**🔹 Question:**  
You are given an **orders** collection that stores customer purchases and a **products** collection that contains product details.

Write a MongoDB aggregation query to:

1. **Filter only completed orders**.
2. **Join the products collection** to get product details for each order.
3. **Flatten the product details array** so that each product appears as a separate document.
4. **Group the data by customerId** to calculate the **total amount spent** by each customer and collect their purchased products.
5. **Sort customers by total amount spent in descending order**.
6. **Skip the top spender (pagination)**.
7. **Limit the output to the second highest spender**.
8. **Project the final result** with customerId, totalAmountSpent, and purchased products.

**📌 Expected Output Example:**

{

"customerId": "C1023",

"totalAmountSpent": 25000,

"products": [

{

"productId": "P2001",

"productName": "Laptop",

"productPrice": 50000,

"amount": 20000

},

{

"productId": "P2002",

"productName": "Smartphone",

"productPrice": 25000,

"amount": 5000

}

]

}

✅ **Can you write the MongoDB aggregation query for this scenario?**

### ****1️⃣ Create the Database****

use ecommerce;

### ****2️⃣ Create the**** products ****Collection****

db.products.insertMany([

{ \_id: 1, name: "Laptop", price: 50000 },

{ \_id: 2, name: "Smartphone", price: 25000 },

{ \_id: 3, name: "Tablet", price: 15000 },

{ \_id: 4, name: "Smartwatch", price: 10000 },

{ \_id: 5, name: "Headphones", price: 5000 },

{ \_id: 6, name: "Gaming Console", price: 40000 },

{ \_id: 7, name: "Camera", price: 35000 },

{ \_id: 8, name: "Wireless Mouse", price: 2000 },

{ \_id: 9, name: "Keyboard", price: 3000 },

{ \_id: 10, name: "Monitor", price: 15000 }

]);

### ****3️⃣ Create the**** orders ****Collection****

db.orders.insertMany([

{ \_id: 101, customerId: "C1001", productId: 1, amount: 50000, status: "completed" },

{ \_id: 102, customerId: "C1001", productId: 2, amount: 25000, status: "completed" },

{ \_id: 103, customerId: "C1002", productId: 3, amount: 15000, status: "completed" },

{ \_id: 104, customerId: "C1002", productId: 4, amount: 10000, status: "completed" },

{ \_id: 105, customerId: "C1003", productId: 5, amount: 5000, status: "completed" },

{ \_id: 106, customerId: "C1003", productId: 6, amount: 40000, status: "completed" },

{ \_id: 107, customerId: "C1004", productId: 7, amount: 35000, status: "completed" },

{ \_id: 108, customerId: "C1004", productId: 8, amount: 2000, status: "completed" },

{ \_id: 109, customerId: "C1005", productId: 9, amount: 3000, status: "completed" },

{ \_id: 110, customerId: "C1005", productId: 10, amount: 15000, status: "completed" },

{ \_id: 111, customerId: "C1006", productId: 1, amount: 50000, status: "completed" },

{ \_id: 112, customerId: "C1006", productId: 2, amount: 25000, status: "completed" },

{ \_id: 113, customerId: "C1007", productId: 3, amount: 15000, status: "completed" },

{ \_id: 114, customerId: "C1007", productId: 4, amount: 10000, status: "completed" },

{ \_id: 115, customerId: "C1008", productId: 5, amount: 5000, status: "completed" },

{ \_id: 116, customerId: "C1008", productId: 6, amount: 40000, status: "completed" },

{ \_id: 117, customerId: "C1009", productId: 7, amount: 35000, status: "completed" },

{ \_id: 118, customerId: "C1009", productId: 8, amount: 2000, status: "completed" },

{ \_id: 119, customerId: "C1010", productId: 9, amount: 3000, status: "completed" },

{ \_id: 120, customerId: "C1010", productId: 10, amount: 15000, status: "completed" }

]);

**Solution:**