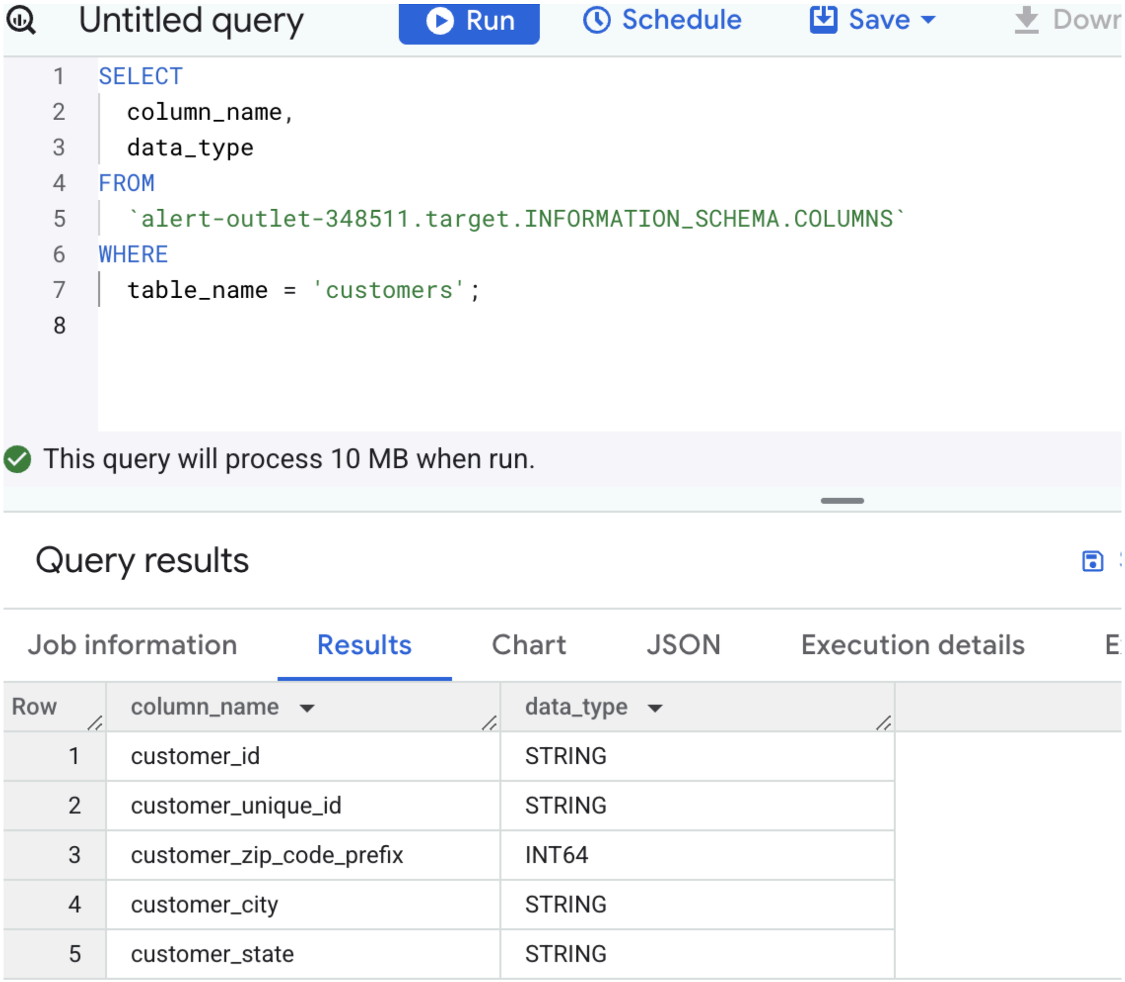
1.Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

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

```
SELECT
 column_name,
 data_type
FROM
 `alert-outlet-348511.target.INFORMATION_SCHEMA.COLUMNS`
WHERE
 table_name = 'customers';
```



```
SELECT
  column_name,
  data_type
FROM
  `alert-outlet-348511.target.INFORMATION_SCHEMA.COLUMNS`
WHERE
  table_name = 'customers';
```

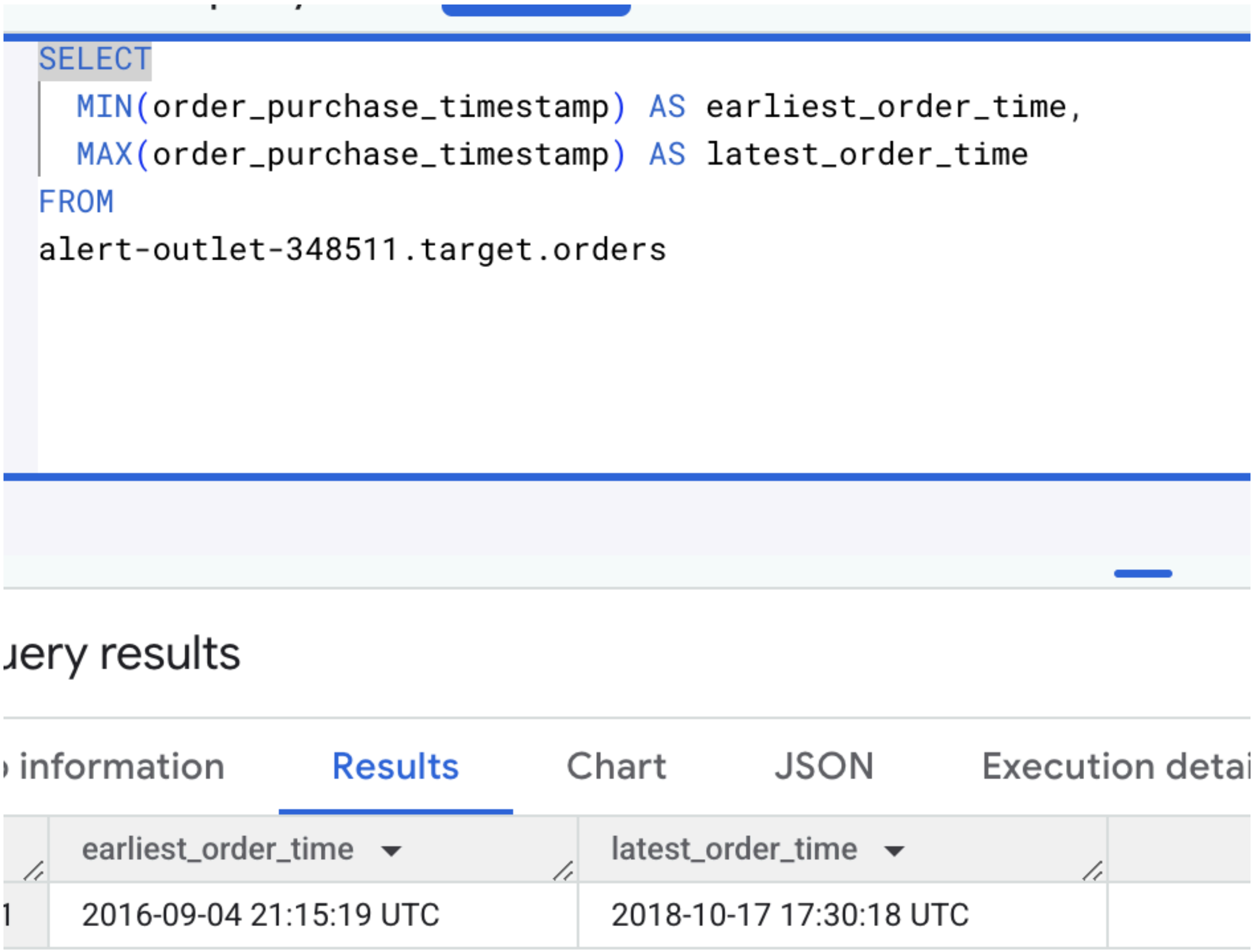
This query will process 10 MB when run.

Query results

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

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

```
SELECT
 MIN(order_purchase_timestamp) AS earliest_order_time,
 MAX(order_purchase_timestamp) AS latest_order_time
FROM
alert-outlet-348511.target.orders
```



```
SELECT
  MIN(order_purchase_timestamp) AS earliest_order_time,
  MAX(order_purchase_timestamp) AS latest_order_time
FROM
alert-outlet-348511.target.orders
```

uery results

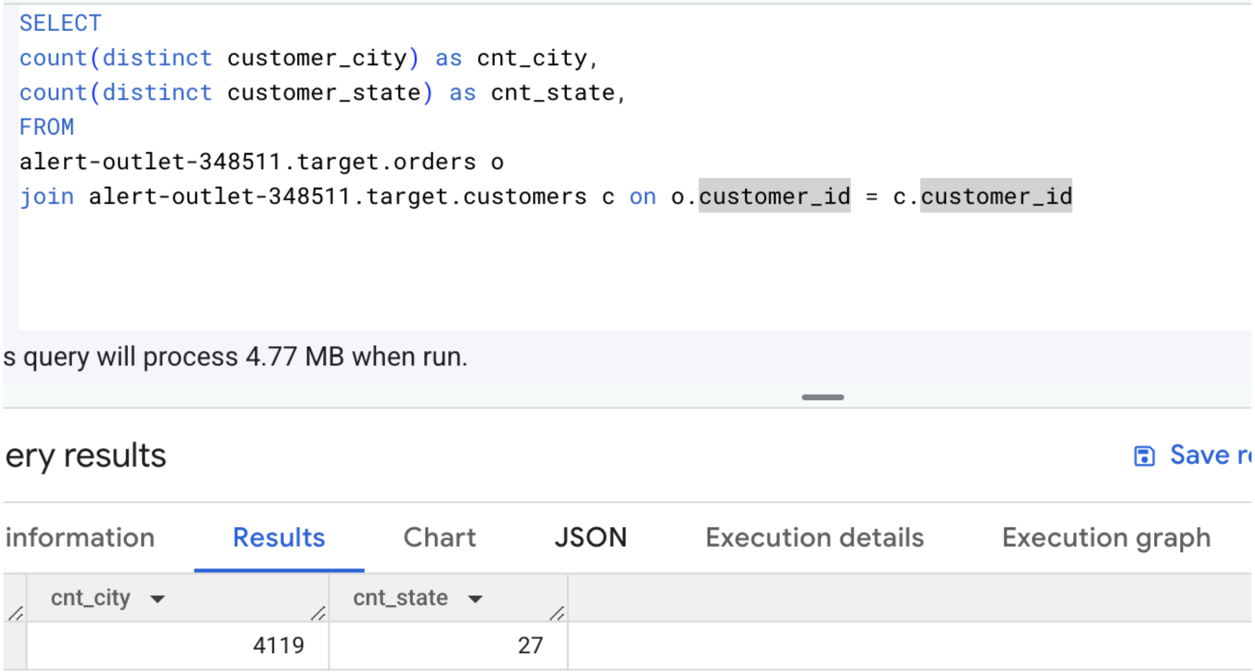
	earliest_order_time	latest_order_time
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

First ever order placed on 2016/09/04
And recent order placed on 2018/10/17

We can say we have the data of 2 years to deal with.

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

```
SELECT
count(distinct customer_city) as cnt_city,

count(distinct customer_state) as cnt_state,

FROM

alert-outlet-348511.target.orders o

join alert-outlet-348511.target.customers c on o.customer_id = c.customer_id
```



```
SELECT
count(distinct customer_city) as cnt_city,
count(distinct customer_state) as cnt_state,
FROM
alert-outlet-348511.target.orders o
join alert-outlet-348511.target.customers c on o.customer_id = c.customer_id
```

s query will process 4.77 MB when run.

ery results

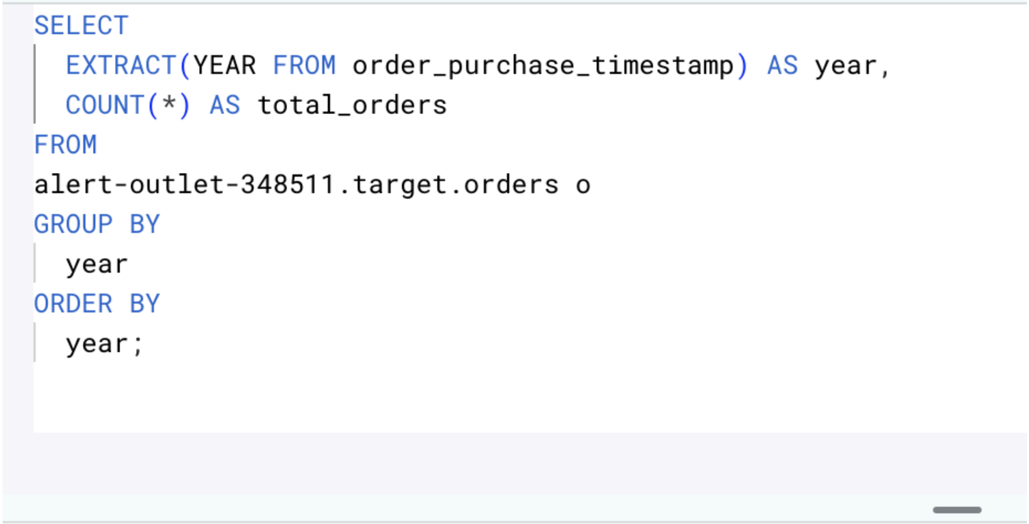
cnt_city	cnt_state
4119	27

There are 27 states out of those states 4119 cities the orders have placed and reached.

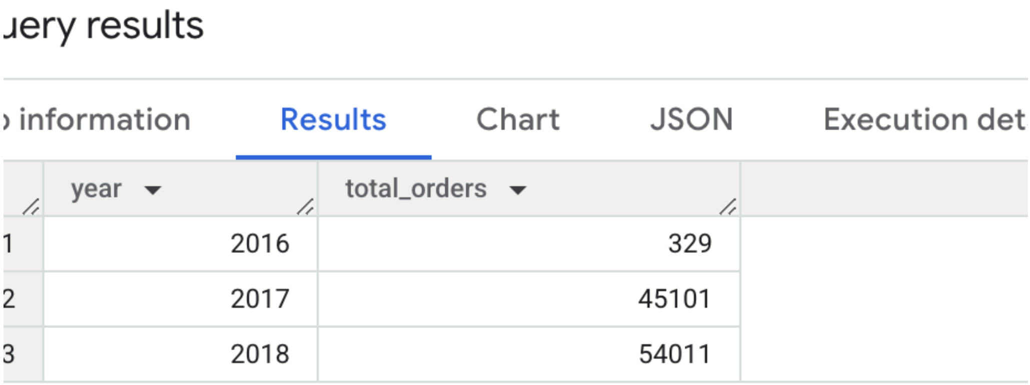
2.In-depth Exploration:

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

```
SELECT
 EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
 COUNT(*) AS total_orders
FROM
alert-outlet-348511.target.orders o
GROUP BY
 year
ORDER BY
 year;
```



```
SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
  COUNT(*) AS total_orders
FROM
alert-outlet-348511.target.orders o
GROUP BY
  year
ORDER BY
  year;
```



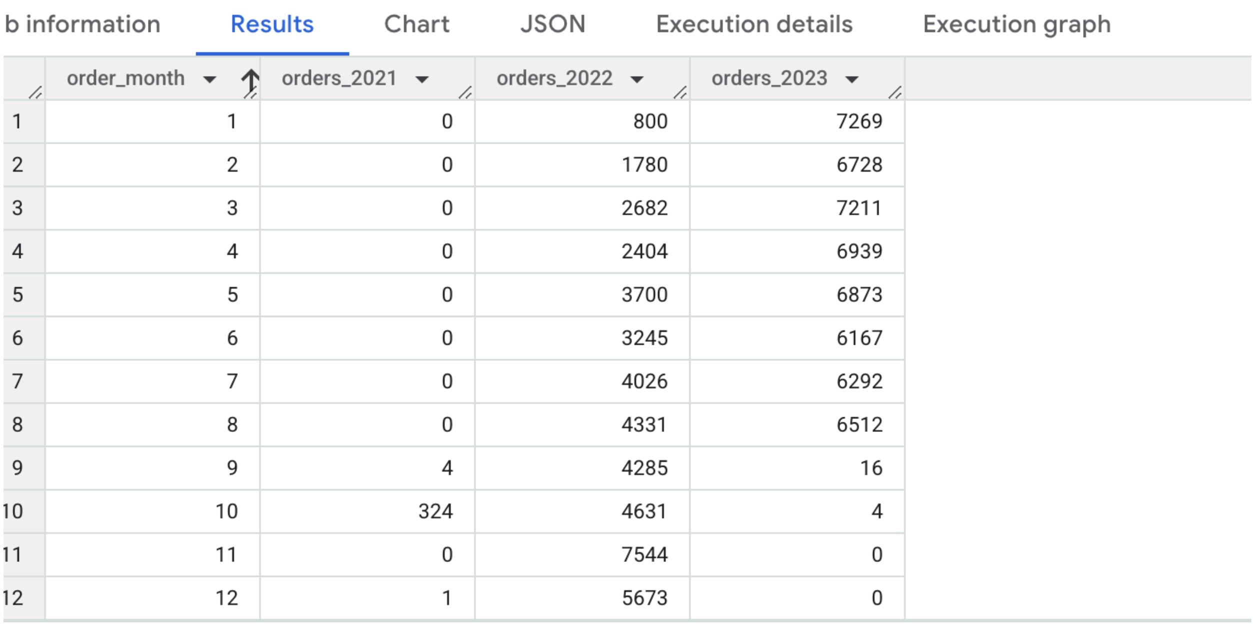
	year	total_orders
1	2016	329
2	2017	45101
3	2018	54011

We can see that the number of orders has been going up year after year, which suggests that more and more customers are placing orders over time.

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

```
 SELECT
 EXTRACT(MONTH FROM order_purchase_timestamp) AS order_month,
 COUNTIF(EXTRACT(YEAR FROM order_purchase_timestamp) = 2016) AS orders_2021,
 COUNTIF(EXTRACT(YEAR FROM order_purchase_timestamp) = 2017) AS orders_2022,
 COUNTIF(EXTRACT(YEAR FROM order_purchase_timestamp) = 2018) AS orders_2023,
FROM
alert-outlet-348511.target.orders
GROUP BY
 order_month
ORDER BY
 order_month;
```

As there is not enough data to decide weather there is a sesonality in orders but we can say in 2017 November month got the highest orders also in 2017 orders are increasing MOM (month on month) but in 2018 it kept on a constant even the orders got declined by a small margin in 2018

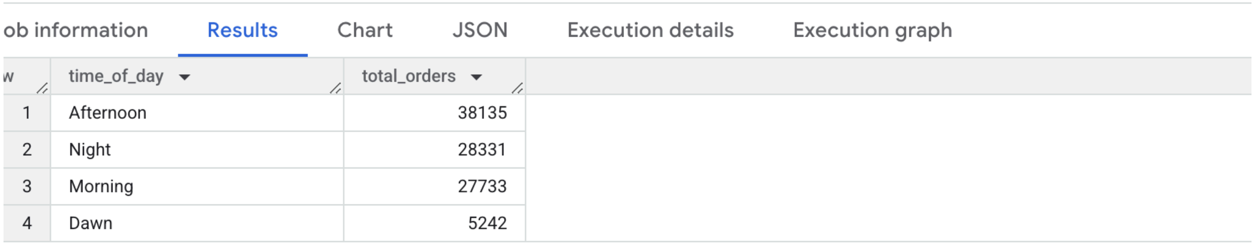


	order_month	orders_2021	orders_2022	orders_2023
1	1	0	800	7269
2	2	0	1780	6728
3	3	0	2682	7211
4	4	0	2404	6939
5	5	0	3700	6873
6	6	0	3245	6167
7	7	0	4026	6292
8	8	0	4331	6512
9	9	4	4285	16
10	10	324	4631	4
11	11	0	7544	0
12	12	1	5673	0

3.During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

```
 SELECT
CASE
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 0 AND 6 THEN 'Dawn'
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 7 AND 12 THEN 'Morning'
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 13 AND 18 THEN 'Afternoon'
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 19 AND 23 THEN 'Night'
   ELSE 'Unknown'
 END AS time_of_day,
 COUNT(*) AS total_orders
FROM
alert-outlet-348511.target.orders
GROUP BY
 time_of_day
ORDER BY
 total_orders DESC;
```

Afternoon (13 to 18 Hrs) is the time most of the Brazilian customers place their orders

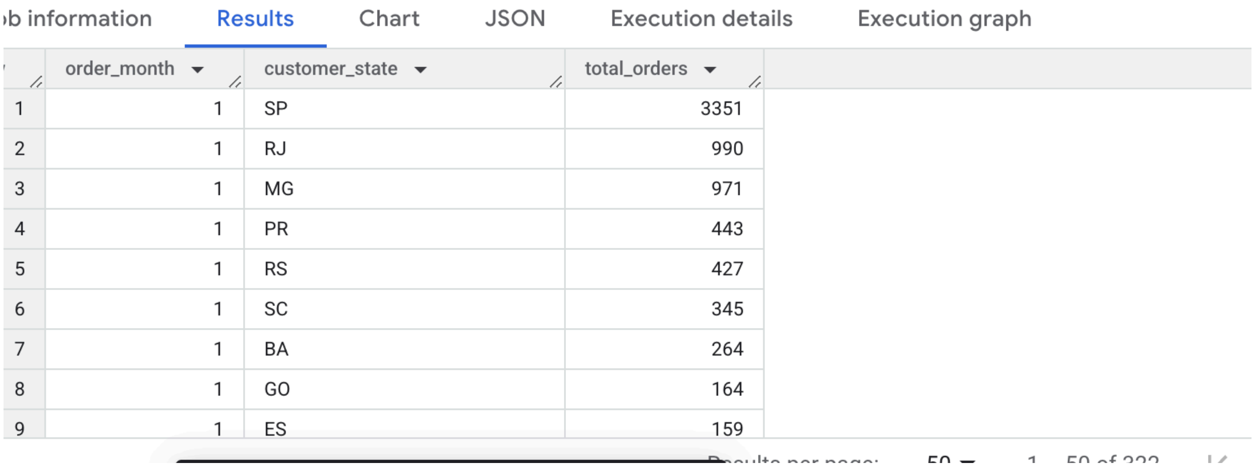


w	time_of_day	total_orders
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

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

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

```
SELECT
 EXTRACT(MONTH FROM order_purchase_timestamp) AS order_month,
 customer_state,
 COUNT(*) AS total_orders
FROM
 `alert-outlet-348511.target.orders` o
 left join alert-outlet-348511.target.customers c on c.customer_id = o.customer_id
GROUP BY
 order_month, customer_state
ORDER BY
 order_month, total_orders DESC;
```



	order_month	customer_state	total_orders
1	1	SP	3351
2	1	RJ	990
3	1	MG	971
4	1	PR	443
5	1	RS	427
6	1	SC	345
7	1	BA	264
8	1	GO	164
9	1	ES	159

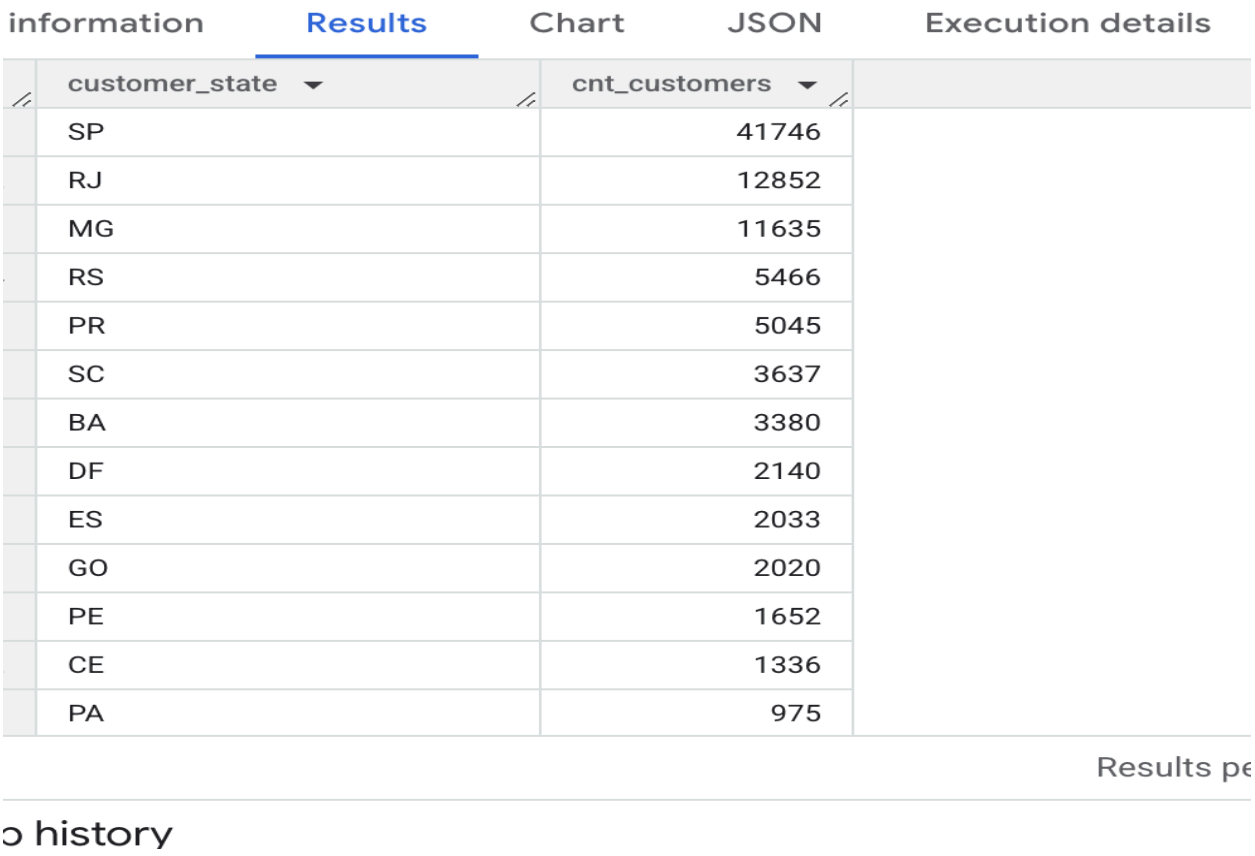
Results per page: 50 1 – 50 of 222

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

```
SELECT
 customer_state,
 COUNT(*) AS cnt_customers
FROM
alert-outlet-348511.target.customers
GROUP BY
1
ORDER BY
2 desc
```

Looking at the customer distribution, it's clear that the majority of customers are concentrated in a few key states SP stands out by far, with a massive **41,746** customers, which is significantly higher than other states. **RJ** follows with **12,852** customers, and **MG** comes in third with **11,635**.

Other states like **RS**, **PR**, and **SC** also have sizable customer bases, but the numbers start to dip as we move to states like **AC** and **RR**, which have relatively few customers—**81** in AC and just **46** in RR.

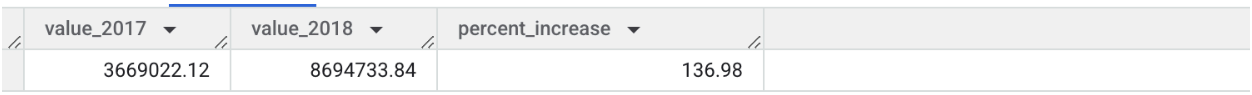


customer_state	cnt_customers
SP	41746
RJ	12852
MG	11635
RS	5466
PR	5045
SC	3637
BA	3380
DF	2140
ES	2033
GO	2020
PE	1652
CE	1336
PA	975

4.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).You can use the "payment_value" column in the payments table to get the cost of orders.

```
WITH payment_summary AS (
 SELECT
   EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
   ROUND(SUM(p.payment_value), 2) AS total_payment_value
 FROM
   `alert-outlet-348511.target.orders` AS o
 JOIN
   `alert-outlet-348511.target.payments` AS p
 USING (order_id)
 WHERE
   EXTRACT(YEAR FROM order_purchase_timestamp) IN (2017, 2018)
   AND EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8
 GROUP BY
   year
),
year_values AS (
 SELECT
   MAX(CASE WHEN year = 2017 THEN total_payment_value END) AS value_2017,
   MAX(CASE WHEN year = 2018 THEN total_payment_value END) AS value_2018
 FROM
   payment_summary
)
SELECT
 value_2017,
 value_2018,
 ROUND(((value_2018 - value_2017) / value_2017) * 100, 2) AS percent_increase
FROM
 year_values;
```

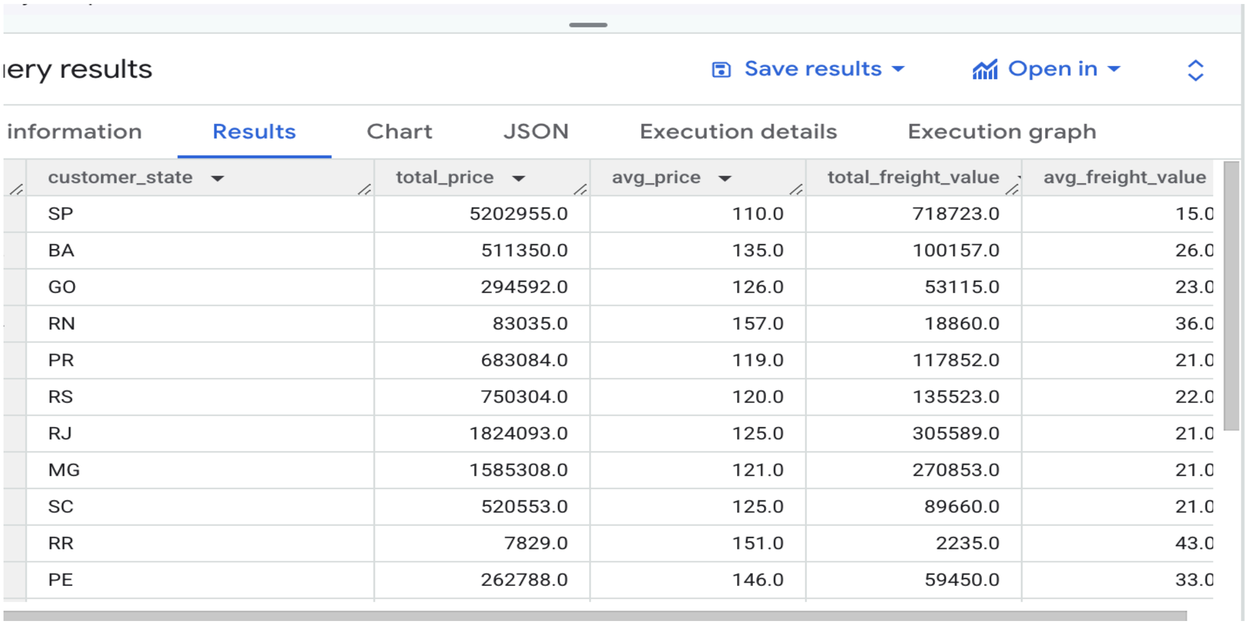


value_2017	value_2018	percent_increase
3669022.12	8694733.84	136.98

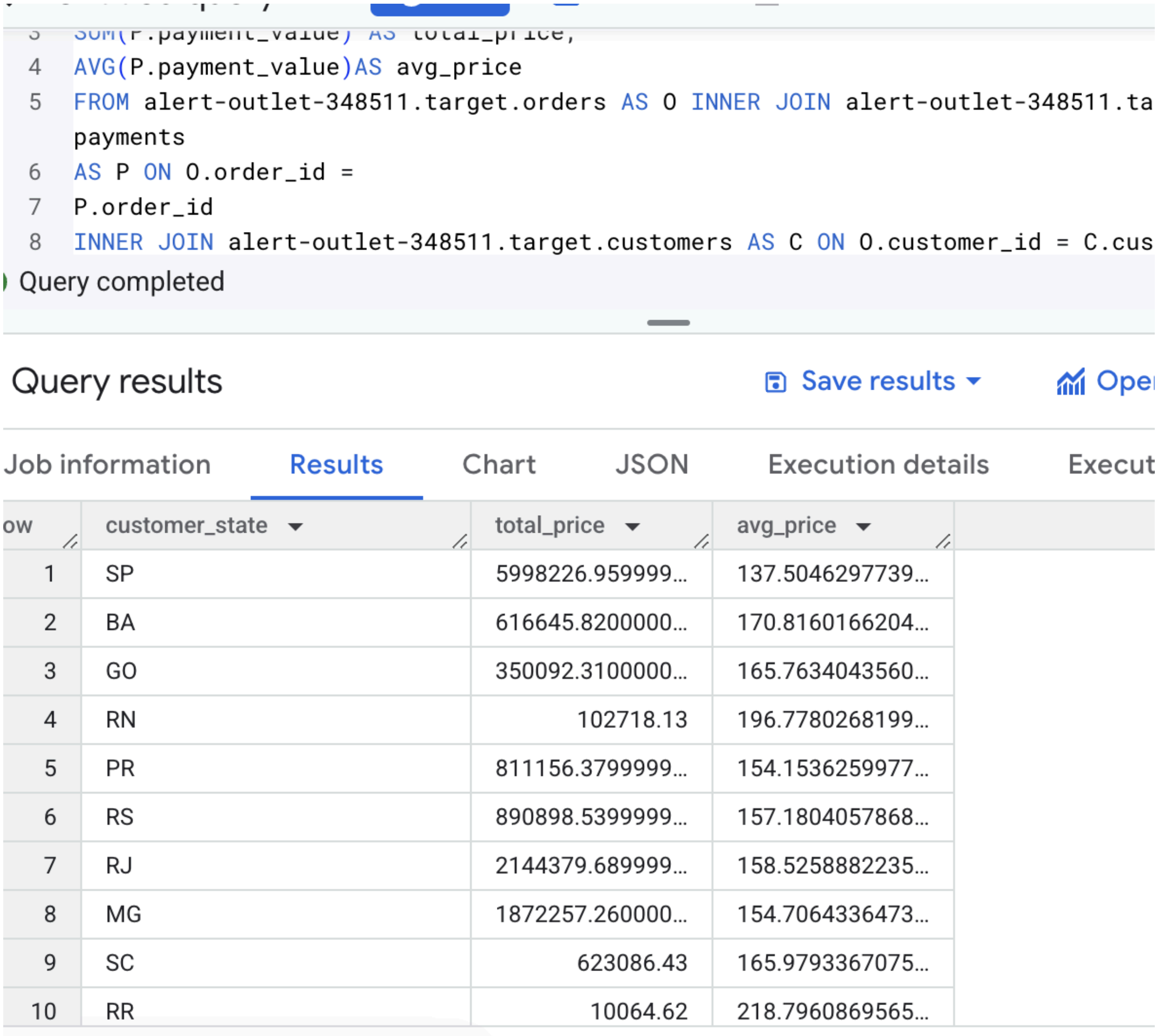
Significant 136% of increase in turnover

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

```
SELECT c.customer_state,round(sum(oi.price))as total_price,
round(avg(oi.price))as avg_price,
round(sum(oi.freight_value))as total_freight_value,
round(avg(oi.freight_value))as avg_freight_value
FROM `alert-outlet-348511.target.order_items`  oi join
alert-outlet-348511.target.orders o on oi.order_id=o.order_id
join alert-outlet-348511.target.customers c on c.customer_id= o.customer_id
group by 1
```



customer_state	total_price	avg_price	total_freight_value	avg_freight_value
SP	5202955.0	110.0	718723.0	15.0
BA	511350.0	135.0	100157.0	26.0
GO	294592.0	126.0	53115.0	23.0
RN	83035.0	157.0	18860.0	36.0
PR	683084.0	119.0	117852.0	21.0
RS	750304.0	120.0	135523.0	22.0
RJ	1824093.0	125.0	305589.0	21.0
MG	1585308.0	121.0	270853.0	21.0
SC	520553.0	125.0	89660.0	21.0
RR	7829.0	151.0	2235.0	43.0
PE	262788.0	146.0	59450.0	33.0



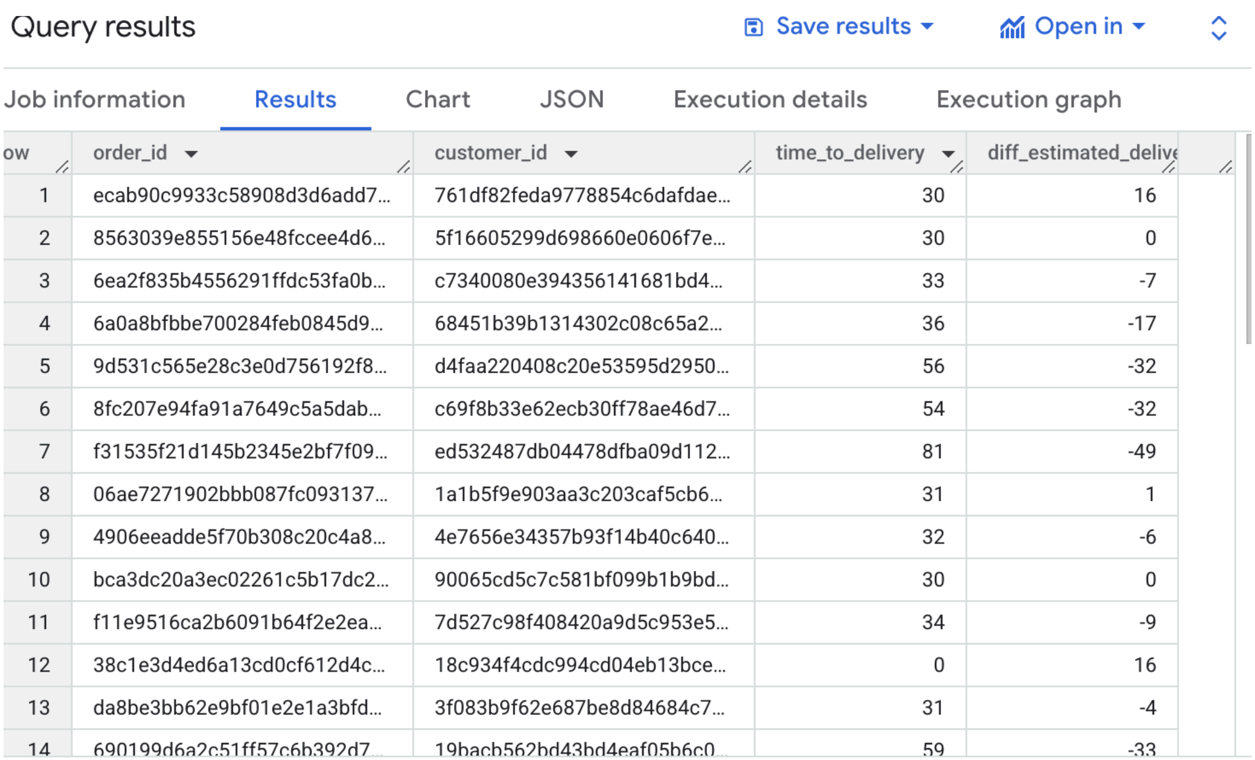
```
SUM(P.payment_value) AS total_price,
AVG(P.payment_value)AS avg_price
FROM alert-outlet-348511.target.orders AS O INNER JOIN alert-outlet-348511.ta
payments
AS P ON O.order_id =
P.order_id
INNER JOIN alert-outlet-348511.target.customers AS C ON O.customer_id = C.cus
```

ow	customer_state	total_price	avg_price
1	SP	5998226.959999...	137.5046297739...
2	BA	616645.8200000...	170.8160166204...
3	GO	350092.3100000...	165.7634043560...
4	RN	102718.13	196.7780268199...
5	PR	811156.3799999...	154.1536259977...
6	RS	890898.5399999...	157.1804057868...
7	RJ	2144379.689999...	158.5258882235...
8	MG	1872257.260000...	154.7064336473...
9	SC	623086.43	165.9793367075...
10	RR	10064.62	218.7960869565...

5.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.
   Also, calculate the difference (in days) between the estimated & actual delivery date of an order.

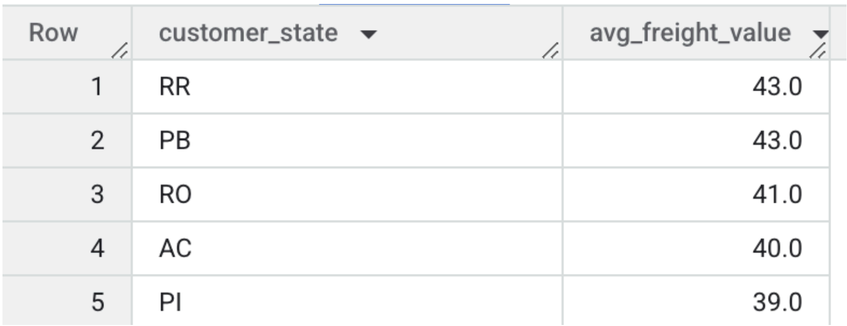
```
SELECT
 order_id,
 customer_id,
 TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
time_to_delivery,
 TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS
diff_estimated_delivery
FROM
 alert-outlet-348511.target.orders
```



ow	order_id	customer_id	time_to_delivery	diff_estimated_delive
1	ecab90c9933c58908d3d6add7…	761df82feda9778854c6dafdae…	30	16
2	8563039e855156e48fccee4d6…	5f16605299d698660e0606f7e…	30	0
3	6ea2f835b4556291ffdc53fa0b…	c7340080e394356141681bd4…	33	-7
4	6a0a8bfbbe700284feb0845d9…	68451b39b1314302c08c65a2…	36	-17
5	9d531c565e28c3e0d756192f8…	d4faa220408c20e53595d2950…	56	-32
6	8fc207e94fa91a7649c5a5dab…	c69f8b33e62ecb30ff78ae46d7…	54	-32
7	f31535f21d145b2345e2bf7f09…	ed532487db04478dfba09d112…	81	-49
8	06ae7271902bbb087fc093137…	1a1b5f9e903aa3c203caf5cb6…	31	1
9	4906eeadde5f70b308c20c4a8…	4e7656e34357b93f14b40c640…	32	-6
10	bca3dc20a3ec02261c5b17dc2…	90065cd5c7c581bf099b1b9bd…	30	0
11	f11e9516ca2b6091b64f2e2ea…	7d527c98f408420a9d5c953e5…	34	-9
12	38c1e3d4ed6a13cd0cf612d4c…	18c934f4cdc994cd04eb13bce…	0	16
13	da8be3bb62e9bf01e2e1a3bfd…	3f083b9f62e687be8d84684c7…	31	-4
14	690199d6a2c51ff57c6b392d7…	19bacb562bd43bd4eaf05b6c0…	59	-33

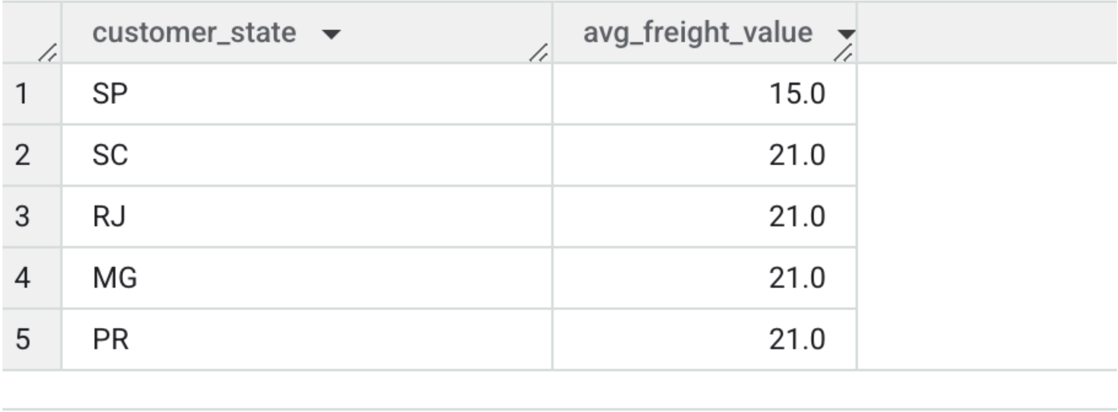
2.Find out the top 5 states with the highest & lowest average freight value.

```
with afv as (

SELECT c.customer_state,
round(avg(oi.freight_value))as avg_freight_value
FROM `alert-outlet-348511.target.order_items`  oi join
alert-outlet-348511.target.orders o on oi.order_id=o.order_id
join alert-outlet-348511.target.customers c on c.customer_id= o.customer_id
group by 1)

-- top5 states
select customer_state, avg_freight_value from afv order by avg_freight_value desc
limit 5
```



Row	customer_state	avg_freight_value
1	RR	43.0
2	PB	43.0
3	RO	41.0
4	AC	40.0
5	PI	39.0

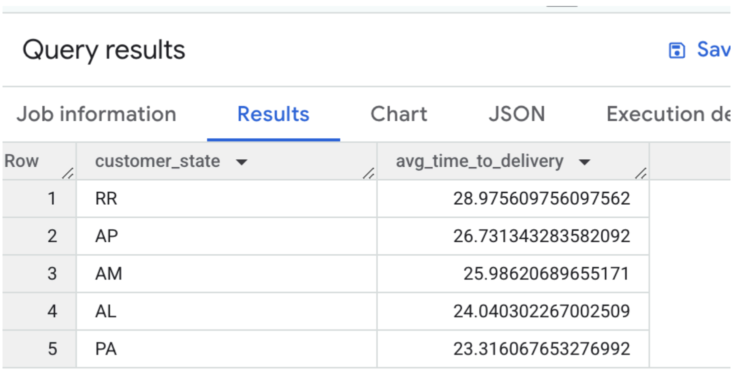
```
-- least 5 states
select customer_state, avg_freight_value from afv order by avg_freight_value  limit 5
```



	customer_state	avg_freight_value
1	SP	15.0
2	SC	21.0
3	RJ	21.0
4	MG	21.0
5	PR	21.0

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

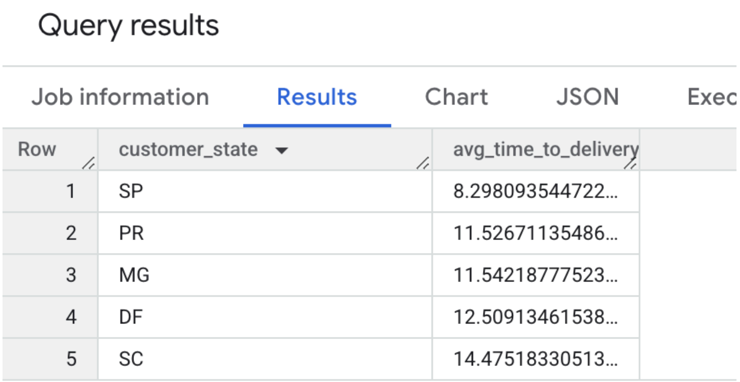
```
with delivery_time as
(SELECT c.customer_state,
avg(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day)) AS
avg_time_to_delivery,
from
alert-outlet-348511.target.orders o
join alert-outlet-348511.target.customers c on c.customer_id= o.customer_id
where order_status='delivered'
group by 1
)
select * from delivery_time order by avg_time_to_delivery desc limit 5
```



Row	customer_state	avg_time_to_delivery
1	RR	28.975609756097562
2	AP	26.731343283582092
3	AM	25.98620689655171
4	AL	24.040302267002509
5	PA	23.316067653276992

—lowest delivery time

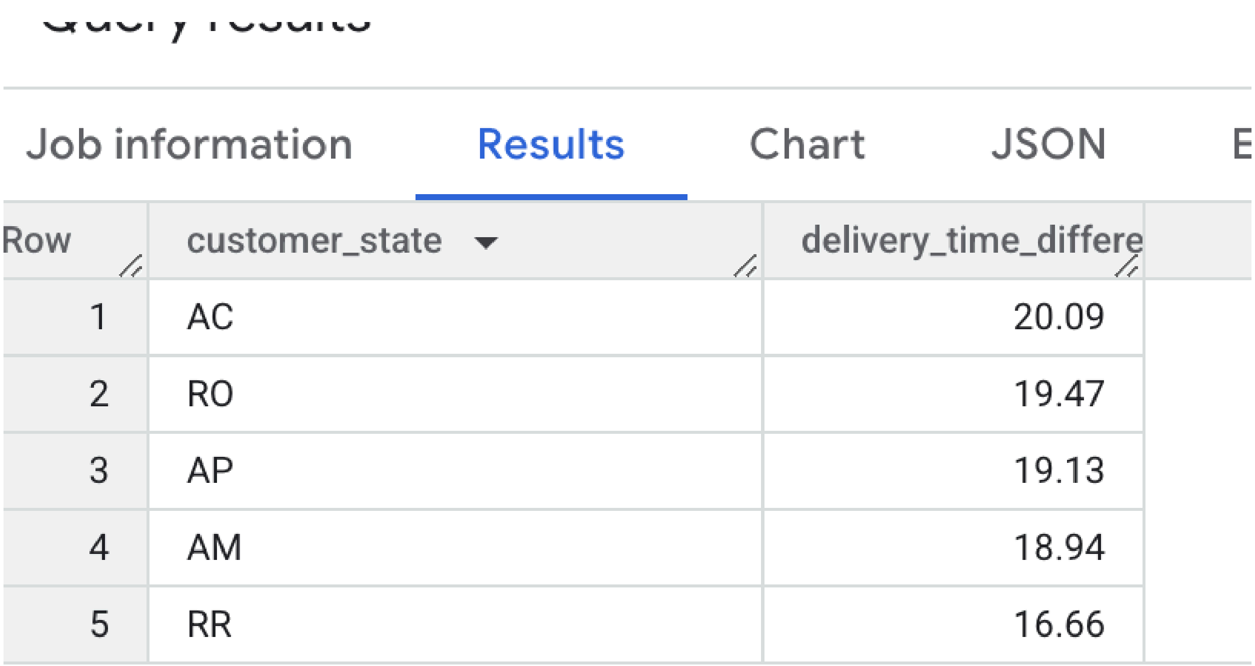
```
select * from delivery_time order by avg_time_to_delivery  limit 5
```



Row	customer_state	avg_time_to_delivery
1	SP	8.298093544722...
2	PR	11.52671135486...
3	MG	11.54218777523...
4	DF	12.50913461538...
5	SC	14.47518330513...

4.Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
SELECT
  customer_state,
  ROUND(AVG(estimated_delivery_time - actual_delivery_time), 2) AS
delivery_time_difference
FROM (
  SELECT
    o.order_id,
    c.customer_state,
    TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
actual_delivery_time,
    TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, day) AS
estimated_delivery_time
  FROM
    `alert-outlet-348511.target.orders` o
  JOIN
    `alert-outlet-348511.target.customers` c
  USING
    (customer_id)
  WHERE
    order_status = "delivered") t1
GROUP BY
  customer_state
ORDER BY
  delivery_time_difference DESC
LIMIT 5;
```

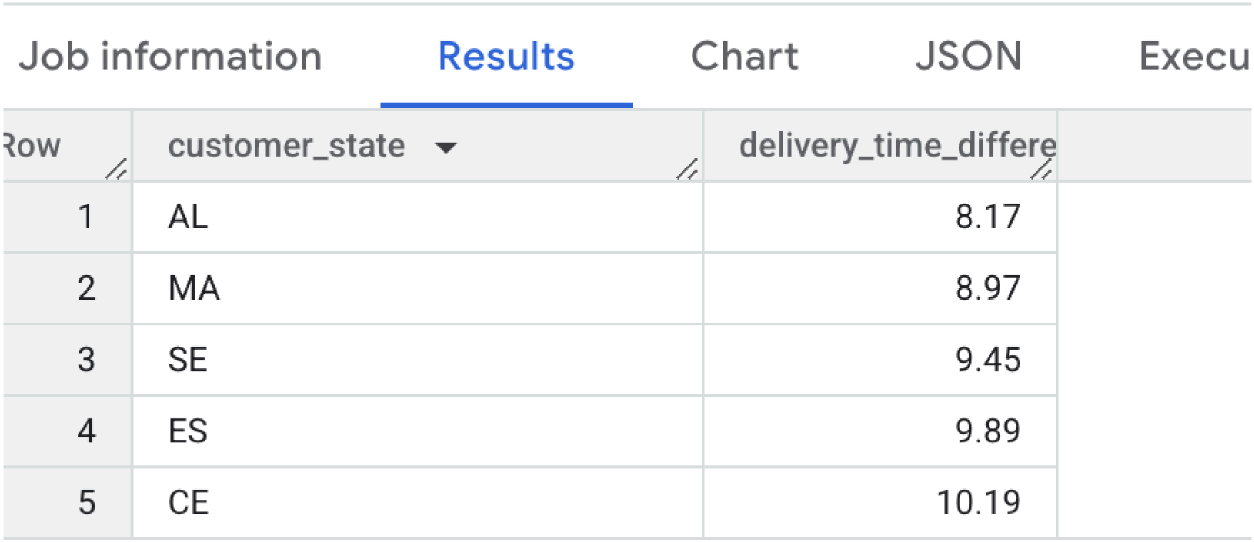


Row	customer_state	delivery_time_differe
1	AC	20.09
2	RO	19.47
3	AP	19.13
4	AM	18.94
5	RR	16.66

States (AC,RO,AP,AM,RR) are faster delivery states compared to estimated time of delivery.

states with late delivery compared to estimated date

```
SELECT
  customer_state,
  ROUND(AVG(estimated_delivery_time - actual_delivery_time), 2) AS
delivery_time_difference
FROM (
  SELECT
    o.order_id,
    c.customer_state,
    TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
actual_delivery_time,
    TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, day) AS
estimated_delivery_time
  FROM
    `alert-outlet-348511.target.orders` o
  JOIN
    `alert-outlet-348511.target.customers` c
  USING
    (customer_id)
  WHERE
    order_status = "delivered") t1
GROUP BY
  customer_state
ORDER BY
  Delivery_time_difference
LIMIT 5;
```



Row	customer_state	delivery_time_differe
1	AL	8.17
2	MA	8.97
3	SE	9.45
4	ES	9.89
5	CE	10.19

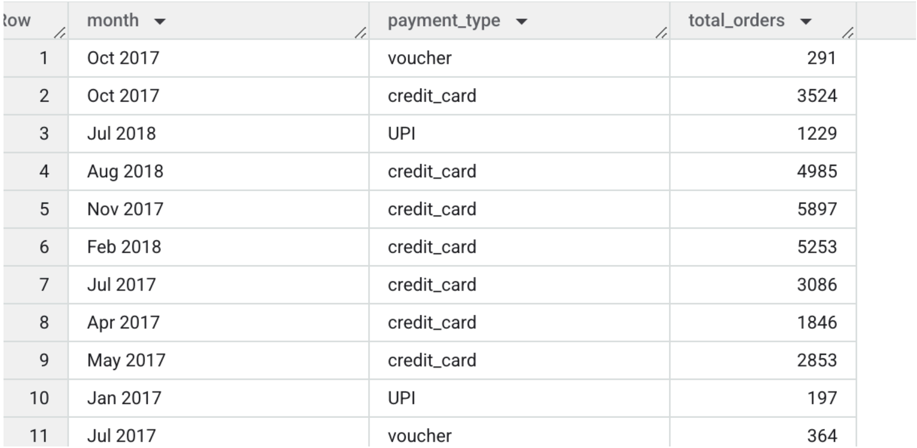
States (AL,MA,SE,ES,CE) are least faster delivery states compared to estimated time of delivery.

Customers ordering in this region will be a bit disappointed for the service if any of the Target competitor focus on this issue and if they address this issue possibility of Target will lose its customer base in this region so need to take care of this Regions.

6.Analysis based on the payments:

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

```
SELECT
  month,
  payment_type,
  COUNT(*) AS total_orders
FROM (
  SELECT
    p.order_id,
    p.payment_type,
    FORMAT_DATE('%b %Y', DATE(ORDER_PURCHASE_TIMESTAMP)) AS month
  FROM
    `alert-outlet-348511.target.payments` p
  JOIN
    `alert-outlet-348511.target.orders` o
  on p.order_id= o.order_id
 ) t1
GROUP BY
1,2
```

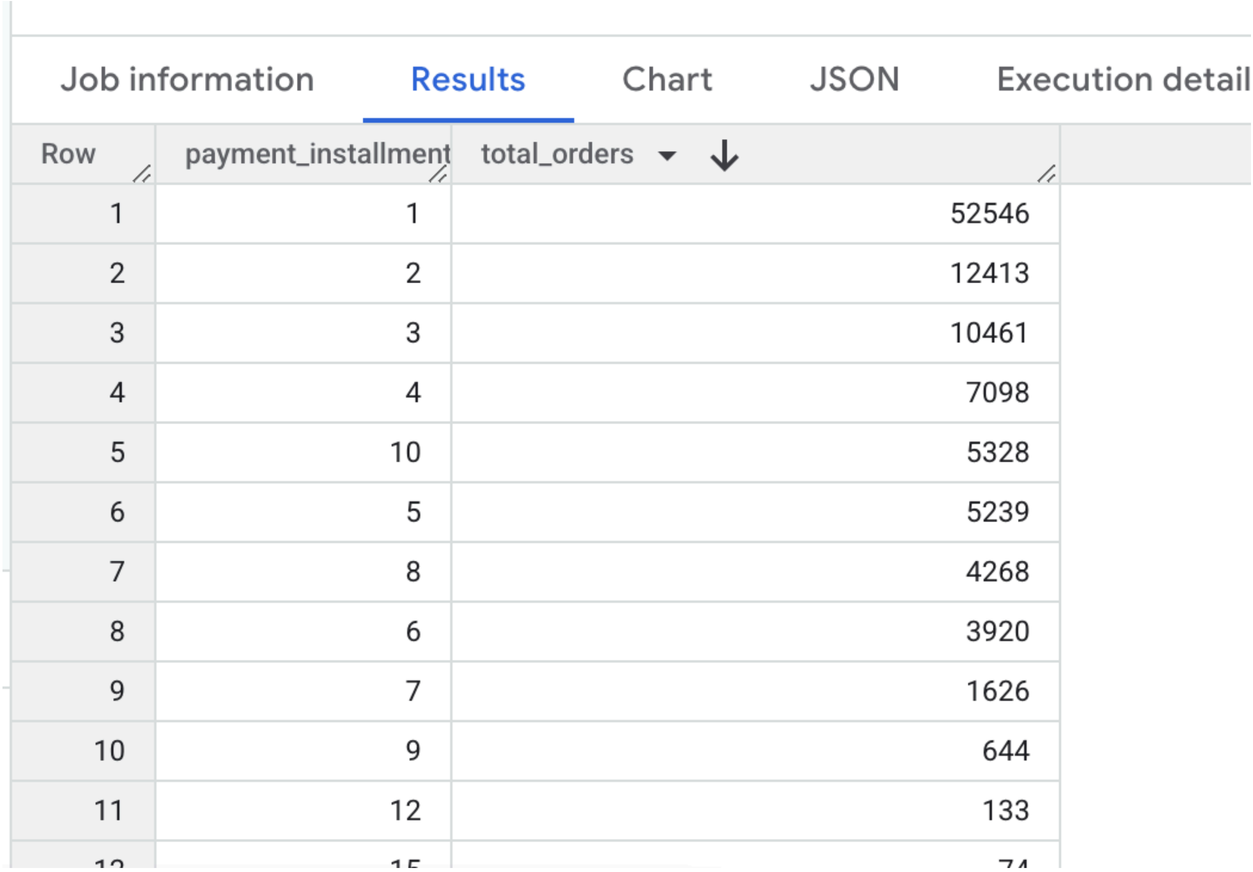


Row	month	payment_type	total_orders
1	Oct 2017	voucher	291
2	Oct 2017	credit_card	3524
3	Jul 2018	UPI	1229
4	Aug 2018	credit_card	4985
5	Nov 2017	credit_card	5897
6	Feb 2018	credit_card	5253
7	Jul 2017	credit_card	3086
8	Apr 2017	credit_card	1846
9	May 2017	credit_card	2853
10	Jan 2017	UPI	197
11	Jul 2017	voucher	364

Most used tender type is Credit card

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

```
SELECT
  payment_installments,
  COUNT(*) AS total_orders
FROM
  `alert-outlet-348511.target.payments`
GROUP BY
  payment_installments;
```



Row	payment_installment	total_orders
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	10	5328
6	5	5239
7	8	4268
8	6	3920
9	7	1626
10	9	644
11	12	133
12	15	74

One installment has the highest orders (52546) Meaning more than half of the orders are not going for regular EMI’s