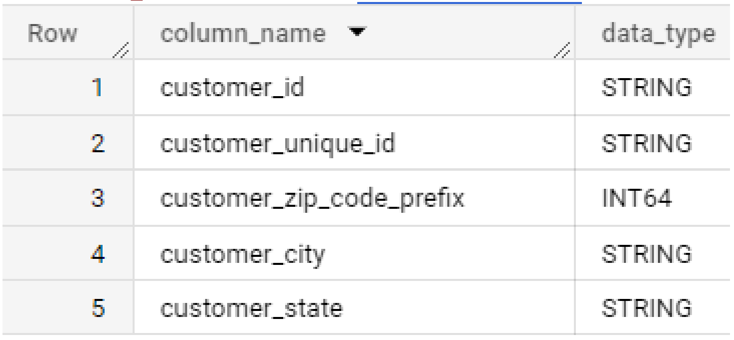
1A) Data type of all columns in the "customers" tab

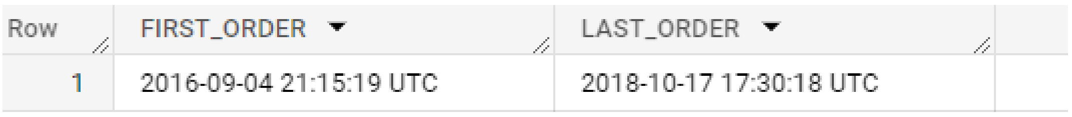
```
SELECT column_name, data_type
FROM `active-axle-390010.TARGET_SQL.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers'
```



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

1B) Get the time range between which the orders were placed.

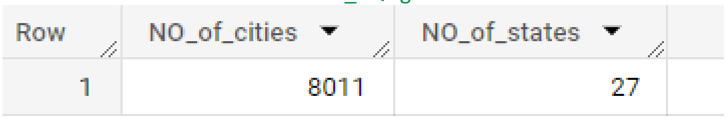
```
SELECT MIN(o.order_purchase_timestamp) AS FIRST_ORDER, MAX(o.order_purchase_timestamp) AS LAST_ORDER
FROM `active-axle-390010.TARGET_SQL.orders` AS o
```



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

1C) Count the number of Cities and States in our dataset.

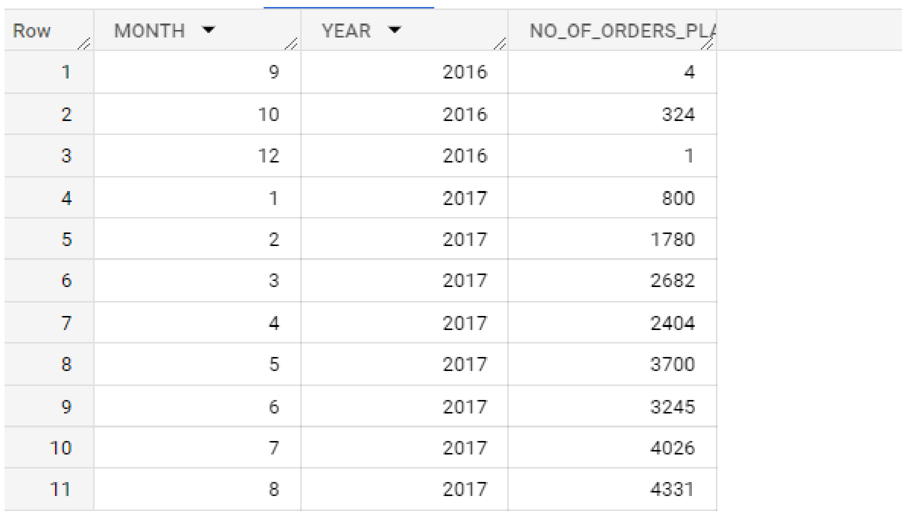
```
SELECT   COUNT(DISTINCT geolocation_city) AS NO_of_cities,
COUNT(DISTINCT geolocation_state) AS NO_of_states
FROM `active-axle-390010.TARGET_SQL.geolocation`
```



Row	NO_of_cities	NO_of_states
1	8011	27

2A) Is there a growing trend in the no. of orders placed over the past years?

```
SELECT EXTRACT(MONTH FROM o.order_purchase_timestamp)AS MONTH, EXTRACT(YEAR FROM
o.order_purchase_timestamp) AS YEAR, COUNT(o.order_id) AS NO_OF_ORDERS_PLACED
FROM `active-axle-390010.TARGET_SQL.orders` AS o
GROUP BY 2,1
ORDER BY 2,1
```



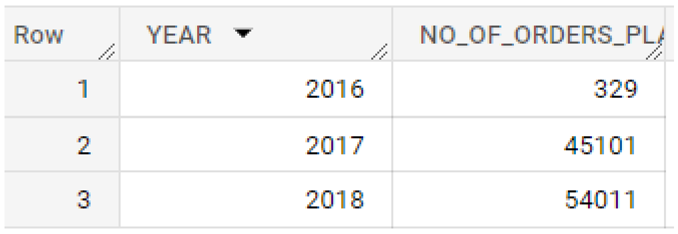
Row	MONTH	YEAR	NO_OF_ORDERS_PL
1	9	2016	4
2	10	2016	324
3	12	2016	1
4	1	2017	800
5	2	2017	1780
6	3	2017	2682
7	4	2017	2404
8	5	2017	3700
9	6	2017	3245
10	7	2017	4026
11	8	2017	4331

Insight:- The number of orders placed increase as the year increases

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

Part 1

```
SELECT  EXTRACT(YEAR FROM o.order_purchase_timestamp) AS YEAR, COUNT(o.order_id) AS NO_OF_ORDERS_PLACED
FROM `active-axle-390010.TARGET_SQL.orders` AS o
GROUP BY 1
ORDER BY 1
```

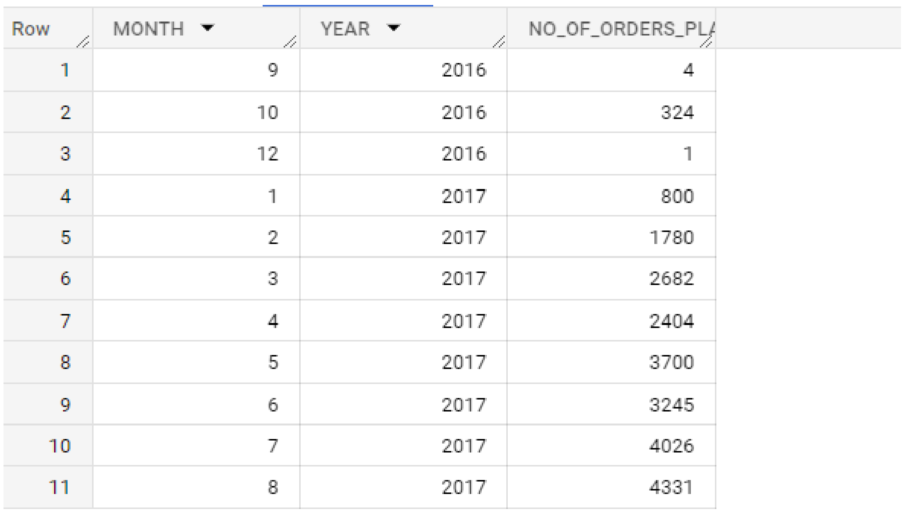


Row	YEAR	NO_OF_ORDERS_PL
1	2016	329
2	2017	45101
3	2018	54011

Insight:- The number of orders placed increase as the year increases

Part 2

```
SELECT EXTRACT(MONTH FROM o.order_purchase_timestamp)AS MONTH, EXTRACT(YEAR FROM
o.order_purchase_timestamp) AS YEAR, COUNT(o.order_id) AS NO_OF_ORDERS_PLACED
FROM `active-axle-390010.TARGET_SQL.orders` AS o
GROUP BY 2,1
ORDER BY 2,1
```

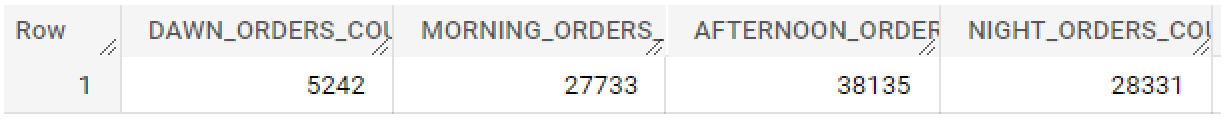


Row	MONTH	YEAR	NO_OF_ORDERS_PL
1	9	2016	4
2	10	2016	324
3	12	2016	1
4	1	2017	800
5	2	2017	1780
6	3	2017	2682
7	4	2017	2404
8	5	2017	3700
9	6	2017	3245
10	7	2017	4026
11	8	2017	4331

Insight: Number of orders placed increase as the month increases in a year and starts to fall at the end of the year after September

2C) 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

```
SELECT SUM(e.Dawn) AS DAWN_ORDERS_COUNT,
SUM(e.Morning) AS MORNING_ORDERS_COUNT,
SUM(e.Afternoon) AS AFTERNOON_ORDERS_COUNT,
SUM(e.Night) AS NIGHT_ORDERS_COUNT FROM (SELECT
CASE WHEN EXTRACT (HOUR FROM order_purchase_timestamp) BETWEEN 0 AND 6 THEN 1 END AS Dawn,
CASE WHEN EXTRACT (HOUR FROM order_purchase_timestamp) BETWEEN 7 AND 12 THEN 1 END AS Morning,
CASE WHEN EXTRACT (HOUR FROM order_purchase_timestamp) BETWEEN 13 AND 18 THEN 1 END AS Afternoon,
CASE WHEN EXTRACT (HOUR FROM order_purchase_timestamp) BETWEEN 19 AND 24 THEN 1 END AS Night
FROM `active-axle-390010.TARGET_SQL.orders` ) AS e
```

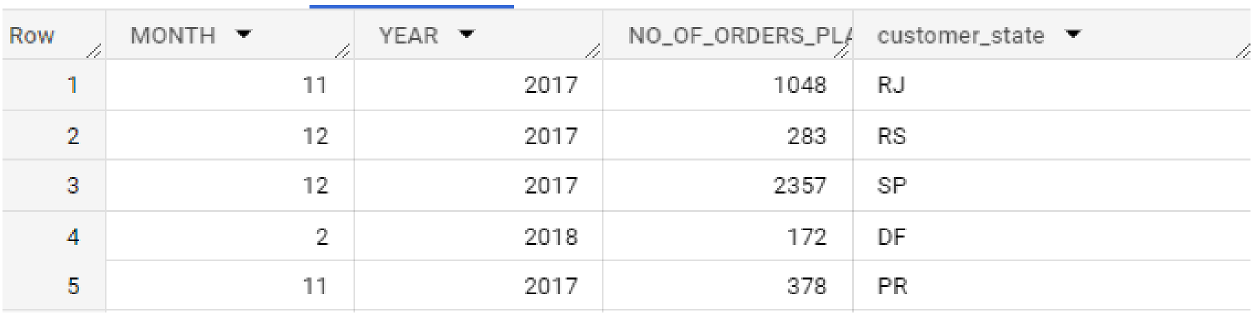


Row	DAWN_ORDERS_CO	MORNING_ORDERS_	AFTERNOON_ORDER	NIGHT_ORDERS_CO
1	5242	27733	38135	28331

Insight:- From above we can determine that mostly Brazilians like to shop between 13-18 hrs which is afternoon

3A) Get the month on month no. of orders placed in each state.

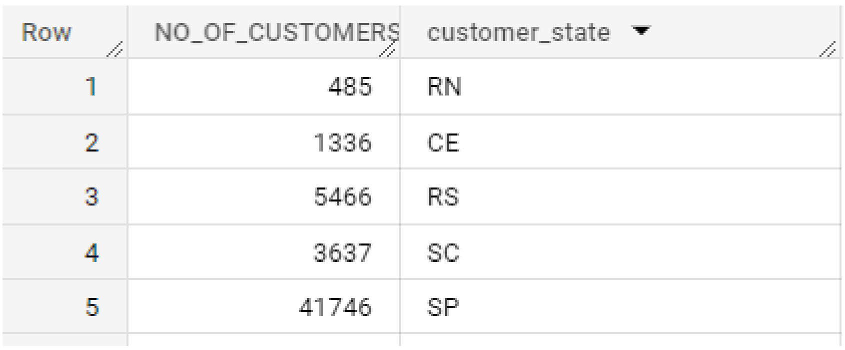
```
SELECT EXTRACT (MONTH FROM o.order_purchase_timestamp)AS MONTH, EXTRACT (YEAR FROM
o.order_purchase_timestamp)AS YEAR, COUNT(o.order_id) AS NO_OF_ORDERS_PLACED, c.customer_state
FROM  `active-axle-390010.TARGET_SQL.orders` AS o
JOIN  `active-axle-390010.TARGET_SQL.customers` AS c
ON o.customer_id=c.customer_id
GROUP BY 1,2,4
```



Row	MONTH	YEAR	NO_OF_ORDERS_PL	customer_state
1	11	2017	1048	RJ
2	12	2017	283	RS
3	12	2017	2357	SP
4	2	2018	172	DF
5	11	2017	378	PR

3B) How are the customers distributed across all the states?

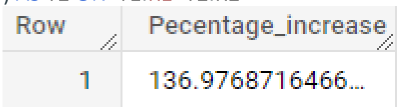
```
SELECT COUNT(c.customer_unique_id ) AS NO_OF_CUSTOMERS, c.customer_state
FROM `active-axle-390010.TARGET_SQL.customers` AS c
GROUP BY c.customer_state
```



Row	NO_OF_CUSTOMERS	customer_state
1	485	RN
2	1336	CE
3	5466	RS
4	3637	SC
5	41746	SP

4A) Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

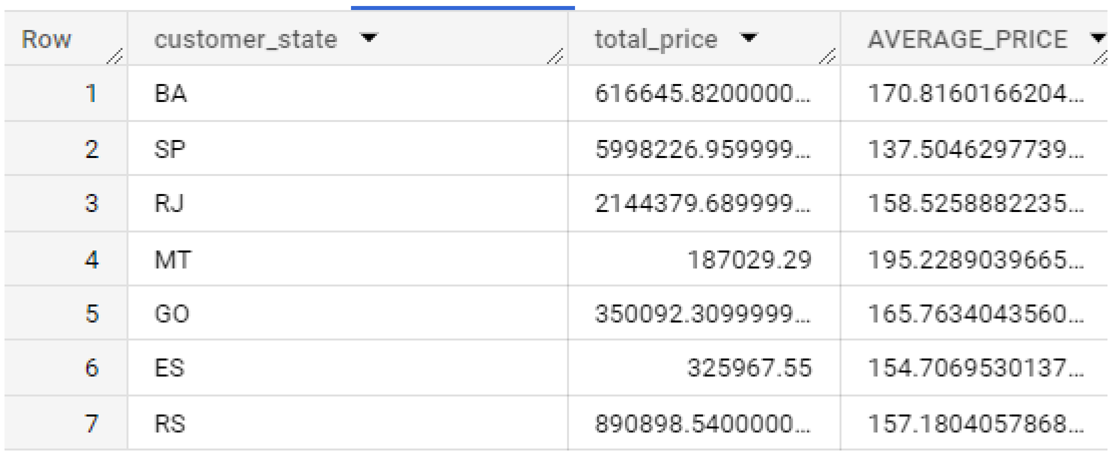
```
SELECT  ((SUM_2018-SUM_2017)/ SUM_2017)*100 AS Pecentage_increase FROM
(SELECT SUM(p.payment_value) AS SUM_2017, "1" AS R1 FROM `active-axle-390010.TARGET_SQL.payments` AS p
JOIN `active-axle-390010.TARGET_SQL.orders` AS o ON p.order_id=o.order_id
WHERE EXTRACT(YEAR FROM o.order_purchase_timestamp) =2017 AND EXTRACT(MONTH FROM
o.order_purchase_timestamp) BETWEEN 1 AND 8) AS t1
JOIN(
SELECT SUM(p.payment_value) AS SUM_2018, "1" AS R2 FROM `active-axle-390010.TARGET_SQL.payments` AS p
JOIN `active-axle-390010.TARGET_SQL.orders` AS o ON p.order_id=o.order_id
WHERE EXTRACT(YEAR FROM o.order_purchase_timestamp) =2018 AND EXTRACT(MONTH FROM
o.order_purchase_timestamp) BETWEEN 1 AND 8
) AS T2 ON  T1.R1=T2.R2
```



Row	Pecentage_increase
1	136.9768716466...

4B) . Calculate the Total & Average value of order price for each state

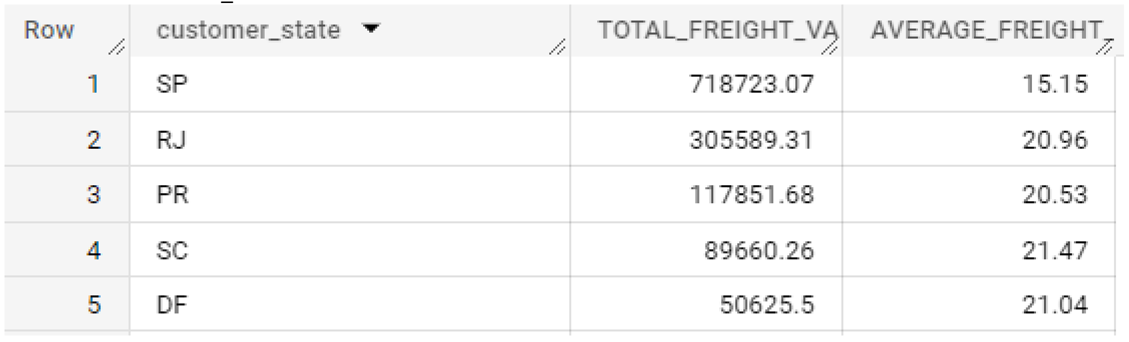
```
SELECT c.customer_state,SUM(p.payment_value) AS total_price, AVG(p.payment_value) AS AVERAGE_PRICE
FROM  `active-axle-390010.TARGET_SQL.payments` AS p
JOIN `active-axle-390010.TARGET_SQL.orders` AS o ON p.order_id=o.order_id
JOIN `active-axle-390010.TARGET_SQL.customers` AS c ON o.customer_id=c.customer_id
GROUP BY c.customer_state
```



Row	customer_state	total_price	AVERAGE_PRICE
1	BA	616645.8200000...	170.8160166204...
2	SP	5998226.959999...	137.5046297739...
3	RJ	2144379.689999...	158.5258882235...
4	MT	187029.29	195.2289039665...
5	GO	350092.3099999...	165.7634043560...
6	ES	325967.55	154.7069530137...
7	RS	890898.5400000...	157.1804057868...

4C) Calculate the Total & Average value of order freight for each state.

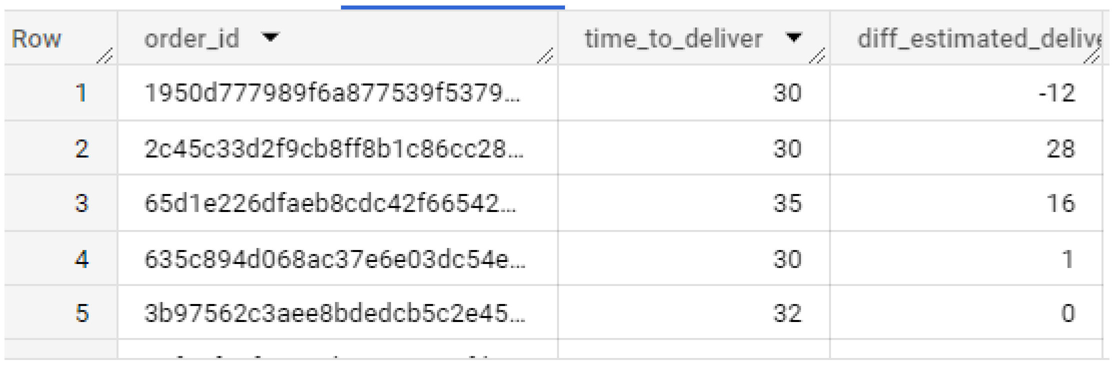
```
SELECT c.customer_state,ROUND(SUM(oi.freight_value),2) AS TOTAL_FREIGHT_VALUE, ROUND(AVG(oi.freight_value),2) AS
AVERAGE_FREIGHT_VALUE

FROM  `active-axle-390010.TARGET_SQL.order_items` AS oi
JOIN `active-axle-390010.TARGET_SQL.orders` AS o ON oi.order_id=o.order_id
JOIN `active-axle-390010.TARGET_SQL.customers` AS c ON o.customer_id=c.customer_id
GROUP BY c.customer_state
```



Row	customer_state	TOTAL_FREIGHT_VA	AVERAGE_FREIGHT_
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	PR	117851.68	20.53
4	SC	89660.26	21.47
5	DF	50625.5	21.04

5A) 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. Do this in a single query.

```
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 `active-axle-390010.TARGET_SQL.orders`
```

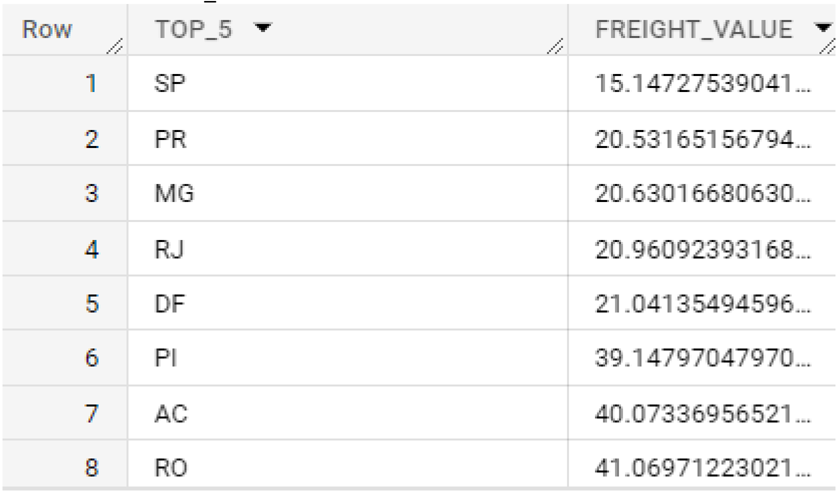


Row	order_id	time_to_deliver	diff_estimated_delive
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	28
3	65d1e226dfaeb8cdc42f66542...	35	16
4	635c894d068ac37e6e03dc54e...	30	1
5	3b97562c3aee8bdedcb5c2e45...	32	0

Insight: Negative value in "*diff_estimated_delivery*" means the order is delivered early than the estimated delivery date

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

```sql
SELECT * FROM (
(SELECT c.customer_state AS TOP_5, AVG(oi.freight_value) AS FREIGHT_VALUE
FROM `active-axle-390010.TARGET_SQL.order_items` as oi
JOIN `active-axle-390010.TARGET_SQL.orders` AS o ON  oi.order_id=o.order_id
JOIN  `active-axle-390010.TARGET_SQL.customers` AS c ON o.customer_id=c.customer_id
GROUP BY c.customer_state
ORDER BY AVG(oi.freight_value) DESC
LIMIT 5)
UNION ALL
(SELECT  c.customer_state AS TOP_5_BOTTOM_5, AVG(oi.freight_value) AS FREIGHT_VALUE
FROM `active-axle-390010.TARGET_SQL.order_items` as oi
JOIN `active-axle-390010.TARGET_SQL.orders` AS o ON  oi.order_id=o.order_id
JOIN  `active-axle-390010.TARGET_SQL.customers` AS c ON o.customer_id=c.customer_id
GROUP BY c.customer_state
ORDER BY AVG(oi.freight_value)
LIMIT 5)) as f
ORDER BY f.FREIGHT_VALUE
```



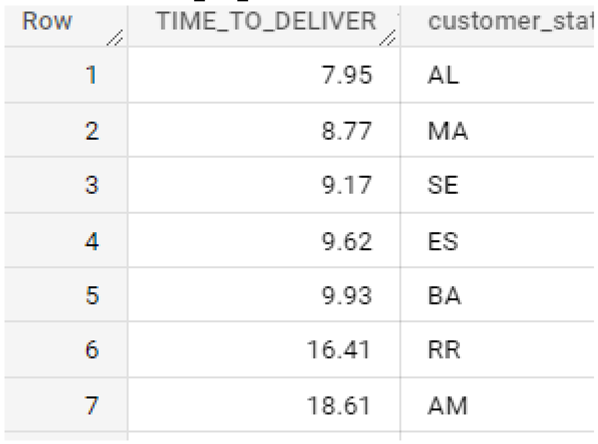
Row	TOP_5	FREIGHT_VALUE
1	SP	15.14727539041...
2	PR	20.53165156794...
3	MG	20.63016680630...
4	RJ	20.96092393168...
5	DF	21.04135494596...
6	PI	39.14797047970...
7	AC	40.07336956521...
8	RO	41.06971223021...

Insight: The first 5 rows represent the bottom 5 states with lowest freight value while the next 5 represent the states with highest freight value

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

```sql
SELECT * FROM
(SELECT ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,
o.order_delivered_customer_date, DAY)),2) AS TIME_TO_DELIVER, c.customer_state
FROM `active-axle-390010.TARGET_SQL.orders` AS o
JOIN `active-axle-390010.TARGET_SQL.customers` AS c
ON o.customer_id=c.customer_id
GROUP BY c.customer_state
ORDER BY TIME_TO_DELIVER
LIMIT 5)
UNION ALL
```

```
(SELECT ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,
o.order_delivered_customer_date,DAY)),2) AS TIME_TO_DELIVER,  c.customer_state
FROM `active-axle-390010.TARGET_SQL.orders` AS o
JOIN `active-axle-390010.TARGET_SQL.customers` AS c
ON o.customer_id=c.customer_id
GROUP BY c.customer_state
ORDER BY TIME_TO_DELIVER DESC
LIMIT 5)
ORDER BY TIME_TO_DELIVER
```

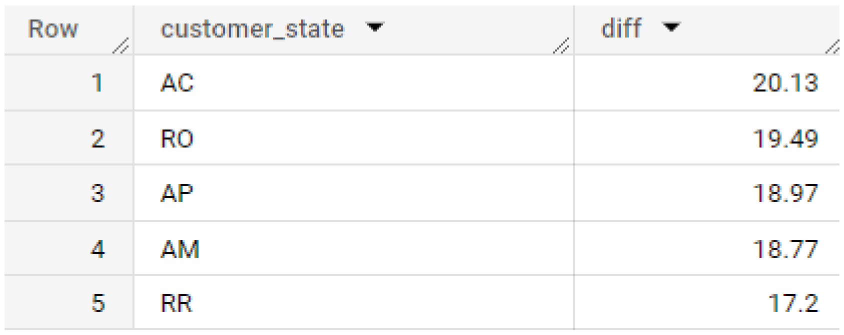


Row	TIME_TO_DELIVER	customer_stat
1	7.95	AL
2	8.77	MA
3	9.17	SE
4	9.62	ES
5	9.93	BA
6	16.41	RR
7	18.61	AM

<u>Insight</u>: The first 5 rows represent the 5 states with fastest delivery time states while the next 5 represent the states with longest delivery time

5D) Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
SELECT customer_state, ROUND((a.avg_estimated-a.avg_actual),2) AS diff FROM
(SELECT c.customer_state, AVG(DATE_DIFF(o.order_estimated_delivery_date,
o.order_purchase_timestamp, DAY)) AS avg_estimated,
       AVG(DATE_DIFF(o.order_delivered_customer_date ,
o.order_purchase_timestamp,DAY)) AS avg_actual
FROM `active-axle-390010.TARGET_SQL.orders` AS o
JOIN `active-axle-390010.TARGET_SQL.customers` AS c
ON o.customer_id=c.customer_id
GROUP BY c.customer_state) a
ORDER BY 2 DESC
LIMIT 5
```

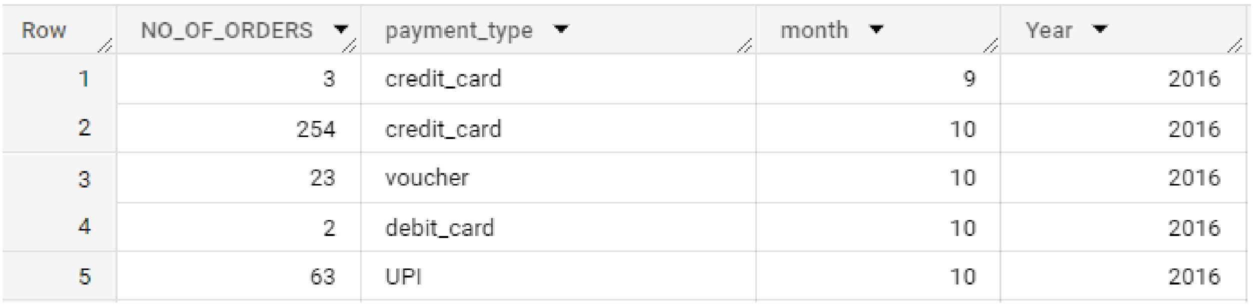


Row	customer_state	diff
1	AC	20.13
2	RO	19.49
3	AP	18.97
4	AM	18.77
5	RR	17.2

<u>Insight</u>: The "*diff*" column represent the difference between the estimated delivery time and actual delivery time of the order, so the greater the "*diff*" value the quicker the order got delivered.

6A) Find the month on month no. of orders placed using different payment types.

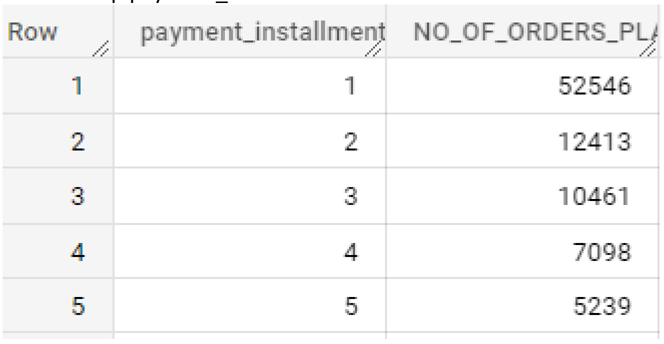
```
SELECT COUNT(o.order_id) AS NO_OF_ORDERS,p.payment_type, EXTRACT(month FROM o.order_purchase_timestamp) AS
month, EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year
FROM `active-axle-390010.TARGET_SQL.payments` AS p
JOIN `active-axle-390010.TARGET_SQL.orders` AS o
ON p.order_id=o.order_id
GROUP BY 2,3,4
ORDER BY 4,3
```



Row	NO_OF_ORDERS	payment_type	month	Year
1	3	credit_card	9	2016
2	254	credit_card	10	2016
3	23	voucher	10	2016
4	2	debit_card	10	2016
5	63	UPI	10	2016

6B) Find the no. of orders placed on the basis of the payment installments that have been paid.

```
SELECT p.payment_installments, COUNT(p.order_id) AS NO_OF_ORDERS_PLACED
FROM `active-axle-390010.TARGET_SQL.payments` AS p
WHERE p.payment_installments>0
GROUP BY p.payment_installments
```



Row	payment_installment	NO_OF_ORDERS_PL
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	5	5239