# Bases de datos en Python: la librería sqlite3 SQLite

Es una biblioteca que provee una base de datos liviana, basada en disco local , que no requiere procesos de servidor y permite acceder a los datos mediante SQL.

Se utiliza para almacenamiento interno y para prototipar aplicaciones, para luego transferir el código a una base de datos mayor.

## sqlite3

El módulo de Python *sqlite3* permite manejar bases de datos *SQLite 3.7.15* y superiores.

### ..1) Primeros pasos

En primer lugar, debemos importar el módulo y crear una conexión, que va a representar la base de datos. En el siguiente ejemplo, asignamos el alias **sq3** al módulo sqlite3 y almacenamos los datos de nuestra base de datos en el archivo **mi\_db.db**.

Creamos una variable *con*, que representará nuestra base de datos.

```
import sqlite3 as sq3
con = sq3.connect('mi_db.db')
cur = con.cursor()
```

Una vez establecida la conexión, creamos un objeto de tipo **Cursor**, al que llamaremos **cur**, que invocará al método **execute()** para ejecutar los comandos SQL.

```
3 cur = con.cursor()
```

Ya estamos listos para preparar las instrucciones que van a crear una tabla, cargar y permitirnos visualizar el primer registro:

```
cur.execute('''CREATE TABLE IF NOT EXISTS
alumnos (legajo VARCHAR(7) PRIMARY KEY,
nombre VARCHAR(30),
apellido VARCHAR(30),
nota DECIMAL(10,0))''')

cur.execute('INSERT INTO alumnos VALUES ("2154448", "Ana", "Pérez", 8.5)')

for registro in cur.execute('SELECT * FROM alumnos'):
print(registro)

15
```

Ejecutamos estas instrucciones por medio del método *commit()* y cerramos la conexión con la base de datos mediante el método *close()*. Debemos asegurarnos de ejecutar *commit()* o los comandos no se ejecutarán y los datos se perderán.

```
16 con.commit()
17 con.close()
```

Estos datos son persistentes y se pueden volver a cargar posteriormente.

**ATENCIÓN!!!!:** No debes llamar a tus archivos .py con el mismo nombre de las librerías que estás probando. Python intentará encontrar los métodos invocados en ese archivo y generará un error. **Esto es válido para cualquier import con el que estemos trabajando.** 

#### ..2) Ahorrando líneas

Si nos encontramos en la situación en la que ya existen datos para nuestra base de datos, tal vez necesitemos hacer una carga inicial.

Podemos agregar varios registros en la misma instrucción, cargándolos previamente a una lista e invocar el método **executemany()**, enviando una consulta paramétrica a la base de datos.

```
lista = [
    ("2154489", "Sebastián", "Álvarez", 7.75),
    ("2154563", "Analía", "Rovere", 8.75),
    ("2154896", "Jaime", "Felice", 8),
    ("2154492", "Rosa", "Medina", 9)
]
cur.executemany('INSERT INTO alumnos VALUES (?,?,?,?)', lista)
```

Debemos recordar comentar las líneas en las que agregamos datos una vez ejecutados, para que el programa no intente volver a agregarlos en la siguiente ejecución del programa.

Podrás conseguir el código completo de este ejemplo en el archivo ejemplo\_sqlite3.py

## ..3) Herramientas adicionales (opcional)

DB Browser for SQLite: https://sqlitebrowser.org/dl/

Programa de interfaz gráfica sencillo y liviano para visualizar rápidamente una base de datos en SQLite.