**50. Write a Python program to perform following operation on MongoDB database**

1. Create collection "Book" with fields Book-name, Book-code, Book-Author, Book-Price, Bookpublication-year
2. Insert 5 documents
3. Find the books whose price between 500-800
4. Update price of book "Python programming" as 1000
5. Display all books in the order of publication Year

from pymongo import MongoClient

# Connect to MongoDB server (Assuming MongoDB is running locally)

client = MongoClient('mongodb://localhost:27017/')

# Select or create the database and collection

db = client['Library']

collection = db['Book']

# i) Create collection "Book" with the fields: Book-name, Book-code, Book-Author, Book-Price, Book-publication-year

# The collection and database are created automatically when we insert a document

# ii) Insert 5 documents

books = [

{"Book-name": "Python Programming", "Book-code": "B001", "Book-Author": "John Doe", "Book-Price": 650, "Book-publication-year": 2020},

{"Book-name": "Data Science", "Book-code": "B002", "Book-Author": "Jane Smith", "Book-Price": 700, "Book-publication-year": 2021},

{"Book-name": "Machine Learning", "Book-code": "B003", "Book-Author": "David Lee", "Book-Price": 750, "Book-publication-year": 2019},

{"Book-name": "Artificial Intelligence", "Book-code": "B004", "Book-Author": "Emily Brown", "Book-Price": 550, "Book-publication-year": 2018},

{"Book-name": "Database Systems", "Book-code": "B005", "Book-Author": "Michael Green", "Book-Price": 800, "Book-publication-year": 2022}

]

# Insert the books into the collection

collection.insert\_many(books)

# iii) Find the books whose price is between 500 and 800

print("Books with price between 500 and 800:")

price\_range\_books = collection.find({"Book-Price": {"$gte": 500, "$lte": 800}})

for book in price\_range\_books:

print(book)

# iv) Update price of book "Python Programming" to 1000

collection.update\_one({"Book-name": "Python Programming"}, {"$set": {"Book-Price": 1000}})

print("\nUpdated price for 'Python Programming' to 1000.")

# v) Display all books in the order of publication year

print("\nBooks sorted by publication year:")

sorted\_books = collection.find().sort("Book-publication-year", 1)

for book in sorted\_books:

print(book)

# Close the connection

client.close()

**51. Write a python program to connect with MongoDB Database. Create collection “restaurants” (Assume suitable structure). Query the “restaurants” collection to**

a. display all the documents in the collection restaurants

b. display the fields restaurant\_id, name, establishment\_year and cuisine for all the documents in the collection restaurant

c. find the restaurants who achieved a score more than 90.

d. arrange the name of the restaurants in ascending order

e. Update restaurant score for the establishment year 2019

from pymongo import MongoClient

# Connect to MongoDB server (Assuming MongoDB is running locally)

client = MongoClient('mongodb://localhost:27017/')

# Create or select the database

db = client['RestaurantDB']

# Create the 'restaurants' collection

collection = db['restaurants']

# Sample data to insert into the collection (Assumed struct ure)

restaurants\_data = [

{"restaurant\_id": "R001", "name": "The Gourmet Kitchen", "establishment\_year": 2015, "cuisine": "Italian", "score": 92},

**{"restaurant\_id": "R002", "name": "Spicy Delights", "establishment\_year": 2018, "cuisine": "Indian", "score": 85},**

{"restaurant\_id": "R003", "name": "Sushi World", "establishment\_year": 2019, "cuisine": "Japanese", "score": 96},

{"restaurant\_id": "R004", "name": "Burgers and Fries", "establishment\_year": 2020, "cuisine": "American", "score": 80},

{"restaurant\_id": "R005", "name": "Tacos and More", "establishment\_year": 2017, "cuisine": "Mexican", "score": 89}

]

# Insert data into the collection

collection.insert\_many(restaurants\_data)

# a) Display all the documents in the collection

print("All restaurants in the collection:")

for restaurant in collection.find():

print(restaurant)

# b) Display the fields restaurant\_id, name, establishment\_year, and cuisine for all documents

print("\nRestaurant details (id, name, establishment\_year, cuisine):")

for restaurant in collection.find({}, {"restaurant\_id": 1, "name": 1, "establishment\_year": 1, "cuisine": 1, "\_id": 0}):

print(restaurant)

# c) Find restaurants with a score more than 90

print("\nRestaurants with a score greater than 90:")

for restaurant in collection.find({"score": {"$gt": 90}}):

print(restaurant)

# d) Arrange the name of the restaurants in ascending order

print("\nRestaurants arranged by name (ascending order):")

for restaurant in collection.find().sort("name", 1): # 1 for ascending order

print(restaurant["name"])

# e) Update the score of restaurants established in 2019

collection.update\_many({"establishment\_year": 2019}, {"$set": {"score": 100}})

print("\nUpdated scores for restaurants established in 2019:")

for restaurant in collection.find({"establishment\_year": 2019}):

print(restaurant)

# Close the connection

client.close()

**52.Write a python program to connect with MongoDB Database. Create collection “movies” (title, writer, year, actors, director...consider this fields in movies collection). Query the “movies” collection to**

a. Get all documents

b. Get all document with director set to “Raj Kapoor”

c. get all documents where actors include "Amitabh Bachchan"

d. get all movies released in the 90s

e. get all movies released before the year 2000 or after 2010

f. Update some documents by adding some extra fields

g. Delete movie “movie\_name”

from pymongo import MongoClient

from datetime import datetime

# Connect to MongoDB server (Assuming MongoDB is running locally)

client = MongoClient('mongodb://localhost:27017/')

# Create or select the database

db = client['MovieDB']

# Create or select the 'movies' collection

collection = db['movies']

# Sample movie data (for illustration)

movies\_data = [

{"title": "Sholay", "writer": "Salim-Javed", "year": 1975, "actors": ["Amitabh Bachchan", "Dharmendra"], "director": "Ramesh Sippy"},

{"title": "Kabhi Kabhie", "writer": "Yes, it’s a love story", "year": 1976, "actors": ["Amitabh Bachchan", "Shashi Kapoor"], "director": "Yash Chopra"},

{"title": "Mera Naam Joker", "writer": "Raj Kapoor", "year": 1970, "actors": ["Raj Kapoor", "Simi Garewal"], "director": "Raj Kapoor"},

{"title": "Dilwale Dulhania Le Jayenge", "writer": "Aditya Chopra", "year": 1995, "actors": ["Shah Rukh Khan", "Kajol"], "director": "Aditya Chopra"},

{"title": "Lagaan", "writer": "Ashutosh Gowariker", "year": 2001, "actors": ["Aamir Khan", "Gracy Singh"], "director": "Ashutosh Gowariker"},

{"title": "3 Idiots", "writer": "Chetan Bhagat", "year": 2009, "actors": ["Aamir Khan", "R. Madhavan"], "director": "Rajkumar Hirani"},

{"title": "Barfi!", "writer": "Anurag Basu", "year": 2012, "actors": ["Ranbir Kapoor", "Priyanka Chopra"], "director": "Anurag Basu"},

{"title": "Dabangg", "writer": "Abhinav Kashyap", "year": 2010, "actors": ["Salman Khan", "Sonakshi Sinha"], "director": "Abhinav Kashyap"}

]

# Insert sample movies data into the collection

collection.insert\_many(movies\_data)

# a) Get all documents

print("All movies:")

for movie in collection.find():

print(movie)

# b) Get all documents where director is "Raj Kapoor"

print("\nMovies directed by Raj Kapoor:")

for movie in collection.find({"director": "Raj Kapoor"}):

print(movie)

# c) Get all documents where actors include "Amitabh Bachchan"

print("\nMovies featuring Amitabh Bachchan:")

for movie in collection.find({"actors": "Amitabh Bachchan"}):

print(movie)

# d) Get all movies released in the 90s

print("\nMovies released in the 90s:")

for movie in collection.find({"year": {"$gte": 1990, "$lt": 2000}}):

print(movie)

# e) Get all movies released before 2000 or after 2010

print("\nMovies released before 2000 or after 2010:")

for movie in collection.find({"year": {"$lt": 2000, "$gt": 2010}}):

print(movie)

# f) Update some documents by adding extra fields (e.g., 'genre' and 'rating')

collection.update\_many({}, {"$set": {"genre": "Drama", "rating": 8.5}}) # Adding 'genre' and 'rating' to all movies

print("\nUpdated movies with 'genre' and 'rating':")

for movie in collection.find():

print(movie)

# g) Delete a movie by its name (e.g., "Sholay")

collection.delete\_one({"title": "Sholay"})

print("\nDeleted movie 'Sholay' from the collection:")

for movie in collection.find():

print(movie)

# Close the connection

client.close()