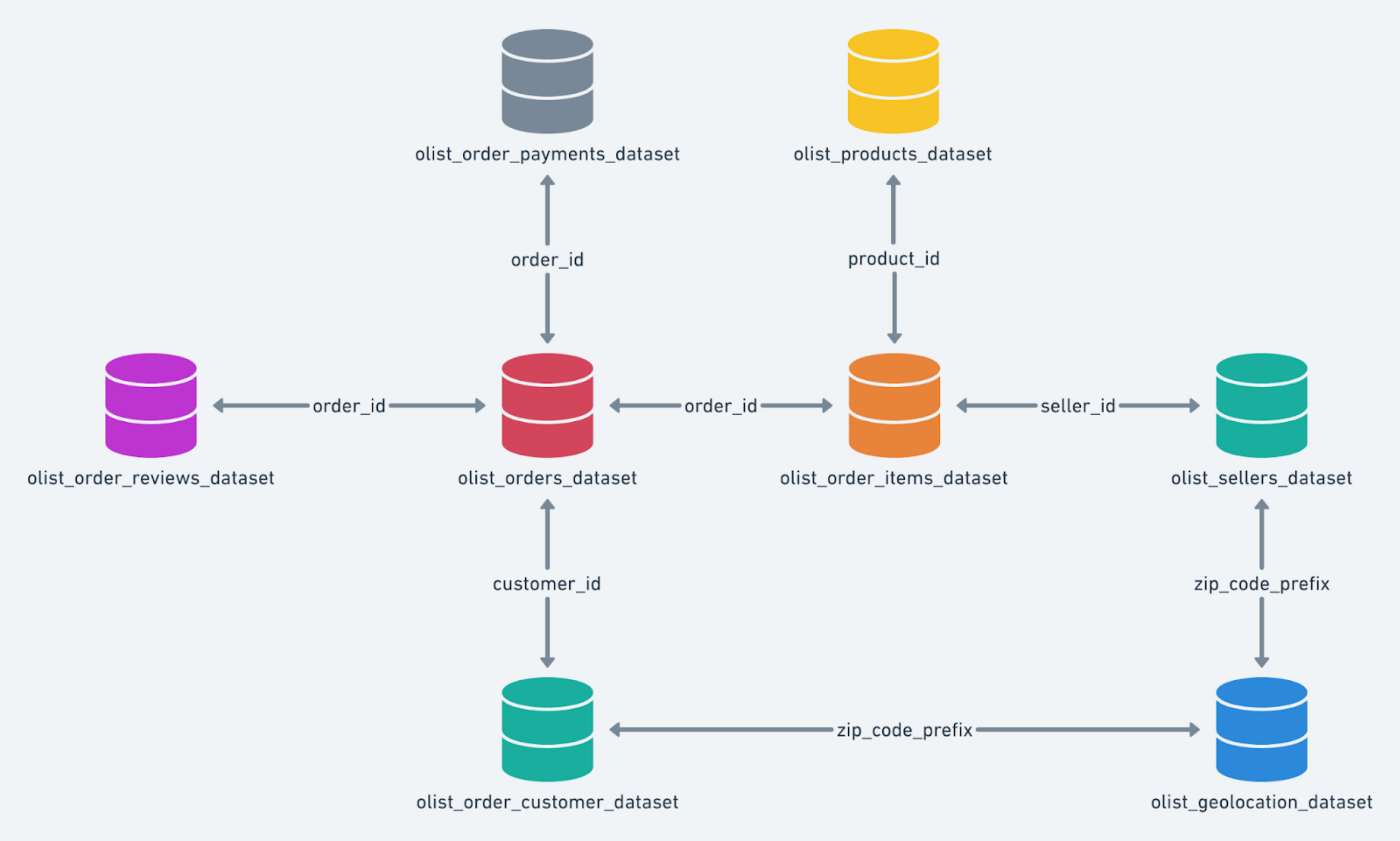
TARGET

Case Study

Stage 1 : Data Exploration

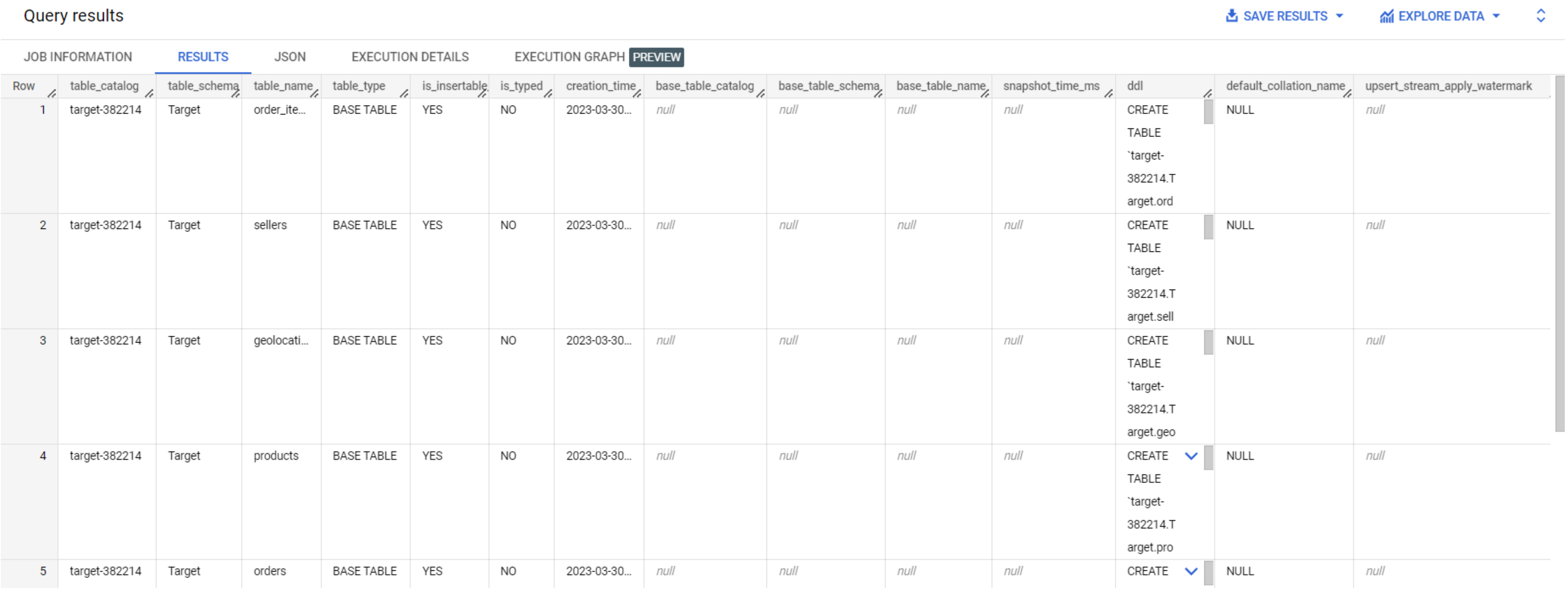


Schema of the **Target** dataset

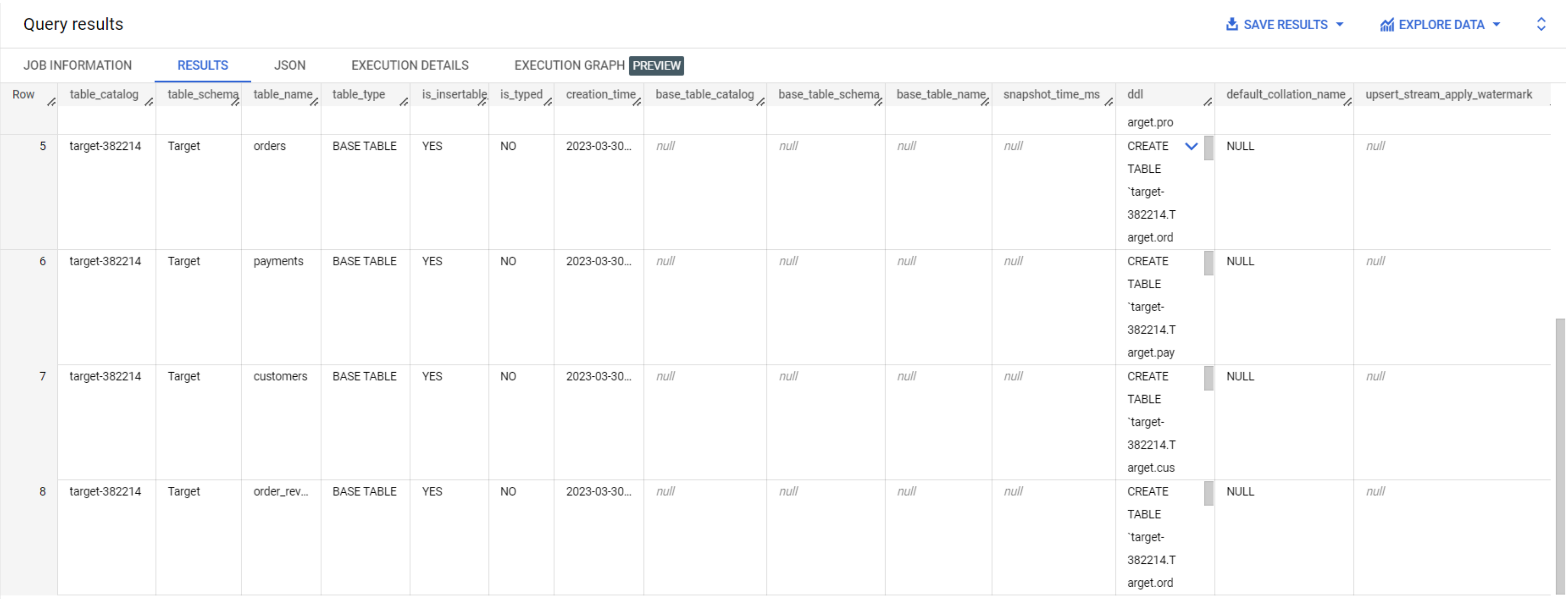
Assessment of the structure and data types of different columns across tables in the given dataset :

Query :

```
SELECT *
FROM Target.INFORMATION_SCHEMA.TABLES;    --Target is the dataset name here
```



Row	table_catalog	table_schema	table_name	table_type	is_insertable	is_typed	creation_time	base_table_catalog	base_table_schema	base_table_name	snapshot_time_ms	ddl	default_collation_name	upsert_stream_apply_watermark
1	target-382214	Target	order_ite...	BASE TABLE	YES	NO	2023-03-30...	null	null	null	null	CREATE TABLE `target-382214.Target.ord	NULL	null
2	target-382214	Target	sellers	BASE TABLE	YES	NO	2023-03-30...	null	null	null	null	CREATE TABLE `target-382214.Target.sell	NULL	null
3	target-382214	Target	geolocati...	BASE TABLE	YES	NO	2023-03-30...	null	null	null	null	CREATE TABLE `target-382214.Target.geo	NULL	null
4	target-382214	Target	products	BASE TABLE	YES	NO	2023-03-30...	null	null	null	null	CREATE TABLE `target-382214.Target.pro	NULL	null
5	target-382214	Target	orders	BASE TABLE	YES	NO	2023-03-30...	null	null	null	null	CREATE	NULL	null


Query results

Row	table_catalog	table_schema	table_name	table_type	is_insertable	is_typed	creation_time	base_table_catalog	base_table_schema	base_table_name	snapshot_time_ms	ddl	default_collation_name	upsert_stream_apply_watermark
												arget.pro		
5	target-382214	Target	orders	BASE TABLE	YES	NO	2023-03-30...	null	null	null	null	CREATE TABLE `target-382214.Target.ord	NULL	null
6	target-382214	Target	payments	BASE TABLE	YES	NO	2023-03-30...	null	null	null	null	CREATE TABLE `target-382214.Target.pay	NULL	null
7	target-382214	Target	customers	BASE TABLE	YES	NO	2023-03-30...	null	null	null	null	CREATE TABLE `target-382214.Target.cus	NULL	null
8	target-382214	Target	order_rev...	BASE TABLE	YES	NO	2023-03-30...	null	null	null	null	CREATE TABLE `target-382214.Target.ord	NULL	null

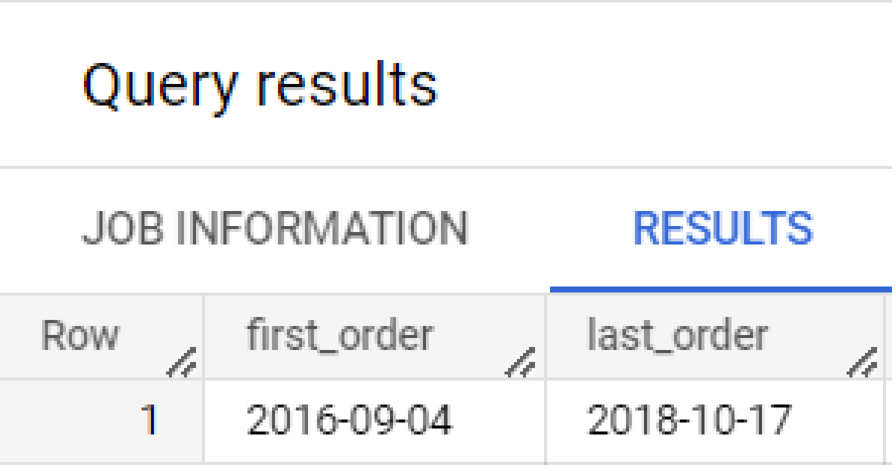
Here,

- **The dataset contains 8 tables namely, customers, geolocation, payments, orders, order_items, sellers, reviews, products**
- **"target-382214" in the column table_catalog is the project ID of the project that contains the dataset.**
- **"Target" in the column table_schema is the name of the dataset that contains the table.**
- **"YES" in the column is_insertable_into states that the table supports DML INSERT statements.**
- **"NO" in the is_types column is always no.**

Analyzing the time period for which the data is given :

Query :

```
SELECT
      EXTRACT(DATE FROM MIN(order_purchase_timestamp)) AS first_order,
      EXTRACT(DATE FROM MAX(order_purchase_timestamp)) AS last_order
FROM `Target.orders`;
```


Query results

Row	first_order	last_order
1	2016-09-04	2018-10-17

Here,

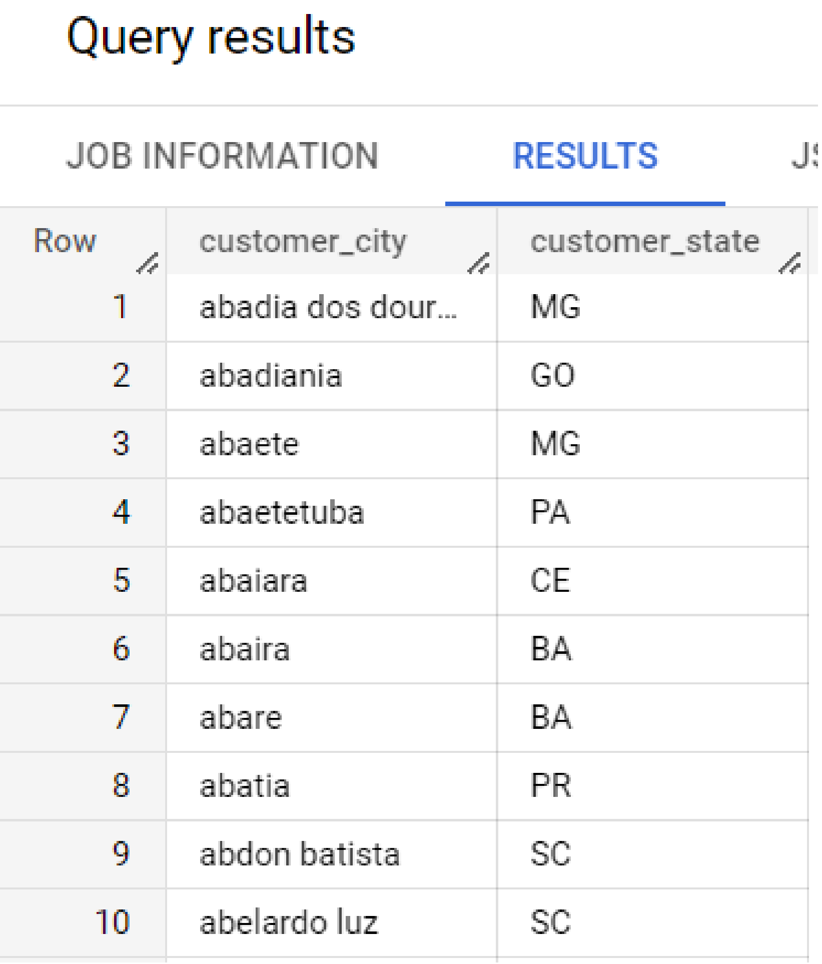
- **The first order was placed on 4 September 2016.**
- **The last order given to us is of 17 October 2018.**
- **Therefore the data that has been provided to us is between 4 September 2016 to 17 October 2018.**

Cities and States of customers ordered during the given period :

Query:

```
SELECT
 DISTINCT c.customer_city,
 c.customer_state
FROM `Target.customers` AS c
JOIN `Target.orders` AS o
```

```
ON c.customer_id = o.customer_id
ORDER BY c.customer_city;
```



Row	customer_city	customer_state
1	abadia dos dour...	MG
2	abadiania	GO
3	abaete	MG
4	abaetetuba	PA
5	abaiara	CE
6	abaira	BA
7	abare	BA
8	abatia	PR
9	abdon batista	SC
10	abelardo luz	SC

Here,

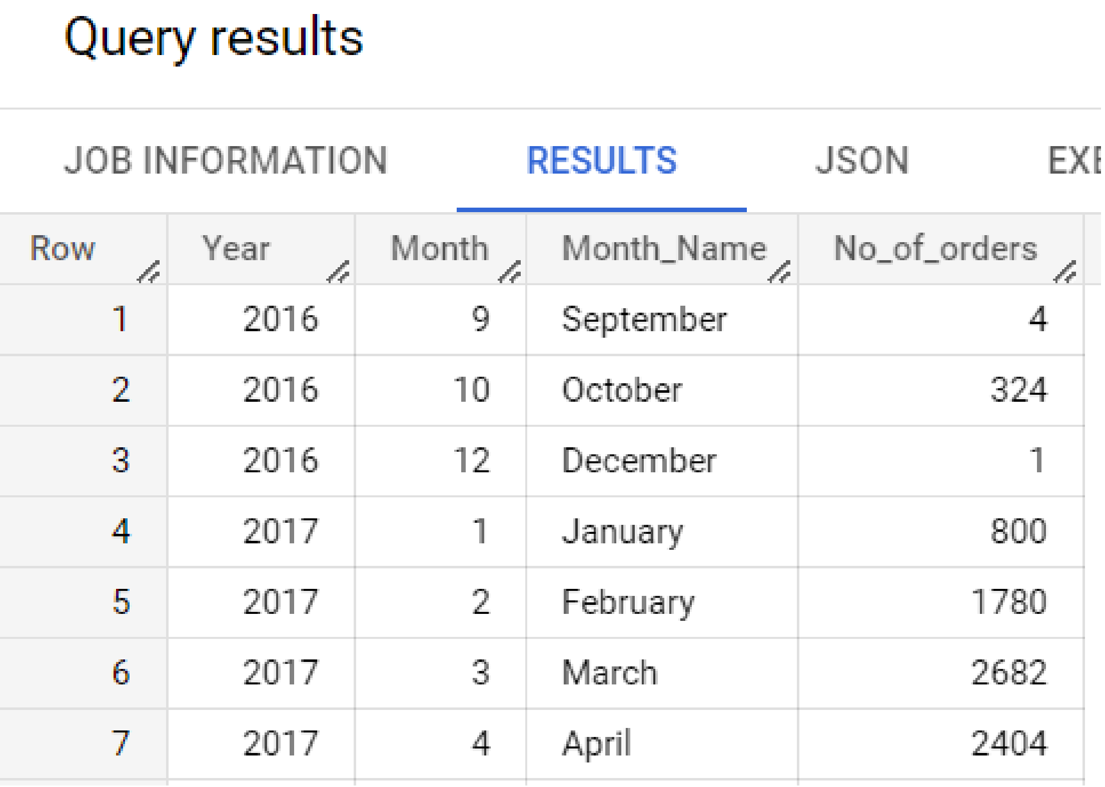
- **These are the cities and states of of customers in Brazil that have ordered between the given period.**

Stage 2 : In-depth Exploration

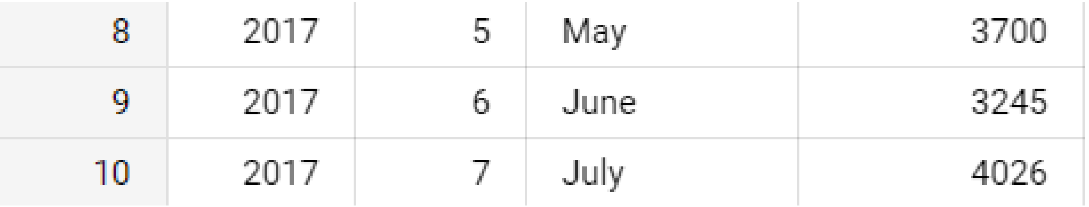
Understanding the trend of e-commerce in Brazil :

Query:

```
SELECT
 EXTRACT(YEAR FROM order_purchase_timestamp) AS Year,
 EXTRACT(MONTH FROM order_purchase_timestamp) AS Month,
 FORMAT_DATETIME("%B",order_purchase_timestamp) AS Month_Name,
 COUNT(*) AS No_of_orders
FROM `Target.orders`
WHERE order_purchase_timestamp IS NOT NULL
GROUP BY Year,Month,Month_Name
ORDER BY Year,Month;
```



Row	Year	Month	Month_Name	No_of_orders
1	2016	9	September	4
2	2016	10	October	324
3	2016	12	December	1
4	2017	1	January	800
5	2017	2	February	1780
6	2017	3	March	2682
7	2017	4	April	2404



8	2017	5	May	3700
9	2017	6	June	3245
10	2017	7	July	4026

Understanding the trend

Assumption :

- Only analyzing this by the number of orders.
- Also considering orders that were cancelled or returned.
- 2016 may be considered as an outlier.

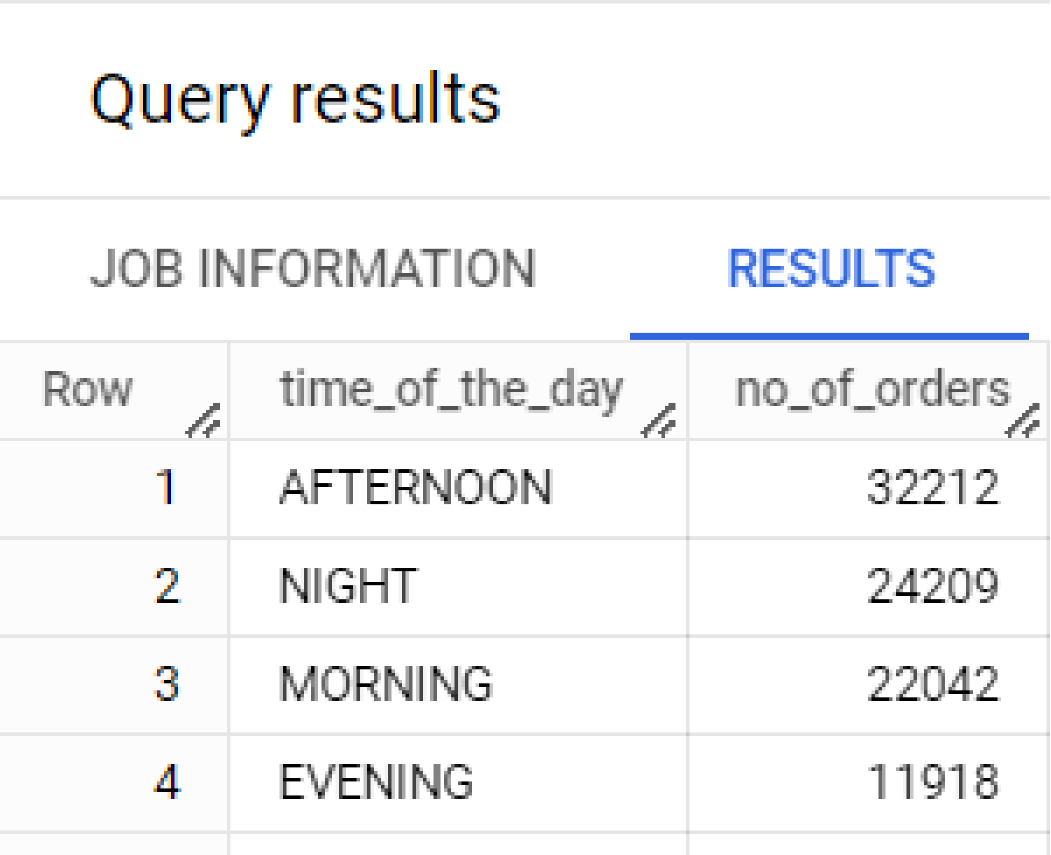
Insights :

- As per the results, e-commerce in Brazil has shown a small amount of growth in 2017 since the number of orders has been consistently increasing. Also, November emerged as the best performing month and noticed almost 50% increase in the number of orders.
- However, the growth was almost stagnant in 2018 since the number of orders is almost similar throughout the year and saw a major dip in September 2018.
- There has been a decent spike in number of orders in the month of August depicting a seasonality when orders between January and September are taken into account.(2016 excluded)

Preferable shopping time for people in Brazil :

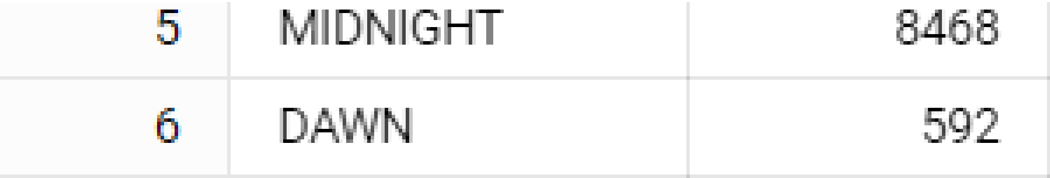
Query:

```
SELECT
 CASE
  WHEN EXTRACT(TIME FROM order_purchase_timestamp) BETWEEN '04:00:00' AND '06:30:00'
  THEN "DAWN"
  WHEN EXTRACT(TIME FROM order_purchase_timestamp) BETWEEN '06:30:01' AND '12:00:00'
  THEN "MORNING"
  WHEN EXTRACT(TIME FROM order_purchase_timestamp) BETWEEN '12:00:01' AND '17:00:00'
  THEN "AFTERNOON"
  WHEN EXTRACT(TIME FROM order_purchase_timestamp) BETWEEN '17:00:01' AND '19:00:00'
  THEN "EVENING"
  WHEN EXTRACT(TIME FROM order_purchase_timestamp) BETWEEN '19:00:01' AND '23:00:00'
  THEN "NIGHT"
  ELSE "MIDNIGHT"
 END AS time_of_the_day,
 COUNT(*) AS no_of_orders
FROM `Target.orders`
GROUP BY time_of_the_day
ORDER BY No_of_orders DESC;
```



Query results

Row	time_of_the_day	no_of_orders
1	AFTERNOON	32212
2	NIGHT	24209
3	MORNING	22042
4	EVENING	11918



5	MIDNIGHT	8468
6	DAWN	592

Assumptions :

- Assuming that when this data was created, it was adjusted according to Brazil.
- Dawn - 04:00:00 to 06:30:00
- Morning - 06:30:01 to 12:00:00
- Afternoon - 12:00:01 to 17:00:00
- Evening - 17:00:01 to 19:00:00
- Night - 19:00:01 to 23:00:00
- Midnight - 23:00:01 to 03:59:59

Intuition :

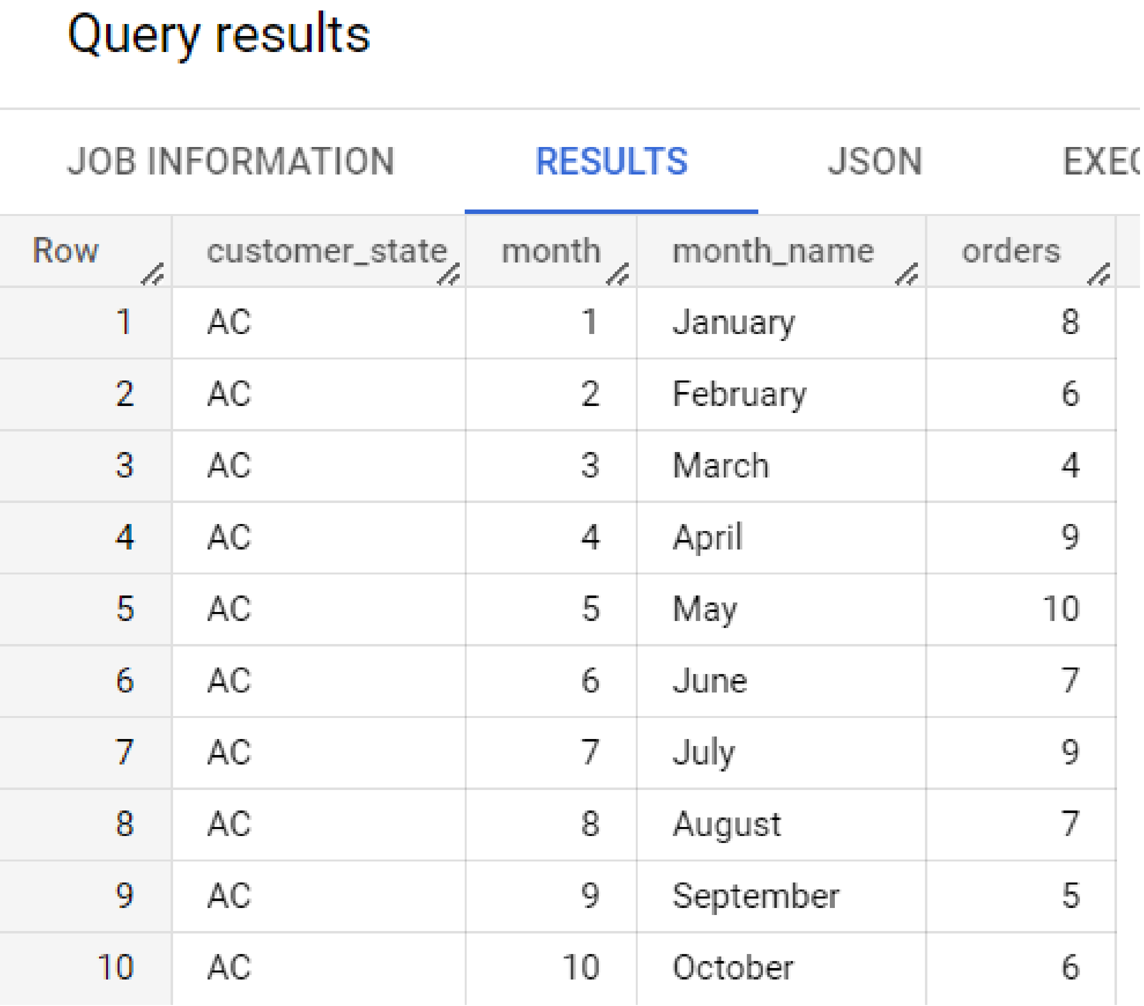
- The people of Brazil mostly tend to order in afternoon(Between 12:00:01 and 17:00:00) and night(Between 19:00:01 and 23:00:00).
- Dawn(Between 04:00:00 AND 06:30:00) has seen the least of orders.

Stage 3 : Evolution of e-commerce orders in Brazil region :

Month on month orders by states :

Query:

```
SELECT
 DISTINCT customer_state,
 EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
 FORMAT_DATETIME("%B",order_purchase_timestamp) AS month_name,
 COUNT(*) AS orders
FROM `Target.customers` AS c
JOIN `Target.orders` AS o
ON c.customer_id = o.customer_id
WHERE order_purchase_timestamp IS NOT NULL
GROUP BY customer_state,month,month_name
ORDER BY customer_state,month;
```



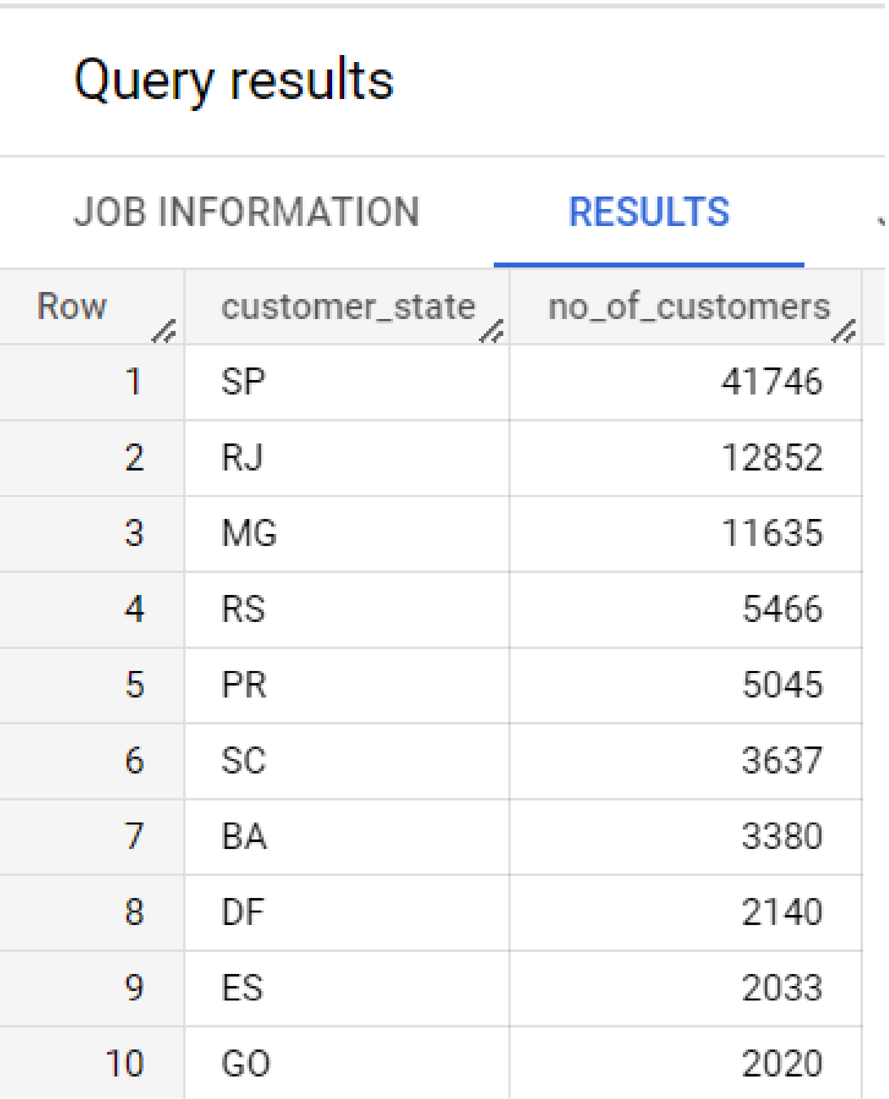
Row	customer_state	month	month_name	orders
1	AC	1	January	8
2	AC	2	February	6
3	AC	3	March	4
4	AC	4	April	9
5	AC	5	May	10
6	AC	6	June	7
7	AC	7	July	9
8	AC	8	August	7
9	AC	9	September	5
10	AC	10	October	6

- This is the month-wise number of orders of every state where orders have been placed.

Distribution of customers across the states in Brazil:

Query:-

```
SELECT
 DISTINCT customer_state,
 COUNT(*) 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
10	GO	2020

Assumptions :

- A customer is someone who has registered on Target's website.

Intuition :

- The customer are scattered throughout the country.
- A large number of customers are concentrated at Sao Paulo, Rio De Janeiro and Minas Gerais.
- Sao Paulo has most number of customers.

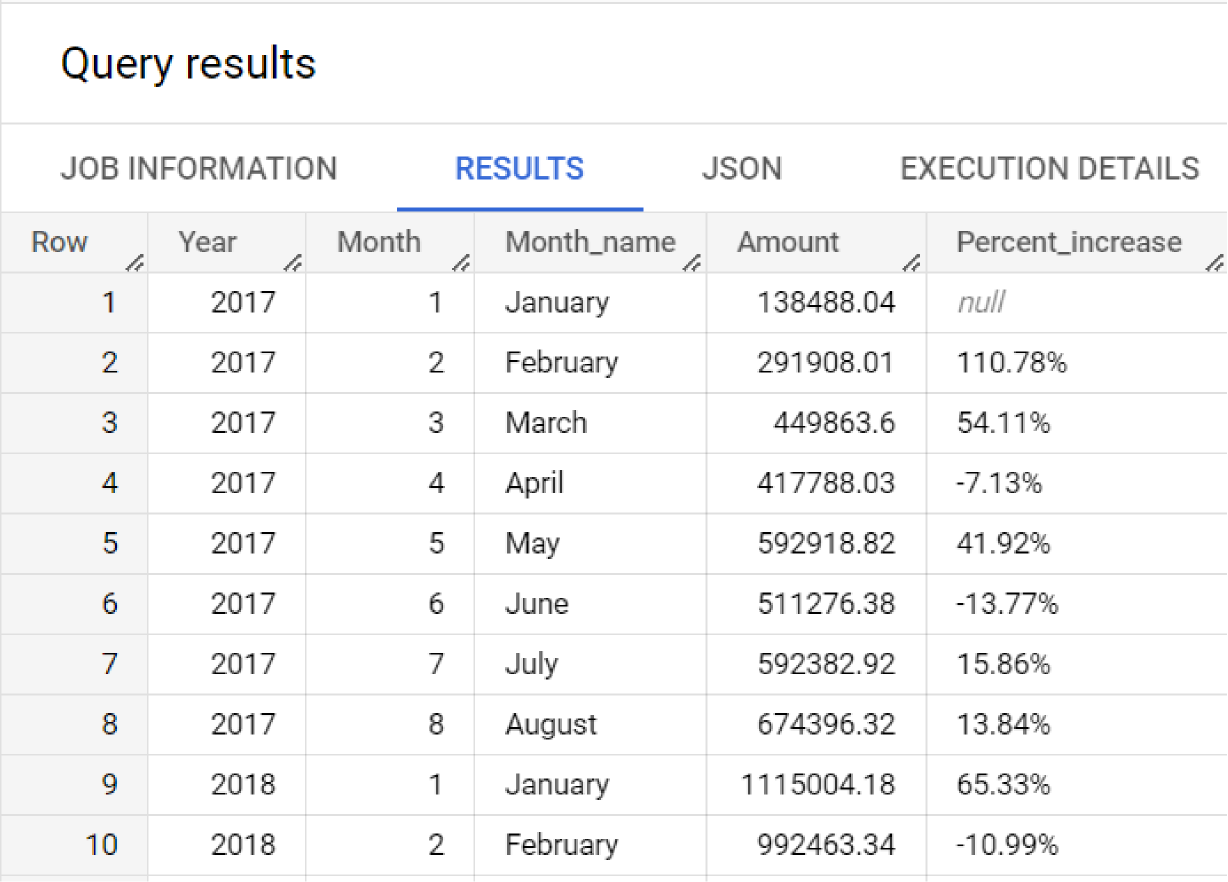
Stage 4 : Impact on Economy

Percent increase in cost of orders from 2017 to 2018 (Between Jan to Aug only):

Month-on-month increment in sales in 2017-18

Query:

```
SELECT *,
 CONCAT(ROUND((Amount - LAG(Amount,1) OVER (ORDER BY Year,Month))/(LAG(Amount,1) OVER (ORDER BY Year,Month))*100,2),"%") AS Percent_increase
FROM(SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) AS Year,
  EXTRACT(MONTH FROM order_purchase_timestamp) AS Month,
  FORMAT_DATETIME("%B",order_purchase_timestamp) AS Month_name,
  ROUND(SUM(payment_value),2) AS Amount
 FROM `Target.payments` AS p
 JOIN `Target.orders` AS o
 ON p.order_id = o.order_id
 WHERE (EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8) AND (EXTRACT(YEAR FROM order_purchase_timestamp) BETWEEN 2017 AND 2018)
 GROUP BY Year,Month,Month_name) AS x
 ORDER BY Year,Month
```



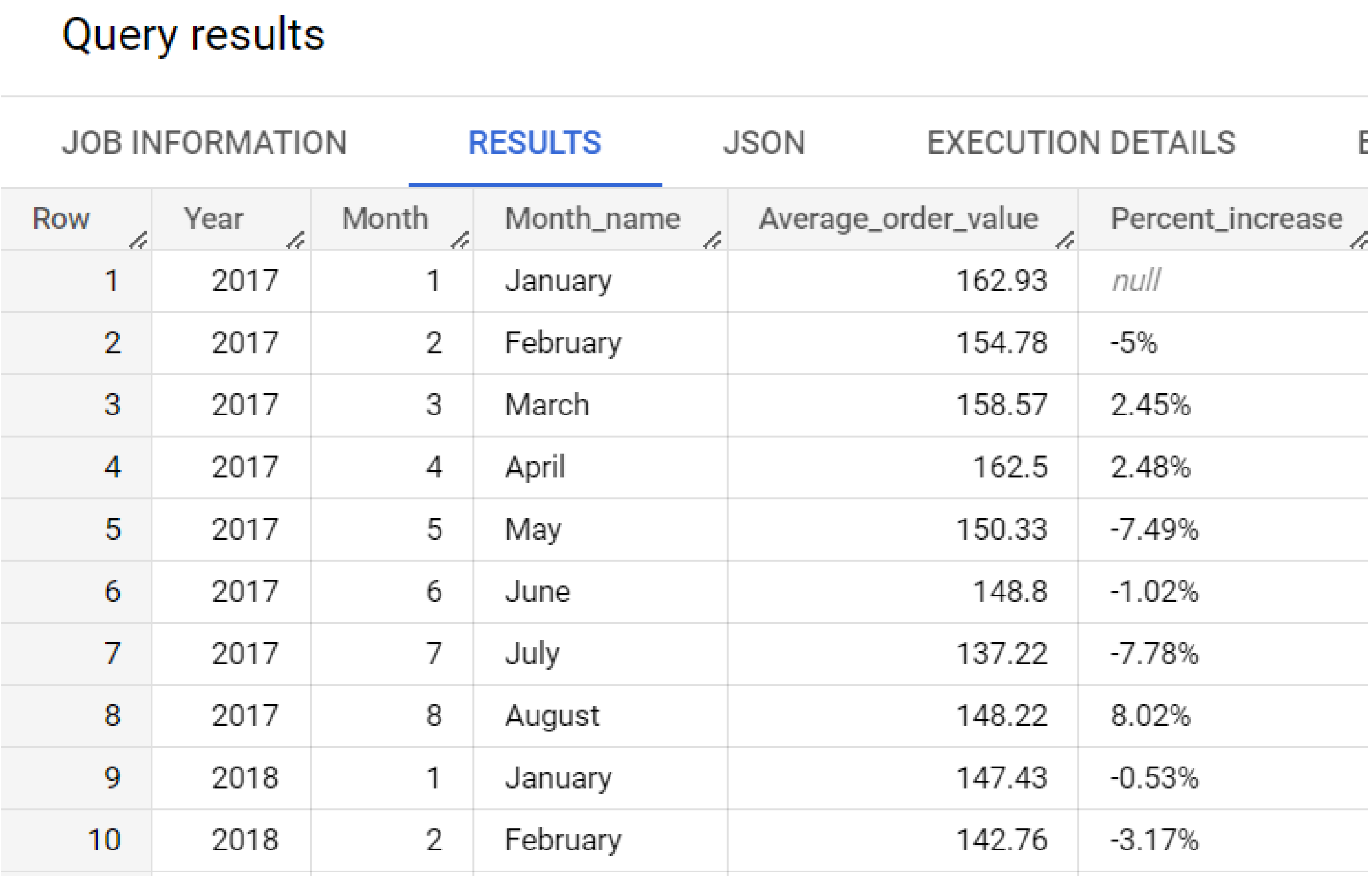
Query results

Row	Year	Month	Month_name	Amount	Percent_increase
1	2017	1	January	138488.04	null
2	2017	2	February	291908.01	110.78%
3	2017	3	March	449863.6	54.11%
4	2017	4	April	417788.03	-7.13%
5	2017	5	May	592918.82	41.92%
6	2017	6	June	511276.38	-13.77%
7	2017	7	July	592382.92	15.86%
8	2017	8	August	674396.32	13.84%
9	2018	1	January	1115004.18	65.33%
10	2018	2	February	992463.34	-10.99%

Month-on-month increment in average order value in 2017-18

Query :

```
SELECT *,
 CONCAT(ROUND((Average_order_value - LAG(Average_order_value,1) OVER (ORDER BY Year,Month))/(LAG(Average_order_value,1) OVER (ORDER BY Year,Month))*100,2),"%") AS Percent_increase
FROM(SELECT
   EXTRACT(YEAR FROM order_purchase_timestamp) AS Year,
   EXTRACT(MONTH FROM order_purchase_timestamp) AS Month,
   FORMAT_DATETIME("%B",order_purchase_timestamp) AS Month_name,
   ROUND(AVG(payment_value),2) AS Average_order_value
 FROM `Target.payments` AS p
 JOIN `Target.orders` AS o
 ON p.order_id = o.order_id
 WHERE (EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8) AND (EXTRACT(YEAR FROM order_purchase_timestamp) BETWEEN 2017 AND 2018)
 GROUP BY Year,Month,Month_name) AS x
 ORDER BY Year,Month
```



Query results

Row	Year	Month	Month_name	Average_order_value	Percent_increase
1	2017	1	January	162.93	null
2	2017	2	February	154.78	-5%
3	2017	3	March	158.57	2.45%
4	2017	4	April	162.5	2.48%
5	2017	5	May	150.33	-7.49%
6	2017	6	June	148.8	-1.02%
7	2017	7	July	137.22	-7.78%
8	2017	8	August	148.22	8.02%
9	2018	1	January	147.43	-0.53%
10	2018	2	February	142.76	-3.17%

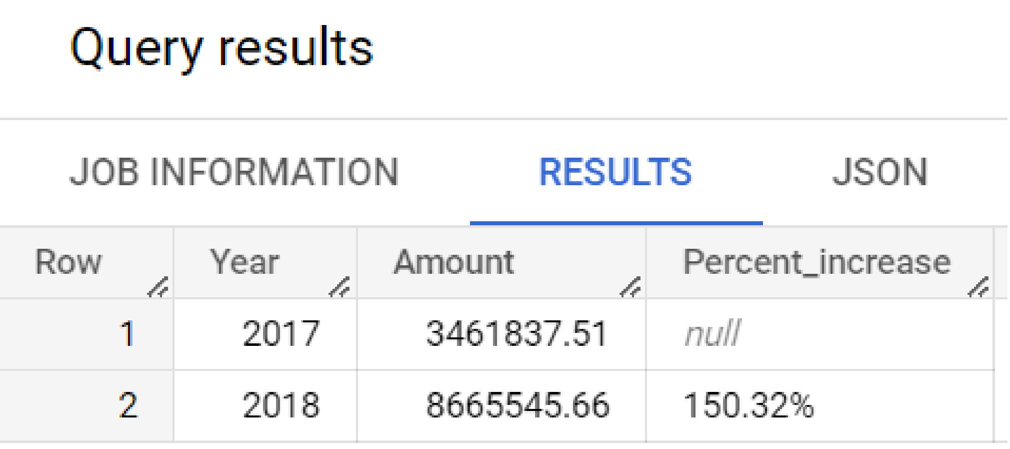
Insights:

- Total sales per month has been gradually increasing from 2017 and 2018.
- Average order value in 2017 has been fluctuating but has increased a little in 2018.

Year-on-year increment in sales from 2017 to 2018(Between January to July only)

Query:

```
SELECT *,
  CONCAT(ROUND((Amount - LAG(Amount,1) OVER (ORDER BY Year))/(LAG(Amount,1) OVER (ORDER BY Year))*100,2),"%") AS Percent_increase
FROM(SELECT
  EXTRACT(YEAR FROM order_delivered_carrier_date) AS Year,
  ROUND(SUM(payment_value),2) AS Amount
 FROM `Target.payments` AS p
 JOIN `Target.orders` AS o
 ON p.order_id = o.order_id
 WHERE (EXTRACT(MONTH FROM order_delivered_carrier_date) BETWEEN 1 AND 8) AND (EXTRACT(YEAR FROM order_delivered_carrier_date) BETWEEN 2017 AND 2018)
 GROUP BY Year)AS x
ORDER BY Year;
```

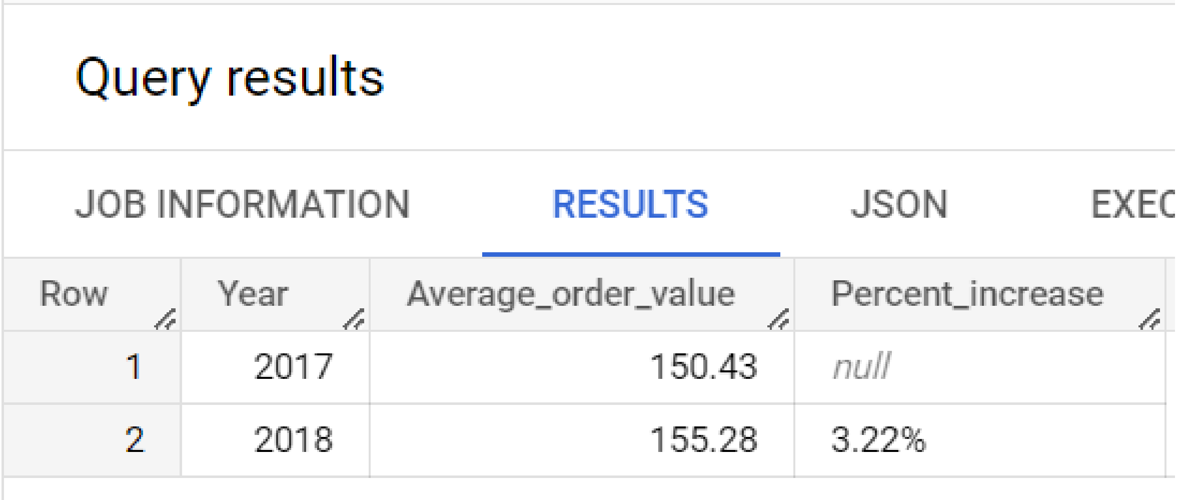


Row	Year	Amount	Percent_increase
1	2017	3461837.51	*null*
2	2018	8665545.66	150.32%

Year-on-year increment in average order value from 2017 to 2018(Between January to July only)

Query :

```
SELECT *,
 CONCAT(ROUND((Average_order_value - LAG(Average_order_value,1) OVER (ORDER BY Year))/(LAG(Average_order_value,1) OVER (ORDER BY Year))*100,2),"%") AS Percent_increase
FROM(SELECT
  EXTRACT(YEAR FROM order_purchase_timestamp) AS Year,
  ROUND(AVG(payment_value),2) AS Average_order_value
 FROM `Target.payments` AS p
 JOIN `Target.orders` AS o
 ON p.order_id = o.order_id
 WHERE (EXTRACT(MONTH FROM order_purchase_timestamp) BETWEEN 1 AND 8) AND (EXTRACT(YEAR FROM order_purchase_timestamp) BETWEEN 2017 AND 2018)
 GROUP BY Year) AS x
 ORDER BY Year
```



Row	Year	Average_order_value	Percent_increase
1	2017	150.43	*null*
2	2018	155.28	3.22%

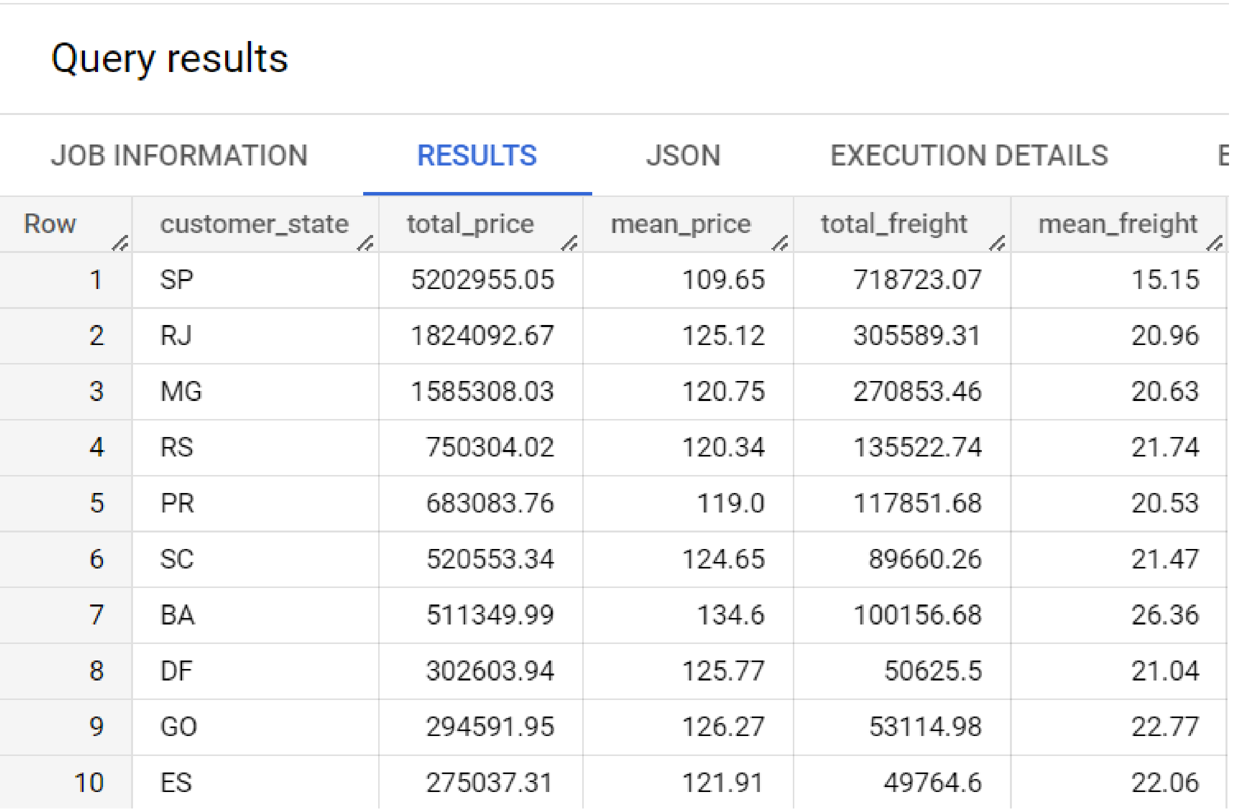
Insights:

- Total sales in 2018 is 150% more than the sales in 2017 which are very good signs for any economy signifying growth.
- Average order value has increased in 2018 by 3.22% as compared to 2017.

Mean & Sum of price and freight value by customer state

Query :

```
SELECT
 customer_state,
 ROUND(SUM(price),2) AS total_price,
 ROUND(AVG(price),2) AS mean_price,
 ROUND(SUM(freight_value),2) AS total_freight,
 ROUND(AVG(freight_value),2) AS mean_freight
FROM `Target.customers` AS c
JOIN `Target.orders` AS o
ON c.customer_id = o.customer_id
JOIN `Target.order_items`AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
ORDER BY total_price DESC
```



Row	customer_state	total_price	mean_price	total_freight	mean_freight
1	SP	5202955.05	109.65	718723.07	15.15
2	RJ	1824092.67	125.12	305589.31	20.96
3	MG	1585308.03	120.75	270853.46	20.63
4	RS	750304.02	120.34	135522.74	21.74
5	PR	683083.76	119.0	117851.68	20.53
6	SC	520553.34	124.65	89660.26	21.47
7	BA	511349.99	134.6	100156.68	26.36
8	DF	302603.94	125.77	50625.5	21.04
9	GO	294591.95	126.27	53114.98	22.77
10	ES	275037.31	121.91	49764.6	22.06

Insights :

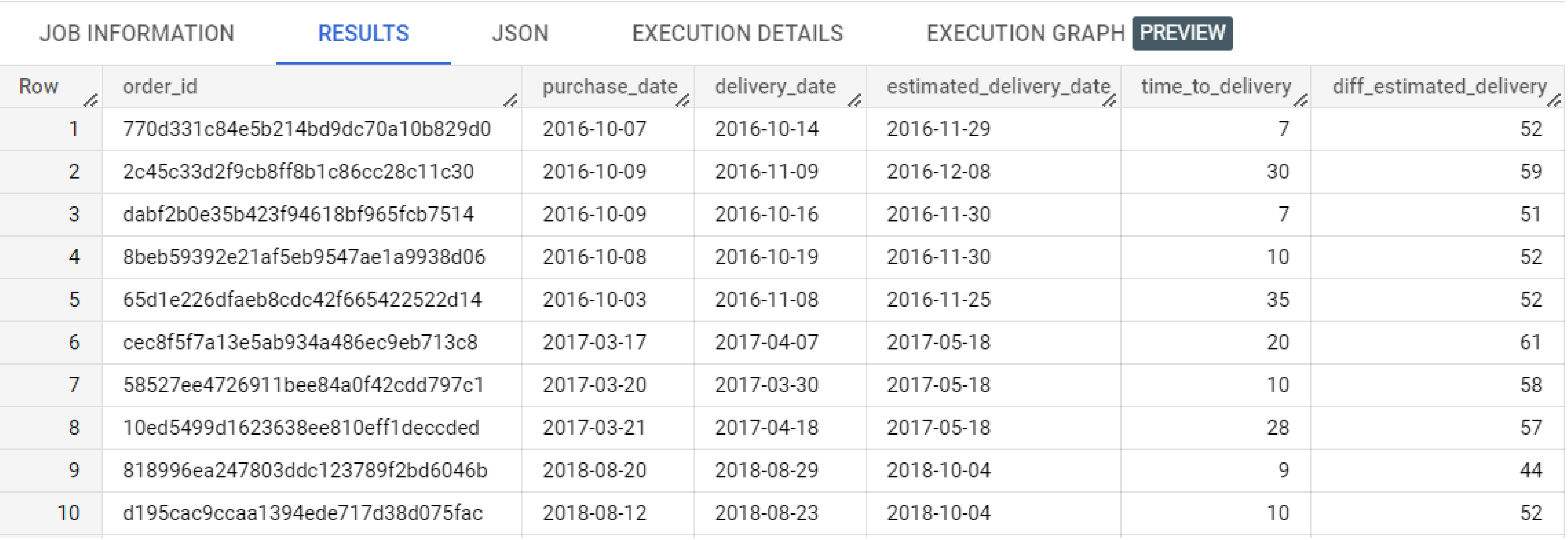
- **Freight value in sates with greater number of orders is lesser when compared to sates with lesser orders.**

Stage 5 : Analysis on sales, freight and delivery time

Days between purchasing, delivering and estimated delivery

Query:-

```
SELECT
 order_id,
 EXTRACT(DATE FROM(order_purchase_timestamp)) AS purchase_date,
 EXTRACT(DATE FROM(order_delivered_customer_date)) AS delivery_date,
 EXTRACT(DATE FROM(order_estimated_delivery_date)) AS estimated_delivery_date,
 DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_to_delivery,
 DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY) AS diff_estimated_delivery
FROM `Target.orders`
WHERE order_delivered_customer_date IS NOT NULL;  --NULL values  probably mean that the order was cancelled
```

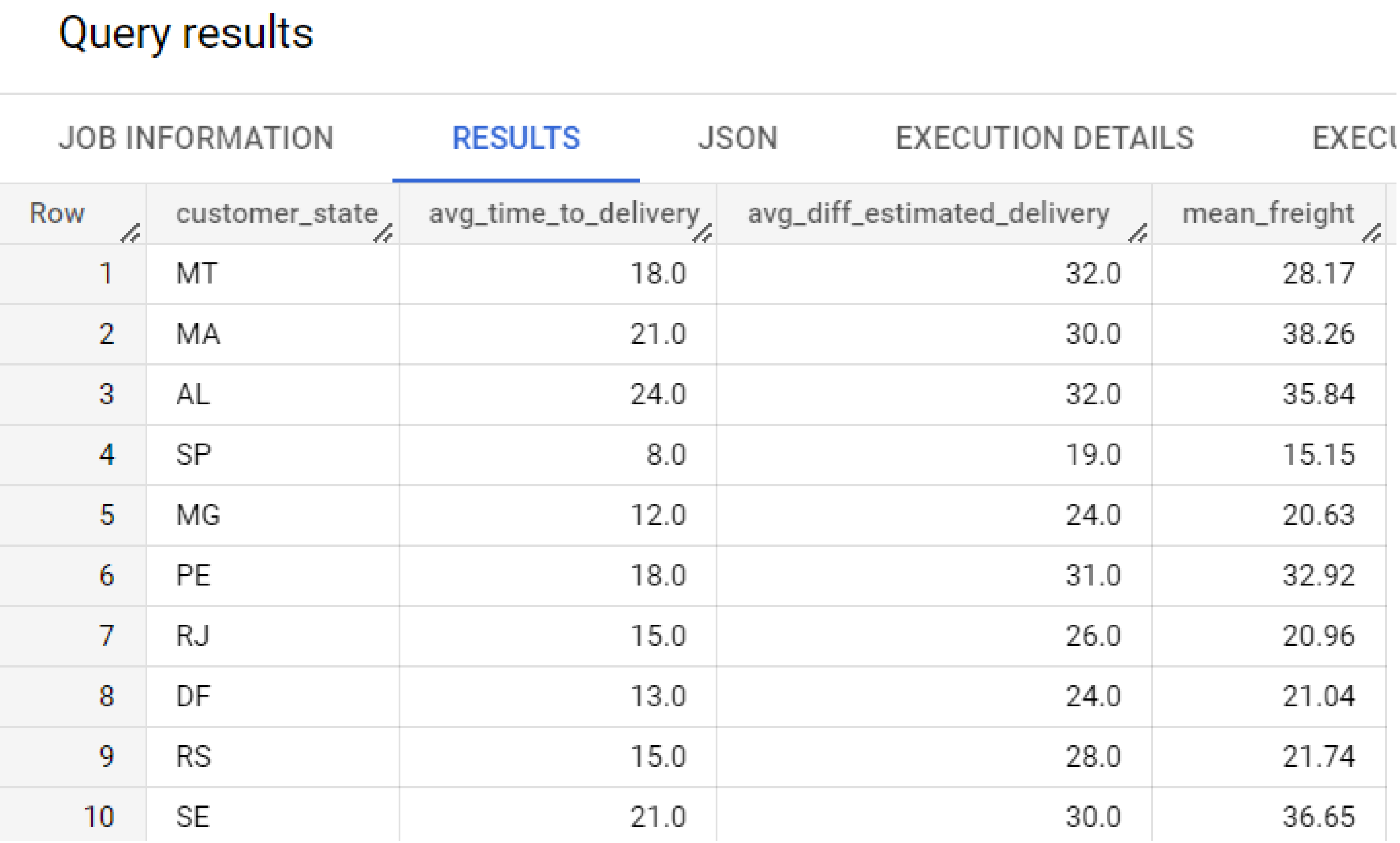


Row	order_id	purchase_date	delivery_date	estimated_delivery_date	time_to_delivery	diff_estimated_delivery
1	770d331c84e5b214bd9dc70a10b829d0	2016-10-07	2016-10-14	2016-11-29	7	52
2	2c45c33d2f9cb8ff8b1c86cc28c11c30	2016-10-09	2016-11-09	2016-12-08	30	59
3	dabf2b0e35b423f94618bf965fcb7514	2016-10-09	2016-10-16	2016-11-30	7	51
4	8beb59392e21af5eb9547ae1a9938d06	2016-10-08	2016-10-19	2016-11-30	10	52
5	65d1e226dfaeb8cdc42f665422522d14	2016-10-03	2016-11-08	2016-11-25	35	52
6	cec8f5f7a13e5ab934a486ec9eb713c8	2017-03-17	2017-04-07	2017-05-18	20	61
7	58527ee4726911bee84a0f42cdd797c1	2017-03-20	2017-03-30	2017-05-18	10	58
8	10ed5499d1623638ee810eff1deccded	2017-03-21	2017-04-18	2017-05-18	28	57
9	818996ea247803ddc123789f2bd6046b	2018-08-20	2018-08-29	2018-10-04	9	44
10	d195cac9ccaa1394ede717d38d075fac	2018-08-12	2018-08-23	2018-10-04	10	52

Analysis by mean of freight_value, time_to_delivery, diff_estimated_delivery on the data grouped by state

Query:-

```
SELECT
 customer_state,
 ROUND(AVG(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))) AS avg_time_to_delivery,
 ROUND(AVG(DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY))) AS avg_diff_estimated_delivery,
 ROUND(AVG(freight_value),2) AS mean_freight
FROM `Target.customers` AS c
JOIN `Target.orders` AS o
ON c.customer_id = o.customer_id
JOIN `Target.order_items`AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
```


Query results

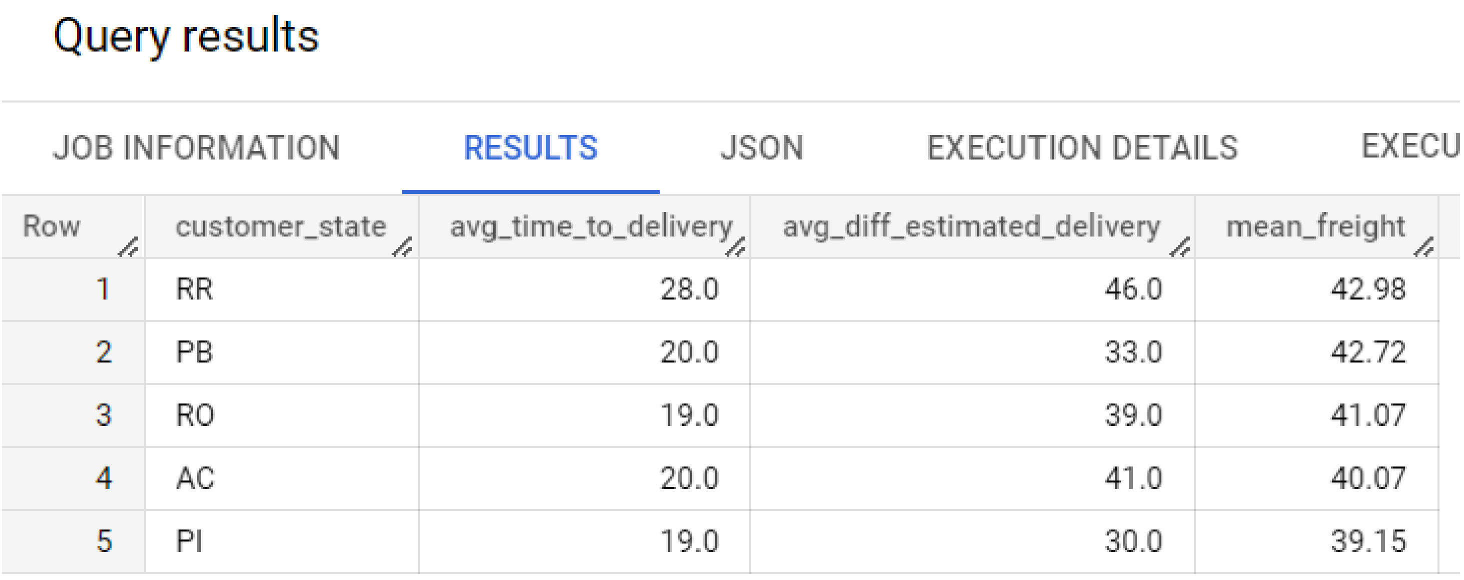
Row	customer_state	avg_time_to_delivery	avg_diff_estimated_delivery	mean_freight
1	MT	18.0	32.0	28.17
2	MA	21.0	30.0	38.26
3	AL	24.0	32.0	35.84
4	SP	8.0	19.0	15.15
5	MG	12.0	24.0	20.63
6	PE	18.0	31.0	32.92
7	RJ	15.0	26.0	20.96
8	DF	13.0	24.0	21.04
9	RS	15.0	28.0	21.74
10	SE	21.0	30.0	36.65

Top 5 states with highest average freight value

Query :-

```
SELECT
 customer_state,
 ROUND(AVG(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))) AS avg_time_to_delivery,
 ROUND(AVG(DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY))) AS avg_diff_estimated_delivery,
 ROUND(AVG(freight_value),2) AS mean_freight
FROM `Target.customers` AS c
```

```
JOIN `Target.orders` AS o
ON c.customer_id = o.customer_id
JOIN `Target.order_items`AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
ORDER BY mean_freight DESC
LIMIT 5
```

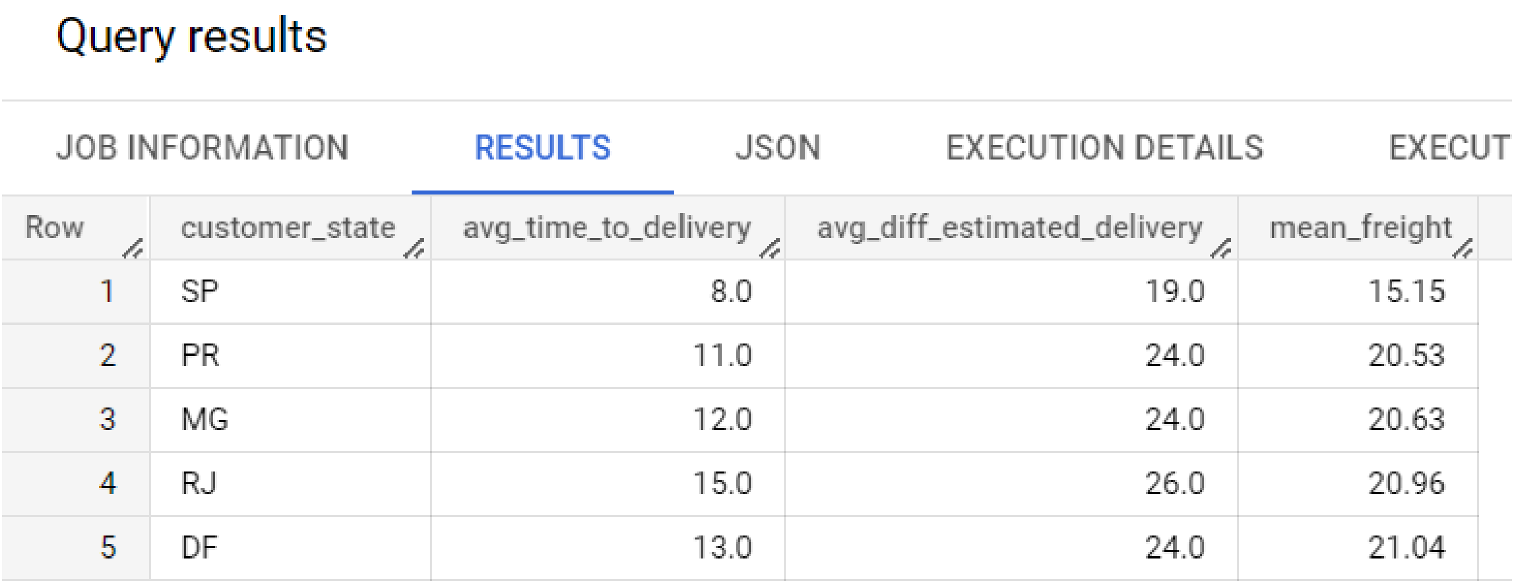


Row	customer_state	avg_time_to_delivery	avg_diff_estimated_delivery	mean_freight
1	RR	28.0	46.0	42.98
2	PB	20.0	33.0	42.72
3	RO	19.0	39.0	41.07
4	AC	20.0	41.0	40.07
5	PI	19.0	30.0	39.15

Top 5 states with lowest average freight value

Query:-

```
SELECT
 customer_state,
 ROUND(AVG(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))) AS avg_time_to_delivery,
 ROUND(AVG(DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY))) AS avg_diff_estimated_delivery,
 ROUND(AVG(freight_value),2) AS mean_freight
FROM `Target.customers` AS c
JOIN `Target.orders` AS o
ON c.customer_id = o.customer_id
JOIN `Target.order_items`AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
ORDER BY mean_freight ASC
LIMIT 5
```

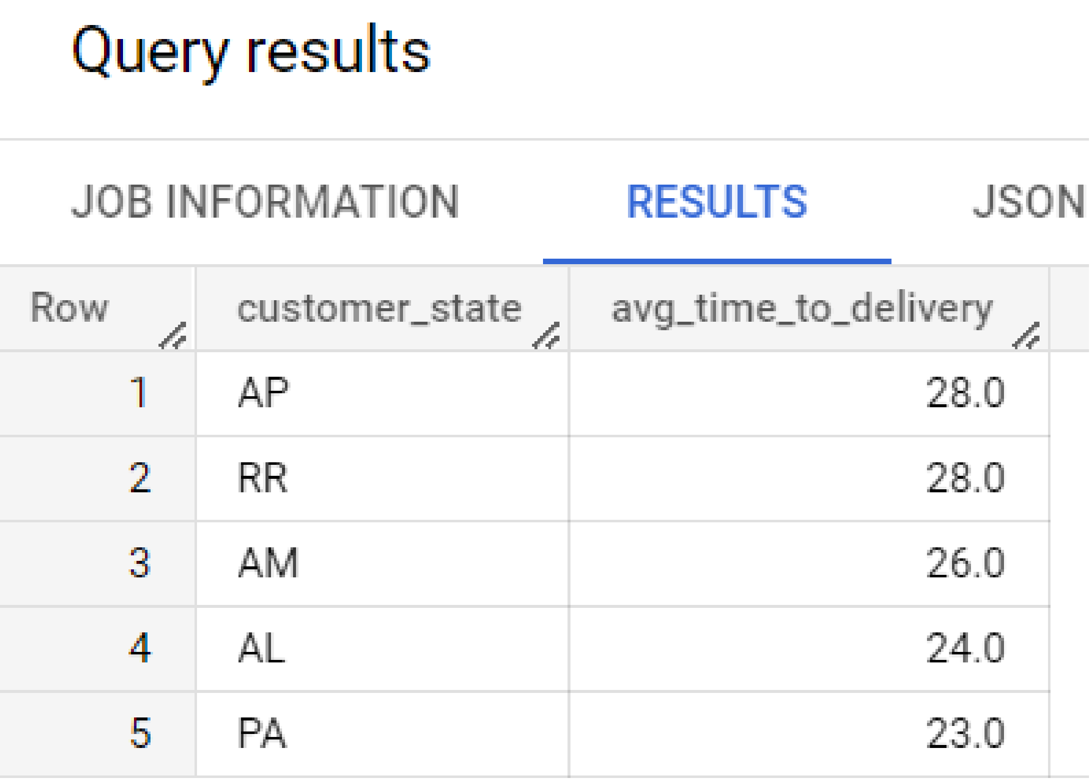


Row	customer_state	avg_time_to_delivery	avg_diff_estimated_delivery	mean_freight
1	SP	8.0	19.0	15.15
2	PR	11.0	24.0	20.53
3	MG	12.0	24.0	20.63
4	RJ	15.0	26.0	20.96
5	DF	13.0	24.0	21.04

Top 5 states with highest average time to delivery

```
SELECT
 customer_state,
 ROUND(AVG(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))) AS avg_time_to_delivery,
FROM `Target.customers` AS c
JOIN `Target.orders` AS o
```

```
ON c.customer_id = o.customer_id
JOIN `Target.order_items`AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
ORDER BY avg_time_to_delivery DESC
LIMIT 5
```

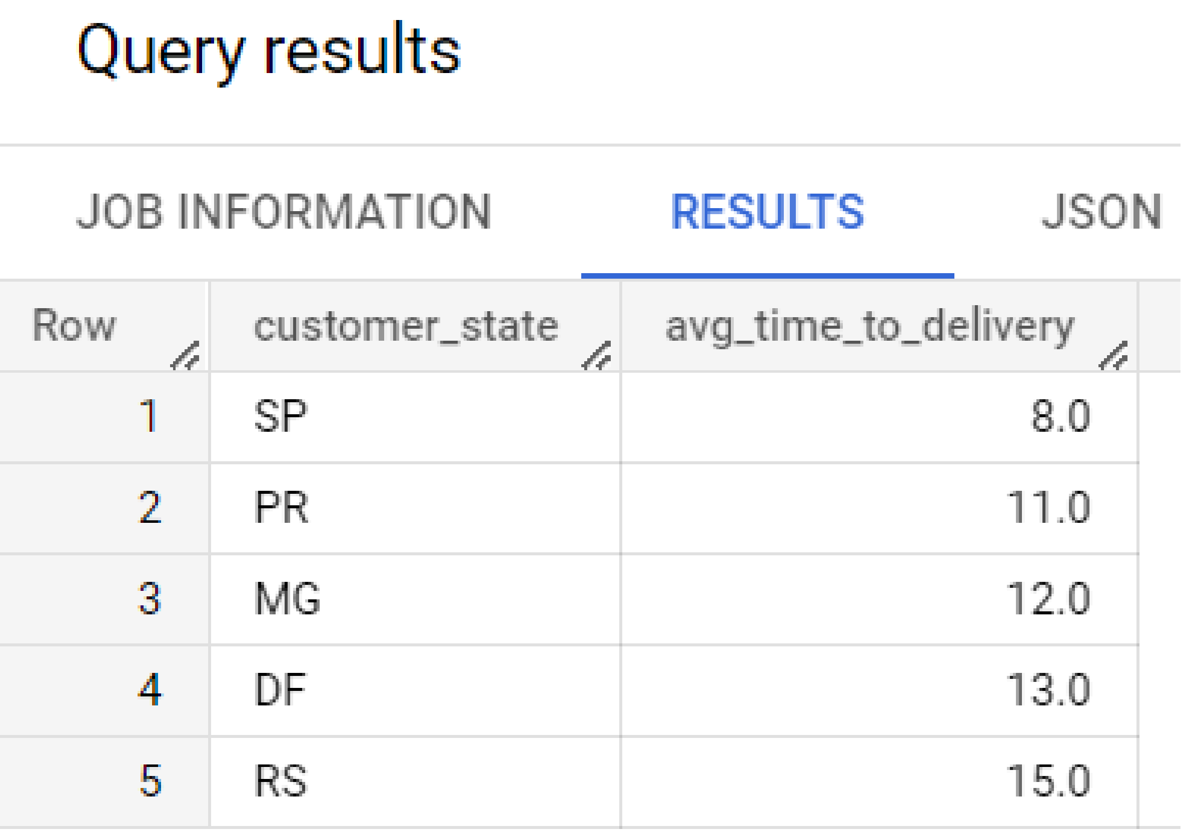


Query results

Row	customer_state	avg_time_to_delivery
1	AP	28.0
2	RR	28.0
3	AM	26.0
4	AL	24.0
5	PA	23.0

Top 5 states with lowest average time to delivery

```
SELECT
 customer_state,
 ROUND(AVG(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))) AS avg_time_to_delivery,
FROM `Target.customers` AS c
JOIN `Target.orders` AS o
ON c.customer_id = o.customer_id
JOIN `Target.order_items`AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
ORDER BY avg_time_to_delivery ASC
LIMIT 5
```

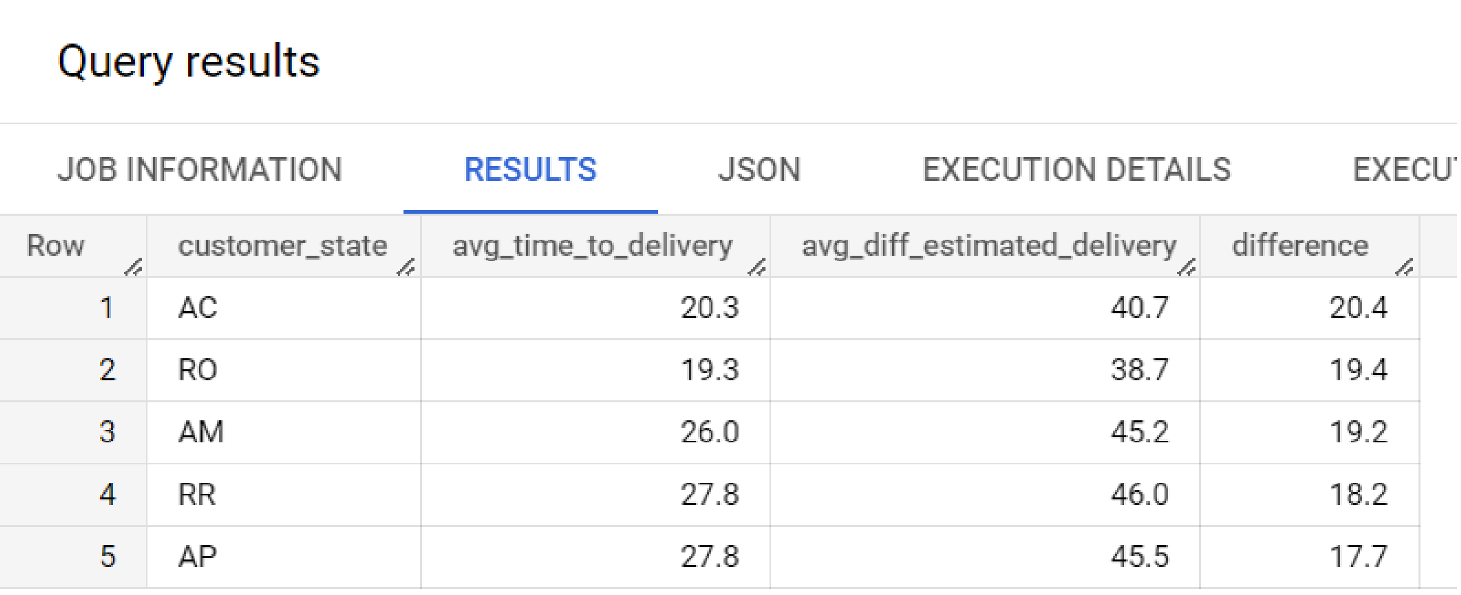


Query results

Row	customer_state	avg_time_to_delivery
1	SP	8.0
2	PR	11.0
3	MG	12.0
4	DF	13.0
5	RS	15.0

Top 5 states where delivery is really fast compared to estimated date

```
SELECT
 customer_state,
 ROUND(AVG(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)),1) AS avg_time_to_delivery,
 ROUND(AVG(DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY)),1) AS avg_diff_estimated_delivery,
 ROUND(AVG(DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY)) - AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)),1) A
S difference,
FROM `Target.customers` AS c
JOIN `Target.orders` AS o
ON c.customer_id = o.customer_id
JOIN `Target.order_items`AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
ORDER BY difference DESC
LIMIT 5
```



Row	customer_state	avg_time_to_delivery	avg_diff_estimated_delivery	difference
1	AC	20.3	40.7	20.4
2	RO	19.3	38.7	19.4
3	AM	26.0	45.2	19.2
4	RR	27.8	46.0	18.2
5	AP	27.8	45.5	17.7

Assumptions :

- Considering the order_delivered_customer_date and not order_delivered_carrier_date since it carrier can mark an order as delivered without it's actually being delivered. So, order_delivered_customer_date seems to be more accurate.

Here,

The actual delivery has been made much before the expected delivery date.

Insights :

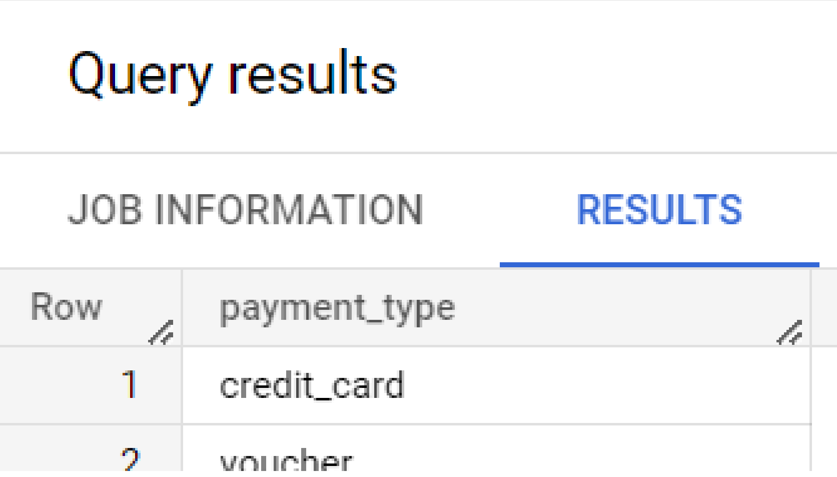
- Orders in states like Sao Paulo which has greater number of orders take lesser time to get delivered. They also have lower freight value.
- States like AP and RR have higher average freight value and higher average delivery days since they have lesser number of orders and are maybe distant(assumption).

Stage 6 : Payment type analysis

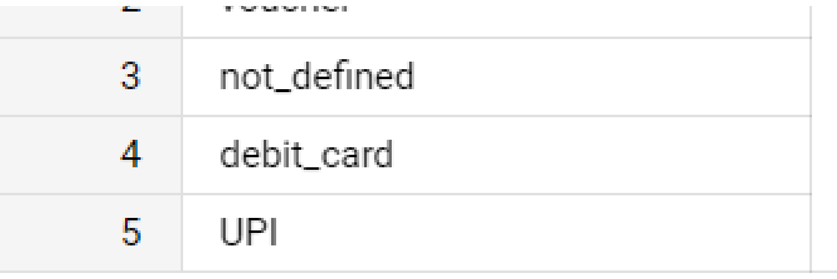
Payment modes used :

Query:-

```
SELECT
 DISTINCT payment_type
FROM `Target.payments`
```



Row	payment_type
1	credit_card
2	voucher

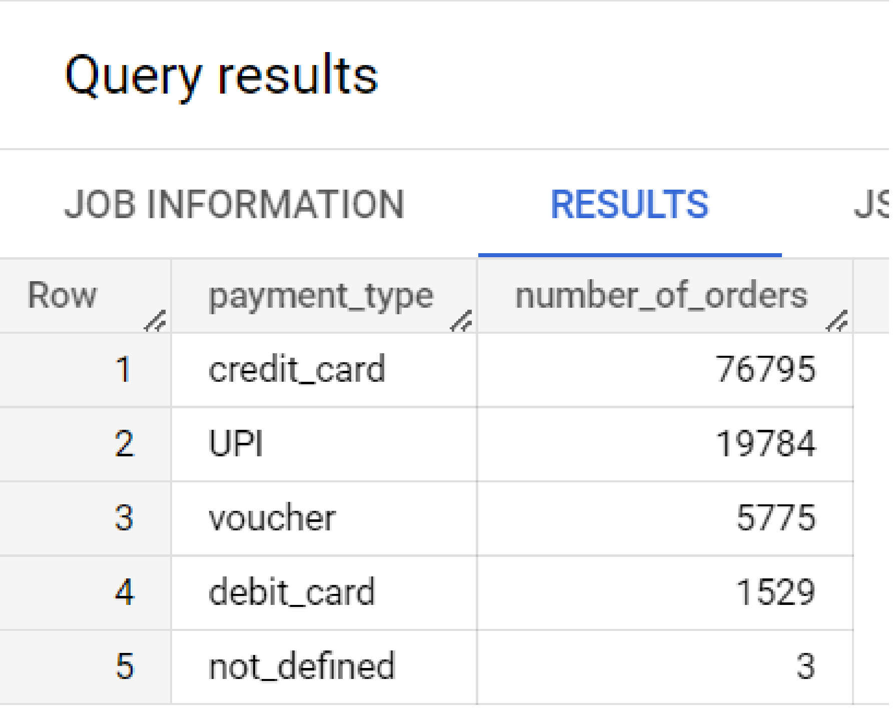


2	voucher
3	not_defined
4	debit_card
5	UPI

Number of orders made by different payment modes :

Query :

```
SELECT
 DISTINCT payment_type,
 COUNT(*) AS number_of_orders
FROM `Target.payments` AS p
JOIN `Target.orders`AS o
ON p.order_id = o.order_id
WHERE EXTRACT(MONTH FROM order_purchase_timestamp) IS NOT NULL
GROUP BY payment_type
ORDER BY number_of_orders DESC
```



Query results

Row	payment_type	number_of_orders
1	credit_card	76795
2	UPI	19784
3	voucher	5775
4	debit_card	1529
5	not_defined	3

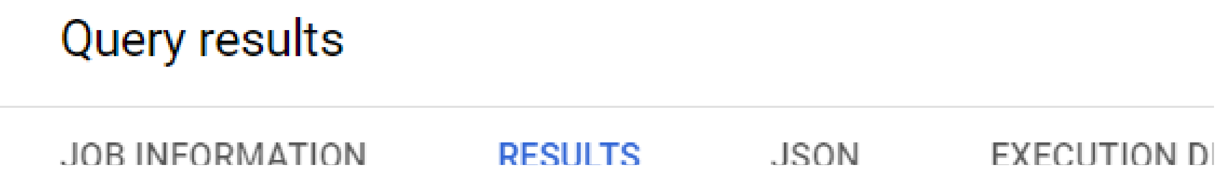
Insights :

- Most of the people in Brazil prefer paying for their orders through credit card.

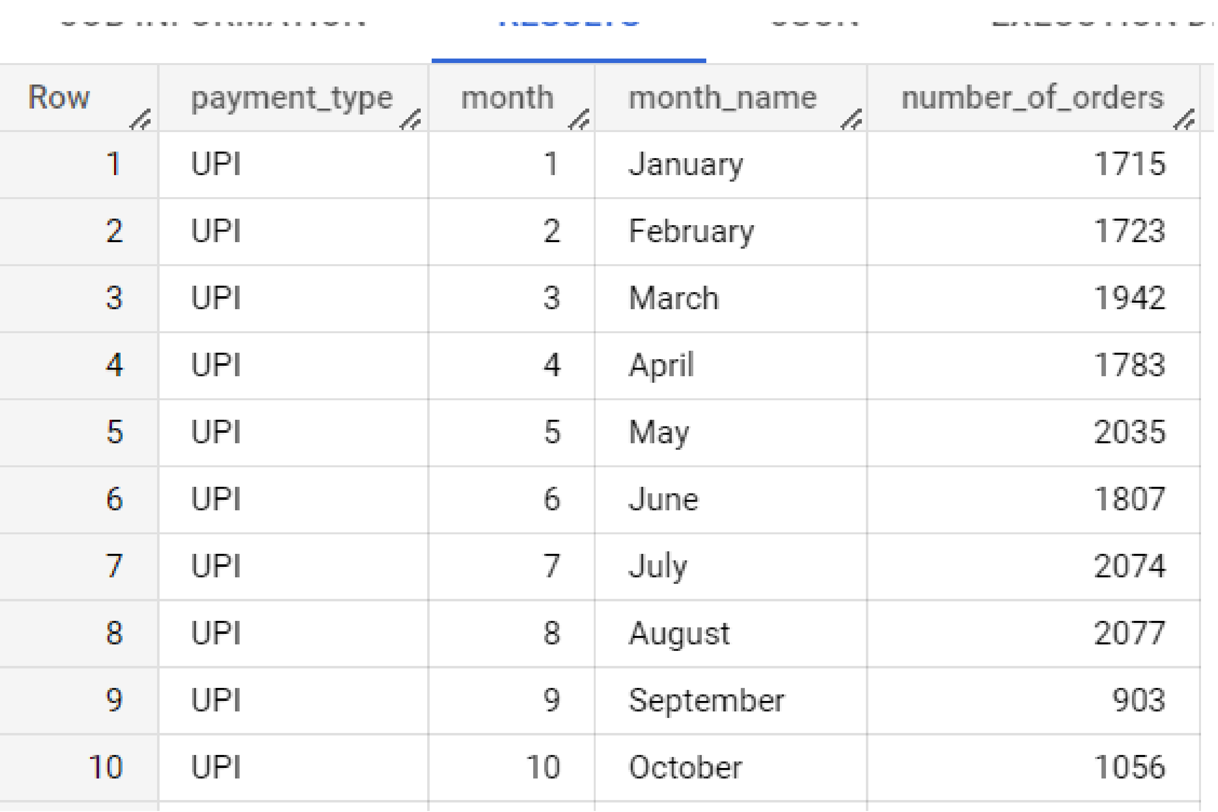
Month over Month count of orders for different payment types :

Query:-

```
SELECT
 DISTINCT payment_type,
 EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
 FORMAT_DATETIME("%B",order_purchase_timestamp) AS month_name,
 COUNT(*) AS number_of_orders
FROM `Target.payments` AS p
JOIN `Target.orders`AS o
ON p.order_id = o.order_id
WHERE EXTRACT(MONTH FROM order_purchase_timestamp) IS NOT NULL
GROUP BY payment_type, month, month_name
ORDER BY payment_type, month;
```



Query results

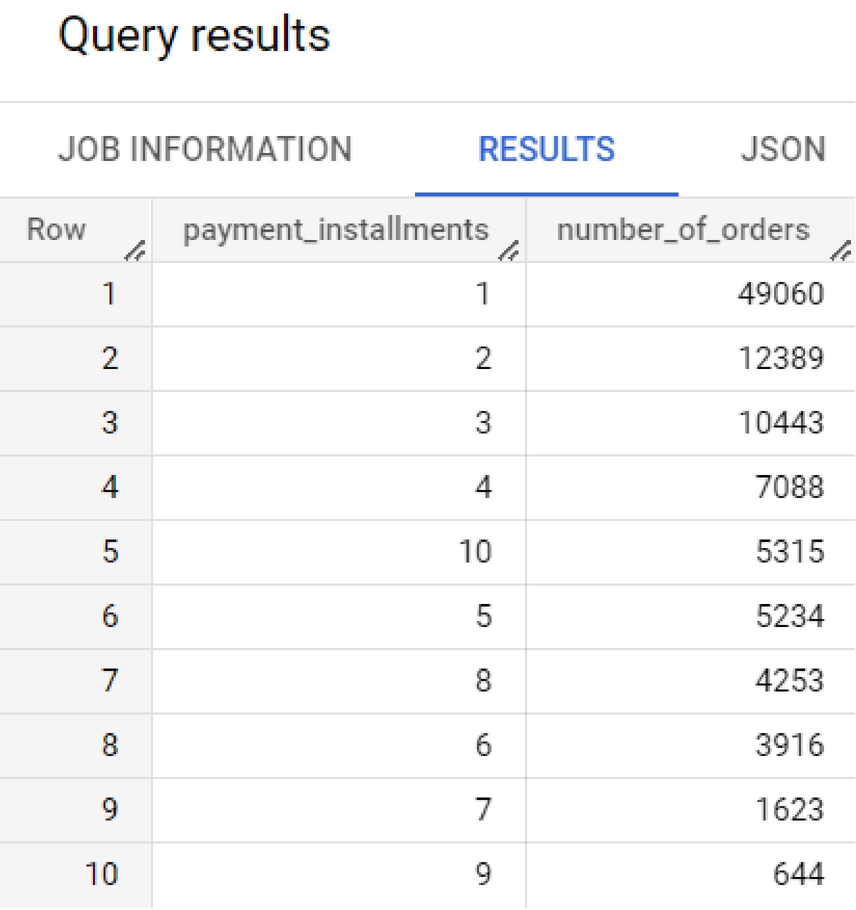


Row	payment_type	month	month_name	number_of_orders
1	UPI	1	January	1715
2	UPI	2	February	1723
3	UPI	3	March	1942
4	UPI	4	April	1783
5	UPI	5	May	2035
6	UPI	6	June	1807
7	UPI	7	July	2074
8	UPI	8	August	2077
9	UPI	9	September	903
10	UPI	10	October	1056

Count of orders based on the no. of payment installments

Query:-

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



Row	payment_installments	number_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

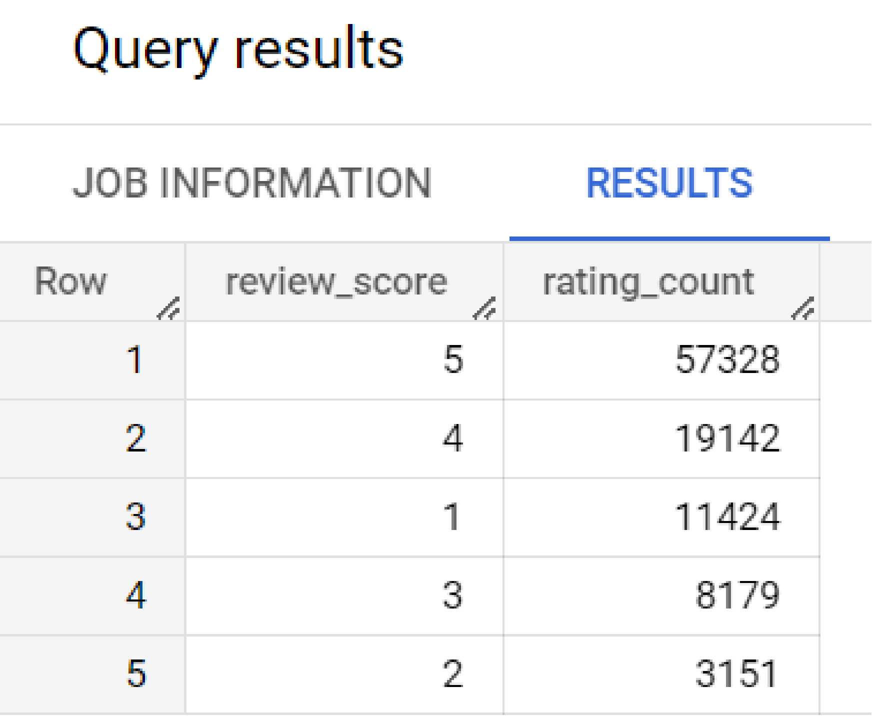
Insights :

- Almost 50% of the orders have been paid in the first installment.

Analyzing reviews :

Query :

```
SELECT
 review_score,
 COUNT(*) AS rating_count
FROM `Target.order_reviews`
GROUP BY review_score
ORDER BY rating_count DESC
```



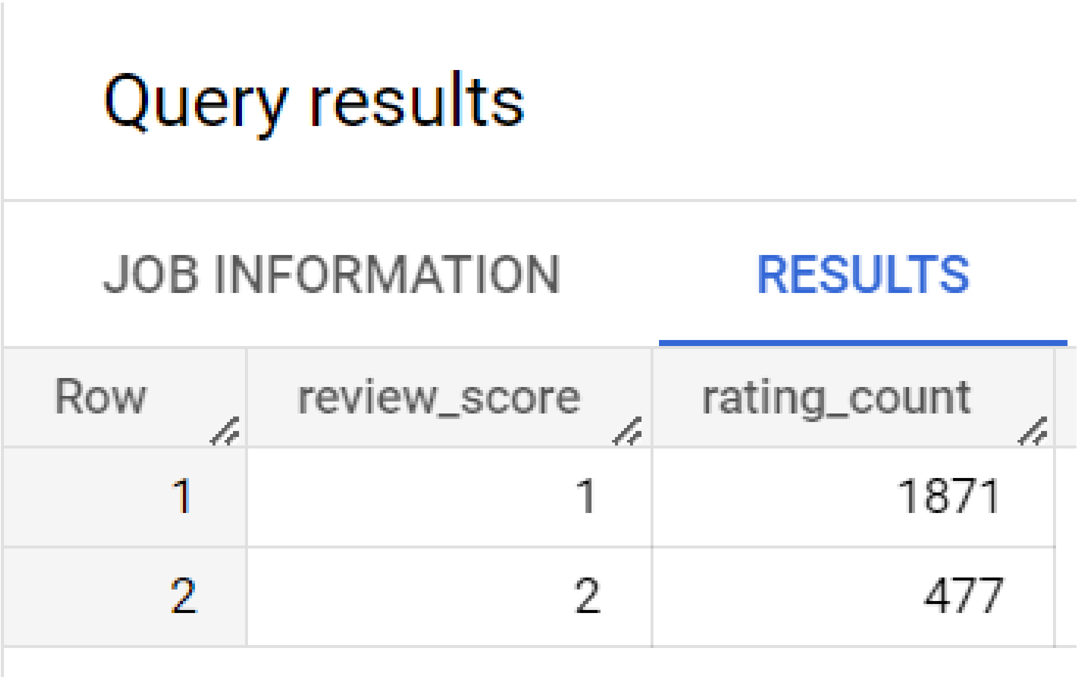
Query results

Row	review_score	rating_count
1	5	57328
2	4	19142
3	1	11424
4	3	8179
5	2	3151

Analyzing number of people who have rated(1 and 2) :

Query :

```
SELECT
 review_score,
 COUNT(*) AS rating_count
FROM `Target.order_reviews`
WHERE review_comment_title IS NOT NULL AND review_score IN (1, 2)
GROUP BY review_score
ORDER BY rating_count DESC
```



Query results

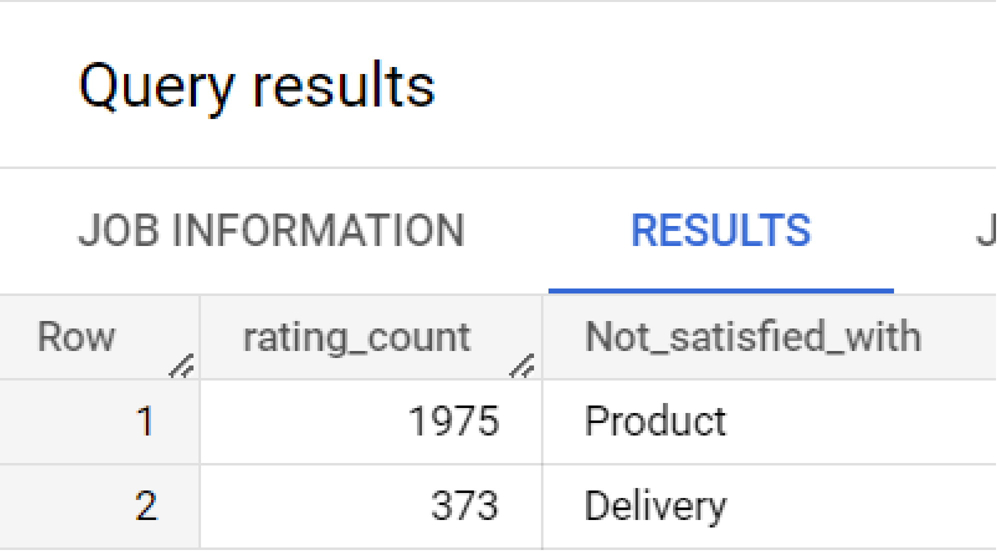
Row	review_score	rating_count
1	1	1871
2	2	477

Analysing probable reasons for ratings(1 and 2) :

Query :

```
SELECT
  COUNT(*) AS rating_count,
  CASE
   WHEN LOWER(review_comment_title) LIKE "%delivery%" OR LOWER(review_comment_title) LIKE "%late%" OR LOWER(review_comment_title) LIKE "%delay%" OR LOWER(review_comment_title) LIKE "%deliver%"OR  LOWER(review_comment_title) LIKE "%date%" OR LOWER(review_comment_title) LIKE "%await%" OR LOWER(review_comment_title) LIKE "%arrive%" OR LOWER(review_comment_title) LIKE "%time%" OR LOWER(review_comment_title) LIKE "%date%"
   THEN "Delivery"
   ELSE "Product"
  END AS Not_satisfied_with
 FROM `Target.order_reviews`
 WHERE review_comment_title IS NOT NULL AND review_score IN (1, 2)
```

```
GROUP BY Not_satisfied_with
```



Row	rating_count	Not_satisfied_with
1	1975	Product
2	373	Delivery

Assumptions :

- Assuming, there are majorly these two reasons a customer is not satisfied with.

Actionable Insights

After analyzing the complete dataset, some of the useful insights that I found out are :

- The e-commerce sector has shown considerable amount of growth in the time period provided and is likely to grow further.
- There is a decent difference between estimated delivery date and actual delivery date. Estimated delivery date is generally higher than the actual delivery date.
- Close to 56% orders are placed between 12:00:01 UTC to 23:00:00 UTC.
- Most of the people are comfortable in paying for their orders in one go.
- Most of the customers are concentrated in SP, RJ and MG.
- More than 55% have rated their purchase 5.
- Majority of people who have rated 1 and 2 are not satisfied with the product.

Recommendations

Some of the recommendations, I would give Target are :

- Estimated delivery date calculation algorithm needs to be fixed since it's pretty inaccurate.
- More targeted marketing campaigns can be run between 12:00:01 and 23:00:00 UTC since the people in Brazil mostly order within this time period.
- Since , most of the orders are from Sao Paulo , Rio De Janeiro and Minas Gerais, awareness needs to be spread in the rest of the states which hardly had people placing orders.
- Time taken to deliver is generally on the higher side which needs to be reduced by taking appropriate measures like automated logistics software, setting up new warehouses, inventory update etc.
- Since most of the people rating 1 and 2 are not satisfied with the product, product quality needs to be improved.
- Since 73% payments are made through credit cards, several benefits(like coupons, cashbacks and vouchers) must be provided to the credit card users to promote customer retention.