UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

Facultad de Ingeniería

Ingeniería en Ciencias de la Computación

COMPUTO PARALELO Y DISTRIBUÍDO

1.16 Actividad 4: Threads y Bases de Datos

Alumna: ADRIAN A. GONZÁLEZ DOMÍNGUEZ *[359834]*

Asesor: JOSE SAUL DE LIRA MIRAMONTES

9 de septiembre de 2024

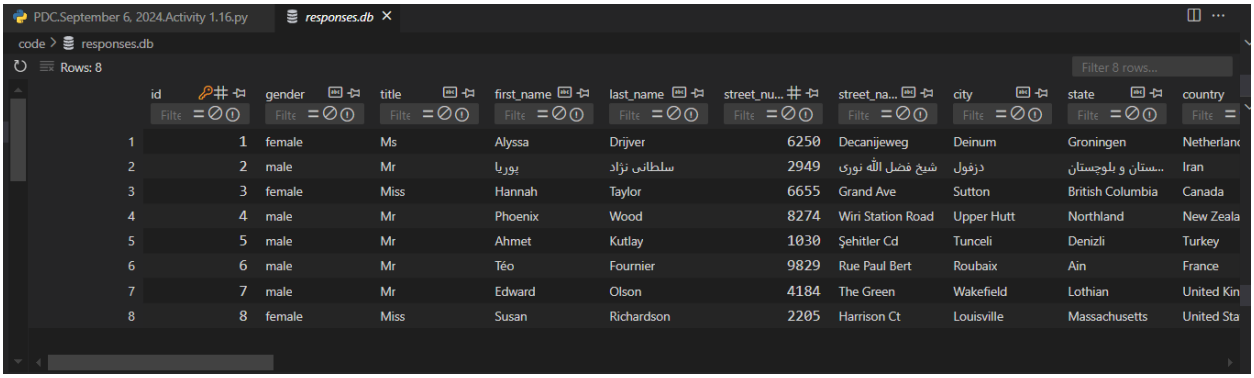
```python
import sqlite3
import requests
import threading
import time

# Crear la tabla con los campos definidos
def create_table():
    conn = sqlite3.connect('responses.db')
    cursor = conn.cursor()
    cursor.execute('''
    CREATE TABLE IF NOT EXISTS users (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        gender TEXT,
        title TEXT,
        first_name TEXT,
        last_name TEXT,
        street_number INTEGER,
        street_name TEXT,
        city TEXT,
        state TEXT,
        country TEXT,
        postcode TEXT,
        latitude TEXT,
        longitude TEXT,
        timezone_offset TEXT,
        timezone_description TEXT,
        email TEXT,
        uuid TEXT,
        username TEXT,
        password TEXT,
        salt TEXT,
        md5 TEXT,
        sha1 TEXT,
        sha256 TEXT,
        dob_date TEXT,
        dob_age INTEGER,
        registered_date TEXT,
        registered_age INTEGER,
        phone TEXT,
```

```
        cell TEXT,
        id_name TEXT,
        id_value TEXT,
        picture_large TEXT,
        picture_medium TEXT,
        picture_thumbnail TEXT,
        nat TEXT,
        seed TEXT,
        results INTEGER,
        page INTEGER,
        version TEXT
    );
    ''')
    conn.commit()
    conn.close()

# Función para hacer una solicitud a la API y almacenar el resultado
def fetch_and_store():
    try:
        response = requests.get('https://randomuser.me/api/')
        if response.status_code == 200:
            data = response.json()
            store_response(data)
        else:
            print(f"Error en la solicitud: {response.status_code}")
    except Exception as e:
        print(f"Error en la solicitud: {e}")

# Almacenar la respuesta en la base de datos
def store_response(data):
    user = data['results'][0]  # Obtenemos el primer resultado
    info = data['info']

    conn = sqlite3.connect('responses.db')
    cursor = conn.cursor()

    cursor.execute('''
    INSERT INTO users (
```

```
        gender, title, first_name, last_name, street_number,
street_name, city, state, country, postcode,
        latitude, longitude, timezone_offset, timezone_description,
email, uuid, username, password, salt, md5, sha1, sha256,
        dob_date, dob_age, registered_date, registered_age, phone,
cell, id_name, id_value, picture_large,
        picture_medium, picture_thumbnail, nat, seed, results, page,
version
    ) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,?,?)
    ''', (
        user['gender'], user['name']['title'], user['name']['first'],
user['name']['last'],
        user['location']['street']['number'],
user['location']['street']['name'], user['location']['city'],
        user['location']['state'], user['location']['country'],
user['location']['postcode'],
        user['location']['coordinates']['latitude'],
user['location']['coordinates']['longitude'],
        user['location']['timezone']['offset'],
user['location']['timezone']['description'],
        user['email'], user['login']['uuid'],
user['login']['username'], user['login']['password'],
        user['login']['salt'], user['login']['md5'],
user['login']['sha1'], user['login']['sha256'],
        user['dob']['date'], user['dob']['age'],
user['registered']['date'], user['registered']['age'],
        user['phone'], user['cell'], user['id']['name'],
user['id']['value'],
        user['picture']['large'], user['picture']['medium'],
user['picture']['thumbnail'],
        user['nat'], info['seed'], info['results'], info['page'],
info['version']
    ))

    conn.commit()
    conn.close()

# Función para manejar los threads
```

```python
def thread_request():
    threads = []
    for _ in range(2):  # 5 requests simultáneas
        thread = threading.Thread(target=fetch_and_store)
        threads.append(thread)
        thread.start()

    for thread in threads:
        thread.join()

# Programa principal
if __name__ == '__main__':
    create_table()  # Crear la tabla si no existe
    while True:
        thread_request()
        time.sleep(10)  # Esperar 10 segundos antes de hacer más
requests
```

```
(venv)
thead@adrigondo MINGW64 /c/Users/thead/Documents/UACH/Seventh Semester/Parallel and Distributing Computing/code
$ python PDC.September\ 6\,\ 2024.Homework.py
Traceback (most recent call last):
  File "C:\Users\thead\Documents\UACH\Seventh Semester\Parallel and Distributing Computing\code\PDC.September 6, 2024.Homework.py", line 116, in <module>
KeyboardInterrupt
```



	id	gender	title	first_name	last_name	street_nu...	street_na...	city	state	country
1	1	female	Ms	Alyssa	Drijver	6250	Decanijeweg	Deinum	Groningen	Netherlanc
2	2	male	Mr	پوریا	سلطانی نژاد	2949	شیخ فضل الله نوری	دزفول	...ستان و بلوچستان	Iran
3	3	female	Miss	Hannah	Taylor	6655	Grand Ave	Sutton	British Columbia	Canada
4	4	male	Mr	Phoenix	Wood	8274	Wiri Station Road	Upper Hutt	Northland	New Zeala
5	5	male	Mr	Ahmet	Kutlay	1030	Şehitler Cd	Tunceli	Denizli	Turkey
6	6	male	Mr	Téo	Fournier	9829	Rue Paul Bert	Roubaix	Ain	France
7	7	male	Mr	Edward	Olson	4184	The Green	Wakefield	Lothian	United Kin
8	8	female	Miss	Susan	Richardson	2205	Harrison Ct	Louisville	Massachusetts	United Sta