بخش اول

پیکر بندی

در این بخش به پیکربندی MongoDB می‌پردازیم. با توجه به قدیمی بودن نسخه MongoDB موجود در repository رسمی Ubuntu 18.04، لازم است تا آخرین نسخه MongoDB از مخزن رسمی MongoDB دریافت گردد. بدین منظور لازم است تا دستورات زیر در bash اجرا شود:

```
$ wget -qO - https://www.mongodb.org/static/pgp/server-4.2.asc | sudo apt-key add -
$ echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.2 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.2.list
$ sudo apt update
$ sudo apt-get install -y mongodb-org
$ cd ~
$ sudo mkdir -p data/db
$ sudo mongod --dbpath ~/data/db
```

پس از انجام این مراحل، آخرین نسخه از MongoDB نصب شده و دیتای آن نیز در دایرکتوری تعریف شده در مسیر کاربر فعلی ذخیره خواهد شد.

گام اول

پس از پیکربندی MongoDB، ابتدا اطلاعات چند کاربر محدود از API سایت randomuser دریافت شده است. برای وارد کردن این اطلاعات به صورت دستی در دیتابیس، از دستورات mongo shell کمک می‌گیریم. خط فرمان mongo shell با وارد کردن دستور زیر باز خواهد شد:

```
$ mongo
```

که پس از آن می‌توان دستورات را خط به خط درون آن وارد کرد. اما این امکان نیز وجود دارد که دستورات mongo shell را در یک فایل اسکریپت درج کرده و همگی را به صورت یکجا اجرا کنیم. این کار می‌تواند توسط اجرای دستور زیر صورت گیرد:

```
$ mongo < step1.js
```

که فایل step1.js شامل دستورات mongo shell بوده و همگی در این مرحله توسط mongo shell اجرا می‌گردند. در این اسکریپ با استفاده از تابع insertOne یک document به یک collection اضافه شده است. همچنین با استفاده از تابع count تعداد documentهای موجود در یک collection قابل بررسی است.

```
use bigdata99
print("Deleting all data in tempusers...")
db.tempusers.deleteMany({})
print("Adding 3 sample data to tempusers...")
db.tempusers.insertOne({
      …
})
db.tempusers.insertOne({
      …
})
db.tempusers.insertOne({
      …
})
print("Count of tempusers: ")
db.tempusers.count()
print("One record of tempusers: ")
db.tempusers.findOne()
```

فایل **step1.js**

در مرحله بعد این فرآیند را با استفاده از پایتون و کتابخانه رسمی pymongo انجام می‌دهیم. در این مرحله با اتصال به درگاه randomuser در هر مرحله اقدام به دریافت اطلاعات ۵ هزار کاربر خواهیم کرد.

اسکریپت مربوط به این بخش در فایل step1.py موجود است. این اسکریپت در هر بار اجرا تا زمانی که درگاه randomuser اطلاعات کاربران را بدست بدهد، اقدام به دریافت داده‌ها در دسته‌های ۵ هزارتایی کرده، سپس آن‌ها را در دیتابیس MongoDB وارد کرده و در ادامه بررسی می‌کند که تا این لحظه چند کاربر در دیتابیس موجود می‌باشند. در صورتی که این تعداد از ۱۰۰ هزار کمتر باشد، مراحل را از ابتدا دنبال می‌کند. در صورتی که این تعداد به ۱۰۰ هزار رسید، کار دریافت دیتا را متوقف کرده و موفقیت آمیز بودن فرآیند را اعلام می‌کند.

با توجه به اینکه پس از تعداد محدود دریافت دسته‌های ۵ هزارتایی کاربر از randomuser، به مدت چند دقیقه پاسخگویی از سمت این API صورت نمی‌گیرد، برای تسریع در فرآیند دریافت دیتا در صورت از بین رفتن اطلاعات داخل دیتابیس، اسکریپت به صورت خودکار ۱۰۰ هزار داده دیتابیس را در صورت کامل بودن تعداد آن‌ها در یک فایل tar.gz. که فشرده شده فایل json داده‌های دریافت شده است به عنوان Backup ذخیره خواهد کرد. این اطلاعات در زمان لازم توسط فایل import.py می‌توانند مجدد به دیتابیس MongoDB وارد شوند.

```
#!/usr/bin/python3
import requests
import json
import time
from pymongo import MongoClient
import tarfile
import os
client = MongoClient()
db = client['bigdata99']
url = 'https://randomuser.me/api/'
api_sleep = 10
api_limit = 5000
total = 100000
#db.users.remove({})
fetched = db.users.count()
while fetched < total:
    response = requests.request('GET', url, params={'nat':'ir',
    'results':api_limit})
    if response.status_code == requests.codes.ok:
       data = response.json()['results']
       try:
              db.users.insert_many(data)
       except Exception as e:
              print("Mongo insert exception: "+str(e))
       fetched = db.users.count()
    else:
       print("Response code error: "+str(response.status_code))
       print("Trying again")
    print(f'Fetched and inserted {fetched} users so far')
    print(f'Waiting for {api_sleep} seconds...')
    time.sleep(api_sleep)
fetched = db.users.count()
print(f'Successfully fetched and inserted {fetched} users into the db.')
with open('users.json', 'w') as file:
    file.write('[')
    for document in db.users.find({}, {'_id': False}):
       file.write(json.dumps(document))
       file.write(',')
    file.write(']')
with tarfile.open('users.tar.gz','w:gz') as tar:
    tar.add('users.json')
os.remove('users.json')
```

فایل **step1.py**

در طی این فرآیند، با استفاده از تابع insertMany از MongoDB امکان وارد کردن دیتای چند کاربر به دیتابیس ممکن خواهد بود. همچنین صدا زدن تابع count روی یک collection تعداد documentهای موجود در آن دیتابیس را بدست می‌دهد و ازین طریق می‌توان اطمینان حاصل کرد که رکوردها در دیتابیس ذخیره شده اند یا خیر.

گام دوم

۱

فایل کد: step2-1.py

```
#!/usr/bin/python3

import pprint
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']
cursor_limit = 10

users = db.users.find({ "dob.age": { "$gt": 50 }, "location.city":
    "نیشابور" }, {"name.first": True, "name.last": True, "dob.age":
    True, "location.city": True, "_id": False}).limit(cursor_limit)
print(f'Found {users.count()} results. Here are {cursor_limit}
    samples:')
for user in users:
    pprint.pprint(user)
```

زمان اجرا: ۰,۶۶۱ ثانیه

تعداد خروجی: ۹۲۳ سند

نمونه خروجی:

```
Found 923 results. Here are 10 samples:
{'dob': {'age': 67},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'مریم', 'last': 'قاسمی'}}
{'dob': {'age': 65},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'آتنا', 'last': 'یزدان'}}
{'dob': {'age': 73},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'مریم', 'last': 'احمدی'}}
{'dob': {'age': 58},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'هادی', 'last': 'سهیلی دار'}}
{'dob': {'age': 63},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'فاطمه', 'last': 'کیمیای'}}
{'dob': {'age': 58},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'علیرضا', 'last': 'پاسارگاد'}}
{'dob': {'age': 52},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'مریم', 'last': 'رضایی'}}
{'dob': {'age': 67},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'سارینا', 'last': 'یزدی'}}
{'dob': {'age': 75},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'رضا', 'last': 'کرامی'}}
{'dob': {'age': 60},
 'location': {'city': 'نیشابور'},
 'name': {'first': 'علی', 'last': 'مرادخانی'}}
```

۲

فایل کد: step2-2.py

```
#!/usr/bin/python3

import pprint
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']
cursor_limit = 10

users = db.users.find({ "registered.age": { "$gt": 20 } },
    {"name.last": True, "location": True, "phone": True,
    "registered.age": True, "_id": False}).limit(cursor_limit)
print(f'Found {users.count()} results. Here are {cursor_limit}
    samples:')
for user in users:
    pprint.pprint(user)
```

زمان اجرا: ۰,۶۴۹ ثانیه

تعداد خروجی: ۰ سند

نمونه خروجی:

```
Found 0 results. Here are 10 samples:
```

۳

فایل کد: step2-3.py

```
#!/usr/bin/python3

import pprint
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']
cursor_limit = 10

db.users.update_many({},[{
    "$set": {
       "dob.year_persian": {
              "$cond": [
                     {
                            "$and": [
                                   { "$gte": [{ "$month": {"$dateFromString":
    {"dateString": "$dob.date"}} }, 3] },
                                   { "$gte": [{ "$dayOfMonth":
    {"$dateFromString": {"dateString": "$dob.date"}} }, 20] },
                            ]
                     },
                     {"$subtract": [{ "$year": {"$dateFromString":
    {"dateString": "$dob.date"}} }, 621]},
                     {"$subtract": [{ "$year": {"$dateFromString":
    {"dateString": "$dob.date"}} }, 622]}
              ]
       },
       "registered.year_persian": {
              "$cond": [
                     {
                            "$and": [
                                   { "$gte": [{ "$month": {"$dateFromString":
    {"dateString": "$registered.date"}} }, 3] },
                                   { "$gte": [{ "$dayOfMonth":
    {"$dateFromString": {"dateString": "$registered.date"}} }, 20] },
                            ]
                     },
                     {"$subtract": [{ "$year": {"$dateFromString":
    {"dateString": "$registered.date"}} }, 621]},
                     {"$subtract": [{ "$year": {"$dateFromString":
    {"dateString": "$registered.date"}} }, 622]}
              ]
       },
    }
}])


users = db.users.find().limit(cursor_limit)
print(f'Found {users.count()} results. Here are {cursor_limit} samples:')
for user in users:
    pprint.pprint(user)
```

زمان اجرا: ۷٫۶۷۵ ثانیه

نمونه خروجی:


```
{'_id': ObjectId('5ebbe8afcb04812d3f977cd3'),
 'cell': '0972-688-2403',
 'dob': {'age': 46, 'date': '1974-06-08T08:05:47.052Z', 'year_persian': 1352},
 'email': 'ln.njty@example.com',
 'gender': 'female',
 'id': {'name': '', 'value': None},
 'location': {'city': 'دزی',
              'coordinates': {'latitude': '-3.1461', 'longitude': '48.0007'},
              'country': 'Iran',
              'postcode': 74197,
              'state': 'ن انفسلگ',
              'street': {'name': 'ی بونج دربن', 'number': 9822},
              'timezone': {'description': 'Tehran', 'offset': '+3:30'}},
 'login': {'md5': '8bf12f6f160202bf61fe73c8a36e7a99',
           'password': 'april1',
           'salt': 'WOFBjs56',
           'sha1': 'b1a1e71e1c1911810372cc78faf06333df702229',
           'sha256': '48428de72f6706bc6d08282586f5bb8546a0a57c0108ec19d6f4032d1e6bef17',
           'username': 'smallkoala365',
           'uuid': '2b69fdfe-bca8-4724-ae1c-64812b32712e'},
 'name': {'first': 'انل ا', 'last': 'ی تاجن', 'title': 'Mrs'},
 'nat': 'IR',
 'phone': '072-14019832',
 'picture': {'large': 'https://randomuser.me/api/portraits/women/13.jpg',
             'medium': 'https://randomuser.me/api/portraits/med/women/13.jpg',
             'thumbnail': 'https://randomuser.me/api/portraits/thumb/women/13.jpg'},
 'registered': {'age': 17,
                'date': '2003-04-27T05:27:41.444Z',
                'year_persian': 1382}}
```


توضیحات:

در این بخش با الحاق aggregation pipeline در دستور updateMany اقدام به بررسی تاریخ‌های تولید و ثبت نام کاربران گردید. الگوریتم ایجاد سال شمسی از تاریخ میلادی به این صورت اعمال شده است که اگر تاریخ پس از روز ۲۰ ام ماه ۳ میلادی بود، از سال میلادی عدد ۶۲۱، و الا عدد ۶۲۲ کم گردد.

۴

فایل کد: step2-4.py

```
#!/usr/bin/python3

import pprint
import itertools
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']
limit = 3

users = db.users.aggregate([
    { "$match":
       { "$expr":
              { "$and": [
                     { "$eq": [
                            { "$month": {"$dateFromString": {"dateString":
    "$dob.date"}} },
                            { "$month": "$$NOW" }
                     ]},
                     { "$eq": [
                            { "$dayOfMonth": {"$dateFromString": {"dateString":
    "$dob.date"}} },
                            { "$dayOfMonth": "$$NOW" }
                     ]}
              ]}
       }
    },
    {
       "$project": {
              "name.first": True,
              "name.last": True,
              "email": True,
              "dob.date": True,
              "_id": False
       }
    }
])


users = list(users)
print(f'Found {len(users)} results. Here are {limit} samples:')
for user in users[:limit]:
    pprint.pprint(user)c
```

زمان اجرا: ۱,۶۵۳ ثانیه

تعداد خروجی: ۲۵۴ سند (تاریخ ۲۰۲۰/۵/۱۹)

نمونه خروجی:

```
Found 254 results. Here are 3 samples:
{'dob': {'date': '1987-05-19T04:29:25.272Z'},
 'email': 'pwry.rdyy@example.com',
 'name': {'first': 'ای‌روپ', 'last': 'ی‌اضر'}}
{'dob': {'date': '1946-05-19T01:18:20.052Z'},
 'email': 'ard.sdr@example.com',
 'name': {'first': 'دارآ', 'last': 'ردص'}}
{'dob': {'date': '1998-05-19T00:50:32.068Z'},
 'email': 'aasl.khmrw@example.com',
 'name': {'first': 'لسع', 'last': 'اورم‌اک'}}
```

۵

فایل کد: step2-5.py

```
#!/usr/bin/python3

import pprint
import itertools
from hashlib import sha256
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']
limit = 3

user_name = "smallkoala365"
password = "april1"

users = db.users.find({"login.username": user_name},{"login.password":
    False})
found = False
for user in users:
    salt = user["login"]["salt"]
    if user["login"]["sha256"] == sha256((password+salt).encode('utf-
    8')).hexdigest():
       print("Matching username and Password has found:")
       pprint.pprint(user)
       found = True
if found == False:
    print("No matching username and password has found.")
```

زمان اجرا: ۰,۶۵۶ ثانیه

تعداد خروجی: ۱ سند (نام کاربری smallkoala365 و رمزعبور april1)

نمونه خروجی:

```
Matching username and Password has found:
{'_id': ObjectId('5ebbe8afcb04812d3f977cd3'),
 'cell': '0972-688-2403',
 'dob': {'age': 46, 'date': '1974-06-08T08:05:47.052Z', 'year_persian': 1352},
 'email': 'ln.njty@example.com',
 'gender': 'female',
 'id': {'name': '', 'value': None},
 'location': {'city': 'دزی',
              'coordinates': {'latitude': '-3.1461', 'longitude': '48.0007'},
              'country': 'Iran',
              'postcode': 74197,
              'state': 'ن اتصلگ',
              'street': {'name': 'یبونج دربن', 'number': 9822},
              'timezone': {'description': 'Tehran', 'offset': '+3:30'}},
 'login': {'md5': '8bf12f6f160202bf61fe73c8a36e7a99',
           'salt': 'WOFBjs56',
           'sha1': 'b1a1e71e1c1911810372cc78faf06333df702229',
           'sha256': '48428de72f6706bc6d08282586f5bb8546a0a57c0108ec19d6f4032d1e6bef17',
           'username': 'smallkoala365',
           'uuid': '2b69fdfe-bca8-4724-ae1c-64812b32712e'},
 'name': {'first': 'انا', 'last': 'یتاچن', 'title': 'Mrs'},
 'nat': 'IR',
 'phone': '072-14019832',
 'picture': {'large': 'https://randomuser.me/api/portraits/women/13.jpg',
             'medium': 'https://randomuser.me/api/portraits/med/women/13.jpg',
             'thumbnail': 'https://randomuser.me/api/portraits/thumb/women/13.jpg'},
 'registered': {'age': 17,
                'date': '2003-04-27T05:27:41.444Z',
                'year_persian': 1382}}
```

توضیحات:

همانطور که در صورت سؤال نیز ذکر شده است، ذخیره رمز عبور در دیتابیس به صورت خام کاری اشتباه است، زیرا که اگر یک حمله کننده به هر نحوی به اطلاعات دیتابیس دسترسی پیدا کند، رمز عبور تمامی کاربران افشاء خواهد شد. برای رفع این مشکل می‌توان به جای خود رمزعبور، تابع Hash را بر آن‌ها اعمال کرده و سپس نتیجه را در دیتابیس ذخیره نمود. تابع Hash خصوصیتی که دارد، یکطرفه بودن آن و عدم امکان یافتن معکوس آن می‌باشد. همچنین تابع Hash ویژگی‌های دیگری نظیر collision resistance و غیره را نیز دارد که توضیح آن‌ها در مجال این گزارش نیست.

توابع Hash مختلفی نظیر SHA1، SHA256 و یا MD4 وجود دارد که می‌توان از آن‌ها بهره جست. امن ترین این توابع SHA256 می‌باشد.

نکته دیگر در این مرحله اینست که اعمال تابع Hash به تنهایی نیز ممکن است امنیت را تأمین نکند، زیرا که حمله کننده می‌تواند از پیش یک دیتاست شامل کلمه عبورهای مختلف و Hash آن‌ها را تدارک دیده باشد و پس از دسترسی به دیتابیس، با یک عملیات Lookup ساده در جدول خود (Rainbow Table) به یافتن مقادیر معکوس Hash اقدام کند. برای جلوگیری از این کار، به ازای هر کاربر یک مقدار تصادفی با نام Salt به ابتدا و یا انتهای رمز عبور کاربر Concate شده و سپس از تمامی آن Hash گرفته می‌شود. همچنین مقدار Salt به صورت خام نیز در کنار Hash گرفته شده در دیتابیس ذخیره می‌گردد. حال در صورتی که یک حمله کننده به دیتابیس دسترسی پیدا کند، با توجه به اینکه رمز عبور هر کاربر با یک Salt نیز الحاق شده است، به ازای هر کاربر می‌بایست یک Rainbow Table تشکیل داده که این مدت زمان انجام عملیات را غیر قابل دسترس می‌نماید.

در دیتاست فعلی مقادیر SHA256 و یک Salt تصادفی ذخیره شده است. با بررسی این دیتاست، مشخص می‌شود که مقدار Hash ذخیره شده برای هر کاربر برابر Concate مقدار Salt به انتهای رمز عبور کاربر بوده که تابع Hash بر روی تمام آن اعمال شده است.

نکته قابل توجه اینست که متأسفانه مقادیر نام کاربری موجود برای کاربران یکتا نیست(!) و با یک مقدار نام کاربری می‌توان چندین کاربر را یافت که این مسئله غیر استاندارد بوده و امکان بررسی ورود یک کاربر را وابسته به مقدار رمز عبور وی نیز می‌نماید. با این حال کد مربوط به این بخش با عنایت به این مسئله طراحی شده است.

در فرآیند کار، می‌خواهیم یک کاربر با نام کاربری مشخص و رمز عبور مشخص را پیدا کنیم. ابتدا از دیتابیس تمامی کاربران با رمز عبور مشخص را می‌یابیم. سپس به ازای هر کاربر، مقدار Salt وی را به رمز

عبور مشخص Concate کرده و از مجموع آن‌ها مقدار SHA256 را می‌یابیم. سپس بررسی می‌کنیم که مقدار SHA256 ذخیره شده برای هر کاربر (که با فرمت HEX بوده) آیا با مقدار بدست آمده از SHA256 حاصل از Concate بدست آمده یکی هست یا خیر. اگر یکی بود، کاربر را به عنوان نتیجه اعلام می‌کنیم.

لازم به ذکر است که در این مرحله با توجه به وجود مقادیر Salt و SHA256 به ازای هر کاربر، نیازی به آپدیت دیتا درون دیتابیس نبوده و از قبل اینکار صورت گرفته است.

گام سوم

۱

فایل کد: step3-1.py

```
#!/usr/bin/python3

import pprint
import itertools
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']

results = db.users.aggregate([
    {
       "$project": {
              "ageGroup": {
                     "$cond": [
                            { "$lte": ["$dob.age", 16] },
                            "young" ,
                            {
                                   "$cond": [
                                          { "$lte": ["$dob.age", 30] },
                                          "adult",
                                          "old"
                                   ]
                            }
                     ]
              },
              "dob.age": True
       }
    },
    {
       "$group": {
              "_id": "$ageGroup",
              "count": {
                     "$sum": 1
              }
       }
    }
])


results = list(results)
for result in results:
    pprint.pprint(result)
```

زمان اجرا: ۰,۶۵۱ ثانیه

نمونه خروجی:

```
{'_id': 'adult', 'count': 16124}
{'_id': 'old', 'count': 83876}
```

توضیحات:

در دیتای دریافت شده، هیچ رکوردی دارای سن زیر ۱۶ سال نبوده است.

۲

فایل کد: step3-2.py

```
#!/usr/bin/python3

import pprint
import itertools
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']

results = db.users.aggregate([
    {
       "$group": {
              "_id": "$location.state",
              "count": {
                     "$sum": 1
              }
       }
    }
])


results = list(results)
for result in results:
    pprint.pprint(result)
```

زمان اجرا: ۰٫۶۵۴ ثانیه

نمونه خروجی:

```
{'_id': 'ن ان م س', 'count': 3291}
{'_id': 'ی وض ر ن ا س ا ر خ', 'count': 3304}
{'_id': 'ن ا ر د ن ز ا م', 'count': 3212}
{'_id': 'ق م', 'count': 3181}
{'_id': 'ن ا ل ی گ', 'count': 3135}
{'_id': 'ی ب ون چ ن ا س ا ر خ', 'count': 3343}
{'_id': 'ن ا ت س د ر ک', 'count': 3342}
{'_id': 'ی ر ا ی ت خ ب و ل ا ج م ر ا ه چ', 'count': 3214}
{'_id': 'س ر ا ف', 'count': 3200}
{'_id': 'ا ر د ب ی ل', 'count': 3202}
{'_id': 'ی ب ر غ ن ا چ ی ا ب ر د ز آ', 'count': 3225}
{'_id': 'ن ا ت س ل گ', 'count': 3149}
{'_id': 'ن ا ت س چ و ل ب و ن ا ت س ی س', 'count': 3261}
{'_id': 'ق ی ن ی و ز', 'count': 3262}
{'_id': 'د ز ی', 'count': 3278}
{'_id': 'ن ا ه ف ص ا', 'count': 3173}
{'_id': 'ن ا ت س ر ل', 'count': 3220}
{'_id': 'ی ل ا م ش ن ا س ا ر خ', 'count': 3199}
{'_id': 'ن ا گ ز م ر ه', 'count': 3262}
{'_id': 'ق ی ش ر ن ا چ ی ا ب ر د ز آ', 'count': 3198}
{'_id': 'د م ج ا ر ی و ب و ه ی و ل ی گ ه ک', 'count': 3175}
{'_id': 'ر ه ش و ب', 'count': 3230}
{'_id': 'ی ز ک ر م', 'count': 3249}
{'_id': 'ن ا ر ه ت', 'count': 3337}
{'_id': 'ن ا ت س ز و خ', 'count': 3230}
{'_id': 'ا ل ی م', 'count': 3218}
{'_id': 'ه ا ش ن ا م ر ک', 'count': 3140}
{'_id': 'ن ا د م ه', 'count': 3201}
{'_id': 'ز ر ی ل ا', 'count': 3121}
{'_id': 'ن ا م ر ک', 'count': 3242}
{'_id': 'ن ا چ ن ز', 'count': 3206}
```

۳

فایل کد: step3-3.py

```
#!/usr/bin/python3

import pprint
import itertools
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']

db.users.update_many(
    { "location.timezone.offset": "+5:00" },
    { "$unset": {"cell": ""} }
)

results = db.users.aggregate([
    {
       "$group": {
              "_id": None,
              "count": {
                     "$sum": {
                            "$cond": [{"$ifNull": ["$cell", False]}, 0, 1]
                     }
              }
       }
    }
])

results = list(results)

for result in results:
    pprint.pprint(result)
```

زمان اجرا: ۰,۶۵۶ ثانیه

نمونه خروجی:

```
{'_id': None, 'count': 3490}
```

۴

فایل کد: step3-4.py

```
#!/usr/bin/python3

import pprint
import itertools
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']

tehran_average = list(db.users.aggregate([
    {
       "$match": { "location.state": "تهران"}
    },
    {
       "$group": {
              "_id": None,
              "average": {
                     "$avg": "$dob.age"
              }
       }
    }
]))[0]["average"]

results = db.users.aggregate([
    {
       "$group": {
              "_id": "$location.state",
              "average": {
                     "$avg": "$dob.age"
              }
       }
    },
    {
       "$project": {
              "_id": True,
              "average": True,
              "difference": {"$subtract": ["$average", tehran_average]}
       }
    }
])


results = list(results)
for result in results:
    pprint.pprint(result)
```

زمان اجرا: ۰,۶۵۷ ثانیه

نمونه خروجی:


```
{'_id': 'دزی', 'average': 48.83343502135448, 'difference': -0.3370774149655631}
{'_id': 'نادفصا',
 'average': 48.81626221241727,
 'difference': -0.35425022390277405}
{'_id': 'نااحسرل',
 'average': 48.7416149068323,
 'difference': -0.42889752948774884}
{'_id': 'یلامش نااسارخ',
 'average': 48.8346358236949,
 'difference': -0.33587661262514246}
{'_id': 'نااگزمره',
 'average': 48.66339668914776,
 'difference': -0.5071157471722856}
{'_id': 'یقرش نااچیابرذآ',
 'average': 48.91400875547217,
 'difference': -0.2565036808478 78}
{'_id': 'دمحاریوب و هیولیگهک',
 'average': 49.333543307086615,
 'difference': 0.16303087076656908}
{'_id': 'رهشوب',
 'average': 48.83653250773994,
 'difference': -0.3339799285801064}
{'_id': 'یزکرم',
 'average': 49.08741151123422,
 'difference': -0.08310092508582301}
{'_id': 'نارهت', 'average': 49.170512436320045, 'difference': 0.0}
{'_id': 'نااحسزوخ',
 'average': 48.55634674922601,
 'difference': -0.6141656870940366}
{'_id': 'مالیا',
 'average': 48.873834679925416,
 'difference': -0.29667775639462945}
{'_id': 'هاشنامرک',
 'average': 49.15732484076433,
 'difference': -0.013187595555713472}
{'_id': 'نادمه',
 'average': 49.10996563573883,
 'difference': -0.0605468005812142}
{'_id': 'زربلا',
 'average': 48.35148990708107,
 'difference': -0.8190225292389783}
{'_id': 'نامرک',
 'average': 48.677359654534236,
 'difference': -0.4931527817858097}
{'_id': 'نااچنز',
 'average': 49.06363069245165,
 'difference': -0.10688174386839222}
```


توضیحات:

ابتدا میانگین برای استان تهران محاسبه شده و در مرحله بعد این اختلاف میانگین بین هر استان و تهران محاسبه شده است. همانطور که مشخص است، مقدار این اختلاف برای خود استان تهران برابر ۰ است.

۵

فایل کد: step3-5.py

```
#!/usr/bin/python3

import pprint
import itertools
from pymongo import MongoClient

client = MongoClient()
db = client['bigdata99']

max_province = list(db.users.aggregate([
    {
       "$group": {
              "_id": "$location.state",
              "count": {
                     "$sum": 1
              }
       }
    },
    {
       "$sort": { "count": -1 }
    },
    {
       "$limit": 1
    }
]))[0]

min_province = list(db.users.aggregate([
    {
       "$group": {
              "_id": "$location.state",
              "count": {
                     "$sum": 1
              }
       }
    },
    {
       "$sort": { "count": 1 }
    },
    {
       "$limit": 1
    }
]))[0]

print("Max Record:")
pprint.pprint(max_province)
print("Min Record:")
pprint.pprint(min_province)
```

زمان اجرا: ۰,۶۵۷ ثانیه

نمونه خروجی:

```
Max Record:
{'_id': 'یبونج ن اس ارخ', 'count': 3343}
Min Record:
{'_id': 'زربل ا', 'count': 3121}
```