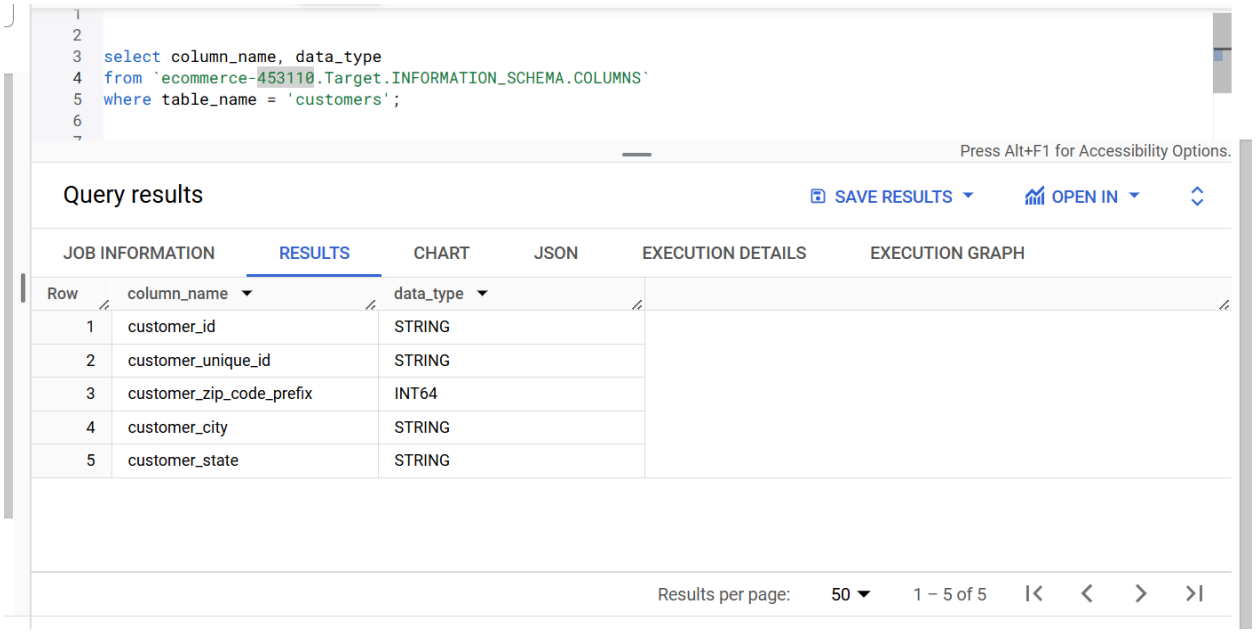
Target

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

```
select column_name, data_type
from `ecommerce-453110.Target.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'customers';
```



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

```
select min(order_purchase_timestamp) as first_order,
      max(order_purchase_timestamp) as last_order
from `Target.orders`
```

Untitled query RUN SAVE DOWNLOAD SHARE This query will process 776.88 KB when ru...

```
select min(order_purchase_timestamp) as first_order,
       max(order_purchase_timestamp) as last_order
from `Target.orders`
```

Press Alt+F1 for Accessibility Options.

Query results SAVE RESULTS OPEN IN

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

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

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

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

Untitled query RUN SAVE DOWNLOAD SHARE This query will process 1.55 MB when run.

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

Press Alt+F1 for Accessibility Options.

Query results SAVE RESULTS OPEN IN

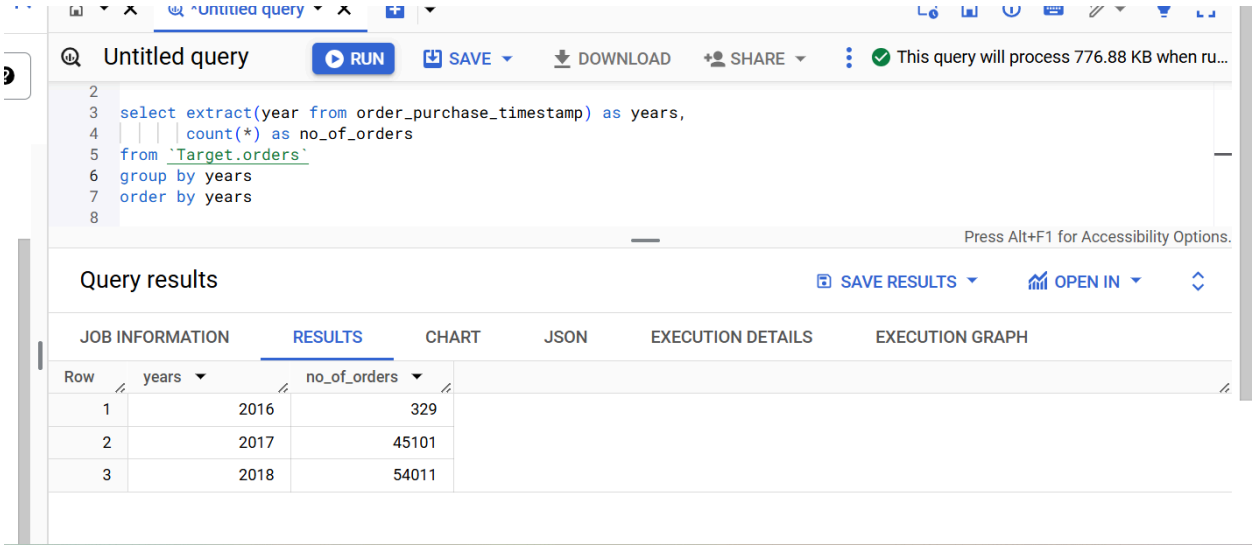
JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	city_count	state_count
1	4119	27

In-depth Exploration:

Q2.A) Is there a growing trend in the no. of orders placed over the past years?

```
select extract(year from order_purchase_timestamp) as years,
        count(*) as no_of_orders
from `Target.orders`
group by years
order by years
```



- **Insights : Yes there is a number of increase in orders over years.**

Q2. B) 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 months,
      count(*) as no_of_orders
from `Target.orders`
group by months
order by no_of_orders desc
```

Untitled query

RUN SAVE DOWNLOAD SHARE This query will process 776.88 KB when ru...

```
select extract(month from order_purchase_timestamp) as months,
       count(*) as no_of_orders
from `Target.orders`
group by months
order by no_of_orders desc
```

Press Alt+F1 for Accessibility Options.

Query results

SAVE RESULTS OPEN IN

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

Row	months	no_of_orders
1	8	10843
2	5	10573
3	7	10318
4	3	9893
5	6	9412
6	4	9343
7	2	8508

Results per page: 50 1 – 12 of 12

- **Insights : Yes no. of orders placed during August, May and July are at peak.**

Q2.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

```
with tab as
(select customer_id,
      extract(hour from order_purchase_timestamp) as hours,
from `Target.orders`)

select count(*) as order_count,
        case when hours between 0 and 6 then 'Dawn'
             when hours between 7 and 12 then 'Morning'
             when hours between 13 and 18 then 'Afternoon'
             when hours between 19 and 23 then 'Night'
```

```
    end as time_of_day
from tab
group by time_of_day
order by order_count desc
```

Query results

SAVE RESULTS OPEN IN

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

Row	order_count	time_of_day
1	38135	Afternoon
2	28331	Night
3	27733	Morning
4	5242	Dawn

Results per page: 50 1 – 4 of 4

- **Insights : The Brazilian customers usually order the most during Afternoon, then Night, Morning and very few orders placed during Dawn.**

Evolution of E-commerce orders in the Brazil region:

Q3.A) Get the month on month no. of orders placed in each state.

```
select c.customer_state,
    extract(month from order_purchase_timestamp) as month,
    count(order_id) as order_count
from `Target.customers` c
inner join `Target.orders` o
on c.customer_id = o.customer_id
group by 1, 2
order by order_count desc, month
```

```
select c.customer_state,
        extract(month from order_purchase_timestamp) as month,
        count(order_id) as order_count
from `Target.customers` c
inner join `Target.orders` o
on c.customer_id = o.customer_id
group by 1, 2
order by month
```

Press Alt+F1 for Accessibility Options.

Query results

SAVE RESULTS OPEN IN

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

Row	customer_state	month	order_count
1	SP	8	4982
2	SP	5	4632
3	SP	7	4381
4	SP	6	4104
5	SP	3	4047

Results per page: 50 1 – 50 of 322

- **Insights : The number of orders placed in SP state during August month is the highest order count (4982).**

Q3.B) How are the customers distributed across all the states?

```
select customer_state,
    count(distinct customer_id) as unique_customers
from `Target.customers`
group by customer_state
order by unique_customers desc
```

```google_sql
select customer_state,
        count(distinct customer_id) as unique_customers
from `Target.customers`
group by customer_state
order by unique_customers desc
```

Press Alt+F1 for Accessibility Options.

Query results

SAVE RESULTS OPEN IN

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRAPH

Row	customer_state	unique_customers
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637

Results per page: 50 1 – 27 of 27

- **Insights : The majority of customers are present in SP state.**

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

Q4.A) Get the % increase in the cost of orders from year 2017 to 2018 *(include months between Jan to Aug only).*

```google_sql
with tab2017 as

(select sum(payment_value) as payment1,

        extract(year from order_purchase_timestamp) as Year,

from `Target.payments` p

inner join `Target.orders` o

on p.order_id = o.order_id

where extract(year from order_purchase_timestamp) = 2017

and extract(month from order_purchase_timestamp) between 1 and 8
```

```
group by 2),

tab2018 as
(select sum(payment_value) as payment2,
        extract(year from order_purchase_timestamp) as Year
from `Target.payments` p
inner join `Target.orders` o
on p.order_id = o.order_id
where extract(year from order_purchase_timestamp) = 2018
and extract(month from order_purchase_timestamp) between 1 and 8
group by 2)

select round(((payment2-payment1)/payment1)*100, 2) as percentage_increase
from tab2017
cross join tab2018
```

Query results

SAVE RESULTS OPEN IN

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

Row	percentage_increase
1	136.98

Results per page: 50 1 – 1 of 1

- **Insights : The % increase in the cost of orders from year 2017 to 2018 *(include months between Jan to Aug only)* is 136.98%**

Q4.B) Calculate the Total & Average value of order price for each state.

```
with cte as
(select customer_state as state,
      sum(price) as Total_price,
      count(distinct o.order_id) as order_count
from `Target.customers` c
left join `Target.orders` o
on c.customer_id = o.customer_id

left join `Target.order_items` i
on o.order_id = i.order_id
group by customer_state)

select state, Total_price,
      (Total_price/order_count) as avg_price
from cte
```

Press Alt+F1 for Accessibility Options.

Query results

SAVE RESULTS OPEN IN

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRAPH

Row	state	Total_price	avg_price
19	RJ	1824092.669999...	141.9306465919...
20	RN	83034.97999999...	171.2061443298...
21	RO	46140.64000000...	182.3740711462...
22	RR	7829.429999999...	170.2049999999...
23	RS	750304.0200000...	137.2674753018...
24	SC	520553.34000001	143.1271212537...
25	SE	58920.85000000...	168.3452857142...
26	SP	5202955.050001...	124.6336187898...
27	TO	49621.74000000...	177.2205000000...

Results per page: 50 1 – 27 of 27

Q4.C) Calculate the Total & Average value of order freight for each state.

```
with cte as
(select customer_state as state,
```

```
    sum(freight_value) as Total_freight,
    count(distinct o.order_id) as order_count
from `Target.customers` c
left join `Target.orders` o
on c.customer_id = o.customer_id

left join `Target.order_items` i
on o.order_id = i.order_id
group by customer_state)

select state, Total_freight,
    (Total_freight/order_count) as avg_freight
from cte
```

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

Row	state	Total_freight	avg_freight
1	AC	3686.749999999...	45.51543209876...
2	AL	15914.58999999...	38.53411622276...
3	AM	5478.89	37.01952702702...
4	AP	2788.500000000...	41.00735294117...
5	BA	100156.6799999...	29.63215384615...
6	CE	48351.58999999...	36.19130988023...
7	DF	50625.49999999...	23.65677570093...
8	ES	49764.59999999...	24.47840629611...
9	GO	53114.97999999...	26.29454455445...
10	MA	31523.77000000...	42.20049531459...

Results per page: 50 1 – 27 of 27

Analysis based on sales, freight and delivery time.

Q5.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.

```
select 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`
where order_status = 'delivered'
```

Query results

SAVE RESULTS OPEN IN

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRAPH

Row	time_to_deliver	diff_estimated_delive
1	209	-181
2	208	-188
3	195	-165
4	194	-166
5	194	-161
6	194	-155
7	191	-175

Results per page: 50 1 – 50 of 96478

Job history REFRESH

- **Insights : The highest delivery time was 209 days and the highest difference (in days) between the estimated & actual delivery date of an order was -181**

Q5.B) Find out the top 5 states with the highest & lowest average freight value.

```
with state_freight as (
select c.customer_state,
    avg(oi.freight_value) as avg_freight_value
from `Target.customers` c
inner join `Target.orders` o
on c.customer_id = o.customer_id
inner join `Target.order_items` oi
on o.order_id = oi.order_id
group by c.customer_state
),
ranked_states as (
 select customer_state,
      avg_freight_value,
      dense_rank() over (order by avg_freight_value) as rank_asc,
```

```
        dense_rank() over (order by avg_freight_value desc) as rank_desc
from state_freight
)
select customer_state, avg_freight_value
from ranked_states
where rank_asc <= 5 OR rank_desc <= 5
order by avg_freight_value;
```

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

Row	customer_state	avg_freight_value
1	SP	15.14727539041...
2	PR	20.53165156794...
3	MG	20.63016680630...
4	RJ	20.96092393168...
5	DF	21.04135494596...
6	PI	39.14797047970...
7	AC	40.07336956521...
8	RO	41.06971223021...
9	PB	42.72380398671...
10	RR	42.98442307692...

Results per page: 50 1 – 10 of 10

Q5.C) Find out the top 5 states with the highest & lowest average delivery time.

```
with state_delivery_time as (
select c.customer_state,
        round(avg(timestamp_diff(order_delivered_customer_date,
order_purchase_timestamp, Day)),2) as avg_delivery_time
from `Target.customers` c
inner join `Target.orders` o
on c.customer_id = o.customer_id
group by c.customer_state
),
ranked_states as (
 select customer_state,
```

```
        avg_delivery_time,
        dense_rank() over (order by avg_delivery_time) as rank_asc,
        dense_rank() over (order by avg_delivery_time desc) as rank_desc
from state_delivery_time
)
select customer_state, avg_delivery_time
from ranked_states
where rank_asc <= 5 OR rank_desc <= 5
order by avg_delivery_time;
```

Query results

SAVE RESULTS OPEN IN

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRAPH

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

Results per page: 50 1 – 10 of 10

Q5.D) Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
with tab as
(select customer_state,
        avg(timestamp_diff(order_delivered_customer_date, order_purchase_timestamp,
Day)) as avg_actual_delivery,
        avg(timestamp_diff(order_estimated_delivery_date, order_purchase_timestamp,
Day)) as avg_estimate_delivery
from `Target.customers` c
inner join `Target.orders` o
```

```
on c.customer_id = o.customer_id
where order_status = 'delivered'
group by customer_state)

select customer_state,
      round((avg_estimate_delivery - avg_actual_delivery),2) as diff
from tab
order by diff desc
limit 5
```

Query results

SAVE RESULTS OPEN IN

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

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

Results per page: 50 1 – 5 of 5

- **Insights : The AC state is really fast as compared to the estimated date of delivery**

Analysis based on the payments

Q6.A) Find the month on month no. of orders placed using different payment types**.**

```
select payment_type,
      extract(year from order_purchase_timestamp) as Year,
      extract(month from order_purchase_timestamp) as Month,
      count(o.order_id) as order_count
from `Target.orders` o
inner join `Target.payments` p
```

```
on o.order_id = p.order_id

group by 1, 2, 3

order by Year, Month
```

Query results

SAVE RESULTS OPEN IN

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRAPH

Row	payment_type	Year	Month	order_count
1	credit_card	2016	9	3
2	voucher	2016	10	23
3	credit_card	2016	10	254
4	UPI	2016	10	63
5	debit_card	2016	10	2
6	credit_card	2016	12	1

Results per page: 50 1 – 50 of 90

Q6.B) Find the no. of orders placed on the basis of the payment installments that have been paid.

```
select payment_installments,

    count(order_id) as order_count

from `Target.payments`

where payment_installments >= 1

group by payment_installments

order by order_count desc
```

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

Row	payment_installment	order_count
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	10	5328
6	5	5239
7	8	4268
8	6	3920

Results per page: 50 1 – 23 of 23

- **Insights : Payment_installment 1 has the highest order_count.**

Recommendation:

The month of August shows the highest sales in the entire year, that shows we can plan more ideas to increase the sales bar. We can provide them some good offers or we can try mid sale and discounts to increase the sale. We can provide some vouchers or gift cards to the customers who come under the medium range buyers category. We can send additional small gifts to boost the customer buying interest like some freebies.