MongoDB

MongoDB Atlas → Service provider

MongoDB — Document-oriented NoSQL database Management System

Open-Source

Document Database

json

Flexibility, Scalability, performance.

Unstructured or Semi-Structured Data

10gen Company created MongoDB

↓

Now MongoDB

Eliot Horowitz
Dwight Merriman
in 2007

first version => 2009

HUMONGOUS => very large

MONGO నుంచి పుట్టింది

SQL => Structure Query Language	MongoDB (NoSQL)
Relational Database Tables	NoSQL – non relational database
Structured Tables fixed rows & colum.	Flexibility in data storage Unstructured / Semi Structured
Suitable for appliction, with well-defined schema fixed data struc	Dynamic Evolving
e-commerce HR Management	CMS, Social Media, Gaming
Ex MySQL, PostgreSQL, Oracle	Ex MongoDB, Cassandra Redis

SQL — Fixed predefined schema, Fixed columns

Row

Table : Students

student_id	First Name	last_nm	age	gra
1	Yaswanth	V	16	11
2	Kishori	K	16	12

Table : Subjects

sub_id	sub_name
1	Mathematics
2	Computer

Table : Grades

student_id	subject_id	marks
1	1	99
2	1	100

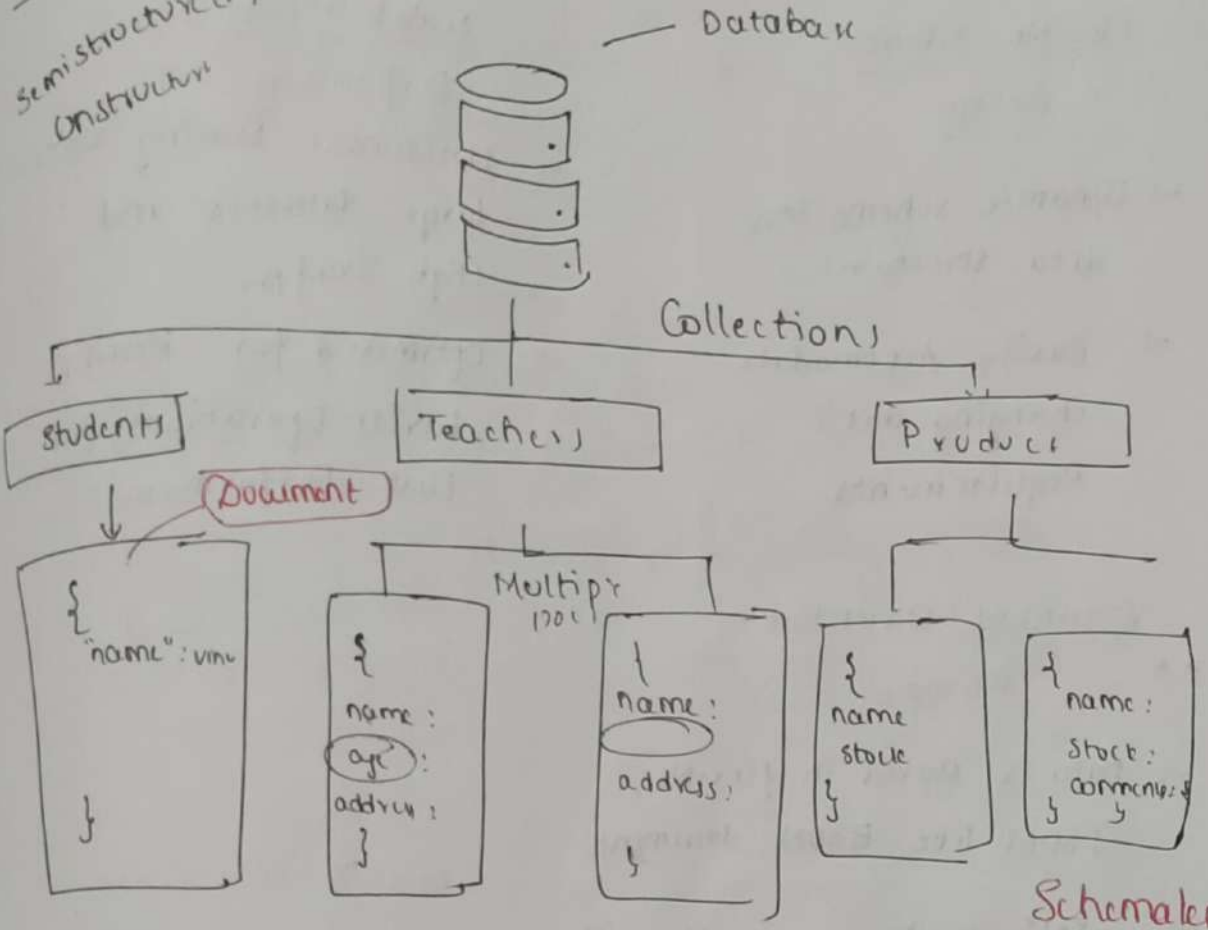
NoSQL in form of documents

Embedded Documents

```
[
  {
    "_id" : "1",
    ...
  },
  {
    "_id" : "2",
    "extra" : 'sport_captain',
    ...
  }
]
```

MongoDB Terminologies

semistructured / unstructured



Schemaless

ಜೊ ಹೈ ಸೊ ಹೈ

ಏಂಡೆ ಏನ್ನಿಲ್ಲ ಅರ್ಥಗರ್ಭಿತ ಹೈ

Key Features of MongoDB

Flexible Schema Design

=> Dynamic schema-less data structures

=> Easily Accomodate changing data Requirements

Scalability & Performance

=> Horizontal scaling supports large datasets and High Traffic

=> Optimised for Read & Write Operations for Fast Performance

Document Oriented Storage

**

=> Data is stored in flexible, JSON like BSON documents

=> Self contained units with rich data types and nested arrays

Dynamic Queries

=> Rich Query language with support for complex queries

=> Utilize Indexes to speed up query execution

* Aggregation Framework: *

=> Perform advanced data transformations and analysis

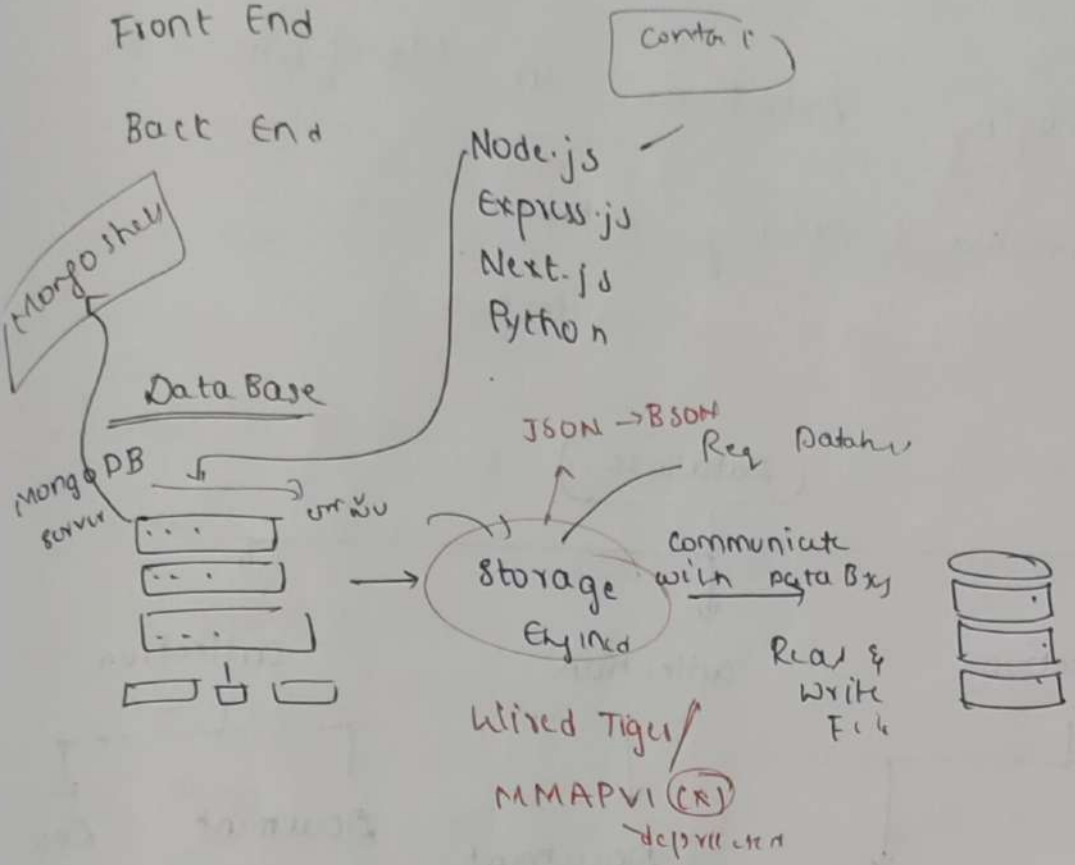
=> Process data using multiple pipeline stages

OpenSource and Community

MongoDB is open-source with a vibrant community

Regular updates, improvements and support

How MongoDB works.



JSON Vs BSON

In MongoDB —— we write in JSON format

BTS => data is stored in BSON
Binary JSON format

By BSON

Achieve High Read & Write Speeds

★ Reduced storage Req

Improved Data Manipulation capabilities

JSON — easy to Read & Write

```
{ "name": "Thapa",
  "age": 29,
  "isStudent": false,
  "scores": [92, 101],
  "address": {
      "city": "pokhara"
  }
}
```

BSON

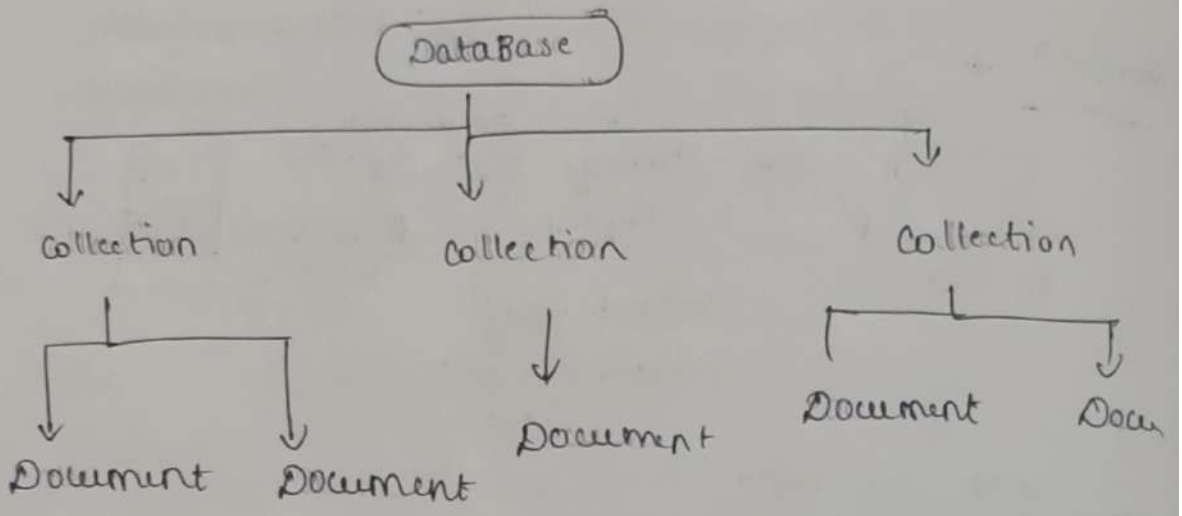
```
\x1e\x00\x00\x00
age\x00
isStudent\x00
```

Installing mongoDB

Managing DataBases in MongoDB

① Creating / Deleting Database

② " " Collection



① show dbs

Create New Database: use <database-name>;

Delete: db.dropDatabase();

① show collections

② db.createCollection('<collection-name>');

③ db.<collection-name>.drop();

cmd

mongo sh

=> cls

show dbs / show databases (s) no error

Error

use students

show dbs

students ఉంటుంది *

use admin

show dbs

Database create చేసిన తర్వాత ఒక collection ఉంటే ఉంటుంది. లేదు అప్పుడు show అవ్వదు

show collections

students > db.createCollection('data')

collection name

show collections

data

db.data.drop() => Delete collection.

db.dropDatabase() => Delete Database.

CRUD → Read
→ Delete
→ Update
Create

Insert Operation in MongoDB

① Inserting Documents in MongoDB

One Document

db.<collection-name>.insertOne({

fieldd1 : value1 ,,

field2 : value2 ,,

.....

});

db.<collection-name>.insertMany(

array of Objects [{field1 : value1, field2 : value2, ...},

{field1 : value1, field2 : value2, ...},

// ....

]);

Document

Direct तो ~~collection~~ create / insert करेंगे collection अपने आप बन जाएगा

use student

db.data.insertOne({ name : 'Radha',

age : 29

})

{ acknowledged : true,

insertedId : ObjectId("64d5c719...")

}

db.data.insertMany(

[{

},

{

},

{

}

]);

Read

db.data.find( )

Doc देखने के लिए

When to Use Quotes and When not to?

① Special char

fieldname ⇒ contains special character or spaces or starts with Number

② Reserved Words

fieldname is a Reserved keyword in Mongodb

Quotes distinguish it from the reserved keyword

```
db.data.insertOne({
    'name' : 'yaswanth',
    age    : 29,
    course name : "CS"
})
```

field, value, with quotes

Error

'course name' : "CS"

Ordered & UnOrdered Inserts

Bulk write

⇒ When executing operations Ordered & UnOrdered determine Batch Behaviour

① Ordered Inserts:

Default Ordered

Stops at 1st Error

```
students > db.data.find()

db.data.insertMany([
    { _id : ObjectId("                "), 'name':ajun, age:30},
    { 'name' : 'arjun', age : 30 },
```

inserted doc = 1

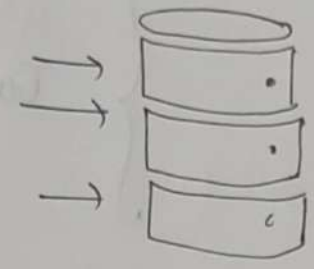
Skipped

$Ordered Insert

{Correct Doc1}

{Wrong Doc2}

{Correct Doc3}



{Correct Doc1}

बीच में गलती हो गई

insertedCount = 1

Only correct Doc before

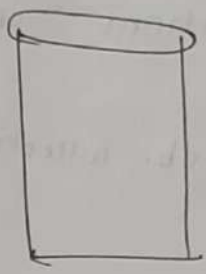
$Ordered = false

{Correct Doc1}

{Wrong Doc2}

{Correct Doc3}

{Correct Doc4}



{Correct Doc1}

{Correct Doc3}

{Correct Doc4}

db.<collection-name>.insertMany([doc1, doc2], {ordered: false});

insertedCount = 2

E11000 → Duplicate Keys

Case Sensitivity in MongoDB

Collections
Field
} Case sensitive

⇒ db.data.find().limit(2)

db.data.find().limit(2)
} Different

Read Operations in MongoDB

① find db.collection_name.find({key:value})

.findOne({key:value}) — first 1 document

db.data.find({'name':' '})

Importing JSON in MongoDB

Data Array of objects into DB.

mongoimport ____.json -d <database> -c <collection>

mongoimport .json -d <shop> -c <products>

mongoimport .json -d <shop> -c products --jsonArray

[{
},
{
},
{}]

Array of objects

{ " " : { },

" " : { " " ,

" " : " "

}, {"

}

c:\Users\Thapa > mongoimport

Sales.json ——— import =)

-- json Array

mongoexport -d shop -c sales -o "E:\\Sales.json"

type 'it' for more

└ iterator

import < 16 MB

Comparison

$eq

$ne

$gt

$gte

$lt

$lte

$in

$nin

db.collectionName.find ({ "field" : { $operator : value } })

db.products.find ({ 'price' : { $eq : 699 } })

: { $in : [2499 , 1099 , 899] }

db.products.find ({ 'price' : { $eq : 699 } })

'price' : { $in : [699 , 129 , 69] }

~~Or~~ price ಇದ್ದರೆ

Display ಆಗುತ್ತದೆ.

Introduction to Cursors 1:27:10

① Automatic Batching Default batch size 101 documents

Cursor Methods

count()

limit()

skip()

sort()

db.products.find ({ 'price' : { $gt : 250 } }).limit(5).skip(2)

.limit(3).sort({ price : 1 })

field Name

1 Ascending

-1 Descending

$ Logical Operators

$and

$or

$not

$nor

$and, $or, $nor

{ $and : [{cond1}, {cond2}, ...] }

{ field : { $not : { operator : value } } }

price > 100 name = Diamond Ring

db.products.find({ $and : [{'price' : {$gt : 100}},
{'name' : 'DiamondRing'}] }

ఇలాంటి Output

.find ({'price' : {$gt : 100}, 'name' : 'DiamondRing'})

↑

implicit and