Name: Joshua Steward Lab 7-7.5

Database Systems October 25, 2017

**Part I. Get Ready (10pts)**

1. In your own words, describe the NoSQL approach and how it differs from the traditional database approach. What are the benefits and caveats to each? And when would one approach be preferred over the other? To support your answer, include one relevant example for both the NoSQL database and an RDBMS.  
   \*Complete answers should be around 1/2 page.  
   \*\*Don’t just copy paste from the slides, Wikipedia or online sources. If you use external sources cite them properly.  
   \*\*\*Use one unique and relevant example to support your claims.  
     
     
   NoSQL denotes a wide range of less than relational databases. Standing for not only SQL, there are many databases systems that utilize this way of going about database design. Some of those include MongoDB, BigTable (used by google), DynamoDB, and many more. What you can imply right off the bat by the fact that a company like google uses a NoSQL style database is that NoSQL is much more scalable and supports massive storage of data much more efficiently. The situations that constitute using one or the other usually boil down to whether the data you need to use needs to be structured and organized into entities and relationships between those entities, or just holding large to massive amounts of data in file or document type storage.

Generally, NoSQL Databases don’t have to have a specified with a schema at the outset. Large databases systems that store massive amounts of data, such as facebook, google, or email structures, generally use some NoSQL alternative. This is due to their flexibility; take for example a social media site like facebook. If programmers had to redefine some structure for how they store user data and links every time they came up with new ideas to improve their system, they would be working with *millions* of data consistency issues. This has been one of the main pros towards moving towards NoSQL alternative; just working with large amounts of semi-structured data.

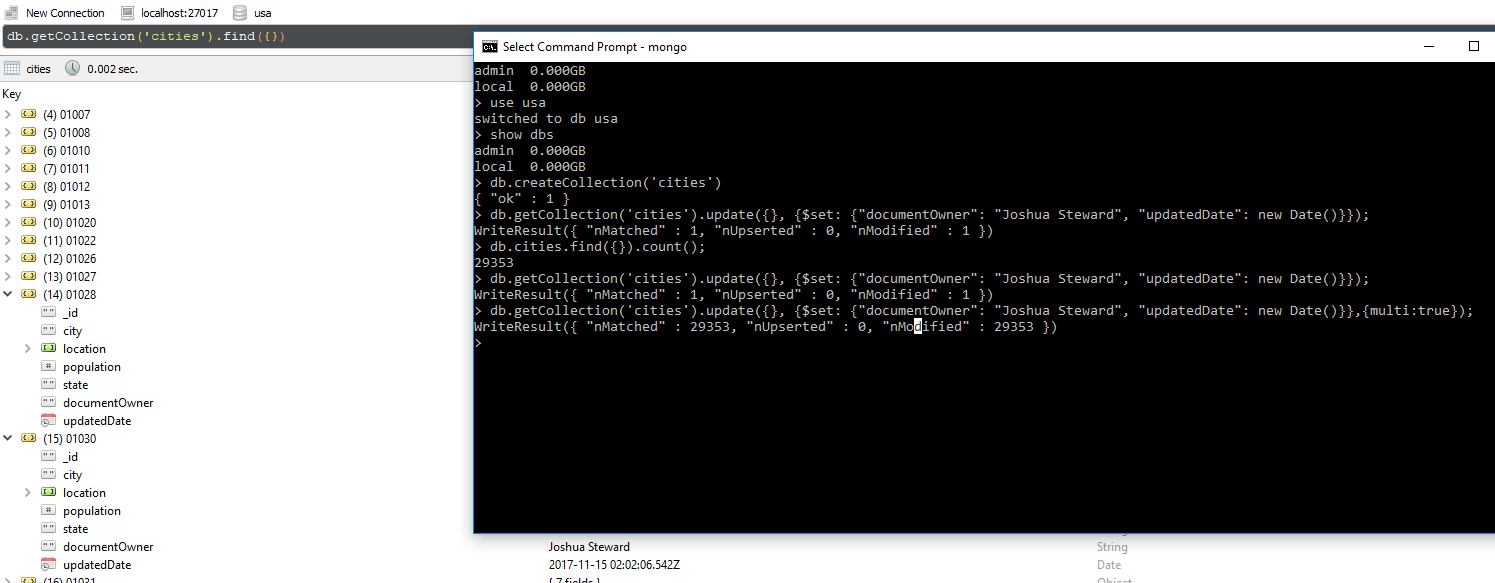
Usually, NoSQL databases are document oriented. This means that data can be stored in entity documents that can then be found, but probably isn’t heavily organized.

Some of the pros towards using NoSQL also involve their ease of use with external systems in processing data. Usually, using an external programming language like java to access data in an SQL database requires understanding how that underlying database is designed. NoSQL allows developers to easily parse through information and run systematic checks for data types. This does, however, decrease efficiency drastically compared to a more structured database.

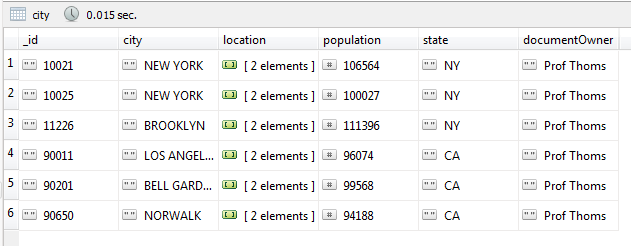
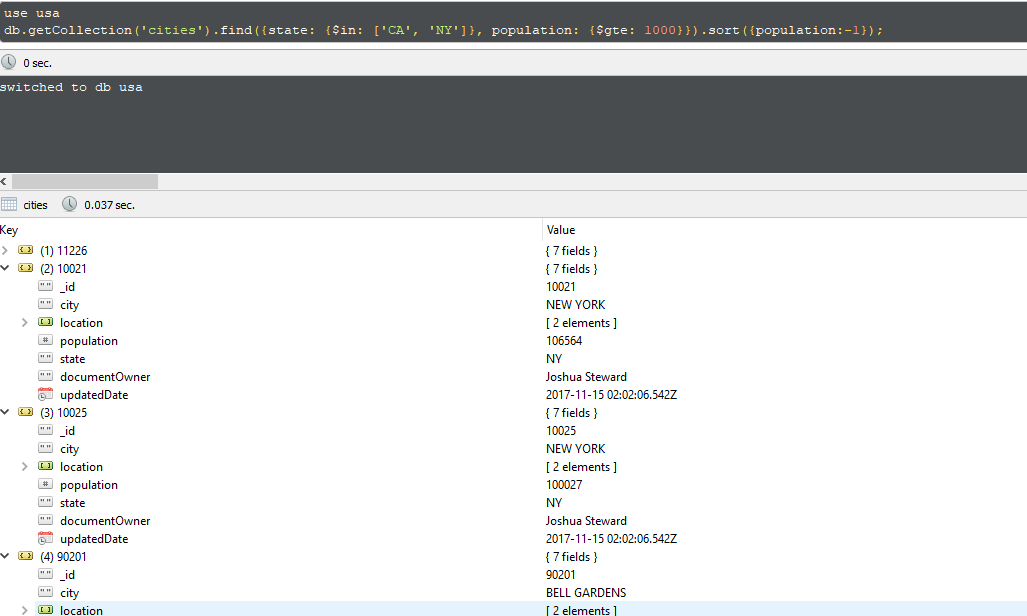
An example where NoSQL databases are preferred would be almost any social media site. Social media requires massive amounts of data, and the type of data being stored is often changing. The searches would be done using more traditional graph and edge type data structure manipulation, with actual user data being stored in object like documents. This allows for the data to be changed and added to when upgrades are available for the system. SQL is better used in situations where more complex queries are necessary, such as a trading card game where different entities have complex interactions with other entities based on types and such.

**Part II. Get Set (10pts)**

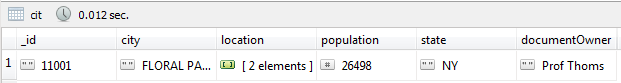
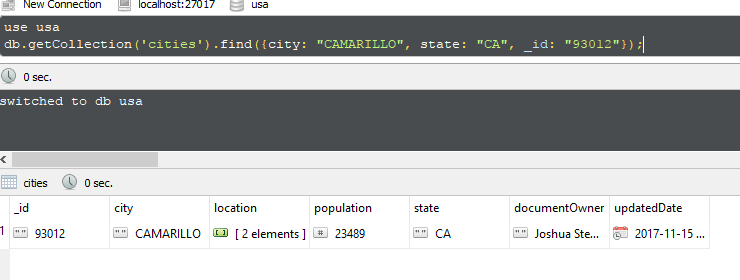
1. Create and run the query to import cities.json into a new database USA under the new collection, city. Update all documents to add the field documentOwner with the value *yourname.*

  
  
  
db.getCollection('cities').update({}, {$set: {"documentOwner": "Joshua Steward", "updatedDate": new Date()}}, {multi:true});

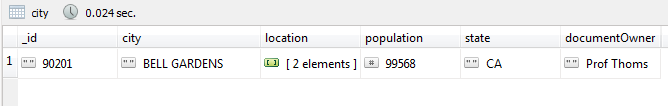
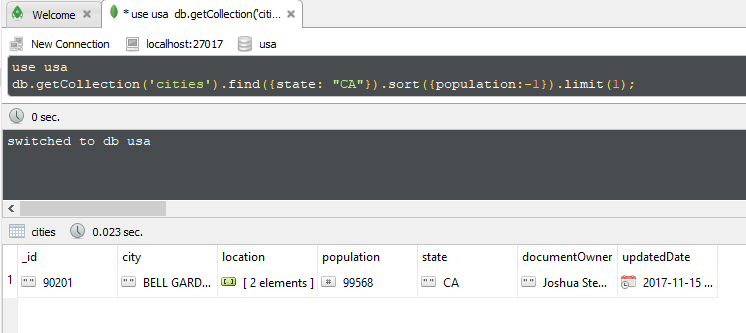
1. Create and run the query to return all documents with information related specifically to California, New York with populations greater than 1k.

1. Create and run the query to return the document for the city where you grew up.

1. Create and run the query to return the document with the largest population in California.

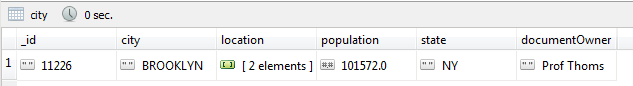
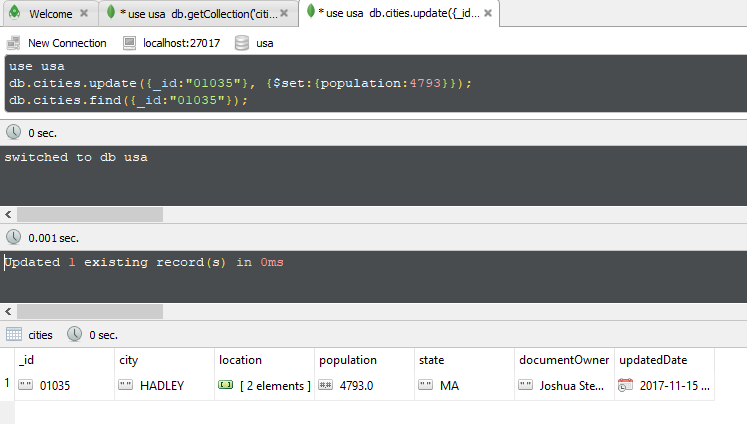
  
  


1. Use <https://zipwho.com> to find the correct population for one new city. Update the document for that city.

Before:

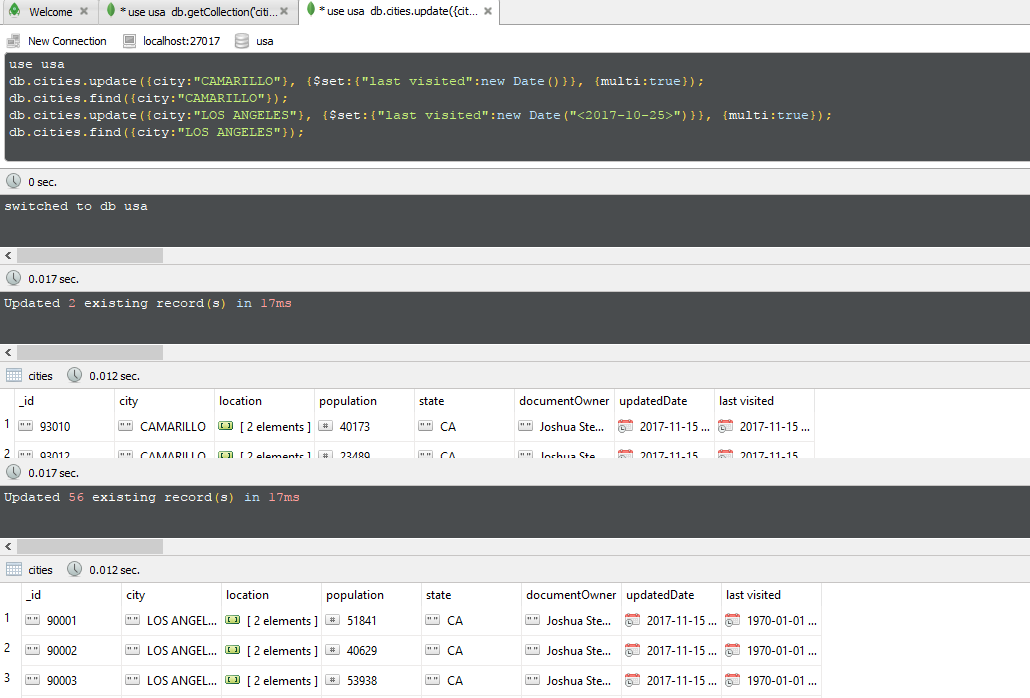
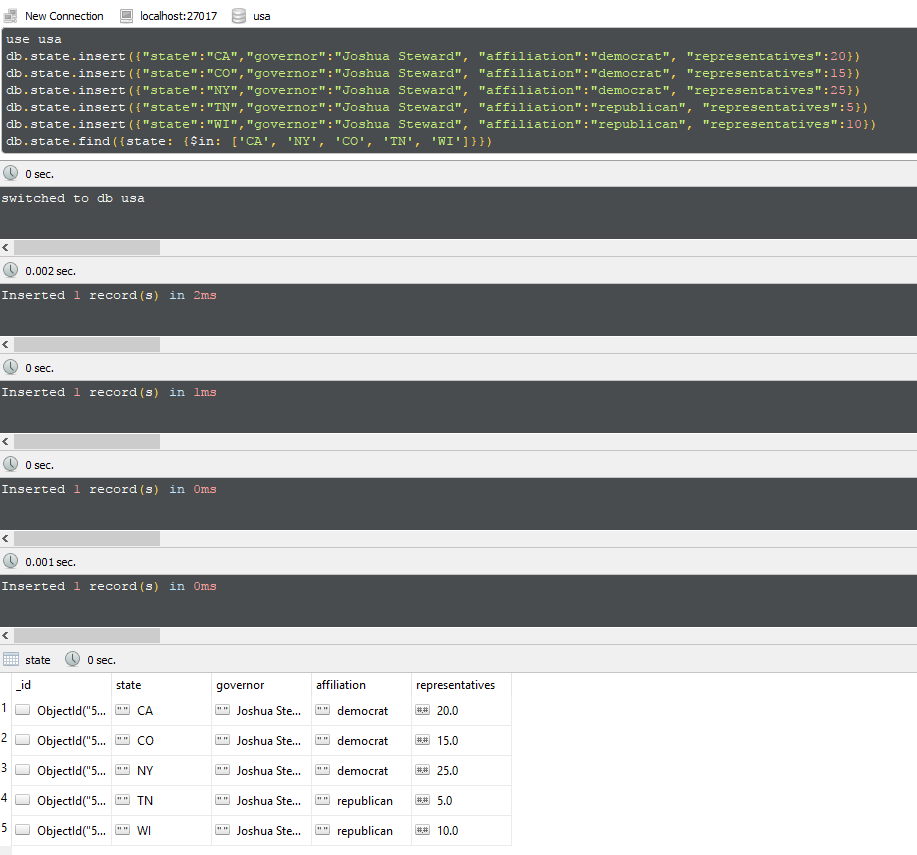
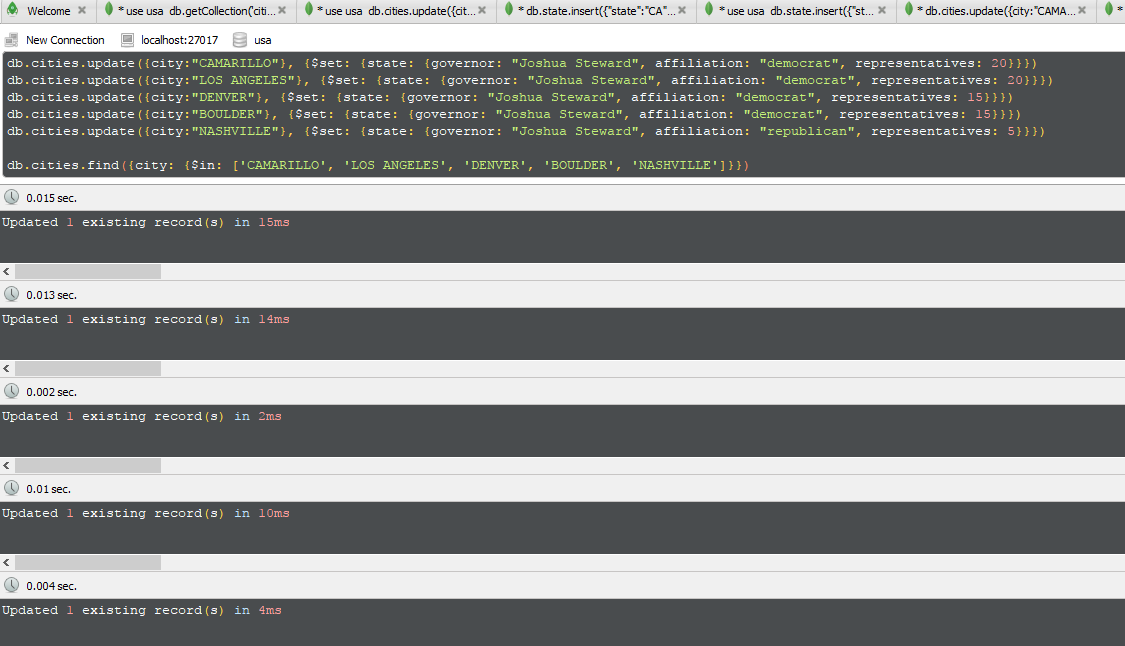
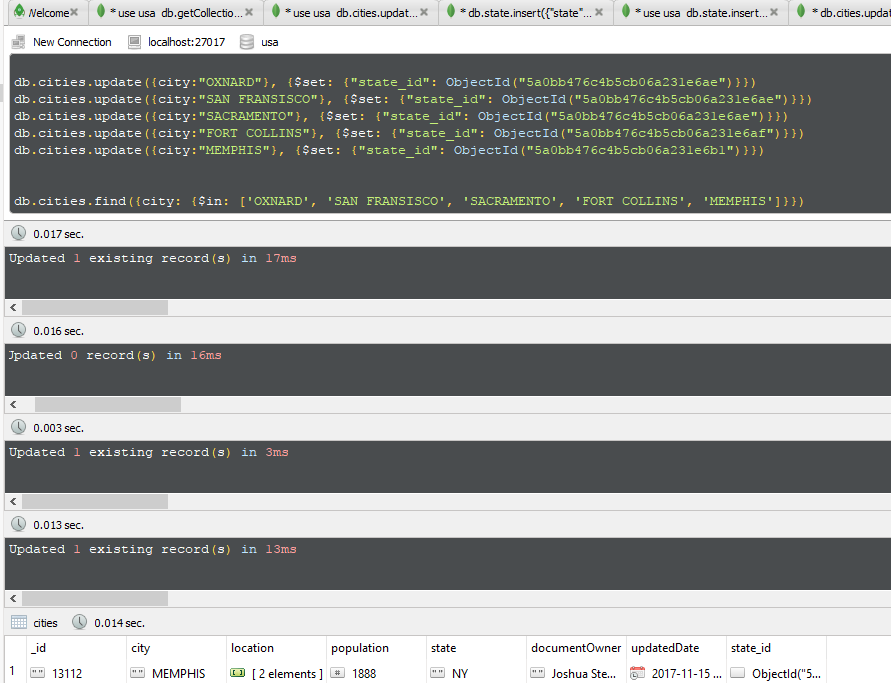


After:

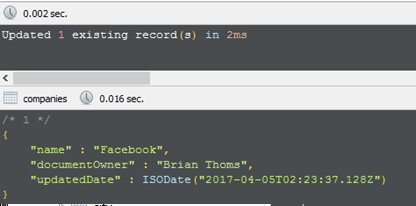
**Part III. Go (10pts)**

Within the USA database, create a new collection named, state, which will store data specific to states. This collection will be used to complete Q8 through Q10.

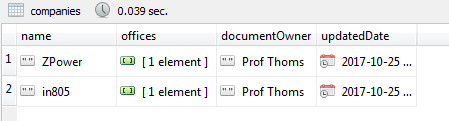
1. Write the insert statements to add documents for cities where you have lived, visited, or wish to visit in the future (up to 5). Include the dates you were there or when you hope to visit. If these documents already exist, update the documents to include the dates you last visited.  
     
   
2. Write the insert statements to add documents for 5 states. State data is up to you, but include at least three to five pieces of state-related data. Also include a field with your name as the governor.  
     
   
3. Write the update statements for 5 documents in the city collection based on the information found in the documents created in Q8 (**embed all state data**).   
     
     
   
4. Write the update statements for five more documents (not updated in Q9) in the city collection based on the information found in the documents created in Q8 (**reference state documents**).  
     
     
   
5. Create the Mongo query that will return state information based on Q10. What issues do you run into when querying MongoDB for referenced data? How can these issues be overcome? Provide 1 example.  
     
   One of the first issues I ran into was the fact that since there are multiples of the same cities, and unique cities are identified by their zip code in this database, it could get difficult to query the correct cities that I had updated.

**Part IV. Go… (10pts)**

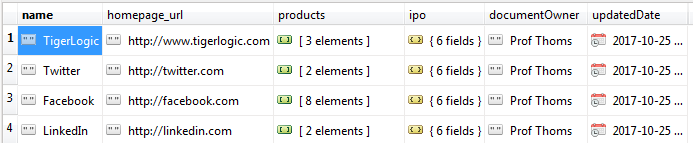
For Part IV, import companies.json into a new database nasdaq, under the collection, companies. For each query, ensure that your name and date is populated as shown below. Do not include the \_id field for any results. The below example returns documents where the company name is Facebook will show two additional fields for documentOwner and updatedDate.



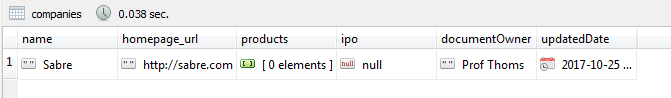
1. Create and run the query to return all companies that have offices in Camarillo, CA. Include fields for the company name, category and address.



1. Create and run the query to return all documents for social-based companies. To determine social companies, use the category code. Include only companies that are publicly traded (IPO is not NULL) and sort your results by the year they were founded in descending order.



1. Create and run the query to return the document for the oldest company where search is a core business function. To determine a company’s core business function, use the tag list.



*Use the mongodb function aggregate() to perform in-line query calculations for the following queries.*

1. Create and run the query to return the company with the most offices in New York.



1. Create and run the query to return the average number of years until IPO for companies that had an initial public offering.

