Cheatography

NoSQL Cheat Sheet

by Xplendit via cheatography.com/25193/cs/6504/

Getting acquainted with MongoDB

show dbs	To see the list of databases in the system
use dbsname	Switched to db dbsname
db.getName()	To find out the currently selected database
show collections	To see the collections in a databases
db.dbname.insert({colomname:"data", colomname:data})	Save some data in the database

Create / add / find data in MongoDB

db.dbname.insert({columnname:'stringdata', columnname:intdata})	To create documents in a collection
db.comedy.find()	To read data from a collection

Conditional Operators

db.collection.find({ "field" : { $gt: value } })	greater than : field > value
db.collection.find({ "field" : { $lt: value } })	less than : field < value
db.collection.find({ "field" : { $gte: value } })	greater than or equal to : field >= value
db.collection.find({ "field" : { $lte: value } })	less than or equal to : field <= value
db.comedy.find({year: { $gt: 2007, $lt: 2011} })	You can also combine these operators to specify ranges
db.comedy.find({year: 2012})	And how do we do an 'equal to' query? Just match the value for the queried key
db.comedy.find({year: {'$lt':2012}}, {name:true})	What if you want to get only some fields in the result?
db.comedy.find({year: {'$lt':2012}}, {name:false})	What if you want to get all, except some fields in the result?
db.comedy.find({year: {$ne: 2011}})	Use $ne for "not equals".
db.comedy.find({year: {$in: [2010,2011,2012]}})	The $in operator is analogous to the SQL IN modifier, allowing you to specify an array of possible matches.

Conditional Operators (cont)

db.comedy.find({year: {$nin: [2010,2011,2012]}})	The $nin operator is similar to $in except that it selects objects for which the specified field does not have any value in the specified array
db.comedy.find({year: {$nor: [2010,2011,2012]}})	The $nor operator lets you use a boolean or expression to do queries. You give $nor a list of expressions, none of which can satisfy the query.
db.comedy.find({$or: [{year: 2012}, {name: 'The hangover'}]})	The $or operator lets you use boolean or in a query. You give $or an array of expressions, any of which can satisfy the query.
db.comedy.find({$and :[{year: {$gt: 2010}}, {year:{$lt: 2012}}]})	The $and operator lets you use boolean and in a query. You give $and an array of expressions, all of which must match to satisfy the query

Update data in MongoDB

db.comedy.update({name:"Ted"}, {'$set':{director:'Seth MacFarlane', cast:['Mark Wahlberg', 'Mila Kunis', 'Matt Walsh', 'Jessica Barth', 'Aedin Mincks']}})	update
db.comedy.update({name:"Ted 2"}, {'$set':{year: 2015}}, {upsert: true})	By specifying upsert: true, applications can indicate, in a single operation, that if no matching documents are found for the update, an insert should be performed.
db.comedy.update({year:2012}, {'$set':{rating: 4}}, {multi: true})	owever, with the multi option, update() can update all documents in a collection that match a query.
db.comedy.update({name:"Ted"}, {'$push':{cast:'Joel McHale'}})	Now how do you update a value which is an array?

Update data in MongoDB (cont)

db.comedy.update({name:"Ted"}, {'$pull':{cast:'Giovanni Ribisi'}})	If we need to remove something from the cast array. We do it this way:
db.comedy.update({name:"Ted"}, {'$pop':{cast:-1}})	We can also use $pop to remove the first element
db.comedy.update({name:"Ted"}, {'$pop':{cast:1}})	We can also use $pop to remove the last element

Dot notation

db.articles.find({'meta.author':'Chad Muska'})	To search an object inside an object, just use the regular JavaScript dot notation of the target object as the key AND quote it.
db.articles.find({'meta.tags':'mongolia'})	You need to query an array? No problem
db.articles.find({'comments.by':'Steve'-})	When the key is an array, the database looks for the object right in the array. You need to look for an object inside an array?
db.articles.find({ comments : { $size: 2 } })	The $size operator matches any array with the specified number of elements.

You cannot use $size to find a range of sizes (for example: arrays with more than 1 element). If you need to query for a range, create an extra size field that you increment when you add elements.

Regular expression

db.comedy.find({name:{$regex: /bill|ted/i}})	We can even use regular expressions in our queries
db.comedy.find({name: /The hangover.*/i });	We can even use regular expressions in our queries
db.comedy.find({name: {$regex: 'The hangover.*', $options: 'i'}});	We can even use regular expressions in our queries
db.comedy.find({name: {$regex: /The hangover.*/i, $nin: ['The Hangover Part II']}});	If you wish to specify both a regex and another operator for the same field, you need to use the $regex clause.
db.comedy.find({name: {$not: /The hangover.*/i}});	The $not meta operator can be used to negate the check performed by a standard operator. For example:

Regular expression (cont)

db.comedy.find('this.year > 2009 && this.name !== "Ted"')	MongoDB queries support JavaScript expressions!
db.comedy.find({$where: 'this.year > 2011'})	MongoDB has another operator called $where using which you can perform SQL's WHERE-like operations.

De rest

db.comedy.count()	This will return the total number of documents in the collection named comedy
db.comedy.count({year: {$gt:2009}})	This will return the total number of documents in the collection named comedy with the value of year more than 2009:
db.comedy.find( ).limit(2)	To limit the collection to just two:
db.comedy.findOne()	Similar to using find().limit(1), there is a function called findOne(), which will get you only one document in the result.
db.comedy.find( ).skip(1).limit(2)-	The skip() expression allows one to specify at which object the database should begin returning results.
db.articles.find ({'meta.tags': {$all: ['mongodb', 'mongo']}});	The $all operator is similar to $in, but instead of matching any value in the specified array all values in the array must be matched.
db.articles.find ({title: {$exists : true}});	Check for existence (or lack thereof) of a field.
db. articles.find().sort({title : -1})	sort() is analogous to the ORDER BY statement in SQL - it requests that items be returned in a particular order. We pass sort() a key pattern which indicates the desired order for the result.

An array can have more elements than those specified by the $all criteria. $all specifies a minimum set of elements that must be matched.

Delete data in MongoDB	
db.comedy.update({name:'Ted'}, {$unset:{cast:1}})	How do you delete a field from a document?
db.comedy.update({$unset:{cast:1}}, false, true)	In case you want to delete a field from all the documents of a collection:
db.comedy.remove({name:'Ted'})	How do you delete a document from a collection?
db.comedy.remove({})	How do you empty a collection of its documents?
db.comedy.drop()	How do you delete / drop a collection?
use movies; db.dropDatabase()	To delete a database select the database and call the db.dropDatabase() on it:

By **Xplendit**

cheatography.com/xplendit/

Published 26th December, 2015.

Last updated 26th December, 2015.