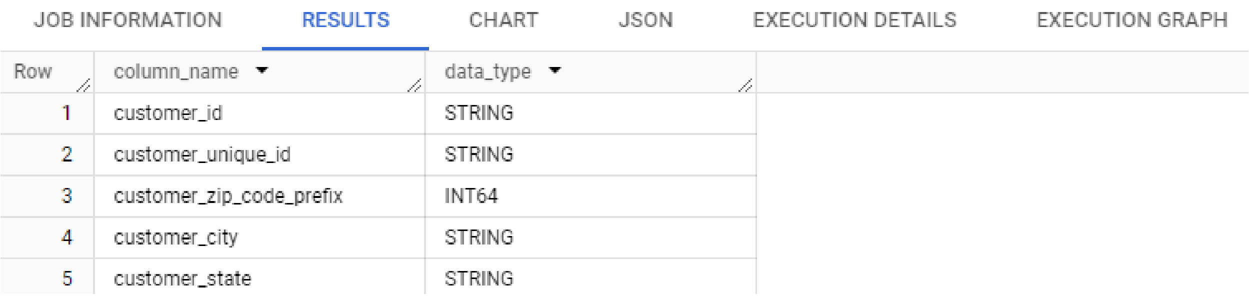
TARGET SQL BUSINESS CASE

Q1.1 Data type of all columns in the "customers" table.

```
SELECT column_name, data_type
FROM `T_company.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers'
```

Insight & Recommendation – Above code represents all the data type present in the table.

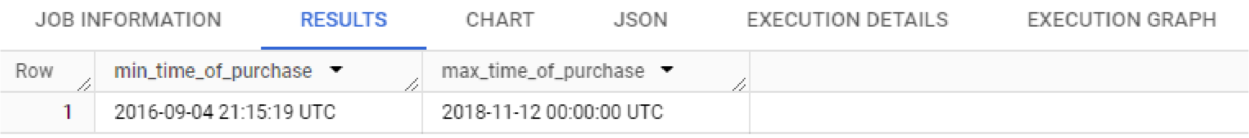


Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

Q1.2 Get the time range between which the orders were placed.

```
SELECT MIN(order_purchase_timestamp) AS min_time_of_purchase,
MAX(order_estimated_delivery_date) AS max_time_of_purchase
FROM `T_company.orders`, `T_company.order_items`
```

Insight & Recommendation – Above code represents the time when the customer is mostly active.

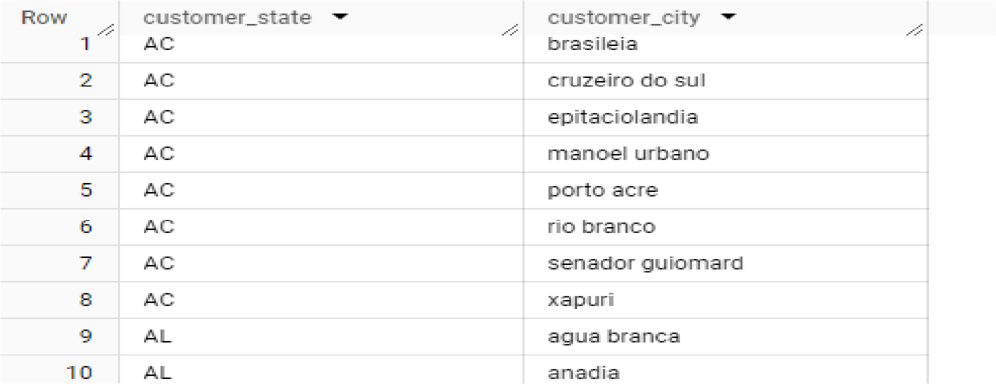


Row	min_time_of_purchase	max_time_of_purchase
1	2016-09-04 21:15:19 UTC	2018-11-12 00:00:00 UTC

Q1.3 Count the Cities & States of customers who ordered during the given period.

```
SELECT DISTINCT customer_state, customer_city
FROM `T_company.customers`
GROUP BY 1, 2
ORDER BY 1, 2
```

Insight & Recommendation – Above code represents the distinct customer from city did order in a given period.

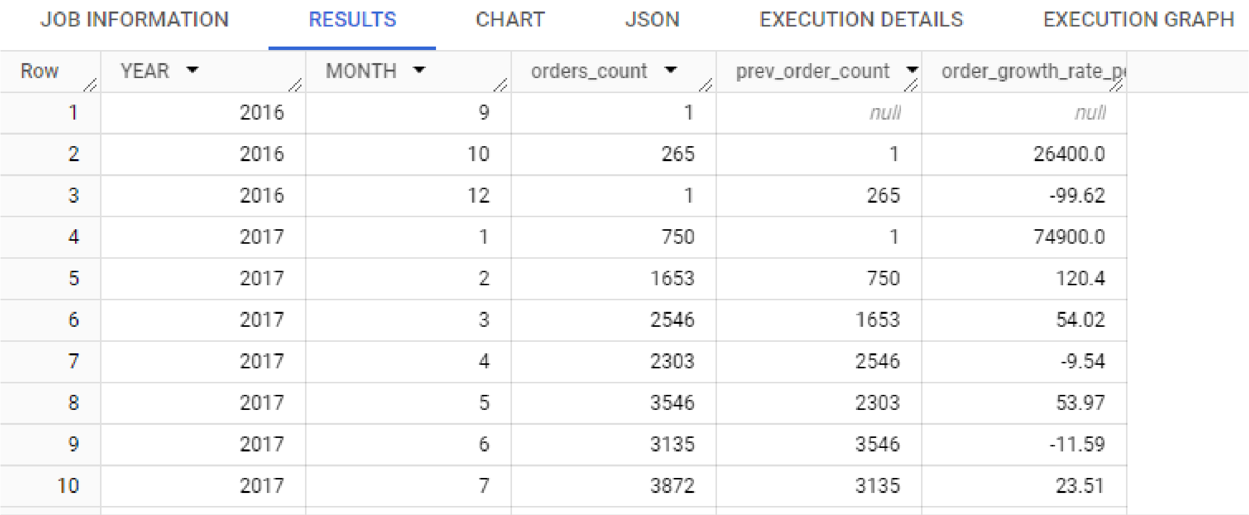


Row	customer_state	customer_city
1	AC	brasileia
2	AC	cruzeiro do sul
3	AC	epitaciolandia
4	AC	manoel urbano
5	AC	porto acre
6	AC	rio branco
7	AC	senador guiomard
8	AC	xapuri
9	AL	agua branca
10	AL	anadia

```
## Q2.1 Is there a growing trend in the no. of orders placed over the past years?

SELECT *, ROUND(((orders_count - prev_order_count) / prev_order_count) * 100, 2) AS
order_growth_rate_percent FROM
(SELECT *, LAG(orders_count) OVER(ORDER BY YEAR, MONTH) AS prev_order_count FROM
(SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS YEAR, EXTRACT(MONTH FROM
order_purchase_timestamp) AS MONTH, COUNT(*) AS orders_count
FROM `T_company.orders`
WHERE order_status = 'delivered'
GROUP BY 1, 2 ORDER BY 1, 2) AS BASE1 ORDER BY YEAR, MONTH) AS BASE2
```

Insight & Recommendation – Above code represents the month on month sales and growth rate. Also through this we can understand what needs to be changed in marketing campaigns and operational changes.



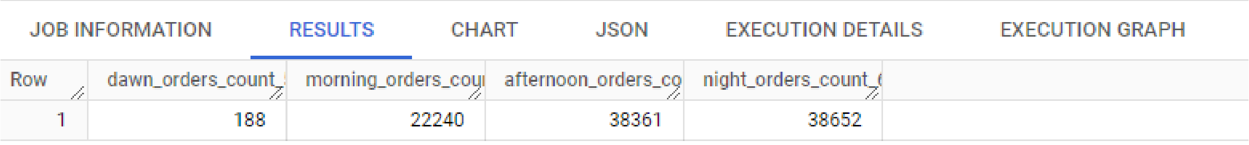
Row	YEAR	MONTH	orders_count	prev_order_count	order_growth_rate_p
1	2016	9	1	null	null
2	2016	10	265	1	26400.0
3	2016	12	1	265	-99.62
4	2017	1	750	1	74900.0
5	2017	2	1653	750	120.4
6	2017	3	2546	1653	54.02
7	2017	4	2303	2546	-9.54
8	2017	5	3546	2303	53.97
9	2017	6	3135	3546	-11.59
10	2017	7	3872	3135	23.51

Q2.2 Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

Q2.3 During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

```
SELECT COUNTIF((TIME(order_purchase_timestamp) >= '05:00:00' AND
TIME(order_purchase_timestamp) <
'06:00:00')) AS dawn_orders_count_5am_6am,
COUNTIF((TIME(order_purchase_timestamp) >= '06:00:00' AND TIME(order_purchase_timestamp) <
'12:00:00')) AS morning_orders_count_6am_12pm,
COUNTIF((TIME(order_purchase_timestamp) >= '12:00:00' AND TIME(order_purchase_timestamp) <
'18:00:00')) AS afternoon_orders_count_12pm_6pm,
COUNTIF((TIME(order_purchase_timestamp) >= '18:00:00' AND TIME(order_purchase_timestamp) <=
'23:59:59') OR (TIME(order_purchase_timestamp) >= '00:00:00' AND
TIME(order_purchase_timestamp) < '05:00:00')) AS night_orders_count_6pm_5am
FROM `T_company.orders`
```

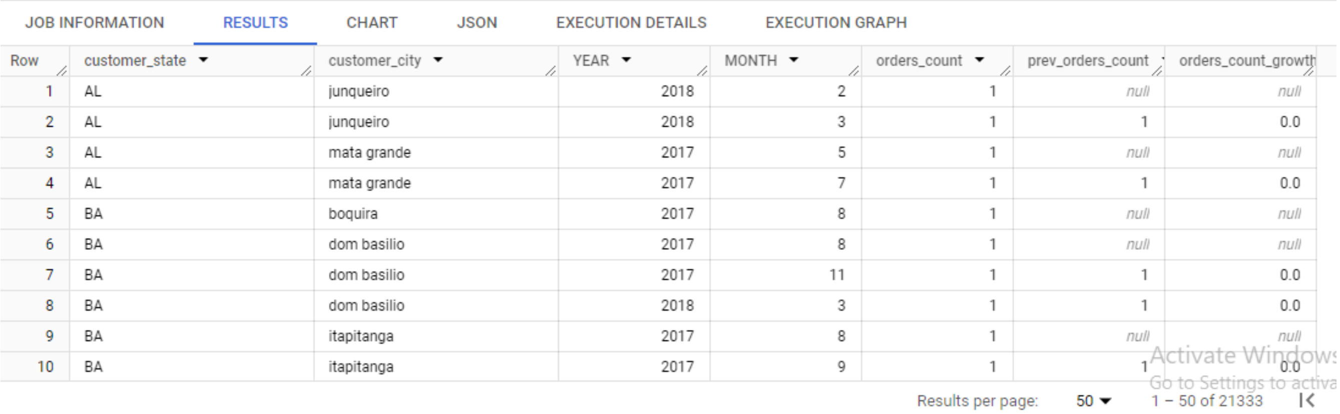
Insight & Recommendation – Above code represents the peak order time when the volume of order is the highest, Also higher no of orders in the morning suggests that customer is mostly active in the morning that relates to his purchasing power and same goes for the customer who purchase at night.



Row	dawn_orders_count_	morning_orders_cou	afternoon_orders_co	night_orders_count_(
1	188	22240	38361	38652

Q3. 1 Get the month on month no. of orders placed in each state.

```
SELECT *, ROUND(((orders_count - prev_orders_count) / prev_orders_count) * 100, 2) AS
orders_count_growth_rate FROM
(SELECT *, LAG(orders_count) OVER(PARTITION BY customer_state, customer_city ORDER BY YEAR,
MONTH) AS prev_orders_count FROM
(SELECT C.customer_state, C.customer_city, BASE1.YEAR, BASE1.MONTH, COUNT(*) AS orders_count
FROM `T_company.customers`  AS C
JOIN
(SELECT *, EXTRACT(MONTH FROM order_purchase_timestamp) AS MONTH, EXTRACT(YEAR FROM
order_purchase_timestamp) AS YEAR FROM `T_company.orders`
WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id GROUP BY 1,2,
3, 4)) AS BASE2
```

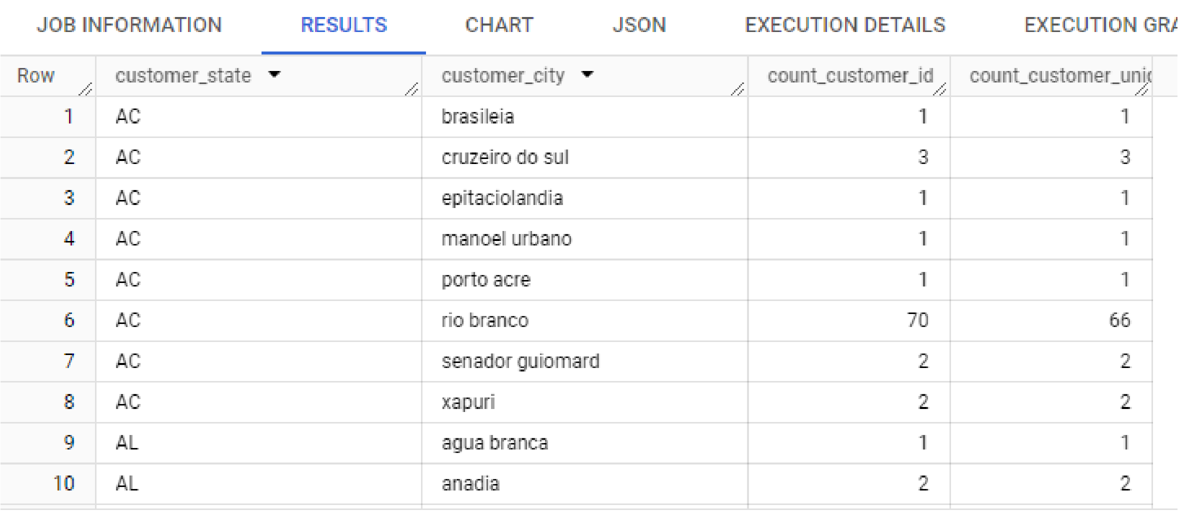


Row	customer_state	customer_city	YEAR	MONTH	orders_count	prev_orders_count	orders_count_growth
1	AL	junqueiro	2018	2	1	*null*	*null*
2	AL	junqueiro	2018	3	1	1	0.0
3	AL	mata grande	2017	5	1	*null*	*null*
4	AL	mata grande	2017	7	1	1	0.0
5	BA	boquira	2017	8	1	*null*	*null*
6	BA	dom basilio	2017	8	1	*null*	*null*
7	BA	dom basilio	2017	11	1	1	0.0
8	BA	dom basilio	2018	3	1	1	0.0
9	BA	itapitanga	2017	8	1	*null*	*null*
10	BA	itapitanga	2017	9	1	1	0.0

Insight & Recommendation – Above code represents the geographical trends in customer behavior, with that the resource allocation will be easy for the business.

```
## Q3. 2 How are the customers distributed across all the states?

SELECT customer_state, customer_city ,
COUNT(DISTINCT customer_id) AS count_customer_id,
COUNT(DISTINCT customer_unique_id) AS count_customer_unique_id
FROM `T_company.customers`
GROUP BY 1, 2
ORDER BY 1, 2
```



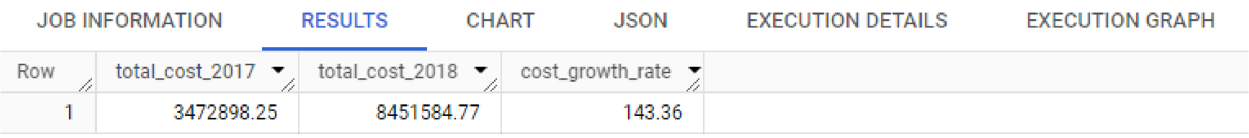
Row	customer_state	customer_city	count_customer_id	count_customer_uniq
1	AC	brasileia	1	1
2	AC	cruzeiro do sul	3	3
3	AC	epitaciolandia	1	1
4	AC	manoel urbano	1	1
5	AC	porto acre	1	1
6	AC	rio branco	70	66
7	AC	senador guiomard	2	2
8	AC	xapuri	2	2
9	AL	agua branca	1	1
10	AL	anadia	2	2

Insight & Recommendation – Above code represents the level of customer engagement with cities and state. Higher count indicates large customer base whereas lower count indicates the area of retention for the organization.

Q4. 1 Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

```
WITH TABLE1 AS
(SELECT ROUND(SUM(price + freight_value), 2) AS total_cost_2017
FROM (SELECT O.*, OI.* FROM `T_company.orders` AS O JOIN `T_company.order_items` AS OI ON
O.order_id = OI.order_id
WHERE O.order_status = 'delivered' AND (EXTRACT(YEAR FROM O.order_purchase_timestamp) = 2017)
AND
EXTRACT(MONTH FROM O.order_purchase_timestamp) BETWEEN 1 AND 8)),
TABLE2 AS (SELECT ROUND(SUM(price + freight_value),2) AS total_cost_2018
FROM (SELECT O.*, OI.* FROM `T_company.orders` AS O JOIN `T_company.order_items` AS OI ON
O.order_id = OI.order_id
WHERE O.order_status = 'delivered' AND (EXTRACT(YEAR FROM O.order_purchase_timestamp) = 2018)
AND
EXTRACT(MONTH FROM O.order_purchase_timestamp) BETWEEN 1 AND 8))
SELECT T1.total_cost_2017, T2.total_cost_2018, ROUND(((T2.total_cost_2018 -
T1.total_cost_2017) / T1.total_cost_2017) * 100, 2) AS cost_growth_rate
FROM TABLE1 AS T1 CROSS JOIN TABLE2 AS T2
```

Insight & Recommendation – Above code represents the overall financial performance of the company. The company can make informed decision to manage cost effectively and drive operational efficiency.

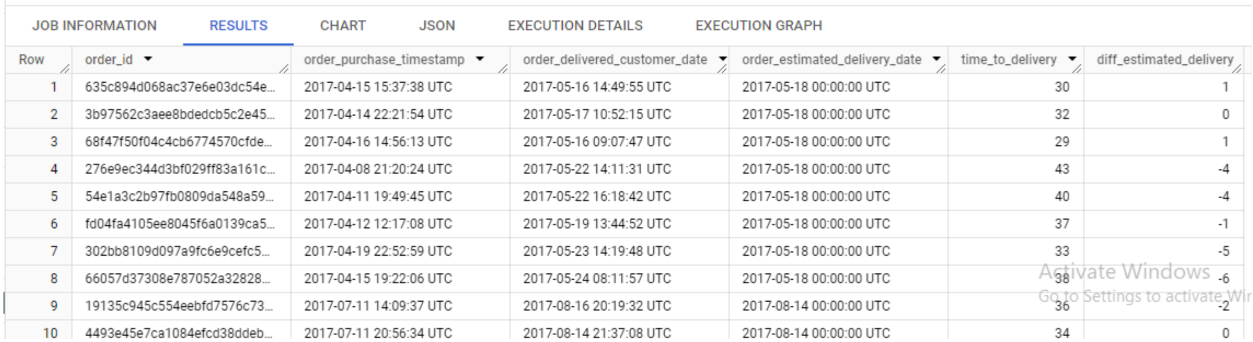


Row	total_cost_2017	total_cost_2018	cost_growth_rate
1	3472898.25	8451584.77	143.36

Q5. 1 Find the no. of days taken to deliver each order from the order’s purchase date as delivery time. Also, calculate the difference (in days) between the estimated & actual delivery date of an order.

```
SELECT order_id, order_purchase_timestamp, order_delivered_customer_date,
order_estimated_delivery_date,
(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) AS
time_to_delivery,
(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
diff_estimated_delivery
FROM `T_company.orders`
WHERE order_status = 'delivered'
```

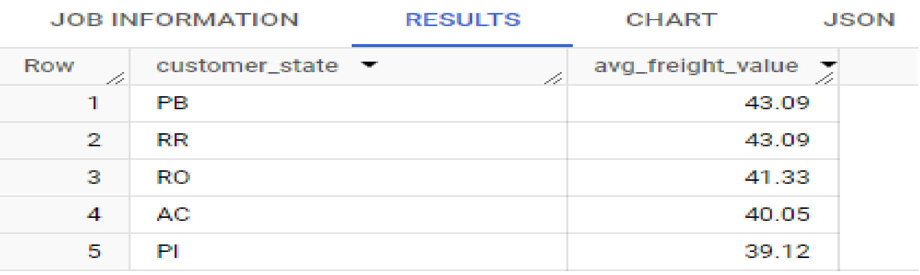
Insight & Recommendation – Analyzing the differences between estimated and actual delivery allows the company to evaluate its delivery performance. Analysis can guide operational improvements to streamline order processing, fulfillment, and shipping processes.



Row	order_id	order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date	time_to_delivery	diff_estimated_delivery
1	635c894d068ac37e6e03dc54e...	2017-04-15 15:37:38 UTC	2017-05-16 14:49:55 UTC	2017-05-18 00:00:00 UTC	30	1
2	3b97562c3aee8bdedcb5c2e45...	2017-04-14 22:21:54 UTC	2017-05-17 10:52:15 UTC	2017-05-18 00:00:00 UTC	32	0
3	68f47f50f04c4cb6774570cfde...	2017-04-16 14:56:13 UTC	2017-05-16 09:07:47 UTC	2017-05-18 00:00:00 UTC	29	1
4	276e9ec344d3bf029ff83a161c...	2017-04-08 21:20:24 UTC	2017-05-22 14:11:31 UTC	2017-05-18 00:00:00 UTC	43	-4
5	54e1a3c2b97fb0809da548a59...	2017-04-11 19:49:45 UTC	2017-05-22 16:18:42 UTC	2017-05-18 00:00:00 UTC	40	-4
6	fd04fa4105ee8045f6a0139ca5...	2017-04-12 12:17:08 UTC	2017-05-19 13:44:52 UTC	2017-05-18 00:00:00 UTC	37	-1
7	302bb8109d097a9fc6e9cefc5...	2017-04-19 22:52:59 UTC	2017-05-23 14:19:48 UTC	2017-05-18 00:00:00 UTC	33	-5
8	66057d37308e787052a32828...	2017-04-15 19:22:06 UTC	2017-05-24 08:11:57 UTC	2017-05-18 00:00:00 UTC	38	-6
9	19135c945c554eebfd7576c73...	2017-07-11 14:09:37 UTC	2017-08-16 20:19:32 UTC	2017-08-14 00:00:00 UTC	36	-2
10	4493e45e7ca1084efcd38ddeb...	2017-07-11 20:56:34 UTC	2017-08-14 21:37:08 UTC	2017-08-14 00:00:00 UTC	34	0

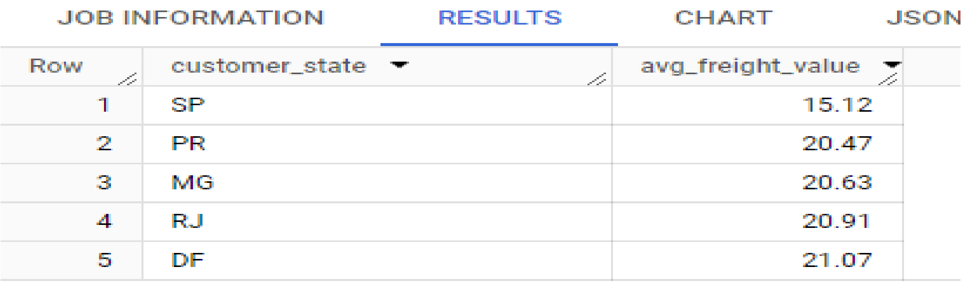
```
##Q5. 2 Find out the top 5 states with the highest & lowest average freight value.

SELECT C.customer_state, ROUND(AVG(OI.freight_value), 2) AS avg_freight_value,
FROM `T_company.customers` AS C
JOIN
(SELECT *,
FROM `T_company.orders`
WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
JOIN
`T_company.order_items` AS OI ON BASE1.order_id = OI.order_id
GROUP BY C.customer_state
ORDER BY AVG(OI.freight_value) DESC LIMIT 5
```



Row	customer_state	avg_freight_value
1	PB	43.09
2	RR	43.09
3	RO	41.33
4	AC	40.05
5	PI	39.12

```
SELECT C.customer_state, ROUND(AVG(OI.freight_value), 2) AS avg_freight_value,
FROM `T_company.customers` AS C
JOIN
(SELECT *,
FROM `T_company.orders`
WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
JOIN
`T_company.order_items` AS OI ON BASE1.order_id = OI.order_id
GROUP BY C.customer_state
ORDER BY AVG(OI.freight_value) ASC
LIMIT 5
```



Row	customer_state	avg_freight_value
1	SP	15.12
2	PR	20.47
3	MG	20.63
4	RJ	20.91
5	DF	21.07

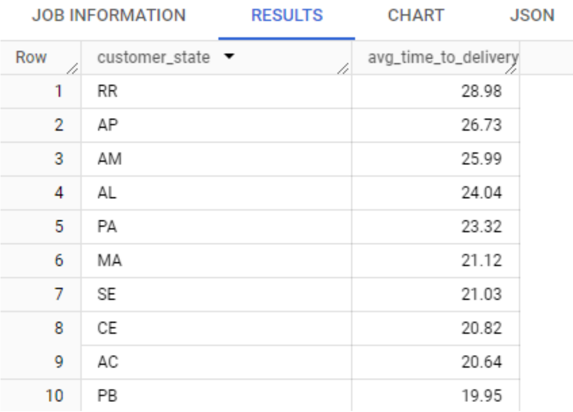
Insight & Recommendation – Higher average freight values in certain states may indicate factors such as distance from distribution centers, logistical challenges, or differences in shipping methods. Providing competitive shipping rates and affordable delivery options can contribute to customer loyalty and retention.

Q5. 3 Find out the top 5 states with the highest & lowest average delivery time.

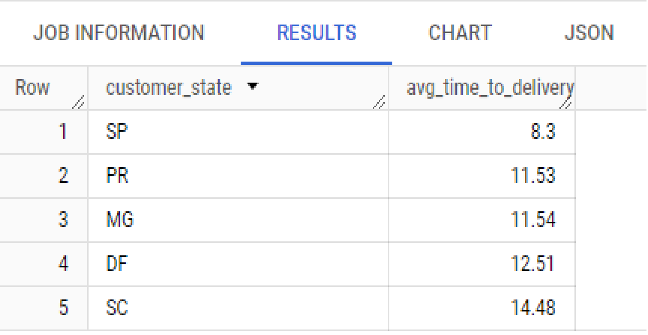
```
SELECT C.customer_state, ROUND(AVG(BASE1.time_to_delivery), 2) AS avg_time_to_delivery,
FROM `T_company.customers` AS C
JOIN
(SELECT *,
TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
time_to_delivery,
FROM `T_company.orders`
WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
GROUP BY C.customer_state
ORDER BY AVG(BASE1.time_to_delivery) DESC
```

```
SELECT C.customer_state, ROUND(AVG(BASE1.time_to_delivery), 2) AS avg_time_to_delivery,
FROM `T_company.customers` AS C
JOIN
(SELECT *,
TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
time_to_delivery
FROM `T_company.orders`
WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
GROUP BY C.customer_state
ORDER BY AVG(BASE1.time_to_delivery) ASC
LIMIT 5
```

Insight & Recommendation - States with longer average times to delivery may experience lower levels of customer satisfaction due to delays in receiving orders. Above code provide insights into the average time it takes for orders to be delivered to customers in each state.



Row	customer_state	avg_time_to_delivery
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32
6	MA	21.12
7	SE	21.03
8	CE	20.82
9	AC	20.64
10	PB	19.95

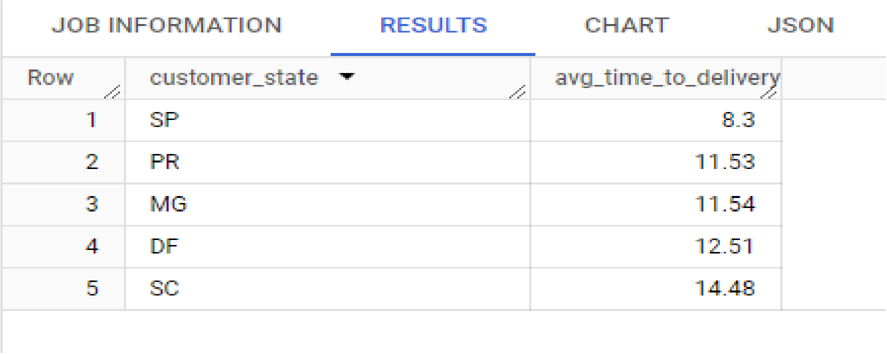


Row	customer_state	avg_time_to_delivery
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48

Q5. 4 Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
SELECT C.customer_state, ROUND(AVG(BASE1.diff_estdel_actdel), 2) AS avg_daydiff_estdel_actdel
FROM `T_company.customers` AS C
JOIN
(SELECT *,
TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS
diff_estdel_actdel
FROM `T_company.orders`
WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
GROUP BY C.customer_state
ORDER BY AVG(BASE1.diff_estdel_actdel) DESC
LIMIT 5
```

Insight & Recommendation - The average difference between the estimated delivery date and the actual delivery date provides insights into the accuracy of delivery estimates provided to customers.



Row	customer_state	avg_time_to_delivery
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48

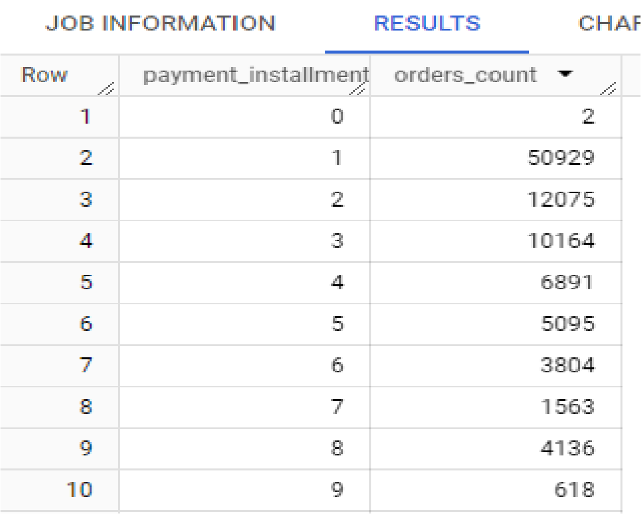
Q6. 1 Find the month on month no. of orders placed using different payment types

Q6. 2 Find the no. of orders placed on the basis of the payment installments that have been paid.

```
SELECT P.payment_installments, COUNT(*) AS orders_count
```

```
FROM `T_company.payments` AS P
JOIN
(SELECT *
FROM `T_company.orders`
WHERE order_status = 'delivered') AS O ON P.order_id = O.order_id
group by 1
```

Insight & Recommendation - Analyzing the count of orders for each payment installment option reveals which payment plans are most commonly utilized by customers. Knowledge of popular payment installment options allows the company to tailor marketing campaigns and promotions to incentivize certain payment methods.



Row	payment_installment	orders_count
1	0	2
2	1	50929
3	2	12075
4	3	10164
5	4	6891
6	5	5095
7	6	3804
8	7	1563
9	8	4136
10	9	618