Big Data

Aggregations with MongoDB

von
Benjamin Ellmer (S2210455012)

Mobile Computing Master
FH Hagenberg

April 28, 2023

Step 1

Install and start MongoDB using docker compose:

```
git clone https://github.com/Digital-Media/fhooe-mongo-dock
   .git
docker compose -f fhooe-mongo-dock/docker-compose.yml up -d
```

Connect to the MongoDB shell in the container:

```
docker exec -it mongodb /bin/bash -c "mongosh"
```

Show and create database:

```
show databases
use onlineshop
```

Insert Data from exercise description:

```
db.orders.insertMany([
    ...
]);
```

Find all rows of collection orders:

```
db.orders.find({})
```

Step 2

My Summary

Map-reduce is a method of processing large volumes of data into useful aggregated results. MongoDB has a mapReduce command that performs map-reduce operations. In this process, MongoDB applies the map phase to each input document and emits key-value pairs, which are then condensed and aggregated using the reduce phase. The results are stored in a collection, and may also pass through a finalize function for further processing. Map-reduce functions in MongoDB are JavaScript-based and can perform arbitrary sorting and limiting before the map stage. They can also use custom JavaScript functions to provide flexibility to the operation, and can write results to a collection or return them inline. When writing to a collection, subsequent map-reduce operations can merge, replace or reduce new results with previous ones.

Step 3

Get sum of total sums:

```
onlineshop> db.orders.aggregate([
    {
        $group: {
            _id: null,
            totalSum: { $sum: "$total_sum" }
        }
    }
])
```

Output:

```
[ { _id: null, totalSum: 435 } ]
```

Get sum of qty * price per user:

```
db.orders.aggregate([
    { $unwind: "$items" },
    { $group: {
        _id: "$user_id",
        sum:{ $sum: { $multiply:
            [ "$items.price", "$items.qty"]
        }}
    }}
])
```

Output:

```
[
  { _id: 2, sum: 125 },
  { _id: 4, sum: 155 },
  { _id: 1, sum: 95 },
  { _id: 3, sum: 60 }
]
```

Get sum of qty * price per user using mapReduce:

```
db.orders.mapReduce(
    function() { this.items.forEach(
        item => emit(this.user_id, item.qty * item.price)
    );},
    function(key, values) { return Array.sum(values); },
    { out: { inline: 1 },
      finalize: function(key, reducedValue) {
        return { _id: key, sum: reducedValue };
      }
    }
)
```

Output:

```
{
  results: [
    { _id: 1, value: { _id: 1, sum: 95 } },
    { _id: 4, value: { _id: 4, sum: 155 } },
    { _id: 2, value: { _id: 2, sum: 125 } },
    { _id: 3, value: { _id: 3, sum: 60 } }
  ],
  ok: 1
}
```

Step 4

Delete all data in the orders collection:

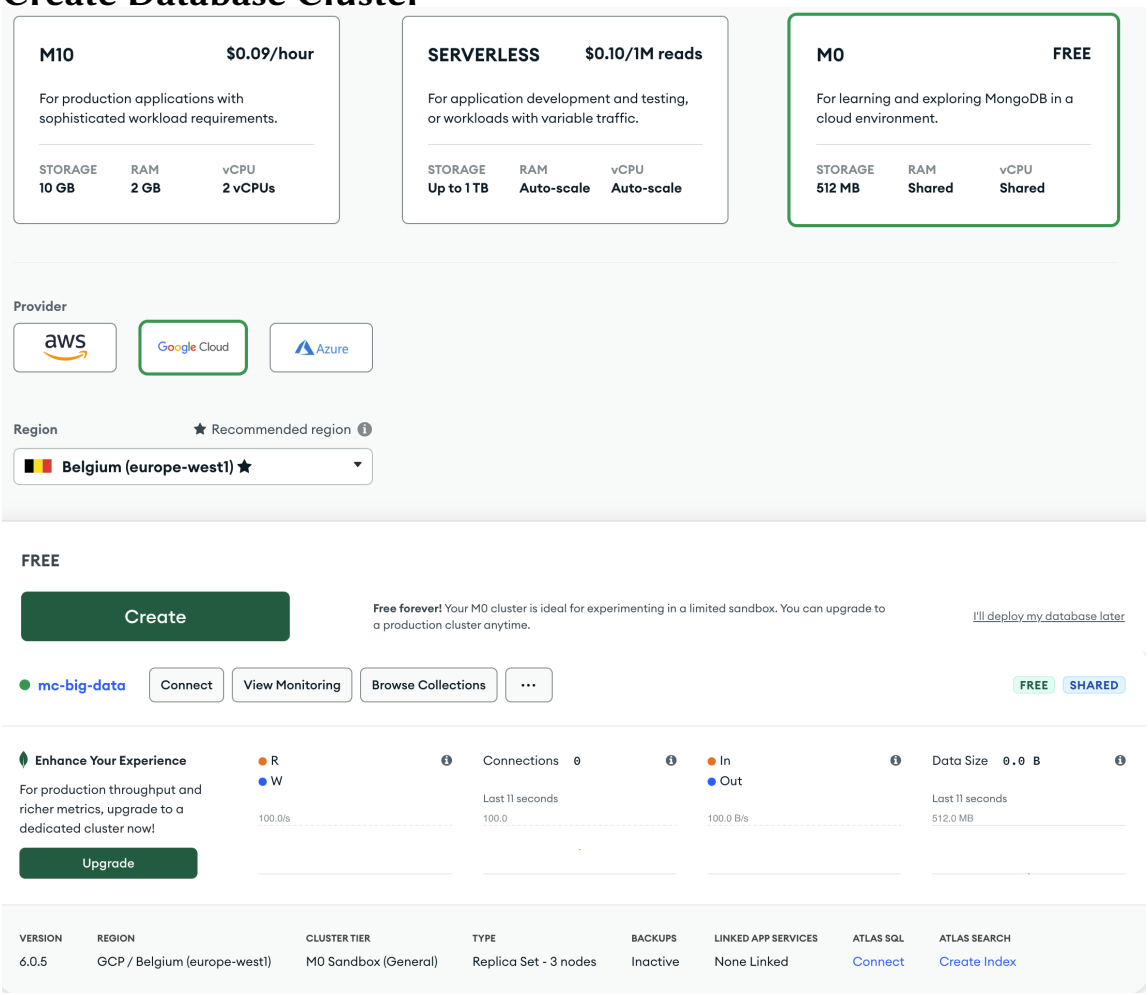
```
db.orders.deleteMany({})
```

Drop the onlineshop database

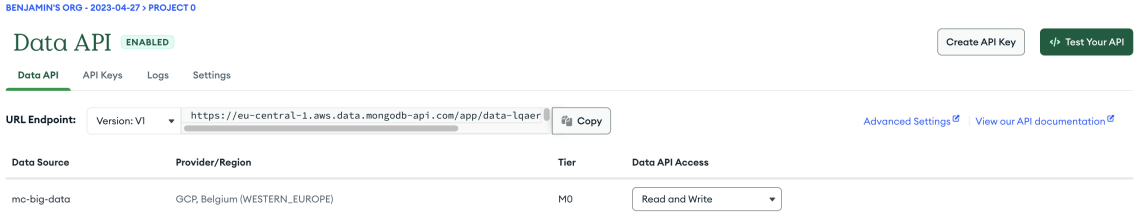
```
db.dropDatabase()
```

Step 6 - Work with MongoDB Altas Data API

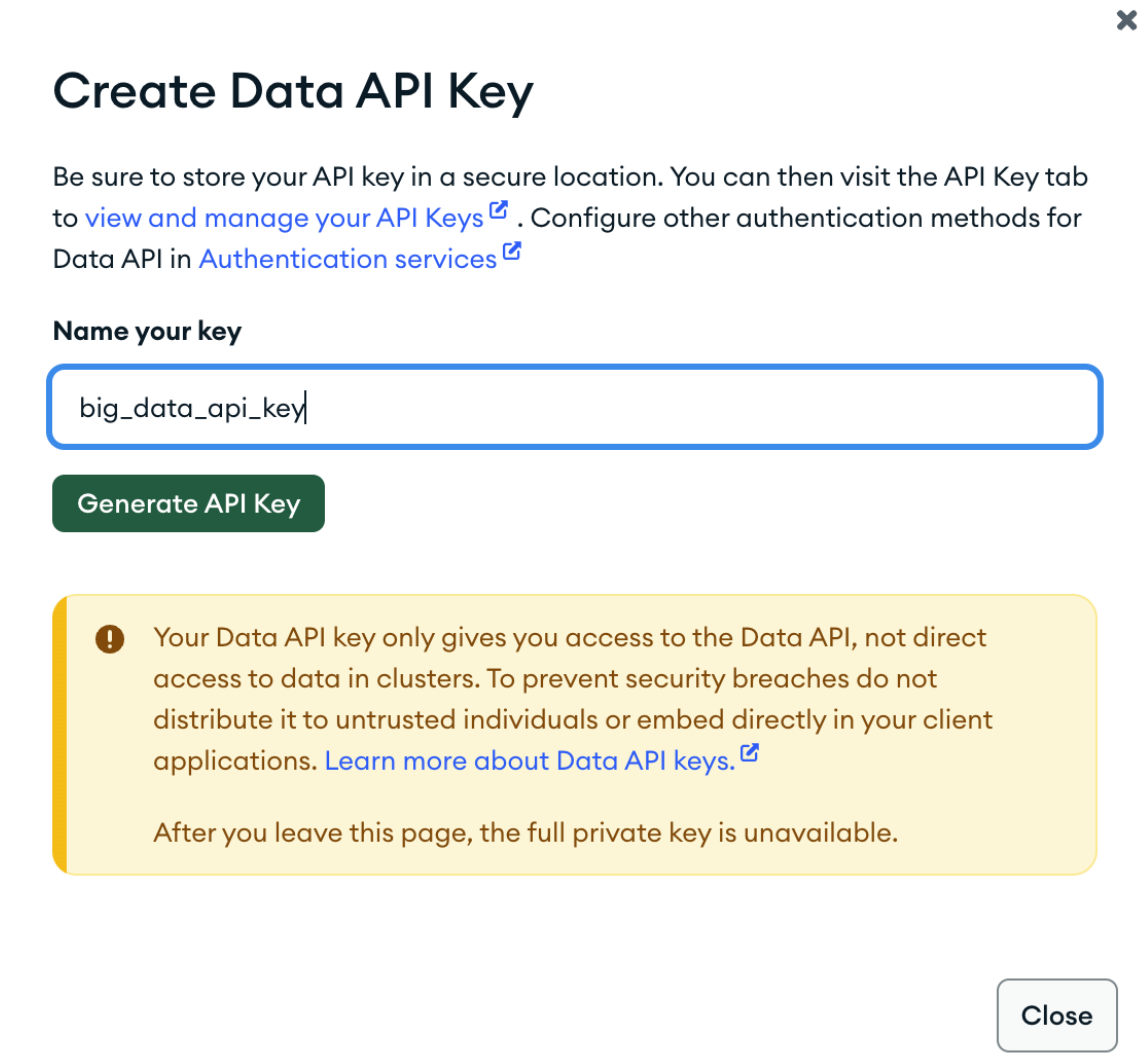
Create Database Cluster



Enable and configure Data API



Create API Key



Create Environment variables in .zshrc

```
export MONGO_ATLAS_PW="***"
export MONGO_ATLAS_API_KEY="***"
```

Connect using mongo shell

```
benjaminellmer ~/repos/mc-big-data-exercises/ex02 main ✗
⚡ mongosh "mongodb+srv://mc-big-data.5nebun9.mongodb.net/myFirstDatabase" --apiVersion 1 --username s2210455012 --password $MONGO_ATLAS_PW
Current Mongosh Log ID: 644c03c0fa9f256f25d68486
Connecting to:          mongodb+srv://<credentials>@mc-big-data.5nebun9.mongodb.net/myFirstDatabase?appName=mongosh+1.7.1
Using MongoDB:          6.0.5 (API Version 1)
Using Mongosh:          1.7.1

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

Atlas atlas-yqkfod-shard-0 [primary] myFirstDatabase>
```

Create Database and Sample Data

```
Atlas atlas-yqkfod-shard-0 [primary] myFirstDatabase> use onlineshop
switched to db onlineshop
Atlas atlas-yqkfod-shard-0 [primary] onlineshop> db.product.insertMany([
...   { pname: "Product A", price: 10.99, status: "published" },
...   { pname: "Product B", price: 29.99, status: "revision" },
...   { pname: "Product C", price: 15.49, status: "published" },
...   { pname: "Product D", price: 5.99, status: "revision" },
...   { pname: "Product E", price: 49.99, status: "published" },
...   { pname: "Product F", price: 9.99, status: "revision" },
...   { pname: "Product G", price: 19.99, status: "published" },
...   { pname: "Product H", price: 25.99, status: "revision" },
...   { pname: "Product I", price: 39.99, status: "published" },
...   { pname: "Product J", price: 8.49, status: "revision" }
... ]);
```

```
Atlas atlas-yqkfod-shard-0 [primary] onlineshop> db.user.insertMany([
...   {
...     first_name: "John",
...     last_name: "Doe",
...     email: "johndoe@example.com",
...     password: "sp7FYDRyD7fejM3OvORhnyePcvg1",
...     date_registered: new Date("2022-01-15"),
...     phone_numbers: {
...       mobile: "123-456-7890",
...       private: "111-222-3333",
...       fax: "444-555-6666"
...     }
...   },
...   {
...     first_name: "Jane",
...     last_name: "Doe",
...     email: "janedoe@example.com",
...     password: "EDjAO66mKgVA0sKFg8NiDn9G75b2",
...     date_registered: new Date("2023-04-28"),
...     phone_numbers: {
...       mobile: "987-654-3210"
...     }
...   }
... ]);
```