TARGET SQL BUSINESS CASE STUDY

SOLUTION:

I) Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

1. Data type of all columns in the "customers" table.

CODE:

```
SELECT column_name, data_type
from Target_SQL_Project.INFORMATION_SCHEMA.COLUMNS
where table_name = 'customers';
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

INSIGHTS:

- Most columns are of data type “STRING”
- Only Customer Zip code contains the INT data type

2. Get the time range between which the orders were placed.

CODE:

```
SELECT
min(order_purchase_timestamp) as first_order,
max(order_purchase_timestamp) as last_order
from Target_SQL_Project.orders;
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	first_order	last_order
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

INSIGHTS:

- The first order starts by September 2016
- The last order was done on October 2018
- So its ranges for more than 2 years in Brazil

3. Count the Cities & States of customers who ordered during the given period.

CODE:

```
select
count(distinct(customer_city)) as city_count,
count(distinct(customer_state)) as state_count
from Target_SQL_Project.customers;
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON

Row	city_count	state_count
1	4119	27

INSIGHTS:

- There are totally 4119 cities and 27 states of customers are ordered during the given period.

II) In-depth Exploration:

1. Is there a growing trend in the no. of orders placed over the past years?

CODE:

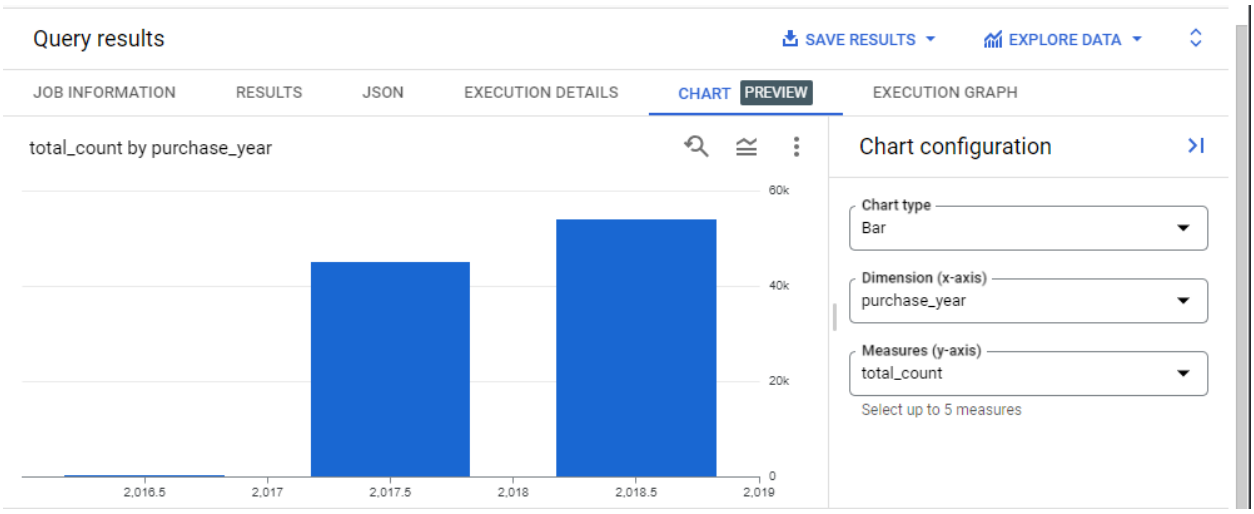
```
select count(*) as total_count,
EXTRACT(YEAR from order_purchase_timestamp) as purchase_year
from Target_SQL_Project.orders
group by EXTRACT(YEAR from order_purchase_timestamp);
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	total_count	purchase_year
1	329	2016
2	45101	2017
3	54011	2018



INSIGHTS:

- There is a growing trend in the number of orders placed in the previous years.
- As we got 329 orders in 2016, 45101 orders in 2017 and the maximum of 54011 orders got in 2018.

2. Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

CODE:

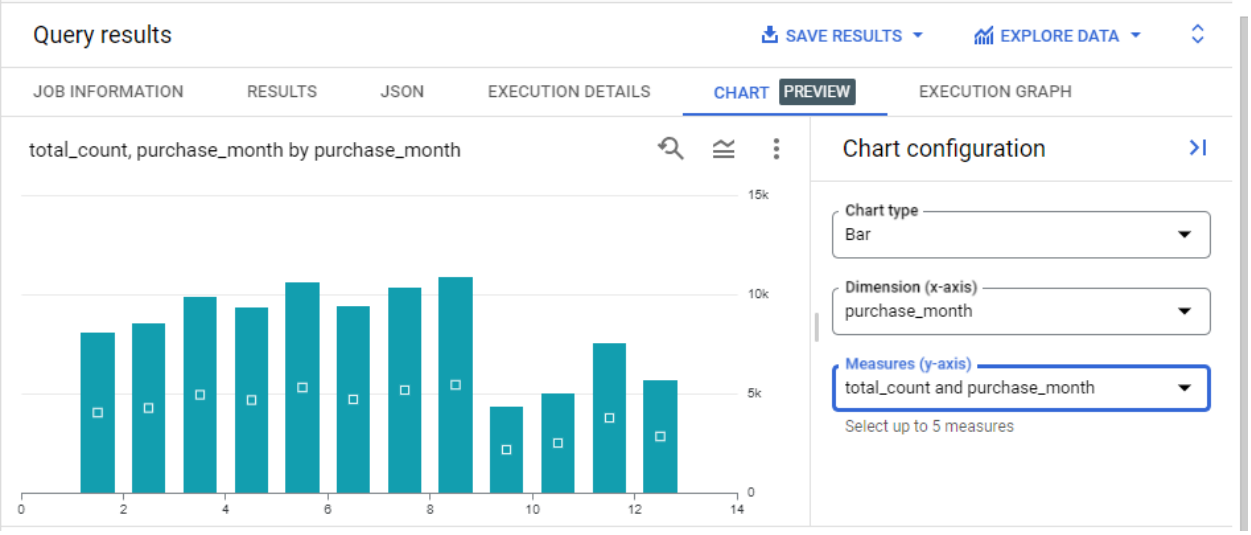
```
SELECT * FROM (
select count(*) as total_count,
EXTRACT(YEAR from order_purchase_timestamp) as purchase_year
from Target_SQL_Project.orders
group by EXTRACT(YEAR from order_purchase_timestamp)) ORDER BY purchase_year;
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	total_count	purchase_month
1	8069	1
2	8508	2
3	9893	3
4	9343	4
5	10573	5
6	9412	6
7	10318	7
8	10843	8
9	4305	9
10	4959	10
11	7544	11
12	5674	12



Query results

JOB INFORMATION | RESULTS | JSON

Row	total_count	purchase_month
1	10843	8
2	10573	5
3	10318	7
4	9893	3
5	9412	6
6	9343	4
7	8508	2
8	8069	1
9	7544	11
10	5674	12
11	4959	10
12	4305	9

INSIGHTS:

- Yes we found some seasonality in the number of orders placed in each month
- We could see that the Months August, May and July has the most number of orders in those years.
- Also, we could see that the months October, September and December has the less number of orders.

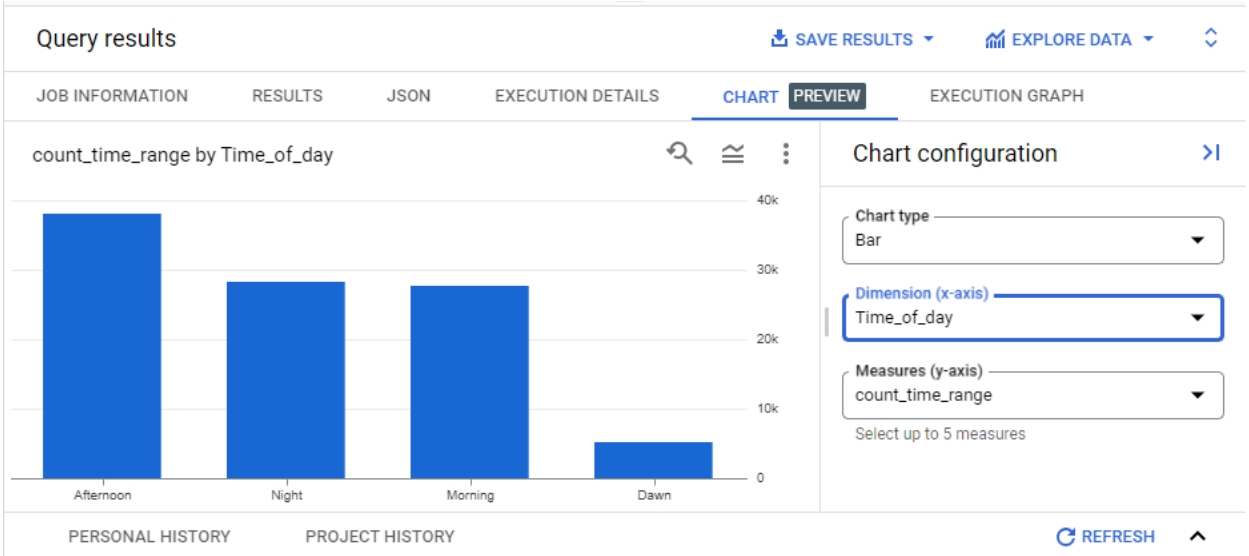
2. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)
   - 0-6 hrs : Dawn
   - 7-12 hrs : Mornings
   - 13-18 hrs : Afternoon
   - 19-23 hrs : Night

CODE:

```
select count (order_id) as count_time_range, Time_of_day
from
(select order_id,order_purchase_timestamp,
case
when (purchase_hour BETWEEN 0 AND 6) then 'Dawn'
when (purchase_hour BETWEEN 7 AND 12) then 'Morning'
when (purchase_hour BETWEEN 13 AND 18) then 'Afternoon'
else 'Night'
end as Time_of_day
from
( select order_id,order_purchase_timestamp,
EXTRACT(HOUR from order_purchase_timestamp) as purchase_hour
```

```
from Target_SQL_Project.orders))
group by Time_of_day
order by count_time_range desc;
```

OUTPUT:



Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	count_time_range	Time_of_day
1	38135	Afternoon
2	28331	Night
3	27733	Morning
4	5242	Dawn

INSIGHTS:

- From the output, we could see that the most number of orders were placed in **Afternoon** (BETWEEN 13 AND 18)
- After that we could see that most orders are in **Night** and **Morning**, also there is no much difference in the number of orders at this time.
- Also it is clear that there is no high number of orders placed in **Dawn** time.

III) Evolution of E-commerce orders in the Brazil region:

1. Get the month on month no. of orders placed in each state.

CODE:

```
SELECT * FROM (
select count(*) as total_count,customer_state,
EXTRACT(MONTH from order_purchase_timestamp) as purchase_month
from
(select ord.order_id,ord.customer_id
,ord.order_purchase_timestamp,cus.customer_state
FROM Target_SQL_Project.orders as ord
LEFT JOIN Target_SQL_Project.customers as cus
ON ord.customer_id = cus.customer_id)
group by EXTRACT(MONTH from order_purchase_timestamp),customer_state ) ORDER BY
total_count desc;
```

OUTPUT:

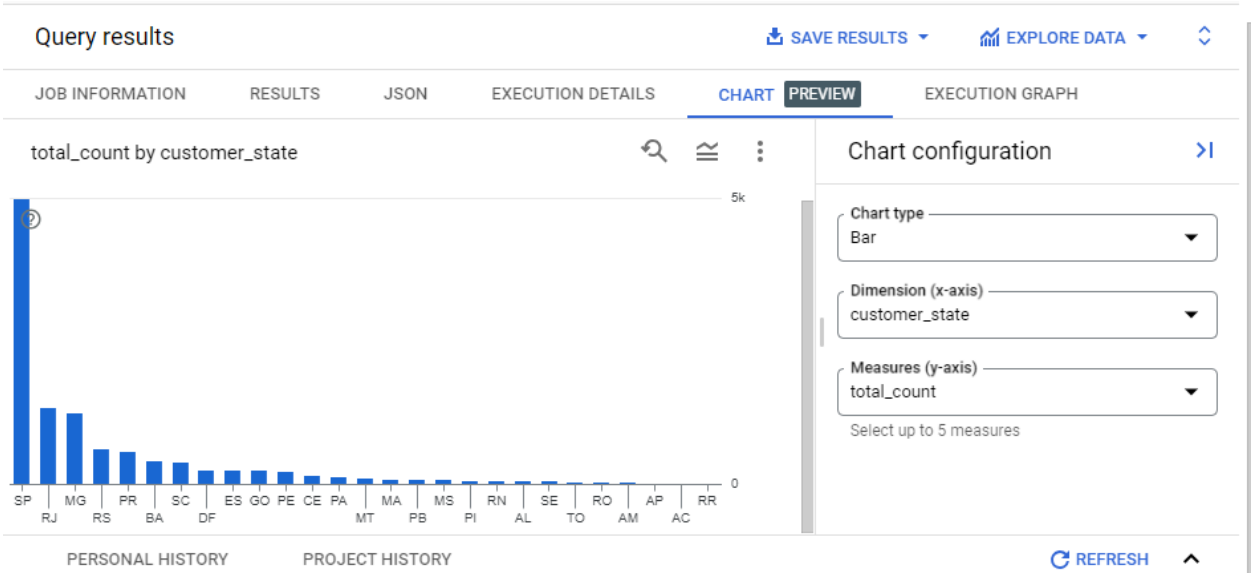
Query results

SAVE RES

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PREVIEW

Row	total_count	customer_state	purchase_month
1	4982	SP	8
2	4632	SP	5
3	4381	SP	7
4	4104	SP	6
5	4047	SP	3
6	3967	SP	4
7	3357	SP	2
8	3351	SP	1
9	3012	SP	11
10	2357	SP	12

Results per page: 50



INSIGHTS:

- In the data we got in Month on month number of orders placed in each state, the customer state **SP** holds most number of orders for all months.
- After that to SP, the customer state **RJ**, **RS** and **MG** holds the most number of orders for maximum number of months.
- Also it is clearly seen that the customer state **AP, AM, RR** has less than 10 orders in more months.

2. How are the customers distributed across all the states?

CODE:

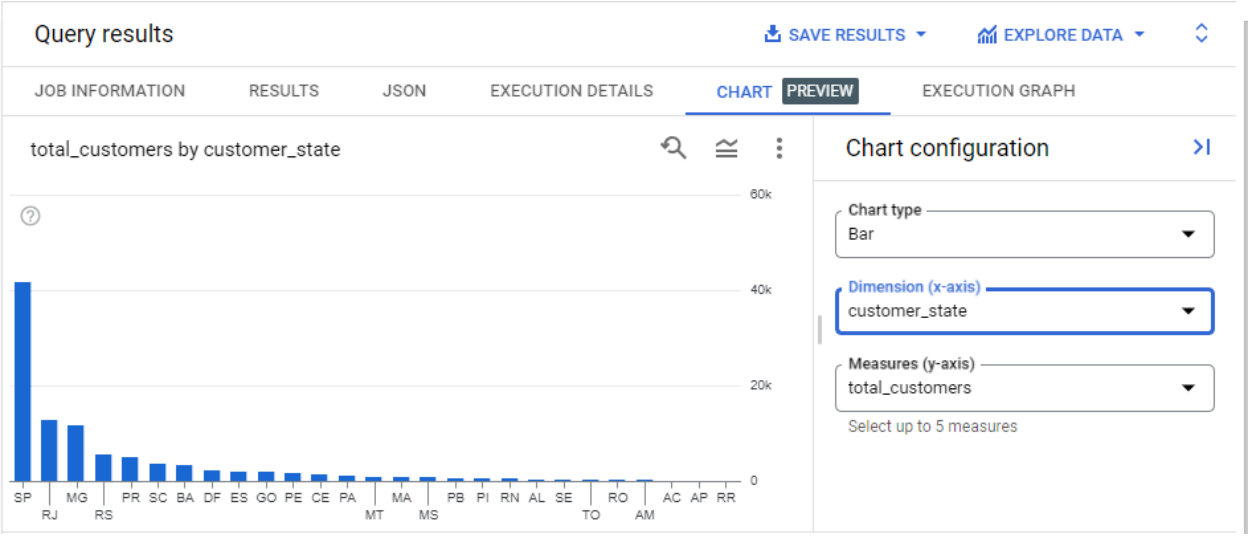
```
select count(*) as total_customers, customer_state
from Target_SQL_Project.customers
group by customer_state
order by total_customers desc;
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	total_customers	customer_state
1	41746	SP
2	12852	RJ
3	11635	MG
4	5466	RS
5	5045	PR
6	3637	SC
7	3380	BA
8	2140	DF
9	2033	ES
10	2020	GO



INSIGHTS:

- It is clear that the most number of customers are part of the SP state, which holds almost 41746 total customers.
- After that the states MG, RJ has the most count of customers.
- Also it is clear that the states RR, AP, AC, AM, RO, TO have the least number of customers has less than 300 customers.

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

1. Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

CODE:

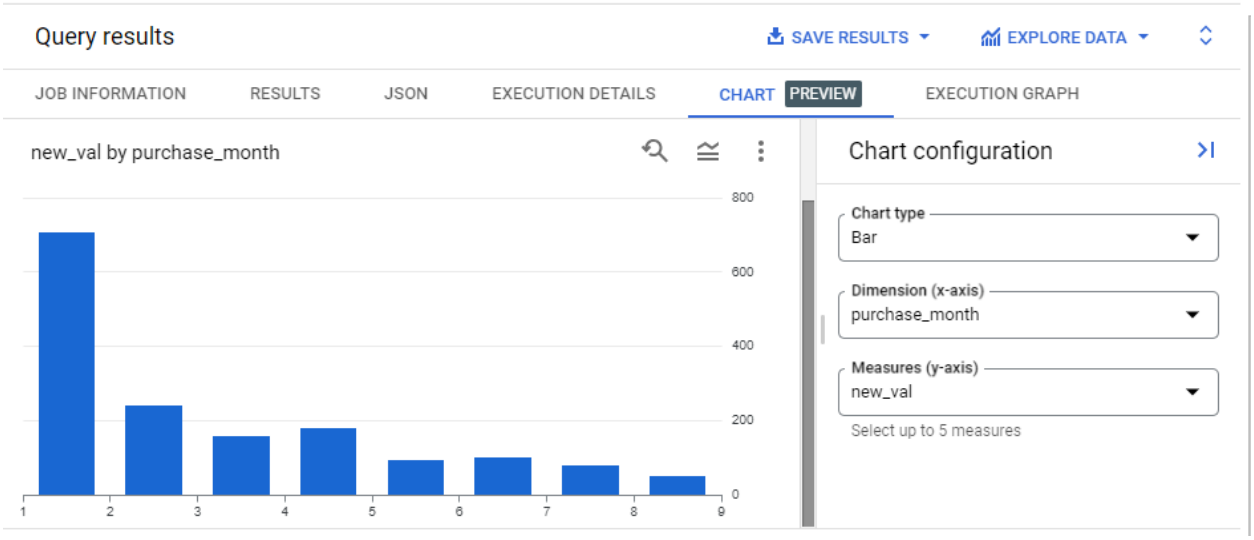
```
select purchase_month, purchase_year,
(val_of_inc / prev_val)*100 as new_val
from
(select  purchase_month, purchase_year, LAG(total_payment_value) OVER (partition by
purchase_month order BY purchase_month,purchase_year)AS prev_val,
total_payment_value - LAG(total_payment_value) OVER (partition by purchase_month order
BY purchase_month,purchase_year) as val_of_inc,
FROM
(select sum(payment_value) as total_payment_value,
 purchase_month, purchase_year from
(select * from
(select order_id, customer_id, payment_value,
EXTRACT(MONTH from order_purchase_timestamp) as purchase_month,
EXTRACT(YEAR from order_purchase_timestamp) as purchase_year from
(select ord.order_id, ord.customer_id, ord.order_purchase_timestamp,pay.payment_value
FROM Target_SQL_Project.orders as ord
LEFT JOIN Target_SQL_Project.payments as pay
ON ord.order_id = pay.order_id))
where purchase_year in (2017,2018) and purchase_month in (1,2,3,4,5,6,7,8))
group by purchase_month, purchase_year
order by purchase_month)
ORDER BY purchase_month, purchase_year);
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAI

Row	purchase_month	purchase_year	new_val
1	1	2017	null
2	1	2018	705.1266954171...
3	2	2017	null
4	2	2018	239.9918145445...
5	3	2017	null
6	3	2018	157.7786066709...
7	4	2017	null
8	4	2018	177.8407701149...
9	5	2017	null
10	5	2018	94.62734375677...



INSIGHTS:

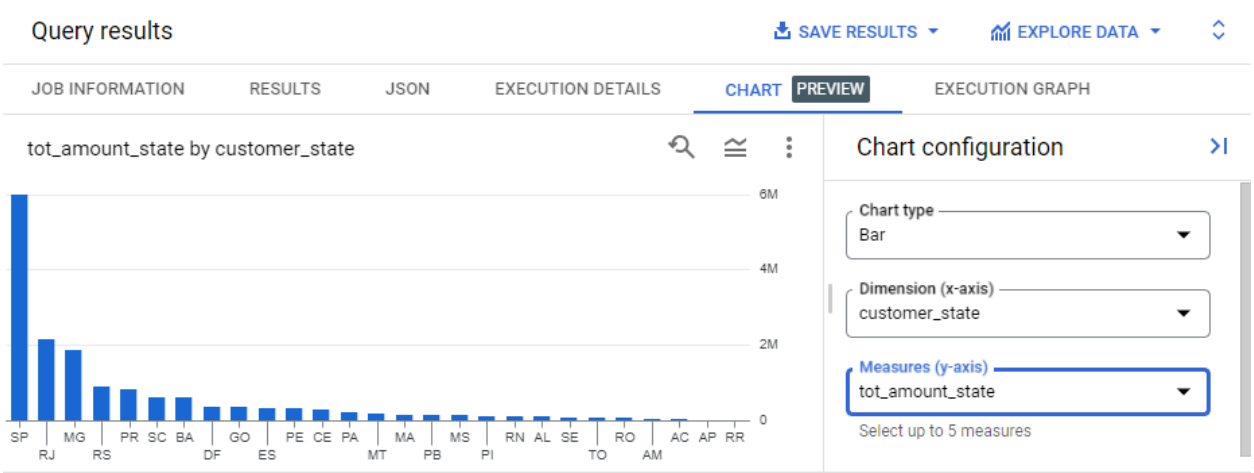
- It is clear that there is an increase in the cost of total orders from 2017 to 218 for every months (From Jan to Aug)
- At maximum the month January has the most % of increase about 705% increase on compared to previous year total order costs.
- Also other months also contributed 250% to 100%
- The lowest % of increase in total cost is for month July and August which is only about 80% and 50%.

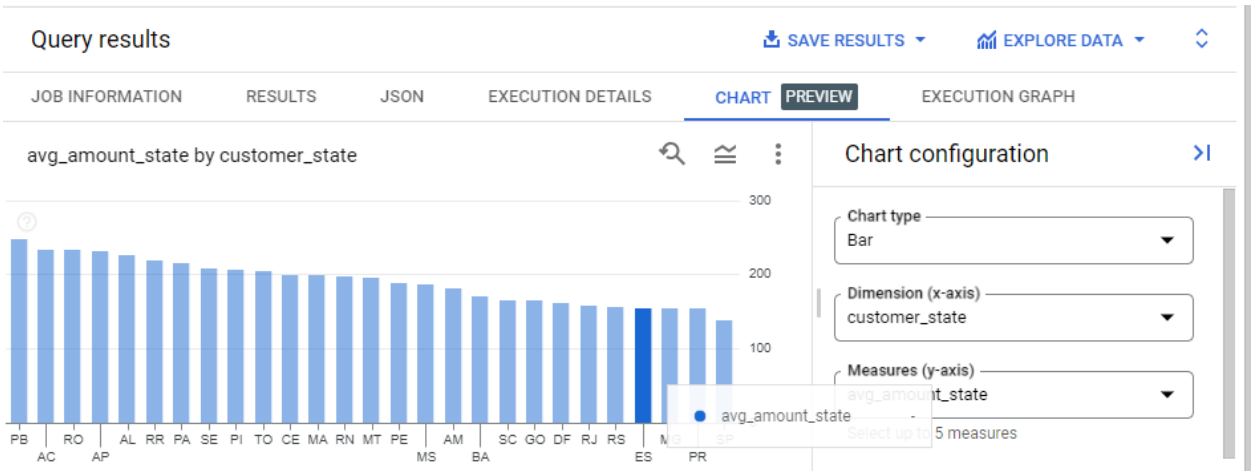
2. Calculate the Total & Average value of order price for each state.

CODE:

```
select
round(sum(payment_value),2) as tot_amount_state,
round(avg(payment_value),2) as avg_amount_state, customer_state from
(select ord.order_id,payment_value,cus.customer_state
FROM Target_SQL_Project.payments as pay
LEFT JOIN Target_SQL_Project.orders as ord
ON pay.order_id=ord.order_id
LEFT JOIN Target_SQL_Project.customers as cus
ON ord.customer_id=cus.customer_id)
group by customer_state
order by tot_amount_state desc;
```

OUTPUT:





Query results

JOB INFORMATION | **RESULTS** | JSON | EXECUTION DETAILS

Row	tot_amount_state	avg_amount_state	customer_state
1	141545.72	248.33	PB
2	19680.62	234.29	AC
3	60866.2	233.2	RO
4	16262.8	232.33	AP
5	96962.06	227.08	AL
6	10064.62	218.8	RR
7	218295.85	215.92	PA
8	75246.25	208.44	SE
9	108523.97	207.11	PI
10	61485.33	204.27	TO

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	tot_amount_state	avg_amount_state	customer_state
1	5998226.96	137.5	SP
2	2144379.69	158.53	RJ
3	1872257.26	154.71	MG
4	890898.54	157.18	RS
5	811156.38	154.15	PR
6	623086.43	165.98	SC
7	616645.82	170.82	BA
8	355141.08	161.13	DF
9	350092.31	165.76	GO
10	325967.55	154.71	ES

INSIGHTS:

- From the data, we could see that the state **SP** has the **highest total value of sales** amount and the next states having the highest are **RJ, MG, RS** respectively.
- Also it is clear that the state **RR, AP, AC, AM** has the **lowest total value of sales** and it is below than 50,000 for the two years.
- For the **average value of sales** for each state, it is clear that the states **PB, AC, RO** has the highest average across Brazil
- Also the **lowest average value of sales** is on the states **SP, PR, MG** ones respectively.
- **Also the one thing is seen that the state SP has the highest total value of sales and the lowest average value of sales.**

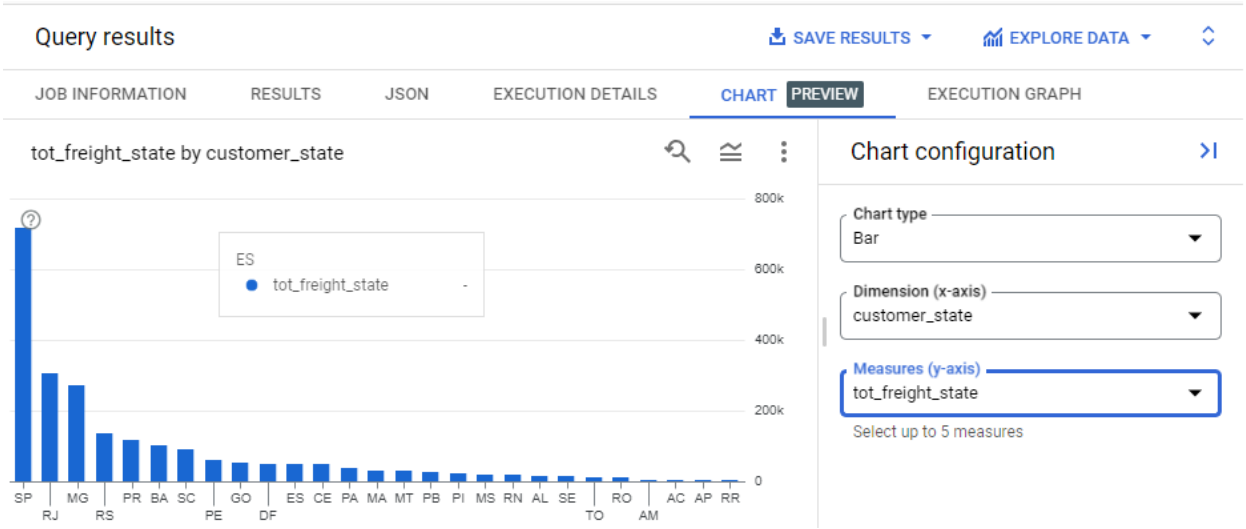
3. Calculate the Total & Average value of order freight for each state.

CODE:

```
select customer_state,
round(sum(freight_value),2) as tot_freight_state,
round(avg(freight_value),2) as avg_freight_state from
(select ord.order_id, items.freight_value, cus.customer_state
FROM Target_SQL_Project.order_items as items
LEFT JOIN Target_SQL_Project.orders as ord
ON items.order_id=ord.order_id
LEFT JOIN Target_SQL_Project.customers as cus
```

```
on ord.customer_id=cus.customer_id)
group by customer_state
order by tot_freight_state desc;
```

OUTPUT:



Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PR

Row	customer_state	tot_freight_state	avg_freight_state
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	MG	270853.46	20.63
4	RS	135522.74	21.74
5	PR	117851.68	20.53
6	BA	100156.68	26.36
7	SC	89660.26	21.47
8	PE	59449.66	32.92
9	GO	53114.98	22.77
10	DF	50625.5	21.04

Results per page:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART

Row	customer_state	tot_freight_state	avg_freight_state
1	RR	2235.19	42.98
2	PB	25719.73	42.72
3	RO	11417.38	41.07
4	AC	3686.75	40.07
5	PI	21218.2	39.15
6	MA	31523.77	38.26
7	TO	11732.68	37.25
8	SE	14111.47	36.65
9	AL	15914.59	35.84
10	PA	38699.3	35.83

Results per page:

INSIGHTS:

- From the data, we could see that the state **SP** has the **highest total value of freight** amount and the next states having the highest are **RJ, MG, RS** respectively.
- Also it is clear that the state **RR, AP, AC, AM** has the **lowest total value of freight** and it is below than 50,000 for the two years.

- For the **average value of freight**for each state, it is clear that the states **RR, PB,, RO** has the highest average across Brazil
- Also the **lowest average value of freight** is on the states **SP, PR, MG** ones respectively.
- **Also the one thing is seen that the state SP has the highest total value of freight and the lowest average value of freight**

V) Analysis based on sales, freight and delivery time.

1. Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
    a. Also, calculate the difference (in days) between the estimated & actual delivery date of an order.
    Do this in a single query.

    You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:
        i. **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
        ii. **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date

CODE:

```
select order_id,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, Day) AS
time_to_deliver,
DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, Day) AS
diff_estimated_delivery
from
(select order_id, order_status, order_purchase_timestamp,
order_delivered_customer_date, order_estimated_delivery_date
from Target_SQL_Project.orders
where order_status = 'delivered');
```

OUTPUT:

Query results

SAVE RES

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PREVIEW

Row	order_id	time_to_deliver	diff_estimated_delivery
1	635c894d068ac37e6e03dc54e...	30	1
2	3b97562c3aee8bdedcb5c2e45...	32	0
3	68f47f50f04c4cb6774570cfde...	29	1
4	276e9ec344d3bf029ff83a161c...	43	-4
5	54e1a3c2b97fb0809da548a59...	40	-4
6	fd04fa4105ee8045f6a0139ca5...	37	-1
7	302bb8109d097a9fc6e9cefc5...	33	-5
8	66057d37308e787052a32828...	38	-6
9	19135c945c554eebfd7576c73...	36	-2
10	4493e45e7ca1084efcd38ddeb...	34	0

Results per page: 50

INSIGHTS:

- From this data, it is clear that the time to delivery of each order is almost 30 to 40 days.
- Also, the maximum of the orders were placed before the estimated date.
- But some orders also had delayed in the delivery of the orders.

2. Find out the top 5 states with the highest & lowest average freight value.

CODE:

```
select customer_state,
round(avg(freight_value),2) as avg_freight_state from
(select ord.order_id, items.freight_value, cus.customer_state
FROM Target_SQL_Project.order_items as items
LEFT JOIN Target_SQL_Project.orders as ord
ON items.order_id=ord.order_id
LEFT JOIN Target_SQL_Project.customers as cus
on ord.customer_id=cus.customer_id)
group by customer_state
order by avg_freight_state desc
LIMIT 5;
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	avg_freight_state
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	avg_freight_state
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

INSIGHTS:

- The top 5 states for the highest of average freight values are **RR, PB, RO, AC, PI.**
- The highest state has the value of 42.98
- The top 5 states for the lowest of average freight values are **SP, PR, MG, RJ, DF.**
- The highest state has the value of 15.15

3. Find out the top 5 states with the highest & lowest average delivery time.

CODE:

```
select
round(avg(time_to_deliver),2) as avg_time_to_deliver,
round(avg(diff_estimated_delivery),2) as avg_diff_estimated_delivery,
  customer_state from
(select order_id, customer_state,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, Day) AS
time_to_deliver,
DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, Day) AS
diff_estimated_delivery
from
(select ord.order_id, ord.order_status, ord.order_purchase_timestamp,
ord.order_delivered_customer_date, ord.order_estimated_delivery_date, cus.customer_state
from Target_SQL_Project.orders as ord
left join Target_SQL_Project.customers as cus
on ord.customer_id=cus.customer_id
where ord.order_status = 'delivered'))
group by customer_state
order by avg_time_to_deliver desc
limit 5;
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART

Row	avg_time_to_deliver	avg_diff_estimated_delivery	customer_state
1	28.98	16.41	RR
2	26.73	18.73	AP
3	25.99	18.61	AM
4	24.04	7.95	AL
5	23.32	13.19	PA

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	avg_time_to_deliver	avg_diff_estimated_c	customer_state
1	8.3	10.13	SP
2	11.53	12.36	PR
3	11.54	12.3	MG
4	12.51	11.12	DF
5	14.48	10.6	SC

INSIGHTS:

- From the modified data it is clear that the top 5 states had the highest average time to deliver the orders are **RR, AP, AM, AL, PA**
- Also the top 5 states which has the lowest time to deliver the orders are **SP, PR, MG, DF, SC**
- Also it is clear that the only state which has the average delivery time within less than 10 days is **SP** which has only **8.3 days**.

3. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.
You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

CODE:

```
select
round(avg(diff_estimated_delivery),2) as avg_diff_estimated_delivery,
  customer_state from
(select order_id, customer_state,
DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, Day) AS
diff_estimated_delivery
from
(select ord.order_id, ord.order_status, ord.order_purchase_timestamp,
ord.order_delivered_customer_date, ord.order_estimated_delivery_date, cus.customer_state
from Target_SQL_Project.orders as ord
left join Target_SQL_Project.customers as cus
on ord.customer_id=cus.customer_id
where ord.order_status = 'delivered'))
group by customer_state
```

```
order by avg_diff_estimated_delivery DESC
limit 5;
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	avg_diff_estimated_c	customer_state
1	7.95	AL
2	8.77	MA
3	9.17	SE
4	9.62	ES
5	9.93	BA

INSIGHTS:

- From the above data it is clear that no states has the average of delivering the orders before the estimated date.
- But when comparing the data, the state AL has the least count of difference in between the days.

VI) Analysis based on sales, freight and delivery time.

1. Find the month on month no. of orders placed using different payment types.

CODE:

```
select count(order_id) as count_of_orders, payment_type,
EXTRACT(MONTH from order_purchase_timestamp) as month from
(select ord.order_id,ord.order_purchase_timestamp, pay.payment_type
from Target_SQL_Project.payments as pay
left join Target_SQL_Project.orders as ord
on pay.order_id=ord.order_id)
group by month,payment_type
order by month, payment_type;
```

OUTPUT:

Query results

SAVE RESULTS

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PREVIEW

Row	count_of_orders	payment_type	month
1	1715	UPI	1
2	6103	credit_card	1
3	118	debit_card	1
4	477	voucher	1
5	1723	UPI	2
6	6609	credit_card	2
7	82	debit_card	2
8	424	voucher	2
9	1942	UPI	3
10	7707	credit_card	3

Results per page: 50 1

INSIGHTS:

- From the retrieved data, it is clear that the maximum of the payments done only by the credit card.
- After to that the payment type chosen is UPI.
- The debit card and the voucher are less in the count on comparing to other types of payments.

2. Find the no. of orders placed on the basis of the payment installments that have been paid.

CODE:

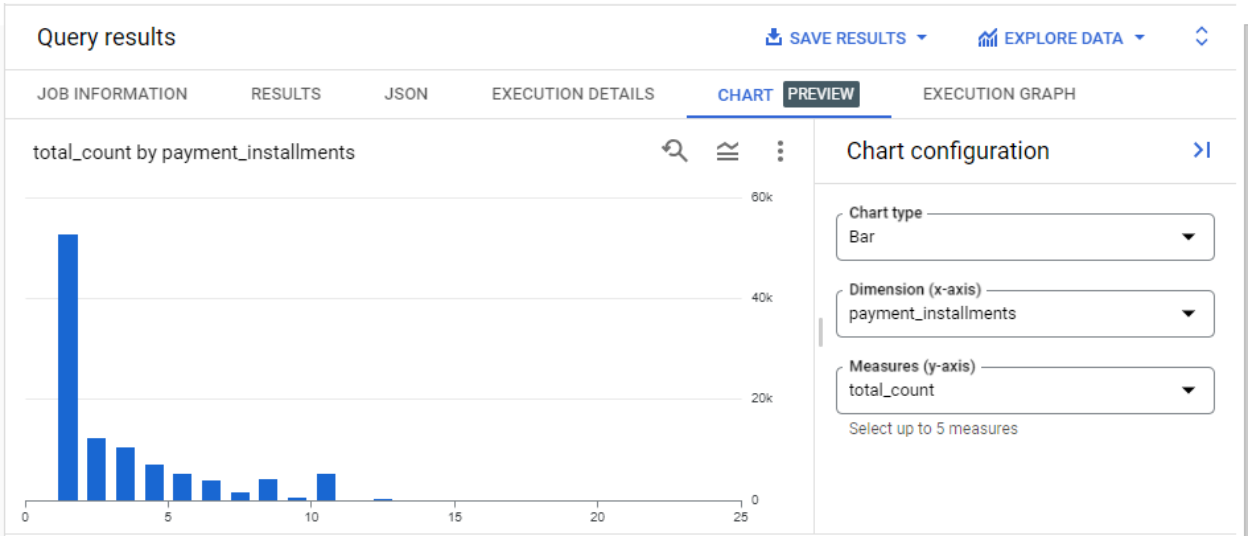
```
select count(order_id) as total_count,payment_installments
from Target_SQL_Project.payments
where payment_installments <> 0
group by payment_installments;
```

OUTPUT:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	total_count	payment_installment
1	52546	1
2	12413	2
3	10461	3
4	7098	4
5	5239	5
6	3920	6
7	1626	7
8	4268	8
9	644	9
10	5328	10



INSIGHTS:

- From the data, it is clear that the maximum number of orders paid by the installments are by single installment only which has more than 50,000 orders.
- Also it is clear that maximum of orders are placed in between installments of 1 to 12, maximum od orders are not taking above that ones.