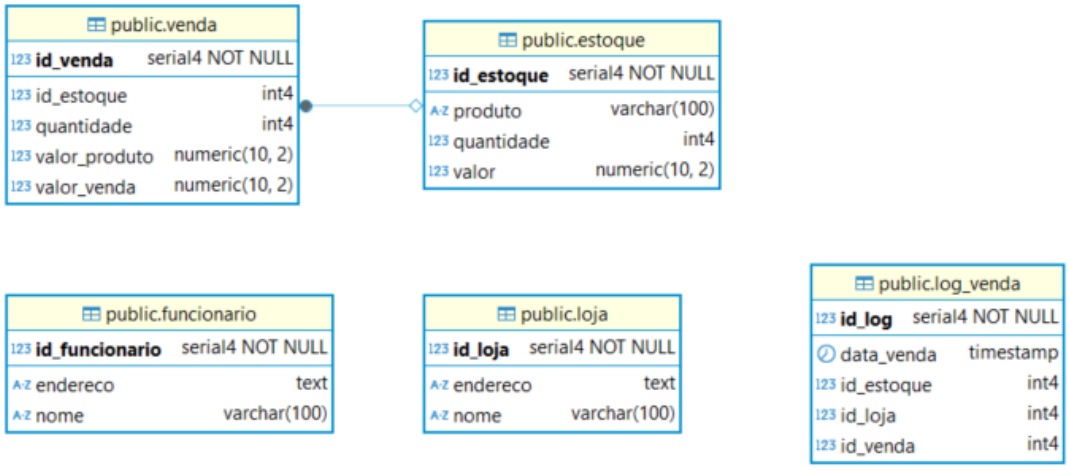
Exercício de Banco de Dados Relacional da P2

1 Crie o banco de dados



Resposta:

2 Realize a Trigger

-- View de vendas realizadas

CREATE OR REPLACE VIEW vendas_realizadas AS

SELECT vendas de vendas, produto de estoque, quantidade de vendas, valor do produto de produtos, valor de venda em vendas

FROM chamar a tabela vendas

JOIN chamar a a tabela de estoque ON configurar as colunas que liga o join;

3 Trigger de log_vendas

-- Trigger para log de vendas

CREATE OR REPLACE FUNCTION nome da função()

RETURNS TRIGGER AS $$

BEGIN

  INSERT INTO log_venda (colunas da tabela)

  VALUES (formato para todas as colunas “NEW.coluna”);

  RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER nome da trigger

AFTER INSERT ON tabela que a trigger vai buscar o dado

FOR EACH ROW

EXECUTE FUNCTION chamar o mesmo nome da trigger anteriror();

Resposta:

4 Crie um usuário e de prioridade de edição para ele e mostre o processo

Resposta:

4 tkinter para criação do serviço:

```
import tkinter as tk
from tkinter import messagebox
import psycopg2

# Conexão com PostgreSQL
conn = psycopg2.connect(
   dbname="db_name", user="user", password="psw", host="host"
)
cur = conn.cursor()

# Funções auxiliares para simular placeholder
def placeholder(entry, texto):
   entry.insert(0, texto)
   entry.bind("<FocusIn>", lambda event: clear_placeholder(event, texto))
   entry.bind("<FocusOut>", lambda event: restore_placeholder(event, texto))

def clear_placeholder(event, texto):
   if event.widget.get() == texto:
      event.widget.delete(0, "end")
```

```
def restore_placeholder(event, texto):
    if event.widget.get() == "":
        event.widget.insert(0, texto)

# Função de cadastro
def cadastrar():
    try:
        nome_loja = entry_loja.get()
        nome_func = entry_func.get()
        produto = entry_produto.get()
        valor = float(entry_valor.get())
        quantidade = int(entry_quant.get())

        # Criação de loja
        cur.execute("INSERT INTO loja (colunas) VALUES (%s, %s) RETURNING id_loja",
(colunas"))
        id_loja = cur.fetchone()[0]

        # Criação de funcionário
        cur.execute("INSERT INTO funcionario (colunas) VALUES (%s, %s)", (nome_func,
"endereco"))

        # Criação de estoque
        cur.execute("INSERT INTO estoque (coluans de estoque) VALUES (%s, %s, %s)
RETURNING id_estoque",
                    (produto, valor, quantidade))
        id_estoque = cur.fetchone()[0]

        # Criação de venda (a trigger cuidará do log)
        cur.execute("""
            INSERT INTO venda ()
            VALUES (%s, %s, %s, %s, %s)
```

```
        """, (id_loja, id_estoque, quantidade, valor, valor * quantidade))

        conn.commit()
        messagebox.showinfo("Cadastro", "Dados inseridos com sucesso!")

    except Exception as e:
        conn.rollback()
        messagebox.showerror("Erro", f"Ocorreu um erro: {e}")

# Função para exibir catálogo
def ver_catalogo():
    cur.execute("SELECT * FROM estoque")
    dados = cur.fetchall()
    texto = "\n".join([f"Produto: {x[1]}, Preço: R${x[2]}, Quantidade: {x[3]}" for x in dados])
    messagebox.showinfo("Catálogo", texto)

# Função para exibir gestão de vendas
def ver_gestao():
    cur.execute("SELECT * FROM vendas_realizadas")
    dados = cur.fetchall()
    texto = "\n".join([f"Venda {x[0]} - {x[1]}: {x[2]} x R${x[3]} = R${x[4]}" for x in dados])
    messagebox.showinfo("Gestão de Vendas", texto)

# Interface gráfica
root = tk.Tk()
root.title("Cadastro de Loja e Vendas")

entry_loja = tk.Entry(root); entry_loja.pack()
placeholder(entry_loja, "Nome da Loja")

entry_func = tk.Entry(root); entry_func.pack()
```

```
placeholder(entry_func, "Nome do Funcionário")

entry_produto = tk.Entry(root); entry_produto.pack()
placeholder(entry_produto, "Produto")

entry_valor = tk.Entry(root); entry_valor.pack()
placeholder(entry_valor, "Valor")

entry_quant = tk.Entry(root); entry_quant.pack()
placeholder(entry_quant, "Quantidade")

btn_cadastrar = tk.Button(root, text="Cadastrar", command=cadastrar)
btn_cadastrar.pack()

btn_cat = tk.Button(root, text="Ver Catálogo", command=ver_catalogo)
btn_cat.pack()

btn_gestao = tk.Button(root, text="Ver Gestão", command=ver_gestao)
btn_gestao.pack()

root.mainloop()
```

Resposta: