Collect pageviews of pages that transclude modules

This notebook is not a tutorial or full work. The full work is in our GitHub repo. This notebook contains experiments and explorations that led to the final code in our project.

What's in this notebook:

- Set up and test API call to fetch pageview count

What's not in this notebook:

- Full script to fetch pageview for all pages that transclude a module and save in user-database
- Use of REST API instead of MWAPI to get pageviews
- Use of Pagedumps to get pageviews

These are not in this notebook but are related to fetching pageviews. Code is in repo mentioned above.

Installs

In [ ]:

```
!pip install toolforge
```

Imports

In [2]:

```
import mwapi
import pprint
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from IPython.display import display
from urllib.parse import unquote
import pymysql

import toolforge

import os
from sqlalchemy import create_engine
from sqlalchemy.sql import text

import mwapi

pp = pprint.PrettyPrinter()
pd.set_option('display.max_columns', None)
```

Utils

In [3]:

```
import uuid
from IPython.display import display_javascript, display_html, display
import json

class RenderJSON(object):
    def __init__(self, json_data):
        if isinstance(json_data, dict):
            self.json_str = json.dumps(json_data)
        else:
            self.json_str = json
        self.uuid = str(uuid.uuid4())

    def _ipython_display_(self):
        display_html('<div id="{}" style="width:100%;"></div>'.format(self.uuid), #height: 600px;
            raw=True
        )
        display_javascript("""
        require(["https://rawgit.com/caldwell/renderjson/master/renderjson.js"], function() {
          document.getElementById('%s').appendChild(renderjson(%s))
        });
        """ % (self.uuid, self.json_str), raw=True)
```

Database

In [4]:

```
def encode_if_necessary(b):
    if type(b) is bytes:
        return b.decode('utf8')
    return b
```

In [5]:

```
user_agent = toolforge.set_user_agent('test-tool-aisha')
```

In [29]:

```
def connectdb(dbname):
    conn = toolforge.connect(
                host=os.environ['MYSQL_HOST'],
                user=os.environ['MYSQL_USERNAME'],
                password=os.environ['MYSQL_PASSWORD'],
                dbname=dbname,
                charset='utf8'
            )
    return conn
```

In [30]:

```
with open('config.txt','r') as file:
    user, password = file.read().split()
def connect_toolsdb(dbname):
    conn = toolforge.toolsdb(
                user=user,
                password=password,
                dbname=dbname,
                charset='utf8'
            )
    return conn
```

In [8]:

```
def query2df(db, query, vals=None):
    conn = connectdb(db)
    with conn.cursor() as cur:
        cur.execute("use "+db+'_p')
        SQL_Query = pd.read_sql_query(query, conn, params=vals)
        df = pd.DataFrame(SQL_Query).applymap(encode_if_necessary)
    conn.close()
    return df
```

Get all dbs

In [9]:

```
conn = connectdb('meta')

with conn.cursor() as cur:
    cur.execute("use meta_p")
    cur.execute("select dbname, url from wiki where is_closed=0")
    map_df = pd.DataFrame(cur, columns=['dbname', 'url'])

map_df.head()
```

Out[9]:

	dbname	url
0	abwiki	https://ab.wikipedia.org
1	acewiki	https://ace.wikipedia.org
2	adywiki	https://ady.wikipedia.org
3	afwiki	https://af.wikipedia.org
4	afwikibooks	https://af.wikibooks.org

Get all Module info

In [10]:

```
def db2link(db):
    return map_df[map_df['dbname']==db]['url'].values[0]
```

In [11]:

```
def get_pages_df(df):
    user_agent = toolforge.set_user_agent('abstract-wiki-ds')
    out_df = pd.DataFrame()
    for wiki, w_df in df.groupby('url'):
        session = mwapi.Session(wiki, user_agent=user_agent)
        pageids = w_df['page_id'].values
        data_list = []
        missed = []

        for pageid in list(pageids):
            params = {
                'action':'query',
                'format':'json',
                'prop':'revisions|info',
                'pageids':pageid,
                'rvprop':'content',
                'rvslots':'main',
                'inprop':'url',
                'formatversion':2
            }

            try:
                result = session.get(params)
                page = result['query']['pages'][0]
                if page['lastrevid']!=0:
#                   pp.pprint(result)
                    title = page['title']
                    url = unquote(page['fullurl'])
                    length = page['length']
                    content_info = page['revisions'][0]['slots']['main']
                    content_format = content_info['contentformat']
                    content_model = content_info['contentmodel']
                    touched = page['touched']
                    content = content_info['content']
                    ns = page['ns']
                    data_list.append([pageid, title, url, length, content, content_format, content_model, touched, ns, wiki])
            except Exception as e:
                missed.append([pageid, wiki])
                print("Miss:", pageid, "from wiki:", wiki, "\n", e)

        print("All pages loaded for %s. Missed: %d, Loaded: %d" \
            %(wiki, len(missed), len(data_list)))
        out_df = out_df.append(pd.DataFrame(data_list,
                                            columns=['pageid', 'titile', 'url', 'length',
                                                     'content', 'content_format', 'content_model',
                                                     'touched', 'ns', 'wiki']))

    print("Done loading missed pages!")

    return out_df
```

In [12]:

```
# pclimit: How many contributors to return.

def get_page(pageid, wiki):
    session = mwapi.Session(wiki, user_agent=user_agent)
    ret = {}
    params = {
            'action':'query',
            'format':'json',
            'prop':'revisions|info|categories|contributors|transcludedin|pageviews|iwlinks|langlinks|linkshere|pageterms|templates',
            'rvprop':'content|flags|tags',
            'rvslots':'main',
            'rvlimit':1,
            'pclimit':10, #'max'
            'tinamespace':'*',
            'tilimit':10, #'max'
            'pageids':pageid,
            'inprop':'url|watchers|protection',
            'iwprop':'url',
            'iwlimit':10, #'max'
            'llprop':'autonym|langname|url',
            'lllimit':10, #'max'
            'lhprop': 'pageid|redirect|title',
            'lhnamespace':'*',
            'lhlimit':10,#'max'
            'tlnamespace':828,
            'tllimit':10,#'max'
            'formatversion':2,
#             'list':'iwbacklinks'
        }
    cur_params = params
    while True:
        result = session.get(cur_params)
        yield result
#         ret = {**ret, **result['query']}
        cur_params = params
        try:
            cn = result['continue']
            for k,v in cn.items():
                cur_params[k]=v
        except:
            break
#     return ret
```

In [ ]:

```
for ret in get_page('310650', db2link('bnwiki')):
    display(RenderJSON(ret))
```

In [16]:

```
d = pd.DataFrame([[310650, db2link('bnwiki')]], columns=['page_id', 'url'])
get_pages_df(d)
```

```
All pages loaded for https://bn.wikipedia.org. Missed: 0, Loaded: 1
Done loading missed pages!
```

Out[16]:

	pageid	titile	url	length	content	content_format	content_model	touched	ns	wiki
0	310650	মডিউল:Convert	https://bn.wikipedia.org/wiki/মডিউল:Convert	114159	-- Convert a value from one unit of measuremen...	text/plain	Scribunto	2021-01-01T18:43:38Z	828	https://bn.wikipedia.org

In [41]:

```
d = pd.DataFrame([[325520, db2link('bnwiki')]], columns=['page_id', 'url'])
get_pages_df(d)
```

```
All pages loaded for https://bn.wikipedia.org. Missed: 0, Loaded: 1
Done loading missed pages!
```

Out[41]:

	pageid	titile	url	length	content	content_format	content_model	touched	ns	wiki
0	325520	মডিউল:Yesno	https://bn.wikipedia.org/wiki/মডিউল:Yesno	951	-- Function allowing for consistent treatment ...	text/plain	Scribunto	2020-12-30T13:00:38Z	828	https://bn.wikipedia.org

Pageviews

In [13]:

```
def get_pageviews(pageid, wiki, days):
    session = mwapi.Session(wiki, user_agent=user_agent)
    params = {
            'action':'query',
            'format':'json',
            'prop':'pageviews',
            'pageids':pageid,
            'pvipdays':days,
            'formatversion':2,
        }
    result = session.get(params)
    cnt = 0
    for k,v in result['query']['pages'][0]['pageviews'].items():
        if v:
            cnt += v
    return cnt
```

In [14]:

```
get_pageviews('310650', db2link('bnwiki'), 60)
```

Out[14]: 9

In [33]:

```
## Not required atm since we dont need page view of `module page`,
# rather page view of pages that transclude this module page

def get_pageviews_list(wiki, days):
    session = mwapi.Session(wiki, user_agent=user_agent)
    params = {
            'action':'query',
            'format':'json',
            'prop':'pageviews',
            'generator':'allpages',
            'gapnamespace':828,
            'pvipdays':days,
            'formatversion':2,
        }
    cur_params = params
    while True:
        result = session.get(cur_params)
        yield result
        cur_params = params
        try:
            cn = result['continue']
            for k,v in cn.items():
                cur_params[k]=v
        except:
            break

for res in get_pageviews_list(db2link('bnwiki'), 60):

    try:

        page_id_views = []

        for page in res['query']['pages']:
            if 'pageviews' in page.keys():
                cnt = 0
                for k,v in page['pageviews'].items():
                    if v:
                        cnt += v
                page_id_views.append((page['pageid'], cnt))

#         save_views(page_id_views)
#         print(page_id_views)

    except Exception as err:
        print("Something went wrong.", err)
```

Done in get_pageviews.py file:

Steps:

- get list of modules from `Scripts` user-db
- for each page, get list of pages it was transcluded in
- get pageviews for those pages and sum them all up
- save the sum for each module in user-db

Check transclusions

In [35]:

```
db = 'bnwiki'
query = ("select count(*) as cnt, tl_title from templatelinks "
         "where tl_namespace=828 "
         "group by tl_title order by cnt desc "
         "limit 10")
conn = connectdb(db)
query2df(db, query)
```

Out[35]:

	cnt	tl_title
0	439724	Arguments
1	407718	Yesno
2	268518	No_globals
3	229652	বার্তার_বাক্স
4	229574	বার্তার_বাক্স/রূপরেখা
5	169651	নামস্থান_শনাক্তকরণ/উপাত্ত
6	169651	নামস্থান_শনাক্তকরণ/রূপরেখা
7	136885	String
8	134174	এর
9	113244	নামস্থান_শনাক্তকরণ

In [40]:

```
db = 'bnwiki'
query = ("select * from page where page_title like 'Yesno'")
conn = connectdb(db)
query2df(db, query)
```

Out[40]:

	page_id	page_namespace	page_title	page_restrictions	page_is_redirect	page_is_new	page_random	page_touched	page_links_updated	page_latest	page_len	page_content_model	page_lang
0	325520	828	Yesno		0	1	0.734161	20201230130038	20201230124520	2355519	951	Scribunto	None
1	516091	10	Yesno		1	1	0.516771	20161013153452	20201128165159	2371843	61	wikitext	None

In [ ]: