH(@date);

    SELECT @Result = CASE
                         WHEN @Month IN ( 12, 1, 2 ) THEN
                             'Winter'
                         WHEN @Month
                              BETWEEN 3 AND 5 THEN
                             'Spring'
                         WHEN @Month
                              BETWEEN 6 AND 8 THEN
                             'Summer'
                         WHEN @Month
                              BETWEEN 9 AND 11 THEN
                             'Autumn'
                         ELSE
                             'ERROR - CANNOT CALCULATE Season'
                     END;

    RETURN @Result;
END;
GO
SELECT C.CustomerId,
       E.EmployeeId,
       O.OrderId,
       Sales.Season(O.OrderDate) AS [Order Season]
FROM Sales.Customer AS C
    CROSS JOIN HumanResources.Employee AS E
    CROSS JOIN Sales.[Order] AS O
WHERE C.CustomerId =
(
    SELECT O.CustomerId WHERE E.EmployeeId = O.EmployeeId
)
GROUP BY C.CustomerId,
         E.EmployeeId,
         O.OrderId,
         Sales.Season(O.OrderDate);
```

RELATIONAL AND JSON OUTPUT (830 ROWS AFFECTED)



	customerid	employeeid	OrderId	Order Season
1	85	5	10248	Summer
2	79	6	10249	Summer
3	34	4	10250	Summer
4	84	3	10251	Summer
5	76	4	10252	Summer
6	34	3	10253	Summer
7	14	5	10254	Summer
8	68	9	10255	Summer
9	88	3	10256	Summer
10	35	4	10257	Summer



```
    {
        "customerid": 85,
        "employeeid": 5,
        "OrderId": 10248,
        "Order Season": "Summer"
    },
    {
        "customerid": 79,
        "employeeid": 6,
        "OrderId": 10249,
        "Order Season": "Summer"
    },
    {
        "customerid": 34,
        "employeeid": 4,
        "OrderId": 10250,
        "Order Season": "Summer"
    },
    {
        "customerid": 84,
        "employeeid": 3,
        "OrderId": 10251,
        "Order Season": "Summer"
    },
    {
        "customerid": 76,
        "employeeid": 4,
        "OrderId": 10252,
        "Order Season": "Summer"
    },
```

WORST #1: A SIMPLE QUERY

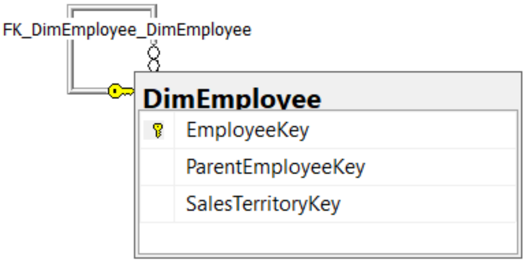
PROBLEM STATEMENT

Return employee’s full name with their birthday using AdventureWorksDW2017

REASON IT IS A WORST

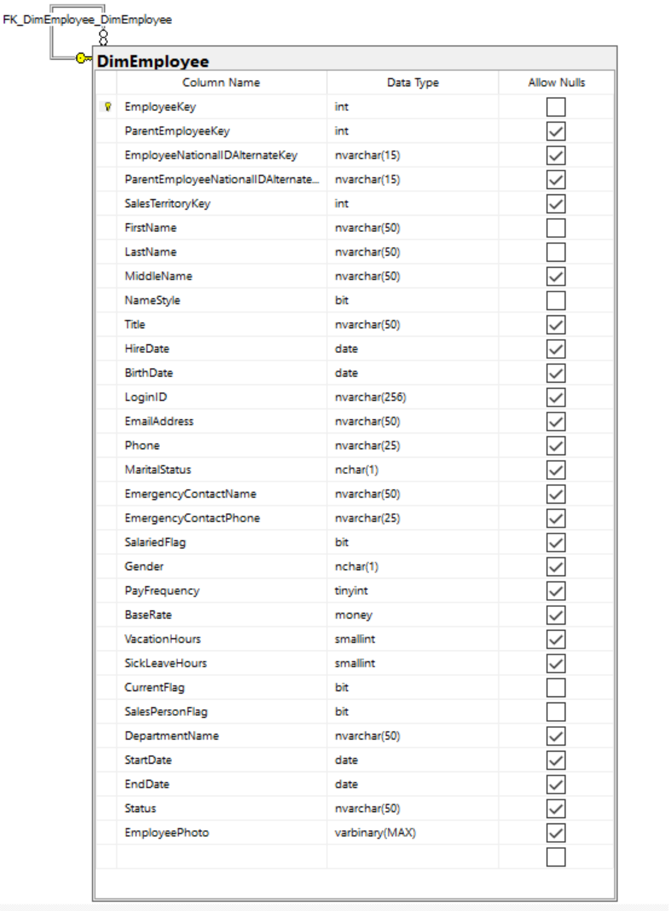
Can utilize concat function

KEY AND STANDARD VIEW OF TABLES USED



DimEmployee

- EmployeeKey
- ParentEmployeeKey
- SalesTerritoryKey



DimEmployee

Column Name	Data Type	Allow Nulls
EmployeeKey	int	☐
ParentEmployeeKey	int	☑
EmployeeNationalIDAlternateKey	nvarchar(15)	☑
ParentEmployeeNationalIDAlternate...	nvarchar(15)	☑
SalesTerritoryKey	int	☑
FirstName	nvarchar(50)	☐
LastName	nvarchar(50)	☐
MiddleName	nvarchar(50)	☑
NameStyle	bit	☐
Title	nvarchar(50)	☑
HireDate	date	☑
BirthDate	date	☑
LoginID	nvarchar(256)	☑
EmailAddress	nvarchar(50)	☑
Phone	nvarchar(25)	☑
MaritalStatus	nchar(1)	☑
EmergencyContactName	nvarchar(50)	☑
EmergencyContactPhone	nvarchar(25)	☑
SalariedFlag	bit	☑
Gender	nchar(1)	☑
PayFrequency	tinyint	☑
BaseRate	money	☑
VacationHours	smallint	☑
SickLeaveHours	smallint	☑
CurrentFlag	bit	☐
SalesPersonFlag	bit	☐
DepartmentName	nvarchar(50)	☑
StartDate	date	☑
EndDate	date	☑
Status	nvarchar(50)	☑
EmployeePhoto	varbinary(MAX)	☑
		☐

TABLE SHOWING COLUMNS PROJECTED IN THE END

Table Name	Column Name
dbo.DimEmployee	FullName (using FirstName, LastName) BirthDate

TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

Not applicable

QUERY OF WORST

```sql
USE AdventureWorksDW2017;

SELECT FirstName + ' ' + LastName AS FullName,
       BirthDate
FROM dbo.DimEmployee;
```

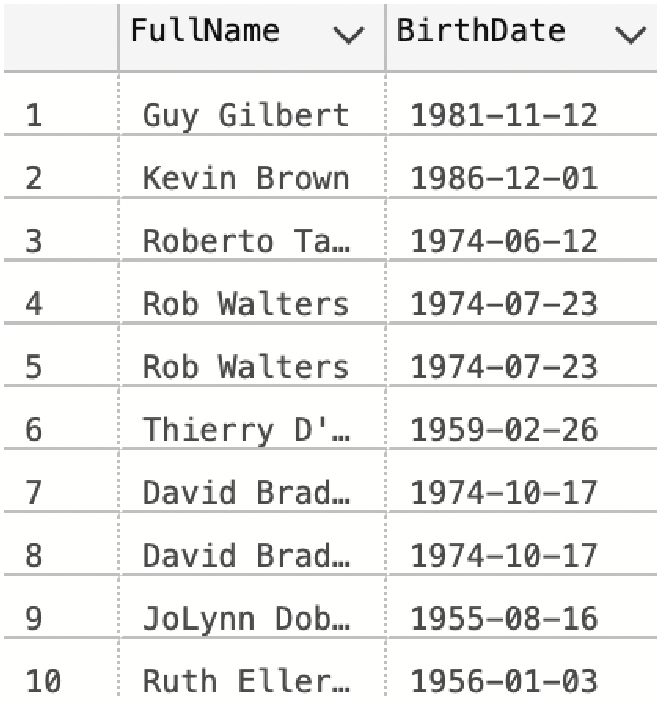
QUERY OF WORST CORRECTED

```sql
USE AdventureWorksDW2017;

SELECT CONCAT(FirstName, ' ', LastName) AS FullName,
       BirthDate
FROM dbo.DimEmployee;
```

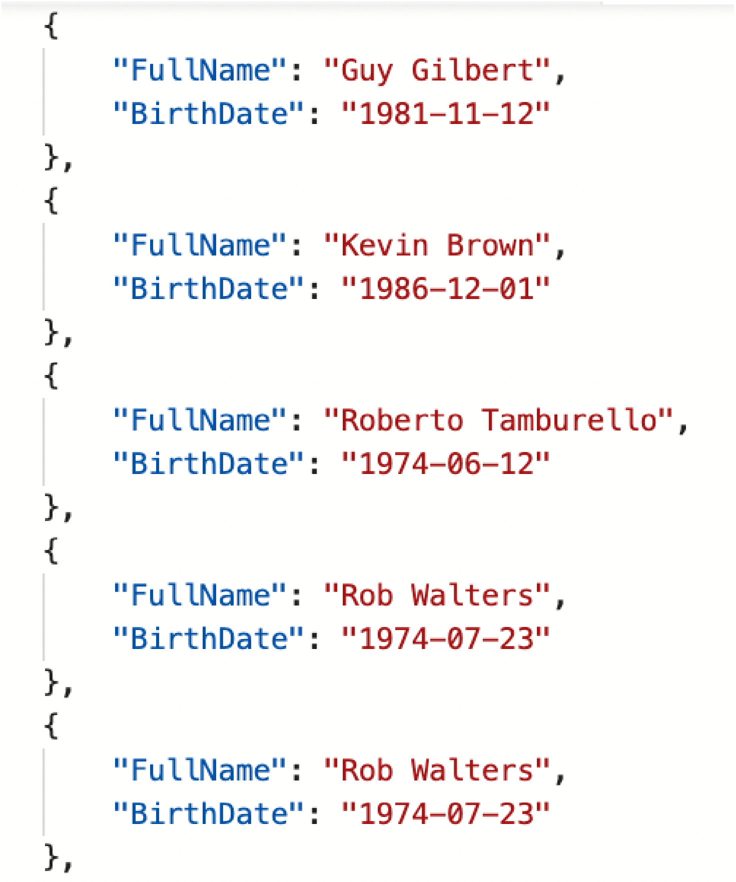
HOW IT WAS CORRECTED:

Used a concat function for better readability

RELATIONAL AND JSON OUTPUT (296 ROWS AFFECTED)



	FullName	BirthDate
1	Guy Gilbert	1981-11-12
2	Kevin Brown	1986-12-01
3	Roberto Ta…	1974-06-12
4	Rob Walters	1974-07-23
5	Rob Walters	1974-07-23
6	Thierry D'…	1959-02-26
7	David Brad…	1974-10-17
8	David Brad…	1974-10-17
9	JoLynn Dob…	1955-08-16
10	Ruth Eller…	1956-01-03



```
{
    "FullName": "Guy Gilbert",
    "BirthDate": "1981-11-12"
},
{
    "FullName": "Kevin Brown",
    "BirthDate": "1986-12-01"
},
{
    "FullName": "Roberto Tamburello",
    "BirthDate": "1974-06-12"
},
{
    "FullName": "Rob Walters",
    "BirthDate": "1974-07-23"
},
{
    "FullName": "Rob Walters",
    "BirthDate": "1974-07-23"
},
```

WORST #2: A MEDIUM QUERY

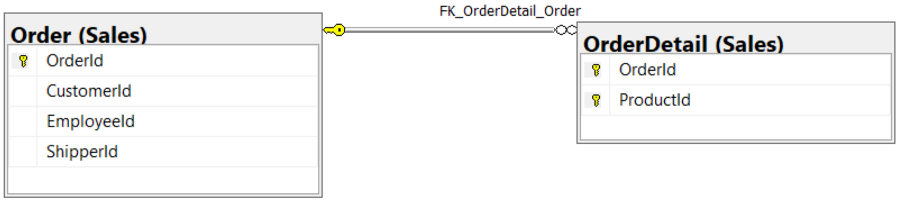
PROBLEM STATEMENT

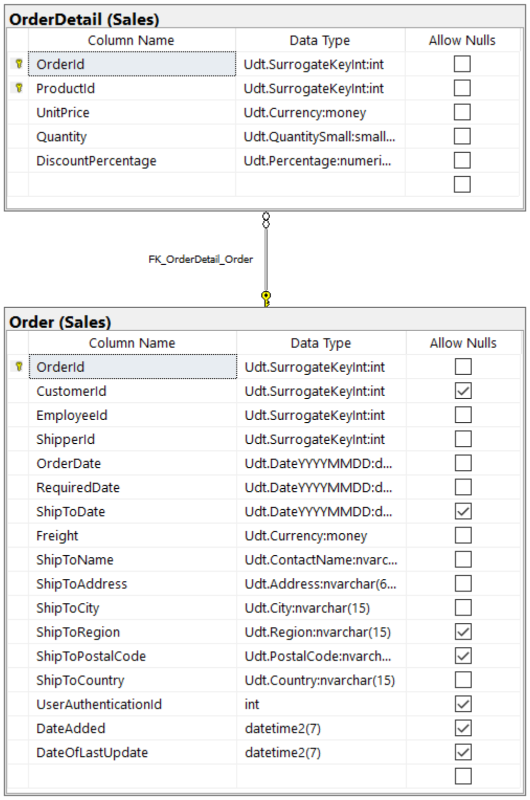
Return custid, emplid, and order year using Northwinds2022TSQLV7

REASON IT IS A WORST

Can use built-in function for full name, also can use inner joins to replace the cross join and subquery

KEY AND STANDARD VIEW OF TABLES USED





OrderDetail (Sales)

	Column Name	Data Type	Allow Nulls
🔑	OrderId	Udt.SurrogateKeyInt:int	☐
🔑	ProductId	Udt.SurrogateKeyInt:int	☐
	UnitPrice	Udt.Currency:money	☐
	Quantity	Udt.QuantitySmall:small...	☐
	DiscountPercentage	Udt.Percentage:numeri...	☐
			☐

Order (Sales)

	Column Name	Data Type	Allow Nulls
🔑	OrderId	Udt.SurrogateKeyInt:int	☐
	CustomerId	Udt.SurrogateKeyInt:int	☑
	EmployeeId	Udt.SurrogateKeyInt:int	☐
	ShipperId	Udt.SurrogateKeyInt:int	☐
	OrderDate	Udt.DateYYYYMMDD:d...	☐
	RequiredDate	Udt.DateYYYYMMDD:d...	☐
	ShipToDate	Udt.DateYYYYMMDD:d...	☑
	Freight	Udt.Currency:money	☐
	ShipToName	Udt.ContactName:nvarc...	☐
	ShipToAddress	Udt.Address:nvarchar(6...	☐
	ShipToCity	Udt.City:nvarchar(15)	☐
	ShipToRegion	Udt.Region:nvarchar(15)	☑
	ShipToPostalCode	Udt.PostalCode:nvarch...	☑
	ShipToCountry	Udt.Country:nvarchar(15)	☐
	UserAuthenticationId	int	☑
	DateAdded	datetime2(7)	☑
	DateOfLastUpdate	datetime2(7)	☑
			☐

TABLE SHOWING COLUMNS PROJECTED IN THE END

Table Name	Column Name
Sales.Order	EmployeeID
Sales.uvw_CustomerOrder	CustomerID Year (using OrderMonth)

TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

Table Name	Column Name	Sort Order
Sales.uvw_CustomerOrder	CustomerID	ASC
Sales.Order	EmployeeId	ASC

QUERY OF WORST

```sql
USE Northwinds2022TSQLV7;

SELECT CO.CustomerId,
       O.EmployeeId,
       YEAR(CO.OrderMonth) AS [Year]
FROM Sales.[Order] AS O
    CROSS JOIN Sales.uvw_CustomerOrder AS CO
WHERE CO.CustomerId =
(
    SELECT O.CustomerId
)
GROUP BY CO.CustomerId,
         O.EmployeeId,
         YEAR(CO.OrderMonth)
ORDER BY CO.CustomerId,
         O.EmployeeId;
```

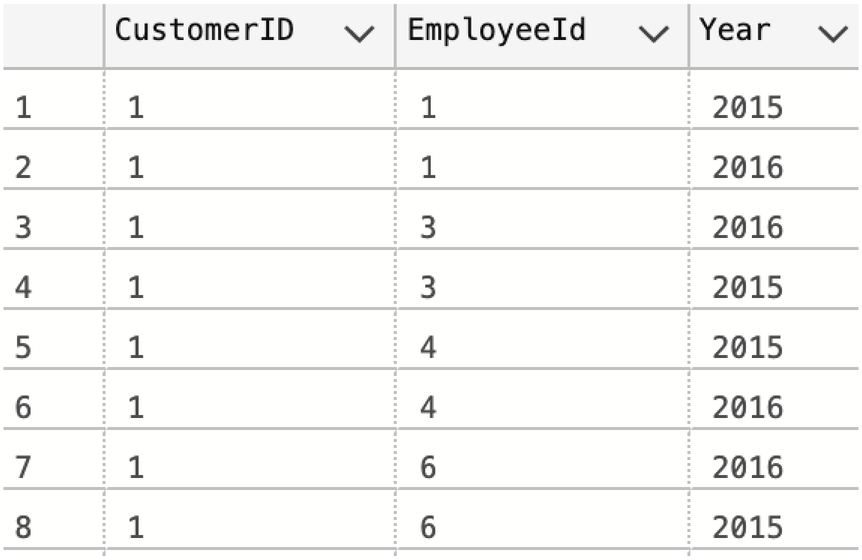
QUERY OF WORST CORRECTED

```
USE Northwinds2022TSQLV7;

SELECT CO.CustomerId,
       O.EmployeeId,
       YEAR(CO.OrderMonth) AS [Year]
FROM Sales.[Order] AS O
    INNER JOIN Sales.uvw_CustomerOrder AS CO
        ON CO.CustomerId = O.CustomerId
GROUP BY CO.CustomerId,
         O.EmployeeId,
         YEAR(CO.OrderMonth)
ORDER BY CO.CustomerId,
         O.EmployeeId;
```

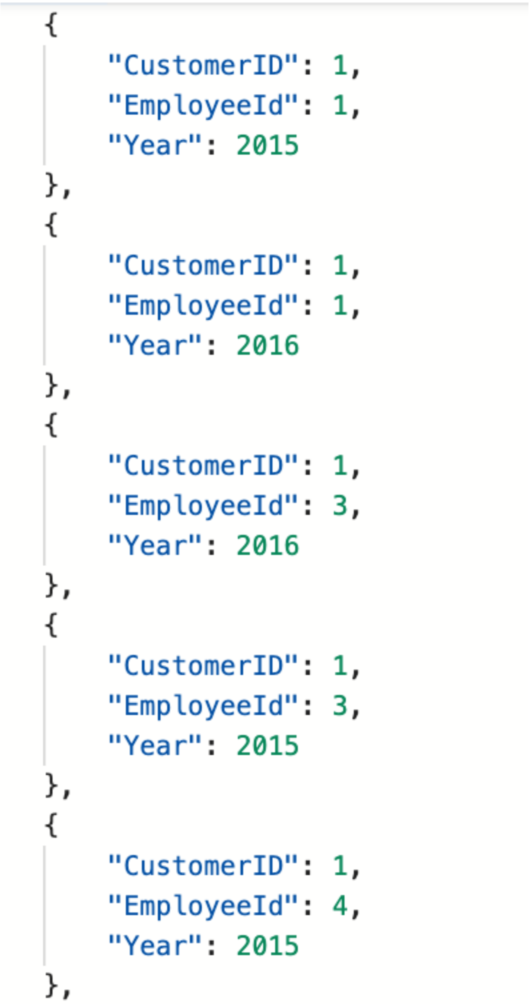
HOW IT WAS CORRECTED:

Used an inner join to replace cross join which now does no need a subquery

RELATIONAL AND JSON OUTPUT (1267 ROWS AFFECTED)



	CustomerID	EmployeeId	Year
1	1	1	2015
2	1	1	2016
3	1	3	2016
4	1	3	2015
5	1	4	2015
6	1	4	2016
7	1	6	2016
8	1	6	2015



```
{
    "CustomerID": 1,
    "EmployeeId": 1,
    "Year": 2015
},
{
    "CustomerID": 1,
    "EmployeeId": 1,
    "Year": 2016
},
{
    "CustomerID": 1,
    "EmployeeId": 3,
    "Year": 2016
},
{
    "CustomerID": 1,
    "EmployeeId": 3,
    "Year": 2015
},
{
    "CustomerID": 1,
    "EmployeeId": 4,
    "Year": 2015
},
```

WORST #3: A COMPLEX QUERY

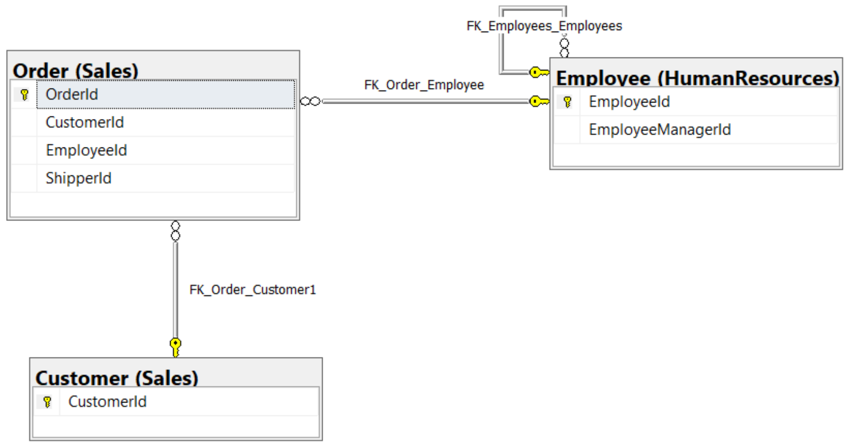
PROBLEM STATEMENT

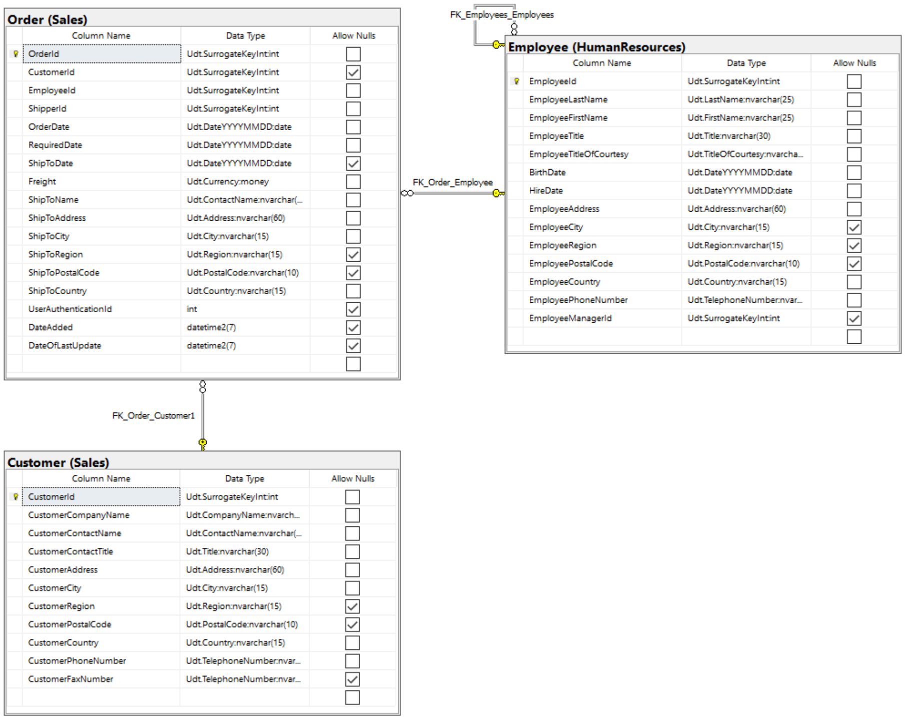
Concats the employee's first and last name to make full name; returns the ordered, custid, emplid, and name using Northwinds2022TSQLV7

REASON IT IS A WORST

Can use a built-in function for full name, also can use inner joins to replace the cross join and subquery

KEY AND STANDARD VIEW OF TABLES USED





Order (Sales)

Column Name	Data Type	Allow Nulls
OrderId	Udt.SurrogateKeyInt:int	☐
CustomerId	Udt.SurrogateKeyInt:int	☑
EmployeeId	Udt.SurrogateKeyInt:int	☐
ShipperId	Udt.SurrogateKeyInt:int	☐
OrderDate	Udt.DateYYYYMMDD:date	☐
RequiredDate	Udt.DateYYYYMMDD:date	☐
ShipToDate	Udt.DateYYYYMMDD:date	☑
Freight	Udt.Currency:money	☐
ShipToName	Udt.ContactName:nvarchar(...	☐
ShipToAddress	Udt.Address:nvarchar(60)	☐
ShipToCity	Udt.City:nvarchar(15)	☐
ShipToRegion	Udt.Region:nvarchar(15)	☑
ShipToPostalCode	Udt.PostalCode:nvarchar(10)	☑
ShipToCountry	Udt.Country:nvarchar(15)	☐
UserAuthenticationId	int	☑
DateAdded	datetime2(7)	☑
DateOfLastUpdate	datetime2(7)	☑
		☐

Employee (HumanResources)

Column Name	Data Type	Allow Nulls
EmployeeId	Udt.SurrogateKeyInt:int	☐
EmployeeLastName	Udt.LastName:nvarchar(25)	☐
EmployeeFirstName	Udt.FirstName:nvarchar(25)	☐
EmployeeTitle	Udt.Title:nvarchar(30)	☐
EmployeeTitleOfCourtesy	Udt.TitleOfCourtesy:nvarcha...	☐
BirthDate	Udt.DateYYYYMMDD:date	☐
HireDate	Udt.DateYYYYMMDD:date	☐
EmployeeAddress	Udt.Address:nvarchar(60)	☐
EmployeeCity	Udt.City:nvarchar(15)	☑
EmployeeRegion	Udt.Region:nvarchar(15)	☑
EmployeePostalCode	Udt.PostalCode:nvarchar(10)	☑
EmployeeCountry	Udt.Country:nvarchar(15)	☐
EmployeePhoneNumber	Udt.TelephoneNumber:nvar...	☐
EmployeeManagerId	Udt.SurrogateKeyInt:int	☑
		☐

Customer (Sales)

Column Name	Data Type	Allow Nulls
CustomerId	Udt.SurrogateKeyInt:int	☐
CustomerCompanyName	Udt.CompanyName:nvarch...	☐
CustomerContactName	Udt.ContactName:nvarchar(...	☐
CustomerContactTitle	Udt.Title:nvarchar(30)	☐
CustomerAddress	Udt.Address:nvarchar(60)	☐
CustomerCity	Udt.City:nvarchar(15)	☐
CustomerRegion	Udt.Region:nvarchar(15)	☑
CustomerPostalCode	Udt.PostalCode:nvarchar(10)	☑
CustomerCountry	Udt.Country:nvarchar(15)	☐
CustomerPhoneNumber	Udt.TelephoneNumber:nvar...	☐
CustomerFaxNumber	Udt.TelephoneNumber:nvar...	☑
		☐

TABLE SHOWING COLUMNS PROJECTED IN THE END

Table Name	Column Name
Sales.Customer	CustomerId
HumanResources.Employee	EmployeeId Name (using EmployeeFirstName, EmployeeLastName)
Sales.Order	OrderId

TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

Not applicable

QUERY OF WORST

```sql
USE Northwinds2022TSQLV7;

DROP FUNCTION IF EXISTS HumanResources.findFullName;
GO
CREATE FUNCTION HumanResources.findFullName
(
    @firstName VARCHAR(30),
    @lastName VARCHAR(30)
)
RETURNS VARCHAR(65)
AS
BEGIN
    DECLARE @Result VARCHAR(65);

    SELECT @Result = CONCAT(@firstName, ' ', @lastName);

    RETURN @Result;
END;
GO
SELECT O.OrderId,
       C.CustomerId,
       E.EmployeeId,
       HumanResources.findFullName(E.EmployeeFirstName, E.EmployeeLastName) AS [NAME]
FROM Sales.Customer AS C
    CROSS JOIN HumanResources.Employee AS E
    CROSS JOIN Sales.[Order] AS O
WHERE C.CustomerId =
(
    SELECT O.CustomerId WHERE E.EmployeeId = O.EmployeeId
)
GROUP BY O.OrderId,
         C.CustomerId,
         E.EmployeeId,
         HumanResources.findFullName(E.EmployeeFirstName, E.EmployeeLastName);
```

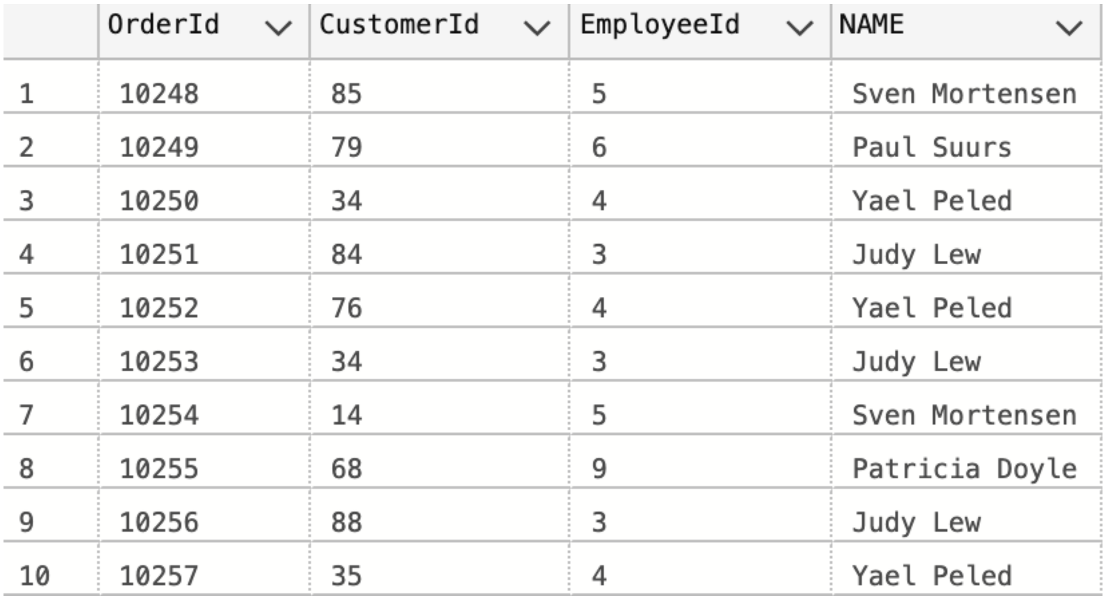
QUERY OF WORST CORRECTED

```sql
USE Northwinds2022TSQLV7;

SELECT O.OrderId,
       C.CustomerId,
       E.EmployeeId,
       CONCAT(E.EmployeeFirstName, ' ', E.EmployeeLastName) AS [NAME]
FROM Sales.Customer AS C
    INNER JOIN Sales.[Order] AS O
        ON C.CustomerId = O.CustomerId
    INNER JOIN HumanResources.Employee AS E
        ON E.EmployeeId = O.EmployeeId
GROUP BY O.OrderId,
         C.CustomerId,
         E.EmployeeId,
         CONCAT(E.EmployeeFirstName, ' ', E.EmployeeLastName);
```

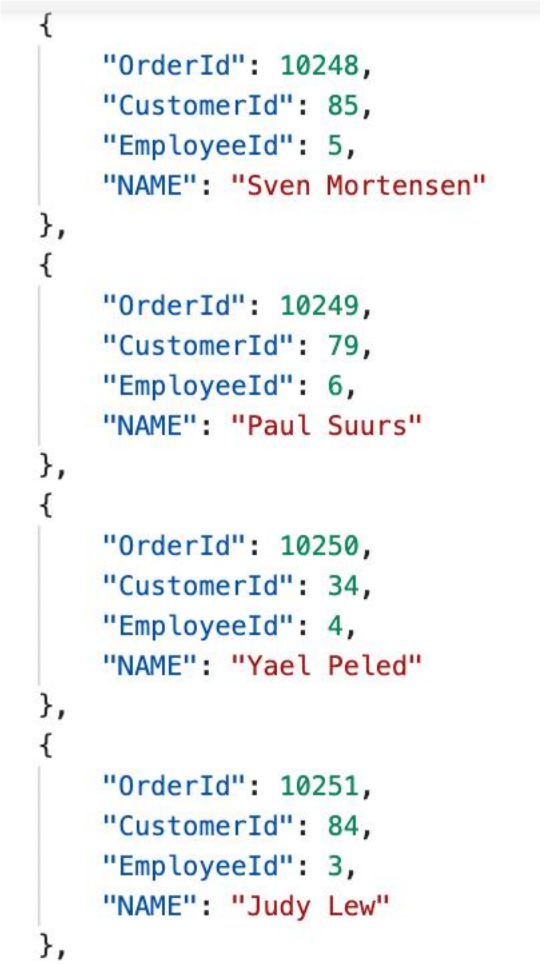
HOW IT WAS CORRECTED:

Used built-in concat function, also used an inner join to replace cross join which now does not need a subquery

RELATIONAL AND JSON OUTPUT (830 ROWS AFFECTED)



	OrderId	CustomerId	EmployeeId	NAME
1	10248	85	5	Sven Mortensen
2	10249	79	6	Paul Suurs
3	10250	34	4	Yael Peled
4	10251	84	3	Judy Lew
5	10252	76	4	Yael Peled
6	10253	34	3	Judy Lew
7	10254	14	5	Sven Mortensen
8	10255	68	9	Patricia Doyle
9	10256	88	3	Judy Lew
10	10257	35	4	Yael Peled



```
{
    "OrderId": 10248,
    "CustomerId": 85,
    "EmployeeId": 5,
    "NAME": "Sven Mortensen"
},
{
    "OrderId": 10249,
    "CustomerId": 79,
    "EmployeeId": 6,
    "NAME": "Paul Suurs"
},
{
    "OrderId": 10250,
    "CustomerId": 34,
    "EmployeeId": 4,
    "NAME": "Yael Peled"
},
{
    "OrderId": 10251,
    "CustomerId": 84,
    "EmployeeId": 3,
    "NAME": "Judy Lew"
},
```