Lab sheet 2(Report requires both queries and screenshots)

Name: J Viswaksena Roll No: AM.EN. U4AIE21035

Part I

Question-1

Run the below code on mongo console. It will insert 5 documents, which will serve as sample data for the next steps.

```
use training
db.languages.insert({"name":"java","type":"object oriented"})
db.languages.insert({"name":"python","type":"general purpose"})
db.languages.insert({"name":"scala","type":"functional"})
db.languages.insert({"name":"c","type":"procedural"})
db.languages.insert({"name":"c++","type":"object oriented"})
```

```
> use training
switched to db training
> show collections
> use training
switched to db training
> db.languages.insert({"name":"java", "type":"object oriented"})
WriteResult({ "nInserted" : 1 })
> db.languages.insert({"name":"scala", "type":"general purpose"})
WriteResult({ "nInserted" : 1 })
> db.languages.insert({"name":"python", "type":"functional"})
WriteResult({ "nInserted" : 1 })
> db.languages.insert({"name":"c", "type":"procedual"})
WriteResult({ "nInserted" : 1 })
> db.languages.insert({"name":"c++", "type":"object oriented"})
WriteResult({ "nInserted" : 1 })
> db.languages.find()
{ "_id" : ObjectId("655ddbb652fb1c663bb8aac7"), "name" : "java", "type" : "object oriented" }
{ "_id" : ObjectId("655ddbd052fb1c663bb8aac8"), "name" : "scala", "type" : "general purpose" }
{ "_id" : ObjectId("655ddc0452fb1c663bb8aac9"), "name" : "python", "type" : "functional" }
{ "_id" : ObjectId("655ddc1852fb1c663bb8aaca"), "name" : "c", "type" : "procedual" }
{ "_id" : ObjectId("655ddc2f52fb1c663bb8aacb"), "name" : "c++", "type" : "object oriented" }
```

Q1) Insert an entry for 'Haskell' programming language which is of type 'functional .

```
> db.languages.insert({"name":"haskell", "type":"functional"})
WriteResult({ "nInserted" : 1 })
> db.languages.find()
{ "_id" : ObjectId("655ddbb652fb1c663bb8aac7"), "name" : "java", "type" : "object oriented" }
{ "_id" : ObjectId("655ddbd052fb1c663bb8aac8"), "name" : "scala", "type" : "general purpose" }
{ "_id" : ObjectId("655ddc0452fb1c663bb8aac9"), "name" : "python", "type" : "functional" }
{ "_id" : ObjectId("655ddc1852fb1c663bb8aaca"), "name" : "c", "type" : "procedual" }
{ "_id" : ObjectId("655ddc2f52fb1c663bb8aacb"), "name" : "c++", "type" : "object oriented" }
{ "_id" : ObjectId("655ddc7b52fb1c663bb8aacc"), "name" : "haskell", "type" : "functional" }
```

Q2) Query for all functional languages.

```
> db.languages.find({"type":"functional"})
{ "_id" : ObjectId("655de0ff52fb1c663bb8aacd"), "name" : "haskell", "type" : "functional" }
{ "_id" : ObjectId("655de10952fb1c663bb8aace"), "name" : "python", "type" : "functional" }
>
```

Q3) Add 'Bjarne Stroustrup' as creator for c++.

```
> db.languages.update({"name":"cpp"},{$set:{"creator":"bjarne stroustrup"}})
WriteResult({ "nMatched" : 0, "nUpserted" : 0, "nModified" : 0 })
> db.languages.find()
{ "_id" : ObjectId("655ddbb652fb1c663bb8aac7"), "name" : "java", "type" : "object oriented" }
{ "_id" : ObjectId("655ddbd052fb1c663bb8aac8"), "name" : "scala", "type" : "general purpose" }
{ "_id" : ObjectId("655ddc0452fb1c663bb8aac9"), "name" : "python", "type" : "functional" }
{ "_id" : ObjectId("655ddc1852fb1c663bb8aaca"), "name" : "c", "type" : "procedual" }
{ "_id" : ObjectId("655ddc2f52fb1c663bb8aacb"), "name" : "c++", "type" : "object oriented" }
{ "_id" : ObjectId("655ddc7b52fb1c663bb8aacc"), "name" : "haskell", "type" : "functional" }
```

Q4) Delete all functional programming languages.

```
> db.languages.remove({"type":"functional"})
WriteResult({ "nRemoved" : 2 })
> db.languages.find()
{ "_id" : ObjectId("655ddbb652fb1c663bb8aac7"), "name" : "java", "type" : "object oriented" }
{ "_id" : ObjectId("655ddbd052fb1c663bb8aac8"), "name" : "scala", "type" : "general purpose" }
{ "_id" : ObjectId("655ddc1852fb1c663bb8aaca"), "name" : "c", "type" : "procedual" }
{ "_id" : ObjectId("655ddc2f52fb1c663bb8aacb"), "name" : "c++", "type" : "object oriented" }
>
```

Part II

Write a Python program that can:

- connect to the mongodb server.
- select a database named **training**.
- select a collection named **mongodb_glossary**.
- insert the following documents into the collection **mongodb_glossary**.

{"database":"a database contains collections"}

{"collection":"a collection stores the documents"}

{"document":"a document contains the data in the form or key value pairs."}

- query and print all the documents in the **training** database and **mongodb_glossary** collection.
- close the connection to the server.

Solution: https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDB0151EN-SkillsNetwork/labs/MongoDB/Lab%20%20Accessing%20MongoDB%20using%20Python.md.html

Python Code:

```
from pymongo import MongoClient
uri = "mongodb://127.0.0.1:27017/"
client = MongoClient(uri)
db = client.training
collection = db.mongodb_glossary
documents_to_insert = [
{"database": "a database contains collections"},
{"collection": "a collection stores the documents"},
{"document": "a document contains the data in the form of key-value
pairs."},
]
```

```
collection.insert_many(documents_to_insert)
documents_all = collection.find()
for document in documents_all:
 print(document)
client.close()
```