 **Start MongoDB Shell:** Open a terminal or command prompt and run the MongoDB shell by typing mongo. Make sure your MongoDB server is running.

 **Create a Database:** In the MongoDB shell, you can create a new database using the use command. For example, let's create a database named sampleDB:

use sampleDB

This command switches to the sampleDB database. If the database doesn't exist, MongoDB will create it.

 **Create a Collection:** Now, let's create a collection within the sampleDB database. A collection is similar to a table in relational databases:

db.createCollection("users")

This command creates a collection named users within the sampleDB database.

 **Insert Documents:** Now, let's insert some sample documents into the users collection:

db.users.insert([

{ name: "John Doe", age: 30, email: "john@example.com" },

{ name: "Jane Smith", age: 25, email: "jane@example.com" },

{ name: "Bob Johnson", age: 35, email: "bob@example.com" }

])

This command inserts three documents into the users collection.

 **Query Data:** You can query the data to retrieve information from the database:

db.users.find()

This command retrieves all documents from the users collection.

### Indexing:

Indexing in MongoDB is crucial for optimizing query performance. Indexes can be created on one or more fields of a collection to improve the speed of data retrieval. MongoDB automatically creates an index on the \_id field by default.

#### Creating an Index:

To create an index on a specific field, you can use the createIndex method. For example, let's create an index on the email field in the users collection:

db.users.createIndex({ email: 1 })

In this example, 1 specifies ascending order. You can use -1 for descending order.

#### Listing Indexes:

To view the existing indexes on a collection, you can use the getIndexes method:

db.users.getIndexes()

### Delete Operations:

MongoDB provides the deleteOne and deleteMany methods to remove documents from a collection.

#### Deleting a Single Document:

To delete a single document based on a specific condition, you can use the deleteOne method. For example, let's delete the document where the name is "John Doe":

db.users.deleteOne({ name: "John Doe" })

#### Deleting Multiple Documents:

To delete multiple documents based on a specific condition, you can use the deleteMany method. For instance, let's delete all documents where the age is greater than 30:

javascript

db.users.deleteMany({ age: { $gt: 30 } })

In this example, $gt is the greater than operator.