



**Curso:** Mestrado Integrado em Informática

**U.C.:** Bases de Dados NoSQL

Folha de Exercícios FE05	
Docente	António Abelha / Hugo Peixoto
Tema:	Introdução ao MongoDB
Turma:	PL
Ano Letivo:	2019-2020 – 1º Semestre
Duração da aula:	2 horas

## 1. customers.json

- [1] Listar todas as bases de dados após a instalação.
- [2] Criar base de dados denominada *"customers"*.
- [3] Verificar a criação da base de dados.
- [4] Criar coleção denominada *"customers"*.
- [5] Validar a criação da coleção.
- [6] Criar um cliente com os seguintes características:  
first\_name: "John", last\_name: "Doe", age: 30
- [7] introduzir 2 clientes na coleção criada com as seguintes características:  
first\_name: "Steven", last\_name: "Williams", gender: "male"  
first\_name: "Mary", last\_name: "Troy", age: 19
- [8] Introduzir mais um cliente com as seguintes características:  
first\_name: "Ric", last\_name: "Foe", address: {street: "4 main st", city: "Boston"}
- [9] Criar um cliente com as seguintes características:  
first\_name: "Ana", last\_name: "Durant", degree: ["phD", "Msc"],  
address: {street: "4 Square Garden", city: "New York"}, age: 32
- [10] Criar um cliente com as seguintes características:  
first\_name: "Natalia", last\_name: "Will", age: 44, gender: "female"
- [11] Listar todos os clientes.
- [12] Listar todos os clientes usando a função pretty().
- [13] Efetuar uma atualização ao cliente 'Ric', colocar idade 45.
- [14] Encontrar todos os clientes que tenham 'Wil' no último nome.
- [15] Efetuar uma atualização ao cliente 'Steven', colocar idade 35.
- [16] Verificar se a idade da cliente 'Ana' é superior a 30 e se sim aumentar a idade em 10 anos.
- [17] O cliente 'Ric' quer que a sua idade seja removida da base de dados.
- [18] Procurar um cliente com o primeiro nome: "Jimmy" e atualizar para as seguintes características:  
first\_name: "Jimmy", last\_name: "Connors", age: 25, gender: male
- [19] Procurar todos os clientes com idade superior ou igual a 25.
- [20] Procurar todos os clientes sexo masculino.
- [21] Apagar o cliente cujo primeiro nome é "Mary".
- [22] Encontrar os clientes com o nome "Ana" ou "Ric".



## 2. restaurants.json

Após instalar o MongoDB, utilize a mongoshell ou Compass para executar os seguintes exercícios. Importe o ficheiro “*restaurants.json*” utilizando o Compass, ou recorrendo ao comando `mongoimport` da mongoshell.

Considere:

```
{
  "address": {
    "building": "1007",
    "coord": [ -73.856077, 40.848447 ],
    "street": "Morris Park Ave",
    "zipcode": "10462"
  },
  "borough": "Bronx",
  "cuisine": "Bakery",
  "grades": [
    { "date": { "$date": 1393804800000 }, "grade": "A", "score": 2 },
    { "date": { "$date": 1378857600000 }, "grade": "A", "score": 6 },
    { "date": { "$date": 1358985600000 }, "grade": "A", "score": 10 },
    { "date": { "$date": 1322006400000 }, "grade": "A", "score": 9 },
    { "date": { "$date": 1299715200000 }, "grade": "B", "score": 14 }
  ],
  "name": "Morris Park Bake Shop",
  "restaurant_id": "30075445"
}
```

Escreva uma query que...:

- [1] liste todos os documentos na coleção *restaurants*.
- [2] liste apenas os campos *restaurant\_id*, *name*, *borough* e *cuisine* para todos os documentos na coleção.
- [3] liste os campos *restaurant\_id*, *name*, *borough* e *cuisine* para todos os documentos na coleção, mas que exclua o campo *\_id*.
- [4] liste os campos *restaurant\_id*, *name*, *borough* e *zipcode* para todos os documentos na coleção, mas que exclua o campo *\_id*.
- [5] liste os restaurantes que estão localizados no bairro (*borough*) "Bronx".
- [6] liste os primeiros 5 restaurantes que estão localizados no bairro (*borough*) "Bronx".
- [7] liste os 5 restaurantes após dos primeiros 5 (do 6º ao 10º) que estão localizados no bairro (*borough*) "Bronx".
- [8] liste todos os restaurantes que têm uma pontuação (*score*) maior que 90.
- [9] liste todos os restaurantes que têm uma pontuação (*score*) maior que 80 mas menor que 100.
- [10] liste todos os restaurantes que estão localizados numa latitude (*coord.0*) menor que -95.754168.
- [11] liste todos os restaurantes cujo tipo de cozinha (*cuisine*) não seja "American ", que a sua pontuação (*score*) seja maior que 70 e a latitude (*address.coord.0*) menor que -65.754168.
- [12] liste todos os restaurantes cujo tipo de cozinha (*cuisine*) não seja "American ", que a sua pontuação (*score*) seja maior que 70 e a latitude (*address.coord.0*) menor que -65.754168. Não utilizando o operador \$and.
- [13] liste todos os restaurantes cujo tipo de cozinha (*cuisine*) não seja to tipo "American " e que tenham atingido uma classificação (*grade*) de "A" mas que não pertençam ao bairro (*borough*) de "Brookly". Deverá ser apresentada de acordo com o tipo de cozinha (*cuisine*) em ordem descendente.
- [14] liste todos os restaurantes que pertençam ao bairro (*borough*) "Bronx" e cujo tipo de cozinha (*cuisine*) seja quer "American" quer "Chinese".



Universidade do Minho  
Departamento de Informática

# FE05

- [15] liste todos os restaurantes cujas coordenadas (*address.coord*) sejam do tipo double (type: 1).
- [16] liste todos os restaurantes que contenham informação da rua (*address.street*).
- [17] liste todos os restaurantes de forma ascendente pelo tipo de cozinha (*cuisine*) e descendente pelo bairro (*borough*).
- [18] liste o *restaurant\_id*, *name*, *address* e localização geográfica (*coord*) para os restaurantes cujo segundo elemento do array da localização geográfica (*coord*) seja maior que 42 e até 52.
- [19] liste os restaurantes (*restaurante\_id*, *name*, *borough*, *cuisine*) que não conseguiram uma pontuação (*score*) maior que 10.
- [20] liste todos os restaurantes (*restaurante\_id*, *name*, *borough* e *cuisine*) que não pertencem ao bairro (*borough*) de "Staten Islan", ou "Queens" ou "Bronx" ou "Brooklin".