CRUD CON PYTHON Y SQLALCHEMY

Introducción: En este proyecto se desarrolló un sistema CRUD (Crear, Leer, Actualizar y Eliminar) utilizando el lenguaje de programación Python y la librería SQLAlchemy. El objetivo principal fue implementar un programa que permita gestionar estudiantes de manera sencilla a través de una base de datos SQLite, la cual se almacena en un archivo local.

Este sistema busca fortalecer el entendimiento de bases de datos, programación orientada a objetos y el uso de ORM (Object Relational Mapper) para simplificar la interacción con la base de datos.

Objetivo General: Desarrollar un sistema CRUD en Python utilizando SQLAlchemy para la gestión de estudiantes en una base de datos SQLite.

Objetivos Específicos:

1. Crear la base de datos estudiantes.db utilizando SQLite.
2. Implementar operaciones CRUD mediante funciones en Python.
3. Incorporar un menú interactivo en consola para que el usuario seleccione las operaciones.
4. Documentar el proceso de instalación, uso y pruebas del sistema.

Herramientas utilizadas:

- **Lenguaje de programación:** Python 3.13
- **ORM:** SQLAlchemy
- **Base de datos:** SQLite
- **IDE:** PyCharm
- **Sistema operativo:** Windows 11

Diseño de la base de datos:

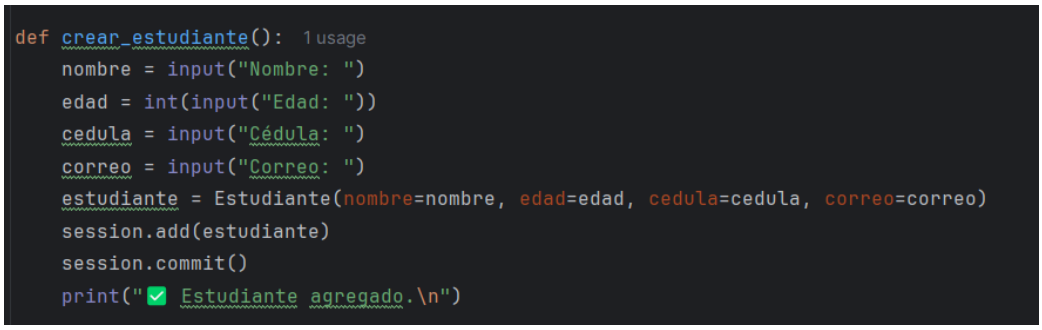
La base de datos cuenta con una sola tabla llamada estudiantes, con los siguientes campos:

- **id:** entero, clave primaria.
- **nombre:** texto.
- **edad:** entero.
- **cedula:** texto único.
- **correo:** texto único.

Desarrollo del sistema:

El programa está dividido en funciones que permiten realizar las operaciones básicas:

- **Crear estudiante:** solicita datos al usuario y los guarda en la base.



```
def crear_estudiante():  1 usage
    nombre = input("Nombre: ")
    edad = int(input("Edad: "))
    cedula = input("Cédula: ")
    correo = input("Correo: ")
    estudiante = Estudiante(nombre=nombre, edad=edad, cedula=cedula, correo=correo)
    session.add(estudiante)
    session.commit()
    print("✅ Estudiante agregado.\n")
```

- **Listar estudiantes:** muestra todos los registros almacenados.

```
def listar_estudiantes():  1 usage
    estudiantes = session.query(Estudiante).all()
    if estudiantes:
        print("\n--- Lista de estudiantes ---")
        for e in estudiantes:
            print(f"{e.id} - {e.nombre} - {e.edad} años - {e.cedula} - {e.correo}")
        print()
    else:
        print("⚠️ No hay estudiantes registrados.\n")
```

- **Actualizar estudiante:** permite modificar la información de un estudiante a partir de su ID.

```
def actualizar_estudiante():  1 usage
    id = int(input("Ingrese el ID del estudiante a actualizar: "))
    estudiante = session.query(Estudiante).filter_by(id=id).first()
    if estudiante:
        print("Deje en blanco si no quiere modificar un campo.")
        nuevo_nombre = input(f"Nuevo nombre ({estudiante.nombre}): ") or estudiante.nombre
        nueva_edad = input(f"Nueva edad ({estudiante.edad}): ")
        nueva_edad = int(nueva_edad) if nueva_edad else estudiante.edad
        nueva_cedula = input(f"Nueva cédula ({estudiante.cedula}): ") or estudiante.cedula
        nuevo_correo = input(f"Nuevo correo ({estudiante.correo}): ") or estudiante.correo

        estudiante.nombre = nuevo_nombre
        estudiante.edad = nueva_edad
        estudiante.cedula = nueva_cedula
        estudiante.correo = nuevo_correo
        session.commit()
        print("✅ Estudiante actualizado.\n")
    else:
        print("❌ Estudiante no encontrado.\n")
```

- **Eliminar estudiante:** elimina un registro según el ID ingresado.

```
def eliminar_estudiante():  1 usage
    id = int(input("Ingrese el ID del estudiante a eliminar: "))
    estudiante = session.query(Estudiante).filter_by(id=id).first()
    if estudiante:
        session.delete(estudiante)
        session.commit()
        print("✅ Estudiante eliminado.\n")
    else:
        print("❌ Estudiante no encontrado.\n")
```

Pruebas de funcionamiento:

En esta sección se presentan capturas de pantalla de la ejecución del programa en consola:

- **Ejemplo de creación de estudiantes:**

```
========= CRUD Estudiantes =========
1. Crear estudiante
2. Listar estudiantes
3. Actualizar estudiante
4. Eliminar estudiante
5. Salir

Seleccione una opción: 1
Nombre: Juan Sebastian
Edad: 22
Cédula: 1000183891
Correo: juansebastian@gmail.com
✅ Estudiante agregado.
```

- **Ejemplo de listado de estudiantes:**

```
========= CRUD Estudiantes =========
1. Crear estudiante
2. Listar estudiantes
3. Actualizar estudiante
4. Eliminar estudiante
5. Salir

Seleccione una opción: 2

--- Lista de estudiantes ---
1 - Juan Sebastian - 22 años - 1000183891 - juansebastian@gmail.com
```

- **Ejemplo de actualización de datos:**

```
========= CRUD Estudiantes =========
1. Crear estudiante
2. Listar estudiantes
3. Actualizar estudiante
4. Eliminar estudiante
5. Salir

Seleccione una opción: 3
Ingrese el ID del estudiante a actualizar: 1
Deje en blanco si no quiere modificar un campo.
Nuevo nombre (Juan Sebastian): Juan
Nueva edad (22):
Nueva cédula (1000183891):
Nuevo correo (juansebastian@gmail.com):
✅ Estudiante actualizado.
```

```
--- Lista de estudiantes ---
1 - Juan - 22 años - 1000183891 - juansebastian@gmail.com
```

- **Ejemplo de eliminación de un registro:**

```
========= CRUD Estudiantes =========
1. Crear estudiante
2. Listar estudiantes
3. Actualizar estudiante
4. Eliminar estudiante
5. Salir

Seleccione una opción: 4
Ingrese el ID del estudiante a eliminar: 1
✅ Estudiante eliminado.
```

```
Seleccione una opción: 2
⚠️ No hay estudiantes registrados.
```

Conclusión: El desarrollo de este CRUD permitió aplicar conocimientos de programación en Python, manejo de bases de datos con SQLite y uso de la librería SQLAlchemy como ORM. Se logró crear un sistema sencillo pero funcional, que puede ser ampliado para su integración con interfaces gráficas o bases de datos más robustas como PostgreSQL o MySQL.