import tkinter as tk

from tkinter import messagebox, ttk

import mysql.connector

def connectar():

    return mysql.connector.connect(

        host="localhost",

        user="root",

        password="",

        database="agenda"

    )

*# Criação da tabela*

def criar\_tabela():

    conexao = connectar()

    cursor = conexao.cursor()

    cursor.execute('''

        CREATE TABLE IF NOT EXISTS musica (

            id INT AUTO\_INCREMENT PRIMARY KEY,

            titulo VARCHAR(255),

            artista VARCHAR(255),

            palavras\_chave TEXT,

            album VARCHAR(255),

            genero VARCHAR(100),

            ano INT,

            duracao\_segundos INT,

            compositor VARCHAR(255),

            gravadora VARCHAR(255),

            caminho\_arquivo TEXT

        )

    ''')

    conexao.commit()

    conexao.close()

*# Inserir música*

def inserir\_musica\_bd(titulo, artista, palavras, album, genero, ano, duracao, compositor, gravadora, caminho):

    conexao = connectar()

    cursor = conexao.cursor()

    cursor.execute('''

        INSERT INTO musica (titulo, artista, palavras\_chave, album, genero, ano, duracao\_segundos, compositor, gravadora, caminho\_arquivo)

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

    ''', (titulo, artista, palavras, album, genero, int(ano), int(duracao), compositor, gravadora, caminho))

    conexao.commit()

    conexao.close()

*# Buscar todas*

def buscar\_musicas():

    conexao = connectar()

    cursor = conexao.cursor()

    cursor.execute('SELECT \* FROM musica')

    musicas = cursor.fetchall()

    conexao.close()

    return musicas

*# Deletar música*

def deletar\_musica\_bd(id\_musica):

    conexao = connectar()

    cursor = conexao.cursor()

    cursor.execute('DELETE FROM musica WHERE id = %s', (id\_musica,))

    conexao.commit()

    conexao.close()

*# Atualizar música*

def atualizar\_musica\_bd(id\_musica, titulo, artista, palavras, album, genero, ano, duracao, compositor, gravadora, caminho):

    conexao = connectar()

    cursor = conexao.cursor()

    cursor.execute('''

        UPDATE musica

        SET titulo = %s, artista = %s, palavras\_chave = %s, album = %s, genero = %s, ano = %s,

            duracao\_segundos = %s, compositor = %s, gravadora = %s, caminho\_arquivo = %s

        WHERE id = %s

    ''', (titulo, artista, palavras, album, genero, int(ano), int(duracao), compositor, gravadora, caminho, id\_musica))

    conexao.commit()

    conexao.close()

*# Tela para adicionar nova música*

def abrir\_janela\_adicionar():

    def salvar\_musica():

        try:

            inserir\_musica\_bd(

                campo\_titulo.get(),

                campo\_artista.get(),

                campo\_palavras.get(),

                campo\_album.get(),

                campo\_genero.get(),

                campo\_ano.get(),

                campo\_duracao.get(),

                campo\_compositor.get(),

                campo\_gravadora.get(),

                campo\_caminho.get()

            )

            messagebox.showinfo("Sucesso", "Música adicionada com sucesso!")

            janela.destroy()

            atualizar\_tabela()

        except Exception as erro:

            messagebox.showerror("Erro", str(erro))

    janela = tk.Toplevel(janela\_principal)

    janela.title("Nova Música")

    labels = ["Título", "Artista", "Palavras-chave", "Álbum", "Gênero", "Ano", "Duração (seg)", "Compositor", "Gravadora", "Caminho"]

    entradas = []

    for i, texto in enumerate(labels):

        tk.Label(janela, text=texto).grid(row=i, column=0, sticky="w")

    campo\_titulo = tk.Entry(janela)

    campo\_artista = tk.Entry(janela)

    campo\_palavras = tk.Entry(janela)

    campo\_album = tk.Entry(janela)

    campo\_genero = tk.Entry(janela)

    campo\_ano = tk.Entry(janela)

    campo\_duracao = tk.Entry(janela)

    campo\_compositor = tk.Entry(janela)

    campo\_gravadora = tk.Entry(janela)

    campo\_caminho = tk.Entry(janela)

    entradas = [campo\_titulo, campo\_artista, campo\_palavras, campo\_album, campo\_genero,

                campo\_ano, campo\_duracao, campo\_compositor, campo\_gravadora, campo\_caminho]

    for i, entrada in enumerate(entradas):

        entrada.grid(row=i, column=1)

    tk.Button(janela, text="Salvar", command=salvar\_musica).grid(row=len(labels), columnspan=2, pady=10)

*# Atualizar tabela principal*

def atualizar\_tabela():

    for item in tabela\_musicas.get\_children():

        tabela\_musicas.delete(item)

    for musica in buscar\_musicas():

        tabela\_musicas.insert("", "end", values=musica)

*# Deletar*

def deletar\_musica\_tabela():

    selecionado = tabela\_musicas.selection()

    if selecionado:

        id\_musica = tabela\_musicas.item(selecionado[0])['values'][0]

        deletar\_musica\_bd(id\_musica)

        atualizar\_tabela()

        messagebox.showinfo("Removido", "Música deletada.")

    else:

        messagebox.showwarning("Atenção", "Selecione uma música para deletar.")

*# Atualizar música selecionada*

def abrir\_janela\_editar():

    selecionado = tabela\_musicas.selection()

    if not selecionado:

        messagebox.showwarning("Atenção", "Selecione uma música.")

        return

    dados = tabela\_musicas.item(selecionado[0])['values']

    id\_musica = dados[0]

    def salvar\_alteracao():

        try:

            atualizar\_musica\_bd(

                id\_musica,

                campo\_titulo.get(),

                campo\_artista.get(),

                campo\_palavras.get(),

                campo\_album.get(),

                campo\_genero.get(),

                campo\_ano.get(),

                campo\_duracao.get(),

                campo\_compositor.get(),

                campo\_gravadora.get(),

                campo\_caminho.get()

            )

            messagebox.showinfo("Sucesso", "Música atualizada!")

            janela.destroy()

            atualizar\_tabela()

        except Exception as erro:

            messagebox.showerror("Erro", str(erro))

    janela = tk.Toplevel(janela\_principal)

    janela.title("Editar Música")

    labels = ["Título", "Artista", "Palavras-chave", "Álbum", "Gênero", "Ano", "Duração (seg)", "Compositor", "Gravadora", "Caminho"]

    valores = dados[1:]

    campos = []

    for i, label in enumerate(labels):

        tk.Label(janela, text=label).grid(row=i, column=0, sticky="w")

    campo\_titulo = tk.Entry(janela)

    campo\_artista = tk.Entry(janela)

    campo\_palavras = tk.Entry(janela)

    campo\_album = tk.Entry(janela)

    campo\_genero = tk.Entry(janela)

    campo\_ano = tk.Entry(janela)

    campo\_duracao = tk.Entry(janela)

    campo\_compositor = tk.Entry(janela)

    campo\_gravadora = tk.Entry(janela)

    campo\_caminho = tk.Entry(janela)

    entradas = [campo\_titulo, campo\_artista, campo\_palavras, campo\_album, campo\_genero,

                campo\_ano, campo\_duracao, campo\_compositor, campo\_gravadora, campo\_caminho]

    for i, (campo, valor) in enumerate(zip(entradas, valores)):

        campo.insert(0, valor)

        campo.grid(row=i, column=1)

    tk.Button(janela, text="Salvar Alterações", command=salvar\_alteracao).grid(row=len(labels), columnspan=2, pady=10)

*# Inicialização*

criar\_tabela()

janela\_principal = tk.Tk()

janela\_principal.title("Catálogo Musical")

janela\_principal.state('zoomed')  *# TELA CHEIA no Windows*

*# Frame principal que contém tudo*

frame\_conteudo = tk.Frame(janela\_principal)

frame\_conteudo.pack(fill="both", expand=True)

*# Frame lateral esquerdo (botões)*

frame\_botoes = tk.Frame(frame\_conteudo, padx=10, pady=10)

frame\_botoes.pack(side="left", fill="y")

*# Botões na vertical*

tk.Button(frame\_botoes, text="Adicionar Música", width=20, command=abrir\_janela\_adicionar).pack(pady=10)

tk.Button(frame\_botoes, text="Editar Música", width=20, command=abrir\_janela\_editar).pack(pady=10)

tk.Button(frame\_botoes, text="Deletar Música", width=20, command=deletar\_musica\_tabela).pack(pady=10)

tk.Button(frame\_botoes, text="Sair", width=20, command=janela\_principal.destroy).pack(pady=10)

*# Frame da tabela de músicas*

frame\_tabela = tk.Frame(frame\_conteudo)

frame\_tabela.pack(side="right", fill="both", expand=True)

*# Scrollbar vertical*

scrollbar\_y = tk.Scrollbar(frame\_tabela)

scrollbar\_y.pack(side="right", fill="y")

*# Scrollbar horizontal*

scrollbar\_x = tk.Scrollbar(frame\_tabela, orient="horizontal")

scrollbar\_x.pack(side="bottom", fill="x")

*# Tabela de músicas*

colunas = ('ID', 'Título', 'Artista', 'Palavras-chave', 'Álbum', 'Gênero', 'Ano', 'Duração', 'Compositor', 'Gravadora', 'Caminho')

tabela\_musicas = ttk.Treeview(frame\_tabela, columns=colunas, show='headings',

                              yscrollcommand=scrollbar\_y.set,

                              xscrollcommand=scrollbar\_x.set)

for coluna in colunas:

    tabela\_musicas.heading(coluna, text=coluna)

    tabela\_musicas.column(coluna, width=150, anchor="w")

tabela\_musicas.pack(fill="both", expand=True)

*# Conecta scrollbars*

scrollbar\_y.config(command=tabela\_musicas.yview)

scrollbar\_x.config(command=tabela\_musicas.xview)

*# Atualiza os dados*

atualizar\_tabela()

*# Inicia a interface*

janela\_principal.mainloop()