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

select

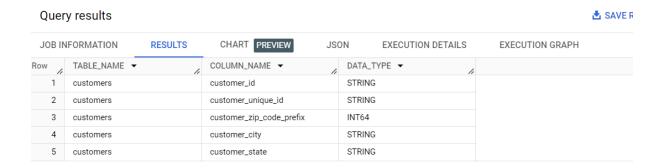
TABLE_NAME,

COLUMN_NAME,

DATA_TYPE

from `target`.INFORMATION_SCHEMA.COLUMNS

where TABLE_NAME = 'customers';



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

select

min(order_purchase_timestamp) as `start_date`,
 max(order_purchase_timestamp) as `end_date`
from `target.orders`;



Insight: Time range for order is placed between 4th September 2016 to 17th October 2018

Recommendation: N/A

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

select

count(distinct c.customer_city) as `city`,
count(distinct c.customer_state) as `state`

from `target.customers` c

inner join

`target.orders` o

on c.customer_id = o.customer_id

where o.order_purchase_timestamp between (select min(order_purchase_timestamp) from `target.orders`) and (select max(order_purchase_timestamp) from `target.orders`);

Query results

JOB IN	IFORMATION	RESULTS	CHART PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAP
Row	city ▼	state ▼	4			
1	4119	9	27			

Insight: Total 4119 cities and 27 states of customers who order the products during 4th September 2016 to 17th October 2018.

Recommendation: N/A

```
2.1. Is there a growing trend in the no. of orders placed over the past years?
select
  t.year,
  t.month,
  t.orders_count,
  sum(t.orders_count) OVER(partition by t.year) as `sum_of_orders`
from
(select
  extract(year from order_purchase_timestamp) as `year`,
  extract(month from order_purchase_timestamp) as `month`,
  count(order_id) as `orders_count`
from `target.orders`
group by year, month
order by year, month) as t
order by t.year
 Query results
```

EXECU	UTION DETAILS	JSON EXEC	RT PREVIEW	RESULTS CHA	FORMATION	JOB IN
		sum_of_orders ▼	orders_count ▼	month ▼	year ▼	Row
		329	324	10	2016	1
		329	4	9	2016	2
		329	1	12	2016	3
		45101	7544	11	2017	4
		45101	5673	12	2017	5
		45101	2404	4	2017	6
		45101	4026	7	2017	7
		45101	4631	10	2017	8
		45101	3245	6	2017	9
		45101	4285	9	2017	10



Insight: There is a growing trend over the period 2016 - 2018

Recommendation: N/A

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

```
extract(month from order_purchase_timestamp) as `month`,
  extract(year from order_purchase_timestamp) as `year`,
  count(order_id) as `total_no_of_orders`

from `target.orders`

group by year,month

order by year,month
```

Row	month ▼	year ▼	10	total_no_of_orders
9	6		2017	3245
10	7		2017	4026
11	8		2017	4331
12	9		2017	4285
13	10		2017	4631
14	11		2017	7544

Row	month ▼	year ▼	total_no_of_orders
18	3	2018	7211
19	4	2018	6939
20	5	2018	6873
21	6	2018	6167
22	7	2018	6292
23	8	2018	6512

Insight: Number of orders are having monthly seasonality during the period July to October month of 2017 and April to July month Of 2018.

Recommendation: Provide Product promotion like

discount,

combo,

advertisement,

EMI options,

Regular Inventory check of high demand products,

Avoid Delivery delays

2.3. 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

```
select
```

CASE

WHEN EXTRACT(HOUR FROM order_purchase_timestamp) > 0 and EXTRACT(HOUR FROM order_purchase_timestamp) < 6 THEN 'Dawn'

WHEN EXTRACT(HOUR FROM order_purchase_timestamp) > 7 and EXTRACT(HOUR FROM order_purchase_timestamp) < 12 THEN 'Mornings'

WHEN EXTRACT(HOUR FROM order_purchase_timestamp) > 13 and EXTRACT(HOUR FROM order_purchase_timestamp) < 18 THEN 'Afternoon'

ELSE

'NIGHT'

END as `Time_of_the_day`,
count(order_id) as `total_no_of_orders`
from target.orders
group by `Time_of_the_day`
order by `total_no_of_orders`

Query results

JOB IN	IFORMATION	RESULTS	CHART PREVIEW	JSON
Row	Time_of_the_day	▼	total_no_of_orders	
1	Dawn		2346	
2	Mornings		20507	
3	Afternoon		25848	
4	NIGHT		50740	

Insight: Most of the orders were placed during Night time and order total is "50740".

Recommendation: N/A

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

```
select
```

```
c.customer_state as `state`,
  extract(year from o.order_purchase_timestamp) as `year`,
  extract(month from o.order_purchase_timestamp) as `month`,
  count(o.order_id) as `no_of_orders`
from `target.customers` c
inner join
  `target.orders` o
  on c.customer_id = o.customer_id
  group by `state`, `year`, `month`
  order by `year`, `month`;
```

Quer	y results							≛ SAVE RESU
JOB IN	IFORMATION	RESULTS	CHART P	REVIEW	JSON	EXECUTION DETAI	LS EXECUTION GRAPH	
Row	state ▼	6	year ▼	11	month ▼	no_of_orders ▼		
1	RR		2	2016	9	1		
2	RS		2	2016	9	1		
3	SP		2	2016	9	2		
4	SP		2	2016	10	113		
5	RS		2	2016	10	24		
6	RJ		2	2016	10	56		
7	MT		2	2016	10	3		
8	GO		2	2016	10	9		
9	MG		2	2016	10	40		
10	CE		2	2016	10	8		
11	SC		2	2016	10	11		
12	AL		2	2016	10	2		

Insight: SP, MG, RJ states have consistent number of orders during January, 2017 – August, 2018.

There are no repeated order from the same customer.

Recommendation: N/A

3.2. How are the customers distributed across all the states? select

```
customer_state as `state`,
count(customer_id) as `no_of_customers`
from `target.customers`
group by `state`
order by `no_of_customers` desc
```

Query results

JOB INFORMATION **RESULTS** CHART PREVIEW **JSON EXECUTION DETAILS EXECUTION** Row / no_of_customers state ▼ 1 SP 41746 2 RJ 12852 3 MG 11635 4 RS 5466 PR 5 5045 SC 3637 7 BA 3380 DF 2140 9 ES 2033 10 GO 2020

Insight: SP, MG, RJ have population more than 10000 so order is placed high among this region.

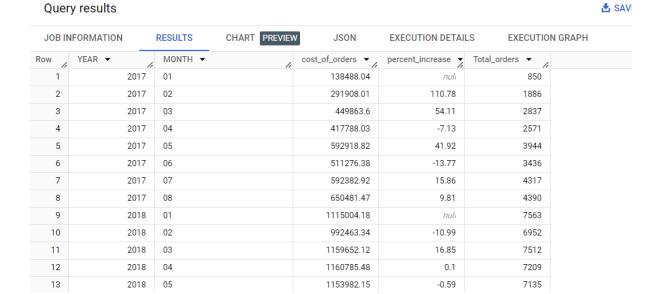
Recommendation: AC, AP, RR states have least customers. Hence perform additional promotion activities to increase sales.

4.1. 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.

```
WITH 'A' as(
select *,
 round(((t.`cost_of_orders` - lag(t.`cost_of_orders`,1) over(order by
t.\month\))/lag(t.\cost of orders\,1) over(order by t.\month\)) * 100,2) as
'percent increase'
from
(
 select
 extract(year from order purchase timestamp) as 'year',
 FORMAT DATE('%m', order purchase timestamp) as `month`,
 round(sum(p.payment_value),2) as `cost_of_orders`,
from 'target.orders' o
inner join
`target.payments` p
on o.order_id = p.order_id
where order_purchase_timestamp between '2017-01-01' and '2017-08-31'
group by `month`, `year`
) as t
order by t. `month`),
'B' as (select *,
 round(((t.`cost_of_orders` - lag(t.`cost_of_orders`,1) over(order by
t.\month\))/lag(t.\cost of orders\,1) over(order by t.\month\)) * 100,2) as
`percent_increase`
from
 select
```

```
extract(year from order purchase timestamp) as 'year',
 FORMAT_DATE('%m', order_purchase_timestamp) as `month`,
 round(sum(p.payment_value),2) as `cost_of_orders`,
from `target.orders` o
inner join
`target.payments` p
on o.order_id = p.order_id
where order_purchase_timestamp between '2018-01-01' and '2018-08-31'
group by `month`, `year`
) as t
order by t. `month`)
select
 A.year as 'YEAR',
 A.month as `MONTH`,
 A.cost_of_orders,
 A.percent_increase
from A
UNION ALL
select
 B.year,
 B.month,
 B.cost_of_orders,
 B.percent_increase
from B
order by `YEAR`, `MONTH`
```



Insight: In 2017, the % increase in the cost of order is on a growing trend. Whereas 2018 shows a declining trend because of reduction in number of orders

In 2018, 4 months (February, May, June, August) having negative percentage value and the same thing in 2017 has (April & June).

Recommendation: The reason for the reduction of the orders has to be analysed and needs improvement.

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

```
c.customer_state as `state`,
  round(sum(oi.price),2) as `Total_price`,
  round(sum(oi.price)/count(o.order_id),2) as `Average_price`
from `target.customers` c
inner join
`target.orders` o
on c.customer_id = o.customer_id
inner join `target.order_items` oi
```

on o.order_id = oi.order_id group by `state` order by `state`

Query results

JOB II	NFORMATION	RESULTS	CHART PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	state ▼	6	Total_price ▼	no_of_orders ▼	Average_price ▼	
1	PB		115268.08	602	191.48	
2	AL		80314.81	444	180.89	
3	AC		15982.95	92	173.73	
4	RO		46140.64	278	165.97	
5	PA		178947.81	1080	165.69	
6	AP		13474.3	82	164.32	
7	PI		86914.08	542	160.36	
8	TO		49621.74	315	157.53	
9	RN		83034.98	529	156.97	
10	CF		227254.71	1478	153.76	

Insight: Average price is high in top three states such as PB, AL, AC.

Recommendation: Can focus to increase in the number of orders.

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

```
c.customer_state as `state`,
    round(sum(oi.freight_value),2) as `Freight_value`,
    count(o.order_id) as `no_of_orders`,
    round(sum(oi.freight_value)/count(o.order_id),2) as
Average_freight_value`
from `target.customers` c
inner join
`target.orders` o
on c.customer_id = o.customer_id
inner join `target.order_items` oi
```

on o.order_id = oi.order_id group by `state` order by `freight_value`

Quer	y results					≛ \$
JOB IN	NFORMATION	RESULTS	CHART PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row /	state ▼	11	Freight_value ▼ //	no_of_orders ▼ //	Average_freight_value ▼ //	
1	RR		2235.19	52	42.98	
2	AP		2788.5	82	34.01	
3	AC		3686.75	92	40.07	
4	AM		5478.89	165	33.21	
5	RO		11417.38	278	41.07	
6	ТО		11732.68	315	37.25	
7	SE		14111.47	385	36.65	
8	AL		15914.59	444	35.84	
9	RN		18860.1	529	35.65	
10	MS		19144.03	819	23.37	

Insight: Less freight value in top three states such as RR, AP, AC.

Recommendation: States like SP, RJ, MG, RS has high freight value. Hence alternative freight vendors can be approached for less amount.

5.1 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

select

```
order_id,

DATE_DIFF(order_delivered_customer_date,
order_purchase_timestamp,day) as `time_to_deliver`,

DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_d
ate,day) as `diff_estimated_delivery`

from `target.orders`

where order_status = 'delivered' and
order_delivered_customer_date is not null and
order_estimated_delivery_date is not null
order by `diff_estimated_delivery`;
```

Query results

JOB IN	FORMATION	RESULTS	CHART	PREVIEW JS0	N EXECUTIO	N DETAILS	EXECUTION GRAPH
Row	order_id ▼		//	time_to_deliver ▼	diff_estimated_delive		
1	1b3190b2dfa9d7	789e1f14c05b647a	14a	208	-188		
2	ca07593549f181	16d26a572e06dc1e	ab6	209	-181		
3	47b40429ed8cc	e3aee91997922754	33f	191	-175		
4	2fe324febf907e	3ea3f2aa9650869fa	15	189	-167		
5	285ab9426d698	2034523a855f55a8	85e	194	-166		
6	440d0d17af5528	315d15a9e41abe49	359	195	-165		
7	c27815f7e3dd0b	926b58552628481	575	187	-162		
8	0f4519c5f1c541	ddec9f21b3bddd53	3a	194	-161		
9	d24e8541128ce	a179a11a65176e0a	196f	175	-161		
10	2d7561026d542	c8dbd8f0daeadf67a	a43	188	-159		

Insight: Maximum number of orders exceeded the estimated delivery date.

Recommendation: can focus to reduce the exceeded the estimated delivery date.

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

WITH 'A' as (select

```
c.customer_state as `state`,
round(AVG(oi.freight_value),2) as `average_freight_value`,
```

RANK() OVER(order by round(AVG(oi.freight_value),2)) as `lowest_value`,
RANK() OVER(order by round(AVG(oi.freight_value),2) desc) as `highest_value`
from `target.customers` c
inner join
`target.orders` o
on c.customer_id = o.customer_id
inner join `target.order_items` oi
on o.order_id = oi.order_id
group by `state`
order by `average_freight_value`)

select `A`.`state`,`A`.`average_freight_value`
from A
where `A`.`lowest_value` <= 5 or `A`.`highest_value` <= 5
order by `A`.`average_freight_value`

Query results

JOB IN	IFORMATION	RESULTS	CHART PREVIEW	JSON	EXECUTION DETAILS	
Row /	state ▼	4	average_freight_valu			
1	SP		15.15			
2	PR		20.53			
3	MG		20.63			
4	RJ		20.96			
5	DF		21.04			
6	PI		39.15			
7	AC		40.07			
8	RO		41.07			
9	PB		42.72			
10	RR		42.98			

Insight: The states of SP, PR, MG, RJ, and DF have the top 5 lowest average freight values and the states of RR, PB, RO, AC, PI have the top 5 highest average freight values.

```
Recommendation: N/A
```

```
5.3 Find out the top 5 states with the highest & lowest average delivery time.
select
t.state,
 round(t.average_delivery_time,2) as `average_delivery_time`
from
(select
  c.customer_state as `state`,
AVG(DATE DIFF(o.order delivered customer date, o.order purchase timesta
mp,DAY)) as `average_delivery_time`,
  RANK() OVER(order by
AVG(DATE DIFF(o.order delivered customer date, o.order purchase timesta
mp,DAY))) as `lowest_value`,
  RANK() OVER(order by
AVG(DATE DIFF(o.order delivered customer date,o.order purchase timesta
mp,DAY)) desc) as 'highest value'
from `target.customers` c
inner join `target.orders` o
on c.customer_id = o.customer_id
group by 'state') as t
where t.lowest value <= 5 or t.highest value <= 5
order by `average_delivery_time`
```

Query results

JOB IN	FORMATION	RESULTS	CHART P	REVIEW	JSON	EXECUTION DET
Row	state ▼	/1	average_delive	ery_tim		
1	SP			8.3		
2	PR		1	1.53		
3	MG		1	1.54		
4	DF		1:	2.51		
5	SC		1-	4.48		
6	PA		2	3.32		
7	AL		2	4.04		
8	AM		2	5.99		
9	AP		2	6.73		
10	RR		2	8.98		

Insight: The states of SP, PR, MG, DF, and SC have the top 5 lowest average freight values and the states of PA, AL, AM, AP, RR have the top 5 highest average freight values.

Recommendation: N/A

5.4 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.

select

c.customer_state as `state`,

CEIL(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_cust omer_date,day))) as `fastest_delivery`

from `target.customers` c

inner join `target.orders` o

on c.customer_id = o.customer_id

```
where o.order_status = 'delivered'
group by `state`
order by `fastest_delivery`
limit 5
```

Query results

JOB IN	IFORMATION	RESULTS	CHART PREVI	JSON	EXECUTION DETA
Row /	state ▼	le	fastest_delivery 🔻	16	
1	AL		8.0		
2	MA		9.0		
3	BA		10.0		
4	CE		10.0		
5	ES		10.0		

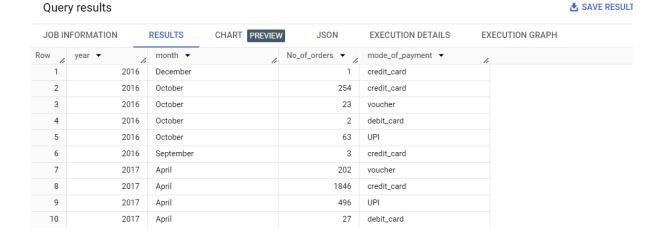
Insight: Fastest delivery state is below,

AL, MA, BA, CE, ES

Recommendation: N/A

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

```
extract(year from o.order_purchase_timestamp) as `year`,
FORMAT_TIMESTAMP('%B', o.order_purchase_timestamp) as `month`,
count( o.order_id) as `No_of_orders`,
p.payment_type as `mode_of_payment`
from `target.orders` o
inner join `target.payments` p
on o.order_id = p.order_id
group by `year`,`month`,`mode_of_payment`
order by `year`,`month`;
```



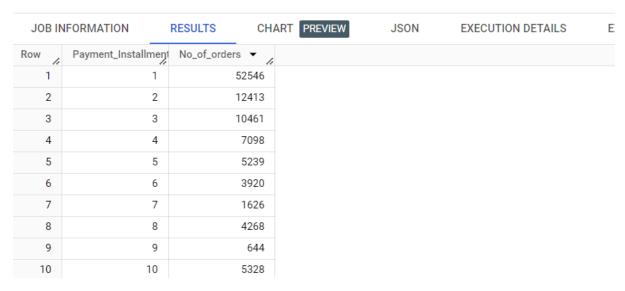
Insight: Most of the orders were placed using "credit card "mode of payment. It also shows growing trend over the months.

Recommendation: N/A

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

```
payment_installments `Payment_Installments`,
  count(o.order_id) as `No_of_orders`
from `target.orders` o
inner join `target.payments` p
on o.order_id = p.order_id
where p.payment_installments >= 1
group by payment_installments
order by `Payment_Installments`
```

Query results



Insight: Company received only one instalment for 55% of the order.

Recommendation: Collection follow up should be improved for quick customer repayment