New Wheels Project

Introduction to SQL

Problem Statement

Business Context

A lot of people in the world share a common desire: to own a vehicle. A car or an automobile is seen as an object that gives the freedom of mobility. Many now prefer pre-owned vehicles because they come at an affordable cost, but at the same time, they are also concerned about whether the after-sales service provided by the resale vendors is as good as the care you may get from the actual manufacturers.

New-Wheels, a vehicle resale company, has launched an app with an end-to-end service from listing the vehicle on the platform to shipping it to the customer's location. This app also captures the overall after-sales feedback given by the customer.

Objective

New-Wheels sales have been dipping steadily in the past year, and due to the critical customer feedback and ratings online, there has been a drop in new customers every quarter, which is concerning to the business. The CEO of the company now wants a quarterly report with all the key metrics sent to him so he can assess the health of the business and make the necessary decisions.

As a data analyst, you see that there is an array of questions that are being asked at the leadership level that need to be answered using data. Import the dump file that contains various tables that are present in the database. Use the data to answer the questions posed and create a quarterly business report for the CEO.

Business Questions

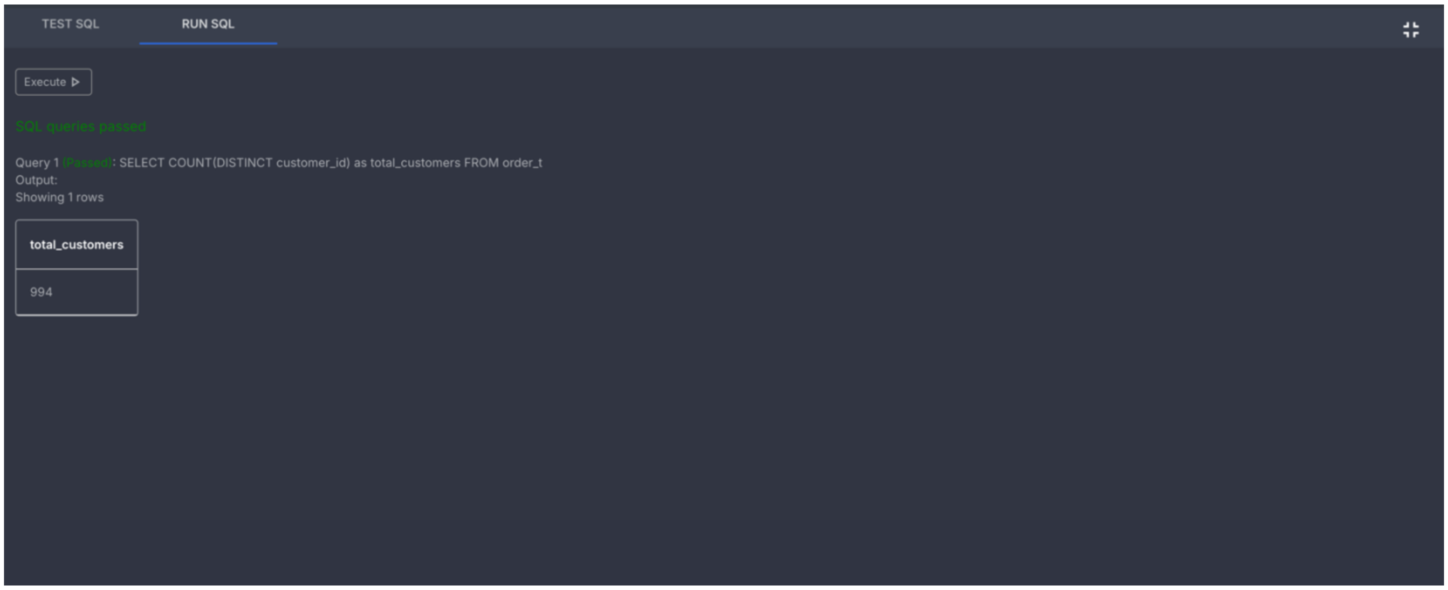
Question 1: Find the total number of customers who have placed orders. What is the distribution of the customers across states?

Solution Query:

Total no. of customers who placed orders:

```
SELECT COUNT(DISTINCT customer_id)
FROM   order_t;
```

Output:



TEST SQL RUN SQL

Execute

SQL queries passed

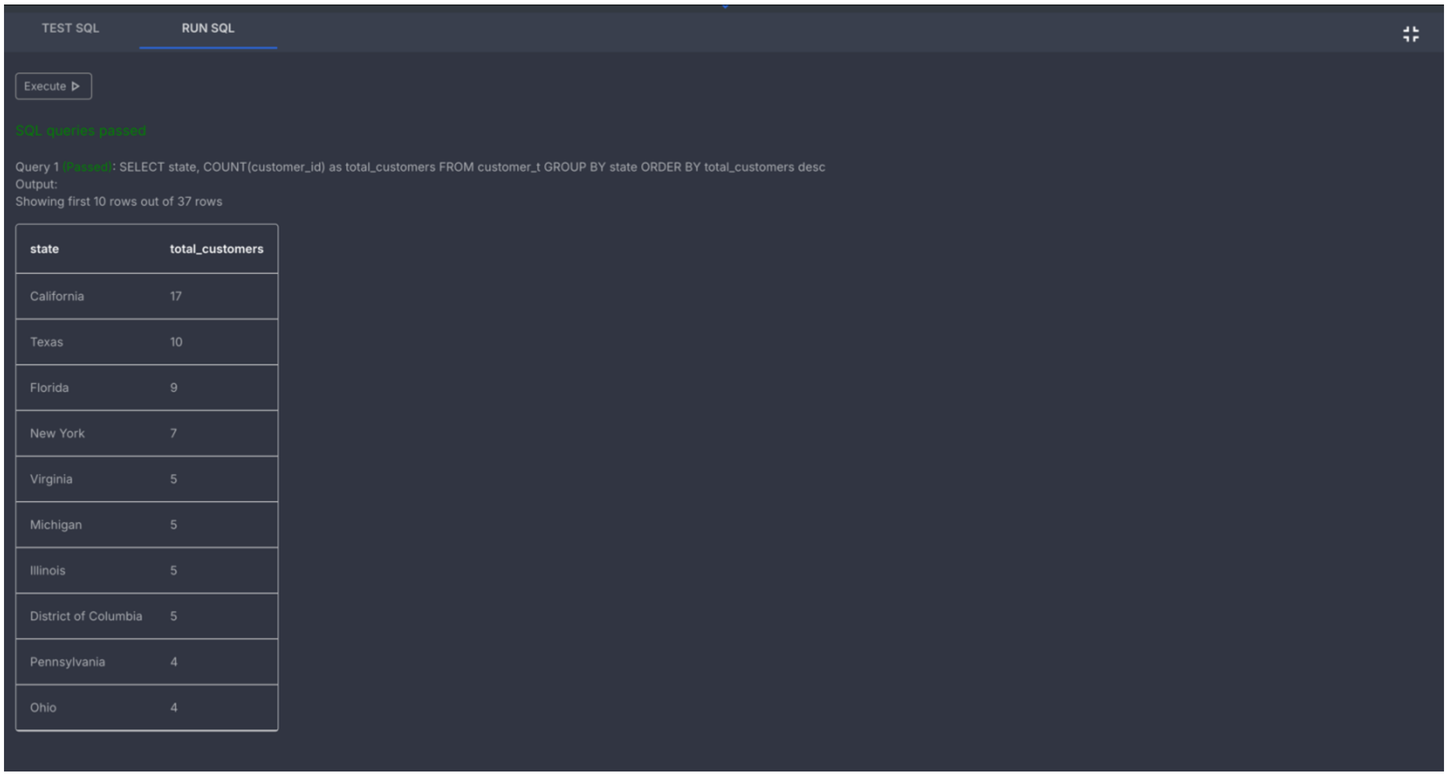
```
Query 1 (Passed): SELECT COUNT(DISTINCT customer_id) as total_customers FROM order_t
Output:
Showing 1 rows
```

total_customers
994

Distribution of customers across states:

```
SELECT
    state,
    COUNT(customer_id) AS total_customers
FROM   customer_t
GROUP  BY state
ORDER  BY total_customers DESC;
```

Output:



```
SQL queries passed

Query 1 (Passed): SELECT state, COUNT(customer_id) as total_customers FROM customer_t GROUP BY state ORDER BY total_customers desc
Output:
Showing first 10 rows out of 37 rows
```

state	total_customers
California	17
Texas	10
Florida	9
New York	7
Virginia	5
Michigan	5
Illinois	5
District of Columbia	5
Pennsylvania	4
Ohio	4

Observations and Insights:

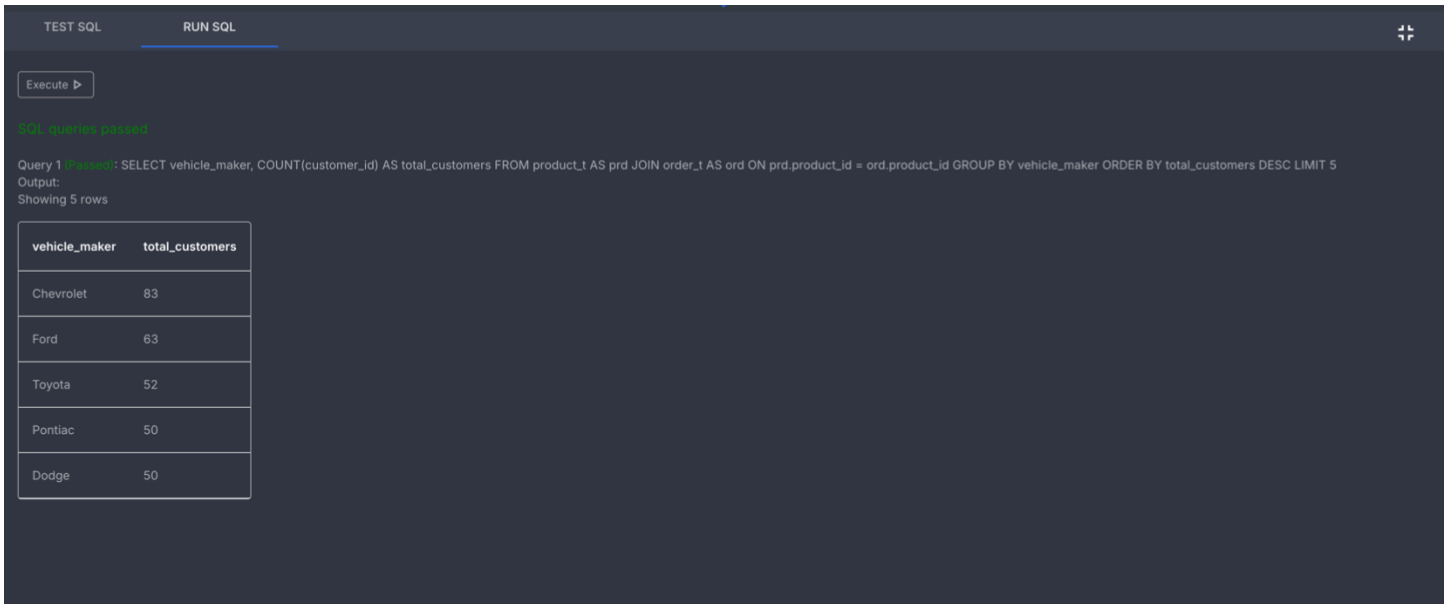
- There are a total of 994 customers who placed orders
- Among the customers, most of them reside in the states of California, Texas, Florida, New York and Virginia

Question 2: Which are the top 5 vehicle makers preferred by the customers?

Solution Query:

```
SELECT
            vehicle_maker,
            COUNT(customer_id) AS customer_count
FROM   product_t AS prd
JOIN order_t AS ord ON prd.product_id = ord.product_id
GROUP  BY vehicle_maker
ORDER  BY customer_count DESC
LIMIT  5;
```

Output:



```
SQL queries passed

Query 1 (Passed): SELECT vehicle_maker, COUNT(customer_id) AS total_customers FROM product_t AS prd JOIN order_t AS ord ON prd.product_id = ord.product_id GROUP BY vehicle_maker ORDER BY total_customers DESC LIMIT 5
Output:
Showing 5 rows
```

vehicle_maker	total_customers
Chevrolet	83
Ford	63
Toyota	52
Pontiac	50
Dodge	50

Observations and Insights:

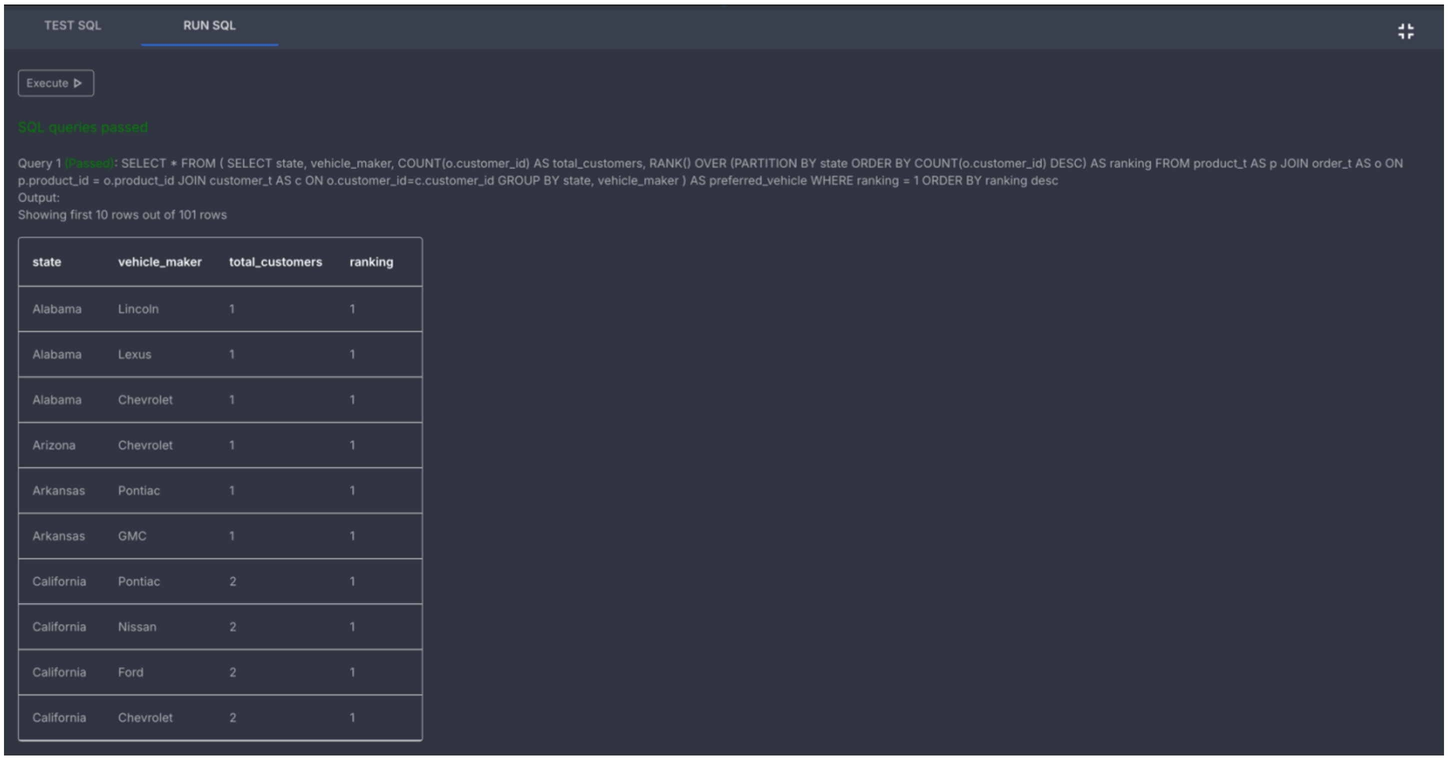
- Among the different vehicle makers, the top 5 vehicle makers preferred by customers are Chevrolet, Ford, Toyota, Pontiac and Dodge
- Chevrolet is the most preferred vehicle maker

Question 3: Which is the most preferred vehicle maker in each state?

Solution Query:

```
SELECT *
FROM
(
            SELECT
             state,
             vehicle_maker,
             COUNT(o.customer_id) AS total_customers,
             RANK() OVER (PARTITION BY state ORDER BY Count(o.customer_id) DESC) AS ranking
        FROM   product_t AS p
        JOIN order_t AS o ON p.product_id = o.product_id
        JOIN customer_t AS c ON o.customer_id = c.customer_id
        GROUP  BY state, vehicle_maker
) AS preferred_vehicle
WHERE  ranking = 1
ORDER  BY ranking DESC;
```

Output:



SQL queries passed

```
Query 1 (Passed): SELECT * FROM ( SELECT state, vehicle_maker, COUNT(o.customer_id) AS total_customers, RANK() OVER (PARTITION BY state ORDER BY COUNT(o.customer_id) DESC) AS ranking FROM product_t AS p JOIN order_t AS o ON p.product_id = o.product_id JOIN customer_t AS c ON o.customer_id=c.customer_id GROUP BY state, vehicle_maker ) AS preferred_vehicle WHERE ranking = 1 ORDER BY ranking desc
Output:
Showing first 10 rows out of 101 rows
```

state	vehicle_maker	total_customers	ranking
Alabama	Lincoln	1	1
Alabama	Lexus	1	1
Alabama	Chevrolet	1	1
Arizona	Chevrolet	1	1
Arkansas	Pontiac	1	1
Arkansas	GMC	1	1
California	Pontiac	2	1
California	Nissan	2	1
California	Ford	2	1
California	Chevrolet	2	1

Observations and Insights:

- Among the different states listed, Chevrolet is the most preferred vehicle maker

Question 4: Find the overall average rating given by the customers. What is the average rating in each quarter?

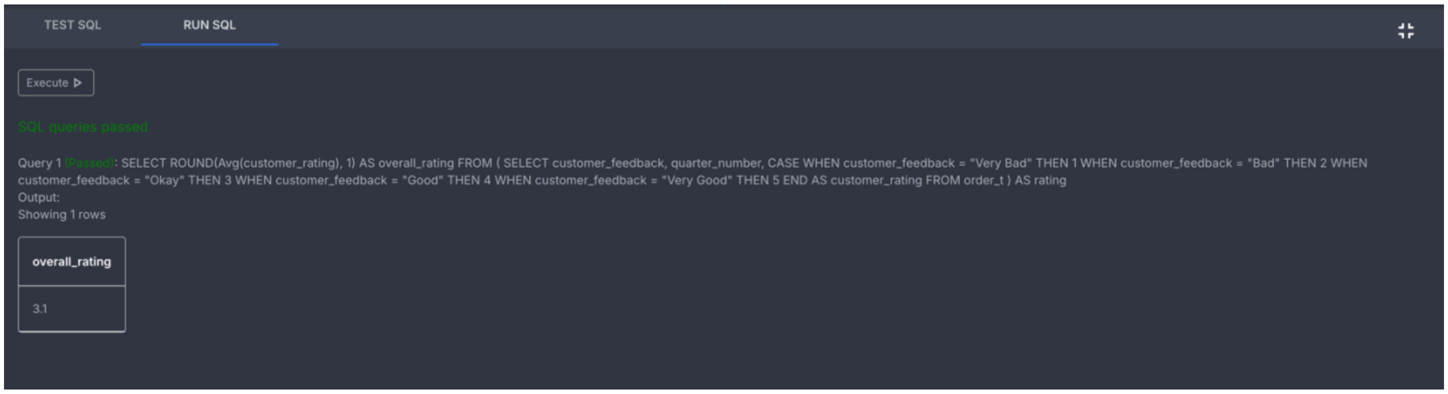
Consider the following mapping for ratings: “Very Bad”: 1, “Bad”: 2, “Okay”: 3, “Good”: 4, “Very Good”: 5

Solution Query:

Overall Average Rating Given by Customers:

```
SELECT
ROUND(AVG(customer_rating), 1) AS overall_rating
FROM
(
            SELECT
             customer_feedback,
             quarter_number,
             CASE
                    WHEN customer_feedback = "Very Bad" THEN 1
                    WHEN customer_feedback = "Bad" THEN 2
                    WHEN customer_feedback = "Okay" THEN 3
                    WHEN customer_feedback = "Good" THEN 4
                    WHEN customer_feedback = "Very Good" THEN 5
             END AS customer_rating
             FROM   order_t
) AS rating;
```

Output:



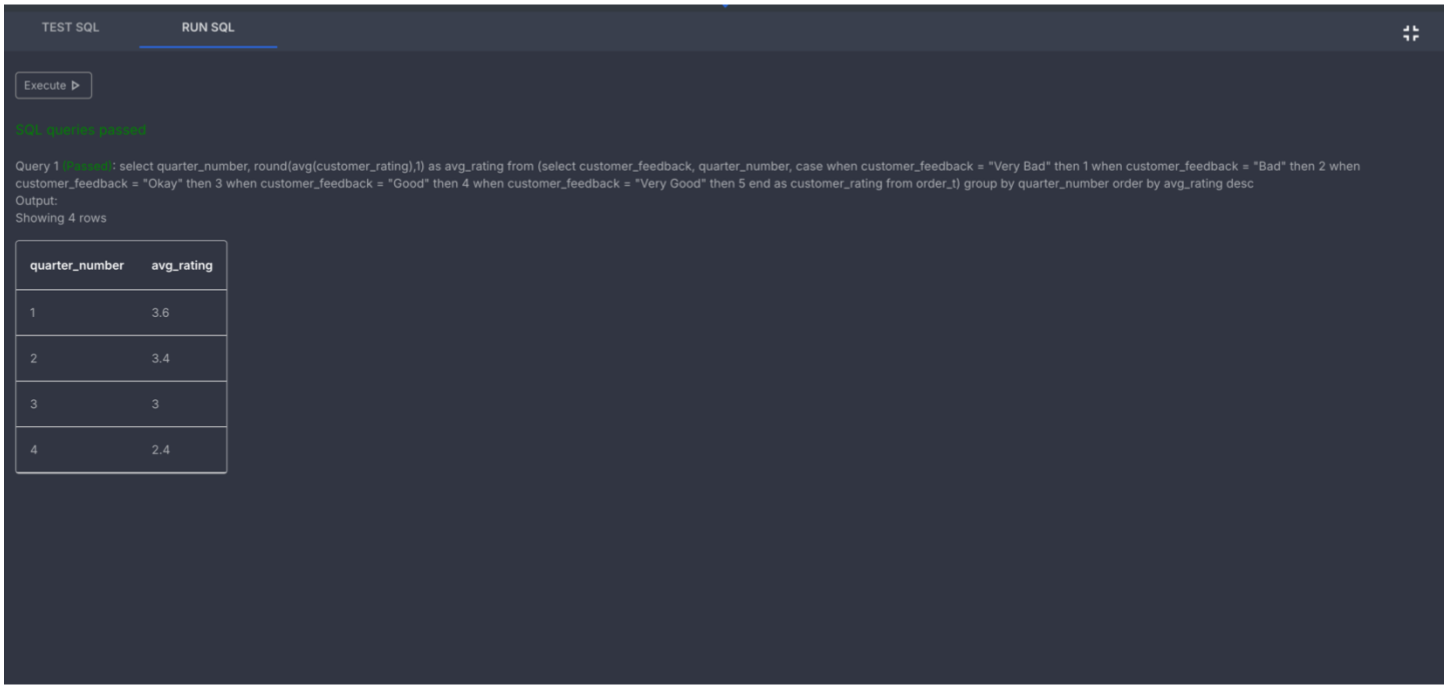
```
SQL queries passed

Query 1 (Passed): SELECT ROUND(Avg(customer_rating), 1) AS overall_rating FROM ( SELECT customer_feedback, quarter_number, CASE WHEN customer_feedback = "Very Bad" THEN 1 WHEN customer_feedback = "Bad" THEN 2 WHEN customer_feedback = "Okay" THEN 3 WHEN customer_feedback = "Good" THEN 4 WHEN customer_feedback = "Very Good" THEN 5 END AS customer_rating FROM order_t ) AS rating
Output:
Showing 1 rows
```

overall_rating
3.1

Average Rating Given in Each Quarter:

```
SELECT
   quarter_number,
   ROUND(AVG(customer_rating),1) AS avg_rating
FROM
(
   SELECT
   customer_feedback,
   quarter_number,
     CASE WHEN customer_feedback = "Very Bad" THEN 1
     WHEN customer_feedback = "Bad" THEN 2
     WHEN customer_feedback = "Okay" THEN 3
     WHEN customer_feedback = "Good" THEN 4
     WHEN customer_feedback = "Very Good" THEN 5
     END AS customer_rating
FROM order_t) AS rating
GROUP BY quarter_number
ORDER BY quarter_number ASC;
```

Output:



```
SQL queries passed

Query 1 (Passed): select quarter_number, round(avg(customer_rating),1) as avg_rating from (select customer_feedback, quarter_number, case when customer_feedback = "Very Bad" then 1 when customer_feedback = "Bad" then 2 when customer_feedback = "Okay" then 3 when customer_feedback = "Good" then 4 when customer_feedback = "Very Good" then 5 end as customer_rating from order_t) group by quarter_number order by avg_rating desc
Output:
Showing 4 rows
```

quarter_number	avg_rating
1	3.6
2	3.4
3	3
4	2.4

Observations and Insights:

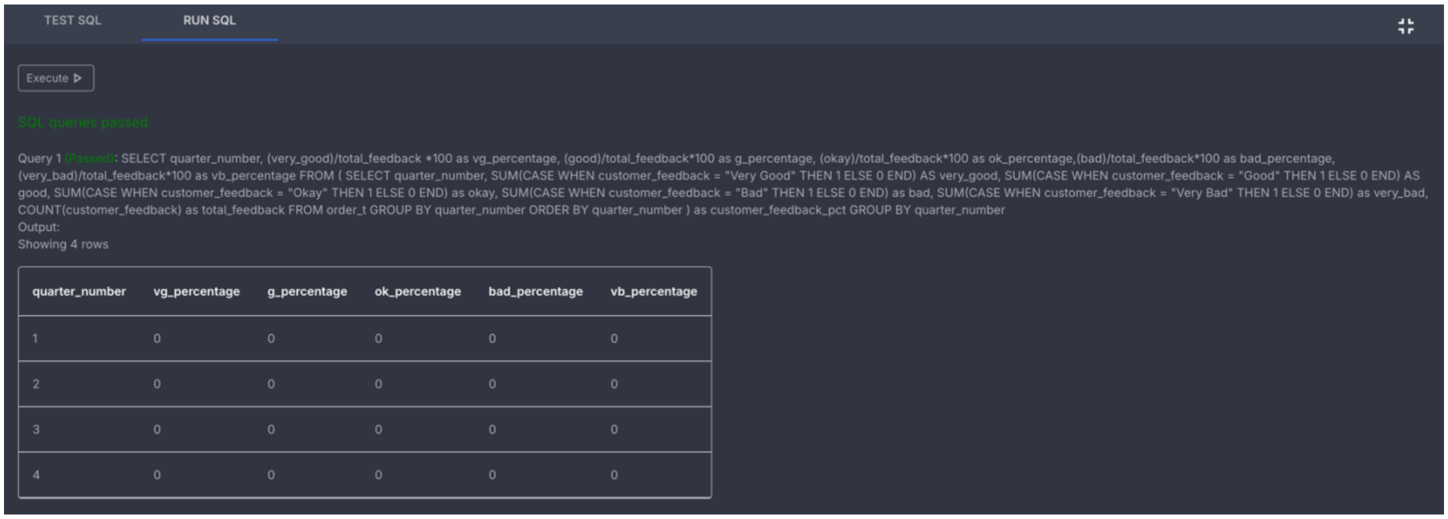
- The overall average rating given by the customers is 3.1
- Q1 has the highest average rating by the customers and Q4 has the least
- There has been approx. 33% decline in the average rating from Q1 to Q4

Question 5: Find the percentage distribution of feedback from the customers. Are customers getting more dissatisfied over time?

Solution Query using Sub-Query:

```
SELECT
   quarter_number,
   (very_good)/total_feedback *100 AS vg_percentage,
   (good)/total_feedback*100 AS g_percentage,
   (okay)/total_feedback*100 AS ok_percentage,
   (bad)/total_feedback*100 AS bad_percentage,
   (very_bad)/total_feedback*100 AS vb_percentage
FROM
(
   SELECT quarter_number,
      SUM(CASE WHEN customer_feedback = "Very Good" THEN 1 ELSE 0 END) AS  very_good,
      SUM(CASE WHEN customer_feedback = "Good" THEN 1 ELSE 0 END) AS good,
      SUM(CASE WHEN customer_feedback = "Okay" THEN 1 ELSE 0 END) AS okay,
      SUM(CASE WHEN customer_feedback = "Bad" THEN 1 ELSE 0  END) AS bad,
      SUM(CASE WHEN customer_feedback = "Very Bad" THEN 1 ELSE 0 END) AS very_bad,
      COUNT(customer_feedback) AS total_feedback
 FROM order_t
 GROUP BY quarter_number
 ORDER BY quarter_number
) AS customer_feedback_pct
GROUP BY quarter_number;
```

Output in SQLPlayground:

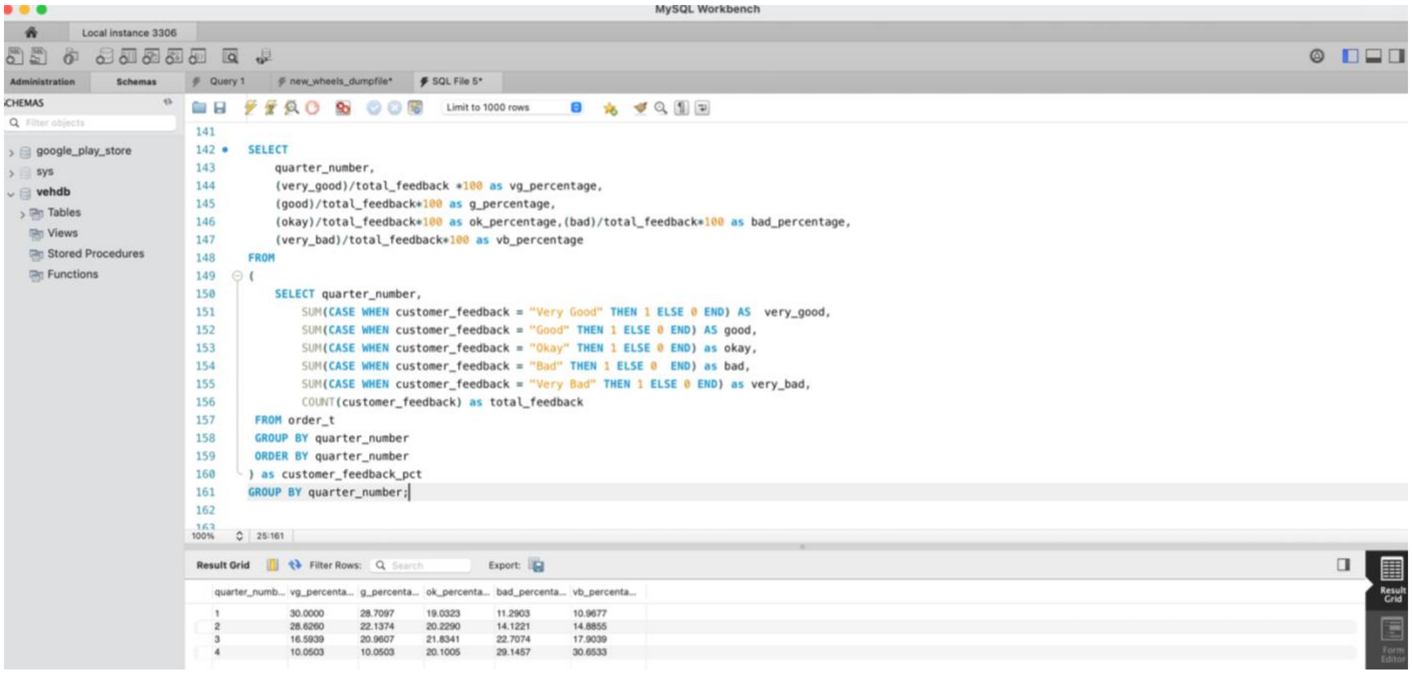


```
SQL queries passed

Query 1 (Passed): SELECT quarter_number, (very_good)/total_feedback *100 as vg_percentage, (good)/total_feedback*100 as g_percentage, (okay)/total_feedback*100 as ok_percentage,(bad)/total_feedback*100 as bad_percentage, (very_bad)/total_feedback*100 as vb_percentage FROM ( SELECT quarter_number, SUM(CASE WHEN customer_feedback = "Very Good" THEN 1 ELSE 0 END) AS very_good, SUM(CASE WHEN customer_feedback = "Good" THEN 1 ELSE 0 END) AS good, SUM(CASE WHEN customer_feedback = "Okay" THEN 1 ELSE 0 END) as okay, SUM(CASE WHEN customer_feedback = "Bad" THEN 1 ELSE 0 END) as bad, SUM(CASE WHEN customer_feedback = "Very Bad" THEN 1 ELSE 0 END) as very_bad, COUNT(customer_feedback) as total_feedback FROM order_t GROUP BY quarter_number ORDER BY quarter_number ) as customer_feedback_pct GROUP BY quarter_number
Output:
Showing 4 rows
```

quarter_number	vg_percentage	g_percentage	ok_percentage	bad_percentage	vb_percentage
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

Note: The Query ran successfully but returns all 0s as shown above in SQL Playground

Output using the same sub-query in MySQL Workbench:



```
SELECT
    quarter_number,
    (very_good)/total_feedback *100 as vg_percentage,
    (good)/total_feedback*100 as g_percentage,
    (okay)/total_feedback*100 as ok_percentage,(bad)/total_feedback*100 as bad_percentage,
    (very_bad)/total_feedback*100 as vb_percentage
FROM
(
    SELECT quarter_number,
        SUM(CASE WHEN customer_feedback = "Very Good" THEN 1 ELSE 0 END) AS  very_good,
        SUM(CASE WHEN customer_feedback = "Good" THEN 1 ELSE 0 END) AS good,
        SUM(CASE WHEN customer_feedback = "Okay" THEN 1 ELSE 0 END) as okay,
        SUM(CASE WHEN customer_feedback = "Bad" THEN 1 ELSE 0  END) as bad,
        SUM(CASE WHEN customer_feedback = "Very Bad" THEN 1 ELSE 0 END) as very_bad,
        COUNT(customer_feedback) as total_feedback
 FROM order_t
 GROUP BY quarter_number
 ORDER BY quarter_number
) as customer_feedback_pct
GROUP BY quarter_number;
```

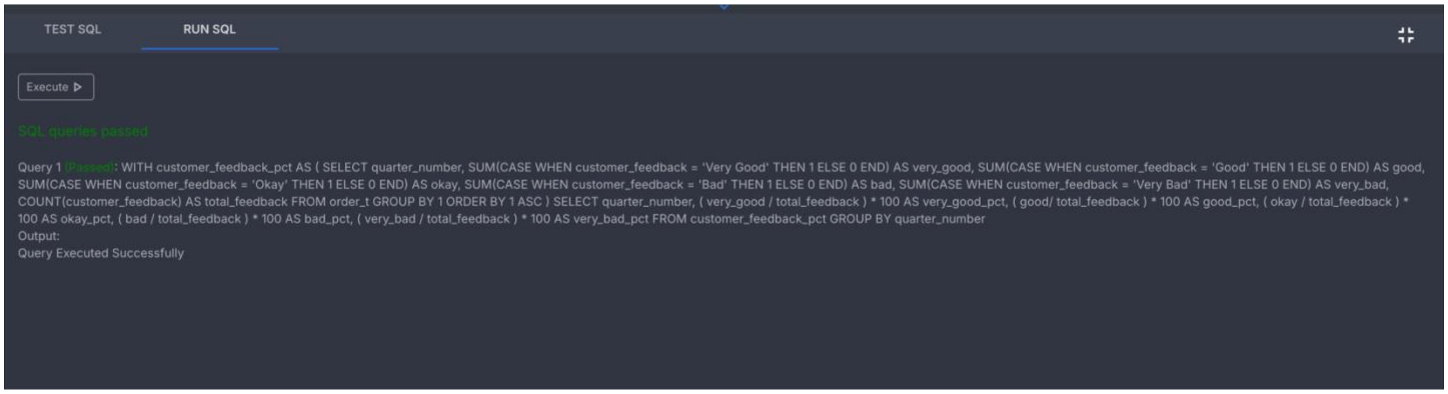
quarter_numb...	vg_percenta...	g_percenta...	ok_percenta...	bad_percenta...	vb_percenta...
1	30.0000	28.7097	19.0323	11.2903	10.9677
2	28.6260	22.1374	20.2290	14.1221	14.8855
3	16.5939	20.9607	21.8341	22.7074	17.9039
4	10.0503	10.0503	20.1005	29.1457	30.6533

Solution Query using Common Table Expressions (CTE):

```
WITH customer_feedback_pct AS
(
        SELECT
                quarter_number,
                SUM(CASE WHEN customer_feedback = 'Very Good' THEN 1 ELSE 0 END) AS very_good,
                SUM(CASE WHEN customer_feedback = 'Good' THEN 1 ELSE 0 END) AS good,
                SUM(CASE WHEN customer_feedback = 'Okay' THEN 1 ELSE 0 END) AS okay,
                SUM(CASE WHEN customer_feedback = 'Bad' THEN 1 ELSE 0 END) AS bad,
                SUM(CASE WHEN customer_feedback = 'Very Bad' THEN 1 ELSE 0 END) AS very_bad,
                COUNT(customer_feedback) AS total_feedback
        FROM order_t
        GROUP BY 1
  ORDER BY 1 ASC
)

SELECT
     quarter_number,
    ( very_good / total_feedback ) * 100 AS very_good_pct,
    ( good/ total_feedback ) * 100  AS good_pct,
    ( okay / total_feedback ) * 100  AS okay_pct,
    ( bad / total_feedback ) * 100    AS bad_pct,
    ( very_bad / total_feedback ) * 100  AS very_bad_pct
```

```
FROM   customer_feedback_pct
GROUP  BY quarter_number;
```

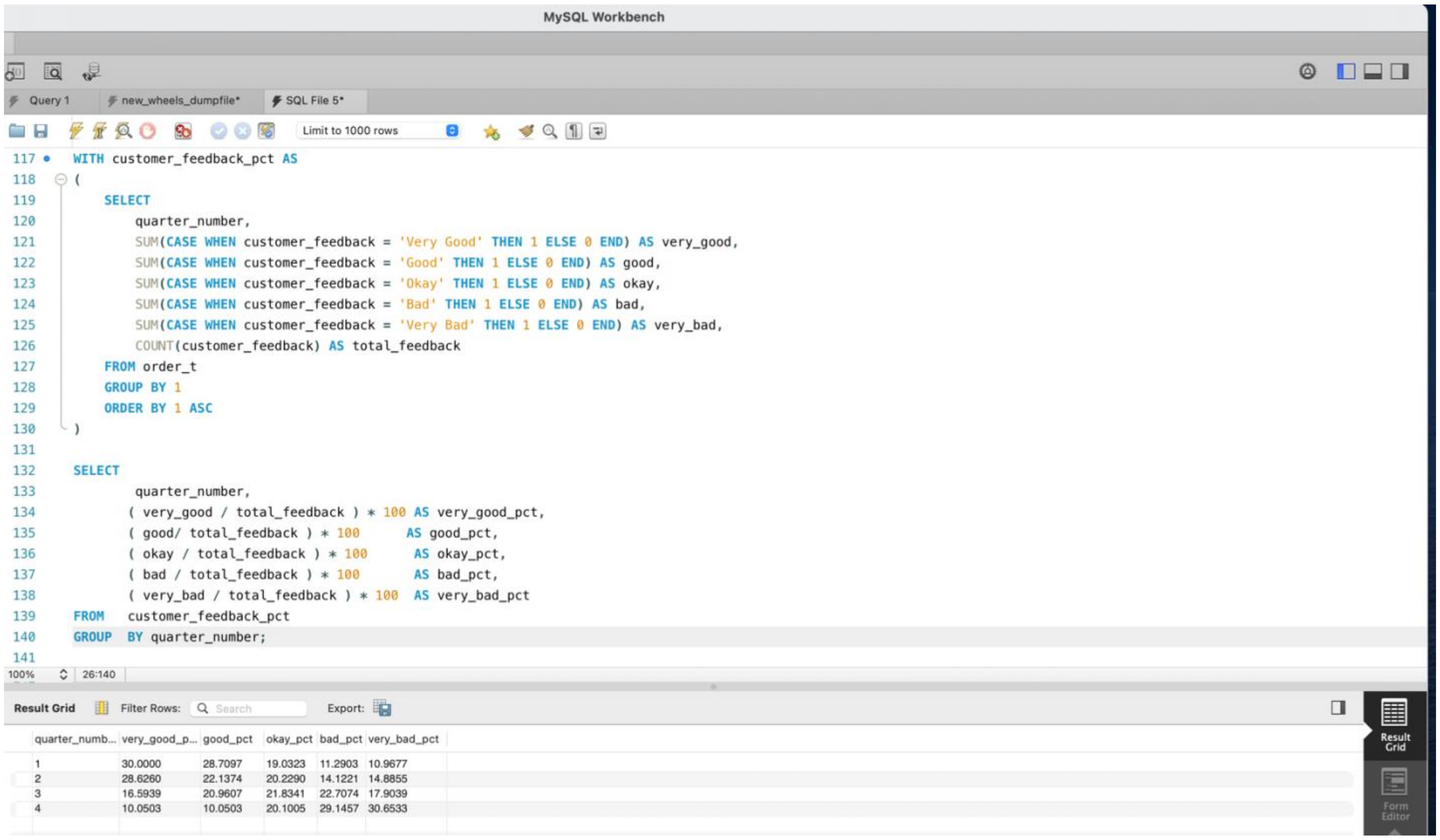
Output:



```
Query 1 (Passed): WITH customer_feedback_pct AS ( SELECT quarter_number, SUM(CASE WHEN customer_feedback = 'Very Good' THEN 1 ELSE 0 END) AS very_good, SUM(CASE WHEN customer_feedback = 'Good' THEN 1 ELSE 0 END) AS good, SUM(CASE WHEN customer_feedback = 'Okay' THEN 1 ELSE 0 END) AS okay, SUM(CASE WHEN customer_feedback = 'Bad' THEN 1 ELSE 0 END) AS bad, SUM(CASE WHEN customer_feedback = 'Very Bad' THEN 1 ELSE 0 END) AS very_bad, COUNT(customer_feedback) AS total_feedback FROM order_t GROUP BY 1 ORDER BY 1 ASC ) SELECT quarter_number, ( very_good / total_feedback ) * 100 AS very_good_pct, ( good/ total_feedback ) * 100 AS good_pct, ( okay / total_feedback ) * 100 AS okay_pct, ( bad / total_feedback ) * 100 AS bad_pct, ( very_bad / total_feedback ) * 100 AS very_bad_pct FROM customer_feedback_pct GROUP BY quarter_number
Output:
Query Executed Successfully
```

Note: Query runs successfully in SQL Playground but returns no rows while using CTEs.

Output in MySQL Workbench:



```
WITH customer_feedback_pct AS
(
    SELECT
        quarter_number,
        SUM(CASE WHEN customer_feedback = 'Very Good' THEN 1 ELSE 0 END) AS very_good,
        SUM(CASE WHEN customer_feedback = 'Good' THEN 1 ELSE 0 END) AS good,
        SUM(CASE WHEN customer_feedback = 'Okay' THEN 1 ELSE 0 END) AS okay,
        SUM(CASE WHEN customer_feedback = 'Bad' THEN 1 ELSE 0 END) AS bad,
        SUM(CASE WHEN customer_feedback = 'Very Bad' THEN 1 ELSE 0 END) AS very_bad,
        COUNT(customer_feedback) AS total_feedback
    FROM order_t
    GROUP BY 1
    ORDER BY 1 ASC
)

SELECT
        quarter_number,
       ( very_good / total_feedback ) * 100 AS very_good_pct,
       ( good/ total_feedback ) * 100      AS good_pct,
       ( okay / total_feedback ) * 100      AS okay_pct,
       ( bad / total_feedback ) * 100      AS bad_pct,
       ( very_bad / total_feedback ) * 100  AS very_bad_pct
FROM   customer_feedback_pct
GROUP  BY quarter_number;
```

quarter_numb...	very_good_p...	good_pct	okay_pct	bad_pct	very_bad_pct
1	30.0000	28.7097	19.0323	11.2903	10.9677
2	28.6260	22.1374	20.2290	14.1221	14.8855
3	16.5939	20.9607	21.8341	22.7074	17.9039
4	10.0503	10.0503	20.1005	29.1457	30.6533

Observations and Insights:

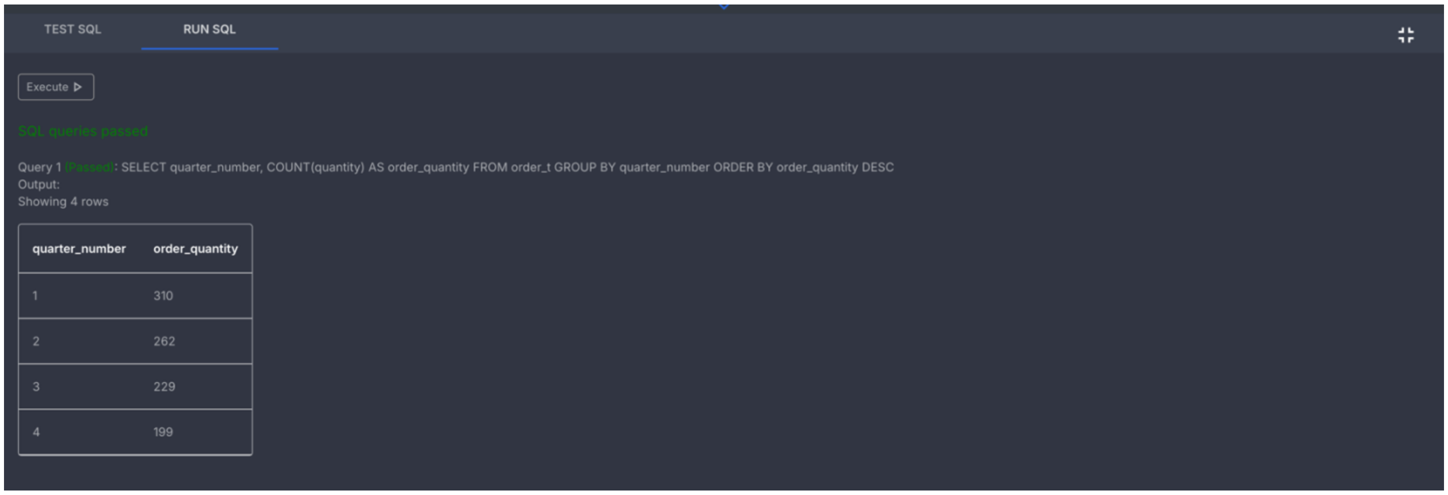
- Based on the above results, the customer feedback percentage for 'Very Good' and 'Good' dropped from Q1 to Q4 and the percentage of 'Bad' and 'Very Bad' has increased
- This indicates that the customers are getting dissatisfied over time

Question 6: What is the trend of the number of orders by quarter?

Solution Query:

```
SELECT
          quarter_number,
          COUNT(quantity) AS order_quantity
FROM   order_t
GROUP  BY quarter_number
ORDER  BY order_quantity DESC;
```

Output:



SQL queries passed

```
Query 1 (Passed): SELECT quarter_number, COUNT(quantity) AS order_quantity FROM order_t GROUP BY quarter_number ORDER BY order_quantity DESC
Output:
Showing 4 rows
```

quarter_number	order_quantity
1	310
2	262
3	229
4	199

Observations and Insights:

- Q1 has the highest number of orders and Q4 has the least number of orders
- There has been a decline in the number of orders placed by customers each quarter

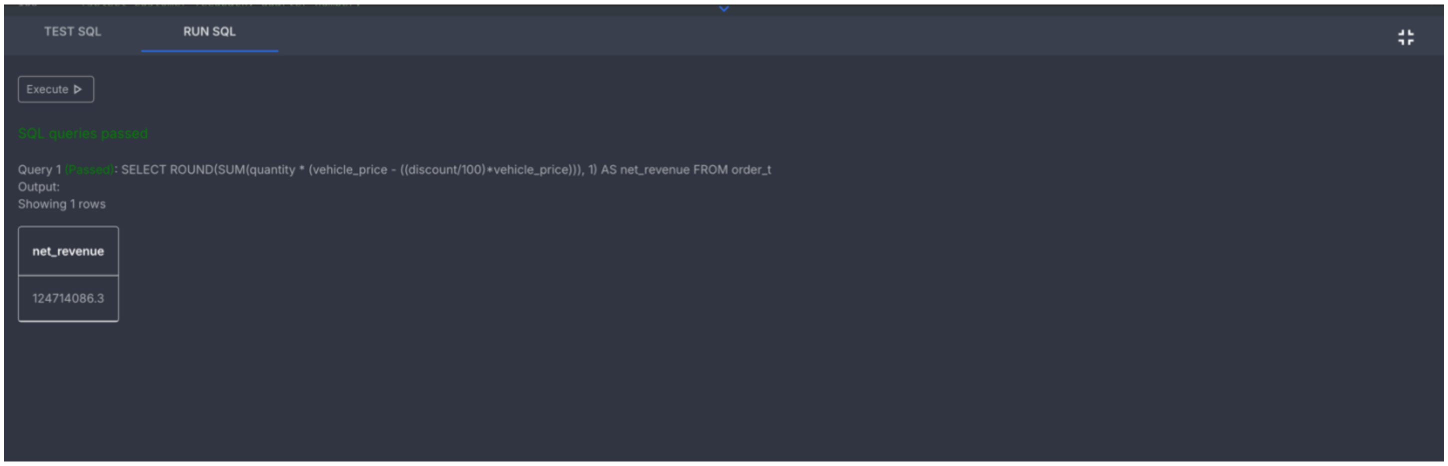
Question 7: Calculate the net revenue generated by the company. What is the quarter-over-quarter % change in net revenue?

Net Revenue generated by the company:

Solution Query:

```
SELECT
        ROUND(SUM(quantity * (vehicle_price - ((discount/100)*vehicle_price))), 1) AS net_revenue
FROM order_t;
```

Output:



```
SQL queries passed

Query 1 (Passed): SELECT ROUND(SUM(quantity * (vehicle_price - ((discount/100)*vehicle_price))), 1) AS net_revenue FROM order_t
Output:
Showing 1 rows
```

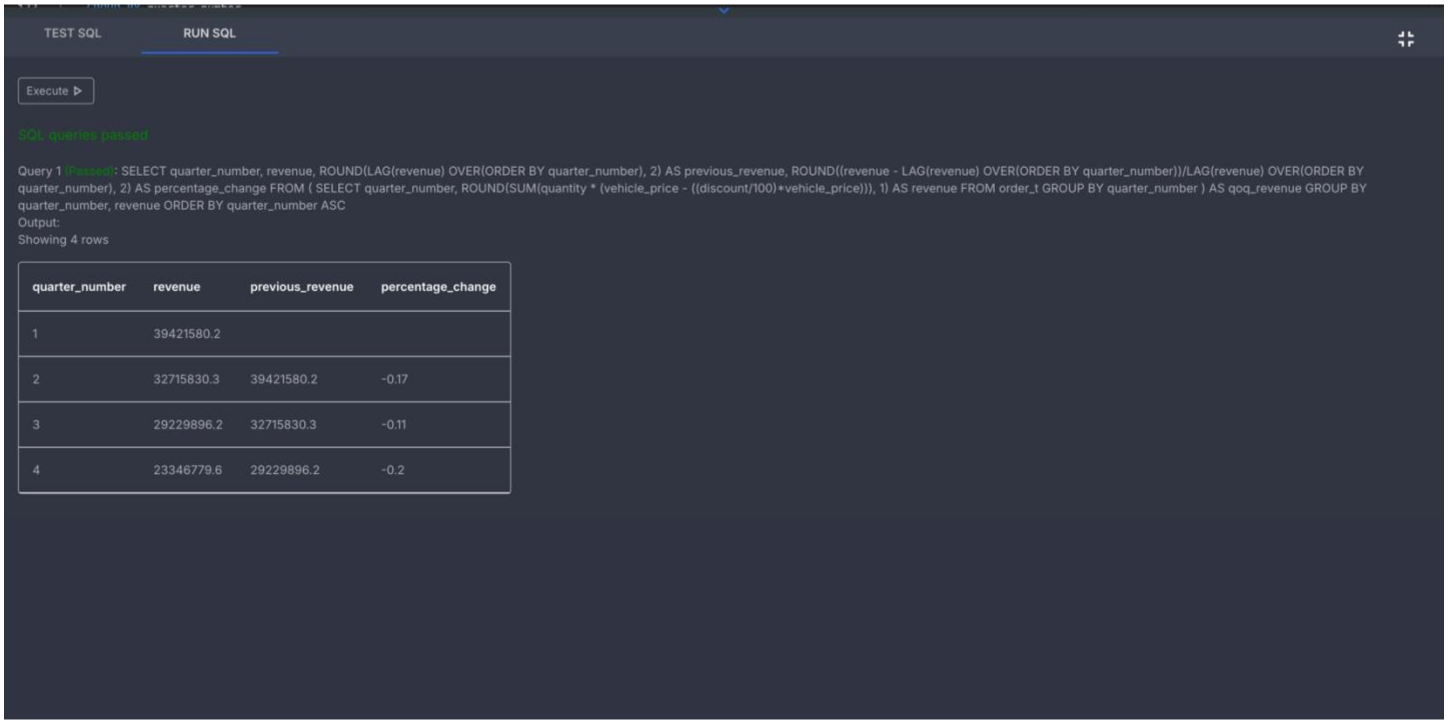
net_revenue
124714086.3

Quarter-over-quarter % change in net revenue:

Solution Query:

```
SELECT
        quarter_number,
        revenue,
        ROUND(LAG(revenue) OVER(ORDER BY quarter_number), 2) AS previous_revenue,
        ROUND((revenue - LAG(revenue) OVER(ORDER BY quarter_number))/LAG(revenue) OVER(ORDER
BY quarter_number), 2) AS percentage_change
FROM
(
        SELECT
                quarter_number,
                ROUND(SUM(quantity * (vehicle_price - ((discount/100)*vehicle_price))), 1) AS revenue
        FROM order_t
        GROUP BY quarter_number
) AS qoq_revenue
GROUP BY quarter_number, revenue
ORDER BY quarter_number ASC;
```

Output:



SQL queries passed

```
Query 1 (Passed): SELECT quarter_number, revenue, ROUND(LAG(revenue) OVER(ORDER BY quarter_number), 2) AS previous_revenue, ROUND((revenue - LAG(revenue) OVER(ORDER BY quarter_number))/LAG(revenue) OVER(ORDER BY quarter_number), 2) AS percentage_change FROM ( SELECT quarter_number, ROUND(SUM(quantity * (vehicle_price - ((discount/100)*vehicle_price))), 1) AS revenue FROM order_t GROUP BY quarter_number ) AS qoq_revenue GROUP BY quarter_number, revenue ORDER BY quarter_number ASC
Output:
Showing 4 rows
```

quarter_number	revenue	previous_revenue	percentage_change
1	39421580.2		
2	32715830.3	39421580.2	-0.17
3	29229896.2	32715830.3	-0.11
4	23346779.6	29229896.2	-0.2

Observations and Insights:

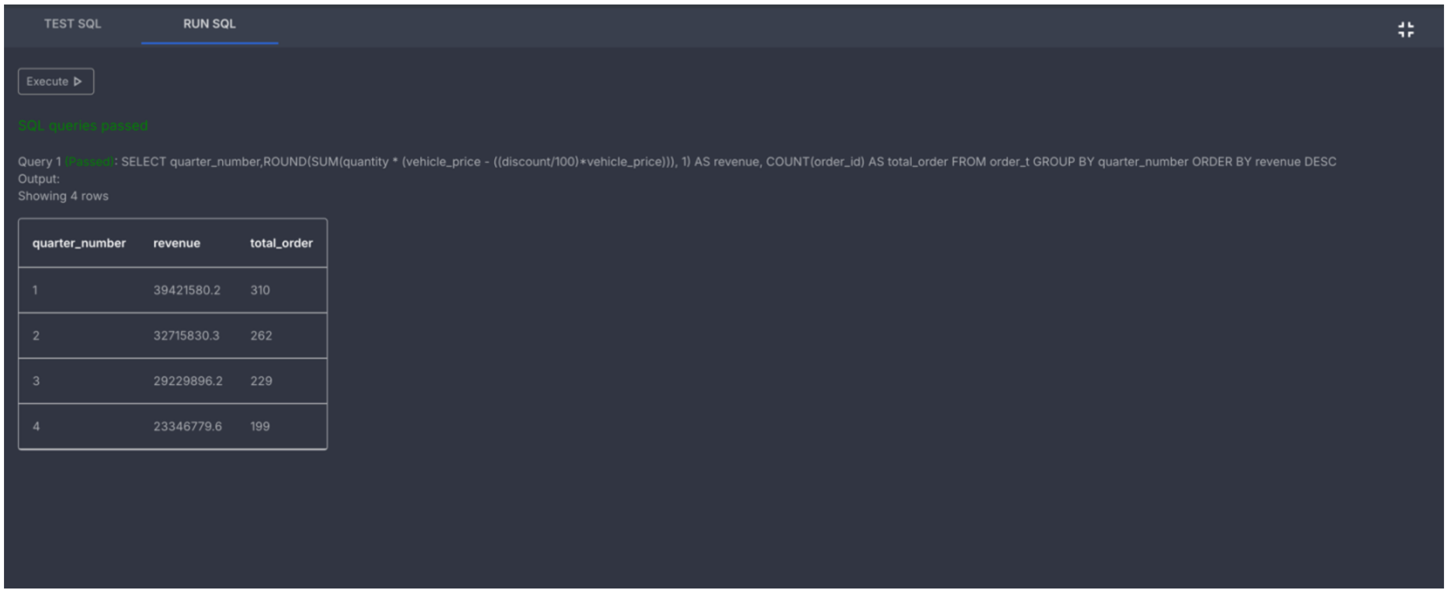
- The net revenue generated by the company is approximately 125M (124714086.3)
- There has been a negative trend in the quarter-over-quarter percentage change in revenue with a decline of approximately 17% from Q1 to Q2, 11% from Q2 to Q3 and about 20% from Q3 to Q4

Question 8: What is the trend of net revenue and orders by quarters?

Solution Query:

```
SELECT
   quarter_number,
   ROUND(SUM(quantity * (vehicle_price - ((discount/100)* vehicle_price))), 1) AS revenue,
   COUNT(order_id) AS total_order
FROM order_t
GROUP BY quarter_number
ORDER BY revenue DESC;
```

Output:



```
SQL queries passed

Query 1 (Passed): SELECT quarter_number,ROUND(SUM(quantity * (vehicle_price - ((discount/100)*vehicle_price))), 1) AS revenue, COUNT(order_id) AS total_order FROM order_t GROUP BY quarter_number ORDER BY revenue DESC
Output:
Showing 4 rows
```

quarter_number	revenue	total_order
1	39421580.2	310
2	32715830.3	262
3	29229896.2	229
4	23346779.6	199

Observations and Insights:

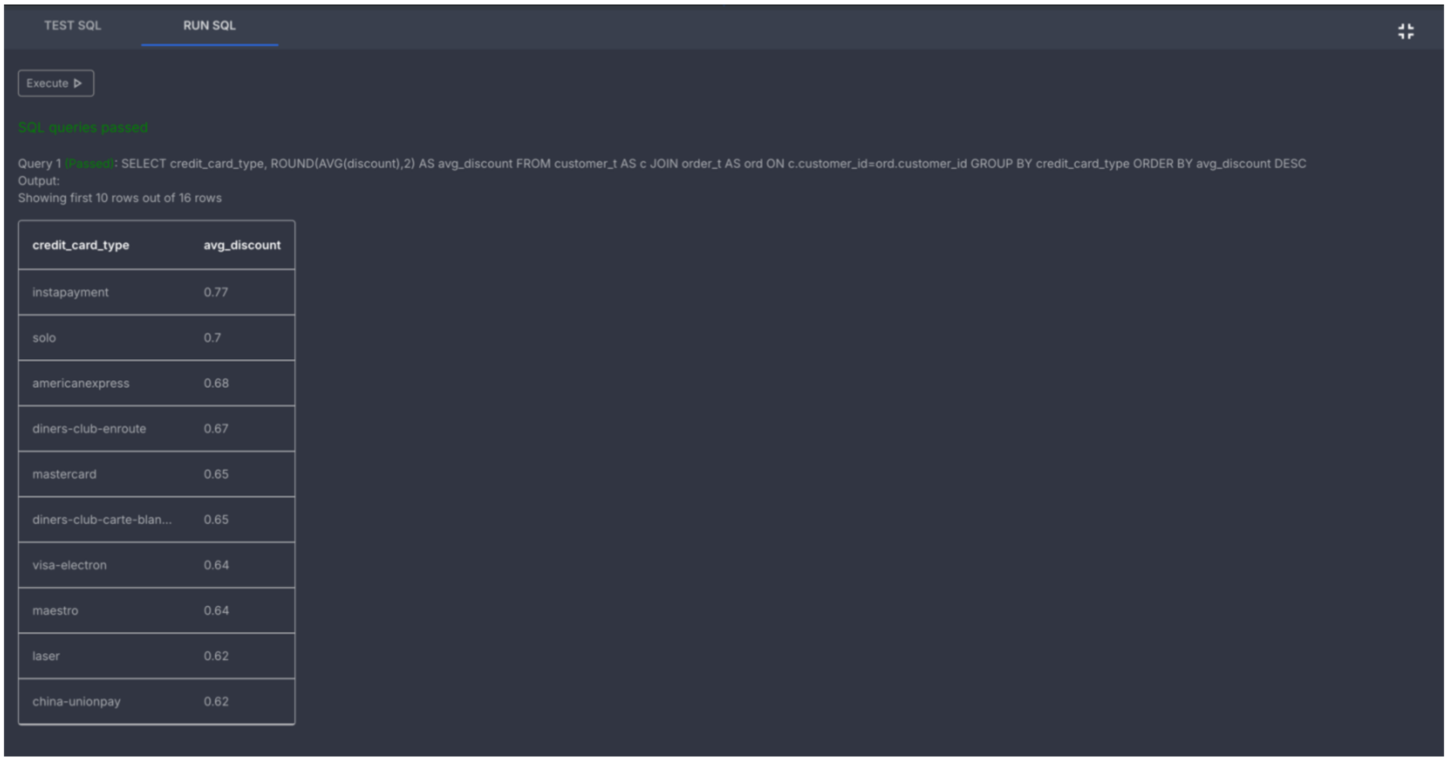
- There has been a decline in the no. of orders placed and revenue from Q1 to Q4
- The percentage decline from Q1 to Q4 in the total revenue is about 50% and that of the no. of orders placed by customers by about 40%

Question 9: What is the average discount offered for different types of credit cards?

Solution Query:

```
SELECT
        credit_card_type,
        ROUND(AVG(discount),2) AS avg_discount
FROM customer_t AS c
JOIN order_t AS ord ON c.customer_id=ord.customer_id
GROUP BY credit_card_type
ORDER BY avg_discount DESC;
```

Output:



SQL queries passed

```
Query 1 (Passed): SELECT credit_card_type, ROUND(AVG(discount),2) AS avg_discount FROM customer_t AS c JOIN order_t AS ord ON c.customer_id=ord.customer_id GROUP BY credit_card_type ORDER BY avg_discount DESC
Output:
Showing first 10 rows out of 16 rows
```

credit_card_type	avg_discount
instapayment	0.77
solo	0.7
americanexpress	0.68
diners-club-enroute	0.67
mastercard	0.65
diners-club-carte-blan...	0.65
visa-electron	0.64
maestro	0.64
laser	0.62
china-unionpay	0.62

Observations and Insights:

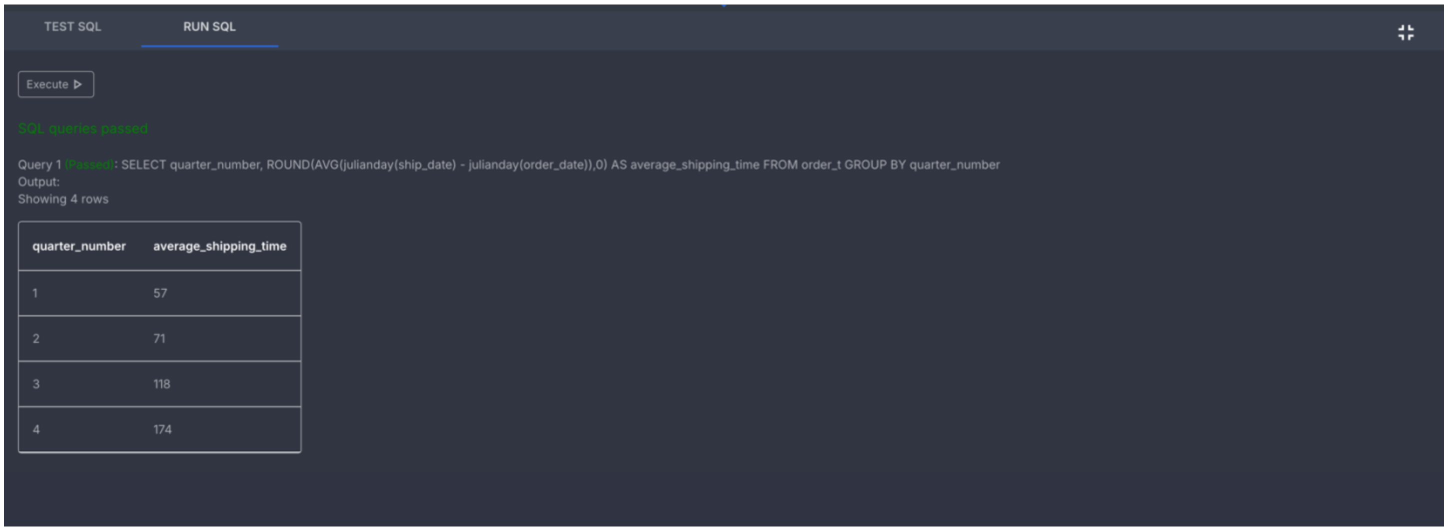
- The average discount offered by credit card type varies from 62% to 77%
- Instapayment and Solo are among the highest, offering more than 70% discount
- Laser and china-unionpay are among the least discount by card type

Question 10: What is the average time taken to ship the placed orders for each quarter?

Solution Query:

```
SELECT
        quarter_number,
        ROUND(AVG(julianday(ship_date) - julianday(order_date)),0) AS average_shipping_time
FROM order_t
GROUP BY quarter_number;
```

Output:



SQL queries passed

```
Query 1 (Passed): SELECT quarter_number, ROUND(AVG(julianday(ship_date) - julianday(order_date)),0) AS average_shipping_time FROM order_t GROUP BY quarter_number
Output:
Showing 4 rows
```

quarter_number	average_shipping_time
1	57
2	71
3	118
4	174

Observations and Insights:

- The average shipping time has increased from Q1 to Q4 by 117 days
- This significant increase in average shipping time indicates more delays in shipping the product to the customer

Business Metrics Overview

Total Revenue	Total Orders	Total Customers	Average Rating
125M (125482804.43)	1000	994	3.1
Last Quarter Revenue	**Last quarter Orders**	**Average Days to Ship**	**% Good Feedback**
23.5M (23496008.2)	199	105	21%(approx.)

Business Recommendations

Based on the above metrics, the following recommendations can be made to New Wheels to address the declining sales:

- Identify the reasons for a decline in the average customer rating per quarter (from 3.1 in Q1 to 2.4 in Q2)
- Conduct surveys to identify the areas of improvement to increase customer satisfaction
- Analyze the root cause for a decrease in the revenue and total number of orders per quarter
- Find out orders that generate more revenue per quarter
- Improve the average shipping time to avoid the delays in shipping the products to the customer