## PAGE:1

# Cred-With-MongoDB-With-CLI

**MongoDB Cheat Sheet**

**Show Databases** 🗂️

show dbs

*Prints the current database:*

db

**Switch Database** 🔄

use <database\_name>

**Show Collections** 📚

bashCopy code

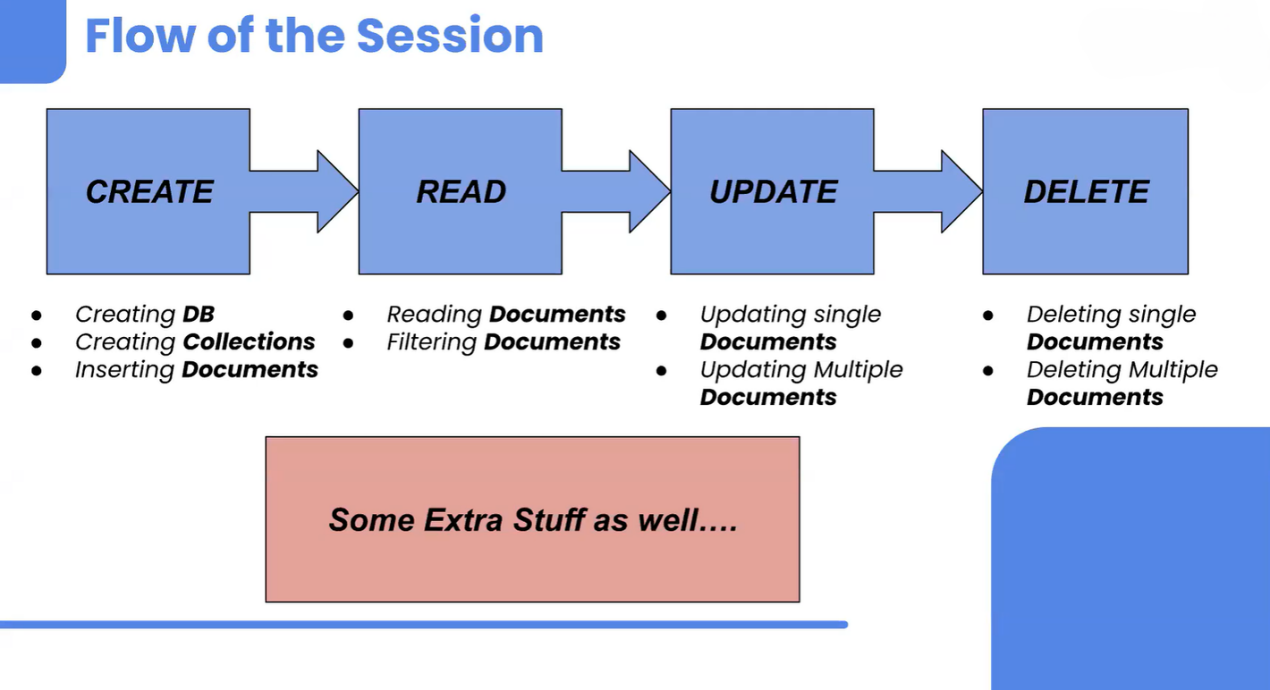
show collections

//delete

db.coll.drop() // removes the collection and its index definitions

db.dropDatabase()

## PAGE:2



# Dummy data:

[

{ name: "Amit Sharma", org: "Tech Innovations", city: "Delhi" },

{ name: "Sita Patel", org: "Innovate Solutions", city: "Mumbai" },

{ name: "Ravi Kumar", org: "Data Technologies", city: "Bangalore" },

{ name: "Anita Desai", org: "Creative Minds", city: "Hyderabad" },

{ name: "Rajesh Singh", org: "NextGen Technologies", city: "Chennai" },

{ name: "Priya Gupta", org: "Alpha Innovations", city: "Kolkata" },

{ name: "Arun Nair", org: "Beta Tech", city: "Pune" },

{ name: "Neha Mehta", org: "Gamma Solutions", city: "Ahmedabad" },

{ name: "Vikram Reddy", org: "Delta Technologies", city: "Jaipur" },

{ name: "Kiran Agarwal", org: "Epsilon Systems", city: "Noida" },

{ name: "Divya Kapoor", org: "Zeta Enterprises", city: "Gurgaon" },

{ name: "Manoj Yadav", org: "Sigma Innovations", city: "Lucknow" },

{ name: "Pooja Sharma", org: "Theta Solutions", city: "Indore" },

{ name: "Sanjay Verma", org: "Omicron Tech", city: "Coimbatore" },

{ name: "Ritu Joshi", org: "Lambda Enterprises", city: "Chandigarh" },

{ name: "Rakesh Kumar", org: "Kappa Solutions", city: "Patna" },

{ name: "Nisha Bhatia", org: "Iota Systems", city: "Bhubaneswar" },

## PAGE:3

{ name: "Kartik Mehta", org: "Eta Innovations", city: "Kanpur" },

{ name: "Sonal Jain", org: "Zeta Solutions", city: "Ranchi" },

{ name: "Vivek Joshi", org: "Omega Tech", city: "Dehradun" }]

# Create:

db.coll.insertOne({name: "Max"})

db.coll.insertMany([{name: "Max"}, {name:"Alex"}]) // ordered bulk inse

db.createCollection("<collection\_name>")

# Read:

db.coll.findOne() // returns a single document

db.coll.find()

db.coll.find({name: "Max", age: 32})

# Update:

db.coll.updateOne({"\_id": 1}, {$set: {"year": 2016, name: "Max"}}

# Delete:

JavaScript

Copy

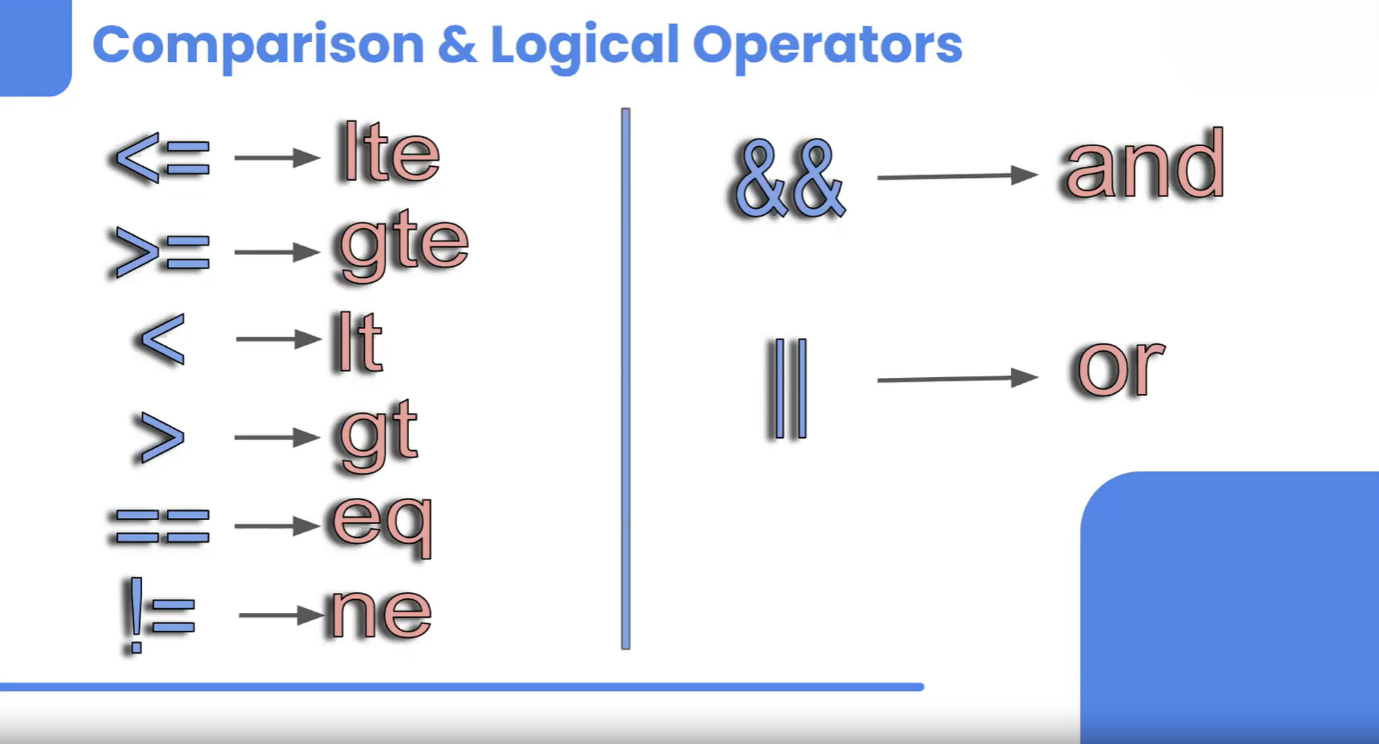
db.coll.deleteOne({name: "Max"})

db.coll.deleteMany({name: "Max"})

db.coll.deleteMany({}) // WARNING! Deletes all the docs but not the collection itself and its index definitions

db.coll.findOneAndDelete({"name": "Max"})

## PAGE:4



## Filtering Data: Advanced Techniques 🔍

**Comparison Operators** ⚖️

Comparison operators allow us to filter data based on various conditions. Here’s how we refer to them in MongoDB:

* **Less than or equal to (≤)**

*MongoDB:* lte

*Example:*

bashCopy code

db.heroes.find({health: {$lte: 50}}).pretty() # Finds heroes with health ≤ 50

* **Less than (<)**

*MongoDB:* lt

*Example:*

bashCopy code

db.heroes.find({health: {$lt: 50}}).pretty() # Finds heroes with health < 50

​

## PAGE:5

**Greater than or equal to (≥)**

*MongoDB:* gte

*Example:*

Shell

Copy

bashCopy code

db.heroes.find({health: {$gte: 50}}).pretty() # Finds heroes with health ≥ 50

​

**Greater than (>)**

*MongoDB:* gt

*Example:*

Shell

Copy

bashCopy code

db.heroes.find({health: {$gt: 50}}).pretty() # Finds heroes with health > 50

​

**Equal to (==)**

*MongoDB:* eq (optional, can use direct equality)

*Example:*

Shell

Copy

bashCopy code

db.heroes.find({health: 86}) # Finds heroes with health == 86

​

**Not equal to (!=)**

*MongoDB:* ne

*Example:*

Shell

Copy

## PAGE:6

bashCopy code

db.heroes.find({health: {$ne: 86}}) # Finds all heroes except those with health == 86

## Notes:

For equality checks, you can directly use the value without eq.

lte, gte, gt, lt can be used to filter data effectively.

**Logical Operators** 🔗​

Logical operators help in combining multiple conditions for more complex queries. We will use the users collection for examples:

**AND**

*Example:*

Shell

Copy

bashCopy code

db.users.find({$and: [{org: ""}, {country: "India"}]})

**OR**

Example:

bashCopy code

db.users.find({$or: [{org: ""}, {country: "India"}]})

​

**💡 Note:**

To add {country: "India"} to all documents in the users collection:

bashCopy code

db.users.updateMany({}, {$set: {country: "India"}})

Range QueriesExample:

bashCopy code

db.heroes.find({$and: [{health: {$gt: 40}}, {health: {$lt: 60}}]}).pretty() # Finds heroes with health between 40 and 60

*💡 Note:*

## PAGE:7

You can combine multiple operators and conditions based on your requirements to refine your queries.

# Dummy data

[

{

"name": "IronMan",

"powers": ["robot", "money"],

"health": 33,

"villains": [

{ "name": "Mandarin", "health": 50 },

{ "name": "Titanium Man", "health": 54 }

],

"metadata": { "favouriteColor": "red", "age": 13 }

},

{

"name": "Batman",

"powers": ["intelligence", "money"],

"health": 40,

"villains": [

{ "name": "TwoFace", "health": 20 },

{ "name": "Redhood", "health": 65 }

],

"metadata": { "favouriteColor": "blue", "age": 44 }

},

{

"name": "Spider-Man",

"powers": ["intelligence"],

"health": 60,

"villains": [

## PAGE:8

{ "name": "Venom", "health": 65 },

{ "name": "Dr. Octavio", "health": 76 }

],

"metadata": { "favouriteColor": "red", "age": 44 }

},

{

"name": "Thor",

"powers": ["god", "magic"],

"health": 57,

"villains": [

{ "name": "Surtur", "health": 50 },

{ "name": "Hela", "health": 87 }

],

"metadata": { "favouriteColor": "yello", "age": 313 }

},

{

"name": "Dr. Strange",

"powers": ["magic"],

"health": 86,

"villains": [

{ "name": "Hela", "health": 87 },

{ "name": "Dormamu", "health": 100 }

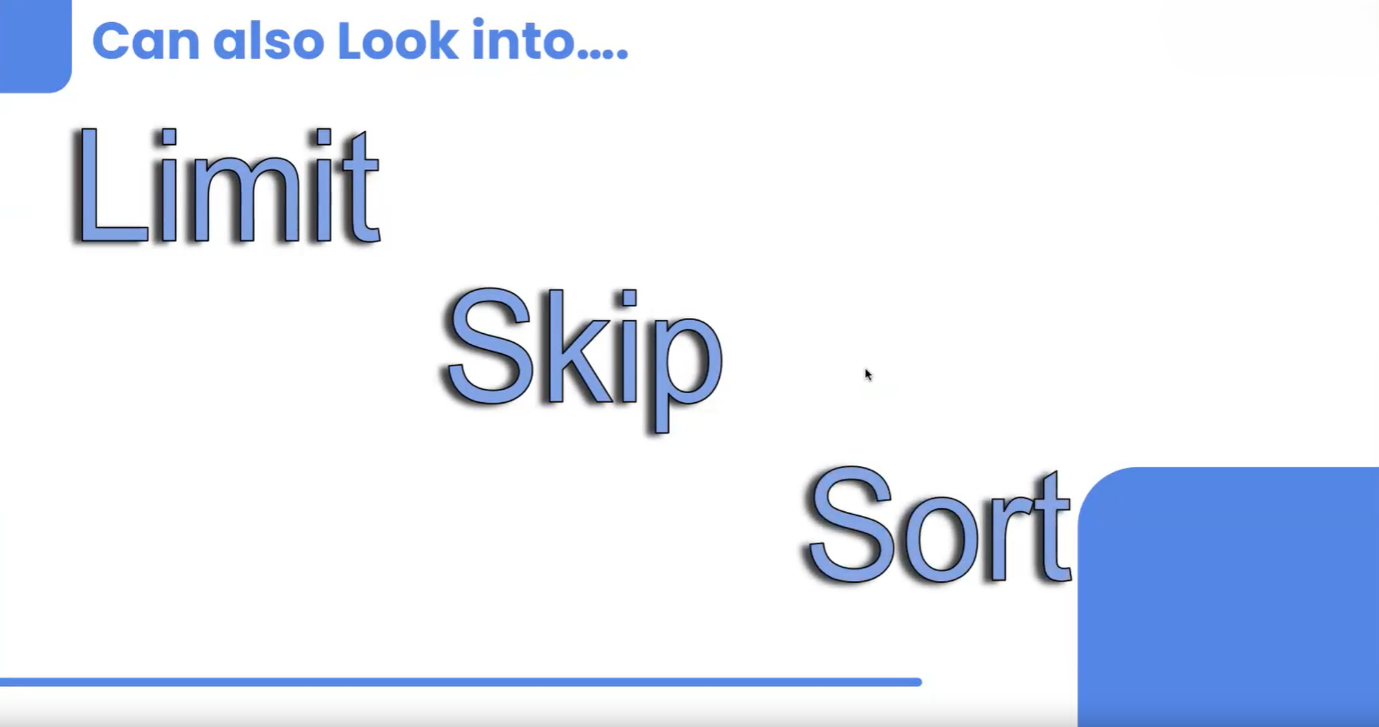
],

"metadata": { "favouriteColor": "orange", "age": 44 }

}

]

## PAGE:9



Limit ⏳​

*Notes:*

We can limit the result; criteria can be anything.

*Example:*

Shell

Copy

bashCopy code

db.heroes.find().limit(2).pretty() # Returns the first two documents

​

Shell

Copy

bashCopy code

db.heroes.find({org:""}).limit(2).pretty()

Skip **🚀​**

*Notes:*

We can use it to skip documents.

*Example:*

## PAGE:10

Shell

Copy

bashCopy code

db.users.find({country:"India"}).skip(2).pretty() # Skips the first 2 documents

​

## We can also combine skip() and limit():

Shell

Copy

bashCopy code

db.users.find({country:"India"}).skip(1).limit(2).pretty() # Skips the first one and limits the result to two documents

💡 Question: Where exactly can we use this in real life?

Answer: This can be used in pagination.

Sort 🔢​

Notes:

It sorts the documents in ascending or descending order.

Example:

bashCopy code

db.heroes.find().sort({health:1}).pretty() # Sorts in ascending order

For descending order, use 1:

bashCopy code

db.heroes.find().sort({health:-1}).pretty() # Sorts in descending order