Business Case: Retail Case Study SQL
Problem Statement:
Assuming I am a data analyst/ scientist at Retail store and have been assigned the task of analyzing the given dataset to extract valuable insights and provide actionable recommendations.

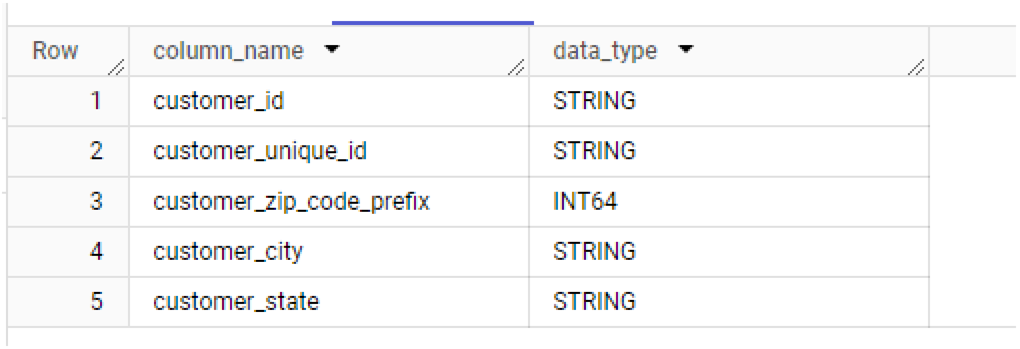
Assumptions:

- The primary assumption about shopping at Retail store is that it is done from both offline (stores) and online.
- This assumption is made because the schema contains elements suggesting online shopping.

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

1. Data type of all columns in the "customers" table.
2. Get the time range between which the orders were placed.
3. Count the Cities & States of customers who ordered during the given period.

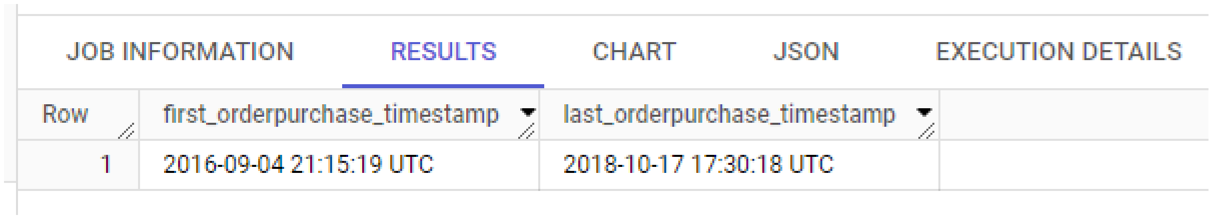
Ans 1.1
Query :-
```
select column_name, data_type
from`Target_SQL.INFORMATION_SCHEMA.COLUMNS`
where table_name = 'customers'
```
Screenshot of output:-



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

Ans 1.2
Query :-
```
select min(order_purchase_timestamp) as first_orderpurchase_timestamp,
max(order_purchase_timestamp) as last_orderpurchase_timestamp
from `Target_SQL.orders`
```

Screenshot of output:-



JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS

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

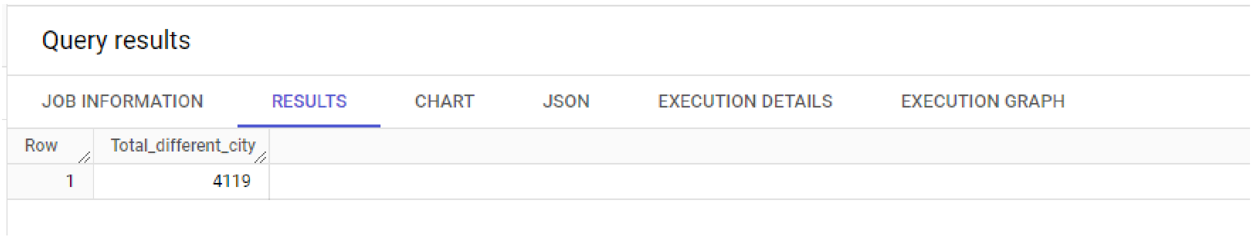
Insights :- First Order was purchased on 4th September 2016 and Last order was purchased on 17th October 2018.

Recommendations :- NA

Ans 1.3

Query :-
```
select count (distinct (customer_city)) as Total_different_city
from `Target_SQL.orders` as O
join
`Target_SQL.customers` as C
on O.customer_id = C.customer_id

select count (distinct (customer_state)) as Total_different_state
from `Target_SQL.customers`
```

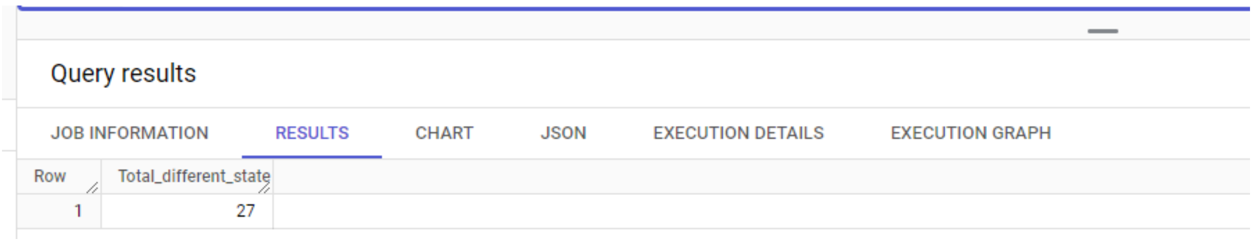
Screenshot of output:-



Query results

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRAPH

Row	Total_different_city
1	4119



Query results

JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS EXECUTION GRAPH

Row	Total_different_state
1	27

Insights :- There are total 4119 distinct cities where orders were placed by the customers.
Recommendations :- NA

Q- 2 In-depth Exploration:

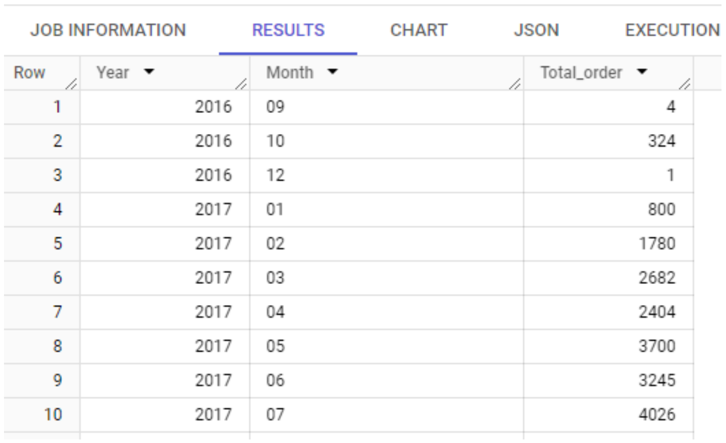
1. Is there a growing trend in the no. of orders placed over the past years?
2. Can we see some kind of monthly seasonality in terms of the no. of orders being placed?
3. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)
    - 0-6 hrs : Dawn
    - 7-12 hrs : Mornings
    - 13-18 hrs : Afternoon
    - 19-23 hrs : Night

Ans 2.1 2.2

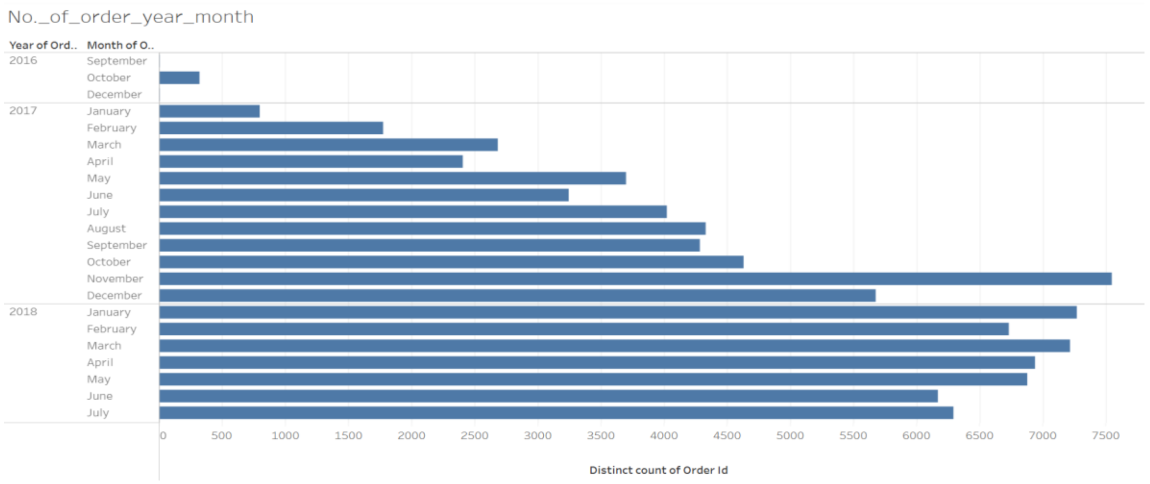
Query :-

```
select Year,Month,count(order_id) as Total_order
from
(select order_id,order_purchase_timestamp,
EXTRACT(year from order_purchase_timestamp) AS Year,
format_date('%m',order_purchase_timestamp) as Month,
from `Target_SQL.orders`
order by order_purchase_timestamp) as tbh
group by Year,Month
order by Year,Month

---Alternative method using CTE

WITH MonthlyOrders AS (
SELECT EXTRACT(year FROM order_purchase_timestamp) AS Year,
format_date('%m', order_purchase_timestamp) AS Month,
order_id
FROM `Target_SQL.orders`
)
SELECT Year Month, COUNT(order_id) AS Total_order
FROM MonthlyOrders
GROUP BY Year,Month
ORDER BY Year,Month;
```

Screenshot of output:-



Row	Year	Month	Total_order
1	2016	09	4
2	2016	10	324
3	2016	12	1
4	2017	01	800
5	2017	02	1780
6	2017	03	2682
7	2017	04	2404
8	2017	05	3700
9	2017	06	3245
10	2017	07	4026



Insights :-

- Sales were continuously growing from the February month of 2017 to November of 2017, but a slice dip was observed in the month of December of 2017.
- Whereas sales were almost stagnant during the January -May of 2018. But sales in this period were among the highest for the company.
- In the month of September of 2018 sales dipped significantly.
- Peaks were observed in the month of Nov(2017), Jan and March (2018)
  Mostly these peaks were observed due to festive season.

Recommendations :-

- Hire more people in the month of November – March.
- Give combo offers with low selling products in the month of November-March for stock clearance.
- Maintain stock inventory for next year for these peak months Nov(2017), Jan and March (2018)

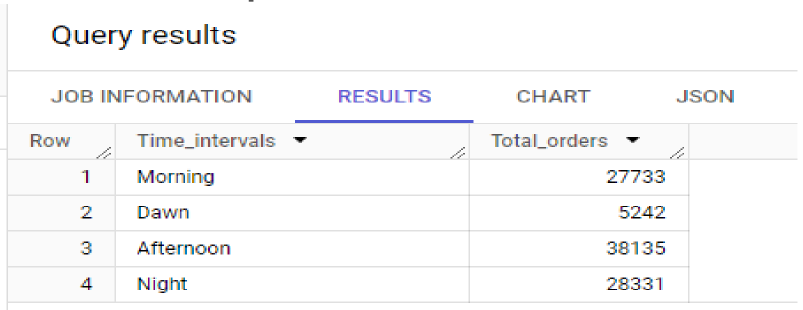
Ans 2.3

Query :-

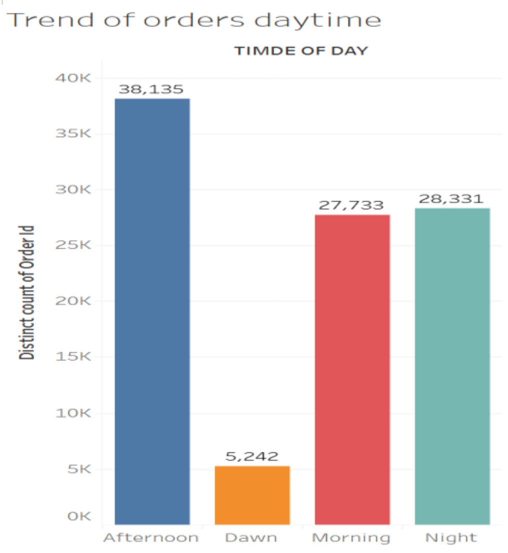
```
select Time_intervals,count(*) as Total_orders
```

```
from
(select *, case when hour between 0 and 6 then 'Dawn'
when hour between 7 and 12 then 'Morning'
when hour between 13 and 18 then 'Afternoon'
when hour between 19 and 23 then 'Night'
else '0'
end as  Time_intervals
from
(select order_id,
EXTRACT(hour from order_purchase_timestamp) AS hour,
from `Target_SQL.orders`)) as tbh
group by Time_intervals
```

Screenshot of output:-



Row	Time_intervals	Total_orders
1	Morning	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331



Insights :-

- Least order was placed during Dawn.
- Most no. of orders was placed in the Afternoon following Night.

Recommendations:-

- Reduce working staff/employees in the Dawn time and shift them to Afternoon and Nighttime or Morning intervals if redundant staff available at the retail.

- Maintain staff strength same for the Morning, Afternoon and Night slot to keep the orders process friction less and with no hindrance.

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

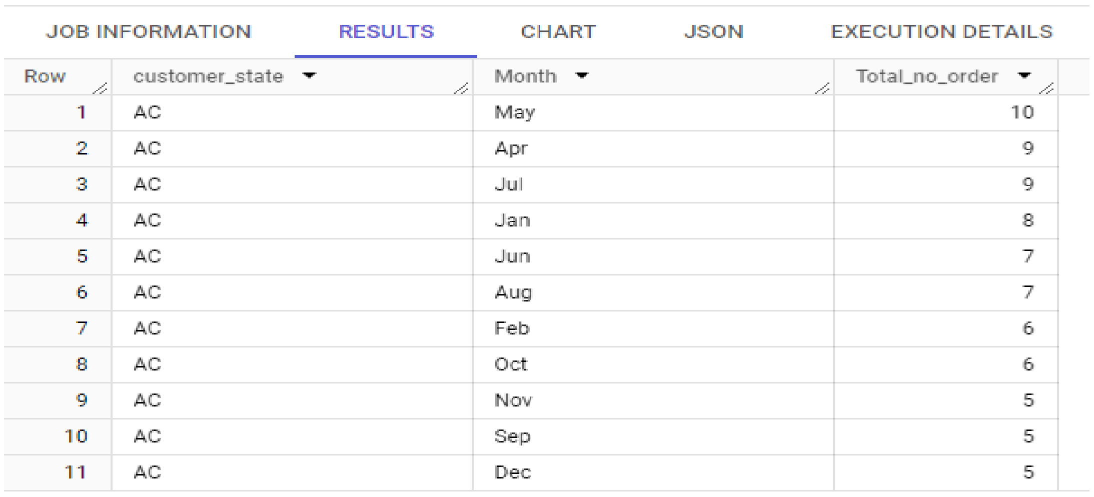
1. Get the month-on-month no. of orders placed in each state.
2. How are the customers distributed across all the states?

Ans 3.1

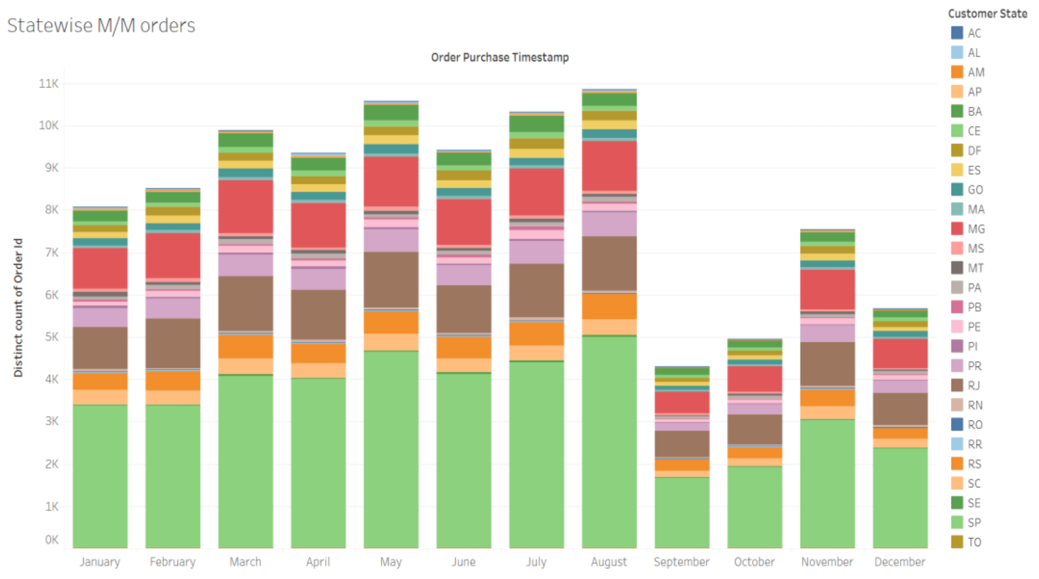
Query :-

```
select customer_state,Month,count(*) as Total_no_order
from
(select *, format_date('%b',order_purchase_timestamp) as Month
from `Target_SQL.orders` as O
join
`Target_SQL.customers` C
on O.customer_id = C.customer_id) as tbh
group by customer_state,Month
order by customer_state ,Total_no_order desc

--- Alternative with CTE

with orders_with_month AS (select c.customer_state, format_date('%b',
o.order_purchase_timestamp) AS order_month
from scalar-dsml-sql-414915`.`BUSINESS_PROJECT_SQL`.`orders` AS o
join `scalar-dsml-sql-414915`.`BUSINESS_PROJECT_SQL`.`customers` AS c
on o.customer_id = c.customer_id )
select customer_state, order_month AS month,count(*) AS total_no_order
from `orders_with_month`
group by customer_state, order_month
order by customer_state,Total_no_order DESC;
```

Screenshot of output:-



Row	customer_state	Month	Total_no_order
1	AC	May	10
2	AC	Apr	9
3	AC	Jul	9
4	AC	Jan	8
5	AC	Jun	7
6	AC	Aug	7
7	AC	Feb	6
8	AC	Oct	6
9	AC	Nov	5
10	AC	Sep	5
11	AC	Dec	5

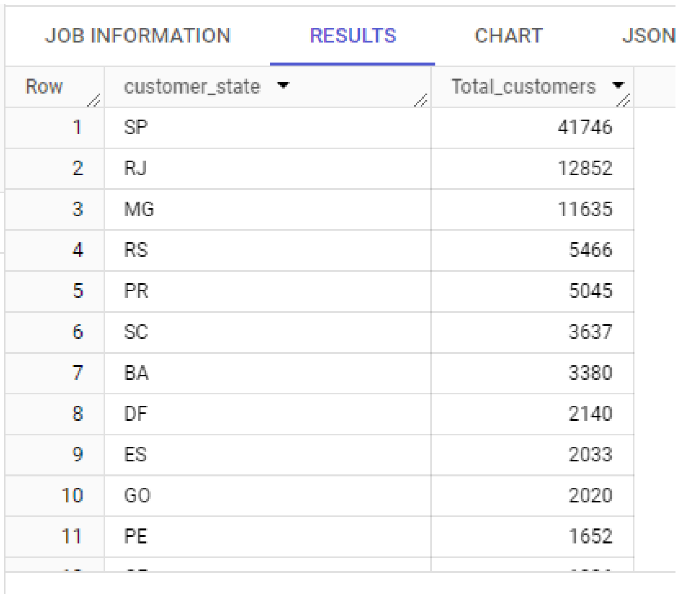


Ans 3.2

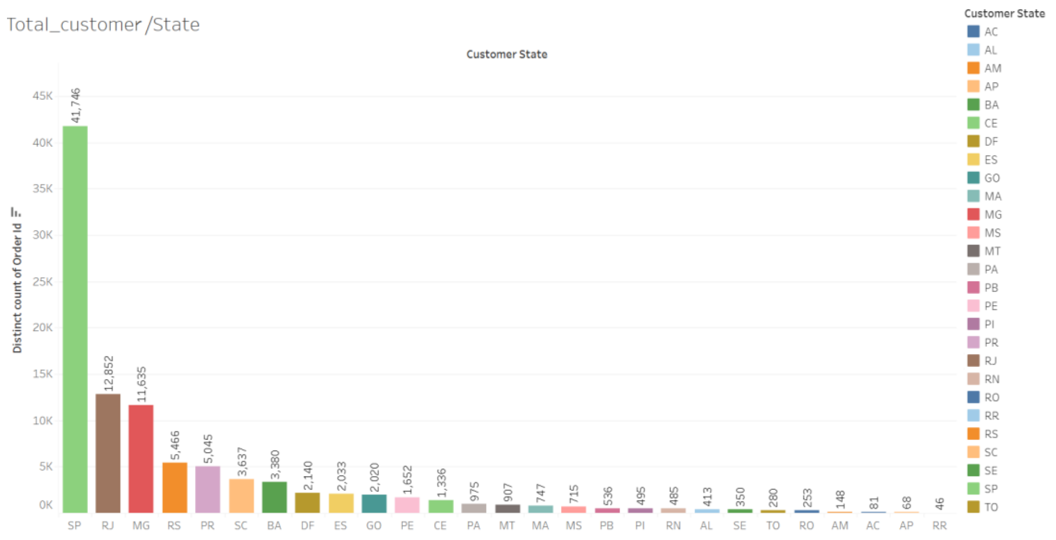
Query :-

```
select C.customer_state,count( distinct O.customer_id) as Total_customers
from `Target_SQL.orders` as O
join
`Target_SQL.customers` C
on O.customer_id = C.customer_id
group by C.customer_state
order by Total_customers desc
```

Screenshot of output:-



Row	customer_state	Total_customers
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
11	PE	1652



Insights :-

- SP state has most no. of customers i.e., 41746.
- Least Unique customers are in RR state i.e. 44
- Majority of customers are in 1 state and not pan Brazil.

Recommendations :-

- It is advisable to stakeholders to expand operations and open new stores in the RJ,MG state as these states are promising in the near future.
- Feedbacks are required from states where no. of customers are less.
- Give discounts in the states where no. of customers is less like SC,BA,DF and so on.
- Business needs to grow pan Brazil and not on just 1 state i.e. SP.

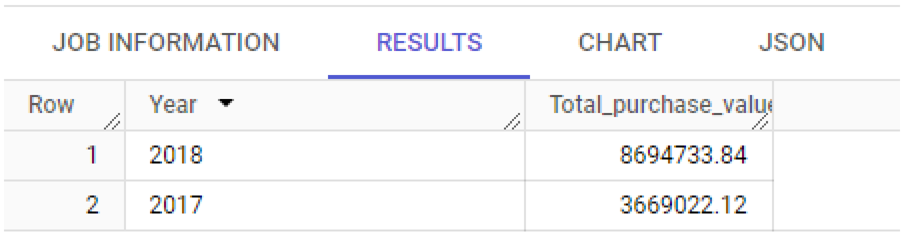
Q-4 Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight, and others.

1. Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).
   You can use the "payment value" column in the payments table to get the cost of orders.
2. Calculate the Total & Average value of order price for each state.
3. Calculate the Total & Average value of order freight for each state.

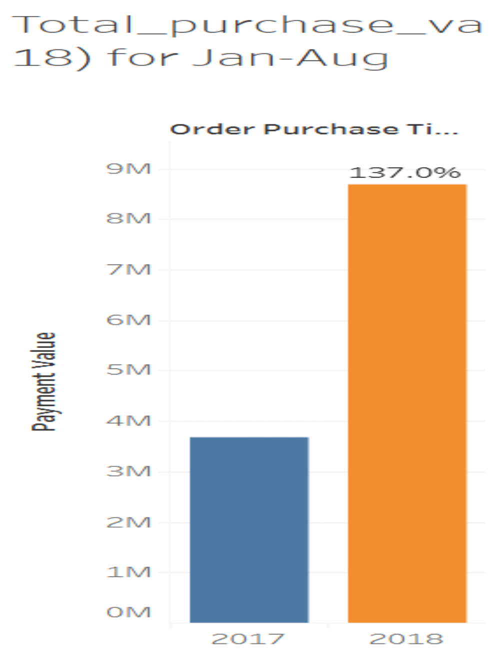
Ans 4.1

```
select  Year,round(sum(payment_value),2) as Total_purchase_value
from
(select P.payment_value, O.order_purchase_timestamp,
format_date('%Y',order_purchase_timestamp) as Year,
format_date('%m',order_purchase_timestamp) as Month
from `Target_SQL.payments` as P
join
`Target_SQL.orders` as O
on P.order_id = O.order_id) as tbh
where Month between '01' and '08'
Group by Year
--- alternative method with CTE

WITH PaymentsWithDateInfo AS (
    SELECT
        P.payment_value,
        EXTRACT(year FROM O.order_purchase_timestamp) AS Year,
        format_date('%m', O.order_purchase_timestamp) AS Month
    FROM `Target_SQL.payments` AS P
    JOIN `Target_SQL.orders` AS O ON P.order_id = O.order_id
)
SELECT
    Year,
    ROUND(SUM(payment_value), 2) AS Total_purchase_value
FROM PaymentsWithDateInfo
WHERE Month BETWEEN '01' AND '08'
GROUP BY
    Year;
```

Screenshot of output:-



Row	Year	Total_purchase_valu
1	2018	8694733.84
2	2017	3669022.12

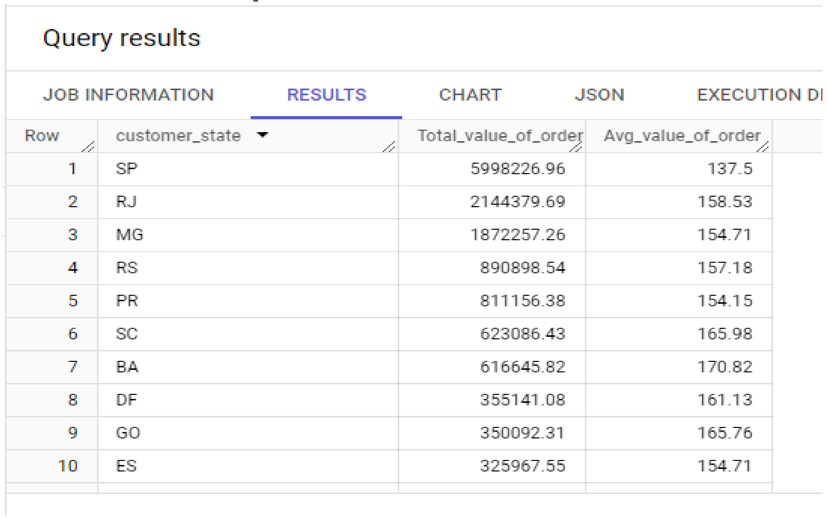


Ans 4.2

Query :-

```
select C.customer_state,
round(sum(P.payment_value),2) as Total_value_of_order,
round(avg(P.payment_value),2) as Avg_value_of_order
from `Target_SQL.orders` as O
join
`Target_SQL.payments` as P
on O.order_id = P.order_id
join
`Target_SQL.customers` as C
on C.customer_id = O.customer_id
group by C.customer_state
order by Total_value_of_order desc  , Avg_value_of_order desc
```

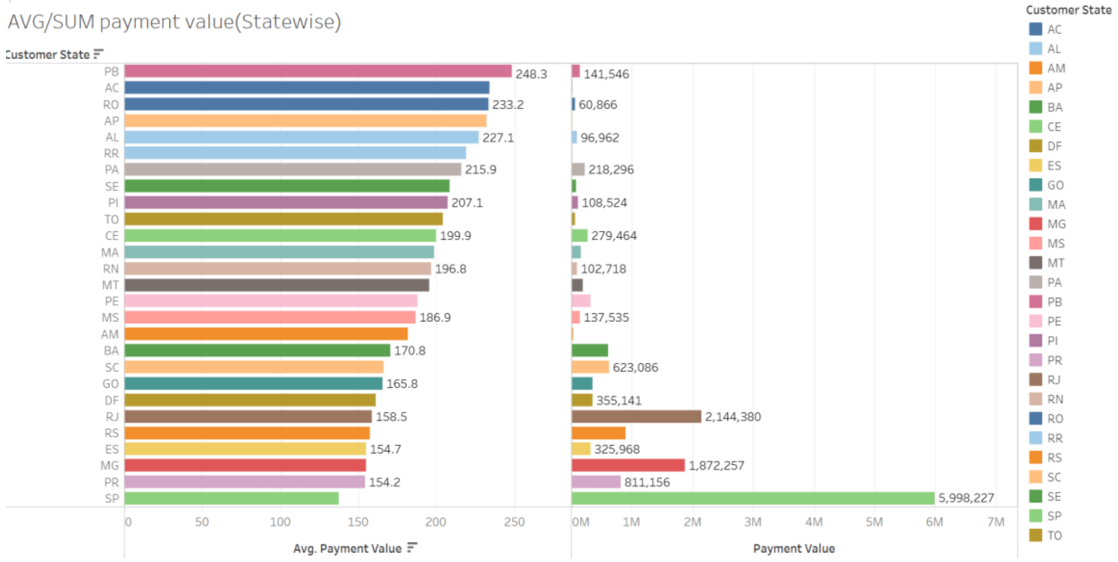
Screenshot of output:-



Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DI

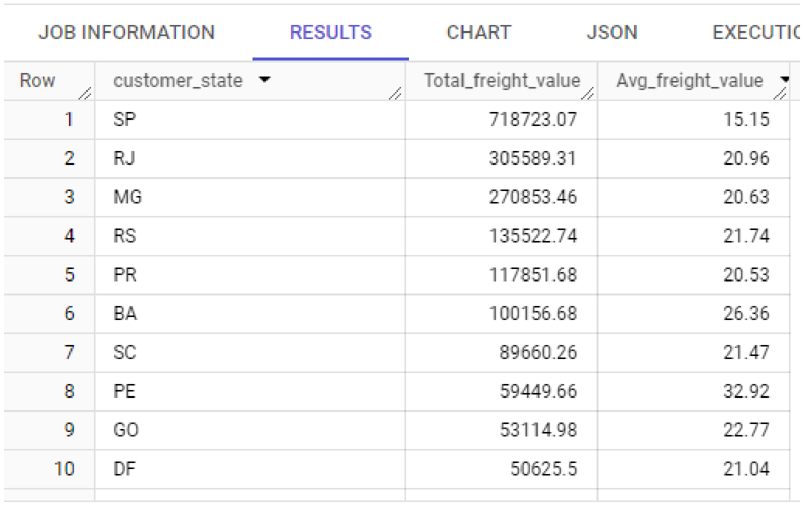
Row	customer_state	Total_value_of_order	Avg_value_of_order
1	SP	5998226.96	137.5
2	RJ	2144379.69	158.53
3	MG	1872257.26	154.71
4	RS	890898.54	157.18
5	PR	811156.38	154.15
6	SC	623086.43	165.98
7	BA	616645.82	170.82
8	DF	355141.08	161.13
9	GO	350092.31	165.76
10	ES	325967.55	154.71



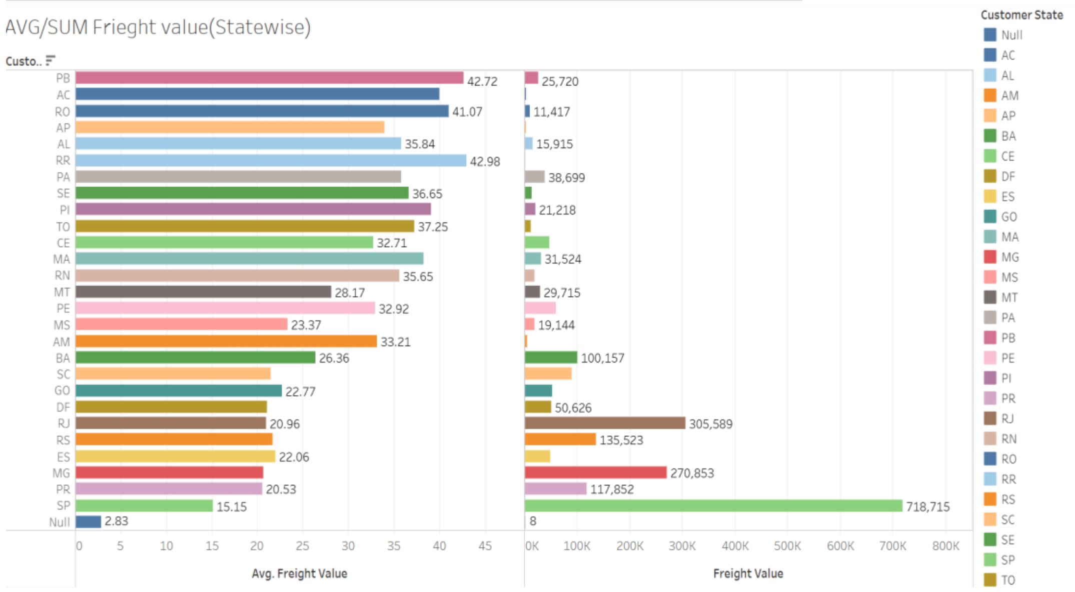
Ans 4.3

```
select C.customer_state,
round(sum(OT.freight_value),2) as Total_freight_value,
round(avg(OT.freight_value),2) as Avg_freight_value
from `Target_SQL.order_items` as OT
left join
`Target_SQL.orders` as O
on OT.order_id = O.order_id
join
`Target_SQL.customers` as C
on O.customer_id = C.customer_id
group by C.customer_state
order by Total_freight_value desc ,Avg_freight_value desc
```

Screenshot of output:-



Row	customer_state	Total_freight_value	Avg_freight_value
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	MG	270853.46	20.63
4	RS	135522.74	21.74
5	PR	117851.68	20.53
6	BA	100156.68	26.36
7	SC	89660.26	21.47
8	PE	59449.66	32.92
9	GO	53114.98	22.77
10	DF	50625.5	21.04



Insights :-

- Total value of orders is maximum for SP state as this state is also having highest no. of customers.
- RR ,AC and PB are having Avg freight value greater than 40.
- Avg freight value of SP is lower than most of the states.
- Total purchased value for the year 2018 has gone up 136% increase from the previous year 2017 which is significantly higher.
- States like PB,AC,AP are having much higher avg value of an order considering their lower no. of customers .

Recommendations :-

- Workforce must increase for the upcoming year i.e 2019 as the business is growing.
- Increment to the employees should be awarded.

- Top 8 states of total value of order should increase their Avg value of order by giving combo offers and discounts on secondary products.

Q – 5 Analysis based on sales, freight and delivery time.

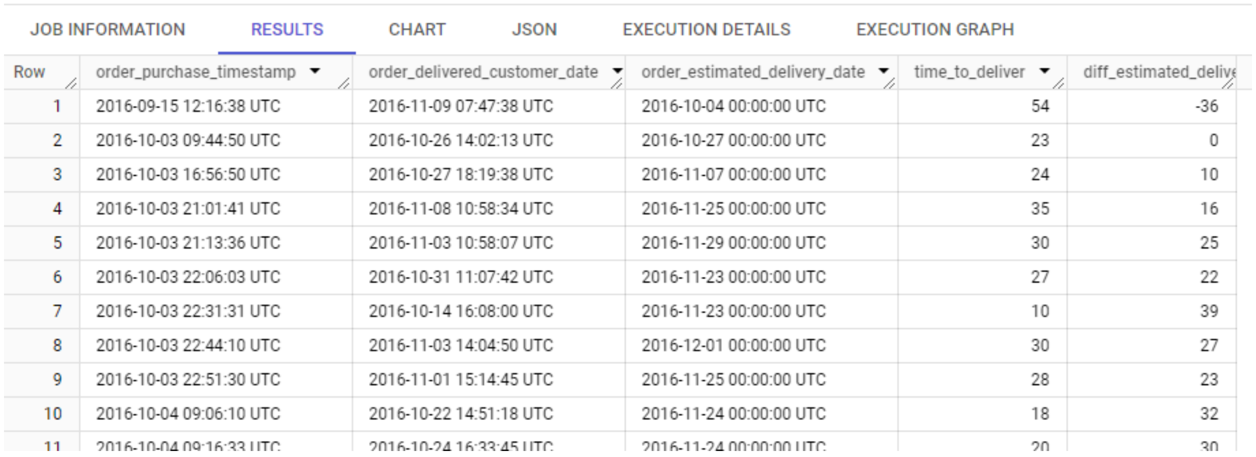
1. Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
   Also, calculate the difference (in days) between the estimated & actual delivery date of an order.
   Do this in a single query.

   You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:
   - **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
   - **diff_estimated_delivery** = order_delivered_customer_date - order_estimated_delivery_date
2. Find out the top 5 states with the highest & lowest average freight value.
3. Find out the top 5 states with the highest & lowest average delivery time.
4. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.
   You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.
5. What is the average time from purchase to shipment and from shipment to delivery?
6. Are there geographic regions (states or cities) with consistently longer shipping times?

Ans 5.1
Query :-

```
select
order_purchase_timestamp,order_delivered_customer_date,order_estimated_delive
ry_date,
date_diff(order_delivered_customer_date , order_purchase_timestamp,day) as
time_to_deliver,
date_diff(order_estimated_delivery_date, order_delivered_customer_date ,day)
as diff_estimated_deliver
from `Target_SQL.orders`
where order_delivered_customer_date is not null
order by order_purchase_timestamp
```

Screenshot of output:-



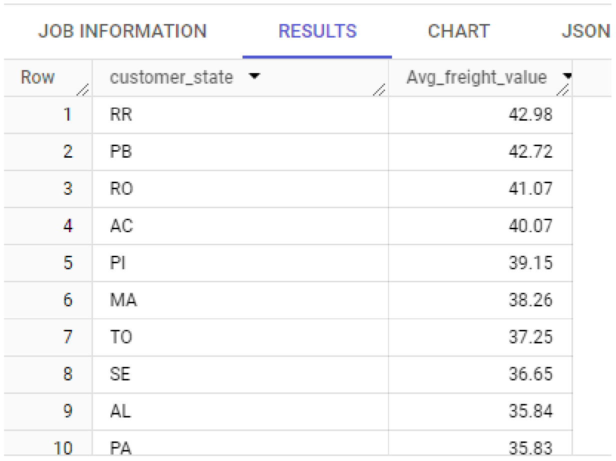
Row	order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date	time_to_deliver	diff_estimated_delive
1	2016-09-15 12:16:38 UTC	2016-11-09 07:47:38 UTC	2016-10-04 00:00:00 UTC	54	-36
2	2016-10-03 09:44:50 UTC	2016-10-26 14:02:13 UTC	2016-10-27 00:00:00 UTC	23	0
3	2016-10-03 16:56:50 UTC	2016-10-27 18:19:38 UTC	2016-11-07 00:00:00 UTC	24	10
4	2016-10-03 21:01:41 UTC	2016-11-08 10:58:34 UTC	2016-11-25 00:00:00 UTC	35	16
5	2016-10-03 21:13:36 UTC	2016-11-03 10:58:07 UTC	2016-11-29 00:00:00 UTC	30	25
6	2016-10-03 22:06:03 UTC	2016-10-31 11:07:42 UTC	2016-11-23 00:00:00 UTC	27	22
7	2016-10-03 22:31:31 UTC	2016-10-14 16:08:00 UTC	2016-11-23 00:00:00 UTC	10	39
8	2016-10-03 22:44:10 UTC	2016-11-03 14:04:50 UTC	2016-12-01 00:00:00 UTC	30	27
9	2016-10-03 22:51:30 UTC	2016-11-01 15:14:45 UTC	2016-11-25 00:00:00 UTC	28	23
10	2016-10-04 09:06:10 UTC	2016-10-22 14:51:18 UTC	2016-11-24 00:00:00 UTC	18	32
11	2016-10-04 09:16:33 UTC	2016-10-24 16:33:45 UTC	2016-11-24 00:00:00 UTC	20	30

Ans 5.2

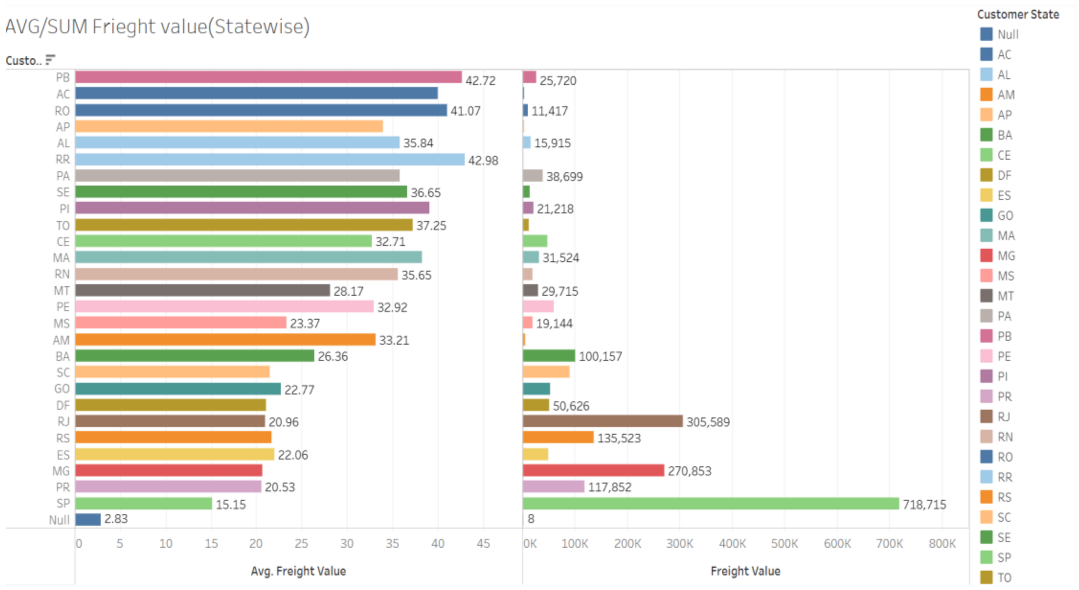
Query :- Top 5 states based on Avg freight value.

```
select C.customer_state, round(avg(OT.freight_value),2) as Avg_freight_value
from `Target_SQL.order_items` as OT
join
`Target_SQL.orders` as O
on OT.order_id = O.order_id
join
`Target_SQL.customers` as C
on O.customer_id = C.customer_id
group by C.customer_state
order by Avg_freight_value desc
```

Screenshot of output:-



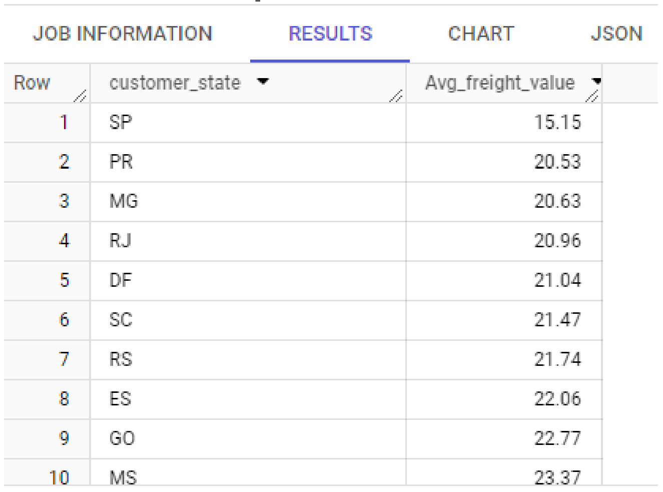
Row	customer_state	Avg_freight_value
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15
6	MA	38.26
7	TO	37.25
8	SE	36.65
9	AL	35.84
10	PA	35.83



Query :- Bottom 5 states based on Avg freight value.

```
select C.customer_state, round(avg(OT.freight_value),2) as Avg_freight_value
from `Target_SQL.order_items` as OT
join
`Target_SQL.orders` as O
on OT.order_id = O.order_id
join
`Target_SQL.customers` as C
on O.customer_id = C.customer_id
group by C.customer_state
order by Avg_freight_value

--- alternatve method with window function
WITH AvgFreight AS (
    SELECT
        C.customer_state,
        ROUND(AVG(OT.freight_value) OVER (PARTITION BY C.customer_state), 2) AS
Avg_freight_value
    FROM `Target_SQL.order_items` AS OT
    JOIN `Target_SQL.orders` AS O ON OT.order_id = O.order_id
    JOIN `Target_SQL.customers` AS C ON O.customer_id = C.customer_id
)
SELECT DISTINCT
    customer_state,
    Avg_freight_value
FROM AvgFreight
ORDER BY
     Avg_freight_value
LIMIT 5;
```

Screenshot of output:-



JOB INFORMATION RESULTS CHART JSON

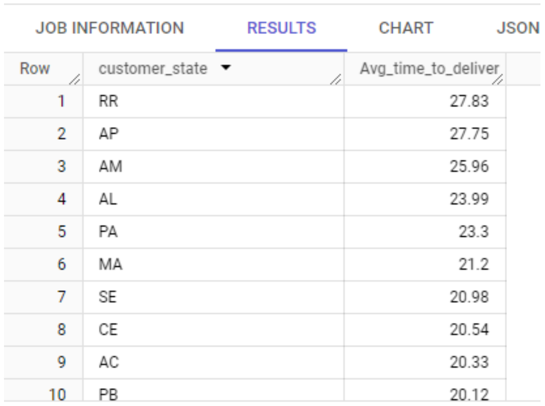
Row	customer_state	Avg_freight_value
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04
6	SC	21.47
7	RS	21.74
8	ES	22.06
9	GO	22.77
10	MS	23.37

Ans 5.3

Query :- Top 5 states based on Avg time to deliver.

```
select C.customer_state,round(avg(date_diff(order_delivered_customer_date ,
order_purchase_timestamp,day)),2) as Avg_time_to_deliver
from `Target_SQL.order_items` as OT
join
`Target_SQL.orders` as O
on OT.order_id = O.order_id
join
`Target_SQL.customers` as C
on O.customer_id = C.customer_id
where order_delivered_customer_date is not null
group by C.customer_state
order by Avg_time_to_deliver desc
```

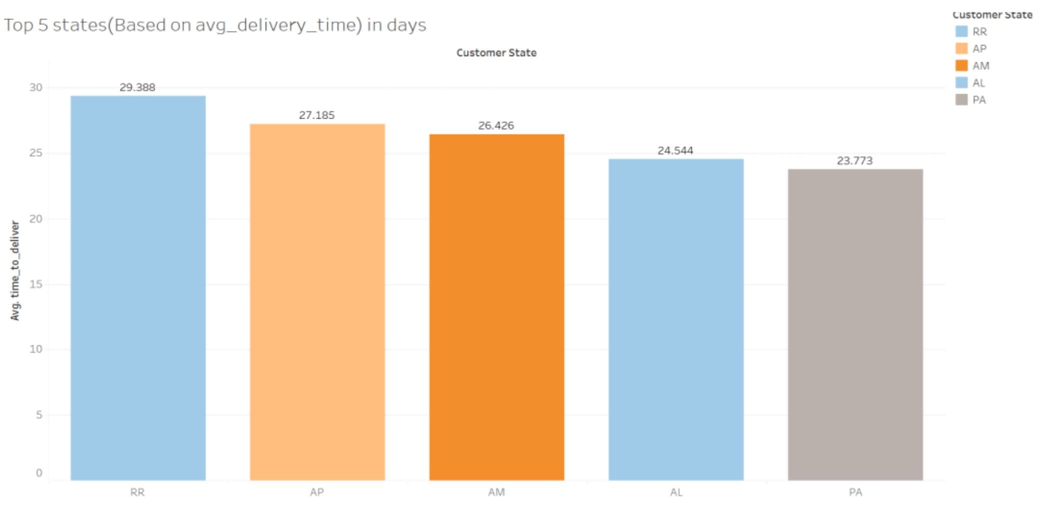
Screenshot of output:-



JOB INFORMATION RESULTS CHART JSON

Row	customer_state	Avg_time_to_deliver
1	RR	27.83
2	AP	27.75
3	AM	25.96
4	AL	23.99
5	PA	23.3
6	MA	21.2
7	SE	20.98
8	CE	20.54
9	AC	20.33
10	PB	20.12

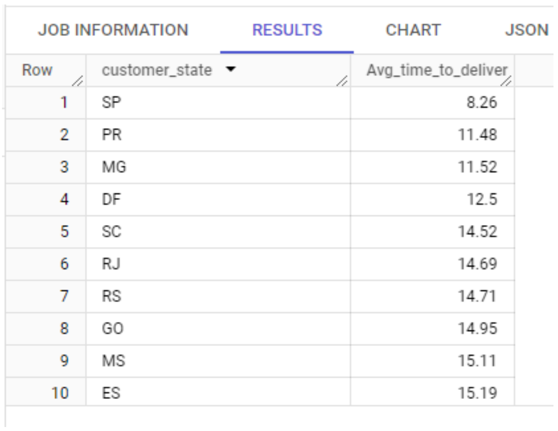
Top 5 states(Based on avg_delivery_time) in days



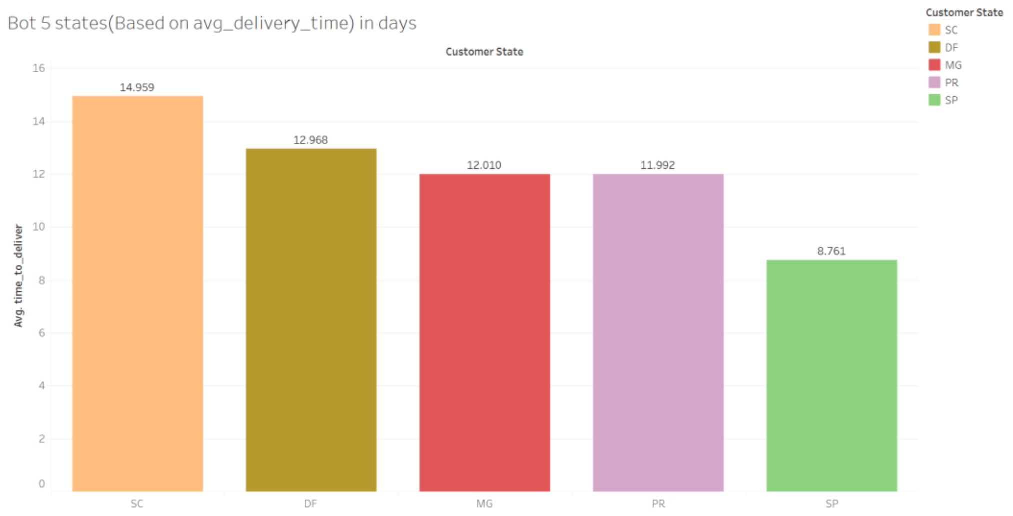
Query :- Bottom 5 states based on Avg time to deliver.

```
select C.customer_state,round(avg(date_diff(order_delivered_customer_date ,
order_purchase_timestamp,day)),2) as Avg_time_to_deliver
from `Target_SQL.order_items` as OT
join
`Target_SQL.orders` as O
on OT.order_id = O.order_id
join
`Target_SQL.customers` as C
on O.customer_id = C.customer_id
where order_delivered_customer_date is not null
group by C.customer_state
order by Avg_time_to_deliver
```

Screenshot of output:-



Row	customer_state	Avg_time_to_deliver
1	SP	8.26
2	PR	11.48
3	MG	11.52
4	DF	12.5
5	SC	14.52
6	RJ	14.69
7	RS	14.71
8	GO	14.95
9	MS	15.11
10	ES	15.19

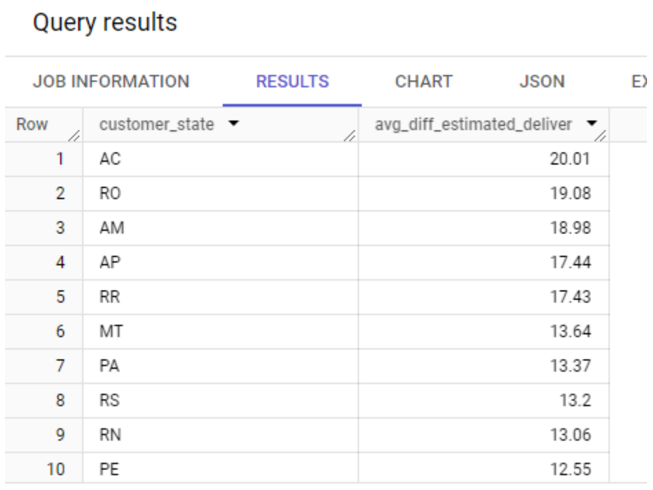


Ans 5.4

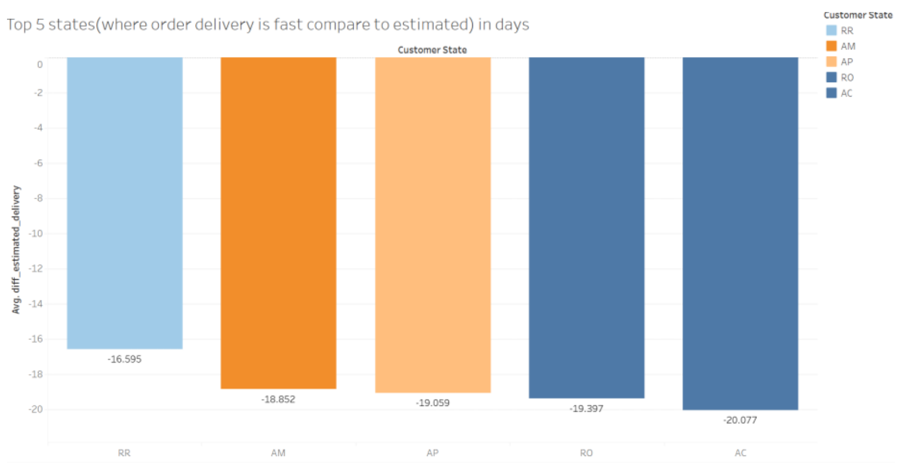
Query :- Top 5 states based on avg difference in estimated deliver

```
select C.customer_state,round(avg(date_diff(order_estimated_delivery_date,
order_delivered_customer_date ,day)),2) as avg_diff_estimated_deliver
from `Target_SQL.order_items` as OT
join
`Target_SQL.orders` as O
on OT.order_id = O.order_id
join
`Target_SQL.customers` as C
on O.customer_id = C.customer_id
where O.order_status like 'delivered'
group by C.customer_state
order by avg_diff_estimated_deliver desc
```

Screenshot of output:-



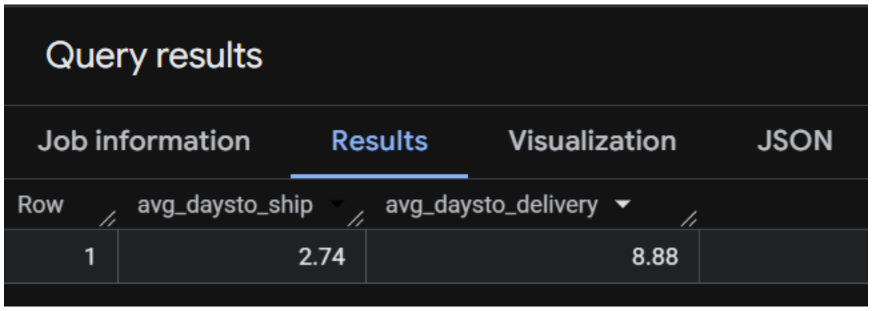
Row	customer_state	avg_diff_estimated_deliver
1	AC	20.01
2	RO	19.08
3	AM	18.98
4	AP	17.44
5	RR	17.43
6	MT	13.64
7	PA	13.37
8	RS	13.2
9	RN	13.06
10	PE	12.55



Ans 5.5

```
with logistic_days as (
select
order_id,customer_id,order_purchase_timestamp,order_delivered_carrier_date,or
der_delivered_customer_date,date_diff(order_delivered_carrier_date,order_purc
hase_timestamp,day) as
days_ship,date_diff(order_delivered_customer_date,order_delivered_carrier_dat
e,day) as days_delivery
from `BUSINESS_PROJECT_SQL.orders`
where order_purchase_timestamp is not null and order_delivered_carrier_date
is not null and order_delivered_customer_date is not null and order_status
like'delivered' and order_delivered_customer_date >=
order_delivered_carrier_date
and order_delivered_carrier_date >= order_purchase_timestamp
order by days_delivery asc)
select round(avg(days_ship),2) as avg_daysto_ship,round(avg(days_delivery),2)
as avg_daysto_delivery
from logistic_days
```

Screenshot of output:-

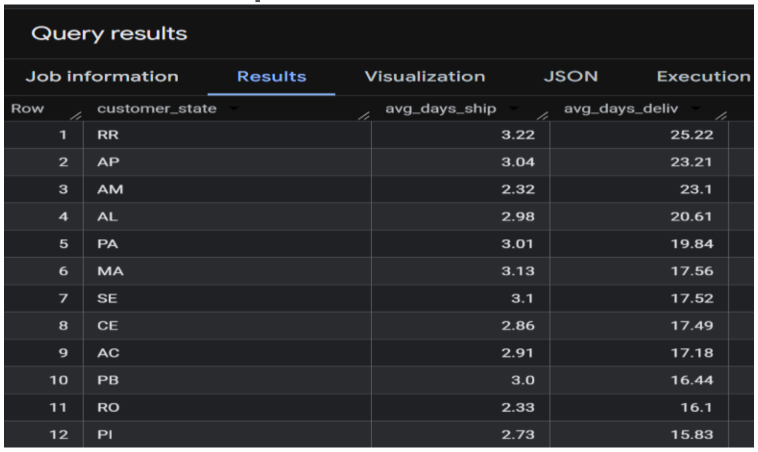


Row	avg_daysto_ship	avg_daysto_delivery
1	2.74	8.88

Ans 5.6

```
with logistic_days as (
select order_id,
customer_id,order_purchase_timestamp,order_delivered_carrier_date,order_deliv
ered_customer_date,date_diff(order_delivered_carrier_date,order_purchase_time
stamp,day) as
days_ship,date_diff(order_delivered_customer_date,order_delivered_carrier_dat
e,day) as days_delivery
from `BUSINESS_PROJECT_SQL.orders`
where order_purchase_timestamp is not null and order_delivered_carrier_date
is not null and order_delivered_customer_date is not null and order_status
like'delivered' and order_delivered_customer_date >=
order_delivered_carrier_date
and order_delivered_carrier_date >= order_purchase_timestamp
order by days_delivery asc),
customer_state_city as  (
select
order_id,days_ship,days_delivery,customer_zip_code_prefix,customer_city,custo
mer_state
from logistic_days
join `BUSINESS_PROJECT_SQL.customers` as c
on logistic_days.customer_id = c.customer_id),
customer_city as(
select customer_city , round(avg(days_ship),2) as
avg_days_ship,round(avg(days_delivery),2) as avg_days_deliv
from customer_state_city
group by customer_city)
select customer_state, round(avg(days_ship),2) as
avg_days_ship,round(avg(days_delivery),2) as avg_days_deliv
from customer_state_city
group by customer_state
order by avg_days_ship desc,avg_days_deliv desc
```

Screenshot of output:-



Query results

Job information | Results | Visualization | JSON | Execution

Row	customer_state	avg_days_ship	avg_days_deliv
1	RR	3.22	25.22
2	AP	3.04	23.21
3	AM	2.32	23.1
4	AL	2.98	20.61
5	PA	3.01	19.84
6	MA	3.13	17.56
7	SE	3.1	17.52
8	CE	2.86	17.49
9	AC	2.91	17.18
10	PB	3.0	16.44
11	RO	2.33	16.1
12	PI	2.73	15.83

Insights :-

- RR,PB,RO,AC PI are top 5 states based on Avg freight value having higher Avg freight value.
- SP,PR,MG, RJ,DF are bottom 5 states based on Avg freight value having lower Avg freight value.
- RR,AP, AM,AL,PA are the top 5 states have taken more time to deliver and calculated based on Avg time to deliver.
- Likewise, SP,PR,MG,DF,SC are bottom 5 states based on Avg time to deliver.
- AC,RO,AM,AP,RR are top 5 states based on avg difference in estimated deliver where order was delivered much prior to estimated delivery date.
- Maximum days took for an order to deliver was 209 days and there were some days where order was delivered on the same day also.
- Most of the orders in March 2017 orders took longer days to deliver i.e around 160-200 days.
- Average time taken for the product from purchasing to shipments is 2.7 days whereas avg time taken from shipment to delivery is 8.9 days.
- RR, AP, AM, AL are among the top states which takes more time for shipping the products compare to other that is more than 20 days.

Recommendations :-

- RR,AP, AM,AL,PA needs to work more on logistics to lower down delivery time and freight value. More delivery partners are required in these states.
- RR,AP, AM,AL,PA states should consult with AC,RO,AM,AP states to efficiently work on logistics vertical.
- Orders in March 2017 need to be reexamined as to why delivery of orders took so many days within meeting of every vertical.
- Logistics and carrier performance should be improved because there is higher delay in days for the product from shipment to delivery especially for the products which are delivered in these states (RR, AP, AM, AL). This is also resulting in higher cost of freight per order.

Q-6. Analysis based on the payments:

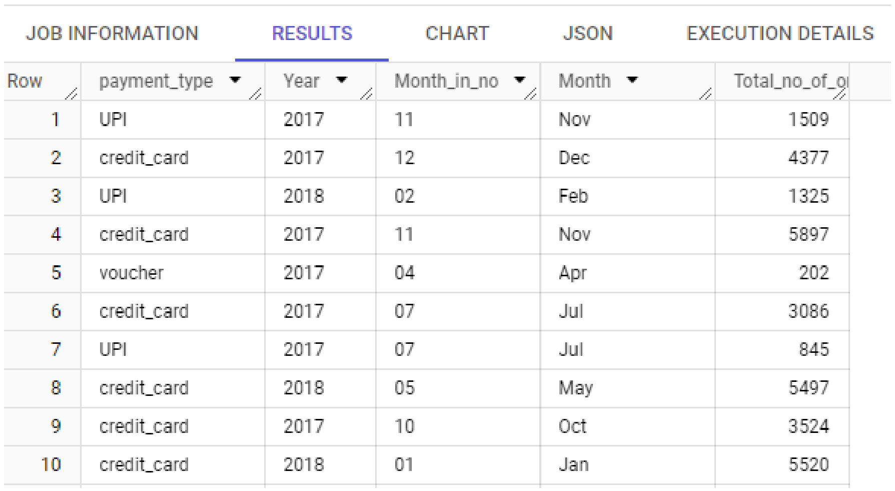
1. Find the month on month no. of orders placed using different payment types.
2. Find the no. of orders placed on the basis of the payment installments that have been paid.
3. How much of the total payment per order is accounted for by the item price and the shipping cost, both individually and combined.
4. What are the top-earning product categories?

Ans 6.1

Query :-

```
select payment_type,Year,Month_in_no,Month,count(payment_type) as
Total_no_of_order
from
(select P.payment_type,
format_date('%Y',order_purchase_timestamp) as Year,
format_date('%m',order_purchase_timestamp) as Month_in_no,
format_date('%b',order_purchase_timestamp) as Month
from `Target_SQL.orders` as O
join
`Target_SQL.payments` as P
on O.order_id = P.order_id
order by Year,Month_in_no) as tbh
group by payment_type,Year,Month_in_no,Month

--Alternative Method using a CTE:

WITH PaymentsWithDateInfo AS (
    SELECT
        P.payment_type,
        FORMAT_DATE('%Y', O.order_purchase_timestamp) AS Year,
        FORMAT_DATE('%m', O.order_purchase_timestamp) AS Month_in_no,
        FORMAT_DATE('%b', O.order_purchase_timestamp) AS Month
    FROM `Target_SQL.orders` AS O
    JOIN `Target_SQL.payments` AS P ON O.order_id = P.order_id
)
SELECT
    payment_type,
    Year,
    Month_in_no,
    Month,
    COUNT(payment_type) AS Total_no_of_order
FROM PaymentsWithDateInfo
GROUP BY
    payment_type,
    Year,
    Month_in_no,
    Month
ORDER BY
    Year,
    Month_in_no;
```

Screenshot of output:-


JOB INFORMATION RESULTS CHART JSON EXECUTION DETAILS

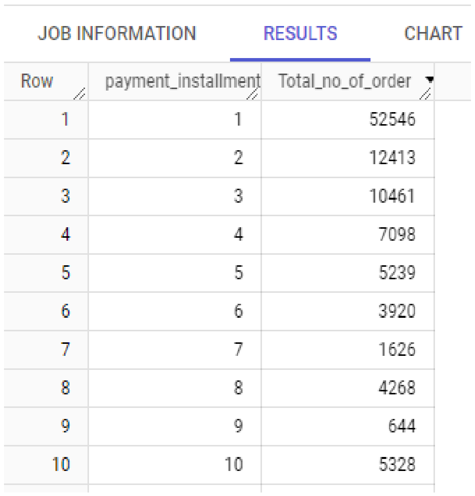
Row	payment_type	Year	Month_in_no	Month	Total_no_of_o
1	UPI	2017	11	Nov	1509
2	credit_card	2017	12	Dec	4377
3	UPI	2018	02	Feb	1325
4	credit_card	2017	11	Nov	5897
5	voucher	2017	04	Apr	202
6	credit_card	2017	07	Jul	3086
7	UPI	2017	07	Jul	845
8	credit_card	2018	05	May	5497
9	credit_card	2017	10	Oct	3524
10	credit_card	2018	01	Jan	5520

Ans 6.2

Query :-

```
select payment_installments,count(payment_installments) as Total_no_of_order
from `Target_SQL.payments` as P
join
`Target_SQL.orders` as O
on P.order_id = O.order_id
where payment_installments <> 0
group by payment_installments
```

Screenshot of output:-


JOB INFORMATION RESULTS CHART

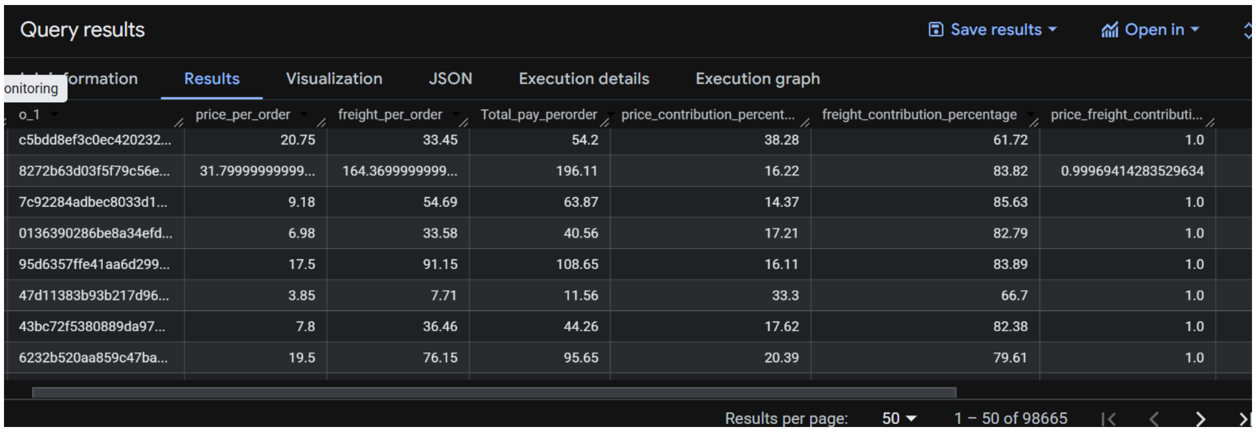
Row	payment_installment	Total_no_of_order
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	5	5239
6	6	3920
7	7	1626
8	8	4268
9	9	644
10	10	5328

Ans 6.3

Query :-

```
with aggregate_table as (
select *
from
(select order_id as o_1,sum(price) as price_per_order,sum(freight_value) as
freight_per_order
from `BUSINESS_PROJECT_SQL.order_items`
group by order_id) as t_1
join
(select order_id as o_2,sum(payment_value) as Total_pay_perorder
from `BUSINESS_PROJECT_SQL.payments`
group by order_id ) as t_2
on t_1.o_1 = t_2.o_2),
contribution as(
select
o_1,price_per_order,freight_per_order,Total_pay_perorder,round((price_per_ord
er/Total_pay_perorder)*100,2) as
price_contribution_percentage,round((freight_per_order/Total_pay_perorder)*10
0,2) as freight_contribution_percentage
from aggregate_table)
select *,Total_pay_perorder/(price_per_order+freight_per_order) as
price_freight_contribution
from contribution
```

Screenshot of output:-



o_1	price_per_order	freight_per_order	Total_pay_perorder	price_contribution_percent...	freight_contribution_percentage	price_freight_contributi...
c5bdd8ef3c0ec420232...	20.75	33.45	54.2	38.28	61.72	1.0
8272b63d03f5f79c56e...	31.79999999999...	164.3699999999...	196.11	16.22	83.82	0.99969414283529634
7c92284adbec8033d1...	9.18	54.69	63.87	14.37	85.63	1.0
0136390286be8a34efd...	6.98	33.58	40.56	17.21	82.79	1.0
95d6357ffe41aa6d299...	17.5	91.15	108.65	16.11	83.89	1.0
47d11383b93b217d96...	3.85	7.71	11.56	33.3	66.7	1.0
43bc72f5380889da97...	7.8	36.46	44.26	17.62	82.38	1.0
6232b520aa859c47ba...	19.5	76.15	95.65	20.39	79.61	1.0

Results per page: 50 1 – 50 of 98665

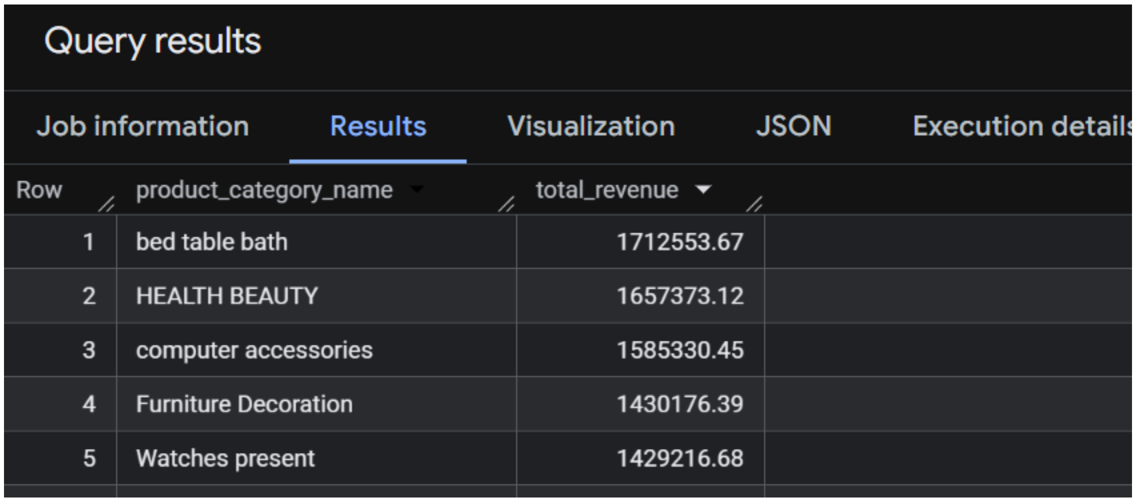
Ans 6.4

Query :-

```
select pr.product_category_name,round(sum(p.payment_value),2) as
total_revenue
from `BUSINESS_PROJECT_SQL.orders`  as o
```

```
join `BUSINESS_PROJECT_SQL.order_items` as i
on o.order_id = i.order_id
join `BUSINESS_PROJECT_SQL.payments` as p
on p.order_id = o.order_id
join `BUSINESS_PROJECT_SQL.products` as pr
on pr.product_id = i.product_id
group by pr.product_category_name
order by total_revenue desc
```

Screenshot of output:-



Row	product_category_name	total_revenue
1	bed table bath	1712553.67
2	HEALTH BEAUTY	1657373.12
3	computer accessories	1585330.45
4	Furniture Decoration	1430176.39
5	Watches present	1429216.68

Insights :-

- 90% orders were of credit type i.e 76795 no. of order.
- Least payment type for orders were of debit card.
- Most of the orders i.e., 52546 have completed 1st payment installment.
- Customers are paying more for freight than for the product itself, on a per-order basis, this is because of high shipping rate for a low-price product and it leads to inefficient logistics.
- Bed table bath, health beauty, computer accessories, furniture decoration, watches present are the top 5 earning product categories.

Recommendation :-

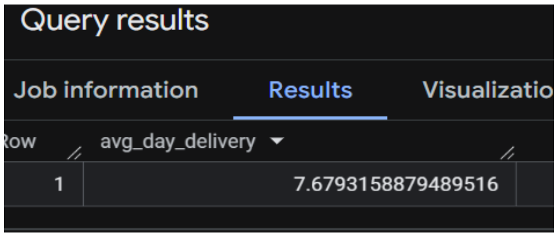
- Offers on UPI and debit card should be promoted by giving offers on these transactions.
- Encourage customers to buy in bundles or set minimum order thresholds to reduce per-unit freight cost.

Q-7. Calculates the average delivery days for orders with high review scores (>3)and low review score (<3)

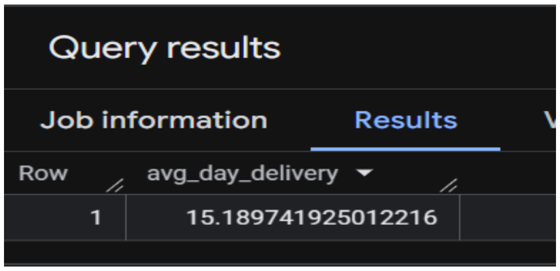
Ans 7
Query :-

```
with logistic_days as (
select
order_id,customer_id,order_purchase_timestamp,order_delivered_carrier_
date,order_delivered_customer_date,date_diff(order_delivered_carrier_d
ate,order_purchase_timestamp,day) as
days_ship,date_diff(order_delivered_customer_date,order_delivered_carr
ier_date,day) as days_delivery
from `BUSINESS_PROJECT_SQL.orders`
where order_purchase_timestamp is not null and
order_delivered_carrier_date is not null and
order_delivered_customer_date is not null and order_status
like'delivered' and order_delivered_customer_date >=
order_delivered_carrier_date
and order_delivered_carrier_date >= order_purchase_timestamp
order by days_delivery asc),
low_review as
(select logistic_days.order_id,days_ship,days_delivery,review_score
from logistic_days
join `BUSINESS_PROJECT_SQL.order_reviews` as  r
on logistic_days.order_id = r.order_id
where review_score <3),
high_review as
(select logistic_days.order_id,days_ship,days_delivery,review_score
from logistic_days
join `BUSINESS_PROJECT_SQL.order_reviews` as  r
on logistic_days.order_id = r.order_id
where review_score >3)
select avg(days_delivery) as avg_day_delivery
from high_review
```

Screenshot of output:-



```
Row   avg_day_delivery
1     7.6793158879489516
```



```
Row   avg_day_delivery
  1   15.189741925012216
```

Insights :-

- For High review score average day delivery is around 8 days whereas for low review score is around 15 days.

Recommendation :-

- Place stock closer to demand centers ware house to shorten last-mile delivery.
- Offer premium “fast shipping” choices for customers willing to pay extra.

FINAL INSIGHTS

- Orders and no. of customers in states like SP are promising and doing amazing business compare to other states .
- Business for the 2019 year will be growing seeing growth in no. of orders.
- Majority of customers are in 1 state i.e SP and not pan Brazil.
- Payment type of orders should be well diversified.
- Customers are paying more for freight than for the product itself, on a per-order basis, this is because of high shipping rate for a low-price product and it leads to inefficient logistics.
- Average time taken for the product from purchasing to shipments is 2.7 days whereas avg time taken from shipment to delivery is 8.9 days.
- RR, AP, AM, AL are among the top states which takes more time for shipping the products compare to other that is more than 20 days.
- For High review score average day delivery is around 8 days whereas for low review score is around 15 days

FINAL RECOMMENDATION

- Offers on secondary products should be given to increase avg order price.
- Ad campaign should be advertised in the states where customers are less.
- Company should work more on Logistics team as to reduce the delivery time.
- Encourage customers to buy in bundles or set minimum order thresholds to reduce per-unit freight cost.
- Logistics and carrier performance should be improved because there is higher delay in days for the product from shipment to delivery especially for the products which are

delivered in these states (RR, AP, AM, AL). This is also resulting in higher cost of freight per order.

- Offer premium “fast shipping” choices for customers willing to pay extra.
- Place stock closer to demand centers ware house to shorten last-mile delivery.