

```
1 /* Cheatsheet for MongoDB
2 Created by: Truly Mittal
3 Date: March 2019
4 */
5
6 /* SHOW DATABASE */
7 show dbs
8
9 /* USE DATABASE */
10 use DATABASE_NAME
11
12 /* current database */
13 db
14
15 /* DROP(delete) DATABASE */
16 db.dropDatabase()
17
18 /* CREATE COLLECTION */
19 db.createCollection("COLLECTION_NAME", {OPTIONS})
20
21 /* DROP(delete) COLLECTION */
22 db.COLLECTION_NAME.drop()
23
24 /* INSERT document */
25 db.users.insert({
26     name: "Truly Mittal",
27     age: 29
28 })
29
30 /* gettings all documents */
31 db.COLLECTION_NAME.find()
32
33 /* pretty the output */
34 db.COLLECTION_NAME.find().pretty()
35
36 /* finding a document by _id */
37 db.users.find({ _id: ObjectId("5c4d7c52eb87a6e0e5fc8949") }).pretty()
38
39 /* Finding by a field, here it is name */
40 db.users.find({name: "Truly Mittal"}).pretty()
41
42 /* update document */
43 db.users.update(
44     { name: "Truly Mittal" },
45     { $set : { age: 29 } }
46 )
47
48 /* unset a field in document */
49 db.users.update(
50     { name: "Truly Mittal" },
51     { $unset : { age: "" } }
52 )
53
54 /* update and auto insert if document not present */
55 db.users.update(
56     { name: "Truly Mittal" },
```

```

57     { name: "Truly Mittal" },
58     { upsert: true }
59 )
60
61 /* Remove a document */
62 db.COLLECTION_NAME.remove({query})
63 db.COLLECTION_NAME.deleteOne({query})
64
65 /* insert multiple documents */
66 db.users.insert(
67     [
68         {
69             name: "Jhon Doe",
70             age: 36,
71             colors: ["red", "black", "blue"],
72             address: {
73                 city: "paris",
74                 country: "france"
75             }
76         },
77         {
78             name: "Hans Down",
79             age: 46,
80             colors: ["yellow", "white"],
81             address: {
82                 city: "london",
83                 country: "uk"
84             }
85         },
86         {
87             name: "Eric Widget",
88             age: 31,
89             colors: ["red", "green", "purple"],
90             address: {
91                 city: "new delhi",
92                 country: "india"
93             }
94         },
95         {
96             name: "Indigo Violet",
97             age: 29,
98             colors: ["red", "black"],
99             address: {
100                 city: "washington dc",
101                 country: "us"
102             }
103         }
104     ]
105 )
106
107 /* count the number of documents */
108 db.users.count() /* Number of documents */
109
110 /* limit the number of fetched records */
111 db.users.find().limit(2) /* limit the number of records */
112
113 /* skip the number of records */

```

```

114 db.users.find().skip(2)
115
116
117 /* sort the record by the key name (1 Asc, -1 Desc) */
118 db.users.find().sort({name: 1})      /* sorts the records in asc and desc */
119
120 /* Update Multiple docs */
121 db.users.updateMany(
122     {}, //Filter
123     {} //Updates
124 )
125
126 db.users.update(
127     {}, //Filter
128     {}, //Updates
129     { multi: true } //required in options if using only update instead of
updateMany
130 )
131
132
133 /* Change the Field Name */
134 db.users.updateMany(
135     {}, //Filter
136     { $rename: { "colours" : "colors" } } //Updates
137 )
138
139 /* AND query */
140 db.users.find({
141     $and: [
142         { name: "Indigo Violet" },
143         { age: 29 }
144     ]
145 })
146
147 /* OR query */
148 db.users.find({
149     $or: [
150         { name: "Indigo Violet" },
151         { age: 29 }
152     ]
153 })
154
155 /* OR with Less than equal to query */
156 db.users.find({
157     $or: [
158         { name: "John Doe" },
159         { age: { $lte : 31 } }
160     ]
161 })
162
163 /* not query */
164 db.users.find({
165     age: {
166         $not: {
167             $lt: 40
168         }
169     }

```

```

170 }).pretty()
171
172 /* array operations
173 adds the whole array as an element inside an array */
174 db.users.update(
175     { name: "Truly Mittal" },
176     { $addToSet : { colors: ["yellow", "pink"] } }
177 )
178
179 /* adds each element of the array as an element inside an array works as a
180 SET */
181 db.users.update(
182     { name: "Truly Mittal" },
183     {
184         $addToSet : {
185             colors: {
186                 $each: ["yellow", "pink"]
187             }
188         }
189     }
190 )
191 /* repeat above also to show that NOTHING happens */
192
193 /* adds each element of the array as an element inside an array NOT works as
194 a SET */
195 db.users.update(
196     { name: "Truly Mittal" },
197     {
198         $push : {
199             colors: {
200                 $each: ["yellow", "pink"]
201             }
202         }
203     }
204 )
205 /* removes all entries from the array */
206 db.users.update(
207     { name: "Truly Mittal" },
208     {
209         $pull : { colors: ["yellow", "pink"] }
210     }
211 )
212
213 db.users.update(
214     { name: "Truly Mittal" },
215     {
216         $pull : { colors: "yellow" }           //removes all enteries of
217         "yellow"
218     }
219 )
220 db.users.update(
221     { name: "Truly Mittal" },
222     {
223         $pull : { colors: "pink" }

```

```

224     }
225 )
226
227 /* The $in operator selects the documents where the value of a field equals
any value in the specified array. */
228 db.users.update(
229     { },
230     {
231         $pull: {
232             colors: {
233                 $in: [ "red", "black" ]
234             }
235         }
236     },
237     { multi: true }
238 )
239
240 /* Increment Operator */
241 db.users.update(
242     { name: "Truly Mittal" },
243     {
244         $inc : { age: 2 }
245     }
246 )
247
248 /* $pop */
249 db.users.update(
250     {name: "Indigo Violet" },
251     { $set : {
252         scores: [ 3, 65, 2 ]
253     }}
254 )
255
256 db.users.update(
257     {name: "Indigo Violet" },
258     { $pop : { scores: 1 } }
259 )
260
261 db.users.updateMany(
262     { },
263     {
264         $push: {
265             colors: {
266                 $each: [ "red", "black" ]
267             }
268         }
269     }
270 )
271
272 /* Setting the current date in field called "lastModified" */
273 db.inventory.updateOne(
274     { name: "Truly Mittal" },
275     { $currentDate: { lastModified: true } }
276 )
277
278 /*

```

```
279 The $currentDate operator sets the value of a field to the current date,
    either as a Date or a timestamp. The default type is Date.
280 { $currentDate: { <field1>: <typeSpecification1>, ... } }
281     <typeSpecification> can be either:
282 1. a boolean true to set the field value to the current date as a Date, or
283 2. a document { $type: "timestamp" } or { $type: "date" } which explicitly
    specifies the type.
284     The operator is case-sensitive and accepts only the lowercase "timestamp"
    or the lowercase "date". */
285
286
287 /* Exporting in docs using mongoexport */
288 mongoexport --db DB_NAME --collection COLLECTION_NAME -o FILENAME.json
289
290 /* Exporting in docs using mongoexport as JSON ARRAY */
291 mongoexport --db DB_NAME --collection COLLECTION_NAME -o FILENAME.json --
    jsonArray
292
293 /* Importing Docs from a file */
294 mongoimport --db DB_NAME --collection COLLECTION_NAME --file FILENAME.json
295
296 /* Importing Docs from a file if it is formatted as a JSON ARRAY */
297 mongoimport --db DB_NAME --collection COLLECTION_NAME --file FILENAME.json -
    -jsonArray
298
299
300
```