Universidade Norte do Paraná – Polo Taquara RJ

Engenharia de Software

Geovane Dias de Araujo – 32256101

Portfólio – Relatório de Aula Prática

Disciplina: Banco de Dados Não Relacional

Rio de Janeiro/RJ
2024

Geovane Dias de Araujo – 32256101

Portfólio – Relatório de Aula Prática

Disciplina: Banco de Dados Não Relacional

Trabalho de portfólio apresentado como requisito parcial para a obtenção de pontos para a média semestral.

Orientador: Vinicius Camargo Prattes

Rio de Janeiro/RJ
2024

Sumário

1. Introdução

Este trabalho tem como objetivo demonstrar que o aluno que o realizou possui a capacidade de realizar tarefas envolvendo a criação de banco de dados não relacionais, estas tarefas envolvem a criação de um banco de dados, a manipulação de coleções, a utilização de métodos CRUD(*Create Read Update Delete*) e a capacidade de realizar pesquisas específicas desses dados.

2. Métodos

Como pré requisito estarei utilizando dois programas recomendados pelo Roteiro da Aula Prática o MongoDB Comunity Server e o MongoDB Compass que já estão devidamente instalados e configurados. Este trabalho não visa a configuração de ambos portanto será apenas descrito que: estarei utilizando a configuração de servidor local exatamente de acordo com a documentação oficial destes programas.

Nesta primeira etapa será executado a criação do banco de dados a ser utilizado através do método "*use*" já que no MongoDB os bancos de dados não são propriamente criados a não ser que recebam algum conteúdo. Portanto, utilizaremos o seguinte código:

```
use lojadb
```

Após verificar que estamos usando o bando de dados que desejamos criar vamos utilizar o método "*createCollection*()" para criar a coleção que usaremos neste projeto.

```
db.createCollection("vendas")
```

Dentro do shell utilizaremos o método "*insertMany*()" para a inserção inicial dos primeiros dados nesta coleção. Aqui é imprtante ressaltar que como estamos utilizando o shell mais atual, o mongosh, não utlizaremos a síntaxe desatualizada "*insert*" que atualmente não é recomendado segundo a documentação oficial do MongoDB. Asssim fica o código 1:

```
db.vendas.insertMany( [
    {
    nome: "João",
    vip: 1,
    email: "joão@email.com",
    telefone: [99991111, 88881111]
    },
    {
```

```
    nome: "Marcos",
    vip: 0,
    telefone: [99992222]
    },
    {
    nome: "Maria",
    vip: 1,
    email: "maria@email.com",
    telefone: [99993333, 88883333, 99883000]
    }
] )
```

Com os documentos criados, iremos inserir novos dados com o uso do método "*updateOne*()". Desta vez inserindo os endereços como uma objetos de dados dentro de cada documento já existente. Utilizaremos o código 2:

```
db.vendas.updateOne({ nome: "João" }, {
    $set: {
        endereço: {
            rua:"Um",
            numero:1000,
            complemento:"Apto 1 Bloco 1",
            cidade:"São Paulo",
            estado:"SP"
        }
    }
})
db.vendas.updateOne({ nome: "Marcos" }, {
    $set: {
        endereço: {
            rua:"Dois",
            numero:4000,
            cidade:"Campinas",
            estado:"SP"
        }
    }
})
db.vendas.updateOne({ nome: "Maria" }, {
    $set: {
        endereço: {
            rua:"Três",
            numero:3000,
            cidade:"Londrina",
            estado:"PR"
        }
    }
```

```
})
```

Nesta etapa iremos inserir os dados de vendas de cada pessoa através do método “*updateOne*()”. Além de inserir os dados de compra como objetos criaremos uma série de matrizes de dados através do uso correto de colchetes. Assim ficará o código 3:

```
db.vendas.updateOne({ nome: "João" }, {
    $set: {
        compras: [
            {
                produto:"notebook",
                valor:3000.00,
                quantia:1
            }
        ]
    }
})
db.vendas.updateOne({ nome: "Marcos" }, {
    $set: {
        compras: [
            {
            produto:"caderno",
            valor:20.00,
            quantia:1
            },
            {
            produto:"caneta",
            valor:3.00,
            quantia:5
            },
            {
            produto:"borracha",
            valor:2.00,
            quantia:2
            }
        ]
    }
})
db.vendas.updateOne({ nome: "Maria" }, {
    $set: {
        compras: [
            {
            produto:"tablet",
            valor:2500.00,
```

```
            quantia:1
            },
            {
            produto:"capa para tablet",
            valor:50.00,
            quantia:1
            }
        ]
    }
})
```

Segundo o que foi pedido no roteiro deste trabalho, realizaremos as seguintes consultas:

1. Uma consulta que retorne todos os documentos da coleção.
2. Uma consulta que localize as informações da cliente “Maria”.
3. Uma busca que retorna os clientes VIPs da loja (VIP = 1). Retorne apenas o campo “nome” de cada um.
4. Uma consulta que exiba as compras efetuadas por “Marcos”
5. Uma consulta que retorne todos os nomes de produtos comprados por todos os clientes.

Primeiramente, usaremos novamente o comando “use” para garantir que estamos usando o banco de dados correto. E então, para cada consulta listada, utilizaremos os seguintes comandos:

Consulta 1:

```
db.vendas.find().pretty()
```

Consulta 2:

```
db.vendas.find({nome:"Maria"})
```

Consulta 3:

```
db.vendas.find( {vip: 1 },
{ _id:0, vip:0, email:0, telefone:0, endereço:0,compras:0 })
```

Consulta 4

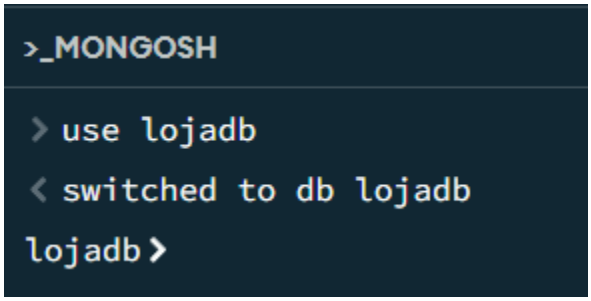
```
db.vendas.find({"nome": "Marcos",} ,
{_id:0, vip:0, email:0, telefone:0, endereço:0})
```

Consulta 5

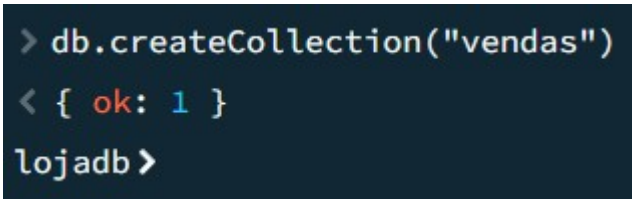
```
db.vendas.aggregate([
  { $unwind: "$compras" },
  { $group: { _id: null, valores: { $addToSet: "$compras.produto" } } }
 ]);
```

3. Resultados

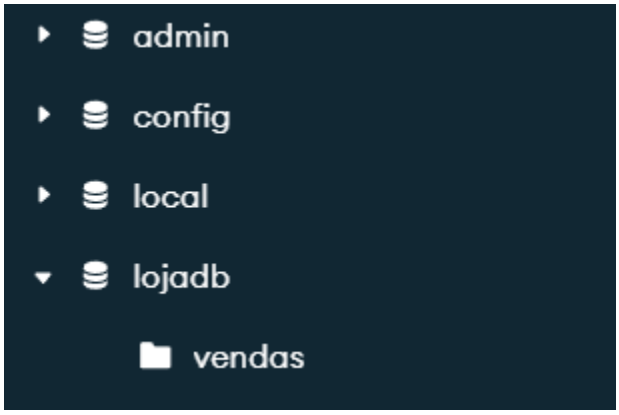
Utilizando o “use” para selecionar o banco de dados a ser criado:



Criando a coleção com o método “*createCollection*()”:



Agora já podemos ver o banco de dados criado através do Compass conforme mostra a foto a seguir, mas continuaremos utilizando o shell.



Com o banco de dados e a coleção que utilizaremos está criado, usaremos o código 1 para inserir os primeiros dados:

```
> db.vendas.insertMany( [
      {
      nome: "João",
      vip: 1,
      email: "joão@email.com",
      telefone: [99991111, 88881111]
      },
      {
      nome: "Marcos",
      vip: 0,
      telefone: [99992222]
      },
      {
      nome: "Maria",
      vip: 1,
      email: "maria@email.com",
      telefone: [99993333, 88883333, 99883000]
      }
  ] )
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('6629a5848437ea171c5dd7da'),
      '1': ObjectId('6629a5848437ea171c5dd7db'),
      '2': ObjectId('6629a5848437ea171c5dd7dc')
    }
  }
```

Logo em seguida utilizaremos o código 2 para inserir os endereços como objetos dentro de cada documento da coleção

Entrada

```
> db.vendas.updateOne({
    nome: "João"
}, {
    $set: {
        endereço: {
            rua:"Um",
            numero:1000,
            complemento:"Apto 1 Bloco 1",
            cidade:"São Paulo",
            estado:"SP"
        }
    }
})
db.vendas.updateOne({
    nome: "Marcos"
}, {
    $set: {
        endereço: {
            rua:"Dois",
            numero:4000,
            cidade:"Campinas",
            estado:"SP"
        }
    }
})
db.vendas.updateOne({
    nome: "Maria"
}, {
    $set: {
        endereço: {
            rua:"Três",
            numero:3000,
            cidade:"Londrina",
            estado:"PR"
        }
    }
})
```

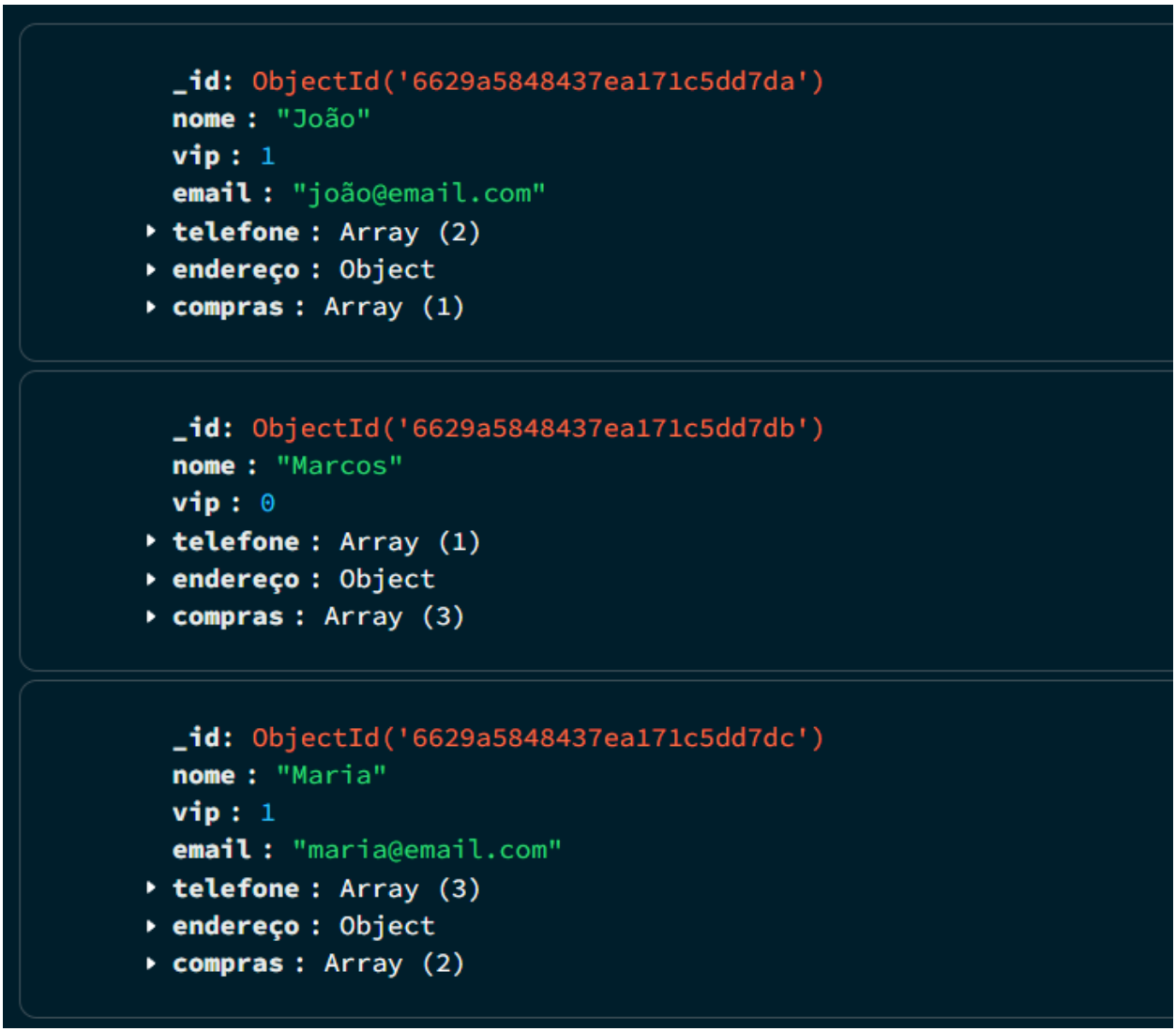
Saída

```
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

Por fim usaremos o código 3 para inserir os as matrizes de objetos em cada documento:

```
>_MONGOSH
> db.vendas.updateOne({
      nome: "João"
  }, {$set: {
          compras: [{
                  produto:"notebook",
                  valor:3000.00,
                  quantia:1
              }]}})
  db.vendas.updateOne({
      nome: "Marcos"
  }, {$set: {
          compras: [{
                  produto:"caderno",
                  valor:20.00,
                  quantia:1
              },{
                  produto:"caneta",
                  valor:3.00,
                  quantia:5
              },{
                  produto:"borracha",
                  valor:2.00,
                  quantia:2
              }]}})
  db.vendas.updateOne({
      nome: "Maria"
  }, {$set: {
          compras: [{
                  produto:"tablet",
                  valor:2500.00,
                  quantia:1
              },{
                  produto:"capa para tablet",
                  valor:50.00,
                  quantia:1
              }]}})
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

Através do Compass podemos ver que todos os dados foram inseridos no formato correto:



Agora que todos os dados estão atualizados realizaremos as 5 consultas propostas no roteiro deste projeto.

Resultados da consulta 1:

```
> db.vendas.find().pretty()
< {
    _id: ObjectId('6629a5848437ea171c5dd7da'),
    nome: 'João',
    vip: 1,
    email: 'joão@email.com',
    telefone: [
      99991111,
      88881111
    ],
    'endereço': {
      rua: 'Um',
      numero: 1000,
      complemento: 'Apto 1 Bloco 1',
      cidade: 'São Paulo',
      estado: 'SP'
    },
    compras: [
      {
        produto: 'notebook',
        valor: 3000,
        quantia: 1
      }
    ]
  }
```

Consulta 1 parte 1

```
{
  _id: ObjectId('6629a5848437ea171c5dd7db'),
  nome: 'Marcos',
  vip: 0,
  telefone: [
    99992222
  ],
  'endereço': {
    rua: 'Dois',
    numero: 4000,
    cidade: 'Campinas',
    estado: 'SP'
  },
  compras: [
    {
      produto: 'caderno',
      valor: 20,
      quantia: 1
    },
    {
      produto: 'caneta',
      valor: 3,
      quantia: 5
    },
    {
      produto: 'borracha',
      valor: 2,
      quantia: 2
    }
  ]
}
```

Consulta 1 parte 2

Consulta 1 parte 3

Resultados da consulta 2:

```
>_MONGOSH

> db.vendas.find({nome:"Maria"})
< {
    _id: ObjectId('6629a5848437ea171c5dd7dc'),
    nome: 'Maria',
    vip: 1,
    email: 'maria@email.com',
    telefone: [
      99993333,
      88883333,
      99883000
    ],
    'endereço': {
      rua: 'Três',
      numero: 3000,
      cidade: 'Londrina',
      estado: 'PR'
    },
    compras: [
      {
        produto: 'tablet',
        valor: 2500,
        quantia: 1
      },
      {
        produto: 'capa para tablet',
        valor: 50,
        quantia: 1
      }
    ]
  }
```

Consulta 2

Resultados da consulta 3:

```
>_MONGOSH
> db.vendas.find( {vip: 1 },
  { _id:0, vip:0, email:0, telefone:0, endereço:0,compras:0 })
< {
    nome: 'João'
  }
  {
    nome: 'Maria'
  }
```

Consulta 3

Resultados da consulta 4:

```
>_MONGOSH
> db.vendas.find({"nome": "Marcos",} ,
  {_id:0, vip:0, email:0, telefone:0, endereço:0})
< {
    nome: 'Marcos',
    compras: [
      {
        produto: 'caderno',
        valor: 20,
        quantia: 1
      },
      {
        produto: 'caneta',
        valor: 3,
        quantia: 5
      },
      {
        produto: 'borracha',
        valor: 2,
        quantia: 2
      }
    ]
  }
```

Consulta 4

Resultados da consulta 5:

```
>_MONGOSH

> db.vendas.aggregate([
      { $unwind: "$compras" },
      { $group: { _id: null, valores: { $addToSet: "$compras.produto" } } }
    ]);
< {
    _id: null,
    valores: [
      'caderno',
      'tablet',
      'caneta',
      'capa para tablet',
      'borracha',
      'notebook'
    ]
  }
```

Consulta 5

4. Conclusão

A conclusão deste trabalho confirma que o aluno possui habilidade na criação e manipulação de bancos de dados não relacionais utilizando o MongoDB. Ao longo do projeto, foi demonstrado a habilidade em criar, ler e atualizar documentos, além de realizar consultas específicas de forma eficiente. Essas capacidades são fundamentais para o desenvolvimento de aplicações modernas que demandam flexibilidade e escalabilidade no armazenamento e gerenciamento de dados. A experiência adquirida neste trabalho proporciona ao aluno uma base sólida para explorar e aprofundar seus conhecimentos em tecnologias de banco de dados não relacionais, capacitando-o para manipular banco de dados de forma clara e objetiva.