

Atividade de Banco de Dados

Acessando um Banco de Dados através do Python

Entre no Play With Docker e siga o tutorial abaixo:

>>Em uma **primeira instância**, criamos o container do Docker, usando o comando:

```
docker run --name mysql-container -e MYSQL_ROOT_PASSWORD=root -e  
MYSQL_DATABASE=BD_LOJA -p 3306:3306 -d mysql:latest
```

>>**Entre** no Container **MySQL** para executar os comandos de criação de tabelas, usando o comando:

```
docker exec -it mysql-container mysql -uroot -proot
```

>>Crie, nessa mesma instância, o banco de dados e a tabela TB_CLIENTES, usando o comando MySQL:

```
USE BD_LOJA  
CREATE TABLE TB_ESCRITORIOS (  
  id int NOT NULL AUTO_INCREMENT,  
  cidade text,  
  phone text,  
  endereco_pt1 text,  
  endereco_pt2 text,  
  estado text,  
  pais text,  
  codigo_postal text,  
  territorio text,  
  PRIMARY KEY (id)  
);
```

```
CREATE TABLE TB_FUNCIONARIOS (  
  id int NOT NULL AUTO_INCREMENT,  
  nome_ultimo varchar(100),  
  nome_primeiro varchar(100),  
  extensao varchar(10),  
  email varchar(255),  
  escritorio_id int,  
  relatorios_para_funcionario_id int,  
  trabalho varchar(100),  
  PRIMARY KEY (id),  
  FOREIGN KEY (escritorio_id) REFERENCES TB_ESCRITORIOS (id),
```

```
FOREIGN KEY (relatorios_para_funcionario_id) REFERENCES TB_FUNCIONARIOS (id)
);
```

```
CREATE TABLE TB_CLIENTES (
  id int NOT NULL AUTO_INCREMENT,
  nome text,
  nome_ultimo varchar(100),
  nome_primeiro varchar(100),
  telefone text,
  endereco_pt1 varchar(255),
  endereco_pt2 varchar(255),
  cidade varchar(50),
  estado varchar(50),
  codigo_postal varchar(20),
  pais varchar(50),
  funcionario_id int,
  limite_credito double,
  PRIMARY KEY (id),
  FOREIGN KEY (funcionario_id) REFERENCES TB_FUNCIONARIOS (id)
);
```

```
CREATE TABLE TB_LINHAS_PRODUTOS (
  id int NOT NULL AUTO_INCREMENT,
  descricao text,
  descricao_html longtext,
  image text,
  PRIMARY KEY (id)
);
```

```
CREATE TABLE TB_PRODUTOS (
  id int NOT NULL AUTO_INCREMENT,
  nome text,
  linha_produto_id int,
  escala text,
  fornecedor text,
  descricao text,
  quantidade_estoque int,
  preco double,
  msrp double,
  PRIMARY KEY (id),
  FOREIGN KEY (linha_produto_id) REFERENCES TB_LINHAS_PRODUTOS (id)
);
```

```
CREATE TABLE TB_PEDIDOS (
  id int NOT NULL AUTO_INCREMENT,
  data_pedido date,
  data_entrega date,
  data_envio date,
```

```
status text,  
comentarios text,  
cliente_id int,  
PRIMARY KEY (id),  
FOREIGN KEY (cliente_id) REFERENCES TB_CLIENTES (id)  
);
```

```
CREATE TABLE TB_DETALHES_PEDIDO (  
pedido_id int NOT NULL,  
produto_id int NOT NULL,  
quantidade_pedida int,  
preco_unitario double,  
numero_linha_pedido int,  
PRIMARY KEY (pedido_id,produto_id),  
FOREIGN KEY (pedido_id) REFERENCES TB_PEDIDOS (id),  
FOREIGN KEY (produto_id) REFERENCES TB_PRODUTOS (id)  
);
```

```
CREATE TABLE TB_PAGAMENTOS (  
id int NOT NULL AUTO_INCREMENT,  
cliente_id int NOT NULL,  
data_pagamento date,  
valor double,  
PRIMARY KEY (id,cliente_id),  
FOREIGN KEY (cliente_id) REFERENCES TB_CLIENTES (id)  
);
```

>>Criamos uma **segunda** instância no PlayWithDocker, e criamos um **ambiente virtual** do python, usando os comandos:

```
python -m venv myenv
```

>>**Entramos** nela, usando:

```
source myenv/bin/activate
```

>>Depois, **instalamos** o **Conector Python**, usando os comandos:

```
pip install mysql-connector-python
```

>>Depois, **criamos** o **arquivo** app.py:

```
vi app.py
```

>>E o **editamos**, colocando esses **comandos**, lembre-se de colocar o ip da primeira instância criada na variável host, veja:

```
import mysql.connector
```

```

from mysql.connector import Error
from datetime import date

def create_connection():
    """Cria uma conexão com o banco de dados MySQL."""
    connection = None
    try:
        connection = mysql.connector.connect(
            host='COLOCAR AQUI O IP DA PRIMEIRA INSTÂNCIA CRIADA!',
            port='3306',
            user='root',
            password='root',
            database='BD_LOJA'
        )
        print("Conexão com o MySQL bem-sucedida")
    except Error as e:
        print(f"Erro '{e}' ocorreu")
    return connection

# CRUD para TB_CLIENTES
def create_cliente(connection, nome, nome_ultimo, nome_primeiro,
telefone, endereco_pt1,
                endereco_pt2, cidade, estado, codigo_postal, pais,
funcionario_id, limite_credito):
    cursor = connection.cursor()
    query = """INSERT INTO TB_CLIENTES (nome, nome_ultimo,
nome_primeiro, telefone,
                endereco_pt1, endereco_pt2, cidade, estado,
codigo_postal, pais,
                funcionario_id, limite_credito)
                VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,
%s)"""
    values = (nome, nome_ultimo, nome_primeiro, telefone, endereco_pt1,
endereco_pt2,
                cidade, estado, codigo_postal, pais, funcionario_id,
limite_credito)
    cursor.execute(query, values)
    connection.commit()
    print("Cliente adicionado com sucesso")

def read_clientes(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_CLIENTES")

```

```

        return cursor.fetchall()

def update_cliente(connection, id, nome, nome_ultimo, nome_primeiro,
telefone, endereco_pt1,
                    endereco_pt2, cidade, estado, codigo_postal, pais,
funcionario_id, limite_credito):
    cursor = connection.cursor()
    query = """UPDATE TB_CLIENTES SET nome=%s, nome_ultimo=%s,
nome_primeiro=%s,
                    telefone=%s, endereco_pt1=%s, endereco_pt2=%s,
cidade=%s, estado=%s,
                    codigo_postal=%s, pais=%s, funcionario_id=%s,
limite_credito=%s WHERE id=%s"""
    values = (nome, nome_ultimo, nome_primeiro, telefone, endereco_pt1,
endereco_pt2,
                    cidade, estado, codigo_postal, pais, funcionario_id,
limite_credito, id)
    cursor.execute(query, values)
    connection.commit()
    print("Cliente atualizado com sucesso")

def delete_cliente(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_CLIENTES WHERE id=%s", (id,))
    connection.commit()
    print("Cliente deletado com sucesso")

# CRUD para TB_PRODUTOS
def create_produto(connection, nome, linha_produto_id, escala,
fornecedor, descricao,
                    quantidade_estoque, preco, msrp):
    cursor = connection.cursor()
    query = """INSERT INTO TB_PRODUTOS (nome, linha_produto_id, escala,
fornecedor,
                    descricao, quantidade_estoque, preco, msrp)
VALUES (%s, %s, %s, %s, %s, %s, %s, %s)"""
    values = (nome, linha_produto_id, escala, fornecedor, descricao,
quantidade_estoque, preco, msrp)
    cursor.execute(query, values)
    connection.commit()
    print("Produto adicionado com sucesso")

def read_produtos(connection):

```

```

        cursor = connection.cursor()
        cursor.execute("SELECT * FROM TB_PRODUTOS")
        return cursor.fetchall()

def update_produto(connection, id, nome, linha_produto_id, escala,
fornecedor,
                    descricao, quantidade_estoque, preco, msrp):
    cursor = connection.cursor()
    query = """UPDATE TB_PRODUTOS SET nome=%s, linha_produto_id=%s,
escala=%s,
                    fornecedor=%s, descricao=%s, quantidade_estoque=%s,
preco=%s, msrp=%s
                    WHERE id=%s"""
    values = (nome, linha_produto_id, escala, fornecedor, descricao,
                quantidade_estoque, preco, msrp, id)
    cursor.execute(query, values)
    connection.commit()
    print("Produto atualizado com sucesso")

def delete_produto(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_PRODUTOS WHERE id=%s", (id,))
    connection.commit()
    print("Produto deletado com sucesso")

# CRUD para TB_PEDIDOS
def create_pedido(connection, data_pedido, data_entrega, data_envio,
status,
                    comentarios, cliente_id):
    cursor = connection.cursor()
    query = """INSERT INTO TB_PEDIDOS (data_pedido, data_entrega,
data_envio,
                    status, comentarios, cliente_id)
                    VALUES (%s, %s, %s, %s, %s, %s)"""
    values = (data_pedido, data_entrega, data_envio, status,
comentarios, cliente_id)
    cursor.execute(query, values)
    connection.commit()
    print("Pedido adicionado com sucesso")

def read_pedidos(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_PEDIDOS")

```

```

        return cursor.fetchall()

def update_pedido(connection, id, data_pedido, data_entrega,
data_envio,
                    status, comentarios, cliente_id):
    cursor = connection.cursor()
    query = """UPDATE TB_PEDIDOS SET data_pedido=%s, data_entrega=%s,
data_envio=%s,
                    status=%s, comentarios=%s, cliente_id=%s WHERE id=%s"""
    values = (data_pedido, data_entrega, data_envio, status,
comentarios,
                    cliente_id, id)
    cursor.execute(query, values)
    connection.commit()
    print("Pedido atualizado com sucesso")

def delete_pedido(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_PEDIDOS WHERE id=%s", (id,))
    connection.commit()
    print("Pedido deletado com sucesso")

# CRUD para TB_DETALHES_PEDIDO
def create_detalhe_pedido(connection, pedido_id, produto_id,
quantidade_pedida,
                    preco_unitario, numero_linha_pedido):
    cursor = connection.cursor()
    query = """INSERT INTO TB_DETALHES_PEDIDO (pedido_id, produto_id,
quantidade_pedida, preco_unitario, numero_linha_pedido)
VALUES (%s, %s, %s, %s, %s)"""
    values = (pedido_id, produto_id, quantidade_pedida, preco_unitario,
numero_linha_pedido)
    cursor.execute(query, values)
    connection.commit()
    print("Detalhe do pedido adicionado com sucesso")

def read_detalhes_pedido(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_DETALHES_PEDIDO")
    return cursor.fetchall()

def update_detalhe_pedido(connection, pedido_id, produto_id,
quantidade_pedida,

```

```

        preco_unitario, numero_linha_pedido):
    cursor = connection.cursor()
    query = """UPDATE TB_DETALHES_PEDIDO SET quantidade_pedida=%s,
        preco_unitario=%s, numero_linha_pedido=%s
        WHERE pedido_id=%s AND produto_id=%s"""
    values = (quantidade_pedida, preco_unitario, numero_linha_pedido,
        pedido_id, produto_id)
    cursor.execute(query, values)
    connection.commit()
    print("Detalhe do pedido atualizado com sucesso")

def delete_detalhe_pedido(connection, pedido_id, produto_id):
    cursor = connection.cursor()
    cursor.execute("""DELETE FROM TB_DETALHES_PEDIDO
        WHERE pedido_id=%s AND produto_id=%s""",
(pedido_id, produto_id))
    connection.commit()
    print("Detalhe do pedido deletado com sucesso")

# CRUD para TB_PAGAMENTOS
def create_pagamento(connection, cliente_id, data_pagamento, valor):
    cursor = connection.cursor()
    query = """INSERT INTO TB_PAGAMENTOS (cliente_id, data_pagamento,
valor)
        VALUES (%s, %s, %s)"""
    cursor.execute(query, (cliente_id, data_pagamento, valor))
    connection.commit()
    print("Pagamento adicionado com sucesso")

def read_pagamentos(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_PAGAMENTOS")
    return cursor.fetchall()

def update_pagamento(connection, id, cliente_id, data_pagamento,
valor):
    cursor = connection.cursor()
    query = """UPDATE TB_PAGAMENTOS SET data_pagamento=%s, valor=%s
        WHERE id=%s AND cliente_id=%s"""
    cursor.execute(query, (data_pagamento, valor, id, cliente_id))
    connection.commit()
    print("Pagamento atualizado com sucesso")

```



```

def delete_pagamento(connection, id, cliente_id):
    cursor = connection.cursor()
    cursor.execute("""DELETE FROM TB_PAGAMENTOS
                      WHERE id=%s AND cliente_id=%s""", (id,
cliente_id))
    connection.commit()
    print("Pagamento deletado com sucesso")

# CRUD para TB_FUNCIONARIOS
def create_funcionario(connection, nome_ultimo, nome_primeiro,
extensao, email,
                      escritorio_id, relatorios_para_funcionario_id,
trabalho):
    cursor = connection.cursor()
    query = """INSERT INTO TB_FUNCIONARIOS (nome_ultimo, nome_primeiro,
extensao,
                      email, escritorio_id, relatorios_para_funcionario_id,
trabalho)
              VALUES (%s, %s, %s, %s, %s, %s, %s)"""
    values = (nome_ultimo, nome_primeiro, extensao, email,
escritorio_id,
              relatorios_para_funcionario_id, trabalho)
    cursor.execute(query, values)
    connection.commit()
    print("Funcionário adicionado com sucesso")

def read_funcionarios(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_FUNCIONARIOS")
    return cursor.fetchall()

def update_funcionario(connection, id, nome_ultimo, nome_primeiro,
extensao,
                      email, escritorio_id,
relatorios_para_funcionario_id, trabalho):
    cursor = connection.cursor()
    query = """UPDATE TB_FUNCIONARIOS SET nome_ultimo=%s,
nome_primeiro=%s,
                      extensao=%s, email=%s, escritorio_id=%s,
                      relatorios_para_funcionario_id=%s, trabalho=%s WHERE
id=%s"""
    values = (nome_ultimo, nome_primeiro, extensao, email,
escritorio_id,

```

```

        relatorios_para_funcionario_id, trabalho, id)
    cursor.execute(query, values)
    connection.commit()
    print("Funcionário atualizado com sucesso")

def delete_funcionario(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_FUNCIONARIOS WHERE id=%s", (id,))
    connection.commit()
    print("Funcionário deletado com sucesso")

# CRUD para TB_ESCRITORIOS
def create_escritorio(connection, cidade, phone, endereco_pt1,
endereco_pt2,
                        estado, pais, codigo_postal, territorio):
    cursor = connection.cursor()
    query = """INSERT INTO TB_ESCRITORIOS (cidade, phone, endereco_pt1,
                        endereco_pt2, estado, pais, codigo_postal, territorio)
                        VALUES (%s, %s, %s, %s, %s, %s, %s, %s)"""
    values = (cidade, phone, endereco_pt1, endereco_pt2, estado, pais,
                codigo_postal, territorio)
    cursor.execute(query, values)
    connection.commit()
    print("Escritório adicionado com sucesso")

def read_escritorios(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_ESCRITORIOS")
    return cursor.fetchall()

def update_escritorio(connection, id, cidade, phone, endereco_pt1,
endereco_pt2,
                        estado, pais, codigo_postal, territorio):
    cursor = connection.cursor()
    query = """UPDATE TB_ESCRITORIOS SET cidade=%s, phone=%s,
endereco_pt1=%s,
                        endereco_pt2=%s, estado=%s, pais=%s, codigo_postal=%s,
                        territorio=%s WHERE id=%s"""
    values = (cidade, phone, endereco_pt1, endereco_pt2, estado, pais,
                codigo_postal, territorio, id)
    cursor.execute(query, values)
    connection.commit()
    print("Escritório atualizado com sucesso")

```

```

def delete_escritorio(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_ESCRITORIOS WHERE id=%s", (id,))
    connection.commit()
    print("Escritório deletado com sucesso")

# CRUD para TB_LINHAS_PRODUTOS
def create_linha_produto(connection, descricao, descricao_html, image):
    cursor = connection.cursor()
    query = """INSERT INTO TB_LINHAS_PRODUTOS (descricao,
descricao_html, image)
                VALUES (%s, %s, %s)"""
    cursor.execute(query, (descricao, descricao_html, image))
    connection.commit()
    print("Linha de produto adicionada com sucesso")

def read_linhas_produtos(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_LINHAS_PRODUTOS")
    return cursor.fetchall()

def update_linha_produto(connection, id, descricao, descricao_html,
image):
    cursor = connection.cursor()
    query = """UPDATE TB_LINHAS_PRODUTOS SET descricao=%s,
descricao_html=%s,
                image=%s WHERE id=%s"""
    cursor.execute(query, (descricao, descricao_html, image, id))
    connection.commit()
    print("Linha de produto atualizada com sucesso")

def delete_linha_produto(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_LINHAS_PRODUTOS WHERE id=%s", (id,))
    connection.commit()
    print("Linha de produto deletada com sucesso")

def main():
    connection = create_connection()
    if connection is None:
        return

```

```
#VOCÊ SÓ ADICIONA ESSAS INSTÂNCIAS SE QUISER TESTAR O
FUNCIONAMENTO!!! RETIRE, SE NÃO FOR NECESSÁRIO.

#Criando algumas instâncias:

# Criar 3 escritórios
create_escritorio(connection, "São Paulo", "11999999999", "Av
Paulista", "1000", "SP", "Brasil", "01000000", "Sudeste")
create_escritorio(connection, "Rio de Janeiro", "218888888888", "Av
Atlântica", "2000", "RJ", "Brasil", "02000000", "Sudeste")
create_escritorio(connection, "Belo Horizonte", "31777777777", "Av
Afonso Pena", "3000", "MG", "Brasil", "03000000", "Sudeste")

# Criar 3 funcionários
create_funcionario(connection, "Silva", "João", "101",
"joao@email.com", 1, None, "Gerente")
create_funcionario(connection, "Santos", "Maria", "102",
"maria@email.com", 2, 1, "Vendedor")
create_funcionario(connection, "Oliveira", "Pedro", "103",
"pedro@email.com", 3, 1, "Vendedor")

# Criar 3 linhas de produtos
create_linha_produto(connection, "Eletrônicos", "<p>Produtos
eletrônicos</p>", "eletronicos.jpg")
create_linha_produto(connection, "Móveis", "<p>Móveis para
casa</p>", "moveis.jpg")
create_linha_produto(connection, "Roupas", "<p>Vestuário",
"roupas.jpg")

# Criar 3 produtos
create_produto(connection, "Smartphone X", 1, "1:1", "TechCorp",
"Smartphone avançado", 100, 1999.99, 2499.99)
create_produto(connection, "Sofá Confort", 2, "1:1", "MoveisLux",
"Sofá 3 lugares", 50, 2999.99, 3499.99)
create_produto(connection, "Camisa Casual", 3, "1:1",
"FashionStyle", "Camisa manga curta", 200, 99.99, 149.99)

# Criar 3 clientes
create_cliente(connection, "José Pereira", "Pereira", "José",
"119999999999", "Rua A", "123", "São Paulo", "SP", "01000000", "Brasil",
1, 5000.00)
create_cliente(connection, "Ana Santos", "Santos", "Ana",
"218888888888", "Rua B", "456", "Rio de Janeiro", "RJ", "02000000",
"Brasil", 2, 3000.00)
```

```

    create_cliente(connection, "Carlos Lima", "Lima", "Carlos",
"31777777777", "Rua C", "789", "Belo Horizonte", "MG", "03000000",
"Brasil", 3, 4000.00)

# Criar 3 pedidos
    create_pedido(connection, date(2024, 2, 1), date(2024, 2, 10),
date(2024, 2, 5), "Entregue", "Entrega normal", 1)
    create_pedido(connection, date(2024, 2, 2), date(2024, 2, 11),
date(2024, 2, 6), "Entregue", "Entrega expressa", 2)
    create_pedido(connection, date(2024, 2, 3), date(2024, 2, 12),
date(2024, 2, 7), "Em processamento", "Aguardando pagamento", 3)

# Criar 3 detalhes de pedido
    create_detalle_pedido(connection, 1, 1, 2, 1999.99, 1)
    create_detalle_pedido(connection, 2, 2, 1, 2999.99, 1)
    create_detalle_pedido(connection, 3, 3, 3, 99.99, 1)

# Criar 3 pagamentos
    create_pagamento(connection, 1, date(2024, 2, 5), 3999.98)
    create_pagamento(connection, 2, date(2024, 2, 6), 2999.99)
    create_pagamento(connection, 3, date(2024, 2, 7), 299.97)

# Ler todas as tabelas
    print("\nLeitura de todas as tabelas:")
    print("\nEscritórios:", read_escritorios(connection))
    print("\nFuncionários:", read_funcionarios(connection))
    print("\nLinhas de Produtos:", read_linhas_produtos(connection))
    print("\nProdutos:", read_produtos(connection))
    print("\nClientes:", read_clientes(connection))
    print("\nPedidos:", read_pedidos(connection))
    print("\nDetalhes de Pedido:", read_detalle_pedido(connection))
    print("\nPagamentos:", read_pagamentos(connection))

# Atualizar uma instância de cada tabela
    update_escritorio(connection, 1, "São Paulo", "11999999999", "Av
Paulista", "2000", "SP", "Brasil", "01000000", "Sudeste")
    update_funcionario(connection, 1, "Silva", "João Paulo", "101",
"joao.paulo@email.com", 1, None, "Gerente Senior")
    update_linha_produto(connection, 1, "Eletrônicos Premium",
"<p>Produtos eletrônicos de alta qualidade</p>",
"eletronicos_premium.jpg")
    update_produto(connection, 1, "Smartphone X Pro", 1, "1:1",
"TechCorp", "Smartphone premium", 100, 2499.99, 2999.99)

```

```

    update_cliente(connection, 1, "José Paulo Pereira", "Pereira",
"José Paulo", "11999999999", "Rua A", "123", "São Paulo", "SP",
"01000000", "Brasil", 1, 6000.00)

    update_pedido(connection, 1, date(2024, 2, 1), date(2024, 2, 9),
date(2024, 2, 4), "Entregue", "Entrega antecipada", 1)
    update_detalhe_pedido(connection, 1, 1, 3, 2499.99, 1)
    update_pagamento(connection, 1, 1, date(2024, 2, 4), 7499.97)


    # Deletar registros na ordem correta (do mais dependente para o
menos dependente)
    # Primeiro, deletar pagamentos

delete_pagamento(connection, 1, 1)


    # Depois, deletar detalhes do pedido
delete_detalhe_pedido(connection, 1, 1)


    # Então, deletar o pedido
delete_pedido(connection, 1)


import mysql.connector
from mysql.connector import Error
from datetime import date

def create_connection():
    """Cria uma conexão com o banco de dados MySQL."""
    connection = None
    try:
        connection = mysql.connector.connect(
            host='192.168.0.18',
            port='3306',
            user='root',
            password='root',
            database='BD_LOJA'
        )
        print("Conexão com o MySQL bem-sucedida")
    except Error as e:
        print(f"Erro '{e}' ocorreu")
    return connection

# CRUD para TB_CLIENTES
def create_cliente(connection, nome, nome_ultimo, nome_primeiro,
telefone, endereco_pt1,

```

```

        endereco_pt2, cidade, estado, codigo_postal, pais,
funcionario_id, limite_credito):
    cursor = connection.cursor()
    query = """INSERT INTO TB_CLIENTES (nome, nome_ultimo,
nome_primeiro, telefone,
        endereco_pt1, endereco_pt2, cidade, estado,
codigo_postal, pais,
        funcionario_id, limite_credito)
        VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,
%s)"""
    values = (nome, nome_ultimo, nome_primeiro, telefone, endereco_pt1,
endereco_pt2,
        cidade, estado, codigo_postal, pais, funcionario_id,
limite_credito)
    cursor.execute(query, values)
    connection.commit()
    print("Cliente adicionado com sucesso")

def read_clientes(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_CLIENTES")
    return cursor.fetchall()

def update_cliente(connection, id, nome, nome_ultimo, nome_primeiro,
telefone, endereco_pt1,
        endereco_pt2, cidade, estado, codigo_postal, pais,
funcionario_id, limite_credito):
    cursor = connection.cursor()
    query = """UPDATE TB_CLIENTES SET nome=%s, nome_ultimo=%s,
nome_primeiro=%s,
        telefone=%s, endereco_pt1=%s, endereco_pt2=%s,
cidade=%s, estado=%s,
        codigo_postal=%s, pais=%s, funcionario_id=%s,
limite_credito=%s WHERE id=%s"""
    values = (nome, nome_ultimo, nome_primeiro, telefone, endereco_pt1,
endereco_pt2,
        cidade, estado, codigo_postal, pais, funcionario_id,
limite_credito, id)
    cursor.execute(query, values)
    connection.commit()
    print("Cliente atualizado com sucesso")

def delete_cliente(connection, id):

```

```

        cursor = connection.cursor()
        cursor.execute("DELETE FROM TB_CLIENTES WHERE id=%s", (id,))
        connection.commit()
        print("Cliente deletado com sucesso")

# CRUD para TB_PRODUTOS
def create_produto(connection, nome, linha_produto_id, escala,
fornecedor, descricao,
                    quantidade_estoque, preco, msrp):
    cursor = connection.cursor()
    query = """INSERT INTO TB_PRODUTOS (nome, linha_produto_id, escala,
fornecedor,
                    descricao, quantidade_estoque, preco, msrp)
                    VALUES (%s, %s, %s, %s, %s, %s, %s, %s)"""
    values = (nome, linha_produto_id, escala, fornecedor, descricao,
                    quantidade_estoque, preco, msrp)
    cursor.execute(query, values)
    connection.commit()
    print("Produto adicionado com sucesso")

def read_produtos(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_PRODUTOS")
    return cursor.fetchall()

def update_produto(connection, id, nome, linha_produto_id, escala,
fornecedor,
                    descricao, quantidade_estoque, preco, msrp):
    cursor = connection.cursor()
    query = """UPDATE TB_PRODUTOS SET nome=%s, linha_produto_id=%s,
escala=%s,
                    fornecedor=%s, descricao=%s, quantidade_estoque=%s,
preco=%s, msrp=%s
                    WHERE id=%s"""
    values = (nome, linha_produto_id, escala, fornecedor, descricao,
                    quantidade_estoque, preco, msrp, id)
    cursor.execute(query, values)
    connection.commit()
    print("Produto atualizado com sucesso")

def delete_produto(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_PRODUTOS WHERE id=%s", (id,))

```



```

        connection.commit()
        print("Produto deletado com sucesso")

# CRUD para TB_PEDIDOS
def create_pedido(connection, data_pedido, data_entrega, data_envio,
status,
                comentarios, cliente_id):
    cursor = connection.cursor()
    query = """INSERT INTO TB_PEDIDOS (data_pedido, data_entrega,
data_envio,
                status, comentarios, cliente_id)
                VALUES (%s, %s, %s, %s, %s, %s)"""
    values = (data_pedido, data_entrega, data_envio, status,
comentarios, cliente_id)
    cursor.execute(query, values)
    connection.commit()
    print("Pedido adicionado com sucesso")

def read_pedidos(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_PEDIDOS")
    return cursor.fetchall()

def update_pedido(connection, id, data_pedido, data_entrega,
data_envio,
                status, comentarios, cliente_id):
    cursor = connection.cursor()
    query = """UPDATE TB_PEDIDOS SET data_pedido=%s, data_entrega=%s,
data_envio=%s,
                status=%s, comentarios=%s, cliente_id=%s WHERE id=%s"""
    values = (data_pedido, data_entrega, data_envio, status,
comentarios,
                cliente_id, id)
    cursor.execute(query, values)
    connection.commit()
    print("Pedido atualizado com sucesso")

def delete_pedido(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_PEDIDOS WHERE id=%s", (id,))
    connection.commit()
    print("Pedido deletado com sucesso")

```

```

# CRUD para TB_DETALHES_PEDIDO

def create_detalhe_pedido(connection, pedido_id, produto_id,
quantidade_pedida,
                                preco_unitario, numero_linha_pedido):
    cursor = connection.cursor()
    query = """INSERT INTO TB_DETALHES_PEDIDO (pedido_id, produto_id,
                                quantidade_pedida, preco_unitario, numero_linha_pedido)
                                VALUES (%s, %s, %s, %s, %s)"""
    values = (pedido_id, produto_id, quantidade_pedida, preco_unitario,
                                numero_linha_pedido)
    cursor.execute(query, values)
    connection.commit()
    print("Detalhe do pedido adicionado com sucesso")

def read_detalhes_pedido(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_DETALHES_PEDIDO")
    return cursor.fetchall()

def update_detalhe_pedido(connection, pedido_id, produto_id,
quantidade_pedida,
                                preco_unitario, numero_linha_pedido):
    cursor = connection.cursor()
    query = """UPDATE TB_DETALHES_PEDIDO SET quantidade_pedida=%s,
                                preco_unitario=%s, numero_linha_pedido=%s
                                WHERE pedido_id=%s AND produto_id=%s"""
    values = (quantidade_pedida, preco_unitario, numero_linha_pedido,
                                pedido_id, produto_id)
    cursor.execute(query, values)
    connection.commit()
    print("Detalhe do pedido atualizado com sucesso")

def delete_detalhe_pedido(connection, pedido_id, produto_id):
    cursor = connection.cursor()
    cursor.execute("""DELETE FROM TB_DETALHES_PEDIDO
                                WHERE pedido_id=%s AND produto_id=%s""",
(pedido_id, produto_id))
    connection.commit()
    print("Detalhe do pedido deletado com sucesso")

# CRUD para TB_PAGAMENTOS

def create_pagamento(connection, cliente_id, data_pagamento, valor):
    cursor = connection.cursor()

```

```

        query = """INSERT INTO TB_PAGAMENTOS (cliente_id, data_pagamento,
valor)
                VALUES (%s, %s, %s)"""
        cursor.execute(query, (cliente_id, data_pagamento, valor))
        connection.commit()
        print("Pagamento adicionado com sucesso")

def read_pagamentos(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_PAGAMENTOS")
    return cursor.fetchall()

def update_pagamento(connection, id, cliente_id, data_pagamento,
valor):
    cursor = connection.cursor()
    query = """UPDATE TB_PAGAMENTOS SET data_pagamento=%s, valor=%s
                WHERE id=%s AND cliente_id=%s"""
    cursor.execute(query, (data_pagamento, valor, id, cliente_id))
    connection.commit()
    print("Pagamento atualizado com sucesso")

def delete_pagamento(connection, id, cliente_id):
    cursor = connection.cursor()
    cursor.execute("""DELETE FROM TB_PAGAMENTOS
                WHERE id=%s AND cliente_id=%s""", (id,
cliente_id))
    connection.commit()
    print("Pagamento deletado com sucesso")

# CRUD para TB_FUNCIONARIOS
def create_funcionario(connection, nome_ultimo, nome_primeiro,
extensao, email,
                escritorio_id, relatorios_para_funcionario_id,
trabalho):
    cursor = connection.cursor()
    query = """INSERT INTO TB_FUNCIONARIOS (nome_ultimo, nome_primeiro,
extensao,
                email, escritorio_id, relatorios_para_funcionario_id,
trabalho)
                VALUES (%s, %s, %s, %s, %s, %s, %s)"""
    values = (nome_ultimo, nome_primeiro, extensao, email,
escritorio_id,
                relatorios_para_funcionario_id, trabalho)

```

```

        cursor.execute(query, values)
        connection.commit()
        print("Funcionário adicionado com sucesso")

def read_funcionarios(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_FUNCIONARIOS")
    return cursor.fetchall()

def update_funcionario(connection, id, nome_ultimo, nome_primeiro,
                        extensao,
                        email, escritorio_id,
                        relatorios_para_funcionario_id, trabalho):
    cursor = connection.cursor()
    query = """UPDATE TB_FUNCIONARIOS SET nome_ultimo=%s,
nome_primeiro=%s,
                        extensao=%s, email=%s, escritorio_id=%s,
                        relatorios_para_funcionario_id=%s, trabalho=%s WHERE
id=%s"""
    values = (nome_ultimo, nome_primeiro, extensao, email,
escritorio_id,
                        relatorios_para_funcionario_id, trabalho, id)
    cursor.execute(query, values)
    connection.commit()
    print("Funcionário atualizado com sucesso")

def delete_funcionario(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_FUNCIONARIOS WHERE id=%s", (id,))
    connection.commit()
    print("Funcionário deletado com sucesso")

# CRUD para TB_ESCRITORIOS
def create_escritorio(connection, cidade, phone, endereco_pt1,
                      endereco_pt2,
                      estado, pais, codigo_postal, territorio):
    cursor = connection.cursor()
    query = """INSERT INTO TB_ESCRITORIOS (cidade, phone, endereco_pt1,
                        endereco_pt2, estado, pais, codigo_postal, territorio)
VALUES (%s, %s, %s, %s, %s, %s, %s, %s)"""
    values = (cidade, phone, endereco_pt1, endereco_pt2, estado, pais,
codigo_postal, territorio)
    cursor.execute(query, values)

```

```

        connection.commit()
        print("Escritório adicionado com sucesso")

def read_escritorios(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_ESCRITORIOS")
    return cursor.fetchall()

def update_escritorio(connection, id, cidade, phone, endereco_pt1,
endereco_pt2,
                        estado, pais, codigo_postal, territorio):
    cursor = connection.cursor()
    query = """UPDATE TB_ESCRITORIOS SET cidade=%s, phone=%s,
endereco_pt1=%s,
                        endereco_pt2=%s, estado=%s, pais=%s, codigo_postal=%s,
                        territorio=%s WHERE id=%s"""
    values = (cidade, phone, endereco_pt1, endereco_pt2, estado, pais,
              codigo_postal, territorio, id)
    cursor.execute(query, values)
    connection.commit()
    print("Escritório atualizado com sucesso")

def delete_escritorio(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_ESCRITORIOS WHERE id=%s", (id,))
    connection.commit()
    print("Escritório deletado com sucesso")

# CRUD para TB_LINHAS_PRODUTOS
def create_linha_produto(connection, descricao, descricao_html, image):
    cursor = connection.cursor()
    query = """INSERT INTO TB_LINHAS_PRODUTOS (descricao,
descricao_html, image)
                        VALUES (%s, %s, %s)"""
    cursor.execute(query, (descricao, descricao_html, image))
    connection.commit()
    print("Linha de produto adicionada com sucesso")

def read_linhas_produtos(connection):
    cursor = connection.cursor()
    cursor.execute("SELECT * FROM TB_LINHAS_PRODUTOS")
    return cursor.fetchall()

```

```

def update_linha_produto(connection, id, descricao, descricao_html,
image):
    cursor = connection.cursor()
    query = """UPDATE TB_LINHAS_PRODUTOS SET descricao=%s,
descricao_html=%s,
                image=%s WHERE id=%s"""
    cursor.execute(query, (descricao, descricao_html, image, id))
    connection.commit()
    print("Linha de produto atualizada com sucesso")

def delete_linha_produto(connection, id):
    cursor = connection.cursor()
    cursor.execute("DELETE FROM TB_LINHAS_PRODUTOS WHERE id=%s", (id,))
    connection.commit()
    print("Linha de produto deletada com sucesso")

def main():
    connection = create_connection()
    if connection is None:
        return

    #VOCÊ SÓ ADICIONA ESSAS INSTÂNCIAS SE QUISER TESTAR O
    #FUNCIONAMENTO!!! RETIRE, SE NÃO FOR NECESSÁRIO.
    #Criando algumas instâncias:

    # Deletar registros na ordem correta (do mais dependente para o
    #menos dependente)
    # Primeiro, deletar pagamentos

    delete_pagamento(connection, 1, 1)

    # Depois, deletar detalhes do pedido
    delete_detalhe_pedido(connection, 1, 1)

    # Então, deletar o pedido
    delete_pedido(connection, 1)

    # Agora podemos deletar o cliente
    delete_cliente(connection, 1)

    # Por fim, deletar o produto que não está mais referenciado
    delete_produto(connection, 1)

```

```
delete_escritorio(connection, 1)

delete_linha_produto(connection, 1)

connection.close()

if __name__ == "__main__":
    main()


# Agora podemos deletar o cliente
delete_cliente(connection, 1)

# Por fim, deletar o produto que não está mais referenciado
delete_produto(connection, 1)

delete_funcionario(connection, 1)

delete_escritorio(connection, 1)

delete_linha_produto(connection, 1)

connection.close()

if __name__ == "__main__":
    main()
```

>>**Execute** o arquivo que você criou:

```
python app.py
```

>>É isso! :)