# XYZ CASE STUDY

Note: For Initial few Questions (Q 1.1, 1.2, 1.3, 2.1) I have used BigQuery as I was facing some issue while downloading MySQL and the later questions are solved using MySQL. Apologies for any inconvenience caused

---

Q1 **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.

**Solution**

**Query** : SHOW COLUMNS FROM customers IN xyz;
OR
DESCRIBE customers; **Output**
:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Field | Type | Null | Key | Default | Extra |
|---|---|---|---|---|---|
| customer_id | text | YES | | NULL | |
| customer_unique_id | text | YES | | NULL | |
| customer_zip_code_prefix | int | YES | | NULL | |
| customer_city | text | YES | | NULL | |
| customer_state | text | YES | | NULL | |

Result 1

**2)** Get the time range between which the orders were placed.

**Solution**:

**Query**: SELECT MIN(order_purchase_timestamp) as `FROM`,
MAX(order_purchase_timestamp) as `TO`
FROM `business-case01.xyz01.orders`;

**Output**:

```
SELECT min(order_purchase_timestamp) as `FROM`, max(order_purchase_timestamp) as `TO` FROM `business-case01.target01.orders`;
```

Press Alt+F1 for Accessibilit

ıery results | SAVE RESULTS | EXPLORE DATA

B INFORMATION | RESULTS | CHART PREVIEW | JSON | EXECUTION DETAILS | EXECUTION GRAPH

| | FROM | TO |
|---|---|---|
| 1 | 2016-09-04 21:15:19 UTC | 2018-10-17 17:30:18 UTC |

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

**Solution**:

**Query**: select count(distinct customer_city) as `No Of Cities`, count(distinct customer_state) as `No Of States`
from `xyz01.customers` join `xyz01.orders` using (customer_id);

**Output**:

```
#select * from `target01.customers`;
select count(distinct customer_city) as `No Of Cities`,count(distinct customer_state) as `No Of States` from `target01.customers` join `target01.orders` using (customer_id);
```

Press Alt+F1 for Accessibility Opti

Query results | SAVE RESULTS | EXPLORE DATA

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXECUTION DETAILS | EXECUTION GRAPH

| Row | No Of Cities | No Of States |
|---|---|---|
| 1 | 4119 | 27 |

---

**Q2) In-depth Exploration:**

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

**Solution**:

**Query** : with cte as (
select extract(year from order_purchase_timestamp) as `YEAR`, count(distinct order_id) as `No Of Orders Placed` from xyz01.orders group by `YEAR` order by YEAR desc)
select *,concat(ifnull(round(((`No Of Orders Placed`-lag(`No Of Orders Placed`) over(order by YEAR))/lag(`No Of Orders Placed`) over(order by YEAR))*100,1),0),'%') as `Increase Percentage`
from cte order by cte.`YEAR` desc;

**Output**:

```
#select * from target01.orders;
with cte as (
select extract(year from order_purchase_timestamp) as `YEAR`,count(distinct order_id) as `No Of Orders Placed` from target01.orders group by `YEAR` order by YEAR desc)
select *,concat(ifnull(round((((`No Of Orders Placed`-lag(`No Of Orders Placed`) over(order by YEAR))/lag(`No Of Orders Placed`) over(order by YEAR))*100,1),0),'%') as `Increase Percentage` from cte order by cte.`YEAR` desc;
```

Press Alt+F1 for Accessibility Opt

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXECUTION DETAILS | EXECUTION GRAPH

| ow | YEAR | No Of Orders Placed | Increase Percentage |
|---|---|---|---|
| 1 | 2018 | 54011 | 19.8% |
| 2 | 2017 | 45101 | 13608.5% |
| 3 | 2016 | 329 | 0% |

**Insights**: Yes, there is a growing trend observed in the no. of orders placed. In the second year (2017), a sudden spike in the no. of orders placed, was observed; probably due missing data before September 2017. The growth seems to achieve steady increased flow from 2017 to 2018.

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

**Approach**: The above question can be solved using 2 approaches without CTE and with CTE respectively

**Approach** 1) Without CTE

**Query** : select month(order_purchase_timestamp) as Mno,
date_format(order_purchase_timestamp,'%M') as `Month`,
count((case when year(order_purchase_timestamp)=2016
then order_id end)) as `Monthly Orders in 2016`, count((case
when year(order_purchase_timestamp)=2017 then
order_id end)) as `Monthly Orders in 2017`, count((case
when year(order_purchase_timestamp)=2018
then order_id end)) as `Monthly Orders in 2018` from orders
group by Month,Mno order by Mno ;

**Output** :

```
select month(order_purchase_timestamp) as Mno,
        date_format(order_purchase_timestamp,'%M') as `Month`,
        count((case when year(order_purchase_timestamp)=2016 then order_id end)) as `Monthly Orders in 2016`,
        count((case when year(order_purchase_timestamp)=2017 then order_id end)) as `Monthly Orders in 2017`,
        count((case when year(order_purchase_timestamp)=2018 then order_id end)) as `Monthly Orders in 2018`
from orders
```

Result Grid

| Mno | Month | Monthly Orders in 2016 | Monthly Orders in 2017 | Monthly Orders in 2018 |
|---|---|---|---|---|
| 1 | January | 0 | 800 | 7269 |
| 2 | February | 0 | 1780 | 6728 |
| 3 | March | 0 | 2682 | 7211 |
| 4 | April | 0 | 2404 | 6939 |
| 5 | May | 0 | 3700 | 6873 |
| 6 | June | 0 | 3245 | 6167 |
| 7 | July | 0 | 4026 | 6292 |
| 8 | August | 0 | 4331 | 6512 |
| 9 | September | 4 | 4285 | 16 |
| 10 | October | 324 | 4631 | 4 |
| 11 | November | 0 | 7544 | 0 |
| 12 | December | 1 | 5673 | 0 |

**Approach** 2)

**Query** :

```
with cte2016 as (
        select extract(year from order_purchase_timestamp) as`YEAR`,
date_format(order_purchase_timestamp,'%M') as m,
count(distinct order_id) as `O16`        from orders
        where extract(year from order_purchase_timestamp)=2016
group by `YEAR`,m,extract(month from order_purchase_timestamp)
order by `YEAR`,extract(month from order_purchase_timestamp) ), cte2017
as(
        select extract(year from order_purchase_timestamp) as`YEAR`,
date_format(order_purchase_timestamp,'%M') as m,
count(distinct order_id) as `O17`        from orders
        where extract(year from order_purchase_timestamp)=2017
group by `YEAR`,m,extract(month from order_purchase_timestamp)        order
by `YEAR`,extract(month from order_purchase_timestamp)),        cte2018 as (
select extract(year from order_purchase_timestamp) as`YEAR`,
date_format(order_purchase_timestamp,'%M') as m,        count(distinct
order_id) as `O18` from orders        where extract(year from
order_purchase_timestamp)=2018        group by `YEAR`,m,extract(month from
order_purchase_timestamp)        order by `YEAR`,extract(month from
order_purchase_timestamp))
        select cte2017.m as Month,ifnull(cte2016.O16,0) as `Monthly Orders in 2016`,
        ifnull(cte2017.O17,0) as `Monthly Orders in 2017`,
ifnull(cte2018.O18,0) as `Monthly Orders in 2018`        from cte2016
right join cte2017        on cte2016.m=cte2017.m left join cte2018 on
cte2017.m=cte2018.m;
```

**Output** :

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Month | Monthly Orders in 2016 | Monthly Orders in 2017 | Monthly Orders in 2018 |
|---|---|---|---|
| January | 0 | 800 | 7269 |
| February | 0 | 1780 | 6728 |
| March | 0 | 2682 | 7211 |
| April | 0 | 2404 | 6939 |
| May | 0 | 3700 | 6873 |
| June | 0 | 3245 | 6167 |
| July | 0 | 4026 | 6292 |
| August | 0 | 4331 | 6512 |
| September | 4 | 4285 | 16 |
| October | 324 | 4631 | 4 |
| November | 0 | 7544 | 0 |
| December | 1 | 5673 | 0 |

Result 13

**Insights**: We could observe that in 2017 there was an increase in the no. of orders placed as the festive season approaches (Christmas,New Year) and the flow continues till 2018 Jan.

In 2017, from July onwards the no. of orders have increased and are above the average sales of that year, wherein all the sales in 2018 are above the average(excluding the missing/incomplete data from September 2019).

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

**Approach**: Assuming that the data/columns (order_purchase_timestamp, etc) in the dataset are in accordance to the Brasilia Standard Time.

**Query** :
```
select case when hour(order_purchase_timestamp)  between 00 and 6
then 'Dawn'
                    when hour(order_purchase_timestamp)  between 7 and 12
                    then 'Mornings'                    when
hour(order_purchase_timestamp)  between 13 and 18                    then
'Afternoon'                    when hour(order_purchase_timestamp)
between 19 and 23                    then 'Night'
                end as `Time Of The Day`,count(order_id) as `No Of Orders`
from orders
        group by `Time Of The Day`
        order by `No Of Orders` desc;
```

**Output** :

```sql
#select * from orders;
select case when hour(order_purchase_timestamp)  between 00 and 6 then 'Dawn'
            when hour(order_purchase_timestamp)  between 7 and 12 then 'Mornings'
            when hour(order_purchase_timestamp)  between 13 and 18 then 'Afternoon'
            when hour(order_purchase_timestamp)  between 19 and 23 then 'Night'
end as `Time Of The Day`,count(order_id) as `No Of Orders`
from orders group by `Time Of The Day` order by `No Of Orders` desc;
```

| Time Of The Day | No Of Orders |
|---|---|
| Afternoon | 38135 |
| Night | 28331 |
| Mornings | 27733 |
| Dawn | 5242 |

**Insights** : The Brazilian customers prefer placing orders mostly during afternoon followed by Night and Mornings.

Recommendations : Xyz could provide offers during the peak hours (Afternoon,Night) to increase sales.

---

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

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

**Solution**

**Query** : select month(order_purchase_timestamp) as Mno,customer_state, date_format(order_purchase_timestamp,'%M') as `Month`, count(case when year(order_purchase_timestamp)=2016 then order_id end) as `Order Placed in 2016`, count(case when year(order_purchase_timestamp)=2017 then order_id end) as `Order Placed in 2017`, count(case when year(order_purchase_timestamp)=2018 then order_id end) as `Order Placed in 2018` from customers join orders using (customer_id) group by customer_state,`Month`,Mno order by customer_state,Mno

**Output**:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Mno | customer_state | Month | Order Placed in 2016 | Order Placed in 2017 | Order Placed in 2018 |
|---|---|---|---|---|---|
| 1 | AC | January | 0 | 2 | 6 |
| 2 | AC | February | 0 | 3 | 3 |
| 3 | AC | March | 0 | 2 | 2 |
| 4 | AC | April | 0 | 5 | 4 |
| 5 | AC | May | 0 | 8 | 2 |
| 6 | AC | June | 0 | 4 | 3 |
| 7 | AC | July | 0 | 5 | 4 |
| 8 | AC | August | 0 | 4 | 3 |
| 9 | AC | September | 0 | 5 | 0 |
| 10 | AC | October | 0 | 6 | 0 |
| 11 | AC | November | 0 | 5 | 0 |
| 12 | AC | December | 0 | 5 | 0 |

Result 37 | Read Only | Context Help | Snippets

Output

Action Output

| # | Time | Action | Message | Duration / Fetch |
|---|---|---|---|---|
| 52 | 08:38:36 | select month(order_purchase_timestamp) as Mno,customer_state,count(order_id) as 'Order Placed',date_f... | 322 row(s) returned | 0.594 sec / 0.000 sec |
| 53 | 08:38:56 | select month(order_purchase_timestamp) as Mno,customer_state,count(order_id) as 'Order Placed',date_f... | 322 row(s) returned | 0.641 sec / 0.000 sec |
| 54 | 09:02:50 | select month(order_purchase_timestamp) as Mno,customer_state,date_format(order_purchase_timestamp,'... | 322 row(s) returned | 0.687 sec / 0.000 sec |
| 55 | 09:06:35 | select month(order_purchase_timestamp) as Mno,customer_state,date_format(order_purchase_timestamp,'... | 322 row(s) returned | 0.703 sec / 0.000 sec |

**Insights**: In states (code) like AC,RR,AL AM, etc; the sales (no of order placed) is observed to be low as compared to SP (one of the developed states in Brazil).

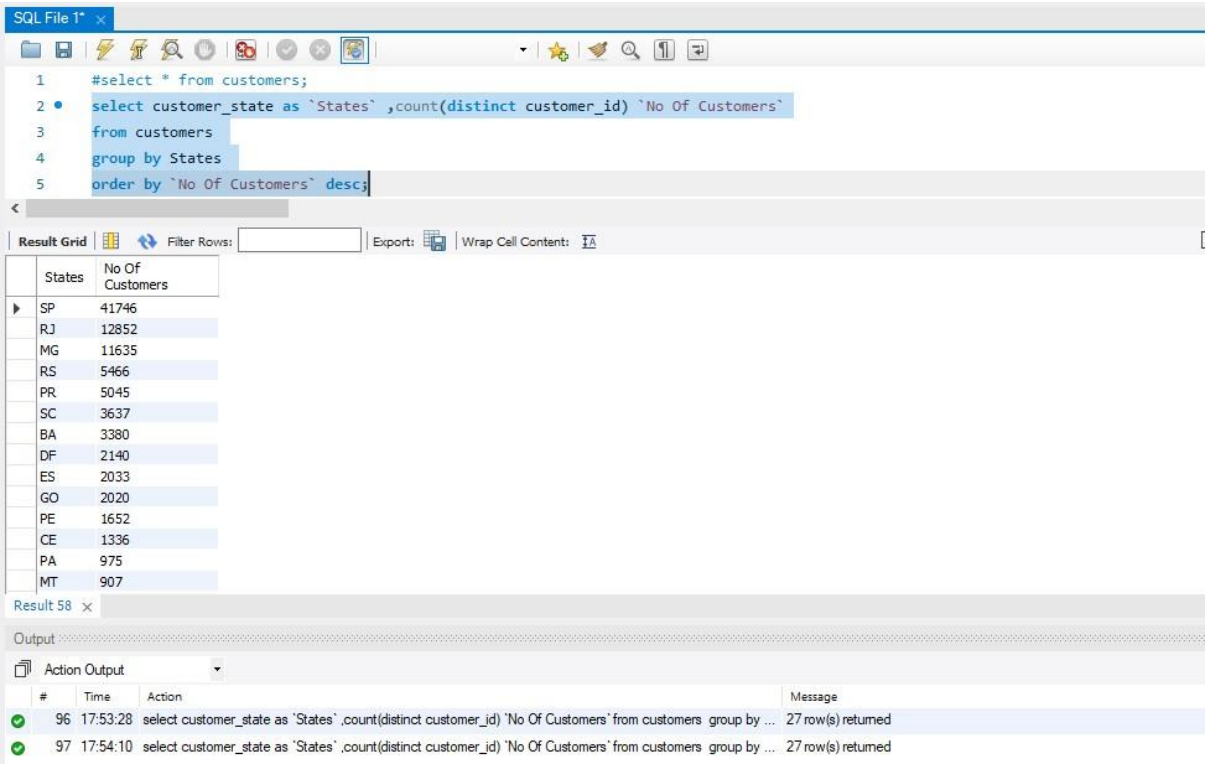
Recommendations : Xyz can revise its pricing strategies, provide more offers, sell products in combo offers to engage more potential customers from states with low sales. The states with less sales are seemingly developing states.

**2)** How are the customers distributed across all the states?

**Solution**

**Query**: select customer_state as `States` ,
count(distinct customer_id) `No Of Customers`
from customers group by States order by
`No Of Customers` desc;

**Output**:




**Insights**: In states (code) like AC,RR,AL AM, etc; the customers are observed to be low which in turn has resulted in low no. of orders placed for these states.

**Recommendations**: Xyz can revise its pricing strategies, provide more offers, sell products in combo offers to engage more potential customers from states with low customers. The aforementioned states are seemingly developing states.

---

**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).

**Solution**:

**Query**: select *,round((((`2018`-`2017`)/`2017`)*100,2) as `Increase Precentage` from

```
(select round(sum(case when Year(order_purchase_timestamp)=2017
                then payment_value end),2) as `2017`,
        round(sum(case when Year(order_purchase_timestamp)=2018
                then payment_value end),2) as `2018`
from orders join payments using (order_id)
```

where month(order_purchase_timestamp) between 1 and 8) as t;

**Output**:

```
select *,round((((`2018`-`2017`)/`2017`)*100,2) as `Increase Precentage` from
  (select round(sum(case when Year(order_purchase_timestamp)=2017 then payment_value end),2) as `2017`,
          round(sum(case when Year(order_purchase_timestamp)=2018 then payment_value end),2) as `2018`
   from orders join payments using (order_id)
   where month(order_purchase_timestamp) between 1 and 8) as t;
```

| 2017 | 2018 | Increase Precentage |
|---|---|---|
| 3669022.12 | 8694733.84 | 136.98 |

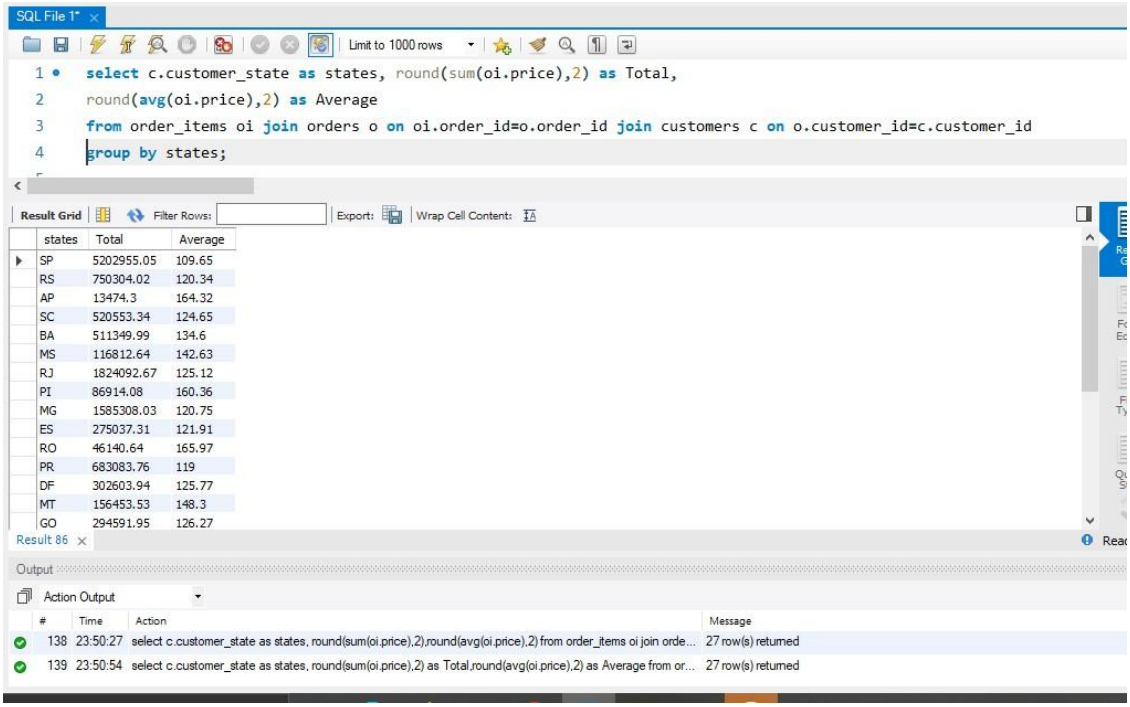
**Insights** : There is an increase of 136.98% cost of order from 2017 to 2018 (Jan - Aug)

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

**Solution** :

**Query** : select c.customer_state as states,
round(sum(oi.price),2) as Total,
round(avg(oi.price),2) as Average from
order_items oi join orders o on
oi.order_id=o.order_id join customers c on
o.customer_id=c.customer_id group by states;

**Output**:




| states | Total | Average |
|---|---|---|
| SP | 5202955.05 | 109.65 |
| RS | 750304.02 | 120.34 |
| AP | 13474.3 | 164.32 |
| SC | 520553.34 | 124.65 |
| BA | 511349.99 | 134.6 |
| MS | 116812.64 | 142.63 |
| RJ | 1824092.67 | 125.12 |
| PI | 86914.08 | 160.36 |
| MG | 1585308.03 | 120.75 |
| ES | 275037.31 | 121.91 |
| RO | 46140.64 | 165.97 |
| PR | 683083.76 | 119 |
| DF | 302603.94 | 125.77 |
| MT | 156453.53 | 148.3 |
| GO | 294591.95 | 126.27 |

**Insights**: States like RR,AP,AC,AM are observed to have low total order price and states such as SP,RJ,MG etc have high total order in comparison.

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

**Solution**

**Query** : select c.customer_state as states,
round(sum(oi.freight_value),2) as Total,
round(avg(oi.freight_value),2) as Average from
order_items oi join orders o on
oi.order_id=o.order_id join customers c on
o.customer_id=c.customer_id group by states;

**Output**:

```
select c.customer_state as states, round(sum(oi.freight_value),2) as Total,
round(avg(oi.freight_value),2) as Average
from order_items oi join orders o on oi.order_id=o.order_id join customers c on o.customer_id=c.customer_id
group by states;
```

| states | Total | Average |
|---|---|---|
| SP | 718723.07 | 15.15 |
| RS | 135522.74 | 21.74 |
| AP | 2788.5 | 34.01 |
| SC | 89660.26 | 21.47 |
| BA | 100156.68 | 26.36 |
| MS | 19144.03 | 23.37 |
| RJ | 305589.31 | 20.96 |
| PI | 21218.2 | 39.15 |
| MG | 270853.46 | 20.63 |
| ES | 49764.6 | 22.06 |
| RO | 11417.38 | 41.07 |
| PR | 117851.68 | 20.53 |
| DF | 50625.5 | 21.04 |
| MT | 29715.43 | 28.17 |
| GO | 53114.98 | 22.77 |

Result 88

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 141 | 23:57:58 | select c.customer_state as states, round(sum(oi.freight_value),2) as Total, round(avg(oi.freight_value),2) as... | 27 row(s) returned |
| 142 | 23:58:01 | select c.customer_state as states, round(sum(oi.freight_value),2) as Total, round(avg(oi.freight_value),2) as... | 27 row(s) returned |

**Insights**: States like RR,AP,AC,AM are observed to have low total freight and states such as SP,RJ,MG etc have high total freight value in comparison.
It is also observed that the **average freight value is higher in the states with lower total freight value** (RR,AP,AC,AM)**, the delivery charges seem to be bit higher for these states than other states.**

**Recommendation:** The delivery charges could be reduced/ optimised in the states such as (RR,AP,AC,AM)

---

**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.

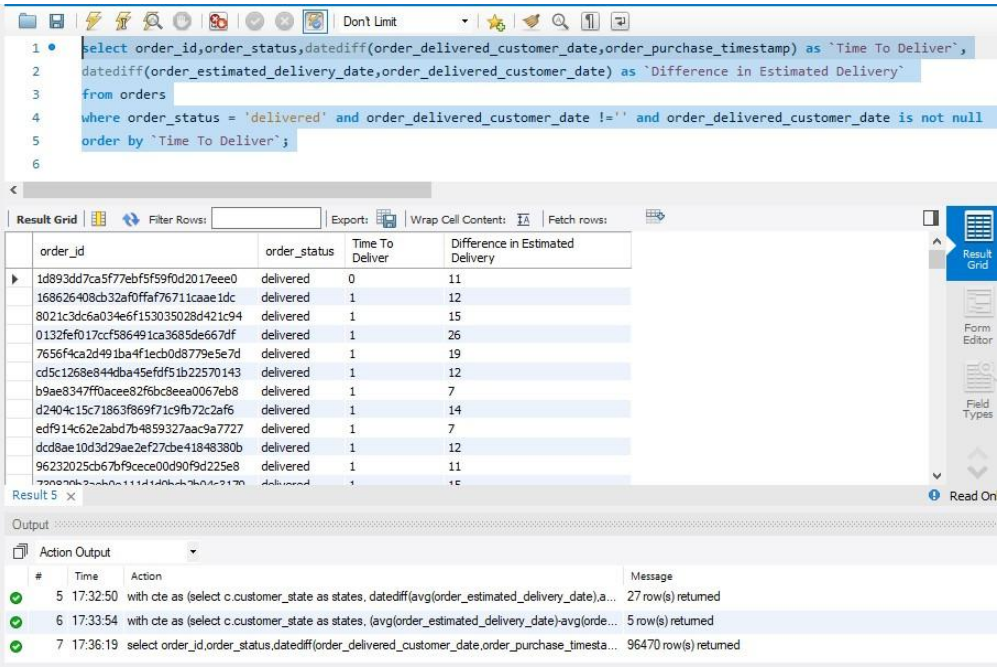
**Solution** :

**Query** :

```
select order_id,order_status,
        datediff(
order_delivered_customer_date,order_purchase_timestamp)
as `Time To Deliver`,
        datediff(
                order_estimated_delivery_date,
```

order_delivered_customer_date) as `Difference in Estimated Delivery` from orders where order_status = 'delivered' and order_delivered_customer_date !='' and order_delivered_customer_date is not null order by `Time To Deliver`;

**Output** :



```
select order_id,order_status,datediff(order_delivered_customer_date,order_purchase_timestamp) as `Time To Deliver`,
datediff(order_estimated_delivery_date,order_delivered_customer_date) as `Difference in Estimated Delivery`
from orders
where order_status = 'delivered' and order_delivered_customer_date !='' and order_delivered_customer_date is not null
order by `Time To Deliver`;
```

| order_id | order_status | Time To Deliver | Difference in Estimated Delivery |
|---|---|---|---|
| 1d893dd7ca5f77ebf5f59f0d2017eee0 | delivered | 0 | 11 |
| 168626408cb32af0ffaf76711caae1dc | delivered | 1 | 12 |
| 8021c3dc6a034e6f153035028d421c94 | delivered | 1 | 15 |
| 0132fef017ccf586491ca3685de667df | delivered | 1 | 26 |
| 7656f4ca2d491ba4f1ecb0d8779e5e7d | delivered | 1 | 19 |
| cd5c1268e844dba45efdf51b22570143 | delivered | 1 | 12 |
| b9ae8347ff0acee82f6bc8eea0067eb8 | delivered | 1 | 7 |
| d2404c15c71863f869f71c9fb72c2af6 | delivered | 1 | 14 |
| edf914c62e2abd7b4859327aac9a7727 | delivered | 1 | 7 |
| dcd8ae10d3d29ae2ef27cbe41848380b | delivered | 1 | 12 |
| 96232025cb67bf9cece00d90f9d225e8 | delivered | 1 | 11 |

| # | Time | Action | Message |
|---|---|---|---|
| 5 | 17:32:50 | with cte as (select c.customer_state as states, datediff(avg(order_estimated_delivery_date),a... | 27 row(s) returned |
| 6 | 17:33:54 | with cte as (select c.customer_state as states, (avg(order_estimated_delivery_date)-avg(orde... | 5 row(s) returned |
| 7 | 17:36:19 | select order_id,order_status,datediff(order_delivered_customer_date,order_purchase_timesta... | 96470 row(s) returned |

#sorted using `Difference in estimated delivery` column

**Query** : select order_id,order_status,
datediff(
order_delivered_customer_date,order_purchase_timestamp)
as `Time To Deliver`,
datediff(
order_estimated_delivery_date,
order_delivered_customer_date)
as `Difference in Estimated Delivery`
from orders
where order_status = 'delivered' and order_delivered_customer_date !='' and order_delivered_customer_date is not null order by `Difference in estimated delivery`;

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

| order_id | order_status | Time To Deliver | Difference in Estimated Delivery |
|---|---|---|---|
| 1b3190b2dfa9d789e1f14c05b647a14a | delivered | 208 | -188 |
| ca07593549f1816d26a572e06dc1eab6 | delivered | 210 | -181 |
| 47b40429ed8cce3aee9199792275433f | delivered | 191 | -175 |
| 2fe324febf907e3ea3f2aa9650869fa5 | delivered | 190 | -167 |
| 285ab9426d6982034523a855f55a885e | delivered | 195 | -166 |
| 440d0d17af552815d15a9e41abe49359 | delivered | 196 | -165 |
| c27815f7e3dd0b926b58552628481575 | delivered | 188 | -162 |
| d24e8541128cea179a11a65176e0a96f | delivered | 175 | -161 |
| 0f4519c5f1c541ddec9f21b3bddd533a | delivered | 194 | -161 |
| 2d7561026d542c8dbd8f0daeadf67a43 | delivered | 188 | -159 |
| 2fb597c2f772eca01b1f5c561bf6cc7b | delivered | 195 | -155 |
| 6e82dcfb5eada6283dba34f164e636f5 | delivered | 183 | -155 |
| ed8e9faf1b75f43ee027103957135663 | delivered | 173 | -153 |
| dfe5f68118c2576143240b8d78e5940a | delivered | 186 | -153 |
| 2ba1366baecad3c3536f27546d129017 | delivered | 181 | -152 |
| 437222e3fd1b07396f1d9ba8c15fba59 | delivered | 187 | -144 |
| 6e6527028de694ccade37f5a15a6d84a | delivered | 166 | -143 |
| a452fba32eab28a4a62af18eed010c0b | delivered | 168 | -138 |

Result 23

**Insights** : When the resulted are sorted using `Difference in estimated delivery` column, the negative figure represents the delay in delivering the products, the maximum delay is of 188 days. There are around 4000 + such order that has been delayed for more than 4 days till 188 days.

The positive figures in `Difference in estimated delivery` columns represents the early delivery of products.

**Recommendations** : The Estimated Delivery Date needs to be optimised to suit the delayed/early delivery of the products.

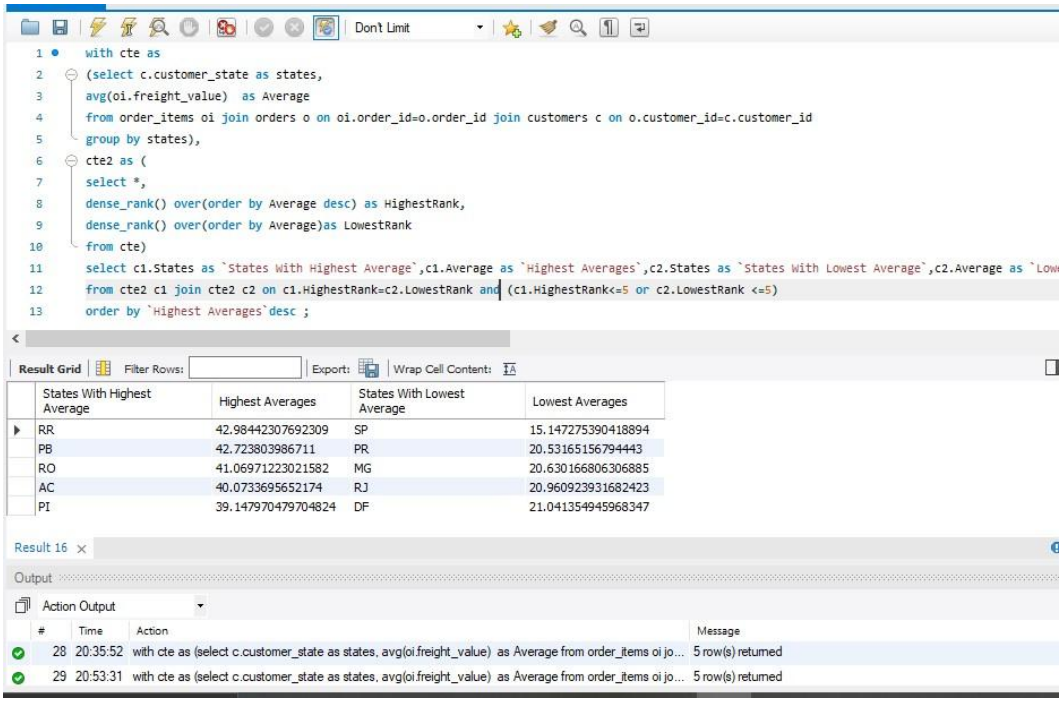
**2)** Find out the top 5 states with the highest & lowest average freight value.

**Solution**:

**Query** :
```
with cte as
(select c.customer_state as states,
avg(oi.freight_value) as Average from
order_items oi join orders o on
oi.order_id=o.order_id join customers c on
o.customer_id=c.customer_id
group by states),
cte2 as ( select *,
dense_rank() over(order by Average desc) as HighestRank,
dense_rank() over(order by Average)as LowestRank from cte)
select c1.States as `States With Highest Average`,
c1.Average as `Highest Averages`, c2.States as `States
With Lowest Average`, c2.Average as `Lowest Averages`
```

from cte2 c1 join cte2 c2 on c1.HighestRank=c2.LowestRank
and (c1.HighestRank<=5 or c2.LowestRank <=5)
order by `Highest Averages`desc ;

**Output** :



```
with cte as
(select c.customer_state as states,
avg(oi.freight_value)  as Average
from order_items oi join orders o on oi.order_id=o.order_id join customers c on o.customer_id=c.customer_id
group by states),
cte2 as (
select *,
dense_rank() over(order by Average desc) as HighestRank,
dense_rank() over(order by Average)as LowestRank
from cte)
select c1.States as `States With Highest Average`,c1.Average as `Highest Averages`,c2.States as `States With Lowest Average`,c2.Average as `Low
from cte2 c1 join cte2 c2 on c1.HighestRank=c2.LowestRank and (c1.HighestRank<=5 or c2.LowestRank <=5)
order by `Highest Averages`desc ;
```

| States With Highest Average | Highest Averages | States With Lowest Average | Lowest Averages |
|---|---|---|---|
| RR | 42.98442307692309 | SP | 15.147275390418894 |
| PB | 42.723803986711 | PR | 20.53165156794443 |
| RO | 41.06971223021582 | MG | 20.630166806306885 |
| AC | 40.0733695652174 | RJ | 20.960923931682423 |
| PI | 39.147970479704824 | DF | 21.041354945968347 |

| # | Time | Action | Message |
|---|---|---|---|
| 28 | 20:35:52 | with cte as (select c.customer_state as states, avg(oi.freight_value) as Average from order_items oi jo... | 5 row(s) returned |
| 29 | 20:53:31 | with cte as (select c.customer_state as states, avg(oi.freight_value) as Average from order_items oi jo... | 5 row(s) returned |

**Recommendations** : It is clearly evident that the freight value is higher in the states RR, PB, RO, AC, which has low potential customers, low cost orders and low orders placed; A reduction in the freight value is necessary, not only that any factor that causes such elevation in the freight value needs to be considers and optimised efficiently.

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

**Solution**:

**Query** : with cte as
(select c.customer_state as states,
avg(datediff(
o.order_delivered_customer_date,
order_purchase_timestamp)) as `avgdt`
from orders o join customers c
on o.customer_id=c.customer_id
group by states), cte2 as (
select *,

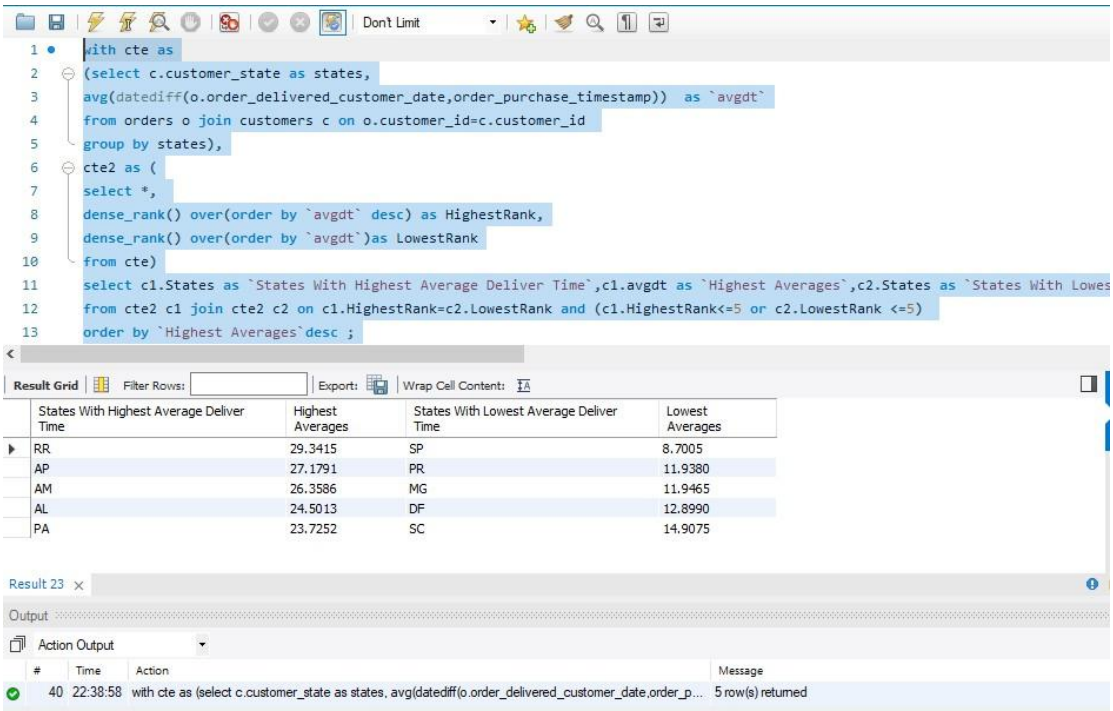
dense_rank() over(order by `avgdt` desc) as HighestRank, dense_rank() over(order by `avgdt`)as LowestRank from cte)

select c1.States as `States With Highest Average Deliver Time`, c1.avgdt as `Highest Averages`,c2.States as `States With Lowest Average Deliver Time`,

c2.avgdt as `Lowest Averages` from cte2 c1 join cte2 c2 on c1.HighestRank=c2.LowestRank and (c1.HighestRank<=5 or c2.LowestRank <=5)

order by `Highest Averages`desc ;

**Output**:




**Insights**: Similar to previous analysis, the States RR, AM etc have higher average delivery time, which is equivalent to a month to deliver the products.

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

**Solution**:

**Query:** with cte as

(select c.customer_state as states,

avg(datediff(order_estimated_delivery_date, order_delivered_customer_date)) as `avgedt` from orders o join customers c on o.customer_id=c.customer_id

group by states),

cte2 as (select *, dense_rank() over(order by `avgedt` desc) as LRank from cte) select States as `States With Fastest Delivery`,`avgedt` from cte2 where LRank<=5
order by LRank;

**Output**:

```
with cte as
(select c.customer_state as states,
avg(datediff(order_estimated_delivery_date,order_delivered_customer_date)) as `avgedt`
from orders o join customers c on o.customer_id=c.customer_id
group by states),
cte2 as (select *,
dense_rank() over(order by `avgedt` desc) as LRank
from cte)
select States as `States With Fastest Delivery`,`avgedt`
from cte2 where LRank<=5
order by LRank;
```

ult Grid | Filter Rows: | Export: | Wrap Cell Content:

| States With Fastest Delivery | avgedt |
|---|---|
| AC | 20.7250 |
| RO | 20.1029 |
| AP | 19.6866 |
| AM | 19.5655 |
| RR | 17.2927 |

ult 8 ×

put

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 9 | 17:39:17 | with cte as (select c.customer_state as states, avg(datediff(order_estimated_delivery_date,or... | 5 row(s) returned |

**Insights** : The states such like RR,AM,AP have highest average delivery time ie (the average number of days required to deliver in these states is approx. 30 days. ). Hence, it is highly possible that the delivery speed might have increased for orders

**6) Analysis based on the payments:**

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

**Solution**:

**Query**: select year(order_purchase_timestamp) as `Year`,
month(order_purchase_timestamp) as Mno,
date_format(order_purchase_timestamp, '%M') as Month,
count(distinct (case when payment_type = 'credit_card'
then order_id end)) as 'Credit Card',
count(distinct(case when payment_type = 'UPI'
then order_id end)) as 'UPI', count(distinct(case when

```
payment_type = 'voucher'                                then
order_id end)) as 'Voucher',                count(distinct(case when
payment_type = 'debit_card'
                                        then order_id end)) as 'Debit Card'
    from payments p join orders o using (order_id)
group by Year,Mno, Month        order by
Year,Mno;
```

**Output** :

```
select year(order_purchase_timestamp) as `Year`,month(order_purchase_timestamp) as Mno,
date_format(order_purchase_timestamp, '%M') as Month,
count(distinct (case when payment_type = 'credit_card' then order_id end)) as 'Credit Card',
count(distinct(case when payment_type = 'UPI' then order_id end)) as 'UPI',
count(distinct(case when payment_type = 'voucher' then order_id end)) as 'Voucher',
count(distinct(case when payment_type = 'debit_card' then order_id end)) as 'Debit Card'
from payments p join orders o using (order_id)
group by Year,Mno, Month
order by Year,Mno;
```

Result Grid

| Year | Mno | Month | Credit Card | UPI | Voucher | Debit Card |
|---|---|---|---|---|---|---|
| 2016 | 9 | September | 3 | 0 | 0 | 0 |
| 2016 | 10 | October | 253 | 63 | 11 | 2 |
| 2016 | 12 | December | 1 | 0 | 0 | 0 |
| 2017 | 1 | January | 582 | 197 | 33 | 9 |
| 2017 | 2 | February | 1347 | 398 | 69 | 13 |
| 2017 | 3 | March | 2008 | 590 | 123 | 31 |
| 2017 | 4 | April | 1835 | 496 | 115 | 27 |
| 2017 | 5 | May | 2833 | 772 | 171 | 30 |
| 2017 | 6 | June | 2452 | 707 | 142 | 27 |
| 2017 | 7 | July | 3072 | 845 | 205 | 22 |

Result 7

Output

Action Output

| # | Time | Action | Message |
|---|---|---|---|
| 10 | 09:09:35 | select year(order_purchase_timestamp) as `Year`,month(order_purchase_timestamp) as Mno, date_format... | 25 row(s) returned |

**Insights**: Brazilians seems to opt for credit card and UPI payments more as compared to voucher and debit card.

The descending order of payment mode is as follows (Credit card being used the highest, UPI used as second highest and Debit card the least):
-Credit Card, UPI, Voucher, Debit Card

**2)** Find the no. of orders placed on the basis of the payment instalments that have been paid.

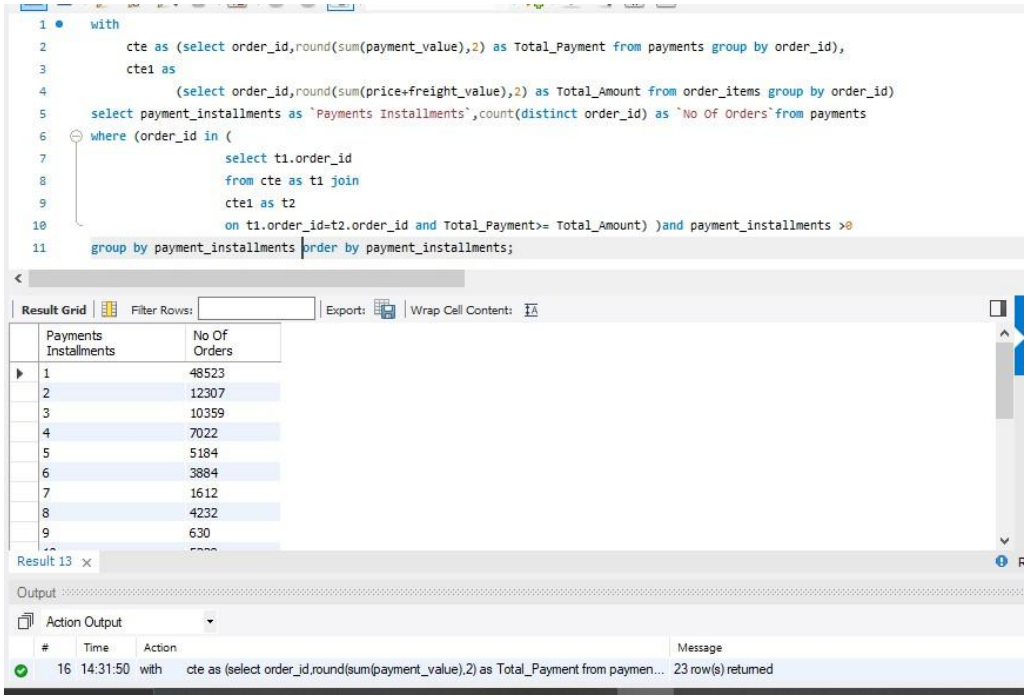
**Solution**:

**Approach** : In the first query, the order_id are fetched for which the payments are paid completely (all the installments are paid) and for the 2nd Query the order_id is fetched on the basis of atleast one installment is paid successfully.

**Approach 1 (All Paid) Query**
: #use xyz;

```
with
    cte as
        (select order_id,round(sum(payment_value),2) as Total_Payment
from payments            group by order_id),      cte1 as
         (select order_id,round(sum(price+freight_value),2) as Total_Amount
from order_items group by order_id) select payment_installments as
`Payments Installments`,          count(distinct order_id) as `No Of
Orders`from payments where (order_id in (
                select t1.order_id
from cte as t1 join
cte1 as t2
                on t1.order_id=t2.order_id and Total_Payment>= Total_Amount) )and
payment_installments >0 group by payment_installments
order by payment_installments;
```

**Output**:

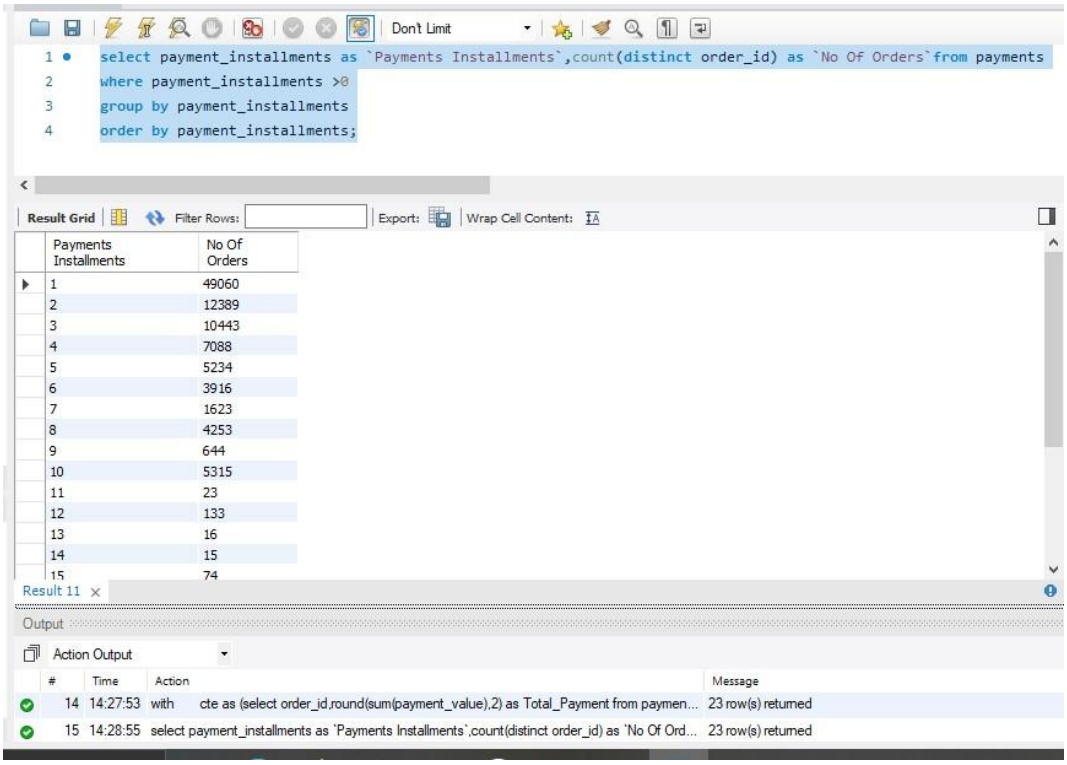



**Approach 2 (Atleast one installment paid):**

**Query** : select payment_installments as `Payments Installments`, count(distinct order_id) as `No Of Orders`from payments

where payment_installments >0 group by payment_installments
order by payment_installments;

**Output** :




| Payments Installments | No Of Orders |
|---|---|
| 1 | 49060 |
| 2 | 12389 |
| 3 | 10443 |
| 4 | 7088 |
| 5 | 5234 |
| 6 | 3916 |
| 7 | 1623 |
| 8 | 4253 |
| 9 | 644 |
| 10 | 5315 |
| 11 | 23 |
| 12 | 133 |
| 13 | 16 |
| 14 | 15 |
| 15 | 74 |

**Insights** : In both the cases, majority of customers have opted for 1 installments and no of orders being proportional to the no of installments; the less the installments the less customer have opted for it.