NoSQL and MongoDB

An Introduction (Part-II)

Working with MongoDB: Commands

- Creating/Connecting with Database
- Creating a Collection (equivalent to a table)
- View Databases/Collections
- CRUD Operations

Creating a Database/Collection

- Creating a database

  > use <dbname>

  Eg. > use sengdb

- Creating a collection

  > db.createCollection(“collectionname”)

  > db.<collectionname>.insert({“Key”: “Value”})

CRUD Operations

- Data Selection/Manipulation Operations
  - Create [SQL - INSERT] : insert()
  - Read [SQL - SELECT] : find()
  - Update [SQL - UPDATE] : update()
  - Delete [SQL DELETE] : remove()
- **Upsert Operation**
  - **Update, OR Insert** if does not Exist : upsert

CREATE: insert()

Create => Insert

> db.mycollection.insert({“name”: “Tom”})

> db.mycollection.insertOne({“name”: “Tom”})

> db.mycollection.insertMany({“name”: “Tom”},{“name”:”John”})

_id field is insert automatically by mongo if not specified

_id field indentifies each record uniquely and used for indexing

READ: find()

- db.mycollection.find()
  - Top 20 docs, ‘it’ command to display more docs
- db.mycollection.find({“city” : “Mumbai” })
- db.mycollection.find().limit(10)
- db.mycollection.find().limit(2).pretty()
- db.mycollection.find().skip(1)

UPDATE: update()

- db.<collectionname>.update(
      {<criteria>},
      {<doc/partial update>}
)

- _id field can not be updated through update operation

UPDATE: Update()

- db.mycollection.update(

```
        {“_id”:"35004"},
        {$set:{name:“Johnson"}}
        )
```

UPDATE or INSERT: Upsert

- db.mycollection.update(
  {“_id”:"35004"},
  {$set:{name:“Johnson"}},
  {upsert:true}
)

DELETE: remove()

- db.mycollection.remove({})
  - removes all documents from the collection

- db.mycollection.remove({“_id”:"35004"})

- db.mycollection.remove({ “x” : /hello/ })
  - removing documents using regular expressions

JSON Basic Data Types

- Number (Integer or Floating Point)
- String (In double quotes)
- Boolean (true or false)
- Array (In square brackets – [])
- Object (In curly brackets – {})
- NULL

- Keys (_ID) value must always be string and must be written in double quotes

BSON

- Standard to represent JSON in binary format
- Internal format used by mongo database and drivers
- Lightweight hence fast
- Provides Scannability & support for Additional data types
- BSON additional data types –
  - Date
  - Binary Data
  - ObjectID
- Additional Info @ http://bsonspec.org

MongoDB Schema

- Schemaless/Have Dynamic Schema
- Flexible – Agile
- Polymorphic data representation
- NO ‘ALTER TABLE’ required
- Dynamically Typed - Things have types but they are resolved at runtime.

References

- www.nosql-database.org
- university.mongodb.com
- docs.mongodb.org

Mongo Shell Commands Quick Reference

- https://docs.mongodb.com/manual/reference/mongo-shell/