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

   A. Data type of columns in a table

```
select column_name, data_type from `sqlbusinesscase-
384615.SQLAssignement.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'customers'
```

Query results

SAVE RESULTS

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

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 `sqlbusinesscase-
384615.SQLAssignement.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'geolocation'
```

Query results

SAVE RESULTS

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

```
select column_name, data_type from `sqlbusinesscase-
384615.SQLAssignement.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'order_items'
```

Query results

SAVE RESULTS

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

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

```
select column_name, data_type from `sqlbusinesscase-
384615.SQLAssignement.INFORMATION_SCHEMA.COLUMNS`

where table_name = 'order_reviews'
```

Query results

SAVE RESULTS

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

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

```
select column_name, data_type from `sqlbusinesscase-
384615.SQLAssignement.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'orders'
```

Query results

SAVE RESULTS

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

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 `sqlbusinesscase-
384615.SQLAssignement.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'payments'
```

Query results

SAVE RESULTS

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

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 `sqlbusinesscase-
384615.SQLAssignement.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'products'
```

Query results

SAVE RESULTS

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

Row	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 `sqlbusinesscase-
384615.SQLAssignement.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'sellers'
```

Query results

SAVE RESULTS

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

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

B. Time period for which the data is given

```
select min(order_purchase_timestamp)as start_time, max(order_purchase_ti
mestamp) as end_time
from `sqlbusinesscase-384615.SQLAssignement.orders`
```

Query results

SAVE RESULTS

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

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

Insights :

The time period is calculated based on the data in 'Orders' table. start_time and end_time gives the total period for which the data is given.

C. Cities and States of customers ordered during the given period

```
select c1.customer_city, c1.customer_state from `sqlbusinesscase-
384615.SQLAssignement.customers` c1
join
`sqlbusinesscase-384615.SQLAssignement.orders` ord
on c1.customer_id = ord.customer_id
where ord.order_purchase_timestamp  between
(select min(order_purchase_timestamp) from `sqlbusinesscase-
384615.SQLAssignement.orders`) and
(select max(order_purchase_timestamp) from `sqlbusinesscase-
384615.SQLAssignement.orders`)
```

Query results

SAVE RESULTS

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

Row	customer_city	customer_state
1	acu	RN
2	acu	RN
3	acu	RN
4	ico	CE
5	ico	CE
6	ico	CE
7	ico	CE
8	ico	CE
9	ico	CE
10	ico	CE

---

2. In-depth Exploration:

   A. 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?

```
select  temp.year, temp.month,temp.order_count,rank() over(partition by yea
r order by order_count desc) as order_quantity_rank  from (
select count(order_id) order_count , extract(year from order_purchase_times
tamp) as year,
extract(month from order_purchase_timestamp) as month from `sqlbusinesscase
-384615.SQLAssignement.orders` o
group by extract(month from order_purchase_timestamp),extract(year from ord
er_purchase_timestamp)) temp
order by temp.year, temp.month
```

Query results

SAVE RESULTS

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

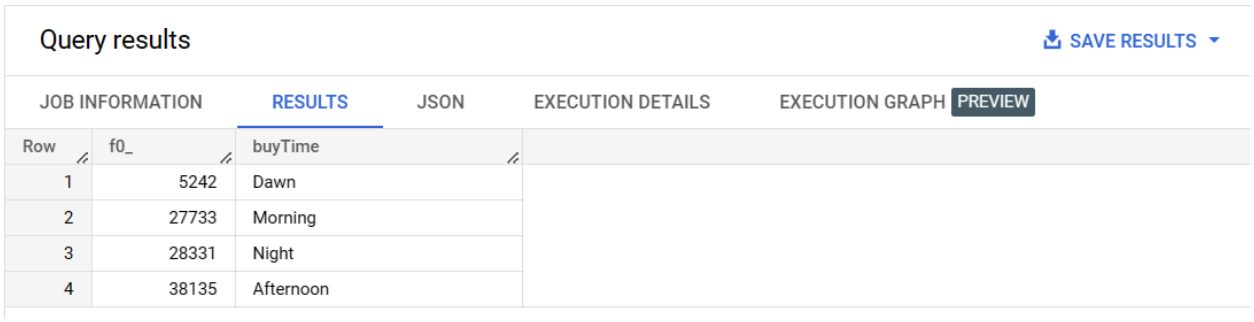
Row	year	month	order_count	order_quantity_rank
1	2016	9	4	2
2	2016	10	324	1
3	2016	12	1	3
4	2017	1	800	12
5	2017	2	1780	11
6	2017	3	2682	9
7	2017	4	2404	10
8	2017	5	3700	7
9	2017	6	3245	8
10	2017	7	4026	6
11	2017	8	4331	4
12	2017	9	4285	5
13	2017	10	4631	3

Insights:

We can see the trend of orders/purchase frequency above in both columns ‘order_count’ and ‘order_quantity_rank’. The output is based on every month and every year.

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

```
select max(t1), buyTime from
select customer_id,temp.buyTime, count(temp.buyTime) over(partition by temp
.buyTime) t1 from
(select customer_id,case
when (extract(hour from order_purchase_timestamp)) between 0 and 6 then 'Da
wn'
when (extract(hour from order_purchase_timestamp)) between 7 and 12 then 'M
orning'
when (extract(hour from order_purchase_timestamp)) between 13 and 18 then '
Afternoon'
else 'Night'
end  as buyTime
from `sqlbusinesscase-384615.SQLAssignement.orders`) temp) temp1
group by temp1.buyTime;
```



Insights : Brazilian customers tend to buy more during ‘Afternoon’

---

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

A. Get month on month orders by states

```
select * from
(select extract(month from o.order_purchase_timestamp) as month, c.customer
_state, count(o.order_id) order_count from `sqlbusinesscase-
384615.SQLAssignement.orders` o
join
`sqlbusinesscase-384615.SQLAssignement.customers` c
on o.customer_id = c.customer_id
group by c.customer_state,month) temp
order by temp.month, temp.customer_state
```

Query results

SAVE RESULTS

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

Row	month	customer_state	order_count
1	1	AC	8
2	1	AL	39
3	1	AM	12
4	1	AP	11
5	1	BA	264
6	1	CE	99
7	1	DF	151
8	1	ES	159
9	1	GO	164
10	1	MA	66
11	1	MG	971
12	1	MS	71

Insights : Month on month orders by states is sorted for month and state in ascending order.

B. Distribution of customers across the states in Brazil

```
select c.customer_state, count(c.customer_id) as customer from `sqlbusiness
case-384615.SQLAssignement.customers` c
group by c.customer_state
order by c.customer_state
```

Query results

SAVE RESULTS

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

Row	customer_state	customer
1	AC	81
2	AL	413
3	AM	148
4	AP	68
5	BA	3380
6	CE	1336
7	DF	2140
8	ES	2033
9	GO	2020
10	MA	747
11	MG	11635
12	MS	715

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

   A. 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 temp1.*,LAG(order_sum) over(order by temp1.year,temp1.month)as prev_
order_sum,
round(((order_sum-
LAG(order_sum) over(order by temp1.year,temp1.month))/LAG(order_sum) over(o
rder by temp1.year,temp1.month))*100,2) as percentage_increase from
(select temp.month,temp.year,round(sum(temp.payment_value),0) order_sum fro
m
(select *, extract(month from o.order_purchase_timestamp) as month, extract
(year from o.order_purchase_timestamp) as year  from
 `sqlbusinesscase-384615.SQLAssignement.orders` o
join
`sqlbusinesscase-384615.SQLAssignement.payments` p
on p.order_id = o.order_id
where extract(month from o.order_purchase_timestamp) between 1 and 8
order by o.order_purchase_timestamp) temp
group by  temp.year,temp.month
) temp1
order by temp1.year, temp1.month
```

Query results

SAVE RESULTS

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

Row	month	year	order_sum	prev_order_sum	percentage_incr
1	1	2017	138488.0	null	null
2	2	2017	291908.0	138488.0	110.78
3	3	2017	449864.0	291908.0	54.11
4	4	2017	417788.0	449864.0	-7.13
5	5	2017	592919.0	417788.0	41.92
6	6	2017	511276.0	592919.0	-13.77
7	7	2017	592383.0	511276.0	15.86
8	8	2017	674396.0	592383.0	13.84
9	1	2018	1115004.0	674396.0	65.33
10	2	2018	992463.0	1115004.0	-10.99

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

```
select c.customer_state,round(sum(i.price),2) price_sum, round(sum(i.freigh
t_value),2) freight_sum,
round(avg(i.price),2) price_mean,
```

```
round(avg(i.freight_value),2) freight_mean,
from `sqlbusinesscase-384615.SQLAssignement.customers` c
join
`sqlbusinesscase-384615.SQLAssignement.orders` o
on c.customer_id = o.customer_id
join
`sqlbusinesscase-384615.SQLAssignement.order_items` i
on i.order_id = o.order_id
group by c.customer_state
```

Query results

SAVE RESULTS

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

Row	customer_state	price_sum	freight_sum	price_mean	freight_mean
1	MT	156453.53	29715.43	148.3	28.17
2	MA	119648.22	31523.77	145.2	38.26
3	AL	80314.81	15914.59	180.89	35.84
4	SP	5202955.05	718723.07	109.65	15.15
5	MG	1585308.03	270853.46	120.75	20.63
6	PE	262788.03	59449.66	145.51	32.92
7	RJ	1824092.67	305589.31	125.12	20.96
8	DF	302603.94	50625.5	125.77	21.04
9	RS	750304.02	135522.74	120.34	21.74
10	SE	58920.85	14111.47	153.04	36.65
11	PR	683083.76	117851.68	119.0	20.53
12	PA	178947.81	38699.3	165.69	35.83

---

5. Analysis on sales, freight and delivery time

   A. Calculate days between purchasing, delivering and estimated delivery

```
select abs(date_diff(order_estimated_delivery_date,order_purchase_timestamp
,day)) as purchase_estimated_delivery_diff,
abs(date_diff(order_delivered_customer_date,order_purchase_timestamp,day))
as purchase_delivery_diff,
abs(date_diff(order_delivered_customer_date,order_estimated_delivery_date,d
ay)) as actual_estimated_delivery_diff,
from `sqlbusinesscase-384615.SQLAssignement.orders`
order by purchase_estimated_delivery_diff desc;
```

Query results

SAVE RESULTS

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

Row	purchase_estimated_delivery_diff	purchase_delivery_diff	actual_estimated_delivery_diff
1	155	20	134
2	149	3	146
3	146	6	139
4	144	null	null
5	144	null	null
6	142	null	null
7	140	16	123
8	116	7	108
9	109	54	55
10	106	null	null
11	101	63	37

Insights : ABS() is used to get the difference(positive) in days. ORDER BY() is included to display values other than NULL as first few rows gave NULL values for 'purchase_delivery_diff' and 'actual_estimated_delivery_diff'

B. Find time_to_delivery & diff_estimated_delivery

```
select abs(date_diff(order_purchase_timestamp, order_delivered_customer_dat
e, day)) as time_to_delivery,
abs(date_diff(order_estimated_delivery_date, order_delivered_customer_date,
 day)) as diff_estimated_delivery,from `sqlbusinesscase-
384615.SQLAssignement.orders`
```

Query results

SAVE RESULTS

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

Row	time_to_delivery	diff_estimated_delivery
1	7	45
2	7	44
3	10	41
4	6	29
5	20	40
6	10	48
7	28	29
8	9	35
9	10	41
10	6	41

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

```
(select round(avg(t.freight_value),2) as freight_avg,
round(avg(t.time_to_delivery),2) as time_to_delivery_avg,
round(avg(t.diff_estimated_delivery),2) as diff_estimated_delivery_avg,
t.customer_state from
(select abs(date_diff(order_purchase_timestamp, order_delivered_customer_da
te, day)) as time_to_delivery,
abs(date_diff(order_estimated_delivery_date, order_delivered_customer_date,
 day)) as diff_estimated_delivery,
o2.freight_value, c1.customer_state
from `sqlbusinesscase-384615.SQLAssignement.orders` o1 join
`sqlbusinesscase-384615.SQLAssignement.order_items` o2
on o1.order_id = o2.order_id
join
`sqlbusinesscase-384615.SQLAssignement.customers` c1
on o1.customer_id = c1.customer_id) t
group by t.customer_state)
```

Query results

SAVE RESULTS

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

Row	freight_avg	time_to_delivery	diff_estimated_c	customer_state
1	28.17	17.51	14.89	MT
2	38.26	21.2	12.66	MA
3	35.84	23.99	12.06	AL
4	15.15	8.26	10.99	SP
5	20.63	11.52	13.13	MG
6	32.92	17.79	14.69	PE
7	20.96	14.69	14.23	RJ
8	21.04	12.5	12.22	DF
9	21.74	14.71	14.46	RS
10	36.65	20.98	13.91	SE
11	20.53	11.48	13.16	PR

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

==> **Highest 5 average freight value**

```
select round(avg(t.freight_value),2) as freight_avg,
 t.customer_state as state from
(select abs(date_diff(order_purchase_timestamp, order_delivered_customer_date,
 day)) as time_to_delivery,
```

```
abs(date_diff(order_estimated_delivery_date, order_delivered_customer_date, da
y)) as diff_estimated_delivery,
o2.freight_value, c1.customer_state
from `sqlbusinesscase-384615.SQLAssignement.orders` o1 join
`sqlbusinesscase-384615.SQLAssignement.order_items` o2
on o1.order_id = o2.order_id
join
`sqlbusinesscase-384615.SQLAssignement.customers` c1
on o1.customer_id = c1.customer_id) t
group by t.customer_state
order by freight_avg desc
limit 5;
```

Query results

SAVE RESULTS

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

Row	freight_avg	state
1	42.98	RR
2	42.72	PB
3	41.07	RO
4	40.07	AC
5	39.15	PI

==> **Lowest 5 average freight value**

```
select round(avg(t.freight_value),2) as freight_avg,
t.customer_state as state from
(select abs(date_diff(order_purchase_timestamp, order_delivered_customer_date,
 day)) as time_to_delivery,
abs(date_diff(order_estimated_delivery_date, order_delivered_customer_date, da
y)) as diff_estimated_delivery,
o2.freight_value, c1.customer_state
from `sqlbusinesscase-384615.SQLAssignement.orders` o1 join
`sqlbusinesscase-384615.SQLAssignement.order_items` o2
on o1.order_id = o2.order_id
join
`sqlbusinesscase-384615.SQLAssignement.customers` c1
on o1.customer_id = c1.customer_id) t
group by t.customer_state
order by freight_avg
limit 5;
```

Query results

SAVE RESULTS

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

Row	freight_avg	state
1	15.15	SP
2	20.53	PR
3	20.63	MG
4	20.96	RJ
5	21.04	DF

E. Top 5 states with highest/lowest average time to delivery

==> **Top 5 states with highest average time to delivery**

```
select
round(avg(t.time_to_delivery),2) as time_to_delivery_avg,
t.customer_state as state from
(select abs(date_diff(order_purchase_timestamp, order_delivered_customer_date,
day)) as time_to_delivery,
abs(date_diff(order_estimated_delivery_date, order_delivered_customer_date, da
y)) as diff_estimated_delivery,
o2.freight_value, c1.customer_state
from `sqlbusinesscase-384615.SQLAssignement.orders` o1 join
`sqlbusinesscase-384615.SQLAssignement.order_items` o2
on o1.order_id = o2.order_id
join
`sqlbusinesscase-384615.SQLAssignement.customers` c1
on o1.customer_id = c1.customer_id) t
group by t.customer_state
order by time_to_delivery_avg DESC
limit 5;
```

Query results

SAVE RESULTS

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

Row	time_to_delivery_avg	state
1	27.83	RR
2	27.75	AP
3	25.96	AM
4	23.99	AL
5	23.3	PA

==> **Top 5 states with lowest average time to delivery**

```sql
select
round(avg(t.time_to_delivery),2) as time_to_delivery_avg,
t.customer_state as state from
(select abs(date_diff(order_purchase_timestamp, order_delivered_customer_date,
 day)) as time_to_delivery,
abs(date_diff(order_estimated_delivery_date, order_delivered_customer_date, da
y)) as diff_estimated_delivery,
o2.freight_value, c1.customer_state
from `sqlbusinesscase-384615.SQLAssignement.orders` o1 join
`sqlbusinesscase-384615.SQLAssignement.order_items` o2
on o1.order_id = o2.order_id
join
`sqlbusinesscase-384615.SQLAssignement.customers` c1
on o1.customer_id = c1.customer_id) t
group by t.customer_state
order by time_to_delivery_avg
limit 5;
```

Query results

SAVE RESULTS

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

Row	time_to_delivery_avg	state
1	8.26	SP
2	11.48	PR
3	11.52	MG
4	12.5	DF
5	14.52	SC

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

==> **Top 5 states where delivery is really fast compared to estimated date**

```sql
select
round(avg(t.diff_estimated_delivery),2) as diff_estimated_delivery_avg,
t.customer_state as state from
(select date_diff(order_purchase_timestamp, order_delivered_customer_date, day
) as time_to_delivery,
date_diff(order_estimated_delivery_date, order_delivered_customer_date, day) a
s diff_estimated_delivery,
o2.freight_value, c1.customer_state
from `sqlbusinesscase-384615.SQLAssignement.orders` o1 join
`sqlbusinesscase-384615.SQLAssignement.order_items` o2
on o1.order_id = o2.order_id
join
`sqlbusinesscase-384615.SQLAssignement.customers` c1
on o1.customer_id = c1.customer_id) t
group by t.customer_state
ORDER BY diff_estimated_delivery_avg
LIMIT 5;
```

Query results

SAVE RESULTS

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

Row	diff_estimated_delivery_avg	state
1	7.98	AL
2	9.11	MA
3	9.17	SE
4	9.77	ES
5	10.12	BA

==> **Top 5 states where delivery is really slow compared to estimated date**

```
select
round(avg(t.diff_estimated_delivery),2) as diff_estimated_delivery_avg,
t.customer_state as state from
(select date_diff(order_purchase_timestamp, order_delivered_customer_date, day
) as time_to_delivery,
date_diff(order_estimated_delivery_date, order_delivered_customer_date, day) a
s diff_estimated_delivery,
o2.freight_value, c1.customer_state
from `sqlbusinesscase-384615.SQLAssignement.orders` o1 join
`sqlbusinesscase-384615.SQLAssignement.order_items` o2
on o1.order_id = o2.order_id
join
`sqlbusinesscase-384615.SQLAssignement.customers` c1
on o1.customer_id = c1.customer_id) t
group by t.customer_state
ORDER BY diff_estimated_delivery_avg desc
LIMIT 5;
```

Query results

SAVE RESULTS

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

Row	diff_estimated_delivery_avg	state
1	20.01	AC
2	19.08	RO
3	18.98	AM
4	17.44	AP
5	17.43	RR

---

6. Payment type analysis:

A. Month over Month count of orders for different payment types

```
select * from (
select count(o.order_id) as order_count,(extract(month from o.order_purchas
e_timestamp)) as month, p.payment_type from `sqlbusinesscase-
384615.SQLAssignement.orders` o
join
`sqlbusinesscase-384615.SQLAssignement.payments` p
on o.order_id = p.order_id
group by extract(month from o.order_purchase_timestamp),p.payment_type) as
temp
order by temp.month, temp.payment_type
```

Query results

SAVE RESULTS

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

Row	order_count	month	payment_type
1	1715	1	UPI
2	6103	1	credit_card
3	118	1	debit_card
4	477	1	voucher
5	1723	2	UPI
6	6609	2	credit_card
7	82	2	debit_card
8	424	2	voucher
9	1942	3	UPI
10	7707	3	credit_card
11	109	3	debit_card
12	591	3	voucher

Insights : the result shows the total number of orders for every month for every payment mode. Result is sorted on both month wise and payment_type wise in ascending order

---

B. Count of orders based on the no. of payment installments

```
select count(o.order_id) as order_count, p.payment_installments from `sqlbu
sinesscase-384615.SQLAssignement.orders` o
join
`sqlbusinesscase-384615.SQLAssignement.payments` p
on o.order_id = p.order_id
group by p.payment_installments
order by p.payment_installments
```

Query results

SAVE RESULTS

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

Row	order_count	payment_installments
1	2	0
2	52546	1
3	12413	2
4	10461	3
5	7098	4
6	5239	5
7	3920	6
8	1626	7
9	4268	8
10	644	9
11	5328	10

Insights : The query result gives the order count based on no of payment installments in ascending order