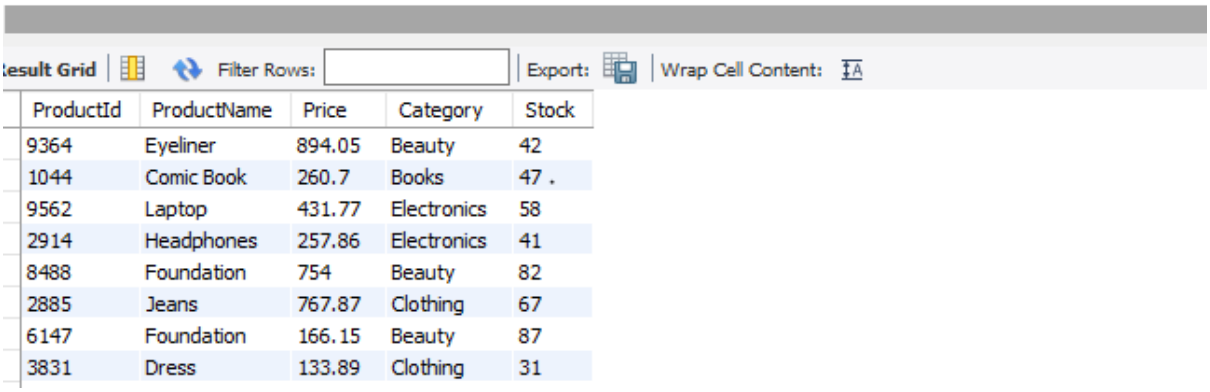
ONLINE RETAIL STORE DATABASE

TASK 01

Develop a database for an online retail store, including products, customers, orders, and payments. This project involves more complex queries and database design. Design tables for products, customers, orders, and payments. Write SQL queries to handle customer orders and payment processing.

List all products

```
1 •  Use retail_sales;
2 •  SELECT * FROM products;
```



ProductId	ProductName	Price	Category	Stock
9364	Eyeliner	894.05	Beauty	42
1044	Comic Book	260.7	Books	47
9562	Laptop	431.77	Electronics	58
2914	Headphones	257.86	Electronics	41
8488	Foundation	754	Beauty	82
2885	Jeans	767.87	Clothing	67
6147	Foundation	166.15	Beauty	87
3831	Dress	133.89	Clothing	31

Find the total number of products available

```
2 • SELECT COUNT(*) AS total_products FROM products;
```

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

total_products
100

Get the total amount of a specific order

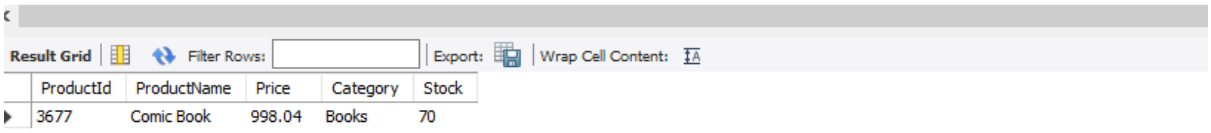
```
1 • Use retail_sales;
2 • SELECT ROUND(SUM(Amount), 2) AS Total_Amount
3   FROM payments
4   WHERE OrderId = 8786;
5
```

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

Total_Amount
1758.37

Find the most expensive product

```
1 • Use retail_sales;
2 • SELECT * FROM products ORDER BY price DESC LIMIT 1;
3
```

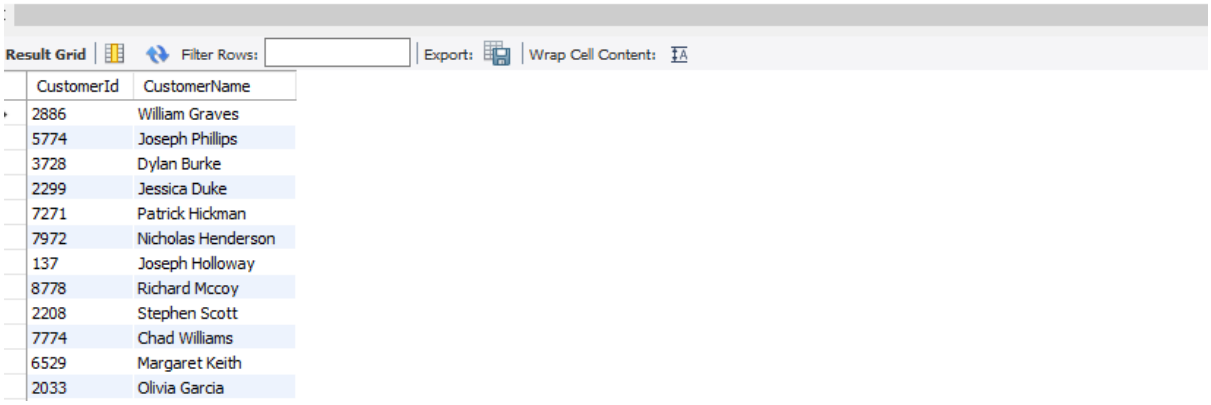


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ProductId	ProductName	Price	Category	Stock
3677	Comic Book	998.04	Books	70

List all customers who have placed an order

```
1 • Use retail_sales;
2 • SELECT DISTINCT Customers.CustomerId, Customers.CustomerName
3   FROM customers Customers
4   JOIN orders o ON Customers.CustomerId = o.CustomerId;
5
6
```



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

CustomerId	CustomerName
2886	William Graves
5774	Joseph Phillips
3728	Dylan Burke
2299	Jessica Duke
7271	Patrick Hickman
7972	Nicholas Henderson
137	Joseph Holloway
8778	Richard Mccoy
2208	Stephen Scott
7774	Chad Williams
6529	Margaret Keith
2033	Olivia Garcia

Calculate the total sales (sum of all orders)

```
1 • Use retail_sales;
2 • SELECT SUM(payments.Amount) AS Total_Sales
3   FROM orders Orders
4   JOIN payments payments ON Orders.OrderId = Payments.OrderId;
5
6
```

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

Total_Sales
95554.62

List all payments made within the last month

```
1 • Use retail_sales;
2 • SELECT * FROM payments
3   WHERE paymentdate >= CURDATE() - INTERVAL 1 MONTH;
4
5
6
```

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

PaymentId	OrderId	PaymentMethod	Amount	PaymentDate
9135	2732	Debit Card	336.2	2024-08-22
9090	2594	Credit Card	632.27	2024-08-26
2423	1338	Net Banking	957.3	2024-08-09
6161	691	UPI	738.35	2024-08-09
9619	3215	PayPal	269.85	2024-08-25

Find the top 5 customers with the highest total order value

```
1 • Use retail_sales;
2 • SELECT customers.customerId, Customers.CustomerName, SUM(payments.amount) AS Total_Spent
3   FROM customers customers
4   JOIN orders orders ON customers.customerId = orders.customerId
5   JOIN payments payments ON orders.OrderId = payments.OrderId
6   GROUP BY customers.customerId, customers.CustomerName
7   ORDER BY total_spent DESC
8   LIMIT 5;
9
10
```

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

customerId	CustomerName	Total_Spent
1841	Terry Wagner	10053.909999999998
2886	William Graves	7176.7
358	Donna Pineda	4546.67
1212	Robert Walker	4500.699999999999
7271	Patrick Hickman	4279

Calculate the average order value by customer

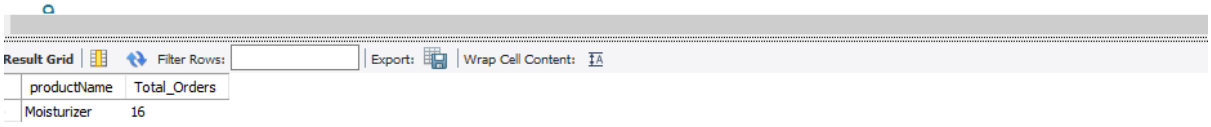
```
1 • Use retail_sales;
2 • SELECT customers.CustomerId, customers.CustomerName, AVG(payments.Amount) AS Avg_Order_Value
3   FROM customers
4   JOIN orders ON customers.CustomerId = orders.CustomerId
5   JOIN payments ON orders.OrderId = payments.OrderId
6   GROUP BY customers.CustomerId, customers.CustomerName;
7
```

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

CustomerId	CustomerName	Avg_Order_Value
1841	Terry Wagner	628.3693749999999
650	Julie Taylor	447.884
6939	Robert Yang	351.98125000000005
3989	Teresa Oliver	551.9216666666667
311	Teresa Zhang	471.345
8778	Richard Mccoy	551.168
5260	Katelyn Martinez	408.0233333333333
7271	Patrick Hickman	356.5833333333333
5774	Joseph Phillips	409.9719999999999
3241	Jessica Hill	252.35

Get the product with the highest number of orders

```
1 • Use retail_sales;
2 • SELECT products.productName, COUNT(orders.orderId) AS Total_Orders
3   FROM products products
4   JOIN orders orders ON Products.productId = orders.productId
5   GROUP BY products.productName
6   ORDER BY Total_Orders DESC
7   LIMIT 1;
8
```



productName	Total_Orders
Moisturizer	16

Find customers who haven't placed any orders

```
1 • Use retail_sales;
2 • SELECT Customers.customerId, CustomerName
3   FROM customers customers
4   LEFT JOIN orders orders ON Customers.customerId = orders.customerId
5   WHERE orders.orderId IS NULL;
6
7
8
```

customerId	CustomerName
4732	Robert Rogers
7699	Aaron Gallagher

Calculate the revenue generated per product category

```
1 • Use retail_sales;
2 • SELECT category, sum(price) AS Total_Revenue
3   FROM products
4   GROUP BY category
5   ORDER BY Total_Revenue DESC;
6
7
8
```

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

category	Total_Revenue
Beauty	11417.149999999998
Books	8966.01
Home	8681.91
Electronics	8616.17
Clothing	8340.460000000001

Monthly sales trends (total sales per month)

```
1 • Use retail_sales;
2 • SELECT DATE_FORMAT(orders.orderDate, '%Y-%m') AS month, SUM(payments.Amount) AS Total_Sales
3   FROM orders orders
4   JOIN payments payments ON orders.orderId = payments.orderId
5   GROUP BY month
6   ORDER BY month;
7
8
```

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

month	Total_Sales
2024-01	9298.720000000001
2024-02	9531.93
2024-03	11766.859999999997
2024-04	12481.9
2024-05	9945.18
2024-06	17026.979999999996
2024-07	11349.8
2024-08	14153.25

Customer segmentation based on total spending

```sql
Use retail_sales;
SELECT customers.customerId, customers.customerName,
       CASE
           WHEN SUM(payments.amount) > 10000 THEN 'VIP'
           WHEN SUM(payments.amount) BETWEEN 5000 AND 10000 THEN 'Regular'
           ELSE 'Occasional'
       END AS customer_segment
FROM customers customers
JOIN orders orders ON customers.customerId = orders.customerId
JOIN payments payments ON orders.orderId = payments.orderId
GROUP BY customers.customerId, customers.customerName;

```

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

customerId	customerName	customer_segment
1841	Terry Wagner	VIP
650	Julie Taylor	Occasional
6939	Robert Yang	Occasional
3989	Teresa Oliver	Occasional
311	Teresa Zhang	Occasional
8778	Richard Mccoy	Occasional
5260	Katelyn Martinez	Occasional
7271	Patrick Hickman	Occasional
5774	Joseph Phillips	Occasional

Identify the most profitable payment method

```sql
Use retail_sales;
SELECT paymentMethod, Round(SUM(payments.amount),2) AS Total_Revenue
FROM payments
JOIN orders ON payments.orderId = orders.orderId
GROUP BY paymentMethod
ORDER BY Total_Revenue Desc
;

```

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

paymentMethod	Total_Revenue
PayPal	21819.11
Credit Card	21337.95
Net Banking	18955.7
Debit Card	18447.54
UPI	14994.32

Find the repeat customers (customers who placed more than one order)

```
1 • Use retail_sales;
2 • SELECT c.customerId, c.customerName, COUNT(o.orderId) AS order_count
3   FROM customers c
4   JOIN orders o ON c.customerId = o.customerId
5   GROUP BY c.customerId, c.customerName
6   HAVING order_count > 1;
7
```

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

customerId	customerName	order_count
2886	William Graves	9
5774	Joseph Phillips	7
3728	Dylan Burke	4
2299	Jessica Duke	4
7271	Patrick Hickman	9
7972	Nicholas Henderson	5
137	Joseph Holloway	2
8778	Richard Mccoy	6
7774	Chad Williams	6
6529	Margaret Keith	2
2033	Olivia Garcia	4
9979	Jennifer Norris	4

Query customer purchase history

```
1 • Use retail_sales;
2 • SELECT c.customerName, p.productName, ord.orderDate, pay.amount
3   FROM customers c
4   JOIN orders ord ON c.customerId = ord.customerId
5   JOIN products p ON ord.productId = p.productId
6   JOIN payments pay ON ord.orderId = pay.orderId
7   WHERE c.customerId = 4790;
8
9
```

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

customerName	productName	orderDate	amount
Douglas Parrish	Non-Fiction Book	2024-04-05	257.33
Douglas Parrish	Perfume	2024-04-17	389