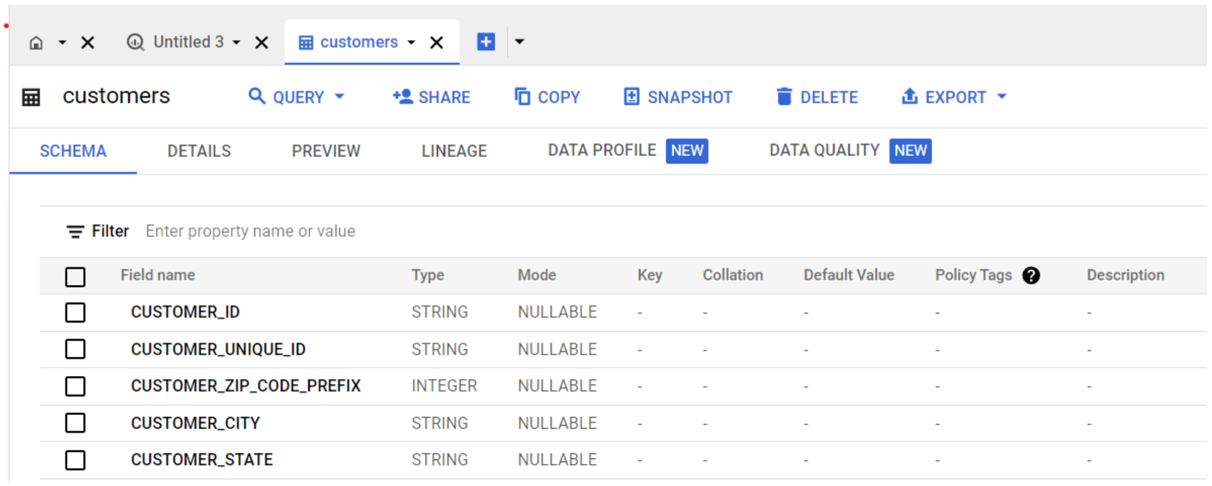
1. Exploratory data analysis:

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

Ans:



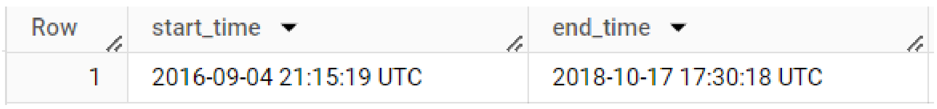
Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
CUSTOMER_ID	STRING	NULLABLE	-	-	-	-	-
CUSTOMER_UNIQUE_ID	STRING	NULLABLE	-	-	-	-	-
CUSTOMER_ZIP_CODE_PREFIX	INTEGER	NULLABLE	-	-	-	-	-
CUSTOMER_CITY	STRING	NULLABLE	-	-	-	-	-
CUSTOMER_STATE	STRING	NULLABLE	-	-	-	-	-

Insights: In customer table customer_id, customer_unique_id, customer_city, and customer_state are STRING data type whereas customer_zip_code_prefix is an INTEGER data type.

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

Ans:

```
select
min(order_purchase_timestamp) as `start_time`,
max(order_purchase_timestamp) as `end_time`
from `Business.orders`;
```



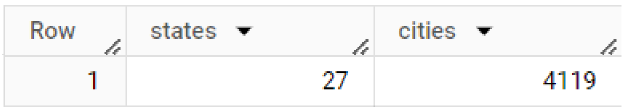
Row	start_time	end_time
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insights: as per UTC time zone, the first order placed on 2016-09-04 at 21hr:15min:19sec and the last order placed on 2018-10-17 at 17hr:30min:18sec

3. How many Cities & States of customers ordered during the given period.

Ans:

```
select count (distinct customer_state) as states,
count (distinct customer_city) as cities
from ` Business.customers` where customer_id in
(select distinct customer_id from ` Business.orders`)
```



Row	states	cities
1	27	4119

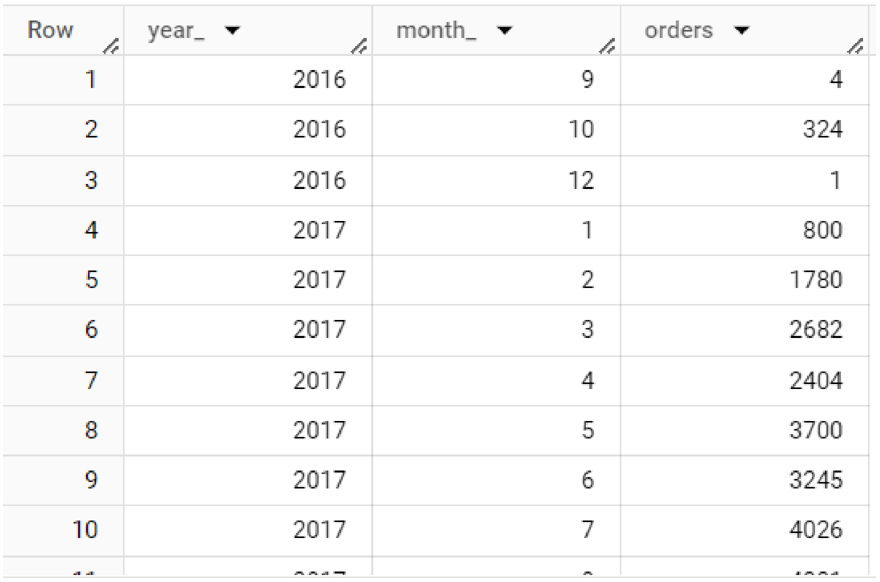
Insights: there were total 4119 number of cities and 27 states of customers who ordered in the given period.

2.In-depth Exploration:

1. What is the growing trend in the no. of orders placed over the past years?

Ans:

```
select extract (year from order_purchase_timestamp) as year_,
extract (month from order_purchase_timestamp) as month_,
count (order_id) as orders
from ` Business.orders`
group by 1,2
order by 1,2
```



Row	year_	month_	orders
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
9	2017	6	3245
10	2017	7	4026

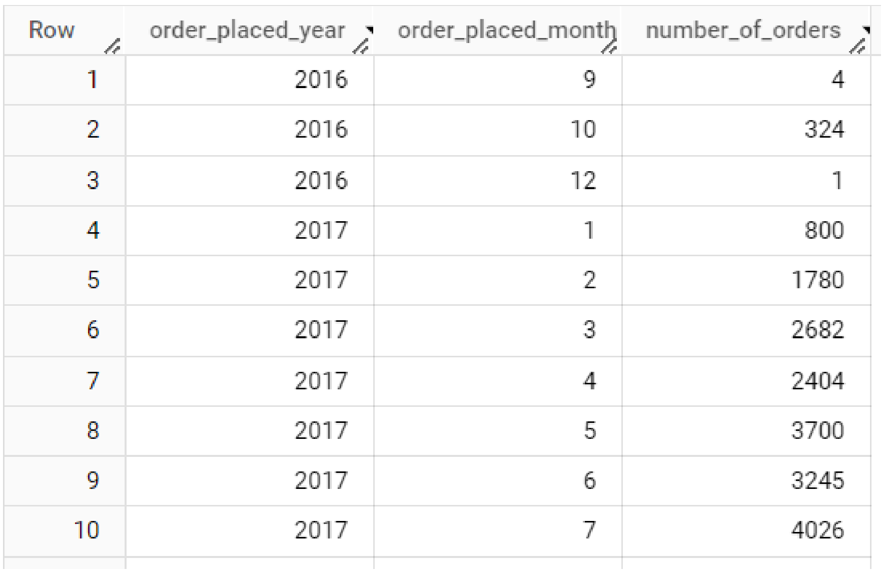
Insights: Yes. As seen in the query results, the number of orders were increased each year when compared to previous year.

Recommendations: the following are some important recommendations to consider to sustain business to meet the growing demands; infrastructure scaling, processes optimization, inventory management, maintaining good relationships with suppliers, marketing strategies, customer support, technological advancements, employee training, and quality control.

2. **Is there any kind of monthly seasonality in terms of the no. of orders being placed?**

Ans:

```
select
  extract (year from order_purchase_timestamp) as `order_placed_year`,
  extract (month from order_purchase_timestamp) as `order_placed_month`,
  count(order_id) as `number_of_orders`
from ` Business.orders`
group by 1, 2
order by 1, 2;
```



Row	order_placed_year	order_placed_month	number_of_orders
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
9	2017	6	3245
10	2017	7	4026

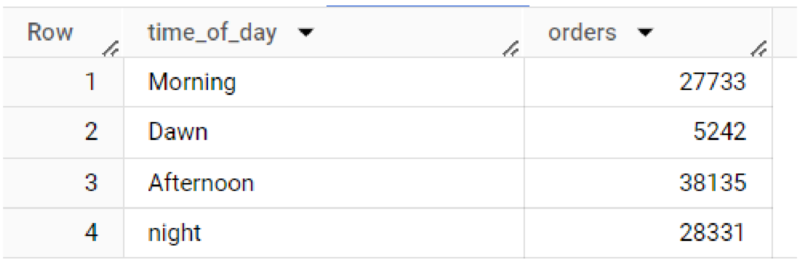
Insights: It was clear that in some months there were more orders whereas in other months less orders. Also, the monthly seasonality is not same for all the years.

Recommendations: more yearly data month wise and additional information (for example, socioeconomic and cultural status) is required for actual understanding of monthly seasonality. Identifying and understanding the demand in specific months and providing offers, discounts or new sales strategies in those periods is good strategy. Also, it is essential to keep the products to meet high demands during those periods. In low demanding months, again understanding is essential but also one can introduce new items or refresh existing ones in order to grasp the customer attention.

3. **What time the customers mostly place their orders in a day? (Dawn, Morning, Afternoon or Night)**

Ans:

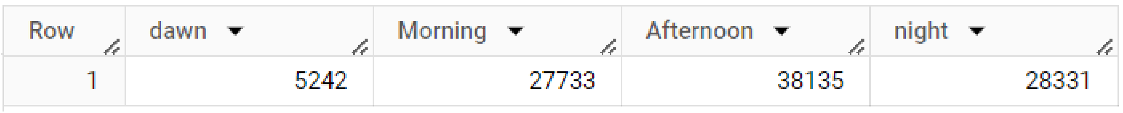
```
select time_of_day, sum(orders) as orders from
(
select *, case when hours between 0 and 6 then 'Dawn'
when hours between 7 and 12 then 'Morning'
when hours between 13 and 18 then 'Afternoon'
else 'night' end as time_of_day
from
(select extract (hour from order_purchase_timestamp) as hours,
count (order_id) as orders
from ` Business.orders`
group by 1
order by 1) base
)
group by 1
```



Row	time_of_day	orders
1	Morning	27733
2	Dawn	5242
3	Afternoon	38135
4	night	28331

Another way:

```
select sum(case when hours between 0 and 6 then orders end) as dawn,
sum(case when hours between 7 and 12 then orders end) as Morning,
sum(case when hours between 13 and 18 then orders end) as Afternoon,
sum(case when hours between 19 and 23 then orders end) as night
from (select extract (hour from order_purchase_timestamp) as hours,
count (order_id) as orders from ` Business.orders` group by 1 order by 1) base
```



Row	dawn	Morning	Afternoon	night
1	5242	27733	38135	28331

Insights: During afternoon the orders were highest followed by night and mornings. Compared to others the orders were less during dawn.

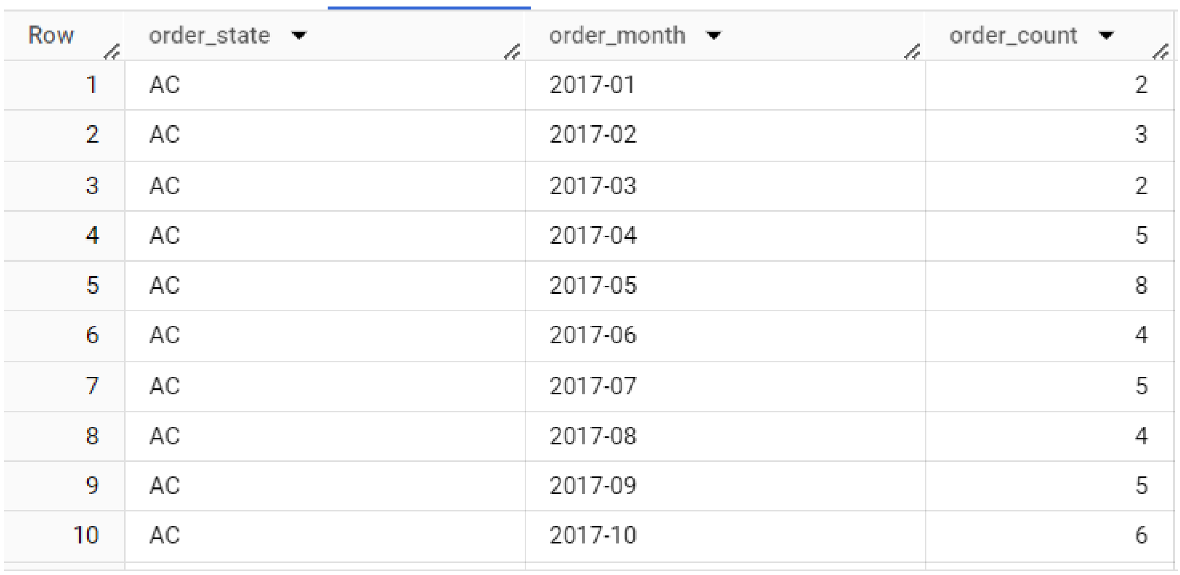
Recommendations: provide fast and efficient services during afternoon, morning, and night times. Utilize social media during peak times to ensure our product as top brand in customer mind set. Take feedback from customers on which time they are comfortable and accordingly provide the flexibility. It is also important to know that what kind of products they are looking for during those specific times and based on that we can create product bundles which drives to order more items.

3. Exploring the E-commerce orders in particular region:

1. Month-on-month orders placed in each state.

Ans:

```
select
  c.customer_state as `order_state`,
  format_datetime('%Y-%m', o.order_purchase_timestamp) as `order_month`,
  count(*) as `order_count`
from ` Business.orders` o inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1,2
order by 1,2;
```



Row	order_state	order_month	order_count
1	AC	2017-01	2
2	AC	2017-02	3
3	AC	2017-03	2
4	AC	2017-04	5
5	AC	2017-05	8
6	AC	2017-06	4
7	AC	2017-07	5
8	AC	2017-08	4
9	AC	2017-09	5
10	AC	2017-10	6

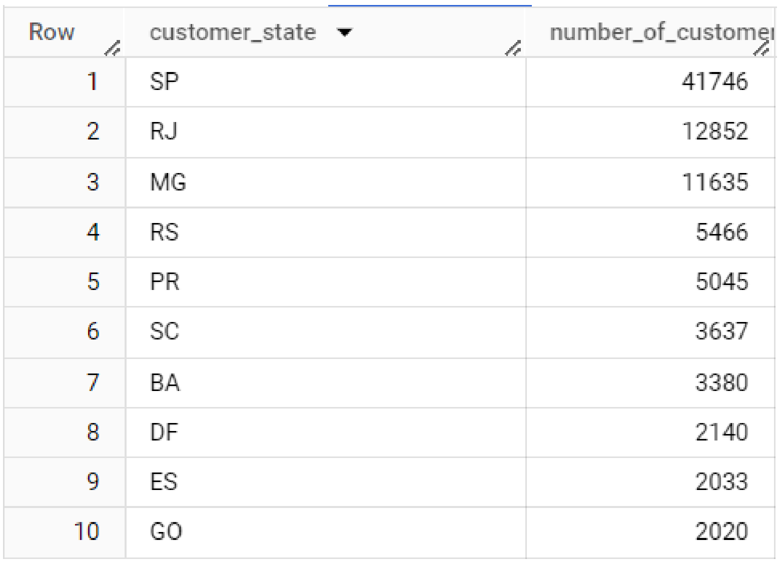
Insights: Here, the data was organized to extract number of orders in each state separately for every month.

Recommendations: identifying high and low performing states; localized promotions and offers; supply chain optimization; investigate on cultural differences in these sates and their impact on product purchase; local partnerships.

2. **Understanding the customer distribution across all the states**

Ans:

```
select
  customer_state,
  count(customer_unique_id) as `number_of_customers`
from ` Business.customers`
group by 1
order by 2 desc;
```



Row	customer_state	number_of_custome
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

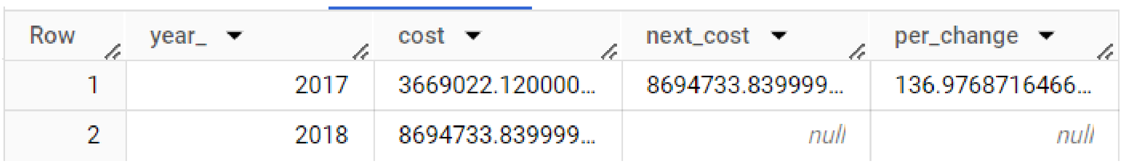
Insights: highest number of customers were present in the state SP followed by RJ and MG. The customers were significantly large in number than any other state.

Recommendations: customer centric approach is important; customer support; personalize marketing strategy for specific states; local product bundles; local partnerships; new marketing strategies; community engagement; maintain branding in those states with high customers; take customer feedbacks.

4. Analysing the economy based on order prices, freight and others.

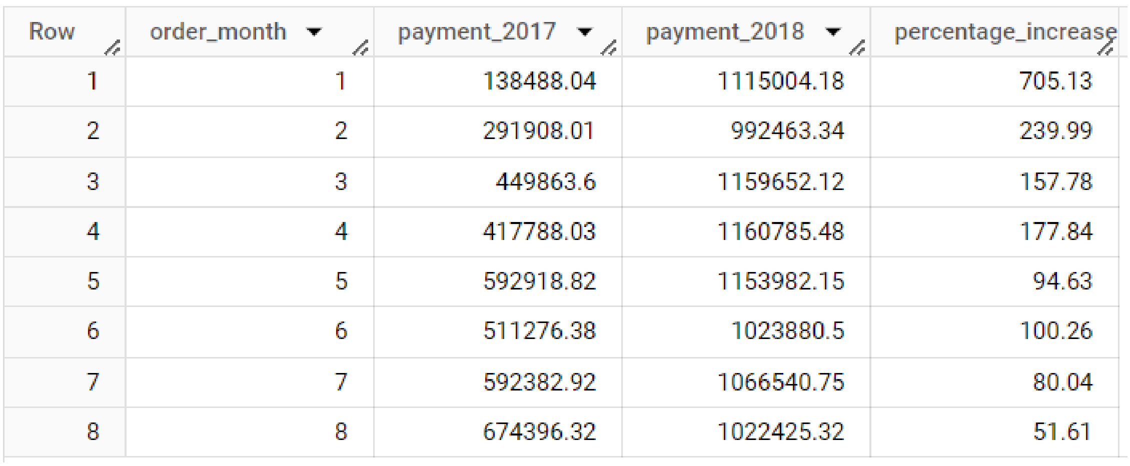
1. **Percentage increase in the cost of orders from year 2017 to 2018.**
   Ans:

```
with cte as (
select extract (year from o.order_purchase_timestamp) as year_,
sum(p.payment_value) as cost
from `Business.payments` p
inner join ` Business.orders` o
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),
next_cost_table as (
select *, lead(cost, 1) over (order by year_ asc) as next_cost from cte)
select *, (next_cost- cost)/cost *100 as per_change from next_cost_table
```



Row	year_	cost	next_cost	per_change
1	2017	3669022.120000...	8694733.839999...	136.9768716466...
2	2018	8694733.839999...	*null*	*null*

```
with ordercosts as (
  select
    extract(year from o.order_purchase_timestamp) as `order_year`,
    extract(month from o.order_purchase_timestamp) as `order_month`,
    round(sum(p.payment_value),2) as `cost_of_orders`
  from ` Business.orders` o
  inner join ` Business.payments` p
  on o.order_id=p.order_id
  where extract(year from o.order_purchase_timestamp) in (2017, 2018) and
        extract(month from o.order_purchase_timestamp) between 1 and 8
  group by 1, 2
  order by 1, 2
)
select
  oc2018.order_month,
  oc2017.cost_of_orders as `payment_2017`,
  oc2018.cost_of_orders as `payment_2018`,
  round((oc2018.cost_of_orders-oc2017.cost_of_orders)/oc2017.cost_of_orders*100, 2)
as `percentage_increase`
  from ordercosts oc2017
  join ordercosts oc2018
  on oc2017.order_month=oc2018.order_month and oc2017.order_year=2017 and
oc2018.order_year=2018
 order by 1;
```



Row	order_month	payment_2017	payment_2018	percentage_increase
1	1	138488.04	1115004.18	705.13
2	2	291908.01	992463.34	239.99
3	3	449863.6	1159652.12	157.78
4	4	417788.03	1160785.48	177.84
5	5	592918.82	1153982.15	94.63
6	6	511276.38	1023880.5	100.26
7	7	592382.92	1066540.75	80.04
8	8	674396.32	1022425.32	51.61

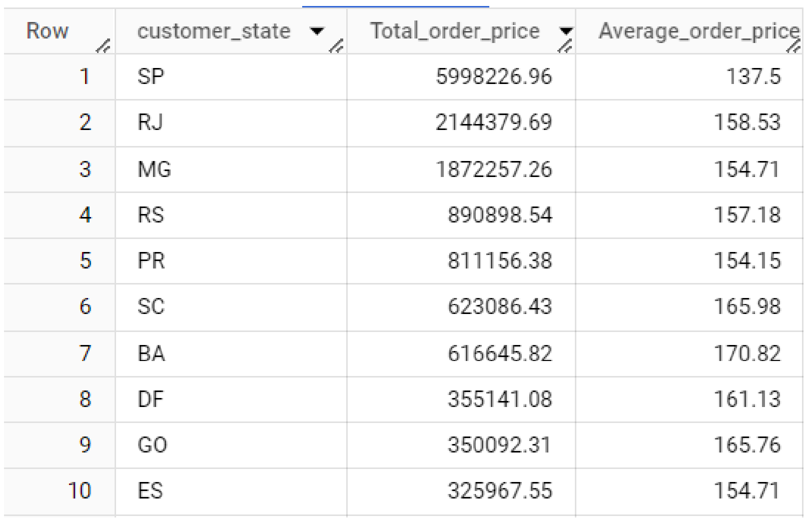
Insights: here the percentage increase in order cost for each month was calculated from year 2017 payments and 2018 payments. For January 2018 the % increase in cost of orders is highest of 705.13% than January 2017. For august month it was low as 51.61% increment.

Recommendations: identify major cost driving factors; marketing and promotional strategies; cost analysis based on product

2. **The Total & Average value of order price for each state.**

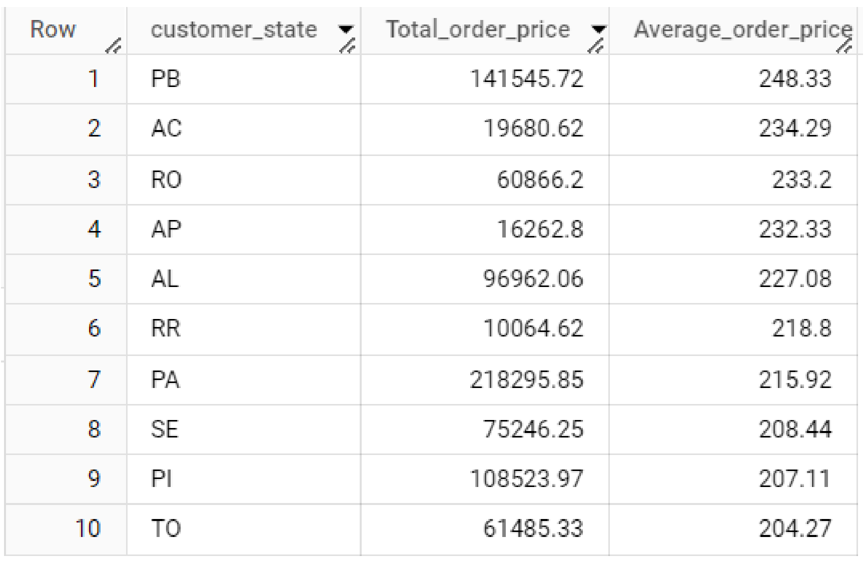
Ans:

```
select
  c.customer_state,
  round(sum(p.payment_value), 2) as `Total_order_price`,
  round(avg(p.payment_value), 2) as `Average_order_price`
from ` Business.payments` p
inner join ` Business.orders` o
on p.order_id=o.order_id
inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1
order by 2 desc, 3 desc;
```



Row	customer_state	Total_order_price	Average_order_price
1	SP	5998226.96	137.5
2	RJ	2144379.69	158.53
3	MG	1872257.26	154.71
4	RS	890898.54	157.18
5	PR	811156.38	154.15
6	SC	623086.43	165.98
7	BA	616645.82	170.82
8	DF	355141.08	161.13
9	GO	350092.31	165.76
10	ES	325967.55	154.71

```
select
  c.customer_state,
  round(sum(p.payment_value), 2) as `Total_order_price`,
  round(avg(p.payment_value), 2) as `Average_order_price`
from ` Business.payments` p
inner join ` Business.orders` o
on p.order_id=o.order_id
inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1
order by 3 desc, 2 desc;
```



Row	customer_state	Total_order_price	Average_order_price
1	PB	141545.72	248.33
2	AC	19680.62	234.29
3	RO	60866.2	233.2
4	AP	16262.8	232.33
5	AL	96962.06	227.08
6	RR	10064.62	218.8
7	PA	218295.85	215.92
8	SE	75246.25	208.44
9	PI	108523.97	207.11
10	TO	61485.33	204.27

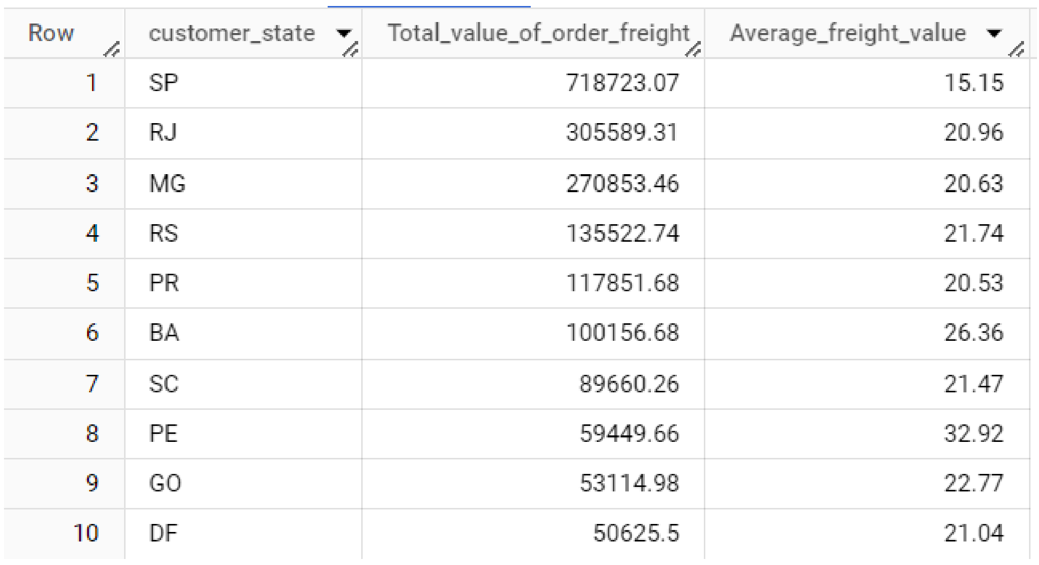
Insights: Here, the information was extracted for each state separately regarding their total and average order price. In one table the state list was provided in descending order based on total order price and in other table the states with descending order of their average order price were provided. The state SP has highest total order price and the state PB has highest average order price.

Recommendations: identifying high and low performing states; product bundles offers; cross selling and upselling; dynamic pricing, supply chain optimization.

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

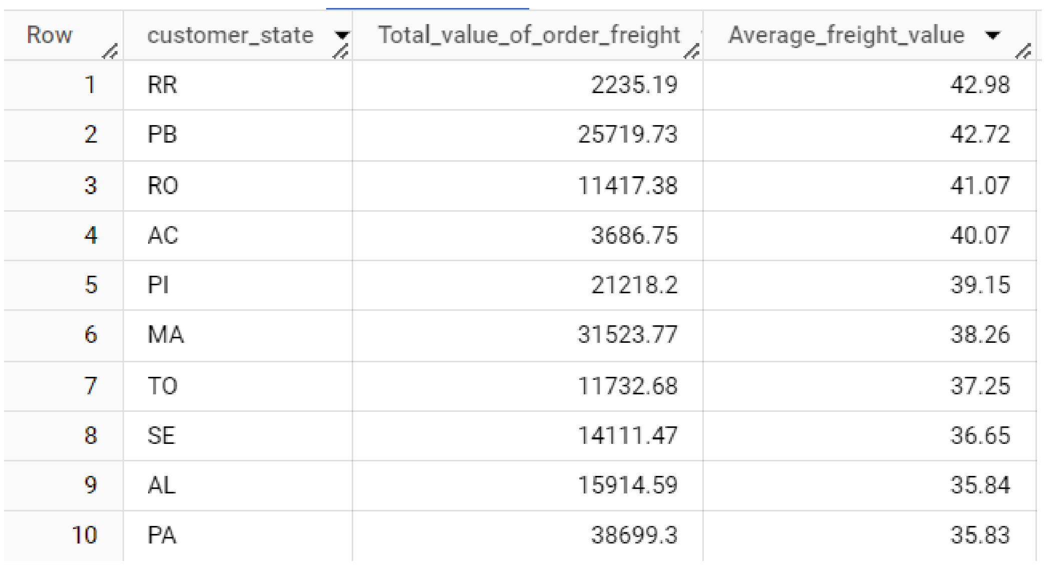
Ans:

```
select
  c.customer_state,
  round(sum(i.freight_value), 2) as `Total_value_of_order_freight`,
  round(avg(i.freight_value), 2) as `Average_freight_value`
from ` Business.order_items` i
inner join ` Business.orders` o
on i.order_id=o.order_id
inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1
order by 2 desc;
```



Row	customer_state	Total_value_of_order_freight	Average_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
7	SC	89660.26	21.47
8	PE	59449.66	32.92
9	GO	53114.98	22.77
10	DF	50625.5	21.04

```
select
  c.customer_state,
  round(sum(i.freight_value), 2) as `Total_value_of_order_freight`,
 round(avg(i.freight_value), 2) as `Average_freight_value`
from ` Business.order_items` i
inner join ` Business.orders` o
on i.order_id=o.order_id
inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1
order by 3 desc;
```



Row	customer_state	Total_value_of_order_freight	Average_freight_value
1	RR	2235.19	42.98
2	PB	25719.73	42.72
3	RO	11417.38	41.07
4	AC	3686.75	40.07
5	PI	21218.2	39.15
6	MA	31523.77	38.26
7	TO	11732.68	37.25
8	SE	14111.47	36.65
9	AL	15914.59	35.84
10	PA	38699.3	35.83

Insights: Here, the information was extracted for each state separately regarding their total and average order freight value. In one table the state list was provided in descending order based on total order freight value and in other table the states with descending order of their average order freight value were provided. The state SP has highest total value of order freight. The state RR has highest average value of order freight.

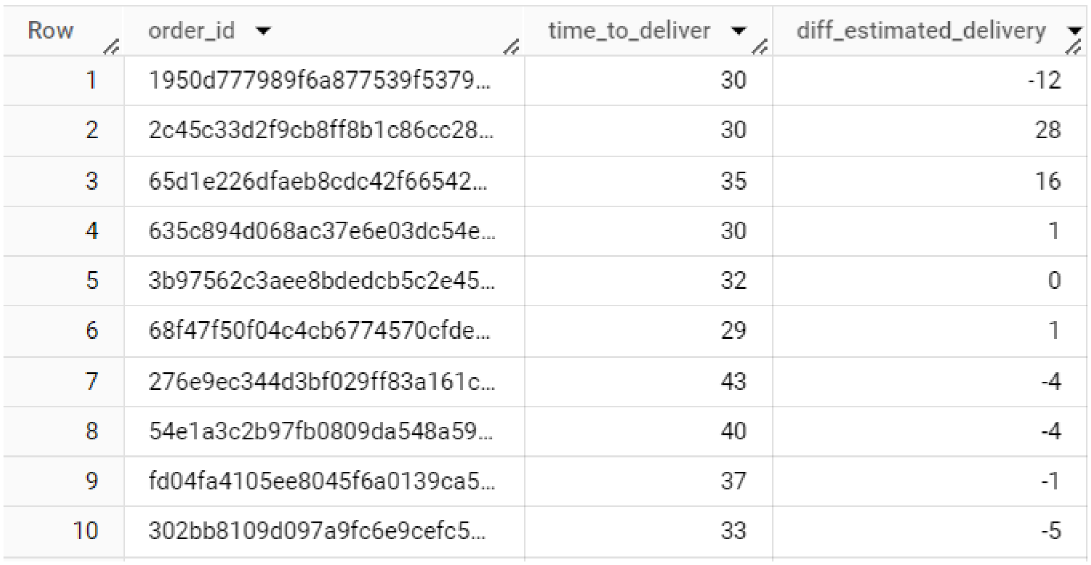
Recommendations: identify high-cost regions; negotiate shipping carriers; optimize shipping routes; freight partnership; shipping strategies; shipping discounts and offers; customer segmentation.

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

1. **Number of days taken to deliver each order from the order's purchase date.**

Ans:

```
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_date, day) as
`diff_estimated_delivery`
from ` Business.orders`;
```



Row	order_id	time_to_deliver	diff_estimated_delivery
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	28
3	65d1e226dfaeb8cdc42f66542...	35	16
4	635c894d068ac37e6e03dc54e...	30	1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	1
7	276e9ec344d3bf029ff83a161c...	43	-4
8	54e1a3c2b97fb0809da548a59...	40	-4
9	fd04fa4105ee8045f6a0139ca5...	37	-1
10	302bb8109d097a9fc6e9cefc5...	33	-5

Insights: here the information regarding number of days taken to delivery and estimated delivery were extracted for each order.

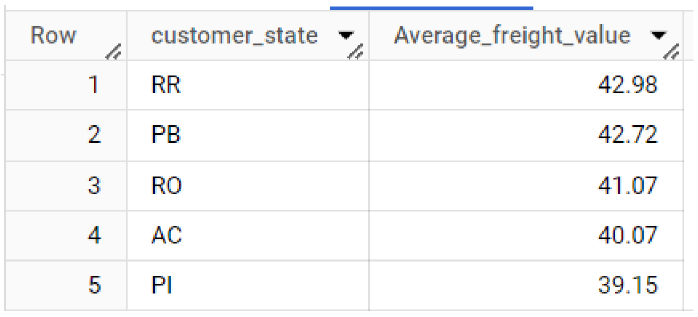
Recommendations: investigate the reasons for delivery time difference between actual and estimated; delivery needs to be improved in those cases where the actual delivery is more than the estimated delivery.

2. **The top states with the highest & lowest average freight value.**

Ans:

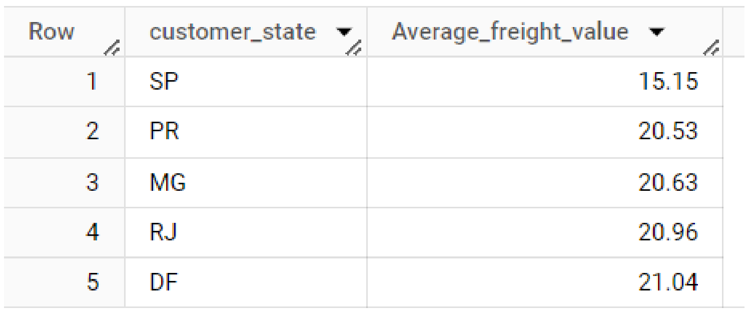
```
select
  c.customer_state,
  round(avg(i.freight_value), 2) as `Average_freight_value`
from ` Business.order_items` i
inner join ` Business.orders` o
```

```
on i.order_id=o.order_id
inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1
order by 2 desc
limit 5;
```



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

```
select
  c.customer_state,
  round(avg(i.freight_value), 2) as `Average_freight_value`
from ` Business.order_items` i
inner join ` Business.orders` o
on i.order_id=o.order_id
inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1
order by 2
limit 5;
```



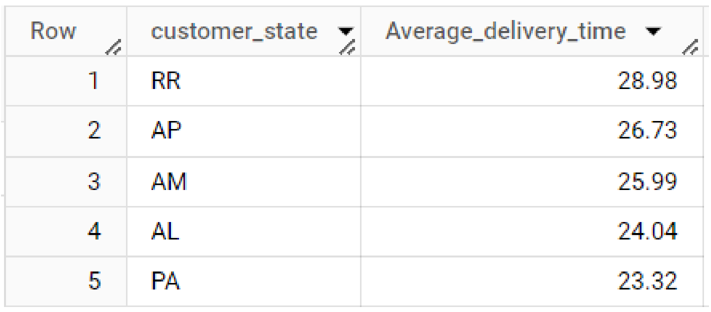
Row	customer_state	Average_freight_value
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

Insights: here in the first table the top 5 states with highest average freight value are extracted whereas in the second table top 5 states with lowest average freight value are extracted.

Recommendations: for states with highest average freight value: optimize shipping routes, negotiate shipping contracts, packing efficiency, establishing regional warehouses; for states with lowest average freight value: explore new possible opportunities for market expansion, attract more customers through customer incentives, customer satisfaction.

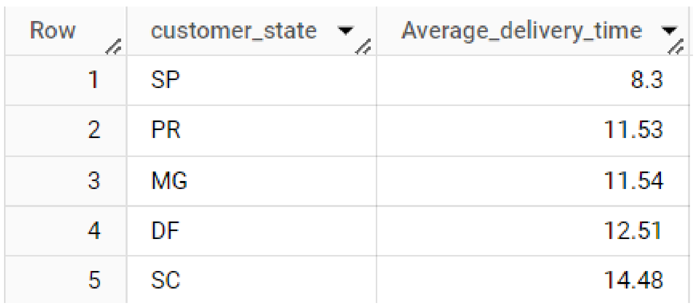
3. **The top states with the highest & lowest average delivery time.**
   Ans:

```
select
  c.customer_state,
  round(avg(date_diff(o.order_delivered_customer_date,
    o.order_purchase_timestamp, day)), 2) as `Average_delivery_time`
from ` Business.orders` o inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1
order by 2 desc
limit 5;
```



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

```
select
  c.customer_state,
  round(avg(date_diff(o.order_delivered_customer_date, o.order_purchase_timestamp,
day)), 2) as `Average_delivery_time`
from ` Business.orders` o inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1
order by 2
limit 5;
```



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

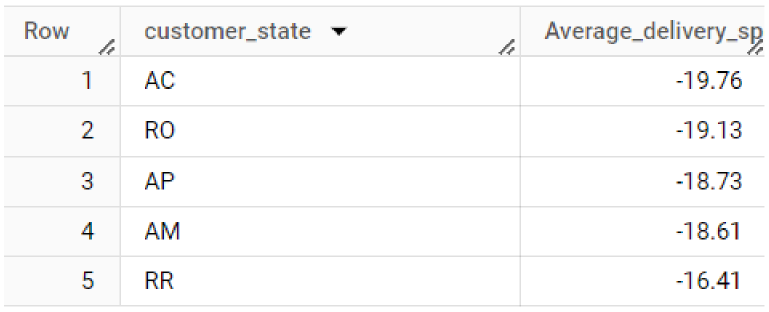
Insights: here in the first table the top 5 states with highest average delivery time are extracted whereas in the second table top 5 states with lowest average delivery time are extracted.

Recommendations: for states with highest average delivery time: identify the constrains in the shipping routes, optimize the shipping processes and routes, collaborate with carriers, establish regional warehouses, and communicate with customers. For states with lowest average delivery time: attract more customers through speed delivery, scalability, maintain high standards, and customer feedbacks.

4. **The top states with order delivery faster than estimated delivery time.**

   Ans:

```
select
  c.customer_state,
  round(avg(date_diff(o.order_delivered_customer_date,
   o.order_estimated_delivery_date, day)), 2) as `Average_delivery_speed`
from ` Business.orders` o inner join ` Business.customers` c
on o.customer_id=c.customer_id
group by 1
order by 2
limit 5;
```



Row	customer_state	Average_delivery_sp
1	AC	-19.76
2	RO	-19.13
3	AP	-18.73
4	AM	-18.61
5	RR	-16.41

Insights: here the top 5 states with fastest delivery than estimated delivery were extracted.

Recommendations: communicate with customers, attract more customers by highlighting the delivery speed, upsell or cross sell more products.

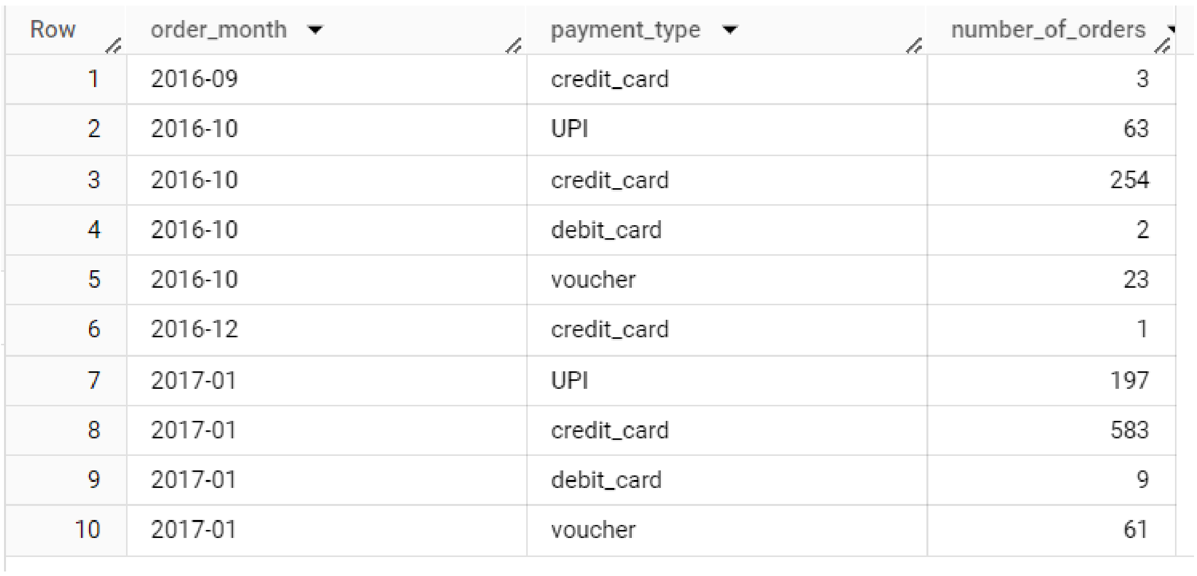
6. Payment analysis:

1. **Orders placed using different payment types.**

Ans:

```
select
  format_datetime('%Y-%m', o.order_purchase_timestamp) as `order_month`,
  p.payment_type,
  count(o.order_id) as `number_of_orders`
from ` Business.orders` o
inner join ` Business.payments` p
on o.order_id=p.order_id
group by 1, 2
```

```
order by 1,2;
```



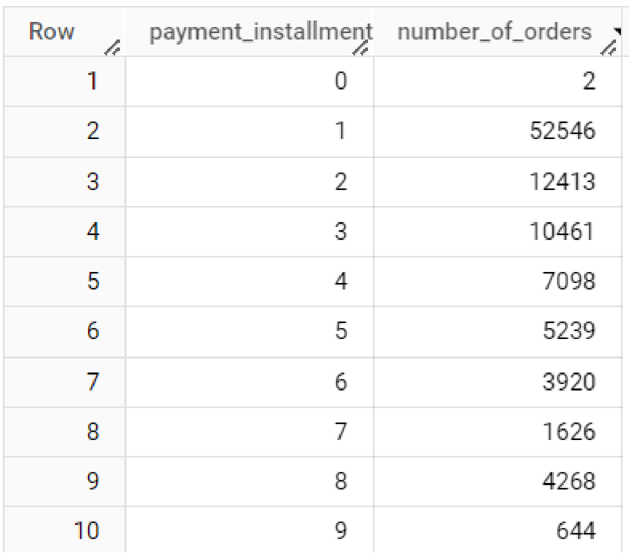
Row	order_month	payment_type	number_of_orders
1	2016-09	credit_card	3
2	2016-10	UPI	63
3	2016-10	credit_card	254
4	2016-10	debit_card	2
5	2016-10	voucher	23
6	2016-12	credit_card	1
7	2017-01	UPI	197
8	2017-01	credit_card	583
9	2017-01	debit_card	9
10	2017-01	voucher	61

Insights: here the data regarding number of orders through different payment types on monthly basis were extracted. The information is grouped and then ordered based on order month and payment time.

Recommendations: payment method promotions, optimize payment process, consider different payment preferences, membership or subscription, security.

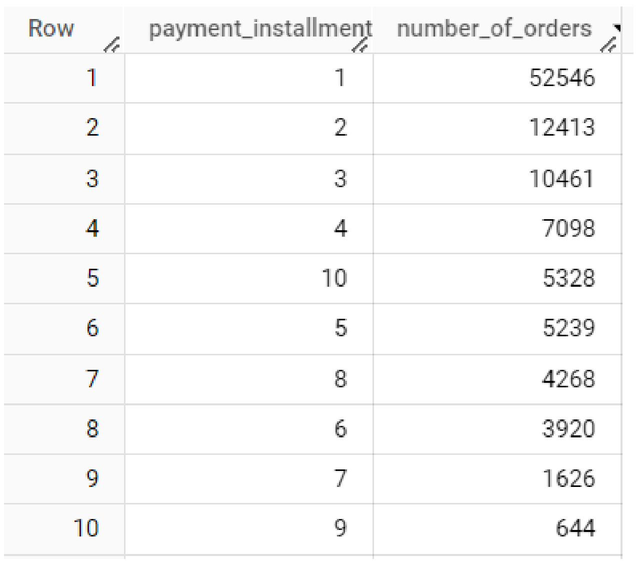
2. **Orders placed on the basis of the paid payment instalments.**
   Ans:

```
select payment_installments, count(order_id) as `number_of_orders`
from ` Business.payments`
group by 1
order by 1;
```



Row	payment_installment	number_of_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

```
select payment_installments, count(order_id) as `number_of_orders`
from ` Business.payments`
group by 1
order by 2 desc;
```



Row	payment_installment	number_of_orders
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	10	5328
6	5	5239
7	8	4268
8	6	3920
9	7	1626
10	9	644

Insights: here number of orders made through different payment instalments were extracted. Highest orders were made through one time payment instalment.

Recommendations: instalment payment promotions, flexibility in payment instalments, financial education, cross-sell options.