E-commerce SQL Case Study

Overview of all the Tables Used :

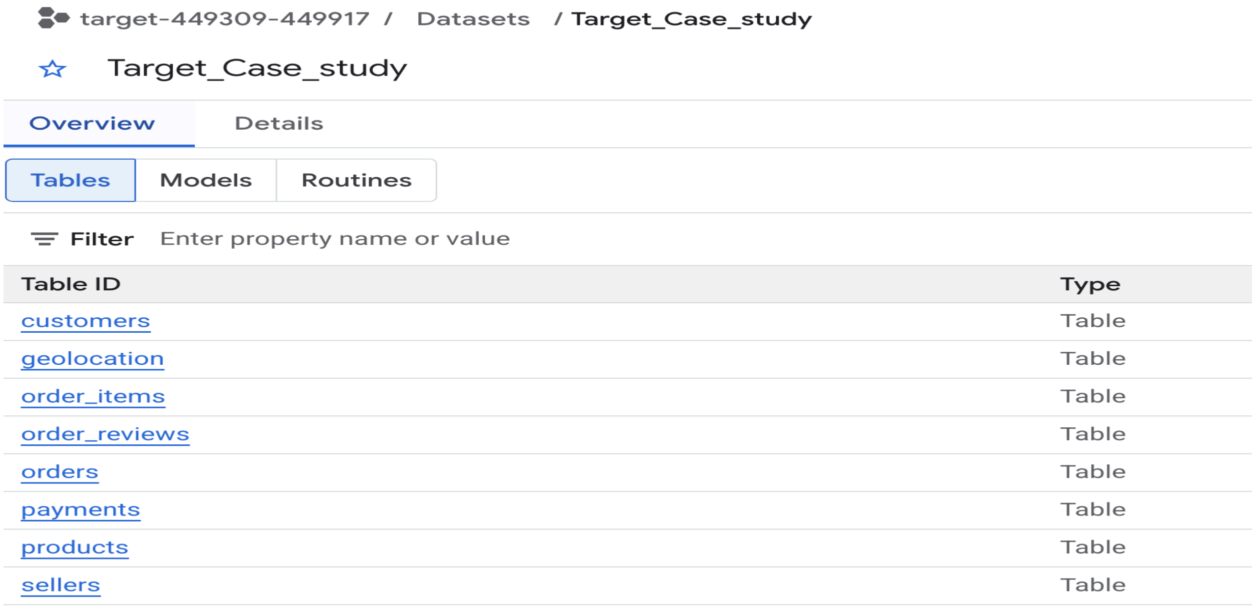
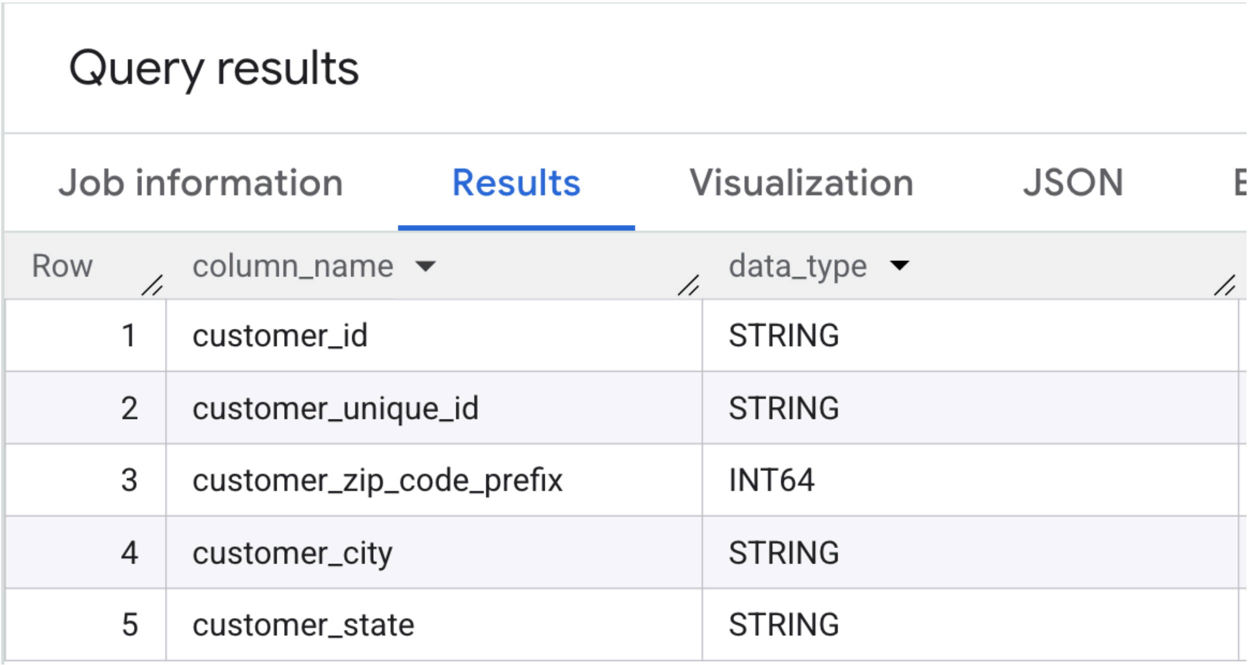

target-449309-449917 / Datasets / Target_Case_study
☆ Target_Case_study
Overview
Details
Tables
Models
Routines
Filter
Enter property name or value

Table ID	Type
customers	Table
geolocation	Table
order_items	Table
order_reviews	Table
orders	Table
payments	Table
products	Table
sellers	Table

I. Customer & Order Overview

A. Data Types of All Columns in the `customers` Table

```
SELECT
 column_name,
 data_type,
 is_nullable
FROM `target-449309-449917.Target_Case_study.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers'
ORDER BY ordinal_position
```



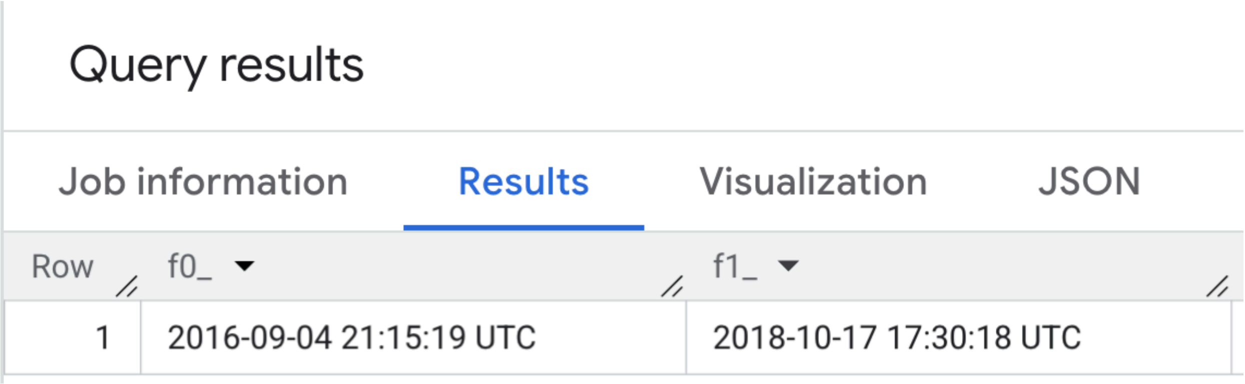
Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

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

```
select min(order_purchase_timestamp),max(order_purchase_timestamp)
from `target-449309-449917.Target_Case_study.orders`
```

Earliest order placed: `2016-09-04 21:15:19 UTC`

Latest order placed: `2018-10-17 17:30:18 UTC`

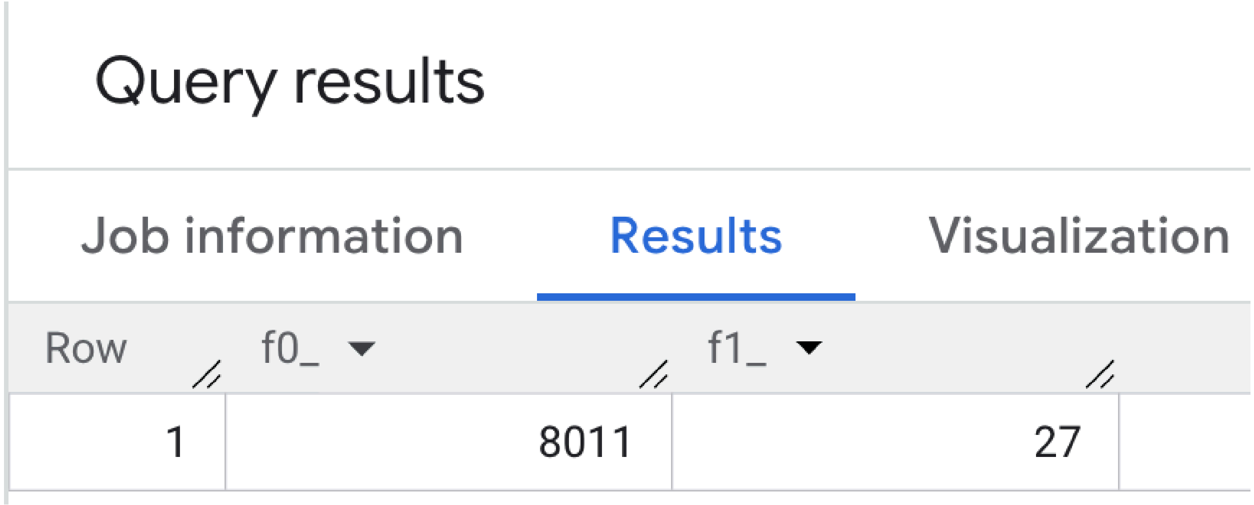


Row	f0_	f1_
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

C. Count of cities and states with customers who placed orders

```
select
count(distinct geolocation_city),
count(distinct geolocation_state) from
`target-449309-449917.Target_Case_study.geolocation`
```

This Shows the geographical spread of the customer base.

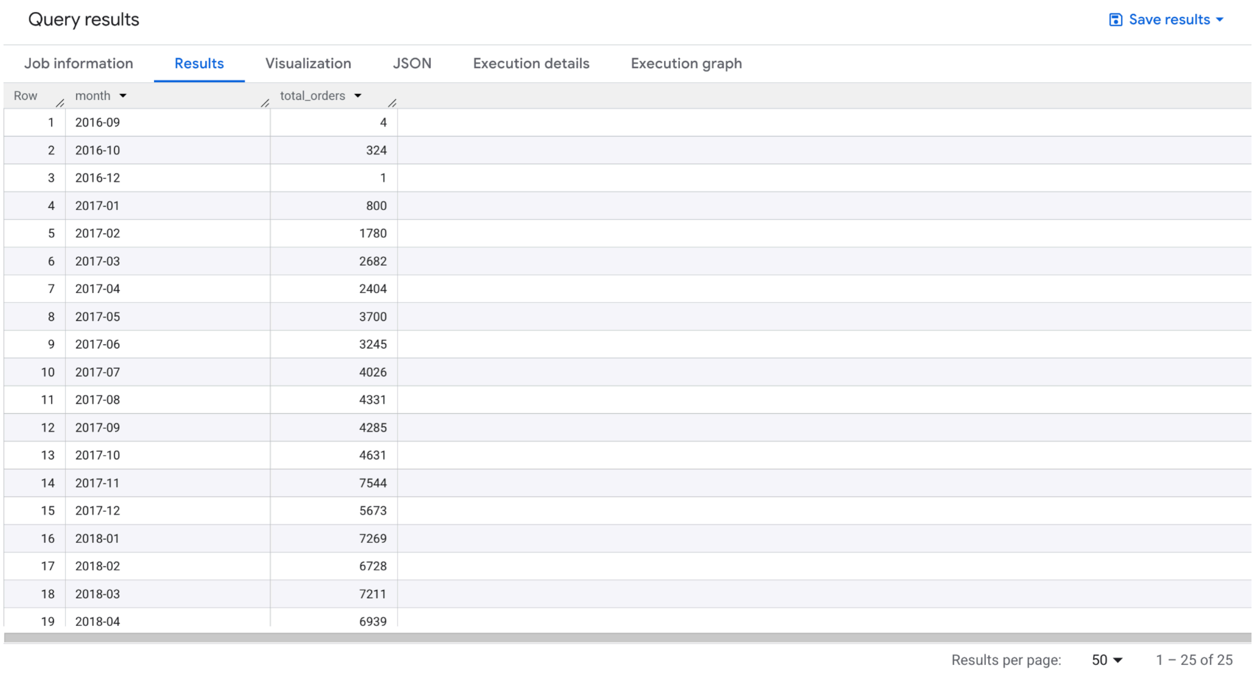


Row	f0_	f1_
1	8011	27

II. Order Trends & Seasonality

A. Month-on-month trend of orders over the years

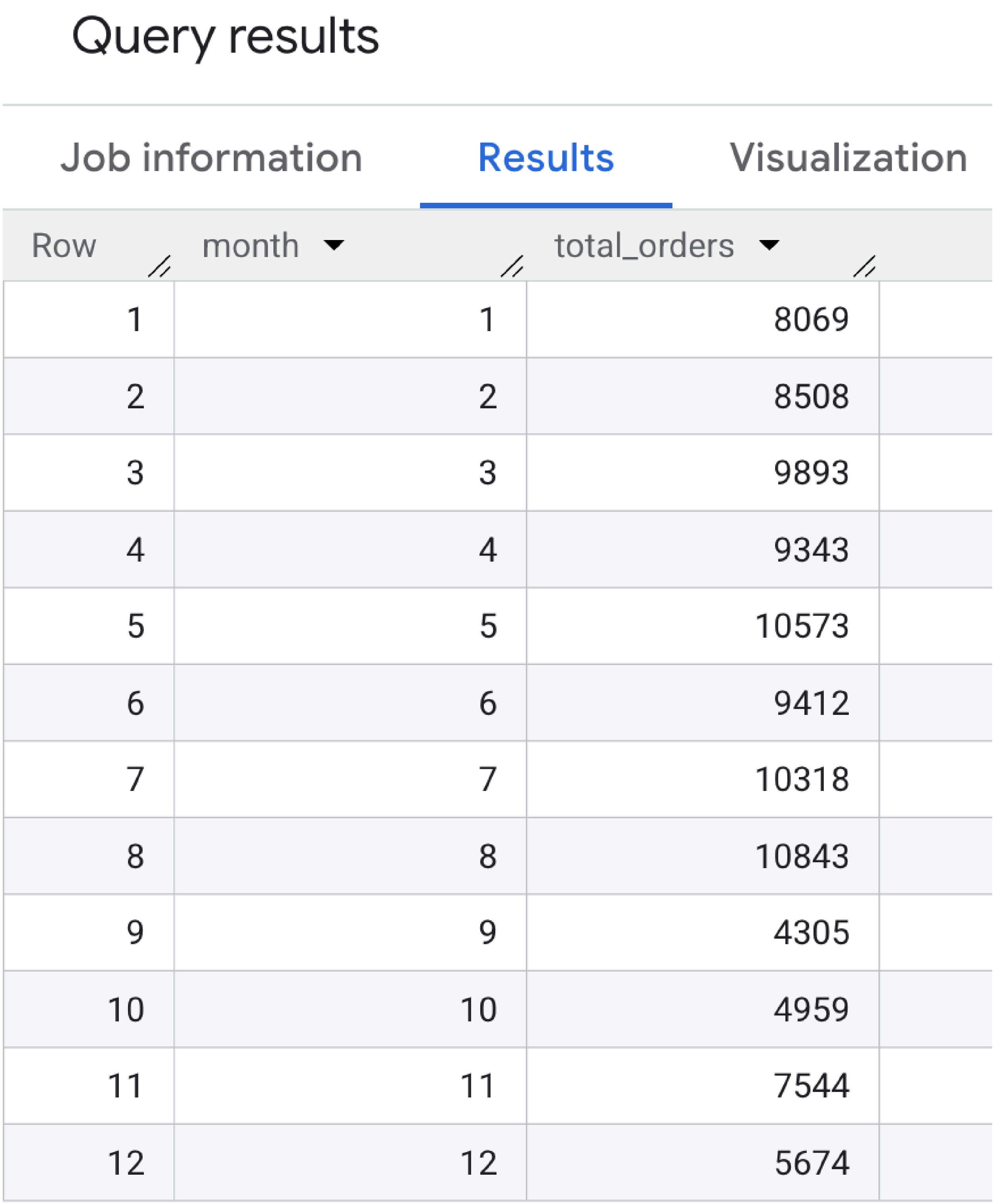
```
SELECT
FORMAT_DATE('%Y-%m', order_purchase_timestamp) AS month,
COUNT(order_id) AS total_orders
from `target-449309-449917.Target_Case_study.orders`
GROUP BY 1
ORDER BY 1;
```



Row	month	total_orders
1	2016-09	4
2	2016-10	324
3	2016-12	1
4	2017-01	800
5	2017-02	1780
6	2017-03	2682
7	2017-04	2404
8	2017-05	3700
9	2017-06	3245
10	2017-07	4026
11	2017-08	4331
12	2017-09	4285
13	2017-10	4631
14	2017-11	7544
15	2017-12	5673
16	2018-01	7269
17	2018-02	6728
18	2018-03	7211
19	2018-04	6939

B. **Monthly seasonality in order placement**

```
SELECT EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
COUNT(order_id) AS total_orders
from `target-449309-449917.Target_Case_study.orders`
GROUP BY 1
ORDER BY 1;
```



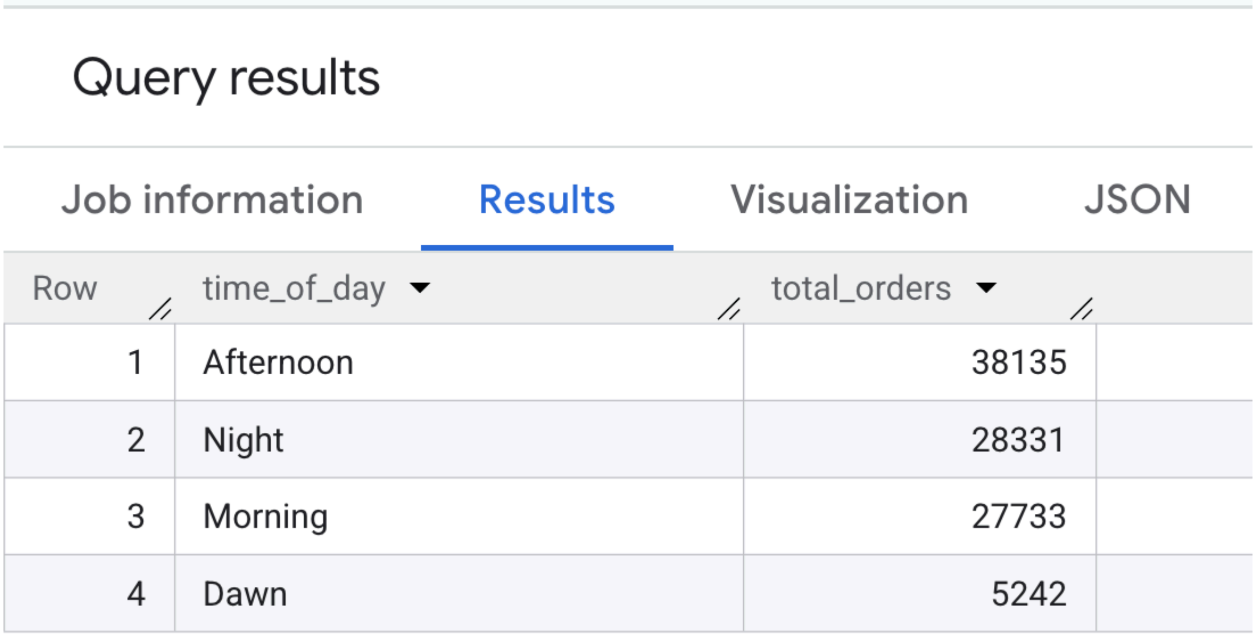
Query results

Row	month	total_orders
1	1	8069
2	2	8508
3	3	9893
4	4	9343
5	5	10573
6	6	9412
7	7	10318
8	8	10843
9	9	4305
10	10	4959
11	11	7544
12	12	5674

Orders peak mostly between February and August, indicating seasonal demand

C. Time of day when customers place orders

```
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"
when  extract(hour from order_purchase_timestamp)  between 19 and 23 then "Night"
end as time_of_day,
count(*) as total_orders
from `target-449309-449917.Target_Case_study.orders`
group by time_of_day
order by total_orders desc
```



Row	time_of_day	total_orders
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

Insight: Most orders are placed during the evening and night hours.

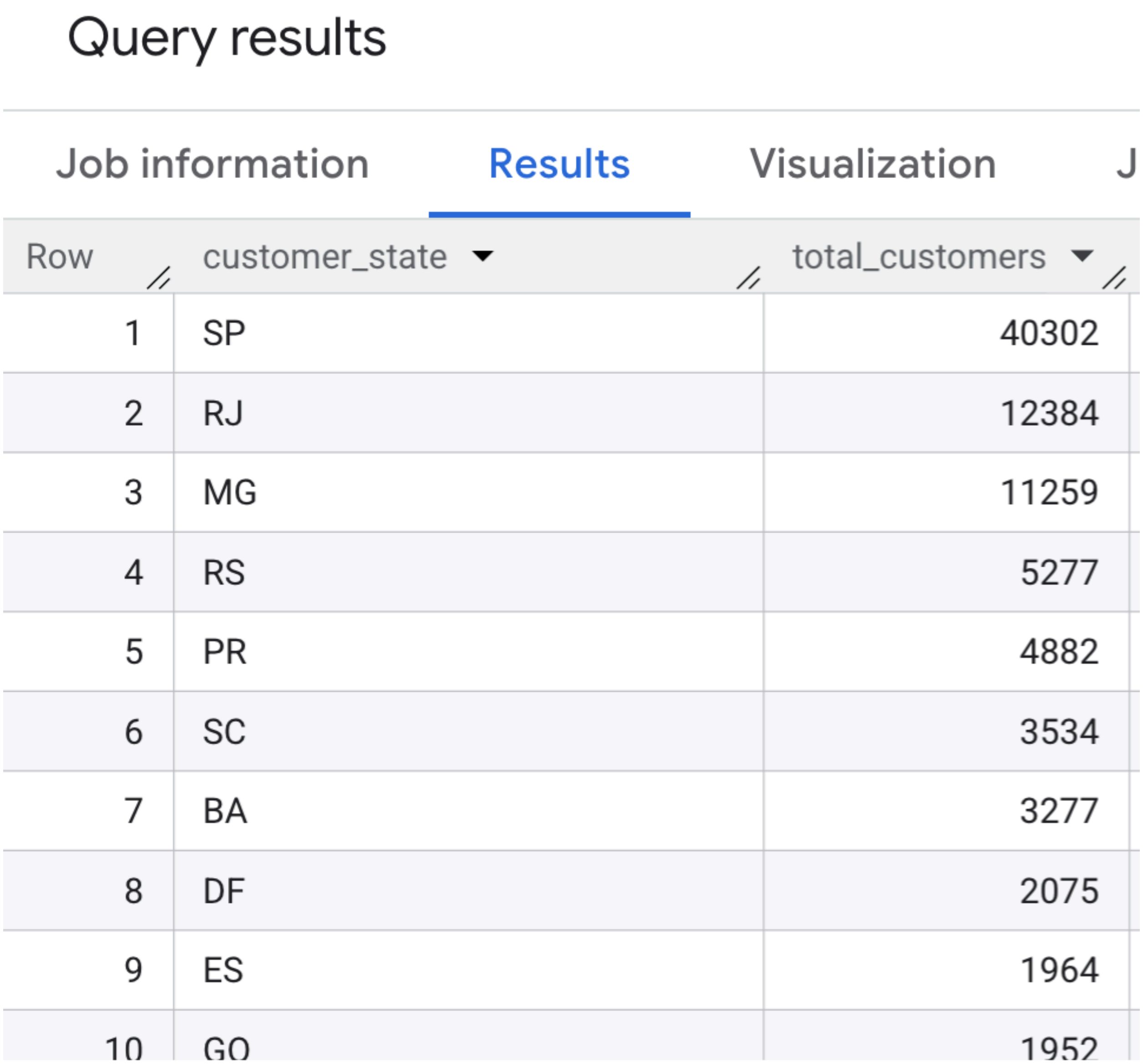
III. Regional Order Analysis (Brazil)

A. Month-on-month orders by state

```
SELECT
c.customer_state,
FORMAT_DATE('%m', o.order_purchase_timestamp) AS month,
COUNT(o.order_id) AS total_orders
FROM `target-449309-449917.Target_Case_study.orders` o
JOIN `target-449309-449917.Target_Case_study.customers` c
ON o.customer_id = c.customer_id
GROUP BY 1,2
ORDER BY 1,2;
```

B. Customer distribution across states

```
SELECT
customer_state,
COUNT(DISTINCT customer_unique_id) AS total_customers
FROM `target-449309-449917.Target_Case_study.customers`
GROUP BY 1
ORDER BY total_customers DESC;
```



Query results

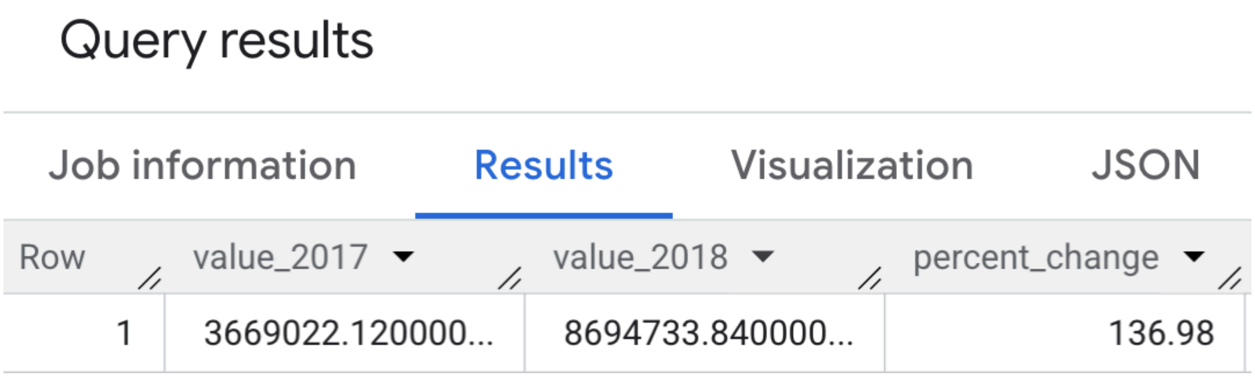
Row	customer_state	total_customers
1	SP	40302
2	RJ	12384
3	MG	11259
4	RS	5277
5	PR	4882
6	SC	3534
7	BA	3277
8	DF	2075
9	ES	1964
10	GO	1952

Insight: Few states contribute a majority of customers.

IV. Economic Impact Analysis

A. % increase in order value from 2017 to 2018 (Jan–Aug)

```
WITH payments_orders AS (
SELECT
p.payment_value,
o.order_purchase_timestamp
FROM `target-449309-449917.Target_Case_study.payments` p
JOIN `target-449309-449917.Target_Case_study.orders` o
ON p.order_id = o.order_id
)
SELECT
SUM(CASE WHEN DATE(order_purchase_timestamp) BETWEEN '2017-01-01' AND
'2017-08-31'
THEN payment_value END) AS value_2017,
SUM(CASE WHEN DATE(order_purchase_timestamp) BETWEEN '2018-01-01' AND
'2018-08-31'
THEN payment_value END) AS value_2018,
ROUND(100 * (
SUM(CASE WHEN DATE(order_purchase_timestamp) BETWEEN '2018-01-01' AND
'2018-08-31'
THEN payment_value END)
-
SUM(CASE WHEN DATE(order_purchase_timestamp) BETWEEN '2017-01-01' AND
'2017-08-31'
THEN payment_value END)
) /
SUM(CASE WHEN DATE(order_purchase_timestamp) BETWEEN '2017-01-01' AND
'2017-08-31'
THEN payment_value END), 2) AS percent_change
FROM payments_orders;
```



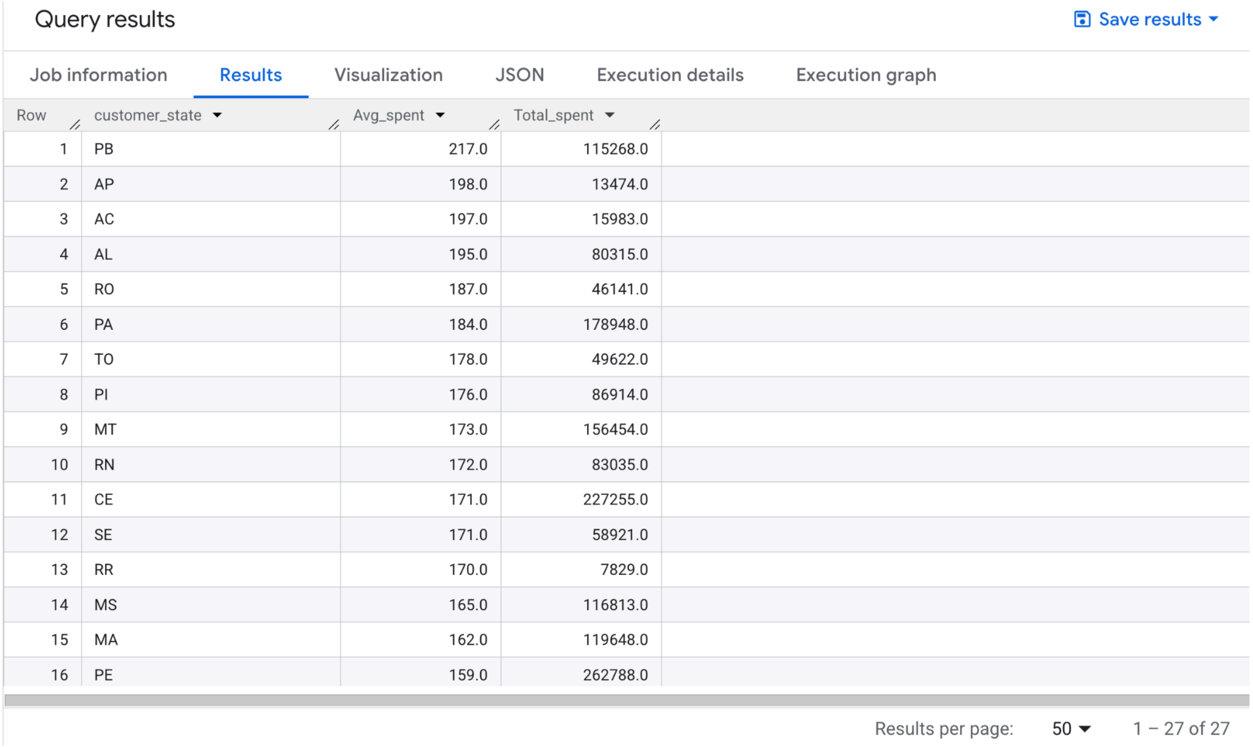
Query results

Row	value_2017	value_2018	percent_change
1	3669022.120000...	8694733.840000...	136.98

Insight: Shows YoY growth in transaction value.

B. Total & average order price by state

```
SELECT
C.customer_state,
round(Sum(price)/count(distinct O.order_id)) as Avg_spent,
round(sum(price)) as Total_spent
FROM
`target-449309-449917.Target_Case_study.orders` O
join
`target-449309-449917.Target_Case_study.customers` C
on O.customer_id = C.customer_id
JOIN
`target-449309-449917.Target_Case_study.order_items` OI
ON O.order_id = OI.order_id

group by 1
order by 2 desc,3 desc
```

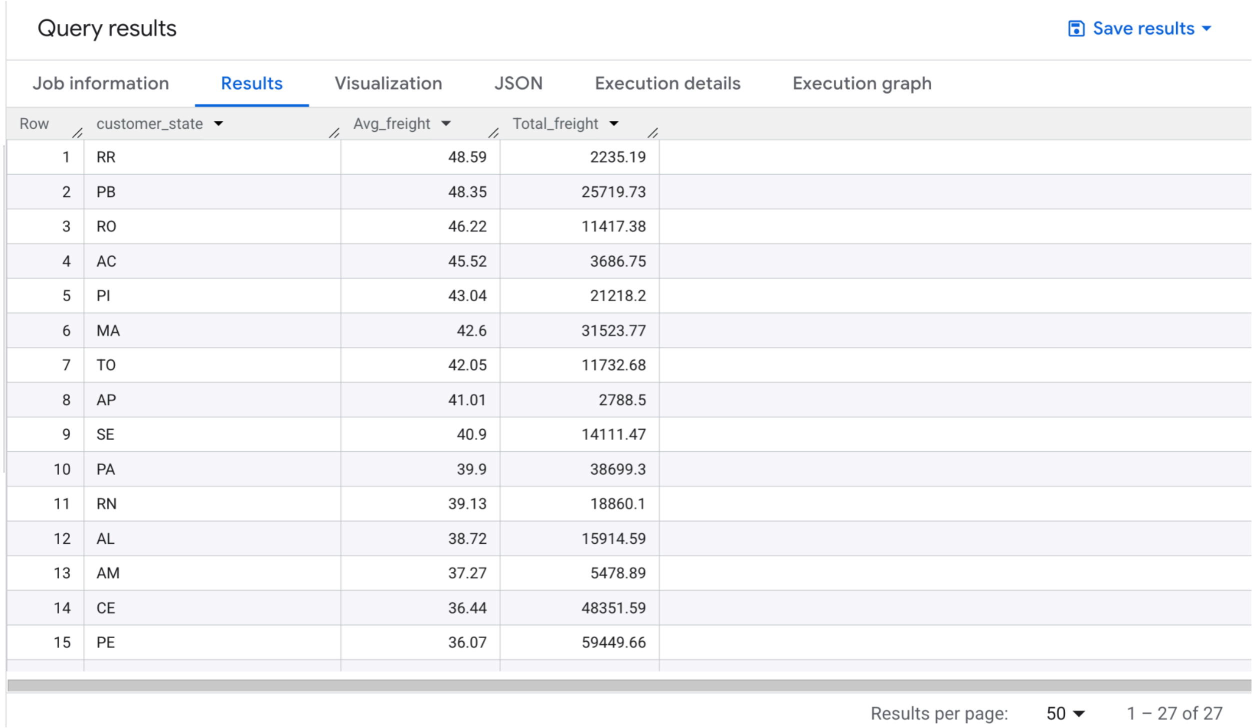


Row	customer_state	Avg_spent	Total_spent
1	PB	217.0	115268.0
2	AP	198.0	13474.0
3	AC	197.0	15983.0
4	AL	195.0	80315.0
5	RO	187.0	46141.0
6	PA	184.0	178948.0
7	TO	178.0	49622.0
8	PI	176.0	86914.0
9	MT	173.0	156454.0
10	RN	172.0	83035.0
11	CE	171.0	227255.0
12	SE	171.0	58921.0
13	RR	170.0	7829.0
14	MS	165.0	116813.0
15	MA	162.0	119648.0
16	PE	159.0	262788.0

Results per page: 50 1 – 27 of 27

C. Total & average freight value by state

```
SELECT
c.customer_state,
ROUND(SUM(oi.freight_value) / COUNT(DISTINCT o.order_id), 2) AS Avg_freight,
ROUND(SUM(oi.freight_value), 2) AS Total_freight
FROM
`target-449309-449917.Target_Case_study.orders` O
join
`target-449309-449917.Target_Case_study.customers` C
on O.customer_id = C.customer_id
JOIN
`target-449309-449917.Target_Case_study.order_items` OI
ON O.order_id = OI.order_id
GROUP BY 1
ORDER BY 2 desc;
```

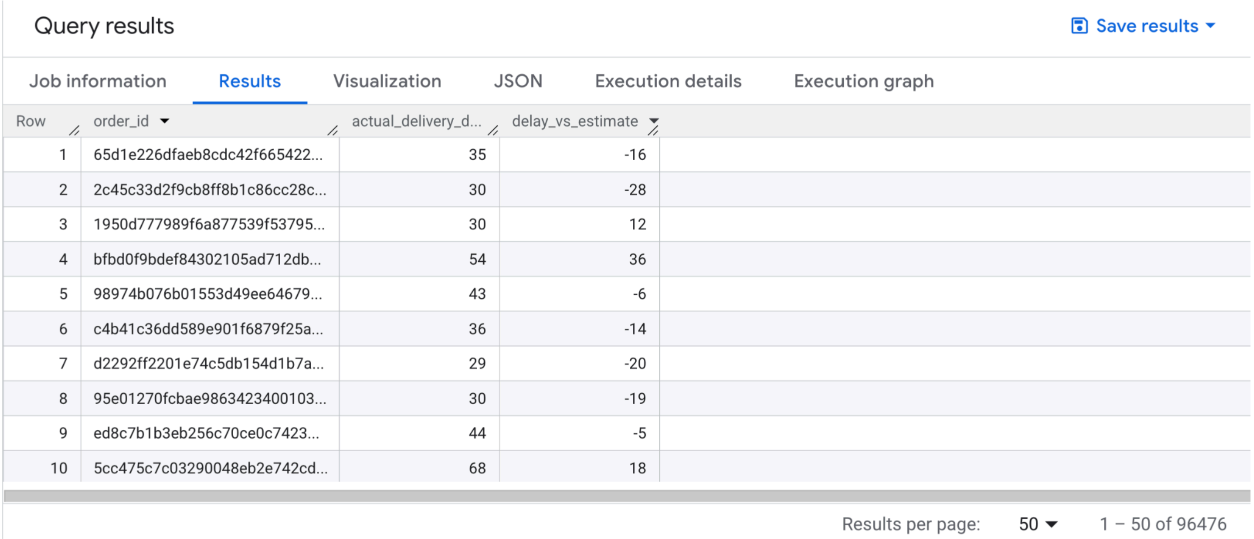


Row	customer_state	Avg_freight	Total_freight
1	RR	48.59	2235.19
2	PB	48.35	25719.73
3	RO	46.22	11417.38
4	AC	45.52	3686.75
5	PI	43.04	21218.2
6	MA	42.6	31523.77
7	TO	42.05	11732.68
8	AP	41.01	2788.5
9	SE	40.9	14111.47
10	PA	39.9	38699.3
11	RN	39.13	18860.1
12	AL	38.72	15914.59
13	AM	37.27	5478.89
14	CE	36.44	48351.59
15	PE	36.07	59449.66

V. Delivery & Logistics Performance

A. Actual delivery time and delay vs estimate

```
SELECT
order_id,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
actual_delivery_days,
DATE_DIFF(order_delivered_customer_date, order_estimated_delivery_date, DAY)
AS delay_vs_estimate
FROM `target-449309-449917.Target_Case_study.orders`
WHERE order_delivered_customer_date IS NOT NULL;
```

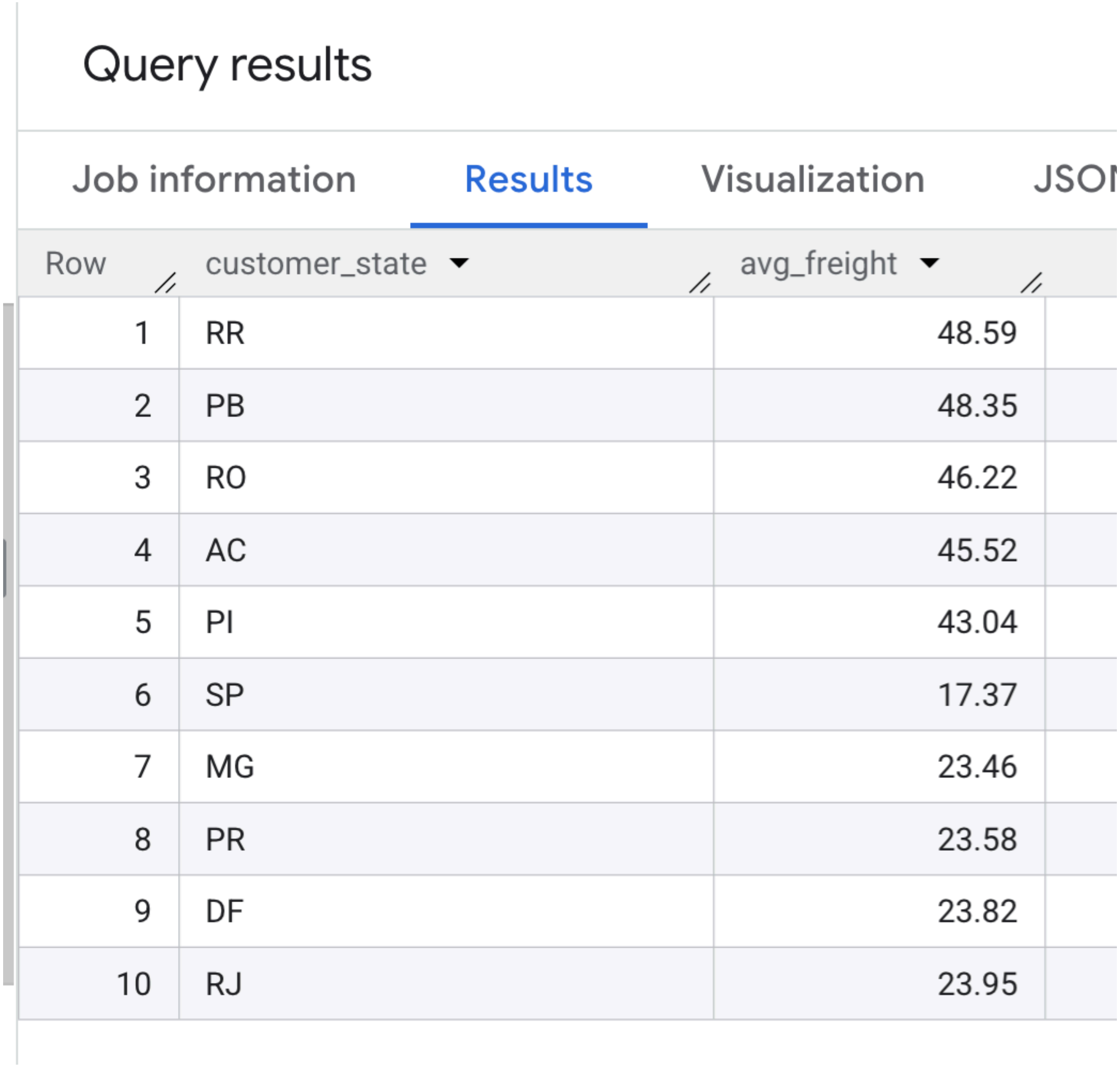


Row	order_id	actual_delivery_d...	delay_vs_estimate
1	65d1e226dfaeb8cdc42f665422...	35	-16
2	2c45c33d2f9cb8ff8b1c86cc28c...	30	-28
3	1950d777989f6a877539f53795...	30	12
4	bfbd0f9bdef84302105ad712db...	54	36
5	98974b076b01553d49ee64679...	43	-6
6	c4b41c36dd589e901f6879f25a...	36	-14
7	d2292ff2201e74c5db154d1b7a...	29	-20
8	95e01270fcbae9863423400103...	30	-19
9	ed8c7b1b3eb256c70ce0c7423...	44	-5
10	5cc475c7c03290048eb2e742cd...	68	18

Results per page: 50 1 – 50 of 96476

B. Top & bottom 5 states by average freight value

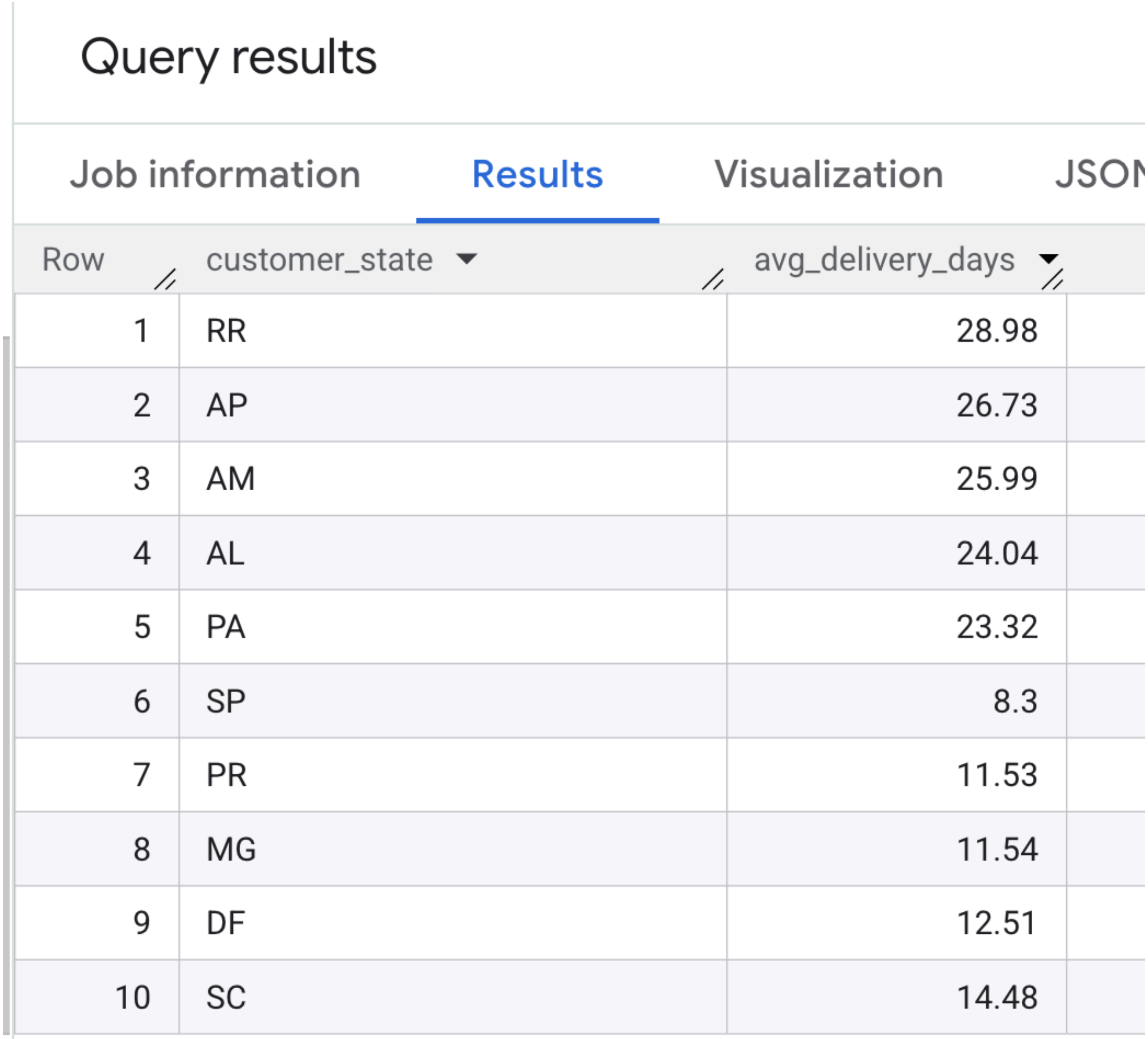
```
WITH freight_avg AS (
 SELECT
   c.customer_state,
   ROUND(SUM(oi.freight_value) / COUNT(DISTINCT o.order_id), 2) AS avg_freight
 FROM `target-449309-449917.Target_Case_study.orders` o
 JOIN `target-449309-449917.Target_Case_study.customers` c
   ON o.customer_id = c.customer_id
 JOIN `target-449309-449917.Target_Case_study.order_items` oi
   ON o.order_id = oi.order_id
 GROUP BY 1
)

SELECT *
FROM (
 SELECT * FROM freight_avg
 ORDER BY avg_freight DESC
 LIMIT 5
)

UNION ALL

SELECT *
FROM (
 SELECT * FROM freight_avg
 ORDER BY avg_freight ASC
 LIMIT 5
);
```



Row	customer_state	avg_freight
1	RR	48.59
2	PB	48.35
3	RO	46.22
4	AC	45.52
5	PI	43.04
6	SP	17.37
7	MG	23.46
8	PR	23.58
9	DF	23.82
10	RJ	23.95

C. Top & bottom 5 states by average delivery time

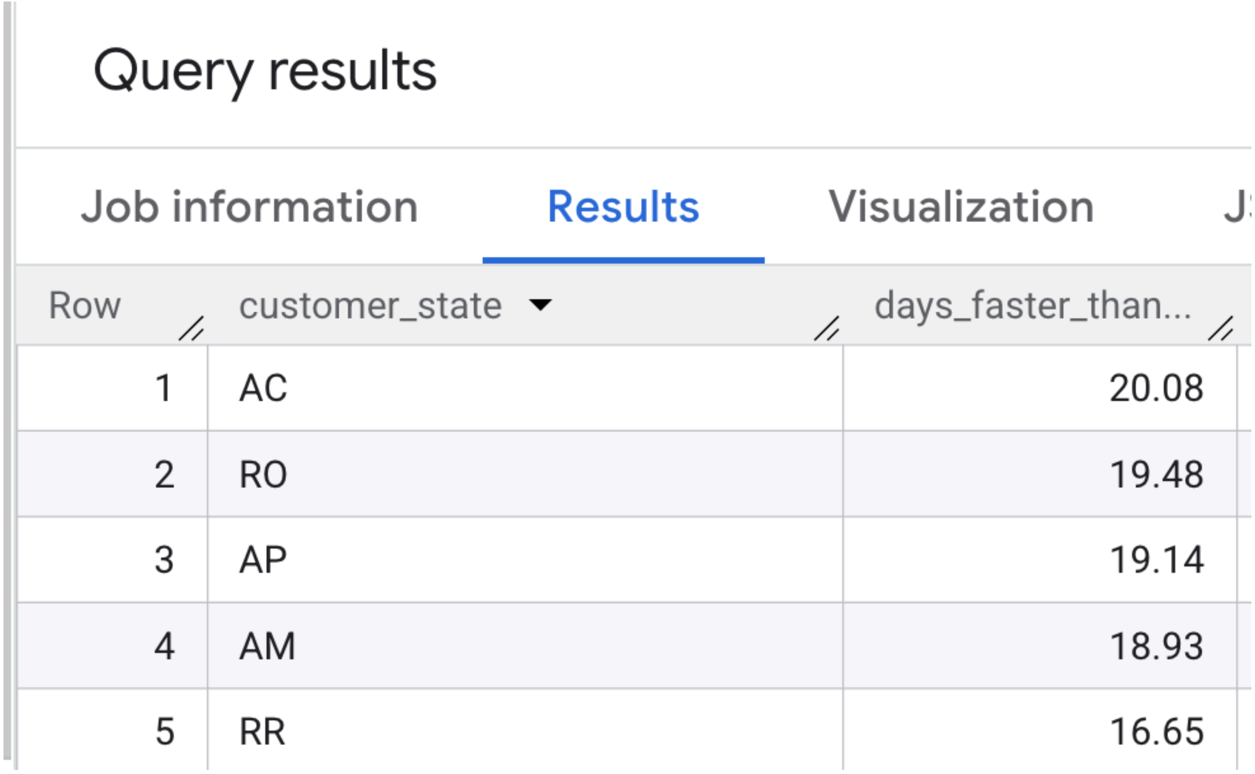
```
WITH delivery_avg AS (
 SELECT c.customer_state,
   ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,
o.order_purchase_timestamp,DAY)),2
   ) AS avg_delivery_days
 FROM `target-449309-449917.Target_Case_study.orders` o
 JOIN `target-449309-449917.Target_Case_study.customers` c
   ON o.customer_id = c.customer_id
 WHERE o.order_delivered_customer_date IS NOT NULL
 GROUP BY 1
)

SELECT *
FROM (
 SELECT * FROM delivery_avg
 ORDER BY avg_delivery_days DESC
 LIMIT 5
)

UNION ALL

SELECT *
FROM (
 SELECT * FROM delivery_avg
 ORDER BY avg_delivery_days ASC
 LIMIT 5
);
```



Row	customer_state	avg_delivery_days
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32
6	SP	8.3
7	PR	11.53
8	MG	11.54
9	DF	12.51
10	SC	14.48

```
D. States with fastest delivery vs estimated date
WITH state_delivery AS (
SELECT
c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,
o.order_purchase_timestamp, DAY)), 2) AS avg_actual,
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,
o.order_purchase_timestamp, DAY)), 2) AS avg_estimated
FROM `target-449309-449917.Target_Case_study.orders` o
```

```
 JOIN `target-449309-449917.Target_Case_study.customers` c
   ON o.customer_id = c.customer_id
WHERE o.order_delivered_customer_date IS NOT NULL
GROUP BY 1
)
SELECT
customer_state,
ROUND(avg_estimated - avg_actual, 2) AS days_faster_than_estimate
FROM state_delivery
ORDER BY days_faster_than_estimate DESC
LIMIT 5;
```

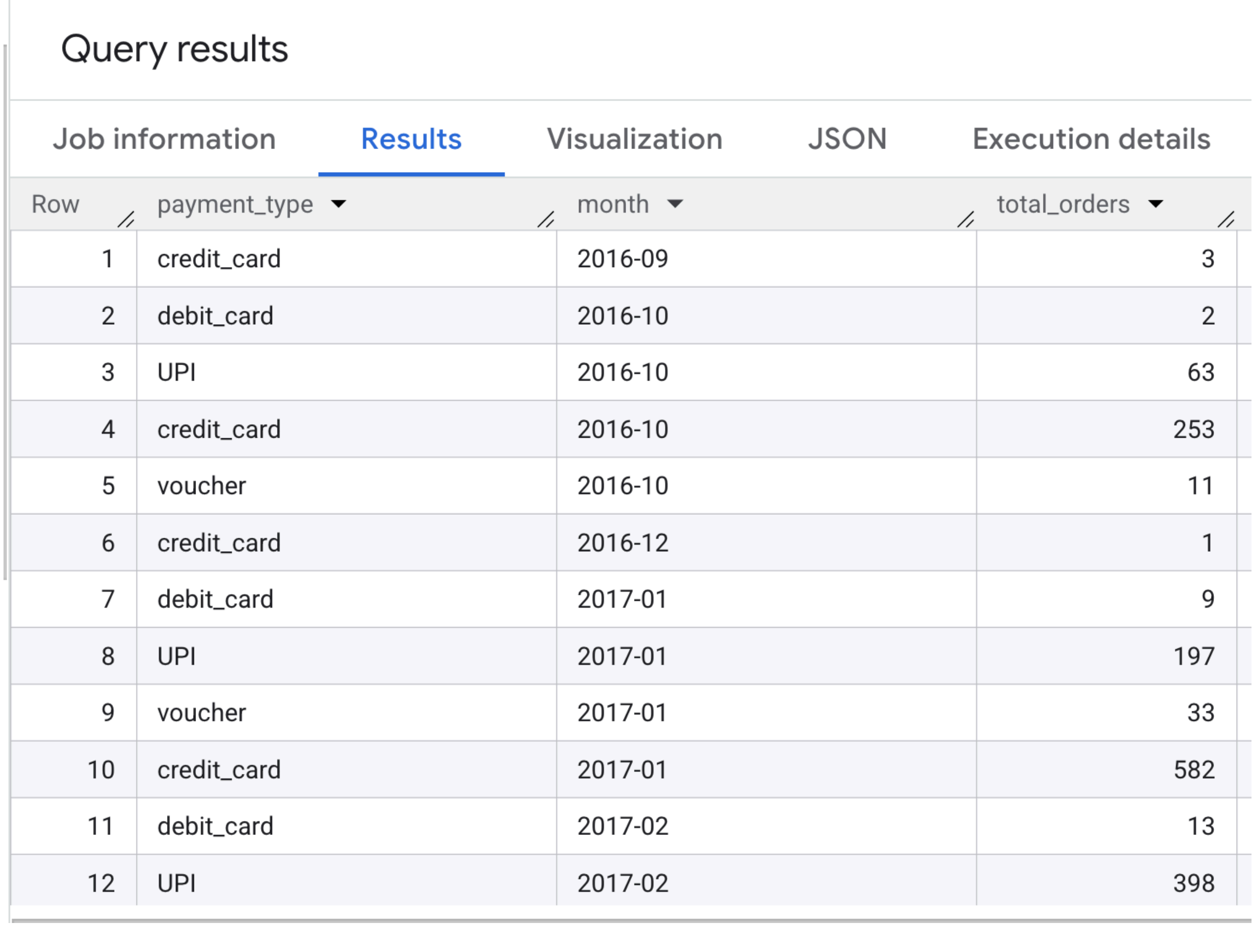


Row	customer_state	days_faster_than...
1	AC	20.08
2	RO	19.48
3	AP	19.14
4	AM	18.93
5	RR	16.65

VI. Payment Analysis

A. Month-on-month orders by payment type

```
SELECT
p.payment_type,
FORMAT_DATE('%Y-%m', o.order_purchase_timestamp) AS month,
COUNT(DISTINCT o.order_id) AS total_orders
FROM `target-449309-449917.Target_Case_study.payments` p
JOIN `target-449309-449917.Target_Case_study.orders` o ON p.order_id = o.order_id
GROUP BY 1,2
ORDER BY 2;
```

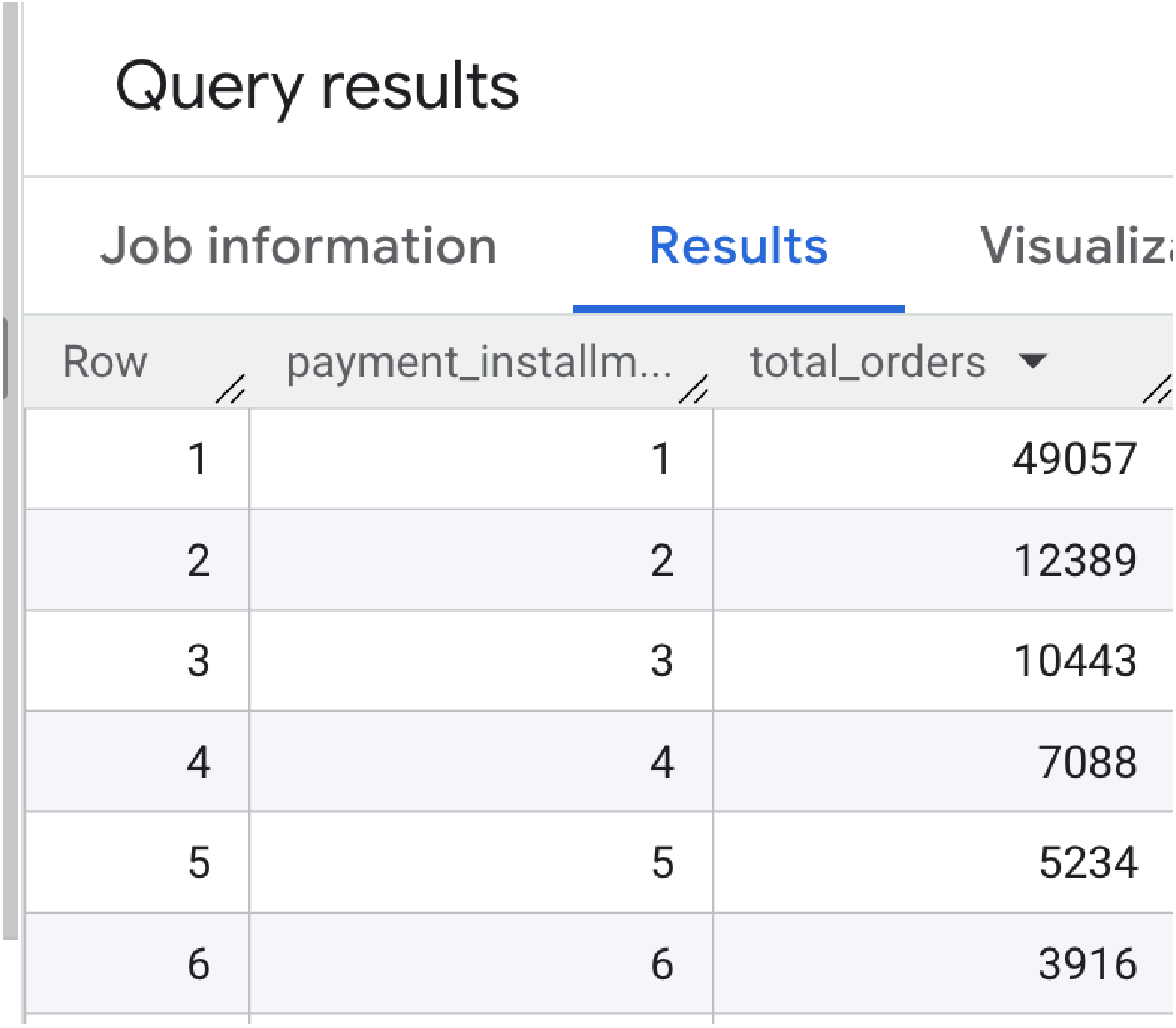


Row	payment_type	month	total_orders
1	credit_card	2016-09	3
2	debit_card	2016-10	2
3	UPI	2016-10	63
4	credit_card	2016-10	253
5	voucher	2016-10	11
6	credit_card	2016-12	1
7	debit_card	2017-01	9
8	UPI	2017-01	197
9	voucher	2017-01	33
10	credit_card	2017-01	582
11	debit_card	2017-02	13
12	UPI	2017-02	398

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

```
SELECT
payment_installments,
COUNT(DISTINCT o.order_id) AS total_orders
FROM `target-449309-449917.Target_Case_study.payments` p
JOIN `target-449309-449917.Target_Case_study.orders` o ON p.order_id = o.order_id
```

```
WHERE payment_installments >= 1
AND payment_value > 0
GROUP BY 1
ORDER BY 1;
Here I am taking payment_installments>=1 and payment_value>0  both purposely because
data is not clean some rows have payment_installments>=1 but payment_value=0
```



Query results

Row	payment_installm...	total_orders
1	1	49057
2	2	12389
3	3	10443
4	4	7088
5	5	5234
6	6	3916