# CSCI 331 PROJECT 1 – INDIVIDUAL PDF

PROFESSOR: PETER HELLER

SECTION: 9:15-10:30 AM

GROUP: G9-5

PDF OF: MIKHAIEL GOMES

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

## CONTENTS

## TOP #1: A [DIFFICULTY] QUERY

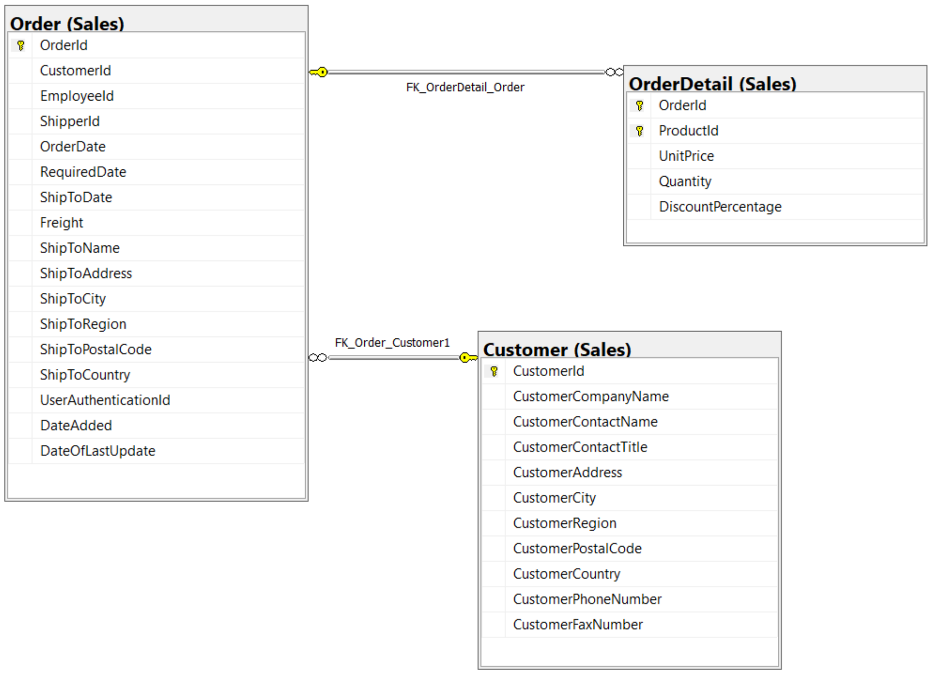
### PROBLEM STATEMENT

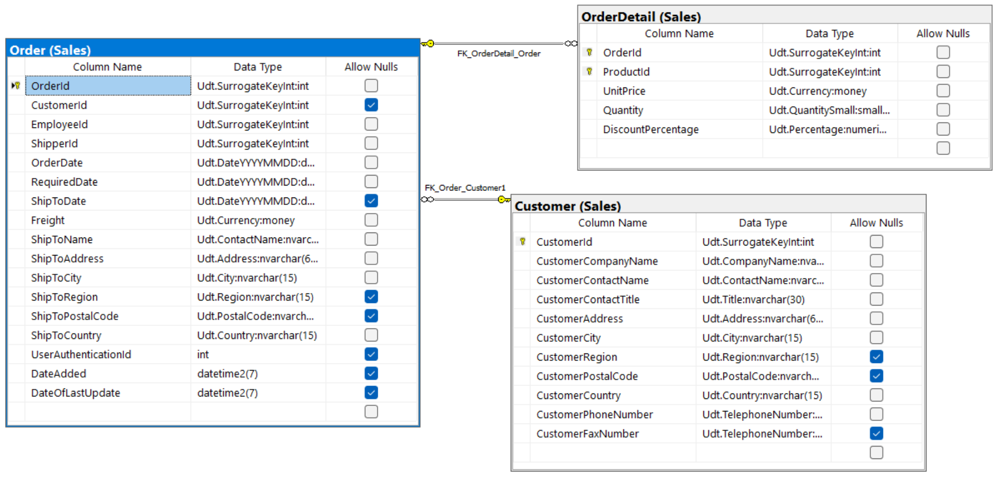
Find the top 5 spending customers using Northwinds2022TSQLV7.

### REASON IT IS A TOP

My solution for it uses a user defined scalar function to find total spent which essentially helps avoiding multiple inner joins in the main query.

### 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) | ✓ |
|  |  | ☐ |

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... | ☐ |
|  |  | ☐ |

Customer (Sales)

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| CustomerId | Udt.SurrogateKeyInt:int | ☐ |
| CustomerCompanyName | Udt.CompanyName:nva... | ☐ |
| CustomerContactName | Udt.ContactName:nvarc... | ☐ |
| CustomerContactTitle | Udt.Title:nvarchar(30) | ☐ |
| CustomerAddress | Udt.Address:nvarchar(6... | ☐ |
| CustomerCity | Udt.City:nvarchar(15) | ☐ |
| CustomerRegion | Udt.Region:nvarchar(15) | ✓ |
| CustomerPostalCode | Udt.PostalCode:nvarch... | ✓ |
| CustomerCountry | Udt.Country:nvarchar(15) | ☐ |
| CustomerPhoneNumber | Udt.TelephoneNumber:... | ☐ |
| CustomerFaxNumber | Udt.TelephoneNumber:... | ✓ |
|  |  | ☐ |

TABLE SHOWING COLUMNS PROJECTED IN THE END

| **Table Name** | **Column Name** |
|---|---|
| Derived | TotalSpent |
| Sales.Customer | CustomerContactName As Name |
| Sales.OrderDetail | ShipToCity As City |

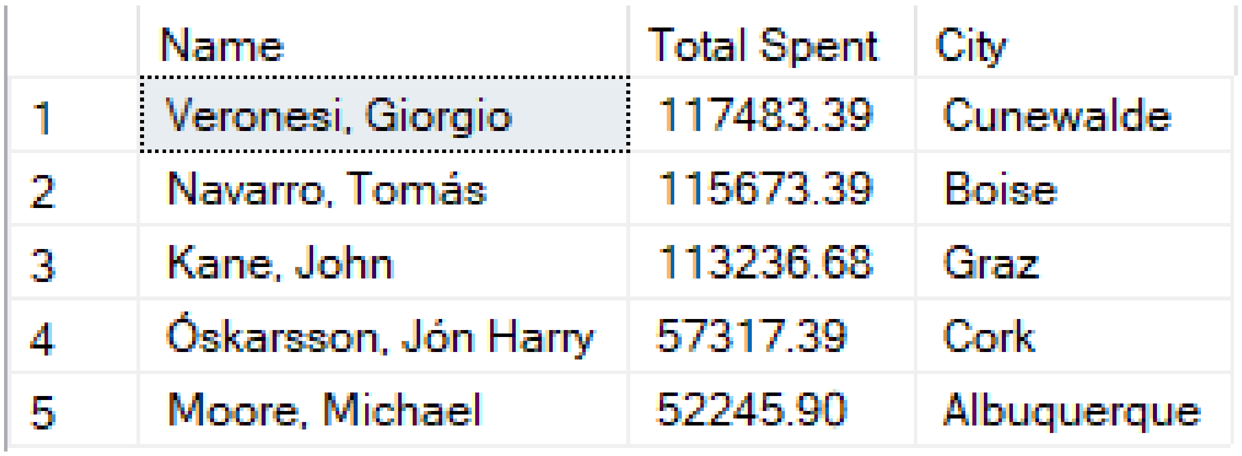
TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

| **Table Name** | **Column Name** | **Sort Order** |
|---|---|---|
| Derived | Total Spent | Decending |

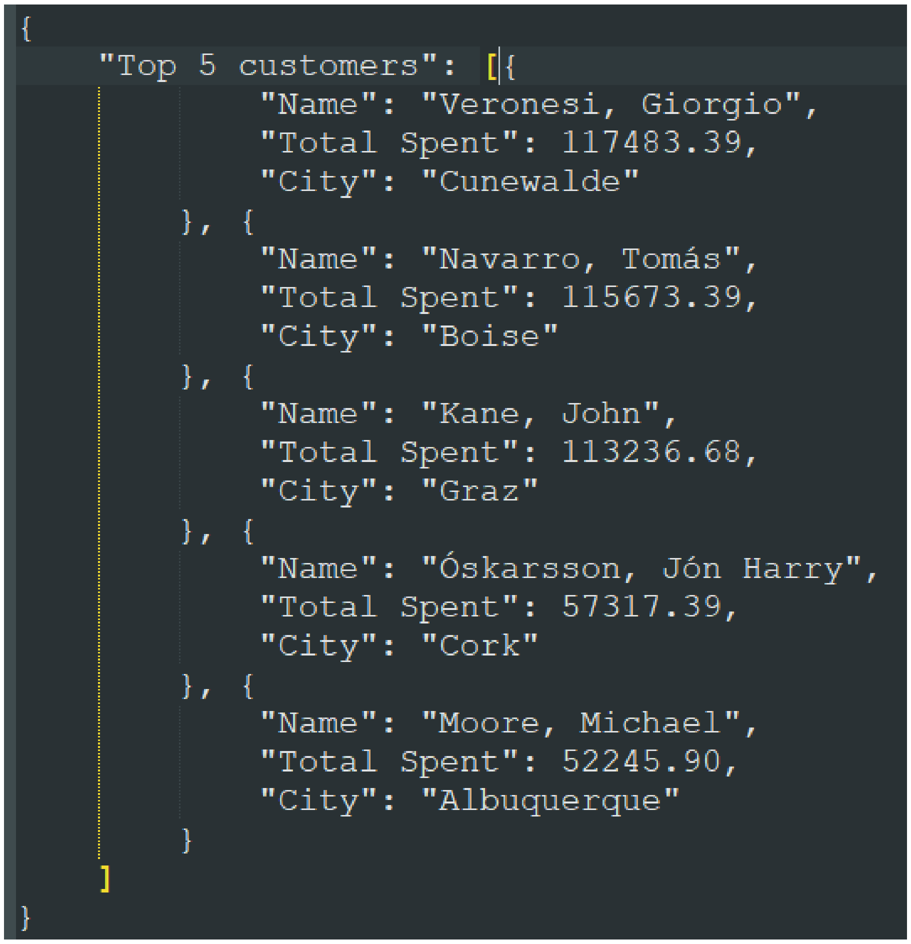
QUERY

```
USE Northwinds2022TSQLV7;
GO

DROP FUNCTION IF EXISTS dbo.TotalSpent;
GO
CREATE FUNCTION dbo.TotalSpent
(
    @name AS VARCHAR(50)
)
RETURNS NUMERIC(18, 2)
AS
BEGIN
    DECLARE @total FLOAT;

    SELECT @total = SUM(OD.UnitPrice * OD.Quantity)
    FROM Sales.[Order] AS O
        INNER JOIN Sales.OrderDetail AS OD
            ON O.OrderId = OD.OrderId
        INNER JOIN Sales.Customer AS C
            ON O.CustomerId = C.CustomerId
    WHERE C.CustomerContactName = @name;

    RETURN @total;
END;
GO

SELECT TOP 5
       C.CustomerContactName AS [Name],
       dbo.TotalSpent(C.CustomerContactName) AS [Total Spent],
       O.ShipToCity AS City
FROM Sales.[Order] AS O
    INNER JOIN Sales.OrderDetail AS OD
        ON O.OrderId = OD.OrderId
    INNER JOIN Sales.Customer AS C
        ON O.CustomerId = C.CustomerId
GROUP BY C.CustomerContactName,
         O.ShipToCity
ORDER BY [Total Spent] DESC;

GO
```

RELATIONAL AND JSON OUTPUT (5 ROWS AFFECTED)



| | Name | Total Spent | City |
|---|---|---|---|
| 1 | Veronesi, Giorgio | 117483.39 | Cunewalde |
| 2 | Navarro, Tomás | 115673.39 | Boise |
| 3 | Kane, John | 113236.68 | Graz |
| 4 | Óskarsson, Jón Harry | 57317.39 | Cork |
| 5 | Moore, Michael | 52245.90 | Albuquerque |



```
{
    "Top 5 customers": [{
            "Name": "Veronesi, Giorgio",
            "Total Spent": 117483.39,
            "City": "Cunewalde"
        }, {
            "Name": "Navarro, Tomás",
            "Total Spent": 115673.39,
            "City": "Boise"
        }, {
            "Name": "Kane, John",
            "Total Spent": 113236.68,
            "City": "Graz"
        }, {
            "Name": "Óskarsson, Jón Harry",
            "Total Spent": 57317.39,
            "City": "Cork"
        }, {
            "Name": "Moore, Michael",
            "Total Spent": 52245.90,
            "City": "Albuquerque"
        }
    ]
}
```

## TOP #2: A COMPLEX QUERY

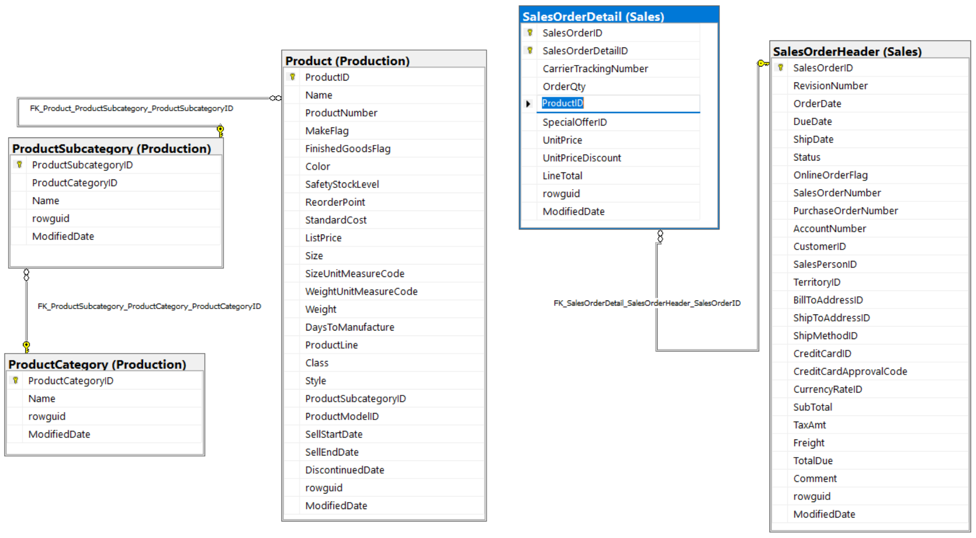
### PROBLEM STATEMENT

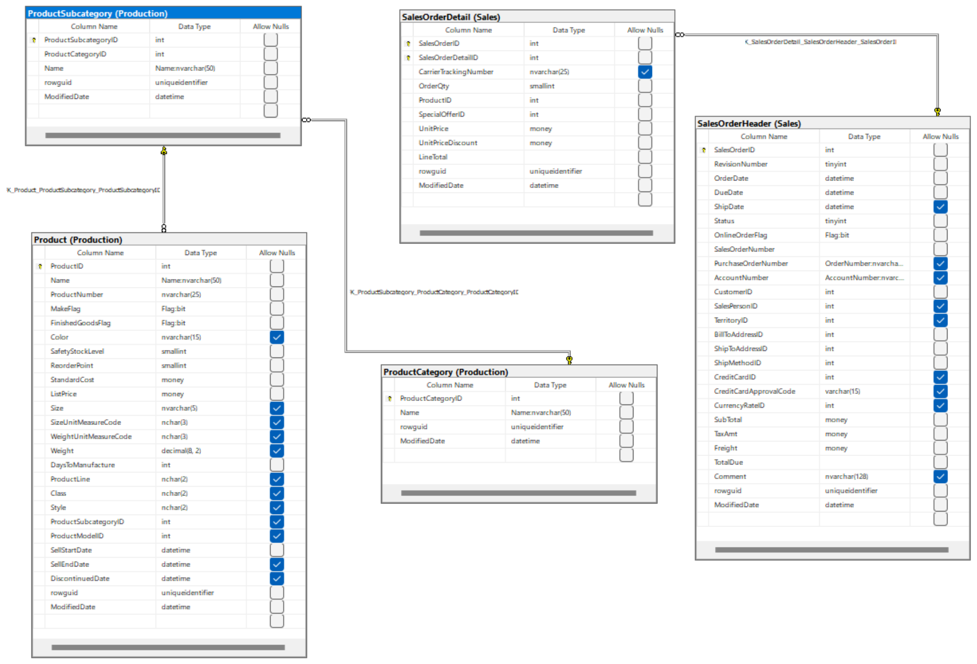
For each category and sub category of products show the total sales made for each quarter in 2013 using AdventureWorks2017

### REASON IT IS A TOP

Utilizes built in functions of SQL to avoid writing UDFs.

### KEY AND STANDARD VIEW OF TABLES USED




'K_Product_ProductSubcategory_ProductSubcategoryIE
'K_ProductSubcategory_ProductCategory_ProductCategoryIE
K_SalesOrderDetail_SalesOrderHeader_SalesOrderII

**ProductSubcategory (Production)**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| ProductSubcategoryID | int | |
| ProductCategoryID | int | |
| Name | Name:nvarchar(50) | |
| rowguid | uniqueidentifier | |
| ModifiedDate | datetime | |

**SalesOrderDetail (Sales)**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| SalesOrderID | int | |
| SalesOrderDetailID | int | |
| CarrierTrackingNumber | nvarchar(25) | ✓ |
| OrderQty | smallint | |
| ProductID | int | |
| SpecialOfferID | int | |
| UnitPrice | money | |
| UnitPriceDiscount | money | |
| LineTotal | | |
| rowguid | uniqueidentifier | |
| ModifiedDate | datetime | |

**Product (Production)**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| ProductID | int | |
| Name | Name:nvarchar(50) | |
| ProductNumber | nvarchar(25) | |
| MakeFlag | Flag:bit | |
| FinishedGoodsFlag | Flag:bit | |
| Color | nvarchar(15) | ✓ |
| SafetyStockLevel | smallint | |
| ReorderPoint | smallint | |
| StandardCost | money | |
| ListPrice | money | |
| Size | nvarchar(5) | ✓ |
| SizeUnitMeasureCode | nchar(3) | ✓ |
| WeightUnitMeasureCode | nchar(3) | ✓ |
| Weight | decimal(8, 2) | ✓ |
| DaysToManufacture | int | |
| ProductLine | nchar(2) | ✓ |
| Class | nchar(2) | ✓ |
| Style | nchar(2) | ✓ |
| ProductSubcategoryID | int | ✓ |
| ProductModelID | int | ✓ |
| SellStartDate | datetime | |
| SellEndDate | datetime | ✓ |
| DiscontinuedDate | datetime | ✓ |
| rowguid | uniqueidentifier | |
| ModifiedDate | datetime | |

**ProductCategory (Production)**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| ProductCategoryID | int | |
| Name | Name:nvarchar(50) | |
| rowguid | uniqueidentifier | |
| ModifiedDate | datetime | |

**SalesOrderHeader (Sales)**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| SalesOrderID | int | |
| RevisionNumber | tinyint | |
| OrderDate | datetime | |
| DueDate | datetime | |
| ShipDate | datetime | ✓ |
| Status | tinyint | |
| OnlineOrderFlag | Flag:bit | |
| SalesOrderNumber | | |
| PurchaseOrderNumber | OrderNumber:nvarcha... | ✓ |
| AccountNumber | AccountNumber:nvarc... | ✓ |
| CustomerID | int | |
| SalesPersonID | int | ✓ |
| TerritoryID | int | ✓ |
| BillToAddressID | int | |
| ShipToAddressID | int | |
| ShipMethodID | int | |
| CreditCardID | int | ✓ |
| CreditCardApprovalCode | varchar(15) | ✓ |
| CurrencyRateID | int | ✓ |
| SubTotal | money | |
| TaxAmt | money | |
| Freight | money | |
| TotalDue | | |
| Comment | nvarchar(128) | ✓ |
| rowguid | uniqueidentifier | |
| ModifiedDate | datetime | |

TABLE SHOWING COLUMNS PROJECTED IN THE END

| Table Name | Column Name |
|---|---|
| Production.ProductCategory | Name as Category |
| Production.ProductSubcategory | Name AS Subcategory |
| Derived | Year<br>Qtr<br>$ Sales |

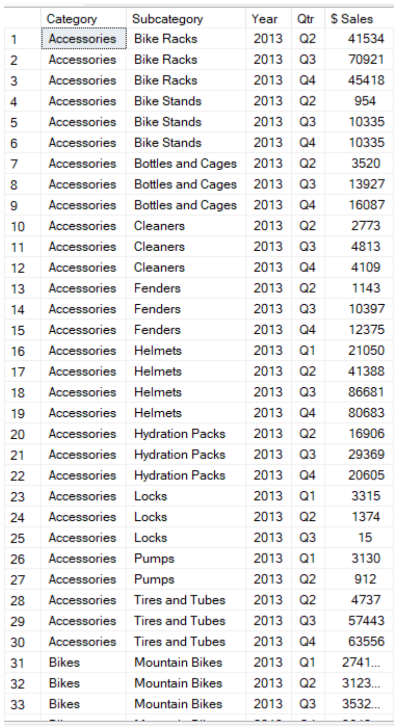
TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

| Table Name | Column Name | Sort Order |
|---|---|---|
| Production.ProductCategory | Name as Category | Ascending |
| Production.ProductSubcategory | Name AS Subcategory | Ascending |
| Qtr | Derived | Ascending |

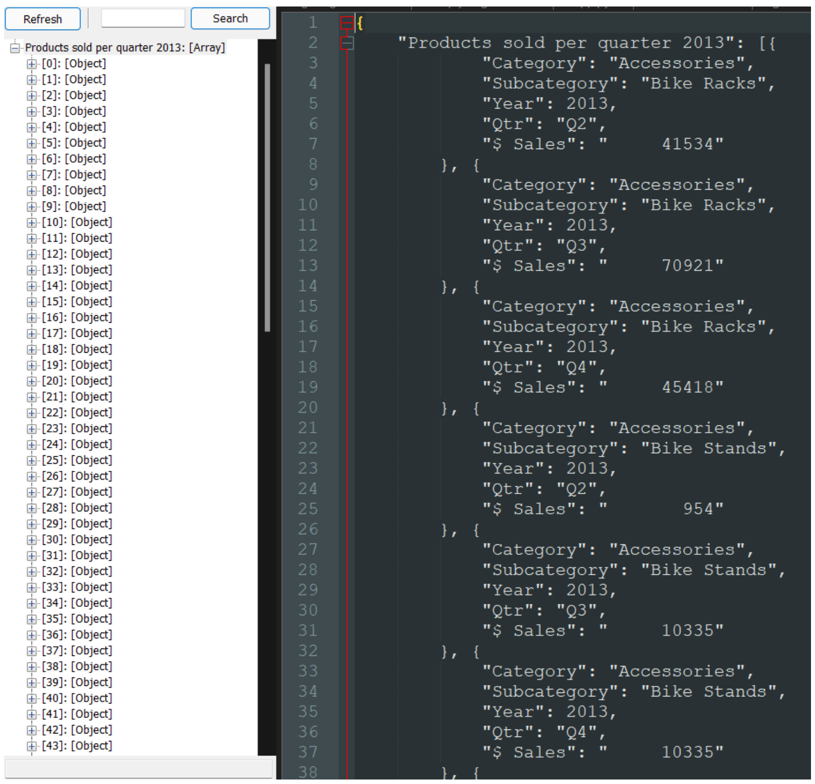
QUERY

```
USE AdventureWorks2017;
GO

SELECT PC.Name AS Category,
       PS.Name AS Subcategory,
       DATEPART(yy, SOH.OrderDate) AS [Year],
       'Q' + DATENAME(qq, SOH.OrderDate) AS [Qtr],
       STR(SUM(DET.UnitPrice * DET.OrderQty)) AS [$ Sales]
FROM Production.ProductSubcategory AS PS
    INNER JOIN Sales.SalesOrderHeader AS SOH
        INNER JOIN Sales.SalesOrderDetail DET
            ON SOH.SalesOrderID = DET.SalesOrderID
        INNER JOIN Production.Product P
            ON DET.ProductID = P.ProductID
        ON PS.ProductSubcategoryID = P.ProductSubcategoryID
    INNER JOIN Production.ProductCategory PC
        ON PS.ProductCategoryID = PC.ProductCategoryID
WHERE YEAR(SOH.OrderDate) = '2013'
GROUP BY DATEPART(yy, SOH.OrderDate),
         PC.Name,
         PS.Name,
         'Q' + DATENAME(qq, SOH.OrderDate),
         PS.ProductSubcategoryID
ORDER BY Category,
         Subcategory,

         [Qtr];
```

## RELATIONAL AND JSON OUTPUT (112 ROWS AFFECTED)



| | Category | Subcategory | Year | Qtr | $ Sales |
|---|---|---|---|---|---|
| 1 | Accessories | Bike Racks | 2013 | Q2 | 41534 |
| 2 | Accessories | Bike Racks | 2013 | Q3 | 70921 |
| 3 | Accessories | Bike Racks | 2013 | Q4 | 45418 |
| 4 | Accessories | Bike Stands | 2013 | Q2 | 954 |
| 5 | Accessories | Bike Stands | 2013 | Q3 | 10335 |
| 6 | Accessories | Bike Stands | 2013 | Q4 | 10335 |
| 7 | Accessories | Bottles and Cages | 2013 | Q2 | 3520 |
| 8 | Accessories | Bottles and Cages | 2013 | Q3 | 13927 |
| 9 | Accessories | Bottles and Cages | 2013 | Q4 | 16087 |
| 10 | Accessories | Cleaners | 2013 | Q2 | 2773 |
| 11 | Accessories | Cleaners | 2013 | Q3 | 4813 |
| 12 | Accessories | Cleaners | 2013 | Q4 | 4109 |
| 13 | Accessories | Fenders | 2013 | Q2 | 1143 |
| 14 | Accessories | Fenders | 2013 | Q3 | 10397 |
| 15 | Accessories | Fenders | 2013 | Q4 | 12375 |
| 16 | Accessories | Helmets | 2013 | Q1 | 21050 |
| 17 | Accessories | Helmets | 2013 | Q2 | 41388 |
| 18 | Accessories | Helmets | 2013 | Q3 | 86681 |
| 19 | Accessories | Helmets | 2013 | Q4 | 80683 |
| 20 | Accessories | Hydration Packs | 2013 | Q2 | 16906 |
| 21 | Accessories | Hydration Packs | 2013 | Q3 | 29369 |
| 22 | Accessories | Hydration Packs | 2013 | Q4 | 20605 |
| 23 | Accessories | Locks | 2013 | Q1 | 3315 |
| 24 | Accessories | Locks | 2013 | Q2 | 1374 |
| 25 | Accessories | Locks | 2013 | Q3 | 15 |
| 26 | Accessories | Pumps | 2013 | Q1 | 3130 |
| 27 | Accessories | Pumps | 2013 | Q2 | 912 |
| 28 | Accessories | Tires and Tubes | 2013 | Q2 | 4737 |
| 29 | Accessories | Tires and Tubes | 2013 | Q3 | 57443 |
| 30 | Accessories | Tires and Tubes | 2013 | Q4 | 63556 |
| 31 | Bikes | Mountain Bikes | 2013 | Q1 | 2741... |
| 32 | Bikes | Mountain Bikes | 2013 | Q2 | 3123... |
| 33 | Bikes | Mountain Bikes | 2013 | Q3 | 3532... |



```
Products sold per quarter 2013: [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]
  [40]: [Object]
  [41]: [Object]
  [42]: [Object]
  [43]: [Object]
```

```
{
    "Products sold per quarter 2013": [{
            "Category": "Accessories",
            "Subcategory": "Bike Racks",
            "Year": 2013,
            "Qtr": "Q2",
            "$ Sales": "     41534"
        }, {
            "Category": "Accessories",
            "Subcategory": "Bike Racks",
            "Year": 2013,
            "Qtr": "Q3",
            "$ Sales": "     70921"
        }, {
            "Category": "Accessories",
            "Subcategory": "Bike Racks",
            "Year": 2013,
            "Qtr": "Q4",
            "$ Sales": "     45418"
        }, {
            "Category": "Accessories",
            "Subcategory": "Bike Stands",
            "Year": 2013,
            "Qtr": "Q2",
            "$ Sales": "       954"
        }, {
            "Category": "Accessories",
            "Subcategory": "Bike Stands",
            "Year": 2013,
            "Qtr": "Q3",
            "$ Sales": "     10335"
        }, {
            "Category": "Accessories",
            "Subcategory": "Bike Stands",
            "Year": 2013,
            "Qtr": "Q4",
            "$ Sales": "     10335"
        }, {
```

## TOP #3: A COMPLEX QUERY

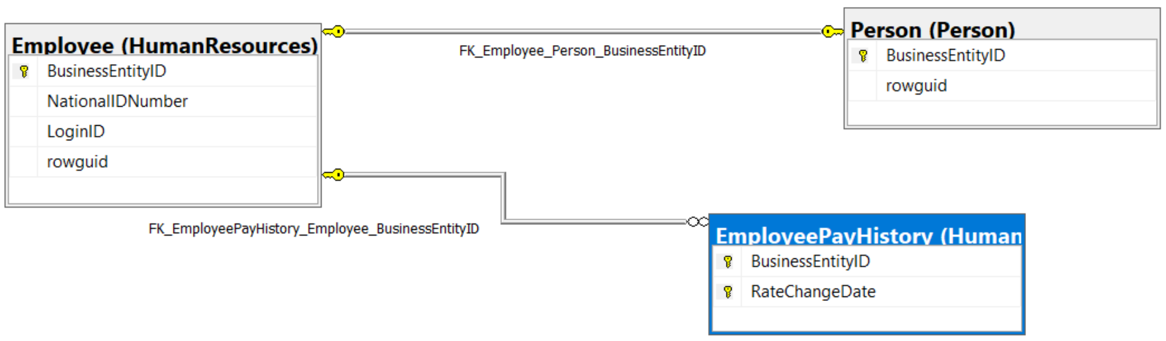
### PROBLEM STATEMENT

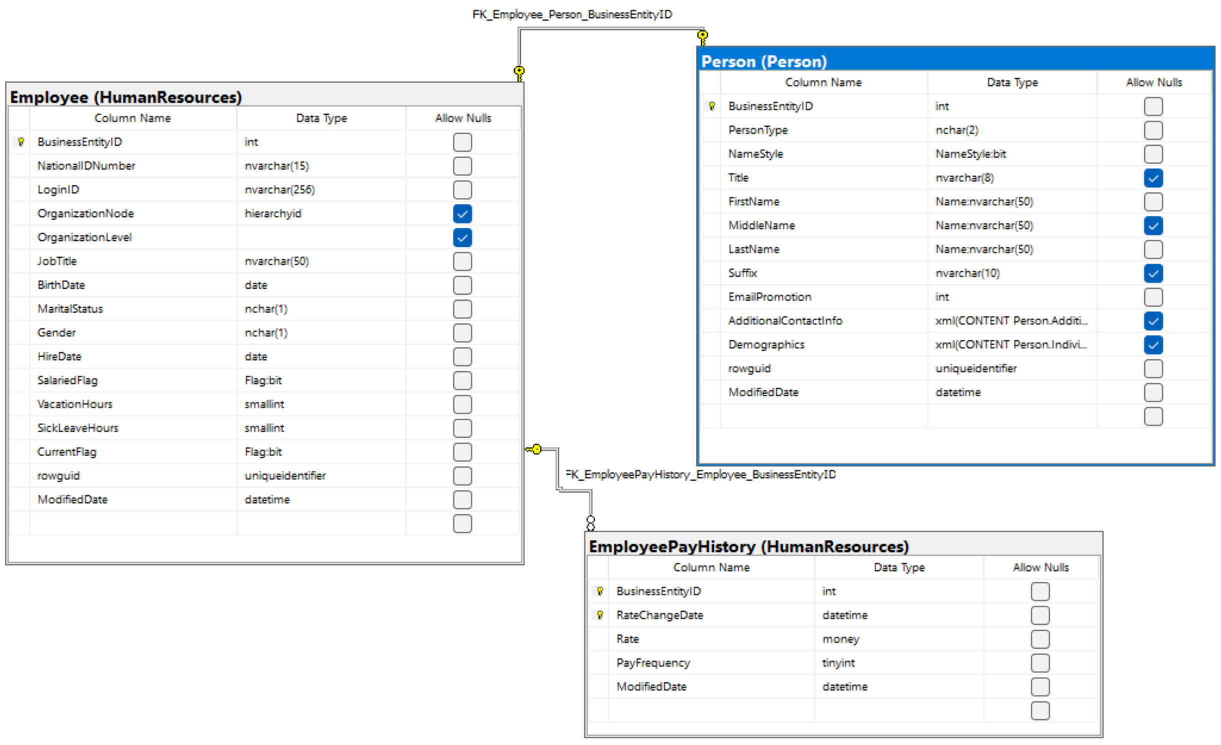
Show all for how many years the employees has been hired for and their pay rates.

### REASON IT IS A TOP

Interesting problem that uses UDF, built in functions, multiple table joins to find the result.

### KEY AND STANDARD VIEW OF TABLES USED


Employee (HumanResources)
BusinessEntityID
NationalIDNumber
LoginID
rowguid
FK_Employee_Person_BusinessEntityID
Person (Person)
BusinessEntityID
rowguid
FK_EmployeePayHistory_Employee_BusinessEntityID
EmployeePayHistory (Human
BusinessEntityID
RateChangeDate


FK_Employee_Person_BusinessEntityID
FK_EmployeePayHistory_Employee_BusinessEntityID

**Employee (HumanResources)**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| BusinessEntityID | int | ☐ |
| NationalIDNumber | nvarchar(15) | ☐ |
| LoginID | nvarchar(256) | ☐ |
| OrganizationNode | hierarchyid | ✓ |
| OrganizationLevel | | ✓ |
| JobTitle | nvarchar(50) | ☐ |
| BirthDate | date | ☐ |
| MaritalStatus | nchar(1) | ☐ |
| Gender | nchar(1) | ☐ |
| HireDate | date | ☐ |
| SalariedFlag | Flag:bit | ☐ |
| VacationHours | smallint | ☐ |
| SickLeaveHours | smallint | ☐ |
| CurrentFlag | Flag:bit | ☐ |
| rowguid | uniqueidentifier | ☐ |
| ModifiedDate | datetime | ☐ |

**Person (Person)**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| BusinessEntityID | int | ☐ |
| PersonType | nchar(2) | ☐ |
| NameStyle | NameStyle:bit | ☐ |
| Title | nvarchar(8) | ✓ |
| FirstName | Name:nvarchar(50) | ☐ |
| MiddleName | Name:nvarchar(50) | ✓ |
| LastName | Name:nvarchar(50) | ☐ |
| Suffix | nvarchar(10) | ✓ |
| EmailPromotion | int | ☐ |
| AdditionalContactInfo | xml(CONTENT Person.Additi... | ✓ |
| Demographics | xml(CONTENT Person.Indivi... | ✓ |
| rowguid | uniqueidentifier | ☐ |
| ModifiedDate | datetime | ☐ |

**EmployeePayHistory (HumanResources)**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| BusinessEntityID | int | ☐ |
| RateChangeDate | datetime | ☐ |
| Rate | money | ☐ |
| PayFrequency | tinyint | ☐ |
| ModifiedDate | datetime | ☐ |

TABLE SHOWING COLUMNS PROJECTED IN THE END

| **Table Name** | **Column Name** |
|---|---|
| Derived | Name<br><br>Worklength |
| HumanResources.EmployeePayHistory | Rate |

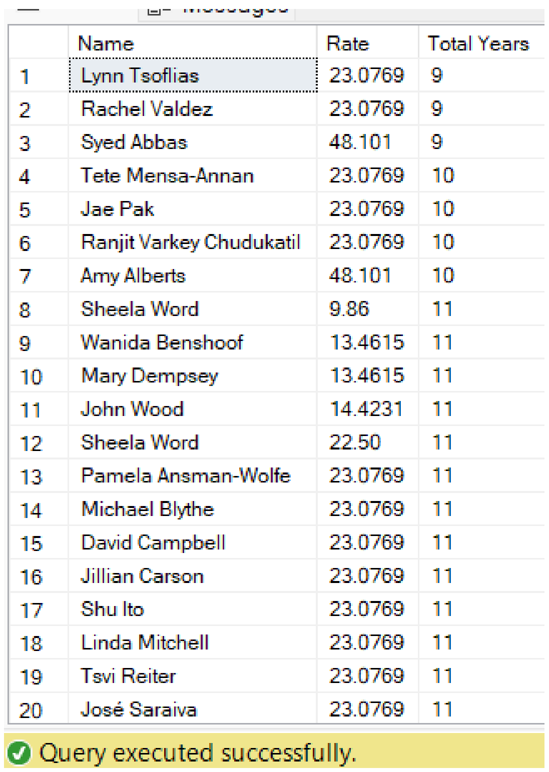
TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

| **Table Name** | **Column Name** | **Sort Order** |
|---|---|---|
| HumanResources.EmployeePayHistory | Rate | ASC |
| Derived | WorkLength | ASC |
| Person.Person | LastName | ASC |

QUERY

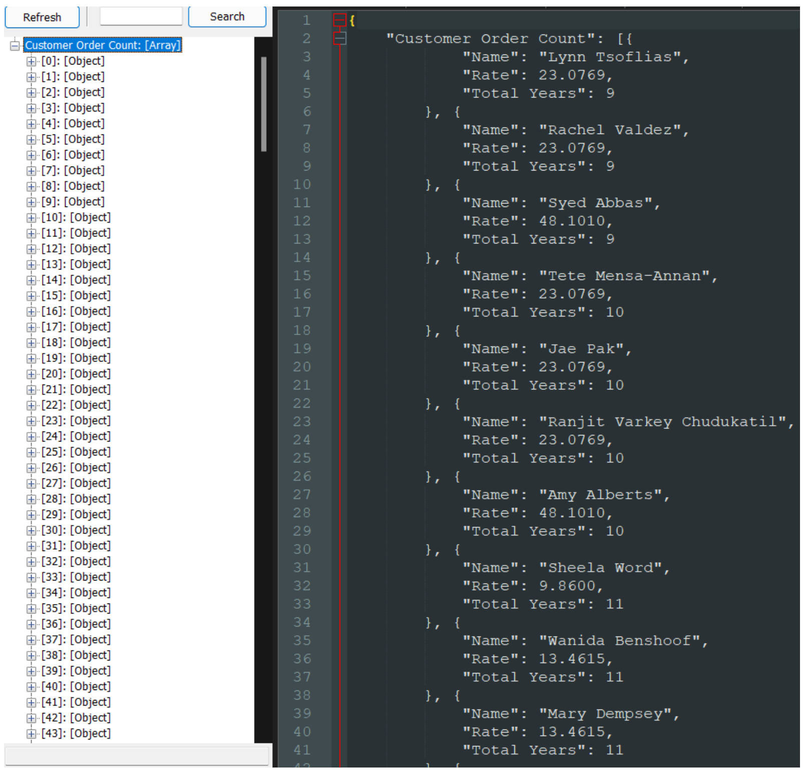
```
USE AdventureWorks2017;
GO

DROP FUNCTION IF EXISTS dbo.Worklength;
GO
CREATE FUNCTION dbo.Worklength
(
    @hired DATE
)
RETURNS INT
AS
BEGIN
    DECLARE @length INT;

    SELECT @length = DATEDIFF(YEAR, @hired, GETDATE())
    FROM HumanResources.Employee
    WHERE HireDate = @hired;

    RETURN @length;
END;
GO

SELECT CONCAT(P.FirstName, ' ', P.LastName) AS [Name],
       H.Rate,
       dbo.Worklength(E.HireDate) AS [Total Years]
FROM HumanResources.Employee AS E
    INNER JOIN Person.Person AS P
        ON E.BusinessEntityID = P.BusinessEntityID
    INNER JOIN HumanResources.EmployeePayHistory AS H
        ON E.BusinessEntityID = H.BusinessEntityID
ORDER BY dbo.Worklength(E.HireDate),
         H.Rate,

         P.LastName;
```

## RELATIONAL AND JSON OUTPUT (316 ROWS AFFECTED)



| | Name | Rate | Total Years |
|---|---|---|---|
| 1 | Lynn Tsoflias | 23.0769 | 9 |
| 2 | Rachel Valdez | 23.0769 | 9 |
| 3 | Syed Abbas | 48.101 | 9 |
| 4 | Tete Mensa-Annan | 23.0769 | 10 |
| 5 | Jae Pak | 23.0769 | 10 |
| 6 | Ranjit Varkey Chudukatil | 23.0769 | 10 |
| 7 | Amy Alberts | 48.101 | 10 |
| 8 | Sheela Word | 9.86 | 11 |
| 9 | Wanida Benshoof | 13.4615 | 11 |
| 10 | Mary Dempsey | 13.4615 | 11 |
| 11 | John Wood | 14.4231 | 11 |
| 12 | Sheela Word | 22.50 | 11 |
| 13 | Pamela Ansman-Wolfe | 23.0769 | 11 |
| 14 | Michael Blythe | 23.0769 | 11 |
| 15 | David Campbell | 23.0769 | 11 |
| 16 | Jillian Carson | 23.0769 | 11 |
| 17 | Shu Ito | 23.0769 | 11 |
| 18 | Linda Mitchell | 23.0769 | 11 |
| 19 | Tsvi Reiter | 23.0769 | 11 |
| 20 | José Saraiva | 23.0769 | 11 |

Query executed successfully.



```
Customer Order Count: [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]
  [40]: [Object]
  [41]: [Object]
  [42]: [Object]
  [43]: [Object]
```

```
{
    "Customer Order Count": [{
            "Name": "Lynn Tsoflias",
            "Rate": 23.0769,
            "Total Years": 9
        }, {
            "Name": "Rachel Valdez",
            "Rate": 23.0769,
            "Total Years": 9
        }, {
            "Name": "Syed Abbas",
            "Rate": 48.1010,
            "Total Years": 9
        }, {
            "Name": "Tete Mensa-Annan",
            "Rate": 23.0769,
            "Total Years": 10
        }, {
            "Name": "Jae Pak",
            "Rate": 23.0769,
            "Total Years": 10
        }, {
            "Name": "Ranjit Varkey Chudukatil",
            "Rate": 23.0769,
            "Total Years": 10
        }, {
            "Name": "Amy Alberts",
            "Rate": 48.1010,
            "Total Years": 10
        }, {
            "Name": "Sheela Word",
            "Rate": 9.8600,
            "Total Years": 11
        }, {
            "Name": "Wanida Benshoof",
            "Rate": 13.4615,
            "Total Years": 11
        }, {
            "Name": "Mary Dempsey",
            "Rate": 13.4615,
            "Total Years": 11
```

## WORST #1: A MEDIUM QUERY

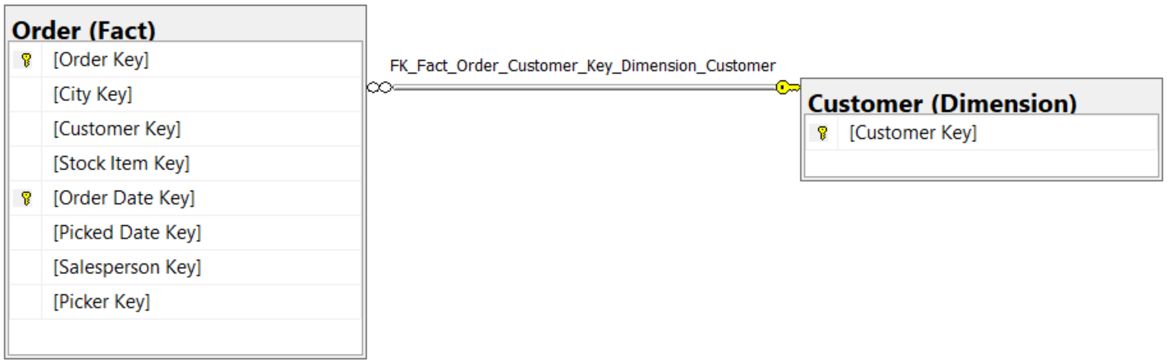
### PROBLEM STATEMENT

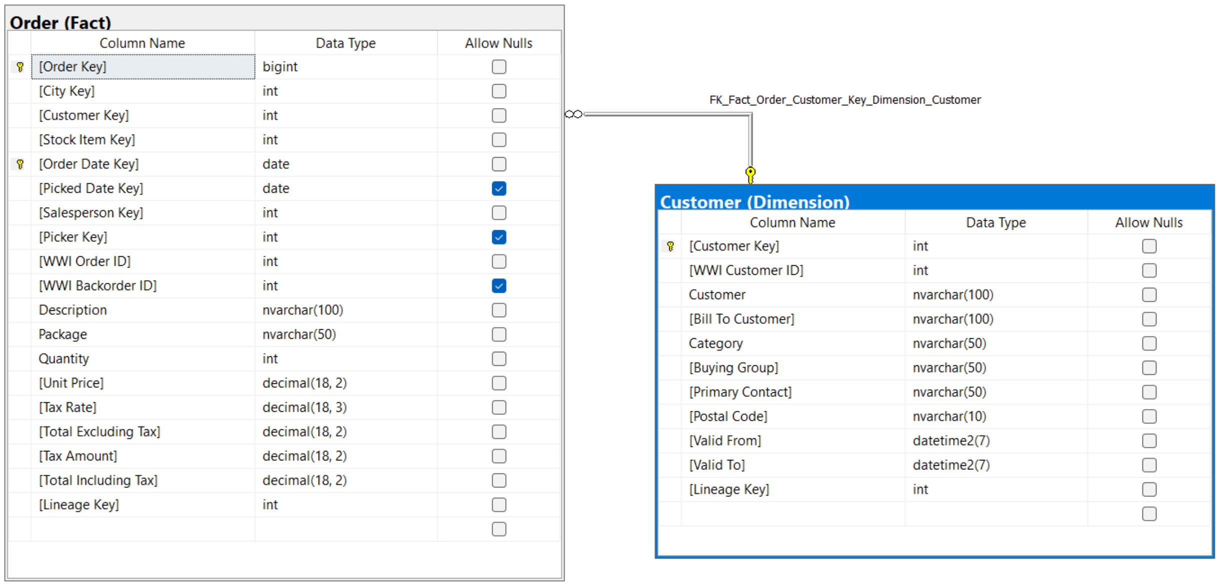
For customers in WideWorldDW find out for the customers who bought developer joke mug, how many have they purchased?

### REASON IT IS A WORST

In the Dimension.Customer table there are information for unknown customers. Which is problematic when we do the joins and get unwanted row in our result table.

### KEY AND STANDARD VIEW OF TABLES USED





**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 | ☐ |
| | | | ☐ |

**Customer (Dimension)**

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | [Customer Key] | int | ☐ |
| | [WWI Customer ID] | int | ☐ |
| | Customer | nvarchar(100) | ☐ |
| | [Bill To Customer] | nvarchar(100) | ☐ |
| | Category | nvarchar(50) | ☐ |
| | [Buying Group] | nvarchar(50) | ☐ |
| | [Primary Contact] | nvarchar(50) | ☐ |
| | [Postal Code] | nvarchar(10) | ☐ |
| | [Valid From] | datetime2(7) | ☐ |
| | [Valid To] | datetime2(7) | ☐ |
| | [Lineage Key] | int | ☐ |
| | | | ☐ |

TABLE SHOWING COLUMNS PROJECTED IN THE END

| Table Name | Column Name |
|---|---|
| Dimension.Customer | Primary Contact |
| Derived | Number Of Mugs |

TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

| Table Name | Column Name | Sort Order |
|---|---|---|
| Derived | Number of Mugs | Desc |

QUERY OF WORST

```
USE WideWorldImportersDW;
GO

SELECT C.[Primary Contact] AS [Customer],
       COUNT(C.[Primary Contact]) AS [Number of Mugs]
FROM Fact.[Order] AS O
    INNER JOIN Dimension.Customer AS C
        ON C.[Customer Key] = O.[Customer Key]
WHERE C.[Customer Key] != 0
      AND LOWER(O.[Description]) LIKE (LOWER(N'%Developer joke mug%'))
GROUP BY C.[Primary Contact]
ORDER BY COUNT(C.[Primary Contact]) DESC;
```

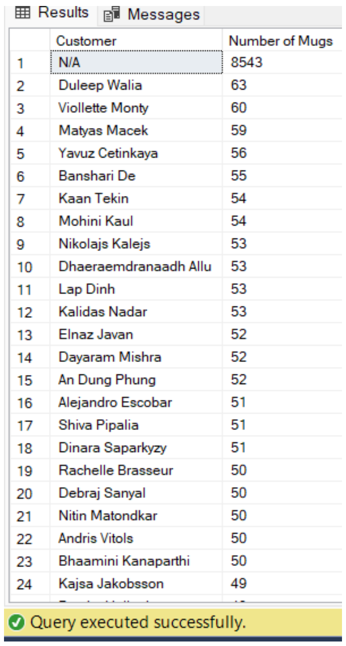
## QUERY OF WORST CORRECTED

```
USE WideWorldImportersDW;
GO

SELECT C.[Primary Contact] AS [Customer],
       COUNT(C.[Primary Contact]) AS [Number of Mugs]
FROM Fact.[Order] AS O
    INNER JOIN Dimension.Customer AS C
        ON C.[Customer Key] = O.[Customer Key]
WHERE C.[Customer Key] != 0
      AND LOWER(O.[Description]) LIKE (LOWER(N'%Developer joke mug%'))
GROUP BY C.[Primary Contact]
ORDER BY COUNT(C.[Primary Contact]) DESC;
```

## HOW IT WAS CORRECTED:

Identified the issue by inspecting the Dimension.Customer table and found that unknown customers were given the Customer Id of 0. Avoiding this ID in the where clause solved the issue.

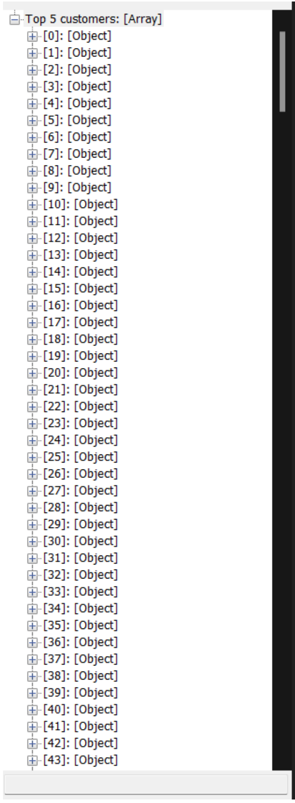
## RELATIONAL AND JSON OUTPUT (403 ROWS AFFECTED)



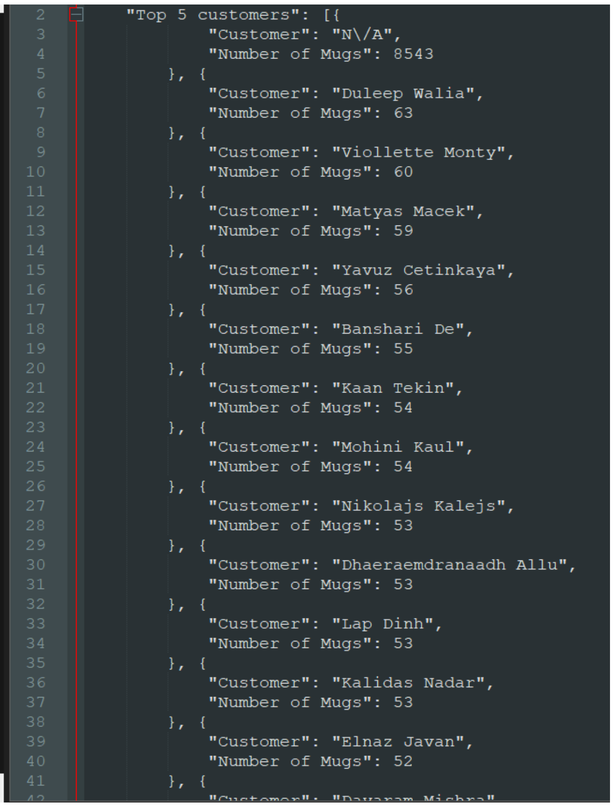
Results Messages

| | Customer | Number of Mugs |
|---|---|---|
| 1 | N/A | 8543 |
| 2 | Duleep Walia | 63 |
| 3 | Viollette Monty | 60 |
| 4 | Matyas Macek | 59 |
| 5 | Yavuz Cetinkaya | 56 |
| 6 | Banshari De | 55 |
| 7 | Kaan Tekin | 54 |
| 8 | Mohini Kaul | 54 |
| 9 | Nikolajs Kalejs | 53 |
| 10 | Dhaeraemdranaadh Allu | 53 |
| 11 | Lap Dinh | 53 |
| 12 | Kalidas Nadar | 53 |
| 13 | Elnaz Javan | 52 |
| 14 | Dayaram Mishra | 52 |
| 15 | An Dung Phung | 52 |
| 16 | Alejandro Escobar | 51 |
| 17 | Shiva Pipalia | 51 |
| 18 | Dinara Saparkyzy | 51 |
| 19 | Rachelle Brasseur | 50 |
| 20 | Debraj Sanyal | 50 |
| 21 | Nitin Matondkar | 50 |
| 22 | Andris Vitols | 50 |
| 23 | Bhaamini Kanaparthi | 50 |
| 24 | Kajsa Jakobsson | 49 |

Query executed successfully.

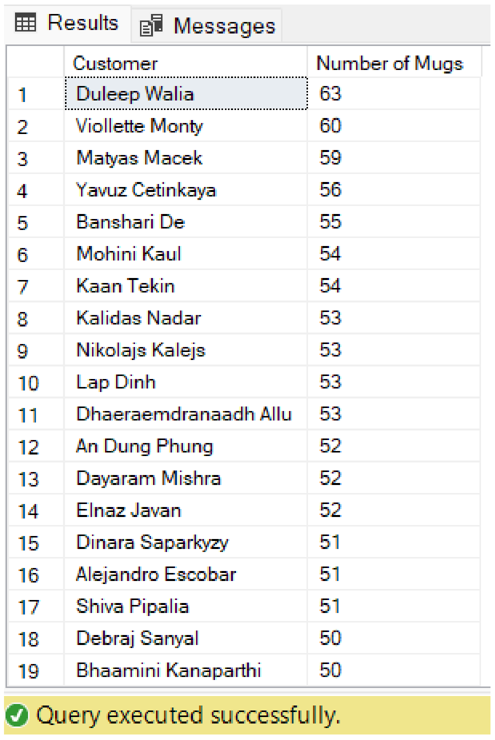


```
Top 5 customers: [Array]
  [0]: [Object]
  [1]: [Object]
  [2]: [Object]
  [3]: [Object]
  [4]: [Object]
  [5]: [Object]
  [6]: [Object]
  [7]: [Object]
  [8]: [Object]
  [9]: [Object]
  [10]: [Object]
  [11]: [Object]
  [12]: [Object]
  [13]: [Object]
  [14]: [Object]
  [15]: [Object]
  [16]: [Object]
  [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]
  [40]: [Object]
  [41]: [Object]
  [42]: [Object]
  [43]: [Object]
```



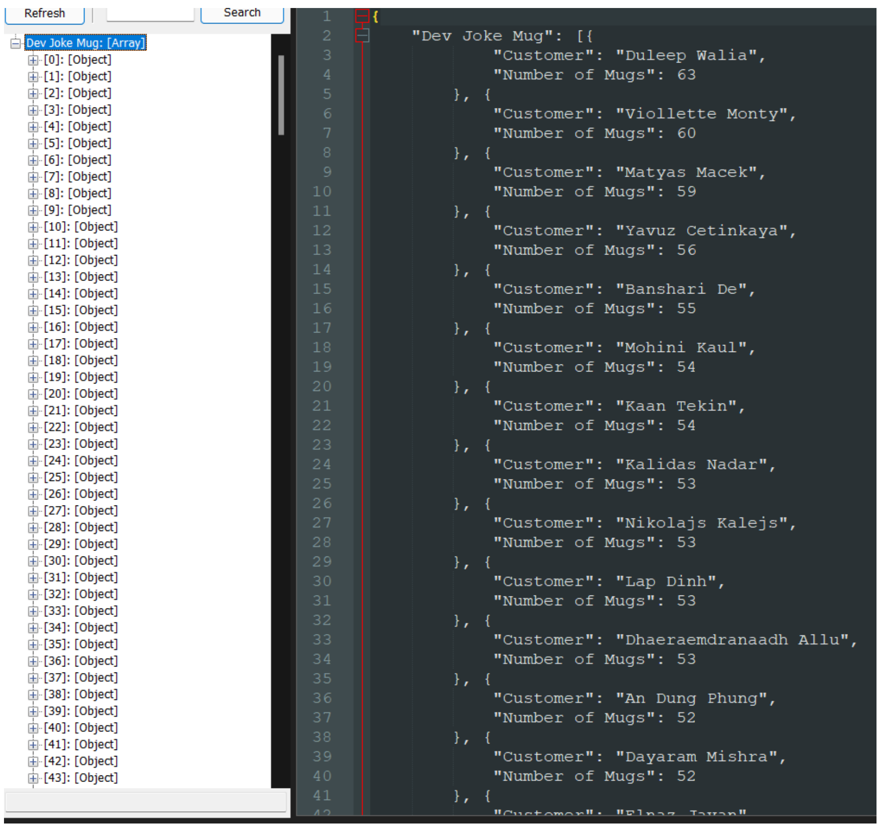
```
    "Top 5 customers": [{
            "Customer": "N\/A",
            "Number of Mugs": 8543
        }, {
            "Customer": "Duleep Walia",
            "Number of Mugs": 63
        }, {
            "Customer": "Viollette Monty",
            "Number of Mugs": 60
        }, {
            "Customer": "Matyas Macek",
            "Number of Mugs": 59
        }, {
            "Customer": "Yavuz Cetinkaya",
            "Number of Mugs": 56
        }, {
            "Customer": "Banshari De",
            "Number of Mugs": 55
        }, {
            "Customer": "Kaan Tekin",
            "Number of Mugs": 54
        }, {
            "Customer": "Mohini Kaul",
            "Number of Mugs": 54
        }, {
            "Customer": "Nikolajs Kalejs",
            "Number of Mugs": 53
        }, {
            "Customer": "Dhaeraemdranaadh Allu",
            "Number of Mugs": 53
        }, {
            "Customer": "Lap Dinh",
            "Number of Mugs": 53
        }, {
            "Customer": "Kalidas Nadar",
            "Number of Mugs": 53
        }, {
            "Customer": "Elnaz Javan",
            "Number of Mugs": 52
        }, {
```

## CORRECTED RELATIONAL AND JSON OUTPUT (402 ROWS AFFECTED)



| | Customer | Number of Mugs |
|---|---|---|
| 1 | Duleep Walia | 63 |
| 2 | Viollette Monty | 60 |
| 3 | Matyas Macek | 59 |
| 4 | Yavuz Cetinkaya | 56 |
| 5 | Banshari De | 55 |
| 6 | Mohini Kaul | 54 |
| 7 | Kaan Tekin | 54 |
| 8 | Kalidas Nadar | 53 |
| 9 | Nikolajs Kalejs | 53 |
| 10 | Lap Dinh | 53 |
| 11 | Dhaeraemdranaadh Allu | 53 |
| 12 | An Dung Phung | 52 |
| 13 | Dayaram Mishra | 52 |
| 14 | Elnaz Javan | 52 |
| 15 | Dinara Saparkyzy | 51 |
| 16 | Alejandro Escobar | 51 |
| 17 | Shiva Pipalia | 51 |
| 18 | Debraj Sanyal | 50 |
| 19 | Bhaamini Kanaparthi | 50 |

Query executed successfully.



```
Dev Joke Mug: [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]
    [40]: [Object]
    [41]: [Object]
    [42]: [Object]
    [43]: [Object]
```

```
{
    "Dev Joke Mug": [{
            "Customer": "Duleep Walia",
            "Number of Mugs": 63
        }, {
            "Customer": "Viollette Monty",
            "Number of Mugs": 60
        }, {
            "Customer": "Matyas Macek",
            "Number of Mugs": 59
        }, {
            "Customer": "Yavuz Cetinkaya",
            "Number of Mugs": 56
        }, {
            "Customer": "Banshari De",
            "Number of Mugs": 55
        }, {
            "Customer": "Mohini Kaul",
            "Number of Mugs": 54
        }, {
            "Customer": "Kaan Tekin",
            "Number of Mugs": 54
        }, {
            "Customer": "Kalidas Nadar",
            "Number of Mugs": 53
        }, {
            "Customer": "Nikolajs Kalejs",
            "Number of Mugs": 53
        }, {
            "Customer": "Lap Dinh",
            "Number of Mugs": 53
        }, {
            "Customer": "Dhaeraemdranaadh Allu",
            "Number of Mugs": 53
        }, {
            "Customer": "An Dung Phung",
            "Number of Mugs": 52
        }, {
            "Customer": "Dayaram Mishra",
            "Number of Mugs": 52
        }, {
            "Customer": "Elnaz Javan"
```

## WORST #2: A [DIFFICULTY] QUERY

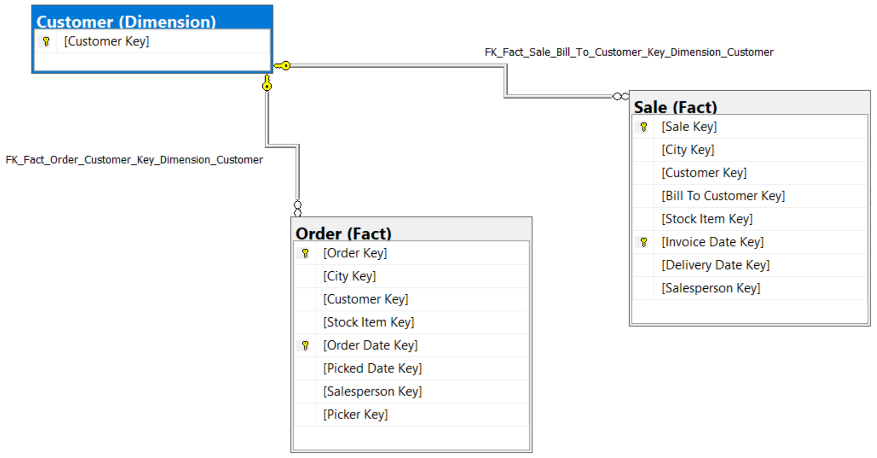
### PROBLEM STATEMENT

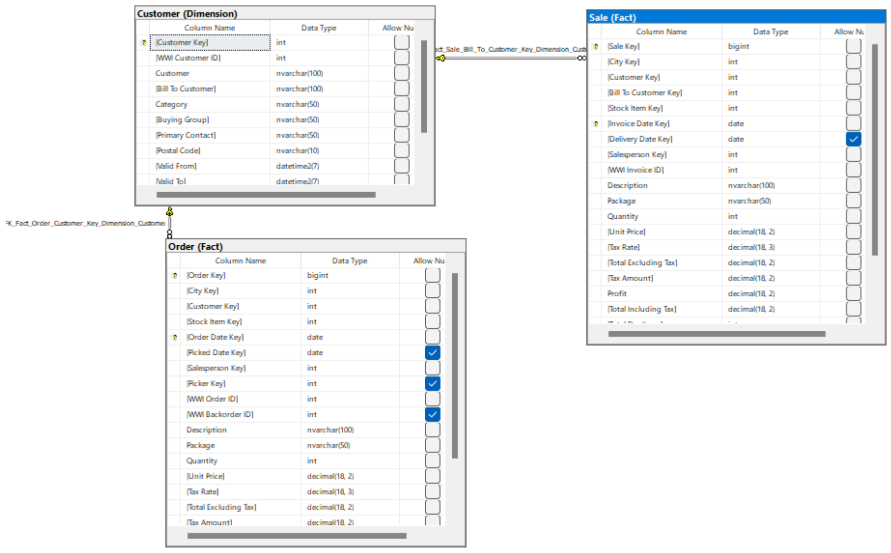
Find information about all the customers who purchased white mug/s in February using WideWorldImportersDW.

### REASON IT IS A WORST

Even though the query works, it doesn't take into consideration that the current REGEX operation can only look of the string 'mug' then 'white'. If there were to be any description where the description said "white mug' then we wouldn't get the desired result.

### KEY AND STANDARD VIEW OF TABLES USED




ct_Sale_Bill_To_Customer_Key_Dimension_Cust
'K_Fact_Order_Customer_Key_Dimension_Custome

**Customer (Dimension)**

| | Column Name | Data Type | Allow Nu |
|---|---|---|---|
| 🔑 | [Customer Key] | int | |
| | [WWI Customer ID] | int | |
| | Customer | nvarchar(100) | |
| | [Bill To Customer] | nvarchar(100) | |
| | Category | nvarchar(50) | |
| | [Buying Group] | nvarchar(50) | |
| | [Primary Contact] | nvarchar(50) | |
| | [Postal Code] | nvarchar(10) | |
| | [Valid From] | datetime2(7) | |
| | [Valid To] | datetime2(7) | |

**Sale (Fact)**

| | Column Name | Data Type | Allow Nu |
|---|---|---|---|
| 🔑 | [Sale Key] | bigint | |
| | [City Key] | int | |
| | [Customer Key] | int | |
| | [Bill To Customer Key] | int | |
| | [Stock Item Key] | int | |
| 🔑 | [Invoice Date Key] | date | |
| | [Delivery Date Key] | date | ✓ |
| | [Salesperson Key] | int | |
| | [WWI Invoice 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) | |
| | Profit | decimal(18, 2) | |
| | [Total Including Tax] | decimal(18, 2) | |

**Order (Fact)**

| | Column Name | Data Type | Allow Nu |
|---|---|---|---|
| 🔑 | [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) | |

TABLE SHOWING COLUMNS PROJECTED IN THE END

| Table Name | Column Name |
|---|---|
| Fact.[Order] | Quantity<br><br>Order Date Key |
| Dimension.Customer | Customer |
| Fact.Sale | Salesperson Key |

TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

| Table Name | Column Name | Sort Order |
|---|---|---|
| Fact.Sale | SalesPerson Key | ASC |
| Fact.[Order] | Order Date Key | ASC |

## QUERY OF WORST

```sql
USE WideWorldImportersDW;

SELECT S.[Salesperson Key] AS [Salesperson Key],
       C.[Customer],
       O.Quantity,
       O.[Order Date Key]
FROM Fact.[Order] AS O
    INNER JOIN Dimension.Customer AS C
        ON C.[Customer Key] = O.[Customer Key]
    INNER JOIN Fact.Sale AS S
        ON S.[Salesperson Key] = O.[Salesperson Key]
WHERE C.[Customer Key] != 0
      AND O.[Description] LIKE N'%mug%%white%'
      AND MONTH(O.[Order Date Key]) = 2
GROUP BY S.[Salesperson Key],
         C.[Customer],
         O.Quantity,
         O.[Order Date Key]
ORDER BY S.[Salesperson Key],

         O.[Order Date Key];
```

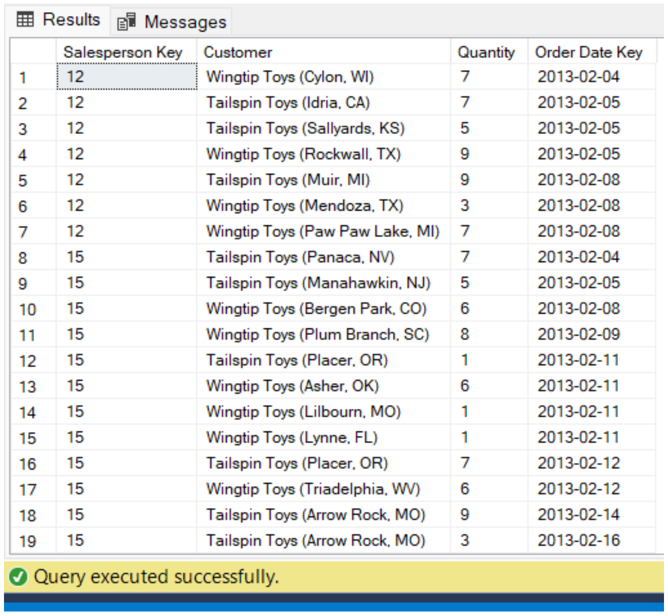
## QUERY OF WORST CORRECTED

```sql
USE WideWorldImportersDW;

SELECT S.[Salesperson Key] AS [Salesperson Key],
       C.[Customer],
       O.Quantity,
       O.[Order Date Key]
FROM Fact.[Order] AS O
    INNER JOIN Dimension.Customer AS C
        ON C.[Customer Key] = O.[Customer Key]
    INNER JOIN Fact.Sale AS S
        ON S.[Salesperson Key] = O.[Salesperson Key]
WHERE C.[Customer Key] != 0
      AND O.[Description] LIKE N'%mug%'
      AND O.[Description] LIKE N'%white%'
      AND MONTH(O.[Order Date Key]) = 2
GROUP BY S.[Salesperson Key],
         C.[Customer],
         O.Quantity,
         O.[Order Date Key]
ORDER BY S.[Salesperson Key],

         O.[Order Date Key];
```

## HOW IT WAS CORRECTED:

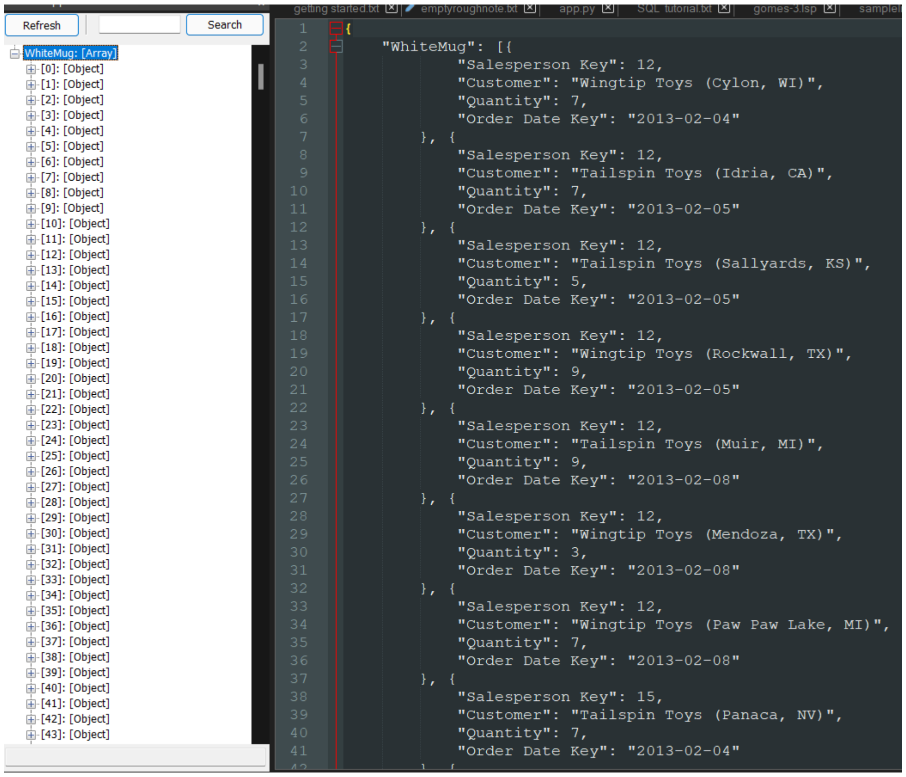
Separate REGEX for ‘white’ and ‘mug’ have been implemented to avoid exceptions.

## RELATIONAL AND JSON OUTPUT (1153 ROWS AFFECTED)



| | Salesperson Key | Customer | Quantity | Order Date Key |
|---|---|---|---|---|
| 1 | 12 | Wingtip Toys (Cylon, WI) | 7 | 2013-02-04 |
| 2 | 12 | Tailspin Toys (Idria, CA) | 7 | 2013-02-05 |
| 3 | 12 | Tailspin Toys (Sallyards, KS) | 5 | 2013-02-05 |
| 4 | 12 | Wingtip Toys (Rockwall, TX) | 9 | 2013-02-05 |
| 5 | 12 | Tailspin Toys (Muir, MI) | 9 | 2013-02-08 |
| 6 | 12 | Wingtip Toys (Mendoza, TX) | 3 | 2013-02-08 |
| 7 | 12 | Wingtip Toys (Paw Paw Lake, MI) | 7 | 2013-02-08 |
| 8 | 15 | Tailspin Toys (Panaca, NV) | 7 | 2013-02-04 |
| 9 | 15 | Tailspin Toys (Manahawkin, NJ) | 5 | 2013-02-05 |
| 10 | 15 | Wingtip Toys (Bergen Park, CO) | 6 | 2013-02-08 |
| 11 | 15 | Wingtip Toys (Plum Branch, SC) | 8 | 2013-02-09 |
| 12 | 15 | Tailspin Toys (Placer, OR) | 1 | 2013-02-11 |
| 13 | 15 | Wingtip Toys (Asher, OK) | 6 | 2013-02-11 |
| 14 | 15 | Wingtip Toys (Lilbourn, MO) | 1 | 2013-02-11 |
| 15 | 15 | Wingtip Toys (Lynne, FL) | 1 | 2013-02-11 |
| 16 | 15 | Tailspin Toys (Placer, OR) | 7 | 2013-02-12 |
| 17 | 15 | Wingtip Toys (Triadelphia, WV) | 6 | 2013-02-12 |
| 18 | 15 | Tailspin Toys (Arrow Rock, MO) | 9 | 2013-02-14 |
| 19 | 15 | Tailspin Toys (Arrow Rock, MO) | 3 | 2013-02-16 |

Query executed successfully.



```
{
    "WhiteMug": [{
            "Salesperson Key": 12,
            "Customer": "Wingtip Toys (Cylon, WI)",
            "Quantity": 7,
            "Order Date Key": "2013-02-04"
        }, {
            "Salesperson Key": 12,
            "Customer": "Tailspin Toys (Idria, CA)",
            "Quantity": 7,
            "Order Date Key": "2013-02-05"
        }, {
            "Salesperson Key": 12,
            "Customer": "Tailspin Toys (Sallyards, KS)",
            "Quantity": 5,
            "Order Date Key": "2013-02-05"
        }, {
            "Salesperson Key": 12,
            "Customer": "Wingtip Toys (Rockwall, TX)",
            "Quantity": 9,
            "Order Date Key": "2013-02-05"
        }, {
            "Salesperson Key": 12,
            "Customer": "Tailspin Toys (Muir, MI)",
            "Quantity": 9,
            "Order Date Key": "2013-02-08"
        }, {
            "Salesperson Key": 12,
            "Customer": "Wingtip Toys (Mendoza, TX)",
            "Quantity": 3,
            "Order Date Key": "2013-02-08"
        }, {
            "Salesperson Key": 12,
            "Customer": "Wingtip Toys (Paw Paw Lake, MI)",
            "Quantity": 7,
            "Order Date Key": "2013-02-08"
        }, {
            "Salesperson Key": 15,
            "Customer": "Tailspin Toys (Panaca, NV)",
            "Quantity": 7,
            "Order Date Key": "2013-02-04"
```

## WORST #3: A MEDIUM QUERY

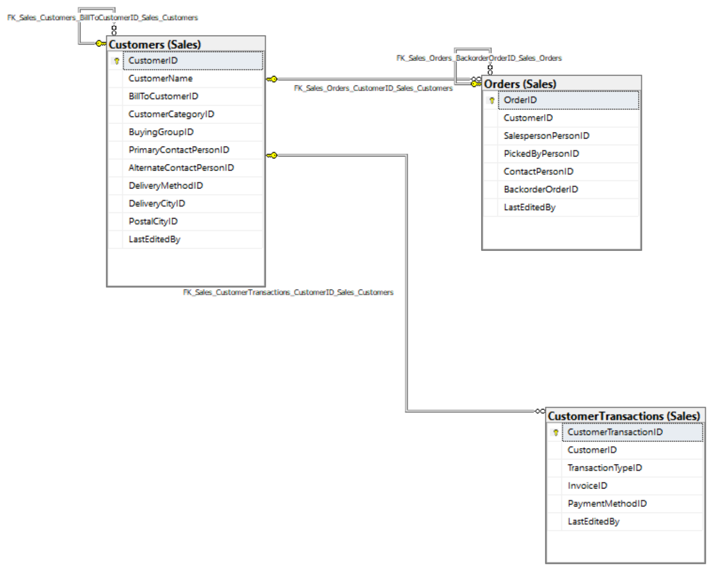
### PROBLEM STATEMENT

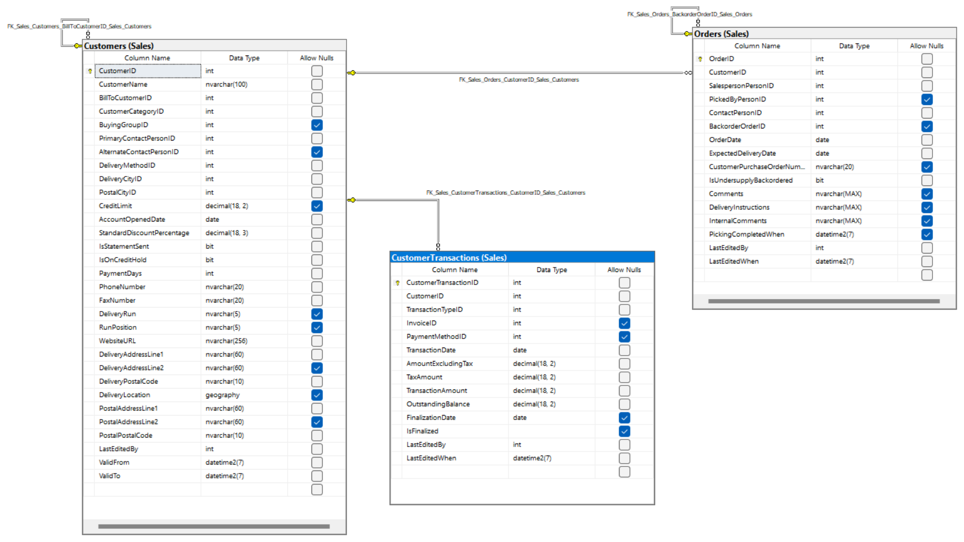
From WideWorldImporters find the regular customers who has average transaction amount above -5000 for the last 3 months of 2015.

### REASON IT IS A WORST

From the REGEX clauses it is not apparent why the Wingtip and Tailspin are being avoided.

### KEY AND STANDARD VIEW OF TABLES USED





FK_Sales_Customers_BillToCustomerID_Sales_Customers
FK_Sales_Orders_BackorderOrderID_Sales_Orders
FK_Sales_Orders_CustomerID_Sales_Customers
FK_Sales_CustomerTransactions_CustomerID_Sales_Customers

Customers (Sales)

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| CustomerID | int | ☐ |
| CustomerName | nvarchar(100) | ☐ |
| BillToCustomerID | int | ☐ |
| CustomerCategoryID | int | ☐ |
| BuyingGroupID | int | ✓ |
| PrimaryContactPersonID | int | ☐ |
| AlternateContactPersonID | int | ✓ |
| DeliveryMethodID | int | ☐ |
| DeliveryCityID | int | ☐ |
| PostalCityID | int | ☐ |
| CreditLimit | decimal(18, 2) | ✓ |
| AccountOpenedDate | date | ☐ |
| StandardDiscountPercentage | decimal(18, 3) | ☐ |
| IsStatementSent | bit | ☐ |
| IsOnCreditHold | bit | ☐ |
| PaymentDays | int | ☐ |
| PhoneNumber | nvarchar(20) | ☐ |
| FaxNumber | nvarchar(20) | ☐ |
| DeliveryRun | nvarchar(5) | ✓ |
| RunPosition | nvarchar(5) | ✓ |
| WebsiteURL | nvarchar(256) | ☐ |
| DeliveryAddressLine1 | nvarchar(60) | ☐ |
| DeliveryAddressLine2 | nvarchar(60) | ✓ |
| DeliveryPostalCode | nvarchar(10) | ☐ |
| DeliveryLocation | geography | ✓ |
| PostalAddressLine1 | nvarchar(60) | ☐ |
| PostalAddressLine2 | nvarchar(60) | ✓ |
| PostalPostalCode | nvarchar(10) | ☐ |
| LastEditedBy | int | ☐ |
| ValidFrom | datetime2(7) | ☐ |
| ValidTo | datetime2(7) | ☐ |
|  |  | ☐ |

CustomerTransactions (Sales)

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| CustomerTransactionID | int | ☐ |
| CustomerID | int | ☐ |
| TransactionTypeID | int | ☐ |
| InvoiceID | int | ✓ |
| PaymentMethodID | int | ✓ |
| TransactionDate | date | ☐ |
| AmountExcludingTax | decimal(18, 2) | ☐ |
| TaxAmount | decimal(18, 2) | ☐ |
| TransactionAmount | decimal(18, 2) | ☐ |
| OutstandingBalance | decimal(18, 2) | ☐ |
| FinalizationDate | date | ✓ |
| IsFinalized |  | ✓ |
| LastEditedBy | int | ☐ |
| LastEditedWhen | datetime2(7) | ☐ |
|  |  | ☐ |

Orders (Sales)

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| OrderID | int | ☐ |
| CustomerID | int | ☐ |
| SalespersonPersonID | int | ☐ |
| PickedByPersonID | int | ✓ |
| ContactPersonID | int | ☐ |
| BackorderOrderID | int | ✓ |
| OrderDate | date | ☐ |
| ExpectedDeliveryDate | date | ☐ |
| CustomerPurchaseOrderNum... | nvarchar(20) | ✓ |
| IsUndersupplyBackordered | bit | ☐ |
| Comments | nvarchar(MAX) | ✓ |
| DeliveryInstructions | nvarchar(MAX) | ✓ |
| InternalComments | nvarchar(MAX) | ✓ |
| PickingCompletedWhen | datetime2(7) | ✓ |
| LastEditedBy | int | ☐ |
| LastEditedWhen | datetime2(7) | ☐ |
|  |  | ☐ |

TABLE SHOWING COLUMNS PROJECTED IN THE END

| Table Name | Column Name |
|---|---|
| Sales.Customers | CustomerName |
| Derived | Average Transaction |

TABLE SHOWING HOW PROJECTION SORTED (IF APPLICABLE)

| Table Name | Column Name | Sort Order |
|---|---|---|
| Derived | Average Transaction | ASC |

## QUERY OF WORST

```
USE WideWorldImporters;

SELECT C.CustomerName AS [Customer Name],
       AVG(T.TransactionAmount) AS [Average Transaction]
FROM Sales.Customers AS C
    INNER JOIN Sales.Orders AS O
        ON C.CustomerID = O.CustomerID
    INNER JOIN Sales.CustomerTransactions AS T
        ON C.CustomerID = T.CustomerID
WHERE T.TaxAmount = 0
      AND O.OrderDate >= '20151001'
      AND O.OrderDate < '20160101'
      AND C.CustomerName NOT LIKE N'Wingtip%'
      AND C.CustomerName NOT LIKE N'Tailspin%'
GROUP BY C.CustomerName
HAVING AVG(T.TransactionAmount) > -5000
ORDER BY AVG(T.TransactionAmount);
```

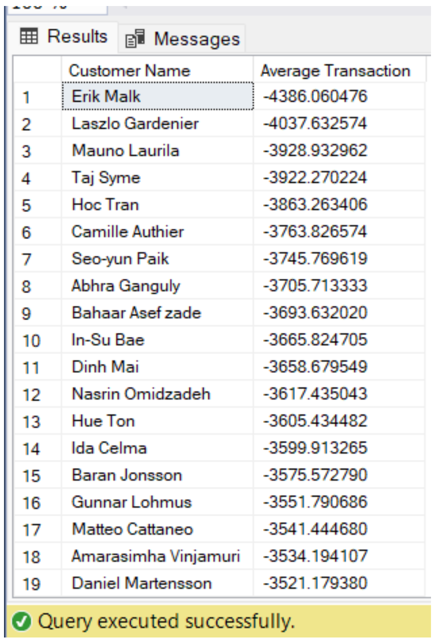
## QUERY OF WORST CORRECTED

```
USE WideWorldImporters;

SELECT C.CustomerName AS [Customer Name],
       AVG(T.TransactionAmount) AS [Average Transaction]
FROM Sales.Customers AS C
    INNER JOIN Sales.Orders AS O
        ON C.CustomerID = O.CustomerID
    INNER JOIN Sales.CustomerTransactions AS T
        ON C.CustomerID = T.CustomerID
WHERE T.TaxAmount = 0
      AND O.OrderDate >= '20151001'
      AND O.OrderDate < '20160101'
      AND LOWER(C.CustomerName)NOT LIKE N'toys%'
GROUP BY C.CustomerName
HAVING AVG(T.TransactionAmount) > -5000
ORDER BY AVG(T.TransactionAmount);
```

### HOW IT WAS CORRECTED:

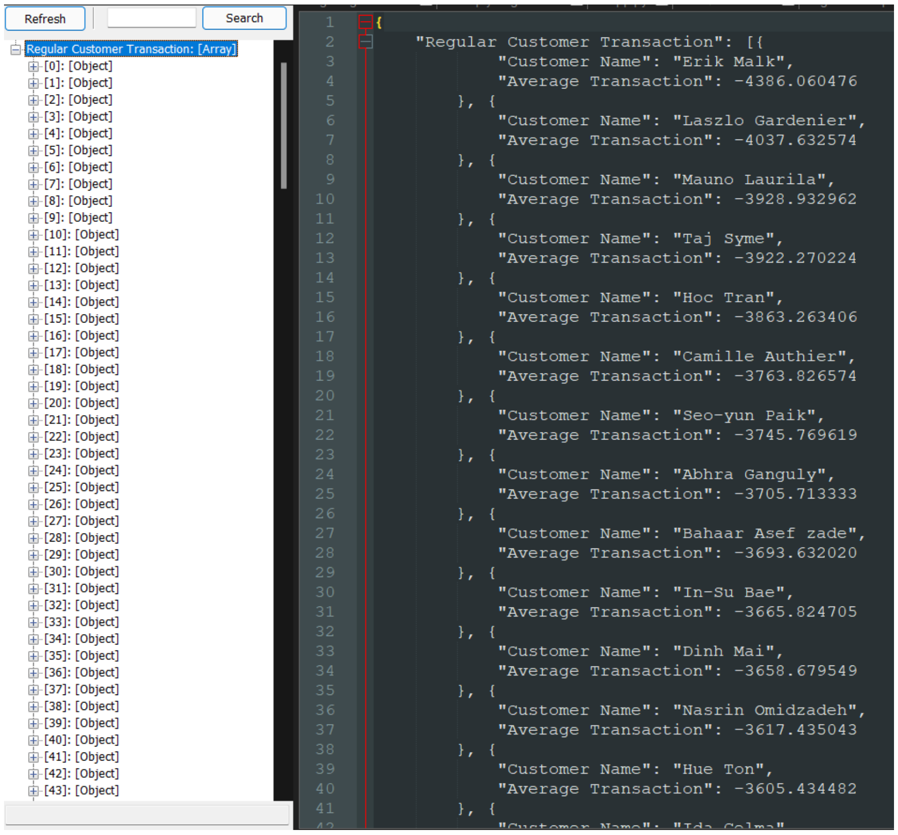
Apart from the toy stores all the other customers are regular customers. And it is logical to check if the customer is not a toy store than looking for each toy stores names individually. It also runs regex for one string rather than multiple, which is optimal.

## RELATIONAL AND JSON OUTPUT (255 ROWS AFFECTED)



|  | Customer Name | Average Transaction |
|---|---|---|
| 1 | Erik Malk | -4386.060476 |
| 2 | Laszlo Gardenier | -4037.632574 |
| 3 | Mauno Laurila | -3928.932962 |
| 4 | Taj Syme | -3922.270224 |
| 5 | Hoc Tran | -3863.263406 |
| 6 | Camille Authier | -3763.826574 |
| 7 | Seo-yun Paik | -3745.769619 |
| 8 | Abhra Ganguly | -3705.713333 |
| 9 | Bahaar Asef zade | -3693.632020 |
| 10 | In-Su Bae | -3665.824705 |
| 11 | Dinh Mai | -3658.679549 |
| 12 | Nasrin Omidzadeh | -3617.435043 |
| 13 | Hue Ton | -3605.434482 |
| 14 | Ida Celma | -3599.913265 |
| 15 | Baran Jonsson | -3575.572790 |
| 16 | Gunnar Lohmus | -3551.790686 |
| 17 | Matteo Cattaneo | -3541.444680 |
| 18 | Amarasimha Vinjamuri | -3534.194107 |
| 19 | Daniel Martensson | -3521.179380 |

Query executed successfully.



```
Regular Customer Transaction: [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]
  [40]: [Object]
  [41]: [Object]
  [42]: [Object]
  [43]: [Object]
```

```
{
    "Regular Customer Transaction": [{
            "Customer Name": "Erik Malk",
            "Average Transaction": -4386.060476
        }, {
            "Customer Name": "Laszlo Gardenier",
            "Average Transaction": -4037.632574
        }, {
            "Customer Name": "Mauno Laurila",
            "Average Transaction": -3928.932962
        }, {
            "Customer Name": "Taj Syme",
            "Average Transaction": -3922.270224
        }, {
            "Customer Name": "Hoc Tran",
            "Average Transaction": -3863.263406
        }, {
            "Customer Name": "Camille Authier",
            "Average Transaction": -3763.826574
        }, {
            "Customer Name": "Seo-yun Paik",
            "Average Transaction": -3745.769619
        }, {
            "Customer Name": "Abhra Ganguly",
            "Average Transaction": -3705.713333
        }, {
            "Customer Name": "Bahaar Asef zade",
            "Average Transaction": -3693.632020
        }, {
            "Customer Name": "In-Su Bae",
            "Average Transaction": -3665.824705
        }, {
            "Customer Name": "Dinh Mai",
            "Average Transaction": -3658.679549
        }, {
            "Customer Name": "Nasrin Omidzadeh",
            "Average Transaction": -3617.435043
        }, {
            "Customer Name": "Hue Ton",
            "Average Transaction": -3605.434482
        }, {
            "Customer Name": "Ida Celma"
```