CET-223

Web Technologies

Experiment # 24

Experiment Title

Performing CRUD Operations in MongoDB

Assessment of CLO(s): 03

Performed on _____

Student Name:			
Roll No.		**Group**	
Semester		**Session**	

Total (Max)	**Performance (03)**	**Viva (03)**	**File (04)**	**Total (10)**
Marks Obtained				
Remarks (if any)				

Experiment evaluated by

Instructor's Name	**Engr. Bilal Iqbal**		
Date		**Signature**	

OBJECTIVE:

To learn and implement the basic CRUD (Create, Read, Update, Delete) operations in MongoDB. These operations form the foundation of interacting with a MongoDB database and are essential for any application that stores and retrieves data.

CRUD Operations:

MongoDB uses collections instead of tables, and documents instead of rows. Each document is represented as a BSON (Binary JSON) object, and CRUD operations allow us to interact with the database.

1. Create Operation

In MongoDB, the `insertOne()` and `insertMany()` methods are used to create new documents in a collection.

Inserting a Single Document

```
// Switch to the database (create if it does not exist)
use FormData;

// Insert a single document into the 'users' collection
db.users.insertOne({
  name: "John Doe",
  age: 28,
  email: "johndoe@example.com",
  address: {
    street: "123 Elm St.",
    city: "Somewhere",
    zip: "12345"
  }
});
```

Inserting Multiple Documents

```
db.users.insertMany([
  {
    name: "Alice Smith",
    age: 30,
    email: "alice@example.com",
    address: {
      street: "456 Oak St.",
      city: "Anywhere",
      zip: "67890"
    }
  },
  {
    name: "Bob Johnson",
    age: 25,
    email: "bob@example.com",
    address: {
      street: "789 Pine St.",
      city: "Everywhere",
      zip: "54321"
    }
```

```
  }
]);
```

2. Read Operation

To retrieve documents, MongoDB provides the `find()` method. The `findOne()` method returns a single document.

Find All Documents:

```
// Find all users in the 'users' collection
db.users.find();
```

Find Documents with a Condition

```
// Find users where age is greater than 25
db.users.find({ age: { $gt: 25 } });
```

Find One Document

```
// Find a user by their email
db.users.findOne({ email: "johndoe@example.com" });
```

3. Update Operation

MongoDB uses `updateOne()`, `updateMany()` methods to update documents in the collection.

Update a Single Document

```
// Update the age of the user with name 'John Doe'
db.users.updateOne(
  { name: "John Doe" },
  { $set: { age: 29 } }
);
```

Update Multiple Documents

```
// Update the age of all users to 30 who are older than 25
db.users.updateMany(
  { age: { $gt: 25 } },
  { $set: { age: 30 } }
);
```

4. Delete Operation:

To remove documents, MongoDB provides `deleteOne()` and `deleteMany()` methods.

Delete a Single Document

```
// Delete the user with name 'Alice Smith'
db.users.deleteOne({ name: "Alice Smith" });
```

Delete Multiple Documents

```
// Delete all users whose age is less than 30
db.users.deleteMany({ age: { $lt: 30 } });
```

Lab Tasks:

Task 1: Create Documents

- Create a collection called books.
- Insert at least 5 documents into the books collection, each containing fields such as:
    - title, author, year_published, genre, price

Task 2: Read Documents

- Retrieve all documents from the `books` collection.
- Find books by a specific author
- Find books where the `price` is greater than 10.

Task 3: Update Documents

- Update the price of a book with a specific title to a new value.
- Update the genre of all books published before 1950 to "Classic".

Task 4: Delete Documents

- Delete a book by its title
- Delete all books that have a price greater than 10.