Fall 2017 Comp 533
Assignment 5
Due: December 1, 2017 at 11:45 PM

The goal of this assignment is to use the NoSQL database, MongoDB. We will reimplement some of our queries and exercises from the previous assignments using the document store and write some new queries and code.

Please make sure that every time you run the set of tests associated with a task, you use a "fresh" copy of the database. If you don't, you are likely to get the wrong answers. You can drop and recreate the collections, or just truncate and reload them.

What to turn in

You must turn in a .js file on Canvas. It must include all javascript code, including the queries to check your results.

Grading

Since the tasks vary in complexity, they are worth different amounts of points.

Within each task the percentage of points will be allotted based on the following guidelines:

- 0: Task/query not attempted, Task/Question does not give any results, or it does not compile
- 25%: Task/query compiles, runs and is most of the way towards a correct answer

- 75%: The task/query and answer it produces are almost correct, but there is a slight or subtle bug in the task.
- 100%: The task/query is correct and gives the right answer

The survey at the end is worth 5 points.

Academic Honesty

The following level of collaboration is allowed on this assignment: You may discuss the assignment with your classmates at a high level. What is not allowed is direct examination of anyone else's code (on a computer, email, whiteboard, etc.) or allowing anyone else to see your code.

You may use the search engine of your choice to lookup the syntax for MongoDB commands and better understand how the commands are used, but may NOT use it to find solutions to the tasks, either as guidance or for actual MongoDB javascript.

You MAY post and discuss results with your classmates.

1 Create the Collections

We're going to start with a clean copy of the database. Execute the following code to remove existing collections and recreate them. Be sure to backup your database, rename the collections or export content if you want to save anything you have created. The data is different from the last assignment and from what was posted earlier in the semester. So be sure to re-download and reload the data!

Commands to drop and load the collections can be found in the assignment5collections.js file. Note that the drop commands are javascript, but you must run the import commands at the command line.

2 Documents queries and actions

2.1 Simple Event, 2 points

Write a query that returns a single document from the event collection.

2.2 Event with criteria, 2 points

Write a query that returns all the documents from the event collection that are held at the location with id 5.

2.3 Events by Time, 3 points

Write a query that returns the 5 events that started closest in time to, but before September 7, 2017. Order the events from closest time to furthest away.

In other words, if we have:
Day1 Day2 Day3 Day4 Day5 Day6 Day7 Day8 Day9 Day10
and we ask for the 3 days before Day7, you would return:
Day6
Day5
Day4

2.4 Ticket Time, 5 points

In the ticket collection, write a query that returns the number of tickets created during event 16 before 4/12/2017 7:10 PM.

2.5 Event Name, 5 points

Change the name of the event named “Pi Day” to “Pie Day”.
Give a query that returns events named “Pi Day” OR “Pie Day”.

Change the name of all events named “GSA Coffee Break” to “GSA Study Break”.
How many documents were updated? (Paste the output of your MongoDB command)
Give a query that returns the number of events named “GSA Study Break” OR “GSA Coffee Break”.

2.6 MenuItems, 15 points

As separate objects (e.g. NOT in an array), list only the names of the menu items on the menu named “Sundae” Use the menu name “Sundae” to filter the results (not the id). Sort alphabetically.

2.7 TotalPrice, 20 points

In an earlier assignment, we wrote an update statement to populate the totalPrice field in the ticket table. Later, we wrote a trigger to perform the same functionality as new products were added to a ticket.

Using the new tkt collection, write (& submit) javascript that will set the value of the totalPrice field to the sum of the price of the products sold on the ticket. Note that there is now a field named “productPrice” in the tkt collection, in the productSold array. You may assume this value is correct and use it in your calculations.

Provide a query that returns only the id and totalPrice for tickets 5, 18, 343, and 1003. Sort the results by totalPrice, descending.

2.8 Turtle Sundae recipe, 20 points

What is the recipe for a “turtle sundae”? Return the component category name, the component name, the quantity, and the unit name. Order by the recipe lines by component category name, then by component name.

3 Open ended questions

3.1 Tkt: ProductSold & ItemSold, 6 points

ProductSold and ItemSold are embedded documents within tkt. List 1 advantage and 1 disadvantage of storing this new information in the tkt collection

3.2 Tkt: ProductPrice, 6 points

Product price is now stored in the tkt table.
List 2 advantages and 1 disadvantage of storing this new information in the tkt collection

3.3 Event collection, 6 points

The event collection is referenced from the tkt collection. List two advantages and one disadvantage of keeping this relationship a reference instead of embedding it in tkt.

3.4 Phone numbers, 4 points

The employee collection contains a single phone number for each employee. If we want to add additional phone numbers, should they be embedded documents or a separate collection? Justify your decision.

4 Survey (5 points)

It took me approximately N hours to complete this assignment, where N is:

What I like most about MongoDB / document stores is:

What I like least about MongoDB / document stores is: