Homework 3 PyMongo queries

Name: Dipak Bange

Semester: Spint 23

1. Install pymongo

```
!pip install pymongo
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pymongo
  Downloading pymongo-4.3.3-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (492 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 492.1/492.1 KB 6.4 MB/s eta 0:00:00
Collecting dnspython<3.0.0,>=1.16.0
  Downloading dnspython-2.3.0-py3-none-any.whl (283 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 283.7/283.7 KB 24.6 MB/s eta 0:00:00
Installing collected packages: dnspython, pymongo
Successfully installed dnspython-2.3.0 pymongo-4.3.3
```

2. Import pymongo and connect to your database on atlas using the connection string

```
from pymongo import MongoClient
from pprint import pprint
from bson import json_util

# set up a connection to the MongoDB cluster
client = MongoClient("mongodb+srv://onlyforcsgo1230:xd1234XD@cluster0.653psma.mongodb.net/test?retryWrites=true&w=majority")

# access a specific database in the cluster
db = client.pokemon_db

# access a specific collection in the database
collection = db["pokemons"]
```

3. Check server status

```
serverStatusResult=db.command("serverStatus")
```

```
serverStatusResult
```

```
      'update': 0,
      'delete': 0,
      'getmore': 0,
      'command': 0,
      'deprecated': {'query': 0, 'getmore': 0}},
     'repl': {'topologyVersion': {'processId': ObjectId('64011098c33a7edef8163748'),
       'counter': 6},
      'hosts': ['ac-isdtpns-shard-00-00.653psma.mongodb.net:27017',
       'ac-isdtpns-shard-00-01.653psma.mongodb.net:27017',
       'ac-isdtpns-shard-00-02.653psma.mongodb.net:27017'],
```

```
 'oldestRequiredTimestampForCrashRecovery': Timestamp(1679677192, 13),
 'supportsPendingDrops': True,
 'dropPendingIdents': 8,
 'supportsSnapshotReadConcern': True,
 'readOnly': False,
 'persistent': True,
 'backupCursorOpen': False,
 'supportsResumableIndexBuilds': True},
'mem': {'bits': 64,
 'resident': 0,
 'virtual': 0,
 'supported': True,
 'mapped': 0,
 'mappedWithJournal': 0},
'metrics': {'aggStageCounters': {'search': 0,
  'searchBeta': 0,
  'searchMeta': 0},
 'operatorCounters': {'match': {'text': 0, 'regex': 0}},
 'atlas': {'connectionPool': {'totalCreated': 532979}}},
'ok': 1.0,
'$clusterTime': {'clusterTime': Timestamp(1679677215, 697),
 'signature': {'hash': b'\x0b\xd8\xf2p\xdf@@4\xe7\x91\x99\xf6{j\x9b\x8a\x0fk\xa7v',
  'keyId': 7164370397394108457}},
'operationTime': Timestamp(1679677215, 694),
'opLatencies': {'reads': {'latency': 267454, 'ops': 89},
 'writes': {'latency': 233524, 'ops': 1},
 'commands': {'latency': 54587090567, 'ops': 3299}},
```

4. Check the list of collections present in the pokemon_db

```
1 db = client.pokemon_db
2 print(db.list_collection_names())
```

```
['pokemons']
```

> If you got till here without any errors, that means you have been able to complete the whole database and cluster creation and then finally connecting to your database using MongoClient. You have just earned **15pts**

5.1 Sample query 1 - Find data about Bulbasaur and Ivysaur

Find out all the details about Bulbasaur and Ivysaur

```
1 result = db.pokemons.find({'$or':[{'name': 'Bulbasaur'}, {'name':'Ivysaur'}]}, {"_id": 0, "id": 0})
```

```
1 print(json_util.dumps(result, indent=2))
```

```
[
  {
    "name": "Bulbasaur",
    "type": [
      "Grass",
      "Poison"
    ],
    "hp": 45,
    "attack": 49,
    "defense": 49,
    "special_attack": 65,
    "special_defense": 65,
    "speed": 45
  },
  {
    "name": "Ivysaur",
    "type": [
      "Grass",
      "Poison"
    ],
    "hp": 60,
    "attack": 62,
    "defense": 63,
    "special_attack": 80,
    "special_defense": 80,
    "speed": 60
  }
]
```

5.2 Sample query 2 - Find all pokemons with type 'Grass' and 'Poison'

Make sure you only fetch the names of the pokemons

Expected output:

[{"name": "Bulbasaur"}, {"name": "Ivysaur"}, {"name": "Venusaur"}, {"name": "Oddish"}, {"name": "Gloom"}, {"name": "Vileplume"}, {"name": "Bellsprout"}, {"name": "Weepinbell"}, {"name": "Victreebel"}, {"name": "Roselia"}, {"name": "Budew"}, {"name": "Roserade"}, {"name": "Foongus"}, {"name": "Amoonguss"}]

```
result = db.pokemons.find({'type': {"$all": ['Grass', 'Poison']}}, {'name': 1, '_id': 0})
```

```
poke = json_util.dumps(result)
```

```
print(poke)
```

```
[{"name": "Bulbasaur"}, {"name": "Ivysaur"}, {"name": "Venusaur"}, {"name": "Oddish"}, {"name": "Gloom"}, {"name": "Vileplume"}, {"name
```

Homework starts from here

Query 1 - 10pts

Find the Pokemon with the second highest "special attack":

Expected answer:

{ "id": 150, "name": "Mewtwo", "type": ["Psychic"], "hp": 106, "attack": 110, "defense": 90, "special_attack": 154, "special_defense": 90, "speed": 130
}

Your answer below -

```
result = db.pokemons.find({}, {"_id": 0}).sort([("special_attack", -1)]).skip(1).limit(1)
```

```
print(json_util.dumps(result, indent=2))
```

```
[
  {
    "id": 150,
    "name": "Mewtwo",
    "type": [
      "Psychic"
    ],
    "hp": 106,
    "attack": 110,
    "defense": 90,
    "special_attack": 154,
    "special_defense": 90,
    "speed": 130
  }
]
```

Query 2 - 15pts

Find the top 5 Pokemon with the highest total power, along with their types:

What is total power of a pokemon?

```
Total power is the summation of the pokemon's attack power, defense power, special attack power, special defense power plus it's speed.
```

Create a new attribute "total_power" using the above description of total power and add the new attribute to the set of existing fields.

As we need top 5 pokemons, you need to sort them by total power as well.

Hints: Some mongodb specific fields that you might need for this query ->

"addFields" "sort" "limit" "project"

Google them, see how they work in an aggregate query. You have previously done things similar to this previously in one of our Mongodb labs, refer that as well.

Expected output:

[{ "name": "Arceus", "total_power": 600 }, { "name": "Palkia", "total_power": 590 }, { "name": "Zekrom", "total_power": 580 }, { "name": "Reshiram", "total_power": 580 }, { "name": "Dialga", "total_power": 580 }]

Your answer below -

```
result = db.pokemons.aggregate(
    [
    {
        '$addFields': {
            'total_power': {
                '$sum': [
                    '$attack', '$defense', '$special_attack', '$special_defense', '$speed'
                ]
            }
        }
    }, {
        '$sort': {
            'total_power': -1
        }
    }, {
        '$limit': 5
    },
    {
      "$project": {
          "_id": 0,
          "name": 1,
          "total_power": 1
      }
    }
]
)
```

```
print(json_util.dumps(result, indent=2))
```

```
[
  {
    "name": "Arceus",
    "total_power": 600
  },
  {
    "name": "Palkia",
    "total_power": 590
  },
  {
    "name": "Zekrom",
    "total_power": 580
  },
  {
    "name": "Reshiram",
    "total_power": 580
  },
  {
    "name": "Dialga",
    "total_power": 580
  }
]
```

Query 3 - 15pts

Find the average HP of each type of Pokemon, sorted by the average HP in descending order:

Hints: Checkout "unwind" and "group" in mongodb

Expected Output:

[{ "_id": "Dragon", "avg_hp": 79.4 }, { "_id": "Ice", "avg_hp": 77.20588235294117 }, { "_id": "Normal", "avg_hp": 76.05504587155963 }, { "_id": "Fighting", "avg_hp": 75.25925925925925 }, { "_id": "Ground", "avg_hp": 74.5625 }, { "_id": "Dark", "avg_hp": 73.15217391304348 }, { "_id": "Psychic", "avg_hp": 72.52439024390245 }, { "_id": "Flying", "avg_hp": 70.23469387755102 }, { "_id": "Water", "avg_hp": 68.95419847328245 }, { "_id": "Fire",

"avg_hp": 68.328125 }, { "_id": "Steel", "avg_hp": 67.91836734693878 }, { "_id": "Rock", "avg_hp": 66.81666666666666 }, { "_id": "Fairy", "avg_hp": 66.7872340425532 }, { "_id": "Grass", "avg_hp": 65.38144329896907 }, { "_id": "Ghost", "avg_hp": 64.81395348837209 }, { "_id": "Electric", "avg_hp": 64.8125 }, { "_id": "Poison", "avg_hp": 62.74242424242424 }, { "_id": "Bug", "avg_hp": 56.493506493506494 }]

Your answer below -

```
result = db.pokemons.aggregate(
    [
      {
        "$unwind": "$type"
      },
      {
        "$group": {
          "_id": "$type",
          "avg_hp": { "$avg": "$hp" }
        }
      },{
        "$project": {
          "_id": 1,
          "avg_hp": 1
        }
      },
     {
         "$sort": {"avg_hp": -1}
     }
    ]
)
```

```
print(json_util.dumps(result, indent=2))
```


```
        "_id": "Fighting",
        "avg_hp": 75.25925925925925
      },
      {
        "_id": "Ground",
        "avg_hp": 74.5625
      },
      {
        "_id": "Dark",
        "avg_hp": 73.15217391304348
      },
      {
        "_id": "Psychic",
        "avg_hp": 72.52439024390245
      },
      {
        "_id": "Flying",
        "avg_hp": 70.23469387755102
      },
      {
        "_id": "Water",
        "avg_hp": 68.95419847328245
      },
      {
        "_id": "Fire",
        "avg_hp": 68.328125
      },
      {
        "_id": "Steel",
        "avg_hp": 67.91836734693878
      },
      {
        "_id": "Rock",
        "avg_hp": 66.81666666666666
```

```
    "avg_hp": 64.8125
  },
  {
    "_id": "Poison",
    "avg_hp": 62.74242424242424
  },
  {
    "_id": "Bug",
    "avg_hp": 56.493506493506494
  }
```


Query 4 - 10pts

Find the average attack power and special attack power of each type of Pokemon, sorted by the average special attack power in descending order:

Hints: Checkout "unwind","group" and "avg" in mongodb

Expected Output:

[{ "_id": "Fire", "avg_attack": 80.5, "avg_special_attack": 87.578125 }, { "_id": "Psychic", "avg_attack": 66.21951219512195, "avg_special_attack": 86.98780487804878 }, { "_id": "Dragon", "avg_attack": 94.66666666666667, "avg_special_attack": 85.6 }, { "_id": "Electric", "avg_attack": 71.70833333333333, "avg_special_attack": 85.16666666666667 }, { "_id": "Ghost", "avg_attack": 76.95348837209302, "avg_special_attack": 79.62790697674419 }, { "_id": "Fairy", "avg_attack": 60.40425531914894, "avg_special_attack": 76.34042553191489 }, { "_id": "Ice", "avg_attack": 78.32352941176471, "avg_special_attack": 75.67647058823529 }, { "_id": "Dark", "avg_attack": 91.47826086956522, "avg_special_attack": 72.65217391304348 }, { "_id": "Flying", "avg_attack": 76.04081632653062, "avg_special_attack": 71.36734693877551 }, { "_id": "Steel", "avg_attack": 91.06122448979592, "avg_special_attack": 70.83673469387755 }, { "_id": "Grass", "avg_attack": 73.29896907216495, "avg_special_attack": 70.65979381443299 }, { "_id": "Water", "avg_attack": 70.38167938931298, "avg_special_attack": 70.58778625954199 }, { "_id": "Poison", "avg_attack": 68.28787878787878, "avg_special_attack": 68.28787878787878 }, { "_id": "Fighting", "avg_attack": 99.74074074074075, "avg_special_attack": 62.351851851851855 }, { "_id": "Rock", "avg_attack": 87.28333333333333, "avg_special_attack": 58.55 }, { "_id": "Normal", "avg_attack": 73.3211009174312, "avg_special_attack": 57.53211009174312 }, { "_id": "Ground", "avg_attack": 87.734375, "avg_special_attack": 56.90625 }, { "_id": "Bug", "avg_attack": 68.16883116883118, "avg_special_attack": 56.22077922077922 }]


```
result = db.pokemons.aggregate(
    [
      {
        "$unwind": "$type"
      },
      {
        "$group": {
          "_id": "$type",
          "avg_attack": { "$avg": "$attack" },
          "avg_special_attack": { "$avg": "$special_attack" }
        }
      },
     {
        "$project": {
          "_id": 1,
          "avg_special_attack": 1,
           "avg_attack": 1
        }
      },
     {
         "$sort": {"avg_special_attack": -1}
     }
    ]
)
```

```
print(json_util.dumps(result, indent=2))
```


```
[
  {
    "_id": "Fire",
    "avg_attack": 80.5,
    "avg_special_attack": 87.578125
  },
  {
    "_id": "Psychic",
    "avg_attack": 66.21951219512195,
    "avg_special_attack": 86.98780487804878
  },
```

```
  {
    "_id": "Dragon",
    "avg_attack": 94.66666666666667,
    "avg_special_attack": 85.6
  },
  {
    "_id": "Electric",
    "avg_attack": 71.70833333333333,
    "avg_special_attack": 85.16666666666667
  },
  {
    "_id": "Ghost",
    "avg_attack": 76.95348837209302,
    "avg_special_attack": 79.62790697674419
  },
  {
    "_id": "Fairy",
    "avg_attack": 60.40425531914894,
    "avg_special_attack": 76.34042553191489
  },
  {
    "_id": "Ice",
    "avg_attack": 78.32352941176471,
    "avg_special_attack": 75.67647058823529
  },
  {
    "_id": "Dark",
    "avg_attack": 91.47826086956522,
    "avg_special_attack": 72.65217391304348
  },
  {
    "_id": "Flying",
    "avg_attack": 76.04081632653062,
    "avg_special_attack": 71.36734693877551
  },
  {
    "_id": "Steel",
    "avg_attack": 91.06122448979592,
    "avg_special_attack": 70.83673469387755
  },
  {
    "_id": "Grass",
    "avg_attack": 73.29896907216495,
    "avg_special_attack": 70.65979381443299
  },
  {
    "_id": "Water"
```


Query 5 - 10pts

Find the types of Pokemon that are weak against a certain type (i.e. have a type in common with the given type, but are not that type themselves):

So if you input the type_to_find as "Fire", it should return all other types except Fire.

Hints: Checkout "match", "project, "in", "ne", "group" in mongodb

Example input: "Water"

Expected Output:

[{ "_id": "Dark" }, { "_id": "Flying" }, { "_id": "Fire" }, { "_id": "Ice" }, { "_id": "Fighting" }, { "_id": "Poison" }, { "_id": "Electric" }, { "_id": "Rock" }, { "_id": "Psychic" }, { "_id": "Bug" }, { "_id": "Fairy" }, { "_id": "Grass" }, { "_id": "Dragon" }, { "_id": "Ground" }, { "_id": "Normal" }, { "_id": "Steel" }, { "_id": "Ghost" }]

```
type_to_find = 'Water'
result = db.pokemons.aggregate(
    [
      { "$project": { "weak": { "$filter": { "input": "$type", "cond": { "$ne": [ "$$this", type_to_find ] } } } } },
      { "$unwind": "$weak" },
      { "$group": { "_id": "$weak"} }
    ]
)
```

```
print(json_util.dumps(result, indent=2))
```


```
  [
    {
      "_id": "Fire"
```

```
    },
    {
      "_id": "Psychic"
    },
    {
      "_id": "Flying"
    },
    {
      "_id": "Fairy"
    },
    {
      "_id": "Ice"
    },
    {
      "_id": "Dragon"
    },
    {
      "_id": "Ghost"
    },
    {
      "_id": "Poison"
    },
    {
      "_id": "Fighting"
    },
    {
      "_id": "Electric"
    },
    {
      "_id": "Rock"
    },
    {
      "_id": "Bug"
    },
    {
      "_id": "Dark"
    },
    {
      "_id": "Grass"
    },
    {
      "_id": "Normal"
    },
    {
      "_id": "Ground"
    },
    {
      "_id": "Steel"
    }
  ]
```


Query 6 - 10pts

Find all the pokemons that would overpower a given fire type pokemon based on their types:

So for example, A Fire type pokemon is generally weak against these types: Water, Rock and Ground.

Charizard is a fire type pokemon. Your task is to printout all the pokemons that Charizard is weak against.

Example input: "Charizard"

Expected Output:

[{ "name": "Squirtle", "type": ["Water"] }, { "name": "Wartortle", "type": ["Water"] }, { "name": "Blastoise", "type": ["Water"] },

.....

.....

{ "name": "Nihilego", "type": ["Rock", "Poison"] }, { "name": "Stakataka", "type": ["Rock", "Steel"] }]

```
fire_pokemon_name = "Charizard"

overpower_pokemons = db.pokemons.aggregate([
    {
        '$match': {
            'name': fire_pokemon_name
        }
    }, {
```

```
        '$project': {
            'type': 1,
            '_id': 0
        }
    },
    {"$unwind": "$type"},
    {"$lookup": {"from": "pokemons",
                 "let": {"weak": "$type"},
                 "pipeline": [
                        {
                          "$match": {
                            "$expr": {
                              "$not": {"$in": ["$$weak", "$type"]},
                            },
                          },
                        },
                      ],
                 "as": "weak_pokemons"}},
    {"$unwind": "$weak_pokemons"},
    {"$project": {"name": "$weak_pokemons.name", "type": "$weak_pokemons.type"}}
])
```

```
print(json_util.dumps(overpower_pokemons, indent=2))
```


```
  "name": "Ivysaur",
  "type": [
    "Grass",
    "Poison"
  ]
},
{
  "name": "Venusaur",
  "type": [
    "Grass",
    "Poison"
  ]
},
{
  "name": "Squirtle",
  "type": [
    "Water"
  ]
},
{
  "name": "Wartortle",
  "type": [
    "Water"
  ]
},
{
  "name": "Blastoise",
  "type": [
    "Water"
  ]
},
{
  "name": "Caterpie",
  "type": [
    "Bug"
  ]
},
{
  "name": "Metapod",
  "type": [
    "Bug"
  ]
},
{
  "name": "Butterfree",
  "type": [
    "Bug",
    "Flying"
```

```
1
```

✓ 0s completed at 1:45 PM