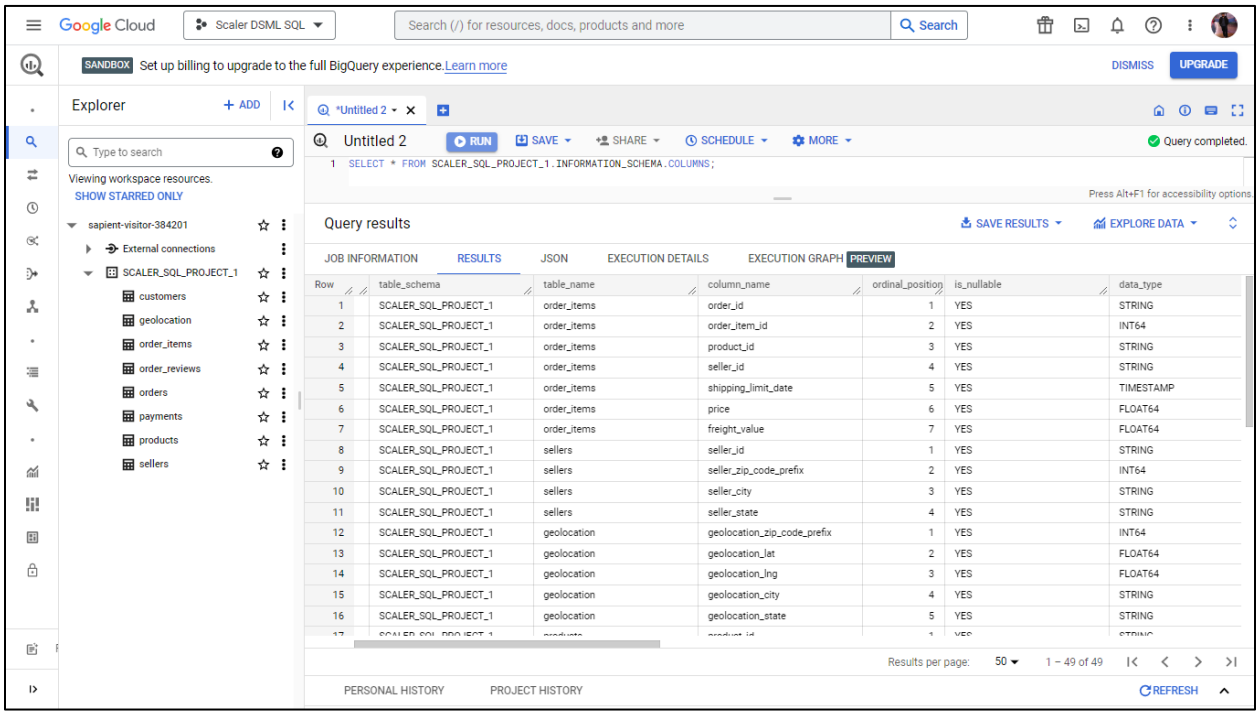
What does 'good' look like?

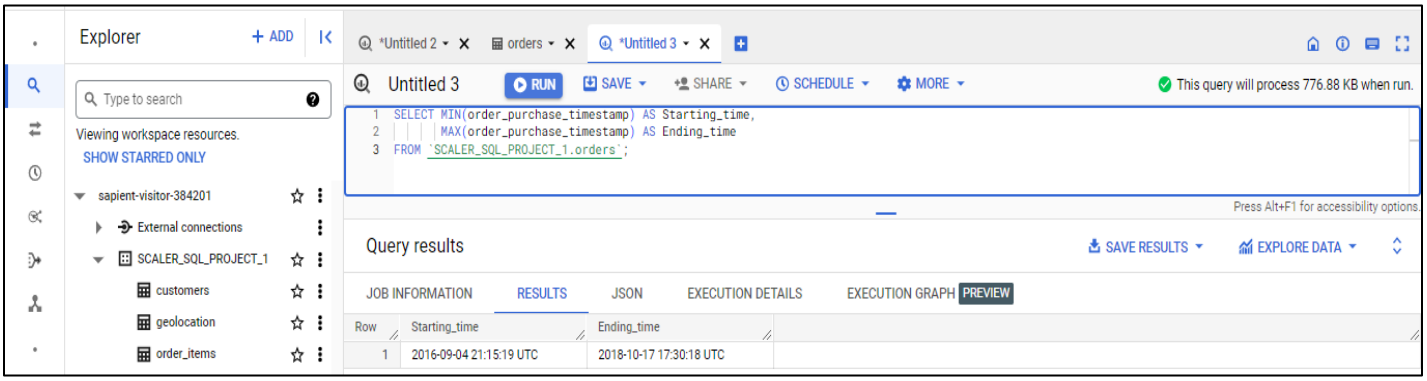
1. **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:**
1. **Data type of columns in a table**

```
SELECT * FROM SCALER_SQL_PROJECT_1.INFORMATION_SCHEMA.COLUMNS;
```



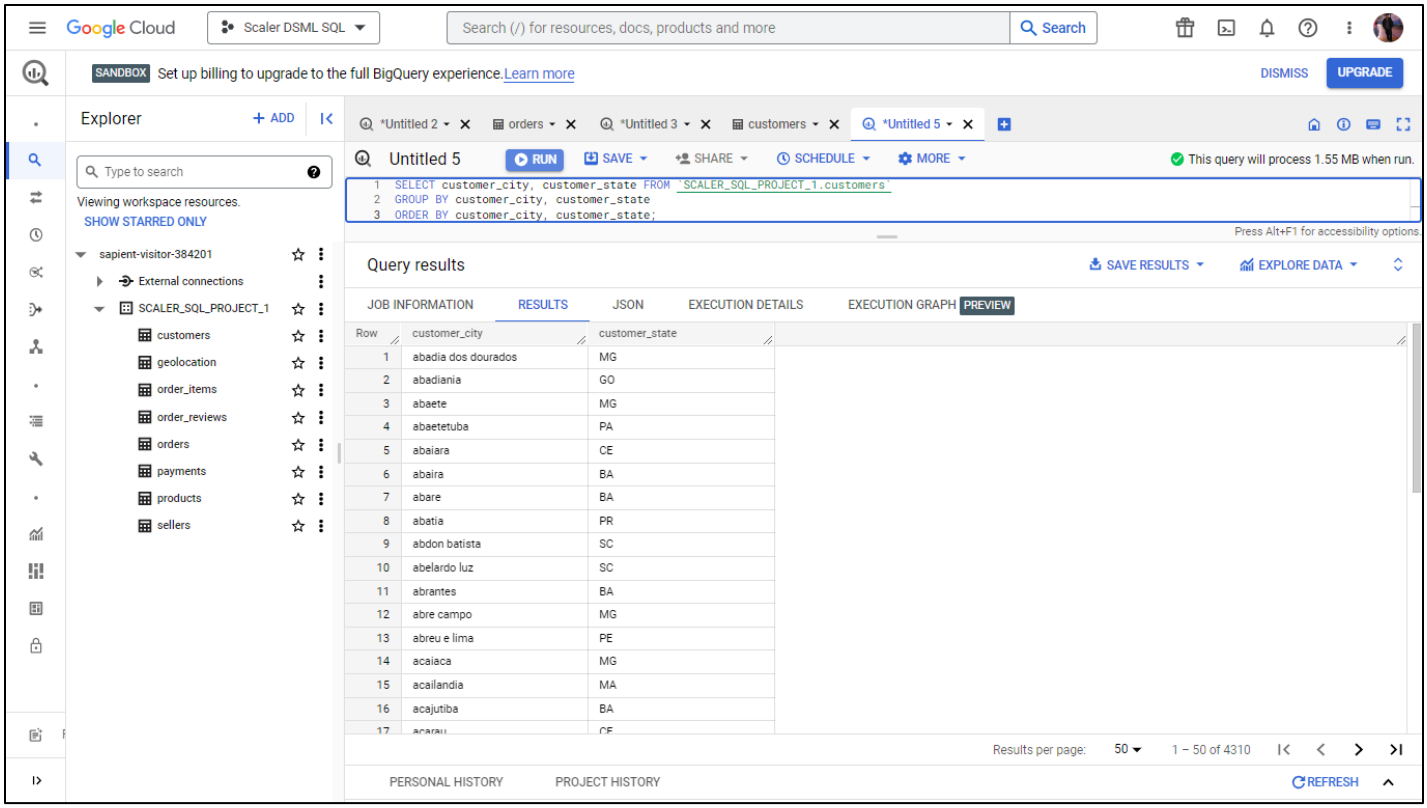
2. **Time period for which the data is given**

```
SELECT MIN(order_purchase_timestamp) AS Starting_time,
       MAX(order_purchase_timestamp) AS Ending_time
FROM `SCALER_SQL_PROJECT_1.orders`;
```



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

```
SELECT customer_city, customer_state FROM `SCALER_SQL_PROJECT_1.customers`
GROUP BY customer_city, customer_state
ORDER BY customer_city, customer_state;
```



Business Insights, Analysis and Recommendation:

- The given data contains data types: Integer, Float, String, Timestamp and Geolocation.
- All the columns are nullable which means they allow nulls.
- The data contains details of e-commerce business in Brazil which have order, customer, payments and sellers' information.
- The data have Orders between 4th September 2016 to 17th October 2018.
- The data have customers from 27 states and 4119 cities.

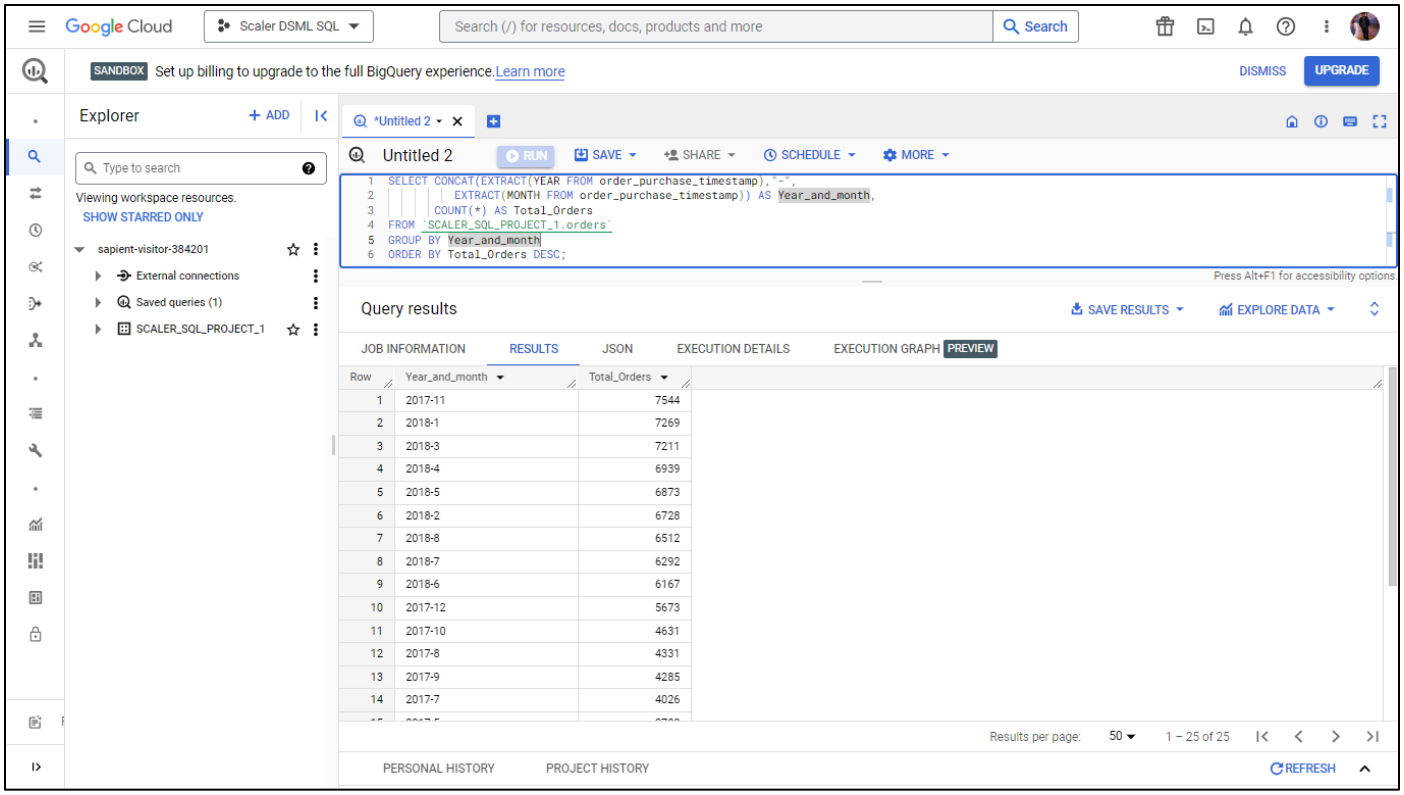
2. In-depth Exploration:

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?

```
SELECT CONCAT(EXTRACT(YEAR FROM order_purchase_timestamp),"-",
      EXTRACT(MONTH FROM order_purchase_timestamp)) AS Year_and_month,
    COUNT(*) AS Total_Orders
FROM `SCALER_SQL_PROJECT_1.orders`
GROUP BY Year_and_month
ORDER BY Total_Orders DESC;
```

Business Insights, Analysis and Recommendation:

- As we can see there is a growing trend in Brazil as the count of Orders is increasing month by month.
- Maximum order was received in November 2017.
- As per the given data and extracted result we cannot say about seasonality peaks in specific month. However, there is November 2017 data where the maximum orders were received and 2018 data is not available for November. There could be a probability of receiving maximum no of orders in 2018 as well.

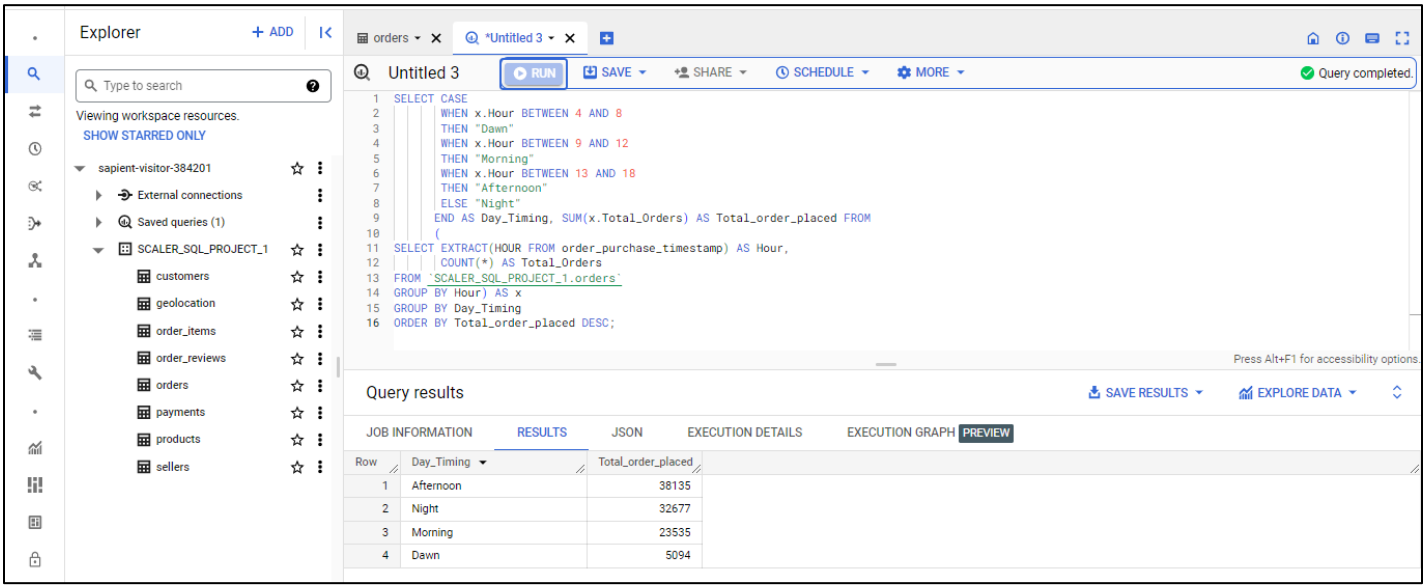


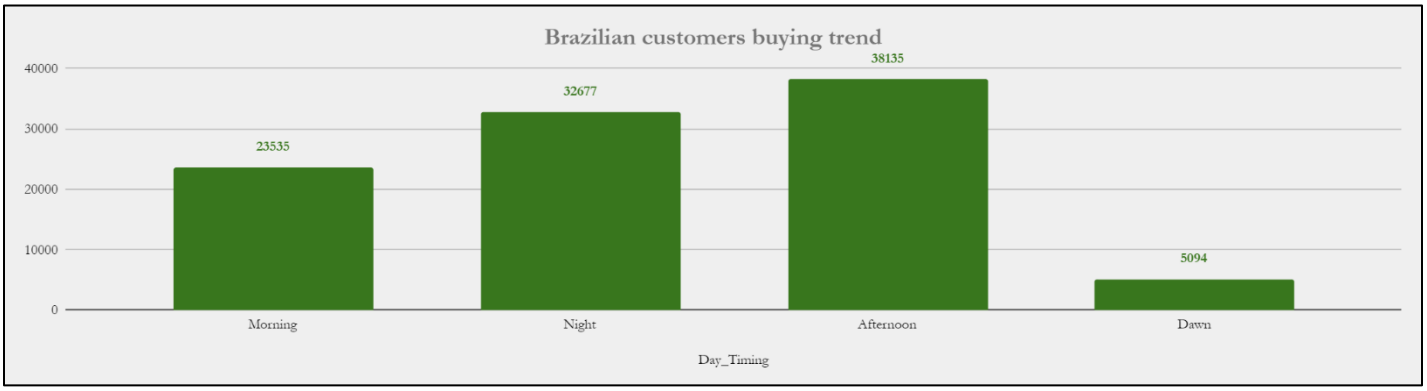
2. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

```
SELECT CASE
    WHEN x.Hour BETWEEN 4 AND 8
    THEN "Dawn"
    WHEN x.Hour BETWEEN 9 AND 12
    THEN "Morning"
    WHEN x.Hour BETWEEN 13 AND 18
    THEN "Afternoon"
    ELSE "Night"
   END AS Day_Timing, SUM(x.Total_Orders) AS Total_order_placed FROM
(SELECT EXTRACT(HOUR FROM order_purchase_timestamp) AS Hour,
    COUNT(*) AS Total_Orders
FROM `SCALER_SQL_PROJECT_1.orders`
GROUP BY Hour) AS x
GROUP BY Day_Timing
ORDER BY Total_order_placed DESC;
```

Business Insights, Analysis and Recommendation:

- Day-Timing have 24 Hour Format which segregated in day timing as per below:
  - ✓ Dawn timing is from 4 o'clock to 8 o'clock.
  - ✓ Morning timing start from 9 o'clock and ends at 12 o'clock.
  - ✓ Afternoon timing starts from 13 o'clock and ends at 18 o'clock.
  - ✓ Night timing start from 19 o'clock and ends at 3 o'clock
- As we can see we have received maximum number of orders in Afternoon timing and need to focus on Night timings where we have received very less orders.

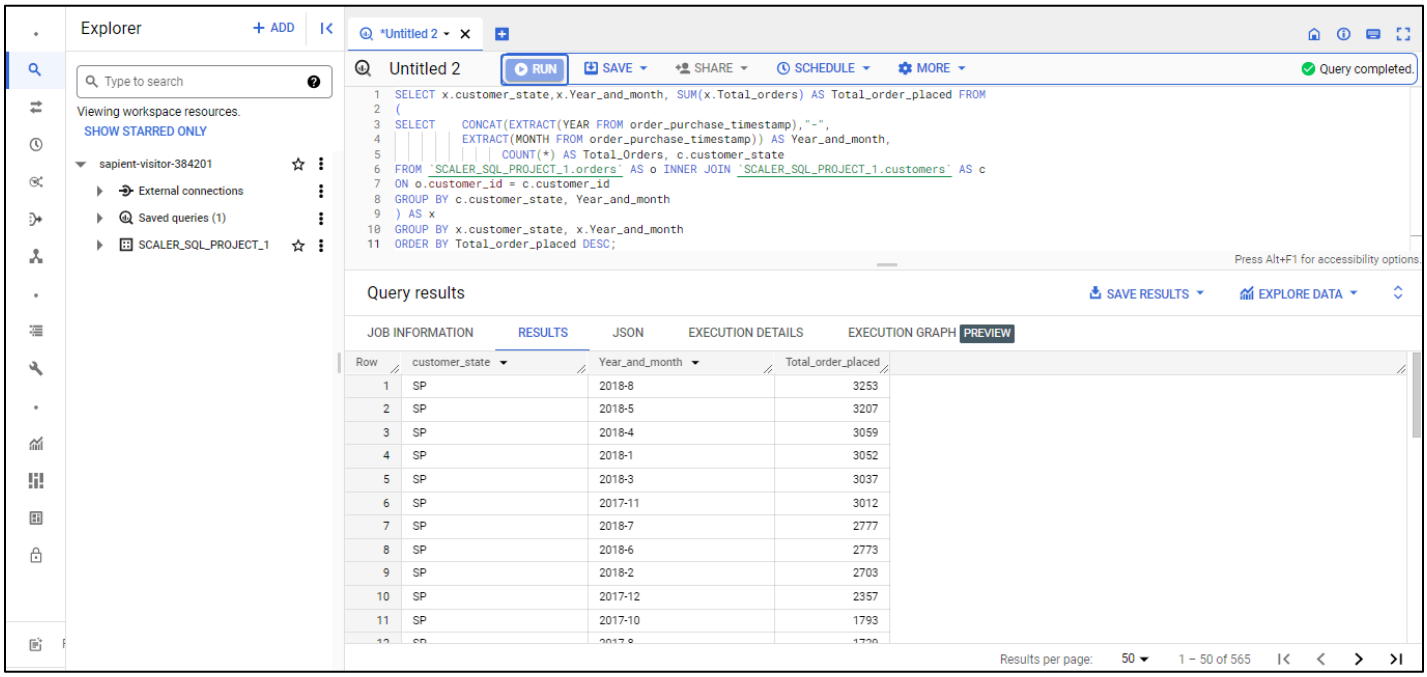




3. Evolution of E-commerce orders in the Brazil region:

1. Get month on month orders by states

```
SELECT x.customer_state,x.Year_and_month, SUM(x.Total_orders) AS Total_order_placed FROM
(
SELECT   CONCAT(EXTRACT(YEAR FROM order_purchase_timestamp),"-",
         EXTRACT(MONTH FROM order_purchase_timestamp)) AS Year_and_month,
         COUNT(*) AS Total_Orders, c.customer_state
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state, Year_and_month
) AS x
GROUP BY x.customer_state, x.Year_and_month
ORDER BY Total_order_placed DESC;
```



State wise Month on Month Sale

Customer State

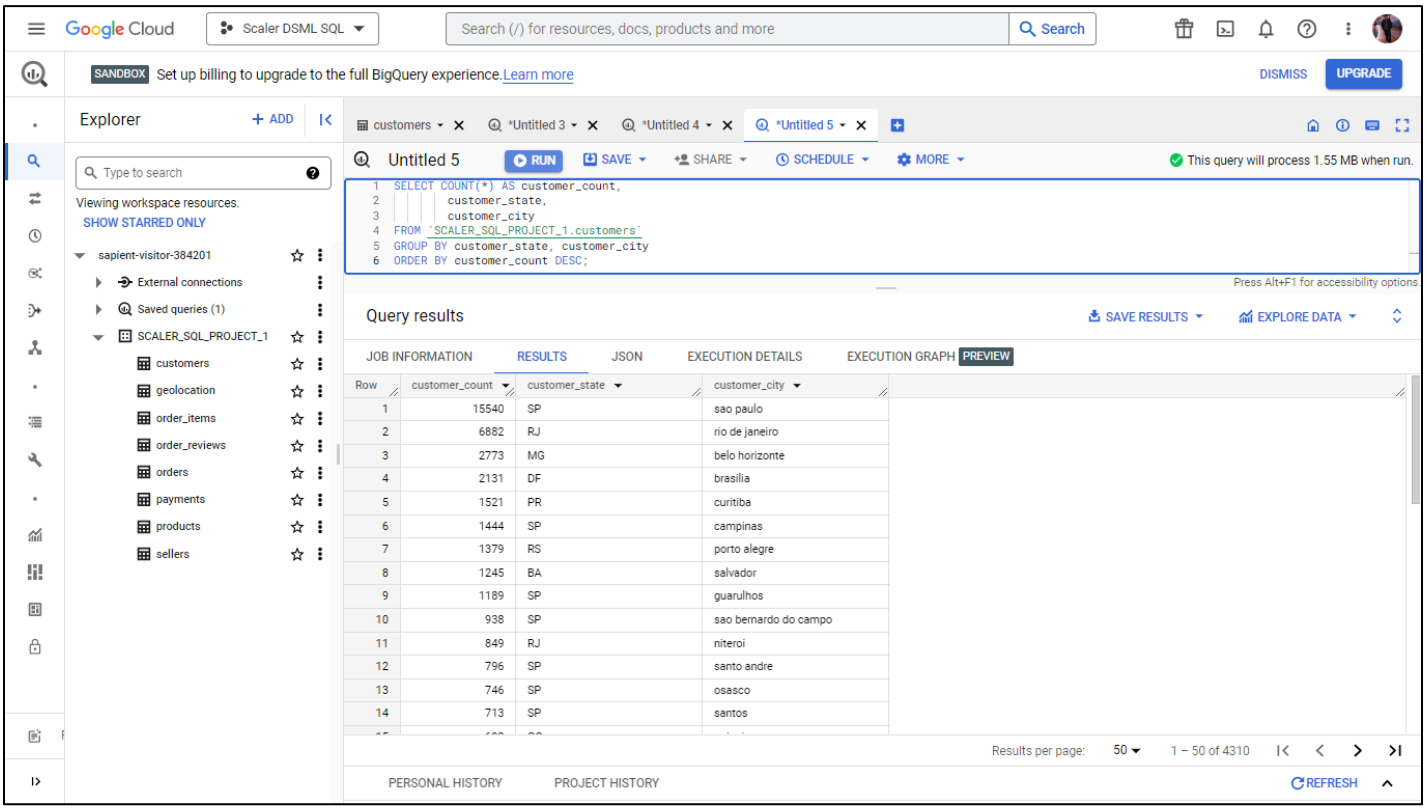
Year and mon..	AC	AL	AM	AP	BA	CE	DF	ES	GO	MA	MG	MS	MT	PA	PB	PE	PI	PR	RJ	RN	RO	RR	RS	SC	SE	SP	TO
2016-10						8	6		9		40					7		19	56				24	11		113	
2017-1					25	9	13	12	18	9	108		11	12		9	7	65	97	5			54	31		299	
2017-2		12	8		59	13	24	34	27	11	259	11	17	25	12	21	12	118	254	8	11		105	59	12	654	7
2017-3		10	5		91	28	57	48	53	24	358	20	16	36	16	45	13	127	395	13	16		151	110	25	1,010	8
2017-4	5	23	13		93	43	35	46	41	27	275	15	27	36	20	40	13	114	338	10	9		139	105	13	908	14
2017-5	8	27	10	5	127	62	64	94	87	33	428	29	37	35	18	68	25	213	488	17	9		208	152	11	1,425	18
2017-6	4	10			106	47	70	80	79	17	363	27	25	38	23	46	14	170	412	13	10		221	116	9	1,331	8
2017-7	5	17	5		155	53	77	83	77	39	453	25	38	39	27	73	20	203	571	27	11		249	158	14	1,604	
2017-8	4	18	5		158	73	87	95	93	40	469	24	38	60	16	85	22	223	562	20	14		299	159	20	1,729	15
2017-9	5	20	9		170	77	97	93	88	42	507	33	35	41	29	76	23	183	609	24	16		278	156	16	1,638	17
2017-10	6	28			166	66	98	100	108	48	560	34	52	54	30	80	23	206	668	23	14		252	178	22	1,793	13
2017-11	5	26	10		250	108	168	170	157	56	943	46	74	70	30	126	31	378	1,048	44	17		422	303	27	3,012	17
2017-12	5	14	6		192	81	131	113	127	41	691	36	50	58	37	103	23	270	783	30	11		283	193	20	2,357	14
2018-1	6	37	12	11	239	90	138	147	146	57	863	70	85	70	31	104	48	378	893	46	20		373	314	20	3,052	17
2018-2		27	8		214	88	172	152	149	56	804	64	67	58	35	125	34	342	922	23	14	5	368	257	15	2,703	21
2018-3		30	9	5	249	98	150	134	146	53	879	59	55	73	39	108	35	377	907	39	13	6	418	252	18	3,037	20
2018-4		28	6	5	225	100	148	142	136	46	786	43	65	71	31	114	37	386	834	32	11		349	246	14	3,059	19
2018-5		19	9	6	241	74	144	134	139	32	762	45	67	40	29	106	31	311	833	22	17		351	227	8	3,207	16
2018-6		24	7		201	74	150	124	105	42	717	49	58	54	28	94	29	308	716	36	12	5	305	205	28	2,773	18
2018-7		23	18	6	250	87	166	123	115	40	658	49	47	57	52	137	32	320	717	29	16	5	316	198	28	2,777	22
2018-8		16			165	57	145	105	120	30	708	35	40	44	30	85	21	333	745	20	9		300	206	23	3,253	13
2018-9																										8	

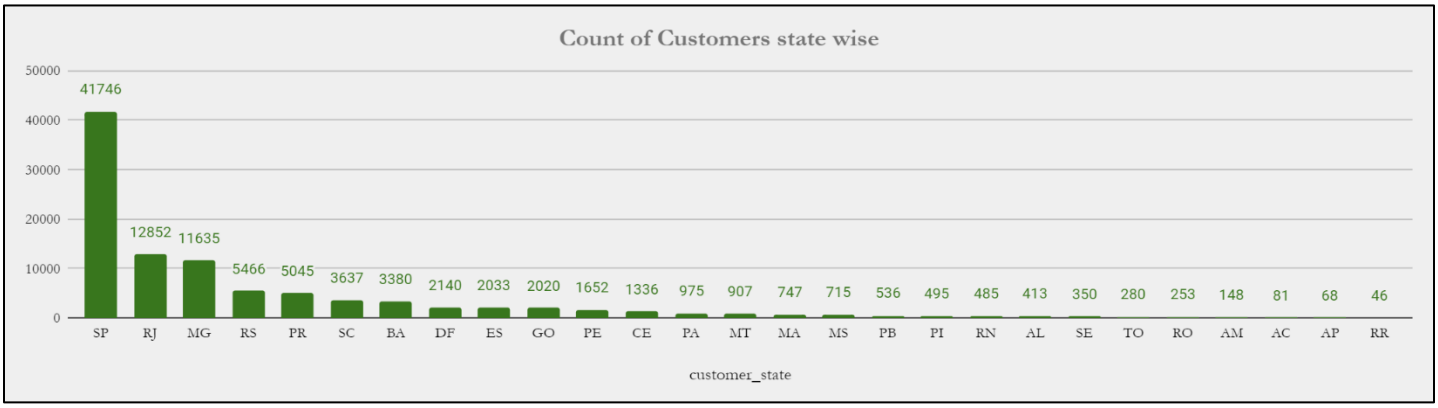
2. **Distribution of customers across the states in Brazil**

```
SELECT COUNT(*) AS customer_count,
    customer_state,
    customer_city
FROM `SCALER_SQL_PROJECT_1.customers`
GROUP BY customer_state, customer_city
ORDER BY customer_count DESC;
```

Business Insights, Analysis and Recommendation:

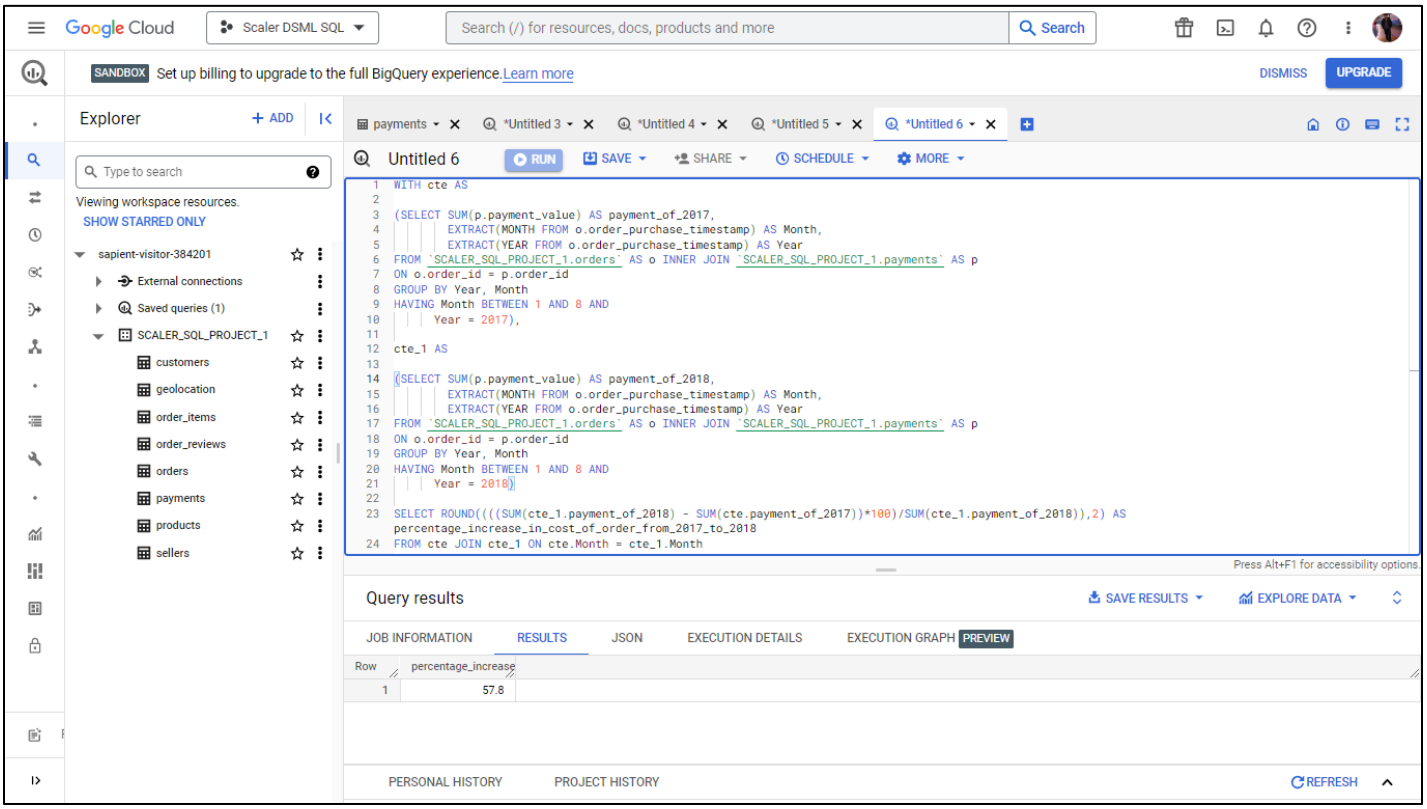
- SP state is dominating in terms of receiving maximum orders compare to order states.
- Maximum orders were placed in August 2018 in SP state.
- 41746 orders placed in SP state out of 99441 orders.
- SP state have placed 42% of total order placed.
- RJ state have placed 2nd highest orders followed by MG state which placed 11635 orders.
- RR, AP & AC received very few orders. 46, 68 & 81 is the count of order placed in these states respectively.
- As SP state have maximum numbers of customers 15540, which can be a reason to placed maximum number of orders as well, followed by RJ 6882 & MG 2773.
- As per extracted data we can see which state have maximum customers placed maximum orders, hence business should invest to build customers from different states which will help to have more orders from different states.





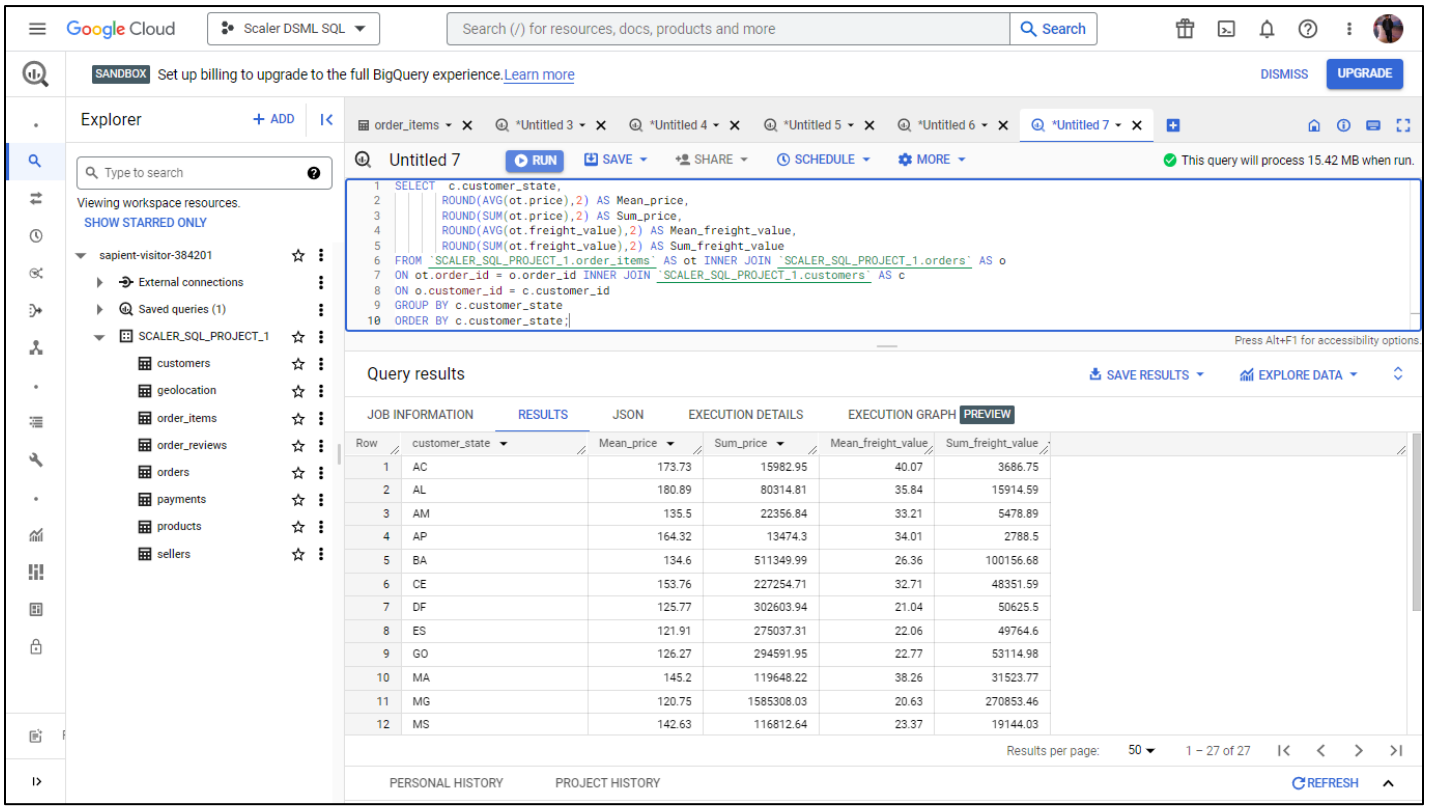
4. **Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.**
    1. **Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use "payment_value" column in payments table**

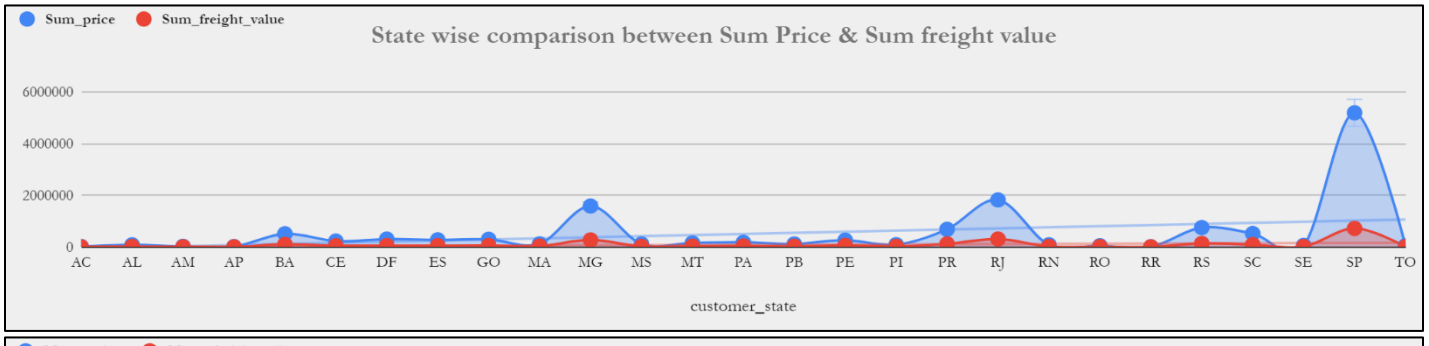
```
WITH cte AS
(
SELECT SUM(p.payment_value) AS payment_of_2017,
      EXTRACT(MONTH FROM o.order_purchase_timestamp) AS Month,
      EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.payments` AS p
ON o.order_id = p.order_id
GROUP BY Year, Month
HAVING Month BETWEEN 1 AND 8 AND
    Year = 2017 ),

cte_1 AS

(SELECT SUM(p.payment_value) AS payment_of_2018,
      EXTRACT(MONTH FROM o.order_purchase_timestamp) AS Month,
      EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.payments` AS p
ON o.order_id = p.order_id
GROUP BY Year, Month
HAVING Month BETWEEN 1 AND 8 AND
    Year = 2018 )

SELECT ROUND((((SUM(cte_1.payment_of_2018) -
SUM(cte.payment_of_2017))*100)/SUM(cte_1.payment_of_2018)),2) AS
percentage_increase_in_cost_of_order_from_2017_to_2018
FROM cte JOIN cte_1 ON cte.Month = cte_1.Month
```

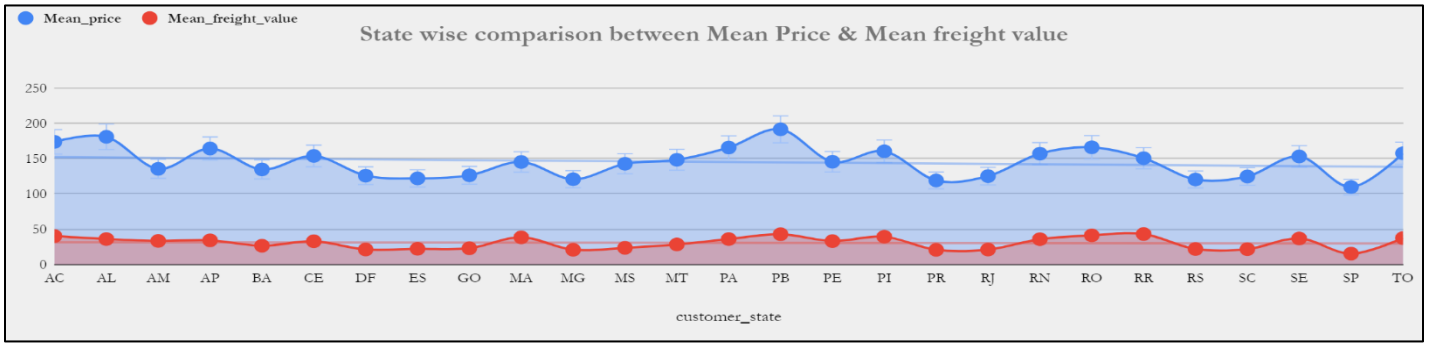


2. Mean & Sum of price and freight value by customer state

```
SELECT  c.customer_state,
     ROUND(AVG(ot.price),2) AS Mean_price,
     ROUND(SUM(ot.price),2) AS Sum_price,
     ROUND(AVG(ot.freight_value),2) AS Mean_freight_value,
     ROUND(SUM(ot.freight_value),2) AS Sum_freight_value
FROM `SCALER_SQL_PROJECT_1.order_items` AS ot INNER JOIN `SCALER_SQL_PROJECT_1.orders` AS o
ON ot.order_id = o.order_id INNER JOIN `SCALER_SQL_PROJECT_1.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY c.customer_state;
```







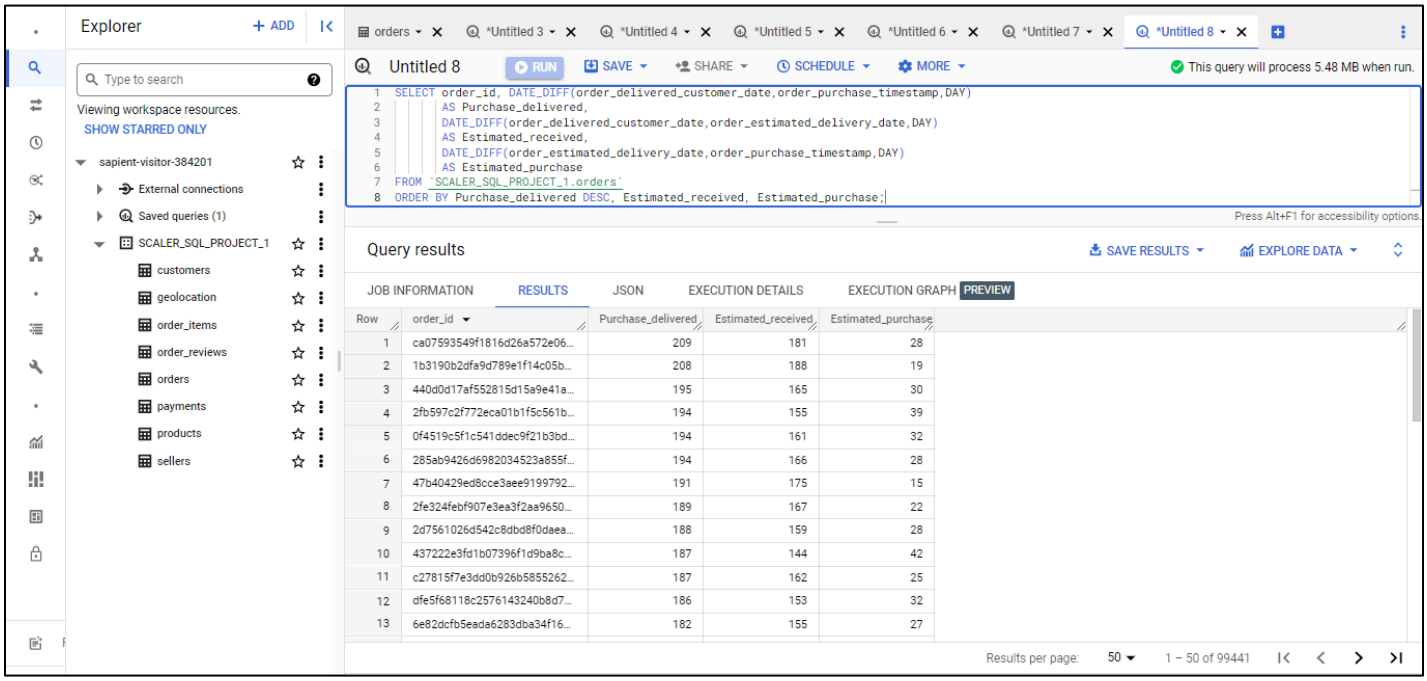
Business Insights, Analysis and Recommendation:

- There is growing trend in Brazil get 57.8% increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only)
- SP state have dominance in terms of order received, customer count and now we can see there is highest revenue generated with SP state with high freight cost.
- Freight value is highest due to maximum order placed from the state to deliver the product to customer.
- Mean freight cost is high in RR, PB, RO, AC, PI in these places providing a free delivery above a certain order value could increase the orders from that region along with revenue

5. Analysis on sales, freight and delivery time

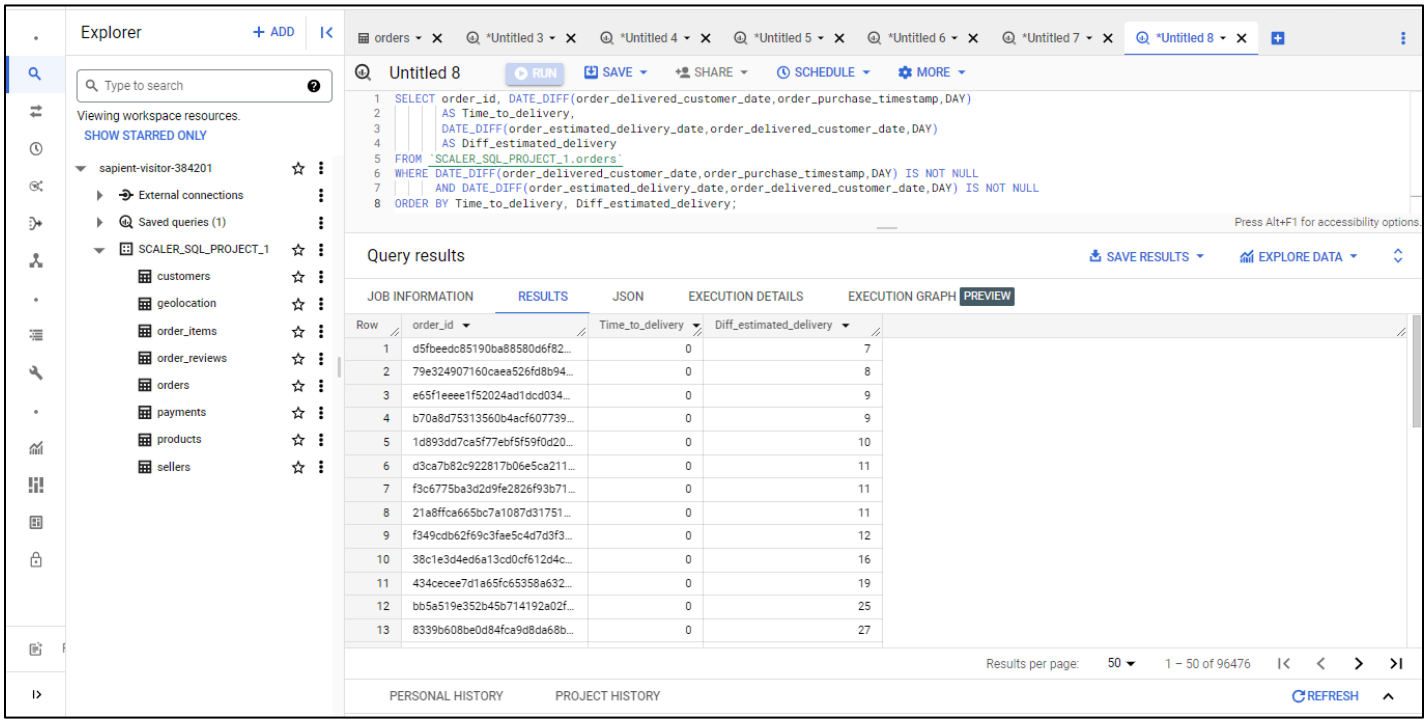
1. Calculate days between purchasing, delivering and estimated delivery

```
SELECT order_id, DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)
    AS Purchase_delivered,
    DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY)
    AS Estimated_received,
    DATE_DIFF(order_estimated_delivery_date,order_purchase_timestamp,DAY)
    AS Estimated_purchase
FROM `SCALER_SQL_PROJECT_1.orders`
ORDER BY Purchase_delivered DESC, Estimated_received, Estimated_purchase;
```



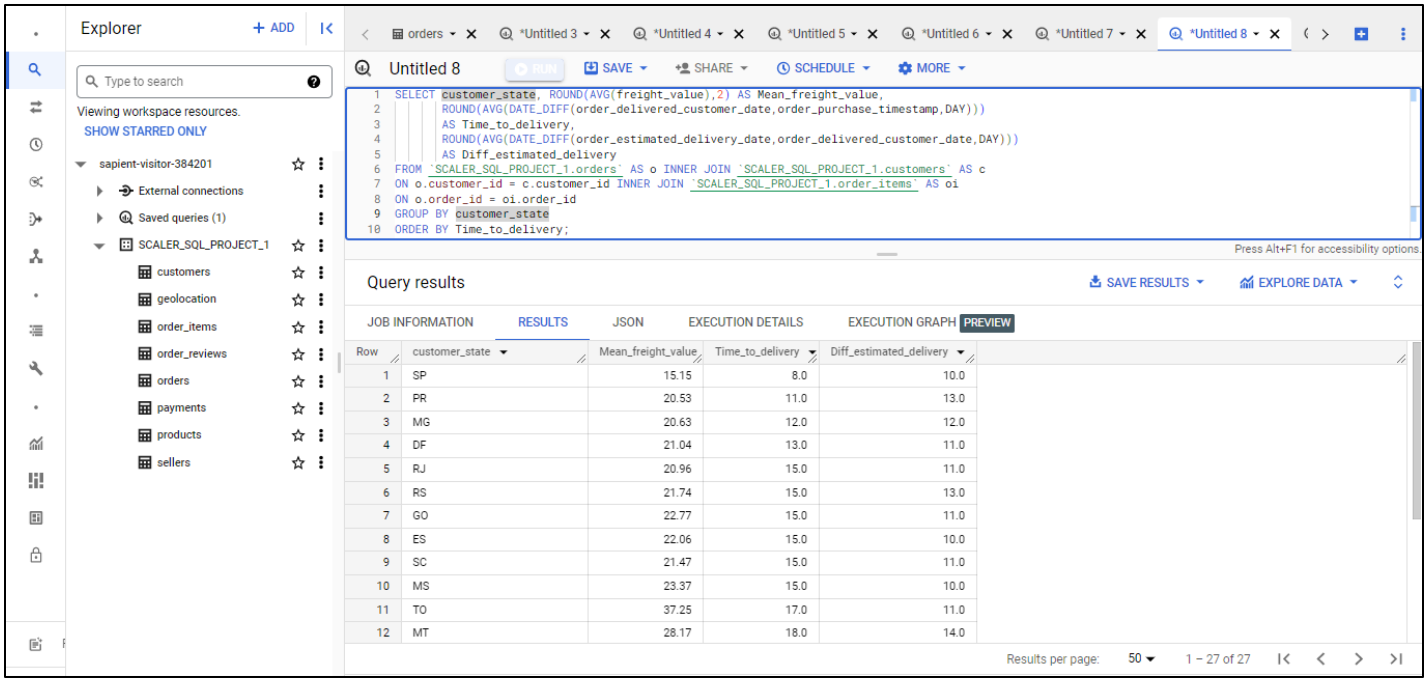
2. **Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:**
   - **time_to_delivery = order_delivered_customer_date-order_purchase_timestamp**
   - **diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date**

```
SELECT order_id, DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)
    AS Time_to_delivery,
    DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)
    AS Diff_estimated_delivery
FROM `SCALER_SQL_PROJECT_1.orders`
WHERE DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) IS NOT NULL
   AND DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) IS NOT NULL
ORDER BY Time_to_delivery, Diff_estimated_delivery;
```



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

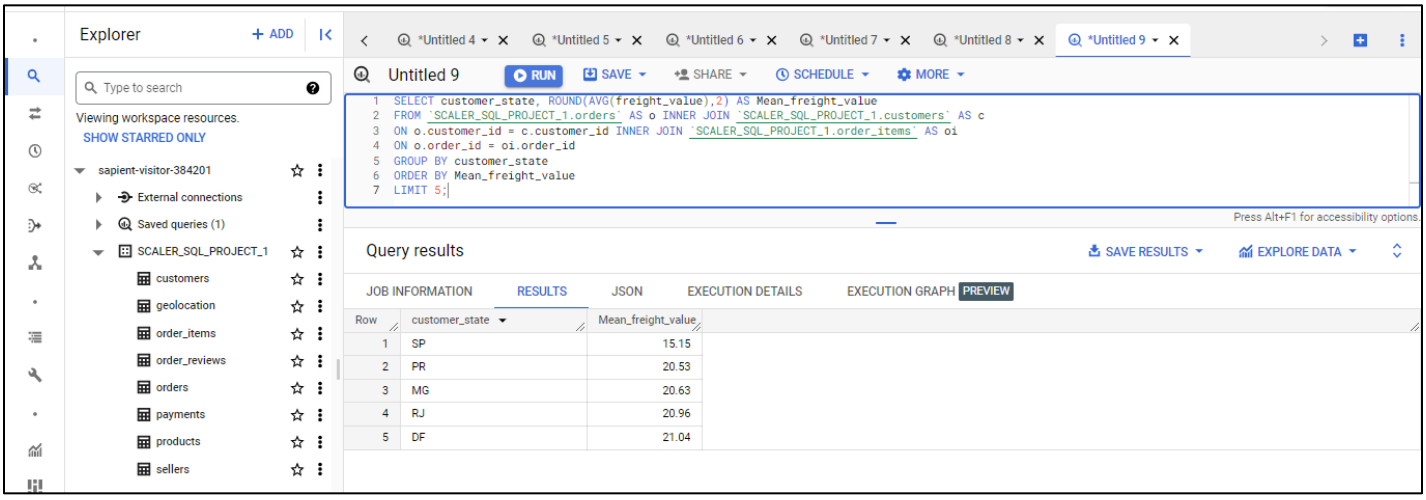
```
SELECT customer_state, ROUND(AVG(freight_value),2) AS Mean_freight_value,
    ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)))
    AS Time_to_delivery,
    ROUND(AVG(DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY)))
    AS Diff_estimated_delivery
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.customers` AS c
ON o.customer_id = c.customer_id INNER JOIN `SCALER_SQL_PROJECT_1.order_items` AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
ORDER BY Time_to_delivery;
```



4. **Sort the data to get the following:**
5. **Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5**

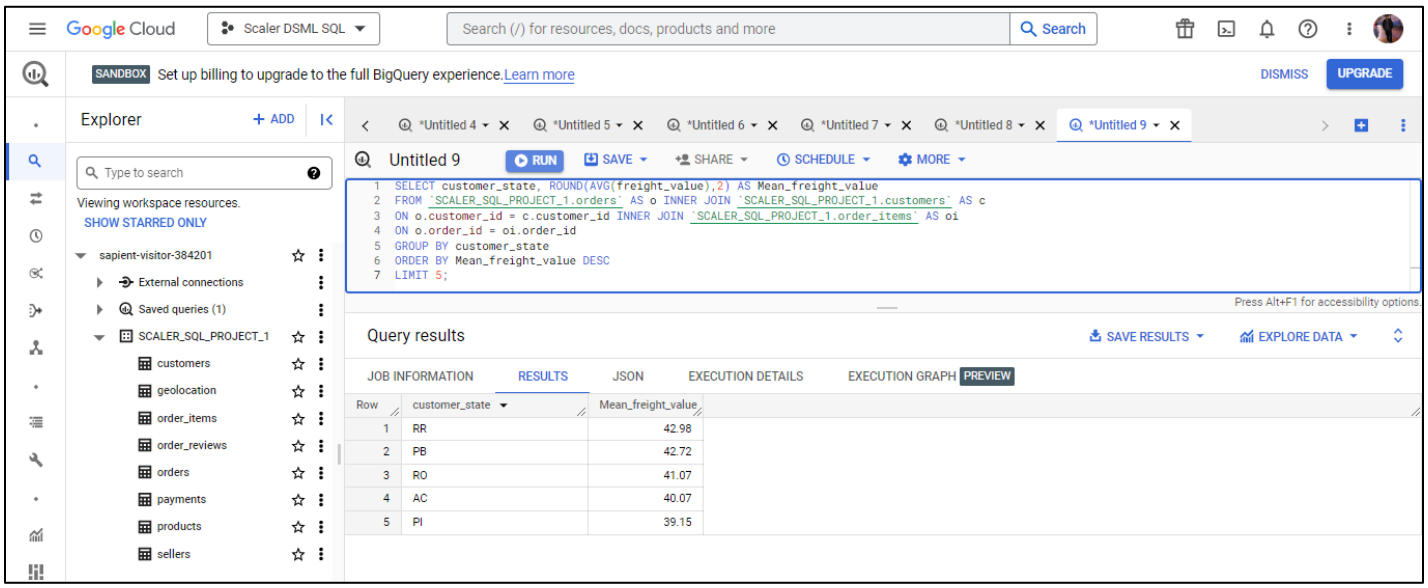
Top 5 states which have lowest average freight value:

```
SELECT customer_state, ROUND(AVG(freight_value),2) AS Mean_freight_value
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.customers` AS c
ON o.customer_id = c.customer_id INNER JOIN `SCALER_SQL_PROJECT_1.order_items` AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
ORDER BY Mean_freight_value
LIMIT 5;
```



Top 5 states which have highest average freight value:

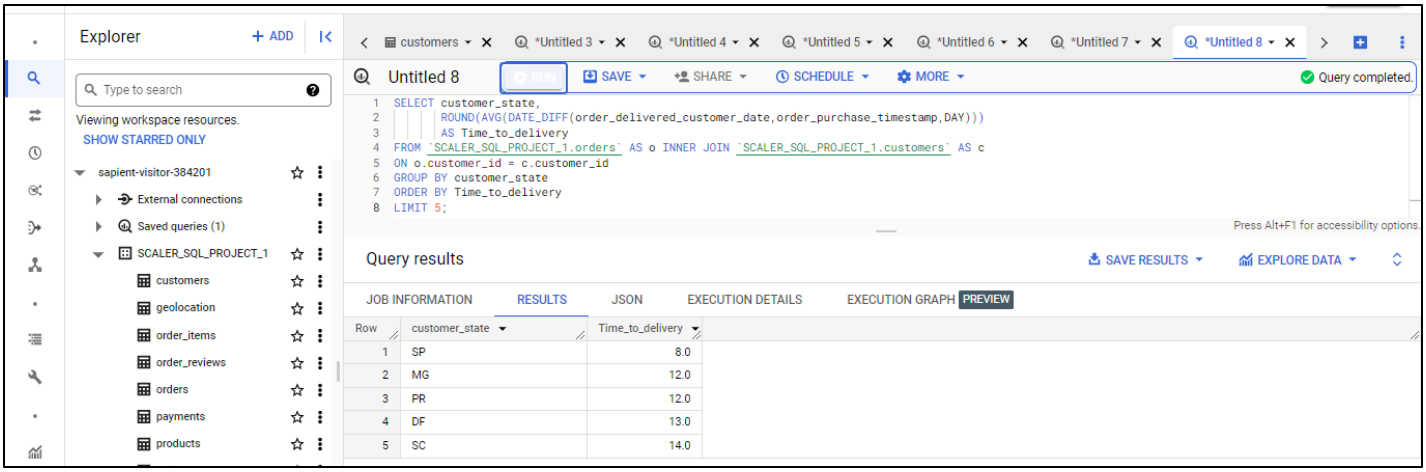
```
SELECT customer_state, ROUND(AVG(freight_value),2) AS Mean_freight_value
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.customers` AS c
ON o.customer_id = c.customer_id INNER JOIN `SCALER_SQL_PROJECT_1.order_items` AS oi
ON o.order_id = oi.order_id
GROUP BY customer_state
ORDER BY Mean_freight_value DESC
LIMIT 5;
```



6. Top 5 states with highest/lowest average time to delivery

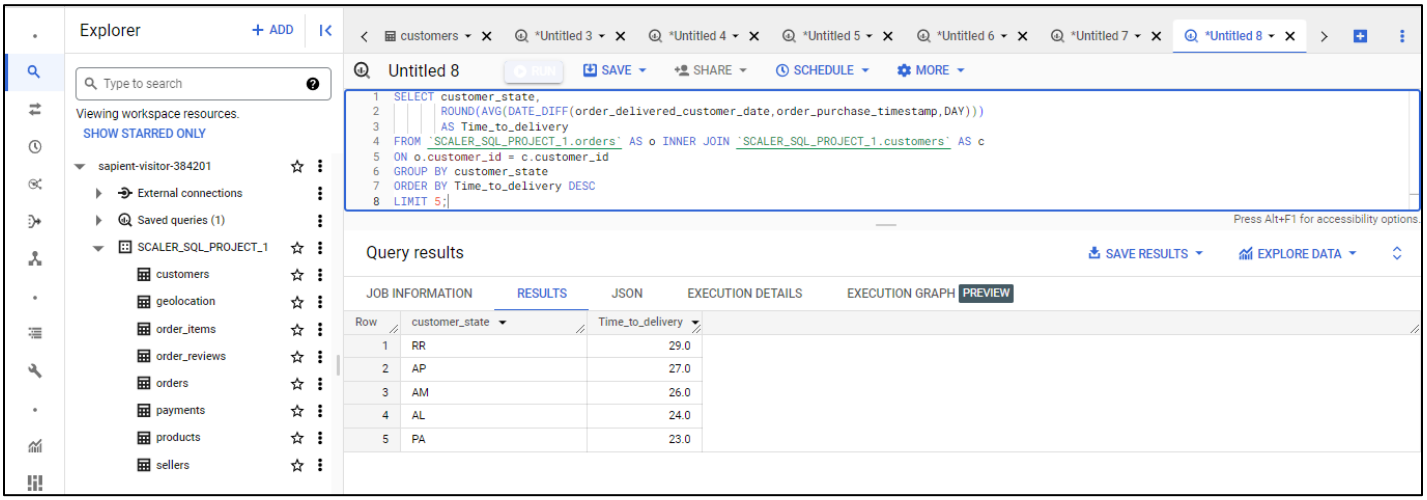
Top 5 states with lowest average time to delivery

```
SELECT customer_state,
    ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)))
    AS Time_to_delivery
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY customer_state
ORDER BY Time_to_delivery
LIMIT 5;
```



Top 5 states with highest average time to delivery

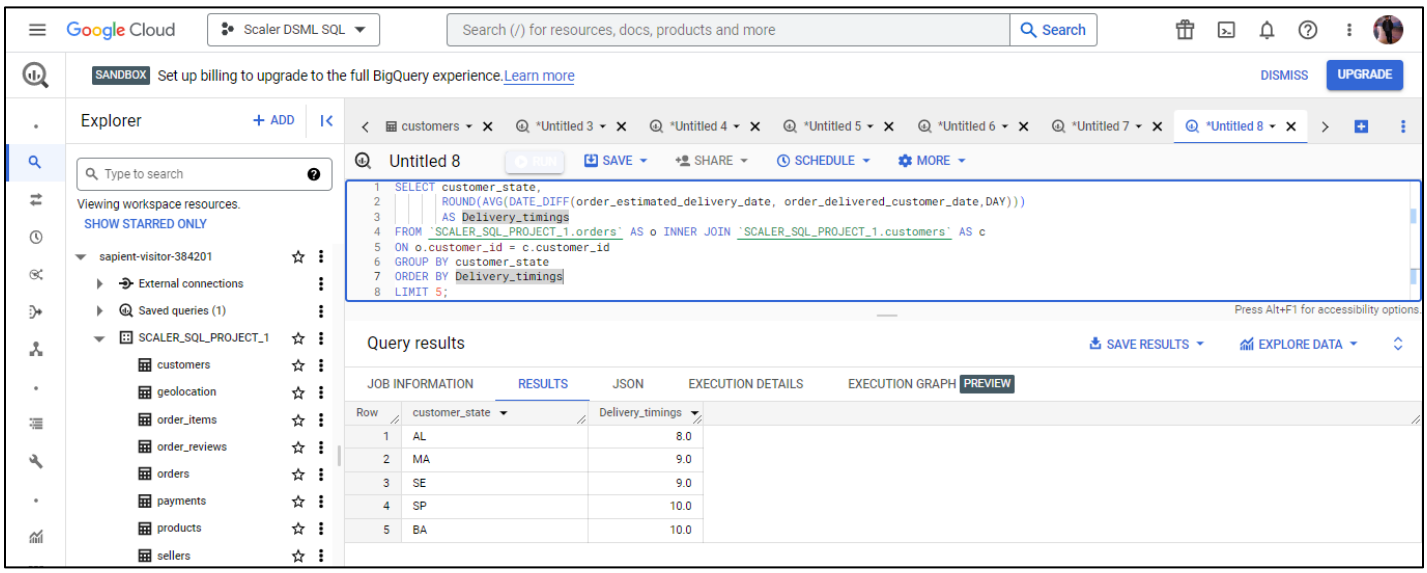
```
SELECT customer_state,
    ROUND(AVG(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY)))
    AS Time_to_delivery
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY customer_state
ORDER BY Time_to_delivery DESC
LIMIT 5;
```



7. **Top 5 states where delivery is really fast/ not so fast compared to estimated date**

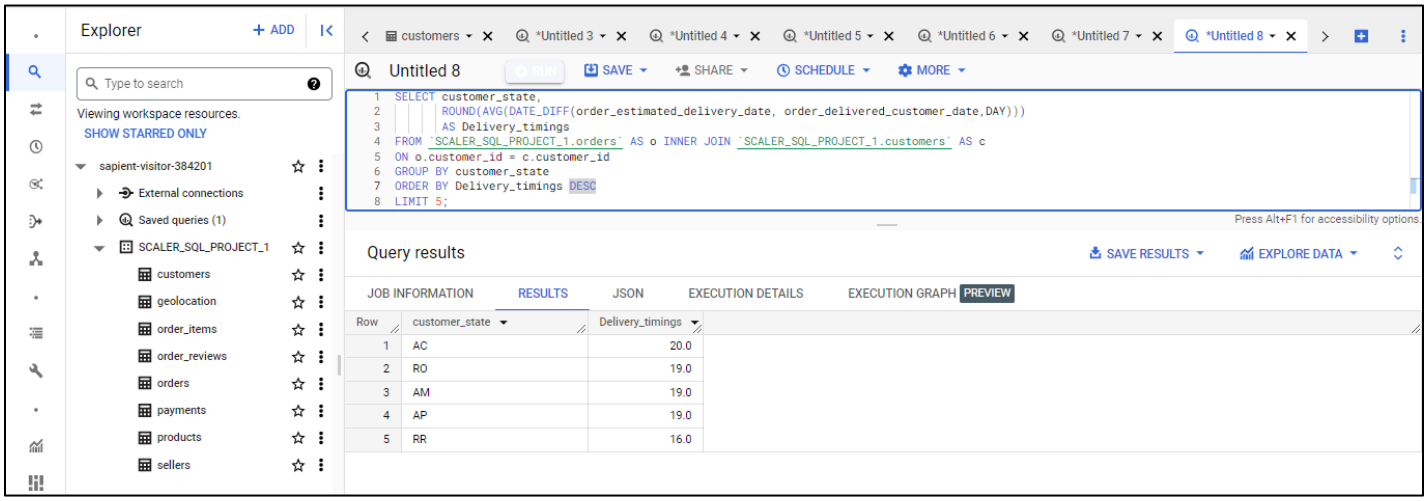
Top 5 states where delivery is really fast compared to estimated date:

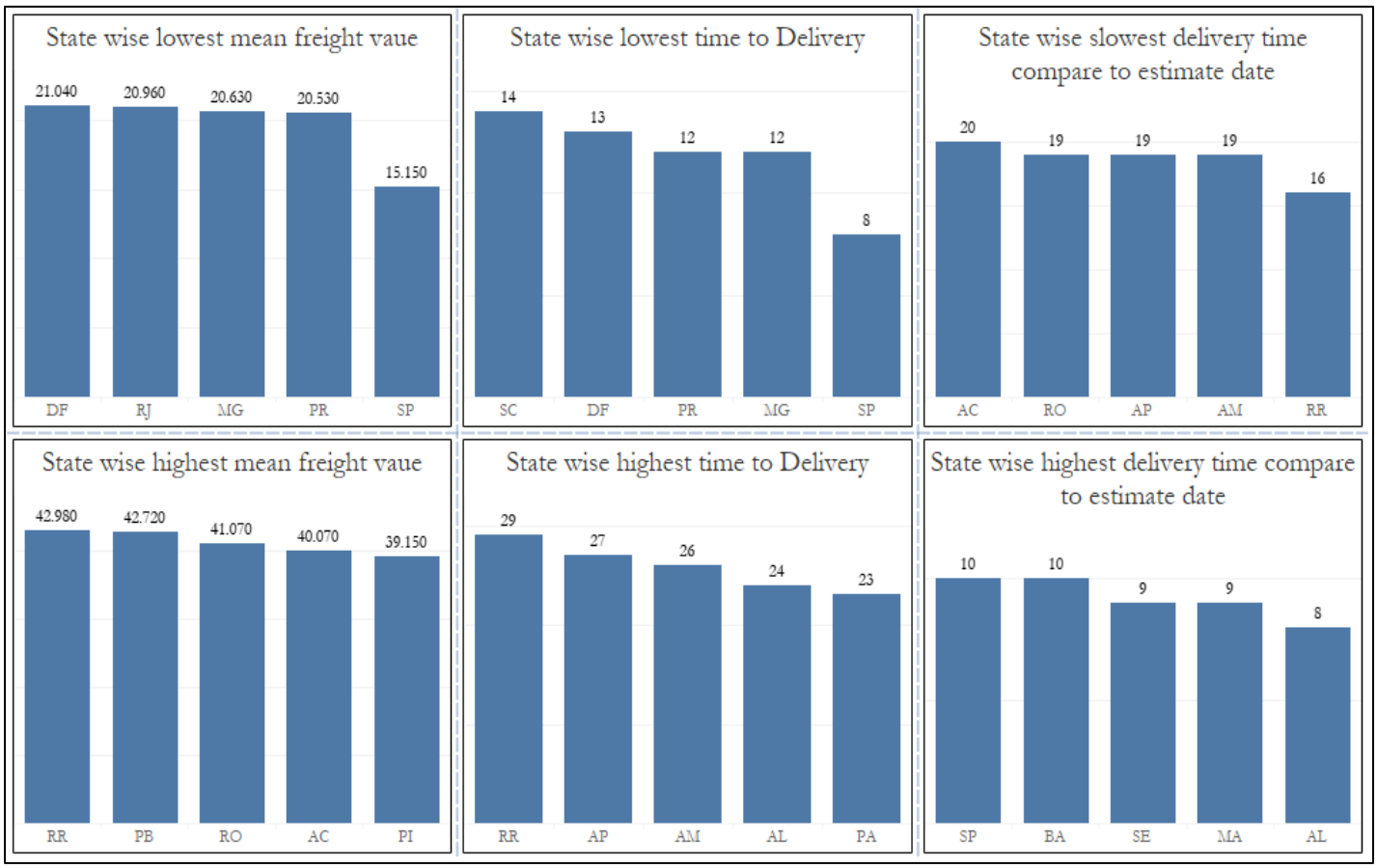
```
SELECT customer_state,
    ROUND(AVG(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date,DAY)))
    AS Delivery_timings
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY customer_state
ORDER BY Delivery_timings
LIMIT 5;
```



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

```sql
SELECT customer_state,
    ROUND(AVG(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date,DAY)))
    AS Delivery_timings
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY customer_state
ORDER BY Delivery_timings DESC
LIMIT 5;
```





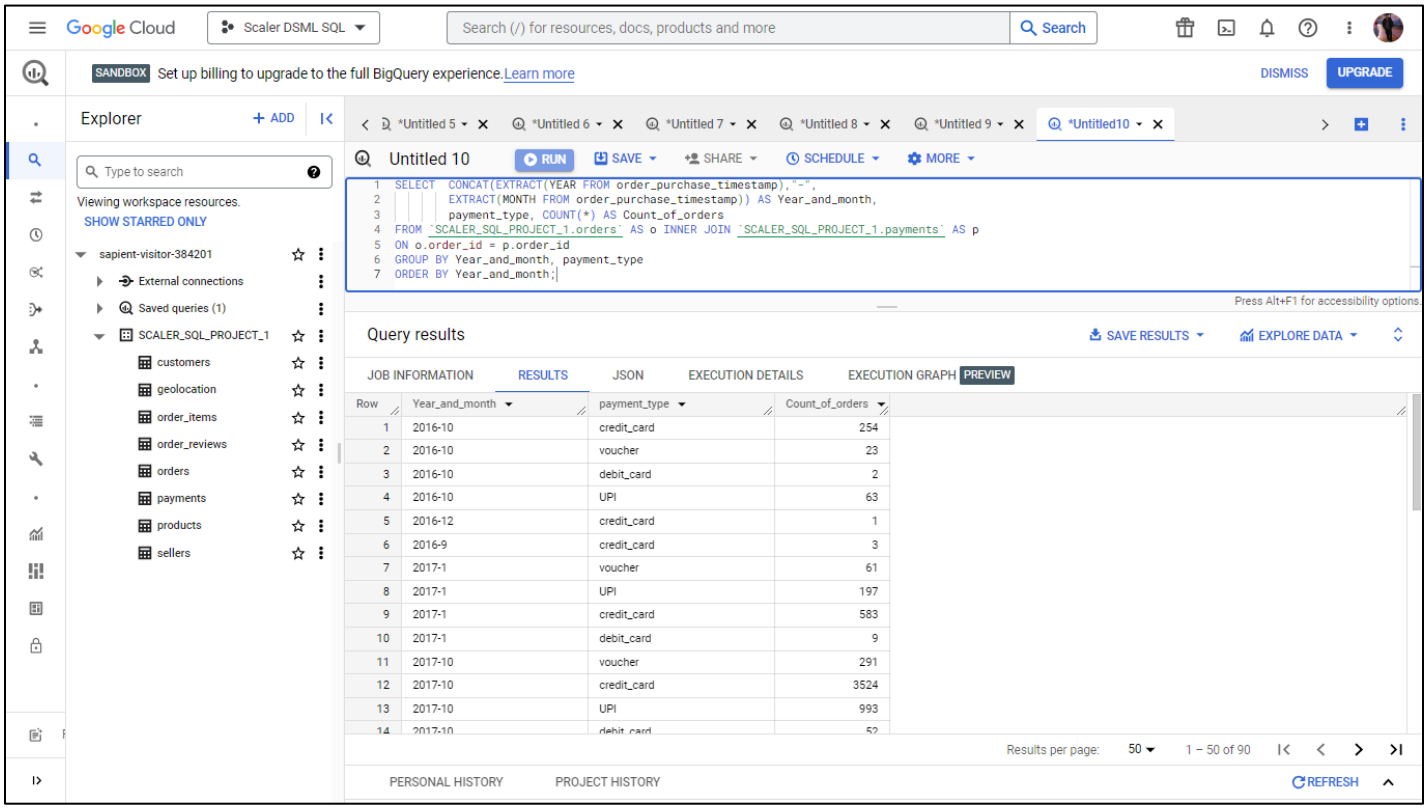
Business Insights, Analysis and Recommendation:

- If the difference is in positive, then it shows the number of days order delayed by those days, if incase the difference is in Negative, then it shows the number of days order received early.
- States SP, PR, MG, DF have the lowest Freight cost and delivery time which due to warehouse and sellers might be in these regions.
- Increment in number of warehouses across the country might increase the number of orders and help to increase the revenue as well.
- Business should invest to have more sellers in across the country which can help to deliver the item earliest to customer which will somehow help to build customer relation and we can see increment in number of orders.

6. Payment type analysis:

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

```
SELECT  CONCAT(EXTRACT(YEAR FROM order_purchase_timestamp),"-",
    EXTRACT(MONTH FROM order_purchase_timestamp)) AS Year_and_month,
    payment_type, COUNT(*) AS Count_of_orders
FROM `SCALER_SQL_PROJECT_1.orders` AS o INNER JOIN `SCALER_SQL_PROJECT_1.payments` AS p
ON o.order_id = p.order_id
GROUP BY Year_and_month, payment_type
ORDER BY Year_and_month;
```

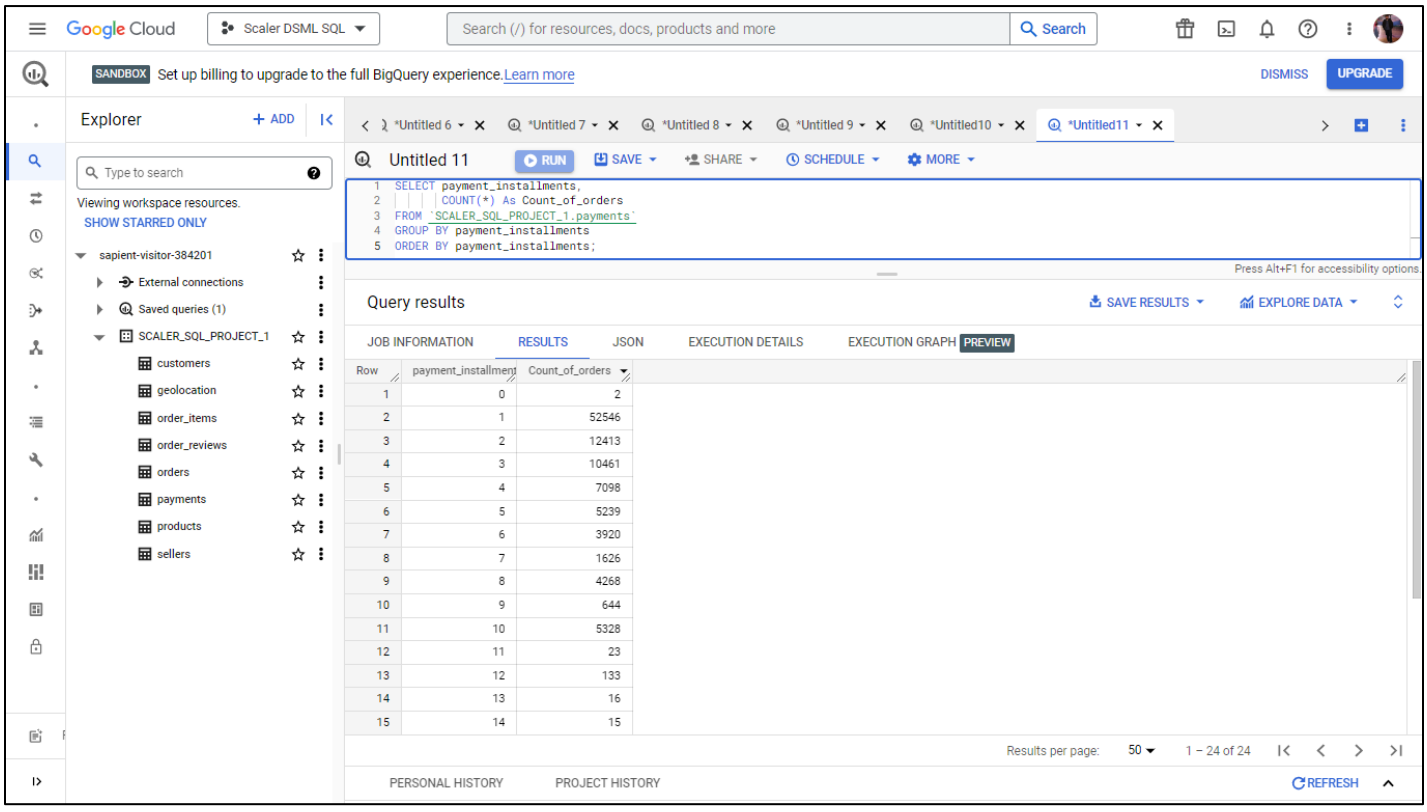


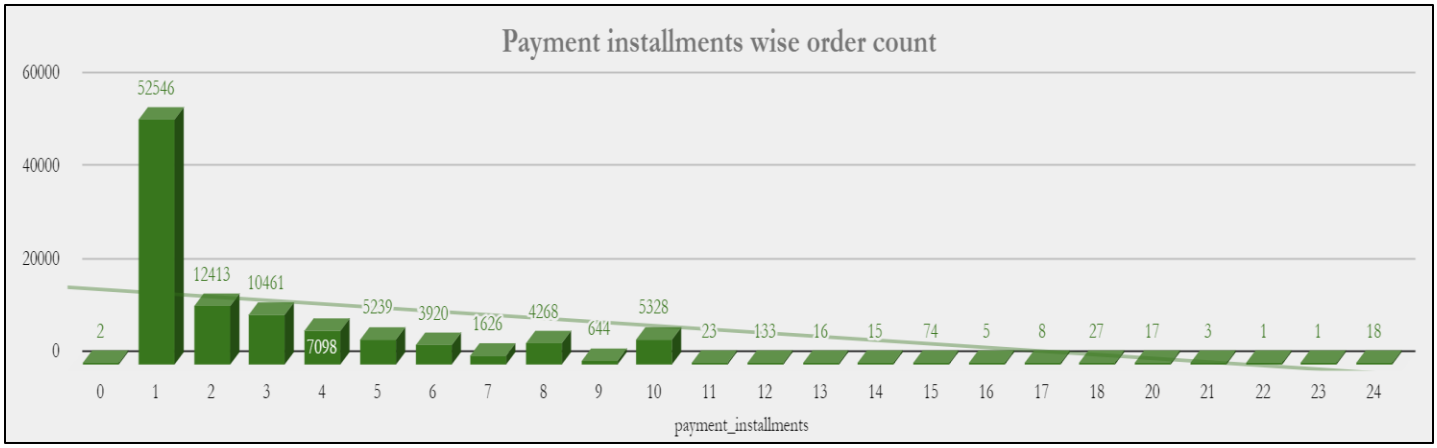
Month over Month count of orders for different payment types

F1	F2 credit_card	debit_card	not_defined	UPI	voucher
2016-9	3				
2016-10	254	2		63	23
2016-12	1				
2017-1	583	9		197	61
2017-2	1,356	13		398	119
2017-3	2,016	31		590	200
2017-4	1,846	27		496	202
2017-5	2,853	30		772	289
2017-6	2,463	27		707	239
2017-7	3,086	22		845	364
2017-8	3,284	34		938	294
2017-9	3,283	43		903	287
2017-10	3,524	52		993	291
2017-11	5,897	70		1,509	387
2017-12	4,377	64		1,160	294
2018-1	5,520	109		1,518	416
2018-2	5,253	69		1,325	305
2018-3	5,691	78		1,352	391
2018-4	5,455	97		1,287	370
2018-5	5,497	51		1,263	324
2018-6	4,813	182		1,100	324
2018-7	4,755	242		1,229	281
2018-8	4,985	277	2	1,139	295
2018-9			1		15
2018-10					4

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

```
SELECT payment_installments,
    COUNT(*) As Count_of_orders
FROM `SCALER_SQL_PROJECT_1.payments`
GROUP BY payment_installments
ORDER BY payment_installments;
```





Business Insights, Analysis and Recommendation:

- Credit card is the most preferred payment type using payments to placed order by customer.
- Business can provide offers while using credit card by customer to attract them to increase value of orders for example we can provide cashback offer if customer is placing order for more than 2999. Which will attract customer to add more items and which will increase are sale.
- UPI method should also be in place while placing order as these days UPI is in trend for paying the amount direct from bank account.
- Most of the orders placed with full payment while placing orders, business should provide offers on EMI option so those customers would be attractive who cannot pay the complete amount at a time. So, they would also place order and net revenue of business would be increase.