Note: this notebook makes hundreds of thousands of API calls and takes hours, and will require you to have your own keys from ravelry. All the data herein is exported to a .csv file stored on github.

To get your own keys, see: https://www.ravelry.com/api.

In [ ]:
```
# import packages

import pandas as pd
import requests
import json
import random
import numpy
```

In [ ]:
```
# open credentials

with open('C:/Users/clare/Documents/Flatiron/PatternRecommender/.secrets/creds.json') as f:
    creds = json.load(f)
```

In [ ]:
```
# ravelry's api does not provide a list of users, but it has ~9,000,000 and they are sequentially
# numbered by order of membership and numbers are not reused.
# tried 500,000 integers betwen 1 and 12,000,000 until I had 100,000 users.

users = []

for i in random.sample(range(1, 12000000), 500000):
    try:
        url ='https://api.ravelry.com/people/' + str(i) +'.json'
        response = requests.get(url, auth=(creds['id'], creds['key']))
        users.append(response.json()['user']['username'])
        user = response.json()['user']['username']
    except ValueError:
        user = 0
        pass
    if len(set(users)) > 100000:
        break
    print(i, len(set(users)), user)
```

In [ ]:
```
users = list(set(users))
```

In [ ]:
```
parsed_data = []
```

In [ ]:
```
# use api to call each users projects if they are knitting projects (not crochet or weaving)
# and based on a pattern, not just knit from the imagination. parse the responses into a list of tuples.

for i, user in enumerate(users):

    url ='https://api.ravelry.com/projects/' + user + '/list.json?sort=completed_'
    response = requests.get(url, auth=(creds['id'], creds['key']))

    try:
        for project in response.json()['projects']:
            if project['craft_name'] == 'Knitting':
                if project['pattern_id'] != None:
                    pattern_url ='https://api.ravelry.com/patterns.json?ids=' + str(int(project['pattern_id']))
                    pattern_response = requests.get(pattern_url, auth=(creds['id'], creds['key']))
                    project_tuple = (user, project['completed'], project['rating'], project['status_name'],
                                     project['pattern_id'],
                                     pattern_response.json()['patterns'][str(int(project['pattern_id']))]['rating_average'],
                                     pattern_response.json()['patterns'][str(int(project['pattern_id']))]['rating_count'],
                                     [x['permalink'] for x in pattern_response.json()['patterns'][str(int(project['pattern_id']))]['pattern_attributes']],
                                     [x['permalink'] for x in pattern_response.json()['patterns'][str(int(project['pattern_id']))]['pattern_categories']])
                    parsed_data.append(project_tuple)

    except ValueError:
        pass
    print(i, len(parsed_data))
```

In [ ]:
```
len(parsed_data)
```

In [ ]:
```
# generate data frame from parsed data.

df = pd.DataFrame(parsed_data, columns = ['user', 'completed', 'rating', 'status', 'pattern_id', 'average_rating', 'rating_count', 'attributes', 'categories'])
```

In [ ]:
```
# export to CSV

df.to_csv('saved_100000_calls.csv', index =False)
```

In [ ]: