Complex Queries and Aggregations in MongoDB

Consider an e-commerce system. Your task is to write Python code using PyMongo to perform advanced filtering, sorting, and aggregation operations.

Dataset Description:

The collection is named orders and contains documents with the following fields:

```
{


    "order_id": "001",
    "customer": {
        "name": "John Doe",
        "email": "john.doe@example.com"
    },
    "items": [
        {"product": "Laptop", "quantity": 1, "price": 1000},
        {"product": "Mouse", "quantity": 2, "price": 50}
    ],
    "order_date": "2024-01-15",
    "status": "shipped",
    "total": 1100
}
```

Task 1: Filtering Data

Write a query to find all orders placed by customers with the name ”John Doe” and with an order total greater than $500. Display the order ID and total amount in the output.

Expected Output:

```
# Example output format
{
    "order_id": "001",
    "total": 1100
}
```

Task 2: Sorting Data

Write a query to retrieve all orders and sort them by order date in descending order and total amount in ascending order.

Expected Output:

```
# Example output
[
    {"order_id": "002", "order_date": "2024-01-20", "total": 300},
    {"order_id": "001", "order_date": "2024-01-15", "total": 1100}
]
```

Task 3: Aggregation - Total Sales per Product

Use the aggregation framework to calculate the total sales for each product. Group by product name and calculate the total revenue generated by each product.

Expected Output:

```
# Example output format
[
    {"_id": "Laptop", "totalSales": 5000},
    {"_id": "Mouse", "totalSales": 300}
]
```

Task 4: Aggregation - Average Order Value per Customer

Use the aggregation framework to calculate the average order value per customer. Group by customer name and compute the average value of the orders they have placed.

Expected Output:

```
# Example output format
[
    {"_id": "John Doe", "averageOrderValue": 750},
    {"_id": "Jane Smith", "averageOrderValue": 500}
]
```

Task 5: Advanced Aggregation - Top 5 Products by Quantity Sold

Write an aggregation query that returns the top 5 products based on the quantity sold. Sort the results by quantity in descending order.

Expected Output:

```
# Example output format
[
    {"_id": "Laptop", "quantitySold": 50},
    {"_id": "Mouse", "quantitySold": 30},
    {"_id": "Monitor", "quantitySold": 25}
]
```