Steps to connect to mongodb

1. Create an account on mongodb site
2. Create a database FREE account
3. Create a Cluster name
4. Click Finish
5. In database access the default user is created, click the edit button to change the password! The password is UNIQUE – DO NOT SHOW IT TO OTHERS!
6. In network access add all/ current IP address (to access the database)
7. Go back to databases and in the connect option choose:
8. Click drivers -> Choose Python -> in pycharm terminal install module python -m pip install "pymongo[srv]"==3.11 -> then run the script to test your connection

from pymongo.mongo\_client import MongoClient

uri = "connection string"

# Create a new client and connect to the server

client = MongoClient(uri)

# Send a ping to confirm a successful connection

try:

client.admin.command('ping')

print("Pinged your deployment. You successfully connected to MongoDB!")

except Exception as e:

print(e)

EXERCISE

from pymongo import MongoClient

# Replace "connection string" with your actual MongoDB connection string

uri = "connection string"

# Create a new client and connect to the MongoDB server using the provided connection string

client = MongoClient(uri)

# Send a ping to confirm a successful connection

try:

client.admin.command('ping')

print("Pinged your deployment. You successfully connected to MongoDB!")

except Exception as e:

print(e)

# Use the connected client for further MongoDB operations

# For example, let's perform basic CRUD operations as exercises

# Create or get a database named "school"

database = client["school"]

# Create a collection named "students" within the "school" database

students\_collection = database["students"]

# Exercise 1: Insert documents into the "students" collection

student1 = {"name": "Alice", "age": 20, "grade": "A"}

student2 = {"name": "Bob", "age": 22, "grade": "B"}

student3 = {"name": "Charlie", "age": 21, "grade": "C"}

# Insert the documents into the collection

students\_collection.insert\_many([student1, student2, student3])

# Exercise 2: Query all students and print their information

print("All students:")

all\_students = students\_collection.find()

for student in all\_students:

print(student)

# Exercise 3: Query students with a specific grade (e.g., "B") and print their information

print("\nStudents with grade 'B':")

grade\_b\_students = students\_collection.find({"grade": "B"})

for student in grade\_b\_students:

print(student)

# Exercise 4: Update the age of a student named "Alice"

students\_collection.update\_one({"name": "Alice"}, {"$set": {"age": 21}})

print("\nAfter updating Alice's age:")

print(students\_collection.find\_one({"name": "Alice"}))

# Exercise 5: Delete a student named "Charlie"

students\_collection.delete\_one({"name": "Charlie"})

print("\nAfter deleting Charlie:")

all\_students\_after\_deletion = students\_collection.find()

for student in all\_students\_after\_deletion:

print(student)

# Close the MongoDB connection

client.close()

**Make a program**

Library Book Management with MongoDB

Objective: Create a Python program that connects to MongoDB, manages books in a library, and stores/retrieves book information.

Create a Database:

Choose a name for your library database, e.g., "library\_db."

Use the connected client to get or create the library database.

Create Collections:

Create two collections: "books" and "authors."

Each collection should store information about books and authors, respectively.

Step 4: Insert Books and Authors

Prepare Data:

Define Python dictionaries with data for books and authors.

Each book should have information like title, author, publication year, etc.

Each author should have details like name, birthdate, nationality, etc.

Insert Data:

Use the "books" and "authors" collections to insert the prepared data into MongoDB.

Establish a relationship between books and authors by referencing author IDs in the books collection.

Step 5: Retrieve Book Information

Retrieve All Books:

Use the "books" collection to retrieve all books from the library.

Print each book's information, including the associated author details.

Retrieve Books by Author:

Allow the user to input an author's name.

Use the "authors" collection to find the author ID.

Use the "books" collection to retrieve and print all books by the specified author.

Step 6: Close the MongoDB Connection

Close Connection:

Always close the MongoDB connection using the close() method on the client object.

SOLUTION

from pymongo import MongoClient

# Replace "your\_connection\_string" with your actual MongoDB connection string

uri = "connection string”

# Create a new client and connect to the MongoDB server

client = MongoClient(uri)

# Get or create the library database and collections

library\_db = client["library\_db"]

books\_collection = library\_db["books"]

authors\_collection = library\_db["authors"]

# Insert sample data into the "authors" collection

author\_data = [

{"name": "J.K. Rowling", "birthdate": "July 31, 1965", "nationality": "British"},

{"name": "George Orwell", "birthdate": "June 25, 1903", "nationality": "British"},

{"name": "Jane Austen", "birthdate": "December 16, 1775", "nationality": "British"},

]

authors\_collection.insert\_many(author\_data)

# Insert sample data into the "books" collection, referencing author IDs

book\_data = [

{"title": "Harry Potter and the Sorcerer's Stone", "publication\_year": 1997, "author\_id": 1},

{"title": "1984", "publication\_year": 1949, "author\_id": 2},

{"title": "Pride and Prejudice", "publication\_year": 1813, "author\_id": 3},

]

books\_collection.insert\_many(book\_data)

# Retrieve all books with author details

all\_books = books\_collection.aggregate([

{

"$lookup": {

"from": "authors",

"localField": "author\_id",

"foreignField": "\_id",

"as": "author\_info"

}

},

{

"$unwind": "$author\_info"

}

])

print("All books in the library:")

for book in all\_books:

print(f"Title: {book['title']}")

print(f"Author: {book['author\_info']['name']}")

print(f"Publication Year: {book['publication\_year']}")

print("-----------------------")

# Retrieve books by a specific author (user input)

author\_name = input("Enter the author's name to retrieve their books: ")

author = authors\_collection.find\_one({"name": author\_name})

if author:

author\_books = books\_collection.find({"author\_id": author["\_id"]})

print(f"\nBooks by {author\_name}:")

for book in author\_books:

print(f"Title: {book['title']}")

print(f"Publication Year: {book['publication\_year']}")

print("-----------------------")

else:

print(f"No author found with the name: {author\_name}")

# Close the MongoDB connection

client.close()