**ITBAN 2**

**ACTIVITY 1**



BANQUIL, JUNMER

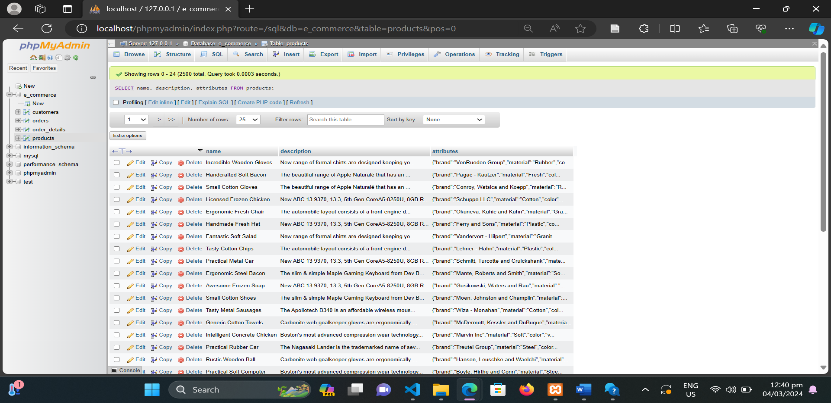
NOTARION, KRISTINE

Adviser

JOHN REY PAULIN

**1. Retrieve Product Information:**

• Write a query to fetch the names and descriptions of all products.

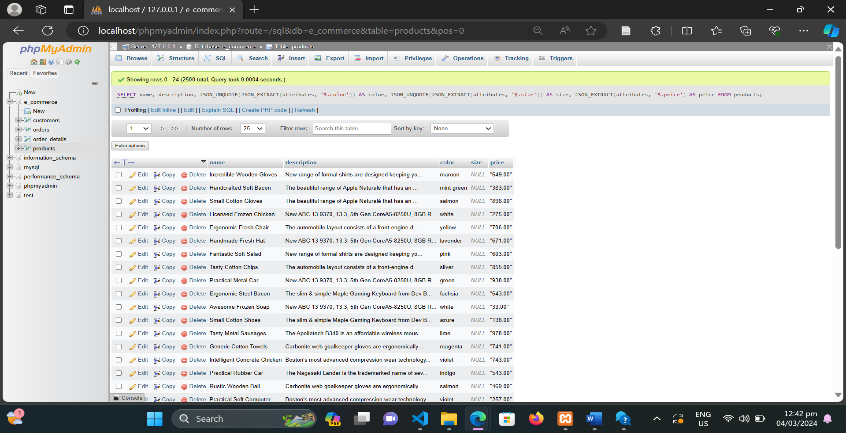
SELECT productName,

productDescription,

productAttributes

FROM products;

• Extend the previous query to include specific attributes such as color, size, and price.

SELECT name, description

JSON\_UNQUOTE(JSON\_EXTRACT

(attributes, '$.color')) AS color,

JSON\_UNQUOTE(JSON\_EXTRACT

(attributes, '$.size')) AS size,

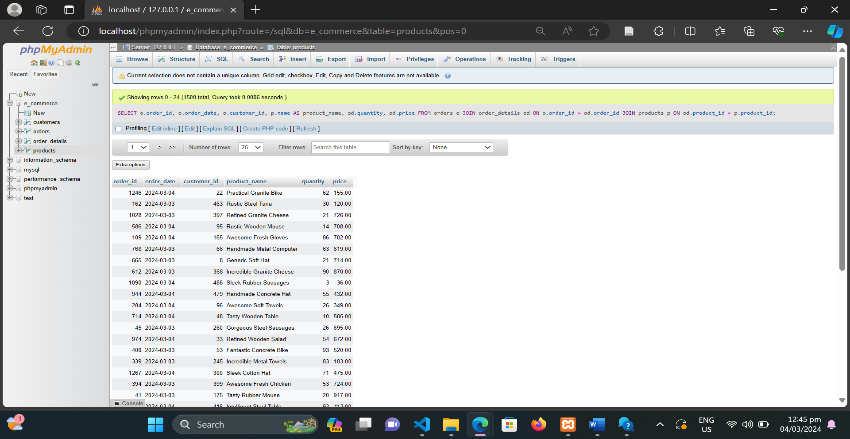
JSON\_EXTRACT(attributes, '$.price')

AS price

FROM products;

**2. Query Orders and Order Details:**

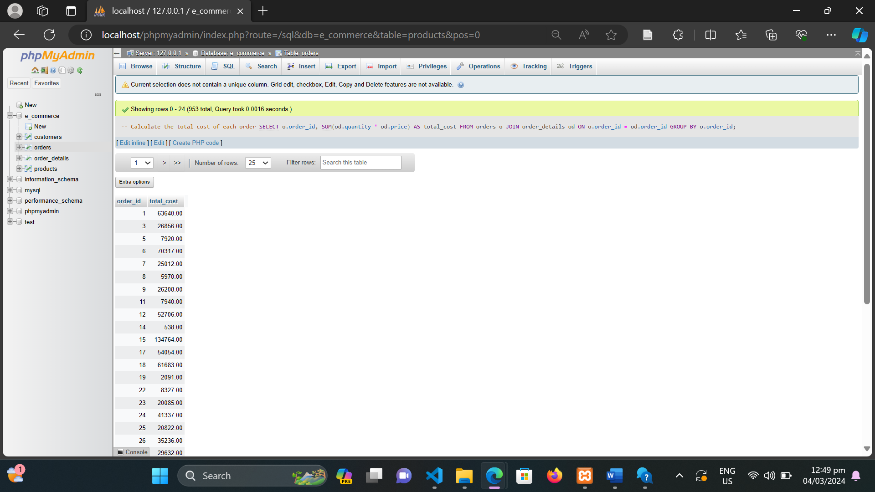
•Retrieve the details of all orders placed, including the order date, customer ID, product name, quantity, and price.

SELECT o.order\_id, o.order\_date, o.customer\_id, p.name AS product\_name od.quantity od.price

FROM orders o

JOINord er\_details od ON o.order\_id = od.order\_id

JOINproducts p ON od.product\_id = p.product\_id;

•Calculate the total cost of each order.

SELECT o.order\_id,

SUM(od.quantity \* od.price) AS total\_cost

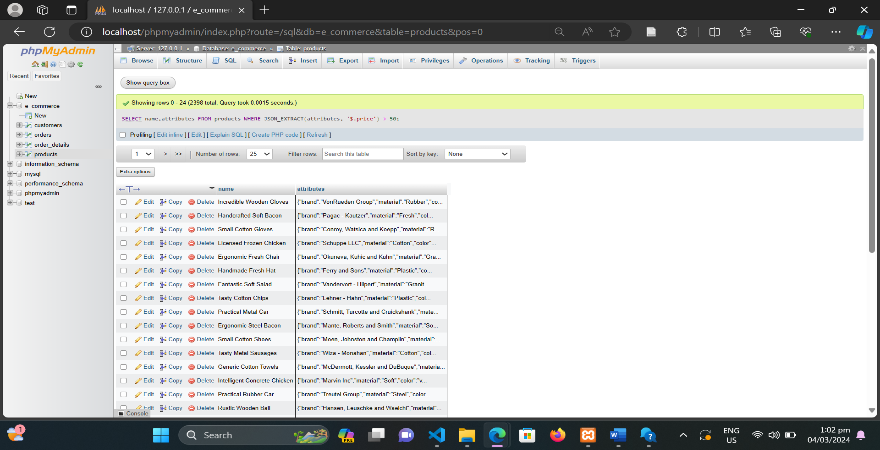
FROM orders o

JOIN order\_details od ON o.order\_id = od.order\_id

GROUP BY o.order\_id;

**3.Filtering Products Based on Attributes:**

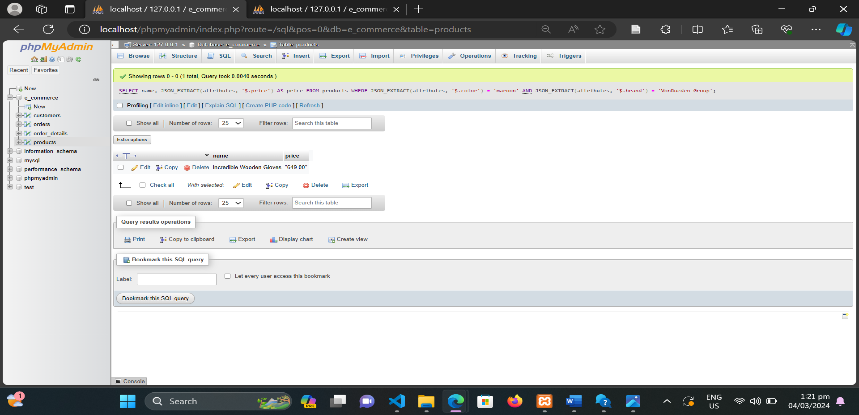
•Write a query to find all products with a price greater than $50.

SELECT name,attributes

FROM products

WHERE JSON\_EXTRACT(attributes, '$.price') > 50;

•Filter products by color and brand, and display their names and prices.



SELECT name, JSON\_EXTRACT(attributes, '$.price') AS price

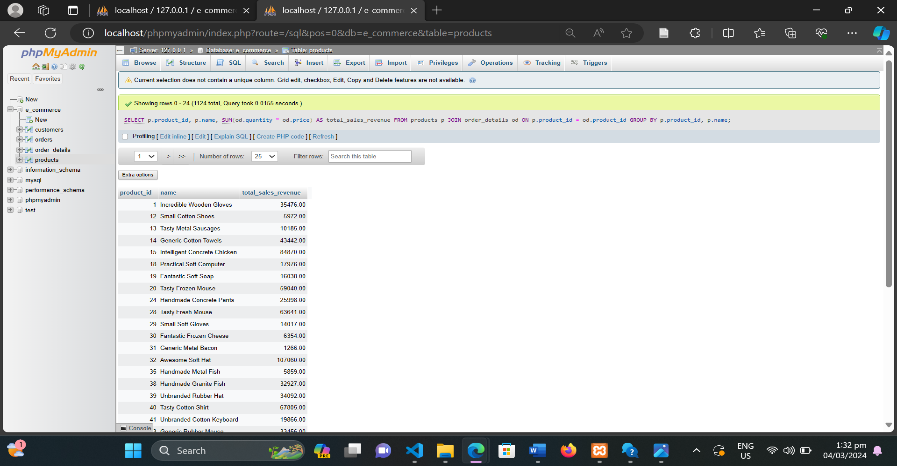
FROM products

WHERE JSON\_EXTRACT(attributes, '$.color') = 'maroon'

AND JSON\_EXTRACT(attributes, '$.brand') = 'VonRueden Group';

**4. Calculating Aggregate Data:**

• Calculate the total sales revenue generated by each product.



SELECT p.product\_id, p.name,

SUM(od.quantity \* od.price)

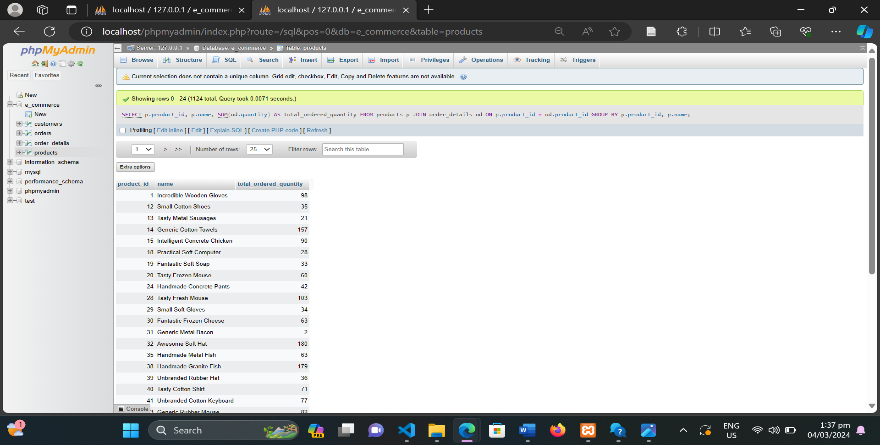
AS total\_sales\_revenue

FROM products p

JOIN order\_details od ON p.product\_id = od.product\_id

GROUP BY p.product\_id, p.name;

• Determine the total quantity of each product ordered.

SELECT p.product\_id, p.name,

SUM(od.quantity) AS total\_ordered\_quantity

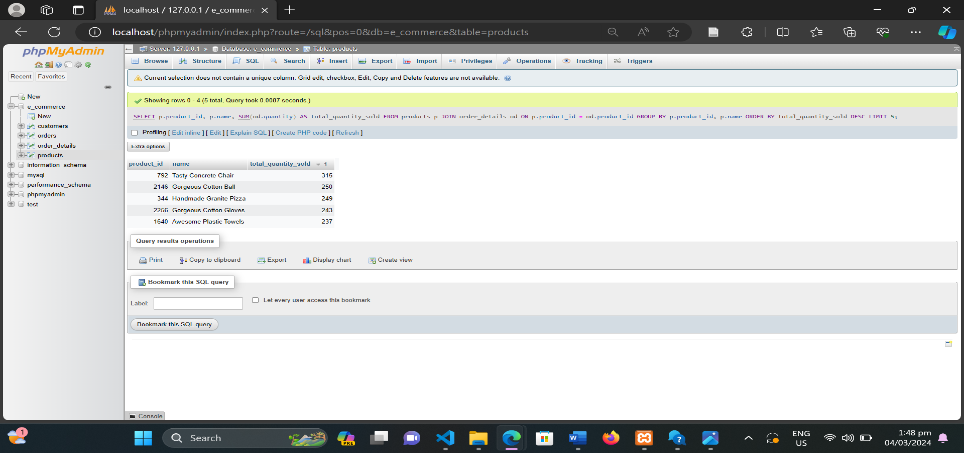
FROM products p

JOIN order\_details od ON p.product\_id = od.product\_id

GROUP BY p.product\_id, p.name;

**5. Advanced Filtering and Aggregation:**

• Find the top 5 best-selling products based on total quantity sold.

SELECT p.product\_id, p.name,

SUM(od.quantity) AS total\_quantity\_sold

FROM products p

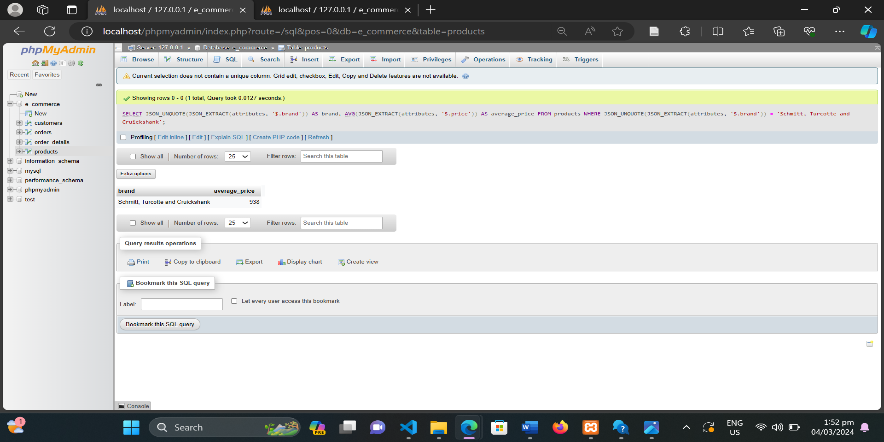
JOIN order\_details od ON p.product\_id = od.product\_id

GROUP BY p.product\_id, p.name

ORDER BY total\_quantity\_sold DESC

LIMIT 5;

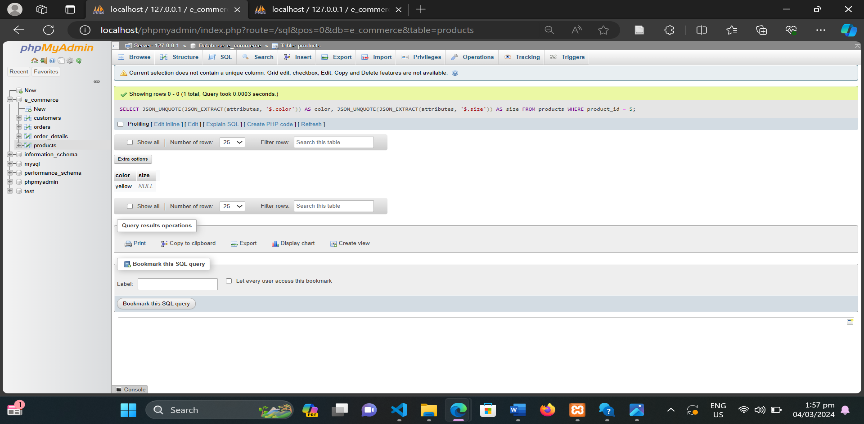
• Identify the average price of products from a specific brand.

SELECT JSON\_UNQUOTE (JSON\_EXTRACT(attributes, '$.brand')) AS brand, AVG(JSON\_EXTRACT(attributes, '$.price')) AS average\_price

FROM products

WHERE JSON\_UNQUOTE (JSON\_EXTRACT(attributes, '$.brand')) = 'Schmitt, Turcotte and Cruickshank';

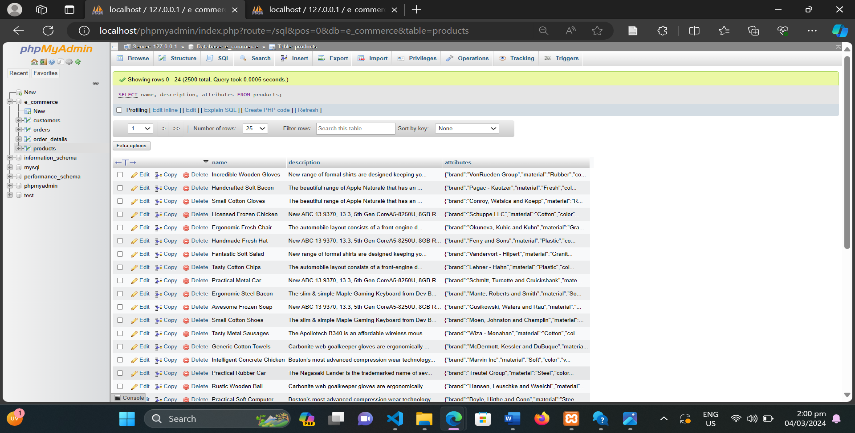
**6. Nested JSON Queries:**

• Retrieve the color and size of a specific product.

SELECT JSON\_UNQUOTE(JSON\_EXTRACT (attributes, '$.color')) AS color, JSON\_UNQUOTE(JSON\_EXTRACT (attributes, '$.size')) AS size

FROM products

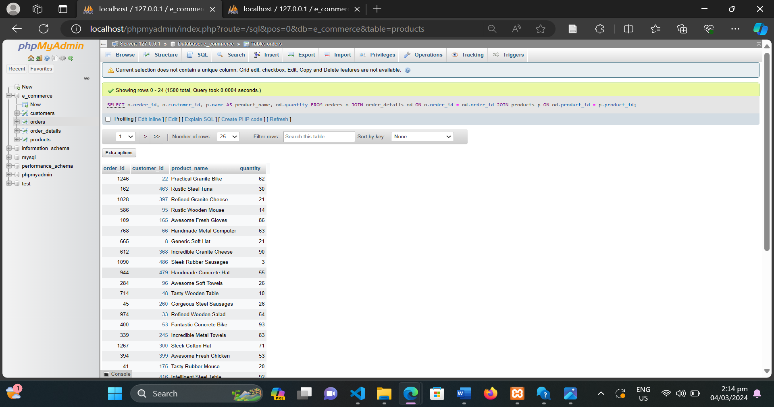
WHERE product\_id = 5;

•Extract and display all available attributes of products in JSON format.

SELECTname, description, attributes

FROM products;

**7. Joining Multiple Tables:**

• Write a query to find all orders placed by customers along with the products ordered and their quantities.

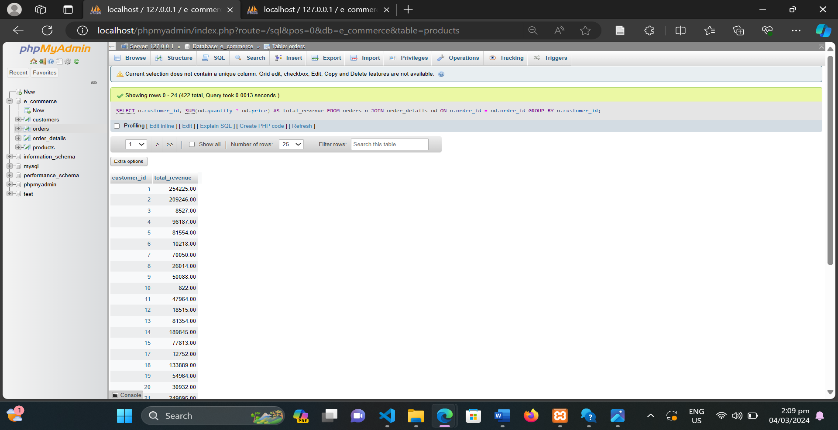
SELECT o.order\_id, o.customer\_id, p.name AS product\_name, od.quantity

FROM orders o

JOIN order\_details od ON o.order\_id = od.order\_id

JOIN products p ON od.product\_id = p.product\_id;

•Calculate the total revenue generated by each customer.

SELECT o.customer\_id,

SUM(od.quantity \* od.price) AS total\_revenue

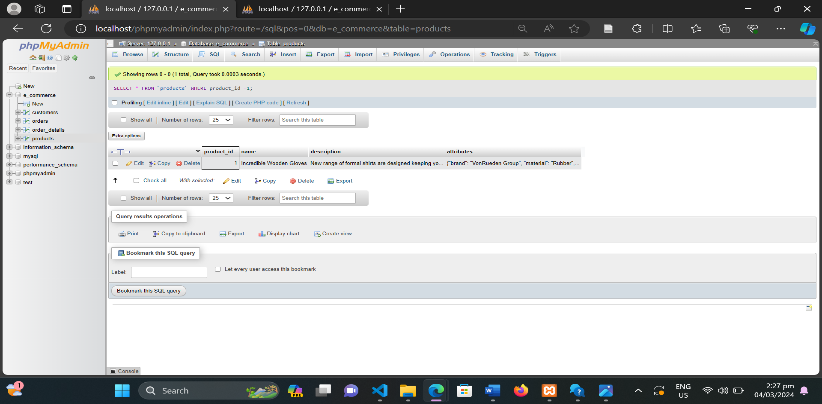
FROM orders o

JOIN order\_details od ON o.order\_id = od.order\_id

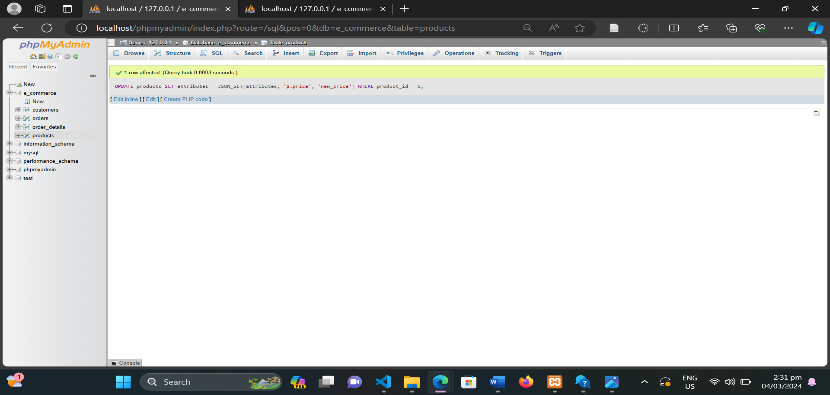
GROUP BY o.customer\_id;

**8. Data Manipulation with JSON Functions:**

• Update the price of a specific product stored as JSON attribute.



SELECT \* FROM `products` WHERE product\_id =1;

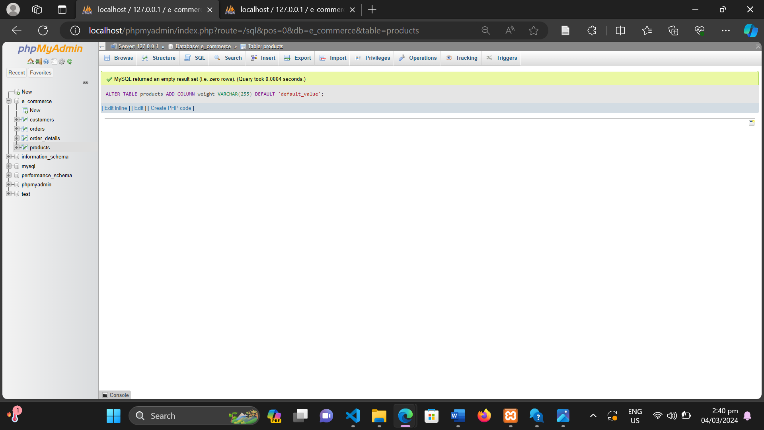


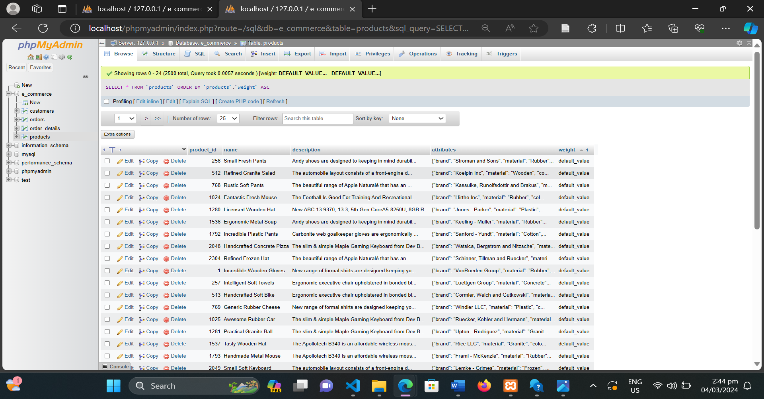
UPDATE products

SET attributes = JSON\_SET(attributes, '$.price', 'new\_price')

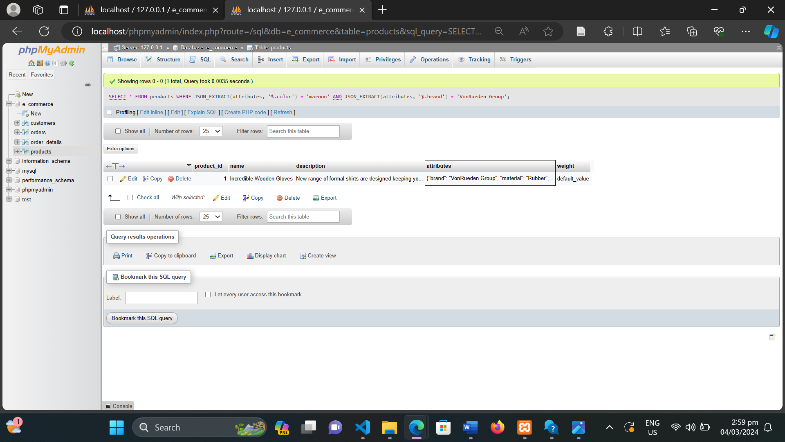
WHERE product\_id = 1;

• Add a new attribute to all products with a default value.

ALTER TABLE products ADD COLUMN **weight** VARCHAR(255)

DEFAULT 'default\_value';

9.Advanced JSON Operations:

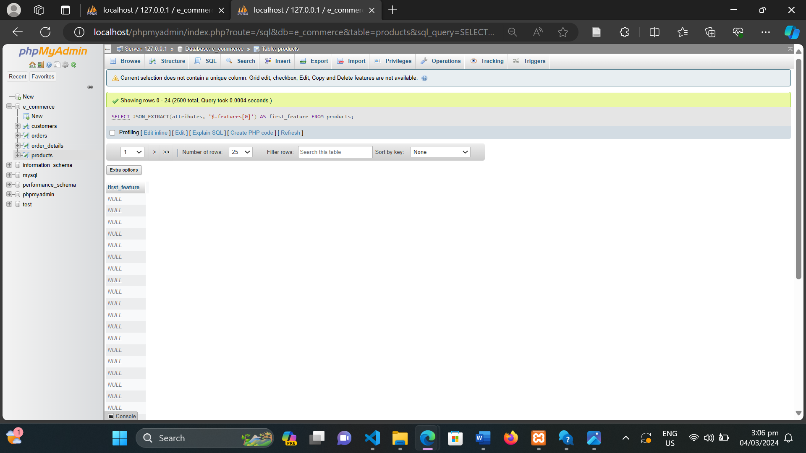
• Find products with specific attributes that match a given criteria using JSON path expressions.

SELECT \* FROM products

WHERE JSON\_EXTRACT(attributes, '$.color') =

'maroon' AND JSON\_EXTRACT(attributes, '$.brand') =

'VonRueden Group';

• Extract and display the first element of an array stored within a JSON attribute.

SELECT JSON\_EXTRACT(attributes, '$.features[0]')

AS first\_feature

FROM products;