



# Fundamentos, Benefícios e Aplicações Práticas



**Ricardo Mello**

Senior Software Engineer, MongoDB Community  
Creator, Oracle Certified Java Developer





# Agenda

- Relational vs. NoSQL
- Introduction to MongoDB
- Data Structure
- Data Modeling
- Basic Operations
- Aggregation Operations
- Components and Tools
  - Compass
  - Atlas
- Hands-On
- What's Next?



# Relational vs. NoSQL

- SQL - Structured Query Language
- Modelo relacional
- Pouco recurso em disco



# Relational vs. NoSQL

id	firstName	lastName	email	phone
1	Ricardo	Mello	ricardohsmello@gmail.com	16992..
2	Maria	Silva	maria@gmail.com	15887..



# Relational vs. NoSQL

id	firstName	lastName	email	email	phone	phone
1	Ricardo	Mello	ricardohsmello@gmail.com	ricas@hotmail.com	16992..	
2	Maria	Silva	maria@gmail.com		15887..	163429..



# Relational vs. NoSQL

id	firstName	lastName	email	email	phone	phone
1	Ricardo	Mello	ricardohsmello@gmail.com	ricas@hotmail.com	16992..	
2	Maria	Silva	maria@gmail.com		15887..	163429..
3	Henrique	Marques				



# Relational vs. NoSQL

```
_id: ObjectId('66561014ad9ee41ee0be83e0')
firstName: "Ricardo"
lastName: "Mello"
▼ email: Array (2)
  0: "ricardohsmello@gmail.com"
  1: "ricas@hotmail.com"
▼ phone: Array (1)
  0: 1699214
```

```
_id: ObjectId('665610d8ad9ee41ee0be83e2')
firstName: "Maria"
lastName: "Silva"
▼ email: Array (1)
  0: "maria@gmail.com"
▼ phone: Array (2)
  0: 15887
  1: 163429
```

```
_id: ObjectId('6656116fad9ee41ee0be83e3')
firstName: "Henrique"
lastName: "Marques"
```



# Relational vs. NoSQL

	SQL Databases	MongoDB
Data storage model	Tables	Collections
Schemas	Yes	No
Scaling	Vertical	Horizontal
Joins	Yes	No





# Relational vs. NoSQL

	SQL Databases	MongoDB
Data storage model	Tables, columns	
Schemas		
Scaling		
Joins		



# Relational vs. NoSQL

	SQL Databases	MongoDB
Data storage model	Tables, columns	JSON
Schemas		
Scaling		
Joins		



# Relational vs. NoSQL

	SQL Databases	MongoDB
Data storage model	Tables, columns	JSON
Schemas	Rigid	
Scaling		
Joins		



# Relational vs. NoSQL

	SQL Databases	MongoDB
Data storage model	Tables, columns	JSON
Schemas	Rigid	Flexible
Scaling		
Joins		



# Relational vs. NoSQL

	SQL Databases	MongoDB
Data storage model	Tables, columns	JSON
Schemas	Rigid	Flexible
Scaling	Vertical (scale-up with a larger server)	
Joins		



# Relational vs. NoSQL

	SQL Databases	MongoDB
Data storage model	Tables, columns	JSON
Schemas	Rigid	Flexible
Scaling	Vertical (scale-up with a larger server)	Horizontal (scale-out across commodity servers)
Joins		



# Relational vs. NoSQL

	SQL Databases	MongoDB
Data storage model	Tables, columns	JSON
Schemas	Rigid	Flexible
Scaling	Vertical (scale-up with a larger server)	Horizontal (scale-out across commodity servers)
Joins	Typically required	



# Relational vs. NoSQL

	SQL Databases	MongoDB
Data storage model	Tables, columns	JSON
Schemas	Rigid	Flexible
Scaling	Vertical (scale-up with a larger server)	Horizontal (scale-out across commodity servers)
Joins	Typically required	Typically not required





# Relational vs. NoSQL

Relacional	MongoDB



# Relational vs. NoSQL

Relacional	MongoDB
Table	



# Relational vs. NoSQL

Relacional	MongoDB
Table	Collection



# Relational vs. NoSQL

Relacional	MongoDB
Table	Collection
Row	



# Relational vs. NoSQL

Relacional	MongoDB
Table	Collection
Row	Document



# Relational vs. NoSQL

Relacional	MongoDB
Table	Collection
Row	Document
Column	



# Relational vs. NoSQL

Relacional	MongoDB
Table	Collection
Row	Document
Column	Field



# Relational vs. NoSQL

Relacional	MongoDB
Table	Collection
Row	Document
Column	Field
Primary Key	





# Relational vs. NoSQL

Relacional	MongoDB
Table	Collection
Row	Document
Column	Field
Primary Key	_id



# Introduction to MongoDB

MongoDB é um banco de dados NoSQL orientado a documentos, desenvolvido pela empresa MongoDB Inc.



# Introduction to MongoDB

1. ***Schema Flexível:*** Permite que documentos dentro de uma coleção tenham diferentes estruturas, facilitando a evolução do modelo de dados.



# Introduction to MongoDB

1. ***Schema Flexível***: Permite que documentos dentro de uma coleção tenham diferentes estruturas, facilitando a evolução do modelo de dados.
2. ***Alta Escalabilidade***: Suporta sharding (distribuição horizontal de dados), o que permite o gerenciamento eficiente de grandes volumes de dados.



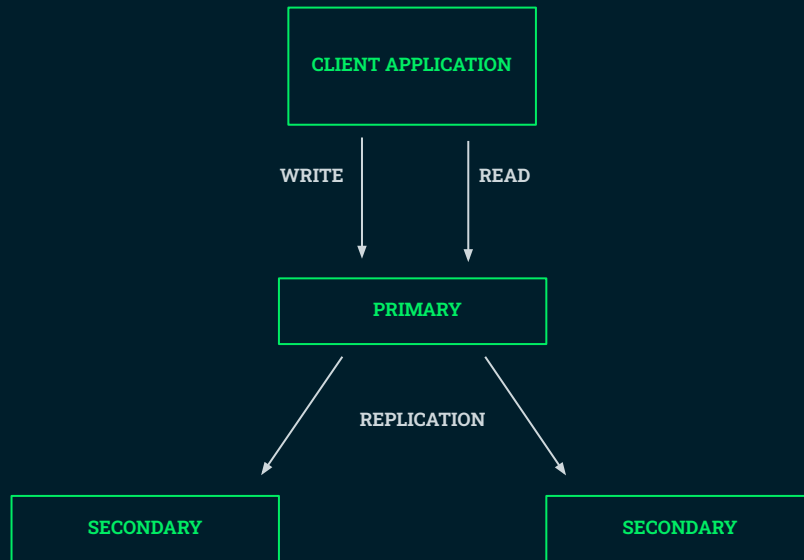
# Introduction to MongoDB

1. ***Schema Flexível***: Permite que documentos dentro de uma coleção tenham diferentes estruturas, facilitando a evolução do modelo de dados.
2. ***Alta Escalabilidade***: Suporta sharding (distribuição horizontal de dados), o que permite o gerenciamento eficiente de grandes volumes de dados.
3. ***Replicação***: Oferece replicação de dados automática para alta disponibilidade e recuperação de desastres.



# Introduction to MongoDB

## Replica Set



# Data Structure



1. **Definição:** A unidade básica de dados no MongoDB é o documento, que é um objeto JSON-like (internamente armazenado em BSON - Binary JSON).
2. **Estrutura:** Um documento contém um conjunto de pares chave-valor. Cada chave é uma string e cada valor pode ser um tipo de dado básico (como string, número, data) ou um documento aninhado (subdocumento) ou um array.
3. **Flexibilidade:** Os documentos dentro de uma coleção não precisam ter a mesma estrutura ou campos, permitindo que dados heterogêneos coexistem na mesma coleção.

# Data Structure



```
{
  "_id": ObjectId("507f191e810c19729de860ea"),
  "name": "Alice",
  "age": 29,
  "address": {
    "street": "123 Main St",
    "city": "Springfield",
    "state": "IL"
  },
  "interests": ["reading", "traveling", "cooking"]
}
```





# Data Modeling

*To develop your schema, instead of thinking about our database, think about the application instead*

# Data Modeling



*To develop your schema, instead of thinking about our database, think about the application instead*

- O que meu aplicativo faz?
- Que dados irei armazenar?
- Como os usuários irão acessar esses dados?
- Quais dados serão mais valiosos para mim?

# Data Modeling



*To develop your schema, instead of thinking about our database, think about the application instead*

- O que meu aplicativo faz? Qual é a aparência dos meus dados
- Que dados irei armazenar? As relações entre os dados
- Como os usuários irão acessar esses dados? as ferramentas que você planeja ter
- Quais dados serão mais valiosos para mim? Os padrões de acesso que podem surgir

# Data Modeling



- Modelagem de dados é o processo de definir como os dados são armazenados.
- Definição dos relacionamentos que existem entre diferentes entidades nos seus dados.

# Data Modeling



## Tipos de relacionamentos de dados

- **1:1** one-to-one
- **1:Many** one-to-many
- **Many:Many** many-to-many

## As duas formas principais de modelar relacionamentos de dados

- **Embedding**
- **Referencing**

# Data Modeling



## Referencing

```
{
  "_id": ObjectId("507f191e810c19729de860eb"),
  "street": "123 Main St",
  "city": "Springfield",
  "state": "IL"
}
```

Address

```
{
  "_id": ObjectId("507f191e810c19729de860ea"),
  "name": "Alice",
  "age": 29,
  "address_id": ObjectId("507f191e810c19729de860eb")
}
```

Person



# Data Modeling

## Embedding

```
{  
  "_id": ObjectId("507f191e810c19729de860ea"),  
  "name": "Alice",  
  "age": 29,  
  "address": {  
    "street": "123 Main St",  
    "city": "Springfield",  
    "state": "IL"  
  }  
}
```

Person



## Referencing

Vantagens	Desvantagens





## Referencing

Vantagens	Desvantagens
Documentos menores	



## Referencing

Vantagens	Desvantagens
Documentos menores	
Evita duplicação	



## Referencing

Vantagens	Desvantagens
Documentos menores	Performance durante leitura
Evita duplicação	



## Referencing

Vantagens	Desvantagens
Documentos menores	Performance durante leitura
Evita duplicação	Consultas adicionais



## Embedding

**Vantagens**

**Desvantagens**




## Embedding

Vantagens	Desvantagens
Desempenho em leituras	



## Embedding

Vantagens	Desvantagens
Desempenho em leituras	
Consistência de dados	



## Embedding

Vantagens	Desvantagens
Desempenho em leituras	Redundância de dados
Consistência de dados	





## Embedding

Vantagens	Desvantagens
Desempenho em leituras	Redundância de dados
Fácil manipulação	Tamanho do documento

# Data Modeling

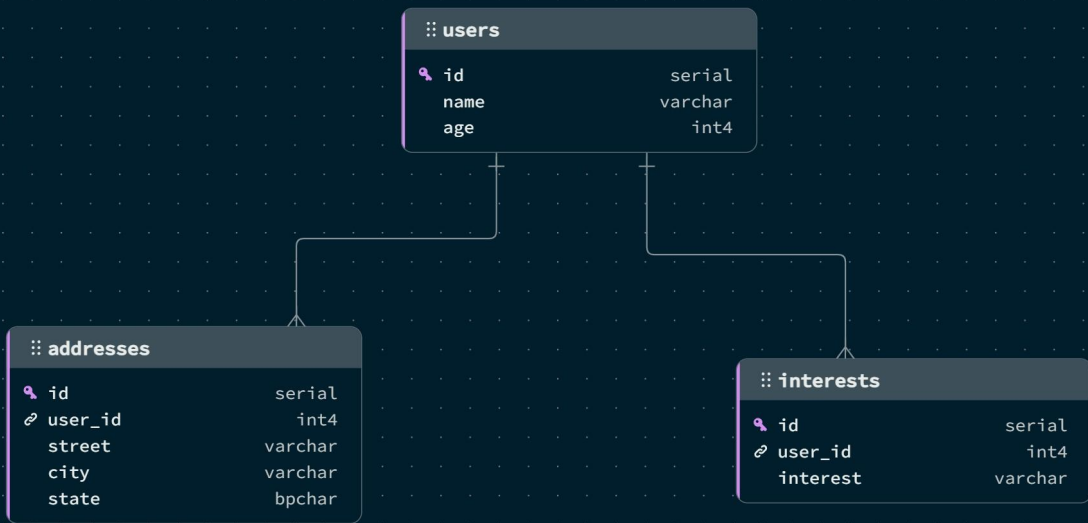


Referencing	Embedding
lado “n” muito grande	Integridade das operações
Uma parte usada com frequência e a outra não	Dados excluídos ou arquivados juntos
Restrição de memória	dados que são acessados juntos devem ser armazenados juntos

# Data Modeling



- *Relacional*



# Data Modeling



- *MongoDB*

:: users	
🔍 _id	object_id
id	integer
name	string
age	integer
interests	[]
id	integer
userId	integer
interest	string
addresses	[]
id	integer
userId	integer
street	string
city	string
state	string



# Basic Operations

- insertOne
- insertMany
- updateOne
- updateMany
- deleteOne
- deleteMany
- replaceOne
- findOne
- find
- count



# Basic Operations

- insertOne

```
db.getCollection('movies').insertOne(  
  {  
    "name": "novo filme",  
    "year": 2024  
  }  
);
```



# Basic Operations

- insertMany

```
db.getCollection('movies').insertMany(  
  [  
    {  
      "name": "novo filme 2026",  
      "year": 2026  
    },  
    {  
      "name": "novo filme 2027",  
      "year": 2027  
    }  
  ]  
);
```



# Basic Operations

- updateOne

```
db.getCollection('movies').updateOne(  
  { "year": 2027 },  
  {  
    $set: { year: 2028 }  
  }  
)
```





# Basic Operations

- updateMany

```
db.getCollection('movies').updateMany(  
  { "year": 2010 },  
  {  
    $set: { active: false }  
  }  
)
```



# Basic Operations

- deleteOne

```
db.getCollection('movies').deleteOne({ "year": 2029 })
```



# Basic Operations

- deleteMany

```
db.getCollection('movies').deleteMany({"year": 2029})
```



# Basic Operations

- replaceOne

```
db.getCollection('movies').replaceOne(  
  { "year" : 2029 },  
  { "year" : 2029, "title" : "novo titulo"},  
  { upsert: true }  
);
```



# Basic Operations

- findOne

```
db.getCollection('movies').findOne()
```



# Basic Operations

- find

```
db.getCollection('movies').find({"year": 2029})
```



# Basic Operations

- count

```
db.getCollection('movies').count()
```



# Aggregation Operations

Operações de agregação processam vários documentos e retornam resultados computados. Você pode usar operações de agregação para:

- Agrupar valores de vários documentos juntos.
- Realizar operações nos dados agrupados para retornar um único resultado.
- Analisar mudanças nos dados ao longo do tempo.





# Aggregation Operations

```
db.orders.aggregate( [  
  // Stage 1: Filter pizza order documents by pizza size  
  {  
    $match: { size: "medium" }  
  },  
  
  // Stage 2: Group remaining documents by pizza name and calculate total quantity  
  {  
    $group: { _id: "$name", totalQuantity: { $sum: "$quantity" } }  
  }  
  
] )
```



# Aggregation Operations

- \$match
- \$group
- \$project
- \$sort
- \$limit
- \$skip
- \$sum
- \$avg
- \$min
- \$max
- \$gt
- \$lt
- \$lookup



# Basic Operations

- \$match

```
db.users.aggregate(  
  [  
    {  
      $match: { age: { $gt: 25 } }  
    }  
  ]  
);
```



# Basic Operations

- \$group

```
db.sales.aggregate([
  {
    $group: {
      _id: "$item",
      totalQuantity: { $sum: "$quantity" }
    }
  }
])
```



# Basic Operations

- \$project

```
db.sales.aggregate([
  {
    $project: {
      _id: 0,
      item: 1,
      quantity: 1,
      price: 0
    }
  }
]);
```



# Basic Operations

- \$sort

```
db.sales.aggregate([  
  {  
    $sort: { price: -1 } // Ordena pelo campo price em ordem decrescente  
  }  
]);
```



# Basic Operations

- \$limit

```
db.sales.aggregate([  
  { $limit: 2 }  
]);
```



# Basic Operations

- \$skip

```
db.sales.aggregate([ // Pula 2 registros
  { $skip: 2 }
]);
```





# Basic Operations

- \$sum

```
db.sales.aggregate([
  {
    $group: {
      _id: "$item",
      totalQuantity: { $sum: "$quantity" }
    }
  }
]);
```



# Basic Operations

- \$avg

```
db.sales.aggregate([
  {
    $group: {
      _id: "$item", // Agrupa por item
      averageQuantity: { $avg: "$quantity" } // Calcula a média dos valores de quantity
    }
  }
]);
```



# Basic Operations

- \$min

```
db.sales.aggregate([
  {
    $group: {
      _id: "$item", // Agrupa por item
      minQuantity: { $min: "$quantity" } // Encontra o valor mínimo de quantity
    }
  }
]);
```



# Basic Operations

- \$max

```
db.sales.aggregate([
  {
    $group: {
      _id: "$item", // Agrupa por item
      minQuantity: { $max: "$quantity" } // Encontra o valor máximo de quantity
    }
  }
]);
```



# Basic Operations

- \$lt

```
db.users.aggregate(  
  [  
    {  
      $match: { age: { $lt: 25 } }  
    }  
  ]  
);
```



# Basic Operations

- \$gt

```
db.users.aggregate(  
  [  
    {  
      $match: { age: { $gt: 25 } }  
    }  
  ]  
);
```



# Basic Operations

- \$lookup

```
db.customers.insertMany(  
  [  
    { "customer_id": 101, "name": "Alice", "city": "New York" },  
    { "customer_id": 102, "name": "Bob", "city": "Los Angeles" }  
  ]  
)
```

```
db.orders.insertMany(  
  [  
    { "_id": 1, "order_id": "A001", "customer_id": 101, "amount": 500 },  
    { "_id": 2, "order_id": "A002", "customer_id": 102, "amount": 200 },  
    { "_id": 3, "order_id": "A003", "customer_id": 101, "amount": 700 }  
  ]  
)
```



# Basic Operations

- \$lookup

```
db.orders.aggregate([
  {
    $lookup: {
      from: "customers", // Coleção para realizar a junção
      localField: "customer_id", // Campo na coleção orders
      foreignField: "customer_id", // Campo na coleção customers
      as: "customerResult" // Nome do campo de saída para os documentos combinados
    }
  }
])
```





# Components and Tools

- Compass - The GUI for MongoDB
- Atlas - Cloud database

# Components and Tools



MongoDB Compass - ktor-api/sample\_mflix.movies

Connect Edit View Collection Help

ktor-api ...

My Queries Performance Databases

Search

admin local my\_database sample\_airbnb sample\_analytics sample\_geospatial sample\_guides sample\_mflix comments embedded\_movies **movies** sessions theaters users sample\_restaurants sample\_supplies sample\_training

My Queries movies

Documents 21.3K Aggregations Schema Indexes 2 Validation

Your pipeline is currently empty. Need help getting started? [Generate aggregation](#)

Explain Export Run Options

Untitled SAVE CREATE NEW EXPORT TO LANGUAGE PREVIEW STAGES TEXT WIZARD

21349 Documents in the collection

Preview of documents

<pre>{   "_id": ObjectId('573a1390f29313caabed42e8'),   "plot": "A group of bandits stage a brazen train hold-up, only to find a determ...",   "genres": Array (2)   runtime: 11   cast: Array (4)   poster: "https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtY...   title: "The Great Train Robbery" }</pre>	<pre>{   "_id": ObjectId('573a1390f29313caabed446f'),   "plot": "A greedy tycoon decides, on a whim, to corner the world market in whea...",   "genres": Array (2)   runtime: 14   cast: Array (4)   num_mflix_comments: 1   title: "A Corner in Wheat" }</pre>	<pre>{   "_id": ObjectId('573a1390f29313caabed4803'),   "plot": "Cartoon figures announce, via comi strip balloons, that they will mov...",   "genres": Array (3)   runtime: 7   cast: Array (1)   num_mflix_comments: 0   poster: "https://m.media-amazon.com/images/M/MV5BMTU3NjE5NzYtY...   title: "The Great Train Robbery" }</pre>
---	---	---

+ Add Stage

[Learn more about aggregation pipeline stages](#)

>\_MONGOSH

# Components and Tools



The screenshot displays the MongoDB Atlas Project Overview page for a project named 'RICAS'. The interface is dark-themed and includes a top navigation bar with the Atlas logo, project name, and various management links. A left sidebar provides navigation for Overview, Deployment, Services, Security, and New On Atlas. The main content area is titled 'Overview' and features several key sections:

- Clusters:** Shows 'Cluster0' with a 'Data Size: 478.45 MB'. It includes buttons for 'Connect', 'Edit configuration', 'Browse collections', 'View monitoring', and 'Add Tag'. A 'Create cluster' button is also present.
- Toolbar:** Displays 'Resources (6)' and 'Tips (4)', along with metrics for Performance (1), Cost (1), and Resilience (2).
- Application Development:** Features a 'Get connection string' button and a dropdown menu currently set to 'Java'. Below this, a message states: 'Your connected applications will appear here along with optimization best practices and features for your drivers. Don't see your apps? [Get updated connection string](#)'.

At the bottom, the system status is 'All Good', and the footer contains copyright information for MongoDB, Inc. (©2024) and links to Status, Terms, Privacy, Atlas Blog, and Contact Sales.



# Agenda

- Hands-On
  - insert
  - update
  - delete
- Scripts VS Code
- Atlas
  - Metrics
  - Charts
- Indexes