Atividade 2
Manipulação de Dados no MongoDB 👾

1. Implemente em Python as **funções de manipulação** da Base de Dados Não Relacional do **Mercado Livre** criado no **MongoDB**

    a. **Insert em todas as coleções**

Clientes

```python
def cadastro_usuario():
    nome = input("Nome do usuário: ")
    sobrenome = input("Sobrenome: ")
    email = input("Email: ")
    if not validar_email(email):
        exibir_mensagem_erro("Email inválido.")
        return
    telefone = input("Telefone: ")
    if not validar_telefone(telefone):
        exibir_mensagem_erro("Telefone inválido.")
        return
    senha = input("Senha: ")

    enderecos = []
    while True:
        enderecos.append(obter_endereco())
        adicionar_mais = input("Deseja adicionar outro endereço? (s/n): ").strip().lower()
        if adicionar_mais != 's':
            break

    data_nascimento = obter_data_nascimento()

    usuario = {
        "nome": nome,
        "sobrenome": sobrenome,
        "email": email,
        "telefone": telefone,
        "senha": senha,
        "enderecos": enderecos,
        "data_nascimento": data_nascimento.strftime("%d/%m/%Y")
    }
    add_usuario(usuario)

def add_usuario(usuario):
    if clientes_collection.find_one({"email": usuario["email"]}):
        exibir_mensagem_erro("O email já está cadastrado.")
    else:
        clientes_collection.insert_one(usuario)
        exibir_mensagem_sucesso("Usuário cadastrado com sucesso!")
```

Vendedores

```python
def cadastro_vendedor():
    nome = input("Nome do vendedor: ")
    sobrenome = input("Sobrenome: ")
    email = input("Email: ")
    if not validar_email(email):
        exibir_mensagem_erro("Email inválido.")
        return
    telefone = input("Telefone: ")
    if not validar_telefone(telefone):
        exibir_mensagem_erro("Telefone inválido.")
        return
    reputacao = float(input("Reputação (0 a 5): "))

    enderecos = []
    while True:
        enderecos.append(obter_endereco())
        adicionar_mais = input("Deseja adicionar outro endereço? (s/n): ").strip().lower()
        if adicionar_mais != 's':
            break

    data_nascimento = obter_data_nascimento()

    vendedor = {
        "nome": nome,
        "sobrenome": sobrenome,
        "email": email,
        "telefone": telefone,
        "reputacao": reputacao,
        "enderecos": enderecos,
        "data_nascimento": data_nascimento.strftime("%d/%m/%Y")
    }
    add_vendedor(vendedor)

def add_vendedor(vendedor):
    if vendedores_collection.find_one({"email": vendedor["email"]}):
        exibir_mensagem_erro("O email já está cadastrado.")
    else:
        vendedores_collection.insert_one(vendedor)
        exibir_mensagem_sucesso("Vendedor cadastrado com sucesso!")
```

Produtos

```python
def cadastro_produto():
    nome = input("Nome do produto: ")
    preco = input("Preço: ")
    if not validar_preco(preco):
        exibir_mensagem_erro("Preço inválido.")
        return
    preco = float(preco)
    descricao = input("Descrição: ")
    id_vendedor = input("ID do vendedor: ")
    estoque = int(input("Estoque: "))
    categoria = input("Categoria: ")
    data_adicao_favoritos = datetime.now().strftime("%d/%m/%Y %H:%M:%S")
    id_produto = str(uuid.uuid4())
    status_disponibilidade = "Disponível" if estoque > 0 else "Indisponível"

    produto = {
        "id_produto": id_produto,
        "nome": nome,
        "preco": preco,
        "descricao": descricao,
        "id_vendedor": id_vendedor,
        "estoque": estoque,
        "categoria": categoria,
        "status_disponibilidade": status_disponibilidade,
        "data_adicao_favoritos": data_adicao_favoritos
    }
    add_produto(produto)

def add_produto(produto):
    produtos_collection.insert_one(produto)
    exibir_mensagem_sucesso("Produto cadastrado com sucesso!")
```

Compras

```python
def cadastro_compra():
    usuario_email = input("Email do usuário: ")
    id_produto = input("ID do produto: ")
    quantidade = int(input("Quantidade: "))

    produto = produtos_collection.find_one({"_id": ObjectId(id_produto)})
    if not produto:
        exibir_mensagem_erro("Produto não encontrado!")
        return

    preco_unitario = produto["preco"]
    preco_total = preco_unitario * quantidade
    data_compra = datetime.now().strftime("%d/%m/%Y %H:%M:%S")
    status_compra = "Pendente"
    id_vendedor = produto["id_vendedor"]
    id_usuario = input("ID do usuário: ")

    compra = {
        "usuario_email": usuario_email,
        "produto_nome": produto["nome"],
        "quantidade": quantidade,
        "preco_unitario": preco_unitario,
        "preco_total": preco_total,
        "data_compra": data_compra,
        "status_compra": status_compra,
        "id_vendedor": id_vendedor,
        "id_usuario": id_usuario,
        "id_produto": ObjectId(id_produto)
    }
    add_compra(compra)

def add_compra(compra):
    compras_collection.insert_one(compra)
    exibir_mensagem_sucesso("Compra registrada com sucesso!")
```

Favoritos

```python
def adicionar_favorito():
    email_usuario = input("Email do usuário: ")
    id_produto = input("ID do produto: ")

    usuario = clientes_collection.find_one({"email": email_usuario})

    if not usuario:
        exibir_mensagem_erro("Usuário não encontrado!")
        return

    favoritos = usuario.get("favoritos", [])
    if id_produto in favoritos:
        exibir_mensagem_erro("Produto já está nos favoritos!")
        return

    favoritos.append(id_produto)
    clientes_collection.update_one({"email": email_usuario}, {"$set": {"favoritos": favoritos}})

    produto = produtos_collection.find_one({"_id": ObjectId(id_produto)})
    if produto:
        print(f"Produto adicionado aos favoritos: Nome: {produto['nome']}, Detalhes: {produto}")
    exibir_mensagem_sucesso("Produto adicionado aos favoritos com sucesso!")
```

b. Update em todas as coleções

Clientes

```python
def atualizacao_usuario():
    emailI = input("Email do usuário à atualizar: ")
    print("Quais campos irá atualizar?")
    print("01 - Nome")
    print("02 - Sobrenome")
    print("03 - Email")
    print("04 - Telefone")
    print("05 - Endereços")
    print("06 - Data de Nascimento")
    print("07 - Senha")
    campos = input("Quais os campos? *modelo: 01,02,03: ")
    campos = campos.split(",")
    novosValores = {}
    for campo in campos:
        campo = int(campo)
        if campo == 1:
            nome = input("Novo nome: ")
            novosValores["nome"] = nome
        elif campo == 2:
            sobrenome = input("Novo sobrenome: ")
            novosValores["sobrenome"] = sobrenome
        elif campo == 3:
            email = input("Novo email: ")
            if not validar_email(email):
                exibir_mensagem_erro("Email inválido.")
                return
            novosValores["email"] = email
        elif campo == 4:
            telefone = input("Novo telefone: ")
            if not validar_telefone(telefone):
                exibir_mensagem_erro("Telefone inválido.")
                return
            novosValores["telefone"] = telefone
        elif campo == 5:
            enderecos = []
            while True:
                enderecos.append(obter_endereco())
                adicionar_mais = input("Deseja adicionar outro endereço? (s/n): ").strip().lower()
                if adicionar_mais != 's':
                    break
            novosValores["enderecos"] = enderecos
        elif campo == 6:
            data_nascimento = obter_data_nascimento()
            novosValores["data_nascimento"] = data_nascimento.strftime("%d/%m/%Y")
        elif campo == 7:
            senha = input("Nova senha: ")
            novosValores["senha"] = senha
    update_usuario(emailI, novosValores)
```

```python
def update_usuario(emailI, novosValores):
    usuario_existente = clientes_collection.find_one({"email": emailI})
    if usuario_existente:
        clientes_collection.update_one({"email": emailI}, {"$set": novosValores})
        exibir_mensagem_sucesso("Usuário atualizado com sucesso!")
    else:
        exibir_mensagem_erro("Usuário não encontrado!")
```

Vendedores

```python
def atualizacao_vendedor():
    emailI = input("Email do vendedor à atualizar: ")
    print("Quais campos irá atualizar?")
    print("01 - Nome")
    print("02 - Sobrenome")
    print("03 - Email")
    print("04 - Telefone")
    print("05 - Endereços")
    print("06 - Data de Nascimento")
    print("07 - Reputação")
    campos = input("Quais os campos? *modelo: 01,02,03: ")
    campos = campos.split(",")
    novosValores = {}
    for campo in campos:
        campo = int(campo)
        if campo == 1:
            nome = input("Novo nome: ")
            novosValores["nome"] = nome
        elif campo == 2:
            sobrenome = input("Novo sobrenome: ")
            novosValores["sobrenome"] = sobrenome
        elif campo == 3:
            email = input("Novo email: ")
            if not validar_email(email):
                exibir_mensagem_erro("Email inválido.")
                return
            novosValores["email"] = email
        elif campo == 4:
            telefone = input("Novo telefone: ")
            if not validar_telefone(telefone):
                exibir_mensagem_erro("Telefone inválido.")
                return
            novosValores["telefone"] = telefone
        elif campo == 5:
            enderecos = []
            while True:
                enderecos.append(obter_endereco())
                adicionar_mais = input("Deseja adicionar outro endereço? (s/n): ").strip().lower()
                if adicionar_mais != 's':
                    break
            novosValores["enderecos"] = enderecos
        elif campo == 6:
            data_nascimento = obter_data_nascimento()
            novosValores["data_nascimento"] = data_nascimento.strftime("%d/%m/%Y")
        elif campo == 7:
            reputacao = float(input("Nova reputação (0 a 5): "))
            novosValores["reputacao"] = reputacao
    update_vendedor(emailI, novosValores)
```

```python
def update_vendedor(emailI, novosValores):
    vendedor_existente = vendedores_collection.find_one({"email": emailI})
    if vendedor_existente:
        vendedores_collection.update_one({"email": emailI}, {"$set": novosValores})
        exibir_mensagem_sucesso("Vendedor atualizado com sucesso!")
    else:
        exibir_mensagem_erro("Vendedor não encontrado!")
```

Produtos

```python
def atualizacao_produto():
    id_produto = input("ID do produto à atualizar: ")
    print("Quais campos irá atualizar?")
    print("01 - Nome")
    print("02 - Preço")
    print("03 - Descrição")
    print("04 - ID do Vendedor")
    print("05 - Estoque")
    print("06 - Categoria")
    print("07 - Data de Adição aos Favoritos")
    campos = input("Quais os campos? *modelo: 01,02,03: ")
    campos = campos.split(",")
    novosValores = {}
    for campo in campos:
        campo = int(campo)
        if campo == 1:
            nome = input("Novo nome: ")
            novosValores["nome"] = nome
        elif campo == 2:
            preco = input("Novo preço: ")
            if not validar_preco(preco):
                exibir_mensagem_erro("Preço inválido.")
                return
            novosValores["preco"] = float(preco)
        elif campo == 3:
            descricao = input("Nova descrição: ")
            novosValores["descricao"] = descricao
        elif campo == 4:
            id_vendedor = input("Novo ID do vendedor: ")
            novosValores["id_vendedor"] = id_vendedor
        elif campo == 5:
            estoque = int(input("Novo estoque: "))
            novosValores["estoque"] = estoque
        elif campo == 6:
            categoria = input("Nova categoria: ")
            novosValores["categoria"] = categoria
        elif campo == 7:
            data_adicao_favoritos = input("Nova data de adição aos favoritos (dd/mm/yyyy hh:mm:ss): ")
            novosValores["data_adicao_favoritos"] = data_adicao_favoritos
    update_produto(id_produto, novosValores)

def update_produto(id_produto, novosValores):
    produto_existente = produtos_collection.find_one({"id_produto": id_produto})
    if produto_existente:
        produtos_collection.update_one({"id_produto": id_produto}, {"$set": novosValores})
        exibir_mensagem_sucesso("Produto atualizado com sucesso!")
    else:
        exibir_mensagem_erro("Produto não encontrado!")
```

Compras

```python
def atualizacao_compra():
    id_compra = input("ID da compra à atualizar: ")
    compra_existente = compras_collection.find_one({"_id": ObjectId(id_compra)})
    if not compra_existente:
        exibir_mensagem_erro("Compra não encontrada!")
        return

    print("Quais campos irá atualizar?")
    print("01 - Email do Usuário")
    print("02 - ID do Produto")
    print("03 - Quantidade")
    print("04 - Status da Compra")
    campos = input("Quais os campos? *modelo: 01,02,03: ")
    campos = campos.split(",")
    novosValores = {}
    for campo in campos:
        campo = int(campo)
        if campo == 1:
            usuario_email = input("Novo email do usuário: ")
            novosValores["usuario_email"] = usuario_email
        elif campo == 2:
            id_produto = input("Novo ID do produto: ")
            produto = produtos_collection.find_one({"_id": ObjectId(id_produto)})
            if not produto:
                exibir_mensagem_erro("Produto não encontrado!")
                return
            novosValores["id_produto"] = ObjectId(id_produto)
            novosValores["produto_nome"] = produto["nome"]
            novosValores["preco_unitario"] = produto["preco"]
        elif campo == 3:
            quantidade = int(input("Nova quantidade: "))
            novosValores["quantidade"] = quantidade
            novosValores["preco_total"] = compra_existente["preco_unitario"] * quantidade
        elif campo == 4:
            status_compra = input("Novo status da compra: ")
            novosValores["status_compra"] = status_compra
    update_compra(id_compra, novosValores)

def update_compra(id_compra, novosValores):
    compra_existente = compras_collection.find_one({"_id": ObjectId(id_compra)})
    if compra_existente:
        compras_collection.update_one({"_id": ObjectId(id_compra)}, {"$set": novosValores})
        exibir_mensagem_sucesso("Compra atualizada com sucesso!")
    else:
        exibir_mensagem_erro("Compra não encontrada!")
```

c. Search em todas as coleções

Clientes

```
def listagem_clientes():
    clientes = clientes_collection.find()
    for cliente in clientes:
        print(cliente)
    voltar_opcoes()
```

Vendedores

```
def listagem_vendedores():
    vendedores = vendedores_collection.find()
    for vendedor in vendedores:
        print(vendedor)
    voltar_opcoes()
```

Produtos

```
def listagem_produtos():
    produtos = produtos_collection.find()
    for produto in produtos:
        print(produto)
    voltar_opcoes()
```

Compras

```
def listagem_compras():
    compras = compras_collection.find()
    for compra in compras:
        print(compra)
    voltar_opcoes()
```

Favoritos

```
def listar_favoritos():
    email_usuario = input("Email do usuário: ")

    usuario = clientes_collection.find_one({"email": email_usuario})

    if not usuario:
        exibir_mensagem_erro("Usuário não encontrado!")
        return

    favoritos = usuario.get("favoritos", [])
    if not favoritos:
        print("Nenhum produto nos favoritos.")
        return

    print("Produtos favoritos:")
    for id_produto in favoritos:
        produto = produtos_collection.find_one({"_id": ObjectId(id_produto)})
        if produto:
            print(f"ID: {produto['_id']}, Nome: {produto['nome']}, Detalhes: {produto}")
    voltar_opcoes()
```

d. Delete em todas as coleções

Clientes

```
def listagem_clientes():
    clientes = clientes_collection.find()
    for cliente in clientes:
        print(cliente)
    voltar_opcoes()
```

Vendedores

```
def listagem_vendedores():
    vendedores = vendedores_collection.find()
    for vendedor in vendedores:
        print(vendedor)
    voltar_opcoes()
```

Produtos

```
def listagem_produtos():
    produtos = produtos_collection.find()
    for produto in produtos:
        print(produto)
    voltar_opcoes()
```

Compras

```
def listagem_compras():
    compras = compras_collection.find()
    for compra in compras:
        print(compra)
    voltar_opcoes()
```

Favoritos

```
def remover_favorito():
    email_usuario = input("Email do usuário: ")
    id_produto = input("ID do produto: ")

    usuario = clientes_collection.find_one({"email": email_usuario})

    if not usuario:
        exibir_mensagem_erro("Usuário não encontrado!")
        return

    favoritos = usuario.get("favoritos", [])
    if id_produto not in favoritos:
        exibir_mensagem_erro("Produto não está nos favoritos!")
        return

    favoritos.remove(id_produto)
    clientes_collection.update_one({"email": email_usuario}, {"$set": {"favoritos": favoritos}})
    exibir_mensagem_sucesso("Produto removido dos favoritos com sucesso!")
```