

EXERCÍCIO 2

ATIVIDADE REALIZADA POR: LARA YSLEN GOMES BARBOSA

RODANDO O CONTAINER

```
[node1] (local) root@192.168.0.63 ~
$ docker run -d -p 3306:3306 --name meu-mysql \
>
"docker run" requires at least 1 argument.
See 'docker run --help'.

Usage:  docker run [OPTIONS] IMAGE [COMMAND] [ARG...]

Create and run a new container from an image
[node1] (local) root@192.168.0.63 ~
$ docker run -d -p 3306:3306 --name meu-mysql \
> -e MYSQL_ROOT_PASSWORD=MinhaSenha \
> -e MYSQL_DATABASE=BD_VENDAS \
> -e MYSQL_USER=seu_usuario \
> -e MYSQL_PASSWORD=sua_senha \
> mysql:latest
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
90dac1e734aa: Pull complete
f1d28d66c159: Pull complete
f6e58dad121f: Pull complete
f3e3a6933245: Pull complete

> -e MYSQL_ROOT_PASSWORD=MinhaSenha \
> -e MYSQL_DATABASE=BD_VENDAS \
> -e MYSQL_USER=seu_usuario \
> -e MYSQL_PASSWORD=sua_senha \
> mysql:latest
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
90dac1e734aa: Pull complete
f1d28d66c159: Pull complete
f6e58dad121f: Pull complete
f3e3a6933245: Pull complete
ccb04e1eb0a7: Pull complete
8d917159a675: Pull complete
f4b7ec9de513: Pull complete
cb6b011730e2: Pull complete
6be37253424c: Pull complete
237611fb5faf: Pull complete
Digest: sha256:297f5ead7043a440ce84b3b0f3b77430a4f233c2578ff15fff8f0de54f67f22d
Status: Downloaded newer image for mysql:latest
8e076fc245071ab59808cae7bb64759564d137e0fe351a18f22476622f2d5943
[node1] (local) root@192.168.0.63 ~
$
```

INSTALAÇÃO DO PYTHON

```
[node2] (local) root@192.168.0.67 ~
$ python3 -m venv venv
source venv/bin/activate
pip install mysql-connector-python

Collecting mysql-connector-python
  Downloading mysql_connector_python-9.3.0-py2.py3-none-any.whl.metadata (7.3 kB)
Downloading mysql_connector_python-9.3.0-py2.py3-none-any.whl (399 kB)
Installing collected packages: mysql-connector-python
Successfully installed mysql-connector-python-9.3.0

[notice] A new release of pip is available: 24.2 -> 25.1.1
[notice] To update, run: pip install --upgrade pip
```

[SCRIPT.PY](#) USADO:

```
import mysql.connector

from mysql.connector import Error

from decimal import Decimal


# Função para criar conexão com o banco BD_VENDAS
def criar_conexao():

    try:

        conexao = mysql.connector.connect(

            host='192.168.0.68', # IP do PlaywithDocker

            port=3306,          # Porta do MySQL no PlaywithDocker

            user='seu_usuario', # seu usuário MySQL

            password='sua_senha', # sua senha MySQL

            database='BD_VENDAS'

        )

        if conexao.is_connected():

            print("Conexão ao BD_VENDAS realizada com sucesso!")

            return conexao

        except Error as e:

            print(f"Erro ao conectar ao MySQL: {e}")

            return None

import mysql.connector

from mysql.connector import Error

from decimal import Decimal


# Função para criar conexão com o banco BD_VENDAS
```

```

def criar_conexao():
    try:

        conexao = mysql.connector.connect(

            host='192.168.0.68', # IP do PlaywithDocker

            port=3306,          # Porta do MySQL no PlaywithDocker

            user='seu_usuario', # seu usuário MySQL

            password='sua_senha', # sua senha MySQL

            database='BD_VENDAS'

        )

        if conexao.is_connected():

            print("Conexão ao BD_VENDAS realizada com sucesso!")

        return conexao

    except Error as e:

        print(f"Erro ao conectar ao MySQL: {e}")

        return None

```

Funções CRUD para TB_Produtos

```

def inserir_produto(conexao, descricao, valor_unitario, qtd_estoque):

    cursor = conexao.cursor()

    # Converter para Decimal para evitar erro de tipos

    valor_unitario_dec = Decimal(str(valor_unitario))

    qtd_estoque_dec = Decimal(str(qtd_estoque))

    valor_estoque = valor_unitario_dec * qtd_estoque_dec

    sql = """

    INSERT INTO TB_Produtos (descricao, valor_unitario, qtd_estoque, valor_estoque)

    VALUES (%s, %s, %s, %s)

```

```
"""
```

```
try:
```

```
    cursor.execute(sql, (descricao, valor_unitario_dec, qtd_estoque_dec, valor_estoque))
```

```
    conexao.commit()
```

```
    print(f"Produto '{descricao}' inserido com sucesso.")
```

```
except Error as e:
```

```
    print(f"Erro ao inserir produto: {e}")
```

```
finally:
```

```
    cursor.close()
```

```
def selecionar_produtos(conexao):
```

```
    cursor = conexao.cursor(dictionary=True)
```

```
    sql = "SELECT * FROM TB_Produtos"
```

```
    try:
```

```
        cursor.execute(sql)
```

```
        resultados = cursor.fetchall()
```

```
        return resultados
```

```
    except Error as e:
```

```
        print(f"Erro ao selecionar produtos: {e}")
```

```
    return []
```

```
    finally:
```

```
        cursor.close()
```

```
def atualizar_produto(conexao, id_produto, descricao=None, valor_unitario=None, qtd_estoque=None):
```

```
    cursor = conexao.cursor()
```

```
    sql_select = "SELECT valor_unitario, qtd_estoque FROM TB_Produtos WHERE id = %s"
```

```
    try:
```

```

cursor.execute(sql_select, (id_produto,))

produto = cursor.fetchone()

if not produto:

    print("Produto não encontrado.")

    return


# Converter valores atuais para Decimal

valor_unitario_atual = Decimal(str(produto[0]))

qtd_estoque_atual = Decimal(str(produto[1]))


novo_valor_unitario = Decimal(str(valor_unitario)) if valor_unitario is not None else
valor_unitario_atual

nova_qtd_estoque = Decimal(str(qtd_estoque)) if qtd_estoque is not None else
qtd_estoque_atual

novo_valor_estoque = novo_valor_unitario * nova_qtd_estoque
finally:

    cursor.close()

```

Funções CRUD para TB_FORNECEDOR

```

def inserir_fornecedor(conexao, nome):

    cursor = conexao.cursor()

    sql = "INSERT INTO TB_FORNECEDOR (nome) VALUES (%s)"

    try:

        cursor.execute(sql, (nome,))

        conexao.commit()

        print(f"Fornecedor '{nome}' inserido com sucesso.")

    except Error as e:

```

```
print(f"Erro ao inserir fornecedor: {e}")
```

```
finally:
```

```
    cursor.close()
```

```
def selecionar_fornecedores(conexao):
```

```
    cursor = conexao.cursor(dictionary=True)
```

```
    sql = "SELECT * FROM TB_FORNECEDOR"
```

```
    try:
```

```
        cursor.execute(sql)
```

```
        resultados = cursor.fetchall()
```

```
        return resultados
```

```
    except Error as e:
```

```
        print(f"Erro ao selecionar fornecedores: {e}")
```

```
        return []
```

```
    finally:
```

```
        cursor.close()
```

```
def atualizar_fornecedor(conexao, id_fornecedor, nome):
```

```
    campos = []
```

```
    valores = []
```

```
    if descricao is not None:
```

```
        campos.append("descricao = %s")
```

```
        valores.append(descricao)
```

```
    if valor_unitario is not None:
```

```
        campos.append("valor_unitario = %s")
```

```
valores.append(novo_valor_unitario)
```

```
if qtd_estoque is not None:
```

```
    campos.append("qtd_estoque = %s")
```

```
    valores.append(nova_qtd_estoque)
```

```
campos.append("valor_estoque = %s")
```

```
valores.append(novo_valor_estoque)
```

```
valores.append(id_produto)
```

```
    sql_update = f"UPDATE TB_Produtos SET {' '.join(campos)} WHERE id = %s"
```

```
    cursor.execute(sql_update, tuple(valores))
```

```
    conexao.commit()
```

```
    print(f"Produto ID {id_produto} atualizado com sucesso.")
```

```
    except Error as e:
```

```
        print(f"Erro ao atualizar produto: {e}")
```

```
    finally:
```

```
        cursor.close()
```

```
def deletar_produto(conexao, id_produto):
```

```
    cursor = conexao.cursor()
```

```
    sql = "DELETE FROM TB_Produtos WHERE id = %s"
```

```
    try:
```

```
        cursor.execute(sql, (id_produto,))
```

```
        conexao.commit()
```

```
        print(f"Produto ID {id_produto} deletado com sucesso.")
```

```

        except Error as e:

print(f"Erro ao deletar produto: {e}")

        valores.append(descricao)

        if valor_unitario is not None:

            campos.append("valor_unitario = %s")

            valores.append(novo_valor_unitario)

            if qtd_estoque is not None:

                campos.append("qtd_estoque = %s")

                valores.append(nova_qtd_estoque)


        campos.append("valor_estoque = %s")

        valores.append(novo_valor_estoque)


        valores.append(id_produto)


        sql_update = f"UPDATE TB_Produtos SET {' '.join(campos)} WHERE id = %s"

        cursor.execute(sql_update, tuple(valores))

        conexao.commit()

        print(f"Produto ID {id_produto} atualizado com sucesso.")

        except Error as e:

            print(f"Erro ao atualizar produto: {e}")

        finally:

            cursor.close()


def deletar_produto(conexao, id_produto):

    cursor = conexao.cursor()

```



```

    sql = "DELETE FROM TB_Produtos WHERE id = %s"

    try:

        cursor.execute(sql, (id_produto,))

        conexao.commit()

        print(f"Produto ID {id_produto} deletado com sucesso.")

    except Error as e:

        print(f"Erro ao deletar produto: {e}")
cursor = conexao.cursor()

    sql = "UPDATE TB_FORNECEDOR SET nome = %s WHERE id = %s"

    try:

        cursor.execute(sql, (nome, id_fornecedor))

        conexao.commit()

        print(f"Fornecedor ID {id_fornecedor} atualizado com sucesso.")

    except Error as e:

        print(f"Erro ao atualizar fornecedor: {e}")

    finally:

        cursor.close()

def deletar_fornecedor(conexao, id_fornecedor):

    cursor = conexao.cursor()

    sql = "DELETE FROM TB_FORNECEDOR WHERE id = %s"

    try:

        cursor.execute(sql, (id_fornecedor,))

        conexao.commit()

        print(f"Fornecedor ID {id_fornecedor} deletado com sucesso.")

    except Error as e:

        print(f"Erro ao deletar fornecedor: {e}")

```

finally:

cursor.close()

Função para calcular o valor total do estoque (soma de valor_estoque de todos os produtos)

def calcular_valor_total_estoque(conexao):

 cursor = conexao.cursor()

 sql = "SELECT SUM(valor_estoque) FROM TB_Produtos"

 try:

 cursor.execute(sql)

 resultado = cursor.fetchone()

 valor_total = resultado[0] if resultado[0] is not None else Decimal('0.00')

 print(f"Valor total do estoque: R\$ {valor_total:.2f}")

 return valor_total

 except Error as e:

 print(f"Erro ao calcular valor total do estoque: {e}")

 return Decimal('0.00')

 finally:

 cursor.close()

Exemplo de uso

if __name__ == "__main__":

 conexao = criar_conexao()

 if conexao:

 criar_tabelas(conexao)

 # Inserir exemplos

```
inserir_produto(conexao, "Teclado Mecânico", 250.00, 10)
```

```
inserir_produto(conexao, "Mouse Gamer", 150.00, 20)
```

```
inserir_fornecedor(conexao, "Fornecedor A")
```

```
inserir_fornecedor(conexao, "Fornecedor B")
```

```
# Listar produtos
```

```
produtos = selecionar_produtos(conexao)
```

```
print("Produtos cadastrados:")
```

```
for p in produtos:
```

```
    print(p)
```

```
# Atualizar produto
```

```
atualizar_produto(conexao, 1, valor_unitario=260.00)
```

```
# Calcular valor total do estoque
```

```
calcular_valor_total_estoque(conexao)
```

```
# Fechar conexão
```

```
conexao.close()
```

RODANDO O [SCRIPT.PY](#):

```
[venv] [node2] (local) root@192.168.0.67 ~
$ python3 meu_script.py
Conexão ao BD VENDAS realizada com sucesso!
Tabelas criadas com sucesso!
Produto 'Teclado Mecânico' inserido com sucesso.
Produto 'Mouse Gamer' inserido com sucesso.
Fornecedor 'Fornecedor A' inserido com sucesso.
Fornecedor 'Fornecedor B' inserido com sucesso.
Produtos cadastrados:
{'id': 1, 'descricao': 'Teclado Mecânico', 'valor_unitario': Decimal('250.00'), 'qtd_estoque': Decimal('10.00'), 'valor_estoque': Decimal('2500.00')}
{'id': 2, 'descricao': 'Mouse Gamer', 'valor_unitario': Decimal('150.00'), 'qtd_estoque': Decimal('20.00'), 'valor_estoque': Decimal('3000.00')}
{'id': 3, 'descricao': 'Teclado Mecânico', 'valor_unitario': Decimal('250.00'), 'qtd_estoque': Decimal('10.00'), 'valor_estoque': Decimal('2500.00')}
{'id': 4, 'descricao': 'Mouse Gamer', 'valor_unitario': Decimal('150.00'), 'qtd_estoque': Decimal('20.00'), 'valor_estoque': Decimal('3000.00')}
Produto ID 1 atualizado com sucesso.
Valor total do estoque: R$ 11100.00
```

