COMP 306: Database Management Systems

Spring 2023 - Homework 5

Yakup Enes Güven – 64045

Question 1)

```
yakupenesguven@admin HW5 % brew services start mongodb-community@6.0
==> Successfully started `mongodb-community` (label: homebrew.mxcl.mongodb-community)
yakupenesguven@admin HW5 % mongosh
Current Mongosh Log ID: 646e3641e5f27c99b6358d4d
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.8.2
Using MongoDB:          6.0.5
Using Mongosh:          1.8.2

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

------
   The server generated these startup warnings when booting
   2023-05-24T19:07:24.502+03:00: Access control is not enabled for the database. Read and write access to data and configur
ation is unrestricted
   2023-05-24T19:07:24.502+03:00: Soft rlimits for open file descriptors too low
------

------
   Enable MongoDB's free cloud-based monitoring service, which will then receive and display
   metrics about your deployment (disk utilization, CPU, operation statistics, etc).

   The monitoring data will be available on a MongoDB website with a unique URL accessible to you
   and anyone you share the URL with. MongoDB may use this information to make product
   improvements and to suggest MongoDB products and deployment options to you.

   To enable free monitoring, run the following command: db.enableFreeMonitoring()
   To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
------

test> show dbs
admin   40.00 KiB
config  12.00 KiB
local   72.00 KiB
test>
```

Figure 1: brew services start mongodb-community@6.0

```
test> use hw5
switched to db hw5
hw5>
```

Figure 2: use hw5

```
yakupenesguven@admin HW5 % mongoimport --db hw5 --collection zipcodes --file zips.json
2023-05-24T19:09:52.016+0300    connected to: mongodb://localhost/
2023-05-24T19:09:52.522+0300    29353 document(s) imported successfully. 0 document(s) failed to import.
yakupenesguven@admin HW5 %
```

Figure 3: mongoimport --db hw5 --collection zipcodes --file zips.json

```
hw5> show collections
zipcodes
hw5>
```

Figure 4: show collections

```
hw5> db.zipcodes.find().limit(15)
[
  {
    _id: '01012',
    city: 'CHESTERFIELD',
    loc: [ -72.833309, 42.38167 ],
    pop: 177,
    state: 'MA'
  },
  {
    _id: '01013',
    city: 'CHICOPEE',
    loc: [ -72.607962, 42.162046 ],
    pop: 23396,
    state: 'MA'
  },
  {
    _id: '01002',
    city: 'CUSHMAN',
    loc: [ -72.51565, 42.377017 ],
    pop: 36963,
    state: 'MA'
  },
  {
    _id: '01020',
    city: 'CHICOPEE',
    loc: [ -72.576142, 42.176443 ],
    pop: 31495,
    state: 'MA'
  },
  {
    _id: '01022',
    city: 'WESTOVER AFB',
    loc: [ -72.558657, 42.196672 ],
    pop: 1764,
    state: 'MA'
  },
  {
    _id: '01026',
    city: 'CUMMINGTON',
    loc: [ -72.905767, 42.435296 ],
    pop: 1484,
    state: 'MA'
  },
  {
    _id: '01027',
    city: 'MOUNT TOM',
    loc: [ -72.679921, 42.264319 ],
    pop: 16864,
    state: 'MA'
  },
  {
    _id: '01028',
    city: 'EAST LONGMEADOW',
    loc: [ -72.505565, 42.067203 ],
    pop: 13367,
    state: 'MA'
  },
  {
    _id: '01011',
    city: 'CHESTER',
    loc: [ -72.988761, 42.279421 ],
    pop: 1688,
    state: 'MA'
  },
  {
    _id: '01030',
    city: 'FEEDING HILLS',
    loc: [ -72.675077, 42.07182 ],
    pop: 11985,
    state: 'MA'
  },
  {
    _id: '01031',
    city: 'GILBERTVILLE',
    loc: [ -72.198585, 42.332194 ],
    pop: 2385,
    state: 'MA'
  },
  {
    _id: '01032',
    city: 'GOSHEN',
    loc: [ -72.844092, 42.466234 ],
    pop: 122,
    state: 'MA'
  },
  {
    _id: '01035',
    city: 'HADLEY',
    loc: [ -72.571499, 42.36062 ],
    pop: 4231,
    state: 'MA'
  },
  {
    _id: '01034',
    city: 'TOLLAND',
    loc: [ -72.908793, 42.070234 ],
    pop: 1652,
    state: 'MA'
  },
  {
    _id: '01036',
    city: 'HAMPDEN',
    loc: [ -72.431823, 42.064756 ],
    pop: 4709,
    state: 'MA'
  }
]
```

Figure 5: db.zipcodes.find().limit(15)

Question 2)

```
hw5> db.zipcodes.find({
...    "state": "CA",
...    "loc.1": { $gt: 35 },
...    "pop": { $gt: 50000 }
... }).sort({ "pop": -1 }).limit(5)
[
  {
    _id: '94501',
    city: 'COAST GUARD ISLA',
    loc: [ -122.260516, 37.764783 ],
    pop: 76110,
    state: 'CA'
  },
  {
    _id: '94110',
    city: 'SAN FRANCISCO',
    loc: [ -122.415344, 37.750858 ],
    pop: 70770,
    state: 'CA'
  },
  {
    _id: '95351',
    city: 'MODESTO',
    loc: [ -121.006033, 37.625022 ],
    pop: 69275,
    state: 'CA'
  },
  {
    _id: '95076',
    city: 'LA SELVA BEACH',
    loc: [ -121.763437, 36.920515 ],
    pop: 68295,
    state: 'CA'
  },
  {
    _id: '94533',
    city: 'FAIRFIELD',
    loc: [ -122.03565, 38.267084 ],
    pop: 65455,
    state: 'CA'
  }
]
hw5>
```

Figure 6:db.zipcodes.find().sort().limit()

Question 3)

```
hw5> db.zipcodes.find({
...   $or: [
...     { state: { $ne: "CA" } },
...     { loc: { $elemMatch: { $lt: -120, $gt: 40 } } }
...   ]
... })
[
  {
    _id: '01012',
    city: 'CHESTERFIELD',
    loc: [ -72.833309, 42.38167 ],
    pop: 177,
    state: 'MA'
  },
  {
    _id: '01013',
    city: 'CHICOPEE',
    loc: [ -72.607962, 42.162046 ],
    pop: 23396,
    state: 'MA'
  },
  {
    _id: '01002',
    city: 'CUSHMAN',
    loc: [ -72.51565, 42.377017 ],
    pop: 36963,
    state: 'MA'
  },
  {
    _id: '01020',
    city: 'CHICOPEE',
    loc: [ -72.576142, 42.176443 ],
    pop: 31495,
    state: 'MA'
  },
  {
    _id: '01022',
    city: 'WESTOVER AFB',
    loc: [ -72.558657, 42.196672 ],
    pop: 1764,
    state: 'MA'
  },
  {
    _id: '01026',
    city: 'CUMMINGTON',
    loc: [ -72.905767, 42.435296 ],
    pop: 1484,
    state: 'MA'
  },
  {
    _id: '01027',
    city: 'MOUNT TOM',
    loc: [ -72.679921, 42.264319 ],
    pop: 16864,
    state: 'MA'
  },
```

Figure 7: db.zipcodes.find()

Question 4)

```
hw5> db.zipcodes.aggregate([
...   {
...     $sort: { pop: -1 }
...   },
...   {
...     $limit: 5
...   },
...   {
...     $project: { _id: 0, city: 1, pop: 1 }
...   }
... ])
[
  { city: 'CHICAGO', pop: 112047 },
  { city: 'BROOKLYN', pop: 111396 },
  { city: 'NEW YORK', pop: 106564 },
  { city: 'NEW YORK', pop: 100027 },
  { city: 'BELL GARDENS', pop: 99568 }
]
```

Figure 8: db.zipcodes.aggregate()

Question 5)

```
hw5> db.zipcodes.aggregate([
...   {
...     $group: {
...       _id: "$state",
...       count: { $sum: 1 }
...     }
...   },
...   {
...     $match: {
...       count: { $gt: 300, $lt: 500 }
...     }
...   },
...   {
...     $sort: {
...       count: 1
...     }
...   }
... ])
[
  { _id: 'MT', count: 314 },
  { _id: 'SC', count: 350 },
  { _id: 'MS', count: 363 },
  { _id: 'OR', count: 384 },
  { _id: 'SD', count: 384 },
  { _id: 'ND', count: 391 },
  { _id: 'ME', count: 410 },
  { _id: 'CO', count: 414 },
  { _id: 'MD', count: 420 },
  { _id: 'LA', count: 464 },
  { _id: 'MA', count: 474 },
  { _id: 'WA', count: 484 }
]
```

Figure 9: db.zipcodes.aggregate()

Question 6)

```
hw5> db.createCollection("customers", {
...    validator: {
...       $jsonSchema: {
...          bsonType: "object",
...          required: ["name", "zipcode", "avg_rating"],
...          properties: {
...             name: {
...                bsonType: "string"
...             },
...             zipcode: {
...                bsonType: "string"
...             },
...             avg_rating: {
...                bsonType: "double",
...                minimum: 0.0,
...                maximum: 10.0
...             },
...             last_order: {
...                bsonType: "object",
...                properties: {
...                   year: {
...                      bsonType: "int"
...                   },
...                   tags: {
...                      bsonType: "array",
...                      items: {
...                         bsonType: "string"
...                      }
...                   }
...                }
...             }
...          }
...       }
...    }
... })
{ ok: 1 }
```

Figure 10: db.createCollection("customers",)

Question 7)

```
hw5> db.customers.insertMany([
...     {
...         name: "Yakup Enes Güven",
...         zipcode: "99503",
...         avg_rating: 8.3
...     },
...     {
...         name: "Andrei T.",
...         zipcode: "90025",
...         avg_rating: 3.5,
...         last_order: {
...             year: 2009
...         }
...     },
...     {
...         name: "Bela T.",
...         zipcode: "33126",
...         avg_rating: 4.9,
...         last_order: {
...             year: 2019,
...             tags: ["art", "melancholy"]
...         }
...     },
...     {
...         name: "Nuri Bilge C.",
...         zipcode: "90010",
...         avg_rating: 6.5,
...         last_order: {
...             year: 3005
...         }
...     }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("646e3a40e5f27c99b6358d4e"),
    '1': ObjectId("646e3a40e5f27c99b6358d4f"),
    '2': ObjectId("646e3a40e5f27c99b6358d50"),
    '3': ObjectId("646e3a40e5f27c99b6358d51")
  }
}
```

Figure 11: db.customers.insertMany()

```
hw5> db.customers.find({})
[
  {
    _id: ObjectId("646e3a40e5f27c99b6358d4e"),
    name: 'Yakup Enes Güven',
    zipcode: '99503',
    avg_rating: 8.3
  },
  {
    _id: ObjectId("646e3a40e5f27c99b6358d4f"),
    name: 'Andrei T.',
    zipcode: '90025',
    avg_rating: 3.5,
    last_order: { year: 2009 }
  },
  {
    _id: ObjectId("646e3a40e5f27c99b6358d50"),
    name: 'Bela T.',
    zipcode: '33126',
    avg_rating: 4.9,
    last_order: { year: 2019, tags: [ 'art', 'melancholy' ] }
  },
  {
    _id: ObjectId("646e3a40e5f27c99b6358d51"),
    name: 'Nuri Bilge C.',
    zipcode: '90010',
    avg_rating: 6.5,
    last_order: { year: 3005 }
  }
]
```

Figure 12: db.customers.find()

Question 8)

```
hw5> db.customers.deleteMany({
...     "last_order.year": { $gt: 2023 }
... })
{ acknowledged: true, deletedCount: 1 }
hw5> db.customers.find({})
[
  {
    _id: ObjectId("646e3a40e5f27c99b6358d4e"),
    name: 'Yakup Enes Güven',
    zipcode: '99503',
    avg_rating: 8.3
  },
  {
    _id: ObjectId("646e3a40e5f27c99b6358d4f"),
    name: 'Andrei T.',
    zipcode: '90025',
    avg_rating: 3.5,
    last_order: { year: 2009 }
  },
  {
    _id: ObjectId("646e3a40e5f27c99b6358d50"),
    name: 'Bela T.',
    zipcode: '33126',
    avg_rating: 4.9,
    last_order: { year: 2019, tags: [ 'art', 'melancholy' ] }
  }
]
```

Figure 13: db.customers.deleteMany()

Question 9)

```
hw5> db.customers.updateOne(
...     { name: "Yakup Enes Güven" },
...     { $set: { avg_rating: 15 } }
... )
Uncaught:
MongoServerError: Document failed validation
Additional information: {
  failingDocumentId: ObjectId("646e3a40e5f27c99b6358d4e"),
  details: {
    operatorName: '$jsonSchema',
    schemaRulesNotSatisfied: [
      {
        operatorName: 'properties',
        propertiesNotSatisfied: [
          {
            propertyName: 'avg_rating',
            details: [ [Object], [Object] ]
          }
        ]
      }
    ]
  }
}
```

Figure 14: db.customers.updateOne()

The error message indicates that the validation failed for the "avg_rating" field. It states that the schema rules for the "avg_rating" property were not satisfied. Based on the previous information, the "avg_rating" field must be a double between 0.0 and 10.0. In this case, it seems that the attempted update set the "avg_rating" field to a value that is outside the allowed range of 0.0 to 10.0. To resolve this issue, we should update the code with a valid value within the specified range for the "avg_rating" field.

Question 10)

```
hw5> db.customers.aggregate([
...     {
...         $lookup: {
...             from: "zipcodes",
...             localField: "zipcode",
...             foreignField: "_id",
...             as: "zipcode_details"
...         }
...     },
...     {
...         $unwind: "$zipcode_details"
...     },
...     {
...         $project: {
...             _id: 0,
...             Name: "$name",
...             City: "$zipcode_details.city",
...             State: "$zipcode_details.state",
...             Zipcode: "$zipcode_details._id"
...         }
...     },
...     {
...         $sort: {
...             Name: 1
...         }
...     }
... ])
[
  {
    Name: 'Andrei T.',
    City: 'LOS ANGELES',
    State: 'CA',
    Zipcode: '90025'
  },
  { Name: 'Bela T.', City: 'MIAMI', State: 'FL', Zipcode: '33126' },
  {
    Name: 'Yakup Enes Güven',
    City: 'ANCHORAGE',
    State: 'AK',
    Zipcode: '99503'
  }
]
```

Figure 15: db.customers.aggregate()

Question 11)

```
[yakupenesguven@admin HW5 % mongoexport --db hw5 --collection customers --out customers.json
2023-05-24T19:34:29.299+0300    connected to: mongodb://localhost/
2023-05-24T19:34:29.313+0300    exported 3 records
```

Figure 16: mongoexport --db hw --collection customers --out customers.json