El programa que se describe, pretende la gestión de una librería y corresponde al primer entregable de la diplomatura en Python dictada durante diciembre de 2023 por el Ing. Juan Barreto.

Este programa utiliza Tkinter para crear una interfaz gráfica básica y SQLite para trabajar con una base de datos. A continuación, se presenta un resumen de lo que hace:

1. **Importaciones:**
   * Importa los módulos necesarios de Tkinter para la interfaz gráfica y de SQLite para la base de datos.
2. **Funciones:**
   * **limpiar\_treeview(tree)**: Borra todos los elementos de un widget Treeview, probablemente utilizado para limpiar la visualización de datos.
   * **cargar\_libros(tree)**: Carga información de libros desde la base de datos a un widget Treeview, un componente de visualización tabular en Tkinter.
   * **conexion()**: Establece la conexión con la base de datos SQLite llamada "base\_librero.db".
   * Funciones para crear tablas en la base de datos (**crear\_tabla\_libros()**, **crear\_tabla\_categorias()**, **crear\_tabla\_autores()**, **crear\_tabla\_editorial()**).
   * Funciones para agregar datos predefinidos a las tablas (**agregar\_generos\_literarios()**, **agregar\_autores()**, **agregar\_editoriales()**).
3. **Manejo de Errores:**
   * Utiliza bloques **try-except** para controlar errores en la creación de tablas y la inserción de datos predefinidos en la base de datos. Imprime mensajes de error si ocurre algún problema.

Los siguientes son los CRUD de los diferentes datos que maneja el programa.

**Funciones CRUD de Categorías:**

1. **buscar\_categoria(nombre\_categoria)**: Busca una categoría en la base de datos según el nombre proporcionado.
2. **buscar\_categorias()**: Recupera todas las categorías existentes en orden alfabético.
3. **guardar\_categoria(nombre\_categoria)**: Agrega una nueva categoría a la base de datos.
4. **guardar\_mod\_categoria(nombre, categoriamod)**: Actualiza el nombre de una categoría existente en la base de datos.
5. **nueva\_categoria()**: Abre una ventana para agregar una nueva categoría. Valida la entrada de datos y muestra mensajes de error si es necesario.
6. **modificar\_categoria()**: Abre una ventana para modificar el nombre de una categoría existente. Valida la entrada de datos y muestra mensajes de error si es necesario.
7. **borrar\_categoria(nombre)**: Elimina una categoría de la base de datos.
8. **eliminar\_categoria()**: Abre una ventana de confirmación para eliminar una categoría seleccionada. Muestra mensajes de éxito o error después de la operación.

.

### Funciones CRUD de Editoriales:

1. **buscar\_editorial(nombre\_editorial)**: Busca una editorial en la base de datos según el nombre proporcionado.
2. **buscar\_editoriales()**: Recupera todas las editoriales existentes en orden alfabético.
3. **guardar\_editorial(nombre\_editorial)**: Agrega una nueva editorial a la base de datos.
4. **guardar\_mod\_editorial(nombre, editorialmod)**: Actualiza el nombre de una editorial existente en la base de datos.
5. **nueva\_editorial()**: Abre una ventana para agregar una nueva editorial. Valida la entrada de datos y muestra mensajes de error si es necesario.
6. **modificar\_editorial()**: Abre una ventana para modificar el nombre de una editorial existente. Valida la entrada de datos y muestra mensajes de error si es necesario.
7. **borrar\_editorial(nombre)**: Elimina una editorial de la base de datos.
8. **eliminar\_editorial()**: Abre una ventana de confirmación para eliminar una editorial seleccionada. Muestra mensajes de éxito o error después de la operación.

### Funciones CRUD de Autores:

1. **buscar\_autor(nombre\_autor)**: Busca un autor en la base de datos según el nombre proporcionado.
2. **buscar\_autores()**: Recupera todos los autores existentes en orden alfabético.
3. **guardar\_autor(nombre\_autor)**: Agrega un nuevo autor a la base de datos.
4. **guardar\_mod\_autor(nombre, autormod)**: Actualiza el nombre de un autor existente en la base de datos.
5. **nuevo\_autor()**: Abre una ventana para agregar un nuevo autor. Valida la entrada de datos y muestra mensajes de error si es necesario.
6. **modificar\_autor()**: Abre una ventana para modificar el nombre de un autor existente. Valida la entrada de datos y muestra mensajes de error si es necesario.
7. **borrar\_autor(nombre)**: Elimina un autor de la base de datos.
8. **eliminar\_autor()**: Abre una ventana de confirmación para eliminar un autor seleccionado. Muestra mensajes de éxito o error después de la operación.

### ¡Funciones CRUD de Libros:

* **buscar\_libro(nombre\_libro)**: Busca un libro por su título en la base de datos y devuelve los detalles si se encuentra.
* **buscar\_libros()**: Obtiene todos los libros de la base de datos y los formatea para mostrarlos en una lista.
* **obtener\_id\_autor(autor), obtener\_id\_editorial(editorial), obtener\_id\_categoria(categoria)**: Estas funciones obtienen el ID correspondiente para el autor, la editorial y la categoría de un libro.
* **guardar\_libro(nombre\_libro, autor, editorial, anio, genero, comentario)**: Guarda un nuevo libro en la base de datos con todos sus detalles.
* **guardar\_mod\_libro(nombre, libromod)**: Actualiza un libro existente en la base de datos.
* **nuevo\_libro()**: Abre una ventana para agregar un nuevo libro con información como título, autor, género, editorial, año y comentarios.
* **eliminar\_libro()**: Abre una ventana para eliminar un libro seleccionado en el treeview.
* **eliminar\_libro\_db(libro\_id)**: Elimina un libro de la base de datos según su ID.

Luego, en la mainloop (**Tkinter**), se crea una aplicación con un menú que tiene opciones para libros, categorías, editoriales y autores. Además, utiliza un **Treeview** para mostrar los libros con sus detalles y permite operaciones CRUD sobre ellos.