

LAB: MONGODB FOR JAVA DEVELOPERS

Document <https://www.mongodb.com/docs/develop-applications/>

Drivers: <https://www.mongodb.com/docs/drivers/java-drivers/>

Download the sample data-

<https://atlas-education-staging.s3.amazonaws.com/sampleddata.archive.gz>

Import command: mongorestore --gzip --archive=sampleddata.archive.gz

PHẦN 1: MONGODB JAVA DRIVER

Document - <https://www.mongodb.com/docs/drivers/java/sync/current/>

```
<dependencies>
  <dependency>
    <groupId>org.mongodb</groupId>
    <artifactId>mongodb-driver-sync</artifactId>
    <version>4.7.0</version>
  </dependency>
</dependencies>
```

PHẦN 2: MONGODB JAVA REACTIVE STREAMS

Document - <https://www.mongodb.com/docs/drivers/reactive-streams/>

```
<dependencies>
  <dependency>
    <groupId>org.mongodb</groupId>
    <artifactId>mongodb-driver-reactivestreams</artifactId>
    <version>4.6.0</version>
  </dependency>
</dependencies>
```

Bài tập chung cho phần 1 và phần 2: Kết nối tới MongoDB Atlas hoặc kết nối tới một Local MongoDB Server để thực hiện các bài tập sau:

BÀI TẬP 1

Thực thi scripts cho sẵn trên MS SQL Server. [Link](#) để download scripts

Viết chương trình cho phép đổ dữ liệu từ các bảng vào MongoDB với các collections tương ứng trong database BikeStores.

Các Documents trong MongoDB có cấu trúc sau:

Product json data
<pre>{ "_id": 133, "brand_name": "Trek", "category_name": "Mountain Bikes",</pre>

```

"colors": [
    "orange",
    "red"
],
"model_year": 2018,
"product_name": "Trek Procaliber Frameset - 2018",
"price": 1499.99
}

```

Customer json data

```

{
  "_id": "7",
  "first_name": "Latasha",
  "last_name": "Hays",
  "address": {
    "city": "Buffalo",
    "state": "NY",
    "street": "7014 Manor Station Rd. ",
    "zip_code": 14215
  },
  "registration_date": {
    "$date": "2000-10-16T17:00:00Z"
  },
  "email": "latasha.hays@hotmail.com",
  "phones": [
    {
      "type": "Home",
      "number": "(716) 986-3359"
    },
    {
      "type": "personal",
      "number": "(322) 064-3093"
    }
  ]
}

```

Staff json data

```

{
  "_id": 9,
  "first_name": "Layla",
  "last_name": "Terrell",
  "phone": {
    "number": "(972) 530-5556",
    "type": "personal"
  },
  "manager_id": 7,
  "email": "layla.terrell@bikes.shop"
}

```

Order json data

```
{
  "_id": "573a1390f29313caabcd421c",
  "order_date": "2016-07-11",
  "customer ": {
    "customer_id": "TOVE11357",
    "full_name": "Tona Velasquez",
    "email": "tona.velasquez@msn.com",
    "phone_number": "645 946-648"
  },
  "order_status": "Complete",
  "shipped_date": "2016-07-11",
  "staff": {
    "staff_id": 2,
    "full_name": "Mireya Copeland",
    "phone_number": "(831) 555-5555"
  },
  "order_details": [
    {
      "quantity": 2,
      "color": "red",
      "product_id": 3,
      "line_total": 1859.9814,
      "price": 999.99,
      "discount": 0.07
    },
    {
      "quantity": 2,
      "color": "blue",
      "product_id": 9,
      "line_total": 5579.9814,
      "price": 2999.99,
      "discount": 0.07
    }
  ],
  "order_total": 97439.9628,
  "shipping_address": {
    "city": "Whitestone",
    "zip_code": "11357",
    "street": "9166 Trout St. ",
    "state": "NY"
  }
}
```

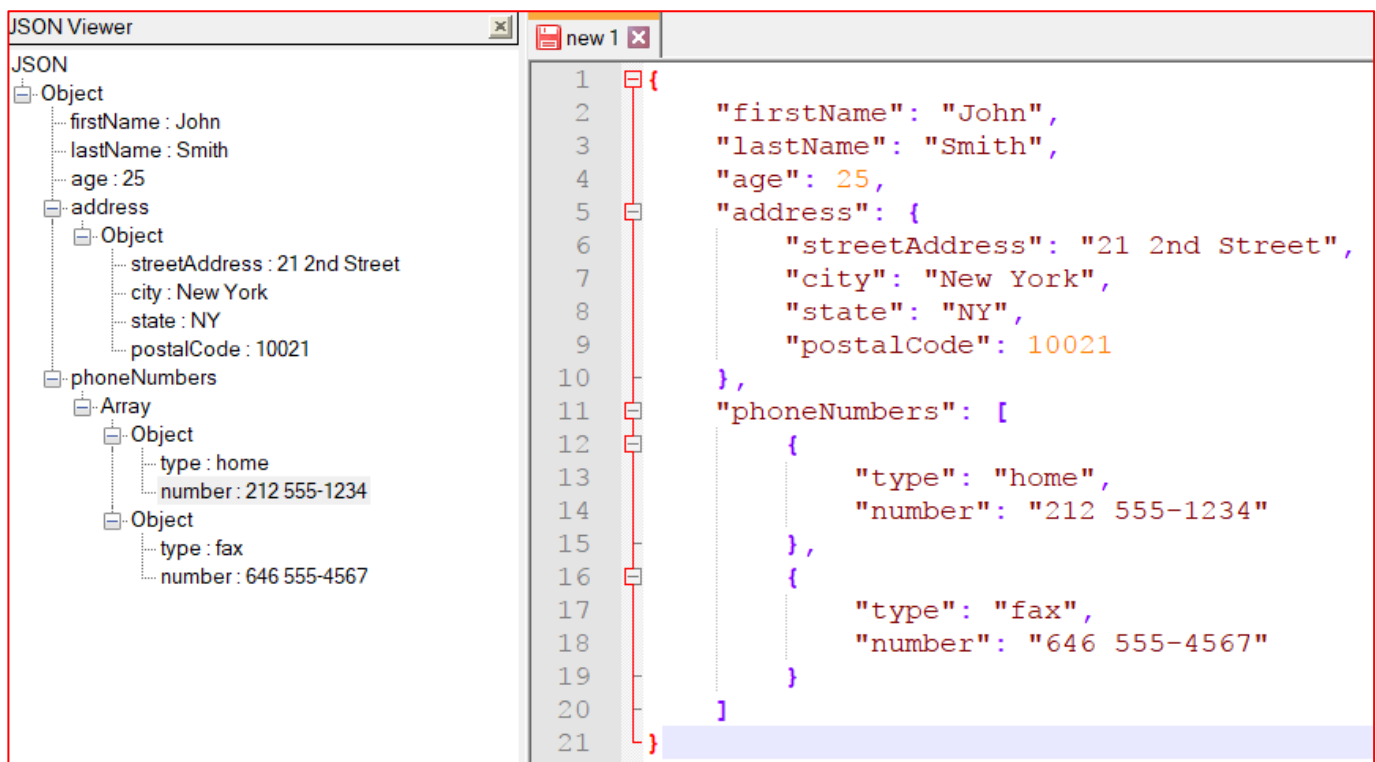
BÀI TẬP 2

Database tên là sample_training, collection có tên zips. Viết các phương thức thực hiện các yêu cầu sau:

1. Hiển thị n documents từ document thứ k.
2. Chèn thêm 1 document mới
3. Cập nhật thông tin của một document khi biết id
4. Tìm các document có city là PALMER
5. Tìm các document có dân số >100000
6. Tìm dân số của thành phố FISHERS ISLAND
7. Tìm các thành phố có dân số từ 10 – 50
8. Tìm tất cả các thành phố của bang MA có dân số trên 500
9. Tìm tất cả các bang (*không trùng*)
10. Tìm tất cả các bang mà có chứa ít nhất 1 thành phố có dân số trên 100000
11. Tính dân số trung bình của mỗi bang
12. Tìm những document của bang 'CT' và thành phố 'WATERBURY'
13. Bang WA có bao nhiêu city (*nếu trùng chỉ tính 1 lần*)
14. Tính số city của mỗi bang (*nếu trùng chỉ tính 1 lần*), kết quả giảm dần theo số city
15. Tìm tất cả các bang có tổng dân số trên 10000000
16. Tìm các document có dân số lớn (*nhỏ*) nhất
17. Tìm bang có tổng dân số lớn nhất
18. Xuất những document có dân số dưới dân số trung bình của mỗi city
19. Dựa vào tập kết quả câu trên, xóa các documents khi biết city

BÀI TẬP 3

Cho tài liệu JSON có cấu trúc như hình sau



The screenshot shows a JSON Viewer application with a tree view on the left and a code editor on the right. The tree view shows a JSON object with the following structure:

- Object
 - firstName: John
 - lastName: Smith
 - age: 25
 - address: Object
 - streetAddress: 21 2nd Street
 - city: New York
 - state: NY
 - postalCode: 10021
 - phoneNumbers: Array
 - Object
 - type: home
 - number: 212 555-1234
 - Object
 - type: fax
 - number: 646 555-4567

The code editor on the right shows the corresponding JSON text:

```

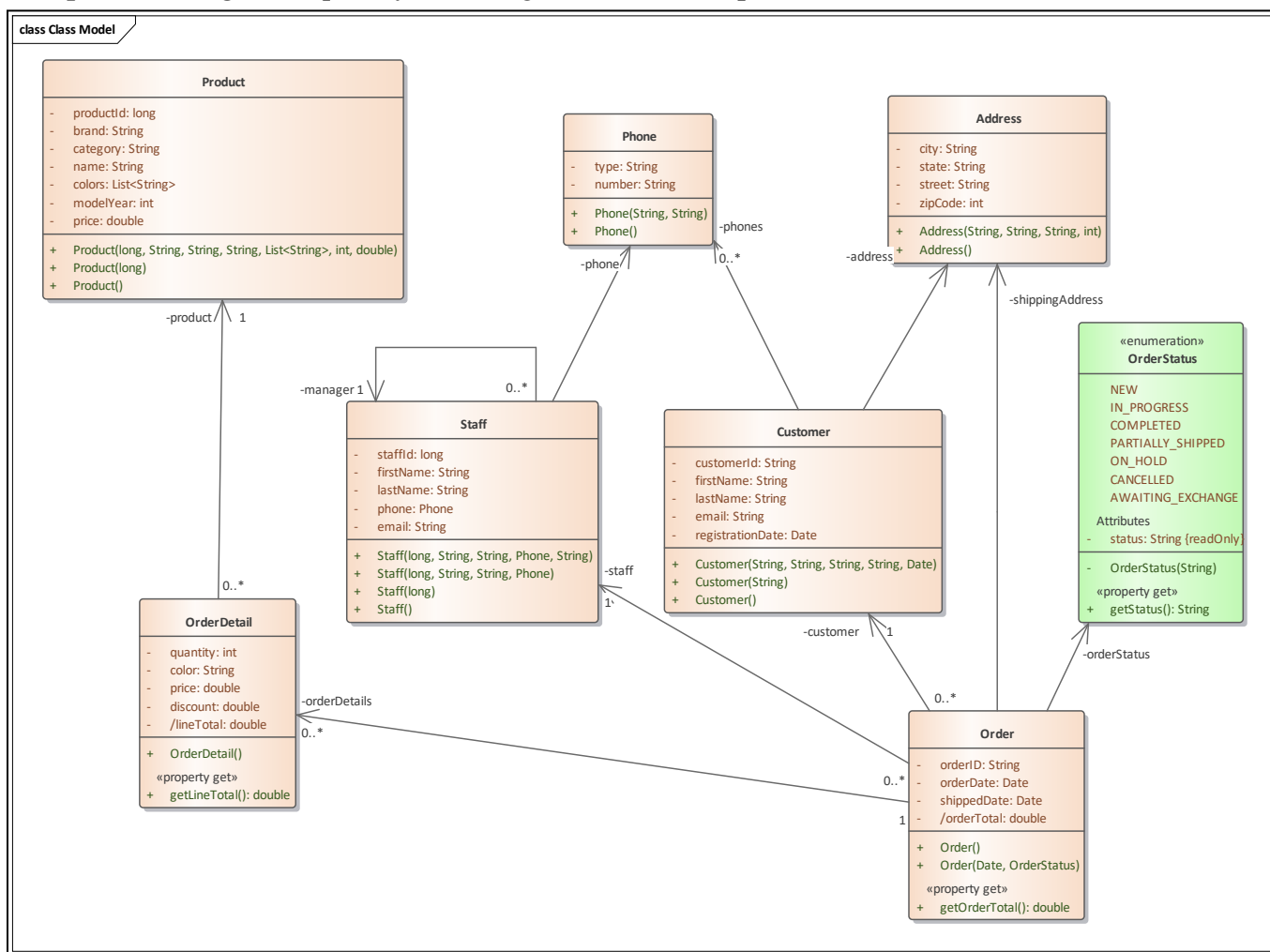
1 {
2   "firstName": "John",
3   "lastName": "Smith",
4   "age": 25,
5   "address": {
6     "streetAddress": "21 2nd Street",
7     "city": "New York",
8     "state": "NY",
9     "postalCode": 10021
10  },
11  "phoneNumbers": [
12    {
13      "type": "home",
14      "number": "212 555-1234"
15    },
16    {
17      "type": "fax",
18      "number": "646 555-4567"
19    }
20  ]
21 }
```

Hãy viết một lớp với các phương thức thực hiện các hành vi Create, Update, Delete và các phương thức Read với các yêu cầu sau:

1. Tìm những person đang sống tại 1 bang cho trước
2. Tìm số điện thoại của 1 person cho trước khi biết firstName và lastName
3. Tìm những person có tuổi trên 50
4. Tính độ tuổi trung bình cho mỗi bang
5. Tính tổng số person cho mỗi bang

BÀI TẬP 4

Một phần chương trình quản lý bán hàng với mô hình lớp như sau:



Các document trong MongoDB có cấu trúc như bài tập 1.

Dữ liệu mẫu: [Link to download data](#)

Restore database cho sẵn vào MongoDB có tên BikeStores với tên các collection tương ứng.

Lập trình viết các phương thức với các yêu cầu sau:

1. CRUD với các phương thức:
 - a. Thêm 1 document, thêm nhiều document: `insertOne`, `insertMany`.
 - b. Cập nhật giá trị thuộc tính của một hoặc nhiều document: `updateOne`, `updateMany`.
 - c. Tìm kiếm và thay thế: `findOneAndUpdate`.

- d. Xóa một hoặc nhiều document: deleteOne, deleteMany, findOneAndDelete
- e. Thêm/xóa thuộc tính của document.
- f. Tìm kiếm theo mã: Tìm khách hàng, nhân viên, sản phẩm, hóa đơn khi biết mã số
- g. Xử lý kết quả tìm kiếm find với: count, size, limit, skip, explain, sort...
2. Tìm danh sách sản phẩm có giá cao nhất.
3. Tìm danh sách sản phẩm chưa bán được lần nào.
4. Thống kê số khách hàng theo từng bang.
+ getNumberCustomerByState() : Map<String, Integer>
5. Tính tổng tiền của đơn hàng khi biết mã số đơn hàng.
6. Đếm số đơn hàng của từng khách hàng.
+ getOrdersByCustomers() : Map<Customer, Integer>
7. Tính tổng số lượng của từng sản phẩm đã bán ra.
+ getTotalProduct(): Map<Product, Integer>
8. Dùng text search để tìm kiếm sản phẩm theo tên sản phẩm.
9. Tính tổng tiền của tất cả các hóa đơn trong một ngày nào đó.
10. Thêm dữ liệu vào từng collection.
11. Cập nhật giá của sản phẩm khi biết mã sản phẩm.
12. Xóa tất cả các khách hàng chưa mua hàng.
13. Danh sách các khách hàng có từ 2 số điện thoại trở lên
14. Thống kê tổng tiền hóa đơn theo tháng / năm.
15. Tìm khách hàng khi biết số điện thoại

BÀI TẬP 5

Database tên là sample_mflix. Viết các phương thức thực hiện các yêu cầu sau:

1. Liệt kê danh sách các đạo diễn có tham gia từ 30 bộ phim trở lên. Thông tin bao gồm: Tên đạo diễn (director) và số bộ phim.
Hướng dẫn: Đạo diễn dựa vào thuộc tính directors; output dạng: [{ director: 'Takashi Miike', 'number of movies': 34 }, ...]
2. Tìm tất cả các đạo diễn có tham gia đạo diễn nhiều bộ phim nhất
Hướng dẫn: Output là: [{ director: 'Woody Allen' }]
3. Liệt kê tựa phim (title) theo từng đạo diễn. Thông tin bao gồm: tên đạo diễn (director) và danh sách tựa phim
Hướng dẫn: Dùng Pivot Data; output dạng : output:[{ movies: ['Lost in America', 'Defending Your Life', 'Mother'], director: 'Albert Brooks'}, ...]
4. Thống kê số bộ phim đã phát hành theo từng năm, sắp xếp giảm dần theo năm
Hướng dẫn: năm dựa vào field released; output dạng:[{ 'number of movies': 10, year: 2016 }, ...]
5. Tìm năm phát hành nhiều bộ phim nhất.

Hướng dẫn: Năm dựa vào field released; output: [{ year: 2014 }]

6. Liệt kê danh sách các tựa phim (title) theo từng quốc gia. Thông tin bao gồm: tên quốc gia và danh sách tựa phim

Hướng dẫn: Tên quốc gia dựa vào thuộc tính countries; Dùng Pivot Data; output dạng:[{ movies: ['Bitter Sugar', 'Red Passport', 'Sugar', 'Jean Gentil', 'Kidnapped for Christ', 'Cèdigo Paz'], country: 'Dominican Republic' }, ...]

7. Đếm số bộ phim theo từng quốc gia, sắp xếp giảm dần theo số bộ phim. Thông tin bao gồm: Tên quốc gia và số bộ phim

Hướng dẫn: Tên quốc gia dựa vào thuộc tính countries; output dạng:[{ country: 'USA', 'number of movies': 11855 },]

8. Tìm những tựa phim (title) phát hành trong tháng 03 năm 2016

Hướng dẫn: Tháng và năm dựa vào thuộc tính released; output là: [{ title: 'Knight of Cups' }, { title: 'Sand Castles' }, { title: 'The Treasure' }]

9. Liệt kê những tựa phim (title) do diễn viên “Frank Powell” hoặc “Charles Wellesley” đóng

Hướng dẫn: Diễn viên dựa vào thuộc tính cast; output là : [{ title: 'A Corner in Wheat' }, { title: 'The Poor Little Rich Girl' }]

10. Tìm những quốc gia phát hành nhiều bộ phim nhất

Hướng dẫn: Tên quốc gia dựa vào thuộc tính countries; output là [{ country: 'USA' }]