MongoDB

NoSQL Datastore

NoSQL Landskab

Key Value

Example:
Riak, Tokyo Cabinet, Redis server, Memcached, Scalaris

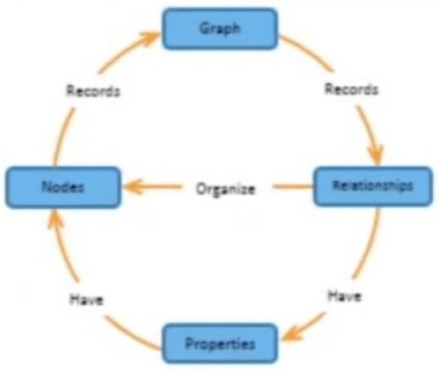
Document-Based

Example:
MongoDB, CouchDB, OrientDB, RavenDB

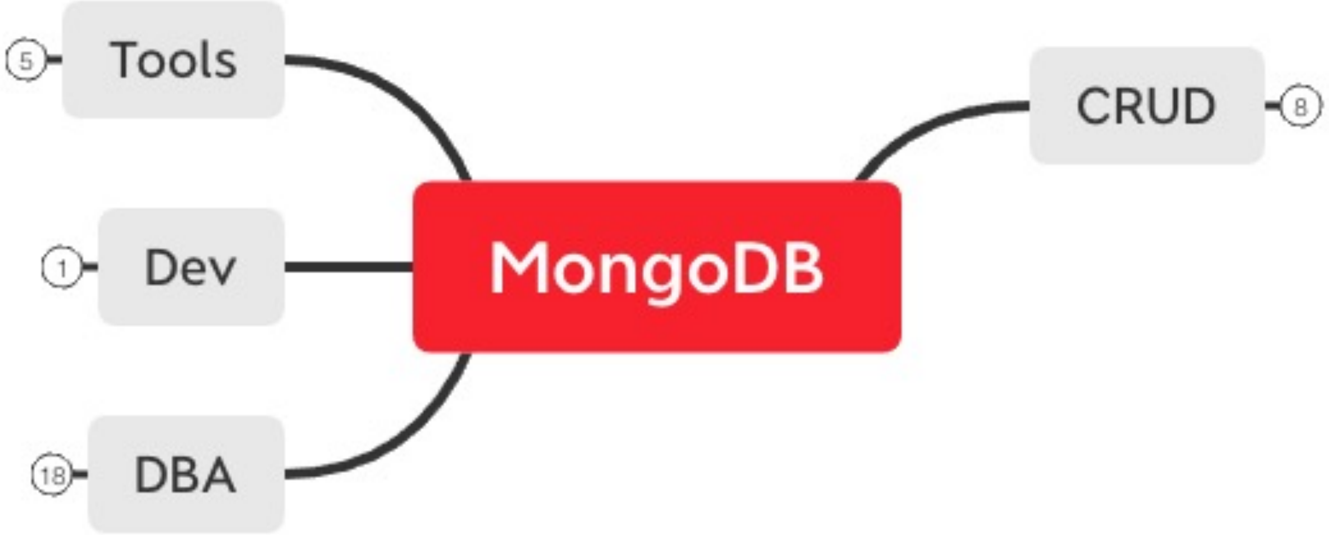
Column-Based

Example:
BigTable, Cassandra, Hbase, Hypertable

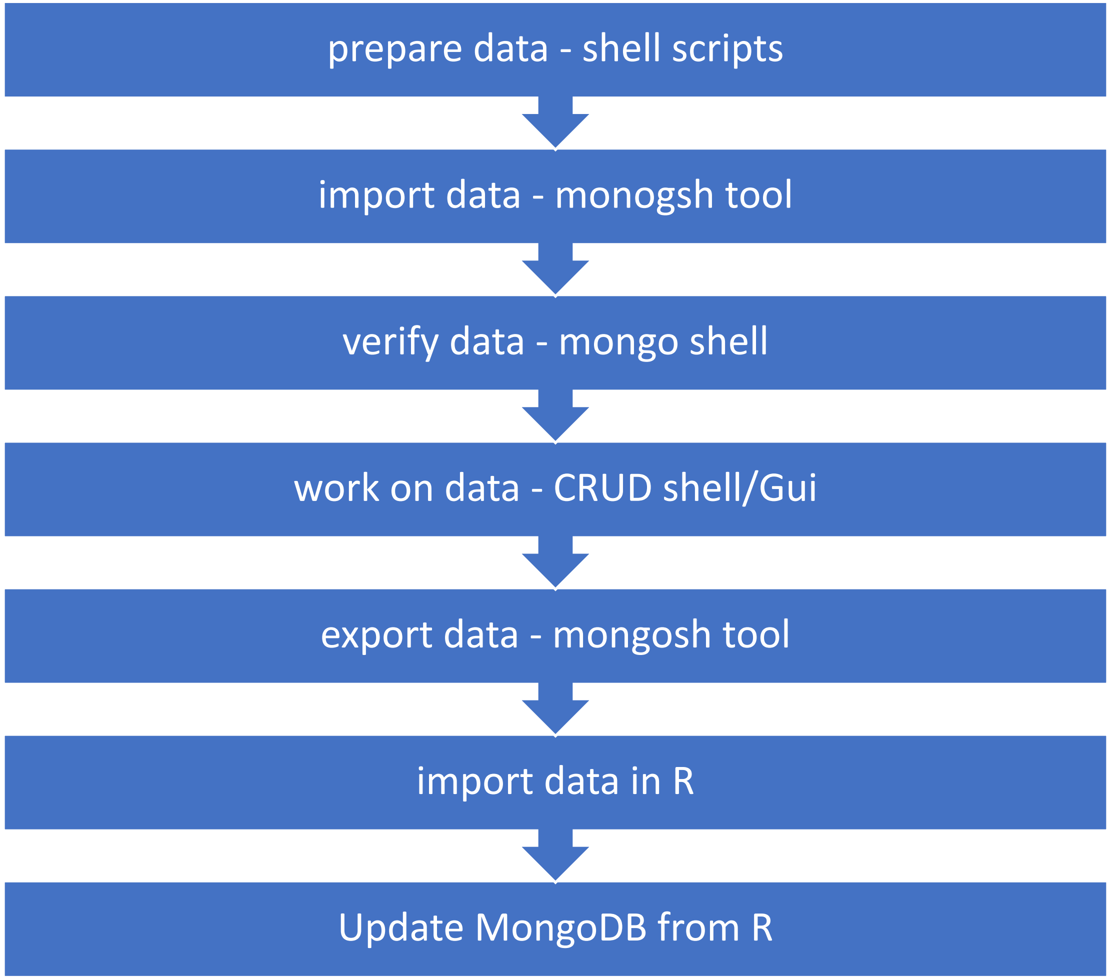
Graph-Based



Example:
Neo4J, InfoGrid, Infinite Graph, Flock DB


MongoDB
Tools
5
Dev
1
DBA
18
CRUD
8

Flow



prepare data - shell scripts

import data - monogsh tool

verify data - mongo shell

work on data - CRUD shell/Gui

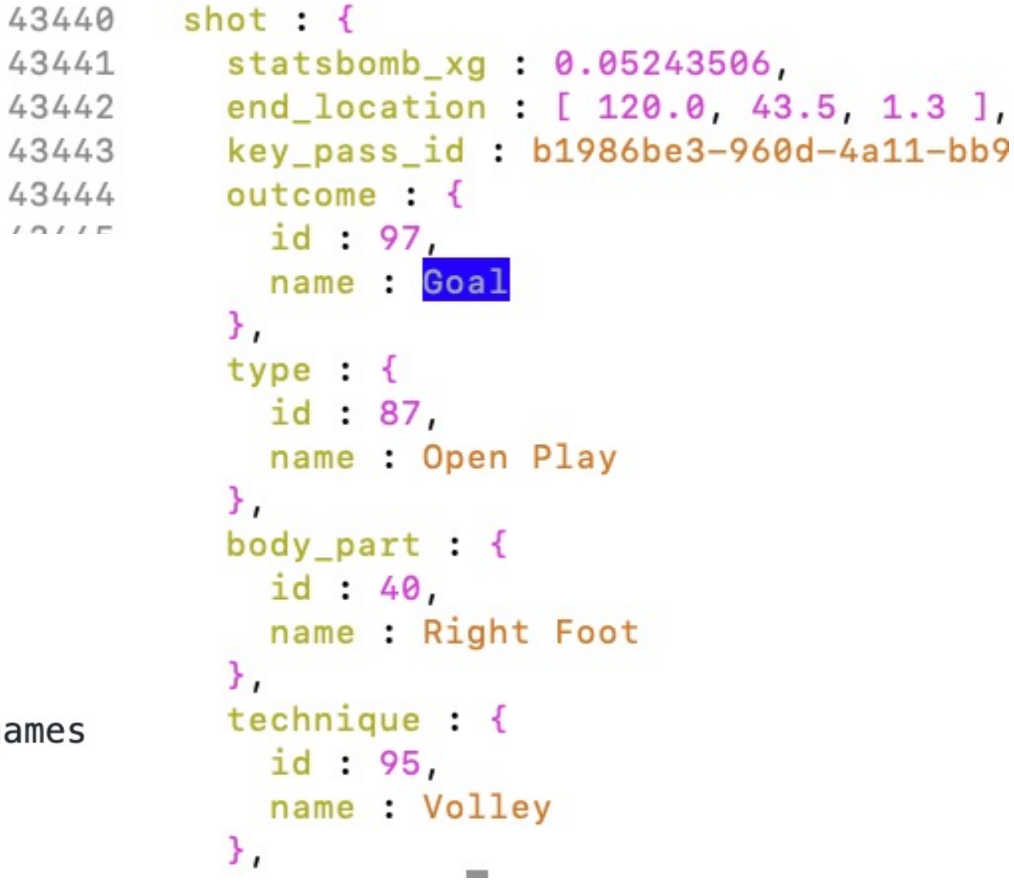
export data - mongosh tool

import data in R

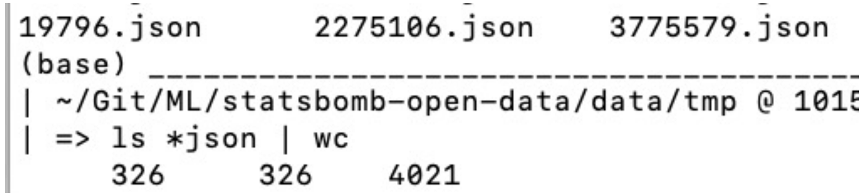
Update MongoDB from R

prepare data - shell scripts

```
#!/usr/bin/bash -x

find ./matches/* -type f | while read x
do
  grep -l female $x
done | while read x
do
  grep match_id $x
done | cut  -d\: -f2 |  cut -d\, -f1 | sort | uniq > listoffemgames
cat listoffemgames  | while  read x
do
  cp events/$x.json tmp
done
for i in tmp/*json; do sed -i 's/^\[//g' $i; done
for i in tmp/*json; do sed -i 's/^} \]/}/g' $i; done
for i in tmp/*json; do sed -i 's/^  \"id/  \"_id/g' $i;done
for i in tmp/*json; do sed -i 's/^}, {/}\n{/g' $i;done
#for i in *json;  do mongoimport -d statsbomb -c matches $i;done
```

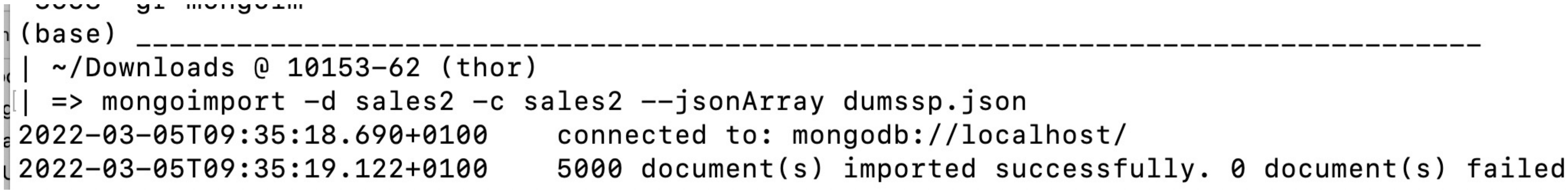


```
shot : {
  statsbomb_xg : 0.05243506,
  end_location : [ 120.0, 43.5, 1.3 ],
  key_pass_id : b1986be3-960d-4a11-bb9
  outcome : {
    id : 97,
    name : Goal
  },
  type : {
    id : 87,
    name : Open Play
  },
  body_part : {
    id : 40,
    name : Right Foot
  },
  technique : {
    id : 95,
    name : Volley
  },
```



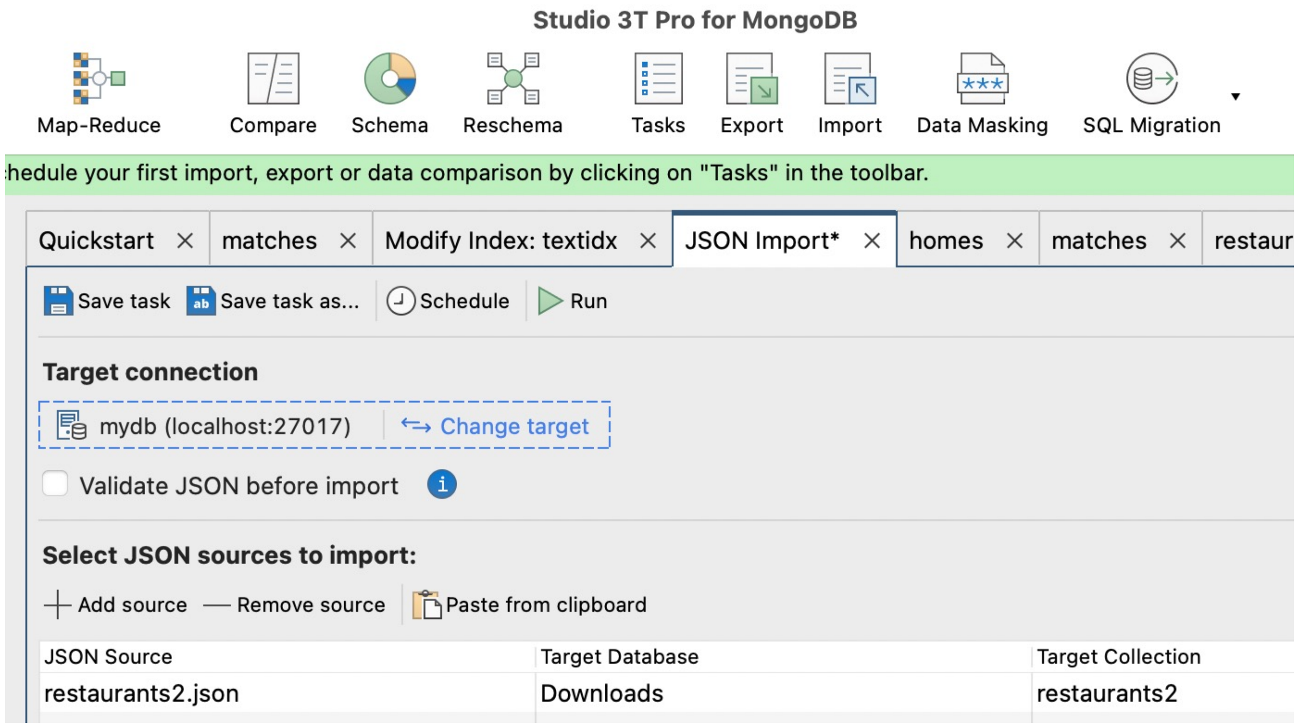
```
19796.json       2275106.json     3775579.json
(base) _______________________________________
| ~/Git/ML/statsbomb-open-data/data/tmp @ 1015
| => ls *json | wc
     326     326    4021
```

import data - monogsh tool



```
(base) ________________________________________________________________________________
| ~/Downloads @ 10153-62 (thor)
| => mongoimport -d sales2 -c sales2 --jsonArray dumssp.json
2022-03-05T09:35:18.690+0100    connected to: mongodb://localhost/
2022-03-05T09:35:19.122+0100    5000 document(s) imported successfully. 0 document(s) failed
```

Or GUI



Studio 3T Pro for MongoDB

:hedule your first import, export or data comparison by clicking on "Tasks" in the toolbar.

Target connection

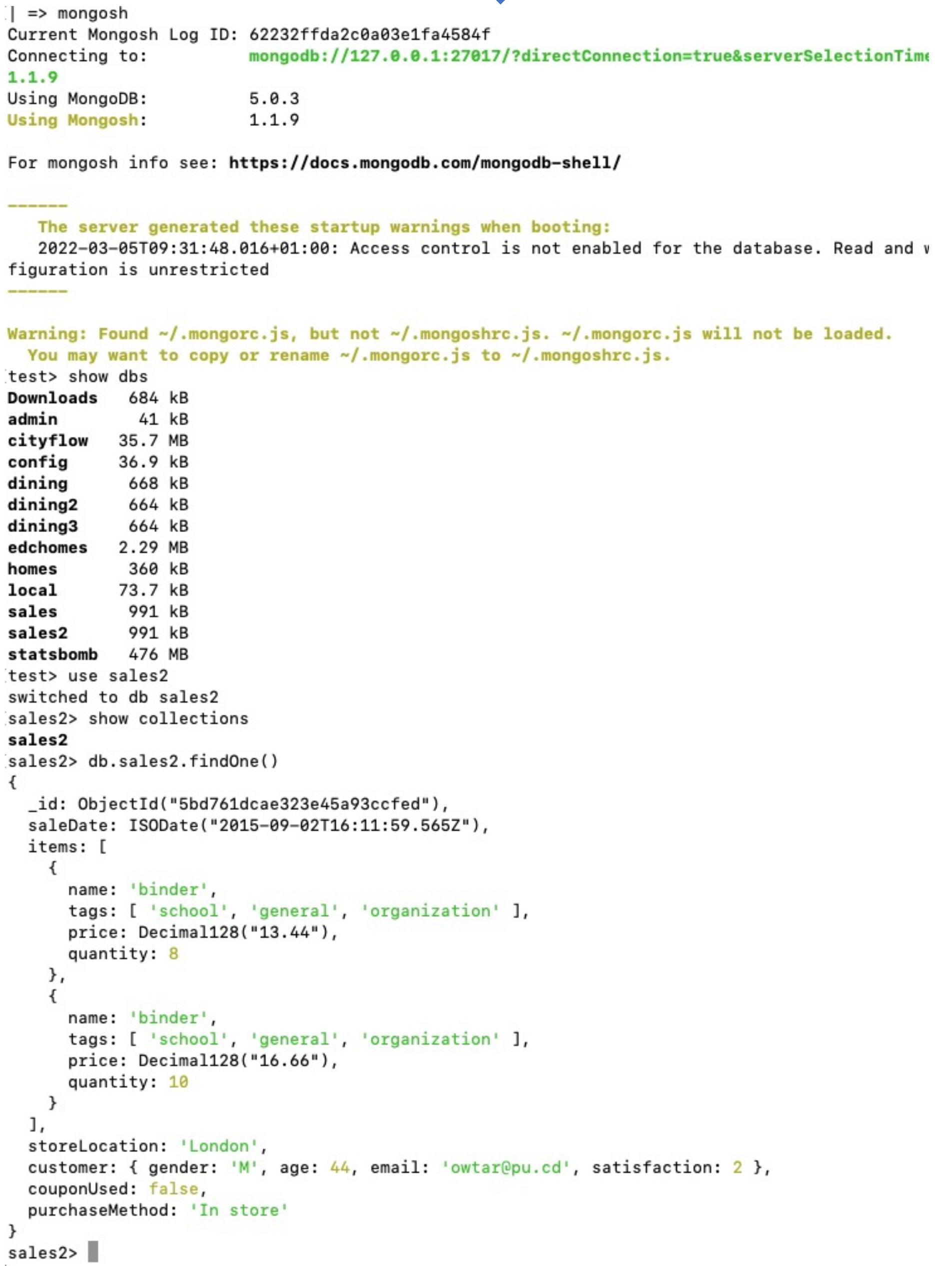
mydb (localhost:27017) Change target

□ Validate JSON before import

Select JSON sources to import:

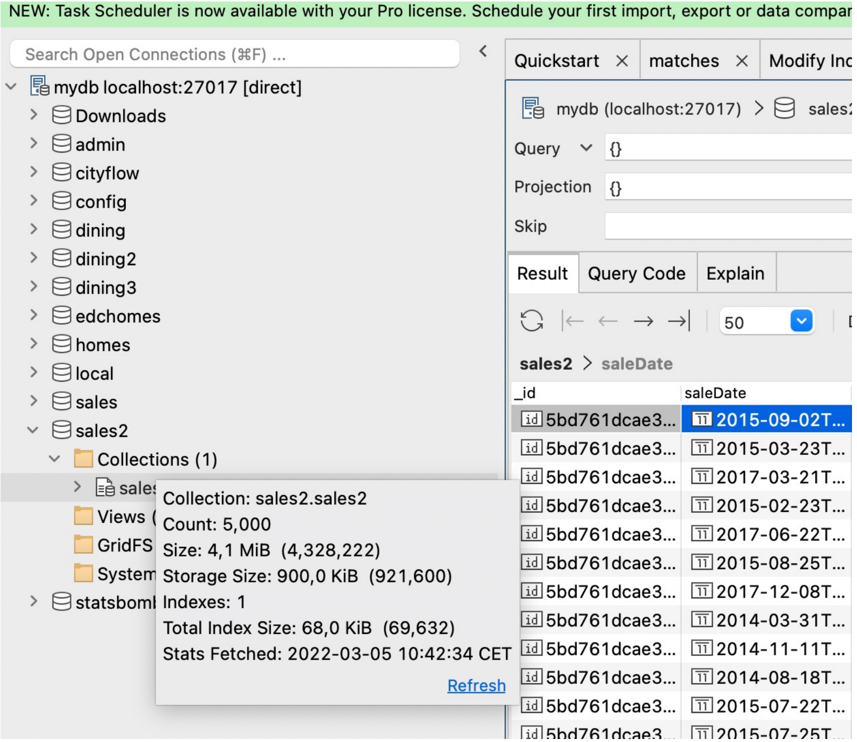
JSON Source	Target Database	Target Collection
restaurants2.json	Downloads	restaurants2

verify data - mongo shell



```
| => mongosh
Current Mongosh Log ID: 62232ffda2c0a03e1fa4584f
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTime
1.1.9
Using MongoDB:          5.0.3
Using Mongosh:          1.1.9

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------
   The server generated these startup warnings when booting:
   2022-03-05T09:31:48.016+01:00: Access control is not enabled for the database. Read and w
figuration is unrestricted
------

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
  You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.
test> show dbs
Downloads    684 kB
admin         41 kB
cityflow    35.7 MB
config      36.9 kB
dining       668 kB
dining2      664 kB
dining3      664 kB
edchomes    2.29 MB
homes        360 kB
local       73.7 kB
sales        991 kB
sales2       991 kB
statsbomb    476 MB
test> use sales2
switched to db sales2
sales2> show collections
sales2
sales2> db.sales2.findOne()
{
  _id: ObjectId("5bd761dcae323e45a93ccfed"),
  saleDate: ISODate("2015-09-02T16:11:59.565Z"),
  items: [
    {
      name: 'binder',
      tags: [ 'school', 'general', 'organization' ],
      price: Decimal128("13.44"),
      quantity: 8
    },
    {
      name: 'binder',
      tags: [ 'school', 'general', 'organization' ],
      price: Decimal128("16.66"),
      quantity: 10
    }
  ],
  storeLocation: 'London',
  customer: { gender: 'M', age: 44, email: 'owtar@pu.cd', satisfaction: 2 },
  couponUsed: false,
  purchaseMethod: 'In store'
}
sales2> 
```

Or GUI



NEW: Task Scheduler is now available with your Pro license. Schedule your first import, export or data compar

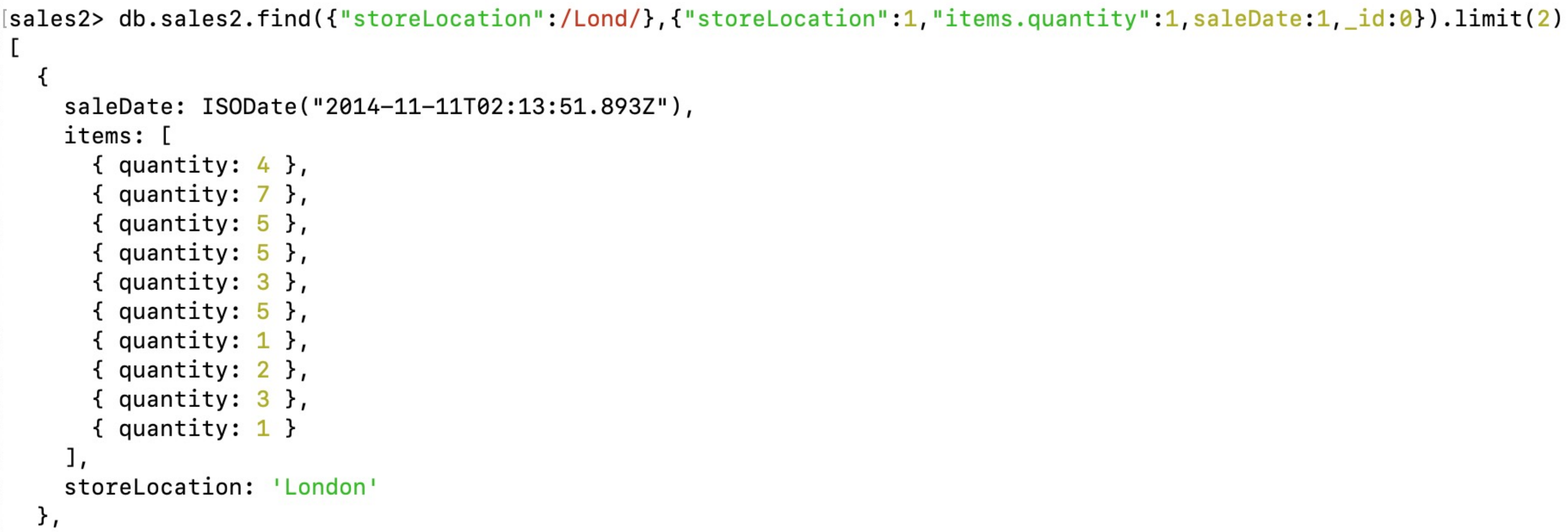
Query {}

Projection {}

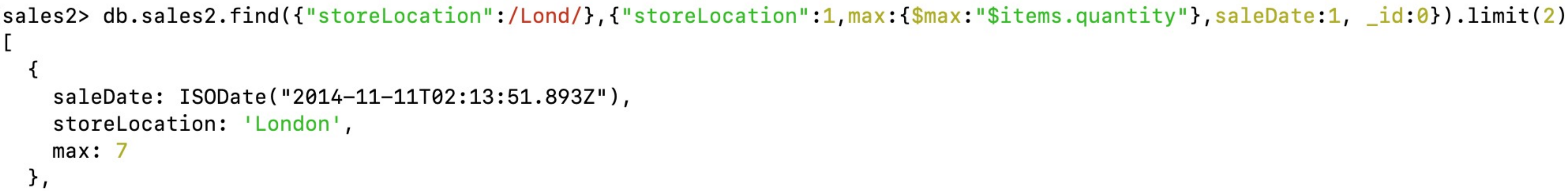
Collection: sales2.sales2
Count: 5,000
Size: 4,1 MiB (4,328,222)
Storage Size: 900,0 KiB (921,600)
Indexes: 1
Total Index Size: 68,0 KiB (69,632)
Stats Fetched: 2022-03-05 10:42:34 CET

_id	saleDate
5bd761dcae3...	2015-09-02T...
5bd761dcae3...	2015-03-23T...
5bd761dcae3...	2017-03-21T...
5bd761dcae3...	2015-02-23T...
5bd761dcae3...	2017-06-22T...
5bd761dcae3...	2015-08-25T...
5bd761dcae3...	2017-12-08T...
5bd761dcae3...	2014-03-31T...
5bd761dcae3...	2014-11-11T...
5bd761dcae3...	2014-08-18T...
5bd761dcae3...	2015-07-22T...
5bd761dcae3...	2015-07-25T...

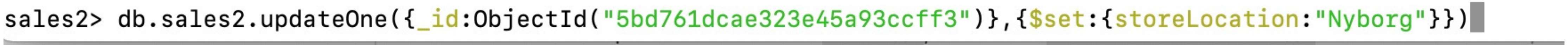
work on data - CRUD shell/Gui



```
sales2> db.sales2.find({"storeLocation":/Lond/},{"storeLocation":1,"items.quantity":1,saleDate:1,_id:0}).limit(2)
[
  {
    saleDate: ISODate("2014-11-11T02:13:51.893Z"),
    items: [
      { quantity: 4 },
      { quantity: 7 },
      { quantity: 5 },
      { quantity: 5 },
      { quantity: 3 },
      { quantity: 5 },
      { quantity: 1 },
      { quantity: 2 },
      { quantity: 3 },
      { quantity: 1 }
    ],
    storeLocation: 'London'
  },
```

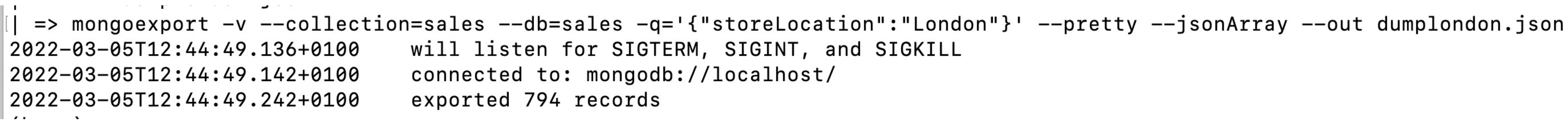


```
sales2> db.sales2.find({"storeLocation":/Lond/},{"storeLocation":1,max:{$max:"$items.quantity"},saleDate:1, _id:0}).limit(2)
[
  {
    saleDate: ISODate("2014-11-11T02:13:51.893Z"),
    storeLocation: 'London',
    max: 7
  },
```

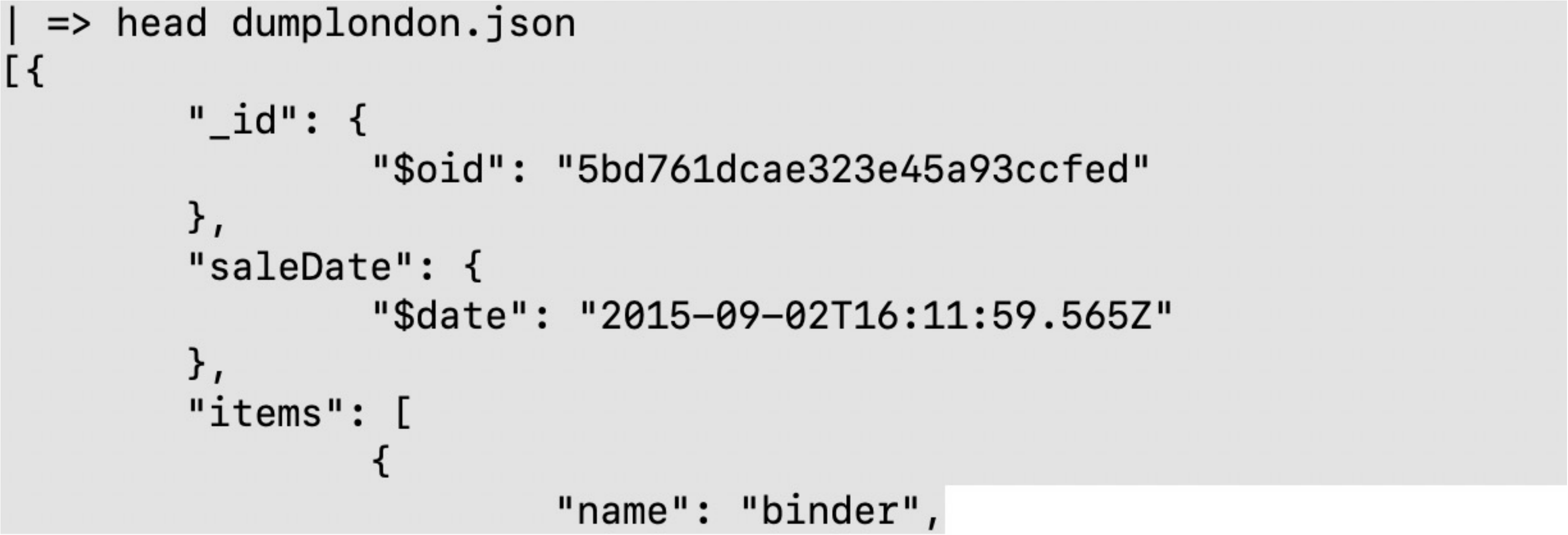


```
sales2> db.sales2.updateOne({_id:ObjectId("5bd761dcae323e45a93ccff3")},{$set:{storeLocation:"Nyborg"}})
```

export data - mongodb tool

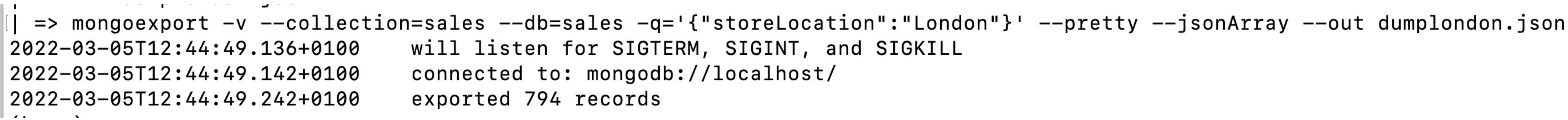


```
| => mongoexport -v --collection=sales --db=sales -q='{"storeLocation":"London"}' --pretty --jsonArray --out dumplondon.json
2022-03-05T12:44:49.136+0100    will listen for SIGTERM, SIGINT, and SIGKILL
2022-03-05T12:44:49.142+0100    connected to: mongodb://localhost/
2022-03-05T12:44:49.242+0100    exported 794 records
```



```
| => head dumplondon.json
[{
        "_id": {
                "$oid": "5bd761dcae323e45a93ccfed"
        },
        "saleDate": {
                "$date": "2015-09-02T16:11:59.565Z"
        },
        "items": [
                {
                        "name": "binder",
```

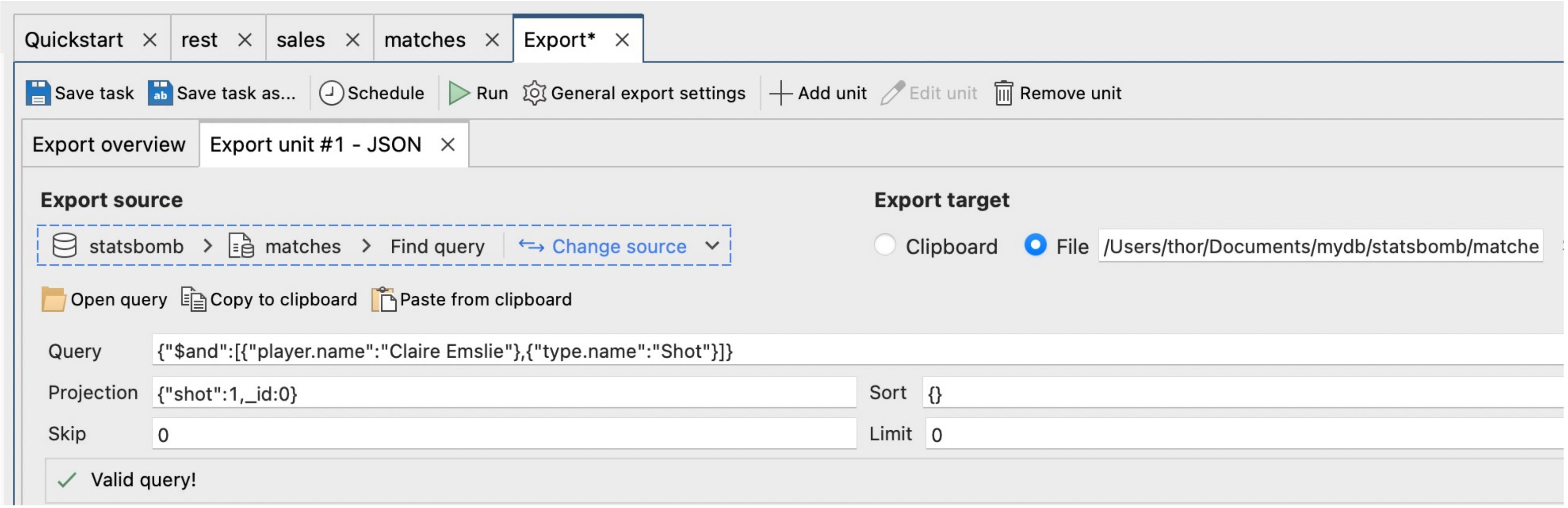
export data - mongodb tool



```
| => mongoexport -v --collection=sales --db=sales -q='{"storeLocation":"London"}' --pretty --jsonArray --out dumplondon.json
2022-03-05T12:44:49.136+0100    will listen for SIGTERM, SIGINT, and SIGKILL
2022-03-05T12:44:49.142+0100    connected to: mongodb://localhost/
2022-03-05T12:44:49.242+0100    exported 794 records
```

```
mongoexport --db=statsbomb --collection=matches --query='{"player.name":{ "$in": ["Claire Emslie","Lucy Graham"]}}' --out dump.json
```

Or GUI



Export source

statsbomb > matches > Find query Change source

Export target

○ Clipboard ● File /Users/thor/Documents/mydb/statsbomb/matche

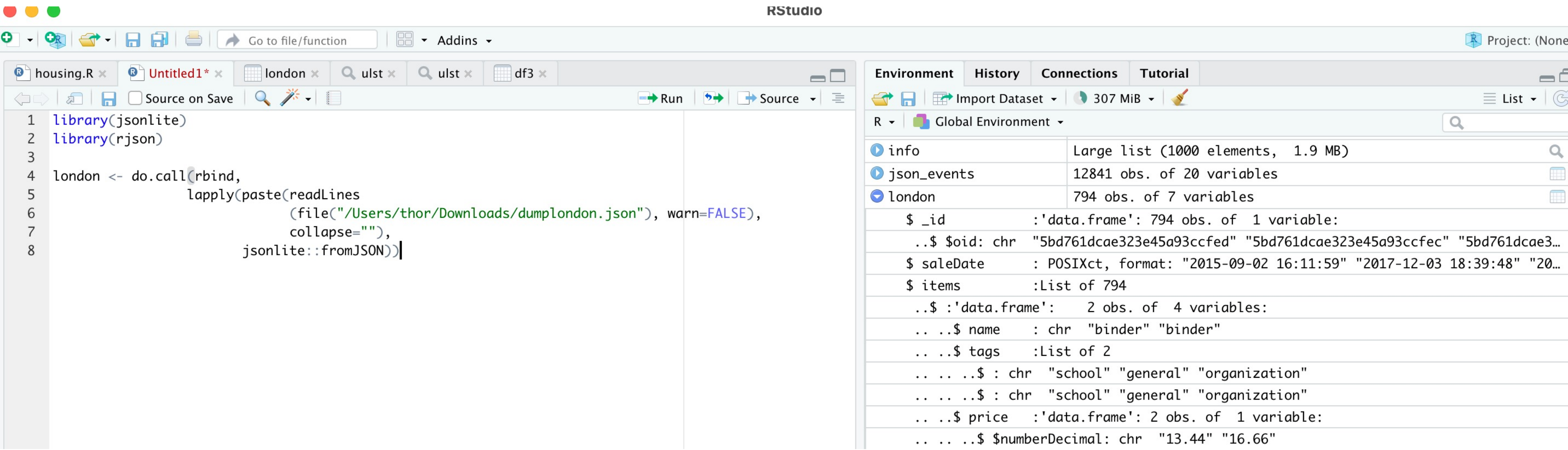
Query {"$and":[{"player.name":"Claire Emslie"},{"type.name":"Shot"}]}

Projection {"shot":1,_id:0} Sort {}

Skip 0 Limit 0

✓ Valid query!

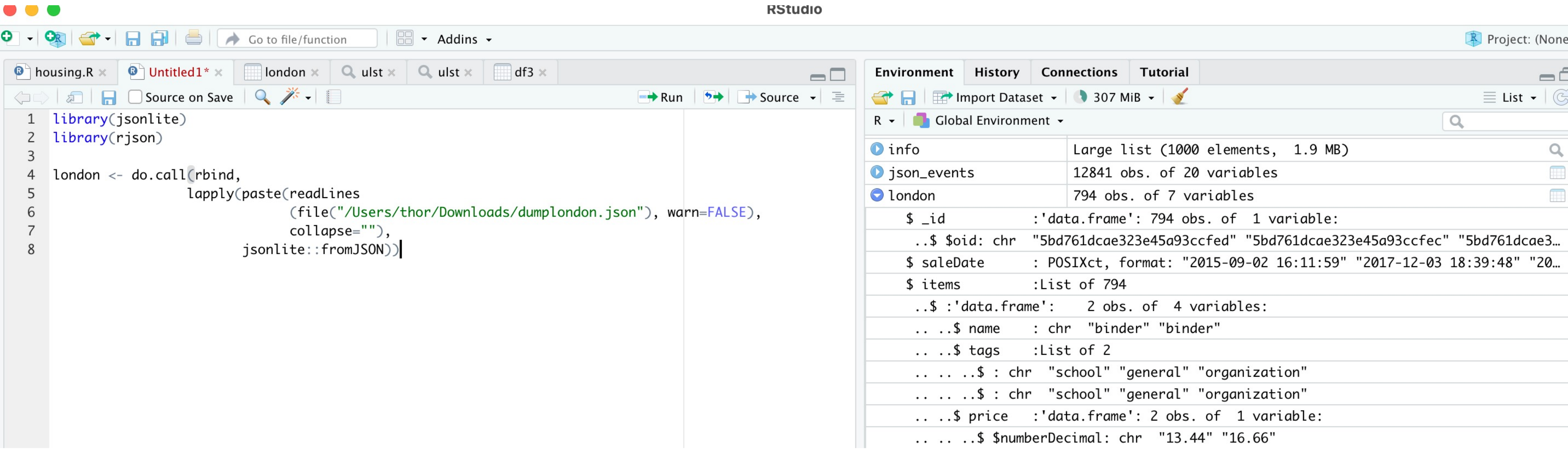
import data in R



```r
library(jsonlite)
library(rjson)

london <- do.call(rbind,
                 lapply(paste(readLines
                              (file("/Users/thor/Downloads/dumplondon.json"), warn=FALSE),
                              collapse=""),
                        jsonlite::fromJSON))
```

```
info            Large list (1000 elements,  1.9 MB)
json_events     12841 obs. of 20 variables
london          794 obs. of 7 variables
    $ _id         :'data.frame': 794 obs. of  1 variable:
     ..$ $oid: chr  "5bd761dcae323e45a93ccfed" "5bd761dcae323e45a93ccfec" "5bd761dcae3…
    $ saleDate    : POSIXct, format: "2015-09-02 16:11:59" "2017-12-03 18:39:48" "20…
    $ items       :List of 794
     ..$ :'data.frame':    2 obs. of  4 variables:
     .. ..$ name     : chr  "binder" "binder"
     .. ..$ tags     :List of 2
     .. .. ..$ : chr  "school" "general" "organization"
     .. .. ..$ : chr  "school" "general" "organization"
     .. ..$ price    :'data.frame': 2 obs. of  1 variable:
     .. .. ..$ $numberDecimal: chr  "13.44" "16.66"
```

import data in R

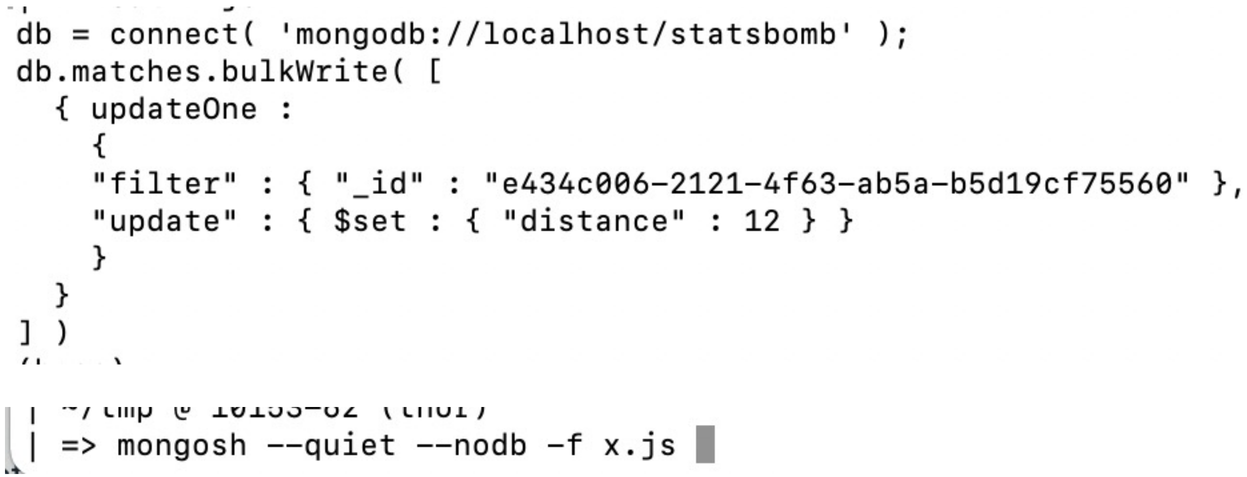


```r
library(jsonlite)
library(rjson)

london <- do.call(rbind,
                 lapply(paste(readLines
                              (file("/Users/thor/Downloads/dumplondon.json"), warn=FALSE),
                              collapse=""),
                        jsonlite::fromJSON))
```

```
info                 Large list (1000 elements,  1.9 MB)
json_events          12841 obs. of 20 variables
london               794 obs. of 7 variables
    $ _id           :'data.frame': 794 obs. of  1 variable:
     ..$ $oid: chr  "5bd761dcae323e45a93ccfed" "5bd761dcae323e45a93ccfec" "5bd761dcae3…
    $ saleDate      : POSIXct, format: "2015-09-02 16:11:59" "2017-12-03 18:39:48" "20…
    $ items         :List of 794
     ..$ :'data.frame':     2 obs. of  4 variables:
     .. ..$ name    : chr  "binder" "binder"
     .. ..$ tags    :List of 2
     .. .. ..$ : chr  "school" "general" "organization"
     .. .. ..$ : chr  "school" "general" "organization"
     .. ..$ price   :'data.frame': 2 obs. of  1 variable:
     .. .. ..$ $numberDecimal: chr  "13.44" "16.66"
```

Update MongoDB from Python

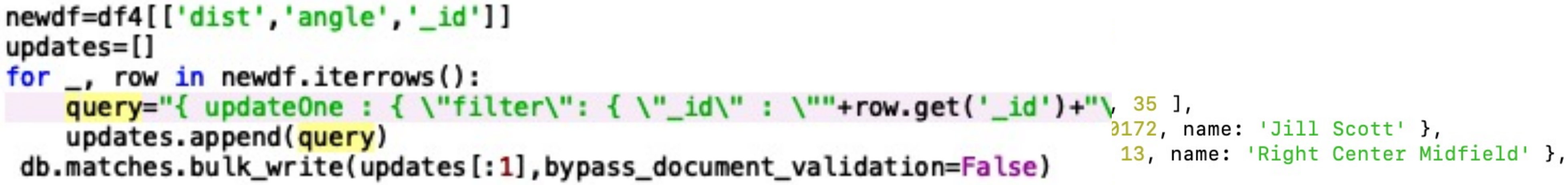
Terminal



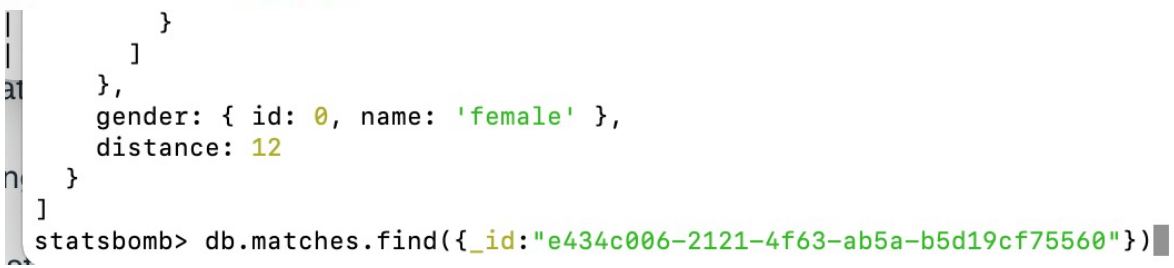
```
db = connect( 'mongodb://localhost/statsbomb' );
db.matches.bulkWrite( [
  { updateOne :
    {
    "filter" : { "_id" : "e434c006-2121-4f63-ab5a-b5d19cf75560" },
    "update" : { $set : { "distance" : 12 } }
    }
  }
] )
```

```
| => mongosh --quiet --nodb -f x.js
```

IDE



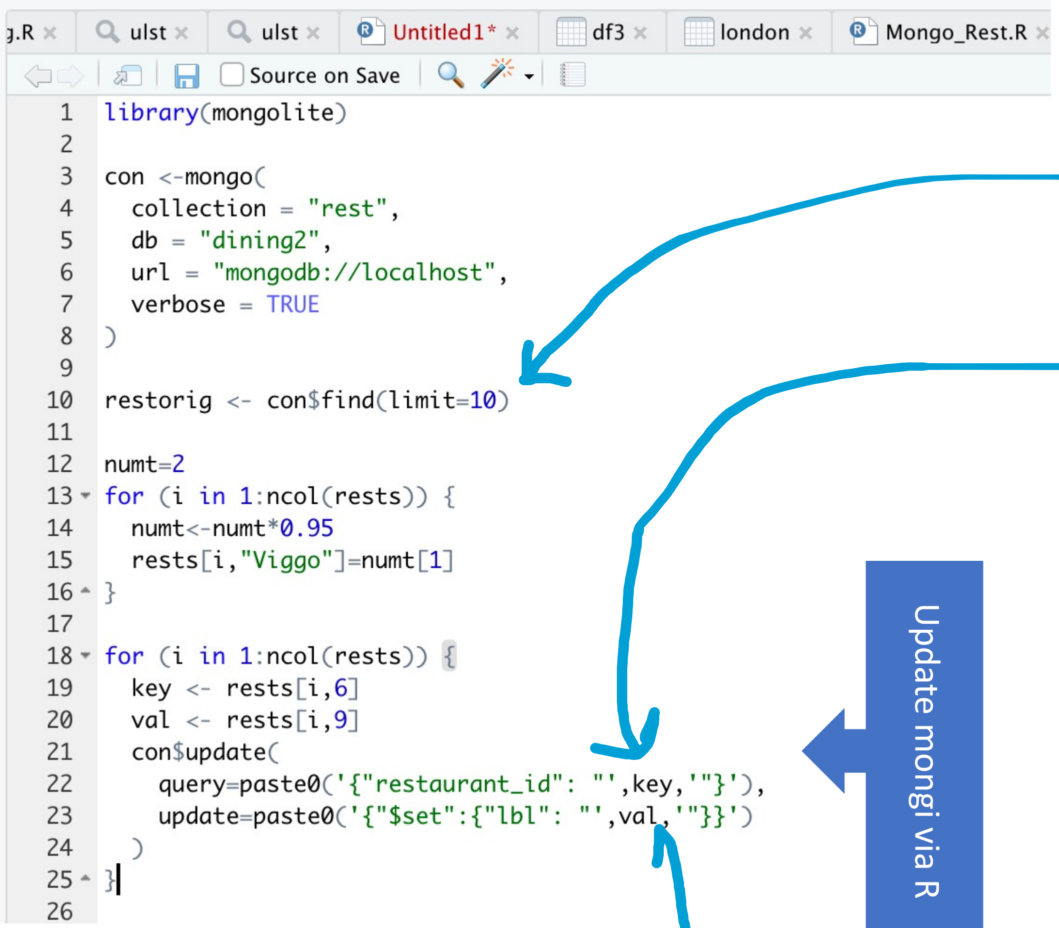
```
newdf=df4[['dist','angle','_id']]
updates=[]
for _, row in newdf.iterrows():
    query="{ updateOne : { \"filter\": { \"_id\" : \""+row.get('_id')+"\
    updates.append(query)
 db.matches.bulk_write(updates[:1],bypass_document_validation=False)
```



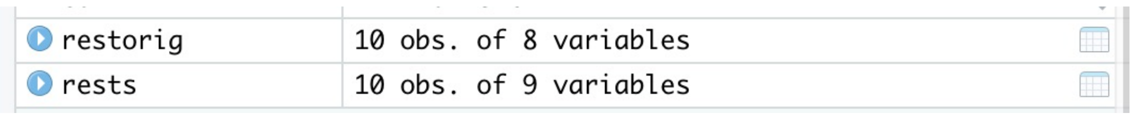
```
, 35 ],
0172, name: 'Jill Scott' },
 13, name: 'Right Center Midfield' },
          }
        ]
      },
      gender: { id: 0, name: 'female' },
      distance: 12
    }
  ]
statsbomb> db.matches.find({_id:"e434c006-2121-4f63-ab5a-b5d19cf75560"})
```

Update MongoDB from R

https://jeroen.github.io/mongolite/

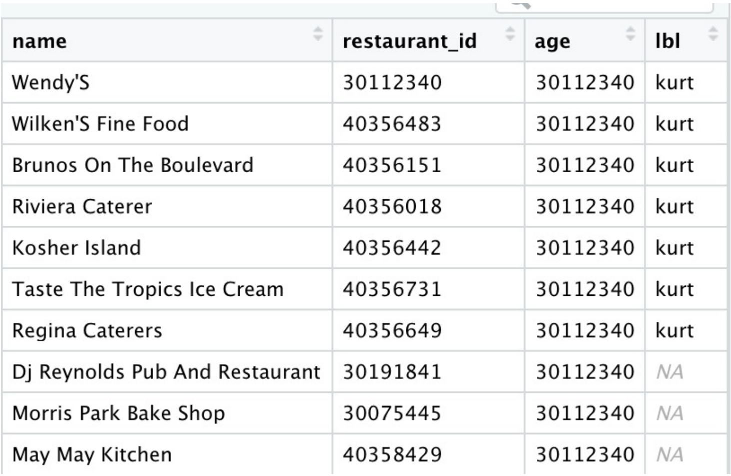


```
library(mongolite)

con <-mongo(
  collection = "rest",
  db = "dining2",
  url = "mongodb://localhost",
  verbose = TRUE
)

restorig <- con$find(limit=10)

numt=2
for (i in 1:ncol(rests)) {
  numt<-numt*0.95
  rests[i,"Viggo"]=numt[1]
}

for (i in 1:ncol(rests)) {
  key <- rests[i,6]
  val <- rests[i,9]
  con$update(
    query=paste0('{"restaurant_id": "',key,'"}'),
    update=paste0('{"$set":{"lbl": "',val,'"}}')
  )
}
```



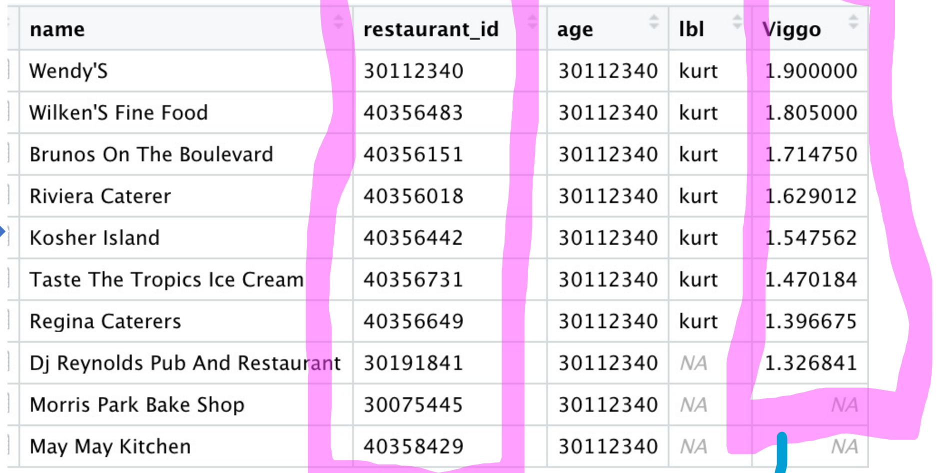
```
restorig   10 obs. of 8 variables
rests      10 obs. of 9 variables
```

From mongo



name	restaurant_id	age	lbl
Wendy'S	30112340	30112340	kurt
Wilken'S Fine Food	40356483	30112340	kurt
Brunos On The Boulevard	40356151	30112340	kurt
Riviera Caterer	40356018	30112340	kurt
Kosher Island	40356442	30112340	kurt
Taste The Tropics Ice Cream	40356731	30112340	kurt
Regina Caterers	40356649	30112340	kurt
Dj Reynolds Pub And Restaurant	30191841	30112340	NA
Morris Park Bake Shop	30075445	30112340	NA
May May Kitchen	40358429	30112340	NA

Modif dataframe



name	restaurant_id	age	lbl	Viggo
Wendy'S	30112340	30112340	kurt	1.900000
Wilken'S Fine Food	40356483	30112340	kurt	1.805000
Brunos On The Boulevard	40356151	30112340	kurt	1.714750
Riviera Caterer	40356018	30112340	kurt	1.629012
Kosher Island	40356442	30112340	kurt	1.547562
Taste The Tropics Ice Cream	40356731	30112340	kurt	1.470184
Regina Caterers	40356649	30112340	kurt	1.396675
Dj Reynolds Pub And Restaurant	30191841	30112340	NA	1.326841
Morris Park Bake Shop	30075445	30112340	NA	NA
May May Kitchen	40358429	30112340	NA	NA

Update mongi via R