Atividade Semestral

Banco de Dados - 2023-1

Pode ser feito em duplas ou individual.

# Título do Projeto: Sistema de Gerenciamento de Pedidos para uma Pizzaria

## Objetivos

Desenvolver um sistema que permita o gerenciamento de pedidos em uma pizzaria utilizando MongoDB. O sistema deve permitir a inserção, consulta, atualização e exclusão de dados (operações CRUD).

## Descrição

Nesta atividade, vocês irão aplicar os conceitos aprendidos sobre banco de dados NoSQL, especificamente utilizando MongoDB, para implementar um sistema de gerenciamento de pedidos para uma pizzaria.

A pizzaria oferece vários tipos de pizzas, com tamanhos e preços diferentes, além de bebidas. Os clientes podem fazer pedidos para entrega ou para comer no local. Cada pedido pode conter múltiplas pizzas e bebidas, e deve-se manter um registro dos detalhes do cliente (nome, endereço para entrega, telefone), data e hora do pedido, itens do pedido e o total do pedido.

Regras de negócio:

1. Cada cliente pode ter múltiplos pedidos, mas cada pedido pertence a um único cliente.
2. Cada pedido pode ter várias pizzas e/ou bebidas, com quantidades variáveis.
3. As pizzas têm tamanhos (pequena, média, grande) que afetam o preço. O preço e as características de cada pizza devem ser armazenados.
4. Os pedidos para entrega devem incluir informações de endereço. Os pedidos para comer no local devem ter a informação do número da mesa.
5. O total do pedido deve ser calculado com base nos itens do pedido.

Cliente:

nome (String)

endereco\_entrega (json)

telefone (String)

Endereco\_entrega

data\_hora (datetime)

taxa\_entrega (double)

rua (Varchar)

numero (String)

complemento (Varchar)

cidade (Varchar)

estado (Varchar)

cep (Varchar)

Pedido:

cliente\_id (Referência ao ID do cliente)

data\_hora (Data e hora do pedido)

tipo\_entrega (String: entrega ou local)

numero\_mesa (Número da mesa para pedidos locais)

itens (Array de objetos contendo os itens do pedido)

Item:

tipo (String: pizza ou bebida)

nome (String)

quantidade (int)

preco\_unitario (Número decimal)

Pizza:

\_id

nome (String)

tamanho (String: pequena, média, grande)

preco\_unitario (double)

quantidade (Número inteiro)

Bebida:

\_id

nome (String)

preco\_unitario (double)

quantidade (Número inteiro)

db.createCollection("clientes")

db.createCollection("pedidos")

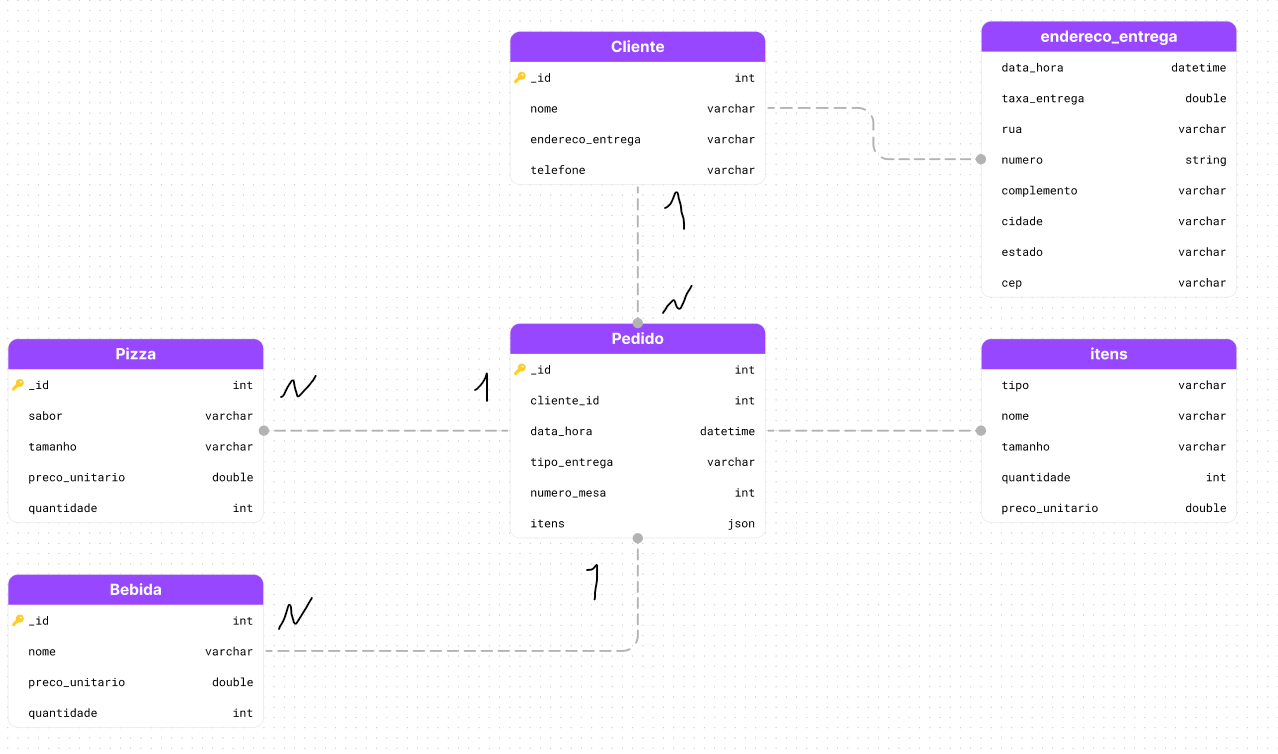
db.createCollection("pizzas")

db.createCollection("bebidas")

## 

## Entregáveis:

1. Diagrama de entidade e relacionamento (DER) para o banco de dados NoSQL. (pode fazer no figma a modelagem)



1. Estruturas JSON para todas as entidades identificadas na modelagem.

**CLIENTE**

{

"nome": "Nome do Cliente",

"endereco\_entrega": {

"rua": "Rua do Cliente",

"numero": "Número do Cliente",

"complemento": "Complemento do Cliente",

"cidade": "Cidade do Cliente",

"estado": "Estado do Cliente",

"cep": "CEP do Cliente"

},

"telefone": "Telefone do Cliente"

}

**PEDIDO**

{

"cliente\_id": "ID do Cliente",

"data\_hora": "Data e Hora do Pedido",

"tipo\_entrega": "Tipo de Entrega (entrega ou local)",

"numero\_mesa": "Número da Mesa (para pedidos locais)",

"itens": [

{

"tipo": "Tipo do Item (pizza ou bebida)",

"nome": "Nome do Item",

"quantidade": "Quantidade do Item",

"preco\_unitario": "Preço Unitário do Item"

},

{

"tipo": "Tipo do Item (pizza ou bebida)",

"nome": "Nome do Item",

"quantidade": "Quantidade do Item",

"preco\_unitario": "Preço Unitário do Item"

},

...

]

}

**PIZZA**

{

"nome": "Nome da Pizza",

"tamanho": "Tamanho da Pizza (pequena, média, grande)",

"preco\_unitario": "Preço Unitário da Pizza",

"quantidade": "Quantidade de Pizzas"

}

**BEBIDA**

{

"nome": "Nome da Bebida",

"preco\_unitario": "Preço Unitário da Bebida",

"quantidade": "Quantidade de Bebidas"

}

1. Comandos MongoDB para criar as coleções e inserir os dados.

db.clientes.insertMany([

{

"nome": "João Silva",

"endereco\_entrega": "Rua A, 123",

"telefone": "(11) 987654321"

},

{

"nome": "Maria Souza",

"endereco\_entrega": "Avenida B, 456",

"telefone": "(22) 123456789"

}

])

db.pedidos.insertOne({

"data\_hora": new Date(),

"tipo\_entrega": "entrega",

"numero\_mesa": null,

"itens": [

{

"tipo": "pizza",

"nome": "Calabresa",

"quantidade": 2,

"preco\_unitario": 25.99

},

{

"tipo": "bebida",

"nome": "Coca-Cola",

"quantidade": 3,

"preco\_unitario": 5.99

}

]

})

db.pizzas.insertMany([

{

"nome": "Calabresa",

"tamanho": "grande",

"preco\_unitario": 25.99,

"quantidade": 10

},

{

"nome": "Mussarela",

"tamanho": "média",

"preco\_unitario": 20.99,

"quantidade": 5

}

])

db.bebidas.insertMany([

{

"nome": "Coca-Cola",

"preco\_unitario": 5.99,

"quantidade": 20

},

{

"nome": "Guaraná",

"preco\_unitario": 4.99,

"quantidade": 15

}

])

1. Comandos MongoDB para realizar atualizações e exclusões de dados.

**Atualização de um documento na coleção "clientes":**

db.clientes.updateOne(

{ "\_id": ObjectId("64930f901e3a4390f3826139") },

{ $set: { "telefone": "(11) 999999999" } }

)

**Atualização de um documento na coleção "pedidos":**

db.pedidos.updateOne(

{ "\_id": ObjectId("64930fb51e3a4390f382613b") },

{ $set: { "tipo\_entrega": "local" } }

)

**Adição de um item em um array de documentos na coleção "pedidos":**

db.pedidos.updateOne(

{ "\_id": ObjectId("64930fb51e3a4390f382613b") },

{ $push: { "itens": { "tipo": "pizza", "nome": "Margherita", "quantidade": 1, "tamanho": "media", "preco\_unitario": 22.99 } } }

)

**Remoção de um documento na coleção "clientes":**

db.clientes.deleteOne({ "\_id": ObjectId("64930f901e3a4390f3826139") })

**Remoção de um item de um array de documentos na coleção "pedidos":**

db.pedidos.updateOne(

{ "\_id": ObjectId("64930fb51e3a4390f382613b") },

{ $pull: { "itens": { "tipo": "pizza", "nome": "Calabresa" } } }

)

1. Comandos MongoDB para as consultas que devem incluir pelo menos uma de cada: consulta simples, consulta com uma condição.

Consulta simples para retornar todos os documentos da coleção "clientes":

db.clientes.find()

Consulta com uma condição para retornar os pedidos com tipo de entrega igual a "entrega":

db.pedidos.find({ "tipo\_entrega": "local" })

Consulta para mostrar os clientes que pediram a pizza Margherita:

db.pedidos.find({ "itens.nome": "Margherita", "itens.tipo": "pizza" })

Consulta para mostrar os clientes que pediram a bebida Coca-Cola:

db.pedidos.find({ "itens.nome": "Coca-Cola", "itens.tipo": "bebida" })

Consulta para mostrar os pedidos que contêm tanto a pizza Margherita quanto a bebida Coca-Cola:

db.pedidos.find({ "itens.nome": { $in: ["Margherita", "Coca-Cola"] } })

1. Comando MongoDB com 2 consultas usando Aggregate.   
    uma que utilize join e outra que utilize group by.
2. Índices apropriados para melhorar o desempenho das consultas. Explicar por que desses índices.
3. Deve ser entregue o link do github com a modelagem, os JSONs e os scripts MongoDB.