Домашняя работа №1. MongoDB

В качестве тестового датасета взят ***Mall_Customer***, с 5-ю колонками.

1. Установка ***mongodb*** и ***pymongo*** через homebrew:

```
Collecting pymongo
  Downloading pymongo-4.0.2-cp38-cp38-macosx_10_9_x86_64.whl (350 kB)
     |                                | 350 kB 1.4 MB/s
Installing collected packages: pymongo
Successfully installed pymongo-4.0.2
WARNING: You are using pip version 21.1.1; however, version 22.0.4 is available.
You should consider upgrading via the '/Library/Frameworks/Python.framework/Versions/3.8/bin/python3.8 -m pip install --upgrade pip' command.
```

2. Запуск базы командой:

```
mongod --config /usr/local/etc/mongod.conf
```

3. Проверка, что данные подгрузились и несколько запросов:

```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> show dbs
admin     0.000GB
config    0.000GB
database  0.000GB
local     0.000GB
> db.collection.find()
> use database
switched to db database
> db.collection.find()
{ "_id" : ObjectId("6226062da8ccade3e1d33f73"), "CustomerID" : 1, "Genre" : "Male", "Age" : 19, "Annual Income (k$)" : 15, "Spending Score (1-100)" : 39 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f74"), "CustomerID" : 8, "Genre" : "Female", "Age" : 23, "Annual Income (k$)" : 18, "Spending Score (1-100)" : 94 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f75"), "CustomerID" : 9, "Genre" : "Male", "Age" : 64, "Annual Income (k$)" : 19, "Spending Score (1-100)" : 3 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f76"), "CustomerID" : 10, "Genre" : "Female", "Age" : 30, "Annual Income (k$)" : 19, "Spending Score (1-100)" : 72 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f77"), "CustomerID" : 11, "Genre" : "Male", "Age" : 67, "Annual Income (k$)" : 19, "Spending Score (1-100)" : 14 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f78"), "CustomerID" : 12, "Genre" : "Female", "Age" : 35, "Annual Income (k$)" : 19, "Spending Score (1-100)" : 99 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f79"), "CustomerID" : 13, "Genre" : "Female", "Age" : 58, "Annual Income (k$)" : 20, "Spending Score (1-100)" : 15 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f7a"), "CustomerID" : 14, "Genre" : "Female", "Age" : 24, "Annual Income (k$)" : 20, "Spending Score (1-100)" : 77 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f7b"), "CustomerID" : 15, "Genre" : "Male", "Age" : 37, "Annual Income (k$)" : 20, "Spending Score (1-100)" : 13 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f7c"), "CustomerID" : 16, "Genre" : "Male", "Age" : 22, "Annual Income (k$)" : 20, "Spending Score (1-100)" : 79 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f7d"), "CustomerID" : 17, "Genre" : "Female", "Age" : 35, "Annual Income (k$)" : 21, "Spending Score (1-100)" : 35 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f7e"), "CustomerID" : 18, "Genre" : "Male", "Age" : 20, "Annual Income (k$)" : 21, "Spending Score (1-100)" : 66 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f7f"), "CustomerID" : 19, "Genre" : "Male", "Age" : 52, "Annual Income (k$)" : 23, "Spending Score (1-100)" : 29 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f80"), "CustomerID" : 3, "Genre" : "Female", "Age" : 20, "Annual Income (k$)" : 16, "Spending Score (1-100)" : 6 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f81"), "CustomerID" : 7, "Genre" : "Female", "Age" : 35, "Annual Income (k$)" : 18, "Spending Score (1-100)" : 6 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f82"), "CustomerID" : 6, "Genre" : "Female", "Age" : 22, "Annual Income (k$)" : 17, "Spending Score (1-100)" : 76 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f83"), "CustomerID" : 4, "Genre" : "Female", "Age" : 23, "Annual Income (k$)" : 16, "Spending Score (1-100)" : 77 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f84"), "CustomerID" : 5, "Genre" : "Female", "Age" : 31, "Annual Income (k$)" : 17, "Spending Score (1-100)" : 40 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f85"), "CustomerID" : 21, "Genre" : "Male", "Age" : 35, "Annual Income (k$)" : 24, "Spending Score (1-100)" : 35 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f86"), "CustomerID" : 22, "Genre" : "Male", "Age" : 25, "Annual Income (k$)" : 24, "Spending Score (1-100)" : 73 }
Type "it" for more
>
```

```
> db.collection.find({ "Age": { $gt: 50 }})
{ "_id" : ObjectId("6226062da8ccade3e1d33f75"), "CustomerID" : 9, "Genre" : "Male", "Age" : 64, "Annual Income (k$)" : 19, "Spending Score (1-100)" : 3 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f77"), "CustomerID" : 11, "Genre" : "Male", "Age" : 67, "Annual Income (k$)" : 19, "Spending Score (1-100)" : 14 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f79"), "CustomerID" : 13, "Genre" : "Female", "Age" : 58, "Annual Income (k$)" : 20, "Spending Score (1-100)" : 15 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f7f"), "CustomerID" : 19, "Genre" : "Male", "Age" : 52, "Annual Income (k$)" : 23, "Spending Score (1-100)" : 29 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f88"), "CustomerID" : 25, "Genre" : "Female", "Age" : 54, "Annual Income (k$)" : 28, "Spending Score (1-100)" : 14 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f8f"), "CustomerID" : 31, "Genre" : "Male", "Age" : 60, "Annual Income (k$)" : 30, "Spending Score (1-100)" : 4 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f91"), "CustomerID" : 33, "Genre" : "Male", "Age" : 53, "Annual Income (k$)" : 33, "Spending Score (1-100)" : 4 }
{ "_id" : ObjectId("6226062da8ccade3e1d33f99"), "CustomerID" : 41, "Genre" : "Female", "Age" : 65, "Annual Income (k$)" : 38, "Spending Score (1-100)" : 35 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fa6"), "CustomerID" : 54, "Genre" : "Male", "Age" : 59, "Annual Income (k$)" : 43, "Spending Score (1-100)" : 60 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fa9"), "CustomerID" : 57, "Genre" : "Female", "Age" : 51, "Annual Income (k$)" : 44, "Spending Score (1-100)" : 50 }
{ "_id" : ObjectId("6226062da8ccade3e1d33faa"), "CustomerID" : 58, "Genre" : "Male", "Age" : 69, "Annual Income (k$)" : 44, "Spending Score (1-100)" : 46 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fad"), "CustomerID" : 61, "Genre" : "Male", "Age" : 70, "Annual Income (k$)" : 46, "Spending Score (1-100)" : 56 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fae"), "CustomerID" : 60, "Genre" : "Male", "Age" : 53, "Annual Income (k$)" : 46, "Spending Score (1-100)" : 46 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fb0"), "CustomerID" : 63, "Genre" : "Female", "Age" : 67, "Annual Income (k$)" : 47, "Spending Score (1-100)" : 52 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fb1"), "CustomerID" : 64, "Genre" : "Female", "Age" : 54, "Annual Income (k$)" : 47, "Spending Score (1-100)" : 59 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fb2"), "CustomerID" : 65, "Genre" : "Male", "Age" : 63, "Annual Income (k$)" : 48, "Spending Score (1-100)" : 51 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fb7"), "CustomerID" : 68, "Genre" : "Female", "Age" : 68, "Annual Income (k$)" : 48, "Spending Score (1-100)" : 48 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fba"), "CustomerID" : 71, "Genre" : "Male", "Age" : 70, "Annual Income (k$)" : 49, "Spending Score (1-100)" : 55 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fbb"), "CustomerID" : 75, "Genre" : "Male", "Age" : 59, "Annual Income (k$)" : 54, "Spending Score (1-100)" : 47 }
{ "_id" : ObjectId("6226062da8ccade3e1d33fbc"), "CustomerID" : 73, "Genre" : "Female", "Age" : 60, "Annual Income (k$)" : 50, "Spending Score (1-100)" : 49 }
```

```
> db.collection.find({Age:57}).pretty();
{
        "_id" : ObjectId("6226062da8ccade3e1d33fc2"),
        "CustomerID" : 81,
        "Genre" : "Male",
        "Age" : 57,
        "Annual Income (k$)" : 54,
        "Spending Score (1-100)" : 51
}
{
        "_id" : ObjectId("6226062da8ccade3e1d34000"),
        "CustomerID" : 141,
        "Genre" : "Female",
        "Age" : 57,
        "Annual Income (k$)" : 75,
        "Spending Score (1-100)" : 5
}
> db.collection.update({Age: 57}, {$set : {Age: 85}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.collection.find({Age:85}).pretty();
{
        "_id" : ObjectId("6226062da8ccade3e1d33fc2"),
        "CustomerID" : 81,
        "Genre" : "Male",
        "Age" : 85,
        "Annual Income (k$)" : 54,
        "Spending Score (1-100)" : 51
}
```

4. Т. к. на выбранном мною датасете сложно замерить производительность, ввиду того что там мало записей и все операции выполняются примерно одинаковое кол-во времени, я нашел в интернете другой сет на котором можно замерить статистику и почувствовать разницу в производительности:
   Без индексов:

```
db.products.find({
   stock: {
      $lt: 100
   }
}).explain("executionStats")

{
        "executionSuccess" : true,
        "nReturned" : 3,
        "executionTimeMillis" : 54744,
        "totalKeysExamined" : 0,
        "totalDocsExamined" : 722531,
        "executionStages" : {
        "stage" : "COLLSCAN",
        ……
}
```

С индексами:

```
db.products.find({
   stock: {
      $lt: 100
   }
}).explain("executionStats")

{
        "executionSuccess" : true,
        "nReturned" : 3,
        "executionTimeMillis" : 31,
        "totalKeysExamined" : 3,
        "totalDocsExamined" : 3,
        "executionStages" : {
        "stage" : "FETCH",
        ……..
        "inputStage" : {
                "stage" : "IXSCAN",
                ……..
                "keyPattern" : {
                   "stock" : 1
                },
                "indexName" : "stock_1",
                "isMultiKey" : false,
                ……..
            }
        }
}
```

Видно, что производительность вырастает в разы, также можно построить график и посмотреть на статистику наглядно, вот такие данные я нашел в интернете: