Aplicación para extracción de información

Reynosa, Tamaulipas. 23 de mayo del 2021

**Objetivo de la actividad:**

Desarrollar un script en Python que obtenga información sobre la pagina de transcripciones de Andrés Manuel López Obrador, y transferir la información a una base de datos, así como desarrollar el esquema relacional y practicar querys en la base de datos.

**Metodología:**

1. **Importar librerías necesarias en python**

from urllib.request import urlopen as uReq

from bs4 import BeautifulSoup as soup

import pandas as pd

from sqlalchemy import create\_engine

import pymysql

1. **Inicializar las listas donde se almacenará la información**

pages = []

titles = []

links = []

texts = []

dates = []

ids = []

tipo = []

image\_links = []

video\_links = []

audio\_links = []

1. **Encontrar el numero de paginas en el sitio web**

my\_url = 'https://lopezobrador.org.mx/transcripciones/'

uClient = uReq(my\_url)

page\_html = uClient.read()

uClient.close()

page\_soup = soup(page\_html, 'html.parser')

n\_pages = page\_soup.findAll('a', {'class':'page-numbers'})

n\_pages = int(n\_pages[8].text)

1. **Iterar sobre el numero de paginas y almacenar los urls en una lista**

for i in range(1, n\_pages + 1):

    pages.append('https://lopezobrador.org.mx/transcripciones/page/{}/'.format(i))

1. **Iterar sobre la lista anterior, ingresar a cada página, leerla y traducir el HTML. Si hay algún problema al correr el script, el try hará que pase a la siguiente pagina.**

for i in pages:

    try:

        temp\_link = []

        uClient = uReq(i)

        page\_html = uClient.read()

        uClient.close()

        page\_soup = soup(page\_html, 'html.parser')

1. **Encontrar el contenedor principal, donde se encuentran aprox. 6 contenedores, encontrar cada uno de esos 6 contenedores y obtener el enlace a cada uno**

        containers = page\_soup.find('div', class\_='row isotope-container')

        containers\_h2 = page\_soup.findAll('h2', {'class':'entry-title'})

        for link in containers\_h2:

            a = link.find('a')

            links.append(a.get('href'))

            temp\_link.append(a.get('href'))

1. **Encontrar el ID de cada contenedor y agregarlo a la lista**

        for id in containers.findAll('article'):

            id\_temp = id.get('id')

            id\_temp = id\_temp[5:]

            ids.append(int(id\_temp))

1. **Ya que estamos en cada pagina con 6 enlaces, vamos a acceder a cada uno de los enlaces y obtener la información**

        for link in temp\_link:

            temp\_text = ''”

            uClient = uReq(link)

            link\_html = uClient.read()

            uClient.close()

            link\_soup = soup(link\_html, 'html.parser')

1. **Obtener el título y clasificarlo**

            title = link\_soup.find('h1', {'class':'entry-title'})

            titles.append(title.text)

            if "matutina" in title.text:

                tipo.append("Matutina")

            elif "Discurso" in title.text:

                tipo.append("Discurso")

            elif "Avances" in title.text:

                tipo.append("Avances")

            elif "inicio" in title.text:

                tipo.append("Inauguración")

            elif "inauguración" in title.text:

                tipo.append("Inauguración")

            elif "evaluación" in title.text:

                tipo.append("Evaluación")

            else:

                tipo.append("Otro")

1. **Obtener el texto y agregarlo**

            text = link\_soup.findAll('p')

            for i in text:

                a = i.text

                temp\_text = temp\_text + a + " "

            texts.append(temp\_text)

1. **Obtener la fecha y agregarla**

            date = link\_soup.find('span', {'class':'entry-date'})

            dates.append(date.text)

1. **Obtener el contenedor donde aparece la multimedia relevante, encontrar y agregar imágenes, video o audio a su lista correspondiente. Si no se encuentra, se agregara un “NULL” a la lista**

            media = link\_soup.find('div', class\_='content-area right-sidebar col-md-8')

            try:

                img = media.find('img')

                image\_links.append(img.get('src'))

            except:

                image\_links.append('NULL')

            try:

                vid = media.find('iframe')

                video\_links.append(vid.get('src'))

            except:

                video\_links.append('NULL')

            try:

                audio = media.find('audio')

                audio\_links.append(audio.get('src'))

            except:

                audio\_links.append('NULL')

    except:

        pass

1. **Se hacen Pruebas**

print(len(titles))

print(len(links))

print(len(texts))

print(len(dates))

print(len(image\_links))

print(len(video\_links))

print(len(audio\_links))

print(len(ids))

print(len(tipo))

1. **Se dividen los datos en 2 tablas, llamadas discurso y multimedia, y se convierten en dataframe para exportarlas a Mysql**

Discurso\_dict = {'id\_discurso': ids, 'Titulo': titles, 'Fecha': dates, 'URL': links, 'Tipo': tipo, 'Texto': texts}

Discurso\_df = pd.DataFrame(Discurso\_dict)

Multimedia\_dict = {'Discurso\_id\_discurso': ids, 'url\_imagen': image\_links, 'url\_video': video\_links, 'url\_audio': audio\_links}

Multimedia\_df = pd.DataFrame(Multimedia\_dict)

1. **Se crea el enlace con la base de datos y se transfieren los dataframe a la base de datos**

engine = create\_engine("mysql+pymysql://{user}:{pw}@127.0.0.1:3306/{db}"

                       .format(user="Angel",

                               pw="angelomar",

                               db="discursos\_de\_amlo"))

Discurso\_df[0:1160].to\_sql('discurso', con = engine, if\_exists = 'append', index=False)

Multimedia\_df[0:1160].to\_sql('multimedia', con = engine, if\_exists = 'append', index=False)

1. **Se desarrolla el esquema relacional, en donde se encuentran 2 tablas (Figura 1). La tabla discurso tiene como llave primaria el id, y como columnas tiene Titulo, Fecha, URL, Tipo y Texto. La tabla multimedia tiene solamente llave foránea el id, como columnas los URL de imagen, video y audio. Es una relación de 1 a muchos entre la tabla discurso y la tabla multimedia**

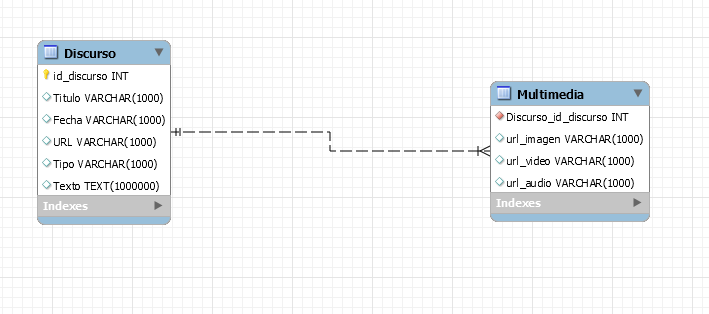


Figura 1

**SQL Querys**

1. **Visualizar todos los datos de cada una de las tablas**

select \* from discurso;

select \* from multimedia;

1. **Muestra los nombres de los tipos distintos de transcripciones**

select distinct Tipo as Tipo\_de\_transcripción

from discurso;

1. **Muestra el ID y la URL de la imagen de la tabla de multimedia cuando imagen y video no son nulos**

select Discurso\_id\_discurso, url\_imagen

FROM multimedia

where url\_imagen is not null

and url\_video is not null;

1. **Selecciona los valores de id de la tabla de multimedia donde el audio es nulo**

select Discurso\_id\_discurso

FROM multimedia

where url\_audio is null;

1. **Cuenta cuantas entradas son de tipo Matutina de la tabla de discurso**

select count(\*)

from discurso

where Tipo = 'Matutina';

1. **Selecciona con el nombre de selección cuantas veces en el texto de la tabla de discurso se menciona la palabra “reelección”.**

SELECT COUNT(\*) as reelección

FROM discurso

WHERE Texto like "%reelección%";

1. **Selecciona con el nombre de selección cuantas veces en el texto de la tabla de discurso se menciona la palabra “neoliberal”.**

SELECT COUNT(\*) as neoliberal

FROM discurso

WHERE Texto like "%neoliberal%";

1. **Selecciona con el nombre de selección cuantas veces en el texto de la tabla de discurso se menciona la palabra “pueblo”.**

SELECT COUNT(\*) as pueblo

FROM discurso

WHERE Texto like "%pueblo%";

1. **Selecciona con el nombre de selección cuantas veces en el texto de la tabla de discurso se menciona la palabra “otros datos”.**

SELECT COUNT(\*) as otros\_datos

FROM discurso

WHERE Texto like "%otros datos%";

1. **Selecciona con el nombre de selección cuantas veces en el texto de la tabla de discurso se menciona la palabra “corrupción”.**

SELECT COUNT(\*) as corrupción

FROM discurso

WHERE Texto like "%corrupción%";

1. **Selecciona el titulo y el tipo de la tabla de discurso donde el tipo sea matutina**

select Titulo, Tipo

FROM discurso

where Tipo = 'Matutina';

1. **Cuenta cuantos tipos distintos de transcripciones hay en la tabla discurso**

select count(distinct Tipo)

from discurso;

1. **Selecciona todos los datos con un inner join entre discurso y multimedia, se juntan en el id y se ordenan por el tipo**

select \*

from discurso

inner join multimedia

on discurso.id\_discurso = multimedia.Discurso\_id\_discurso

order by Tipo;

1. **Selecciona el id, titulo e imagen de un inner join entre las tablas discurso y multimedia, que se juntan en el id y se ordenan por el id de forma descendiente**

select id\_discurso, Titulo, url\_imagen

from discurso

inner join multimedia

on discurso.id\_discurso = multimedia.Discurso\_id\_discurso

order by id\_discurso desc;

1. **Se seleccionan todos los datos de un inner join entre discurso y multimedia, se juntan en el id y se agrupan en el tipo**

select \*

from discurso

inner join multimedia

on discurso.id\_discurso = multimedia.Discurso\_id\_discurso

group by Tipo;

1. **Se seleccionan todos los datos de un cross join entre discurso y multimedia, se selecciona el resultado donde el tipo sea matutina y con fecha del 20 de mayo del 2020**

select \*

from discurso

cross join multimedia

where Tipo = "Matutina"

and Fecha = "mayo 20, 2020";

1. **Se hace una consulta anidada, donde se obtiene el id de multimedia donde el audio es nulo, y de el se selecciona la fecha de discurso que coincida con ese id de la consulta anidada.**

select Fecha

from discurso

where id\_discurso in

(

select Discurso\_id\_discurso

from multimedia

where url\_audio is null

);

**Conclusiones:**

Se aprendió sobre como extraer información de paginas de internet a través de Python, como almacenarla en una base de datos y como escribir consultas para obtener información. Como datos interesantes, amlo ha tenido 578 mañaneras, en las cuales ha mencionado 125 veces la palabra “reelección”, 644 veces la palabra “neoliberal”, 1126 la palabra “pueblo”, 47 veces las palabras “otros datos” y 974 veces la palabra “corrupción”.