Business Case: Target SQL

Below Insights and analysis are provided by Khizer Ismail a mentee at scaler as part of their SQL curriculum. These are just for educational and learning purpose.

Detailed data insights and recommendations provided on the given data set of Target Brazil.

Raw data sets received:

Below 8 csv files of Target Brazil:

orders.csv
payments.csv
products.csv
sellers.csv
customers.csv
geolocation.csv
order_items.csv
order_reviews.csv

Database tool used for this project: MySQL workbench 8.0

Visualization tool used(optional): Tableau

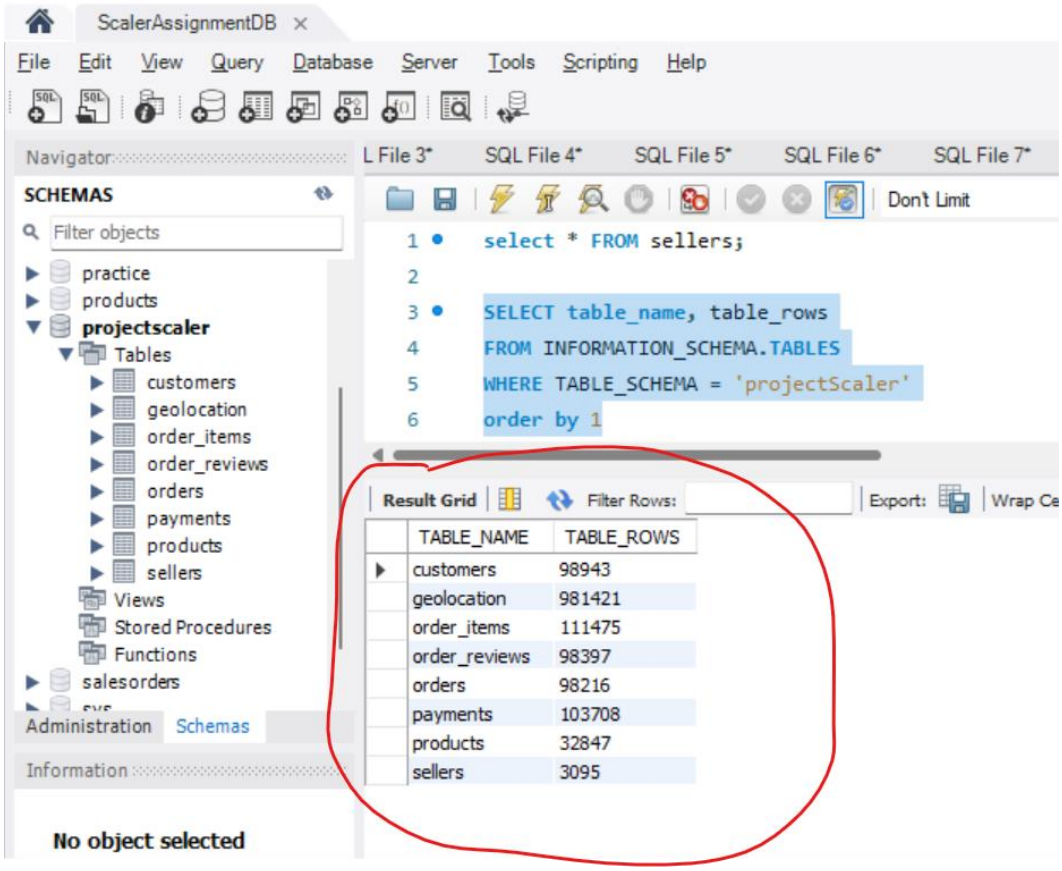
Other tools used: MS excel, MS paint.

1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset

Task: 1.1. Data type of columns in a table

All the provided 8 csv files were successfully loaded into MySQL workbench using MySQL CLI.

Below you can see the number of records for each data set by running the SQL query as highlighted-



Then a check-up was done on the data types and precision/character lengths of each table's column. You can see below the SQL query used and its result in the tabular format-

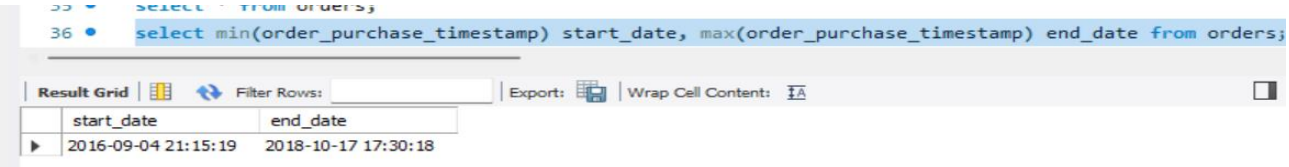
```
SELECT
TABLE_NAME, ORDINAL_POSITION, COLUMN_NAME, DATA_TYPE
,CHARACTER_MAXIMUM_LENGTH, NUMERIC_PRECISION
FROM INFORMATION_SCHEMA.COLUMNS
 WHERE table_schema = 'projectScaler'
 ORDER BY 1,2
```

TABLE_NAME	ORDINAL_POSITION	COLUMN_NAME	DATA_TYPE	CHARACTER_MAXIMUM_LENGTH	NUMERIC_PRECISION
customers	1	customer_id	text	65535	NULL
customers	2	customer_unique_id	text	65535	NULL
customers	3	customer_zip_code_prefix	text	65535	NULL
customers	4	customer_city	text	65535	NULL
customers	5	customer_state	text	65535	NULL
geolocation	1	geolocation_zip_code_prefix	text	65535	NULL
geolocation	2	geolocation_lat	double	NULL	22
geolocation	3	geolocation_lng	double	NULL	22
geolocation	4	geolocation_city	text	65535	NULL
geolocation	5	geolocation_state	text	65535	NULL
order_items	1	order_id	text	65535	NULL
order_items	2	order_item_id	int	NULL	10
order_items	3	product_id	text	65535	NULL
order_items	4	seller_id	text	65535	NULL
order_items	5	shipping_limit_date	text	65535	NULL
order_items	6	price	double	NULL	22
order_items	7	freight_value	double	NULL	22
order_reviews	1	review_id	text	65535	NULL
order_reviews	2	order_id	text	65535	NULL
order_reviews	3	review_score	int	NULL	10
order_reviews	4	review_comment_title	text	65535	NULL
order_reviews	5	review_creation_date	text	65535	NULL
order_reviews	6	review_answer_timestamp	text	65535	NULL
orders	1	order_id	text	65535	NULL
orders	2	customer_id	text	65535	NULL
orders	3	order_status	text	65535	NULL
orders	4	order_purchase_timestamp	text	65535	NULL
orders	5	order_approved_at	text	65535	NULL
orders	6	order_delivered_carrier_date	text	65535	NULL
orders	7	order_delivered_customer_date	text	65535	NULL
orders	8	order_estimated_delivery_date	text	65535	NULL
payments	1	order_id	text	65535	NULL
payments	2	payment_sequential	int	NULL	10
payments	3	payment_type	text	65535	NULL
payments	4	payment_installments	int	NULL	10
payments	5	payment_value	double	NULL	22
products	1	product_id	text	65535	NULL
products	2	product category	text	65535	NULL
products	3	product_name_length	int	NULL	10
products	4	product_description_length	int	NULL	10
products	5	product_photos_qty	int	NULL	10
products	6	product_weight_g	int	NULL	10
products	7	product_length_cm	int	NULL	10
products	8	product_height_cm	int	NULL	10
products	9	product_width_cm	int	NULL	10
sellers	1	seller_id	text	65535	NULL
sellers	2	seller_zip_code_prefix	text	65535	NULL
sellers	3	seller_city	text	65535	NULL
sellers	4	seller_state	text	65535	NULL

Task 1.2. Time period for which the data is given

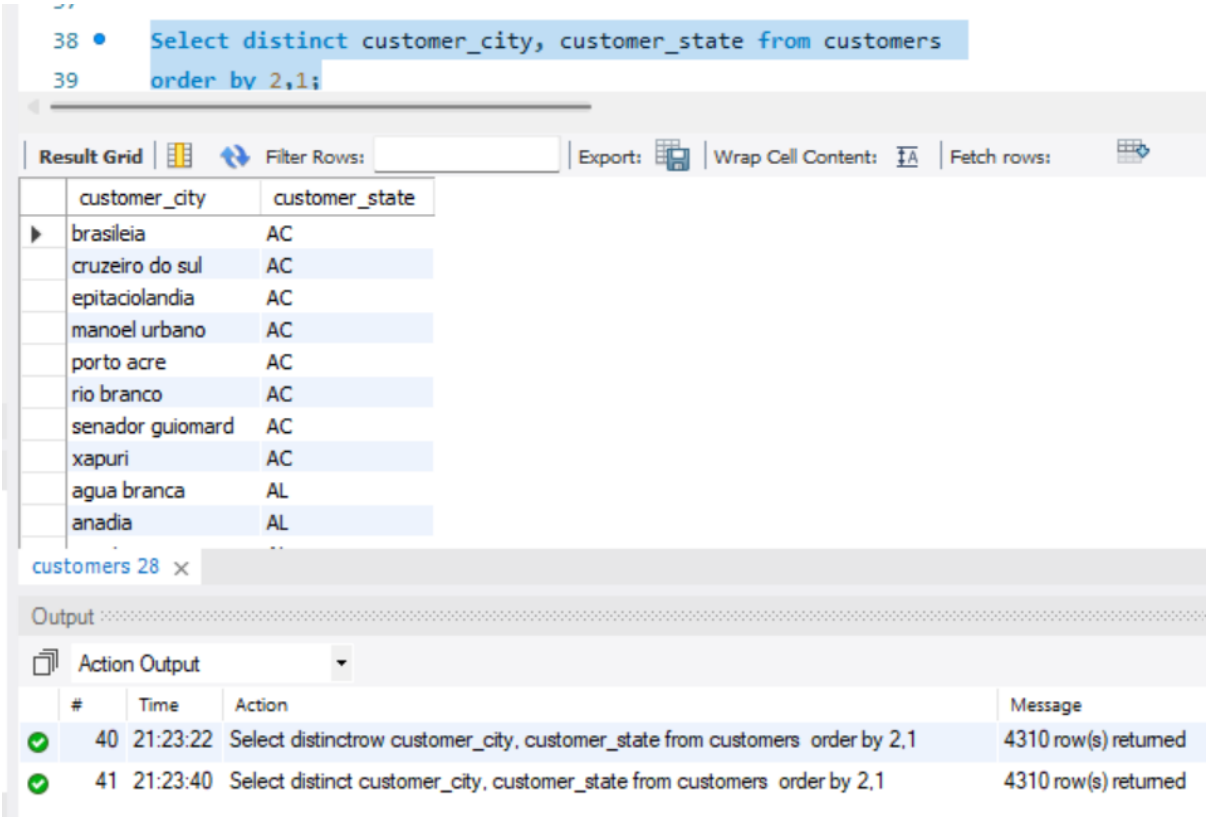
Below SQL query was used on Orders table to fetch the time period of the data provided for analysis

```
select min(order_purchase_timestamp) start_date, max(order_purchase_timestamp) end_date from orders;
```



Task 1.3. Cities and States of customers ordered during the given period

```
Select distinct customer_city, customer_state from customers
order by 2,1;
```



1. In-depth Exploration:

Task: 2.1. Is there a growing trend on e-commerce in Brazil?

How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

As you can see in the below SQL query output the number of orders where the customers are purchasing is increasing MoM and YoY. So we can conclude that there is a growing trend of e-commerce business in Brazil.

```
select year(order_purchase_timestamp)as YEAR,
month(order_purchase_timestamp)as MONTH, count(order_id) as TotalOrders
from orders
group by YEAR,MONTH
order by YEAR , MONTH ;
```

```
73  select year(order_purchase_timestamp)as YEAR,
74  month(order_purchase_timestamp)as MONTH, count(order_id) as TotalOrders
75  from orders
76  group by YEAR,MONTH
77  order by YEAR , MONTH  ;
```

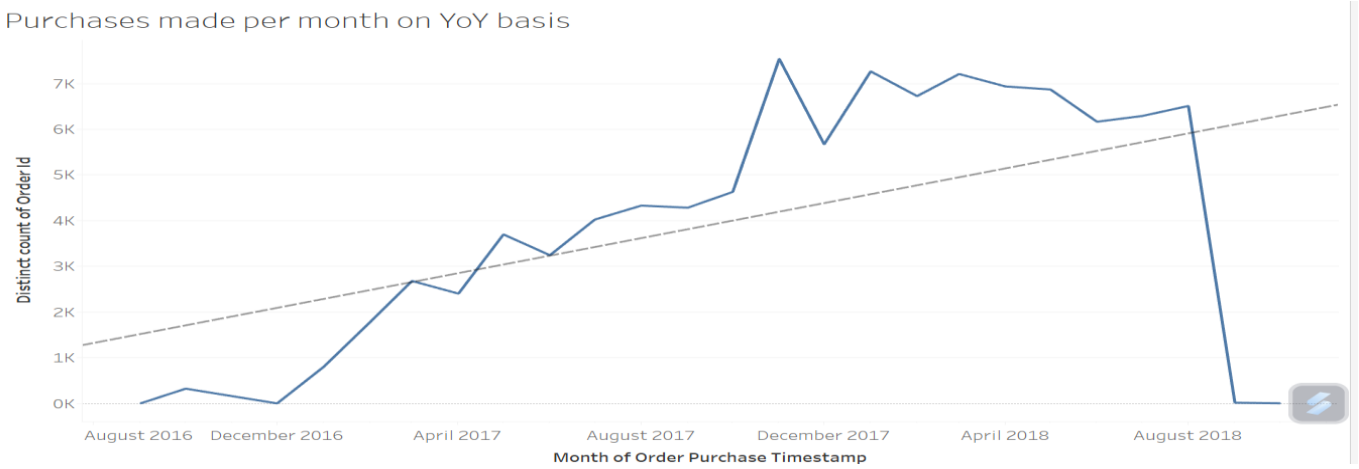
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

YEAR	MONTH	TotalOrders
2016	9	4
2016	10	324
2016	12	1
2017	1	800
2017	2	1780
2017	3	2682
2017	4	2404
2017	5	3700
2017	6	3245
2017	7	4026
2017	8	4331
2017	9	4285
2017	10	4631
2017	11	7544
2017	12	5673
2018	1	7269

Result 118

Output

You can see the growing trend from the below tableau screenshot as well.



```
73
74  select year(order_purchase_timestamp)as YEAR,
75  month(order_purchase_timestamp)as MONTH, count(order_id) as TotalOrders
76  from orders
77  group by YEAR,MONTH
78  order by TotalOrders desc;
79
```

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

YEAR	MONTH	TotalOrders
2017	11	7544
2018	1	7269
2018	3	7211
2018	4	6939
2018	5	6873
2018	2	6728
2018	8	6512

Result 126

Output

Highest orders were in the month of Nov 2017 followed by Jan 2018 , march 2018 and so on as you can see above.

But with my observation and analysis I could see that there was a steady growth in the trend in the provided. And I saw that in the months of October and November the customer purchase is increasing and its falling for some extent in the month of December, again a slow pick up starting from January. *Means too many customers login into the ecommerce before the Christmas period.*

Task: 2.2. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Most of the Brazilian customers tend to make the purchases at afternoon time followed by night then Morning and at last during the dawn.

Below is the snapshot you can see from the SQL query written.

```
80  with TimeInterval as
81  (
82  select order_id, CASE
83  WHEN hour(order_purchase_timestamp) >=4  AND hour(order_purchase_timestamp) <=7 then 'Dawn'
84  WHEN hour(order_purchase_timestamp) >7  AND hour(order_purchase_timestamp) <=12 then 'Morning'
85  WHEN hour(order_purchase_timestamp) >12  AND hour(order_purchase_timestamp) <=18 then 'Afternoon'
86  ELSE 'Night'
87  END as timeOfDay
88  from orders
89  )
90  select timeOfDay, count(order_id) TotalOrders
91  from TimeInterval
92  group by timeOfDay
```

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

timeOfDay	TotalOrders
Afternoon	38135
Night	32677
Morning	26502
Dawn	2127

```
with TimeInterval as

(
select order_id, CASE
WHEN hour(order_purchase_timestamp) >=4  AND hour(order_purchase_timestamp) <=7 then 'Dawn'
WHEN hour(order_purchase_timestamp) >7  AND hour(order_purchase_timestamp) <=12 then
'Morning'
WHEN hour(order_purchase_timestamp) >12  AND hour(order_purchase_timestamp) <=18 then
'Afternoon'
ELSE 'Night'
END as timeOfDay
from orders
)
select timeOfDay, count(order_id) TotalOrders
from TimeInterval
group by timeOfDay
order by TotalOrders desc;
```

2. Evolution of E-commerce orders in the Brazil region:

Task: 3.1. 1. Get month on month orders by states

The month-on-month total orders for each respective states of Brazil have been derived by the below SQL query:

```
71  select c.customer_state
72  ,month(o.order_purchase_timestamp)as MONTH
73  , count(o.order_id) as TotalOrders
74  from orders o
75  join customers c using(customer_id)
76  group by c.customer_state, month
77  order by c.customer_state, month
```

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

customer_state	MONTH	TotalOrders
AC	1	8
AC	2	6
AC	3	4
AC	4	9
AC	5	10
AC	6	7
AC	7	9
AC	8	7
AC	9	5
AC	10	6

Result 139

Output

Action Output

#	Time	Action	Message
258	20:29:49	select c.customer_state ,month(o.order_purchase_timestamp)as MONTH , count(o....	Error Code: 1064. You
259	20:30:10	select c.customer_state ,month(o.order_purchase_timestamp)as MONTH , count(o....	322 row(s) returned

Task: 3.2. Distribution of customers across the states in Brazil

The distribution of total customer count per Brazil state is fetched by the below SQL query-

```
79  select count(customer_id) as TotalCusts, customer_state
80  from customers
81  group by customer_state
82  order by TotalCusts desc
83
```

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

TotalCusts	customer_state
41746	SP
12852	RJ
11635	MG
5466	RS
5045	PR
3637	SC
3380	BA
2140	DF
2033	ES

Result 145

Output

Action Output

#	Time	Action	Message
267	20:35:55	select count(customer_id) as TotalCusts, customer_state from customers group by c...	27 row(s) returned
268	20:36:24	select count(customer_id) as TotalCusts, customer_state from customers group by c...	27 row(s) returned

4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others

Task: 4.1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use "payment_value" column in payments table

Below SQL query was applied to calculate the % increase on payments received for MoM between Jan 2017 till Aug 2018.

```
-- SQL query to find the %increase in revenue genarated with moM from 2017 Jan –2018 Jul
-- this cte function is extracting the year, month and calculating the total payment recieved per month between Jan 2017 to Aug 2018 using where clause and joins
with OrdeAmtBtwMnYr as
(
select round(sum(p.payment_value),2) as CurMonTotAmt, YEAR(o.order_purchase_timestamp) as YEAR,
MONTH(o.order_purchase_timestamp) as Month
from payments p
join orders o using(order_id)
where YEAR(o.order_purchase_timestamp) between '2017' and '2018'
AND MONTH(o.order_purchase_timestamp) between '1' AND '8'
group by Month, YEAR
```

```
order by Month, YEAR
),
-- this cte gets the previous month total payment recieved using lag function
PrevMonCalc as
(
select Month, YEAR, CurMonTotAmt,
lag(CurMonTotAmt,1) over(order by YEAR,Month) as PrevMonTotAmnt
from OrdeAmtBtwMnYr
)
-- the below query brings the output of MoM % in total amount using formula
select *,
concat(round((((CurMonTotAmt-PrevMonTotAmnt)/PrevMonTotAmnt)*100),2), '%') PercentIncrease
from PrevMonCalc;
```

Month	YEAR	CurMonTotAmt	PrevMonTotAmnt	PercentIncrease
1	2017	138488.04	NULL	NULL
2	2017	291908.01	138488.04	110.78%
3	2017	449863.6	291908.01	54.11%
4	2017	417788.03	449863.6	-7.13%
5	2017	592918.82	417788.03	41.92%
6	2017	511276.38	592918.82	-13.77%
7	2017	592382.92	511276.38	15.86%
8	2017	674396.32	592382.92	13.84%
1	2018	1115004.18	674396.32	65.33%
2	2018	992463.34	1115004.18	-10.99%
3	2018	1159652.12	992463.34	16.85%
4	2018	1160785.48	1159652.12	0.1%

Result 202 ×

Output

Action Output

#	Time	Action	Message
366	12:00:05	with OrdeAmtBtwMnYr as -- this cte function is extracting the year, month and calcul...	16 row(s) returned

Task: 4.2. Mean & Sum of price and freight value by customer state

Calculated the sum, average of the prices from order items table, and also calculated the freight price sum and average per state respectively.

```
-- mean and sum of price and frieght value by customer state
select distinct c.customer_state,
round(sum(oi.price) over(partition by c.customer_state),2) as SumOfPrice,
round(avg(oi.price) over(partition by c.customer_state),2) as AvgPrice,
round(sum(oi.freight_value) over(partition by c.customer_state),2) as SumOfFreight,
round(avg(oi.freight_value) over(partition by c.customer_state),2) as AvgFreight
from order_items oi
join orders o on o.order_id = oi.order_id
join customers c on c.customer_id = o.customer_id;
```

Result-

```
122    round(sum(oi.price) over(partition by c.customer_state),2) as SumOfPrice,
123    round(avg(oi.price) over(partition by c.customer_state),2) as AvgPrice,
124    round(sum(oi.freight_value) over(partition by c.customer_state),2) as SumOfFreight,
125    round(avg(oi.freight_value) over(partition by c.customer_state),2) as AvgFreight
126    from order_items oi
```

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

customer_state	SumOfPrice	AvgPrice	SumOfFreight	AvgFreight
AC	15982.95	173.73	3686.75	40.07
AL	80314.81	180.89	15914.59	35.84
AM	22356.84	135.5	5478.89	33.21
AP	13474.3	164.32	2788.5	34.01
BA	511349.99	134.6	100156.68	26.36
CE	227254.71	153.76	48351.59	32.71
DF	302603.94	125.77	50625.5	21.04
ES	275037.31	121.91	49764.6	22.06
GO	294591.95	126.27	53114.98	22.77
MA	119648.22	145.2	31523.77	38.26
MG	1585308.03	120.75	270853.46	20.63

Result 164

Output

Action Output

#	Time	Action	Message
295	21:47:31	select distinct c.customer_state, round(sum(oi.price) over(partition by c.customer_st...	27 row(s) returned

Actionable insights: The observation is that when the average freight price is low then the sum price of the items is also more. We can conclude that the market is big in the state of SP since number of products are more and average freight process are less.

With a similar observation on RR state where the average freight is the highest so the sum price of the order items is also less, meaning the market capture in the state of RR is the least.

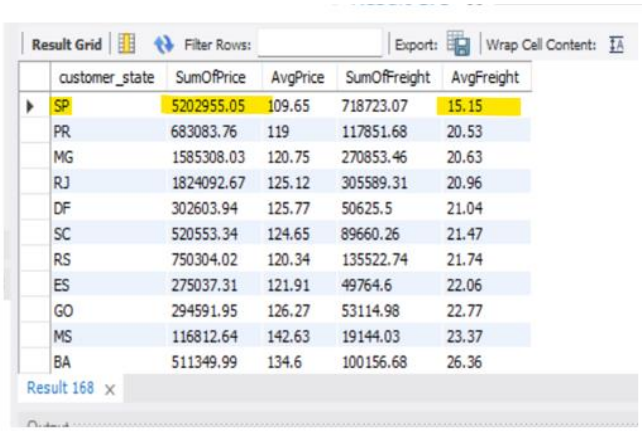
So, if you relate this with the data from task 3.1 where highest customers are in the biggest market share state of SP and lowest number of customers in the smallest market share.

TotalCusts	customer_state
41746	SP
12852	RJ
11635	MG
5466	RS
5045	PR

Result 173

TotalCusts	customer_state
46	RR
68	AP
81	AC
148	AM
253	RO

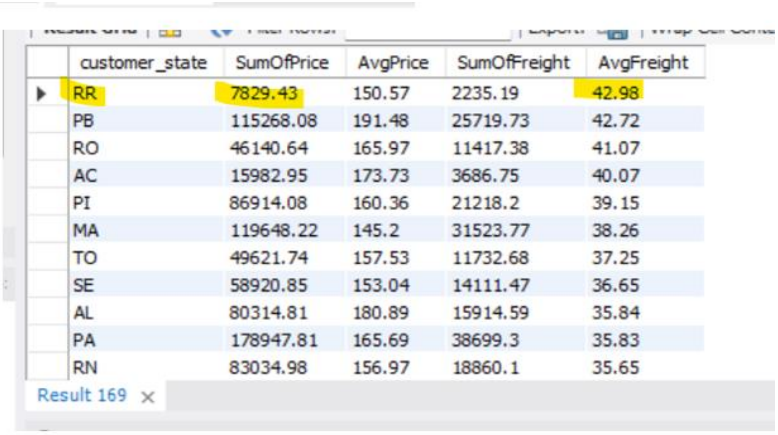
Result 174



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

customer_state	SumOfPrice	AvgPrice	SumOfFreight	AvgFreight
SP	5202955.05	109.65	718723.07	15.15
PR	683083.76	119	117851.68	20.53
MG	1585308.03	120.75	270853.46	20.63
RJ	1824092.67	125.12	305589.31	20.96
DF	302603.94	125.77	50625.5	21.04
SC	520553.34	124.65	89660.26	21.47
RS	750304.02	120.34	135522.74	21.74
ES	275037.31	121.91	49764.6	22.06
GO	294591.95	126.27	53114.98	22.77
MS	116812.64	142.63	19144.03	23.37
BA	511349.99	134.6	100156.68	26.36

Result 168 ×



customer_state	SumOfPrice	AvgPrice	SumOfFreight	AvgFreight
RR	7829.43	150.57	2235.19	42.98
PB	115268.08	191.48	25719.73	42.72
RO	46140.64	165.97	11417.38	41.07
AC	15982.95	173.73	3686.75	40.07
PI	86914.08	160.36	21218.2	39.15
MA	119648.22	145.2	31523.77	38.26
TO	49621.74	157.53	11732.68	37.25
SE	58920.85	153.04	14111.47	36.65
AL	80314.81	180.89	15914.59	35.84
PA	178947.81	165.69	38699.3	35.83
RN	83034.98	156.97	18860.1	35.65

Result 169 ×

Recommendation point: Noting at the above analysis. If Target is planning to expand business in RR then it has to either look for better freight pricing and include more products and product lines to capture the market. OR check what is stopping customers to onboard in the ecommerce platform. Or if it's not in the way to add those then better not to plan anything on expansion on RR state.

5. Analysis on sales, freight and delivery time

Task: 5.1. Calculate days between purchasing, delivering and estimated delivery

Here the approach I took to calculate the days between purchasing, delivering and estimated delivery is- Took datediff function to calculate . How much time was taken between purchasing a product and delivered to the carried, then how much time it took by the carrier to deliver it to the customer, then the total time customer waited for his delivery from the date of purchase, then finally the difference between the estimated delivery time by the ecommerce website and actual delivery time to the customer.

Below is the SQL query to fetch the result and its result screenshot-

```
-- Calculate days between purchasing, delivering and estimated delivery

select
concat(datediff(order_delivered_carrier_date, order_purchase_timestamp ), ' days') as
Days_between_purchase_to_carrier,
concat(datediff(order_delivered_customer_date,order_delivered_carrier_date), ' days') as
Days_carrier_took_to_deliver,
concat(datediff(order_delivered_customer_date, order_purchase_timestamp ), ' days') as
Total_time_for_delivery_since_purchase,
concat(datediff(order_estimated_delivery_date, order_delivered_customer_date), ' days') as
Days_between_estimated_and_actual_delivery
from orders;
```

```
170  -- Calculate days between purchasing, delivering and estimated delivery
171  select
172  concat(datediff(order_delivered_carrier_date, order_purchase_timestamp ), ' days') as Days_between_purchas
173  concat(datediff(order_delivered_customer_date,order_delivered_carrier_date), ' days') as Days_carrier_took
174  concat(datediff(order_delivered_customer_date, order_purchase_timestamp ), ' days') as Total_time_for_deli
```

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

Days_between_purchase_to_carrier	Days_carrier_took_to_deliver	Total_time_for_delivery_since_purchase	Days_between_estimated_and_actual_deliv
2 days	6 days	8 days	8 days
2 days	12 days	14 days	6 days
0 days	9 days	9 days	18 days
4 days	10 days	14 days	13 days
1 days	2 days	3 days	10 days
2 days	15 days	17 days	6 days
NULL	NULL	NULL	NULL

Task 5.2. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

o time_to_delivery = order_purchase_timestamp-order_delivered_customer_date

o diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

```
-- Find time_to_delivery & diff_estimated_delivery
select
concat(datediff(order_delivered_customer_date, order_purchase_timestamp ), ' days') as time_to_delivery ,
concat(datediff(order_estimated_delivery_date, order_delivered_customer_date), ' days') as
diff_estimated_delivery
from orders
```

```
179  -- Find time_to_delivery & diff_estimated_delivery
180  select
181  concat(datediff(order_delivered_customer_date, order_purchase_timestamp ), ' days') a
182  concat(datediff(order_estimated_delivery_date, order_delivered_customer_date), ' days
183  from orders
184
```

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

time_to_delivery	diff_estimated_delivery
8 days	8 days
14 days	6 days
9 days	18 days
14 days	13 days
3 days	10 days
17 days	6 days
NULL	NULL
10 days	12 days

Result 235 ×

Output

Task: 5.3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

-– Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery
select
c.customer_state as State,
round(avg(oi.freight_value),2) as Avg_Freight_rate,
concat(round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),2), ' days') as Avg_time_to_delivery,
concat(round(avg(datediff(order_estimated_delivery_date, order_delivered_customer_date)),2), ' days') as Avg_diff_estimated_delivery
from orders o
join order_items oi using(order_id)
join customers c on o.customer_id = c.customer_id
group by c.customer_state;

```
197  -- CORRECT: Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery
198  select
199  c.customer_state as State,
200  round(avg(oi.freight_value),2) as Avg_Freight_rate,
201  concat(round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),2), ' days') as Avg_ti
202  concat(round(avg(datediff(order_estimated_delivery_date, order_delivered_customer_date)),2), ' days') as A
203  from orders o
204  join order_items oi using(order_id)
205    join customers c on o.customer_id = c.customer_id
```

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

State	Avg_Freight_rate	Avg_time_to_delivery	Avg_diff_estimated_delivery
GO	22.77	15.34 days	12.29 days
MG	20.63	11.92 days	13.34 days
SP	15.15	8.66 days	11.21 days
MS	23.37	15.46 days	11.23 days
RJ	20.96	15.07 days	12.01 days
MA	38.26	21.59 days	9.91 days

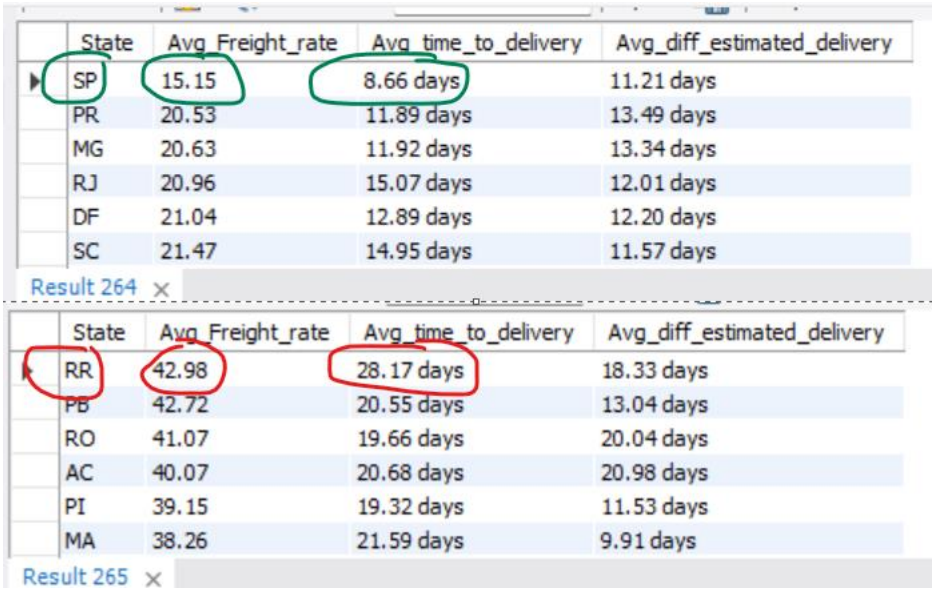
Result 256 × Read Onl

Output

Action Output

#	Time	Action	Message
448	16:56:22	select c.customer_state as State, round(avg(oi.freight_value),2) as Avg_Freight_rat...	27 row(s) returned

Insights: In the state SP the average freight rate is the lowest and the average time to delivery is also the lowest. Meaning there are multiple carriers with attractive pricing hence delivery is also much ahead of estimated days.
On the counter side in the state of RR even when the customers are paying the highest average freight prices their delivery time is much more than the customers in the state of SP.
Please find below-



State	Avg_Freight_rate	Avg_time_to_delivery	Avg_diff_estimated_delivery
SP	15.15	8.66 days	11.21 days
PR	20.53	11.89 days	13.49 days
MG	20.63	11.92 days	13.34 days
RJ	20.96	15.07 days	12.01 days
DF	21.04	12.89 days	12.20 days
SC	21.47	14.95 days	11.57 days

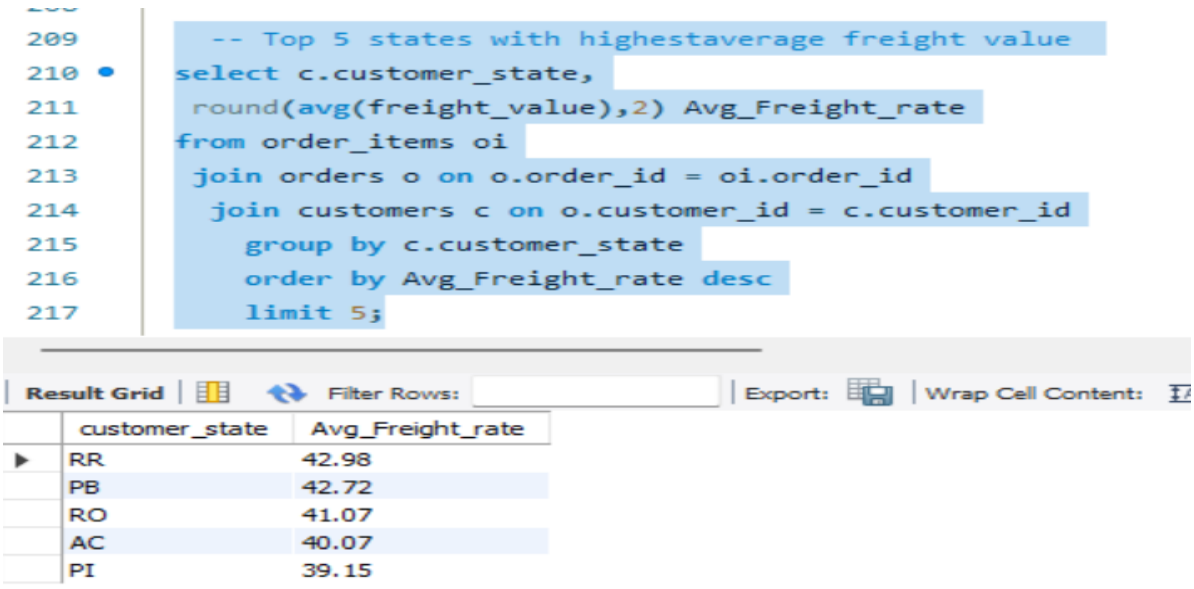
Result 264

State	Avg_Freight_rate	Avg_time_to_delivery	Avg_diff_estimated_delivery
RR	42.98	28.17 days	18.33 days
PB	42.72	20.55 days	13.04 days
RO	41.07	19.66 days	20.04 days
AC	40.07	20.68 days	20.98 days
PI	39.15	19.32 days	11.53 days
MA	38.26	21.59 days	9.91 days

Result 265

Task: 5.5. 5. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

```
-- Top 5 states with highest average freight value
select c.customer_state,
 round(avg(freight_value),2) Avg_Freight_rate
from order_items oi
 join orders o on o.order_id = oi.order_id
  join customers c on o.customer_id = c.customer_id
          group by c.customer_state
          order by Avg_Freight_rate desc
   limit 5;
```

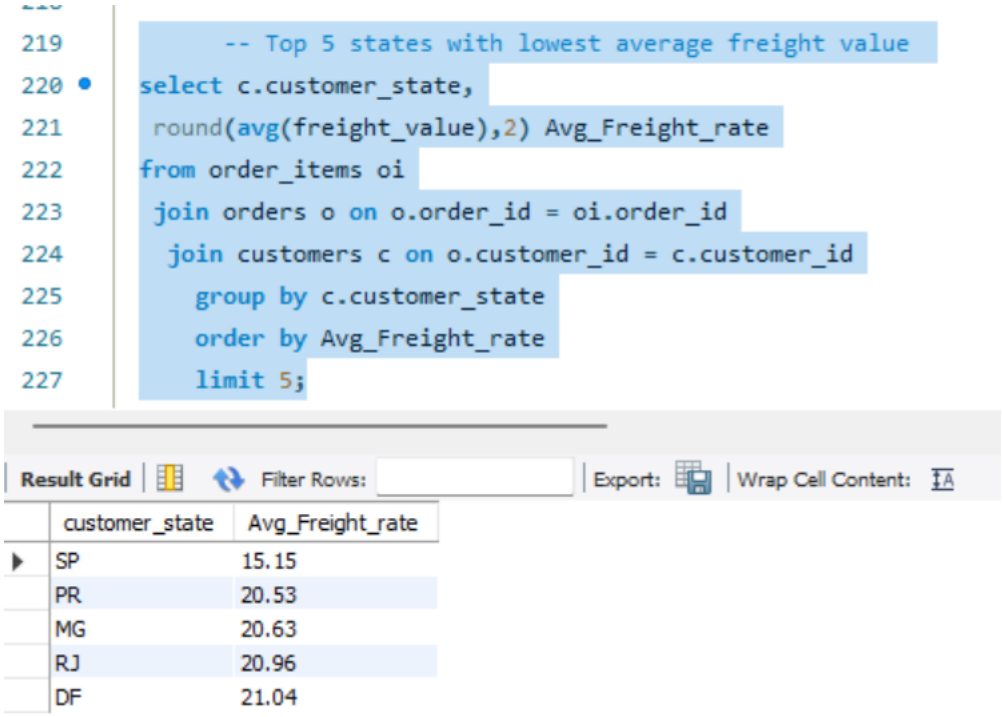


```
209   -- Top 5 states with highestaverage freight value
210 • select c.customer_state,
211    round(avg(freight_value),2) Avg_Freight_rate
212   from order_items oi
213    join orders o on o.order_id = oi.order_id
214     join customers c on o.customer_id = c.customer_id
215       group by c.customer_state
216       order by Avg_Freight_rate desc
217       limit 5;
```

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

customer_state	Avg_Freight_rate
RR	42.98
PB	42.72
RO	41.07
AC	40.07
PI	39.15

```
-- Top 5 states with lowest average freight value
select c.customer_state,
 round(avg(freight_value),2) Avg_Freight_rate
from order_items oi
 join orders o on o.order_id = oi.order_id
  join customers c on o.customer_id = c.customer_id
        group by c.customer_state
        order by Avg_Freight_rate
        limit 5;
```



customer_state	Avg_Freight_rate
SP	15.15
PR	20.53
MG	20.63
RJ	20.96
DF	21.04

Task: 5.6. Top 5 states with highest/lowest average time to delivery

```
-- Top 5 states with lowest average time to delivery
with a as(
select c.customer_state as State,
round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),2) as orderByTime,
concat(round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),2), ' days') as
Avg_time_to_delivery
from orders o
join order_items oi using(order_id)
 join customers c using(customer_id)
  group by c.customer_state
  order by orderByTime
  )
  select State,Avg_time_to_delivery
  from a
 limit 5;
```

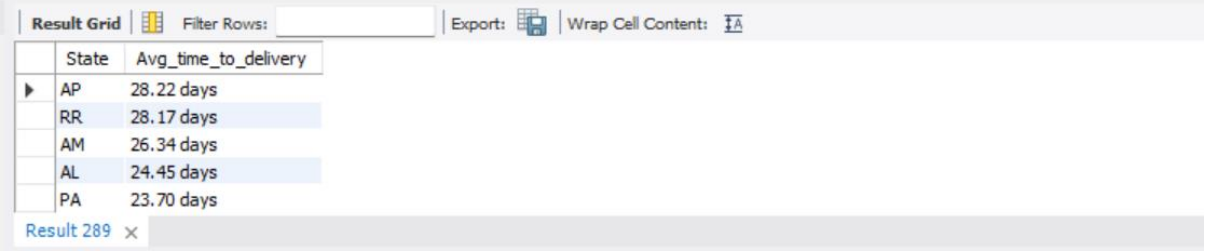
```
241  with a as(
242  select c.customer_state as State,
243  round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),2) as o
244  concat(round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),
245  from orders o
246  join order_items oi using(order_id)
247    join customers c using(customer_id)
248      group by c.customer_state
249      order by orderByTime
250      )
```

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

State	Avg_time_to_delivery
SP	8.66 days
PR	11.89 days
MG	11.92 days
DF	12.89 days
SC	14.95 days

```
-- Top 5 states with highest average time to delivery
with a as(
select c.customer_state as State,
round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),2) as orderByTime,
concat(round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),2), ' days') as
Avg_time_to_delivery
from orders o
join order_items oi using(order_id)
  join customers c using(customer_id)
    group by c.customer_state
    order by orderByTime desc
    )
    select State,Avg_time_to_delivery
    from a
    limit 5;
```

```
with a as(
select c.customer_state as State,
round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),2) as orderByTime,
concat(round(avg(datediff(order_delivered_customer_date, order_purchase_timestamp)),2), ' days'
from orders o
join order_items oi using(order_id)
  join customers c using(customer_id)
    group by c.customer_state
    order by orderByTime desc
    )
    select State,Avg_time_to_delivery
    from a
    limit 5;
```



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

State	Avg_time_to_delivery
AP	28.22 days
RR	28.17 days
AM	26.34 days
AL	24.45 days
PA	23.70 days

Result 289

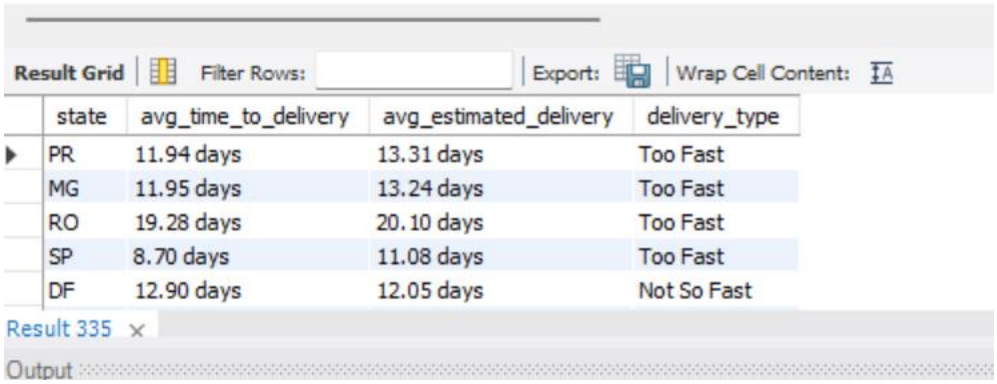
Task: 5.7. Top 5 states where delivery is really fast/ not so fast compared to estimated date

-- Task: 5.7. Top 5 states where delivery is really fast/ not so fast compared to estimated date

States with ‘Too fast’ delivery:

```
with delivery as(
  select customer_id
  ,datediff(order_delivered_customer_date, order_purchase_timestamp ) as time_to_delivery
, datediff(order_estimated_delivery_date, order_delivered_customer_date) as diff_estimated_delivery
  from orders
)
select c.customer_state as state
, concat(round(avg(a.time_to_delivery),2), ' days') as avg_time_to_delivery
, concat(round(avg(diff_estimated_delivery),2), ' days') as avg_estimated_delivery
, case
when avg(a.time_to_delivery) < avg(diff_estimated_delivery) then 'Too Fast'
ELSE 'Not So Fast'
end as delivery_type
from delivery a
join customers c
on a.customer_id = c.customer_id
group by c.customer_state
order by delivery_type desc, avg_time_to_delivery
limit 5;
```

```
327     end as delivery_type
328     from delivery a
329     join customers c
330     on a.customer_id = c.customer_id
331     group by c.customer_state
332     order by delivery_type desc, avg_time_to_delivery
333     limit 5;
334
```



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

state	avg_time_to_delivery	avg_estimated_delivery	delivery_type
PR	11.94 days	13.31 days	Too Fast
MG	11.95 days	13.24 days	Too Fast
RO	19.28 days	20.10 days	Too Fast
SP	8.70 days	11.08 days	Too Fast
DF	12.90 days	12.05 days	Not So Fast

Result 335 ×

Output

States with ‘Not so fast’ delivery:

```
with delivery as(
  select customer_id
  ,datediff(order_delivered_customer_date, order_purchase_timestamp ) as time_to_delivery
, datediff(order_estimated_delivery_date, order_delivered_customer_date) as diff_estimated_delivery
  from orders
)
select c.customer_state as state
, concat(round(avg(a.time_to_delivery),2), ' days') as avg_time_to_delivery
, concat(round(avg(diff_estimated_delivery),2), ' days') as avg_estimated_delivery
, case
when avg(a.time_to_delivery) < avg(diff_estimated_delivery) then 'Too Fast'
ELSE 'Not So Fast'
end as delivery_type
from delivery a
join customers c
on a.customer_id = c.customer_id
group by c.customer_state
order by delivery_type asc, avg_time_to_delivery
limit 5;
```

```
with delivery as(
  select customer_id
  ,datediff(order_delivered_customer_date, order_purchase_timestamp ) as time_to_delivery
, datediff(order_estimated_delivery_date, order_delivered_customer_date) as diff_estimated_delivery
  from orders
)
```

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

state	avg_time_to_delivery	avg_estimated_delivery	delivery_type
DF	12.90 days	12.05 days	Not So Fast
SC	14.91 days	11.51 days	Not So Fast
RJ	15.24 days	11.77 days	Not So Fast
RS	15.25 days	13.91 days	Not So Fast
GO	15.54 days	12.19 days	Not So Fast

Result 336

Output

6. Payment type analysis:

Task: 6.1. Month over Month count of orders for different payment types

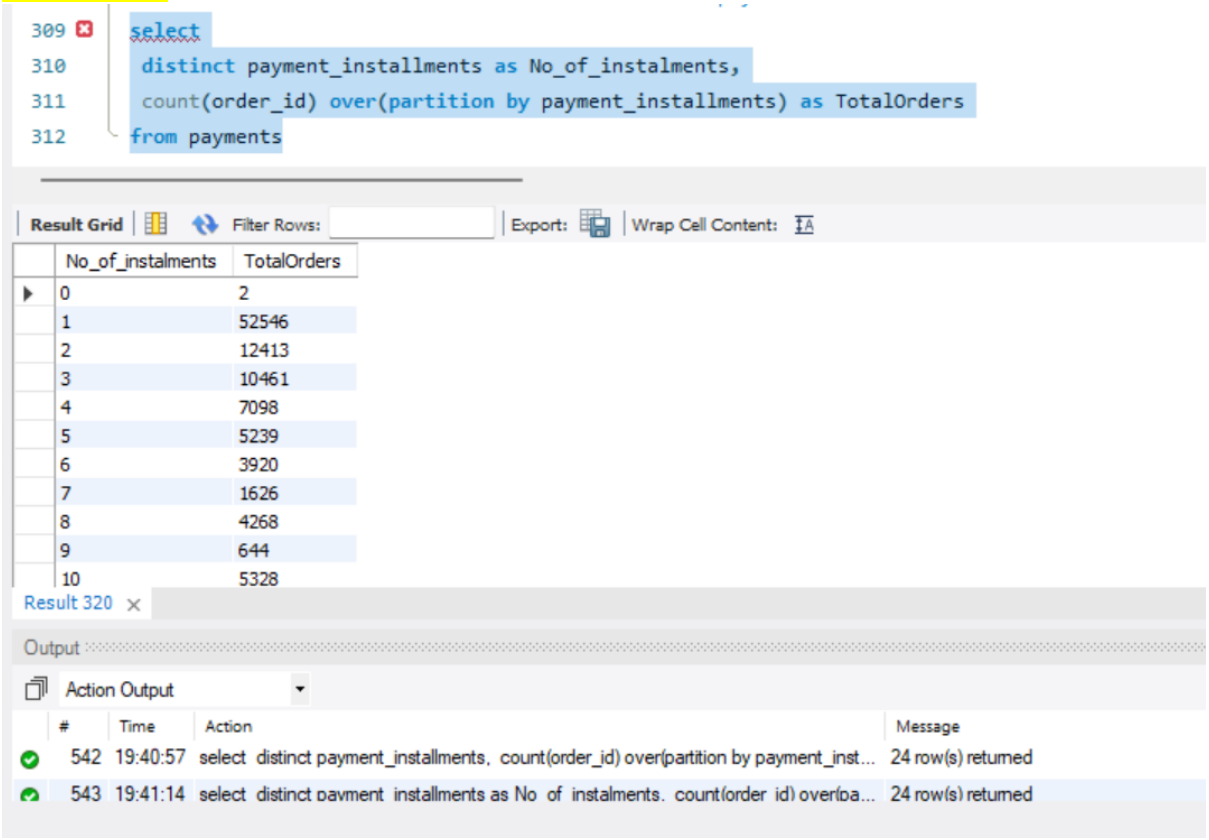
```
select
YEAR(o.order_purchase_timestamp)as YEAR
, month(o.order_purchase_timestamp)as MONTH
, p.payment_type
,count(o.order_id) as TotalOrders
from orders o
join payments p using(order_id)
group by  YEAR, Month, p.payment_type
order by Month, TotalOrders desc;
```

YEAR	MONTH	payment_type	TotalOrders
2018	1	credit_card	5520
2018	1	UPI	1518
2017	1	credit_card	583
2018	1	voucher	416
2017	1	UPI	197
2018	1	debit_card	109
2017	1	voucher	61
2017	1	debit_card	9
2018	2	credit_card	5253
2017	2	credit_card	1356
2018	2	UPI	1325
2017	2	UPI	398
2018	2	voucher	305

Result 351

Task: 6.2. Count of orders based on the no. of payment instalments

```
select
 distinct payment_installments as No_of_instalments,
 count(order_id) over(partition by payment_installments) as TotalOrders
from payments;
```



As we can see there are more orders based paid on credit card and on single payment and less orders on instalments more than 12 months. We can conclude that the customer buying tendency is good by paying at one shot rather than opting for instalments. There can be multiple reasons for this, may be banks are offering more interest rates or there are no attractive offers on EMI more than 12 months like that.