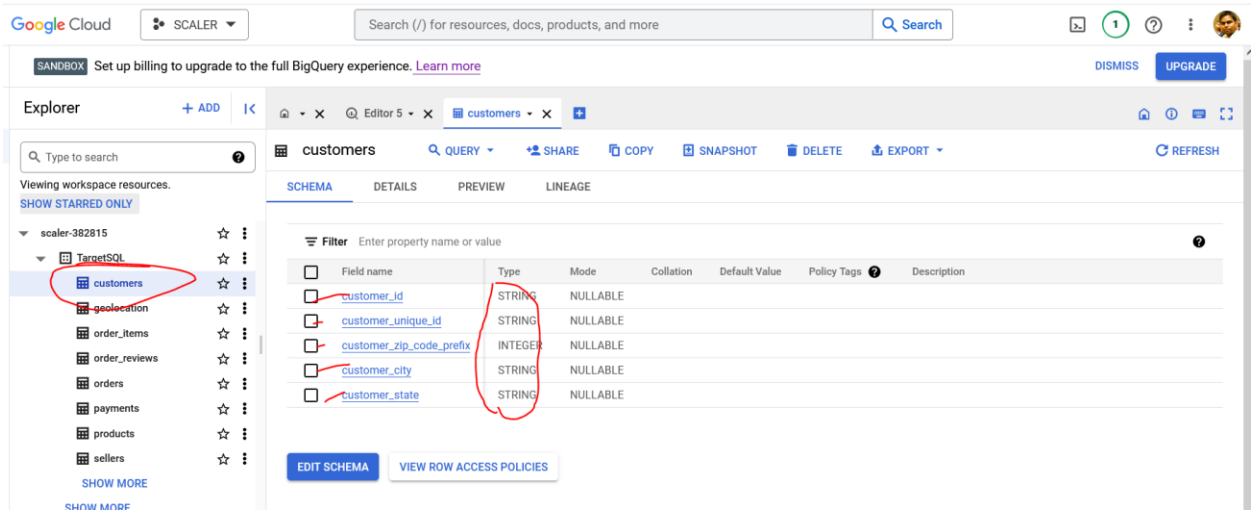
Business Case: Target SQL

```
-- Data type of columns in a table
```

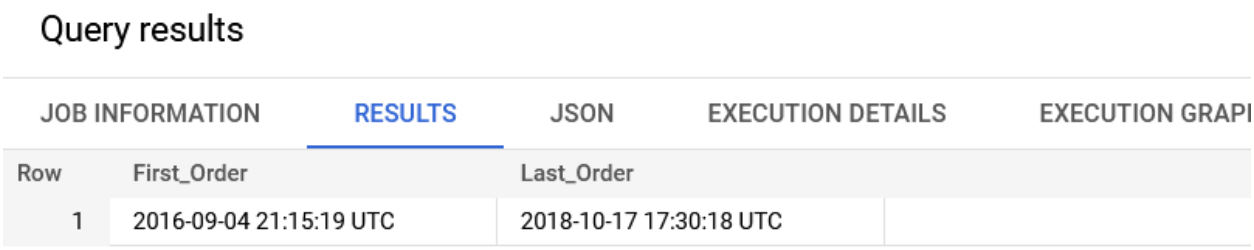
Snapshot:



```sql
-- Time preriod of which data is given

SELECT  MIN(order_purchase_timestamp) AS First_Order,
        MAX(order_purchase_timestamp) AS Last_Order
FROM `scaler-382815.TargetSQL.orders`
```

Snapshot:



- Cities and States of customers ordered during the given period

```
SELECT DISTINCT
    customer_city,
    customer_state
FROM `TargetSQL.orders` o
JOIN    `TargetSQL.customers` c
ON o.customer_id = c.customer_id
```

Snapshot:

Query results

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

Row	customer_city	customer_state
1	acu	RN
2	ico	CE
3	ipe	RS
4	ipu	CE
5	ita	SC
6	itu	SP
7	jau	SP
8	luz	MG
9	poa	SP
10	uba	MG
11	una	BA

```
-- Is there a growing trend on e-
commerce in Brazil? How can we describe a comple
te scenario? Can we see some seasonality with pe
aks at specific months?
```

There is a growing trend in number of transaction each year. **It seems target started its business in Brazil in the last few months of 2016(hence less number in 2016)**.

It is also visible that there was huge demand in the months of 5th, 7th and 8th. It may be because of the seasonality.

There is a festival called "Festival de Cachaça- Drink, Dance & Be Merry" that occurs in mid Aug, would have impacted the sales in July & August Months.

Yearly Trends:

```sql
SELECT DISTINCT
    EXTRACT(year FROM order_purchase_timestamp) yr,
    COUNT(order_id) AS TOTAL_orders
FROM
`TargetSQL.orders`
GROUP BY yr
ORDER BY yr
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	yr	TOTAL_orders
1	2016	329
2	2017	45101
3	2018	54011

Monthly Trends:

```sql
SELECT DISTINCT
    EXTRACT(month FROM order_purchase_timestamp) mnth,
    COUNT(order_id) AS TOTAL_orders
FROM
`TargetSQL.orders`
GROUP BY mnth
ORDER BY mnth
```

Snapshot: Monthly trends

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	mnth	TOTAL_orders
2	2	8508
3	3	9893
4	4	9343
5	5	10573
6	6	9412
7	7	10318
8	8	10843

- What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

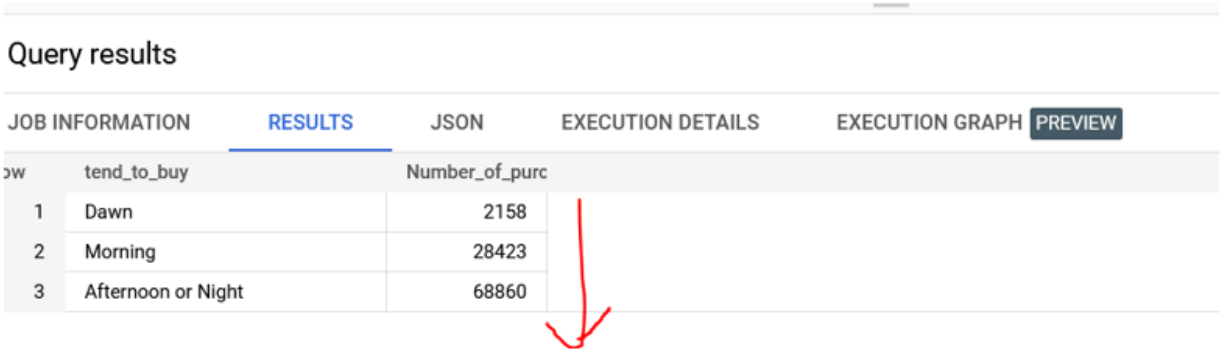
```
WITH CTE AS (SELECT
    CASE WHEN EXTRACT(hour FROM order_purchase_timestamp) BETWEEN 1 AND 4 THEN "Dawn"
         WHEN EXTRACT(hour FROM order_purchase_timestamp) BETWEEN 5 AND 12 THEN "Morni
ng"
         ELSE "Afternoon or Night"
         END AS tend_to_buy
  FROM `TargetSQL.orders`)

  SELECT
    tend_to_buy,
    COUNT(*) AS Number_of_purchase
  FROM CTE
  GROUP BY tend_to_buy
  ORDER BY Number_of_purchase
```

Brazilian customers tend to buy in evening or night (after 12PM).

Note: I have considered dawn (from 1 to 4), Morning (from 5 to 12) and rest Evening or Night

Snapshot:



```
--      Get month on month orders by states

SELECT DISTINCT
    geolocation_state,
    EXTRACT(month FROM order_purchase_timestamp) AS mnth,
    COUNT(order_id) OVER(PARTITION BY geolocation_state ORDER BY EXTRACT(month FROM or
der_purchase_timestamp)) AS sles
FROM `TargetSQL.orders`o
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.geolocation` g
ON c.customer_zip_code_prefix = g.geolocation_zip_code_prefix
```

Snapshot:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

ow	geolocation_state	mnth	sles
1	AM	1	392
2	AM	2	944
3	AM	3	1369
4	AM	4	2145
5	AM	5	2882
6	AM	6	3314
7	AM	7	4326
8	AM	8	4722
9	AM	9	5055
10	AM	10	5104
11	AM	11	5463

- Distribution of customers across the states in Brazil

```
SELECT DISTINCT
    geolocation_state,
    COUNT(c.customer_id) OVER(PARTITION BY geolocation_state) AS total_cust_cnt
FROM `TargetSQL.orders`o
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.geolocation` g
ON c.customer_zip_code_prefix = g.geolocation_zip_code_prefix
ORDER BY total_cust_cnt
```

Snapshot:

Query results

JOB INFORMATION | RESULTS | JSON

ow	geolocation_state	total_cust_cnt
1	RR	2087
2	AP	4912
3	AM	5587
4	AC	7688
5	TO	17509
6	RN	20595
7	RO	21244
8	PI	23913
9	SE	24584
10	PB	27714
11	AL	24861

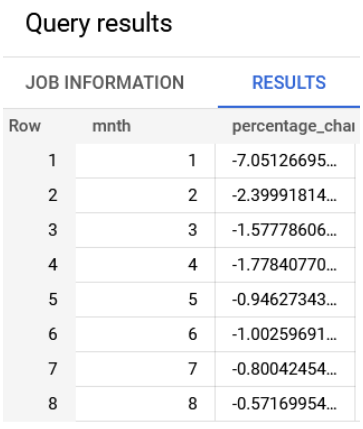
Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

```
WITH CTE1 AS (SELECT
EXTRACT(month FROM order_purchase_timestamp) AS mnth_17,
SUM(payment_value) AS payment_value_17
FROM `TargetSQL.payments` p
JOIN `TargetSQL.orders` o
ON p.order_id = o.order_id
WHERE order_purchase_timestamp BETWEEN "2017-01-01" AND "2017-08-31"
GROUP BY mnth_17
ORDER BY mnth_17),

CTE2 AS (SELECT
EXTRACT(month FROM order_purchase_timestamp) AS mnth_18,
SUM(payment_value) AS payment_value_18
FROM `TargetSQL.payments` p
JOIN `TargetSQL.orders` o
ON p.order_id = o.order_id
WHERE order_purchase_timestamp BETWEEN "2018-01-01" AND "2018-08-31"
GROUP BY mnth_18
ORDER BY mnth_18)

SELECT
    mnth_17 AS mnth,
    (payment_value_17-payment_value_18)/payment_value_17 AS percentage_change
FROM CTE1 AS CTE1
JOIN CTE2 AS CTE2
ON CTE1.mnth_17=CTE2.mnth_18
ORDER BY mnth
```

Snapshot:



Query results

JOB INFORMATION | RESULTS

Row	mnth	percentage_chai
1	1	-7.05126695...
2	2	-2.39991814...
3	3	-1.57778606...
4	4	-1.77840770...
5	5	-0.94627343...
6	6	-1.00259691...
7	7	-0.80042454...
8	8	-0.57169954...

- Mean & Sum of price and freight value by customer state

```
SELECT DISTINCT
        customer_state,
    ROUND(AVG(payment_value) OVER(PARTITION BY customer_state),2) AS mean_price,
    ROUND(AVG(freight_value) OVER(PARTITION BY customer_state),2) AS mean_freight,
    ROUND(SUM(payment_value) OVER(PARTITION BY customer_state),2) AS total_price,
    ROUND(SUM(freight_value) OVER(PARTITION BY customer_state),2) AS total_freight
FROM `TargetSQL.orders`o
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.payments`p
ON o.order_id = p.order_id
JOIN `TargetSQL.order_items`oi
ON oi.order_id = o.order_id
```

Snapshot:

Query results

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

ow	customer_state	mean_price	mean_freight	total_price	total_freight
1	AM	203.24	33.08	34753.3	5656.54
2	RS	176.89	21.83	1147277.0	141579.69
3	MT	228.27	28.97	256804.62	32592.32
4	RO	230.37	40.97	65886.0	11717.47
5	GO	211.47	22.73	513879.0	55237.53
6	PE	199.25	32.78	376377.27	61923.56
7	PB	283.23	43.26	180984.19	27641.72
8	BA	196.99	26.32	797410.36	106538.62
9	TO	213.22	39.68	72281.17	13450.6
10	MG	170.56	20.63	2326151.64	281301.31
11	RR	239.66	42.98	12462.21	2235.19

- 5.1 Calculate days between purchasing, delivering and estimated delivery

```
SELECT
    DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
      pur_vs_delivery,

      DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date, day) AS
      delivery_vs_Estimated
FROM `TargetSQL.orders`
```

Snapshot:

JOB INFORMATION | RESULTS

Row	pur_vs_delivery	delivery_vs_Esti
1	30	-12
2	30	28
3	35	16
4	30	1
5	32	0
6	29	1
7	43	-4
8	40	-4
9	37	-1
10	33	-5

Recommendations: 6.5% deliveries are getting delayed, Target can work on that to reduce the missed deleveries.

2965 customers `order_delivered_customer_date is missing in the table`

- 5.2 Find time_to_delivery & diff_estimated_delivery

```
SELECT
    DATE_DIFF(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to_delivery,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS diff_estimated_delivery
FROM `TargetSQL.orders`
```

Snapshot:

Query results

JOB INFORMATION | RESULTS

Row	time_to_delivery	diff_estimated_c
1	-30	-12
2	-30	28
3	-35	16
4	-30	1
5	-32	0
6	-29	1
7	-43	-4
8	-40	-4
9	-37	-1
10	-33	-5
11	-38	-6

- 5.3 Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

```
SELECT
    geolocation_state,
    ROUND(AVG(freight_value) OVER(PARTITION BY geolocation_state),2) AS mean_of_freigh
t_value,
    DATE_DIFF(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to
_delivery,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS di
ff_estimated_delivery
FROM `TargetSQL.orders` o
JOIN `TargetSQL.order_items`oi
ON o.order_id = oi.order_id
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.geolocation`g
ON c.customer_zip_code_prefix = g.geolocation_zip_code_prefix
```

Snapshot:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXE

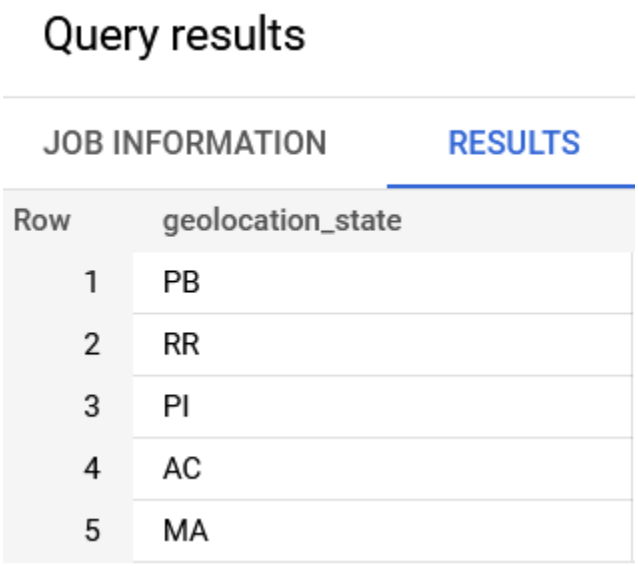
ow	geolocation_state	mean_of_freight	time_to_delivery	diff_estimated_c
1	RS	21.52	-41	-13
2	RS	21.52	-41	-13
3	RS	21.52	-41	-13
4	RS	21.52	-41	-13
5	RS	21.52	-41	-13
6	RS	21.52	-41	-13
7	RS	21.52	-41	-13
8	RS	21.52	-41	-13
9	RS	21.52	-41	-13
10	RS	21.52	-36	-13
11	RS	21.52	-36	-13

- Top 5 states with highest average freight valu e

```
WITH CTE AS(
    SELECT
    geolocation_state,
    freight_value,
    DATE_DIFF(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to
_delivery,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS di
ff_estimated_delivery
FROM `TargetSQL.orders` o
JOIN `TargetSQL.order_items`oi
ON o.order_id = oi.order_id
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.geolocation`g
ON c.customer_zip_code_prefix = g.geolocation_zip_code_prefix)

SELECT
    geolocation_state
FROM CTE
GROUP BY geolocation_state
ORDER BY AVG(freight_value) DESC
LIMIT 5
```

Snapshot:



Query results

JOB INFORMATION | RESULTS

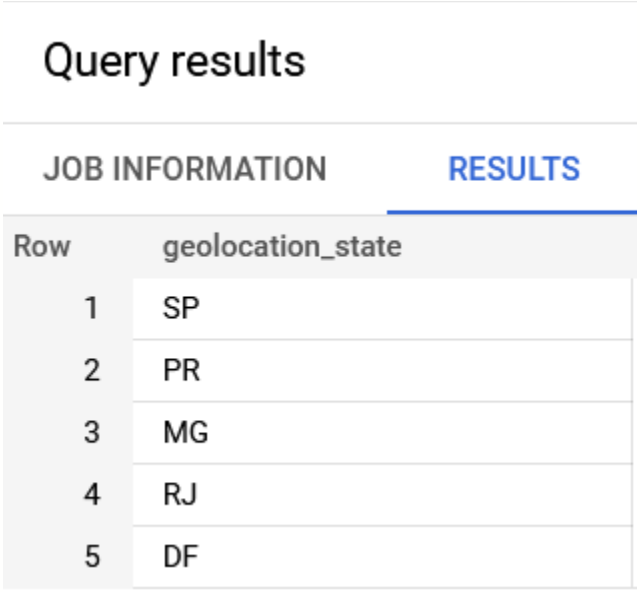
Row	geolocation_state
1	PB
2	RR
3	PI
4	AC
5	MA

- Top 5 states with lowest average freight value

```
WITH CTE AS(
    SELECT
    geolocation_state,
    freight_value,
    DATE_DIFF(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to
_delivery,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS di
ff_estimated_delivery
FROM `TargetSQL.orders` o
JOIN `TargetSQL.order_items`oi
ON o.order_id = oi.order_id
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.geolocation`g
ON c.customer_zip_code_prefix = g.geolocation_zip_code_prefix)

SELECT
    geolocation_state
FROM CTE
GROUP BY geolocation_state
ORDER BY AVG(freight_value) ASC
LIMIT 5
```

Snapshot:



Query results

JOB INFORMATION | RESULTS

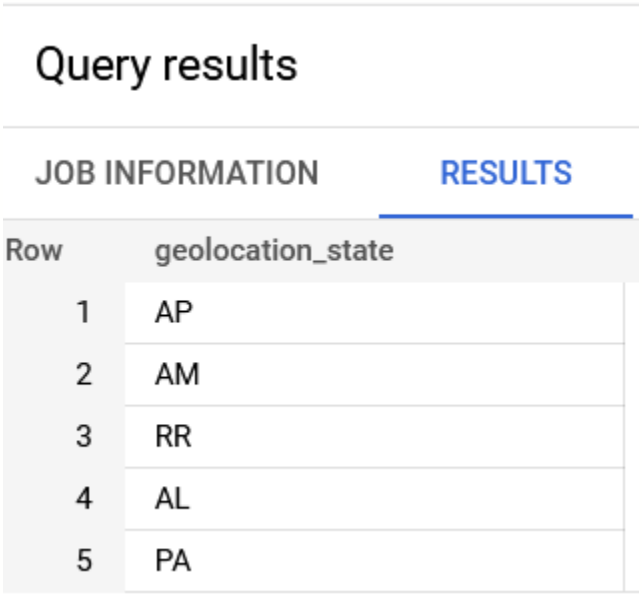
Row	geolocation_state
1	SP
2	PR
3	MG
4	RJ
5	DF

- Top 5 states with highest average time to deli very

```
WITH CTE AS(
    SELECT
    geolocation_state,
    freight_value,
    DATE_DIFF(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to
_delivery,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS di
ff_estimated_delivery
FROM `TargetSQL.orders` o
JOIN `TargetSQL.order_items`oi
ON o.order_id = oi.order_id
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.geolocation`g
ON c.customer_zip_code_prefix = g.geolocation_zip_code_prefix)

SELECT
    geolocation_state
FROM CTE
GROUP BY geolocation_state
ORDER BY AVG(time_to_delivery) ASC
LIMIT 5
```

Snapshot:



Query results

JOB INFORMATION | RESULTS

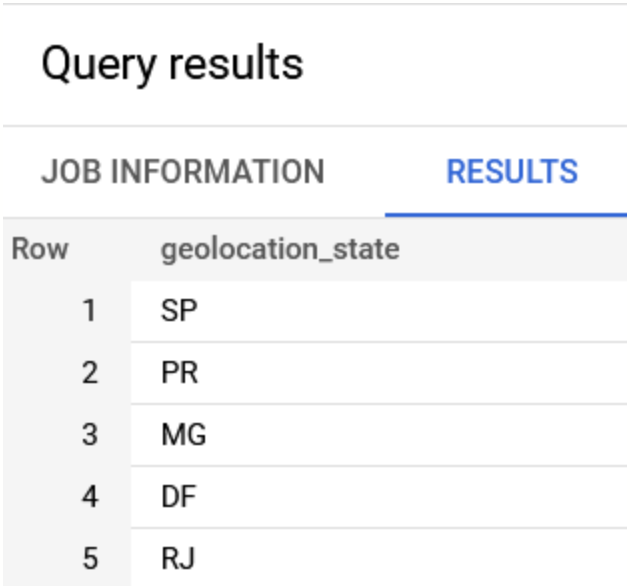
Row	geolocation_state
1	AP
2	AM
3	RR
4	AL
5	PA

- Top 5 states with Lowest average time to deliv ery

```
WITH CTE AS(
    SELECT
    geolocation_state,
    freight_value,
    DATE_DIFF(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to
_delivery,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS di
ff_estimated_delivery
FROM `TargetSQL.orders` o
JOIN `TargetSQL.order_items`oi
ON o.order_id = oi.order_id
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.geolocation`g
ON c.customer_zip_code_prefix = g.geolocation_zip_code_prefix)

SELECT
    geolocation_state
FROM CTE
GROUP BY geolocation_state
ORDER BY AVG(time_to_delivery) DESC
LIMIT 5
```

Snapshot:



Query results

JOB INFORMATION | RESULTS

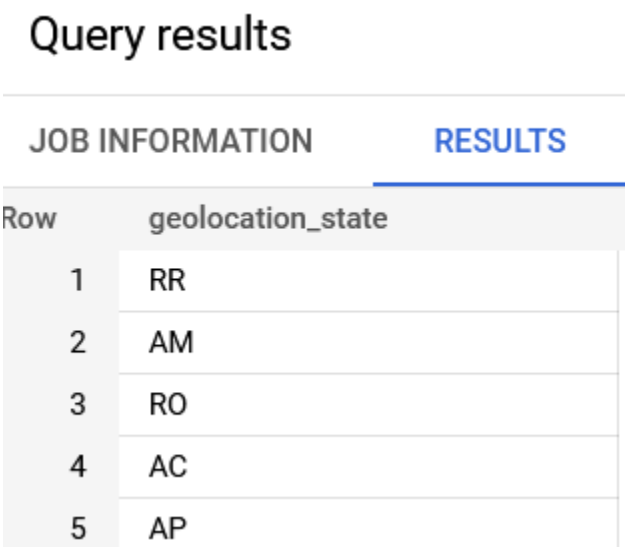
Row	geolocation_state
1	SP
2	PR
3	MG
4	DF
5	RJ

- Top 5 states where delivery is really fast

```
WITH CTE AS(
    SELECT
    geolocation_state,
    freight_value,
    DATE_DIFF(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to
_delivery,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS di
ff_estimated_delivery
FROM `TargetSQL.orders` o
JOIN `TargetSQL.order_items`oi
ON o.order_id = oi.order_id
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.geolocation`g
ON c.customer_zip_code_prefix = g.geolocation_zip_code_prefix)

SELECT
    geolocation_state
    -- ,
    -- AVG(freight_value) AS mean_of_freight_value,
    -- AVG(time_to_delivery) AS mean_time_to_delivery,
    -- AVG(diff_estimated_delivery) AS mean_diff_estimated_delivery
FROM CTE
GROUP BY geolocation_state
ORDER BY AVG(diff_estimated_delivery) DESC
LIMIT 5
```

Snapshot:



Query results

JOB INFORMATION | RESULTS

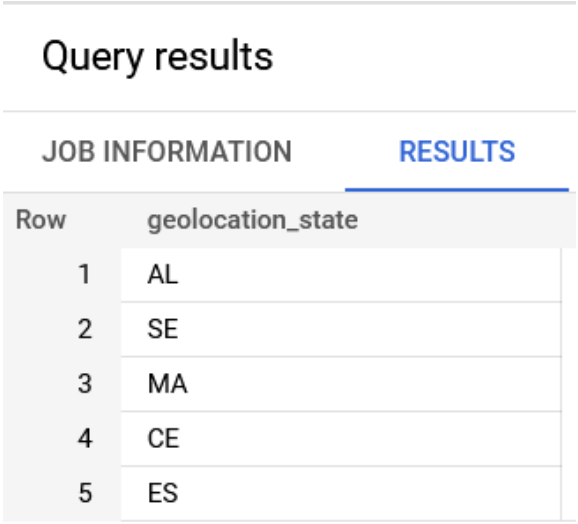
Row	geolocation_state
1	RR
2	AM
3	RO
4	AC
5	AP

- Top 5 states where delivery is not so fast compared to estimated date

```
WITH CTE AS(
    SELECT
    geolocation_state,
    freight_value,
    DATE_DIFF(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to
_delivery,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS di
ff_estimated_delivery
FROM `TargetSQL.orders` o
JOIN `TargetSQL.order_items`oi
ON o.order_id = oi.order_id
JOIN `TargetSQL.customers`c
ON o.customer_id = c.customer_id
JOIN `TargetSQL.geolocation`g
ON c.customer_zip_code_prefix = g.geolocation_zip_code_prefix)

SELECT
    geolocation_state
    -- ,
    -- AVG(freight_value) AS mean_of_freight_value,
    -- AVG(time_to_delivery) AS mean_time_to_delivery,
    -- AVG(diff_estimated_delivery) AS mean_diff_estimated_delivery
FROM CTE
GROUP BY geolocation_state
ORDER BY AVG(diff_estimated_delivery) ASC
LIMIT 5
```

Snapshot:



Query results

JOB INFORMATION | RESULTS

Row	geolocation_state
1	AL
2	SE
3	MA
4	CE
5	ES

- Month over Month count of orders for different payment types

```
WITH CTE AS (SELECT DISTINCT
    EXTRACT(MONTH FROM order_purchase_timestamp) AS mnth,
    EXTRACT(YEAR FROM order_purchase_timestamp) yr,
    payment_type,
    COUNT(*) OVER(PARTITION BY EXTRACT(YEAR FROM order_purchase_timestamp),EXTRACT(MON
TH FROM order_purchase_timestamp), payment_type) AS total_orders
FROM `TargetSQL.payments` p
JOIN `TargetSQL.orders`o
ON p.order_id = o.order_id)

SELECT CONCAT(mnth,"'",yr) AS MOM,
payment_type,
total_orders
FROM CTE
ORDER BY yr, mnth
```

Snapshot:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXE(

Row	MOM	payment_type	total_orders
1	9'2016	credit_card	3
2	10'2016	UPI	63
3	10'2016	debit_card	2
4	10'2016	voucher	23
5	10'2016	credit_card	254
6	12'2016	credit_card	1
7	1'2017	UPI	197
8	1'2017	debit_card	9
9	1'2017	voucher	61
10	1'2017	credit_card	583
11	2'2017	debit_card	13

- Count of orders based on the no. of payment in stalments