Department of Computer Engineering Faculty of Engineering

Kasetsart University

HW# NoSQL & MongoDB

In this homework, you should create evidence of your work such as a picture with description in PDF and submit the Github link of your homework. Make sure that your Github link is public.

1) You're creating a database to contain a set of sensor measurements from a two-dimensional grid. Each measurement is a time-sequence of readings, and each reading contains ten labeled values. Should you use the relational model or MongoDB? Please justify your answer

Ans: If the sensor data has a fixed structure and the application requires complex queries, use a **relational database model** like **MySQL**. If the sensor data structure is flexible and the application requires scalability and performance, use a **NoSQL** database like **MongoDB**.

2) For each of the following applications
   a. IoT
   b. E-commerce
   c. Gaming
   d. Finance

   Propose an appropriate Relational Model or MongoDB database schema. For each application, clearly justify your choice of database.

   Ans:

a. IoT: MongoDB is a suitable choice for IoT applications as it can handle large volumes of unstructured data from various sources and provide high scalability and performance.

b. E-commerce: A relational database model is a suitable choice for E-commerce applications as it can handle complex data relationships, support transactions, and provide strong consistency guarantees.

c. Gaming: For multiplayer gaming applications, a NoSQL database like MongoDB may be a better choice as it can handle real-time updates and provide high scalability and performance.

d. Finance: A relational database model is a suitable choice for finance applications as it provides strong consistency guarantees and supports complex transactions and queries.

The choice of the database depends on the specific requirements of the application, including the type of data being managed, the volume of data, the need for scalability and performance, and the complexity of the data relationships.

3) Create a MongoDB database with the following information.

1) ({"name":"Ramesh","subject":"maths","marks":87})
2) ({"name":"Ramesh","subject":"english","marks":59})
3) ({"name":"Ramesh","subject":"science","marks":77})
4) ({"name":"Rav","subject":"maths","marks":62})
5) ({"name":"Rav","subject":"english","marks":83})
6) ({"name":"Rav","subject":"science","marks":71})
7) ({"name":"Alison","subject":"maths","marks":84})
8) ({"name":"Alison","subject":"english","marks":82})
9) ({"name":"Alison","subject":"science","marks":86})
10) ({"name":"Steve","subject":"maths","marks":81})
11) ({"name":"Steve","subject":"english","marks":89})
12) ({"name":"Steve","subject":"science","marks":77})
13) ({"name":"Jan","subject":"english","marks":0,"reason":"absent"})

Create a MongoDB database with the given information by create a collection called "students"

Give MongoDB statements (with results) for the following queries

- **Find the total marks for each student across all subjects.**

```
geeegrace — mongosh mongodb://<credentials>@127.0.0.1:27017/...
}
studentslen> db.students.aggregate([
...   {$group: {_id: "$name", total_marks: {$sum: "$marks"}}}
... ])
[
  { _id: 'Rav', total_marks: 216 },
  { _id: 'Ramesh', total_marks: 223 },
  { _id: 'Steve', total_marks: 247 },
  { _id: 'Jan', total_marks: 0 },
  { _id: 'Alison', total_marks: 252 }
]
studentslen>
```

- **Find the minimum marks scored by each student.**

```
geeegrace — mongosh mongodb://<credentials>@127.0.0.1:27...
studentslen> db.students.aggregate([
...   {$group: {_id: "$subject", max_marks: {$max: "$marks"}}}
... ])
[
  { _id: 'english', max_marks: 89 },
  { _id: 'science', max_marks: 86 },
  { _id: 'maths', max_marks: 87 }
]
studentslen>
```

- **Find the minimum marks scored by each student.**

```
geeegrace — mongosh mongodb://<credentials>@127.0.0.1:27...
]
studentslen> db.students.aggregate([
...   {$group: {_id: "$name", min_marks: {$min: "$marks"}}}
... ])
[
  { _id: 'Ramesh', min_marks: 59 },
  { _id: 'Rav', min_marks: 62 },
  { _id: 'Alison', min_marks: 82 },
  { _id: 'Jan', min_marks: 0 },
  { _id: 'Steve', min_marks: 77 }
]
studentslen>
```

- **Find the top two subjects based on average marks.**

```
geeegrace — mongosh mongodb://<credentials>@127.0.0.1:27...
studentslen> db.students.aggregate([
...   {$group: {_id: "$subject", avg_marks: {$avg: "$marks"}}},
...   {$sort: {avg_marks: -1}},
...   {$limit: 2}
... ])
[
  { _id: 'maths', avg_marks: 78.5 },
  { _id: 'science', avg_marks: 77.75 }
]
studentslen>
```