

Nombre de la asignatura:  
Tecnologías Emergentes.

Nombre de la actividad:  
Examen 3P

Nombre del alumno:  
Miguel Angel Ramirez Rodriguez.

Registro:  
20310487

Fecha de elaboración  
16/06/2024

Para llevar a cabo este examen explicare paso a paso el desarrollo de esta actividad.

Primero empezamos instalando las librerías Pandas y Requests desde la pantalla de comandos.

```
Símbolo del sistema
Microsoft Windows [Versión 10.0.22631.3737]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Migue>pip install pandas
Collecting pandas
  Downloading pandas-2.2.2-cp310-cp310-win_amd64.whl (11.6 MB)
    11.6/11.6 MB 8.4 MB/s eta 0:00:00
Collecting tzdata>=2022.7
  Downloading tzdata-2024.1-py2.py3-none-any.whl (345 kB)
    345.4/345.4 kB 10.8 MB/s eta 0:00:00
Collecting python-dateutil>=2.8.2
  Downloading python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
    229.9/229.9 kB 7 eta 0:00:00
Collecting pytz>=2020.1
  Downloading pytz-2024.1-py2.py3-none-any.whl (505 kB)
    505.5/505.5 kB 10.5 MB/s eta 0:00:00
Collecting numpy>=1.22.4
  Downloading numpy-2.0.0-cp310-cp310-win_amd64.whl (16.5 MB)
    16.5/16.5 MB 8.3 MB/s eta 0:00:00
Collecting six>=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: pytz, tzdata, six, python-dateutil, pandas
Successfully installed numpy-2.0.0 pandas-2.2.2 python-dateutil-2.9.0.post0 pytz-2024.1 six-1.16.0 tzdata-2024.1

[notice] A new release of pip available: 22.2.1 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
Símbolo del sistema
C:\Users\Migue>pip install requests
Collecting requests
  Downloading requests-2.32.3-py3-none-any.whl (64 kB)
    64.9/64.9 kB 437.5 kB/s eta 0:00:00
Collecting urllib3<3,>=1.21.1
  Downloading urllib3-2.2.1-py3-none-any.whl (121 kB)
    121.1/121.1 kB 644.3 kB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.7-py3-none-any.whl (66 kB)
    66.8/66.8 kB 1.2 MB/s eta 0:00:00
Collecting charset-normalizer<4,>=2
  Downloading charset_normalizer-3.3.2-cp310-cp310-win_amd64.whl (100 kB)
    100.3/100.3 kB 1.4 MB/s eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2024.6.2-py3-none-any.whl (164 kB)
    164.4/164.4 kB 1.1 MB/s eta 0:00:00
Installing collected packages: urllib3, idna, charset-normalizer, certifi, requests
Successfully installed certifi-2024.6.2 charset-normalizer-3.3.2 idna-3.7 requests-2.32.3 urllib3-2.2.1

[notice] A new release of pip available: 22.2.1 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Una vez instalando las librerías procedemos a trabajar con el código en lenguaje Python.

```
*ExamenTE_20310487.py - C:/Users/Migue/Desktop/Universidad/8vo/ExamenTE_20310487.py (3.10.6)*
File Edit Format Run Options Window Help
import requests
import pandas as pd
import hashlib
import time
import sqlite3
import json
#Obtener datos de la API de REST Countries
url = "https://restcountries.com/v3.1/all"
response = requests.get(url)
countries = response.json()
#Listas para almacenar los datos
data = []
#Procesar cada país
for country in countries:
    start_time = time.time()
    name = country.get('name', {}).get('common', 'Unknown')
    languages = country.get('languages', {})
    language_names = list(languages.values())
    if language_names:
        language = language_names[0]
        #Encriptar el idioma con SHA1
        language_shal = hashlib.sha1(language.encode()).hexdigest()
    else:
        language_shal = 'Unknown'
    end_time = time.time()
    processing_time = end_time - start_time
    data.append([name, language_shal, processing_time])
#Crear un DataFrame con Pandas
df = pd.DataFrame(data, columns=['Country', 'Language_SHA1', 'Time'])
#Calcular estadísticas de tiempo
total_time = df['Time'].sum()
average_time = df['Time'].mean()
min_time = df['Time'].min()
max_time = df['Time'].max()
print(f"Total Time: {total_time:.6f} seconds")
print(f"Average Time: {average_time:.6f} seconds")
print(f"Min Time: {min_time:.6f} seconds")
print(f"Max Time: {max_time:.6f} seconds")
#Guardar los datos en una base de datos SQLite
conn = sqlite3.connect('countries.db')
df.to_sql('countries', conn, if_exists='replace', index=False)
#Generar y guardar un archivo JSON con los datos
json_data = df.to_json(orient='records')
with open('data.json', 'w') as json_file:
    json_file.write(json_data)
#Cerrar la conexión a la base de datos
conn.close()
```

### Importación de bibliotecas

- requests: Se utiliza para hacer solicitudes HTTP a la API de REST Countries y obtener los datos.
- pandas: Se usa para trabajar con estructuras de datos tabulares, especialmente para crear y manipular DataFrames.
- hashlib: Proporciona funciones de hash criptográfico, en este caso se usa para encriptar el idioma con SHA1.
- time: Se usa para medir el tiempo de procesamiento de cada país.
- sqlite3: Se utiliza para interactuar con la base de datos SQLite.
- json: Permite manejar archivos JSON, en este caso para guardar los datos en formato JSON.

### Obtener datos de la API de REST Countries

url = "https://restcountries.com/v3.1/all"

Se define la URL de la API de REST Countries que devuelve información de todos los países.

### Procesar cada país

data = []: Inicializa una lista vacía donde se almacenarán los datos de cada país.

El bucle for country in countries: itera sobre cada país en la lista countries.

start\_time = time.time(): Marca el tiempo de inicio del procesamiento para cada país.

name = country.get('name', {}).get('common', 'Unknown'): Obtiene el nombre común del país. Si no hay nombre común, se establece como "Unknown".

languages = country.get('languages', {}): Obtiene el diccionario de idiomas hablados en el país.

language\_names = list(languages.values()): Convierte los valores del diccionario de idiomas en una lista.

if language\_names:: Verifica si hay algún idioma en la lista.

language\_sha1 = hashlib.sha1(language.encode()).hexdigest(): Encripta el primer idioma de la lista con SHA1 y guarda el resultado como language\_sha1.

else: Si no hay idiomas, establece language\_sha1 como "Unknown".

end\_time = time.time(): Marca el tiempo de finalización del procesamiento para cada país.

processing\_time = end\_time - start\_time: Calcula el tiempo total que tomó procesar la información del país.

`data.append([name, language_sha1, processing_time])`: Agrega una lista con el nombre del país, el idioma encriptado y el tiempo de procesamiento a la lista `data`.

### Crear un DataFrame con Pandas

`pd.DataFrame(data, columns=['Country', 'Language_SHA1', 'Time'])`: Crea un DataFrame de Pandas llamado `df` con los datos almacenados en la lista `data`. Los nombres de las columnas se especifican como 'Country', 'Language\_SHA1' y 'Time'.

### Calcular estadísticas de tiempo

`df['Time'].sum()`: Calcula la suma de todos los tiempos de procesamiento.

`df['Time'].mean()`: Calcula el tiempo promedio de procesamiento.

`df['Time'].min()`: Encuentra el tiempo mínimo de procesamiento.

`df['Time'].max()`: Encuentra el tiempo máximo de procesamiento.

`print()`: Imprime las estadísticas de tiempo calculadas con formato de seis decimales.

### Guardar los datos en una base de datos SQLite

`conn = sqlite3.connect('countries.db')`

`df.to_sql('countries', conn, if_exists='replace', index=False)`

`sqlite3.connect('countries.db')`: Crea una conexión a la base de datos SQLite llamada `countries.db` o la abre si ya existe.

`df.to_sql('countries', conn, if_exists='replace', index=False)`: Guarda el DataFrame `df` en la tabla `countries` de la base de datos SQLite. Si la tabla ya existe, la reemplaza (`if_exists='replace'`). No incluye el índice del DataFrame en la tabla (`index=False`).

Generar y guardar un archivo JSON con los datos

`df.to_json(orient='records')`: Convierte el DataFrame `df` en formato JSON con orientación a registros. Esto significa que cada fila del DataFrame se convierte en un objeto JSON.

`with open('data.json', 'w') as json_file`: Abre el archivo `data.json` en modo de escritura.

`json_file.write(json_data)`: Escribe el JSON generado (`json_data`) en el archivo `data.json`.

## Cerrar la conexión a la base de datos

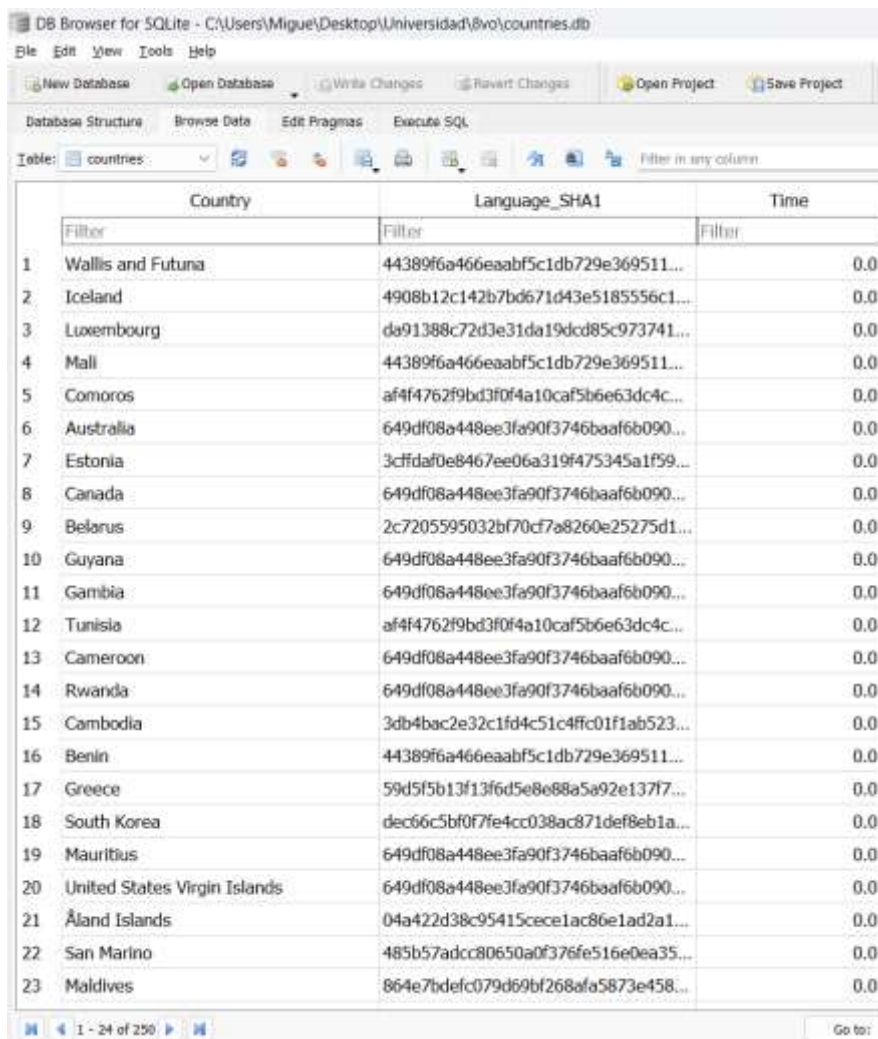
conn.close(): Cierra la conexión a la base de datos SQLite countries.db después de completar todas las operaciones de escritura.

## Una vez desarrollado el código los resultados fueron los siguientes:

Aquí calculamos el tiempo aproximado en el que se procesaron los datos de la api.

```
===== RESTART: C:/Users/Migue/Desktop/Universidad/8vo/ExamenTE_20310487.py =====
Total Time: 0.002741 seconds
Average Time: 0.000011 seconds
Min Time: 0.000000 seconds
Max Time: 0.001664 seconds
>>> |
```

En la tabla de SQLite se almacenaron los datos de la api, después de descargar SQLite abrimos la aplicación y buscamos el documento countries.db ya que así le dimos el nombre en el archivo .py



DB Browser for SQLite - C:\Users\Migue\Desktop\Universidad\8vo\countries.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project

Database Structure Browse Data Edit Pragma Execute SQL

Table: countries

	Country	Language_SHA1	Time
	Filter	Filter	Filter
1	Wallis and Futuna	44389f6a466eaabf5c1db729e369511...	0.0
2	Iceland	4908b12c142b7bd671d43e5185556c1...	0.0
3	Luxembourg	da91388c72d3e31da19dcd85c973741...	0.0
4	Mali	44389f6a466eaabf5c1db729e369511...	0.0
5	Comoros	af4f4762f9bd3f0f4a10caf5b6e63dc4c...	0.0
6	Australia	649df08a448ee3fa90f3746baaf6b090...	0.0
7	Estonia	3cfdaf0e8467ee06a319f475345a1f59...	0.0
8	Canada	649df08a448ee3fa90f3746baaf6b090...	0.0
9	Belarus	2c7205595032bf70cf7a8260e25275d1...	0.0
10	Guyana	649df08a448ee3fa90f3746baaf6b090...	0.0
11	Gambia	649df08a448ee3fa90f3746baaf6b090...	0.0
12	Tunisia	af4f4762f9bd3f0f4a10caf5b6e63dc4c...	0.0
13	Cameroon	649df08a448ee3fa90f3746baaf6b090...	0.0
14	Rwanda	649df08a448ee3fa90f3746baaf6b090...	0.0
15	Cambodia	3db4bac2e32c1fd4c51c4ffc01f1ab523...	0.0
16	Benin	44389f6a466eaabf5c1db729e369511...	0.0
17	Greece	59d5f5b13f13f6d5e8e88a5a92e137f7...	0.0
18	South Korea	dec66c5bf0f7fe4cc038ac871def8eb1a...	0.0
19	Mauritius	649df08a448ee3fa90f3746baaf6b090...	0.0
20	United States Virgin Islands	649df08a448ee3fa90f3746baaf6b090...	0.0
21	Åland Islands	04a422d38c95415cece1ac86e1ad2a1...	0.0
22	San Marino	485b57adcc80650a0f376fe516e0ea35...	0.0
23	Maldives	864e7bdefc079d69bf268afa5873e458...	0.0

1 - 24 of 250

Go to:



## Archivo data.json

```
Ver Ir Ejecutar Terminal ... → trabajos_gulp

data.json x Examen.py Extensión: Python

C:\Users> Miguel > Desktop > Universidad > Ivo > data.json > ...

1 [{"Country": "Wallis and Futuna", "Language_SHA1": "44389f6a466eaabf5c1db729e369511134e2b03b", "Time": 0.0}, {"Country": "Iceland", "Language_SHA1": "4908b12c142b7bd571d43e5185556c158bb0999c", "Time": 0.0}, {"Country": "Luxembourg", "Language_SHA1": "da91388c72d3e11da19dcd85c97374197748485d", "Time": 0.0}, {"Country": "Mali", "Language_SHA1": "44389f6a466eaabf5c1db729e369511134e2b03b", "Time": 0.0}, {"Country": "Comoros", "Language_SHA1": "af4f4762f9bd3f0f4a10caf5b6e63dc4ce543724", "Time": 0.0}, {"Country": "Australia", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "Estonia", "Language_SHA1": "3cfffdafe8467ee06a319f475345a1f59ece46c5", "Time": 0.0}, {"Country": "Canada", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "Belarus", "Language_SHA1": "2c7205595032bf70cf7a8260e25275d136c5ea00", "Time": 0.0}, {"Country": "Guyana", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "Gambia", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "Tunisia", "Language_SHA1": "af4f4762f9bd3f0f4a10caf5b6e63dc4ce543724", "Time": 0.0}, {"Country": "Cameroon", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "Rwanda", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "Cambodia", "Language_SHA1": "3db4bac2e32c1fd45c14c4ff01f1ab523d35aa5f", "Time": 0.0}, {"Country": "Benin", "Language_SHA1": "44389f6a466eaabf5c1db729e369511134e2b03b", "Time": 0.0}, {"Country": "Greece", "Language_SHA1": "59d5f5b13f13f6d5e8e88a5a92e137f7d64b2954", "Time": 0.0}, {"Country": "South Korea", "Language_SHA1": "de66c5bf0f7fe40c038ac871def8eb1ac31c146", "Time": 0.0}, {"Country": "Mauritius", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "United States Virgin Islands", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "Uoocsland Islands", "Language_SHA1": "04a422d38c95a15cece1ac86e1ad2a1030048c03", "Time": 0.0}, {"Country": "San Marino", "Language_SHA1": "485b57dacc80650a0f376fe516e0ea35fef68007", "Time": 0.0}, {"Country": "Maldives", "Language_SHA1": "864e7bdefc079d69bf268afa5873e45801b48a97", "Time": 0.0}, {"Country": "Vanuatu", "Language_SHA1": "634acfe873fe40c04677f04fcdff8c6f585be83", "Time": 0.0}, {"Country": "Malawi", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "Egypt", "Language_SHA1": "af4f4762f9bd3f0f4a10caf5b6e63dc4ce543724", "Time": 0.0}, {"Country": "Senegal", "Language_SHA1": "44389f6a466eaabf5c1db729e369511134e2b03b", "Time": 0.0}, {"Country": "Georgia", "Language_SHA1": "aca441ddd2e0d07643b87c1b24a828fa5b43e42", "Time": 0.0}, {"Country": "New Zealand", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "Cape Verde", "Language_SHA1": "23882c757954a0789bf02aba9e6dd01f539bc738", "Time": 0.0}, {"Country": "Italy", "Language_SHA1": "485b57dacc80650a0f376fe516e0ea35fef68007", "Time": 0.0}, {"Country": "Monaco", "Language_SHA1": "44389f6a466eaabf5c1db729e369511134e2b03b", "Time": 0.0}, {"Country": "Slovakia", "Language_SHA1": "d6a8b06eb489f96a2fcd5d10dc35b13982cbe", "Time": 0.0}, {"Country": "Uruguay", "Language_SHA1": "8df7163b1b2af42d36011e00d22c0f9891ec0b0", "Time": 0.0}, {"Country": "Laos", "Language_SHA1": "c1439807deac3e6c9290cfd97415a4b15f4ba6c8", "Time": 0.0}, {"Country": "Faroe Islands", "Language_SHA1": "9f9f264815f8de2fa0a0756083fbbf633ed8ba1", "Time": 0.0}, {"Country": "Niue", "Language_SHA1": "649df08a448ee3fa90f3746baaf6b0907df42c91", "Time": 0.0}, {"Country": "North Macedonia",
```