MongoDB Lab2

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

```
C:\Program Files\MongoDB\Server\6.0\bin>mongoimport --db iti --collection zips --file "C:\Users\Kimo Store\Desktop\MongoDB\Day2\zips.json"
2023-02-26T02:27:12.768+0200    connected to: mongodb://localhost/
2023-02-26T02:27:15.055+0200    29353 document(s) imported successfully. 0 document(s) failed to import.

C:\Program Files\MongoDB\Server\6.0\bin>
```

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

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

```
iti> db.zips.find({state:"NY"})
[
  {
    _id: '06390',
    city: 'FISHERS ISLAND',
    loc: [ -72.017834, 41.263934 ],
    pop: 329,
    state: 'NY'
  },
  {
    _id: '10002',
    city: 'NEW YORK',
    loc: [ -73.987681, 40.715231 ],
    pop: 84143,
    state: 'NY'
  },
  {
    _id: '10001',
    city: 'NEW YORK',
    loc: [ -73.996705, 40.74838 ],
    pop: 18913,
    state: 'NY'
  },
  {
    _id: '10003',
    city: 'NEW YORK',
    loc: [ -73.989223, 40.731253 ],
    pop: 51224,
    state: 'NY'
  },
```

3 – find all zip codes whose population is greater than or equal to 1000

```
db.zips.find({pop:{$gte:1000}})
```

```
iti> db.zips.find({pop:{$gte:1000}})
[
  {
    _id: '01002',
    city: 'CUSHMAN',
    loc: [ -72.51565, 42.377017 ],
    pop: 36963,
    state: 'MA'
  },
  {
    _id: '01010',
    city: 'BRIMFIELD',
    loc: [ -72.188455, 42.116543 ],
    pop: 3706,
    state: 'MA'
  },
  {
    _id: '01008',
    city: 'BLANDFORD',
    loc: [ -72.936114, 42.182949 ],
    pop: 1240,
    state: 'MA'
  },
```

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

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

```
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:{$gt:55,$lt:65}},{_id:0,pop:1})
```

```
iti> db.zips.find({loc:{$gt:55,$lt:65}},{_id:0,pop:1})
[
  { pop: 14436 }, { pop: 15891 },
  { pop: 12534 }, { pop: 32383 },
  { pop: 7979 },  { pop: 7907 },
  { pop: 20128 }, { pop: 29857 },
  { pop: 17094 }, { pop: 18356 },
  { pop: 15192 }, { pop: 8116 },
  { pop: 119 },   { pop: 481 },
  { pop: 1186 },  { pop: 296 },
  { pop: 7188 },  { pop: 320 },
  { pop: 352 },   { pop: 0 }
]
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})
```

```
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({$or:[{state:{$ne:"AK"}},
{state:{$ne:"NY"}}]}, {$inc:{pop: 0.2}})
```

```
iti> db.zips.updateMany({$or:[{state:{$ne:"AK"}}, {state:{$ne:"NY"}}]}, {$inc:{pop: 0.2}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 29353,
  modifiedCount: 29353,
  upsertedCount: 0
}
iti>
```

After Update:

```
iti> db.zips.find({state:"AK"})
[
  {
    _id: '99501',
    city: 'ANCHORAGE',
    loc: [ -149.876077, 61.211571 ],
    pop: 14436.2,
    state: 'AK'
  },
  {
    _id: '99502',
    city: 'ANCHORAGE',
    loc: [ -150.093943, 61.096163 ],
    pop: 15891.2,
    state: 'AK'
  },
```

```
iti> db.zips.find({state:"NY"})
[
  {
    _id: '06390',
    city: 'FISHERS ISLAND',
    loc: [ -72.017834, 41.263934 ],
    pop: 329.2,
    state: 'NY'
  },
  {
    _id: '10002',
    city: 'NEW YORK',
    loc: [ -73.987681, 40.715231 ],
    pop: 84143.2,
    state: 'NY'
  },
```

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.updateMany({loc:{$lt:-71}, state:{$ne:'MA'}, pop:{$lt:200}},{$set:{pop:0}})

```
iti> db.zips.updateMany({loc:{$lt:-71}, state:{$ne:'MA'}, pop:{$lt:200}},{$set:{pop:0}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1516,
  modifiedCount: 1516,
  upsertedCount: 0
}
iti>
```

After Update:

```
iti> db.zips.find({pop:0})
[
  {
    _id: '02815',
    city: 'CLAYVILLE',
    loc: [ -71.670589, 41.777762 ],
    pop: 0,
    state: 'RI',
    check: 'false'
  },
  {
    _id: '02836',
    city: 'RICHMOND',
    loc: [ -71.683992, 41.477694 ],
    pop: 0,
    state: 'RI',
    check: 'false'
  },
```

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 documet in the database but change the _id to avoid duplications.

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

```
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

```
iti> db.zips.aggregate([{$match:{state:'AK'}}, {$group: {_id:"$state", TotalSummationpopulation:{$sum:"$pop"}}}])
[ { _id: 'AK', TotalSummationpopulation: 539495.8 } ]
iti> db.zips.aggregate([{$match:{state:'PA'}}, {$group: {_id:"$state", TotalSummationpopulation:{$sum:"$pop"}}}])
[ { _id: 'PA', TotalSummationpopulation: 11877118 } ]
iti>
```

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

db.zips.find({$or:[{state:{$ne:"AK"}}, {state:{$ne:"NY"}}]}).limit(5)

```
iti> db.zips.find({$or:[{state:{$ne:"AK"}}, {state:{$ne:"NY"}}]}).limit(5)
[
  {
    _id: '99501',
    loc: [ -149.876077, 61.211571 ],
    pop: 14436.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
  {
    _id: '99502',
    loc: [ -150.093943, 61.096163 ],
    pop: 15891.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
  {
    _id: '99503',
    loc: [ -149.893844, 61.189953 ],
    pop: 12534.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
  {
    _id: '99504',
    loc: [ -149.74467, 61.203696 ],
    pop: 32383.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
  {
    _id: '99505',
    loc: [ -149.675454, 61.275256 ],
    pop: 7979.2,
    state: 'AK',
    check: 'false',
```

3. Get sum of population that state equal to AK and their latitude between 55, 65
db.zips.aggregate([{$match:{state:{$eq:'AK'}, loc:{$gt:55,$lt:65}}}, {$group: {_id:"$state", TotalSummationpopulation:{$sum:"$pop"}}}])

```
iti> db.zips.aggregate([{$match:{state:{$eq:'AK'}, loc:{$gt:55,$lt:65}}}, {$group: {_id:"$state", TotalSummationpopulation:{$sum:"$pop"}}}])
[ { _id: 'AK', TotalSummationpopulation: 535817.4 } ]
iti>
```

4. Sort Population of document that state in AK, PA and skip first 7 document
db.zips.find({state:'PA', state:'AK'}).skip(7)

```
iti> db.zips.find({state:'PA', state:'AK'}).skip(7)
[
  {
    _id: '99508',
    loc: [ -149.810085, 61.205959 ],
    pop: 29857.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
  {
    _id: '99515',
    loc: [ -149.897401, 61.119381 ],
    pop: 17094.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
  {
    _id: '99516',
    loc: [ -149.779998, 61.10541 ],
    pop: 18356.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
```

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", minpop:{$min:"$pop"}, maxpop:{$max:"$pop"}}}, {$out:"mypop"}])

```
iti> db.mypop.find()
[
  { _id: 'NC', minpop: 0, maxpop: 69179.2 },
  { _id: 'OH', minpop: 0, maxpop: 66674.2 },
  { _id: 'IA', minpop: 0, maxpop: 52105.2 },
  { _id: 'UT', minpop: 0, maxpop: 55999.2 },
  { _id: 'PA', minpop: 0, maxpop: 80454.2 },
  { _id: 'DE', minpop: 0, maxpop: 50573.2 },
  { _id: 'IL', minpop: 0, maxpop: 112047.2 },
  { _id: 'WY', minpop: 0, maxpop: 33107.2 },
  { _id: 'ID', minpop: 0, maxpop: 40912.2 },
  { _id: 'WV', minpop: 0, maxpop: 70185.2 },
  { _id: 'WI', minpop: 0, maxpop: 57187.2 },
  { _id: 'MT', minpop: 0, maxpop: 40121.2 },
  { _id: 'MO', minpop: 0, maxpop: 54994.2 },
  { _id: 'CA', minpop: 0, maxpop: 99568.2 },
  { _id: 'HI', minpop: 0, maxpop: 62915.2 },
  { _id: 'AR', minpop: 0, maxpop: 53532.2 },
  { _id: 'OK', minpop: 0, maxpop: 45542.2 },
  { _id: 'WA', minpop: 0, maxpop: 50515.2 },
  { _id: 'MA', minpop: 0.2, maxpop: 65046.2 },
  { _id: 'NH', minpop: 0, maxpop: 41438.2 }
]
Type "it" for more
iti>
```

6. Write an aggregation expression to calculate the average population of a zip code (postal code) by state
db.zips.aggregate([{$group:{_id:"$state", Averagepop:{$avg:"$pop"} }}])

```
iti> db.zips.aggregate([{$group:{_id:"$state", Averagepop:{$avg:"$pop"} }}])
[
  { _id: 'NC', Averagepop: 9400.492482269503 },
  { _id: 'OH', Averagepop: 10770.995829195632 },
  { _id: 'IA', Averagepop: 3007.7310195227765 },
  { _id: 'UT', Averagepop: 8394.480975609757 },
  { _id: 'PA', Averagepop: 8146.17146776406 },
  { _id: 'DE', Averagepop: 12564.475471698112 },
  { _id: 'IL', Averagepop: 9236.950687146322 },
  { _id: 'WY', Averagepop: 3215.0642857142857 },
  { _id: 'ID', Averagepop: 4111.86393442623 },
  { _id: 'WV', Averagepop: 2719.4076219512194 },
  { _id: 'WI', Averagepop: 6830.388268156425 },
  { _id: 'MT', Averagepop: 2525.1738853503184 },
  { _id: 'MO', Averagepop: 5135.466800804829 },
  { _id: 'CA', Averagepop: 19622.725725593667 },
  { _id: 'HI', Averagepop: 13847.814999999999 },
  { _id: 'AR', Averagepop: 4064.0325259515566 },
  { _id: 'OK', Averagepop: 5362.733447098975 },
  { _id: 'WA', Averagepop: 10051.412809917356 },
  { _id: 'MA', Averagepop: 12693.079746835443 },
  { _id: 'NH', Averagepop: 5082.355045871559 }
]
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 } }])

```
iti> db.zips.aggregate( [ { $sort : { state : 1, city:1 } } ] )
[
  {
    _id: '99501',
    loc: [ -149.876077, 61.211571 ],
    pop: 14436.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
  {
    _id: '99502',
    loc: [ -150.093943, 61.096163 ],
    pop: 15891.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
  {
    _id: '99503',
    loc: [ -149.893844, 61.189953 ],
    pop: 12534.2,
    state: 'AK',
    check: 'false',
    country: 'ANCHORAGE'
  },
  {
    _id: '99504',
```

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 } }])

```
iti> db.zips.aggregate( [ { $sort : { state : -1, city:-1 }
[   loc: [ -149.74467, 61.203696 ],
  { pop: 32383.2,
    _id: '82001',
    loc: [ -104.796234, 41.143719 ],
    pop: 33107.2,HORAGE'
    state: 'WY',
    check: 'false',
    country: 'CHEYENNE'
  },loc: [ -149.675454, 61.275256 ],
  { pop: 7979.2,
    _id: '82007',
    loc: [ -104.810745, 41.108433 ],
    pop: 15050.2,T RICHARDSON'
    state: 'WY',
    check: 'false',
    country: 'CHEYENNE'
  },loc: [ -149.812667, 61.251531 ],
  { pop: 7907.2,
    _id: '82009',
    loc: [ -104.802328, 41.183566 ],
    pop: 22028.2,ENDORF AFB'
    state: 'WY',
    check: 'false',
    country: 'CHEYENNE'
  },loc: [ -149.828912, 61.153543 ],
  { pop: 20128.2,
    _id: '82051',
    loc: [ -105.819708, 41.562721 ],
    pop: 0,: 'ANCHORAGE'
    state: 'WY',
    check: 'false',
    country: 'LARAMIE'
  },loc: [ -149.810085, 61.205959 ],
  { pop: 29857.2,
    _id: '82050',
    loc: [ -104.150542, 41.434237 ],
    pop: 310.2,NCHORAGE'
```

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:"$state", Averagepop:{$avg:"$pop"}}}])

```
iti> db.zips.aggregate([{$match:{state:{$in:["CA","NY"]}, pop:{$gt:25000}}},{$group:{_id:"$state", Averagepop:{$avg:"$pop"}}}])
[
  { _id: 'NY', Averagepop: 44495.01893004115 },
  { _id: 'CA', Averagepop: 41498.58888888889 }
]
iti>
```

10. Return the average populations for cities in each state

```
db.zips.aggregate([{$group:{_id:{State:"$state", City:"$city"}, Averagepop:{$avg:"$pop"}}}])
```

```
iti> db.zips.aggregate([{$group:{_id:{State:"$state", City:"$city"}, Averagepop:{$avg:"$pop"}}}])
[
  { _id: { State: 'IL' }, Averagepop: 9236.950687146322 },
  { _id: { State: 'NH' }, Averagepop: 5082.355045871559 },
  { _id: { State: 'WY' }, Averagepop: 3215.0642857142857 },
  { _id: { State: 'ND' }, Averagepop: 1604.3805626598466 },
  { _id: { State: 'AK' }, Averagepop: 2766.6451282051285 },
  { _id: { State: 'DE' }, Averagepop: 12564.475471698112 },
  { _id: { State: 'VT' }, Averagepop: 2308.759670781893 },
  { _id: { State: 'FL' }, Averagepop: 15778.842039800995 },
  { _id: { State: 'KY' }, Averagepop: 4535.386402966626 },
  { _id: { State: 'TN' }, Averagepop: 8376.45498281787 },
  { _id: { State: 'GA' }, Averagepop: 10200.476850393701 },
  { _id: { State: 'AL' }, Averagepop: 7124.849382716049 },
  { _id: { State: 'DC' }, Averagepop: 25286.350000000002 },
  { _id: { State: 'MD' }, Averagepop: 11379.451428571429 },
  { _id: { State: 'MO' }, Averagepop: 5135.466800804829 },
  { _id: { State: 'LA' }, Averagepop: 9087.799137931033 },
  { _id: { State: 'OK' }, Averagepop: 5362.733447098975 },
  { _id: { State: 'ID' }, Averagepop: 4111.86393442623 },
  { _id: { State: 'ME' }, Averagepop: 2992.0243902439024 },
  { _id: { State: 'MI' }, Averagepop: 10609.918493150684 }
]
Type "it" for more
iti>
```