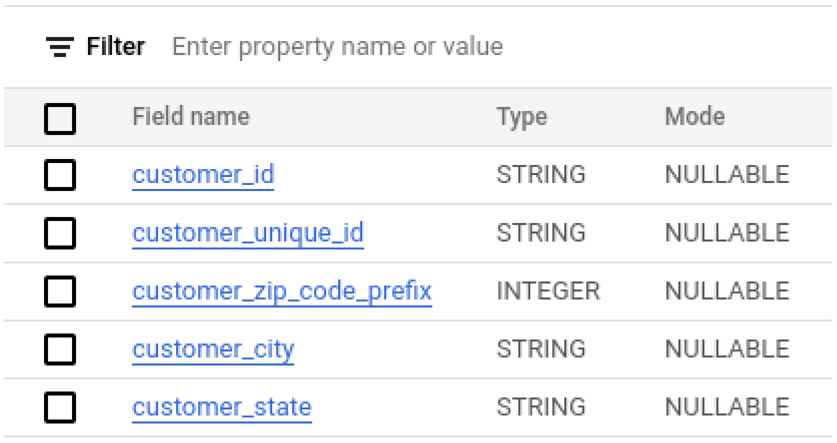
I. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset.

A. Data type of all columns in the “customers” table.



Filter Enter property name or value

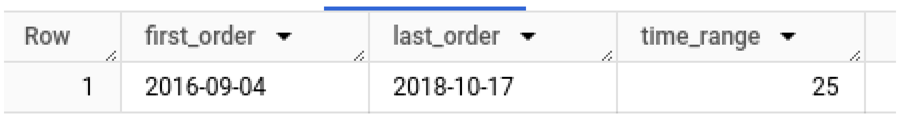
Field name	Type	Mode
customer_id	STRING	NULLABLE
customer_unique_id	STRING	NULLABLE
customer_zip_code_prefix	INTEGER	NULLABLE
customer_city	STRING	NULLABLE
customer_state	STRING	NULLABLE

Insights: customer table holds most of string data type when compared to others.

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

```
select min(date(order_purchase_timestamp)) as first_order,
max(date(order_purchase_timestamp)) as last_order,
date_diff(max(date(order_purchase_timestamp)),min(date(order_purchase_timestamp)),month) as
time_range
from `target_sql.orders`;
```

Output:



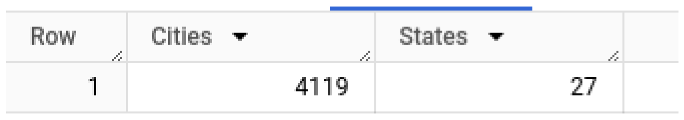
Row	first_order	last_order	time_range
1	2016-09-04	2018-10-17	25

Insights: time range is 25 months between the orders

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

```
select
count(distinct(customer_city)) as Cities,
count(distinct(customer_state)) as States
from target_sql.orders o
inner join target_sql.customers c using(customer_id);
```

Output:

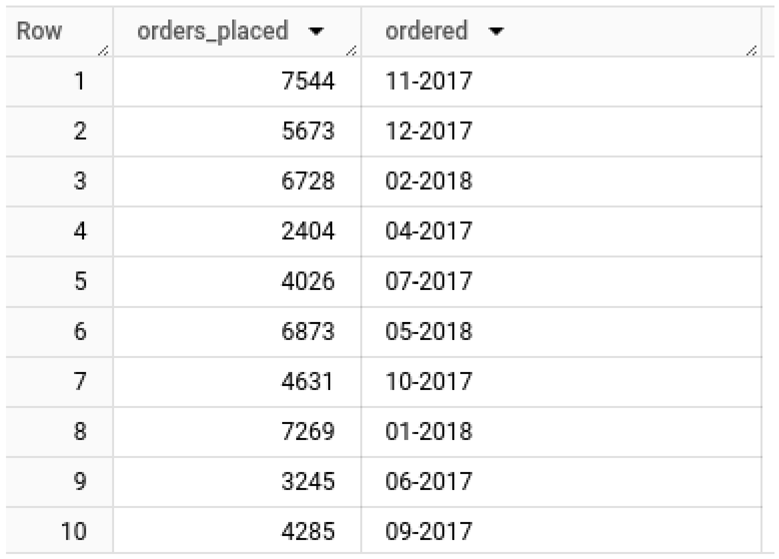


Row	Cities	States
1	4119	27

Insights: In the given period Customers from 4119 Cities which are in 27 States are ordered.

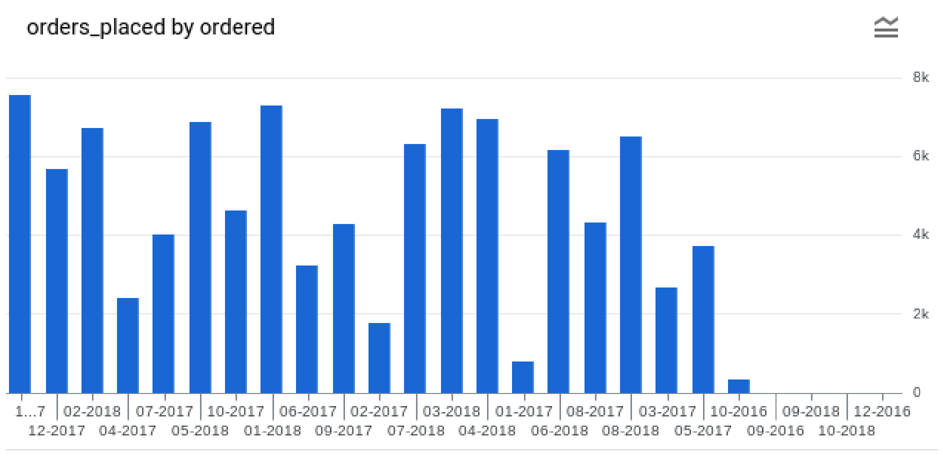
```
#II. In-depth Exploration:
#A. Is there a growing trend in the no. of orders placed over the past years?
select Count(order_status) as orders_placed,
format_datetime('%m-%Y',order_purchase_timestamp) as ordered
from `target_sql.orders`
group by format_datetime('%m-%Y',order_purchase_timestamp);
```

Output:



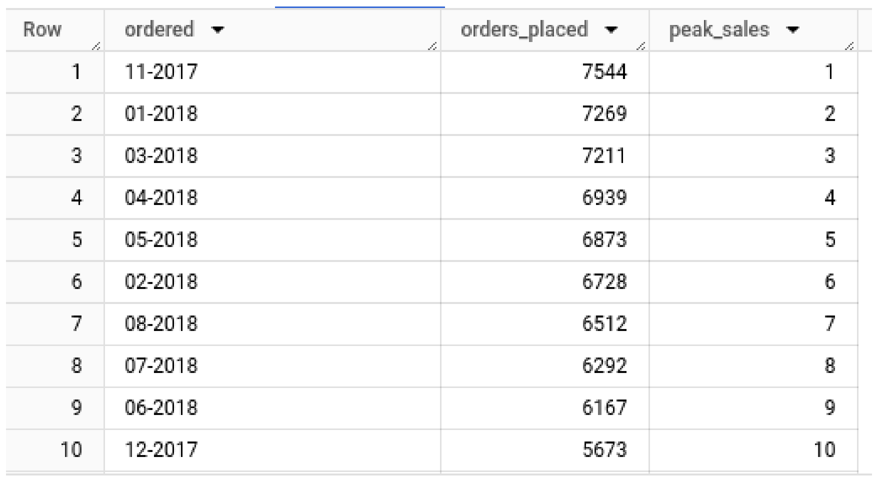
Row	orders_placed	ordered
1	7544	11-2017
2	5673	12-2017
3	6728	02-2018
4	2404	04-2017
5	4026	07-2017
6	6873	05-2018
7	4631	10-2017
8	7269	01-2018
9	3245	06-2017
10	4285	09-2017

Insights:



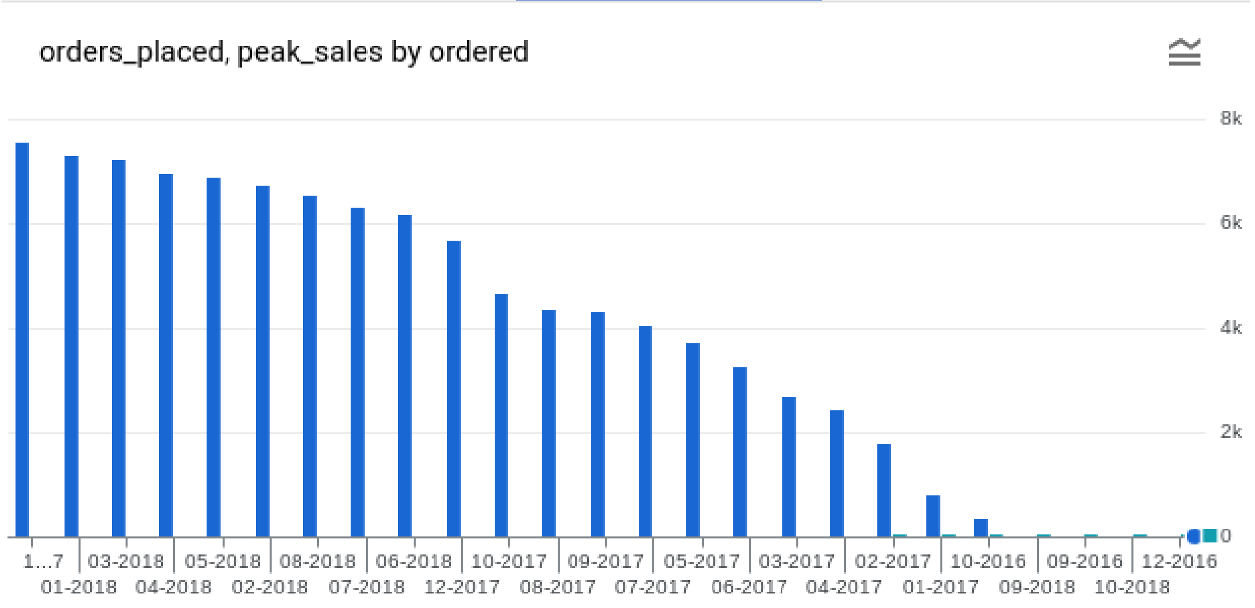
```
#B. Can we see some kind of monthly seasonality in terms of the no. of orders being
placed?
select ordered,orders_placed,
dense_rank() over(order by orders_placed desc) as peak_sales
from(
select Count(order_status) as orders_placed,
format_datetime('%m-%Y',order_purchase_timestamp) as ordered
from `target_sql.orders`
group by format_datetime('%m-%Y',order_purchase_timestamp)
)
order by peak_sales;
```

Output:



Row	ordered	orders_placed	peak_sales
1	11-2017	7544	1
2	01-2018	7269	2
3	03-2018	7211	3
4	04-2018	6939	4
5	05-2018	6873	5
6	02-2018	6728	6
7	08-2018	6512	7
8	07-2018	6292	8
9	06-2018	6167	9
10	12-2017	5673	10

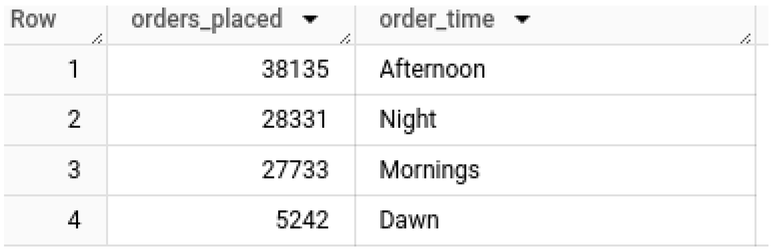
Insights:



```
#C. 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 Count(order_status) as orders_placed,
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 13 and 18 then 'Afternoon'
else 'Night'
end as order_time
from target_sql.orders
group by order_time
order by orders_placed desc;
```

Output:

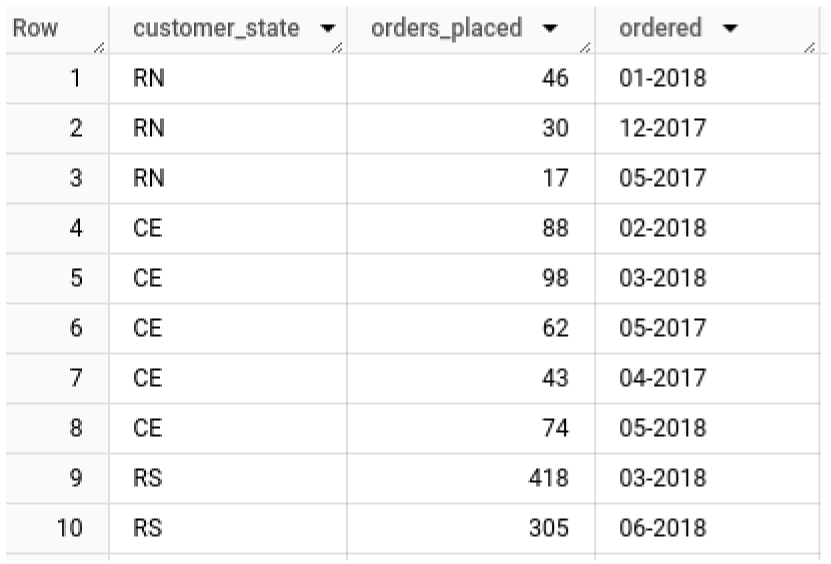


Row	orders_placed	order_time
1	38135	Afternoon
2	28331	Night
3	27733	Mornings
4	5242	Dawn

Insights: Most of the orders are placed in Afternoon time.

```
#III. Evolution of E-commerce orders in the Brazil region:
#A. Get the month on month no. of orders placed in each state.
select customer_state, Count(order_status) as orders_placed,
format_datetime('%m-%Y',order_purchase_timestamp) as ordered
from `target_sql.orders`
join `target_sql.customers` using(customer_id)
group by customer_state, format_datetime('%m-%Y',order_purchase_timestamp);
```

Output:

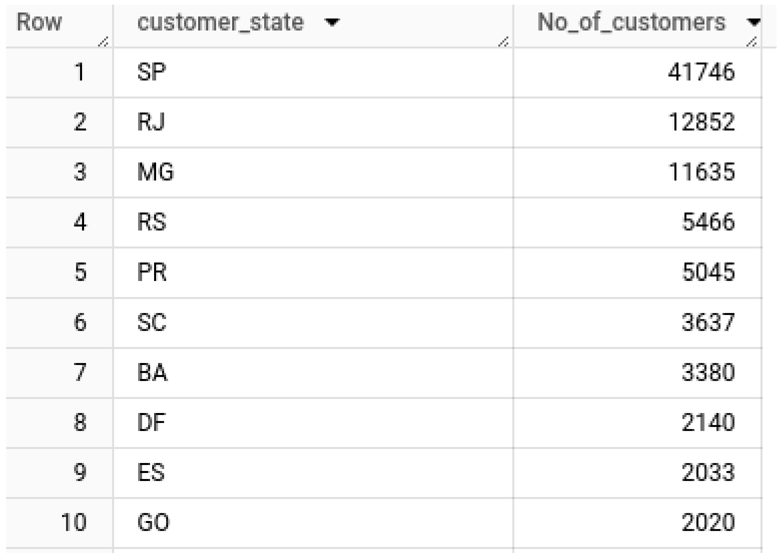


Row	customer_state	orders_placed	ordered
1	RN	46	01-2018
2	RN	30	12-2017
3	RN	17	05-2017
4	CE	88	02-2018
5	CE	98	03-2018
6	CE	62	05-2017
7	CE	43	04-2017
8	CE	74	05-2018
9	RS	418	03-2018
10	RS	305	06-2018

Insights:
State wise month on month orders in Brazil.

```
#B. How are the customers distributed across all the states?
Select customer_state,count(distinct customer_id) as No_of_customers
from `target_sql.customers`
group by customer_state
order by No_of_customers desc;
```

Output:



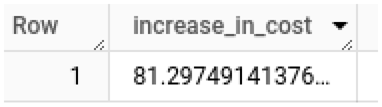
Row	customer_state	No_of_customers
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

Insights: Most of the customers are from SP,RJ and MG states.

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

```
#A. Get the % increase in the cost of orders from year 2017 to 2018 (include months
between Jan to Aug only).
select
((max(t.cost_of_orders)-min(cost_of_orders))/avg(t.cost_of_orders))*100 as
increase_in_cost
from(
select
distinct extract(year from order_purchase_timestamp) as year,
sum(payment_value) over(partition by extract(year from order_purchase_timestamp)) as
cost_of_orders,
from `target_sql.payments`
join `target_sql.orders`using(order_id)
where extract(month from order_purchase_timestamp) between 01 and 08) t;
```

Output:

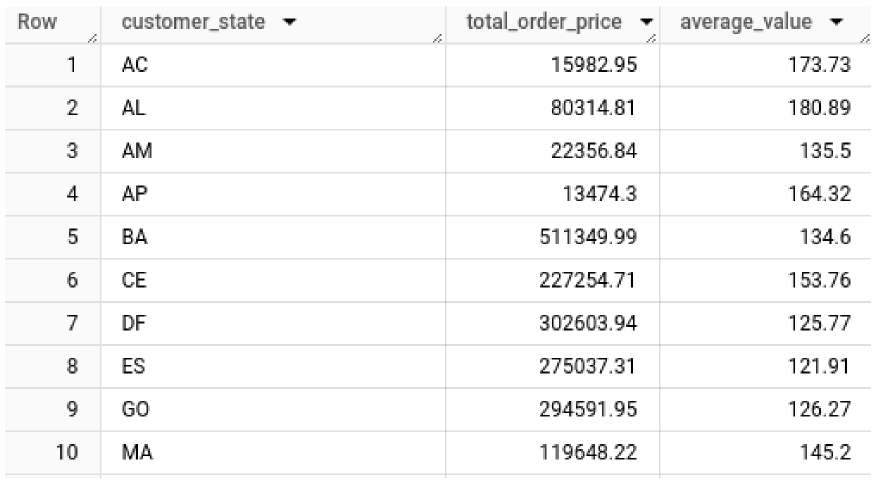


Row	increase_in_cost
1	81.29749141376...

Insights:
81.297% increase in cost of orders from year 2017 to 2018.

```
#B. Calculate the Total & Average value of order price for each state.
Select
distinct c.customer_state,
round(sum(oi.price),2) as total_order_price,
round(avg(oi.price),2) as average_value
from `target_sql.order_items` oi
join `target_sql.orders` o using(order_id)
join `target_sql.customers` c using(customer_id)
group by c.customer_state
order by c.customer_state;
```

Output:



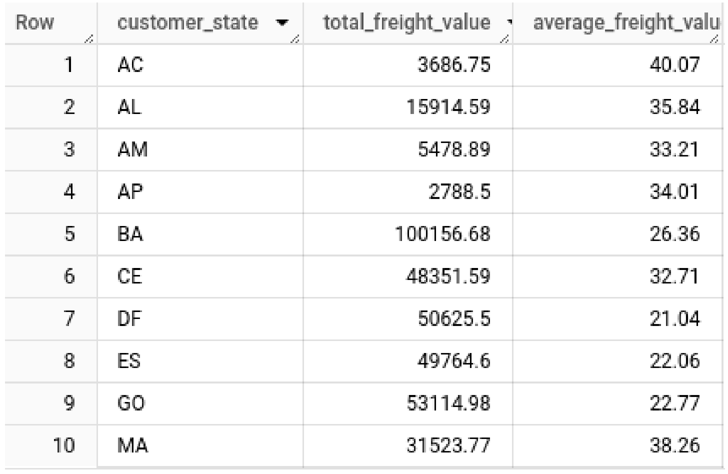
Row	customer_state	total_order_price	average_value
1	AC	15982.95	173.73
2	AL	80314.81	180.89
3	AM	22356.84	135.5
4	AP	13474.3	164.32
5	BA	511349.99	134.6
6	CE	227254.71	153.76
7	DF	302603.94	125.77
8	ES	275037.31	121.91
9	GO	294591.95	126.27
10	MA	119648.22	145.2

Insights: The total order price and average price for each state.

```
#C. Calculate the Total & Average value of order freight for each state.

Select
distinct c.customer_state,
round(sum(oi.freight_value),2) as total_freight_value,
round(avg(oi.freight_value),2) as average_freight_value
from `target_sql.order_items` oi
join `target_sql.orders` o using(order_id)
join `target_sql.customers` c using(customer_id)
group by c.customer_state
order by c.customer_state;
```

Output:



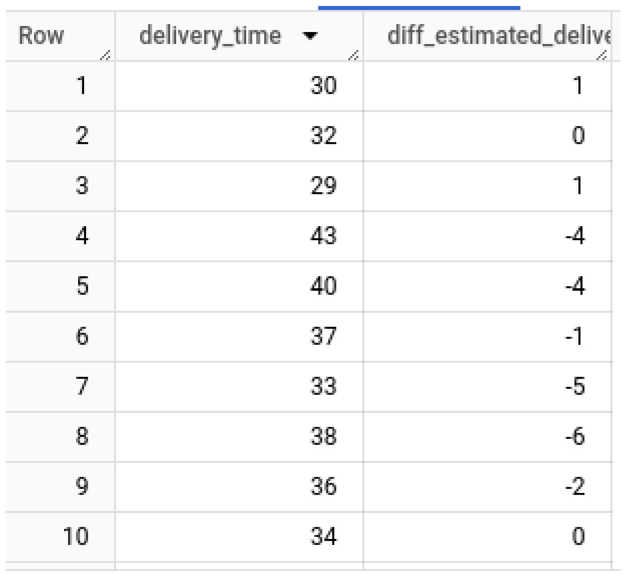
Row	customer_state	total_freight_value	average_freight_valu
1	AC	3686.75	40.07
2	AL	15914.59	35.84
3	AM	5478.89	33.21
4	AP	2788.5	34.01
5	BA	100156.68	26.36
6	CE	48351.59	32.71
7	DF	50625.5	21.04
8	ES	49764.6	22.06
9	GO	53114.98	22.77
10	MA	31523.77	38.26

Insights: The total and average freight value for each state

```
#V. Analysis based on sales, freight and delivery time.
#A. 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.


select
date_diff(order_delivered_customer_date,order_purchase_timestamp,day) as
delivery_time,
date_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as
diff_estimated_delivery
from `target_sql.orders`
where order_status = 'delivered';
```

Output:

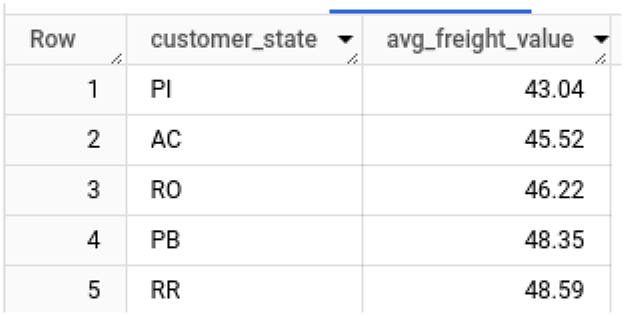


Row	delivery_time	diff_estimated_delive
1	30	1
2	32	0
3	29	1
4	43	-4
5	40	-4
6	37	-1
7	33	-5
8	38	-6
9	36	-2
10	34	0

Insights: most of the orders are delivered before estimated time.

```
#B. Find out the top 5 states with the highest & lowest average freight value.
select
t.customer_state,t.avg_freight_value
from(
select
c.customer_state,
round(sum(freight_value)/count(distinct order_id),2) as avg_freight_value
from `target_sql.order_items` oi
join `target_sql.orders` o using(order_id)
join `target_sql.customers` c using(customer_id)
group by c.customer_state
order by avg_freight_value desc
limit 5) t
order by t.avg_freight_value;
```

Output:

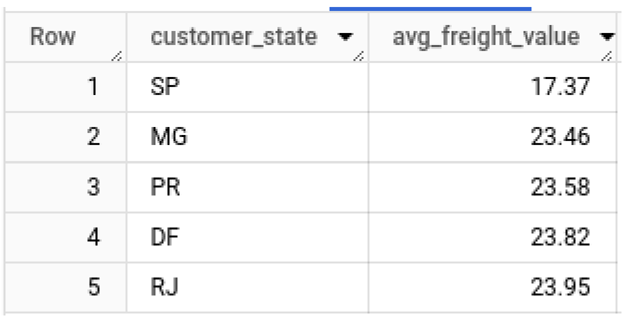


Row	customer_state	avg_freight_value
1	PI	43.04
2	AC	45.52
3	RO	46.22
4	PB	48.35
5	RR	48.59

Insights: the top five states with highest average freight value in increasing order.

```
select
c.customer_state,
round(sum(freight_value)/count(distinct order_id),2) as avg_freight_value
from `target_sql.order_items` oi
join `target_sql.orders` o using(order_id)
join `target_sql.customers` c using(customer_id)
group by c.customer_state
order by avg_freight_value
limit 5;
```

Output:

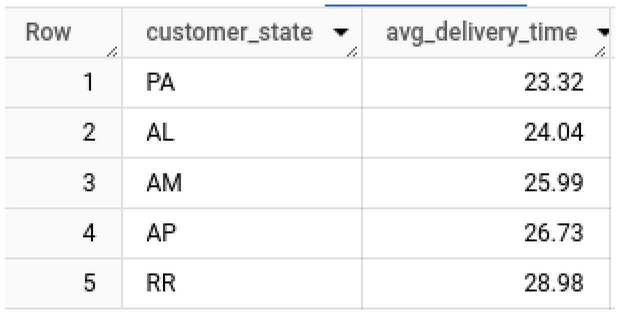


Row	customer_state	avg_freight_value
1	SP	17.37
2	MG	23.46
3	PR	23.58
4	DF	23.82
5	RJ	23.95

Insights: The top five states with lowest average freight value.

```
#C. Find out the top 5 states with the highest & lowest average delivery time.
select
t.customer_state,t.avg_delivery_time
from(
select
customer_state,
round(avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day)),2) as
avg_delivery_time
from `target_sql.orders`
join `target_sql.customers` using(customer_id)
group by customer_state
order by avg_delivery_time desc
limit 5
) t
order by avg_delivery_time;
```

Output:

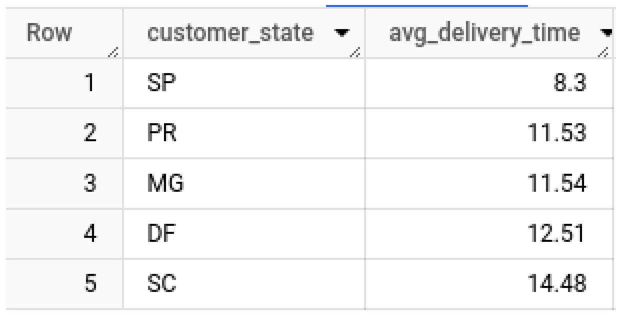


Row	customer_state	avg_delivery_time
1	PA	23.32
2	AL	24.04
3	AM	25.99
4	AP	26.73
5	RR	28.98

Insights: the top 5 states with average delivery time in increasing order.

```
select
customer_state,
round(avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day)),2) as
avg_delivery_time
from `target_sql.orders`
join `target_sql.customers` using(customer_id)
group by customer_state
order by avg_delivery_time
limit 5;
```

Output:

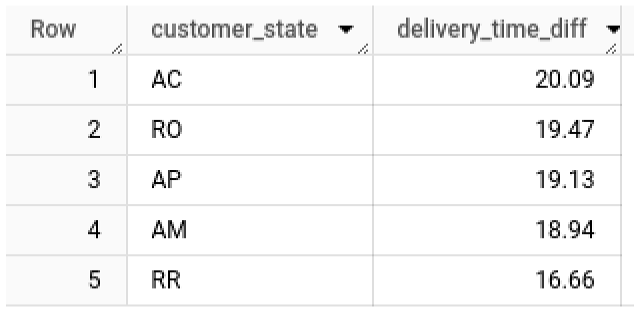


Row	customer_state	avg_delivery_time
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48

Insights:The top 5 states with lowest delivery time in increasing order.

```
#D. Find out the top 5 states where the order delivery is really fast as compared to
the estimated date of delivery.
select distinct t.customer_state,
round(t.estimated_delivery-t.delivery_time,2) as delivery_time_diff
from(
select
distinct customer_state,
avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,day)) as
delivery_time,
avg(date_diff(order_estimated_delivery_date,order_purchase_timestamp,day)) as
estimated_delivery
from `target_sql.orders`
join `target_sql.customers`using(customer_id)
where order_status = 'delivered'and order_estimated_delivery_date is not null and
order_delivered_customer_date is not null
group by customer_state
) t
order by delivery_time_diff desc
limit 5;
```

Output:



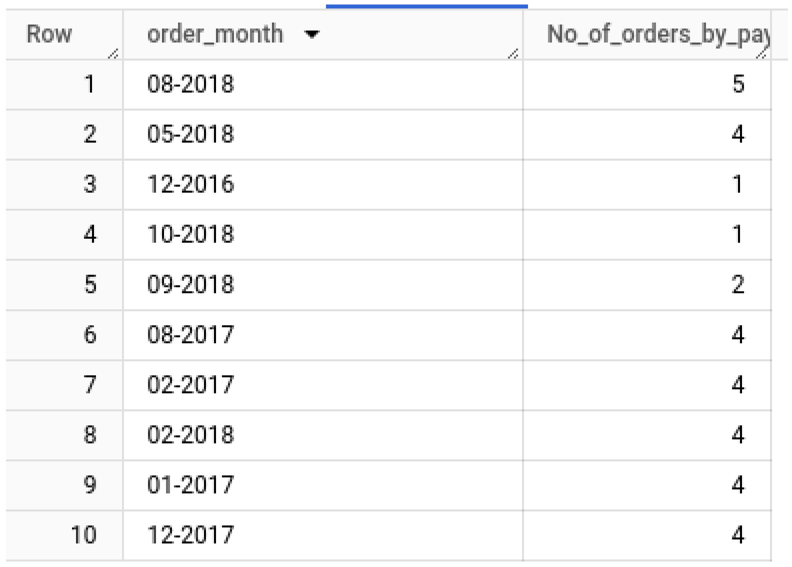
Row	customer_state	delivery_time_diff
1	AC	20.09
2	RO	19.47
3	AP	19.13
4	AM	18.94
5	RR	16.66

Insights: The top 5 states which delivery time is fast when compared to estimated time.

```
#VI. Analysis based on the payments:
#A. Find the month on month no. of orders placed using different payment types.

select
distinct format_datetime('%m-%Y',order_purchase_timestamp) as order_month,
count(distinct payment_type) over(partition by
format_datetime('%m-%Y',order_purchase_timestamp)) as No_of_orders_by_payment_types
from `target_sql.orders`
join `target_sql.payments`using(order_id);
```

Output:

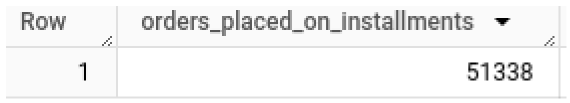


Row	order_month	No_of_orders_by_pay
1	08-2018	5
2	05-2018	4
3	12-2016	1
4	10-2018	1
5	09-2018	2
6	08-2017	4
7	02-2017	4
8	02-2018	4
9	01-2017	4
10	12-2017	4

Insights: month on month orders placed on different payment types.

```
#B. Find the no. of orders placed on the basis of the payment installments that have
been paid.
select
count(order_id) as orders_placed_on_installments
from `target_sql.payments`
where payment_installments > 1;
```

Output:



Row	orders_placed_on_installments
1	51338

Insights: 51338 orders are placed on installments.