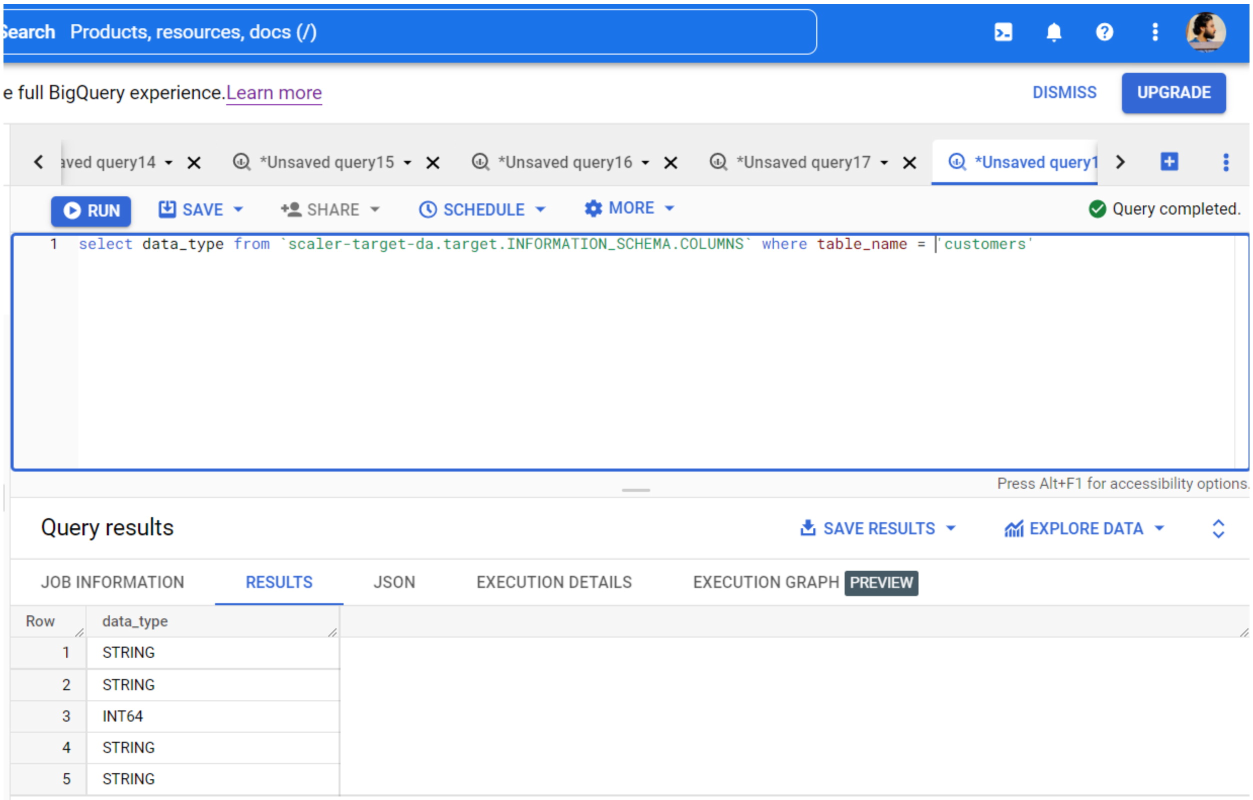
# Target Answer Sheet

1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset
    1. Data type of columns in a table
    2. Time period for which the data is given
    3. Cities and States covered in the dataset

Solution:

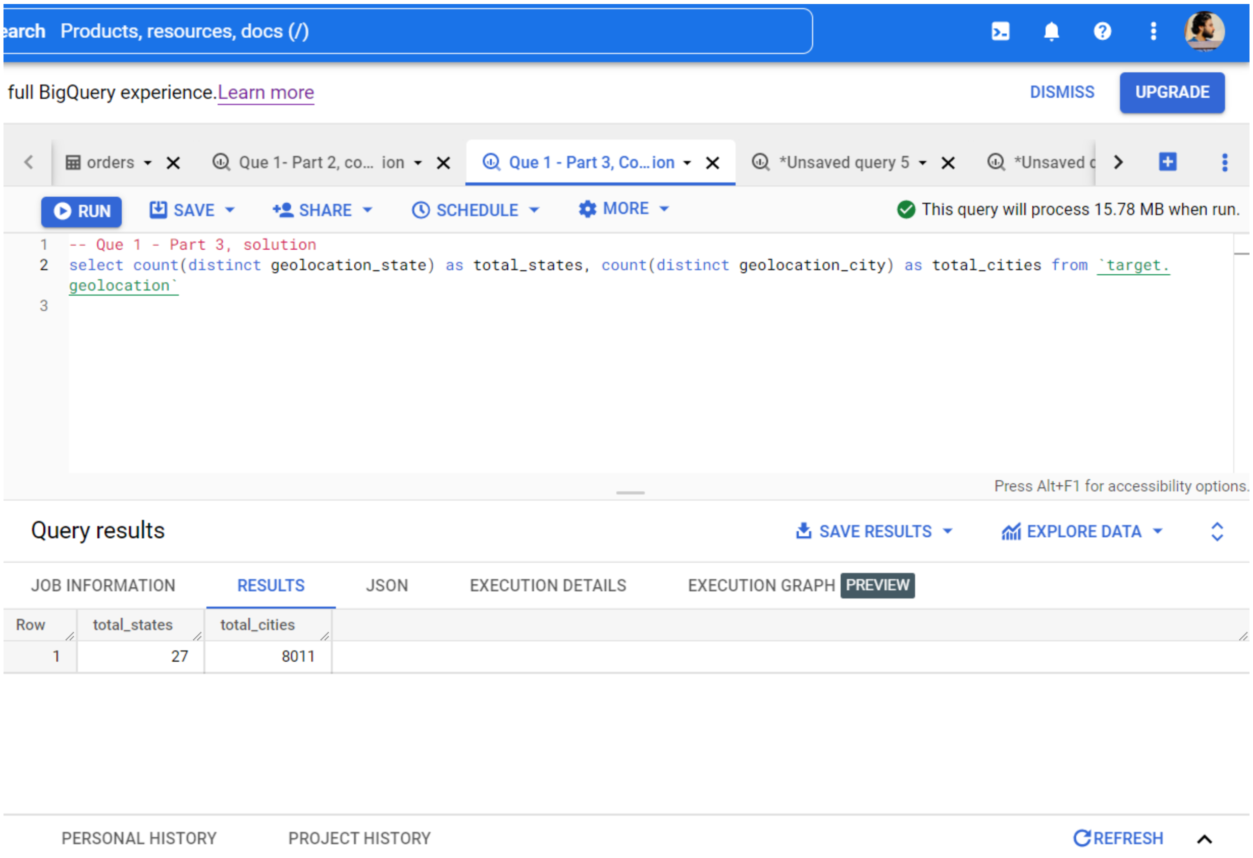
Que 1 – Part 1



```sql
select data_type from `scaler-target-da.target.INFORMATION_SCHEMA.COLUMNS` where table_name = 'customers'
```

| Row | data_type |
|---|---|
| 1 | STRING |
| 2 | STRING |
| 3 | INT64 |
| 4 | STRING |
| 5 | STRING |

## Que 1 – Part 2

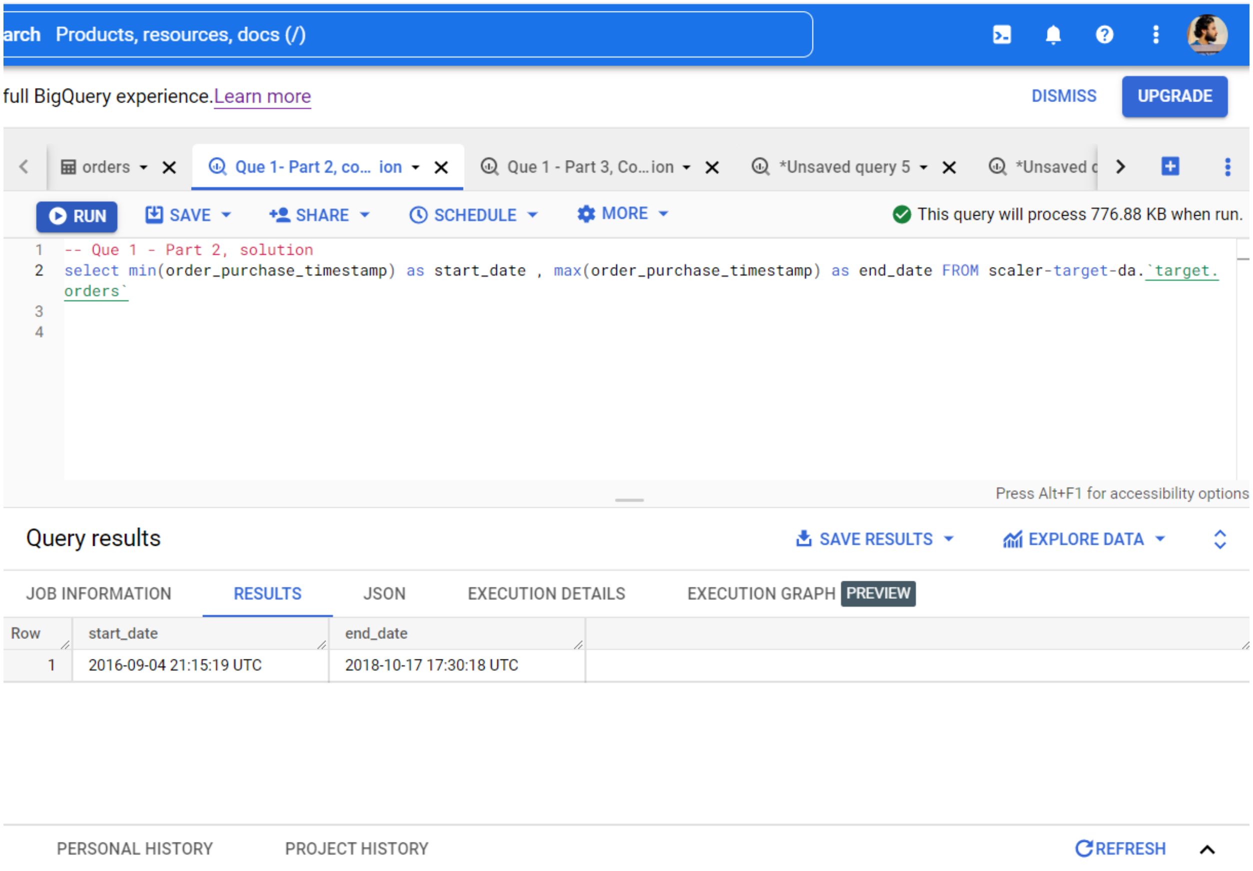


This query will process 15.78 MB when run.

```sql
-- Que 1 - Part 3, solution
select count(distinct geolocation_state) as total_states, count(distinct geolocation_city) as total_cities from `target.
geolocation`
```

| Row | total_states | total_cities |
|---|---|---|
| 1 | 27 | 8011 |

Que 1 - Part 3



full BigQuery experience.Learn more

This query will process 776.88 KB when run.

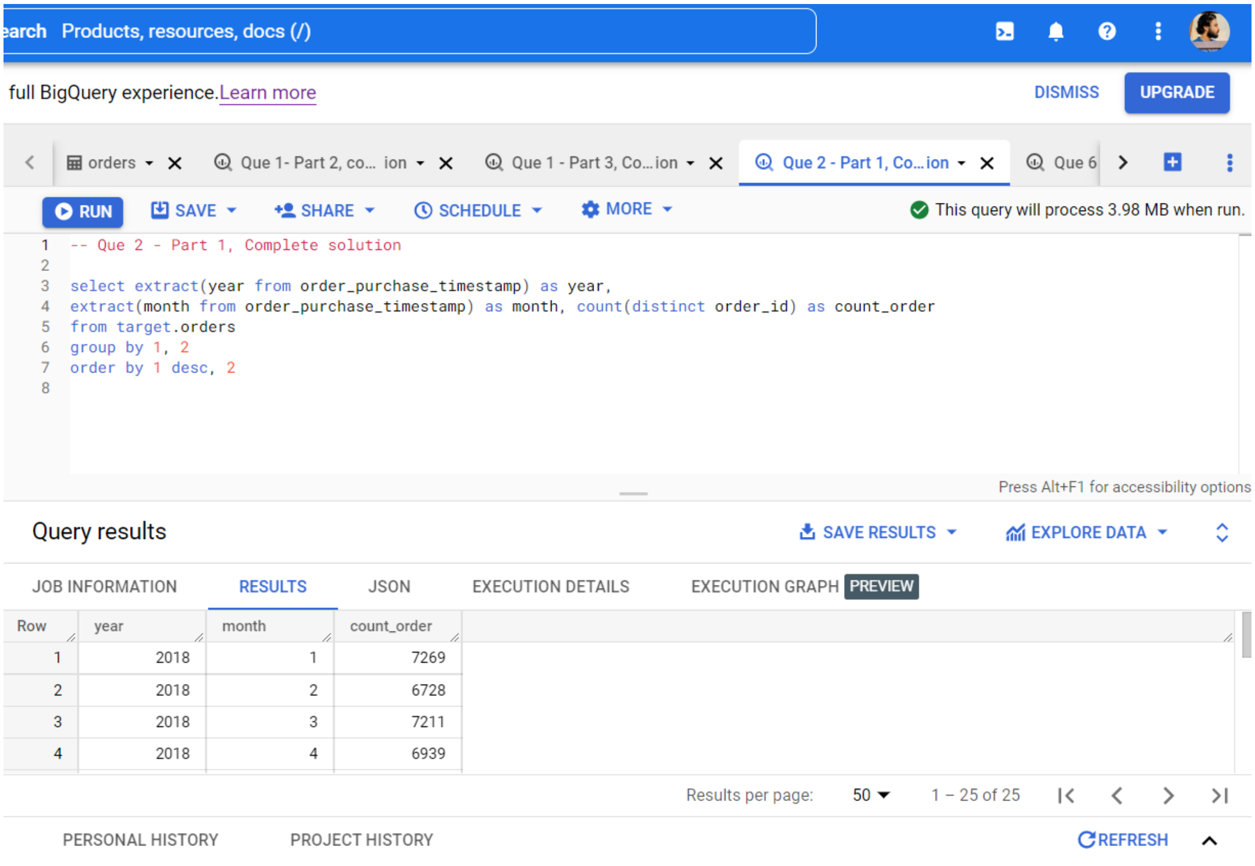
```
-- Que 1 - Part 2, solution
select min(order_purchase_timestamp) as start_date , max(order_purchase_timestamp) as end_date FROM scaler-target-da.`target.orders`
```

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

2. In-depth Exploration:

    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?
    2. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Que 2 – Part 1



full BigQuery experience.Learn more

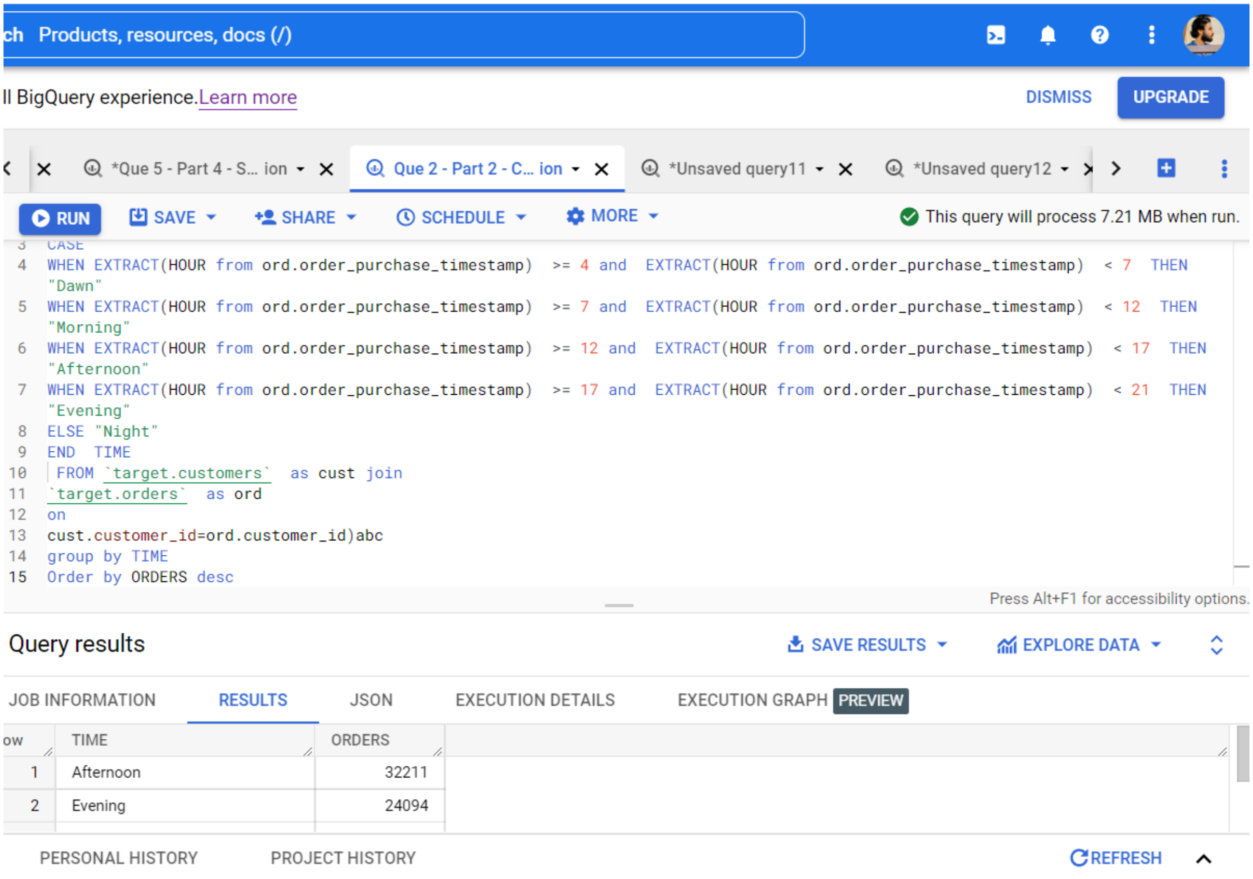
This query will process 3.98 MB when run.

```
-- Que 2 - Part 1, Complete solution

select extract(year from order_purchase_timestamp) as year,
extract(month from order_purchase_timestamp) as month, count(distinct order_id) as count_order
from target.orders
group by 1, 2
order by 1 desc, 2
```

| Row | year | month | count_order |
|---|---|---|---|
| 1 | 2018 | 1 | 7269 |
| 2 | 2018 | 2 | 6728 |
| 3 | 2018 | 3 | 7211 |
| 4 | 2018 | 4 | 6939 |

Results per page: 50 1 – 25 of 25

## Que 2 – Part 2



ll BigQuery experience.Learn more

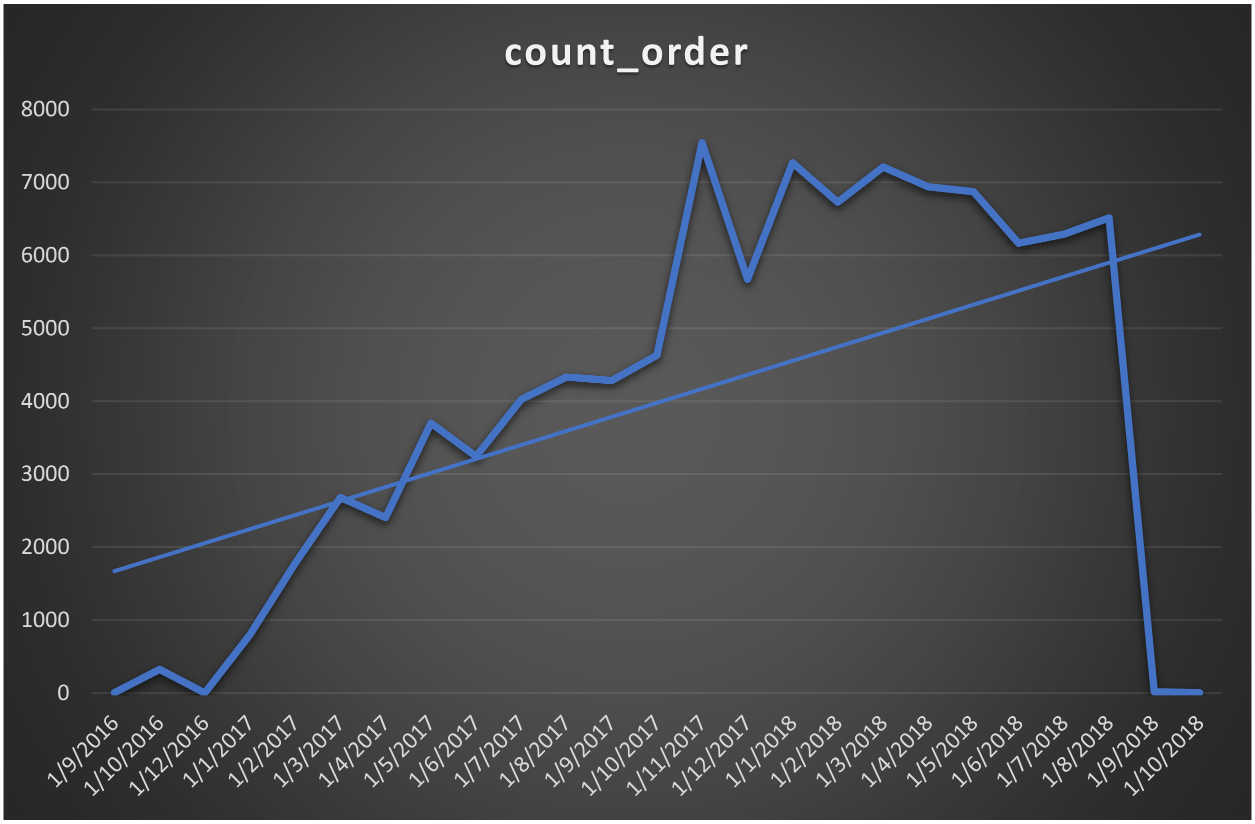
This query will process 7.21 MB when run.

```
CASE
WHEN EXTRACT(HOUR from ord.order_purchase_timestamp)  >= 4 and  EXTRACT(HOUR from ord.order_purchase_timestamp)  < 7  THEN
"Dawn"
WHEN EXTRACT(HOUR from ord.order_purchase_timestamp)  >= 7 and  EXTRACT(HOUR from ord.order_purchase_timestamp)  < 12  THEN
"Morning"
WHEN EXTRACT(HOUR from ord.order_purchase_timestamp)  >= 12 and  EXTRACT(HOUR from ord.order_purchase_timestamp)  < 17  THEN
"Afternoon"
WHEN EXTRACT(HOUR from ord.order_purchase_timestamp)  >= 17 and  EXTRACT(HOUR from ord.order_purchase_timestamp)  < 21  THEN
"Evening"
ELSE "Night"
END  TIME
 FROM `target.customers`  as cust join
`target.orders`  as ord
on
cust.customer_id=ord.customer_id)abc
group by TIME
Order by ORDERS desc
```

Press Alt+F1 for accessibility options.

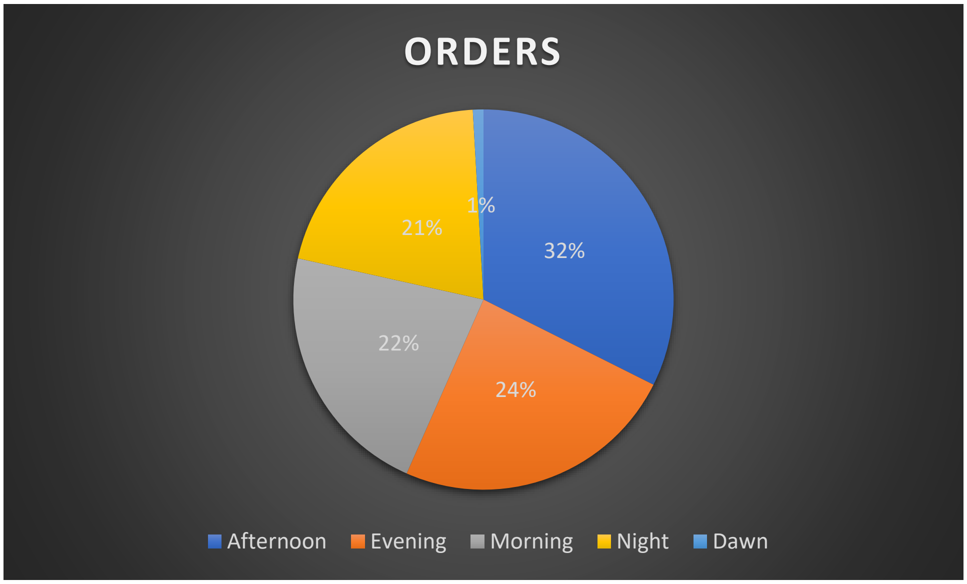
Query results

| ow | TIME | ORDERS |
|---|---|---|
| 1 | Afternoon | 32211 |
| 2 | Evening | 24094 |



With the help of above graph and trendline we can clearly determine a growing trend in e-commerce in Brazil from Dec, 2016 till Aug, 2018, with peaking in Nov, 2017 and a drastic decline in Sep, 2018.

Also, to note that there is no such seasonality found.

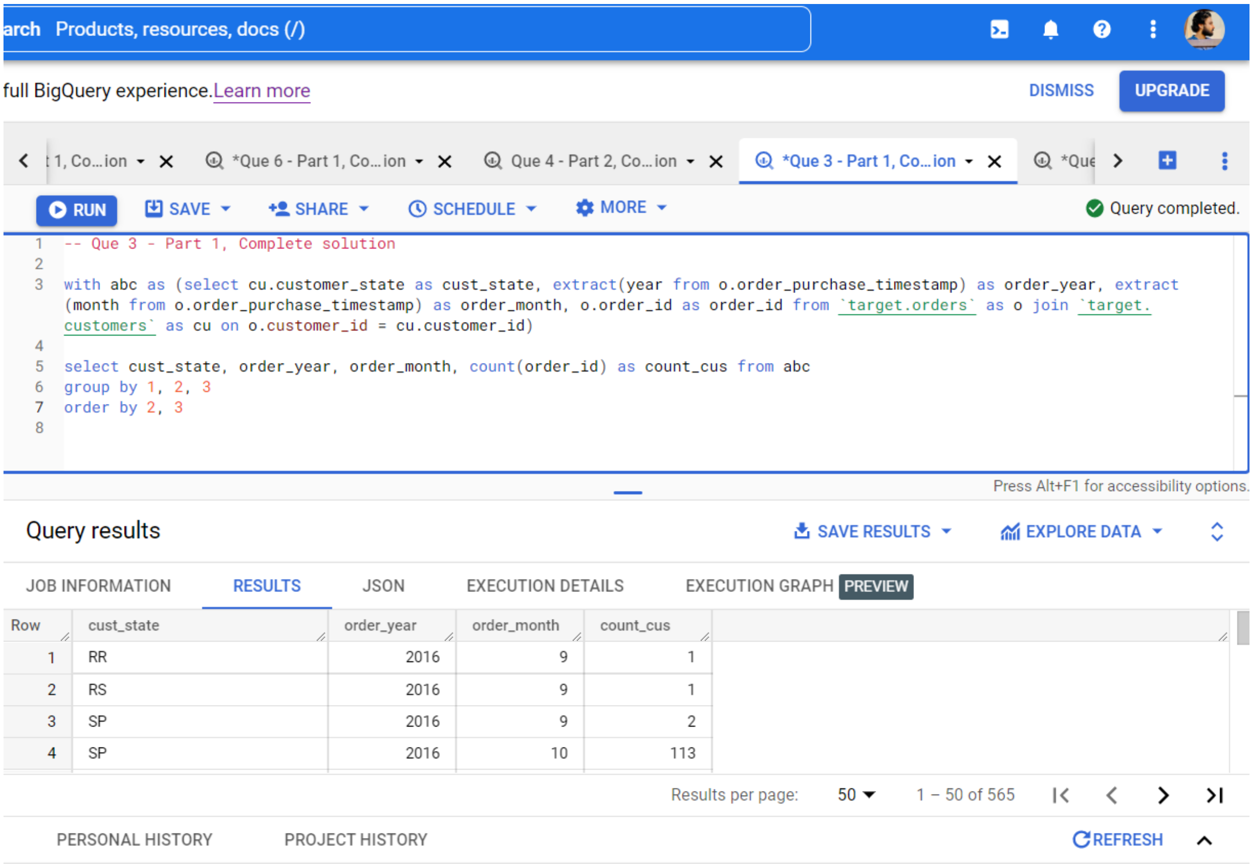


Also, from the above chart it is clearly indicated that maximum customers, i.e. (32211) are buying in afternoon time, followed by evening. And only a handful of users i.e. only 896 are ordering at dawn.

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

    1. Get month on month orders by states
    2. Distribution of customers across the states in Brazil

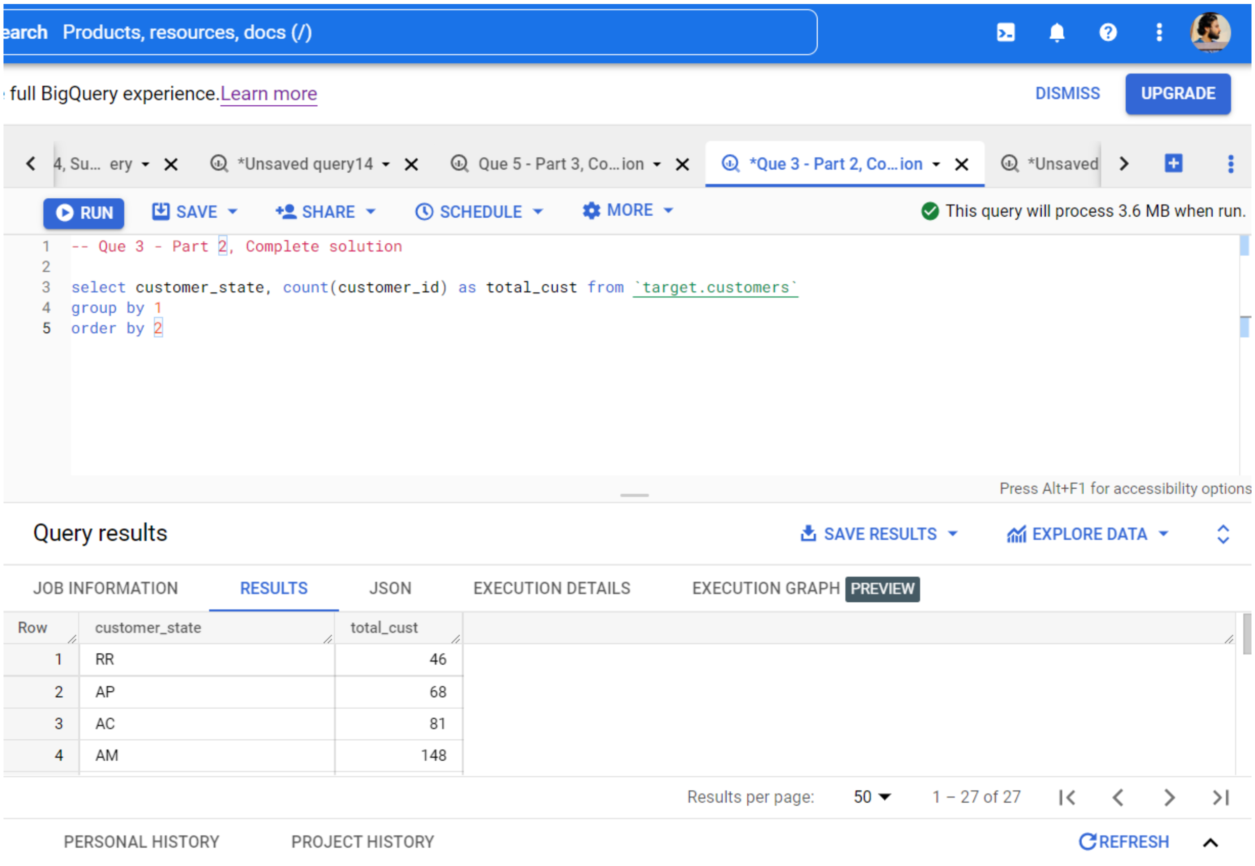
Que 3 – Part 1



```sql
-- Que 3 - Part 1, Complete solution

with abc as (select cu.customer_state as cust_state, extract(year from o.order_purchase_timestamp) as order_year, extract
(month from o.order_purchase_timestamp) as order_month, o.order_id as order_id from `target.orders` as o join `target.
customers` as cu on o.customer_id = cu.customer_id)

select cust_state, order_year, order_month, count(order_id) as count_cus from abc
group by 1, 2, 3
order by 2, 3
```

| Row | cust_state | order_year | order_month | count_cus |
|---|---|---|---|---|
| 1 | RR | 2016 | 9 | 1 |
| 2 | RS | 2016 | 9 | 1 |
| 3 | SP | 2016 | 9 | 2 |
| 4 | SP | 2016 | 10 | 113 |

## Que 3 – Part 2



This query will process 3.6 MB when run.

```sql
-- Que 3 - Part 2, Complete solution

select customer_state, count(customer_id) as total_cust from `target.customers`
group by 1
order by 2
```

| Row | customer_state | total_cust |
|---|---|---|
| 1 | RR | 46 |
| 2 | AP | 68 |
| 3 | AC | 81 |
| 4 | AM | 148 |

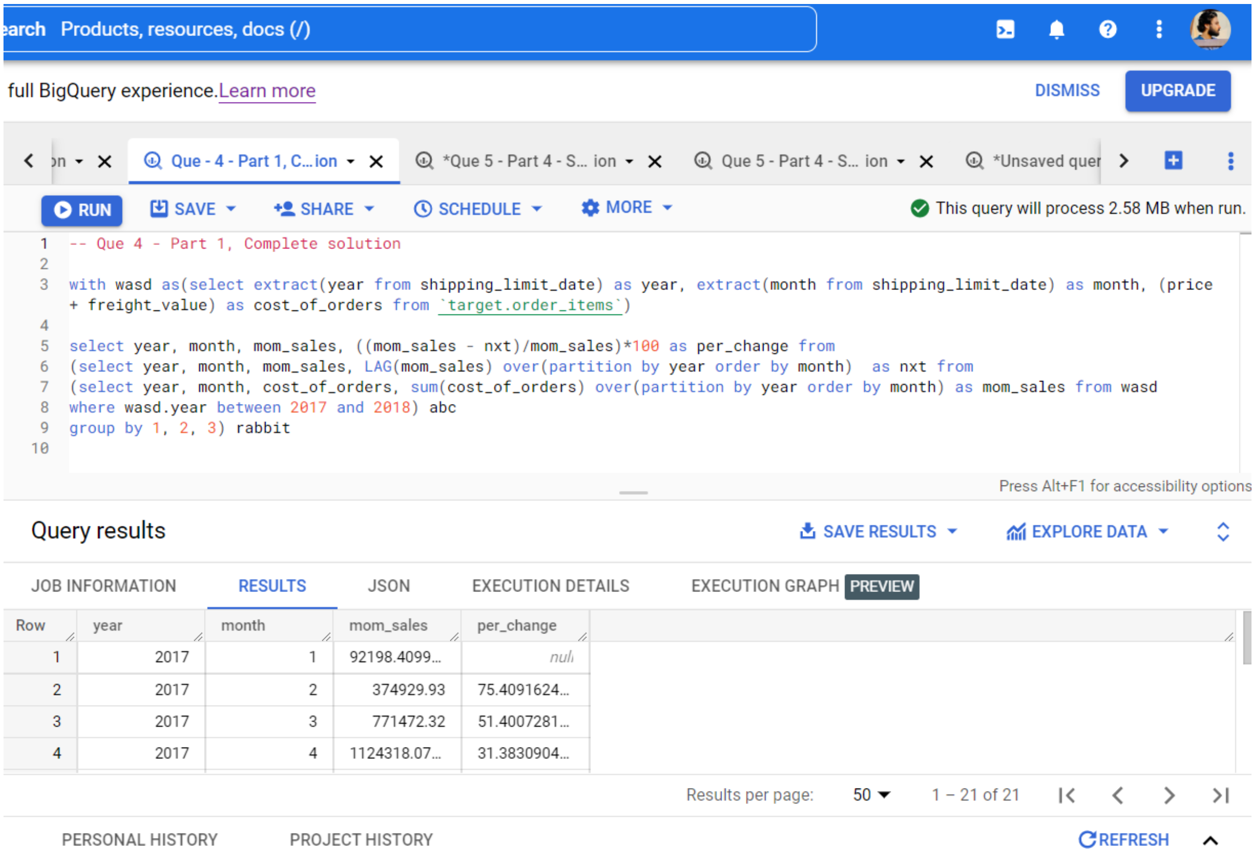
Results per page: 50 1 – 27 of 27

The maximum number of orders comes from the state of **Sao Paulo** and the least from **Roraima**.

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

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

Que 4 – Part 1



full BigQuery experience.Learn more

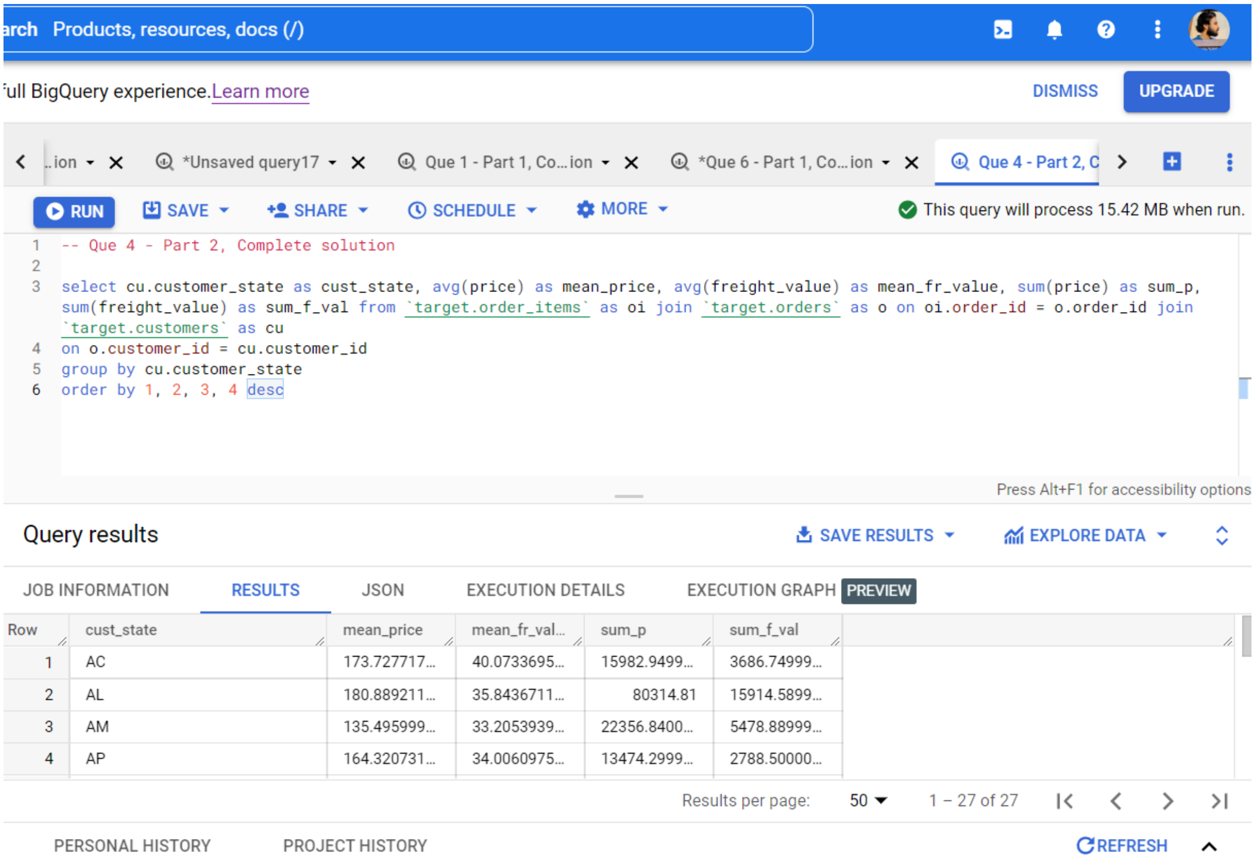
This query will process 2.58 MB when run.

```
-- Que 4 - Part 1, Complete solution

with wasd as(select extract(year from shipping_limit_date) as year, extract(month from shipping_limit_date) as month, (price + freight_value) as cost_of_orders from `target.order_items`)

select year, month, mom_sales, ((mom_sales - nxt)/mom_sales)*100 as per_change from
(select year, month, mom_sales, LAG(mom_sales) over(partition by year order by month)  as nxt from
(select year, month, cost_of_orders, sum(cost_of_orders) over(partition by year order by month) as mom_sales from wasd
where wasd.year between 2017 and 2018) abc
group by 1, 2, 3) rabbit

```

Query results

| Row | year | month | mom_sales | per_change |
|---|---|---|---|---|
| 1 | 2017 | 1 | 92198.4099... | null |
| 2 | 2017 | 2 | 374929.93 | 75.4091624... |
| 3 | 2017 | 3 | 771472.32 | 51.4007281... |
| 4 | 2017 | 4 | 1124318.07... | 31.3830904... |

## Que 4 – Part 2

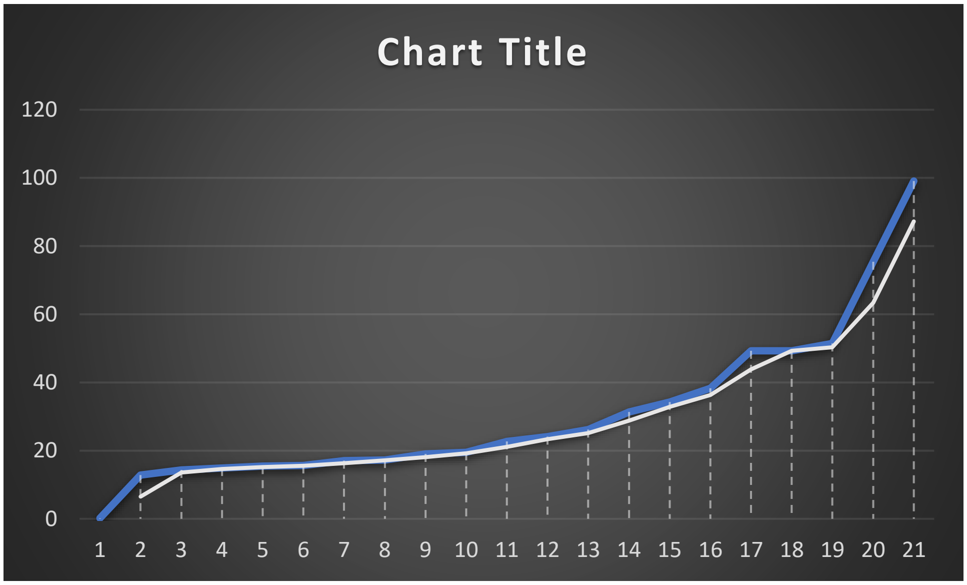


This query will process 15.42 MB when run.

```
-- Que 4 - Part 2, Complete solution

select cu.customer_state as cust_state, avg(price) as mean_price, avg(freight_value) as mean_fr_value, sum(price) as sum_p,
sum(freight_value) as sum_f_val from `target.order_items` as oi join `target.orders` as o on oi.order_id = o.order_id join
`target.customers` as cu
on o.customer_id = cu.customer_id
group by cu.customer_state
order by 1, 2, 3, 4 desc
```

| Row | cust_state | mean_price | mean_fr_val... | sum_p | sum_f_val |
|---|---|---|---|---|---|
| 1 | AC | 173.727717... | 40.0733695... | 15982.9499... | 3686.74999... |
| 2 | AL | 180.889211... | 35.8436711... | 80314.81 | 15914.5899... |
| 3 | AM | 135.495999... | 33.2053939... | 22356.8400... | 5478.88999... |
| 4 | AP | 164.320731... | 34.0060975... | 13474.2999... | 2788.50000... |

Results per page: 50 1 – 27 of 27

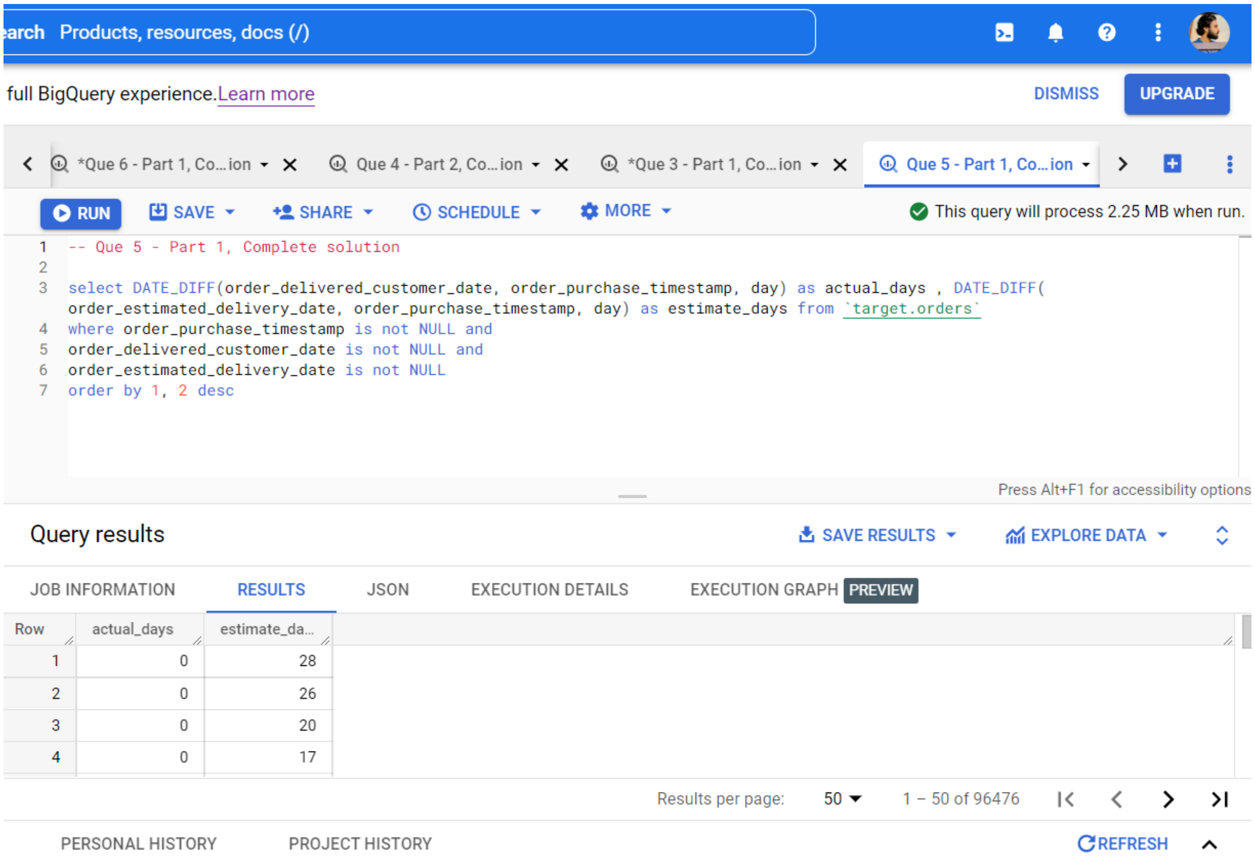


It is certain that the percentage change MOM is in uptrend,

5. Analysis on sales, freight and delivery time

1. Calculate days between purchasing, delivering and estimated delivery
2. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:
    - time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
    - diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date
3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery
4. Sort the data to get the following:
5. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5
6. Top 5 states with highest/lowest average time to delivery
7. Top 5 states where delivery is really fast/ not so fast compared to estimated date

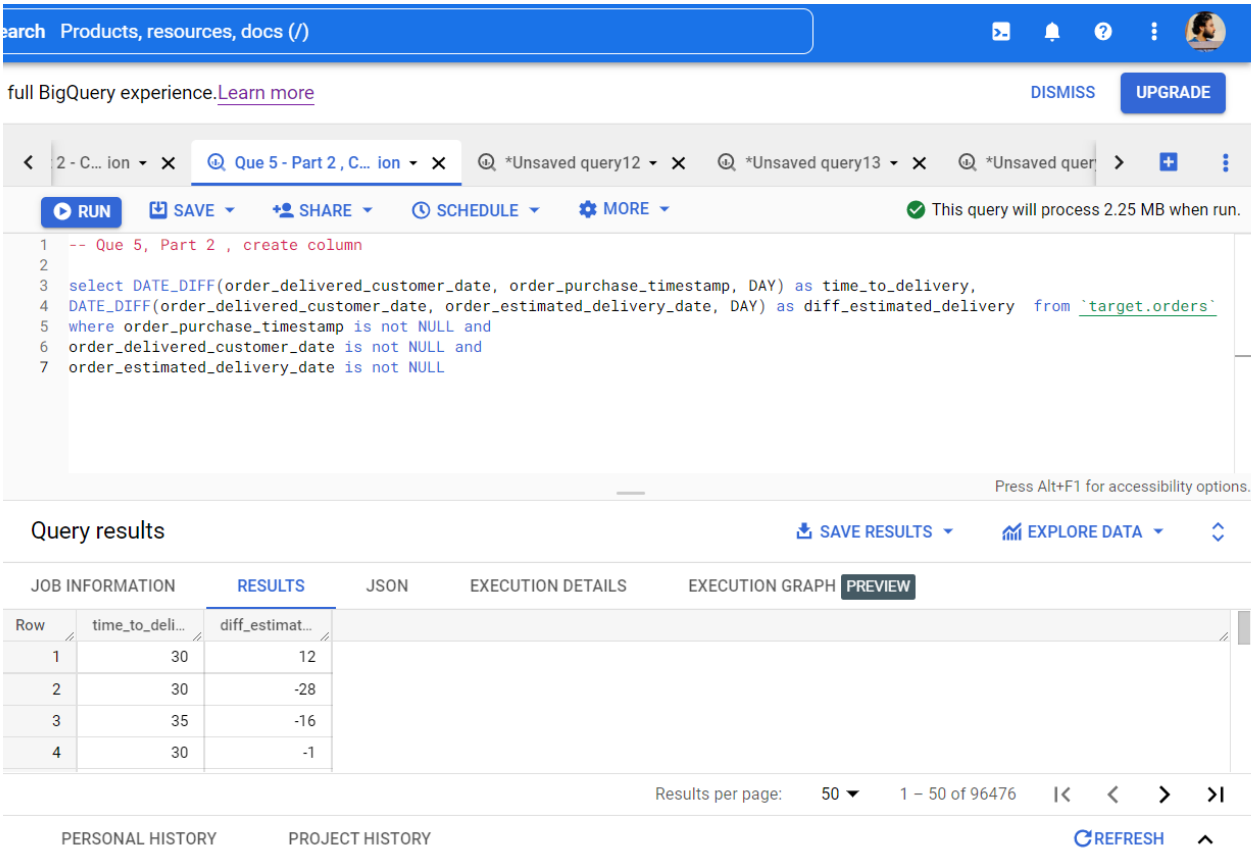
Que 5 – Part 1



```
-- Que 5 - Part 1, Complete solution

select DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) as actual_days , DATE_DIFF(
order_estimated_delivery_date, order_purchase_timestamp, day) as estimate_days from `target.orders`
where order_purchase_timestamp is not NULL and
order_delivered_customer_date is not NULL and
order_estimated_delivery_date is not NULL
order by 1, 2 desc
```

| Row | actual_days | estimate_da... |
|---|---|---|
| 1 | 0 | 28 |
| 2 | 0 | 26 |
| 3 | 0 | 20 |
| 4 | 0 | 17 |

## Que 5 – Part 2



This query will process 2.25 MB when run.

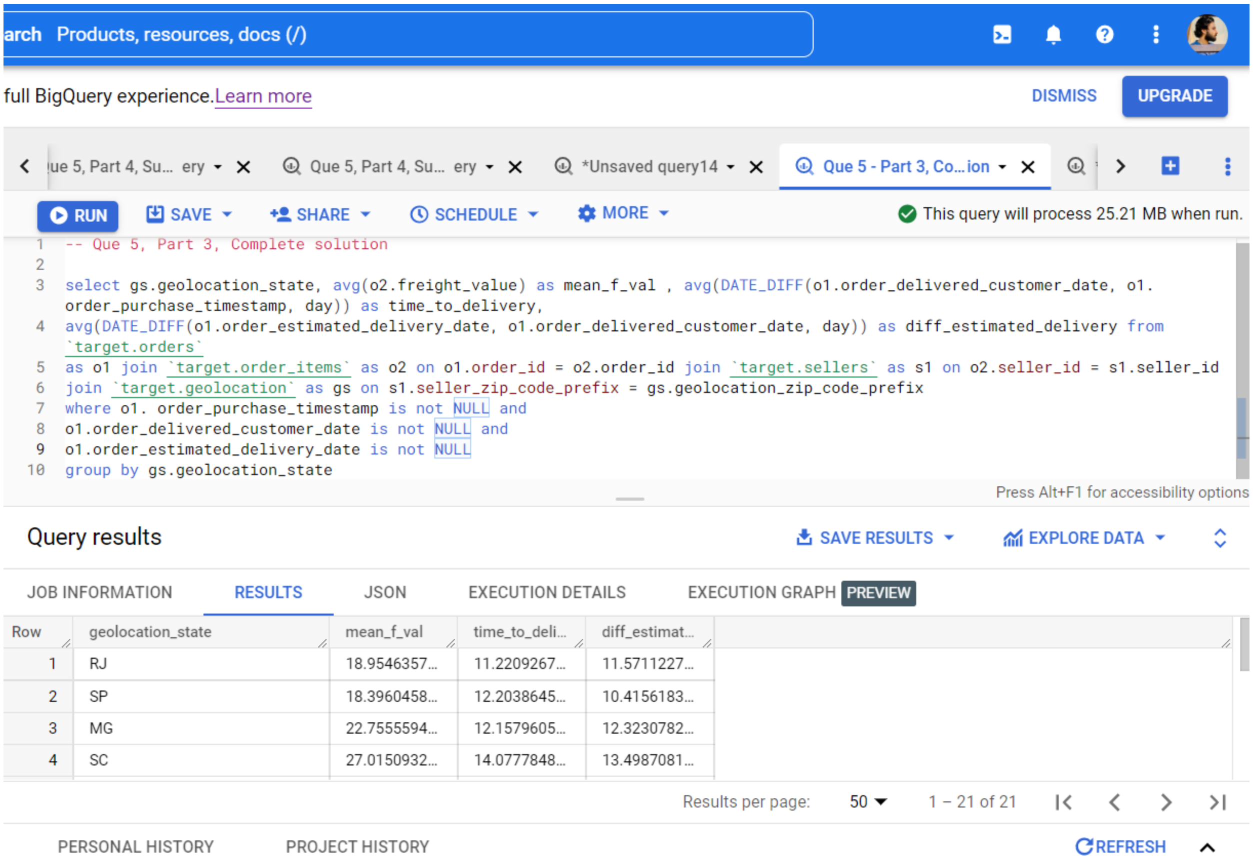
```sql
-- Que 5, Part 2 , create column

select DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) as time_to_delivery,
DATE_DIFF(order_delivered_customer_date, order_estimated_delivery_date, DAY) as diff_estimated_delivery  from `target.orders`
where order_purchase_timestamp is not NULL and
order_delivered_customer_date is not NULL and
order_estimated_delivery_date is not NULL
```

Query results

| Row | time_to_deli... | diff_estimat... |
|---|---|---|
| 1 | 30 | 12 |
| 2 | 30 | -28 |
| 3 | 35 | -16 |
| 4 | 30 | -1 |

1 – 50 of 96476

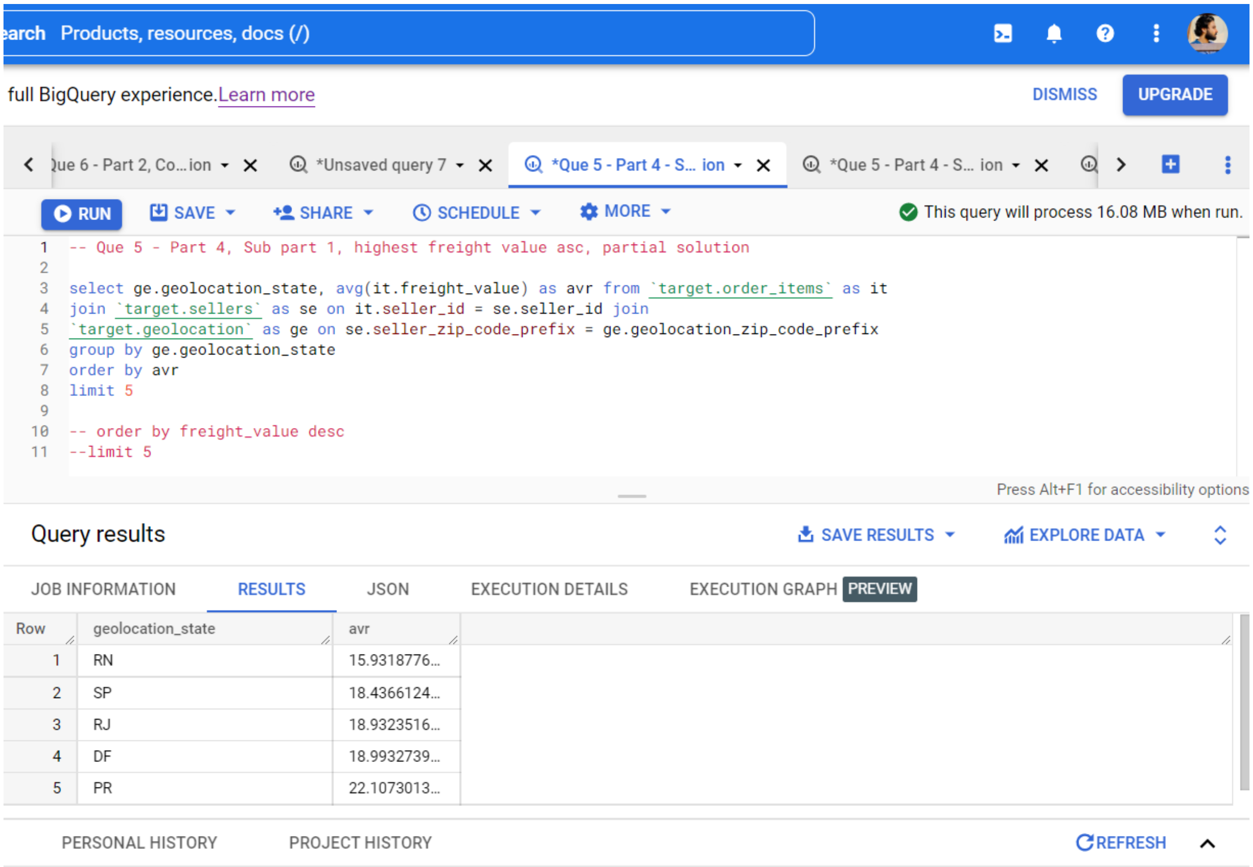
## Que 5 – Part 3



```
-- Que 5, Part 3, Complete solution

select gs.geolocation_state, avg(o2.freight_value) as mean_f_val , avg(DATE_DIFF(o1.order_delivered_customer_date, o1.
order_purchase_timestamp, day)) as time_to_delivery,
avg(DATE_DIFF(o1.order_estimated_delivery_date, o1.order_delivered_customer_date, day)) as diff_estimated_delivery from
`target.orders`
as o1 join `target.order_items` as o2 on o1.order_id = o2.order_id join `target.sellers` as s1 on o2.seller_id = s1.seller_id
join `target.geolocation` as gs on s1.seller_zip_code_prefix = gs.geolocation_zip_code_prefix
where o1. order_purchase_timestamp is not NULL and
o1.order_delivered_customer_date is not NULL and
o1.order_estimated_delivery_date is not NULL
group by gs.geolocation_state
```

| Row | geolocation_state | mean_f_val | time_to_deli... | diff_estimat... |
|---|---|---|---|---|
| 1 | RJ | 18.9546357... | 11.2209267... | 11.5711227... |
| 2 | SP | 18.3960458... | 12.2038645... | 10.4156183... |
| 3 | MG | 22.7555594... | 12.1579605... | 12.3230782... |
| 4 | SC | 27.0150932... | 14.0777848... | 13.4987081... |

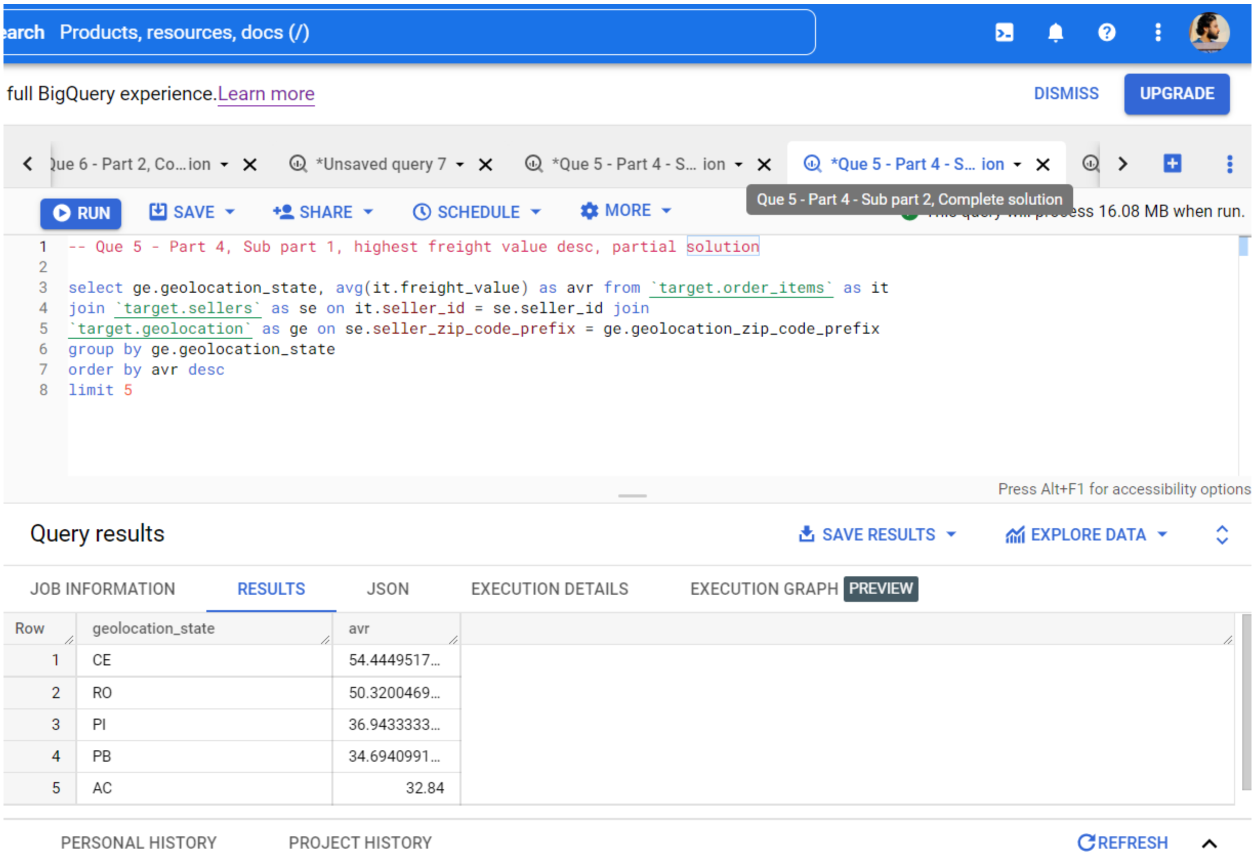
## Que 5 – Part 4 – Subpart 1 – Asc



```sql
-- Que 5 - Part 4, Sub part 1, highest freight value asc, partial solution

select ge.geolocation_state, avg(it.freight_value) as avr from `target.order_items` as it
join `target.sellers` as se on it.seller_id = se.seller_id join
`target.geolocation` as ge on se.seller_zip_code_prefix = ge.geolocation_zip_code_prefix
group by ge.geolocation_state
order by avr
limit 5

-- order by freight_value desc
--limit 5
```

| Row | geolocation_state | avr |
|---|---|---|
| 1 | RN | 15.9318776... |
| 2 | SP | 18.4366124... |
| 3 | RJ | 18.9323516... |
| 4 | DF | 18.9932739... |
| 5 | PR | 22.1073013... |

Que 5 – Part 4 – Subpart 1 – Desc

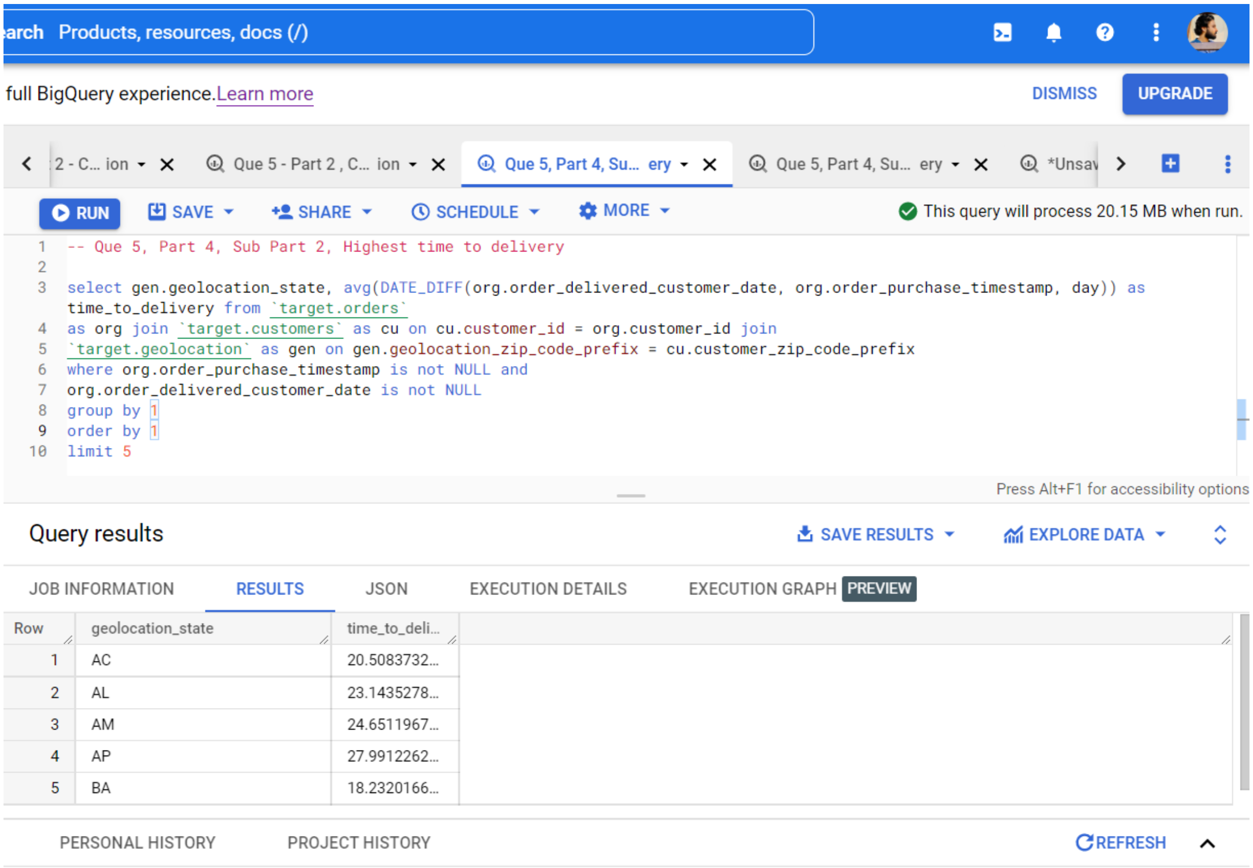


```
-- Que 5 - Part 4, Sub part 1, highest freight value desc, partial solution

select ge.geolocation_state, avg(it.freight_value) as avr from `target.order_items` as it
join `target.sellers` as se on it.seller_id = se.seller_id join
`target.geolocation` as ge on se.seller_zip_code_prefix = ge.geolocation_zip_code_prefix
group by ge.geolocation_state
order by avr desc
limit 5
```

Query results

| Row | geolocation_state | avr |
|---|---|---|
| 1 | CE | 54.4449517... |
| 2 | RO | 50.3200469... |
| 3 | PI | 36.9433333... |
| 4 | PB | 34.6940991... |
| 5 | AC | 32.84 |

Que 5 – Part 4 – Subpart 2 – High

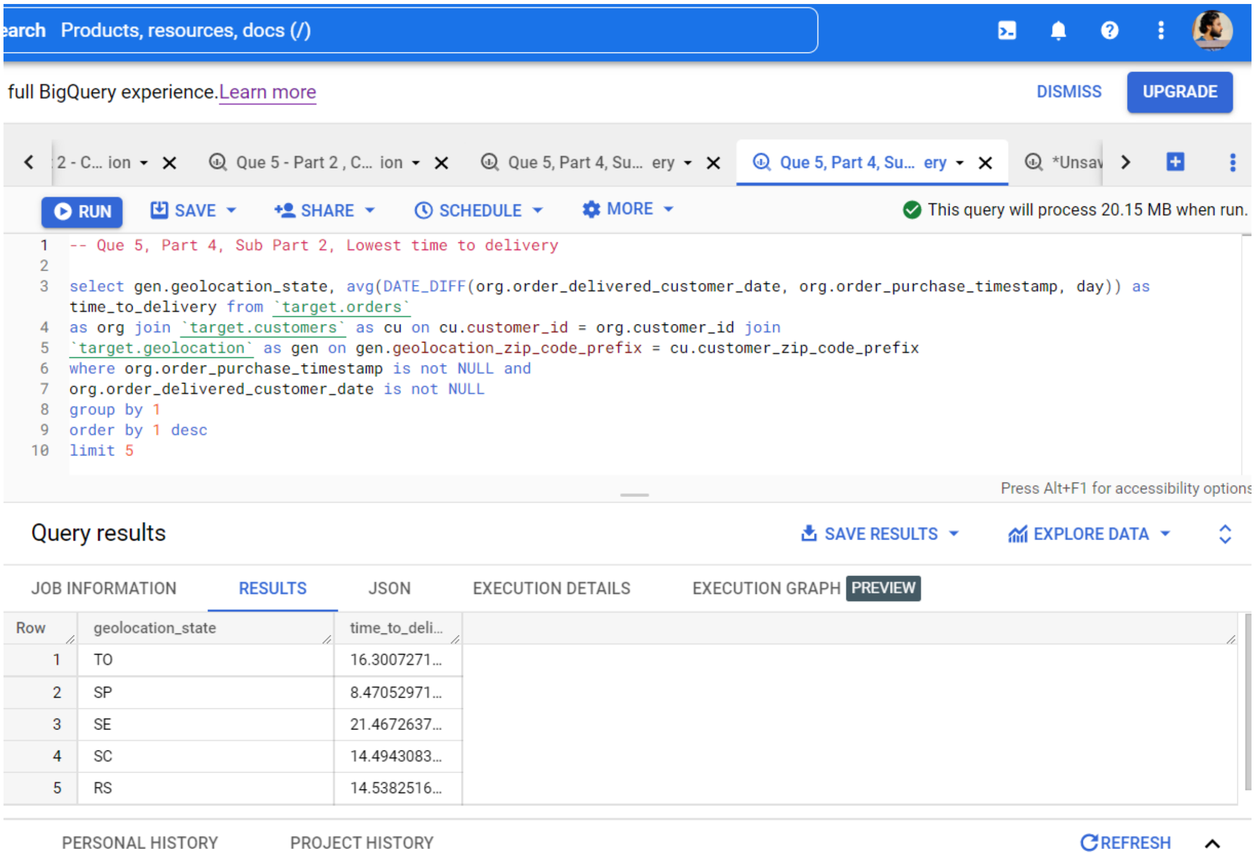


This query will process 20.15 MB when run.

```sql
-- Que 5, Part 4, Sub Part 2, Highest time to delivery

select gen.geolocation_state, avg(DATE_DIFF(org.order_delivered_customer_date, org.order_purchase_timestamp, day)) as
time_to_delivery from `target.orders`
as org join `target.customers` as cu on cu.customer_id = org.customer_id join
`target.geolocation` as gen on gen.geolocation_zip_code_prefix = cu.customer_zip_code_prefix
where org.order_purchase_timestamp is not NULL and
org.order_delivered_customer_date is not NULL
group by 1
order by 1
limit 5
```

| Row | geolocation_state | time_to_deli... |
|---|---|---|
| 1 | AC | 20.5083732... |
| 2 | AL | 23.1435278... |
| 3 | AM | 24.6511967... |
| 4 | AP | 27.9912262... |
| 5 | BA | 18.2320166... |

Que 5 – Part 4 _ Subpart 2 – Low



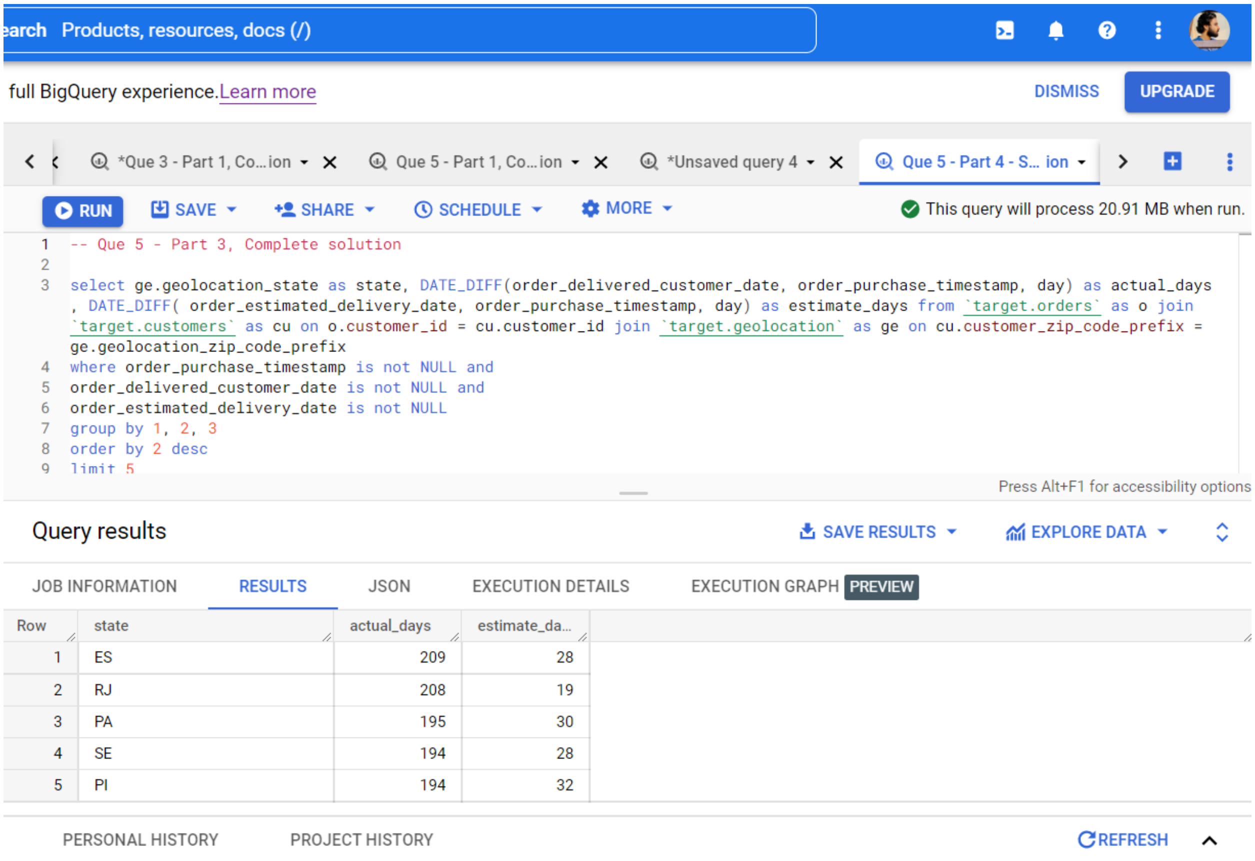
This query will process 20.15 MB when run.

```
-- Que 5, Part 4, Sub Part 2, Lowest time to delivery

select gen.geolocation_state, avg(DATE_DIFF(org.order_delivered_customer_date, org.order_purchase_timestamp, day)) as time_to_delivery from `target.orders`
as org join `target.customers` as cu on cu.customer_id = org.customer_id join
`target.geolocation` as gen on gen.geolocation_zip_code_prefix = cu.customer_zip_code_prefix
where org.order_purchase_timestamp is not NULL and
org.order_delivered_customer_date is not NULL
group by 1
order by 1 desc
limit 5
```

Query results

| Row | geolocation_state | time_to_deli... |
|---|---|---|
| 1 | TO | 16.3007271... |
| 2 | SP | 8.47052971... |
| 3 | SE | 21.4672637... |
| 4 | SC | 14.4943083... |
| 5 | RS | 14.5382516... |

## Que 5 – Part 4 – Subpart 3 – Fast delivery



full BigQuery experience.Learn more

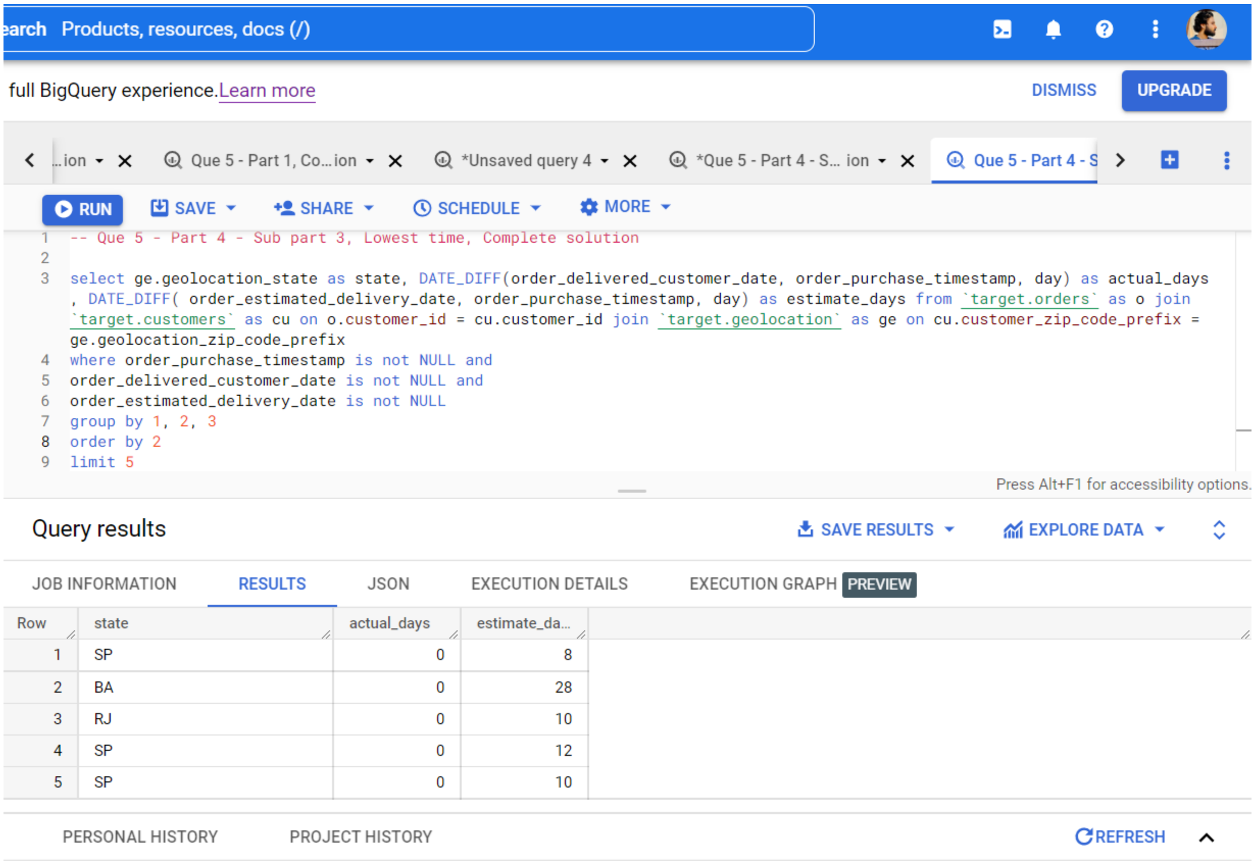
This query will process 20.91 MB when run.

```
-- Que 5 - Part 3, Complete solution

select ge.geolocation_state as state, DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) as actual_days
, DATE_DIFF( order_estimated_delivery_date, order_purchase_timestamp, day) as estimate_days from `target.orders` as o join
`target.customers` as cu on o.customer_id = cu.customer_id join `target.geolocation` as ge on cu.customer_zip_code_prefix =
ge.geolocation_zip_code_prefix
where order_purchase_timestamp is not NULL and
order_delivered_customer_date is not NULL and
order_estimated_delivery_date is not NULL
group by 1, 2, 3
order by 2 desc
limit 5
```

Press Alt+F1 for accessibility options

| Row | state | actual_days | estimate_da... |
|---|---|---|---|
| 1 | ES | 209 | 28 |
| 2 | RJ | 208 | 19 |
| 3 | PA | 195 | 30 |
| 4 | SE | 194 | 28 |
| 5 | PI | 194 | 32 |

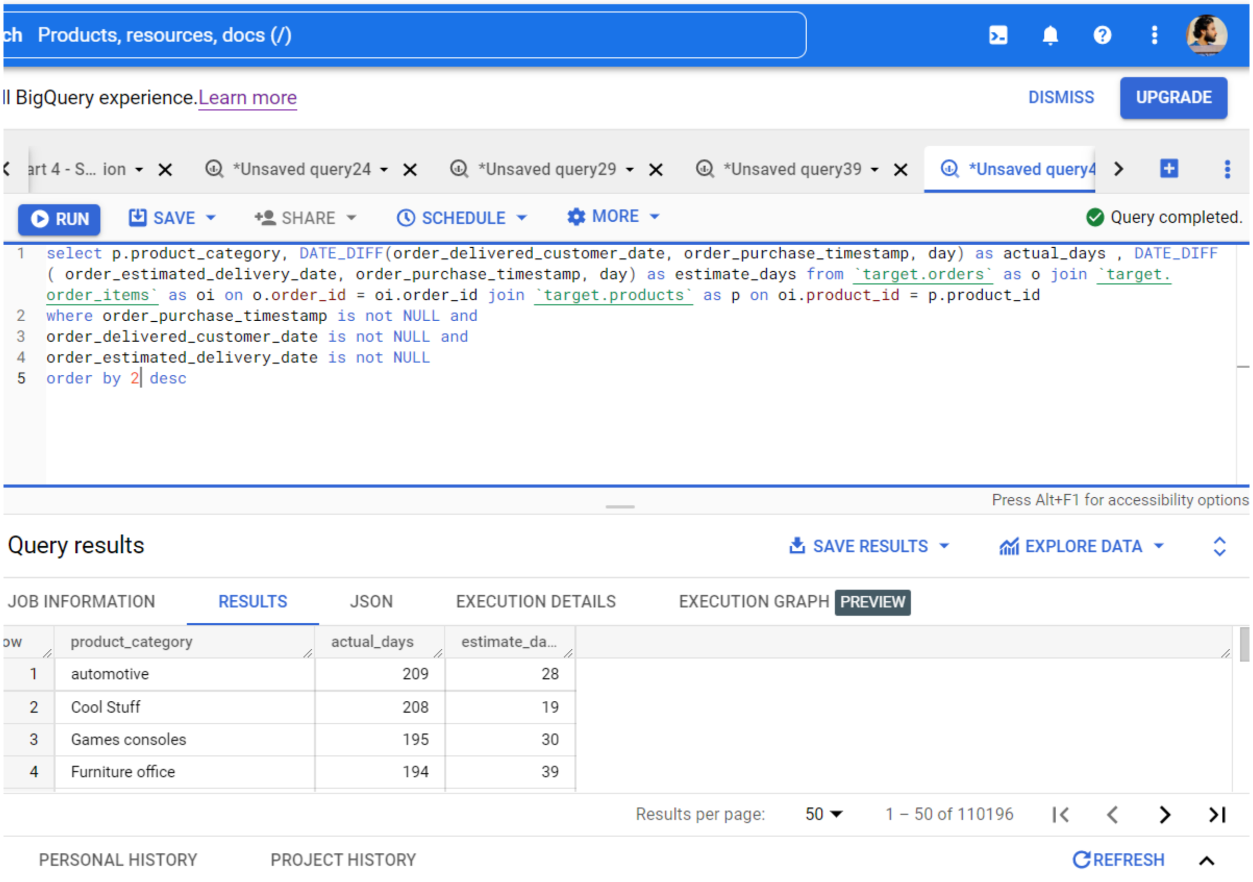
## Que 5 – Part 4 – Subpart 3 – Slow delivery



```
-- Que 5 - Part 4 - Sub part 3, Lowest time, Complete solution

select ge.geolocation_state as state, DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) as actual_days
, DATE_DIFF( order_estimated_delivery_date, order_purchase_timestamp, day) as estimate_days from `target.orders` as o join
`target.customers` as cu on o.customer_id = cu.customer_id join `target.geolocation` as ge on cu.customer_zip_code_prefix =
ge.geolocation_zip_code_prefix
where order_purchase_timestamp is not NULL and
order_delivered_customer_date is not NULL and
order_estimated_delivery_date is not NULL
group by 1, 2, 3
order by 2
limit 5
```

| Row | state | actual_days | estimate_da... |
|---|---|---|---|
| 1 | SP | 0 | 8 |
| 2 | BA | 0 | 28 |
| 3 | RJ | 0 | 10 |
| 4 | SP | 0 | 12 |
| 5 | SP | 0 | 10 |

## Insights



Query completed.

```
select p.product_category, DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) as actual_days , DATE_DIFF
( order_estimated_delivery_date, order_purchase_timestamp, day) as estimate_days from `target.orders` as o join `target.
order_items` as oi on o.order_id = oi.order_id join `target.products` as p on oi.product_id = p.product_id
where order_purchase_timestamp is not NULL and
order_delivered_customer_date is not NULL and
order_estimated_delivery_date is not NULL
order by 2 desc
```

| ow | product_category | actual_days | estimate_da... |
|---|---|---|---|
| 1 | automotive | 209 | 28 |
| 2 | Cool Stuff | 208 | 19 |
| 3 | Games consoles | 195 | 30 |
| 4 | Furniture office | 194 | 39 |

Results per page: 50 1 – 50 of 110196

It is clear that the top 5 products which incurs maximum time are from the Category:

- Automobile
- Cool Stuff
- Games consoles
- Furniture office
- Musical Instruments

Also, the products which are delivered with in one day are from the Category:

- Babies
- Housewares
- Computer accessories
- Toys
- Watches present

The quickest order delivering states are **Sao Paulo** and **Rio de Janeiro**, with maximum actual delivery time is **Espirito Santo** from South Brazil.

Top 5 states with highest average freight value are:

- Rio Grande do Norte
- Sao Paulo
- Rio de Janeiro
- Federal District
- Parana

Top 5 states with lowest average freight value are:

- Ceara
- Piaui
- Parabia
- Acre
- Espirito Santo

Top 5 states with highest time to delivery are:

- Espirito Santo
- Rio de Janerio
- Para
- Southeastern region of Brazil
- Piaui

Top 5 states with lowest time to delivery are:

- Sao Paulo
- Bahia
- Rio de Janerio
- Parana
- Santa Catarina

Top 5 states with highest time to delivery are:

- Acre
- Federative units of Brazil
- Amazonas
- Amapa
- Bahia

Top 5 states with lowest time to delivery are:

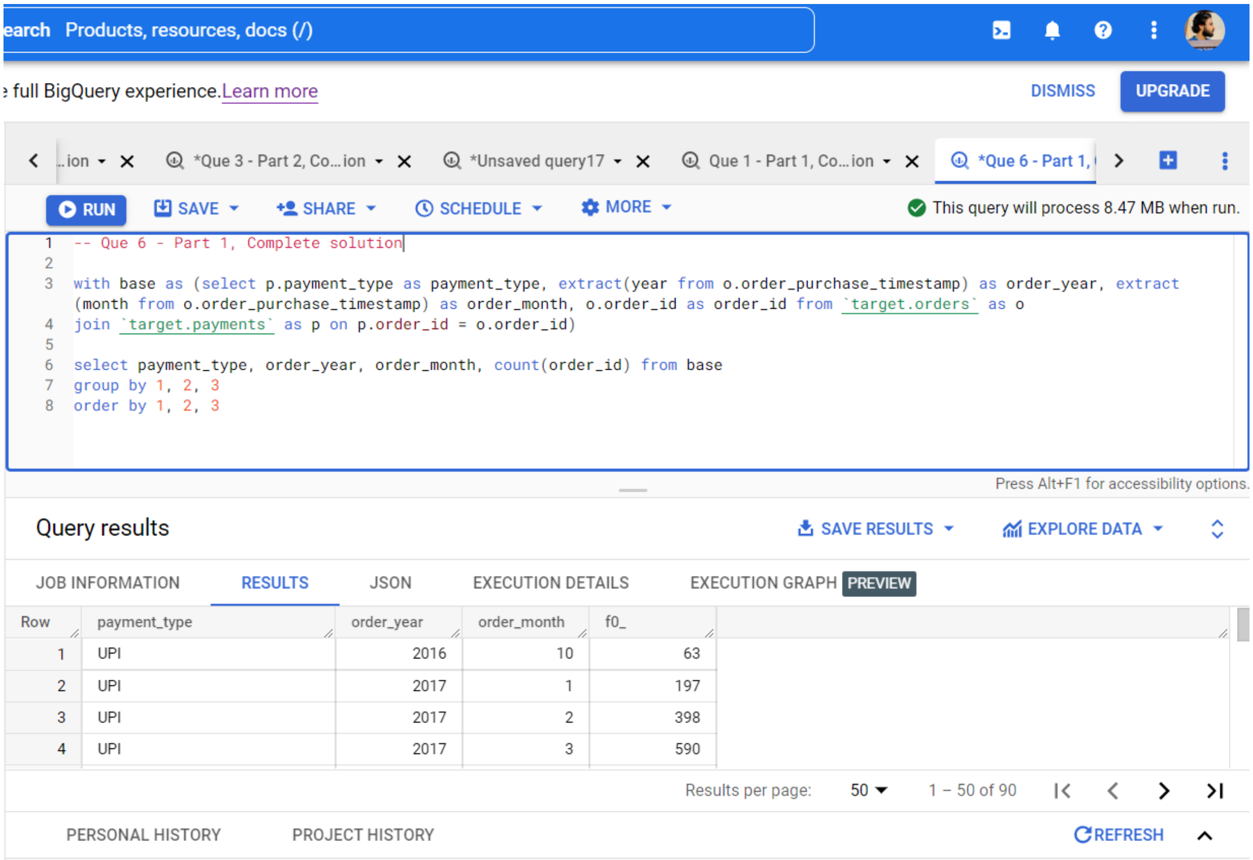
- Sao Paulo

- Southeastern region of Brazil
- Santa Catarina
- Rio de Janerio
- Federative units of Brazil

6. Payment type analysis:

1. Month over Month count of orders for different payment types
2. Count of orders based on the no. of payment installments

Que 6 – Part 1

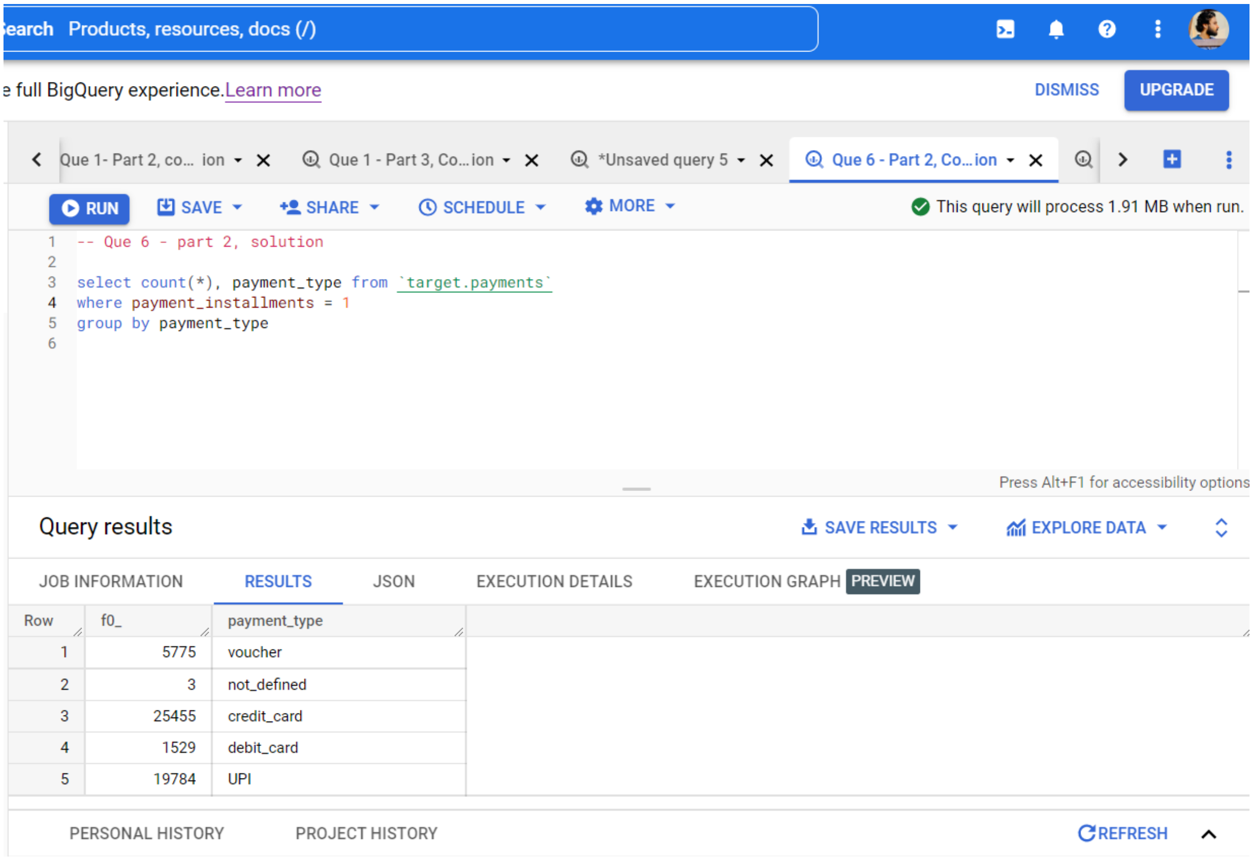


This query will process 8.47 MB when run.

```
-- Que 6 - Part 1, Complete solution

with base as (select p.payment_type as payment_type, extract(year from o.order_purchase_timestamp) as order_year, extract
(month from o.order_purchase_timestamp) as order_month, o.order_id as order_id from `target.orders` as o
join `target.payments` as p on p.order_id = o.order_id)

select payment_type, order_year, order_month, count(order_id) from base
group by 1, 2, 3
order by 1, 2, 3
```

| Row | payment_type | order_year | order_month | f0_ |
|---|---|---|---|---|
| 1 | UPI | 2016 | 10 | 63 |
| 2 | UPI | 2017 | 1 | 197 |
| 3 | UPI | 2017 | 2 | 398 |
| 4 | UPI | 2017 | 3 | 590 |

Que 6 – Part 2



```
-- Que 6 - part 2, solution

select count(*), payment_type from `target.payments`
where payment_installments = 1
group by payment_type
```

| Row | f0_ | payment_type |
|---|---|---|
| 1 | 5775 | voucher |
| 2 | 3 | not_defined |
| 3 | 25455 | credit_card |
| 4 | 1529 | debit_card |
| 5 | 19784 | UPI |

It is very evident from the table that **UPI** transactions are increasing with increase in time surpassing **debit card** payments.

## Recommendations:

As per the analysis of result set:

- The region that fall short in most area is **Espirito Santo,** as the most common problem being the delivery time. Located at a costal site of Brazil, a warehouse between Minas Gerais and Bahia might be able to solve a part of the issue, considering the decline in number of sellers.
- Also the highest increase in UPI transactions as indicated by the result set is from the Federative units of Brazil.
- Also on interpreting the results of graphs shown above, a reason of drop in count of orders after Nov, 2017 can be due to inevitable increase in overall cost of order; upon relative analysis.

# Thanks