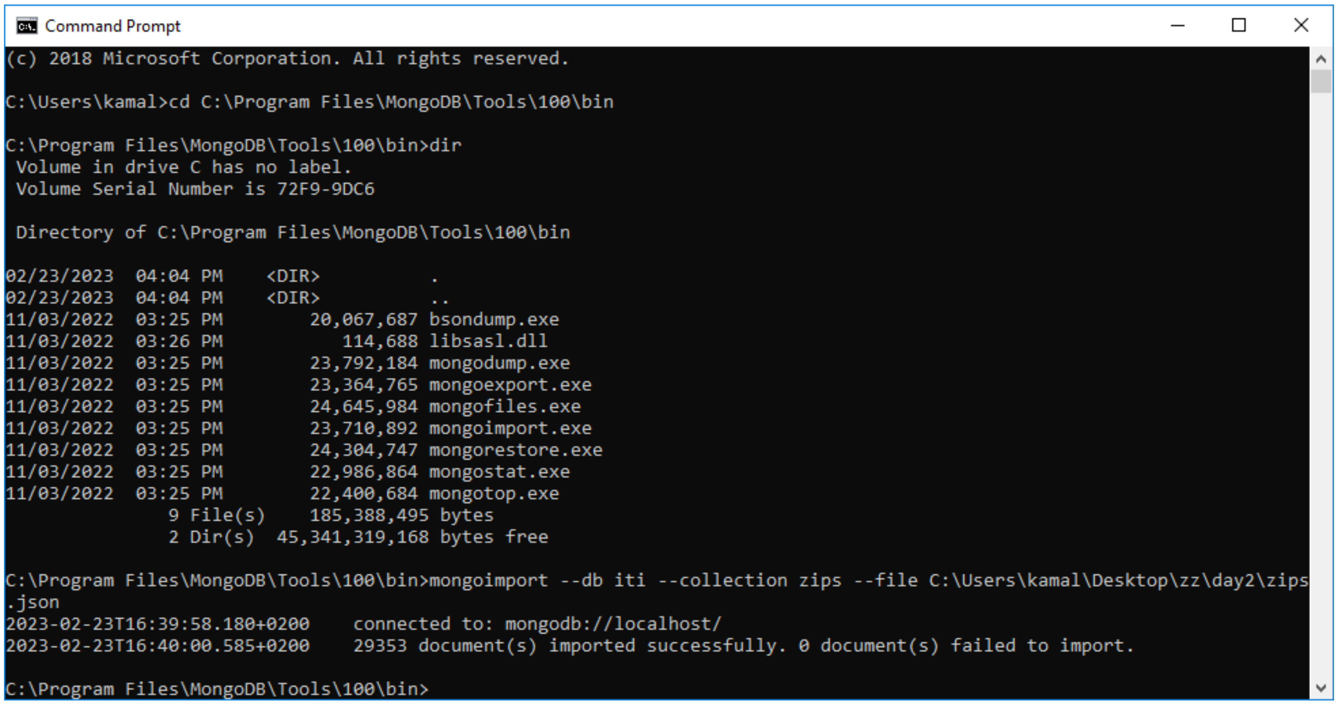
MongoDB Lab2

1 - Download the following json file and import it into a collection named “zips” into “iti” database

mongoimport --db iti --collection zips –file C:\Users\kamal\Desktop\zz\day2\zips.json

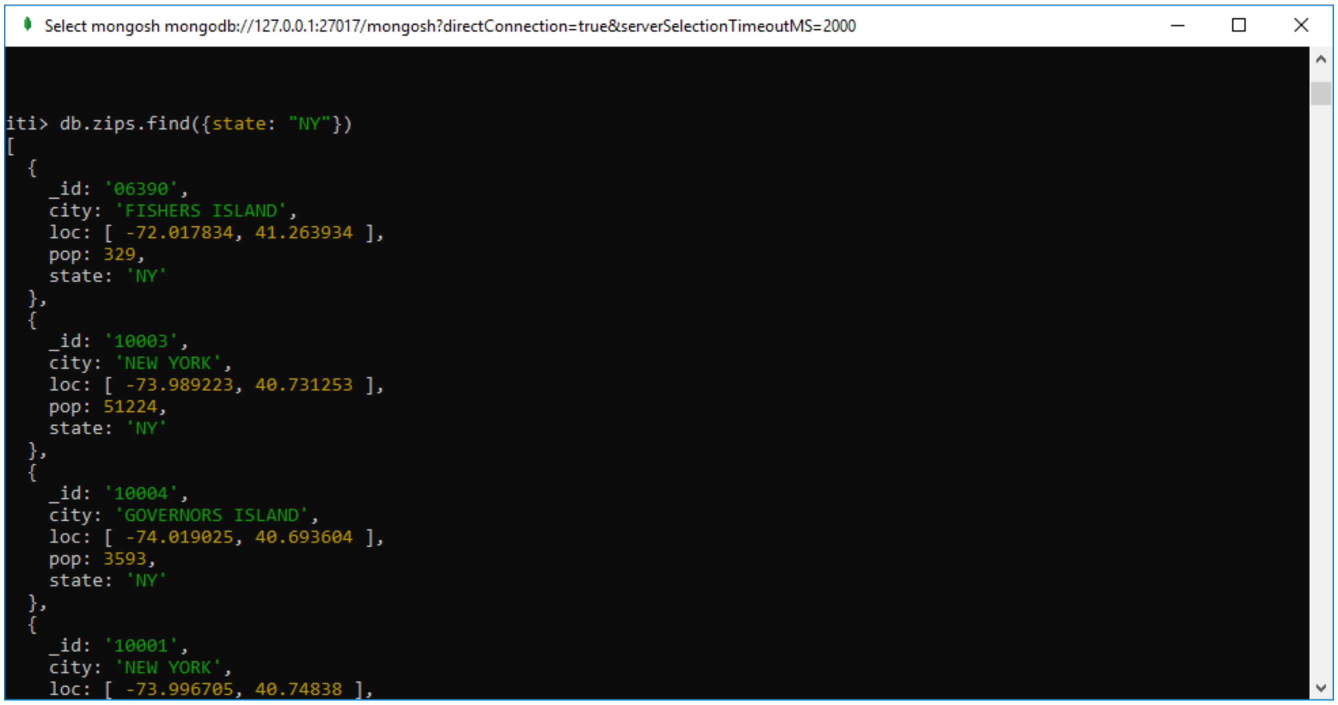


Command Prompt

```
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\kamal>cd C:\Program Files\MongoDB\Tools\100\bin

C:\Program Files\MongoDB\Tools\100\bin>dir
 Volume in drive C has no label.
 Volume Serial Number is 72F9-9DC6

 Directory of C:\Program Files\MongoDB\Tools\100\bin

02/23/2023  04:04 PM    <DIR>          .
02/23/2023  04:04 PM    <DIR>          ..
11/03/2022  03:25 PM        20,067,687 bsondump.exe
11/03/2022  03:26 PM           114,688 libsasl.dll
11/03/2022  03:25 PM        23,792,184 mongodump.exe
11/03/2022  03:25 PM        23,364,765 mongoexport.exe
11/03/2022  03:25 PM        24,645,984 mongofiles.exe
11/03/2022  03:25 PM        23,710,892 mongoimport.exe
11/03/2022  03:25 PM        24,304,747 mongorestore.exe
11/03/2022  03:25 PM        22,986,864 mongostat.exe
11/03/2022  03:25 PM        22,400,684 mongotop.exe
               9 File(s)    185,388,495 bytes
               2 Dir(s)  45,341,319,168 bytes free

C:\Program Files\MongoDB\Tools\100\bin>mongoimport --db iti --collection zips --file C:\Users\kamal\Desktop\zz\day2\zips
.json
2023-02-23T16:39:58.180+0200    connected to: mongodb://localhost/
2023-02-23T16:40:00.585+0200    29353 document(s) imported successfully. 0 document(s) failed to import.

C:\Program Files\MongoDB\Tools\100\bin>
```

2 – find all documents which contains data related to “NY” state

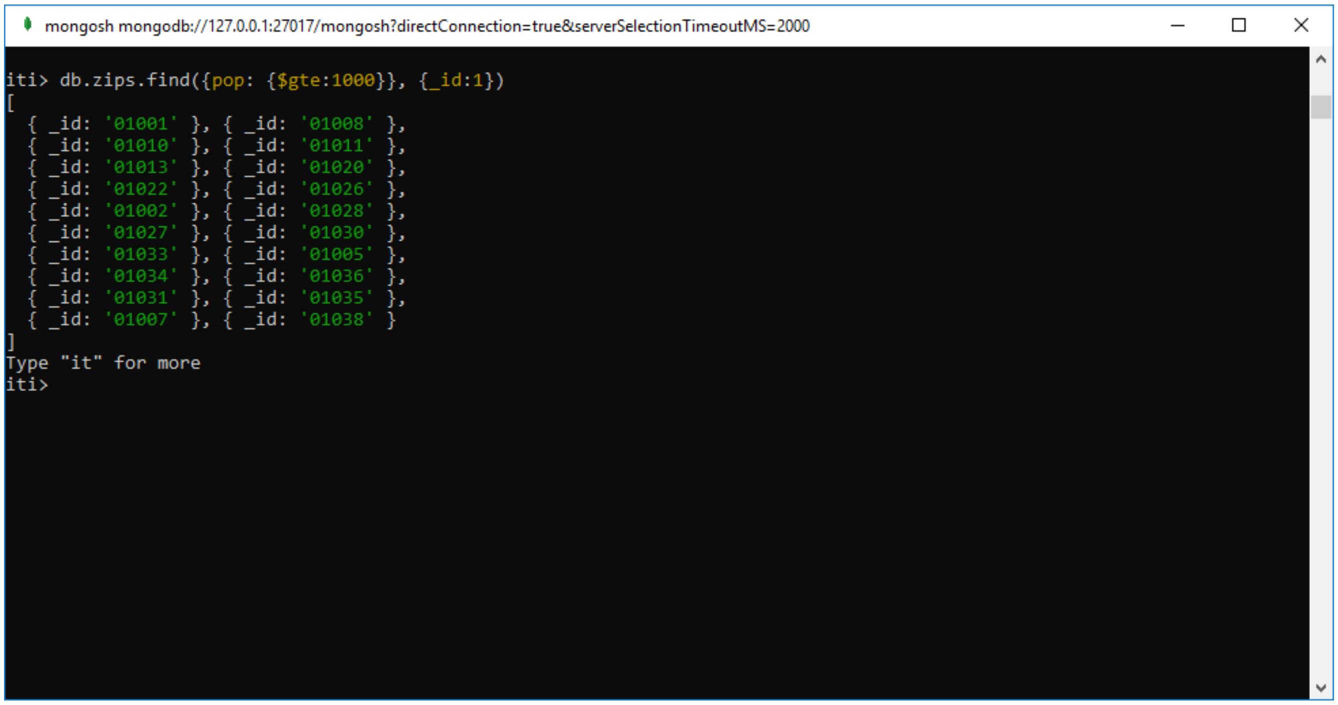
db.zips.find({state: "NY"})


Select mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.find({state: "NY"})
[
  {
    _id: '06390',
    city: 'FISHERS ISLAND',
    loc: [ -72.017834, 41.263934 ],
    pop: 329,
    state: 'NY'
  },
  {
    _id: '10003',
    city: 'NEW YORK',
    loc: [ -73.989223, 40.731253 ],
    pop: 51224,
    state: 'NY'
  },
  {
    _id: '10004',
    city: 'GOVERNORS ISLAND',
    loc: [ -74.019025, 40.693604 ],
    pop: 3593,
    state: 'NY'
  },
  {
    _id: '10001',
    city: 'NEW YORK',
    loc: [ -73.996705, 40.74838 ],
```

3 – find <u>all zip codes</u> whose population is <u>greater than or equal to 1000</u>

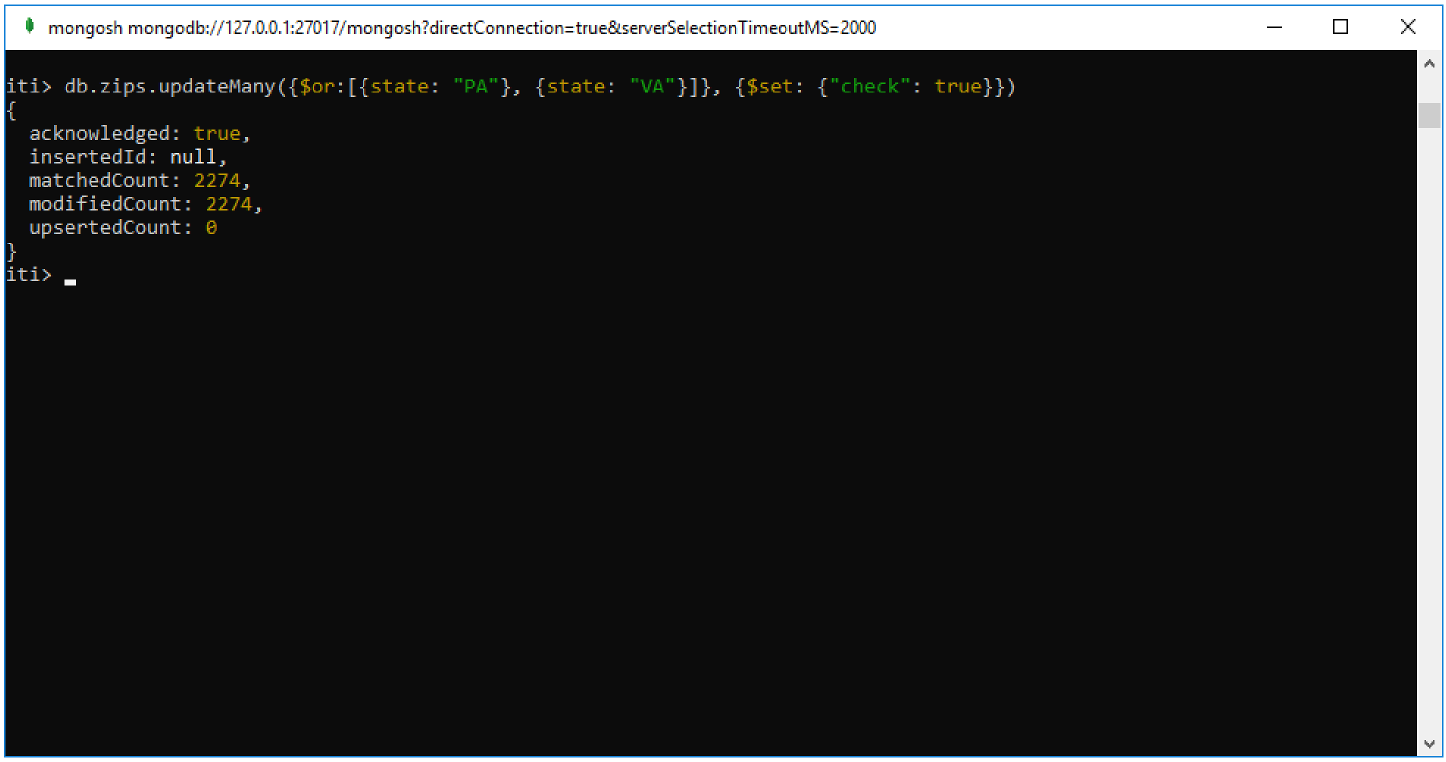
db.zips.find({pop:{$gte:1000}},{_id:1})


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.find({pop: {$gte:1000}}, {_id:1})
[
  { _id: '01001' }, { _id: '01008' },
  { _id: '01010' }, { _id: '01011' },
  { _id: '01013' }, { _id: '01020' },
  { _id: '01022' }, { _id: '01026' },
  { _id: '01002' }, { _id: '01028' },
  { _id: '01027' }, { _id: '01030' },
  { _id: '01033' }, { _id: '01005' },
  { _id: '01034' }, { _id: '01036' },
  { _id: '01031' }, { _id: '01035' },
  { _id: '01007' }, { _id: '01038' }
]
Type "it" for more
iti>
```

4 – add a new boolean field called “check” and set its value to true for “PA” and “VA” state

```
db.zips.updateMany({},{$set: {"check": false}})
db.zips.updateMany({$or:[{state: "PA"}, {state: "VA"}]}, {$set: {"check": true}})
```

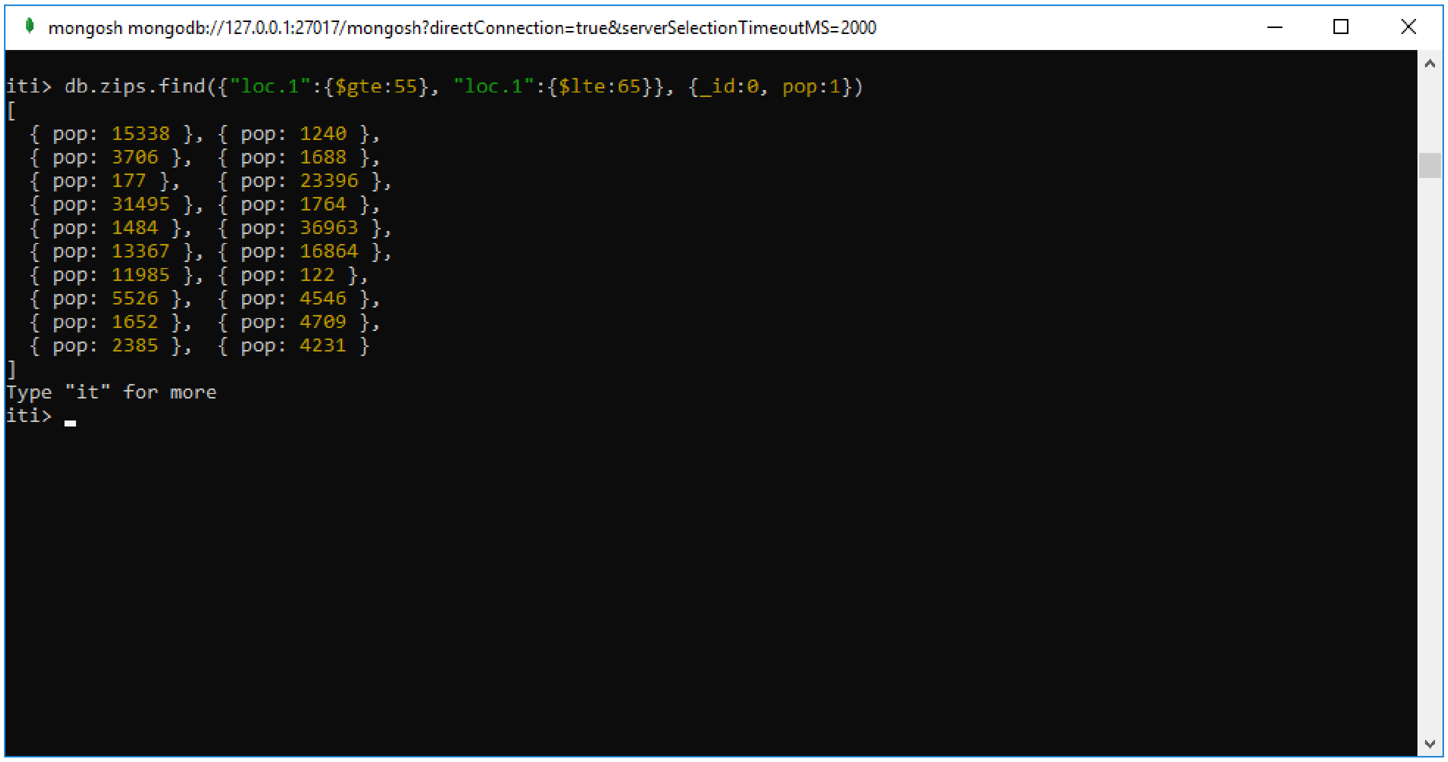


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.updateMany({$or:[{state: "PA"}, {state: "VA"}]}, {$set: {"check": true}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2274,
  modifiedCount: 2274,
  upsertedCount: 0
}
iti>
```

5 – using zip codes find all cities whose latitude is between 55 and 65 and show the population only.

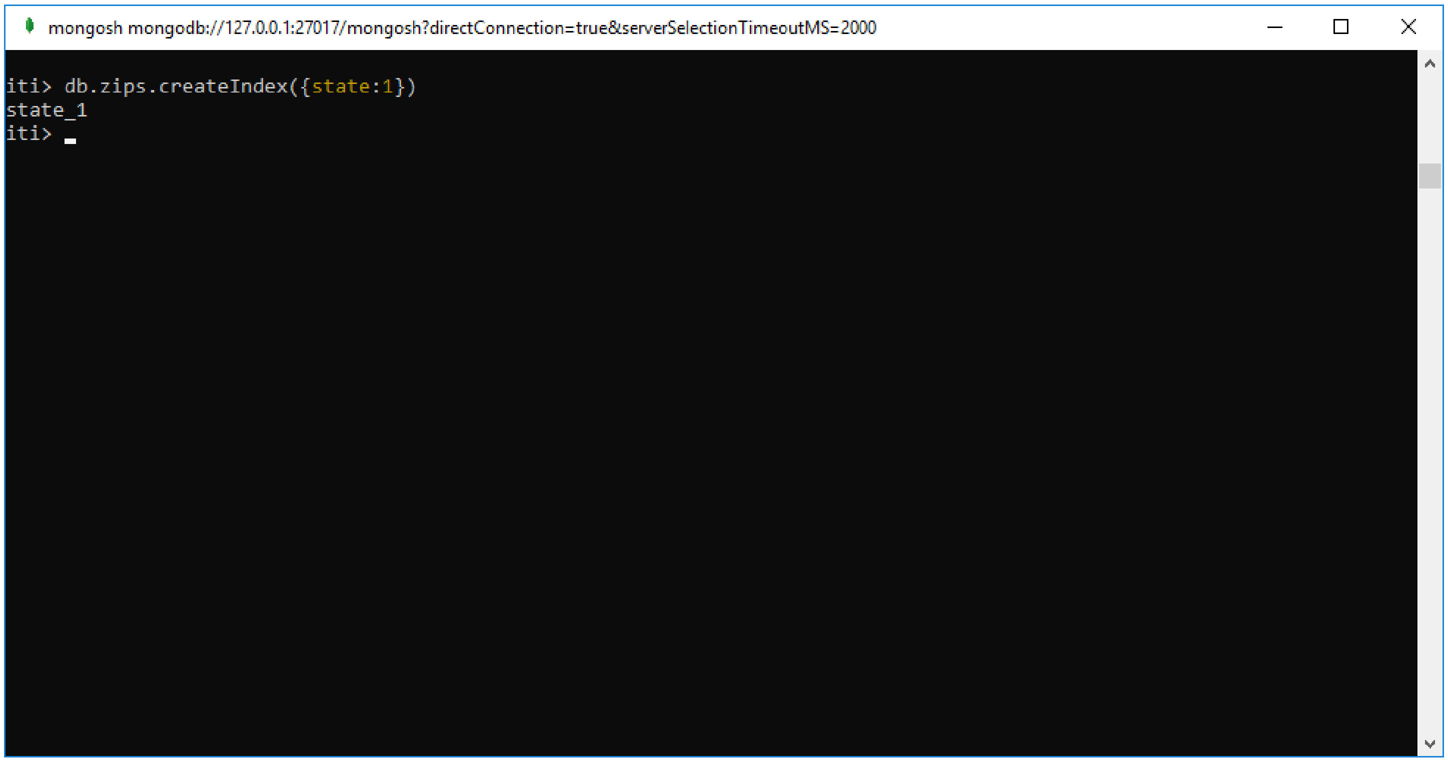
```
db.zips.find({"loc.1":{$gte:55}, "loc.1":{$lte:65}}, {_id:0, pop:1})
```


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.find({"loc.1":{$gte:55}, "loc.1":{$lte:65}}, {_id:0, pop:1})
[
  { pop: 15338 }, { pop: 1240 },
  { pop: 3706 },  { pop: 1688 },
  { pop: 177 },   { pop: 23396 },
  { pop: 31495 }, { pop: 1764 },
  { pop: 1484 },  { pop: 36963 },
  { pop: 13367 }, { pop: 16864 },
  { pop: 11985 }, { pop: 122 },
  { pop: 5526 },  { pop: 4546 },
  { pop: 1652 },  { pop: 4709 },
  { pop: 2385 },  { pop: 4231 }
]
Type "it" for more
iti> 
```

6 – create index for states to be able to select it quickly and check any query explain using the index only.

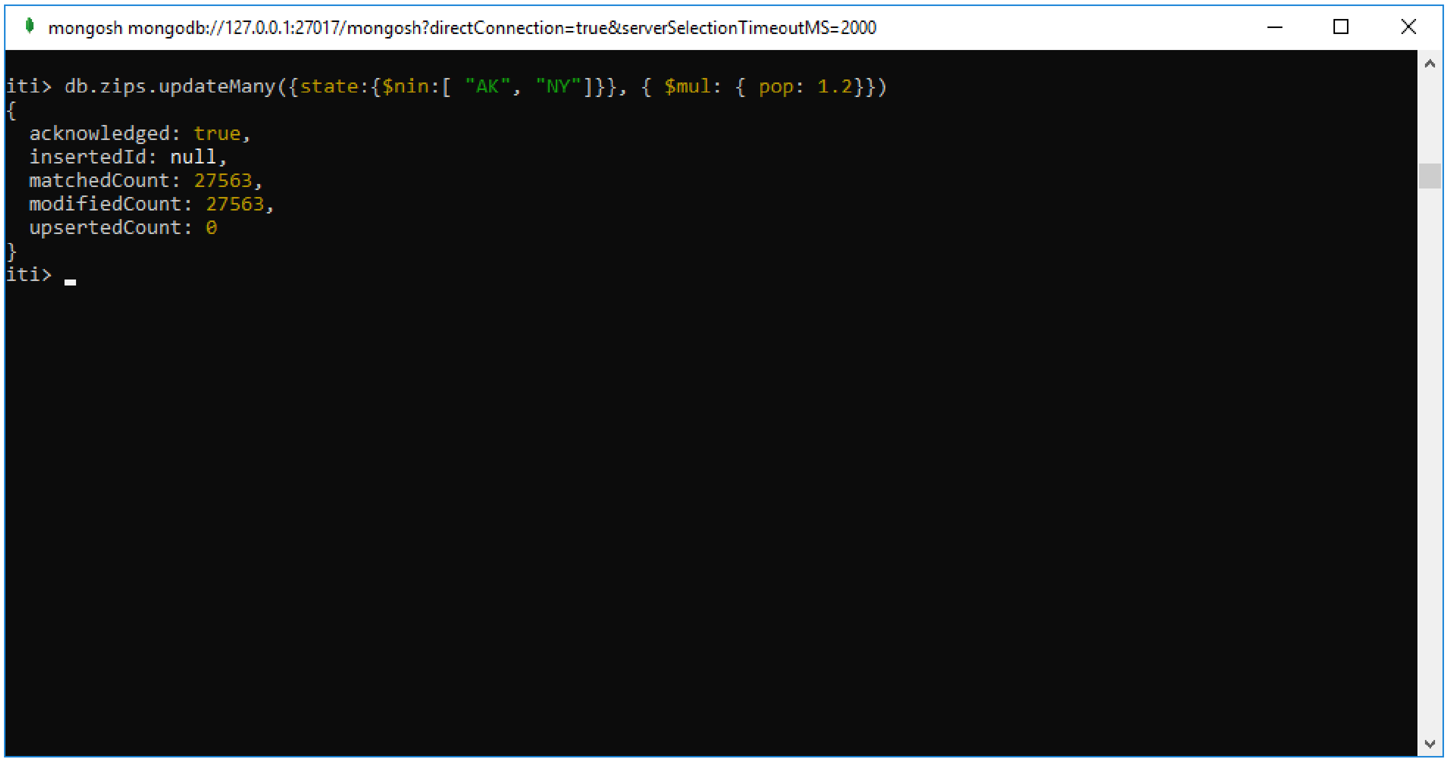
db.zips.createIndex({state:1})


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.createIndex({state:1})
state_1
iti> _
```

7 – increase the population by 0.2 for all cities which doesn't located in “AK” nor “NY”

db.zips.updateMany({state:{$nin:["AK", "NY"]}}, { $mul: { pop: 1.2}})

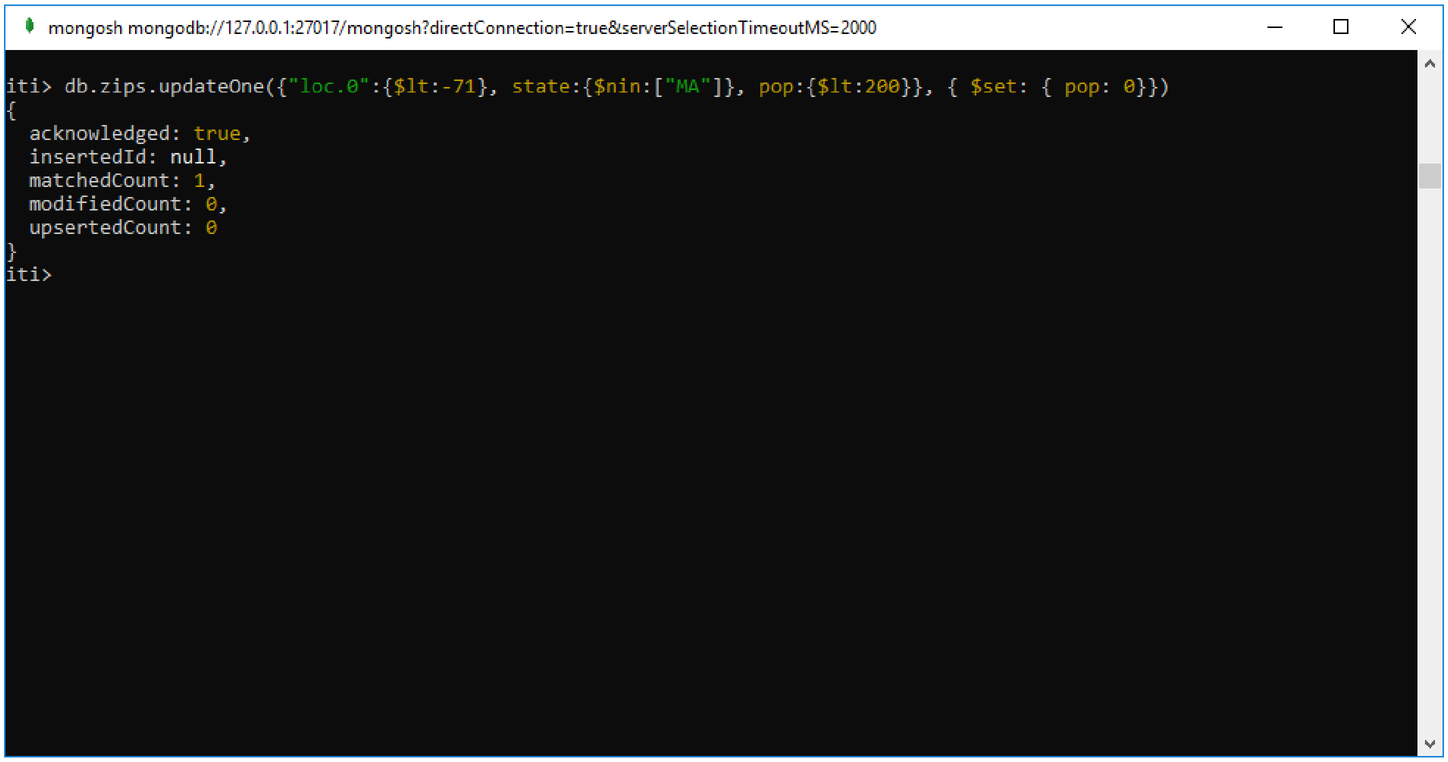


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.updateMany({state:{$nin:[ "AK", "NY"]}}, { $mul: { pop: 1.2}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 27563,
  modifiedCount: 27563,
  upsertedCount: 0
}
iti> _
```

8 – update only one city whose longitude is lower than -71 and is not located in “MA” state, set its population to 0 if zipcode population less than 200.

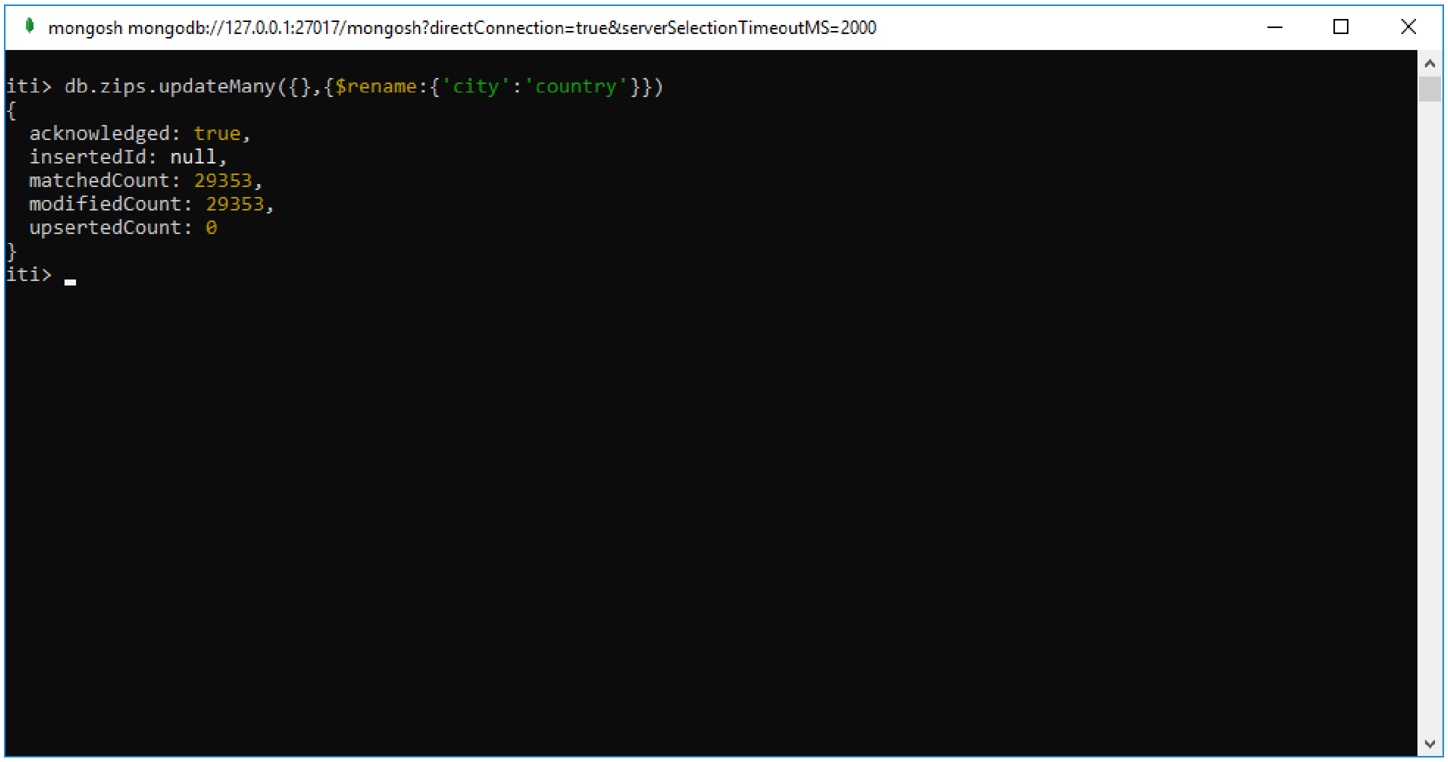
```
db.zips.update({"loc.0":{$lt:-71}, state:{$nin:["MA"]}, pop:{$lt:200}}, { $set: { pop: 0}})
```


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.updateOne({"loc.0":{$lt:-71}, state:{$nin:["MA"]}, pop:{$lt:200}}, { $set: { pop: 0}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
iti>
```

9 – update all documents whose city field is a string, rename its city field to be country and if there isn't any, add new document the same as the first document in the database but change the _id to avoid duplications.

db.zips.updateMany({},{$rename:{'city':'country'}})



mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.updateMany({},{$rename:{'city':'country'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 29353,
  modifiedCount: 29353,
  upsertedCount: 0
}
iti> _
```

Hint: use Variables

****

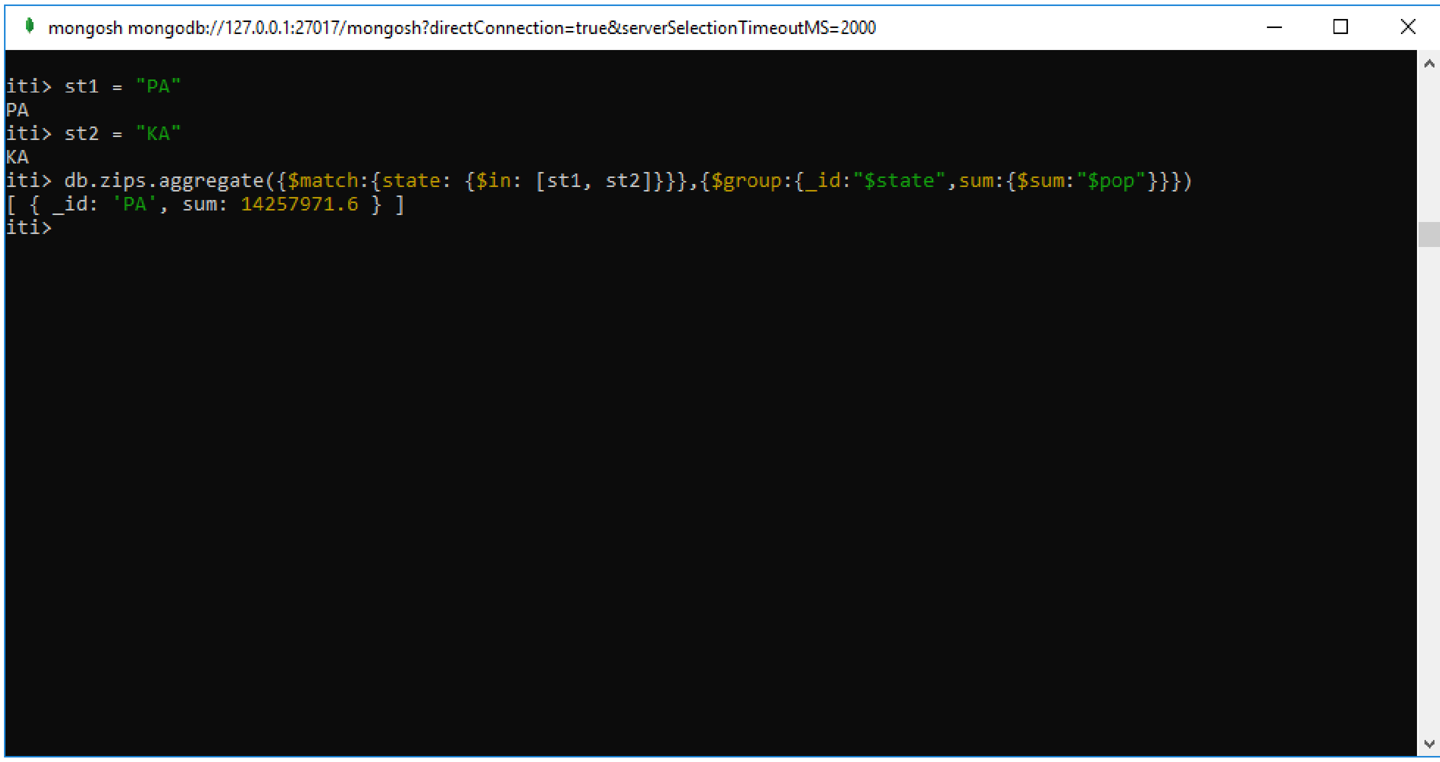
part2

1. Get sum of population that state in PA, KA

   st1 = "PA"

   st2 = "KA"

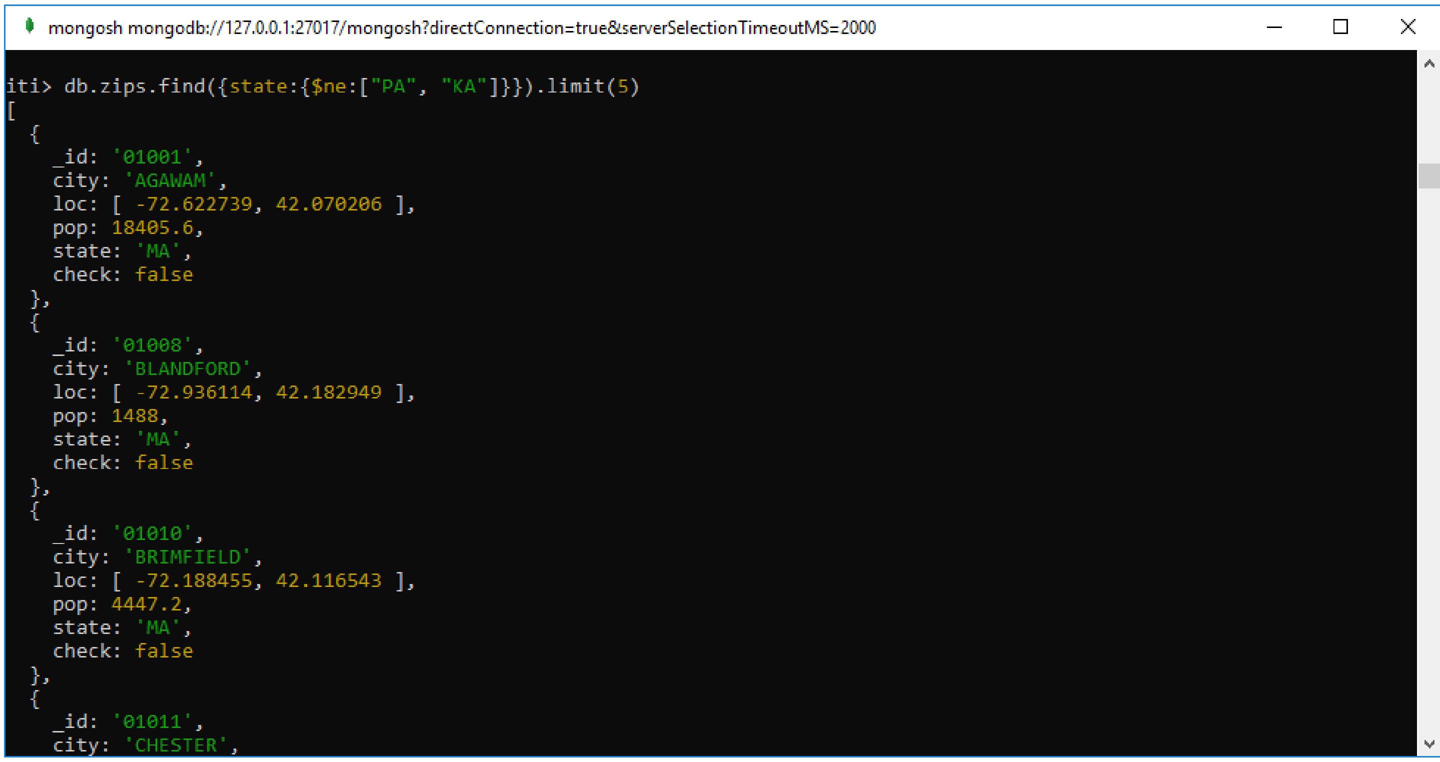
db.zips.aggregate({$match:{state: {$in: [st1, st2]}}},{$group:{_id:"$state",sum:{$sum:"$pop"}}})


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> st1 = "PA"
PA
iti> st2 = "KA"
KA
iti> db.zips.aggregate({$match:{state: {$in: [st1, st2]}}},{$group:{_id:"$state",sum:{$sum:"$pop"}}})
[ { _id: 'PA', sum: 14257971.6 } ]
iti>
```

2. Get only 5 documents that state not equal to PA, KA

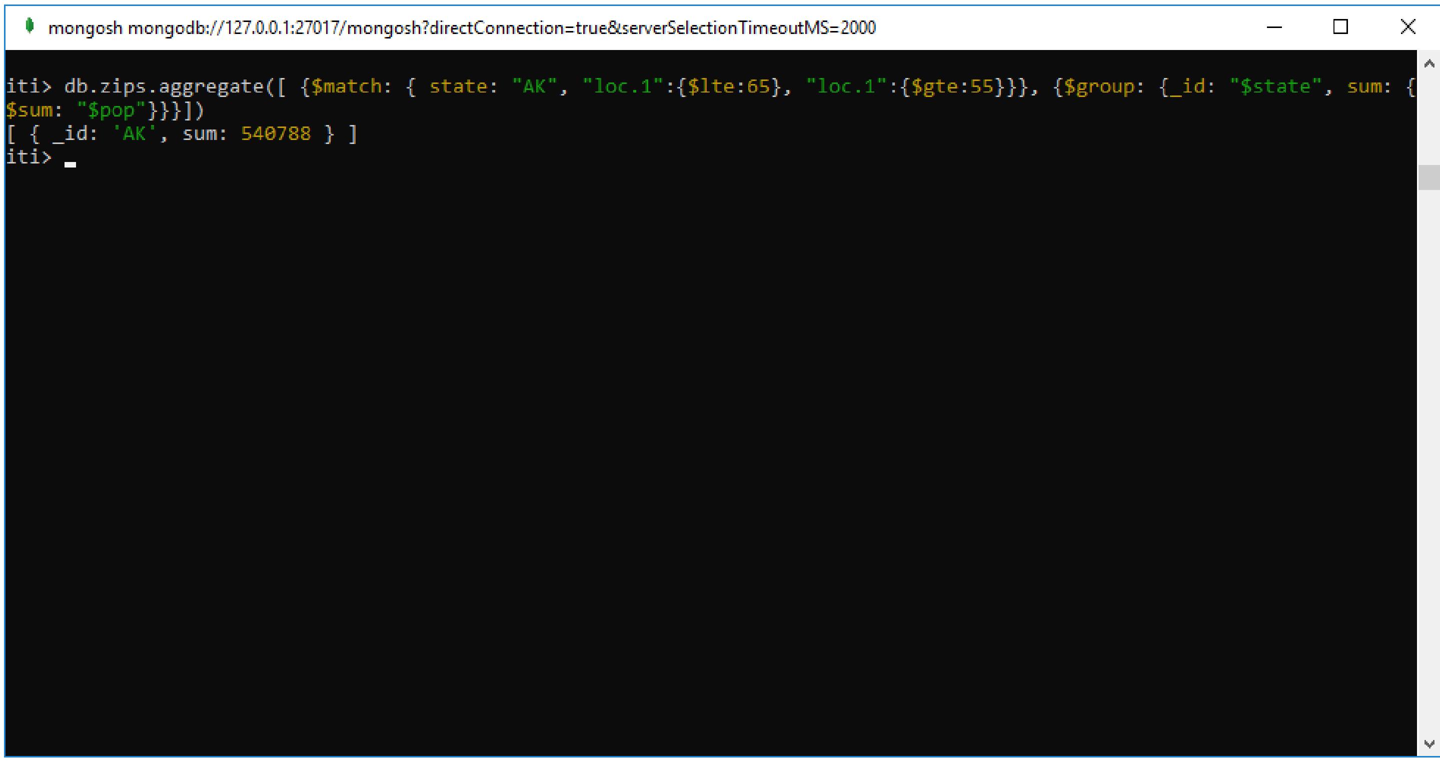
db.zips.find({state:{$ne:["PA", "KA"]}}).limit(5)


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.find({state:{$ne:["PA", "KA"]}}).limit(5)
[
  {
    _id: '01001',
    city: 'AGAWAM',
    loc: [ -72.622739, 42.070206 ],
    pop: 18405.6,
    state: 'MA',
    check: false
  },
  {
    _id: '01008',
    city: 'BLANDFORD',
    loc: [ -72.936114, 42.182949 ],
    pop: 1488,
    state: 'MA',
    check: false
  },
  {
    _id: '01010',
    city: 'BRIMFIELD',
    loc: [ -72.188455, 42.116543 ],
    pop: 4447.2,
    state: 'MA',
    check: false
  },
  {
    _id: '01011',
    city: 'CHESTER',
```

3. Get sum of population that state equal to AK and their latitude between 55, 65

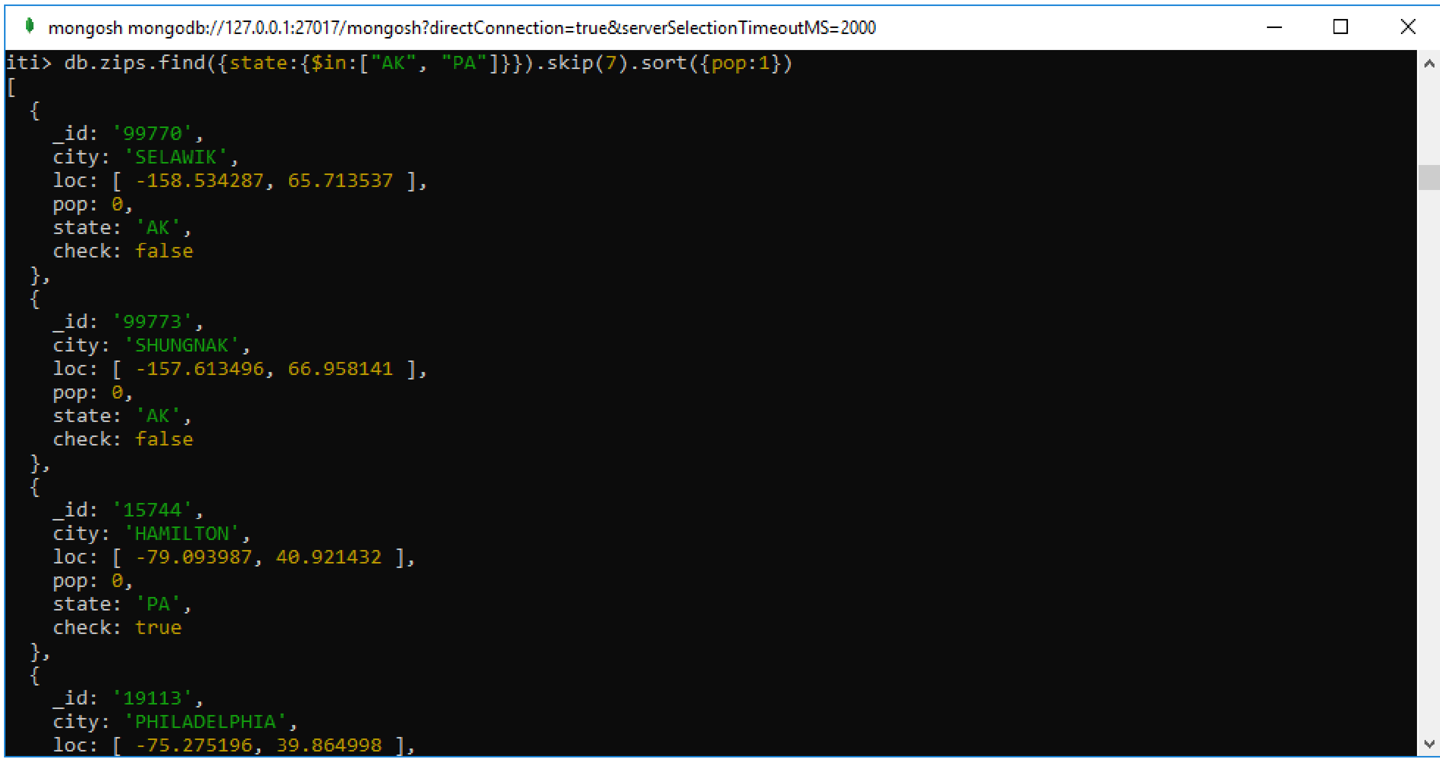
db.zips.aggregate([{$match: { state: "AK", "loc.1":{$lte:65}, "loc.1":{$gte:55}}}, {$group: {_id: "$state", sum: {$sum: "$pop"}}}])


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.aggregate([ {$match: { state: "AK", "loc.1":{$lte:65}, "loc.1":{$gte:55}}}, {$group: {_id: "$state", sum: {
$sum: "$pop"}}}])
[ { _id: 'AK', sum: 540788 } ]
iti> _
```

4. Sort Population of document that state in AK, PA and skip first 7 document

db.zips.find({state:{$in:["AK", "PA"]}}).skip(7).sort({pop:1})


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

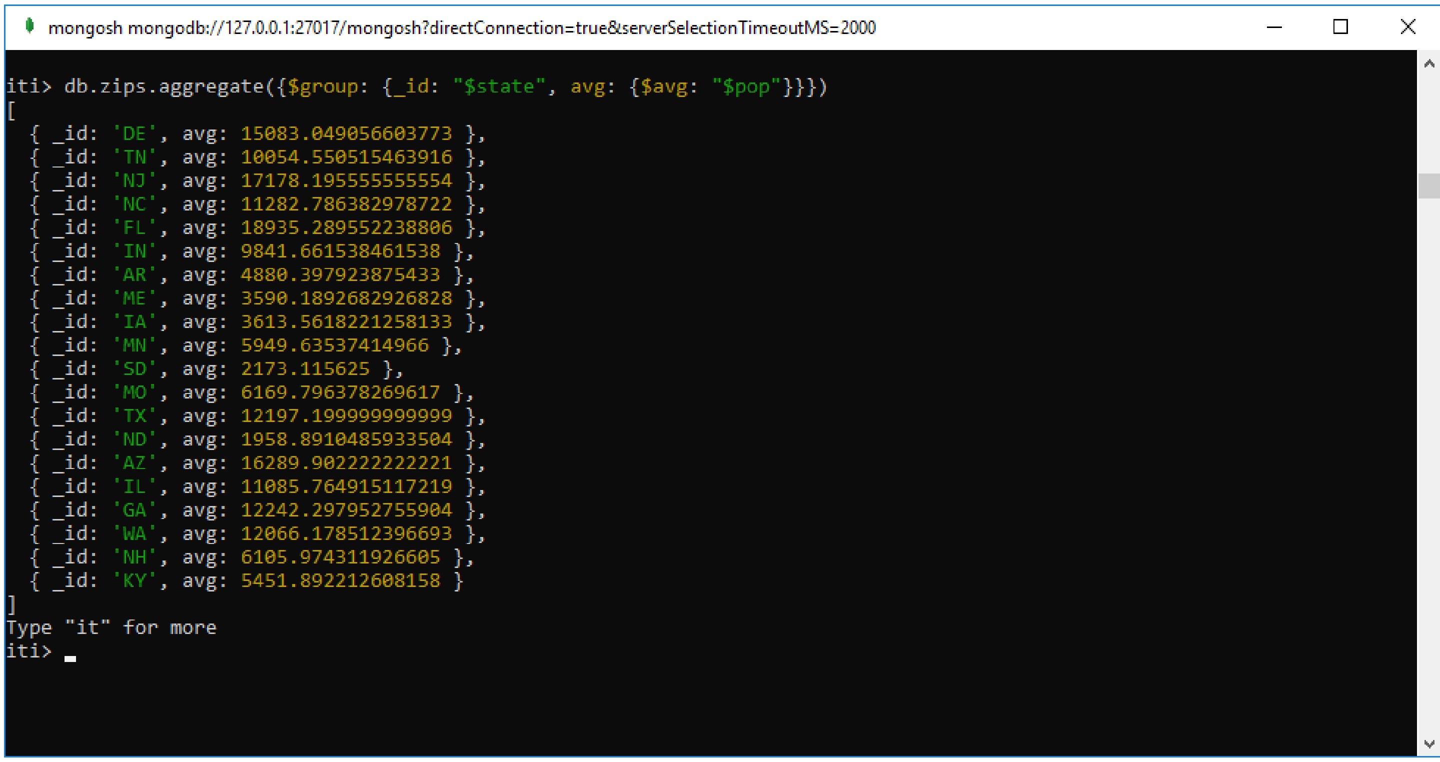
```
iti> db.zips.find({state:{$in:["AK", "PA"]}}).skip(7).sort({pop:1})
[
  {
    _id: '99770',
    city: 'SELAWIK',
    loc: [ -158.534287, 65.713537 ],
    pop: 0,
    state: 'AK',
    check: false
  },
  {
    _id: '99773',
    city: 'SHUNGNAK',
    loc: [ -157.613496, 66.958141 ],
    pop: 0,
    state: 'AK',
    check: false
  },
  {
    _id: '15744',
    city: 'HAMILTON',
    loc: [ -79.093987, 40.921432 ],
    pop: 0,
    state: 'PA',
    check: true
  },
  {
    _id: '19113',
    city: 'PHILADELPHIA',
    loc: [ -75.275196, 39.864998 ],
```

5. Get smallest population and greatest population of each state and save the result in collection named "mypop" on your machine colleague

db.zips.aggregate({$group: {_id: "$state", max: {$max: "$pop"}, min: {$min: "$pop"}}},{$out: {db: "iti", coll: "mypop"}})

6. Write an aggregation expression to calculate the average population of a zip code (postal code) by state

db.zips.aggregate({$group: {_id: "$state", avg: {$avg: "$pop"}}})

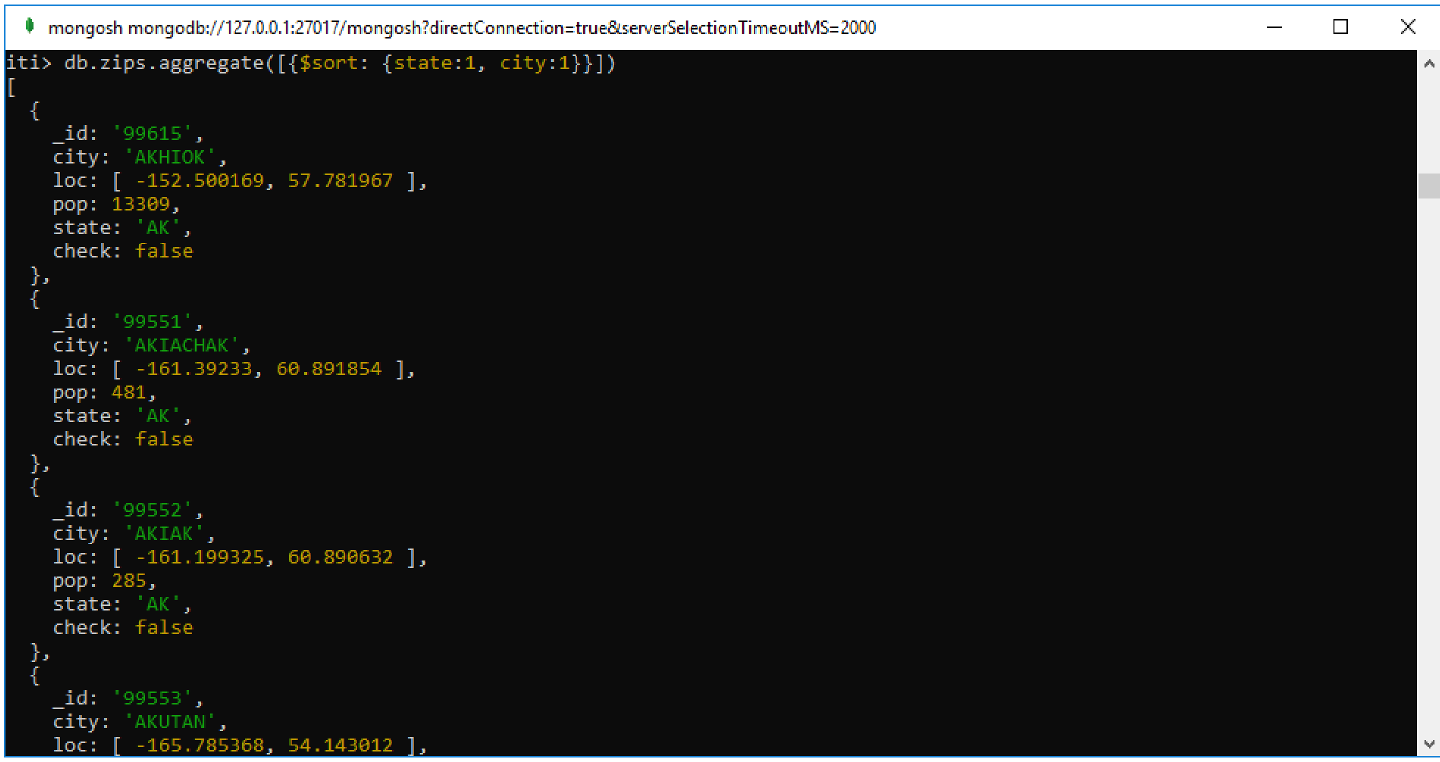


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.aggregate({$group: {_id: "$state", avg: {$avg: "$pop"}}})
[
  { _id: 'DE', avg: 15083.049056603773 },
  { _id: 'TN', avg: 10054.550515463916 },
  { _id: 'NJ', avg: 17178.195555555554 },
  { _id: 'NC', avg: 11282.786382978722 },
  { _id: 'FL', avg: 18935.289552238806 },
  { _id: 'IN', avg: 9841.661538461538 },
  { _id: 'AR', avg: 4880.397923875433 },
  { _id: 'ME', avg: 3590.1892682926828 },
  { _id: 'IA', avg: 3613.5618221258133 },
  { _id: 'MN', avg: 5949.63537414966 },
  { _id: 'SD', avg: 2173.115625 },
  { _id: 'MO', avg: 6169.796378269617 },
  { _id: 'TX', avg: 12197.199999999999 },
  { _id: 'ND', avg: 1958.8910485933504 },
  { _id: 'AZ', avg: 16289.902222222221 },
  { _id: 'IL', avg: 11085.764915117219 },
  { _id: 'GA', avg: 12242.297952755904 },
  { _id: 'WA', avg: 12066.178512396693 },
  { _id: 'NH', avg: 6105.974311926605 },
  { _id: 'KY', avg: 5451.892212608158 }
]
Type "it" for more
iti> 
```

7. Write an aggregation query with just a sort stage to sort by (state, city), both ascending

db.zips.aggregate([{$sort: {state:1, city:1}}])

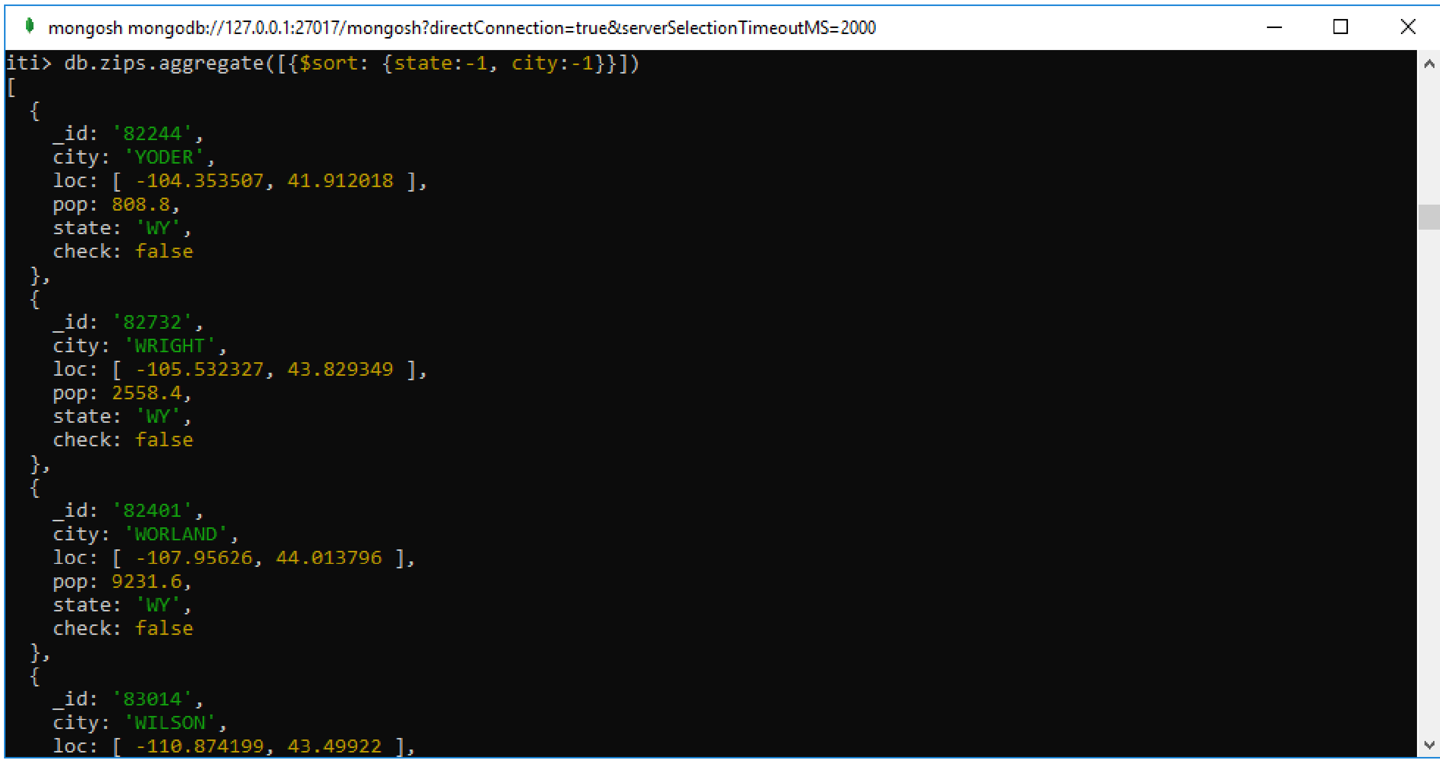


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.aggregate([{$sort: {state:1, city:1}}])
[
  {
    _id: '99615',
    city: 'AKHIOK',
    loc: [ -152.500169, 57.781967 ],
    pop: 13309,
    state: 'AK',
    check: false
  },
  {
    _id: '99551',
    city: 'AKIACHAK',
    loc: [ -161.39233, 60.891854 ],
    pop: 481,
    state: 'AK',
    check: false
  },
  {
    _id: '99552',
    city: 'AKIAK',
    loc: [ -161.199325, 60.890632 ],
    pop: 285,
    state: 'AK',
    check: false
  },
  {
    _id: '99553',
    city: 'AKUTAN',
    loc: [ -165.785368, 54.143012 ],
```

8. Write an aggregation query with just a sort stage to sort by (state, city), both descending

db.zips.aggregate([{$sort: {state:-1, city:-1}}])

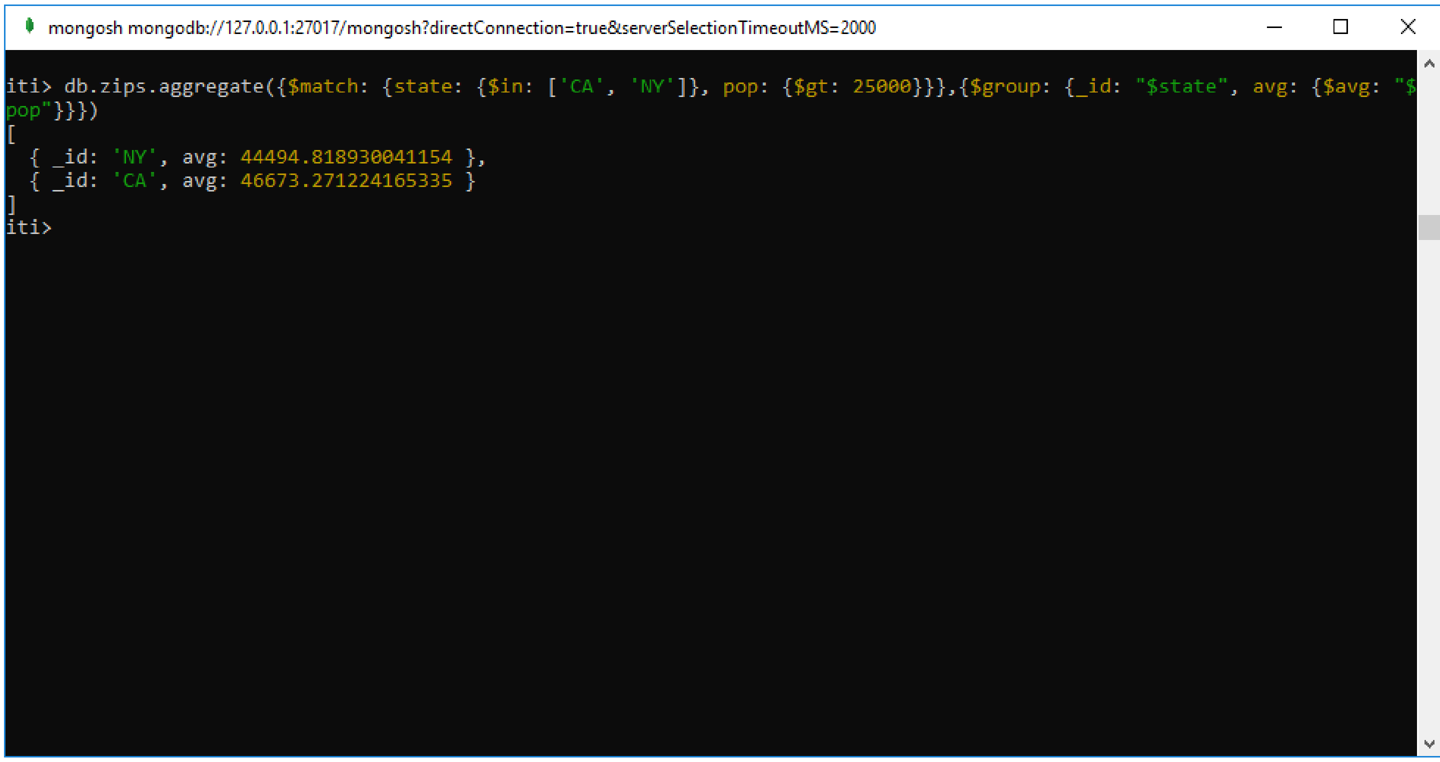


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.aggregate([{$sort: {state:-1, city:-1}}])
[
  {
    _id: '82244',
    city: 'YODER',
    loc: [ -104.353507, 41.912018 ],
    pop: 808.8,
    state: 'WY',
    check: false
  },
  {
    _id: '82732',
    city: 'WRIGHT',
    loc: [ -105.532327, 43.829349 ],
    pop: 2558.4,
    state: 'WY',
    check: false
  },
  {
    _id: '82401',
    city: 'WORLAND',
    loc: [ -107.95626, 44.013796 ],
    pop: 9231.6,
    state: 'WY',
    check: false
  },
  {
    _id: '83014',
    city: 'WILSON',
    loc: [ -110.874199, 43.49922 ],
```

9. Calculate the average population of cities in California (abbreviation CA) and New York (NY) (taken together) with populations over 25,000

db.zips.aggregate({$match: {state: {$in: ['CA', 'NY']}, pop: {$gt: 25000}}},{$group: {_id: "$state", avg: {$avg: "$pop"}}})

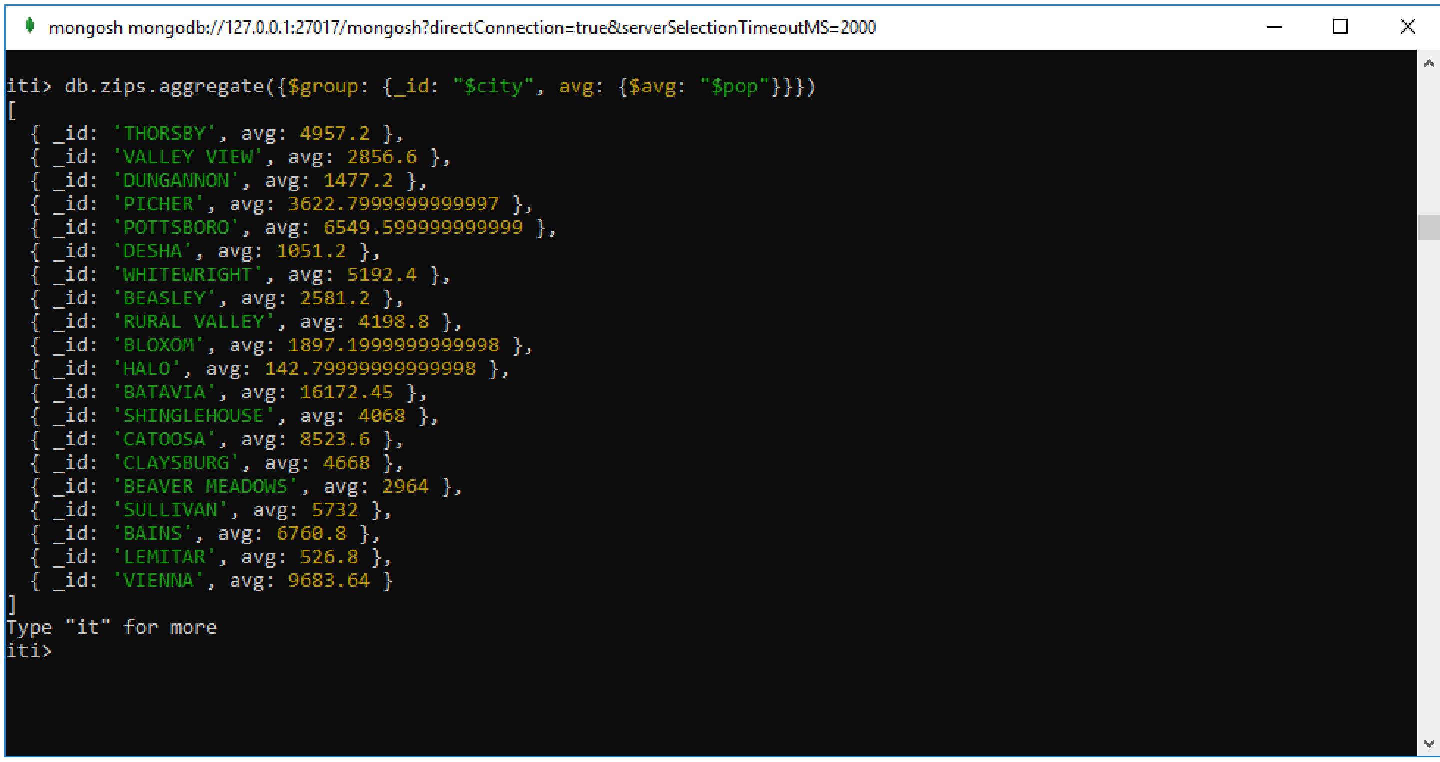


mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.aggregate({$match: {state: {$in: ['CA', 'NY']}, pop: {$gt: 25000}}},{$group: {_id: "$state", avg: {$avg: "$
pop"}}})
[
  { _id: 'NY', avg: 44494.818930041154 },
  { _id: 'CA', avg: 46673.271224165335 }
]
iti>
```

10. Return the average populations for cities in each state

db.zips.aggregate({$group: {_id: "$city", avg: {$avg: "$pop"}}})



mongosh mongodb://127.0.0.1:27017/mongosh?directConnection=true&serverSelectionTimeoutMS=2000

```
iti> db.zips.aggregate({$group: {_id: "$city", avg: {$avg: "$pop"}}})
[
  { _id: 'THORSBY', avg: 4957.2 },
  { _id: 'VALLEY VIEW', avg: 2856.6 },
  { _id: 'DUNGANNON', avg: 1477.2 },
  { _id: 'PICHER', avg: 3622.7999999999997 },
  { _id: 'POTTSBORO', avg: 6549.599999999999 },
  { _id: 'DESHA', avg: 1051.2 },
  { _id: 'WHITEWRIGHT', avg: 5192.4 },
  { _id: 'BEASLEY', avg: 2581.2 },
  { _id: 'RURAL VALLEY', avg: 4198.8 },
  { _id: 'BLOXOM', avg: 1897.1999999999998 },
  { _id: 'HALO', avg: 142.79999999999998 },
  { _id: 'BATAVIA', avg: 16172.45 },
  { _id: 'SHINGLEHOUSE', avg: 4068 },
  { _id: 'CATOOSA', avg: 8523.6 },
  { _id: 'CLAYSBURG', avg: 4668 },
  { _id: 'BEAVER MEADOWS', avg: 2964 },
  { _id: 'SULLIVAN', avg: 5732 },
  { _id: 'BAINS', avg: 6760.8 },
  { _id: 'LEMITAR', avg: 526.8 },
  { _id: 'VIENNA', avg: 9683.64 }
]
Type "it" for more
iti>
```