Friday, 25 September 2020

Submit all experiments in one ipynb file.

1 pymongo insert (30 mins)

1.1 Objectives

1. Can use pymongo to store data
2. Understand the document in mongoDB
3. Learn the new function **insert_many**
4. Knows the difference between **insert_one** and **insert_many** function

1.2 Description

Read the score.csv file which is attached in the lab zip file. The file format is same as in the previous lab as shown below.

```
5995778521,4,0,4,6,7
5974831121,6,5,0,6,4
5992211621,9,2,0,0,4
5987519321,3,5,1,4,3
5979431521,5,7,1,8,2
5976635721,5,6,5,5,1
5911249921,0,6,1,3,7
5906374521,9,0,7,3,8
5946204221,8,4,4,1,6
5963287221,4,0,5,5,9
```

Store score of all students in the collection `scores` in a new database. The format of each document should be `{'id':xxxxxxxxx,'quiz 1':x,'quiz 2':x,'quiz 3':x,'quiz 4':x,'quiz 5':x,'sum':xx}` as an example below.

```
{'id':5995778521,'quiz 1':4,'quiz 2':0,'quiz 3:4',
'quiz 4':6,'quiz 5':7,'sum':21}
```

To create a database, just simply refer a database name which does not exist on your server as the example below.

```
client = pymongo.MongoClient(connection_string)
db = client.student_scores
```

1.3 Procedure

Use google colab for this experiment. Write your program in two cells.

1. Use your own connection string generated from your cluster.

2. Read all lines of student scores. Store each line in a dictionary. Insert the obtained dictionaries to the database, one by one.

3. Delete all documents you've just inserted from the first line using a bin button next to the collection's name. Then, read all lines of student scores. Store each line in a dictionary. Keep all dictionaries in a list. Add the obtained list using the function **insert_many** which is explained below. **Please observe the time used by *insert_many* function comparing to the time used by adding one by one record using *insert_one* function.**

```
# list_of_data is a variable which stores a list of
    dictionaries
# scores is a variable which point to the
   collection
db.scores.insert_many(list_of_data,ordered=False)
```

1.4 Sample Output

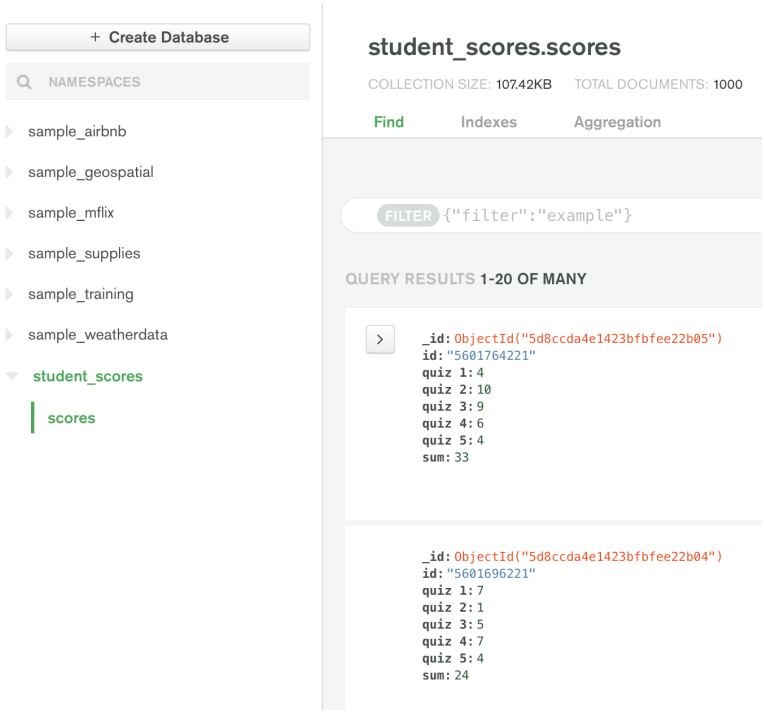

Figure 1: *Sample output*

2 pymongo find_one (15 mins)

2.1 Objectives

1. Can use pymongo to find data using the find_one method
2. Understand the document in mongoDB

2.2 Description

Use the find_one method to retrieve scores of the input student id.

2.3 Procedure

Read a student_id from the keyboard. Find the sum score of the obtained id. Print the result to the screen.

2.4 Sample Output

```
Please enter an id: 5600148421
Sum score: 26
```

3 pymongo find (condition and sort) (15 mins)

3.1 Objectives

1. Can use `find` method with condition
2. Can sort the results returned from the `find` method

3.2 Description

To retrieve data from mongoDB, we always use some conditions or sort the documents from a collection. In this experiment, please select only the student scores which is greater than 30 and sort descending.

Pymongo can handle the conditions using a dictionary with some specific keys, for example `$gt` or `$lt` and provide the threshold in the value of such keys. An example of condition is shown below. The output is student's id and scores printed on the screen.

```
docs=db.scores.find({'sum':{'$gt':26}}).sort('sum',
    pymongo.DESCENDING)
```

3.3 Sample Output

```
5639110921 10 10 8 10 8 46
5676237921 10 10 9 9 7 45
5698907421 10 10 6 10 7 43
5635209721 9 10 8 10 5 42
5692820921 7 10 8 9 8 42
5696614321 5 10 9 10 8 42
5616166921 7 10 6 10 8 41
5631371621 9 9 6 9 8 41
5654331721 8 8 7 10 8 41
5663510921 8 7 8 10 8 41
5672488021 8 10 6 10 7 41
```

4 pymongo find (15 mins)

4.1 Objectives

1. Can use pymongo to find data using the find method
2. Understand the document in mongoDB
3. Can use the key and sub key to find data

4.2 Description

Normally, we can send a key you want to find as a parameter in the form of dictionary. However, in some cases like the one in the sample air bnb database. The country of the place is hidden in the key 'address' and 'country' as shown in the figure below. If you want to use this key you have to use the key `'address.country'` and provide the country you want as the value of this key, such as

`air_bnb_db.listingsAndReviews.find('address.country': 'Brazil')`.

In this experiment, please print all places in Brazil from the `sample_airbnb.listingsAndReviews`

Figure 2: *`country` in `address` key*

4.3 Sample Output

```
Horto flat with small garden, Brazil
Apt Linda Vista Lagoa - Rio, Brazil
Copacabana Apartment Posto 6, Brazil
Catete’s Colonial Big Hause Room B, Brazil
Ótimo Apto proximo Parque Olimpico, Brazil
Apartamento zona sul do RJ, Brazil
```