TASK 1 – DATA DISCOVERY

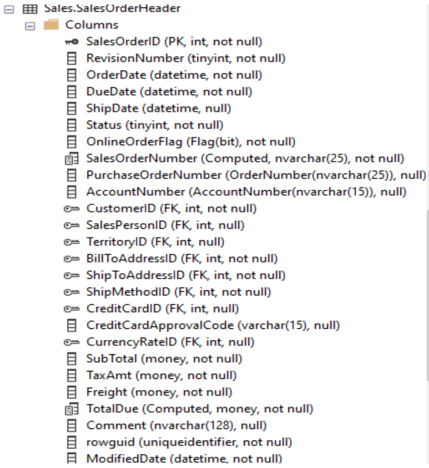
We focus solely on the sales process carried out in the Adventure Cycles company. To ease your task, please note that there are two main tables covering information about sale orders: SalesOrderHeader and SalesOrderDetail.

1. Identify main tables and attributes related to orders:
   i. Where the detailed information about orders is stored?
   ii. Are there different types of orders?
   iii. Are there different statuses of orders?
   iv. Which numerical data can be used to measure the performance of an order?
2. Identify main tables and attributes related to products.
   i. Which table(s) contain information related to products, like product's name?
   ii. Are products organized in some manner, like dairy products, clothes, etc.?
   iii. What additional information about products is available?
3. Identify main tables and attributes related to different types of customers.
   i. Are there different types of customers – name them?
4. Identify main tables and attributes related to employees handling orders.
5. Identify main tables and attributes related to sales locations – at what level of detail sales locations are defined.
6. How could you define sales in this database? Where is such sales data located?

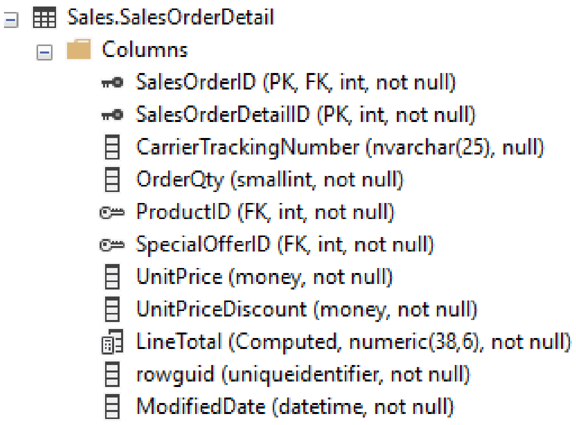
TASK 1 - SOLUTIONS:

1. *iDENTIFY MAIN TABLES AND ATTRIBUTES RELATED TO ORDERS:*

- *Below images are displaying the columns for the Sales.SalesOrderDetail and Sales.SalesOrderHeader tables in SQL Management Studio are provided for a thorough examination, which will be further explored with additional question related to tables and attributes itself.*



```
Sales.SalesOrderHeader
  Columns
    SalesOrderID (PK, int, not null)
    RevisionNumber (tinyint, not null)
    OrderDate (datetime, not null)
    DueDate (datetime, not null)
    ShipDate (datetime, null)
    Status (tinyint, not null)
    OnlineOrderFlag (Flag(bit), not null)
    SalesOrderNumber (Computed, nvarchar(25), not null)
    PurchaseOrderNumber (OrderNumber(nvarchar(25)), null)
    AccountNumber (AccountNumber(nvarchar(15)), null)
    CustomerID (FK, int, not null)
    SalesPersonID (FK, int, null)
    TerritoryID (FK, int, null)
    BillToAddressID (FK, int, not null)
    ShipToAddressID (FK, int, not null)
    ShipMethodID (FK, int, not null)
    CreditCardID (FK, int, null)
    CreditCardApprovalCode (varchar(15), null)
    CurrencyRateID (FK, int, null)
    SubTotal (money, not null)
    TaxAmt (money, not null)
    Freight (money, not null)
    TotalDue (Computed, money, not null)
    Comment (nvarchar(128), null)
    rowguid (uniqueidentifier, not null)
    ModifiedDate (datetime, not null)
```



```
Sales.SalesOrderDetail
  Columns
    SalesOrderID (PK, FK, int, not null)
    SalesOrderDetailID (PK, int, not null)
    CarrierTrackingNumber (nvarchar(25), null)
    OrderQty (smallint, not null)
    ProductID (FK, int, not null)
    SpecialOfferID (FK, int, not null)
    UnitPrice (money, not null)
    UnitPriceDiscount (money, not null)
    LineTotal (Computed, numeric(38,6), not null)
    rowguid (uniqueidentifier, not null)
    ModifiedDate (datetime, not null)
```

Both tables are frequently used in the AdventureWorks database to hold detailed order-related data.

I. Where the detailed information about orders is stored?

Item detailed information about each item included in a Sales Order is primarily stored in the SalesOrderDetail table. The SalesOrderDetail table contains, details about individual items in each order, multiple rows for each sales order, with each row representing a line item or product included in the order. The table plays a crucial role in tracking the details of sales order.

For instance:

SalesOrderID(int, Not Null): Unique ID. Establishes a relationship with the SalesOrderHeader table.

OrderQty(Smallint, Not Null): The quantity of the product ordered.

Product ID(int, Not Null): The identifier of the product included in the order.

Unit Price(Money, Not Null): The price per unit of the product.

UnitPriceDiscount(money, Not Null): Any discount applied to the unit price for the product.

General information about each sales order, is primarily stored in the SalesOrderHeader table. The table plays a crucial role in tracking the details of sales order The SalesOrderHeader table contains, a row for each Sales order and includes various attributes providing information about each order. It represents the header||summary level of the order.

For instance:

SalesOrderID(PK, int, Not Null): A unique identifier for each sales order.

CustomerID(int, Not Null): Customer identification number.

Order Date(datetime, Not Null): Date the order was received.

Ship Date(datetime, Not Null): Shipment date of the order.

Total due(Computed, money, Not Null): Total amount due.

Status(tinyint, Not null): Status of order (pending, shipped, cancelled)

II. Are there different types of orders?

III. Are there different statuses of orders?

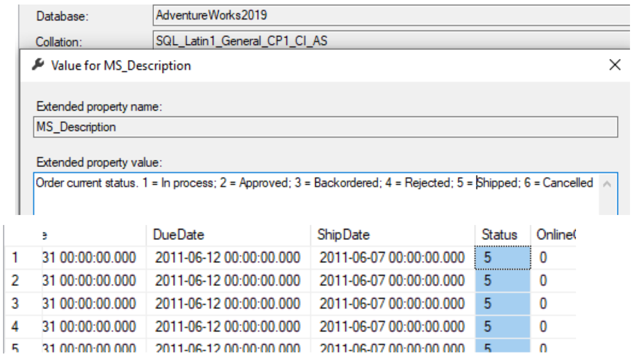
Yes! Within the database, orders have different statuses, signifying their current (position)stage within the progression of order processing and fullfillment workflow.
The current order processing status is indicated by the Status(tinyint, Not Null) column of the SalesOrderHeader table.

When accessing the extended properties for the "status" column, we observe the following values associated with it:

Order status.

1 = In process
2 = Approved
3 = Backordered
4 = Rejected
5 = Shipped
6 = Cancelled

Below image is the representation of the extended properties of status attribute in SalesOrderHeader table and example results from the ‘`Sales.SalesOrderHeader;`‘ *table that mostly indicates as the orders was shipped*



Database: AdventureWorks2019

Collation: SQL_Latin1_General_CP1_CI_AS

Extended property name: MS_Description

Extended property value: Order current status. 1 = In process; 2 = Approved; 3 = Backordered; 4 = Rejected; 5 = Shipped; 6 = Cancelled

	e	DueDate	ShipDate	Status	Online(
1	31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000	5	0
2	31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000	5	0
3	31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000	5	0
4	31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000	5	0
5	31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000	5	0

IV. Which numerical data can be used to measure the performance of an order?

Numerical attributes from the Sales.SalesOrderHeader and Sales.SalesOrderDetail tables are shown below which can be used for performance measurement of order.

<u>From Sales.SalesOrderHeader table:</u>

1. *TotalDue: The total amount due for the order, including product prices, taxes, and shipping fees. Allows to evaluate the total sales value of the order for overall profit and revenue performance.*
2. *SubTotal:The subtotal amount of the order, excluding taxes and shipping fees. Helps to find out revenue generated from the total sales without consideration of the taxes, shipping fees etc…*
3. *TaxAmt: The amount of tax applied to the order. Can be used to measure the total cost of order.*
4. *Freight: The freight or shipping charges for the order. The cost associated with the shipping an order to the customer, which can affect the overall profit of the order.*
5. *DiscountAmt: The total amount of discounts applied to the order, which can impact the overall profit and customer satisfaction.*

<u>From Sales.SalesOrderDetail table:</u>

1. *OrderQty: The quantity of each product ordered within the order. May indicate the volume of sales or demand for specific products.*
2. *UnitPrice: The price per unit of each product. Can be assessed to the revenue generated from each product sold within the order. Higher unit prices can indicate higher profit margins per product sold. Also, price changing may result with pricing strategies and customer preferences.*
3. *UnitPriceDiscount: If any discount applied to the unit price of each product. Analyzing discount applied to unit prices, may result with evaluating effectiveness of discount strategies in sales and attracting the customers.*
4. *LineTotal: The total amount for each product, calculated as follows: OrderQty * (UnitPrice - UnitPriceDiscount). Analyzing line total values can help to identify top-selling products and optimize product offerings based on sales performance.*

   <u>For instance:</u>
   This indicates that after accounting for all expenses (taxes, shipping charges and discounts) and the total cost associated with fulfilling the order, the net profit earned from the order.

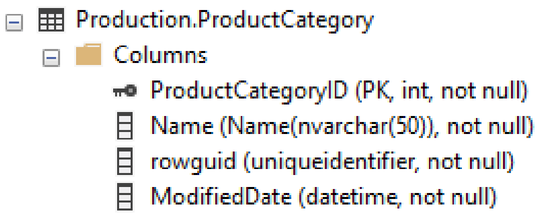
   Order Profit=(TotalDue − TaxAmt – Freight − DiscountAmt) − TotalCost

Businesses can learn more about a variety of order performance factors, such as revenue generation, profitability, sales volume, and discount effectiveness, by examining these numerical features from both tables.

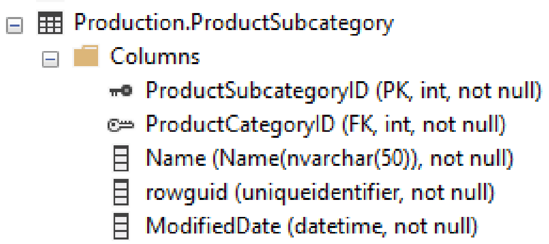
---

2. *IDENTIFY MAIN TABLES AND ATTRIBUTES RELATED TO PRODUCTS.*

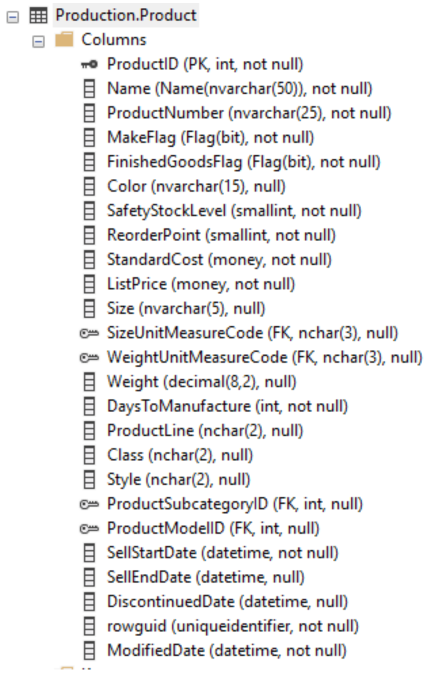
- *Below images are displaying the columns for the Production.Product, Production.ProductModel and Production.ProductCategory tables in SQL Management Studio are provided for a thorough examination, which will be further explored with additional question related to tables and attributes itself.*



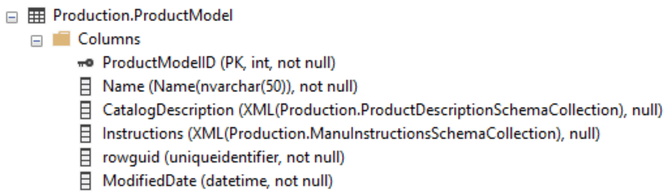
```
⊟ Production.ProductCategory
   ⊟ Columns
        ProductCategoryID (PK, int, not null)
        Name (Name(nvarchar(50)), not null)
        rowguid (uniqueidentifier, not null)
        ModifiedDate (datetime, not null)
```



```
⊟ Production.ProductSubcategory
   ⊟ Columns
        ProductSubcategoryID (PK, int, not null)
        ProductCategoryID (FK, int, not null)
        Name (Name(nvarchar(50)), not null)
        rowguid (uniqueidentifier, not null)
        ModifiedDate (datetime, not null)
```



```
⊟ Production.Product
   ⊟ Columns
        ProductID (PK, int, not null)
        Name (Name(nvarchar(50)), not null)
        ProductNumber (nvarchar(25), not null)
        MakeFlag (Flag(bit), not null)
        FinishedGoodsFlag (Flag(bit), not null)
        Color (nvarchar(15), null)
        SafetyStockLevel (smallint, not null)
        ReorderPoint (smallint, not null)
        StandardCost (money, not null)
        ListPrice (money, not null)
        Size (nvarchar(5), null)
        SizeUnitMeasureCode (FK, nchar(3), null)
        WeightUnitMeasureCode (FK, nchar(3), null)
        Weight (decimal(8,2), null)
        DaysToManufacture (int, not null)
        ProductLine (nchar(2), null)
        Class (nchar(2), null)
        Style (nchar(2), null)
        ProductSubcategoryID (FK, int, null)
        ProductModelID (FK, int, null)
        SellStartDate (datetime, not null)
        SellEndDate (datetime, null)
        DiscontinuedDate (datetime, null)
        rowguid (uniqueidentifier, not null)
        ModifiedDate (datetime, not null)
```



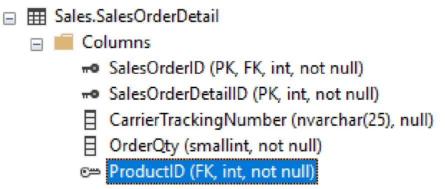
```
⊟ Production.ProductModel
   ⊟ Columns
        ProductModelID (PK, int, not null)
        Name (Name(nvarchar(50)), not null)
        CatalogDescription (XML(Production.ProductDescriptionSchemaCollection), null)
        Instructions (XML(Production.ManuInstructionsSchemaCollection), null)
        rowguid (uniqueidentifier, not null)
        ModifiedDate (datetime, not null)
```

All tables are frequently used in the AdventureWorks database to hold detailed product-related data.

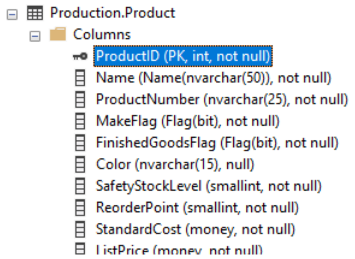
I. *Which table(s) contain information related to products, like product's name?*

Since in this task we intent to sorely focus on the order-related, sales process tables which are Sales.SalesOrderDetail and Sales.SalesOrderHeader, When these both tables are examined, the only way to access product-related information like product's name etc. is the ProductID attribute from the SalesOrderDetail table.

Below pictures are representation of Sales.SalesOrderDetail table of Foreign key(ProductID(FK, int, not null)) which refers to Production.product table attributes that contains details about each individual product.



```
⊟ Sales.SalesOrderDetail
   ⊟ Columns
        SalesOrderID (PK, FK, int, not null)
        SalesOrderDetailID (PK, int, not null)
        CarrierTrackingNumber (nvarchar(25), null)
        OrderQty (smallint, not null)
        ProductID (FK, int, not null)
```



```
⊟ Production.Product
   ⊟ Columns
        ProductID (PK, int, not null)
        Name (Name(nvarchar(50)), not null)
        ProductNumber (nvarchar(25), not null)
        MakeFlag (Flag(bit), not null)
        FinishedGoodsFlag (Flag(bit), not null)
        Color (nvarchar(15), null)
        SafetyStockLevel (smallint, not null)
        ReorderPoint (smallint, not null)
        StandardCost (money, not null)
        ListPrice (money, not null)
```

The table serves as the main repository for product information within the database.

<u>ProductID:</u>

The ProdactID in the SalesOrderDetail table access to product information through the Production.Product table. That's why the ProductID in the SalesOrderDetil table is associated with Production.Product table.

Production.Product table typically stores details about the product information, including the product's name, number, color, size, list price, model, category etc.…

Foreign keys, when associated with other tables; enable access to additional product information >> These FK specify model and subcategory of products. And allows access to more specific product attributes.

- *ProductModelID,*
- *ProductSubCategory(->ProductCategory with FK), > Are stored in separate tables. Becomes better to understand product to which categories and subcategories are belongs to.*

These FK indicates the weight and size of product. Gives more info about the physical characteristics of product.

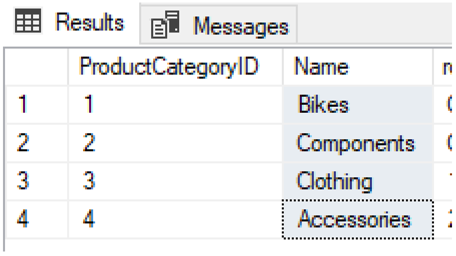
- *weightUnitMeasureCode,*
- *SizeUnitMeasureCode.*

<u>II. Are products organized in some manner, like dairy products, clothes, etc.?</u>

The data primarily provides a structured way to manage to analyze sales data related to the products sold. However, well-organized Production categories are:

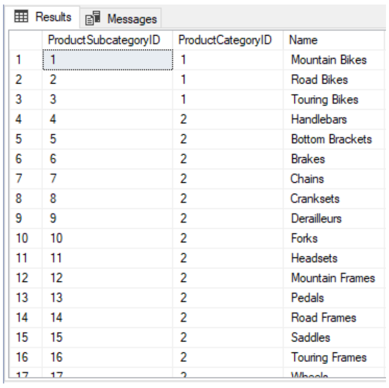
- *Below there are images of the category and subcategory of the Production table, contains Categories as,*

Production.ProductCategroy table



	ProductCategoryID	Name
1	1	Bikes
2	2	Components
3	3	Clothing
4	4	Accessories

Production.ProducSubCategory table

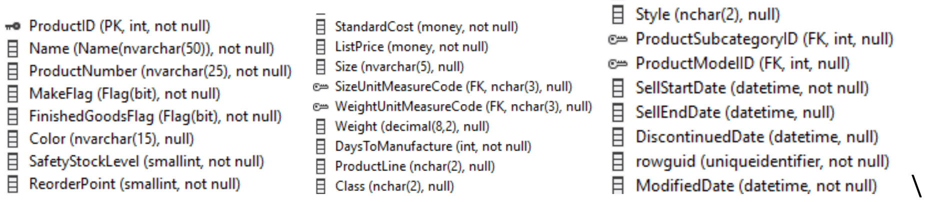


	ProductSubcategoryID	ProductCategoryID	Name
1	1	1	Mountain Bikes
2	2	1	Road Bikes
3	3	1	Touring Bikes
4	4	2	Handlebars
5	5	2	Bottom Brackets
6	6	2	Brakes
7	7	2	Chains
8	8	2	Cranksets
9	9	2	Derailleurs
10	10	2	Forks
11	11	2	Headsets
12	12	2	Mountain Frames
13	13	2	Pedals
14	14	2	Road Frames
15	15	2	Saddles
16	16	2	Touring Frames
17	17	2	Wheels

Through the Production.ProductCategory table;

products are organized in a certain order. Categories include defined categories such as 'Bikes', 'Components', 'Clothing' and 'Accessories'. These categories are structured to facilitate the shopping process by dividing products into some specific groups. Therefore, the database shows that the products are organized in a certain order."

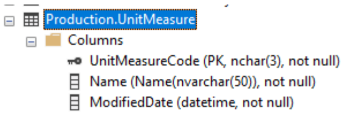
III. What additional information about products is available?
Production.Product table:
Details regarding various product attributes, as presented in the Product.product table image.



ProductID (PK, int, not null)
Name (Name(nvarchar(50)), not null)
ProductNumber (nvarchar(25), not null)
MakeFlag (Flag(bit), not null)
FinishedGoodsFlag (Flag(bit), not null)
Color (nvarchar(15), null)
SafetyStockLevel (smallint, not null)
ReorderPoint (smallint, not null)
StandardCost (money, not null)
ListPrice (money, not null)
Size (nvarchar(5), null)
SizeUnitMeasureCode (FK, nchar(3), null)
WeightUnitMeasureCode (FK, nchar(3), null)
Weight (decimal(8,2), null)
DaysToManufacture (int, not null)
ProductLine (nchar(2), null)
Class (nchar(2), null)
Style (nchar(2), null)
ProductSubcategoryID (FK, int, null)
ProductModelID (FK, int, null)
SellStartDate (datetime, not null)
SellEndDate (datetime, null)
DiscontinuedDate (datetime, null)
rowguid (uniqueidentifier, not null)
ModifiedDate (datetime, not null)

Since Foreign keys(FK) in the tables are presents relationship with another to product-related table based on a common attribute, categorizes the product and provides additional informations about the products. FK tables are;

SizeUnitMeasureCode(FK) and WeightUnitMeasureCode referencing the UnitMeasureCode(PK) in the Production.UnitMeasure.

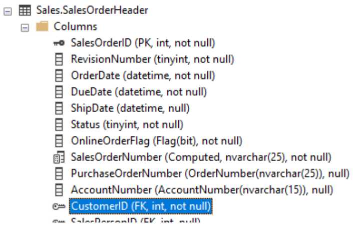


Production.UnitMeasure
Columns
UnitMeasureCode (PK, nchar(3), not null)
Name (Name(nvarchar(50)), not null)
ModifiedDate (datetime, not null)

3 IDENTIFY MAIN TABLES AND ATTRIBUTES RELATED TO DIFFERENT TYPES OF CUSTOMERS

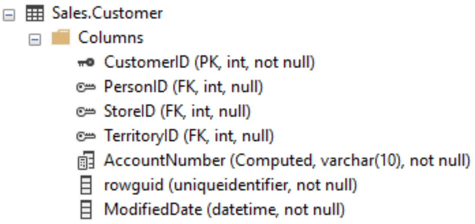
To identify the main tables and attributes related to different types of customers, We look at the tables that stores customer information and their associated attributes.

CustomerID(FK) referencing to Sales.Customer table where we can examine in detail



Sales.SalesOrderHeader
Columns
SalesOrderID (PK, int, not null)
RevisionNumber (tinyint, not null)
OrderDate (datetime, not null)
DueDate (datetime, not null)
ShipDate (datetime, null)
Status (tinyint, not null)
OnlineOrderFlag (Flag(bit), not null)
SalesOrderNumber (Computed, nvarchar(25), not null)
PurchaseOrderNumber (OrderNumber(nvarchar(25)), null)
AccountNumber (AccountNumber(nvarchar(15)), null)
CustomerID (FK, int, not null)
SalesPersonID (FK, int, null)

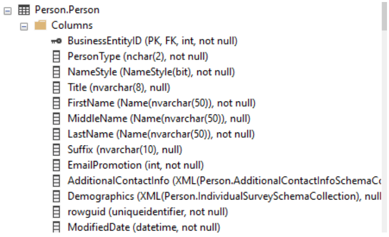
The Sales.Customer table stores information about the customers who made purchases.
The table contains each individual customers. And its connectivity with other table in database.



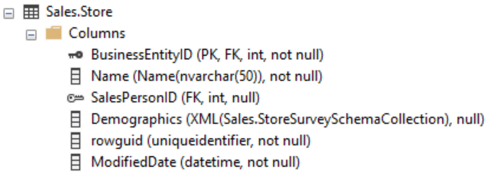
Sales.Customer
Columns
CustomerID (PK, int, not null)
PersonID (FK, int, null)
StoreID (FK, int, null)
TerritoryID (FK, int, null)
AccountNumber (Computed, varchar(10), not null)
rowguid (uniqueidentifier, not null)
ModifiedDate (datetime, not null)

PersonID: this column will reference to the Person.PersonID from the Person.Person table.
StoreID: if customer is associated with a any specific store, than attribute will be referencing to the Store.BusinessEntityID from the Sales.Store table.
TerritoryID: the column referencing the TerritoryID from the Sales.SalesTerritotry table

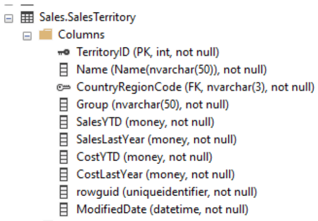
The images below are representation of Foreign keys from Sales.Customer table that its attributes are reference to.



- Person.Person
  - Columns
    - BusinessEntityID (PK, FK, int, not null)
    - PersonType (nchar(2), not null)
    - NameStyle (NameStyle(bit), not null)
    - Title (nvarchar(8), null)
    - FirstName (Name(nvarchar(50)), not null)
    - MiddleName (Name(nvarchar(50)), null)
    - LastName (Name(nvarchar(50)), not null)
    - Suffix (nvarchar(10), null)
    - EmailPromotion (int, not null)
    - AdditionalContactInfo (XML(Person.AdditionalContactInfoSchemaC
    - Demographics (XML(Person.IndividualSurveySchemaCollection), null
    - rowguid (uniqueidentifier, not null)
    - ModifiedDate (datetime, not null)



- Sales.Store
  - Columns
    - BusinessEntityID (PK, FK, int, not null)
    - Name (Name(nvarchar(50)), not null)
    - SalesPersonID (FK, int, null)
    - Demographics (XML(Sales.StoreSurveySchemaCollection), null)
    - rowguid (uniqueidentifier, not null)
    - ModifiedDate (datetime, not null)



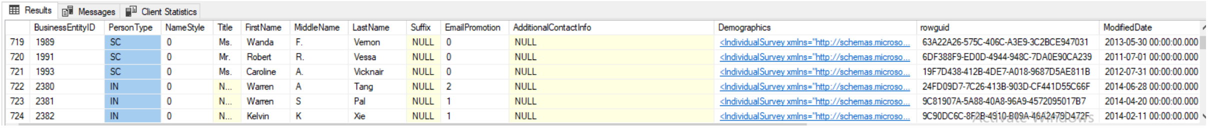
- Sales.SalesTerritory
  - Columns
    - TerritoryID (PK, int, not null)
    - Name (Name(nvarchar(50)), not null)
    - CountryRegionCode (FK, nvarchar(3), not null)
    - Group (nvarchar(50), not null)
    - SalesYTD (money, not null)
    - SalesLastYear (money, not null)
    - CostYTD (money, not null)
    - CostLastYear (money, not null)
    - rowguid (uniqueidentifier, not null)
    - ModifiedDate (datetime, not null)

Sales.SalesPerson table and Sales.Store table are related to other tables in context of sales transactions. Sales.SalesOrderHeader and Sales.SalesOrderDetail tables are linking sales transactions to salesperson and store them.
For example; SalesOrderHeader.SalesPersonID column references the SalesPerson.BusinessEntityID column, indicates the salesperson is actual responsible for the sale.
SalesOrderHeader.StoreID column references the Store.BusinessEntityID column. Also indicates that the store is associated with the sale.

These tables and attributes are providing informations about different types of customers, store-related customer

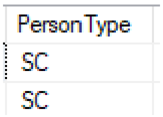
I. Are there different types of customers – name them?

There are two types of customers

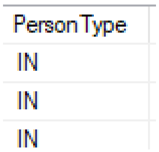


Results | Messages | Client Statistics

	BusinessEntityID	PersonType	NameStyle	Title	FirstName	MiddleName	LastName	Suffix	EmailPromotion	AdditionalContactInfo	Demographics	rowguid	ModifiedDate
719	1989	SC	0	Ms.	Wanda	F.	Vernon	NULL	0	NULL	<IndividualSurvey xmlns="http://schemas.microso...	63A22A26-575C-406C-A3E9-3C2BCE947031	2013-05-30 00:00:00.000
720	1991	SC	0	Mr.	Robert	R.	Vessa	NULL	0	NULL	<IndividualSurvey xmlns="http://schemas.microso...	6DF388F9-ED0D-4944-948C-7DA0E90CA239	2011-07-01 00:00:00.000
721	1993	SC	0	Ms.	Caroline	A.	Vicknair	NULL	0	NULL	<IndividualSurvey xmlns="http://schemas.microso...	19F7D438-412B-4DE7-A018-9687D5AE811B	2012-07-31 00:00:00.000
722	2380	IN	0	N...	Warren	A	Tang	NULL	2	NULL	<IndividualSurvey xmlns="http://schemas.microso...	24FD09D7-7C26-413B-903D-CF441D55C66F	2014-06-28 00:00:00.000
723	2381	IN	0	N...	Warren	S	Pal	NULL	1	NULL	<IndividualSurvey xmlns="http://schemas.microso...	9C81907A-5A88-40A8-96A9-4572095017B7	2014-04-20 00:00:00.000
724	2382	IN	0	N...	Kelvin	K	Xie	NULL	1	NULL	<IndividualSurvey xmlns="http://schemas.microso...	9C90DC6C-8F2B-4910-B09A-46A2479D472F	2014-02-11 00:00:00.000



PersonType
SC
SC



PersonType
IN
IN
IN

- two types of customers are represented as

1- Store Customer
2- Individual Customer

Since in the database there is not any specific column that labels the type of customer, we can infer different types based on available information that we got through the informations.

Individual Customers:

- *Rather than organizations, its typical individual customers.*

Business Customers:

- *Organizations or businesses rather than individual customers.*

WholeSale Customers:

- *Customers that purchase the product in large number of quantities with probably resale purpose. They may have specific pricing agreement and discount based on wholesale.*

Online Customers:

- *Who primarily make purchases through online app etc…*

Corporate Customers:

- *Business customers that are large companies. They might have requirement and wholesale process compared to small customers.*

Small Business Customers:

- *Business customers with needs and budgets. No large corporation.*

Repeat Customers:

- *Customer who makes multiple purchases. Maybe promotion to keep them.*

One-Time Customers:

- *Customer that one-time purchases the product and not return in future. They might be targeted by the company to have a different approach in*

The types of customers that indicated above are specific types of customers that may used in further process and classifications may vary depending on the context of business and detail of database.
By analyzing the customers, as a company we can target the customers and define the new market strategies for future organizations.

4 IDENTIFY MAIN TABLES AND ATTRIBUTES RELATED TO EMPLOYEES HANDLING ORDERS

In the database, tables and attributes are related to employees handling orders involves in terms of sales process.

There below main tables and associated attributes:

- *SalesPerson*

The SalesPerson table contains information specific to salespeople. It establishes a relationship with the Employee table and includes details such as email addresses etc…

- *Employee*

The table stores genera information about all the employees in the company which involved in handling orders. Includes details such as job titles, hire dates, and gender.

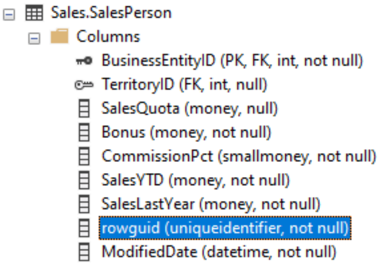
- *Person*

This table serves as a bridge between the transaction and other broader tables and is often combined with other tables, allowing more detailed queries for information. By storing personal details, enables various business processes and services in its database. it's particularly relevant for employees who interact with customers and manage orders.

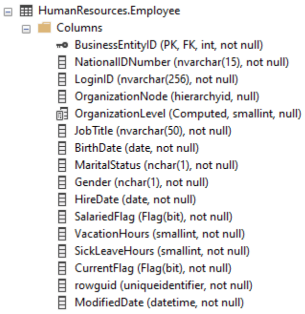
- *Address*

The Address table contains detailed information about physical addresses associated with employees and customers. Each address is uniquely identified by an AddressID. The table includes attributes such as street address (AddressLine1), city, state or province, and postal or ZIP code. These details are essential for order fulfillment, shipping, and billing processes. In the context of employees handling orders, addresses can be used to specify the location of the employee's workplace or contact information for communication purposes.

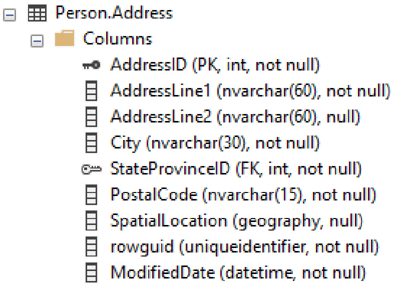
- *SalesOrderHeader*
- *SalesOrderDetail*



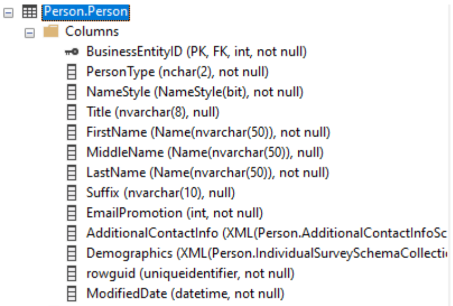
Sales.SalesPerson
Columns
BusinessEntityID (PK, FK, int, not null)
TerritoryID (FK, int, null)
SalesQuota (money, null)
Bonus (money, not null)
CommissionPct (smallmoney, not null)
SalesYTD (money, not null)
SalesLastYear (money, not null)
rowguid (uniqueidentifier, not null)
ModifiedDate (datetime, not null)



HumanResources.Employee
Columns
BusinessEntityID (PK, FK, int, not null)
NationalIDNumber (nvarchar(15), not null)
LoginID (nvarchar(256), not null)
OrganizationNode (hierarchyid, null)
OrganizationLevel (Computed, smallint, null)
JobTitle (nvarchar(50), not null)
BirthDate (date, not null)
MaritalStatus (nchar(1), not null)
Gender (nchar(1), not null)
HireDate (date, not null)
SalariedFlag (Flag(bit), not null)
VacationHours (smallint, not null)
SickLeaveHours (smallint, not null)
CurrentFlag (Flag(bit), not null)
rowguid (uniqueidentifier, not null)
ModifiedDate (datetime, not null)



Person.Address
Columns
AddressID (PK, int, not null)
AddressLine1 (nvarchar(60), not null)
AddressLine2 (nvarchar(60), null)
City (nvarchar(30), not null)
StateProvinceID (FK, int, not null)
PostalCode (nvarchar(15), not null)
SpatialLocation (geography, null)
rowguid (uniqueidentifier, not null)
ModifiedDate (datetime, not null)



Person.Person
Columns
BusinessEntityID (PK, FK, int, not null)
PersonType (nchar(2), not null)
NameStyle (NameStyle(bit), not null)
Title (nvarchar(8), null)
FirstName (Name(nvarchar(50)), not null)
MiddleName (Name(nvarchar(50)), null)
LastName (Name(nvarchar(50)), not null)
Suffix (nvarchar(10), null)
EmailPromotion (int, not null)
AdditionalContactInfo (XML(Person.AdditionalContactInfoSc
Demographics (XML(Person.IndividualSurveySchemaCollecti
rowguid (uniqueidentifier, not null)
ModifiedDate (datetime, not null)

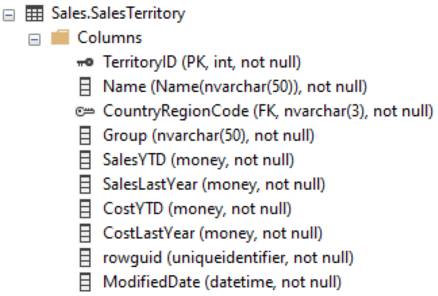
Can be even more categorized asHuman Resources:

- *EmployeeDepartmentHistory*
- *EmployeePayHistory*
- *SalesTerritory*
- *JobCandidate*
- *Department:Shift*

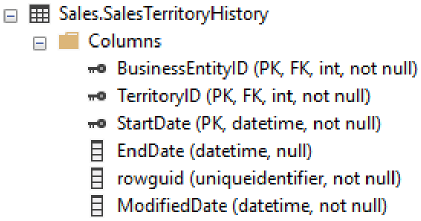
5 IDENTIFY MAIN TABLES AND ATTRIBUTES RELATED TO SALES lOCATIONS- AT WHAT LEVEL OF DETALS SALES LOCATIONS ARE DEFINED

- *Sales locations are defined at a high level of detail through the SalesTerritory table:*

Sales.SalesOrderHeader table TerritoryID refers to the SalesTerritory table.



Sales.SalesTerritory
Columns
TerritoryID (PK, int, not null)
Name (Name(nvarchar(50)), not null)
CountryRegionCode (FK, nvarchar(3), not null)
Group (nvarchar(50), not null)
SalesYTD (money, not null)
SalesLastYear (money, not null)
CostYTD (money, not null)
CostLastYear (money, not null)
rowguid (uniqueidentifier, not null)
ModifiedDate (datetime, not null)



Sales.SalesTerritoryHistory
Columns
BusinessEntityID (PK, FK, int, not null)
TerritoryID (PK, FK, int, not null)
StartDate (PK, datetime, not null)
EndDate (datetime, null)
rowguid (uniqueidentifier, not null)
ModifiedDate (datetime, not null)

Each sales territory represents region where the sales activities are managed. The attributes of the SalesTerritory table provide information about the territory's identifier, name, associated country or region(country state, province, city, district.), sales and cost data, and modification details.

The level of detail for defining sales locations can change depending on factors like market size, the sales structure of the organization, and the need for efficient sales management and analysis.

6 *HOW COULD YOU DEFINE SALES IN THIS DATABASE? wHERE IS SUCH SALES DATA LOCATED?*

Sales are defined as transactions where products are exchanged for monetary value. These transactions are characteristic of OLTP(Online Transaction Processing) relational databases, where daily business transaction operations are efficiently managed!

Since primary 2 main explained in detail in previously question with the provided images of:

- *SalesOrderHeader*
- *SalesOrderDetail*

By querying and analyzing these tables; By querying and analyzing these tables, we can gain a thorough understanding of the sales operations at database.

This includes details about products sold, customers served, and revenue generated. These transactions are integral to OLTP relational databases due to their role in facilitating daily business operations and managing sales processes efficiently.

Comments:

In the AdventureWorks OLTP (Online Transaction Processing) schema is a relational database that stores all the information about sales order placed by the customers. We solely focused on sales orders.

…

TASK 2 – SQL Queries

SQL warmup task. Please do not use the build-in query editor – focus on pure SQL statements! Using AdventureWorks database please prepare a series of SQL queries:

1. *Prepare a SQL query that provides information about the global order amount (money) of the AdventureWorks business. (Use SalesOrderHeader table).*
2. *Prepare a SQL query that provides information about the global sales amount (money) of the AdventureWorks business. (Use SalesOrderHeader table)*
3. *Prepare a SQL query that provides information about the global sales amount (money) and volume (items sold) of the AdventureWorks business. (Use SalesOrderHeader, SalesOrderDetails tables)*
4. *Prepare a SQL query that provides information about the sales amount in individual years of the AdventureWorks business. (Use SalesOrderHeader table)*
5. *Prepare a SQL query that provides the global profit of the business. (Use SalesOrderHeader, SalesOrderDetails and Product tables)*
    a. *Assume that product's cost (available in Product table) hasn't changed over time.*

All results should be additionally stored in a single text .sql file. After completing the task, double check your approach and please upload the file to ePortal. Note that the task allows for submission of multiple files.

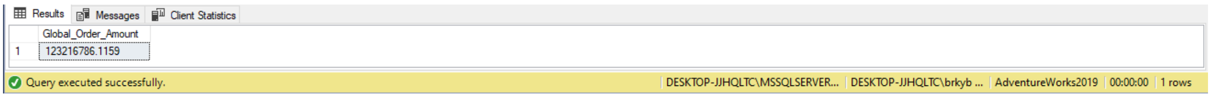
Task 2 - Solutions:

Use this section to provide your solutions, please remember to present an expert from the obtained results (several records, along with headers) and basic metadata (like number of rows), try to explain your approach:

1. *Prepare a SQL query that provides information about the global order amount (money) of the AdventureWorks business. (Use SalesOrderHeader table).*

```sql
SELECT SUM(TotalDue) AS Global_Order_Amount
FROM Sales.SalesOrderHeader;
```

Execution Results:

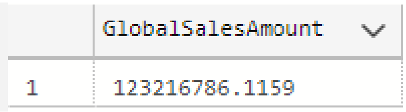


	Global_Order_Amount
1	123216786.1159

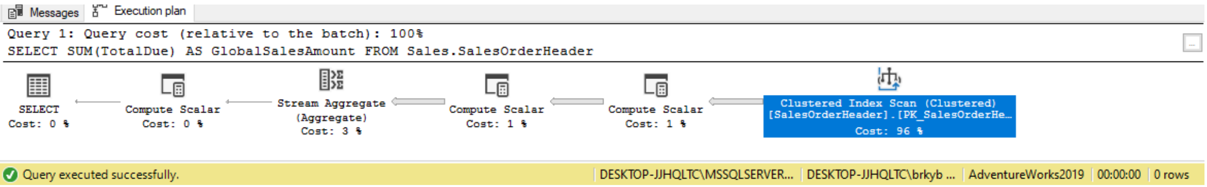
Query executed successfully. | DESKTOP-JJHQLTC\MSSQLSERVER... | DESKTOP-JJHQLTC\brkyb ... | AdventureWorks2019 | 00:00:00 | 1 rows

2. *Prepare a SQL query that provides information about the global sales amount (money) of the AdventureWorks business. (Use SalesOrderHeader table)*

```sql
SELECT SUM(TotalDue) AS GlobalSalesAmount
FROM Sales.SalesOrderHeader;
```

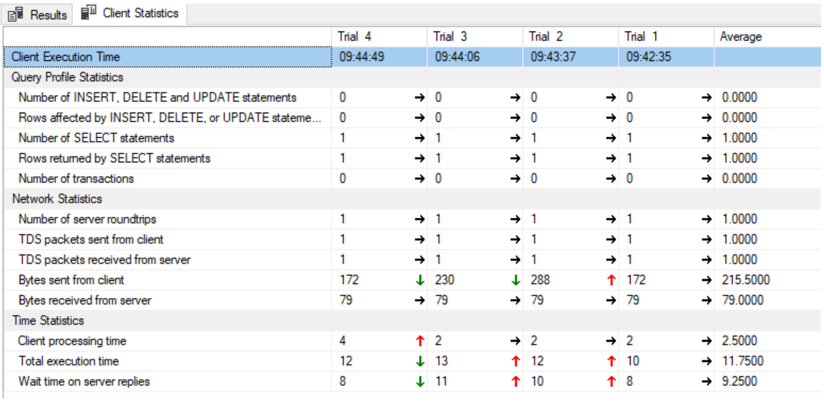


	GlobalSalesAmount
1	123216786.1159



```
Query 1: Query cost (relative to the batch): 100%
SELECT SUM(TotalDue) AS GlobalSalesAmount FROM Sales.SalesOrderHeader
```

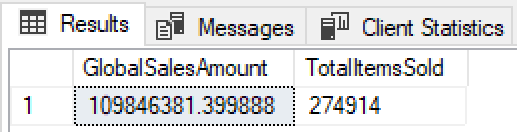
Query executed successfully. | DESKTOP-JJHQLTC\MSSQLSERVER... | DESKTOP-JJHQLTC\brkyb ... | AdventureWorks2019 | 00:00:00 | 0 rows



	Trial 4	Trial 3	Trial 2	Trial 1	Average
Client Execution Time	09:44:49	09:44:06	09:43:37	09:42:35	
Query Profile Statistics					
Number of INSERT, DELETE and UPDATE statements	0 →	0 →	0 →	0 →	0.0000
Rows affected by INSERT, DELETE, or UPDATE stateme...	0 →	0 →	0 →	0 →	0.0000
Number of SELECT statements	1 →	1 →	1 →	1 →	1.0000
Rows returned by SELECT statements	1 →	1 →	1 →	1 →	1.0000
Number of transactions	0 →	0 →	0 →	0 →	0.0000
Network Statistics					
Number of server roundtrips	1 →	1 →	1 →	1 →	1.0000
TDS packets sent from client	1 →	1 →	1 →	1 →	1.0000
TDS packets received from server	1 →	1 →	1 →	1 →	1.0000
Bytes sent from client	172 ↓	230 ↓	288 ↑	172 →	215.5000
Bytes received from server	79 →	79 →	79 →	79 →	79.0000
Time Statistics					
Client processing time	4 ↑	2 →	2 →	2 →	2.5000
Total execution time	12 ↓	13 ↑	12 ↑	10 →	11.7500
Wait time on server replies	8 ↓	11 ↑	10 ↑	8 →	9.2500

3. *Prepare a SQL query that provides information about the global sales amount (money) and volume (items sold) of the AdventureWorks business. (Use SalesOrderHeader, SalesOrderDetails tables).*

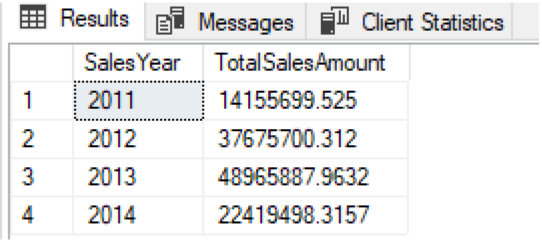
```sql
SELECT
    SUM(SOD.LineTotal) AS GlobalSalesAmount,
    SUM(SOD.OrderQty) AS TotalItemsSold
FROM
    Sales.SalesOrderHeader SOH
JOIN
    Sales.SalesOrderDetail SOD
        ON SOH.SalesOrderID = SOD.SalesOrderID;
```



	GlobalSalesAmount	TotalItemsSold
1	109846381.399888	274914

4. *Prepare a SQL query that provides information about the sales amount in individual years of the AdventureWorks business. (Use SalesOrderHeader table)*

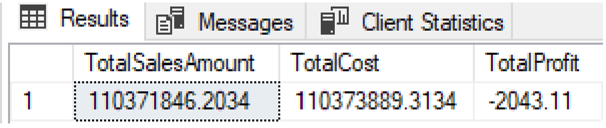
```
SELECT
    YEAR(OrderDate) AS SalesYear,
    SUM(TotalDue) AS TotalSalesAmount
FROM
    Sales.SalesOrderHeader
GROUP BY
    YEAR(OrderDate)
ORDER BY

    SalesYear;
```



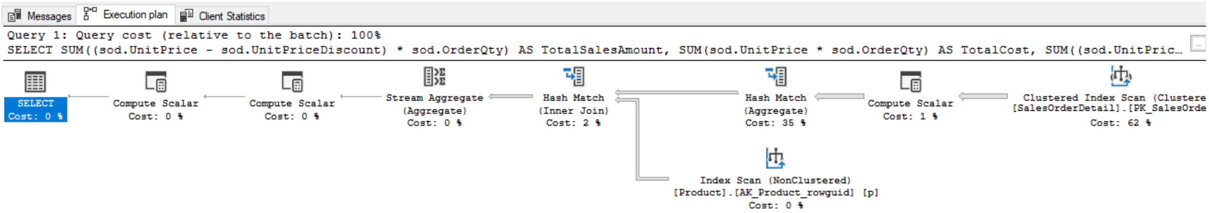
	SalesYear	TotalSalesAmount
1	2011	14155699.525
2	2012	37675700.312
3	2013	48965887.9632
4	2014	22419498.3157

5. *Prepare a SQL query that provides the global profit of the business. (Use SalesOrderHeader, SalesOrderDetails and Product tables)*

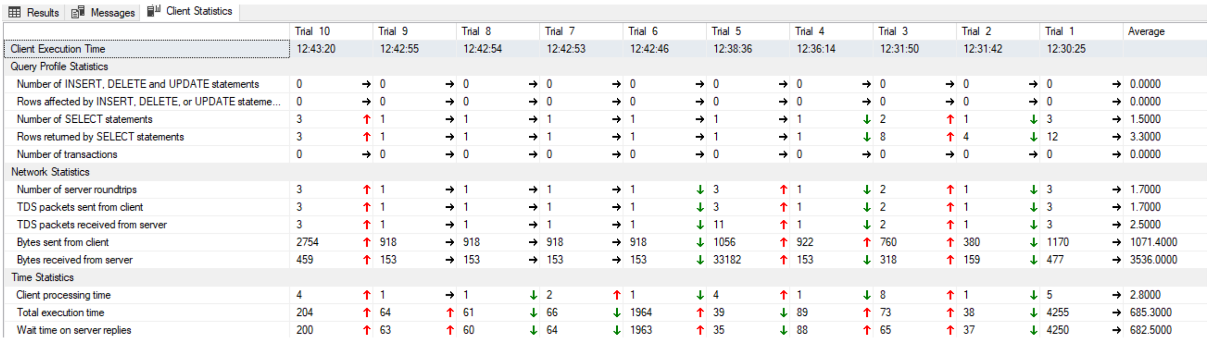
```
SELECT
    SUM((sod.UnitPrice - sod.UnitPriceDiscount) * sod.OrderQty) AS TotalSalesAmount,
    SUM(sod.UnitPrice * sod.OrderQty) AS TotalCost,
    SUM((sod.UnitPrice - sod.UnitPriceDiscount) * sod.OrderQty) - SUM(sod.UnitPrice *
sod.OrderQty) AS TotalProfit
FROM
    Sales.SalesOrderHeader soh
JOIN
    Sales.SalesOrderDetail sod ON soh.SalesOrderID = sod.SalesOrderID
JOIN

    Production.Product p ON sod.ProductID = p.ProductID;
```



	TotalSalesAmount	TotalCost	TotalProfit
1	110371846.2034	110373889.3134	-2043.11



```
Query 1: Query cost (relative to the batch): 100%
SELECT SUM((sod.UnitPrice - sod.UnitPriceDiscount) * sod.OrderQty) AS TotalSalesAmount, SUM(sod.UnitPrice * sod.OrderQty) AS TotalCost, SUM((sod.UnitPric…
```

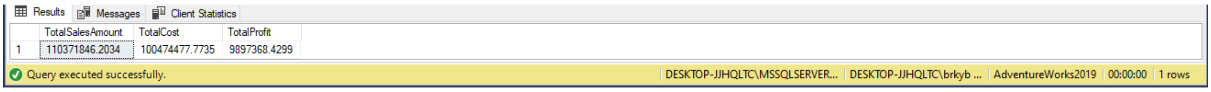


Results | Messages | Client Statistics

	Trial 10	Trial 9	Trial 8	Trial 7	Trial 6	Trial 5	Trial 4	Trial 3	Trial 2	Trial 1	Average
Client Execution Time	12:43:20	12:42:55	12:42:54	12:42:53	12:42:46	12:38:36	12:36:14	12:31:50	12:31:42	12:30:25	
Query Profile Statistics											
Number of INSERT, DELETE and UPDATE statements	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0.0000
Rows affected by INSERT, DELETE, or UPDATE stateme...	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0.0000
Number of SELECT statements	3 ↑	1 →	1 →	1 →	1 →	1 →	1 ↓	2 ↑	1 ↓	3 →	1.5000
Rows returned by SELECT statements	3 ↑	1 →	1 →	1 →	1 →	1 →	1 ↓	8 ↑	4 ↓	12 →	3.3000
Number of transactions	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0.0000
Network Statistics											
Number of server roundtrips	3 ↑	1 →	1 →	1 →	1 ↓	3 ↑	1 ↓	2 ↑	1 ↓	3 →	1.7000
TDS packets sent from client	3 ↑	1 →	1 →	1 →	1 ↓	3 ↑	1 ↓	2 ↑	1 ↓	3 →	1.7000
TDS packets received from server	3 ↑	1 →	1 →	1 →	1 ↓	11 ↑	1 ↓	2 ↑	1 ↓	3 →	2.5000
Bytes sent from client	2754 ↑	918 →	918 →	918 →	918 ↓	1056 ↑	922 ↑	760 ↑	380 ↓	1170 →	1071.4000
Bytes received from server	459 ↑	153 →	153 →	153 →	153 ↓	33182 ↑	153 ↓	318 ↑	159 ↓	477 →	3536.0000
Time Statistics											
Client processing time	4 ↑	1 →	1 ↓	2 ↑	1 ↓	4 ↑	1 ↓	8 ↑	1 ↓	5 →	2.8000
Total execution time	204 ↑	64 ↑	61 ↓	66 ↓	1964 ↑	39 ↓	89 ↑	73 ↑	38 ↓	4255 →	685.3000
Wait time on server replies	200 ↑	63 ↑	60 ↓	64 ↓	1963 ↑	35 ↓	88 ↑	65 ↑	37 ↓	4250 →	682.5000

a. Assume that product’s cost (available in Product table) hasn’t changed over time.

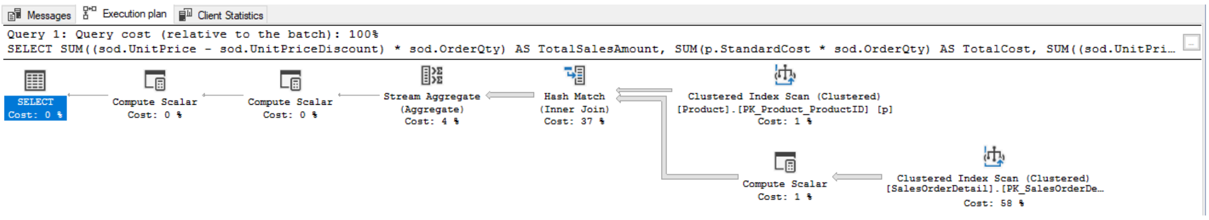
```
SELECT
    SUM((sod.UnitPrice - sod.UnitPriceDiscount) * sod.OrderQty) AS TotalSalesAmount,
    SUM(p.StandardCost * sod.OrderQty) AS TotalCost,
    SUM((sod.UnitPrice - sod.UnitPriceDiscount) * sod.OrderQty) - SUM(p.StandardCost *
sod.OrderQty) AS TotalProfit
FROM
    Sales.SalesOrderHeader soh
JOIN
    Sales.SalesOrderDetail sod ON soh.SalesOrderID = sod.SalesOrderID
JOIN
    Production.Product p ON sod.ProductID = p.ProductID;
```



Results | Messages | Client Statistics

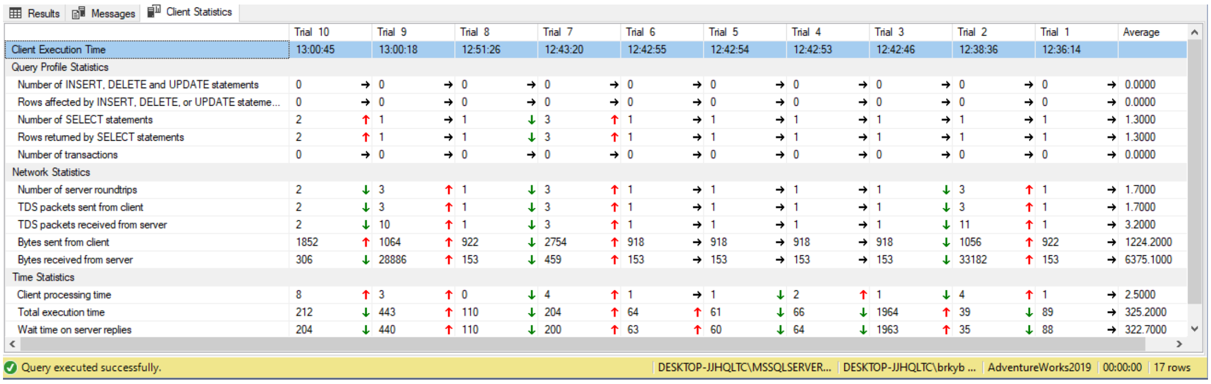
	TotalSalesAmount	TotalCost	TotalProfit
1	110371846.2034	100474477.7735	9897368.4299

Query executed successfully. | DESKTOP-JJHQLTC\MSSQLSERVER... | DESKTOP-JJHQLTC\brkyb ... | AdventureWorks2019 | 00:00:00 | 1 rows



Messages | Execution plan | Client Statistics

```
Query 1: Query cost (relative to the batch): 100%
SELECT SUM((sod.UnitPrice - sod.UnitPriceDiscount) * sod.OrderQty) AS TotalSalesAmount, SUM(p.StandardCost * sod.OrderQty) AS TotalCost, SUM((sod.UnitPri…
```



Results | Messages | Client Statistics

	Trial 10	Trial 9	Trial 8	Trial 7	Trial 6	Trial 5	Trial 4	Trial 3	Trial 2	Trial 1	Average
Client Execution Time	13:00:45	13:00:18	12:51:26	12:43:20	12:42:55	12:42:54	12:42:53	12:42:46	12:38:36	12:36:14	
Query Profile Statistics											
Number of INSERT, DELETE and UPDATE statements	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0.0000
Rows affected by INSERT, DELETE, or UPDATE stateme...	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0.0000
Number of SELECT statements	2 ↑	1 →	1 ↓	3 ↑	1 →	1 →	1 →	1 →	1 →	1 →	1.3000
Rows returned by SELECT statements	2 ↑	1 →	1 ↓	3 ↑	1 →	1 →	1 →	1 →	1 →	1 →	1.3000
Number of transactions	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0 →	0.0000
Network Statistics											
Number of server roundtrips	2 ↓	3 ↑	1 ↓	3 ↑	1 →	1 →	1 →	1 ↓	3 ↑	1 →	1.7000
TDS packets sent from client	2 ↓	3 ↑	1 ↓	3 ↑	1 →	1 →	1 →	1 ↓	3 ↑	1 →	1.7000
TDS packets received from server	2 ↓	10 ↑	1 ↓	3 ↑	1 →	1 →	1 →	1 ↓	11 ↑	1 →	3.2000
Bytes sent from client	1852 ↑	1064 ↑	922 ↓	2754 ↑	918 →	918 →	918 →	918 ↓	1056 ↑	922 →	1224.2000
Bytes received from server	306 ↓	28886 ↑	153 ↓	459 ↑	153 →	153 →	153 →	153 ↓	33182 ↑	153 →	6375.1000
Time Statistics											
Client processing time	8 ↑	3 ↑	0 ↓	4 ↑	1 →	1 ↓	2 ↑	1 ↓	4 ↑	1 →	2.5000
Total execution time	212 ↓	443 ↑	110 ↓	204 ↑	64 ↑	61 ↓	66 ↓	1964 ↑	39 ↓	89 →	325.2000
Wait time on server replies	204 ↓	440 ↑	110 ↓	200 ↑	63 ↑	60 ↓	64 ↓	1963 ↑	35 ↓	88 →	322.7000

Query executed successfully. | DESKTOP-JJHQLTC\MSSQLSERVER... | DESKTOP-JJHQLTC\brkyb ... | AdventureWorks2019 | 00:00:00 | 17 rows