MongoDB Lab2

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

```
D:\ITI\MongoDB\code\day2>mongoimport --host localhost --db iti --collection zips --file zips.json
2023-02-24T12:56:08.027+0200    connected to: mongodb://localhost/
2023-02-24T12:56:08.604+0200    29353 document(s) imported successfully. 0 document(s) failed to import.
```

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

```
Type "it" for more
iti> db.zips.find({state:"NY"})
[
  {
```

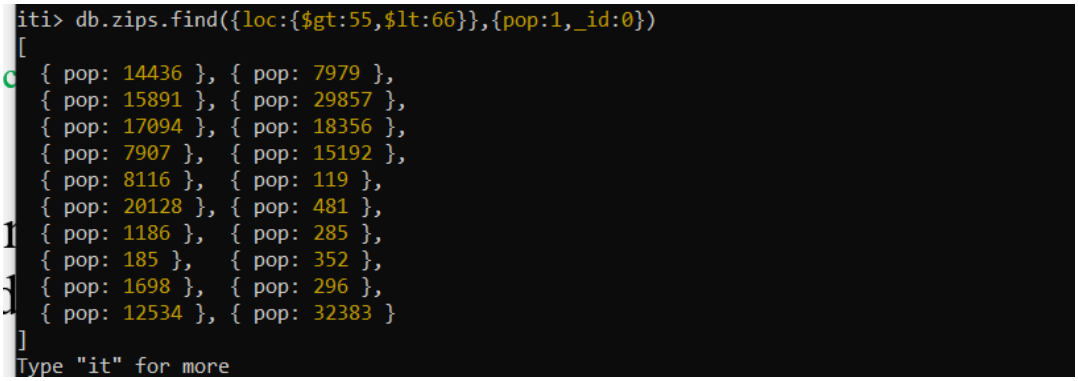
3 – find all zip codes whose population is greater than or equal to 1000

```
iti> db.zips.find({$or:[{pop:{$gt:1000}},{pop:1000}]})
[
```

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

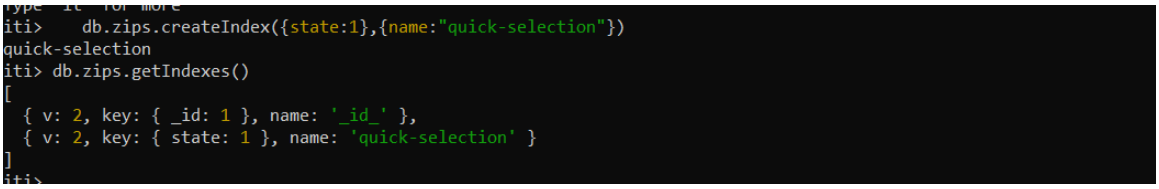
```
iti> db.zips.updateMany({$or:[{state:"PA"},{state:"VA"}]} , {$set:{"check":"true"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2274,
  modifiedCount: 2274,
  upsertedCount: 0
}
```

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



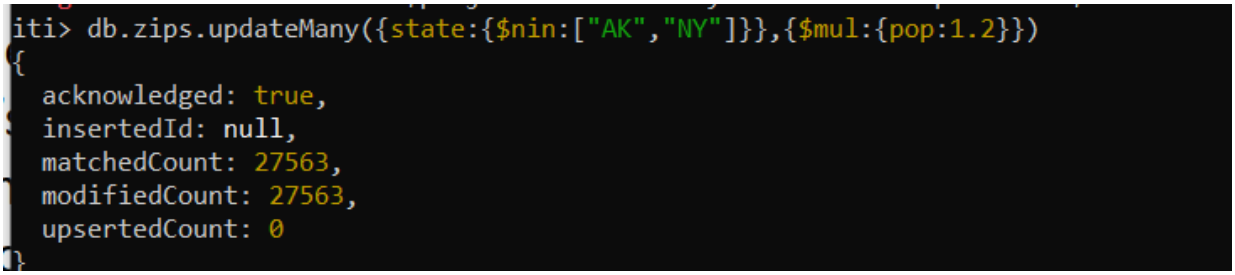
```
iti> db.zips.find({loc:{$gt:55,$lt:66}},{pop:1,_id:0})
[
  { pop: 14436 }, { pop: 7979 },
  { pop: 15891 }, { pop: 29857 },
  { pop: 17094 }, { pop: 18356 },
  { pop: 7907 },  { pop: 15192 },
  { pop: 8116 },  { pop: 119 },
  { pop: 20128 }, { pop: 481 },
  { pop: 1186 },  { pop: 285 },
  { pop: 185 },   { pop: 352 },
  { pop: 1698 },  { pop: 296 },
  { pop: 12534 }, { pop: 32383 }
]
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.



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

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({"loc.0":{$lt:71},state:{$ne:"MA"},pop:{$lt:200}},{$set:{pop:0}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  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 documet in the database but change the _id to avoid duplications.

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

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

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

```
iti> db.zips.find({state:{$nin:["PA","KA"]}}).limit(5)
[
  {
    _id: '99501',
    loc: [ -149.876077, 61.211571 ],
    pop: 14436,
    state: 'AK',
    country: 'ANCHORAGE'
  },
  {
    _id: '99505',
    loc: [ -149.675454, 61.275256 ],
    pop: 7979,
```

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

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

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

```
iti> db.zips.aggregate([ {$match:{state:{$in:["KA","PA"]}}},{$sort:{pop:1}} ]).skip(7)
[
  { pop: 17094,
    _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'
```

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",greatest:{$max:"$pop"},smallest:{$min:"$pop"}}},{$out:'mypop'})
```

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

```
iti> db.zips.aggregate({$group:{_id:"$country",AvgPopulation:{$avg:"$pop"}}})
[
  { _id: 'FISHER', AvgPopulation: 1196.2 },
  { _id: 'PIERMONT', AvgPopulation: 1310.6 },
  { _id: 'SHELTER ISLAND H', AvgPopulation: 1119 },
  { _id: 'ALTAMONT', AvgPopulation: 2295.742857142857 },
  { _id: 'DOLA', AvgPopulation: 307.2 },
  { _id: 'BOMBAY', AvgPopulation: 1042 },
  { _id: 'WHEATON', AvgPopulation: 30409.499999999996 },
  { _id: 'CUMBERLAND CITY', AvgPopulation: 1449.6 },
  { _id: 'BOOMER', AvgPopulation: 2575.2 },
  { _id: 'ORISKANY FALLS', AvgPopulation: 2192 },
  { _id: 'FORT SUPPLY', AvgPopulation: 1466.3999999999999 },
  { _id: 'PINOLE', AvgPopulation: 20304 },
  { _id: 'BLANKET', AvgPopulation: 2314.7999999999997 },
```

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',
    country: 'AKHIOK'
  },
  {
```

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

```
Type "it" for more
iti> db.zips.aggregate( {$sort:{state:-1,country:-1}} )
[
  {
    _id: '82244',
    loc: [ -104.353507, 41.912018 ],
    pop: 808.8,
    state: 'WY',
    country: 'YODER'
  },
  {
```

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: { $and: [{ state: { $in: ["CA", "NY"] } }, { pop: { $gt: 25000 } }] } }, { $group:
{ _id: '$state', average: { $avg: "$pop" } } })
[
  { _id: 'NY', average: 44494.818930041154 },
  { _id: 'CA', average: 46673.271224165335 }
]
```

10. Return the average populations for cities in each state

```
iti> db.zips.aggregate( { $group: { _id: '$state', average: { $avg: "$pop" } } })
[
  { _id: 'TN', average: 10054.550515463916 },
  { _id: 'MI', average: 12733.283561643837 },
  { _id: 'CT', average: 14998.247908745247 },
  { _id: 'MD', average: 13661.082857142857 },
  { _id: 'WI', average: 8198.495530726257 },
  { _id: 'ID', average: 4951.224590163934 },
  { _id: 'MT', average: 3053.304458598726 },
  { _id: 'NY', average: 11279.248902821317 },
  { _id: 'WY', average: 3887.382857142857 },
  { _id: 'UT', average: 10084.975609756097 },
  { _id: 'CA', average: 23552.683377308706 },
  { _id: 'CO', average: 9547.115942028986 },
  { _id: 'GA', average: 12242.297952755904 },
```