

1. **Database Setup:** Create a new MongoDB database called myDatabase
2. **Collection Creation:** Create a collection named users within the myDatabase database.

Using mongosh:

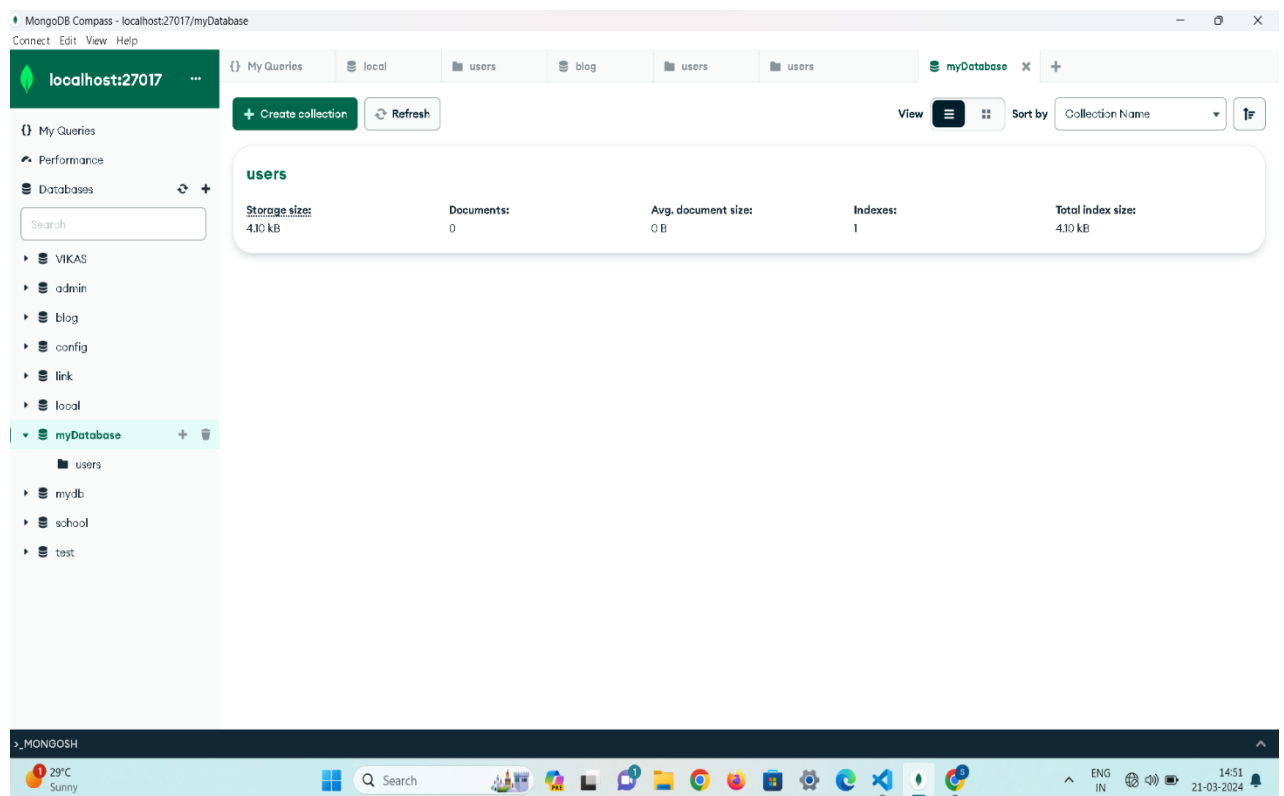
use myDatabase

switched to db myDatabase

```
db.createCollection("users")
```

```
{ ok: 1 }
```

Output: -



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

```
db.users.insertMany([{"name":"Sattar","email":"sattarpathan@gmail.com","age":24},{"name":"Madhu","email":"madhubairabhoina@gmail.com","age":21},{"name":"Subhani","email":"subhanishaik@gmail.com","age":26},{"name":"Samba","email":"sambaramishetty@gmail.com","age":32},{"name":"Zakeer","email":"zakeerbilla@gmail.com","age":35}])
```

acknowledged: true,

insertedIds: {

'0': ObjectId('65fbfdc05e817de316f6a8ec'),

'1': ObjectId('65fbfdc05e817de316f6a8ed'),

'2': ObjectId('65fbfdc05e817de316f6a8ee'),

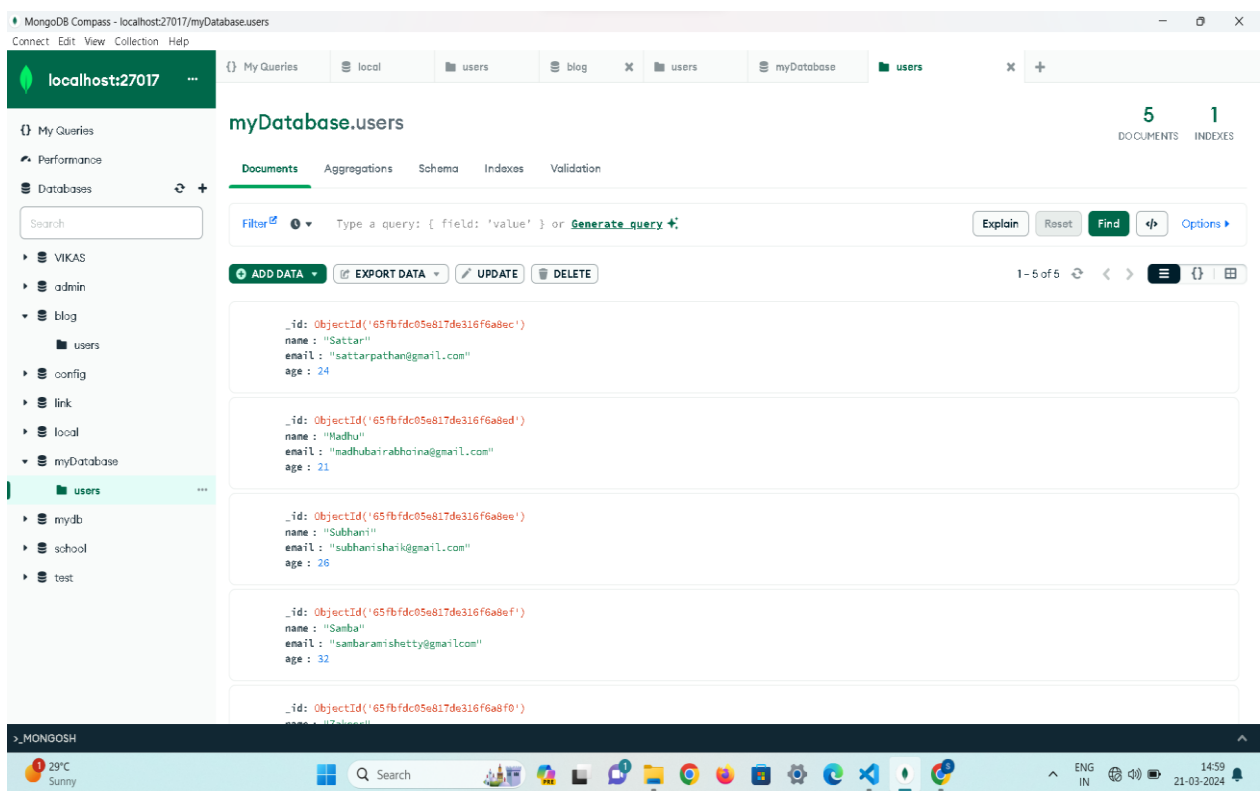
'3': ObjectId('65fbfdc05e817de316f6a8ef'),

'4': ObjectId('65fbfdc05e817de316f6a8f0')

}

}

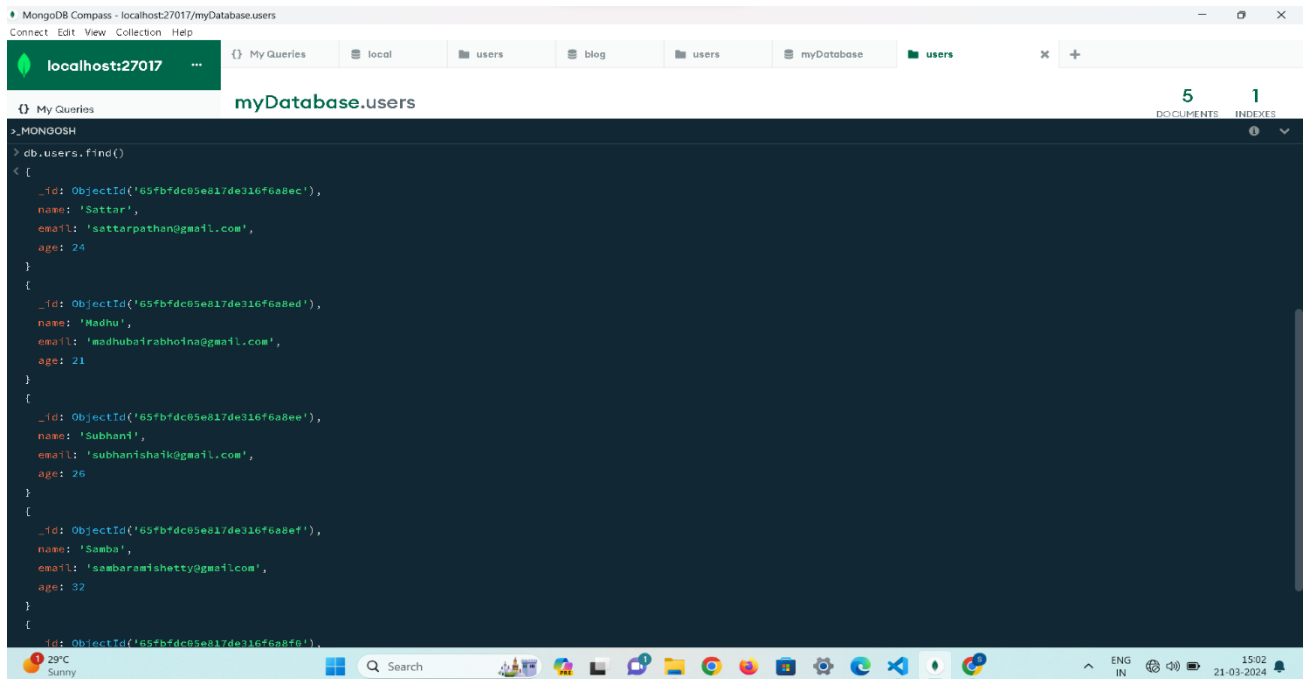
Output: -



4. Querying: Write queries to retrieve:

1. All users from the users collection

```
db.myDatabase.find()
```



2. Users with an age greater than or equal to 30.

```
>_MONGOSH
> db.users.find({age:{$gte:30}})
< [
  {
    _id: ObjectId('65fbfdc05e817de316f6a8ef'),
    name: 'Samba',
    email: 'sambaramishetty@gmailcom',
    age: 32
  },
  {
    _id: ObjectId('65fbfdc05e817de316f6a8f0'),
    name: 'Zakeer',
    email: 'zakeerbilla@gmail.com',
    age: 35
  }
]
myDatabase> |
```

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

```
>_MONGOSH
>
> db.users.updateOne({"email":"sattarpathan@gmail.com"},{$set:{age:23}})
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
> db.users.find()
< [
  {
    _id: ObjectId('65fbfdc05e817de316f6a8ec'),
    name: 'Sattar',
    email: 'sattarpathan@gmail.com',
    age: 23
  },
  {
    _id: ObjectId('65fbfdc05e817de316f6a8ed'),
    name: 'Madhu',
    email: 'madhubairabhoina@gmail.com',
    age: 21
  }
]
```

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

```
>_MONGOSH
>
> db.users.deleteOne({"email":"madhubairabhoina@gmail.com"})
< {
  acknowledged: true,
  deletedCount: 1
}
> db.users.find()
< [
  {
    _id: ObjectId('65fbfdc05e817de316f6a8ec'),
    name: 'Sattar',
    email: 'sattarpathan@gmail.com',
    age: 23
  },
  {
    _id: ObjectId('65fbfdc05e817de316f6a8ee'),
    name: 'Subhani',
    email: 'subhanishaik@gmail.com',
    age: 26
  },
  {
    _id: ObjectId('65fbfdc05e817de316f6a8ef'),
    name: 'Samba',
    email: 'sambaramishetty@gmailcom',
    age: 32
  },
  {
    _id: ObjectId('65fbfdc05e817de316f6a8f0'),
    name: 'Zakeer'
  }
]
```

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

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

```
email_1
```

```
db.users.getIndexes()
```

```
[  
  { v: 2, key: { _id: 1 }, name: '_id_' },  
  { v: 2, key: { email: 1 }, name: 'email_1' }  
]
```

The screenshot shows the MongoDB Compass interface. On the left, the 'Databases' sidebar lists 'myDatabase' with its collections: 'users', 'config', 'link', 'local', 'mydb', 'school', and 'test'. The 'users' collection is selected. The main panel shows the 'Indexes' tab for 'myDatabase.users'. It displays two indexes: '_id_' (REGULAR, 36.9 KB, 1 document since Thu Mar 21 2024) and 'email_1' (REGULAR, 20.5 KB, 0 documents since Thu Mar 21 2024). The 'email_1' index is highlighted, indicating it is the current selection.

Name and Definition	Type	Size	Usage	Properties
> _id_	REGULAR ⓘ	36.9 KB	1 (since Thu Mar 21 2024)	UNIQUE ⓘ
> email_1	REGULAR ⓘ	20.5 KB	0 (since Thu Mar 21 2024)	