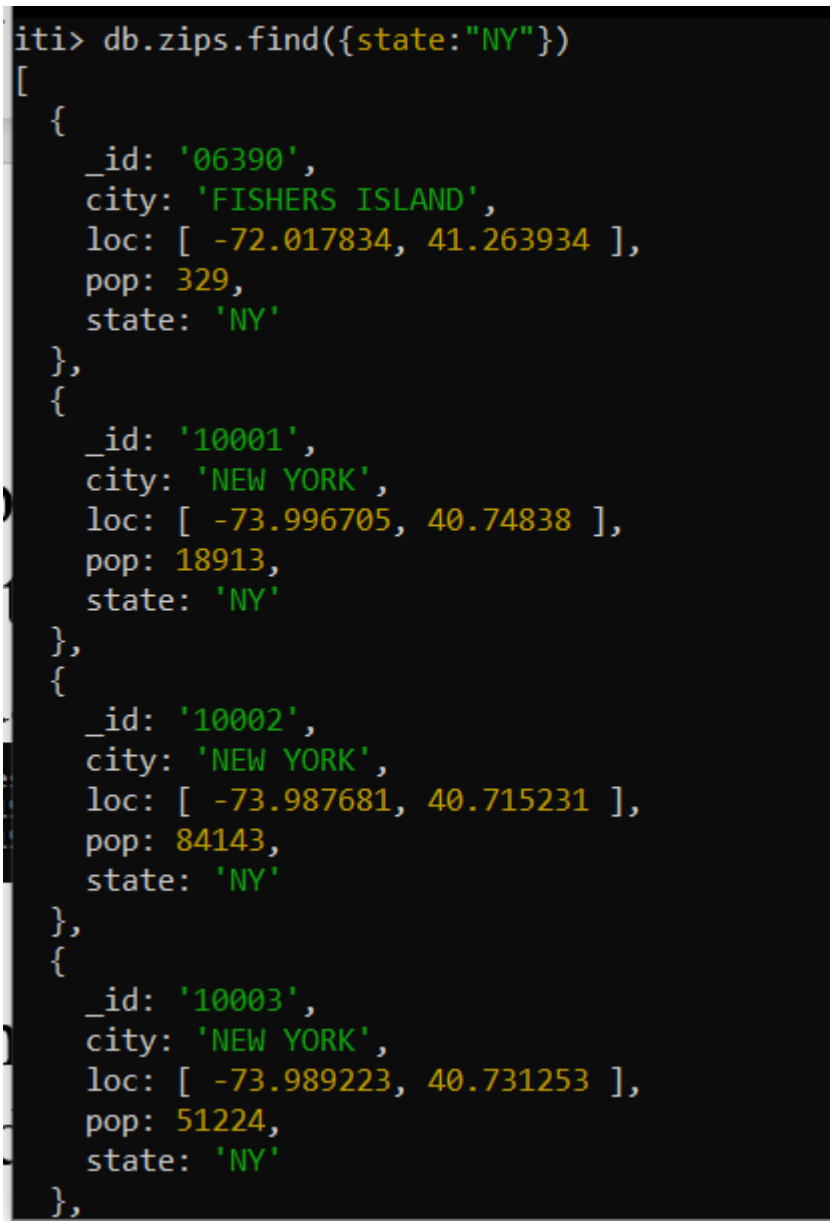
MongoDB Lab2

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 D:\ITI\MongoDB\day2\zips.json
2023-02-27T03:15:40.868+0200    connected to: mongodb://localhost/
2023-02-27T03:15:41.135+0200    29353 document(s) imported successfully. 0 document(s) failed to import.
```

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



```
iti> db.zips.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: '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

```
iti> db.zips.find({pop:{$gte:1000}})
[
  {
    _id: '01020',
    city: 'CHICOPEE',
    loc: [ -72.576142, 42.176443 ],
    pop: 31495,
    state: 'MA'
  },
  {
    _id: '01033',
    city: 'GRANBY',
    loc: [ -72.520001, 42.255704 ],
    pop: 5526,
    state: 'MA'
  },
  {
    _id: '01034',
    city: 'TOLLAND',
    loc: [ -72.908793, 42.070234 ],
    pop: 1652,
    state: 'MA'
  },
  {
    _id: '01027',
    city: 'MOUNT TOM',
    loc: [ -72.679921, 42.264319 ],
    pop: 16864,
    state: 'MA'
  },
```

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

```
iti> db.zips.updateMany({},{$set:{check:false}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 29353,
  modifiedCount: 29353,
  upsertedCount: 0
}
```

```
iti> db.zips.updateMany({state:{$in:["PA","VA"]}},{$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.

```
iti> db.zips.find({loc:{$gt:55,$lt:66}},{pop:1,_id:0})
[
  { pop: 15891 }, { pop: 14436 },
  { pop: 12534 }, { pop: 7907 },
  { pop: 20128 }, { pop: 32383 },
  { pop: 7979 },  { pop: 17094 },
  { pop: 29857 }, { pop: 8116 },
  { pop: 18356 }, { pop: 119 },
  { pop: 15192 }, { pop: 481 },
  { pop: 1186 },  { pop: 185 },
  { pop: 285 },   { pop: 352 },
  { pop: 1698 },  { pop: 7188 }
]
```

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

```
iti> db.zips.createIndex( { state: 1 })
state_1
iti> db.zips.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { state: 1 }, name: 'state_1' }
]
```

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

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

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.

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

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.

```
iti>  db.zips.updateMany({city:{$type:"string"}, city:{$exists:false}},{$set:{"_id":"0100100","city":"AG
AWAM","loc":[-72.622739,42.070206],"pop":15338,"state":"MA"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
iti> db.zips.updateMany({city:{$type:"string"},city:{$exists:true}},{$rename:{"city":"country"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 29353,
  modifiedCount: 29353,
  upsertedCount: 0
}
```

Hint: use Variables

****

****

part2

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

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

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

```
iti> db.zips.aggregate([{$match : { state :{$nin:["PA","KA"]}} },{ $limit : 5}])
[
  {
    _id: '99502',
    loc: [ -150.093943, 61.096163 ],
    pop: 15891,
    state: 'AK',
    check: false,
    country: 'ANCHORAGE'
  },
  {
    _id: '99501',
    loc: [ -149.876077, 61.211571 ],
    pop: 14436,
    state: 'AK',
    check: false,
    country: 'ANCHORAGE'
  },
  {
    _id: '99503',
    loc: [ -149.893844, 61.189953 ],
    pop: 12534,
    state: 'AK',
    check: false,
    country: 'ANCHORAGE'
  },
  {
    _id: '99506',
    loc: [ -149.812667, 61.251531 ],
    pop: 7907,
    state: 'AK',
    check: false,
    country: 'ELMENDORF AFB'
  },
  {
    _id: '99507',
    loc: [ -149.828912, 61.153543 ],
    pop: 20128,
    state: 'AK',
    check: false,
    country: 'ANCHORAGE'
  }
]
```

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

```
iti> db.zips.aggregate( [ { $match: { state: "AK", "loc.1": { $gt: 55, $lt: 65 } } }, { $group: { _id: '
$state', totalPop: { $sum: "$pop" } } }])
[ { _id: 'AK', totalPop: 524636 } ]
```

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

```
ti> db.zips.aggregate( [ { $match: { state: { $in: ["PA", "KA"] } } }, { $sort: { pop: 1 } }, { $skip:
 }])

 {
   _id: '16334',
   loc: [ -79.445929, 41.326077 ],
   pop: 32.4,
   state: 'PA',
   check: true,
   country: 'MARBLE'
 },
 {
   _id: '16871',
   loc: [ -78.034056, 41.186798 ],
   pop: 40.8,
   state: 'PA',
   check: true,
   country: 'POTTERSDALE'
 },
 {
   _id: '16217',
   loc: [ -79.19708, 41.338366 ],
   pop: 43.199999999999996,
   state: 'PA',
   check: true,
   country: 'COOKSBURG'
 },
 {
   _id: '16331',
   loc: [ -79.588249, 41.290215 ],
   pop: 51.6,
   state: 'PA',
   check: true,
   country: 'KOSSUTH'
 },
 {
   _id: '15730',
   loc: [ -78.922015, 40.951816 ],
   pop: 52.8,
   state: 'PA',
   check: true,
   country: 'COOLSPRING'
 },
 {
   _id: '15780',
   loc: [ -79.083263, 40.922851 ],
   pop: 52.8,
   state: 'PA',
```

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

```
iti> db.zips.aggregate( [ { $group: { _id: '$state', MaxPopCountary: { $max: "$pop" }, MinPopCountary: {
 $min: "$pop" } } }, { $out: 'mypop' }])

iti> db.mypop.find().limit(2)
[
  { _id: 'IN', MaxPopCountary: 67851.59999999999, MinPopCountary: 90 },
  {
    _id: 'ND',
    MaxPopCountary: 50634,
    MinPopCountary: 14.399999999999999
  }
]
```

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

```
iti> db.zips.aggregate( [ { $group: { _id: '$state', avgPop: { $avg: "$pop" } } }])
[
  { _id: 'PA', avgPop: 9779.130041152262 },
  { _id: 'IN', avgPop: 9841.661538461538 },
  { _id: 'SC', avgPop: 11954.410285714284 },
  { _id: 'ND', avgPop: 1958.8910485933504 },
  { _id: 'ID', avgPop: 4951.224590163934 },
  { _id: 'TX', avgPop: 12197.199999999999 },
  { _id: 'CO', avgPop: 9547.115942028986 },
  { _id: 'OR', avgPop: 8882.253125 },
  { _id: 'NV', avgPop: 13867.303846153845 },
  { _id: 'NC', avgPop: 11282.786382978722 },
  { _id: 'FL', avgPop: 18935.289552238806 },
  { _id: 'VT', avgPop: 2779.0518518518516 },
  { _id: 'MI', avgPop: 12733.283561643837 },
  { _id: 'DC', avgPop: 30345 },
  { _id: 'MN', avgPop: 5949.63537414966 },
  { _id: 'RI', avgPop: 17447.26956521739 },
  { _id: 'SD', avgPop: 2173.115625 },
  { _id: 'MS', avgPop: 8506.499173553719 },
  { _id: 'WI', avgPop: 8198.495530726257 },
  { _id: 'NE', avgPop: 3299.245296167247 }
]
```

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

```
iti> db.zips.aggregate( [ { $sort: { state: 1, country: 1 } }])
[
  {
    _id: '99615',
    loc: [ -152.500169, 57.781967 ],
    pop: 13309,
    state: 'AK',
    check: false,
    country: 'AKHIOK'
  },
  {
    _id: '99551',
    loc: [ -161.39233, 60.891854 ],
    pop: 481,
    state: 'AK',
    check: false,
    country: 'AKIACHAK'
  },
  {
    _id: '99552',
    loc: [ -161.199325, 60.890632 ],
    pop: 285,
    state: 'AK',
    check: false,
    country: 'AKIAK'
  },
  {
    _id: '99553',
    loc: [ -165.785368, 54.143012 ],
    pop: 589,
    state: 'AK',
    check: false,
    country: 'AKUTAN'
  },
  {
    _id: '99554',
    loc: [ -164.60228, 62.746967 ],
    pop: 1186,
    state: 'AK',
```

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

```
iti> db.zips.aggregate( [ { $sort: { state: -1, country: -1 } }])
[
  {
    _id: '82244',
    loc: [ -104.353507, 41.912018 ],
    pop: 808.8,
    state: 'WY',
    check: false,
    country: 'YODER'
  },
  {
    _id: '82732',
    loc: [ -105.532327, 43.829349 ],
    pop: 2558.4,
    state: 'WY',
    check: false,
    country: 'WRIGHT'
  },
  {
    _id: '82401',
    loc: [ -107.95626, 44.013796 ],
    pop: 9231.6,
    state: 'WY',
    check: false,
    country: 'WORLAND'
  },
  {
    _id: '83014',
    loc: [ -110.874199, 43.49922 ],
    pop: 1318.8,
    state: 'WY',
    check: false,
    country: 'WILSON'
  },
  {
    _id: '82201',
    loc: [ -104.967852, 42.049467 ],
    pop: 7142.4,
    state: 'WY',
    check: false,
    country: 'WHEATLAND'
  },
  {
    _id: '82450',
    loc: [ -109.432629, 44.47967 ],
    pop: 256.8,
    state: 'WY',
    check: false,
```

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

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

10. Return the average populations for cities in each state

```
iti> db.zips.aggregate( [ { $group: { _id: '$state', avgPop: { $avg: "$pop" } } }])
[
  { _id: 'IN', avgPop: 9841.661538461538 },
  { _id: 'ND', avgPop: 1958.8910485933504 },
  { _id: 'ID', avgPop: 4951.224590163934 },
  { _id: 'SC', avgPop: 11954.410285714284 },
  { _id: 'TX', avgPop: 12197.199999999999 },
  { _id: 'CO', avgPop: 9547.115942028986 },
  { _id: 'NC', avgPop: 11282.786382978722 },
  { _id: 'NV', avgPop: 13867.303846153845 },
  { _id: 'FL', avgPop: 18935.289552238806 },
  { _id: 'SD', avgPop: 2173.115625 },
  { _id: 'DC', avgPop: 30345 },
  { _id: 'VT', avgPop: 2779.0518518518516 },
  { _id: 'MN', avgPop: 5949.63537414966 },
  { _id: 'MI', avgPop: 12733.283561643837 },
  { _id: 'OR', avgPop: 8882.253125 },
  { _id: 'RI', avgPop: 17447.26956521739 },
  { _id: 'MS', avgPop: 8506.499173553719 },
  { _id: 'AK', avgPop: 2792.7128205128206 },
  { _id: 'WI', avgPop: 8198.495530726257 },
  { _id: 'NY', avgPop: 11279.248902821317 }
]
Type "it" for more
```