



# **IBD-016 – BANCO DE DADOS - NÃO RELACIONAL**

---

Prof. Me. Anderson Vanin



# O sistema na maneira tradicional

[https://pt.wikipedia.org/wiki/Lista\\_de\\_ganhadores\\_do\\_Pr%C3%AAmio\\_IgNobel](https://pt.wikipedia.org/wiki/Lista_de_ganhadores_do_Pr%C3%AAmio_IgNobel)



The screenshot shows the Wikipedia page for the Ig Nobel Prize winners list. The page title is "Lista de ganhadores do Prêmio IgNobel". The page content includes a list of winners from 1991 to 2009, with the 1991 winners listed below the introduction. The page also features a sidebar with a table of contents and a top navigation bar with links for "Artigo", "Discussão", "Ler", "Editar", "Ver histórico", and "Ferramentas".

**Lista de ganhadores do Prêmio IgNobel** 13 línguas

Artigo Discussão Ler Editar Ver histórico Ferramentas

Origem: Wikipédia, a enciclopédia livre.

Esta página **cita fontes**, mas que **não cobrem todo o conteúdo**. Ajude a **inserir referências**. Conteúdo não verificável pode ser **removido**.—*Encontre fontes: ABW • Google (N • L • A)* (Outubro de 2019)

Este artigo carece de **reciclagem de acordo com o livro de estilo**. Sinta-se livre para editá-lo(a) para que este(a) possa atingir um **nível de qualidade superior**. (Janeiro de 2011)

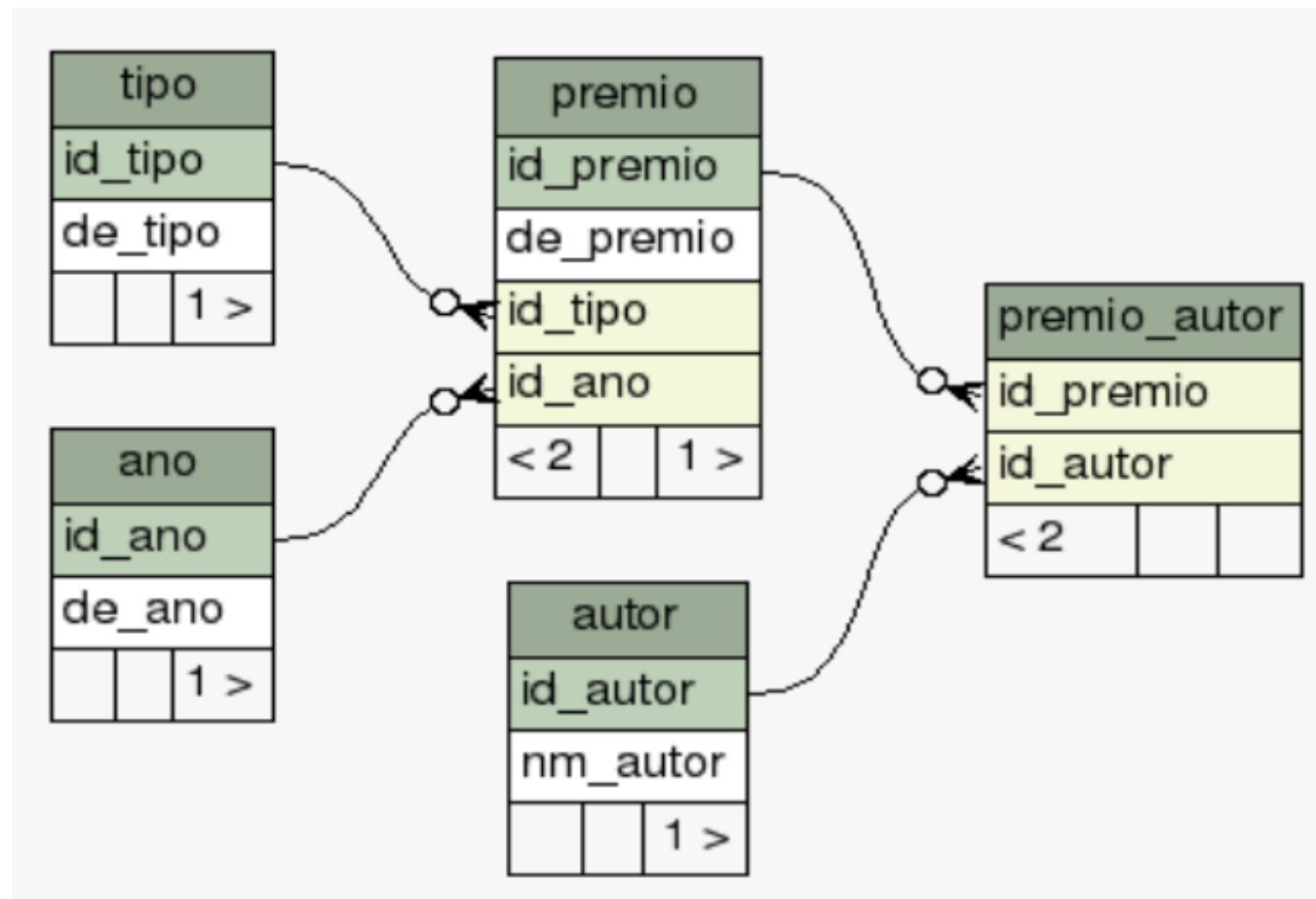
Esta é uma **lista de vencedores do Prêmio IgNobel**, desde a primeira edição em 1991. O **IgNobel** é um prêmio dado em uma cerimônia anual, no mês de setembro, na **Universidade de Harvard**, aos autores de pesquisas, experimentos e outras atividades inusitadas, nas diversas áreas da ciência. Todos os prêmios foram dados por fatos reais (exceto três em 1991 e um em 1994, devido a erros de divulgação). O lema do evento é *"primeiro fazer as pessoas rirem, e depois pensarem"*.<sup>[1]</sup>

**1991**

- Química:** **Jacques Benveniste**, produtivo e dedicado correspondente da revista *Nature*, pela persistência na descoberta de que a **água** é um líquido inteligente e por demonstrar que ela é capaz de se lembrar de acontecimentos muito depois de todos os vestígios desses acontecimentos terem desaparecido.
- Medicina:** **Alan Kligerman**, idealizador do alívio digestivo utilizando vapor e inventor do **Beano**,<sup>[2]</sup> pelo seu trabalho pioneiro com líquidos antigás, que previnem inchaço, **flatulência**, desconforto e embarços.
- Educação:** **James Danforth Quayle** (na época, vice-presidente dos EUA), tido como de "baixo **QI**", com declarações confusas, demonstrando assim, mais do que ninguém, a necessidade de educação científica.<sup>[3]</sup>
- Biologia:** **Robert Klerk Graham**, seleccionador de **sementes** e **profeta** da **reprodução**, pelo seu desenvolvimento pioneiro de um "**Banco para Seleção Embrionária**" ("*Repository for Germinal Choice*"), que aceita doações apenas de laureados com o **Nobel** e atletas olímpicos.
- Economia:** **Michael Robert Milken**, gigante da **Wall Street** e pai do **Junk Bond**,<sup>[4]</sup> para quem o mundo está endividado.
- Literatura:** **Erich von Däniken**, narrador visionário e autor do livro *Eram os Deuses Astronautas?*, por explicar como a civilização humana foi

# O sistema na maneira tradicional

Pela lista da Wikipedia, conseguimos separar quatro informações: **ano**, **tipo**, **autor** e **descrição do prêmio**. Basicamente, pegamos as quatro informações e criamos uma tabela para cada um, e como um prêmio IgNobel pode ter vários autores, criamos uma tabela auxiliar **premio\_autor**.



# O sistema na maneira tradicional

Vamos listar algumas reflexões sobre esse modelo:

- 1) ao montarmos o modelo, pensamos em desnormalizar toda informação, isolando em quatro tabelas distintas;
- 2) como um mesmo prêmio pode ter vários autores, precisamos criar uma tabela auxiliar ***prêmio\_autor***;
- 3) montamos toda estrutura baseada nas limitações de um banco de dados (no mundo real não existe representação da tabela auxiliar ***prêmio\_autor***);
- 4) não pensamos no que a aplicação vai fazer, pensamos apenas em arrumar os dados;
- 5) em caso de lentidão, revemos os SQLs e criamos índices.

# O sistema na maneira tradicional

Para exibir uma página como da Wikipedia, é preciso fazer uma consulta envolvendo todas as tabelas criadas:

```
select
p.de_premio, t.de_tipo, a.de_ano , au.nm_autor
from premio p, tipo t, ano a, premio_autor pa, autor au
where p.id_premio = pa.id_premio
and p.id_tipo = t.id_tipo
and p.id_ano = a.id_ano
and pa.id_autor = au.id_autor
```

# O sistema na maneira tradicional

Como a página da Wikipedia tem muitos acessos, se eles tivessem feito da maneira convencional, o site com certeza não seria tão rápido.

Portanto, **pensando na aplicação e não nos dados**, o ideal seria que tudo estivesse organizado de acordo com a necessidade do negócio e não com formas normais do mundo relacional.

Se a página da Wikipedia **exibe tudo de uma vez**, o correto seria a informação estar **concentrada em apenas um lugar (uma tabela)**. Essa prática já é conhecida no mundo do **Data Warehouse**, chamada de **desnormalização**.

**No MongoDB, organizamos os dados em função da aplicação.**

# Resumindo

Da maneira convencional, a sua aplicação obedece às regras do seu banco de dados; no MongoDB é o contrário: é a sua aplicação que manda e os dados são organizados conforme a necessidade do sistema



# MongoDB

```
{  
  "ano" : 1992,  
  "tipo" : "Medicina",  
  "autores" : [  
    "F. Kanda",  
    "E. Yagi",  
    "M. Fukuda",  
    "K. Nakajima",  
    "T. Ohta",  
    "O. Nakata"],  
  "premio" : "Elucidación dos Componentes Químicos Responsáveis  
    pelo Chulé do Pé (Elucidation of Chemical  
    Compounds Responsible for Foot Malodour),  
    especialmente pela conclusão de que as pessoas  
    que pensam que têm chulé, têm, e as que pensam  
    que não têm, não têm."  
}
```

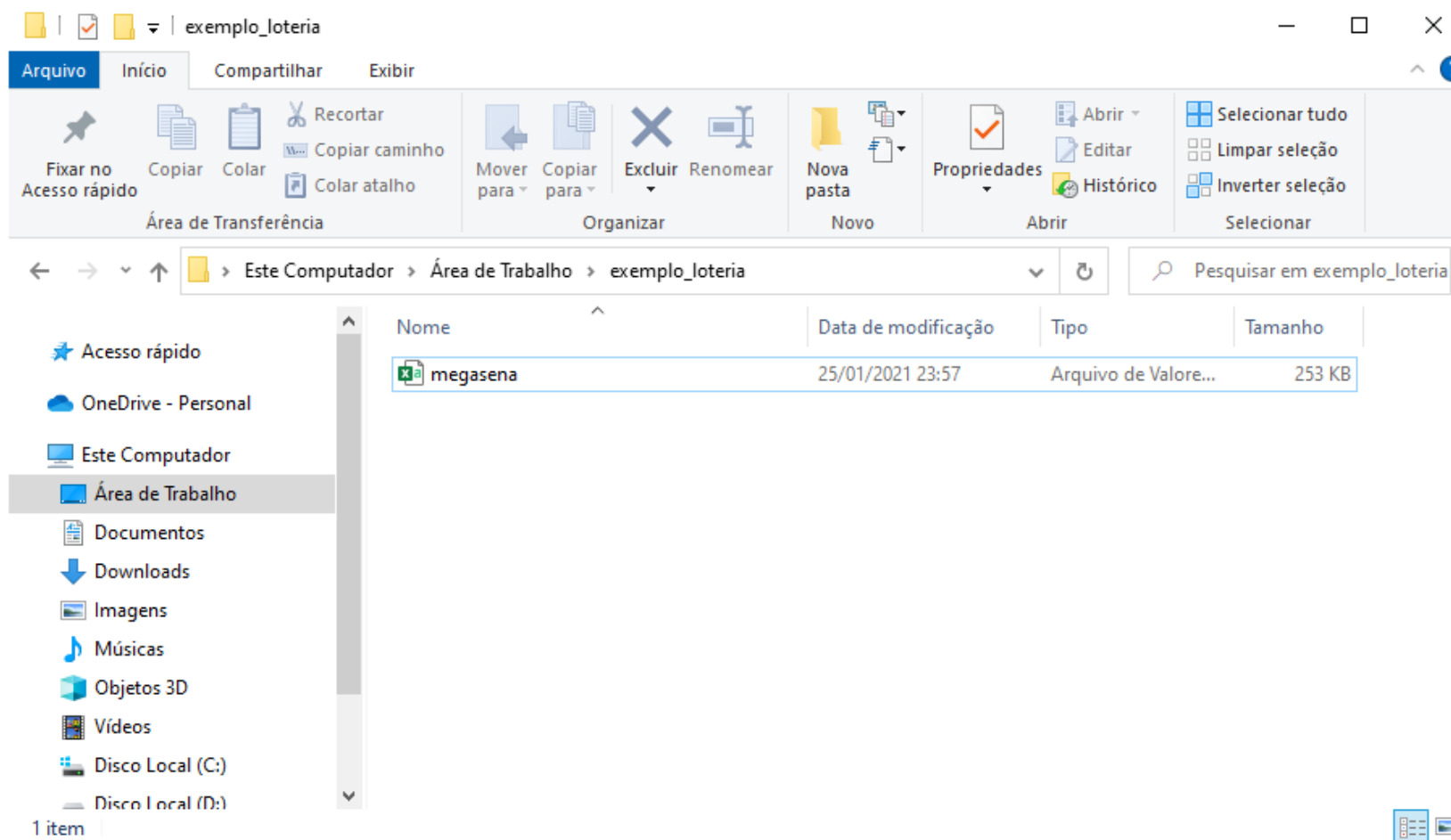


# Assim...

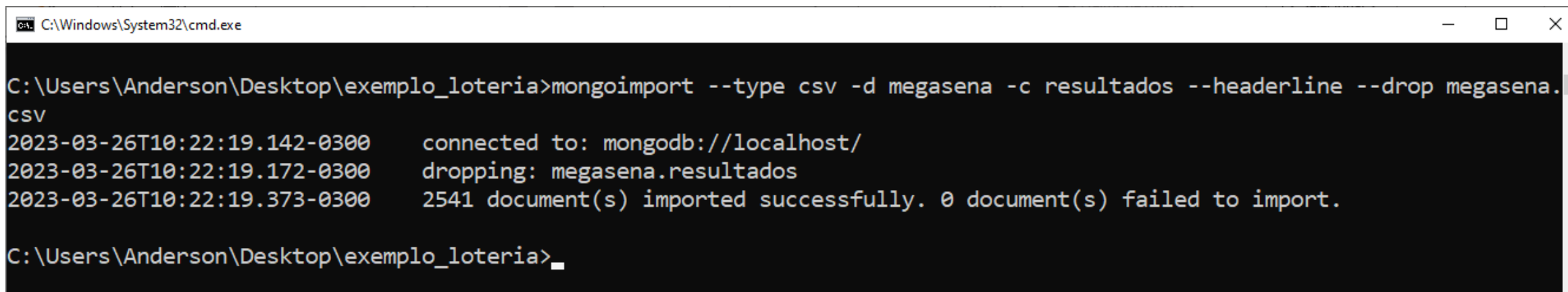
Assim, em um único registro temos todas as informações de que precisamos. Resumindo:

- De quantas **tabelas** precisamos para exibir um prêmio? **Cinco.**
- De quantas **collections** precisamos para exibir um prêmio? **Uma.**

<https://github.com/boaglio/mongodb-casadocodigo>



mongoimport --type csv -d megasena -c resultados --headerline --drop megasena.csv



```
C:\Windows\System32\cmd.exe

C:\Users\Anderson\Desktop\exemplo_loteria>mongoimport --type csv -d megasena -c resultados --headerline --drop megasena.csv
2023-03-26T10:22:19.142-0300    connected to: mongodb://localhost/
2023-03-26T10:22:19.172-0300    dropping: megasena.resultados
2023-03-26T10:22:19.373-0300    2541 document(s) imported successfully. 0 document(s) failed to import.

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

show dbs

```
> show dbs
admin          0.000GB
bancofatec     0.000GB
config         0.000GB
local          0.000GB
megasena       0.000GB
primeirobanco 0.000GB
segundobanco   0.000GB
```

Verificando e alterando para o BD desejado.

```
> db
test
> use megasena
switched to db megasena
```

Verificando todas as collections do BD megasena.

```
> show collections  
resultados
```



Para conferir a quantidade de registros importados: **db.resultados.count()**

```
> db.resultados.count()  
2541
```

# Analogia de uso

- MongoDB:

```
db.megasena.count()
```

- SQL Relacional:

```
select count(*) from megasena
```

# Pesquisando find()

```
> db.resultados.find().pretty()
{
  "_id" : ObjectId("6420470bd01ed8f300b57279"),
  "Concurso" : 1,
  "Data Sorteio" : "11/03/1996",
  "1ª Dezena" : 4,
  "2ª Dezena" : 5,
  "3ª Dezena" : 30,
  "4ª Dezena" : 33,
  "5ª Dezena" : 41,
  "6ª Dezena" : 52,
  "Arrecadacao_Total" : 0,
  "Ganhadores_Sena" : 0,
  "Cidade" : "",
  "UF" : "",
  "Rateio_Sena" : 0,
  "Ganhadores_Quina" : 17,
  "Rateio_Quina" : "39158,92",
  "Ganhadores_Quadra" : 2016,
  "Rateio_Quadra" : "330,21",
  "Acumulado" : "SIM",
  "Valor_Acumulado" : "1714650,23",
  "Estimativa_Prêmio" : 0,
  "Acumulado_Mega_da_Virada" : 0
}
```

# Ganhadores de SP

```
db.resultados.find({"UF": "SP"}).pretty()
```

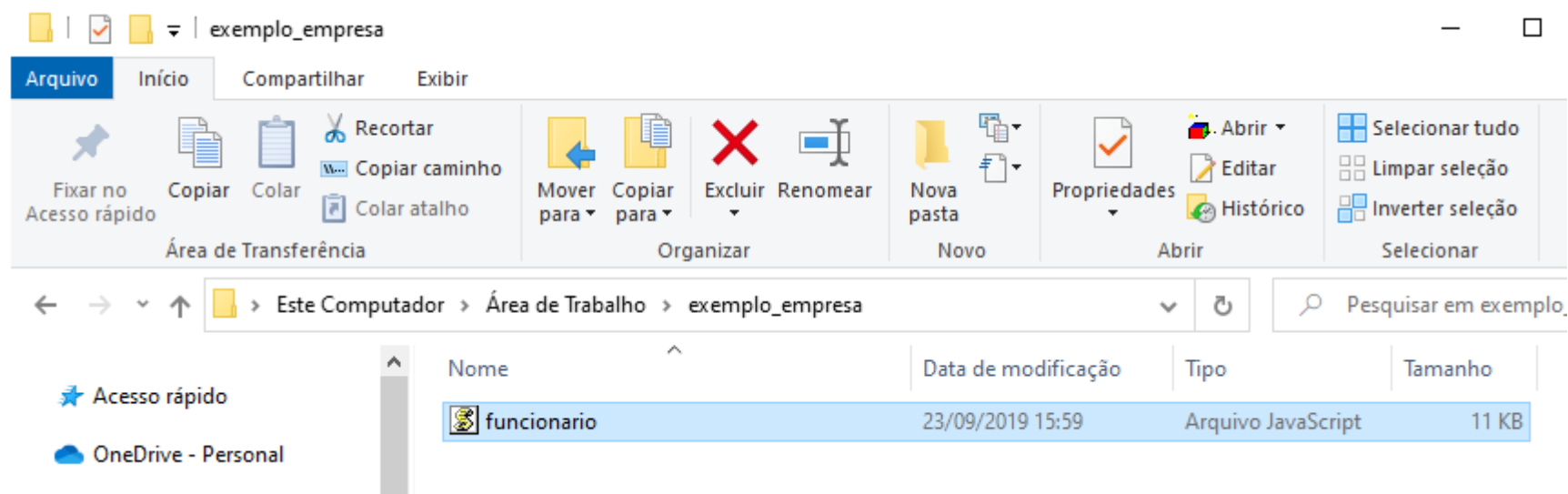
```
> db.resultados.find({"UF": "SP"}).pretty()
{
  "_id" : ObjectId("6420470bd01ed8f300b5727a"),
  "Concurso" : "",
  "Data Sorteio" : "",
  "1ª Dezena" : "",
  "2ª Dezena" : "",
  "3ª Dezena" : "",
  "4ª Dezena" : "",
  "5ª Dezena" : "",
  "6ª Dezena" : "",
  "Arrecadacao_Total" : "",
  "Ganhadores_Sena" : "",
  "Cidade" : "",
  "UF" : "SP",
  "Rateio_Sena" : "",
  "Ganhadores_Quina" : "",
  "Rateio_Quina" : "",
  "Ganhadores_Quadra" : "",
  "Rateio_Quadra" : "",
  "Acumulado" : "",
  "Valor_Acumulado" : "",
  "Estimativa_Prêmio" : "",
  "Acumulado_Mega_da_Virada" : ""
}
```

# Total de Ganhadores de SP

```
db.resultados.find({"UF": "SP"}).pretty().count()
```

```
> db.resultados.find({"UF": "SP"}).pretty().count()  
224
```

# Importando de um JS



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

# Importando de um JS

funcionario - Bloco de Notas

Arquivo Editar Formatar Exibir Ajuda

```
// =====  
// Banco de Dados EMPRESA  
// =====  
  
// Cria o banco de dados empresa  
use empresa  
  
// Exclui banco de dados selecionado  
// db.dropDatabase();  
  
// Exclui a coleção funcionario  
// db.funcionario.drop();  
  
// Remove todos os documentos da coleção funcionario  
db.funcionario.remove({});  
  
// Insere um documento na coleção funcionario  
db.funcionario.insertOne({  
  "_id" : NumberInt(1),  
  "nome" : "CARLOS MACEDO CERRI",  
  "depto" : NumberInt(3),  
  "funcao" : "VENDEDOR",  
  "salario" : 1530.00,  
  "admissao" : new ISODate('2010-05-25T20:06:20.123Z'),  
  "casado" : true,  
  "skill" : "EXCELENTE RELACIONAMENTO INTERPESSOAL",  
  "endereco" : {  
    "logradouro" : "AVENIDA PERNAMBUCO",  
    "bairro" : "BRASIL",  
    "uf" : "BA",  
    "numero" : NumberInt(102),  
    "cidade" : "VITÓRIA DA CONQUISTA"
```



```
> db.funcionario.find()
{ "_id" : 1, "nome" : "CARLOS MACEDO CERRI", "depto" : 3, "funcao" : "VENDEDOR", "salario" : 1530, "admissao" : ISODate("2010-05-25T20:06:20.123Z"), "casado" : true, "skill" : "EXCELENTE RELACIONAMENTO INTERPESSOAL", "endereco" : { "logradouro" : "AVENIDA PERNAMBUCO", "bairro" : "BRASIL", "uf" : "BA", "numero" : 102, "cidade" : "VITÓRIA DA CONQUISTA", "cep" : "45000-000" }, "hobbies" : [ "TÊNIS DE MESA", "YOGA", "VIAGEM" ], "notas" : [ { "criterio" : "PRODUÇÃO", "nota" : 4.8 }, { "criterio" : "CONVÍVIO", "nota" : 9.2 } ], "avaliacoes" : [ 73, 87, 77 ], "feedbacks" : [ true, true ] }
```

**Seleciona os nomes e os endereços dos funcionários que são programadores e trabalham no departamento 3.**

```
> db.funcionario.find({"funcao":"PROGRAMADOR","depto":3},{nome:1,endereco:1})
{ "_id" : 7, "nome" : "WILSON CAMPOS MACEDO", "endereco" : { "logradouro" : "AVENIDA TANCREDO NEVES", "bairro" : "CENTRO", "uf" : "BA", "numero" : 355, "cidade" : "VITÓRIA DA CONQUISTA", "cep" : "45000-000" } }
{ "_id" : 8, "nome" : "AUGUSTO AGUIAR SOUZA", "endereco" : { "logradouro" : "AVENIDA GETÚLIO VARGAS", "bairro" : "VILA NOVA", "uf" : "BA", "numero" : 92, "cidade" : "POÇÕES", "cep" : "45260-000" } }
>
```