Project 4 CS585/DS 503: Big Data Management Spring 2020

"Working with NoSQL Data Engine"

Total Points: 100 points
Given Out: Monday, 27th April, 2020
Due Date: Saturday, 9th May, 2020 (5:00PM)

Teams: Project is to be done in teams, called Project4-team.
Team members will be assigned by CS585/DS503 staff.

Project Scope and Submission

In this project, you will work with one representative NoSQL data engine, in particular, MongoDB. The task is to create, update and query unstructured data using MongoDB. Please see the lecture notes as well as on-line resources such as the MongoDB manual. If you find a resource that you like, please also share it with all your CS585 student colleagues on CANVAS. MongoDB has been provided to you in the Virtual Machine and/or you can install it also locally on your own machine.

Submission Mechanism

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 brief **report (.pdf or .doc)** containing explanations of your solutions and design choices with MongoDB.
3. In this report, you should describe **your team's methodology,** i.e., how often you met, how you communicated, how you shared the work and how finally you derived to an agreed-up project deliverable. **Relative tasks accomplished** by each of your team members are to be specified if the work was not done in close collaboration.
4. **Each team member independently** submits peer team comments to CS585/DS503 staff via https://forms.gle/eczSJY2R32CYxU7g6. This is your personal assessment of your own contributions and effort as well as the contributions of your team member to this project. These comments will be treated confidentially. *You will not be given a grade, until you submit your survey.* You are invited to talk to the CS585/DS503 about your team and/or project, as needed.

Expected Teamwork: *It is expected that all team members would first produce the full solutions for ALL problems to the best of their ability and in a timely fashion – or, if needed, you directly work on them today as a team. Thereafter, 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.*

Problem 1: Unstructured Data Modeling and Database Updating. [40 points]

First, you create a MongoDB database, a collection "famous-people", and insert into this collection the 10 documents from this link: https://docs.mongodb.com/manual/reference/bios-example-collection/ Then apply the following update 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 1940.

4) Report all people who got more than 1 award.

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 at 2 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 1999.

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

Problem 2: Query Specification over Unstructured Data in MongoDB. [30 points]

Create a MongoDB database, a collection “famous-people”, and insert into this collection 10 documents from this link: *https://docs.mongodb.com/manual/reference/bios-example-collection/* Then apply the following update operations to this collection.

1) Write an aggregation query that groups by the award name, i.e., the “award” field inside the “awards” array, and reports the number of times each award has been given. (Hint: Use Map-Reduce mechanism)

2) 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)

3) Write an aggregation query that groups by the birth year, i.e., the year within the “birth” field, and report both a total count and an array of _ids for each birth year.

4) Report the document with the smallest and largest _ids. You may first want to find the values of the smallest and largest, and then report their corresponding documents.

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

6) 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. Parent-Child Relationships in MongoDB. [30 pts]

This problem is based on the examples of modeling hierarchical relationships in MongoDB in Slides 17 to 33 in MongoDB Collection Modeling.

1) Assume we model the records and relationships in Figure 1 using the Parent-Referencing model (Slides 17 to 33 in MongoDB Collection Modeling). Write a query to report the ancestors of "MongoDB". The output should be an array containing values [{Name: "Databases", Level: 1},
{Name: "Programming", Level: 2},
{Name: "Books", Level: 3}]

("Level" is the distance from "MongoDB" node to the other node, which you need to compute.)

2) Assume we model the records and relationships in Figure 1 using the Parent-Referencing model. You are given only the root node, i.e., _id = "Books", write a query that reports the height of the tree. (It should be 4 in our case).

3) Assume we model the records and relationships in Figure 1 using the Child-Referencing model (also in Slides 17 to 33 in MongoDB Collection Modeling Slide Deck). Write a query to report the parent of "dbm".

4) Assume we model the records and relationships in Figure 1 using the Child-Referencing model. Write a query to report the descendants of "Books". The output should be an array containing values ["Programming", "Languages", "Databases", "MongoDB", "dbm"]

5) Assume we model the records and relationships in Figure 1 using the Child-Referencing model. Write a query to report the siblings "Databases".

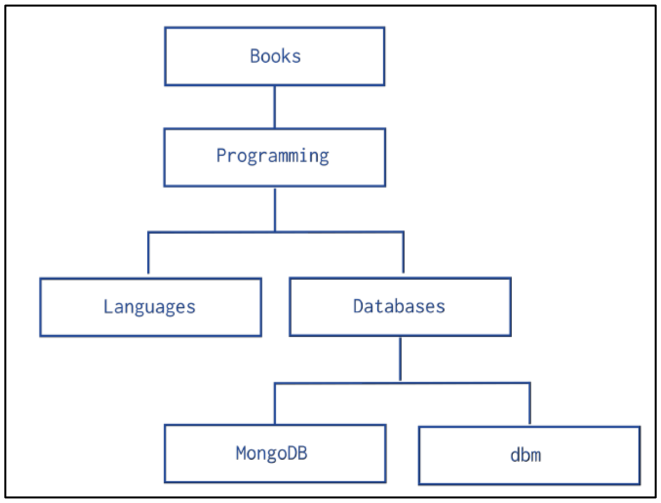

Figure 1: Tree Structure Relationships