Project 4 for CS585/DS503: Big Data Management - Fall 2017

Working with NoSQL Systems, in particular, MongoDB

Total Points: **100 points**

Given Out: **Monday, 13th Nov, 2017**

Due Date: **Monday, 4th Dec, 2017 (11:59PM)**

Submit the project via CANVAS.

Team Project:

Project is to be done in teams of two (2) students each, called your Project4-team. Team members will be assigned by CS585 staff.

References: Lecture notes and MongoDB Manual

Project Scope and Submission

In this project, you will work with MongoDB. Please see the lecture notes for several sources where to either work with MongoDB via a terminal or where to download MongoDB to install it locally on your own machine. The MongoDB version provided in the Virtual Machine does not have all features covered in class, given it is not the latest version.

Submission Mechanism

Submit all your CRUD statements in a single file electronically using CANVAS system.

You will submit below as one **single zip file** via CANVAS, namely:

1. This should contain a single text file containing your **MongoDB statements**.
2. This should also include a **report (.pdf or .doc)** containing explanations of your solutions.
3. In this report, you indicate the **relative tasks accomplished** by each of your team members to this project. You are also expected to describe **the team methodology,** i.e., how often you meet, how you communicated, how you shared documents, what tasks where accomplished in each order, and how the final results were put together and by which team member. It is expected that all team members would first produce the full solutions for ALL problems. That then as a team you would discuss your solutions, agree on the overall best answer to each problem, and then you submit a final result that everyone in the team agrees to.
4. Lastly, **each of you will independently** submit your personal assessment of your own and the team's efforts, contributions and team spirit to the CS585 staff via a survey (https://goo.gl/forms/UkZ4rMbb5yyXYUNE3). These comments will be treated confidentially.

Problem 1: Database Modeling, Loading and Modification. [42 points]

First, you create a MongoDB database, a collection "famous-people", and insert into this collection the 10 documents from this link: http://docs.mongodb.org/manual/reference/bios-example-collection/ Then apply the following 10 CRUD (Create/Read/Update/Delete) operations to this collection.

1) Write an operation(s) that changes the _id of "John McCarthy" to value 100.

2) Write an operation(s) that inserts the following new records into the collection:

```
{
  "_id" : 20,
  "name" : {
    "first" : "Mary",
    "last" : "Sally"
  },
  "birth" : ISODate("1933-08-27T04:00:00Z"),
  "death" : ISODate("1984-11-07T04:00:00Z"),
  "contribs" : [
    "C++",
    "Simula"
  ],
  "awards" : [
    {
      "award" : "WPI Award",
      "year" : 1999,
      "by" : "WPI"
    }
  ]
}
```

```
{
  "_id" : 30,
  "name" : {
    "first" : "Ming",
    "last" : "Zhang"
  },
  "birth" : ISODate("1911-04-12T04:00:00Z"),
  "death" : ISODate("2000-11-07T04:00:00Z"),
  "contribs" : [
    "C++",
    "FP",
    "Python",
  ],
  "awards" : [
    {
      "award" : "WPI Award",
      "year" : 1960,
      "by" : "WPI"
    },
    {
      "award" : "Turing Award",
      "year" : 1960,
      "by" : "ACM"
    }
  ]
}
```

3) Report all documents of people who got a "Turing Award" after 1960.

4) Report all people who got more than 2 awards.

5) Update the document of "Guido van Rossum" to add "Python" to the contribution list.

6) Insert a new field called "comments" of type array into document of "Mary Sally" storing the comments: "taught in 3 universities", "was an amazing pioneer", "lived in Worcester."

7) For each contribution by “Mary Sally”, say contribution “C”, list the people’s first and last names who have the same contribution “C”. For example, since “Mary Sally” has two contributions in “C++” and “Simula”, the output for her should be similar to:

{Contribution: “C++”,
People: [{first: “Mary”, last: “Sally”}, {first: “Ming”, last: “Zhang”}]},
{ Contribution: “Simula”,
… .}

8) Report all documents where the first name matches the regular expression “Jo*”, where “*” means any number of characters. Report the documents sorted by the last name.

9) Update the award of document _id =30, which is given by WPI, and set the year to 1965.

10) Add (copy) all the contributions of document _id = 3 to that of document _id = 30.

Problem 2: Aggregation and Advanced Query Specification. [28 points]

As for problem 1, create your database called “famous-people” by again loading the same 10 documents from: http://docs.mongodb.org/manual/reference/bios-example-collection/
We want to assure that the class starts with the identical content in their MongoDB database.

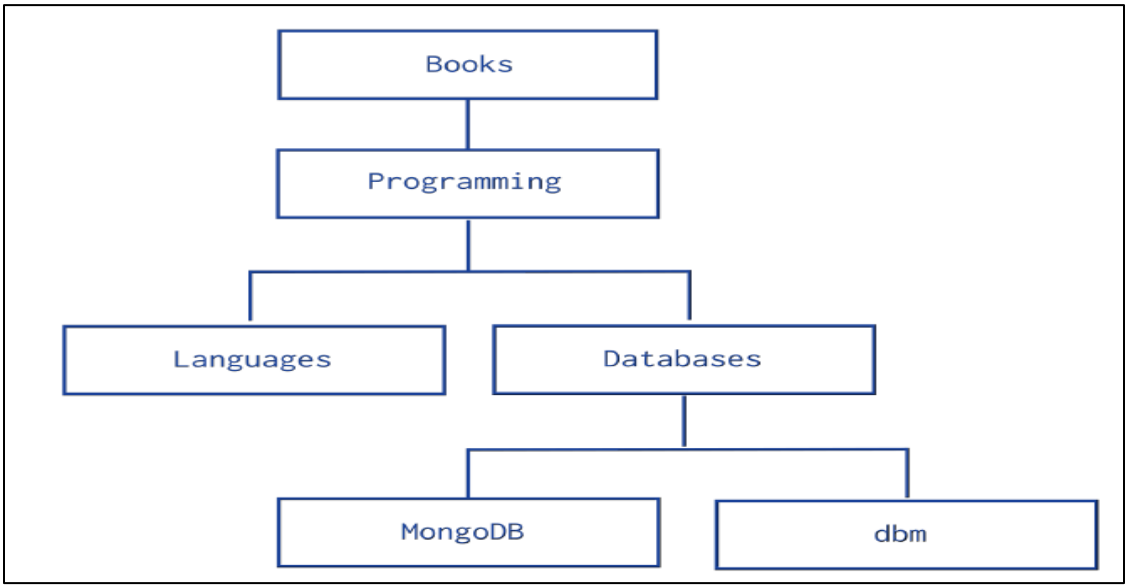
1) Write an aggregation query that groups by the award name, i.e., the “award” field inside the “awards” array, and for each award reports an array of the years for when this award has been given (Hint: Use map-reduce or use aggregation mechanism)

2) Write an aggregation query that groups by the birth year, i.e., the year within the “birth” field, and then report the count for each birth year (Hint: Use aggregate or map-reduce mechanism)

3) Report the document with the smallest and largest _ids. (Hint: Find first the values of the smallest and largest, and then after that find and report their documents.)

4) Search for and report all documents containing either “Turing” or “National Medal” as text substring. (Hint: Use $text operator to represent the string search).

Problem 3: Database Modeling Using Parent-Child Referencing. [30 points]



First assume we model the records and relationships in the above figure using the Parent-Referencing model.

1. Write a query to report the ancestors of "MongoDB", returning as output an array containing values:

   [{Name: "Databases", Level: 1},
   {Name: "Programming", Level: 2},
   {Name: "Books", Level: 3}]

   where level corresponds to the distance from "MongoDB" to other nodes.

2. Given only the root node, i.e., _id = "Books", write a query that reports the height of the tree. It the above case, this would return 4.

Next assume we model the records and relationships in the above figure using the Child-Referencing model.

1. Write a query to report the parent of "MongoDB".

2. Write a query to report the descendants of "Programming". The output should be an array containing values ["Languages", "Databases", "MongoDB", "dbm"]

3. Write a query to report the siblings of "Databases".

- the end -