



INSERINDO DOCUMENTOS

```
db.albums.insert({  
  "nome" : "Master of Puppets",  
  "dataLancamento" : new Date(1986, 2, 3),  
  "duracao" : 3286  
})
```

```
db.albums.insert({  
  "nome" : "...And Justice for All",  
  "dataLancamento" : new Date(1988, 7, 25),  
  "duracao" : 3929  
})
```

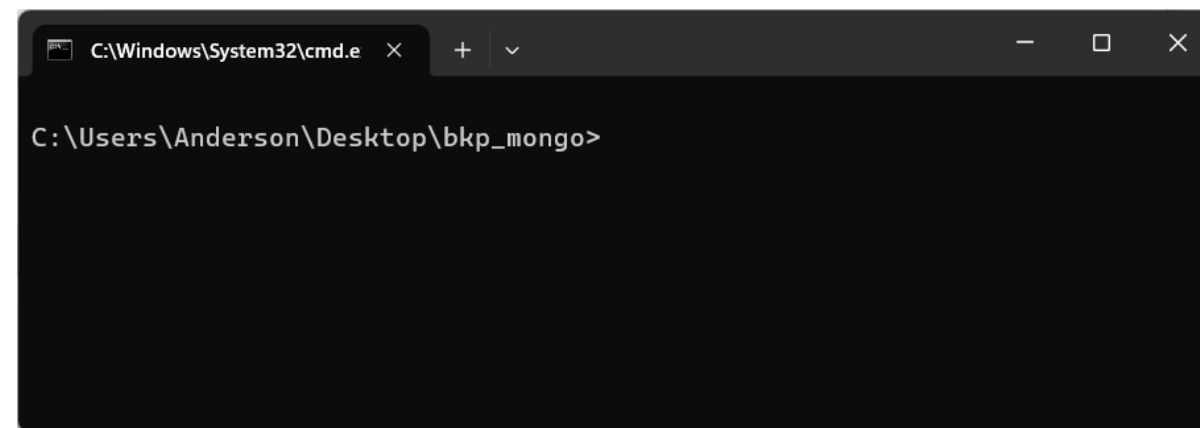
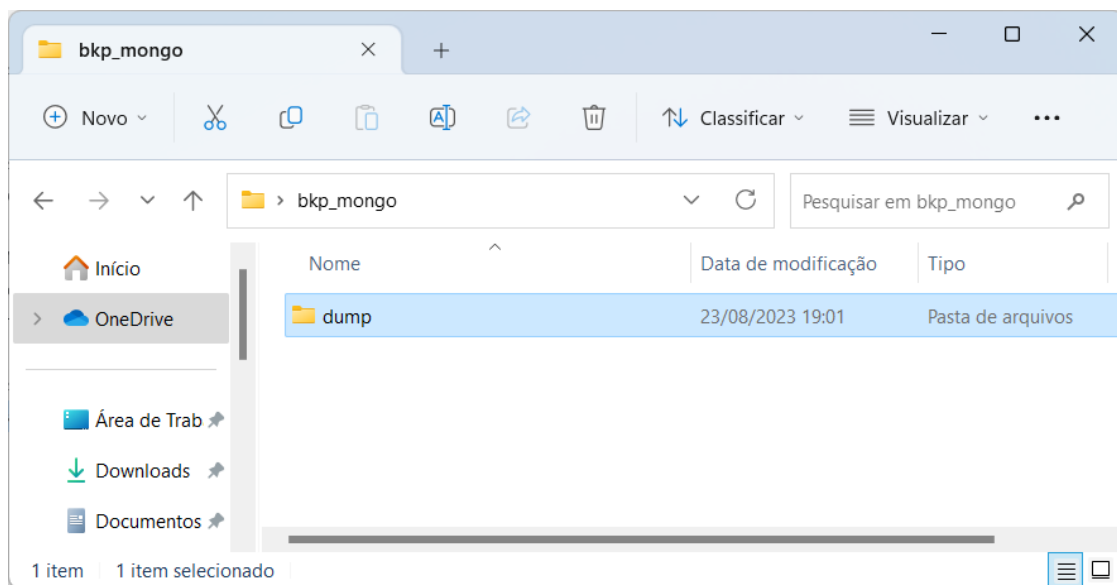
```
db.albums.insert({  
  "nome" : "Peace Sells... but Who's Buying?",  
  "duracao" : 2172,  
  "estudioGravacao" : "Music Grinder Studios",  
  "dataLancamento" : new Date(1986, 8, 19)  
})
```

```
db.albums.insert({  
  "nome" : "Reign in Blood",  
  "dataLancamento" : new Date(1986, 9, 7),  
  "artistaCapa" : "Larry Carroll",  
  "duracao" : 1738  
})
```

```
db.albums.insert({  
  "nome" : "Among the Living",  
  "produtor" : "Eddie Kramer"  
})
```

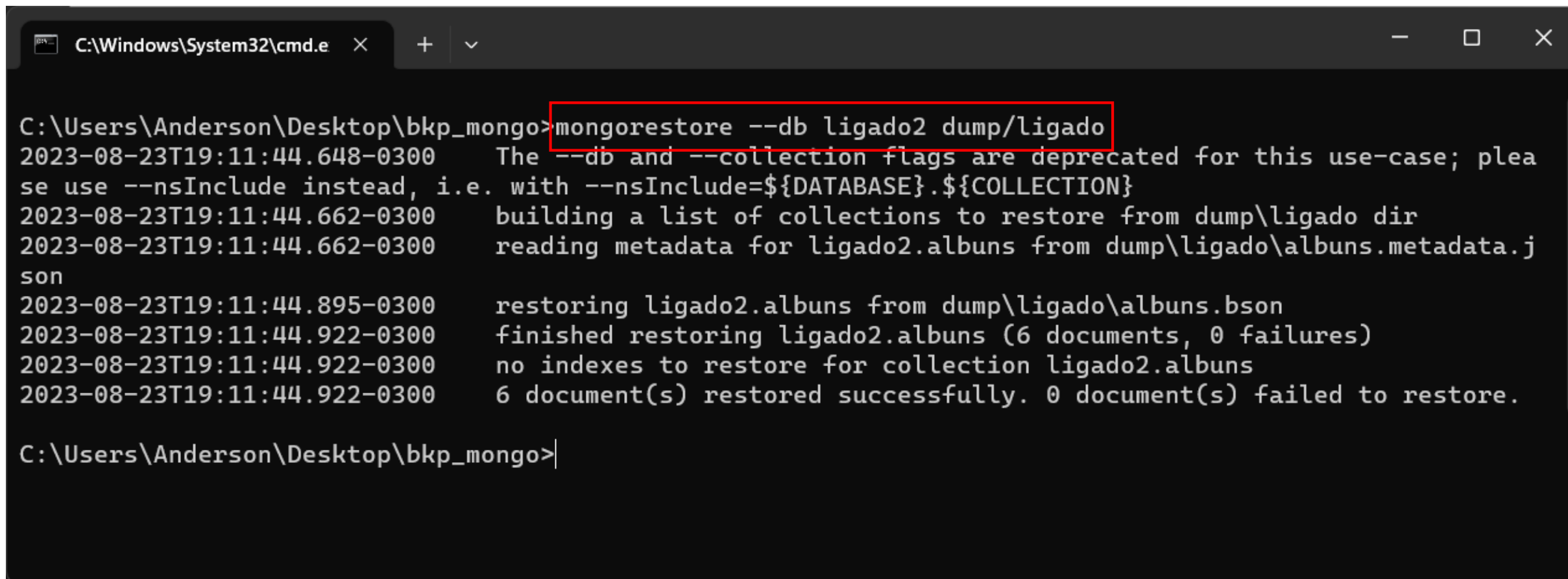
IMPORTANDO UM BANCO DE DADOS NO MONGODB

Abra a pasta na qual estão os arquivos do banco de dados exportado anteriormente e abra um prompt de comando (cmd).



IMPORTANDO UM BANCO DE DADOS NO MONGODB

Digite o comando: **mongorestore --db ligado2 dump/ligado**



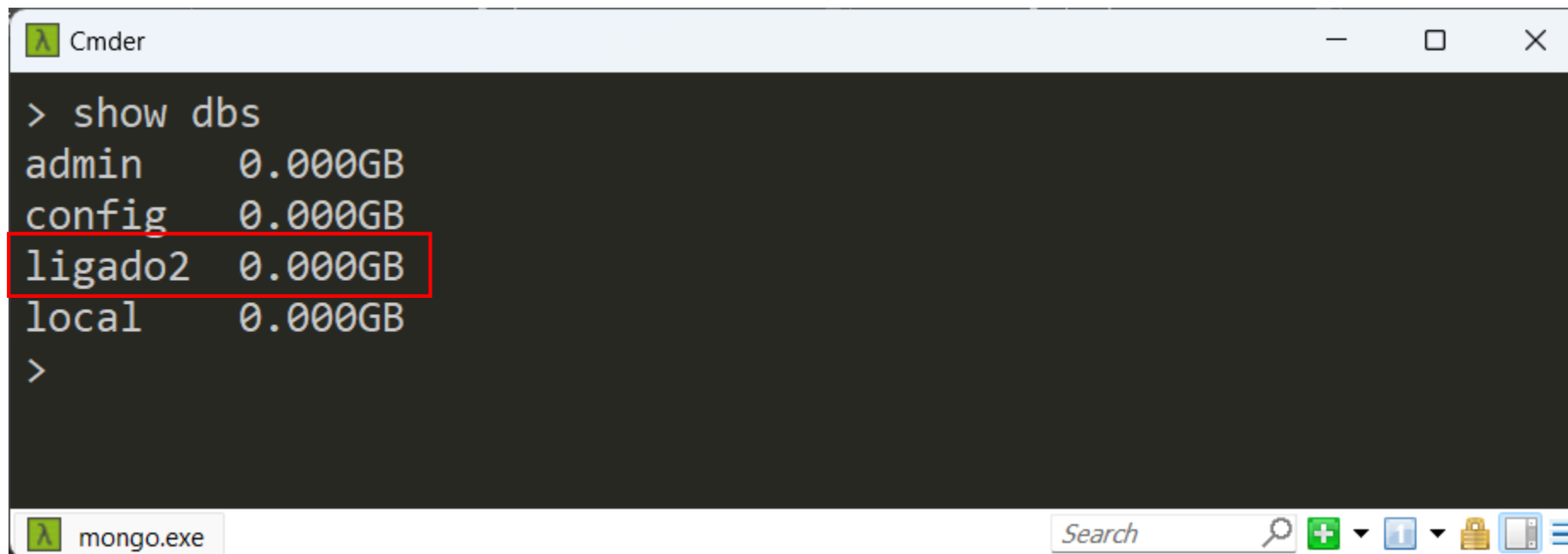
```
C:\Windows\System32\cmd.e  X  +  v

C:\Users\Anderson\Desktop\bkp_mongo>mongorestore --db ligado2 dump/ligado
2023-08-23T19:11:44.648-0300 The --db and --collection flags are deprecated for this use-case; please use --nsInclude instead, i.e. with --nsInclude=${DATABASE}.${COLLECTION}
2023-08-23T19:11:44.662-0300 building a list of collections to restore from dump\ligado dir
2023-08-23T19:11:44.662-0300 reading metadata for ligado2.albuns from dump\ligado\albuns.metadata.json
2023-08-23T19:11:44.895-0300 restoring ligado2.albuns from dump\ligado\albuns.bson
2023-08-23T19:11:44.922-0300 finished restoring ligado2.albuns (6 documents, 0 failures)
2023-08-23T19:11:44.922-0300 no indexes to restore for collection ligado2.albuns
2023-08-23T19:11:44.922-0300 6 document(s) restored successfully. 0 document(s) failed to restore.

C:\Users\Anderson\Desktop\bkp_mongo>
```

VERIFICANDO A IMPORTAÇÃO

Digite o comando: **show dbs**



```
λ Cmder
> show dbs
admin      0.000GB
config     0.000GB
ligado2    0.000GB
local      0.000GB
>
```

The screenshot shows a terminal window titled 'Cmdr' with a dark background. The command prompt is at the top, followed by the command 'show dbs'. The output lists four databases: 'admin' (0.000GB), 'config' (0.000GB), 'ligado2' (0.000GB), and 'local' (0.000GB). The 'ligado2' entry is highlighted with a red rectangular box. At the bottom of the terminal, there is a taskbar with a 'mongo.exe' icon and a search bar.

RESUMO DE COMANDOS DA AULA

- **MOSTRA BANCOS**

```
show dbs
```

- **CRIANDO UM BANCO**

```
use bancoteste
```

- **MOSTRA BANCOS**

```
show dbs
```

(o banco criado não aparece pois não tem nenhuma collection criada nele)

- **VERIFICANDO EM QUE BD ESTOU**

```
db
```

- **CRIANDO UMA COLLECTION**

uma collection é como se fosse cada registro de um BD relacional. Aqui chamamos de documentos

```
db.documentos.insertOne({nome:"Anderson",idade:48})
```

- **FAZENDO UMA PESQUISA SIMPLES**

```
db.documentos.find()
```

RESUMO DE COMANDOS DA AULA

- CRIANDO OUTRA COLLECTION

```
db.documentos.insertOne({nome:"Fulano",idade:48,profissao:"PROFESSOR"})
```

- FAZENDO UMA PESQUISA SIMPLES

```
db.documentos.find()
```

- FAZENDO UMA PESQUISA POR NOME

```
db.documentos.find({nome:"Fulano"})
```

- VISUALIZANDO OS DADOS INSERIDOS

```
db.documentos.find()
```

- MELHORANDO A VISUALIZAÇÃO

```
db.documentos.find().pretty()
```

- FAZENDO UM BKP DO BANCO DE DADOS

criar uma pasta, entrar pelo cmd nesta pasta e executar o comando:

```
mongodump --db bancoteste
```

RESUMO DE COMANDOS DA AULA

- APAGANDO UM BANCO DE DADOS

entrar no prompt do mongo e selecionar o banco desejado
use bancoteste

em seguida usar o comando
`db.dropDatabase()`

- IMPORTANDO UM BANCO DE DADOS

entrar em um prompt de comando do windows na pasta onde esta o BKP do Banco
`mongorestore --db bancoteste2 dump/bancoteste`

REALIZAR AS ETAPAS ANTERIORES E VERIFICAR OS ARQUIVOS IMPORTADOS NOVAMENTE

BUSCANDO DOCUMENTOS NO MONGODB

Além da busca por igualdade, é possível fazer buscas mais complexas no MongoDB utilizando os operadores de comparação disponíveis, equivalentes a "maior que", "menor que", "diferente", entre outros.

BUSCANDO DOCUMENTOS NO MONGODB

Nome	Descrição
\$gt	Corresponde a valores que são maiores que o valor específico na query.
\$gte	Corresponde a valores que são maiores ou iguais ao valor específico na query.
\$in	Corresponde a quaisquer valores que existem em um array específico em uma query.
\$lt	Corresponde a valores que são menores que o valor específico na query.
\$lte	Corresponde a valores que são menores ou iguais que o valor específico na query.
\$ne	Corresponde a todos os valores que não são iguais ao valor específico na query.
\$nin	Corresponde a valores que não existem em um array específico da query.

BUSCANDO DOCUMENTOS NO MONGODB

A sintaxe para utilizar esses operadores é {"nomeDoCampo" : {"operador" : " valor "}} . Por exemplo, para buscar os álbuns com duração menor que 30 minutos, procuraremos os documentos cujo campo duracao seja menor que 1800 . Estamos armazenando a duração em segundos, por isso usamos o valor 1800, que é o equivalente a 60 multiplicado por 30.

BUSCANDO DOCUMENTOS NO MONGODB

```
db.albums.find({"duracao" : {"$lt" : 1800}})
```

```
{ "_id" : ObjectId("54c6c49e91b5bfb09cb9194b"),  
  "nome" : "Reign in Blood",  
  "dataLancamento" : ISODate("1986-10-07T03:00:00Z"),  
  "artistaCapa" : "Larry Carroll", "duracao" : 1738 }
```

```
SELECT *  
FROM albums a  
WHERE a.duracao < 1800
```

BUSCANDO DOCUMENTOS NO MONGODB

Agora que estamos começando a nos familiarizar com a interface de buscas do MongoDB, podemos tentar montar a query que retorna todos os álbuns lançados em 1986. A maneira mais simples para fazer esse tipo de busca por intervalos, seja de datas ou de números, é aplicando dois filtros: um onde o campo deve ser maior ou igual que o início do intervalo E menor que o fim do intervalo.

BUSCANDO DOCUMENTOS NO MONGODB

Nome	Descrição
\$and	Junta <i>query clauses</i> com uma lógica E retorna todos os documentos que combinam com ambas condições.
\$nor	Junta <i>query clauses</i> com uma lógica NEM retorna todos os documentos que falham em combinar ambas as condições.
\$not	Inverte o efeito de uma <i>query expression</i> e retorna os documentos que não combinam com a condição.
\$or	Junta <i>query clauses</i> com uma lógica OU retorna todos os documentos que combinam com ambas condições.

BUSCANDO DOCUMENTOS NO MONGODB

Os filtros que queremos executar são: data de lançamento maior ou igual que 01/01/1986 — que é `{"dataLancamento" : {$gte : new Date(1986, 0, 1)}}` — E data de lançamento menor que 01/01/1987 — ou `{"dataLancamento" : {$lt : new Date(1987, 0, 1)}}` . No final, teremos:

```
> db.albums.find(  
  {$and : [{"dataLancamento" : {$gte : new Date(1986, 0, 1)}},  
           {"dataLancamento" : {$lt : new Date(1987, 0, 1)}}]}  
)
```


BUSCANDO DOCUMENTOS NO MONGODB

Para a nossa busca de um intervalo de datas, o resultado seria:

```
> db.albuns.find({"dataLancamento" :  
    {"$gte" : new Date(1986, 1, 1),  
    "$lt"   : new Date(1987, 1, 1)}})
```

REMOVENDO DOCUMENTOS NO MONGODB

Para remover o álbum cujo nome é "...And Justice for All" basta usar o função remove no lugar de find

```
> db.albuns.remove({"nome": "...And Justice for All"})  
WriteResult({ "nRemoved" : 1 })
```

ALTERANDO DOCUMENTOS NO MONGODB

Vamos aproveitar e adicionar a duração no álbum "Among the Living" da seguinte maneira:

```
> db.albuns.update({"nome" : "Among the Living"},  
                  {"duracao" : 3013})  
WriteResult({ "nMatched" : 1,  
              "nUpserted" : 0,  
              "nModified" : 1 })
```

ALTERANDO DOCUMENTOS NO MONGODB

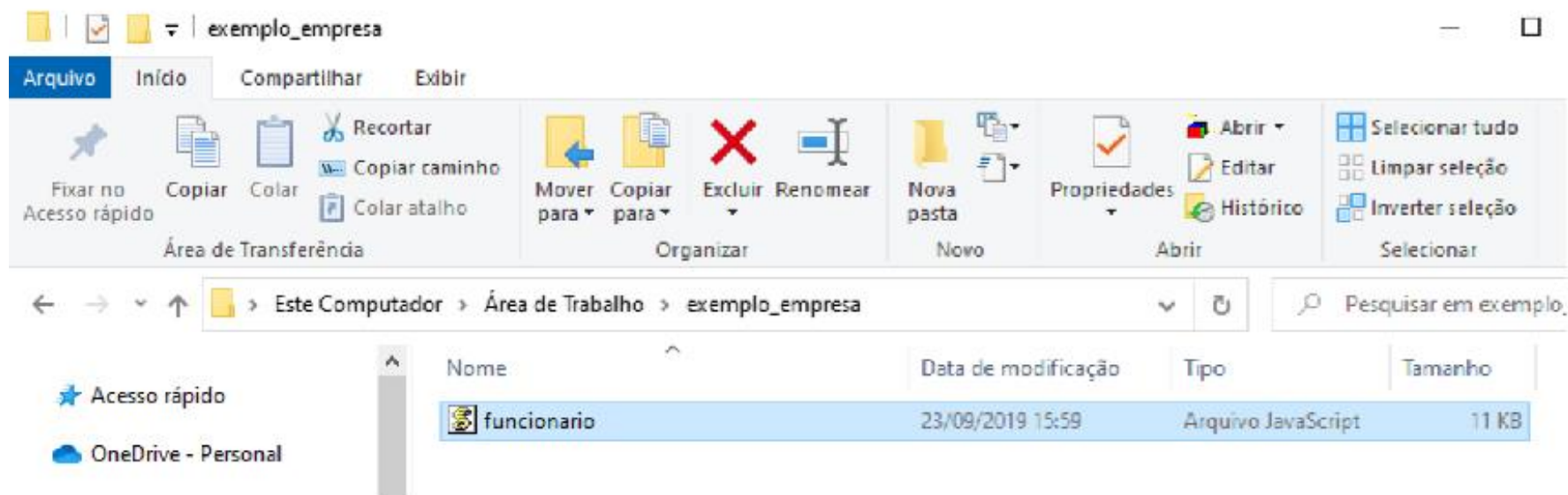
```
> db.albuns.find({"_id" : ObjectId("54c828b5342dda43e93bee1e")})
{ "_id" : ObjectId("54c828b5342dda43e93bee1e"),
  "duracao" : 3013 }
```

```
> db.albuns.update(
  {"_id" : ObjectId("54c828b5342dda43e93bee1e")},
  {$set : {"nome" : "Among the Living",
           "produtor" : "Eddie Kramer"}})
```

```
WriteResult({ "nMatched" : 1,
               "nUpserted" : 0,
               "nModified" : 1 })
```

```
> db.albuns.find({"_id" : ObjectId("54c828b5342dda43e93bee1e")})
{ "_id" : ObjectId("54c828b5342dda43e93bee1e"),
  "duracao" : 3013,
  "nome" : "Among the Living",
  "produtor" : "Eddie Kramer" }
```

IMPORTANDO DE UM ARQUIVO JS



```
C:\Users\Anderson\Desktop\exemplo_empresa>mongo <funcionario.js
```

Observação: Utilizar a base de dados Empresa disponível no Github [aqui](#)

EXERCÍCIOS

Utilizar o método `find()` para realizar as consultas das Questões 1 a 12.

1. Selecionar os nomes e os endereços dos funcionários que são programadores e trabalham no departamento 3.
2. Selecionar os funcionários com função de programador e de analista que ganham acima de R\$ 2.000,00. Classificar em ordem crescente pela função e nome do funcionário.
3. Selecionar os funcionários com salário maior do que R\$ 2.600,00 que sejam analistas, ou tenham exatamente um filho. Mostrar o nome, a função, o salário, a quantidade de filhos e o bairro onde mora do funcionário. Ordenar em ordem decrescente pelo bairro.

EXERCÍCIOS

4. Quais os funcionários possuem hobby e não têm filhos? Mostrar o nome, o hobby e o salário.
5. Quais os funcionários possuem filhos, recebe salário maior do que R\$ 1.500,00 e tem como hobby futebol ou xadrez? Mostrar o nome, a quantidade de filhos, os hobbies e o salário.
6. Quais os funcionários têm como hobby futebol e tênis de mesa? Não mostrar o salário e a data de admissão.
7. Quais os funcionários têm 4 hobbies, sendo que pelo menos 1 hobby seja natação? Mostrar o nome e os hobbies.

EXERCÍCIOS

8. Quais os funcionários que não ocupem a função de secretária, não tem tênis de mesa como hobby e é do departamento 3?
9. Quantos funcionários do departamento 3 ganham entre R\$ 1.200,50 e R\$ 1.600,00?
10. Quantos funcionários moram em Vitória da Conquista e possuem algum hobby?
11. Recuperar os funcionários que possuem os três maiores salários, excluindo o funcionário com o maior salário. Não mostrar os hobbies, as notas, as avaliações, os feedbacks e o endereço. Utilizar os métodos limit e skip.

EXERCÍCIOS

12. Quais os funcionários têm 3 ou 4 feedbacks? Mostrar o nome e os feedbacks ordenado em ordem decrescente pelo nome.