Input required for script:

1. Facebook Group Admin's "Email address"
2. Facebook Group Admin's "Password"
3. User Access Token

4. Group id - An integer which shows up in the homepage URL of group e.g. "https://www.facebook.com/groups/1662317732110904" has group id as "1662317732110904".

Computer Requirements -

1. Python 2.7 needs to be installed on computer to run this script. Ref - https://www.python.org/download/releases/2.7/
2. Mozilla Firefox browser needs to be present on computer. Ref- https://www.mozilla.org/en-CA/firefox/new/

Facebook Group Sitemap Structure Analysis:

Algorithm/Code Highlights:

1. Functions extract_posts(), extract_comments() and extract_replies() have been written to extract all data using Facebook API. These functions call each other to extract - a. Combined content of posts/comments/replies. b. A unique id for each post/comment/reply. c. Their corresponding timestamps. An HTTPS GET request is sent to the Facebook for this data extraction.
2. Function user_values() is the core function that extract all other data which is not feasible with Facebook API. It uses Selenium to extract - a. All user ids of each post, comment and reply. b. Seen count of each post and exact user ids of users who have seen that post (seen count + user ids of users who have seen the post). c. All reactions (Like, Love, Haha, Wow, Sad, Angry, any special reactions that come around sometimes like Pride reaction etc.) along with the user ids of users who reacted is being extracted for each post, comment and reply. (reactions+ user ids pairs for each post/comment/reply).
3. Facebook has a complete Sitemap hierarchy for group, post, comment/reply, post seen list, reactions for post, reactions for comments/replies URLs -

   a. group_url="https://www.facebook.com/groups/"+group_id+"/"

   b. post_url="https://www.facebook.com/groups/"+group_id+"/"+post_id+"/"

   c. comment_or_reply_url="https://www.facebook.com/groups/"+group_id+"/"+post_id+"/?comment_id="+ids

   d. seen_url='https://www.facebook.com/ufi/group/seenby/profile/browser/?id='+post_id+'&av='+unique_id

   e. reactions_url='https://www.facebook.com/ufi/reaction/profile/browser/?ft_ent_identifier='+post_id+'&av='+unique_id

   f. reactions_url='https://www.facebook.com/ufi/reaction/profile/browser/?ft_ent_identifier='+ids+'&av='+unique_id

   g. "https://www.facebook.com/profile.php?id=" + "any user id" gives us profile of the particular person/user.

1. These variables names have been used as: a. group_id: A unique group id. b. post_id: A unique id for a particular post. c. comment_id: A unique id for a particular comment or reply. d. unique_id: A unique id for each Admin which creates the group (which is the user id/profile id of the Admin.) e. ids: A unique id for a particular comment or reply.
2. Each URL is opened with Selenium making it possible to extract data one at a time from each page. This helps in overcoming the hidden "comment/reply/seen user ids/reaction user ids" issue.
3. This code also handles edge cases like no reactions, 0 seen counts etc and it also does exception handling if in case browser freezes, it will restart itself and resume from the same place.

Import Tkinter for UI (User Interface) Help Doc for installing Tkinter before importing - https://tkdocs.com/tutorial/install.html

In [1]:

```
from Tkinter import *
import Tkinter
import tkMessageBox
```

Import simple libraries for sending http/https requests and fetching data re library for regular expressions and json for handling json formats

In [2]:

```
import httplib, urllib
import re
import json
```

Import selenium for browser automation. Help Docs for installing Selenium - https://pypi.org/project/selenium/ Download GeckoDriver for Mozilla - https://github.com/mozilla/geckodriver/releases

In [3]:

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.common.exceptions import TimeoutException
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support.ui import Select
from selenium.webdriver.common.by import By
from selenium.webdriver.support import expected_conditions
from selenium.common.exceptions import NoSuchElementException
```

import time for handling time components

In [4]:

```
import time
```

call_limit variable to restrict 200 calls per hour and to resume from the same place after an hour

In [5]:

```
call_limit=0
```

x variable to which we keep appending the extracted data from API to make the final data

In [6]:

```
x='{"data":\n  {'
```

Restarts the browser each time Browser freezes. This function helps resume the data extraction from the same place.

In [7]:

```
def restart_browser(driver, email_id, password):
    time.sleep(5)
    driver.close()
    time.sleep(5)
    driver = webdriver.Firefox()

    #setting a time limit of 15 seconds for data extraction from a webpage
    driver.set_page_load_timeout(15)
    driver.implicitly_wait(5)

    #opens Facebook homepage and logs in automatically
    driver.get('https://www.facebook.com/')
    print "Opened facebook..."
    a = driver.find_element_by_id('email')
    a.send_keys(email_id)
    print "Email Id entered..."
    b = driver.find_element_by_id('pass')
    b.send_keys(password)
    print "Password entered..."
    c = driver.find_element_by_id('loginbutton')
    c.click()
    return driver
```

Takes values from the UI, processes it and outputs the final data as json format in a text file.

In [12]:

```
def user_values():

    token=Entry.get(E1)
    token=str(token)

    group_id=Entry.get(E2)
    group_id=str(group_id)

    email_id=Entry.get(E3)
    email_id=str(email_id)

    password=Entry.get(E4)
    password=str(password)

    #extract_posts is the function which returns all Facebook Group data from API.
    k=extract_posts(token, group_id)
    k=k+'\n}'

    #Selenium code to open facebook group and extract data
    posts=re.findall(r'(\{"post":(.|\n)*?)ZZENDZZAA1214', k, re.MULTILINE)
    driver = webdriver.Firefox()

    #setting a time limit of 15 seconds for data extraction from a webpage
    driver.set_page_load_timeout(15)
    driver.implicitly_wait(5)

    #opens Facebook homepage and logs in automatically
    driver.get('https://www.facebook.com/')
    print "Opened facebook..."
    a = driver.find_element_by_id('email')
    a.send_keys(email_id)
    print "Email Id entered..."
    b = driver.find_element_by_id('pass')
    b.send_keys(password)
    print "Password entered..."
    c = driver.find_element_by_id('loginbutton')
    c.click()


    #Code to extract user ids, seen counts, reactions etc from posts/comments/replies
    for i in range(len(posts)):

        post=posts[i][0]

        #Post opened
        post_id=re.findall(r'"post_id":\d+_(\d+)', post, re.MULTILINE)
        post_id=post_id[0]
        post_id=str(post_id)
        post_url="https://www.facebook.com/groups/"+group_id+"/"+post_id+"/"
        finished=0
        while finished == 0:
            try:
                driver.get(post_url)
                finished=1

            except Exception as e:
                driver=restart_browser(driver, email_id, password)

        post_html_code=(driver.page_source).encode('utf-8')


        #post_user_id extracted
        post_user_id=re.findall(r'(>FROM NOTIFICATIONS<.*?</form>)', post_html_code, re.MULTILINE)
        post_user_id=str(post_user_id)
        post_user_id=re.findall(r'span class="fwn fcg".*?ajaxify="/groups/member_bio/bio_dialog/\?g
roup_id=\d+\&amp;member_id=(\d+)\&', post_user_id, re.MULTILINE)
        post_user_id=post_user_id[0]
        post_user_id=str(post_user_id)
        post_user_id='"post_user_id":'+post_user_id+','


        #seen count page opened and seen count, seen_user_ids extracted (seen count+user_ids of
users who have seen the post.)
        unique_id=re.findall(r'"USER_ID":"(\d+)"', post_html_code, re.MULTILINE)
```

```
        unique_id=unique_id[0]
        unique_id=str(unique_id)

        seen_url='https://www.facebook.com/ufi/group/seenby/profile/browser/?id='+post_id+'&av='+ur
ique_id
        finished=0
        while finished == 0:
            try:
                driver.get(seen_url)
                finished=1

            except Exception as e:
                driver=restart_browser(driver, email_id, password)

        seen_html_code=(driver.page_source).encode('utf-8')

        seen_count=re.findall(r'div class="_4neg">SEEN\ (\d+)</div>', seen_html_code, re.MULTI
LINE)

        if len(seen_count)>0:
            seen_count=seen_count[0]
            seen_count=str(seen_count)

            seen_user_ids=re.findall(r'(id="groups_seen_by_profile_browser_seen">.*?</ul>)', seen_h
tml_code, re.MULTILINE)
            seen_user_ids=seen_user_ids[0]
            seen_user_ids=re.findall(r'location=profile_browser"\s*data-
hovercard="/ajax/hovercard/user\.php\?id=(\d+)\&', seen_user_ids, re.MULTILINE)
            seen_user_ids=','.join([str(x) for x in seen_user_ids])

            seen_count='"seen_count":'+seen_count+','
            seen_count=seen_count+'"seen_user_ids":['+seen_user_ids+'],'

        else:
            seen_count=str(0)
            seen_count='"seen_count":'+seen_count+','


        #reactions page opened and reactions_user_ids extracted(reactions+user_id pairs) for each
post
        reactions_url='https://www.facebook.com/ufi/reaction/profile/browser/?ft_ent_identifier='+p
ost_id+'&av='+unique_id

        finished=0
        while finished == 0:
            try:
                driver.get(reactions_url)
                finished=1

            except Exception as e:
                driver=restart_browser(driver, email_id, password)

        reactions_html_code=(driver.page_source).encode('utf-8')

        reactions_user_ids=re.findall(r'(>People Who Reacted<.*?role="complementary")',
reactions_html_code, re.MULTILINE)
        reactions_user_ids=reactions_user_ids[0]
        reactions_user_ids=re.findall(r'div class="_3p56">(.{1,10})</div>.{1,400}data-
hovercard="/ajax/hovercard/user\.php\?id=(\d+)\&', reactions_user_ids, re.MULTILINE)

        reactions_user_ids=str(reactions_user_ids).strip('[]')
        if reactions_user_ids == "":
            reactions_user_ids="null"
        reactions_user_ids='"reactions_user_ids":['+reactions_user_ids+']'


        #adding post_user_id+seen_count+reactions_user_ids along with post_id to output data
        b2=str(post_id)
        z2=b2+','+post_user_id+seen_count+reactions_user_ids
        k=re.sub(b2, z2, k)


        #comment/reply opened
        comment_or_reply_id=re.findall(r'"(reply_id|comment_id)":(\d+)', post, re.MULTILINE)

        for i in range(len(comment_or_reply_id)):
```

```
            #Comment or Reply page html extracted
            ids=str(comment_or_reply_id[i][1])
            comment_or_reply_url="https://www.facebook.com/groups/"+group_id+"/"+post_id+"/?comment
_id="+ids

            finished=0
            while finished == 0:
                try:
                    driver.get(comment_or_reply_url)
                    finished=1

                except Exception as e:
                    driver=restart_browser(driver, email_id, password)

            comment_or_reply_html_code=(driver.page_source).encode('utf-8')

            #comment_or_reply_user_id extracted
            tag='legacyid:"'+ids+'",author:"(\d+)"'
            comment_or_reply_user_id=re.findall(tag, comment_or_reply_html_code, re.MULTILINE)
            comment_or_reply_user_id=comment_or_reply_user_id[0]
            comment_or_reply_user_id=str(comment_or_reply_user_id)
            comment_or_reply_user_id='"user_id":'+comment_or_reply_user_id+','


            #reactions page opened and reactions_user_ids extracted(reactions+user_id pairs) for
each comment/reply
            reactions_url='https://www.facebook.com/ufi/reaction/profile/browser/?ft_ent_identifie
='+ids+'&av='+unique_id

            finished=0
            while finished == 0:
                try:
                    driver.get(reactions_url)
                    finished=1

                except Exception as e:
                    driver=restart_browser(driver, email_id, password)

            reactions_html_code=(driver.page_source).encode('utf-8')
            reactions_user_ids=re.findall(r'(>People Who Reacted<.*?role="complementary")', reactic
ns_html_code, re.MULTILINE)
            reactions_user_ids=reactions_user_ids[0]
            reactions_user_ids=re.findall(r'div class="_3p56">(.{1,10})</div>.{1,400}data-
hovercard="/ajax/hovercard/user\.php\?id=(\d+)\&', reactions_user_ids, re.MULTILINE)

            reactions_user_ids=str(reactions_user_ids).strip('[]')
            if reactions_user_ids == "":
                reactions_user_ids="null"
            reactions_user_ids='"reactions_user_ids":['+reactions_user_ids+']'


            #adding comment_or_reply_user_id+reactions_user_ids along with post_id to output data
            b2=str(ids)
            z2=b2+','+comment_or_reply_user_id+reactions_user_ids
            k=re.sub(b2, z2, k)


    #After successful data extraction, a message box is shown to the user.
    k=k.replace("ZZENDZZAA1214", "")
    tkMessageBox.showinfo("success", "Data has been collected in data.txt")
    data_file = open("alldata.txt", "w")
    data_file.write(k)
    data_file.close()

    #Browser closed automatically after data extraction.
    time.sleep(5)
    driver.close()
    time.sleep(5)
```

extract_posts function extracts all posts' content and further calls extract_comments and extract_replies function to extract comments and replies content as well. It returns -

1. Combined content of posts/comments/replies.
2. A unique id for each post/comment/reply.
3. Their corresponding timestamps.

An HTTPS GET request is sent to the Facebook for this data extraction. This complete task is done by using Facebook API.

Note: content here means the post/comment/reply itself.

In [13]:

```
def extract_posts(token, group_id):

    global x
    global call_limit
    headers = {
                'access_token': token,
                }

    #Request Parameters
    params = urllib.urlencode({
                'fields': 'feed',
                'access_token': token,
                })

    #Establish https connection with Facebook API.
    #Checking call limit of 195 calls. Facebook allows 200 calls/hour. But we considered 195 calls
to be on safe side.
    #Once the call limit is reached, it resumes API calls again after 3660 seconds or 61 minutes.
    try:
        call_limit=call_limit+1
        if call_limit == 195:
            time.sleep(3660)
        conn = httplib.HTTPSConnection('graph.facebook.com')
        conn.request("GET", ("/"+group_id+"?%s") % params, "{body}", headers)
        response = conn.getresponse()
        data = response.read()
        conn.close()
    except Exception as e:
        print("[Errno {0}] {1}".format(e.errno, e.strerror))


    data = json.loads(data)

    m=len(data["feed"]["data"])
    if m>0:
        for n in range(len(data["feed"]["data"])):
            try:
                k=data["feed"]["data"][n]["message"]
                k=k.encode('utf-8')
                if k:
                    time_stamp=str(data["feed"]["data"][n]["updated_time"])
                    post_id=str(data["feed"]["data"][n]["id"])
                    x=x+'\n\n
{"post":"'+k+'","'+'"post_id":'+post_id+','+'"time_stamp":'+time_stamp+'},'
                    try:
                        extract_comments(token, post_id)
                    except Exception as e:
                        pass
                    x=x+"\nZZENDZZAA1214"
            except Exception as e:
                continue
        x=x+'\n  }'

    return x
```

extract_comments function extracts all comments' content and further calls extract_replies function to extract replies content as well. It returns -

1. Combined content of comments and replies for any particular post.
2. A unique id for each comment and reply.
3. Their corresponding timestamps.

An HTTPS GET request is sent to the Facebook for this data extraction. This complete task is done by using Facebook API.

In [14]:

```
def extract_comments(token, post_id):
```

```
    global x
    global call_limit
    headers = {
                'access_token': token,
                }

    #Request Parameters
    params = urllib.urlencode({
                'fields': 'comments',
                'access_token': token,
                })

    #Establish https connection with Facebook API.
    #Checking call limit of 195 calls. Facebook allows 200 calls/hour. But we considered 195 calls
to be on safe side.
    #Once the call limit is reached, it resumes API calls again after 3660 seconds or 61 minutes.
    try:
        call_limit=call_limit+1
        if call_limit == 195:
            time.sleep(3660)
        conn = httplib.HTTPSConnection('graph.facebook.com')
        conn.request("GET", ("/"+post_id+"?%s") % params, "{body}", headers)
        response = conn.getresponse()
        data = response.read()
        #print(data)
        conn.close()
    except Exception as e:
        print("[Errno {0}] {1}".format(e.errno, e.strerror))


    data = json.loads(data)
    m=len(data["comments"]["data"])
    if m>0:
        x=x+'\n'+'    {'
        for n in range(len(data["comments"]["data"])):
            try:
                k=data["comments"]["data"][n]["message"]
                k=k.encode('utf-8')
                if k:
                    time_stamp=str(data["comments"]["data"][n]["created_time"])
                    comment_id=str(data["comments"]["data"][n]["id"])

x=x+'{"comment":"'+k+'",'+'"comment_id":'+comment_id+','+'"time_stamp":'+time_stamp+'},'
                    x=x+'\n'+'     '
                    try:
                        extract_replies(token, comment_id)
                    except Exception as e:
                        pass
            except Exception as e:
                continue

        x=x+'}'

    return x
```

extract_replies function extracts all replies' content. It returns -

1. Content of replies for any particular post/comment.
2. A unique id for each reply.
3. Their corresponding timestamps.

An HTTPS GET request is sent to the Facebook for this data extraction. This complete task is done by using Facebook API.

In [15]:

```
def extract_replies(token, comment_id):

    global x
    global call_limit
    headers = {
                'access_token': token,
                }

    #Request Parameters
```

```
    params = urllib.urlencode({
                'fields': 'comments',
                'access_token': token,
                })

    #Establish https connection with Facebook API.
    #Checking call limit of 195 calls. Facebook allows 200 calls/hour. But we considered 195 calls
to be on safe side.
    #Once the call limit is reached, it resumes API calls again after 3660 seconds or 61 minutes.
    try:
        call_limit=call_limit+1
        if call_limit == 195:
            time.sleep(3660)
        conn = httplib.HTTPSConnection('graph.facebook.com')
        conn.request("GET", ("/"+comment_id+"?%s") % params, "{body}", headers)
        response = conn.getresponse()
        data = response.read()
        conn.close()
    except Exception as e:
        print("[Errno {0}] {1}".format(e.errno, e.strerror))


    data = json.loads(data)

    m=len(data["comments"]["data"])
    if m>0:
        x=x+'\n'+'      {'
        for n in range(len(data["comments"]["data"])):
            try:
                k=data["comments"]["data"][n]["message"]
                k=k.encode('utf-8')
                if k:
                    time_stamp=str(data["comments"]["data"][n]["created_time"])
                    reply_id=str(data["comments"]["data"][n]["id"])

x=x+'{"reply":"'+k+'",'+'"reply_id":'+reply_id+','+'"time_stamp":'+time_stamp+'},'
                    x=x+'\n'+'      '
            except Exception as e:
                continue

        x=x+'}'
    return x
```

Tkinter library provides support for user intreface. It takes -

1. Facebook Group Admin's "Email address"
2. Facebook Group Admin's "Password"
3. User Access Token
4. Group id from user in a textbox and passes it over to user_values() function to use further.

In [16]:

```
top = Tkinter.Tk()
L1 = Label(top, text="Facebook Data Extractor",).grid(row=0,column=1)

L2 = Label(top, text="Access Token",).grid(row=1,column=0)
E1 = Entry(top, bd =5)
E1.grid(row=1,column=1)

L3 = Label(top, text="FB Group Id",).grid(row=2,column=0)
E2 = Entry(top, bd =5)
E2.grid(row=2,column=1)

L4 = Label(top, text="Email ID",).grid(row=3,column=0)
E3 = Entry(top, bd =5)
E3.grid(row=3,column=1)

L5 = Label(top, text="Password",).grid(row=4,column=0)
E4 = Entry(top, bd =5)
E4.grid(row=4,column=1)

B=Button(top, text ="Collect Data",command=user_values).grid(row=5,column=1,)

Button(top, text="Quit", command=top.destroy).grid(row=6,column=1,)
top.mainloop()
```