TARGET SQL BUSINESS CASE

1. Exploratory Analysis

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

Query :

```
select column_name,data_type
from `target.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

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

Query:

```
select min(order_purchase_timestamp) as min_order_time,
max(order_purchase_timestamp) as max_order_time
from `target.orders`;
```

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

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

Query:

```
select customer_id, count(customer_city) as Cities_count,
count(customer_state) as State_count
from `target.customers` group by customer_id;
```

Row	customer_id	Cities_count	State_count
1	0735e7e4298a2ebbb4664934...	1	1
2	903b3d86e3990db01619a4eb...	1	1
3	38c97666e962d4fea7fd6a83e...	1	1
4	77c2f46cf580f4874c9a5751c2...	1	1
5	4d3ef4cfffb8ad4767c199c36a...	1	1
6	3000841b86e1fbe9493b52324...	1	1
7	3c325415ccc7e622c66dec4bc...	1	1
8	04f3a7b250e3be964f01bf22bc...	1	1
9	894202b8ef01f4719a4691e79...	1	1
10	9d715b9fb75a9d081c14126c0...	1	1

2. In- Depth Exploration

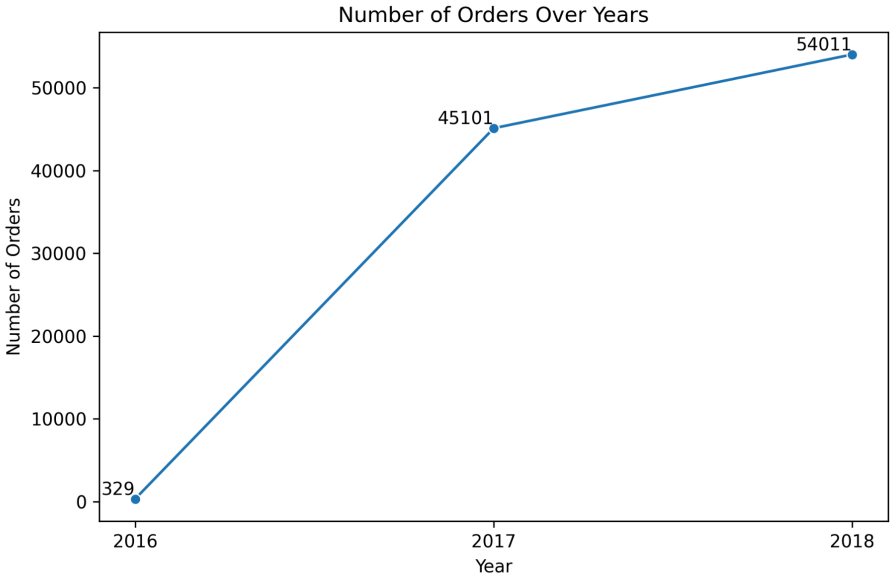
1. Is there a growing trend in the no. of orders placed over the past years?

Query:

```
select extract(year from order_purchase_timestamp) as order_year,
count(*) as count_orders
from `target.orders`
group by order_year
order by order_year;
```

Row	order_year	count_orders
1	2016	329
2	2017	45101
3	2018	54011

Yes there is a significant increase in order placed by customers over the past years.



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

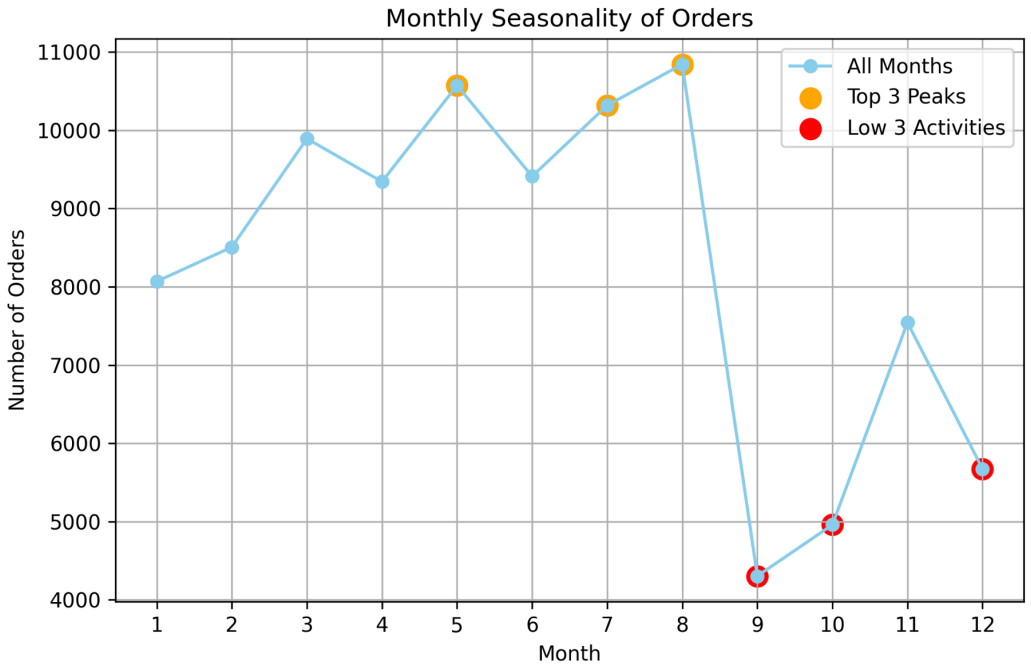
Query:

```
select extract(month from order_purchase_timestamp) as order_month,
count(*) as count_orders
from `target.orders`
group by order_month order by order_month;
```

Row	order_month	count_orders
1	1	8069
2	2	8508
3	3	9893
4	4	9343
5	5	10573
6	6	9412
7	7	10318
8	8	10843
9	9	4305
10	10	4959
11	11	7544
12	12	5674

Monthly Order Seasonality Summary

- **Peaks**: May, July, and August stand out as peak months with higher order counts.
- **Lows**: January and September display lower activity, indicating off-peak periods.



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

Query:

```
select
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 time_of_day,
count(*) as no_of_orders
from `target.orders`
group by time_of_day;
```

Row	time_of_day	no_of_orders
1	Mornings	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

Time of Day for Order Placement Summary

- **Preferential Timing:** Customers predominantly place orders during the Afternoon (13:00 - 18:00 hrs), indicating a peak in order activity during these hours.
- **Other Active Periods:** Following the Afternoon, significant order placement occurs during the Morning (07:00 - 12:00 hrs) and Night (19:00 - 23:00 hrs) timeframes.
- **Lower Activity:** The Dawn (00:00 - 06:00 hrs) period records comparatively lower order counts, suggesting reduced customer activity during these early hours.

3. Evolution of E-Commerce orders in Brazil Region

1. Get the month-on-month number of orders placed in each state

Query:

```
with customer_orders as
(select * from `target.orders` as o
inner join `target.customers` as c
on o.customer_id = c.customer_id)
```

```sql
select extract(month from order_purchase_timestamp) as month,
customer_state,
count(*) as no_of_orders
from customer_orders
group by month, customer_state
order by month, customer_state;
```

Row	month	customer_state	no_of_orders
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

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

Query:

```sql
select customer_state,count(distinct customer_id) as No_of_target_customers
from `target.customers`
group by customer_state
order by customer_state;
```

Row	customer_state	No_of_target_customers
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

Distribution of Customers across various states

- **States with Lots and Fewer Customers:**

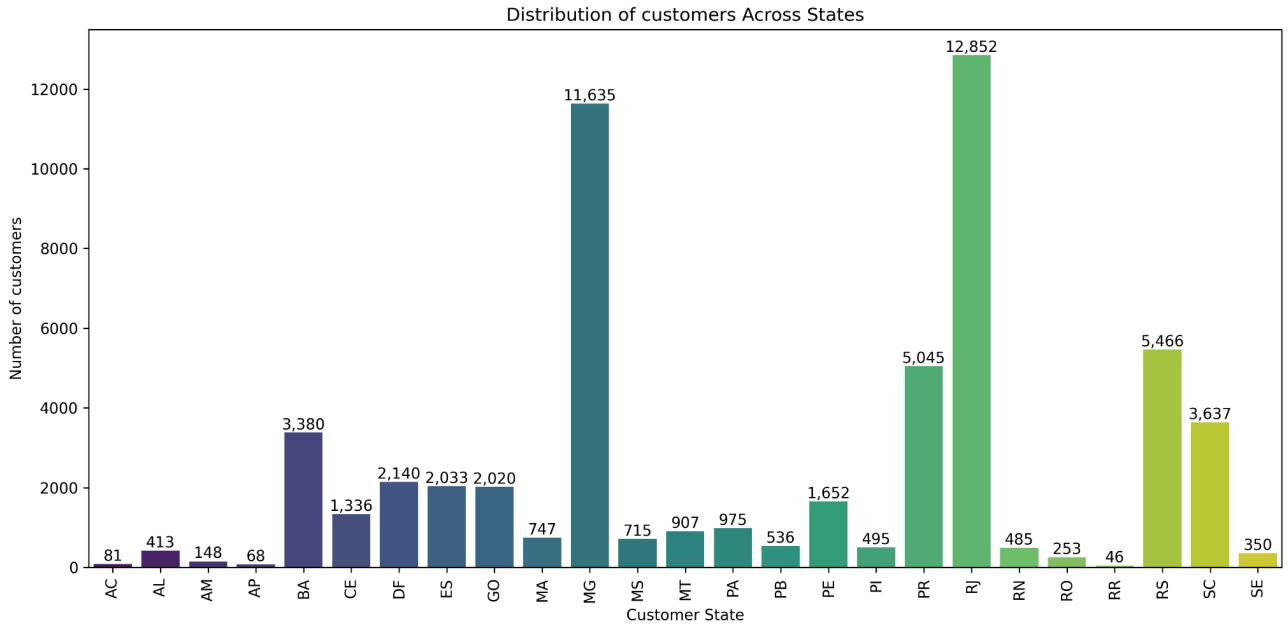
  Certain states, like MG (Minas Gerais) and RJ (Rio de Janeiro) , have a lot of these target customers, while places like RR (Roraima) and AP (Amapa) have very few.

- **What it Means for Business:**

  Places with more customers might be better for doing business, while places with fewer customers might be more challenging.

- **Why It Matters:**

  Because customer numbers vary a lot, businesses need to think differently about how they sell and market in each place. They might need special plans for each area.



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

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.

   Query:

```
with percentage_increase as
(select extract(year from o.order_purchase_timestamp) as year,
sum(p.payment_value) as cost_of_orders
from `target.orders` as o
```

```
join `target.payments` as p
on o.order_id = p.order_id
where order_purchase_timestamp between '2017-01-01' and '2018-08-31'
group by year)

select round((cost_of_orders - lag(cost_of_orders) over(order by cost_of_orders)) /
lag(cost_of_orders) over(order by cost_of_orders), 2) * 100 as percentage_increase
from percentage_increase;
```

Row	percentage_increase
1	*null*
2	20.0

The recorded **20.0%** increase in order costs from 2017 to 2018 (Jan to Aug) reflects a significant rise in expenditure. This surge signals a positive trend, suggesting heightened sales or larger order values during this period. The percentage rise indicates a notable growth potential for the business during these months.

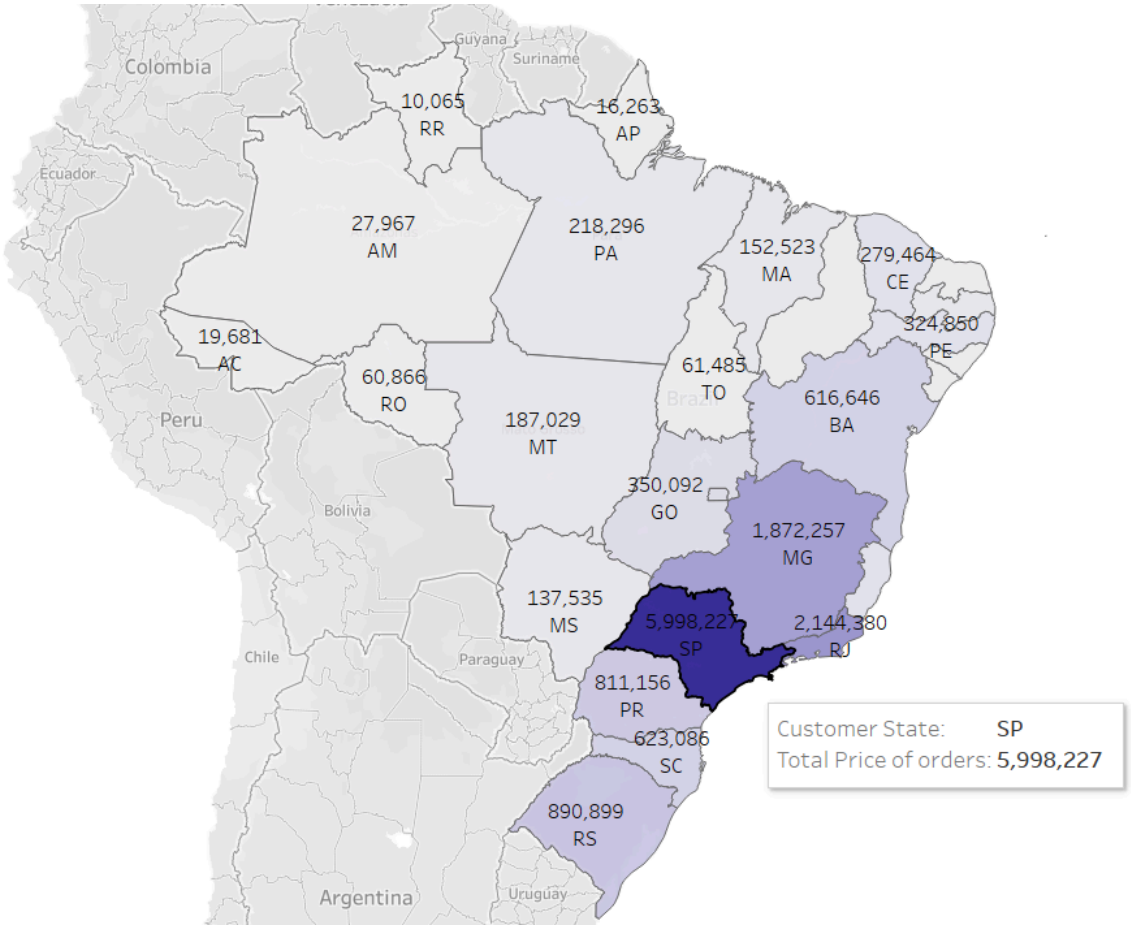
2. Calculate the Total & Average value of Order price of each state.

Query:

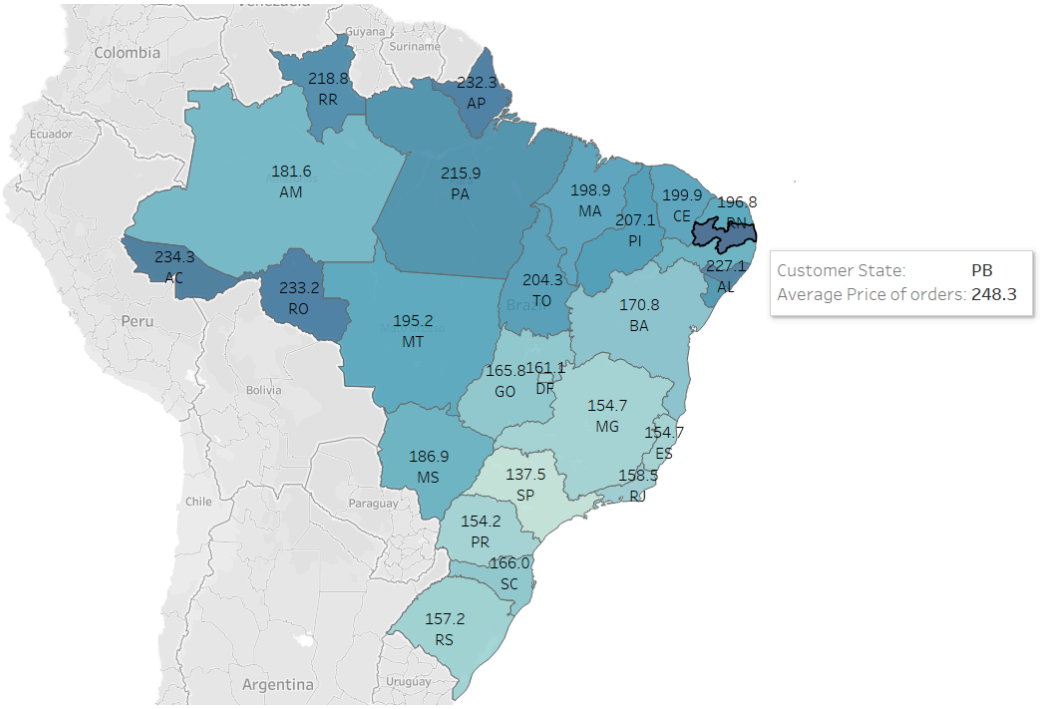
```
select c.customer_state,
sum(p.payment_value) as Total_Price_of_orders,
avg(p.payment_value) as Average_Price_of_orders
from `target.orders` as o
join
`target.payments` as p
on o.order_id = p.order_id
join `target.customers` as c
on o.customer_id = c.customer_id
group by c.customer_state;
```

Row	customer_state	Total_Price_of_order	Average_Price_of_or
1	RJ	2144379.689999...	158.5258882235...
2	RS	890898.5399999...	157.1804057868...
3	SP	5998226.959999...	137.5046297739...
4	DF	355141.0800000...	161.1347912885...
5	PR	811156.3799999...	154.1536259977...
6	MT	187029.2900000...	195.2289039665...
7	MA	152523.0200000...	198.8566101694...
8	AL	96962.05999999...	227.0774238875...
9	MG	1872257.260000...	154.7064336473...
10	PE	324850.4400000...	187.9921527777...

Total order value per state



Average order value per state



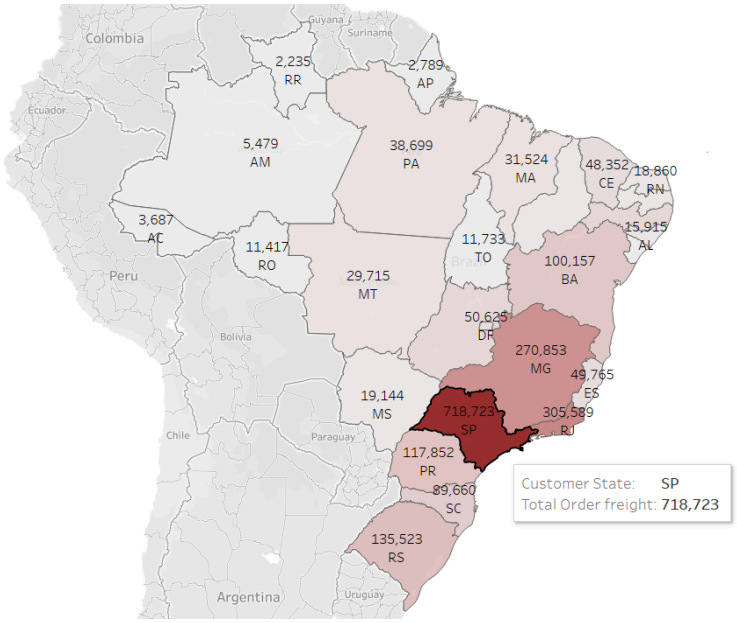
3. Calculate the Total & Average value of order freight for each state.

Query:

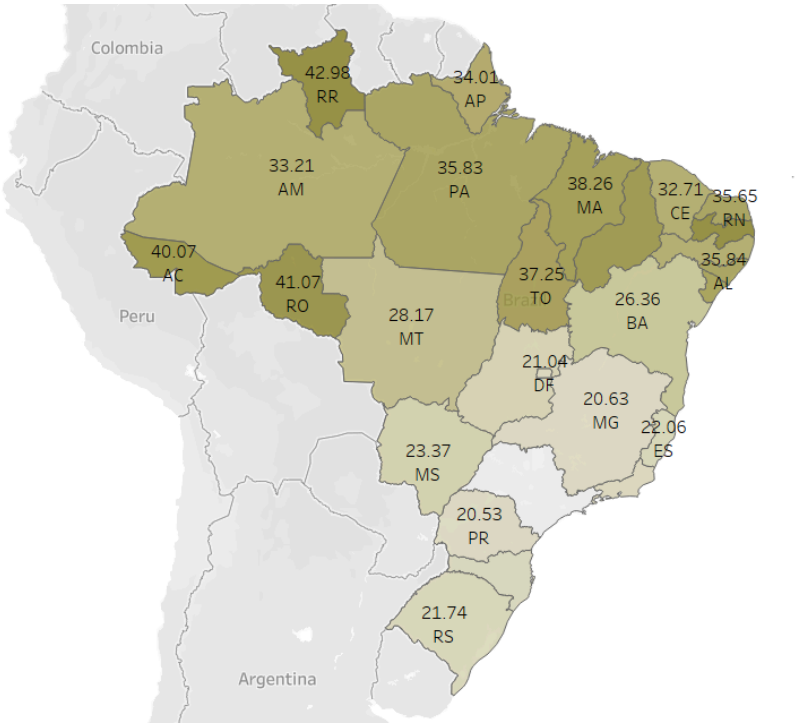
```
select c.customer_state,
sum(oi.freight_value) as Total_Order_freight,
avg(oi.freight_value) as Average_Order_freight
from `target.order_items` as oi
join `target.orders` as o
on oi.order_id = o.order_id
join `target.customers` as c
on o.customer_id = c.customer_id
group by c.customer_state;
```

Row	customer_state	Total_Order_freight	Average_Order_freight
1	SP	718723.06999999378	15.147275390419132
2	RJ	305589.31000000431	20.960923931682483
3	PR	117851.68000000058	20.531651567944269
4	SC	89660.260000000053	21.470368773946323
5	DF	50625.499999999418	21.041354945968422
6	MG	270853.4600000073	20.630166806306651
7	PA	38699.300000000047	35.832685185185213
8	BA	100156.67999999922	26.36395893656228
9	GO	53114.979999999705	22.766815259322772
10	RS	135522.74000000197	21.735804330392952
11	TO	11732.679999999998	37.246603174603166
12	AM	5478.8900000000012	33.205393939393922
13	MA	31523.77000000004	38.257002427184474
14	PE	59449.659999999873	32.917862679955654
15	ES	49764.599999999722	22.058776595744732
16	AL	15914.589999999989	35.843671171171167
17	MT	29715.430000000109	28.166284360189572
18	RN	18860.099999999973	35.652362948960366

Total Freight Value per each state



Average Freight Value per each state



5. 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.

   Query:

```
select date_diff(order_delivered_customer_date, order_purchase_timestamp,day) as
Delivered_in,
order_estimated_delivery_date - order_delivered_customer_date as Difference
from `target.orders`;
```

Row	Delivered_in	Difference
1	30	0-0 0 -310:3:51
2	30	0-0 0 681:6:10
3	35	0-0 0 397:1:26
4	30	0-0 0 33:10:5
5	32	0-0 0 13:7:45
6	29	0-0 0 38:52:13
7	43	0-0 0 -110:11:31
8	40	0-0 0 -112:18:42
9	37	0-0 0 -37:44:52
10	33	0-0 0 -134:19:48

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

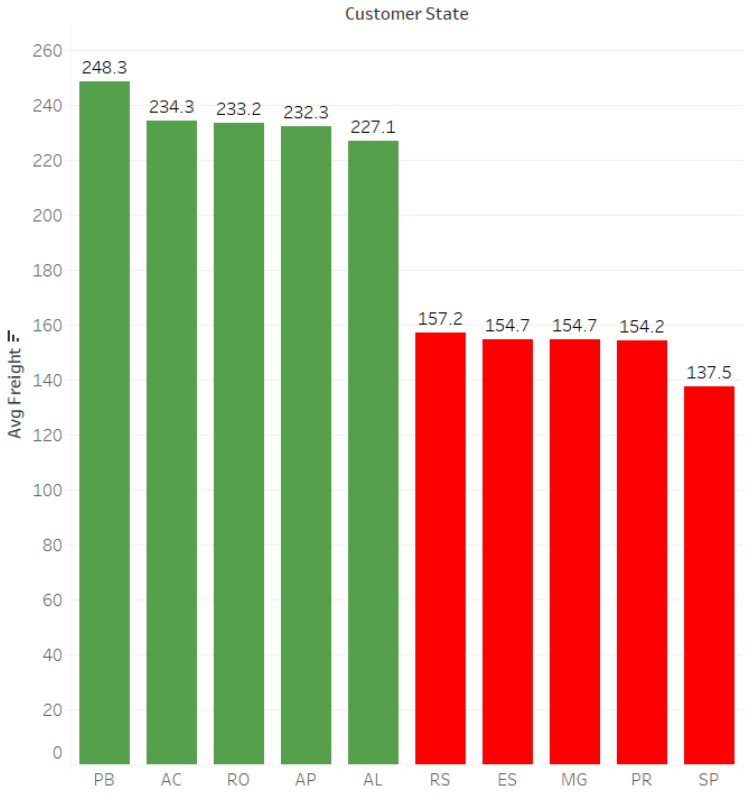
Query:

```
with states_avg_payment as (
  select c.customer_state, avg(p.payment_value) as avg_freight
  from `target.orders` o
  join `target.payments` p on o.order_id = p.order_id
  join `target.customers` c on o.customer_id = c.customer_id
  group by c.customer_state
)
select * from
(
  select customer_state, avg_freight
  from states_avg_payment
  order by avg_freight desc
  limit 5
) highest
union all
select * from
(
  select customer_state, avg_freight
  from states_avg_payment
  order by avg_freight asc
  limit 5
) lowest;
```

Row	customer_state	avg_freight
1	PB	248.3258245614...
2	AC	234.2930952380...
3	RO	233.2038314176...
4	AP	232.3257142857...
5	AL	227.0774238875...
6	SP	137.5046297739...
7	PR	154.1536259977...
8	MG	154.7064336473...
9	ES	154.7069530137...
10	RS	157.1804057868...

The top 5 states with the highest average freight values are Paraíba (PB), Acre (AC), Rondônia (RO), Amapá (AP), and Alagoas (AL). On the other hand, São Paulo (SP), Paraná (PR), Minas Gerais (MG), Espírito Santo (ES), and Rio Grande do Sul (RS) have the lowest average freight values.

Highest & Lowest Frieght Value across each state



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

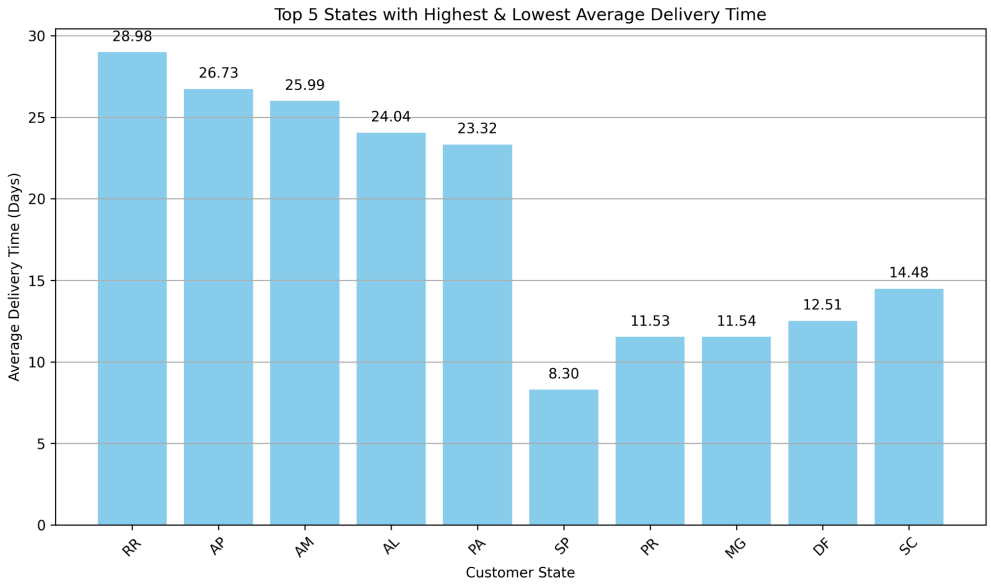
Query:

```
with average_delivery as
(
  select c.customer_state,
  avg(date_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,day)) as
average_delivery_time
  from `target.orders` as o
  join
  `target.customers` as c
  on o.customer_id = c.customer_id
  group by c.customer_state
)

select * from
(select customer_state, average_delivery_time from average_delivery order by
average_delivery_time desc limit 5) as high
union all
select * from
(select customer_state, average_delivery_time from average_delivery order by
average_delivery_time limit 5) as low;
```

Row	customer_state	average_delivery_time
1	RR	28.975609756097562
2	AP	26.731343283582085
3	AM	25.986206896551728
4	AL	24.040302267002513
5	PA	23.316067653276981
6	SP	8.2980614890725874
7	PR	11.526711354864908
8	MG	11.543813298106569
9	DF	12.509134615384616
10	SC	14.479560191711331

- **Top States with Longest Delivery Times**: Roraima (RR), Amapá (AP), Amazonas (AM), Alagoas (AL), and Pará (PA) exhibit the highest average delivery times, exceeding 23 days on average.
- **States with Shortest Delivery Times:** São Paulo (SP), Paraná (PR), Minas Gerais (MG), Distrito Federal (DF), and Santa Catarina (SC) demonstrate notably lower average delivery times, with figures below 15 days on average.
- Enhance delivery in regions with longer times like Roraima (RR), Amapá (AP), Amazonas (AM), Alagoas (AL), and Pará (PA). Set up regional centers, employ tech solutions, and gather feedback for better service.



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

Query:

```
with state_delivery as (
  select c.customer_state,
    avg(date_diff(o.order_delivered_customer_date, o.order_purchase_timestamp,day))
as avg_actual_delivery,
    avg(date_diff(o.order_estimated_delivery_date ,
o.order_delivered_customer_date,day)) as avg_estimated_delivery
  from target.orders o
  join target.customers c on o.customer_id = c.customer_id
  group by c.customer_state
)
select customer_state,
  avg_actual_delivery,
  avg_estimated_delivery,
  (avg_estimated_delivery - avg_actual_delivery) as delivery_diff
from state_delivery
order by delivery_diff desc
limit 5;
```

Row	customer_state	avg_actual_delivery	avg_estimated_delivery	delivery_diff
1	SP	8.2980614890725874	10.135325348808554	1.837263859735...
2	PR	11.526711354864908	12.364208815762742	0.837497460897...
3	MG	11.543813298106569	12.296961690885075	0.753148392778...
4	RO	18.913580246913586	19.13168724279836	0.218106995884...
5	AC	20.637500000000003	19.762500000000006	-0.87499999999...

- **Efficient Delivery States:** São Paulo (SP), Paraná (PR), Minas Gerais (MG), Rondônia (RO), and Acre (AC) are the top five states where orders are delivered notably faster than the estimated dates.
- **SP Leads with Efficiency:** São Paulo demonstrates the most efficient delivery, with an average difference of approximately 1.84 days between actual and estimated delivery dates.
- **Minor Variances in Other States:** States like Paraná, Minas Gerais, and Rondônia also exhibit relatively faster delivery, with differences ranging from approximately 0.75 to 0.22 days ahead of the estimated delivery times.

6. Analysis based on Payments

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

Query:

```
select format_date('%Y-%m',order_purchase_timestamp) as order_month,
       payment_type,
       count(*) as num_orders
from `target.orders` o
join `target.payments` p on o.order_id = p.order_id
group by order_month, payment_type
order by order_month;
```

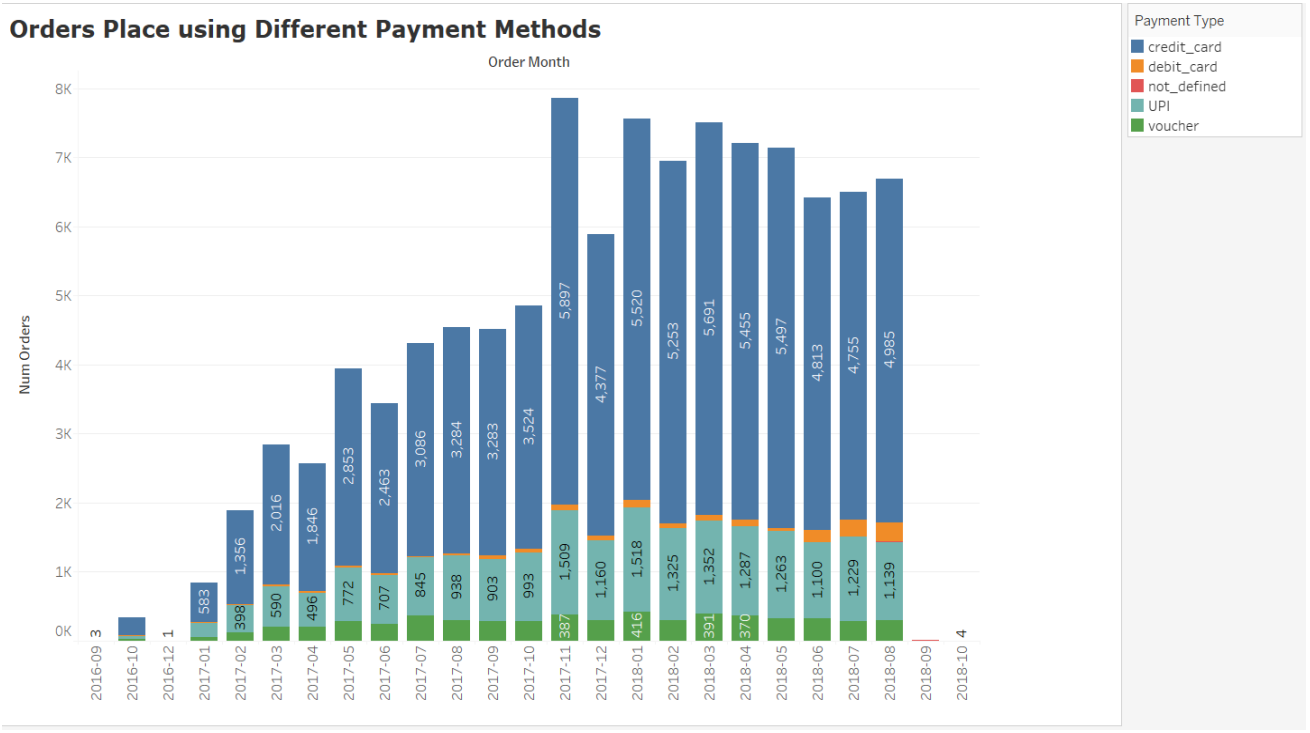
Row	order_month	payment_type	num_orders
1	2016-09	credit_card	3
2	2016-10	credit_card	254
3	2016-10	UPI	63
4	2016-10	voucher	23
5	2016-10	debit_card	2
6	2016-12	credit_card	1
7	2017-01	credit_card	583
8	2017-01	UPI	197
9	2017-01	voucher	61
10	2017-01	debit_card	9
11	2017-02	credit_card	1356
12	2017-02	UPI	398
13	2017-02	voucher	119

- **Credit Card Dominance:** Credit card payments consistently dominate across all observed periods, showing a rising trend from 2016 to 2018.
- **Stable Usage of Other Methods:** UPI, vouchers, and debit cards maintain relatively stable usage patterns over time.
- **Decline in Voucher Usage:** Voucher usage starts high in 2016 but experiences a declining trend, notably from 2017 onwards.

What should be done ?

- **Promote Credit Card Benefits:** Capitalize on the prevalent credit card usage by introducing tailored promotions or loyalty programs to sustain and attract more credit card users.

- **<u>Revitalize Voucher Usage:</u>** Investigate reasons behind the decline in voucher usage and implement targeted campaigns or incentives to reignite interest among customers.
- **<u>Diversify Payment Promotions:</u>** Encourage usage of UPI and debit cards by offering incentives or partnerships to maintain a balanced payment ecosystem.



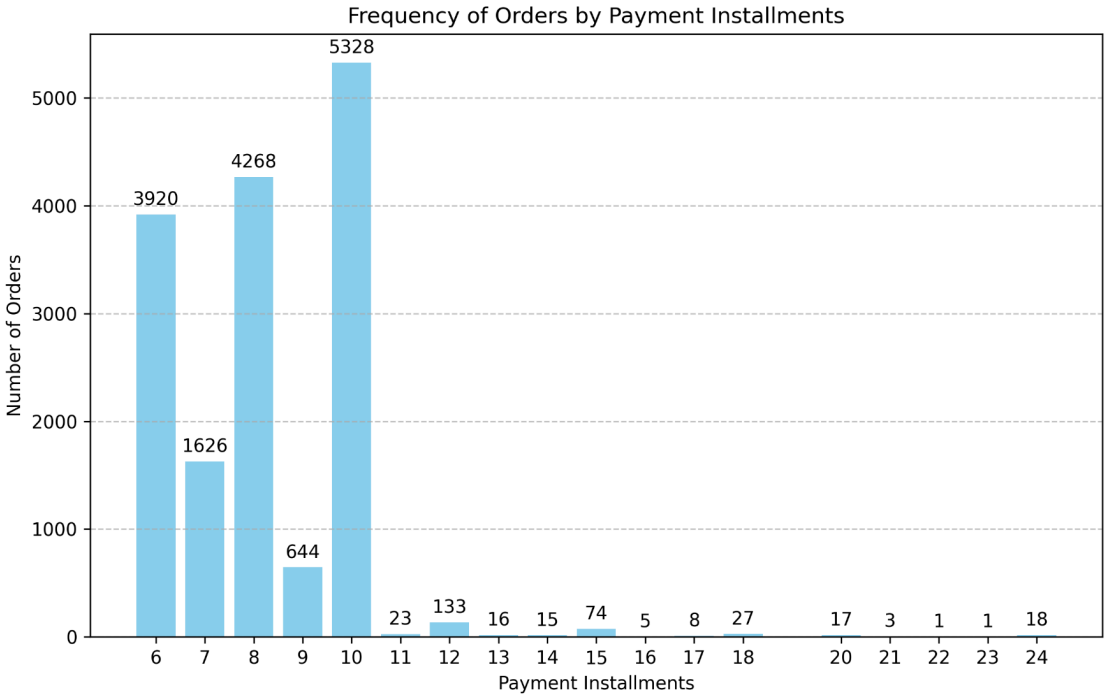
2. Find the no. of orders placed on the basis of the payment instalments that have been paid.

Query:

```
select payment_installments,
       count(*) as num_orders
from `target.orders` o
join `target.payments` p on o.order_id = p.order_id
where payment_installments is not null
group by payment_installments
order by payment_installments;
```

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

- **<u>Installments vs. Order Frequency:</u>** As the number of payment installments increases, the frequency of orders diminishes significantly.
- **<u>Common Payment Installments:</u>** Orders with 1 to 10 payment installments are substantially more prevalent, with a significant decline in orders beyond 10 installments.



7. Actionable insights and Recommendation

1. Top 10 Customer by overall spending

Query:

```
select c.customer_id,
       sum(p.payment_value) as total_spend
from `target.orders` o
join `target.payments` p on o.order_id = p.order_id
join `target.customers` c on o.customer_id = c.customer_id
group by c.customer_id
order by total_spend desc
limit 10;
```

Row	customer_id	total_spend
1	1617b1357756262bfa56ab541...	13664.08
2	ec5b2ba62e574342386871631...	7274.88
3	c6e2731c5b391845f6800c974...	6929.31
4	f48d464a0baaea338cb25f816...	6922.21
5	3fd6777bbce08a352fddd04e4...	6726.66
6	05455dfa7cd02f13d132aa7a6...	6081.54
7	df55c14d1476a9a3467f13126...	4950.34
8	e0a2412720e9ea4f26c1ac985...	4809.44
9	24bbf5fd2f2e1b359ee7de94de...	4764.34
10	3d979689f636322c62418b634...	4681.78

- Customer Spend Diversity: Observing a diverse range of spending from $13,664.08 to $4,681.78 among top customers suggests tailoring loyalty programs or exclusive offers based on spending levels to incentivize higher spend.
- Targeted Engagement: Implementing personalized offers based on past purchases or exclusive discounts on frequently bought items can enhance engagement and foster customer loyalty, encouraging repeat purchases.
- Strategic Insights: Utilize data insights from high spenders to refine product offerings, optimize marketing strategies, and enhance customer experiences to retain these valuable customers and potentially encourage them to become brand advocates.