Title: DB Assignment 5
Your Name: Antonio J. Cima
Date: 11/22/2024

*I would like to note the elephant in the room… the time this assignment was submitted. As you know I was unable to complete many tasks while in the emergency room and as such my other professors assigned the missing work on friday, saturday, and sunday, leaving me little time to work on this assignment. I will ask you this in person too, but due to my "unique" circumstances, would it be possible to remove the penalty for the late submission for me? I am essentially asking for a late-extension for this assignment. Thank you for considering my proposal, NOW, Onto the actual assignment.*

**Query #1**

```
Show output from: Debug
Verbose logs are written to:
C:\Users\tonio\AppData\Local\Temp\visualstudio-js-debugger.txt
#1. Data spans from 27 years.
The thread 'app.js [27012]' (0) has exited with code 0 (0x0).
The program 'JavaScript debug adapter: app.js [27012]' has exited with code 4294967295 (0xffffffff).
The program 'app.js' has exited with code 4294967295 (0xffffffff).
```

Some notes about MongoDB, I like this a lot better than mySQL already. This just makes sense to me, and in all honesty is easier to grasp in my own opinion. There are less steps to calculate data and find it too. Now, onto the actual Query itself, for Query 1, we had to find how many years the data has been collect from, we do this by using aggregate(), which is essentially a find method which allows us to search through groups of fields in the collection with different filters. In this, we use min and max for year to then get the difference which leaves us with our data range! You will note when we use aggregate(), we have this ID field, aggregate() requires an ID, even if it is null to work, so you will be seeing that in future queries.

**Query #2**

```
Verbose logs are written to:
C:\Users\tonio\AppData\Local\Temp\visualstudio-js-debugger.txt
#2. Data spans from 47 distinct states and 885548 total states.
The thread 'app.js [40612]' (0) has exited with code 0 (0x0).
The program 'JavaScript debug adapter: app.js [40612]' has exited with code 4294967295 (0xffffffff).
The program 'app.js' has exited with code 4294967295 (0xffffffff).
```

For this query, we were tasked with finding the amount of states that were seen in this dataset, I was unsure wether we were asking for unique or total states, so I just did both. I accomplished this by using two different queries, first I simply used the distinct() method to get the unique states, and then I used countDocuments(). countDocuments() is similar to aggregate but it doesn't require an ID and simply counts the number of results. I combined these answers into 1 string and got our result.

**Query #3**

```
Verbose logs are written to:
C:\Users\tonio\AppData\Local\Temp\visualstudio-js-debugger.txt
#3. The mystery query is: 657
The thread 'app.js [4324]' (0) has exited with code 0 (0x0).
The program 'JavaScript debug adapter: app.js [4324]' has exited with code 4294967295 (0xffffffff).
The program 'app.js' has exited with code 4294967295 (0xffffffff).
```

For this query, we were given a **MYSTERY?** We were given this query to run: db.unemployment.find({Rate : {$lt: 1.0}}).count(), and run it I did… at least that's what I wish I could say; for some reason when I put this into my javaScript code, it did not work, so I reworked the query to better fit my system, keeping the integrity of the original query. I use countDocuments() to replace the find() and count() methods entirely; then I put in the original query of Rate: {$lt: 1.0}}) and we get 657. Now with my GREAT detective skills, I can figure out that we are counting the amount of unemployment rates that are lower than 1%.

**Query #4**

```
Verbose logs are written to:
C:\Users\tonio\AppData\Local\Temp\visualstudio-js-debugger.txt
#4. There are 91430 amount of counties with unemployment higher than 10%
The thread 'app.js [34988]' (0) has exited with code 0 (0x0).
The program 'JavaScript debug adapter: app.js [34988]' has exited with code 4294967295 (0xffffffff).
The program 'app.js' has exited with code 4294967295 (0xffffffff).
```

For this query, we needed to find all counties that had a higher unemployment rate than 10%. So I know for this query it asks for all counties and not the count, but realistically, 91,340 counties are not fitting on this page alone, so I opted for the more manageable result. For this we use the find() method; this method is just a method that allows us to search using only 1 condition.

**Query #5**

```
Verbose logs are written to:
C:\Users\tonio\AppData\Local\Temp\visualstudio-js-debugger.txt
#5. The average unemployment rate across all states is: 6.1750097115006755%
The thread 'app.js [22176]' (0) has exited with code 0 (0x0).
The program 'JavaScript debug adapter: app.js [22176]' has exited with code 4294967295 (0xffffffff).
The program 'app.js' has exited with code 4294967295 (0xffffffff).
```

So for this query we are tasked with finding the average rate of unemployment throughout the states; I was a bit confused about this query, as I was unsure if I was supposed to find each average for each state, or just the states in average. I opted for the ladder, using the ambiguity to my advantage in order to save myself some time. I use the aggregate() method here instead of find() method just because for some reason find() was giving me issues, but when I switched to aggregate() it seemed to function correctly. We use keyword avg to get the average Rate, which saves us a lot of time.

**Query #6**

```
Verbose logs are written to:
C:\Users\tonio\AppData\Local\Temp\visualstudio-js-debugger.txt
#6. There are 310792 amount of counties with unemployment higher than 5% and lower than 8%
The thread 'app.js [9376]' (0) has exited with code 0 (0x0).
The program 'JavaScript debug adapter: app.js [9376]' has exited with code 4294967295 (0xffffffff).
The program 'app.js' has exited with code 4294967295 (0xffffffff).
```

For query #6 we needed to do something very similar to query #4, but instead of going above 10%, we needed to stay within the ranges of 5% and 8%. We do this by using the find() method and using the keywords lt and gt to then filter counties between this range (lt meaning less than and gt meaning greater than)

**Query #7**

```
Verbose logs are written to:
C:\Users\tonio\AppData\Local\Temp\visualstudio-js-debugger.txt
#7. The state with the highest unemployment rate is Colorado with the rate being: 58.4%
The thread 'app.js [33148]' (0) has exited with code 0 (0x0).
The program 'JavaScript debug adapter: app.js [33148]' has exited with code 4294967295 (0xffffffff).
The program 'app.js' has exited with code 4294967295 (0xffffffff).
```

Query #7, all we have to do is find the state with the HIGHEST unemployment rate. We do this by using aggregate(). (Also at this point I know what the difference is between aggregate() and find(). aggregate() is mainly used for grouping and find() is used for gathering static information.) Anyways, we use aggregate() to sort the grouped data by descending order (highest first) and then limit it by 1 to get our answer… COLORADO!

**Query #8**

```
Verbose logs are written to:
C:\Users\tonio\AppData\Local\Temp\visualstudio-js-debugger.txt
#8. There are 510173 amount of counties with unemployment higher than 5%
The thread 'app.js [28964]' (0) has exited with code 0 (0x0).
The program 'JavaScript debug adapter: app.js [28964]' has exited with code 4294967295 (0xffffffff).
The program 'app.js' has exited with code 4294967295 (0xffffffff).
```

Query #8 requires us to do basically query 4 again but search for all that are higher than 5%. We adjust numbers and it works, not really much to discuss here! 😁

**Query #9**

```
State: Montana, Year: 2015, Average Rate: 4.2889880952380945%
State: Nebraska, Year: 2015, Average Rate: 2.8778673835125446%
State: Nevada, Year: 2015, Average Rate: 6.861274509803922%
State: New Hampshire, Year: 2015, Average Rate: 3.4225%
State: New Jersey, Year: 2015, Average Rate: 6.051587301587301%
State: New Mexico, Year: 2015, Average Rate: 7.243434343434344%
State: New York, Year: 2015, Average Rate: 5.582258064516129%
State: North Carolina, Year: 2015, Average Rate: 6.4425%
State: North Dakota, Year: 2015, Average Rate: 3.2528301886792454%
State: Ohio, Year: 2015, Average Rate: 5.314583333333333%
State: Oklahoma, Year: 2015, Average Rate: 4.746536796536796%
State: Oregon, Year: 2015, Average Rate: 6.551620370370371%
State: Pennsylvania, Year: 2015, Average Rate: 5.472636815920398%
State: Rhode Island, Year: 2015, Average Rate: 5.528333333333333%
State: South Carolina, Year: 2015, Average Rate: 7.039673913043479%
State: South Dakota, Year: 2015, Average Rate: 3.656439393939394%
State: Tennessee, Year: 2015, Average Rate: 6.737982456140351%
State: Texas, Year: 2015, Average Rate: 4.775262467191601%
State: Utah, Year: 2015, Average Rate: 4.798563218390805%
State: Vermont, Year: 2015, Average Rate: 4.214880952380953%
State: Virginia, Year: 2015, Average Rate: 5.140664160401003%
State: Washington, Year: 2015, Average Rate: 6.835683760683761%
State: West Virginia, Year: 2015, Average Rate: 7.743787878787878%
State: Wisconsin, Year: 2015, Average Rate: 5.12662037037037%
State: Wyoming, Year: 2015, Average Rate: 4.068840579710145%
State: Alabama, Year: 2016, Average Rate: 6.773756218905473%
State: Arizona, Year: 2016, Average Rate: 7.732777777777779%
State: Arkansas, Year: 2016, Average Rate: 4.656666666666666%
State: California, Year: 2016, Average Rate: 6.920219435736677%
State: Colorado, Year: 2016, Average Rate: 3.416145833333333%
State: Connecticut, Year: 2016, Average Rate: 5.128125%
State: Delaware, Year: 2016, Average Rate: 4.430555555555555%
State: Hawaii, Year: 2016, Average Rate: 3.4145833333333333%
State: Idaho, Year: 2016, Average Rate: 4.36780303030303%
State: Illinois, Year: 2016, Average Rate: 6.280800653594771%
State: Indiana, Year: 2016, Average Rate: 4.615851449275362%
State: Iowa, Year: 2016, Average Rate: 3.8750841750841754%
State: Kansas, Year: 2016, Average Rate: 3.9787301587301585%
State: Kentucky, Year: 2016, Average Rate: 6.358125%
State: Louisiana, Year: 2016, Average Rate: 7.2244140625%
State: Maine, Year: 2016, Average Rate: 4.2437499999999995%
State: Maryland, Year: 2016, Average Rate: 4.909027777777777%
State: Massachusetts, Year: 2016, Average Rate: 4.140476190476191%
State: Michigan, Year: 2016, Average Rate: 5.566566265060241%
State: Minnesota, Year: 2016, Average Rate: 4.571264367816092%
```

Lastly Query 9… Query 9 is a- well a lot. It's every state for every year's average. So, the list is WAY too big to display so you'll have to take this cut off screenshot and my word as law here. So… the explanation of how I did this! We're again using aggregate() to group, and here is when I find out we can use the ID to group the fields State and Year together! We then get the average unemployment rate for that state for that year, and then we put it into a list which then we, well, list!

That is all, thank you!