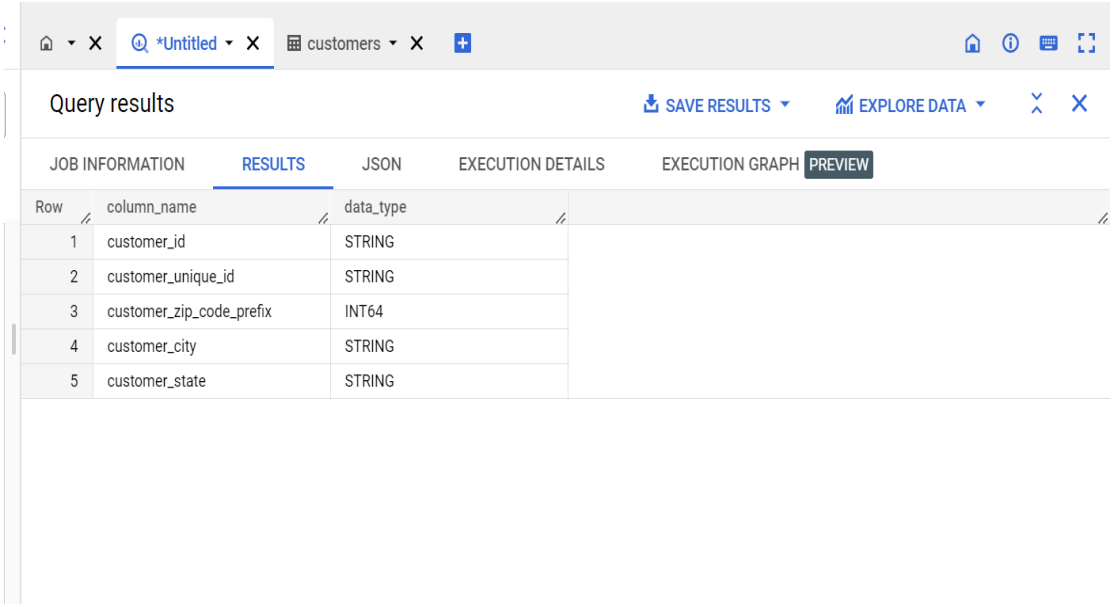
Q1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset.

Sub Q1. Data type of columns in a table.

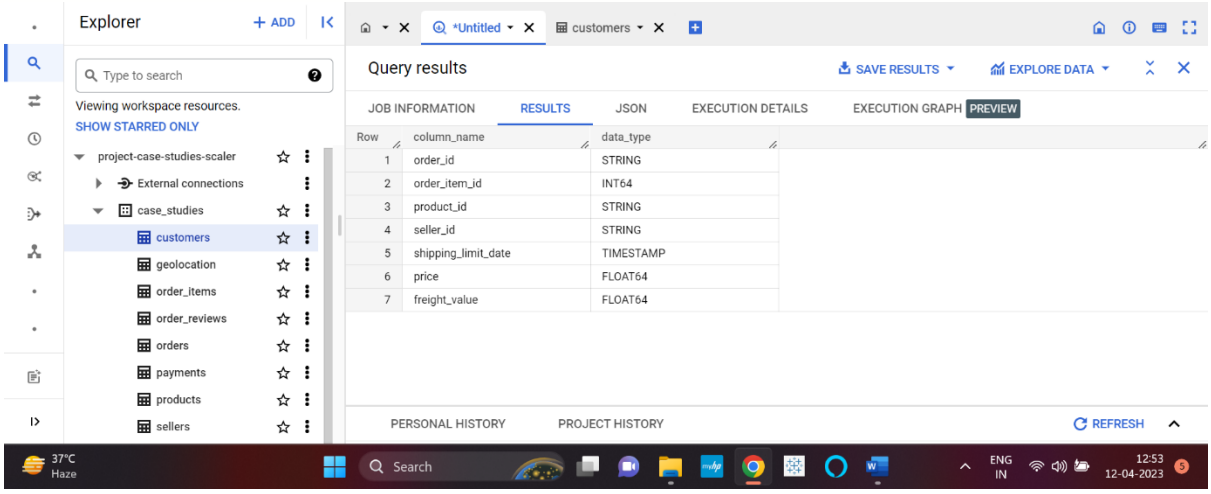
For Customer:

```
SELECT column_name, data_type
FROM `project-case-studies-scaler`.case_studies.INFORMATION_SCHEMA.COLUMNS
WHERE
table_name ='customers';
```



Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

```
SELECT column_name, data_type
FROM `project-case-studies-scaler`.case_studies.INFORMATION_SCHEMA.COLUMNS
WHERE
table_name ='order_items';
```



Row	column_name	data_type
1	order_id	STRING
2	order_item_id	INT64
3	product_id	STRING
4	seller_id	STRING
5	shipping_limit_date	TIMESTAMP
6	price	FLOAT64
7	freight_value	FLOAT64

```sql
SELECT column_name, data_type
FROM `project-case-studies-scaler`.case_studies.INFORMATION_SCHEMA.COLUMNS
WHERE
table_name ='geolocation';
```

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	column_name	data_type
1	geolocation_zip_code_prefix	INT64
2	geolocation_lat	FLOAT64
3	geolocation_lng	FLOAT64
4	geolocation_city	STRING
5	geolocation_state	STRING

PERSONAL HISTORY PROJECT HISTORY REFRESH

Search ENG IN 12:52 12-04-2023

```sql
SELECT column_name, data_type
FROM `project-case-studies-scaler`.case_studies.INFORMATION_SCHEMA.COLUMNS
WHERE
table_name ='order_reviews';
```

Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS

Row	column_name	data_type
1	review_id	STRING
2	order_id	STRING
3	review_score	INT64
4	review_comment_title	STRING
5	review_creation_date	TIMESTAMP
6	review_answer_timestamp	TIMESTAMP

```sql
SELECT column_name, data_type
FROM `project-case-studies-scaler`.case_studies.INFORMATION_SCHEMA.COLUMNS
WHERE
table_name ='orders';
```

*Untitled customers

Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS

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

```
SELECT column_name, data_type
FROM `project-case-studies-scaler`.case_studies.INFORMATION_SCHEMA.COLUMNS
WHERE
table_name ='payments';
```

Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS

Row	column_name	data_type
1	order_id	STRING
2	payment_sequential	INT64
3	payment_type	STRING
4	payment_installments	INT64
5	payment_value	FLOAT64

```
SELECT column_name, data_type
FROM `project-case-studies-scaler`.case_studies.INFORMATION_SCHEMA.COLUMNS
WHERE
table_name ='products';
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

ow	column_name	data_type
1	product_id	STRING
2	product_category	STRING
3	product_name_length	INT64
4	product_description_length	INT64
5	product_photos_qty	INT64
6	product_weight_g	INT64
7	product_length_cm	INT64
8	product_height_cm	INT64
9	product_width_cm	INT64

```
SELECT column_name, data_type
FROM `project-case-studies-scaler`.case_studies.INFORMATION_SCHEMA.COLUMNS
WHERE
table_name ='sellers';
```

*Untitled | customers

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

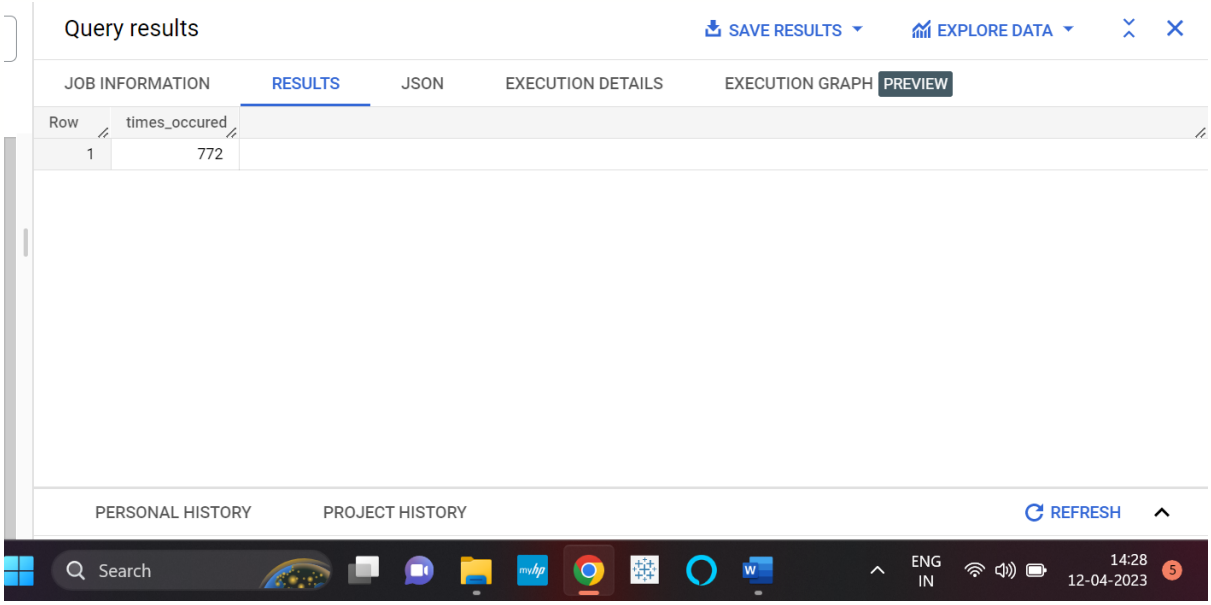
Row	column_name	data_type
1	seller_id	STRING
2	seller_zip_code_prefix	INT64
3	seller_city	STRING
4	seller_state	STRING

Sub Q2. Time period for which the data is given.

Here in this question only in order is the table where the timestamp is given followed by the order_review table also but there is no closing time in order_review as well.

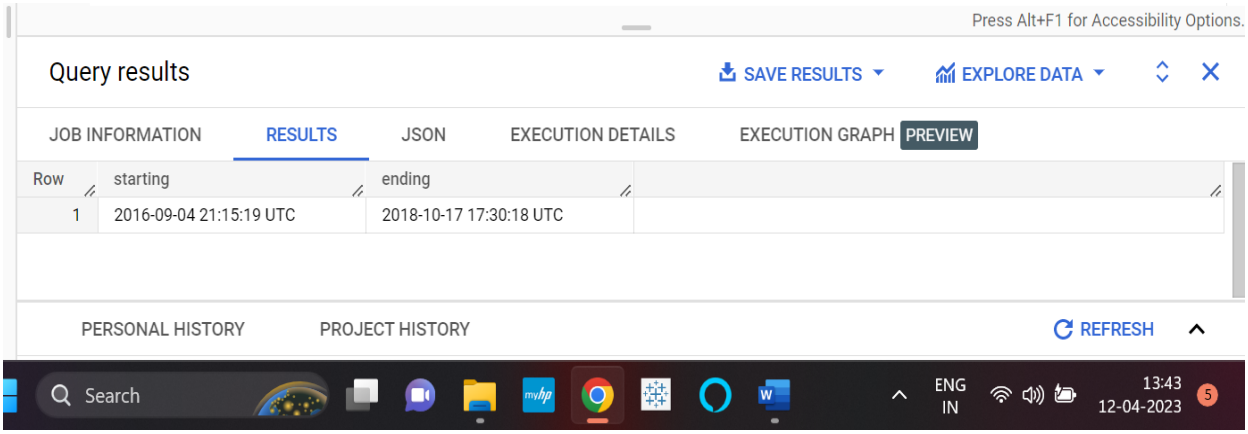
Time period can only be seen in the order table with purchase column.

```
select timestamp_diff(max(order_purchase_timestamp),min(order_purchase_timestamp),
day) as times_occured
from `project-case-studies-scaler.case_studies.orders`;
```



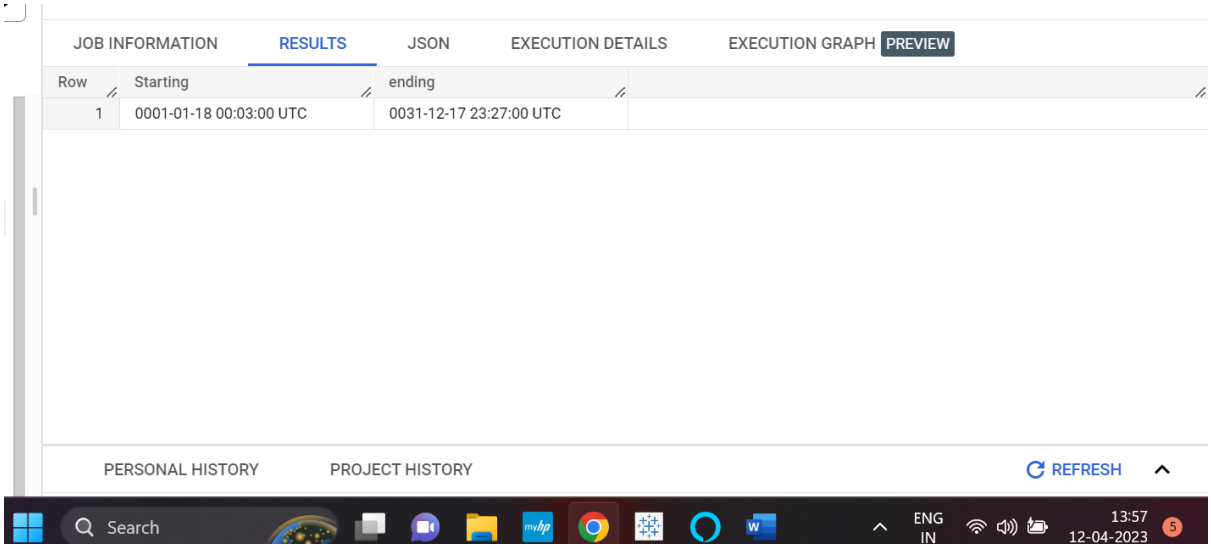
Sub Q2. The time period difference between the times of orders.

```
select min(order_purchase_timestamp) as starting, max(order_purchase_timestamp) as
ending
from `project-case-studies-scaler.case_studies.orders`;
```



Here, we have order_review table from which we fetch the time stamp date as it has not been mentioned from which table, we have to get the results.

```
select min(review_answer_timestamp) as Starting, max(review_answer_timestamp) as en
ding
from `project-case-studies-scaler.case_studies.order_reviews`;
```



Sub Q3. Cities and States of customers ordered during the given period.

```
select c.customer_id, c.customer_city, c.customer_state
from `project-case-studies-scaler.case_studies.customers` as c
join `project-case-studies-scaler.case_studies.orders` as o
on c.customer_id = o.customer_id
where
o.order_purchase_timestamp between
(
 select min(order_purchase_timestamp) from `project-case-studies-
scaler.case_studies.orders`
) and
(
  select max(order_purchase_timestamp) from `project-case-studies-
scaler.case_studies.orders`
```

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

Row	customer_id	customer_city	customer_state
31	d51d980e3ed06eec0c963264d...	itu	SP
32	ac5446c8015d744e1e3b3be6e...	itu	SP
33	73079efbc7fdf6ce0d8b8c4b1f...	itu	SP
34	481a25b5582aa352b9e8bbd2...	itu	SP
35	68ce6b05de7e3b0d5103fb995...	itu	SP
36	457aa02da26931909b20f4f92f...	itu	SP
37	32c2f812cff3dacab1dff78ccad...	itu	SP
38	a866f67b44dbcc8612120c2bc...	itu	SP
39	befb47ca7f3e74109bb67b4dd...	itu	SP
40	029c5f9144457c97ecaa33bc4...	itu	SP
41	11adf848a888e157c13ef9b59...	itu	SP

Results per page: 50 1 – 50 of 99441

Q2. In-depth Exploration:

Sub Q1. 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?

- Here we can find this complete scenario with the orders table as in orders it has been mentioned about the date with the month! While complete scenario specifies the whole date which can come with the month and year.

```
select

count(order_id) as counting_orders,
extract(year FROM order_purchase_timestamp) as year,
extract(month from order_purchase_timestamp) as month
from `project-case-studies-scaler.case_studies.orders`
group by year, month
order by year, month;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION

Row	counting_orders	year	month
1	4	2016	9
2	324	2016	10
3	1	2016	12
4	800	2017	1
5	1780	2017	2
6	2682	2017	3
7	2404	2017	4
8	3700	2017	5
9	3245	2017	6
10	4026	2017	7

We assume here the data for the 10 rows where we can find the year as 2016 and years as 2017, where the number of the orders are increasing while we have group the year and the

months wisely from September 2016 to July 2017 and the maximum number of the customer where purchased the orders as a bulk from the grouping on July 2017 as a month.

sub Q2. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Questions specifies the timing about the customer when they tend to buy as we have to mentioned over here for the specific time according to the period of the day.

Here we can find the details of the customer purchases with the timing from the table **order.**

```
SELECT COUNT(customer_id) AS customer_count,
CASE
WHEN EXTRACT(hour FROM order_purchase_timestamp) BETWEEN 4 AND 6 THEN  'Dawn'
WHEN EXTRACT(hour FROM order_purchase_timestamp) BETWEEN 7 AND 13 THEN 'Morning'
WHEN EXTRACT(hour FROM order_purchase_timestamp) BETWEEN 13 AND 19 THEN 'Afternoon'
ELSE 'Night'
END AS Purchase_time
FROM `project-case-studies-scaler.case_studies.orders`
GROUP BY Purchase_time
ORDER BY Purchase_time
```

JOB INFORMATION | RESULTS | JSON

Row	customer_count	Purchase_time
1	37599	Afternoon
2	896	Dawn
3	34251	Morning
4	26695	Night

Here from the result we can predict the customer which taken as group by as the counting with all of them comes with the major purchasing their order inn **Afternoon with 37599** followed by morning with **34251** with the night.

Q3 .Evolution of E-commerce orders in the Brazil region:

Sub Q1. Get month on month orders by states

```
SELECT *,
ROUND((z.order_id_count - (LAG(z.order_id_count,1)
OVER(PARTITION BY z.customer_state ORDER BY z.customer_state, Year, Month)))*100/LA
G(z.order_id_count,1)
OVER(PARTITION BY z.customer_state ORDER BY z.customer_state, Year, Month),2) AS
over_month
FROM
(SELECT DISTINCT c.customer_state,COUNT(o.order_id) AS order_id_count,
EXTRACT(year from order_purchase_timestamp) AS Year,
EXTRACT(month FROM order_purchase_timestamp) AS Month
FROM `project-case-studies-scaler.case_studies.orders` AS o
JOIN `project-case-studies-scaler.case_studies.customers` AS c
USING(customer_id)
GROUP BY c.customer_state, Year, Month
ORDER BY c.customer_state, Year, Month) AS z
ORDER BY z.customer_state, Year, Month
```

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

Row	customer_state	order_id_count	Year	Month	over_month
1	AC	2	2017	1	null
2	AC	3	2017	2	50.0
3	AC	2	2017	3	-33.3
4	AC	5	2017	4	150.0
5	AC	8	2017	5	60.0
6	AC	4	2017	6	-50.0
7	AC	5	2017	7	25.0
8	AC	4	2017	8	-20.0
9	AC	5	2017	9	25.0
10	AC	6	2017	10	20.0

Results per page: 10 1 – 10 of 565

PERSONAL HISTORY PROJECT HISTORY REFRESH

Search ENG IN 11:56 13-04-2023

Not exactly; there is a rising trend in orders that was initially seen in 2016 and afterwards sharply reversed in 2018. At first, we see March of 2017. We got to witness a peak in March 2017 as 150. We can also detect peaks in 2017 November and 2018 January and March. Using the available data, we see that the March peak is repeated, pointing to seasonality in the area.

SubQ2. Distribution of customers across the states in Brazil.

```
SELECT x.customer_state,
concat(ROUND(x.c_count*100/SUM(x.c_count) OVER(),2),'%') AS percentage_distribution
_showcase
FROM
(
SELECT DISTINCT COUNT(customer_id) AS c_count, customer_state
FROM `project-case-studies-scaler.case_studies.customers`
```

```sql
GROUP BY customer_state
) AS x
ORDER BY percentage_distribution_showcase DESC
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	percentage_distribution_showcase
1	RS	5.5%
2	PR	5.07%
3	SP	41.98%
4	SC	3.66%
5	BA	3.4%
6	DF	2.15%
7	ES	2.04%
8	GO	2.03%
9	RJ	12.92%

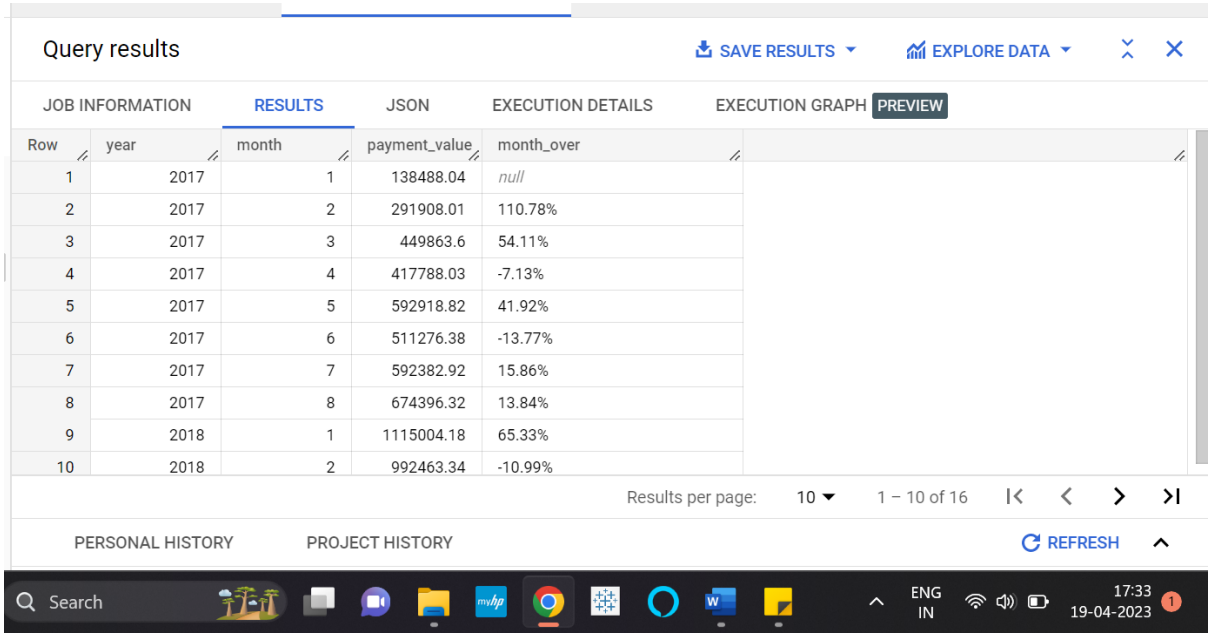
Load more

Here we see the probability as the most selling the distribution sector for the state called “SP” which covers the maximum output as the 41.98% from the entire path value as compared to all the states customers.

Q4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

> **Sub Q1.** 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

```
SELECT *,
concat(ROUND((te_mp.payment_value - (LAG(te_mp.payment_value)
OVER(ORDER BY year, month)))*100/LAG(te_mp.payment_value)
OVER(ORDER BY year, month),2),"%") AS month_over
FROM
(
  SELECT
EXTRACT(YEAR FROM(order_purchase_timestamp)) AS year,
EXTRACT(MONTH FROM(order_purchase_timestamp)) AS month,
ROUND(SUM(p.payment_value),2) AS payment_value,
FROM `project-case-studies-scaler.case_studies.orders` AS o
JOIN `project-case-studies-scaler.case_studies.payments` AS p
ON o.order_id = p.order_id
GROUP BY year,month
ORDER BY year,month asc
) AS te_mp
WHERE te_mp.year BETWEEN 2017 AND 2018 AND te_mp.month BETWEEN 1 AND 8
ORDER BY year, month
```



Row	year	month	payment_value	month_over
1	2017	1	138488.04	null
2	2017	2	291908.01	110.78%
3	2017	3	449863.6	54.11%
4	2017	4	417788.03	-7.13%
5	2017	5	592918.82	41.92%
6	2017	6	511276.38	-13.77%
7	2017	7	592382.92	15.86%
8	2017	8	674396.32	13.84%
9	2018	1	1115004.18	65.33%
10	2018	2	992463.34	-10.99%

Sub Q2. Mean & Sum of price and freight value by customer state

```
select
cust.customer_state,
round(sum(price),2) as Sum_price,
round(sum(freight_value),2) as Sum_frieght,
round(avg(price),2) as Mean_price,
round(avg(freight_value),2) as Mean_freight
from `project-case-studies-scaler.case_studies.order_items` as or_it join
`project-case-studies-scaler.case_studies.orders` as o
on or_it.order_id = o.order_id
join `project-case-studies-scaler.case_studies.customers` as cust
on o.customer_id = cust.customer_id
group by cust.customer_state;
```

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	Sum_price	Sum_frieght	Mean_price	Mean_freight
1	SP	5202955.05	718723.07	109.65	15.15
2	RJ	1824092.67	305589.31	125.12	20.96
3	PR	683083.76	117851.68	119.0	20.53
4	SC	520553.34	89660.26	124.65	21.47
5	DF	302603.94	50625.5	125.77	21.04
6	MG	1585308.03	270853.46	120.75	20.63
7	PA	178947.81	38699.3	165.69	35.83
8	BA	511349.99	100156.68	134.6	26.36
9	GO	294591.95	53114.98	126.27	22.77
10	RS	750304.02	135522.74	120.34	21.74

Results per page: 10 1 – 10 of 27

PERSONAL HISTORY PROJECT HISTORY REFRESH

Search ENG IN 16:09 19-04-2023

Q5. Analysis on sales, freight and delivery time

Sub Q1. Calculate days between purchasing, delivering and estimated delivery.

```
SELECT order_id,customer_id,
date_diff(order_estimated_delivery_date,order_purchase_timestamp, Day) as estimated
,
date_diff(order_delivered_customer_date, order_purchase_timestamp, Day) as purchasi
ng,
date_diff(order_estimated_delivery_date, order_delivered_customer_date, Day) as del
ivery
FROM `project-case-studies-scaler.case_studies.orders`;
```

Query results

SAVE RESULTS EXPLORE DA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	order_id	customer_id	estimated	purchasing	delivery
1	f88aac7ebccb37f19725a0753...	b50a0774cd941fa6d114ea6f8...	50	null	null
2	790cd37689193dca0d00d2feb...	53e76dd2ac2339c712daa2fe7...	6	null	null
3	49db7943d60b6805c3a41f547...	9cff8d557e02418fe939f23fafe...	44	null	null
4	063b573b88fc80e516aba87df...	285195a5b585842e25bd1ef90...	54	null	null
5	a68ce1686d536ca72bd2dadc4...	d7bed5fac093a4136216072ab...	56	null	null
6	45973912e490866800c0aea8f...	912f108a7026f25f99240a5c4c...	54	null	null
7	cda873529ca7ab71f677d5ec1...	76c74aaff2f3f7355f46d9818a...	56	null	null
8	ead20687129da8f5d89d831bb...	b296edf5dacd218b6457fddcb...	41	null	null
9	6f028ccb7d612af251aa442a1f...	3a0a5fd64eaf4a5c0e6030043...	3	null	null
10	8733c8d440c173e524d2fab80...	c561230659c12a017bdb3a607...	3	null	null

Results per page: 10 1 – 10 of 99441

Sub Q2. 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

```
SELECT order_id,date_diff(order_delivered_customer_date,order_purchase_timestamp, D
ay) AS time_to_delivery,

date_diff(order_delivered_customer_date,order_estimated_delivery_date,Day) AS diff_
estimated_delivery
FROM `project-case-studies-scaler.case_studies.orders`;
```

Row	order_id	time_to_delivery	diff_estimated_d
1	1950d777989f6a877539f5379...	30	12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	-28
3	65d1e226dfaeb8cdc42f66542...	35	-16
4	635c894d068ac37e6e03dc54e...	30	-1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	-1
7	276e9ec344d3bf029ff83a161c...	43	4
8	54e1a3c2b97fb0809da548a59...	40	4
9	fd04fa4105ee8045f6a0139ca5...	37	1
10	302bb8109d097a9fc6e9cefc5...	33	5

Results per page: 10 1 – 10 of 99441

PERSONAL HISTORY PROJECT HISTORY REFRESH

Search ENG IN 10:00 21-04-2023

Sub Q3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

```
SELECT DISTINCT c.customer_state,
AVG(oi.freight_value) AS AFV,
AVG(date_diff(order_delivered_customer_date,order_purchase_timestamp, Day)) AS time
_to_delivery,
AVG(date_diff(order_delivered_customer_date,order_estimated_delivery_date,Day)) AS
diff_estimated_delivery
FROM `project-case-studies-scaler.case_studies.orders` AS o
JOIN `project-case-studies-scaler.case_studies.order_items` AS oi
on o.order_id = oi.order_id
JOIN `project-case-studies-scaler.case_studies.customers` AS c
on c.customer_id = o.customer_id
GROUP BY c.customer_state;
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXE

Row	customer_state	AFV	time_to_delivery	diff_estimated_c
1	MT	28.1662843...	17.5081967...	-13.6393442...
2	MA	38.2570024...	21.2037500...	-9.10999999...
3	AL	35.8436711...	23.9929742...	-7.97658079...
4	SP	15.1472753...	8.25960855...	-10.2655943...
5	MG	20.6301668...	11.5155221...	-12.3971510...
6	PE	32.9178626...	17.7920962...	-12.5521191...
7	RJ	20.9609239...	14.6893821...	-11.1444931...
8	DF	21.0413549...	12.5014861...	-11.2747346...
9	RS	21.7358043...	14.7082993...	-13.2030001...
10	SE	36.6531688...	20.9786666...	-9.16533333...

Results per page

Sort the data to get the following:

Sub Q5. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

```
select
customer_state,
avg(ot.freight_value) as average_freight_value
from `project-case-studies-scaler.case_studies.order_items` as ot
join `project-case-studies-scaler.case_studies.orders` as o
on ot.order_id=o.order_id
join `project-case-studies-scaler.case_studies.customers` as c
on o.customer_id=c.customer_id
group by c.customer_state
order by average_freight_value desc
limit 5;
```

JOB INFORMATION | RESULTS | JSON | E

Row	customer_state	average_freight_
1	RR	42.9844230...
2	PB	42.7238039...
3	RO	41.0697122...
4	AC	40.0733695...
5	PI	39.1479704...

From the result we can see the highest freight value for the customer state RR is 42 and lowest freight value for the customer state PI is 39.

Sub Q6

Top 5 states with highest/lowest average time to delivery

```
select
customer_state,
avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,day)) as avera
ge_time_delivery
from `project-case-studies-scaler.case_studies.order_items` as ot
join `project-case-studies-scaler.case_studies.orders` as o
on ot.order_id=o.order_id
join `project-case-studies-scaler.case_studies.customers` as c
on o.customer_id=c.customer_id
group by c.customer_state
order by average_time_delivery desc
limit 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EX

Row	customer_state	average_time_de
1	RR	27.8260869...
2	AP	27.7530864...
3	AM	25.9631901...
4	AL	23.9929742...
5	PA	23.3017077...

Here is the highest time for average deliver y for the state RR is 27.82 and the lowest state for the average delivery date is the PA is 23.30

Sub Q7. Top 5 states where delivery is really fast/ not so fast compared to estimated date

```
select *,
round((estimated_delivery-delivery),3) as avg_diff
from
(
select
customer_state,
round(avg(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,day)),3)
 as delivery,
round(avg(DATE_DIFF(order_estimated_delivery_date,order_purchase_timestamp,day)),3)
 as estimated_delivery
from `project-case-studies-scaler.case_studies.order_items` as ot
join `project-case-studies-scaler.case_studies.orders` as o
on ot.order_id=o.order_id
join `project-case-studies-scaler.case_studies.customers` as c
on o.customer_id=c.customer_id
group by c.customer_state
)
order by avg_diff
```

```sql
desc limit 5;
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECU

Row	customer_state	delivery	estimated_delive	avg_diff
1	AC	20.33	40.696	20.366
2	RO	19.282	38.651	19.369
3	AM	25.963	45.206	19.243
4	RR	27.826	45.981	18.155
5	AP	27.753	45.488	17.735

Q6. Payment type analysis:

Sub Q1. Month over Month count of orders for different payment types.

```
select *,
round(((po.order_count-
lag(po.order_count) over(partition by payment_type order by Year,Month))*100)/lag(p
o.order_count) over(partition by payment_type order by Year,Month)) as Month_orders
from
(
select
count(o.order_id) as order_count,
payment_type,
extract(month from order_purchase_timestamp) as Month,
extract(year from order_purchase_timestamp) as Year
from `project-case-studies-scaler.case_studies.orders` as o
join `project-case-studies-scaler.case_studies.payments` as p
on o.order_id=p.order_id
group by payment_type,Year,Month
order by payment_type,Year,Month
) as po
```

Untitled | *2023-04-12 17:5... one

Query results

SAVE RESULTS

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

Row	order_count	payment_type	Month	Year	Month_orders
1	2	debit_card	10	2016	null
2	9	debit_card	1	2017	350.0
3	13	debit_card	2	2017	44.0
4	31	debit_card	3	2017	138.0
5	27	debit_card	4	2017	-13.0
6	30	debit_card	5	2017	11.0
7	27	debit_card	6	2017	-10.0
8	22	debit_card	7	2017	-19.0
9	34	debit_card	8	2017	55.0
10	43	debit_card	9	2017	26.0

Here we have derived the situation with the payment type by grouping and we have counted the order, as we can see the most of the order have been placed with the help of “Debit card”. As the maximum number of customers used their debit card for the order purchase.

Sub Q2.

Count of orders based on the no. of payment instalments.

```
SELECT p.payment_installments, COUNT(p.order_id) AS orders_count
FROM `project-case-studies-scaler.case_studies.payments` AS p
JOIN `project-case-studies-scaler.case_studies.orders` AS o
on p.order_id = o.order_id
GROUP BY p.payment_installments;
```

Untitled *2023-04-12 17:

Query results

JOB INFORMATION RESULTS JSON

Row	payment_installr	orders_count
1	1	52546
2	7	1626
3	10	5328
4	6	3920
5	2	12413
6	4	7098
7	3	10461
8	8	4268
9	9	644
10	5	5239