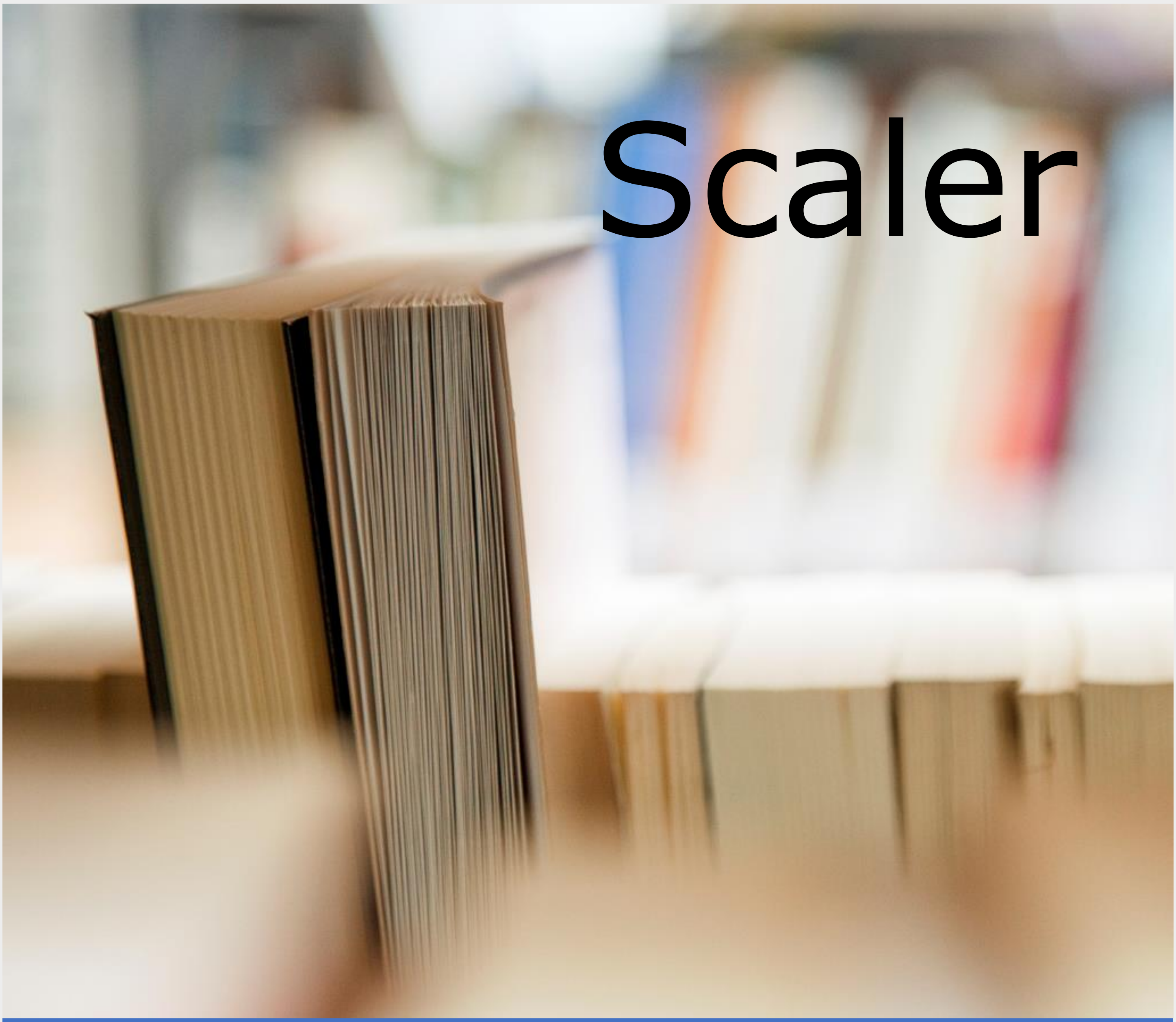


Business Case: Target SQL

Queries and Insights

10/11/23

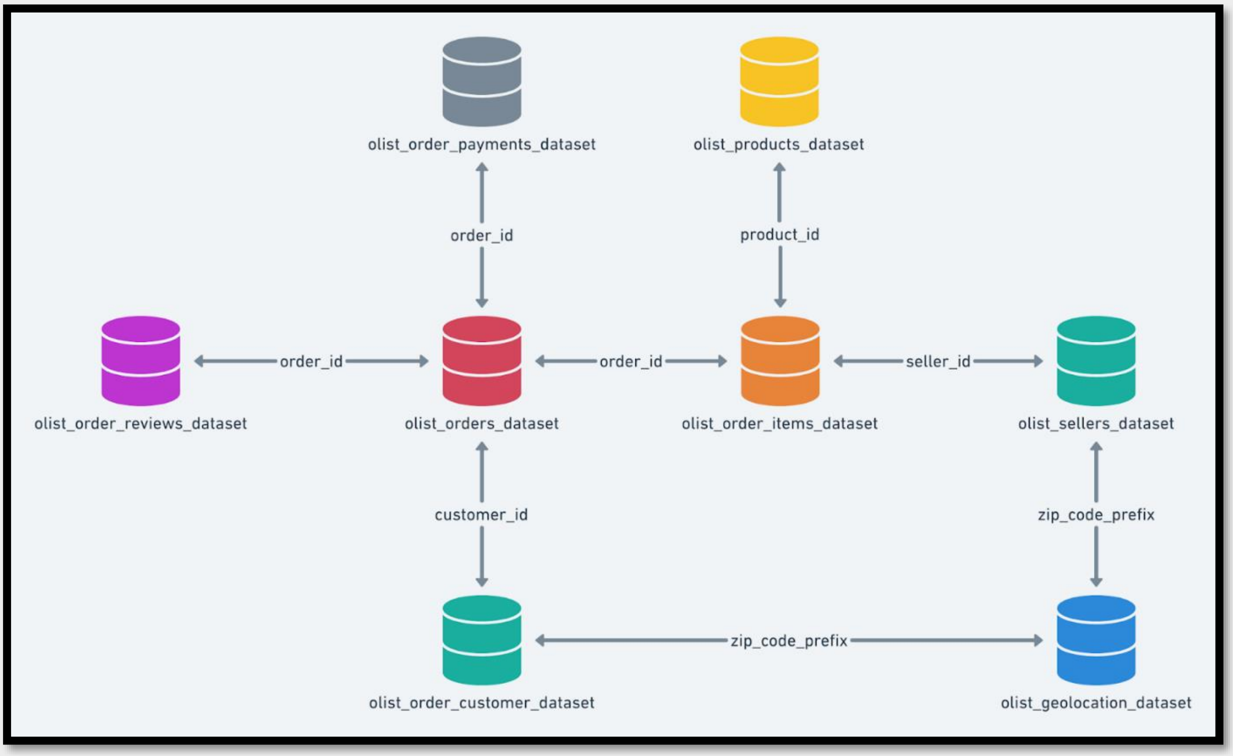
SQL

Business Case: Target SQL

Contents

Business case: Target

ER Diagram/Schema



Problem Statement

Assuming you are a data analyst/ scientist at Target, you have been assigned the task of analyzing the given dataset to extract valuable insights and provide actionable recommendations.

Business Case: Target SQL

Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

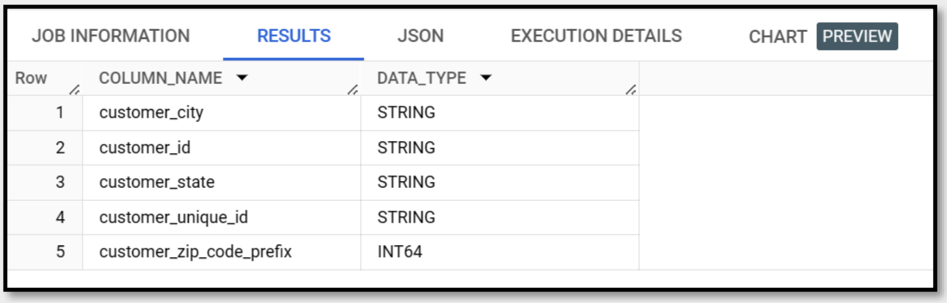
Question 1a

Data type of all columns in the **customer's** table.

Query

```
select COLUMN_NAME, DATA_TYPE
from scaler-dsml-sql-396914.Target_SQL_Business_Case.INFORMATION_SCHEMA.COLUMNS
where table_name = 'customers'
order by 1
```

Output Screenshot



Row	COLUMN_NAME	DATA_TYPE
1	customer_city	STRING
2	customer_id	STRING
3	customer_state	STRING
4	customer_unique_id	STRING
5	customer_zip_code_prefix	INT64

Insights

Therefore, from the output we can conclude that 4 columns have **String** datatype and 1 column with **Integer**.

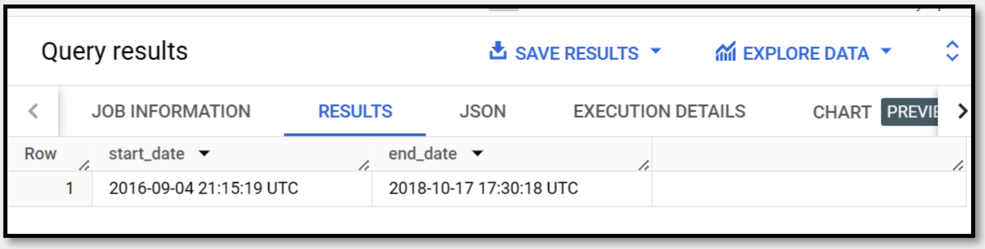
Question 1b

Get the time range between which the orders were placed.

Query

```
select min(order_purchase_timestamp) as start_date,
max(order_purchase_timestamp) as end_date
from `Target_SQL_Business_Case.orders`
```

Output Screenshot



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

Business Case: Target SQL

Insights

As a conclusion, Order purchase was started on **4th of Sep 2016** and ended on **17th of Oct 2018**, along with the start time **21:15:19** and end time **17:30:18** as per UTC time zone.

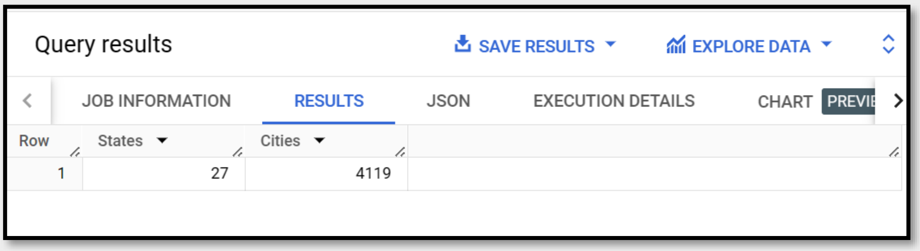
Question 1C

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

Query

```
select count(distinct c.customer_state) as States,
       count(distinct c.customer_city) as Cities
from `Target_SQL_Business_Case.customers` c
inner join `Target_SQL_Business_Case.orders` o
on o.customer_id = c.customer_id
```

Output Screenshot



Row	States	Cities
1	27	4119

Insights

As per the output, we have unique **27** states and **4119** cities that the customers have ordered in the particular period

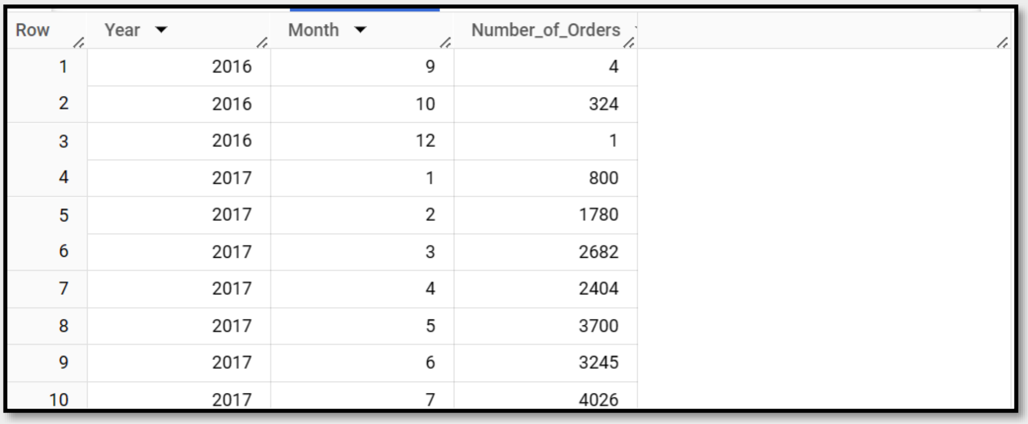
Question 2A

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

Query

```
select
extract(year from order_purchase_timestamp) as Year,
extract(month from order_purchase_timestamp) as Month,
count(*) as Number_of_Orders
from `Target_SQL_Business_Case.orders`
group by 1,2
order by 1,2
```

Output Screenshot



Row	Year	Month	Number_of_Orders
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
9	2017	6	3245
10	2017	7	4026

Insights

- The report provides a monthly breakdown of order volumes over the years, allowing for a clear understanding of seasonal trends and customer demand patterns.
- And also, By analyzing this data, businesses can identify peak months, plan inventory management more effectively, optimize marketing strategies, and anticipate customer needs.

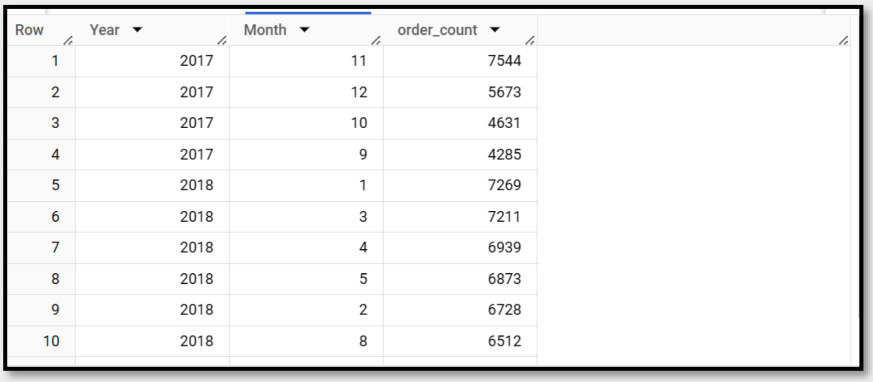
Question 2B

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

Query

```
with Monthly_orders as (
select
extract(year from order_purchase_timestamp) as Year,
extract(month from order_purchase_timestamp) as Month,
count(*) as order_count
from `Target_SQL_Business_Case.orders`
group by 1,2
order by 1,2)

select Year, Month, order_count
from
(select Year, Month, order_count,
dense_rank() over(partition by Month order by order_count desc) as RNK
from Monthly_orders)
where RNK =1
order by 1,3 desc
```

Output Screenshot



Row	Year	Month	order_count
1	2017	11	7544
2	2017	12	5673
3	2017	10	4631
4	2017	9	4285
5	2018	1	7269
6	2018	3	7211
7	2018	4	6939
8	2018	5	6873
9	2018	2	6728
10	2018	8	6512

Insights

- As per the output, the number of counts varies as per the month and year. We can infer that there was a month with a significantly higher number of orders than other months is 11th month of 2017 which has **7544** orders.
- This could be due to various reasons such as a **Sale** or **Promotion**, a **New Product Launch**, or **Seasonal Demand**.

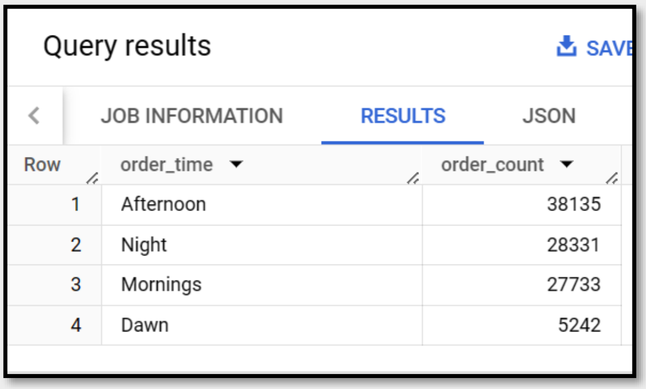
Question 2C

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

Query

```sql
select
case
when extract(hour from order_purchase_timestamp) between 0 and 6 then 'Dawn'
when extract(hour from order_purchase_timestamp) between 7 and 12 then 'Mornings'
when extract(hour from order_purchase_timestamp) between 13 and 18 then 'Afternoon'
when extract(hour from order_purchase_timestamp) between 19 and 23 then 'Night' end
as order_time, count(*) as order_count
from `Target_SQL_Business_Case.orders`
group by 1
order by 2 desc
```

Output Screenshot



Row	order_time	order_count
1	Afternoon	38135
2	Night	28331
3	Mornings	27733
4	Dawn	5242

Business Case: Target SQL

Insights

The majority of orders are placed during the **Afternoon** time period indicating a peak in customer activity. The various reasons such as people placing orders after their lunch time as a break, or taking advantage of early bird discounts, or maybe most of the sales starts at afternoon.

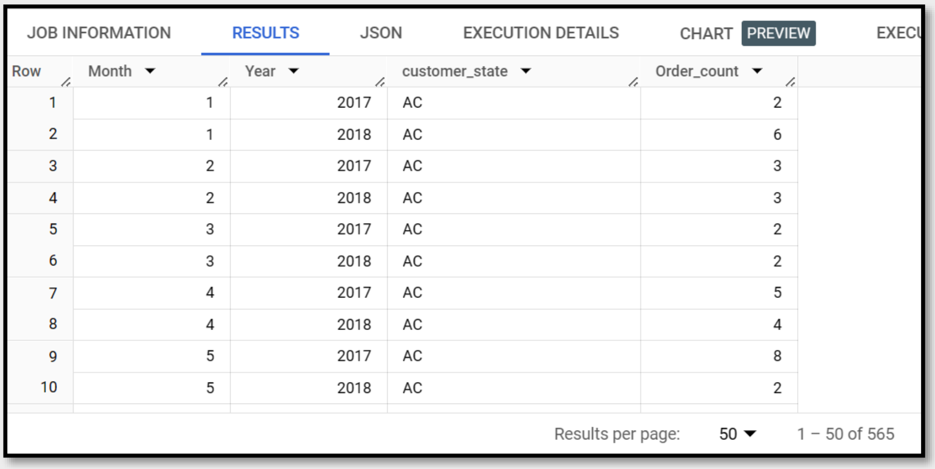
Question 3A

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

Query

```
select
Extract(month from order_purchase_timestamp) as Month,
Extract(year from order_purchase_timestamp) as Year, C.customer_state, count(*) as
Order_count
from `Target_SQL_Business_Case.customers` C
inner join `Target_SQL_Business_Case.orders` O
on O.customer_id = C.customer_id
group by 1,2,3
order by 3,1,2
```

Output Screenshot



Row	Month	Year	customer_state	Order_count
1	1	2017	AC	2
2	1	2018	AC	6
3	2	2017	AC	3
4	2	2018	AC	3
5	3	2017	AC	2
6	3	2018	AC	2
7	4	2017	AC	5
8	4	2018	AC	4
9	5	2017	AC	8
10	5	2018	AC	2

Results per page: 50 1 – 50 of 565

Insights

- The output provides a detailed breakdown of orders placed by customers, categorized by month, year, and customer state. By examining this data, patterns in customer purchasing behaviour across different states and time periods can be identified.
- This insight can be used to offers valuable information for forecasting demand and planning promotions, ensuring a more targeted and efficient approach to customer engagement and sales.

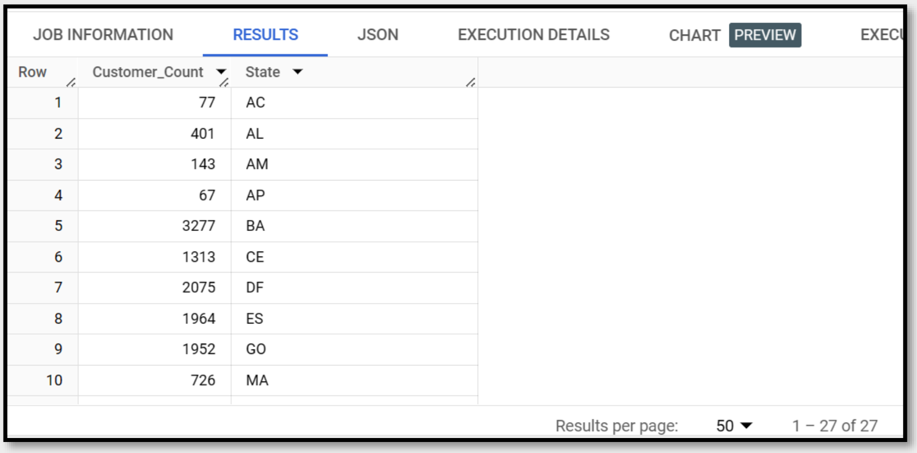
Question 3B

How are the customers distributed across all the states?

Query

```
select count(distinct customer_unique_id) as Customer_Count, customer_state as
State
from `Target_SQL_Business_Case.customers`
group by 2
order by 2
```

Output Screenshot



Row	Customer_Count	State
1	77	AC
2	401	AL
3	143	AM
4	67	AP
5	3277	BA
6	1313	CE
7	2075	DF
8	1964	ES
9	1952	GO
10	726	MA

Results per page: 50 1 – 27 of 27

Insights

- As per the output it provides a count of unique customers for each state, indicating the distribution of customers across different regions.
- This insight highlights the importance of focusing marketing campaigns and resources based on the density of customers in specific states, ensuring a more personalized and effective approach to customer engagement.

Question 4A

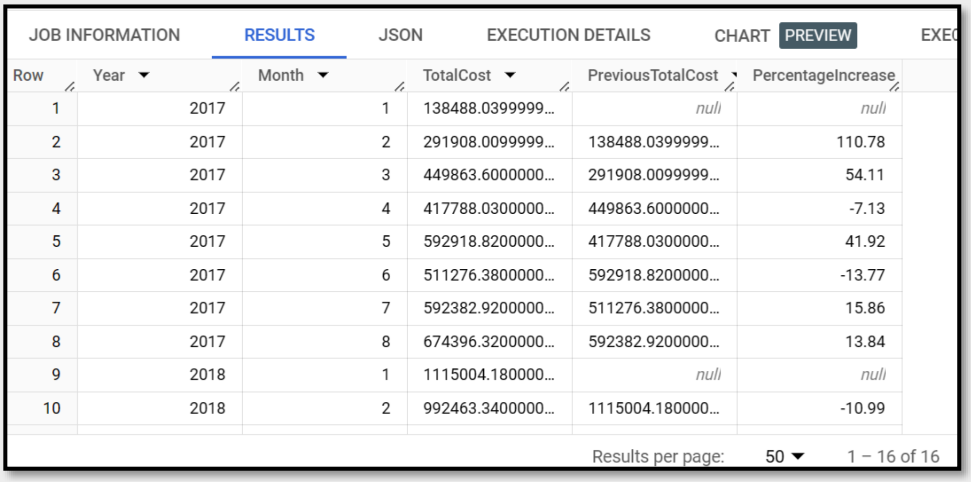
Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only

Query

```
With OrderCosts as (
  Select
    Extract(month from order_purchase_timestamp) as Month,
    Extract(year from order_purchase_timestamp) as Year,
    P.payment_value
  from
    `Target_SQL_Business_Case.payments` P
  Inner join
```

```
    `Target_SQL_Business_Case.orders` O
  On
    P.order_id = O.order_id
  Where
    Extract(year from order_purchase_timestamp) between 2017 and 2018
    and Extract(month from order_purchase_timestamp) between 1 and 8
)

Select
  Year,
  Month,
  Sum(payment_value) As TotalCost,
  Lag(Sum(payment_value), 1) over (Partition by Year Order by Month) AS
PreviousTotalCost,
  round(((sum(payment_value) - Lag(Sum(payment_value), 1) over (Partition by Year
Order by Month)) / Lag(Sum(payment_value), 1) over (Partition by Year Order by
Month)) * 100, 2) AS PercentageIncrease
From
  OrderCosts
Group by
  Year, Month
Order by
  Year, Month;
```

Output Screenshot



Row	Year	Month	TotalCost	PreviousTotalCost	PercentageIncrease
1	2017	1	138488.0399999...	null	null
2	2017	2	291908.0099999...	138488.0399999...	110.78
3	2017	3	449863.6000000...	291908.0099999...	54.11
4	2017	4	417788.0300000...	449863.6000000...	-7.13
5	2017	5	592918.8200000...	417788.0300000...	41.92
6	2017	6	511276.3800000...	592918.8200000...	-13.77
7	2017	7	592382.9200000...	511276.3800000...	15.86
8	2017	8	674396.3200000...	592382.9200000...	13.84
9	2018	1	1115004.180000...	null	null
10	2018	2	992463.3400000...	1115004.180000...	-10.99

Results per page: 50 1 – 16 of 16

Insights

Here, in this output we can identify the monthly percentage patterns over the period from January 2017 to August 2018.

- Positive percentage increases signify growing sales, potentially indicating successful marketing campaigns, enhanced customer engagement, or expanding product offerings.
- Negative percentage increases might indicate seasonal fluctuations, market challenges, or changing customer behaviour.

Analysing these trends enables businesses to identify successful strategies, adjust marketing and sales efforts accordingly and customer satisfaction.

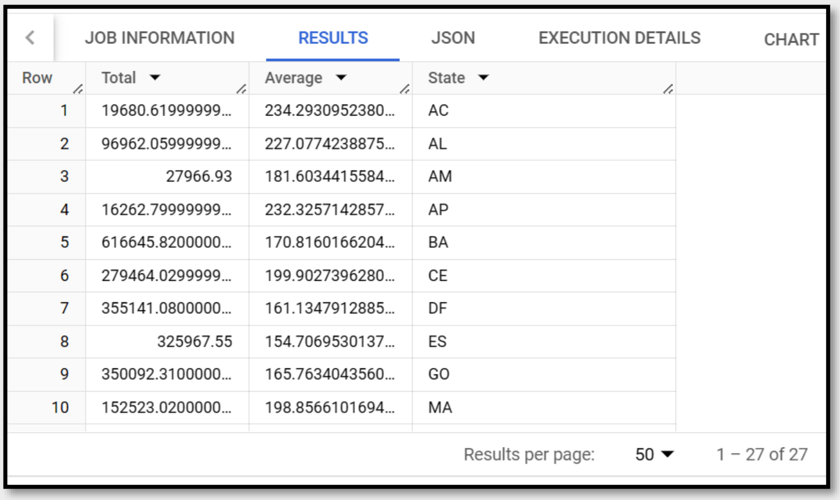
Question 4B

Calculate the Total & Average value of order price for each state.

Query

```
select sum(P.payment_value) as Total,
       avg(payment_value) as Average,
       C.customer_state as State

from `Target_SQL_Business_Case.payments` P
inner join `Target_SQL_Business_Case.orders` O
on P.order_id = O.order_id

inner join `Target_SQL_Business_Case.customers` C
on C.customer_id = O.customer_id
group by 3
order by 3
```

Output Screenshot



Row	Total	Average	State
1	19680.61999999...	234.2930952380...	AC
2	96962.05999999...	227.0774238875...	AL
3	27966.93	181.6034415584...	AM
4	16262.79999999...	232.3257142857...	AP
5	616645.8200000...	170.8160166204...	BA
6	279464.0299999...	199.9027396280...	CE
7	355141.0800000...	161.1347912885...	DF
8	325967.55	154.7069530137...	ES
9	350092.3100000...	165.7634043560...	GO
10	152523.0200000...	198.8566101694...	MA

Results per page: 50 ▾ 1 – 27 of 27

Insights

- The report summarizes the total and average payment values for each state, providing valuable information into regional customer spending patterns.
- We can identify states where customers tend to make larger or more frequent purchases, allowing for targeted marketing campaigns and product offerings.

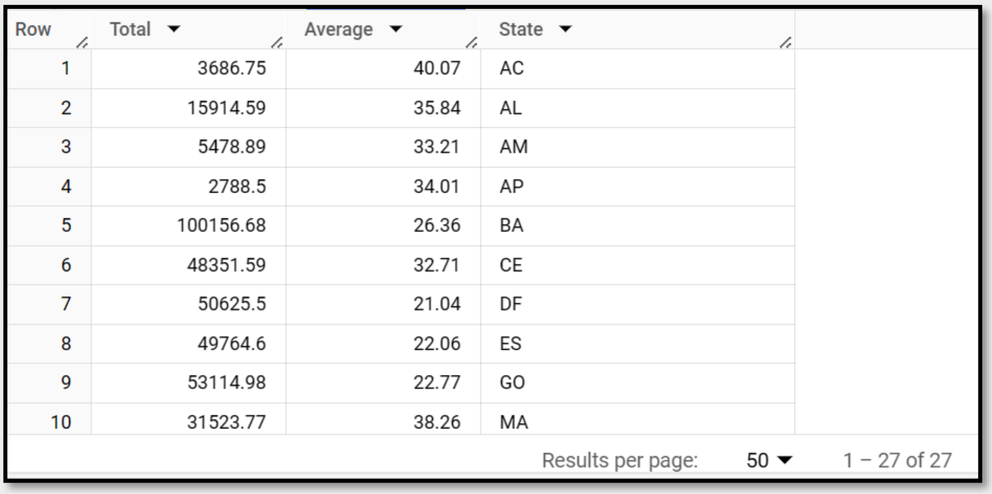
Question 4c

Calculate the Total & Average value of order freight for each state.

Query

```
select round(Sum(OI.freight_value), 2) as Total,
       round(Avg(OI.freight_value), 2) as Average,
       C.customer_state as State
from `Target_SQL_Business_Case.customers` C
inner join `Target_SQL_Business_Case.orders` O
on C.customer_id = O.customer_id
inner join `Target_SQL_Business_Case.order_items` OI
on OI.order_id = O.order_id
group by 3
order by 3
```

Output Screenshot



Row	Total	Average	State
1	3686.75	40.07	AC
2	15914.59	35.84	AL
3	5478.89	33.21	AM
4	2788.5	34.01	AP
5	100156.68	26.36	BA
6	48351.59	32.71	CE
7	50625.5	21.04	DF
8	49764.6	22.06	ES
9	53114.98	22.77	GO
10	31523.77	38.26	MA

Results per page: 50 ▼ 1 – 27 of 27

Insights

As per the output it shows the total and average freight costs incurred by customers in each state.

For states with higher average freight costs, optimizing logistics and delivery routes could lead to cost savings. The States with lower average freight costs might indicate time to distribution centres or efficient logistics operations.

Question 5A

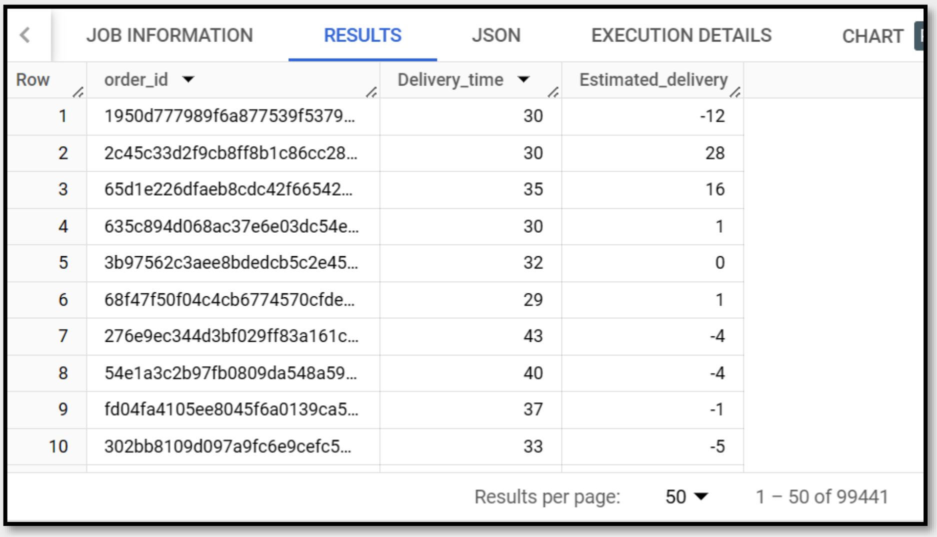
Find the no. of days taken to deliver each order from the order's purchase date as delivery time.

Also, calculate the difference (in days) between the estimated & actual delivery date of an order.

Query

```
  select order_id,
  date_diff(order_delivered_customer_date, order_purchase_timestamp, day) as
Delivery_time,
  date_diff(order_estimated_delivery_date, order_delivered_customer_date, day) as
Estimated_delivery
  from `Target_SQL_Business_Case.orders`
```

Output Screenshot



Row	order_id	Delivery_time	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

Insights

From this output we can understand that how quickly orders are being delivered and whether they are arriving earlier or later than the estimated delivery dates. It can provide insights into their delivery efficiency.

Question 5B

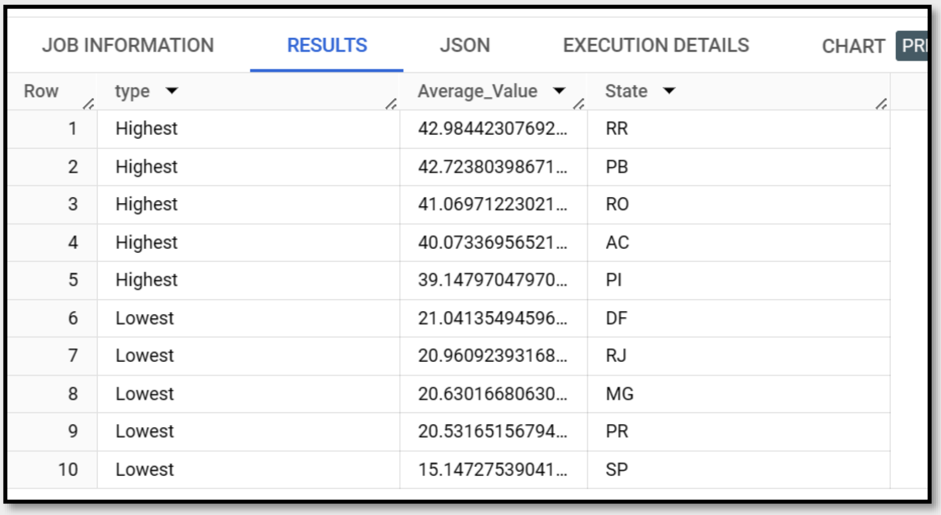
Find out the top 5 states with the highest & lowest average freight value

Query

```
with Final as (select avg(OI.freight_value) Average_Value,
 dense_rank() over(order by avg(OI.freight_value) desc) as RankHigh,
 dense_rank() over(order by avg(OI.freight_value) ) as RankLow,
C.customer_state as State
from `Target_SQL_Business_Case.customers` C
inner join `Target_SQL_Business_Case.orders` O
on C.customer_id = O.customer_id
inner join `Target_SQL_Business_Case.order_items` OI
on OI.order_id = O.order_id
group by 4)

select 'Highest' as type, Average_Value, State
from Final
where RankHigh <= 5
```

```
UNION ALL
select 'Lowest' as type, Average_Value, State
from Final
where RankLow <= 5

Order by 2 desc
```

Output Screenshot



Row	type	Average_Value	State
1	Highest	42.98442307692...	RR
2	Highest	42.72380398671...	PB
3	Highest	41.06971223021...	RO
4	Highest	40.07336956521...	AC
5	Highest	39.14797047970...	PI
6	Lowest	21.04135494596...	DF
7	Lowest	20.96092393168...	RJ
8	Lowest	20.63016680630...	MG
9	Lowest	20.53165156794...	PR
10	Lowest	15.14727539041...	SP

Insights

From this output we can say that highest freight value is 42.98 for RR state and lowest freight value is 15.14 for SP state.

Question 5C

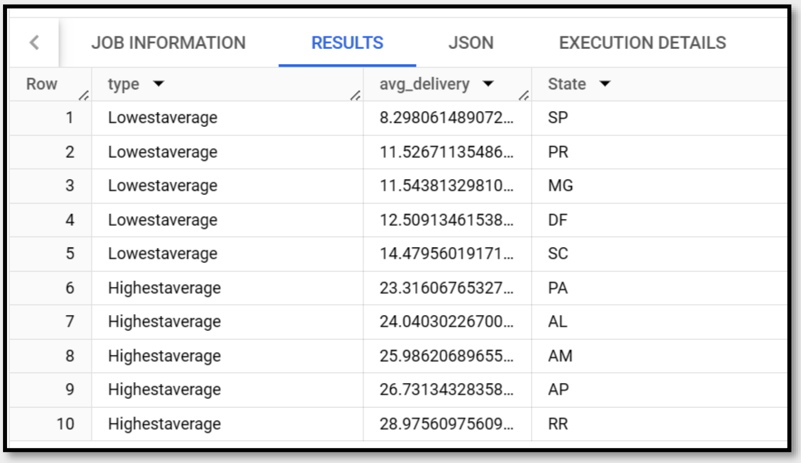
Find out the top 5 states with the highest & lowest average delivery time

Query

```
with final as (select O.order_id,
  date_diff(order_delivered_customer_date, order_purchase_timestamp,
day)delivery_time,
  C.customer_state as State
from `Target_SQL_Business_Case.customers` C
inner join `Target_SQL_Business_Case.orders` O
on C.customer_id = O.customer_id),

value as (select avg(delivery_time) as avg_delivery,
         dense_rank() over(order by avg(delivery_time) desc) as high,
         dense_rank() over(order by avg(delivery_time)) as low, State
         from final
         group by 4)

select "Highestaverage" as type, avg_delivery, State
from value
where high <= 5
```

```
union all
select "Lowestaverage" as type, avg_delivery, State
from value
where low <= 5

order by 2
```

Output Screenshot



Row	type	avg_delivery	State
1	Lowestaverage	8.298061489072...	SP
2	Lowestaverage	11.52671135486...	PR
3	Lowestaverage	11.54381329810...	MG
4	Lowestaverage	12.50913461538...	DF
5	Lowestaverage	14.47956019171...	SC
6	Highestaverage	23.31606765327...	PA
7	Highestaverage	24.04030226700...	AL
8	Highestaverage	25.98620689655...	AM
9	Highestaverage	26.73134328358...	AP
10	Highestaverage	28.97560975609...	RR

Insights

From this output we can understand that how quickly orders are being delivered. Basically, we can find the delivery efficiency from this output.

Here the State SP has the average lowest delivery time. Using this insight, we can analyze where the time lag is happening.

Question 5D

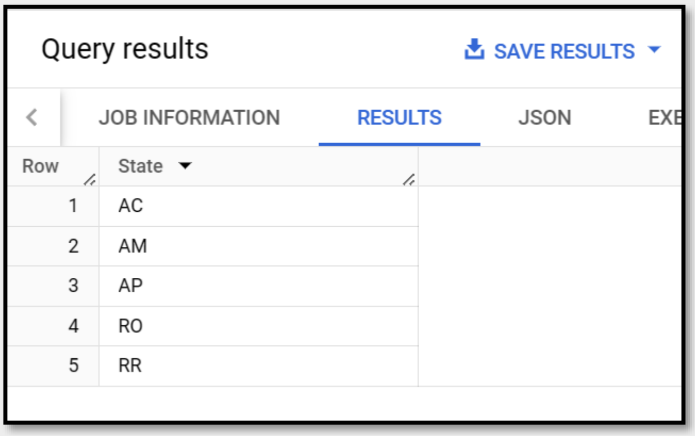
Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

Query

```
with final as (select O.order_id,
  date_diff(order_estimated_delivery_date, order_delivered_customer_date, day) as
Estimated_delivery,
  C.customer_state as State
from `Target_SQL_Business_Case.customers` C
inner join `Target_SQL_Business_Case.orders` O
on C.customer_id = O.customer_id),

value as (select round(avg(Estimated_delivery),2) as Avg_Estimated_delivery,
         dense_rank() over(order by avg(Estimated_delivery) desc) as High,
           State
                 from final
                 group by 3)
```

```
select  State
from value
where High <= 5
order by 1
```

Output Screenshot



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

Insights

The output gives us the top 5 states to deliver the order before the estimated delivery time.

With this insight it is transparent that the customer satisfaction will be enhanced but also serves as a benchmark for optimizing logistics and delivery processes in other areas.

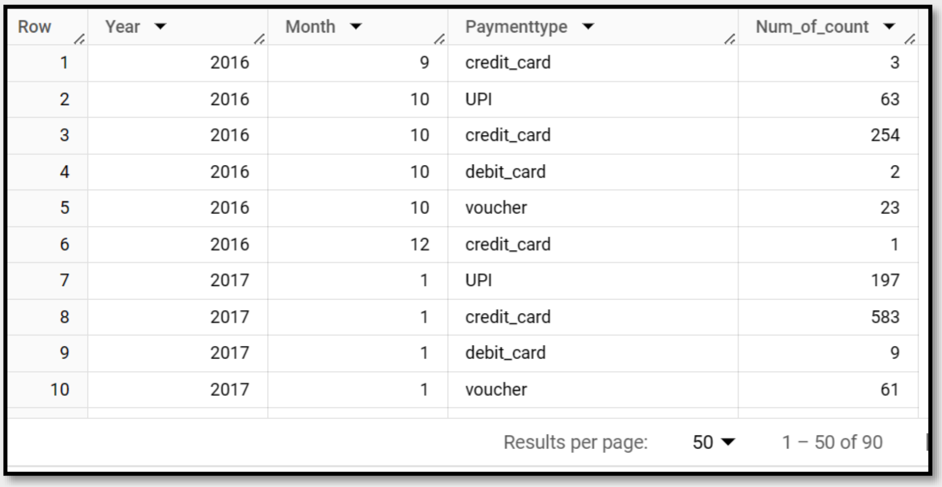
Question 6A

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

Query

```
with final as (select  Extract(year from order_purchase_timestamp) as Year,
Extract(month from order_purchase_timestamp) as Month, P.payment_type as
Paymenttype, count(*) as order_count
from `Target_SQL_Business_Case.orders` O
inner join `Target_SQL_Business_Case.payments` P
on O.order_id = P.order_id
group by 1, 2, 3
order by 1,2,3)

select Year, Month, Paymenttype, sum(order_count) as Num_of_count
from final
group by 1,2,3
order by 1,2,3
```

Output Screenshot



Row	Year	Month	Paymenttype	Num_of_count
1	2016	9	credit_card	3
2	2016	10	UPI	63
3	2016	10	credit_card	254
4	2016	10	debit_card	2
5	2016	10	voucher	23
6	2016	12	credit_card	1
7	2017	1	UPI	197
8	2017	1	credit_card	583
9	2017	1	debit_card	9
10	2017	1	voucher	61

Results per page: 50 1 – 50 of 90

Insights

With the above output most of the customers has used Credit Crad as their payment method.

With this insight we can conclude that in which month the customer has used which type of payment mode. For example, towards the year end,

and starting of the new year he might have used credit card due to festivals and other celebrations. Then later he might have used debitcard or EMI.

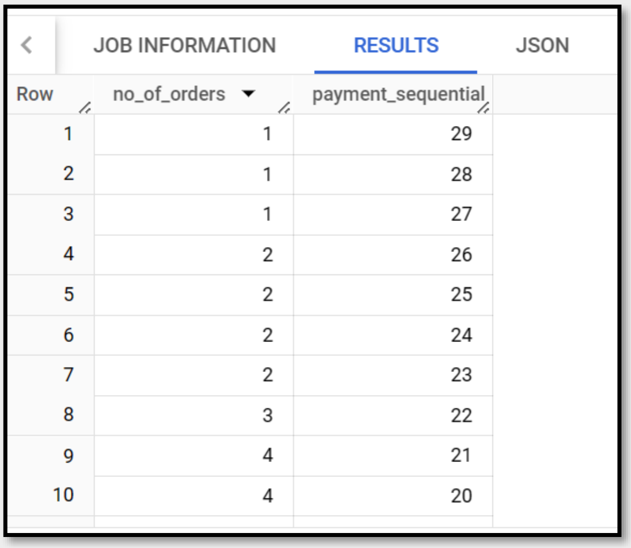
Question 6B

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

Query

```
select count(distinct order_id) as no_of_orders, payment_sequential
from `Target_SQL_Business_Case.payments`
where payment_sequential >= 1
group by 2
order by 2 desc
```

Output Screenshot



Row	no_of_orders	payment_sequential
1	1	29
2	1	28
3	1	27
4	2	26
5	2	25
6	2	24
7	2	23
8	3	22
9	4	21
10	4	20

Insights

Here with this output, it is useful that the how many orders are being placed using this instalment method.