In [1]:

```
!pip install PyAthena
```

```
Requirement already satisfied: PyAthena in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (2.3.0)
Requirement already satisfied: botocore>=1.5.52 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from PyAthena) (1.20.83)
Requirement already satisfied: boto3>=1.4.4 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from PyAthena) (1.17.83)
Requirement already satisfied: tenacity>=4.1.0 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from PyAthena) (7.0.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from boto3>=1.4.4->PyAthena) (0.10.0)
Requirement already satisfied: s3transfer<0.5.0,>=0.4.0 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from boto3>=1.4.4->PyAthena) (0.4.2)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from botocore>=1.5.52->PyAthena) (1.26.5)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from botocore>=1.5.52->PyAthena) (2.8.1)
Requirement already satisfied: six>=1.5 in /home/ec2-user/anaconda3/envs/python3/lib/python3.6/site-packages (from python-dateutil<3.0.0,>=2.1->botocore>=1.5.52->PyAthena) (1.15.0)
```

In [2]:

```
import io
import boto3
import pandas as pd
import os
from pyathena import connect
from pyathena.pandas.util import as_pandas
from pyathena.pandas.cursor import PandasCursor
import time

## connect Athena with s3 & test
cursor = connect(s3_staging_dir='s3://query-results-bucket-athena-2021/',region_name='ap-northeast-2',cursor_class=PandasCursor).cursor()
```

In [3]:

```
#  1+2 -
def get_doc_num(ipc) :
    query = " SELECT dataset1.doc_num, dataset1.family_appno \
                FROM tm_database.df_us_family_parquet AS dataset1 \
                WHERE dataset1.doc_num IN ( \
                    SELECT dataset1.doc_no  \
                    FROM tm_database.df_us_datamart_assignee_parquet AS dataset1 \
                    WHERE dataset1.app_year_month >= '200501' AND dataset1.app_year_month <= '201812'  AND dataset1.new_ipc_code ='"+ipc+"' \
                ) \
                group by dataset1.doc_num, dataset1.family_appno"
    return cursor.execute(query).as_pandas()
```

In [4]:

```
#  3   _1
def get_appno() :
    query = "SELECT distinct dataset3.family_appno , dataset3.family_docno \
                FROM tm_database.df_us_family_parquet AS dataset3 \
                WHERE dataset3.family_docno = dataset3.doc_num"# and family_docno LIKE '%B2%' "

    return cursor.execute(query).as_pandas()
```

In [5]:

```
#  3   _2
def get_ctltr() :
    query = "select doc_num, new_citation_doc_num \
            from tm_database.df_us_ctltr_parquet \
            group by doc_num, new_citation_doc_num"

    return cursor.execute(query).as_pandas()
```

In [6]:

```
#get raw data
def get_raw_data(ipc):
    raw_doc_num = get_doc_num(ipc)
    raw_appno = get_appno()
```

```
    raw_ctltr = get_ctltr()

    return raw_doc_num, raw_appno, raw_ctltr
```

In [7]:

```
#doc no     family_appno
def get_family_docno_by_doc_num(raw_doc_num, raw_appno):
    doc_family_merge = pd.merge(raw_doc_num[['doc_num']], raw_appno, left_on='doc_num', right_on = 'family_docno').drop_duplicates()

    #family appno  family_docno   (B2    )
    doc_appno_merge = pd.merge(doc_family_merge, raw_appno, how='left', on='family_appno')
    doc_appno_merge = doc_appno_merge[(doc_appno_merge.family_docno_y.str.contains('B2'))]

    return doc_appno_merge
```

In [8]:

```
#ctltr      doc_num or family_docno        ctltr doc_num count   (       )
def get_ctltr_doc_num_count(doc_appno_merge, raw_doc_num, raw_ctltr):
    ctltr_join_df = pd.merge(doc_appno_merge, raw_ctltr, left_on='family_docno_y', right_on='new_citation_doc_num').drop_duplicates()
    doc_num_df = pd.merge(raw_doc_num[['doc_num']], raw_ctltr, left_on='doc_num', right_on = 'new_citation_doc_num').drop_duplicates()

    ctltr_join_df = ctltr_join_df[['doc_num_x', 'doc_num_y']]
    ctltr_join_df = ctltr_join_df.rename(columns = {"doc_num_x": "doc_num", "doc_num_y": "result_doc_num"})
    doc_num_df = doc_num_df[['doc_num_x', 'doc_num_y']]
    doc_num_df = doc_num_df.rename(columns = {"doc_num_x": "doc_num", "doc_num_y": "result_doc_num"})
    new_citation_df = pd.concat([ctltr_join_df, doc_num_df])
    new_citation_df = new_citation_df.drop_duplicates()

    return new_citation_df
```

In [9]:

```
#family_count   citation_freq count    , sum
def create_final_set(raw_doc_num, new_citiation_df):
    family_count = raw_doc_num.groupby(by='doc_num', as_index=False).agg({'family_appno': pd.Series.nunique}).sort_values(by='family_appno',
ascending=False)
    family_count_result = family_count.rename(columns = {"family_appno": "family_counts"})
    new_doc_num_count = new_citiation_df.groupby(by='doc_num', as_index=False).agg({'result_doc_num': pd.Series.nunique})
    new_doc_num_count_result = new_doc_num_count.rename(columns = {"result_doc_num": "citation_freq"})

    final_result = pd.merge(family_count_result, new_doc_num_count_result, how="left", on='doc_num')
    final_result = final_result.fillna(0)
    final_result['citation_freq'] = final_result.citation_freq.astype(int)
    final_result['sum'] = final_result.sum(axis=1)
    final_result = final_result.sort_values(by='sum',ascending=False)

    return final_result
```

In [10]:

```
def file_save(ipc, final_result):
    if "/" in ipc:
        ipca=ipc.split("/")
    filename = "data/material_IPC_cagr30/" + ipca[0] + "_" + ipca[1]  + "_VIP(" + str(len(final_result)) + ")_family_citation_frequencies.csv"
    final_result.to_csv(filename, index=False)
```

In [11]:

```
# get applicant information from us_datamart
#def get_applicant_info_VIP(doc_num) :
#    query = " SELECT dataset1.doc_no, dataset1.applicant1, dataset1.applicant2, dataset1.applicant3, dataset1.applicant4, dataset1.applicant5 \
#             FROM tm_database.df_us_datamart_assignee_parquet AS dataset1 \
#             WHERE dataset1.doc_no = '"+doc_num+"' "
#    return cursor.execute(query).as_pandas()
```

In [12]:

```
# get applicant information from us_datamart  doc_no         in ('a', 'b', 'c',,,)
def get_applicant_info_VIP(doc_num) :
    query = " SELECT dataset1.doc_no, dataset1.applicant1, dataset1.applicant2, dataset1.neo_company_univ1, dataset1.neo_company_univ2 \
            FROM tm_database.df_us_datamart_assignee_parquet AS dataset1 \
            WHERE dataset1.doc_no in ("+doc_num+") "
    return cursor.execute(query).as_pandas()
```

In [ ]:

```
## signle IPC analysis
# ipc = "A61K45/06"
# start = time.time()
#raw_doc_num, raw_appno, raw_ctltr = get_raw_data(ipc)
#doc_appno_merge = get_family_docno_by_doc_num(raw_doc_num, raw_appno)
#new_citation_df = get_ctltr_doc_num_count(doc_appno_merge, raw_doc_num, raw_ctltr)
#final_result = create_final_set(raw_doc_num, new_citation_df)
#file_save(ipc, final_result)
#r_time = round((time.time()-start)/60, 2)
#print('Running time :', r_time, 'minutes')  */


## a number of IPC(s) analysis

# ipc_list = ['G06F17/30', 'B29C67/00']

#analysis_file_name = 'data/material_IPC_cagr50/' + '2005_to_2018_sub_class_ipc_no_63_top_500_pat_cagr_min0.5_max10_applicant_cagr_min0.5_max10_analysis_data.csv'

analysis_file_name = 'data/material_IPC_cagr30/' + '2005_to_2018_sub_class_ipc_no_63_top_500_pat_cagr_min0.3_max0.5_applicant_cagr_min0.3_max0.5_analysis_data.csv'

# analysis_file_name = 'data/' + '2005_to_2018_sub_class_ipc_no_63_top_500_pat_cagr_min0.15_max0.3_applicant_cagr_min0.15_max0.3_analysis_data.csv'

tg_ipc = pd.read_csv(analysis_file_name, index_col=0)

ipc_list = list(tg_ipc.index)

#ipc_list = list(tg_ipc.index)[0:1]

#ipc_list = ['G06F3/041', 'G06F3/0484', 'G06F3/01', 'H04N5/232', 'G06F3/0482']


top_rank = 100

for idx, ipc in enumerate(ipc_list):
    start = time.time()

    if idx == 0:
        raw_doc_num, raw_appno, raw_ctltr = get_raw_data(ipc)
    else:
        raw_doc_num = get_doc_num(ipc)

    doc_appno_merge = get_family_docno_by_doc_num(raw_doc_num, raw_appno)
    new_citation_df = get_ctltr_doc_num_count(doc_appno_merge, raw_doc_num, raw_ctltr)
    final_result = create_final_set(raw_doc_num, new_citation_df)
    file_save(ipc, final_result)

    final_result = final_result.reset_index(drop=True)
    VIP_doc_num_list = list(final_result['doc_num'])[0:top_rank]

    # get_applicant_info_VIP(doc_num)                    "'a', 'b', 'c',,,"
    (','.join(map("'{0}'".format, VIP_doc_num_list)))

    VIP_applicants_list = []
    temp_list = []

    top_final_result = final_result.iloc[0:top_rank].rename(columns={'doc_num':'doc_no'})
    top_applicants = pd.DataFrame([])


    applicants = get_applicant_info_VIP(','.join(map("'{0}'".format, VIP_doc_num_list)))
    applicants = applicants.drop_duplicates().fillna(-1)

    top_final_result = pd.merge(top_final_result, applicants, how='left', on='doc_no')

    top_final_result.reset_index(drop=True)

    if "/" in ipc:
        ipca=ipc.split("/")

    top_final_result.to_csv("data/material_IPC_cagr30/" + ipca[0] + "_" + ipca[1]  + "_VIP_" + "top" + str(top_rank) + "_family_citation_frequencies_applicants.csv")

    r_time = round((time.time()-start)/60, 2)
    print('Running time(',ipc,'): ', r_time, 'minutes')
```

```
Running time( A61K47/10 ):  6.46 minutes
Running time( C22C38/02 ):  2.26 minutes
```

Running time(C22C38/02): 2.26 minutes
Running time(C12N15/10): 2.23 minutes
Running time(A61K31/713): 2.26 minutes
Running time(A61K47/26): 2.04 minutes
Running time(A61Q19/08): 2.04 minutes
Running time(A61K47/02): 1.95 minutes
Running time(A61K8/92): 2.0 minutes
Running time(A61K31/167): 1.97 minutes
Running time(A61L27/54): 1.93 minutes
Running time(A61K39/39): 1.94 minutes
Running time(A61K47/18): 1.89 minutes
Running time(A61K35/28): 1.93 minutes
Running time(A61Q5/12): 1.95 minutes
Running time(A61Q19/10): 1.9 minutes
Running time(H01F27/29): 1.92 minutes
Running time(A61K9/51): 1.95 minutes
Running time(A61K8/04): 1.98 minutes
Running time(A61L27/36): 1.93 minutes
Running time(A61K47/22): 1.93 minutes
Running time(B01J20/30): 1.93 minutes
Running time(C12P19/14): 1.96 minutes
Running time(B01J21/06): 1.93 minutes
Running time(A61Q19/02): 1.88 minutes

In [ ]:

In [ ]:

In [ ]: