Aula 12: MongoDB

# O grande problema da persistência

## O grande problema de persistência: arquivos

Até este ponto do curso, trabalhamos com um sistema de arquivos para fazer nossos armazenamentos principais.

No entanto, sabemos claramente que existem alguns problemas com o uso de arquivos, como:

* Ter que atualizar todo o arquivo quando fazemos uma alteração.
* Ter que ler todo o arquivo quando procuramos alguns dados.
* Sem proteção ao querer mover ou modificar algo.

## A necessidade de mudança

Talvez você não tenha percebido tais problemas até agora, porém, quando começamos a trabalhar com centenas, milhares ou milhões de dados, realmente nos deparamos com um grande problema de performance.

É por isso que uma solução mais otimizada para trabalhar com grandes volumes de dados começou a ser buscada: surgiram os bancos de dados.

## Base de dados

Um banco de dados nada mais é do que uma coleção organizada de dados. Esses dados devem compartilhar algum contexto e são armazenados para serem posteriormente convertidos em informações úteis para serem usadas dentro de um sistema.

A base de dados será apenas responsável por armazenar esses dados, o uso que damos a eles posteriormente não depende mais dela.

### Por que um banco de dados é útil?

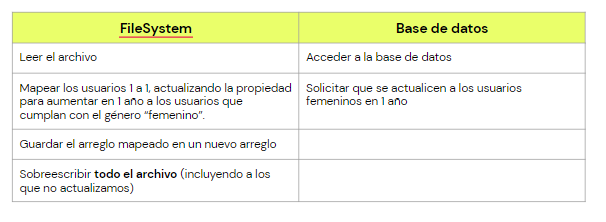
Algumas das coisas que podemos apontar sobre a utilidade de um banco de dados são:

* **Armazenamento mais seguro** : os dados que residem em um banco de dados não são modificáveis diretamente, portanto, não podem ser alterados tão facilmente.
* **Segmentação de dados:** Podemos separar os dados em "contextos", permitindo assim separar os dados de interesse.
  + Separe clientes em potencial de apenas clientes interessados.
  + Separe os produtos existentes dos produtos fora de estoque
  + Separe diferentes usuários por gênero, nacionalidade, plano, etc.
* **Gestão simples uma vez configurada** : Uma vez definidos os principais esquemas de nosso banco de dados, podemos realizar operações sobre esses dados, como:
  + Filtro
  + Ordem
  + Pesquise dados específicos.
  + Atualize um conjunto de dados sem afetar ou tocar em outros dados.

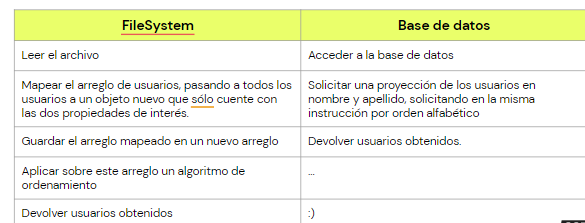
Quão trabalhoso é fazer isso com arquivos?

## Comparação entre FileSystem e Database

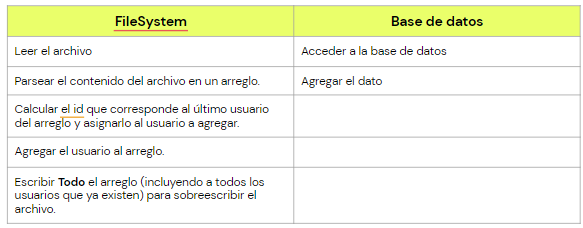
### Tarefa: Aumentar a idade de todos os usuários do sexo feminino em 1 ano



### Tarefa: Retornar apenas o nome e o sobrenome de todos os usuários, classificados em ordem alfabética



### Tarefa: Adicionar um novo usuário com um ID de incremento automático



# Modelo relacional e não relacional

## A necessidade de diferentes modelos de banco de dados

Uma vez que entendemos que o banco de dados nos ajuda a manter os dados organizados, é hora de entender quando usar um modelo relacional ou um modelo não relacional.

Um **banco de dados relacional** refere-se à estrutura, relacionamento, dependência e mudança controlada.

Um **banco de dados não relacional** refere-se a algo menos estruturado, com relacionamentos e dependências mais flexíveis e mudanças extremamente rápidas.

## modelo relacional

Refere-se a modelos de dados em que são necessárias estruturas mais firmes e rígidas nos dados. Além disso, eles são usados em dados mais controlados

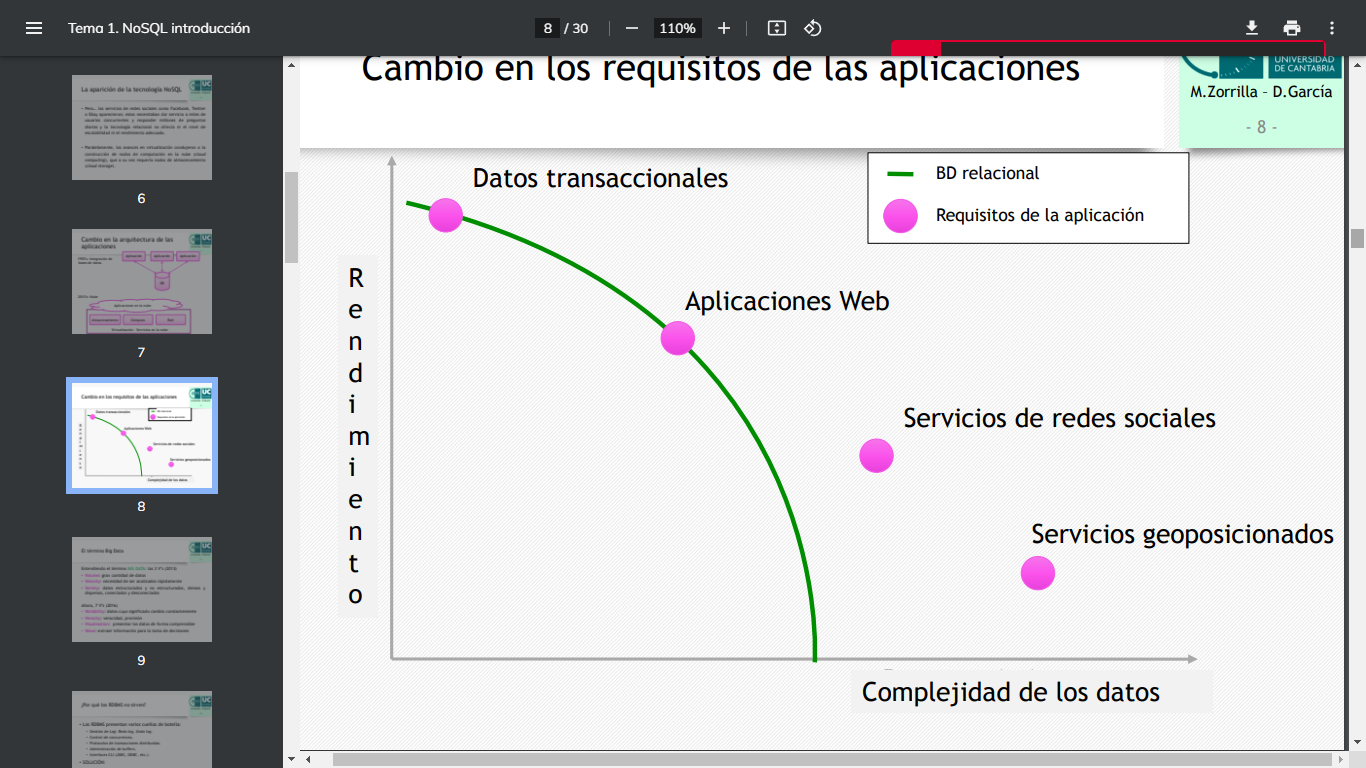
* Eles contam com tabelas, colunas e linhas para gerenciar seus dados.
* Eles permitem que tabelas sejam conectadas a partir de “relacionamentos” baseados em chaves primárias e estrangeiras.

É usado para dados de controle, como:

* sistemas bancários
* Sistemas climáticos (não tempo).
* sistemas de filme.
* Sua linguagem é SQL (Structured Query Language)
* Alguns sistemas SQL são:
  + postgresql
  + Oráculo
  + mysql
  + MariaDB

## O problema dos bancos de dados relacionais

Quando as aplicações que precisamos aumentam seus requisitos, os dados mudam mais rápido e ficam mais complexos, ficam mais inconsistentes e por isso nosso banco de dados relacional começa a ficar lento.



## Solução: introduzir um modelo mais flexível

Um modelo é desenvolvido onde os dados são mais flexíveis, tanto na estrutura quanto na associação.

Tudo isso para criar dados projetados para desempenho, não para consistência imediata.

## modelo não relacional

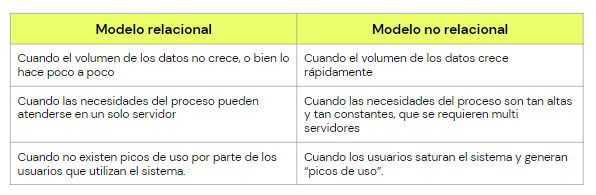
A flexibilidade dos dados torna-os consideravelmente mais rápidos em termos de acessibilidade.

* Pode ser baseado em:
  + valor chave
  + documentos
  + gráficos
  + memória
* São bancos de dados muito úteis para organizar e gerenciar informações não estruturadas, ou quando não há uma ideia clara dos dados a serem armazenados.
* Alto grau de escalabilidade e desempenho
* Não usa SQL como linguagem
* Alguns sistemas NoSQL são
  + MongoDBGenericName
  + Redis
  + DynamoDBName

**Importante!**

**O fato de bancos de dados não relacionais serem utilizados em aplicações modernas com maior mobilidade de dados não significa que os bancos de dados relacionais não funcionem mais. Os bancos de dados relacionais continuam sendo amplamente exigidos em determinados setores do mercado (sistemas bancários, sistemas de filmes, clima, etc.)**

## Quando usar cada modelo?



# Investigando bancos de dados não relacionais: MongoDB

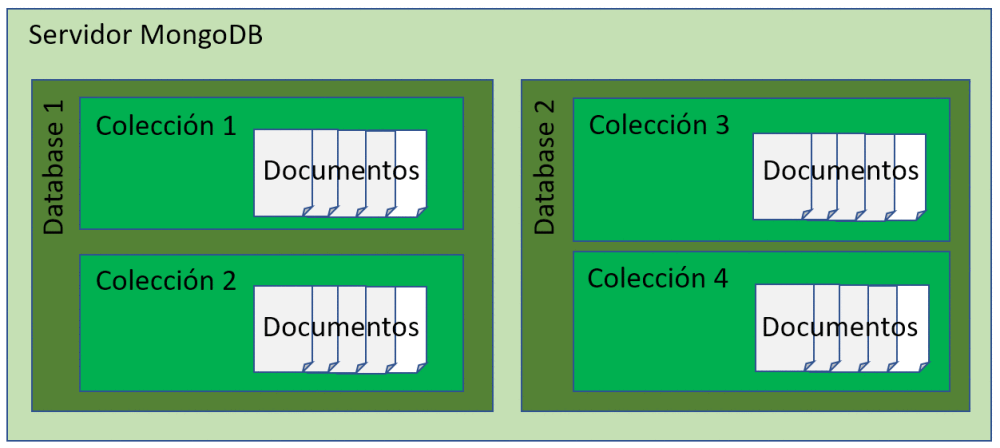
## MongoDBGenericName

* Banco de dados não relacional orientado a documentos
* Em vez de tabelas, ele opta por usar coleções,
* Cada documento que inserimos em uma coleção pode ter uma estrutura diferente
* Pode ser usado localmente ou na nuvem

## MongoDB: recursos

* Armazena dados em documentos flexíveis semelhantes a JSON: a estrutura de dados pode mudar com o tempo.
* O modelo de documento é atribuído a objetos no código do aplicativo para facilitar o trabalho com os dados.
* Consultas ad hoc, indexação e agregação em tempo real oferecem maneiras poderosas de acessar e analisar dados.
* O MongoDB é um banco de dados distribuído em seu núcleo, portanto, alta disponibilidade, escalabilidade horizontal e distribuição geográfica são integrados e fáceis de usar.
* O uso do MongoDB é gratuito.

## MongoDB: arquitetura



## documentos

Uma das grandes vantagens de um documento é que ele se baseia no conceito de chave-valor, isso, como você sabe, é muito semelhante a um objeto como o que você trabalhou por tanto tempo em javascript

Eles não são propriamente um "objeto" para chamá-los dessa forma, mas o MongoDB trabalha com uma extensão dos arquivos conhecida como BSON. Isso é o que realmente permanece no banco de dados.

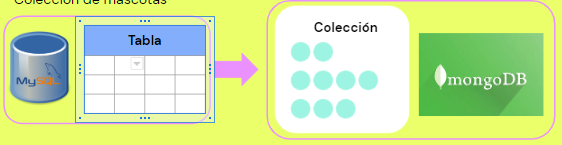
Os esquemas de um banco de dados em MongoDB, com a ajuda de elementos como mangusto, são facilmente manipulados, pois permite que sejam definidos com uma estrutura quase idêntica à de um objeto.

Use para controles deslizantes de texto e imagem. Se não for suficiente, não sobrecarregue, use outro com o mesmo título para indicar que o mesmo módulo continua.

## Coleções no MongoDB

Sempre que pensarmos em um grupo de dados, nos referiremos a ele como uma coleção, na qual armazenaremos cada documento individual (semelhante a uma tabela com seus registros).

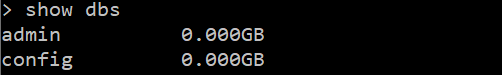
* coleção de usuários
* Coleção de produtos
* coleção de animais de estimação



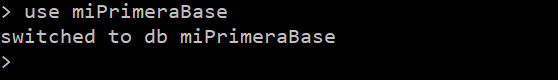
# Depois de instalar o MongoDB... vamos tentar comandos que podem ser úteis para você

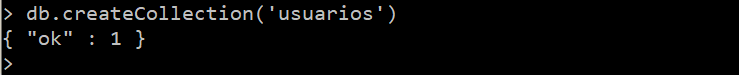
## Comandos iniciais: criação

* show dbs mostrará a lista de bancos de dados que temos ativos.



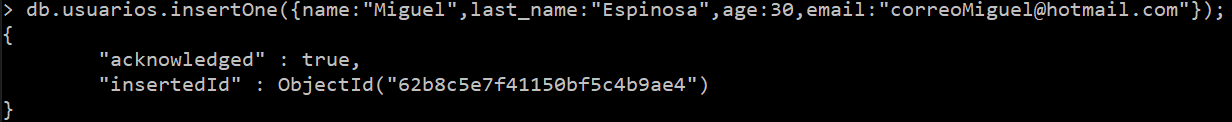
* use <nome> cria um novo banco de dados (caso não exista) e se posiciona nele para utilizá-lo.



* db.createCollection(name) criará nossa primeira coleção para começar a alimentar os documentos.
* 

## Inserindo um dado na coleção "users"

Uma vez que saibamos qual coleção vamos usar, usaremos o comando “insertOne”, passando o objeto que queremos salvar na dita coleção:



Observe como é simples! a sintaxe base.collection.method é bastante fácil de lembrar, porque nos lembra dos métodos que frequentemente usamos ao trabalhar com objetos e classes em javascript.

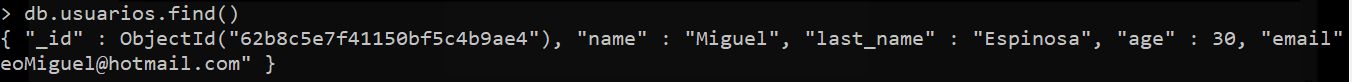
Além disso, o documento inserido é um objeto com propriedades chave-valor, o que nos facilita a visualização e o entendimento do que estamos inserindo.

Agora é necessário visualizar as informações.

## 

## Lendo informações do nosso banco de dados

Usando a mesma sintaxe de um objeto ou classe javascript, lemos a coleção com o método find().



Percebemos como o resultado é o documento que acabamos de inserir na coleção. Sem arquivos, sem problemas!

Além disso, o MongoDB autogerencia um id para nós, o que garantirá que tenhamos uma chave irrepetível, evitando que tenhamos que programar nosso próprio gerenciamento de identificadores.