‘Target-SQL’ - Project Queries

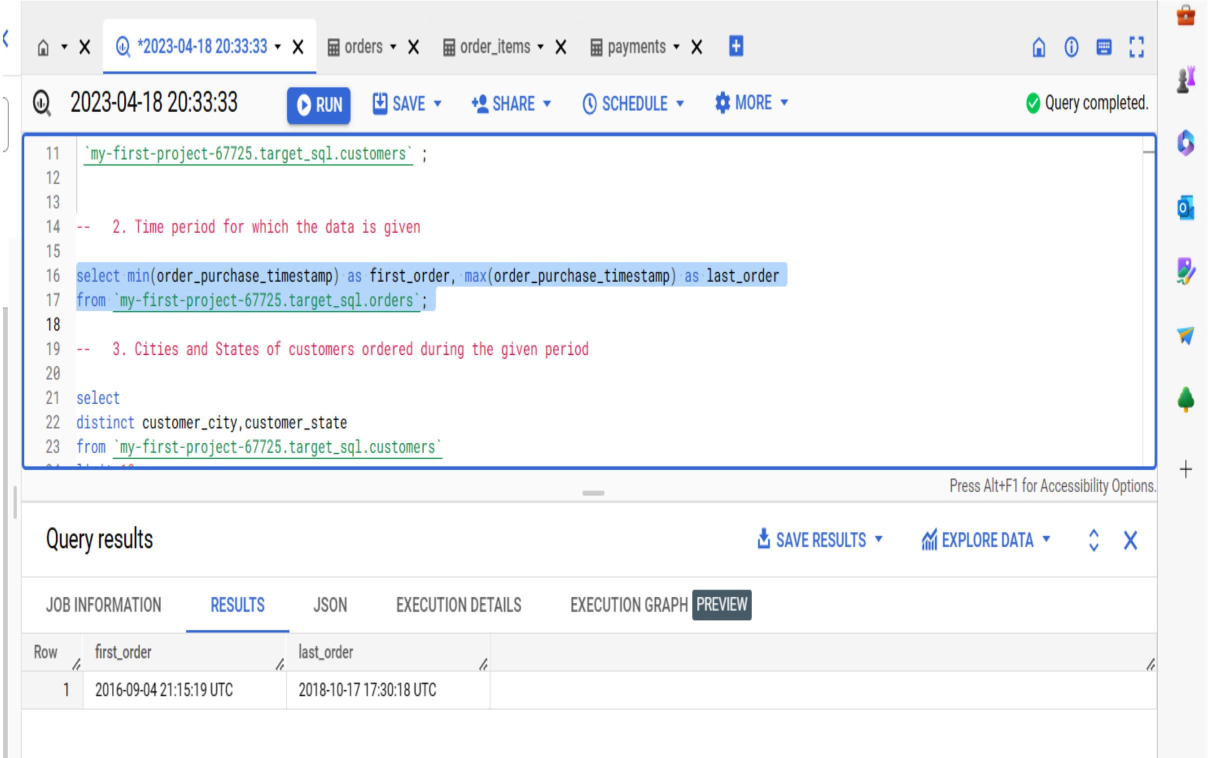
- 1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset

-- 1. Data type of columns in a table

```
SELECT column_name, data_type
FROM `my-first-project-67725.target_sql.customers`.INFORMATION_SCHEMA.COLUMNS;
```

-- 2. Time period for which the data is given

```
select min(order_purchase_timestamp) as first_order, max(order_purchase_timestamp) as last_order
from `my-first-project-67725.target_sql.orders`;
```

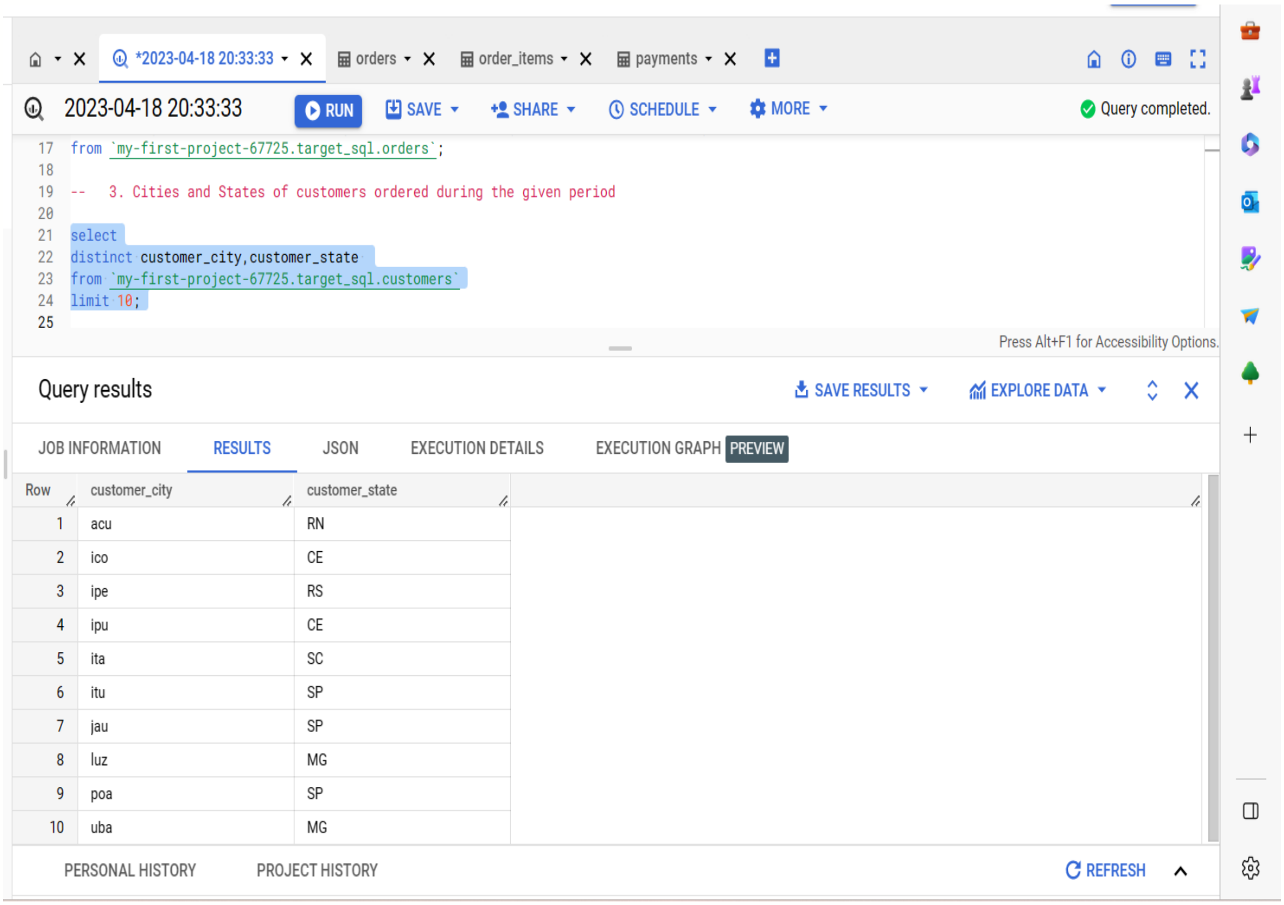


```
`my-first-project-67725.target_sql.customers` ;


--   2. Time period for which the data is given

select min(order_purchase_timestamp) as first_order, max(order_purchase_timestamp) as last_order
from `my-first-project-67725.target_sql.orders`;

--   3. Cities and States of customers ordered during the given period

select
distinct customer_city,customer_state
from `my-first-project-67725.target_sql.customers`
```

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

‘Target-SQL’ - Project Queries

```
--   3. Cities and States of customers ordered during the given period

select
distinct customer_city,customer_state
from `my-first-project-67725.target_sql.customers`
limit 10;
```



```
from `my-first-project-67725.target_sql.orders`;

--   3. Cities and States of customers ordered during the given period

select
distinct customer_city,customer_state
from `my-first-project-67725.target_sql.customers`
limit 10;
```

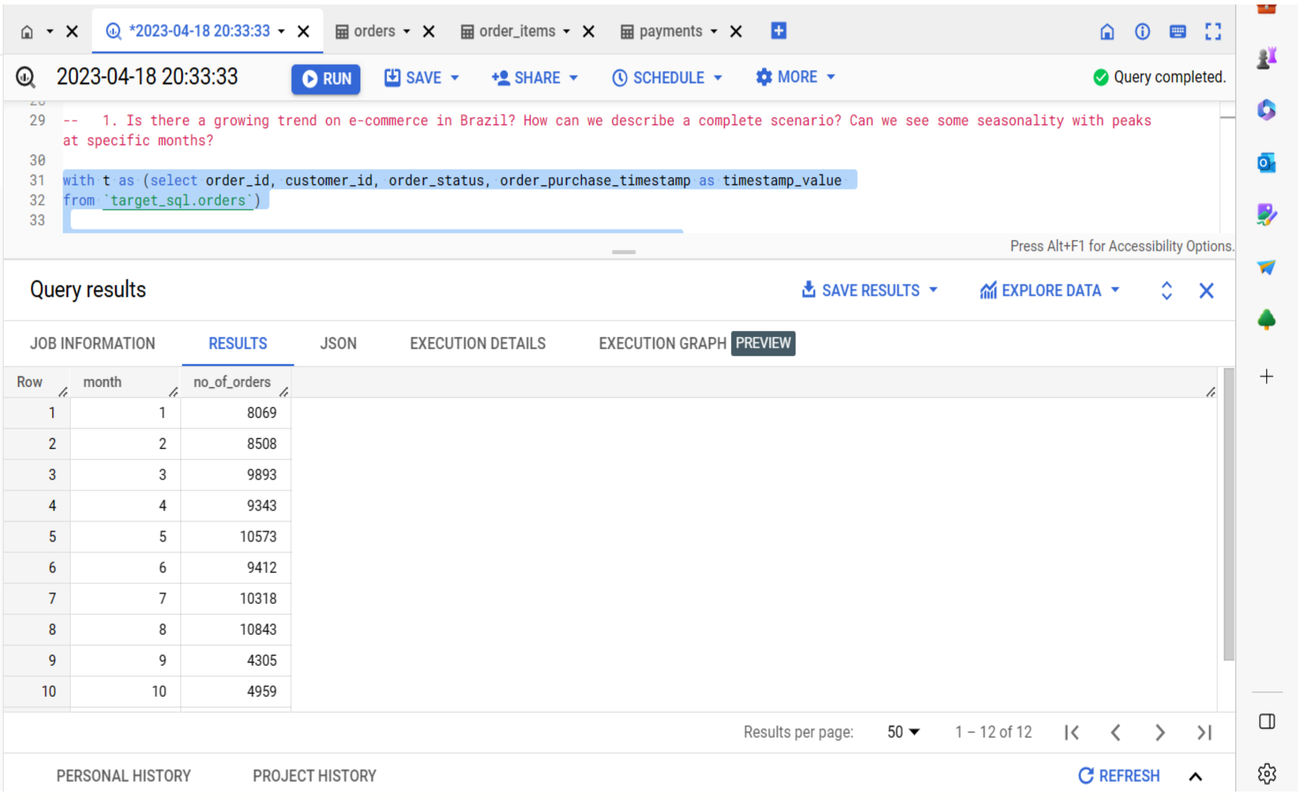
Row	customer_city	customer_state
1	acu	RN
2	ico	CE
3	ipe	RS
4	ipu	CE
5	ita	SC
6	itu	SP
7	jau	SP
8	luz	MG
9	poa	SP
10	uba	MG

‘Target-SQL’ - Project Queries

-- 2. In-depth Exploration:

-- 1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

```
with t as (select order_id, customer_id, order_status, order_purchase_timestamp as timestamp_value
from `target_sql.orders`)

select EXTRACT(month from timestamp_value) as month, count(*) as no_of_orders
from t
group by 1
order by 1;
```

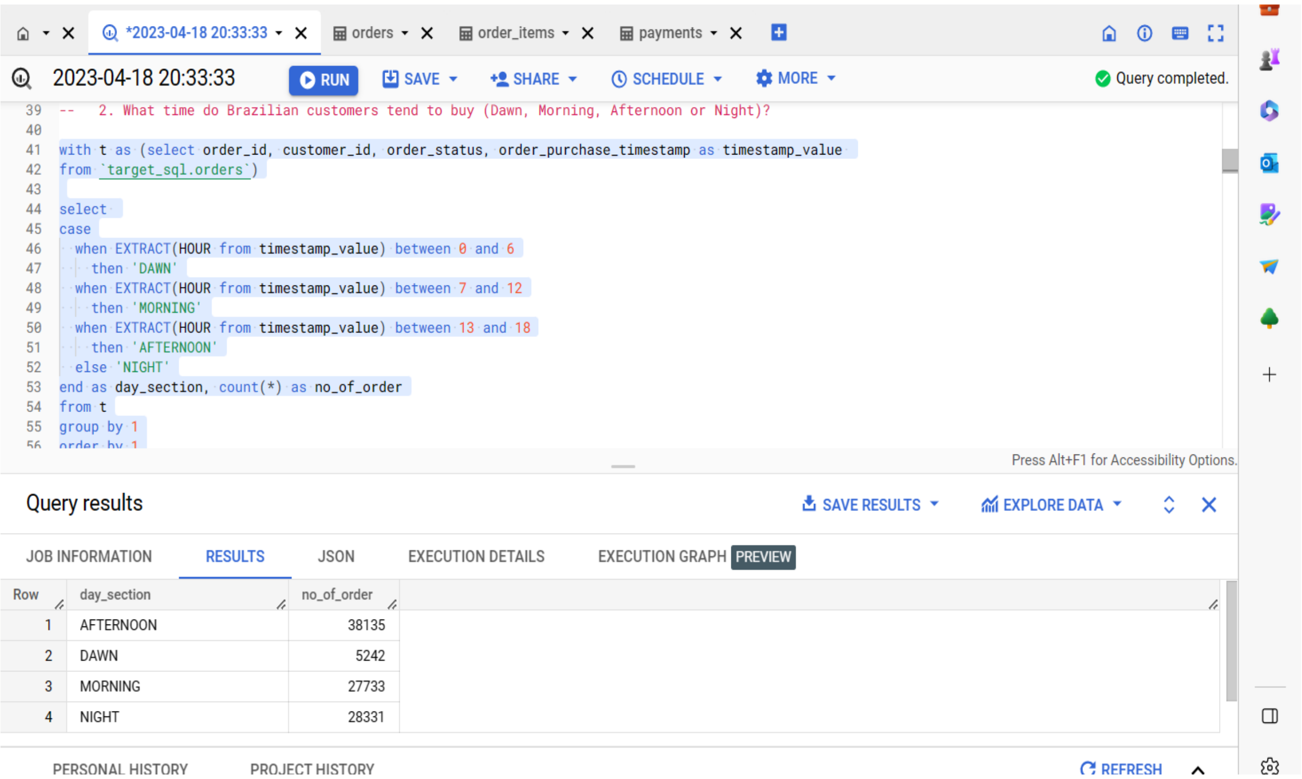


```
--   1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks
at specific months?

with t as (select order_id, customer_id, order_status, order_purchase_timestamp as timestamp_value
from `target_sql.orders`)
```

Row	month	no_of_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

‘Target-SQL’ - Project Queries

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

with t as (select order_id, customer_id, order_status, order_purchase_timestamp as timestamp_value
from `target_sql.orders`)

select
case
  when EXTRACT(HOUR from timestamp_value) between 0 and 6
    then 'DAWN'
  when EXTRACT(HOUR from timestamp_value) between 7 and 12
    then 'MORNING'
  when EXTRACT(HOUR from timestamp_value) between 13 and 18
    then 'AFTERNOON'
  else 'NIGHT'
end as day_section, count(*) as no_of_order
from t
group by 1
order by 1
```



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

with t as (select order_id, customer_id, order_status, order_purchase_timestamp as timestamp_value
from `target_sql.orders`)

select
case
  when EXTRACT(HOUR from timestamp_value) between 0 and 6
    then 'DAWN'
  when EXTRACT(HOUR from timestamp_value) between 7 and 12
    then 'MORNING'
  when EXTRACT(HOUR from timestamp_value) between 13 and 18
    then 'AFTERNOON'
  else 'NIGHT'
end as day_section, count(*) as no_of_order
from t
group by 1
order by 1
```

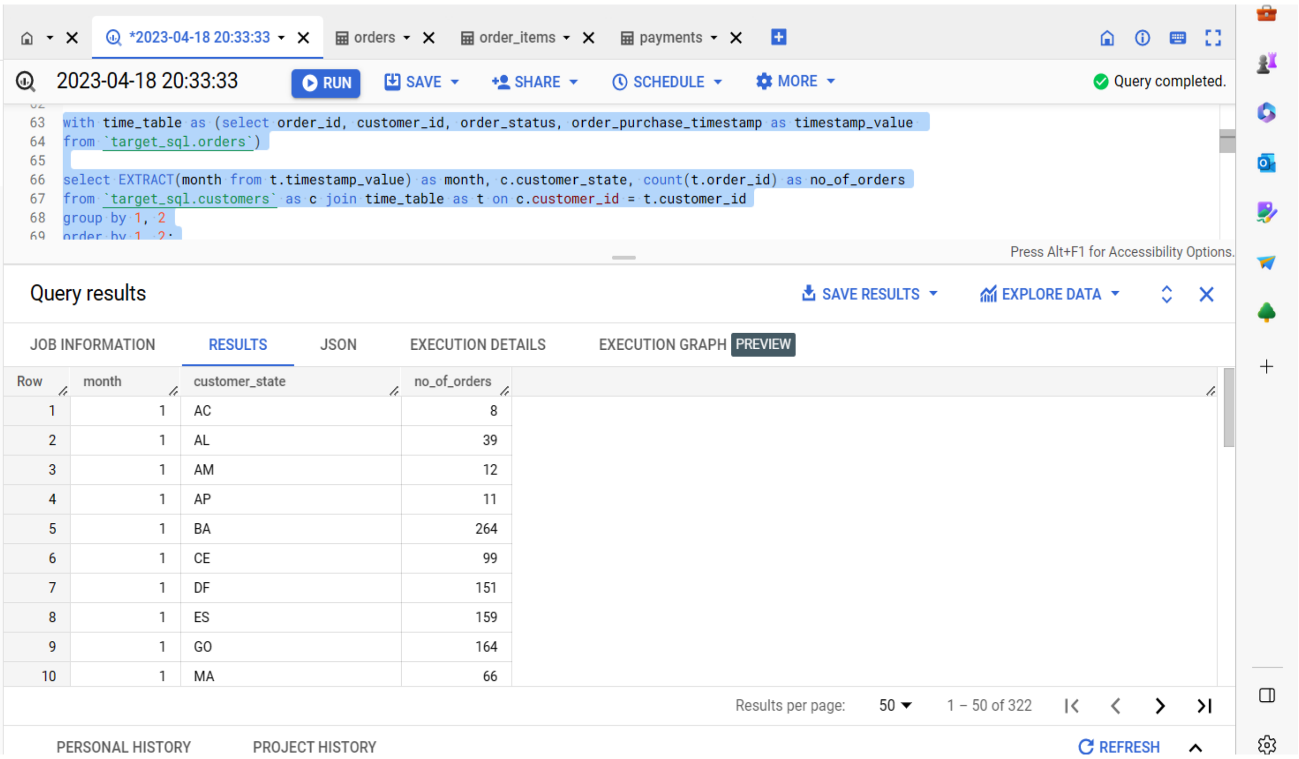
Row	day_section	no_of_order
1	AFTERNOON	38135
2	DAWN	5242
3	MORNING	27733
4	NIGHT	28331

‘Target-SQL’ - Project Queries

-- 3. Evolution of E-commerce orders in the Brazil region:

-- 1. Get month on month orders by states

```
with time_table as (select order_id, customer_id, order_status, order_purchase_timestamp
as timestamp_value
from `target_sql.orders`)

select EXTRACT(month from t.timestamp_value) as month, c.customer_state, count(t.order_id) as no_of_orders
from `target_sql.customers` as c join time_table as t on c.customer_id = t.customer_id
group by 1, 2
order by 1, 2;
```



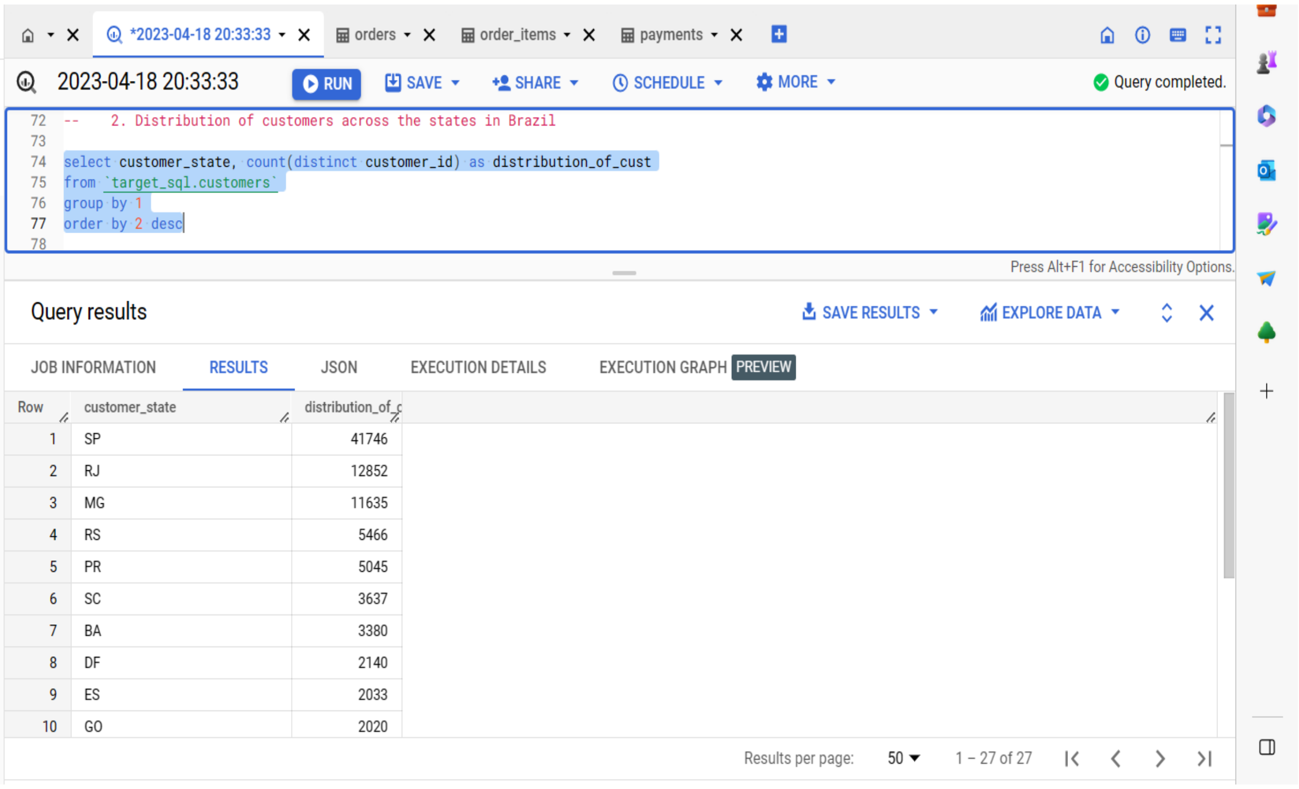
```
with time_table as (select order_id, customer_id, order_status, order_purchase_timestamp as timestamp_value
from `target_sql.orders`)

select EXTRACT(month from t.timestamp_value) as month, c.customer_state, count(t.order_id) as no_of_orders
from `target_sql.customers` as c join time_table as t on c.customer_id = t.customer_id
group by 1, 2
order by 1, 2;
```

Row	month	customer_state	no_of_orders
1	1	AC	8
2	1	AL	39
3	1	AM	12
4	1	AP	11
5	1	BA	264
6	1	CE	99
7	1	DF	151
8	1	ES	159
9	1	GO	164
10	1	MA	66

‘Target-SQL’ - Project Queries

--   2. Distribution of customers across the states in Brazil

```
select customer_state, count(distinct customer_id) as distribution_of_cust
from `target_sql.customers`
group by 1
order by 2 desc
```



```
--    2. Distribution of customers across the states in Brazil

select customer_state, count(distinct customer_id) as distribution_of_cust
from `target_sql.customers`
group by 1
order by 2 desc
```

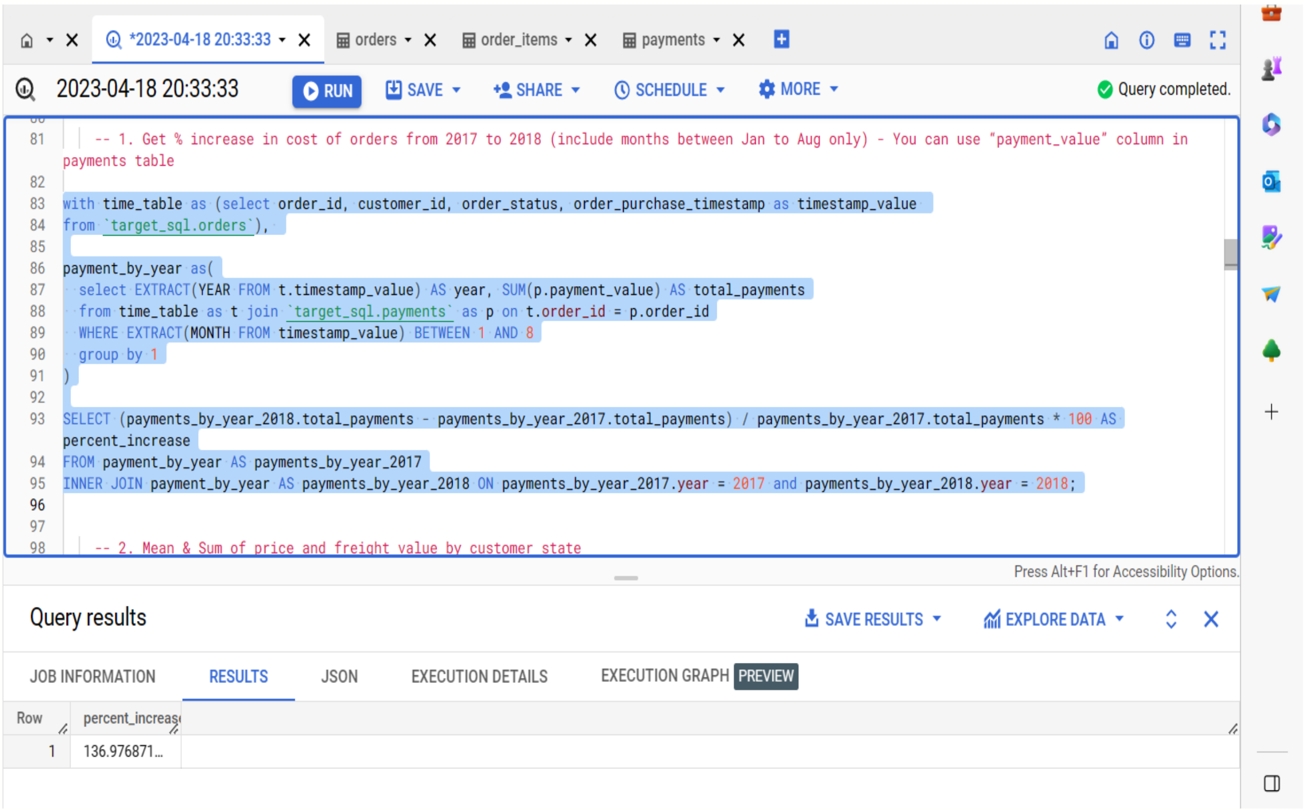
Row	customer_state	distribution_of_c
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

-- 4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

- 1. 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

‘Target-SQL’ - Project Queries

```
with time_table as (select order_id, customer_id, order_status, order_purchase_timestamp as timestamp_value
from `target_sql.orders`),

payment_by_year as(
  select EXTRACT(YEAR FROM t.timestamp_value) AS year, SUM(p.payment_value) AS total_payments
  from time_table as t join `target_sql.payments` as p on t.order_id = p.order_id
  WHERE EXTRACT(MONTH FROM timestamp_value) BETWEEN 1 AND 8
  group by 1
)

SELECT (payments_by_year_2018.total_payments - payments_by_year_2017.total_payments) / payments_by_year_2017.total_payments * 100 AS percent_increase
FROM payment_by_year AS payments_by_year_2017
INNER JOIN payment_by_year AS payments_by_year_2018 ON payments_by_year_2017.year = 2017 and payments_by_year_2018.year = 2018;
```

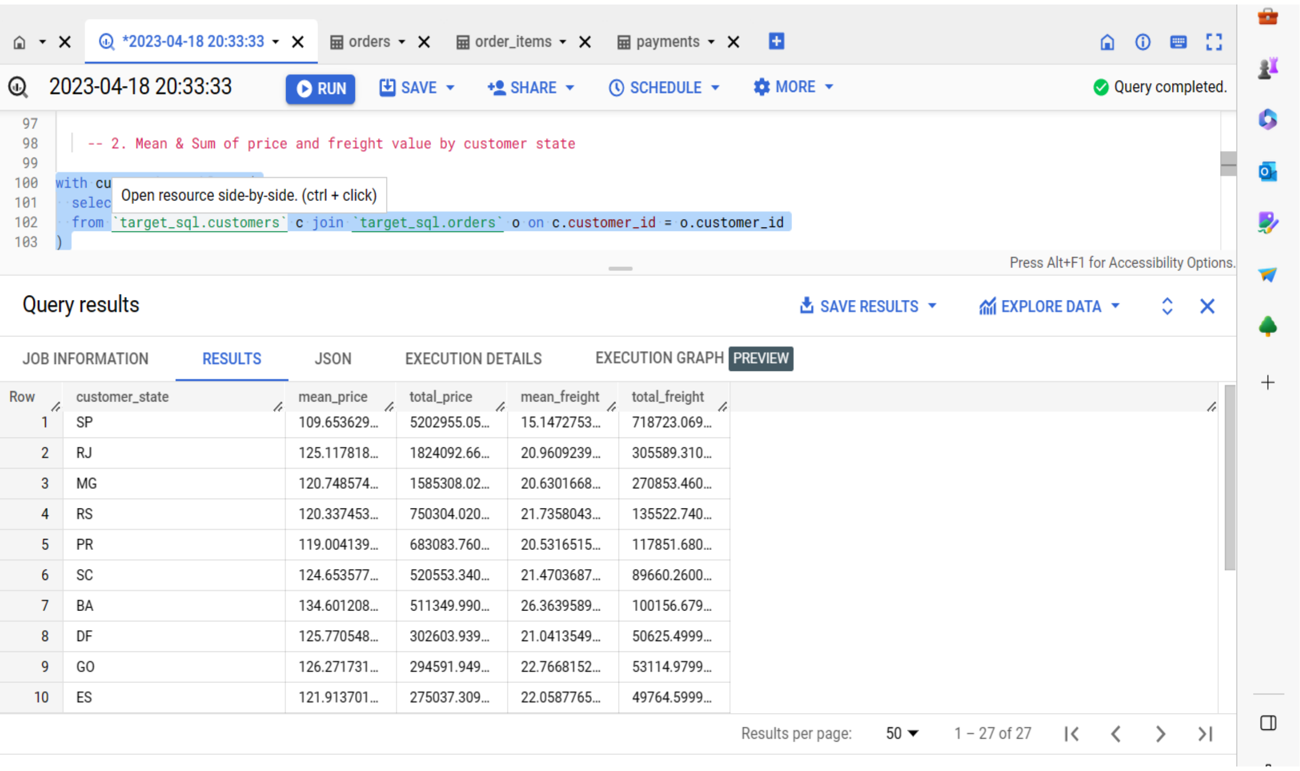


```
  -- 1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use "payment_value" column in payments table

with time_table as (select order_id, customer_id, order_status, order_purchase_timestamp as timestamp_value
from `target_sql.orders`),

payment_by_year as(
  select EXTRACT(YEAR FROM t.timestamp_value) AS year, SUM(p.payment_value) AS total_payments
  from time_table as t join `target_sql.payments` as p on t.order_id = p.order_id
  WHERE EXTRACT(MONTH FROM timestamp_value) BETWEEN 1 AND 8
  group by 1
)

SELECT (payments_by_year_2018.total_payments - payments_by_year_2017.total_payments) / payments_by_year_2017.total_payments * 100 AS percent_increase
FROM payment_by_year AS payments_by_year_2017
INNER JOIN payment_by_year AS payments_by_year_2018 ON payments_by_year_2017.year = 2017 and payments_by_year_2018.year = 2018;


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

Row	percent_increas
1	136.976871...

‘Target-SQL’ - Project Queries

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

with cust_order_table as(
  select c.customer_state, o.order_id
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT c_t.customer_state, AVG(o2.price) AS mean_price, SUM(o2.price) AS total_price,
 AVG(o2.freight_value) AS mean_freight, SUM(o2.freight_value) AS total_freight

FROM cust_order_table c_t
INNER JOIN `target_sql.order_items` o2 ON c_t.order_id = o2.order_id
GROUP BY c_t.customer_state
ORDER BY total_price DESC;
```



Query completed.

```

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

with cu
  selec
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)
```

Open resource side-by-side. (ctrl + click)

Press Alt+F1 for Accessibility Options.

Query results

Row	customer_state	mean_price	total_price	mean_freight	total_freight
1	SP	109.653629...	5202955.05...	15.1472753...	718723.069...
2	RJ	125.117818...	1824092.66...	20.9609239...	305589.310...
3	MG	120.748574...	1585308.02...	20.6301668...	270853.460...
4	RS	120.337453...	750304.020...	21.7358043...	135522.740...
5	PR	119.004139...	683083.760...	20.5316515...	117851.680...
6	SC	124.653577...	520553.340...	21.4703687...	89660.2600...
7	BA	134.601208...	511349.990...	26.3639589...	100156.679...
8	DF	125.770548...	302603.939...	21.0413549...	50625.4999...
9	GO	126.271731...	294591.949...	22.7668152...	53114.9799...
10	ES	121.913701...	275037.309...	22.0587765...	49764.5999...

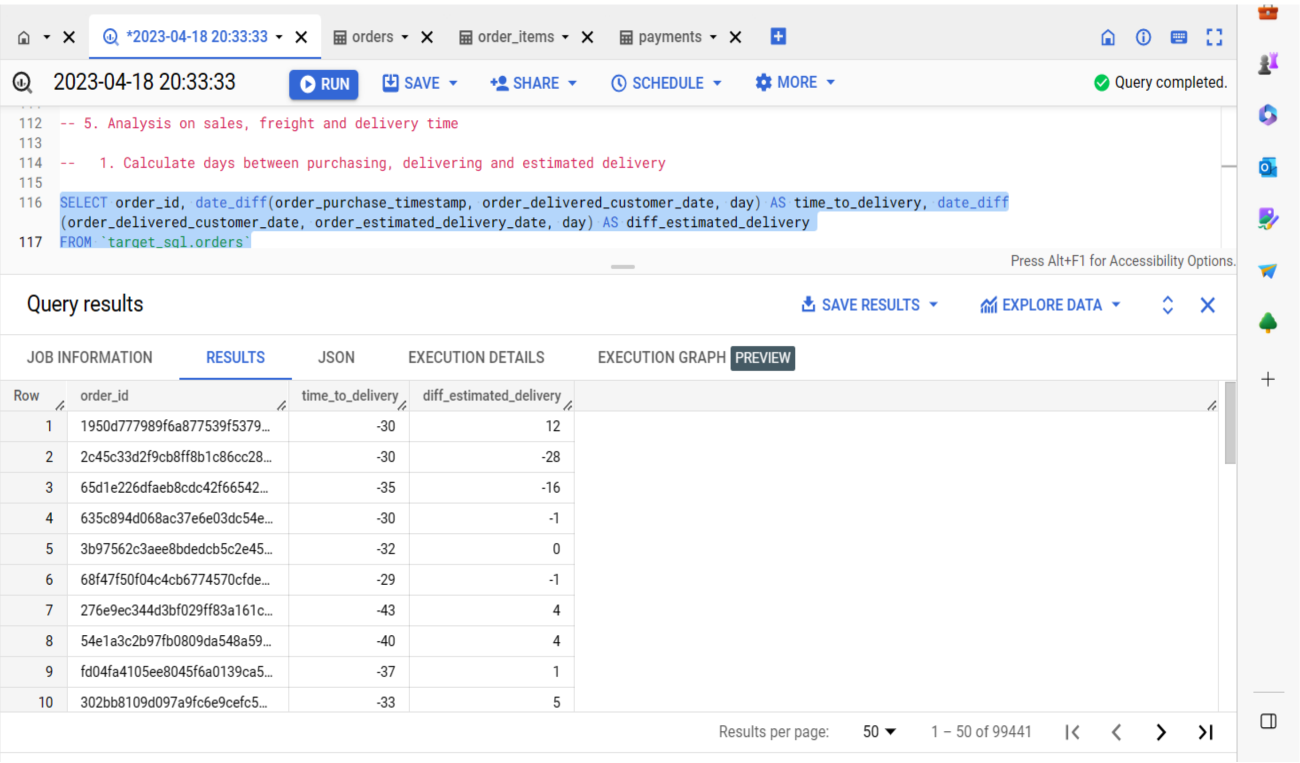
Results per page: 50 1 – 27 of 27

‘Target-SQL’ - Project Queries

-- 5. Analysis on sales, freight and delivery time

-- 1. Calculate days between purchasing, delivering and estimated delivery

```
SELECT order_id, date_diff(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to_delivery, date_diff(order_delivered_customer_date, order_estimated_delivery_date, day) AS diff_estimated_delivery
FROM `target_sql.orders`
```



```
-- 5. Analysis on sales, freight and delivery time

--   1. Calculate days between purchasing, delivering and estimated delivery

SELECT order_id, date_diff(order_purchase_timestamp, order_delivered_customer_date, day) AS time_to_delivery, date_diff
(order_delivered_customer_date, order_estimated_delivery_date, day) AS diff_estimated_delivery
FROM `target_sql.orders`
```

Query results

Row	order_id	time_to_delivery	diff_estimated_delivery
1	1950d777989f6a877539f5379...	-30	12
2	2c45c33d2f9cb8ff8b1c86cc28...	-30	-28
3	65d1e226dfaeb8cdc42f66542...	-35	-16
4	635c894d068ac37e6e03dc54e...	-30	-1
5	3b97562c3aee8bdedcb5c2e45...	-32	0
6	68f47f50f04c4cb6774570cfde...	-29	-1
7	276e9ec344d3bf029ff83a161c...	-43	4
8	54e1a3c2b97fb0809da548a59...	-40	4
9	fd04fa4105ee8045f6a0139ca5...	-37	1
10	302bb8109d097a9fc6e9cefc5...	-33	5

Results per page: 50 1 – 50 of 99441

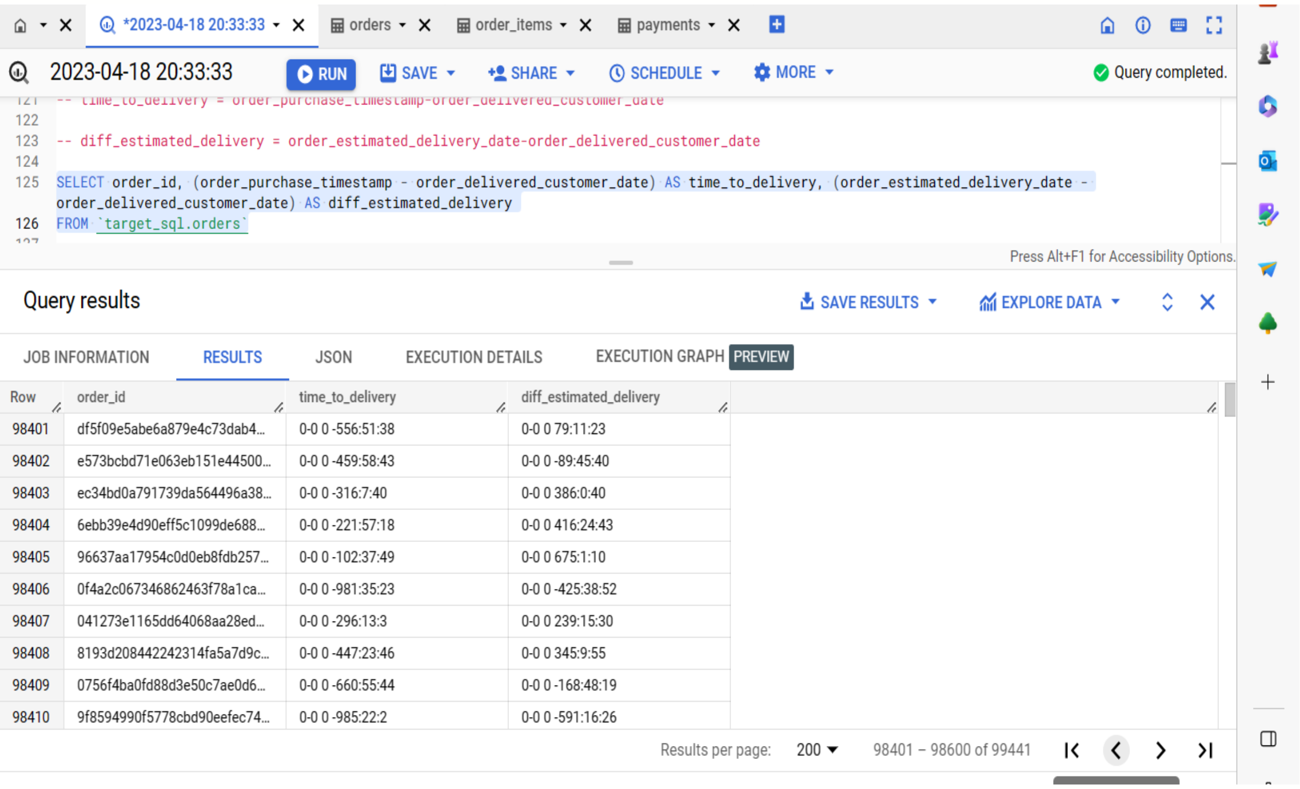
‘Target-SQL’ - Project Queries

--   2. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

-- time_to_delivery = order_purchase_timestamp-order_delivered_customer_date

-- diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

```
SELECT order_id, (order_purchase_timestamp - order_delivered_customer_date) AS time_to_delivery, (order_estimated_delivery_date - order_delivered_customer_date) AS diff_estimated_delivery
FROM `target_sql.orders`
```



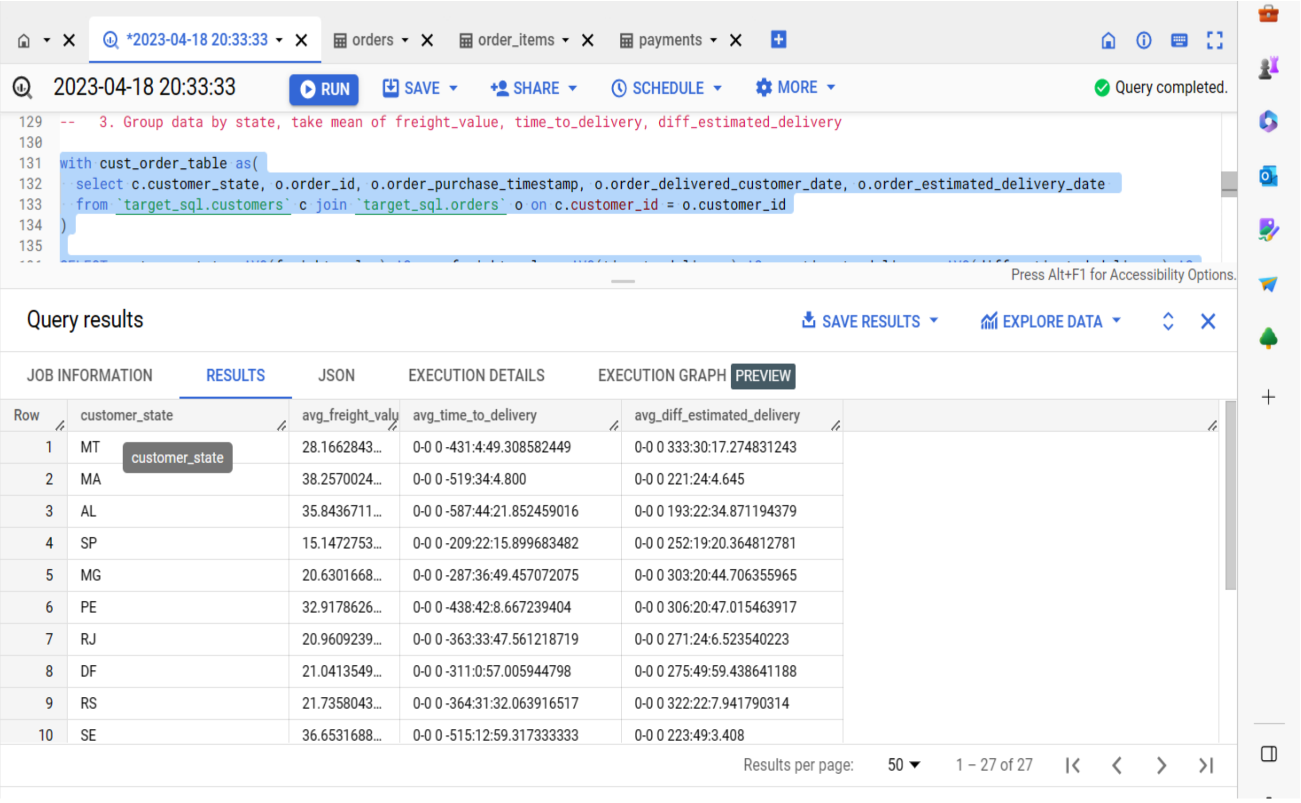
```
-- time_to_delivery = order_purchase_timestamp-order_delivered_customer_date

-- diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

SELECT order_id, (order_purchase_timestamp - order_delivered_customer_date) AS time_to_delivery, (order_estimated_delivery_date -
order_delivered_customer_date) AS diff_estimated_delivery
FROM `target_sql.orders`
```

Row	order_id	time_to_delivery	diff_estimated_delivery
98401	df5f09e5abe6a879e4c73dab4...	0-0 0 -556:51:38	0-0 0 79:11:23
98402	e573bcbd71e063eb151e44500...	0-0 0 -459:58:43	0-0 0 -89:45:40
98403	ec34bd0a791739da564496a38...	0-0 0 -316:7:40	0-0 0 386:0:40
98404	6ebb39e4d90eff5c1099de688...	0-0 0 -221:57:18	0-0 0 416:24:43
98405	96637aa17954c0d0eb8fdb257...	0-0 0 -102:37:49	0-0 0 675:1:10
98406	0f4a2c067346862463f78a1ca...	0-0 0 -981:35:23	0-0 0 -425:38:52
98407	041273e1165dd64068aa28ed...	0-0 0 -296:13:3	0-0 0 239:15:30
98408	8193d208442242314fa5a7d9c...	0-0 0 -447:23:46	0-0 0 345:9:55
98409	0756f4ba0fd88d3e50c7ae0d6...	0-0 0 -660:55:44	0-0 0 -168:48:19
98410	9f8594990f5778cbd90eefec74...	0-0 0 -985:22:2	0-0 0 -591:16:26

Results per page: 200 98401 – 98600 of 99441

‘Target-SQL’ - Project Queries

```
        - 3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated delivery

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT customer_state, AVG(freight_value) AS avg_freight_value, AVG(time_to_delivery)
 AS avg_time_to_delivery, AVG(diff_estimated_delivery) AS avg_diff_estimated_delivery
FROM (
    SELECT c_t.customer_state, o1.freight_value, c_t.order_purchase_timestamp - c_t.order_delivered_customer_date AS time_to_delivery, c_t.order_estimated_delivery_date - c_t.order_delivered_customer_date AS diff_estimated_delivery
    FROM cust_order_table c_t
    INNER JOIN `target_sql.order_items` o1 ON c_t.order_id = o1.order_id
) AS order_data
GROUP BY customer_state
```

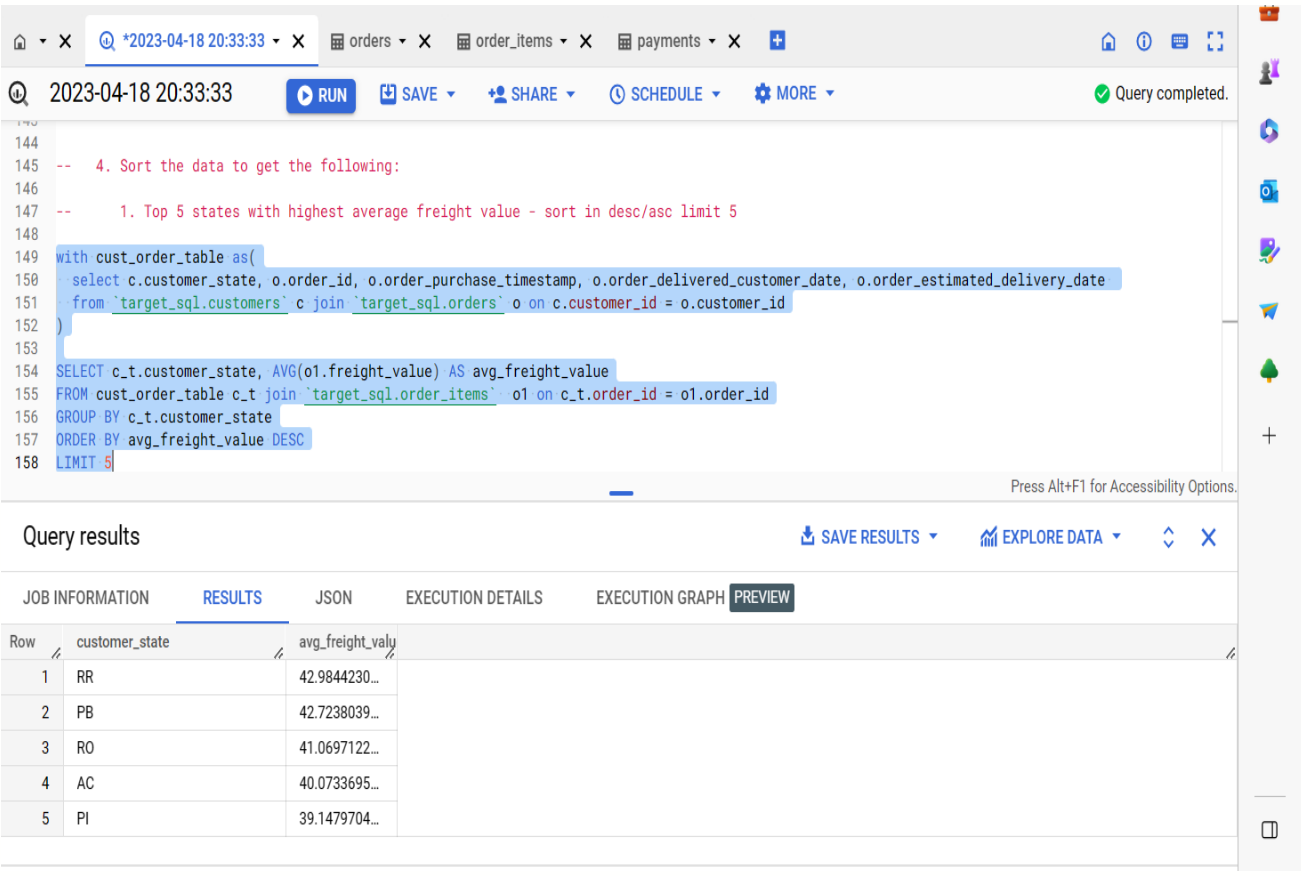


```
--   3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)
```

Row	customer_state	avg_freight_valu	avg_time_to_delivery	avg_diff_estimated_delivery
1	MT	28.1662843...	0-0 0 -431:4:49.308582449	0-0 0 333:30:17.274831243
2	MA	38.2570024...	0-0 0 -519:34:4.800	0-0 0 221:24:4.645
3	AL	35.8436711...	0-0 0 -587:44:21.852459016	0-0 0 193:22:34.871194379
4	SP	15.1472753...	0-0 0 -209:22:15.899683482	0-0 0 252:19:20.364812781
5	MG	20.6301668...	0-0 0 -287:36:49.457072075	0-0 0 303:20:44.706355965
6	PE	32.9178626...	0-0 0 -438:42:8.667239404	0-0 0 306:20:47.015463917
7	RJ	20.9609239...	0-0 0 -363:33:47.561218719	0-0 0 271:24:6.523540223
8	DF	21.0413549...	0-0 0 -311:0:57.005944798	0-0 0 275:49:59.438641188
9	RS	21.7358043...	0-0 0 -364:31:32.063916517	0-0 0 322:22:7.941790314
10	SE	36.6531688...	0-0 0 -515:12:59.317333333	0-0 0 223:49:3.408

‘Target-SQL’ - Project Queries

```
--   4. Sort the data to get the following:

--      1. Top 5 states with highest average freight value - sort in desc/asc limit 5

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT c_t.customer_state, AVG(o1.freight_value) AS avg_freight_value
FROM cust_order_table c_t join `target_sql.order_items`  o1 on c_t.order_id = o1.order_id
GROUP BY c_t.customer_state
ORDER BY avg_freight_value DESC
LIMIT 5
```

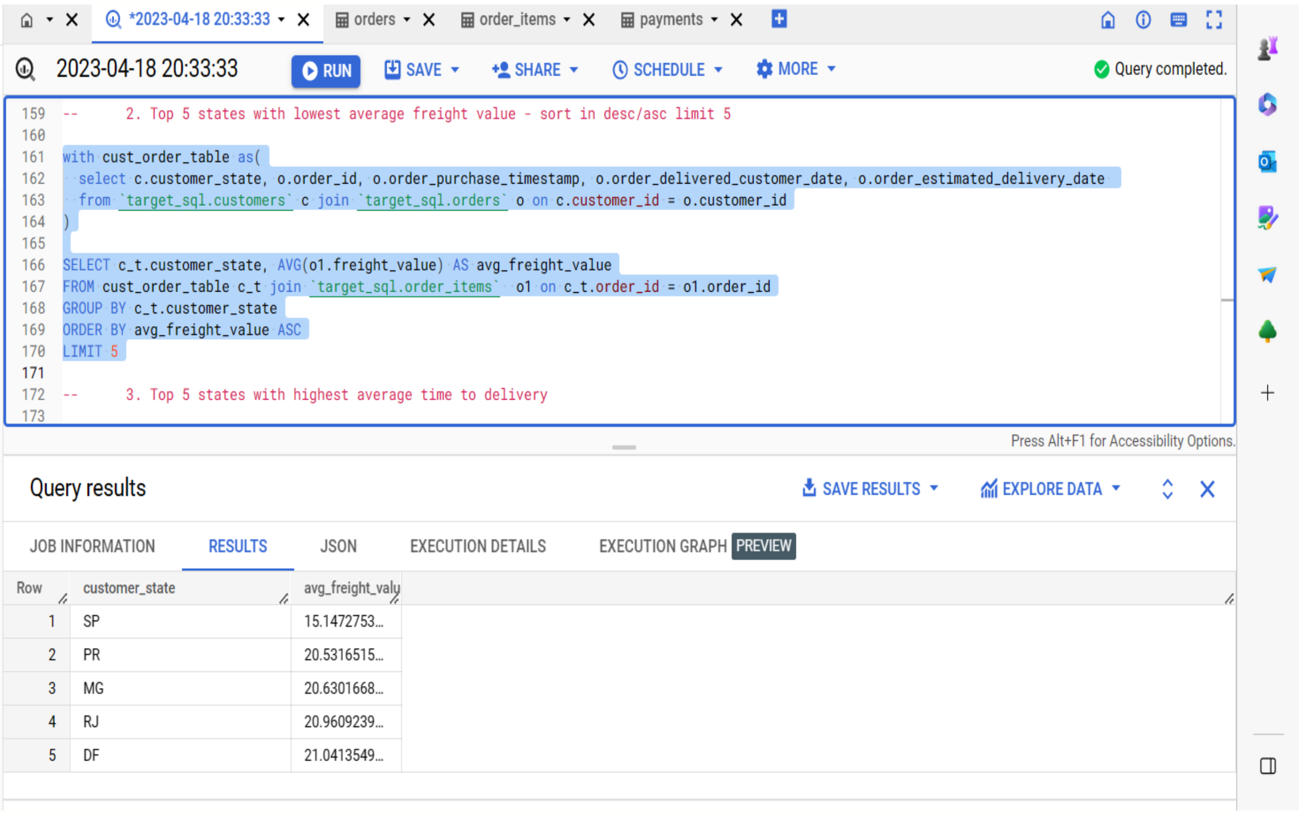


```
--    4. Sort the data to get the following:

--       1. Top 5 states with highest average freight value - sort in desc/asc limit 5

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT c_t.customer_state, AVG(o1.freight_value) AS avg_freight_value
FROM cust_order_table c_t join `target_sql.order_items` o1 on c_t.order_id = o1.order_id
GROUP BY c_t.customer_state
ORDER BY avg_freight_value DESC
LIMIT 5
```

Row	customer_state	avg_freight_valu
1	RR	42.9844230…
2	PB	42.7238039…
3	RO	41.0697122…
4	AC	40.0733695…
5	PI	39.1479704…

‘Target-SQL’ - Project Queries

```
--      2. Top 5 states with lowest average freight value - sort in desc/asc limit 5

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT c_t.customer_state, AVG(o1.freight_value) AS avg_freight_value
FROM cust_order_table c_t join `target_sql.order_items`  o1 on c_t.order_id = o1.order_id
GROUP BY c_t.customer_state
ORDER BY avg_freight_value ASC
LIMIT 5
```



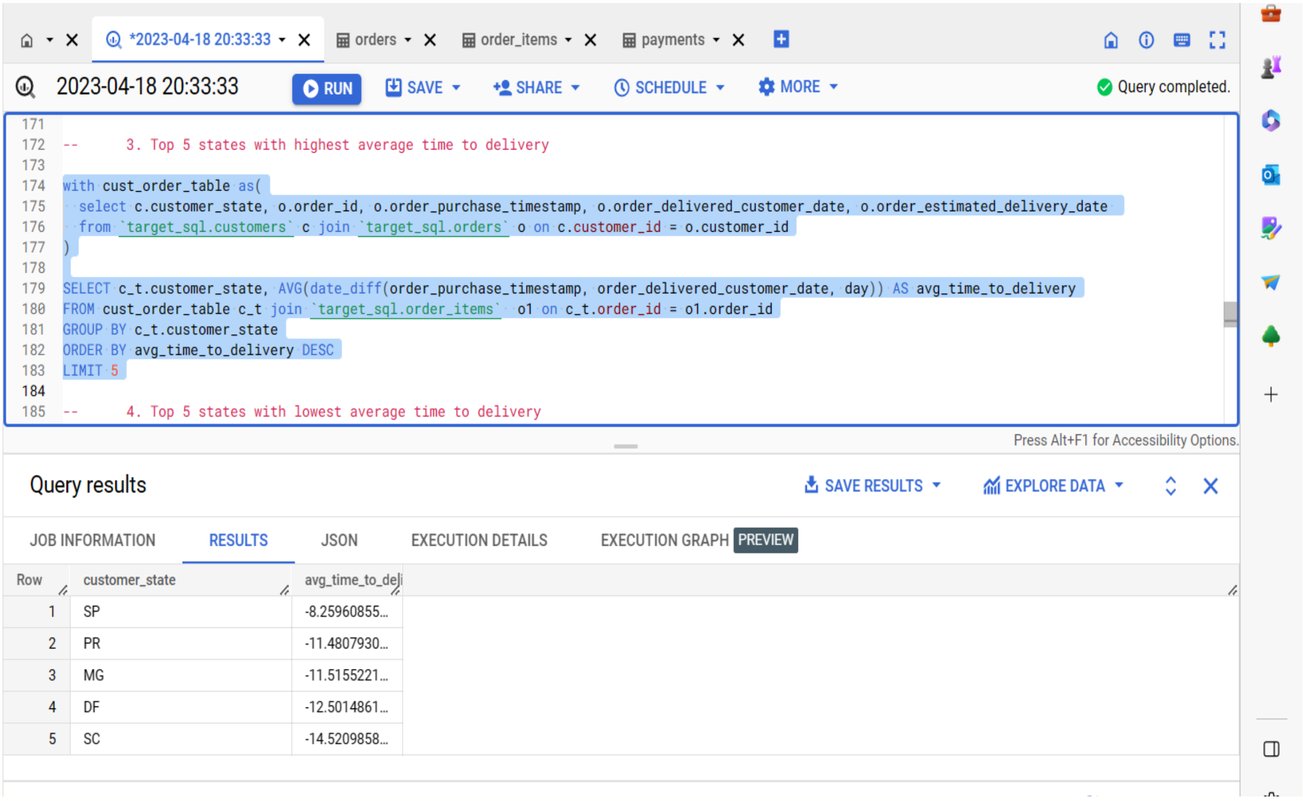
```
--      2. Top 5 states with lowest average freight value - sort in desc/asc limit 5

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT c_t.customer_state, AVG(o1.freight_value) AS avg_freight_value
FROM cust_order_table c_t join `target_sql.order_items`  o1 on c_t.order_id = o1.order_id
GROUP BY c_t.customer_state
ORDER BY avg_freight_value ASC
LIMIT 5

--      3. Top 5 states with highest average time to delivery
```

Query results

Row	customer_state	avg_freight_valu
1	SP	15.1472753...
2	PR	20.5316515...
3	MG	20.6301668...
4	RJ	20.9609239...
5	DF	21.0413549...

‘Target-SQL’ - Project Queries

```
--      3. Top 5 states with highest average time to delivery

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT c_t.customer_state, AVG(date_diff(order_purchase_timestamp, order_delivered_customer_date, day)) AS avg_time_to_delivery
FROM cust_order_table c_t join `target_sql.order_items`  o1 on c_t.order_id = o1.order_id
GROUP BY c_t.customer_state
ORDER BY avg_time_to_delivery DESC
LIMIT 5
```

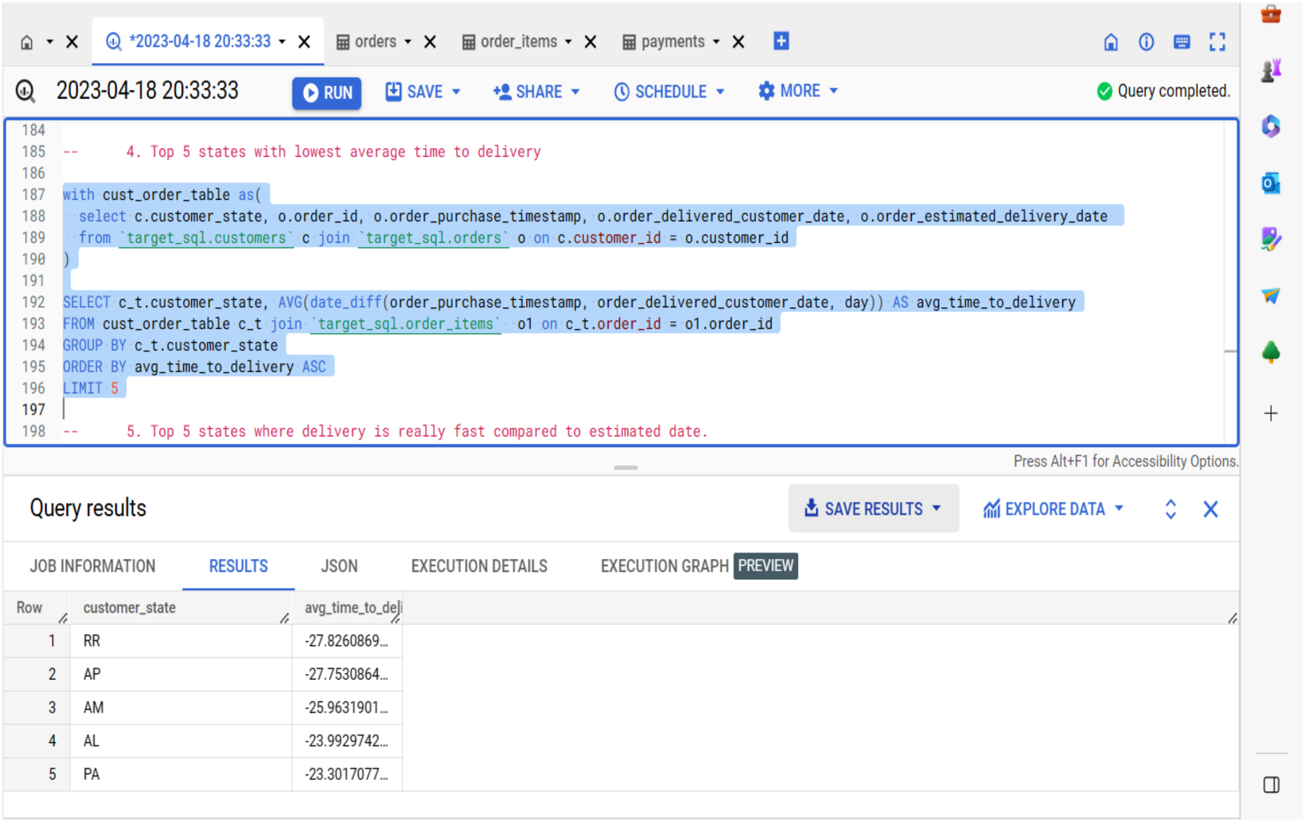


```
--      3. Top 5 states with highest average time to delivery

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT c_t.customer_state, AVG(date_diff(order_purchase_timestamp, order_delivered_customer_date, day)) AS avg_time_to_delivery
FROM cust_order_table c_t join `target_sql.order_items`  o1 on c_t.order_id = o1.order_id
GROUP BY c_t.customer_state
ORDER BY avg_time_to_delivery DESC
LIMIT 5

--      4. Top 5 states with lowest average time to delivery
```

Row	customer_state	avg_time_to_deli
1	SP	-8.25960855...
2	PR	-11.4807930...
3	MG	-11.5155221...
4	DF	-12.5014861...
5	SC	-14.5209858...

‘Target-SQL’ - Project Queries

```
--      4. Top 5 states with lowest average time to delivery

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT c_t.customer_state, AVG(date_diff(order_purchase_timestamp, order_delivered_customer_date, day)) AS avg_time_to_delivery
FROM cust_order_table c_t join `target_sql.order_items`  o1 on c_t.order_id = o1.order_id
GROUP BY c_t.customer_state
ORDER BY avg_time_to_delivery ASC
LIMIT 5
```

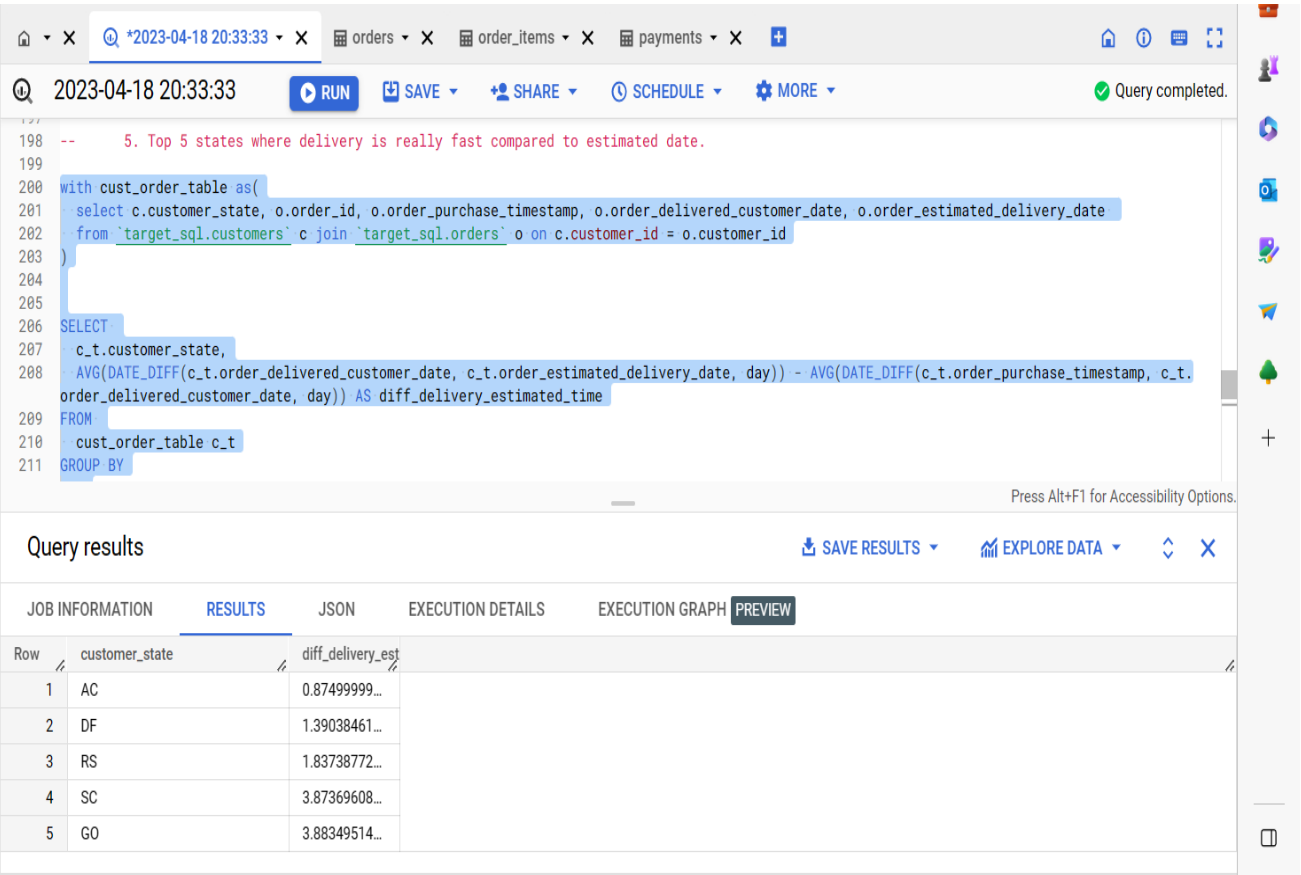


```
--      4. Top 5 states with lowest average time to delivery

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)

SELECT c_t.customer_state, AVG(date_diff(order_purchase_timestamp, order_delivered_customer_date, day)) AS avg_time_to_delivery
FROM cust_order_table c_t join `target_sql.order_items`  o1 on c_t.order_id = o1.order_id
GROUP BY c_t.customer_state
ORDER BY avg_time_to_delivery ASC
LIMIT 5

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

Row	customer_state	avg_time_to_deli
1	RR	-27.8260869...
2	AP	-27.7530864...
3	AM	-25.9631901...
4	AL	-23.9929742...
5	PA	-23.3017077...

‘Target-SQL’ - Project Queries

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

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_cust
omer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)


SELECT
  c_t.customer_state,
  AVG(DATE_DIFF(c_t.order_delivered_customer_date, c_t.order_estimated_delivery_date
, day)) - AVG(DATE_DIFF(c_t.order_purchase_timestamp, c_t.order_delivered_customer_
date, day)) AS diff_delivery_estimated_time
FROM
  cust_order_table c_t
GROUP BY
  1
HAVING
  AVG(DATE_DIFF(c_t.order_delivered_customer_date, c_t.order_estimated_delivery_date
, day)) > AVG(DATE_DIFF(c_t.order_purchase_timestamp, c_t.order_delivered_customer_
date, day))
ORDER BY
  2 ASC
LIMIT 5
```



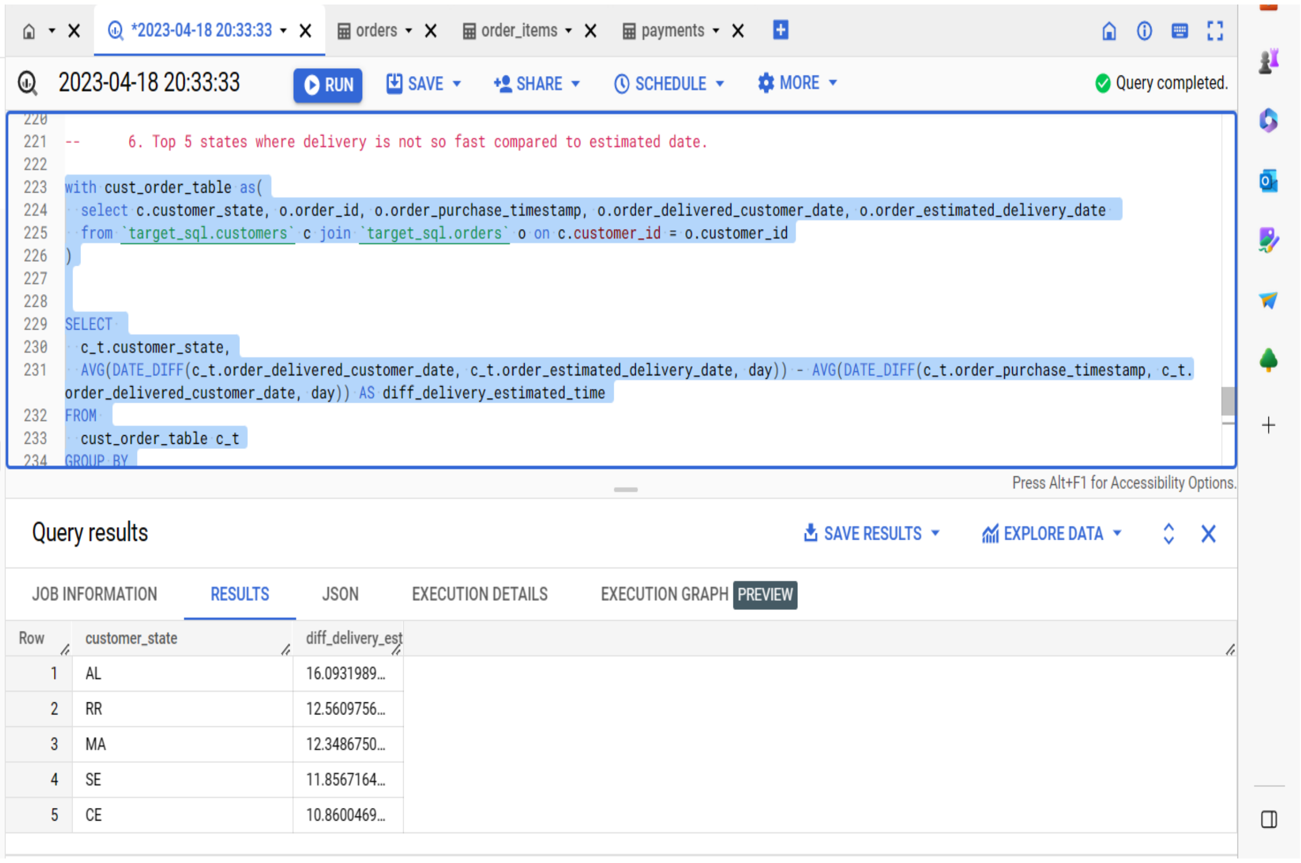
```
--     5. Top 5 states where delivery is really fast compared to estimated date.

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)


SELECT
  c_t.customer_state,
  AVG(DATE_DIFF(c_t.order_delivered_customer_date, c_t.order_estimated_delivery_date, day)) - AVG(DATE_DIFF(c_t.order_purchase_timestamp, c_t.
order_delivered_customer_date, day)) AS diff_delivery_estimated_time
FROM
  cust_order_table c_t
GROUP BY
```

Query results

Row	customer_state	diff_delivery_est
1	AC	0.87499999...
2	DF	1.39038461...
3	RS	1.83738772...
4	SC	3.87369608...
5	GO	3.88349514...

‘Target-SQL’ - Project Queries

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


with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)


SELECT
  c_t.customer_state,
  AVG(DATE_DIFF(c_t.order_delivered_customer_date, c_t.order_estimated_delivery_date, day)) - AVG(DATE_DIFF(c_t.order_purchase_timestamp, c_t.order_delivered_customer_date, day)) AS diff_delivery_estimated_time
FROM
  cust_order_table c_t
GROUP BY
  1
HAVING
  AVG(DATE_DIFF(c_t.order_delivered_customer_date, c_t.order_estimated_delivery_date, day)) > AVG(DATE_DIFF(c_t.order_purchase_timestamp, c_t.order_delivered_customer_date, day))
ORDER BY
  2 DESC
LIMIT 5
```

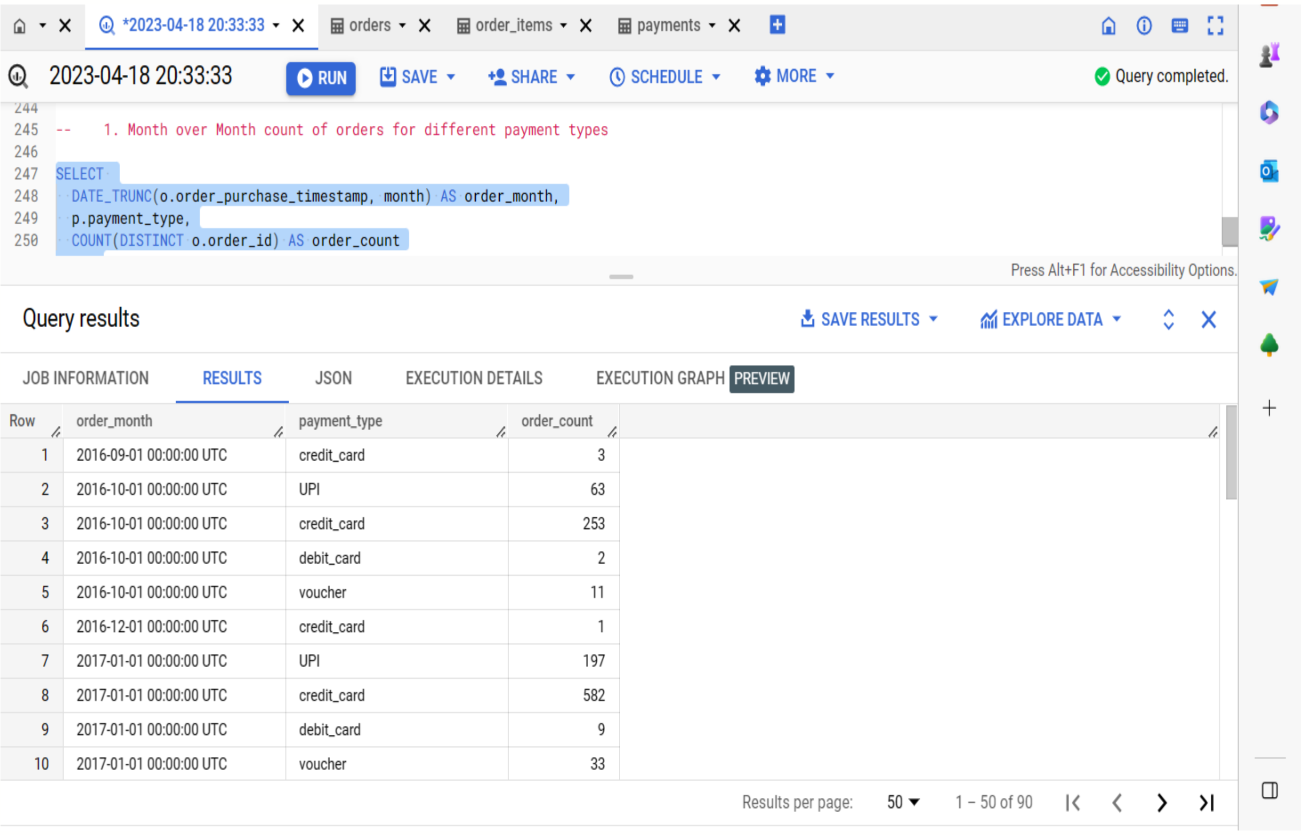


```
--      6. Top 5 states where delivery is not so fast compared to estimated date.

with cust_order_table as(
  select c.customer_state, o.order_id, o.order_purchase_timestamp, o.order_delivered_customer_date, o.order_estimated_delivery_date
  from `target_sql.customers` c join `target_sql.orders` o on c.customer_id = o.customer_id
)


SELECT
  c_t.customer_state,
  AVG(DATE_DIFF(c_t.order_delivered_customer_date, c_t.order_estimated_delivery_date, day)) - AVG(DATE_DIFF(c_t.order_purchase_timestamp, c_t.order_delivered_customer_date, day)) AS diff_delivery_estimated_time
FROM
  cust_order_table c_t
GROUP BY
```

Row	customer_state	diff_delivery_est
1	AL	16.0931989...
2	RR	12.5609756...
3	MA	12.3486750...
4	SE	11.8567164...
5	CE	10.8600469...

‘Target-SQL’ - Project Queries

```
-- 6. Payment type analysis:

--    1. Month over Month count of orders for different payment types

SELECT
  DATE_TRUNC(o.order_purchase_timestamp, month) AS order_month,
  p.payment_type,
  COUNT(DISTINCT o.order_id) AS order_count
FROM
  `target_sql.orders` o
  JOIN `target_sql.payments` p ON o.order_id = p.order_id
GROUP BY
  1, 2
ORDER BY
  1 ASC, 2 ASC
```

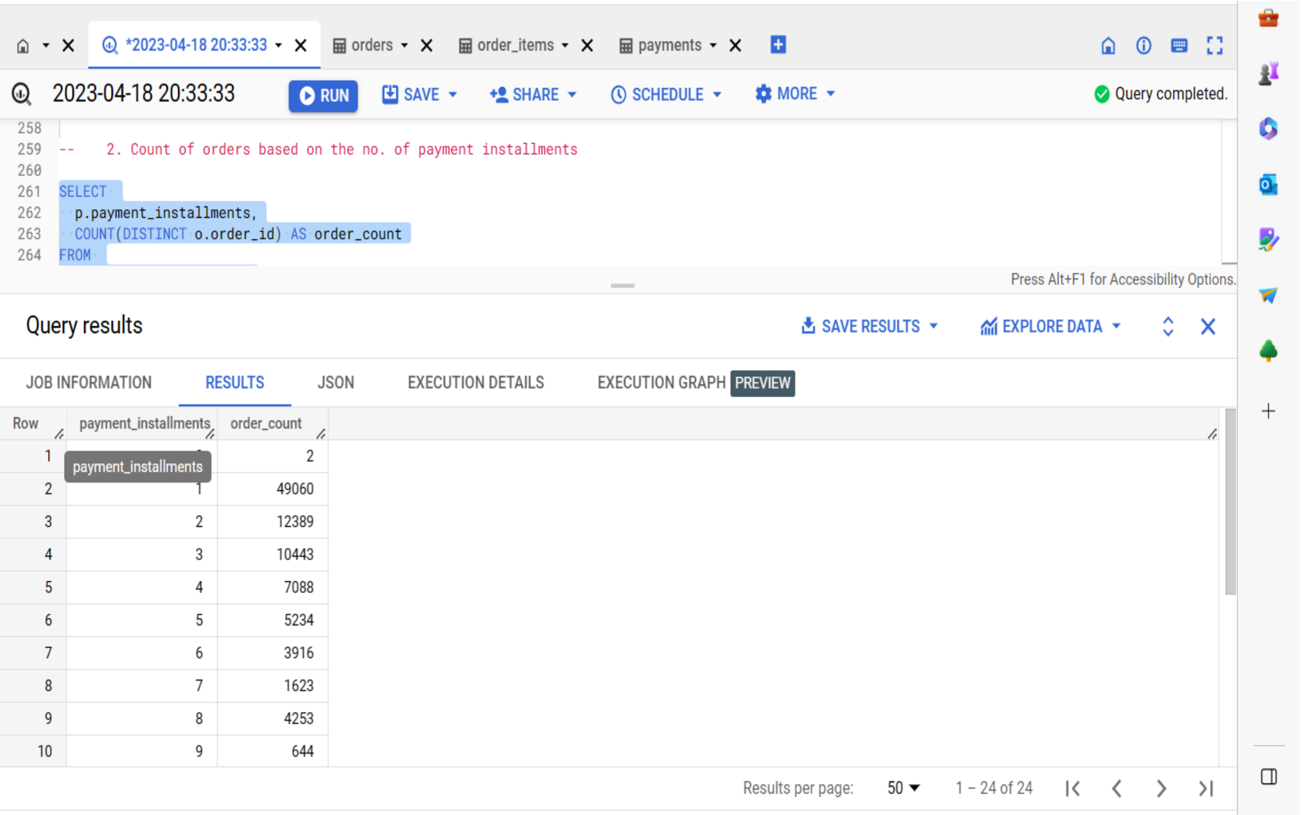


```
--    1. Month over Month count of orders for different payment types

SELECT
  DATE_TRUNC(o.order_purchase_timestamp, month) AS order_month,
  p.payment_type,
  COUNT(DISTINCT o.order_id) AS order_count
```

Row	order_month	payment_type	order_count
1	2016-09-01 00:00:00 UTC	credit_card	3
2	2016-10-01 00:00:00 UTC	UPI	63
3	2016-10-01 00:00:00 UTC	credit_card	253
4	2016-10-01 00:00:00 UTC	debit_card	2
5	2016-10-01 00:00:00 UTC	voucher	11
6	2016-12-01 00:00:00 UTC	credit_card	1
7	2017-01-01 00:00:00 UTC	UPI	197
8	2017-01-01 00:00:00 UTC	credit_card	582
9	2017-01-01 00:00:00 UTC	debit_card	9
10	2017-01-01 00:00:00 UTC	voucher	33

‘Target-SQL’ - Project Queries

```
--    2. Count of orders based on the no. of payment installments

SELECT
  p.payment_installments,
  COUNT(DISTINCT o.order_id) AS order_count
FROM
  `target_sql.orders` o
  JOIN `target_sql.payments` p ON o.order_id = p.order_id
GROUP BY
  1
ORDER BY
  1 ASC
```



```
--    2. Count of orders based on the no. of payment installments

SELECT
  p.payment_installments,
  COUNT(DISTINCT o.order_id) AS order_count
FROM
```

Row	payment_installments	order_count
1	0	2
2	1	49060
3	2	12389
4	3	10443
5	4	7088
6	5	5234
7	6	3916
8	7	1623
9	8	4253
10	9	644

‘Target-SQL’ - Project Queries

ACTIONABLE INSIGHTS

1. From the data we observe that there is increase in the number of orders placed in the month starting from May to August. This growth in trend starts from March and reaches its peak in August and then it starts decreasing and increase a bit in November then saturates in December.

2. Customers timing of purchasing is maximum in the Afternoon period a bit less in Morning and Night time and lowest during Dawn.

3. Using customers table we find that state SP, RJ, MG are the top three states having maximum number of customers and SP being the topmost over 40K customers whereas AC, AP and RR are the bottom 3 states having least number of customers.

4. Company had an almost 137% more growth in year 2018 compared to 2017 i.e. there is almost 137% increase in sales in year 2018.

5. States that are having high mean price value also have high mean freight value.

6. By analysing the orders table we observe that delivery time is inconsistent as well as difference in estimated delivery, some of the times delivery is too late by the company.

7. The states with low number of customers tend to have high freight value as compared to the states with large number of customers.

8. **Majority of the customers use their credit cards as their payment method.**

9. **Payment instalments ranging between 1 to 10 have maximum number of customers whereas larger the installment range less there is chance to attract the customer also there are only 2 person with zero instalments which indicates people are attracted to the installment scheme.**

RECOMMENDATIONS

1. **Since purchases are high between between March and August which also includes the Carnival festival of Brazil in Feb and Mar, we can provide various discounts on popular products, we can do buy on get 1 offer or we can organise some kind of competition including various prizes during this period of the year to increase the sales. And these offers can also vary during various periods of the day when customer purchase are high.**

2. **We observe that mean price of a state is somehow related to mean freight value we can improve our sales by regulating the freight value and analysing waste of logistic and transportation resources.**

3. **States with less number of customers can increase the margin of their estimated delivery time as these state have high mean price values compared to the states with large customer size, by regulating the delivery time these states can decrease their freight cost value which can also improve the mean price value.**

‘Target-SQL’ - Project Queries

4. **By estimating the delivery time precisely we can increase our sales and lessen our losses in freight value.**

5. **We can give some special discount to the people using credit cards as their payment method this can encourage the use of credit cards and may help in increasing the sales.**

6. **We should encourage the payment using installments we can also some discounts depending on their product and duration of their installments, lesser the duration more can be the discount (max 20%) and vice versa.**