pymongo demonstration

Based on examples from https://www.mongodb.com/blog/post/getting-started-with-python-and-mongodb (https://www.mongodb.com/blog/post/getting-started-with-python-and-mongodb)

In [1]:

```
import sys
```

In [2]:

```
!{sys.executable} -m pip install pymongo --user
```

```
Requirement already satisfied: pymongo in /Users/alexk/Library/Python/
3.9/lib/python/site-packages (3.12.1)
WARNING: Value for scheme.headers does not match. Please report this t
o <https://github.com/pypa/pip/issues/10151>
distutils: /Users/alexk/Library/Python/3.9/include/python3.9/UNKNOWN
sysconfig: /Users/alexk/Library/Python/3.9/include/UNKNOWN
WARNING: Additional context:
user = True
home = None
root = None
prefix = None
WARNING: You are using pip version 21.2.1; however, version 21.3.1 is
 available.
You should consider upgrading via the '/Library/Frameworks/Python.fram
ework/Versions/3.9/bin/python3.9 -m pip install --upgrade pip' comman
d.
```

In [3]:

```
# pprint library is used to make the output look more pretty
from pprint import pprint

# randint library needed to generate some randomise content of the database
from random import randint
```

To establish a connection to MongoDB with PyMongo you use the MongoClient class.

Connect to MongoDB, change the URL to reflect your own connection string

Installation and running: https://docs.mongodb.com/manual/tutorial/install-mongodb-on-os-x/ (https://docs.mongodb.com/manual/tutorial/install-mongodb-on-os-x/)

If you don't want/need a background service you can just run:

```
    mongod --config /usr/local/etc/mongod.conf
```

and terminate it after the demo

In [4]:

```
from pymongo import MongoClient
```

In [5]:

```
#Step 1: Connect to MongoDB - Note: Change connection string as needed
client = MongoClient(port=27017)
db=client.business
```

In [6]:

```
#Step 2: Create sample data
names = ['Kitchen','Animal','State', 'Tastey', 'Big','City','Fish', 'Pizza','Goat',
company_type = ['LLC','Inc','Company','Corporation']
company_cuisine = ['Pizza', 'Bar Food', 'Fast Food', 'Italian', 'Mexican', 'Americar
for x in range(1, 501):
    business = {
        'name' : names[randint(0, (len(names)-1))] + ' ' + names[randint(0, (len(nam
        'rating' : randint(1, 5),
        'cuisine' : company_cuisine[randint(0, (len(company_cuisine)-1))]
    }
    #Step 3: Insert business object directly into MongoDB via insert_one
    result=db.reviews.insert_one(business)
    #Step 4: Print to the console the ObjectID of the new document
    print('Created {0} of 500 as {1}'.format(x,result.inserted_id))
#Step 5: Tell us that you are done
print('finished creating 500 business reviews')
```

```
Created 1 of 500 as 61965e71e18af34b722e9815
Created 2 of 500 as 61965e71e18af34b722e9816
Created 3 of 500 as 61965e71e18af34b722e9817
Created 4 of 500 as 61965e71e18af34b722e9818
Created 5 of 500 as 61965e71e18af34b722e9819
Created 6 of 500 as 61965e71e18af34b722e981a
Created 7 of 500 as 61965e71e18af34b722e981b
Created 8 of 500 as 61965e71e18af34b722e981c
Created 9 of 500 as 61965e71e18af34b722e981d
Created 10 of 500 as 61965e71e18af34b722e981e
Created 11 of 500 as 61965e71e18af34b722e981f
Created 12 of 500 as 61965e71e18af34b722e9820
Created 13 of 500 as 61965e71e18af34b722e9821
Created 14 of 500 as 61965e71e18af34b722e9822
Created 15 of 500 as 61965e71e18af34b722e9823
Created 16 of 500 as 61965e71e18af34b722e9824
Created 17 of 500 as 61965e71e18af34b722e9825
Created 18 of 500 as 61965e71e18af34b722e9826
Created 19 of 500 as 61965e71e18af34b722e9827
```

In [7]:

```
fivestar = db.reviews.find_one({'rating': 4})
print(fivestar)
```

```
{'_id': ObjectId('618d40c5ceb4d06ed8f151d8'), 'name': 'Pizza Salty Cor
poration', 'rating': 4, 'cuisine': 'Pizza'}
```

In [8]:

```
fivestar = db.reviews.find_one({'cuisine': 'Pizza'})
print(fivestar)
```

```
{'_id': ObjectId('618d40c5ceb4d06ed8f151d8'), 'name': 'Pizza Salty Cor
poration', 'rating': 4, 'cuisine': 'Pizza'}
```

In [9]:

```
rating = 5
fivestarcount = db.reviews.find({'rating': rating})
print(fivestarcount.collection.count_documents({'rating': rating}))
```

```
354
```

In [10]:

```
# Now let's use the aggregation framework to sum the occurrence of each rating acros
print('\nThe sum of each rating occurance across all data grouped by rating ')
stargroup=db.reviews.aggregate(
# The Aggregation Pipeline is defined as an array of different operations
[
# The first stage in this pipe is to group data
{ '$group':
    { '_id': "$rating",
     "count" :
                 { '$sum' :1 }
    }
},
# The second stage in this pipe is to sort the data
{"$sort":  { "_id":1}
}
# Close the array with the ] tag
] )
# Print the result
for group in stargroup:
    print(group)
```

```

The sum of each rating occurance across all data grouped by rating
{'_id': 1, 'count': 362}
{'_id': 2, 'count': 370}
{'_id': 3, 'count': 368}
{'_id': 4, 'count': 346}
{'_id': 5, 'count': 354}
```

Updating data with PyMongo

In [11]:

```
ASingleReview = db.reviews.find_one({"cuisine":"Vegetarian"})
print('A sample document:')
pprint(ASingleReview)

result = db.reviews.update_one({'_id' : ASingleReview.get('_id') }, {'$inc': {'likes
print('Number of documents modified : ' + str(result.modified_count))

UpdatedDocument = db.reviews.find_one({'_id':ASingleReview.get('_id')})
print('The updated document:')
pprint(UpdatedDocument)
```

```
A sample document:
{'_id': ObjectId('618d40c5ceb4d06ed8f151d7'),
 'cuisine': 'Vegetarian',
 'likes': 3,
 'name': 'Fish Tastey Corporation',
 'rating': 5}
Number of documents modified : 1
The updated document:
{'_id': ObjectId('618d40c5ceb4d06ed8f151d7'),
 'cuisine': 'Vegetarian',
 'likes': 4,
 'name': 'Fish Tastey Corporation',
 'rating': 5}
```

Deleting documents

In [12]:

```
result = db.reviews.delete_many({"cuisine": "Bar Food"})
```

In [13]:

```
result
```

Out[13]:

```
<pymongo.results.DeleteResult at 0x7fe9a0757bc0>
```

In [14]:

```
result.deleted_count
```

Out[14]:

```
67
```

Administrative interface

In [15]:

```
client = MongoClient("mongodb://localhost:27017/")
db=client.admin
```

In [16]:

```
# Issue the serverStatus command and print the results
serverStatusResult=db.command("serverStatus")
pprint(serverStatusResult)
```

```
{'asserts': {'msg': 0,
             'regular': 0,
             'rollovers': 0,
             'tripwire': 0,
             'user': 7,
             'warning': 0},
 'catalogStats': {'capped': 0,
                  'collections': 1,
                  'internalCollections': 3,
                  'internalViews': 0,
                  'timeseries': 0,
                  'views': 0},
 'connections': {'active': 3,
                 'available': 2042,
                 'awaitingTopologyChanges': 2,
                 'current': 6,
                 'exhaustHello': 0,
                 'exhaustIsMaster': 0,
                 'threaded': 6,
                 'totalCreated': 6}
```