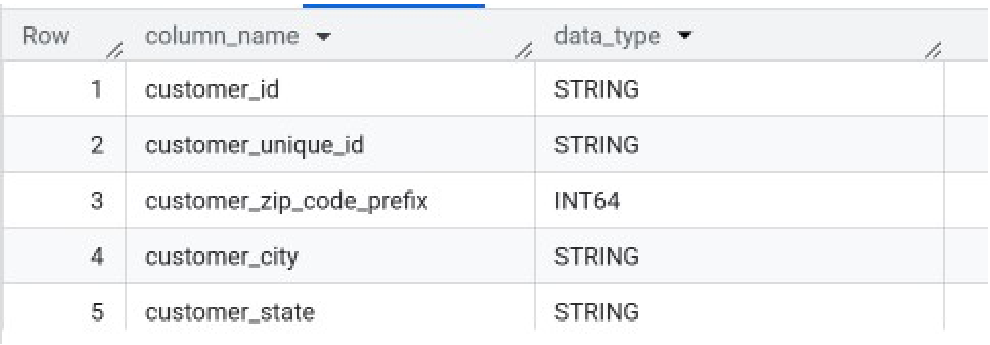
1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

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

Query

```
  SELECT
  column_name,
  data_type
FROM
  `New_project_case_study.INFORMATION_SCHEMA.COLUMNS`
WHERE
  table_name = 'customers';
```

Result



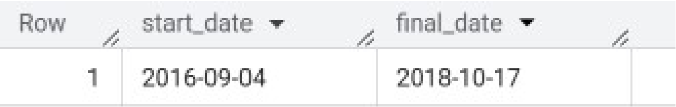
Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

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

Query

```
select min(date(order_purchase_timestamp)) as  start_date,
max(date(order_purchase_timestamp)) as final_date
from `New_project_case_study.orders`
```

Result



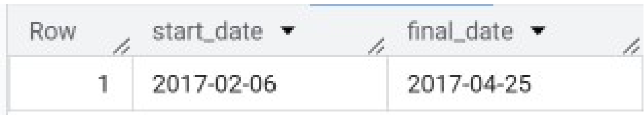
Row	start_date	final_date
1	2016-09-04	2018-10-17

The time range between the start date and final_date is approximately 2 years and 1 month.

Query

```
select min(date(order_purchase_timestamp)) as  start_date,

max(date(order_purchase_timestamp)) as final_date

from `New_project_case_study.orders`

where order_status = 'approved'
```

Result



Row	start_date	final_date
1	2017-02-06	2017-04-25

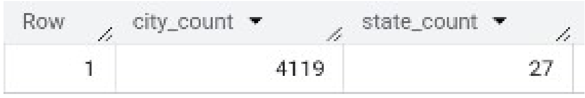
The time range between the start date and final_date when order_Status is approved is approximately 2 months.

3. Count the Cities & States of customers who ordered during the given period.

Query

```
select count(distinct(customer_city)) as city_count ,
count(distinct(customer_state)) as  city_state

from `New_project_case_study.orders` o join
`New_project_case_study.customers` c

on o.customer_id = c.customer_id

where date(order_purchase_timestamp) between

(select  min(date(order_purchase_timestamp)) from
`New_project_case_study.orders`) and

(select max(date(order_purchase_timestamp)) from
`New_project_case_study.orders`)
```

Result



Row	city_count	state_count
1	4119	27

There is 4119 unique city and 27 unique state during this order period.

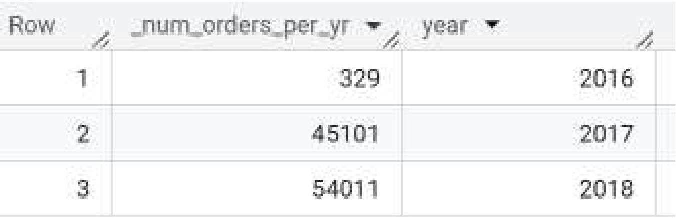
2. In-depth Exploration:

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

Query

```
select count(order_id)_num_orders_per_yr,extract(year from
order_purchase_timestamp) as year
 from `New_project_case_study.orders`
group by 2
order by 1
```

Result



Row	_num_orders_per_yr	year
1	329	2016
2	45101	2017
3	54011	2018

we can see that most of the orders placed in the year 2017 and 2018 as compared to the 2016 orders. Trend is increasing and 2018 recorded the highest number of orders

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

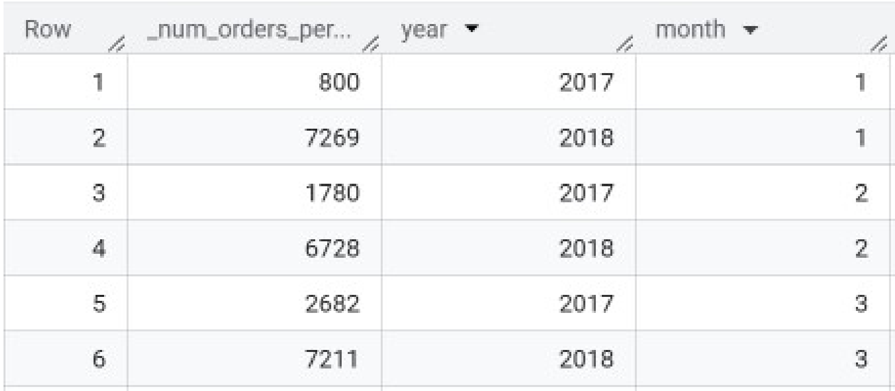
Query

```
or SELECT
   COUNT(order_id) AS num_orders_per_month,
   EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
   EXTRACT(MONTH FROM order_purchase_timestamp) AS month
```

```
FROM
    `New_project_case_study.orders`
GROUP BY
    year, month
ORDER BY
    month, year;
```

Result

to see the month on month analysis, if take instance of first month of 2017 and 2018



Row	_num_orders_per...	year	month
1	800	2017	1
2	7269	2018	1
3	1780	2017	2
4	6728	2018	2
5	2682	2017	3
6	7211	2018	3

line graph is good way to see the yearwise data, the trend shows the increasing in number of orders

3.Question -- 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

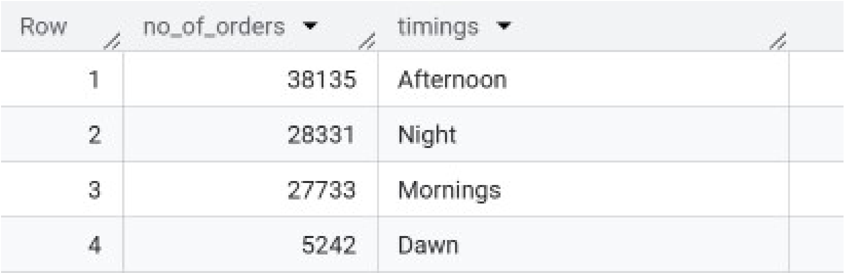
-- 13-18 hrs : Afternoon

-- 19-23 hrs : Night

Query

```
select count(order_id) as no_of_orders,
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 'Mornings'
when(extract(hour from order_purchase_timestamp) between 7 and 12)
then 'Mornings'
when(extract(hour from order_purchase_timestamp) between 13 and 18)
then 'Afternoon'
else 'Night'
end as `timings`
from `New_project_case_study.orders`
group by 2
order by 1 desc
```

Result



Row	no_of_orders	timings
1	38135	Afternoon
2	28331	Night
3	27733	Mornings
4	5242	Dawn

As we can see that afternoon has the highest number of orders with suffienctly high enough from the Night and Mornings which are almost nearby, but the dawn is the timings we can conclude for sure the no.of orders is lowest.

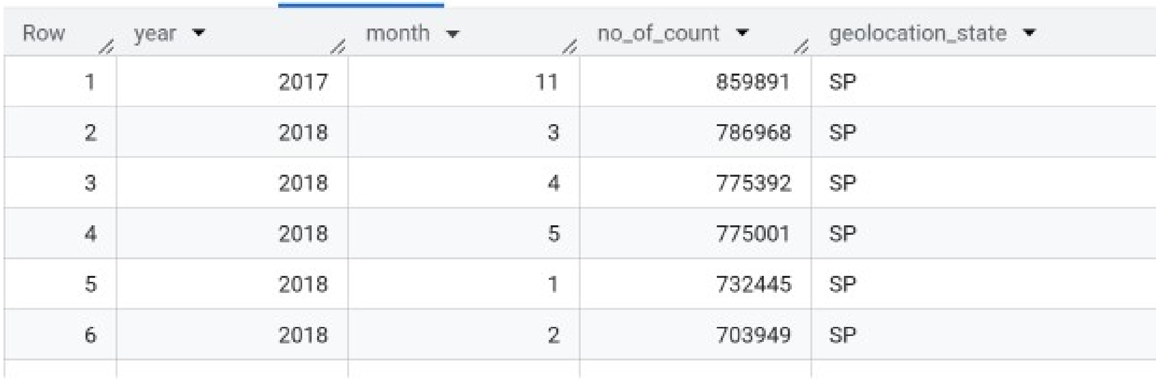
3. Evolution of E-commerce orders in the Brazil region:

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

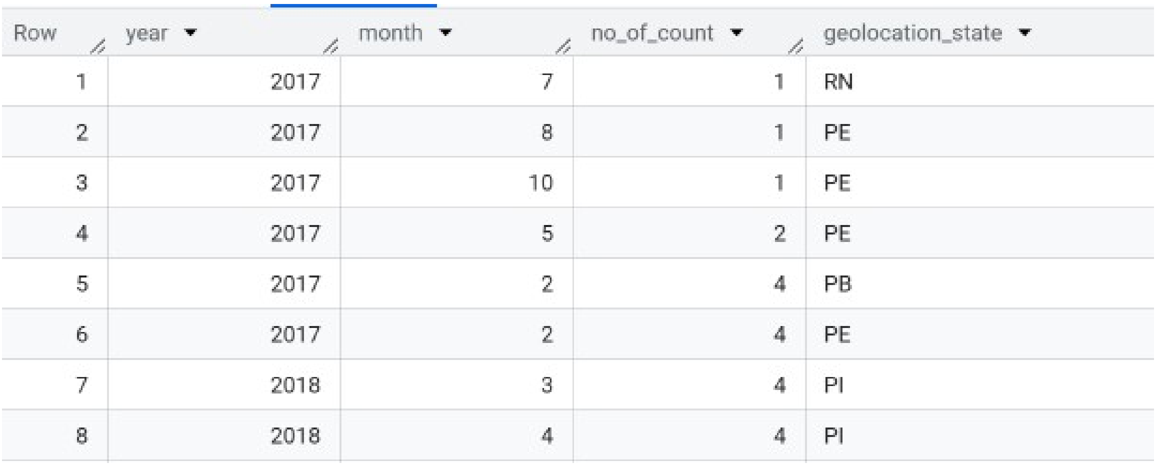
Query : can also use the customer table directly

```
select extract(year from order_purchase_timestamp) as year,
extract(month from order_purchase_timestamp) as month,
count(o.order_id) as no_of_count,geolocation_state
from `New_project_case_study.orders`
as o join `New_project_case_study.order_items` as oi
on o.order_id = oi.order_id
join `New_project_case_study.sellers` as s
on oi.seller_id = s.seller_id
join `New_project_case_study.geolocation` as g
on seller_zip_code_prefix = geolocation_zip_code_prefix
group by 1,2,4
order by 2,3 desc
```

Result



Row	year	month	no_of_count	geolocation_state
1	2017	11	859891	SP
2	2018	3	786968	SP
3	2018	4	775392	SP
4	2018	5	775001	SP
5	2018	1	732445	SP
6	2018	2	703949	SP



Row	year	month	no_of_count	geolocation_state
1	2017	7	1	RN
2	2017	8	1	PE
3	2017	10	1	PE
4	2017	5	2	PE
5	2017	2	4	PB
6	2017	2	4	PE
7	2018	3	4	PI
8	2018	4	4	PI

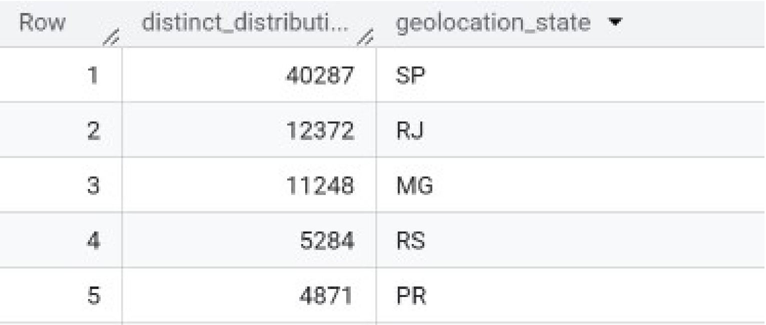
it's obvious that São Paulo has the highest number of orders being placed while doing month on month while state like Rio Grande de Norte, Pernambuco amd Paraiba has the lowest numbers of orders palced.

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

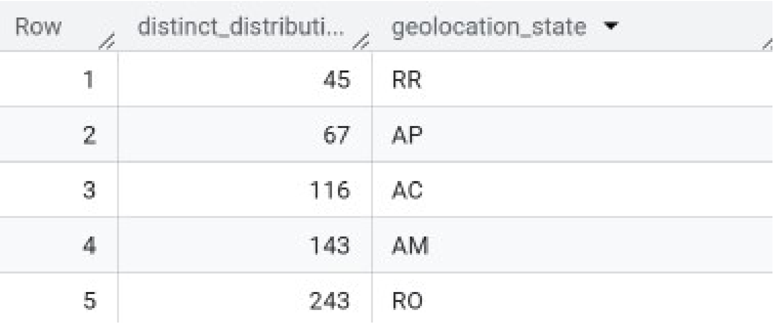
Query

```
select  count( distinct customer_unique_id) as distinct_distribution_count,
geolocation_state
from  `New_project_case_study.Customers` as c
join `New_project_case_study.geolocation`as g
on customer_zip_code_prefix = geolocation_zip_code_prefix
group by geolocation_state
order by 1 desc
```

Result



Row	distinct_distributi...	geolocation_state
1	40287	SP
2	12372	RJ
3	11248	MG
4	5284	RS
5	4871	PR



Row	distinct_distributi...	geolocation_state
1	45	RR
2	67	AP
3	116	AC
4	143	AM
5	243	RO

The Insights make it conspicious enought that Sao Paolo, Rio de Janario, Minas Gerais has highest number of customer distribution while Rio Grande de Norte, Amapa, Acre has the lowest number of customers.

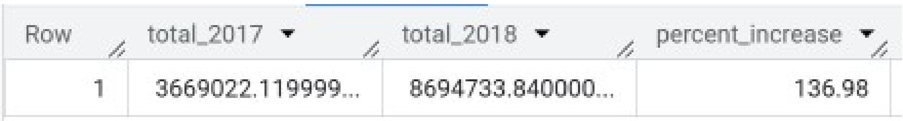
4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

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.

Query

```
with yearly_costs as (
select extract(year from order_purchase_timestamp) as year,
 sum(p.payment_value) as total_order_cost
from `New_project_case_study.orders` as o
join `New_project_case_study.payments` as p
on o.order_id = p.order_id
where extract(year from order_purchase_timestamp)  in (2017,2018) and
extract(month from order_purchase_timestamp)  between 1 and 8
group by year )

SELECT MAX(CASE WHEN year = 2017 THEN total_order_cost END)
AS total_2017,
  MAX(CASE WHEN year = 2018 THEN total_order_cost END) AS total_
2018,
Round(100.0 * (MAX(CASE WHEN year = 2018 THEN total_order_cost
END) -
                   MAX(CASE WHEN year = 2017 THEN total_order_cost
END)) /
                   MAX(CASE WHEN year = 2017 THEN total_order_cost
END), 2) AS percent_increase
FROM yearly_costs;
```

Result



Row	total_2017	total_2018	percent_increase
1	3669022.119999...	8694733.840000...	136.98

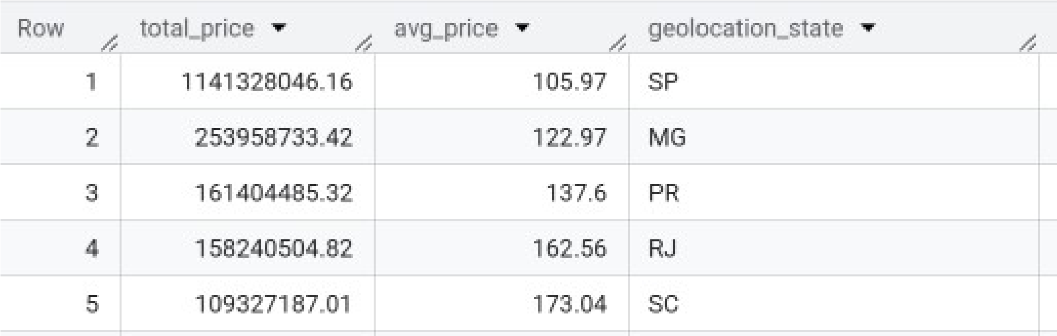
There is 136 % increase in cost of orders from the year 2017 to 2018 with months between (1 and 8). that 's a quite a growth.

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

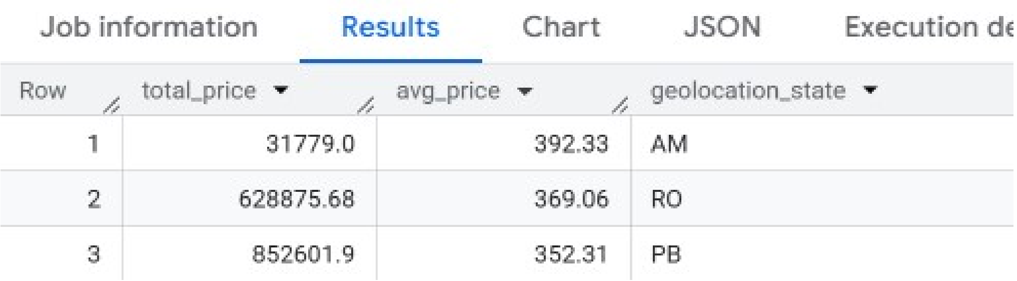
Query

```
select round(sum(price),2) as total_price, round(avg(price),2) as avg_price,
   geolocation_state
  from `New_project_case_study.order_items` as oi
  join `New_project_case_study.sellers` as s
  on oi.seller_id = s.seller_id
  join `New_project_case_study.geolocation`
  on seller_zip_code_prefix = geolocation_zip_code_prefix
  group by geolocation_state
  order by 1,2
```

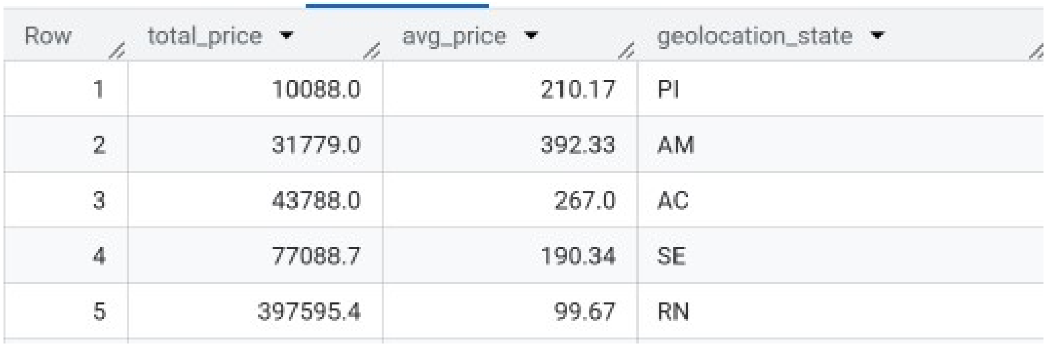
Result



Row	total_price	avg_price	geolocation_state
1	1141328046.16	105.97	SP
2	253958733.42	122.97	MG
3	161404485.32	137.6	PR
4	158240504.82	162.56	RJ
5	109327187.01	173.04	SC



Row	total_price	avg_price	geolocation_state
1	31779.0	392.33	AM
2	628875.68	369.06	RO
3	852601.9	352.31	PB



Row	total_price	avg_price	geolocation_state
1	10088.0	210.17	PI
2	31779.0	392.33	AM
3	43788.0	267.0	AC
4	77088.7	190.34	SE
5	397595.4	99.67	RN

still Sao paolo has the highest in terms of total price but avg_price i highest in AM,RO,PB.

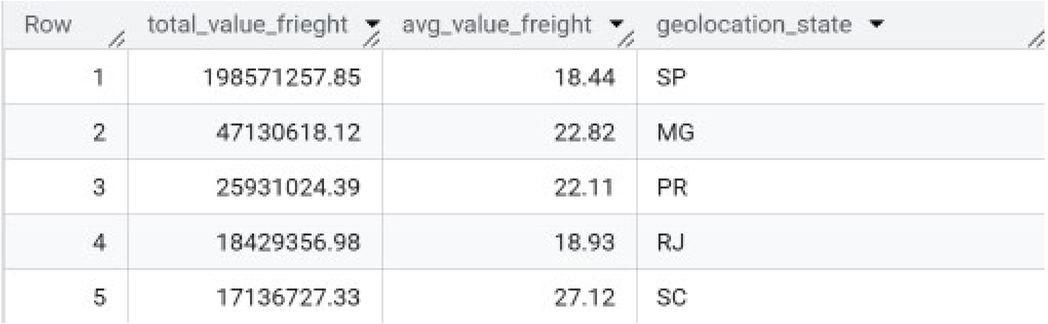
3. Calculate the Total & Average value of order freight for each state.

Query

```
select round(sum(freight_value),2) as total_value_frieght,
round(avg(freight_value),2) as avg_value_freight,
geolocation_state
from `New_project_case_study.order_items` as oi
join `New_project_case_study.sellers` as s
on oi.seller_id = s.seller_id
join `New_project_case_study.geolocation`
on seller_zip_code_prefix = geolocation_zip_code_prefix
```

```
group by geolocation_state
order by 1
```

Result



Row	total_value_frieght	avg_value_freight	geolocation_state
1	198571257.85	18.44	SP
2	47130618.12	22.82	MG
3	25931024.39	22.11	PR
4	18429356.98	18.93	RJ
5	17136727.33	27.12	SC

Ques 5. Analysis based on sales, freight and delivery time.

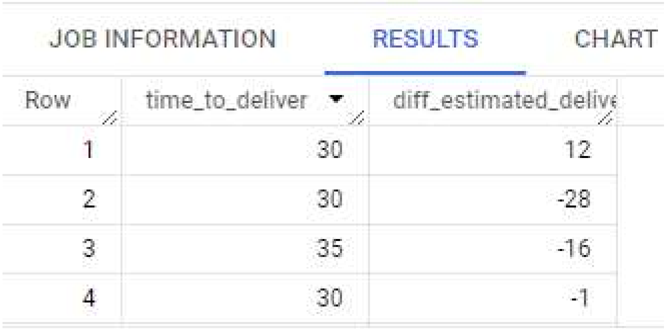
1.Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
Also, calculate the difference (in days) between the estimated & actual delivery date of an order.
Do this in a single query.

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_delivered_customer_date - order_estimated_delivery_date

Query

```
  SELECT DATE_DIFF(order_delivered_customer_date,
order_purchase_timestamp, DAY) AS
  time_to_deliver,
    date_diff(order_delivered_customer_date,order_estimated_delivery_date,
    day) as diff_estimated_delivery
    FROM `New_project_case_study.Orders`
```

Result



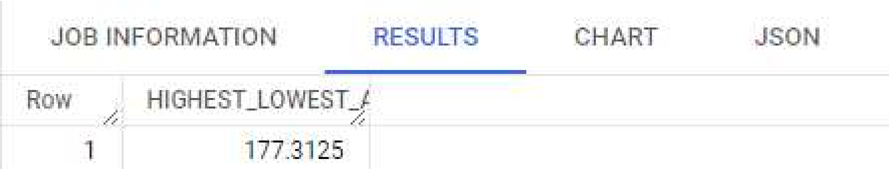
Row	time_to_deliver	diff_estimated_delive
1	30	12
2	30	-28
3	35	-16
4	30	-1

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

Query

```
 select ((top_5_avg + lowest_5_avg)/2) AS HIGHEST_LOWEST_AVG from
(
SELECT avg(H.freight_value) as top_5_avg, avg(L.freight_value) as lowest_5
_avg
from
(select * from
(select dense_rank() over(order by freight_value desc) as
rank_freight,freight_value,geolocation_state
from `New_project_case_study.Order_items` as oi
join `New_project_case_study.sellers` as s
on oi.seller_id = s.seller_id
join `New_project_case_study.geolocation`
on seller_zip_code_prefix = geolocation_zip_code_prefix
group by geolocation_state, freight_value
order by 1 ) as t
where rank_freight between 1 and 5) AS H

 JOIN

(select * from
(select dense_rank() over(order by freight_value asc) as
rank_freight,freight_value,geolocation_state
from `New_project_case_study.Order_items` as oi
join `New_project_case_study.sellers` as s
on oi.seller_id = s.seller_id
join `New_project_case_study.geolocation`
on seller_zip_code_prefix = geolocation_zip_code_prefix
group by geolocation_state, freight_value
order by 1 ) as t
where rank_freight between 1 and 5) AS L
ON H.rank_freight = L.rank_freight
) as next
```

Result



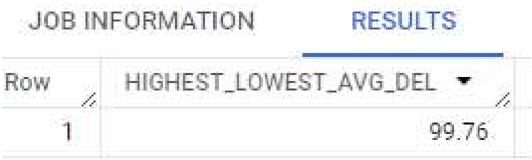
Row	HIGHEST_LOWEST_A
1	177.3125

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

Query

```
SELECT ROUND(((top_five_avg+lowest_five_avg)/2),2)as HIGHEST_LOWEST_AVG_DEL FROM
 (
select avg(table_one.time_to_deliver) as top_five_avg, avg(table_two.time_to_deliver) as
lowest_five_avg
from
(
select * from (
select *, dense_rank() over(order by main.time_to_deliver desc ) as delivery_time_rank
from
(
SELECT DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
time_to_deliver,
 geolocation_state,
FROM `New_project_case_study.Orders` as o
join `New_project_case_study.Order_items` as oi
on o.order_id = oi.order_id
join `New_project_case_study.sellers` as s
on oi.seller_id = s.seller_id
join `New_project_case_study.geolocation`
on seller_zip_code_prefix = geolocation_zip_code_prefix
) as main
group by geolocation_state,
time_to_deliver
order by delivery_time_rank
) as second_main
where delivery_time_rank between 1 and 5
) as table_one

join

(
select * from (
select *, dense_rank() over(order by main.time_to_deliver) as delivery_time_rank
from
(
SELECT DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
time_to_deliver,
 geolocation_state,
FROM `New_project_case_study.Orders` as o
join `New_project_case_study.Order_items` as oi
on o.order_id = oi.order_id
join `New_project_case_study.sellers` as s
on oi.seller_id = s.seller_id
join `New_project_case_study.geolocation`
on seller_zip_code_prefix = geolocation_zip_code_prefix
) as main
group by geolocation_state,time_to_deliver
order by delivery_time_rank desc
) as second_main
where delivery_time_rank between 1 and 5
) as table_two

on table_one.delivery_time_rank = table_two.delivery_time_rank
 ) AS FINAL
```

Result



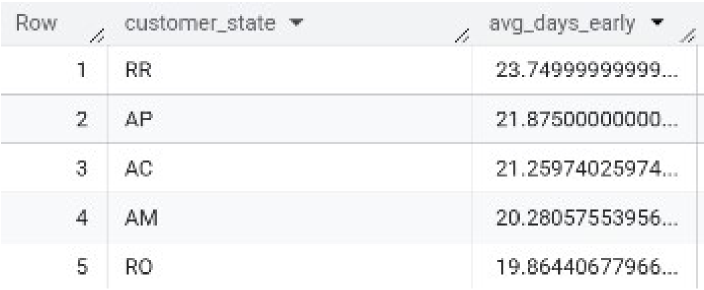
Row	HIGHEST_LOWEST_AVG_DEL
1	99.76

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.

Query

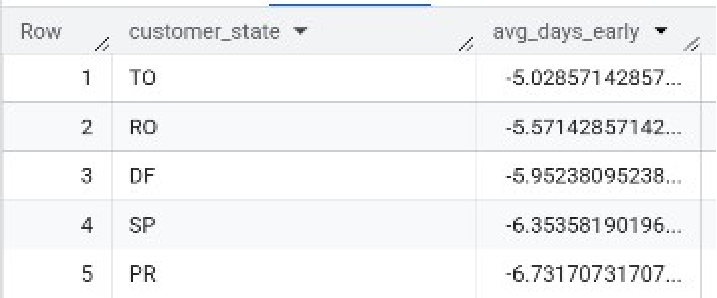
```
SELECT
  customer_state,
  AVG(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, hour)) AS
avg_days_early
FROM `New_project_case_study.orders` o
JOIN `New_project_case_study.customers` c
USING (customer_id)
WHERE order_estimated_delivery_date IS NOT NULL
  AND order_delivered_customer_date IS NOT NULL
  AND order_delivered_customer_date < order_estimated_delivery_date
GROUP BY customer_state
ORDER BY avg_days_early DESC
LIMIT 5
```

Result



Row	customer_state	avg_days_early
1	RR	23.74999999999...
2	AP	21.87500000000...
3	AC	21.25974025974...
4	AM	20.28057553956...
5	RO	19.86440677966...

delivery take longer than expected in these states

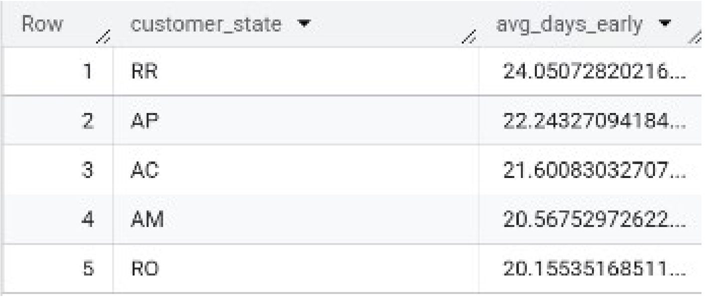


Row	customer_state	avg_days_early
1	TO	-5.02857142857...
2	RO	-5.57142857142...
3	DF	-5.95238095238...
4	SP	-6.35358190196...
5	PR	-6.73170731707...

The states where delivery is way good are the RR,AP,AC,AM,RO.
there also states where we can improve our delivery services and know what causing this delay are TO,Ro,DF,SP,PR.

or

```
WITH state_avg_delivery AS (
  SELECT
    customer_state,
    AVG(UNIX_SECONDS(order_estimated_delivery_date)) AS avg_estimated_sec,
    AVG(UNIX_SECONDS(order_delivered_customer_date)) AS avg_actual_sec
  FROM `New_project_case_study.orders` o
  JOIN `New_project_case_study.customers` c
  USING (customer_id)
  WHERE order_estimated_delivery_date IS NOT NULL
    AND order_delivered_customer_date IS NOT NULL
    AND order_delivered_customer_date < order_estimated_delivery_date
  GROUP BY customer_state
)

SELECT
  customer_state,
  (avg_estimated_sec - avg_actual_sec) / 86400 AS avg_days_early
FROM state_avg_delivery
ORDER BY avg_days_early DESC
LIMIT 5
```



Row	customer_state	avg_days_early
1	RR	24.05072820216...
2	AP	22.24327094184...
3	AC	21.60083032707...
4	AM	20.56752972622...
5	RO	20.15535168511...

this seem better to me first the average time by state then the differnece

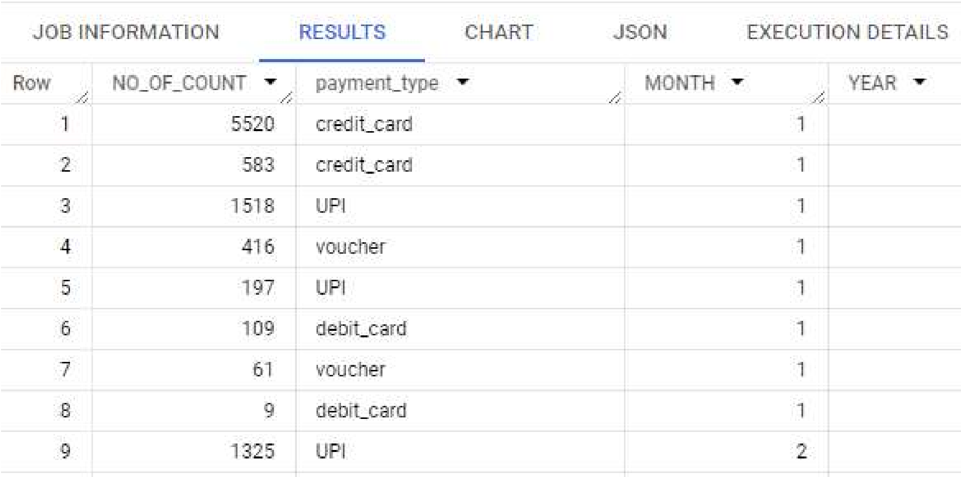
6.Analysis based on the payments:

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

Query

```
SELECT
count(p.order_id) AS NO_OF_COUNT,p.payment_type,
EXTRACT(MONTH FROM order_purchase_timestamp) AS MONTH,
EXTRACT(year FROM order_purchase_timestamp) AS YEAR
from `New_project_case_study.Orders` as o
join
`New_project_case_study.payments` as p
on o.order_id = p.order_id
group by EXTRACT(year FROM
order_purchase_timestamp),
EXTRACT(MONTH FROM
order_purchase_timestamp),
payment_type
ORDER BY MONTH
```

Result



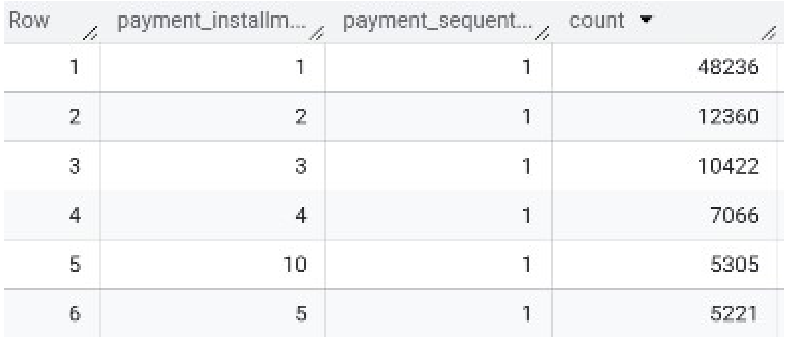
Row	NO_OF_COUNT	payment_type	MONTH	YEAR
1	5520	credit_card	1	
2	583	credit_card	1	
3	1518	UPI	1	
4	416	voucher	1	
5	197	UPI	1	
6	109	debit_card	1	
7	61	voucher	1	
8	9	debit_card	1	
9	1325	UPI	2	

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

Query

```
SELECT
  payment_installments,
  payment_sequential,
  COUNT(DISTINCT order_id) AS count
FROM `New_project_case_study.payments`
WHERE payment_installments != 0
GROUP BY payment_installments, payment_sequential;
```

Result



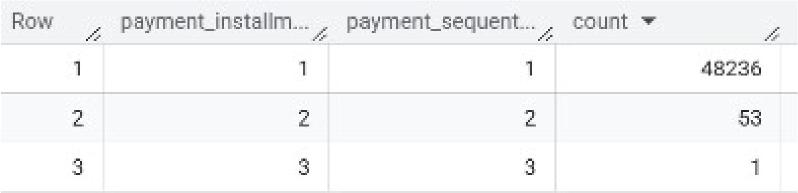
Row	payment_installm...	payment_sequent...	count
1	1	1	48236
2	2	1	12360
3	3	1	10422
4	4	1	7066
5	10	1	5305
6	5	1	5221

or

Query

```
SELECT
  payment_installments,
  payment_sequential,
  COUNT(DISTINCT order_id) AS count
FROM `New_project_case_study.payments`
WHERE payment_installments != 0  and payment_installments =payment_sequential
GROUP BY payment_installments, payment_sequential
order by 3 desc
```

Result



Row	payment_installm...	payment_sequent...	count
1	1	1	48236
2	2	2	53
3	3	3	1

Recommendations

1.Focus on Growing Markets:

- Invest in marketing and logistics infrastructure in underpenetrated states like Acre, Amapá, and Roraima.
- Potential for market expansion and new customer acquisition.

2. Boost Afternoon Promotions:

- Run special deals, ads, or flash sales during the afternoon hours to capture peak buying behavior.

3. Optimize Logistics in Delayed States:

- Conduct a root-cause analysis of delays in TO, DF, PR.
- Strengthen partnerships with local delivery agents or build warehouses nearby.

4. Capitalize on Installment Preferences:

- Promote easy financing or no-cost EMI options to push higher-ticket products.
- Highlight flexible payment options during checkout.

5. Seasonal Campaigns:

- Plan major promotional campaigns around seasonal spikes observed in monthly ordering patterns.

6. Encourage Early Deliveries:

- Celebrate fast deliveries in marketing to build customer trust.
- Use fast states (like RR, AP) as benchmarks for performance improvements elsewhere.