Pymongo

Unit 5

Introduction

- **Pymongo** is a python distribution containing tools for working with mongodb and is the recommended way to work with mongodb from python.

Prerequisite

- Python 3 and mongodb should be installed.
- Start the mongodb server.

Installation

- For Linux/mac OS platforms, issue the below command to add the <pymongo driver>.
  - sudo pip install pymongo
- For windows platforms, issue the below command to add the <pymongo driver>
  - python -m pip install pymongo
- To get a specific version of pymongo:
  - python -m pip install pymongo==3.5.1
- To upgrade using pip
  - python -m pip install --upgrade pymongo

How to connect to mongodb

- To make a connection to the database a mongo client has to be created against the running the <mongod> instance.
- For this, provide the arguments indicating the host and port where the database is running.
- If the mongodb server is running locally <default port for mongodb is 27017>, then write
  - from pymongo import MongoClient
  - con = mongoclient('localhost', 27017)
- If working on a large hybrid setup where the application server runs on a separate machine provide the ip address of that machine while creating the mongo client.
  - from pymongo import mongoclient
  - con = mongoclient('192.168.1.2', 27017)
- To connect on the default <host/port>, give the below command
  - con = mongoclient()
- One more method
  - conn = pymongo.mongoclient("mongodb://localhost")

How to create a database in mongodb?

- Mongodb voluntarily creates a database as you start to use it.
- For the testing purpose, execute the below step for db creation.
  - db = con.testdb
- Another approach is to use the dictionary-style access for db creation.
  - db = client['testdb']

How to access a collection in mongodb?

- A collection is a group of documents stored in the database.
  - It's same as a table in RDBMS.
- Access a mongodb collection in the same way as accessing the database in the last point.
  - my_coll = db.coll_name
- Or
- do it in the dictionary-style.
  - my_coll = db['coll_name']

How to add documents to a collection?

- Mongodb models data in JSON format.
- It uses the dictionary to store the records.
  - emp_rec = {'name':emp_name, 'address':emp_addr, 'id':emp_id}
- To work with collections, python mongodb module exposes a set of methods.
  - For example, the <insert()> method
    - rec_id = my_coll.insert(emp_rec)

How to query data in a collection?

- The python mongodb driver also gives you a method <find()> to query data from any mongodb collection.
- Run the <pretty()> method to format the query results.
- Here is the code for you to follow.
  - testdb.coll_name.find()
- To use pretty() if required
  - testdb.coll_name.find().pretty(){
    "_id" : objectid("7abf53ce1220a0213d"),
    "name" : emp_name,
    "address" : emp_addr,
    "id" : emp_id
    }

How to update data in a collection?

- To modify a collection, use any of the following python mongodb methods.
  - <update_one()>,
  - <update_many()>.
- Use the <$set> macro to change values.
- Note that the output is stored into a variable.
  - ret = db.my_coll.update_one(
    {"name": "post"},
    {"$set": {"category": "programming"}, "$currentdate": {"lastmodified": true}})
- To verify the result
  - ret.modified_count

How to remove data from a collection?

- The methods to delete the documents.
  - <delete_one()>,
  - <delete_many()>.
- Check out the below code snippet for removing more than one documents.
  - ret = db.posts.delete_many({"category": "general"})
- Call the following method to print the no. Of deleted records.
  - ret.deleted_count

How to drop the collection?

- To drop the created mongodb collection after completing the transactions call the method as given below.
  - con.drop()

How to close the connection?

- To close the open mongodb connection after completing the transactions call the method as given below.
  - con.close()

Programs

1.
```
#Program to create a db called pes and collection mca that contains 10
#documents having same value in field 'name'
import pymongo
from pymongo import MongoClient
# connect to the db on standard port
conn = pymongo.MongoClient("mongodb://localhost")

database = conn.pes            # attach to db
coll = database.mca            # specify the collection

for i in range(10):
   coll.insert_one({"name": "Lekha"})

docs = coll.find()
for i in docs:
   print(i)

#coll.drop();
```

2.
```
#Program to create a db called pes and collection mca that contains 10
#documents having field 'name' taken randomly from a dictionary called
#'people', 'number' having a random generated between 0 and 100 and
#user_id
import pymongo
import math
import random

# connect to the db on standard port
conn = pymongo.MongoClient("mongodb://localhost")

database = conn.pes                  # attach to db
coll = database.mca                  # specify the collection

people = ['lekha', 'isha', 'krishna', 'manish', 'varshini', 'pushpa']

for i in range(10):
   user_id = i;
   name = people[int(math.floor(random.random()*len(people)))];
   number = math.floor(random.random()*100);
   x = { "user_id": user_id, "name": name, "number": number };
   coll.insert(x);

#use a variable to store the documents
docs = coll.find()
for i in docs:
   print(i)

#num = coll.find().count()
#Using the docs try to get the number of records present in the
#collection
num=docs.count()
print(num)
```

3.
```
#Program to create a db called pes and collection mca that contains 10
#documents having field 'name' taken randomly from a dictionary called
#'people', 'number' having a random generated between 0 and 100 and
#user_id
#Uses try and exception handling to handle errors

import pymongo
import math
import random

# connect to the db on standard port
conn = pymongo.MongoClient("mongodb://localhost")


database = conn.pes                 # attach to db
coll = database.mca                 # specify the collection

def insert():
   people = ['lekha', 'isha', 'krishna', 'manish', 'varshini', 'pushpa']

   for i in range(10):
      user_id = i;
      name = people[int(math.floor(random.random()*len(people)))];
      number = math.floor(random.random()*100);
      x = { "user_id": user_id, "name": name, "number": number };
      coll.insert(x);

insert()

try:
   docs = coll.find()
   for i in docs:
      print(i)

except Exception as e:
   print ("Error trying to read collection:", type(e), e)

num = coll.find().count()
print(num)
```

4.
```
#Program to update a db called pes and collection mca and Update with all
#names as Lekha
import pymongo
import math
import random
from pymongo import MongoClient

# connect to the db on standard port
conn = pymongo.MongoClient("mongodb://localhost")


database = conn.pes                 # attach to db
coll = database.mca                 # specify the collection

coll.update_many({}, {'$set': {"name": "Lekha"}})

docs = coll.find()
for i in docs:
   print(i)

num = coll.find().count()
print("The total records updated are")
print(num)
```

```
#Delete all records
rec=coll.delete_many({})

print("The number of records deleted are ")
print(rec.deleted_count)
```

5.
```
#Program to use dictionary to insert documents
#Use dictionary to display the documents with specifications
import pymongo
import sys
from pymongo import MongoClient

# connect to the db on standard port
connection = pymongo.MongoClient()

db = connection.pes                  # attach to db
collection = db.mca                  # specify the collection


st = {'name': 'Lekha', 'number': 10}

collection.insert_one(st)

q = {'name': 'Lekha'}
p = {'_id': 0, 'name': 1}

doc = collection.find(q, p)

for d in doc:
    print (d)
print(doc.count())

doc = collection.find()

for d in doc:
    print (d)
print(doc.count())
```

6.
```
#Program using the function add to insert document into the collection
#called mca and database called pes
import pymongo
import math
import random

# connect to the db on standard port
conn = pymongo.MongoClient("mongodb://localhost")

try:
   database = conn.pes                  # attach to db
   coll = database.mca                  # specify the collection

   def add():
      people = ['lekha', 'isha', 'krishna', 'manish', 'varshini',
'pushpa']

      for i in range(10):
         user_id = i;
         name = people[math.floor(random.random()*len(people))];
         number = math.floor(random.random()*100);
         x = { "user_id": user_id, "name": name, "number": number };
         coll.insert(x);
   add()

except Exception as e:
   print ("Error trying to connect:", type(e), e)
```

```
num = coll.find().count()
print(num)
```

7.
```
#Program to insert documents into a collection called emp using the
#function
#The user types the details to be entered and the number of records to be
#entered.

import pymongo
import sys
from pymongo import MongoClient

# connect to the db on standard port
connection = pymongo.MongoClient("mongodb://localhost:27017")

db = connection.pes  # attach to db
collection = db.emp  # specify the collection

# Function to insert data into mongo db
def insert():
    try:
        employeeId = input('Enter Employee id :')
        employeeName = input('Enter Name :')
        employeeAge = input('Enter age :')
        employeeCountry = input('Enter Country :')
        db.emp.insert_one(
            {
                "id": employeeId,
                "name": employeeName,
                "age": employeeAge,
                "country": employeeCountry
            })
    except Exception as e:
        print(str(e))

n=input("Entering the number of documents needed")
for i in range(0,int(n)):
    insert()
```

8.
```
#Python Program to insert, delete, find based on user's choice
import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["pes"]

mycol = mydb["customers"]


def insert():
    custId = input('Enter Customer id :')
    custName = input('Enter Name :')
    custAge = int(input('Enter age :'))
    custAddress = input('Enter Address :')
    mycol.insert_one(
        {
            "id": custId,
            "name": custName,
            "age": custAge,
            "address": custAddress
        })


n=input("Entering the number of documents needed")
for i in range(0,int(n)):
```

```
    insert()

print("All details of Customers")
for x in mycol.find({}, {"_id": 0, "name": 1, "address": 1, "age": 1}):
    print(x)

mydoc = mycol.find({"age": {"$gte": 50}}, {"_id": 0, "name": 1})

print("All customers having an age greater than 50")
for x in mydoc:
    print(x)

add= input("Enter the address to be found")
'''print("All customers staying in ") & add'''
mydoc = mycol.find({"address":add}, {"_id":0, "name": 1})
for x in mydoc:
    print(x)
c = mycol.find({"address": add},{}).count()
print(c)

id = input("Enter the id of the record to be deleted")
res=mycol.find_one_and_delete({"id": id})
print(res)
mydoc=mycol.find({}, {"_id": 0, "id": 1, "name": 1, "address": 1, "age":
1})
for x in mydoc:
    print(x)
```

9.

```
#Aggregate functions in pymongo

import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["pes"]

mycol = mydb["employee"]


def insert():
    empId = input('Enter Customer id :')
    empName = input('Enter Name :')
    empSalary = int(input('Enter Salary :'))
    empDept = input('Enter the Department')
    mycol.insert_one(
        {
            "id": empId,
            "name": empName,
            "salary": empSalary,
            "dept": empDept
        })


n=input("Entering the number of documents needed")
for i in range(0,int(n)):
    insert()

print("The documents inserted are")
for x in mycol.find({}, {"_id": 0, "name": 1, "salary":  1, "dept": 1}):
    print(x)
pipe = [{'$group': {'_id': "$dept", 'Salary_sum' : {'$sum': "$salary"}}}]
print("sum of salary of all employee working in the same department")
for x in mycol.aggregate(pipeline = pipe):
    print(x)
pipe = [{'$group' : {'_id' : 'null', 'salary_sum' : {'$sum' :
"$salary"}}}]
```

```
print("sum of salary of all employees")
for x in mycol.aggregate(pipeline = pipe):
    print(x)

print(" sum of salary of all employee in same department where salary >
800000")
pipe=[
  { '$match': { 'salary' : { '$gt': 800000} } },
  {'$group' : {'_id' : "$dept", 'salary_sum' : {'$sum' : "$salary"}}}]
for x in mycol.aggregate(pipeline = pipe):
    print(x)
```