**from pandas import \*  
options.mode.chained\_assignment = None  
import matplotlib  
matplotlib.use('TkAgg')  
import numpy as np  
import seaborn as sns  
import matplotlib.pyplot as plt  
sns.set\_style("white")  
from incf.countryutils import transformations  
  
df\_artists=read\_csv('artist\_data.csv')  
art\_vars=['artist','artistRole','artistId','title','dateText','medium','creditLine','year','acquisitionYear',  
 'width','height','units']  
df\_artwork=read\_csv('artwork\_data.csv',na\_values=['no date'],usecols=art\_vars,low\_memory=False)  
df\_art\_credit=df\_artwork[df\_artwork.creditLine.notnull()]  
df\_art\_credit['acquisition']=df\_art\_credit.apply(lambda row: row.creditLine.split()[0].lower(),axis=1)  
df\_art\_credit.acquisition=df\_art\_credit.acquisition.replace({'given':'gift','offered':'accepted',  
 '[uncovered':'uncovered','acquired':'partial'})  
def find\_acquisition(entry):  
 if 'presented' in entry.lower():  
 return 'presented'  
 elif 'acquired' in entry.lower():  
 return 'acquired'  
df\_art\_credit.acquisition[df\_art\_credit.acquisition=='artist'] = \  
 df\_art\_credit[df\_art\_credit.acquisition=='artist'].apply(lambda row: find\_acquisition(row.creditLine),axis=1)  
df\_year\_known=df\_art\_credit[df\_art\_credit.year.notnull()]  
df\_year\_known.year=df\_year\_known.year.replace({'c.1997-9':'1998'})  
df\_year\_known.year=df\_year\_known.year.astype(np.int64)  
df\_year\_known\_small=df\_year\_known[(df\_year\_known.acquisition=='accepted') |  
 (df\_year\_known.acquisition=='bequeathed') |  
 (df\_year\_known.acquisition=='presented') |  
 (df\_year\_known.acquisition=='purchased')]  
df\_year\_known\_small=df\_year\_known\_small[df\_year\_known\_small.year<= df\_year\_known\_small.acquisitionYear].reset\_index()  
df\_year\_known\_small.acquisitionYear=df\_year\_known\_small.acquisitionYear.astype(np.int64)  
h = sns.FacetGrid(df\_year\_known\_small,col='acquisition',hue='acquisition',margin\_titles=True,  
 xlim=(1823, 2013),ylim=(1545, 2013),palette=sns.color\_palette("hls", 4))  
h.map(plt.scatter,'acquisitionYear','year',s=25,alpha=.25)  
for ax in h.axes.flat:  
 ax.plot((1823,2013),(1823,2013),c=".2",ls="--")  
h.set\_xlabels(label='Acquisition Year')  
h.set\_ylabels(label="Year of 'Completion'")  
h.set\_titles(['Accepted','Bequeathed','Presented','Purchased'])  
sns.plt.show()  
def replace\_country(entry):  
 if entry == 'Unknown':  
 return 'Unknown'  
 if ('United Kingdom' in entry) | ('Braintree' in entry) | ('Egremont' in entry) | ('Kensington' in entry) | \  
 ('Liverpool' in entry) | ('London' in entry) | ('Canterbury' in entry) | ('Plymouth' in entry) | \  
 ('Epsom' in entry) | ('Wimbledon' in entry) | ('Blackheath' in entry) | ('Bermondsey' in entry) | \  
 ('Douglas' in entry) | ('Melmerby' in entry) | ('Isle of Man' in entry) | ('Stoke on Trent' in entry) | \  
 ('Beckington' in entry) | ('Edinburgh' in entry) | ('Hertfordshire' in entry) | ('Bristol' in entry) | \  
 ('Rochdale' in entry) | ('Montserra' in entry) | ('Saint H\xc3\xa9lier' in entry):  
 return 'United Kingdom of Great Britain & Northern Ireland'  
 if ('United States' in entry) | ('Staten Island' in entry):  
 return 'United States of America'  
 if ('Polska' in entry) | ('Schlesien' in entry) | ('Niederschlesien' in entry):  
 return 'Poland'  
 if "Yisra'el" in entry:  
 return 'Israel'  
 if "Deutschland" in entry:  
 return 'Germany'  
 if 'Italia' in entry:  
 return 'Italy'  
 if 'Argentina' in entry:  
 return 'Argentina'  
 if ('Schweiz' in entry) | ('Solothurn' in entry):  
 return 'Switzerland'  
 if 'Suomi' in entry:  
 return 'Finland'  
 if 'Zhonghua' in entry:  
 return 'China'  
 if ('France' in entry) | ('Auteuil' in entry) | ('Charlieu' in entry):  
 return 'France'  
 if 'T\xc3\xbcrkiye' in entry:  
 return 'Turkey'  
 if 'Iraq' in entry:  
 return 'Iraq'  
 if 'Belgi\xc3\xab' in entry:  
 return 'Belgium'  
 if 'Rossiya' in entry:  
 return 'Russian Federation'  
 if 'Malaysia' in entry:  
 return 'Malaysia'  
 if 'Portugal' in entry:  
 return 'Portugal'  
 if 'Nederland' in entry:  
 return 'Netherlands'  
 if 'M\xc3\xa9xico' in entry:  
 return 'Mexico'  
 if 'Espa\xc3\xb1a' in entry:  
 return 'Spain'  
 if 'Brasil' in entry:  
 return 'Brazil'  
 if 'Ukrayina' in entry:  
 return 'Ukraine'  
 if 'Per\xc3\xba' in entry:  
 return 'Peru'  
 if 'Pakistan' in entry:  
 return 'Pakistan'  
 if 'Nihon' in entry:  
 return 'Japan'  
 if '\xc3\x8eran' in entry:  
 return 'Iran'  
 if 'Venezuela' in entry:  
 return 'Venezuela'  
 if 'Viet Nam' in entry:  
 return 'Vietnam'  
 if 'Rom\xc3\xa2nia' in entry:  
 return 'Romania'  
 if ('Australia' in entry) | ('Perth' in entry):  
 return 'Australia'  
 if "Al-Jaza'ir" in entry:  
 return 'Algeria'  
 if 'Canada' in entry:  
 return 'Canada'  
 if ('Sverige' in entry) | ('Stockholm' in entry):  
 return 'Sweden'  
 if '\xc3\x89ire' in entry:  
 return 'Ireland'  
 if 'New Zealand' in entry:  
 return 'New Zealand'  
 if 'Zambia' in entry:  
 return 'Zambia'  
 if 'Guyana' in entry:  
 return 'Guyana'  
 if 'Prathet Thai' in entry:  
 return 'Thailand'  
 if ('Beograd' in entry) | ('Novi Sad' in entry) | ('\xc5\xa0id' in entry):  
 return 'Serbia'  
 if ('Brno' in entry) | ('Cesk\xc3\xa1 Republika' in entry):  
 return 'Czech Republic'  
 if '\xc3\x96sterreich' in entry:  
 return 'Austria'  
 if 'South Africa' in entry:  
 return 'South Africa'  
 if 'Uganda' in entry:  
 return 'Uganda'  
 if 'Norge' in entry:  
 return 'Norway'  
 if 'Bharat' in entry:  
 return 'India'  
 if 'Bosna i Hercegovina' in entry:  
 return 'Bosnia and Herzegovina'  
 if 'Slovenija' in entry:  
 return 'Slovenia'  
 if 'Cuba' in entry:  
 return 'Cuba'  
 if 'Colombia' in entry:  
 return 'Colombia'  
 if 'Latvija' in entry:  
 return 'Latvia'  
 if 'Bulgaria' in entry:  
 return 'Bulgaria'  
 if 'Belarus' in entry:  
 return 'Belarus'  
 if 'Danmark' in entry:  
 return 'Denmark'  
 if 'Chile' in entry:  
 return 'Chile'  
 if 'Cameroun' in entry:  
 return 'Cameroon'  
 if "Al-Lubnan" in entry:  
 return 'Lebanon'  
 if 'Misr'in entry:  
 return 'Egypt'  
 if 'Makedonija' in entry:  
 return 'Macedonia'  
 if 'As-Sudan' in entry:  
 return 'Sudan'  
 if 'Eesti' in entry:  
 return 'Estonia'  
 if 'Slovensk\xc3\xa1 Republika' in entry:  
 return 'Slovakia (Slovak Republic)'  
 if 'B\xc3\xa9nin' in entry:  
 return 'Benin'  
 if 'Hrvatska' in entry:  
 return 'Croatia'  
 if 'Bahamas' in entry:  
 return 'Bahamas'  
 if 'Indonesia' in entry:  
 return 'Indonesia'  
 if 'Tanzania' in entry:  
 return 'Tanzania'  
 if 'Bangladesh' in entry:  
 return 'Bangladesh'  
 if 'Tunis' in entry:  
 return 'Tunisia'  
 if 'Magyarorsz\xc3\xa1g' in entry:  
 return "Hungary"  
 if 'Moldova' in entry:  
 return 'Moldova'  
 if 'Mauritius' in entry:  
 return 'Mauritius'  
 if ("Taehan Min'guk" in entry) | ("Choson Minjujuui In'min Konghwaguk" in entry):  
 return 'Korea'  
 if "Suriyah" in entry:  
 return 'Syrian Arab Republic'  
 if '\xc3\x8dsland' in entry:  
 return 'Iceland'  
 if 'Pilipinas' in entry:  
 return 'Philippines'  
 if 'Jamaica' in entry:  
 return 'Jamaica'  
 if 'Kenya' in entry:  
 return 'Kenya'  
 if 'Malta' in entry:  
 return 'Malta'  
 if 'Panam\xc3\xa1' in entry:  
 return 'Panama'  
 if 'Nicaragua' in entry:  
 return 'Nicaragua'  
 if 'Sri Lanka' in entry:  
 return 'Sri Lanka'  
 if 'Lietuva' in entry:  
 return 'Lithuania'  
 if 'Luxembourg' in entry:  
 return 'Luxembourg'  
 if 'Chung-hua Min-kuo' in entry:  
 return 'Taiwan'  
 if 'Lao' in entry:  
 return "Lao People's Democratic Republic"  
 if 'Shqip\xc3\xabria' in entry:  
 return 'Albania'  
 if 'Ell\xc3\xa1s' in entry:  
 return 'Greece'  
 if 'Charlotte Amalie' in entry:  
 return 'United States Virgin Islands'  
 return entry  
def find\_continent(country):  
 if country=='Unknown':  
 return 'Unknown'  
 else:  
 return transformations.cn\_to\_ctn(country)  
df\_artists[df\_artists.placeOfBirth.isnull()]='Unknown'  
df\_artists['country']=df\_artists.apply(lambda x: replace\_country(x.placeOfBirth),axis=1)  
df\_artists['continent']=df\_artists.apply(lambda x: find\_continent(x.country),axis=1)  
def get\_continent(artistId):  
 return np.array\_str(df\_artists['continent'][df\_artists['id']==artistId].values)[2:-2]  
df\_post\_1950=df\_year\_known\_small[(df\_year\_known\_small.year >= 1950)]  
df\_post\_1950['Artist Birth Country']=df\_post\_1950.apply(lambda x: get\_continent(x.artistId),axis=1)  
df\_post\_1950\_cut=df\_post\_1950[(df\_post\_1950['Artist Birth Country']!='Europe')]  
df\_post\_1950\_cut["Year Artwork 'Completed'"] = df\_post\_1950\_cut.year  
grouped=DataFrame({'pieces': df\_post\_1950\_cut.groupby(["Year Artwork 'Completed'",'Artist Birth Country']).size()}).reset\_index()  
grouped\_rect=grouped.pivot('Artist Birth Country',"Year Artwork 'Completed'",'pieces')  
grouped\_rect=grouped\_rect.fillna(0)  
sns.heatmap(grouped\_rect, yticklabels=['Unknown','Africa','Asia','North America','Oceania','South America'])  
plt.show()**