Project for BS/MS - Big Data Management & Analytics

"Working with NoSQL Data Engine"

If you are not looking for the extra 1 graduate credit for the BS/MS for this course, you do NOT need to do this MongoDB project.

If you are intending to complete this project, please notify Professor Rundensteiner before the end of C term. It is fine if you registered in the C2025 term or D2025. Note that I'll not be regularly checking the CANVAS site for your project 4. Thus, please submit your BS/MS project 4 directly to Professor Rundensteiner with a clear subject line of "Extra BS/MS credit project for CS4433/DS4433 2025.

For BS/MS students that want to receive a total of 4.5 undergraduate credits for this course (equivalent to 3 graduate credits i.e., a full graduate course) towards the BS/MS degree, they need to complete this 4th project. Note that you will need to register for an additional independent study project listing the course name CS4433 or DS4433 as title and Prof. Rundensteiner as faculty on that registration.

If you are taking this course for BS/MS credits, but you only want to get the regular 3 undergraduate credits (which is equivalent to 2 graduate credits), then you do NOT need to complete this 4th project !

Total Points: 30 points

Due Date: Typically, by end of the term or semester that you are registering for this.

Logistic: This is an individual project, and not a team-based project. You need to submit your own work. You need to provide a statement as part of your submission That the work has all been done by you. If you use an LLM for any part of The project, you are required to report for what you used the LLM for. And, You can only be given credit for the effort that you actually put into the project.

Project Scope:

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 on CANVAS that you will self-study as well as on-line resources below.

Submission Mechanism

You will submit below as one **single zip file** containing:

1. a single text file containing your **MongoDB statements**.
2. a brief **report (.pdf or .doc)** with explanations of your solutions and design choices

Depending when you submit, the course CANVAS site may not be open. Thus, please email your solution to the instructor directly with a clear subject line labeled "BS/MS for DS4433".

Instructions, Manuals and Examples:

Online terminals for command and data model learning:

i. https://docs.mongodb.com/manual/tutorial/getting-started/
ii. https://mongoplayground.net/

Manuals and tutorials for MongoDB (with example code snippets:

i. https://dongodb.com/manual/
ii. https://docs.mongodb.com/manual/tutorial/

MongoDB on Cloud via MongoDB Atlas with instructions for configuration (optional):

i. https://www.mongodb.com/cloud/atlas

Problem 1: Unstructured Data Modeling and Database Updating. [10 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 "Grace Hopper" to value 333.

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

```
{
  "_id" : 222,
  "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" : 333,
  "name" : {
    "first" : "Ming",
    "last" : "Zhang"
  },
  "birth" : ISODate("1911-04-12T04:00:00Z"),
  "death" : ISODate("2000-11-07T04:00:00Z"),
  "contribs" : [
    "C++",
    "FP",
    "Python",
    "Simula"
  ],
  "awards" : [
    {
      "award" : "WPI Award",
      "year" : 1960,
      "by" : "WPI"
    },
    {
      "award" : "Turing Award",
      "year" : 1960,
      "by" : "ACM"
    }
  ]
}
```

3) Report all documents of people who got at least one “Turing Award” after 1940.

4) Report all people who got more than one “Turing Award”.

5) Update the document of “Grace Hopper” to add “was an inspiring pioneer overcoming the glass ceiling in computing” to the list of her contributions.

6) Insert a new field called “comments” of type array into document of “Mary Sally” storing the comments: “taught at 2 universities”, “was an inspiring pioneer overcoming the glass ceiling in computing”, “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, for each of “Mary Sally”’s contributions, the output should be :

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

8) Add (copy) all the contributions of document _id = 3 to that of document _id = 6.

9) 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.

10) Update the award of document _id =30, which is given by WPI, and set the year to 1999.

Problem 2: Query Specification over Unstructured Data in MongoDB. [10 points]

Create a MongoDB database: Ccollection called “famous-people”, and insert into this collection again the 10 documents from this link: https://docs.mongodb.com/manual/reference/bios-example-collection/

Task: 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 for each year it reports both a total count of people with that same birth year as well as an array of _ids of people with that same birth year.

4) Report the document with the smallest and largest _ids. As strategy, you may first want to find the values of the smallest and largest _id first, and only then find and return 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" **or** both as text substring.
(Hint: Use $text operator to represent the string search).

7) Search for and report all documents containing both "Turing" **and** "National Medal" as text substring.
(Hint: Use $text operator to represent the string search).

Problem 3. Parent-Child Relationships in MongoDB. [10 pts]

This problem is based on the examples of modeling hierarchical relationships in MongoDB (Will be in slide deck in MongoDB Collection Modeling on May 3, 2021).

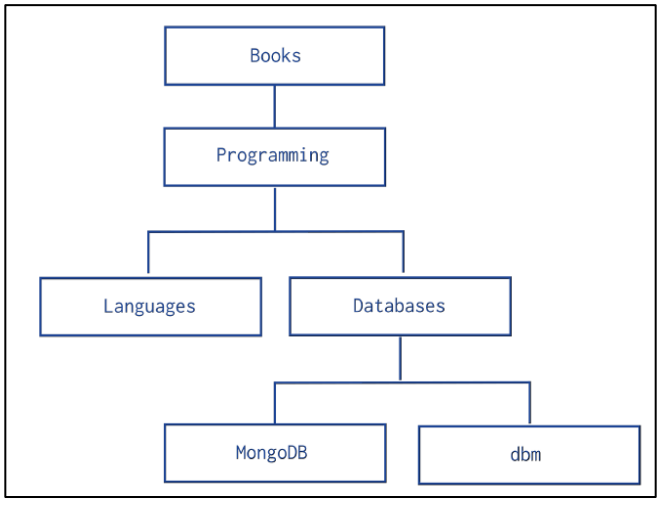

Figure 1: Tree Structure Relationships

1) Assume we model the records and relationships in Figure 1 above using the Parent-Referencing model (See MongoDB Collection Modeling slide deck in class).
   Write a query to report the ancestors of “MongoDB”. The output should be an array containing the following values:

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

   Where “Level” here denotes the distance from the “MongoDB” node target to the other node, which you need to compute.

2) Again assume that 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.
   (Hint: Test your query on multiple nodes! Also, for Books, the result would be 4).

3) Assume we model the records and relationships in Figure 1 using the Child-Referencing model Write a query to report the parent of “MongoDB”.

4) Assume we model the records and relationships in Figure 1 using the Child-Referencing model. Write a query to report the descendants of “Programming”. The output should be an array containing values [“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 of “Languages”.

****