*** Updating and Deleting Data. Atomic Operations. ***

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

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

6. Query and update the sample data set (See reference docs at https://docs.mongodb.com/manual/reference/operator/update/)

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

```
test> db.zips.count()
29353
test>
```

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

```
29353
test> db.zips.find( { pop: { $lt: 30 } } )
[
  {
    _id: '01338',
    city: 'BUCKLAND',
    loc: [ -72.764124, 42.615174 ],
    pop: 16,
    state: 'MA'
  },
  {
    _id: '02163',
    city: 'CAMBRIDGE',
    loc: [ -71.141879, 42.364005 ],
    pop: 0,
    state: 'MA'
  },
  {
    _id: '03291',
    city: 'WEST NOTTINGHAM',
    loc: [ -71.111006, 43.133971 ],
    pop: 27,
    state: 'NH'
  },
  {
    _id: '04013',
    city: 'BUSTINS ISLAND',
    loc: [ -70.042247, 43.79602 ],
    pop: 0,
    state: 'ME'
  },
  {
    _id: '04109',
    city: 'CUSHING ISLAND',
    loc: [ -70.202201, 43.674971 ],
    pop: 28,
    state: 'ME'
  },
  {
    _id: '04235',
    city: 'FRYE',
    loc: [ -70.565319, 44.599482 ],
    pop: 28,
    state: 'ME'
  },
  {
    _id: '04563',
    city: 'CUSHING',
    loc: [ -69.272061, 43.986741 ],
    pop: 12,
    state: 'ME'
  },
  {
    _id: '04570',
    city: 'SQUIRREL ISLAND',
    loc: [ -69.630974, 43.809031 ],
    pop: 3,
    state: 'ME'
  },
  {
    _id: '05405',
    city: 'UNIV OF VERMONT',
    loc: [ -73.2002, 44.477733 ],
    pop: 0,
```

The city of BUSTINS ISLAND has a new population of 5 (was 0) so we update it
> Update one document
db.zips.updateOne({ _id: "04013" }, { $set: { "pop": 5} })
Now check again the population of BUSTINS ISLAND. Should be 5.

```
test> db.zips.updateOne({ _id: "04013" }, { $set: { "pop": 5} })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
test>
```

All cities with a population under 30 should get a new field with the value "tinycity" equal to true
> db.zips.updateMany({ pop: { $lt: 30 } }, { $set: { "tinycity": true} })

```
test> db.zips.updateMany({ pop: { $lt: 30 } }, { $set: { "tinycity": true} })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 233,
  modifiedCount: 233,
  upsertedCount: 0
}
test>
```

Add 50 to the population of all cities
> db.zips.updateMany({ }, { $inc: { "pop": 50} })

```
}
test> db.zips.updateMany({ }, { $inc: { "pop": 50} })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 29353,
  modifiedCount: 29353,
  upsertedCount: 0
}
test>
```

Remove the "tinycity" field from all documents
> db.zips.updateMany({ }, { $unset: { tinycity:"" } })

```
test> db.zips.updateMany({ }, { $unset: { tinycity:"" } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 29353,
  modifiedCount: 233,
  upsertedCount: 0
}
test>
```

Find the city with ZIP 60623...
> db.zips.find({ _id: "60623"})

```
test> db.zips.find({ _id: "60623"})
[
  {
    _id: '60623',
    city: 'CHICAGO',
    loc: [ -87.7157, 41.849015 ],
    pop: 112097,
    state: 'IL'
  }
]
test>
```

... and delete it

db.zips.deleteOne({ _id: "60623"})

```
test> db.zips.deleteOne({ _id: "60623"})
{ acknowledged: true, deletedCount: 1 }
test> db.zips.find({ _id: "60623"})

test>
```

Delete all cities in the state Illinois

> db.zips.deleteMany({ state: "IL"})

```
test> db.zips.deleteMany({ state: "IL"})
{ acknowledged: true, deletedCount: 1236 }
test>
```

Find the city with ZIP 11226 and increase its population by 1500 (in atomic operation)

```
> db.zips.findAndModify({
    query: { _id: "11226" },
    sort: { pop: 1 },
    update: { $inc: { pop: 1500 } },
    upsert: true
})
```

```
test> db.zips.findAndModify({
... query: { _id: "11226" },
... sort: { pop: 1 },
... update: { $inc: { pop: 1500 } },
... upsert: true
... })
{
  _id: '11226',
  city: 'BROOKLYN',
  loc: [ -73.956985, 40.646694 ],
  pop: 111446,
  state: 'NY'
}
test>
```

Try to find a city that doesn't exist (with ZIP 300672). It will be "upserted" (inserted if it doesn't exist, updated if it does)

```
> db.zips.findAndModify({
    query: { _id: "300672" },
    sort: { pop: 1 },
```

```
    update: { $inc: { pop: 1500 } },
    upsert: true
})
```

```
test> db.zips.findAndModify({
... query: { _id: "300672" },
... sort: { pop: 1 },
... update: { $inc: { pop: 1500 } },
...  upsert: true
... })
null
```

Check the newly inserted city

```
> db.zips.find( { _id: "300672" } )
```

```
test> db.zips.find( { _id: "300672" } )
[ { _id: '300672', pop: 1500 } ]
test>
```

NEXT:

We will store some binary files (movies, mind you!) into our MongoDB instance. Familiarize yourself with GridFS (see reference docs at https://docs.mongodb.com/manual/core/gridfs/).