2023

# Business Case: Target SQL

DEBOJYOTI SEN

# Business Case: Target SQL

## # Actionable Insights :-

# The data is given for the time period between September-2016 to October-2018.

# Clearly shown that, sales trends are growing year after year. As startup in 2016, growth was remarkable in 2016 to 2017. And growth percentage in 2017 to 2018 was good, but as compare to previous its very less.

# Sales trends are peaks in the month of August, May and July respectively.

# Brazilian customer tends to buy mostly in Afternoon time.

# State 'SP','RJ' and 'MG' has most customer across the Brazil.

# Cost of orders remarkably increased in 2018 over 2017.

# In some state delivery was very delayed.

# Count of orders are more in case of less instalment.

## # Recommendations :-

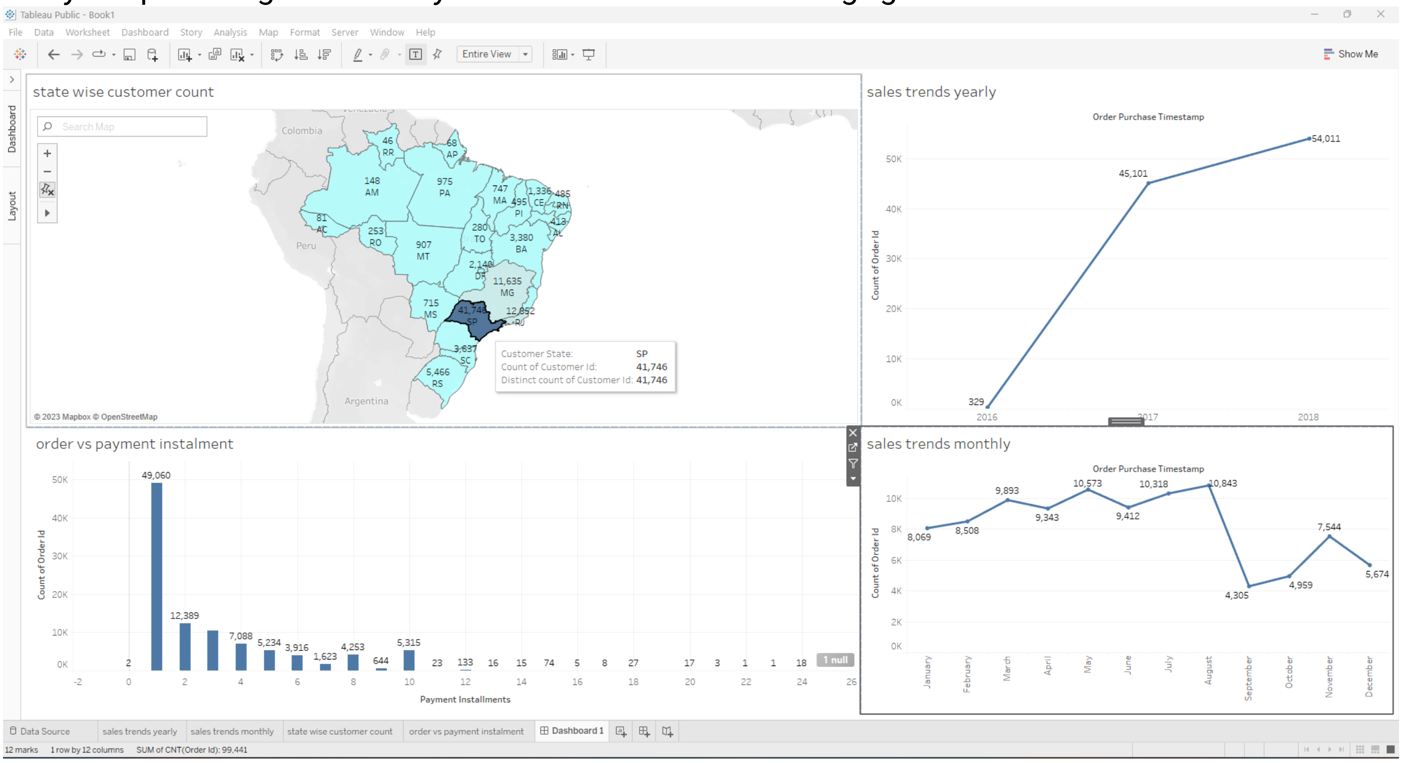
# For continuous growth in business, look after for expanding the territory and follow through good marketing strategies.

# By giving some lucrative offer or discount sale like marketing strategies in the remaining month, sales can be boosted up. And for this peak three month (August, May and July) giving some lucrative offer, sales can be boosted up further.

# By giving "Happy Hour Sale" like marketing strategies in afternoon, sales can be boosted up further.

# By doing various marketing and campaign in the remaining states sales can be boost up.

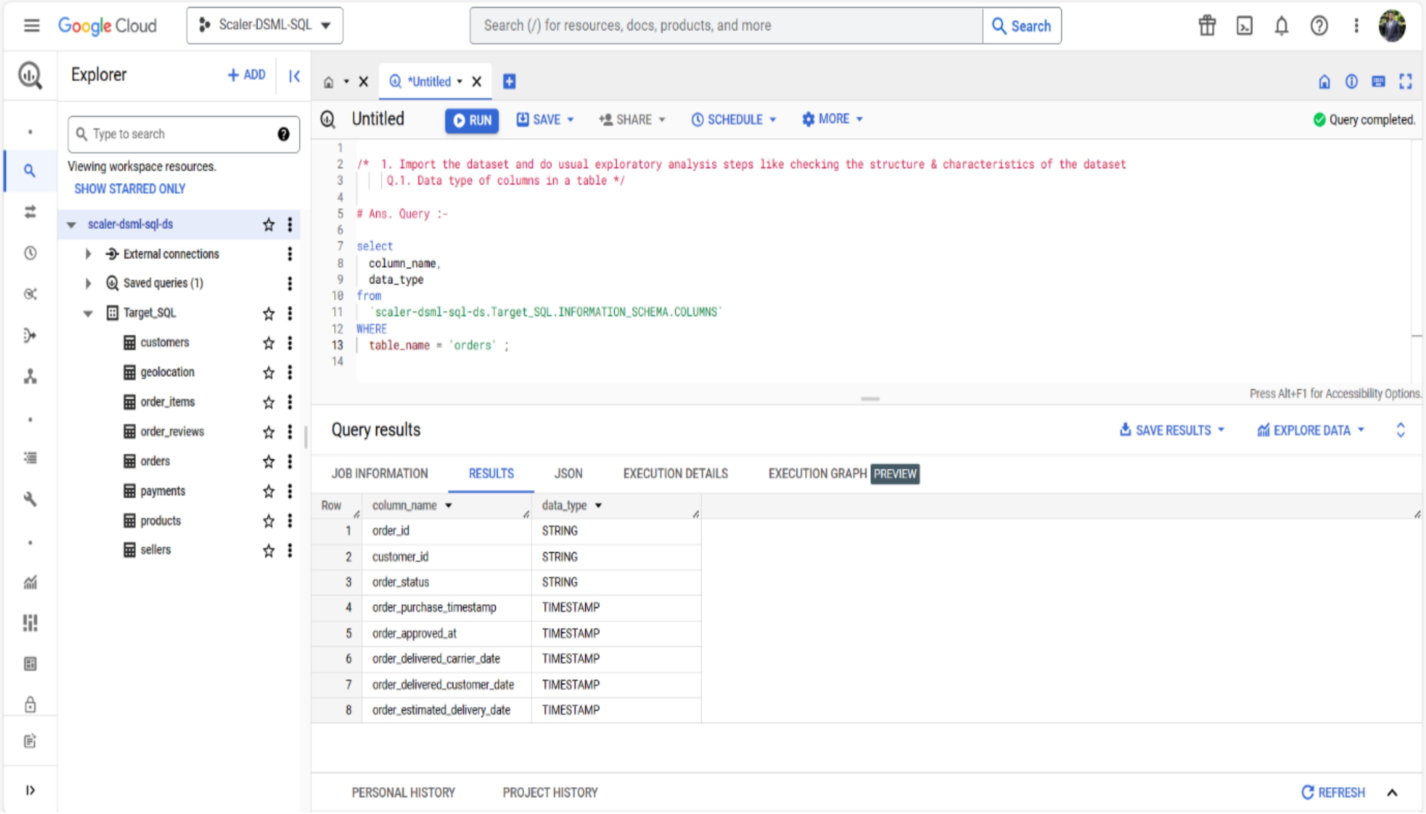
# By improving delivery time can able to engage more customer.



Customer State: SP
Count of Customer Id: 41,746
Distinct count of Customer Id: 41,746

12 marks 1 row by 12 columns SUM of CNT(Order Id): 99,441

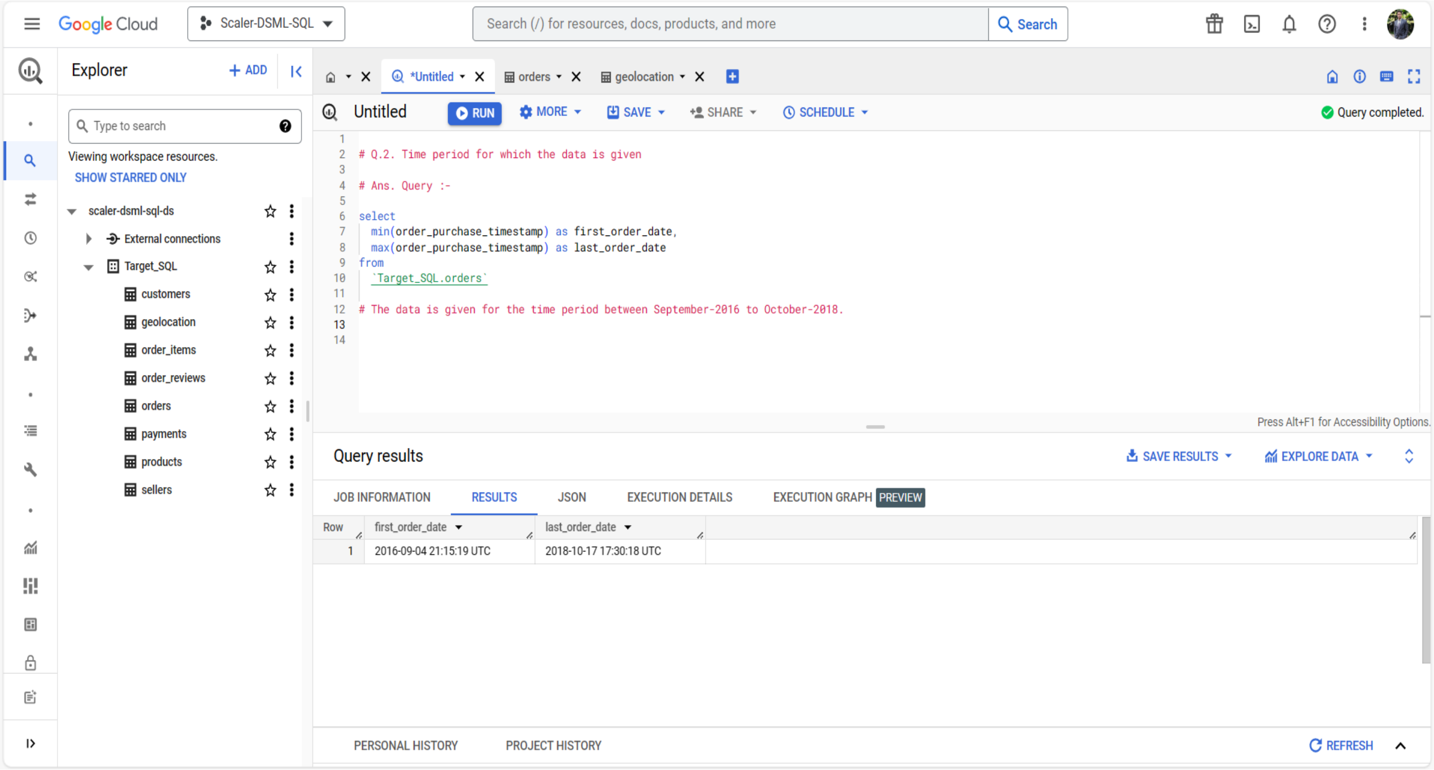
```
*  1. Import the dataset and do usual exploratory analysis steps like
checking the structure & characteristics of the dataset

     Q.1. Data type of columns in a table */

# Ans. Query :-

select
  column_name,
  data_type
from
  `scaler-dsml-sql-ds.Target_SQL.INFORMATION_SCHEMA.COLUMNS`
WHERE
  table_name = 'orders' ;
```



```
/*  1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset
    Q.1. Data type of columns in a table */

# Ans. Query :-

select
  column_name,
  data_type
from
  `scaler-dsml-sql-ds.Target_SQL.INFORMATION_SCHEMA.COLUMNS`
WHERE
  table_name = 'orders' ;
```

| Row | column_name | data_type |
|---|---|---|
| 1 | order_id | STRING |
| 2 | customer_id | STRING |
| 3 | order_status | STRING |
| 4 | order_purchase_timestamp | TIMESTAMP |
| 5 | order_approved_at | TIMESTAMP |
| 6 | order_delivered_carrier_date | TIMESTAMP |
| 7 | order_delivered_customer_date | TIMESTAMP |
| 8 | order_estimated_delivery_date | TIMESTAMP |

```
# Q.2. Time period for which the data is given

# Ans. Query :-

select
  min(order_purchase_timestamp) as first_order_date,
  max(order_purchase_timestamp) as last_order_date
from
  `Target_SQL.orders`
```

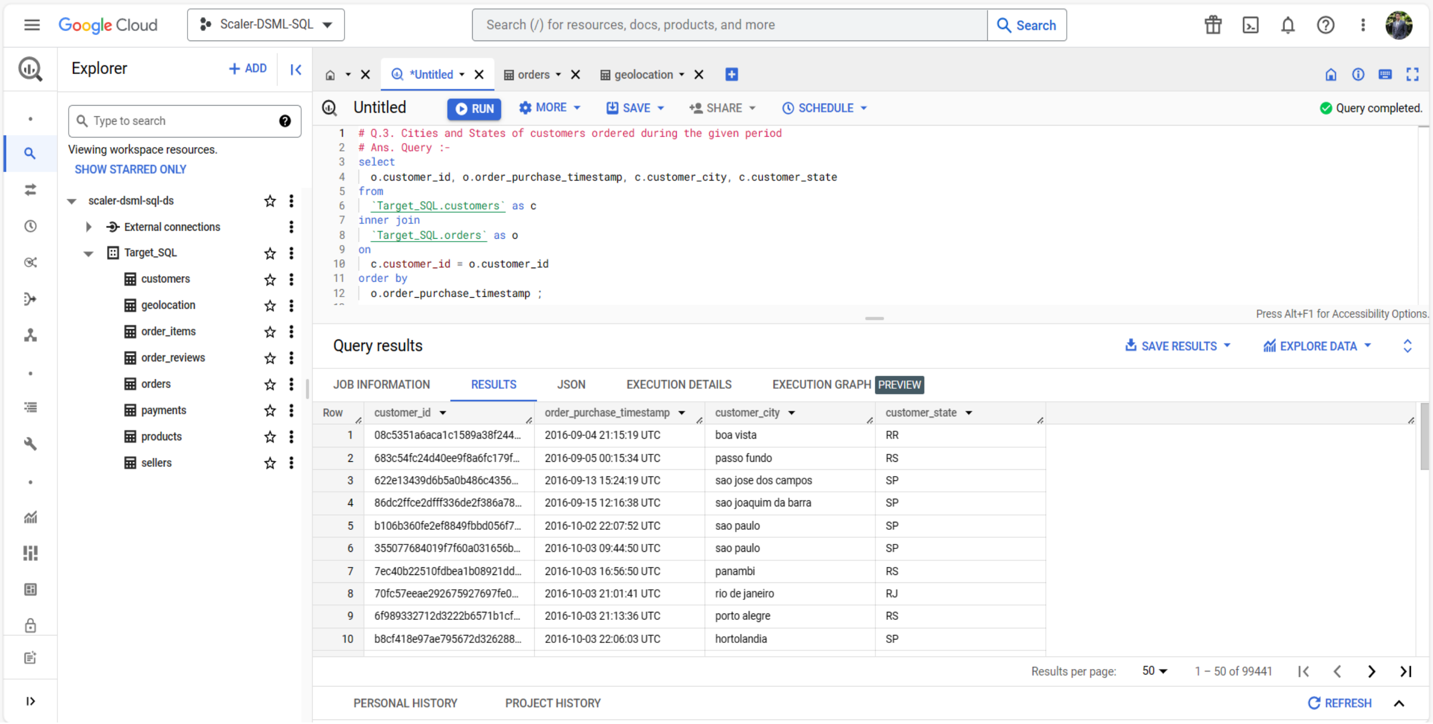


```

# Q.2. Time period for which the data is given

# Ans. Query :-

select
  min(order_purchase_timestamp) as first_order_date,
  max(order_purchase_timestamp) as last_order_date
from
  `Target_SQL.orders`

# The data is given for the time period between September-2016 to October-2018.

```

| Row | first_order_date | last_order_date |
|---|---|---|
| 1 | 2016-09-04 21:15:19 UTC | 2018-10-17 17:30:18 UTC |

# Insights :- The data is given for the time period between September-2016 to October-2018.

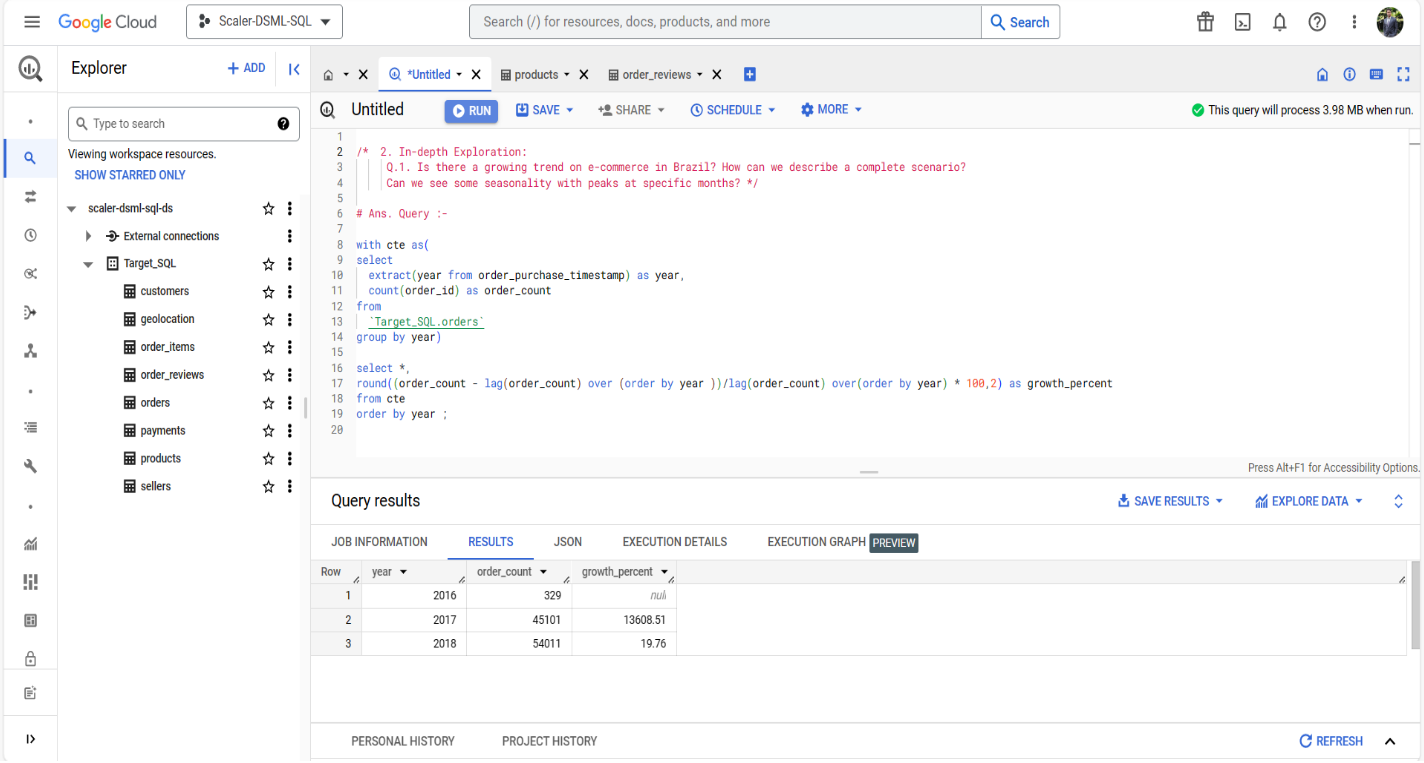
```
# Q.3. Cities and States of customers ordered during the given period

# Ans. Query :-

select
  o.customer_id, o.order_purchase_timestamp, c.customer_city, c.customer_state
from
  `Target_SQL.customers` as c
inner join
  `Target_SQL.orders` as o
on
  c.customer_id = o.customer_id
order by
  o.order_purchase_timestamp ;
```



```
# Q.3. Cities and States of customers ordered during the given period
# Ans. Query :-
select
  o.customer_id, o.order_purchase_timestamp, c.customer_city, c.customer_state
from
  `Target_SQL.customers` as c
inner join
  `Target_SQL.orders` as o
on
  c.customer_id = o.customer_id
order by
  o.order_purchase_timestamp ;
```

| Row | customer_id | order_purchase_timestamp | customer_city | customer_state |
|---|---|---|---|---|
| 1 | 08c5351a6aca1c1589a38f244... | 2016-09-04 21:15:19 UTC | boa vista | RR |
| 2 | 683c54fc24d40ee9f8a6fc179f... | 2016-09-05 00:15:34 UTC | passo fundo | RS |
| 3 | 622e13439d6b5a0b486c4356... | 2016-09-13 15:24:19 UTC | sao jose dos campos | SP |
| 4 | 86dc2ffce2dfff336de2f386a78... | 2016-09-15 12:16:38 UTC | sao joaquim da barra | SP |
| 5 | b106b360fe2ef8849fbbd056f7... | 2016-10-02 22:07:52 UTC | sao paulo | SP |
| 6 | 355077684019f7f60a031656b... | 2016-10-03 09:44:50 UTC | sao paulo | SP |
| 7 | 7ec40b22510fdbea1b08921dd... | 2016-10-03 16:56:50 UTC | panambi | RS |
| 8 | 70fc57eeae292675927697fe0... | 2016-10-03 21:01:41 UTC | rio de janeiro | RJ |
| 9 | 6f989332712d3222b6571b1cf... | 2016-10-03 21:13:36 UTC | porto alegre | RS |
| 10 | b8cf418e97ae795672d326288... | 2016-10-03 22:06:03 UTC | hortolandia | SP |

```
/*  2. In-depth Exploration:
     Q.1. Is there a growing trend on e-commerce in Brazil? How can we
describe a complete scenario?
     Can we see some seasonality with peaks at specific months? */

# Ans. Query :-

with cte as(
select
  extract(year from order_purchase_timestamp) as year,
  count(order_id) as order_count
from
  `Target_SQL.orders`
group by year)

select *,
round((order_count - lag(order_count) over (order by year ))/lag(order_count)
over(order by year) * 100,2) as growth_percent
from cte
order by year ;
```



```
/*  2. In-depth Exploration:
  Q.1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario?
  Can we see some seasonality with peaks at specific months? */

# Ans. Query :-

with cte as(
select
  extract(year from order_purchase_timestamp) as year,
  count(order_id) as order_count
from
  `Target_SQL.orders`
group by year)

select *,
round((order_count - lag(order_count) over (order by year ))/lag(order_count) over(order by year) * 100,2) as growth_percent
from cte
order by year ;
```

| Row | year | order_count | growth_percent |
|---|---|---|---|
| 1 | 2016 | 329 | null |
| 2 | 2017 | 45101 | 13608.51 |
| 3 | 2018 | 54011 | 19.76 |

# Actionable Insights :- Clearly shown that, sales trends are growing year after

year. As startup in 2016, growth was remarkable in 2016 to 2017. And growth percentage in 2017 to 2018 was good, but as compare to previous its very less.

# Recommendations :- For continuous growth in business, look after for expanding the

territory and follow through good marketing strategies.

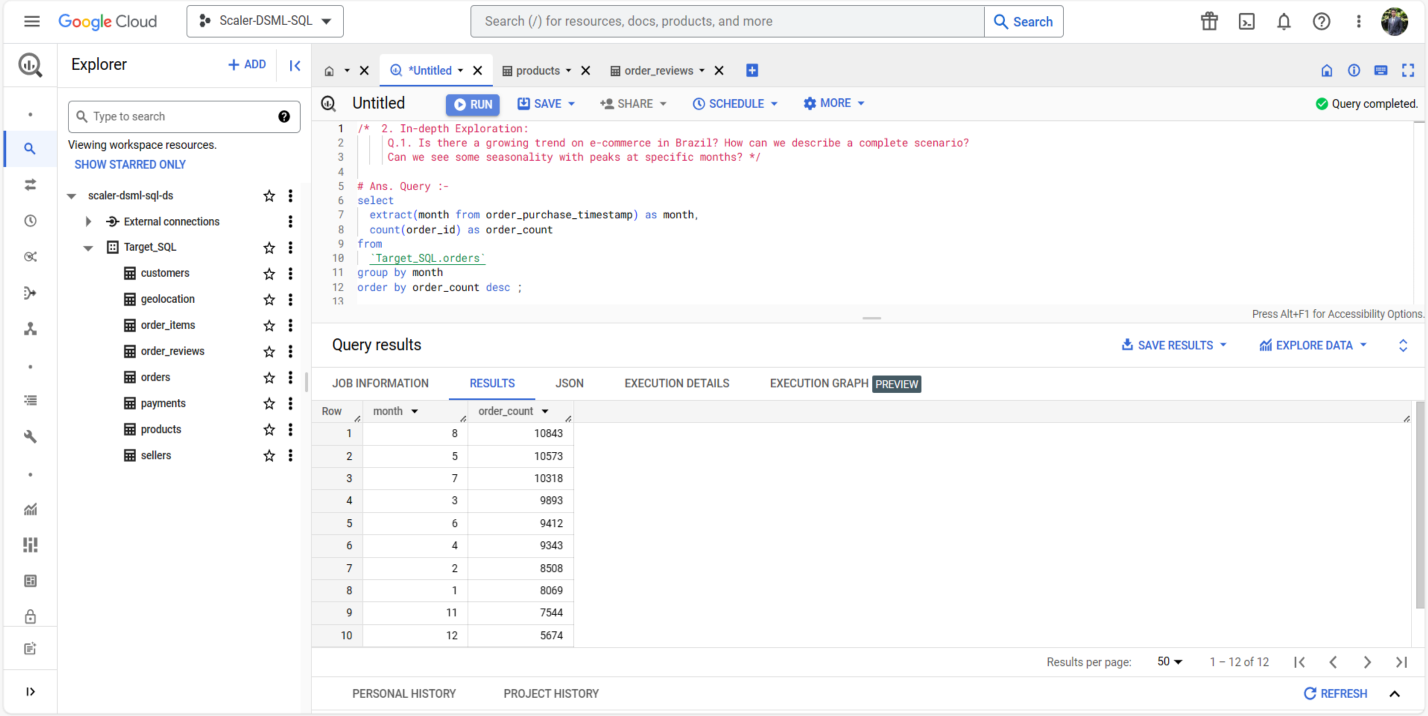
## How can we describe a complete scenario?

Can we see some seasonality with peaks at specific months?

# Ans. Query :-

```
select
  extract(month from order_purchase_timestamp) as month,
  count(order_id) as order_count
from
  `Target_SQL.orders`
group by month
order by order_count desc ;
```

## as seen in the query, in the month of August, May and June is highest orders procured.



```
/*  2. In-depth Exploration:
        Q.1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario?
        Can we see some seasonality with peaks at specific months? */

# Ans. Query :-
select
  extract(month from order_purchase_timestamp) as month,
  count(order_id) as order_count
from
  `Target_SQL.orders`
group by month
order by order_count desc ;
```

Query completed.

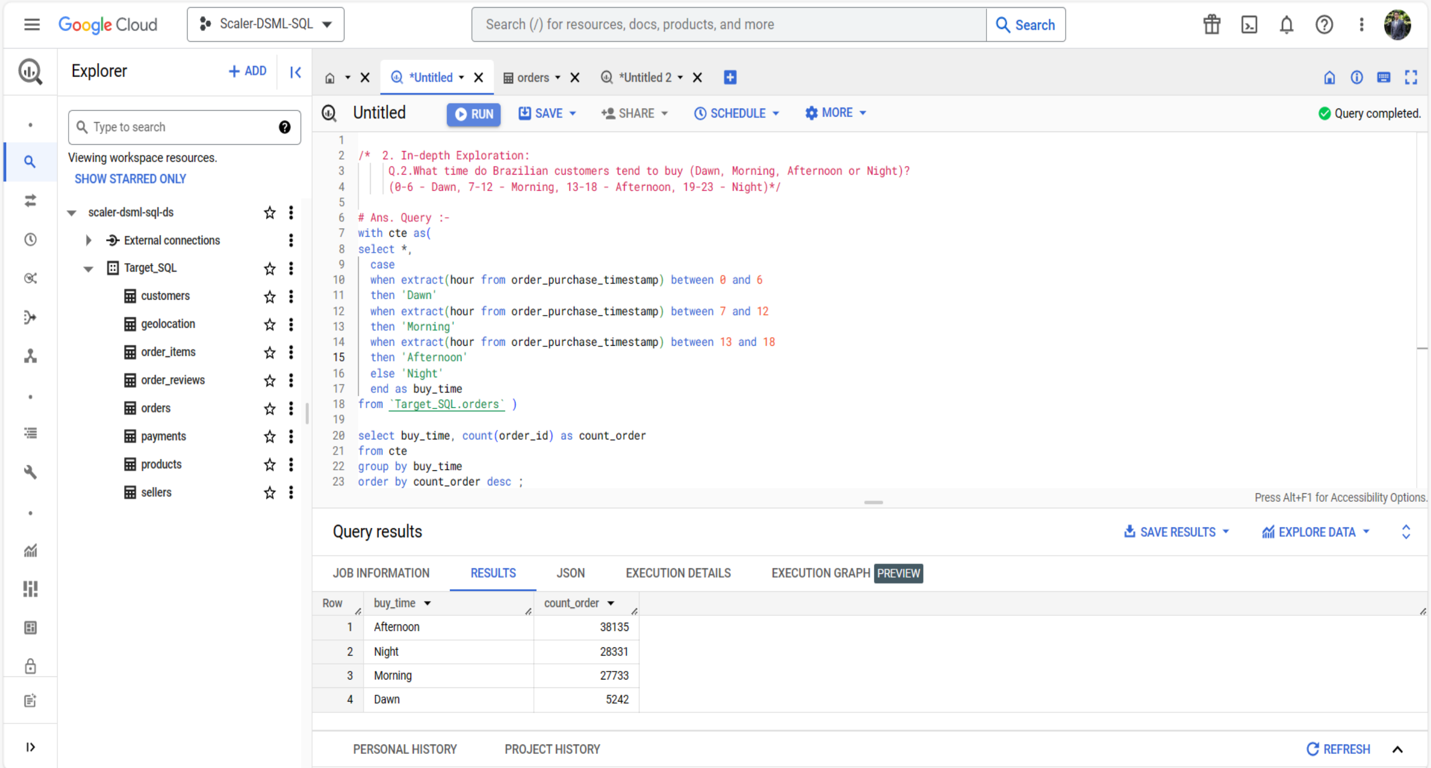
| Row | month | order_count |
|---|---|---|
| 1 | 8 | 10843 |
| 2 | 5 | 10573 |
| 3 | 7 | 10318 |
| 4 | 3 | 9893 |
| 5 | 6 | 9412 |
| 6 | 4 | 9343 |
| 7 | 2 | 8508 |
| 8 | 1 | 8069 |
| 9 | 11 | 7544 |
| 10 | 12 | 5674 |

Results per page: 50 1 – 12 of 12

# Actionable Insights :- Sales trends are peaks in the month of August, May and July respectively.

# Recommendations :- By giving some lucrative offer or discount sale like marketing strategies in the remaining month , sales can be boosted up. And for this peak three month (August, May and July) giving some lucrative offer , sales can be boosted up further.

```
/*  2. In-depth Exploration:
     Q.2.What time do Brazilian customers tend to buy (Dawn, Morning,
Afternoon or Night)?
     (0-6 - Dawn, 7-12 - Morning, 13-18 - Afternoon, 19-23 - Night)*/

# Ans. Query :-

with cte as(
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'
  else 'Night'
  end as buy_time
from `Target_SQL.orders` )

select buy_time, count(order_id) as count_order
from cte
group by buy_time
order by count_order desc ;
```



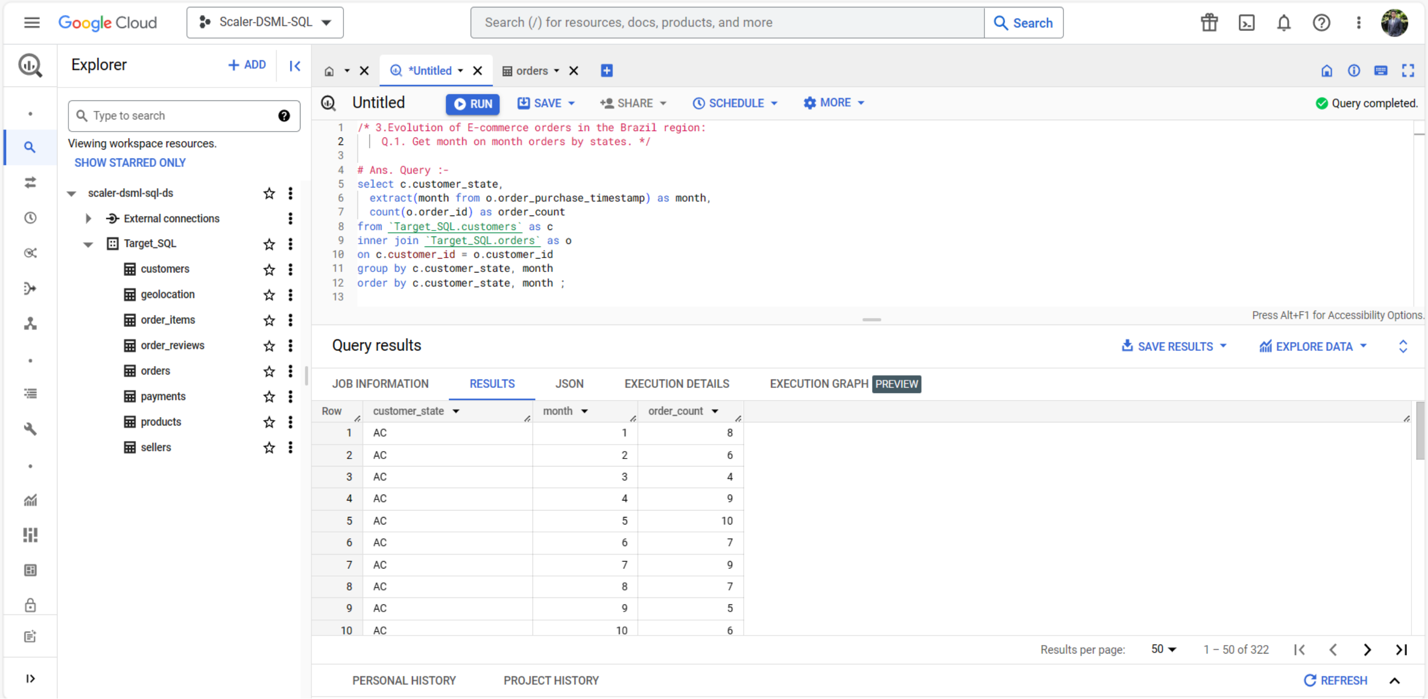
```
/*  2. In-depth Exploration:
    Q.2.What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?
    (0-6 - Dawn, 7-12 - Morning, 13-18 - Afternoon, 19-23 - Night)*/

# Ans. Query :-
with cte as(
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'
  else 'Night'
  end as buy_time
from `Target_SQL.orders` )

select buy_time, count(order_id) as count_order
from cte
group by buy_time
order by count_order desc ;
```

| Row | buy_time | count_order |
|---|---|---|
| 1 | Afternoon | 38135 |
| 2 | Night | 28331 |
| 3 | Morning | 27733 |
| 4 | Dawn | 5242 |

# Actionable Insights :- Brazilian customer tend to buy mostly in Afternoon time.

# Recommendations :- By giving “Happy Hour Sale” like marketing strategies in afternoon, sales can be boosted up further.

```
/* 3.Evolution of E-commerce orders in the Brazil region:
Q.1. Get month on month orders by states. */

# Ans. Query :-

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



```
/* 3.Evolution of E-commerce orders in the Brazil region:
  | Q.1. Get month on month orders by states. */

# Ans. Query :-
select c.customer_state,
  extract(month from o.order_purchase_timestamp) as month,
  count(o.order_id) as order_count
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on c.customer_id = o.customer_id
group by c.customer_state, month
order by c.customer_state, month ;
```

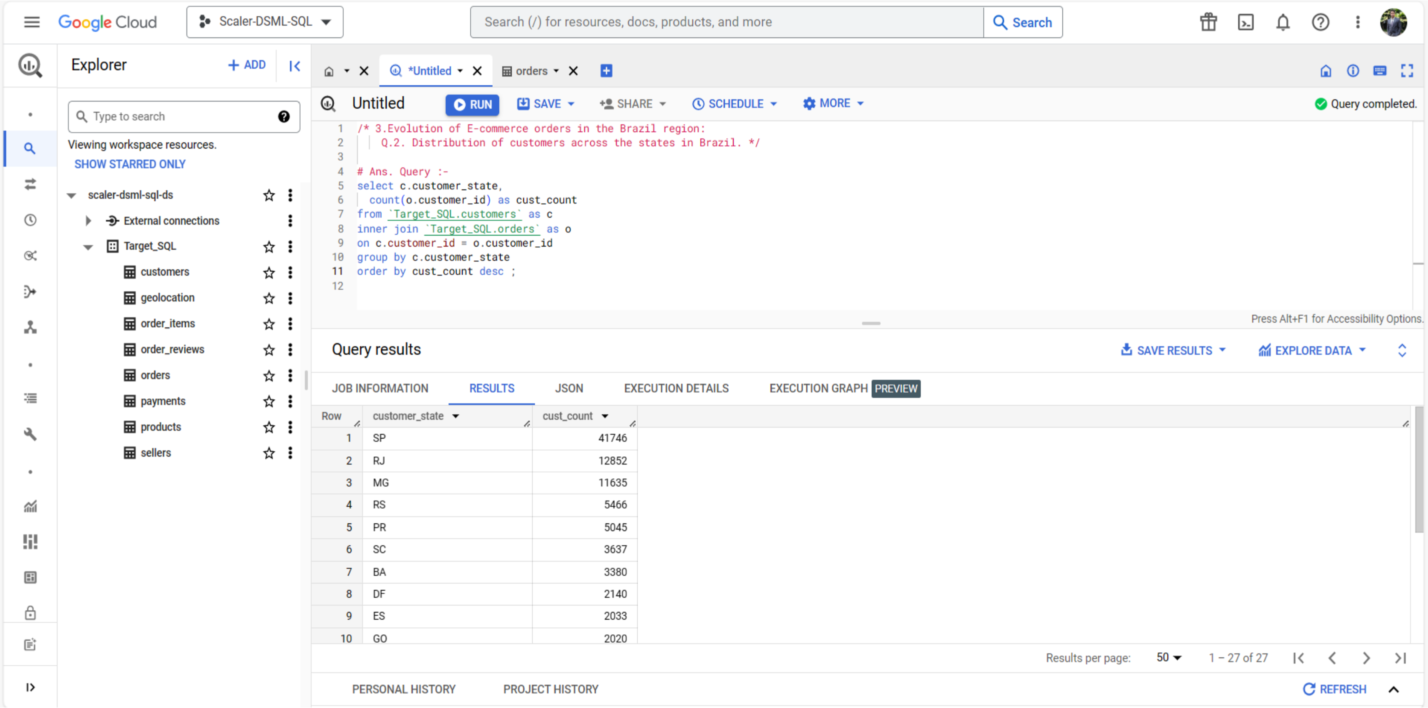
| Row | customer_state | month | order_count |
|---|---|---|---|
| 1 | AC | 1 | 8 |
| 2 | AC | 2 | 6 |
| 3 | AC | 3 | 4 |
| 4 | AC | 4 | 9 |
| 5 | AC | 5 | 10 |
| 6 | AC | 6 | 7 |
| 7 | AC | 7 | 9 |
| 8 | AC | 8 | 7 |
| 9 | AC | 9 | 5 |
| 10 | AC | 10 | 6 |

Results per page: 50 1 – 50 of 322

Q.2. Distribution of customers across the states in Brazil. */

# Ans. Query :-

```
select c.customer_state,
  count(o.customer_id) as cust_count
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on c.customer_id = o.customer_id
group by c.customer_state
order by cust_count desc ;
```



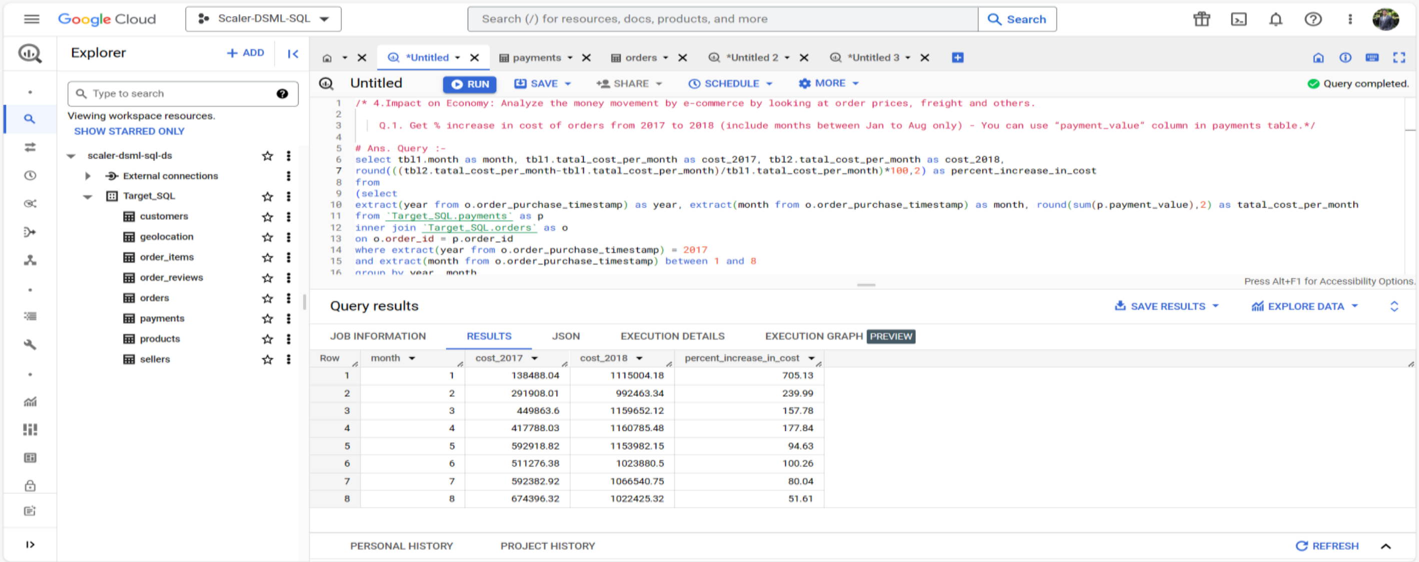
```
/* 3.Evolution of E-commerce orders in the Brazil region:
    Q.2. Distribution of customers across the states in Brazil. */

# Ans. Query :-
select c.customer_state,
  count(o.customer_id) as cust_count
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on c.customer_id = o.customer_id
group by c.customer_state
order by cust_count desc ;
```

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

# Actionable Insights :- State ‘SP’,’RJ’ and ‘MG’ has most customer across the Brazil.

# Recommendations :- By doing various marketing and campaign in the remaining states sales can be boost up.

```
/* 4.Impact on Economy: Analyze the money movement by e-commerce by
looking at order prices, freight and others.
    Q.1. Get % increase in cost of orders from 2017 to 2018 (include
months between Jan to Aug only) - You can use “payment_value” column in
payments table.*/
# Ans. Query :-
select tbl1.month as month, tbl1.tatal_cost_per_month as cost_2017,
tbl2.tatal_cost_per_month as cost_2018,
round(((tbl2.tatal_cost_per_month-
tbl1.tatal_cost_per_month)/tbl1.tatal_cost_per_month)*100,2) as
percent_increase_in_cost
from
(select
extract(year from o.order_purchase_timestamp) as year, extract(month from
o.order_purchase_timestamp) as month, round(sum(p.payment_value),2) as
tatal_cost_per_month
from `Target_SQL.payments` as p
inner join `Target_SQL.orders` as o
on o.order_id = p.order_id
where extract(year from o.order_purchase_timestamp) = 2017
and extract(month from o.order_purchase_timestamp) between 1 and 8
group by year, month
order by month) tbl1
inner join
(select
extract(year from o.order_purchase_timestamp) as year, extract(month from
o.order_purchase_timestamp) as month, round(sum(p.payment_value),2) as
tatal_cost_per_month
from `Target_SQL.payments` as p
inner join `Target_SQL.orders` as o
on o.order_id = p.order_id
where extract(year from o.order_purchase_timestamp) = 2018
and extract(month from o.order_purchase_timestamp) between 1 and 8
group by year, month
order by month) tbl2
on tbl1.month = tbl2.month
order by month ;
```



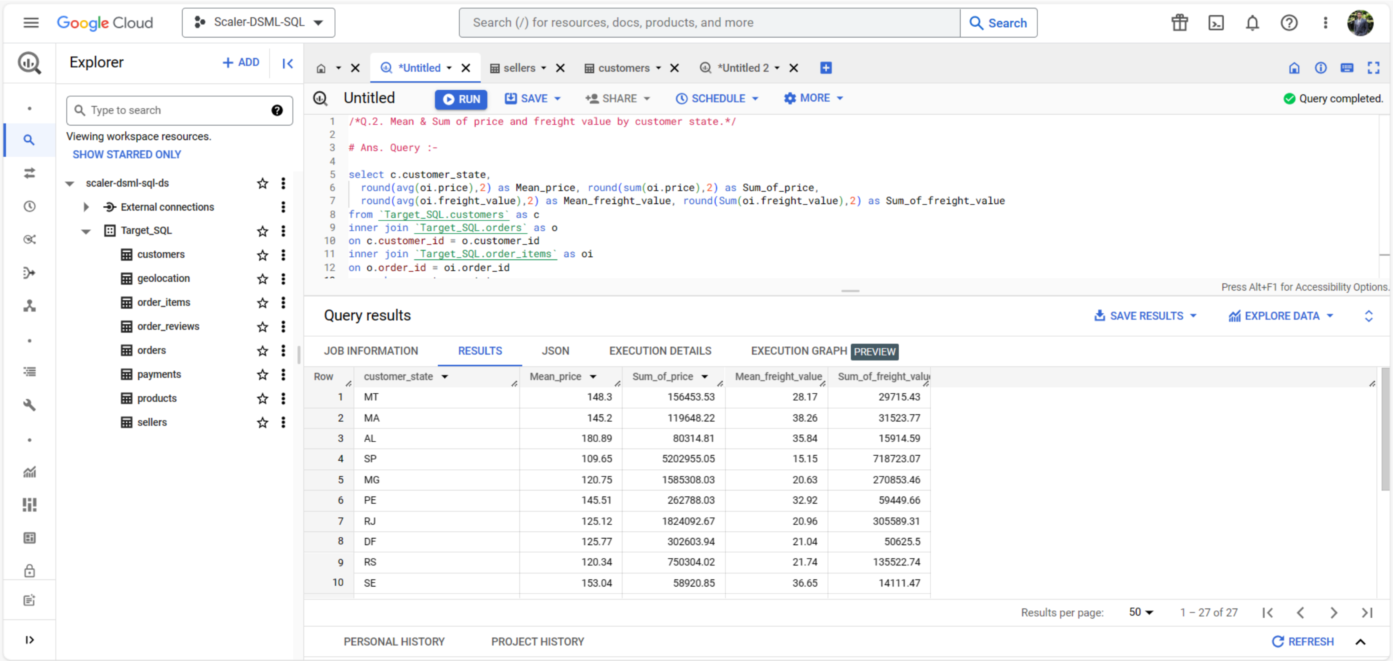
```
/* 4.Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

    Q.1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table.*/

# Ans. Query :-
select tbl1.month as month, tbl1.tatal_cost_per_month as cost_2017, tbl2.tatal_cost_per_month as cost_2018,
round(((tbl2.tatal_cost_per_month-tbl1.tatal_cost_per_month)/tbl1.tatal_cost_per_month)*100,2) as percent_increase_in_cost
from
(select
extract(year from o.order_purchase_timestamp) as year, extract(month from o.order_purchase_timestamp) as month, round(sum(p.payment_value),2) as tatal_cost_per_month
from `Target_SQL.payments` as p
inner join `Target_SQL.orders` as o
on o.order_id = p.order_id
where extract(year from o.order_purchase_timestamp) = 2017
and extract(month from o.order_purchase_timestamp) between 1 and 8
group by year, month
```

Query results

| Row | month | cost_2017 | cost_2018 | percent_increase_in_cost |
|---|---|---|---|---|
| 1 | 1 | 138488.04 | 1115004.18 | 705.13 |
| 2 | 2 | 291908.01 | 992463.34 | 239.99 |
| 3 | 3 | 449863.6 | 1159652.12 | 157.78 |
| 4 | 4 | 417788.03 | 1160785.48 | 177.84 |
| 5 | 5 | 592918.82 | 1153982.15 | 94.63 |
| 6 | 6 | 511276.38 | 1023880.5 | 100.26 |
| 7 | 7 | 592382.92 | 1066540.75 | 80.04 |
| 8 | 8 | 674396.32 | 1022425.32 | 51.61 |

# Actionable Insights :- Cost of orders remarkably increased in 2018 over 2017.

```
Q.2. Mean & Sum of price and freight value by customer state.*/

# Ans. Query :-

select c.customer_state,
  round(avg(oi.price),2) as Mean_price, round(sum(oi.price),2) as Sum_of_price,
  round(avg(oi.freight_value),2) as Mean_freight_value, round(Sum(oi.freight_value),2) as
Sum_of_freight_value
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on c.customer_id = o.customer_id
inner join `Target_SQL.order_items` as oi
on o.order_id = oi.order_id
group by c.customer_state ;
```

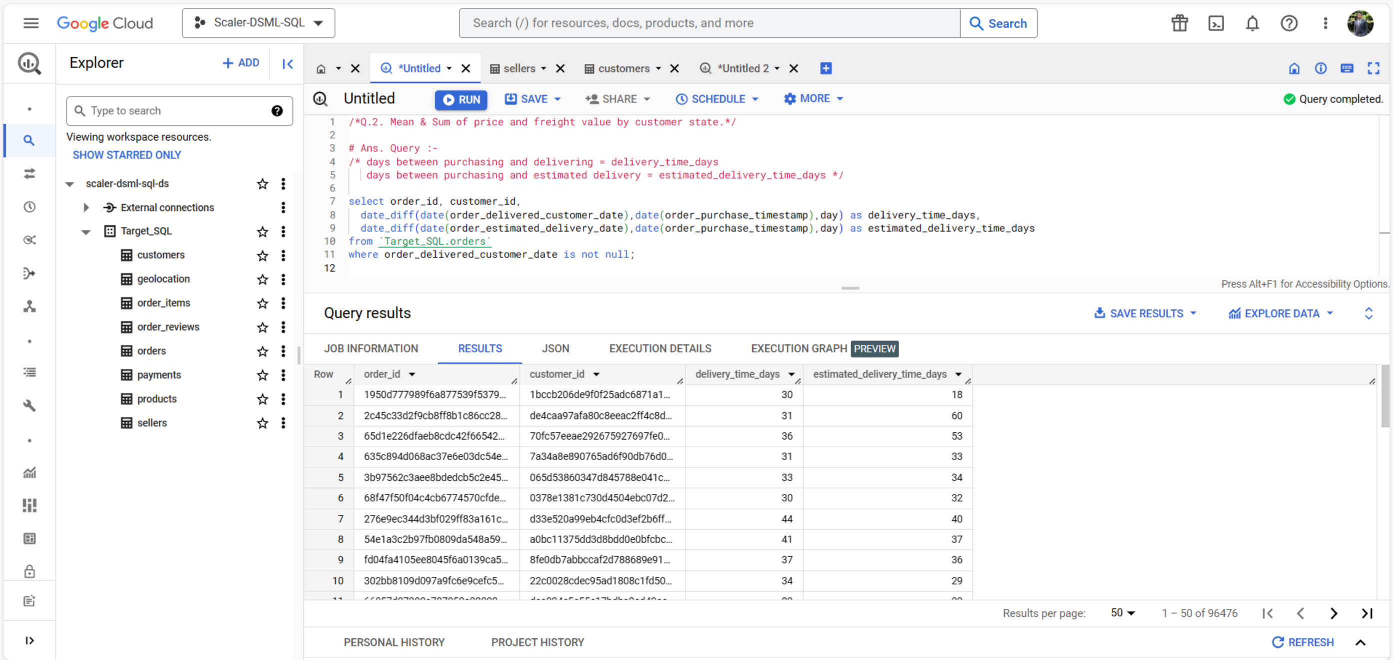


```
/*Q.2. Mean & Sum of price and freight value by customer state.*/

# Ans. Query :-

select c.customer_state,
  round(avg(oi.price),2) as Mean_price, round(sum(oi.price),2) as Sum_of_price,
  round(avg(oi.freight_value),2) as Mean_freight_value, round(Sum(oi.freight_value),2) as Sum_of_freight_value
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on c.customer_id = o.customer_id
inner join `Target_SQL.order_items` as oi
on o.order_id = oi.order_id
```

Query results

| Row | customer_state | Mean_price | Sum_of_price | Mean_freight_value | Sum_of_freight_valu |
|---|---|---|---|---|---|
| 1 | MT | 148.3 | 156453.53 | 28.17 | 29715.43 |
| 2 | MA | 145.2 | 119648.22 | 38.26 | 31523.77 |
| 3 | AL | 180.89 | 80314.81 | 35.84 | 15914.59 |
| 4 | SP | 109.65 | 5202955.05 | 15.15 | 718723.07 |
| 5 | MG | 120.75 | 1585308.03 | 20.63 | 270853.46 |
| 6 | PE | 145.51 | 262788.03 | 32.92 | 59449.66 |
| 7 | RJ | 125.12 | 1824092.67 | 20.96 | 305589.31 |
| 8 | DF | 125.77 | 302603.94 | 21.04 | 50625.5 |
| 9 | RS | 120.34 | 750304.02 | 21.74 | 135522.74 |
| 10 | SE | 153.04 | 58920.85 | 36.65 | 14111.47 |

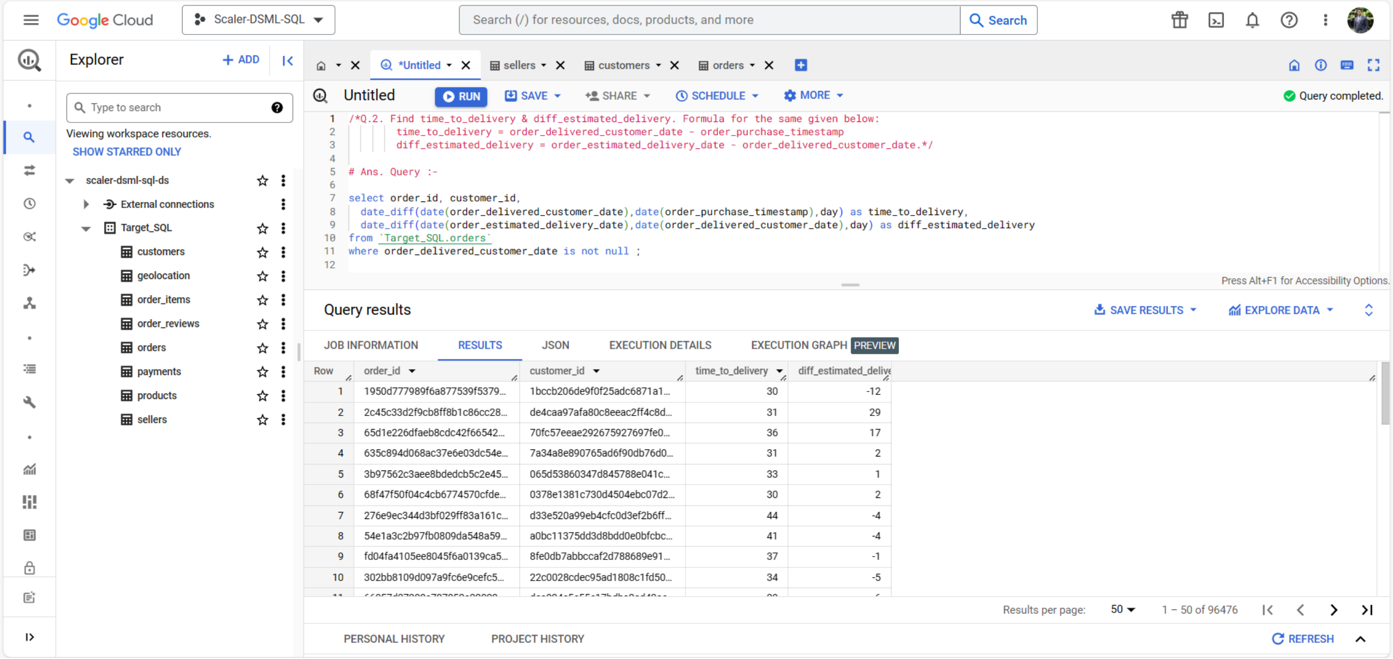
```
/* 5. Analysis on sales, freight and delivery time
    Q.1. Calculate days between purchasing, delivering and estimated delivery.*/

# Ans. Query :-
/* days between purchasing and delivering = delivery_time_days
   days between purchasing and estimated delivery = estimated_delivery_time_days */


select order_id, customer_id,
  date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day) as
delivery_time_days,
  date_diff(date(order_estimated_delivery_date),date(order_purchase_timestamp),day) as
estimated_delivery_time_days
from `Target_SQL.orders`
where order_delivered_customer_date is not null ;
```



```
/*Q.2. Mean & Sum of price and freight value by customer state.*/

# Ans. Query :-
/* days between purchasing and delivering = delivery_time_days
   days between purchasing and estimated delivery = estimated_delivery_time_days */

select order_id, customer_id,
  date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day) as delivery_time_days,
  date_diff(date(order_estimated_delivery_date),date(order_purchase_timestamp),day) as estimated_delivery_time_days
from `Target_SQL.orders`
where order_delivered_customer_date is not null;
```

| Row | order_id | customer_id | delivery_time_days | estimated_delivery_time_days |
|---|---|---|---|---|
| 1 | 1950d777989f6a877539f5379... | 1bccb206de9f0f25adc6871a1... | 30 | 18 |
| 2 | 2c45c33d2f9cb8ff8b1c86cc28... | de4caa97afa80c8eeac2ff4c8d... | 31 | 60 |
| 3 | 65d1e226dfaeb8cdc42f66542... | 70fc57eeae292675927697fe0... | 36 | 53 |
| 4 | 635c894d068ac37e6e03dc54e... | 7a34a8e890765ad6f90db76d0... | 31 | 33 |
| 5 | 3b97562c3aee8bdedcb5c2e45... | 065d53860347d845788e041c... | 33 | 34 |
| 6 | 68f47f50f04c4cb6774570cfde... | 0378e1381c730d4504ebc07d2... | 30 | 32 |
| 7 | 276e9ec344d3bf029ff83a161c... | d33e520a99eb4cfc0d3ef2b6ff... | 44 | 40 |
| 8 | 54e1a3c2b97fb0809da548a59... | a0bc11375dd3d8bdd0e0bfcbc... | 41 | 37 |
| 9 | fd04fa4105ee8045f6a0139ca5... | 8fe0db7abbccaf2d788689e91... | 37 | 36 |
| 10 | 302bb8109d097a9fc6e9cefc5... | 22c0028cdec95ad1808c1fd50... | 34 | 29 |

```
Q.2. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:
        time_to_delivery = order_delivered_customer_date - order_purchase_timestamp
        diff_estimated_delivery = order_estimated_delivery_date -
order_delivered_customer_date.*/

# Ans. Query :-

select order_id, customer_id,
  date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day) as
time_to_delivery,
  date_diff(date(order_estimated_delivery_date),date(order_delivered_customer_date),day) as
diff_estimated_delivery
from `Target_SQL.orders`
where order_delivered_customer_date is not null ;
```



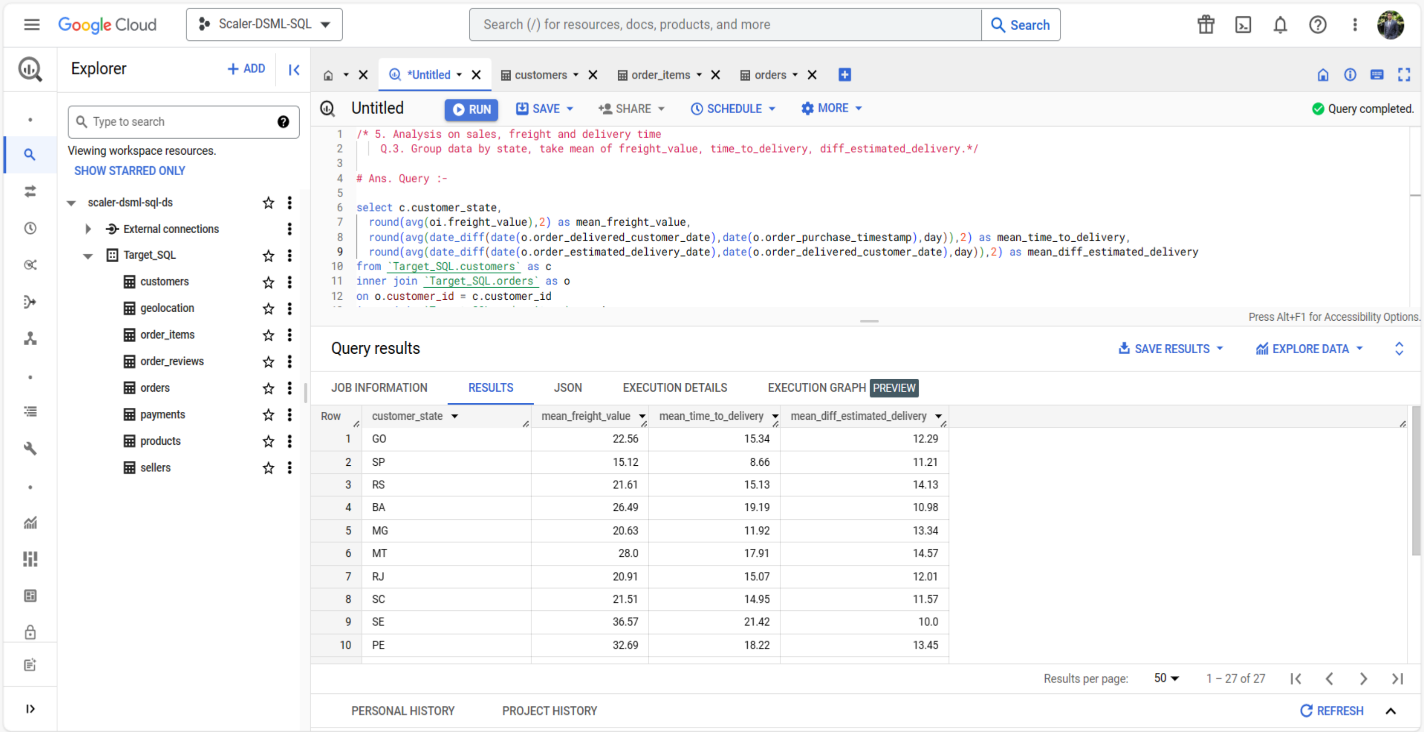
```
/*Q.2. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:
        time_to_delivery = order_delivered_customer_date - order_purchase_timestamp
        diff_estimated_delivery = order_estimated_delivery_date - order_delivered_customer_date.*/

# Ans. Query :-

select order_id, customer_id,
  date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day) as time_to_delivery,
  date_diff(date(order_estimated_delivery_date),date(order_delivered_customer_date),day) as diff_estimated_delivery
from `Target_SQL.orders`
where order_delivered_customer_date is not null ;
```

Query results

| Row | order_id | customer_id | time_to_delivery | diff_estimated_delive |
|---|---|---|---|---|
| 1 | 1950d777989f6a877539f5379... | 1bccb206de9f0f25adc6871a1... | 30 | -12 |
| 2 | 2c45c33d2f9cb8ff8b1c86cc28... | de4caa97afa80c8eeac2ff4c8d... | 31 | 29 |
| 3 | 65d1e226dfaeb8cdc42f66542... | 70fc57eeae292675927697fe0... | 36 | 17 |
| 4 | 635c894d068ac37e6e03dc54e... | 7a34a8e890765ad6f90db76d0... | 31 | 2 |
| 5 | 3b97562c3aee8bdedcb5c2e45... | 065d53860347d845788e041c... | 33 | 1 |
| 6 | 68f47f50f04c4cb6774570cfde... | 0378e1381c730d4504ebc07d2... | 30 | 2 |
| 7 | 276e9ec344d3bf029ff83a161c... | d33e520a99eb4cfc0d3ef2b6ff... | 44 | -4 |
| 8 | 54e1a3c2b97fb0809da548a59... | a0bc11375dd3d8bdd0e0bfcbc... | 41 | -4 |
| 9 | fd04fa4105ee8045f6a0139ca5... | 8fe0db7abbccaf2d788689e91... | 37 | -1 |
| 10 | 302bb8109d097a9fc6e9cefc5... | 22c0028cdec95ad1808c1fd50... | 34 | -5 |

Results per page: 50 1 – 50 of 96476

```
Q.3. Group data by state, take mean of freight_value, time_to_delivery,
diff_estimated_delivery.*/

# Ans. Query :-

select c.customer_state,
  round(avg(oi.freight_value),2) as mean_freight_value,
  round(avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day
)),2) as mean_time_to_delivery,
  round(avg(date_diff(date(o.order_estimated_delivery_date),date(o.order_delivered_customer_date
),day)),2) as mean_diff_estimated_delivery
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on o.customer_id = c.customer_id
inner join `Target_SQL.order_items` as oi
on o.order_id = oi.order_id
group by c.customer_state ;
```

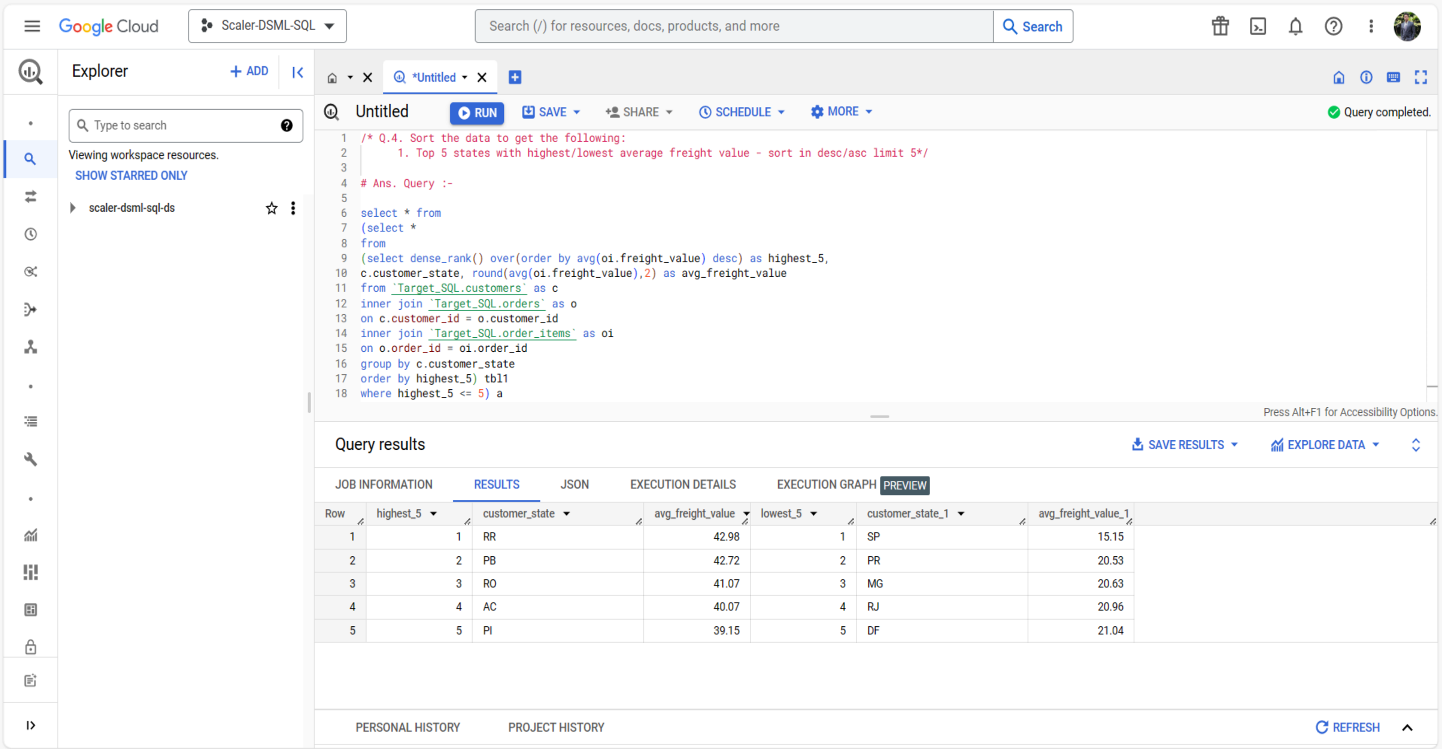


```
/* 5. Analysis on sales, freight and delivery time
  Q.3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery.*/

# Ans. Query :-

select c.customer_state,
  round(avg(oi.freight_value),2) as mean_freight_value,
  round(avg(date_diff(date(o.order_delivered_customer_date),date(o.order_purchase_timestamp),day)),2) as mean_time_to_delivery,
  round(avg(date_diff(date(o.order_estimated_delivery_date),date(o.order_delivered_customer_date),day)),2) as mean_diff_estimated_delivery
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on o.customer_id = c.customer_id
```

Query results

| Row | customer_state | mean_freight_value | mean_time_to_delivery | mean_diff_estimated_delivery |
|---|---|---|---|---|
| 1 | GO | 22.56 | 15.34 | 12.29 |
| 2 | SP | 15.12 | 8.66 | 11.21 |
| 3 | RS | 21.61 | 15.13 | 14.13 |
| 4 | BA | 26.49 | 19.19 | 10.98 |
| 5 | MG | 20.63 | 11.92 | 13.34 |
| 6 | MT | 28.0 | 17.91 | 14.57 |
| 7 | RJ | 20.91 | 15.07 | 12.01 |
| 8 | SC | 21.51 | 14.95 | 11.57 |
| 9 | SE | 36.57 | 21.42 | 10.0 |
| 10 | PE | 32.69 | 18.22 | 13.45 |

```
Q.4. Sort the data to get the following:
      1. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5*/

# Ans. Query :-

select * from
(select *
from
(select dense_rank() over(order by avg(oi.freight_value) desc) as highest_5,
c.customer_state, round(avg(oi.freight_value),2) as avg_freight_value
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on c.customer_id = o.customer_id
inner join `Target_SQL.order_items` as oi
on o.order_id = oi.order_id
group by c.customer_state
order by highest_5) tbl1
where highest_5 <= 5) a
inner join
(select *
from
(select dense_rank() over(order by avg(oi.freight_value)) as lowest_5,
c.customer_state, round(avg(oi.freight_value),2) as avg_freight_value
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on c.customer_id = o.customer_id
inner join `Target_SQL.order_items` as oi
on o.order_id = oi.order_id
group by c.customer_state
order by lowest_5) tbl2
where lowest_5 <= 5) b
on a.highest_5 = b.lowest_5
order by a.highest_5, b.lowest_5 ;
```

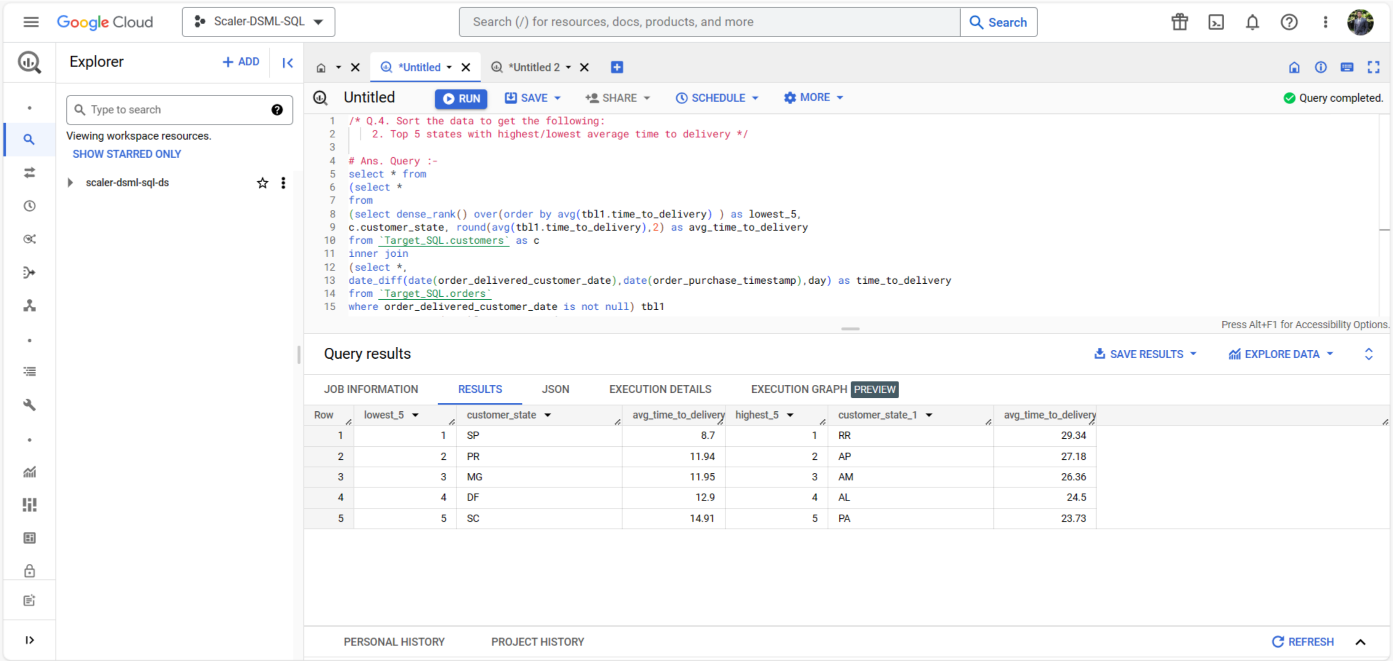


```
/* Q.4. Sort the data to get the following:
      1. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5*/

# Ans. Query :-

select * from
(select *
from
(select dense_rank() over(order by avg(oi.freight_value) desc) as highest_5,
c.customer_state, round(avg(oi.freight_value),2) as avg_freight_value
from `Target_SQL.customers` as c
inner join `Target_SQL.orders` as o
on c.customer_id = o.customer_id
inner join `Target_SQL.order_items` as oi
on o.order_id = oi.order_id
group by c.customer_state
order by highest_5) tbl1
where highest_5 <= 5) a
```

Query results

| Row | highest_5 | customer_state | avg_freight_value | lowest_5 | customer_state_1 | avg_freight_value_1 |
|---|---|---|---|---|---|---|
| 1 | 1 | RR | 42.98 | 1 | SP | 15.15 |
| 2 | 2 | PB | 42.72 | 2 | PR | 20.53 |
| 3 | 3 | RO | 41.07 | 3 | MG | 20.63 |
| 4 | 4 | AC | 40.07 | 4 | RJ | 20.96 |
| 5 | 5 | PI | 39.15 | 5 | DF | 21.04 |

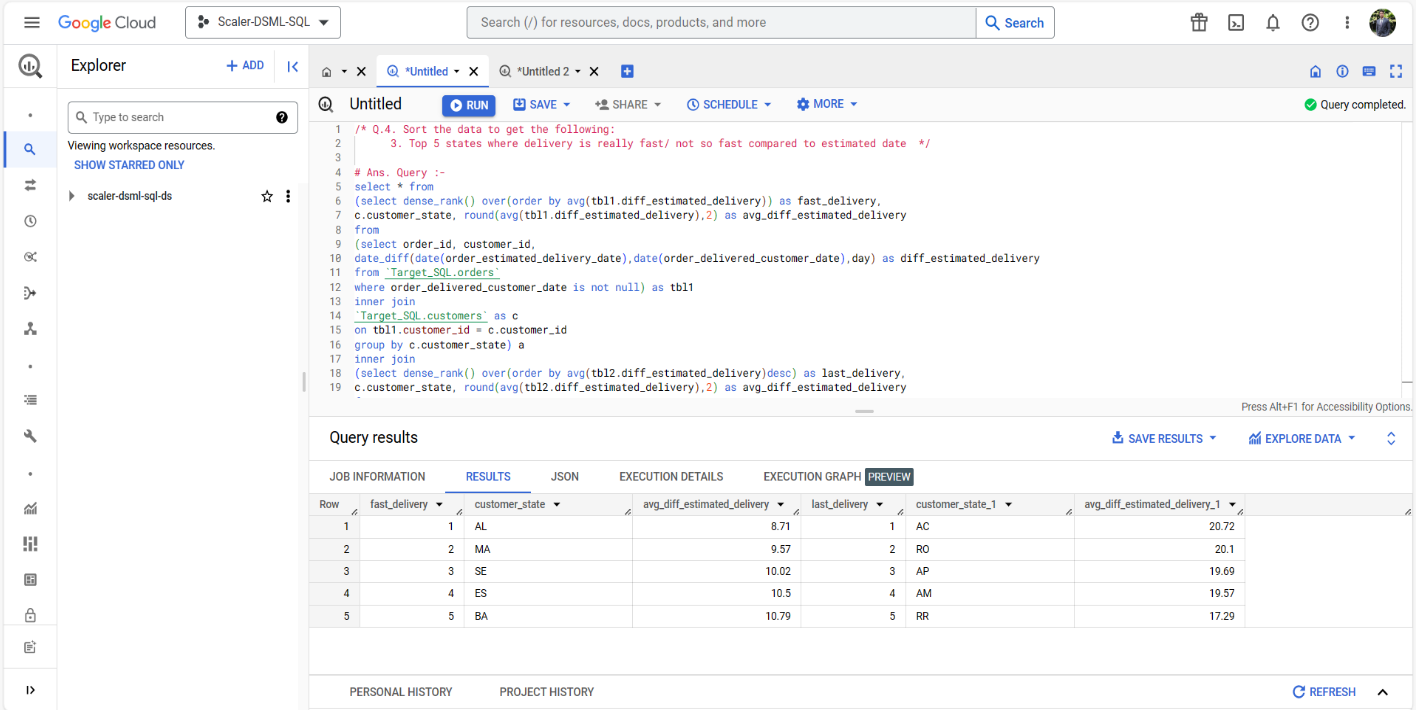
```
/* Q.4. Sort the data to get the following:
    2. Top 5 states with highest/lowest average time to delivery */

# Ans. Query :-
select * from
(select *
from
(select dense_rank() over(order by avg(tbl1.time_to_delivery) ) as lowest_5,
c.customer_state, round(avg(tbl1.time_to_delivery),2) as avg_time_to_delivery
from `Target_SQL.customers` as c
inner join
(select *,
date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day) as
time_to_delivery
from `Target_SQL.orders`
where order_delivered_customer_date is not null) tbl1
on c.customer_id = tbl1.customer_id
group by c.customer_state
order by lowest_5) x
where lowest_5 <= 5) a
inner join
(select *
from
(select dense_rank() over(order by avg(tbl2.time_to_delivery) desc ) as highest_5,
c.customer_state, round(avg(tbl2.time_to_delivery),2) as avg_time_to_delivery
from `Target_SQL.customers` as c
inner join
(select *,
date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day) as
time_to_delivery
from `Target_SQL.orders`
where order_delivered_customer_date is not null) tbl2
on c.customer_id = tbl2.customer_id
group by c.customer_state
order by highest_5) y
where highest_5 <= 5) b
on a.lowest_5 = b.highest_5
order by a.lowest_5, b.highest_5 ;
```



```
/* Q.4. Sort the data to get the following:
    2. Top 5 states with highest/lowest average time to delivery */

# Ans. Query :-
select * from
(select *
from
(select dense_rank() over(order by avg(tbl1.time_to_delivery) ) as lowest_5,
c.customer_state, round(avg(tbl1.time_to_delivery),2) as avg_time_to_delivery
from `Target_SQL.customers` as c
inner join
(select *,
date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day) as time_to_delivery
from `Target_SQL.orders`
where order_delivered_customer_date is not null) tbl1
```

| Row | lowest_5 | customer_state | avg_time_to_delivery | highest_5 | customer_state_1 | avg_time_to_delivery |
|---|---|---|---|---|---|---|
| 1 | 1 | SP | 8.7 | 1 | RR | 29.34 |
| 2 | 2 | PR | 11.94 | 2 | AP | 27.18 |
| 3 | 3 | MG | 11.95 | 3 | AM | 26.36 |
| 4 | 4 | DF | 12.9 | 4 | AL | 24.5 |
| 5 | 5 | SC | 14.91 | 5 | PA | 23.73 |

```
/* Q.4. Sort the data to get the following:
      3. Top 5 states where delivery is really fast/ not so fast compared to estimated date  */

# Ans. Query :-
select * from
(select dense_rank() over(order by avg(tbl1.diff_estimated_delivery)) as fast_delivery,
c.customer_state, round(avg(tbl1.diff_estimated_delivery),2) as avg_diff_estimated_delivery
from
(select order_id, customer_id,
date_diff(date(order_estimated_delivery_date),date(order_delivered_customer_date),day) as
diff_estimated_delivery
from `Target_SQL.orders`
where order_delivered_customer_date is not null) as tbl1
inner join
`Target_SQL.customers` as c
on tbl1.customer_id = c.customer_id
group by c.customer_state) a
inner join
(select dense_rank() over(order by avg(tbl2.diff_estimated_delivery)desc) as last_delivery,
c.customer_state, round(avg(tbl2.diff_estimated_delivery),2) as avg_diff_estimated_delivery
from
(select order_id, customer_id,
date_diff(date(order_estimated_delivery_date),date(order_delivered_customer_date),day) as
diff_estimated_delivery
from `Target_SQL.orders`
where order_delivered_customer_date is not null) as tbl2
inner join
`Target_SQL.customers` as c
on tbl2.customer_id = c.customer_id
group by c.customer_state) b
on a.fast_delivery = b.last_delivery
where a.fast_delivery <= 5 and b.last_delivery <= 5
order by a.fast_delivery, b.last_delivery ;
```



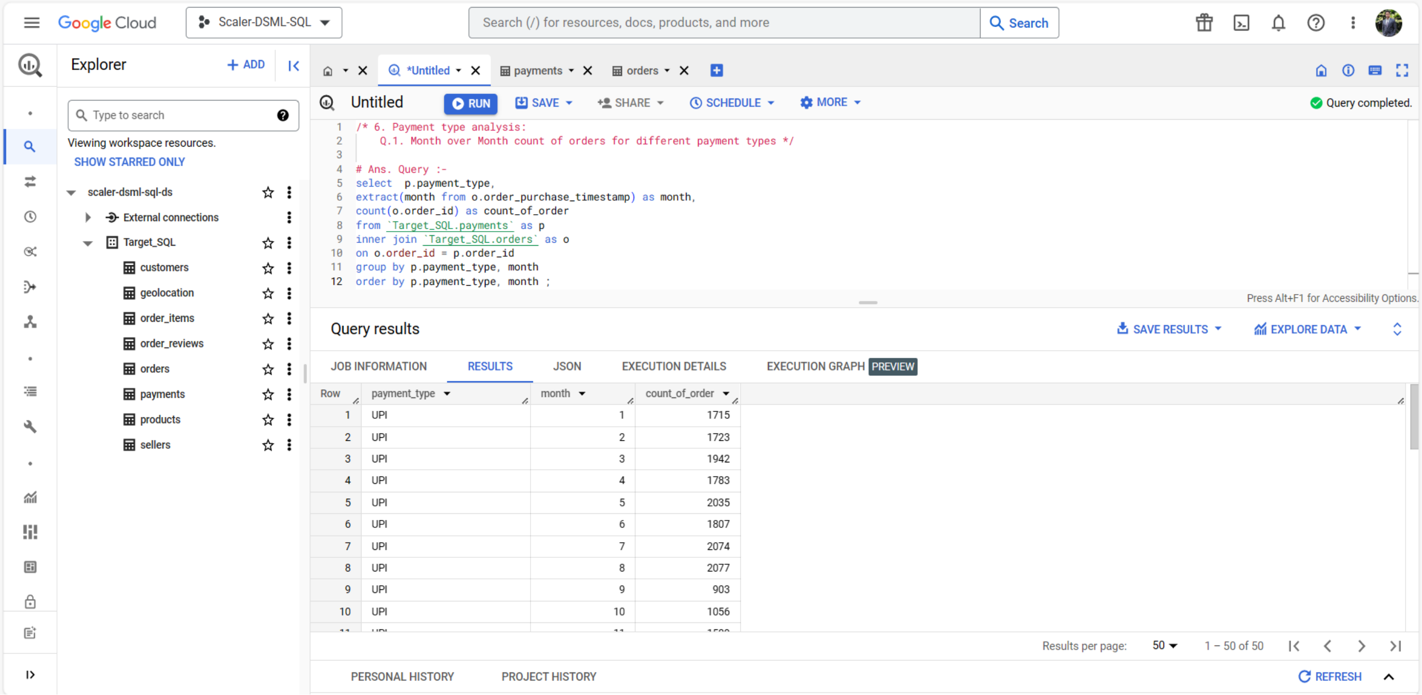
```
/* Q.4. Sort the data to get the following:
      3. Top 5 states where delivery is really fast/ not so fast compared to estimated date  */

# Ans. Query :-
select * from
(select dense_rank() over(order by avg(tbl1.diff_estimated_delivery)) as fast_delivery,
c.customer_state, round(avg(tbl1.diff_estimated_delivery),2) as avg_diff_estimated_delivery
from
(select order_id, customer_id,
date_diff(date(order_estimated_delivery_date),date(order_delivered_customer_date),day) as diff_estimated_delivery
from `Target_SQL.orders`
where order_delivered_customer_date is not null) as tbl1
inner join
`Target_SQL.customers` as c
on tbl1.customer_id = c.customer_id
group by c.customer_state) a
inner join
(select dense_rank() over(order by avg(tbl2.diff_estimated_delivery)desc) as last_delivery,
c.customer_state, round(avg(tbl2.diff_estimated_delivery),2) as avg_diff_estimated_delivery
```

| Row | fast_delivery | customer_state | avg_diff_estimated_delivery | last_delivery | customer_state_1 | avg_diff_estimated_delivery_1 |
|---|---|---|---|---|---|---|
| 1 | 1 | AL | 8.71 | 1 | AC | 20.72 |
| 2 | 2 | MA | 9.57 | 2 | RO | 20.1 |
| 3 | 3 | SE | 10.02 | 3 | AP | 19.69 |
| 4 | 4 | ES | 10.5 | 4 | AM | 19.57 |
| 5 | 5 | BA | 10.79 | 5 | RR | 17.29 |

# Actionable Insights :- In some state delivery was very delayed.

# Recommendations :- By improving delivery time can able to engage more customer.

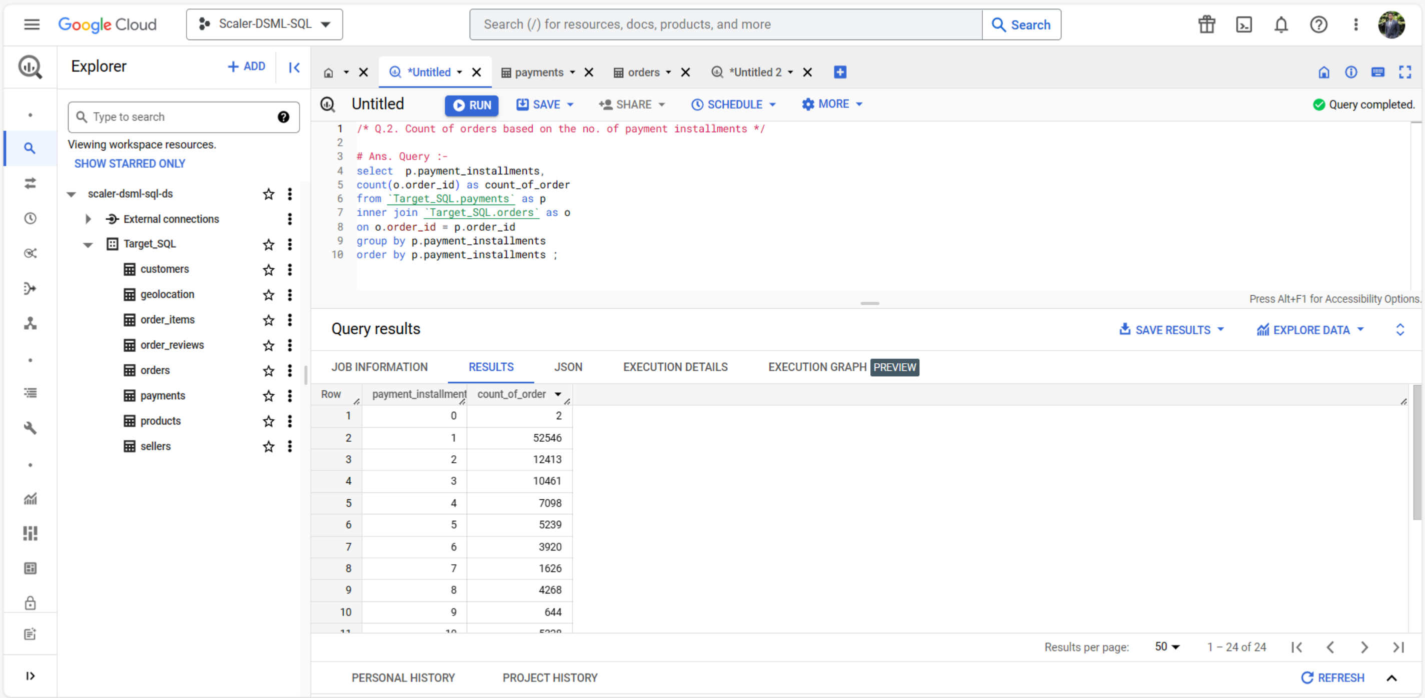
```
/* 6. Payment type analysis:
    Q.1. Month over Month count of orders for different payment types */

# Ans. Query :-
select  p.payment_type,
extract(month from o.order_purchase_timestamp) as month,
count(o.order_id) as count_of_order
from `Target_SQL.payments` as p
inner join `Target_SQL.orders` as o
on o.order_id = p.order_id
group by p.payment_type, month
order by p.payment_type, month ;
```



```
/* 6. Payment type analysis:
    Q.1. Month over Month count of orders for different payment types */

# Ans. Query :-
select  p.payment_type,
extract(month from o.order_purchase_timestamp) as month,
count(o.order_id) as count_of_order
from `Target_SQL.payments` as p
inner join `Target_SQL.orders` as o
on o.order_id = p.order_id
group by p.payment_type, month
order by p.payment_type, month ;
```

| Row | payment_type | month | count_of_order |
|---|---|---|---|
| 1 | UPI | 1 | 1715 |
| 2 | UPI | 2 | 1723 |
| 3 | UPI | 3 | 1942 |
| 4 | UPI | 4 | 1783 |
| 5 | UPI | 5 | 2035 |
| 6 | UPI | 6 | 1807 |
| 7 | UPI | 7 | 2074 |
| 8 | UPI | 8 | 2077 |
| 9 | UPI | 9 | 903 |
| 10 | UPI | 10 | 1056 |

```
/* Q.2. Count of orders based on the no. of payment installments */

# Ans. Query :-

select  p.payment_installments,
count(o.order_id) as count_of_order
from `Target_SQL.payments` as p
inner join `Target_SQL.orders` as o
on o.order_id = p.order_id
group by p.payment_installments
order by count_of_order desc ;
```



```
/* Q.2. Count of orders based on the no. of payment installments */

# Ans. Query :-
select  p.payment_installments,
count(o.order_id) as count_of_order
from `Target_SQL.payments` as p
inner join `Target_SQL.orders` as o
on o.order_id = p.order_id
group by p.payment_installments
order by p.payment_installments ;
```

Query results

| Row | payment_installment | count_of_order |
|---|---|---|
| 1 | 0 | 2 |
| 2 | 1 | 52546 |
| 3 | 2 | 12413 |
| 4 | 3 | 10461 |
| 5 | 4 | 7098 |
| 6 | 5 | 5239 |
| 7 | 6 | 3920 |
| 8 | 7 | 1626 |
| 9 | 8 | 4268 |
| 10 | 9 | 644 |

Results per page: 50 1 – 24 of 24

# Actionable Insights :- Count of orders are more in case of less instalment .