

CSP554—Big Data Technologies

Assignment #13

Worth: 5 points ALL EXTRA CREDIT

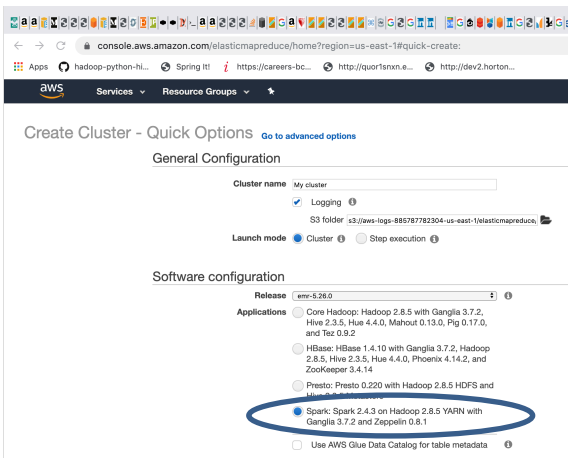
Due at the time you submit your final project or paper

Assignments should be uploaded via the Blackboard portal.

Set Up:

Step A – Start an EMR cluster

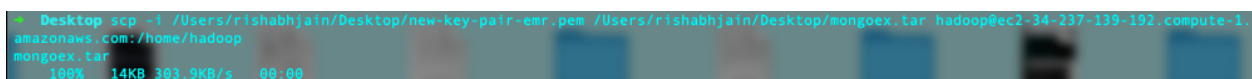
Start up an EMR/Hadoop cluster as previously, but instead of choosing the “Core Hadoop” configuration chose the “Spark” configuration (see below), otherwise proceed as before.



Step B – Download the assignment software (mongoex.tar, mongodb-org-4.2.repo) to master node

Download “mongoex.tar” (included as a file with the assignment) to your PC or MAC. Now, using “scp” copy this file to the EMR master node using something like the following (just an example):

```
scp -i ./emr-key-pair-2.cer /Users/nachdaph/csp554-fall-2021/assignments/mongoex.tar  
hadoop@ec2-44-199-215-205.compute-1.amazonaws.com:/home/hadoop
```



Now download “mongodb-org-4.2.repo” (included as a file with the assignment) to your PC or MAC. Now, using “scp” copy this file to the EMR master node using something like the following (just an example):

scp -i ./emr-key-pair-2.cer /Users/nachdaph/csp554-fall-2021/assignments/mongodb-org-4.2.repo hadoop@ec2-44-199-215-205.compute-1.amazonaws.com:/home/hadoop

```
Desktop scp -i /Users/rishabhjain/Desktop/new-key-pair-emr.pem /Users/rishabhjain/Desktop/mongodb-org-4.2.repo hadoop@ec2-34-237-139-192.c
ompute-1.amazonaws.com:/home/hadoop
mongodb-org-4.2.repo
100% 197 5.3KB/s 00:00
```

Step C – Install assignment software (mongoex.zip, mongodb-org-4.2.repo)

Enter the following into a terminal window which you have connected to the EMR master node. Going forward we will call this terminal connection Init-Term:

```
sudo cp mongodb-org-4.2.repo /etc/yum.repos.d
```

Then enter this into Init-Term to unzip mongoex.tar:

```
tar -xvf mongoex.tar
```

```
[hadoop@ip-172-31-9-80 ~]$ tar -xvf mongoex.tar
./_demo1.js
demo1.js
demo2.js
demo3.js
demo4.js
demo5.js
demo6.js
demo7.js
demo8.js
demo9.js
load.js
[hadoop@ip-172-31-9-80 ~]$ ls -lt
total 60
-rwxr-xr-x 1 hadoop hadoop 197 Apr 29 04:05 mongodb-org-4.2.repo
-rwxr-xr-x 1 hadoop hadoop 13824 Apr 29 04:05 mongoex.tar
-rw-r--r-- 1 hadoop hadoop 1747 Nov 16 2019 load.js
-rw-r--r-- 1 hadoop hadoop 87 Nov 16 2019 demo6.js
-rw-r--r-- 1 hadoop hadoop 148 Nov 16 2019 demo7.js
-rw-r--r-- 1 hadoop hadoop 81 Nov 16 2019 demo1.js
-rw-r--r-- 1 hadoop hadoop 100 Nov 16 2019 demo2.js
-rw-r--r-- 1 hadoop hadoop 93 Nov 16 2019 demo3.js
-rw-r--r-- 1 hadoop hadoop 98 Nov 16 2019 demo4.js
-rw-r--r-- 1 hadoop hadoop 58 Nov 16 2019 demo5.js
-rw-r--r-- 1 hadoop hadoop 66 Nov 16 2019 demo8.js
-rw-r--r-- 1 hadoop hadoop 78 Nov 16 2019 demo9.js
[hadoop@ip-172-31-9-80 ~]$
```

Step D – Install and start MongoDB

Enter the following into Init-Term to install MongoDB:

```
sudo yum install -y mongodb-org-4.2.15 mongodb-org-server-4.2.15 mongodb-org-shell-4.2.15
mongodb-org-mongos-4.2.15 mongodb-org-tools-4.2.15
```

Now enter this into Init-Term to start mongod:

```
sudo systemctl start mongod
```

```

Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
Installing : mongodb-org-shell-4.2.15-1.amzn1.x86_64 1/5
Installing : mongodb-org-mongos-4.2.15-1.amzn1.x86_64 2/5
Installing : mongodb-org-tools-4.2.15-1.amzn1.x86_64 3/5
Installing : mongodb-org-server-4.2.15-1.amzn1.x86_64 4/5
Installing : mongodb-org-4.2.15-1.amzn1.x86_64 5/5
Verifying : mongodb-org-4.2.15-1.amzn1.x86_64 1/5
Verifying : mongodb-org-server-4.2.15-1.amzn1.x86_64 2/5
Verifying : mongodb-org-tools-4.2.15-1.amzn1.x86_64 3/5
Verifying : mongodb-org-mongos-4.2.15-1.amzn1.x86_64 4/5
Verifying : mongodb-org-shell-4.2.15-1.amzn1.x86_64 5/5

Installed:
mongodb-org.x86_64 0:4.2.15-1.amzn1      mongodb-org-mongos.x86_64 0:4.2.15-1.amzn1
mongodb-org-server.x86_64 0:4.2.15-1.amzn1  mongodb-org-shell.x86_64 0:4.2.15-1.amzn1
mongodb-org-tools.x86_64 0:4.2.15-1.amzn1

Complete!
[hadoop@ip-172-31-9-80 ~]$ sudo systemctl start mongod
[hadoop@ip-172-31-9-80 ~]$

```

Step E – Start the MongoDB Shell (Command Line Interpreter)

Open a second terminal connection to the EMR master node. Going forward we will call this terminal connection: CLI-Term.

You will use this terminal window to start and run the mongodb shell as follows:

```
mongo
```

Step F – Edit mongo query language files

Open a third terminal connection to the EMR master node. Going forward we will call this terminal connection: CLI-Term. You will use this terminal window to run the ‘vi’ editor to create your Mongo code files.

As an alternative you could edit your MongoDB code files on your PC/MAC and then ‘scp’ them to the EMR mater node.

Step G – Setting up the assignment database

Now, in the MongoDB shell, using the CLI-Term, create a database called “assignment” by entering the following into the MongoDB shell:

```
use assignment;
```

```
> use assignment;
switched to db assignment
```

This will set the shell variable ‘db’ to this new database.

Load a collection called ‘unicorns’ with sample data by executing the script load.js in the MongoDB shell as follows (don’t cut and paste this, type it in manually):

```
load('./load.js');
```

```
> load('./load.js');
true
>
```

Note, look at the content of the script file (via the other terminal window you have opened to the EC2 instance) to see how each unicorn is described.

```
[hadoop@ip-172-31-9-80 ~]$ cat load.js
db.unicorns.insert({name: 'Horny',
  dob: new Date(1992,2,13,7,47),
  loves: ['carrot','papaya'],
  weight: 600,
  gender: 'm',
  vampires: 63});
db.unicorns.insert({name: 'Aurora',
  dob: new Date(1991, 0, 24, 13, 0),
  loves: ['carrot', 'grape'],
  weight: 450,
  gender: 'f',
  vampires: 43});
db.unicorns.insert({name: 'Unicrom',
  dob: new Date(1973, 1, 9, 22, 10),
  loves: ['energon', 'redbull'],
  weight: 984,
  gender: 'm'
});
```

Confirm this has all worked by executing the following command in the MongoDB shell:

```
db.unicorns.find();
```

```

> use mongo;
> use test;
> db.unicorns.find();
{"_id": "624b671b6fddeb3201bcd18e", "name": "Horny", "dob": ISODate("1992-02-13T07:47:00Z"), "loves": ["carrot", "papaya"], "weight": 600, "gender": "m", "vampires": 63 }
{"_id": "624b671b6fddeb3201bcd18f", "name": "Aurora", "dob": ISODate("1991-01-24T13:00:00Z"), "loves": ["carrot", "grape"], "weight": 450, "gender": "f", "vampires": 43 }
{"_id": "624b671b6fddeb3201bcd190", "name": "Unicrom", "dob": ISODate("1973-01-09T22:10:00Z"), "loves": ["energon", "redbull"], "weight": 984, "gender": "m", "vampires": 182 }
{"_id": "624b671b6fddeb3201bcd191", "name": "Roodoodle", "dob": ISODate("1979-09-18T10:44:00Z"), "loves": ["apple", "weight": 575, "gender": "m", "vampires": 59 }
{"_id": "624b671b6fddeb3201bcd192", "name": "Solitary", "dob": ISODate("1988-05-04T02:01:00Z"), "loves": ["apple", "carrot", "chocolate"], "weight": 550, "gender": "f", "vampires": 80 }
{"_id": "624b671b6fddeb3201bcd193", "name": "Ayna", "dob": ISODate("1998-03-07T08:30:00Z"), "loves": ["strawberry", "lemon"], "weight": 733, "gender": "f", "vampires": 40 }
{"_id": "624b671b6fddeb3201bcd194", "name": "Kenny", "dob": ISODate("1997-07-01T10:42:00Z"), "loves": ["grape", "lemon"], "weight": 630, "gender": "m", "vampires": 39 }
{"_id": "624b671b6fddeb3201bcd195", "name": "Raleigh", "dob": ISODate("2005-05-03T00:57:00Z"), "loves": ["apple", "sugar"], "weight": 421, "gender": "m", "vampires": 2 }
{"_id": "624b671b6fddeb3201bcd196", "name": "Luis", "dob": ISODate("2001-10-08T10:53:00Z"), "loves": ["apple", "watermelon"], "weight": 601, "gender": "f", "vampires": 33 }
{"_id": "624b671b6fddeb3201bcd197", "name": "Pilot", "dob": ISODate("1997-03-01T05:03:00Z"), "loves": ["apple", "watermelon"], "weight": 650, "gender": "m", "vampires": 54 }
{"_id": "624b671b6fddeb3201bcd198", "name": "Nimue", "dob": ISODate("1999-12-20T16:15:00Z"), "loves": ["grape", "carrot"], "weight": 540, "gender": "f" }
{"_id": "624b671b6fddeb3201bcd199", "name": "Dune", "dob": ISODate("1976-07-18T10:10:00Z"), "loves": ["grape", "watermelon"], "weight": 704, "gender": "m", "vampires": 105 }

```

Note, the files named “demo*.js” (also included in the mongoex.tar file) provide examples of how to operate in the unicorn collection. These are a VERY good idea to review and understand and will present you with information helpful in completing the assignment. Also, try them out by typing something like

```
load('./demo1.js');
```

```
> load('./demo1.js');
true
>
```

Exercises:

Exercise 1) (1 point)

Write a command that finds all unicorns having weight less than 500 pounds. Include the code you executed and some sample output as the result of this exercise. Recall you can place the command, if you choose, into a file, say ‘ex1.js’ and execute it with the load command as above and similarly for the following exercises.

Sol) Command used:- “db.unicorns.find({weight : {\$lt : 500}});”

```
> db.unicorns.find({weight : {$lt : 500}});
{"_id": "624b671b6fddeb3201bcd18f", "name": "Aurora", "dob": ISODate("1991-01-24T13:00:00Z"), "loves": ["carrot", "grape"], "weight": 450, "gender": "f", "vampires": 43 }
{"_id": "624b671b6fddeb3201bcd195", "name": "Raleigh", "dob": ISODate("2005-05-03T00:57:00Z"), "loves": ["apple", "sugar"], "weight": 421, "gender": "m", "vampires": 2 }
>
```

Exercise 2) (1 point)

Write a command that finds all unicorns who love apples. Hint, search for “apple”. Include the code you executed and some sample output as the result of this exercise.

Sol) Command used: - “db.unicorns.find({loves: {\$in:['apple']}});”

```
> db.unicorns.find(loves: {$in: ['apple']})
{ "_id" : ObjectId("426b671b6fdeb3201bc291"), "name" : "Roonoodles", "dob" : ISODate("1979-06-18T18:44:00Z"), "loves" : [ "apple" ], "weight" : 375, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("426b671b6fdeb3201bc292"), "name" : "Solnara", "dob" : ISODate("1985-07-04T02:01:00Z"), "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("426b671b6fdeb3201bc293"), "name" : "Raleigh", "dob" : ISODate("2005-05-03T08:37:00Z"), "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("426b671b6fdeb3201bc294"), "name" : "Sela", "dob" : ISODate("2001-05-08T14:15:00Z"), "loves" : [ "apple", "watermelon" ], "weight" : 591, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("426b671b6fdeb3201bc297"), "name" : "Pilot", "dob" : ISODate("1997-03-01T05:03:00Z"), "loves" : [ "apple", "watermelon" ], "weight" : 658, "gender" : "m", "vampires" : 54 }
```

Exercise 3) (1 point)

Write a command that adds a unicorn with the following attributes to the collection. Note dob means “Date of Birth.”

Attribute	Value(s)
name	Malini
dob	11/03/2008
loves	pears, grapes
weight	450
gender	F
vampires	23
horns	1

Include the code you executed to insert this unicorn into the collection along with the output of a find command showing it is in the collection.

Sol) Command used: - “db.unicorns.insert({name: 'Malini', dob: new Date(2008, 11, 03), loves: ['pears', 'grapes'], weight: 450, gender: 'F', vampires: 23, horns : 1});”

```
> db.unicorns.insert({name: 'Malini', dob: new Date(2008, 11, 03), loves: ['pears', 'grapes'], weight: 450, gender: 'F', vampires: 23, horns : 1});
WriteResult({ "nInserted" : 1 })
>
```

Exercise 4) (1 point)

Write a command that updates the above record to add apricots to the list of things Malini loves. Include the code you executed and some sample output showing the addition.

Sol) Command used: - “db.unicorns.update({name: 'Malini'}, {\$set : {loves: ['pears', 'grapes', 'apricots']}});”

```
> db.unicorns.update({name: 'Malini'}, {$set : {loves: ['pears', 'grapes', 'apricots']}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

Exercise 5) (1 point)

Write a command that deletes all unicorns with weight more than 600 pounds. Include the code you executed and some sample output as the result of this exercise.

Sol) Command used: - “db.unicorns.remove({weight: {\$gt : 600}});”

```
> db.unicorns.remove(weight: { $gt : 600 });  
WriteResult({ "nRemoved" : 6 })  
> |
```