Data Modeling for Superstore Database

Database Creation and Table Setup:

The first step was to create the "Superstore DB" database. Within this database, five essential tables were constructed using SQL queries: *Customers*, *Products*, *Geography*, *Orders*, and *Order Detail*.

SQLQuery2.sql -...AHMED\Ahmed (53))*

```sql
-- Create Customers Table
CREATE TABLE Customers (
    CustomerID VARCHAR(50) PRIMARY KEY,
    CustomerName VARCHAR(100),
    Segment VARCHAR(50)


-- Create Products Table
CREATE TABLE Products (
    ProductID VARCHAR(50),
    ProductName VARCHAR(255),
    Category VARCHAR(50),
    SubCategory VARCHAR(50),
    PRIMARY KEY (ProductID, ProductName)  -- Composite primary key


-- Create Geography Table
CREATE TABLE Geography (
    PostalCode VARCHAR(10) PRIMARY KEY,
    City VARCHAR(100),
    State VARCHAR(50),
    Country VARCHAR(50),
    Region VARCHAR(50)
```

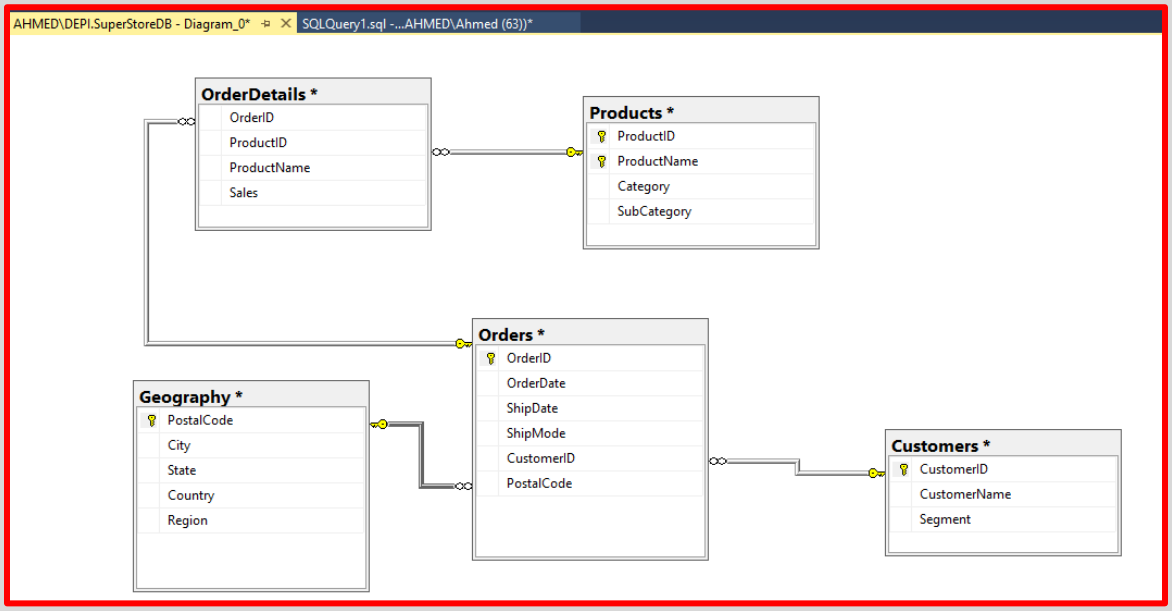
```sql
-- Create Orders Table
CREATE TABLE Orders (
    OrderID VARCHAR(50) PRIMARY KEY,
    OrderDate DATE,
    ShipDate DATE,
    ShipMode VARCHAR(50),
    CustomerID VARCHAR(50),
    PostalCode VARCHAR(10),
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),
    FOREIGN KEY (PostalCode) REFERENCES Geography(PostalCode)


-- Create OrderDetails Table
CREATE TABLE OrderDetails (
    OrderID VARCHAR(50),
    ProductID VARCHAR(50),
    ProductName VARCHAR(255),
    Sales DECIMAL(10, 2),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID, ProductName) REFERENCES Products(ProductID, ProductName)  -- Composite foreign key

~
```

Establishing Relationships and Diagram:

After creating the tables, the next phase involved linking them together to form a relational model. The relationships between the tables were established and visualized in a diagram, representing the connections between entities.



Data Import:

The dataset was then imported into the database. This data was added into the respective tables, adhering to the established relational structure.

Data Extraction and Loading:

Data was extracted from external sources and loaded into the relational tables using SQL queries. This ensured the accurate and structured integration of data into the database.

final inserting da...(AHMED\Ahmed (62))*

```
INSERT INTO Customers (CustomerID, CustomerName, Segment)
SELECT DISTINCT Customer_ID, Customer_Name, Segment
FROM Clean_11


INSERT INTO Geography (PostalCode, City, State, Country, Region)
SELECT DISTINCT Postal_Code, City, State, Country, Region
FROM Clean_11

INSERT INTO Products (ProductID, ProductName, Category, SubCategory)
SELECT DISTINCT Product_ID, Product_Name, Category, Sub_Category
FROM Clean_11

INSERT INTO Orders (OrderID, OrderDate, ShipDate, ShipMode, CustomerID, PostalCode)
SELECT DISTINCT Order_ID, Order_Date, Ship_Date, Ship_Mode, Customer_ID, Postal_Code
FROM Clean_11


INSERT INTO OrderDetails (OrderID, ProductID, ProductName, Sales)
SELECT DISTINCT Order_ID, Product_ID, Product_Name, Sales
FROM Clean_11
```

Insights and KPIs Extraction:

Using SQL queries, various insights and key performance indicators (KPIs) were extracted from the database. These insights provided valuable information for business analysis.

SQLQuery1.sql -...AHMED\Ahmed (53))*

```sql
--1 top 10 customer by total sale
SELECT TOP 10 Customers.CustomerID, SUM(OrderDetails.Sales) AS TotalSales
FROM Customers
JOIN Orders ON Customers.CustomerID = Orders.CustomerID
JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
GROUP BY Customers.CustomerID
ORDER BY TotalSales DESC;
```

100 %

Results | Messages

	CustomerID	TotalSales
1	SM-20320	25043.07
2	TC-20980	19052.22
3	RB-19360	15117.35
4	TA-21385	14595.62
5	AB-10105	14473.57
6	KL-16645	14175.23
7	SC-20095	14142.34
8	HL-15040	12873.30
9	SE-20110	12209.44
10	CC-12370	12129.08

SQLQuery1.sql -...AHMED\Ahmed (53))*

```sql
--2  customer segment generate highiset sales
SELECT Customers.Segment, SUM(OrderDetails.Sales) AS TotalSales
FROM Customers
JOIN Orders ON Customers.CustomerID = Orders.CustomerID
JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
GROUP BY Customers.Segment
ORDER BY TotalSales DESC;
```

100 %

Results | Messages

	Segment	TotalSales
1	Consumer	1148060.39
2	Corporate	688494.04
3	Home Office	424700.86

SQLQuery1.sql -...AHMED\Ahmed (53))*

```sql
--3 Regions with the highest customer count

SELECT Geography.Region, COUNT(DISTINCT Customers.CustomerID) AS CustomerCount
FROM Customers
JOIN Orders ON Customers.CustomerID = Orders.CustomerID
JOIN Geography ON Orders.PostalCode = Geography.PostalCode
GROUP BY Geography.Region
ORDER BY CustomerCount DESC;
```

100 %

Results | Messages

	Region	CustomerCount
1	West	681
2	East	669
3	Central	626
4	South	509

SQLQuery1.sql -...AHMED\Ahmed (53))*

```sql
--4 Unique customers who placed orders in each year

SELECT YEAR(Orders.OrderDate) AS OrderYear, COUNT(DISTINCT Customers.CustomerID) AS UniqueCustomers
FROM Customers
JOIN Orders ON Customers.CustomerID = Orders.CustomerID
GROUP BY YEAR(Orders.OrderDate)
ORDER BY OrderYear;
```

100 %

Results | Messages

	OrderYear	UniqueCustomers
1	2015	589
2	2016	567
3	2017	635
4	2018	690

```sql
--5  most sold product categories by total revenue

SELECT  Products.Category, SUM(OrderDetails.Sales) AS TotalRevenue
FROM Products
JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID
GROUP BY Products.Category
ORDER BY TotalRevenue DESC
```

100 %

Results | Messages

	Category	TotalRevenue
1	Technology	884735.18
2	Furniture	750406.85
3	Office Supplies	722640.75

SQLQuery1.sql -...AHMED\Ahmed (53))*

```sql
--6  Highest sales products in each region:

SELECT top 10 Geography.Region, Products.ProductName, SUM(OrderDetails.Sales) AS TotalSales
FROM Products
JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID
JOIN Orders ON OrderDetails.OrderID = Orders.OrderID
JOIN Geography ON Orders.PostalCode = Geography.PostalCode
GROUP BY Geography.Region, Products.ProductName
ORDER BY Geography.Region, TotalSales DESC
```

100 %

Results | Messages

	Region	ProductName	TotalSales
1	Central	Canon imageCLASS 2200 Advanced Copier	17499.95
2	Central	Lexmark MX611dhe Monochrome Laser Printer	14279.91
3	Central	Ibico EPK-21 Electric Binding System	11339.94
4	Central	GBC Ibimaster 500 Manual ProClick Binding System	10653.72
5	Central	GBC DocuBind P400 Electric Binding System	8710.33
6	Central	HON 5400 Series Task Chairs for Big and Tall	6939.70
7	Central	Fellowes PB500 Electric Punch Plastic Comb Bindi...	6100.75
8	Central	Canon PC1060 Personal Laser Copier	4899.93
9	Central	Mitel MiVoice 5330e IP Phone	4179.85
10	Central	Martin Yale Chadless Opener Electric Letter Opener	4164.05

SQLQuery1.sql -...AHMED\Ahmed (53))*

```sql
--6 Percentage of sales contributed by each product sub-category

SELECT Products.SubCategory,
       SUM(OrderDetails.Sales) * 100.0 / (SELECT SUM(Sales) FROM OrderDetails) AS PercentageOfSales
FROM Products
JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID
GROUP BY Products.SubCategory
```

100 %

Results | Messages

	SubCategory	PercentageOfSales
1	Supplies	2.052854
2	Storage	9.749377
3	Phones	15.667307
4	Fasteners	0.132755
5	Copiers	6.467561
6	Chairs	14.363196
7	Bookcases	5.604623
8	Machines	8.598890
9	Art	1.184579
10	Envelopes	0.713233
11	Binders	9.172350
12	Labels	0.546055
13	Furnishings	4.248649
14	Accessories	8.392082
15	Appliances	4.715488
16	Paper	3.690809
17	Tables	8.968941

SQLQuery1.sql -...AHMED\Ahmed (53))*

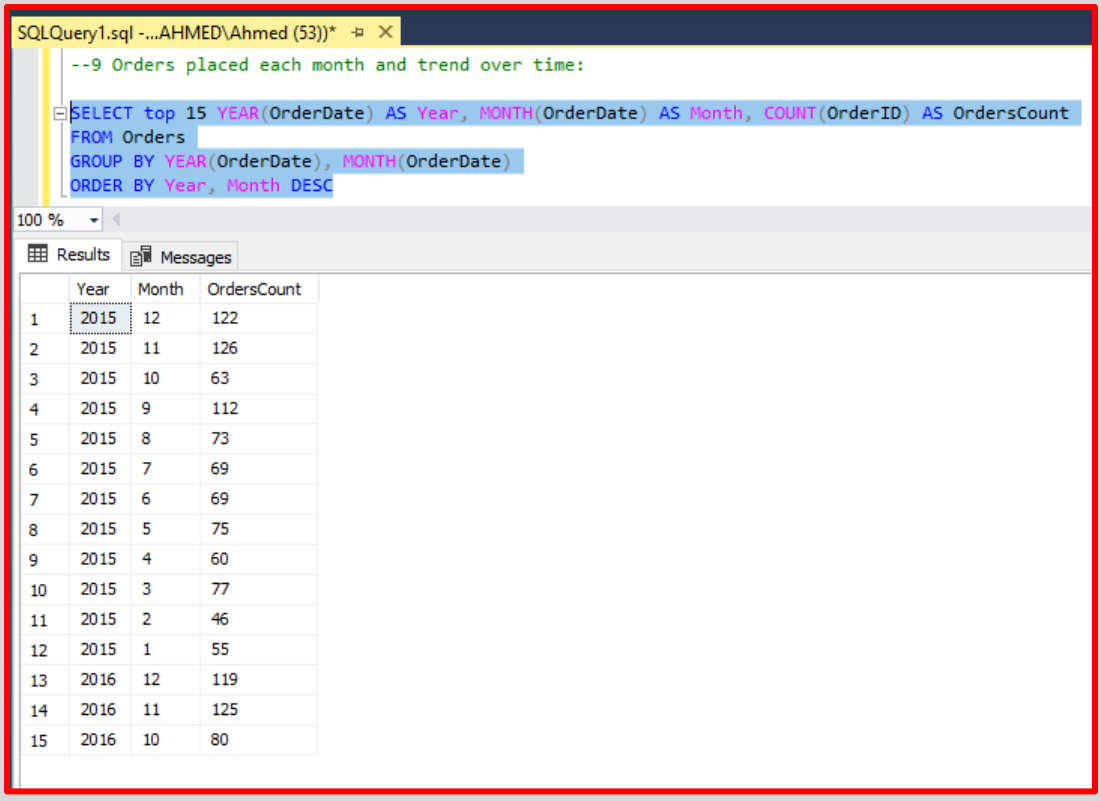
```sql
--8 Average time between order date and ship date by ship mode

SELECT ShipMode, AVG(DATEDIFF(DAY, Orders.OrderDate, Orders.ShipDate)) AS AvgShippingTime
FROM Orders
GROUP BY ShipMode
```

100 %

Results | Messages

	ShipMode	AvgShippingTime
1	First Class	7
2	Same Day	0
3	Standard Class	10
4	Second Class	3



SQLQuery1.sql -...AHMED\Ahmed (53))*

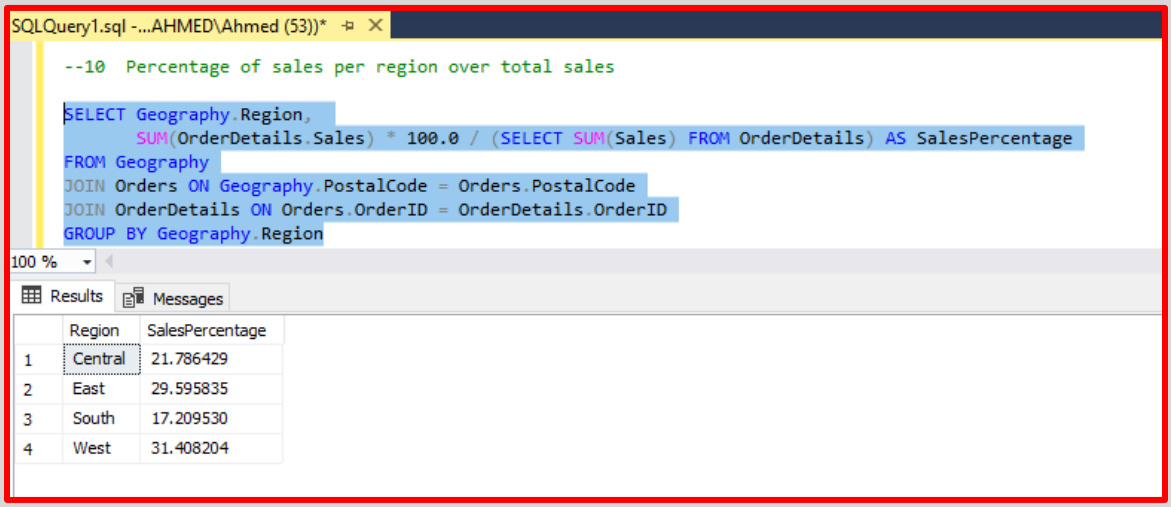
```sql
--9 Orders placed each month and trend over time:

SELECT top 15 YEAR(OrderDate) AS Year, MONTH(OrderDate) AS Month, COUNT(OrderID) AS OrdersCount
FROM Orders
GROUP BY YEAR(OrderDate), MONTH(OrderDate)
ORDER BY Year, Month DESC
```

100 %

Results | Messages

	Year	Month	OrdersCount
1	2015	12	122
2	2015	11	126
3	2015	10	63
4	2015	9	112
5	2015	8	73
6	2015	7	69
7	2015	6	69
8	2015	5	75
9	2015	4	60
10	2015	3	77
11	2015	2	46
12	2015	1	55
13	2016	12	119
14	2016	11	125
15	2016	10	80



SQLQuery1.sql -...AHMED\Ahmed (53))*

```sql
--10  Percentage of sales per region over total sales

SELECT Geography.Region,
       SUM(OrderDetails.Sales) * 100.0 / (SELECT SUM(Sales) FROM OrderDetails) AS SalesPercentage
FROM Geography
JOIN Orders ON Geography.PostalCode = Orders.PostalCode
JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
GROUP BY Geography.Region
```

100 %

Results | Messages

	Region	SalesPercentage
1	Central	21.786429
2	East	29.595835
3	South	17.209530
4	West	31.408204

Database Backup:

Finally, a complete backup of the database was performed to ensure data security and availability for future use.