1) Download the following json file import it

```
C:\Program Files\MongoDB\Tools\100\bin>mongoimport --host localhost --db iti --collection zips --file zips.json
2023-02-24T15:30:45.883+0200    connected to: mongodb://localhost/
2023-02-24T15:30:46.372+0200    29353 document(s) imported successfully. 0 document(s) failed to import.

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

2) Find all document which contains data related to “NY”

```
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'
  },
```

3) Find all zip codes whose population is greater than or equal 1000

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

4) Add new Boolean field called “check”

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

5) Find all cities whose latitude between 55 and 65

```
iti> db.zips.find({"loc.1":{$gt:55,$lt:66}},{pop:1,_id:0})
[
  { pop: 14436 }, { pop: 12534 },
  { pop: 32383 }, { pop: 7979 },
  { pop: 7907 },  { pop: 15891 },
  { pop: 18356 }, { pop: 8116 },
  { pop: 17094 }, { pop: 29857 },
  { pop: 15192 }, { pop: 285 },
  { pop: 481 },   { pop: 1186 },
  { pop: 20128 }, { pop: 1698 },
  { pop: 185 },   { pop: 119 },
  { pop: 320 },   { pop: 0 }
]
```

6) Create Index for states

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

7) Increase the population by 0.2 for all cities

```
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”

```
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

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

Part2:

1) Get the sum of population that state in pa,ka

```
iti> db.zips.aggregate([ { $match: { state: { $in: ["PA", "KA"] } } }, { $group: { _id: "$state", totalPop: { $sum: "$p
p" } } }])
[ { _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: '99501',
    loc: [ -149.876077, 61.211571 ],
    pop: 14436,
    state: 'AK',
    country: 'ANCHORAGE'
  },
  {
    _id: '99503',
    loc: [ -149.893844, 61.189953 ],
    pop: 12534,
    state: 'AK',
    country: 'ANCHORAGE'
  },
  {
    _id: '99504',
    loc: [ -149.74467, 61.203696 ],
    pop: 32383,
    state: 'AK',
    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:"country",totalPop:{$sum:"$pop"}}}
... ])
[ { _id: 'country', totalPop: 524570 } ]
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}
... ])
[
  {
    _id: '16334',
    loc: [ -79.445929, 41.326077 ],
    pop: 32.4,
    state: 'PA',
    check: 'true',
    country: 'MARBLE'
  },
  {
    _id: '16871',
    loc: [ -78.034056, 41.186798 ],
```

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'}
... )

iti> show collections
iti.students
mypop
zips
iti> db.mypop.find()
[
  { _id: 'MS', greatest: 56361.6, smallest: 0 },
  { _id: 'AR', greatest: 64238.399999999994, smallest: 0 },
  { _id: 'VA', greatest: 82230, smallest: 0 },
  { _id: 'WA', greatest: 60618, smallest: 2.4 },
```

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: 'WOODBURN', AvgPopulation: 7432.799999999999 },
  { _id: 'CALUMET CITY', AvgPopulation: 43278 },
  { _id: 'STARFORD', AvgPopulation: 1020 },
  { _id: 'YORKTOWN', AvgPopulation: 3144.24 },
  { _id: 'ROWAN', AvgPopulation: 486 },
  { _id: 'PIERCE CITY', AvgPopulation: 3354 },
  { _id: 'ROZET', AvgPopulation: 1080 },
```

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'
  },
  {
    _id: '99551',
    loc: [ -161.39233, 60.891854 ],
    pop: 481,
```

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',
    country: 'YODER'
  },
  {
    _id: '82732',
    loc: [ -105.532327, 43.829349 ],
    pop: 2558.4,
    state: 'WY',
    country: 'WRIGHT'
  },
```

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([
... {$group:{_id:{state:"$state",city:"$city"},pop:{$sum:"$pop"}}},
... {$group:{_id:"$_id.state",avgCityPop:{$avg:"$pop"}}}
... ])
[
  { _id: 'RI', avgCityPop: 1203861.5999999999 },
  { _id: 'IA', avgCityPop: 3331704 },
  { _id: 'AZ', avgCityPop: 4398273.6 },
  { _id: 'NC', avgCityPop: 7954364.399999999 },
  { _id: 'NE', avgCityPop: 1893766.7999999998 },
  { _id: 'VA', avgCityPop: 7417774.8 },
  { _id: 'WA', avgCityPop: 5840030.399999999 },
```