

Node.JS

Lecture 7
MongoDB

Road Map

- ▶ MongoDB
 - ▶ Pros and Cons
 - ▶ Installation
 - ▶ CRUD in MongoDB
 - ▶ Hands On
- 

MongoDb

- ▶ MongoDB is a leading NoSQL database and an open source document database.
- ▶ It is primarily written in C++.
- ▶ Highly scalable and performance oriented database.
- ▶ Document and Collection based.
 - Normally in shape of BSON(Binary JSON)

MongoDb

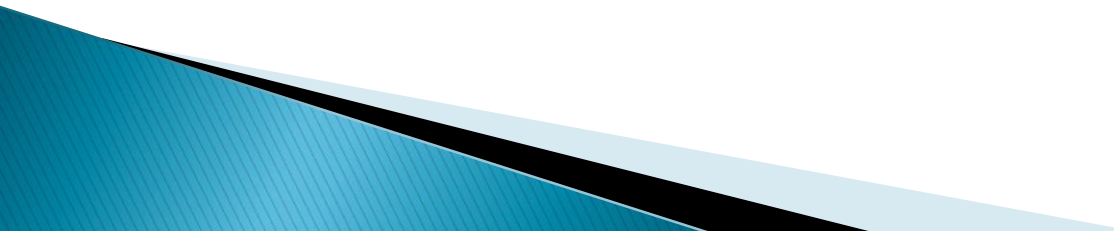
- ▶ **1. Database:** – A single MongoDB server consists of various databases where each database is a physical container of collections.
- 2. Collection:** – A collection in MongoDB is equivalent to a database table and it exists within a single database. It includes a group of MongoDB documents.
- 3. Document:** – Document can be defined as an instance of a MongoDB collection. It includes a set of key value pairs. All documents include a dynamic schema which means documents which comprise of same collection do not need to have same set of fields and structure.

Pros and Cons

▶ Pros:

- MongoDB is a schema-free database. It does not follow a typical schema design as in relational database management system where it shows a number of tables and relationships between these tables. In MongoDB, there is no concept of relationship.
- MongoDB is easy to scale.
- It uses internal memory which enables faster access to the data.
- Structure of object designed is crystal clear.

▶ Cons:

- It provides less flexibility with querying.
 - No support for transactions.
 - No structure of table designed. Logic is associated with document in JSON format.
- 

Why MongoDB

- ▶ **Query Support.** Whereas many NoSQL solutions enables you to access the data only through the keys, MongoDB offers to query regarding the intended fields and specific ranges (*range query*), also it offers you to query with *regular expressions*.
- ▶ **Secondary Index Support.** As well as the querying with respect to intended fields, defining these fields as secondary index provides to access data in a high performance.
- ▶ **Master–Slave Replication Support.** Directing the read and write operations to separate servers, running a slave server as a master server when the master service is inaccessible is a very important positive value undoubtedly.
- ▶ **Sharding Support.** Spreading the large–scaled data across multiple servers is a feature that makes different MongoDB among its kinds.
- ▶ **MapReduce Support.**
- ▶ **Driver Support** for many Software Languages

MongoDb

- ▶ Download mongoDB and install(mongodb.org)
- ▶ Check from command prompt
 - `Mongod -version`
- ▶ Check for free cloud hosting
 - Atlas etc
- ▶ Default: connect to port 27017 on localhost
- ▶ `MongoD.exe //Server File`
- ▶ `Mongo.exe //Client Access File`

Installing Instructions

- ▶ Download community or enterprise edition
- ▶ Install complete
 - Exclude/uncheck compass if internet is slow
 - Create folder “data” in c drive
 - Create folder db in data folder
 - Run cmd. Change directory to c:/program Files/Mongodb/server/4.2/bin
 - Run monogod command in cmd
 - It will start listening at port(remain open)
 - Server is running now
 - Double click mongo.exe in same folder. It is client connector and start running commands.

Basic Commands

- ▶ Db command is to check current db
 - > db
 - test
- ▶ Db.help() will show you commands available
- ▶ Db.stats() The statistics will display the database name, collections and documents associated with them.
- ▶ Use [db] command is used to change the db. It will also create new db if not exist
 - > use presidents
 - switched to db presidents
- ▶ Show dbs to show all databases storage values
 - > show dbs
 - admin (empty)
 - local 0.078GB

Collections

- ▶ Show collections is to show the tables
 - > show collections;
 - system.indexes
- ▶ MongoDB creates collection in place of tables. It uses `db.createCollection` for creating a new collection in database.
- ▶ `db.createCollection(name,options)`

CRUD Commands

- ▶ Adding a record in collection(table) of db, it will create collection if not there before
 - `db.people.insert({name: 'Bill Clinton'})`
- ▶ Search record
 - `db.people.find()` _id field is auto assigned by db
 - `db.people.find({name: 'Bill Clinton'})` more specific query
 - `db.people.find({name: {$regex: '^Bill'}})` using regular expression
 - `db.people.findOne({name: {$regex: '^George'}})` find one record
 - `db.people.find().limit(2);` only two records to search(first two)
- ▶ Updating record
 - `db.people.update({name: 'Bill Clinton'}, {$set: {name: 'William Clinton', terms: 2}})` here first search then change the values. Terms are number of record changes.
 - The `update()` method updates the value in the document while the `save()` method on the other hand replaces the existing document.
 - `db.COLLECTION_NAME.save({_id:ObjectId(),NEW_DATA})`

Basic Commands

- ▶ Deleting Record

- `> db.people.remove({name: {$regex: 'Bush$'}}, {justOne: true})`

- ▶ Deleting Collections

- `> db.people.drop()`

- ▶ Deleting Databases

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

- ▶ It will delete all databases. Better to delete database with name.