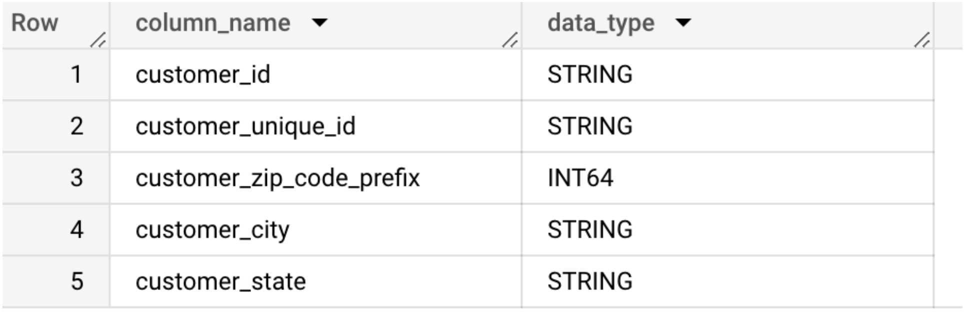
I. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

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

```
SELECT column_name,data_type FROM
`scalar-dsml-sql-class-1.target_sql.INFORMATION_SCHEMA.COLUMNS`
where table_name = "customers"
```

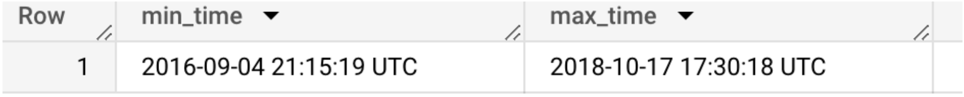


Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

```
Inference/Insights :- There are column names like
customer_id,customer_unique_id,customer_city and customer_state having string
data_type and customer_zip_code_prefix having integer data type in customers table
schema.
```

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

```
SELECT min(order_purchase_timestamp)as min_time,max(order_purchase_timestamp) as
max_time FROM `scalar-dsml-sql-class-1.target_sql.orders`
```

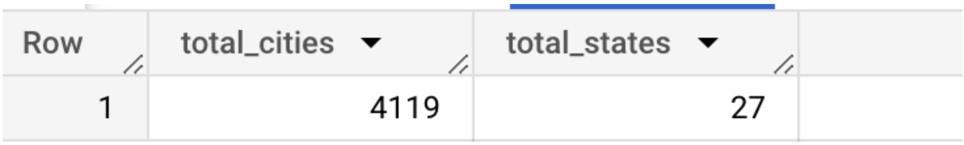


Row	min_time	max_time
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

```
Inference/Insights:- The orders are placed between 2016-09-04 21:15:19 and 2018-10-17
17:30:18
```

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

```
SELECT count(distinct customer_city) as total_cities,count(distinct customer_state) as
total_states FROM `scalar-dsml-sql-class-1.target_sql.customers`
```



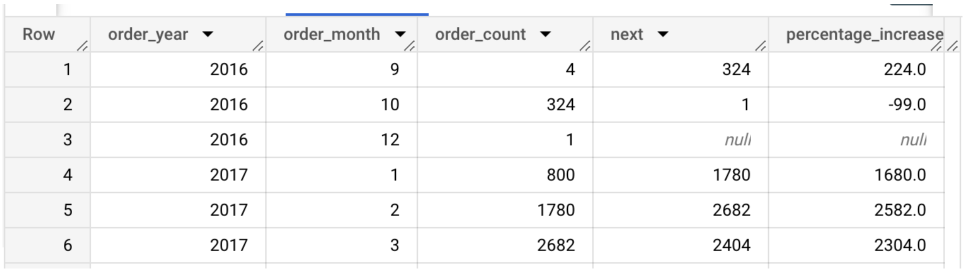
Row	total_cities	total_states
1	4119	27

Inference/Insights :- The total number of cities and states from which the cutomers order are 4119 and 27 respectively

II.In-depth Exploration:

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

```
with cte as (select extract(year from order_purchase_timestamp) as order_year,extract
(month from order_purchase_timestamp) as order_month,count(*) as order_count from
`scalar-dsml-sql-class-1.target_sql.orders`
group by order_year,order_month )
select *,lead(order_count) over(partition by order_year order by order_month) as next,
(lead(order_count,1) over (partition by order_year order by order_month))- order_count
/ order_count * 100 as percentage_increase
from cte
order by order_year,order_month
```

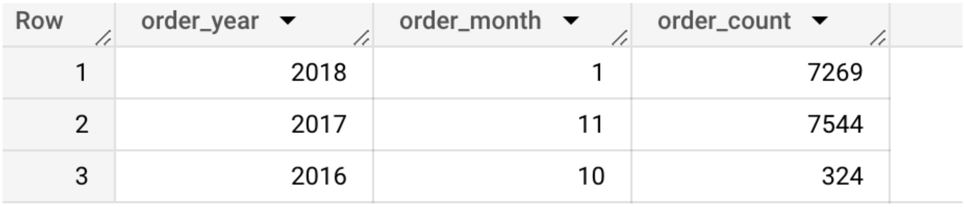


Row	order_year	order_month	order_count	next	percentage_increase
1	2016	9	4	324	224.0
2	2016	10	324	1	-99.0
3	2016	12	1	null	null
4	2017	1	800	1780	1680.0
5	2017	2	1780	2682	2582.0
6	2017	3	2682	2404	2304.0

Inference/Insights :-
1)In the year of 2016,there is a percentage increase in the number of orders from 9th to 10th month by 224% and then from 10 to 12 there is a 99% decrease in the number of orders.
2) In the year of 2017 there was a percentage increase from month on month constantly.

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

```
with cte as(select extract(year from order_purchase_timestamp) as order_year,
extract(month from order_purchase_timestamp) as order_month,count(*) as order_count
from `scalar-dsml-sql-class-1.target_sql.orders`
group by order_year,order_month),
cte_2 as (select order_year,order_month,order_count,dense_rank() over(partition by
order_year order by order_count desc) as ranked from cte)
select order_year,order_month,order_count from cte_2
where ranked = 1
```



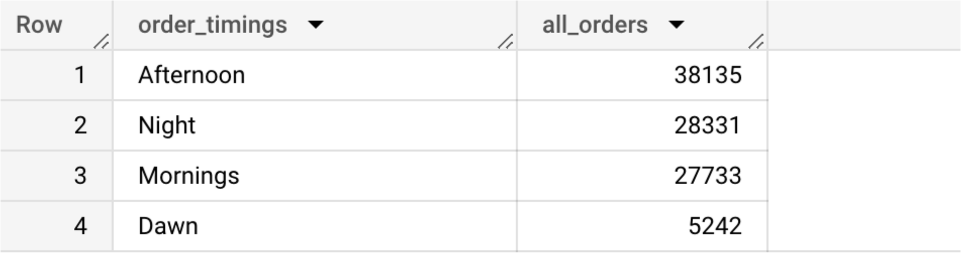
Row	order_year	order_month	order_count
1	2018	1	7269
2	2017	11	7544
3	2016	10	324

```
Insights:-
1)The number of orders placed in the month of January in 2018 is highest with 7269
orders.
2)The number of orders placed in the month of November in 2017 is highest with 7544
orders.
3) The number of orders placed in the month of october in 2018 is highest with 324
orders.
```

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

```
with cte as(
SELECT extract (hour from order_purchase_timestamp) as hour_of_order,count(order_id)
as num_orders from `scalar-dsml-sql-class-1.target_sql.orders`
group by 1),
cte_2 as(select *,
CASE
WHEN hour_of_order BETWEEN 0 and 6 THEN "Dawn"
WHEN hour_of_order between 7 and 12 THEN "Mornings"
WHEN hour_of_order between 13 and 18 THEN "Afternoon"
WHEN hour_of_order between 19 and 23 THEN "Night"
END AS order_timings from cte)
select order_timings,sum(num_orders) as all_orders from cte_2
group by 1
order by 2 desc
```



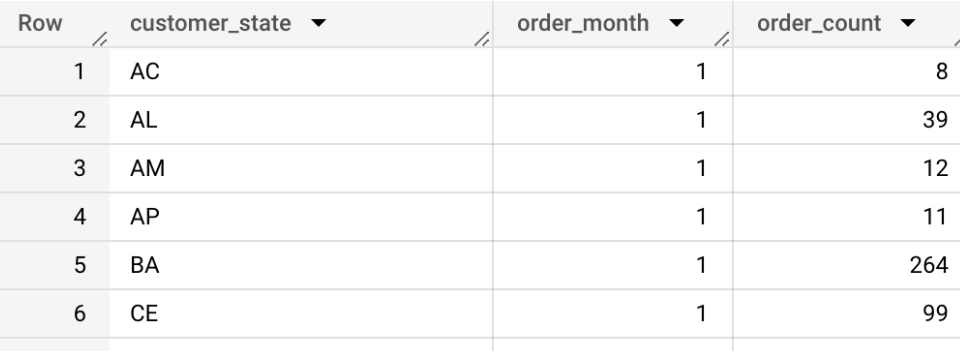
Row	order_timings	all_orders
1	Afternoon	38135
2	Night	28331
3	Mornings	27733
4	Dawn	5242

Inference:- The number of orders that are placed is highest in the afternoons in Brazil with 38135 orders.

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

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

```
with cte as(select * from `scalar-dsml-sql-class-1.target_sql.customers`c
left join `scalar-dsml-sql-class-1.target_sql.orders`o
on c.customer_id = o.customer_id)
SELECT customer_state,
extract(month from order_purchase_timestamp) as order_month,count(order_id) as
order_count
FROM cte
group by order_month,customer_state
order by order_month,customer_state
```



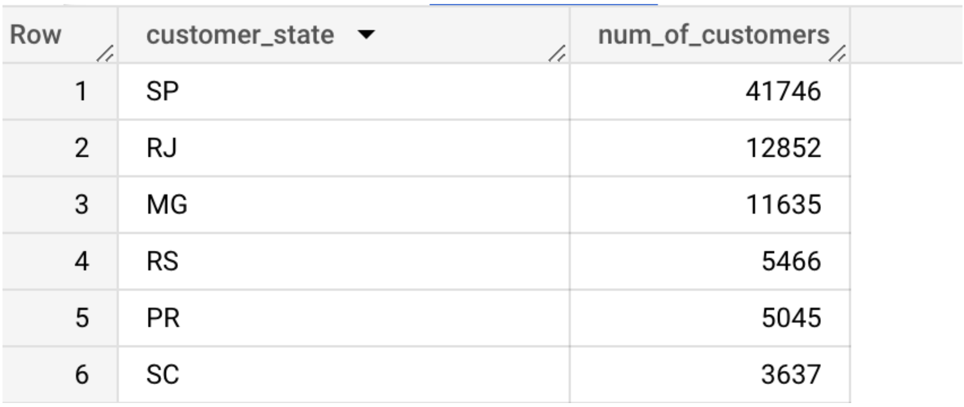
Row	customer_state	order_month	order_count
1	AC	1	8
2	AL	1	39
3	AM	1	12
4	AP	1	11
5	BA	1	264
6	CE	1	99

Insights :-
The number of customers in the month of January from the states AC,AL,AM are 8,39,12 respectively and so on

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

```
select customer_state,count(distinct customer_id) as num_of_customers from
`scalar-dsml-sql-class-1.target_sql.customers`
group by customer_state
```

```
order by num_of_customers desc
```



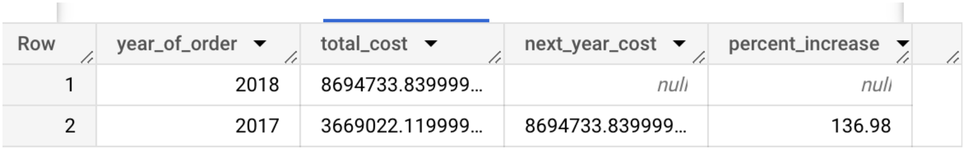
Row	customer_state	num_of_customers
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637

Insights:-The number of customers in the Customer_state SP is the highest with 41,746 customers

IV.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).

```
with cte_1 as
(select extract(year from o.order_purchase_timestamp)as year_of_order,extract(month
from o.order_purchase_timestamp) as month_of_order,sum(p.payment_value) as
cost_of_orders
from `target_sql.orders`o
join `target_sql.payments`p
on o.order_id = p.order_id
where extract(year from o.order_purchase_timestamp) between 2017 and 2018 and
extract(month from o.order_purchase_timestamp) between 1 and 8
group by 1,2
order by 1,2
),
cte_2 as
(select year_of_order,sum(cost_of_orders) as total_cost from cte_1
group by 1 ),
next_row as
(select *,lead(total_cost,1) over(order by year_of_order asc) as next_year_cost from
cte_2)
select *,round((next_year_cost-total_cost)/total_cost *100,2) as percent_increase from
next_row
```

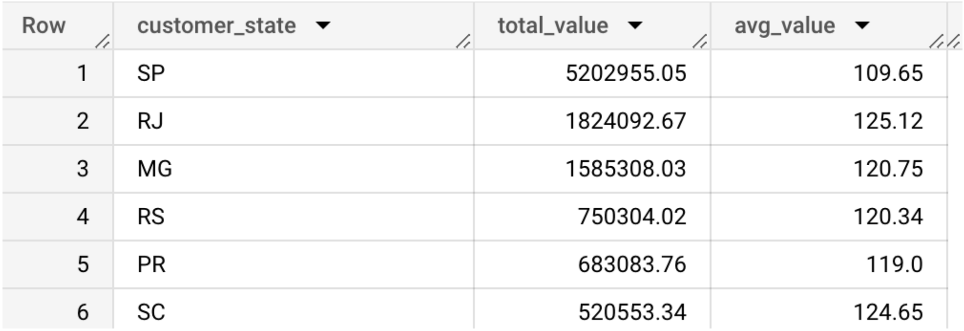


Row	year_of_order	total_cost	next_year_cost	percent_increase
1	2018	8694733.839999...	null	null
2	2017	3669022.119999...	8694733.839999...	136.98

Insights/Inference:- There is a 137% increase in the cost of orders from 2017 to 2018

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

```
select
customer_state,round(sum(price),2) as total_value,round(avg(price),2) as avg_value
from `scalar-dsml-sql-class-1.target_sql.customers` c
left join `scalar-dsml-sql-class-1.target_sql.orders` o
on c.customer_id = o.customer_id
left join `scalar-dsml-sql-class-1.target_sql.order_items` t
on o.order_id = t.order_id
group by customer_state
order by total_value desc,avg_value desc
```



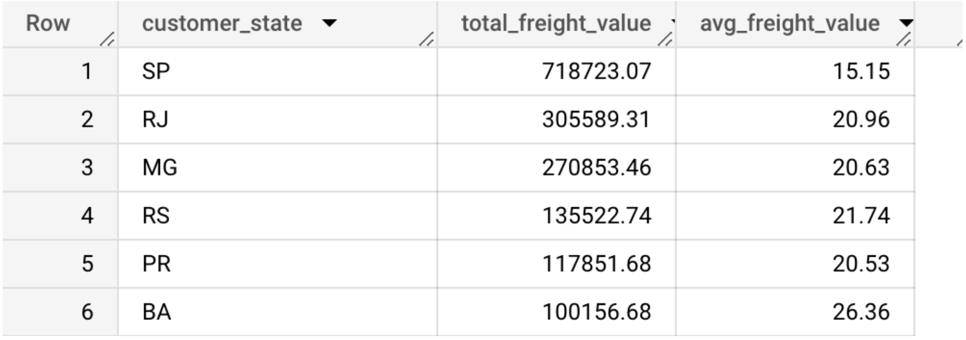
Row	customer_state	total_value	avg_value
1	SP	5202955.05	109.65
2	RJ	1824092.67	125.12
3	MG	1585308.03	120.75
4	RS	750304.02	120.34
5	PR	683083.76	119.0
6	SC	520553.34	124.65

Insights:-
The total order price and average order price of customer_state SP are 5202955.05 and 109.65 respectively.similarly for other states are retrieved.

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

```
select
customer_state,round(sum(freight_value),2) as
total_freight_value,round(avg(freight_value),2) as avg_freight_value from
`scalar-dsml-sql-class-1.target_sql.customers` c
left join `scalar-dsml-sql-class-1.target_sql.orders` o
on c.customer_id = o.customer_id
left join `scalar-dsml-sql-class-1.target_sql.order_items` t
```

```
on o.order_id = t.order_id
group by customer_state
order by total_freight_value desc,avg_freight_value desc
```



Row	customer_state	total_freight_value	avg_freight_value
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	MG	270853.46	20.63
4	RS	135522.74	21.74
5	PR	117851.68	20.53
6	BA	100156.68	26.36

Insights:-The total freight value and average freight value for the state SP are 718723.07,15.15 respectively.similarly for other states are retrieved.

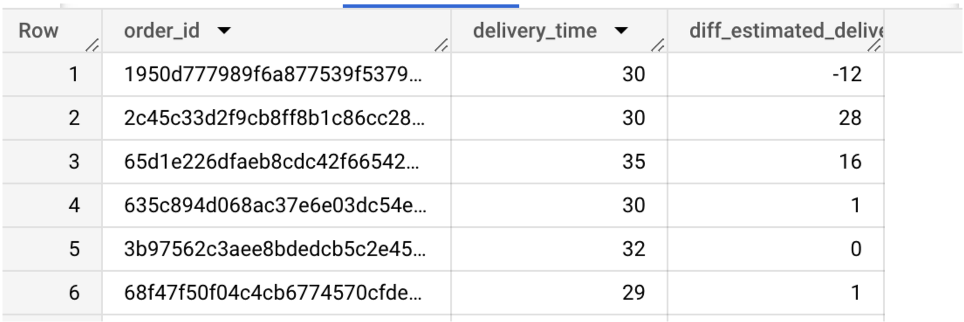
V.Analysis based on sales, freight and delivery time.

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_estimated_delivery_date - order_delivered_customer_date

```
SELECT order_id,
timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day) as
delivery_time,
timestamp_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as
diff_estimated_delivery
FROM `scalar-dsml-sql-class-1.target_sql.orders`
```

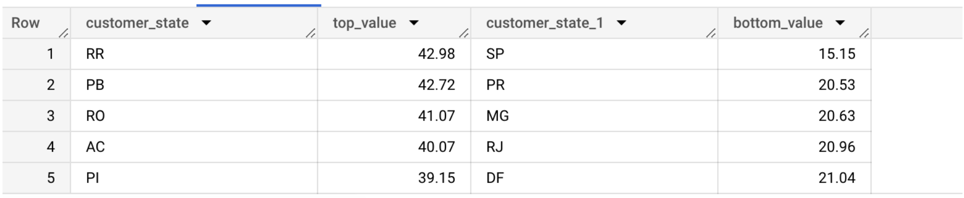


Row	order_id	delivery_time	diff_estimated_delive
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	28
3	65d1e226dfaeb8cdc42f66542...	35	16
4	635c894d068ac37e6e03dc54e...	30	1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	1

Insights:-The delivery time and difference between estimated and actual delivery time for order_id 1 is 30 days and -12 days respectively and so on

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

```
with cte as(select
round(avg(freight_value),2) as avg_freight_value,customer_state from
`scalar-dsml-sql-class-1.target_sql.customers` c
left join `scalar-dsml-sql-class-1.target_sql.orders` o
on c.customer_id = o.customer_id
left join `scalar-dsml-sql-class-1.target_sql.order_items` t
on o.order_id = t.order_id
group by customer_state)
select a.customer_state,a.avg_freight_value as top_value
,b.customer_state,b.avg_freight_value as bottom_value from
(select customer_state,avg_freight_value,
row_number()over(order by avg_freight_value desc) as top_five from cte)a
inner join
(select customer_state,avg_freight_value,
row_number() over (order by avg_freight_value asc)as bottom_five from cte) b
on a.top_five = b.bottom_five and a.top_five<= 5 and b.bottom_five <= 5
```

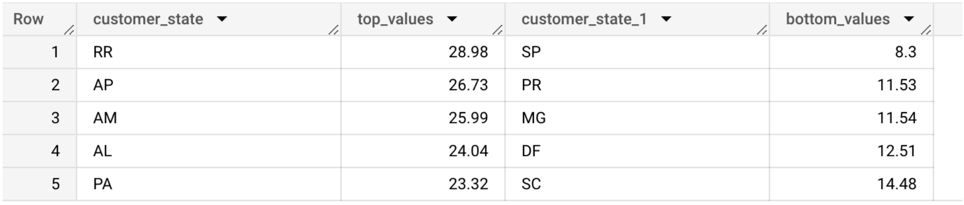


Row	customer_state	top_value	customer_state_1	bottom_value
1	RR	42.98	SP	15.15
2	PB	42.72	PR	20.53
3	RO	41.07	MG	20.63
4	AC	40.07	RJ	20.96
5	PI	39.15	DF	21.04

Insights:-
 1) The top five states having highest averge frieght value are RR,PB,RO,AC,PI with 42.98 as highest average freight value for RR state.
2) The bottom five states with lowest average freight value are SP,PR,MG,RJ,DF with 15.15 is the lowest average freight value for SP state.

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

```
with cte as (SELECT c.customer_state,
round(avg(timestamp_diff(order_delivered_customer_date,order_purchase_timestamp,day)),
2) as avg_delivery_time from `target_sql.orders`o
join `target_sql.customers`c
on c.customer_id = o.customer_id
group by 1 )
select a.customer_state,a.avg_delivery_time as
top_values,b.customer_state,b.avg_delivery_time as bottom_values from (
select customer_state,avg_delivery_time,row_number() over (order by avg_delivery_time
desc)as top_five from cte)a
inner join
(select customer_state,avg_delivery_time,row_number()over (order by avg_delivery_time
asc) as bottom_five from cte)b
on a.top_five = b.bottom_five and a.top_five <= 5 and b.bottom_five <= 5
```



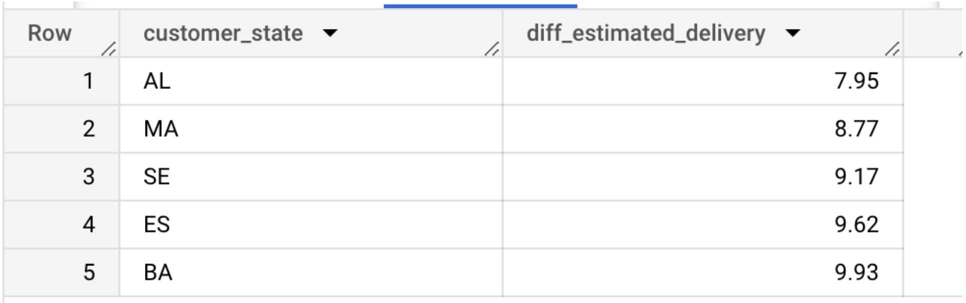
Row	customer_state	top_values	customer_state_1	bottom_values
1	RR	28.98	SP	8.3
2	AP	26.73	PR	11.53
3	AM	25.99	MG	11.54
4	AL	24.04	DF	12.51
5	PA	23.32	SC	14.48

```
Insights:-
1) The top five states having highest averge  delivery time   are RR,AP,AM,Al,PA with
28.98 as highest average delivery time for RR state.
2) The bottom five states with lowest average freight value are SP,PR,MG,DF,SC with
8.3 as the lowest average freight value for SP state.
```

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 customer_state,
round(avg(timestamp_diff(order_estimated_delivery_date,order_delivered_customer_date,d
ay)),2) as diff_estimated_delivery
FROM `scalar-dsml-sql-class-1.target_sql.orders` o
left join `target_sql.customers`c
on c.customer_id = o.customer_id
where order_delivered_customer_date is not null
group by 1
order by 2 asc
```

```
limit 5
```



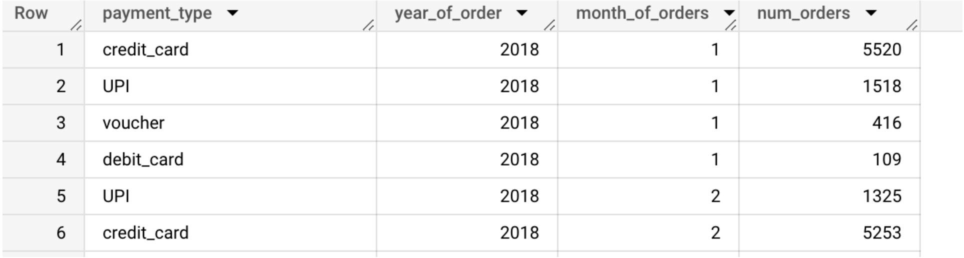
Row	customer_state	diff_estimated_delivery
1	AL	7.95
2	MA	8.77
3	SE	9.17
4	ES	9.62
5	BA	9.93

Insights:- The top 5 states having fastest order delivery are AL,MA,SE,ES,BA in which state AL having lowest delivery time with 7.95 days

VI.Analysis based on the payments:

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

```
select payment_type,extract(year from order_purchase_timestamp) as
year_of_order,extract(month from order_purchase_timestamp) as
month_of_orders,count(o.order_id) as num_orders from `target_sql.orders`o
join `target_sql.payments`p
on o.order_id = p.order_id
group by 1,2,3
order by 3
```



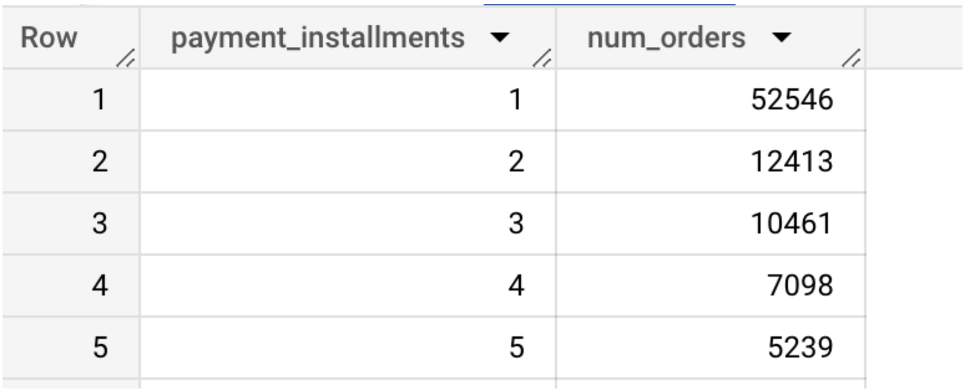
Row	payment_type	year_of_order	month_of_orders	num_orders
1	credit_card	2018	1	5520
2	UPI	2018	1	1518
3	voucher	2018	1	416
4	debit_card	2018	1	109
5	UPI	2018	2	1325
6	credit_card	2018	2	5253

Insights:-

1. The number of orders in the month of january in 2018 using credit_card are 5520.
2. The number of orders in the month of january in 2018 using UPI are 1516.
3. The number of orders in the month of january in 2018 using voucher are 416.
4. The number of orders in the month of january in 2018 using debit_card are 109.

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

```
select payment_installments,count(o.order_id) as num_orders from `target_sql.orders`o
join `target_sql.payments` p
on o.order_id = p.order_id
where payment_installments <> 0
group by 1
order by 1
```



Row	payment_installments	num_orders
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	5	5239

Insights:- 1) one payment installment has been paid for 52546 orders.
2) Two payment installments have been paid for 12413 orders and so on