

## Assignment-4: Creating a Database Using MongoDB and Mongosh

**Name:** Sanampudi Siva Parvathi

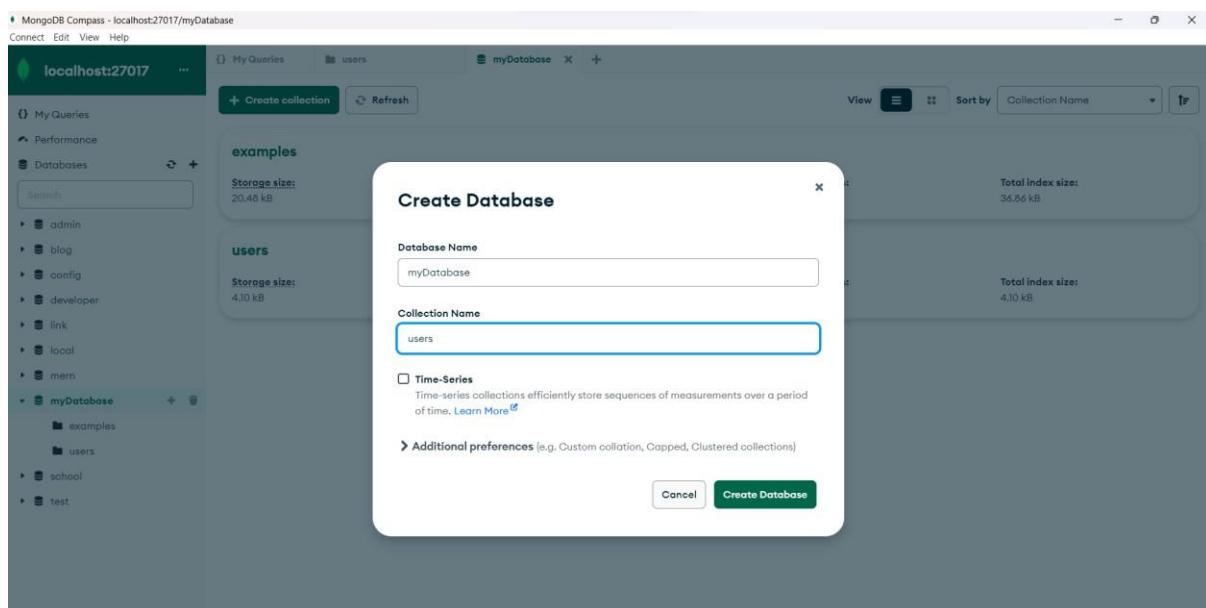
**Roll number:** 2022MCA16078

**Assignment-4:** Creating a Database Using MongoDB and Mongosh

**College name:** Sri Padmavati Mahila Visvavidyalayam

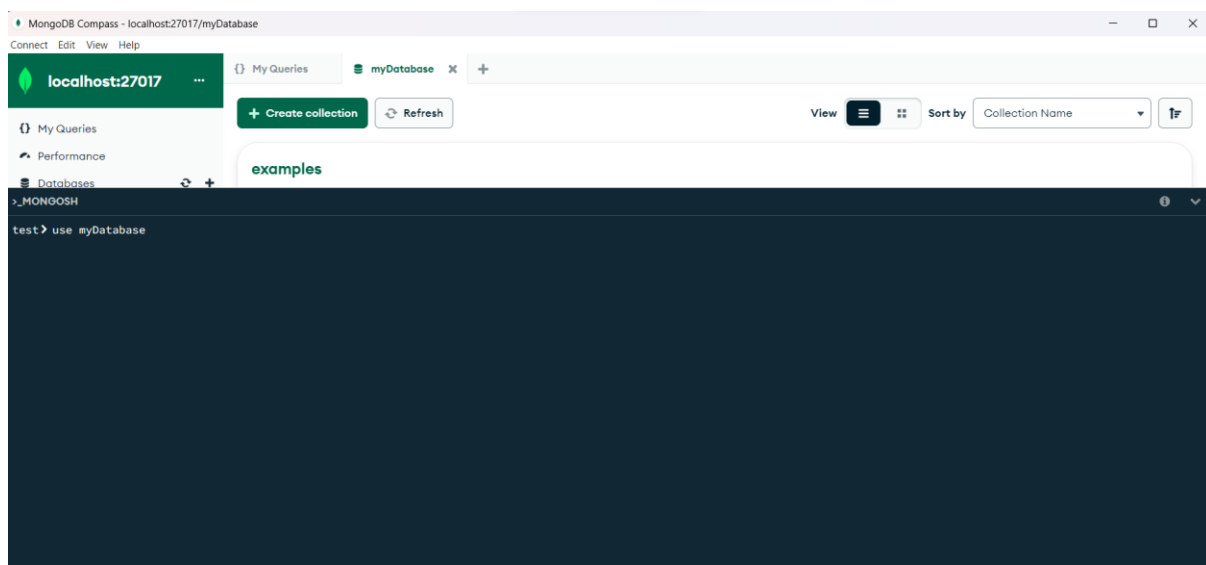
**Database Setup:** Create a new MongoDB database called 'myDatabase'.

**Collection Creation:** Create a collection named users within the myDatabase database.



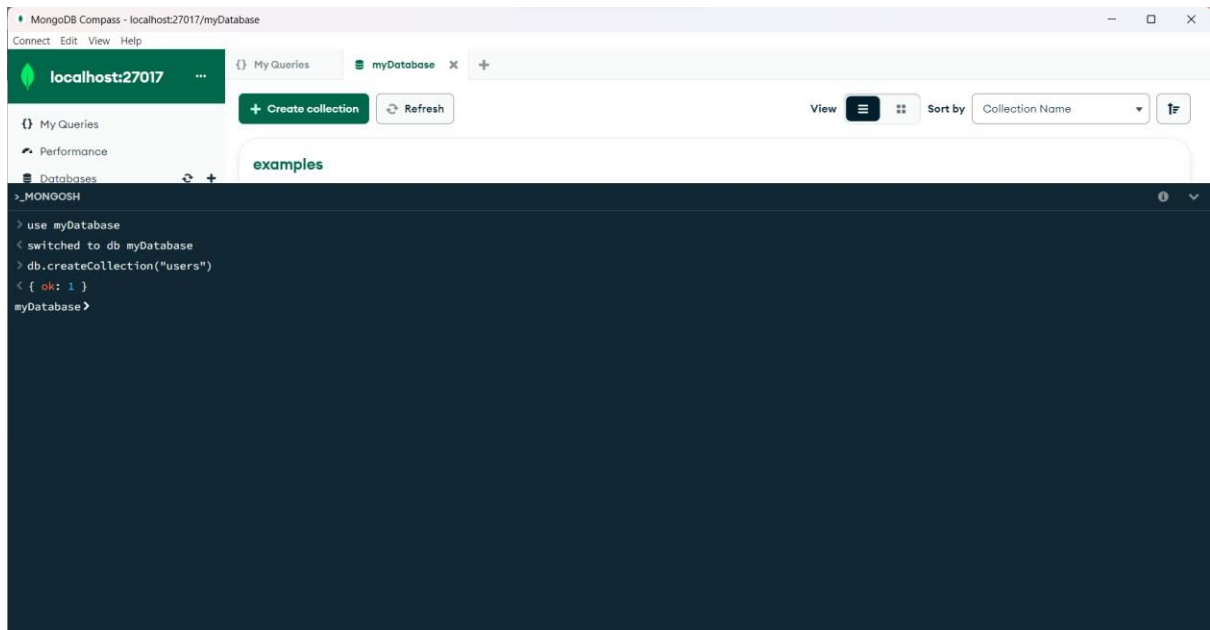
Below are the steps to achieve the tasks using MongoDB commands:**Database setup in MONGOSH :**

use myDatabase

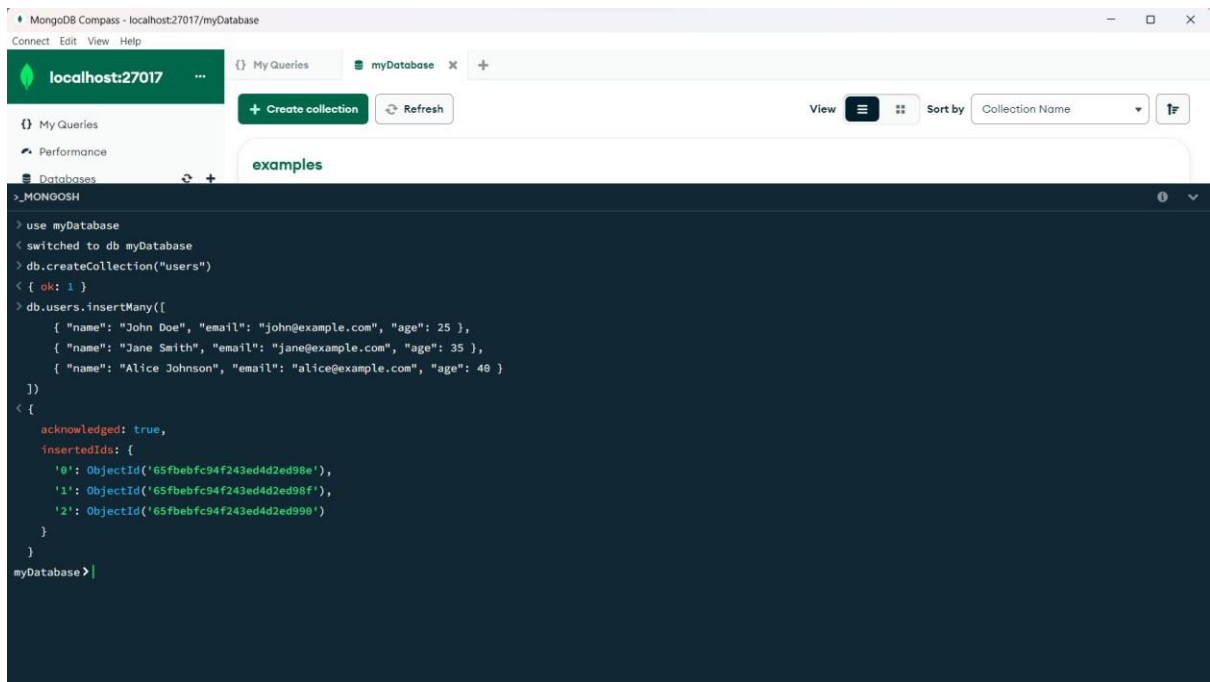


## Collection Creation in MONGOSH:

db.createCollection("users")

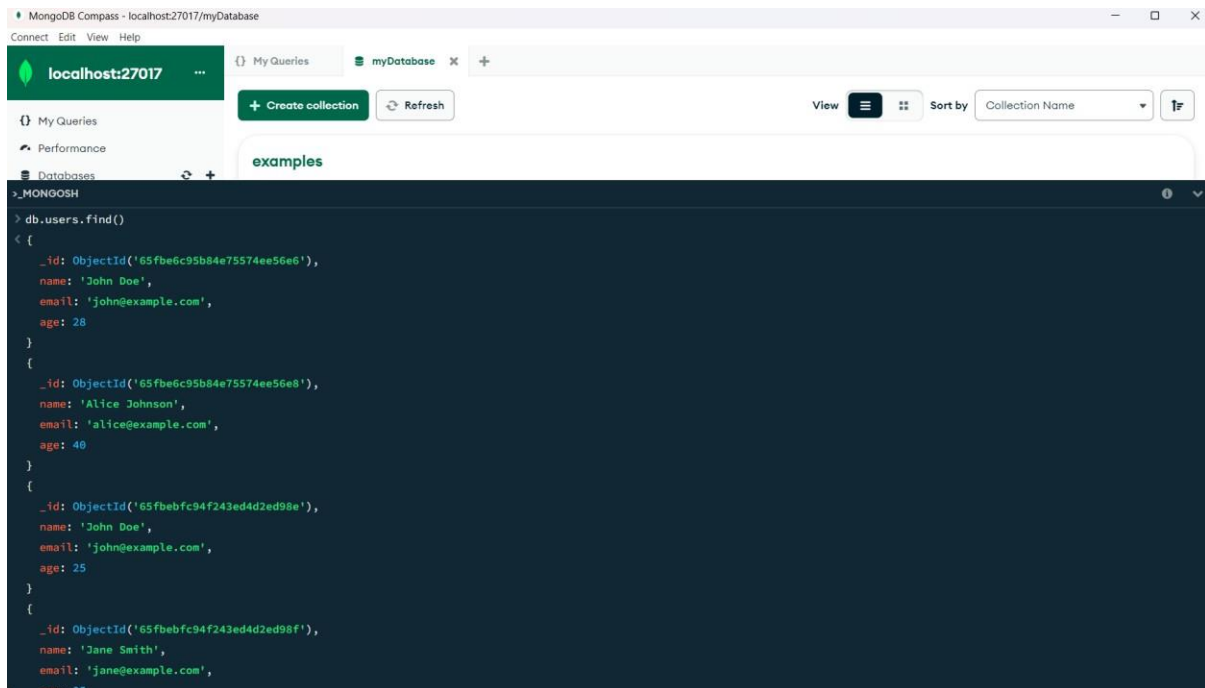


**Document Insertion:** Insert at least three documents into the users collection, each representing a user with fields such as name, email, and age.



**Querying:** Write queries to retrieve: All users from the users collection. Users with an age greater than or equal to 30.

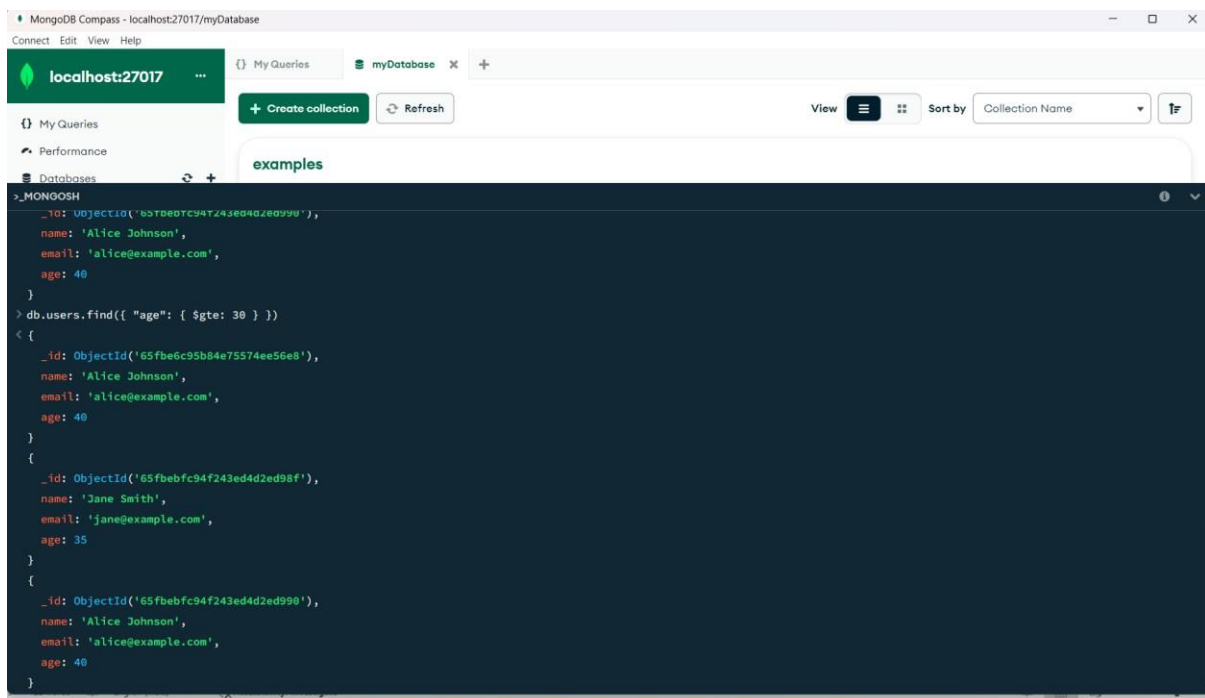
### All users from the users collection:



The screenshot shows the MongoDB Compass interface. The left sidebar displays the database structure with 'localhost:27017' and 'myDatabase'. The main panel shows a query result for the 'users' collection. The query is `db.users.find()`. The result is a JSON array of four user documents.

```
>_MONGOOSH
> db.users.find()
< [
  {
    _id: ObjectId('65fbc95b84e75574ee56e6'),
    name: 'John Doe',
    email: 'john@example.com',
    age: 28
  },
  {
    _id: ObjectId('65fbc95b84e75574ee56e8'),
    name: 'Alice Johnson',
    email: 'alice@example.com',
    age: 40
  },
  {
    _id: ObjectId('65fbc95b84e75574ee56e9'),
    name: 'John Doe',
    email: 'john@example.com',
    age: 25
  },
  {
    _id: ObjectId('65fbc95b84e75574ee56ea'),
    name: 'Jane Smith',
    email: 'jane@example.com',
    age: 35
  }
]
```

### Users with an age greater than or equal to 30:

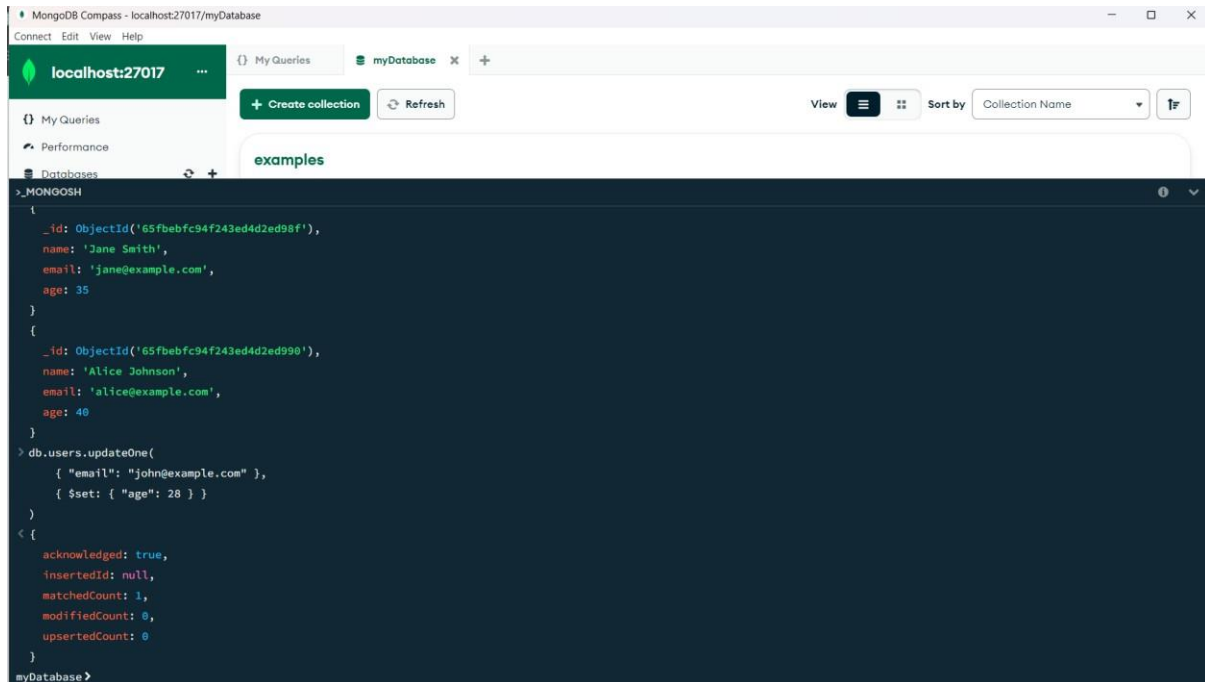


The screenshot shows the MongoDB Compass interface. The left sidebar displays the database structure with 'localhost:27017' and 'myDatabase'. The main panel shows a query result for the 'users' collection. The query is `db.users.find({ "age": { $gte: 30 } })`. The result is a JSON array of two user documents.

```
>_MONGOOSH
> db.users.find({ "age": { $gte: 30 } })
< [
  {
    _id: ObjectId('65fbc95b84e75574ee56e8'),
    name: 'Alice Johnson',
    email: 'alice@example.com',
    age: 40
  },
  {
    _id: ObjectId('65fbc95b84e75574ee56ea'),
    name: 'Jane Smith',
    email: 'jane@example.com',
    age: 35
  }
]
```

**Update Operation:** Update the age of a user with a specific email address.

**Update the age of a user with a specific email address (e.g., "john@example.com"):**



The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017/myDatabase'. The left sidebar shows the 'My Queries' tab. The main area displays a query result for the 'examples' collection. The result shows two documents: one for 'Jane Smith' with age 35, and one for 'Alice Johnson' with age 40. Below the query result, the MongoDB shell command is shown: `db.users.updateOne({ "email": "john@example.com" }, { $set: { "age": 28 } })`. The output of the command is shown as a JSON object: `{ acknowledged: true, insertedId: null, matchedCount: 1, modifiedCount: 0, upsertedCount: 0 }`.

```
>_MONQOSH
{
  "_id": ObjectId('65fbebfc94f243ed4d2ed98f'),
  "name": 'Jane Smith',
  "email": 'jane@example.com',
  "age": 35
}
{
  "_id": ObjectId('65fbebfc94f243ed4d2ed990'),
  "name": 'Alice Johnson',
  "email": 'alice@example.com',
  "age": 40
}
> db.users.updateOne(
  { "email": "john@example.com" },
  { $set: { "age": 28 } }
)
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
myDatabase>
```

**Deletion Operation:** Delete a user document based on a specific email address.

**Delete a user document based on a specific email address (e.g., "jane@example.com"):**



The screenshot shows the MongoDB shell command: `db.users.deleteOne({ "email": "jane@example.com" })`. The output is shown as a JSON object: `{ acknowledged: true, deletedCount: 1 }`.

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

**Index Creation:** Create an index on the email field of the users collection.

**Create an index on the email field of the users collection:**



The screenshot shows the MongoDB shell command: `db.users.createIndex({ "email": 1 })`. The output is shown as a JSON object: `{ email_1 }`.

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