Amazon-Themed Database MySQL Assignment

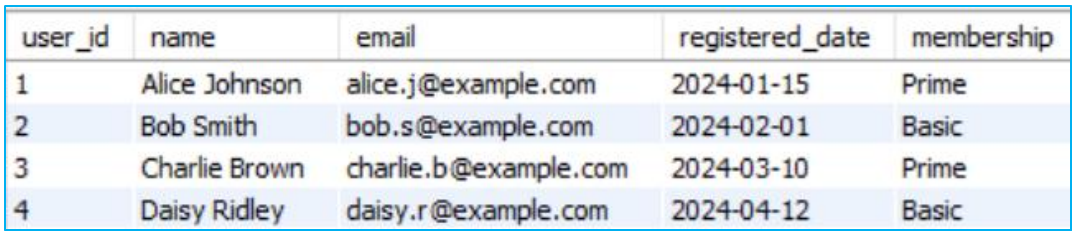
Dr. Subramani

```
-- Create AmazonDB
CREATE DATABASE amazondb;
USE amazondb;

-- Create 1. Users Table
CREATE TABLE users(
user_id INT PRIMARY KEY AUTO_INCREMENT,
name VARCHAR(100) NOT NULL,
email varchar(150) UNIQUE NOT NULL,
registered_date DATE NOT NULL,
membership ENUM("Basic", "Prime") DEFAULT "Basic"
);

-- Create 2. Products Table
CREATE TABLE products(
product_id INT PRIMARY KEY AUTO_INCREMENT,
name VARCHAR(200) NOT NULL,
price DECIMAL(10,2) NOT NULL,
category VARCHAR(100) NOT NULL,
stock INT NOT NULL
);

-- Create 3. Orders Table
CREATE TABLE orders(
order_id INT PRIMARY KEY AUTO_INCREMENT,
user_id INT NOT NULL,
```

```sql
FOREIGN KEY (user_id) REFERENCES users(user_id),
order_date DATE NOT NULL,
total_amount DECIMAL(10,2) NOT NULL
);

-- Create 4. OrderDetails Table
CREATE TABLE orderdetails(
order_details_id INT PRIMARY KEY AUTO_INCREMENT,
order_id INT NOT NULL,
FOREIGN KEY (order_id) REFERENCES orders(order_id),
product_id INT NOT NULL,
FOREIGN KEY (product_id) REFERENCES products(product_id),
quantity INT NOT NULL);
```

-- Data Insertion

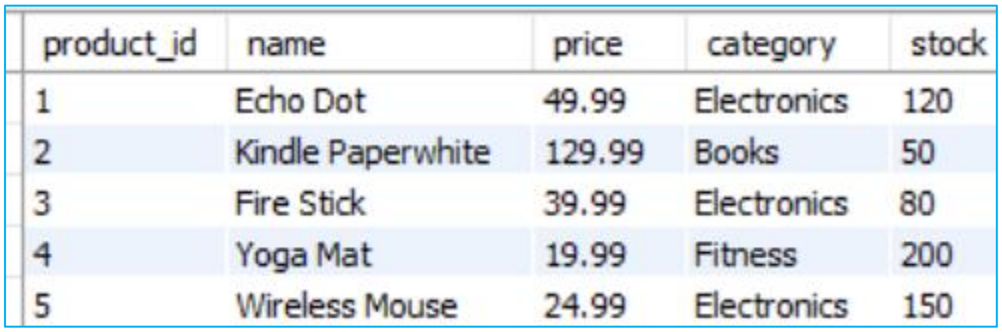
```sql
-- Data Insert into 1. Users Table:
INSERT INTO Users (name, email, registered_date, membership) VALUES
('Alice Johnson', 'alice.j@example.com', '2024-01-15', 'Prime'),
('Bob Smith', 'bob.s@example.com', '2024-02-01', 'Basic'),
('Charlie Brown', 'charlie.b@example.com', '2024-03-10', 'Prime'),
('Daisy Ridley', 'daisy.r@example.com', '2024-04-12', 'Basic');
SELECT * from Users;
```



user_id	name	email	registered_date	membership
1	Alice Johnson	alice.j@example.com	2024-01-15	Prime
2	Bob Smith	bob.s@example.com	2024-02-01	Basic
3	Charlie Brown	charlie.b@example.com	2024-03-10	Prime
4	Daisy Ridley	daisy.r@example.com	2024-04-12	Basic

-- Data Insert into 2. Products Table:

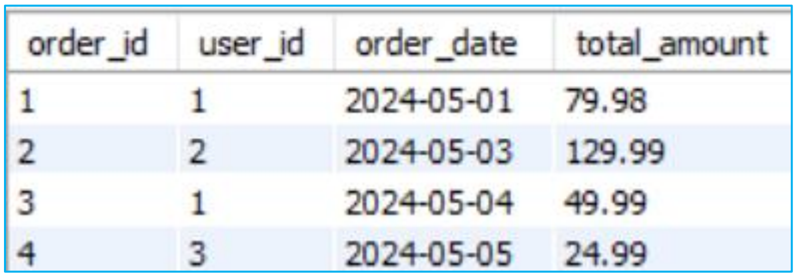
```
INSERT INTO Products (name, price, category, stock) VALUES
('Echo Dot', 49.99, 'Electronics', 120),
('Kindle Paperwhite', 129.99, 'Books', 50),
('Fire Stick', 39.99, 'Electronics', 80),
('Yoga Mat', 19.99, 'Fitness', 200),
('Wireless Mouse', 24.99, 'Electronics', 150);
SELECT * from Products;
```



product_id	name	price	category	stock
1	Echo Dot	49.99	Electronics	120
2	Kindle Paperwhite	129.99	Books	50
3	Fire Stick	39.99	Electronics	80
4	Yoga Mat	19.99	Fitness	200
5	Wireless Mouse	24.99	Electronics	150

-- Data Insert into 3. Orders Table:

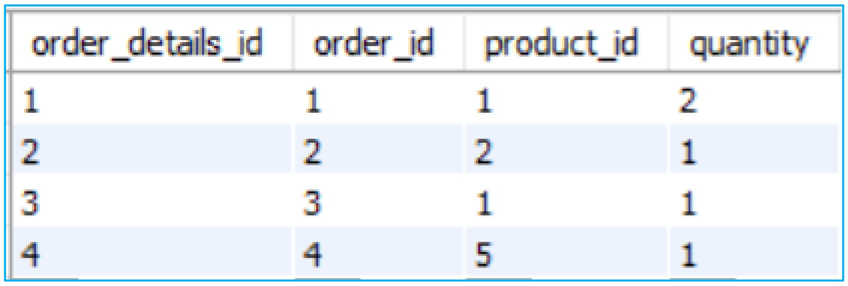
```
INSERT INTO Orders (user_id, order_date, total_amount) VALUES
(1, '2024-05-01', 79.98),
(2, '2024-05-03', 129.99),
(1, '2024-05-04', 49.99),
(3, '2024-05-05', 24.99);
SELECT * from Orders;
```



order_id	user_id	order_date	total_amount
1	1	2024-05-01	79.98
2	2	2024-05-03	129.99
3	1	2024-05-04	49.99
4	3	2024-05-05	24.99

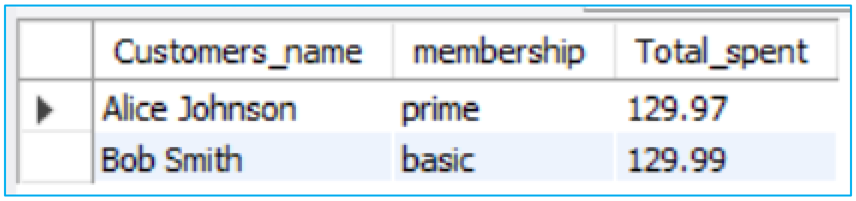
-- Data Insert into 4. OrderDetails Table:

```
INSERT INTO OrderDetails (order_id, product_id, quantity) VALUES
(1, 1, 2),
(2, 2, 1),
(3, 1, 1),
(4, 5, 1);
SELECT * from OrderDetails;
```



order_details_id	order_id	product_id	quantity
1	1	1	2
2	2	2	1
3	3	1	1
4	4	5	1

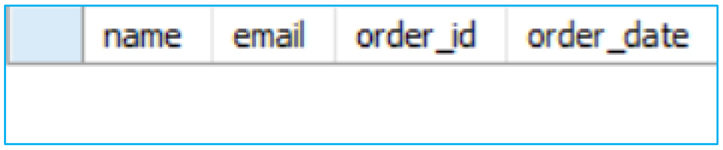
```
-- Assignment Questions:
-- 1. List all customers who have made purchases 1. of more than $80.
select
            users.name AS Customers_name,
   users.membership,
   SUM(orders.total_amount) AS Total_spent
FROM users
LEFT JOIN orders
USING(user_id)
GROUP BY users.user_id
HAVING Total_spent > 80;
```



Customers_name	membership	Total_spent
Alice Johnson	prime	129.97
Bob Smith	basic	129.99

-- 2. Retrieve all orders placed in the last 280 days along with the customer name and email.

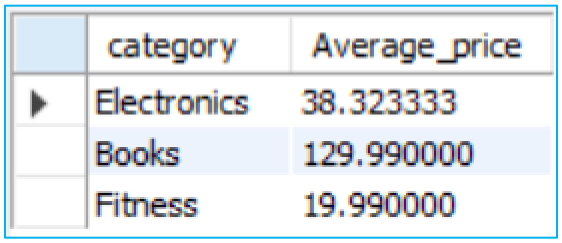
```
SELECT
users.name,
users.email,
orders.order_id,
orders.order_date
FROM
orders
LEFT JOIN users
USING(user_id)
WHERE orders.order_date >= DATE_SUB(CURDATE(), INTERVAL 280 DAY)
ORDER BY orders.order_date DESC;
```



name	email	order_id	order_date

-- 3. Find the average product price for each category?

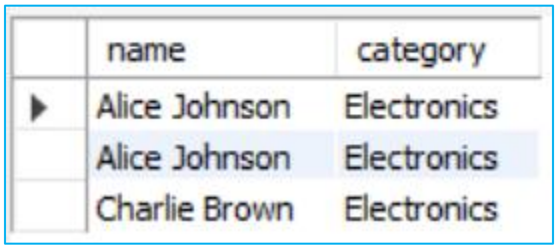
```
SELECT category, AVG(price) AS Average_price FROM products GROUP BY category;
```



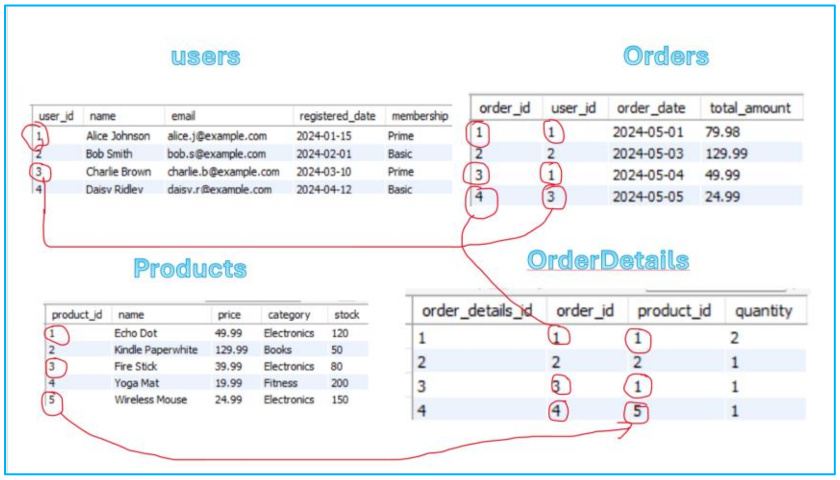
category	Average_price
Electronics	38.323333
Books	129.990000
Fitness	19.990000

-- 4. List all customers who have purchased a product from the category Electronics.?

```
SELECT users.name, products.category
FROM users
JOIN orders ON users.user_id = orders.user_id
JOIN orderdetails ON orders.order_id = orderdetails.order_id

JOIN products ON orderdetails.product_id = products.product_id

WHERE products.category = "Electronics";
```



name	category
Alice Johnson	Electronics
Alice Johnson	Electronics
Charlie Brown	Electronics



users

user_id	name	email	registered_date	membership
1	Alice Johnson	alice.j@example.com	2024-01-15	Prime
2	Bob Smith	bob.s@example.com	2024-02-01	Basic
3	Charlie Brown	charlie.b@example.com	2024-03-10	Prime
4	Daisy Ridley	daisy.r@example.com	2024-04-12	Basic

Orders

order_id	user_id	order_date	total_amount
1	1	2024-05-01	79.98
2	2	2024-05-03	129.99
3	1	2024-05-04	49.99
4	3	2024-05-05	24.99

Products

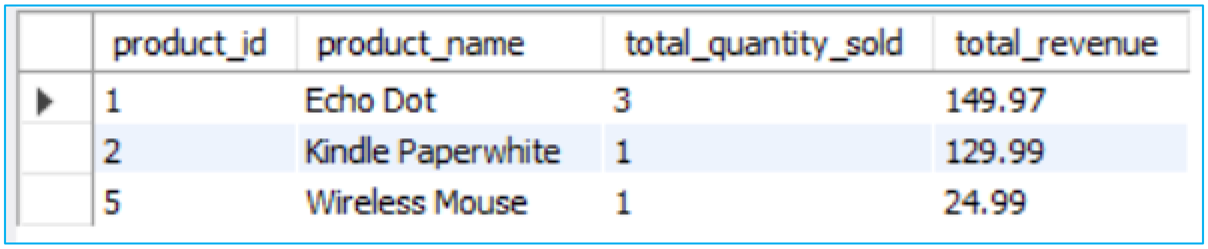
product_id	name	price	category	stock
1	Echo Dot	49.99	Electronics	120
2	Kindle Paperwhite	129.99	Books	50
3	Fire Stick	39.99	Electronics	80
4	Yoga Mat	19.99	Fitness	200
5	Wireless Mouse	24.99	Electronics	150

OrderDetails

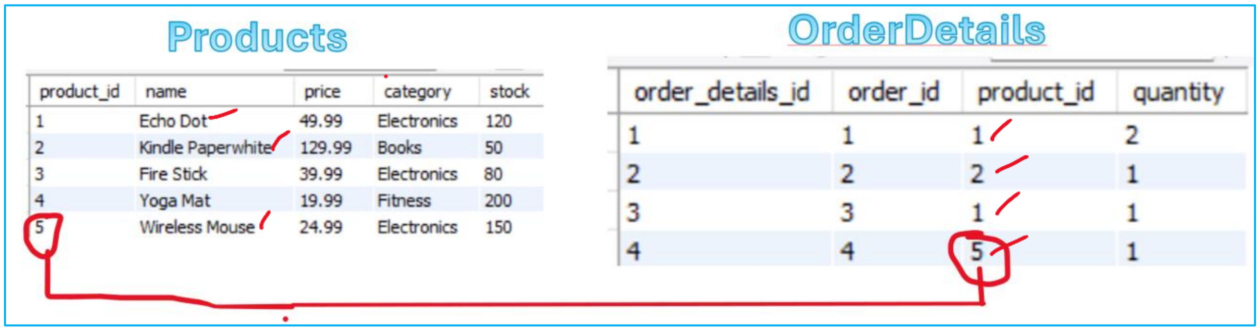
order_details_id	order_id	product_id	quantity
1	1	1	2
2	2	2	1
3	3	1	1
4	4	5	1

-- 5. Find the total number of products sold and the total revenue generated for each product?

```
SELECT

orderdetails.product_id,

products.Name AS product_name,

sum(orderdetails.quantity) AS total_quantity_sold,

sum(orderdetails.quantity * products.price) AS total_revenue

FROM orderdetails

JOIN products ON orderdetails.product_id = products.product_id

GROUP BY orderdetails.product_id;
```



product_id	product_name	total_quantity_sold	total_revenue
1	Echo Dot	3	149.97
2	Kindle Paperwhite	1	129.99
5	Wireless Mouse	1	24.99



Products

product_id	name	price	category	stock
1	Echo Dot	49.99	Electronics	120
2	Kindle Paperwhite	129.99	Books	50
3	Fire Stick	39.99	Electronics	80
4	Yoga Mat	19.99	Fitness	200
5	Wireless Mouse	24.99	Electronics	150

OrderDetails

order_details_id	order_id	product_id	quantity
1	1	1	2
2	2	2	1
3	3	1	1
4	4	5	1

-- 6. Update the price of all products in the Books category, increasing it by 10% Query.

SELECT

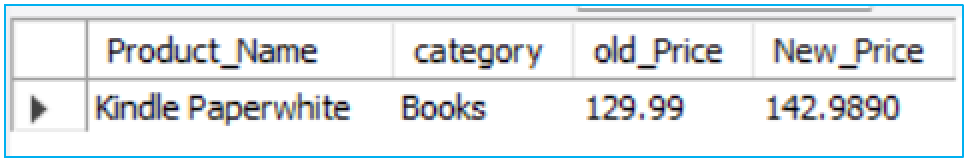
name AS Product_Name,

category,

price AS old_Price,

(price+(price * 0.10)) AS New_Price

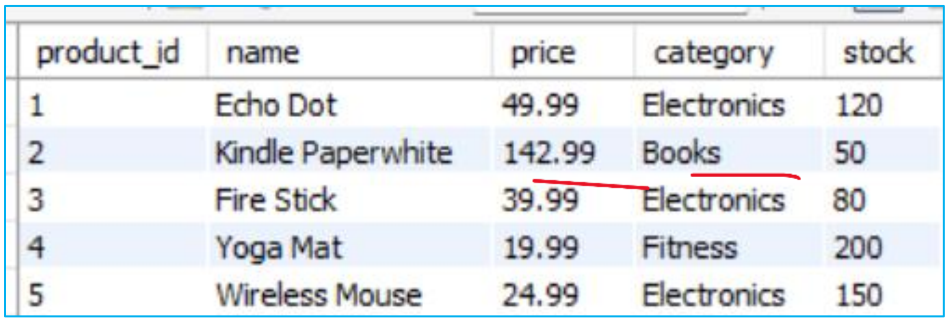
FROM products WHERE category="Books";



Product_Name	category	old_Price	New_Price
Kindle Paperwhite	Books	129.99	142.9890

UPDATE products SET price = (price+(price * 0.10)) WHERE category = "Books";

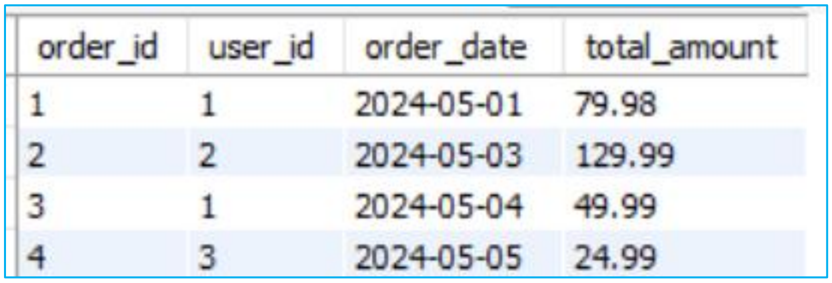
SELECT * from products;



product_id	name	price	category	stock
1	Echo Dot	49.99	Electronics	120
2	Kindle Paperwhite	142.99	Books	50
3	Fire Stick	39.99	Electronics	80
4	Yoga Mat	19.99	Fitness	200
5	Wireless Mouse	24.99	Electronics	150

-- 7. Remove all orders that were placed before 2020.

DELETE FROM orders where order_date < '2020-01-01';



order_id	user_id	order_date	total_amount
1	1	2024-05-01	79.98
2	2	2024-05-03	129.99
3	1	2024-05-04	49.99
4	3	2024-05-05	24.99

-- 8. Write a query to fetch the order details, including customer name, product name, and

-- quantity, for orders placed on 2024-05-01.

SELECT

users.name AS Customer_Name,

products.name AS Ordered_Product,

orderdetails.quantity AS Ordered_Quantity,

orders.order_date AS Order_Date

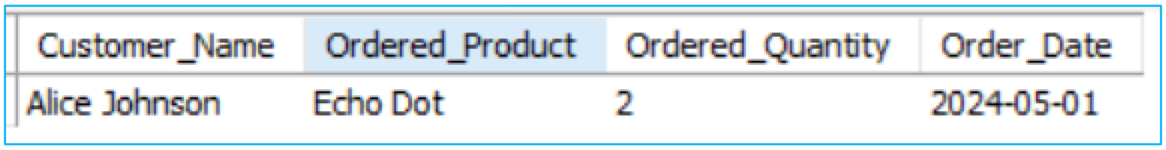
From users

JOIN orders ON users.user_id = orders.user_id

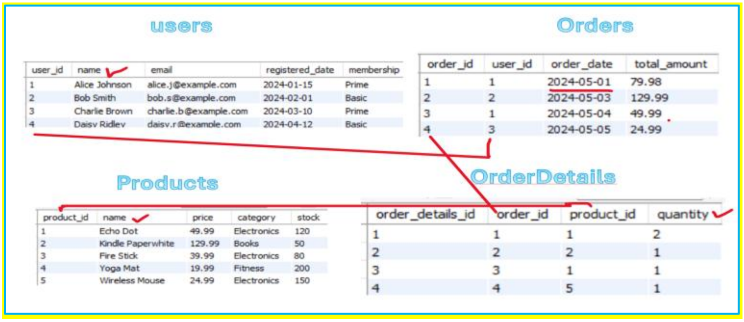
JOIN orderdetails ON orders.order_id = orderdetails.order_id

JOIN products ON orderdetails.product_id = products.product_id

WHERE orders.order_date = "2024-05-01";



Customer_Name	Ordered_Product	Ordered_Quantity	Order_Date
Alice Johnson	Echo Dot	2	2024-05-01



users

user_id	name ✓	email	registered_date	membership
1	Alice Johnson	alice.j@example.com	2024-01-15	Prime
2	Bob Smith	bob.s@example.com	2024-02-01	Basic
3	Charlie Brown	charlie.b@example.com	2024-03-10	Prime
4	Daisy Ridley	daisy.r@example.com	2024-04-12	Basic

Orders

order_id	user_id	order_date	total_amount
1	1	2024-05-01	79.98
2	2	2024-05-03	129.99
3	1	2024-05-04	49.99
4	3	2024-05-05	24.99

Products

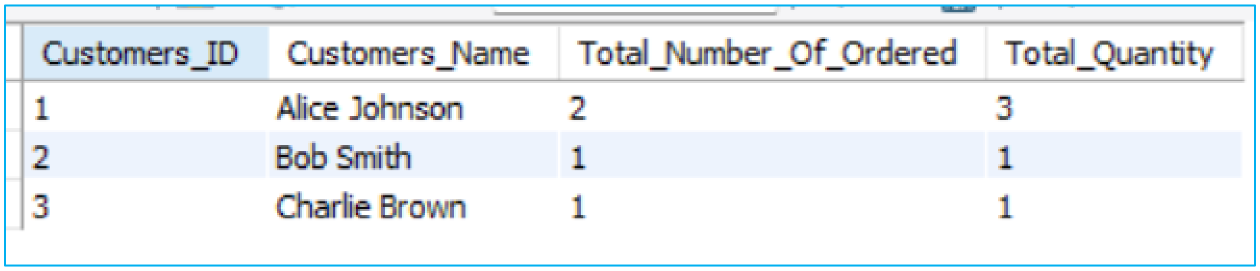
product_id	name ✓	price	category	stock
1	Echo Dot	49.99	Electronics	120
2	Kindle Paperwhite	129.99	Books	50
3	Fire Stick	39.99	Electronics	80
4	Yoga Mat	19.99	Fitness	200
5	Wireless Mouse	24.99	Electronics	150

OrderDetails

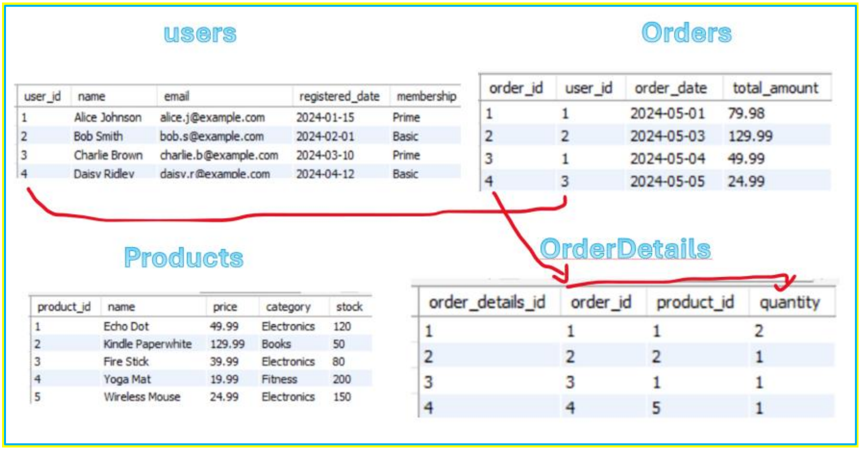
order_details_id	order_id	product_id	quantity ✓
1	1	1	2
2	2	2	1
3	3	1	1
4	4	5	1

-- 9. Fetch all customers and the total number of orders they have placed.

```
SELECT

users.user_id AS Customers_ID,

users.name AS Customers_Name,

COUNT(orderdetails.product_id) AS Total_Number_Of_Ordered,

Sum(orderdetails.quantity) AS Total_Quantity

FROM users

LEFT JOIN orders ON users.user_id = orders.user_id

JOIN orderdetails ON orders.order_id = orderdetails.order_id

GROUP BY users.name, users.user_id ;
```



Customers_ID	Customers_Name	Total_Number_Of_Ordered	Total_Quantity
1	Alice Johnson	2	3
2	Bob Smith	1	1
3	Charlie Brown	1	1



user_id	name	email	registered_date	membership
1	Alice Johnson	alice.j@example.com	2024-01-15	Prime
2	Bob Smith	bob.s@example.com	2024-02-01	Basic
3	Charlie Brown	charlie.b@example.com	2024-03-10	Prime
4	Daisy Ridley	daisy.r@example.com	2024-04-12	Basic

order_id	user_id	order_date	total_amount
1	1	2024-05-01	79.98
2	2	2024-05-03	129.99
3	1	2024-05-04	49.99
4	3	2024-05-05	24.99

product_id	name	price	category	stock
1	Echo Dot	49.99	Electronics	120
2	Kindle Paperwhite	129.99	Books	50
3	Fire Stick	39.99	Electronics	80
4	Yoga Mat	19.99	Fitness	200
5	Wireless Mouse	24.99	Electronics	150

order_details_id	order_id	product_id	quantity
1	1	1	2
2	2	2	1
3	3	1	1
4	4	5	1

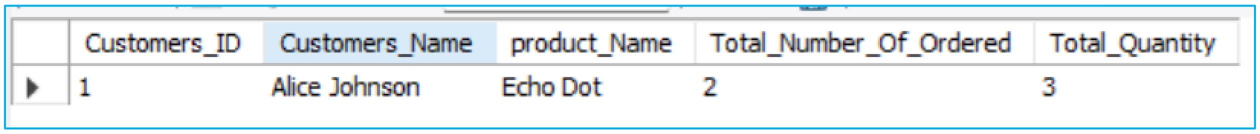
-- 10. Retrieve the average rating for all products in the Electronics category.

-- No rating column exists in the current database

-- 11. List all customers who purchased more than 1 units of any product, including the product name and total quantity purchased.

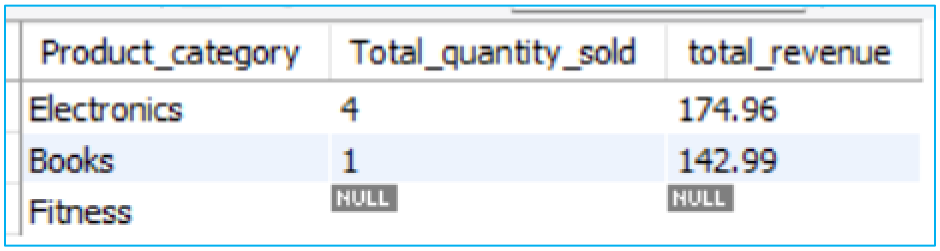
```
SELECT
```

```
users.user_id AS Customers_ID,
users.name AS Customers_Name,
products.name AS Product_Name,
COUNT(orderdetails.product_id) AS Total_Number_Of_Ordered,
Sum(orderdetails.quantity) AS Total_Quantity
FROM users
LEFT JOIN orders ON users.user_id = orders.user_id
JOIN orderdetails ON orders.order_id = orderdetails.order_id
JOIN products ON orderdetails.product_id = products.product_id
GROUP BY users.name, users.user_id, products.name
HAVING Total_Quantity > 1;
```



Customers_ID	Customers_Name	product_Name	Total_Number_Of_Ordered	Total_Quantity
1	Alice Johnson	Echo Dot	2	3

```
-- 12. Find the total revenue generated by each category along with the category name.
SELECT
products.category AS Product_category,
sum(orderdetails.quantity) AS Total_quantity_sold,
sum(orderdetails.quantity * products.price) AS Total_revenue
FROM products
LEFT JOIN orderdetails ON orderdetails.product_id = products.product_id
GROUP BY products.category;
```



Product_category	Total_quantity_sold	total_revenue
Electronics	4	174.96
Books	1	142.99
Fitness	NULL	NULL