DATA MINING LAB

Name : Uday Bolla
Roll No :22cs3024
Branch :Cse

Query for Creating Table and Inserting Values :

#Creating table

```
USE my_database;

CREATE TABLE FactSales (
    DateKey INT,
    ProductKey INT,
    CustomerKey INT,
    StoreKey INT,
    QtySold INT,
    SalesAmount DECIMAL(10, 2),
    DiscountAmount DECIMAL(10, 2),
    TaxAmount DECIMAL(10, 2),
    NetSalesAmount DECIMAL(10, 2)
);


CREATE TABLE DimProduct (
    ProductKey INT PRIMARY KEY,
    ProductName VARCHAR(100),
    Category VARCHAR(50),
    SubCategory VARCHAR(50),
    Brand VARCHAR(50),
    Price DECIMAL(10, 2)
);

CREATE TABLE DimCustomer (
    CustomerKey INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100),
    PhoneNo VARCHAR(15),
    Address VARCHAR(200),
    City VARCHAR(50),
    State VARCHAR(50),
    Country VARCHAR(50),
    LoyaltyLevel VARCHAR(50)
);
```

```
CREATE TABLE DimDate (
    DateKey INT PRIMARY KEY,
    FullDate DATE,
    Day INT,
    Month INT,
    Year INT,
    DayName VARCHAR(15),
    MonthName VARCHAR(15),
    Quarter INT
);

CREATE TABLE DimStore (
    StoreKey INT PRIMARY KEY,
    Region VARCHAR(50),
    ManagerName VARCHAR(50),
    OpeningDate DATE
);
```

Inserting Values

```
INSERT INTO DimProduct (ProductKey, ProductName, Category, SubCategory, Brand, Price) SELECT
1, 'Laptop', 'Electronics', 'Computers', 'BrandA', 1000.00
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM DimProduct WHERE ProductKey = 1);

INSERT INTO DimProduct (ProductKey, ProductName, Category, SubCategory, Brand, Price) SELECT
2, 'Smartphone', 'Electronics', 'Mobile Phones', 'BrandB', 500.00
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM DimProduct WHERE ProductKey = 2);

INSERT INTO DimProduct (ProductKey, ProductName, Category, SubCategory, Brand, Price) SELECT
3, 'Tablet', 'Electronics', 'Tablets', 'BrandC', 300.00
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM DimProduct WHERE ProductKey = 3);



INSERT INTO DimCustomer (CustomerKey, FirstName, LastName, Email, PhoneNo, Address, City, State, Country, LoyaltyLevel)
SELECT 1, 'John', 'Doe', 'john@example.com', '1234567890', '123 Main St', 'New York', 'NY', 'USA', 'Gold'
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM DimCustomer WHERE CustomerKey = 1);

INSERT INTO DimCustomer (CustomerKey, FirstName, LastName, Email, PhoneNo, Address, City, State, Country, LoyaltyLevel)
SELECT 2, 'Jane', 'Smith', 'jane@example.com', '0987654321', '456 Elm St', 'Los Angeles', 'CA', 'USA', 'Silver'
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM DimCustomer WHERE CustomerKey = 2);



INSERT INTO DimDate (DateKey, FullDate, Day, Month, Year, DayName, MonthName, Quarter) SELECT
20230101, '2023-01-01', 1, 1, 2023, 'Sunday', 'January', 1
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM DimDate WHERE DateKey = 20230101);
```

```
INSERT INTO DimDate (DateKey, FullDate, Day, Month, Year, DayName, MonthName, Quarter)
SELECT 20230102, '2023-01-02', 2, 1, 2023, 'Monday', 'January', 1
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM DimDate WHERE DateKey = 20230102);


INSERT INTO DimStore (StoreKey, Region, ManagerName, OpeningDate)
SELECT 1, 'North', 'Alice', '2020-01-01'
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM DimStore WHERE StoreKey = 1);

INSERT INTO DimStore (StoreKey, Region, ManagerName, OpeningDate)
SELECT 2, 'South', 'Bob', '2021-01-01'
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM DimStore WHERE StoreKey = 2);

INSERT INTO FactSales (DateKey, ProductKey, CustomerKey, StoreKey, QtySold, SalesAmount, DiscountAmount, TaxAmount,
NetSalesAmount)
SELECT 20230101, 1, 1, 1, 2, 2000.00, 100.00, 180.00, 1720.00
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM FactSales WHERE DateKey = 20230101 AND ProductKey = 1 AND CustomerKey = 1 AND
StoreKey = 1);

INSERT INTO FactSales (DateKey, ProductKey, CustomerKey, StoreKey, QtySold, SalesAmount, DiscountAmount, TaxAmount,
NetSalesAmount)
SELECT 20230102, 2, 2, 2, 3, 1500.00, 50.00, 135.00, 1315.00
FROM DUAL
WHERE NOT EXISTS (SELECT 1 FROM FactSales WHERE DateKey = 20230102 AND ProductKey = 2 AND CustomerKey = 2
AND StoreKey = 2);
```

1. Dimensions and Facts

1. **Create a fact table and at least three dimension tables for a sales dataset. Populate them with sample data.**
2. **Write SQL queries to calculate:**
    - **Total sales amount for all stores.**
    - **Sales amount for a specific product category.**

1. Dimensions and Facts

Create a fact table and at least three dimension tables for a sales dataset. Populate them with sample data.

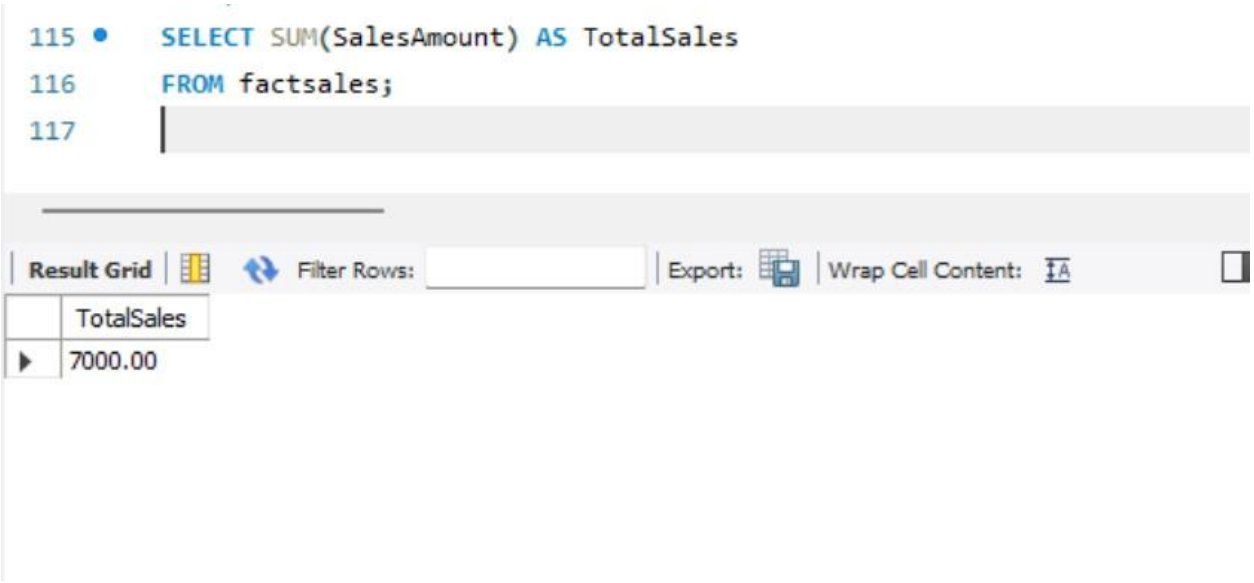
Write SQL queries to calculate:

Total sales amount for all stores.

Sales amount for a specific product category.

Query :

```
SELECT SUM(SalesAmount) AS TotalSales
FROM factsales;
```



2. Data Cubes

1. Create a data cube from sales data with dimensions: Product, Time, and Region. The measure is Sales Amount.
2. Write queries to calculate:
    - Total sales for all regions (aggregation across the Region dimension).
    - Sales per product category across all months.

2. Data Cubes

Create a data cube from sales data with dimensions: Product, Time, and Region. The measure is Sales Amount.
Write queries to calculate:
Total sales for all regions (aggregation across the Region dimension). Sales per product category across all months.

Query :

```
SELECT
   p.ProductName AS Product,
   d.MonthName AS Month,
   s.Region AS Region,
   SUM(f.SalesAmount) AS TotalSales
FROM
   FactSales f
JOIN
   DimProduct p ON f.ProductKey = p.ProductKey
JOIN
```

```
    DimDate d ON f.DateKey = d.DateKey
JOIN
    DimStore s ON f.StoreKey = s.StoreKey
GROUP BY
    p.ProductName, d.MonthName, s.Region
ORDER BY
    p.ProductName, d.MonthName, s.Region;
```

```
118     -- 2) Data Cubes
119     -- Create a data cube for Sales Amount with dimensions: Product, Time, and Re
120 •   SELECT
121         p.ProductName AS Product,
122         d.MonthName AS Month,
123         s.Region AS Region,
124         SUM(f.SalesAmount) AS TotalSales
125     FROM
126         FactSales f
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

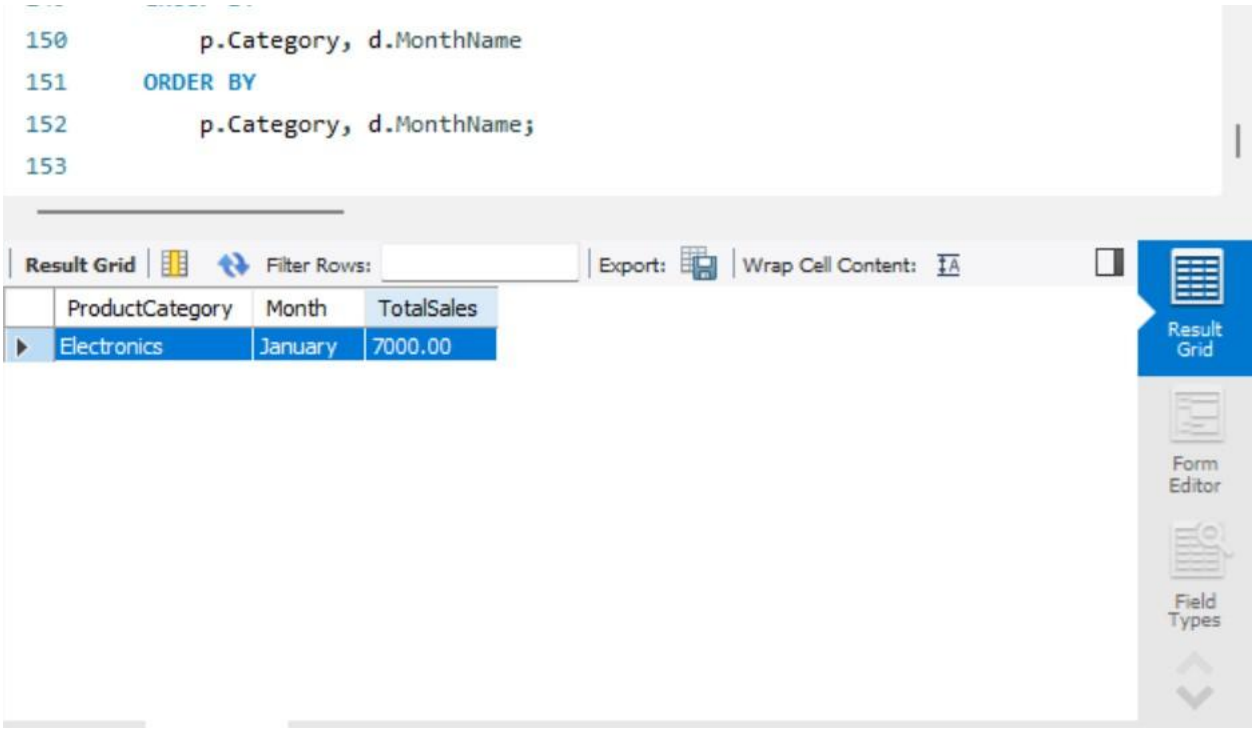
Product	Month	Region	TotalSales
Laptop	January	North	4000.00
Smartphone	January	South	3000.00

Result Grid | Form Editor | Field Types

Query

```
SELECT
    p.Category AS ProductCategory,
    d.MonthName AS Month,
    SUM(f.SalesAmount) AS TotalSales
FROM
    FactSales f
JOIN
    DimProduct p ON f.ProductKey = p.ProductKey
JOIN
    DimDate d ON f.DateKey = d.DateKey
GROUP BY
```

```
    p.Category, d.MonthName
ORDER BY
    p.Category, d.MonthName;
```



3. Hierarchies

1. Create a hierarchy for the Time dimension: Year > Quarter > Month > Day. Write queries to roll up and drill down sales data along this hierarchy.

2. Define a hierarchy for the Product dimension: Category > Sub-Category > Product. Analyze sales at each level.

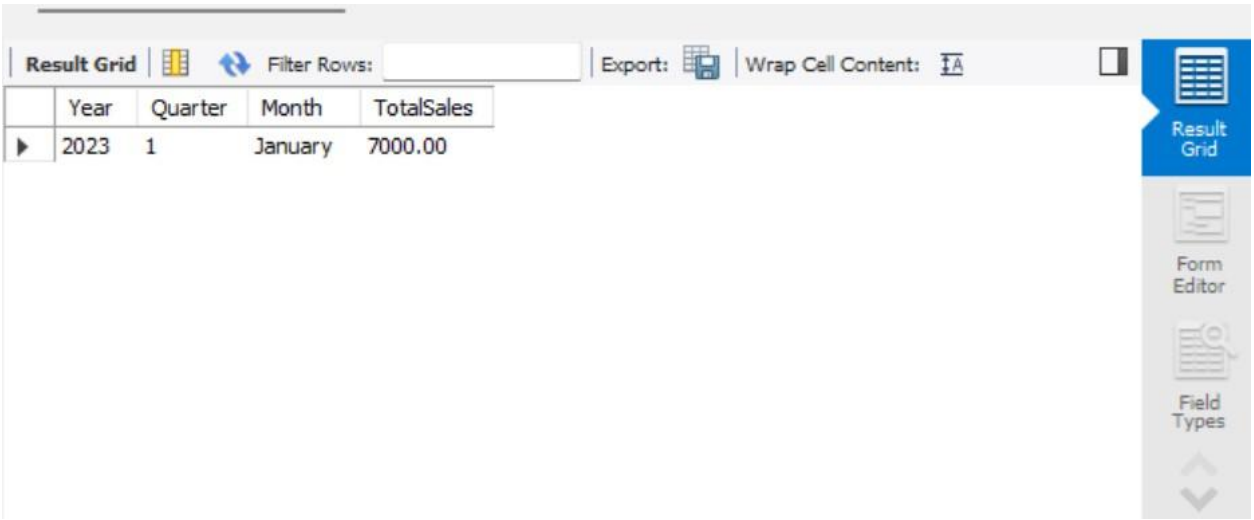
3. Hierarchies

1. Create a hierarchy for the `Time` dimension: `Year > Quarter > Month > Day`. Write queries to roll up and drill down sales data along this hierarchy.
2. Define a hierarchy for the `Product` dimension: `Category > Sub-Category > Product`. Analyze sales at each level.

1 - Roll Up

SELECT

```
    d.Year AS Year,
    d.Quarter AS Quarter,
    d.MonthName AS Month,
    SUM(f.SalesAmount) AS TotalSales
FROM
    FactSales f
JOIN
    DimDate d ON f.DateKey = d.DateKey
GROUP BY
    d.Year, d.Quarter, d.MonthName
ORDER BY
    d.Year, d.Quarter, d.MonthName;
```

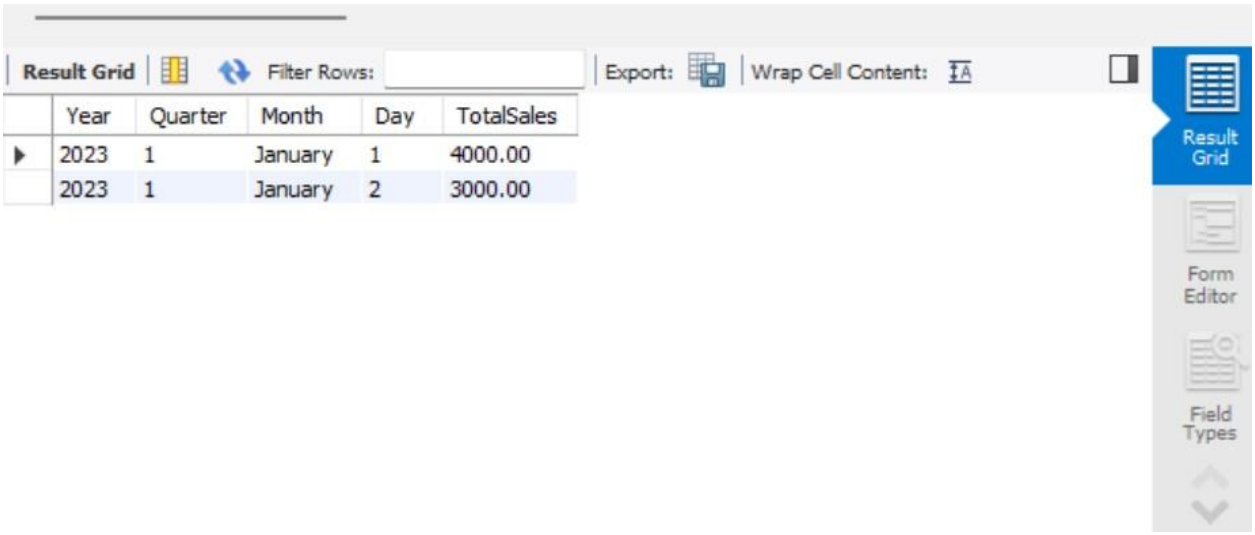


6. Roll Up and Drill Down

1. Roll up sales data from Day to Month and then to Year. Analyze how the aggregation changes.
2. Drill down from Region to City and analyze sales trends.

1-Drill Down

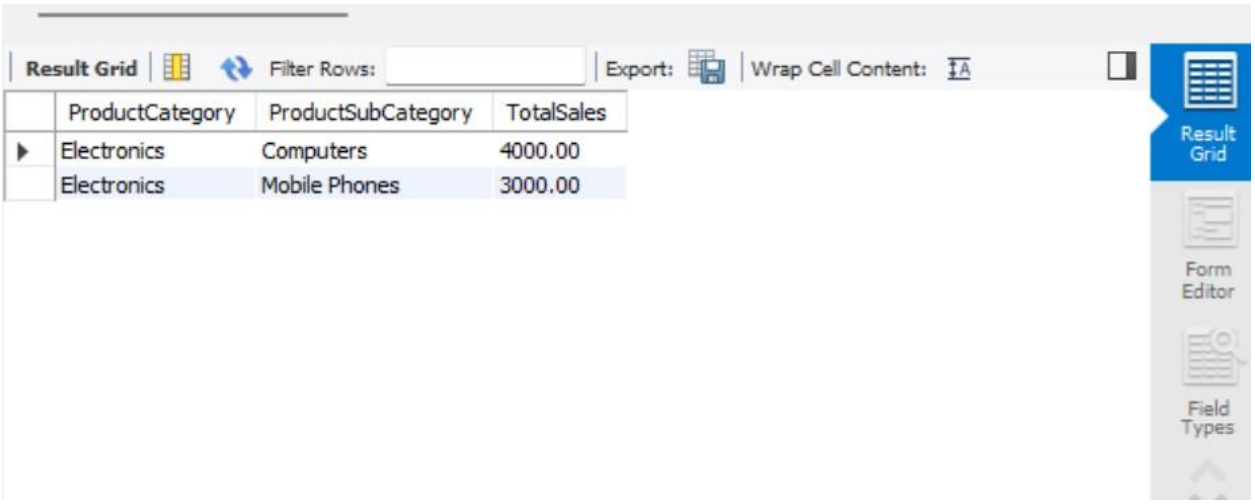
```
SELECT
    d.Year AS Year,
    d.Quarter AS Quarter,
    d.MonthName AS Month,
    d.Day AS Day,
    SUM(f.SalesAmount) AS TotalSales
FROM
    FactSales f
JOIN
    DimDate d ON f.DateKey = d.DateKey
GROUP BY
    d.Year, d.Quarter, d.MonthName, d.Day
ORDER BY
    d.Year, d.Quarter, d.MonthName, d.Day;
```



Year	Quarter	Month	Day	TotalSales
2023	1	January	1	4000.00
2023	1	January	2	3000.00

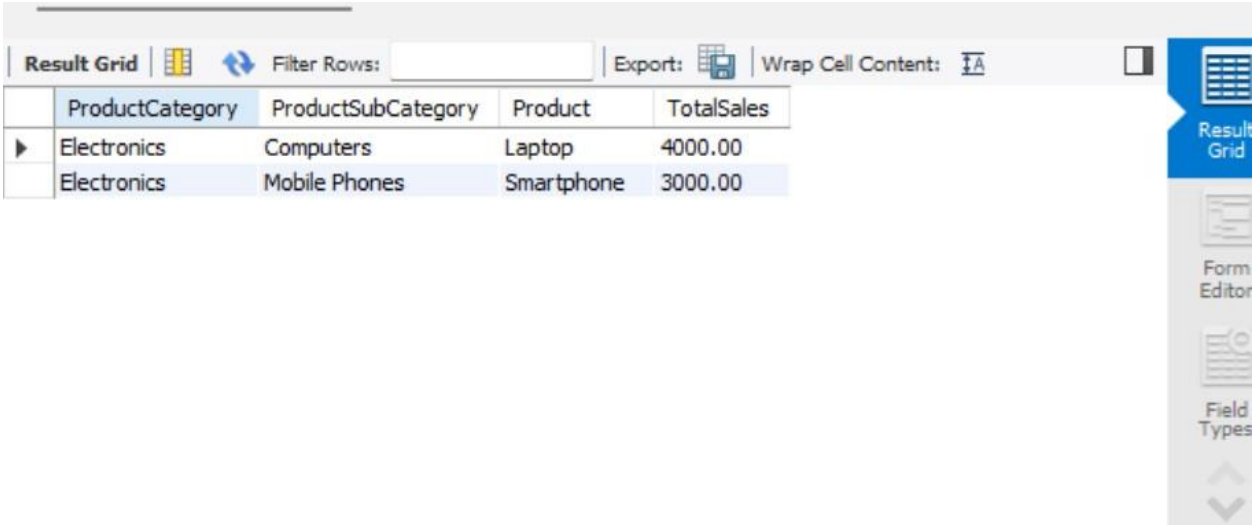
2 - Roll Up

```
SELECT
   p.Category AS ProductCategory,
   p.SubCategory AS ProductSubCategory,
   SUM(f.SalesAmount) AS TotalSales
FROM
   FactSales f
JOIN
   DimProduct p ON f.ProductKey = p.ProductKey
GROUP BY
   p.Category, p.SubCategory
ORDER BY
   p.Category, p.SubCategory;
```



2 - Drill Down

```
SELECT
   p.Category AS ProductCategory,
   p.SubCategory AS ProductSubCategory,
   p.ProductName AS Product,     SUM(f.SalesAmount) AS
TotalSales FROM
   FactSales f
JOIN
   DimProduct p ON f.ProductKey = p.ProductKey
GROUP BY
   p.Category, p.SubCategory, p.ProductName
ORDER BY
   p.Category, p.SubCategory, p.ProductName;
```



8. Slice and Dice

1. Slice the data cube for sales in 2024 only.
2. Dice the data cube to select sales data for Electronics in the North region during Q4.

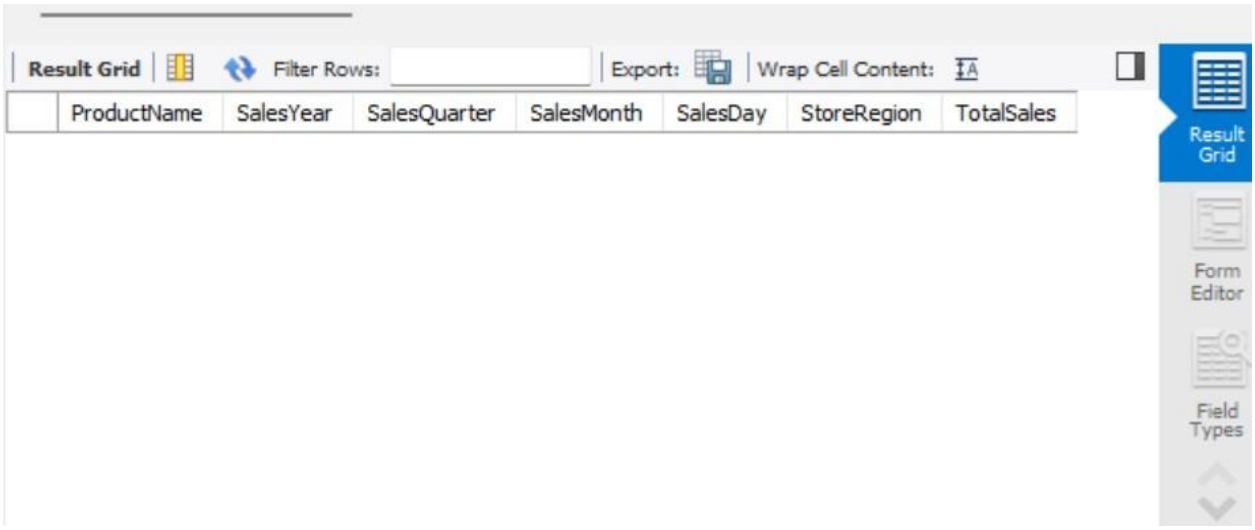
---

8. Slice and Dice

1. Slice the data cube for sales in `2024` only.
2. Dice the data cube to select sales data for `Electronics` in the `North` region during `Q4`. 8-

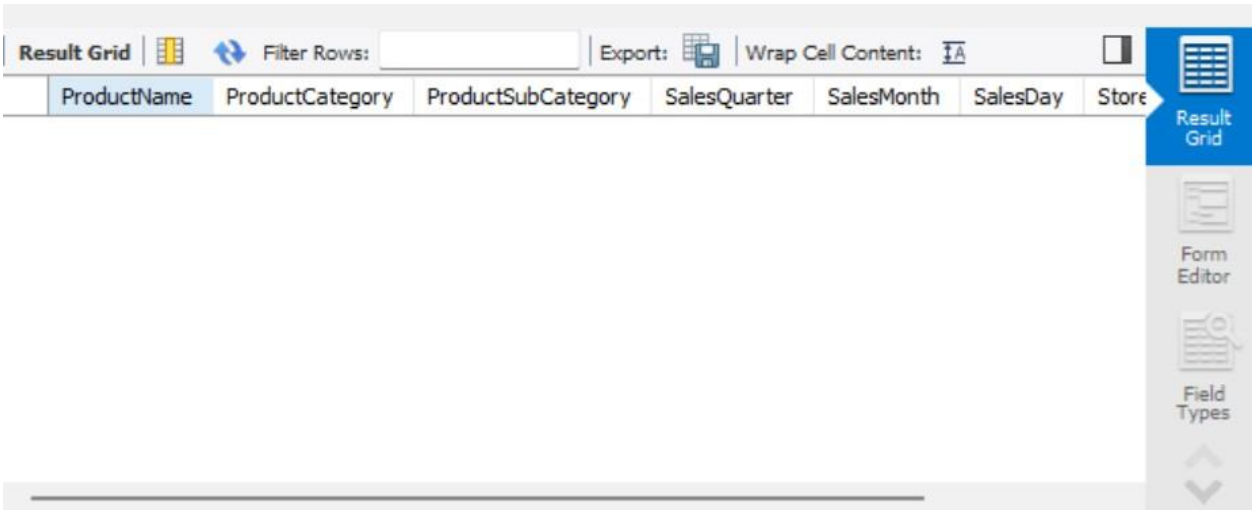
Slicing the data cube

```
SELECT
   p.ProductName,
   d.Year AS SalesYear,
   d.Quarter AS SalesQuarter,
   d.MonthName AS SalesMonth,
   d.Day AS SalesDay,
   s.Region AS StoreRegion,
   SUM(f.SalesAmount) AS TotalSales
FROM
   FactSales f
JOIN
   DimProduct p ON f.ProductKey = p.ProductKey
JOIN
   DimDate d ON f.DateKey = d.DateKey
JOIN
   DimStore s ON f.StoreKey = s.StoreKey
WHERE
   d.Year = 2024  -- Slice by the year 2024
GROUP BY
   p.ProductName, d.Year, d.Quarter, d.MonthName, d.Day, s.Region
ORDER BY
   d.Year, d.Quarter, d.MonthName, d.Day;
```



8 - Dice the data cube

```
SELECT
   p.ProductName,
   p.Category AS ProductCategory,
   p.SubCategory AS ProductSubCategory,
   d.Quarter AS SalesQuarter,
   d.MonthName AS SalesMonth,
   d.Day AS SalesDay,
   s.Region AS StoreRegion,
   SUM(f.SalesAmount) AS TotalSales
FROM
   FactSales f
JOIN
   DimProduct p ON f.ProductKey = p.ProductKey
JOIN
   DimDate d ON f.DateKey = d.DateKey
JOIN
   DimStore s ON f.StoreKey = s.StoreKey
WHERE
   p.Category = 'Electronics'    -- Filter for Electronics category
   AND s.Region = 'North'        -- Filter for North region
   AND d.Quarter = 4             -- Filter for Q4 (Quarter 4)
GROUP BY
   p.ProductName, p.Category, p.SubCategory, d.Quarter, d.MonthName, d.Day, s.Region
ORDER BY
   d.Quarter, d.MonthName, d.Day;
```



6. Roll Up and Drill Down

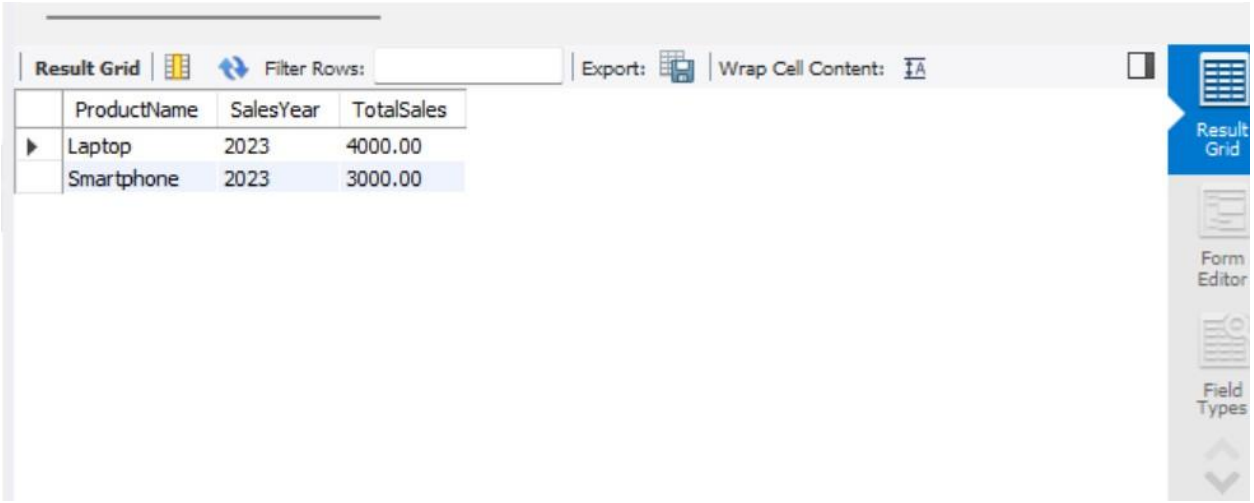
Roll up sales data from Day to Month and then to Year. Analyze how the aggregation changes. Drill down from Region to City and analyze sales trends.

```
6 - Roll Up
SELECT
   p.ProductName,
   d.Year AS SalesYear,
   d.MonthName AS SalesMonth,
   d.Day AS SalesDay,
   SUM(f.SalesAmount) AS TotalSales
FROM
   FactSales f
JOIN
   DimProduct p ON f.ProductKey = p.ProductKey
JOIN
   DimDate d ON f.DateKey = d.DateKey
GROUP BY
   p.ProductName, d.Year, d.MonthName, d.Day
ORDER BY
   d.Year, d.MonthName, d.Day;

-- 2. Aggregation at Month level (Roll-up from Day to Month)
SELECT
   p.ProductName,
   d.Year AS SalesYear,
   d.MonthName AS SalesMonth,
   SUM(f.SalesAmount) AS TotalSales
FROM
```

```
    FactSales f
JOIN
    DimProduct p ON f.ProductKey = p.ProductKey
JOIN
    DimDate d ON f.DateKey = d.DateKey
GROUP BY
    p.ProductName, d.Year, d.MonthName
ORDER BY
    d.Year, d.MonthName;

-- 3. Aggregation at Year level (Roll-up from Month to Year)
SELECT
    p.ProductName,
    d.Year AS SalesYear,
    SUM(f.SalesAmount) AS TotalSales
FROM
    FactSales f
JOIN
    DimProduct p ON f.ProductKey = p.ProductKey
JOIN
    DimDate d ON f.DateKey = d.DateKey
GROUP BY
    p.ProductName, d.Year ORDER BY
    d.Year;
```

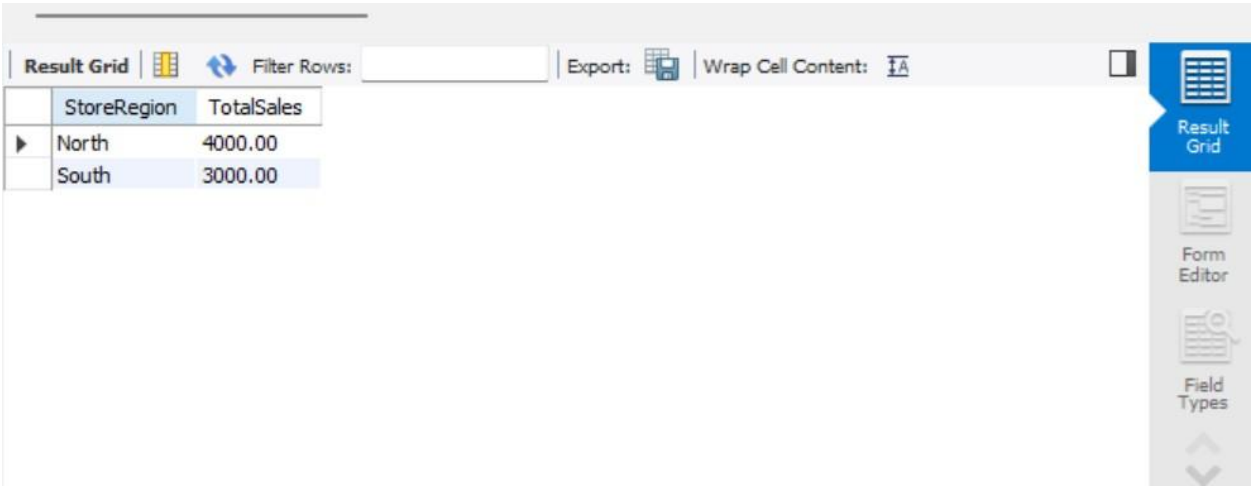


6 - Drill Down

```
SELECT
    s.Region AS StoreRegion,
    SUM(f.SalesAmount) AS TotalSales
FROM
```

```
    FactSales f
JOIN
    DimStore s ON f.StoreKey = s.StoreKey
GROUP BY
    s.Region
ORDER BY
    s.Region;

-- 2. Drill down to City level
SELECT
    s.Region AS StoreRegion,
    s.City AS StoreCity,
    SUM(f.SalesAmount) AS TotalSales
FROM
    FactSales f
JOIN
    DimStore s ON f.StoreKey = s.StoreKey
GROUP BY
    s.Region, s.City
ORDER BY
    s.Region, s.City;
```



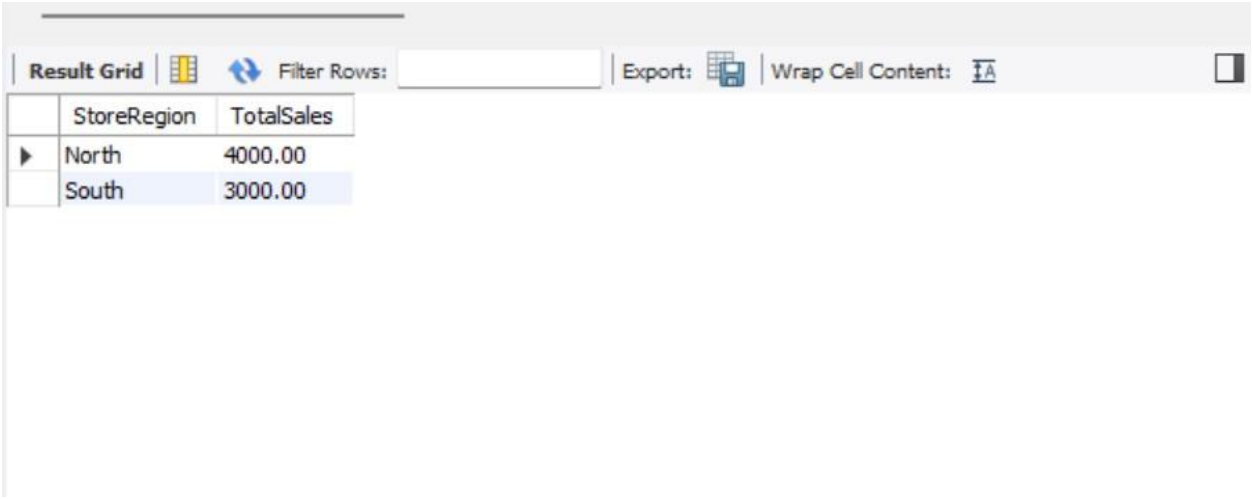
9. Pivot

1. Pivot the sales data cube to view Region as rows and Product Category as columns.
2. Analyze which region and category combination generated the highest sales.

9. Pivot

Pivot the sales data cube to view Region as rows and Product Category as columns. Analyze which region and category combination generated the highest sales.

```
SELECT
   s.Region AS StoreRegion,
   SUM(CASE WHEN p.Category = 'Electronics' THEN f.SalesAmount ELSE 0 END) AS
ElectronicsSales,
   SUM(CASE WHEN p.Category = 'Furniture' THEN f.SalesAmount ELSE 0 END) AS FurnitureSales,
   SUM(CASE WHEN p.Category = 'Clothing' THEN f.SalesAmount ELSE 0 END) AS ClothingSales
FROM
   FactSales f
JOIN
   DimProduct p ON f.ProductKey = p.ProductKey
JOIN
   DimStore s ON f.StoreKey = s.StoreKey
GROUP BY
   s.Region
ORDER BY
   s.Region;
```

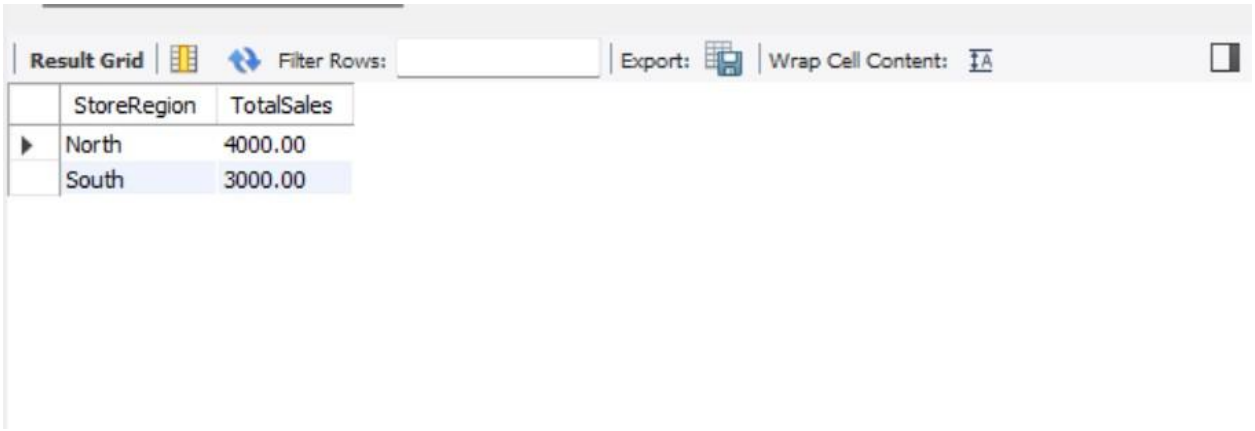


	StoreRegion	TotalSales
▶	North	4000.00
	South	3000.00

9- 2

```
SELECT
   s.Region AS StoreRegion,
   p.Category AS ProductCategory,
   SUM(f.SalesAmount) AS TotalSales
FROM
   FactSales f
```

```
JOIN
   DimProduct p ON f.ProductKey = p.ProductKey
JOIN
   DimStore s ON f.StoreKey = s.StoreKey
GROUP BY
   s.Region, p.Category
ORDER BY
   TotalSales DESC
LIMIT 1;
```



	StoreRegion	TotalSales
▶	North	4000.00
	South	3000.00