```
import sqlite3
import urllib.error
import ssl
from urllib.parse import urljoin
from urllib.parse import urlparse
from urllib.request import urlopen
from bs4 import BeautifulSoup

# Ignore SSL certificate errors
ctx = ssl.create_default_context()
||
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE

conn = sqlite3.connect('spider.sqlite')
cur = conn.cursor()

cur.execute('''CREATE TABLE IF NOT EXISTS Pages
    (id INTEGER PRIMARY KEY, url TEXT UNIQUE, html TEXT,
     error INTEGER, old_rank REAL, new_rank REAL)''')

cur.execute('''CREATE TABLE IF NOT EXISTS Links
    (from_id INTEGER, to_id INTEGER, UNIQUE(from_id, to_id))''')

cur.execute('''CREATE TABLE IF NOT EXISTS Webs (url TEXT UNIQUE)''')

# Check to see if we are already in progress...
cur.execute('SELECT id,url FROM Pages WHERE html is NULL and error is NULL ORDER BY
RANDOM() LIMIT 1')
row = cur.fetchone()
if row is not None:
    print("Restarting existing crawl.  Remove spider.sqlite to start a fresh crawl.")
else :
    starturl = input('Enter web url or enter: ')
    if ( len(starturl) < 1 ) : starturl = 'http://www.dr-chuck.com/'
    if ( starturl.endswith('/') ) : starturl = starturl[:-1]
    web = starturl
    if ( starturl.endswith('.htm') or starturl.endswith('.html') ) :
        pos = starturl.rfind('/')
        web = starturl[:pos]

    if ( len(web) > 1 ) :
        cur.execute('INSERT OR IGNORE INTO Webs (url) VALUES ( ? )', ( web, ) )
        cur.execute('INSERT OR IGNORE INTO Pages (url, html, new_rank) VALUES ( ?,
NULL, 1.0 )', ( starturl, ) )
        conn.commit()

# Get the current webs
cur.execute('''SELECT url FROM Webs''')
webs = list()
for row in cur:
    webs.append(str(row[0]))

print(webs)

many = 0
while True:
    if ( many < 1 ) :
        sval = input('How many pages:')
```

```
        if ( len(sval) < 1 ) : break
        many = int(sval)
    many = many - 1

    cur.execute('SELECT id,url FROM Pages WHERE html is NULL and error is NULL ORDER
BY RANDOM() LIMIT 1')
    try:
        row = cur.fetchone()
        # print row
        fromid = row[0]
        url = row[1]
    except:
        print('No unretrieved HTML pages found')
        many = 0
        break

    print(fromid, url, end=' ')

    # If we are retrieving this page, there should be no links from it
    cur.execute('DELETE from Links WHERE from_id=?', (fromid, ) )
    try:
        document = urlopen(url, context=ctx)

        html = document.read()
        if document.getcode() != 200 :
            print("Error on page: ",document.getcode())
            cur.execute('UPDATE Pages SET error=? WHERE url=?', (document.getcode(),
url) )

        if 'text/html' != document.info().get_content_type() :
            print("Ignore non text/html page")
            cur.execute('DELETE FROM Pages WHERE url=?', ( url, ) )
            conn.commit()
            continue

        print('('+str(len(html))+')', end=' ')

        soup = BeautifulSoup(html, "html.parser")
    except KeyboardInterrupt:
        print('')
        print('Program interrupted by user...')
        break
    except:
        print("Unable to retrieve or parse page")
        cur.execute('UPDATE Pages SET error=-1 WHERE url=?', (url, ) )
        conn.commit()
        continue

    cur.execute('INSERT OR IGNORE INTO Pages (url, html, new_rank) VALUES ( ?, NULL,
1.0 )', ( url, ) )
    cur.execute('UPDATE Pages SET html=? WHERE url=?', (memoryview(html), url ) )
    conn.commit()

    # Retrieve all of the anchor tags
    tags = soup('a')
    count = 0
    for tag in tags:
        href = tag.get('href', None)
        if ( href is None ) : continue
        # Resolve relative references like href="/contact"
```

```
        up = urlparse(href)
        if ( len(up.scheme) < 1 ) :
            href = urljoin(url, href)
        ipos = href.find('#')
        if ( ipos > 1 ) : href = href[:ipos]
        if ( href.endswith('.png') or href.endswith('.jpg') or href.endswith('.gif')
) : continue
        if ( href.endswith('/') ) : href = href[:-1]
        # print href
        if ( len(href) < 1 ) : continue

    # Check if the URL is in any of the webs
        found = False
        for web in webs:
            if ( href.startswith(web) ) :
                found = True
                break
        if not found : continue

        cur.execute('INSERT OR IGNORE INTO Pages (url, html, new_rank) VALUES ( ?,
NULL, 1.0 )', ( href, ) )
        count = count + 1
        conn.commit()

        cur.execute('SELECT id FROM Pages WHERE url=? LIMIT 1', ( href, ))
        try:
            row = cur.fetchone()
            toid = row[0]
        except:
            print('Could not retrieve id')
            continue
        # print fromid, toid
        cur.execute('INSERT OR IGNORE INTO Links (from_id, to_id) VALUES ( ?, ? )', (
fromid, toid ) )


    print(count)

cur.close()

```