**MongoDB(Pratice)**

**By AfreenSultana A**

**What is NoSQL?**

NoSQL is a non-relational database system designed to store and manage large volumes of unstructured or semi-structured data. It avoids complex joins, supports flexible schemas, and is easy to scale horizontally. It is ideal for distributed data systems.

**What is MongoDB?**

MongoDB is an open-source NoSQL document database that stores data in flexible, JSON-like documents. It is known for high performance, high availability and easy scalability making it suitable for modern applications.

**Data Format – BSON**

MongoDB stores data as BSON (Binary JSON), which is a binary-encoded format of JSON. It allows efficient storage and faster access, and supports more data types than regular JSON.

**MongoDB vs RDBMS**

* RDBMS uses tables, rows (tuples), and columns.
* MongoDB uses collections, documents, and fields.
* RDBMS supports joins between tables, while MongoDB uses embedded documents to achieve similar functionality with better performance.

use studentDB

switched to db studentDB

db.students.insertMany([

{ name: "Afreen", age: 22, course: "Math", status: "A" },

{ name: "Dhivakar", age: 25, course: "Physics", status: "B" },

{ name: "Charlie", age: 30, course: "Chemistry", status: "A" },

{ name: "Zainab", age: 21, course: "Math", status: "C" },

{ name: "Sonu", age: 27, course: "Physics", status: "A" },

{ name: "Monu", age: 29, course: "Biology", status: "B" },

{ name: "Grace", age: 24, course: "Chemistry", status: "A" }

])

{

acknowledged: true,

insertedIds: {

'0': ObjectId('687e5861b5e82a729beeee35'),

'1': ObjectId('687e5861b5e82a729beeee36'),

'2': ObjectId('687e5861b5e82a729beeee37'),

'3': ObjectId('687e5861b5e82a729beeee38'),

'4': ObjectId('687e5861b5e82a729beeee39'),

'5': ObjectId('687e5861b5e82a729beeee3a'),

'6': ObjectId('687e5861b5e82a729beeee3b')

}

}

db.students.find()

{

\_id: ObjectId('687e0854aa80975b9eeec4a9'),

name: 'Afreen',

age: 20,

course: 'Data Engineering'

}

{

\_id: ObjectId('687e0cb1aa80975b9eeec4aa'),

name: 'Dhivakar',

age: 22,

course: 'python'

}

{

\_id: ObjectId('687e207caa80975b9eeec4ac'),

name: 'Zainab',

age: 24,

course: 'AI'

}

{

\_id: ObjectId('687e207caa80975b9eeec4ad'),

name: 'Monu',

age: 32,

course: 'C++'

}

{

\_id: ObjectId('687e5861b5e82a729beeee35'),

name: 'Afreen',

age: 22,

course: 'Math',

status: 'A'

}

{

\_id: ObjectId('687e5861b5e82a729beeee36'),

name: 'Dhivakar',

age: 25,

course: 'Physics',

status: 'B'

}

{

\_id: ObjectId('687e5861b5e82a729beeee37'),

name: 'Charlie',

age: 30,

course: 'Chemistry',

status: 'A'

}

{

\_id: ObjectId('687e5861b5e82a729beeee38'),

name: 'Zainab',

age: 21,

course: 'Math',

status: 'C'

}

{

\_id: ObjectId('687e5861b5e82a729beeee39'),

name: 'Sonu',

age: 27,

course: 'Physics',

status: 'A'

}

{

\_id: ObjectId('687e5861b5e82a729beeee3a'),

name: 'Monu',

age: 29,

course: 'Biology',

status: 'B'

}

{

\_id: ObjectId('687e5861b5e82a729beeee3b'),

name: 'Grace',

age: 24,

course: 'Chemistry',

status: 'A'

}

db.students.find({},{name:1, age:1,\_id:0})

{

name: 'Afreen',

age: 20

}

{

name: 'Dhivakar',

age: 22

}

{

name: 'Zainab',

age: 24

}

{

name: 'Monu',

age: 32

}

{

name: 'Afreen',

age: 22

}

{

name: 'Dhivakar',

age: 25

}

{

name: 'Charlie',

age: 30

}

{

name: 'Zainab',

age: 21

}

{

name: 'Sonu',

age: 27

}

{

name: 'Monu',

age: 29

}

{

name: 'Grace',

age: 24

}

db.students.find({age:{$gt:25}},{name:1,age:1,course:1,\_id:0})

{

name: 'Monu',

age: 32,

course: 'C++'

}

{

name: 'Charlie',

age: 30,

course: 'Chemistry'

}

{

name: 'Sonu',

age: 27,

course: 'Physics'

}

{

name: 'Monu',

age: 29,

course: 'Biology'

}

db.students.find({age:{$gte:23,$lt:30}})

{

\_id: ObjectId('687e207caa80975b9eeec4ac'),

name: 'Zainab',

age: 24,

course: 'AI'

}

{

\_id: ObjectId('687e5861b5e82a729beeee36'),

name: 'Dhivakar',

age: 25,

course: 'Physics',

status: 'B'

}

{

\_id: ObjectId('687e5861b5e82a729beeee39'),

name: 'Sonu',

age: 27,

course: 'Physics',

status: 'A'

}

{

\_id: ObjectId('687e5861b5e82a729beeee3a'),

name: 'Monu',

age: 29,

course: 'Biology',

status: 'B'

}

{

\_id: ObjectId('687e5861b5e82a729beeee3b'),

name: 'Grace',

age: 24,

course: 'Chemistry',

status: 'A'

}

db.students.find({name:/^A/})

{

\_id: ObjectId('687e0854aa80975b9eeec4a9'),

name: 'Afreen',

age: 20,

course: 'Data Engineering'

}

{

\_id: ObjectId('687e5861b5e82a729beeee35'),

name: 'Afreen',

age: 22,

course: 'Math',

status: 'A'

}

db.students.deleteMany({course:{$in:["AI","Java","C++","Data Engineering"]}})

{

acknowledged: true,

deletedCount: 3

}

db.students.find({name:/u/})

{

\_id: ObjectId('687e5861b5e82a729beeee39'),

name: 'Sonu',

age: 27,

course: 'Physics',

status: 'A'

}

{

\_id: ObjectId('687e5861b5e82a729beeee3a'),

name: 'Monu',

age: 29,

course: 'Biology',

status: 'B'

}

db.students.countDocuments()

8

db.students.countDocuments({age:{$gt:25}})

3

db.students.distinct("status")

[ 'A', 'B', 'C' ]

db.students.distinct("course")

[ 'Biology', 'Chemistry', 'Math', 'Physics', 'python' ]

db.studemts.findone()

[TypeError:](file:///C:\Users\parve\AppData\Local\MongoDBCompass\app-1.46.6\resources\app.asar\build\index.html) db.studemts.findone is not a function

db.students.findOne()

{

\_id: ObjectId('687e0cb1aa80975b9eeec4aa'),

name: 'Dhivakar',

age: 22,

course: 'python'

}

db.students.find().limit(5).skip(2)

{

\_id: ObjectId('687e5861b5e82a729beeee36'),

name: 'Dhivakar',

age: 25,

course: 'Physics',

status: 'B'

}

{

\_id: ObjectId('687e5861b5e82a729beeee37'),

name: 'Charlie',

age: 30,

course: 'Chemistry',

status: 'A'

}

{

\_id: ObjectId('687e5861b5e82a729beeee38'),

name: 'Zainab',

age: 21,

course: 'Math',

status: 'C'

}

{

\_id: ObjectId('687e5861b5e82a729beeee39'),

name: 'Sonu',

age: 27,

course: 'Physics',

status: 'A'

}

{

\_id: ObjectId('687e5861b5e82a729beeee3a'),

name: 'Monu',

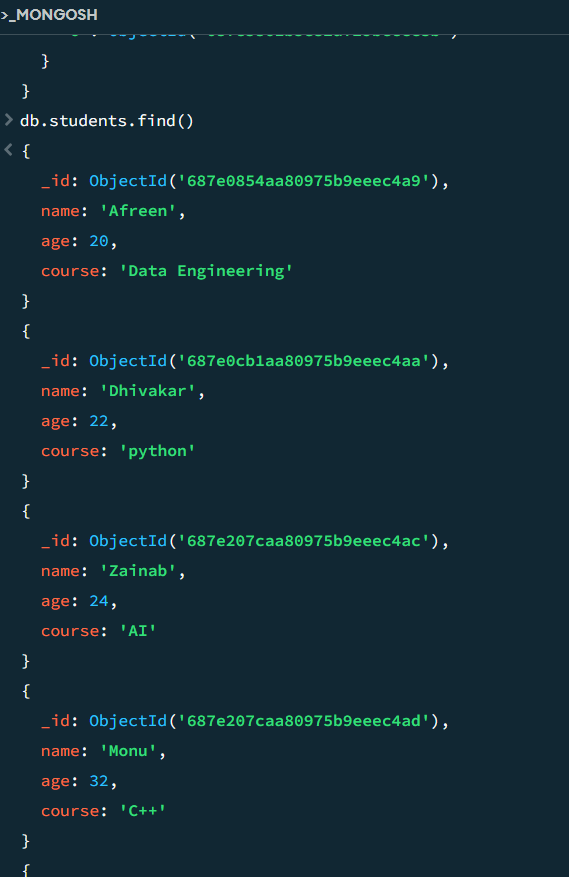
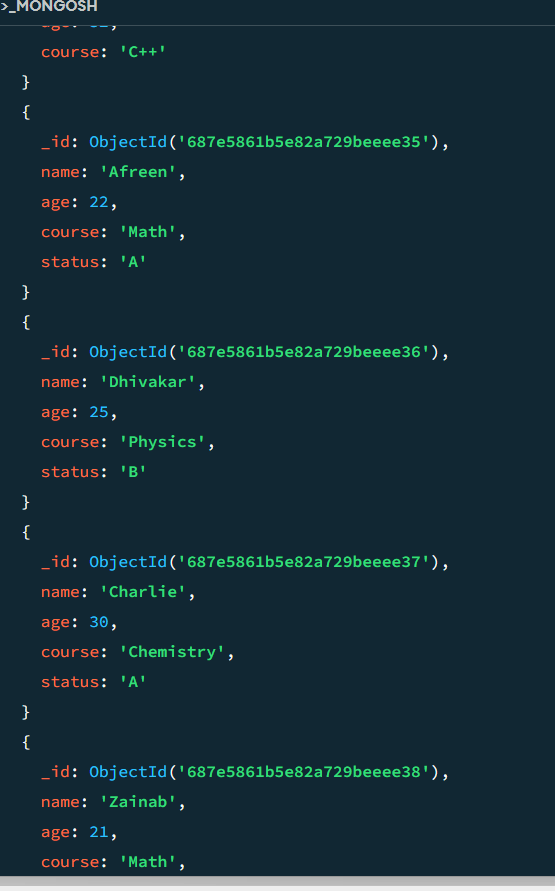
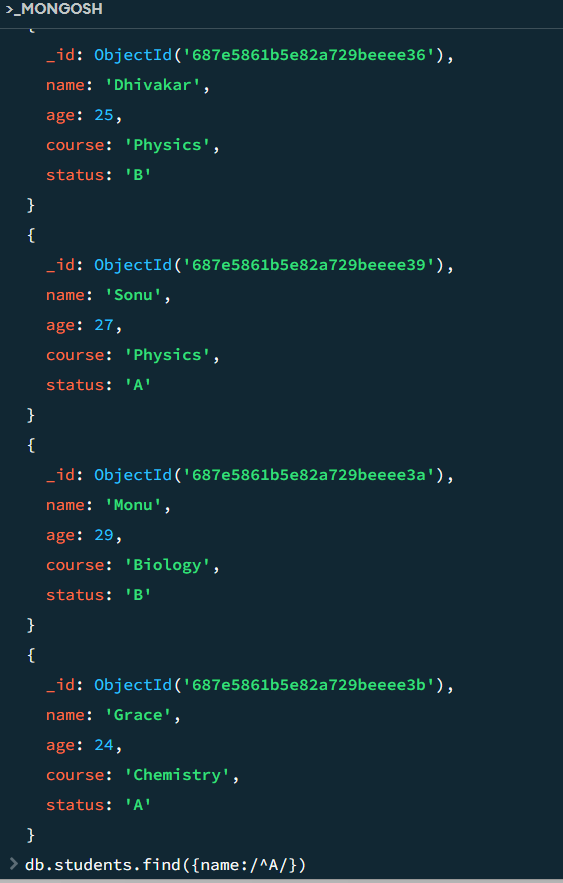
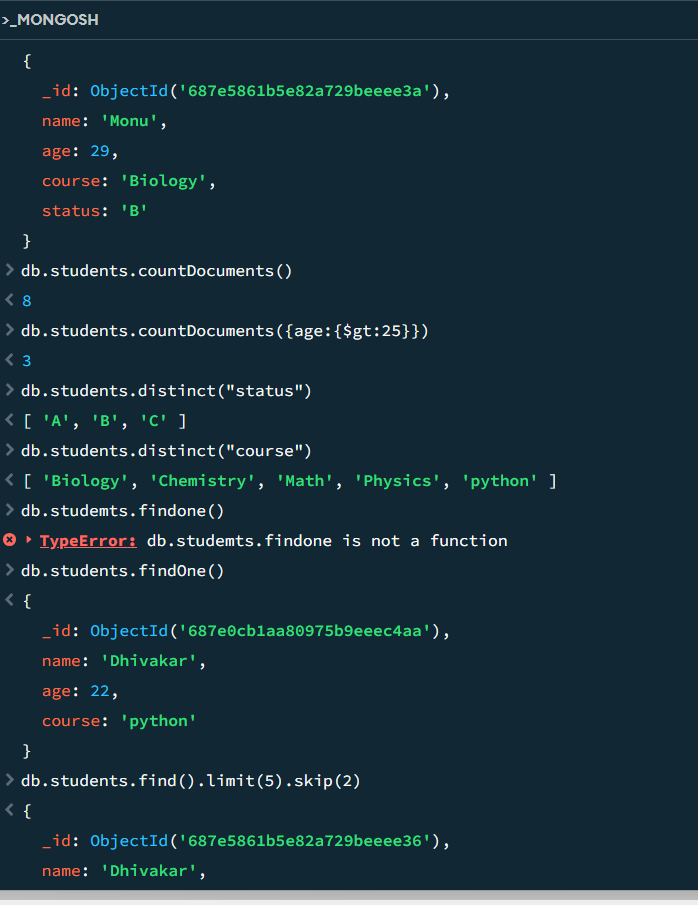
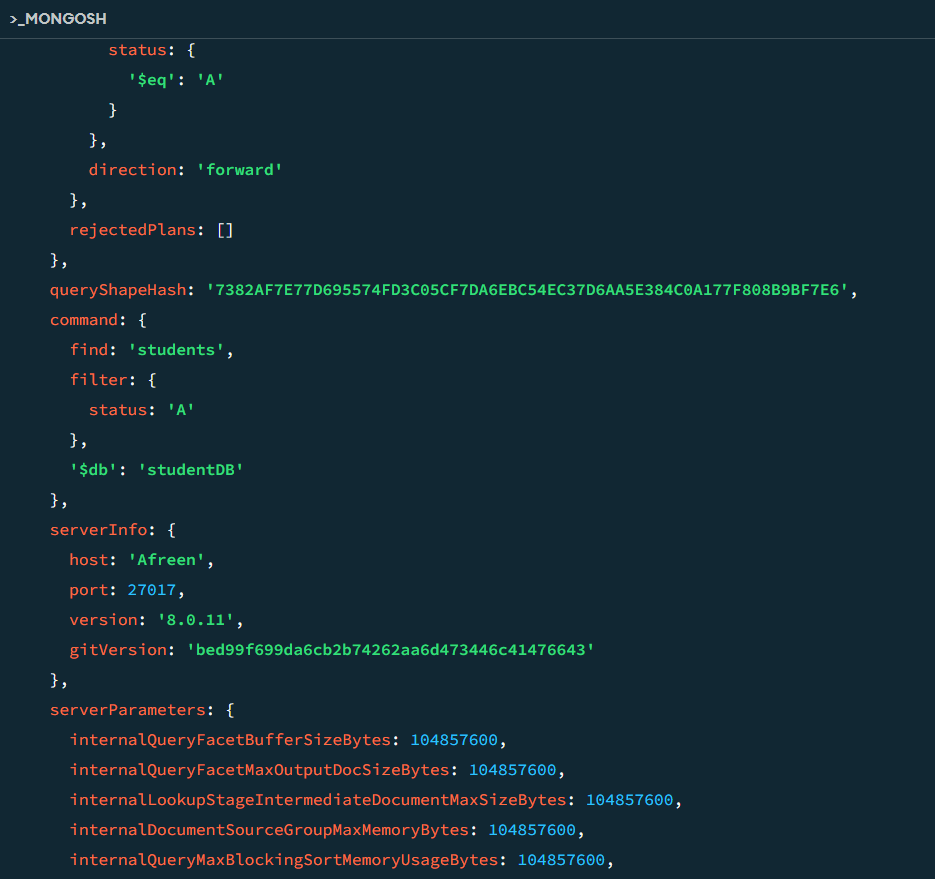
age: 29,

course: 'Biology',

status: 'B'

}

db.students.find({status:"A"}).explain()

(screenshots are in reverse order) ****