# Intro to MongoDB I

Discussion material introduced with Unit 2

I435 / I535 / B669: Management, Access and Uses of Big Data
Fall 2016
School of Informatics and Computing
Indiana University

# What is MongoDB

- MongoDB is a cross-platform, document oriented database.
- Database
  - Database is the container for collections.
- Collection
  - Collection is a group of documents.
- Document
  - Document is a set of key-value pairs

| MySQL | MongoDB |
|---|---|
| Table | Collection |
| Row | Document |
| Column | Field |
| Joins | Embedded documents, linking |

# MongoDB Data Modeling

- Sample document in MongoDB (key-value pair)

```
{
    _id: ObjectID(7df78ad8902c)
    title: 'MongoDB sample',
    description: 'This is a sample document',
    tags: ['MongoDB', 'sample', 'NoSQL'],
    number: 1000,
    comments: [
        {
            user: 'luoyu',
            message: 'first message'
        },
        {
            user: 'luoyu',
            message: 'second message'
        }
    ]
}
```

# Using MongoDB instead of MySQL

- No schema definition necessary beforehand
- Documents don’t need to all have the same fields
- No complex relationship between documents
- Strong query ability. Dynamic queries on documents using a document-based query language
- Use internal memory for storing the working set
- Document representation corresponds to native programming data structures, e.g. lists and dicts in Python.

# Data Modeling

- **_id** can be any data type. In this, case **ObjectId** is a 12 bytes hexadecimal number which assures the uniqueness of every document in a collection.
- MySQL has same constraint: Primary Key
- For MySQL, having a primary key is optional. But for MongoDB, there must be a **_id** in the document. (If you don't input the **_id**, MongoDB will automatically generate the**_id** for the document)

# Data Modeling

- MongoDB supports many data types

| String | UTF-8 valid | Object | Another Document |
|---|---|---|---|
| Integer | Numerical value | Null | Null value |
| Boolean | True/False | Symbol | Specific symbol type |
| Double | Floating point values | Date | Current date or time in UNIX time format |
| Min/Max key | Compare a value against the lowest and highest BSON elements | Object ID | Document's ID |
| Arrays | Arrays of list or multiple values into one key | Binary data | Binary data |
| Timestamp | Handy for recording when documents are modified | Code | JavaScript code |
| Regular expression | Regular expression | | |

# Data Modeling

- JavaScript Object Notation – JSON
  - Open, human and machine readable standard
- MongoDB use JSON documents to store records, just as tables and rows of relational database.

```
{
    _id: ObjectID(7df78ad8902c)
    title: 'MongoDB sample',
    description: 'This is a sample document',
    tags: ['MongoDB', 'sample', 'NoSQL'],
    number: 1000,
    comments: [
        {
            user: 'luoyu',
            message: 'first message'
        },
        {
            user: 'luoyu',
            message: 'second message'
        }
    ]
}
```

# Data Modeling

```
{
    _id: ObjectID(7df78ad8902c)
    title: 'MongoDB sample',
    description: 'This is a sample document',
    tags: 'MongoDB',
    number: 1000
}
```

- Special symbols:
  - { }
  - [ ]
  - ,
  - :

# JSON

- Support two data structures:
  - Objects
    - For example:
      - name : ‘sample’
      - location : {‘name’ : ‘Bloomington’}
  - Ordered list
    - For example:
      - list : [‘String’, ‘Boolean’]
      - list : [{‘type’ : ‘String’} , {‘type’ : ‘Boolean’}]

# Data Modeling

- In class exercise: Describe the following course information in a MongoDB document.
  - Course: Management, Access, and Use of Big and Complex Data
  - Unit List:
    - Unit 0 : Introduction
    - Unit 1 : Big Data Intro
    - Unit 2 : Data Pipelines
  - AI: Yu, Dimitar

# Solution

```
{
    _id : ObjectID(7df78ad8902c) or _id : 'any value',
    CourseName : 'Management, Access, and Use of Big and Complex Data',
    Unitlist : [{'Unit 0': 'Introduction'},
                {'Unit 1': 'Big Data Intro'},
                {'Unit 2': 'Data Pipelines'}
                ],
    people : [
        {
            person: 'Yu',
            role: 'AI'
        },
        {
            person: 'Dimitar',
            role: 'AI'
        }
    ]
}
```

# Document Relationships

- Relationship
  - **One-to-one** relationship, e.g. a course and its name
  - **One-to-many** relationship, e.g. professor and her courses
  - **Many-to-many** relationship, e.g. courses and students

# Embedded Document

- Relationship
  - One document contains all information or other documents
- Simple but Heavy
  - Simple: easy to put all information in one document.
  - Heavy: hard to read the document.

# Embedded Document

```
{
    _id : ObjectID(7df78ad8902c) or _id : 'any value',
    CourseName : 'Management, Access, and Use of Big and Complex Data',
    Unitlist : [{'Unit 0': 'Introduction'},
                {'Unit 1': 'Big Data Intro'},
                {'Unit 2': 'Data Pipelines'}
                ],
    people : [
        {
            person: 'Yu',
            role: 'AI'
        },
        {
            person: 'Dimitar',
            role: 'AI'
        }
    ]
}
```

# Document References

- References store the relationship between data by including links or references from one document to another.
- In MongoDB, the documents will contain a field that will reference the address document's id filed.
- MySQL equivalent: Primary and Foreign key
- Example

```
{
    _id : ObjectID(7df78ad8902c),
    address_ids: [
        ObjectId("52ffc4a5d85242602e000000"),
        ObjectId("52ffc4a5d85242602e000001")
    ]
}
```

# Exercise

- Like the previous, but use different documents for the course, unit list and Ais, and use appropriate references.
- Use references structure to build the document.
  - Course: Management, Access, and Use of Big and Complex Data
  - Unit List:
    - Unit 0 : Introduction
    - Unit 1 : Big Data Intro
    - Unit 2 : Data Pipelines
  - AI: Yu, Dimitar

# Solution 1

```
{
    _id : ObjectID(7df78ad8902c) or _id : 'any value',
    CourseName : 'Management, Access, and Use of Big and Complex Data',
    Unitlist : ['Unit 0': ObjectId("52ffc4a5d85242602e000000"),
                'Unit 1': ObjectId("52ffc4a5d85242602e000001"),
                'Unit 2': ObjectId("52ffc4a5d85242602e000002")
    ],
    comments : [
        {
            person: 'Yu',
            role: 'AI'
        },
        {
            person: 'Dimitar',
            role: 'AI'
        }
    ]
}

{
    _id : ObjectId("52ffc4a5d85242602e000000"),
    title: 'Introduction',
}

{
    _id : ObjectId("52ffc4a5d85242602e000001"),
    title: 'Big Data Intro',
}

{
    _id : ObjectId("52ffc4a5d85242602e000001"),
    title: 'Data Pipelines'
}
```

# Solution 2

```
{
    _id : ObjectID(7df78ad8902c) or _id : 'any value',
    CourseName : 'Management, Access, and Use of Big and Complex Data',
    Unitlist : [{'Unit 0': 'Introduction'},
                {'Unit 1': 'Big Data Intro'},
                {'Unit 2': 'Data Pipelines'}],
    AIs : [
            ObjectId("52ffc4a5d85242602e000000"),
            ObjectId("52ffc4a5d85242602e000001")
    ]
}


{
    _id : ObjectId("52ffc4a5d85242602e000000"),
    person: 'Yu',
    role : 'AI'
}

{
    _id : ObjectId("52ffc4a5d85242602e000001"),
    person: 'Dimitar',
    role : 'AI'
}
```