rmongodb Cheat Sheet

https://github.com/mongosoup/rmongodb

General Package Handling

install from CRAN

```
> install.packages("rmongodb")
```

install dev version from GitHub

```
> library(devtools)
> install_github("rmongodb", "mongosoup")
```

load rmongodb package

```
> library(rmongodb)
```

get help overview

```
> ??rmongodb
```

Connection Handling

connect to localhost

```
> mongo <- mongo.create()
```

connect to external mongoDB

```
> mongo <- mongo.create(host="127.1.1.1:27017",
+             username="USER", password="XXX",
+             db="database")
```

check for working connection

```
> mongo.is.connected(mongo)
```

disconnect from mongoDB

```
> mongo.destroy(mongo)
```

Get Database Information

get databases and collections of a database

```
> mongo.get.databases(mongo)
> mongo.get.database.collections(mongo, "rmongodb")
```

get errors from mongoDB

```
> mongo.get.err(mongo)
> mongo.get.server.err(mongo)
> mongo.get.server.err.string(mongo)
```

deal with mongoDB replica sets

```
> mongo.get.primary(mongo)
> mongo.get.hosts(mongo)
```

Get Info about Documents

count all elements in collection

```
> mongo.count(mongo, "rmongodb.zips")
```

Hint: Collection name is a namespace with 'database.collection'.
get all values for one key (in this case "city")

```
> mongo.get.values(mongo, "rmongodb.zips", "city")
```

Querying Data

count documents of a special query

```
> mongo.count(mongo, "rmongodb.zips",
+             query='{"state":"AL"}')
```

find one document and returns BSON object.

```
> bson <- mongo.findOne(mongo, "rmongodb.zips",
+                       query='{"state":"AL"}')
```

find all documents and returns mongo cursor

```
> cursor <- mongo.find(mongo, "rmongodb.zips",
+                      query='{"state":"AL"}')
```

convert cursor to R object

```
> mongo.cursor.to.list(cursor)
```

direct query and create R object

```
> mongo.find.all(mongo, "rmongodb.zips",
+             query='{"state":"AL"}')
```

more query options

skip first 5 documents and limit query to 10 results

```
> mongo.find.all(mongo, "rmongodb.zips",
+             query='{"state":"AL"}',
+             skip=5, limit=10)
```

only return special keys / fields (in this case "city" and "pop") and sort by a key (in this case "pop")

```
> mongo.find.all(mongo, "rmongodb.zips",
+             query='{"state":"AL"}',
+             fields='{"city":1, "pop":1, "_id":0}',
+             sort='{"pop":1}')
```

more mongoDB query examples

an "and" query

```
> mongo.find.all(mongo, "rmongodb.zips",
+             query='{"state":"AL", "city":"ACMAR"}')
```

a comparing query

```
> mongo.find.all(mongo, "rmongodb.zips",
+             query='{"pop":{"$gte":80000}}')
```

count documents where key / field exists

```
> mongo.count(mongo, "rmongodb.zips",
+             query='{"loc":{"$exists":1}}')
```

Hint: a good starting point for more queries are the mongoDB references and tutorials:
http://docs.mongodb.org/manual/tutorial/query-documents/
http://docs.mongodb.org/manual/reference/sql-comparison/

Dealing with BSON objects

convert BSON to R object

```
> mongo.bson.to.list(bson)
```

get one key (in this case "state") out of BSON object

```
> mongo.bson.value(bson, "state")
```

creating BSON objects

Hint: due to new JSON to BSON functionality this is no longer required. Since version 1.3 you can directly query with JSON!
convert JSON to BSON

```
> mongo.bson.from.JSON('{"state":"AL"}')
```

old way to create same BSON object

```
> buf <- mongo.bson.buffer.create()
> mongo.bson.buffer.append(buf, "state", "AL")
> b <- mongo.bson.from.buffer(buf)
```

old way still available. check help files.

```
> ?mongo.bson
```

Importing and Updating Data

insert one document to collection

```
> mongo.insert(mongo, "rmongodb.insert",
+             '{"user":"markus", "city":"munich"}')
```

insert many documents to collection

```
> bson1 <- mongo.bson.from.JSON(
+   '{"user":"markus", "city":"munich"}')
> bson2 <- mongo.bson.from.JSON(
+   '{"user":"peter", "city":"New York"}')
> mongo.insert.batch(mongo, "rmongodb.insert",
+                   list(bson1, bson2))
```

add index to collection

```
> mongo.index.create(mongo, "rmongodb.insert", '{"user":1}')
```

update one document in the collection

```
> mongo.update(mongo, "rmongodb.insert",
+             '{"user":"markus"}',
+             '{"user":"markus", "city":"berlin"}')
```

Aggregation Framework

grouping and matching on zips example data. Creates BSON output object

```
> pipe_1 <- mongo.bson.from.JSON('{"$group":
+                                 {"_id":"$state", "totalPop":
+                                 {"$sum":"$pop"}}}')
> pipe_2 <- mongo.bson.from.JSON('{"$match":
+                                 {"totalPop":
+                                 {"$gte":15000000}}}')
> cmd_list <- list(pipe_1, pipe_2)
> bson <- mongo.aggregation(mongo, "rmongodb.zips", cmd_list)
```

MongoSoup
YOU CODE, WE COOK.

https://www.mongosoup.de/rmongodb.html