The Ultimate MongoDB Cheat Sheet

-Gaurav Pandey

Motivation Tree

www.motivationtree.com

MongoDB Cheat Sheet

Motivation Tree

MongoDB Help

```
db.help
```

Show All Databases

```
show dbs
```

Drop database

```
db.dropDatabase()
```

To check collections list

```
Show collections
```

Create collection

```
db.createCollection(name)
```

Insert document in collection

```
db.COLLECTION_NAME.insert(document)
```

Get collection document

```
db.COLLECTION_NAME.find()
```

Update document

```
db.COLLECTION_NAME.update
(SELECTION_CRITERIA, UPDATED_DATA)
```

Delete document

```
db.COLLECTION_NAME.remove(DELLETION
_CRITTERIA)
```

Save document

```
db.COLLECTION_NAME.save({
_id:ObjectId(),NEW_DATA})
```

Get All Rows Formatted

```
db.posts.find().pretty()
```

Find Rows

```
db.posts.find({ category: 'React' })
```

Sort Rows

```
db.user.find().sort({ marks: 1 }).  - # asc
db.user.find().sort({ marks: -1 }). - # desc
```

Limit Rows

```
db.user.find().limit(10).pretty()
```

Foreach

```
db.user.find().forEach(function(doc) {
  print("story: " + doc.title) })
```

Find One Row

```
db.posts.findOne({ category: 'React' })
```

Motivation Tree

Find Specific Fields

```
db.user.find({ title: 'Post One' }, {
  title: 1,
  author: 1
})
```

Update Row

```
db.user.update({ name: 'grv' },
{
  name: 'srv',
  body: 'text',
},{
  upsert: true
})
```

To check collections list

```
Show collections
```

Create collection

```
db.createCollection(name)
```

Insert document in collection

```
db.COLLECTION_NAME.insert(document)
```

Get collection document

```
db.COLLECTION_NAME.find()
```

Update document

```
db.COLLECTION_NAME.update
(SELECTION_CRITERIA, UPDATED_DATA)
```

Delete document

```
db.COLLECTION_NAME.save({
_id:ObjectId(),NEW_DATA})
```

Get All Rows Formatted

```
db.posts.find().pretty()
```

Find Rows

```
db.posts.find({ category: 'React' })
```

Sort Rows

```
db.user.find().sort({ marks: 1 }).  - # asc
db.user.find().sort({ marks: -1 }). - # desc
```

Limit Rows

```
db.user.find().limit(10).pretty()
```

Foreach

```
db.user.find().forEach(function(doc) {
  print("story: " + doc.title) })
```

Find One Row

```
db.posts.findOne({ category: 'React' })
```

Create

- db.coll.insertOne({name: "Max"})
- db.coll.insert([{name: "Max"}, {name:"Alex"}]) // ordered bulk insert
- db.coll.insert([{name: "Max"}, {name:"Alex"}], {ordered: false}) // unordered bulk insert
- db.coll.insert({date: ISODate()})
- db.coll.insert({name: "Max"}, {"writeConcern": {"w": "majority", "wtimeout": 5000}})

Read

- db.coll.findOne() // returns a single document
- db.coll.find() // returns a cursor - show 20 results - "it" to display more
- db.coll.find().pretty()
- db.coll.find({name: "Max", age: 32}) // implicit logical "AND".
- db.coll.find({date: ISODate("2020-09-25T13:57:17.180Z")})
- db.coll.find({name: "Max", age: 32}).explain("executionStats") // or "queryPlanner" or "allPlansExecution"
- db.coll.distinct("name")

Motivation Tree

// Count
- db.coll.count({age: 32}) // estimation based on collection metadata
- db.coll.estimatedDocumentCount() // estimation based on collection metadata
- db.coll.countDocuments({age: 32}) // alias for an aggregation pipeline - accurate count

// Comparison
- db.coll.find({"year": {$gt: 1970}})
- db.coll.find({"year": {$gte: 1970}})
- db.coll.find({"year": {$lt: 1970}})
- db.coll.find({"year": {$lte: 1970}})
- db.coll.find({"year": {$ne: 1970}})
- db.coll.find({"year": {$in: [1958, 1959]}})
- db.coll.find({"year": {$nin: [1958, 1959]}})

// Logical
- db.coll.find({name:{$not: {$eq: "Max"}}})
- db.coll.find({$or: [{"year" : 1958}, {"year" : 1959}]})
- db.coll.find({$nor: [{price: 1.99}, {sale: true}]})
- db.coll.find({
- $and: [
- {$or: [{qty: {$lt :10}}, {qty :{$gt: 50}}]},
- {$or: [{sale: true}, {price: {$lt: 5 }}]}
-]
- })

Motivation Tree

```
// Element
  • db.coll.find({name: {$exists: true}})
  • db.coll.find({"zipCode": {$type: 2 }})
  • db.coll.find({"zipCode": {$type: "string"}})

// Aggregation Pipeline
  • db.coll.aggregate([
 {$match: {status: "A"}},
 {$group: {_id: "$cust_id", total: {$sum: "$amount"}}},
 {$sort: {total: -1}}
])

// Text search with a "text" index
  • db.coll.find({$text: {$search: "cake"}}, {score: {$meta:
    "textScore"}}).sort({score: {$meta: "textScore"}})

// Regex
  • db.coll.find({name: /^Max/})   // regex: starts by letter "M"
  • db.coll.find({name: /^Max$/i}) // regex case insensitive
```

Motivation Tree

```
// Array
  • db.coll.find({tags: {$all: ["Realm", "Charts"]}})
  • db.coll.find({field: {$size: 2}}) // impossible to index - prefer storing the
    size of the array & update it
  • db.coll.find({results: {$elemMatch: {product: "xyz", score: {$gte: 8}}}})

// Projections
  • db.coll.find({"x": 1}, {"actors": 1})              // actors + _id
  • db.coll.find({"x": 1}, {"actors": 1, "_id": 0})    // actors
  • db.coll.find({"x": 1}, {"actors": 0, "summary": 0}) // all but "actors" and
    "summary"

// Sort, skip, limit
  • db.coll.find({}).sort({"year": 1, "rating": -1}).skip(10).limit(3)

// Read Concern
  • db.coll.find().readConcern("majority")
```