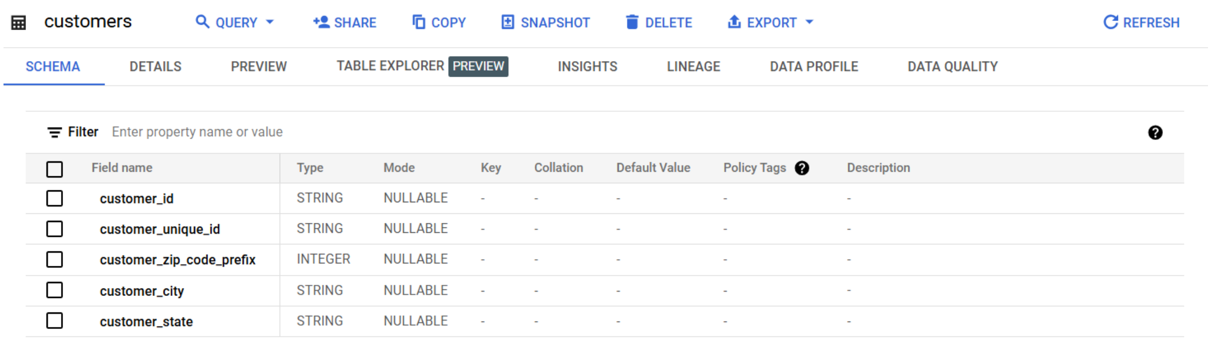
QUESTION 1)

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

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

OUTPUT:



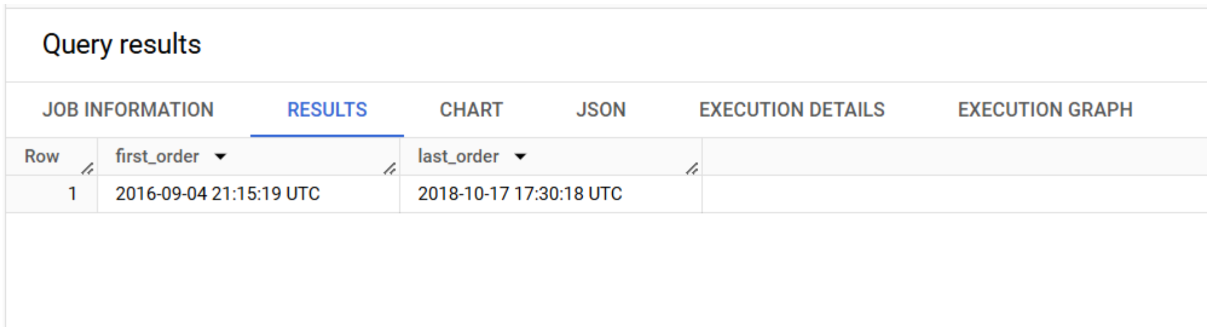
Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
customer_id	STRING	NULLABLE	-	-	-	-	-
customer_unique_id	STRING	NULLABLE	-	-	-	-	-
customer_zip_code_prefix	INTEGER	NULLABLE	-	-	-	-	-
customer_city	STRING	NULLABLE	-	-	-	-	-
customer_state	STRING	NULLABLE	-	-	-	-	-

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

QUERY:

```
SELECT MIN(order_purchase_timestamp) AS first_order,
MAX(order_purchase_timestamp) AS last_order
FROM `Business_Case_Target_SQL.orders`;
```

OUTPUT:



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

INSIGHTS:

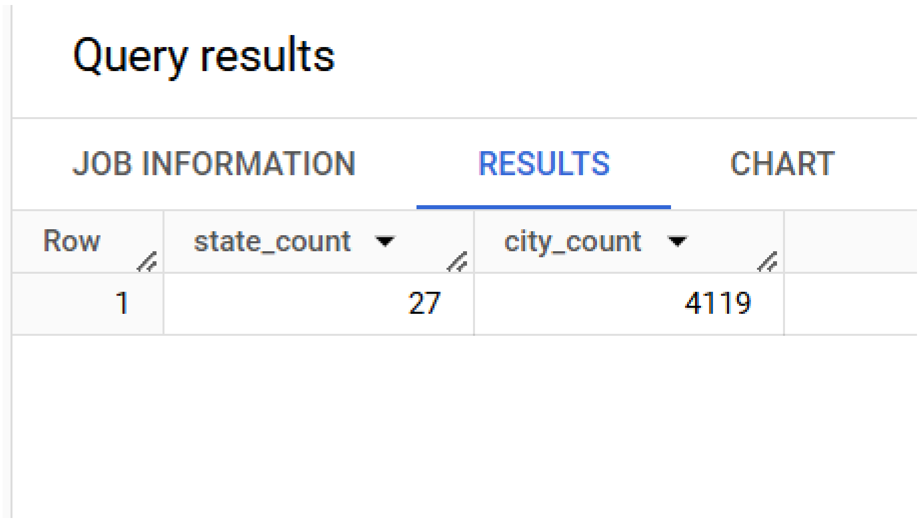
The orders were placed between **2016-09-04 21:15:19 UTC** and **2018-10-17 17:30:18 UTC**.

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

QUERY:

```
SELECT COUNT(DISTINCT c.customer_state) AS state_count,
COUNT(DISTINCT c.customer_city) AS city_count
FROM
`Business_Case_Target_SQL.customers` c
JOIN
`Business_Case_Target_SQL.orders` o
ON c.customer_id = o.customer_id;
```

OUTPUT:



Row	state_count	city_count
1	27	4119

INSIGHTS:

The total number of states from which customers placed orders is **27**, and the total number of cities is **4119**.

QUESTION 2)

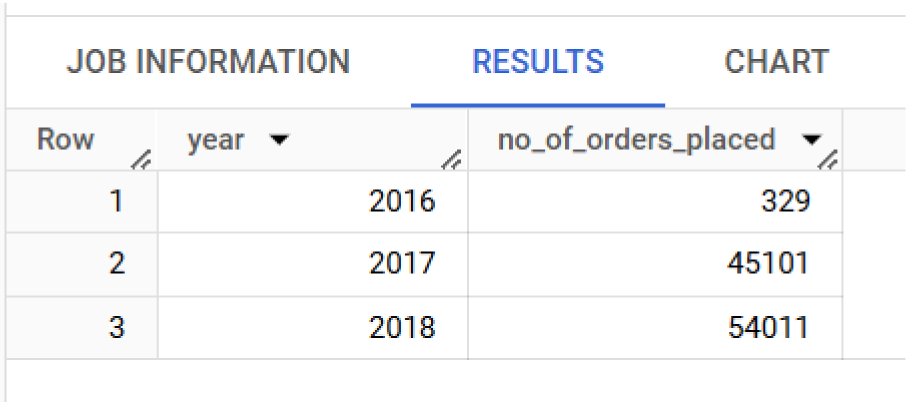
In-depth Exploration:

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

QUERY:

```
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS
year,COUNT(customer_id) AS no_of_orders_placed
FROM
`Business_Case_Target_SQL.orders`
GROUP BY EXTRACT(YEAR FROM order_purchase_timestamp)
ORDER BY year;
```

OUTPUT:



Row	year	no_of_orders_placed
1	2016	329
2	2017	45101
3	2018	54011

INSIGHTS:

Yes, there is a growing trend in the number of orders placed over the years. Here's the data:

- **2016**: 329 orders
- **2017**: 45,101 orders
- **2018**: 54,011 orders

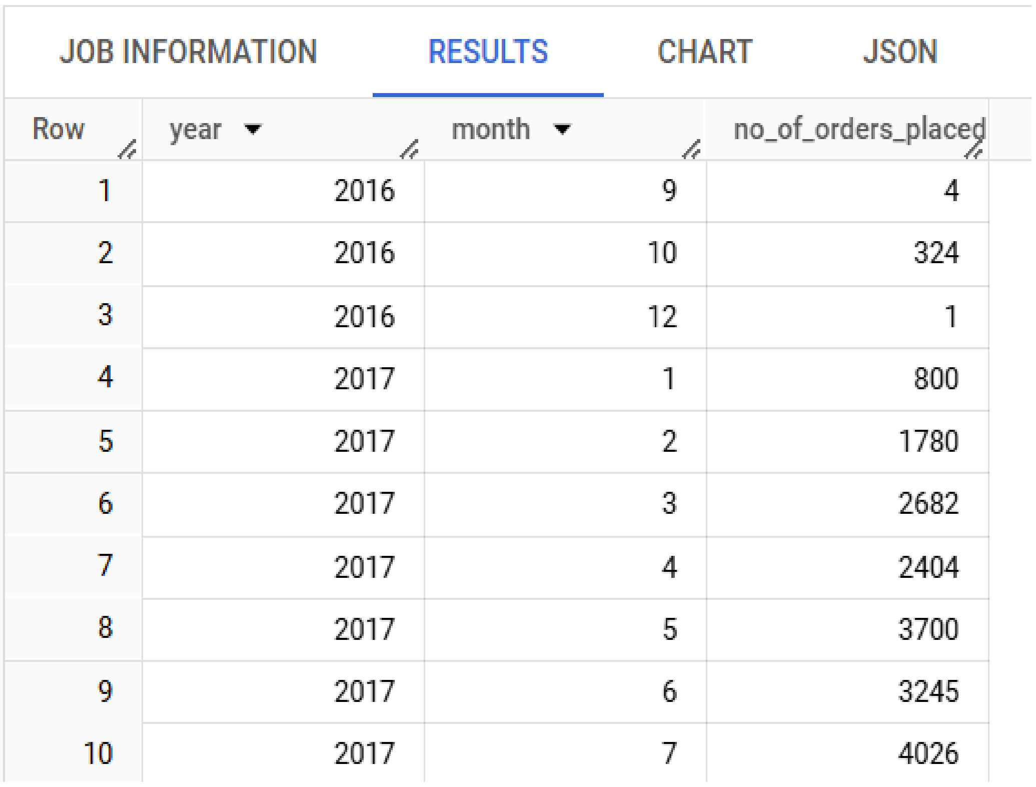
The significant increase in orders each year clearly indicates a consistent upward trend.

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

QUERY:

```
SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS
year,EXTRACT(MONTH FROM order_purchase_timestamp) AS
month,COUNT(customer_id) AS no_of_orders_placed
FROM
`Business_Case_Target_SQL.orders`
GROUP BY EXTRACT(YEAR FROM
order_purchase_timestamp),EXTRACT(MONTH FROM
order_purchase_timestamp)
ORDER BY year,month;
```

OUTPUT :



Row	year	month	no_of_orders_placed
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, there is seasonality in the data:

1. November-December: Significant increase due to Black Friday and Christmas shopping.
2. January-February: High activity during summer.
3. September: Notable dip in number of orders placed.

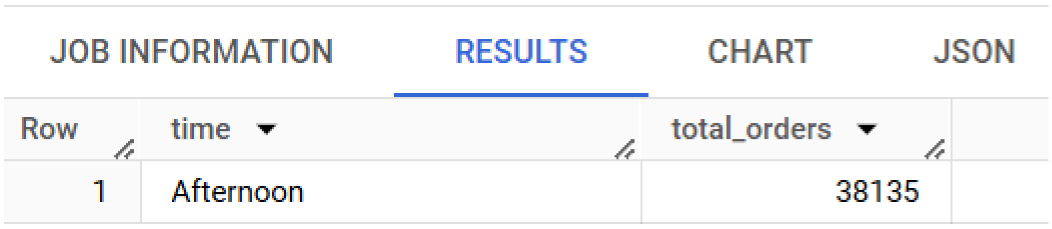
3) During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- 0-6 hrs : Dawn
- 7-12 hrs : Mornings
- 13-18 hrs : Afternoon
- 19-23 hrs : Night

QUERY:

```
WITH hour_detail AS(
  SELECT
  EXTRACT(HOUR FROM order_purchase_timestamp) AS hour,
  customer_id
  FROM
  `Business_Case_Target_SQL.orders`
),
grouping_by_time AS(
  SELECT
  CASE
    WHEN hour BETWEEN 0 AND 6 THEN 'Dawn'
    WHEN hour BETWEEN 7 AND 12 THEN 'Mornings'
    WHEN hour BETWEEN 13 AND 18 THEN 'Afternoon'
    ELSE 'Night'
  END AS time,
  COUNT(customer_id) AS orders_placed
  FROM
  hour_detail
  GROUP BY hour
  ORDER BY orders_placed
)
SELECT
time,
SUM(orders_placed) AS total_orders
FROM
grouping_by_time
GROUP BY time
ORDER BY total_orders DESC
LIMIT 1;
```

OUTPUT:



Row	time	total_orders
1	Afternoon	38135

INSIGHTS:

Brazilian customers mostly place their orders during the **Afternoon (13-18 hours)**. The total number of orders placed during this time period is **38,135**.

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

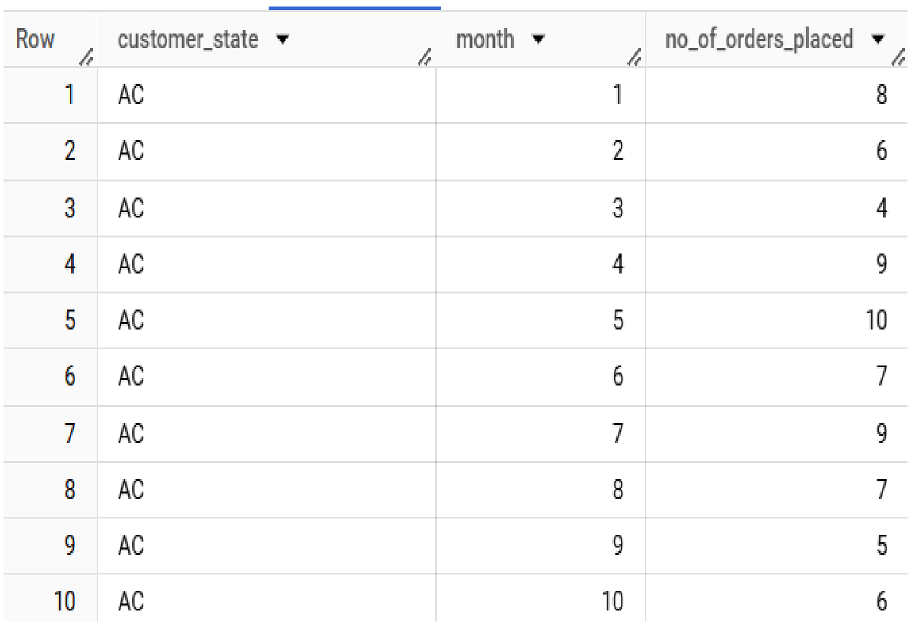
1) Get the month on month no. of orders placed in each state.

QUERY:

```
SELECT c.customer_state,EXTRACT(MONTH FROM
o.order_purchase_timestamp) AS
month,COUNT(c.customer_id) AS no_of_orders_placed

FROM `Business_Case_Target_SQL.orders` o
JOIN
`Business_Case_Target_SQL.customers` c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state,
EXTRACT(MONTH FROM o.order_purchase_timestamp)
ORDER BY c.customer_state,month;
```

OUTPUT:



Row	customer_state	month	no_of_orders_placed
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6

INSIGHTS:

This is the month-on-month count of orders placed in each state.

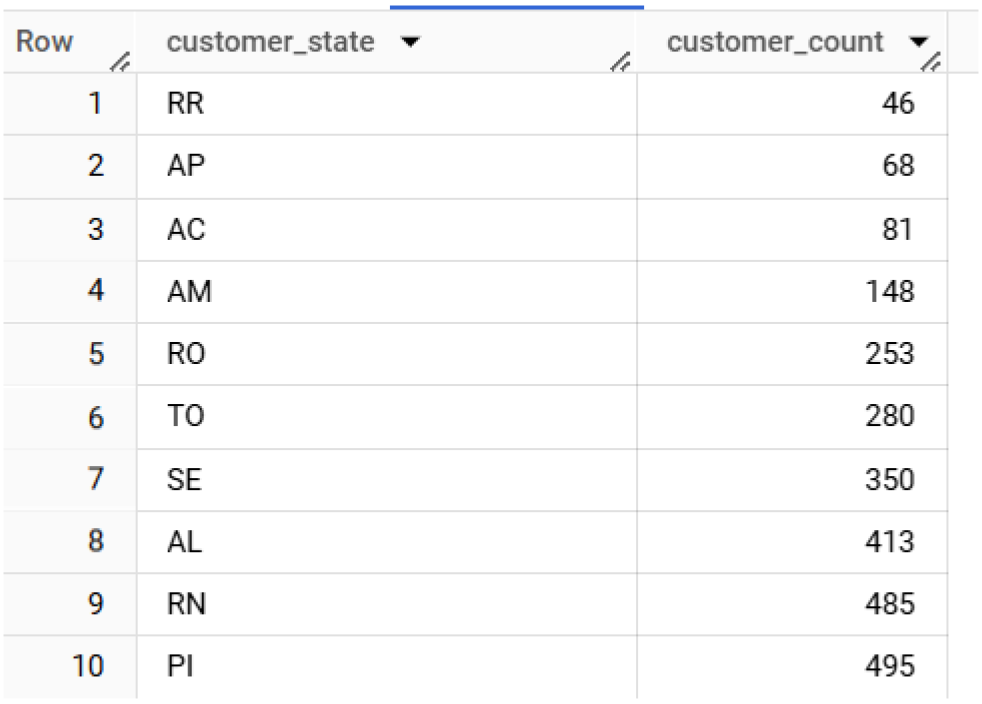
2) How are the customers distributed across all the states?

QUERY:

```
SELECT

customer_state,
COUNT(customer_id) AS customer_count
FROM
`Business_Case_Target_SQL.customers`
GROUP BY customer_state
ORDER BY customer_count;
```

OUTPUT:



Row	customer_state	customer_count
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

INSIGHTS

This is the distribution of customer count across all the states.

QUESTION 4)

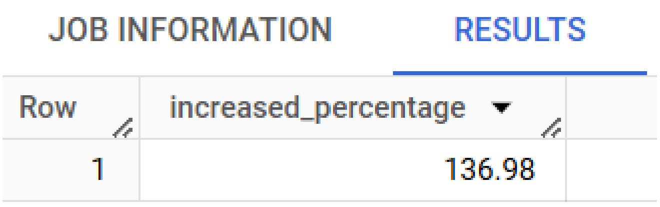
Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

1) Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).
You can use the "payment_value" column in the payments table to get the cost of orders.

QUERY:

```
WITH each_year_price_value AS
(
  SELECT SUM(p.payment_value) AS
total_value,EXTRACT(YEAR FROM
o.order_purchase_timestamp) AS year
  FROM
  `Business_Case_Target_SQL.orders` o
  JOIN
  `Business_Case_Target_SQL.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  EXTRACT(YEAR FROM
o.order_purchase_timestamp)
)
SELECT
ROUND(((SUM(CASE WHEN year = 2018 THEN total_value
END) -  SUM(CASE WHEN year = 2017 THEN total_value
END))/SUM(CASE WHEN year = 2017 THEN total_value
END))*100,2) AS increased_percentage
FROM each_year_price_value;
```

OUTPUT:


JOB INFORMATION RESULTS

Row	increased_percentage
1	136.98

INSIGHTS:

There was a 136.98% increase from 2017 to 2018.

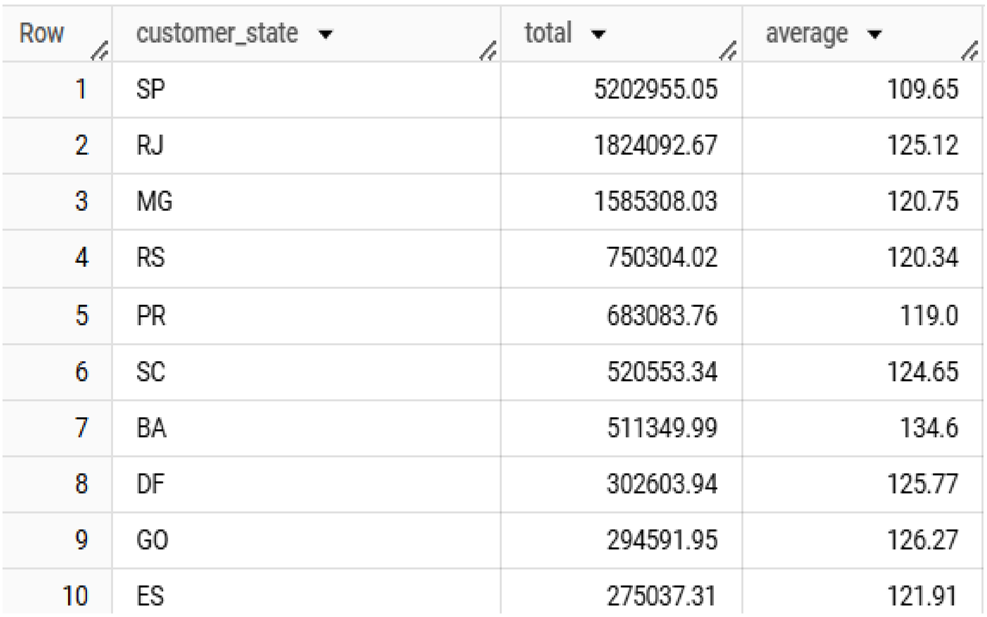
2) Calculate the Total & Average value of order price for each state.

QUERY:

```
SELECTc.customer_state, ROUND(SUM(oi.price),2) AS
total, ROUND(AVG(oi.price),2) AS average

FROM
`Business_Case_Target_SQL.orders` o
JOIN
`Business_Case_Target_SQL.order_items` oi
ON o.order_id = oi.order_id
JOIN
`Business_Case_Target_SQL.customers` c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY total DESC,average DESC;
```

OUTPUT:



Row	customer_state	total	average
1	SP	5202955.05	109.65
2	RJ	1824092.67	125.12
3	MG	1585308.03	120.75
4	RS	750304.02	120.34
5	PR	683083.76	119.0
6	SC	520553.34	124.65
7	BA	511349.99	134.6
8	DF	302603.94	125.77
9	GO	294591.95	126.27
10	ES	275037.31	121.91

INSIGHTS:

This is the total and average order price for all the states.

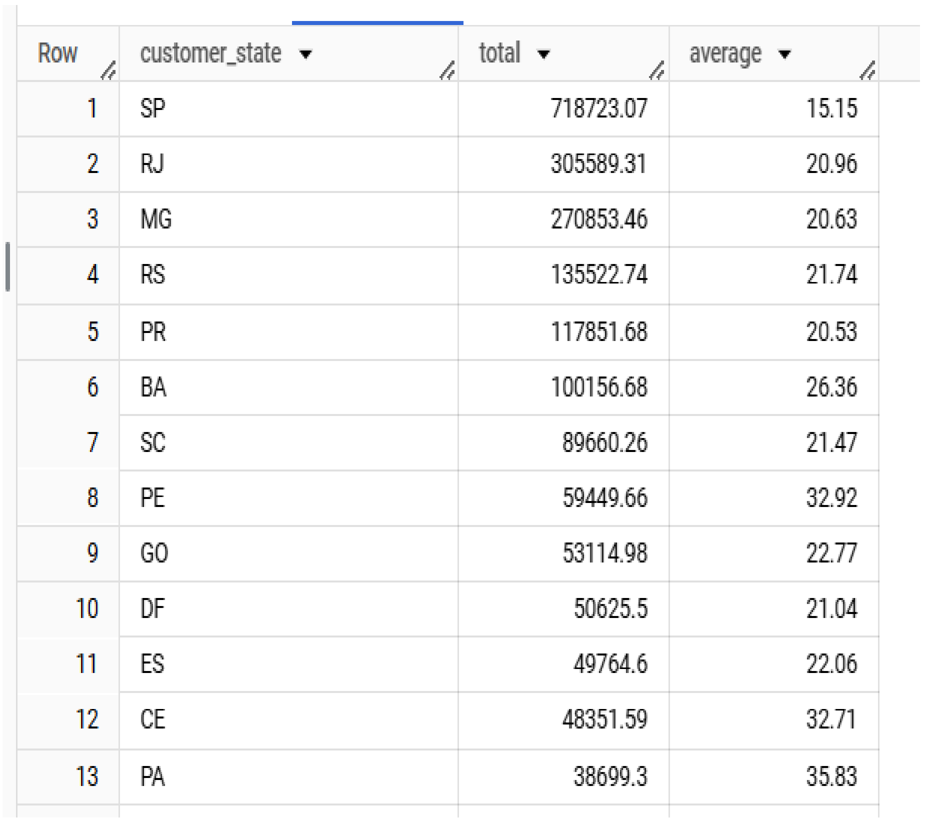
3) Calculate the Total & Average value of order freight for each state.

QUERY:

```
SELECTc.customer_state,
ROUND(SUM(oi.freight_value),2) AS total,
ROUND(AVG(oi.freight_value),2) AS average

FROM
`Business_Case_Target_SQL.orders` o
JOIN
`Business_Case_Target_SQL.order_items` oi
ON o.order_id = oi.order_id
JOIN
`Business_Case_Target_SQL.customers` c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY total DESC,average DESC;
```

OUTPUT:



Row	customer_state	total	average
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
11	ES	49764.6	22.06
12	CE	48351.59	32.71
13	PA	38699.3	35.83

INSIGHTS:

This is the total and average order freight for all the states.

QUESTION 5)

Analysis based on sales, freight and delivery time.

1) Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
Also, calculate the difference (in days) between the estimated & actual delivery date of an order.
Do this in a single query.

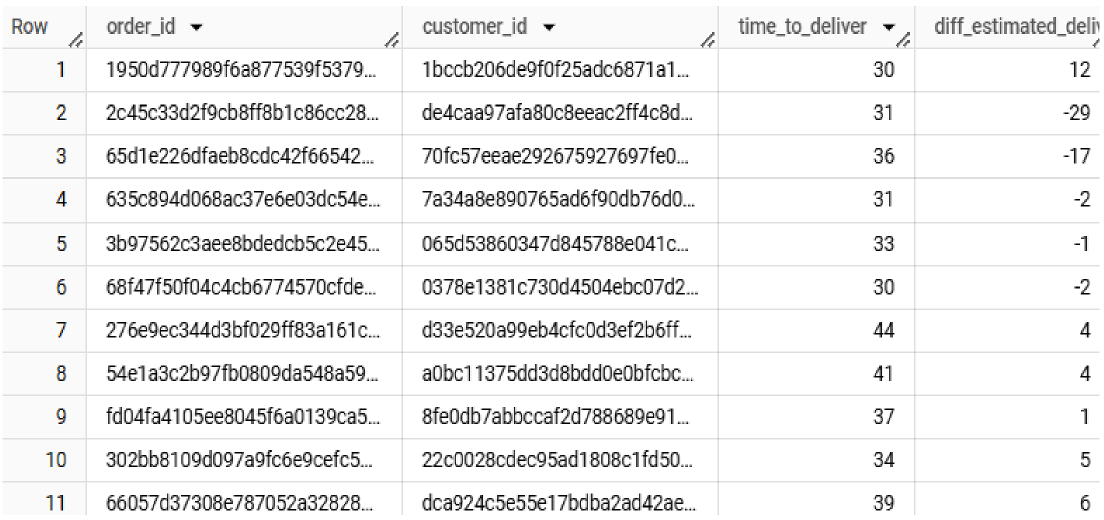
You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

- **time_to_deliver = order_delivered_customer_date - order_purchase_timestamp**
- **diff_estimated_delivery = order_delivered_customer_date - order_estimated_delivery_date**

QUERY:

```
SELECT
order_id,customer_id,
DATE_DIFF(order_delivered_customer_date,order_purchase_ti
mestamp,DAY) AS time_to_deliver,
DATE_DIFF(order_delivered_customer_date,order_estimated_d
elivery_date,DAY) AS diff_estimated_delivery
FROM
`Business_Case_Target_SQL.orders`
```

OUTPUT:



Row	order_id	customer_id	time_to_deliver	diff_estimated_deli
1	1950d777989f6a877539f5379...	1bccb206de9f0f25adc6871a1...	30	12
2	2c45c33d2f9cb8ff8b1c86cc28...	de4caa97afa80c8eeac2ff4c8d...	31	-29
3	65d1e226dfaeb8cdc42f66542...	70fc57eeae292675927697fe0...	36	-17
4	635c894d068ac37e6e03dc54e...	7a34a8e890765ad6f90db76d0...	31	-2
5	3b97562c3aee8bdedcb5c2e45...	065d53860347d845788e041c...	33	-1
6	68f47f50f04c4cb6774570cfde...	0378e1381c730d4504ebc07d2...	30	-2
7	276e9ec344d3bf029ff83a161c...	d33e520a99eb4cfc0d3ef2b6ff...	44	4
8	54e1a3c2b97fb0809da548a59...	a0bc11375dd3d8bdd0e0bfcbc...	41	4
9	fd04fa4105ee8045f6a0139ca5...	8fe0db7abbccaf2d788689e91...	37	1
10	302bb8109d097a9fc6e9cefc5...	22c0028cdec95ad1808c1fd50...	34	5
11	66057d37308e787052a32828...	dca924c5e55e17bdba2ad42ae...	39	6

INSIGHTS:

This is the time taken to deliver and the difference in the estimated delivery for all orders.

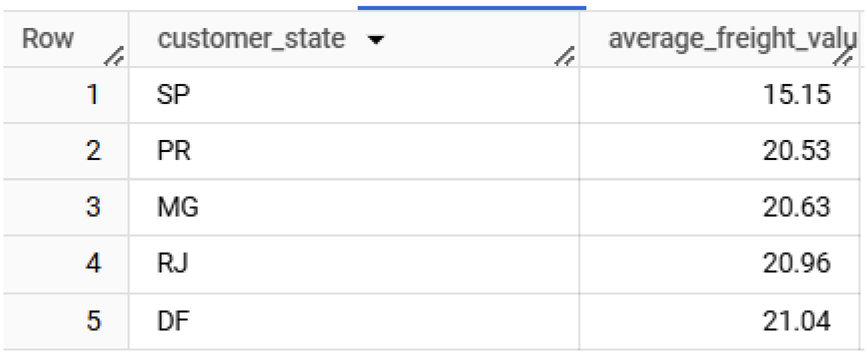
2) Find out the top 5 states with the highest & lowest average freight value.

TOP 5 STATES WITH LOWEST AVERAGE FRIEGHT VALUE:

QUERY:

```
SELECT c.customer_state, ROUND(AVG(oi.freight_value),2)
AS average_freight_value
FROM
Business_Case_Target_SQL.orders  o
JOIN
Business_Case_Target_SQL.customers c
ON o.customer_id = c.customer_id
JOIN
Business_Case_Target_SQL.order_items oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY average_freight_value
LIMIT 5;
```

OUTPUT:



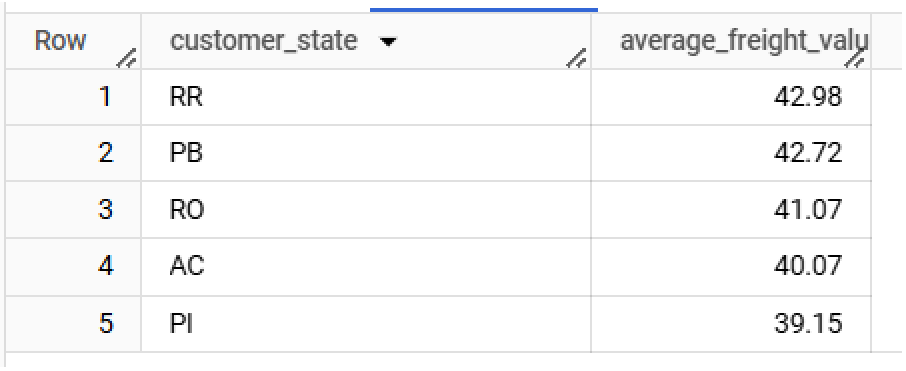
Row	customer_state	average_freight_valu
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

TOP 5 STATES WITH HIGHEST AVERAGE FRIEGHT VALUE:

QUERY:

```
SELECT c.customer_state, ROUND(AVG(oi.freight_value),2)
AS average_freight_value
FROM
Business_Case_Target_SQL.orders  o
JOIN
Business_Case_Target_SQL.customers c
ON o.customer_id = c.customer_id
JOIN
Business_Case_Target_SQL.order_items oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY average_freight_value DESC
LIMIT 5;
```

OUTPUT:



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

INSIGHTS:

These are the top 5 states with the lowest and highest average freight values.

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

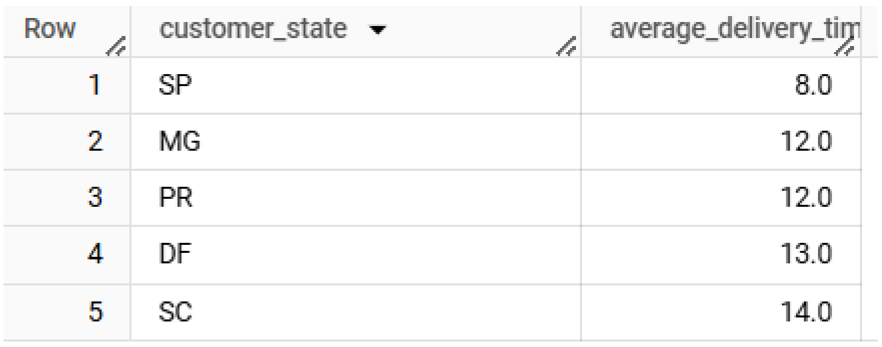
TOP 5 STATES WITH LOWEST AVERAGE DELIVERY TIME:

QUERY:

```
SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.ord
er_purchase_timestamp,DAY))) AS average_delivery_time

FROM
Business_Case_Target_SQL.orders  o
JOIN
Business_Case_Target_SQL.customers c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY average_delivery_time
LIMIT 5;
```

OUTPUT:



Row	customer_state	average_delivery_tim
1	SP	8.0
2	MG	12.0
3	PR	12.0
4	DF	13.0
5	SC	14.0

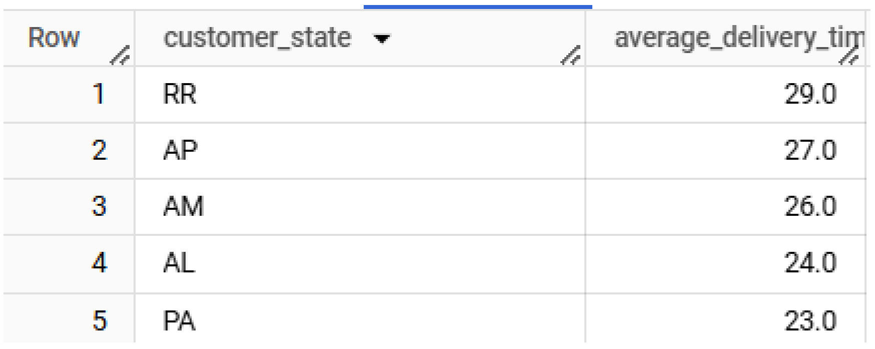
TOP 5 STATES WITH HIGHEST AVERAGE DELIVERY TIME:

QUERY:

```
SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.ord
er_purchase_timestamp,DAY))) AS average_delivery_time

FROM
Business_Case_Target_SQL.orders  o
JOIN
Business_Case_Target_SQL.customers c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY average_delivery_time DESC
LIMIT 5;
```

OUTPUT:



Row	customer_state	average_delivery_tim
1	RR	29.0
2	AP	27.0
3	AM	26.0
4	AL	24.0
5	PA	23.0

INSIGHTS:

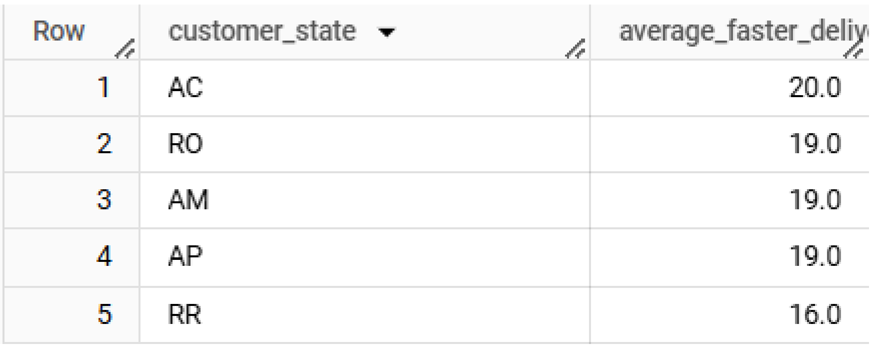
These are the top 5 states with the lowest and highest average delivery time.

4)Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.
You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

QUERY:

```
SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.ord
er_delivered_customer_date,DAY))) AS
average_faster_delivery
FROM
Business_Case_Target_SQL.orders  o
JOIN
Business_Case_Target_SQL.customers c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY average_faster_delivery DESC
LIMIT 5;
```

OUTPUT:



Row	customer_state	average_faster_deliv
1	AC	20.0
2	RO	19.0
3	AM	19.0
4	AP	19.0
5	RR	16.0

INSIGHTS:

These are the top 5 states with faster delivery times.

QUESTION 6)

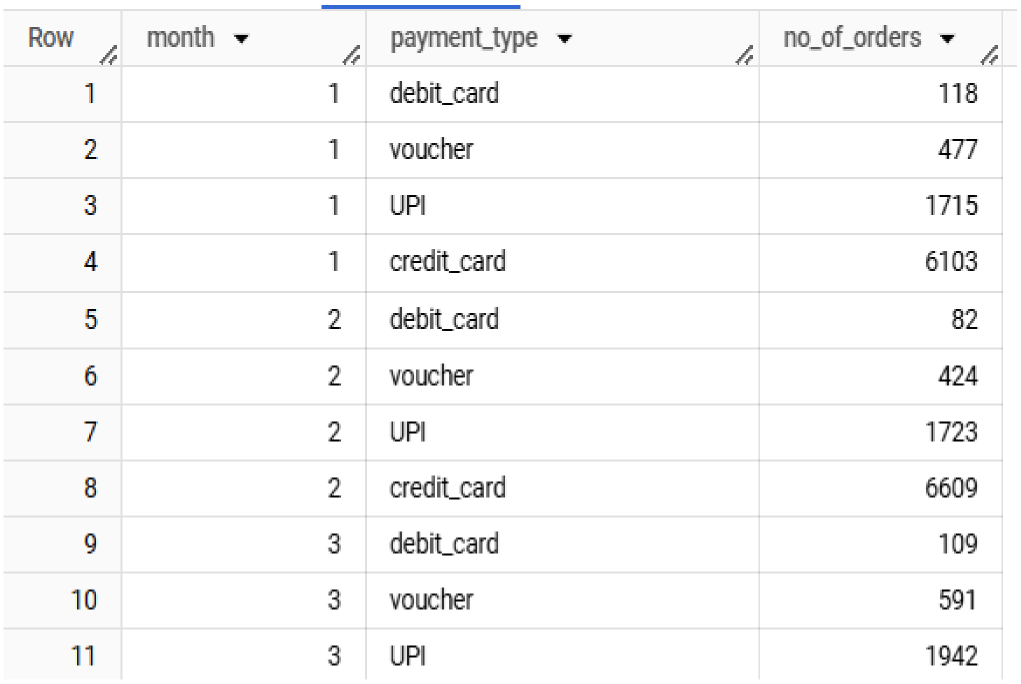
Analysis based on the payments:

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

QUERY:

```
SELECT EXTRACT(MONTH FROM o.order_purchase_timestamp) AS
month, p.payment_type, COUNT(*) AS no_of_orders
FROM
Business_Case_Target_SQL.orders  o
JOIN
Business_Case_Target_SQL.payments p
ON o.order_id = p.order_id
GROUP BY EXTRACT(MONTH FROM
o.order_purchase_timestamp),p.payment_type
ORDER BY month, no_of_orders;
```

OUPUT:



Row	month	payment_type	no_of_orders
1	1	debit_card	118
2	1	voucher	477
3	1	UPI	1715
4	1	credit_card	6103
5	2	debit_card	82
6	2	voucher	424
7	2	UPI	1723
8	2	credit_card	6609
9	3	debit_card	109
10	3	voucher	591
11	3	UPI	1942

INSIGHTS:

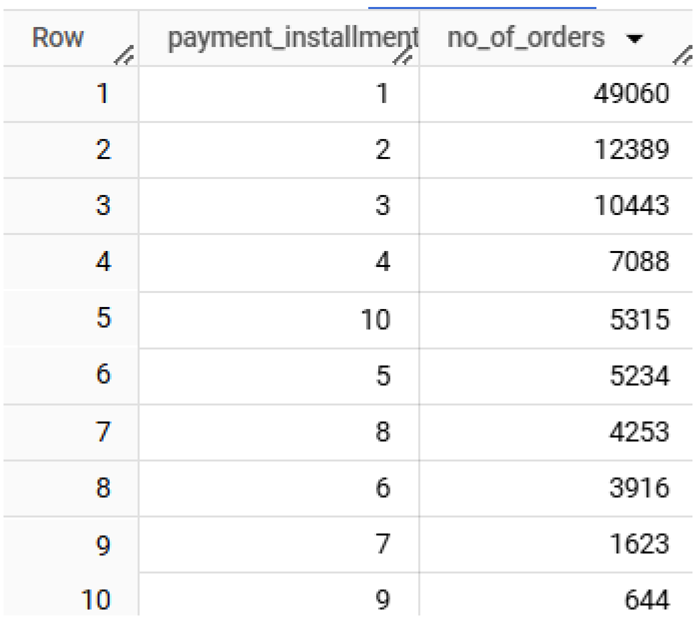
This is the month-on-month count for each payment type.

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

QUERY:

```
SELECT payment_installments, COUNT(DISTINCT order_id) AS
no_of_orders
FROM
`Business_Case_Target_SQL.payments`
GROUP BY payment_installments
ORDER BY no_of_orders DESC;
```

OUTPUT:



Row	payment_installment	no_of_orders
1	1	49060
2	2	12389
3	3	10443
4	4	7088
5	10	5315
6	5	5234
7	8	4253
8	6	3916
9	7	1623
10	9	644

INSIGHTS:

This is the number of orders for each payment installment.