

Aayush Shah

19BCE245

11 November 2022

Big Data Analytics

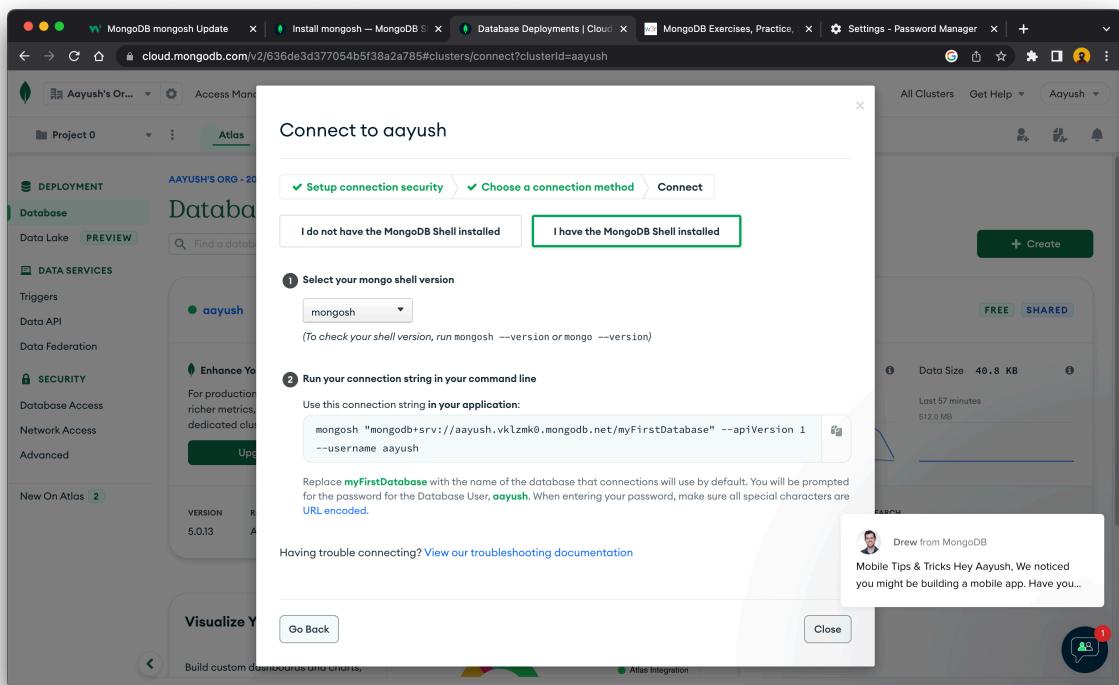
Practical 8

Aim

Setup MongoDB environment in your system. Import Restaurant Dataset and perform CRUD operation.

Using MongoDB Atlas (online database)

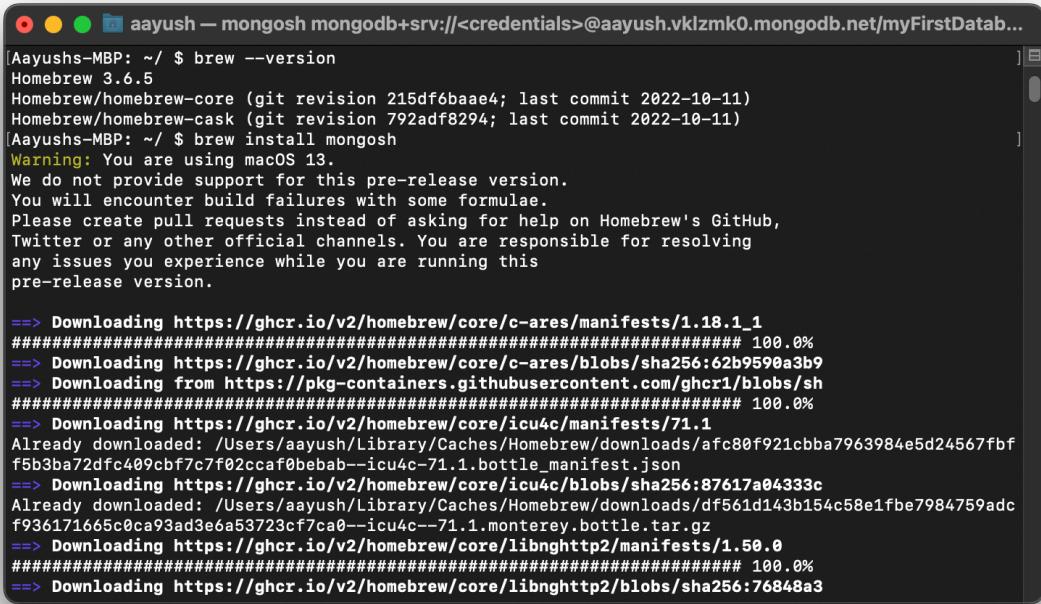
- Creating account and setting up on <https://www.mongodb.com/atlas/database>



- Installing *mongosh* with *HomeBrew*

Also we can use inbuilt mongosh command prompt which comes with *mongodb compass*

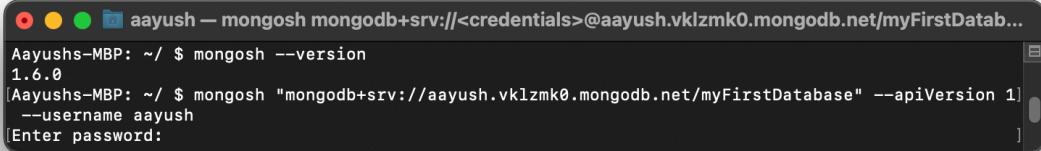
Which can be downloaded through <https://www.mongodb.com/products/compass>



```
[Aayushs-MBP: ~ $ brew --version
Homebrew 3.6.5
Homebrew/homebrew-core (git revision 215df6baae4; last commit 2022-10-11)
Homebrew/homebrew-cask (git revision 792adf8294; last commit 2022-10-11)
[Aayushs-MBP: ~ $ brew install mongosh
Warning: You are using macOS 13.
We do not provide support for this pre-release version.
You will encounter build failures with some formulae.
Please create pull requests instead of asking for help on Homebrew's GitHub,
Twitter or any other official channels. You are responsible for resolving
any issues you experience while you are running this
pre-release version.

==> Downloading https://ghcr.io/v2/homebrew/core/c-ares/manifests/1.18.1_1
#####
 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/c-ares/blobs/sha256:62b9590a3b9
#####
 100.0%
==> Downloading from https://pkg-containers.githubusercontent.com/ghcr1/blobs/sha256:62b9590a3b9
#####
 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/icu4c/manifests/71.1
Already downloaded: /Users/aayush/Library/Caches/Homebrew/downloads/afc80f921cbba7963984e5d24567fbff5b3ba72dfc409cbf7c7f02ccaf0bebab--icu4c-71.1.bottle_manifest.json
==> Downloading https://ghcr.io/v2/homebrew/core/icu4c/blobs/sha256:87617a04333c
Already downloaded: /Users/aayush/Library/Caches/Homebrew/downloads/df561d143b154c58e1fbe7984759adc936171665c0ca93ad3e6a53723cf7ca0--icu4c--71.1.monterey.bottle.tar.gz
==> Downloading https://ghcr.io/v2/homebrew/core/libnghttp2/manifests/1.50.0
#####
 100.0%
==> Downloading https://ghcr.io/v2/homebrew/core/libnghttp2/blobs/sha256:76848a3
```

3. Checking *mongodb* version and connecting to the database



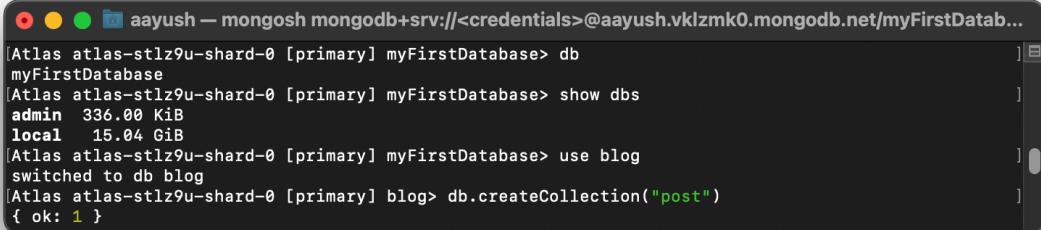
```
[Aayushs-MBP: ~ $ mongosh --version
1.6.0
[Aayushs-MBP: ~ $ mongosh "mongodb+srv://aayush.vklzmk0.mongodb.net/myFirstDatabase" --apiVersion 1
--username aayush
[Enter password:
```

4. Various basic commands execution

4.1. *db* : check current database

4.2. *Show dbs* : check available database

4.3. *Use <name>* : create new database named as <name>



```
[Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> db
myFirstDatabase
[Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> show dbs
admin 336.00 KiB
local 15.04 GiB
[Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> use blog
switched to db blog
[Atlas atlas-stlz9u-shard-0 [primary] blog> db.createCollection("post")
{ ok: 1 }
```

5. *insertOne* command for inserting a document

```
[Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> show dbs
[blog      8.00 KiB
[admin    336.00 KiB
[local    15.04 GiB
[Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> use blog
switched to db blog
[Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.insertOne({
...   title:"Post intro to MongoDB",
...   body:"Thank you Aayush for using me",
...   category:"News",
...   likes:4,
...   tags:["news", "events"],
...   date:Date()
... })
{
  acknowledged: true,
  insertedId: ObjectId("636ded782a1aa668f6a52ac8")
}
```

6. *insertMany* command for inserting more than one documents

```
Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.insertMany([
...   {
...     title: "Post Title 2",
...     body: "Body of post.",
...     category: "Event",
...     likes: 2,
...     tags: ["news", "events"],
...     date: Date()
...   },
...   {
...     title: "Post Title 3",
...     body: "Body of post.",
...     category: "Technology",
...     likes: 3,
...     tags: ["news", "events"],
...     date: Date()
...   },
...   {
...     title: "Post Title 4",
...     body: "Body of post.",
...     category: "Event",
...     likes: 4,
...     tags: ["news", "events"],
...     date: Date()
...   }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("636dee042a1aa668f6a52ac9"),
    '1': ObjectId("636dee042a1aa668f6a52aca"),
    '2': ObjectId("636dee042a1aa668f6a52acb")
  }
}
```

7. Different usecase of *find* to display documents

7.1. *find()* : display whole collection

7.2. *find({},{_id:0,name:1})* : Only show name field without showing ids

```
aayush — mongosh mongodb+srv://<credentials>@aayush.vklzmk0.mongodb.net/myFirstDatabase... [Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.find()
[
  {
    _id: ObjectId("636ded782a1aa668f6a52ac8"),
    title: 'Post intro to MongoDB',
    body: 'Thank you Aayush for using me',
    category: 'News',
    likes: 4,
    tags: [ 'news', 'events' ],
    date: 'Fri Nov 11 2022 12:06:40 GMT+0530 (India Standard Time)'
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52ac9"),
    title: 'Post Title 2',
    body: 'Body of post.',
    category: 'Event',
    likes: 2,
    tags: [ 'news', 'events' ],
    date: 'Fri Nov 11 2022 12:09:00 GMT+0530 (India Standard Time)'
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52aca"),
    title: 'Post Title 3',
    body: 'Body of post.',
    category: 'Technology',
    likes: 3,
    tags: [ 'news', 'events' ],
    date: 'Fri Nov 11 2022 12:09:00 GMT+0530 (India Standard Time)'
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52acb"),
    title: 'Post Title 4',
    body: 'Body of post.',
    category: 'Event',
    likes: 4,
    tags: [ 'news', 'events' ],
    date: 'Fri Nov 11 2022 12:09:00 GMT+0530 (India Standard Time)'
  }
]
[Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.find({}, {title:1})
[
  {
    _id: ObjectId("636ded782a1aa668f6a52ac8"),
    title: 'Post intro to MongoDB'
  },
  { _id: ObjectId("636dee042a1aa668f6a52ac9"), title: 'Post Title 2' },
  { _id: ObjectId("636dee042a1aa668f6a52aca"), title: 'Post Title 3' },
  { _id: ObjectId("636dee042a1aa668f6a52acb"), title: 'Post Title 4' }
]
[Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.find({}, { _id:0, title:1 })
[
  { title: 'Post intro to MongoDB' },
  { title: 'Post Title 2' },
  { title: 'Post Title 3' },
  { title: 'Post Title 4' }
]
[Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.find({}, { _id:0, title:2 })
[
  { title: 'Post intro to MongoDB' },
  { title: 'Post Title 2' },
]
```

8. Query for finding documents which contains intro in title field and use of *findOne* for only giving one output

```
aayush — mongosh mongodb+srv://<credentials>@aayush.vklzmk0.mongodb.net/myFirstDatabase... [Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.find({title:{$regex:"intro"}}, {body:1})
[
  {
    _id: ObjectId("636ded782a1aa668f6a52ac8"),
    body: 'Thank you Aayush for using me'
  }
]
[Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.findOne()
{
  _id: ObjectId("636ded782a1aa668f6a52ac8"),
  title: 'Post intro to MongoDB',
  body: 'Thank you Aayush for using me',
  category: 'News',
  likes: 4,
  tags: [ 'news', 'events' ],
  date: 'Fri Nov 11 2022 12:06:40 GMT+0530 (India Standard Time)'
}]
```

9. Different uses of *findOne*

```

[Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.findOne({category:"News"})
{
  _id: ObjectId("636ded782a1aa668f6a52ac8"),
  title: 'Post intro to MongoDB',
  body: 'Thank you Aayush for using me',
  category: 'News',
  likes: 4,
  tags: [ 'news', 'events' ],
  date: 'Fri Nov 11 2022 12:06:40 GMT+0530 (India Standard Time)'
}
Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.find({}, {_id: 0, title: 1, date: 1})
[
  {
    title: 'Post intro to MongoDB',
    date: 'Fri Nov 11 2022 12:06:40 GMT+0530 (India Standard Time)'
  },
  {
    title: 'Post Title 2',
    date: 'Fri Nov 11 2022 12:09:00 GMT+0530 (India Standard Time)'
  },
  {
    title: 'Post Title 3',
    date: 'Fri Nov 11 2022 12:09:00 GMT+0530 (India Standard Time)'
  },
  {
    title: 'Post Title 4',
    date: 'Fri Nov 11 2022 12:09:00 GMT+0530 (India Standard Time)'
  }
]
Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.find({}, {category: 0})
[
  {
    _id: ObjectId("636ded782a1aa668f6a52ac8"),
    title: 'Post intro to MongoDB',
    body: 'Thank you Aayush for using me',
    likes: 4,
    tags: [ 'news', 'events' ],
    date: 'Fri Nov 11 2022 12:06:40 GMT+0530 (India Standard Time)'
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52ac9"),
    title: 'Post Title 2',
    body: 'Body of post.',
    likes: 2,
    tags: [ 'news', 'events' ],
    date: 'Fri Nov 11 2022 12:09:00 GMT+0530 (India Standard Time)'
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52aca"),
    title: 'Post Title 3',
    body: 'Body of post.',
    likes: 3,
    tags: [ 'news', 'events' ],
    date: 'Fri Nov 11 2022 12:09:00 GMT+0530 (India Standard Time)'
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52acb"),
    title: 'Post Title 4',
    body: 'Body of post.',
    likes: 4,
  }
]

```

10. Update document

- To update an existing document we can use the *updateOne()* or *updateMany()* methods.
- The first parameter is a query object to define which document or documents should be updated.
- The second parameter is an object defining the updated

Here we'll perform update operation to update the like count of specific title

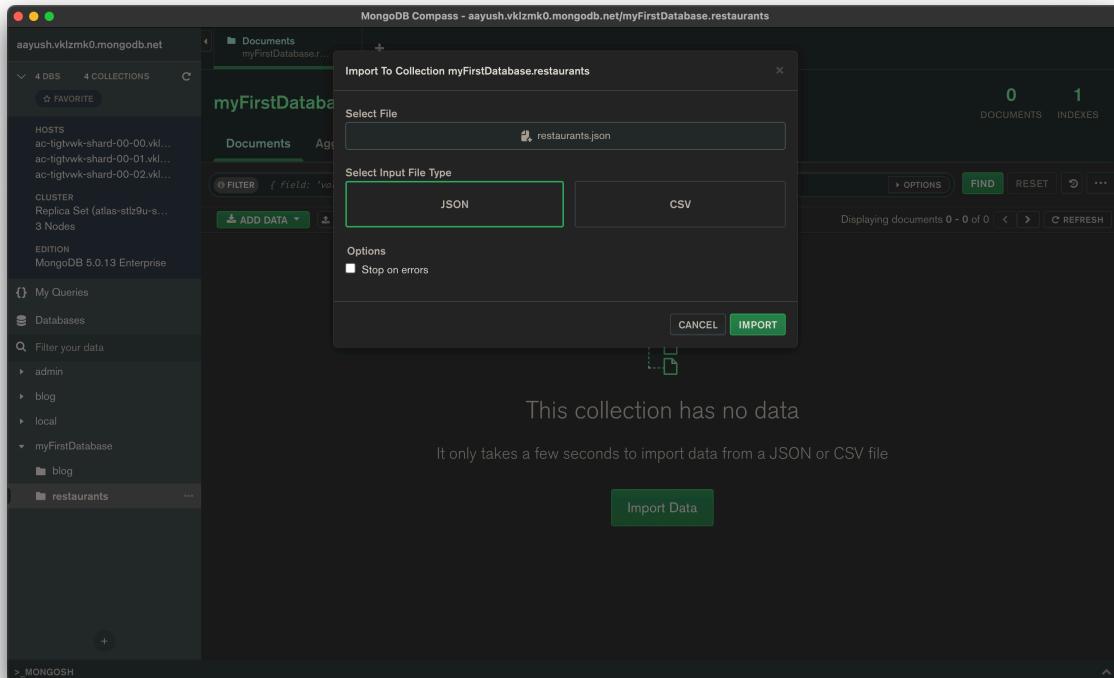
10.1.First query displays the whole table

10.2.Second query is for updating data

10.3.Third query is for displaying updated data

```
aayush — mongosh mongodb+srv://<credentials>@aayush.vklzmk0.mongodb.net/myFirstDatabase?retryWrites=true&w=majority
matchedCount: 1,
modifiedCount: 1,
upsertedCount: 0
}
[Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.find({}, {title:1, likes:1})
[
  {
    _id: ObjectId("636ded782a1aa668f6a52ac8"),
    title: 'Post intro to MongoDB',
    likes: 4
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52ac9"),
    title: 'Post Title 2',
    likes: 2
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52aca"),
    title: 'Post Title 3',
    likes: 3
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52acb"),
    title: 'Post Title 4',
    likes: 4
  }
]
Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.updateOne({title:"Post Title 2"}, {$set:{likes:10}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
[Atlas atlas-stlz9u-shard-0 [primary] blog> db.posts.find({}, {title:1, likes:1})
[
  {
    _id: ObjectId("636ded782a1aa668f6a52ac8"),
    title: 'Post intro to MongoDB',
    likes: 4
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52ac9"),
    title: 'Post Title 2',
    likes: 10
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52aca"),
    title: 'Post Title 3',
    likes: 3
  },
  {
    _id: ObjectId("636dee042a1aa668f6a52acb"),
    title: 'Post Title 4',
    likes: 4
  }
]
Atlas atlas-stlz9u-shard-0 [primary] blog>
```

11. Importing json data file (restaurants.json) and display documents in collection
db.restaurants.find()



```
[ 3 0 0 ] yashay - mongosh mongoDB+srv://<credentials>@yayush.vklzmk0.mongodb.net/myFirstDatabase?ssl=true&replicaSet=atlas-1t19u9-shard-0[primary]> show dbs
atlas 0.000 KB
myFirstDatabase 40.00 KB
System 30.00 KB
local 15.84 GB
[ 3 0 0 ] yashay - mongosh mongoDB+srv://<credentials>@yayush.vklzmk0.mongodb.net/myFirstDatabase?ssl=true&replicaSet=atlas-1t19u9-shard-0[primary]> db
[ 3 0 0 ] yashay - mongosh mongoDB+srv://<credentials>@yayush.vklzmk0.mongodb.net/myFirstDatabase?ssl=true&replicaSet=atlas-1t19u9-shard-0[primary]> db.restaurants.find()
{
  _id: ObjectId("63707fd3801aa981c384ab98f"),
  address: {
    building: "1007",
    coord: [-73.856877, 40.848447],
    street: "Morris Park Ave",
    zipcode: "10462"
  },
 borough: "Bronx",
  cuisine: "Bakery",
  grades: [
    {
      date: ISODate("2014-03-03T00:00:00.000Z"),
      grade: "A",
      score: 2
    },
    {
      date: ISODate("2013-09-11T00:00:00.000Z"),
      grade: "A",
      score: 6
    },
    {
      date: ISODate("2013-01-24T00:00:00.000Z"),
      grade: "A",
      score: 10
    },
    {
      date: ISODate("2011-11-23T00:00:00.000Z"),
      grade: "A",
      score: 9
    },
    {
      date: ISODate("2011-03-10T00:00:00.000Z"),
      grade: "B",
      score: 14
    }
  ],
  name: "Morris Park Bake Shop",
  restaurant_id: "6067545"
}
{
  _id: ObjectId("63707fd3801aa981c384ab98e"),
  address: {
    building: "469",
    coord: [-73.961784, 40.662942],
    street: "Flatbush Avenue",
    zipcode: "11220"
  },
  ...
```

```
ayash@ayash-mongosh:~/ccredentials>@ayash.vklzmk0.mongodb.net/myFirstDatabase>
...adoop(3.4.1/libexec/          -- bash           -- bash           -- bash ...
    },
    {
        date: ISODate("2013-05-28T00:00:00.000Z"),
        grade: 'A',
        score: 9
    },
    {
        date: ISODate("2012-12-07T00:00:00.000Z"),
        grade: 'A',
        score: 83
    },
    {
        date: ISODate("2012-03-29T00:00:00.000Z"),
        grade: 'A',
        score: 11
    }
}
},
name: 'Glorious Food',
restaurant_id: '#48361521'
},
{
_id: ObjectId('63707fd8801a8981c384a82*'),
address: {
    building: '208A',
    coord: [-73.9829239, 40.6588753],
    street: 'Prospect Park West',
    zipcode: '11215'
},
borough: 'Brooklyn',
cuisine: 'American',
grades: [
    {
        date: ISODate("2014-11-19T00:00:00.000Z"),
        grade: 'A',
        score: 11
    },
    {
        date: ISODate("2013-11-14T00:00:00.000Z"),
        grade: 'A',
        score: 12
    },
    {
        date: ISODate("2012-12-05T00:00:00.000Z"),
        grade: 'A',
        score: 13
    },
    {
        date: ISODate("2012-05-17T00:00:00.000Z"),
        grade: 'A',
        score: 11
    }
],
name: 'The Movable Feast',
restaurant_id: '#48361606'
}
]
Type 'i*' for more
```

12. Query to display the first 5 restaurant which is in the borough Bronx

```
db.restaurants.find({ "borough": "Bronx" }).limit(5);
```

```
[Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> db.restaurants.find({ "borough": "Bronx" }).limit(5)
[{"_id": ObjectId("63707fd3801a8981c384ab8f"),
  "address": {"building": "1007", "coord": [-73.856877, 40.848447], "street": "Morris Park Ave", "zipcode": "10462"}, "borough": "Bronx", "cuisine": "Bakery", "grades": [{"date": ISODate("2014-03-03T00:00:00.000Z"), "grade": "A", "score": 2}, {"date": ISODate("2013-09-11T00:00:00.000Z"), "grade": "A", "score": 6}, {"date": ISODate("2013-01-24T00:00:00.000Z"), "grade": "A", "score": 10}, {"date": ISODate("2011-11-23T00:00:00.000Z"), "grade": "A", "score": 9}, {"date": ISODate("2011-03-10T00:00:00.000Z"), "grade": "B", "score": 14}], "name": "Morris Park Bake Shop", "restaurant_id": "30075445"}, {"_id": ObjectId("63707fd3801a8981c384ab99"),
  "address": {"building": "2300", "coord": [-73.8786113, 40.8502883], "street": "Southern Boulevard", "zipcode": "10460"}, "borough": "Bronx", "cuisine": "American", "grades": [{"date": ISODate("2014-05-28T00:00:00.000Z"), "grade": "A", "score": 11}]}]
```

13. Query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx.

```
db.restaurants.find({ "borough": "Bronx" }).skip(5).limit(5)
```

```
[Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> db.restaurants.find({ "borough": "Bronx" }).skip(5).limit(5)
[{"_id": ObjectId("63707fd3801a8981c384abcc"),
  "address": {"building": "658", "coord": [-73.81363999999999, 40.82941100000001], "street": "Clarence Ave", "zipcode": "10465"}, "borough": "Bronx", "cuisine": "American", "grades": [{}]}]
```

14. Query to find the restaurants who achieved a score more than 90.

```
db.restaurants.find({grades : { $elemMatch:{ "score":{$gt : 90}}}});
```

```
Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> db.restaurants.find({grades : { $elemMatch:{ "score":{$gt : 90}}}})
[
  {
    _id: ObjectId("63707fd3801a8981c384aced"),
    address: {
      building: '65',
      coord: [ -73.9782725, 40.7624022 ],
      street: 'West 54 Street',
      zipcode: '10019'
    },
    borough: 'Manhattan',
    cuisine: 'American',
    grades: [
      {
        date: ISODate("2014-08-22T00:00:00.000Z"),
        grade: 'A',
        score: 11
      },
      {
        date: ISODate("2014-03-28T00:00:00.000Z"),
        grade: 'C',
        score: 131
      },
      {
        date: ISODate("2013-09-25T00:00:00.000Z"),
        grade: 'A',
        score: 11
      },
      {
        date: ISODate("2013-04-08T00:00:00.000Z"),
        grade: 'B',
        score: 25
      },
      {
        date: ISODate("2012-10-15T00:00:00.000Z"),
        grade: 'A',
        score: 11
      },
      {
        date: ISODate("2011-10-19T00:00:00.000Z"),
        grade: 'A',
        score: 13
      }
    ],
    name: "Murals On 54/Randolph's S",
    restaurant_id: '40372466'
  },
  {
    _id: ObjectId("63707fd3801a8981c384ad8e"),
    address: {
      building: '345',
      coord: [ -73.9864626, 40.7266739 ],
      street: 'East 6 Street',
      zipcode: '10003'
    },
    borough: 'Manhattan',
    cuisine: 'Indian',
    grades: [
      {
        date: ISODate("2014-09-15T00:00:00.000Z"),
        grade: 'A',
        score: 5
      },
      {
        date: ISODate("2014-01-14T00:00:00.000Z"),
        grade: 'A',
        score: 8
      }
    ]
  }
]
```

15. Query to find the restaurants that achieved a score, more than 80 but less than 100.

```
Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> db.restaurants.find({grades : { $elemMatch:{ "score":{$gt : 80, $lt : 100}}}})
[{"_id": ObjectId("63707fd3801a8981c384ad8e"), "address": {"building": "345", "coord": [-73.9864626, 40.7266739], "street": "East 6 Street", "zipcode": "10003"}, "borough": "Manhattan", "cuisine": "Indian", "grades": [{"date": ISODate("2014-09-15T00:00:00.000Z"), "grade": "A", "score": 5}, {"date": ISODate("2014-01-14T00:00:00.000Z"), "grade": "A", "score": 8}]}]
```

16. Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns



```

Atlas atlas-stlz9u-shard-0 [primary] myFirstDatabase> db.restaurants.find().sort(
...                               {"name": -1}
[...
[
  {
    _id: ObjectId("63707fd3801a8981c384ac4e"),
    address: {
      building: '6946',
      coord: [ -73.8811834, 40.7017759 ],
      street: 'Myrtle Avenue',
      zipcode: '11365'
    },
    borough: 'Queens',
    cuisine: 'German',
    grades: [
      {
        date: ISODate("2014-09-24T00:00:00.000Z"),
        grade: 'A',
        score: 11
      },
      {
        date: ISODate("2014-04-17T00:00:00.000Z"),
        grade: 'A',
        score: 7
      },
      {
        date: ISODate("2013-03-12T00:00:00.000Z"),
        grade: 'A',
        score: 13
      },
      {
        date: ISODate("2012-10-02T00:00:00.000Z"),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate("2012-05-09T00:00:00.000Z"),
        grade: 'A',
        score: 13
      },
      {
        date: ISODate("2011-12-28T00:00:00.000Z"),
        grade: 'B',
        score: 24
      }
    ],
    name: 'Zum Stammtisch',
    restaurant_id: '40367377'
  },
  {
    _id: ObjectId("63707fd8801a8981c384b788"),
    address: {
      building: '107109',
      coord: [ -73.9744668, 40.731155 ],
      street: 'Avenue C',
      zipcode: '10009'
    },
    ...
  }
]
]

```

Conclusion

From this practical, I installed mongoDB shell **mongosh** and used mongosh shell online as well as in mongodb compass. I also executed different queries for different crud operations. Moreover, I learnt how to import json file manually through mongodb compass.