SQLite

SQLite

- SQLite possui tipo flexível, que significa que especificar o tipo de dado de cada coluna é opcional.
- Para confirmar uma transação é necessário rodar o comando commit.
- O comando commit salva as alterações no banco de dados.
- Lembre-se sempre de fechar a conexão ao final do programa, caso contrário ela pode ficar aberta consumindo recursos.

Objetos SQLite

- A biblioteca possui 3 objetos principais: Connection (conexão), Cursor (cursor), e Row (linha).
- **Conection:** representa o banco de dados em si, é utilizada para criação de cursores, e controle de transações.
- **Cursor:** utilizado para executar comandos no banco de dados e receber os resultados.
- **Row:** representa uma linha da tabela e suas colunas.

Tipos de Dados - SQLite e Python

- SQLite nativamente suporta os seguintes tipos de dados:
  - NULL
  - INTEGER
  - REAL
  - TEXT
  - BLOB

SQLite e Python

- Umas das bibliotecas Python utilizadas para manipular SQLite é a PySQLite.
- PySQLite é parte das bibliotecas padrões do python.
- A biblioteca cria automaticamente um banco de dados em arquivos caso não exista. Caso exista, ela abre uma conexão.

Tipos de Dados

Python type	SQLite type
`None`	`NULL`
`int`	`INTEGER`
`float`	`REAL`
`str`	`TEXT`
`bytes`	`BLOB`

SQL

Criação de Tabela

CREATE TABLE nome_tabela(coluna1, coluna2, ...);

Inserção de dados

INSERT INTO nome_tabela VALUES (valor1, valor2, ...);

Consulta de dados

SELECT coluna1, …. FROM nome_tabela;

Inserção de múltiplas linhas

```
INSERT INTO nome_tabela
VALUES
    (valor1, valor2, ...),
    (valor1, valor2, ...),
    (valor1, valor2, ...),
    ...
;
```

Remoção de tabela

```
DROP TABLE nome_tabela;
```

Python

Biblioteca e conexão

```
# Biblioteca e Conexão
import sqlite3 as sql
conexao = sql.connect('projeto.db')
```

Cursor e Executando comandos

```
## Criando tabela
resultado = cursor.execute
(create_table)
## Inserindo dados
cursor.execute(insert)
## Consultando dados
resultado = cursor.execute(select)
resultado.fetchone()
## Confirmando alterações
conexao.commit()
#conexao.rollback()
```

Inserindo múltiplas linhas

```
## Inserindo múltiplas linhas
data = [
    (valor1, valor2),
    (valor3, valor4),
    (valor5, valor6)
]
cursor.executemany('INSERT INTO
nome_tabela VALUES(?,?,?)', data)
# ou
# cursor.executemany('INSERT INTO
nome_tabela VALUES(:chave1,:chave2,
:chave3)', {chave1:valor1,
chave2:valor2, chave3: valor3})
'''Sempre utilize ? no lugar de
f-strings para evitar SQL Injection
'''
resultado.fetchall()
conexao.commit()
```

Rows e Acessando colunas

```
# Rows e Acessando colunas
resultado = cursor.execute(select)
linha = resultado.fetchone()
linha.keys()
linha[0]
linha[1]
linha['coluna1']
linha['coluna2']
```

Apagando tabelas e Encerrando Conexão

```
# Apagando tabela
cursor.execute(drop)
conexao.commit()
cursor.execute(select)

# Encerrando a conexão
conexao.close()
```

Prática

Referências

- https://docs.python.org/3/library/sqlite3.html#sqlite-and-python-types
- https://www.sqlitetutorial.net/sqlite-python/