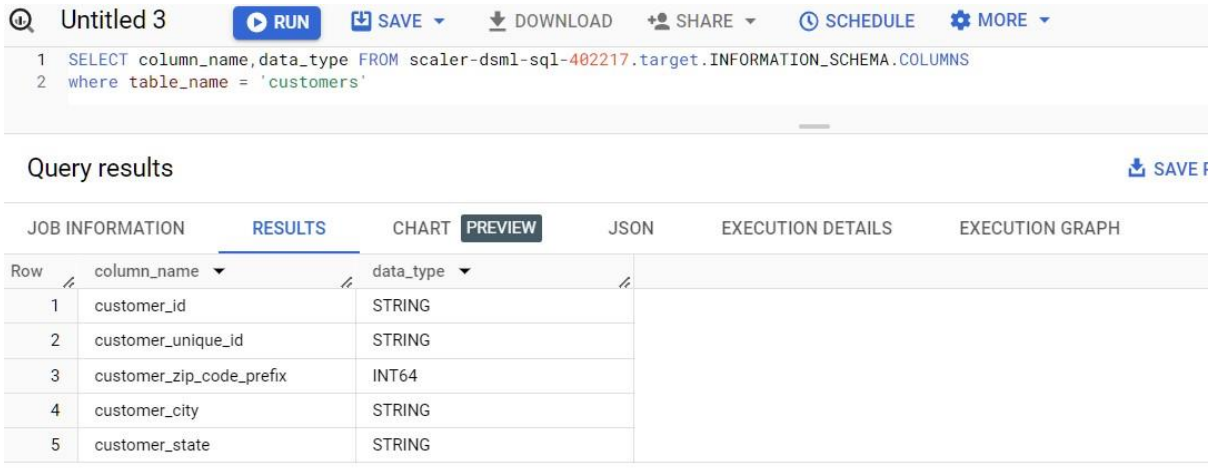
Q1 A. /* Data type of all columns in the “customers” table */

SELECT column_name, data_type FROM scale-dsml-sql-402217.target.INFORMATION_SCHEMA.COLUMNS where table_name = ‘customers’

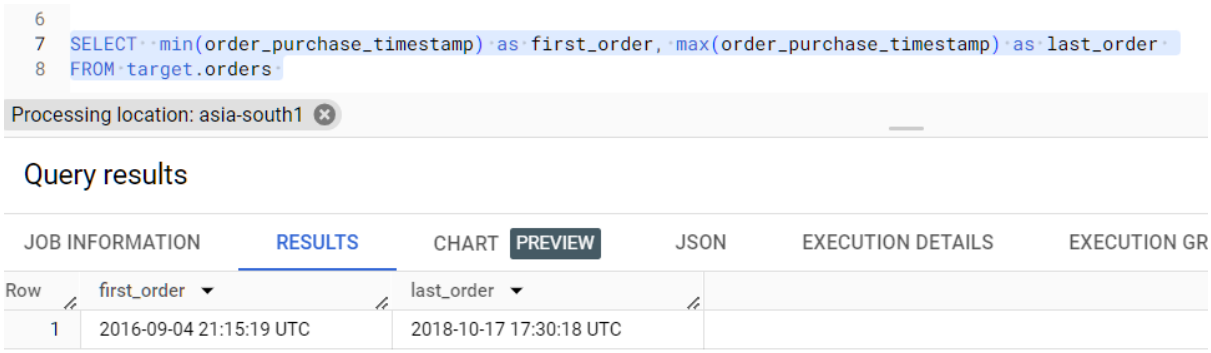


INSIGHT:- The query aim to retrieve details about the columns in the customer table including their names

---

Q1 B. /* Get the time range between which the orders were placed.*/

```
SELECT  order_status, min(order_purchase_timestamp) as first_order,
max(order_purchase_timestamp) as last_order
   FROM target.orders
```

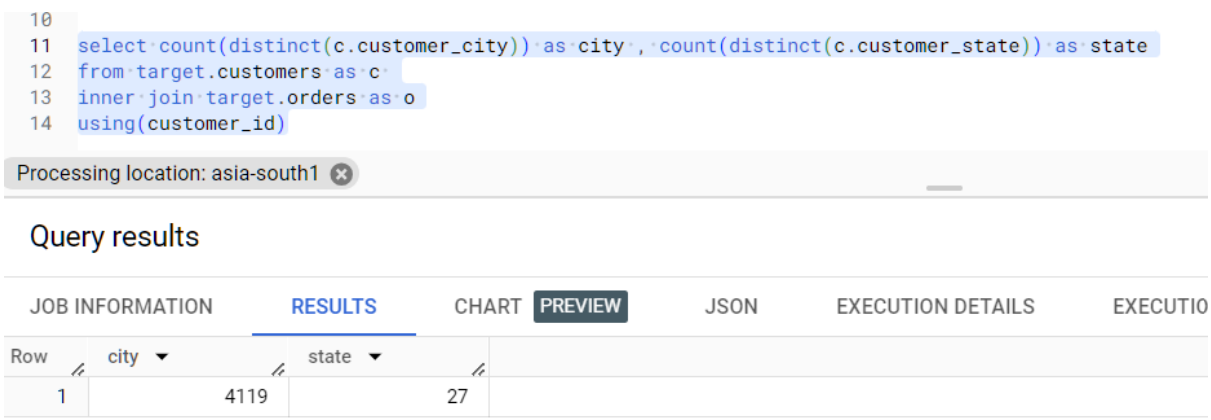


INSIGHT:- By this query we get the time range of first order and last order place in the given period of time

---

Q1 C. Count the Cities & States of customers who ordered during the given period.

```
select count(distinct(c.customer_city)) as city ,
count(distinct(c.customer_state)) as state
   from target.customers as c
   inner join target.orders as o
   using(customer_id)
```



INSIGHT:- From this query we are getting to know that total number of city and state were our customer are distributed in Brazil

Recommendations:- Based on the city and states need to improve logistics services

---

2 A

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

select month,( order_placed - last_month_orders) as no_of_order_increased from
(
select month, count(order_id) as order_placed , lag(count(order_id)) over(order by
month ) as last_month_orders from
(
select order_id, format_date('%Y-%m',order_purchase_timestamp) as month from
target.orders
)
group by month
order by month )
where order_placed >last_month_orders
```

Row	month	no_of_order_increase
1	2016-10	320
2	2017-01	799
3	2017-02	980
4	2017-03	902
5	2017-05	1296
6	2017-07	781
7	2017-08	305
8	2017-10	346
9	2017-11	2913
10	2018-01	1596

INSIGHT Here we are getting to know that no of order has increase on first 5 months

RECOMMENDATION We can improve servers at time

---

2 B

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

select month, count(order_id) as order_placed from
(
select order_id, format_date('%Y-%m',order_purchase_timestamp) as month from
target.orders
)
group by month
order by order_placed desc
```

Query results

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

Row	month	order_placed
1	2017-11	7544
2	2018-01	7269
3	2018-03	7211
4	2018-04	6939
5	2018-05	6873
6	2018-02	6728
7	2018-08	6512
8	2018-07	6292
9	2018-06	6167
10	2017-12	5673

INSIGHT we are getting monthly seasonality order placed

RECOMMENDATION More order are placed at following months

---

2 C

```
/* . 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 */

select day, count(order_id) as count_of_order from(
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'
end as day,
order_id
from target.orders)
group by day
order by count_of_order DESC limit 1
```

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON

Row	day	count_of_order
1	Afternoon	38135

INSIGHT More order are placed afternoon time

RECOMMENDATION we can make app more stable at afternoon time

---

3 A

```
/* Get the month on month no. of orders placed in each state.*/

select c.customer_state,o.month, count(o.count_order) as order_placed from
target.customers as c left join
(select month, customer_id, count(order_id) as count_order from
(select order_id, FORMAT_DATE('%Y-%m', order_purchase_timestamp)as month,
customer_id from target.orders)
group by month,  customer_id) as o
using(customer_id)
group by c.customer_state, o.month
order by c.customer_state
```

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

Row	customer_state	month	order_placed
1	AC	2017-11	5
2	AC	2018-04	4
3	AC	2017-02	3
4	AC	2017-04	5
5	AC	2017-06	4
6	AC	2017-08	4
7	AC	2017-05	8
8	AC	2018-03	2
9	AC	2018-02	3
10	AC	2018-01	6
11	AC	2017-07	5
12	AC	2017-10	6

INSIGHT we are getting to know that order placed in each state on month

RECOMMENDATION By the given data we can improve services to states

---

3 B

```
/* How are the customers distributed across all the states? */

select geolocation_state, count(distinct(customer_id)) as count_unique_customers
from
(select c.customer_zip_code_prefix, c.customer_id, g.geolocation_state from
target.customers as c left join
target.geolocation as g on c.customer_zip_code_prefix =
g.geolocation_zip_code_prefix)
group by geolocation_state
order by count_unique_customers desc
```

Row	geolocation_state	count_unique_custor
1	SP	41731
2	RJ	12839
3	MG	11624
4	RS	5473
5	PR	5034
6	SC	3651
7	BA	3371
8	ES	2027
9	GO	2011
10	DF	1974

INSIGHT we are getting to know that states were our customer are located

RECOMMENDATION we need to expand for remaining state

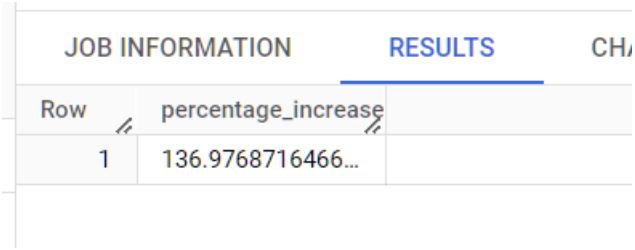
---

4 A

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

SELECT
  ((SUM(CASE WHEN EXTRACT(YEAR from o.order_purchase_timestamp) = 2018 THEN
p.payment_value ELSE 0 END) -
    SUM(CASE WHEN EXTRACT(YEAR from o.order_purchase_timestamp) = 2017 THEN
p.payment_value ELSE 0 END)) /
```

```
   SUM(CASE WHEN EXTRACT(YEAR from o.order_purchase_timestamp) = 2017 THEN
p.payment_value ELSE 0 END)) * 100 AS percentage_increase
FROM
  target.orders as o
  inner join target.payment as p using(order_id)
WHERE
  EXTRACT(YEAR from o.order_purchase_timestamp) IN (2017, 2018) AND
  EXTRACT(MONTH from o.order_purchase_timestamp) BETWEEN 1 AND 8;
```



JOB INFORMATION | RESULTS | CHA

Row	percentage_increase
1	136.9768716466...

INSIGHT we getting to know that % increase in the year between 2017 to 2018

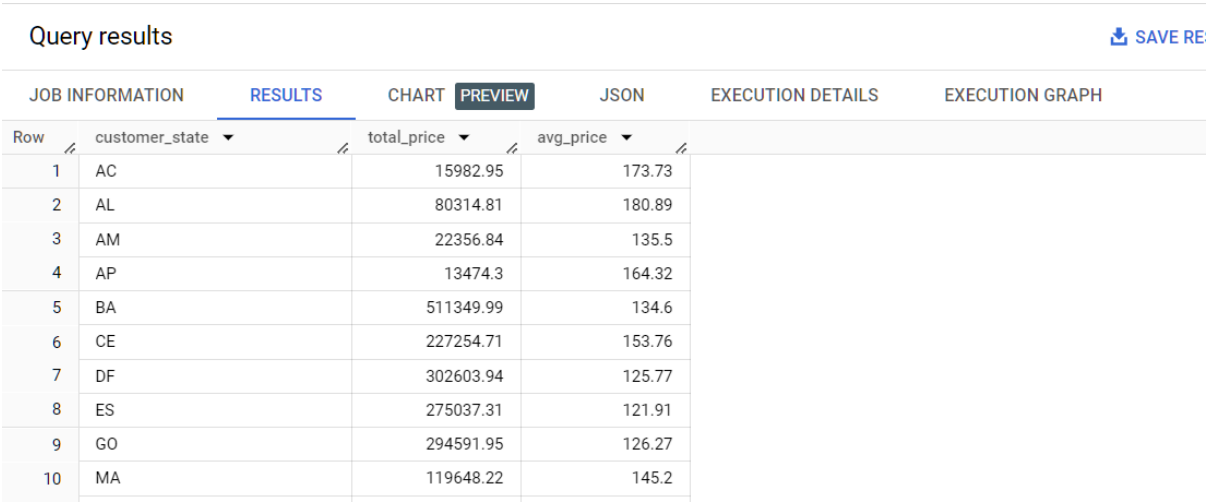
RECOMMENDATION By these we are well performed well in 2018

---

4 B

```
/*Calculate the Total & Average value of order price for each state.*/

select p.customer_state, round(sum(ot.price),2) as
total_price,round(avg(ot.price),2) as avg_price from
(select o.order_id,o.price, oo.customer_id from target.order_items as o inner join
target.orders as oo using(order_id)) as ot
inner join
(SELECT customer_id, customer_state FROM target.customers) as p
using(customer_id)
group by p.customer_state
order by  p.customer_state
```



Query results

SAVE RE

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

Row	customer_state	total_price	avg_price
1	AC	15982.95	173.73
2	AL	80314.81	180.89
3	AM	22356.84	135.5
4	AP	13474.3	164.32
5	BA	511349.99	134.6
6	CE	227254.71	153.76
7	DF	302603.94	125.77
8	ES	275037.31	121.91
9	GO	294591.95	126.27
10	MA	119648.22	145.2

INSIGHT we are getting to know that sum and avg price

RECOMMENDATION we need to improve the least values

---

4 C

```
/*  Calculate the Total & Average value of order freight for each state. */

select p.customer_state, round(sum(ot.freight_value),2) as
sum_freight_value,round(avg(ot.freight_value),2) as avg_freight_value from
(select o.order_id,o.freight_value, oo.customer_id from target.order_items as o
inner join target.orders as oo using(order_id)) as ot
inner join
(SELECT customer_id, customer_state FROM target.customers) as p
using(customer_id)
group by p.customer_state
order by  p.customer_state
```

Query results

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

Row	customer_state	sum_freight_value	avg_freight_value
1	AC	3686.75	40.07
2	AL	15914.59	35.84
3	AM	5478.89	33.21
4	AP	2788.5	34.01
5	BA	100156.68	26.36
6	CE	48351.59	32.71
7	DF	50625.5	21.04
8	ES	49764.6	22.06
9	GO	53114.98	22.77
10	MA	31523.77	38.26

INSIGHT we are getting to know that sum and avg freight

RECOMMENDATION we need to improve the least values

---

5 A

```
/*A. 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.
Do this in a single query.*/
```

```sql
select order_id,
timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,DAY) as
time_to_deliver,
timestamp_diff(order_estimated_delivery_date,order_delivered_customer_date, DAY) as
diff_estimated_delivery from target.orders
order by time_to_deliver desc
```

Row	order_id	time_to_deliver	diff_estimated_delive
1	ca07593549f1816d26a572e06...	209	-181
2	1b3190b2dfa9d789e1f14c05b...	208	-188
3	440d0d17af552815d15a9e41a...	195	-165
4	0f4519c5f1c541ddec9f21b3bd...	194	-161
5	285ab9426d6982034523a855f...	194	-166
6	2fb597c2f772eca01b1f5c561b...	194	-155
7	47b40429ed8cce3aee9199792...	191	-175
8	2fe324febf907e3ea3f2aa9650...	189	-167
9	2d7561026d542c8dbd8f0daea...	188	-159
10	437222e3fd1b07396f1d9ba8c...	187	-144

INSIGHT Here we got time to deliver and difference where negative values indicates faster delivery time

RECOMMENDATION By data we need get know time to deliver

---

5 B

```sql
/*Find out the top 5 states with the highest & lowest average freight value.*/

select customer_state, avg_freight from
(select p.customer_state, round(avg(ot.freight_value),2) as avg_freight from
(select o.order_id,o.freight_value, oo.customer_id from target.order_items as o
inner join target.orders as oo using(order_id)) as ot
inner join
(SELECT customer_id, customer_state FROM target.customers) as p
using(customer_id)
group by p.customer_state
order by avg_freight desc limit 5)
union all
select customer_state, avg_freight from
(select p.customer_state, round(avg(ot.freight_value),2) as avg_freight from
(select o.order_id,o.freight_value, oo.customer_id from target.order_items as o
inner join target.orders as oo using(order_id)) as ot
inner join
(SELECT customer_id, customer_state FROM target.customers) as p
using(customer_id)
group by p.customer_state
order by avg_freight limit 5)
order by avg_freight desc
```

JOB INFORMATION | RESULTS | CHART | PREVIEW | JS

Row	customer_state	avg_freight
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15
6	DF	21.04
7	RJ	20.96
8	MG	20.63
9	PR	20.53
10	SP	15.15

INSIGHT here we got top 5 states with top and least freight time

RECOMMENDATION we need to improve freight time for the least states

---

```
5 C
/* . Find out the top 5 states with the highest & lowest average delivery time.*/
select customer_state, avg_time from
(SELECT p.customer_state,round(AVG(o.time_to_deliver),2) AS avg_time FROM
(SELECT customer_id,TIMESTAMP_DIFF(
TIMESTAMP(order_delivered_customer_date),TIMESTAMP(order_purchase_timestamp),DAY)
AS time_to_deliver FROM target.orders) AS o
INNER JOIN
(SELECT customer_id, customer_state FROM target.customers) AS p USING(customer_id)
GROUP BY p.customer_state
order by avg_time desc limit 5) as t1
union all
select customer_state, avg_time from
(SELECT p.customer_state,round(AVG(o.time_to_deliver),2) AS avg_time FROM
(SELECT customer_id,TIMESTAMP_DIFF(
TIMESTAMP(order_delivered_customer_date),TIMESTAMP(order_purchase_timestamp),DAY)
AS time_to_deliver FROM target.orders) AS o
INNER JOIN
(SELECT customer_id, customer_state FROM target.customers) AS p USING(customer_id)
GROUP BY p.customer_state
order by avg_time  limit 5) as t2
order by avg_time desc
```

Query results

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXI

Row	customer_state	avg_time
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32
6	SC	14.48
7	DF	12.51
8	MG	11.54
9	PR	11.53
10	SP	8.3

INSIGHT here we got top 5 states with faster delivery and least delivery time

RECOMMENDATION we need to improve delivery time for the least states

---

5 D

```
/* 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. */

select customer_state from
(select c.customer_state, round(avg(o.days_difference),2) as avg_day from
(select customer_id,
order_status,order_delivered_customer_date,order_estimated_delivery_date,
timestamp_DIFF(timestamp(order_estimated_delivery_date),timestamp(order_delivered_c
ustomer_date), DAY) AS days_difference from target.orders
where order_status = 'delivered'
order by days_difference desc) as o inner join
(select customer_id, customer_state from target.customers) as c using(customer_id)
group by c.customer_state
order by avg_day desc, customer_state limit 5) as t
```

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXECUTION

Row	customer_state
1	AC
2	RO
3	AP
4	AM
5	RR

INSIGHT By these query we get know that top 5 states were delivery

RECOMMENDATION need to improve the delivery support for other states also

---

6 A

```
./* Find the month on month no. of orders placed using different payment types.*/

select p.payment_type, o.year_month, count(o.order_id) as order_place
from (select payment_type, order_id from target.payment
group by payment_type, order_id) as p
inner join
(SELECT
  order_id,
  FORMAT_DATE('%Y-%m', order_purchase_timestamp) AS year_month
FROM
  target.orders) as o
on o.order_id = p.order_id
group by p.payment_type, o.year_month
```

Query results

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

Row	payment_type	year_month	order_place
1	credit_card	2016-09	3
2	UPI	2016-10	63
3	credit_card	2016-10	253
4	debit_card	2016-10	2
5	voucher	2016-10	11
6	credit_card	2016-12	1
7	UPI	2017-01	197
8	credit_card	2017-01	582
9	debit_card	2017-01	9
10	voucher	2017-01	33

INSIGHT By this query we are getting know how custom using the different payment gate

RECOMMENDATION By give more offer on card we can gain more user

---

6.B

```
/*Find the no. of orders placed on the basis of the payment installments that have
been paid.*/

select count(*) from (
select p.order_id, p.payment_installments from target.payment as p inner join
`target.orders` as o
on o.order_id = p.order_id
where p.payment_installments > 1) as t
```

```
/*Find the no. of orders placed on the basis of the payment installments that have
been paid.
Hint: We want you to count the no. of orders placed based on the no. of
payment installments where at least one installment has been successfully
paid.*/

select count(*) from (
select p.order_id, p.payment_installments from target.payment as p inner join `target.orders` as o
on o.order_id = p.order_id
where p.payment_installments > 1) as t
```

Query results

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

Row	f0_
1	51338

INSIGHT:- From the above query we are getting to know that count of payment greater than one

RECOMMENDATION : we can get more profit we order are made in emi