Project 1 – Databases

You are a consultant at a company called Aggies Forever. The company has decided to create a fitness aware environment in the workplace. They have assigned a team to work specifically on their fitness portal called AggieFit. They want to design a system that will track the fitness activities of the employees and reward them for maintaining their health.

To maintain some anonymity, there is a separate portal in which employees create a unique ID for AggieFit. The map between the unique ID for the fitness portal and employee records are hidden from the AggieFit team. The employees can, however, choose to expose their data. The employees can also choose to add their personal details (e.g., height, weight, age etc) to AggieFit. Employees must choose from a list of fitness goals. They will then report their activities by integrating their fitness devices.

Your task is to assist the AggieFit team in designing their system by providing prototypes and comments. As they design their system, they ask you for help in two aspects of their system.

They release their first task to you by 9/15/2017 and the second task by 9/22/2017. They expect to have your input on both tasks by 10/6/2017, 11:59pm CDT.

Task 1: P1-T1 – Fitness data on MongoDB

Due: 10/6/2017, 11:59pm CDT

Early bird deadline: 9/25/2017, 11:59pm CDT

1. Provided information

As their resident database expert, they want you to design a database for AggieFit. They have some dummy fitness data for you to play around with. Each employee uses a different device to track their activities, and some employees use multiple devices. The employees may have shared their personal information with AggieFit. Some employees may also have decided to make their employment information available to AggieFit.

2. Installation

They want you to use MongoDB, a document-oriented database, to store the fitness data. First, they want you to install a MongoDB client in your machine and verify that you can connect to their database.

NOTE: You will learn about MongoDB in the week of 9/18.

Download and install MongoDB:

1. Download MongoDB from: https://www.mongodb.com/download-center#community
2. Install MongoDB: https://docs.mongodb.com/manual/administration/install-community/
3. Connect the MongoDB client on 34.233.78.56 and use the database `fitness_X,` by running `use  fitness_X`, where `X` your UIN. E.g., if your UIN is 111001111, you will execute `use fitness_111001111`. Verify that you are connected by running the command: `db.fitness_X.find().pretty()`. The query should return some values if you are connected to the database.
4. AggieFit team suggests you to use python to connect to MongoDB and provide you with some helper scripts under MongoDB-Pythons-scripts. You are free to use other tools. To use python,
    a. Download and install python 2.7.13 on: https://www.python.org/downloads/
    b. Run python and install pymongo: http://api.mongodb.com/python/current/installation.html
    c. Under eCampus -> Content -> Project 1 -> P1-T1 files, you will find python scripts to connect to MongoDB. A function to connect to a remote MongoDB database is provided in `mongoConnect.py`. You will need to modify `constants.py`. Update `db` and `collection` variables to `fitness_X`, where `X` is your UIN. Run `queries.py` to verify that your installation is completed. The script should print some values if you are connected to the database.

Tip: Watch an example of how to complete steps 3 and 4 (c) in the example video posted on eCampus -> Content -> Project 1. Note that the example is on a different server and database.

3. The tasks

WARNING: Watch lecture videos released on 9/18/2017 before attempting this step.

P1-T1-a: The AggieFit team tells you that their senior management wanted to use a relational database for this application, but they think MongoDB fits better. Now, they need to convince the senior management that their choice is indeed appropriate. They ask you to provide a short write-up to validate their choice of database, specifically, what features of MongoDB makes it more suitable for their application.

P1-T1-b: AggieFit team has dumped some dummy data into a collection in a MongoDB database. Additionally, they have provided you with more fitness data under eCampus -> Content -> Project 1 -> FitnessData. The data is in a file "`dummy-fitness.json`". It contains the fitness data in JSON format. Updates to employees' personal data and new tags for users come in a JSON formatted file. They provide a sample update file for a dummy employee in `user1001-new.json`.

They want you to run queries ordered as follows:

WQ1. Add the data in `dummy-fitness.json` to the MongoDB database

WQ2. Update the database with data from `user1001-new.csv`.

RQ1. Count the number of employees whose data is in the AggieFit database.

RQ2. Retrieve employees who have been tagged as "`active`".

RQ3. Retrieve employees that have a goal activity duration greater than 60 minutes.

RQ4. Aggregate the total activity duration for each employee. If the employee does not have activity duration in their data, you can report their total activity duration as 0.

P1-T1-c: Currently, all the data for AggieFit is stored in one single server. The team is not sure if more servers are required. They are asking you for your opinion and comments on what could go wrong if they store all their data in one server.

If they do decide to put their data on different servers, they want to know whether each server should contain copies of the same data, or they should split the data among the servers. They want you to provide your inputs on what they need to consider before making the decision.

Note: If any of the information provided is unclear, please ask for clarification on Piazza.

5. What the AggieFit team wants you to do

You should do the following to complete this task.

WARNING: Watch lecture videos released on 9/18/2017 before attempting this step.

Do the following on the database:

P1-T1-b: Execute the 6 queries on your database. You will need to execute the queries in the order specified by your client.

You need to **prepare a report** that contains the following:

1. The time you took to complete this task
2. P1-T1-a: A section on the features of MongoDB that make it suitable for the application. If you disagree with your client, you will have to provide a thorough explanation.
3. P1-T1-b: Include any comments you have about improving the design of the database.
4. P2-T1-c:
    i. A section with your comments and opinions about what could go wrong with placing all their data in one server.
    ii. A section with your comments and opinions on how data should be spread across multiple servers.

You may use the python interface to connect to MongoDB. In `queries.py`, you will find comments that explain the syntax to perform the basic operations required in this assignment. You can use them to send the commands to the database.

6. Project submission and grading rubric

You submit a zip file containing your report and code. The submission "button" is located on eCampus -> Content -> Project 1.

The zip file should include your report, and a file containing your queries for WQ1, WQ2, RQ1, RQ2, RQ3, and RQ4. If you use a client library (e.g., python), you can submit the source code in a single file (e.g., queries.py) along with instructions on how to run the code.

You need not use github for this challenge.

This challenge is worth 60 points, distributed as follows:

P1-T1-a: 5 points

P1-T1-b: 45 points

P1-T1-c: 10 points

If you submit by the early bird deadline, you get 5 points extra.

Partial credit will be applied if the answers are not thorough.