Experiment – 6: MongoDB

Name of Student	**PRANAV POL**
Class Roll No	**D15A 41**
D.O.P.	
D.O.S.	
Sign and Grade	

1) **Aim:** To study CRUD operations in MongoDB
2) **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.

3) **Theory:**
   A. Describe some of the features of MongoDB?
   B. What are Documents and Collections in MongoDB?
   C. When to use MongoDB?
   D. What is Sharding in MongoDB?

Answers:-

1) MongoDB is a popular NoSQL database known for its scalability and flexibility. Some of its key features include:

- **Document-Oriented Storage**: Instead of rows and columns, MongoDB stores data in JSON-like documents which allows for varied data models.
- **Schema-less**: MongoDB collections do not enforce document structure by default, allowing different documents in the same collection to have different fields.
- **Scalability**: MongoDB supports horizontal scaling via sharding, which allows large databases to be partitioned across many servers.
- **Indexing**: MongoDB supports various indexing techniques to optimize query performance.
- **Aggregation Framework**: Allows for the performance of advanced data processing and analysis operations directly within the database.
- **Replication**: Provides high availability and redundancy with replica sets, ensuring data consistency across multiple servers.
- **ACID Transactions**: Supports multi-document ACID (Atomicity, Consistency, Isolation, Durability) transactions to ensure data integrity.
- **Geospatial Queries**: Provides support for querying geospatial data.

2) **Documents**: In MongoDB, data is stored as documents, which are similar to JSON objects. Each document consists of key-value pairs. Documents are flexible in structure and can store various data types, including nested documents and arrays.
   Collections: Collections are a group of MongoDB documents. They are equivalent to tables in relational databases. Collections do not enforce a schema, meaning documents within a collection can have different structures.

3) MongoDB is suitable for various use cases, including:
   - Applications requiring flexible and dynamic schemas, such as content management systems or user profiles.
   - Big data applications needing horizontal scaling and high throughput, like analytics platforms.
   - Applications needing to store hierarchical data or complex data structures, such as social networks or real-time analytics.
   - Projects requiring geospatial data storage and queries, such as location-based services.
   - Applications demanding high availability and redundancy.

4) Sharding is a method for distributing data across multiple servers. It enables horizontal scaling by partitioning large datasets into smaller, more manageable pieces, called shards. Each shard is a separate database instance, and together, they form a single logical database. Sharding helps in managing large volumes of data, improves read and write performance, and ensures high availability. MongoDB uses a sharding key to determine how data is distributed across shards.

4) **Output:**

```
F:\email-spam-classifier-main>mongosh
Current Mongosh Log ID: 67c9683465201bfe65fa4213
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2
.4.0
Using MongoDB:          8.0.5
Using Mongosh:          2.4.0

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

------
   The server generated these startup warnings when booting
   2025-03-06T13:57:49.477+05:30: Access control is not enabled for the database. Read and write access to data and conf
iguration is unrestricted
------

test> use inventory
switched to db inventory
```

```
inventory> db.createCollection("products")
{ ok: 1 }
inventory> db.products.insertMany([
...     { ProductID: 1, ProductName: "Laptop", Category: "Electronics", Price: 800, Stock: 10 },
...     { ProductID: 2, ProductName: "Smartphone", Category: "Electronics", Price: 500, Stock: 25 },
...     { ProductID: 3, ProductName: "TV", Category: "Electronics", Price: 600, Stock: 15 },
...     { ProductID: 4, ProductName: "Headphones", Category: "Electronics", Price: 150, Stock: 50 },
...     { ProductID: 5, ProductName: "Refrigerator", Category: "Appliances", Price: 1000, Stock: 8 },
...     { ProductID: 6, ProductName: "Washing Machine", Category: "Appliances", Price: 700, Stock: 5 },
...     { ProductID: 7, ProductName: "Microwave", Category: "Appliances", Price: 200, Stock: 20 },
...     { ProductID: 8, ProductName: "Blender", Category: "Appliances", Price: 100, Stock: 30 },
...     { ProductID: 9, ProductName: "Tablet", Category: "Electronics", Price: 300, Stock: 40 },
...     { ProductID: 10, ProductName: "Camera", Category: "Electronics", Price: 450, Stock: 12 }
... ])
...
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('67c9685065201bfe65fa4214'),
    '1': ObjectId('67c9685065201bfe65fa4215'),
    '2': ObjectId('67c9685065201bfe65fa4216'),
    '3': ObjectId('67c9685065201bfe65fa4217'),
    '4': ObjectId('67c9685065201bfe65fa4218'),
    '5': ObjectId('67c9685065201bfe65fa4219'),
    '6': ObjectId('67c9685065201bfe65fa421a'),
    '7': ObjectId('67c9685065201bfe65fa421b'),
    '8': ObjectId('67c9685065201bfe65fa421c'),
    '9': ObjectId('67c9685065201bfe65fa421d')
  }
}
```

```
inventory> db.products.find().pretty()
[
  {
    _id: ObjectId('67c9685065201bfe65fa4214'),
    ProductID: 1,
    ProductName: 'Laptop',
    Category: 'Electronics',
    Price: 800,
    Stock: 10
  },
  {
    _id: ObjectId('67c9685065201bfe65fa4215'),
    ProductID: 2,
    ProductName: 'Smartphone',
    Category: 'Electronics',
    Price: 500,
    Stock: 25
  },
  {
    _id: ObjectId('67c9685065201bfe65fa4216'),
    ProductID: 3,
    ProductName: 'TV',
    Category: 'Electronics',
    Price: 600,
    Stock: 15
  },
```

```
inventory> db.products.find({ Category: "Electronics" }).pretty()
[
  {
    _id: ObjectId('67c9685065201bfe65fa4214'),
    ProductID: 1,
    ProductName: 'Laptop',
    Category: 'Electronics',
    Price: 800,
    Stock: 10
  },
  {
    _id: ObjectId('67c9685065201bfe65fa4215'),
    ProductID: 2,
    ProductName: 'Smartphone',
    Category: 'Electronics',
    Price: 500,
    Stock: 25
  },
  {
    _id: ObjectId('67c9685065201bfe65fa4216'),
    ProductID: 3,
    ProductName: 'TV',
    Category: 'Electronics',
    Price: 600,
    Stock: 15
  },
```

```
inventory> db.products.find().sort({ ProductName: 1 })
[
  {
    _id: ObjectId('67c9685065201bfe65fa421b'),
    ProductID: 8,
    ProductName: 'Blender',
    Category: 'Appliances',
    Price: 100,
    Stock: 30
  },
  {
    _id: ObjectId('67c9685065201bfe65fa421d'),
    ProductID: 10,
    ProductName: 'Camera',
    Category: 'Electronics',
    Price: 450,
    Stock: 12
  },
```

```
inventory> db.products.find().limit(5)
[
  {
    _id: ObjectId('67c9685065201bfe65fa4214'),
    ProductID: 1,
    ProductName: 'Laptop',
    Category: 'Electronics',
    Price: 800,
    Stock: 10
  },
  {
    _id: ObjectId('67c9685065201bfe65fa4215'),
    ProductID: 2,
    ProductName: 'Smartphone',
    Category: 'Electronics',
    Price: 500,
    Stock: 25
  },
  {
    _id: ObjectId('67c9685065201bfe65fa4216'),
    ProductID: 3,
    ProductName: 'TV',
    Category: 'Electronics',
    Price: 600,
    Stock: 15
  },
```

```
inventory> db.products.find({ ProductName: "Laptop" }, { Category: 1, _id: 0 })
[ { Category: 'Electronics' } ]
inventory> db.products.find({ Category: "Electronics" }).count()
6
```

```
inventory> db.products.find({}, { _id: 0 }).pretty()
[
  {
    ProductID: 1,
    ProductName: 'Laptop',
    Category: 'Electronics',
    Price: 800,
    Stock: 10
  },
  {
    ProductID: 2,
    ProductName: 'Smartphone',
    Category: 'Electronics',
    Price: 500,
    Stock: 25
  },
  {
    ProductID: 3,
    ProductName: 'TV',
    Category: 'Electronics',
    Price: 600,
    Stock: 15
  },
```

```
inventory> db.products.distinct("Category")
[ 'Appliances', 'Electronics' ]
inventory>  db.products.find({ Category: "Electronics", Price: { $gt: 50, $lt: 100 } }).pretty()

inventory>  db.products.find({ Category: "Electronics", Price: { $gt: 50, $lt: 100 } }).pretty()

inventory> db.products.updateOne({ ProductName: "Laptop" }, { $set: { Price: 850 } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
inventory> db.products.deleteOne({ ProductName: "Laptop" })
{ acknowledged: true, deletedCount: 1 }
```

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.