TARGET BUSINESS CASE

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

```
SELECT column_name,data_type
FROM `project-case-401610.project.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers'
```

```
SELECT
  column_name,
  data_type
FROM
  `project-case-401610.project.INFORMATION_SCHEMA.COLUMNS`
WHERE
  table_name = 'customers'
```

Query results

SAVE RESU

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAIL

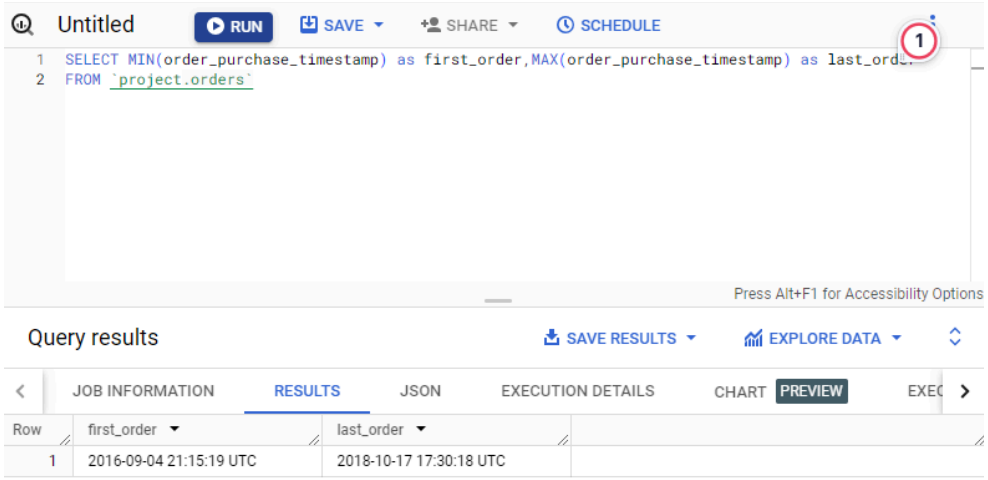
Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

INSIGHTS:

customer_id , customer_unique_id, customer_city, customer_state
Are string data types.
Customer_zip_code_prefix is an Integer data type.

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

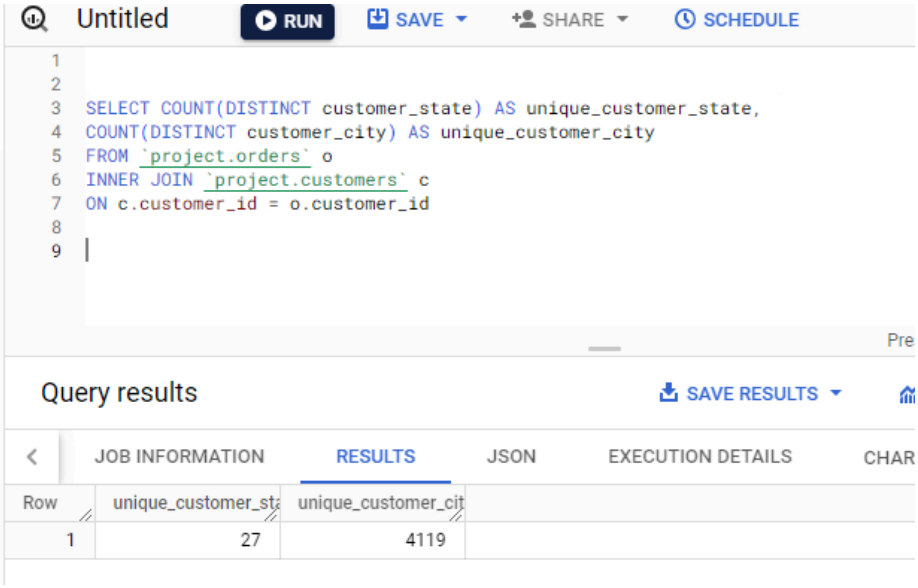
```
SELECT
MIN(order_purchase_timestamp) as first_order,
MAX(order_purchase_timestamp) as last_order
FROM `project.orders`
```



INSIGHTS
1st order placed on 4th september 2016 at around 9:15pm
Last order placed on 17th october 2018 at around 5:30pm

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

```
SELECT COUNT(DISTINCT customer_state) AS unique_customer_state,
COUNT(DISTINCT customer_city) AS unique_customer_city
FROM `project.orders` o
INNER JOIN `project.customers` c
ON c.customer_id = o.customer_id
```

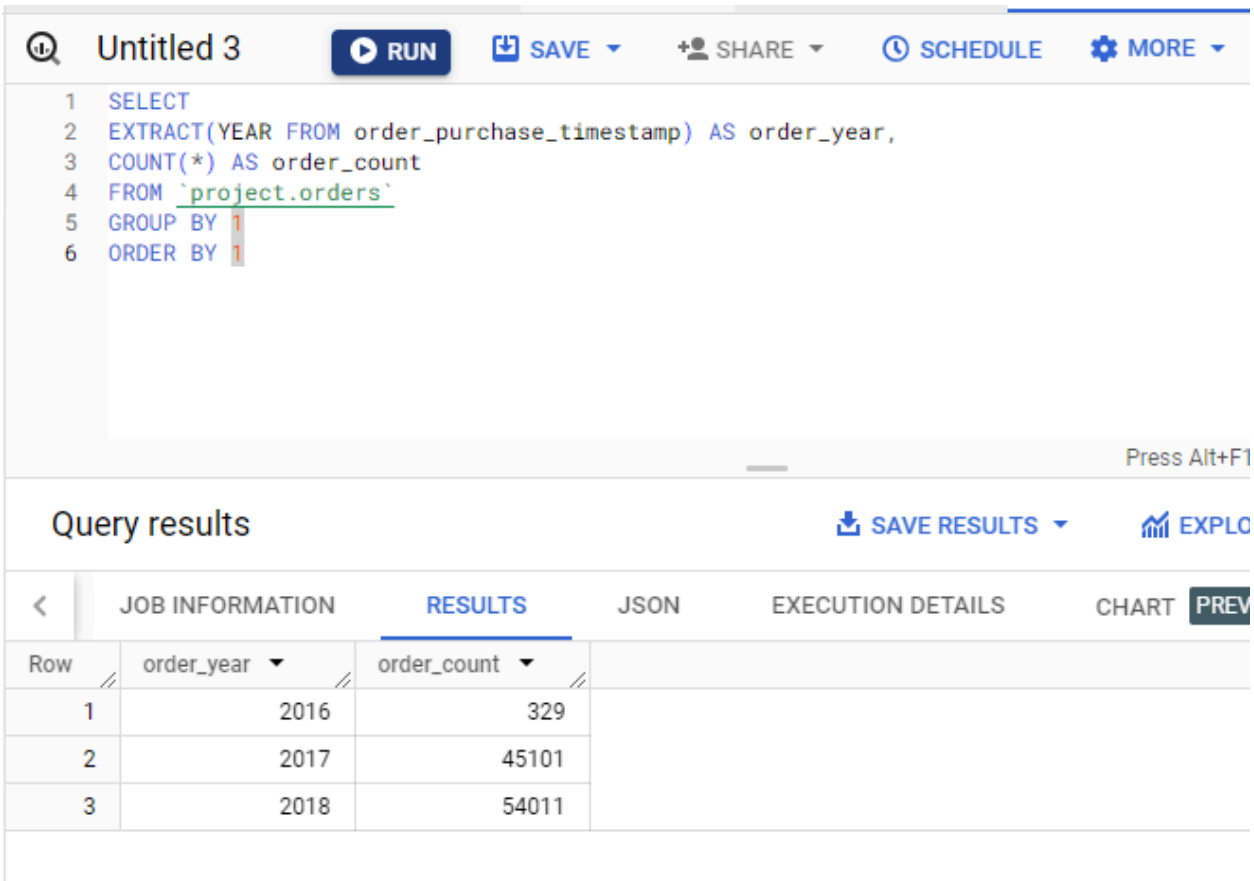


INSIGHTS
There are 27 unique states and 4119 unique cities.

II. In-depth Exploration:

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

```
SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
COUNT(*) AS order_count
FROM `project.orders`
GROUP BY 1
ORDER BY 1
```



Row	order_year	order_count
1	2016	329
2	2017	45101
3	2018	54011

INSIGHTS

In 2016 there are 329 orders were placed

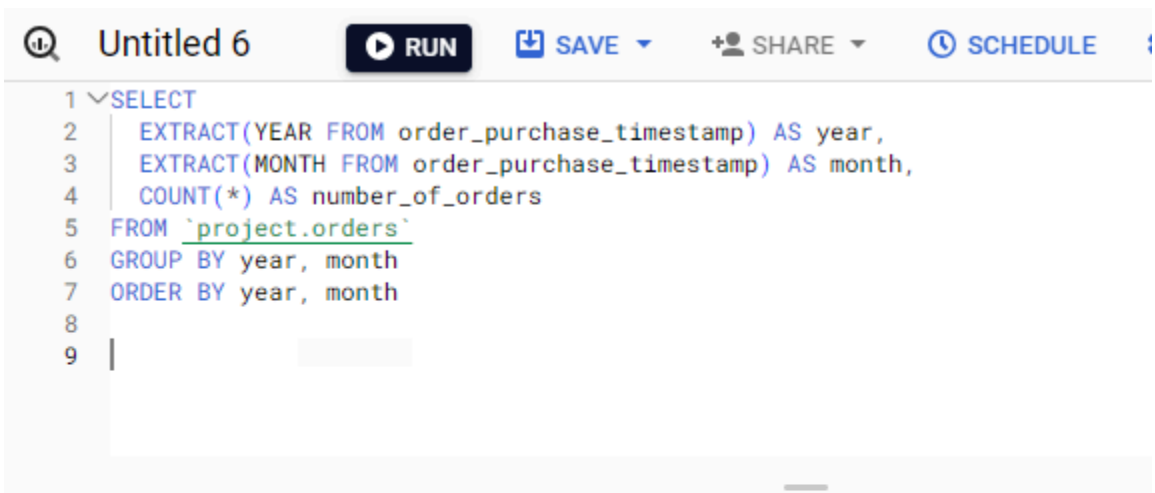
In 2017 there are 45101 orders were placed

In 2018 there are 54011 orders were placed

Therefore the trend has largely increased after 2016 and 2017. Trend is Increased in order.

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

```
SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
  EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
  COUNT(*) AS number_of_orders
FROM `project.orders`
GROUP BY year, month
ORDER BY year, month
```



Query results

SAVE RESULTS

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	year	month	number_of_orders
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800

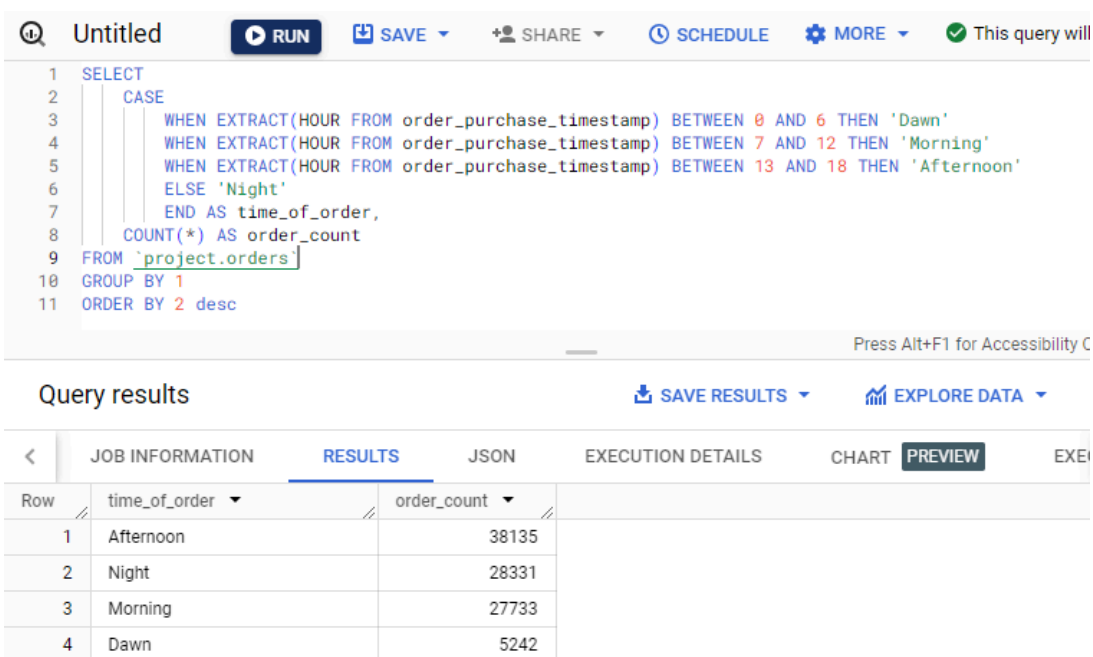
INSIGHTS

There are 25 rows every month orders.

C.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
    CASE
        WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 0 AND 6 THEN 'Dawn'
            WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 7 AND 12 THEN
'Morning'
            WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 13 AND 18 THEN
'Afternoon'
 ELSE 'Night'
 END AS time_of_order,
COUNT(*) AS order_count
FROM `project.orders`
GROUP BY 1
ORDER BY 2 desc
```



Row	time_of_order	order_count
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

INSIGHTS

Afternoon has the most order placed in brazil (38135)

III. Evolution of E-commerce orders in the Brazil region:

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

```sql
SELECT
    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
    c.customer_state,
    COUNT(*) AS num_orders
    FROM project.orders o
    JOIN project.customers c
    ON o.customer_id = c.customer_id
    GROUP BY 1,2,3
    ORDER BY 1,2,3
    LIMIT 10
```

```sql
SELECT
    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
    c.customer_state,
    COUNT(*) AS num_orders
FROM project.orders o
JOIN project.customers c
ON o.customer_id = c.customer_id
GROUP BY 1,2,3
ORDER BY 1,2,3
LIMIT 10
```

Press Alt+F1 for Access

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PREVIEW

w	year	month	customer_state	num_orders
1	2016	9	RR	1
2	2016	9	RS	1
3	2016	9	SP	2
4	2016	10	AL	2
5	2016	10	BA	4

INSIGHTS

I've limited the output to 10. There are 565 rows.

In 2016 orders started from September.

B. How are the customers distributed across all the states?

```
SELECT
customer_state,
COUNT(customer_unique_id) AS unique_customers
FROM `project.customers`
GROUP BY 1
ORDER BY 1
```

```
SELECT
    customer_state,
    COUNT(customer_unique_id) AS unique_customers
FROM `project.customers`
GROUP BY 1
ORDER BY 1
LIMIT 10
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECU

Row	customer_state	unique_customers
1	AC	81
2	AL	413
3	AM	148
4	AP	68
5	BA	3380
6	CE	1336
7	DF	2140
8	ES	2033

INSIGHTS

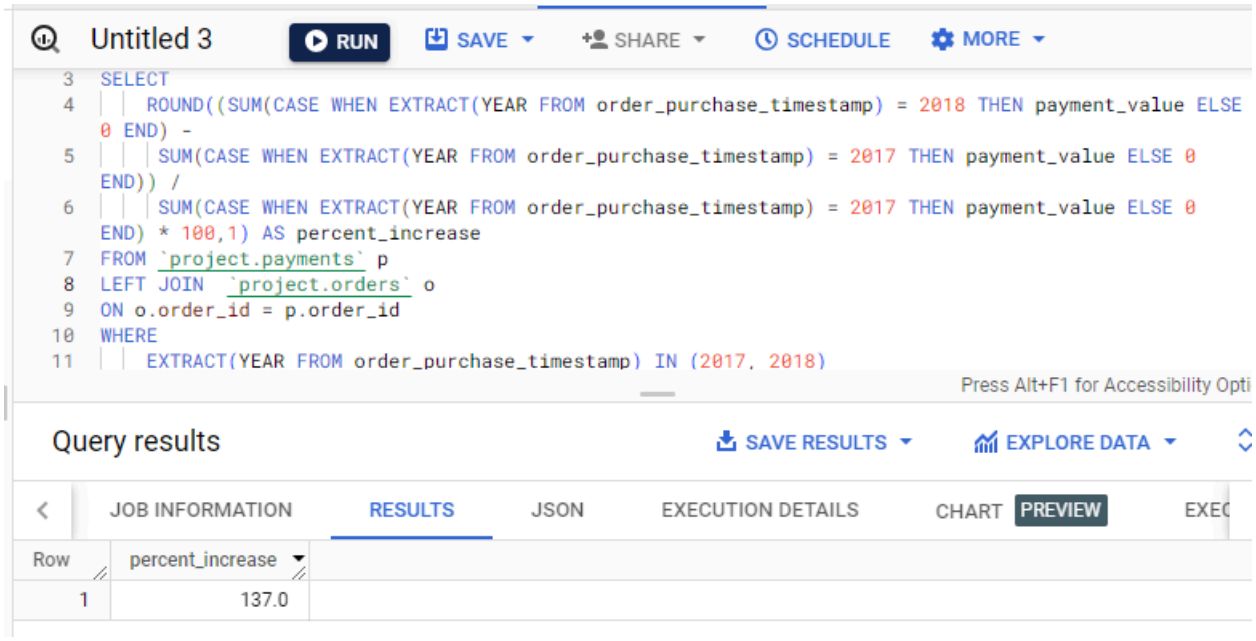
I've limited the output to 10. There are 27 rows.

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

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

```
SELECT
      ROUND((SUM(CASE WHEN EXTRACT(YEAR FROM order_purchase_timestamp) = 2018 THEN payment_value ELSE 0 END) -
          SUM(CASE WHEN EXTRACT(YEAR FROM order_purchase_timestamp) = 2017 THEN payment_value ELSE 0 END)) /
          SUM(CASE WHEN EXTRACT(YEAR FROM order_purchase_timestamp) = 2017 THEN payment_value ELSE 0 END) * 100,1) AS percent_increase
FROM `project.payments` p
```

```
LEFT JOIN  `project.orders` o
ON o.order_id = p.order_id
WHERE
    EXTRACT(YEAR FROM order_purchase_timestamp) IN (2017, 2018)
    AND EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8
```



INSIGHTS

From 2017 t0 2018

From January to August we have approximately 137% increase in the cost of orders

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

```
SELECT c.customer_state,
    ROUND(SUM(price)) AS total_order_price,
    ROUND(AVG(price)) AS average_order_price
FROM `project.orders` o
JOIN project.order_items ot
ON o.order_id = ot.order_id
JOIN `project.customers` c
ON c.customer_id = o.customer_id
GROUP BY 1
ORDER BY 1
LIMIT 10
```

Untitled 4 RUN SAVE SHARE SCHEDULE MORE

```
SELECT c.customer_state,
    ROUND(SUM(price)) AS total_order_price,
    ROUND(AVG(price)) AS average_order_price
FROM `project.orders` o
JOIN project.order_items ot
ON o.order_id = ot.order_id
JOIN `project.customers` c
ON c.customer_id = o.customer_id
GROUP BY 1
ORDER BY 1
LIMIT 10
```

Press Alt+F

Query results SAVE RESULTS EXPLO

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREV

Row	customer_state	total_order_price	average_order_price
1	AC	15983.0	174.0
2	AL	80315.0	181.0
3	AM	22357.0	135.0
4	AP	13474.0	164.0
5	BA	511350.0	135.0

INSIGHTS

I've limited the output to 10. There are 27 rows.

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

```
SELECT c.customer_state,
    ROUND(SUM(freight_value)) AS total_freight_value,
    ROUND(AVG(freight_value)) AS average_freight_value
FROM `project.orders` o
JOIN project.order_items ot
ON o.order_id = ot.order_id
JOIN `project.customers` c
ON c.customer_id = o.customer_id
GROUP BY 1
ORDER BY 1
LIMIT 10
```

Untitled 4 RUN SAVE SHARE SCHEDULE MORE Thi

```
SELECT c.customer_state,
    ROUND(SUM(freight_value)) AS total_freight_value,
    ROUND(AVG(freight_value)) AS average_freight_value
FROM `project.orders` o
JOIN project.order_items ot
ON o.order_id = ot.order_id
JOIN `project.customers` c
ON c.customer_id = o.customer_id
GROUP BY 1
ORDER BY 1
LIMIT 10
```

Press Alt+F1 for Acce

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW

Row	customer_state	total_freight_value	average_freight_valu
1	AC	3687.0	40.0
2	AL	15915.0	36.0
3	AM	5479.0	33.0
4	AP	2789.0	34.0

INSIGHTS

I've limited the output to 10. There are 27 rows.

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

A. 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,
TIMESTAMP_DIFF(order_delivered_customer_date,  order_purchase_timestamp,  DAY)  AS
time_to_deliver,
TIMESTAMP_DIFF(order_estimated_delivery_date,  order_delivered_customer_date,  DAY)  AS
diff_estimated_delivery
FROM `project.orders`
ORDER BY 1
LIMIT 10
```

```
SELECT
order_id,
TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_to_deliver,
TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS
diff_estimated_delivery
FROM `project.orders`
ORDER BY 1
LIMIT 10
```

Press Alt+F1 for Accessibility Opti

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS CHART PREVIEW JSON EXECUTION DETAILS EXEC

Row	order_id	time_to_deliver	diff_estimated_delive
3	000229ec398224ef6ca0b5 7da...	7	13
4	00024acbcdf0a6daa1e931b03...	6	5
5	00042b26cf59d7ce69dfabb4e...	25	15
6	00048cc3ae777c65dbb7d2a06...	6	14
7	00054e8431b9d7675808bcb8...	8	16
8	000576fe39319847cbb9d288c...	5	15
9	0005a1a1728c9d785b8e2b08...	9	0
10	0005f50442cb953dcd1d21e1f...	2	18

INSIGHTS

I analyzed using timestamps to find days. I've limited the output to 10.

There are 99441 rows.

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

```
(SELECT c.customer_state, AVG(ot.freight_value) AS avg_freight
FROM project.customers c
JOIN project.orders o
ON c.customer_id = o.customer_id
JOIN `project.order_items` ot
ON o.order_id = ot.order_id
GROUP BY 1
ORDER BY 2
LIMIT 5)

UNION ALL

(SELECT c.customer_state, AVG(ot.freight_value) AS avg_freight
FROM project.customers c
JOIN project.orders o
ON c.customer_id = o.customer_id
JOIN `project.order_items` ot
```

```
ON o.order_id = ot.order_id
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5)
```

Row	customer_state	avg_freight
1	SP	15.14727539041...
2	PR	20.53165156794...
3	MG	20.63016680630...
4	RJ	20.96092393168...
5	DF	21.04135494596...
6	RR	42.98442307692...
7	PB	42.72380398671...
8	RO	41.06971223021...
9	AC	40.07336956521...
10	PI	39.14797047970...

INSIGHTS

I've been finding out the top 5 and bottom 5 of average freight value.

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

```
(SELECT
c.customer_state,
ROUND(AVG(DATE_DIFF(order_delivered_customer_date,  order_purchase_timestamp,DAY)))  AS
avg_delivery_time
FROM project.customers c
JOIN project.orders o
ON c.customer_id = o.customer_id
WHERE order_delivered_customer_date IS NOT NULL
GROUP BY 1
ORDER BY 2
LIMIT 5)

UNION ALL

(SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(order_delivered_customer_date,  order_purchase_timestamp,DAY)))  AS
avg_delivery_time
```

```
FROM project.customers c
JOIN project.orders o
ON c.customer_id = o.customer_id
WHERE order_delivered_customer_date IS NOT NULL
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5)
```

Row	customer_state	avg_delivery_time
1	SP	8.0
2	MG	12.0
3	PR	12.0
4	DF	13.0
5	SC	14.0
6	RR	29.0
7	AP	27.0
8	AM	26.0
9	AL	24.0
10	PA	23.0

INSIGHTS

I've been finding out the top 5 and bottom 5 of average approx delivery days.

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

```
SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date,DA
Y))) AS avg_delivery_speed
FROM project.orders o
JOIN project.customers c
ON c.customer_id = o.customer_id
WHERE order_status = 'delivered'
GROUP BY 1
ORDER BY 2
LIMIT 5
```

```
SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date,DAY))) AS avg_delivery_speed
FROM project.orders o
JOIN project.customers c
ON c.customer_id = o.customer_id
WHERE order_status = 'delivered'
GROUP BY 1
ORDER BY 2
LIMIT 5
```

Press Alt+F1 for Accessibility

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS CHART PREVIEW JSON EXECUTION DETAILS EX

Row	customer_state	avg_delivery_speed
1	AL	8.0
2	MA	9.0
3	SE	9.0
4	CE	10.0
5	MS	10.0

INSIGHTS

Average speed at approximately 8 days. I've been used round() function to find the approximate days.

VI. Analysis based on the payments:

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

```
SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
  EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
  payment_type,
  COUNT(*) AS number_of_orders
FROM project.payments p
JOIN project.orders o
ON p.order_id = o.order_id
GROUP BY year, month, payment_type
ORDER BY year, month, payment_type
LIMIT 10
```

```
SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
  EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
  payment_type,
  COUNT(*) AS number_of_orders
FROM project.payments p
JOIN project.orders o
ON p.order_id = o.order_id
GROUP BY year, month, payment_type
ORDER BY year, month, payment_type
LIMIT 10
```

Press Alt+F1 for Acce

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PREVIEW

Row	year	month	payment_type	number_of_orders
1	2016	9	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

INSIGHTS

I've limited the output to 10. There are 90 rows.

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

```
SELECT payment_installments,
COUNT(DISTINCT p.order_id) AS number_of_orders
FROM project.payments p
JOIN project.orders o
ON o.order_id = p.order_id
GROUP BY payment_installments
HAVING payment_installments = 0
ORDER BY payment_installments
```

Untitled 6 RUN SAVE SHARE

```sql
SELECT payment_installments,
COUNT(DISTINCT p.order_id) AS number_of_orders
FROM project.payments p
JOIN project.orders o
ON o.order_id = p.order_id
GROUP BY payment_installments
HAVING payment_installments = 0
ORDER BY payment_installments
```

Query results SAVE

JOB INFORMATION | RESULTS | JSON | EXECUTION D

Row	payment_installment	number_of_orders
1	0	2

INSIGHTS

There are only 2 orders that are fully paid.