|  |
| --- |
|  |

**LAB 4: Data Modeling Techniques in MongoDB**

|  |  |
| --- | --- |
| **Track** | Data & AI |
| **Role** | Software Development - Backend |
| **Level** | Level 2 |
| **Course** | NoSQL Databases |
| **Scope** | * Setting Up MongoDB Atlas Cloud Cluster * Design and implement various relationship types effectively. * Configure Python to manage files using GridFS. * Upload, retrieve, and verify files in MongoDB Atlas. * Implementing CRUD Operations in MongoDB Atlas |
| **Prerequisite** | * Basic understanding of MongoDB commands and relationships. * A working MongoDB Atlas account and cluster configured for access. * Familiarity with database CRUD operations. |

**Objective**

To configure and utilize MongoDB Atlas for managing database relationships, perform CRUD operations, and leverage GridFS for file storage and retrieval.

**Tasks**

**Dataset (also available in resources): telecom\_dataset\_v2.json**

1. **Set Up MongoDB Atlas Cloud Cluster**
   * Go to MongoDB Atlas
   * Sign up for a free-tier account
   * Configure Network Access
   * Create a Database User
   * Get Your Connection String
2. **Designing and Implementing One-to-One Relationships**
   * Create the Customers Collection
   * Create the BillingDetails Collection
3. **Implementing One-to-Many Relationships**
   * Create a collection Subscriptions
4. **Implementing Many-to-Many Relationships**
   * Create an intermediary collection
5. **Python Setup for GridFS**
   * Create Python Script
   * Establish Connection to MongoDB Atlas
6. **Upload and Retrieve Files Using GridFS**
   * Upload a Large File to GridFS
   * Retrieve the File from GridFS
7. **Verify GridFS Data in MongoDB Atlas**
   * Check MongoDB Atlas Collections
   * Query File Metadata
8. **Implement CRUD Operation in MongoDB Atlas Database**
   * Connect to MongoDB Atlas
   * Switch to the database
   * CRUD Operations in MongoDB Shell

**Outcomes**

* Set up MongoDB Atlas and connected successfully.
* Designed and implemented flexible data relationships.
* Configured Python with MongoDB and GridFS support.
* Uploaded, retrieved, and verified files via GridFS.
* Performed CRUD operations to manage MongoDB data.

**Solution**

1. **Set Up MongoDB Atlas Cloud Cluster:**

* Go to MongoDB Atlas:
  + Go to MongoDB Atlas.A computer screen shot of a computer keyboard

    Description automatically generated
  + Sign up for a free-tier account.

A screenshot of a computer

Description automatically generated

* Sign up for a free-tier account.
  + A screenshot of a computer

    Description automatically generatedLog in to MongoDB Atlas, then create a new cluster in a region of your choice (choose the M0 free-tier for this lab).

A screenshot of a computer

Description automatically generated

* + MongoDB Atlas will automatically create the necessary resources for your cluster.
* Configure Network Access:
  + Under Security in the left sidebar, go to Network Access and click Add IP Address.
  + Add your current IP address to the whitelist.

A screenshot of a computer

Description automatically generated

* Create a Database User:
  + Under Security, click on Database Access, and then Add New Database User.

A screenshot of a computer

Description automatically generated

* + Create a user with readWrite permissions for your cluster.

A screenshot of a computer

Description automatically generated

* Get Your Connection String:
  + Click Connect on the cluster's dashboard, then Connect your application.

A screenshot of a computer

Description automatically generated

* + Copy the connection string that looks like:

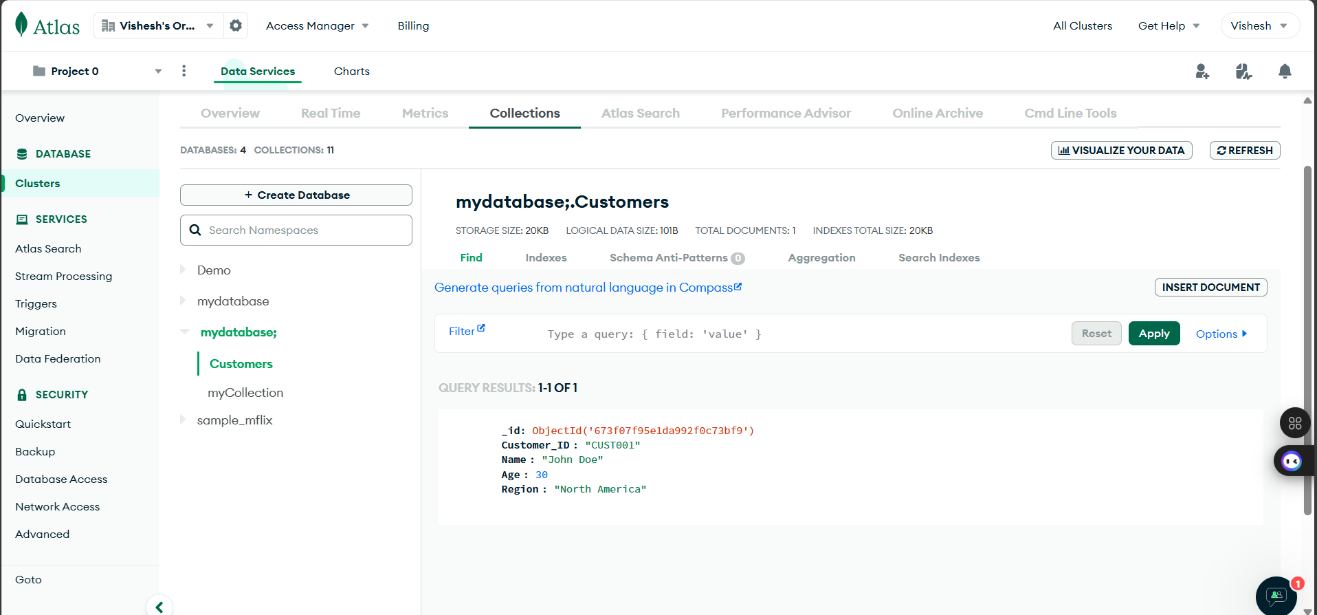
mongodb+srv://Vishesh6609:<db\_password>@cluster0.9fsal.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0

1. **Designing and Implementing One-to-One Relationships:**

* Create the Customers Collection
  + Insert customer information into the Customers collection.



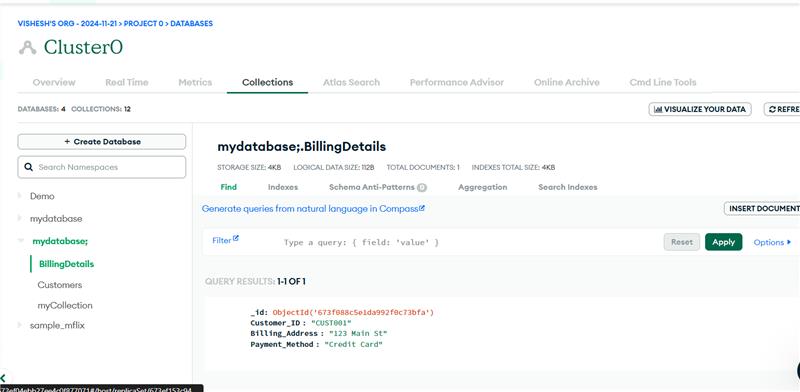
* + Output:



* Create the BillingDetails Collection:

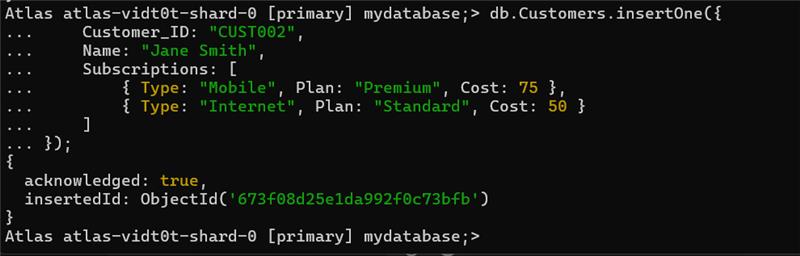


* + Output:

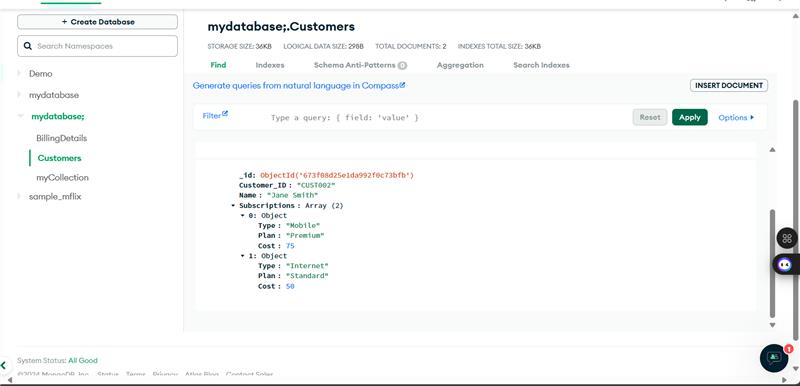


1. **Implementing** **One-to-Many Relationships:**

* Creating One-to-Many Relationships:
  + Create a collection Subscriptions where a customer can have multiple subscriptions, such as Mobile, Internet, and TV.



* + Output:

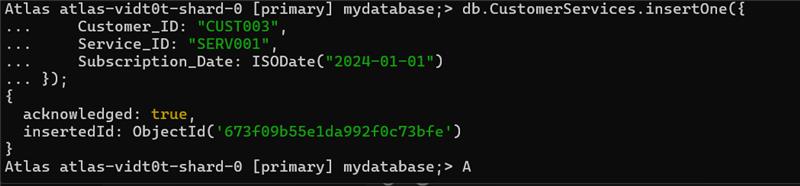


1. **Implementing Many-to-Many Relationships:**

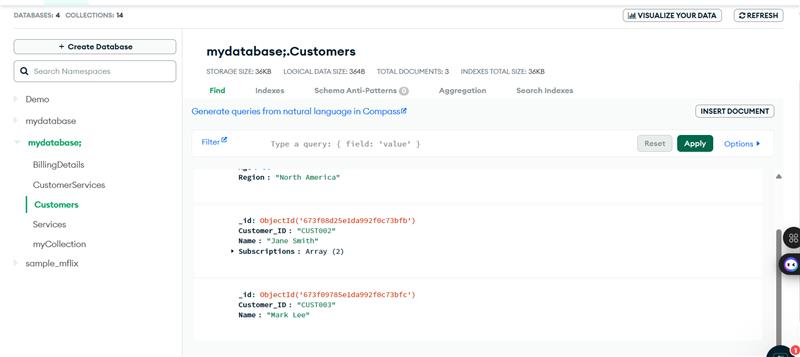
* Create an intermediary collection:
  + Create an intermediary collection to model customers subscribed to multiple services.

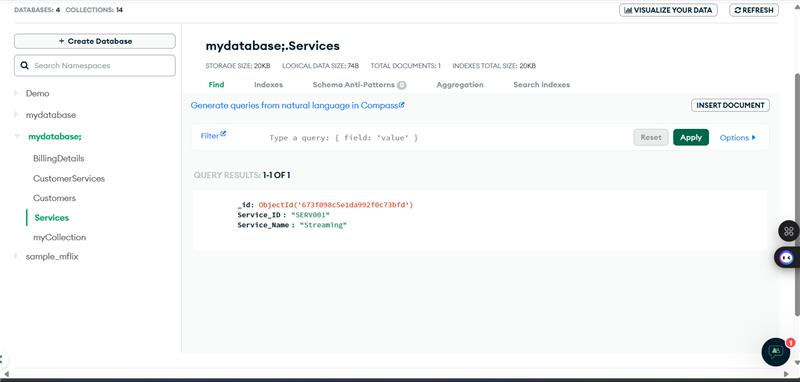


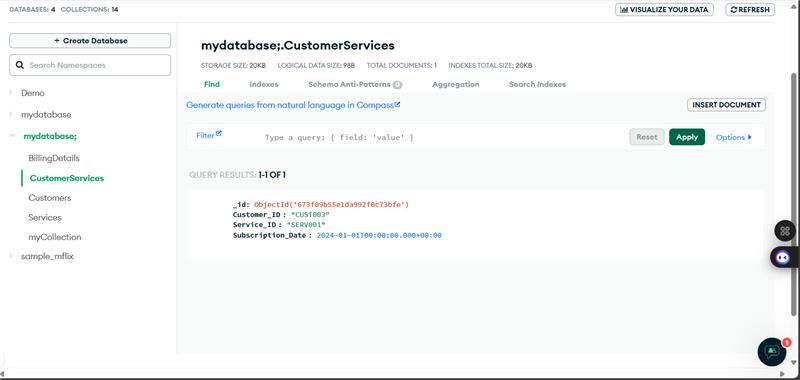




* + Output:







1. **Python Setup for GridFS:**

* Use Create Python Script:
  + Open a text editor and create a new file called gridfs\_example.py.
* Establish Connection to MongoDB Atlas:

import pymongo

import gridfs

# Replace <username>, <password>, and <dbname> with your Atlas credentials

client = pymongo.MongoClient("mongodb+srv://Vishesh6609:Vishesh6609@cluster0.9fsal.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0")

# Connect to the database and initialize GridFS

db = client["mydatabase"]  # Replace with your database name

fs = gridfs.GridFS(db)

1. **Upload and Retrieve Files Using GridFS:**

* Upload a Large File to GridFS:
  + You will upload a sample file (e.g., large\_file.mp4) to your MongoDB Atlas cluster using GridFS.

# Open the file and upload to GridFS

with open("large\_file.mp4", "rb") as file:

    file\_id = fs.put(file, filename="large\_file.mp4")

    print(f"File stored with ID: {file\_id}")

* Retrieve the File from GridFS:
  + Now, retrieve the uploaded file from GridFS and save it locally.

# Retrieve the file from GridFS

file\_data = fs.get(file\_id)

with open("downloaded\_large\_file.mp4", "wb") as file:

    file.write(file\_data.read())

1. **Verify GridFS Data in MongoDB Atlas:**

* Check MongoDB Atlas Collections:
  + In the Atlas dashboard, navigate to Collections.
  + You should see a fs.files collection that contains metadata about your uploaded file, such as filename, file size, and upload date.
  + The fs.chunks collection will store the chunks of your uploaded file.

A screenshot of a browser window

Description automatically generated

* Query File Metadata:
  + You can query for file metadata (e.g., filename) from the fs.files collection directly in the Atlas UI or via MongoDB shell:

A screenshot of a computer

Description automatically generated

1. **Implement CRUD Operation in MongoDB Atlas Database:**

* Connect to MongoDB Atlas:

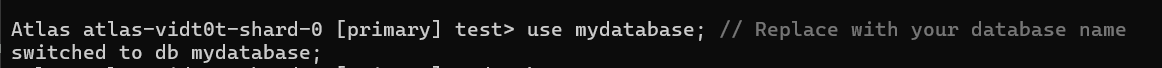
mongosh "mongodb+srv://<username>:<password>@cluster0.mongodb.net/<database>?retryWrites=true&w=majority"

**A screenshot of a computer program

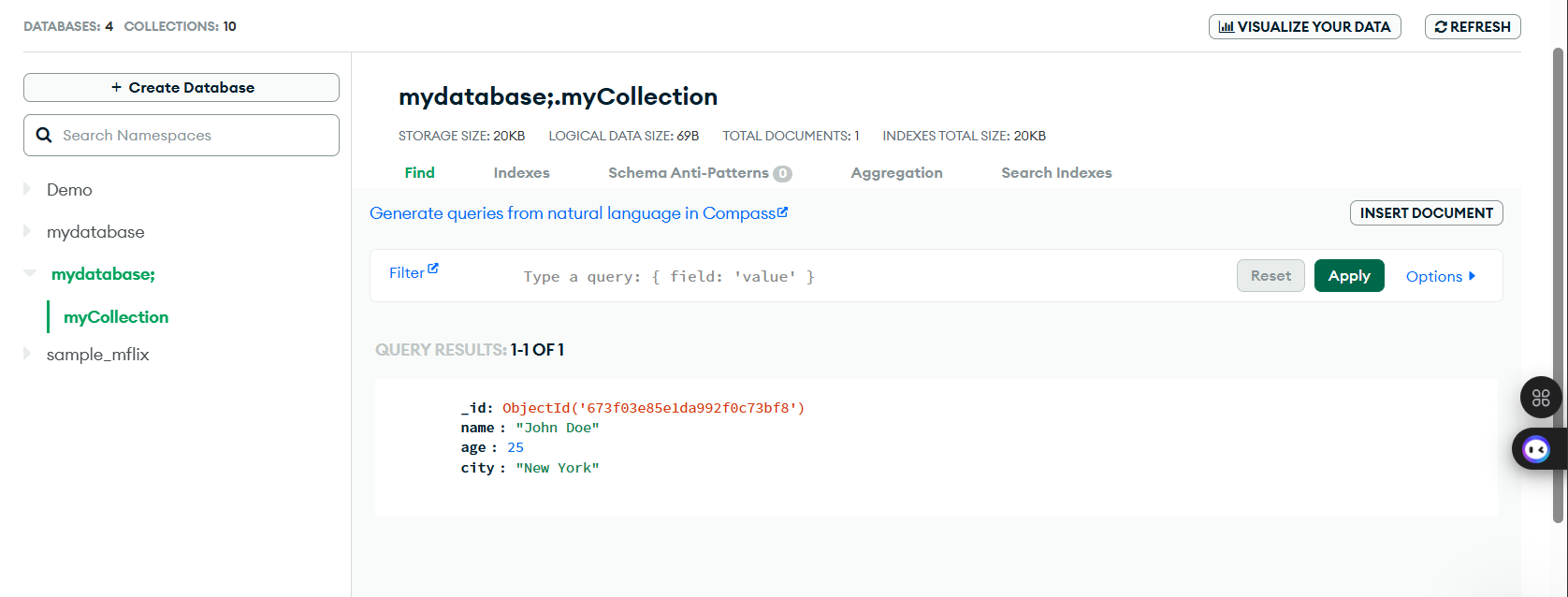
Description automatically generated**

* Switch to the database:

use mydatabase; // Replace with your database name

****

* CRUD Operations in MongoDB Shell
  + Create (Insert a Document)

**A computer screen with white text and green letters

Description automatically generated**

* + Read (Find a Document)

**A blue and white rectangular object

Description automatically generated with medium confidence**

* + Update (Modify a Document)

**A screenshot of a computer

Description automatically generatedA screen shot of a computer

Description automatically generated**

* + Delete (Remove a Document)

****

A screenshot of a computer

Description automatically generated