BANCO DE DADOS 2

MONGODB

Professor: Nícollas Cretton

E-mail: nicollas.cretton@professor.faminas.edu.br

CONSULTANDO BANCOS E COLLECTIONS

- Para consultar quais bancos existem utilize o seguinte comando:
  - **show** dbs;
  - **show** databases;

- Para consultar quais coleções existem utilize o seguinte comando:
  - **show** collections;

CRIANDO UM BANCO OU TROCANDO

- Para criar um banco, utilize o seguinte comando:
  - **use** NomeDoBD;
    - Um banco só se torna permanente se existir uma (ou mais) coleção dentro dele.

- O mesmo comando pode ser utilizado para trocar de banco.

CRIANDO UMA COLLECTION

- Para criar uma coleção, utilize o seguinte comando:
  - **db.createCollection(**"NomeDaCo"**)**;
    - Não são passados os campos (colunas), pois cada documento possui a flexibilidade de ter os seus.

- **Não é a melhor forma de se criar uma coleção.** O comum é criarmos a coleção junto com a criação de um documento (lançamento dos dados).

CRIANDO UMA COLLECTION JUNTO COM UM DOCUMENTO

- Para criar uma coleção (junto com um de seus documentos), utilize o seguinte comando:
  - **db.**NomeDaCo**.insertOne({**"nomeCampo": "dado"**})**;
    - Pense em um documento como um registro no banco de dados relacional.
    - É possível colocar o nome do campo sem aspas duplas, porém não é o recomendado por não seguir as regras do JSON.
- É possível ainda criar uma coleção junto com um index.

CAPPED COLLECTION

- **Capped collections** são coleções de **tamanho fixo** que funcionam como se tivesse uma memória circular.
  - Quando atinge o limite, os **documentos mais antigos são automaticamente removidos** para dar espaço aos novos.

- Para criar uma capped collection, utilize o seguinte comando:
  - **db.createCollection(**"NomeDaCo", **{ capped:** true, **size:** 10240, **max:** 100 **})**;
    - **capped:** true → Define a coleção como capped.
    - **size:** 10240 → Define o tamanho máximo (10 KB).
    - **max:** 100 → (Opcional) Limita o máximo de 100 documentos.

CRIANDO UM DOCUMENTO

- Para criar um documento podemos utilizar os comandos:
  - **insertOne()**
  - **insertMany()**

- Cada documento no MongoDB é armazenado em uma coleção e garante:
  - **Garante que ele tenha um _id único** (se não for especificado, um será gerado automaticamente).
  - **Mantém a estrutura flexível**, ou seja, documentos na mesma coleção podem ter campos diferentes

CRIANDO UM DOCUMENTO – INSERTONE()

- Insere **apenas um documento** na coleção.
- Estrutura:
  - **db.**NomeDaCo**.insertOne({**"nomeCampo": "dado"**})**;
- **Exemplo:**
  - **db.**NomeDoCo**.insertOne({**
    - "nome" : "Teclado Mecânico",
    - "preco" : 250.00,
    - "estoque" : 30,
    - "categoria" : "Periféricos"
  - **})**;

CRIANDO UM DOCUMENTO – INSERTMANY()

- Insere **vários documentos de uma vez** na coleção.
- Estrutura:
  - db.NomeDaCo.insertMany([{"nomeCampo": "dado"}, {nomeCampo": "dado"}]);
- **Exemplo:**
  - db.NomeDoCo.insertMany([
    - { nome: "Mouse Gamer", preco: 180.00, estoque: 20, categoria: "Periféricos" },
    - { nome: "Monitor 27"", preco: 1200.00, estoque: 10, categoria: "Monitores" },
    - { nome: "Headset", preco: 350.00, estoque: 15, categoria: "Áudio" }
  -]);

RELACIONAMENTO ENTRE COLEÇÕES

- No MongoDB os relacionamentos entre coleções não funcionam da mesma forma.

- Existem duas formas de representar relações entre documentos:
  - **Embedded Documents (Documentos Incorporados)**
  - **References (Referências entre documentos)**

- A escolha entre embedded e referências depende do contexto.

RELACIONAMENTO ENTRE COLEÇÕES

Embedded Documents (Documentos Incorporados)

- Os dados relacionados ficam dentro do próprio documento principal.
- Documentos incorporados são guardados em um campo ou array.
- Query mais rápidas, uma vez que pega toda informação diretamente do mesmo lugar.

- **Usados quando:**
  - Documentos incorporados são exclusivos do documento pai
  - Documentos incorporados não mudam com frequencia
  - Documentos incorporados não precisam ser atualizados em massa
  - A quantidade e tamanho dos documentos incorporados são pequenos
  - Relacionamentos 1:1 e 1:N, se fizer sentido.

RELACIONAMENTO ENTRE COLEÇÕES

Embedded Documents (Documentos Incorporados)

- **Exemplo 1:1:**

```
db.pessoa.insertOne({
   "nome": "João Silva",
   "email": "joao@email.com",
   "data_nascimento": ISODate("1998-04-15"),
   "endereco": {
      "rua": "Rua A",
      "numero": 123,
      "cidade": "São Paulo",
      "estado": "SP",
      "cep": "01000-000"
   }
})
```

RELACIONAMENTO ENTRE COLEÇÕES

Embedded Documents (Documentos Incorporados)

- **Exemplo 1:N:**
  - db.produto.insertOne({
  - "cliente": "Carlos Souza",
  - "data_pedido": "2024-02-23",
  - "itens": [
  - { "produto": "Mouse Gamer", "quantidade": 2, "preco": 180.00 },
  - { "produto": "Teclado Mecânico", "quantidade": 1, "preco": 250.00 }
  -]})

RELACIONAMENTO ENTRE COLEÇÕES

➢ References (Referências entre documentos)

- Criação de um campo ou coleção para representar o relacionamento entre coleções.
  - Não é exatamente igual ao SQL, não é criado um vínculo entre as coleções.
- O Campo de relacionamento pode guardar um único valor ou um array.
- Menos redundância que o Embedded.
- Atualização mais fácil.
- Query mais lenta, comparado com o Embedded.

➢ **Usados quando:**

- Acredita que a quantidade de dados irá crescer
- Atualização frequente dos dados
- Pesquisas mais complexas
- Dados podem ser compartilhados
- Relacionamentos 1:1, 1:N e N:N

RELACIONAMENTO ENTRE COLEÇÕES

➢References (Referências entre documentos) 1:1

- db.pessoa.insertOne({
- "nome": "João Silva",
- "email": "joao@email.com",
- "data_nascimento": ISODate("1998-04-15"),
- "endereco_id": ObjectId("98b3e5b2f4a9d4c8a1e4a1a2")
- })

- db. endereco.insertOne({
- "_id": ObjectId("98b3e5b2f4a9d4c8a1e4a1a2"),
- "rua": "Rua A",
- "numero": 123,
- "cidade": "São Paulo",
- "estado": "SP",
- "cep": "01000-000"
- })

RELACIONAMENTO ENTRE COLEÇÕES

➢ References (Referências entre documentos) 1:N

- db.autor.insertOne({
- "_id": ObjectId("65b3e5b2f4a9d4c8a1e4a1a1"),
- "nome": "J.K. Rowling",
- "nacionalidade": "Britânica"
- })

- db.livro.insertOne({
- "titulo": "Harry Potter e a Pedra Filosofal",
- "ano": 1997,
- "autor_id": ObjectId("65b3e5b2f4a9d4c8a1e4a1a1")
- })

RELACIONAMENTO ENTRE COLEÇÕES

➢References (Referências entre documentos) N:N

```
db.aluno.insertOne({
   "_id": ObjectId("65b3e5b2f4a9d4c8a1e4a1a1"),
   "nome": "Ana Souza",
   "cursos": [ObjectId("98b3e5b2f4a9d4c8a1e4a1a2"), ObjectId("98b3e5b2f4a9d4c8a1e4a1a3")]
})
```

```
db.curso.insertMany([
   {"_id": ObjectId("98b3e5b2f4a9d4c8a1e4a1a2"),
   "nome": "Banco de Dados NoSQL"},
   {"_id": ObjectId("98b3e5b2f4a9d4c8a1e4a1a3"),
   "nome": "Desenvolvimento Web"}
])
```

RELACIONAMENTO ENTRE COLEÇÕES

➢References (Referências entre documentos) N:N

- db.aluno.insertOne({
- "_id": ObjectId("65b3e5b2f4a9d4c8a1e4a1a1"),
- "nome": "Ana Souza" })

- db.curso.insertMany([
- {"_id": ObjectId("98b3e5b2f4a9d4c8a1e4a1a2"),
- "nome": "Banco de Dados NoSQL"},
- {"_id": ObjectId("98b3e5b2f4a9d4c8a1e4a1a3"),
- "nome": "Desenvolvimento Web"}])

- db.matricula.insertOne({
- "aluno_id": ObjectId("65b3e5b2f4a9d4c8a1e4a1a1"),
- "curso_id": ObjectId("98b3e5b2f4a9d4c8a1e4a1a2") })

TIPOS DE DADOS: DATAS

- No MongoDB, as datas são armazenadas como objetos do tipo **ISODate**.
  - Segue a estrutura obrigatória de YYYY-MM-DD
  - Precisa estar entre aspas duplas

- Para inserir uma data utilize:
  - ... Data: **ISODate(**"2024-02-24"**)**

PROJETO DE ACOMPANHAMENTO

➢ **Crie um banco de dados que possua alunos, disciplinas e tipo de disciplina.**

- O aluno deve ter um id, primeiro_nome, sobrenome e data_nascimento.
- A disciplina deve ter id, nome e seu tipo de disciplina.
- O tipo de disciplina possui id e nome.
- Alunos podem fazer várias disciplinas e disciplinas podem ter vários alunos.
- Cada disciplina pode ter somente um tipo de disciplina, cada tipo de disciplina pode representar várias disciplinas.

PROJETO DE ACOMPANHAMENTO

- **Insira dados no Banco de Dados criado na última aula.**
  - **Os dados dos alunos são:**
  - **O aluno deve ter um id, primeiro_nome, sobrenome e data_nascimento.**
    - Leonardo Nascimento Souza Cruz, 30/05/2002
    - Guilherme Riguiti de Oliveira, 03/04/1997
    - Pedro Henrique Carlos de Souza Machado, 12/03/1999
    - Marcos Tarcisio Buettel Mota, 04/04/1991

PROJETO DE ACOMPANHAMENTO

- **Insira dados no Banco de Dados criado na última aula.**
  - **Os dados dos alunos são:**
  - **O aluno deve ter um id, primeiro_nome, sobrenome e data_nascimento.**
    - Lucas dos Santos Narciso, 31/07/2001
    - Luiz Octavio Barros Rodrigues, 12/12/1996
    - João Marcus da Silva Sousa, 17/09/1999
    - Flávia Aparecida Ferrari Toscano, 03/10/2001

PROJETO DE ACOMPANHAMENTO

- **Insira dados no Banco de Dados criado na última aula.**
  - **Os dados dos tipos de disciplina são:**
  - **O tipo de disciplina possui id e nome.**
    - Formação Básica
    - Práticas Profissionais
    - Formação Complementar
    - Projeto Aplicado à Comunidade
    - Optativa

PROJETO DE ACOMPANHAMENTO

- **Insira dados no Banco de Dados criado na última aula.**
  - **Os dados das disciplinas são:**
  - **A disciplina deve ter id, nome e seu tipo de disciplina.**
    - Laboratório de Banco de Dados, Práticas Profissionais
    - Optativa I, Optativa
    - Governança em Tecnologia da Informação, Formação Complementar
    - Linguagem de Programação I, Práticas Profissionais

PROJETO DE ACOMPANHAMENTO

- **Insira dados no Banco de Dados criado na última aula.**
  - **Os dados das disciplinas são:**
  - **A disciplina deve ter id, nome e seu tipo de disciplina.**
    - Algoritmos, Formação Básica
    - Engenharia de Software, Práticas Profissionais
    - Gestão de Projetos, Formação Complementar
    - Projeto Aplicado à Comunidade IV, Projeto Aplicado à Comunidade
    - Estruturas de Dados, Formação Básica

PROJETO DE ACOMPANHAMENTO

- **Insira dados no Banco de Dados criado na última aula.**
  - **As disciplinas de cada aluno são:**
  - **O aluno_disciplina deve ter um id_aluno e id_disciplina.**
    - **Leonardo Nascimento Souza Cruz** faz Laboratório de Banco de Dados; Governança em Tecnologia da Informação; Engenharia de Software; Projeto Aplicado à Comunidade IV.
    - **Guilherme Riguiti de Oliveira** faz Optativa I; Governança em Tecnologia da Informação; Gestão de Projetos; Projeto Aplicado à Comunidade IV.
    - **Pedro Henrique Carlos de Souza Machado** faz Optativa I; Linguagem de Programação I; Gestão de Projetos; Projeto Aplicado à Comunidade IV.

PROJETO DE ACOMPANHAMENTO

- **Insira dados no Banco de Dados criado na última aula.**
  - **As disciplinas de cada aluno são:**
  - **O aluno_disciplina deve ter um id_aluno e id_disciplina.**
    - **Marcos Tarcisio Buettel Mota** faz Laboratório de Banco de Dados; Optativa I; Governança em Tecnologia da Informação; Linguagem de Programação I; Algoritmos; Engenharia de Software; Gestão de Projetos; Projeto Aplicado à Comunidade IV; Estruturas de Dados.
    - **Lucas dos Santos Narciso** faz Laboratório de Banco de Dados; Optativa I; Engenharia de Software; Gestão de Projetos; Projeto Aplicado à Comunidade IV.

PROJETO DE ACOMPANHAMENTO

- **Insira dados no Banco de Dados criado na última aula.**
  - **As disciplinas de cada aluno são:**
  - **O aluno_disciplina deve ter um id_aluno e id_disciplina.**
    - **Luiz Octavio Barros Rodrigues** faz Governança em Tecnologia da Informação; Engenharia de Software; Projeto Aplicado à Comunidade IV.
    - **João Marcus da Silva Sousa** faz Governança em Tecnologia da Informação; Linguagem de Programação I; Projeto Aplicado à Comunidade IV.
    - **Flávia Aparecida Ferrari Toscano** faz Laboratório de Banco de Dados; Linguagem de Programação I; Algoritmos; Gestão de Projetos; Projeto Aplicado à Comunidade IV.

BANCO DE DADOS 2

MONGODB

Professor: Nícollas Cretton

E-mail: nicollas.cretton@professor.faminas.edu.br