*** Querying a MongoDB database ***

1. Make sure you have MongoDB Database Tools installed (https://docs.mongodb.com/database-tools/installation/installation/).

2. Make sure you have a local MongoDB instance running (see lab #2) or use the remote MongoDB Atlas instance (see lab #1).

3. Download zips sample data
https://github.com/ozlerhakan/mongodb-json-files

4. Use mongoimport tool to import the sample dataset
Syntax: mongoimport <options> <connection-string> <file>
mongoimport --db=test --collection=zips
mongodb://mongoadmin:secret@localhost:27888/?authSource=admin zips.json

```
PS C:\Program Files\MongoDB\Server\7.0\bin> mongoimport --db test --collection zips C:\Users\razva\Documents\Master_Poli
\Baze_date\zips.json
2024-05-18T23:42:08.113+0300    connected to: mongodb://localhost/
2024-05-18T23:42:08.440+0300    29353 document(s) imported successfully. 0 document(s) failed to import.
PS C:\Program Files\MongoDB\Server\7.0\bin>
```

5. Connect to MongoDB using shell
mongosh mongodb://mongoadmin:secret@localhost:27888/?authSource=admin

6. Query the sample data set

Count the number of zip codes in the collection
> db.zips.count()

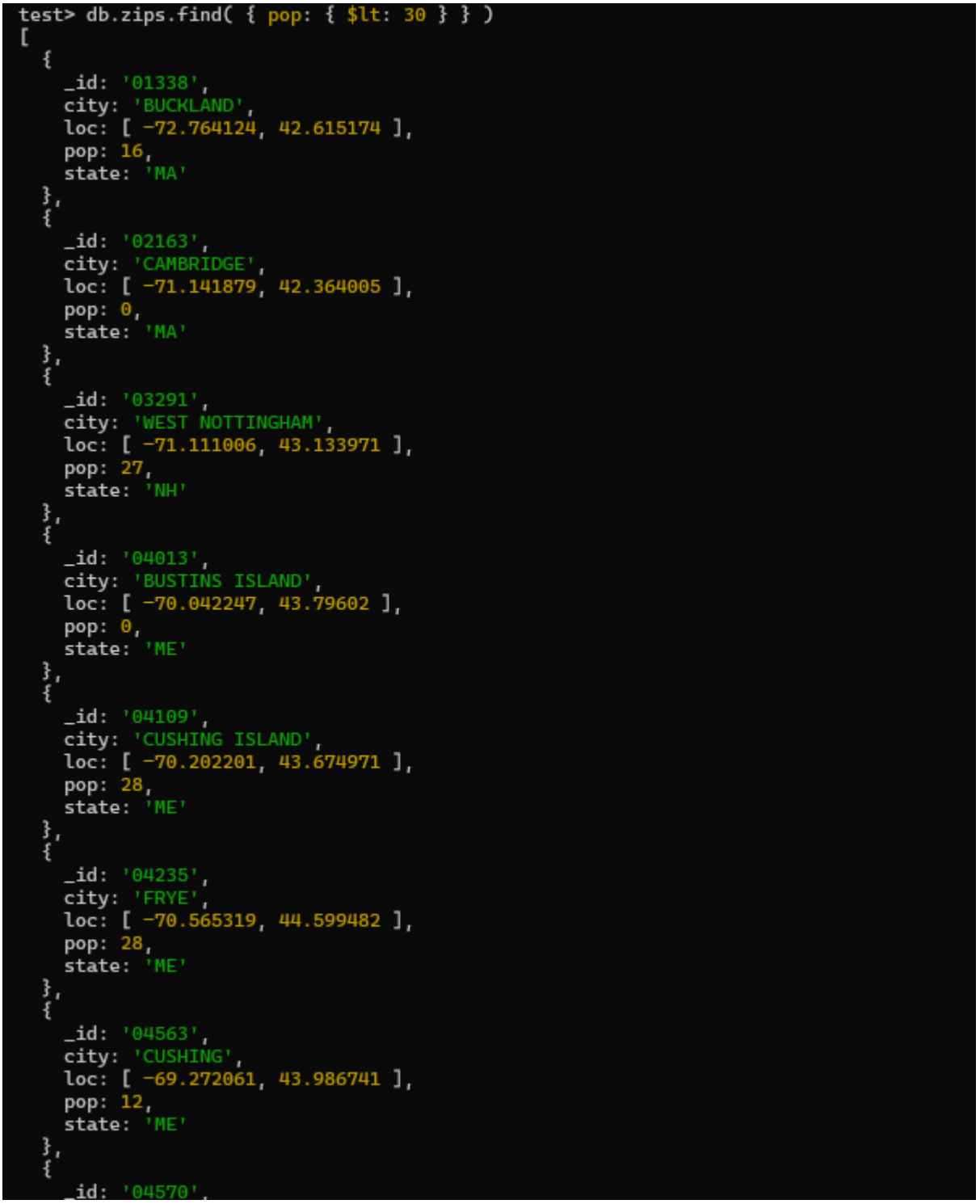
```
PS C:\Users\razva> mongosh
Current Mongosh Log ID: 66491324311d78a3c09e23fd
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2
.0.0
Using MongoDB:          7.0.9
Using Mongosh:          2.0.0
mongosh 2.2.6 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------
   The server generated these startup warnings when booting
   2024-05-18T23:27:27.210+03:00: Access control is not enabled for the database. Read and write access to data and conf
iguration is unrestricted
------

test> show collections
zips
test> db.zips.count()
DeprecationWarning: Collection.count() is deprecated. Use countDocuments or estimatedDocumentCount.
29353
test>
```

Find cities with the population less than 30
> db.zips.find( { pop: { $lt: 30 } } )




Find cities with the population more than 100k
> db.zips.find( { pop: { $gt: 100000 } } )

```
Type "it" for more
test> db.zips.find( { pop: { $gt: 100000 } } )
[
  {
    _id: '10021',
    city: 'NEW YORK',
    loc: [ -73.958805, 40.768476 ],
    pop: 106564,
    state: 'NY'
  },
  {
    _id: '10025',
    city: 'NEW YORK',
    loc: [ -73.968312, 40.797466 ],
    pop: 100027,
    state: 'NY'
  },
  {
    _id: '11226',
    city: 'BROOKLYN',
    loc: [ -73.956985, 40.646694 ],
    pop: 111396,
    state: 'NY'
  },
  {
    _id: '60623',
    city: 'CHICAGO',
    loc: [ -87.7157, 41.849015 ],
    pop: 112047,
    state: 'IL'
  }
]
test>
```

Find cities with the population more than 100k, but return only the city and population

> db.zips.find( { pop: { $gt: 100000 }}, { city: 1, pop: 1} )

```
test> db.zips.find( { pop: { $gt: 100000 }}, { city: 1, pop: 1} )
[
  { _id: '10021', city: 'NEW YORK', pop: 106564 },
  { _id: '10025', city: 'NEW YORK', pop: 100027 },
  { _id: '11226', city: 'BROOKLYN', pop: 111396 },
  { _id: '60623', city: 'CHICAGO', pop: 112047 }
]
test>
```

Find distinct states

> db.zips.distinct("state")

```
test> db.zips.distinct("state")
[
  'AK', 'AL', 'AR', 'AZ', 'CA', 'CO', 'CT',
  'DC', 'DE', 'FL', 'GA', 'HI', 'IA', 'ID',
  'IL', 'IN', 'KS', 'KY', 'LA', 'MA', 'MD',
  'ME', 'MI', 'MN', 'MO', 'MS', 'MT', 'NC',
  'ND', 'NE', 'NH', 'NJ', 'NM', 'NV', 'NY',
  'OH', 'OK', 'OR', 'PA', 'RI', 'SC', 'SD',
  'TN', 'TX', 'UT', 'VA', 'VT', 'WA', 'WI',
  'WV', 'WY'
]
test>
```

Find states that have cities with a population over 80k

> db.zips.distinct("state", {pop: { $gt: 80000}})

```
test> db.zips.distinct("state", {pop: { $gt: 80000}})
[ 'CA', 'IL', 'MI', 'NY', 'PA' ]
test>
```

NEXT:
We will update and delete documents in the sample data inside our MongoDB instance. Familiarize yourself with the update and delete MongoDB operators (see reference docs at https://docs.mongodb.com/manual/reference/operator/update/)