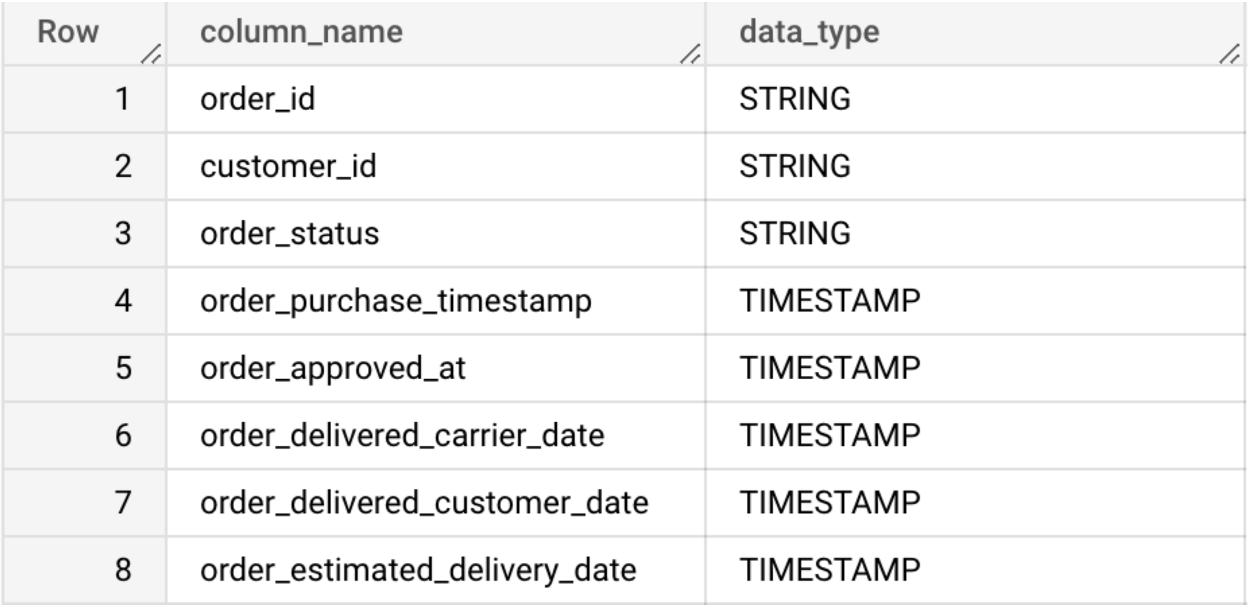
TARGET SQL BUSINESS CASE STUDY

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

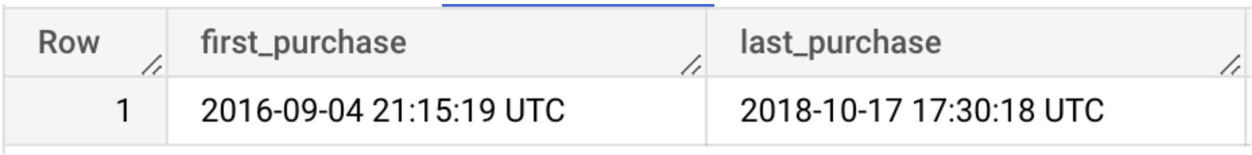
1. Data type of columns in a table
2. Time period for which the data is given
3. Cities and States of customers ordered during the given period

```
SELECT
 column_name,
 data_type
FROM
 Target.INFORMATION_SCHEMA.COLUMNS
WHERE
 table_name = "orders"
```



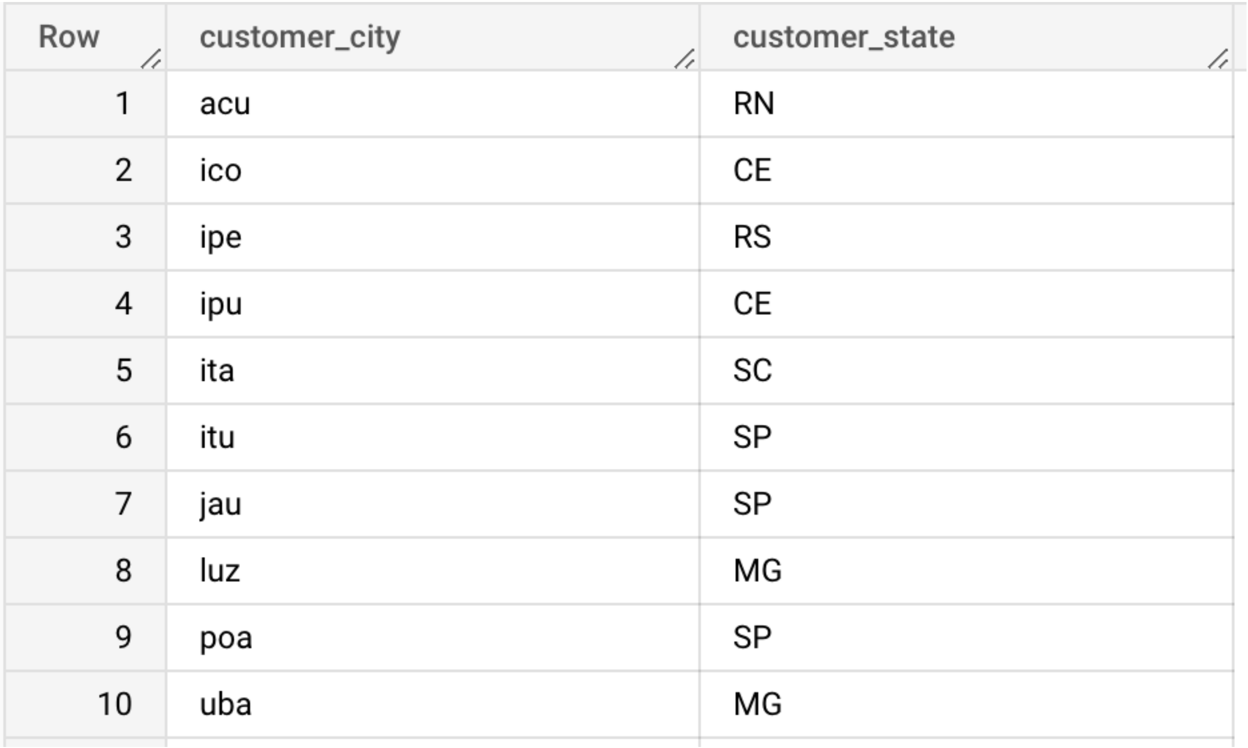
Row	column_name	data_type
1	order_id	STRING
2	customer_id	STRING
3	order_status	STRING
4	order_purchase_timestamp	TIMESTAMP
5	order_approved_at	TIMESTAMP
6	order_delivered_carrier_date	TIMESTAMP
7	order_delivered_customer_date	TIMESTAMP
8	order_estimated_delivery_date	TIMESTAMP

```
SELECT
 MIN(order_purchase_timestamp) AS first_purchase,
 MAX(order_purchase_timestamp) AS last_purchase
FROM
 `Target.orders`
```



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

```
select distinct customer_city, customer_state
from `Target.customers`
```

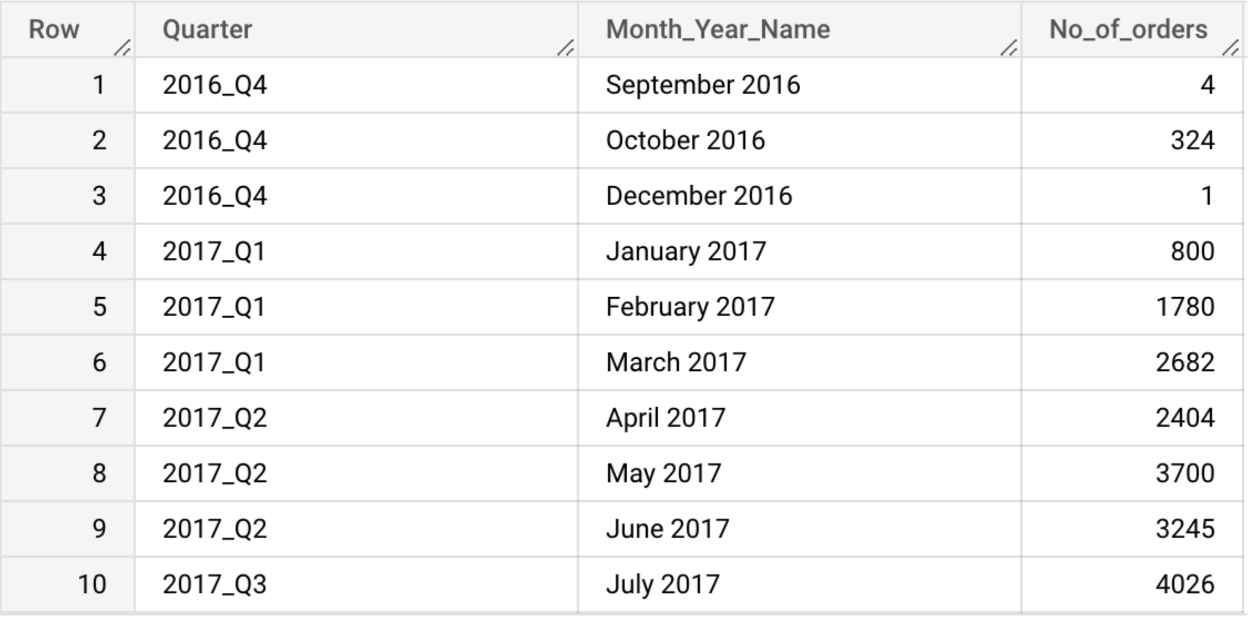


Row	customer_city	customer_state
1	acu	RN
2	ico	CE
3	ipe	RS
4	ipu	CE
5	ita	SC
6	itu	SP
7	jau	SP
8	luz	MG
9	poa	SP
10	uba	MG

2. In-depth Exploration:

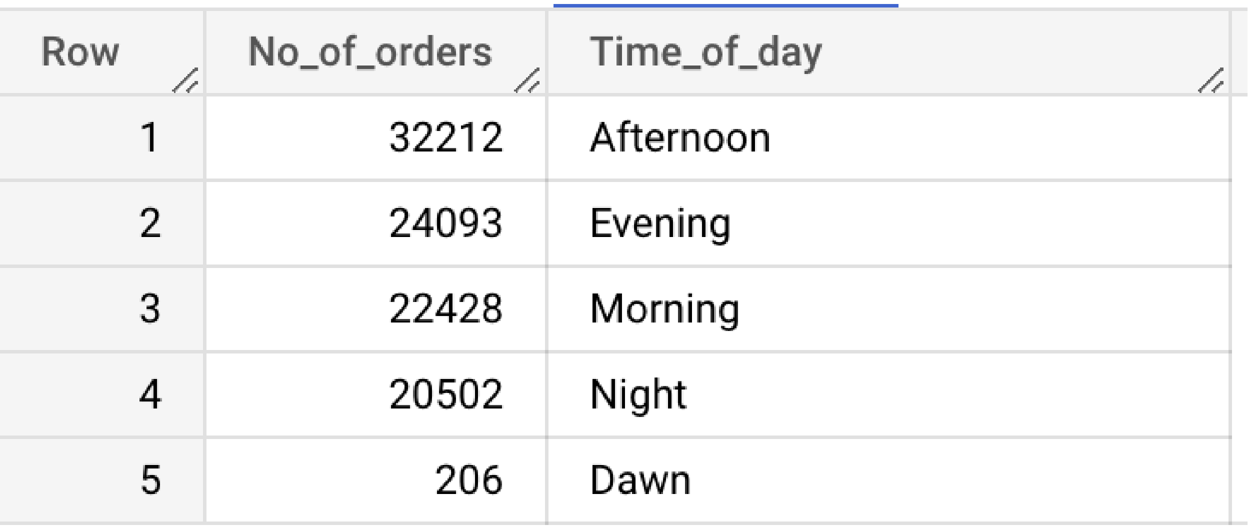
1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?
2. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

```
select t1.Quarter, t1.Month_Year_Name, t1.No_of_orders from (
select
count(order_id) as No_of_orders,
extract(year from order_purchase_timestamp) as year, extract(month from
order_purchase_timestamp) as month,
format_date('%B %Y', order_purchase_timestamp) as Month_Year_Name,
case
when (extract(month from order_purchase_timestamp) between 9 and 12) and extract(year from
order_purchase_timestamp) = 2016 then '2016_Q4'
when (extract(month from order_purchase_timestamp) between 1 and 3) and extract(year from
order_purchase_timestamp) = 2017 then '2017_Q1'
when (extract(month from order_purchase_timestamp) between 4 and 6) and extract(year from
order_purchase_timestamp) = 2017 then '2017_Q2'
when (extract(month from order_purchase_timestamp) between 7 and 9) and extract(year from
order_purchase_timestamp) = 2017 then '2017_Q3'
when (extract(month from order_purchase_timestamp) between 10 and 12) and extract(year from
order_purchase_timestamp) = 2017 then '2017_Q4'
when (extract(month from order_purchase_timestamp) between 1 and 3) and extract(year from
order_purchase_timestamp) = 2018 then '2018_Q1'
when (extract(month from order_purchase_timestamp) between 4 and 6) and extract(year from
order_purchase_timestamp) = 2018 then '2018_Q2'
when (extract(month from order_purchase_timestamp) between 7 and 9) and extract(year from
order_purchase_timestamp) = 2018 then '2018_Q3'
when (extract(month from order_purchase_timestamp) between 10 and 12) and extract(year from
order_purchase_timestamp) = 2018 then '2018_Q4'
end as Quarter
from `Target.orders`
group by year, month, Month_Year_Name, Quarter
) as t1
order by t1.Quarter, t1.year, t1.month;
```



Row	Quarter	Month_Year_Name	No_of_orders
1	2016_Q4	September 2016	4
2	2016_Q4	October 2016	324
3	2016_Q4	December 2016	1
4	2017_Q1	January 2017	800
5	2017_Q1	February 2017	1780
6	2017_Q1	March 2017	2682
7	2017_Q2	April 2017	2404
8	2017_Q2	May 2017	3700
9	2017_Q2	June 2017	3245
10	2017_Q3	July 2017	4026

```
select
count(order_id) as No_of_orders,
case
 when extract (time from order_purchase_timestamp) between '04:00:00' and '05:00:00' then
"Dawn"
 when extract (time from order_purchase_timestamp) between "05:00:00" and "12:00:00" then
"Morning"
 when extract (time from order_purchase_timestamp) between "12:00:00" and "17:00:00" then
"Afternoon"
 when extract (time from order_purchase_timestamp) between "17:00:00" and "21:00:00" then
"Evening"
 else "Night"
end as Time_of_day
from `Target.orders`
group by Time_of_day
order by No_of_orders desc
```

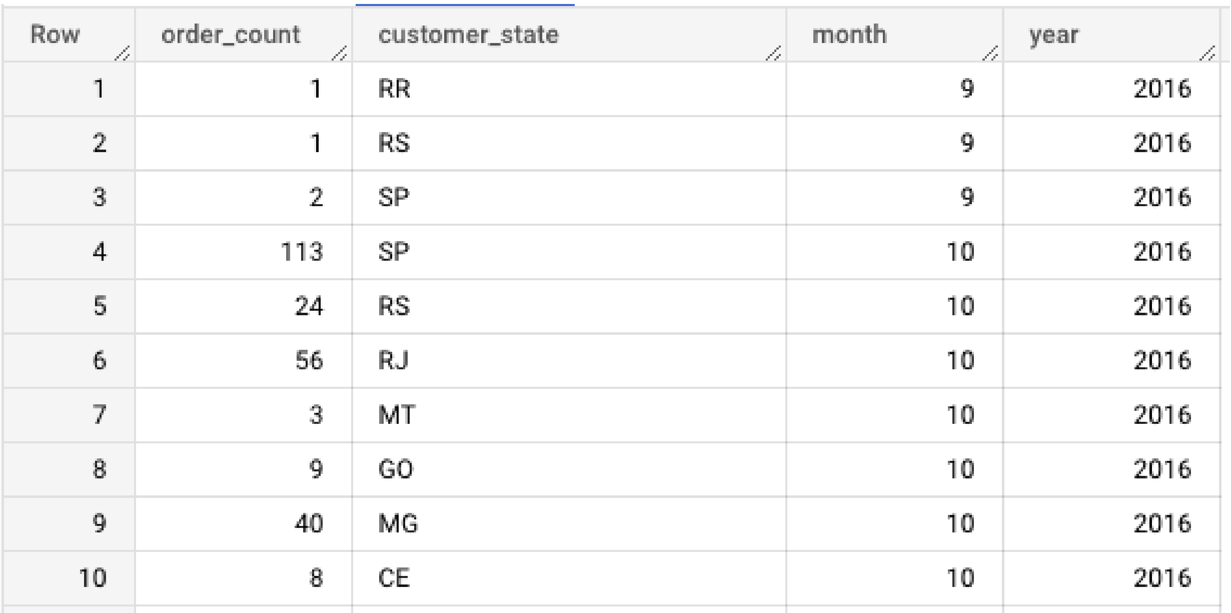


Row	No_of_orders	Time_of_day
1	32212	Afternoon
2	24093	Evening
3	22428	Morning
4	20502	Night
5	206	Dawn

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

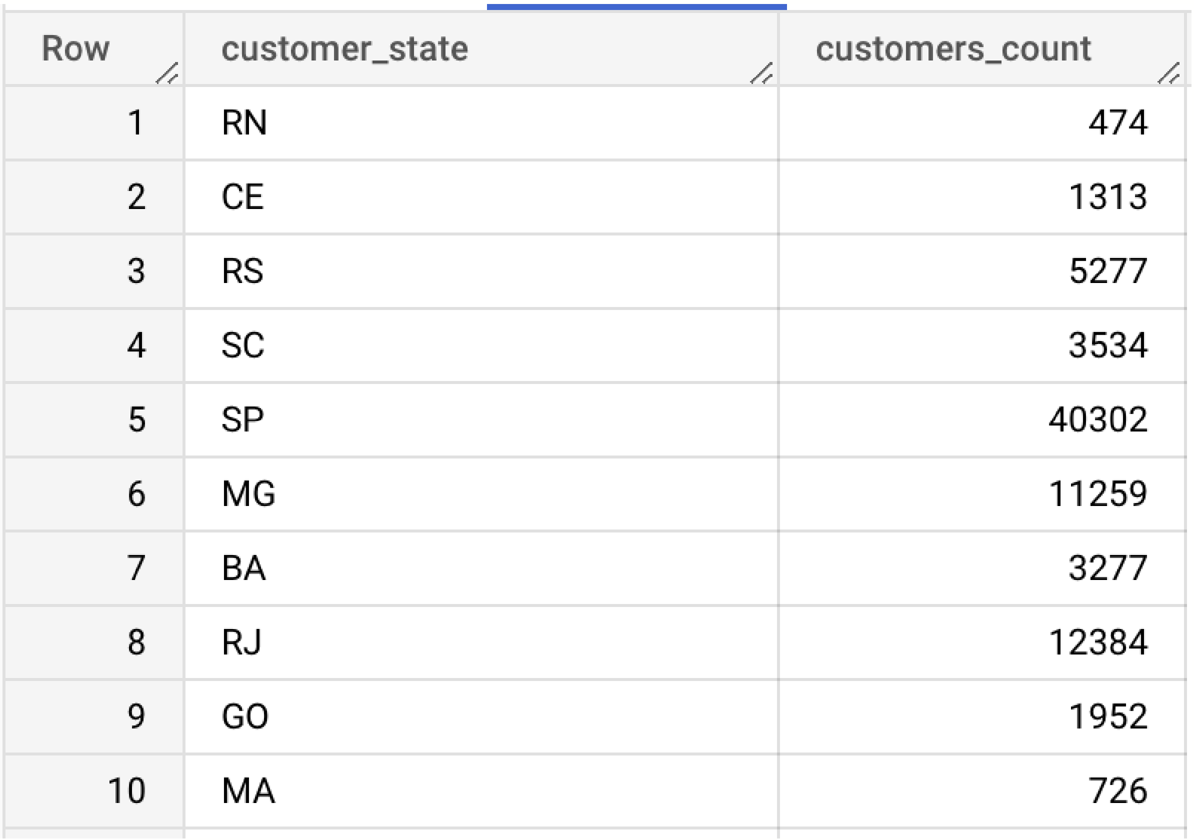
1. Get month on month orders by states
2. Distribution of customers across the states in Brazil

```
with temp as
(select
count(order_id) as order_count,
c.customer_state,
extract (month from order_purchase_timestamp) as month,
extract (year from order_purchase_timestamp) as year
from `Target.orders` as o
left join `Target.customers` as c on c.customer_id = o.customer_id
group by c.customer_state, month, year )
select * from temp
group by order_count, customer_state, month, year
order by year,month
```



Row	order_count	customer_state	month	year
1	1	RR	9	2016
2	1	RS	9	2016
3	2	SP	9	2016
4	113	SP	10	2016
5	24	RS	10	2016
6	56	RJ	10	2016
7	3	MT	10	2016
8	9	GO	10	2016
9	40	MG	10	2016
10	8	CE	10	2016

```
SELECT
 customer_state,
 COUNT( DISTINCT customer_unique_id) AS customers_count
FROM
 `Target.customers`
GROUP BY
 customer_state
```

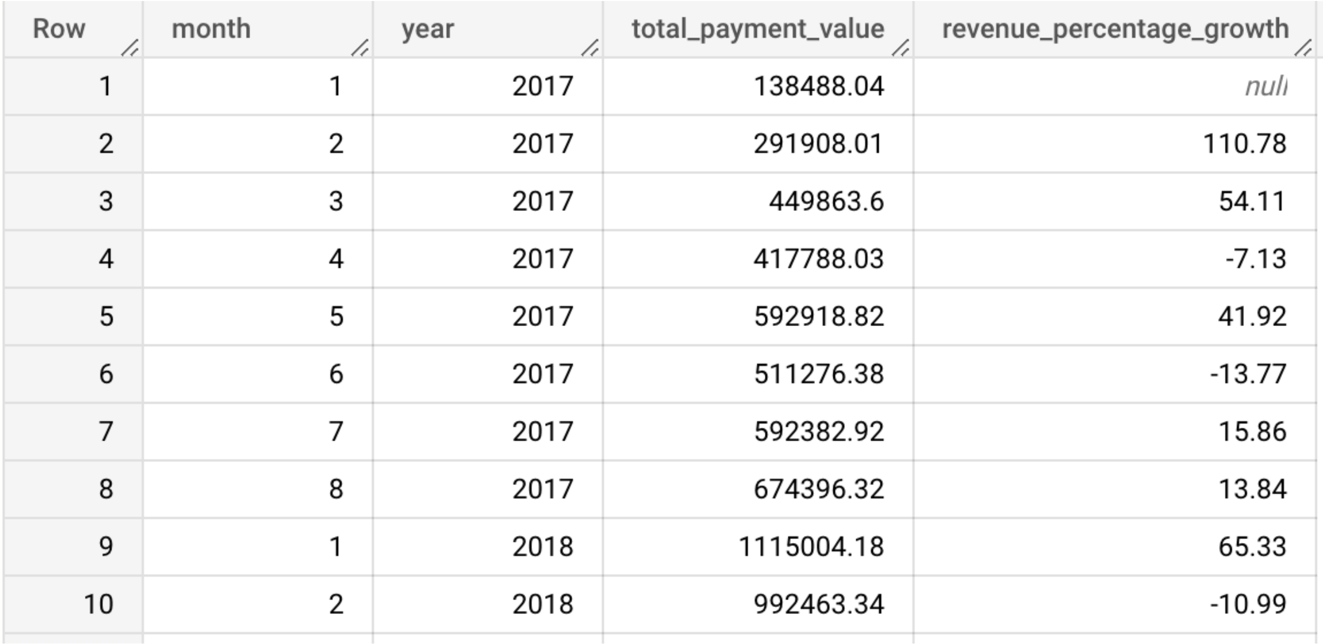


Row	customer_state	customers_count
1	RN	474
2	CE	1313
3	RS	5277
4	SC	3534
5	SP	40302
6	MG	11259
7	BA	3277
8	RJ	12384
9	GO	1952
10	MA	726

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

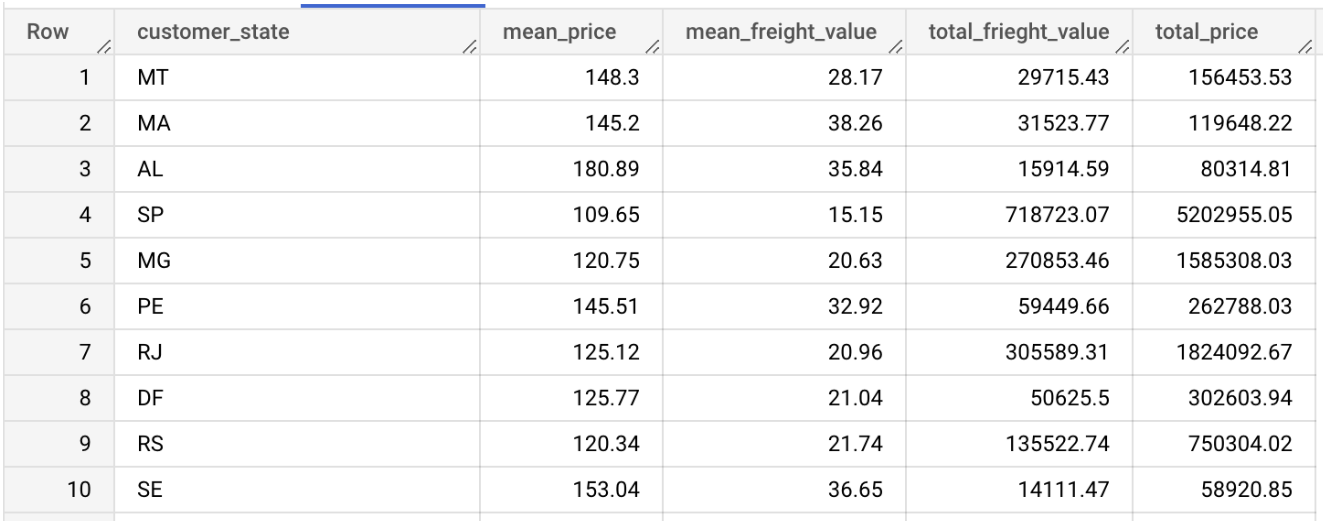
1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table
2. Mean & Sum of price and freight value by customer state

```
select *, round((total_payment_value - LAG (total_payment_value) OVER (ORDER BY year,month
ASC))/LAG (total_payment_value) OVER (ORDER BY year,month ASC)*100,2) AS
revenue_percentage_growth
from
(select
extract (month from order_purchase_timestamp) as month,
extract (year from order_purchase_timestamp) as year,
round(sum(p.payment_value),2) as total_payment_value
from `Target.orders` as o
join `Target.payments` as p on p.order_id = o.order_id
group by year, month
having month between 1 and 8)
order by year, month
```



Row	month	year	total_payment_value	revenue_percentage_growth
1	1	2017	138488.04	*null*
2	2	2017	291908.01	110.78
3	3	2017	449863.6	54.11
4	4	2017	417788.03	-7.13
5	5	2017	592918.82	41.92
6	6	2017	511276.38	-13.77
7	7	2017	592382.92	15.86
8	8	2017	674396.32	13.84
9	1	2018	1115004.18	65.33
10	2	2018	992463.34	-10.99

```
select
c.customer_state,
round(avg(p.price),2) as mean_price,
round(avg(p.freight_value),2) as mean_freight_value,
round(sum(p.freight_value),2) as total_frieght_value,
round(sum(p.price),2) as total_price
from `Target.orders` as o
join `Target.customers`as c on c.customer_id = o.customer_id
join `Target.order_items` as p on p.order_id = o.order_id
where (order_delivered_customer_date, order_purchase_timestamp,
order_estimated_delivery_date) is not null
group by c.customer_state
```



Row	customer_state	mean_price	mean_freight_value	total_frieght_value	total_price
1	MT	148.3	28.17	29715.43	156453.53
2	MA	145.2	38.26	31523.77	119648.22
3	AL	180.89	35.84	15914.59	80314.81
4	SP	109.65	15.15	718723.07	5202955.05
5	MG	120.75	20.63	270853.46	1585308.03
6	PE	145.51	32.92	59449.66	262788.03
7	RJ	125.12	20.96	305589.31	1824092.67
8	DF	125.77	21.04	50625.5	302603.94
9	RS	120.34	21.74	135522.74	750304.02
10	SE	153.04	36.65	14111.47	58920.85

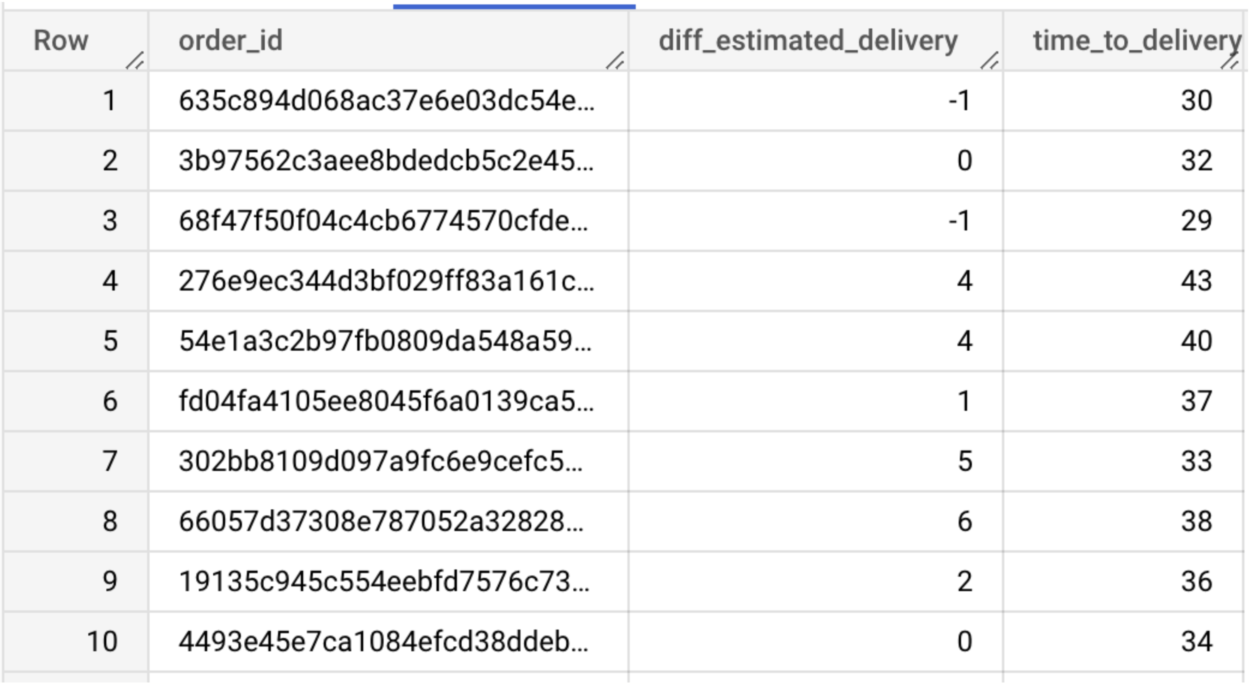
5. Analysis on sales, freight and delivery time

1. Calculate days between purchasing, delivering and estimated delivery
2. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

   time_to_delivery = order_purchase_timestamp-order_delivered_customer_date

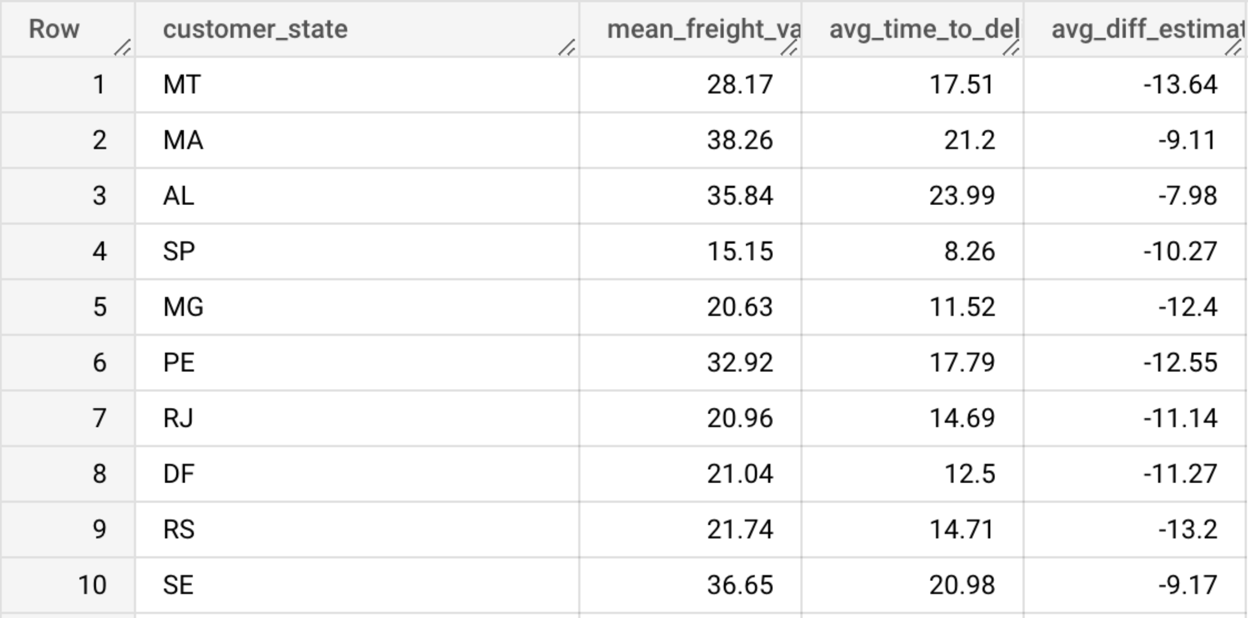
   diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

```
select
order_id,
TIMESTAMP_DIFF (order_delivered_customer_date, order_estimated_delivery_date,day) as
diff_estimated_delivery,
TIMESTAMP_DIFF (order_delivered_customer_date,order_purchase_timestamp,day) as
time_to_delivery
from `Target.orders`
where order_status = "delivered"
```



Row	order_id	diff_estimated_delivery	time_to_delivery
1	635c894d068ac37e6e03dc54e...	-1	30
2	3b97562c3aee8bdedcb5c2e45...	0	32
3	68f47f50f04c4cb6774570cfde...	-1	29
4	276e9ec344d3bf029ff83a161c...	4	43
5	54e1a3c2b97fb0809da548a59...	4	40
6	fd04fa4105ee8045f6a0139ca5...	1	37
7	302bb8109d097a9fc6e9cefc5...	5	33
8	66057d37308e787052a32828...	6	38
9	19135c945c554eebfd7576c73...	2	36
10	4493e45e7ca1084efcd38ddeb...	0	34

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



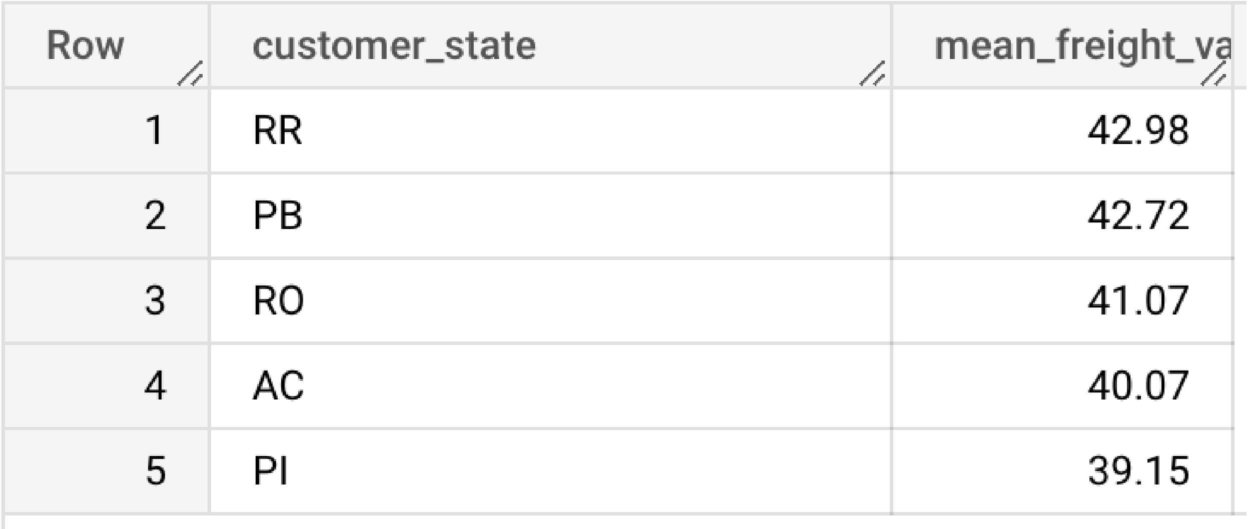
Row	customer_state	mean_freight_va	avg_time_to_del	avg_diff_estimat
1	MT	28.17	17.51	-13.64
2	MA	38.26	21.2	-9.11
3	AL	35.84	23.99	-7.98
4	SP	15.15	8.26	-10.27
5	MG	20.63	11.52	-12.4
6	PE	32.92	17.79	-12.55
7	RJ	20.96	14.69	-11.14
8	DF	21.04	12.5	-11.27
9	RS	21.74	14.71	-13.2
10	SE	36.65	20.98	-9.17

4. Sort the data to get the following:

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

Highest average freight value :

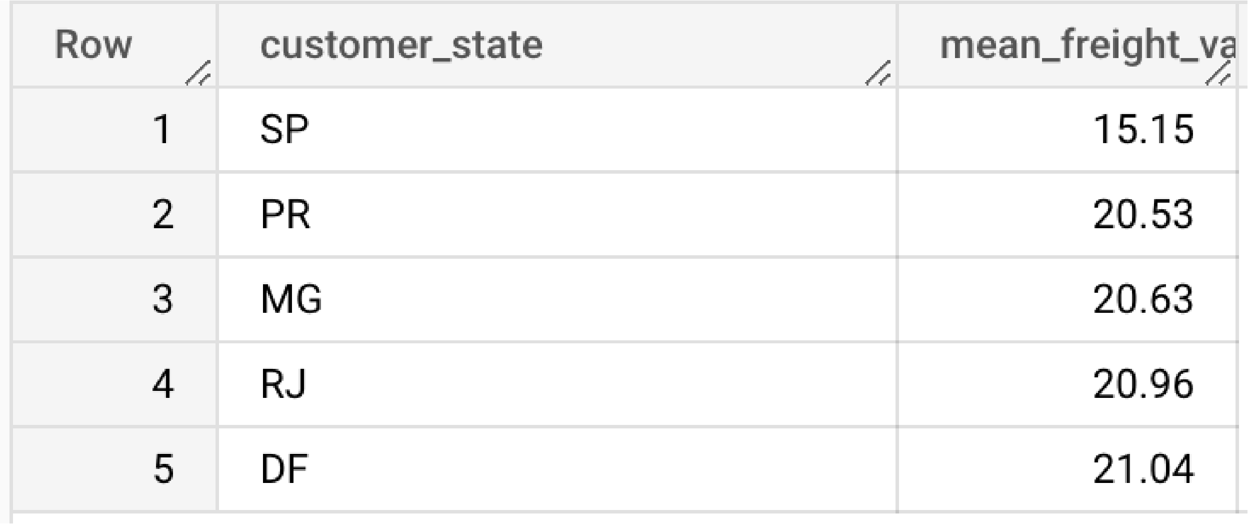
```
select t1.customer_state, t1.mean_freight_value
from
(select
c.customer_state, round(avg(p.freight_value),2) as mean_freight_value
from `Target.orders` as o
join `Target.customers`as c on c.customer_id = o.customer_idavg
join `Target.order_items` as p on p.order_id = o.order_id
where (order_delivered_customer_date, order_purchase_timestamp,
order_estimated_delivery_date) is not null
group by c.customer_state) as t1
order by t1.mean_freight_value desc
limit 5;
```



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

Lowest average freight value :

```
select t1.customer_state, t1.mean_freight_value
from
(select
c.customer_state, round(avg(p.freight_value),2) as mean_freight_value
from `Target.orders` as o
join `Target.customers`as c on c.customer_id = o.customer_id
join `Target.order_items` as p on p.order_id = o.order_id
where (order_delivered_customer_date, order_purchase_timestamp,
order_estimated_delivery_date) is not null
group by c.customer_state) as t1
order by t1.mean_freight_value
limit 5;
```

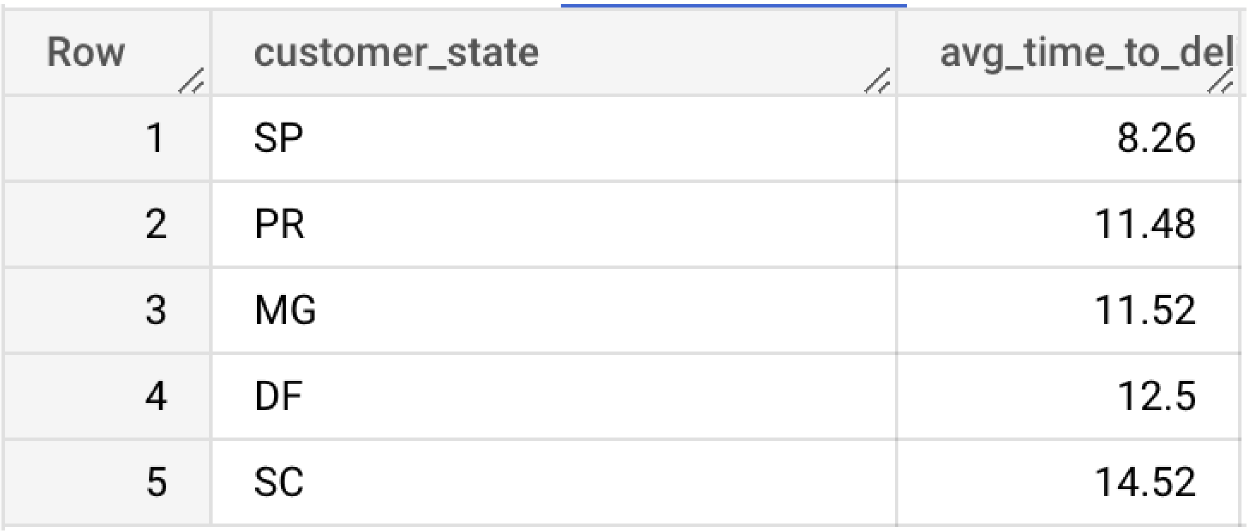


Row	customer_state	mean_freight_va
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

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

Lowest average time to Delivery :

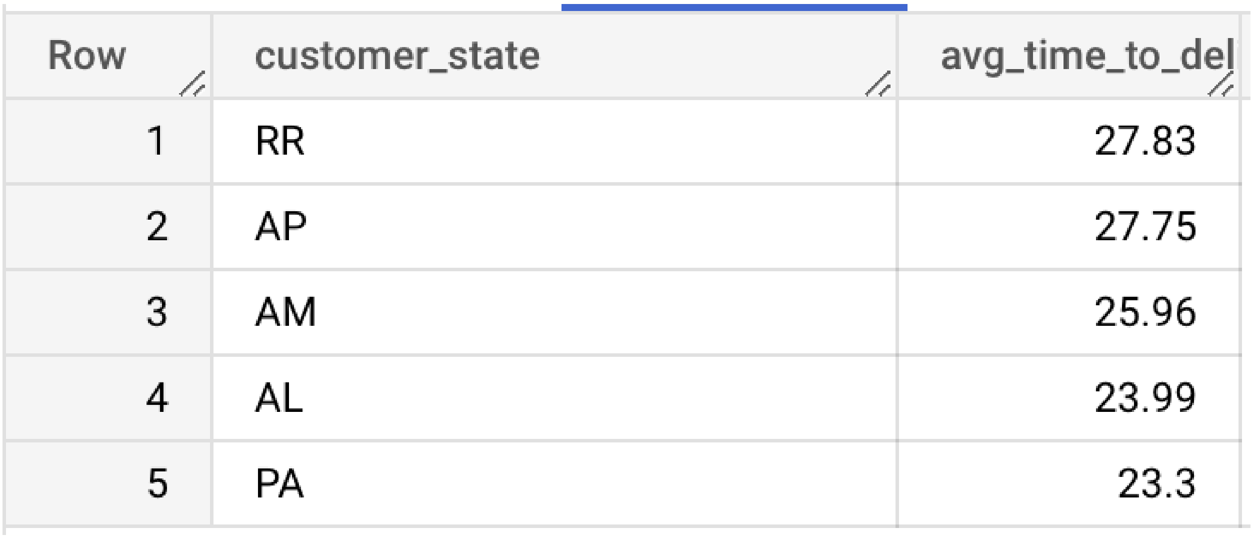
```
select t1.customer_state, t1.avg_time_to_delivery
from
(select
c.customer_state,
round(avg(TIMESTAMP_DIFF (order_delivered_customer_date,order_purchase_timestamp,day)),2)
as avg_time_to_delivery,
from `Target.orders` as o
join `Target.customers`as c on c.customer_id = o.customer_id
join `Target.order_items` as p on p.order_id = o.order_id
where (order_delivered_customer_date, order_purchase_timestamp,
order_estimated_delivery_date) is not null
group by c.customer_state) as t1
order by t1.avg_time_to_delivery
limit 5;
```



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

Highest average time to Delivery :

```
select t1.customer_state, t1.avg_time_to_delivery
from
(select
c.customer_state,
round(avg(TIMESTAMP_DIFF (order_delivered_customer_date,order_purchase_timestamp,day)),2)
as avg_time_to_delivery,
from `Target.orders` as o
join `Target.customers`as c on c.customer_id = o.customer_id
join `Target.order_items` as p on p.order_id = o.order_id
where (order_delivered_customer_date, order_purchase_timestamp,
order_estimated_delivery_date) is not null
group by c.customer_state) as t1
order by t1.avg_time_to_delivery desc
limit 5;
```

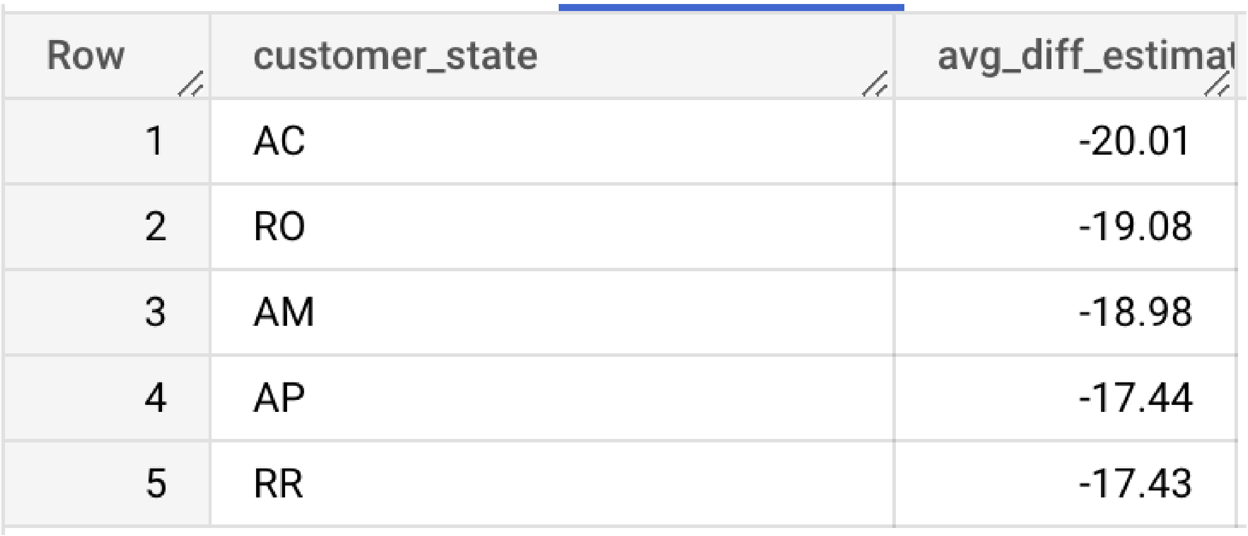


Row	customer_state	avg_time_to_del
1	RR	27.83
2	AP	27.75
3	AM	25.96
4	AL	23.99
5	PA	23.3

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

Not so fast Delivery compared to estimated delivery date :

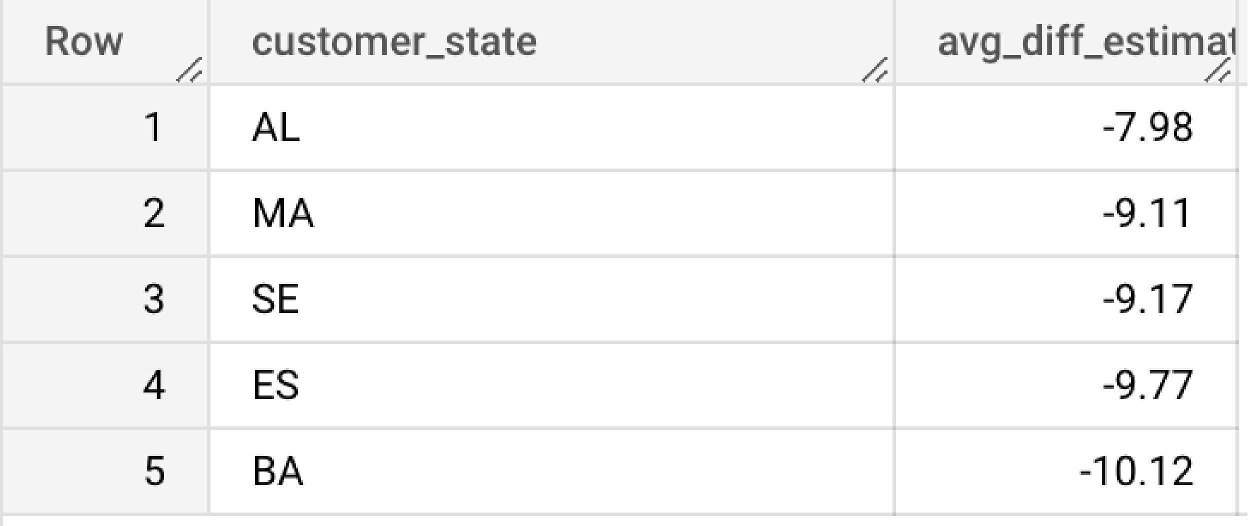
```
select t1.customer_state, t1.avg_diff_estimated_delivery
from
(select
c.customer_state,
round(avg(TIMESTAMP_DIFF (order_delivered_customer_date,
order_estimated_delivery_date,day)),2) as avg_diff_estimated_delivery
from `Target.orders` as o
join `Target.customers`as c on c.customer_id = o.customer_id
join `Target.order_items` as p on p.order_id = o.order_id
where (order_delivered_customer_date, order_purchase_timestamp,
order_estimated_delivery_date) is not null
group by c.customer_state) as t1
order by t1.avg_diff_estimated_delivery
limit 5;
```



Row	customer_state	avg_diff_estimat
1	AC	-20.01
2	RO	-19.08
3	AM	-18.98
4	AP	-17.44
5	RR	-17.43

Fast Delivery compared to estimated delivery date :

```
select t1.customer_state, t1.avg_diff_estimated_delivery
from
(select
c.customer_state,
round(avg(TIMESTAMP_DIFF (order_delivered_customer_date,
order_estimated_delivery_date,day)),2) as avg_diff_estimated_delivery
from `Target.orders` as o
join `Target.customers`as c on c.customer_id = o.customer_id
join `Target.order_items` as p on p.order_id = o.order_id
where (order_delivered_customer_date, order_purchase_timestamp,
order_estimated_delivery_date) is not null
group by c.customer_state) as t1
order by t1.avg_diff_estimated_delivery desc
limit 5;
```

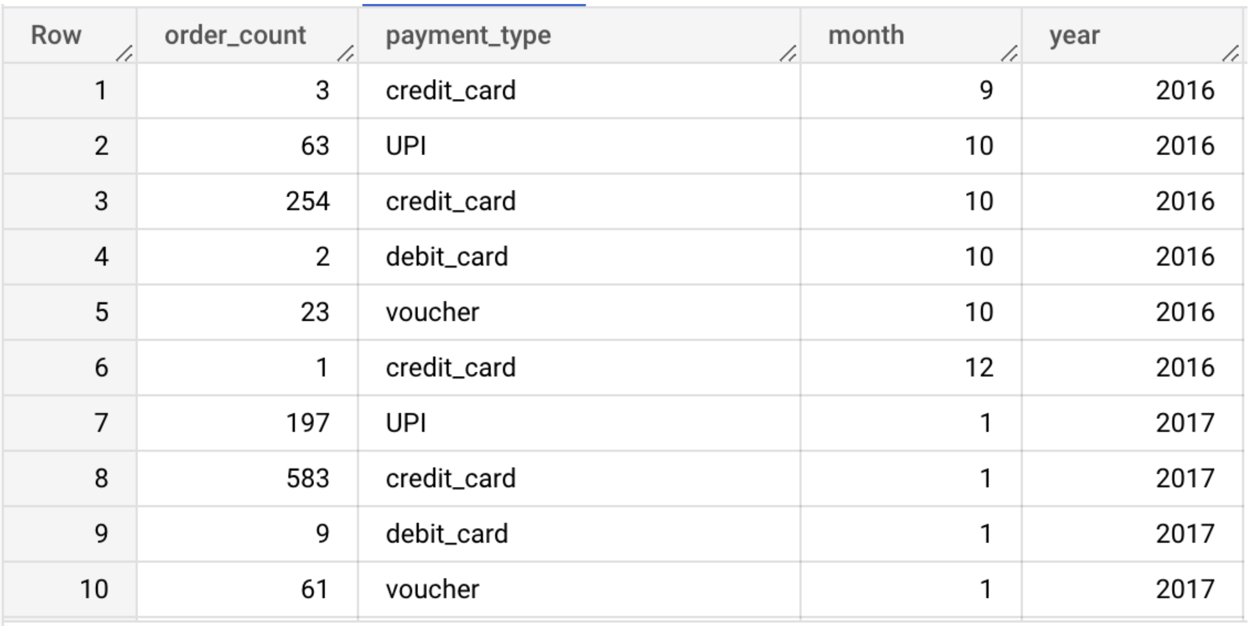


Row	customer_state	avg_diff_estimat
1	AL	-7.98
2	MA	-9.11
3	SE	-9.17
4	ES	-9.77
5	BA	-10.12

6. Payment type analysis:

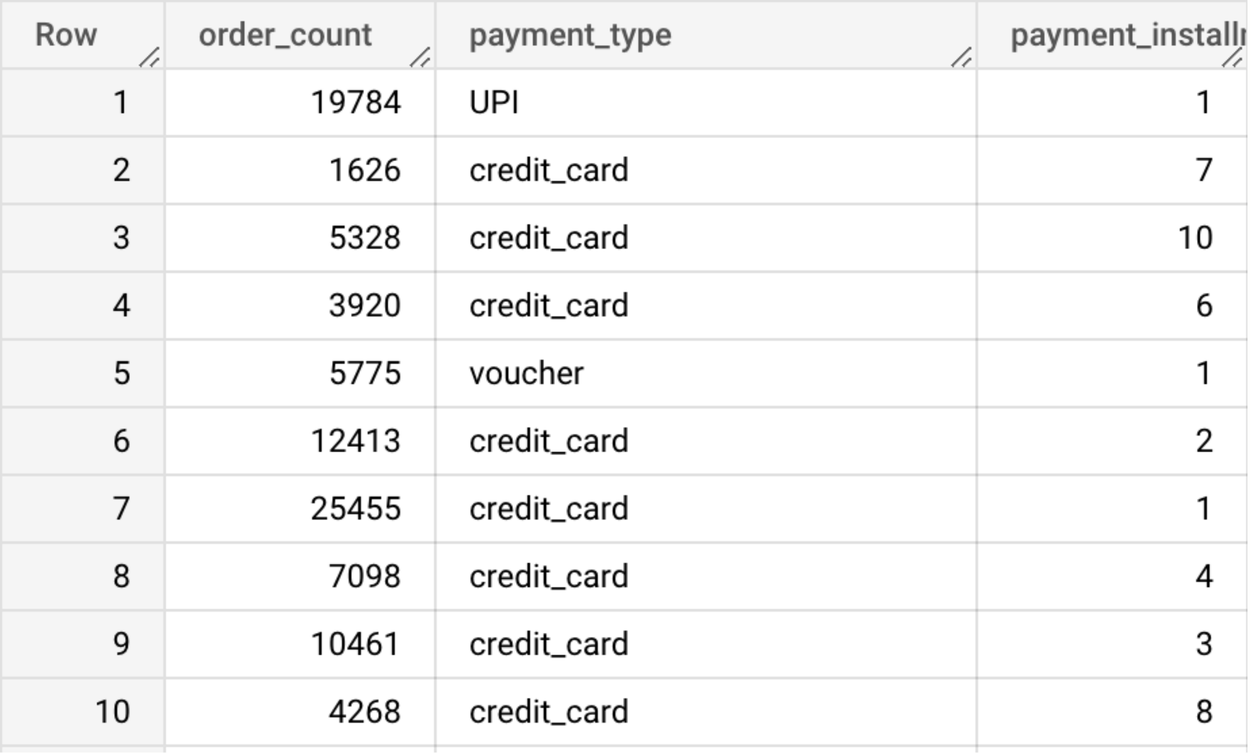
1. Month over Month count of orders for different payment types
2. Count of orders based on the no. of payment installments

```
select
count(o.order_id) as order_count,
p.payment_type,
extract (month from order_purchase_timestamp) as month,
extract (year from order_purchase_timestamp) as year
from `Target.orders` as o
join `Target.payments` as p on p.order_id = o.order_id
group by p.payment_type, month, year
order by year,month
```



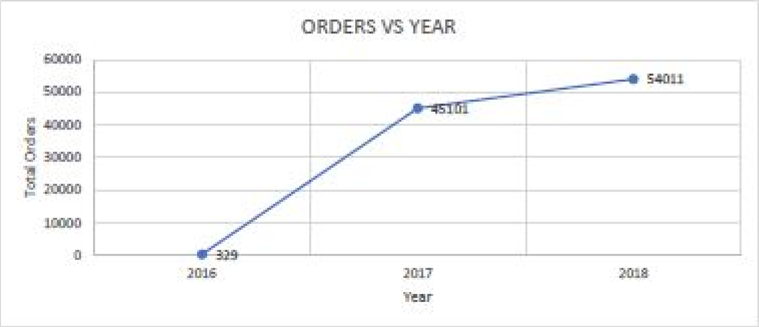
Row	order_count	payment_type	month	year
1	3	credit_card	9	2016
2	63	UPI	10	2016
3	254	credit_card	10	2016
4	2	debit_card	10	2016
5	23	voucher	10	2016
6	1	credit_card	12	2016
7	197	UPI	1	2017
8	583	credit_card	1	2017
9	9	debit_card	1	2017
10	61	voucher	1	2017

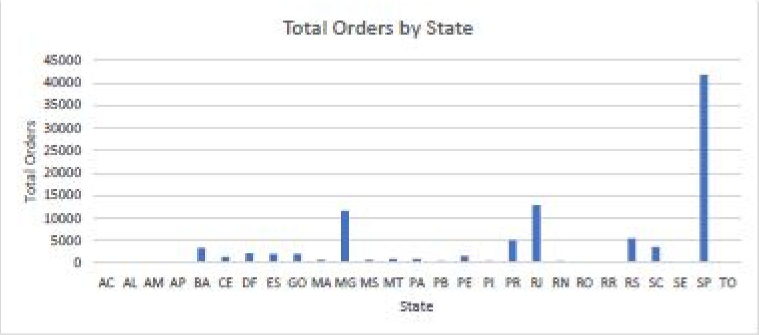
```
select
count(o.order_id) as order_count, p.payment_type,
p.payment_installments
from `Target.orders` as o
join `Target.payments` as p on o.order_id = p.order_id
group by p.payment_installments, p.payment_type;
```

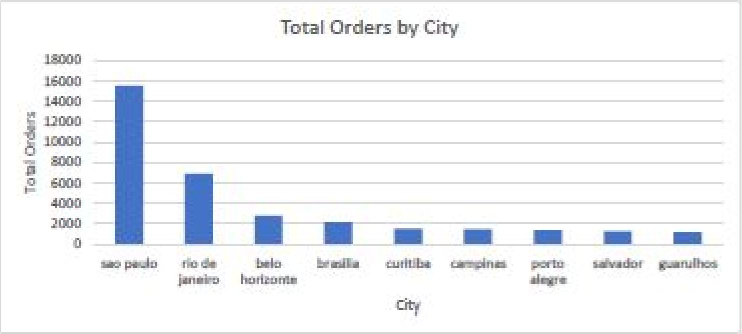


Row	order_count	payment_type	payment_install
1	19784	UPI	1
2	1626	credit_card	7
3	5328	credit_card	10
4	3920	credit_card	6
5	5775	voucher	1
6	12413	credit_card	2
7	25455	credit_card	1
8	7098	credit_card	4
9	10461	credit_card	3
10	4268	credit_card	8

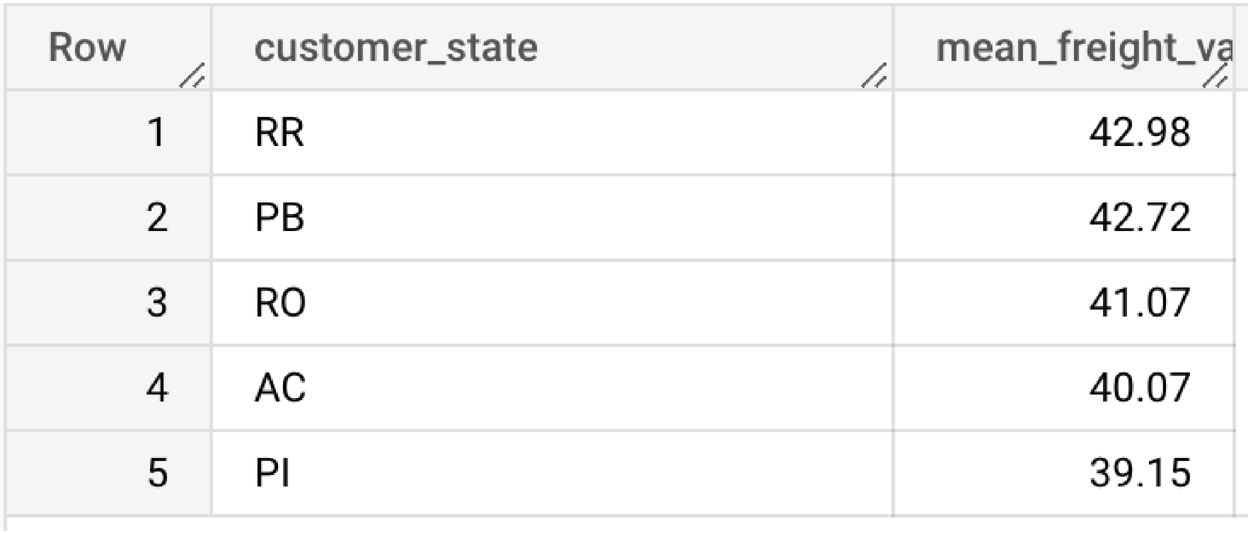
INSIGHTS AND RECOMMENDATION





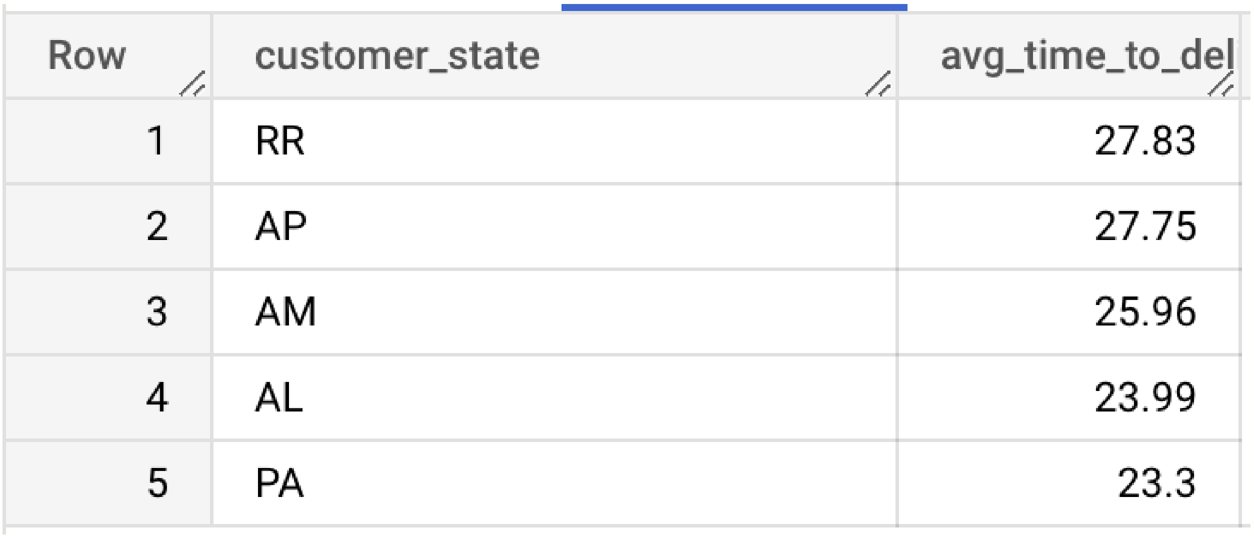


Gain valuable insights into order patterns across different states, cities and compare year-over-year trends for a comprehensive analysis.



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

Above table represents Highest average freight value by states



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

Above table represents Highest average time to Delivery

Streamlining processes within the transport department is an effective way to reduce costs and improve operational efficiency. This can include implementing modern technologies such as transportation management systems (TMS) to automate and optimize the planning, execution, and settlement of shipments. By reducing the time and resources spent on manual tasks, the company can increase the speed and accuracy of deliveries, which can improve customer satisfaction and ultimately drive stronger sales.

In addition, the company can also look at ways to optimize its supply chain network to reduce transportation costs, such as consolidating shipments and choosing the most cost-effective modes of transportation. By focusing on the states with the highest average freight value and time to delivery, the company can prioritize its efforts and allocate resources where they will have the biggest impact.