db.orders.insertMany([

{

customerName: "John Doe",

orderDate: new Date("2022-03-15"),

products: [{ productName: "Laptop", price: 45000, quantity: 1 }],

mobileNo: "1234567890",

totalAmount: 45000

},

{

customerName: "Alice Smith",

orderDate: new Date("2022-05-10"),

products: [

{ productName: "Mobile", price: 15000, quantity: 2 },

{ productName: "PenDrive", price: 500, quantity: 1 }

],

mobileNo: "2345678901",

totalAmount: 30500

},

{

customerName: "Bob Johnson",

orderDate: new Date("2022-02-20"),

products: [

{ productName: "Headphones", price: 3000, quantity: 1 },

{ productName: "PenDrive", price: 500, quantity: 1 }

],

mobileNo: "3456789012",

totalAmount: 3500

},

{

customerName: "Charlie Brown",

orderDate: new Date("2022-06-01"),

products: [{ productName: "Tablet", price: 22000, quantity: 1 }],

mobileNo: "4567890123",

totalAmount: 22000

},

{

customerName: "ABC",

orderDate: new Date("2022-03-25"),

products: [{ productName: "Laptop", price: 27000, quantity: 1 }],

mobileNo: "5678901234",

totalAmount: 27000

}

]);  
  
  
1. Display all documents in a collection

db.orders.find().pretty();  
  
2. List the customers in ascending order of their names

db.orders.find({}, { customerName: 1, \_id: 0 }).sort({ customerName: 1 });

3. Display all the orders placed before April 2022

db.orders.find({ orderDate: { $lt: new Date("2022-04-01") } });

4. Display the Name of Customer who purchased an order whose price is more than 25000

db.orders.find({ "products.price": { $gt: 25000 } }, { customerName: 1, \_id: 0 });

5. Display all orders that contain the product "PenDrive"

db.orders.find({ "products.productName": "PenDrive" });

6. Update Order\_date of any order purchased by Customer "ABC"

db.orders.updateOne(

{ customerName: "ABC" },

{ $set: { orderDate: new Date() } }

);

7. List all documents with orders that contain products whose quantity is less than 10

db.orders.find({ "products.quantity": { $lt: 10 } });

8. Display the Mob No of customers who have the highest buying total

db.orders.find().sort({ totalAmount: -1 }).limit(1).forEach(doc => print(doc.mobileNo));

9. Perform Create Index, Get Index, and Drop Index operations on the collection

db.orders.createIndex({ customerName: 1 });

db.orders.getIndexes();

db.orders.dropIndex("customerName\_1");

10. Using MapReduce/Aggregation, display the total orders per customer

db.orders.aggregate([

{

$group: {

\_id: "$customerName",

totalOrders: { $sum: 1 }

}

}

]);