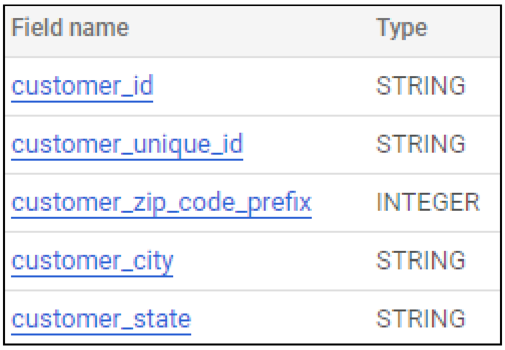
Q1: Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset.

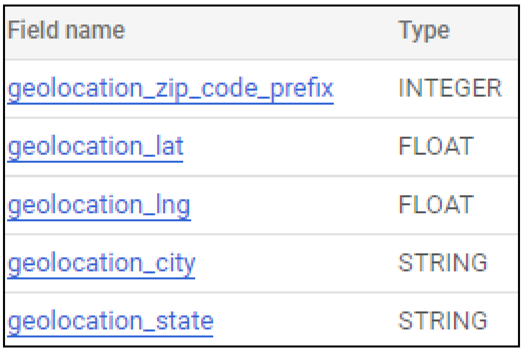
1a: Data type of columns in a table

Customers Table:



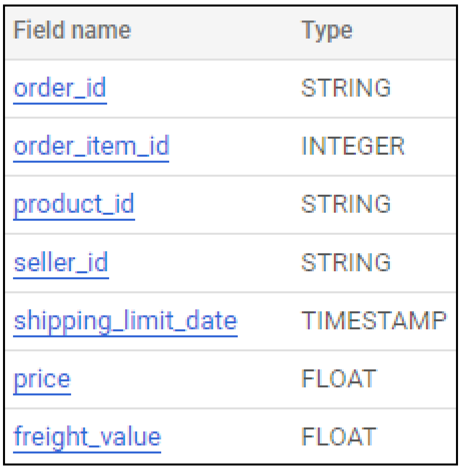
Field name	Type
customer_id	STRING
customer_unique_id	STRING
customer_zip_code_prefix	INTEGER
customer_city	STRING
customer_state	STRING

Geolocation Table:



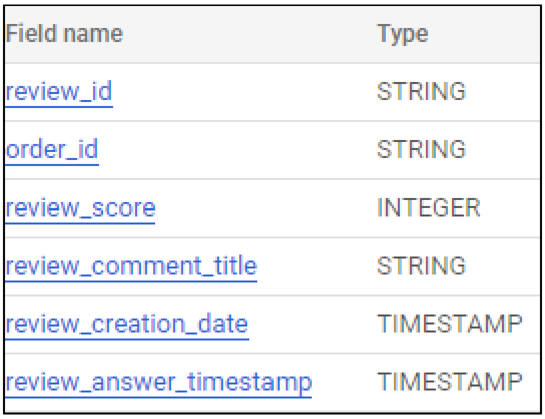
Field name	Type
geolocation_zip_code_prefix	INTEGER
geolocation_lat	FLOAT
geolocation_lng	FLOAT
geolocation_city	STRING
geolocation_state	STRING

Orders_item Table:



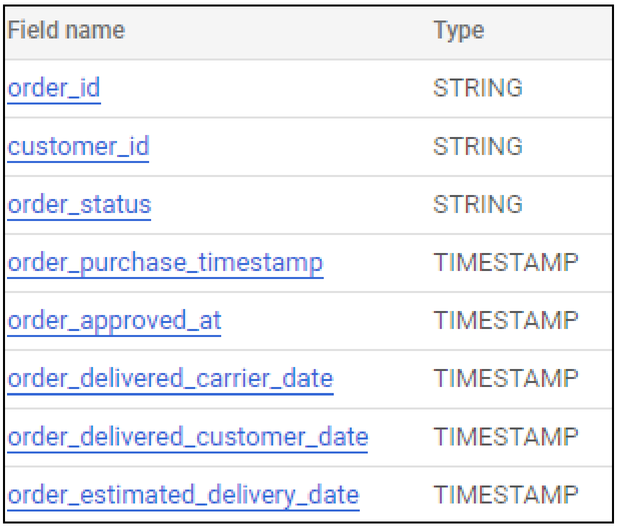
Field name	Type
order_id	STRING
order_item_id	INTEGER
product_id	STRING
seller_id	STRING
shipping_limit_date	TIMESTAMP
price	FLOAT
freight_value	FLOAT

Orders_review Table:



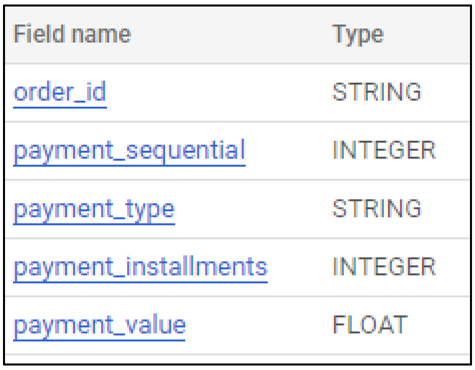
Field name	Type
review_id	STRING
order_id	STRING
review_score	INTEGER
review_comment_title	STRING
review_creation_date	TIMESTAMP
review_answer_timestamp	TIMESTAMP

Orders Table:



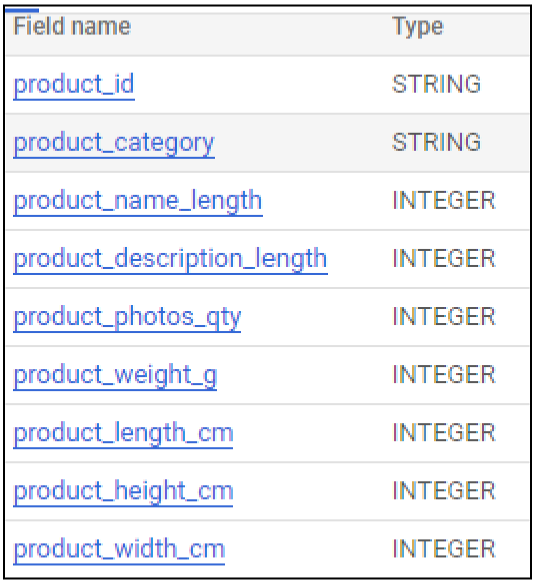
Field name	Type
order_id	STRING
customer_id	STRING
order_status	STRING
order_purchase_timestamp	TIMESTAMP
order_approved_at	TIMESTAMP
order_delivered_carrier_date	TIMESTAMP
order_delivered_customer_date	TIMESTAMP
order_estimated_delivery_date	TIMESTAMP

Payments Table:



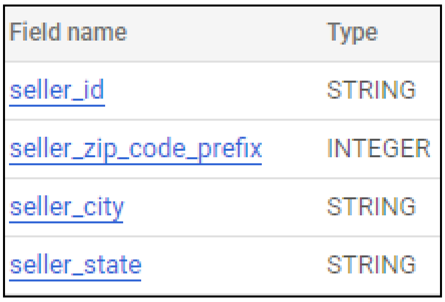
Field name	Type
order_id	STRING
payment_sequential	INTEGER
payment_type	STRING
payment_installments	INTEGER
payment_value	FLOAT

Products Table:



Field name	Type
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

Sellers Table:



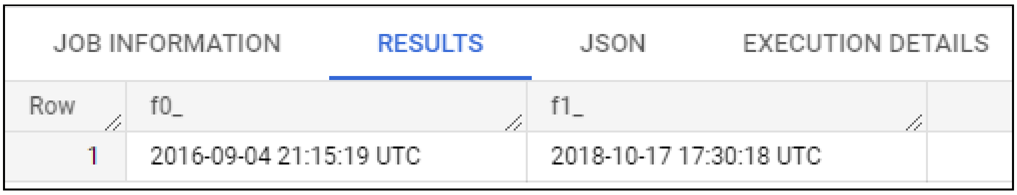
Field name	Type
seller_id	STRING
seller_zip_code_prefix	INTEGER
seller_city	STRING
seller_state	STRING

Q1b: Time period for which the data is given.

Query:

```
SELECT MIN(order_purchase_timestamp), MAX(order_purchase_timestamp) FROM target_sql.orders
```

Result:



Row	f0_	f1_
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

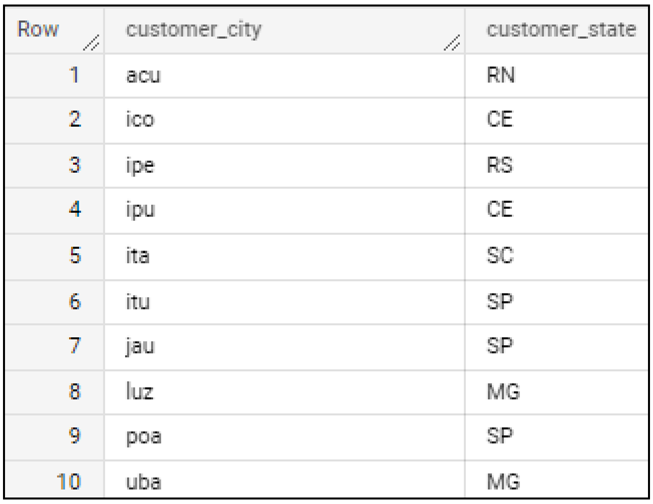
Q1c: Cities and States of customers ordered during the given period.

Query:

```
SELECT DISTINCT(c.customer_city), c.customer_state from `target_sql.customers` c

INNER JOIN `target_sql.orders` o on o.customer_id = c.customer_id
```

Result:



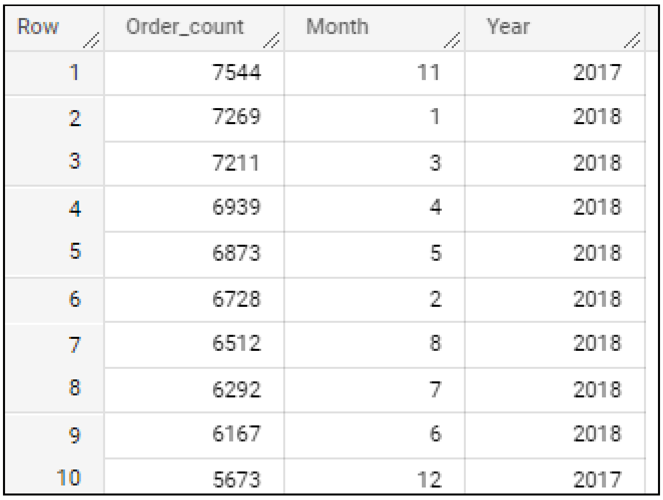
Row	customer_city	customer_state
1	acu	RN
2	ico	CE
3	ipe	RS
4	ipu	CE
5	ita	SC
6	itu	SP
7	jau	SP
8	luz	MG
9	poa	SP
10	uba	MG

Q2a: 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?

Query:

```
SELECT count(order_id) AS Order_count, EXTRACT(MONTH FROM order_purchase_timestamp) AS Mont
h, EXTRACT(YEAR FROM order_purchase_timestamp) AS Year
from `target_sql.orders`
group by Month,Year
ORDER BY Order_count desc;
```

Result:



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

Actionable Insights: We can infer that No. of orders peak at the end and start of the year, especially in November 2017 and January 2018.

Recommendations: The no. of orders are less in the month for September. We can suggest to run special campaigns or offer various discounts to increase the sales.

Q2b: What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

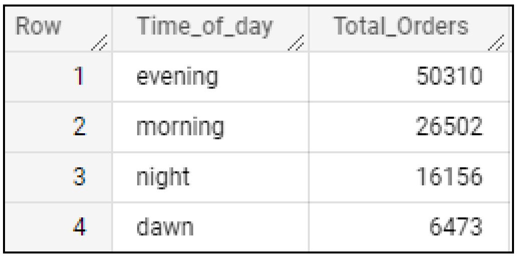
Query:

```
SELECT CASE
        WHEN hour >= 0  AND hour <= 7  THEN 'dawn'
        WHEN hour >  7  AND hour <= 12 THEN 'morning'
        WHEN hour > 12  AND hour <= 20 THEN 'evening'
        WHEN hour > 20 THEN 'night'
       END AS Time_of_day
     , count(*) AS Total_Orders

  from (
    SELECT EXTRACT(hour from order_purchase_timestamp) as hour
    from `target_sql.orders`
  ) t

  group by 1
  order by count(*) desc;
```

Result:



Row	Time_of_day	Total_Orders
1	evening	50310
2	morning	26502
3	night	16156
4	dawn	6473

Actionable Insights: We can see that most of the orders are placed in evening which is expected. Many of the orders are also placed in 'dawn' which is also a good thing for the business.

Recommendations: To increase the sales in morning and night the company should provide some additional discounts/schemes for morning/night so that number of orders also gets increased during this time of the day.

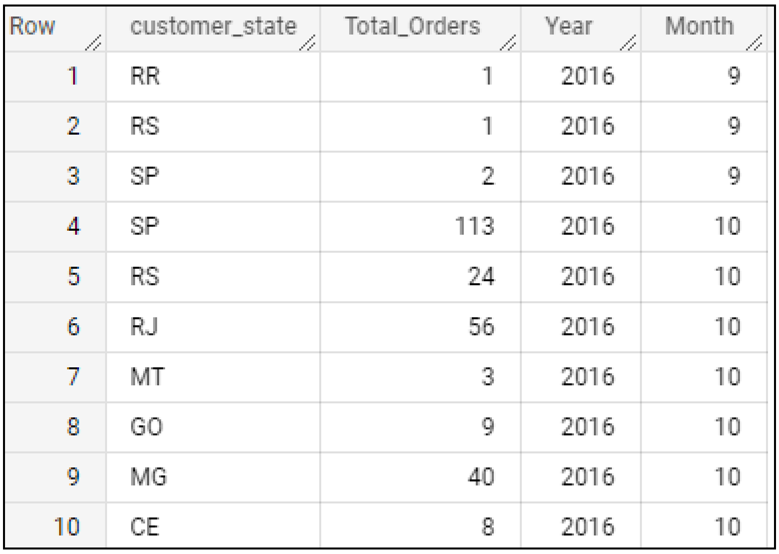
Q3a: Evolution of E-commerce orders in the Brazil region: Get month on month orders by states.

Query:

```
SELECT c.customer_state, COUNT(o.order_id) as Total_Orders,
EXTRACT(year from o.order_purchase_timestamp) AS Year,
EXTRACT(month from o.order_purchase_timestamp) AS Month from `target_sql.orders` o
```

```
left join `target_sql.customers` c on c.customer_id = o.customer_id
GROUP BY c.customer_state,Year,Month
ORDER BY Year, Month
```

Result:



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

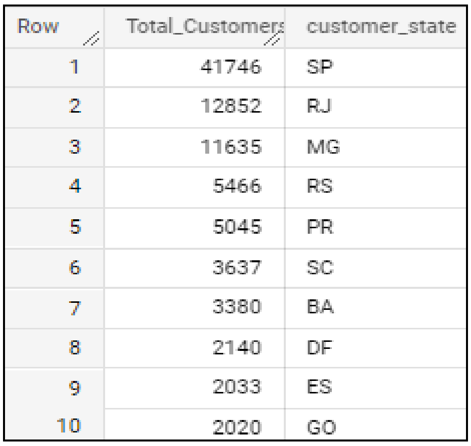
Actionable Insights: If we compare the no. of orders for a particular month in the year 2016 with that of 2017 or 2018 than we can infer that the Number of orders are drastically increased in almost every state which can be considered a very good business in terms of time period.

Recommendations: We should find a common pattern for all the years i.e., need to figure out for which months the sales are low for each year. For that particular month we can roll out special offers or discounts to increase the sales.

Q3b: Distribution of customers across the states in Brazil

```
Query:
SELECT COUNT(customer_unique_id) as Total_Customers, customer_state from `target_sql.custom
ers`
group by customer_state
order by Total_Customers desc;

Result:
```



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

Actionable Insights: Top 10 states are present in the table above which draws the highest customers. For state which are having less customers, analysis must be done to find out the reason behind this.

Q4a: 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.

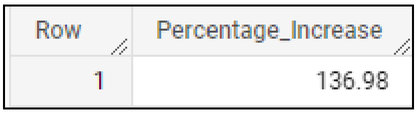
Query:

```
WITH temp1 as
(SELECT SUM(p.payment_value) as total_payment_2017 from `farmers_market.payments` p
  LEFT JOIN `target_sql.orders` o on o.order_id = p.order_id
  where EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2017 AND EXTRACT(MONTH FROM o.order
_purchase_timestamp) IN (1,2,3,4,5,6,7,8)
),

temp2 as(
 SELECT SUM(p.payment_value) as total_payment_2018 from `farmers_market.payments` p
  LEFT JOIN `target_sql.orders` o on o.order_id = p.order_id
  where EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2018 AND EXTRACT(MONTH FROM o.order
_purchase_timestamp) IN (1,2,3,4,5,6,7,8)
)

SELECT ROUND((total_payment_2018-
total_payment_2017)/total_payment_2017 * 100,2) as Percentage_Increase  from temp1,temp2
```

Result:



Row	Percentage_Increase
1	136.98

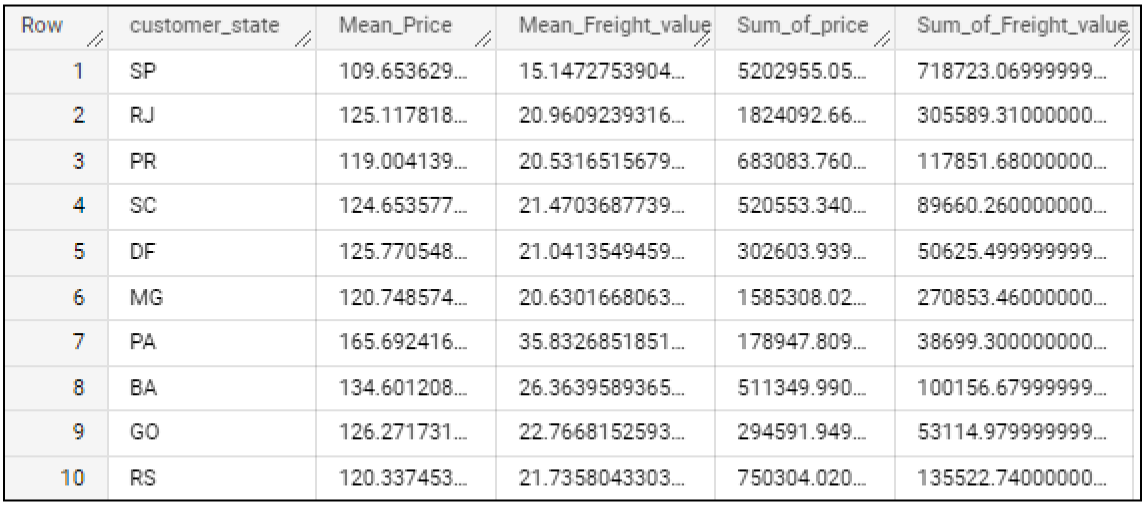
Actionable Insight: The percentage increase in cost of orders from 2017 to 2018 is very high which can be considered a good sign for the business.

Q4b: Mean & Sum of price and freight value by customer state.

Query:

```
SELECT c.customer_state, AVG(oi.price) as Mean_Price, AVG(oi.freight_value) as Mean_Freight
_value, SUM(oi.price) as Sum_of_price, SUM(oi.freight_value) as Sum_of_Freight_value
from `target_sql.order_items` oi left join `target_sql.orders` o on o.order_id = oi.order_i
d
LEFT JOIN `target_sql.customers` c on c.customer_id = o.customer_id
group by c.customer_state
```

Result:



Row	customer_state	Mean_Price	Mean_Freight_value	Sum_of_price	Sum_of_Freight_value
1	SP	109.653629...	15.1472753904...	5202955.05...	718723.06999999...
2	RJ	125.117818...	20.9609239316...	1824092.66...	305589.31000000...
3	PR	119.004139...	20.5316515679...	683083.760...	117851.68000000...
4	SC	124.653577...	21.4703687739...	520553.340...	89660.260000000...
5	DF	125.770548...	21.0413549459...	302603.939...	50625.499999999...
6	MG	120.748574...	20.6301668063...	1585308.02...	270853.46000000...
7	PA	165.692416...	35.8326851851...	178947.809...	38699.300000000...
8	BA	134.601208...	26.3639589365...	511349.990...	100156.67999999...
9	GO	126.271731...	22.7668152593...	294591.949...	53114.979999999...
10	RS	120.337453...	21.7358043303...	750304.020...	135522.74000000...

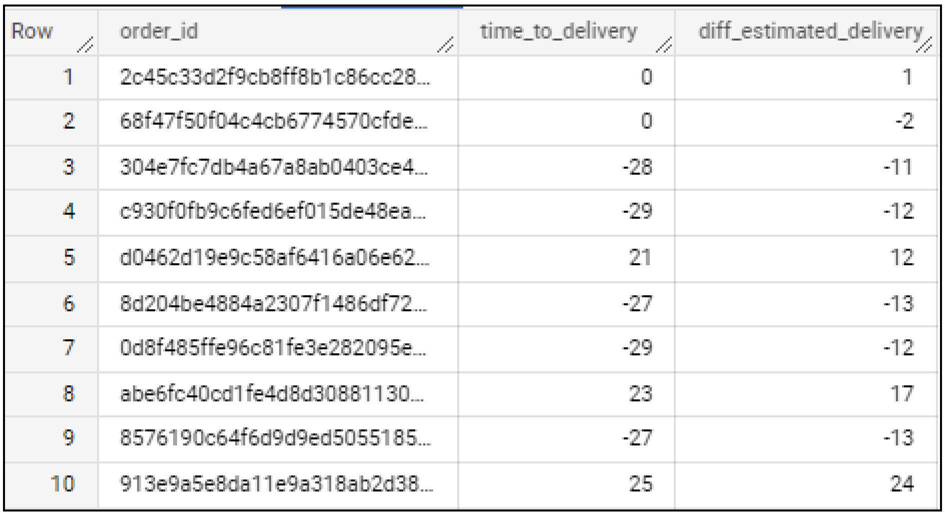
<u>Actionable Insights:</u> We can infer from the above result is that the Freight value increases gradually when the mean price of the order is increased.

Q5a. Analysis on sales, freight and delivery time - Calculate days between purchasing, delivering and estimated delivery.

Query:

```
SELECT order_id, EXTRACT(Date FROM order_delivered_customer_date) - EXTRACT(Date FROM order
_purchase_timestamp) as time_to_delivery,
EXTRACT(Date FROM order_delivered_customer_date) - EXTRACT(Date FROM order_estimated_delive
ry_date)  as diff_estimated_delivery
from `target_sql.orders`
```

Result:



Row	order_id	time_to_delivery	diff_estimated_delivery
1	2c45c33d2f9cb8ff8b1c86cc28...	0	1
2	68f47f50f04c4cb6774570cfde...	0	-2
3	304e7fc7db4a67a8ab0403ce4...	-28	-11
4	c930f0fb9c6fed6ef015de48ea...	-29	-12
5	d0462d19e9c58af6416a06e62...	21	12
6	8d204be4884a2307f1486df72...	-27	-13
7	0d8f485ffe96c81fe3e282095e...	-29	-12
8	abe6fc40cd1fe4d8d30881130...	23	17
9	8576190c64f6d9d9ed5055185...	-27	-13
10	913e9a5e8da11e9a318ab2d38...	25	24

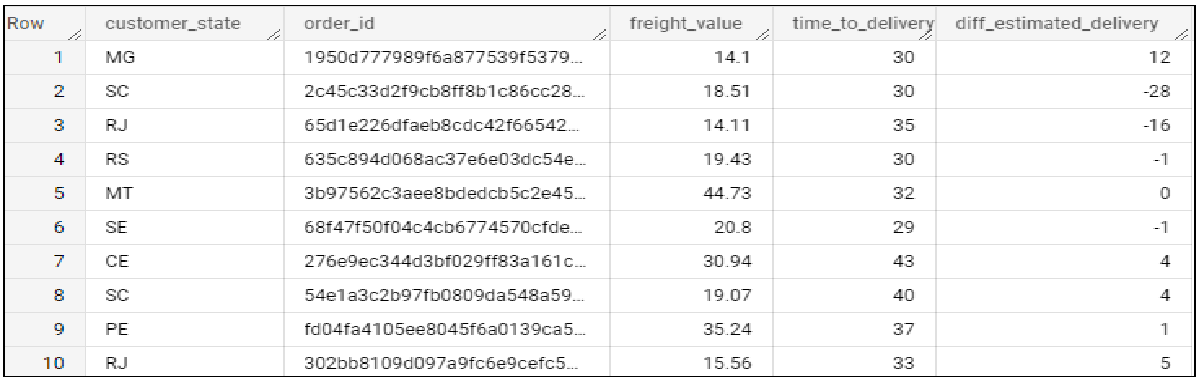
<u>Actionable Insights</u>: For maximum orders the time of delivery was greater than estimated delivery which is a concern for the company as it may lead to loss of **repeated** customers.

Q5c - Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery.

Query:

```
SELECT c.customer_state, o.order_id, AVG(or1.freight_value) as freight_value, EXTRACT(day F
ROM order_delivered_customer_date) - EXTRACT(day FROM order_purchase_timestamp) as time_to_
delivery,
EXTRACT(day FROM order_delivered_customer_date) - EXTRACT(day FROM order_estimated_delivery
_date)  as diff_estimated_delivery
from target_sql.orders o left join target_sql.customers c on c.customer_id = o.customer_id
left join target_sql.order_items or1 on or1.order_id = o.order_id
group by c.customer_state, o.order_id,time_to_delivery,diff_estimated_delivery
```

Result:



Row	customer_state	order_id	freight_value	time_to_delivery	diff_estimated_delivery
1	MG	1950d777989f6a877539f5379...	14.1	30	12
2	SC	2c45c33d2f9cb8ff8b1c86cc28...	18.51	30	-28
3	RJ	65d1e226dfaeb8cdc42f66542...	14.11	35	-16
4	RS	635c894d068ac37e6e03dc54e...	19.43	30	-1
5	MT	3b97562c3aee8bdedcb5c2e45...	44.73	32	0
6	SE	68f47f50f04c4cb6774570cfde...	20.8	29	-1
7	CE	276e9ec344d3bf029ff83a161c...	30.94	43	4
8	SC	54e1a3c2b97fb0809da548a59...	19.07	40	4
9	PE	fd04fa4105ee8045f6a0139ca5...	35.24	37	1
10	RJ	302bb8109d097a9fc6e9cefc5...	15.56	33	5

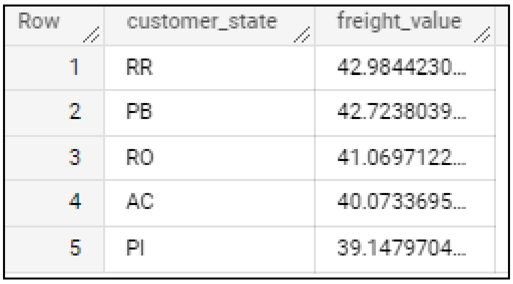
<u>Recommendations</u>: In the states were time of actual delivery is greater than estimated delivery, company must try to setup a unit/warehouse were the products ordered frequently are stocked beforehand. This can reduce the delivery time of the products.

Q5d - Top 5 states with highest average freight value - sort in desc- limit 5.

Query:

```
SELECT c.customer_state,AVG(or1.freight_value) as freight_value
from target_sql.orders o left join target_sql.customers c on c.customer_id = o.customer_id
left join target_sql.order_items or1 on or1.order_id = o.order_id
group by c.customer_state
order by freight_value desc
limit 5;
```

```
Result: Below are the states with highest Average freight value
```



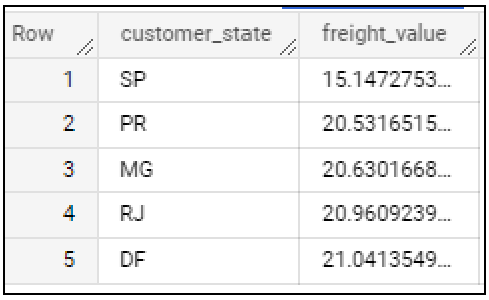
Row	customer_state	freight_value
1	RR	42.9844230...
2	PB	42.7238039...
3	RO	41.0697122...
4	AC	40.0733695...
5	PI	39.1479704...

Q5d - Top 5 states with lowest average freight value - sort in asc - limit 5.

Query:

```
SELECT c.customer_state, EXTRACT(day FROM order_delivered_customer_date) - EXTRACT(day FROM
 order_purchase_timestamp) as time_to_delivery
from target_sql.orders o left join target_sql.customers c on c.customer_id = o.customer_id
group by c.customer_state,time_to_delivery
order by time_to_delivery desc
LIMIT 5;
```

Result: Below are the states with lowest Average freight value



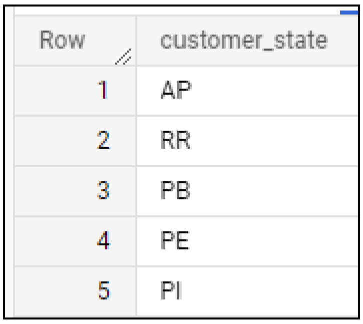
Row	customer_state	freight_value
1	SP	15.1472753…
2	PR	20.5316515…
3	MG	20.6301668…
4	RJ	20.9609239…
5	DF	21.0413549…

Q5e - Top 5 states with highest average time to delivery.

Query:

```
SELECT c.customer_state, AVG(EXTRACT(day FROM order_delivered_customer_date) - EXTRACT(day
FROM order_purchase_timestamp)) as time_to_delivery
from target_sql.orders o left join target_sql.customers c on c.customer_id = o.customer_id
left join target_sql.order_items or1 on or1.order_id = o.order_id
group by c.customer_state
order by time_to_delivery desc
LIMIT 5;
```

Result:



Row	customer_state
1	AP
2	RR
3	PB
4	PE
5	PI

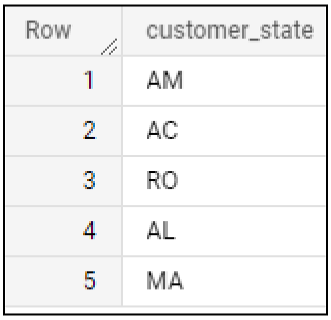
Q5e - Top 5 states with lowest average time to delivery.

Query:

```
SELECT c.customer_state, AVG(EXTRACT(day FROM order_delivered_customer_date) - EXTRACT(day
FROM order_purchase_timestamp)) as time_to_delivery
from target_sql.orders o left join target_sql.customers c on c.customer_id = o.customer_id
left join target_sql.order_items or1 on or1.order_id = o.order_id
```

```
group by c.customer_state
order by time_to_delivery asc
LIMIT 5;
```

Result:



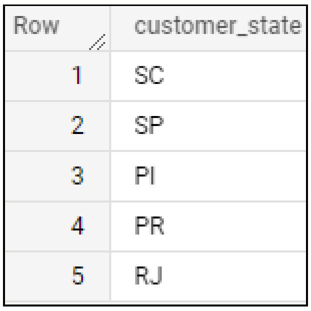
Row	customer_state
1	AM
2	AC
3	RO
4	AL
5	MA

Q5f: Top 5 states where delivery is really not so fast compared to estimated date.

Query:

```
SELECT customer_state from (select c.customer_state, EXTRACT(day from o.order_delivered_customer_date) - EXTRACT(day from
o.order_estimated_delivery_date) as Delivery_days from `target_sql.orders` o left join
`target_sql.customers` c on c.customer_id = o.customer_id)t
GROUP BY customer_state,Delivery_days
ORDER BY Delivery_days desc
limit 5;
```

Result:



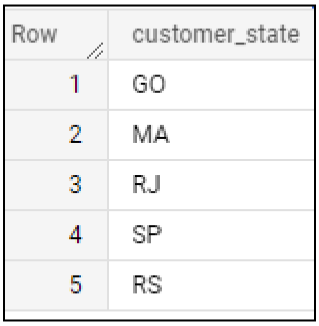
Row	customer_state
1	SC
2	SP
3	PI
4	PR
5	RJ

Q5f: Top 5 states where delivery is really fast compared to estimated date.

Query:

```
SELECT customer_state from (select c.customer_state, EXTRACT(day from o.order_delivered_customer_date) - EXTRACT(day from
o.order_estimated_delivery_date) as Delivery_days from `target_sql.orders` o left join
`target_sql.customers` c on c.customer_id = o.customer_id)t
where t.Delivery_days IS NOT NULL
GROUP BY customer_state,Delivery_days
ORDER BY Delivery_days asc
limit 5;
```

Result:



Row	customer_state
1	GO
2	MA
3	RJ
4	SP
5	RS

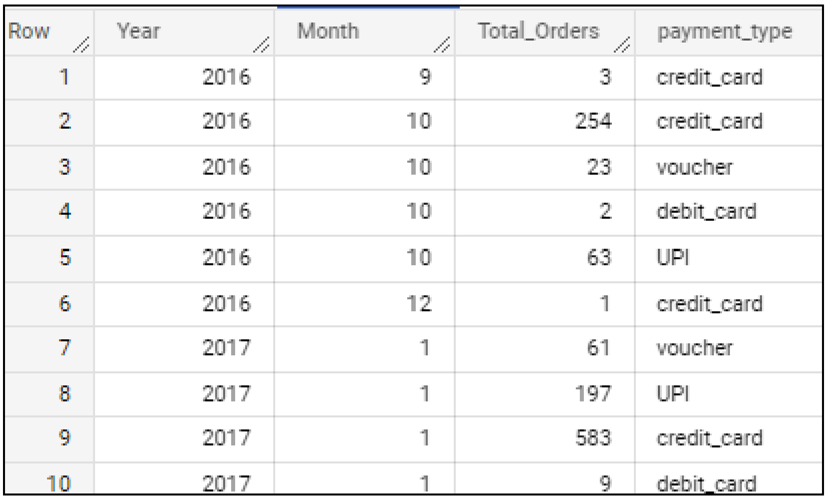
Recommendations: In the above states were delivery is taking more time than the expected time, company should hire more delivery agents to speed up the process of delivery.

Q6a: Month over Month count of orders for different payment types.

Query:

```
select EXTRACT(year from o.order_purchase_timestamp) as Year,EXTRACT(month from o.order_pur
chase_timestamp) as Month, COUNT(o.order_id) as Total_Orders, p.payment_type from
`target_sql.payments` p left join `target_sql.orders` o on o.order_id = p.order_id
group by Year,Month,p.payment_type
order by Year,Month
```

Result:



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

Actionable Insights: From the overall result, most of the payments are being done cashless i.e., through credit card or UPI. This indicates people prefer to trade cashless as there is some element of risk involved in dealing with cash.

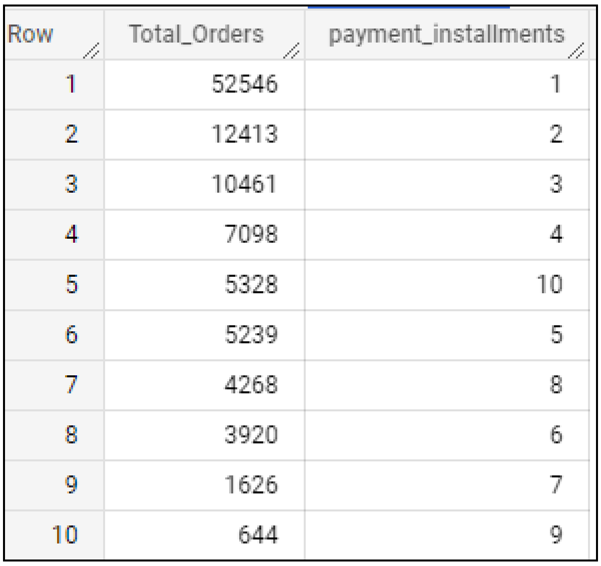
Q6b: Count of orders based on the no. of payment instalments.

Query:

```
select COUNT(o.order_id) as Total_Orders, p.payment_installments from
```

```
`target_sql.payments` p left join `target_sql.orders` o on o.order_id = p.order_id
group by p.payment_installments
order by Total_Orders desc
```

Result:



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

Actionable Insights: Most of the payments are done in only 1 instalment which is good thing for the growth of the business as there is a regular cash flow.

Recommendations: For the orders that have more than 5 instalments we can ask the vendor/customer to decrease the instalments for the betterment of the company revenue month-wise/quarter-wise and also cash flow.

Overall Actionable Insight from the results incurred:

We can see that the total orders has increased in 2018 compared to 2017. The mode of payment is mostly done cashless in this digital world. The peak time of the day to day business is Evening followed by Morning and the payment is mostly done in either 1, 2 or 3 instalments.

Overall Recommendations from the results incurred:

Sometimes the delivery of the product is exceeding the estimated delivery time which might affect the sales. In order to solve this problem company should try to setup a warehouse in that state were the number of orders receiving are on the higher side. By doing so and adding some extra delivery agents to the workforce might solve this problem. Also some states were lagging in overall performance of sales. In such states all the factors must be taken into consideration as to why it is lagging in terms of sales, repetitive business etc.