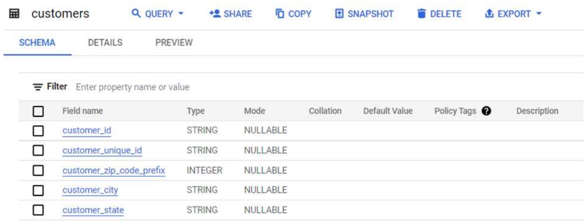
Problem Statement

Explore Retail Giant’s dataset to find insights about the business.

1) Initial exploration of the dataset like checking the characteristics of the data

A. The column and data type in each table were checked.

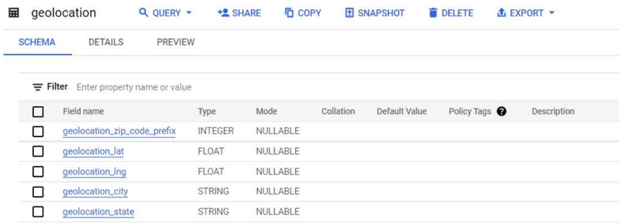
a. Customer Table



□	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
□	customer_id	STRING	NULLABLE				
□	customer_unique_id	STRING	NULLABLE				
□	customer_zip_code_prefix	INTEGER	NULLABLE				
□	customer_city	STRING	NULLABLE				
□	customer_state	STRING	NULLABLE				

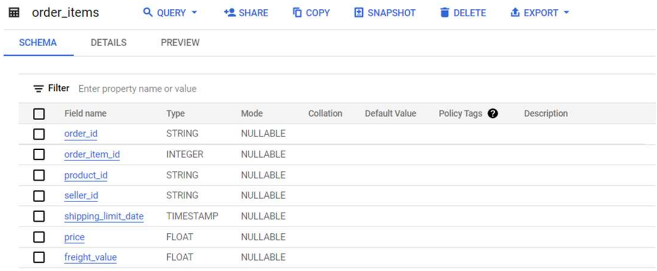
There are 5 columns namely customer_id, customer_inique_id, customer_zip_code_prefix, customer_city and customer_state

b. Geolocation table



□	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
□	geolocation_zip_code_prefix	INTEGER	NULLABLE				
□	geolocation_lat	FLOAT	NULLABLE				
□	geolocation_lng	FLOAT	NULLABLE				
□	geolocation_city	STRING	NULLABLE				
□	geolocation_state	STRING	NULLABLE				

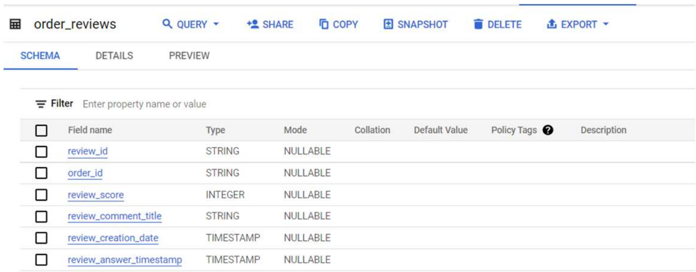
c. Order_Items



□	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
□	order_id	STRING	NULLABLE				
□	order_item_id	INTEGER	NULLABLE				
□	product_id	STRING	NULLABLE				
□	seller_id	STRING	NULLABLE				
□	shipping_limit_date	TIMESTAMP	NULLABLE				
□	price	FLOAT	NULLABLE				
□	freight_value	FLOAT	NULLABLE				

There are 7 columns namely order_id, order_item_id, product_id, seller_id, shipping_limit_date, price, and freight_value

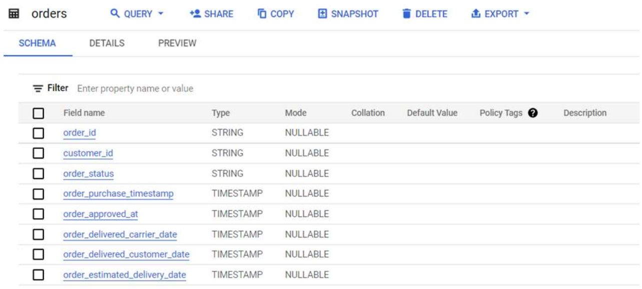
d. Order_reviews



□	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
□	review_id	STRING	NULLABLE				
□	order_id	STRING	NULLABLE				
□	review_score	INTEGER	NULLABLE				
□	review_comment_title	STRING	NULLABLE				
□	review_creation_date	TIMESTAMP	NULLABLE				
□	review_answer_timestamp	TIMESTAMP	NULLABLE				

There are 6 columns namely review_id, order_id, review_score, review_comment_title, review_creation_date and review_answer_timestamp

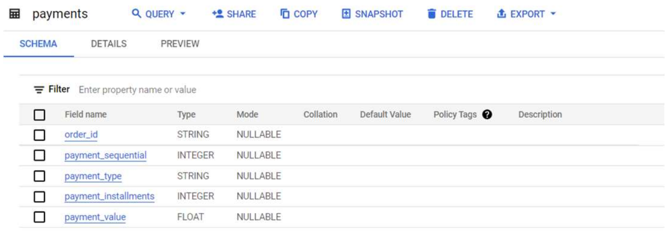
e. Orders



□	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
□	order_id	STRING	NULLABLE				
□	customer_id	STRING	NULLABLE				
□	order_status	STRING	NULLABLE				
□	order_purchase_timestamp	TIMESTAMP	NULLABLE				
□	order_approved_at	TIMESTAMP	NULLABLE				
□	order_delivered_carrier_date	TIMESTAMP	NULLABLE				
□	order_delivered_customer_date	TIMESTAMP	NULLABLE				
□	order_estimated_delivery_date	TIMESTAMP	NULLABLE				

There are 8 columns namely order_id, customer_id, order_status, order_purchase_timestamp, ofer_approved_at, order_delivered_carrier_date, order_delivered_customer_date and order_estimated_delivery_date

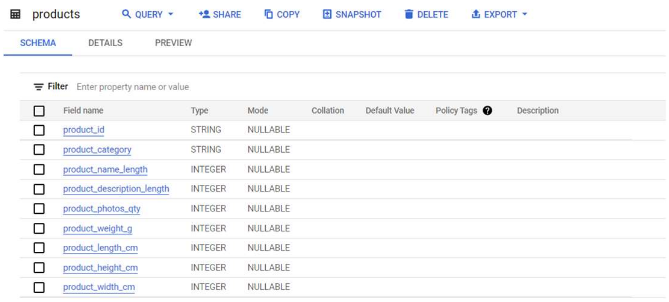
f. Payments



□	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
□	order_id	STRING	NULLABLE				
□	payment_sequential	INTEGER	NULLABLE				
□	payment_type	STRING	NULLABLE				
□	payment_installments	INTEGER	NULLABLE				
□	payment_value	FLOAT	NULLABLE				

There are 5 columns namely order_id, paymenet_sequential, payment_tupe, payment_installments and payment_value

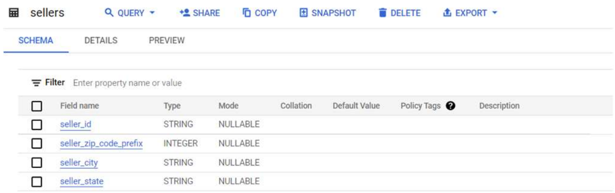
g. products



□	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
□	product_id	STRING	NULLABLE				
□	product_category	STRING	NULLABLE				
□	product_name_length	INTEGER	NULLABLE				
□	product_description_length	INTEGER	NULLABLE				
□	product_photos_qty	INTEGER	NULLABLE				
□	product_weight_g	INTEGER	NULLABLE				
□	product_length_cm	INTEGER	NULLABLE				
□	product_height_cm	INTEGER	NULLABLE				
□	product_width_cm	INTEGER	NULLABLE				

There are 9 columns namely product_id, product_category, product_name_length, procut_description_length, product_photos_qty, product_weight_g, procduct_length_cum, product_height_cum, and product_width_cum.

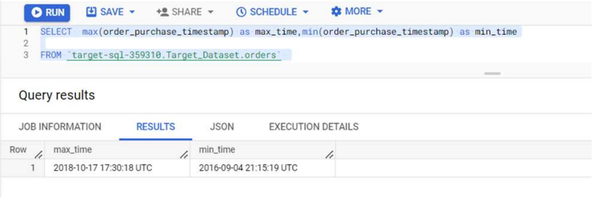
h. sellers



Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
seller_id	STRING	NULLABLE				
seller_zip_code_prefix	INTEGER	NULLABLE				
seller_city	STRING	NULLABLE				
seller_state	STRING	NULLABLE				

There are 5 columns namely seller_id, seller_zip_code_prefix, seller_city, and seller_state

B. Time period for which the data is given



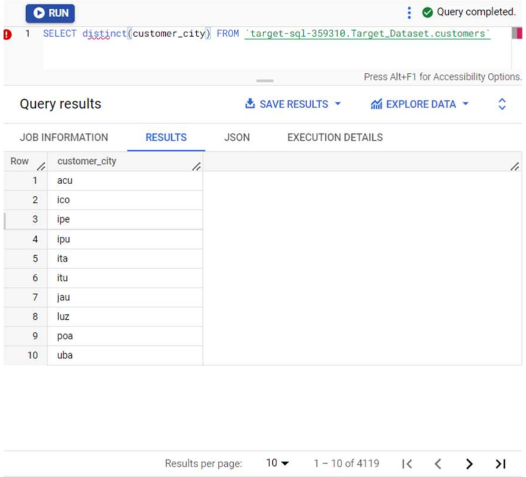
```
SELECT  max(order_purchase_timestamp) as max_time,min(order_purchase_timestamp) as min_time

FROM `target-sql-359310.Target_Dataset.orders`
```

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

This data is given for the time period from 2016-09-04 21:15:19 UTC to 2018-10-17 17:30:18 UTC

C. Cities and States covered in the dataset

Cities



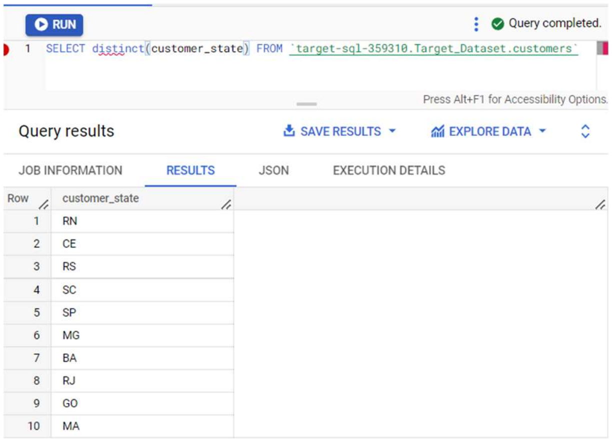
```
SELECT distinct(customer_city) FROM `target-sql-359310.Target_Dataset.customers`
```

Row	customer_city
1	acu
2	ico
3	ipe
4	ipu
5	ita
6	itu
7	jau
8	luz
9	poa
10	uba

Results per page: 10 1 – 10 of 4119

Total 4119 cities have been identified from where Retail Giant Services

States



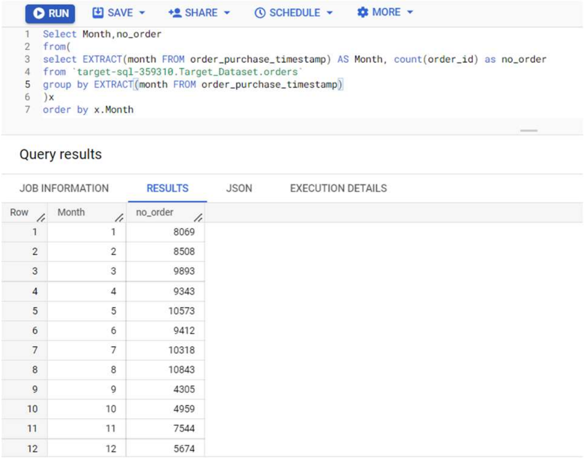
```sql
SELECT distinct(customer_state) FROM `target-sql-359310.Target_Dataset.customers`
```

Row	customer_state
1	RN
2	CE
3	RS
4	SC
5	SP
6	MG
7	BA
8	RJ
9	GO
10	MA

A total of 27 States was identified where Retail Giant Services

2) In-depth Exploration

A. Monthwise order flow (seasonality)

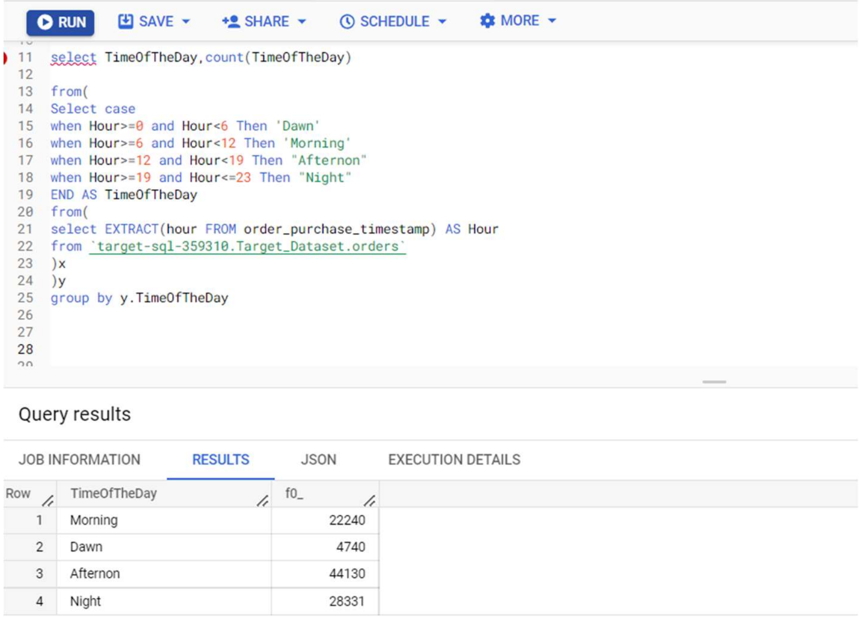


```sql
Select Month,no_order
from(
select EXTRACT(month FROM order_purchase_timestamp) AS Month, count(order_id) as no_order
from `target-sql-359310.Target_Dataset.orders`
group by EXTRACT(month FROM order_purchase_timestamp)
)x
order by x.Month
```

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

The number of orders increases gradually from the start of the year and peaks during the months from May-Aug. After that number of orders reduces.

B. Time at which Brazil people tend to buy



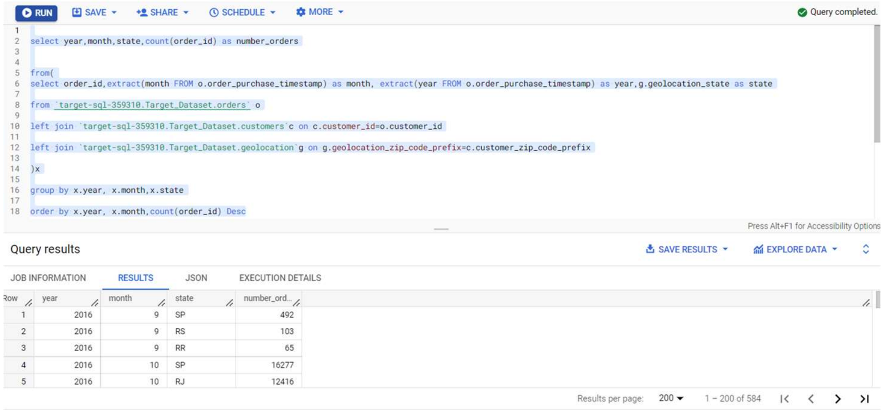
```sql
select TimeOfTheDay,count(TimeOfTheDay)

from(
Select case
when Hour>=0 and Hour<6 Then 'Dawn'
when Hour>=6 and Hour<12 Then 'Morning'
when Hour>=12 and Hour<19 Then "Afternon"
when Hour>=19 and Hour<=23 Then "Night"
END AS TimeOfTheDay
from(
select EXTRACT(hour FROM order_purchase_timestamp) AS Hour
from `target-sql-359310.Target_Dataset.orders`
)x
)y
group by y.TimeOfTheDay
```

Row	TimeOfTheDay	f0_
1	Morning	22240
2	Dawn	4740
3	Afternon	44130
4	Night	28331

- More people tend to buy during the afternoon session (maybe during lunchtime) and followed by the morning time.
- During Dawn least people make the purchases

3) Evolution of E-commerce orders in the Brazil region

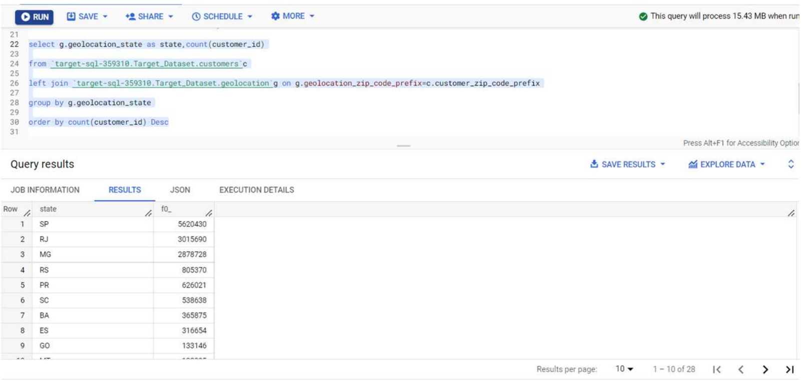
A. Get month-on-month orders by region, states



```sql

select year,month,state,count(order_id) as number_orders


from(
select order_id,extract(month FROM o.order_purchase_timestamp) as month, extract(year FROM o.order_purchase_timestamp) as year,g.geolocation_state as state

from `target-sql-359310.Target_Dataset.orders` o

left join `target-sql-359310.Target_Dataset.customers`c on c.customer_id=o.customer_id

left join `target-sql-359310.Target_Dataset.geolocation`g on g.geolocation_zip_code_prefix=c.customer_zip_code_prefix

)x

group by x.year, x.month,x.state

order by x.year, x.month,count(order_id) Desc
```

Row	year	month	state	number_ord...
1	2016	9	SP	492
2	2016	9	RS	103
3	2016	9	RR	65
4	2016	10	SP	16277
5	2016	10	RJ	12416

B. How are customers distributed in Brazil



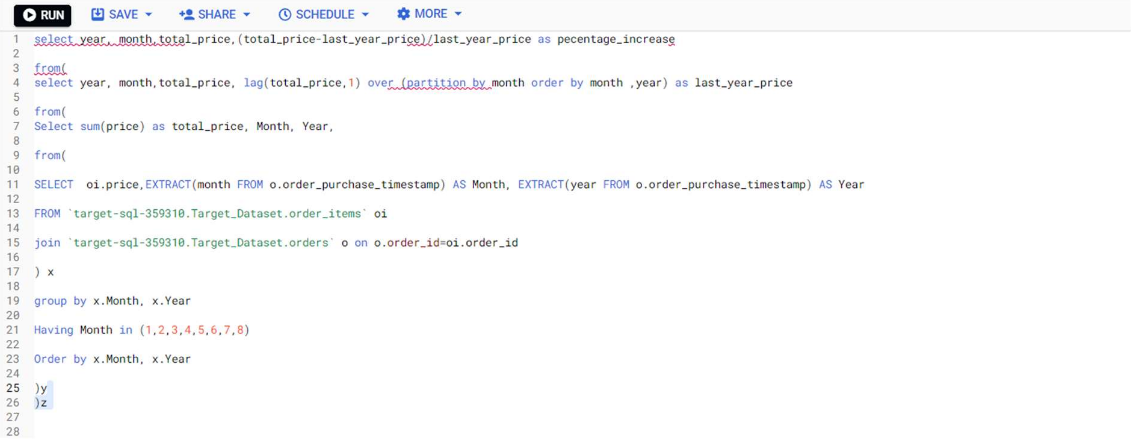
```
select g.geolocation_state as state,count(customer_id)

from `target-sql-359310.Target_Dataset.customers`c

left join `target-sql-359310.Target_Dataset.geolocation`g on g.geolocation_zip_code_prefix=c.customer_zip_code_prefix

group by g.geolocation_state

order by count(customer_id) Desc
```

Row	state	f0_
1	SP	5620430
2	RJ	3015690
3	MG	2878728
4	RS	805370
5	PR	626021
6	SC	538638
7	BA	365875
8	ES	316654
9	GO	133146

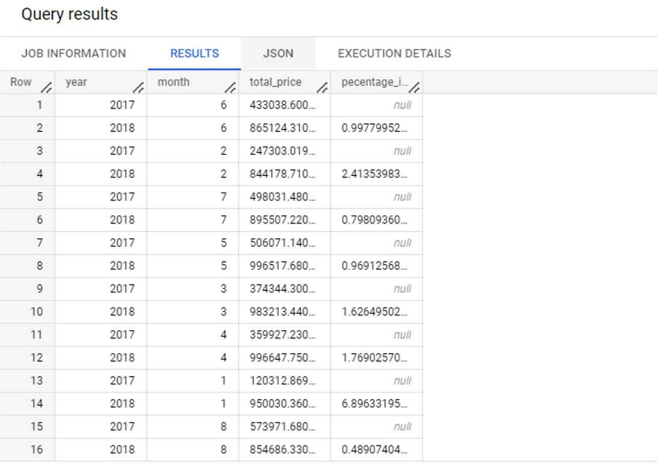
More customer are from SP, RJ and MG.

4) Impact on Economy

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



```
select year, month,total_price,(total_price-last_year_price)/last_year_price as pecentage_increase

from(
select year, month,total_price, lag(total_price,1) over (partition by month order by month ,year) as last_year_price

from(
Select sum(price) as total_price, Month, Year,

from(

SELECT  oi.price,EXTRACT(month FROM o.order_purchase_timestamp) AS Month, EXTRACT(year FROM o.order_purchase_timestamp) AS Year

FROM `target-sql-359310.Target_Dataset.order_items` oi

join `target-sql-359310.Target_Dataset.orders` o on o.order_id=oi.order_id

) x

group by x.Month, x.Year

Having Month in (1,2,3,4,5,6,7,8)

Order by x.Month, x.Year

)y
)z
```



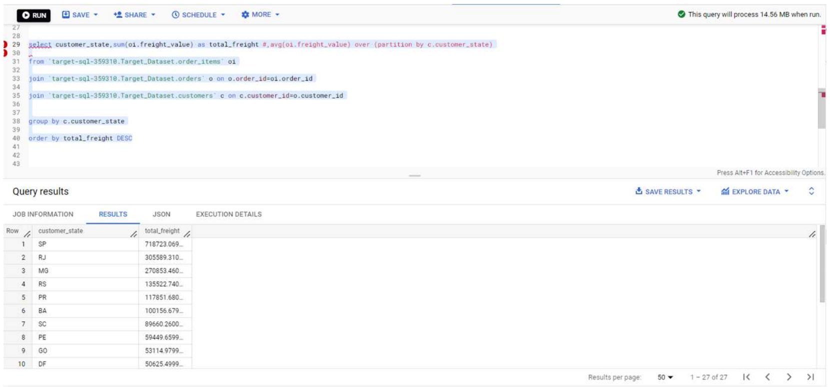
Row	year	month	total_price	pecentage_i…
1	2017	6	433038.600…	null
2	2018	6	865124.310…	0.99779952…
3	2017	2	247303.019…	null
4	2018	2	844178.710…	2.41353983…
5	2017	7	498031.480…	null
6	2018	7	895507.220…	0.79809360…
7	2017	5	506071.140…	null
8	2018	5	996517.680…	0.96912568…
9	2017	3	374344.300…	null
10	2018	3	983213.440…	1.62649502…
11	2017	4	359927.230…	null
12	2018	4	996647.750…	1.76902570…
13	2017	1	120312.869…	null
14	2018	1	950030.360…	6.89633195…
15	2017	8	573971.680…	null
16	2018	8	854686.330…	0.48907404…

Month on month increase in the total value of sales was found. The maximum increase happened in the month of Jan2017 to Jan2018. Sales almost became 7 times.

This is followed by Feb2017 to Feb2018 with the sale going up more than 2.4 times.

B. Mean & Sum of price and freight value by customer state

a. Sum

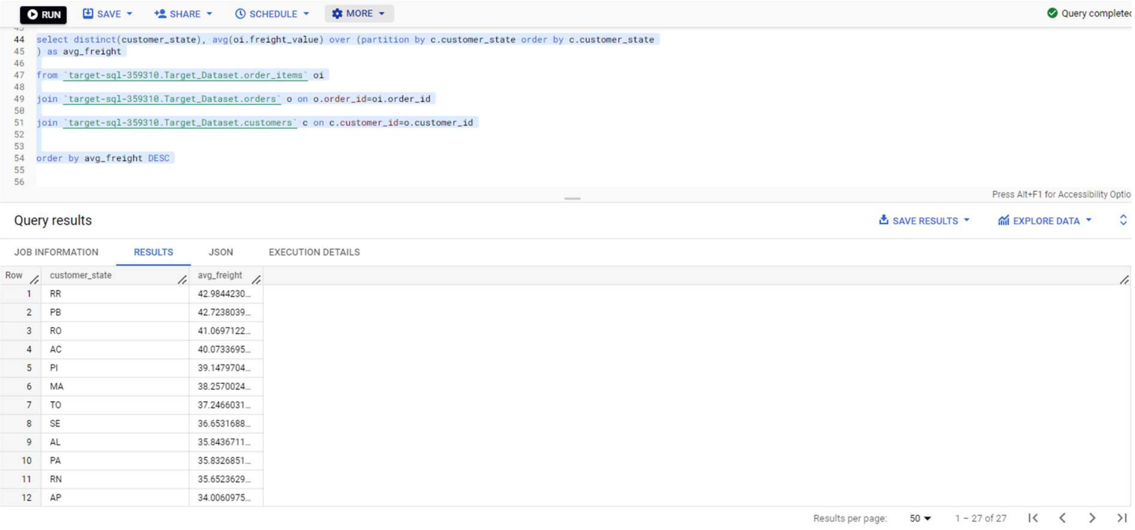


```
select customer_state,sum(oi.freight_value) as total_freight #,avg(oi.freight_value) over (partition by c.customer_state)

from `target-sql-359310.Target_Dataset.order_items` oi

join `target-sql-359310.Target_Dataset.orders` o on o.order_id=oi.order_id

join `target-sql-359310.Target_Dataset.customers` c on c.customer_id=o.customer_id


group by c.customer_state

order by total_freight DESC
```

Row	customer_state	total_freight
1	SP	718723.069...
2	RJ	305589.310...
3	MG	270853.460...
4	RS	135522.740...
5	PR	117851.680...
6	BA	100156.679...
7	SC	89660.2600...
8	PE	59449.6599...
9	GO	53114.9799...
10	DF	50625.4999...

The freight charges are more being paid by the people in the cities SP and RJ. This may be because of the long delivery distance to these cities from sellers.

b. Mean

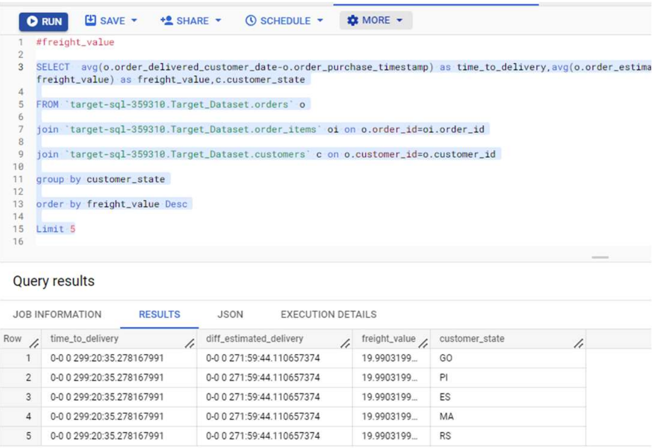


```
select distinct(customer_state), avg(oi.freight_value) over (partition by c.customer_state order by c.customer_state
) as avg_freight

from `target-sql-359310.Target_Dataset.order_items` oi

join `target-sql-359310.Target_Dataset.orders` o on o.order_id=oi.order_id

join `target-sql-359310.Target_Dataset.customers` c on c.customer_id=o.customer_id


order by avg_freight DESC
```

Row	customer_state	avg_freight
1	RR	42.9844230...
2	PB	42.7238039...
3	RO	41.0697122...
4	AC	40.0733695...
5	PI	39.1479704...
6	MA	38.2570024...
7	TO	37.2466031...
8	SE	36.6531688...
9	AL	35.8436711...
10	PA	35.8326851...
11	RN	35.6523629...
12	AP	34.0060975...

On average, the delivery charges per order remain the same for all cities with RR and PB coming on the top with 42.98 and 42.72 per order respectively.

5) Analysis of sales, freight, and delivery time

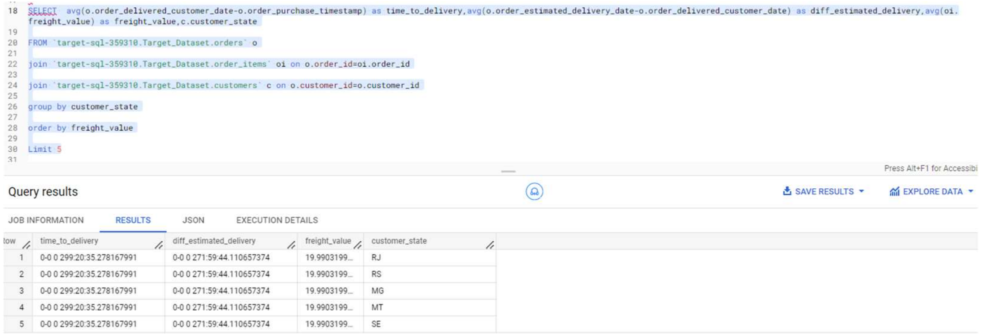


```
#freight_value

SELECT  avg(o.order_delivered_customer_date-o.order_purchase_timestamp) as time_to_delivery,avg(o.order_estima
freight_value) as freight_value,c.customer_state

FROM `target-sql-359310.Target_Dataset.orders` o

join `target-sql-359310.Target_Dataset.order_items` oi on o.order_id=oi.order_id

join `target-sql-359310.Target_Dataset.customers` c on o.customer_id=o.customer_id

group by customer_state

order by freight_value Desc

Limit 5
```

Query results

Row	time_to_delivery	diff_estimated_delivery	freight_value	customer_state
1	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	GO
2	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	PI
3	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	ES
4	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	MA
5	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	RS

Top 5 states with the highest freight rate is GO, PI, ES, MA, RS

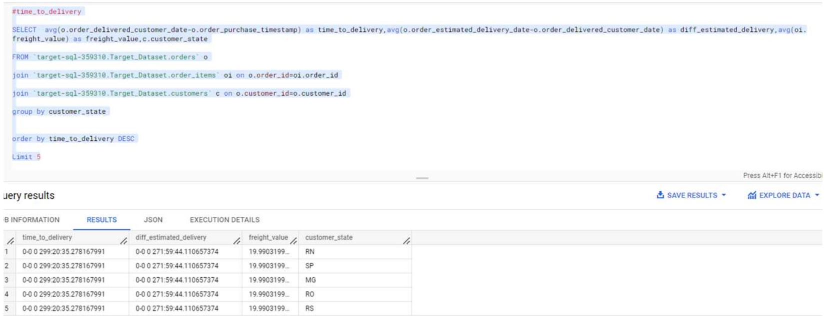


```
SELECT  avg(o.order_delivered_customer_date-o.order_purchase_timestamp) as time_to_delivery,avg(o.order_estimated_delivery_date-o.order_delivered_customer_date) as diff_estimated_delivery,avg(oi.
freight_value) as freight_value,c.customer_state

FROM `target-sql-359310.Target_Dataset.orders` o

join `target-sql-359310.Target_Dataset.order_items` oi on o.order_id=oi.order_id

join `target-sql-359310.Target_Dataset.customers` c on o.customer_id=o.customer_id

group by customer_state

order by freight_value

Limit 5
```

Query results

tow	time_to_delivery	diff_estimated_delivery	freight_value	customer_state
1	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	RJ
2	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	RS
3	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	MG
4	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	MT
5	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	SE

Top 5 states with the Lowest freight rate is RJ, RS, MG, MT, SE

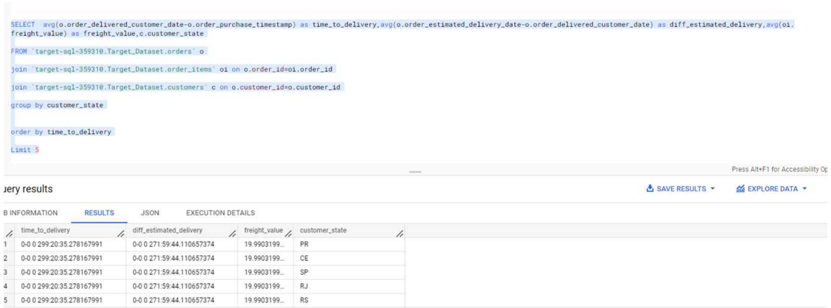


```
#time_to_delivery

SELECT  avg(o.order_delivered_customer_date-o.order_purchase_timestamp) as time_to_delivery,avg(o.order_estimated_delivery_date-o.order_delivered_customer_date) as diff_estimated_delivery,avg(oi.
freight_value) as freight_value,c.customer_state

FROM `target-sql-359310.Target_Dataset.orders` o

join `target-sql-359310.Target_Dataset.order_items` oi on o.order_id=oi.order_id

join `target-sql-359310.Target_Dataset.customers` c on o.customer_id=o.customer_id

group by customer_state


order by time_to_delivery DESC

Limit 5
```

uery results

	time_to_delivery	diff_estimated_delivery	freight_value	customer_state
1	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	RN
2	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	SP
3	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	MG
4	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	RO
5	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	RS

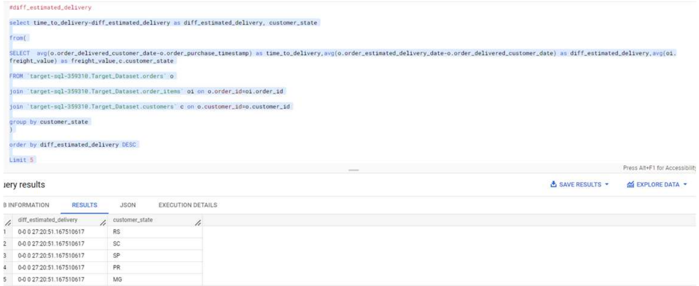
Average Delivery time is highest in states RN, SP, MG, RO and RS



```
SELECT  avg(o.order_delivered_customer_date-o.order_purchase_timestamp) as time_to_delivery,avg(o.order_estimated_delivery_date-o.order_delivered_customer_date) as diff_estimated_delivery,avg(oi.
freight_value) as freight_value,c.customer_state

FROM `target-sql-359310.Target_Dataset.orders` o

join `target-sql-359310.Target_Dataset.order_items` oi on o.order_id=oi.order_id

join `target-sql-359310.Target_Dataset.customers` c on o.customer_id=o.customer_id

group by customer_state


order by time_to_delivery

Limit 5
```

	time_to_delivery	diff_estimated_delivery	freight_value	customer_state
1	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	PR
2	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	CE
3	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	SP
4	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	RJ
5	0-0 0 299:20:35.278167991	0-0 0 271:59:44.110657374	19.9903199...	RS

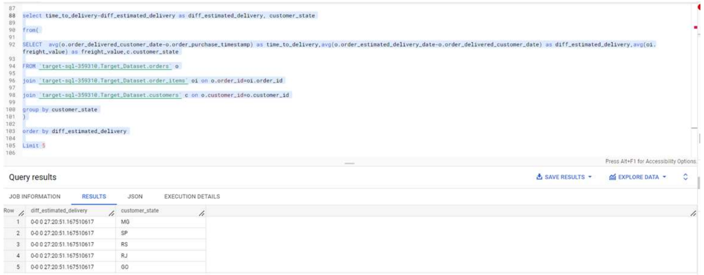
Average Delivery time is lowest in states PR, CE, SP, RJ and RS



```
#diff_estimated_delivery

select time_to_delivery-diff_estimated_delivery as diff_estimated_delivery, customer_state

from(

SELECT  avg(o.order_delivered_customer_date-o.order_purchase_timestamp) as time_to_delivery,avg(o.order_estimated_delivery_date-o.order_delivered_customer_date) as diff_estimated_delivery,avg(oi.
freight_value) as freight_value,c.customer_state

FROM `target-sql-359310.Target_Dataset.orders` o

join `target-sql-359310.Target_Dataset.order_items` oi on o.order_id=oi.order_id

join `target-sql-359310.Target_Dataset.customers` c on o.customer_id=o.customer_id

group by customer_state
)

order by diff_estimated_delivery DESC

Limit 5
```

	diff_estimated_delivery	customer_state
1	0-0 0 27:20:51.167510617	RS
2	0-0 0 27:20:51.167510617	SC
3	0-0 0 27:20:51.167510617	SP
4	0-0 0 27:20:51.167510617	PR
5	0-0 0 27:20:51.167510617	MG

Average Estimated Delivery time is highest in states RS, SC, SP, PR and MG

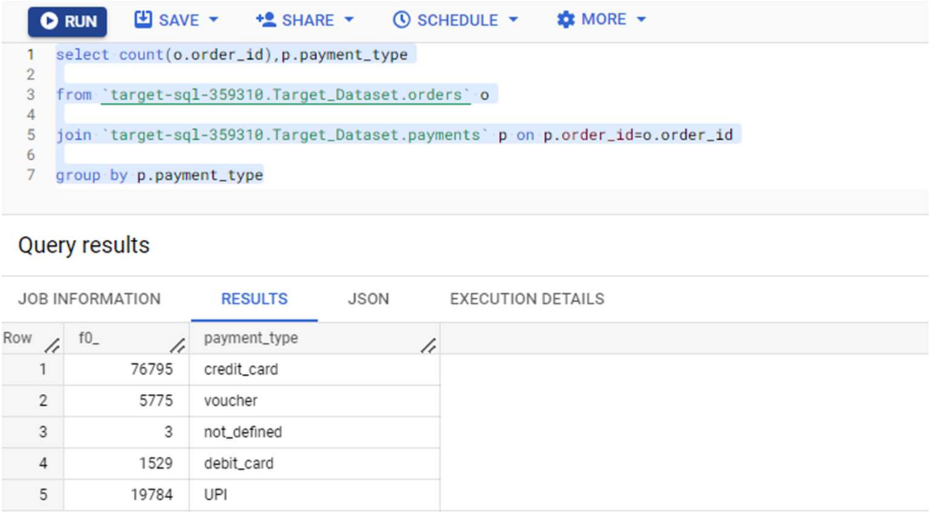


```

select time_to_delivery-diff_estimated_delivery as diff_estimated_delivery, customer_state

from(

SELECT  avg(o.order_delivered_customer_date-o.order_purchase_timestamp) as time_to_delivery,avg(o.order_estimated_delivery_date-o.order_delivered_customer_date) as diff_estimated_delivery,avg(oi.
freight_value) as freight_value,c.customer_state

FROM `target-sql-359310.Target_Dataset.orders` o

join `target-sql-359310.Target_Dataset.order_items` oi on o.order_id=oi.order_id

join `target-sql-359310.Target_Dataset.customers` c on o.customer_id=o.customer_id

group by customer_state
)

order by diff_estimated_delivery

Limit 5

```

Row	diff_estimated_delivery	customer_state
1	0-0 0 27:20:51.167510617	MG
2	0-0 0 27:20:51.167510617	SP
3	0-0 0 27:20:51.167510617	RS
4	0-0 0 27:20:51.167510617	RJ
5	0-0 0 27:20:51.167510617	GO

Average Estimated Delivery time is lowest in states MG, SP, RS, RJ, GO

6) Payment type analysis

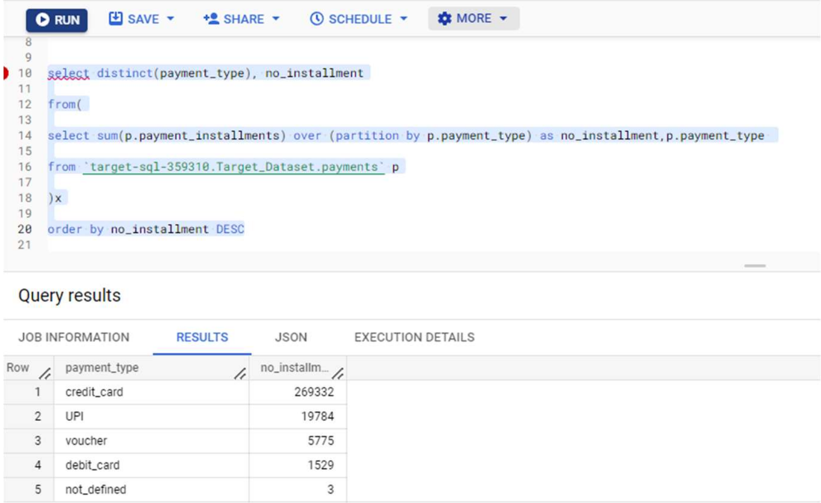


```
select count(o.order_id),p.payment_type

from `target-sql-359310.Target_Dataset.orders` o

join `target-sql-359310.Target_Dataset.payments` p on p.order_id=o.order_id

group by p.payment_type
```

Query results

Row	f0_	payment_type
1	76795	credit_card
2	5775	voucher
3	3	not_defined
4	1529	debit_card
5	19784	UPI

More people are using the credit card for transactions followed by UPI.

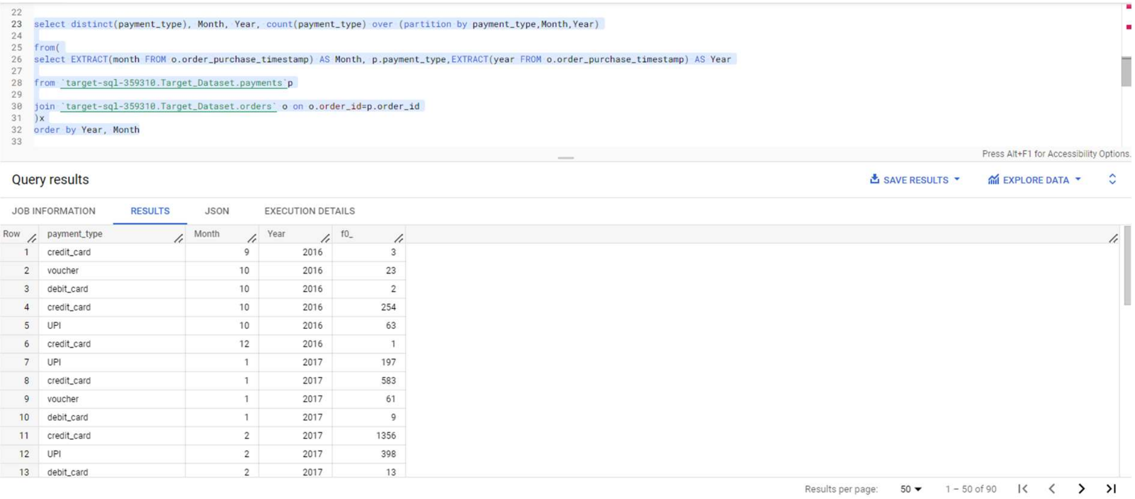


```


select distinct(payment_type), no_installment

from(

select sum(p.payment_installments) over (partition by p.payment_type) as no_installment,p.payment_type

from `target-sql-359310.Target_Dataset.payments` p

)x

order by no_installment DESC

```

Query results

Row	payment_type	no_installm...
1	credit_card	269332
2	UPI	19784
3	voucher	5775
4	debit_card	1529
5	not_defined	3

More people are taking instalments in credit card followed by UPI.



```
select distinct(payment_type), Month, Year, count(payment_type) over (partition by payment_type,Month,Year)

from(
select EXTRACT(month FROM o.order_purchase_timestamp) AS Month, p.payment_type,EXTRACT(year FROM o.order_purchase_timestamp) AS Year

from `target-sql-359310.Target_Dataset.payments`p

join `target-sql-359310.Target_Dataset.orders` o on o.order_id=p.order_id
)x
order by Year, Month

```

Row	payment_type	Month	Year	f0_
1	credit_card	9	2016	3
2	voucher	10	2016	23
3	debit_card	10	2016	2
4	credit_card	10	2016	254
5	UPI	10	2016	63
6	credit_card	12	2016	1
7	UPI	1	2017	197
8	credit_card	1	2017	583
9	voucher	1	2017	61
10	debit_card	1	2017	9
11	credit_card	2	2017	1356
12	UPI	2	2017	398
13	debit_card	2	2017	13

This query was used to find the month on month payment type preferred by the customers