Week 9: Unstructured data; storage

Objective: In this lab session, you will learn:

1. Image data storage
2. Text data storage

MongoDB is a NoSQL database management system known for its flexibility, scalability, and ease of use. Unlike traditional relational databases, MongoDB stores data in flexible, JSON-like documents, making it suitable for various applications.

This section will use MongoDB Atlas and Google Colab.

MongoDB Atlas is a fully managed cloud database service provided by MongoDB, offering a hassle-free way to deploy, manage, and scale MongoDB databases in the cloud.

Step 1. Register on MongoDB Atlas

- Visit the MongoDB Atlas website here.
- Click the "Try Free" button to create an account and follow the registration process.

Step 2: Setting up Users

- After logging in to MongoDB Atlas
- Create a new project, and select the project
- Navigate to the "Database Access" tab from the left sidebar.
- Click the "Add New Database User" button to create a new user
- Enter the username and password for the new user and assign appropriate permissions

Step 3: Setting up IPs

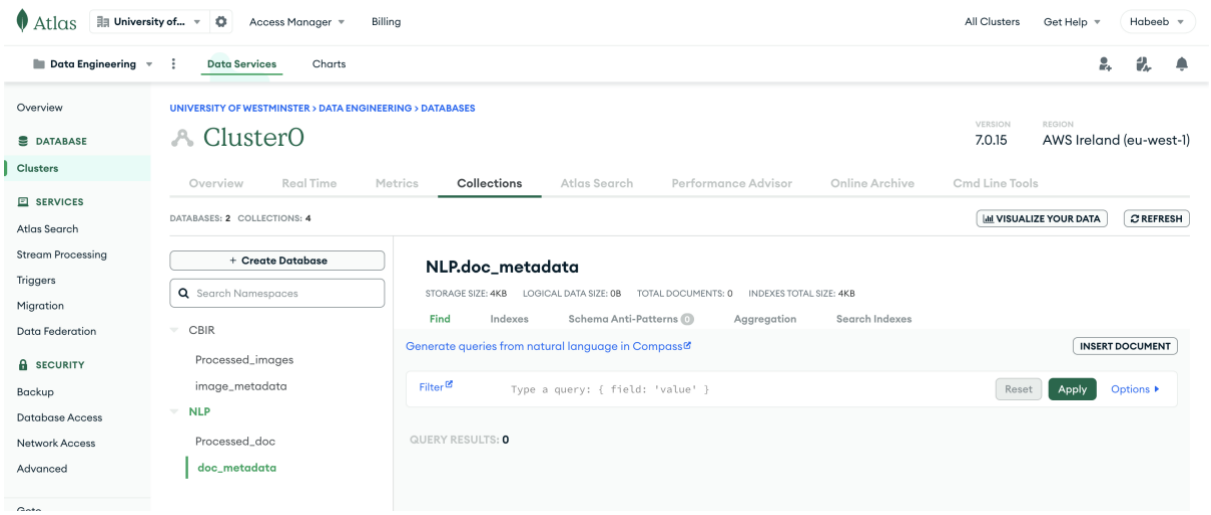
- In the MongoDB Atlas dashboard, go to the "Network Access" tab from the left sidebar.
- Click on the "Add IP Address" button.
- Add your current IP address to the whitelist to allow connections from your location.
- Add IP address: 0.0.0.0/0 to access the database from Google Collab

Step 4: Creating Clusters

- Go to the "Clusters" tab from the left sidebar and click on the "Create New Cluster"
- Choose the free cluster – M0, choose AWS as a cloud provider, Ireland as the region
- Click on the "Create Cluster" button to provision the cluster.

Step 5: Create a database and collection name

- Go to the cluster then click on add my own data
- Enter database and collection name. Enter one collection name while defining database.
- Come back to the created database, and click on the + to create another collection
- For the Image data, create CBIR db. and two collections: processed images and image metadata
- For the text data, create an NLP db. And two collections: processed doc and doc metadata



Step 6: Getting Connection String

- Once the cluster is created, click on the "Connect" button.
- Select "Connect Your Application" and copy the connection string provided. This string will be used to connect to MongoDB Atlas from Google Colab.
- Make sure you refer to the correct database and collection every time you make the connections

Step 7: Basic Installation

- Open Google Colab in your browser.
- Create a new notebook and name it week9_tutorial.

Step 8: Basic Installation

- In the first code cell, install the pymongo library by running 'pip install pymongo'.
- Import the necessary libraries in the next code cell:

```
import pymongo
from pymongo import MongoClient
```

Step 9: Connect to MongoDB Atlas:

- Paste the connection string obtained from MongoDB Atlas into your code.
- Replace <password> with the password for the MongoDB user you created and <dbname> with the database name you want to connect to.
- Use the following code snippet:

```
Image_connection = pymongo.MongoClient("mongodb+srv://baloguh:xzSW3Mq9rorf7pzH@cluster0.4ppxp.mongodb.net/CBIR?retryWrites=true&w=majority")


# Connect to the specified database and collection
db = Image_connection["CBIR"]  # Your database name
```

Step 10: Accessing the Database:

- Once connected, you can access your MongoDB database and perform operations like querying, inserting, updating, and deleting data.

Step 11: Test the connection by listing the collections

```
# Test the connection by listing collections (optional)
try:
    print(db.list_collection_names())
except Exception as e:
    print(f"Connection failed: {e}")
```

Step 12: Storing the Image metadata files into the database

- Upload the metadata file BB
- Select the collection where you want to query/load into the CBIR database

```
#lets do loading of the image metadata to image metadata collection
#defined the collection
image_metadata_collection = db["image_metadata"]  # Your collection name
```

```
import json
# Load JSON metadata file
with open('/content/image_metadata.json') as file:
    image1_data = json.load(file)

# Insert data into MongoDB collection
if isinstance(image1_data, list):
    image_metadata_collection.insert_many(image1_data)  # For a list of documents
else:
    image_metadata_collection.insert_one(image1_data)  # For a single document
```

Step 13: Load the processed image data from a folder to the database

- Upload the image data folder from BB
- Select the collection where you want to query/load into the CBIR database

```
#lets do loading of the processed image to processed image collection
#defined the collection
processed_collection = db["Processed_images"]  # Your collection name
```

```
import os
from bson import Binary

# Load and insert images
image_folder_path = '/content/Images'

for filename in os.listdir(image_folder_path):
    if filename.endswith(".png") or filename.endswith(".jpg"):
        with open(os.path.join(image_folder_path, filename), "rb") as image_file:
            binary_image = Binary(image_file.read())
            image_doc = {
                "filename": filename,
                "image_data": binary_image
            }
            processed_collection.insert_one(image_doc)
```

Step 14: Query the database to see if the processed image was loaded

- Let's check if one of the images, for example, image12.jpg, is in the DB

```
# Example query to find an image by filename
query = {"filename": "Image12.jpg"}
image_document = processed_collection.find_one(query)

if image_document:
    print("Image found:", image_document["filename"])
else:
    print("No image found with the specified criteria.")
```

Text data storage in MongoDB Atlas

Building on the earlier connection created, metadata was loaded, and textual data from week 8 was processed in MongoDB.

Step 1: Create a new connection; refer to NLP as a database.

```
text_connection = pymongo.MongoClient("mongodb+srv://baloguh:xzSW3Mq9rorf7pzH@cluster0.4ppxp.mongodb.net/NLP?retryWrites=true&w=majority")


# Connect to the specified database and collection
db = text_connection["NLP"]  # database name
```

```
# Test the connection by listing collections (optional)
try:
    print(db.list_collection_names())
except Exception as e:
    print(f"Connection failed: {e}")
```

Step 2: load the metadata of the processed document file.

```
#lets do loading of the image metadata to image metadata collection
#defined the collection
doc_metadata_collection = db["doc_metadata"]  # collection name
```

```
import json
# Load JSON metadata file
with open('/content/text_metadata.json') as file:
    doc1_data = json.load(file)

# Insert data into MongoDB collection
if isinstance(doc1_data, list):
    doc_metadata_collection.insert_many(doc1_data)  # For a list of documents
else:
    doc_metadata_collection.insert_one(doc1_data)  # For a single document
```

Step 3: load the processed document files

```
#lets do loading of the processed image to processed image collection
#defined the collection
processed_doc_collection = db["Processed_doc"]  # Your collection name
```

```
import os
from datetime import datetime

# Specify folder containing your documents
document_folder_path = '/content/Documents'

# Iterate over all files in the specified folder
for filename in os.listdir(document_folder_path):
    file_path = os.path.join(document_folder_path, filename)

    # Ensure we only process files (skip directories)
    if os.path.isfile(file_path):
        with open(file_path, "r") as file:
            # Read the contents of the file as text
            file_content = file.read()

            # Create a document for insertion into MongoDB
            document_data = {
                "filename": filename,
                "content": file_content,
                "metadata": {
                    "processed_date": datetime.utcnow().isoformat(),
                    "tags": ["text", "document"]  # Add any relevant tags as needed
                }
            }

            # Insert the document into the collection
            processed_collection.insert_one(document_data)
```

Extra activities.

This is part of the image texture extraction from week 8.

Here, Features like Contrast, and dissimilarity among others from GLCM are computed.

```
from PIL import Image
import numpy as np
from skimage.feature import graycomatrix, graycoprops
from skimage.color import rgb2gray

im_frame = Image.open('/content/fruits.jpg')
image = (255*rgb2gray(np.array(im_frame))).astype(np.uint8) # 'Your_2D_np_array'


# Generate GLCM
distances = [50] # Offset
angles = [np.pi/2]  # Vertical Direction
glcm = graycomatrix(image, distances=distances, angles=angles,levels=255)


# Calculate Features from GLCM
contrast = graycoprops(glcm, 'contrast')
dissimilarity = graycoprops(glcm, 'dissimilarity')
homogeneity = graycoprops(glcm, 'homogeneity')
energy = graycoprops(glcm, 'energy')
correlation = graycoprops(glcm, 'correlation')
```