Title: DB Assignment 5

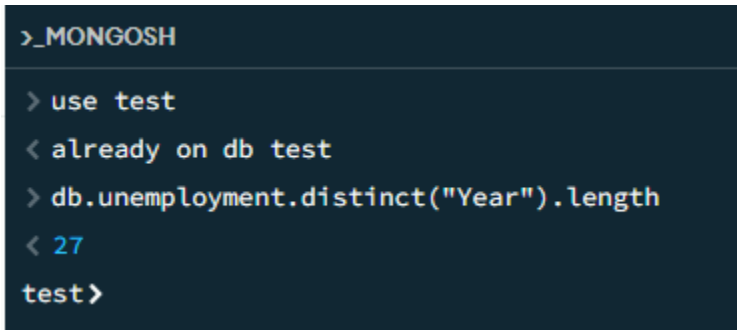
Name: Aidan Elm

Date: 2024-11-21

Question 1

db.unemployment.distinct("Year").length

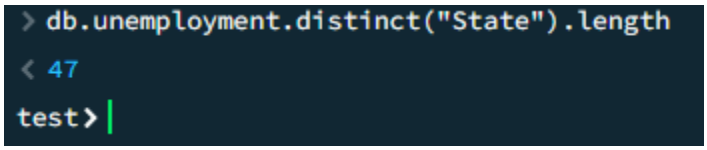
In this query, we use distinct to get unique years, then length to count them.



Question 2

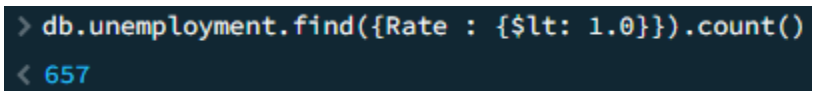
db.unemployment.distinct("State").length

The same as above, except with states instead of years.



Question 3

Answer is 657.



Question 4

db.unemployment.find({Rate: {$gt: 10.0}}, {County: 1})

Here, we use find with a condition for an unemployment rate greater than 10%.

```
> db.unemployment.find({Rate: {$gt: 10.0}}, {County: 1})
< {
    _id: ObjectId('673fc91ee33023d74b31db62'),
    County: 'Kemper County'
  }
  {
    _id: ObjectId('673fc91ee33023d74b31db65'),
    County: 'Jefferson County'
  }
  {
    _id: ObjectId('673fc91ee33023d74b31db67'),
    County: 'Sharkey County'
  }
  {
    _id: ObjectId('673fc91ee33023d74b31db68'),
    County: 'Tunica County'
  }
  {
    _id: ObjectId('673fc91ee33023d74b31db6d'),
    County: 'Noxubee County'
```

Question 5

db.unemployment.aggregate([{$group: {averageRate: {$avg: "$Rate"}, _id: null}}])

In this query, we use aggregate with group to calculate the average rate across all records using $avg.

```
> db.unemployment.aggregate([{$group: {averageRate: {$avg: "$Rate"}, _id: null}}])
< {
    _id: null,
    averageRate: 6.1750097115006755
  }
```

Question 6

db.unemployment.find({Rate: {$gte: 5.0, $lte: 8.0}}, {County: 1})

Similar to Question 4, but using both greater than or equal to and less than or equal to for a range between 5% and 8%.

```
> db.unemployment.find({Rate: {$gte: 5.0, $lte: 8.0}}, {County: 1})
< {
    _id: ObjectId('673fc91ee33023d74b31db5e'),
    County: 'Newton County'
  }
  {
    _id: ObjectId('673fc91ee33023d74b31db60'),
    County: 'Monroe County'
  }
  {
    _id: ObjectId('673fc91ee33023d74b31db61'),
    County: 'Hinds County'
  }
  {
    _id: ObjectId('673fc91ee33023d74b31db63'),
    County: 'Calhoun County'
  }
  {
    _id: ObjectId('673fc91ee33023d74b31db64'),
    County: 'Clarke County'
```

Question 7

db.unemployment.aggregate([{$group: {_id: "$State", avgRate: {$avg: "$Rate"}}}, {$sort: {avgRate: -1}}, {$limit: 1}])

Here, we group by state to get average rates, sort in descending order, and limit to 1 to get the highest.

```
> db.unemployment.aggregate([{$group: {_id: "$State", avgRate: {$avg: "$Rate"}}}, {$sort: {avgRate: -1}}, {$limit: 1}])
< {
    _id: 'Arizona',
    avgRate: 9.274588477366255
  }
test>
```

Question 8

db.unemployment.find({Rate: {$gt: 5.0}}, {County: 1}).count()

Similar to Question 4, but we add count() to get the total number of matching records.

```
> db.unemployment.find({Rate: {$gt: 5.0}}, {County: 1}).count()
< 510173
```

Question 9

db.unemployment.aggregate([{$group: {_id: {state: "$State", year: "$Year"}, avgRate: {$avg: "$Rate"}}}])

In this query, we group by both state and year together and then get average rates.

```
> db.unemployment.aggregate([{$group: {_id: {state: "$State", year: "$Year"}, avgRate: {$avg: "$Rate"}}}])
< {
    _id: {
      state: 'Indiana',
      year: 1998
    },
    avgRate: 3.4572463768115944
  }
  {
    _id: {
      state: 'Washington',
      year: 2011
    },
    avgRate: 10.351495726495726
  }
  {
    _id: {
```

Question 10

db.unemployment.aggregate([{$group: {_id: "$State", totalRate: {$sum: "$Rate"}}}])

Here, we group by state and use $sum instead of $avg to get the total of all rates.

```
> db.unemployment.aggregate([{$group: {_id: "$State", totalRate: {$sum: "$Rate"}}}])
< {
    _id: 'Wisconsin',
    totalRate: 135667.7
  }
  {
    _id: 'Nebraska',
    totalRate: 93707.6
  }
  {
    _id: 'Maine',
    totalRate: 32472.5
  }
  {
    _id: 'Minnesota',
    totalRate: 152320.9
  }
  {
    _id: 'Montana',
    totalRate: 96261.5
  }
```

Question 11

db.unemployment.aggregate([{$match: {Year: {$gte: 2015}}}, {$group: {_id: "$State", totalRate: {$sum: "$Rate"}}}])

The same as Question 10, but we add a match stage first to filter for years 2015 and after.

```
> db.unemployment.aggregate([{$match: {Year: {$gte: 2015}}}, {$group: {_id: "$State", totalRate: {$sum: "$Rate"}}}])
< {
    _id: 'Nevada',
    totalRate: 2612.7
  }
  {
    _id: 'Minnesota',
    totalRate: 9315.1
  }
  {
    _id: 'Maine',
    totalRate: 1758.8
  }
  {
    _id: 'Wisconsin',
    totalRate: 8482.5
  }
  {
    _id: 'Nebraska',
    totalRate: 6656.4
  }
```