MongoDB Tips

Basics

MongoDB is a “NoSQL” database that works really well with collections of JSON documents. Version 4.2 has already been installed on your course VM. You can find the its complete documentation here, although for the purpose of this course we will work primarily with the MongoDB “shell” for querying.

By default, the MongoDB server is not running on your VM. To start/stop the server, use the following commands:

```
sudo service mongod start
sudo service mongod stop
```

There is an example MongoDB database dump for a database about the U.S. Congress. To create a MongoDB database named `congress` from this dump, run:

```
mongorestore --db congress /opt/dbcourse/examples/congress/mongodb-
dump/congress
```

Working with the MongoDB shell

While the server is running, you can start an interactive MongoDB shell using the command `mongo`. Once you are inside the MongoDB shell, there are a few essential commands:

- `show dbs` shows database names
- `use bar` sets the current database to `bar` (for example)
- `show collections` shows the collections in the current database
- `db.foo.find()` lists documents in the collection named `foo` in the current database
- `exit` exists from the MongoDB shell

You can type complex, multi-line `find` and `aggregate` queries inside the MongoDB shell and see the result of your query in a piecemeal fashion: when there are more result objects to return, you will be prompted to type “`it`” to see the the next batch of result objects.

You might find an alternative way of running the MongoDB shell (in an “immediate mode”) more convenient. Here, you’d write your query in a `.js` file (in plain text). For example, suppose the file `query.js` contains the following MongoDB query:

```
db.committees.aggregate([

        { $match: { _id: "SSJU" } },

        { $addFields: {

                subcommittee_names: { $map: {

                        input: "$subcommittees",

                        as: "sub",

                        in: "$$sub.displayname",

                } },

        } },

        { $project: {

                members: false,

                subcommittees: false,

        } },

]).toArray()
```

Note the `.toArray()` call at the end of the query, which causes the output to be printed in a nice format. To run this query, use the following command:

```
mongo --quiet congress < query.js
```

Here, `congress` is the name of the databases that we are querying; the `--quiet` flag tells MongoDB to just output query results. If the output is long, you can always "pipe" it to a "pager" program like `less` (see help with Linux Basics for details):

```
mongo --quiet congress < query.js | less
```

Recall that you can press "`q`" to exit from the paging mode.

You can also specify a short query (or command) directly on the command line, using the `--eval` flag. For example, the following command prints out all documents in the `people` collection of the `congress` database:

```
mongo --quiet congress --eval 'db.people.find().toArray()' | less
```

MongoDB query reference

The myriad of MongoDB query methods can be confusing to go through, but here are some especially helpful pointers:

- SQL to MongoDB Mapping Chart, which can be used as a quick reference for the `find()` function (as well as data modification methods).
- SQL to Aggregation Mapping Chart, which can be used as a quick reference for the `aggregate()` function.

Other useful MongoDB commands

```
# insert a new document into the products collections of toy database;

# if the database and/or collection did not exist before, they will be
created:

mongo --quiet toy --eval 'db.products.insert({_id: 10190, name: "Market
Street"})'

# update a document (to add a new field):

mongo --quiet toy --eval 'db.products.update({_id: 10190}, {price:
100.00})'

# list the documents in a collection:

mongo --quiet toy --eval 'db.products.find().toArray()'

# create a dump of the toy database in directory mongodb-dump/toy/:

mongodump --db toy --out mongodb-dump

# drop the toy database

mongo --quiet toy --eval 'db.dropDatabase()'

# restore the toy database from the dump

mongorestore --db toy mongodb-dump/toy
```

Working with MongoDB in Python

Under the directory `/opt/dbcourse/examples/pymongo/` on your VM, you will find an example program. See `pymongo` documentation for additional help.