Projected sales of main products in 2013
Distribution of market share among the major industry players
Share of market activity
Passive market share
Projected sales of main products in 2013

Sales Intelligence Transformation: Empowering Axon Classic Cars with PowerBI and SQL

Author:

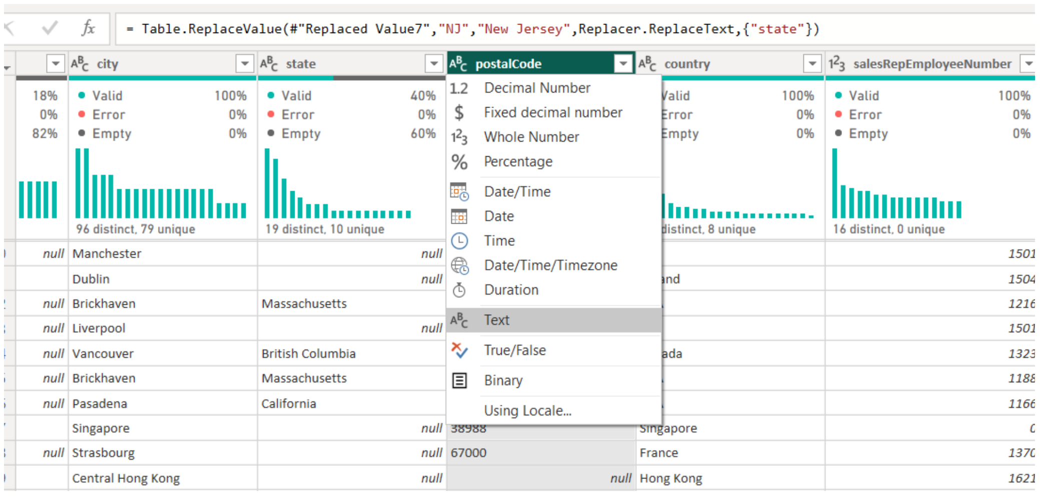
Govind G
https://github.com/govindgopidas

1. **<u>Understanding the business Problem</u>**:
   - A small company Axon, which is a retailer selling classic cars, is facing issues in managing and analyzing their sales data. The sales team is struggling to make sense of the data and they do not have a centralized system to manage and analyze the data. The management is unable to get accurate and up-to-date sales reports, which is affecting the decision-making process.
   - To address this issue, the company has decided to implement a Business Intelligence (BI) tool that can help them manage and analyze their sales data effectively. They have shortlisted Microsoft PowerBI and SQL as the BI tools for this project.
   - The goal of the capstone project is to design and implement a BI solution using PowerBI and SQL that can help the company manage and analyze their sales data effectively.

2. **<u>Set Up Your Environment</u>**:
   - Install and set up Microsoft PowerBI and MySQL if you haven't already.

3. **<u>Database Setup</u>**:
   - Create a new MySQL database as "classicmodels" using the provided schema.
   - Load the data into MySQL. You can use the provided link to download the database schema.

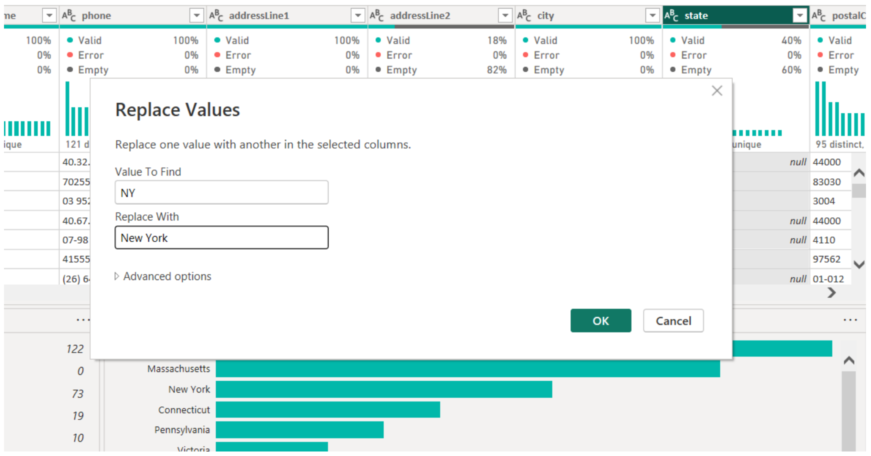
4. **<u>Data Extraction and Transformation</u>**:
   - Used PowerBI to connect to the MySQL database as a data source (You can use SSMS also).
   - Extract data from the relevant tables (Customers, Products, Orders, etc.) into PowerBI.
   - Perform data cleaning and transformation as necessary. This includes handling duplicates, missing values, and ensuring data consistency, This includes following processes:
   - In the customer table, the column postal code is not displaying correctly, that is because the data type is set to whole numbers, but it is containing text data, so i changed it into the text data type.



```
= Table.ReplaceValue(#"Replaced Value7","NJ","New Jersey",Replacer.ReplaceText,{"state"})
```

	city	state	postalCode	country	salesRepEmployeeNumber
18% / 0% / 82%	Valid 100% / Error 0% / Empty 0%	Valid 40% / Error 0% / Empty 60%		Valid 100% / Error 0% / Empty 0%	Valid 100% / Error 0% / Empty 0%
	96 distinct, 79 unique	19 distinct, 10 unique		distinct, 8 unique	16 distinct, 0 unique
null	Manchester	null			1501
	Dublin	null		nd	1504
null	Brickhaven	Massachusetts			1216
null	Liverpool	null			1501
null	Vancouver	British Columbia		ada	1323
null	Brickhaven	Massachusetts			1188
null	Pasadena	California			1166
	Singapore	null	38988	Singapore	0
null	Strasbourg	null	67000	France	1370
	Central Hong Kong	null	null	Hong Kong	1621

- Some of the US states are showing abbreviations only so I changed it into the original names of the states using ‘Replace values’



Replace Values

Replace one value with another in the selected columns.

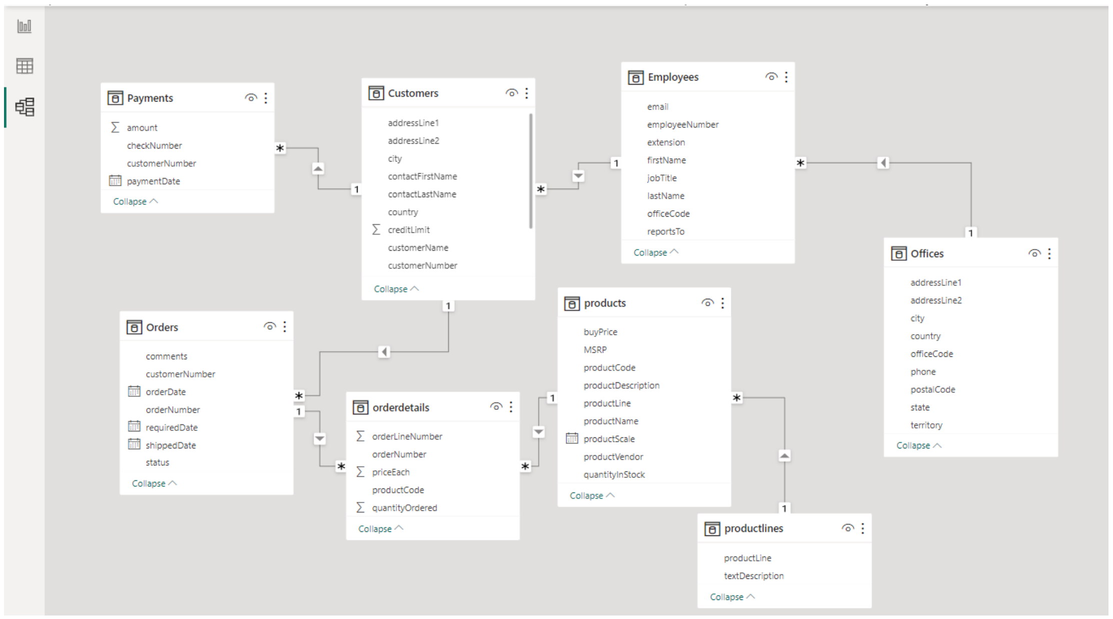
Value To Find: NY

Replace With: New York

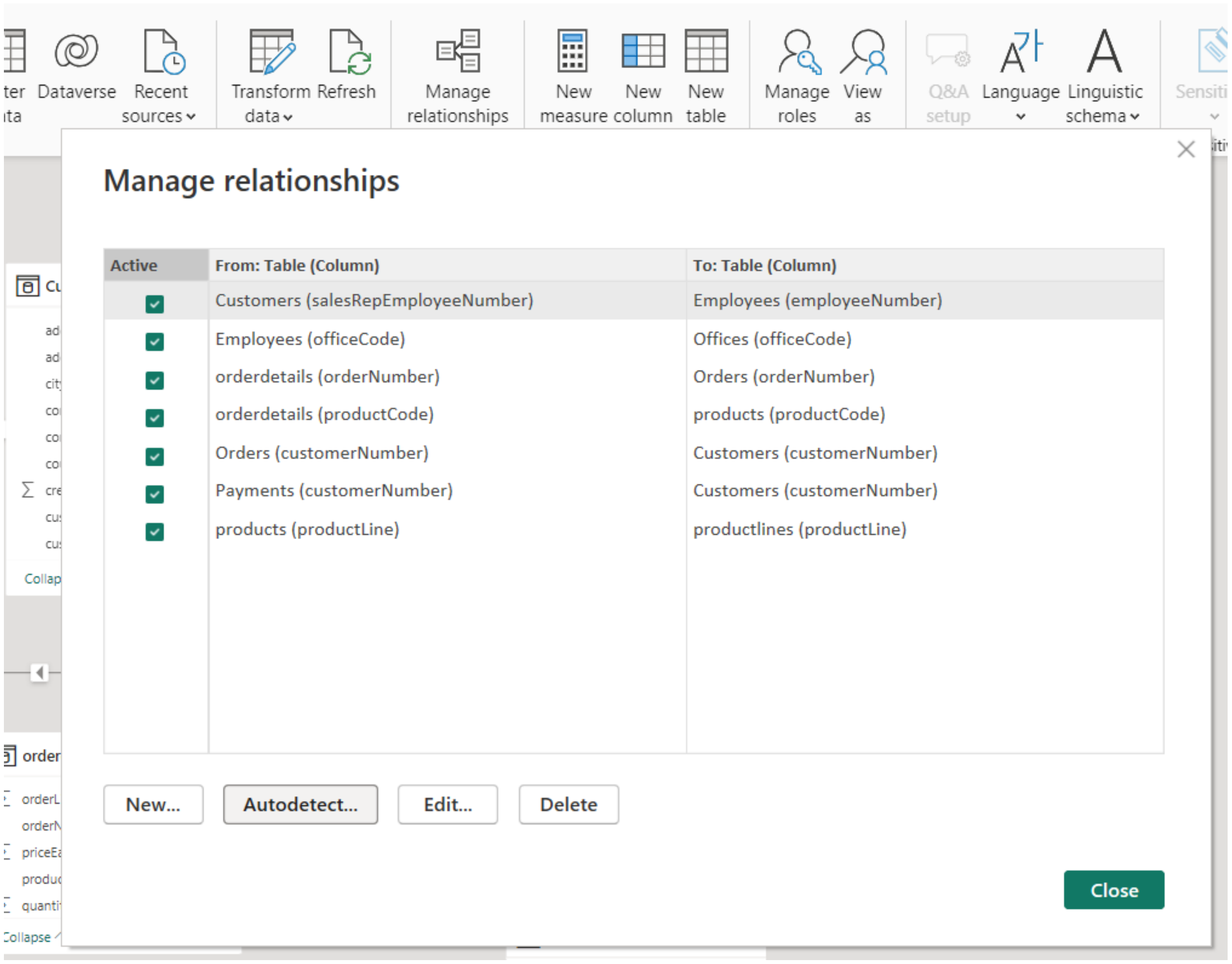
- Then set the first row as header, so that it acts as a title for the type of information they will find in each column**.**
- Removed the empty columns html descriptions and images from the table product lines.
- Then loaded to the power bi desktop by clicking on close and apply.

5. <u>Data Modeling</u>:

- ***Designing a data model in PowerBI that captures the relationships between the tables involves defining how different tables are connected or related to each other. This is crucial for creating meaningful and interactive reports and dashboards***.
- For defining the relationships we need to Identify the key tables -> Understanding the table relationships and then -> Define the relationship
- Open your PowerBI project and go to the "Model" view.
- Create relationships between tables by connecting the relevant fields (columns) that serve as keys. To do this:
    i. Click on the table from which you want to create a relationship (e.g., "Orders").
    ii. Select the field (e.g., "customer_number") that links to another table (e.g., "Customers").
    iii. Drag and drop the field onto the corresponding field in the related table (e.g., "Customers" -> "customer_number").
    iv. Repeat this process for all relevant relationships.



- You can also do this process by clicking on the “Manage relationships” option and click on autodetect (The power bi will automatically detect relationships).

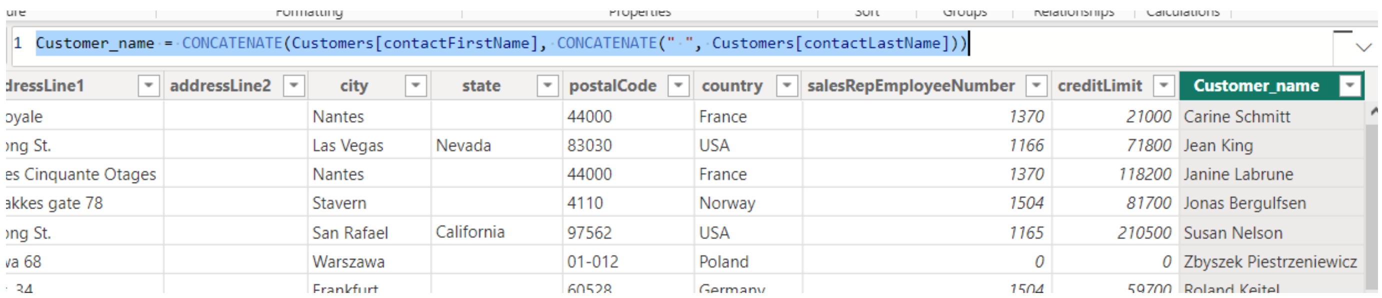


Manage relationships

Active	From: Table (Column)	To: Table (Column)
✓	Customers (salesRepEmployeeNumber)	Employees (employeeNumber)
✓	Employees (officeCode)	Offices (officeCode)
✓	orderdetails (orderNumber)	Orders (orderNumber)
✓	orderdetails (productCode)	products (productCode)
✓	Orders (customerNumber)	Customers (customerNumber)
✓	Payments (customerNumber)	Customers (customerNumber)
✓	products (productLine)	productlines (productLine)

- ***Creating calculated columns and measures using DAX (Data Analysis Expressions) to facilitate analysis.***
- Created a new column that is “Customer_name” by concatenating the columns “customerlastname” and “customerfirstname”. Using the dax expression:

```
Customer_name = CONCATENATE(Customers[contactFirstName], CONCATENATE(" ",
Customers[contactLastName]))
```

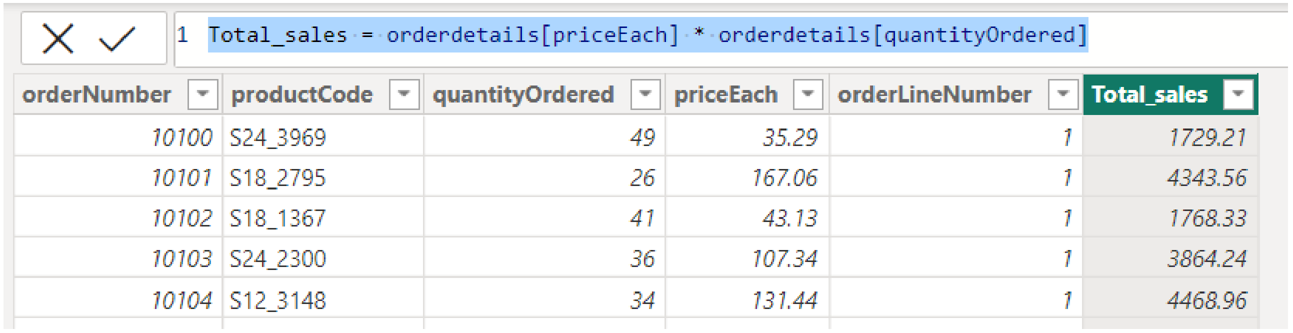


```
Customer_name = CONCATENATE(Customers[contactFirstName], CONCATENATE(" ", Customers[contactLastName]))
```

dressLine1	addressLine2	city	state	postalCode	country	salesRepEmployeeNumber	creditLimit	Customer_name
oyale		Nantes		44000	France	1370	21000	Carine Schmitt
ong St.		Las Vegas	Nevada	83030	USA	1166	71800	Jean King
es Cinquante Otages		Nantes		44000	France	1370	118200	Janine Labrune
akkes gate 78		Stavern		4110	Norway	1504	81700	Jonas Bergulfsen
ong St.		San Rafael	California	97562	USA	1165	210500	Susan Nelson
va 68		Warszawa		01-012	Poland	0	0	Zbyszek Piestrzeniewicz
34		Frankfurt		60528	Germany	1504	59700	Roland Keitel

- Created a new column in order details table that is “Total_sales” by multiplying the columns “quantity ordered” and “price each”. Using the dax expression:

```
Total_sales = orderdetails[priceEach] * orderdetails[quantityOrdered]
```

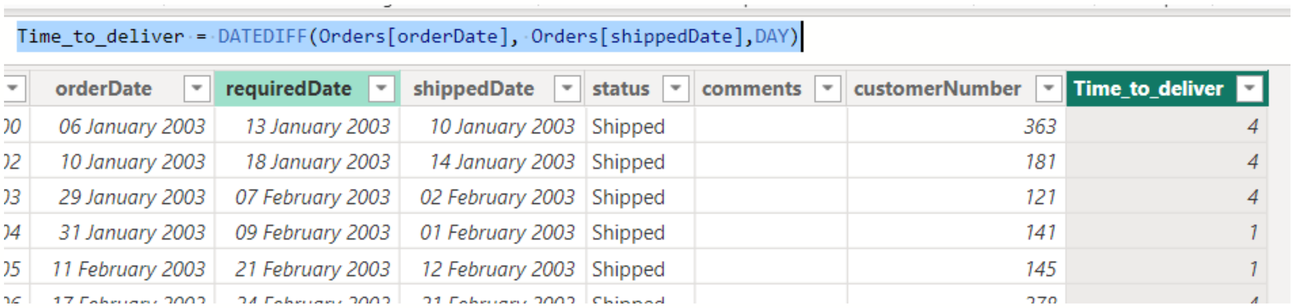


```
1 Total_sales = orderdetails[priceEach] * orderdetails[quantityOrdered]
```

orderNumber	productCode	quantityOrdered	priceEach	orderLineNumber	Total_sales
10100	S24_3969	49	35.29	1	1729.21
10101	S18_2795	26	167.06	1	4343.56
10102	S18_1367	41	43.13	1	1768.33
10103	S24_2300	36	107.34	1	3864.24
10104	S12_3148	34	131.44	1	4468.96

- Created a new column in the table orders that is “Time_to_delivery” by calculating the difference between the columns “order date” and “shipped date”. Using the dax function ”DATEDIFF”:

```
Time_to_deliver = DATEDIFF(Orders[orderDate], Orders[shippedDate],DAY)
```

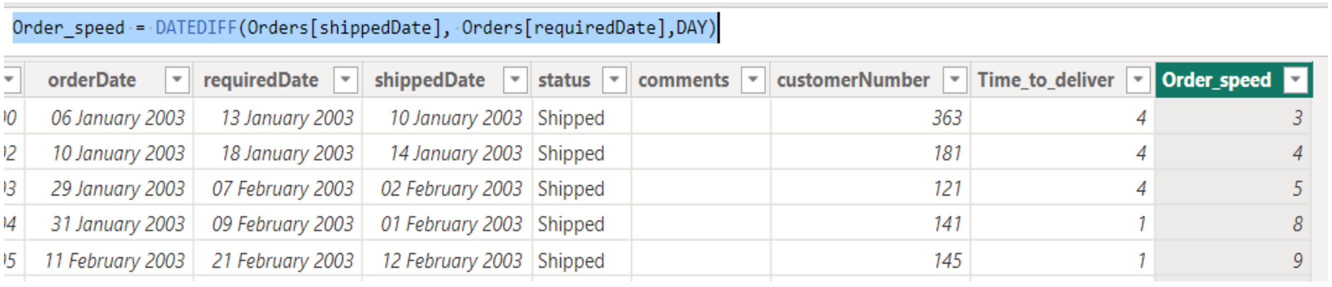


```
Time_to_deliver = DATEDIFF(Orders[orderDate], Orders[shippedDate],DAY)
```

	orderDate	requiredDate	shippedDate	status	comments	customerNumber	Time_to_deliver
)0	06 January 2003	13 January 2003	10 January 2003	Shipped		363	4
)2	10 January 2003	18 January 2003	14 January 2003	Shipped		181	4
)3	29 January 2003	07 February 2003	02 February 2003	Shipped		121	4
)4	31 January 2003	09 February 2003	01 February 2003	Shipped		141	1
)5	11 February 2003	21 February 2003	12 February 2003	Shipped		145	1

- Created another column that is “Order_speed” by calculating the difference between the columns “order required date” and “order delivered date” using the dax function “DATEDIFF”.

```
Order_speed = DATEDIFF(Orders[shippedDate], Orders[requiredDate],DAY)
```

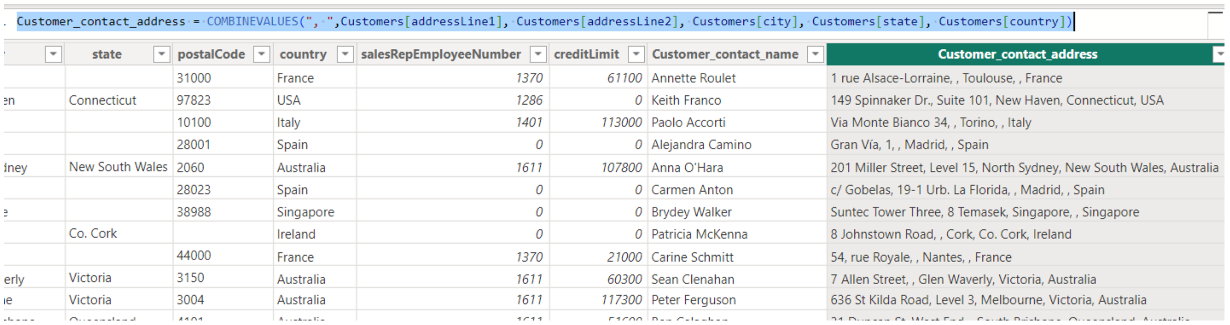


```
Order_speed = DATEDIFF(Orders[shippedDate], Orders[requiredDate],DAY)
```

	orderDate	requiredDate	shippedDate	status	comments	customerNumber	Time_to_deliver	Order_speed
)0	06 January 2003	13 January 2003	10 January 2003	Shipped		363	4	3
)2	10 January 2003	18 January 2003	14 January 2003	Shipped		181	4	4
)3	29 January 2003	07 February 2003	02 February 2003	Shipped		121	4	5
)4	31 January 2003	09 February 2003	01 February 2003	Shipped		141	1	8
)5	11 February 2003	21 February 2003	12 February 2003	Shipped		145	1	9

- Created an “Customers_contact_address” column for convenience in the customers table by combining different tables.

```
Customer_contact_address = COMBINEVALUES(", ",Customers[addressLine1],
Customers[addressLine2], Customers[city], Customers[state], Customers[country])
```

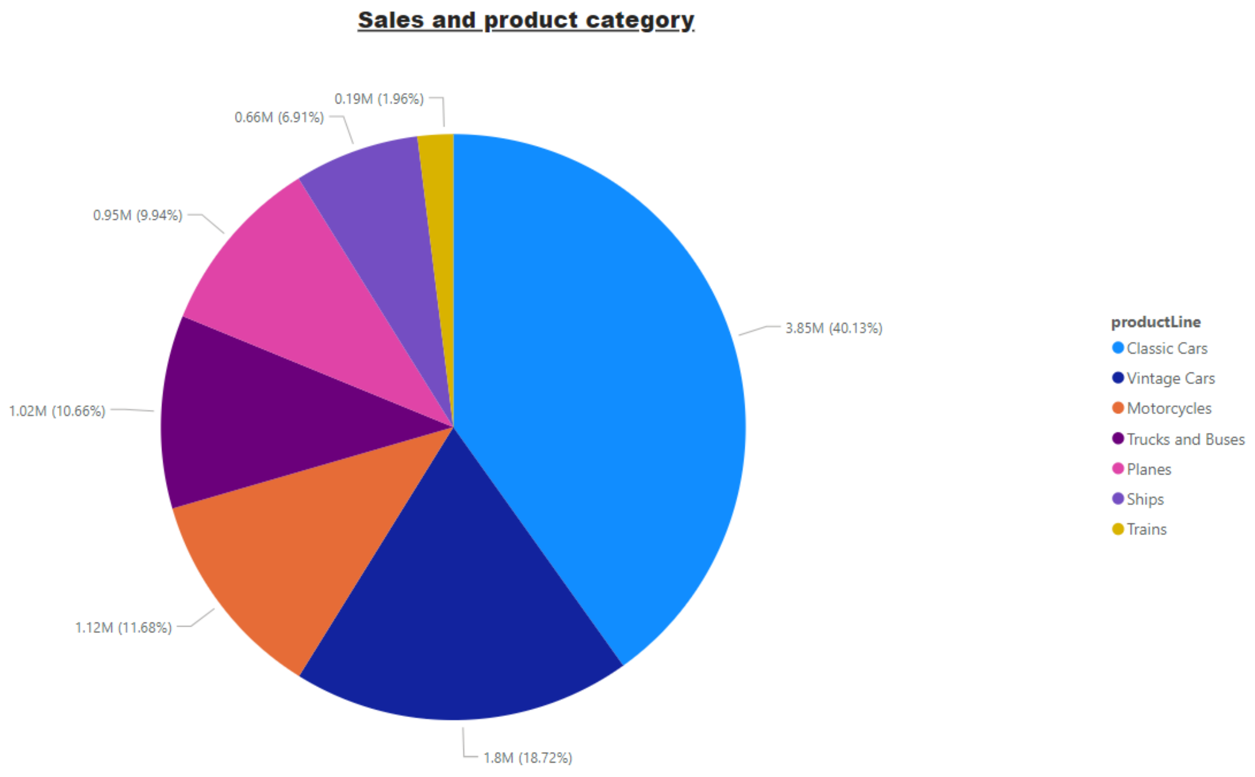


```
Customer_contact_address = COMBINEVALUES(", ",Customers[addressLine1], Customers[addressLine2], Customers[city], Customers[state], Customers[country])
```

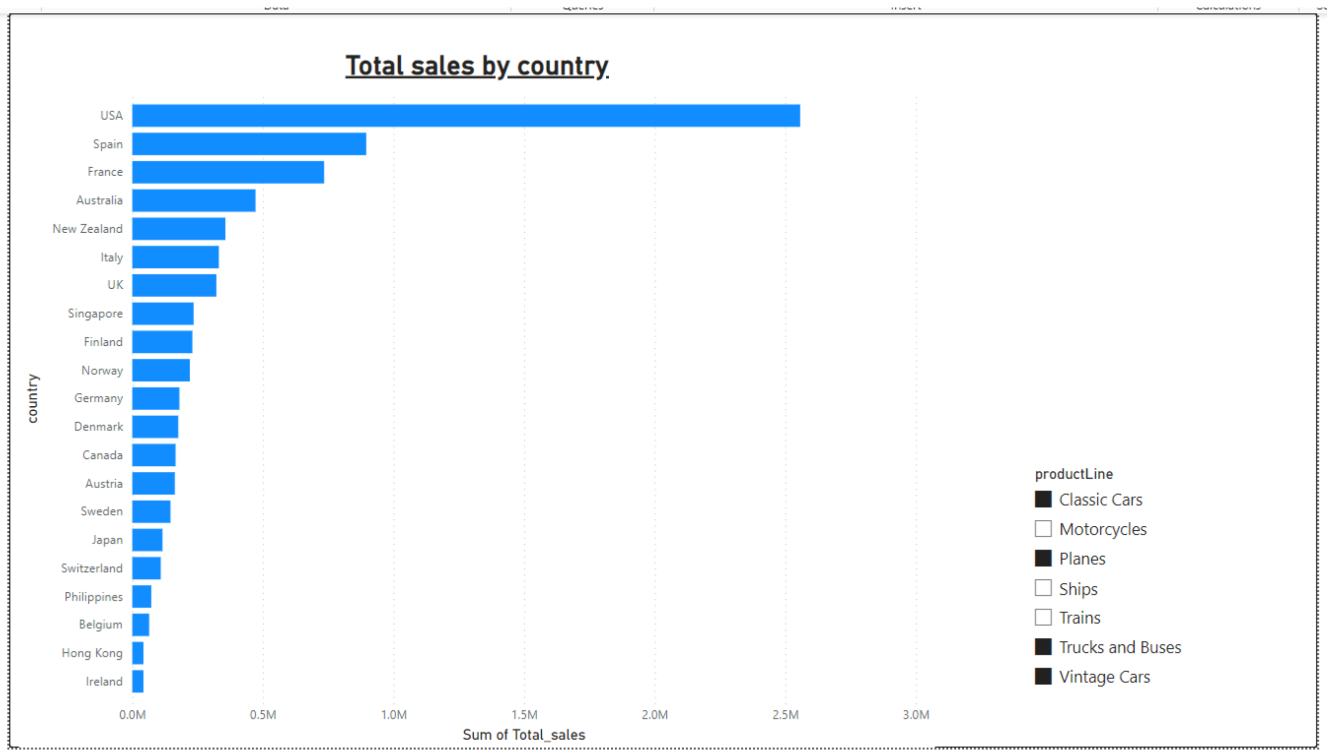
	state	postalCode	country	salesRepEmployeeNumber	creditLimit	Customer_contact_name	Customer_contact_address
		31000	France	1370	61100	Annette Roulet	1 rue Alsace-Lorraine, , Toulouse, , France
en	Connecticut	97823	USA	1286	0	Keith Franco	149 Spinnaker Dr., Suite 101, New Haven, Connecticut, USA
		10100	Italy	1401	113000	Paolo Accorti	Via Monte Bianco 34, , Torino, , Italy
		28001	Spain	0	0	Alejandra Camino	Gran Vía, 1, , Madrid, , Spain
dney	New South Wales	2060	Australia	1611	107800	Anna O'Hara	201 Miller Street, Level 15, North Sydney, New South Wales, Australia
		28023	Spain	0	0	Carmen Anton	c/ Gobelas, 19-1 Urb. La Florida, , Madrid, , Spain
e		38988	Singapore	0	0	Brydey Walker	Suntec Tower Three, 8 Temasek, Singapore, , Singapore
	Co. Cork		Ireland	0	0	Patricia McKenna	8 Johnstown Road, , Cork, Co. Cork, Ireland
		44000	France	1370	21000	Carine Schmitt	54, rue Royale, , Nantes, , France
erly	Victoria	3150	Australia	1611	60300	Sean Clenahan	7 Allen Street, , Glen Waverly, Victoria, Australia
e	Victoria	3004	Australia	1611	117300	Peter Ferguson	636 St Kilda Road, Level 3, Melbourne, Victoria, Australia

6. **Dashboard and Report Creation**:

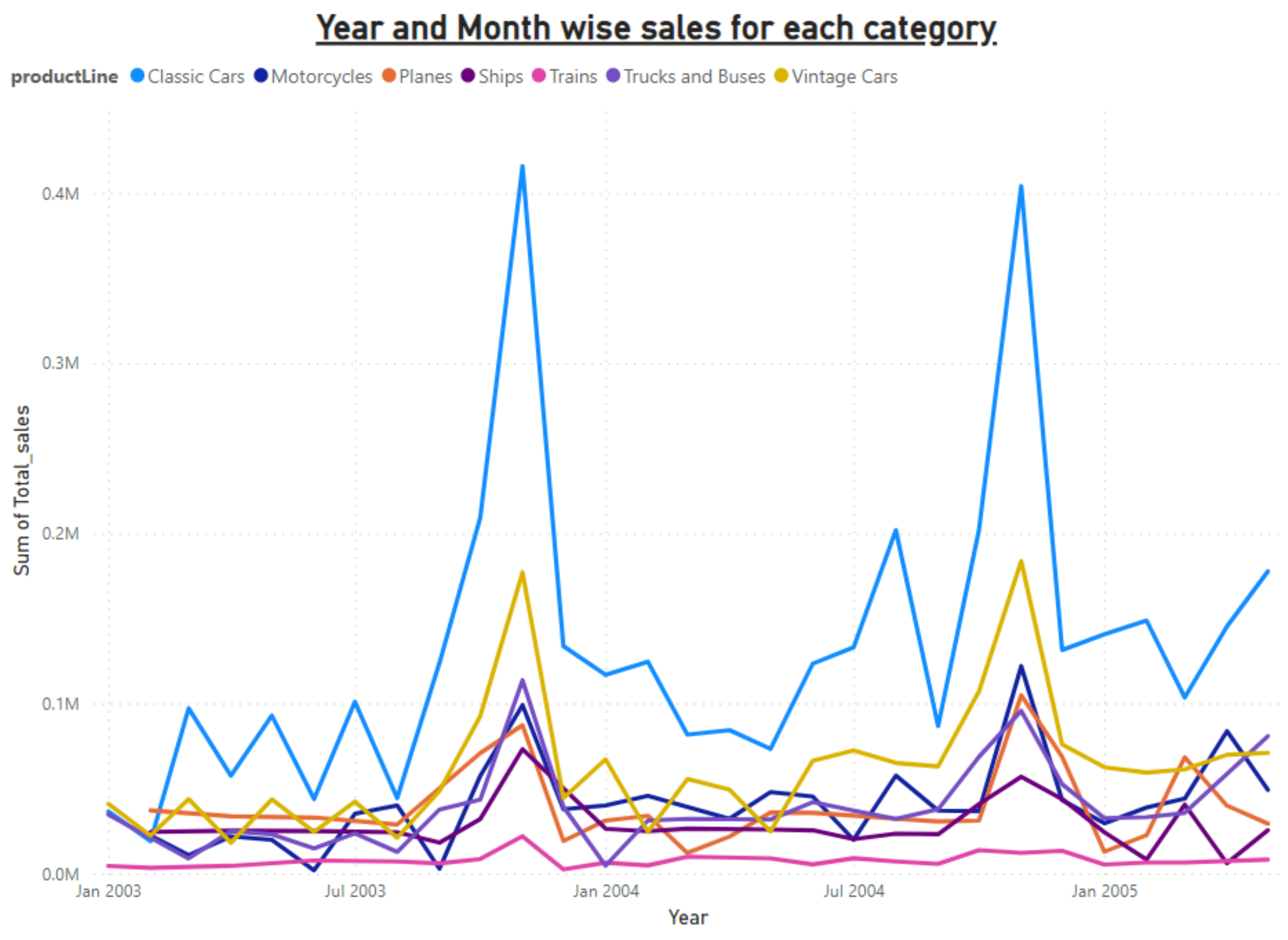
- ***Used PowerBI's visualization tools to design interactive dashboards and reports.***
- ***Created charts, graphs, and tables to visualize the data effectively.***
- Created a pie chart to compare the sales by product category.



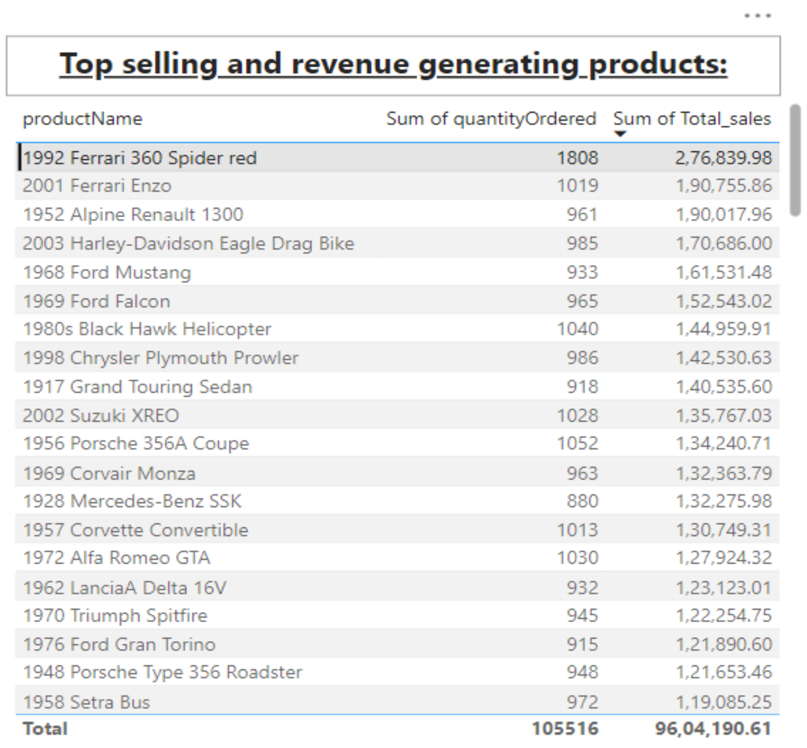
- Created a Bar chart to compare the sales by Country and applied a slicer with product category.



- Created a line chart to identify Time Series Sales Trends of each category of products.



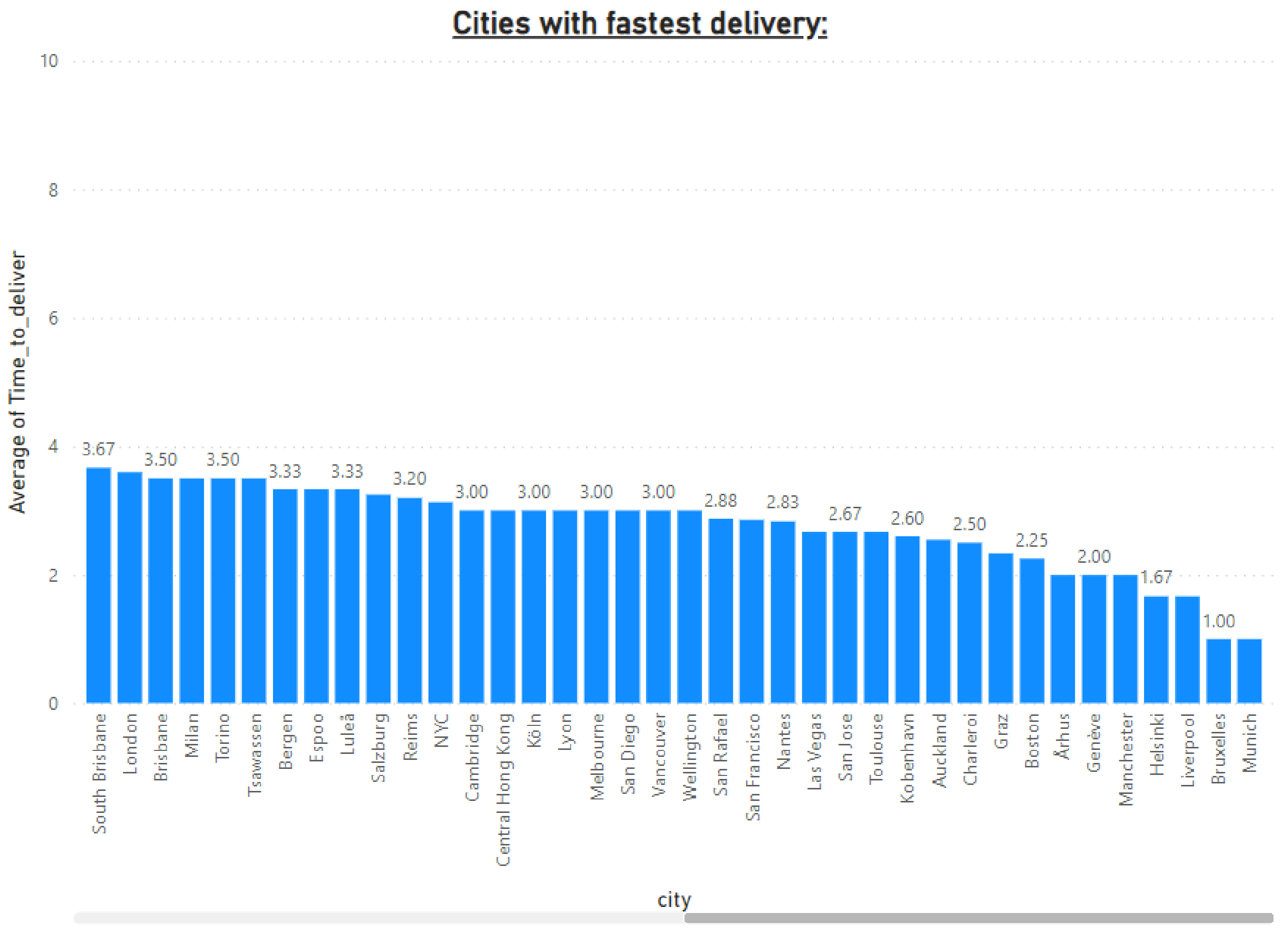
- Created a table to visualize the top selling and revenue generating product:



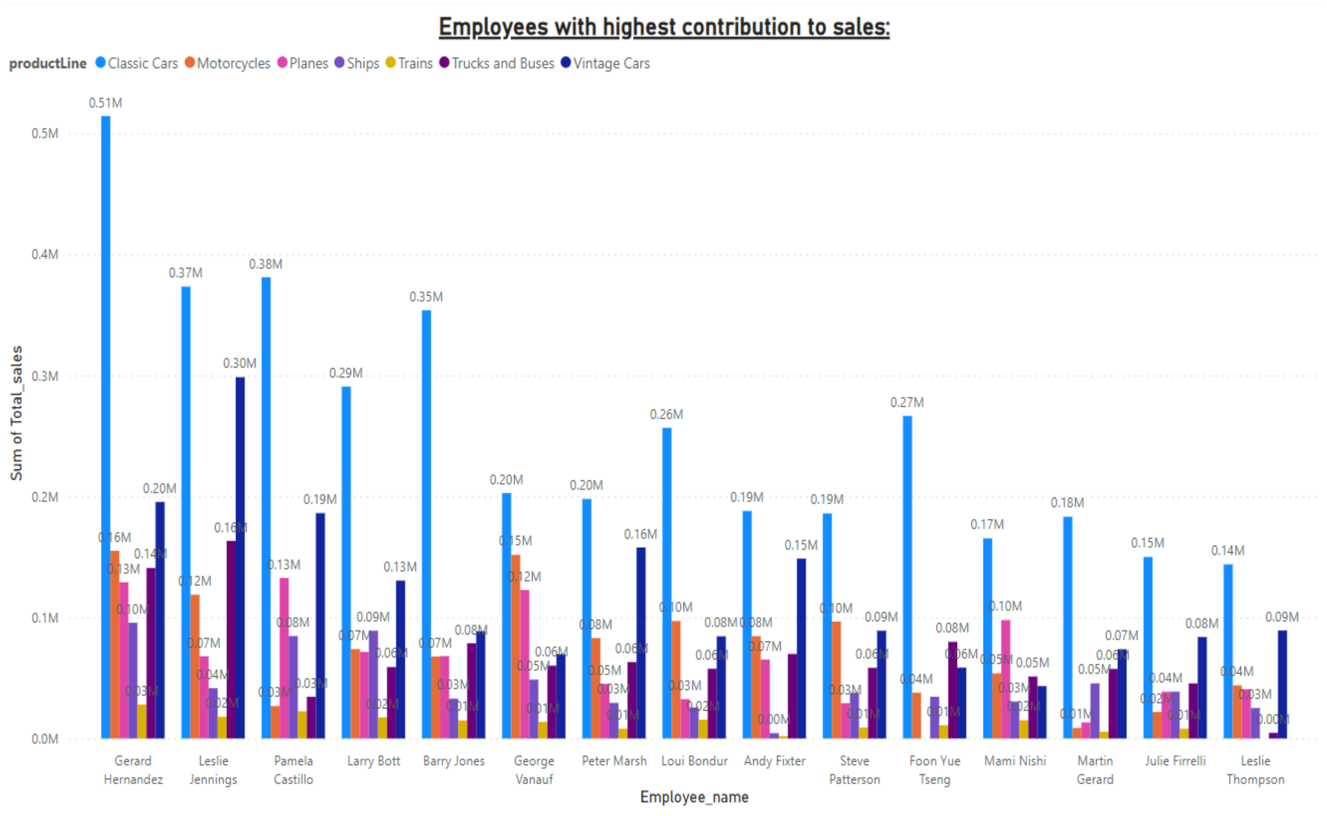
Top selling and revenue generating products:

productName	Sum of quantityOrdered	Sum of Total_sales
1992 Ferrari 360 Spider red	1808	2,76,839.98
2001 Ferrari Enzo	1019	1,90,755.86
1952 Alpine Renault 1300	961	1,90,017.96
2003 Harley-Davidson Eagle Drag Bike	985	1,70,686.00
1968 Ford Mustang	933	1,61,531.48
1969 Ford Falcon	965	1,52,543.02
1980s Black Hawk Helicopter	1040	1,44,959.91
1998 Chrysler Plymouth Prowler	986	1,42,530.63
1917 Grand Touring Sedan	918	1,40,535.60
2002 Suzuki XREO	1028	1,35,767.03
1956 Porsche 356A Coupe	1052	1,34,240.71
1969 Corvair Monza	963	1,32,363.79
1928 Mercedes-Benz SSK	880	1,32,275.98
1957 Corvette Convertible	1013	1,30,749.31
1972 Alfa Romeo GTA	1030	1,27,924.32
1962 LanciaA Delta 16V	932	1,23,123.01
1970 Triumph Spitfire	945	1,22,254.75
1976 Ford Gran Torino	915	1,21,890.60
1948 Porsche Type 356 Roadster	948	1,21,653.46
1958 Setra Bus	972	1,19,085.25
Total	**105516**	**96,04,190.61**

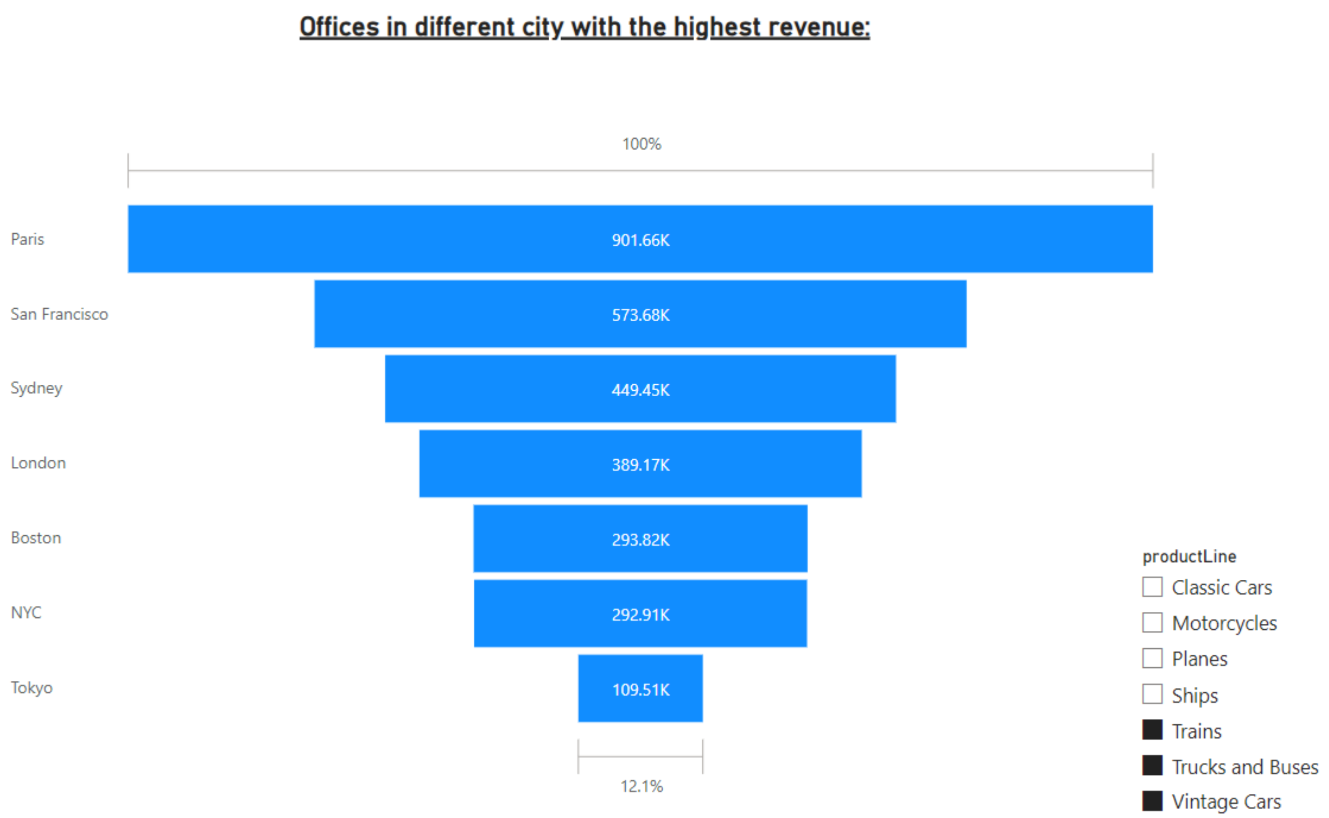
○ Created a column chart to visualize the top cities that take least time to delivery the products.



○ Created a clustered column chart to visualize the employees with highest contribution to sales by category:



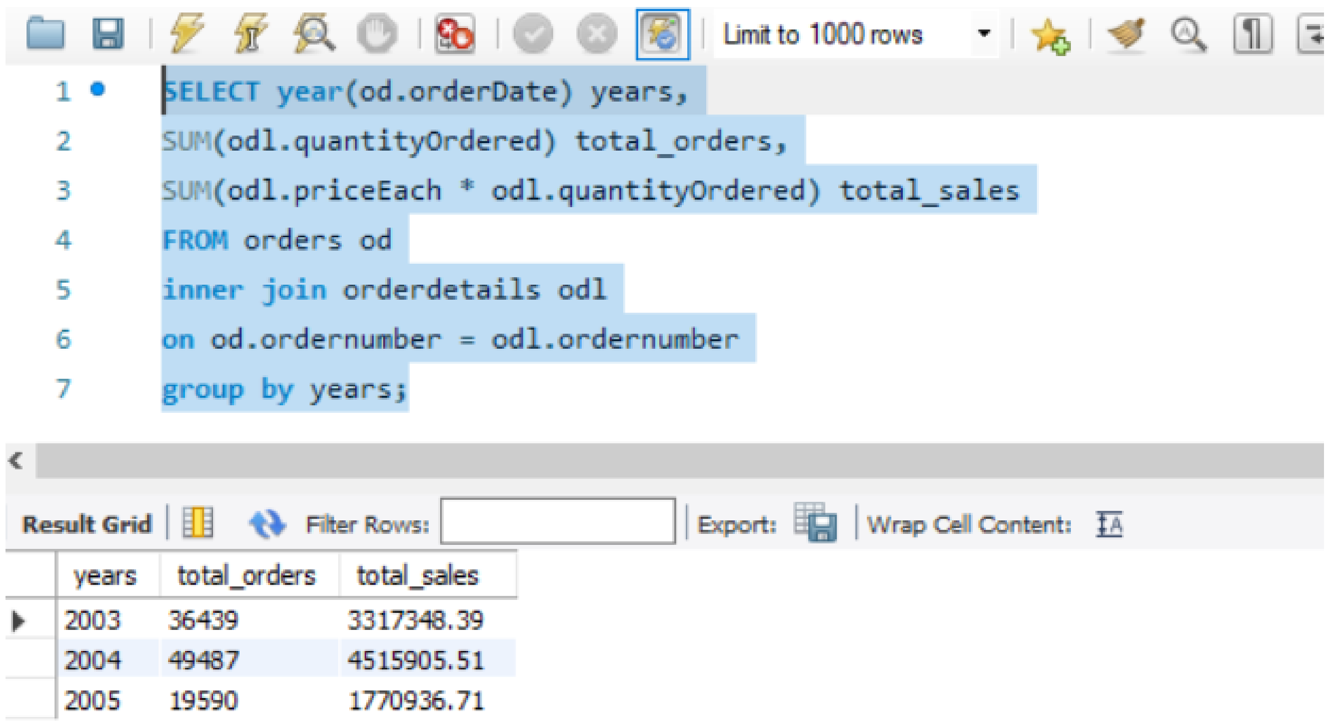
- Created a Funnel chart to visualize offices in different cities and their revenues with category slicer.



-
- Ensure that your dashboards provide valuable insights into sales performance, customer behavior, and other relevant metrics.
- Make use of DAX functions for more advanced calculations.

7. **Advanced Analytics with SQL**:
   - Used SQL to perform advanced analytics on the data to extract insights.
   - Created a table on the results grid to compare the year wise sales report:

```
SELECT year(od.orderDate) years,

SUM(odl.quantityOrdered) total_orders,

SUM(odl.priceEach * odl.quantityOrdered) total_sales

FROM orders od

inner join orderdetails odl

on od.ordernumber = odl.ordernumber

group by years;
```

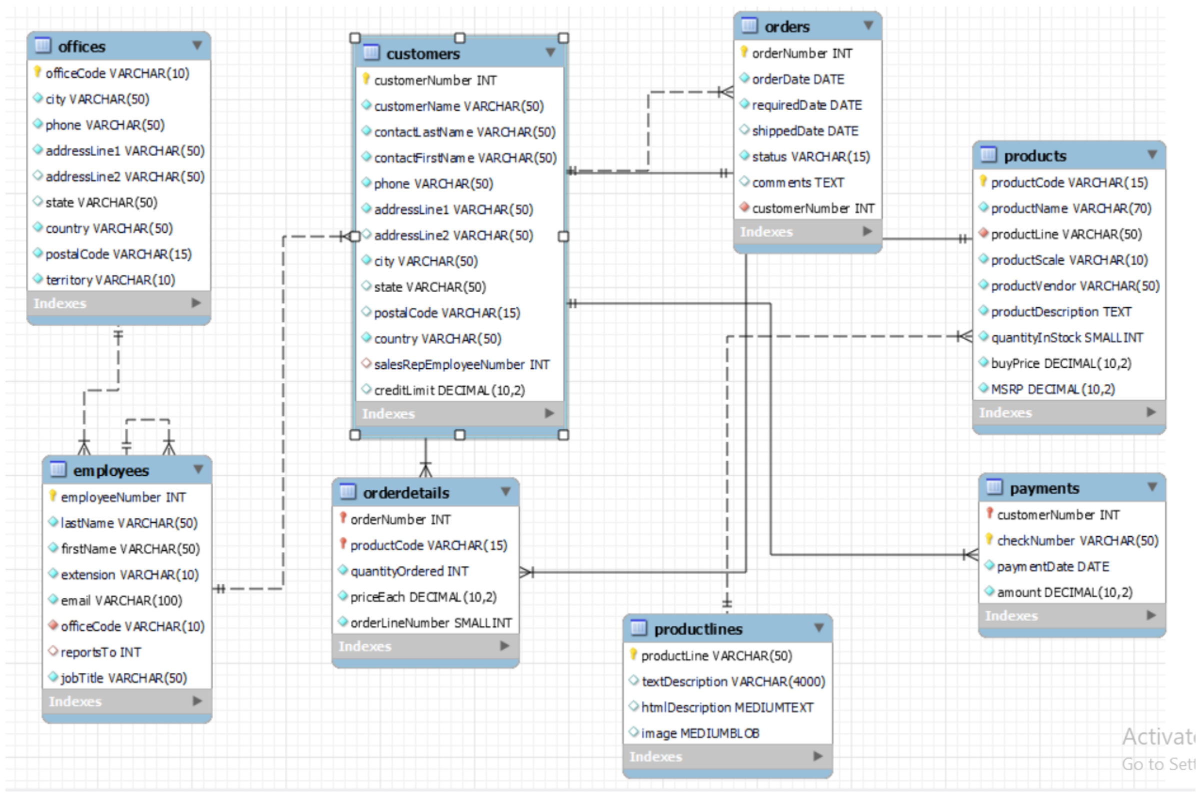


```sql
SELECT year(od.orderDate) years,
SUM(odl.quantityOrdered) total_orders,
SUM(odl.priceEach * odl.quantityOrdered) total_sales
FROM orders od
inner join orderdetails odl
on od.ordernumber = odl.ordernumber
group by years;
```

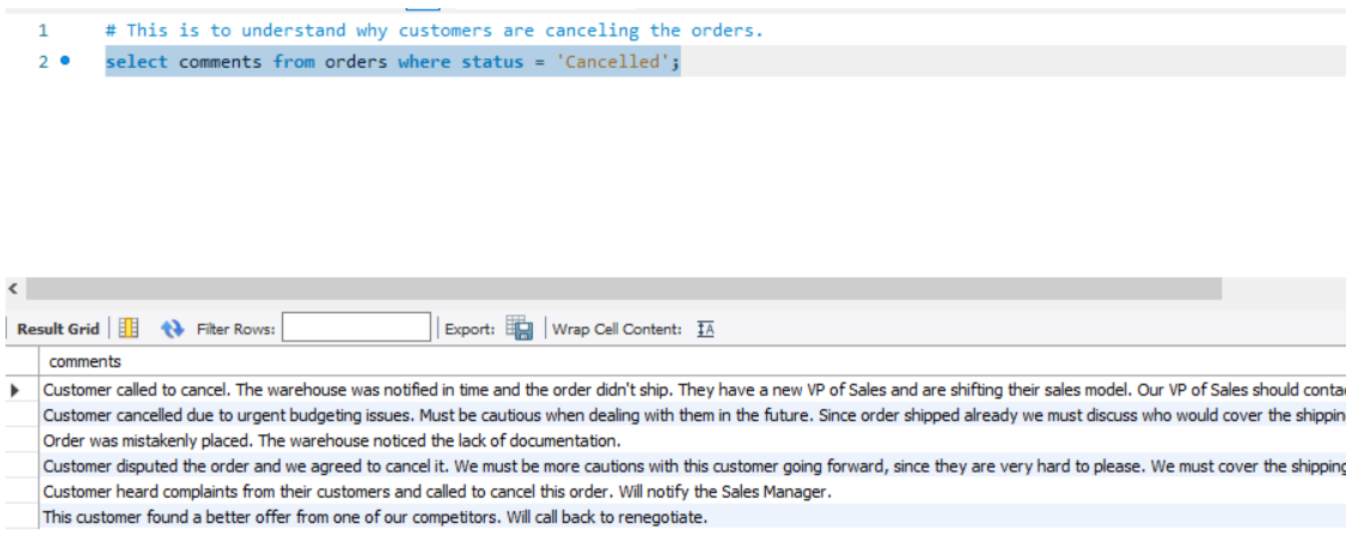
years	total_orders	total_sales
2003	36439	3317348.39
2004	49487	4515905.51
2005	19590	1770936.71

- Created an EER diagram (Enhanced Entity-Relationship), an essential part of the modeling interface in MySQL Workbench. EER diagrams provide a visual representation of the relationships among the tables in the model.

  To create an eer diagram Click on Database -> Reverse Engineer->Select your stored connection from the dropdown-> Select your Database, then click Next-> Select the Tables of the Database which you want to be visible on the ER Diagram, then click Execute-> Then click finish



- Analyzed the orders table for canceled orders and there comments, to understand why customers are canceling the orders:

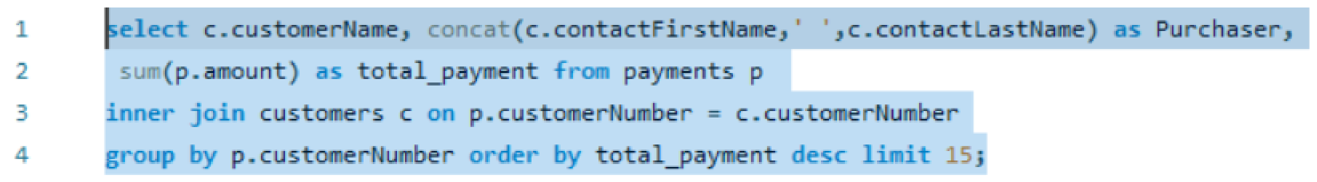


```
# This is to understand why customers are canceling the orders.
select comments from orders where status = 'Cancelled';
```

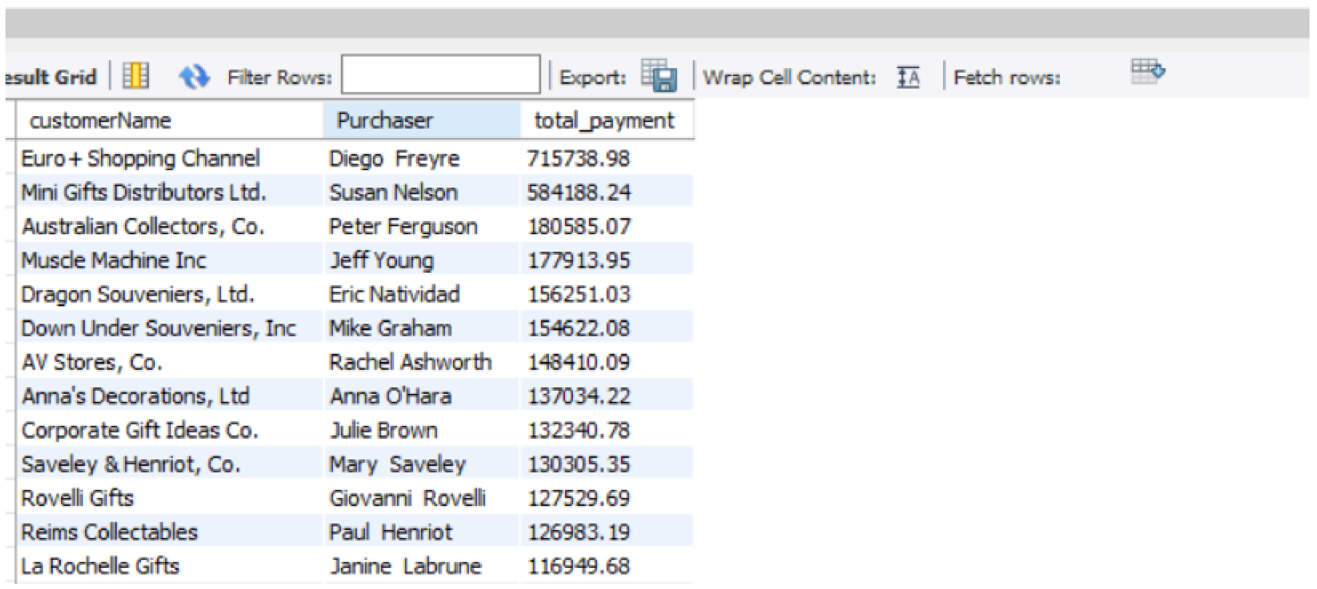
comments
Customer called to cancel. The warehouse was notified in time and the order didn't ship. They have a new VP of Sales and are shifting their sales model. Our VP of Sales should conta
Customer cancelled due to urgent budgeting issues. Must be cautious when dealing with them in the future. Since order shipped already we must discuss who would cover the shippin
Order was mistakenly placed. The warehouse noticed the lack of documentation.
Customer disputed the order and we agreed to cancel it. We must be more cautions with this customer going forward, since they are very hard to please. We must cover the shipping
Customer heard complaints from their customers and called to cancel this order. Will notify the Sales Manager.
This customer found a better offer from one of our competitors. Will call back to renegotiate.

- Analyzed the payment table with the customer table to find the top 15 customers with highest trade. For that i used several functions like concat, sum(), inner join, grout by, order by, limit, etc.

```
select c.customerName, concat(c.contactFirstName,' ',c.contactLastName) as
Purchaser,

 sum(p.amount) as total_payment from payments p

inner join customers c on p.customerNumber = c.customerNumber

group by p.customerNumber order by total_payment desc limit 15;
```



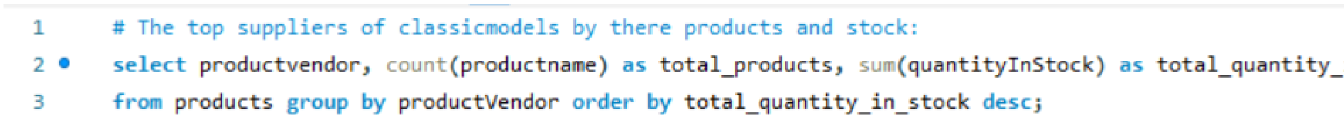
```
select c.customerName, concat(c.contactFirstName,' ',c.contactLastName) as Purchaser,
 sum(p.amount) as total_payment from payments p
inner join customers c on p.customerNumber = c.customerNumber
group by p.customerNumber order by total_payment desc limit 15;
```



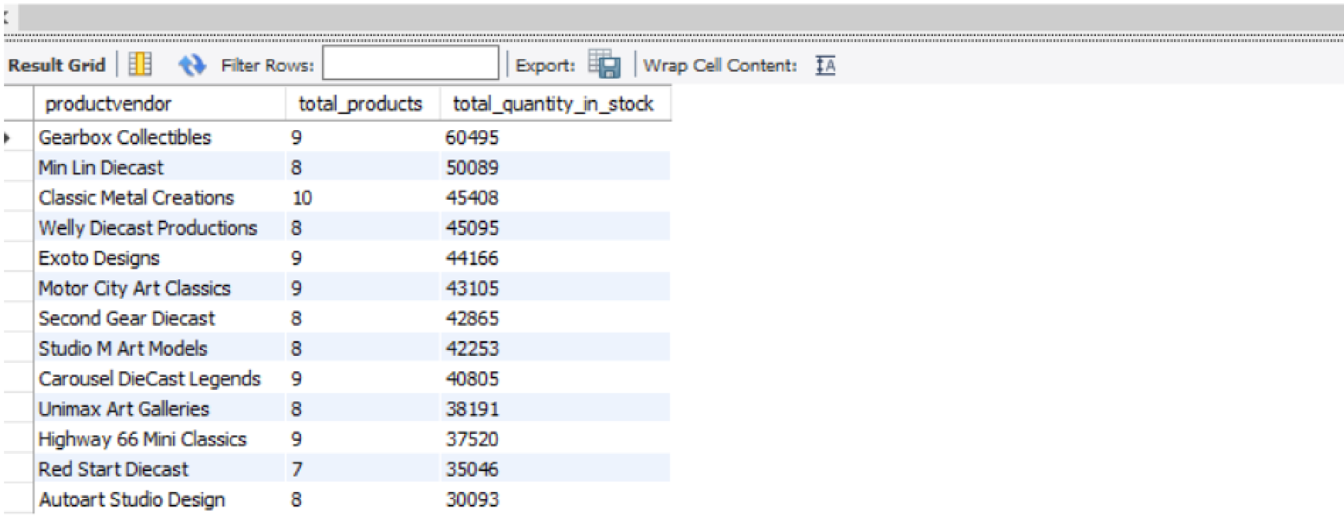
customerName	Purchaser	total_payment
Euro+ Shopping Channel	Diego Freyre	715738.98
Mini Gifts Distributors Ltd.	Susan Nelson	584188.24
Australian Collectors, Co.	Peter Ferguson	180585.07
Muscle Machine Inc	Jeff Young	177913.95
Dragon Souveniers, Ltd.	Eric Natividad	156251.03
Down Under Souveniers, Inc	Mike Graham	154622.08
AV Stores, Co.	Rachel Ashworth	148410.09
Anna's Decorations, Ltd	Anna O'Hara	137034.22
Corporate Gift Ideas Co.	Julie Brown	132340.78
Saveley & Henriot, Co.	Mary Saveley	130305.35
Rovelli Gifts	Giovanni Rovelli	127529.69
Reims Collectables	Paul Henriot	126983.19
La Rochelle Gifts	Janine Labrune	116949.68

- Created a table to identify the top suppliers of classicmodels by there products and stock:

```
select productvendor, count(productname) as total_products,
sum(quantityInStock) as total_quantity_in_stock from products group by
productVendor order by total_quantity_in_stock desc;
```



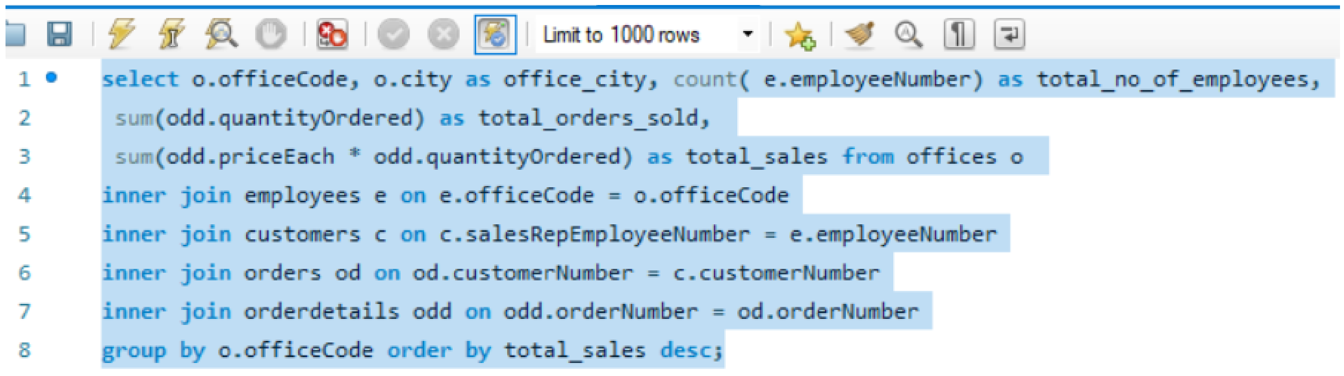
```
# The top suppliers of classicmodels by there products and stock:
select productvendor, count(productname) as total_products, sum(quantityInStock) as total_quantity_
from products group by productVendor order by total_quantity_in_stock desc;
```



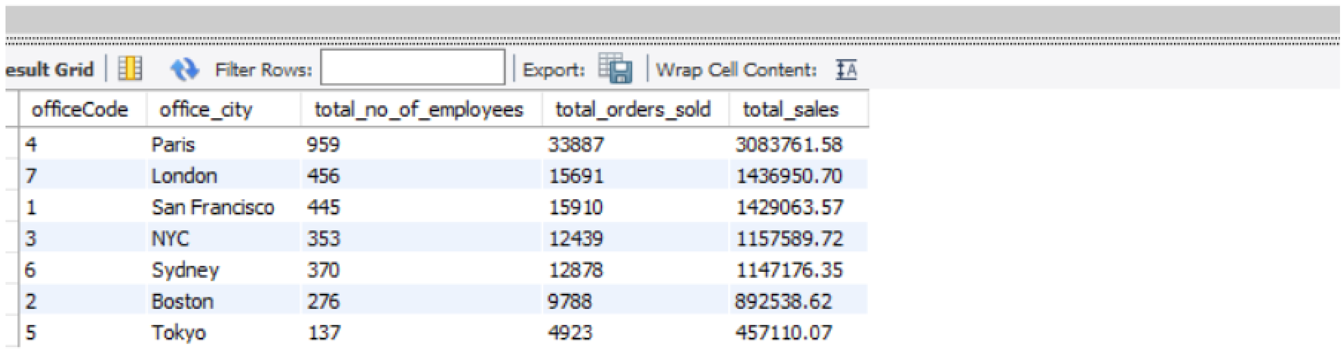
productvendor	total_products	total_quantity_in_stock
Gearbox Collectibles	9	60495
Min Lin Diecast	8	50089
Classic Metal Creations	10	45408
Welly Diecast Productions	8	45095
Exoto Designs	9	44166
Motor City Art Classics	9	43105
Second Gear Diecast	8	42865
Studio M Art Models	8	42253
Carousel DieCast Legends	9	40805
Unimax Art Galleries	8	38191
Highway 66 Mini Classics	9	37520
Red Start Diecast	7	35046
Autoart Studio Design	8	30093

- Created a table on the results grid to examine the offices by their total quantity of products sold and total sales with the number of employees in each office:

```
select o.officeCode, o.city as office_city, count( e.employeeNumber) as
total_no_of_employees,

 sum(odd.quantityOrdered) as total_orders_sold, sum(odd.priceEach *
odd.quantityOrdered) as total_sales from offices o

inner join employees e on e.officeCode = o.officeCode

inner join customers c on c.salesRepEmployeeNumber = e.employeeNumber

inner join orders od on od.customerNumber = c.customerNumber

inner join orderdetails odd on odd.orderNumber = od.orderNumber

group by o.officeCode order by total_sales desc;
```



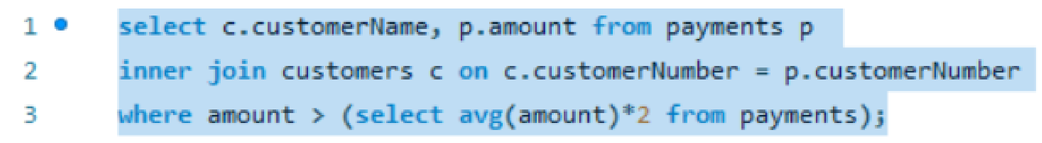
```sql
select o.officeCode, o.city as office_city, count( e.employeeNumber) as total_no_of_employees,
 sum(odd.quantityOrdered) as total_orders_sold,
 sum(odd.priceEach * odd.quantityOrdered) as total_sales from offices o
inner join employees e on e.officeCode = o.officeCode
inner join customers c on c.salesRepEmployeeNumber = e.employeeNumber
inner join orders od on od.customerNumber = c.customerNumber
inner join orderdetails odd on odd.orderNumber = od.orderNumber
group by o.officeCode order by total_sales desc;
```



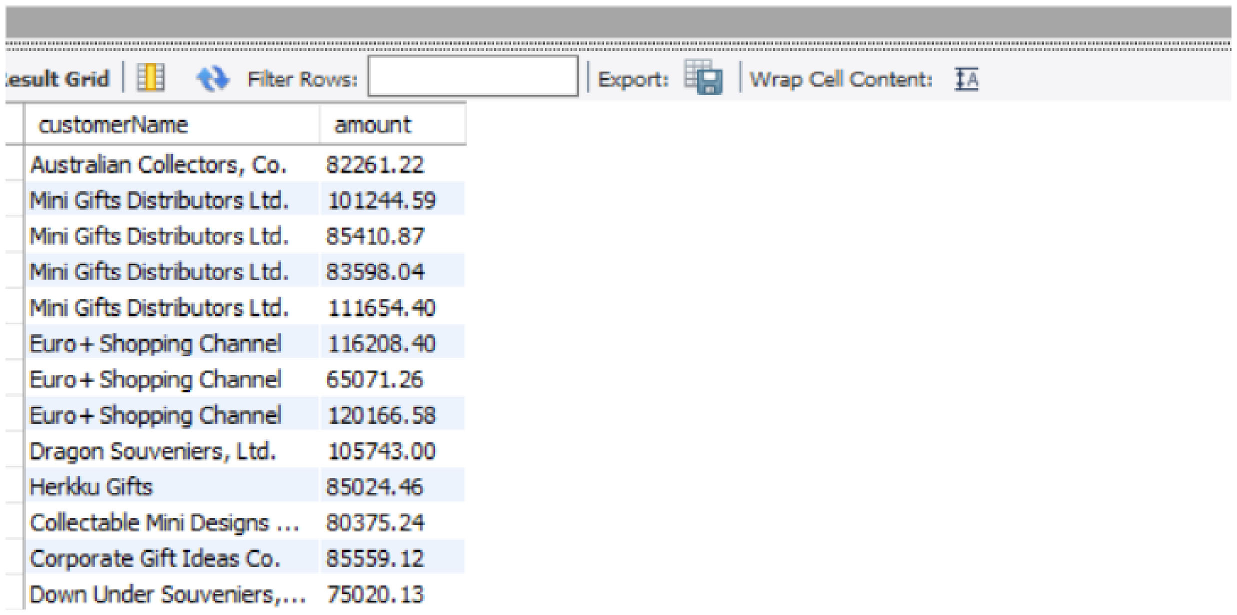
officeCode	office_city	total_no_of_employees	total_orders_sold	total_sales
4	Paris	959	33887	3083761.58
7	London	456	15691	1436950.70
1	San Francisco	445	15910	1429063.57
3	NYC	353	12439	1157589.72
6	Sydney	370	12878	1147176.35
2	Boston	276	9788	892538.62
5	Tokyo	137	4923	457110.07

- Created a table on the results grid to showcase the customers with more than twice the average payment:

```sql
select c.customerName, p.amount from payments p

inner join customers c on c.customerNumber = p.customerNumber

where amount > (select avg(amount)*2 from payments);
```



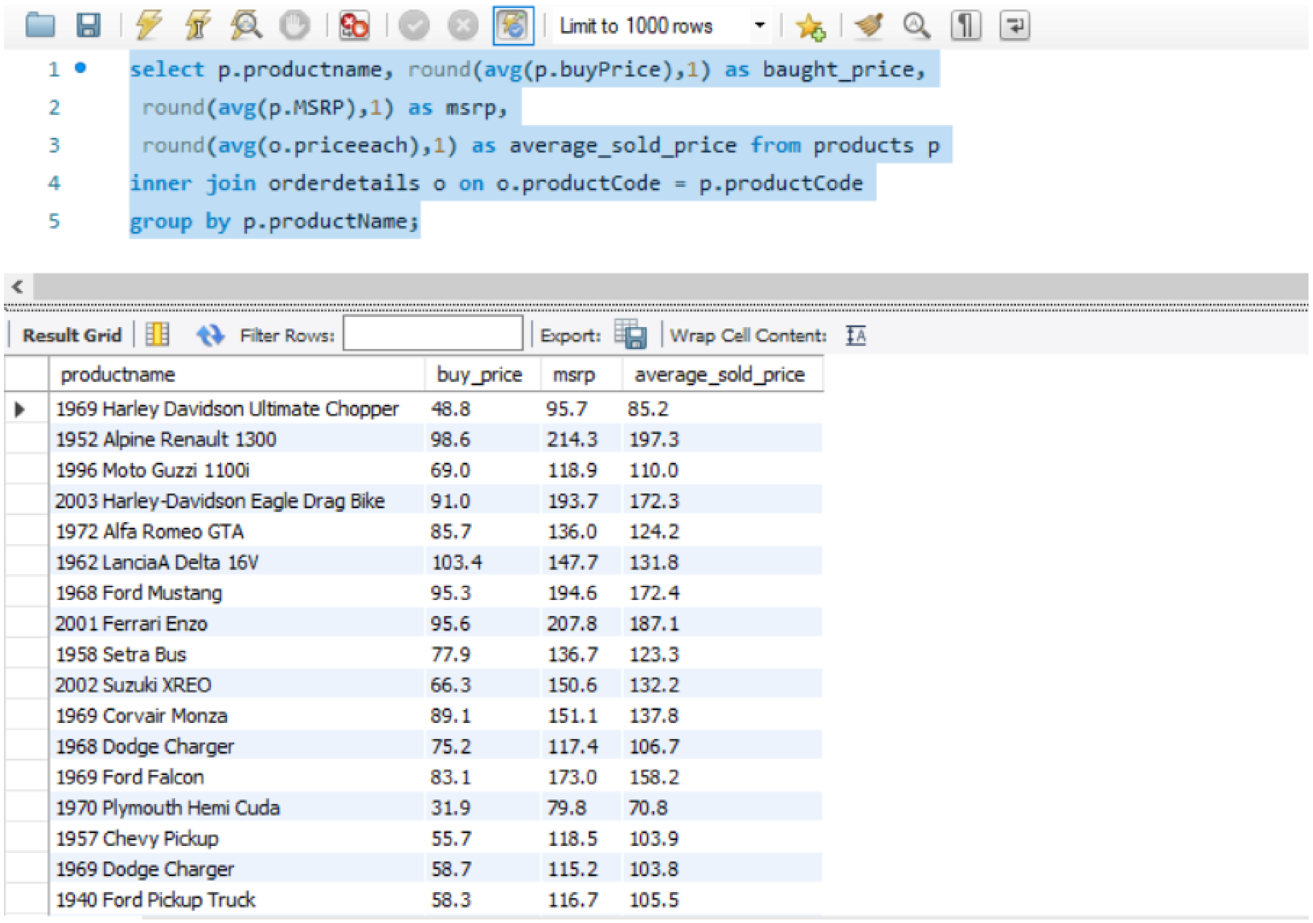
```sql
select c.customerName, p.amount from payments p
inner join customers c on c.customerNumber = p.customerNumber
where amount > (select avg(amount)*2 from payments);
```



customerName	amount
Australian Collectors, Co.	82261.22
Mini Gifts Distributors Ltd.	101244.59
Mini Gifts Distributors Ltd.	85410.87
Mini Gifts Distributors Ltd.	83598.04
Mini Gifts Distributors Ltd.	111654.40
Euro+ Shopping Channel	116208.40
Euro+ Shopping Channel	65071.26
Euro+ Shopping Channel	120166.58
Dragon Souveniers, Ltd.	105743.00
Herkku Gifts	85024.46
Collectable Mini Designs ...	80375.24
Corporate Gift Ideas Co.	85559.12
Down Under Souveniers,...	75020.13

- Created a table on the results grid to compare the buy price vs msrp vs sold price:

```
select p.productname, round(avg(p.buyPrice),1) as baught_price,
 round(avg(p.MSRP),1) as msrp,
 round(avg(o.priceeach),1) as average_sold_price from products p
inner join orderdetails o on o.productCode = p.productCode
group by p.productName;
```

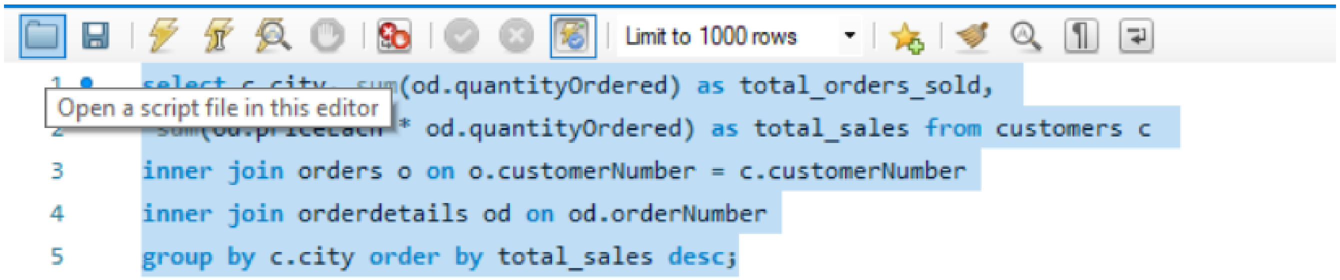


```
select p.productname, round(avg(p.buyPrice),1) as baught_price,
 round(avg(p.MSRP),1) as msrp,
 round(avg(o.priceeach),1) as average_sold_price from products p
inner join orderdetails o on o.productCode = p.productCode
group by p.productName;
```

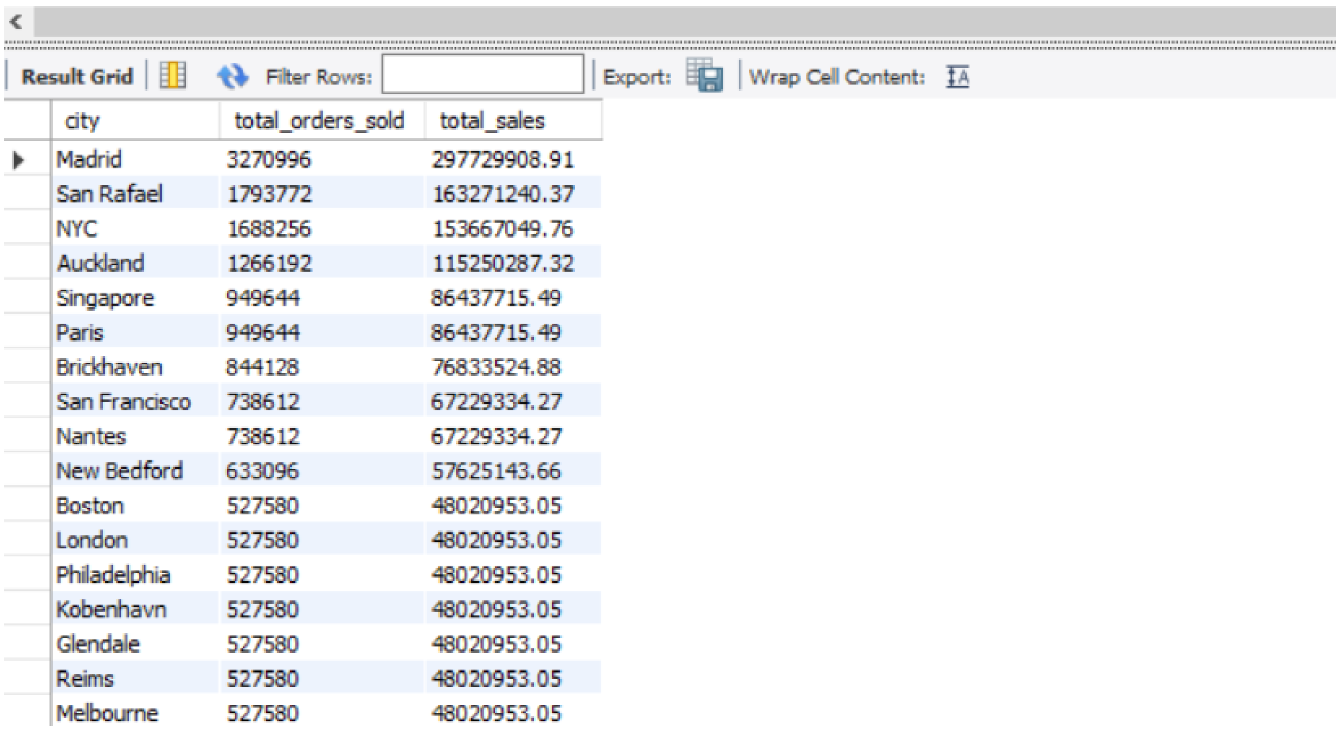
productname	buy_price	msrp	average_sold_price
1969 Harley Davidson Ultimate Chopper	48.8	95.7	85.2
1952 Alpine Renault 1300	98.6	214.3	197.3
1996 Moto Guzzi 1100i	69.0	118.9	110.0
2003 Harley-Davidson Eagle Drag Bike	91.0	193.7	172.3
1972 Alfa Romeo GTA	85.7	136.0	124.2
1962 LanciaA Delta 16V	103.4	147.7	131.8
1968 Ford Mustang	95.3	194.6	172.4
2001 Ferrari Enzo	95.6	207.8	187.1
1958 Setra Bus	77.9	136.7	123.3
2002 Suzuki XREO	66.3	150.6	132.2
1969 Corvair Monza	89.1	151.1	137.8
1968 Dodge Charger	75.2	117.4	106.7
1969 Ford Falcon	83.1	173.0	158.2
1970 Plymouth Hemi Cuda	31.9	79.8	70.8
1957 Chevy Pickup	55.7	118.5	103.9
1969 Dodge Charger	58.7	115.2	103.8
1940 Ford Pickup Truck	58.3	116.7	105.5

- Created a table on the results grid to visualize the city wise sales to examine which city has the highest sales:

```
select c.city, sum(od.quantityOrdered) as total_orders_sold,

 sum(od.priceEach * od.quantityOrdered) as total_sales from customers c

inner join orders o on o.customerNumber = c.customerNumber

inner join orderdetails od on od.orderNumber

group by c.city order by total_sales desc;
```



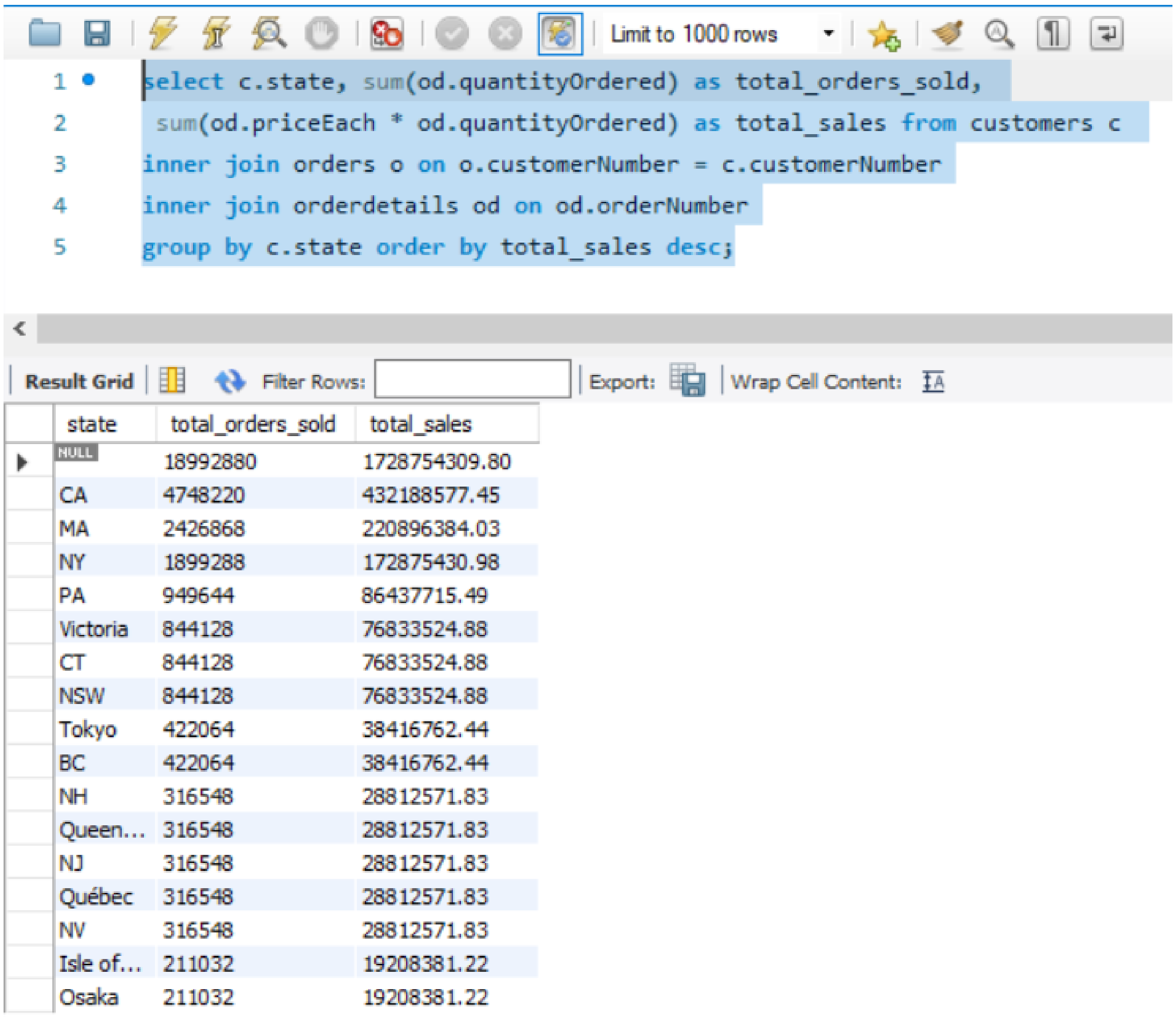
```
select c.city, sum(od.quantityOrdered) as total_orders_sold,
 sum(od.priceEach * od.quantityOrdered) as total_sales from customers c
inner join orders o on o.customerNumber = c.customerNumber
inner join orderdetails od on od.orderNumber
group by c.city order by total_sales desc;
```



city	total_orders_sold	total_sales
Madrid	3270996	297729908.91
San Rafael	1793772	163271240.37
NYC	1688256	153667049.76
Auckland	1266192	115250287.32
Singapore	949644	86437715.49
Paris	949644	86437715.49
Brickhaven	844128	76833524.88
San Francisco	738612	67229334.27
Nantes	738612	67229334.27
New Bedford	633096	57625143.66
Boston	527580	48020953.05
London	527580	48020953.05
Philadelphia	527580	48020953.05
Kobenhavn	527580	48020953.05
Glendale	527580	48020953.05
Reims	527580	48020953.05
Melbourne	527580	48020953.05

- Created a table on the results grid to visualize the state wise sales to examine which state has the highest sales:

```
select c.state, sum(od.quantityOrdered) as total_orders_sold,

 sum(od.priceEach * od.quantityOrdered) as total_sales from customers c

inner join orders o on o.customerNumber = c.customerNumber

inner join orderdetails od on od.orderNumber

group by c.state order by total_sales desc;
```

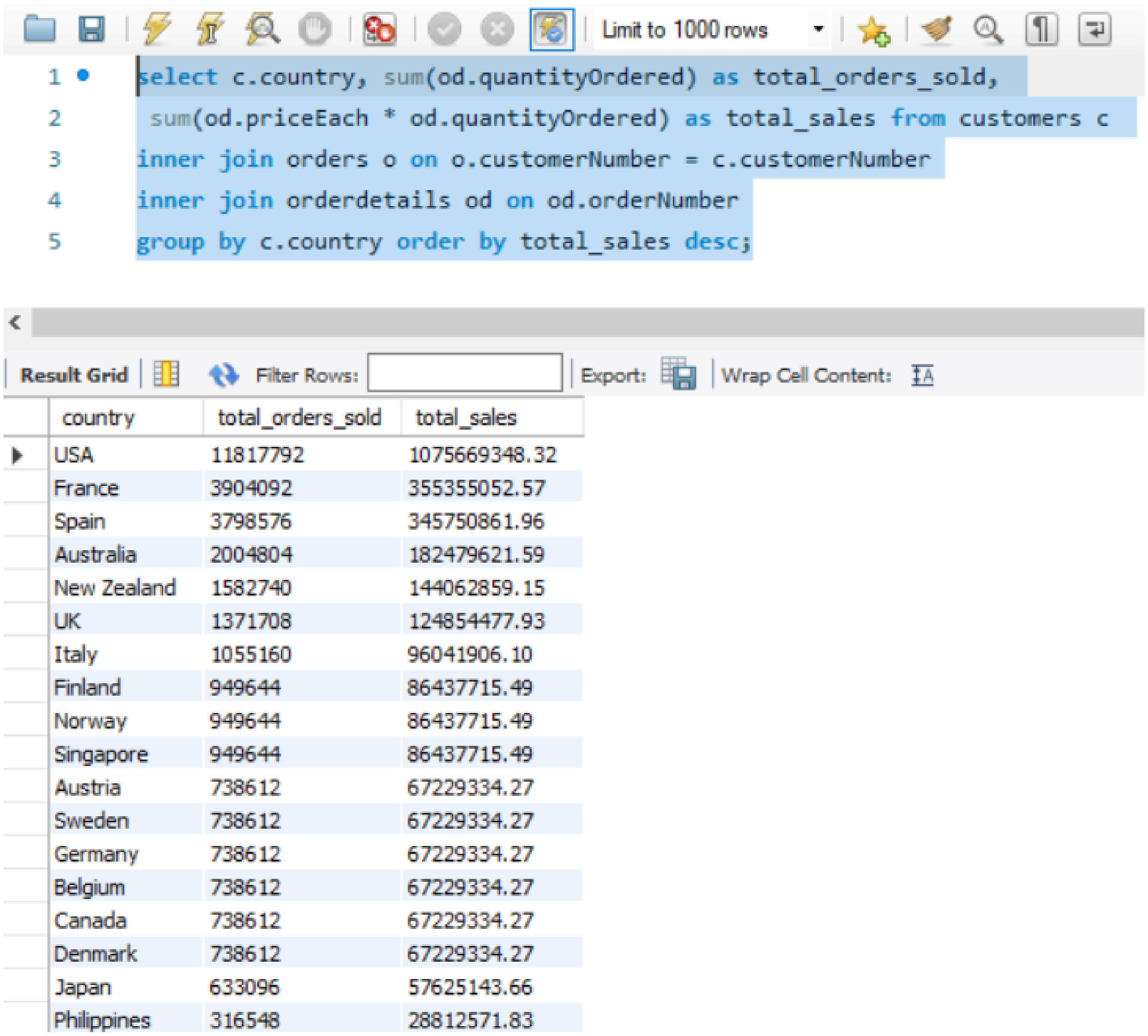


```
select c.state, sum(od.quantityOrdered) as total_orders_sold,
 sum(od.priceEach * od.quantityOrdered) as total_sales from customers c
inner join orders o on o.customerNumber = c.customerNumber
inner join orderdetails od on od.orderNumber
group by c.state order by total_sales desc;
```

state	total_orders_sold	total_sales
NULL	18992880	1728754309.80
CA	4748220	432188577.45
MA	2426868	220896384.03
NY	1899288	172875430.98
PA	949644	86437715.49
Victoria	844128	76833524.88
CT	844128	76833524.88
NSW	844128	76833524.88
Tokyo	422064	38416762.44
BC	422064	38416762.44
NH	316548	28812571.83
Queen...	316548	28812571.83
NJ	316548	28812571.83
Québec	316548	28812571.83
NV	316548	28812571.83
Isle of...	211032	19208381.22
Osaka	211032	19208381.22

- Created a table on the results grid to visualize the country wise sales to examine which country has the highest sales:

```
select c.country, sum(od.quantityOrdered) as total_orders_sold,

 sum(od.priceEach * od.quantityOrdered) as total_sales from customers c

inner join orders o on o.customerNumber = c.customerNumber

inner join orderdetails od on od.orderNumber

group by c.country order by total_sales desc;
```



```
select c.country, sum(od.quantityOrdered) as total_orders_sold,
 sum(od.priceEach * od.quantityOrdered) as total_sales from customers c
inner join orders o on o.customerNumber = c.customerNumber
inner join orderdetails od on od.orderNumber
group by c.country order by total_sales desc;
```

country	total_orders_sold	total_sales
USA	11817792	1075669348.32
France	3904092	355355052.57
Spain	3798576	345750861.96
Australia	2004804	182479621.59
New Zealand	1582740	144062859.15
UK	1371708	124854477.93
Italy	1055160	96041906.10
Finland	949644	86437715.49
Norway	949644	86437715.49
Singapore	949644	86437715.49
Austria	738612	67229334.27
Sweden	738612	67229334.27
Germany	738612	67229334.27
Belgium	738612	67229334.27
Canada	738612	67229334.27
Denmark	738612	67229334.27
Japan	633096	57625143.66
Philippines	316548	28812571.83