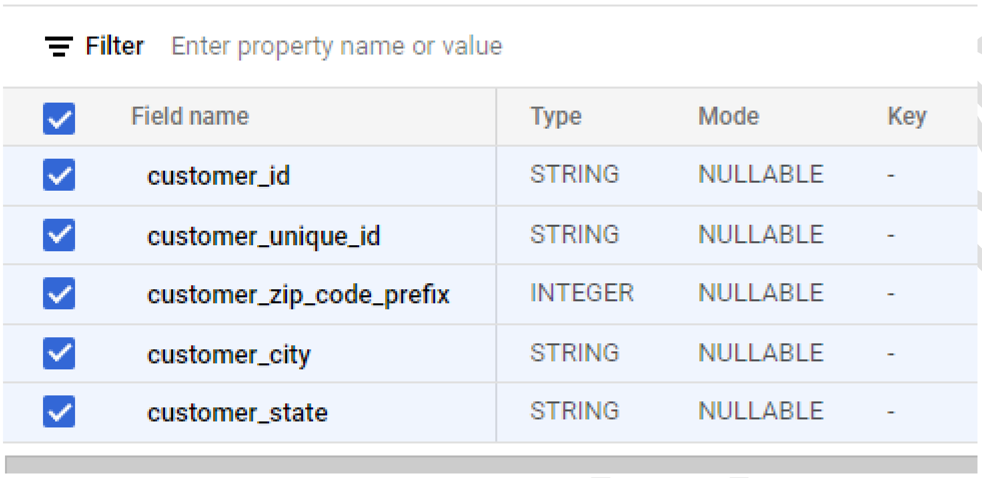
Business Case: Target SQL

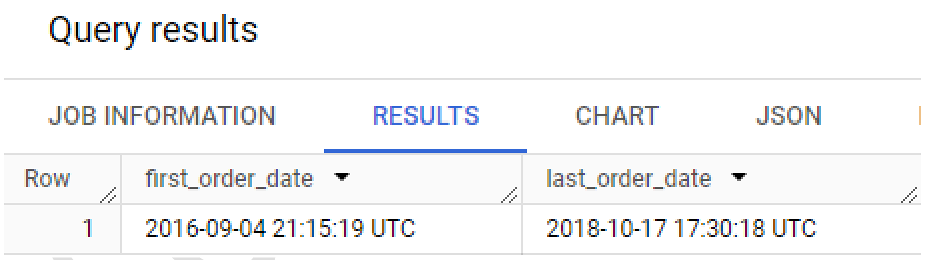
```
# What does 'good' look like?

# 1) Import the dataset and do usual exploratory analysis steps like
#checking the structure & characteristics of the dataset:
# A) Data type of all columns in the "customers" table.
SELECT * FROM `Target.customers`
```



Filter Enter property name or value

Field name	Type	Mode	Key
customer_id	STRING	NULLABLE	-
customer_unique_id	STRING	NULLABLE	-
customer_zip_code_prefix	INTEGER	NULLABLE	-
customer_city	STRING	NULLABLE	-
customer_state	STRING	NULLABLE	-

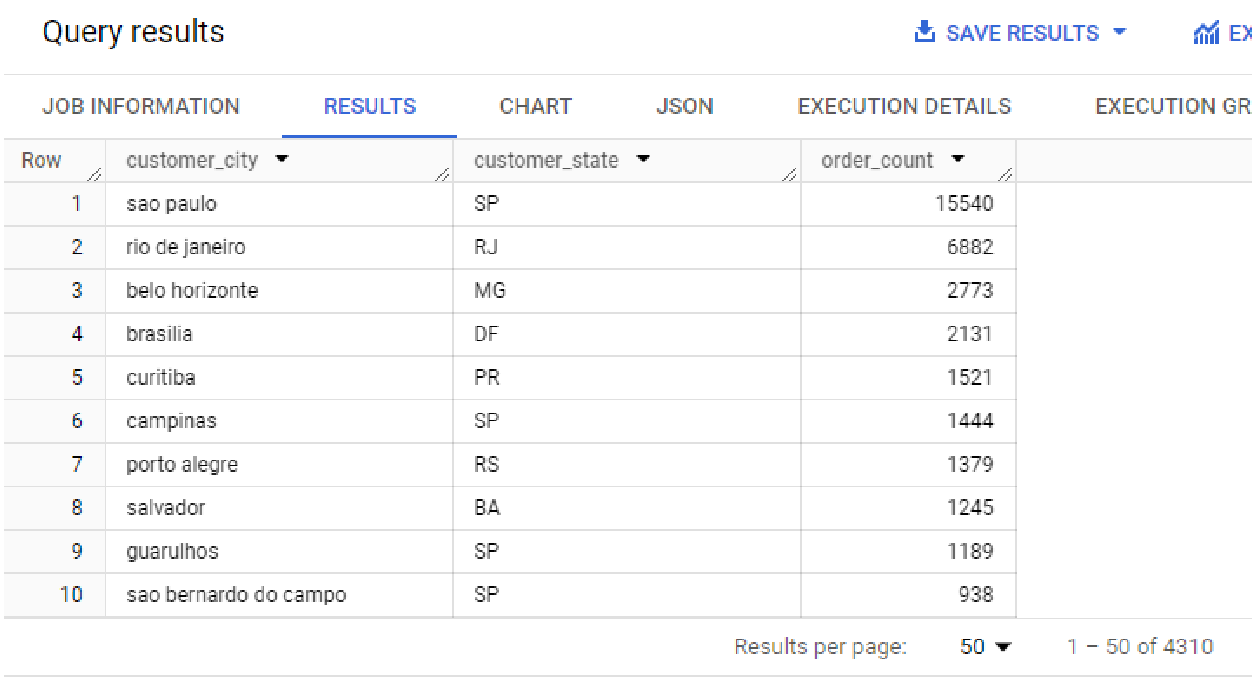
```
# B) Get the time range between which the orders were placed.
SELECT
MIN(order_purchase_timestamp) AS first_order_date,
MAX(order_purchase_timestamp) AS last_order_date,
FROM `Target.orders`
```



Query results

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

```
# C) Count the Cities & States of customers who ordered during the given period.
SELECT DISTINCT c.customer_city, c.customer_state,
COUNT(o.customer_id) AS order_count
FROM `Target.orders` o
JOIN `Target.customers` c
ON o.customer_id = c.customer_id
GROUP BY 1,2
ORDER BY 3 DESC;
```


Query results

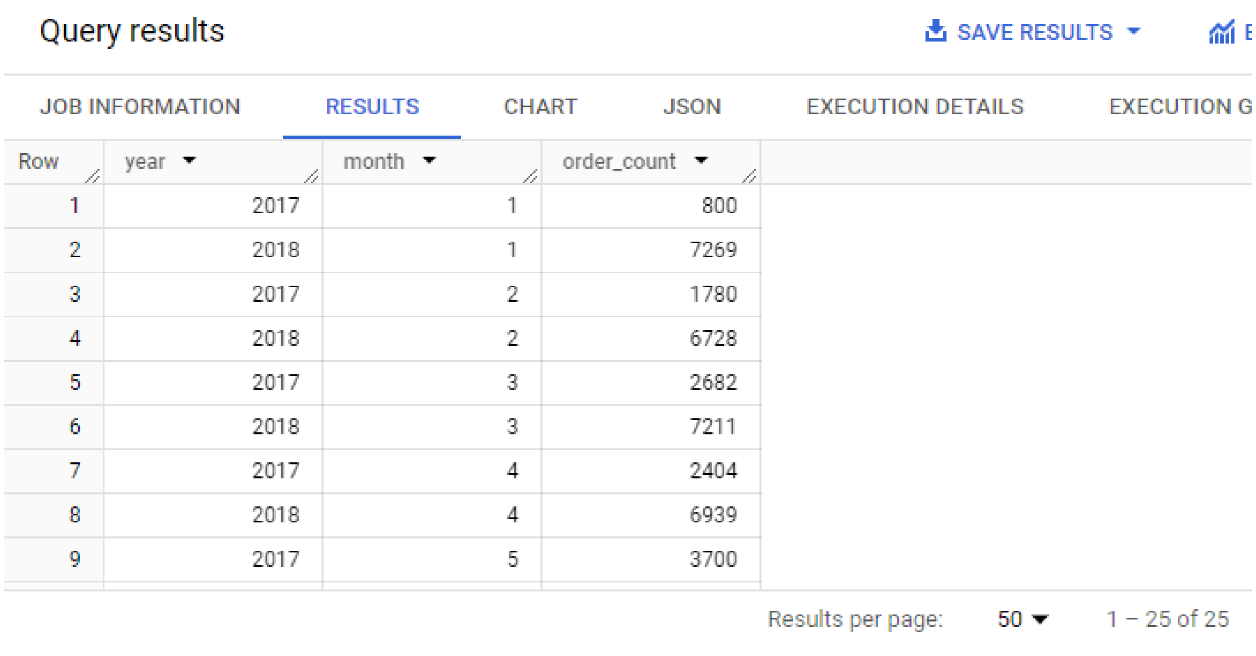
SAVE RESULTS EX

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GR

Row	customer_city	customer_state	order_count
1	sao paulo	SP	15540
2	rio de janeiro	RJ	6882
3	belo horizonte	MG	2773
4	brasilia	DF	2131
5	curitiba	PR	1521
6	campinas	SP	1444
7	porto alegre	RS	1379
8	salvador	BA	1245
9	guarulhos	SP	1189
10	sao bernardo do campo	SP	938

Results per page: 50 1 – 50 of 4310

```
# 2) In-depth Exploration:
# 1) Is there a growing trend in the no. of orders placed over the past years?
SELECT
EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
COUNT(DISTINCT o.order_id) AS order_count
FROM `Target.orders` o
JOIN `Target.customers` c
ON o.customer_id = c.customer_id
GROUP BY month, year
ORDER BY month, year;
```


Query results

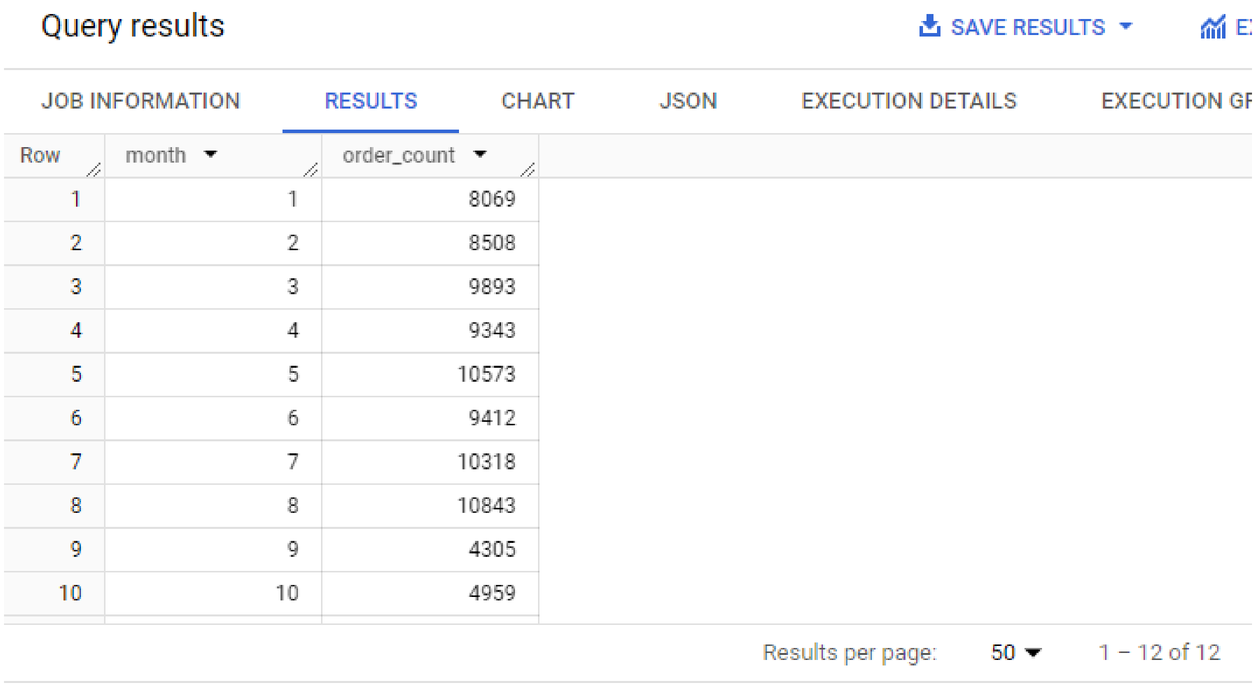
SAVE RESULTS E

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION G

Row	year	month	order_count
1	2017	1	800
2	2018	1	7269
3	2017	2	1780
4	2018	2	6728
5	2017	3	2682
6	2018	3	7211
7	2017	4	2404
8	2018	4	6939
9	2017	5	3700

Results per page: 50 1 – 25 of 25

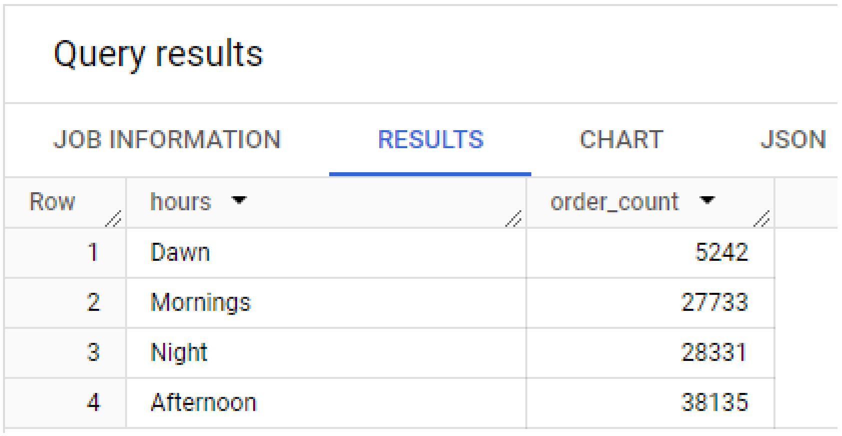
```
# 2) Can we see some kind of monthly seasonality in terms of the no. of orders being placed?
SELECT EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
COUNT(DISTINCT order_id) AS order_count
FROM Target.orders
GROUP BY month
ORDER BY month;
```



Row	month	order_count
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

Results per page: 50 1 – 12 of 12

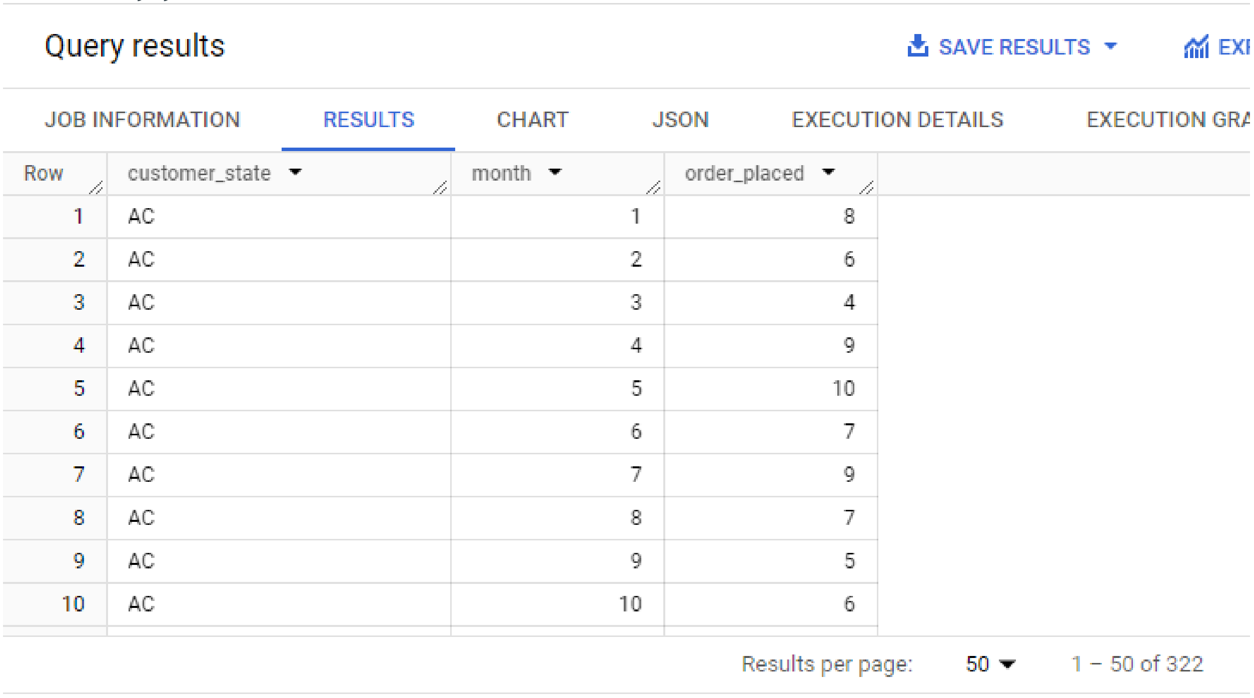
```
# 3) During what time of the day, do the Brazilian customers mostly place their orders? (Dawn,
Morning, Afternoon or Night)
# 0-6 hrs : Dawn
# 7-12 hrs : Mornings
# 13-18 hrs : Afternoon
# 19-23 hrs : Night

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 'Mornings'
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 hours,
COUNT(o.order_id) AS order_count
FROM `Target.orders` o
JOIN `Target.customers` c
ON o.customer_id = c.customer_id
GROUP BY hours
ORDER BY order_count;
```


Query results

JOB INFORMATION RESULTS CHART JSON

Row	hours	order_count
1	Dawn	5242
2	Mornings	27733
3	Night	28331
4	Afternoon	38135

```
# 3) Evolution of E-commerce orders in the Brazil region:
# 1) Get the month on month no. of orders placed in each state.

SELECT c.customer_state,
EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
COUNT(order_purchase_timestamp) AS order_placed
FROM `Target.orders` o
JOIN `Target.customers` c
ON o.customer_id = c.customer_id
GROUP BY 1,2
ORDER BY 1,2;
```

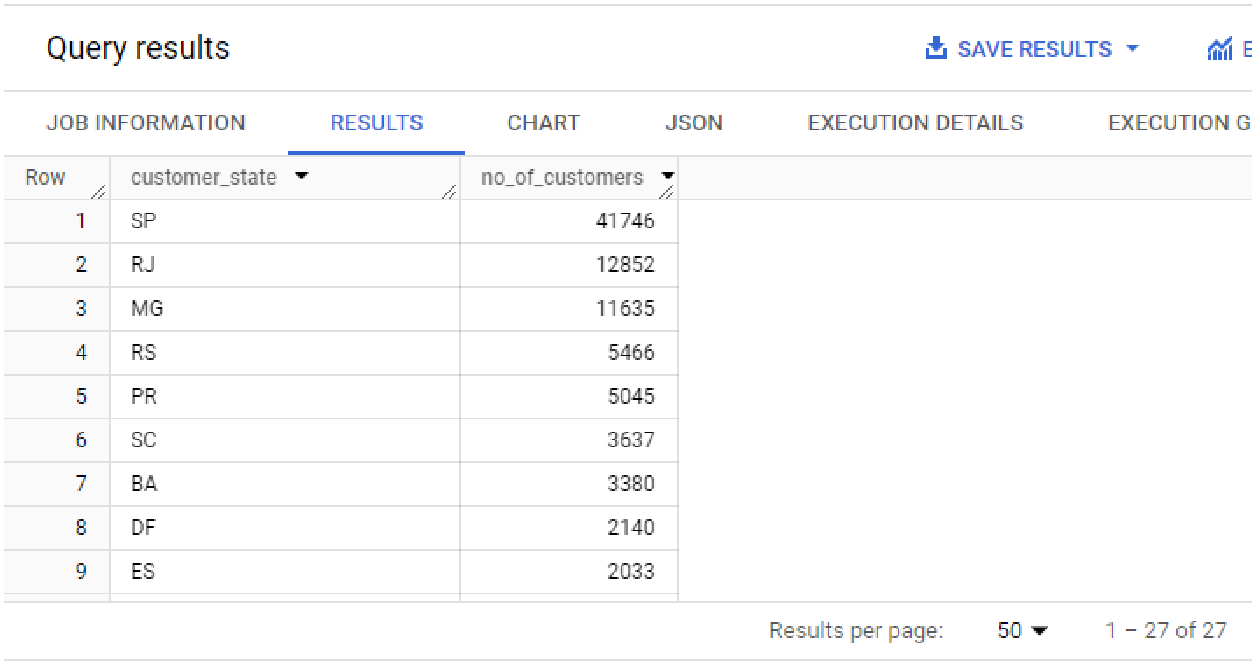

Query results

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRA

Row	customer_state	month	order_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

Results per page: 50 1 – 50 of 322

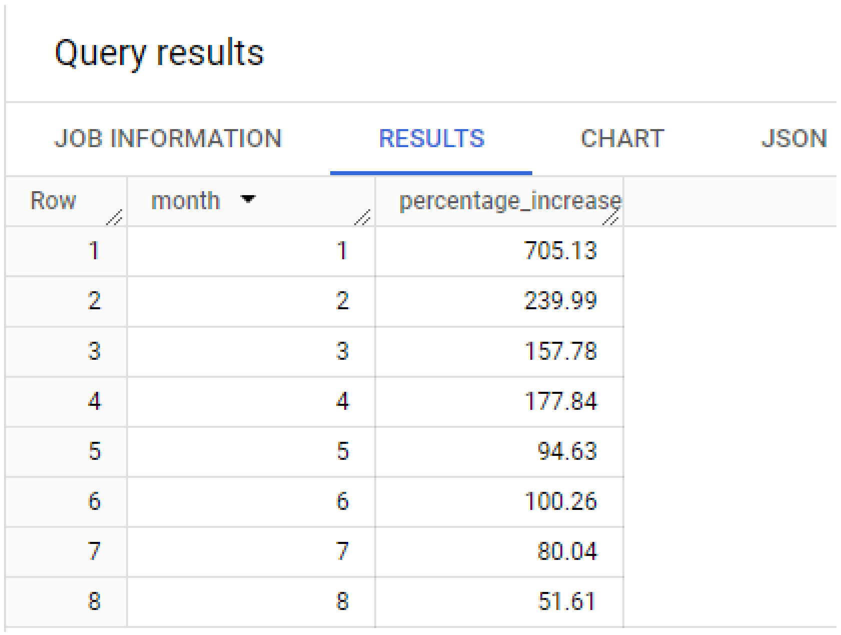
```
# 2) How are the customers distributed across all the states?
SELECT customer_state,
COUNT(customer_id) AS no_of_customers
FROM `Target.customers`
GROUP BY customer_state
ORDER BY no_of_customers DESC;
```



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

Results per page: 50 1 – 27 of 27

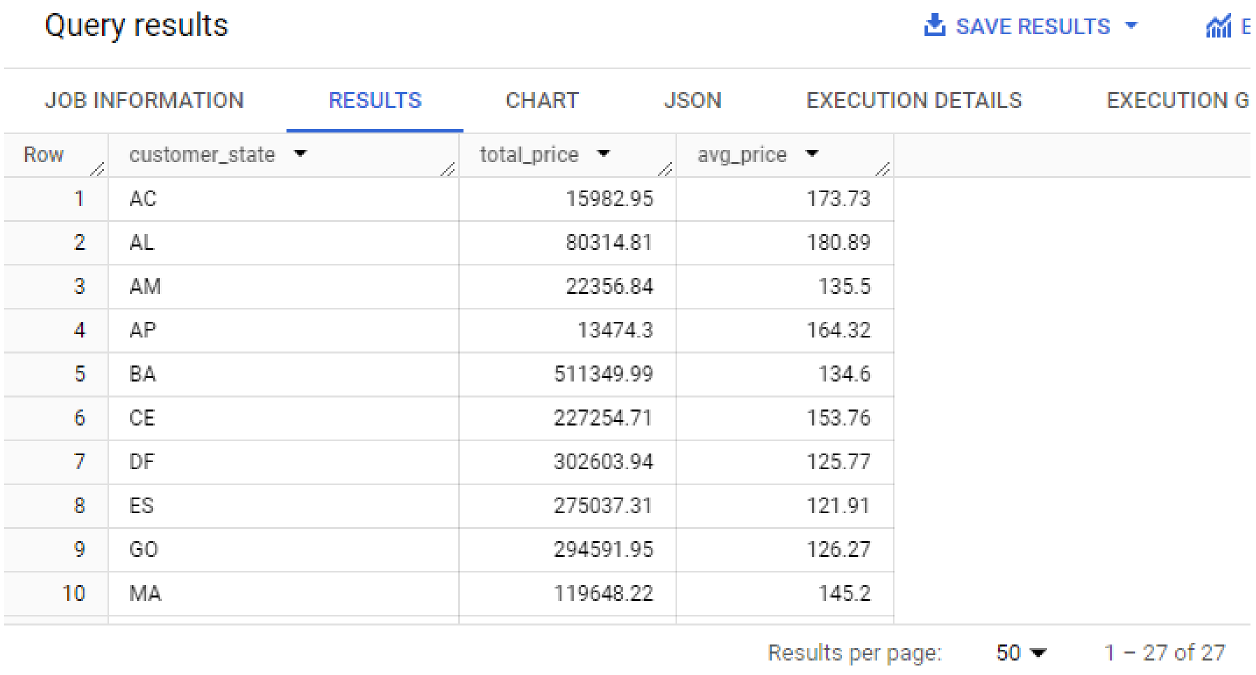
```
# 4) Impact on Economy: Analyze the money movement by e-commerce by looking at order prices,
freight and others.
# 1) Get the % increase in the cost of orders from year 2017 to 2018 (include months between
Jan to Aug only).
#    You can use the "payment_value" column in the payments table to get the cost of orders.

  SELECT EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
  ROUND((
  (SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2018 AND EXTRACT (MONTH FROM
o.order_purchase_timestamp)
   BETWEEN 1 AND 8 THEN p.payment_value END) -

   SUM(CASE WHEN EXTRACT (YEAR FROM o.order_purchase_timestamp) = 2017 AND EXTRACT (MONTH FROM
o.order_purchase_timestamp)
   BETWEEN 1 AND 8 THEN p.payment_value END)) /

   SUM(CASE WHEN EXTRACT (YEAR FROM o.order_purchase_timestamp) = 2017 AND EXTRACT (MONTH FROM
o.order_purchase_timestamp)
   BETWEEN 1 AND 8 THEN p.payment_value END) * 100),2) AS percentage_increase

   FROM `Target.orders` o
   JOIN `Target.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 1
   ORDER BY 1;
```


Query results

JOB INFORMATION RESULTS CHART JSON

Row	month	percentage_increase
1	1	705.13
2	2	239.99
3	3	157.78
4	4	177.84
5	5	94.63
6	6	100.26
7	7	80.04
8	8	51.61

```
# 2) Calculate the Total & Average value of order price for each state.
SELECT c.customer_state,
ROUND(SUM(oi.price),2) AS total_price,
ROUND(AVG(oi.price),2) AS avg_price,
FROM `Target.orders` o
JOIN `Target.order_items` oi
ON o.order_id = oi.order_id
JOIN `Target.customers` c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY c.customer_state;
```


Query results

SAVE RESULTS

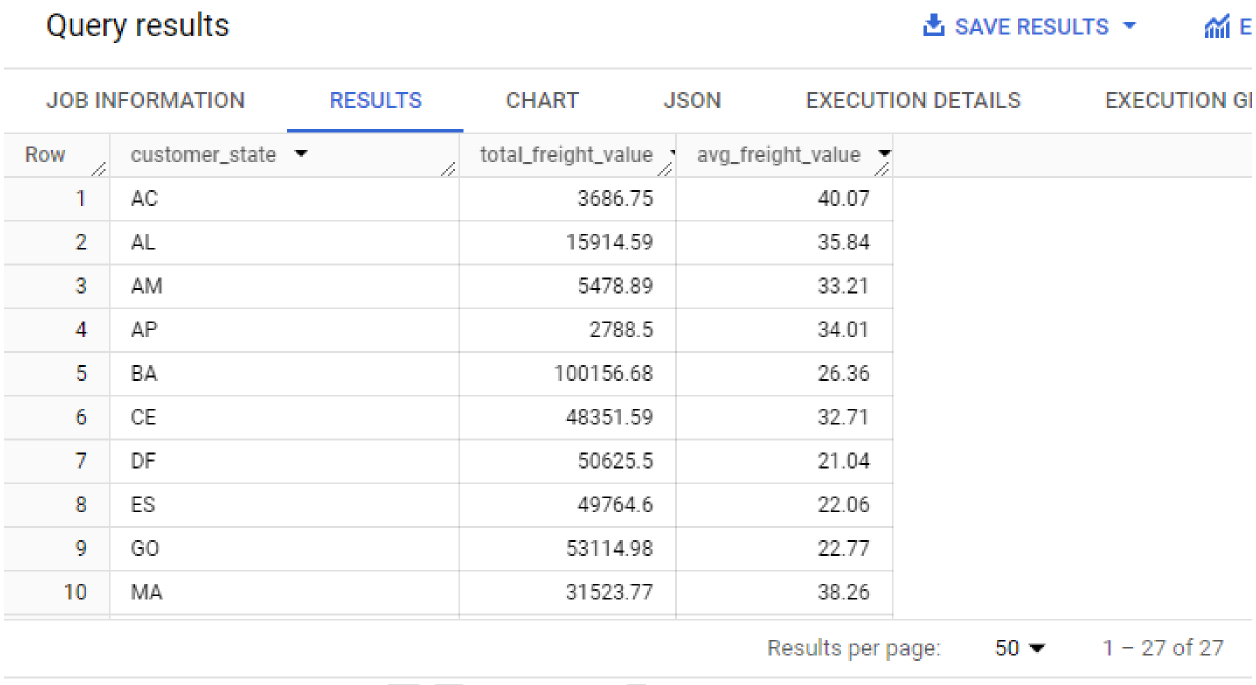
JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION G

Row	customer_state	total_price	avg_price
1	AC	15982.95	173.73
2	AL	80314.81	180.89
3	AM	22356.84	135.5
4	AP	13474.3	164.32
5	BA	511349.99	134.6
6	CE	227254.71	153.76
7	DF	302603.94	125.77
8	ES	275037.31	121.91
9	GO	294591.95	126.27
10	MA	119648.22	145.2

Results per page: 50 1 – 27 of 27

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

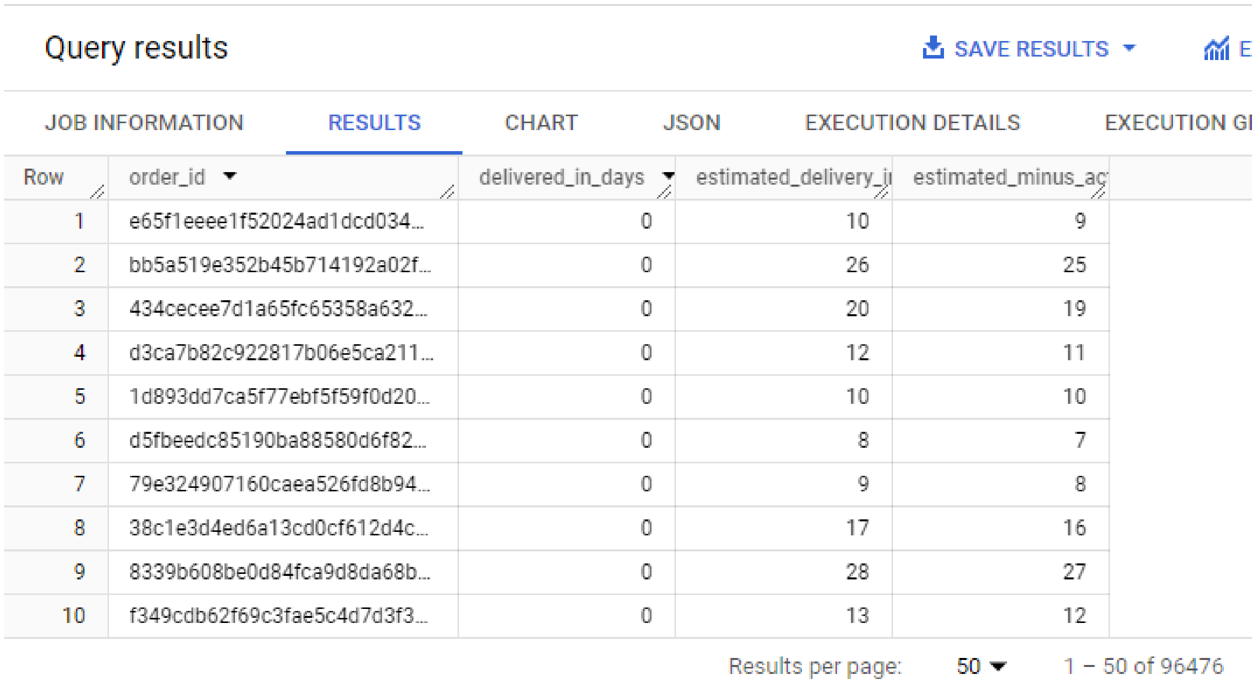
```
SELECT c.customer_state,
ROUND(SUM(oi.freight_value),2) AS total_freight_value,
ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
FROM `Target.orders` o
JOIN `Target.order_items` oi
ON o.order_id = oi.order_id
JOIN `Target.customers` c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY c.customer_state;
```



Row	customer_state	total_freight_value	avg_freight_value
1	AC	3686.75	40.07
2	AL	15914.59	35.84
3	AM	5478.89	33.21
4	AP	2788.5	34.01
5	BA	100156.68	26.36
6	CE	48351.59	32.71
7	DF	50625.5	21.04
8	ES	49764.6	22.06
9	GO	53114.98	22.77
10	MA	31523.77	38.26

Results per page: 50 1 – 27 of 27

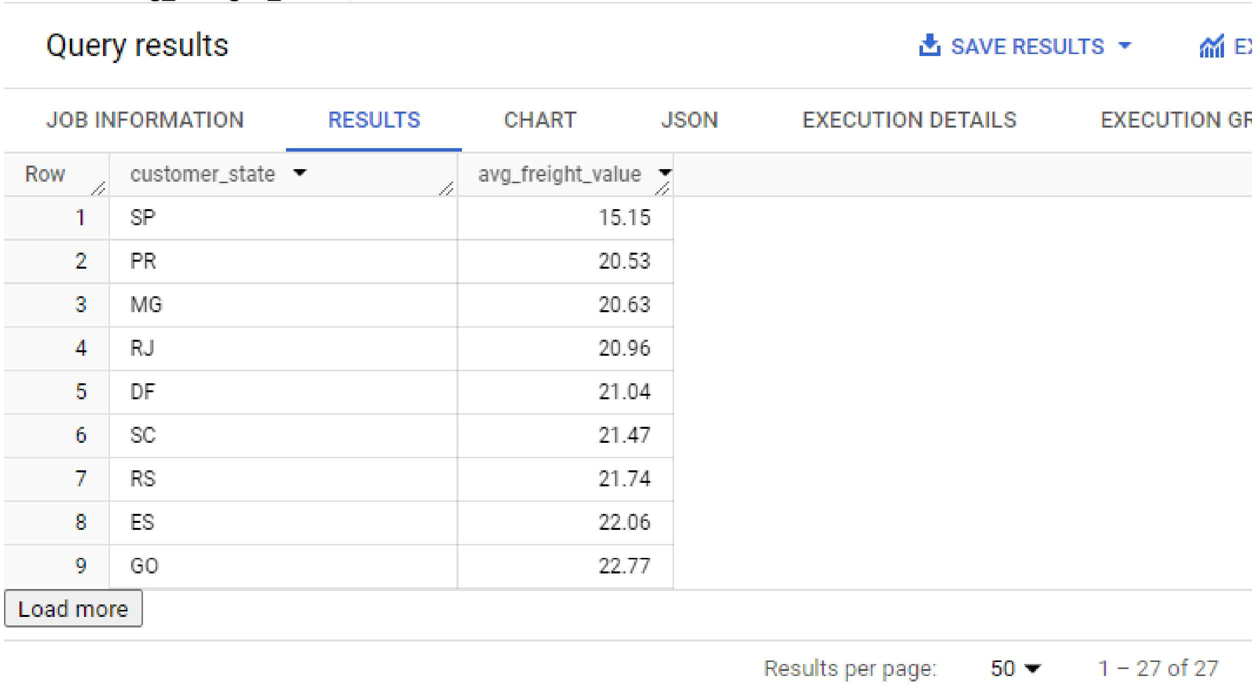
```
# 5) Analysis based on sales, freight and delivery time.
# 1) Find the no. of days taken to deliver each order from the order’s purchase date as
delivery time.
# Also, calculate the difference (in days) between the estimated & actual delivery date of an
order.
# Do this in a single query.
# You can 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

SELECT order_id,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS delivered_in_days,
DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY) AS
estimated_delivery_in_days,
DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS
estimated_minus_actual_delivery_days,
FROM `Target.orders`
WHERE DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) IS NOT NULL
ORDER BY delivered_in_days;
```



Row	order_id	delivered_in_days	estimated_delivery_i	estimated_minus_ac
1	e65f1eeee1f52024ad1dcd034...	0	10	9
2	bb5a519e352b45b714192a02f...	0	26	25
3	434cecee7d1a65fc65358a632...	0	20	19
4	d3ca7b82c922817b06e5ca211...	0	12	11
5	1d893dd7ca5f77ebf5f59f0d20...	0	10	10
6	d5fbeedc85190ba88580d6f82...	0	8	7
7	79e324907160caea526fd8b94...	0	9	8
8	38c1e3d4ed6a13cd0cf612d4c...	0	17	16
9	8339b608be0d84fca9d8da68b...	0	28	27
10	f349cdb62f69c3fae5c4d7d3f3...	0	13	12

Results per page: 50 1 – 50 of 96476

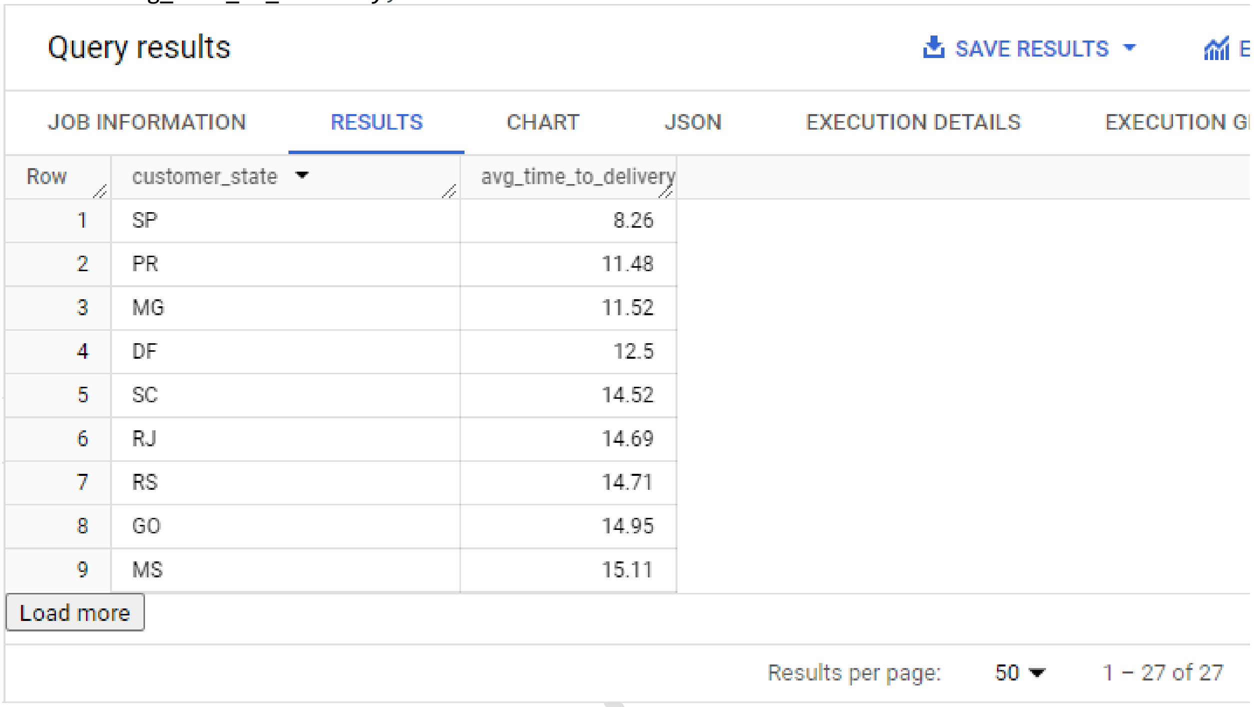
```
# 2) Find out the top 5 states with the highest & lowest average freight value.
SELECT c.customer_state,
ROUND(AVG(i.freight_value), 2) AS avg_freight_value,
FROM `Target.orders` o
JOIN `Target.order_items` i
ON o.order_id = i.order_id
JOIN `Target.customers` c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_freight_value;
```



Row	customer_state	avg_freight_value
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04
6	SC	21.47
7	RS	21.74
8	ES	22.06
9	GO	22.77

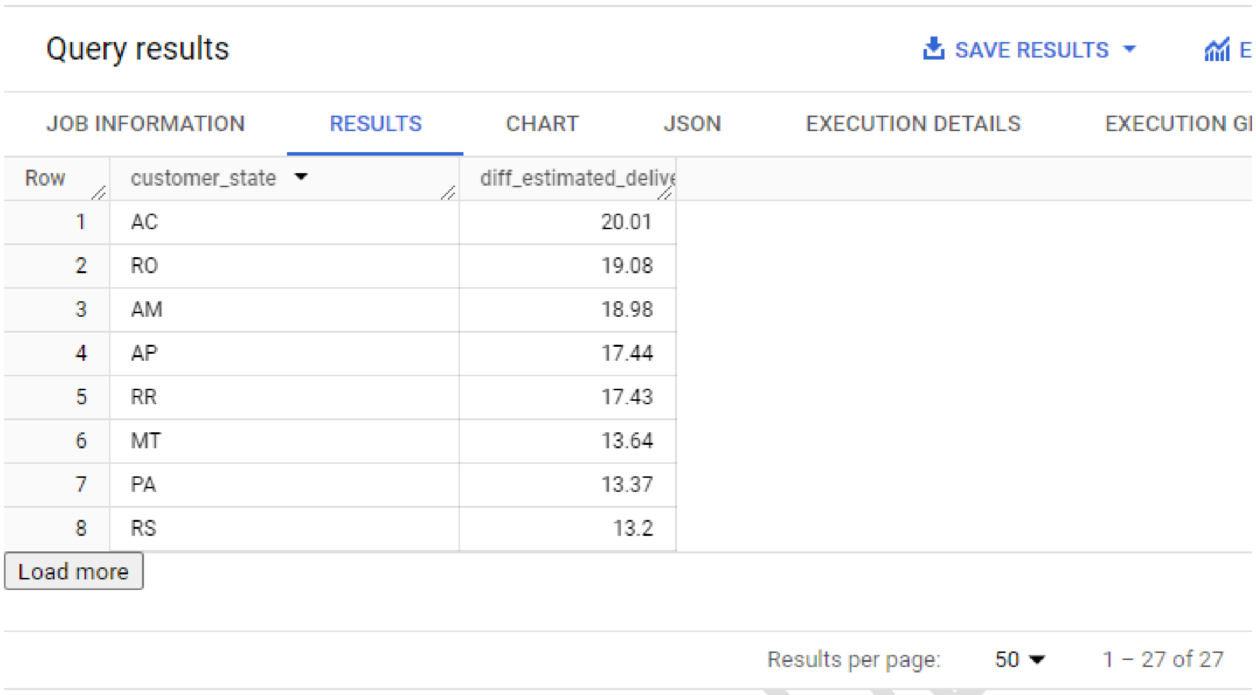
Results per page: 50 1 – 27 of 27

```
# 3) Find out the top 5 states with the highest & lowest average delivery time.
SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date, order_purchase_timestamp, DAY)), 2) AS
avg_time_to_delivery,
FROM `Target.orders` o
JOIN `Target.order_items` i
ON o.order_id = i.order_id
JOIN `Target.customers` c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_time_to_delivery;
```



Row	customer_state	avg_time_to_delivery
1	SP	8.26
2	PR	11.48
3	MG	11.52
4	DF	12.5
5	SC	14.52
6	RJ	14.69
7	RS	14.71
8	GO	14.95
9	MS	15.11

```
# 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.
SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, DAY)), 2)
AS diff_estimated_delivery
FROM `Target.orders` o
JOIN `Target.order_items` i
ON o.order_id = i.order_id
JOIN `Target.customers` c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY diff_estimated_delivery DESC;
```


Query results

SAVE RESULTS

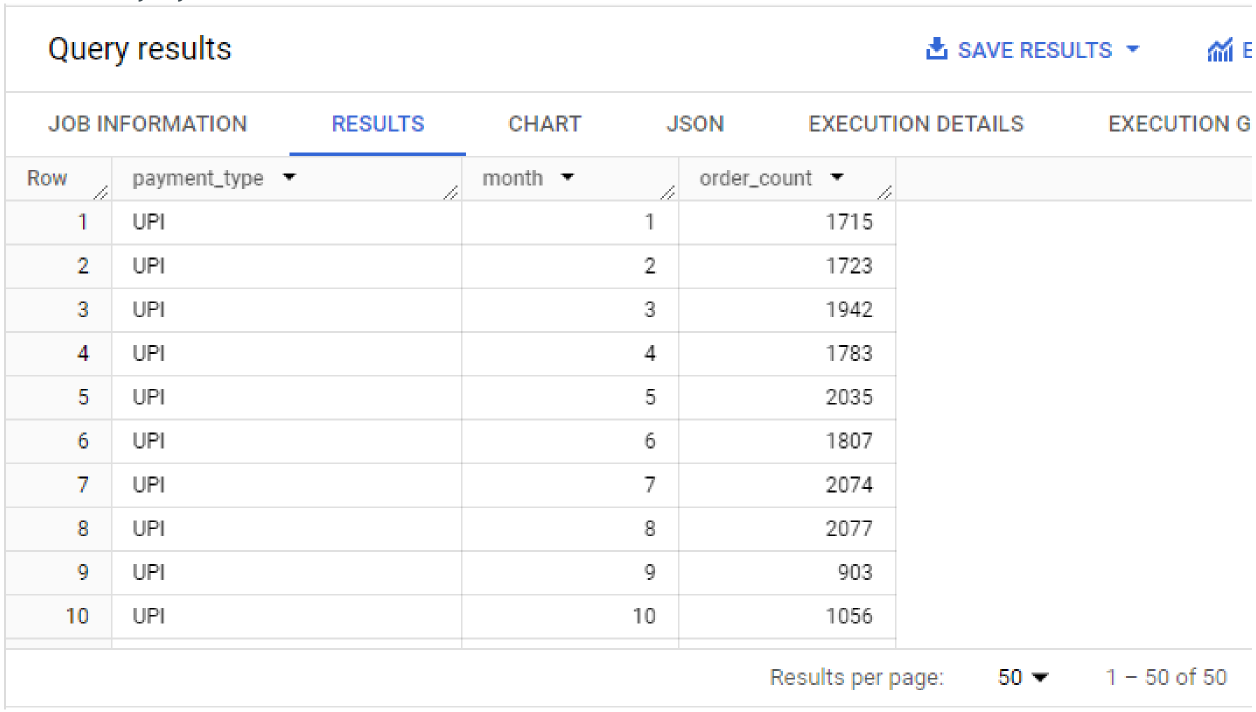
JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION GI

Row	customer_state	diff_estimated_delive
1	AC	20.01
2	RO	19.08
3	AM	18.98
4	AP	17.44
5	RR	17.43
6	MT	13.64
7	PA	13.37
8	RS	13.2

Load more

Results per page: 50 1 – 27 of 27

```
# 6) Analysis based on the payments:
# Find the month on month no. of orders placed using different payment types.
SELECT p.payment_type,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
COUNT(DISTINCT o.order_id) AS order_count
FROM `Target.orders` o
JOIN `Target.payments` p
ON o.order_id = p.order_id
GROUP BY 1, 2
ORDER BY 1, 2;
```


Query results

SAVE RESULTS

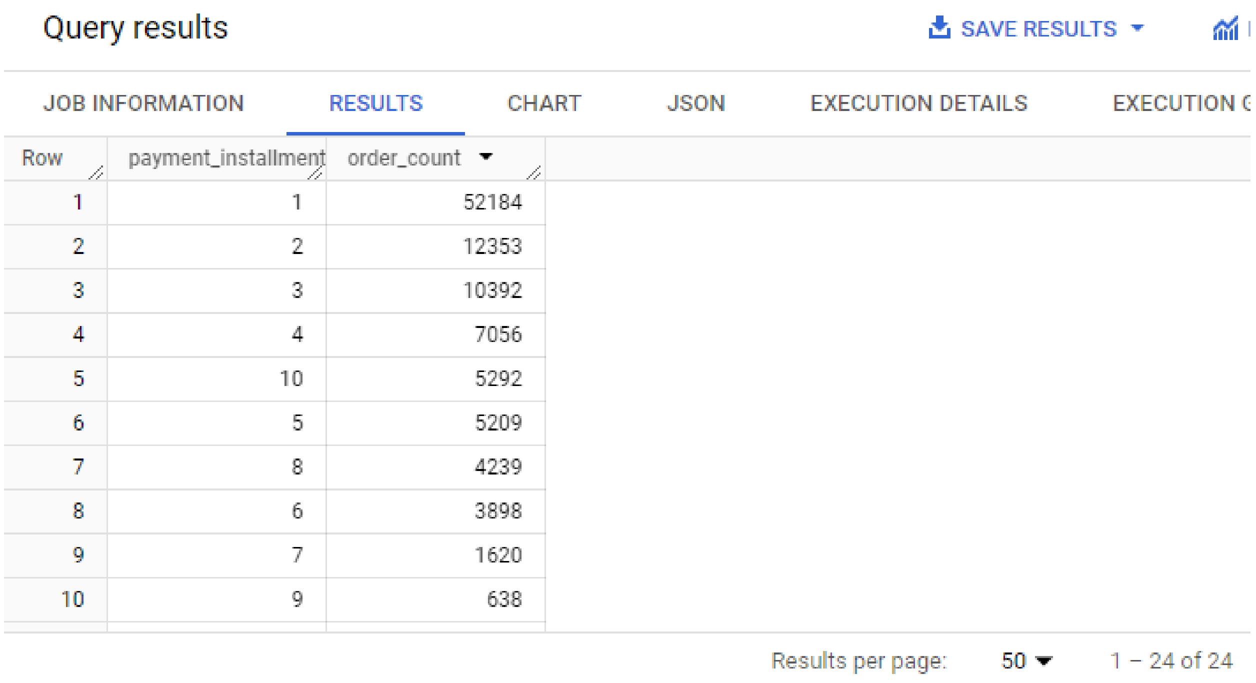
JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS | EXECUTION G

Row	payment_type	month	order_count
1	UPI	1	1715
2	UPI	2	1723
3	UPI	3	1942
4	UPI	4	1783
5	UPI	5	2035
6	UPI	6	1807
7	UPI	7	2074
8	UPI	8	2077
9	UPI	9	903
10	UPI	10	1056

Results per page: 50 1 – 50 of 50

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

```sql
SELECT p.payment_installments,
COUNT(o.order_id) AS order_count
FROM `Target.orders` o
JOIN `Target.payments` p
ON o.order_id = p.order_id
WHERE o.order_status != 'canceled'
GROUP BY p.payment_installments
ORDER BY order_count DESC;
```



Query results

Row	payment_installment	order_count
1	1	52184
2	2	12353
3	3	10392
4	4	7056
5	10	5292
6	5	5209
7	8	4239
8	6	3898
9	7	1620
10	9	638

Results per page: 50 1 – 24 of 24