MongoDB Lab2

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

$ sudo apt install mongo-tools

$ sudo mongoimport --db iti --collection umongo --file /home/fierro98/Desktop/zips.json

```
fierro98@ubuntu:~$ sudo mongoimport --db iti --collection umongo --file /home/fi
erro98/Desktop/zips.json
2022-03-24T06:49:44.855-0700    connected to: localhost
2022-03-24T06:49:46.241-0700    imported 29353 documents
```

```
> show collections
friends
students
umongo
```

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

> db.umongo.find({state: "NY"})

(or)

> db. umongo.aggregate([{$match:{ "state": "NY"}])

```
> db.umongo.find({state: "NY"})
{ "_id" : "06390", "city" : "FISHERS ISLAND", "loc" : [ -72.017834, 41.263934 ], "pop" : 329, "state" : "NY" }
{ "_id" : "10001", "city" : "NEW YORK", "loc" : [ -73.996705, 40.74838 ], "pop" : 18913, "state" : "NY" }
{ "_id" : "10002", "city" : "NEW YORK", "loc" : [ -73.987681, 40.715231 ], "pop" : 84143, "state" : "NY" }
{ "_id" : "10004", "city" : "GOVERNORS ISLAND", "loc" : [ -74.019025, 40.693604 ], "pop" : 3593, "state" : "NY" }
{ "_id" : "10005", "city" : "NEW YORK", "loc" : [ -74.008344, 40.705649 ], "pop" : 202, "state" : "NY" }
{ "_id" : "10006", "city" : "NEW YORK", "loc" : [ -74.013474, 40.708451 ], "pop" : 119, "state" : "NY" }
{ "_id" : "10007", "city" : "NEW YORK", "loc" : [ -74.007022, 40.713905 ], "pop" : 3374, "state" : "NY" }
{ "_id" : "10009", "city" : "NEW YORK", "loc" : [ -73.979591, 40.726188 ], "pop" : 57426, "state" : "NY" }
{ "_id" : "10010", "city" : "NEW YORK", "loc" : [ -73.981328, 40.737476 ], "pop" : 24907, "state" : "NY" }
{ "_id" : "10011", "city" : "NEW YORK", "loc" : [ -73.99963, 40.740225 ], "pop" : 46560, "state" : "NY" }
{ "_id" : "10012", "city" : "NEW YORK", "loc" : [ -73.998284, 40.72553 ], "pop" : 26365, "state" : "NY" }
{ "_id" : "10013", "city" : "NEW YORK", "loc" : [ -74.002529, 40.718511 ], "pop" : 21860, "state" : "NY" }
{ "_id" : "10014", "city" : "NEW YORK", "loc" : [ -74.005421, 40.73393 ], "pop" : 31147, "state" : "NY" }
{ "_id" : "10016", "city" : "NEW YORK", "loc" : [ -73.978134, 40.744281 ], "pop" : 51561, "state" : "NY" }
{ "_id" : "10017", "city" : "NEW YORK", "loc" : [ -73.970661, 40.75172 ], "pop" : 12465, "state" : "NY" }
{ "_id" : "10018", "city" : "NEW YORK", "loc" : [ -73.992503, 40.754713 ], "pop" : 4834, "state" : "NY" }
{ "_id" : "10019", "city" : "NEW YORK", "loc" : [ -73.985834, 40.765069 ], "pop" : 36602, "state" : "NY" }
{ "_id" : "10020", "city" : "NEW YORK", "loc" : [ -73.982347, 40.759729 ], "pop" : 393, "state" : "NY" }
{ "_id" : "10021", "city" : "NEW YORK", "loc" : [ -73.958805, 40.768476 ], "pop" : 106564, "state" : "NY" }
{ "_id" : "10022", "city" : "NEW YORK", "loc" : [ -73.965703, 40.757091 ], "pop" : 31870, "state" : "NY" }
Type "it" for more
> db. umongo.aggregate({$match:{ "state": "NY"}})
{ "_id" : "06390", "city" : "FISHERS ISLAND", "loc" : [ -72.017834, 41.263934 ], "pop" : 329, "state" : "NY" }
{ "_id" : "10001", "city" : "NEW YORK", "loc" : [ -73.996705, 40.74838 ], "pop" : 18913, "state" : "NY" }
{ "_id" : "10002", "city" : "NEW YORK", "loc" : [ -73.987681, 40.715231 ], "pop" : 84143, "state" : "NY" }
{ "_id" : "10004", "city" : "GOVERNORS ISLAND", "loc" : [ -74.019025, 40.693604 ], "pop" : 3593, "state" : "NY" }
{ "_id" : "10005", "city" : "NEW YORK", "loc" : [ -74.008344, 40.705649 ], "pop" : 202, "state" : "NY" }
{ "_id" : "10006", "city" : "NEW YORK", "loc" : [ -74.013474, 40.708451 ], "pop" : 119, "state" : "NY" }
{ "_id" : "10007", "city" : "NEW YORK", "loc" : [ -74.007022, 40.713905 ], "pop" : 3374, "state" : "NY" }
{ "_id" : "10009", "city" : "NEW YORK", "loc" : [ -73.979591, 40.726188 ], "pop" : 57426, "state" : "NY" }
{ "_id" : "10010", "city" : "NEW YORK", "loc" : [ -73.981328, 40.737476 ], "pop" : 24907, "state" : "NY" }
{ "_id" : "10011", "city" : "NEW YORK", "loc" : [ -73.99963, 40.740225 ], "pop" : 46560, "state" : "NY" }
{ "_id" : "10012", "city" : "NEW YORK", "loc" : [ -73.998284, 40.72553 ], "pop" : 26365, "state" : "NY" }
{ "_id" : "10013", "city" : "NEW YORK", "loc" : [ -74.002529, 40.718511 ], "pop" : 21860, "state" : "NY" }
{ "_id" : "10014", "city" : "NEW YORK", "loc" : [ -74.005421, 40.73393 ], "pop" : 31147, "state" : "NY" }
{ "_id" : "10016", "city" : "NEW YORK", "loc" : [ -73.978134, 40.744281 ], "pop" : 51561, "state" : "NY" }
{ "_id" : "10017", "city" : "NEW YORK", "loc" : [ -73.970661, 40.75172 ], "pop" : 12465, "state" : "NY" }
{ "_id" : "10018", "city" : "NEW YORK", "loc" : [ -73.992503, 40.754713 ], "pop" : 4834, "state" : "NY" }
{ "_id" : "10019", "city" : "NEW YORK", "loc" : [ -73.985834, 40.765069 ], "pop" : 36602, "state" : "NY" }
{ "_id" : "10020", "city" : "NEW YORK", "loc" : [ -73.982347, 40.759729 ], "pop" : 393, "state" : "NY" }
{ "_id" : "10021", "city" : "NEW YORK", "loc" : [ -73.958805, 40.768476 ], "pop" : 106564, "state" : "NY" }
{ "_id" : "10022", "city" : "NEW YORK", "loc" : [ -73.965703, 40.757091 ], "pop" : 31870, "state" : "NY" }
Type "it" for more
```

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

> db.umongo.find({$or:[{pop: 1000}, {pop: {$gt:1000}}]}, {_id:1})

```
> db.umongo.find({$or:[{pop: 1000}, {pop: {$gt:1000}}]}, {_id:1})
{ "_id" : "01001" }
{ "_id" : "01005" }
{ "_id" : "01007" }
{ "_id" : "01008" }
{ "_id" : "01010" }
{ "_id" : "01011" }
{ "_id" : "01013" }
{ "_id" : "01020" }
{ "_id" : "01022" }
{ "_id" : "01026" }
{ "_id" : "01027" }
{ "_id" : "01028" }
{ "_id" : "01030" }
{ "_id" : "01031" }
{ "_id" : "01033" }
{ "_id" : "01034" }
{ "_id" : "01035" }
{ "_id" : "01002" }
{ "_id" : "01036" }
{ "_id" : "01038" }
Type "it" for more
> it
{ "_id" : "01039" }
{ "_id" : "01050" }
{ "_id" : "01040" }
{ "_id" : "01054" }
{ "_id" : "01056" }
{ "_id" : "01057" }
{ "_id" : "01060" }
{ "_id" : "01068" }
{ "_id" : "01069" }
{ "_id" : "01072" }
{ "_id" : "01073" }
{ "_id" : "01075" }
{ "_id" : "01077" }
{ "_id" : "01080" }
{ "_id" : "01081" }
{ "_id" : "01082" }
{ "_id" : "01085" }
{ "_id" : "01089" }
{ "_id" : "01092" }
{ "_id" : "01095" }
Type "it" for more
```

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

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

(or)

```
> db.umongo.aggregate( [
  {
    $match: {$or:[{state: "PA"}, {state: "VA"}]}
  },
  {
    $addFields: {"check": true}
  }
] )
```

```
> db.umongo.updateMany({$or:[{state: "PA"}, {state: "VA"}]}, {$set: {"check": true}})
{ "acknowledged" : true, "matchedCount" : 2274, "modifiedCount" : 2274 }
```

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

```
> db.umongo.find({"loc.1":{$gt:55}, "loc.1":{$lt:65}}, {_id:0, pop:1})
```

```
> db.umongo.find({"loc.1":{$gt:55}, "loc.1":{$lt:65}}, {_id:0, pop:1})
{ "pop" : 15338 }
{ "pop" : 4546 }
{ "pop" : 10579 }
{ "pop" : 1240 }
{ "pop" : 3706 }
{ "pop" : 1688 }
{ "pop" : 177 }
{ "pop" : 23396 }
{ "pop" : 31495 }
{ "pop" : 1764 }
{ "pop" : 1484 }
{ "pop" : 16864 }
{ "pop" : 13367 }
{ "pop" : 11985 }
{ "pop" : 2385 }
{ "pop" : 122 }
{ "pop" : 5526 }
{ "pop" : 1652 }
{ "pop" : 4231 }
{ "pop" : 36963 }
Type "it" for more
```

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

```
> db.umongo.createIndex({state:1})
```

```
> db.umongo.createIndex({state:1})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
```

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

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

```
> db.umongo.updateMany({state:{$nin:[ "AK", "NY"]}}, { $mul: { pop: 1.2 }})
{ "acknowledged" : true, "matchedCount" : 27563, "modifiedCount" : 27563 }
```

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

part2

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

> db.umongo.aggregate({$match:{state: {$in: ['PA', 'KA']}}},{$group:{_id:"$state",sum:{$sum:"$pop"}}})

```
> db.umongo.aggregate({$match:{state:{ $in: ['PA', 'KA']}}},{$group:{_id:"$state",sum:{$sum:"$pop"}}})
{ "_id" : "PA", "sum" : 14254629.6 }
```

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

> db.umongo.aggregate([{ $match: { state: {$nin:["PA", "KA"]} }} , { $limit: 5}])

(or)

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

```
> db.umongo.aggregate([ { $match: { state: {$nin:["PA", "KA"]} }} , { $limit: 5}])
{ "_id" : "99501", "city" : "ANCHORAGE", "loc" : [ -149.876077, 61.211571 ], "pop" : 14436, "state" : "AK" }
{ "_id" : "99502", "city" : "ANCHORAGE", "loc" : [ -150.093943, 61.096163 ], "pop" : 15891, "state" : "AK" }
{ "_id" : "99503", "city" : "ANCHORAGE", "loc" : [ -149.893844, 61.189953 ], "pop" : 12534, "state" : "AK" }
{ "_id" : "99504", "city" : "ANCHORAGE", "loc" : [ -149.74467, 61.203696 ], "pop" : 32383, "state" : "AK" }
{ "_id" : "99505", "city" : "FORT RICHARDSON", "loc" : [ -149.675454, 61.275256 ], "pop" : 7979, "state" : "AK" }
```

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

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

```
> db.umongo.aggregate([ {$match: { state: "AK", "loc.1":{$lte:65}, "loc.1":{$gte:55}}}, {$group: {_id: "$state", sum: {$sum: "$pop"}}}])
{ "_id" : "AK", "sum" : 535790 }
```

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

> db.umongo.aggregate([{$match: {$or:[{state: "PA"}, { state: "AK"}]}}, {$sort:{pop: 1}}, {$skip: 7}])

(or)

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

```
> db.umongo.aggregate([{$match: {$or:[{state: "PA"}, { state: "AK"}]}}, {$sort:{pop: 1}}, {$skip: 7}])
{ "_id" : "99574", "city" : "CHENEGA BAY", "loc" : [ -147.943316, 60.102558 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99575", "city" : "CROOKED CREEK", "loc" : [ -158.002483, 61.818072 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99579", "city" : "EGEGIK", "loc" : [ -157.342202, 58.206174 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99580", "city" : "EKWOK", "loc" : [ -157.478211, 59.362792 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99581", "city" : "EMMONAK", "loc" : [ -164.131298, 62.827404 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99590", "city" : "GRAYLING", "loc" : [ -159.404907, 63.372013 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99591", "city" : "SAINT GEORGE ISL", "loc" : [ -169.547257, 56.60324 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99607", "city" : "KALSKAG", "loc" : [ -160.3261, 61.541006 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99583", "city" : "FALSE PASS", "loc" : [ -163.436845, 54.841028 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99630", "city" : "MEKORYUK", "loc" : [ -166.283583, 60.365679 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99633", "city" : "NAKNEK", "loc" : [ -156.705405, 58.885699 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99638", "city" : "NIKOLSKI", "loc" : [ -168.788427, 52.988337 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99647", "city" : "PEDRO BAY", "loc" : [ -153.821856, 59.92238 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99648", "city" : "PERRYVILLE", "loc" : [ -159.259333, 55.945289 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99649", "city" : "PILOT POINT", "loc" : [ -157.449272, 57.595193 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99651", "city" : "PLATINUM", "loc" : [ -162.043201, 58.63364 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99653", "city" : "PORT ALSWORTH", "loc" : [ -154.433803, 60.636416 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99656", "city" : "RED DEVIL", "loc" : [ -157.195969, 61.735389 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99665", "city" : "SHAGELUK", "loc" : [ -159.52816, 62.661092 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99668", "city" : "SLEETMUTE", "loc" : [ -157.118284, 61.634555 ], "pop" : 0, "state" : "AK" }
```

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

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

```
> db.umongo.aggregate({$group: {_id: "$state", avg: {$avg: "$pop"}}})
{ "_id" : "AK", "avg" : 2766.502564102564 }
{ "_id" : "WA", "avg" : 12062.352892561983 }
{ "_id" : "OR", "avg" : 8874.234375 }
{ "_id" : "ID", "avg" : 4940.498360655737 }
{ "_id" : "WY", "avg" : 3867.4714285714285 }
{ "_id" : "NE", "avg" : 3295.5679442508713 }
{ "_id" : "KS", "avg" : 4145.82041958042 }
{ "_id" : "ND", "avg" : 1937.198976982097 }
{ "_id" : "NJ", "avg" : 17176.14222222222 }
{ "_id" : "HI", "avg" : 16620.12 }
{ "_id" : "VT", "avg" : 2773.0518518518516 }
{ "_id" : "PA", "avg" : 9776.837860082303 }
{ "_id" : "LA", "avg" : 10906.055172413793 }
{ "_id" : "MT", "avg" : 3035.5987261146497 }
{ "_id" : "MS", "avg" : 8505.242975206611 }
{ "_id" : "NY", "avg" : 11273.891536050156 }
{ "_id" : "NV", "avg" : 13861.915384615384 }
{ "_id" : "SD", "avg" : 2158.559375 }
{ "_id" : "CA", "avg" : 23549.550395778362 }
{ "_id" : "MO", "avg" : 6166.609255533199 }
Type "it" for more
```

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

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

```
> db.umongo.aggregate([{$sort: {state:1, city:1}}])
{ "_id" : "99615", "city" : "AKHIOK", "loc" : [ -152.500169, 57.781967 ], "pop" : 13309, "state" : "AK" }
{ "_id" : "99551", "city" : "AKIACHAK", "loc" : [ -161.39233, 60.891854 ], "pop" : 481, "state" : "AK" }
{ "_id" : "99552", "city" : "AKIAK", "loc" : [ -161.199325, 60.890632 ], "pop" : 285, "state" : "AK" }
{ "_id" : "99553", "city" : "AKUTAN", "loc" : [ -165.785368, 54.143012 ], "pop" : 589, "state" : "AK" }
{ "_id" : "99554", "city" : "ALAKANUK", "loc" : [ -164.60228, 62.746967 ], "pop" : 1186, "state" : "AK" }
{ "_id" : "99555", "city" : "ALEKNAGIK", "loc" : [ -158.619882, 59.269688 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99720", "city" : "ALLAKAKET", "loc" : [ -152.712155, 66.543197 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99786", "city" : "AMBLER", "loc" : [ -156.455652, 67.46951 ], "pop" : 0, "state" : "AK" }
{ "_id" : "99721", "city" : "ANAKTUVUK PASS", "loc" : [ -151.679005, 68.11878 ], "pop" : 260, "state" : "AK" }
{ "_id" : "99501", "city" : "ANCHORAGE", "loc" : [ -149.876077, 61.211571 ], "pop" : 14436, "state" : "AK" }
{ "_id" : "99502", "city" : "ANCHORAGE", "loc" : [ -150.093943, 61.096163 ], "pop" : 15891, "state" : "AK" }
{ "_id" : "99503", "city" : "ANCHORAGE", "loc" : [ -149.893844, 61.189953 ], "pop" : 12534, "state" : "AK" }
{ "_id" : "99504", "city" : "ANCHORAGE", "loc" : [ -149.74467, 61.203696 ], "pop" : 32383, "state" : "AK" }
{ "_id" : "99507", "city" : "ANCHORAGE", "loc" : [ -149.828912, 61.153543 ], "pop" : 20128, "state" : "AK" }
{ "_id" : "99508", "city" : "ANCHORAGE", "loc" : [ -149.810085, 61.205959 ], "pop" : 29857, "state" : "AK" }
{ "_id" : "99515", "city" : "ANCHORAGE", "loc" : [ -149.897401, 61.119381 ], "pop" : 17094, "state" : "AK" }
{ "_id" : "99516", "city" : "ANCHORAGE", "loc" : [ -149.779998, 61.10541 ], "pop" : 18356, "state" : "AK" }
{ "_id" : "99517", "city" : "ANCHORAGE", "loc" : [ -149.936111, 61.190136 ], "pop" : 15192, "state" : "AK" }
{ "_id" : "99518", "city" : "ANCHORAGE", "loc" : [ -149.886571, 61.154862 ], "pop" : 8116, "state" : "AK" }
{ "_id" : "99744", "city" : "ANDERSON", "loc" : [ -149.1718, 64.300693 ], "pop" : 300, "state" : "AK" }
Type "it" for more
```

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

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

```
> db.umongo.aggregate([{$sort: {state:-1, city:-1}}])
{ "_id" : "82244", "city" : "YODER", "loc" : [ -104.353507, 41.912018 ], "pop" : 808.8, "state" : "WY" }
{ "_id" : "82732", "city" : "WRIGHT", "loc" : [ -105.532327, 43.829349 ], "pop" : 2558.4, "state" : "WY" }
{ "_id" : "82401", "city" : "WORLAND", "loc" : [ -107.95626, 44.013796 ], "pop" : 9231.6, "state" : "WY" }
{ "_id" : "83014", "city" : "WILSON", "loc" : [ -110.874199, 43.49922 ], "pop" : 1318.8, "state" : "WY" }
{ "_id" : "82201", "city" : "WHEATLAND", "loc" : [ -104.967852, 42.049467 ], "pop" : 7142.4, "state" : "WY" }
{ "_id" : "82450", "city" : "WAPITI", "loc" : [ -109.432629, 44.47967 ], "pop" : 256.8, "state" : "WY" }
{ "_id" : "82336", "city" : "WAMSUTTER", "loc" : [ -108.151674, 41.658278 ], "pop" : 619.1999999999999, "state" : "WY" }
{ "_id" : "82243", "city" : "VETERAN", "loc" : [ -104.370899, 41.982091 ], "pop" : 0, "state" : "WY" }
{ "_id" : "82242", "city" : "VAN TASSELL", "loc" : [ -104.315073, 42.830004 ], "pop" : 462, "state" : "WY" }
{ "_id" : "82730", "city" : "UPTON", "loc" : [ -104.635159, 44.089271 ], "pop" : 1626, "state" : "WY" }
{ "_id" : "82240", "city" : "TORRINGTON", "loc" : [ -104.191662, 42.062377 ], "pop" : 11490, "state" : "WY" }
{ "_id" : "82084", "city" : "TIE SIDING", "loc" : [ -105.446222, 41.052785 ], "pop" : 0, "state" : "WY" }
{ "_id" : "83127", "city" : "THAYNE", "loc" : [ -111.011354, 42.933026 ], "pop" : 606, "state" : "WY" }
{ "_id" : "82442", "city" : "TEN SLEEP", "loc" : [ -107.415305, 43.997848 ], "pop" : 834, "state" : "WY" }
{ "_id" : "82729", "city" : "SUNDANCE", "loc" : [ -104.383696, 44.405755 ], "pop" : 2192.4, "state" : "WY" }
{ "_id" : "82842", "city" : "STORY", "loc" : [ -107.049229, 44.607169 ], "pop" : 0, "state" : "WY" }
{ "_id" : "83126", "city" : "SMOOT", "loc" : [ -110.922351, 42.619238 ], "pop" : 496.79999999999995, "state" : "WY" }
{ "_id" : "82334", "city" : "SINCLAIR", "loc" : [ -107.109083, 41.779741 ], "pop" : 636, "state" : "WY" }
{ "_id" : "82649", "city" : "SHOSHONI", "loc" : [ -108.100667, 43.245707 ], "pop" : 720, "state" : "WY" }
{ "_id" : "82801", "city" : "SHERIDAN", "loc" : [ -106.964795, 44.78486 ], "pop" : 24030, "state" : "WY" }
Type "it" for more
```

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

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

```
> db.umongo.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 }
```

10. Return the average populations for cities in each state

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

```
> db.umongo.aggregate({$group: {_id: "$city", avg: {$avg: "$pop"}}})
{ "_id" : "WRANGELL", "avg" : 2573 }
{ "_id" : "POINT BAKER", "avg" : 426 }
{ "_id" : "BREVIG MISSION", "avg" : 0 }
{ "_id" : "KLAWOCK", "avg" : 851 }
{ "_id" : "HYDER", "avg" : 0 }
{ "_id" : "HYDABURG", "avg" : 891 }
{ "_id" : "THORNE BAY", "avg" : 744 }
{ "_id" : "SKAGWAY", "avg" : 692 }
{ "_id" : "CHALKYITSIK", "avg" : 0 }
{ "_id" : "WAINWRIGHT", "avg" : 492 }
{ "_id" : "BORDER", "avg" : 1805 }
{ "_id" : "SHUNGNAK", "avg" : 0 }
{ "_id" : "SHISHMAREF", "avg" : 456 }
{ "_id" : "POINT HOPE", "avg" : 640 }
{ "_id" : "NULATO", "avg" : 492 }
{ "_id" : "NOORVIK", "avg" : 534 }
{ "_id" : "NENANA", "avg" : 393 }
{ "_id" : "LAKE MINCHUMINA", "avg" : 0 }
{ "_id" : "MANLEY HOT SPRIN", "avg" : 0 }
{ "_id" : "KOTZEBUE", "avg" : 3347 }
Type "it" for more
```