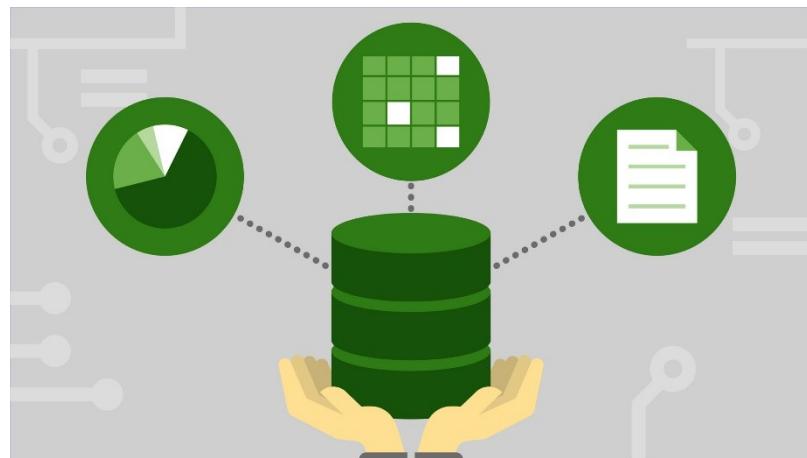


به نام خدا



دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده برق و کامپیوتر



آزمایشگاه پایگاه داده

ستور کار شماره ۶

آتیه آرمین

۸۱۰۱۹۷۶۴۸

مهرماه ۱۴۰۰

گزارش دستورکار انجام شده

بخش اول

در این بخش ابتدا در mongo shell، با استفاده از دستور `db.tweets.insert(...)` به صورت تک تک ۱۰ های دریافت شده با تکه کد پایتون را وارد کردم. بعد دیدم در صورت پروژه نوشته شده است که یک دیتابیس به اسم sahamyab با ای به نام tweets collection باید داشته باشیم ولی دیتابیس ساخته شده برای من نامش test بود. به همین دلیل یک خروجی json از داده های اضافه شده گرفته، سپس یک دیتابیس به نام tweets collection ای به نام sahamyab درست کرده و از طریق add data، این ۱۰ را به آن اضافه کردم. با توجه به داده های ذخیره شده مشخص است که ای به نام `_id` به صورت خودکار به داده ها اضافه شده است.

در ادامه همانطور که گفته شده است تکه کد را به صورت زیر تغییر داده تا ۵۰ توییت بتوان دریافت کرد. البته زمان بسیار زیادی طول کشید و بیش از ۱۱ توییت به توییت ها اضافه نشد و توییت های جدیدی پیدا نشد. به همین خاطر از فایل json ای که در گروه درس توسط یکی از دانشجویان قرار گرفته بود که شامل ۵۰ توییت بود، برای ادامه گزارش کار استفاده کردم که در عکس سوم نیز خروجی آن مشخص است.

```

# Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
untitled.py
Users > atieharmain > Desktop > untitled.py ...
1  from pymongo import MongoClient
2  import requests, time
3
4  URL = 'https://www.sahamyab.com/guest/twitter/list?v=1.0'
5  client = MongoClient('localhost', 27017)
6  db = client.sahamyab
7  tweets = db.tweets
8
9  while (tweets.count_documents({}) < 500):
10     response = requests.request('GET', URL, headers={'User-Agent': 'Chrome/61'})
11     if response.status_code == requests.codes.ok:
12       items = response.json()['items']
13       for item in items:
14         try:
15           tweets.replace_one({'_id': item['_id']}, item, upsert=True)
16           print(item)
17         except Exception as e:
18           print("Mongo Exception: " + str(e))
19     time.sleep(60)
20

```

2

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with 'Local' selected, showing '5 DBS' and '3 COLLECTIONS'. Under 'sahamyab', there are 'tweets' and 'test' collections. The main area is titled 'sahamyab.tweets' and shows 'DOCUMENTS 10'. Each document is represented by a snippet of JSON. One document's content is partially visible:

```

_id: ObjectID("61c87a09eae95a3ea0ca9e8")
id: "406642768"
sendTime: "2021-12-26T13:58:07Z"
sendTimePersian: "۱۴۰۰/۱۰/۰۵ ۱۷:۲۸"
senderName: "Kaypix"
senderUsername: "ahmad09"
senderProfileImage: "8eb7b21b-4f27-4cd8-9ad5-0ad73a61db8a"
content: "**** در شهر هاشمی شهر ۲۱۱۶۷ میلیون راید فروخته اند پیش از ۱۰۰****"
type: "postDate: \"140527101922"
scoredPostDate: "140527101922"
finalPullDatePersian: ""

```

This screenshot shows the same MongoDB Compass interface as above, but with a significantly larger dataset. The 'DOCUMENTS' count is now '500'. The list view displays more documents, each with a unique '_id' and a similar structure to the ones in the first screenshot, though the content is mostly redacted with '****'.

بخش دوم)

در این بخش یک تکه کد پایتون به صورت زیر پیاده سازی شده است. در این کد ابتدا روی تمام tweet ها پیمایش میکنیم، در هر محتوای قسمت content را جدا کرده و سپس hashtag های آن را توسط تابع findall از کتابخانه re، پیدا کرده و تبدیل به یک لیست میکنیم و سپس با دستور update_one این لیست از hashtag ها را به صورت یک فیلد به آن رکورد ذخیره شده از آن tweet اضافه میکنیم. زمان انجام این دستور نیز در عکس سوم نشان داده شده است.

The screenshot shows a Mac OS X desktop environment. At the top, there's a standard OS X menu bar with options like Code, File, Edit, Selection, View, Go, Run, Terminal, Window, Help, and a status bar showing 'Persian - Standard' and '100%'. Below the menu is a code editor window titled 'untitled.py 2' containing Python code. The code imports MongoClient, requests, time, and re modules, connects to a local MongoDB instance, and iterates over tweets in the 'tweets' collection. It extracts hashtags from each tweet's content using re.findall and updates the tweet document with a new 'hashtags' field containing the found hashtags. The bottom part of the screenshot shows the MongoDB Compass interface. On the left, the database structure is shown with 'sahamyab.tweets' selected. In the main pane, the 'Documents' tab is active, displaying 500 documents. One specific document is expanded, showing its detailed fields including '_id', 'content' (containing Persian text), 'type' ('Tweet'), and 'hashtags' (an array). Two red arrows point from the code editor towards this expanded document in the MongoDB interface, illustrating the connection between the two.

```

(base) atieharmin@Atiehs-MacBook-Pro Desktop % python untitled.py
0.2286670207977295 seconds
(base) atieharmin@Atiehs-MacBook-Pro Desktop %

```

بخش سوم)

۱- با استفاده از دستور `find` میتوان کاربرانی که `mediaContentType` توابیت آن ها `jpeg/image` است و آن ها مقدار دارد بدهست آورده و نام آن ها را نشان داد. درواقع در محیط گرافیکی میتوان این شرط را برای بخش `filter` و سپس شرط نشان دادن نام را در بخش `project` اضافه کرد. زمان اجرا نیز در عکس دوم نشان داده شده است.

The screenshot shows the MongoDB Compass interface. On the left, the sidebar shows the database structure: Local, 5 DBs, 3 Collections, sahamyab.tweets selected. The main area displays the results of a `find` query on the 'tweets' collection. The query is:

```

  FILTER: { "mediaContentType": "image/jpeg", "parentId": {"$exists": "true"} }
  PROJECT: { "senderUsername": 1, "_id": 0 }
  SORT: { field: -1 } or [ { "field": -1 } ]
  COLLATION: { locale: 'simple' }
  
```

The results list several senderUsernames:

- senderUsername: "ermostafaa"
- senderUsername: "vorodi97bours"
- senderUsername: "peymanjanchora"
- senderUsername: "mo-g"
- senderUsername: "ghhhhhh"
- senderUsername: "tanha."
- senderUsername: "mostly"
- senderUsername: "maahr"
- senderUsername: "trader251926"

The screenshot shows the MongoDB Compass interface with the 'Explain Plan' tab selected. The same query is run on the 'tweets' collection. The explain plan details the execution steps:

- PROJECTION_SIMPLE**: nReturned: 22, Execution Time: 0 ms.
- COLLSCAN**: nReturned: 22, Execution Time: 0 ms., Documents Examined: 500.

A note indicates: "No index available for this query."

۲ - بازه ۱۵ دقیقه‌ای دلخواه را از روی tweet یکی از sendtimePersian query‌ها انتخاب شده است. در عکس دوم زمان احرا نشان داده است. نکته این است که برای بزرگتر مساوی و کمتر استفاده می‌شوند.

۳- برای این query نیز مانند دو دستور قبلی عمل میکنیم. در این query، از عملگر eq که همان مساوی است استفاده شده است. برای استفاده از این عملگر باید از expr استفاده شود.

The screenshot shows the MongoDB Compass interface. On the left, the database structure is visible with 'sahamyab.tweets' selected. The main pane displays the results of a query. The query details are as follows:

```

FILTER: { "$expr": { "$eq": [ { "$dateFromString": { "dateString": "$sendTime" } }, { "$dateFromString": { "dateString": "$dateString" } } ] } }
PROJECT: { "senderName": 1, "senderProfileImage": 1, "_id": 0 }
SORT: { field: -1 } or [ { "field": -1 } ]
COLLATION: { locale: 'simple' }

MAX TIME MS: 60000
SKIP: 0
LIMIT: 0
  
```

The results pane shows 32 documents returned from 500 examined. The first few documents in the results list are:

- senderName: "دوچرخه سوار سپهران" senderProfileImage: "f4004390-10d7-46e9-a858-07e83c3e81a6"
- senderName: "Ali" senderProfileImage: "default"
- senderName: "Mohsenfazal167" senderProfileImage: "default"
- senderName: "سونا دانلورپرس" senderProfileImage: "default"
- senderName: "محمد حسنه" senderProfileImage: "default"
- senderName: "Ali" senderProfileImage: "default"
- senderName: "Mohsenfazal167" senderProfileImage: "default"

The screenshot shows the MongoDB Compass interface with the 'Explain Plan' tab selected. The query details are identical to the previous screenshot. The results pane now displays the execution plan:

```

FILTER: { "$expr": { "$eq": [ { "$dateFromString": { "dateString": "$sendTime" } }, { "$dateFromString": { "dateString": "$dateString" } } ] } }
PROJECT: { "senderName": 1, "senderProfileImage": 1, "_id": 0 }
SORT: { field: -1 } or [ { "field": -1 } ]
COLLATION: { locale: 'simple' }

MAX TIME MS: 60000
SKIP: 0
LIMIT: 0
  
```

The 'Query Performance Summary' section shows the following metrics:

- Documents Returned: 32
- Actual Query Execution Time (ms): 13
- Index Keys Examined: 0
- Sorted in Memory: no
- Documents Examined: 500
- No index available for this query.

The execution plan details two stages:

- PROJECTION_SIMPLE**: nReturned: 32, Execution Time: 0 ms
- COLLSCAN**: nReturned: 32, Execution Time: 1 ms

The 'COLLSCAN' stage is circled in blue.

در هر یک از query های بالا برای بدست آوردن زمان اجرا میتوان به جای نمای درختنی، از raw json استفاده کرد تا دقیق‌تر زمان اجرا را بررسی کنیم. به طور مثال برای query سوم، raw json به صورت زیر است که در بخش executionstats میتوان اطلاعات دقیق‌تری را پیدا کرد:

بخش چهارم)

برای پیاده‌سازی این دستورات چون روش گرافیکی آن را پیدا نکردم در پایتون و با استفاده از همان pymongo پیاده‌سازی کردم. برای این دستورها از aggregate استفاده شده است.

- برای پیاده‌سازی این دستور ابتدا روی کاربران group by میکنیم سپس روی تعداد توییت‌ها گروه بندی انجام می‌دهیم.

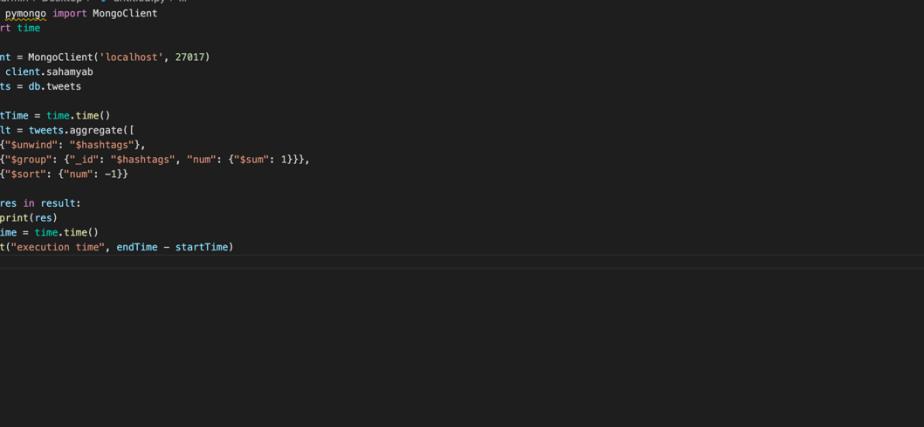
```

Code File Edit Selection View Go Run Terminal Window Help
untitled.py
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
untitled.py 1
1 from pymongo import MongoClient
2 import time
3
4 client = MongoClient('localhost', 27017)
5 db = client.sahamyab
6 tweets = db.tweets
7
8 startTime = time.time()
9 result = tweets.aggregate(
10 [
11     {"$group": {"_id": "$senderUsername", "num": {"$sum": 1}}},
12     {"$group": {"_id": {"$cond": {
13         "if": { "$eq": [ "$num", 1 ] },
14         "then": "one tweet", "else": {"$cond": {
15             "if": { "$lt": [ "$num", 4 ] },
16             "then": "two/three tweets",
17             "else": {"$cond": {
18                 "if": { "$gt": [ "$num", 3 ] },
19                 "then": "more than three tweets",
20                 "else": "$REMOVE"
21             }}
22         }},
23         "numUsers": {"$sum": 1}
24     }}
25 ]
26 )
27
28 for res in result:
29     print(res)
30 endTime = time.time()
31 print("execution time", endTime - startTime)
32

```

```
[base] atieharmin@Atiehs-MacBook-Pro Desktop % python untitled.py
{'_id': 'more than three tweets', 'numUsers': 10}
{'_id': 'two/three tweets', 'numUsers': 66}
{'_id': 'one tweet', 'numUsers': 307}
execution time 0.0035881996154785156
(base) atieharmin@Atiehs-MacBook-Pro Desktop %
```

2- برای آنکه تمام hashtag ها را داشته باشیم از دستور unwind استفاده میکنیم. سپس روی hashtag ها groupby انجام داده و تعداد tweet های هر دسته را بدست می آوریم. در نهایت با `$sort -1` به صورت نزولی مرتب شده اند.

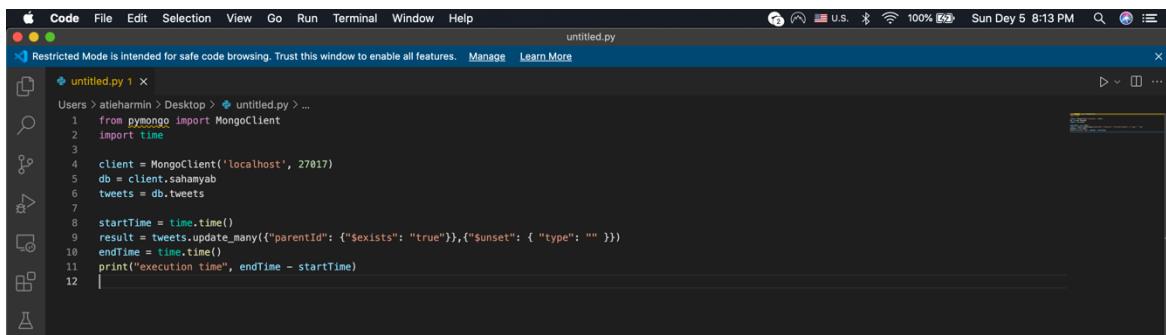


A screenshot of a Mac OS X desktop environment. At the top, there's a dark blue menu bar with icons for Apple, Code, File, Edit, Selection, View, Go, Run, Terminal, Window, and Help. The title bar shows "untitled.py". The system status bar at the top right indicates it's Sunday, May 5, 7:37 PM, with battery level at 100%, signal strength, and network connectivity. Below the menu bar is a toolbar with various icons for file operations like Open, Save, Find, and Print. A status message "Restricted Mode is intended for safe code browsing. Trust this window to enable all features." is displayed above the main content area. The main content area contains a code editor with Python code for MongoDB aggregation:

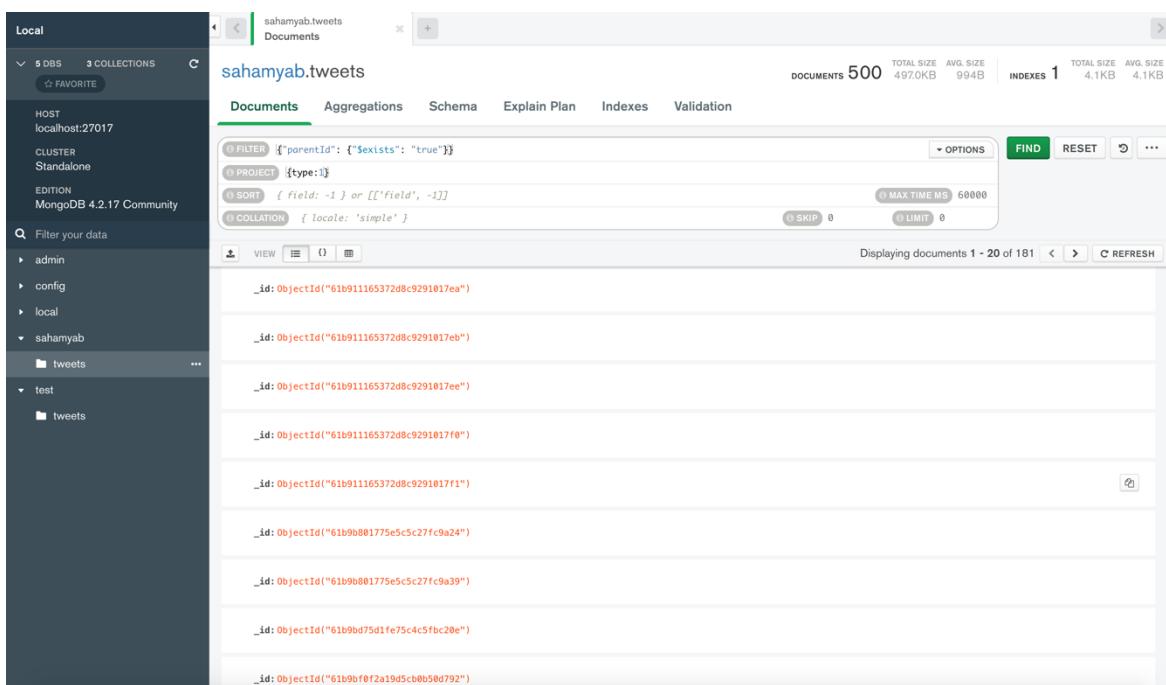
```
untitled.py 1

Users > altheramin > Desktop > untitled.py > ...
1   from pymongo import MongoClient
2   import time
3
4   client = MongoClient('localhost', 27017)
5   db = client.sahamyab
6   tweets = db.tweets
7
8   startTime = time.time()
9   result = tweets.aggregate([
10     {"$unwind": "$shashtags"},
11     {"$group": {"_id": "shashtags", "num": {"$sum": 1}}},
12     {"$sort": {"num": -1}}
13   ])
14   for res in result:
15     print(res)
16   endTime = time.time()
17   print("execution time", endTime - startTime)
18
```

۳- در این query با استفاده از دستور update_many آن هایی که parentId دارند را، type شان را میکنیم. همانطور که در عکس دوم مشخص است هیچ یک از آن هایی که parentId دارند، فیلد type را ندارند.



```
Code File Edit Selection View Go Run Terminal Window Help
untitled.py 1 ×
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
untitled.py
1 from pymongo import MongoClient
2 import time
3
4 client = MongoClient('localhost', 27017)
5 db = client.sahamyab
6 tweets = db.tweets
7
8 startTime = time.time()
9 result = tweets.update_many({"$exists": "true"}, {"$unset": { "type": "" }})
10 endTime = time.time()
11 print("execution time", endTime - startTime)
12 |
```



Local

5 DBS 3 COLLECTIONS C

HOST localhost:27017

CLUSTER Standalone

EDITION MongoDB 4.2.17 Community

Filter your data

- admin
- config
- local
- sahamyab
 - tweets
- test
- tweets

sahamyab.tweets

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER: {"\$exists": "true"} PROJECT: {type:1} SORT: {field: -1} or [{"field": -1}] COLLATION: { locale: "simple" }

MAX TIME MS: 60000 SKIP: 0 LIMIT: 0

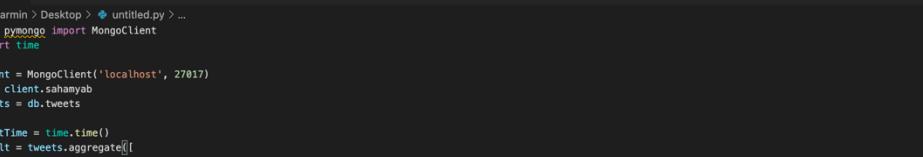
DOCUMENTS 500 TOTAL SIZE 497.0KB AVG. SIZE 994B INDEXES 1 TOTAL SIZE 4.1KB AVG. SIZE 4.1KB

Displaying documents 1 - 20 of 181

_id	type	parentId
ObjectId("61b911165372d8c9291017ea")		
ObjectId("61b911165372d8c9291017eb")		
ObjectId("61b911165372d8c9291017ee")		
ObjectId("61b911165372d8c9291017f0")		
ObjectId("61b911165372d8c9291017f1")		
ObjectId("61b9b801775e5c5c27fc9a24")		
ObjectId("61b9b801775e5c5c27fc9a39")		
ObjectId("61b9bd75d1fe75c4c5fb20e")		
ObjectId("61b9bf0f2a19d5cb0b50d792")		

```
[base] atieharmin@Atiehs-MacBook-Pro Desktop % python untitled.py
execution time 0.00626110305786133
[base] atieharmin@Atiehs-MacBook-Pro Desktop %
```

4- برای این کار از همان کد بخش دو استفاده میکنیم. در اینجا ابتدا result را که type آن command cursor است را به لیست cast میکنیم، سپس بیشترین مقدار که اولین است و کمترین مقدار ها که برابر ۱ است را چاپ میکنیم.



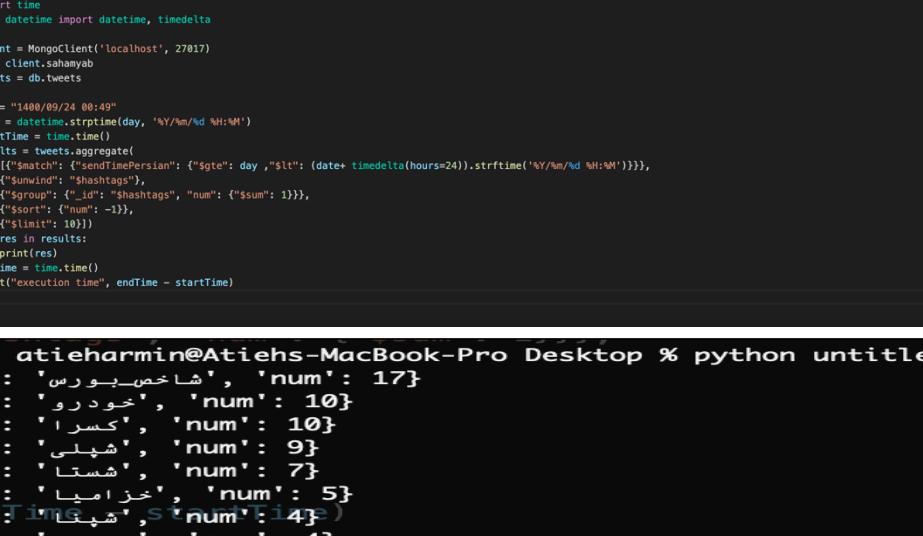
untitled.py

Restricted Mode is intended for safe code browsing. Trust this window to enable all features. [Manage](#) [Learn More](#)

```
untitled.py 1 ×

Users > ateharmin > Desktop > untitled.py > ...
1   from pymongo import MongoClient
2   import time
3
4   client = MongoClient('localhost', 27017)
5   db = client.sahamyah
6   tweets = db.tweets
7
8   startTime = time.time()
9   result = tweets.aggregate([
10      {"$unwind": "$hashtags"}, 
11      {"$group": {"_id": "$hashtags", "num": {"$sum": 1}}}, 
12      {"$sort": {"num": -1}} ])
13
14   res = list(result)
15   maximum = []
16   minimum = []
17   maximum.append(res[0]["_id"])
18   for i in res:
19       if(i["num"] == maximum[0]):
20           maximum.append(i["_id"])
21       if(i["num"] == 1):
22           minimum.append(i["_id"])
23
24   print("maximum: ", maximum)
25   print("minimum: ", minimum)
26
27   endTime = time.time()
28   print("execution time", endTime - startTime)
```

5- برای این query باید یک روز مشخص و رودی داده شود. سپس ابتدا تمام توابیت های آن روز مشخص را بدست آورده و سپس hashtag های آن ها را مانند قبلاً یکی کرده و مرتب میکنیم و ۱۰ تای اول آن ها را نشان می دهیم.



A screenshot of a Mac OS X desktop environment. At the top, there's a menu bar with Apple, Code, File, Edit, Selection, View, Go, Run, Terminal, Window, Help, and a search icon. The system status bar on the right shows battery level (100%), signal strength, and the date and time (Sun Dec 5 8:46 PM). Below the menu bar is a toolbar with various icons for file operations like Open, Save, Print, and Find.

The main area shows a terminal window titled "untitled.py" and a code editor window titled "untitled.py 1".

Terminal Output:

```
(base) atieharmin@Atiehs-MacBook-Pro Desktop % python untitled.py
[{"_id": "شامیا", "num": 17}, {"_id": "خودرو", "num": 10}, {"_id": "کسرای", "num": 10}, {"_id": "شهلی", "num": 9}, {"_id": "هستا", "num": 7}, {"_id": "خرامیا", "num": 5}, {"_id": "شینا", "num": 4}, {"_id": "پتروول", "num": 4}, {"_id": "دی", "num": 3}, {"_id": "نوری", "num": 3}]
execution time 0.0073430538177490234
(base) atieharmin@Atiehs-MacBook-Pro Desktop %
```

Code Editor Content:

```
untitled.py 1
Users > atieharmin > Desktop > untitled.py > ...
1  from pymongo import MongoClient
2  import time
3  from datetime import datetime, timedelta
4
5  client = MongoClient('localhost', 27017)
6  db = client.sahamyab
7  tweets = db.tweets
8
9  day = "1400/09/24 00:49"
10 date = datetime.strptime(day, '%Y/%m/%d %H:%M')
11 startTime = time.time()
12 results = tweets.aggregate([
13     {"$match": {"$and": [{"$gte": "day", "$lt": (date+ timedelta(hours=24)).strftime('%Y/%m/%d %H:%M')}]}},
14     {"$unwind": "$hashtags"},
15     {"$group": {"_id": "$hashtags", "num": {"$sum": 1}}},
16     {"$sort": {"_id": -1}},
17     {"$limit": 10}}
18 for res in results:
19     print(res)
20 endTime = time.time()
21 print("execution time", endTime - startTime)
22
```

6 - در این query مانند قبلي عمل ميکنيم. ابتدا تمام توبييت هاي يك روز را بحسب آورده و سپس روی نام افراد کرده و تعداد tweet هر کدام را بحسب میآوريم و سپس مرتب ميکنيم و نام اولين نفر را چاپ ميکنيم.

```

Code File Edit Selection View Go Run Terminal Window Help
untitled.py
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
untitled.py 1 ×
Users > atieharmin > Desktop > untitled.py > ...
5   client = pymongo.MongoClient('localhost', 27017)
6   db = client.sahamayab
7   tweets = db.tweets
8
9   day = "1400/09/24 00:49"
10  date = datetime.strptime(day, "%Y/%m/%d %H:%M")
11  startTime = time.time()
12  result = tweets.aggregate([
13    {"$match": {"sendTimePersian": {"$gte": day, "$lt": (date + timedelta(hours=24)).strftime("%Y/%m/%d %H:%M")}}},
14    {"$group": {"_id": "$senderUsername", "num": {"$sum": 1}}},
15    {"$sort": {"num": -1}}])
16  print(list(result)[0])
17  endTime = time.time()
18  print("execution time", endTime - startTime)
19

```

```

(base) atieharmin@Atiehs-MacBook-Pro Desktop % python untitled.py
{'_id': '2266mazi', 'num': 4}
execution time 0.009315967559814453
(base) atieharmin@Atiehs-MacBook-Pro Desktop %

```