Centro Paula Souza

Fatec Jacareí
Prof. Francisco de Moura

BANCO DE DADOS NÃO RELACIONAL

Modelagem de Dados no MongoDB e Relacionamentos

Professora: Lucineide Pimenta

Tópicos da aula

Fatec
Jacareí
Prof. Francisco de Moura

- **Objetivo**
  - Compreender como modelar dados no MongoDB, comparando com o PostgreSQL, e explorar estratégias para representar relacionamentos entre documentos.
- **Tópicos**
  - Introdução à Modelagem de Dados no MongoDB
  - Estruturando Documentos no MongoDB
- **Exercícios**

Introdução ao MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Conteúdo:**
- **O que é o MongoDB?**
  - O MongoDB é um banco de dados orientado a documentos, onde cada registro é armazenado em formato JSON (ou BSON, que é uma versão binária de JSON).
- **Diferença entre bancos relacionais (SQL) e não relacionais (NoSQL)**
  - **Relacionais:** usam tabelas com linhas e colunas (Ex.: MySQL, PostgreSQL).
  - **Não relacionais:** armazenam dados de formas diferentes, como documentos (JSON), grafos ou colunas largas.

Sobre Compass x Mongosh

Fatec
Jacareí
Prof. Francisco de Moura

MongoDB Compass

Interface gráfica (GUI)

- Visual
- Ideal para iniciantes
- Mostra bancos, coleções e documentos
- Ajuda a entender estrutura

Mongosh

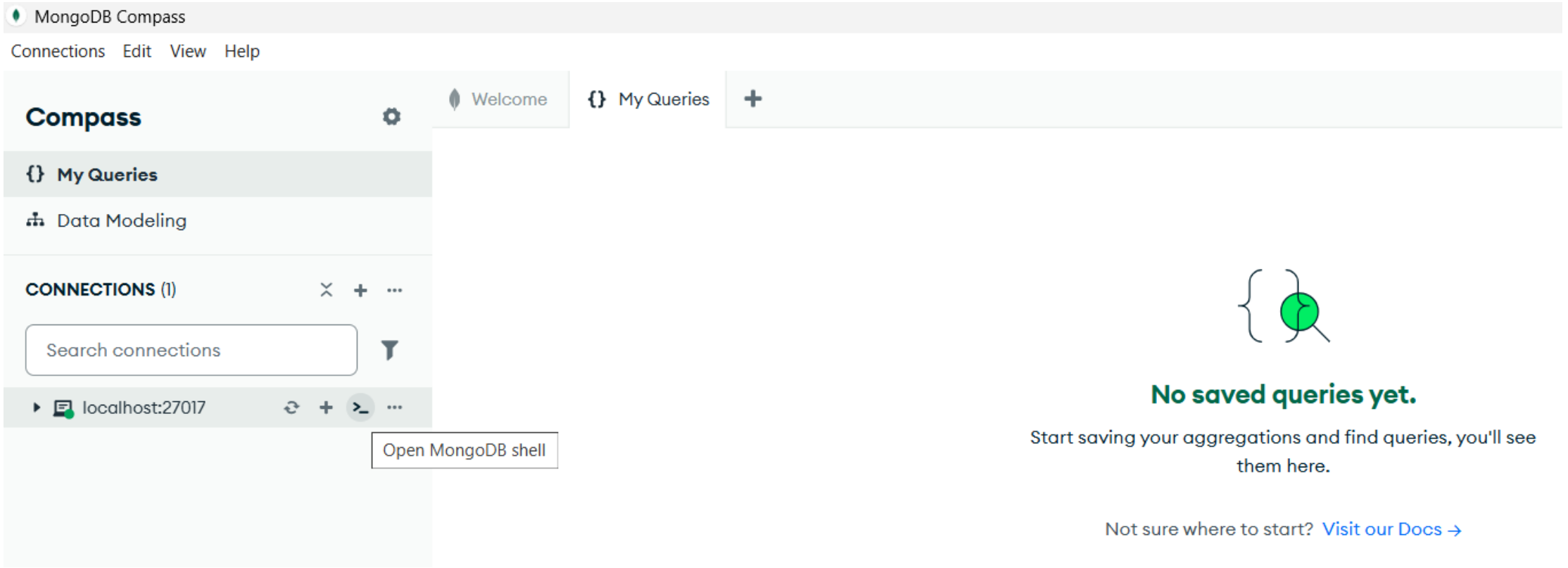
Terminal (linha de comando)

- Mais técnico
- Usado em produção
- Ideal para automação
- Desenvolve raciocínio lógico

Sobre Compass x Mongosh

Fatec
Jacareí
Prof. Francisco de Moura

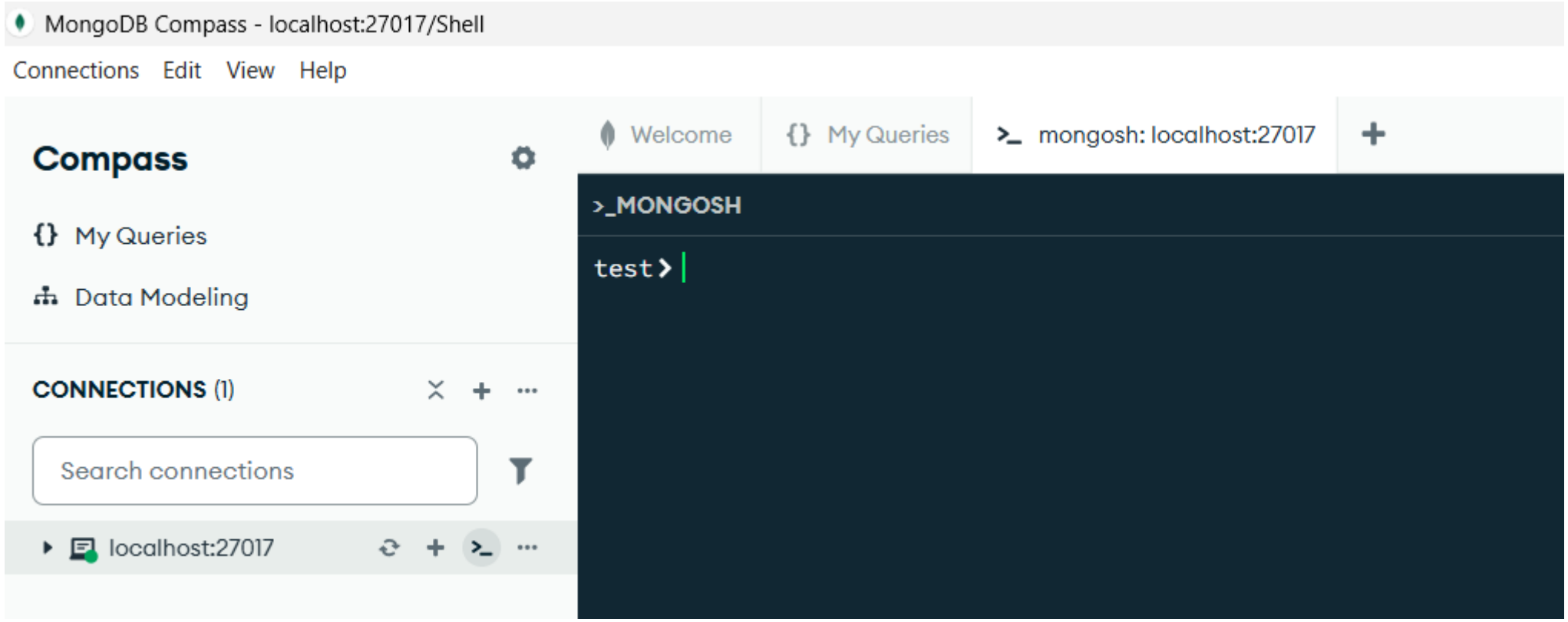
- Quando abrimos o MongoDB Compass podemos trabalhar no modo de interface (Compass) ou no modo terminal (Mongosh).



Sobre Compass x Mongosh

Fatec
Jacareí
Prof. Francisco de Moura

- Quando abrimos o MongoDB Compass e usamos o mongosh, ele abre no banco **test**.



Sobre Compass x Mongosh

Fatec
Jacareí
Prof. Francisco de Moura

- **Por que o MongoDB abre no banco test?**
  - Quando você abre o **mongosh**, o MongoDB:
    - Conecta ao servidor
    - Seleciona automaticamente o banco chamado **test**
    - Se nenhum banco for especificado, ele usa test como padrão
  - Isso acontece porque o MongoDB **precisa estar "apontando" para algum banco**, mesmo que ele ainda não exista fisicamente.

Sobre Compass x Mongosh

Fatec
Jacareí
Prof. Francisco de Moura

- Diferença entre o PostgreSQL e o MongoDB:
  - No **PostgreSQL:** o banco precisa ser criado formalmente antes de usar.
  - No **MongoDB:** o banco só é realmente criado quando você insere um documento nele.

PostgreSQL	MongoDB
Banco precisa ser criado com CREATE DATABASE	Banco só existe após inserir dados
Tabelas	Coleções
Linhas	Documentos
Colunas fixas	Estrutura flexível
SQL	Comandos JSON-like

Introdução ao Banco de Dados Não Relacional e MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Principais Componentes:**
  - **Coleções:** Equivalentes às tabelas em bancos relacionais.
  - **Documentos:** Registros individuais que armazenam dados.

- **Comandos Básicos:**

  show dbs // *Listar bancos de dados.*

  Atenção! *Somente é possível listar/visualizar o BD após a criação da primeira coleção.*

Introdução ao Banco de Dados Não Relacional e MongoDB

- **Principais Componentes:**
  - **Coleções:** Equivalentes às tabelas em bancos relacionais.
  - **Documentos:** Registros individuais que armazenam dados.

- **Comandos Básicos:**

```
use lojaGames
show dbs // Ainda não aparece.
db.clientes.insertOne({ nome: "Ana" })
show dbs // Agora aparece.
```

Aí sim o banco passa a existir fisicamente.

Introdução ao Banco de Dados Não Relacional e MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Pergunta:**
  - *No postgresql podemos criar uma tabela vazia, no mongodb podemos criar uma coleção vazia também?*

Introdução ao Banco de Dados Não Relacional e MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- Sim, é possível criar uma coleção vazia no MongoDB.
  - **Criando coleção vazia:**

```
use lojaGames
db.createCollection("clientes")
show collections // Vai aparecer clientes, mesmo sem documentos.
```

  - **Criando uma coleção com dados:**

```
use lojaGames
db.clientes.insertOne({ nome: "Lucas" }) // Se clientes não existir, o MongoDB cria automaticamente.
```

    Essa é a forma mais usada.

Introdução ao Banco de Dados Não Relacional e MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Definição da estrutura:**
  - No **PostgreSQL** você precisa definir:

```
CREATE TABLE clientes (
   id SERIAL PRIMARY KEY,
   nome VARCHAR(100),
   email VARCHAR(100)
);
```

  - No **MongoDB**:

```
db.createCollection("clientes")
```

    - Não define estrutura.
    - A estrutura só “nasce” quando os documentos são inseridos.

Introdução ao Banco de Dados Não Relacional e MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Pergunta:**
  - *Então o MongoDB não tem controle de estrutura?*

Introdução ao Banco de Dados Não Relacional e MongoDB

- Tem sim!
  Pode usar **validação de schema (JSON Schema Validation)** ao criar a coleção.

```
db.createCollection("clientes", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["nome", "email"],
      properties: {
        nome: { bsonType: "string" },
        email: { bsonType: "string" }
      }
    }
  }
})
```

Isso aproxima o MongoDB de um modelo mais estruturado, como no PostgreSQL.

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Conclusão:**
  - No **PostgreSQL**, primeiro eu crio a tabela e depois insiro os dados.
  - No **MongoDB**, eu posso criar a coleção antes, mas normalmente ela nasce quando o primeiro documento é inserido.

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Pergunta:**
  - *Se eu não defino tipo nem tamanho... como o MongoDB sabe qual é o tipo do campo?*

Comparação Prática: PostgreSQL vs MongoDB

- Ele descobre o **tipo** automaticamente com base no valor inserido.
- **Exemplo:**

```
db.clientes.insertOne({
   nome: "Ana",
   idade: 25,
   ativo: true
})
```

O MongoDB entende:

- **"Ana"**: String
- **25**: Number
- **True**: Boolean

Ele identifica o tipo pelo valor.

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- E o tamanho? (**tipo VARCHAR(100)**)
- Aqui está a grande diferença:
  - No **MongoDB** NÃO existe tamanho fixo como VARCHAR(100).
  - No **PostgreSQL** você precisa definir:
    - **nome VARCHAR(100)**
  - No MongoDB:

    { nome: "Ana" }
- Ele **não** impõe limite de tamanho para *string*.
- A única limitação real é:

Um documento inteiro pode ter no máximo 16MB.

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Pergunta:**
  - *Então o MongoDB não tem estrutura?*

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- Tem sim... mas é flexível.
- No **MongoDB**:
  - A estrutura nasce a partir dos dados inseridos.
  - Cada documento pode ter campos diferentes.
  - Não existe rigidez estrutural obrigatória.
- **Exemplo**:

  { nome: "Lucas", idade: 25 }

  { nome: "Ana" }

  { nome: "Carlos", email: "carlos@email.com" }

Todos podem existir na mesma coleção.
No PostgreSQL isso não seria permitido.

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Pergunta:**
  - *Então é "bagunça"?*

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- O **MongoDB** é flexível, mas o desenvolvedor é responsável pela organização.
- Em sistemas reais usamos:
  - Validação de Schema
  - Boas práticas de modelagem
  - Padrão consistente de documentos

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- **Pergunta:**
  - *Se eu quiser definir tipos como no PostgreSQL?*

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- Você pode usar **validação de schema**:

```
db.createCollection("clientes", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["nome", "idade"],
      properties: {
        nome: {
          bsonType: "string"
        },
        idade: {
          bsonType: "int"
        }
      }
    }
  }
})
```

Agora:

- **nome** precisa ser *string*
- **idade** precisa ser número *inteiro*
- Campos obrigatórios são verificados

Aqui você aproxima MongoDB do modelo relacional.

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

PostgreSQL	MongoDB
Estrutura definida antes	Estrutura definida pelos dados
Tipo obrigatório	Tipo inferido automaticamente
Tamanho definido (VARCHAR)	Sem tamanho fixo
Estrutura rígida	Estrutura flexível
Schema obrigatório	Schema opcional

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

- No **PostgreSQL** eu defino a estrutura **antes** de inserir os dados.
- No **MongoDB** a estrutura **nasce** com os dados.
- Se eu quiser controlar, eu posso adicionar **validação**.

Comparação Prática: PostgreSQL vs MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

1-Criação do banco de dados:

- **PostgreSQL:**
  Criação do banco de dados:

```
CREATE DATABASE clima
```

- **Criação da tabela:**

```
CREATE TABLE dados_meteorologicos (
        id SERIAL PRIMARY KEY,
        cidade VARCHAR (50),
        temperatura DECIMAL,
        umidade DECIMAL, data DATE
);
```

- **MongoDB:**
  - **Use clima**

Comparando Operações com PostgreSQL e MongoDB

Fatec Jacareí
Prof. Francisco de Moura

2-Inserção de dados:

- **PostgreSQL:**

```
INSERT INTO dados_meteorológicos (cidade, temperatura, umidade, data) VALUES ('São Paulo', 25, 70, '2025-02-10');
INSERT INTO dados_meteorológicos (cidade, temperatura, umidade, data) VALUES ('Rio de Janeiro', 28.3, 65, '2025-02-11');
INSERT INTO dados_meteorológicos (cidade, temperatura, umidade, data) VALUES ('Belo Horizonte', 30, 88, '2025-02-12');
INSERT INTO dados_meteorológicos (cidade, temperatura, umidade, data) VALUES ('Vitória', 19, 80, '2025-02-12').
```

- **MongoDB:**

```
db.dados_meteorologicos.insertOne({ cidade: "São Paulo", temperatura: 25, umidade: 70, data: "2025-02-10" })
db.dados_meteorologicos.insertOne({ cidade: "Rio de Janeiro", temperatura: 28.3, umidade: 65, data: "2025-02-11" })
db.dados_meteorologicos.insertOne({ cidade: "Belo Horizonte", temperatura: 30, umidade: 88, data: "2025-02-12" })
db.dados_meteorologicos.insertOne({ cidade: "Vitória", temperatura: 19, umidade: 80, data: "2025-02-12" })
```

Primeiros Comandos com MongoDB

Fatec Jacareí
Prof. Francisco de Moura

3-Listagem de dados:

- **PostgreSQL:**

```
SELECT * FROM dados_meteorológicos;
```

id	cidade	temperatura	umidade	data
1	Lisboa	30.5	70	2025-02-10

- **MongoDB:**

```
db.dados_meteorológicos .find()
```

- **Resultado:**

Primeiros Comandos com MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

3-Listagem de dados:

- **PostgreSQL:**

```
SELECT * FROM dados_meteorológicos WHERE cidade = 'São Paulo';
```

- **MongoDB:**

```
db.dados_meteorológicos.find({cidade: "São Paulo"})
```

- **Resultado:**

```
{ "_id": ObjectId("xxxx"), "cidade": " São Paulo", "temperatura": 25, "umidade": 70, "data": "2025-02-10" }
```

Primeiros Comandos com MongoDB

Fatec
Jacareí
Prof. Francisco de Moura

4-Atualização de Dados:

Alterar a temperatura de São Paulo

- **PostgreSQL:**

```
UPDATE dados_meteorológicos SET temperatura=13 WHERE cidade='São Paulo';
```

- **MongoDB:**

```
db.dados_meteorológicos.updateOne({ cidade: " São Paulo"}, { $set: { temperatura: 13 } })
```

- Resultado:

```
{ "_id": ObjectId("xxxx"), "cidade": " São Paulo", "temperatura": 13, "umidade": 70, "data": "2025-02-10" }
```

Deletando Dados

Fatec
Jacareí
Prof. Francisco de Moura

5-Exclusão de dados:

- **PostgreSQL:**

```
DELETE FROM dados_meteorológicos WHERE cidade='São Paulo';
```

- **MongoDB:**

```
db.dados_meteorológicos.deleteOne({ cidade: " São Paulo"})
```

Operações CRUD com MongoDB e PostgreSQL Lado a Lado

Fatec
Jacareí
Prof. Francisco de Moura

Operação	PostgreSQL	MongoDB
Inserir	`INSERT INTO clima (...) VALUES (...)`	`db.clima.insertOne({ ... })`
Consultar	`SELECT * FROM clima WHERE ...`	`db.clima.find({ ... })`
Atualizar	`UPDATE clima SET ... WHERE ...`	`db.clima.updateOne({ ... }, { $set: { ... } })`
Excluir	`DELETE FROM clima WHERE ...`	`db.clima.deleteOne({ ... })`

Referências Bibliográficas

Fatec
Jacareí
Prof. Francisco de Moura

- **Material de apoio:**

- Chodorow, Kristina. *MongoDB: The Definitive Guide*. O'Reilly Media, 2013.
- *PostgreSQL Documentation*. Disponível em: https://www.postgresql.org/docs/
- *MongoDB Documentation*. Disponível em: https://www.mongodb.com/docs/
- Cattell, Rick. *Scalable SQL and NoSQL Data Stores*. ACM, 2011.

Bibliografia Básica

Fatec
Jacareí
Prof. Francisco de Moura

- BOAGLIO, Fernando**. MongoDB:** Construa novas aplicações com novas tecnologias. São Paulo: Casa do Código, 2015.
- ELMASRI, R.; NAVATHE, S. B. **Sistemas de Banco de Dados:** Fundamentos e Aplicações. 7ed. São Paulo: Pearson, 2019.
- SADALAGE, P.; FOWLER, M. **Nosql Essencial:** Um Guia Conciso Para o Mundo Emergente da Persistência Poliglota. São Paulo: Novatec, 2013.
- SINGH, Harry. **Data Warehouse**: conceitos, tecnologias, implementação e gerenciamento. São Paulo: Makron Books, 2001.

Bibliografia Complementar

Fatec
Jacareí
Prof. Francisco de Moura

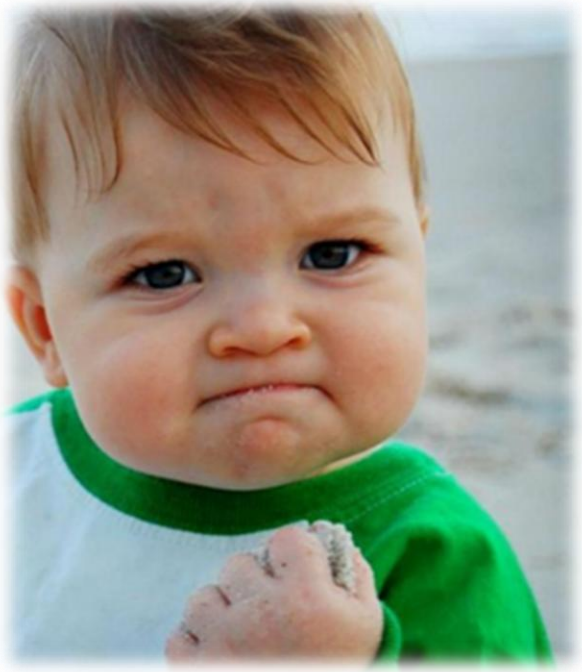
- FAROULT, Stephane. **Refatorando Aplicativos SQL**. Rio de Janeiro: Alta Books, 2009.
- PANIZ, D. **NoSQL:** Como armazenar os dados de uma aplicação moderna. Casa do Código, 2016.
- SOUZA, M. **Desvendando o Mongodb**. Rio de Janeiro: Ciência Moderna, 2015.

Dúvidas?

Fatec
Jacareí
Prof. Francisco de Moura

Considerações Finais

Fatec
Jacareí
Prof. Francisco de Moura



Professora:

Lucineide Pimenta

Bom semestre à todos!