Big Data – Homework 4 - MongoDB
125 points + 50 extra credit

Submit: A single source file, runnable in the mongo shell or suitable IDE. One section per question, separated by empty lines and comments (use #). **Name your file by your netid: e.g. jcr365-hw4.zip**

MongoDB – option 1

For this homework, any instance of MongoDB will work, and it is encouraged.

I recommend the Docker mongodb image, or one of the options in the MongoDB install page (hosted, local, etc).

Docker: https://hub.docker.com/_/mongo
Mongo Install: https://www.mongodb.com/docs/v4.2/installation/

MongoDB – option 2

There is also a MongoDB instance in Jupyterhub, accessible through the command line or a notebook. Each student has a login using your **netid**.

In a JupyterHub terminal:

```
conda activate bigdata-fall22

mongo --host=mongo-csgy-6513-fall.db  --authenticationDatabase=<netid>_db -
u <netid> -p <netid>
```

Once in mongodb, change to your own database:

```
>use <netid>_db;
```

To test you can read/write to collection foo:

```
>db.foo.insert( { x: 1, y: 1 } )

>db.foo.find()
```

If your login does not work, contact the professor.

Using Python w/MongoDB:

```
from pymongo import MongoClient

client = MongoClient('mongo-csgy-6513-fall.db',

                   username=’<netid>’,

                   password="<netid>",

                   authSource = "<netid>_db")

db=client.<netid>_db

db

col = db.foo

myquery = { "x": 1 }

mydoc = col.find(myquery)

for x in mydoc: print(x)

db.logout()
```

1. Create a database and load the data - 25 Points

Dataset: restaurants.json
Also available in the ‘shared/’ folder of JupyterHub

You can use any method to load your data. The purpose of the problem is to make you load data into Mongo.

Option 1: command line.
Use the ‘**mongoimport’** command line tool, available in the ‘shared/mongotools/’ folder, to import the data into a new table.

For instructions on how to use the import tool, see:
https://www.mongodb.com/docs/database-tools/mongoimport/

Option 2: Python + pymongo

Other options: whatever works for you

2. Write MongoDB queries for (4 points each): 100 points

1. Count the number of documents in the collection.
2. Display all the documents in the collection.
3. Display: restaurant_id, name, borough and cuisine for all the documents
4. Display: restaurant_id, name, borough and cuisine, but exclude field **_id,** for all the documents in the collection
5. Display: restaurant_id, name, borough and zip code, exclude the field **_id** for all the documents in the collection.
6. Display all the restaurants in the Bronx.
7. Display the first 5 restaurants in the Bronx

8. Display the second 5 restaurants (skipping the first 5) in the Bronx.

9. Find the restaurants with a score more than 85.

10. Find the restaurants that achieved a score, more than 80 but less than 100.

11. Find the restaurants which locate in latitude value less than -95.754168.

12. Find the restaurants that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than -65.754168.

13. Find the restaurants which do not prepare any cuisine of 'American' and achieved a score more than 70 and located in the longitude less than -65.754168. (without using $and operator).

14. Find the restaurants which do not prepare any cuisine of 'American ' and achieved a grade point 'A' and not in the borough of Brooklyn, sorted by cuisine in descending order.

15. Find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Wil' as first three letters for its name.

16. Find the restaurant Id, name, borough and cuisine for those restaurants which contain 'ces' as last three letters for its name.

17. Find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Reg' as three letters somewhere in its name.

18. Find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish

19. Find the restaurant Id, name, borough and cuisine for those restaurants which belong to the boroughs of Staten Island or Queens or Bronx or Brooklyn.

20. Find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronx or Brooklyn

21. Find the restaurant Id, name, borough and cuisine for those restaurants which achieved a score below 10.

22. Find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinese' or restaurant's name begins with letter 'Wil'.

23. Find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates.

24. Find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".

25. Find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and up to 52.

3. Extra Credit: 50 points

Datasets: historical-events.json, meteorites.json, worldcities.csv

Some Background: geospatial logic is possible in MongoDB using the geopastial library/facilities. https://docs.mongodb.com/manual/geospatial-queries/

1. **15 points: (historical-events.json):** Count the number of events per year**.** **Note**, you will need to create a new date field from the string provided. Assume numbers are years if the entry is not a valid date.

2. **25 points: (meteorites.json, worldcities.csv):** Use the MongoDB geospatial facilities to ***find the nearest city to each meteorite "fallen" (not found) since the year 1950, inclusive***. Distance is between coordinates, straight line.
   Note: 'worldcities' is a CSV file. You will need to import into MongoDB **AND** clean-up the double quotes.
   Note: Use the $near operator and select this closest entry per city