ASSIGNMENT-4

NAME: GUDE PRADEEP

College: Kallam Haranadhareddy Institute of Technology

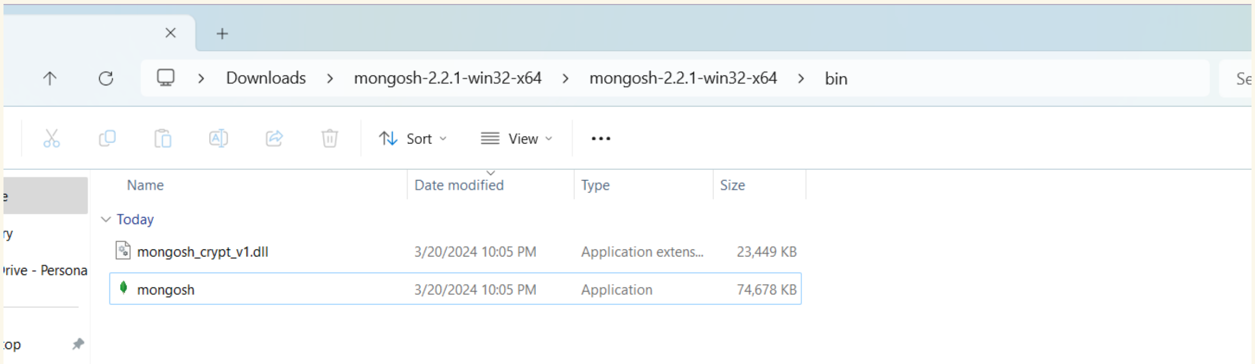
E mail: 208x1a0520@khitguntur.ac.in

All required commands and steps

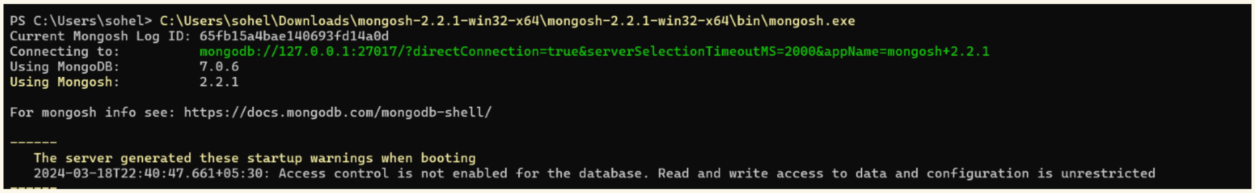
// Step 1: Connect to MongoDB server using Mongosh

Using mongosh shell

Mongosh.exe


Downloads > mongosh-2.2.1-win32-x64 > mongosh-2.2.1-win32-x64 > bin
Sort
View
Today

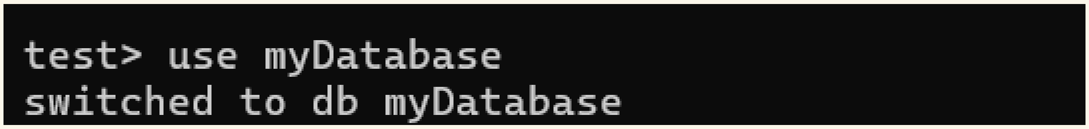
Name	Date modified	Type	Size
mongosh_crypt_v1.dll	3/20/2024 10:05 PM	Application extens...	23,449 KB
mongosh	3/20/2024 10:05 PM	Application	74,678 KB



```
PS C:\Users\sohel> C:\Users\sohel\Downloads\mongosh-2.2.1-win32-x64\mongosh-2.2.1-win32-x64\bin\mongosh.exe
Current Mongosh Log ID: 65fb15a4bae140693fd14a0d
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.2.1
Using MongoDB:          7.0.6
Using Mongosh:          2.2.1

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

------
   The server generated these startup warnings when booting
   2024-03-18T22:40:47.661+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------
```

// Step 2: Create a new database named myDatabase

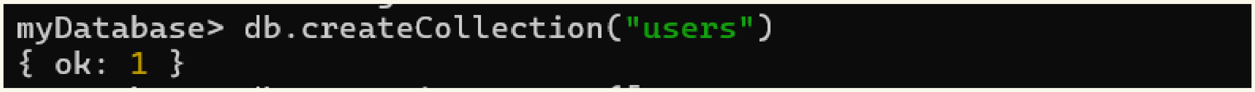
use myDatabase



```
test> use myDatabase
switched to db myDatabase
```

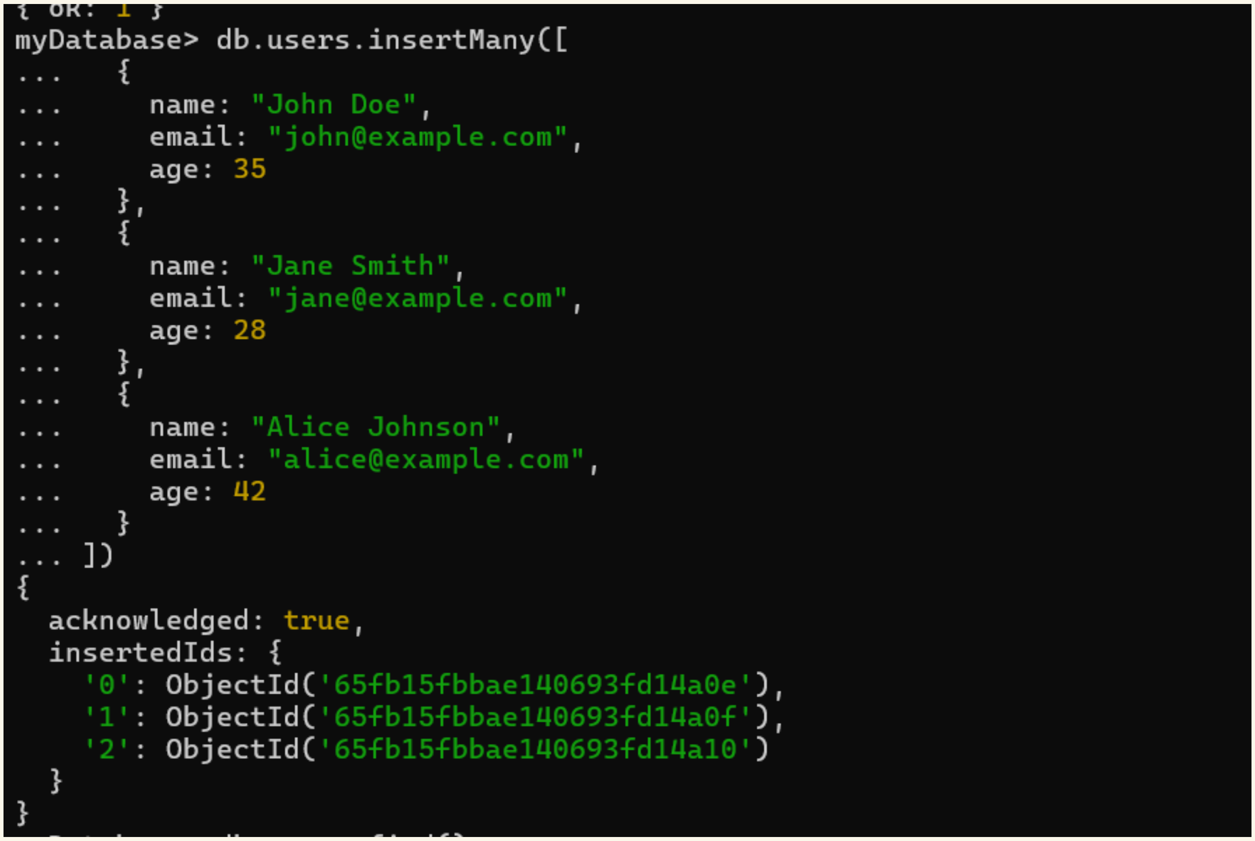
// Step 3: Create a collection named users within the myDatabase database

db.createCollection("users")



```
myDatabase> db.createCollection("users")
{ ok: 1 }
```

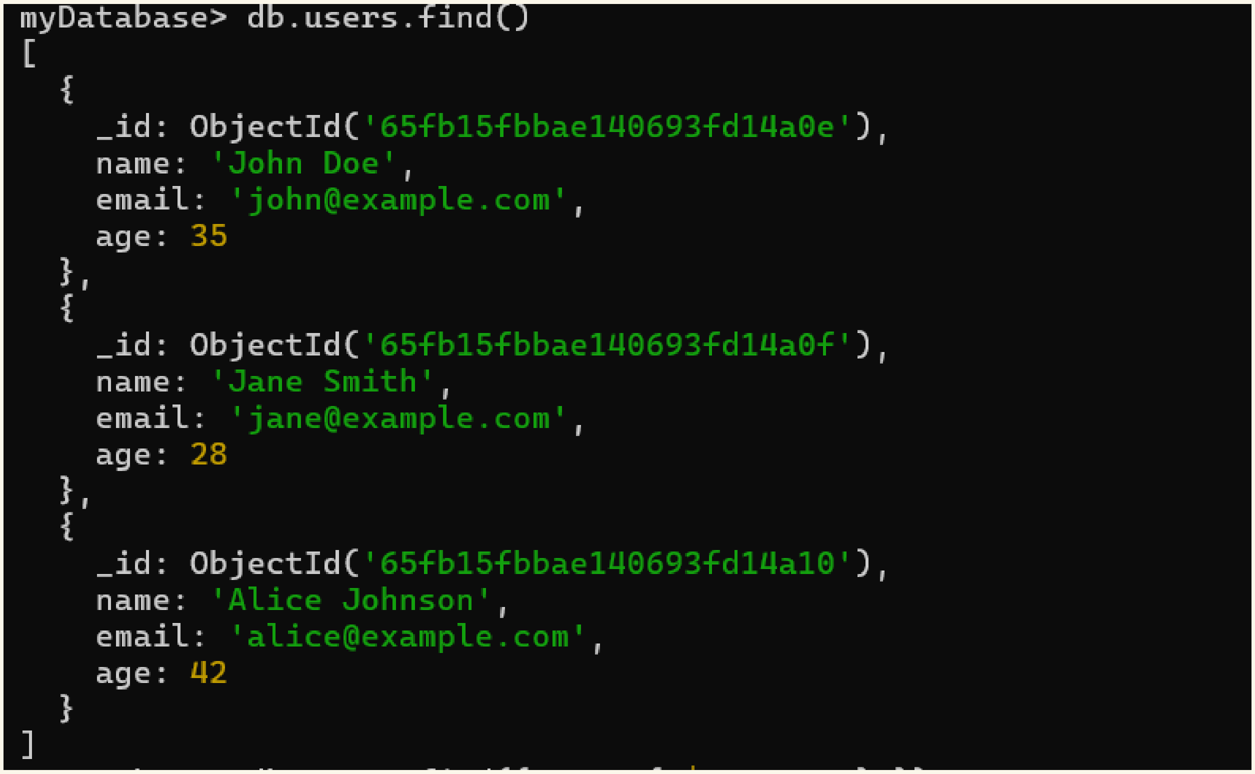
```
// Insert at least three documents into the users collection

db.users.insertMany([

  {

    name: "John Doe",

    email: "john@example.com",

    age: 35

  },

  {

    name: "Jane Smith",

    email: "jane@example.com",

    age: 28

  },

  {

    name: "Alice Johnson",

    email: "alice@example.com",

    age: 42

  }

])
```



```
{ ok: 1 }
myDatabase> db.users.insertMany([
...   {
...     name: "John Doe",
...     email: "john@example.com",
...     age: 35
...   },
...   {
...     name: "Jane Smith",
...     email: "jane@example.com",
...     age: 28
...   },
...   {
...     name: "Alice Johnson",
...     email: "alice@example.com",
...     age: 42
...   }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('65fb15fbbae140693fd14a0e'),
    '1': ObjectId('65fb15fbbae140693fd14a0f'),
    '2': ObjectId('65fb15fbbae140693fd14a10')
  }
}
```

// Retrieve all users from the users collection

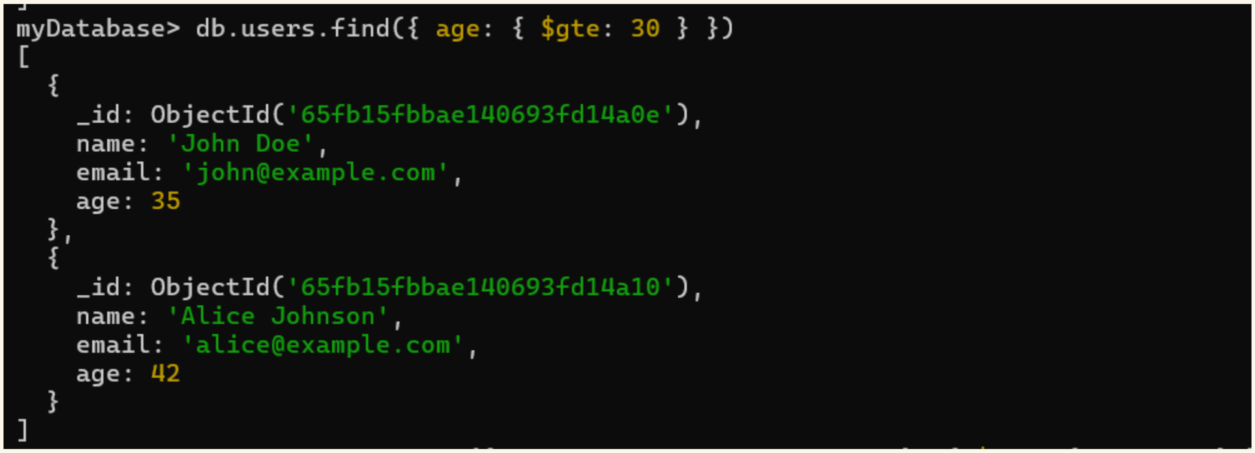
db.users.find()



```
myDatabase> db.users.find()
[
  {
    _id: ObjectId('65fb15fbbae140693fd14a0e'),
    name: 'John Doe',
    email: 'john@example.com',
    age: 35
  },
  {
    _id: ObjectId('65fb15fbbae140693fd14a0f'),
    name: 'Jane Smith',
    email: 'jane@example.com',
    age: 28
  },
  {
    _id: ObjectId('65fb15fbbae140693fd14a10'),
    name: 'Alice Johnson',
    email: 'alice@example.com',
    age: 42
  }
]
```

// Retrieve users with an age greater than or equal to 30

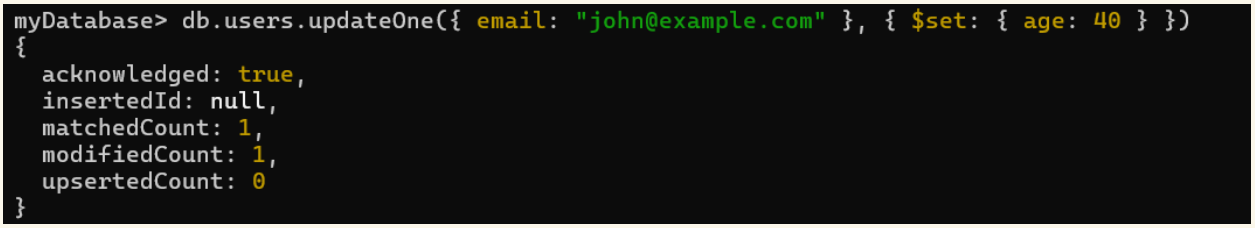
db.users.find({ age: { $gte: 30 } })



```
myDatabase> db.users.find({ age: { $gte: 30 } })
[
  {
    _id: ObjectId('65fb15fbbae140693fd14a0e'),
    name: 'John Doe',
    email: 'john@example.com',
    age: 35
  },
  {
    _id: ObjectId('65fb15fbbae140693fd14a10'),
    name: 'Alice Johnson',
    email: 'alice@example.com',
    age: 42
  }
]
```

// Update the age of a user with a specific email address

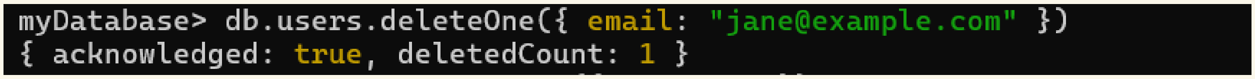
db.users.updateOne({ email: "john@example.com" }, { $set: { age: 40 } })



```
myDatabase> db.users.updateOne({ email: "john@example.com" }, { $set: { age: 40 } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

// Delete a user document based on a specific email address

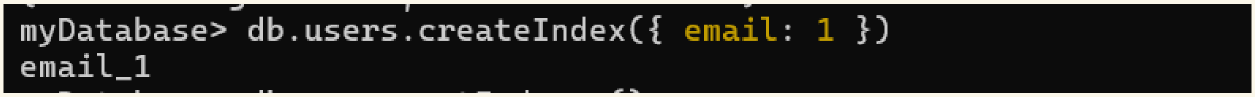
db.users.deleteOne({ email: "jane@example.com" })



```
myDatabase> db.users.deleteOne({ email: "jane@example.com" })
{ acknowledged: true, deletedCount: 1 }
```

// Create an index on the email field of the users collection

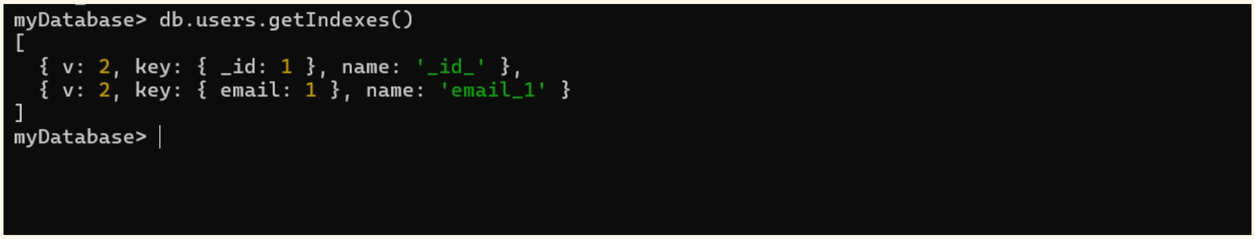
db.users.createIndex({ email: 1 })



```
myDatabase> db.users.createIndex({ email: 1 })
email_1
```

// Verify the index creation

db.users.getIndexes()



```
myDatabase> db.users.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { email: 1 }, name: 'email_1' }
]
myDatabase> |
```

// Exit Mongosh

Exit

```
]
myDatabase> exit
```

- **Switch to the database or create if not exists**: The **use** command is used to select a database. If the specified database (**myDatabase** in this case) doesn't exist, it will be created.
- **Create a collection named users**: This command creates a new collection named **users** within the selected database (**myDatabase**). Collections are analogous to tables in relational databases.
- **Insert at least three documents into the users collection**: The **insertMany()** function is used to insert multiple documents into the **users** collection. In this case, it inserts three documents, each representing a user with fields such as name, email, and age.
- **Retrieve all users from the users collection**: The **find()** function without any arguments is used to retrieve all documents from the **users** collection.
- **Retrieve users with an age greater than or equal to 30**: The **find()** function with a query object is used to retrieve documents from the **users** collection that match the specified criteria. In this case, it retrieves users with an age greater than or equal to 30.
- **Update the age of a user with a specific email address**: The **updateOne()** function is used to update a single document in the **users** collection that matches the specified filter (email address in this case). It sets the **age** field to 40 for the user with the email address "john@example.com".
- **Delete a user document based on a specific email address**: The **deleteOne()** function is used to delete a single document from the **users** collection that matches the specified filter (email address in this case). It deletes the document with the email address "jane@example.com".
- **Create an index on the email field of the users collection**: This command creates an index on the **email** field of the **users** collection. Indexes are used to improve query performance.
- **Verify the index creation**: The **getIndexes()** function is used to retrieve information about the indexes on the **users** collection. It verifies that the index on the **email** field was created successfully.
- **Exit Mongosh**: This command exits the Mongosh shell.