MongodB. mongo means huge

JSON (Javascript object Notation)

- Document based Stored data.
- NO SQL

MSQL relational database	MongoDB non-relational database.
database	database
table	collection (collection of documents)
row (record)	document (BSON)
column (field)	field
CRUD operation	Big data & real time web appn

MDB e.g. cluster → Database → Collection → doc, field

MySQL schema → database → table → row, column.

In folder .\mongod ① mongos mongod

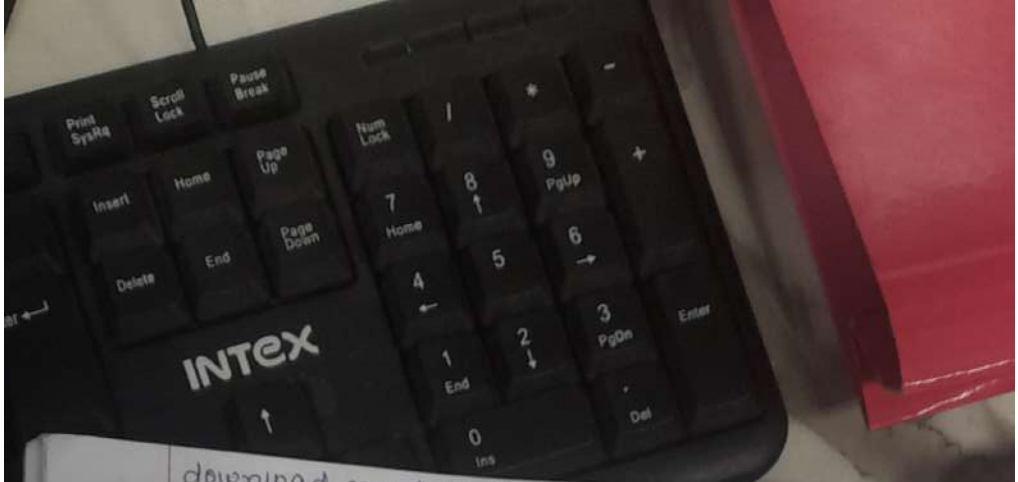
to change datadirectory
.\mongod --help — ②
go to dbpath arg - Directory for data files
execute .\mongod --dbpath C:\data\dB1

compass → to visualize data. — GUI

- Create database - Name → collection name
- you must have to create one atleast 1 collection.

- we dont need schema here. (Structure
If my one document has some columns then there is no need that the other document should have Same column name.

- mongodB is a document oriented NoSQL database system.
- Stores JSON document structure. (BSON)
- built for modern application (High scalable)



download mongodb & mongoshell.

27017

JSON (TEXT)	BSON (Binary)
• Standard file format	• Binary file format
• less fast	faster
• less space	more space is consumed
• Transmission of data	storage of data
• No such technique	Faster encoding & decoding techn
• key-value pair only used for transmission of data	• lightweight, fast & traversable

MEAN stack, MEARN stack developer uses mongodb

Difference bet'n mongod & mango

	"mango" is the command line-shell that connects to a specific instance of mango	"mongod" is the "mango Daemon" its basically the host process for the database
mongo is for running query & mongod is accepting that query & connecting to dB	"mango" works to give command to the "mongod"	"mongod" is the part that manages the database with the hardware.

Employee
{
"name" : "Amit",
"age" : 27, -- fresher
"city" : "Noida"
"identify" : {
"adhar" : 44422222,
"pan" : "TUFEU59050"
}
}

New employee added
{ "name" : "Nitin",
"age" : 29,
"city" : "Nagpur",
"identity" : {
"adhar" : 545767987654,
"pan" : "TUFEU9060"
}
"previous" : [
"amazon", "TCS"]
}

we will change only document not collection
here we don't have schema thatswhy the flexibility of storage data is increasing.
we can add as many feilds in the data.

- It has relations
- data is stored together

company behind mongodB → mongodB's office in Palo Alto, California.

To create database in mongodB server.
run command on shell →
use new_database_Name

① Create collection → table
db. collection_name. insertOne ({name:"chetana"
; age:27} age:1})
↓
Table

② select or read data

db. collection_name. find()

Here we will see one additional column which is called unique identifier which identifies each data in the document.

Nested document.
Suppose i have one ~~collection~~ document which has feilds name, age, city & i have 100 of students data. now i want to add their ~~adhb~~ adhar card and pan card data. so will update the ~~collection~~ document.

To update the document →

```
db. ~~studenes~~ students. updateOne ({name:"chetana"}
                 {$ set: {idcards:
                 {hasPancard:"No",
                 hasAdharcard:"Yes"}}})
```

now we can find the data using findOne()
db. students. findOne({name: 'chetana"})

100 levels of nested is possible in mongodB
every document has limit in size. so we can save data upto 16mb of size

Now i want to update all the documents.

```
db.students.updateMany({},
    {$set:{hobbies:['Painting','Reading']}});
```

Searching particular document:

```
db.students.find({hobbies:"cooking"})
db.students.find({hobbies:"cooking"}).count()
```

Now i want the students which has Adharcard.

```
db.Students.find({"idcard.hasAdharCard":"yes"})
                  ⇓        ⇓
                 key.     key
```

CRUD Operations

Create
Read
Update
Delete

CReate

* If you want to insert one document.
  insertOne(data, options)
* for many documents
  insertMany(data, options)

to check collections (Table) → show collections

Page No.
Date:
YOUVA

Read :

find (filter, options) → multiple documents
findOne (filter, options) → single document.

Update :

updateOne (filter, data, options)
updateMany (filter, data, options)
replaceOne (filter, data, options)

delete :

deleteOne (filter, options)
deleteMany (filter, options)