Data Analytics Assignment2

Group members: Chang Tian, Huan Chen, Jiayu Chen

Design data structures to represent data:

We use Python to import three .dat files as three collections into Mongodb database.

First, we read lines from .dat files.

Then, we parse the line to get the desired word and convert all words to lowercase for higher searching speed.

In the end, we insert documents into MongoDB.

code of importing data:

```
from pymongo import MongoClient

client = MongoClient('localhost', 27017)
db = client['DAweek2']
movies = db.movies

with open("movies.dat", encoding="utf-8") as file:
    for line in file:
        line = line.strip("\n")
        word = line.split("::")

        movieID = word[0]
        movieTitle = word[1].lower()
        genre = word[2].lower().split("|")

        movie = {"MovieID": movieID,
                 "Title": movieTitle,
                 "Genres": genre
                 }

        movies.insert_one(movie)

print("movies.dat loading finished")
```

```
tags=db.tags
with open("tags.dat", encoding="utf-8") as file:
    for line in file:
        line = line.strip("\n")
        word = line.split("::")

        userID = int(word[0])
        movieID = int(word[1])
        tag = word[2].lower()
        timestamp=word[3]

        tag = {"UserID": userID,
                 "movieID": movieID,
                 "Tag": tag,
                 "Timestamp": timestamp
                 }

        tags.insert_one(tag)

print("tags.dat loading finished")
```

Data Analytics Assignment2

Group members: Chang Tian, Huan Chen, Jiayu Chen

```
ratings = db.ratings
with open("ratings.dat",encoding="utf-8") as file:
    for line in file:
        line =line.strip("\n")
        word =line.split("::")

        userID= int(word[0])
        movieID= int(word[1])
        rating= float(word[2])
        timestamp= word[3]

        rating ={"UserID": userID,
                "MovieID":movieID,
                "Rating": rating,
                "Timestamp": timestamp
            }
        ratings.insert_one(rating)

print("ratings.dat loading finished")
```

Questions:

We use python to make a connection to database and get collections.

```
from pymongo import MongoClient

client = MongoClient('localhost', 27017)
db = client['DAweek2']
tagscollection = db['tags']
moviescollection = db['movies']
ratingscollection = db['ratings']
```

1. What genre is the movie CopyCat in?

Code:

```
genres = moviescollection.find({'Title':{'$regex':'CopyCat'.lower()}})
for genre in genres:
    print(genre['Genres'])
```

Output:

['crime', 'drama', 'horror', 'mystery', 'thriller']

2. what genre has the most movies?

Code:

Data Analytics Assignment2

Group members: Chang Tian, Huan Chen, Jiayu Chen

```
from bson.code import Code
mapper = Code("""
              function(){
                  this.Genres.forEach(function(genre){
                      emit(genre,1);
                  });
               }
               """)

reducer = Code("""
               function( key, values) {
                   var total = 0;
                   for (var i=0; i<values.length; i++) {
                       total += values[i];
                   }
                   return total;
                }
                """)
genres = db.movies.map_reduce(mapper, reducer,"myresults")

for genre in genres.find({}).sort("value",-1).limit(1):
    print (genre)
```

Output:

```
{'_id': 'drama', 'value': 5339.0}
```

3. what tags did user 146 use to describe the movie "2001: A Space Odyssey”?

Code:

```
findMovie = movies.find({'Title':{'$regex':'^2001: a space odyssey'}})
print(findMovie)
for movie in findMovie:
    movieID = movie['MovieID']

findUser = tags.find({'UserID': 146,'movieID': int(movieID)})
for user in findUser:
    print(user['Tag'])

print("finished!")
```

Output:

```
<pymongo.cursor.Cursor object at 0x11254ab70>
```

4. What are the top 5 movies with the highest avg rating?

Code:

Data Analytics Assignment2

Group members: Chang Tian, Huan Chen, Jiayu Chen

```
'''
find the top 5 movies' information with the highest avg rating
'''
res = ratingscollection.aggregate([
    {"$group":
     {"_id":"$MovieID",
      "ratingAve":{"$avg":"$Rating"}
      }
     },
    {"$sort":{"ratingAve":-1, "_id":-1}},
    {"$limit":5}
    ])

'''
find the top 5 movies' movieID and movieName
'''
movieIDs = []
movieNames = []

for item in res:
    id = item["_id"]
    movieIDs.append(id)

    name=moviescollection.find_one({"MovieID":id})["Title"]
    movieNames.append(name)

print(movieIDs)
print(movieNames)
```

Output:

```
[33264, 64275, 42783, 51209, 53355]
["satan's tango (sátántangó) (1994)", 'blue light, the (das blaue licht) (1932)'
, 'shadows of forgotten ancestors (1964)', 'fighting elegy (kenka erejii) (1966)
', 'sun alley (sonnenallee) (1999)']
```

5) Write 3 different queries of your choice to demonstrate that your data storage is working.

- get all tags user 20 used to decribe movies

```
'''
get all tags user 20 used to decribe movies
'''
userTags=[]
user20 = tags.find({"UserID":20})
for user in user20:
    tag = user['Tag']
    userTags.append(tag)

print(userTags)
print("Finished!")
```

Output:

```
['politics', 'satire', 'chick flick 212', 'hanks', 'ryan', 'action', 'bond', 'sp
oof', 'star wars', 'bloody', 'kung fu', 'tarantino']
```

- find 5 movies with largest number of ratings

Data Analytics Assignment2

Group members: Chang Tian, Huan Chen, Jiayu Chen

```
'''
find 5 movies with largest number of ratings
'''
res = ratings.aggregate([
    {"$group":
        {"_id":"$MovieID",
        "rating_num": {"$sum":1}
        }
    },
        {"$sort":{"rating_num":-1}},
        {"$limit":5}
 ])
movieIDs =[]
movieNames = []
for group in res:
    id = group["_id"]
    movieIDs.append(id)
    name = movies.find_one({"MovieID": id})["Title"]
    movieNames.append(name)
print(movieIDs)
print(movieNames)
print("Finished!")
```

Output:

```
[296, 356, 593, 480, 318]
['pulp fiction (1994)', 'forrest gump (1994)', 'silence of the lambs, the (1991)
', 'jurassic park (1993)', 'shawshank redemption, the (1994)']
```

3.find all the movies with rating over than 3 rated by user 2

```
'''
find all the movies with rating over than 3 rated by user 2
'''

res = ratings.find({
    "Rating": {"$gte":3, "$lte":5},
    "UserID": 2
})
movieIDs =[]
movieNames = []
for parts in res:
    id = parts["MovieID"]
    movieIDs.append(id)

    name = movies.find_one({"MovieID": id})["Title"]
    movieNames.append(name)

print(movieIDs)
print(movieNames)
```

Output:

Data Analytics Assignment2

Group members: Chang Tian, Huan Chen, Jiayu Chen

```
[110, 151, 260, 376, 539, 590, 719, 733, 736, 780, 786, 1049, 1073, 1210, 1356,
1391, 1544]
['braveheart (1995)', 'rob roy (1995)', 'star wars: episode iv - a new hope (a.k
.a. star wars) (1977)', 'river wild, the (1994)', 'sleepless in seattle (1993)',
 'dances with wolves (1990)', 'multiplicity (1996)', 'rock, the (1996)', 'twiste
r (1996)', 'independence day (a.k.a. id4) (1996)', 'eraser (1996)', 'ghost and t
he darkness, the (1996)', 'willy wonka & the chocolate factory (1971)', 'star wa
rs: episode vi - return of the jedi (1983)', 'star trek: first contact (1996)',
'mars attacks! (1996)', 'lost world: jurassic park, the (jurassic park 2) (1997)
']
```