Data Analytics Portfolio Project - Sales Data Analysis using SQL

Data Cleaning

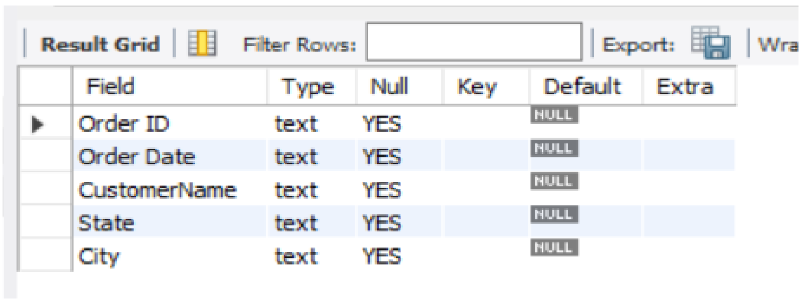
I wanted to check the data type of all columns of all 3 tables and for that i wrote these commands.

DESCRIBE listoforders;

DESCRIBE orderdetails;

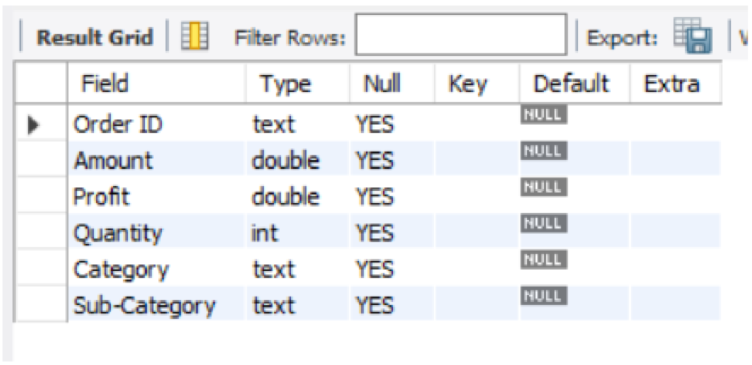
DESCRIBE salestarget;

List of Orders Table



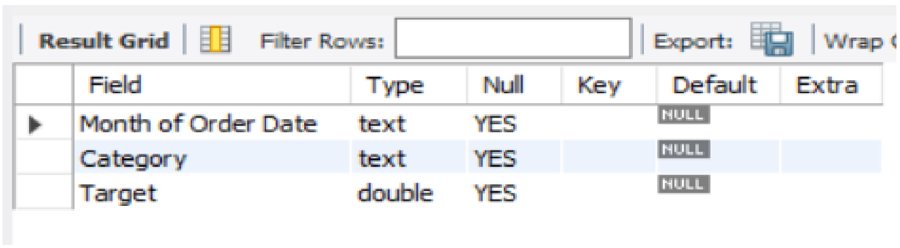
Field	Type	Null	Key	Default	Extra
Order ID	text	YES		NULL	
Order Date	text	YES		NULL	
CustomerName	text	YES		NULL	
State	text	YES		NULL	
City	text	YES		NULL	

Order Details Table



Field	Type	Null	Key	Default	Extra
Order ID	text	YES		NULL	
Amount	double	YES		NULL	
Profit	double	YES		NULL	
Quantity	int	YES		NULL	
Category	text	YES		NULL	
Sub-Category	text	YES		NULL	

Sales Target Table



Field	Type	Null	Key	Default	Extra
Month of Order Date	text	YES		NULL	
Category	text	YES		NULL	
Target	double	YES		NULL	

Removing Spaces from Column Names of "listoforders" table

Column "Order ID"

```
ALTER TABLE listoforders

CHANGE `Order ID` Order_ID text(15);
```

Column "Order Date"

```
ALTER TABLE listoforders

CHANGE `Order Date` Order_Date text(15);
```

In "listoforders" table Date format is in 2 different types so changing the date format to one single format

```
UPDATE listoforders

SET Order_Date = REPLACE(Order_Date, '-', '/');
```

Deleting blank rows from "listoforders" table

```
DELETE FROM listoforders WHERE Order_ID = '' OR Order_Date = '' OR

CustomerName = '' OR

State = '' OR City = '' ;
```

Now in "listoforders" table changing the datatype of Order_Date column from text to date

```
UPDATE listoforders

SET Order_Date = str_to_date(Order_Date, '%d/%m/%Y');
```

Removing Spaces from Column Names of "orderdetails" table

Column Order ID

```
ALTER TABLE orderdetails

CHANGE `Order ID` Order_ID text(15);
```

Column Sub-Category

```
ALTER TABLE orderdetails

CHANGE `Sub-Category` SubCategory text(20);
```

Removing Spaces from Column Names of "salestarget" table

Column Month of Order Date

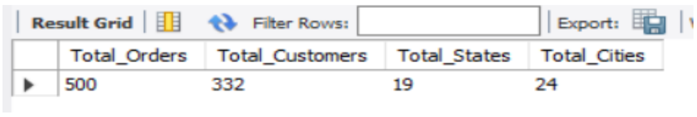
```
ALTER TABLE salestarget

CHANGE `Month of Order Date` Month_of_Order_Date text(15);
```

Data Analysis

1 - Find the number of orders, customers, cities, and states

```
SELECT Count(distinct(Order_ID)) AS Total_Orders,

Count(Distinct(CustomerName)) AS Total_Customers,

Count(Distinct(State)) AS Total_States,

Count(Distinct(City)) AS Total_Cities

FROM

listoforders;
```

Output


Result Grid | Filter Rows: | Export:

	Total_Orders	Total_Customers	Total_States	Total_Cities
▶	500	332	19	24

2 - Find the new customers who made purchases in the year 2019. Only show the top 5 new customers and their respective cities and states. Order the result by the amount they spent.

To solve this problem i am creating a VIEW by combining "listoforders" and "orderdetails" table by common column Order_ID.

Step - 1

Create VIEW combined AS

select L.*, O.Amount, O.Profit, O.Quantity, O.Category, O.SubCategory

FROM listoforders L

inner join

orderdetails O

ON

L.Order_ID = O.Order_ID;

Step - 2

SELECT Distinct CustomerName, City, State, Sum(Amount) AS Sales

FROM combined

WHERE CustomerNAME NOT IN (

SELECT DISTINCT CustomerName

FROM combined

WHERE Order_Date LIKE '2018%')

AND

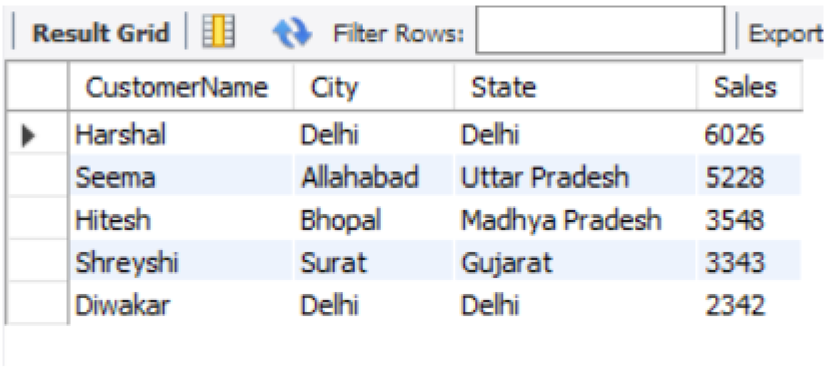
Order_Date LIKE '2019%'

GROUP BY CustomerName, City, State

ORDER BY Sales DESC

Limit 5;

Output



CustomerName	City	State	Sales
Harshal	Delhi	Delhi	6026
Seema	Allahabad	Uttar Pradesh	5228
Hitesh	Bhopal	Madhya Pradesh	3548
Shreyshi	Surat	Gujarat	3343
Diwakar	Delhi	Delhi	2342

An alternative way to solve this problem

SELECT Distinct CustomerName, City, State, Sum(Amount) AS Sales

FROM combined

WHERE CustomerNAME IN (

SELECT DISTINCT CustomerName

FROM combined

WHERE Order_Date LIKE '2019%')

AND

CustomerNAME NOT IN (

SELECT DISTINCT CustomerName

FROM combined

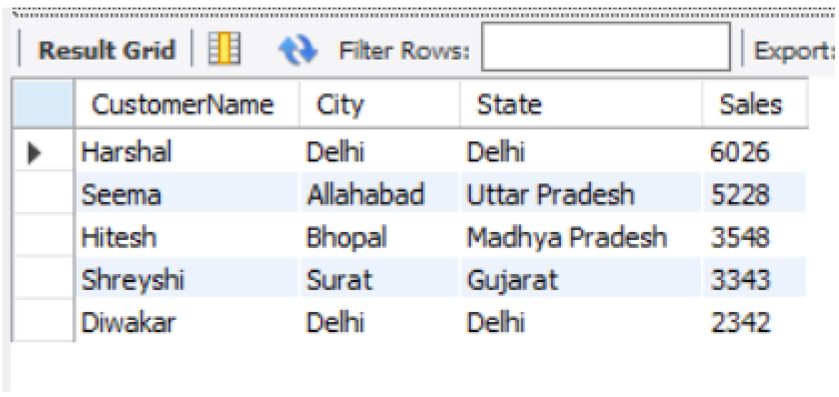
WHERE Order_Date LIKE '2018%')

GROUP BY CustomerName, City, State

ORDER BY Sales DESC

Limit 5;

Output



CustomerName	City	State	Sales
Harshal	Delhi	Delhi	6026
Seema	Allahabad	Uttar Pradesh	5228
Hitesh	Bhopal	Madhya Pradesh	3548
Shreyshi	Surat	Gujarat	3343
Diwakar	Delhi	Delhi	2342

As you can see the result of both queries is exactly the same.

3 - Find the top 10 profitable states & cities so that the company can expand its business. Determine the number of products sold and the number of customers in these top 10 profitable states & cities.

SELECT State, City,

count(distinct(CustomerName)) AS Total_Customers,

SUM(profit) AS Total_Profit,

sum(Quantity) AS Products_Sold

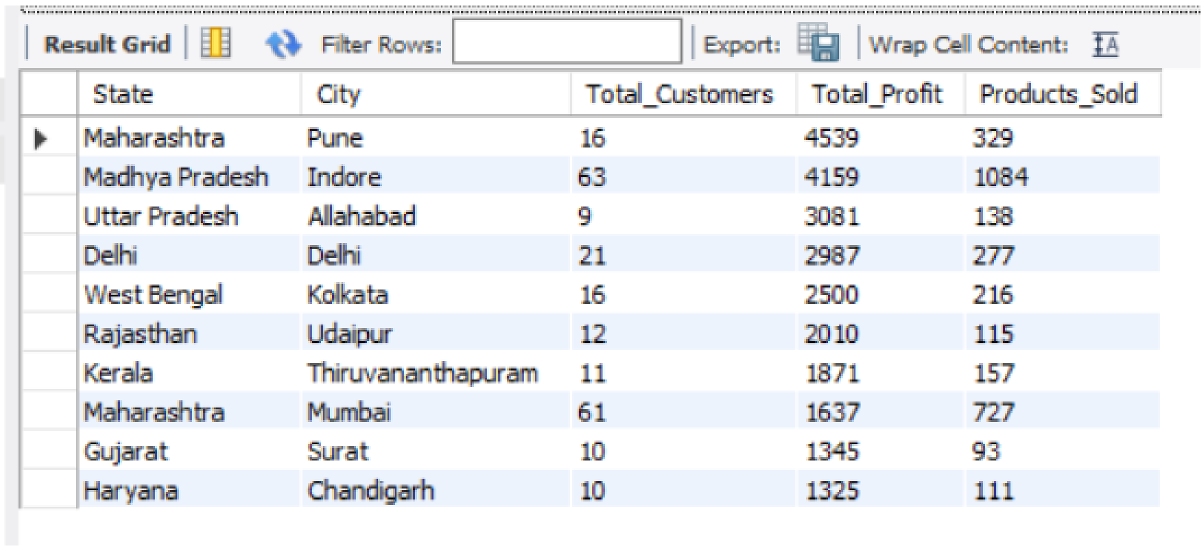
FROM combined

GROUP BY State, City

ORDER BY Total_Profit DESC

Limit 10;

Output



State	City	Total_Customers	Total_Profit	Products_Sold
Maharashtra	Pune	16	4539	329
Madhya Pradesh	Indore	63	4159	1084
Uttar Pradesh	Allahabad	9	3081	138
Delhi	Delhi	21	2987	277
West Bengal	Kolkata	16	2500	216
Rajasthan	Udaipur	12	2010	115
Kerala	Thiruvananthapuram	11	1871	157
Maharashtra	Mumbai	61	1637	727
Gujarat	Surat	10	1345	93
Haryana	Chandigarh	10	1325	111

4 - Display the details (in terms of “order_date”, “order_id”, “State”, and “CustomerName”) for the first order in each state. Order the result by “order_id”.

SELECT Order_ID, Order_Date, CustomerName, State, City

FROM

(SELECT *,

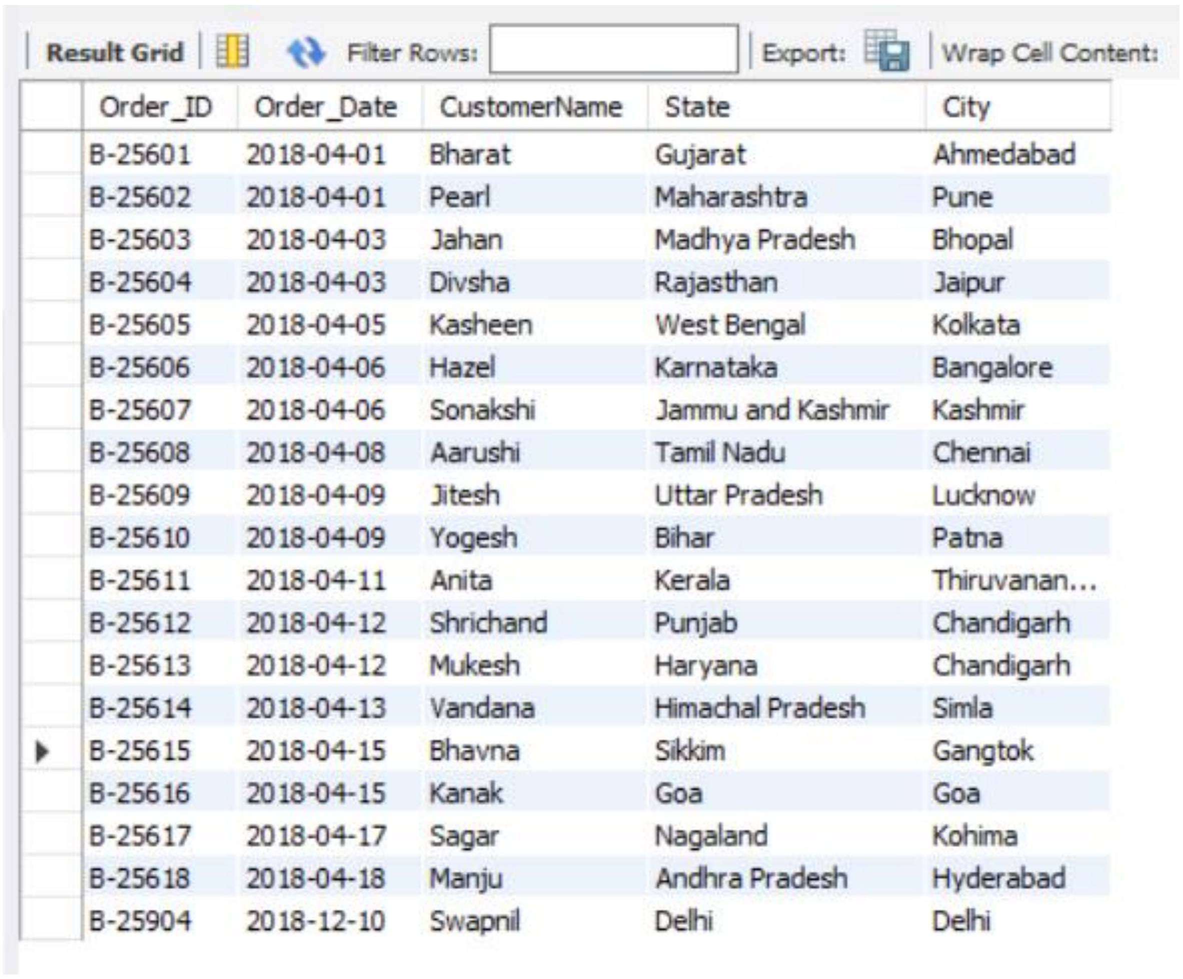
row_number() OVER (partition by State order by order_id) AS RowNumber

FROM combined) AS subquery

WHERE RowNumber = 1

order by order_id;

Output



Order_ID	Order_Date	CustomerName	State	City
B-25601	2018-04-01	Bharat	Gujarat	Ahmedabad
B-25602	2018-04-01	Pearl	Maharashtra	Pune
B-25603	2018-04-03	Jahan	Madhya Pradesh	Bhopal
B-25604	2018-04-03	Divsha	Rajasthan	Jaipur
B-25605	2018-04-05	Kasheen	West Bengal	Kolkata
B-25606	2018-04-06	Hazel	Karnataka	Bangalore
B-25607	2018-04-06	Sonakshi	Jammu and Kashmir	Kashmir
B-25608	2018-04-08	Aarushi	Tamil Nadu	Chennai
B-25609	2018-04-09	Jitesh	Uttar Pradesh	Lucknow
B-25610	2018-04-09	Yogesh	Bihar	Patna
B-25611	2018-04-11	Anita	Kerala	Thiruvanan...
B-25612	2018-04-12	Shrichand	Punjab	Chandigarh
B-25613	2018-04-12	Mukesh	Haryana	Chandigarh
B-25614	2018-04-13	Vandana	Himachal Pradesh	Simla
B-25615	2018-04-15	Bhavna	Sikkim	Gangtok
B-25616	2018-04-15	Kanak	Goa	Goa
B-25617	2018-04-17	Sagar	Nagaland	Kohima
B-25618	2018-04-18	Manju	Andhra Pradesh	Hyderabad
B-25904	2018-12-10	Swapnil	Delhi	Delhi

5 - Determine the number of orders and sales for different days of the week.

select dayname(Order_Date) AS Day,

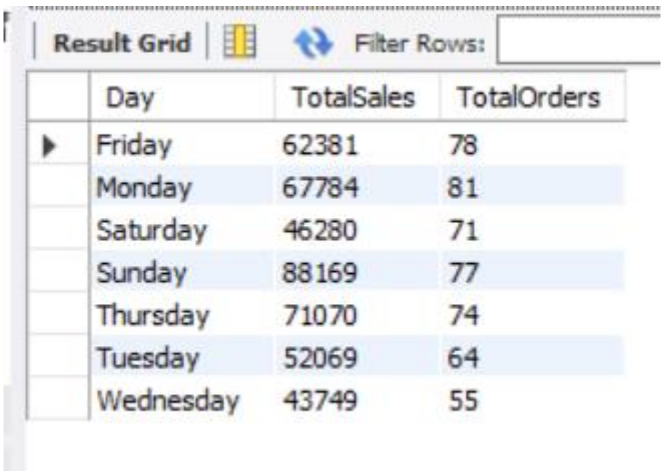
sum(Amount) AS TotalSales,

count(distinct(order_id)) AS TotalOrders

FROM combined

group by Day;

Output



Day	TotalSales	TotalOrders
Friday	62381	78
Monday	67784	81
Saturday	46280	71
Sunday	88169	77
Thursday	71070	74
Tuesday	52069	64
Wednesday	43749	55

6 - Check the monthly profitability and monthly quantity sold to see if there are patterns in the dataset.

select concat(monthname(Order_Date),"-",Year(Order_Date)) AS Month_Year,

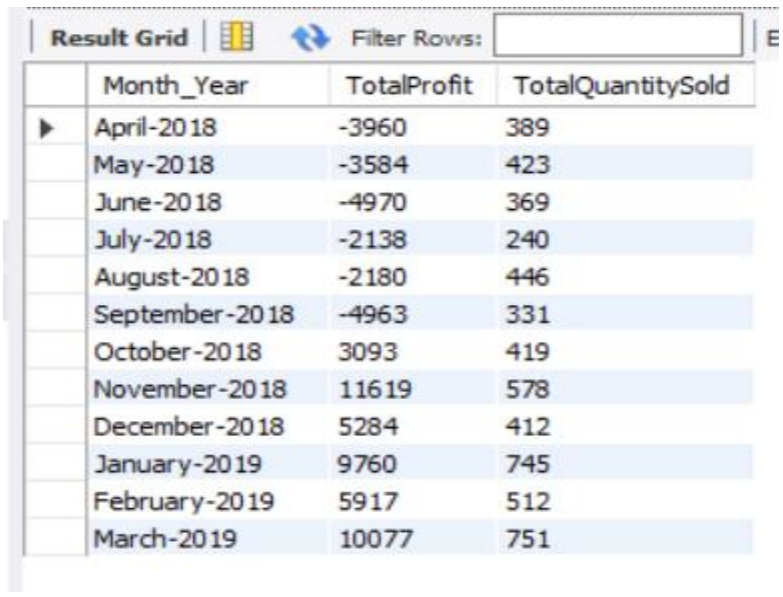
sum(profit) AS TotalProfit,

sum(quantity) AS TotalQuantitySold

FROM combined

group by Month_Year;

Output



Month_Year	TotalProfit	TotalQuantitySold
April-2018	-3960	389
May-2018	-3584	423
June-2018	-4970	369
July-2018	-2138	240
August-2018	-2180	446
September-2018	-4963	331
October-2018	3093	419
November-2018	11619	578
December-2018	5284	412
January-2019	9760	745
February-2019	5917	512
March-2019	10077	751

7 - Determine the number of times that salespeople hit or failed to hit the sales target for each category.

Step - 1

```
CREATE temporary TABLE table1 (

select concat(substr(monthname(order_date),1,3),"-",substr(YEAR(order_date),3,2)) AS Month_Year,

sum(amount) AS Sales, category

from combined

group by Month_Year, category

);
```

Step - 2

```
Create temperory TABLE table2

select s.Month_of_Order_Date, s.Category, s.Target, t.sales,

CASE

WHEN t.sales >= s.Target THEN "Hit"

WHEN t.sales < s.Target THEN "Fail"

END AS Hit_or_Fail

FROM salestarget s

join table1 t

ON
```

s.Month_of_Order_Date = t.Month_Year AND s.Category = t.Category;

Step - 3

select category,

count(case

when Hit_or_Fail = "Hit" THEN "Hit"

END) AS Hit,

count(case

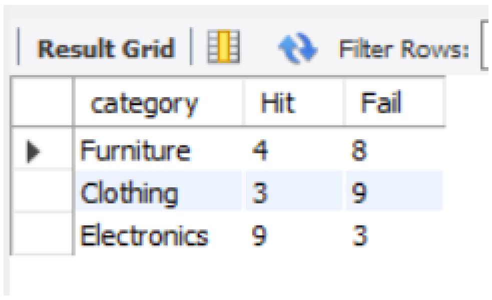
when Hit_or_Fail = "Fail" THEN "Fail"

END) AS Fail

FROM table2

group by category;

Output



category	Hit	Fail
Furniture	4	8
Clothing	3	9
Electronics	9	3

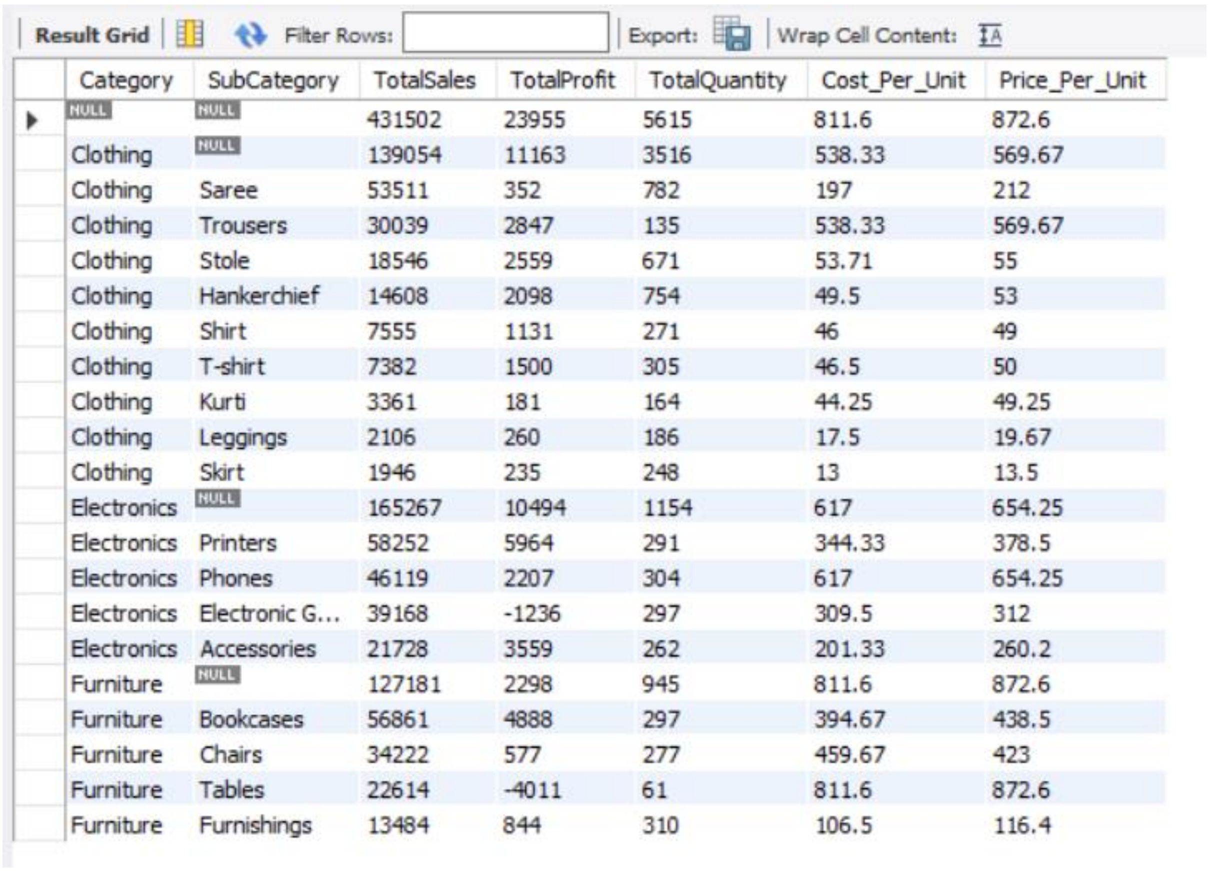
8 - Find the total sales, total profit, and total quantity sold for each category and sub-category. Return the maximum cost and maximum price for each sub-category too.

SELECT Category, SubCategory,

sum(Amount) AS TotalSales,

```
sum(Profit) AS TotalProfit,

sum(Quantity) AS TotalQuantity,

round(max((Amount-Profit)/Quantity),2) AS Cost_Per_Unit,

round(max(Amount/Quantity),2) AS Price_Per_Unit

FROM orderdetails

group by Category, SubCategory

WITH ROLLUP

order by Category, TotalSales DESC;
```

Output



Category	SubCategory	TotalSales	TotalProfit	TotalQuantity	Cost_Per_Unit	Price_Per_Unit
NULL	NULL	431502	23955	5615	811.6	872.6
Clothing	NULL	139054	11163	3516	538.33	569.67
Clothing	Saree	53511	352	782	197	212
Clothing	Trousers	30039	2847	135	538.33	569.67
Clothing	Stole	18546	2559	671	53.71	55
Clothing	Hankerchief	14608	2098	754	49.5	53
Clothing	Shirt	7555	1131	271	46	49
Clothing	T-shirt	7382	1500	305	46.5	50
Clothing	Kurti	3361	181	164	44.25	49.25
Clothing	Leggings	2106	260	186	17.5	19.67
Clothing	Skirt	1946	235	248	13	13.5
Electronics	NULL	165267	10494	1154	617	654.25
Electronics	Printers	58252	5964	291	344.33	378.5
Electronics	Phones	46119	2207	304	617	654.25
Electronics	Electronic G...	39168	-1236	297	309.5	312
Electronics	Accessories	21728	3559	262	201.33	260.2
Furniture	NULL	127181	2298	945	811.6	872.6
Furniture	Bookcases	56861	4888	297	394.67	438.5
Furniture	Chairs	34222	577	277	459.67	423
Furniture	Tables	22614	-4011	61	811.6	872.6
Furniture	Furnishings	13484	844	310	106.5	116.4

9 - Comparing total sales of each category by the years 2019 and 2018.

Step - 1

```
Create temporary table sales2019 (

select Category,

year(Order_Date) AS Year,

sum(Amount) AS TotalSales,

sum(Quantity) AS TotalUnitsOrdered

FROM combined

WHERE Order_Date LIKE "2019%"

GROUP BY Category, Year

ORDER BY Year, Category

);
```

Step - 2

```
Create temporary table sales2018 (

select Category,

year(Order_Date) AS Year,

sum(Amount) AS TotalSales,

sum(Quantity) AS TotalUnitsOrdered

FROM combined
```

WHERE Order_Date LIKE "2018%"

GROUP BY Category, Year

ORDER BY Year, Category

);

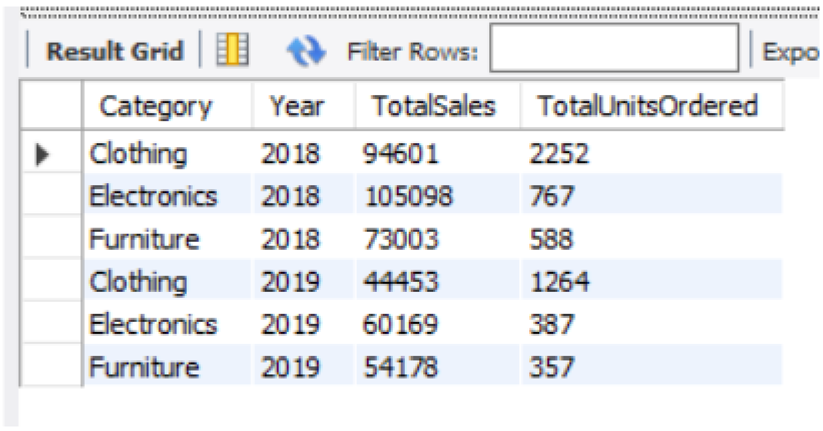
Step - 3

Select * FROM sales2018

UNION ALL

Select * FROM sales2019;

Output



Category	Year	TotalSales	TotalUnitsOrdered
Clothing	2018	94601	2252
Electronics	2018	105098	767
Furniture	2018	73003	588
Clothing	2019	44453	1264
Electronics	2019	60169	387
Furniture	2019	54178	357

10 - Comparing total sales of each month with sales of the previous month.

Step - 1

SET sql_mode=(SELECT REPLACE(@@sql_mode,'ONLY_FULL_GROUP_BY',''));

Step - 2

ALTER TABLE combined

ADD Column Month_Year Date;

```
ALTER TABLE combined

Modify Month_Year varchar(100);
```

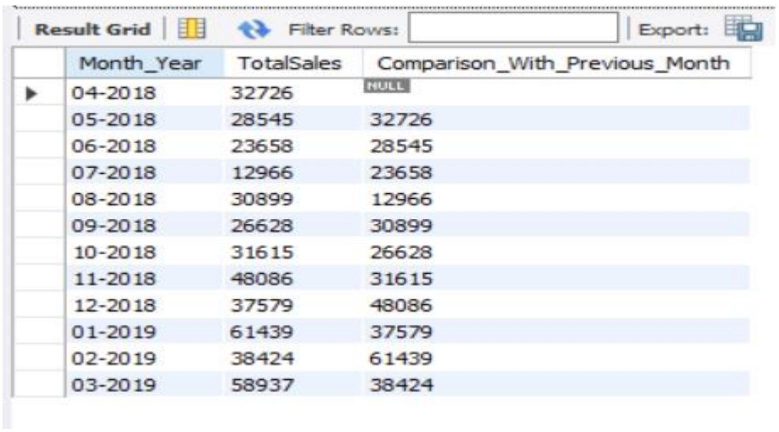
Step - 3

```
SET Month_Year = date_format(Order_Date, '%m-%Y');
```

Step - 4

```
select

Month_Year,

sum(Amount) AS TotalSales,

lag(sum(Amount),1) OVER (ORDER BY Right(Month_Year, 4) ASC) AS
Comparison_With_Previous_Month

FROM combined

group by Month_Year

order by Right(Month_Year, 4) ASC;
```

Output



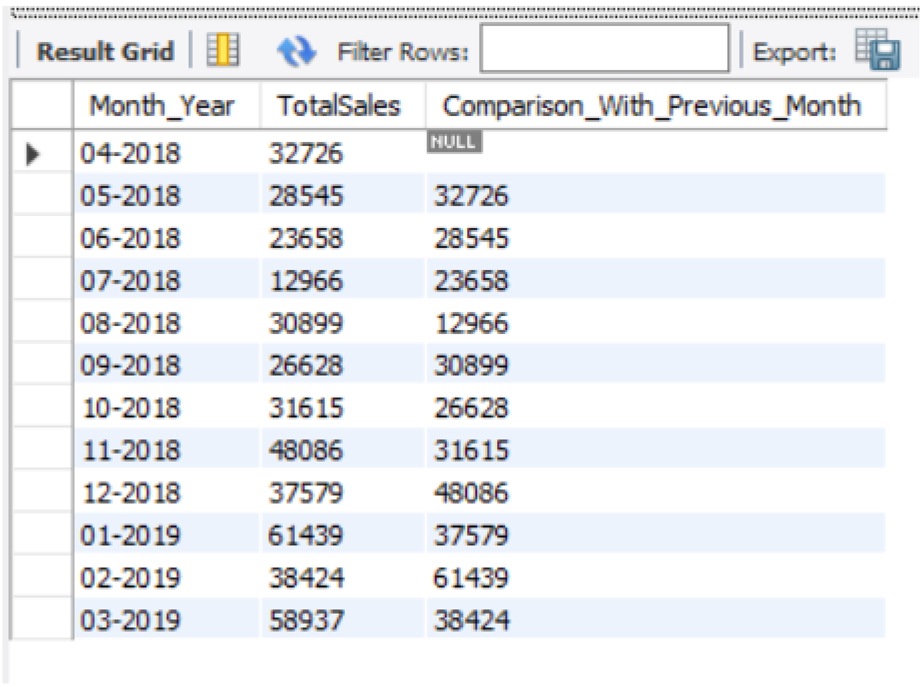
Result Grid | Filter Rows: | Export:

	Month_Year	TotalSales	Comparison_With_Previous_Month
▶	04-2018	32726	NULL
	05-2018	28545	32726
	06-2018	23658	28545
	07-2018	12966	23658
	08-2018	30899	12966
	09-2018	26628	30899
	10-2018	31615	26628
	11-2018	48086	31615
	12-2018	37579	48086
	01-2019	61439	37579
	02-2019	38424	61439
	03-2019	58937	38424

Alternative Way to solve the same problem.

```
SELECT

  Month_Year,

  TotalSales,

  LAG(TotalSales, 1) OVER (ORDER BY Right(Month_Year, 4) ASC) AS
Comparison_With_Previous_Month

FROM (

  SELECT

    DATE_FORMAT(Order_Date, '%m-%Y') AS Month_Year,

    SUM(Amount) AS TotalSales

  FROM

    combined

  GROUP BY

    Month_Year

) AS sales_by_month

ORDER BY

  Right(Month_Year, 4) ASC;
```

Output



Result Grid | Filter Rows: | Export:

Month_Year	TotalSales	Comparison_With_Previous_Month
04-2018	32726	NULL
05-2018	28545	32726
06-2018	23658	28545
07-2018	12966	23658
08-2018	30899	12966
09-2018	26628	30899
10-2018	31615	26628
11-2018	48086	31615
12-2018	37579	48086
01-2019	61439	37579
02-2019	38424	61439
03-2019	58937	38424

As you can see both queries are returning the same result. Similarly if you want to compare sales of a month with next month then just replace "lag" with "lead"

11 - Calculating Percentage of Total Sales by Category

SELECT Category,

SUM(Amount) AS Total_Sales,

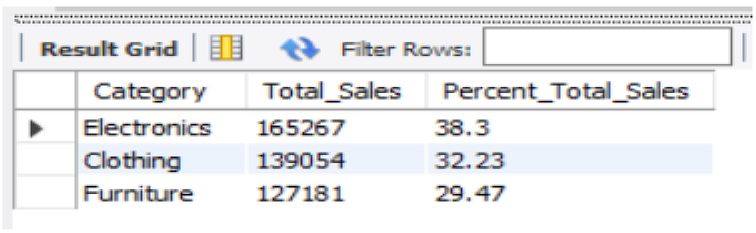
round((SUM(Amount) * 100.0 / (SELECT SUM(Amount) FROM orderdetails)), 2) AS Percent_Total_Sales

FROM orderdetails

GROUP BY Category

order by Percent_Total_Sales DESC;

Output



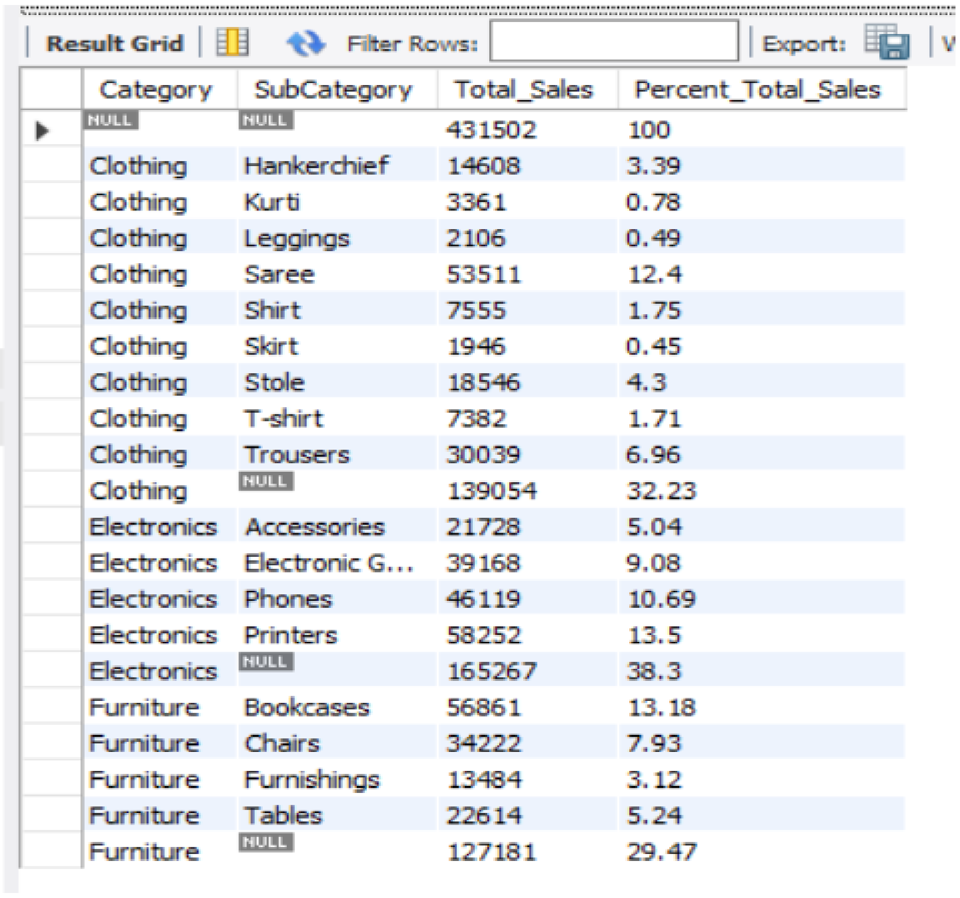
Result Grid | Filter Rows:

Category	Total_Sales	Percent_Total_Sales
Electronics	165267	38.3
Clothing	139054	32.23
Furniture	127181	29.47

12 - Calculating the Percentage of Total Sales by Category and SubCategory and also calculating subtotals.

```
SELECT Category, SubCategory,

SUM(Amount) AS Total_Sales,

round((SUM(Amount) * 100.0 / (SELECT SUM(Amount) FROM orderdetails)), 2) AS
Percent_Total_Sales

FROM orderdetails

GROUP BY Category, SubCategory

WITH rollup

order by Category;
```

Output

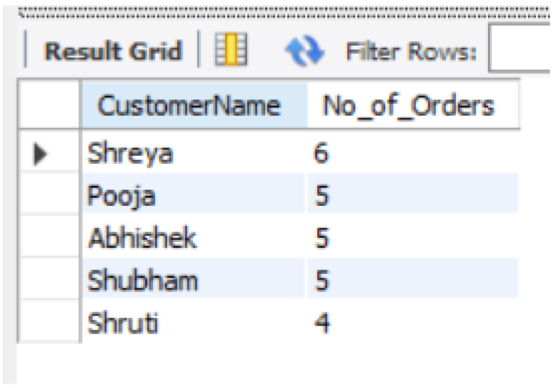


Category	SubCategory	Total_Sales	Percent_Total_Sales
NULL	NULL	431502	100
Clothing	Hankerchief	14608	3.39
Clothing	Kurti	3361	0.78
Clothing	Leggings	2106	0.49
Clothing	Saree	53511	12.4
Clothing	Shirt	7555	1.75
Clothing	Skirt	1946	0.45
Clothing	Stole	18546	4.3
Clothing	T-shirt	7382	1.71
Clothing	Trousers	30039	6.96
Clothing	NULL	139054	32.23
Electronics	Accessories	21728	5.04
Electronics	Electronic G...	39168	9.08
Electronics	Phones	46119	10.69
Electronics	Printers	58252	13.5
Electronics	NULL	165267	38.3
Furniture	Bookcases	56861	13.18
Furniture	Chairs	34222	7.93
Furniture	Furnishings	13484	3.12
Furniture	Tables	22614	5.24
Furniture	NULL	127181	29.47

13 - Top 5 Customers with the most no. of orders

```
select CustomerName, count(distinct(Order_ID)) AS No_of_Orders

FROM

combined_v2

group by CustomerName

order by No_of_Orders DESC

Limit 5;
```

Output

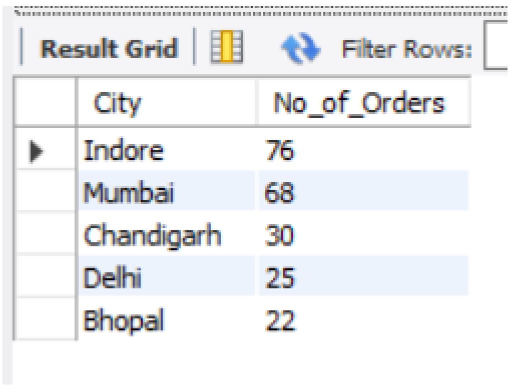


CustomerName	No_of_Orders
Shreya	6
Pooja	5
Abhishek	5
Shubham	5
Shruti	4

14 - Top 5 Cities with the most no. of orders

```
select City, count(distinct(Order_ID)) AS No_of_Orders

FROM

combined_v2

group by City

order by No_of_Orders DESC

Limit 5;
```

Output



	City	No_of_Orders
▶	Indore	76
	Mumbai	68
	Chandigarh	30
	Delhi	25
	Bhopal	22

15 - Top 5 States with the most no. of orders

select State, count(distinct(Order_ID)) AS No_of_Orders

FROM

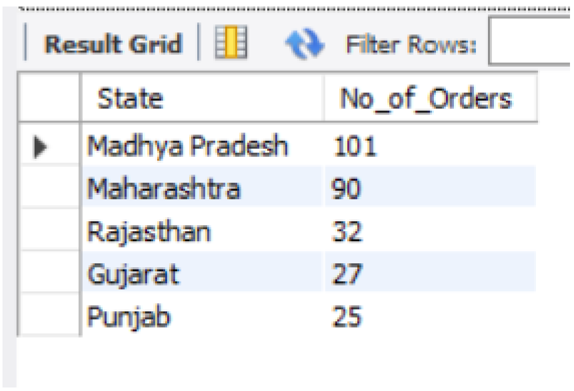
combined_v2

group by State

order by No_of_Orders DESC

Limit 5;

Output



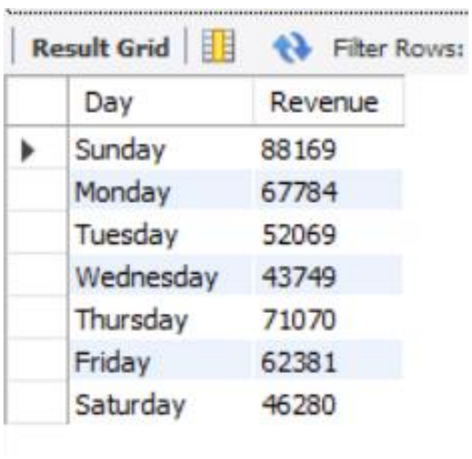
	State	No_of_Orders
▶	Madhya Pradesh	101
	Maharashtra	90
	Rajasthan	32
	Gujarat	27
	Punjab	25

16 - Find the total revenue for each day of the week

select dayname(Order_Date) AS Day, sum(Amount) AS Revenue

```
from combined

group by Day

order by

CASE Day

WHEN 'Sunday' THEN 1

WHEN 'Monday' THEN 2

WHEN 'TUESDAY' THEN 3

WHEN 'WEDNESDAY' THEN 4

WHEN 'THURSDAY' THEN 5

WHEN 'FRIDAY' THEN 6

WHEN 'SATURDAY' THEN 7

END;
```

Output

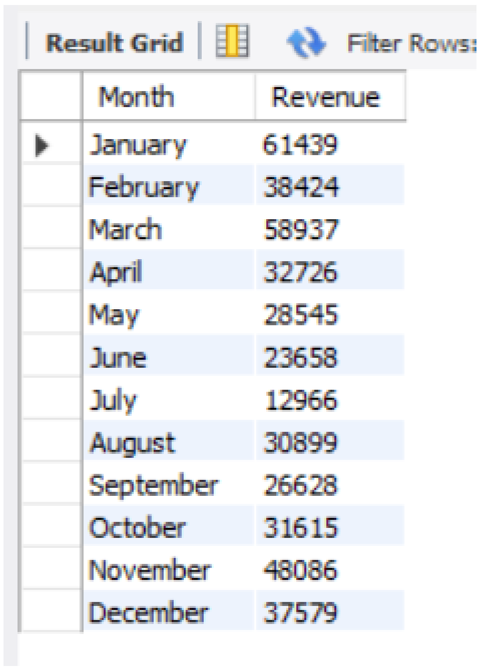


Day	Revenue
Sunday	88169
Monday	67784
Tuesday	52069
Wednesday	43749
Thursday	71070
Friday	62381
Saturday	46280

17 - Find the Total revenue generated in each month

```
select monthname(Order_Date) AS Month, sum(Amount) AS Revenue

from combined

group by Month

order by

CASE Month

WHEN 'January' THEN 1

WHEN 'February' THEN 2

WHEN 'March' THEN 3

WHEN 'April' THEN 4

WHEN 'May' THEN 5

WHEN 'June' THEN 6

WHEN 'July' THEN 7

WHEN 'August' THEN 8

WHEN 'September' THEN 9

WHEN 'October' THEN 10

WHEN 'November' THEN 11

WHEN 'December' THEN 12

END;
```

Output



Month	Revenue
January	61439
February	38424
March	58937
April	32726
May	28545
June	23658
July	12966
August	30899
September	26628
October	31615
November	48086
December	37579

18 - Total profit by each month

SELECT monthname(Order_Date) AS Month_Name,

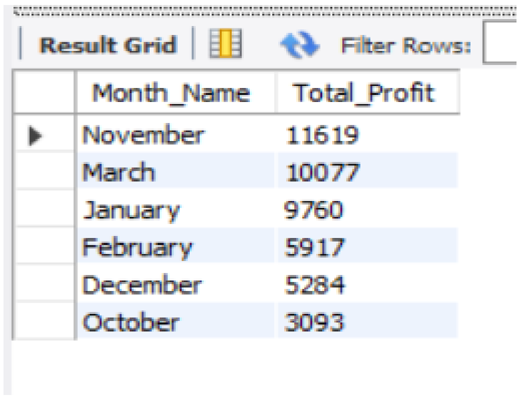
sum(profit) AS Total_Profit

from combined

group by Month_Name

having Total_Profit > 0

order by Total_Profit DESC;



Month_Name	Total_Profit
November	11619
March	10077
January	9760
February	5917
December	5284
October	3093

19 - Total loss by each month

SELECT monthname(Order_Date) AS Month_Name,

sum(profit) AS Total_Profit

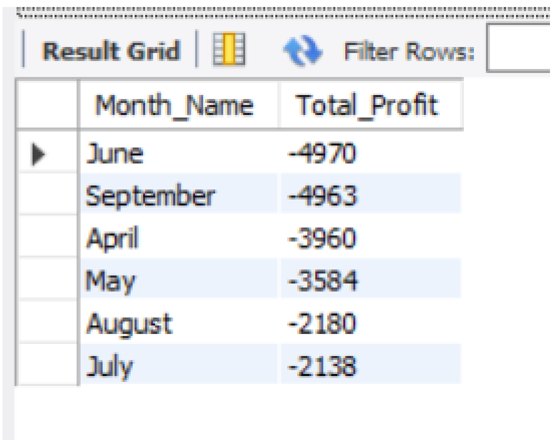
from combined

group by Month_Name

having Total_Profit < 0

order by Total_Profit;

Output



Month_Name	Total_Profit
June	-4970
September	-4963
April	-3960
May	-3584
August	-2180
July	-2138

20 - Top 5 dates on which the highest revenue was generated

select Order_Date as Date, sum(Amount) AS Revenue

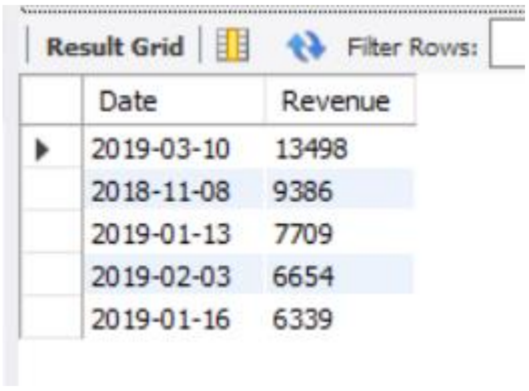
from combined

group by Date

order by Revenue DESC

limit 5;

Output



Date	Revenue
2019-03-10	13498
2018-11-08	9386
2019-01-13	7709
2019-02-03	6654
2019-01-16	6339

21 - Top 2 sub-categories which generated the most revenue from each category and how much percent is their contribution in total revenue brought in by their main category

Step - 1

```
Create temporary table clothing (

select Category, Subcategory,

sum(Amount) AS Revenue,

concat(round((sum(amount)*100/(select sum(amount) from combined WHERE category =
'Clothing')),2), "%") AS Percentage_Revenue

from combined

WHERE Category = 'Clothing'

group by Category, Subcategory

order by Revenue DESC

Limit 2

);
```

Step - 2

```
Create temporary table Electronics (
select Category, Subcategory,
sum(Amount) AS Revenue,
concat(round((sum(amount)*100/(select sum(amount) from combined WHERE category = 'Electronics')),2), "%") AS Percentage_Revenue
from combined
WHERE Category = 'Electronics'
group by Category, Subcategory
order by Revenue DESC
Limit 2
);
```

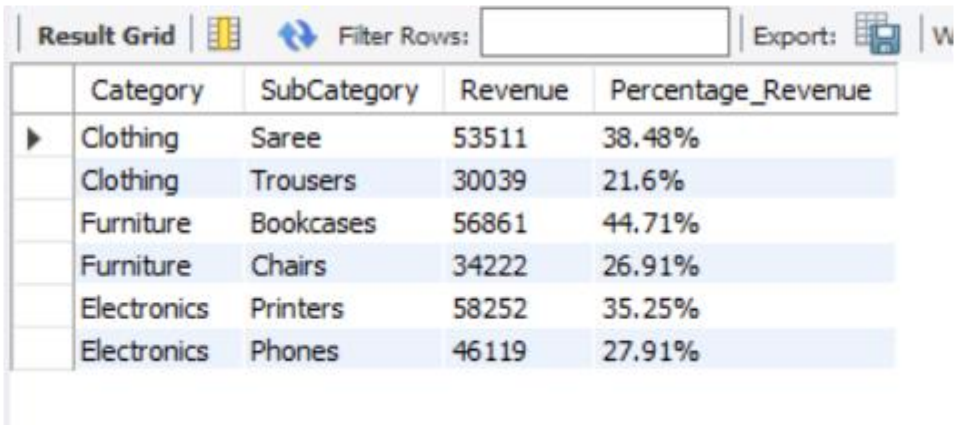
Step - 3

```
Create temporary table furniture (
select Category, Subcategory,
sum(Amount) AS Revenue,
concat(round((sum(amount)*100/(select sum(amount) from combined WHERE category = 'Furniture')),2), "%") AS Percentage_Revenue
from combined
WHERE Category = 'Furniture'
group by Category, Subcategory
order by Revenue DESC

Limit 2

);
```

Step - 4

```
select * from clothing

UNION ALL

select * from furniture

UNION ALL

select * from electronics;
```

Output



Result Grid | Filter Rows: | Export:

Category	SubCategory	Revenue	Percentage_Revenue
Clothing	Saree	53511	38.48%
Clothing	Trousers	30039	21.6%
Furniture	Bookcases	56861	44.71%
Furniture	Chairs	34222	26.91%
Electronics	Printers	58252	35.25%
Electronics	Phones	46119	27.91%