MongoDB with R

Dr. Zahid Ansari

Table of Contents

1 Installing MongoDB 1
1.1 Install MongoDB Community Edition 1
1.2 Processing Big Data using MongoDB with R 2
1.3 Basic MongoDB commands and Importing data into MongoDB 3
1.3.1 Importing Data into MongoDB 3
2 MongoDB with R using the mongolite package 8

1 Installing MongoDB

setwd("D:/AMU Computer Science/Courses/Big Data Analytics/Big Data Analytics Using R/Ch6")

1.1 Install MongoDB Community Edition

- Download the installer: Download the MongoDB Community .msi installer from the following link: https://www.mongodb.com/try/download/community?tck=docs_server
- In the Version dropdown, select the version of MongoDB to download.
- In the Platform dropdown, select Windows.
- In the Package dropdown, select msi.
- Click Download.
- Run the MongoDB installer.
- For example, from the Windows Explorer/File Explorer: Go to the directory where you downloaded the MongoDB installer (.msi file). By default, this is your Downloads directory.
- Double-click the .msi file.
- Follow the MongoDB Community Edition installation wizard: The wizard steps you through the installation of MongoDB and MongoDB Compass.

- Choose Setup Type: You can choose either the Complete (recommended for most users) or Custom setup type. The Complete setup option installs MongoDB and the MongoDB tools to the default location. The Custom setup option allows you to specify which executables are installed and where.
- Service Configuration: Starting in MongoDB 4.0, you can set up MongoDB as a Windows service during the install or just install the binaries.

MongoDB Service: Starting in MongoDB 4.0, you can configure and start MongoDB as a Windows service during the install, and the MongoDB service is started upon successful installation.

- Select Install MongoD as a Service MongoDB as a service.
- Run the service as Network Service user (Default)

Install MongoDB Compass: To have the wizard install MongoDB Compass - select Install MongoDB Compass (Default). When ready, click Install.

- Once the installation is complete go to the start menu and invoke MongoDBCompass

1.2 Processing Big Data using MongoDB with R

- Once the preceding installations complete we may now configure our R environment by installing several R packages that will be used for carrying out data processing activities on a data stored and managed in MongoDB directly from the RStudio. These packages will include mongolite, rmongodb, and RMongo:

```
install.packages("mongolite")
install.packages("devtools")
library(devtools)
install.packages("rJava")
library(rJava)
install_github("tc/Rmongo")
library(devtools)
install_github("mongosoup/rmongodb")
```

- Also, we can install other R packages which may prove useful later (note that some of these packages have probably already been installed by you earlier.

```
install.packages(c("Rcpp", "RJSONIO", "bitops",
"digest", "functional", "stringr", "psych", "plyr", "reshape2", "caTools",
"R.methodsS3", "Hmisc", "memoise", "lazyeval", "rjson", "ggplot2",
"jsonlite", "data.table", "lubridate"), repos =
"http://cran.r-project.org/")
```

- The installation process should take a few minutes. After it completes, you should have MongoDB and RStudio Server fully configured and ready to use.
- Next we will discuss the essential steps of importing data into MongoDB and to practice querying and processing data using MongoDB shell commands.

1.3 Basic MongoDB commands and Importing data into MongoDB

- To run commands in mongosh, you must first connect to a MongoDB deployment.
- Go to the start menu and invoke MongoDBCompass
- Click on MONGOSH at the bottom left of MongoDBCompass window to get the MongoDB Shell command prompt.
- To show the avialable databases

```
show databases
```

admin 40.00 KiB config 108.00 KiB local 40.00 KiB

```
show dbs
```

admin 40.00 KiB config 108.00 KiB local 40.00 KiB

- To show the current database

```
db
```

<test

- To clear the screen

```
cls
```

1.3.1 Importing Data into MongoDB

- The data that we will be using in this tutorial is the Land Registry Price Paid Data, which is regularly published by the Land Registry in the United Kingdom and is available for public use under the Open Government Licence.
- According to the description of the original data downloadable from https://data.gov.uk/dataset/land-registry-monthly-price-paid-data.
- The Price Paid Data (PPD) contains prices of residential properties sold in England and Wales and other essential information about these properties such as their tenure (freehold/leasehold), transaction date, specific address of the property (for example street name, postcode, town, country, and other details), the type of the property for example whether a detached, terraced house or a flat, and several other variables.
- The original full data includes nearly 25 million records (as of April 2016) and covers the period from January 1995 until the current month. we will be using the PPD Data for the year 2015 only. It will provide us with a complex enough variety of variables and types of data.
- You can download the 2015 Price Paid Data from https://data.gov.uk/dataset/land-registry-monthly-price-paid-data.

- Once you download the data to a directory, in the new command window cd to the location where you store the key pair file
- I have downloaded file pp-2015.csv in the directory D:Computer ScienceData AnalyticsData Analytics Using R

```
C:\Users\ASUS>d:

D:\>cd D:\AMU Computer Science\Courses\Big Data Analytics\Big Data Analytics
Using R\Ch6

D:\AMU Computer Science\Courses\Big Data Analytics\Big Data Analytics Using
R\Ch6>
```

- Because the data file lacks any variable names, we have to provide them in a new file called pp-2015-variables.csv
- Once you have changed the directory to the location where these files reside, issue the following command to import the data to MongoDB

```
mongoimport mongodb://localhost:27017 --db=houses --collection=prices --
type=csv --fieldFile=pp-2015-variables.csv --file=pp-2015.csv

D:\AMU Computer Science\Courses\Big Data Analytics\Big Data Analytics Using
R\Ch6>mongoimport mongodb://localhost:27017 --db=houses --collection=prices -
-type=csv --fieldFile=pp-2015-variables.csv --file=pp-2015.csv
```

2022-10-18T11:54:50.491+0530 connected to: mongodb://localhost:27017 2022-10-18T11:54:53.491+0530 [##……..] houses.prices 11.8MB/108MB (10.9%) 2022-10-18T11:54:56.491+0530 [#####……..] houses.prices 24.6MB/108MB (22.7%) 2022-10-18T11:54:59.491+0530 [########……..] houses.prices 36.9MB/108MB (34.0%) 2022-10-18T11:55:02.492+0530 [##########……..] houses.prices 48.8MB/108MB (45.0%) 2022-10-18T11:55:05.492+0530 [#############……..] houses.prices 61.0MB/108MB (56.2%) 2022-10-18T11:55:08.492+0530 [################……..] houses.prices 72.6MB/108MB (67.0%) 2022-10-18T11:55:11.500+0530 [##################……] houses.prices 84.5MB/108MB (77.9%) 2022-10-18T11:55:14.493+0530 [####################…] houses.prices 95.6MB/108MB (88.1%) 2022-10-18T11:55:17.497+0530 [#######################.] houses.prices 108MB/108MB (99.3%) 2022-10-18T11:55:17.692+0530 [########################] houses.prices 108MB/108MB (100.0%) 2022-10-18T11:55:17.692+0530 1048576 document(s) imported successfully. 0 document(s) failed to import.

- The last line of the output confirms the final number of imported documents (1048576 for our data).
- Now we may log back in to MongoDB shell and check whether a new database named houses and its collection named prices have been successfully created:

```
show dbs
```

admin 40.00 KiB config 108.00 KiB houses 85.94 MiB local 40.00 KiB

- The result of the show dbs command confirms that a new database called houses has been created. We can indicate that we want to use this database in further operations:

```
use houses
```

'switched to db houses'

- We may also view the stored collections of documents within the houses databases by invoking the show collections statement:

```
show collections
```

prices

You can easily check the number of documents (records) in the collection using the following command:

```
db.prices.find().count()
```

1048576

- The preceding query is a standard MongoDB shell statement that contains the find() function that works just like the SELECT method in SQL, and it also includes the count() function – similar to the length() function from R, which simply calculates the number of rows in the data. The output confirms the total number of documents imported to MongoDB.

- Now, it's also a good time to practice a few more complex MongoDB queries. For example, let's see how the single entry of the 100th document looks like. In this case, we skip the first 99 records of the matching documents and will limit the results to only one record using the limit() function, which will in fact be our requested 100th document:

```
db.prices.find().skip(99).limit(1)
```

{ _id: ObjectId("634e4ca6c98d0df35bd059d3"), uniqueID: '{E05400EE-6336-4B26-B351-F2AF727BD56F}', price: 235000, transferDate: '07-04-2015 00:00', propType: 'T', oldNew: 'N', tenure: 'F', town: 'MANCHESTER', district: 'TRAFFORD', county: 'GREATER MANCHESTER', ppdCat: 'A', recordStatus: 'A' }

- As you can see from the preceding listing, the output includes an index variable called _id. It's a default MongoDB indexing field that is assigned to each document in a collection when the data is imported.

- Let's now calculate how many residential properties have been registered in the Land Registry database in 2015 in Manchester:

```
db.prices.find({town: "Manchester"}).count()
```

0

- Is it really true that there are no entries for Manchester? Note the spelling of the values in the town variable – MongoDB is case sensitive, so let's try MANCHESTER instead:

```
db.prices.find({town: "MANCHESTER"}).count()
```

17635

- We may want to aggregate average property prices for each county in England and Wales, and sort them in decreasing order from the most expensive counties to the most affordable ones.
- In this case we will use an aggregation pipeline framework for MongoDB:

```
db.prices.aggregate([ { $group : { _id: "$county", avgPrice: { $avg:"$price"
} } }, { $sort: { avgPrice: -1 } }])
```

{ _id: 'GREATER LONDON', avgPrice: 635741.7564671129 } { _id: 'WINDSOR AND MAIDENHEAD', avgPrice: 556088.639401655 } { _id: 'SURREY', avgPrice: 503970.73804979614 } { _id: 'WOKINGHAM', avgPrice: 466070.63935309974 } { _id: 'BUCKINGHAMSHIRE', avgPrice: 448583.92565454385 } { _id: 'HERTFORDSHIRE', avgPrice: 411555.09850531246 } { _id: 'OXFORDSHIRE', avgPrice: 393890.5836498761 } { _id: 'BRACKNELL FOREST', avgPrice: 386941.57501847745 } { _id: 'WEST BERKSHIRE', avgPrice: 385121.45817321906 } { _id: 'BRIGHTON AND HOVE', avgPrice: 364731.131314821 } { _id: 'READING', avgPrice: 358480.6139306059 } { _id: 'POOLE', avgPrice: 344365.6019098777 } { _id: 'BATH AND NORTH EAST SOMERSET', avgPrice: 341239.1913739669 } { _id: 'HAMPSHIRE', avgPrice: 335726.91895170184 } { _id: 'WEST SUSSEX', avgPrice: 334840.4510589582 } { _id: 'ISLES OF SCILLY', avgPrice: 333479 } { _id: 'CAMBRIDGESHIRE', avgPrice: 308173.9238509217 } { _id: 'ESSEX', avgPrice: 302634.5398758357 } { _id: 'MONMOUTHSHIRE', avgPrice: 300757.52004860266 } { _id: 'DORSET', avgPrice: 298633.3098070018 } Type "it" for more

- We could iterate the results further to include more affordable regions in England and Wales by simply typing the it command after the MongoDB prompt sign.

- After several iterations you will be presented with the most affordable counties as measured by the average residential property prices paid in 2015:

it { _id: 'SLOUGH', avgPrice: 293764.8151540383 } { _id: 'KENT', avgPrice: 291926.6504041194 } { _id: 'RUTLAND', avgPrice: 290318.29511918273 } { _id: 'CITY OF BRISTOL', avgPrice: 287382.347826087 } { _id: 'MILTON KEYNES', avgPrice: 285851.2949805585 } { _id: 'WILTSHIRE', avgPrice: 282944.2037572254 } { _id: 'CENTRAL BEDFORDSHIRE', avgPrice: 281238.22831561277 } { _id: 'GLOUCESTERSHIRE', avgPrice: 275383.51803547476 } { _id: 'EAST SUSSEX', avgPrice: 274674.25624599616 } { _id: 'SOUTH GLOUCESTERSHIRE', avgPrice: 269372.0038488454 } { _id: 'DEVON', avgPrice: 266546.9685523438 } { _id: 'SOUTHEND-ON-SEA', avgPrice: 264032.88650023116 } { _id: 'WARWICKSHIRE', avgPrice: 263823.2939704779 } { _id: 'YORK', avgPrice: 262932.38939475094 } { _id: 'BOURNEMOUTH', avgPrice: 258169.5081337551 } { _id: 'BEDFORD', avgPrice: 257781.59908186688 } { _id: 'NORTH SOMERSET', avgPrice:

255240.22301587302 } { _id: 'CHESHIRE EAST', avgPrice: 253245.87919858575 } { _id: 'SUFFOLK', avgPrice: 252178.2538806485 } { _id: 'THURROCK', avgPrice: 249698.37192883366 } Type "it" for more

- Finally, we will aggregate the property prices for each type of property and town name in Essex, and as before we will sort the results in descending order based on the average price paid. We will also limit the results to the top 10 highest averages only:

```
db.prices.aggregate([ { $match: { county: "ESSEX" } }, { $group : { _id: { town: "$town", propType: "$propType" }, avgPrice: { $avg: "$price" } } }, { $sort: { avgPrice: -1 } }, { $limit: 10 }])
```

{ _id: { town: 'LOUGHTON', propType: 'O' }, avgPrice: 2094992.375 } { _id: { town: 'CHELMSFORD', propType: 'O' }, avgPrice: 1909638.396551724 } { _id: { town: 'EPPING', propType: 'O' }, avgPrice: 1860545.7142857143 } { _id: { town: 'BRENTWOOD', propType: 'O' }, avgPrice: 1602123.4285714286 } { _id: { town: 'COLCHESTER', propType: 'O' }, avgPrice: 1355844.2333333334 } { _id: { town: 'WICKFORD', propType: 'O' }, avgPrice: 1350142.857142857 } { _id: { town: 'HOCKLEY', propType: 'O' }, avgPrice: 1239776.8 } { _id: { town: 'LONDON', propType: 'D' }, avgPrice: 1223998 } { _id: { town: 'MALDON', propType: 'O' }, avgPrice: 1070610 } { _id: { town: 'FRINTON-ON-SEA', propType: 'O' }, avgPrice: 1052118 }

- The most expensive residential properties in Essex are the ones with the property type set to "O", which denotes "Other".
- This category includes properties that are not officially recognized as detached, semi-detached, or terraced houses, flats, and maisonettes. They may then include some other unclassified properties (for example, hotels or farmhouses, and others).
- The inclusion of such properties can skew the results, so we will make sure that these properties are not taken into account by providing an exclusion clause in the redact method and adding it to the previously run query:

```
db.prices.aggregate([ { $match: { county: "ESSEX" } }, { $redact: { $cond: { if: { $eq: [ "$propType", "O" ] }, then: "$$PRUNE", else: "$$DESCEND" } } } ,{ $group : { _id: { town: "$town", propType: "$propType"}, avgPrice: { $avg: "$price" } } }, { $sort: { avgPrice: -1 } }, { $limit:10 }])
```

{ _id: { town: 'LONDON', propType: 'D' }, avgPrice: 1223998 } { _id: { town: 'CHIGWELL', propType: 'D' }, avgPrice: 1046256.3461538461 } { _id: { town: 'LOUGHTON', propType: 'D' }, avgPrice: 882813.5754716981 } { _id: { town: 'INGATESTONE', propType: 'D' }, avgPrice: 856240.3698630137 } { _id: { town: 'BUCKHURST HILL', propType: 'D' }, avgPrice: 847496.8378378379 } { _id: { town: 'EPPING', propType: 'D' }, avgPrice: 738859.5736434108 } { _id: { town: 'ROYSTON', propType: 'D' }, avgPrice: 705833.3333333334 } { _id: { town: 'ONGAR', propType: 'D' }, avgPrice: 689547.3684210526 } { _id: { town: 'BRENTWOOD', propType: 'D' }, avgPrice: 676699.3905472637 } { _id: { town: 'UPMINSTER', propType: 'D' }, avgPrice: 650000 }

- As you can see from the preceding examples, the queries can get extremely convoluted and sometimes very difficult to read and understand.

- However, the MongoDB aggregation pipeline is extremely powerful and fast – it can flexibly and quickly produce aggregations and calculations depending on the user's needs of Big Data within seconds.
- In the next section, we will attempt to perform similar crunching operations on the same data stored in MongoDB, but this time directly from the RStudio Server's console.

2 MongoDB with R using the mongolite package

- For the reasons stated at the end of the preceding section, there has been a strong need for a convenient, lightweight, and flexible R package that would offer a user-friendly interface for management and processing of data stored in MongoDB.
- The mongolite package, authored by Jeroen Ooms and MongoDB, Inc., fulfills this role very well. You can access its vignettes and all its help files through CRAN at https://cran.r-project.org/web/packages/mongolite/index.html, and the development versions are available at GitHub: http://github.com/jeroenooms/mongolite.
- As was the case with other MongoDB packages with R, we have already installed mongolite. We simply need to load it to prepare for the first use during the R session:

```
library(mongolite)
```

- It's always recommended to obtain some basic information about available functions before beginning to work with any specific R packages.
- It may be surprising to notice that, in fact, the mongolite package only contains one function called mongo().
- This however, allows users to apply a number of specific methods responsible for performing operations and queries on the data.
- In the first step, we need to create the usual connection to a specified database and collection on the local MongoDB:

```
m <- mongo(collection = "prices", db = "houses", url =
"mongodb://localhost:27017")
```

- The created connection object m displays all possible methods (and their arguments/parameters) of data processing using mongolite:

```
m

## <Mongo collection> 'prices'
##  $aggregate(pipeline = "{}", options = "{\"allowDiskUse\":true}", handler
= NULL, pagesize = 1000, iterate = FALSE)
##  $count(query = "{}")
##  $disconnect(gc = TRUE)
```

```
##  $distinct(key, query = "{}")
##  $drop()
##  $export(con = stdout(), bson = FALSE, query = "{}", fields = "{}", sort =
"{\"_id\":1}")
##  $find(query = "{}", fields = "{\"_id\":0}", sort = "{}", skip = 0, limit
= 0, handler = NULL, pagesize = 1000)
##  $import(con, bson = FALSE)
##  $index(add = NULL, remove = NULL)
##  $info()
##  $insert(data, pagesize = 1000, stop_on_error = TRUE, ...)
##  $iterate(query = "{}", fields = "{\"_id\":0}", sort = "{}", skip = 0,
limit = 0)
##  $mapreduce(map, reduce, query = "{}", sort = "{}", limit = 0, out = NULL,
scope = NULL)
##  $remove(query, just_one = FALSE)
##  $rename(name, db = NULL)
##  $replace(query, update = "{}", upsert = FALSE)
##  $run(command = "{\"ping\": 1}", simplify = TRUE)
##  $update(query, update = "{\"$set\":{}}", filters = NULL, upsert = FALSE,
multiple = FALSE)
```

- We may, for instance, begin from a simple calculation of the total number of documents in the collection, just like we did with other MongoDB-related R packages. We will achieve it by issuing the following command:

```
m$count()
```

```
## [1] 1048576
```

- This first example shows you how easy it is to use mongolite – It retrieves all essential information about the database and its collection from the connection object, thus simplifying the syntax and increasing the performance of the code.
- In order to query the data, we can use the find() method.
- From the previously shown output of the connection object, we can see that the find() method comes with some default parameters; for example, the indexing variable _id is suppressed in all result sets, the data is not sorted, and there is no limit on the amount of returned documents. Of course, we can simply override these defaults, which we are going to do later. However, for the time being, let's print all records for which the property paid price was lower than £100,000 and all residential properties of interest were detached houses:

```
subset1 <- m$find('{"price":{"$lt":100000}, "propType":{"$eq":"D"}}')
subset1
```

```
##                                  uniqueID price
## 1  {4CF5ED99-E50C-4C36-A81A-EF31FC33C5BE} 85000
## 2  {21E5FEB6-5E60-2439-E050-A8C06205342E} 21000
## 3  {21E5FEB6-6932-2439-E050-A8C06205342E} 75000
## 4  {25EA59FA-4BE4-4D50-E050-A8C0630562D0} 95000
## 5  {2757C4ED-502A-4CE1-B466-9401C258371F} 81000
## 6  {21E5FEB6-60C1-2439-E050-A8C06205342E} 90000
```

```
## 7  {25EA59FA-49AE-4D50-E050-A8C0630562D0} 33000
## 8  {25EA59FA-4C01-4D50-E050-A8C0630562D0} 97500
## 9  {31B0D3EC-8FA8-4525-8C63-940E5A028CD9} 95000
## 10 {21E5FEB6-7C54-2439-E050-A8C06205342E} 98000
## 11 {25EA59FA-47B9-4D50-E050-A8C0630562D0} 85000
## 12 {B33DAAAF-80A8-46A4-84A8-9B23332859DD} 70000
## 13 {25EA59FA-4C32-4D50-E050-A8C0630562D0} 66500
## 14 {21E5FEB6-6DA5-2439-E050-A8C06205342E} 85000
## 15 {25EA59FA-527B-4D50-E050-A8C0630562D0} 37500
## 16 {C3A8E2D0-72B0-4E28-935A-EF52F5758723} 65000
## 17 {ADBB4451-11E0-4049-A696-9B2D217E1B3F} 63000
## 18 {25EA59FA-47C0-4D50-E050-A8C0630562D0} 66000
## 19 {25EA59FA-47D1-4D50-E050-A8C0630562D0} 90000
## 20 {58E5E5A4-22BE-453E-893E-9EBDFA34E34C} 98500
## 21 {21E5FEB7-2182-2439-E050-A8C06205342E} 70000
## 22 {21E5FEB7-218B-2439-E050-A8C06205342E} 82500
## 23 {25EA59FA-56BC-4D50-E050-A8C0630562D0} 68000
## 24 {7B1294A1-A7DA-4E83-AF45-F6733BB065BF} 63000
## 25 {00EC04C2-AAD9-4D12-BABF-F9F3E0F7097F} 57500
## 26 {21E5FEB7-01EC-2439-E050-A8C06205342E} 65631
## 27 {25EA59FA-52B2-4D50-E050-A8C0630562D0} 95500
## 28 {35AA92EE-FAAA-4F79-AB42-EF6A861AED40} 76000
## 29 {31FE19D1-4A88-44C1-B8A9-A264AE1F3799} 40000
## 30 {21E5FEB7-3C94-2439-E050-A8C06205342E} 85000
## 31 {21E5FEB7-4EDC-2439-E050-A8C06205342E} 95000
## 32 {21E5FEB6-FD4C-2439-E050-A8C06205342E} 95750
## 33 {25EA59FA-4816-4D50-E050-A8C0630562D0} 96000
## 34 {21E5FEB7-4112-2439-E050-A8C06205342E} 35000
## 35 {25EA59FA-4845-4D50-E050-A8C0630562D0} 82000
## 36 {3F7823F3-96C2-4AB5-B98A-F302916A88B1} 20000
## 37 {21E5FEB7-3CBC-2439-E050-A8C06205342E} 72000
## 38 {79E63A8B-D807-467B-B261-FD9758D20CED} 81000
## 39 {21E5FEB7-4141-2439-E050-A8C06205342E} 86000
## 40 {21E5FEB7-414F-2439-E050-A8C06205342E} 78000
## 41 {25EA59FA-4A7F-4D50-E050-A8C0630562D0} 78000
## 42 {2790F32F-C318-47AE-8D41-97CD489559D4} 68000
## 43 {3943A792-D25B-4371-8892-9EF137D84139} 90000
## 44 {9B3B0D1A-80ED-4EC8-B2A7-FA1C31B4B2E0} 98000
## 45 {21E5FEB7-3ABA-2439-E050-A8C06205342E} 90969
## 46 {D8E1AE03-62A0-4218-8384-A28FDD4AF0CF} 51000
## 47 {25EA59FA-4AAE-4D50-E050-A8C0630562D0} 65000
## 48 {25EA59FA-4ABD-4D50-E050-A8C0630562D0} 90000
## 49 {21E5FEB7-1F6E-2439-E050-A8C06205342E} 70000
## 50 {3F51C1B0-2274-4AFB-A8DB-9464DAD42C65} 85000
## 51 {21E5FEB7-43B0-2439-E050-A8C06205342E} 99950
## 52 {8BC9F001-4900-4308-85DF-F6B29382B2A1} 87750
## 53 {21E5FEB7-45E0-2439-E050-A8C06205342E} 80000
## 54 {25EA59FA-48A6-4D50-E050-A8C0630562D0} 97000
## 55 {25EA59FA-48A8-4D50-E050-A8C0630562D0} 85000
## 56 {21E5FEB7-02EE-2439-E050-A8C06205342E} 70500
```

```
## 57 {4BE1FE75-1AAB-4353-AC90-97E7EE57C8EB} 60000
## 58 {21E5FEB7-05F1-2439-E050-A8C06205342E} 48888
## 59 {C5867684-4360-4CCF-AF03-9B7D25C0BEAC} 81000
## 60 {21E5FEB7-1F86-2439-E050-A8C06205342E} 70000
## 61 {05091FF5-9CBE-4C4B-BD02-9F0DCE97614E} 97000
## 62 {21E5FEB7-3F60-2439-E050-A8C06205342E} 96100
## 63 {F8BEA5F4-3870-44A5-A4A3-9B819AB6583B} 98000
## 64 {21E5FEB7-43E0-2439-E050-A8C06205342E} 69500
## 65 {21E5FEB7-4CBF-2439-E050-A8C06205342E} 70000
## 66 {336955E1-5889-46F9-9309-97F1E5F5ACAA} 99777
## 67 {21E5FEB6-FFE1-2439-E050-A8C06205342E} 80000
## 68 {21E5FEB7-0627-2439-E050-A8C06205342E} 85000
## 69 {21E5FEB7-1FB6-2439-E050-A8C06205342E} 60000
## 70 {21E5FEB7-3D49-2439-E050-A8C06205342E} 82500
## 71 {35D7980E-40FB-4F33-942E-A2B603D0A4FE} 70000
## 72 {21E5FEB7-4CE2-2439-E050-A8C06205342E} 45000
## 73 {25EA59FA-4B1C-4D50-E050-A8C0630562D0} 80000
## 74 {C7939E1E-312C-42F5-A6DC-97FC6F1064BE} 80000
## 75 {7252FB8C-D350-4165-B9DB-FA4B9CAFB085} 70000
## 76 {10BBDB78-36C0-4FD0-87A8-F343F819A6F4} 85500
## 77 {21E5FEB7-41F6-2439-E050-A8C06205342E} 55000
## 78 {21E5FEB7-441B-2439-E050-A8C06205342E} 64000
## 79 {21E5FEB7-464D-2439-E050-A8C06205342E} 85000
## 80 {25EA59FA-4B40-4D50-E050-A8C0630562D0} 91500
## 81 {21E5FEB6-F1AE-2439-E050-A8C06205342E} 67000
## 82 {21E5FEB7-1C6D-2439-E050-A8C06205342E} 85000
## 83 {21E5FEB7-3B72-2439-E050-A8C06205342E} 72000
## 84 {21E5FEB7-3FCA-2439-E050-A8C06205342E} 71113
## 85 {25EA59FA-53D9-4D50-E050-A8C0630562D0} 90000
## 86 {D9B28ECB-B35D-4070-B397-949DE3121CF2} 85000
## 87 {F2E1E8DC-83E6-4151-8BD8-9BB17ECBBBC1} 58000
## 88 {21E5FEB7-3DD8-2439-E050-A8C06205342E} 63000
## 89 {C97A73AE-FC59-4CD5-935F-981E5A6CAD95} 89950
## 90 {21E5FEB7-445A-2439-E050-A8C06205342E} 90000
##        transferDate propType oldNew tenure
## 1  26-06-2015 00:00        D      N      F
## 2  22-04-2015 00:00        D      N      L
## 3  07-05-2015 00:00        D      N      F
## 4  20-11-2015 00:00        D      N      F
## 5  09-07-2015 00:00        D      N      F
## 6  22-04-2015 00:00        D      N      F
## 7  02-10-2015 00:00        D      N      L
## 8  30-10-2015 00:00        D      N      F
## 9  18-05-2015 00:00        D      N      F
## 10 15-07-2015 00:00        D      N      F
## 11 06-11-2015 00:00        D      N      F
## 12 26-08-2015 00:00        D      N      F
## 13 20-10-2015 00:00        D      N      F
## 14 22-04-2015 00:00        D      N      F
## 15 06-11-2015 00:00        D      N      F
```

```
## 16 12-05-2015 00:00        D      N      F
## 17 05-03-2015 00:00        D      N      F
## 18 16-10-2015 00:00        D      N      F
## 19 06-10-2015 00:00        D      N      F
## 20 21-08-2015 00:00        D      N      F
## 21 27-02-2015 00:00        D      N      F
## 22 27-02-2015 00:00        D      N      F
## 23 10-11-2015 00:00        D      N      F
## 24 05-01-2015 00:00        D      N      F
## 25 20-02-2015 00:00        D      N      F
## 26 23-01-2015 00:00        D      N      F
## 27 16-10-2015 00:00        D      N      F
## 28 26-08-2015 00:00        D      N      F
## 29 02-04-2015 00:00        D      N      F
## 30 12-06-2015 00:00        D      N      F
## 31 12-03-2015 00:00        D      N      F
## 32 06-01-2015 00:00        D      N      F
## 33 16-11-2015 00:00        D      N      F
## 34 09-03-2015 00:00        D      N      F
## 35 13-11-2015 00:00        D      N      F
## 36 07-05-2015 00:00        D      N      F
## 37 31-03-2015 00:00        D      N      F
## 38 24-07-2015 00:00        D      N      F
## 39 12-06-2015 00:00        D      N      F
## 40 18-06-2015 00:00        D      N      F
## 41 09-10-2015 00:00        D      N      F
## 42 20-01-2015 00:00        D      N      F
## 43 05-03-2015 00:00        D      N      F
## 44 02-02-2015 00:00        D      N      F
## 45 13-03-2015 00:00        D      N      F
## 46 08-05-2015 00:00        D      N      F
## 47 12-11-2015 00:00        D      N      F
## 48 19-10-2015 00:00        D      N      F
## 49 24-02-2015 00:00        D      N      F
## 50 12-06-2015 00:00        D      N      F
## 51 01-05-2015 00:00        D      N      F
## 52 23-03-2015 00:00        D      N      F
## 53 29-05-2015 00:00        D      N      F
## 54 17-11-2015 00:00        D      N      F
## 55 09-11-2015 00:00        D      N      F
## 56 11-02-2015 00:00        D      N      F
## 57 06-08-2015 00:00        D      N      F
## 58 29-01-2015 00:00        D      Y      F
## 59 16-01-2015 00:00        D      N      F
## 60 22-01-2015 00:00        D      Y      F
## 61 28-07-2015 00:00        D      N      F
## 62 09-04-2015 00:00        D      Y      F
## 63 19-01-2015 00:00        D      N      F
## 64 15-06-2015 00:00        D      N      F
## 65 18-02-2015 00:00        D      N      F
```

```
## 66 12-06-2015 00:00        D      N      F
## 67 06-02-2015 00:00        D      N      F
## 68 13-01-2015 00:00        D      N      F
## 69 26-02-2015 00:00        D      N      F
## 70 06-03-2015 00:00        D      N      L
## 71 16-09-2015 00:00        D      N      F
## 72 18-06-2015 00:00        D      N      F
## 73 03-09-2015 00:00        D      N      F
## 74 19-02-2015 00:00        D      N      F
## 75 30-04-2015 00:00        D      N      F
## 76 06-02-2015 00:00        D      N      F
## 77 18-06-2015 00:00        D      N      F
## 78 17-06-2015 00:00        D      N      F
## 79 30-06-2015 00:00        D      N      F
## 80 16-11-2015 00:00        D      N      F
## 81 15-01-2015 00:00        D      N      F
## 82 09-01-2015 00:00        D      N      F
## 83 25-03-2015 00:00        D      N      F
## 84 30-04-2015 00:00        D      Y      F
## 85 11-11-2015 00:00        D      N      F
## 86 16-03-2015 00:00        D      N      F
## 87 12-02-2015 00:00        D      N      F
## 88 17-03-2015 00:00        D      N      F
## 89 16-07-2015 00:00        D      N      F
## 90 18-05-2015 00:00        D      N      F
##                   town                  district
## 1                GOOLE  EAST RIDING OF YORKSHIRE
## 2          MABLETHORPE              EAST LINDSEY
## 3               HEXHAM           NORTHUMBERLAND
## 4              SWANSEA                  SWANSEA
## 5              GAERWEN         ISLE OF ANGLESEY
## 6               BOSTON                   BOSTON
## 7            CAERNARFON                  GWYNEDD
## 8              BRIDGEND                 BRIDGEND
## 9    LLANDRINDOD WELLS                    POWYS
## 10             BRIDGEND        RHONDDA CYNON TAFF
## 11           TREHARRIS            MERTHYR TYDFIL
## 12              WIGTON                ALLERDALE
## 13           EBBW VALE             BLAENAU GWENT
## 14          SCUNTHORPE        NORTH LINCOLNSHIRE
## 15       BRIERLEY HILL                   DUDLEY
## 16       HAVERFORDWEST             PEMBROKESHIRE
## 17          SCUNTHORPE        NORTH LINCOLNSHIRE
## 18             SWANSEA                  SWANSEA
## 19           EBBW VALE             BLAENAU GWENT
## 20             BURNLEY                  BURNLEY
## 21           AMMANFORD           CARMARTHENSHIRE
## 22          HORNCASTLE              EAST LINDSEY
## 23           HALESOWEN                   DUDLEY
## 24             PRESCOT                 KNOWSLEY
```

```
## 25       WOLVERHAMPTON             WOLVERHAMPTON
## 26             LINCOLN            NORTH KESTEVEN
## 27             BILSTON                    DUDLEY
## 28                RHYL                     CONWY
## 29          ROSSENDALE                ROSSENDALE
## 30              BOSTON                    BOSTON
## 31           ALFRETON                  BOLSOVER
## 32         BOURNEMOUTH                BOURNEMOUTH
## 33            LLANELLI           CARMARTHENSHIRE
## 34               LEEDS                     LEEDS
## 35                RHYL               DENBIGHSHIRE
## 36              DURHAM              COUNTY DURHAM
## 37            SPALDING              SOUTH HOLLAND
## 38          NOTTINGHAM        CITY OF NOTTINGHAM
## 39              DUDLEY                    DUDLEY
## 40      STOKE-ON-TRENT             STOKE-ON-TRENT
## 41                RHYL               DENBIGHSHIRE
## 42              HEXHAM             NORTHUMBERLAND
## 43          PRESTEIGNE                     POWYS
## 44            BRADFORD                   BRADFORD
## 45             WREXHAM                    WREXHAM
## 46            EGREMONT                   COPELAND
## 47             NEWTOWN                      POWYS
## 48             SWANSEA                    SWANSEA
## 49               DERBY              CITY OF DERBY
## 50   LLANDRINDOD WELLS                     POWYS
## 51              REDCAR      REDCAR AND CLEVELAND
## 52  BARTON-UPON-HUMBER        NORTH LINCOLNSHIRE
## 53            SLEAFORD            NORTH KESTEVEN
## 54                RHYL               DENBIGHSHIRE
## 55           PONTYPOOL                    TORFAEN
## 56         SWADLINCOTE          SOUTH DERBYSHIRE
## 57               NEATH         NEATH PORT TALBOT
## 58             SWANSEA                    SWANSEA
## 59             HORNSEA  EAST RIDING OF YORKSHIRE
## 60         STOURBRIDGE                    DUDLEY
## 61            ALFRETON              AMBER VALLEY
## 62                YORK                     SELBY
## 63                RHYL                     CONWY
## 64             BILSTON             WOLVERHAMPTON
## 65 NEWCASTLE UPON TYNE            NORTH TYNESIDE
## 66     BURY ST EDMUNDS            ST EDMUNDSBURY
## 67             GRANTHAM            SOUTH KESTEVEN
## 68             CANNOCK              CANNOCK CHASE
## 69           HALESOWEN                    DUDLEY
## 70              DURHAM              COUNTY DURHAM
## 71             CHESTER CHESHIRE WEST AND CHESTER
## 72      CLACTON-ON-SEA                  TENDRING
## 73             SWANSEA                    SWANSEA
## 74          NOTTINGHAM        CITY OF NOTTINGHAM
```

```
## 75  SUTTON-IN-ASHFIELD                  ASHFIELD
## 76            BARNSLEY                  BARNSLEY
## 77      CLACTON-ON-SEA                  TENDRING
## 78        HECKMONDWIKE                  KIRKLEES
## 79          PONTYPRIDD        RHONDDA CYNON TAFF
## 80            ABERGELE                     CONWY
## 81             BILSTON                    DUDLEY
## 82              LONDON            WALTHAM FOREST
## 83            BRADFORD                  BRADFORD
## 84               LEEDS                     LEEDS
## 85             WALSALL                   WALSALL
## 86                RHYL                     CONWY
## 87            DEWSBURY                  KIRKLEES
## 88             SWANSEA         NEATH PORT TALBOT
## 89                RHYL                     CONWY
## 90            WINSFORD CHESHIRE WEST AND CHESTER
##                        county ppdCat recordStatus
## 1   EAST RIDING OF YORKSHIRE      A            A
## 2               LINCOLNSHIRE      B            A
## 3             NORTHUMBERLAND      B            A
## 4                    SWANSEA      A            A
## 5           ISLE OF ANGLESEY      A            A
## 6               LINCOLNSHIRE      B            A
## 7                    GWYNEDD      A            A
## 8                   BRIDGEND      A            A
## 9                     POWYS      A            A
## 10        RHONDDA CYNON TAFF      B            A
## 11            MERTHYR TYDFIL      A            A
## 12                   CUMBRIA      A            A
## 13             BLAENAU GWENT      A            A
## 14        NORTH LINCOLNSHIRE      B            A
## 15             WEST MIDLANDS      A            A
## 16             PEMBROKESHIRE      A            A
## 17        NORTH LINCOLNSHIRE      A            A
## 18                   SWANSEA      A            A
## 19             BLAENAU GWENT      B            A
## 20                LANCASHIRE      A            A
## 21           CARMARTHENSHIRE      B            A
## 22              LINCOLNSHIRE      B            A
## 23             WEST MIDLANDS      A            A
## 24                MERSEYSIDE      A            A
## 25             WEST MIDLANDS      A            A
## 26              LINCOLNSHIRE      B            A
## 27             WEST MIDLANDS      A            A
## 28                     CONWY      A            A
## 29                LANCASHIRE      A            A
## 30              LINCOLNSHIRE      B            A
## 31                DERBYSHIRE      B            A
## 32               BOURNEMOUTH      B            A
## 33           CARMARTHENSHIRE      A            A
```

```
## 34            WEST YORKSHIRE      B            A
## 35              DENBIGHSHIRE      A            A
## 36             COUNTY DURHAM      A            A
## 37              LINCOLNSHIRE      B            A
## 38        CITY OF NOTTINGHAM      A            A
## 39             WEST MIDLANDS      B            A
## 40            STOKE-ON-TRENT      B            A
## 41              DENBIGHSHIRE      A            A
## 42            NORTHUMBERLAND      A            A
## 43                      POWYS     A            A
## 44            WEST YORKSHIRE      A            A
## 45                    WREXHAM     B            A
## 46                    CUMBRIA     A            A
## 47                      POWYS     A            A
## 48                    SWANSEA     A            A
## 49             CITY OF DERBY      B            A
## 50                      POWYS     A            A
## 51      REDCAR AND CLEVELAND      B            A
## 52        NORTH LINCOLNSHIRE      A            A
## 53              LINCOLNSHIRE      B            A
## 54              DENBIGHSHIRE      A            A
## 55                    TORFAEN     A            A
## 56               DERBYSHIRE       B            A
## 57         NEATH PORT TALBOT      A            A
## 58                    SWANSEA     B            A
## 59  EAST RIDING OF YORKSHIRE      A            A
## 60            WEST MIDLANDS       B            A
## 61               DERBYSHIRE       A            A
## 62           NORTH YORKSHIRE      B            A
## 63                      CONWY     A            A
## 64            WEST MIDLANDS       B            A
## 65            TYNE AND WEAR       B            A
## 66                    SUFFOLK     A            A
## 67             LINCOLNSHIRE       B            A
## 68            STAFFORDSHIRE       B            A
## 69            WEST MIDLANDS       B            A
## 70            COUNTY DURHAM       B            A
## 71 CHESHIRE WEST AND CHESTER      A            A
## 72                      ESSEX     B            A
## 73                    SWANSEA     A            A
## 74        CITY OF NOTTINGHAM      A            A
## 75           NOTTINGHAMSHIRE      A            A
## 76           SOUTH YORKSHIRE      A            A
## 77                      ESSEX     B            A
## 78            WEST YORKSHIRE      B            A
## 79        RHONDDA CYNON TAFF      B            A
## 80                      CONWY     A            A
## 81             WEST MIDLANDS      B            A
## 82            GREATER LONDON      B            A
## 83            WEST YORKSHIRE      B            A
```

```
## 84            WEST YORKSHIRE      B            A
## 85             WEST MIDLANDS      A            A
## 86                     CONWY      A            A
## 87            WEST YORKSHIRE      A            A
## 88         NEATH PORT TALBOT      B            A
## 89                     CONWY      A            A
## 90 CHESHIRE WEST AND CHESTER      B            A
##  [ reached 'max' / getOption("max.print") -- omitted 1736 rows ]
```

- While creating the output, the mongolite package provides the user with very handy information on the total number of returned documents. It also informs us that the output will be automatically simplified into a data.frame object. This is a very useful functionality that definitely saves a bit of processing time.
- We can now inspect the structure of the resulting object using the standard str() command:

```
str(subset1)
```

```
## 'data.frame':    1826 obs. of  11 variables:
##  $ uniqueID    : chr  "{4CF5ED99-E50C-4C36-A81A-EF31FC33C5BE}" "{21E5FEB6-
5E60-2439-E050-A8C06205342E}" "{21E5FEB6-6932-2439-E050-A8C06205342E}"
"{25EA59FA-4BE4-4D50-E050-A8C0630562D0}" ...
##  $ price       : int  85000 21000 75000 95000 81000 90000 33000 97500
95000 98000 ...
##  $ transferDate: chr  "26-06-2015 00:00" "22-04-2015 00:00" "07-05-2015
00:00" "20-11-2015 00:00" ...
##  $ propType    : chr  "D" "D" "D" "D" ...
##  $ oldNew      : chr  "N" "N" "N" "N" ...
##  $ tenure      : chr  "F" "L" "F" "F" ...
##  $ town        : chr  "GOOLE" "MABLETHORPE" "HEXHAM" "SWANSEA" ...
##  $ district    : chr  "EAST RIDING OF YORKSHIRE" "EAST LINDSEY"
"NORTHUMBERLAND" "SWANSEA" ...
##  $ county      : chr  "EAST RIDING OF YORKSHIRE" "LINCOLNSHIRE"
"NORTHUMBERLAND" "SWANSEA" ...
##  $ ppdCat      : chr  "A" "B" "B" "A" ...
##  $ recordStatus: chr  "A" "A" "A" "A" ...
```

- Of course, when dealing with large datasets it's unlikely that you would want to retrieve all the variables of the data.
- But in mongolite you can easily specify which fields you would like to include in the results set. Apart from projections, you may also use other commands known from the MongoDB shell, for example, sort, skip, or limit. The good thing about how these other methods are implemented in the find query in mongolite is that you don't have to set them using the JSON format, they simply work in the same way as parameters of any R function. - The only exception to that rule is the sort parameter, which takes a short JSON entry to define the variable for which data is going to be sorted and the direction of sorting.

- In the following example we will return all documents with detached properties and prices lower than £100,000. We will, however, only include the price and town fields in the results set and we will order the prices from the most expensive to the cheapest. Just in case, we will limit the output to the first 10,000 matched documents:
- If you are working with extremely large datasets, it is a good practice to limit the output of result sets to a small amount of documents, especially during the early stages of code testing. Once you are sure that your code produces the desired output you may increase the size of the returned objects depending on your available resources.

```
subset2 <- m$find('{"price":{"$lt":100000}, "propType":{"$eq":"D"}}',
                  fields = '{"_id":0, "price":1, "town":1}',
                  sort = '{"price":-1}', skip = 0, limit = 10000)
str(subset2)

## 'data.frame':    1826 obs. of  2 variables:
##  $ price: int  99995 99995 99995 99995 99995 99995 99995 99955 99950 99950
...
##  $ town : chr  "WISBECH" "WISBECH" "SPALDING" "HARTLEPOOL" ...

head(subset2, n=5)

##   price           town
## 1 99995        WISBECH
## 2 99995        WISBECH
## 3 99995       SPALDING
## 4 99995     HARTLEPOOL
## 5 99995 CLACTON-ON-SEA
```

- By restricting the structure of the final output to just two variables of interest, we have essentially decreased the size of the returned object from 650.4 KB in subset1 to as little as 43.6 KB in subset2.
- In mongolite, we can also perform typical MongoDB-style aggregations using the aggregate() method. Here, we need to pass the full aggregation pipeline in JSON format.
- The following example calculates two basic statistics: the number of records (a new count variable will be created) and the average price for all properties in each of the 112 counties in England and Wales (a new avgPrice variable will be created). The sorted results set in the form of a data.frame can be obtained as follows:

```
houses.agr <- m$aggregate('[{"$group": {"_id":"$county", "count":{"$sum":1},
"avgPrice":{"$avg":"$price"} }}, {"$sort":{"avgPrice": -1} }]')
head(houses.agr, n=10)

##                         _id  count avgPrice
## 1          GREATER LONDON 133715 635741.8
## 2  WINDSOR AND MAIDENHEAD   3142 556088.6
## 3                  SURREY  23054 503970.7
```

```
## 4               WOKINGHAM   3710 466070.6
## 5         BUCKINGHAMSHIRE  11191 448583.9
## 6           HERTFORDSHIRE  22212 411555.1
## 7             OXFORDSHIRE  12110 393890.6
## 8        BRACKNELL FOREST   2706 386941.6
## 9          WEST BERKSHIRE   3383 385121.5
## 10      BRIGHTON AND HOVE   5978 364731.1
```

- As expected, the aggregation returned 112 records in total. From the results, we clearly see, that the most expensive properties are located in Greater London, followed by the counties of Windsor and Maidenhead, and Surrey – all stereotypically associated with upper-class residents and (less stereotypically) high living costs. Very often, when dealing with categorical variables, it is useful to view all possible values that are contained within such variables. This can be achieved in mongolite by the distinct() method. For example, if you wish to list all the unique values for two fields of county and propType, you can do it as follows:

```
m$distinct("county")
```

```
##   [1] "BATH AND NORTH EAST SOMERSET"
##   [2] "BEDFORD"
##   [3] "BLACKBURN WITH DARWEN"
##   [4] "BLACKPOOL"
##   [5] "BLAENAU GWENT"
##   [6] "BOURNEMOUTH"
##   [7] "BRACKNELL FOREST"
##   [8] "BRIDGEND"
##   [9] "BRIGHTON AND HOVE"
##  [10] "BUCKINGHAMSHIRE"
##  [11] "CAERPHILLY"
##  [12] "CAMBRIDGESHIRE"
##  [13] "CARDIFF"
##  [14] "CARMARTHENSHIRE"
##  [15] "CENTRAL BEDFORDSHIRE"
##  [16] "CEREDIGION"
##  [17] "CHESHIRE EAST"
##  [18] "CHESHIRE WEST AND CHESTER"
##  [19] "CITY OF BRISTOL"
##  [20] "CITY OF DERBY"
##  [21] "CITY OF KINGSTON UPON HULL"
##  [22] "CITY OF NOTTINGHAM"
##  [23] "CITY OF PETERBOROUGH"
##  [24] "CITY OF PLYMOUTH"
##  [25] "CONWY"
##  [26] "CORNWALL"
##  [27] "COUNTY DURHAM"
##  [28] "CUMBRIA"
##  [29] "DARLINGTON"
##  [30] "DENBIGHSHIRE"
##  [31] "DERBYSHIRE"
```

```
##  [32] "DEVON"
##  [33] "DORSET"
##  [34] "EAST RIDING OF YORKSHIRE"
##  [35] "EAST SUSSEX"
##  [36] "ESSEX"
##  [37] "FLINTSHIRE"
##  [38] "GLOUCESTERSHIRE"
##  [39] "GREATER LONDON"
##  [40] "GREATER MANCHESTER"
##  [41] "GWYNEDD"
##  [42] "HALTON"
##  [43] "HAMPSHIRE"
##  [44] "HARTLEPOOL"
##  [45] "HEREFORDSHIRE"
##  [46] "HERTFORDSHIRE"
##  [47] "ISLE OF ANGLESEY"
##  [48] "ISLE OF WIGHT"
##  [49] "ISLES OF SCILLY"
##  [50] "KENT"
##  [51] "LANCASHIRE"
##  [52] "LEICESTER"
##  [53] "LEICESTERSHIRE"
##  [54] "LINCOLNSHIRE"
##  [55] "LUTON"
##  [56] "MEDWAY"
##  [57] "MERSEYSIDE"
##  [58] "MERTHYR TYDFIL"
##  [59] "MIDDLESBROUGH"
##  [60] "MILTON KEYNES"
##  [61] "MONMOUTHSHIRE"
##  [62] "NEATH PORT TALBOT"
##  [63] "NEWPORT"
##  [64] "NORFOLK"
##  [65] "NORTH EAST LINCOLNSHIRE"
##  [66] "NORTH LINCOLNSHIRE"
##  [67] "NORTH SOMERSET"
##  [68] "NORTH YORKSHIRE"
##  [69] "NORTHAMPTONSHIRE"
##  [70] "NORTHUMBERLAND"
##  [71] "NOTTINGHAMSHIRE"
##  [72] "OXFORDSHIRE"
##  [73] "PEMBROKESHIRE"
##  [74] "POOLE"
##  [75] "PORTSMOUTH"
##  [76] "POWYS"
##  [77] "READING"
##  [78] "REDCAR AND CLEVELAND"
##  [79] "RHONDDA CYNON TAFF"
##  [80] "RUTLAND"
##  [81] "SHROPSHIRE"
```

```
##  [82] "SLOUGH"
##  [83] "SOMERSET"
##  [84] "SOUTH GLOUCESTERSHIRE"
##  [85] "SOUTH YORKSHIRE"
##  [86] "SOUTHAMPTON"
##  [87] "SOUTHEND-ON-SEA"
##  [88] "STAFFORDSHIRE"
##  [89] "STOCKTON-ON-TEES"
##  [90] "STOKE-ON-TRENT"
##  [91] "SUFFOLK"
##  [92] "SURREY"
##  [93] "SWANSEA"
##  [94] "SWINDON"
##  [95] "THE VALE OF GLAMORGAN"
##  [96] "THURROCK"
##  [97] "TORBAY"
##  [98] "TORFAEN"
##  [99] "TYNE AND WEAR"
## [100] "WARRINGTON"
## [101] "WARWICKSHIRE"
## [102] "WEST BERKSHIRE"
## [103] "WEST MIDLANDS"
## [104] "WEST SUSSEX"
## [105] "WEST YORKSHIRE"
## [106] "WILTSHIRE"
## [107] "WINDSOR AND MAIDENHEAD"
## [108] "WOKINGHAM"
## [109] "WORCESTERSHIRE"
## [110] "WREKIN"
## [111] "WREXHAM"
## [112] "YORK"

m$distinct("propType")

## [1] "D" "F" "O" "S" "T"
```

- The mongolite package also offers one extremely powerful functionality that is missing in both rmongodb and RMongo. Strictly speaking, it can perform MapReduce operations just like the MapReduce jobs we introduced you to earlier when discussing Big Data analytics with Hadoop and R in Chapter 4, Hadoop and MapReduce Framework for R. However, the tricky thing about it is that the mapreduce() method, which is responsible for MapReduce implementation through the mongolite package, requires the mapper and reduce functions to be written in JavaScript – a skill that may sometimes be beyond the comfort zone for a large number of data scientists. The following is a very simple example of a MapReduce job using the mongolite package that calculates frequencies for two crossed factors of county and property type (propType):

```
houses.xtab <- m$mapreduce(
map = "function(){emit({county:this.county, propType:this.propType}, 1)}",
```

```
reduce = "function(id, counts){return Array.sum(counts)}"
)
houses.xtab
```

```
##                       _id.county _id.propType value
## 1                     DERBYSHIRE            O   225
## 2             GREATER MANCHESTER            S 15245
## 3                      WILTSHIRE            O   165
## 4                    BOURNEMOUTH            S   457
## 5                  TYNE AND WEAR            O   338
## 6                ISLES OF SCILLY            T     7
## 7                     PORTSMOUTH            S   391
## 8                GLOUCESTERSHIRE            T  3689
## 9                     HARTLEPOOL            S   450
## 10                     LEICESTER            O   144
## 11                     WILTSHIRE            D  3408
## 12               WORCESTERSHIRE             T  2645
## 13                       NORFOLK            D  7118
## 14                         DEVON            D  5881
## 15                     THURROCK             S   883
## 16               MERTHYR TYDFIL             T   455
## 17                         YORK             T  1192
## 18               WEST BERKSHIRE             O    62
## 19                  OXFORDSHIRE             O   225
## 20               NORTHUMBERLAND             O    96
## 21                    DARLINGTON            T   630
## 22      NORTH EAST LINCOLNSHIRE             S   858
## 23              CITY OF PLYMOUTH            F   997
## 24                 MILTON KEYNES            T  1712
## 25               WEST BERKSHIRE             S   841
## 26                       SWINDON            F   831
## 27                MIDDLESBROUGH             D   545
## 28                         ESSEX            T  8384
## 29                       BEDFORD            T  1107
## 30                   CEREDIGION             D   478
## 31                CITY OF DERBY             O    82
## 32                   CAERPHILLY             F    55
## 33                   PORTSMOUTH             O    62
## 34                  WARWICKSHIRE            S  3347
## 35  BATH AND NORTH EAST SOMERSET            F   896
## 36      EAST RIDING OF YORKSHIRE            F   418
## 37                 MILTON KEYNES            D  1652
## 38      EAST RIDING OF YORKSHIRE            O    88
## 39                  OXFORDSHIRE             T  3465
## 40               LEICESTERSHIRE             S  4329
## 41                       CUMBRIA            T  3495
## 42              NOTTINGHAMSHIRE             F   685
## 43               NORTH SOMERSET             O    55
## 44                     WILTSHIRE            T  2789
## 45          REDCAR AND CLEVELAND            O    41
```

```
## 46              BRACKNELL FOREST           D   655
## 47            RHONDDA CYNON TAFF           S  1017
## 48                         ESSEX           S  8389
## 49                    SHROPSHIRE           T  1229
## 50             BRIGHTON AND HOVE           D   549
## 51                       SWINDON           D   985
## 52               CARMARTHENSHIRE           S   862
## 53                    EAST SUSSEX           F  3432
## 54                       SOMERSET           F  1246
## 55                      HAMPSHIRE           O   450
## 56              CITY OF PLYMOUTH           T  2086
## 57           CITY OF PETERBOROUGH           T  1201
## 58                  BLAENAU GWENT           O    22
## 59                         SURREY           O   401
## 60                   LINCOLNSHIRE           O   243
## 61                        SWANSEA           S  1120
## 62                       BRIDGEND           D   765
## 63                  LEICESTERSHIRE           O   212
## 64              CITY OF NOTTINGHAM           F   743
## 65                   HERTFORDSHIRE           O   396
## 66                         RUTLAND           F    44
## 67                CITY OF BRISTOL           F  3461
## 68                           LUTON           D   388
## 69                  NORTH SOMERSET           F  1114
## 70                   BLAENAU GWENT           T   581
## 71                  MERTHYR TYDFIL           D   162
## 72                   CITY OF DERBY           T  1190
## 73                      WARRINGTON           O    68
## 74                        BRIDGEND           S   918
## 75                         TORFAEN           T   509
## 76                          TORBAY           F   890
## 77                BRACKNELL FOREST           F   620
## 78                       DERBYSHIRE           F   692
## 79                            CONWY           D   781
## 80                  CAMBRIDGESHIRE           T  3413
## 81                 SOUTH YORKSHIRE           F  1865
## 82                         BRIDGEND           T   805
## 83           CITY OF PETERBOROUGH           D  1030
## 84                           POOLE           S   518
## 85              NEATH PORT TALBOT           D   584
## 86                         RUTLAND           O    12
## 87                CITY OF PLYMOUTH           O    65
## 88     CHESHIRE WEST AND CHESTER           S  2131
## 89          CENTRAL BEDFORDSHIRE           O    65
## 90                     LANCASHIRE           F  1687
## 91     CHESHIRE WEST AND CHESTER           T  1740
## 92                  MONMOUTHSHIRE           T   341
## 93                GREATER LONDON           O  3234
## 94                   EAST SUSSEX           O   194
## 95                        SUFFOLK           D  4961
```

```
## 96                         POWYS            S   417
## 97                 PEMBROKESHIRE            O    36
## 98                LEICESTERSHIRE            T  2968
## 99                     HAMPSHIRE            D  8427
## 100                MONMOUTHSHIRE            D   759
## 101   CITY OF KINGSTON UPON HULL            F   239
## 102                NORTHUMBERLAND            T  1715
## 103                      CORNWALL            T  3580
## 104                ISLE OF WIGHT            D  1079
## 105                         ESSEX            F  6693
## 106                        SURREY            S  5544
## 107                       CARDIFF            O    98
## 108                        MEDWAY            O    84
## 109                  BOURNEMOUTH            O    58
## 110                  SOUTHAMPTON            O    95
## 111                    WOKINGHAM            D  1415
## 112                         POWYS            D   905
## 113                          YORK            S  1088
## 114                TYNE AND WEAR            T  5713
## 115                       GWYNEDD            T   728
## 116                        HALTON            F   113
## 117           GREATER MANCHESTER            F  8064
## 118              SOUTH YORKSHIRE            T  6447
## 119       WINDSOR AND MAIDENHEAD            O    56
## 120                   HARTLEPOOL            F    79
## 121                      READING            F  1531
## 122                         POWYS            O    53
## 123                  LINCOLNSHIRE            F   807
## 124                HERTFORDSHIRE            F  6515
## 125                   WARRINGTON            D  1008
## 126         CITY OF PETERBOROUGH            O    75
## 127                MONMOUTHSHIRE            S   421
## 128              ISLE OF ANGLESEY            D   534
## 129                   SHROPSHIRE            O   131
## 130                   LANCASHIRE            O   416
## 131                         YORK            O    56
## 132                WORCESTERSHIRE            S  3180
## 133                         MEDWAY            S  1181
## 134        SOUTH GLOUCESTERSHIRE            O    75
## 135                  BOURNEMOUTH            T   298
## 136                MERTHYR TYDFIL            S   194
## 137                    FLINTSHIRE            S  1019
## 138         CENTRAL BEDFORDSHIRE            T  2339
## 139        SOUTH GLOUCESTERSHIRE            S  1561
## 140           CITY OF NOTTINGHAM            S  1218
## 141               STOKE-ON-TRENT            D   644
## 142                        POOLE            O    57
## 143           NORTH LINCOLNSHIRE            S  1046
## 144                PEMBROKESHIRE            T   520
## 145                        DEVON            T  5169
```

```
## 146                WEST MIDLANDS           O   823
## 147                STAFFORDSHIRE           T  3291
## 148                STAFFORDSHIRE           S  4760
## 149        BLACKBURN WITH DARWEN           O    49
## 150                   EAST SUSSEX          D  3678
## 151                       NORFOLK          O   268
## 152                    WOKINGHAM           F   556
## 153                      CARDIFF           S  1383
## 154                      GWYNEDD           S   360
## 155             NORTHAMPTONSHIRE           D  5023
## 156            BRIGHTON AND HOVE           F  3161
## 157                  BOURNEMOUTH           F  2326
## 158              SOUTHEND-ON-SEA           F  1636
## 159                   MERSEYSIDE           F  3271
## 160              CITY OF BRISTOL           D   418
## 161                   FLINTSHIRE           O    55
## 162        SOUTH GLOUCESTERSHIRE           T  1890
## 163                       DORSET           S  1513
## 164                      NEWPORT           S   785
## 165 BATH AND NORTH EAST SOMERSET           D   824
## 166                LEICESTERSHIRE          F   739
## 167                 WARWICKSHIRE           T  3322
## 168                TYNE AND WEAR           S  5578
## 169   CITY OF KINGSTON UPON HULL           T  2338
## 170              CITY OF BRISTOL           S  1737
## 171             ISLE OF ANGLESEY           T   248
## 172                    BLACKPOOL           F   189
## 173                HEREFORDSHIRE           T   757
## 174                        LUTON           O    73
## 175                      SUFFOLK           F  1455
## 176    CHESHIRE WEST AND CHESTER           F   825
## 177               NORTH SOMERSET           T  1086
## 178                  WEST SUSSEX           S  4204
## 179              CARMARTHENSHIRE           D  1203
## 180                      NEWPORT           T   954
## 181                 OXFORDSHIRE           D  3282
## 182         REDCAR AND CLEVELAND           S   848
## 183            NEATH PORT TALBOT           O    42
## 184                CHESHIRE EAST           T  2268
## 185     EAST RIDING OF YORKSHIRE           S  2214
## 186                       WREKIN           D  1040
## 187                HEREFORDSHIRE           D  1318
## 188                TYNE AND WEAR           D  2484
## 189           NORTH LINCOLNSHIRE           F    65
## 190                CITY OF DERBY           D  1048
## 191              NORTHUMBERLAND           F   483
## 192                WEST MIDLANDS           D  6521
## 193              CAMBRIDGESHIRE           O   220
## 194         CENTRAL BEDFORDSHIRE           F  1069
## 195             CITY OF PLYMOUTH           S  1165
```

```
## 196   CITY OF KINGSTON UPON HULL           O    95
## 197                  FLINTSHIRE           T   423
## 198              NOTTINGHAMSHIRE           D  5475
## 199                      DORSET           F  1590
## 200                     TORFAEN           F    60
## 201               WORCESTERSHIRE           O   229
## 202                       WREKIN           T   902
## 203                 DENBIGHSHIRE           D   727
## 204                       LUTON           T  1160
## 205                CHESHIRE EAST           S  2360
## 206            RHONDDA CYNON TAFF           D   756
## 207   CITY OF KINGSTON UPON HULL           D   433
## 208              BRACKNELL FOREST           S   455
## 209                WEST YORKSHIRE           F  4174
## 210                 WARWICKSHIRE           D  3551
## 211              CITY OF PLYMOUTH           D   552
## 212                       WREXHAM           T   434
## 213                       SWINDON           S  1058
## 214                MIDDLESBROUGH           F   106
## 215                       WREKIN           S   976
## 216                      SOMERSET           O   210
## 217               ISLES OF SCILLY           D     5
## 218             NEATH PORT TALBOT           S   814
## 219                       HALTON           T   627
## 220                GREATER LONDON           S 16373
## 221               NORTH YORKSHIRE           T  3341
## 222                      SOMERSET           D  3737
## 223                COUNTY DURHAM           D  1917
## 224               STOKE-ON-TRENT           S  1480
## 225               CAMBRIDGESHIRE           D  4380
## 226                      CARDIFF           F  1599
## 227                   MERSEYSIDE           O   531
## 228                         KENT           F  6068
## 229                ISLE OF WIGHT           T   704
## 230               CARMARTHENSHIRE           T   663
## 231                   FLINTSHIRE           D  1087
## 232                WEST MIDLANDS           S 13443
## 233                  SOUTHAMPTON           D   608
## 234                   MERSEYSIDE           S  7380
## 235         BLACKBURN WITH DARWEN           F    72
## 236                GREATER LONDON           T 33571
## 237              STOCKTON-ON-TEES           T   802
## 238                 CITY OF DERBY           S  1493
## 239                         DEVON           O   280
## 240        WINDSOR AND MAIDENHEAD           D   783
## 241                       SWANSEA           T  1200
## 242                  WEST SUSSEX           O   244
## 243             RHONDDA CYNON TAFF           O    55
## 244                ISLES OF SCILLY           O     2
## 245                       DORSET           T  2107
```

```
## 246                PEMBROKESHIRE           S   504
## 247                COUNTY DURHAM           O   180
## 248            NEATH PORT TALBOT           F    57
## 249              GLOUCESTERSHIRE           O   278
## 250                     HAMPSHIRE           T  7120
## 251                MONMOUTHSHIRE           F    99
## 252                   SHROPSHIRE           F   556
## 253           NORTH LINCOLNSHIRE           D  1139
## 254                       MEDWAY           D   662
## 255      NORTH EAST LINCOLNSHIRE           D   620
## 256                       TORBAY           D   775
## 257       WINDSOR AND MAIDENHEAD           F   949
## 258                     BEDFORD           O    48
## 259                    WOKINGHAM           O    31
## 260        SOUTH GLOUCESTERSHIRE           F   880
## 261                    WOKINGHAM           T   755
## 262                         KENT           T  9497
## 263         CITY OF PETERBOROUGH           F   442
## 264                WEST MIDLANDS           T 13717
## 265              BUCKINGHAMSHIRE           T  2708
## 266                       MEDWAY           T  2431
## 267                 TYNE AND WEAR           F  3354
## 268                WEST BERKSHIRE           T   803
## 269                NORTHUMBERLAND           D  1638
## 270                    MERSEYSIDE           D  3109
## 271                   SHROPSHIRE           D  2126
## 272           NORTH LINCOLNSHIRE           O    40
## 273                MILTON KEYNES           F  1003
## 274                       TORFAEN           S   314
## 275                     CORNWALL           D  4268
## 276           GREATER MANCHESTER           D  6927
## 277                       SWANSEA           F   395
## 278                       NEWPORT           O    36
## 279        THE VALE OF GLAMORGAN           D   683
## 280                     BLACKPOOL           O    75
## 281                         KENT           O   510
## 282                    FLINTSHIRE           F    67
## 283                  PEMBROKESHIRE           D   903
## 284        BLACKBURN WITH DARWEN           D   294
## 285                      GWYNEDD           F   113
## 286                      SUFFOLK           O   254
## 287                MERTHYR TYDFIL           O    14
## 288                 HERTFORDSHIRE           T  6602
## 289                       SLOUGH           S   399
## 290        BLACKBURN WITH DARWEN           T   965
## 291    CHESHIRE WEST AND CHESTER           D  2236
## 292                   CAERPHILLY           T  1023
## 293                   PORTSMOUTH           D   167
## 294             BRIGHTON AND HOVE           S   795
## 295                       CONWY           O    51
```

```
## 296                  DERBYSHIRE            D  4914
## 297               LEICESTERSHIRE           D  5240
## 298              GLOUCESTERSHIRE           D  3901
## 299                  EAST SUSSEX           S  2479
## 300                    BRIDGEND            F   139
## 301         REDCAR AND CLEVELAND           D   506
## 302              BUCKINGHAMSHIRE           O   187
## 303               STOKE-ON-TRENT           O    87
## 304                  WEST SUSSEX           T  4740
## 305      NORTH EAST LINCOLNSHIRE           O    49
## 306              NOTTINGHAMSHIRE           S  4829
## 307                     READING            O    53
## 308                 DENBIGHSHIRE           T   242
## 309                  OXFORDSHIRE           F  2098
## 310                    BLACKPOOL           D   196
## 311      NORTH EAST LINCOLNSHIRE           F   100
## 312        SOUTH GLOUCESTERSHIRE           D  1310
## 313           CITY OF NOTTINGHAM           D   643
## 314       WINDSOR AND MAIDENHEAD           T   754
## 315                  LANCASHIRE            S  6312
## 316                  CEREDIGION            O    28
## 317                      TORBAY            S   627
## 318           CITY OF NOTTINGHAM           O   116
## 319                     CUMBRIA            S  2662
## 320                STAFFORDSHIRE           D  4923
## 321                     CUMBRIA            F   858
## 322                       LUTON            S  1188
## 323               WEST YORKSHIRE           T 13243
## 324                COUNTY DURHAM           F   281
## 325                     GWYNEDD            D   626
## 326         CENTRAL BEDFORDSHIRE           S  1792
## 327                   CAERPHILLY           D   512
## 328              NORTH YORKSHIRE           S  3129
## 329              SOUTH YORKSHIRE           D  4538
## 330                     GWYNEDD            O    51
## 331                       POOLE            F  1028
## 332                HERTFORDSHIRE           S  4517
## 333                     READING            T  1361
##  [ reached getOption("max.print") -- omitted 167 rows ]
##  [ reached 'max' / getOption("max.print") -- omitted 60 rows ]
```