Instructions

For each answer, please include your answer as text, and any screenshot(s) which demonstrate your answer was executed. Most importantly, make sure to include evidence your answer is correct. This will most likely be a screenshot. If you had issues, problems, or had to make assumptions include them in your answer.

Your Answers:

1. Write a MongoDB Query to retrieve Country name, population, and capital for all countries in the collection.

   > db.countries.find({}, {“name”: 1, “population”: 2, “capital”: 3 })

```
> db.countries.find( {}, { "name": 1, "population": 2, "capital": 3 })
{ "_id" : ObjectId("612da376727886cd3da2b4b1"), "name" : "Albania", "capital" : "Tirana", "population" : 2893005 }
{ "_id" : ObjectId("612da376727886cd3da2b4b2"), "name" : "Andorra", "capital" : "Andorra la Vella", "population" : 76949 }
{ "_id" : ObjectId("612da376727886cd3da2b4b3"), "name" : "Åland Islands", "capital" : "Mariehamn", "population" : 28875 }
{ "_id" : ObjectId("612da376727886cd3da2b4b4"), "name" : "Austria", "capital" : "Vienna", "population" : 8602112 }
{ "_id" : ObjectId("612da376727886cd3da2b4b5"), "name" : "Belarus", "capital" : "Minsk", "population" : 9485300 }
{ "_id" : ObjectId("612da376727886cd3da2b4b6"), "name" : "Bosnia and Herzegovina", "capital" : "Sarajevo", "population" : 3791622 }
{ "_id" : ObjectId("612da376727886cd3da2b4b7"), "name" : "Bulgaria", "capital" : "Sofia", "population" : 7202198 }
{ "_id" : ObjectId("612da376727886cd3da2b4b8"), "name" : "Belgium", "capital" : "Brussels", "population" : 11248330 }
{ "_id" : ObjectId("612da376727886cd3da2b4b9"), "name" : "Croatia", "capital" : "Zagreb", "population" : 4246800 }
{ "_id" : ObjectId("612da376727886cd3da2b4ba"), "name" : "Cyprus", "capital" : "Nicosia", "population" : 858000 }
{ "_id" : ObjectId("612da376727886cd3da2b4bb"), "name" : "Czech Republic", "capital" : "Prague", "population" : 10537818 }
{ "_id" : ObjectId("612da376727886cd3da2b4bc"), "name" : "Denmark", "capital" : "Copenhagen", "population" : 5678348 }
{ "_id" : ObjectId("612da376727886cd3da2b4bd"), "name" : "Estonia", "capital" : "Tallinn", "population" : 1313271 }
{ "_id" : ObjectId("612da376727886cd3da2b4be"), "name" : "Faroe Islands", "capital" : "Tórshavn", "population" : 48846 }
{ "_id" : ObjectId("612da376727886cd3da2b4bf"), "name" : "Finland", "capital" : "Helsinki", "population" : 5485215 }
{ "_id" : ObjectId("612da376727886cd3da2b4c0"), "name" : "Germany", "capital" : "Berlin", "population" : 81083600 }
{ "_id" : ObjectId("612da376727886cd3da2b4c1"), "name" : "France", "capital" : "Paris", "population" : 66186000 }
{ "_id" : ObjectId("612da376727886cd3da2b4c2"), "name" : "Greece", "capital" : "Athens", "population" : 10846979 }
{ "_id" : ObjectId("612da376727886cd3da2b4c3"), "name" : "Gibraltar", "capital" : "Gibraltar", "population" : 32734 }
{ "_id" : ObjectId("612da376727886cd3da2b4c4"), "name" : "Guernsey", "capital" : "St. Peter Port", "population" : 65150 }
Type "it" for more
>
```

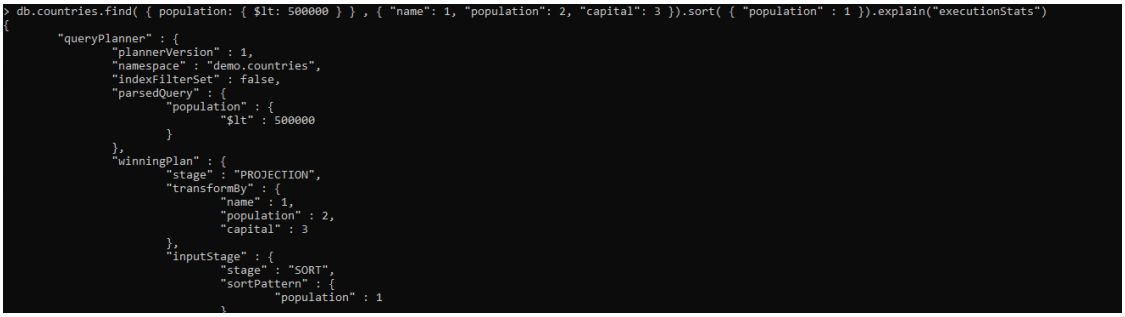
2. Write a MongoDB Query to retrieve Country name, population, and capital for all countries with a population under 500,000 sorted by population.

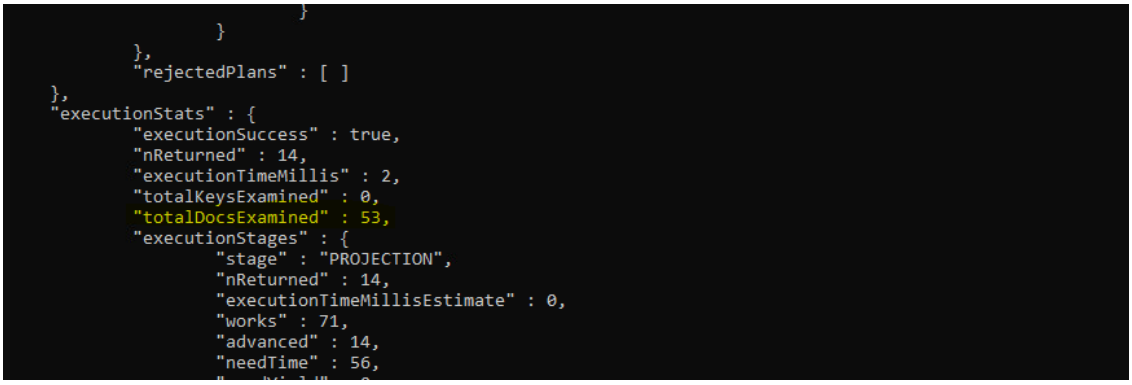
   > db.countries.find({population: { $lt: 500000 } }, { “name”: 1, “population” : 2, “capital” : 3 }).sort({ “population” : 1 })

```
> db.countries.find( { population: { $lt: 500000 } } , { "name": 1, "population": 2, "capital": 3 }).sort( { "population" : 1 })
{ "_id" : ObjectId("612da376727886cd3da2b4c6"), "name" : "Holy See", "capital" : "Rome", "population" : 451 }
{ "_id" : ObjectId("612da376727886cd3da2b4e0"), "name" : "Svalbard and Jan Mayen", "capital" : "Longyearbyen", "population" : 2562 }
{ "_id" : ObjectId("612da376727886cd3da2b4b3"), "name" : "Åland Islands", "capital" : "Mariehamn", "population" : 28875 }
{ "_id" : ObjectId("612da376727886cd3da2b4c3"), "name" : "Gibraltar", "capital" : "Gibraltar", "population" : 32734 }
{ "_id" : ObjectId("612da376727886cd3da2b4dc"), "name" : "San Marino", "capital" : "City of San Marino", "population" : 32831 }
{ "_id" : ObjectId("612da376727886cd3da2b4cd"), "name" : "Liechtenstein", "capital" : "Vaduz", "population" : 37370 }
{ "_id" : ObjectId("612da376727886cd3da2b4d4"), "name" : "Monaco", "capital" : "Monaco", "population" : 37800 }
{ "_id" : ObjectId("612da376727886cd3da2b4be"), "name" : "Faroe Islands", "capital" : "Tórshavn", "population" : 48846 }
{ "_id" : ObjectId("612da376727886cd3da2b4c4"), "name" : "Guernsey", "capital" : "St. Peter Port", "population" : 65150 }
{ "_id" : ObjectId("612da376727886cd3da2b4b2"), "name" : "Andorra", "capital" : "Andorra la Vella", "population" : 76949 }
{ "_id" : ObjectId("612da376727886cd3da2b4c9"), "name" : "Isle of Man", "capital" : "Douglas", "population" : 84497 }
{ "_id" : ObjectId("612da376727886cd3da2b4cb"), "name" : "Jersey", "capital" : "Saint Helier", "population" : 99000 }
{ "_id" : ObjectId("612da376727886cd3da2b4c7"), "name" : "Iceland", "capital" : "Reykjavík", "population" : 330610 }
{ "_id" : ObjectId("612da376727886cd3da2b4d0"), "name" : "Malta", "capital" : "Valletta", "population" : 445426 }
>
```

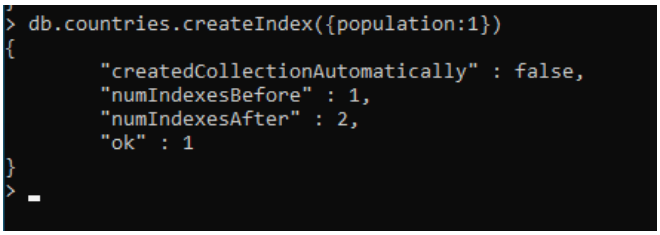
3. Use the**. explain(“executionStats”)** method to analyze the query you wrote in the previous step. Write an index to improve the performance of the query, then perform another explain to demonstrate it worked. Include the code of the index you wrote, the and the relevant output of the execution stats which demonstrate the index is being used.

analyze the query from the previous step
> db.countries.find({population: { $lt: 500000 } }, { “name”: 1, “population” : 2, “capital” : 3 }).sort({ “population” : 1 }).explain(“executionStats”)

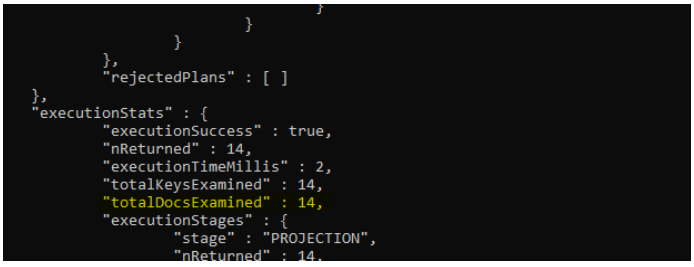




write an index to improve the performance of the query
> db.countries.createIndex({population:1})



run the query again after creating the index and check the performance metrics
> db.countries.find({population: { $lt: 500000 } }, { “name”: 1, “population” : 2, “capital” : 3 }).sort({ “population” : 1 }).explain(“executionStats”)



total docs examined this time 14 vs 53 in the original query

4. Select the most appropriate Redis data structure to store the following information:

Product ID	Name	Qty On Hand	Unit Price
1	Apple	7	2.49
2	Banana	12	1.99
3	Cherry	9	4.99

Execute the commands to store this information in Redis. Make sure to namespace your key and each of the fields should be retrievable under the key used.

create the key values

> mset product1:ProductId 1 product1:Name Apple product1:QtyOnHand 7 product1:UnitPrice 2.49
> mset product2:ProductId 2 product2:Name Banana product2:QtyOnHand 12 product2:UnitPrice 1.99
> mset product3:ProductId 3 product3:Name Cherry product3:QtyOnHand 9 product3:UnitPrice 4.99

show all the keys that were just created

>keys product*:*

```
127.0.0.1:6379> keys product*:*
 1) "product1:Name"
 2) "product1:UnitPrice"
 3) "product3:ProductId"
 4) "product1:QtyOnHand"
 5) "product2:ProductId"
 6) "product3:Name"
 7) "product1:ProductId"
 8) "product2:QtyOnHand"
 9) "product2:UnitPrice"
10) "product3:UnitPrice"
11) "product2:Name"
12) "product3:QtyOnHand"
```

run a couple of mgets to get the key values out

```
127.0.0.1:6379> get product1:Name
"Apple"
127.0.0.1:6379> mget product1:Name product1:UnitPrice
1) "Apple"
2) "2.49"
127.0.0.1:6379>
```

5. Select the most appropriate Redis data structure to store the following information:
The 2018 Golden Snowball Competition for the Upstate NY City with the Highest Snowfall. Scores updated hourly.

City	Syracuse	Rochester	Buffalo
Snowfall Inches	97	68	84

Execute the commands to store this information in Redis. Make sure to namespace your key and each of the snowfall values should be updatable. For example, you should be able to add 10 inches to Buffalo to make it 94. You should be able to display the information upon request.

create the key values
> zadd snowfall:city 97 Syracuse 68 Rochester 85 Buffalo

show that the key values were entered properly
> lrange snowfall:city 0 -1

```
127.0.0.1:6379> zrange snowfall:city 0 -1 withscores
1) "Rochester"
2) "68"
3) "Buffalo"
4) "85"
5) "Syracuse"
6) "97"
127.0.0.1:6379>
```

show that buffalo can be updated to 94
> zadd snowfall:city 94 Buffalo
> zrange snowfall:city 0 -1 withscores

```
127.0.0.1:6379> zrange snowfall:city 0 -1 withscores
1) "Rochester"
2) "68"
3) "Buffalo"
4) "94"
5) "Syracuse"
6) "97"
127.0.0.1:6379>
```