PROJECT

Business Case - Target SQL

Sibimanyu M

1) Exploratory Analysis

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

QUERY

```
SELECT column_name, data_type
FROM `target-project-389509`.Target.INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = "customers"
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

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

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

INSIGHT

INFORMATION_SCHEMA.COLUMNS view allows us to get information about all columns present in a table.

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

QUERY

```
SELECT
 MAX(order_purchase_timestamp) AS Maximum,
 MIN(order_purchase_timestamp) AS Minimum,
 TIMESTAMP_DIFF(MAX(order_purchase_timestamp),MIN(order_purchase_timestamp),DAY) as
Days
FROM
 `Target.orders`
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	Maximum	Minimum	Days
1	2018-10-17 17:30:18 UTC	2016-09-04 21:15:19 UTC	772

INSIGHT

MIN AND MAX functions are used to get the extremes of the timestamp, TIMESTAMP_DIFF used to get the range of the timestamp.

3. Count the number of Cities and States in our dataset

QUERY

```
SELECT
       COUNT(DISTINCT(geolocation_city)) AS Cities,
       COUNT(DISTINCT(geolocation_state)) AS States
FROM
 `Target.geolocation`
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	Cities	States
1	8011	27

INSIGHT

DISTINCT function identifies unique rows in the column.

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,
EXTRACT(MONTH FROM order_purchase_timestamp) AS Month,
COUNT(*) AS Count
FROM `Target.orders`
GROUP BY Year,Month
ORDER BY 1,2
LIMIT 10
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

Row	Year	Month	Count
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

INSIGHT

We can see that the trend has been increasing over the years which is backed up by the output data points. If we plot a line chart with these data points having time on x-axis and no. of orders in y-axis, we can see that there is an growing trend over the past years.

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

QUERY

```
SELECT
EXTRACT(MONTH FROM order_purchase_timestamp) AS Month, COUNT(*) AS Count
FROM `Target.orders`
GROUP BY Month
ORDER BY Month
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

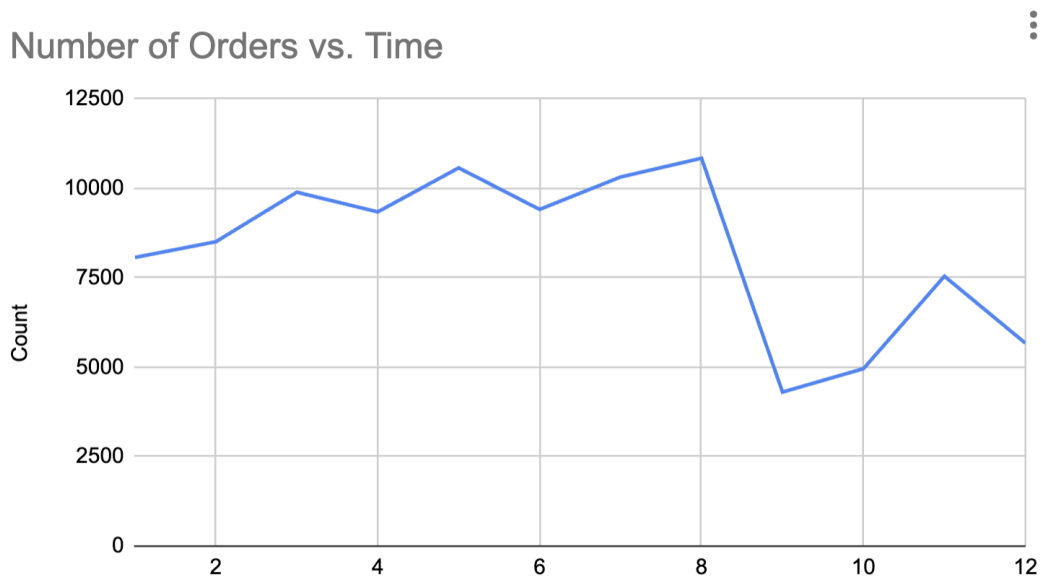
Row	Month	Count
1	1	8069
2	2	8508
3	3	9893
4	4	9343
5	5	10573
6	6	9412
7	7	10318

8	8	10843
9	9	4305
10	10	4959
11	11	7544
12	12	5674

Results per page: 50 1 – 12 of 12

INSIGHT

We can see a slight increase in orders from months 1 to 8 (Jan - Aug) and dip in the rest of the months in all years, we plot the data points on chart to infer insights,August has the maximum order count, studying the monthly seasonality,we need to stock up more in the start of those months in order to meet the demand.



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

```
SELECT
 CASE
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 0 AND EXTRACT(HOUR FROM
order_purchase_timestamp) <= 6 THEN 'Dawn'
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 7 AND EXTRACT(HOUR FROM
order_purchase_timestamp) <= 12 THEN 'Morning'
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 13 AND EXTRACT(HOUR FROM
order_purchase_timestamp) <= 18 THEN 'Afternoon'
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 19 AND EXTRACT(HOUR FROM
order_purchase_timestamp) <= 23 THEN 'Night'
 END AS time_of_day,
 COUNT(*) AS order_count
FROM
 `Target.orders`
GROUP BY
 time_of_day
ORDER BY
 order_count DESC;
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

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

INSIGHT

We can see that the order count was maximum during noon and minimum during dawn. We can see that during Afternoon the maximum number of orders were placed.

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
   EXTRACT(MONTH FROM order_purchase_timestamp) AS Month,
   cus.customer_state,
   COUNT(*) AS num_orders
FROM
   `Target.orders` AS ord
LEFT JOIN `Target.customers` AS cus
ON ord.customer_id = cus.customer_id
GROUP BY
   Month,
   cus.customer_state
ORDER BY
   customer_state,
   Month;
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

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

Row	Month	customer_state	num_orders
1	1	AC	8
2	2	AC	6
3	3	AC	4
4	4	AC	9
5	5	AC	10
6	6	AC	7
7	7	AC	9
8	8	AC	7
9	9	AC	5
10	10	AC	6
11	11	AC	5
12	12	AC	5
13	1	AL	39
14	2	AL	39

Results per page: 30 1 – 30 of 322

INSIGHT

We have the number of orders placed in each month ordered by state.

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

QUERY

```
SELECT
  customer_state,
  COUNT(DISTINCT customer_id) AS num_unique_customers,
  ROUND(COUNT(DISTINCT customer_id)/SUM(COUNT(DISTINCT customer_id))OVER()*100 ,2) as
Percentage
FROM
  `Target.customers`
GROUP BY
  customer_state
ORDER BY
  customer_state
```

RESULT

Query results

SAVE RESULTS EXPLORE DAT

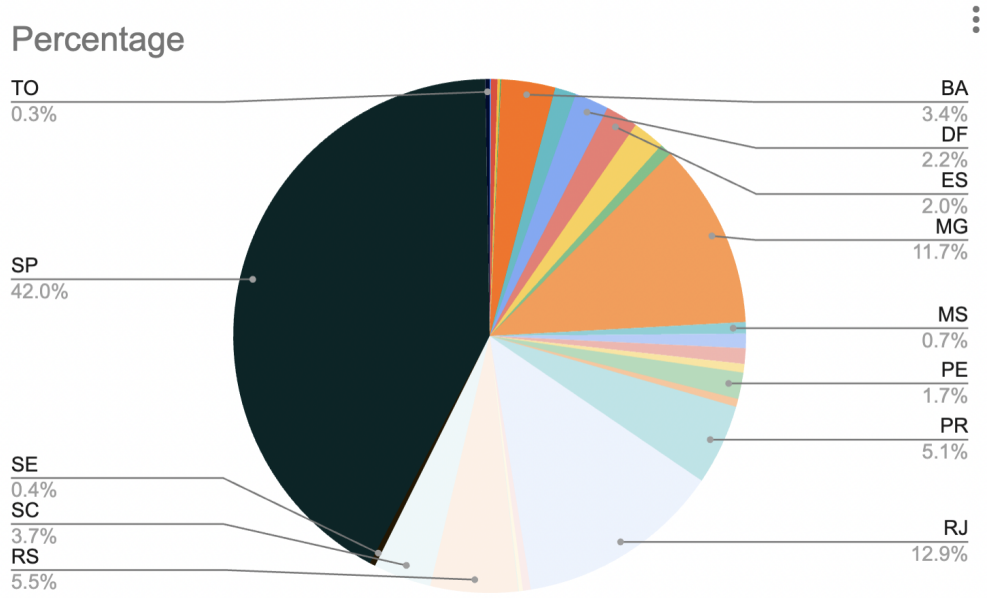
JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH

Row	customer_state	num_unique_custom	Percentage
1	AC	81	0.08
2	AL	413	0.42
3	AM	148	0.15
4	AP	68	0.07
5	BA	3380	3.4
6	CE	1336	1.34
7	DF	2140	2.15

Results per page: 50 1 – 27 of 27

INSIGHT

We have the number of unique customers in grouped by each state. We use percentage to show the distribution of the datapoints. Column Percentage sums to 100 (Occam's razor 1=1 test), since distribution must add upto 100.



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

```
SELECT ((total_cost_2018 - total_cost_2017) / total_cost_2017) * 100 AS
percentage_increase
FROM
 (SELECT
   (SELECT SUM(payment_value)
    FROM `Target.payments` as p
    JOIN `Target.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
total_cost_2017,
   (SELECT SUM(payment_value)
    FROM `Target.payments` as p
    JOIN `Target.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
total_cost_2018
 ) AS costs;
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

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

Row	percentage_increase
1	136.97687164665447

INSIGHT

The percentage increase is almost 137%.

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

QUERY

```
SELECT c.customer_state,
    ROUND(SUM(oi.price),2) AS total_order_price,
    ROUND(AVG(oi.price),2) AS average_order_price,
    ROUND(MAX(oi.price),2) AS maximum_order_price,
    ROUND(MIN(oi.price),2) AS minimum_order_price
FROM `Target.order_items` AS oi
INNER JOIN `Target.orders` AS o ON
oi.order_id = o.order_id
INNER JOIN `Target.customers` AS c ON
o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY c.customer_state;
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

Row	customer_state	total_order_price	average_order_price	maximum_order_price	minimum_order_pric
1	AC	15982.95	173.73	1200.0	12.9
2	AL	80314.81	180.89	1798.0	9.0
3	AM	22356.84	135.5	1688.0	8.5
4	AP	13474.3	164.32	1437.0	13.65
5	BA	511349.99	134.6	2999.89	5.2
6	CE	227254.71	153.76	2690.0	7.8
7	DF	302603.94	125.77	3999.0	4.9
8	ES	275037.31	121.91	6729.0	5.99
9	GO	294591.95	126.27	2740.0	3.9
10	MA	119648.22	145.2	2499.75	6.99
11	MG	1585308.03	120.75	4099.99	3.85

Results per page: 50 1 – 27 of 27

INSIGHT

We have the Total & Average value of order price for each state ordered by state.We can get extra details such as maximum value of the order price and minimum using functions.

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

QUERY

```
SELECT c.customer_state,
     ROUND(SUM(oi.freight_value),2) AS total_order_freight,
     ROUND(AVG(oi.freight_value),2) AS average_order_freight,
     ROUND(MAX(oi.freight_value),2) AS maximum_order_freight,
     ROUND(MIN(oi.freight_value),2) AS minimum_order_freight
FROM `Target.orders` AS o
INNER JOIN `Target.customers` AS c ON
o.customer_id = c.customer_id
INNER JOIN `Target.order_items` AS oi ON
oi.order_id = o.order_id
GROUP BY c.customer_state
ORDER BY c.customer_state;
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

Row	customer_state	total_order_freight	average_order_freight	maximum_order_freight	minimum_order_freight
1	AC	3686.75	40.07	108.36	14.86
2	AL	15914.59	35.84	314.4	0.0
3	AM	5478.89	33.21	165.75	3.96
4	AP	2788.5	34.01	133.39	0.0
5	BA	100156.68	26.36	284.6	0.0
6	CE	48351.59	32.71	250.57	0.0
7	DF	50625.5	21.04	294.76	0.0
8	ES	49764.6	22.06	321.88	0.0
9	GO	53114.98	22.77	174.95	0.0
10	MA	31523.77	38.26	245.75	0.0
11	MG	270853.46	20.63	322.1	0.0

Results per page: 50 1 – 27 of 27

INSIGHT

We have the Total & Average value of order freight price for each state ordered by state asked in the question.We get extra insights such as maximum and minimum for the freight order.

5) 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.

QUERY

```
WITH cte as
(
SELECT
 o.order_id,
 c.customer_state,
 DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp, Day) AS
time_to_deliver,
 DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, Day) AS
diff_estimated_delivery
FROM `Target.orders` as o
LEFT JOIN `Target.customers` as c
ON o.customer_id = c.customer_id
)

SELECT * FROM cte
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

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

Results per page: 50 1 – 50 of 99441

INSIGHT

We can find other parameters such as minimum,maximum days to deliver an order. We can group them by state and estimate how quick are the orders are delivered.

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

QUERY

```
SELECT customer_state,average_order_freight, maximum_order_freight,
minimum_order_freight
FROM
(
SELECT c.customer_state,
   ROUND(AVG(oi.freight_value),2) AS average_order_freight,
   ROUND(MAX(oi.freight_value),2) AS maximum_order_freight,
   ROUND(MIN(oi.freight_value),2) AS minimum_order_freight
FROM `Target.orders` AS o
LEFT JOIN `Target.customers` AS c ON
o.customer_id = c.customer_id
LEFT JOIN `Target.order_items` AS oi ON
oi.order_id = o.order_id
GROUP BY c.customer_state
)
ORDER BY average_order_freight DESC
LIMIT 5
```

RESULT

Query results

SAVE RESULTS EXPLORE DAT

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

Row	customer_state	average_order_freight	maximum_order_freight	minimum_order_freight
1	RR	42.98	144.86	25.38
2	PB	42.72	317.47	0.0
3	RO	41.07	217.53	0.0
4	AC	40.07	108.36	14.86
5	PI	39.15	409.68	0.0

QUERY

```
SELECT customer_state,average_order_freight, maximum_order_freight,
minimum_order_freight
FROM
(
SELECT c.customer_state,
   ROUND(AVG(oi.freight_value),2) AS average_order_freight,
   ROUND(MAX(oi.freight_value),2) AS maximum_order_freight,
   ROUND(MIN(oi.freight_value),2) AS minimum_order_freight
FROM `Target.orders` AS o
LEFT JOIN `Target.customers` AS c ON
o.customer_id = c.customer_id
LEFT JOIN `Target.order_items` AS oi ON
oi.order_id = o.order_id
GROUP BY c.customer_state
)
ORDER BY average_order_freight
LIMIT 5
```

RESULT

Query results

SAVE RESULTS EXPLORE DAT

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

Row	customer_state	average_order_freight	maximum_order_freight	minimum_order_freight
1	SP	15.15	339.59	0.0
2	PR	20.53	375.28	0.0
3	MG	20.63	322.1	0.0
4	RJ	20.96	207.78	0.0
5	DF	21.04	294.76	0.0

INSIGHT

We have the Highest & Lowest 5 states with average freight value. We get more details about the range of the freight value from the minimum and maximum.

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

QUERY

```
SELECT customer_state, AVG(time_to_deliver) as Average_delivery_time
FROM
(
SELECT
 c.customer_state,
 DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, Day) AS
time_to_deliver
FROM
`Target.orders` AS o
LEFT JOIN `Target.customers` AS c ON
o.customer_id = c.customer_id
)
GROUP BY customer_state
ORDER BY Average_delivery_time DESC
LIMIT 5
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	Average_delivery_tim
1	RR	28.97560975609...
2	AP	26.73134328358...
3	AM	25.98620689655...
4	AL	24.04030226700...
5	PA	23.31606765327...

QUERY

```
SELECT customer_state, AVG(time_to_deliver) as Average_delivery_time
FROM
(
SELECT
 c.customer_state,
 DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date, Day) AS
time_to_deliver
FROM
`Target.orders` AS o
LEFT JOIN `Target.customers` AS c ON
o.customer_id = c.customer_id
)
GROUP BY customer_state
ORDER BY Average_delivery_time
LIMIT 5
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	Average_delivery_tim
1	SP	8.298061489072...
2	PR	11.52671135486...
3	MG	11.54381329810...
4	DF	12.50913461538...
5	SC	14.47956019171...

INSIGHT

We have the Highest & Lowest 5 states with average delivery time.

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

QUERY

```
WITH cte as
(
SELECT
 o.order_id,
 c.customer_state,
DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, Day) AS
DiffEstimatedDelivery
FROM `Target.orders` as o
LEFT JOIN `Target.customers` as c ON o.customer_id = c.customer_id
)

SELECT
cte.customer_state,
AVG(cte.DiffEstimatedDelivery) AS AverageDiffEstimatedDelivery
```

```
FROM cte GROUP BY cte.customer_state ORDER BY AVG(cte.DiffEstimatedDelivery)
LIMIT 5
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH | PRE

Row	customer_state	Averge_DiffEstimate
1	AL	7.947103274559...
2	MA	8.768479776847...
3	SE	9.173134328358...
4	ES	9.618546365914...
5	BA	9.934889434889...

INSIGHT

We have top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

6) Analysis based on the payments:

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

QUERY

```
SELECT
EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS Month,
payment_type,
COUNT(*) AS order_count
FROM `Target.payments` AS p
LEFT JOIN `Target.orders` AS o ON p.order_id = o.order_id
GROUP BY Year, Month, payment_type
ORDER BY Year, Month;
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

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

Row	Year	Month	payment_type	order_count
1	2016	9	credit_card	3
2	2016	10	credit_card	254
3	2016	10	voucher	23
4	2016	10	debit_card	2
5	2016	10	UPI	63
6	2016	12	credit_card	1

Results per page: 50 1 – 50 of 90

INSIGHT

Result obtained shows us the order count of the different types of payment made in month and year.

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

QUERY

```
SELECT
 payment_installments,
 COUNT(*) AS order_count
FROM `Target.payments`
WHERE payment_installments > 0
GROUP BY payment_installments;
```

RESULT

Query results

SAVE RESULTS EXPLORE DATA

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

Row	payment_installment	order_count
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	5	5239
6	6	3920

Results per page: 50 1 – 23 of 23

INSIGHT

We have number of payment installments ranging from 1 to 24 and their respective order counts.