PyMongo

CMPSC 305 – Database Systems

ALLEGHENY COLLEGE

Let’s code

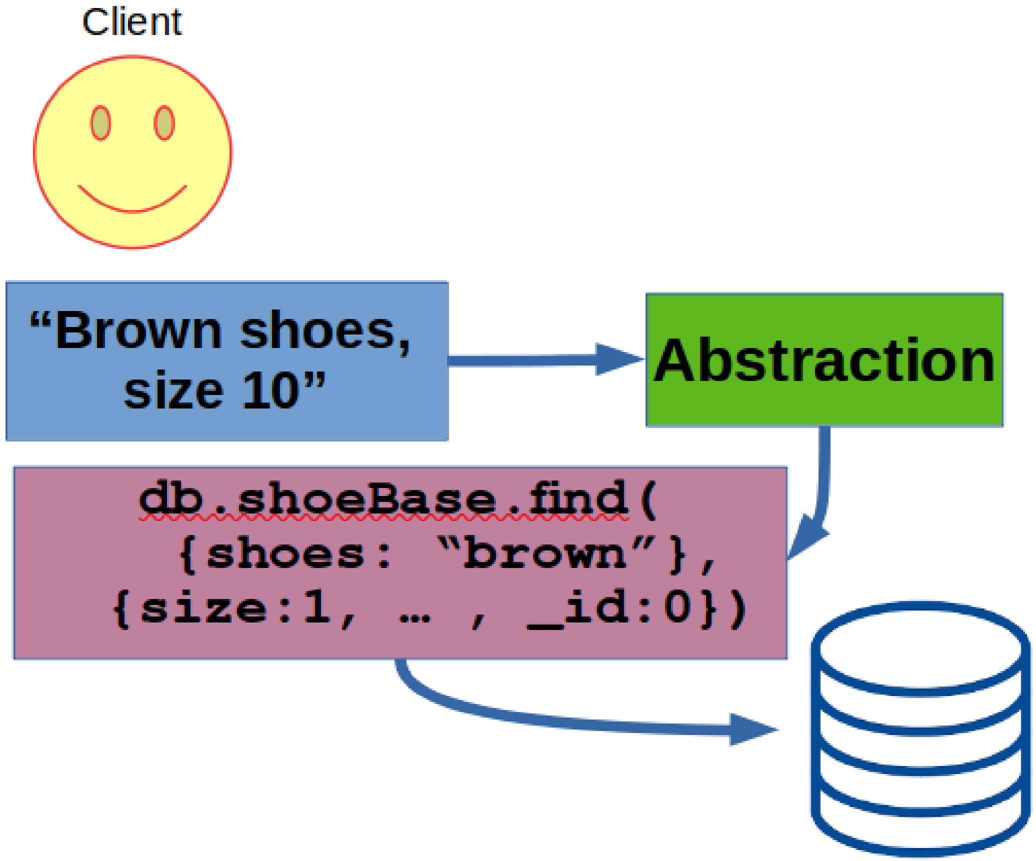
output

Abstraction

To make some process abstract is to hide (automate) some of the details that serve to complicate the process. The idea behind abstraction here is to create a more user-friendly experience by removing some of the the complexities of using Mongo databases.

Let’s code

output



```
db.shoeBase.find(
  {shoes: “brown”},
{size:1, … , _id:0})
```

Setting Up Virtual Environment

- Start the bash (make sure the Docker MongoDB container is running!)

```
sudo docker exec -it mongodb bash
```

Below are commands to enter when inside the container

- Download updated package information with apt

```
apt-get update
```

- Install an editor, Python3, Pip

```
apt-get install nano
apt-get install vim
apt-get install python3-pip
apt-get install python3-venv
```

Setting Up Virtual Environment

- Create a project directory

```
mkdir -p /workspace
cd /workspace
```

- Create virtual environment using Python

```
python3 -m venv myenv
# see the file tree
find . -not -path '*/\.*'
```

- Activate myenv the virtual environment

```
source myenv/bin/activate  # macOS/Linux
myenv\Scripts\activate     # Windows
```

- Deactivate the virtual environment

```
deactivate
```

- Install the pymongo software packages in the environment

```
pip3 install pymongo
```

Tools

Create file in; /mongodata and locate in container; /data/db

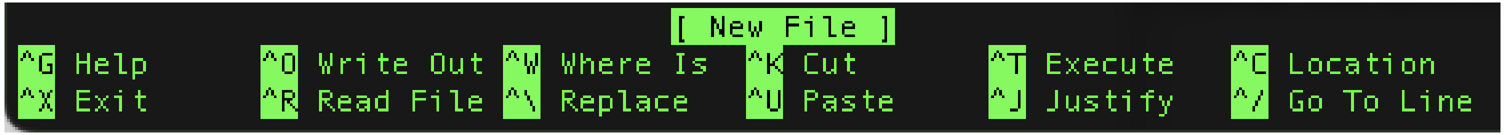
You could use Nano to begin coding
(or VSCode – just be sure you are in the correct directory when you work!!)

```
nano pymongoDemo.py
```

After coding, exit Nano and run your code

```
python3 pymongoDemo.py
```

Main Nano Menu Items



```
                                    [ New File ]
^G Help          ^O Write Out  ^W Where Is    ^K Cut         ^T Execute     ^C Location
^X Exit          ^R Read File  ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line
```

- Control-O :: ^O : Save
- Control-X :: ^X : Exit

Boilerplate code

Create file in; /mongodata and locate in container; /data/db

```
#!/usr/bin/env python3

# libraries
from pymongo import MongoClient
import string

# creating connections for communicating with MongoDB
client = MongoClient('localhost:27017')
db = client.mongodemo # The name of the collection is mongodemo
```

Boilerplate code

Create file in; /mongodata and locate in container; /data/db

```
# Define functions here!

# User Interaction

print("\t [+] Data BEFORE addition")
read() # call read function

print("\t [+] Insert some data")
insert() # call insert function()

print("\t [+] Data AFTER addition")
read() # call read function to view the changes

print("\t [+] Update Data")
update() # call update to ask for new information to replace existing

print("\t [+] Data AFTER Update")
read() # call read function to view the changes
```

Read Function

```
def read():
        """ function to read records from mongo db """
        try:
                empCol = db.Employee.find()
                print("\n Found: all data from DataEmployee \n")

                for emp in empCol:
                        print(f"\t [+] {emp}")
        except Exception as e:
                print(str(e))
# end of read()
```

Insert

```
def insert():
        """ Function to insert data into mongo db """
        employeeId = input('Enter Employee id :')
        employeeFirstName = input('Enter FirstName :')
        employeeLastName = input('Enter LastName :')
        employeeAge = input('Enter age :')
        employeeCountry = input('Enter Country :')

        # insert the data into the base
        try:
                db.Employee.insert_one(
                {
                "id": employeeId,
                "firstName":employeeFirstName,
                "lastName":employeeLastName,
                "age":employeeAge,
                "country":employeeCountry
                })
                print("\nInserted data successfully\n")

        except Exception as e:
                print(str(e))
# end of insert()
```

Update

```
def update():
        """ Function to update record to mongo db """
        print(" Update:")
        try:
                employeeId = input(' Enter Employee id :')
                employeeFirstName = input(' Enter FirstName :')
                employeeLastName = input(' Enter LastName :')
                employeeAge = input(' Enter age :')
                employeeCountry = input(' Enter Country :')

                # update the record with the new information
                db.Employee.update_one(
                {"id": employeeId},
                {
                "$set": {
                "firstName":employeeFirstName,
                "lastName":employeeLastName,
                "age":employeeAge,
                "country":employeeCountry
                }})
                print("\nRecords updated successfully. \n")
        except Exception as e:
                print(str(e))
        # end of update()
```

How do we find this database using Mongosh?

Useful commands for inside the container

Load the MongoDB Shell

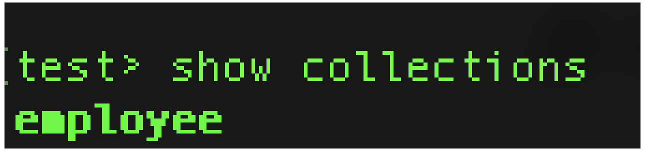
```
mongosh
```

How do we find this database using Mongosh?

Useful commands for inside the container

From Inside the MongoDB Shell: List collections

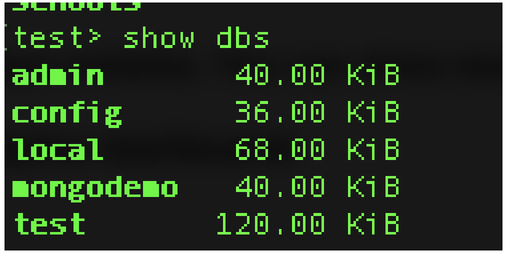
```
show collections
```



```
test> show collections
employee
```

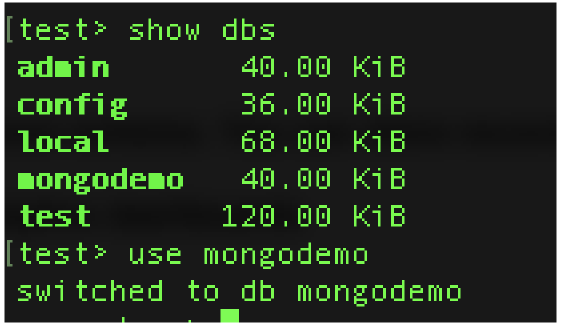
List the databases

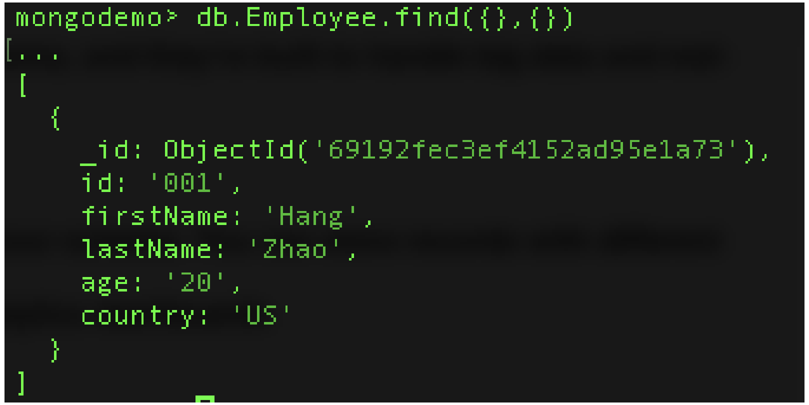
```
show dbs
```



```
test> show dbs
admin       40.00 KiB
config      36.00 KiB
local       68.00 KiB
mongodemo   40.00 KiB
test       120.00 KiB
```

Engaging the database made from PyMongo


```
[test> show dbs
admin         40.00 KiB
config        36.00 KiB
local         68.00 KiB
mongodemo     40.00 KiB
test         120.00 KiB
[test> use mongodemo
switched to db mongodemo
```


```
mongodemo> db.Employee.find({},{})
[...
 [
   {
     _id: ObjectId('69192fec3ef4152ad95e1a73'),
     id: '001',
     firstName: 'Hang',
     lastName: 'Zhao',
     age: '20',
     country: 'US'
   }
 ]
```

Choose mongodemo (where PyMongo placed all data)

```
use mongodemo
```

Do a catch-all query

```
db.Employee.find({},{})
```

Consider This ...

THINK

- Can you make a database in MongoDB?
- Can you create code to manage the data you use to populate this database?
- Can you think of applications for your code to other areas?