****note- analysis will be mentioned and colour differently**

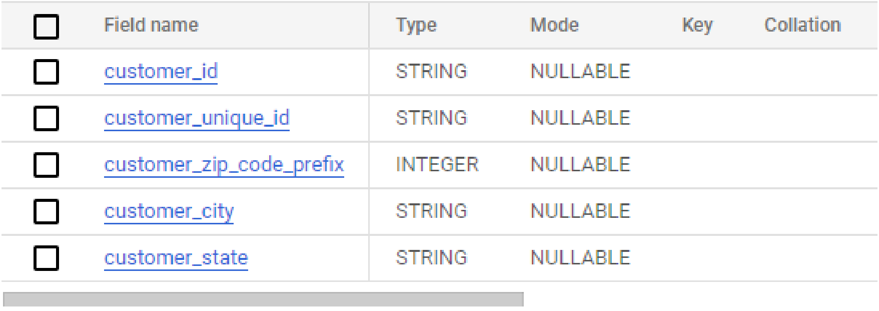
1.Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset

1. Data type of columns in a table

Data type of columns in a table

Table Name	Column Name	Type
customers	coustomer_id	STIRNG
	cutomer_unique_id	STIRNG
	customer_zip_code_prefix	INTEGER
	customer_city	STRING
	customer_state	STRING
geolocation	geolocation_zip_code_prefix	INTEGER
	geolocation_lat	FLOAT
	geolocation_lng	FLOAT
	geolocation_city	STRING
	geolocation_state	STRING
order_items	order_id	STRING
	order_item_id	INTEGER
	product_id	STRING
	seller_id	STRING
	shipping_limit_date	TIMESTAMP
	price	FLOAT
	freight_value	FLOAT
order_reveiws	review_id	STRING
	order_id	STRING
	review_score	INTEGER
	review_comment_title	STRING
	review_creation_date	TIMESTAMP
	review_answer_timestamp	TIMESTAMP
orders	order_id	STRING
	customer_id	STRING
	order_status	STRING
	order_purchase_timestamp	TIMESTAMP
	order_approved_at	TIMESTAMP
	order_delivered_carrier_date	TIMESTAMP
	order_delivered_customer_date	TIMESTAMP
	order_estimated_delivery_date	TIMESTAMP
payments	order_id	STRING
	payment_sequential	INTEGER
	payment_type	STRING
	payment_installments	INTEGER
	payment_value	FLOAT
products	product_id	STRING
	product_category	STRING

	product_name_length	INTEGER
	product_description_length	INTEGER
	product_photos_qty	INTEGER
	product_weight_g	INTEGER
	product_length_cm	INTEGER
	product_height_cm	INTEGER
	product_width_cm	INTEGER
seller	seller_id	STRING
	seller_zip_code_prefix	INTEGER
	seller_city	STRING
	sellar_state	STRING



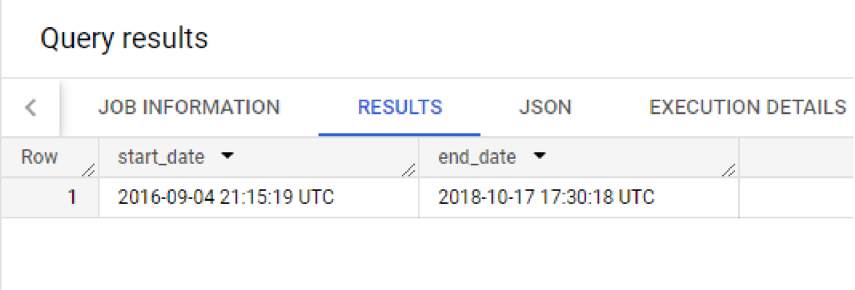
□	Field name	Type	Mode	Key	Collation
□	customer_id	STRING	NULLABLE		
□	customer_unique_id	STRING	NULLABLE		
□	customer_zip_code_prefix	INTEGER	NULLABLE		
□	customer_city	STRING	NULLABLE		
□	customer_state	STRING	NULLABLE		

Analysis:- There are mainly 4 types of datatype in these tables:

- String(22)
- Float(5)
- Integer(14)
- Timestamp(8)

2. Time period for which the data is given

```
SELECT
MIN(order_purchase_timestamp)AS start_date,
MAX(order_purchase_timestamp) AS end_date
FROM `e_commerce.orders`;
```



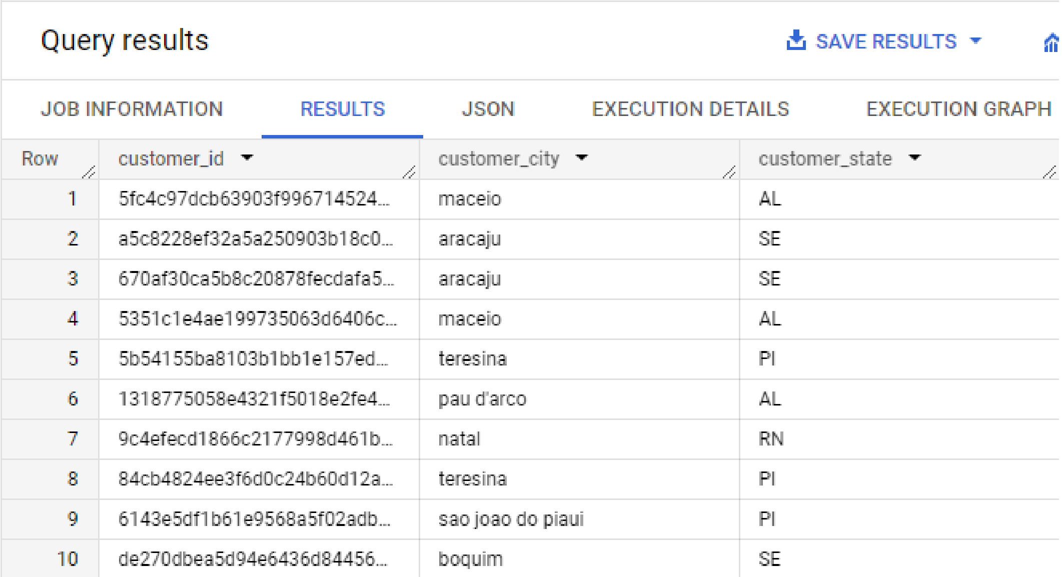
Row	start_date	end_date
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Analysis:- Based on the query result it is understood that the given data is between the time frame of 04-09-2016 to 17-10-2018

3. Cities and States of customers ordered during the given period

```
SELECT
o.customer_id,
c.customer_city,
c.customer_state
FROM `e_commerce.orders` AS o
LEFT JOIN
`e_commerce.customers` AS c
ON
o.customer_id = c.customer_id;
```

Query Result



Row	customer_id	customer_city	customer_state
1	5fc4c97dcb63903f996714524...	maceio	AL
2	a5c8228ef32a5a250903b18c0...	aracaju	SE
3	670af30ca5b8c20878fecdafa5...	aracaju	SE
4	5351c1e4ae199735063d6406c...	maceio	AL
5	5b54155ba8103b1bb1e157ed...	teresina	PI
6	1318775058e4321f5018e2fe4...	pau d'arco	AL
7	9c4efecd1866c2177998d461b...	natal	RN
8	84cb4824ee3f6d0c24b60d12a...	teresina	PI
9	6143e5df1b61e9568a5f02adb...	sao joao do piaui	PI
10	de270dbea5d94e6436d84456...	boquim	SE

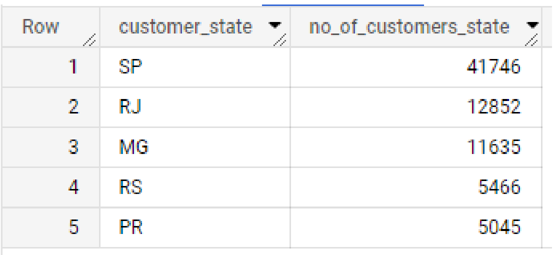
Analysis:- The query result shows the total list of customers distributed across each cities of the respective states.

3 a)Number of customer across state.

```
SELECT
c.customer_state,
COUNT(c.customer_state) AS no_of_customers_state
FROM `e_commerce.orders` AS o
LEFT JOIN
`e_commerce.customers` AS c
ON
o.customer_id = c.customer_id
group by c.customer_state;
```

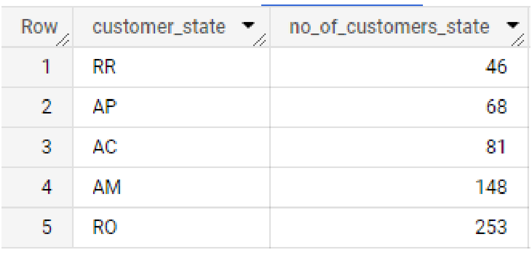
Query Result

Top5 :-



Row	customer_state	no_of_customers_state
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045

Bottom5 :-



Row	customer_state	no_of_customers_state
1	RR	46
2	AP	68
3	AC	81
4	AM	148
5	RO	253

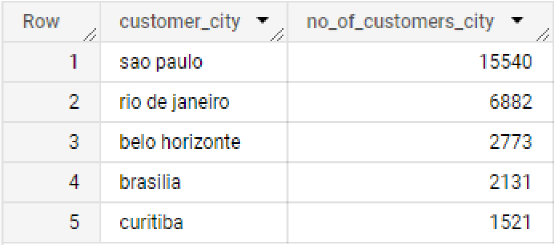
Analysis:- The above two table shows the top 5 states highest no of customers and top 5 states of lowest no of customers

3 b)Number of customer across state.

```
SELECT
c.customer_city,
COUNT(c.customer_city) AS no_of_customers_city
FROM `e_commerce.orders` AS o
LEFT JOIN
`e_commerce.customers` AS c
ON
o.customer_id = c.customer_id
group by c.customer_city;
```

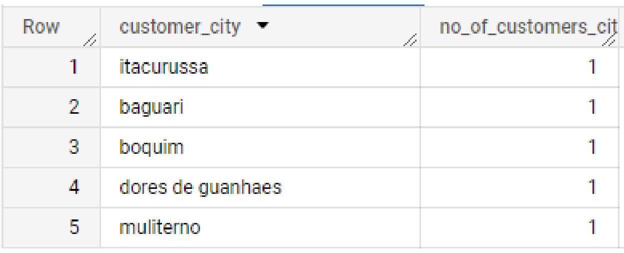
Query Result

Top5 :-



Row	customer_city	no_of_customers_city
1	sao paulo	15540
2	rio de janeiro	6882
3	belo horizonte	2773
4	brasilia	2131
5	curitiba	1521

Bottom5 :-



Row	customer_city	no_of_customers_cit
1	itacurussa	1
2	baguari	1
3	boquim	1
4	dores de guanhaes	1
5	muliterno	1

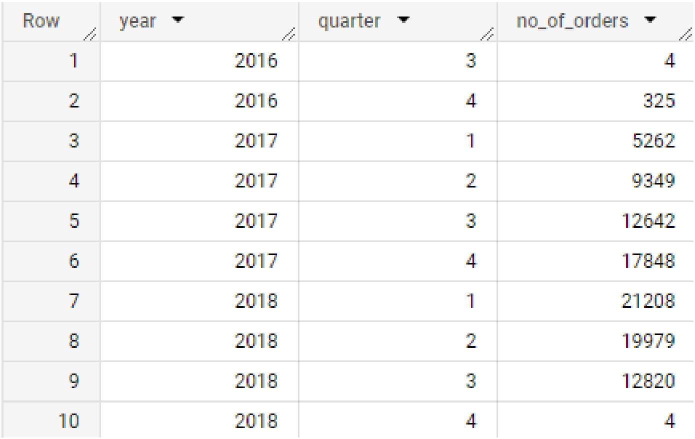
Analysis:- The above two table shows the top 5 cities highest no of customers and top 5 cities of lowest no of customers

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?**

```
SELECT
*
FROM(
SELECT
EXTRACT (YEAR FROM order_purchase_timestamp) AS year,
EXTRACT(QUARTER FROM order_purchase_timestamp) AS quarter,
COUNT(order_id) AS no_of_orders
FROM `e_commerce.orders`
GROUP BY EXTRACT(QUARTER FROM order_purchase_timestamp),EXTRACT (YEAR FROM
order_purchase_timestamp)
)AS x
ORDER BY x.year,x.quarter;
```

Query Result:-



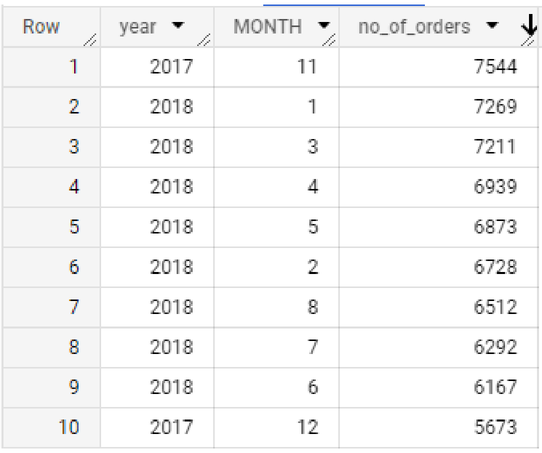
Row	year	quarter	no_of_orders
1	2016	3	4
2	2016	4	325
3	2017	1	5262
4	2017	2	9349
5	2017	3	12642
6	2017	4	17848
7	2018	1	21208
8	2018	2	19979
9	2018	3	12820
10	2018	4	4

Analysis:- Based on the trend we can see exponential growth in number of orders and it reaches its maximum during the first quarter of the year 2018,after this point there seems to be an exponential delay the in trend and finally by the end of 4th quarter number of orders have reached its minimum.

Can we see some seasonality with peaks at specific months?

```
SELECT
*
FROM(
SELECT
EXTRACT (YEAR FROM order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS MONTH,
COUNT(order_id) AS no_of_orders
FROM `e_commerce.orders`
GROUP BY EXTRACT(MONTH FROM order_purchase_timestamp),EXTRACT (YEAR FROM
order_purchase_timestamp)
)AS x
ORDER BY x.year,x.month ;
```

Query Result:-



Row	year	MONTH	no_of_orders
1	2017	11	7544
2	2018	1	7269
3	2018	3	7211
4	2018	4	6939
5	2018	5	6873
6	2018	2	6728
7	2018	8	6512
8	2018	7	6292
9	2018	6	6167
10	2017	12	5673

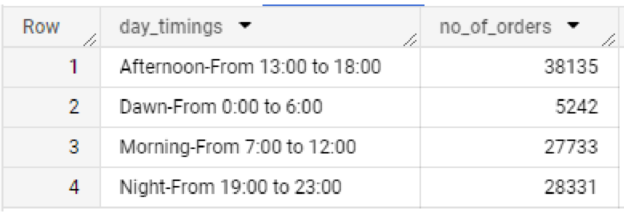
Analysis:- Based on the table, most no of orders were placed during October month of year 2017 followed by month January in 2018,

2) **What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?**

```
SELECT
x.day_timings,
count(x.order_id) AS no_of_orders
FROM(
SELECT
order_id,
EXTRACT (HOUR FROM order_purchase_timestamp) AS time,
CASE
WHEN EXTRACT (HOUR FROM order_purchase_timestamp) BETWEEN 0 AND 6
THEN "Dawn-From 0:00 to 6:00 "
WHEN EXTRACT (HOUR FROM order_purchase_timestamp) BETWEEN 7 AND 12
THEN "Morning-From 7:00 to 12:00 "
WHEN EXTRACT (HOUR FROM order_purchase_timestamp) BETWEEN 13 AND 18
THEN "Afternoon-From 13:00 to 18:00 "
ELSE "Night-From 19:00 to 23:00  "
END AS day_timings
FROM `e_commerce.orders`
)AS x
GROUP BY x.day_timings
```

```
ORDER BY x.day_timings;
```

Query Result:-



Row	day_timings	no_of_orders
1	Afternoon-From 13:00 to 18:00	38135
2	Dawn-From 0:00 to 6:00	5242
3	Morning-From 7:00 to 12:00	27733
4	Night-From 19:00 to 23:00	28331

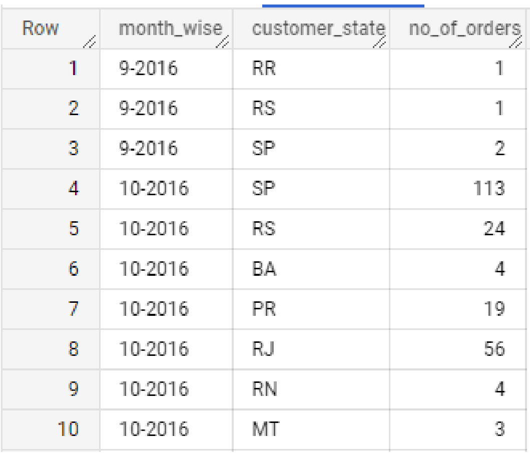
Analysis:- From the table we understand that highest number of orders are during afternoon hours between a time interval of 13:00 to 18:00, where as comparatively lowest number of orders are during Dawn or early Morning between a time interval of 0:00 to 6:00.

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

1) Get month on month by states

```
SELECT
CONCAT(x.month,"-",x.year)AS month_wise,
x.customer_state,
COUNT(DISTINCT x.order_id) AS no_of_orders
FROM(
SELECT
EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
o.customer_id,
c.customer_state,
o.order_id
FROM `e_commerce.orders` AS o
RIGHT JOIN
`e_commerce.customers` AS c
ON
o.customer_id = c.customer_id
) AS x
GROUP BY x.customer_state,x.month,x.year
ORDER BY x.year,x.month;
```

Query Result:-

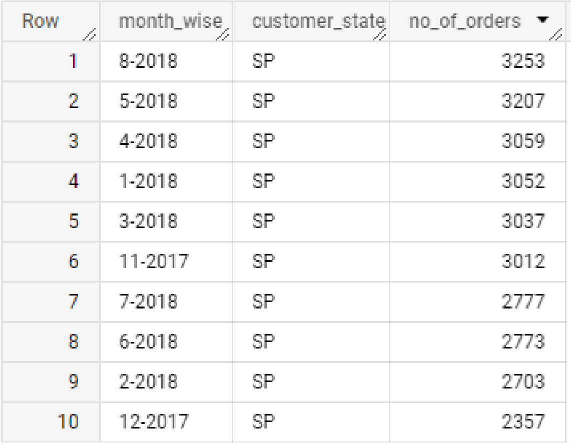


Row	month_wise	customer_state	no_of_orders
1	9-2016	RR	1
2	9-2016	RS	1
3	9-2016	SP	2
4	10-2016	SP	113
5	10-2016	RS	24
6	10-2016	BA	4
7	10-2016	PR	19
8	10-2016	RJ	56
9	10-2016	RN	4
10	10-2016	MT	3

Analysis:- The following table shown the monthly wise number of orders in each state of Brazil.

1a) Highest number of orders across state

Query Result:-



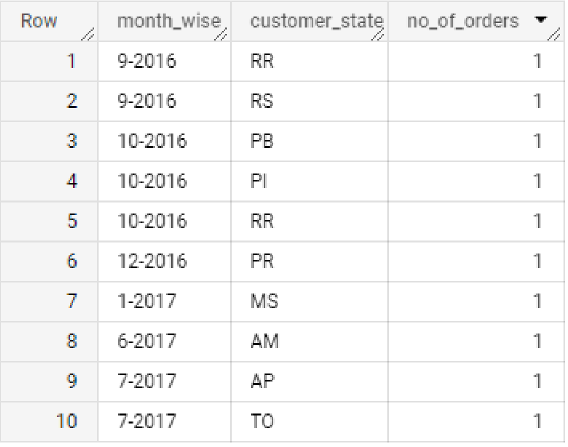
Row	month_wise	customer_state	no_of_orders
1	8-2018	SP	3253
2	5-2018	SP	3207
3	4-2018	SP	3059
4	1-2018	SP	3052
5	3-2018	SP	3037
6	11-2017	SP	3012
7	7-2018	SP	2777
8	6-2018	SP	2773
9	2-2018	SP	2703
10	12-2017	SP	2357

Analysis:- Based on the table country code ”SP” which is São Paulo has the highest number of orders consistently around many months of the year 2018 and 2017 respectively

***Note:- exact name is from internet source**

1b) Lowest number of order across state

Query Result:-



Row	month_wise	customer_state	no_of_orders
1	9-2016	RR	1
2	9-2016	RS	1
3	10-2016	PB	1
4	10-2016	PI	1
5	10-2016	RR	1
6	12-2016	PR	1
7	1-2017	MS	1
8	6-2017	AM	1
9	7-2017	AP	1
10	7-2017	TO	1

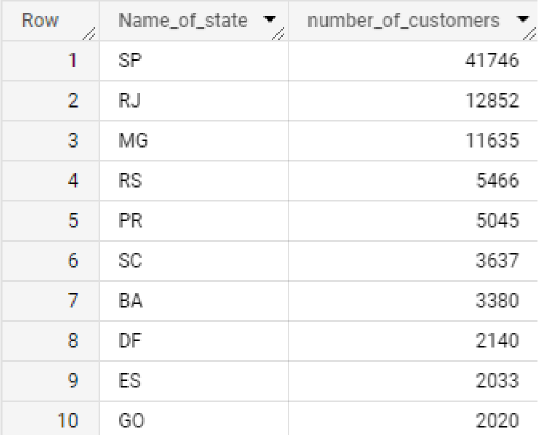
Analysis:- Countries with code- RR,RS,PB,PI etc. Least number of orders are shared by numerous states scattered across different regions of Brazil. It is also visible that most of it happened during the initial months of our time interval.

2) Distribution of customers across the states in Brazil

```
SELECT
c.customer_state AS Name_of_state,
count(c.customer_id) AS number_of_customers
FROM `e_commerce.orders` AS o
```

```
LEFT JOIN
`e_commerce.customers` AS c
ON
o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY count(c.customer_id) DESC;
```

Query Result:-

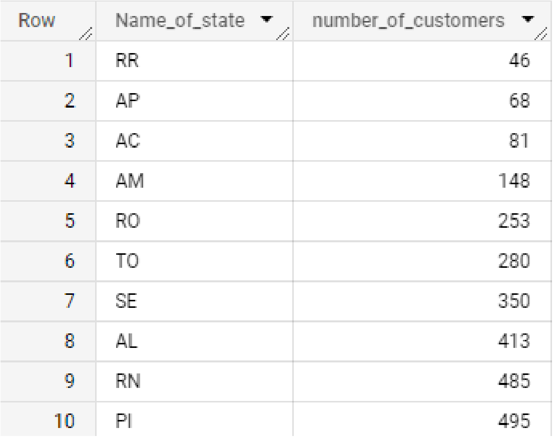


Row	Name_of_state	number_of_customers
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

Analysis:- Most of the customers are distributed among São Paulo(SP), Rio de Janeiro(RJ), Minas Gerais(MG). São Paulo has the highest number of customers. They also have the highest number of orders compared to other states.

***Note:- exact name is from internet source**

Query Result:-



Row	Name_of_state	number_of_customers
1	RR	46
2	AP	68
3	AC	81
4	AM	148
5	RO	253
6	TO	280
7	SE	350
8	AL	413
9	RN	485
10	PI	495

Analysis:- Roraima(RR) shows the least number of customers. Based on the previous analysis they have also shown the least number of orders.

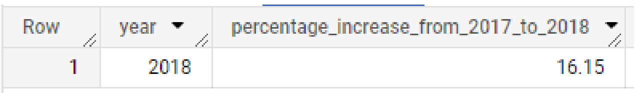
***Note:- exact name is from internet source**

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**

```
WITH monthwise_order AS (SELECT
x.month_year,
x.year,
SUM(x.payment_value) as monthpayment
FROM( SELECT
      EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
      EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
      CONCAT (EXTRACT(MONTH FROM o.order_purchase_timestamp),"-",EXTRACT(YEAR FROM
o.order_purchase_timestamp)) AS month_year,
      p.payment_value
      FROM `e_commerce.orders` AS o
      LEFT JOIN
      `e_commerce.payments` AS 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 7
      ORDER BY year,month) as x
  GROUP BY x.month_year,x.year)
  SELECT
  y.year,
  ((y.totalcost_2018-y.totalcost_2017)/y.totalcost_2017)*100 AS
percentage_increase_from_2017_to_2018
  FROM(
    SELECT
    year,
    sum(CASE WHEN month_year BETWEEN "1-2018" and "7-2018" THEN monthpayment ELSE 0
END) AS totalcost_2018,
    sum(CASE WHEN month_year BETWEEN "1-2017" and "7-2017" THEN monthpayment ELSE 0
END) AS totalcost_2017
    FROM monthwise_order
    GROUP BY year) AS y
  WHERE y.year=2018;
```

Query Result:-



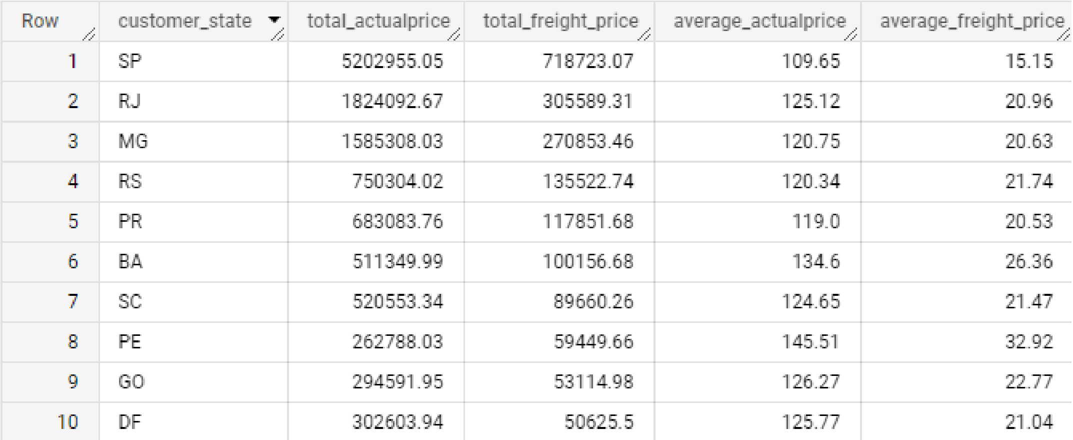
Row	year	percentage_increase_from_2017_to_2018
1	2018	16.15

Analysis:- Based on the analysis, it seems from year 2017 to 2018, there has been increase of 16.15% in the Overall cost of orders.

2) **Mean & Sum of price and freight value by customer state**

```
SELECT
x.customer_state,
SUM(x.price) AS total_actualprice,
SUM(x.freight_value) AS total_regionwise_price,
AVG(x.price) AS average_actualprice,
AVG(x.freight_value) AS avearage_regionwise_price
FROM(
SELECT
c.customer_state,
ot.price,
ot.freight_value
FROM `e_commerce.orders` AS o
RIGHT JOIN
`e_commerce.order_items` AS ot
ON
o.order_id = ot.order_id
RIGHT JOIN
`e_commerce.customers`AS c
ON
o.customer_id = c.customer_id
) AS x
GROUP BY x.customer_state
ORDER BY x.customer_state;
```

Query Result:-



Row	customer_state	total_actualprice	total_freight_price	average_actualprice	average_freight_price
1	SP	5202955.05	718723.07	109.65	15.15
2	RJ	1824092.67	305589.31	125.12	20.96
3	MG	1585308.03	270853.46	120.75	20.63
4	RS	750304.02	135522.74	120.34	21.74
5	PR	683083.76	117851.68	119.0	20.53
6	BA	511349.99	100156.68	134.6	26.36
7	SC	520553.34	89660.26	124.65	21.47
8	PE	262788.03	59449.66	145.51	32.92
9	GO	294591.95	53114.98	126.27	22.77
10	DF	302603.94	50625.5	125.77	21.04

Analysis:- Based on the trend highest value for actual prize and highest value for Freight_value (Price rate at which a product is delivered from one point to another) are seen in São Paulo ("SP") with average sum prize and freight value around 109.65 and 15.15 respectively.

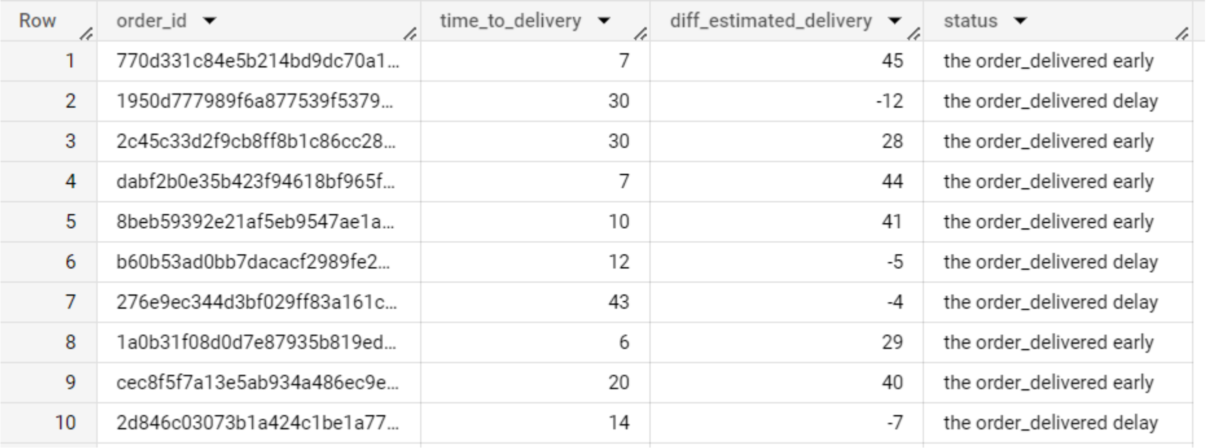
***Note:- exact name is from internet source**

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_delivered_customer_date-order_purchase_timestamp
   diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

```
SELECT
order_id,
DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) AS
time_to_delivery,
DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) AS
diff_estimated_delivery,
  CASE
    WHEN DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)
> 0 THEN "the order_delivered early"
    WHEN DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)
< 0 THEN "the order_delivered delay"
    WHEN DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)
= 0 THEN "the order_delivered exactly on estimated date"
    ELSE NULL
  END AS status
FROM `e_commerce.orders`;
```

Query Result:-



Row	order_id	time_to_delivery	diff_estimated_delivery	status
1	770d331c84e5b214bd9dc70a1...	7	45	the order_delivered early
2	1950d777989f6a877539f5379...	30	-12	the order_delivered delay
3	2c45c33d2f9cb8ff8b1c86cc28...	30	28	the order_delivered early
4	dabf2b0e35b423f94618bf965f...	7	44	the order_delivered early
5	8beb59392e21af5eb9547ae1a...	10	41	the order_delivered early
6	b60b53ad0bb7dacacf2989fe2...	12	-5	the order_delivered delay
7	276e9ec344d3bf029ff83a161c...	43	-4	the order_delivered delay
8	1a0b31f08d0d7e87935b819ed...	6	29	the order_delivered early
9	cec8f5f7a13e5ab934a486ec9e...	20	40	the order_delivered early
10	2d846c03073b1a424c1be1a77...	14	-7	the order_delivered delay

Analysis:- Based on the table, we understand that,

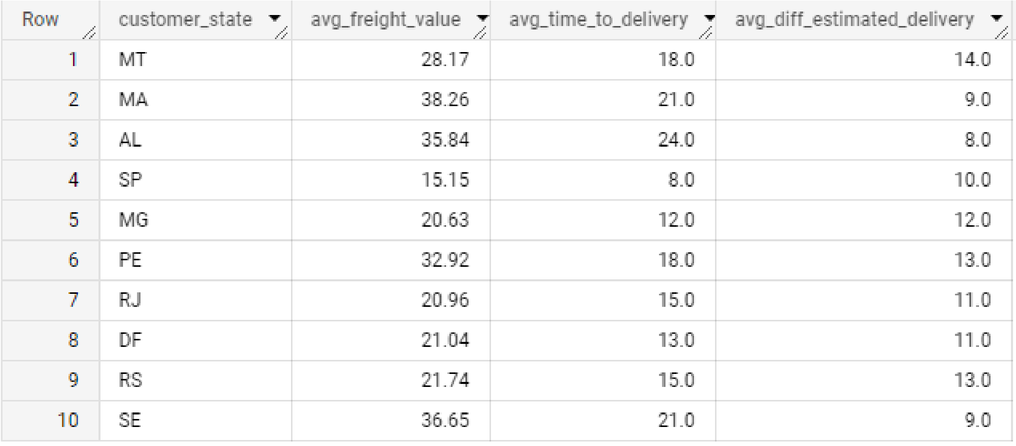
1. if diff_estimated_delivery is a negative value then the order was delivered late by the value as number of days.
2. if diff_estimated_delivery is a postive value then the order was delivered early by the value as number of days.
3. if diff_estimated_delivery is a zero then the order was delivered exactlu on the estimated date.

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

SELECT

```
c.customer_state,
AVG(ot.freight_value) AS avg_freight_value,
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)
)) AS avg_time_to_delivery,
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date
,DAY))) AS avg_diff_estimated_delivery
FROM `e_commerce.orders` AS o
JOIN
`e_commerce.order_items` AS ot
ON o.order_id = ot.order_id
JOIN
`e_commerce.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state ;
```

Query Result:-

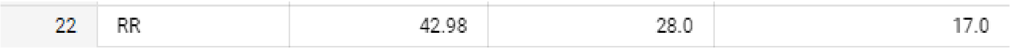


Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	MT	28.17	18.0	14.0
2	MA	38.26	21.0	9.0
3	AL	35.84	24.0	8.0
4	SP	15.15	8.0	10.0
5	MG	20.63	12.0	12.0
6	PE	32.92	18.0	13.0
7	RJ	20.96	15.0	11.0
8	DF	21.04	13.0	11.0
9	RS	21.74	15.0	13.0
10	SE	36.65	21.0	9.0

Analysis:- Based on the query result table for each state across Brazil the average value of Freight price is represented along with the average time taken from the order placed to the delivery of the order and also average difference of day between the estimated and delivered dates are also represented.
Based on these the previous and present information from queries, following are the analysis for the states with highest and lower no of orders respectively:-

i) São Paulo ("SP")- the highest number of orders are from SP. The state has an average of 15.15 freight value, orders were delivered in an average of 8 days and on an average delivery was done around 10days early from the estimated date of delivery

ii) Roraima("RR") - lowest number of orders are from RR. The state has an average of 42.98 freight value, orders were delivered in an average of 28 days and on an average delivery was done around 17days early from the estimated date of delivery



22	RR	42.98	28.0	17.0

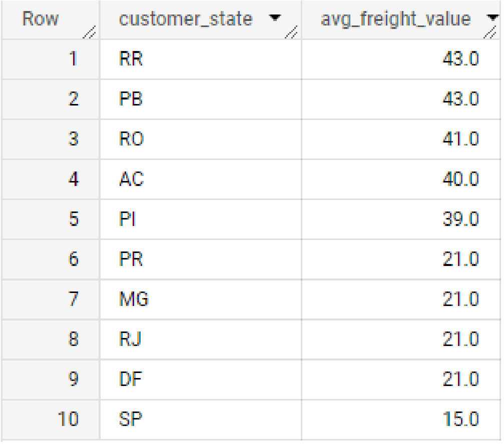
***Note:- exact name is from internet source**

4) Sort the date to get the following

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

```
WITH top AS (SELECT
c.customer_state,
AVG(ot.freight_value) AS avg_freight_value
FROM `e_commerce.orders` AS o
JOIN
`e_commerce.order_items` AS ot
ON o.order_id = ot.order_id
JOIN
`e_commerce.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY AVG(ot.freight_value) DESC
LIMIT 5),
bottom AS (SELECT
c.customer_state,
AVG(ot.freight_value) AS avg_freight_value,
FROM `e_commerce.orders` AS o
JOIN
`e_commerce.order_items` AS ot
ON o.order_id = ot.order_id
JOIN
`e_commerce.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY AVG(ot.freight_value) ASC
LIMIT 5)
SELECT
customer_state,
avg_freight_value
FROM (
  select customer_state, top.avg_freight_value
  from top
  union all
  select customer_state, bottom.avg_freight_value
  from bottom
)
ORDER BY avg_freight_value DESC;
```

Query Result:-



Row	customer_state	avg_freight_value
1	RR	43.0
2	PB	43.0
3	RO	41.0
4	AC	40.0
5	PI	39.0
6	PR	21.0
7	MG	21.0
8	RJ	21.0
9	DF	21.0
10	SP	15.0

Analysis:- The following table represent the 5 state with highest average freight value and lowest average respectively, first 5 in the table are highest and next 5 are the lowest states.

So based on this we can conclude that highest no

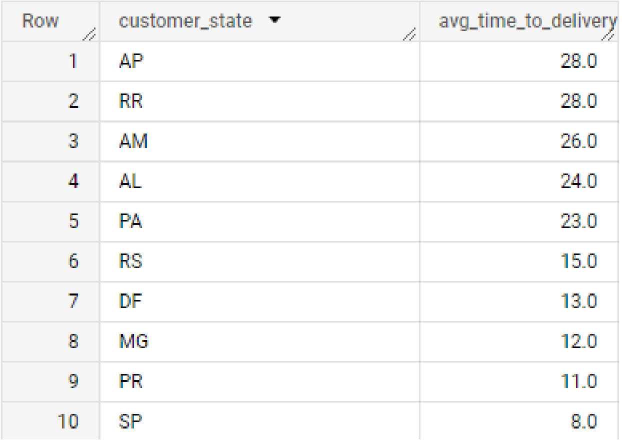
a) RR (Roraima) has the highest average freight value and that has resulted in lowest no of orders from this area.
b) SP (São Paulo) has the lowest average freight value and that has resulted in highest no of orders from this area.

***Note:- exact name is from internet source**

6) Top 5 states with highest/lowest average time to delivery

```
WITH highest AS (SELECT
c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)
)) AS avg_time_to_delivery,
FROM `e_commerce.orders` AS o
JOIN
`e_commerce.order_items` AS ot
ON o.order_id = ot.order_id
JOIN
`e_commerce.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)
)) DESC
LIMIT 5
),
lowest AS (SELECT
c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)
)) AS avg_time_to_delivery,
FROM `e_commerce.orders` AS o
JOIN
`e_commerce.order_items` AS ot
ON o.order_id = ot.order_id
JOIN
`e_commerce.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)
)) ASC
LIMIT 5)
SELECT
customer_state,
avg_time_to_delivery
FROM (
  SELECT
    customer_state,
    highest.avg_time_to_delivery
  FROM highest
UNION ALL
  SELECT
    customer_state,
    lowest.avg_time_to_delivery
  FROM lowest
)
ORDER BY avg_time_to_delivery DESC ;
```

Query Result:-



Row	customer_state	avg_time_to_delivery
1	AP	28.0
2	RR	28.0
3	AM	26.0
4	AL	24.0
5	PA	23.0
6	RS	15.0
7	DF	13.0
8	MG	12.0
9	PR	11.0
10	SP	8.0

Analysis:- The following table represent the 5 state with highest average time to delivery and lowest average time to delivery respectively, first 5 in the table are highest and next 5 are the lowest states.

So based on this we can conclude that highest no

a) RR(Roraima) and AP(Amapá) has taken the highest average time to delivery order, that can be a reason for its decrease in the number of orders.

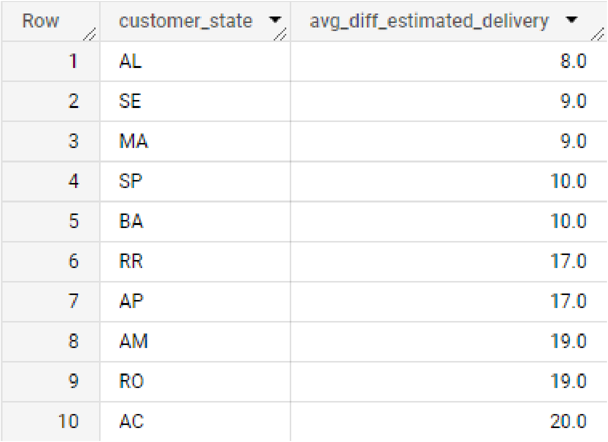
b) SP (São Paulo) has taken lowest average time to delivery which has help in the increase in number of orders.

***Note:- exact name is from internet source**

7) Top 5 states where delivery is really fast/ not so fast compared to estimated date

```
WITH highest AS (SELECT
c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date
,DAY))) AS avg_diff_estimated_delivery,
FROM `e_commerce.orders` AS o
JOIN
`e_commerce.order_items` AS ot
ON o.order_id = ot.order_id
JOIN
`e_commerce.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date
,DAY))) DESC
LIMIT 5
),
lowest AS (SELECT
c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date
,DAY))) AS avg_diff_estimated_delivery,
FROM `e_commerce.orders` AS o
JOIN
`e_commerce.order_items` AS ot
ON o.order_id = ot.order_id
JOIN
`e_commerce.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date
,DAY))) ASC
LIMIT 5)
SELECT
customer_state,
avg_diff_estimated_delivery
FROM (
  SELECT
    customer_state,
    highest.avg_diff_estimated_delivery
  FROM highest
UNION ALL
  SELECT
    customer_state,
    lowest.avg_diff_estimated_delivery
  FROM lowest
)
ORDER BY avg_diff_estimated_delivery ASC;
```

Query Result:-



Row	customer_state	avg_diff_estimated_delivery
1	AL	8.0
2	SE	9.0
3	MA	9.0
4	SP	10.0
5	BA	10.0
6	RR	17.0
7	AP	17.0
8	AM	19.0
9	RO	19.0
10	AC	20.0

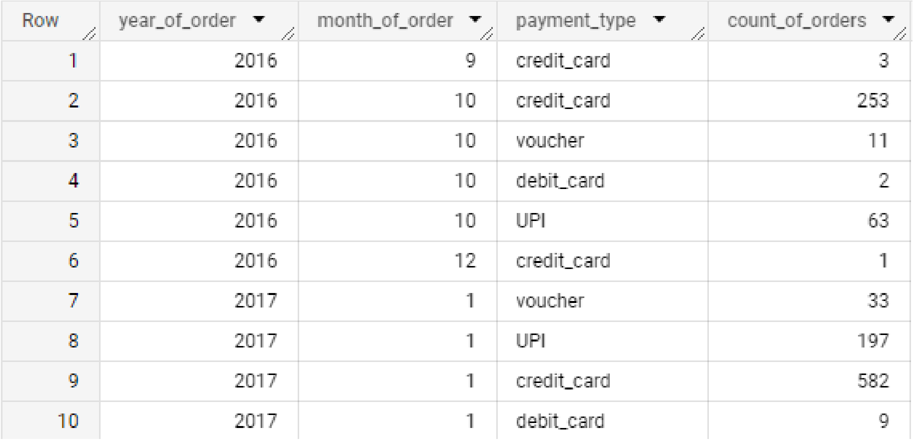
Analysis:- Based on the table generated, we can conclude that states with highest number of order have lowest average difference between the estimated and delivered date and vice versa.

8) Payment type analysis:

1. Month over Month count of orders for different payment types

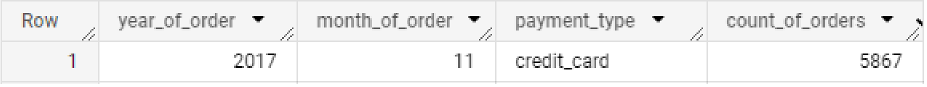
```
SELECT
x.year_of_order,
x.month_of_order,
x.payment_type,
x.count_of_orders
FROM(
SELECT
p.payment_type,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month_of_order,
EXTRACT (YEAR FROM o.order_purchase_timestamp) AS year_of_order,
COUNT (DISTINCT p.order_id) AS count_of_orders
FROM `e_commerce.payments` AS p
JOIN
`e_commerce.orders` AS o
ON
p.order_id = o.order_id
GROUP BY p.payment_type,EXTRACT(MONTH FROM o.order_purchase_timestamp),EXTRACT
(YEAR FROM o.order_purchase_timestamp)
) AS x
ORDER BY x.year_of_order,x.month_of_order;
```

Query Result:-



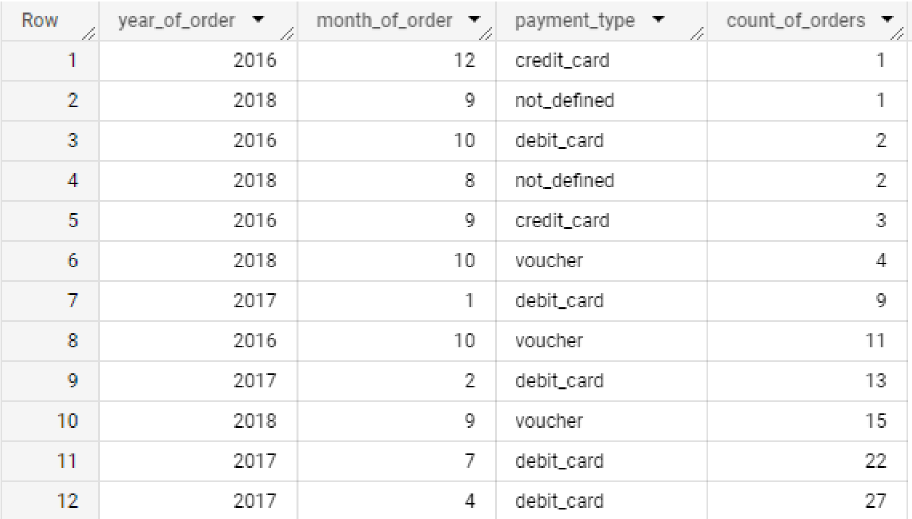
Row	year_of_order	month_of_order	payment_type	count_of_orders
1	2016	9	credit_card	3
2	2016	10	credit_card	253
3	2016	10	voucher	11
4	2016	10	debit_card	2
5	2016	10	UPI	63
6	2016	12	credit_card	1
7	2017	1	voucher	33
8	2017	1	UPI	197
9	2017	1	credit_card	582
10	2017	1	debit_card	9

Analysis:- a) Credit card has been used as the most commonly means of payment with the peak number of order during November of Year 2017.



Row	year_of_order	month_of_order	payment_type	count_of_orders
1	2017	11	credit_card	5867

b) On an average debit cards and vouchers has been used very rarely as a payment type.

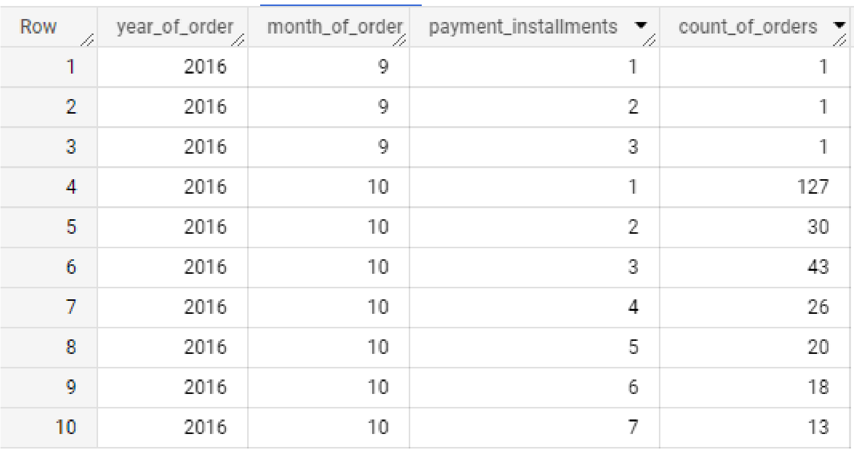


Row	year_of_order	month_of_order	payment_type	count_of_orders
1	2016	12	credit_card	1
2	2018	9	not_defined	1
3	2016	10	debit_card	2
4	2018	8	not_defined	2
5	2016	9	credit_card	3
6	2018	10	voucher	4
7	2017	1	debit_card	9
8	2016	10	voucher	11
9	2017	2	debit_card	13
10	2018	9	voucher	15
11	2017	7	debit_card	22
12	2017	4	debit_card	27

2. Count of orders based on the no. of payment instalments

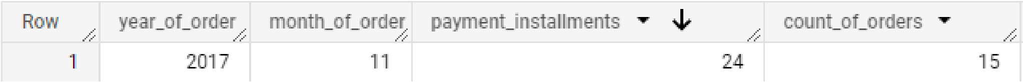
```
SELECT
x.year_of_order,
x.month_of_order,
x.payment_installments,
x.count_of_orders
FROM(
SELECT
p.payment_installments,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month_of_order,
EXTRACT (YEAR FROM o.order_purchase_timestamp) AS year_of_order,
COUNT (DISTINCT p.order_id) AS count_of_orders
FROM `e_commerce.payments` AS p
JOIN
`e_commerce.orders` AS o
ON
p.order_id = o.order_id
GROUP BY p.payment_installments,EXTRACT(MONTH FROM
o.order_purchase_timestamp),EXTRACT (YEAR FROM o.order_purchase_timestamp)
) AS x
ORDER BY x.year_of_order,x.month_of_order;
```

Query Result:-



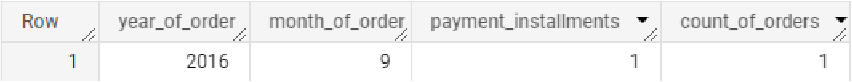
Row	year_of_order	month_of_order	payment_installments	count_of_orders
1	2016	9	1	1
2	2016	9	2	1
3	2016	9	3	1
4	2016	10	1	127
5	2016	10	2	30
6	2016	10	3	43
7	2016	10	4	26
8	2016	10	5	20
9	2016	10	6	18
10	2016	10	7	13

Analysis:- a) Highest number of instalment was lasted around 24 months and was during the month of November of the year 2017.



Row	year_of_order	month_of_order	payment_installments	count_of_orders
1	2017	11	24	15

b)Lowest number of instalment was lasted for 0 months and was during the month of April of the year 2018.



Row	year_of_order	month_of_order	payment_installments	count_of_orders
1	2016	9	1	1