JSON dataset


```
[
  {
    "_id": { "$oid": "65c2f1234a1b5e7890ab1234" },
    "name": "Alice Johnson",
    "email": "alice@example.com",
    "age": 28,
    "isActive": true,
    "createdAt": { "$date": "2023-01-15T08:30:00Z" },
    "orders": [
      {
        "orderId": { "$oid": "65c2f1234a1b5e7890ab5678" },
        "amount": 250.75,
        "items": ["Laptop", "Mouse"],
        "orderDate": { "$date": "2023-02-10T12:45:00Z" },
        "status": "Delivered"
      },
      {
        "orderId": { "$oid": "65c2f1234a1b5e7890ab9876" },
        "amount": 50.0,
        "items": ["Keyboard"],
        "orderDate": { "$date": "2023-03-05T14:00:00Z" },
        "status": "Shipped"
      }
    ],
    "address": {
      "street": "123 Main St",
      "city": "New York",
      "zipcode": "10001"
    },
    "preferences": {
      "newsletter": true,
      "notifications": ["email", "sms"]
    }
  },
  {
    "_id": { "$oid": "65c2f5678a1b5e7890ab5678" },
    "name": "Bob Smith",
    "email": "bob@example.com",
    "age": 35,
    "isActive": false,
    "createdAt": { "$date": "2022-11-20T10:15:00Z" },
    "orders": [
      {
        "orderId": { "$oid": "65c2f5678a1b5e7890ab9999" },
        "amount": 120.99,
        "items": ["Tablet"],
        "orderDate": { "$date": "2023-04-12T09:30:00Z" },
        "status": "Pending"
      }
    ],
    "address": {
      "street": "456 Park Ave",
      "city": "Los Angeles",
      "zipcode": "90012"
    },
    "preferences": {
      "newsletter": false,
      "notifications": ["email"]
    }
  },
  {
    "_id": { "$oid": "65c2f9999a1b5e7890ab8888" },
    "name": "Charlie Brown",
    "email": "charlie@example.com",
    "age": 40,
    "isActive": true,
    "createdAt": { "$date": "2023-06-01T16:45:00Z" },
    "orders": [],
    "address": {
      "street": "789 Broadway",
      "city": "San Francisco",
      "zipcode": "94105"
    },
    "preferences": {
      "newsletter": true,
      "notifications": ["push"]
    }
  }
]
```

✅ Create Operations (insertOne(), insertMany())

📌 Insert a Single User (insertOne)

```
db.users.insertOne({
    name: "David Miller",
    email: "david@example.com",
    age: 29,
    isActive: true,
    createdAt: new Date("2023-08-21T10:00:00Z"),
    orders: [
      {
        orderId: ObjectId(),
        amount: 89.99,
        items: ["Smartwatch"],
        orderDate: new Date("2023-09-01T14:30:00Z"),
        status: "Delivered"
      }
    ],
    address: {
      street: "987 Sunset Blvd",
      city: "Chicago",
      zipcode: "60601"
    },
    preferences: {
      newsletter: true,
      notifications: ["email", "push"]
    }
  });
```

📌 Insert Multiple Users (insertMany)

```
db.users.insertMany([
   {
     name: "Eve Adams",
     email: "eve@example.com",
     age: 33,
     isActive: false,
     createdAt: new Date("2023-07-10T12:15:00Z"),
     orders: [],
     address: {
       street: "159 Pine St",
       city: "Seattle",
       zipcode: "98101"
     },
     preferences: {
       newsletter: false,
       notifications: ["sms"]
     }
   },
   {
     name: "Frank White",
     email: "frank@example.com",
     age: 45,
     isActive: true,
     createdAt: new Date("2023-06-25T09:45:00Z"),
     orders: [
       {
         orderId: ObjectId(),
         amount: 150.50,
         items: ["Headphones", "Charger"],
         orderDate: new Date("2023-07-15T16:00:00Z"),
         status: "Shipped"
       }
     ],
     address: {
       street: "789 Elm St",
       city: "Boston",
       zipcode: "02108"
     },
     preferences: {
       newsletter: true,
       notifications: ["email"]
     }
   }
 ]);
```

✅ Read Operations (find(), findOne())

📌 Find All Users (find())

```
db.users
.find();
```

📌 Find One User (findOne())

```
db.users
.findOne({ email: "alice@example.com" });
```

📌 Find Users Older Than 30 ($gt - Greater Than)

```
db.users
.find({ age: { $gt: 30 } });
```

📌 Find Users Aged Between 25 and 40 ($gte, $lte)

```
db.users
.find({ age: { $gte: 25, $lte: 40 } });
```

📌 Find Users from Specific Cities ($in Operator)

```
db.users
.find({ "address.city": { $in: ["New York", "Los Angeles"] } });
```

✅ Update Operations (updateOne(), updateMany())

📌 Update a User's Age ($set Operator)

```
db.users
.updateOne({ email: "alice@example.com" }, { $set: { age: 30 } });
```

📌 Update Multiple Users: Set isActive: false for Users Over 40 (updateMany())

```
db.users
.updateMany({ age: { $gt: 40 } }, { $set: { isActive: false } });
```

📌 Remove a Field ($unset Operator)

```
db.users
.updateOne({ email: "alice@example.com"},{ $unset: { preferences: ""}});
```

📌 Increment Age by 1 ($inc Operator)

```
db.users
.updateOne({ email: "bob@example.com" }, { $inc: { age: 1 } });
```

📌 Add a New Notification Method ($push Operator)

```
db.users
.updateOne({ email: "alice@example.com" },
{ $push: { "preferences.notifications": "whatsapp" } });
```

📌 Remove an Item from Notifications ($pull Operator)

```
db.users
.updateOne({ email: "alice@example.com" },
{ $pull: { "preferences.notifications": "sms" } });
```

✅ Delete Operations (deleteOne(), deleteMany())

📌 Delete a Single User (deleteOne())

```
db.users.deleteOne({ email: "alice@example.com" });
```

📌 Delete All Inactive Users (deleteMany())

```
db.users.deleteMany({ isActive: false });
```

Ecommerce

Users.JSON


```
[
    {
      "_id": "507f1f77bcf86cd799439011",
      "name": "John Doe",
      "email": "john.doe@example.com",
      "age": 30,
      "address": {
        "city": "New York",
        "zip": "10001"
      },
      "orders": [
        "607f1f77bcf86cd799439021",
        "607f1f77bcf86cd799439022"
      ]
    },
    {
      "_id": "507f1f77bcf86cd799439012",
      "name": "Jane Smith",
      "email": "jane.smith@example.com",
      "age": 25,
      "address": {
        "city": "Los Angeles",
        "zip": "90001"
      },
      "orders": [
        "607f1f77bcf86cd799439023"
      ]
    }
  ]
```

products.JSON


```
[
  {
    "_id": "607f1f77bcf86cd799439031",
    "name": "Laptop",
    "price": 1200,
    "category": "Electronics",
    "stock": 50
  },
  {
    "_id": "607f1f77bcf86cd799439032",
    "name": "Smartphone",
    "price": 800,
    "category": "Electronics",
    "stock": 100
  },
  {
    "_id": "607f1f77bcf86cd799439033",
    "name": "Headphones",
    "price": 150,
    "category": "Accessories",
    "stock": 200
  }
]
```

orders.JSON


```
[
    {
        "_id": "607f1f77bcf86cd799439021",
        "userId": "507f1f77bcf86cd799439011",
        "products": [
            {
                "productId": "607f1f77bcf86cd799439031",
                "quantity": 1
            },
            {
                "productId": "607f1f77bcf86cd799439033",
                "quantity": 2
            }
        ],
        "totalAmount": 1500,
        "status": "completed",
        "orderDate": "2023-10-01T10:00:00Z"
    },
    {
        "_id": "607f1f77bcf86cd799439022",
        "userId": "507f1f77bcf86cd799439011",
        "products": [
            {
                "productId": "607f1f77bcf86cd799439032",
                "quantity": 1
            }
        ],
        "totalAmount": 800,
        "status": "pending",
        "orderDate": "2023-10-02T12:00:00Z"
    },
    {
        "_id": "607f1f77bcf86cd799439023",
        "userId": "507f1f77bcf86cd799439012",
        "products": [
            {
                "productId": "607f1f77bcf86cd799439033",
                "quantity": 1
            }
        ],
        "totalAmount": 150,
        "status": "completed",
        "orderDate": "2023-10-03T14:00:00Z"
    }
]
```

1. CRUD Operations

```
Insert (with string ID for _id):
     db.users.insertOne({
     "_id": "507f1f77bcf86cd799439013",  // String ID
      "name": "Alice Brown",
     "email": "alice.brown@example.com",
     "age": 28,
      "address": {
         "city": "Chicago",
         "zip": "60601"
       },
       "orders": []
     });


Update (with string ID for _id):
     db.users.updateOne(
       { _id: "507f1f77bcf86cd799439011" },  String ID
       { $set: { age: 31 } }
      );


 Delete (with string ID for _id):
     db.users
     .deleteOne({ _id: "507f1f77bcf86cd799439013" });  // String ID
```

2. Querying

```
Find all users older than 25:
     db.users.find({ age: { $gt: 25 } });

Find orders with status "completed":
     db.orders.find({ status: "completed" });

Find products in the "Electronics" category:
     db.products.find({ category: "Electronics" });
```

3. Aggregation

```
Total revenue by user:
     db.orders
     .aggregate([{ $group: { _id: "$userId",
     totalRevenue: { $sum: "$totalAmount" } } }
      ]);


Average order amount:
     db.orders
     .aggregate([ { $group: { _id: null,
     averageAmount: { $avg: "$totalAmount" } } }
      ]);
```

4. Indexing

```
Create an index on the email field in the users collection:
     db.users.createIndex({ email: 1 });

Create a compound index on category and price in the products collection:
     db.products.createIndex({ category: 1, price: 1 });
```

5. Transactions

```
Transfer an order from one user to another (with string IDs for _id):
     const session = db.getMongo().startSession();
     session.startTransaction();
     try {
     const users = session.getDatabase("test").users;
     const orders = session.getDatabase("test").orders;

Remove order from user1
 users.updateOne(
   { _id: "507f1f77bcf86cd799439011" },  // String ID
   { $pull: { orders: "607f1f77bcf86cd799439021" } },  // String ID
```

```
    { session }
  );

  Add order to user2
      users.updateOne(
        { _id: "507f1f77bcf86cd799439012" },  // String ID
        { $push: { orders: "607f1f77bcf86cd799439021" } }, // String ID
        { session }
      );

        session.commitTransaction();
      } catch (error) {
       session.abortTransaction();
      } finally {
       session.endSession();
      }
```

6. Sharding

```
Enable sharding on the orders collection:
      sh.enableSharding("test");
      sh.shardCollection("test.orders", { userId: 1 });
```