**CS421/621 Lab05**

**MongoDB & Poker API**

**Objectives:**

* A quick overview of MongoDB
* Connecting to MongoDB command line client
* Poker API exercise

MongoDB (or Mongo, for short) is a database that allows us to store data on disk, but not in a relational format. Instead, Mongo is a document-oriented database that conceptually allows us to store objects organized as collections in the JSON or JavaScript Object Notation format. (Technically, MongoDB stores its data in the BSON format, but from our perspective we can think of it as JSON.) In addition, it allows us to interact with it completely in JavaScript smoothly. Mongo can be used for more complex data storage needs, such as storing user accounts or comments in blog posts. It can even be used to store binary data like images. Today we are going to show how to interact with the MongoDB command line client.

Like Redis, Mongo offers a command-line client that allows us to directly interact with the data store. You can fire up the Mongo client by typing "mongo" at the command line of your guest machine:

Firstly, navigate to Chapter7 folder and start Vagrant   
$ vagrant up

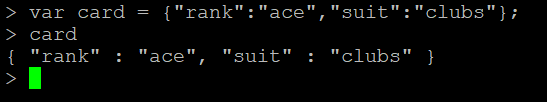
$vagrant ssh

Now you can type mongo and hit enter to get into mongodb.

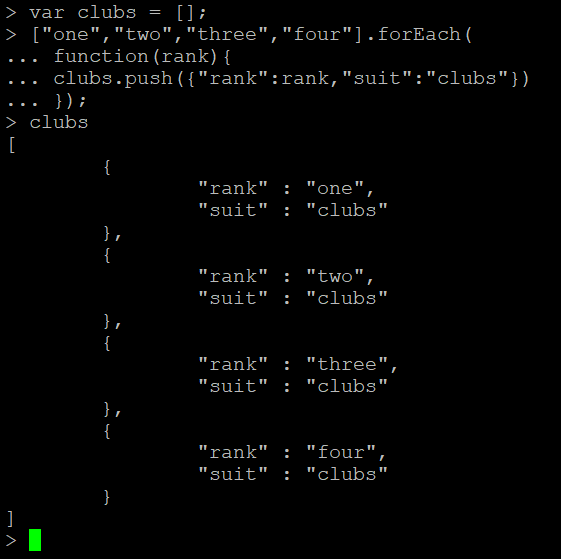


You may see some start-up warnings when Mongo first launches, but that’s totally normal.

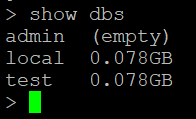
One immediate difference between Redis and Mongo is that we can interact with it using JavaScript. For example, here we create a variable called card and store an object in it:



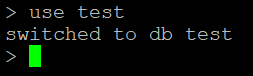
Likewise, we can create and manipulate arrays. Notice that in this example, we don’t complete our JavaScript statement before the end of each line. When we press Enter, Mongo responds with three dots letting us know that the previous statement was incomplete. Mongo will automatically execute the first full JavaScript statement:



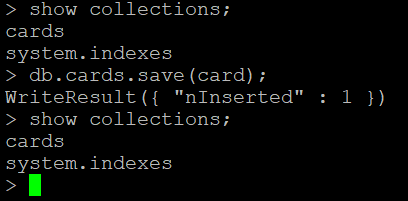
In other words, the Mongo command-line client works a little bit like the JavaScript console in Chrome. But those similarities end when we want to start storing data. Mongo organizes data as documents, which we can think of as JSON objects. It stores collections of documents in databases. We can see the databases in our MongoDB by using the show dbs command:



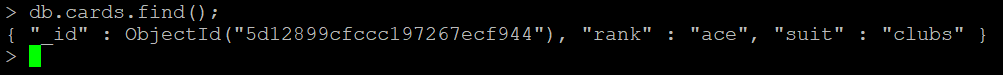
The local db is always there. We can switch to a different database with the use command:



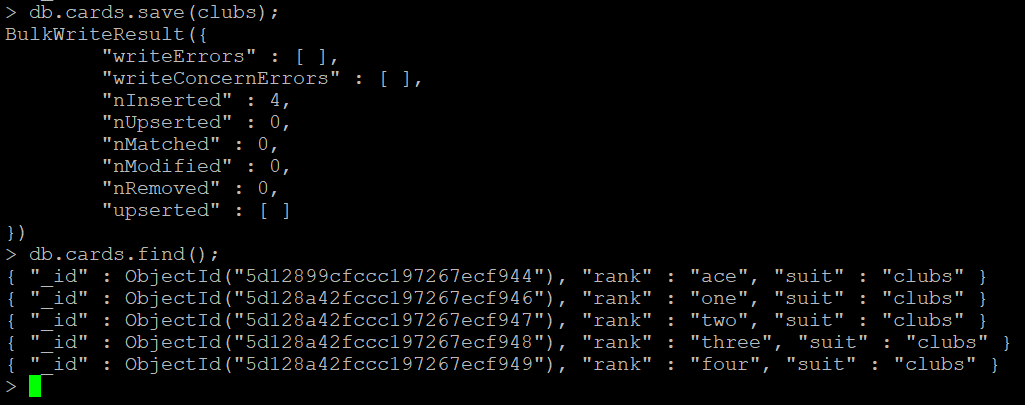
Once we’ve selected a database to use, we can access it through the db object. We can save objects to a collection by calling the save function on the collection. If the collection doesn’t already exist, Mongo will create it for us. Here, we save the card that we created earlier in our collection:



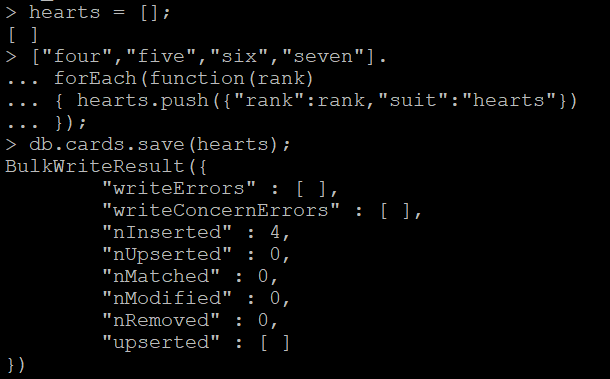
You’ll see that the cards collection doesn’t exist before we save an object to it. We can use the collection’s find function with no arguments to see what documents are stored:



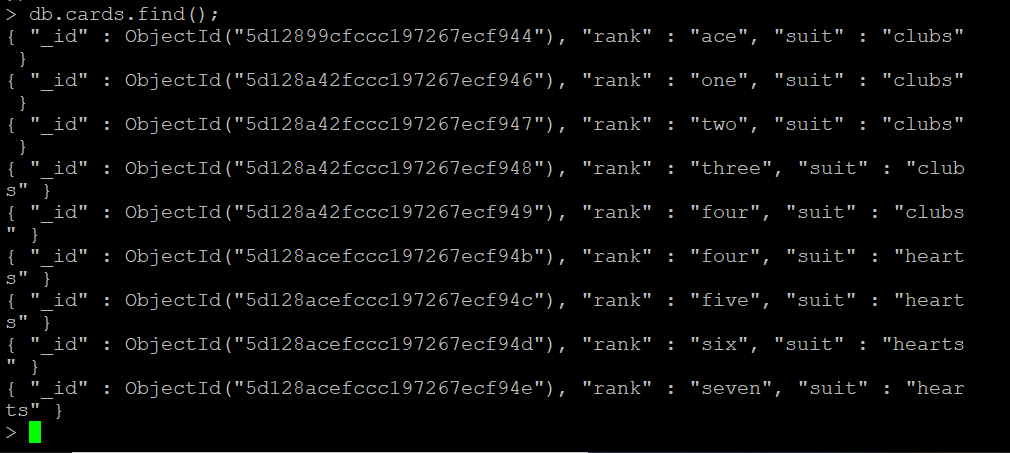
In addition to rank and suit properties, a card also has an \_id associated with it. For the most part, every document in a MongoDB collection has one of these associated with it. In addition to saving a single item, we can also add multiple documents to the collection in one call to save. Here we do that with the club’s array that we created previously:



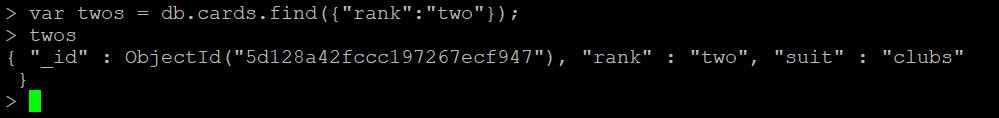
We can also add more objects to the db:



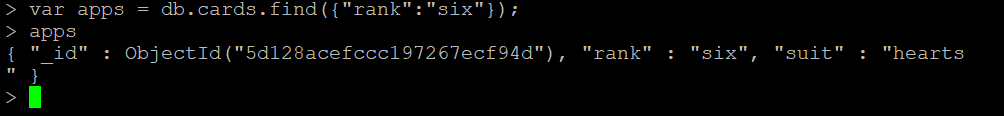
Now, lets find the entire collection of objects by doing db.cards.find();



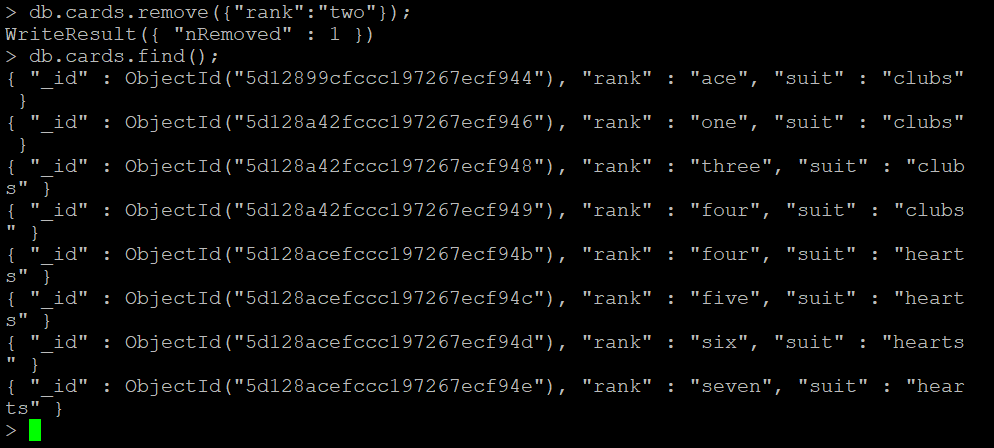
Once we have enough varying documents in our collection, we can retrieve them by creating queries from JSON objects that represent the properties of the documents we want to retrieve. For example, we can retrieve all card documents with a rank of two and store them in a variable called twos:



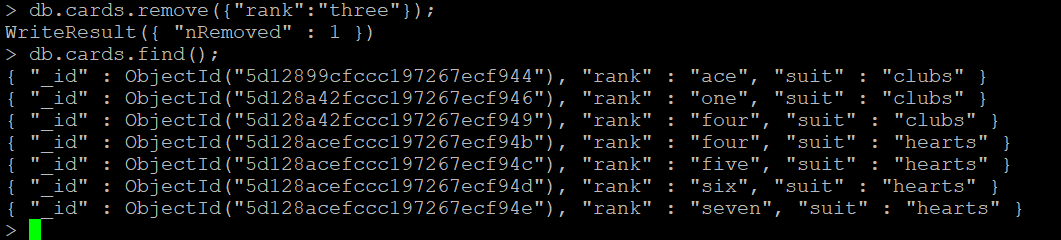
Let us try once using another object from that collection.



We can also remove the elements from the collection by calling the remove method and sending in a query:

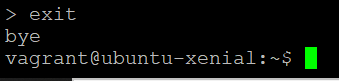


Let us try for another object.

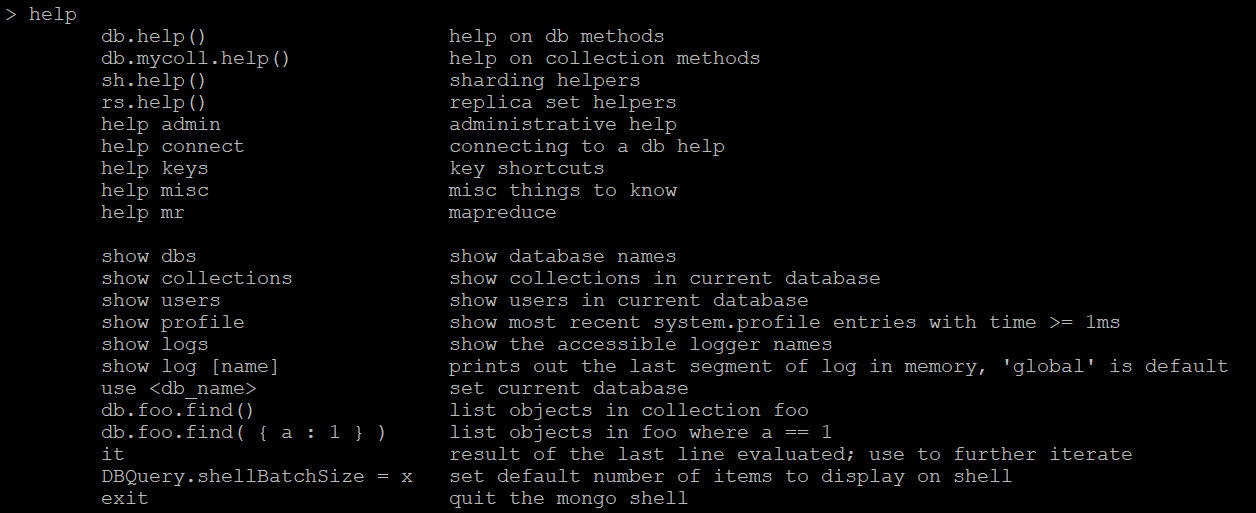


You must have noticed that it worked, similarly we need to remove for all the objects. Either it can be done separately or by removing all the documents from the collection.

Or we can remove all the documents from the collection by calling remove with an empty query. It goes like db.cards.remove(); Then, please do db.cards.find(); again to check, you should be seeing all the objects removed. You can quit from the MongoDB shell by typing exit. Once you do that, you should see something like this:



If you need help with any commands, please type help in the MongoDB shell. You should see something like this:



Similar to Redis, MongoDB offers an interactive MongoDB tutorial <https://university.mongodb.com/>) that you can try out in your web browser. I encourage you to work your way through this tutorial to learn a little more about Mongo’s functionality and types of queries that are available.