

### **EXPERIMENT NO. 6 - MongoDB**

<b>Name of Student</b>	<b>Eesha Chavan</b>
<b>Class Roll No</b>	<b>D15A 08</b>
<b>D.O.P.</b>	
<b>D.O.S.</b>	
<b>Sign and Grade</b>	

**AIM:** To study CRUD operations in MongoDB

#### **PROBLEM STATEMENT:**

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

- a. Create a database named "inventory".
- b. Create a collection named "products" with the fields: (ProductID, ProductName, Category, Price, Stock).
- c. Insert 10 documents into the "products" collection.
- d. Display all the documents in the "products" collection.
- e. Display all the products in the "Electronics" category.
- f. Display all the products in ascending order of their names.
- g. Display the details of the first 5 products.
- h. Display the categories of products with a specific name.
- i. Display the number of products in the "Electronics" category.
- j. Display all the products without showing the "\_id" field.
- k. Display all the distinct categories of products.
- l. Display products in the "Electronics" category with prices greater than 50 but less than 100.
- m. Change the price of a product.
- n. Delete a particular product entry.

## **THEORY:**

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

- **Document-Oriented:** Stores data as flexible, JSON-like documents (BSON).
- **Flexible Schema:** No fixed structure, supports dynamic data.
- **Horizontal Scalability:** Uses sharding to manage large datasets.
- **Replication:** Ensures high availability with replica sets.
- **Indexing:** Supports various indexes for faster query execution.
- **Aggregation Framework:** Provides powerful data processing using pipelines.
- **Ad-hoc Queries:** Enables complex queries with ease.

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

**Documents:** JSON-like records storing data in key-value pairs. Example:

```
{  
  "_id": "101",  
  "name": "Alice", "age": 28,  
  "email": "alice@example.com"  
}
```

**Collections:** A group of documents, equivalent to tables in relational databases. They don't enforce strict schemas, allowing flexibility.

### 3. When to use MongoDB?

- **Big Data Applications:** Efficient for large, unstructured data.
- **E-commerce Platforms:** Ideal for product catalogs with dynamic attributes.
- **Content Management Systems (CMS):** Supports frequent changes in data models.
- **Real-Time Analytics:** Processes and analyzes data rapidly.
- **IoT and Mobile Apps:** Manages sensor data and app data effectively.

- **Social Networks:** Scales well for user-generated content.

#### 4. What is Sharding in MongoDB?

**Sharding:** Distributes data across multiple servers to handle large datasets.

**Shard Key:** A field in documents used to split data across shards.

#### Components:

- Shards: Store actual data.
- Config Servers: Maintain metadata and sharding configuration.
- Mongos: Routes queries to the appropriate shards.

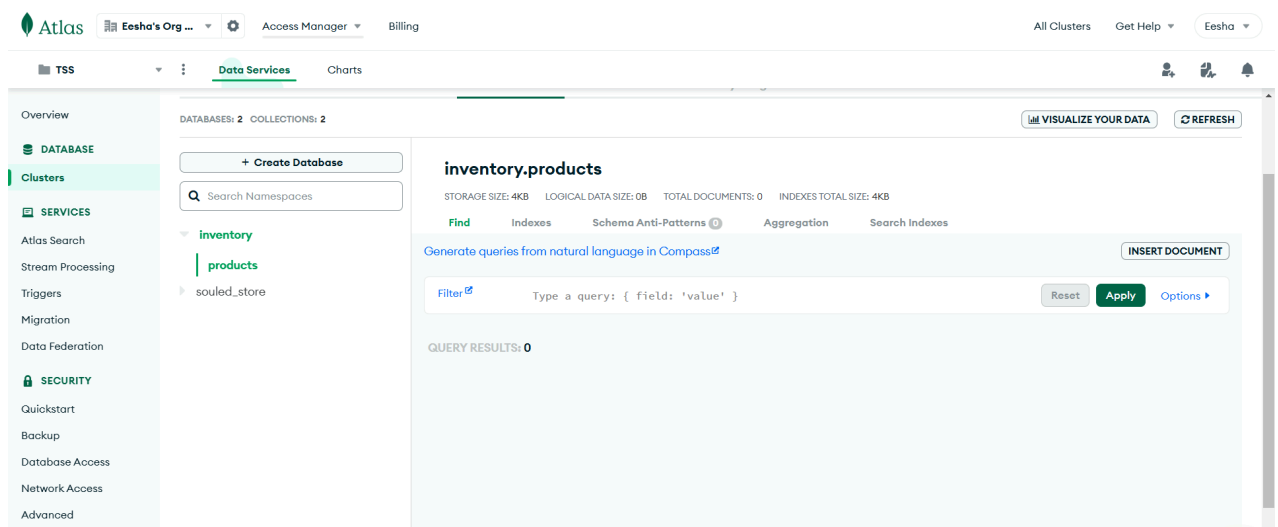
#### Benefits:

- Supports large-scale data management.
- Improves read and write performance.
- Ensures fault tolerance and high availability.

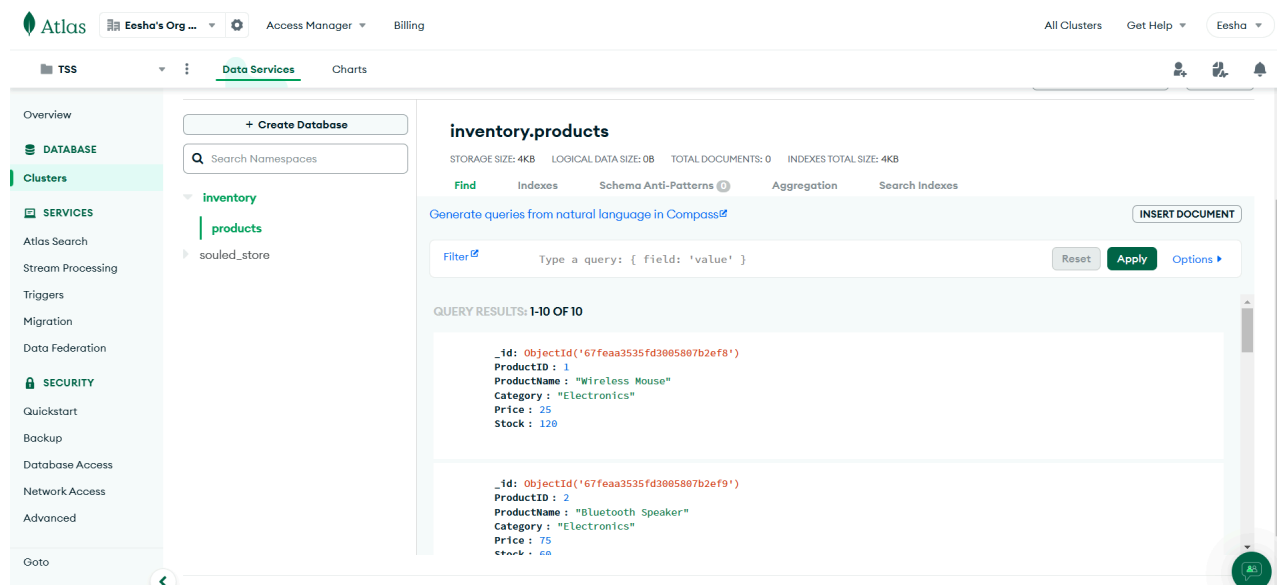
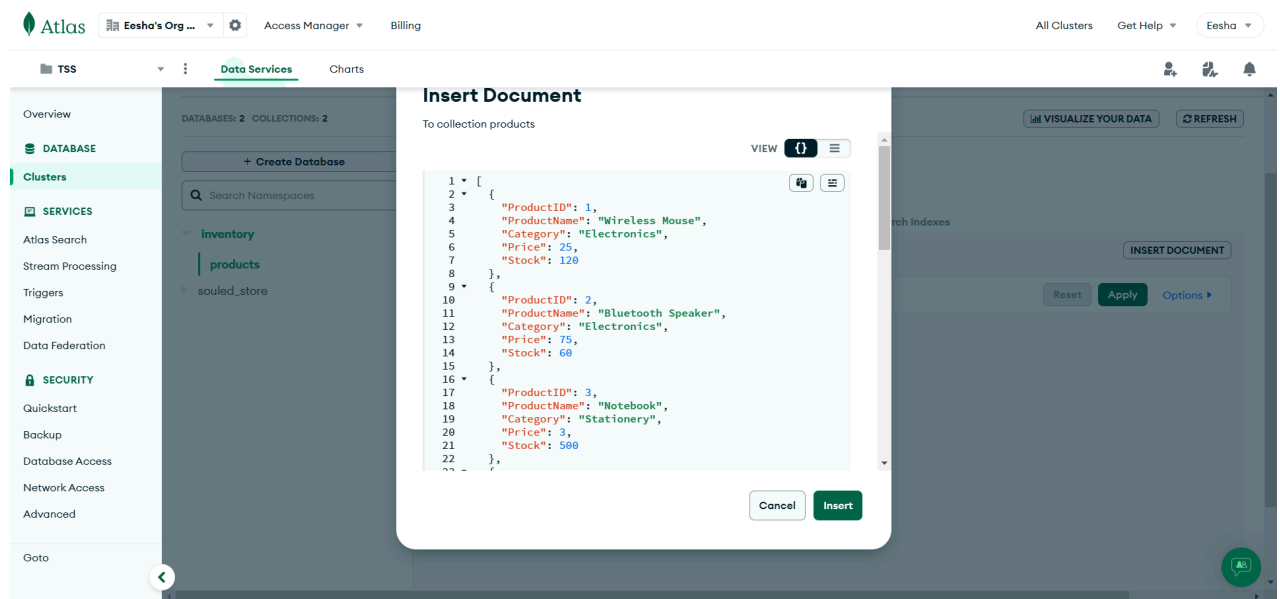
#### OUTPUT:

#### Insert Data (Create Operation)

1. Open your inventory collection.
2. Click "Insert Document" (top-right).



## Add data to the database -



## Read Data (Retrieve Documents)

1. Click on the inventory collection.
2. In the "FILTER" field, enter queries to retrieve data.

### a) Get all products:

- Query:

`{}`

The screenshot shows the MongoDB Compass interface. On the left, the 'inventory' namespace is expanded, showing the 'products' collection. The main panel displays the 'inventory.products' collection with a storage size of 4KB and 0 documents. The 'Find' tab is active, and the filter field is empty. The query results show two documents:

```
{ "_id": "ObjectId('67feaa3535fd3005807b2ef8')", "ProductID": 1, "ProductName": "Wireless Mouse", "Category": "Electronics", "Price": 25, "Stock": 120 }
```

```
{ "_id": "ObjectId('67feaa3535fd3005807b2ef9')", "ProductID": 2, "ProductName": "Bluetooth Speaker", "Category": "Electronics", "Price": 75, "Stock": 50 }
```

b) Get a specific product by ProductID:

- Query:

```
{ "ProductID": 1 }
```

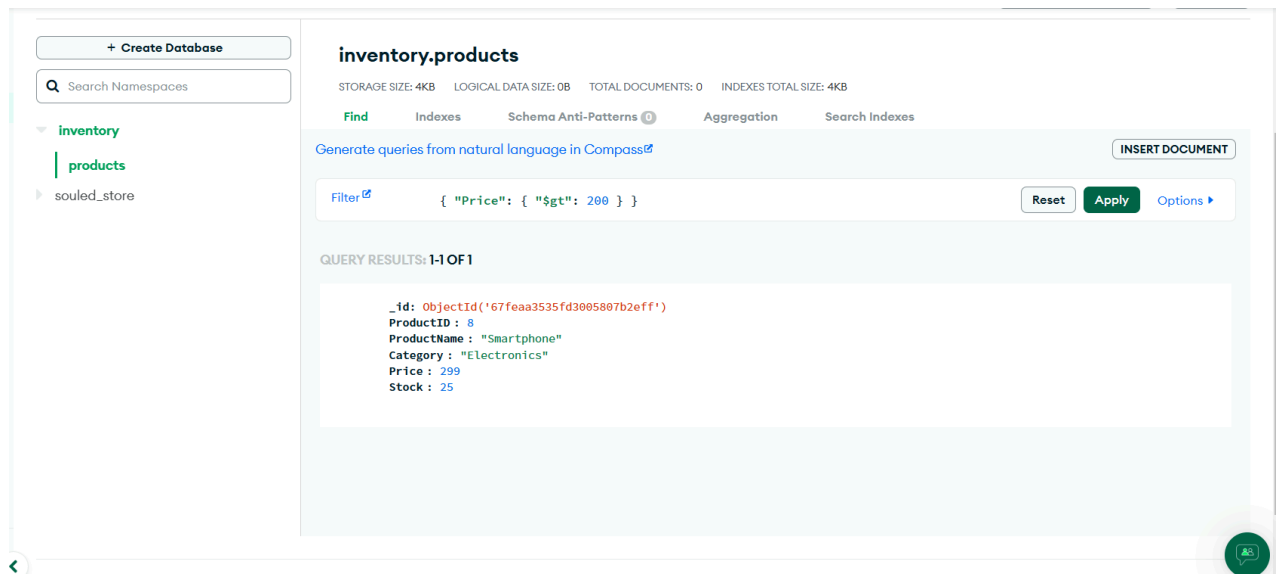
The screenshot shows the MongoDB Compass interface. The 'Find' tab is active, and the filter field contains the query: `{ "ProductID": 1 }`. The query results show one document:

```
{ "_id": "ObjectId('67feaa3535fd3005807b2ef8')", "ProductID": 1, "ProductName": "Wireless Mouse", "Category": "Electronics", "Price": 25, "Stock": 120 }
```

c) Get products with price greater than 200:

- Query:

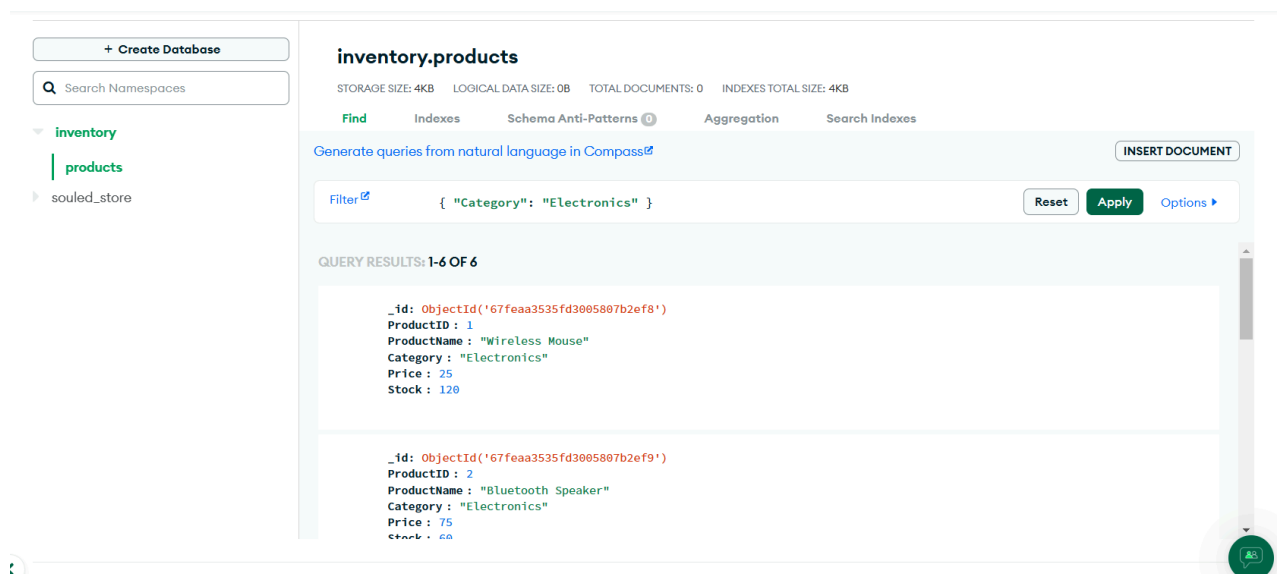
```
{ "Price": { "$gt": 200 } }
```



d) Get all products in the "Electronics" category:

- Query:

```
{ "Category": "Electronics" }
```



## Update Data

a) Update the price of a product:

Filter Query (to find the product):

```
{ "ProductID": 1 }
```

Update Query:

```
{ "$set": { "Price": 349 } }
```

- Click "Update".

DATABASES: 2 COLLECTIONS: 2

+ Create Database

Search Namespaces

inventory

products

souled\_store

inventory.products

STORAGE SIZE: 4KB LOGICAL DATA SIZE: 0B TOTAL DOCUMENTS: 0 INDEXES TOTAL SIZE: 4KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass

INSERT DOCUMENT

Filter { "ProductID": 1 }

Reset Apply Options

QUERY RESULTS: 1-1 OF 1

1	_id: ObjectId('67feaa3535fd3005807b2ef8')	ObjectId
2	ProductID: 1	Int32
3	ProductName: "Wireless Mouse"	String
4	Category: "Electronics"	String
5	Price: 25	Int32
6	Stock: 350	Int32

Document modified.

CANCEL UPDATE

- b) Add a new field "Discount" to all products:

Filter Query:

{ "Category": "Electronics" }

Update Query:

Cluster0 8.0.6 AWS Mumbai (ap-south-1)

Overview Real Time Metrics Collections Atlas Search Query Insights Performance Advisor Online Archive Cmd Line Tools Inf

DATABASES: 2 COLLECTIONS: 2

+ Create Database

Search Namespaces

inventory

products

souled\_store

inventory.products

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 131KB TOTAL DOCUMENTS: 10 INDEXES TOTAL SIZE: 20KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass

INSERT DOCUMENT

Filter { "Category": "Electronics" }

Reset Apply Options

QUERY RESULTS: 1-6 OF 6

1	_id: ObjectId('67feaa3535fd3005807b2ef8')	ObjectId
2	ProductID: 1	Int32
3	ProductName: "Wireless Mouse"	String
4	Category: "Electronics"	String
5	Price: 25	Int32
6	Stock: 350	Int32
7	Discount: "True"	String

## Delete Data

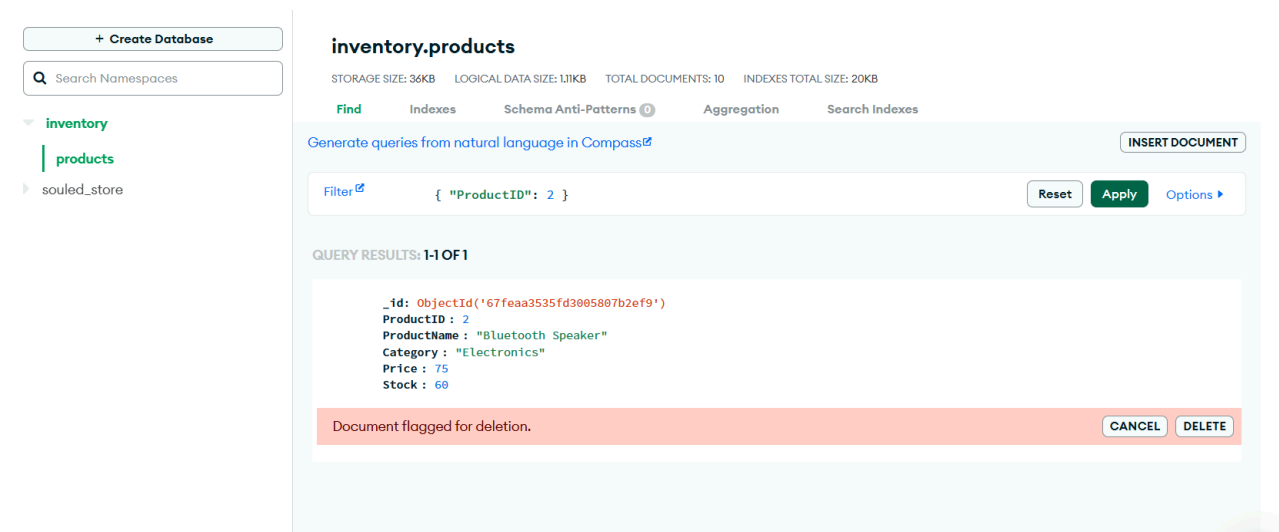
1. Click on the inventory collection.
2. Click "FILTER" and enter the query to find the document you want to delete.
3. Click "DELETE".

- a) Delete a specific product:

Filter Query:

```
{ "ProductID": 2 }
```

- Click "Delete One".

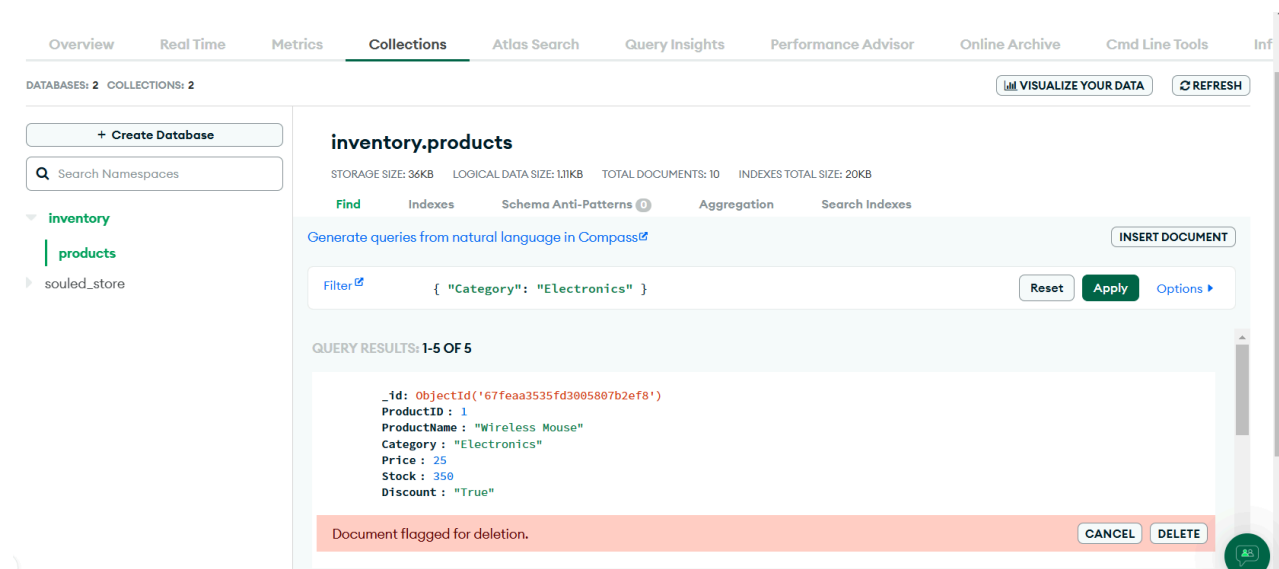


b) Delete all products in the "Electronics" category:

Filter Query:

```
{ "Category": "Electronics" }
```

- Click "Delete Many".



**CONCLUSION** :Through this experiment, we successfully performed CRUD operations in MongoDB, including creating a database, inserting documents, querying data, updating records, and deleting entries. We also explored filtering data, sorting, and aggregation queries.

MongoDB's document-oriented structure and flexible schema make it an ideal choice for handling large-scale, unstructured data in real-world applications.