MongoDB Lab

Be prepared to demonstrate your knowledge of all of the following concepts and operations.  Keep track of your scripts and any learning and notes in this document.

On the due date you will meet with the instructor to demonstrate your work and discuss what you have learned. You may use this document as reference during the pass off.

* Get MongoDB up and running on your machine. No clustering or sharding is required.
* You can download MongoDB to windows or set it up on a VM with CentOS
* What are the MongoDB equivalents of an:
* RDBMS database: collection
* RDBMS table: item
* RDBMS row: field
* What is the case sensitivity of MongoDB?
  1. Yes, it’s case sensitive
* Schema: Lets assume you are creating an Application to store a person’s Contacts (Name, Phone Numbers, Email Addresses, Nickname, etc.)
* Design two schemas to hold your contact information.
* Schema 1 should use a single collection to hold all information.
  + - 1. A Collection of people, Name, Phone, Email, Nickname, etc are all fields
* Schema 2 should use multiple collections to store your contact information.

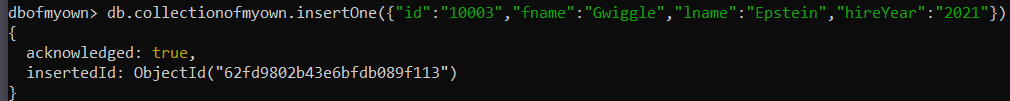
I have no clue how to make a schema with multiple collections that isn’t just splitting things up unnecessarily

Like, we could have a collection of phone numbers, and a collection of emails

Where both collections have an ID, and Name field so we can find the number and email of “Bruce Campbell“, but that’s an objectively bad design

* What are the pros and cons of each of the above solutions?

Schema 1 is faster to search through, but less secure.

* 1. Schema 2 is slower to search through, but more secure and its easier to find specific information you are looking for
* What are the costs and benefits of denormalization?
  1. Denormalization risks duplicate data which is inefficient (wastes space), but it also gives us the flexibility to have only the fields that are necessary, which is helpful in the exact same way duplicate data isn’t. (saves space)
* Import all 10,000 people records from Lab 1 into MongoDB.
* What is the significance of the \_id field on each document?
  1. It’s a unique id for every item, it’s generated with a timestamp (and more) so it should always be unique
* Demonstrate all four CRUD operations on a single document by key on the people collection in the Mongo console.
  1. C
  2. 
  3. R
  4. Text

     Description automatically generated
  5. U
  6. Text

     Description automatically generated
  7. D
  8. 
* Demonstrate the difference between replacing a document and adding a new property to a document in the Mongo console.
* Complete the following queries on your new data set:
* Return a single document by \_id. (In SQL: …Where PrimaryKey = x)

Text

Description automatically generated

* Return all the documents with the same value for a field. (Where SomeField = x)

Text

Description automatically generated

* Return all the documents who’s field value contains a string (similar to like ‘%test%’ in SQL)
  1. Text

     Description automatically generated
* Return all the documents \_having\_ a particular field. (Is there a SQL equivalent for this?)
  1. Text

     Description automatically generated
* List the documents missing a particular field.
  1. Text

     Description automatically generated
* Return the number of documents that have a particular value for a field. (Count(\*) …. Where Field = ‘test’)
  1. 
* Write an aggregate (sum, average, count etc.) grouped by a particular field. (Select state, sum(population) … Group by state)

states.aggregate([{‘group’:{‘\_id’:none,’pop’:{‘$sum’:population}}}])

(our data doesn’t have much to be aggregated)

* Be prepared to explain the MongoDB equivalent of
* Select \* From Customers c Join Orders r On c.CustId = r.CustId Join OrderDetails d on r.OrderId = d.OrderId
* Joins don’t exist, so we us $lookup with another lil’ json object. It returns us a lil’ guy of a collection :)

Take a slower query from above and add an index to improve its performance. Take a look at <http://docs.mongodb.org/manual/tutorial/analyze-query-plan/> to help you.

* Summarize the steps to create a sharded cluster in MongoDB. You don’t need to carry out the steps unless you want to… just be able to describe the major working components and the English-language steps to configure them.
  1. We gotta set up 3 different things, first we set up our shards, we want an odd amount.
  2. Then we set up 3 config servers, these guys know which shards have which information
  3. Then we set up a router to handle all the config server and shard communication for us!
* Compare and contrast the strengths and weaknesses of MongoDB vs
* Relational Databases

Mongo is like hella more flexible, and the loose schema is convenient and quick to work with. A drawback of the loose schema means that if we get sloppy it won’t stop us from making mistakes.

* Serialized Files
  1. Serialized files aren’t a database, we kinda just access them and there’s little to optimize, they are there we have to look at each individual file until we find what we want
* HashMap/HashTable
  1. Hash maps take more storage, but are hella fast. We know EXACTLY where that one thing is, if you look at our nifty little map
* Be prepared to describe what it might take to convert from a relational database system to a MongoDB database.
  1. We gotta normalize all our data, and then port it over, hopefully automatically? If not, that sucks
* Be prepared to discuss three examples of software systems for which Mongo would be a wise database choice.
  1. I am prepared
* Be prepared to discuss examples of software systems for which Mongo would be a poor database choice.

I am prepared;/.;./