Project 10: MongoDB

Note: Using Ruby and the mongo gem is encouraged, but you may use a MongoDB library for the language of your choice, if you wish.

Objectives
Understand the document-oriented data model as implemented by MongoDB.
Using Ruby, learn how to write a simple program that communicates with a hosted MongoDB instance.

Overview
In this project, you will educate yourself about MongoDB, a popular, scalable, document-oriented database management system that uses a binary version of JSON to represent "documents." Using Ruby and the **mongo** gem, create a simple program that reads and writes data to/from a MongoDB instance.

Your program will connect to a remote MongoDB server hosted by MongoHQ.

Prerequisites
You should have a working Ruby environment (1.9.3 preferred).

Read the MongoDB introduction. Be sure to follow the links for *database*, *collection*, and *document* under "MongoDB Data Model."

Read the first half of the MongoDB Read Operations guide, to get a feel for how queries work. Also read the MongoDB Write Operations guide. Make note of the documentation for Create, Read, Update and Delete. You will refer to these while working on your program.

Lastly, read about how MongoDB models relationships as Database References.

Setup
For this project, you simply need to install the MongoDB gem and the BSON gem:

```
gem install mongo
gem install bson
```

Experiment
Visit http://www.mongodb.org/ and click the "Try it Out" menu item. Spend thirty minutes working through the MongoDB tutorial before proceeding.

Next, point your web browser to https://app.mongohq.com/ and sign in with the username junk@humanoriented.com and the "typical csci403 password" (what's up with nosql?).

Explore the MongoHQ interface and you will notice there is a database for you called “zoolicious.” Explore that database and its collections. What do you see?

Under the “Admin” section you will find the Mongo URI that you will need in order for your Ruby program to connect to the zoolicious database.

Create a Simple Program
Create a new text file called **zoolicious.rb** inside a new project10 directory. It should look like this:

```
require ‘mongo’
```

Execute the program (eg, `ruby redis.rb`) and you should see no errors. Do not proceed further if you witness an error -- ask on Piazza or in person about how to resolve the error. Be sure to post the contents of your .rb file and the exact error output.

Now you’re off to the races. Remember to take a look at the Admin section for the zoolicious database to obtain the MongoDB url. It looks something like this:

```
mongodb://<user>:<password>@staff.mongohq.com:10033/zoolicious
```

You should use `csci403` as the user and the usual “csci403 password” for the password.

Now exercise the mongo Ruby API. Your program should list the zoo names, habitat names and descriptions, and the animal names, descriptions and cuteness values.

In addition, you will find that some animals are associated with particular habitats. Your list of animals should also display the habitats associated with each animal, if such a relationship exists. (How might you use the “object id’s” to display the associated habitats?)

Lastly, your program should prompt the user for an animal name, description, and cuteness value (an integer), and store a new animal in the database.

Exploration
Remember, MongoDB does not support joins between collections. Instead it uses “Object IDs” to refer to other documents. What does this mean for an application that needs to display data that is associated with a particular document? How is this different from the relational query approach that uses joins? (Think: how many queries are involved here?)

Presentation

Good writing is important, **especially when it comes to code**. *Writing with poor readability, lack of appropriate whitespace, misspellings, etc will be handed back for re-writing (no grade penalty).*

Submit a zip of your code via blackboard.

Grading Criteria (90 points)
Complete implementation of a simple program that communicates with the remote MongoDB server using the mongo gem (or use the language and MongoDB library of your choice), and fulfills the requirements above. (90 total points).
- writing quality (30)
- implementation (60)

Please submit your work via Blackboard by 7AM on the due date.

Bonus / Extra Credit
Write your program using an object-to-document mapper such as MongoMapper, Mongoid or Mongomatic, such that your program contains no explicit database code itself. (Think about how your program might resemble the code in the ActiveRecord project assignment.)