```
#encoding:utf-8

from bs4 import BeautifulSoup
import urllib.request
from tkinter import *
from tkinter import messagebox
import sqlite3
import lxml

# lineas para evitar error SSL
import os, ssl
if (not os.environ.get('PYTHONHTTPSVERIFY', '') and
getattr(ssl, '_create_unverified_context', None)):
    ssl._create_default_https_context = ssl._create_unverified_context


def cargar():
    respuesta = messagebox.askyesno(title="Confirmar",message="Esta seguro que
quiere recargar los datos. \nEsta operaciÃ³n puede ser lenta")
    if respuesta:
        almacenar_bd()

def extraer_elementos():
    lista=[]

    for num_paginas in range(0,3):
        url =
"https://www.vinissimus.com/es/vinos/tinto/index.html?cursor="+str(num_paginas*3
6)
        f = urllib.request.urlopen(url)
        s = BeautifulSoup(f, "lxml")
        lista_una_pagina = s.find_all("div", class_="product-list-item")
        lista.extend(lista_una_pagina)

    return lista


def almacenar_bd():
    conn = sqlite3.connect('vinos.db')
    conn.text_factory = str
    conn.execute("DROP TABLE IF EXISTS VINO")
    conn.execute("DROP TABLE IF EXISTS TIPOS_UVAS")
    conn.execute('''CREATE TABLE VINO
       (NOMBRE            TEXT NOT NULL,
        PRECIO            REAL,
        DENOMINACION      TEXT,
        BODEGA            TEXT,
        TIPO_UVAS         TEXT);''')
    conn.execute('''CREATE TABLE TIPOS_UVAS
       (NOMBRE            TEXT NOT NULL);''')

    lista_vinos = extraer_elementos()

    tipos_uva=set() # conjunto con los distintos tipos de uvas que al final se
```

```
almacena en la tabla TIPOS_UVAS
    for vino in lista_vinos:
        datos = vino.find("div",class_=["details"])
        nombre = datos.a.h2.string.strip()
        bodega = datos.find("div", class_=["cellar-name"]).string.strip()
        denominacion  = datos.find("div",class_=["region"]).string.strip()
        uvas = "".join(datos.find("div",class_=["tags"]).stripped_strings)
        # almacenamos en el conjunto de tipos de uvas
        for uva in uvas.split("/"):
            tipos_uva.add(uva.strip())
        precio = list(vino.find("p",class_=["price"]).stripped_strings)[0]
        #si tiene descuento el prcio es el del descuento
        dto =
vino.find("p",class_=["price"]).find_next_sibling("p",class_="dto")
        if dto:
            precio = list(dto.stripped_strings)[0]

        conn.execute("""INSERT INTO VINO (NOMBRE, PRECIO, DENOMINACION, BODEGA,
TIPO_UVAS) VALUES (?,?,?,?,?)""",
                     (nombre, float(precio.replace(',','.')), denominacion,
bodega, uvas))
    conn.commit()

    #insertamos el la tabla TIPOS_UVAS los elemento sdel conjunto tipos_uva
    for u in list(tipos_uva):
        conn.execute("""INSERT INTO TIPOS_UVAS (NOMBRE) VALUES (?)""",
                     (u,))
    conn.commit()

    cursor = conn.execute("SELECT COUNT(*) FROM VINO")
    cursor1 = conn.execute("SELECT COUNT(*) FROM TIPOS_UVAS")
    messagebox.showinfo("Base Datos",
                        "Base de datos creada correctamente \nHay " +
str(cursor.fetchone()[0]) + " vinos y "
                        + str(cursor1.fetchone()[0]) + " tipos de uvas")
    conn.close()

def listar_todos():
    conn = sqlite3.connect('vinos.db')
    conn.text_factory = str
    cursor = conn.execute("SELECT NOMBRE, PRECIO, BODEGA, DENOMINACION FROM
VINO")
    conn.close
    listar_vinos(cursor)


def buscar_por_denominacion():
    def listar(event):
            conn = sqlite3.connect('vinos.db')
            conn.text_factory = str
            cursor = conn.execute("SELECT NOMBRE, PRECIO, BODEGA, DENOMINACION
FROM VINO WHERE DENOMINACION LIKE '%" + str(entry.get()) + "%'")
            conn.close
            listar_vinos(cursor)
```

```
    conn = sqlite3.connect('vinos.db')
    conn.text_factory = str
    cursor = conn.execute("SELECT DISTINCT DENOMINACION FROM VINO")
    denominaciones = [d[0] for d in cursor]


    ventana = Toplevel()
    label = Label(ventana,text="Seleccione una denominaciÃ³n de origen: ")
    label.pack(side=LEFT)
    entry = Spinbox(ventana, width= 30, values=denominaciones)
    entry.bind("<Return>", listar)
    entry.pack(side=LEFT)

    conn.close


def buscar_por_precio():
    def listar(event):
            conn = sqlite3.connect('vinos.db')
            conn.text_factory = str
            cursor = conn.execute("SELECT NOMBRE, PRECIO, BODEGA, DENOMINACION
FROM VINO WHERE PRECIO <= ? ORDER BY PRECIO", (str(entry.get()),))
            conn.close
            listar_vinos(cursor)
    ventana = Toplevel()
    label = Label(ventana, text="Indique el precio mÃ¡ximo: ")
    label.pack(side=LEFT)
    entry = Entry(ventana)
    entry.bind("<Return>", listar)
    entry.pack(side=LEFT)



def buscar_por_uvas():
    def listar(event):
            conn = sqlite3.connect('vinos.db')
            conn.text_factory = str
            cursor = conn.execute("SELECT NOMBRE, TIPO_UVAS FROM VINO where
TIPO_UVAS LIKE '%" + str(tipo_uva.get()) + "%'")
            conn.close
            listar_por_uvas(cursor)

    conn = sqlite3.connect('vinos.db')
    conn.text_factory = str
    cursor = conn.execute("SELECT NOMBRE FROM TIPOS_UVAS")

    tipos_uva=[u[0] for u in cursor]

    v = Toplevel()
    label = Label(v,text="Seleccione el tipo de uva: ")
    label.pack(side=LEFT)
    tipo_uva = Spinbox(v, width= 30, values=tipos_uva)
    tipo_uva.bind("<Return>", listar)
```

```
    tipo_uva.pack(side=LEFT)

    conn.close()



def listar_por_uvas(cursor):
    v = Toplevel()
    sc = Scrollbar(v)
    sc.pack(side=RIGHT, fill=Y)
    lb = Listbox(v, width=150, yscrollcommand=sc.set)
    for row in cursor:
        s = 'VINO: ' + row[0]
        lb.insert(END, s)
        lb.insert(END, "-----------------------------------------------------")
        s = "     TIPOS DE UVA: " + row[1]
        lb.insert(END, s)
        lb.insert(END, "\n\n")
    lb.pack(side=LEFT, fill=BOTH)
    sc.config(command=lb.yview)



def listar_vinos(cursor):
    v = Toplevel()
    sc = Scrollbar(v)
    sc.pack(side=RIGHT, fill=Y)
    lb = Listbox(v, width=150, yscrollcommand=sc.set)
    for row in cursor:
        s = 'VINO: ' + row[0]
        lb.insert(END, s)
        lb.insert(END,
"-------------------------------------------------------------------------")
        s = "     PRECIO: " + str(row[1]) + ' | BODEGA: ' + row[2]+ ' |
DENOMINACION: ' + row[3]
        lb.insert(END, s)
        lb.insert(END,"\n\n")
    lb.pack(side=LEFT, fill=BOTH)
    sc.config(command=lb.yview)



def ventana_principal():
    raiz = Tk()

    menu = Menu(raiz)

    #DATOS
    menudatos = Menu(menu, tearoff=0)
    menudatos.add_command(label="Cargar", command=cargar)
    menudatos.add_command(label="Listar", command=listar_todos)
    menudatos.add_command(label="Salir", command=raiz.quit)
    menu.add_cascade(label="Datos", menu=menudatos)
```

```
    #BUSCAR
    menubuscar = Menu(menu, tearoff=0)
    menubuscar.add_command(label="DenominaciÃ³n",
command=buscar_por_denominacion)
    menubuscar.add_command(label="Precio", command=buscar_por_precio)
    menubuscar.add_command(label="Uvas", command=buscar_por_uvas)
    menu.add_cascade(label="Buscar", menu=menubuscar)

    raiz.config(menu=menu)

    raiz.mainloop()



if __name__ == "__main__":
    ventana_principal()
```