Graph & Modern Databases

COMP1835

1

1

MongoDB

2

2

Objectives

- To introduce MongoDB
- To discuss data model
- To learn how to create and manipulate collections
- To use CRUD commands with documents

3

3

Part 1

4

4

Overview

- MongoDB is a document database designed for ease of development and scaling.
- MongoDB offers both a *Community* and an *Enterprise* version of the database
- First publicly released in 2009
- It was designed as a scalable database with performance and easy data access as a core design goals
- Its name comes from the word 'Humongous'
- It is easy to install and implement

5

5

MongoDb Facts

- Official Online Resources
  - www.mongodb.org
- History
  - Created at 10gen (now MongoDB Inc) in 2007
  - It was initially developed as a PAAS (Platform As A Service). Later in 2009, it was introduced in the market as an open source database server that was maintained and supported by MongoDB Inc.
- Technologies and Language
  - Implemented in C++.
- Access Methods
  - A JavaScript command-line interface. Drivers exist for a number of languages including C, C#, C++, Erlang. Haskell, Java, JavaScript, Perl, PHP, Python, Ruby, and Scala.
  - Several GUI are available:
    - MongoDB Compass, from MongoDb, community and full enterprise versions
    - NoSQLBooster (formerly MongoBooster) from NoSQLBooster Inc, free version
    - Studio 3T from Studio3T Inc, paid versions (Core, Pro, Enterprise)
    - Robo 3T (formerly Robomongo) – open source

- Query Language
  - SQL-like query language.
- Open-Source License
  - Community version (comes with community version of MongoDB Compass)
- Who Uses It
  - Now 4,100+ companies using MongoDB:
    - Google, Facebook, Adobe, GAP, ebay, PayPal, HM Revenue & Customs, Met Office…

6

6

Advantages

- **Schema less**
  - MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.
  - Structure of a single object is clear.
- **Rich Query Language**
  - MongoDB supports dynamic queries on documents using a document-based query language to support read and write operations (CRUD), Data Aggregation, Text Search and Geospatial Queries.
- **High Performance**
  - MongoDB provides high performance data persistence.
  - It supports embedded data models and reduces I/O activity on database system.
  - Indexes support faster queries and can include keys from embedded documents and arrays.
- **High Availability**
  - MongoDB's replication facility, called replica set, provides: automatic failover and data redundancy.
  - A replica set is a group of MongoDB servers that maintain the same data set, providing redundancy and increasing data availability.
- **No SQL injection**
  - MongoDB is not susceptible to SQL injections (putting SQL statements in web forms or other input from the browser that compromises the DB security) because objects are stored as objects, not by using SQL strings.
- Some common places where MongoDB is used include mobile apps, product catalog, real-time personalization, content management and applications built around the relational data model and SQL.

7

7

Disadvantages

- MongoDB
  - is a NoSQL database and as a result, it is not ACID compliant.
    - In the applications where ACID compliance (for example, applications that require database-level transactions) is mandatory, MongoDB cannot be used.
  - doesn't support joins.
  - doesn't have the provision for stored procedures.
  - It scales well in a narrow range, sharding is annoying and very complex (for example, Cassandra scales better, but it is a different type store)
- Map/Reduce is somewhat slow
- It is hard to secure MongoDB properly without going with an Enterprise license.

8

8

Databases and Collections

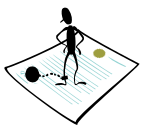
- MongoDB stores data records in BSON documents on disk.
  - BSON is a binary representation of JSON documents, though BSON data format provides more data types than JSON.
- BSON documents are stored in collections:
  - Collections are similar to tables in relational databases.
  - However, in MongoDB, a collection is not enforced by a strict schema
  - Documents in a collection can have a slightly different structure from one another
- Collections are stored in databases.
- Namespace:
  - The namespace is a combination of the database name and the name of the collection or index:

```
[database-name].[collection-or-index-name]
```

9

9

Naming Restrictions



- Database names
  - are not case sensitive
  - cannot be empty and must have fewer than 64 characters
  - cannot contain any of the following characters: /\. "$*<>:|? (for MongoDB deployments running on Windows)
  - cannot contain any of the following characters: /\. "$ (for MongoDB deployments running on Unix and Linux systems)
- Collection names
  - should begin with an underscore or a letter character
  - cannot:
    - contain the $.
    - be an empty string (e.g. "").
    - contain the null character.
    - begin with the system. prefix. (Reserved for internal use.)
- Namespace
  - The maximum length of the collection namespace, which includes the database name, the dot (.) separator, and the collection name (i.e. <database>.<collection>), is 120 bytes.

10

10

Document

- A record in MongoDB is a document, which is a data structure composed of field and value pairs
  - similar to JSON objects
- The values of fields may include other documents, arrays, and arrays of documents.

```
{
  name:"Peter",
  age: 25,
  city:"London",
  modules:["COMP1802","COMP1707"],
}
```

field: value
field: value
field: value
field: value

- MongoDB supports no more than 100 levels of nesting for BSON documents

11

11

Documents

- Documents (i.e. objects) correspond to native data types in many programming languages.
- Embedded documents and arrays reduce need for expensive joins.
- Maximum size of a document in MongoDB is 16MB
  - The maximum document size helps ensure that a single document cannot use excessive amount of RAM or, during transmission, excessive amount of bandwidth.
- The field names cannot contain:
  - null characters, dot (.) or dollar signs($)
  - `_id` field name is reserved for the Object ID (a unique ID for the system)

12

12

Data Model

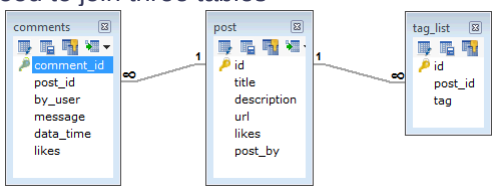
- Before implementing a MongoDB database, you need to have a clear understanding of how the data will be stored and accessed:
  - What basic objects will my application be using?
  - What is the relationship between the different object types (one-to-one, one-to-many, or many-to-many)?
  - How often will new objects be added to the database? Deleted? Changed?
  - How will objects be accessed – by ID, property values, comparisons, other?
  - How will groups of objects types be accessed – common ID, common property values, other?

13

13

Data Model – Example

- Imagine a client needs a database design for his blog/website and see the differences between RDBMS and MongoDB schema design. Website has the following requirements.
  - Every post has the unique title, description and url.
  - Every post can have one or more tags.
  - Every post has the name of its publisher and total number of likes.
  - Every post has comments given by users along with their name, message, data-time and likes.
  - On each post, there can be zero or more comments.
- In RDBMS schema, design for above requirements will have minimum three tables.
  - So, to query data you need to join three tables



14

14

Data Model – Example

- MongoDB design in this case will have one collection ***post*** and the following structure:
- In order to query data in MongoDB, you would need to access one collection only.

```
{
   _id: POST_ID
   title: TITLE_OF_POST,
   description: POST_DESCRIPTION,
   by: POST_BY,
   url: URL_OF_POST,
   tags: [TAG1, TAG2, TAG3],
   likes: TOTAL_LIKES,
   comments: [
      {
         user: COMMENT_BY,
         message: TEXT,
         dateCreated: DATE_TIME,
         like: LIKES
      },
      {
         user: COMMENT_BY,
         message: TEXT,
         dateCreated: DATE_TIME,
         like: LIKES
      }
   ]
}
```

15

Quiz

1. A collection and a document in MongoDB is equivalent to which of the SQL concepts respectively?
   A. Table and Row
   B. Table and Column
   C. Column and Row
   D. Database and Table
2. What is the maximum size of a MongoDB document?
   A. 12 MB
   B. 16 MB
   C. 64 MB
   D. There is no maximum size. It depends on the RAM.

16

16

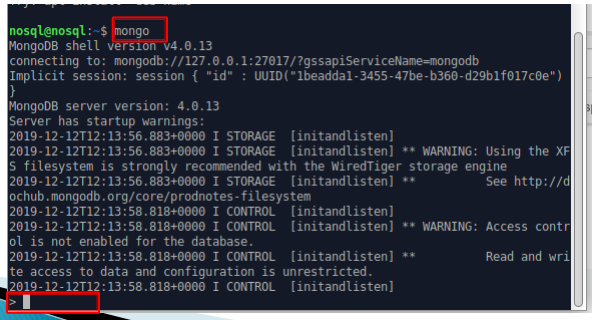
Part 2

17

17

The mongo Shell

- The mongo shell is an interactive JavaScript interface to MongoDB.
- You can use the mongo shell to query and update data as well as perform administrative operations.
- To start mongoDB shell use **mongo** command



18

18

Create Database

- To create a database in MongoDB you need to use the following command:

  use DATABASE_NAME

  - The command will create a new database if it doesn't exist, otherwise it will return the existing database.
    - In MongoDB default database is **`test.`** If you didn't create any database, then collections will be stored in **`test`** database.

```
>use mydb
switched to db mydb

>db
mydb

>show dbs
local      0.78125GB
test       0.23012GB
```

To create/switch to your database

To check your currently selected database

To see all your databases

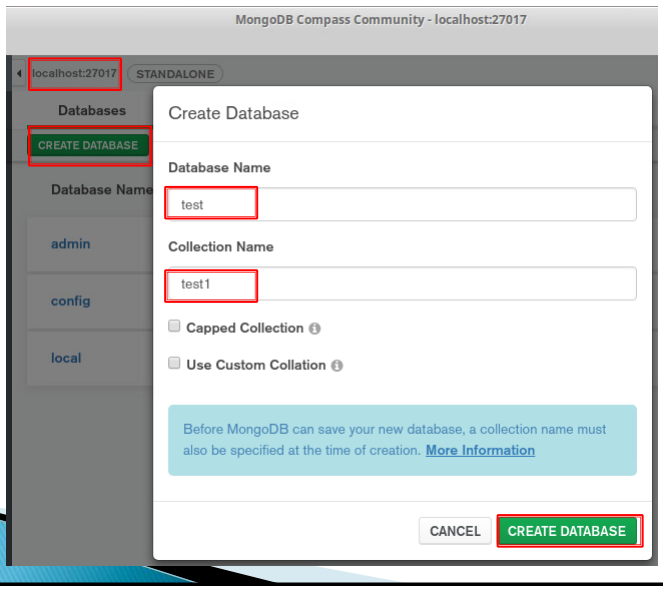
Where is mydb?

  - Newly created database **`mydb`** is not present in list.
    - To display database, you need to insert at least one document into it.

19

19

Create Database in MongoDB Compass



20

20

Drop Database

- After a database has been created, it exists in Mongo DB until the administrator deletes it.
- To delete a database from the MongoDB shell, use the **`dropDatabase()`** method.
- Example:

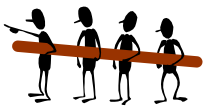
```
>use mydb
switched to db mydb

>db.dropDatabase()
>{ "dropped" : "mydb", "ok" : 1 }
>
```

- In JavaScript

```
myConn = new Mongo ("localhost");
newDB = myConn.getDB("myDB")
newDB.dropDatabase();
```

21

Create Collections



- A collection is simply a grouping of documents that have the same or similar purpose.
  - You use **`Collection`** object to access documents in the collection, add documents, query them, etc.
  - To create a collection use **`db.createCollection(name, options)`**
    - Where: name is name of collection to be created.
    - Options is an optional parameter= document and is used to specify configuration of collection.
- Example:

```
>use test
switched to db test
>db.createCollection("video_records")
{ "ok" : 1 }
>
>show collections
video_records
```

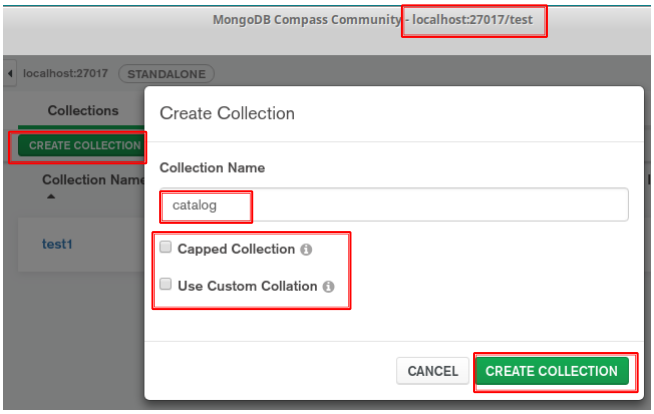
To see all your collections

- In MongoDB, you don't need to create collection. MongoDB creates collection automatically, when you insert some document.

```
>db.mycollection.insert({"name" : "mycatalog"})
>show collections
mycollection
video_records
```

22

Create Collection in MongoDB Compass



- Capped collection
  - A fixed-sized collection that automatically overwrites its oldest entries when it reaches its maximum size; it is similar to circular buffers
- Collation
  - allows users to specify language-specific rules for string comparison, such as rules for letter case and accent marks.

23

23

Drop Collection

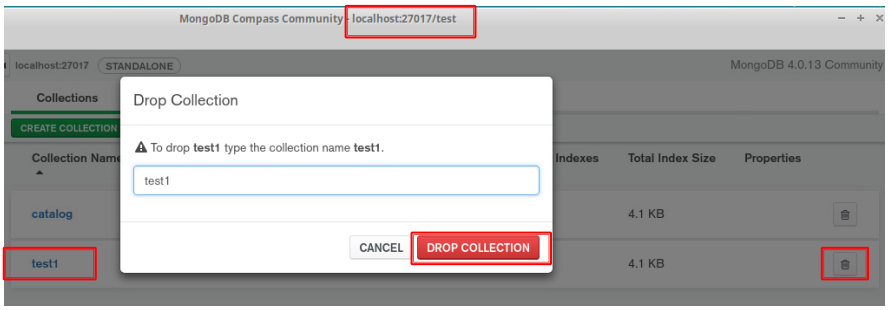
- In order to drop a collection from the database use **`db.collection.drop()`**
  - The method will return true, if the selected collection is dropped successfully, otherwise it will return false.
- Example:

```
>use test
switched to db test
>db.createCollection("mycollection")
{ "ok" : 1 }
>
>show collections
mycollection
video_records

>db.mycollection.drop()
true
>show collections
video_records
```

24

24

Drop Collection in MongoDB Compass



25

25

Quiz

1. What command will you use to list all available databases?
   A. show databases
   B. show db
   C. show dbs
   D. show alldb

2. What is default port for MongoDB server?
   A. 12701
   B. 27017
   C. 27071
   D. 3306

3. Use database_name operation switches to the specified database. If the database does not currently exist, that operation will create the database.
   A. True
   B. False

26

26

Part 3

Insert Document

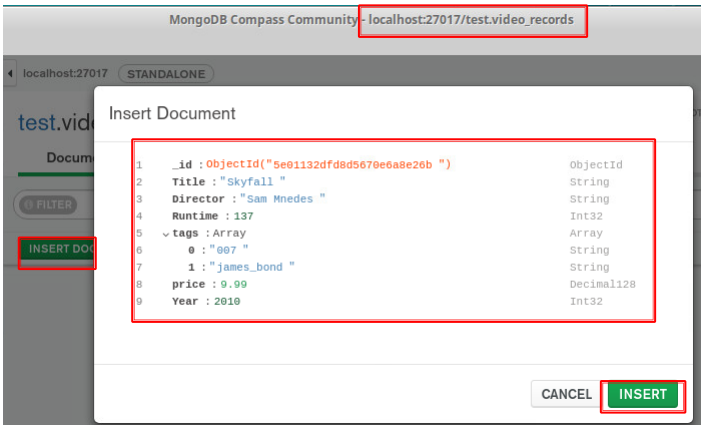
- To insert document into MongoDB collection, you need to use
  - **`insert()`** or **`save()`** methods.

```
>db.video_records.insert({
   _id: 1 ),
   Title: "Skyfall",
   Director: "Sam Mendes",
   RunTime: 137,
   tags: ["007","james_bond"],
   likes: 1000,
   price: 9.99,
   Year: 2010
})
```

```
>db.video_records.save({
   _id: 1 ),
   Title: "Skyfall",
   Director: "Sam Mendes",
   RunTime: 137,
   tags: ["007","james_bond"],
   likes: 1000,
   price: 9.99,
   Year: 2010

})
```

- If we don't specify the _id parameter, then MongoDB assigns a unique ObjectId for this document.
  - _id is 12 bytes hexadecimal number unique for every document in a collection.

Insert Document using MongoDB Compass



29

29

Data Types

- MongoDB stores documents on disk in the BSON serialization format.
- BSON supports the following data types as values in documents:
  - Each data type has a corresponding number (an integer ID number from 1 to 255) that can be used with the $type operator to query documents by BSON type.
  - You can specify either the number or alias for the BSON type

Type	Number	Alias
Double	1	“double”
String	2	“string”
Object	3	“object”
Array	4	“array”
Binary data	5	“binData”
ObjectId	7	“objectId”
Boolean	8	“bool”

Type	Number	Alias
Date	9	“date”
Null	10	“null”
Regular Expression	11	“regex”
JavaScript	13	“javascript”
32-bit integer	16	“int”
Timestamp	17	“timestamp”
64-bit integer	18	“long”

30

30

Query Document

- To query data from MongoDB collection, you need to use MongoDB's **`find()`** method.
  - Method returns a cursor to the documents that match the query criteria.
  - To display the results in a formatted way, you can use **`pretty()`** method.

```
> use test
switched to db test
> db.video_recordings.find()
> db.video_records.find()
{ "_id" : ObjectId("5e01132dfd8d5670e6a8e26b"), "Title" : "Skyfall", "Director"
: "Sam Mnedes", "Runtime" : 137, "tags" : [ "007", "james_bond" ], "price" : Num
berDecimal("9.99"), "Year" : 2010 }
> db.video_records.find().pretty()
{
        "_id" : ObjectId("5e01132dfd8d5670e6a8e26b"),
        "Title" : "Skyfall",
        "Director" : "Sam Mnedes",
        "Runtime" : 137,
        "tags" : [
                "007",
                "james_bond"
        ],
        "price" : NumberDecimal("9.99"),
        "Year" : 2010
}
>
```

31

31

findOne() method

- Returns one document that satisfies the specified query criteria on the collection.
  - If multiple documents satisfy the query, this method returns the first document according to the natural order which reflects the order of documents on the disk.
  - Although similar to the find() method, the findOne() method returns a document rather than a cursor.
- Example:

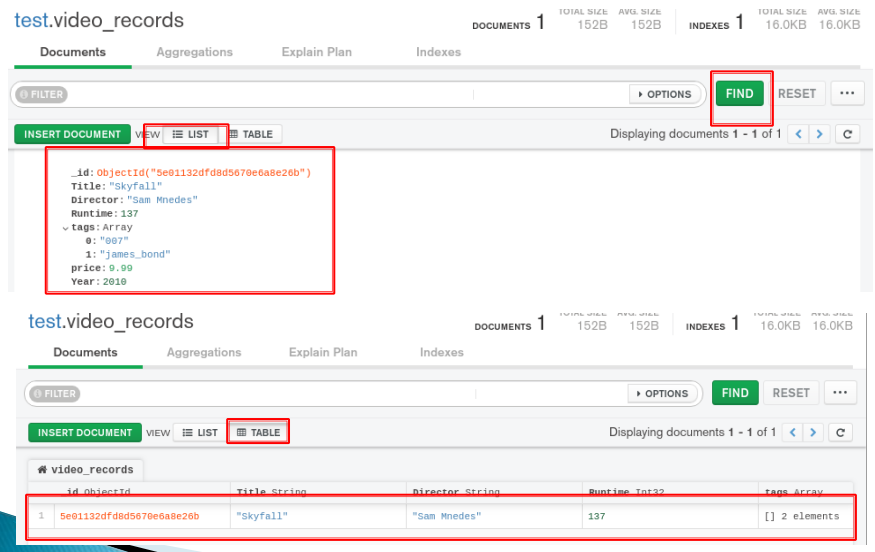
```
>db.video_records.findOne()
```

  - returns a single document from the *video_records* collection

32

32

Query document in MongoDB Compass



33

33

Where Condition

- To query the document on the basis of some condition, you can use following operations:

Operation	Example	RDBMS Equivalent
Equality	`db.video_records.find({"Title":"Skyfall"}).pretty()`	where Title = 'Skyfall'
Less Than	`db.video_records.find({"Runtime":{$lt:250}}).pretty()`	where Runtime < 250
Less Than Equals	`db.video_records.find({"Runtime":{$lte:50}}).pretty()`	where Runtime <= 50
Greater Than	`db.video_records.find({"Runtime":{$gt:50}}).pretty()`	where Runtime > 50
Greater Than Equals	`db.video_records.find({"Runtime":{$gte:50}}).pretty()`	where likes >= 50
Not Equals	`db.video_records.find({"Runtime":{$ne:50}}).pretty()`	where likes != 50

- Also query by type:

```
>db.video_records.find({Title: {$type:10}})
>db.video_records.find({release_date: {$type:9}})
```

34

34

Logical Operators

- AND
  - In the **`find()`** method, if you use **`$and`** operator and pass multiple keys by separating them by ',' then MongoDB treats it as AND condition.

```
>db.video_records.find({$and:[{"Title":"Skyfall"},{"Runtime":137}]}).pretty()
```

- OR
  - To query documents based on the OR condition, you need to use **`$or`** keyword.

```
>db.video_records.find({$or:[{"Title":"Skyfall"},{"Runtime":137}]}).pretty()
```

- AND and OR

```
>db.video_records.find({
          $and: [
                "Runtime": {$gt:100},
                 $or: [{"Title": "Skyfall"} , {"Director":"Sam Mendes"}]
                 ]
                 }).pretty()
```

35

Update Document

- To update the values in the existing document use the **`update()`** method.
- Syntax:

```
>db.COLLECTION_NAME.update(SELECTION_CRITERIA, UPDATED_DATA)
```

- Example:

```
>db.video_records.update({"title":"Skyfall"},{$set:{"title":"Casino Royal"}})
```

- By default, MongoDB will update only a single document.
  - To update multiple documents, you need to set a parameter 'multi' to true.

```
>db.video_records.update({"title":"Skyfall"},{$set:{"title":"Casino Royal"}}
,{multi:true})
```

- The **`save()`** method replaces the existing document with the new document passed in the save() method.

```
>db.COLLECTION_NAME.save({_id:ObjectId(),NEW_DATA})
```

36

36

Delete Document

- To remove a document from the collection use **`remove()`** method, which accepts two parameters:

```
>db.COLLECTION_NAME.remove(DELETION_CRITERIA,justOne)
```

  - deletion criteria − (Optional) deletion criteria according to which the documents will be removed.
  - justOne − (Optional) if set to true or 1, then it will remove only one document.

```
>db.video_records.remove({"Title":"Casino Royal"})
```

- If there are multiple records and you want to delete only the first record, then set **`justOne`** parameter in remove() method.

```
> db.video_records.remove({"Title":"Casino Royal"}, 1)
```

- If you don't specify deletion criteria, then MongoDB will delete all documents from the collection.
  - This is equivalent of SQL's truncate command.

```
>db.video_records.remove({})
>db.video_records.find()
>
```

37

37

Quiz

1. What function do you use to format the query results in mongo shell?
   A. format()
   B. pretty()
   C. print()

2. MongoDB documents are represented as XML.
   A. True
   B. False

3. MongoDB supports query joins between collections.
   A. True
   B. False

38

38

Further reading

- Books:
  - Bradshaw S., Chodorow K., MongoDB: The Definitive Guide, O'Reilly, 2019
  - Banker K., Bakkum P., MongoDB in Action, Manning Publications, 2014
- MongoDB Documentation:
  - CRUD Operations
  - https://docs.mongodb.com/manual/crud/
- Articles:
  - 9 Best MongoDB GUI Tools in 2020:
  - https://www.guru99.com/top-20-mongodb-tools.html

39

39

Essentials

- ✓ Introduced MongoDB
- ✓ Discussed data model
- ✓ Created and manipulated collections
- ✓ Used CRUD commands with documents

40

40