EXPERIMENT NO. 6

Name of Student	**Shreyash Ganesh Dhekane**
Class Roll No	**16**
D.O.P.	**18-03-2025**
D.O.S.	**25-03-2025**
Sign and Grade	

AIM: To study CRUD operations in MongoDB

PROBLEM STATEMENT:

A) Create a database, create a collection, insert data, query and manipulate data using various MongoDB operations.

1. Create a database named "inventory".
2. Create a collection named "products" with the fields: (ProductID, ProductName, Category, Price, Stock).
3. Insert 10 documents into the "products" collection.
4. Display all the documents in the "products" collection.
5. Display all the products in the "Electronics" category.
6. Display all the products in ascending order of their names.
7. Display the details of the first 5 products.
8. Display the categories of products with a specific name.
9. Display the number of products in the "Electronics" category.
10. Display all the products without showing the "_id" field.
11. Display all the distinct categories of products.
12. Display products in the "Electronics" category with prices greater than 50 but less than 100.
13. Change the price of a product.
14. Delete a particular product entry.

THEORY:

1. **Describe some of the features of MongoDB?**

   MongoDB is a NoSQL , document oriented database that stores data in a flexible, JSON-like format. Some of its key features include:

   Key Features of MongoDB:

   Schema-less (Flexible Data Model) – Unlike relational databases, MongoDB allows documents in the same collection to have different structures, making it highly flexible.

   Scalability (Sharding) – MongoDB supports horizontal scaling by distributing data across multiple servers using sharding, ensuring high availability and performance.

   Replication (High Availability) – Data is replicated across multiple servers using replica sets, preventing data loss and ensuring fault tolerance.

   Indexing – MongoDB supports different types of indexes to improve query performance, reducing the time taken to fetch data.

   Aggregation Framework – Provides powerful tools to perform complex data transformations and analytics, similar to SQL's GROUP BY and JOIN operations.

   A**CID Transactions** – Ensures data consistency by supporting multi-document transactions, making MongoDB suitable for critical applications.

2. **What are Documents and Collections in MongoDB?**

   In MongoDB:

   Documents

   A document is a single record in a MongoDB collection. It is stored in a JSON-like **BSON (Binary JSON)** format and consists of key-value pairs. Each document can have a different structure, providing flexibility.

```
{
  "_id": 1,
  "name": "John Doe",
  "age": 25,
  "city": "New York"
}
```

Collections

A collection is a group of related documents, similar to a table in relational databases. Unlike tables, collections do not enforce a fixed schema, allowing different documents to have varied structures.

```
{ "_id": 1, "name": "John Doe", "age": 25 }
{ "_id": 2, "name": "Jane Smith", "city": "Los Angeles" }
```

3. When to use MongoDB ?

MongoDB is ideal for applications requiring flexibility, scalability, and high-speed data handling. It is best suited for modern, dynamic applications with large and evolving datasets.

Use Cases:

1. **Flexible Schema** – When your data structure is not fixed and evolves over time.
2. **High Read/Write Operations** – Best for applications with heavy real-time data processing.
3. **Scalability** – Suitable for large datasets that require horizontal scaling using sharding.
4. **Real-Time Analytics** – Ideal for applications needing fast data aggregation and analysis.
5. **Geospatial Data** – Useful for location-based services like maps, ride-sharing, and delivery apps.
6. **Cloud-Based Applications** – Works well with cloud-native and distributed applications.

4. What is Sharding in MongoDB?

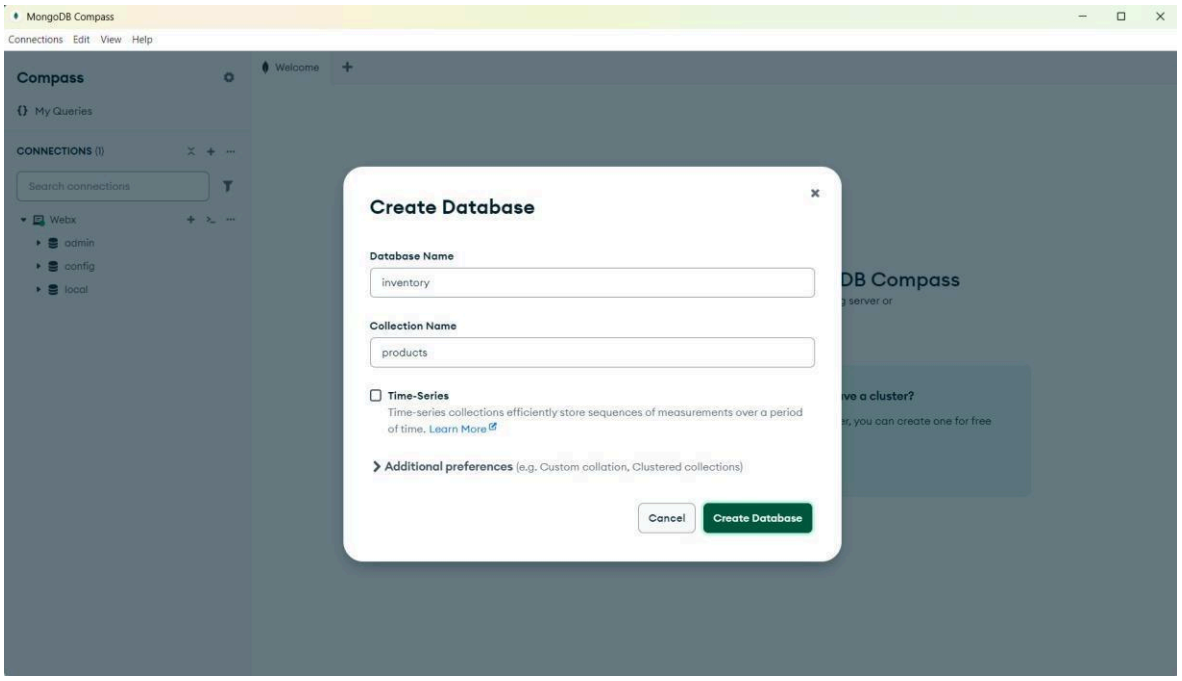
Sharding is a method of distributing large datasets across multiple servers to improve performance and scalability. It helps MongoDB handle massive amounts of data and high traffic efficiently.

How Sharding Works?

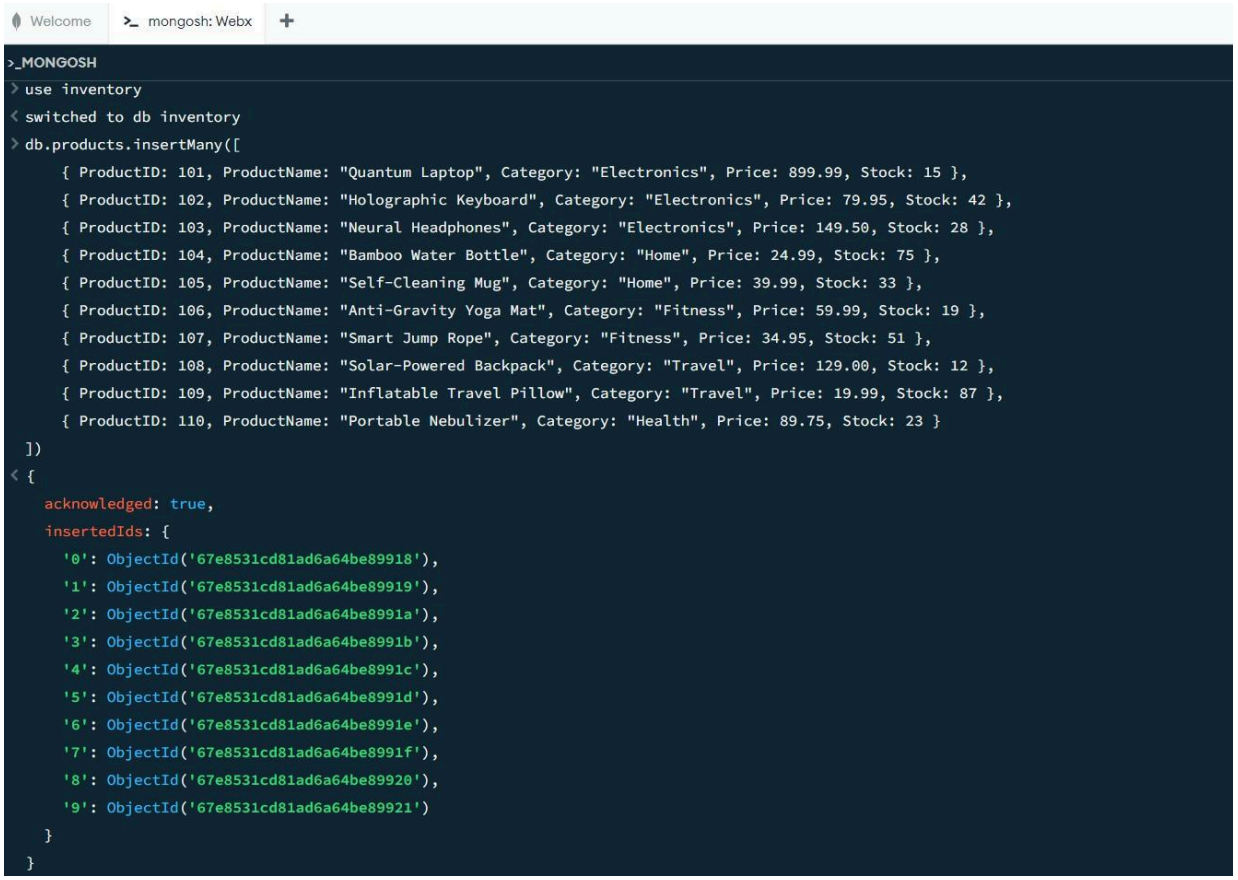
Data is divided into smaller **chunks** and distributed across multiple servers (shards). A **Shard** is a MongoDB instance that stores part of the data.
A **Config Server** keeps track of shard metadata and data distribution.
A **Query Router (mongos)** directs client queries to the appropriate shard.

OUTPUT: -

- Create a database named inventory and create a collection named "products" with the fields: (ProductID, ProductName, Category, Price, Stock)



- Insert 10 documents into the “products” collection



- Display all the documents in the "products" collection.

```
>_MONGOSH
> db.products.find().pretty()
< {
    _id: ObjectId('67e8531cd81ad6a64be89918'),
    ProductID: 101,
    ProductName: 'Quantum Laptop',
    Category: 'Electronics',
    Price: 899.99,
    Stock: 15
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be89919'),
    ProductID: 102,
    ProductName: 'Holographic Keyboard',
    Category: 'Electronics',
    Price: 79.95,
    Stock: 42
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be8991a'),
    ProductID: 103,
    ProductName: 'Neural Headphones',
    Category: 'Electronics',
    Price: 149.5,
    Stock: 28
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be8991b'),
    ProductID: 104,
    ProductName: 'Bamboo Water Bottle',
```

- Display all the products in the "Electronics" category.

```
>_MONGOSH
> db.products.find({ Category: "Electronics" }).pretty()
< {
    _id: ObjectId('67e8531cd81ad6a64be89918'),
    ProductID: 101,
    ProductName: 'Quantum Laptop',
    Category: 'Electronics',
    Price: 899.99,
    Stock: 15
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be89919'),
    ProductID: 102,
    ProductName: 'Holographic Keyboard',
    Category: 'Electronics',
    Price: 79.95,
    Stock: 42
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be8991a'),
    ProductID: 103,
    ProductName: 'Neural Headphones',
    Category: 'Electronics',
    Price: 149.5,
    Stock: 28
  }
```

- Display all the products in ascending order of their names.

```
>_MONGOSH
> db.products.find().sort({ ProductName: 1 }).pretty()
< {
    _id: ObjectId('67e8531cd81ad6a64be8991d'),
    ProductID: 106,
    ProductName: 'Anti-Gravity Yoga Mat',
    Category: 'Fitness',
    Price: 59.99,
    Stock: 19
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be8991b'),
    ProductID: 104,
    ProductName: 'Bamboo Water Bottle',
    Category: 'Home',
    Price: 24.99,
    Stock: 75
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be89919'),
    ProductID: 102,
    ProductName: 'Holographic Keyboard',
    Category: 'Electronics',
    Price: 79.95,
    Stock: 42
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be89920'),
    ProductID: 109,
```

- Display the details of the first 5 products.

```
>_MONGOSH
> db.products.find().limit(5).pretty()
< {
    _id: ObjectId('67e8531cd81ad6a64be89918'),
    ProductID: 101,
    ProductName: 'Quantum Laptop',
    Category: 'Electronics',
    Price: 899.99,
    Stock: 15
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be89919'),
    ProductID: 102,
    ProductName: 'Holographic Keyboard',
    Category: 'Electronics',
    Price: 79.95,
    Stock: 42
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be8991a'),
    ProductID: 103,
    ProductName: 'Neural Headphones',
    Category: 'Electronics',
    Price: 149.5,
    Stock: 28
  }
  {
    _id: ObjectId('67e8531cd81ad6a64be8991b'),
    ProductID: 104,
    ProductName: 'Bamboo Water Bottle',
```

- Display the categories of products with a specific name.

```
>_MONGOSH
> db.products.find({ ProductName: "Quantum Laptop" }, { Category: 1, _id: 0 }).pretty()
< {
    Category: 'Electronics'
  }
```

- Display the number of products in the "Electronics" category.

```
>_MONGOSH
> db.products.countDocuments({ Category: "Electronics" })
< 3
```

- Display all the products without showing the "_id" field.

```
>_MONGOSH
> db.products.find({}, { _id: 0 }).pretty()
< {
    ProductID: 101,
    ProductName: 'Quantum Laptop',
    Category: 'Electronics',
    Price: 899.99,
    Stock: 15
  }
  {
    ProductID: 102,
    ProductName: 'Holographic Keyboard',
    Category: 'Electronics',
    Price: 79.95,
    Stock: 42
  }
  {
    ProductID: 103,
    ProductName: 'Neural Headphones',
    Category: 'Electronics',
    Price: 149.5,
    Stock: 28
  }
  {
    ProductID: 104,
    ProductName: 'Bamboo Water Bottle',
    Category: 'Home',
    Price: 24.99,
    Stock: 75
```

- Display all the distinct categories of products.

```
>_MONGOSH
> db.products.distinct("Category")
< [ 'Electronics', 'Fitness', 'Health', 'Home', 'Travel' ]
```

- Display products in the "Electronics" category with prices greater than 50 but less than 100.

```
>_MONGOSH
> db.products.find({ Category: "Electronics", Price: { $gt: 50, $lt: 100 } }).pretty()
< {
    _id: ObjectId('67e8531cd81ad6a64be89919'),
    ProductID: 102,
    ProductName: 'Holographic Keyboard',
    Category: 'Electronics',
    Price: 79.95,
    Stock: 42
  }
```

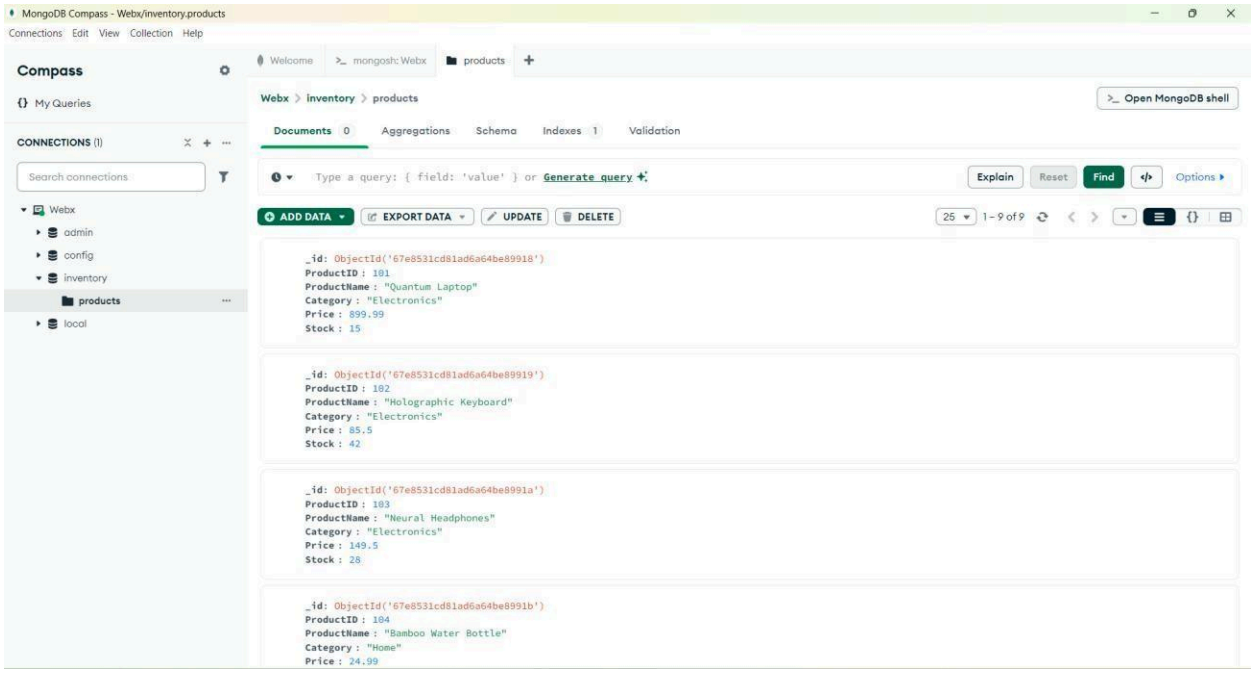
- Change the price of a product.

```
>_MONGOSH
> db.products.updateOne(
    { ProductName: "Holographic Keyboard" },
    { $set: { Price: 85.50 } }
  )
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

- Delete a particular product entry.

```
>_MONGOSH
> db.products.deleteOne({ ProductName: "Inflatable Travel Pillow" })
< {
    acknowledged: true,
    deletedCount: 1
  }
```

- Final data in the database



Conclusion: -

The practical effectively demonstrates CRUD operations in MongoDB, covering database creation, data insertion, querying, updating, and deletion. It showcases how to structure a collection, retrieve data using filters and sorting, and manipulate records efficiently. By performing operations like filtering by category, updating product details, and deleting entries, this exercise highlights MongoDB's flexibility and powerful querying capabilities for managing dynamic datasets.