M1 Informatique
Systèmes de gestion de documents

TP 3

Python – PyMongo

Python est un langage de scripts qui permet d'écrire des programmes simples ou complexes.
Il a une syntaxe simplifiée, très compacte, qui intègre une approche orientée objet avec de l'héritage multiple et de la surcharge d'opérateurs. Il possède des caractéristiques spécifiques, comme la dynamicité qui permet l'interprétation de chaînes de caractères, la possibilité pour un objet de se rajouter ou supprimer des attributs ou des méthodes, ou même de changer de classe, etc.
De nombreuses documentations sont disponibles en ligne sur ce langage, le site https://docs.python.org/fr/3/tutorial/index.html propose un tutoriel pour une prise en main du langage python.
Le driver PyMongo permet d'accéder et de travailler avec une base de données MongoDB sous un interpréteur Python.

Exercice 1 - Prise en main du langage Python

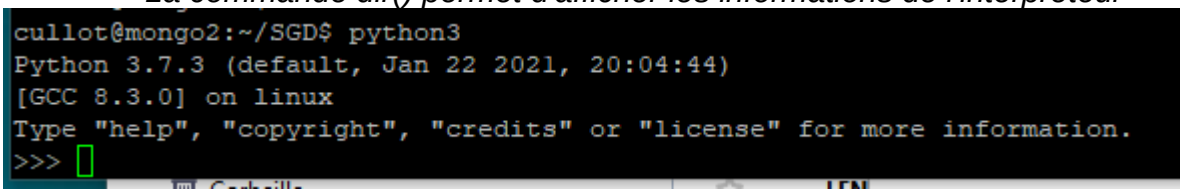
En salle TP, plusieurs versions de l'interpréteur python dont la version 3.7.3, sont disponibles.
Lancement de l'interpréteur pour python

1. Se connecter au serveur mongo2 par
   ssh login@mongo2 puis donner votre mot de passe réseau

rq : le serveur mongo a été réinstallé (c'est maintenant mongo2), par rapport aux précédents TP

2. Se placer dans le répertoire de travail (qui pourra contenir les scripts python .py), par exemple SGD
3. Lancer l'interpréteur avec la commande
   python3

La commande dir() permet d'afficher les informations de l'interpréteur



En utilisant le tutoriel de python https://docs.python.org/fr/3/tutorial/index.html
Etudier et tester les exemples donnés pour

a. Les instructions usuelles (if et for)
b. Les structures de données (listes, tuples et séquences)

Taper les commandes sous un éditeur et effectuer des copier/coller sous l'interpréteur, ou bien créer des scripts python par exemple test.py, et charger ce fichier sous l'interpréteur à l'aide de la commande : import test si le fichier est dans le répertoire courant
Vous pouvez aussi taper vos commandes directement sous l'interpréteur pour les petits tests

Exercice 2 : Prise en main de PyMongo

Le driver pyMongo permet d'accéder à une base de données MongoDB depuis l'interpréteur python. En salle de TP (ou en VPN dans les salles), pour établir une connexion de l'interpréteur python pour accéder à la base de données python, suivre les étapes suivantes (pour l'utilisateur xx123456), sous l'interpréteur python3

```
>>> from pymongo import MongoClient
>>> from bson.objectid import ObjectId
>>> c=MongoClient("mongo2.iem", port=27017, username=" XX123456 ", password=" XX123456
", authSource=" XX123456 ", authMechanism="SCRAM-SHA-1")
>>> db=c.XX123456          → si XX123456 est le nom de l'instance de la base
>>> db                     → donne les informations de connexion
>>> from pprint import pprint → pour utiliser pprint
```

Il est alors possible d'effectuer des requêtes sur les collections qui ont été créées dans la base avec des commandes "similaires" à celles disponibles dans mongoDB comme ci-dessous, avec une collection personnes ou une collection agence.

```
>>>
>>> db.personnes.find_one({ "nom" : "Durand"})
{'age': 45.0, '_id': ObjectId('5a119071b540c085a9c4d9fc'), 'nom': 'Durand'}
>>>
>>>
>>> collection=db.personnes
>>> collection.find_one()
{'age': 32.0, '_id': ObjectId('5a118f89b540c085a9c4d9fb'), 'nom': 'Dupont'}
>>>
>>> collection.find_one({ "nom" : "Durand"})
{'age': 45.0, '_id': ObjectId('5a119071b540c085a9c4d9fc'), 'nom': 'Durand'}
>>>
>>> collection.find_one({ '_id' : ObjectId('5a118f89b540c085a9c4d9fb')})
{'age': 32.0, '_id': ObjectId('5a118f89b540c085a9c4d9fb'), 'nom': 'Dupont'}
>>>
>>>
```

Il est aussi possible d'intégrer du script python et les commandes de pyMongo.

```
>>>
>>> col=db.agences
>>> col.find_one({ "numAgence" : "444"})
{'numAgence': '444', '_id': ObjectId('5a08a5576ed51876e7af6181'), 'adresse': 'Dijon', 'nom': 'Agence Dijon Valmy'}
>>>
>>> for doc in col .find(): doc
...
{'numAgence': '234', '_id': ObjectId('5a08a22f6ed51876e7af617e'), 'adresse': 'Dijon', 'nom': 'Agence Dijon Valmy'}
{'numAgence': '234', '_id': 110.0, 'adresse': 'Dijon', 'nom': 'Agence Dijon Valmy'}
{'numAgence': '555', '_id': ObjectId('5a08a2326ed51876e7af617f'), 'adresse': 'Dijon', 'nom': 'Agence Dijon Mansard'}
{'numAgence': '555', '_id': ObjectId('5a08a2326ed51876e7af6180'), 'adresse': 'Dijon', 'nom': 'Agence Dijon Universite'}
{'numAgence': '444', '_id': ObjectId('5a08a5576ed51876e7af6181'), 'adresse': 'Dijon', 'nom': 'Agence Dijon Valmy'}
>>>
```

En utilisant la documentation du guide de MongoDB (chapitre Python et MongoDB) et collections et les requêtes vues au TD2 et TP2, tester certaines requêtes de votre choix, sous l'interpréteur python, en les adaptant.

Exercice 3 : PyMongo

En se basant sur des documents (issus de l'exo5 du TD1), centré sur « étudiant » de la forme donnée en annexe, donner les scripts python répondant aux questions suivantes :

1. Quelle est la moyenne des notes de l'étudiant Boris Karloff ? (en utilisant un « aggregate » puis sans l'utiliser)
2. Quelle est la note maximale obtenue par un étudiant de la collection ? (en utilisant un « aggregate » puis sans l'utiliser)
3. Calculer le pourcentage d'étudiants admis (dont la moyenne est supérieure ou égale à 10) (sans « aggregate »)
4. Quel est l'étudiant qui a validé le plus de crédits ECTS ? Afficher son nom et le nombre de crédits validés (sans « aggregate »)
5. Afficher les titres des cours qui n'ont pas de prérequis, sans doublon (en utilisant un « aggregate »)
6. Quel est le nombre de prérequis que doit avoir suivis l'étudiant Marcel Campion ? (en utilisant un « aggregate »)
7. Tester d'autres scripts de votre choix (éventuellement sur d'autres collections utilisées en TP2)

ANNEXE (Collection etudiants)

```
db.etudiants.insert(
 {
    "_id": "6546",
    "nom": "Marcel Campion",
    "UE": [{"id": "info1a",
            "titre": "Java",
            "description": "initiation a la prog",
            "credits": 6,
            "prerequis": [ ],
            "note": 06
           },
           {"id": "info1b",
            "titre": "Web",
            "description": "initiation a la prog web",
            "credits": 12,
            "prerequis": [ ],
            "note": 15
           },
           {"id": "info4c",
            "titre": "Fondements de l'informatique",
            "description": "prog rec",
            "credits": 8,
            "prerequis": [{"id": "info1a",
                           "titre": "Java",
                           "description": "initiation a la
prog",
                           "credits": 6
                          }],
            "note": 04}
          ]
 }
)
```

```
db.etudiants.insert(
{
    "_id": "3215",
    "nom": "Boris Karloff",
    "UE": [{"id": "info2a",
            "titre": "Java",
            "description": "POO",
            "credits": 6,
            "prerequis": [
              {"id": "info1a",
               "titre": "Java",
               "description": "initiation a la prog",
               "credits": 6
              }
            ],
            "note": 17
           },
           {"id": "info2b",
            "titre": "Interfaces visuelles",
            "description": "Swing",
            "credits": 9,
            "prerequis": [
              {"id": "info1a",
               "titre": "Java",
               "description": "initiation a la prog",
               "credits": 6
              }
            ],
            "note": 6
           },
           {"id": "info1b",
            "titre": "Web",
            "description": "initiation a la prog web",
            "credits": 12,
            "prerequis": [ ],
            "note": 15
           }
          ]
 }
)
```