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

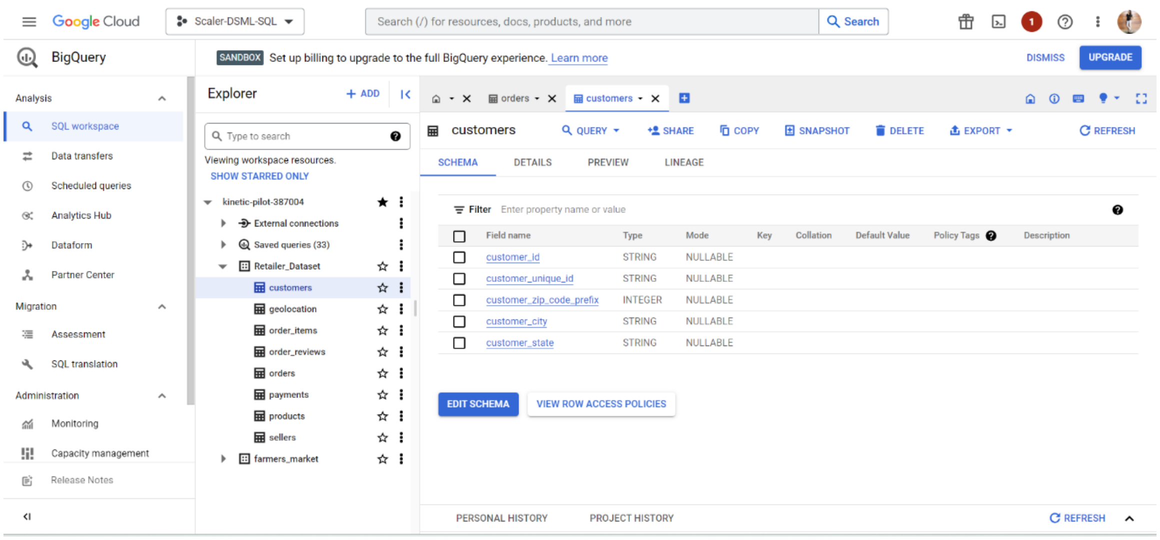
    1) Data type of all columns in the "customers" table.

Answer:

1.Click on SQL workpace->Explorer-> table_name-> schema
2. SELECT * FROM myDataset.INFORMATION_SCHEMA.TABLES;

Field_name | DataType
1.customer_id | String
2.customer_unique_id | String
3.customer_zip_code_prefix | Integer
4.customer_city | String
5.customer_state | String

Snapshot:



customers

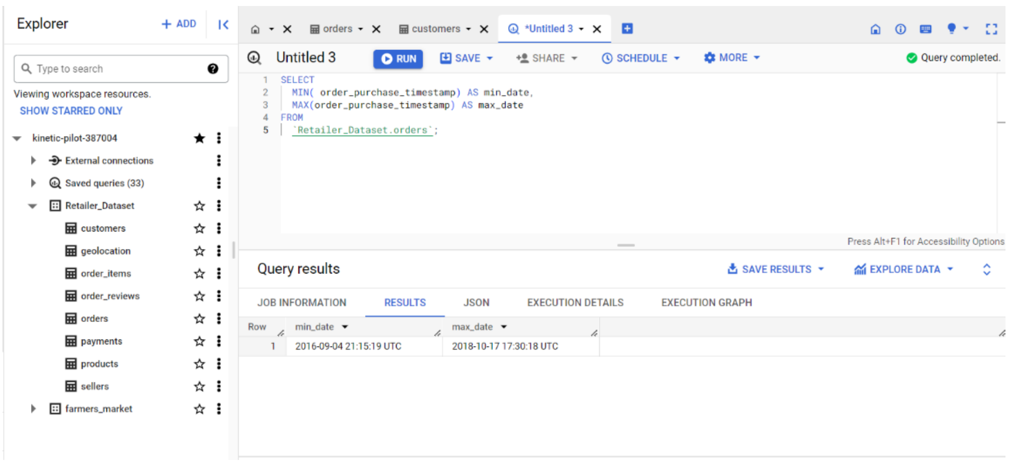
Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
customer_id	STRING	NULLABLE					
customer_unique_id	STRING	NULLABLE					
customer_zip_code_prefix	INTEGER	NULLABLE					
customer_city	STRING	NULLABLE					
customer_state	STRING	NULLABLE					

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

Answer:

```
SELECT
  MIN( order_purchase_timestamp) AS min_date,
  MAX(order_purchase_timestamp) AS max_date
FROM
  `Retailer_Dataset.orders`;
```

Snapshot:



```
SELECT
  MIN( order_purchase_timestamp) AS min_date,
  MAX(order_purchase_timestamp) AS max_date
FROM
  `Retailer_Dataset.orders`;
```

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

Insights / Recommendation:

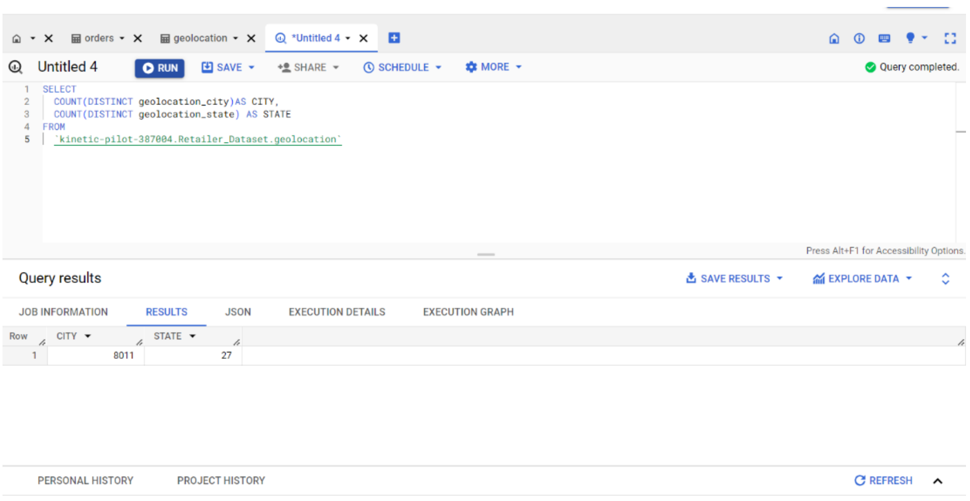
1. Here date range is 2016-09-04 to 2018-10-17 that means we have not fully two years data from 2016-Jan to 2018-Dec.

3) Count the number of Cities and States in our dataset.

Answer:

```
SELECT
  COUNT(DISTINCT geolocation_city)AS CITY,
  COUNT(DISTINCT geolocation_state) AS STATE
FROM
  `kinetic-pilot-387004.Retailer_Dataset.geolocation`
```

Snapshot:



```
SELECT
  COUNT(DISTINCT geolocation_city)AS CITY,
  COUNT(DISTINCT geolocation_state) AS STATE
FROM
  `kinetic-pilot-387004.Retailer_Dataset.geolocation`
```

Row	CITY	STATE
1	8011	27

Insight/Recommendation:

With this data we can find that how many different state and different city is there where we can have customers and sellers we can connect them with each other.
I have recommendation here we can also get insight of each state means each state how many city are there which have service with this we can find which state have strong network of retail and which state we still have to work on.

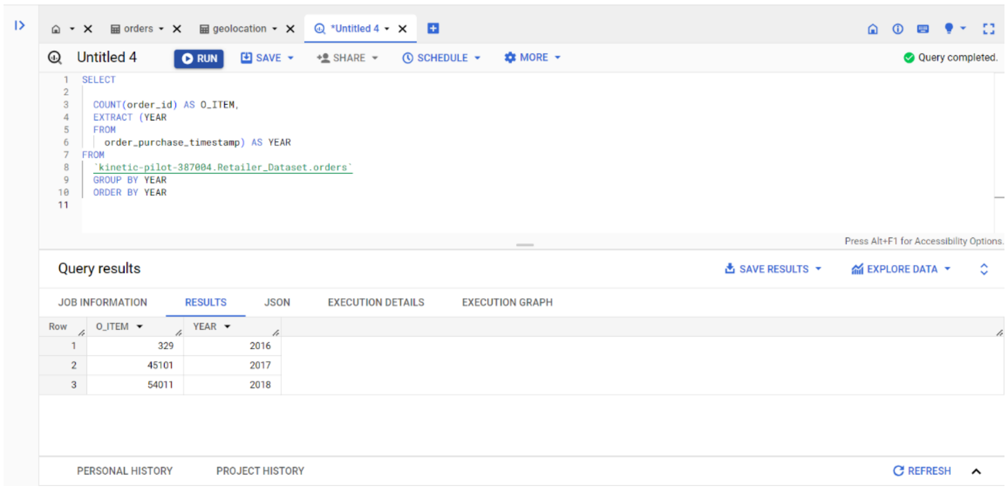
2. In-depth Exploration:

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

Answer:

```
SELECT
  COUNT(order_id) AS O_ITEM,
  EXTRACT (YEAR
  FROM
    order_purchase_timestamp) AS YEAR
FROM
  `kinetic-pilot-387004.Retailer_Dataset.orders`
  GROUP BY YEAR
  ORDER BY YEAR
```

Snapshot:



```
SELECT

  COUNT(order_id) AS O_ITEM,
  EXTRACT (YEAR
  FROM
    order_purchase_timestamp) AS YEAR
FROM
  `kinetic-pilot-387004.Retailer_Dataset.orders`
  GROUP BY YEAR
  ORDER BY YEAR
```

Row	O_ITEM	YEAR
1	329	2016
2	45101	2017
3	54011	2018

Insight/Recommendation:

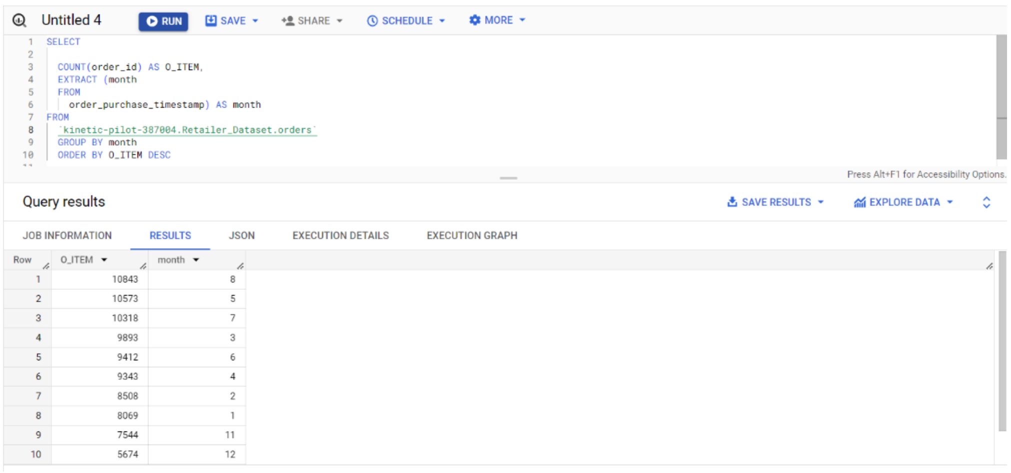
With the data we can observe that item order increase with year passing.

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

Answer:

```
SELECT
  COUNT(order_id) AS O_ITEM,
  EXTRACT (month
  FROM
    order_purchase_timestamp) AS month
FROM
  `kinetic-pilot-387004.Retailer_Dataset.orders`
  GROUP BY month
  ORDER BY O_ITEM DESC
```

Snapshot:



```
SELECT

  COUNT(order_id) AS O_ITEM,
  EXTRACT (month
  FROM
    order_purchase_timestamp) AS month
FROM
  `kinetic-pilot-387004.Retailer_Dataset.orders`
  GROUP BY month
  ORDER BY O_ITEM DESC
```

Query results

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

Insight/Recommendation:

Yes, We can see according to data that people are more buying in August, May, July and less buy in November and December. It give us peoples mind set and timing when they mostly buy.

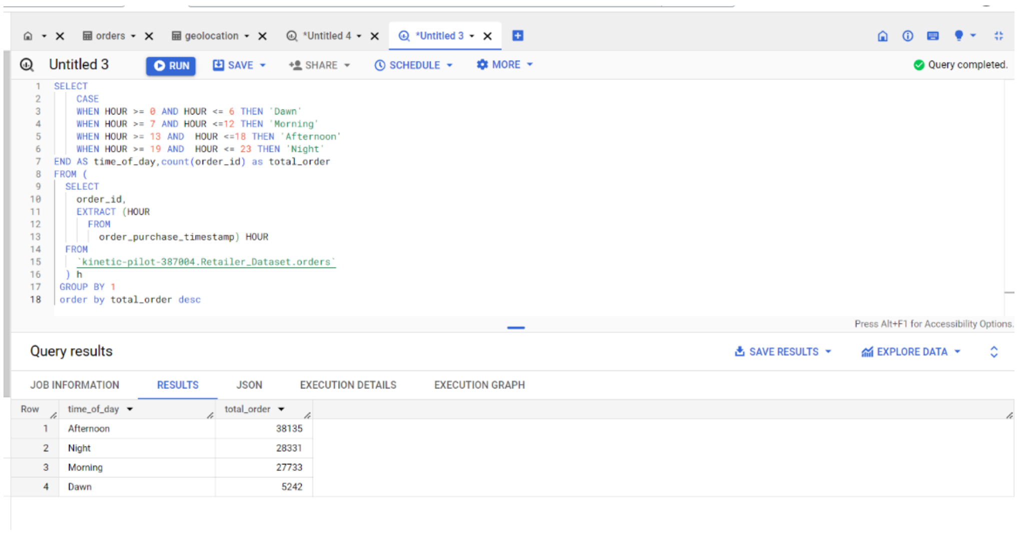
3) During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- 0-6 hrs : Dawn
- 7-12 hrs : Mornings
- 13-18 hrs : Afternoon
- 19-23 hrs : Night

Answer:

```
SELECT
    CASE
    WHEN HOUR >= 0 AND HOUR <= 6 THEN 'Dawn'
    WHEN HOUR >= 7 AND HOUR <=12 THEN 'Morning'
    WHEN HOUR >= 13 AND  HOUR <=18 THEN 'Afternoon'
    WHEN HOUR >= 19 AND  HOUR <= 23 THEN 'Night'
END AS time_of_day,count(order_id) as total_order
FROM (
  SELECT
    order_id,
    EXTRACT (HOUR
      FROM
        order_purchase_timestamp) HOUR
  FROM
    `kinetic-pilot-387004.Retailer_Dataset.orders`
  ) h
 GROUP BY 1
 order by total_order desc
```

Snapshot:



Untitled 3

Query completed.

```
SELECT
    CASE
    WHEN HOUR >= 0 AND HOUR <= 6 THEN 'Dawn'
    WHEN HOUR >= 7 AND HOUR <=12 THEN 'Morning'
    WHEN HOUR >= 13 AND  HOUR <=18 THEN 'Afternoon'
    WHEN HOUR >= 19 AND  HOUR <= 23 THEN 'Night'
END AS time_of_day,count(order_id) as total_order
FROM (
  SELECT
    order_id,
    EXTRACT (HOUR
      FROM
        order_purchase_timestamp) HOUR
  FROM
    `kinetic-pilot-387004.Retailer_Dataset.orders`
  ) h
 GROUP BY 1
 order by total_order desc
```

Query results

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

Insight/Recommendation:

With the data we can find that people are more buying in Afternoon and less buy in dawn.
In general we used to thought people are buy in night but with data we can find that people have buying tendency in day is Afternoon

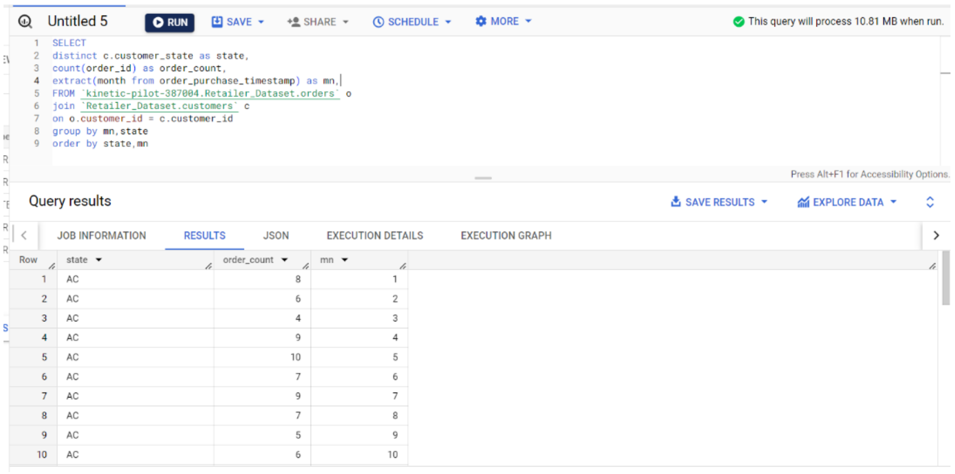
3. Evolution of E-commerce orders in the Brazil region:

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

Answer:

```
select
distinct c.customer_state as state,
count(order_id) as order_count,
extract(month from order_purchase_timestamp) as mn,
from `kinetic-pilot-387004.Retailer_Dataset.orders` o
join `Retailer_Dataset.customers` c
on o.customer_id = c.customer_id
group by mn,state
order by state,mn
```

Snapshot:



Untitled 5

This query will process 10.81 MB when run.

```
SELECT
distinct c.customer_state as state,
count(order_id) as order_count,
extract(month from order_purchase_timestamp) as mn,
FROM `kinetic-pilot-387004.Retailer_Dataset.orders` o
join `Retailer_Dataset.customers` c
on o.customer_id = c.customer_id
group by mn,state
order by state,mn
```

Query results

Row	state	order_count	mn
1	AC	8	1
2	AC	6	2
3	AC	4	3
4	AC	9	4
5	AC	10	5
6	AC	7	6
7	AC	9	7
8	AC	7	8
9	AC	5	9
10	AC	6	10

Insight/Recommendation:

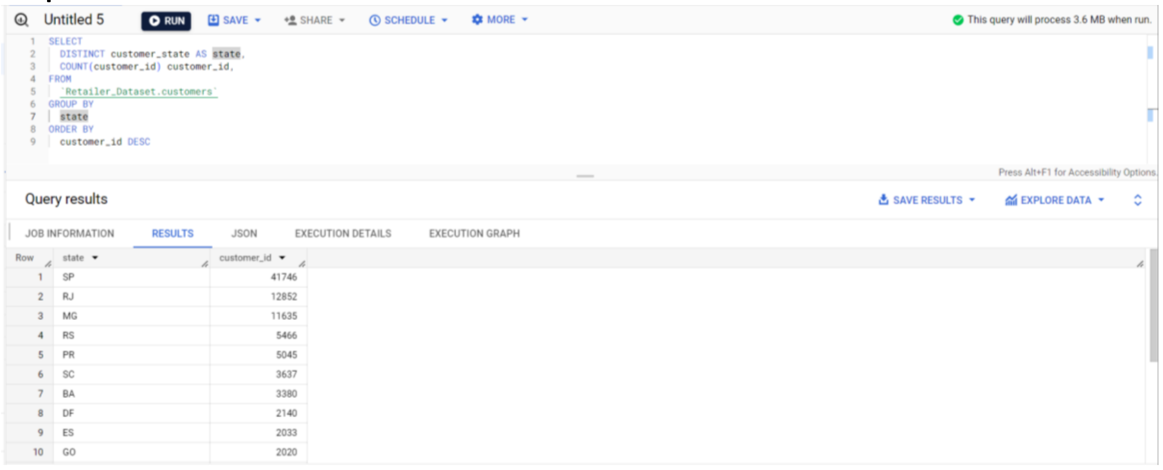
Here we have data for each state for each month how many order placed. With this we can observe different states people's tendency to which month they more. And if we go deeper with data we can find reason for this.

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

Answer:

```
SELECT
DISTINCT customer_state AS state,
COUNT(customer_id) customer_id,
FROM
`Retailer_Dataset.customers`
GROUP BY
state
ORDER BY
customer_id DESC
```

Snapshot:



This query will process 3.6 MB when run.

```
SELECT
  DISTINCT customer_state AS state,
  COUNT(customer_id) customer_id,
FROM
  `Retailer_Dataset.customers`
GROUP BY
  state
ORDER BY
  customer_id DESC
```

Press Alt+F1 for Accessibility Options.

Row	state	customer_id
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

Insight/Recommendation:

With the data you can find that “SP” state have more customers and “GO” state have less customers. With this data we get to know in which state we have more customers and which state have less. In which state we have to do more work to attract customers.

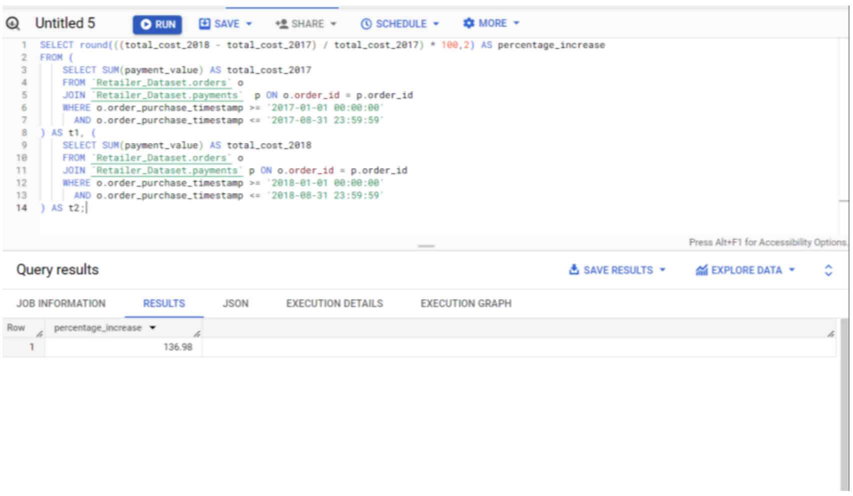
4. Impact on Economy: Analyse the money movement by e-commerce by looking at order prices, freight and others.

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.

Answer:

```
SELECT round(((total_cost_2018 - total_cost_2017) / total_cost_2017) * 100,2) AS percentage_increase
FROM (
    SELECT SUM(payment_value) AS total_cost_2017
    FROM `Retailer_Dataset.orders` o
    JOIN `Retailer_Dataset.payments`  p ON o.order_id = p.order_id
    WHERE o.order_purchase_timestamp >= '2017-01-01 00:00:00'
      AND o.order_purchase_timestamp <= '2017-08-31 23:59:59'
) AS t1, (
    SELECT SUM(payment_value) AS total_cost_2018
    FROM `Retailer_Dataset.orders` o
    JOIN `Retailer_Dataset.payments` p ON o.order_id = p.order_id
    WHERE o.order_purchase_timestamp >= '2018-01-01 00:00:00'
      AND o.order_purchase_timestamp <= '2018-08-31 23:59:59'
) AS t2;
```

Snapshot:



```
SELECT round(((total_cost_2018 - total_cost_2017) / total_cost_2017) * 100,2) AS percentage_increase
FROM (
    SELECT SUM(payment_value) AS total_cost_2017
    FROM `Retailer_Dataset.orders` o
    JOIN `Retailer_Dataset.payments`  p ON o.order_id = p.order_id
    WHERE o.order_purchase_timestamp >= '2017-01-01 00:00:00'
      AND o.order_purchase_timestamp <= '2017-08-31 23:59:59'
) AS t1, (
    SELECT SUM(payment_value) AS total_cost_2018
    FROM `Retailer_Dataset.orders` o
    JOIN `Retailer_Dataset.payments` p ON o.order_id = p.order_id
    WHERE o.order_purchase_timestamp >= '2018-01-01 00:00:00'
      AND o.order_purchase_timestamp <= '2018-08-31 23:59:59'
) AS t2;
```

Row	percentage_increase
1	136.98

Insight/Recommendation:

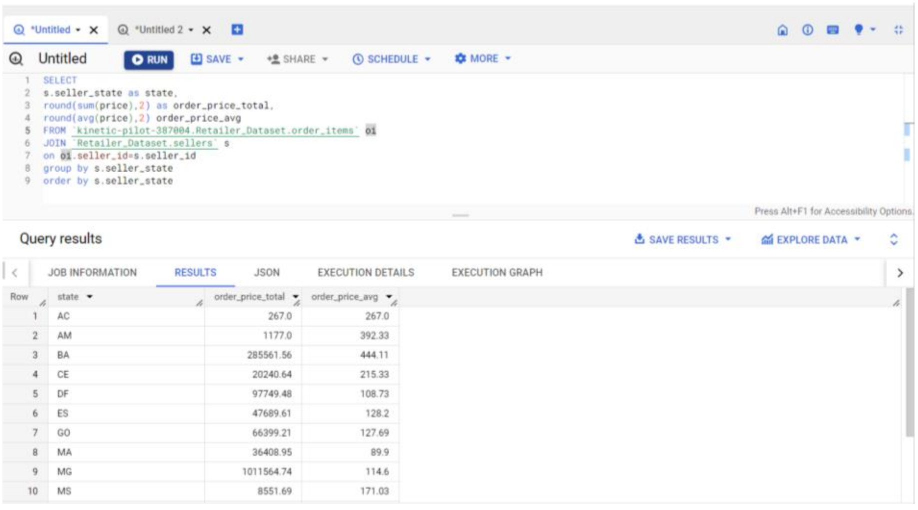
The data showing increase % of payment value from 2017 to 2018. Which means people's buying capacity also increase and more people are buying from company in 2018 compare to 2017.

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

Answer:

```
select
s.seller_state as state,
round(sum(price),2) as order_price_total,
round(avg(price),2) order_price_avg
FROM `kinetic-pilot-387004.Retailer_Dataset.order_items` oi
JOIN `Retailer_Dataset.sellers` s
on oi.seller_id=s.seller_id
group by s.seller_state
order by s.seller_state
```

Snapshot:



```
SELECT
s.seller_state as state,
round(sum(price),2) as order_price_total,
round(avg(price),2) order_price_avg
FROM `kinetic-pilot-387004.Retailer_Dataset.order_items` oi
JOIN `Retailer_Dataset.sellers` s
on oi.seller_id=s.seller_id
group by s.seller_state
order by s.seller_state
```

Row	state	order_price_total	order_price_avg
1	AC	267.0	267.0
2	AM	1177.0	392.33
3	BA	285561.56	444.11
4	CE	20240.64	215.33
5	DF	97749.48	108.73
6	ES	47689.61	128.2
7	GO	66399.21	127.69
8	MA	36408.95	89.9
9	MG	1011564.74	114.6
10	MS	8551.69	171.03

Insight/Recommendation:

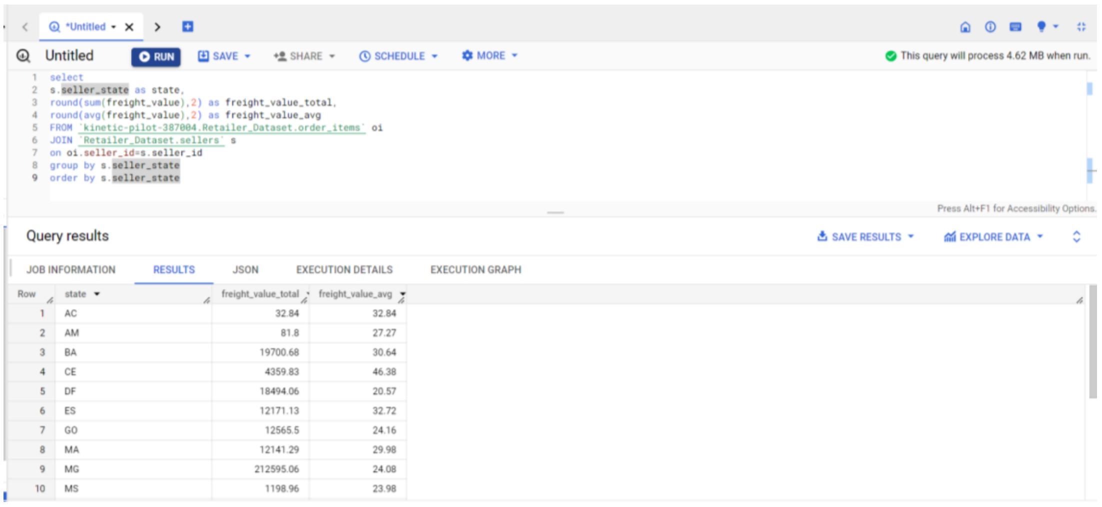
With the data we can get to know for each state about their total order price and average order price. With that we can observe which state are buying more with their average price.

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

Answer:

```
select
s.seller_state as state,
round(sum(freight_value),2) as freight_value_total,
round(avg(freight_value),2) as freight_value_avg
FROM `kinetic-pilot-387004.Retailer_Dataset.order_items` oi
JOIN `Retailer_Dataset.sellers` s
on oi.seller_id=s.seller_id
group by s.seller_state
order by s.seller_state
```

Snapshot:



```
select
s.seller_state as state,
round(sum(freight_value),2) as freight_value_total,
round(avg(freight_value),2) as freight_value_avg
FROM `kinetic-pilot-387004.Retailer_Dataset.order_items` oi
JOIN `Retailer_Dataset.sellers` s
on oi.seller_id=s.seller_id
group by s.seller_state
order by s.seller_state
```

Row	state	freight_value_total	freight_value_avg
1	AC	32.84	32.84
2	AM	81.8	27.27
3	BA	19700.68	30.64
4	CE	4359.83	46.38
5	DF	18494.06	20.57
6	ES	12171.13	32.72
7	GO	12565.5	24.16
8	MA	12141.29	29.98
9	MG	212595.06	24.08
10	MS	1198.96	23.98

Insight/Recommendation:

Every order placed from different state and deliver from different state so their freight value is different from each other so with this data we can find which state have more freight value and it will more add delivery charges.

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.
Do this in a single query.

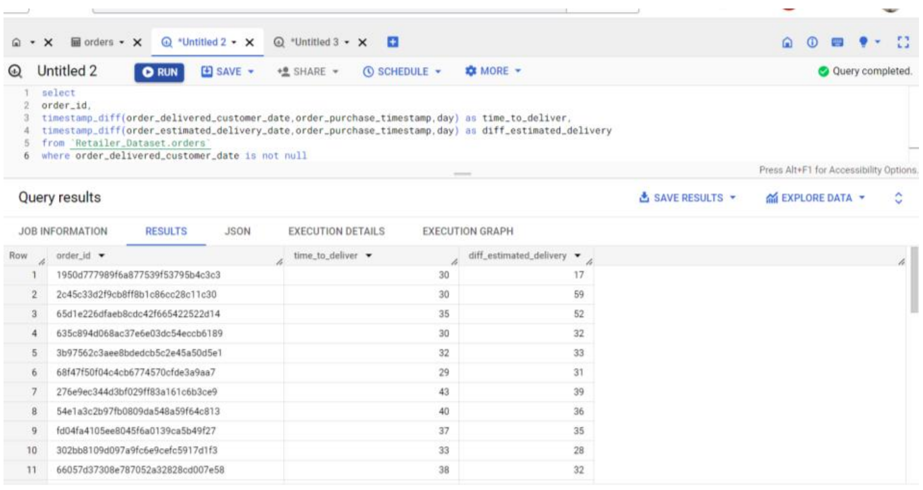
You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

- **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
- **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date

Answer:

```
select
order_id,
timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day) as time_to_deliver,
timestamp_diff(order_estimated_delivery_date,order_purchase_timestamp,day) as diff_estimated_delivery
from `Retailer_Dataset.orders`
where order_delivered_customer_date is not null
```

Snapshot:



Untitled 2 — Query completed.

```
select
order_id,
timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day) as time_to_deliver,
timestamp_diff(order_estimated_delivery_date,order_purchase_timestamp,day) as diff_estimated_delivery
from `Retailer_Dataset.orders`
where order_delivered_customer_date is not null
```

Query results

Row	order_id	time_to_deliver	diff_estimated_delivery
1	1950d777989f6a877539f53795b4c3c3	30	17
2	2c45c33d2f9cb8ff8b1c86cc28c11c30	30	59
3	65d1e226dfaeb8cdc42f665422522d14	35	52
4	635c894d068ac37e6e03dc54eccb6189	30	32
5	3b97562c3aee8bdedcb5c2e45a50d5e1	32	33
6	68f47f50f04c4cb6774570cfde3a9aa7	29	31
7	276e9ec344d3bf029ff83a161c6b3ce9	43	39
8	54e1a3c2b97fb0809da548a59f64c813	40	36
9	fd04fa4105ee8045f6a0139ca5b49f27	37	35
10	302bb8109d097a9fc6e9cefc5917d1f3	33	28
11	66057d37308e787052a32828cd007e58	38	32

Insight/Recommendation:

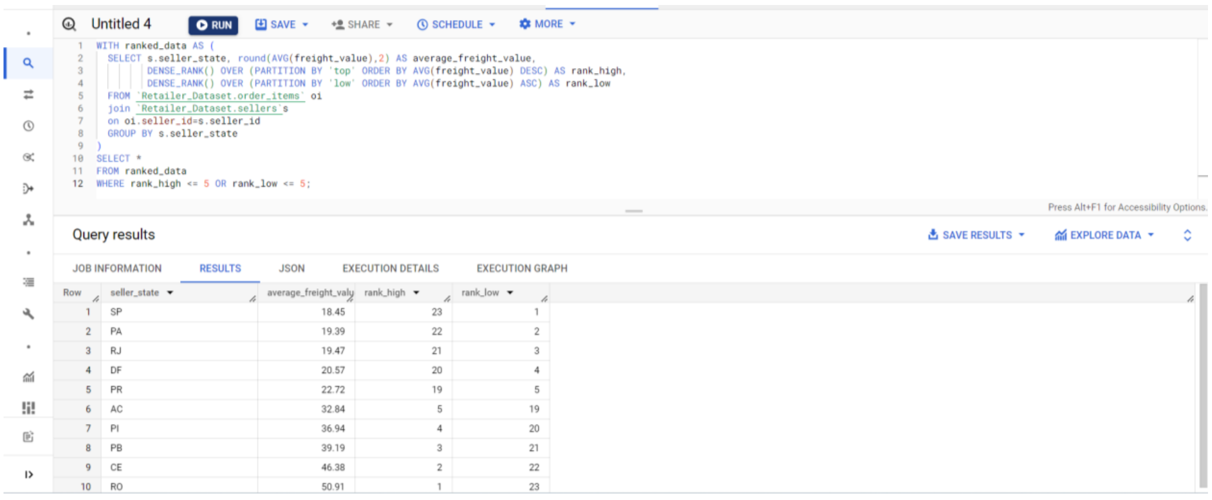
In data we can find delivery time and difference between actually delivery and estimated delivery. With this data we can find which order delivered late after estimated date and how many days I was late.
Here, In data there is in customer's delivered date is null in for some orders it indicate that order is shipped from sellers but not reached at customer's It has maybe customer cancel order after created it.
So In this particular query I used "not null condition" to only find delivered orders.

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

Answer:

```
WITH ranked_data AS (
  SELECT s.seller_state, round(AVG(freight_value),2) AS average_freight_value,
         DENSE_RANK() OVER (PARTITION BY 'top' ORDER BY AVG(freight_value) DESC) AS rank_high,
         DENSE_RANK() OVER (PARTITION BY 'low' ORDER BY AVG(freight_value) ASC) AS rank_low
  FROM `Retailer_Dataset.order_items` oi
  join `Retailer_Dataset.sellers`s
  on oi.seller_id=s.seller_id
  GROUP BY s.seller_state
)
SELECT *
FROM ranked_data
WHERE rank_high <= 5 OR rank_low <= 5;
```

Snapshot:



Untitled 4

```
WITH ranked_data AS (
  SELECT s.seller_state, round(AVG(freight_value),2) AS average_freight_value,
        DENSE_RANK() OVER (PARTITION BY 'top' ORDER BY AVG(freight_value) DESC) AS rank_high,
        DENSE_RANK() OVER (PARTITION BY 'low' ORDER BY AVG(freight_value) ASC) AS rank_low
  FROM `Retailer_Dataset.order_items` oi
  join `Retailer_Dataset.sellers`s
  on oi.seller_id=s.seller_id
  GROUP BY s.seller_state
)
SELECT *
FROM ranked_data
WHERE rank_high <= 5 OR rank_low <= 5;
```

Query results

Row	seller_state	average_freight_valu	rank_high	rank_low
1	SP	18.45	23	1
2	PA	19.39	22	2
3	RJ	19.47	21	3
4	DF	20.57	20	4
5	PR	22.72	19	5
6	AC	32.84	5	19
7	PI	36.94	4	20
8	PB	39.19	3	21
9	CE	46.38	2	22
10	RO	50.91	1	23

Insight/Recommendation:

Here, in data we found top 5 highest and 5 lowest average freight value of states.
With this we can find that which state have more freight value and which is not.

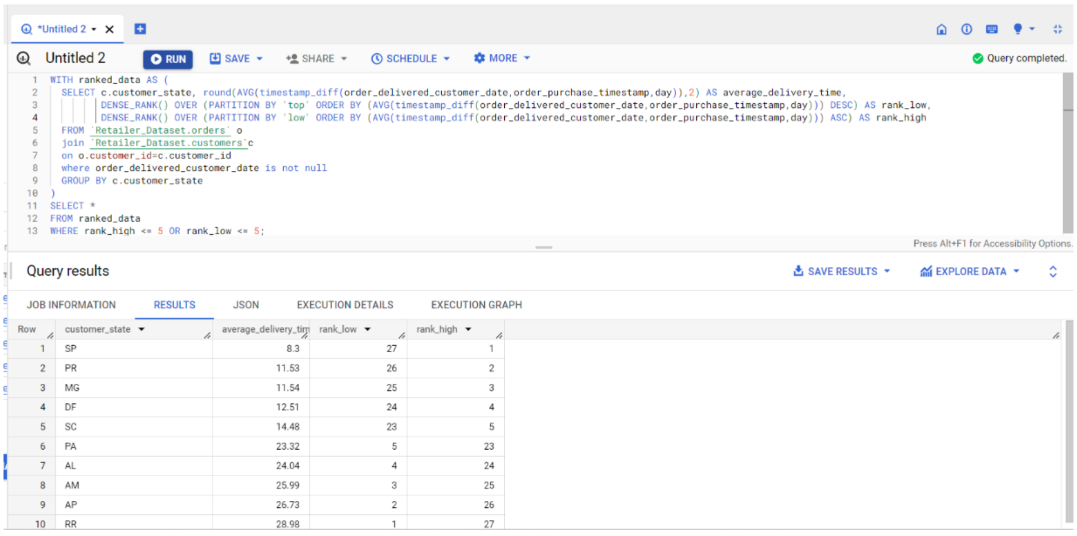
Here I tried to write in only one query we can separately find the data also.

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

Answer:

```
WITH ranked_data AS (
  SELECT c.customer_state,
round(AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day)),2) AS
average_delivery_time,
         DENSE_RANK() OVER (PARTITION BY 'top' ORDER BY
(AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day))) DESC) AS rank_low,
         DENSE_RANK() OVER (PARTITION BY 'low' ORDER BY
(AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day))) ASC) AS rank_high
  FROM `Retailer_Dataset.orders` o
  join `Retailer_Dataset.customers`c
  on o.customer_id=c.customer_id
  where order_delivered_customer_date is not null
  GROUP BY c.customer_state
)
SELECT *
FROM ranked_data
WHERE rank_high <= 5 OR rank_low <= 5;
```

Snapshot:



```
WITH ranked_data AS (
  SELECT c.customer_state, round(AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day)),2) AS average_delivery_time,
         DENSE_RANK() OVER (PARTITION BY 'top' ORDER BY (AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day))) DESC) AS rank_low,
         DENSE_RANK() OVER (PARTITION BY 'low' ORDER BY (AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day))) ASC) AS rank_high
  FROM `Retailer_Dataset.orders` o
  join `Retailer_Dataset.customers`c
  on o.customer_id=c.customer_id
  where order_delivered_customer_date is not null
  GROUP BY c.customer_state
)
SELECT *
FROM ranked_data
WHERE rank_high <= 5 OR rank_low <= 5;
```

Row	customer_state	average_delivery_tim	rank_low	rank_high
1	SP	8.3	27	1
2	PR	11.53	26	2
3	MG	11.54	25	3
4	DF	12.51	24	4
5	SC	14.48	23	5
6	PA	23.32	5	23
7	AL	24.04	4	24
8	AM	25.99	3	25
9	AP	26.73	2	26
10	RR	28.98	1	27

Insight/Recommendation:

In this data we can find which state take less delivery the order and which state take more time to deliver order and with rank we can filter them into top 5 livery time and bottom 5 with highest delivery time.

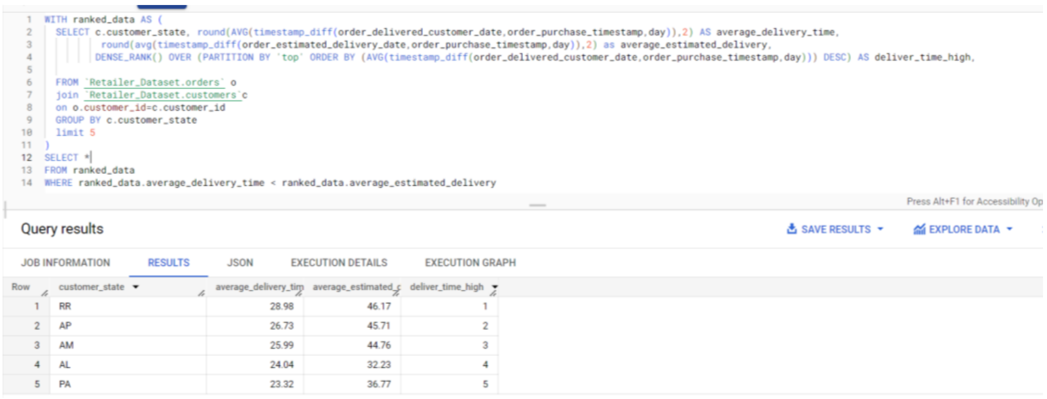
4) Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.
You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

Answer:

```
WITH ranked_data AS (
  SELECT c.customer_state,
round(AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day)),2) AS
average_delivery_time,
          round(avg(timestamp_diff(order_estimated_delivery_date,order_purchase_timestamp,day)),2) as
average_estimated_delivery,
         DENSE_RANK() OVER (PARTITION BY 'top' ORDER BY
(AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day))) DESC) AS
deliver_time_high,

  FROM `Retailer_Dataset.orders` o
  join `Retailer_Dataset.customers`c
  on o.customer_id=c.customer_id
  GROUP BY c.customer_state
  limit 5
)
SELECT *
FROM ranked_data
WHERE ranked_data.average_delivery_time < ranked_data.average_estimated_delivery
```

Snapshot:



```
WITH ranked_data AS (
  SELECT c.customer_state, round(AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day)),2) AS average_delivery_time,
        round(avg(timestamp_diff(order_estimated_delivery_date,order_purchase_timestamp,day)),2) as average_estimated_delivery,
        DENSE_RANK() OVER (PARTITION BY 'top' ORDER BY (AVG(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day))) DESC) AS deliver_time_high,

  FROM `Retailer_Dataset.orders` o
  join `Retailer_Dataset.customers`c
  on o.customer_id=c.customer_id
  GROUP BY c.customer_state
  limit 5
)
SELECT *
FROM ranked_data
WHERE ranked_data.average_delivery_time < ranked_data.average_estimated_delivery
```

Row	customer_state	average_delivery_tim	average_estimated_c	deliver_time_high
1	RR	28.98	46.17	1
2	AP	26.73	45.71	2
3	AM	25.99	44.76	3
4	AL	24.04	32.23	4
5	PA	23.32	36.77	5

Insight/Recommendation:

Here with this data we can find which state have fastest delivery service compare to estimated delivery time. Using dense_rank() function we can give them rank according their fastest delivery time get first rank.

We display only top 5 state which have fastest delivery time.

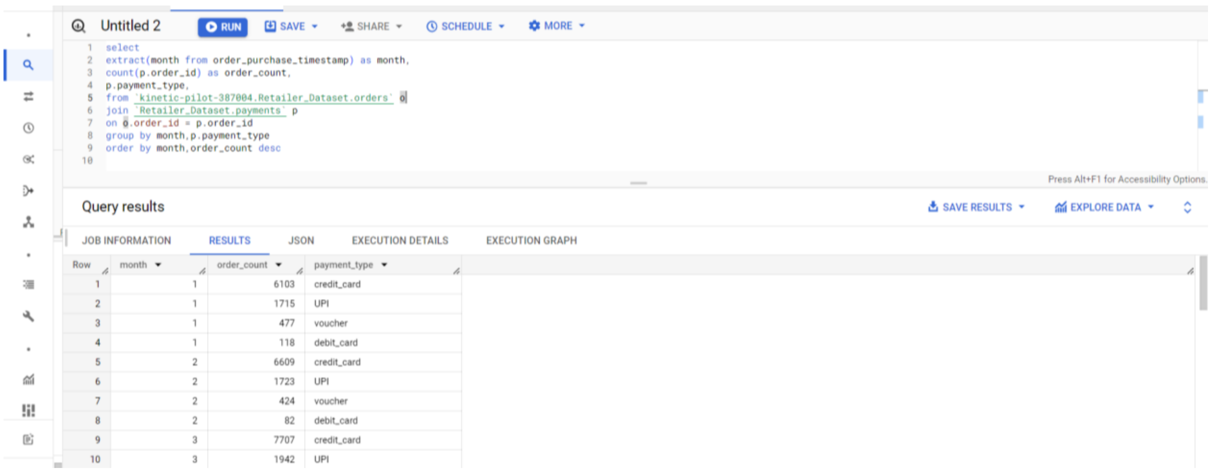
6. Analysis based on the payments:

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

Answer:

```
select
extract(month from order_purchase_timestamp) as month,
count(p.order_id) as order_count,
p.payment_type,
from `kinetic-pilot-387004.Retailer_Dataset.orders` o
join `Retailer_Dataset.payments` p
on o.order_id = p.order_id
group by month,p.payment_type
order by month,order_count desc
```

Snapshot:



```
select
extract(month from order_purchase_timestamp) as month,
count(p.order_id) as order_count,
p.payment_type,
from `kinetic-pilot-387004.Retailer_Dataset.orders` o
join `Retailer_Dataset.payments` p
on o.order_id = p.order_id
group by month,p.payment_type
order by month,order_count desc
```

Query results

Row	month	order_count	payment_type
1	1	6103	credit_card
2	1	1715	UPI
3	1	477	voucher
4	1	118	debit_card
5	2	6609	credit_card
6	2	1723	UPI
7	2	424	voucher
8	2	82	debit_card
9	3	7707	credit_card
10	3	1942	UPI

Insight/Recommendation:

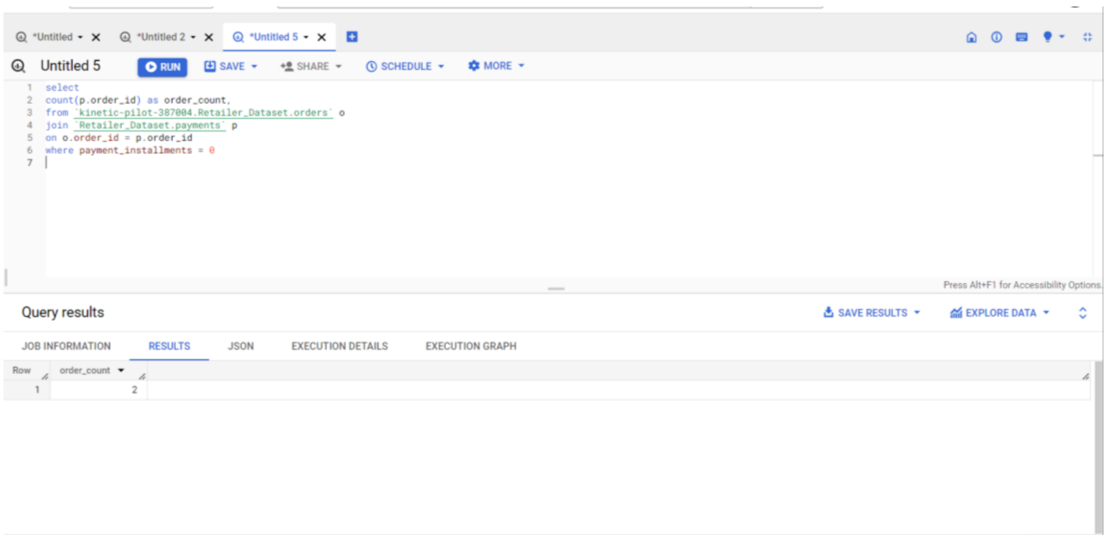
Using this query we can find in every month how many order occur using different payment of method. Here we can observe with month one and month two payment_type column that people most prefer credit_card as payment_type.
Then UPI as their payment_method.

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

Answer:

```
select
count(p.order_id) as order_count,
from `kinetic-pilot-387004.Retailer_Dataset.orders` o
join `Retailer_Dataset.payments` p
on o.order_id = p.order_id
where payment_installments = 0
```

Snapshot:



```
select
count(p.order_id) as order_count,
from `kinetic-pilot-387004.Retailer_Dataset.orders` o
join `Retailer_Dataset.payments` p
on o.order_id = p.order_id
where payment_installments = 0
```

Query results

Row	order_count
1	2

Insight/Recommendation:

With this data you can find there is only 2 orders which has fully paid other orders are on EMI options who still have remain their instalments to pay. Here I have one recommendation add one more column of remaining instalments of orders with this we can find which have less instalments remain.