

## Mãos na massa: Armazenando e consultando os dados

Chegou a hora de você executar o que foi visto na aula! Para isso, execute os passos listados abaixo.

---

1) Ainda com o script **Prepara\_e\_Armazena.R** aberto no **RStudio**, e com o *data frame* **df\_OVNI** em memória, de acordo com o que foi feito na última aula, instale o *package* **mongolite** e coloque-o em memória:

```
install.packages('mongolite')  
library(mongolite)
```

2) Antes de mais nada, **no terminal/prompt de comando do seu sistema operacional**, rode o servidor do MongoDB:

**Importante:** O diretório abaixo é o mesmo utilizado no vídeo, então você deve modificá-lo para o seu diretório de dados específico.

```
mongod --dbpath C:\MongoDB_Store\data\db
```

3) Caso você já tenha o *database* **ovni** criado e queira deletá-lo, execute o comando abaixo, novamente **no terminal/prompt de comando do seu sistema operacional**:

```
mongo ovni --eval "db.dropDatabase()" --quiet
```

4) Voltando ao **RStudio**, no seu console, faça a conexão com o MongoDB:

```
m <- mongo("ovnis", url = "mongodb://localhost:27017/ovni")
```

Esse objeto **m** é que fará a ponte com o MongoDB.

5) Ainda no **RStudio**, no seu console, com a conexão feita, insira os dados:

```
inserted_OVNIs <- m$insert(df_OVNI)
```

6) Voltando ao **terminal/prompt de comando do seu sistema operacional**, você pode se conectar à base de dados:

```
mongo  
use ovni
```

E contar quantos documentos a coleção **ovnis** contém:

```
db.ovnis.count()
```

Além disso, você pode ver todos os OVNI's, ordenados por tipo:

```
db.ovnis.find( {}, { _id: 0 } ).sort ( { Shape : 1 } )
```

A saída é muito confusa, então utilize uma interface gráfica, o **Robomongo**.

7) Então, abra o **Robomongo**, se conecte ao servidor e acesse, no *database* **ovnis**, a coleção **ovni**. Clique com o botão direito do mouse e selecione **View Documents**. Faça novamente a consulta acima, clique em **Run** e veja todos os OVNI's, ordenados por tipo.

8) Ainda no **Robomongo**, veja quantas ocorrências há por estado, desta vez utilizando o método **aggregate**, do MongoDB:

```
db.ovnis.aggregate ( [ { $group : { _id : "$State" , Views : { $sum : 1 } } }, { $sort: { Views : -1
```



9) Você também pode fazer consultas pelo R, com o `m` criado mais acima, que é um ponteiro para a conexão do MongoDB, utilizando o método `find`. Por exemplo, para buscar todas as ocorrências da cidade Phoenix:

```
m$find ( '{"City": "Phoenix" }' )
```

10) Você também pode criar novos *data frames* baseados nos dados do MongoDB. Por exemplo, as ocorrências do estado da Califórnia:

```
ca <- m$find ( '{"State": "CA" }' )
```

11) Você pode ver quantas observações e variáveis esse *data frame* tem, suas primeiras observações e visualizar esses dados em forma tabular:

```
dim(ca)  
head(ca)  
View(ca)
```