1. **Instillations**
    - https://www.mongodb.com/download-center/community
2. Download and Install MongoDB community server
    - Create a separate installation location/directory "mongodb" (for windows, c:\mongodb) and install your MongoDB in that location instead of default location. (this will be helpful later when you are starting the service)
    - Start custom installation option
    - Uncheck "Install MongoD as a Service" option and hit next.
3. Create 3 folders inside mongodb after installations
    - Create a folder "data" c:\mongodb\data
        - Create a folder "db" inside data folder created above. c:\mongodb\data\db
    - Create a folder "log" c:\mongodb\log
4. Use Command Interpreter (cmd for windows, open with admin privilege)
    - Change the path from command line to where you have your mongoDB\bin folder
    - Now type
        - mongod –directoryperdb –dbpath c:\mongodb\data\db –logpath c:\mongodb\log\mongo.log –logappend –install
5. Start the MongoDB service.
    - Type of the followings
        - net start MongoDB

---

1. Start MongoDB
    - type mongo to start mongo shell
    - Cls to clear the screen

2. To show the databases
    - show dbs
    - use <database name> will use and switch to that database. If there's no database, this command will create one.
    - db will tell you current db
    - **[Exercise]** Create a database "Company"
    - **[Exercise]** Create a database "University"

3. [Exercise] To drop a database,
    - Use db to find the current database

- db.dropDatabase();
- **[Exercise]** Drop "University"

4. **[Exercise]** Create user for the database “Company”

```
db.createUser(
  {
   user: "John",
   pwd: "1234",
   roles: [ "readWrite", "dbAdmin" ]
  }
)
```

- MongoDB stores BSON documents, i.e. data records, in collections; the collections in databases. BSON is a binary representation of JSON documents
- Database is a physical container for collections.
- Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

5. **[Exercise]** Create a collection “customers” for Company database
   - db.createCollection(‘customers’);
   - show collections

6. Insert documents to collection
   - db.customers.insert({first_name:"Jon", last_name:"Doe"});
   - **[Exercise]** Create 5 customers and the fields for their first_name and last_name:
   - John Smith, Alicia Zelaya, Jennifer Wallace, Ahmad Jabbar, James Borg

7. find data in the customers collection
   - db.customers.find();
   - db.customers.find().pretty();
   - **[Exercise]** Find the data for document where the first_name is Jennifer
     - https://docs.mongodb.com/manual/reference/operator/query-comparison/
     - db.customers.find({first_name:{$eq:"Ahmad"}})
     - Use regex to find partial match db.customers.find({first_name: /Ah/})
   - Projection to whitelist fields to pass into output
     - db.customers.find({}, {first_name: true})

8. Multiple documents at once using array format
   - db.customers.insert([{first_name:"Sam", last_name:"Smith"} , {first_name:"Jade", last_name:"Smith", gender:"female"}]);

9. **[Exercise]** use an array to insert following to a database "petshop" and collection "pets"

```
use petshop
db.pets.insert({name: "Mikey", species: "Gerbil"})
db.pets.insert({name: "Davey Bungooligan", species: "Piranha"})
db.pets.insert({name: "Suzy B", species: "Cat"})
db.pets.insert({name: "Mikey", species: "Hotdog"})
db.pets.insert({name: "Terrence", species: "Sausagedog"})
db.pets.insert({name: "Philomena Jones", species: "Cat"})
```

- Add another piranha called Pete, and a naked mole rat called Henry.
- Use find to list all the pets. Find the ID of Mikey the Gerbil.
- Use find to find all the gerbils.
- Find all the creatures named Mikey.
- Find all the creatures named Mikey who are gerbils.
- Find all the creatures with the string "dog" in their species

10. Update
    - db.customers.update({first_name:"Sam"}, {first_name:"Sam", last_name:"Smith", gender:"male"})
    - You need to repeat all the fields with their data. Otherwise document will replace by just the fields available in the update statement. Use the $set operator instead.

    - Use the set operator for that
        - db.customers.update({first_name:"Sam"}, {$set:{gender:"male"}});
        - **[Exercise]** Update all the customers to include gender and age fields.

    - Use inc operator to increment numerical values
        - db.customers.update({first_name:"Sam"}, {$set:{age:40}});
        - db.customers.update({first_name:"Sam"}, {$inc:{age:5}});

    - Use unset to remove a field
        - {$unset: {field1:"", …}}
        - db.customers.update({first_name:"Sam"}, {$unset:{age:""}});
    - Use the upsert to insert if the update fails because document is not there
        - db.customers.update({first_name:"May"}, {first_name:"May", last_name:"June"}, {upsert: true});

11. Remove
    - db.customers.remove({}) // remove all the documents
    - db.customers.remove({first_name:"Sam"}, {justone: true})
    - justone will delete only first document it finds, otherwise it will delete all

12. Import
    - Import json files to the database
    - Exit from the mongo: type "exit" and then type the following in the command line. Your path should still be mongodb\bin
        - **[Exercise]** First download the file from and save it somewhere https://www.cs.odu.edu/~sampath/courses/f19/cs620/files/data/stocks.json
        - mongoimport --db stocks --collection stocks --file stocks.json

Submit the screen capture of your exercises to Activity 12 at Piazza.