RECORD LINKAGE AND MEASUREMENT

By Maryah Garner and Siqi Wang

Table of Contents

- Set Up Environment
- Organization names
- PI Names
  - Cleaning PI Names
- Develop Measures
- Checkpoint 1
- Link Data on PI Name
- Develop measures for change over time
- Results
- Checkpoint 2

We are going to read in projects data from 2015, create measures, and then link it to the 'DF_2017.csv' data that has already been prepared for you using the RePORTER_PRJ_C_FY2017_new data. This is a non-standard approach to performing analytics for data with the same structure over different years.

We are using this approach to give you the extremely valuable experience of linking data without overwhelming you with different data sources. You will be able to use the code and skills developed in this notebook to link data from different sources in the future.

Set Up Environment

In [1]:
```
# general use imports
import pandas as pd
```

Read in Project Data

In [2]:
```
# Specify a path with the data folder
# Change "NAME" to your name as recorded on your computer
# path = '/Users/wsq/Desktop/Advanced Data Analytics and Evidence Building/PADM
Path = '/Users/wsq/Desktop/Advanced Data Analytics and Evidence Building/PADM-G
```

Read in Pre-processed 2017 Data

First, we will read in the Preprocessed 2017 (`DF_2017.csv`) data. If you still need to, make sure you download this data from Britespace and put it in your `PADM-GP_2505/Data` folder.

In [3]:
```
# Read-in the DF_2017 CSV file
DF_2017 = pd.read_csv(Path + '/DF_2017.csv')

# View the first 5 observations
DF_2017.head(5)
```

Out[3]:

	PI_NAME	Total_Projects_2017	Total_Cost_All_2017	Total_NCI_Projects_2017	NCI_Total_C
0	CARROLL, WILLIAM L.	28	1010125	28.0	
1	FISHER, EDWARD A	9	5139805	NaN	
2	BUYON, JILL P	8	3506062	NaN	
3	ROTROSEN, JOHN P	7	5786823	NaN	
4	AIFANTIS, IANNIS	6	2654279	6.0	2

Read in 2015 Projects Data

Next, we will read in the raw projects data from 2015. We will clean this data and use it to develop a data frame that has one observation per PI (for PIs from NYU), similar to the `DF_2017` seen above.

In [4]:
```
# Read-in a CSV file
grants_2015 = pd.read_csv(Path + '/Projects/RePORTER_PRJ_C_FY2015_new.csv', enc

# View the first 5 observations
grants_2015.head()
```

```
/var/folders/rl/4_s3mzkj6ll5h2_4qxy9sftw0000gn/T/ipykernel_30706/2757390899.p
y:2: DtypeWarning: Columns (8) have mixed types. Specify dtype option on impor
t or set low_memory=False.
  grants_2015 = pd.read_csv(Path + '/Projects/RePORTER_PRJ_C_FY2015_new.csv',
encoding='latin-1')
```

Out[4]:

	APPLICATION_ID	ACTIVITY	ADMINISTERING_IC	APPLICATION_TYPE	ARRA_FUNDED	AWARI
0	8857469	T32	EY	5.0	N	
1	8931014	R03	HD	5.0	N	
2	8849766	R37	HD	5.0	N	
3	8900204	U59	EH	5.0	N	
4	8908443	F32	AI	1.0	N	

5 rows × 46 columns

Organization Names

We want to see how the organizations are recorded in the data

First, we will select variables that will help us understand better how organizations are recorded in the data and then look at the data.

In [5]:
```
#View the first 5 observations for select varables
grants_2015[['APPLICATION_ID', 'ORG_DUNS', 'ORG_IPF_CODE', 'ORG_NAME', 'ORG_STA
```

Out[5]:

	APPLICATION_ID	ORG_DUNS	ORG_IPF_CODE	ORG_NAME	ORG_STATE
0	8857469	42250712	6463801.0	UNIVERSITY OF PENNSYLVANIA	PA
1	8931014	4868105	6939101.0	RESEARCH TRIANGLE INSTITUTE	NC
2	8849766	120989983	1742101.0	WHITEHEAD INSTITUTE FOR BIOMEDICAL RES	MA
3	8900204	929922664	6963501.0	RHODE ISLAND STATE DEPT OF HEALTH	RI
4	8908443	804355790	577507.0	UNIVERSITY OF CALIFORNIA, SAN DIEGO	CA

Subsetting string

We will subset the grants_2015 for projects from NYU

- In order to use the str.contains function, we first need to convert the `ORG_NAME` variable into a string

In [6]:
```
# Convert ORG_NAME into a string variable
grants_2015['ORG_NAME'] = grants_2015['ORG_NAME'].astype(str)
grants_2015['ORG_NAME'].head(2)
```

Out[6]:
```
0     UNIVERSITY OF PENNSYLVANIA
1    RESEARCH TRIANGLE INSTITUTE
Name: ORG_NAME, dtype: object
```

In [7]:
```
# Subsetting by a keyword in a string using str.contains()
# Select all observations that have "NEW YORK UNIVERSITY" in the organization n
df_NYU = grants_2015[grants_2015['ORG_NAME'].str.contains('NEW YORK UNIVERSITY'


# View the first 2 observations
df_NYU.head(2)
```

Out[7]:

	APPLICATION_ID	ACTIVITY	ADMINISTERING_IC	APPLICATION_TYPE	ARRA_FUNDED	AWA
18	8844008	R13	HS	1.0	N	
180	8843273	T32	HD	5.0	N	

2 rows × 46 columns

In [8]:
```
# View the first 5 observations of select variables
df_NYU[['APPLICATION_ID', 'ORG_DUNS', 'ORG_IPF_CODE', 'ORG_NAME', 'ORG_STATE']]
```

Out[8]:

	APPLICATION_ID	ORG_DUNS	ORG_IPF_CODE	ORG_NAME	ORG_STATE
18	8844008	41968306	5998301.0	NEW YORK UNIVERSITY	NY
180	8843273	121911077	5998304.0	NEW YORK UNIVERSITY SCHOOL OF MEDICINE	NY
301	8926934	41968306	5998301.0	NEW YORK UNIVERSITY	NY
361	9001539	121911077	5998304.0	NEW YORK UNIVERSITY SCHOOL OF MEDICINE	NY
390	8820075	41968306	5998301.0	NEW YORK UNIVERSITY	NY

As you can see NEW YORK UNIVERSITY and NEW YORK UNIVERSITY SCHOOL OF MEDICINE are considered different organizations in this data. Depending on your research question, it might be more appropriate to consider them as the same organization. You will find many organizations exhibit simular patterns, so you need to be thoughtful as to what you consider to be a single organization in your research projects.

We will group ORG_NAME to get the number of Projects that were awarded for each. This will allow us to see the names of all ORG_NAMEs that have "NEW YORK UNIVERSITY" in its name. In this case there are only 2, but some organizations have more then two ORG_NAMEs which might not all displayed using the head command, so you need to be careful.

In [9]:
```
# calculate how many projects (unique application ids) that were awarded by eac
df_NYU.groupby('ORG_NAME')['APPLICATION_ID'].nunique().sort_values(ascending=Fa
```

Out[9]:
```
ORG_NAME
NEW YORK UNIVERSITY SCHOOL OF MEDICINE    501
NEW YORK UNIVERSITY                       153
Name: APPLICATION_ID, dtype: int64
```

Replace ORG_NAME

For the purpose of this notebook, we will treat both NEW YORK UNIVERSITY and NEW YORK UNIVERSITY SCHOOL OF MEDICINE as the same organization.

We will create a list of possible names for NEW YORK UNIVERSITY and then replace all Org names in that list with 'NEW YORK UNIVERSITY'.

In [10]:
```
# Save a list of possible names for the organization (NYU in this case)
possible_NYU_names = ['NEW YORK UNIVERSITY', 'NEW YORK UNIVERSITY SCHOOL OF MEI

# Replace any instance you see a string in the list of possible_org_names with
df_NYU = df_NYU.replace(possible_NYU_names, 'NEW YORK UNIVERSITY')

# recalculate how many projects that were awarded by each ORG_NAME
df_NYU.groupby('ORG_NAME')['APPLICATION_ID'].nunique().sort_values(ascending=Fa
```

Out[10]:
```
ORG_NAME
NEW YORK UNIVERSITY    654
Name: APPLICATION_ID, dtype: int64
```

In your projects, you can create a list like we did for NYU, for each group of ORG_NAMEs you want to consolidate into a single organization. Then use the `replace` function for each of your created lists.

PI Names

Recall our goal is to develop an analytic dataframe that we can link to the `DF_2017` data, linking on PI_NAME. To do this, we need to understand how the PI names are recorded.

- We will begin by counting the number of projects for each of the unique `PI_NAMEs`. We will then carefully examine the `PI_NAMEs` in the output.

In [11]:
```
# Calculate how many Projects each unique observation for PI_NAMEs has
df_NYU_CI = df_NYU.groupby(['PI_NAMEs','PI_IDS'])['APPLICATION_ID'].nunique().s

# View the first 10 observations
df_NYU_CI.head(10)
```

Out[11]:

	PI_NAMEs	PI_IDS	APPLICATION_ID
0	CARROLL, WILLIAM L.;	1897735;	27
1	COSTA, MAX ;	1884428;	11
2	MOVSHON, J ANTHONY;	1965976;	6
3	RUDY, BERNARDO ;	1867106;	5
4	WISNIEWSKI, THOMAS M;	1866058;	5
5	FISHELL, GORDON J;	1899280;	5
6	WU, XUE-RU ;	6115649;	4
7	OGEDEGBE, OLUGBENGA G. (contact); WILLIAMS, OL...	2667413 (contact); 8840097;	4
8	DESPLAN, CLAUDE ;	2452885;	4
9	NEEL, BENJAMIN G.;	1863874;	4

As you can see from the output in row seven above (or if you read the codebook), sometimes more than one PI exists in an award. Thus we cannot treat each unique value recorded as the PI_NAMEs as a unique PI. We will look at all of the projects OGEDEGBE, OLUGBENGA G is a PI on, to exemplify this further.

In [12]:
```
# Show full text in a cell
pd.set_option('display.max_colwidth', -1)

# Show all PI_NAMEs which contain name "OLUGBENGA"
df_NYU_CI[df_NYU_CI['PI_NAMEs'].str.contains('OLUGBENGA', na = False)]
```

```
/var/folders/rl/4_s3mzkj6ll5h2_4qxy9sftw0000gn/T/ipykernel_30706/1312624811.py:2: FutureWarning: Passing a negative integer is deprecated in version 1.0 and will not be supported in future version. Instead, use None to not limit the column width.
  pd.set_option('display.max_colwidth', -1)
```

Out[12]:

	PI_NAMEs	PI_IDS	APPLICATION_ID
7	OGEDEGBE, OLUGBENGA G. (contact); WILLIAMS, OLAJIDE ;	2667413 (contact); 8840097;	4
70	OGEDEGBE, OLUGBENGA G.;	2667413;	2
79	JEAN-LOUIS, GIRARDIN (contact); OGEDEGBE, OLUGBENGA G.;	1873301 (contact); 2667413;	2
148	OGEDEGBE, OLUGBENGA G.; WALL, STEPHEN P (contact);	2667413; 7897936 (contact);	1
321	AMOAH, ALBERT GEORGE BAIDOE; BINKA, FRED NEWTON; COOPER, RICHARD STANLEY; OGEDEGBE, OLUGBENGA G. (contact);	10720990; 10523236; 8225116; 2667413 (contact);	1
441	JEAN-LOUIS, GIRARDIN (contact); OGEDEGBE, OLUGBENGA G.; WATKINS, BEVERLY-XAVIERA ;	1873301 (contact); 2667413; 7062916;	1

As you can see OGEDEGBE, OLUGBENGA G was a PI on 11 projects, not just the two projects that are associated with the string that only contains his name

Cleaning PI Names

To clean the PI names, we will split up the names when more than one is recorded and then create a new column with a single PI for each observation.

Transform each element of a list-like to a row, replicating index values

We will create a temporary dataframe with an index number and a single PI name for each observation. The index number will be repeated for each PI that is split up from a list of PIs in the PI_NAMEs variable.

- We will use the str.split function to split up the PI names, so we first need to convert PI_NAMEs into a string variable
- We will then use the `str.split` function to separate the list of names into a different element for each PI. We are splitting the string at each `;`
- We use the `explode` function to transform each element into a row, replicating index values

In [13]:
```
# First we need to convert ORG_NAME into a string variable
df_NYU['PI_NAMEs'] = df_NYU['PI_NAMEs'].astype(str)
```

In [14]:
```
# Make a temperary dataframe that creates an observation for each PI.
# Use the explode function to split the PI_Names at the ;

temp = df_NYU['PI_NAMEs'].str.split(';').explode().reset_index()
```

```
# Rename the PI_NAMEs variable
temp = temp.rename(columns = {'PI_NAMEs': 'PI_NAME'})

# look at the first 5 observations
temp.head()
```

Out[14]:

	index	PI_NAME
0	18	CIOTOLI, CARLO
1	18	
2	180	TREISMAN, JESSICA E
3	180	
4	301	DAW, NATHANIEL DOUGLASS (contact)

Remove observations with empty strings

Because we split the PI_NAMEs using a `;`, and there is a `;` at the end of every PI (even those without a name proceeding it in the string), we created many observations with an empty string for PI_NAME. We will proceed with removing the empty string.

In [15]:
```
# Only keep observations in the temp dataframe where PI_NAME is not an empty st
temp = temp[temp['PI_NAME'] !=""]

# View the first 4 observations
temp.head(4)
```

Out[15]:

	index	PI_NAME
0	18	CIOTOLI, CARLO
2	180	TREISMAN, JESSICA E
4	301	DAW, NATHANIEL DOUGLASS (contact)
5	301	SHOHAMY, DAPHNA

Remove undesired section of the string

We are almost there, however we still need to remove the `(contact)` part of the string from some of the names; otherwise PI_names like `DAW, NATHANIEL DOUGLASS (contact)` will not link to `DAW, NATHANIEL DOUGLASS`

In [16]:
```
# For the the PI_NAME variable, use the str.replace fundtion to replace ` (cont
temp['PI_NAME'] = temp['PI_NAME'].str.replace(' \(contact\)', '')

temp.head()
```

```
/var/folders/rl/4_s3mzkj6ll5h2_4qxy9sftw0000gn/T/ipykernel_30706/1310223397.p
y:2: FutureWarning: The default value of regex will change from True to False
in a future version.
  temp['PI_NAME'] = temp['PI_NAME'].str.replace(' \(contact\)', '')
```

Out[16]:

	index	PI_NAME
0	18	CIOTOLI, CARLO
2	180	TREISMAN, JESSICA E
4	301	DAW, NATHANIEL DOUGLASS
5	301	SHOHAMY, DAPHNA
7	361	BURDEN, STEVEN

Merge dataframes

Now that we have a clean list of PIs, we will merge this temporary dataframe with df_NYU. After doing so, we will see the new PI_NAME variable as well as the old PI_NAMEs variable. Note all other variables will repeat for each of the separated PI names.

First, let's look at the df_NYU dataframe to make sure we are ready to merge on the index.

In [17]:
```
# view the select variables for the firts 4 observations
df_NYU[['APPLICATION_ID', 'PI_NAMEs','FULL_PROJECT_NUM', 'PROJECT_TITLE','PROJE
```

Out[17]:

	APPLICATION_ID	PI_NAMEs	FULL_PROJECT_NUM	PROJECT_TITLE	PROJECT_STA
18	8844008	CIOTOLI, CARLO ;	1R13HS022520-01A1	Quality Improvement in College Health Conference	8/1/2(
180	8843273	TREISMAN, JESSICA E;	5T32HD007520-17	TRAINING PROGRAM IN DEVELOPMENTAL GENETICS	5/1/19
301	8926934	DAW, NATHANIEL DOUGLASS (contact); SHOHAMY, DAPHNA ;	5R01DA038891-02	CRCNS: Computational and neural mechanisms of memory-guided decisions	9/15/2(
361	9001539	BURDEN, STEVEN ;	1R01AG051490-01	THE ROLE OF AGRIN/LRP4/MUSK/DOK-7 SIGNALING IN DISASSEMBLY OF NEUROMUSCULAR SYNAPSES DURING AGING.	9/30/2(
390	8820075	HOYT VON TRAPP, GARDINER ;	5F31DC013502-02	Impact of early hearing loss on cortical encoding during perceptual task performa	3/3/2(

reset the index

Note that the index reflects the column number from the full dataframe, and the column number lines up with the index numbers from the temperary dataframe. Thus, we need to reset the index before merging.

In [18]:
```
# Reset the index
df_NYU2 = df_NYU.reset_index()

# view the select variables for the first 4 observations
df_NYU2[['index', 'APPLICATION_ID', 'PI_NAMEs','FULL_PROJECT_NUM', 'PROJECT_TIT
```

Out[18]:

	index	APPLICATION_ID	PI_NAMEs	FULL_PROJECT_NUM	PROJECT_TITLE	PROJECT_
0	18	8844008	CIOTOLI, CARLO ;	1R13HS022520-01A1	Quality Improvement in College Health Conference	
1	180	8843273	TREISMAN, JESSICA E;	5T32HD007520-17	TRAINING PROGRAM IN DEVELOPMENTAL GENETICS	
2	301	8926934	DAW, NATHANIEL DOUGLASS (contact); SHOHAMY, DAPHNA ;	5R01DA038891-02	CRCNS: Computational and neural mechanisms of memory-guided decisions	9/
3	361	9001539	BURDEN, STEVEN ;	1R01AG051490-01	THE ROLE OF AGRIN/LRP4/MUSK/DOK-7 SIGNALING IN DISASSEMBLY OF NEUROMUSCULAR SYNAPSES DURING AGING.	9/
4	390	8820075	HOYT VON TRAPP, GARDINER ;	5F31DC013502-02	Impact of early hearing loss on cortical encoding during perceptual task performa	

In [19]:
```
# Merge this temperary dataframe with df_NYU

df_NYU3 = df_NYU2.merge(temp, on = 'index')


# view the select variables for the firts 4 observations
df_NYU3[['APPLICATION_ID', 'PI_NAMEs', 'PI_NAME','FULL_PROJECT_NUM', 'PROJECT_T
```

Out[19]:

	APPLICATION_ID	PI_NAMEs	PI_NAME	FULL_PROJECT_NUM	PROJECT_TITLE	PRO
0	8844008	CIOTOLI, CARLO ;	CIOTOLI, CARLO	1R13HS022520-01A1	Quality Improvement in College Health Conference	
1	8843273	TREISMAN, JESSICA E;	TREISMAN, JESSICA E	5T32HD007520-17	TRAINING PROGRAM IN DEVELOPMENTAL GENETICS	
2	8926934	DAW, NATHANIEL DOUGLASS (contact); SHOHAMY, DAPHNA ;	DAW, NATHANIEL DOUGLASS	5R01DA038891-02	CRCNS: Computational and neural mechanisms of memory-guided decisions	
3	8926934	DAW, NATHANIEL DOUGLASS (contact); SHOHAMY, DAPHNA ;	SHOHAMY, DAPHNA	5R01DA038891-02	CRCNS: Computational and neural mechanisms of memory-guided decisions	
4	9001539	BURDEN, STEVEN ;	BURDEN, STEVEN	1R01AG051490-01	THE ROLE OF AGRIN/LRP4/MUSK/DOK-7 SIGNALING IN DISASSEMBLY OF NEUROMUSCULAR SYNAPSES DURING AGING.	

Develop Measures

Using the 2015 data, we want to create measures at the PI level that will allow us to research how the `Cancer Moonshot` influenced the people working on cancer research.

All Projects

- The number of projects
- The total value all of projects

All NCI Projects

- Number of projects from the National cancer institute
- Value of projects from the National cancer institute

New NCI Projects

- Number of new projects from the National cancer institute
- Value of new projects from the National cancer institute

Most Common CI for each PI?

All Projects

Number of Projects

In [20]:
```
# Calculate how many Projects each PI has
PI = df_NYU3.groupby(['PI_NAME'])['APPLICATION_ID'].nunique().sort_values(ascer

# Convert into a dataframe and reset index
PI = PI.to_frame().reset_index()

# Rename APPLICATION_ID to Total_Projects_2015
PI.rename(columns={'APPLICATION_ID':'Total_Projects_2015'}, inplace = True)

# View the first 2 observations
PI.head(2)
```

Out[20]:

	PI_NAME	Total_Projects_2015
0	CARROLL, WILLIAM L.	27
1	COSTA, MAX	12

Total Value of All Projects

In [21]:
```
# Calculate the sum of the total costs for each PI
Value_PI = df_NYU3.groupby('PI_NAME')['TOTAL_COST'].sum()

# Convert into a dataframe and reset index
Value_PI = Value_PI.to_frame().reset_index()

# Rename TOTAL_COST to Total_Cost_All_2015
Value_PI.rename(columns={'TOTAL_COST':'Total_Cost_All_2015'}, inplace = True)

# View the first 2 observations
Value_PI.head(2)
```

Out[21]:

	PI_NAME	Total_Cost_All_2015
0	BINKA, FRED NEWTON	222617.0
1	BRAITHWAITE, RONALD SCOTT	168491.0

In [22]:
```
# Look at how many rows and columns there are
Value_PI.shape
```

Out[22]:
```
(508, 2)
```

In [23]:
```
# Marge together  `PI` and `Value_PI on PI_NAME`, creating a new data frame cal
# Use an outer merge.
DF_2015 = pd.merge(PI, Value_PI, on='PI_NAME', how = 'outer')

# Look at how many rows and columns there are
DF_2015.shape
```

Out[23]:
```
(508, 3)
```

In [24]:
```
# View the first 5 observations
DF_2015.head()
```

Out[24]:

	PI_NAME	Total_Projects_2015	Total_Cost_All_2015
0	CARROLL, WILLIAM L.	27	432951.0
1	COSTA, MAX	12	2563657.0
2	OGEDEGBE, OLUGBENGA G.	7	3222047.0
3	WILLIAMS, OLAJIDE	7	2488067.0
4	MOVSHON, J ANTHONY	7	1968716.0

All Projects from the National Cancer Institute¶

Number of Projects from the National Cancer Institute

In [25]:
```
# Use a conditional statement do create a new dataframe were there are only pro
NCI = df_NYU3[df_NYU3['IC_NAME'] == 'NATIONAL CANCER INSTITUTE']
NCI.shape
```

Out[25]:
```
(119, 48)
```

In [26]:
```
NCI.to_csv(Path + "example_data.csv", encoding='utf8')
```

In [27]:
```
# Calculate how many NCI Projects each PI has
NCI_PI = NCI.groupby(['PI_NAME'])['APPLICATION_ID'].nunique().sort_values(ascer

# Convert into a dataframe and reset index
NCI_PI = NCI_PI.to_frame().reset_index()

# Let's correct the columns names, this shouldn't be APPLICATION_ID but Total_N
NCI_PI.rename(columns={'APPLICATION_ID':'Total_NCI_Projects_2015'}, inplace = T

# View the first 5 observations
NCI_PI.head()
```

Out[27]:

	PI_NAME	Total_NCI_Projects_2015
0	CARROLL, WILLIAM L.	27
1	NEEL, BENJAMIN G.	4
2	AIFANTIS, IANNIS	4
3	GOLD, LESLIE INA	3
4	WU, XUE-RU	3

In [28]:
```
# Look at how many rows and columns there are before merging
NCI_PI.shape
```

Out[28]:
```
(68, 2)
```

In [29]:
```
# Merge NCI_PI into the DF_2015 data frame
DF_2015 = pd.merge(DF_2015, NCI_PI, on='PI_NAME', how = 'outer')

# Look at how many rows and columns there are after merging
DF_2015.shape
```

Out[29]:
```
(508, 4)
```

Note, we used an outer merge, so we are maintaining the total number of observations from the `DF_2015` data frame. If we had used an `inner merge,` we would only capture the PIs in both dataframes. This would be bad, since it would be dropping all the PIs who are not on an NCI grant.

Total value of National Cancer Instute Projects

In [30]:
```
# Calculate the sum of the total NCI costs for each PI
NCI_Value = NCI.groupby('PI_NAME')['TOTAL_COST'].sum().sort_values(ascending=Fa

# Convert into a dataframe and reset index
NCI_Value = NCI_Value.to_frame().reset_index()

# Rename the colums New_TOTAL_COST
NCI_Value.rename(columns={'TOTAL_COST':'NCI_Total_Cost_2015'}, inplace=True)

# View the first 3 observations
NCI_Value.head(3)
```

Out[30]:

	PI_NAME	NCI_Total_Cost_2015
0	NEEL, BENJAMIN G.	3160475.0
1	AIFANTIS, IANNIS	1535056.0
2	WU, XUE-RU	1312471.0

In [31]:
```
# Look at how many rows and columns there are before merging
NCI_Value.shape
```

Out[31]:
```
(68, 2)
```

In [32]:
```
# Merge the NCI_Value into the DF_2015 data frame
DF_2015 = pd.merge(DF_2015, NCI_Value, on='PI_NAME', how = 'outer')

# Look at how many rows and columns there are after merging
DF_2015.shape
```

Out[32]:
```
(508, 5)
```

In [33]:
```
# View the first 3 observations
DF_2015.head(3)
```

Out[33]:

	PI_NAME	Total_Projects_2015	Total_Cost_All_2015	Total_NCI_Projects_2015	NCI_Total_
0	CARROLL, WILLIAM L.	27	432951.0	27.0	
1	COSTA, MAX	12	2563657.0	NaN	
2	OGEDEGBE, OLUGBENGA G.	7	3222047.0	NaN	

New grants from the National Cancer Institute¶

New projects

We need to identify all projects that started this year we will do this in 3 steps

1. Convert PROJECT_START into a date variable
2. Use the project start date to create a `project_start_year` column
3. Create a new data frame that only has a observations where the `PROJECT_START_YEAR` is 2015

In [34]:
```
# convert PROJECT_START into a date variable
NCI['PROJECT_START'] = pd.to_datetime(NCI['PROJECT_START'])

# Create a new year variable from the PROJECT_START
NCI['PROJECT_START_YEAR'] = NCI['PROJECT_START'].dt.year

# View the first 3 observarions for select variables
NCI[['PROJECT_START_YEAR', 'PROJECT_START']].head(3)
```

```
/var/folders/rl/4_s3mzkj6ll5h2_4qxy9sftw0000gn/T/ipykernel_30706/1589043985.p
y:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  NCI['PROJECT_START'] = pd.to_datetime(NCI['PROJECT_START'])
/var/folders/rl/4_s3mzkj6ll5h2_4qxy9sftw0000gn/T/ipykernel_30706/1589043985.p
y:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  NCI['PROJECT_START_YEAR'] = NCI['PROJECT_START'].dt.year
```

Out[34]:

	PROJECT_START_YEAR	PROJECT_START
8	1985.0	1985-09-30
22	2012.0	2012-04-23
23	2012.0	2012-04-23

In [35]:
```
# Create a new data frame that has a variable where the PROJECT_START_YEAR is 2
NCI_new = NCI[NCI['PROJECT_START_YEAR'] == 2015]

# View the first 3 observations for select variables
NCI_new[['PROJECT_START_YEAR', 'PROJECT_START']].head(3)
```

Out[35]:

	PROJECT_START_YEAR	PROJECT_START
78	2015.0	2015-08-12
146	2015.0	2015-05-01
168	2015.0	2015-05-05

In [36]:
```
NCI_new.shape
```

Out[36]:
```
(14, 49)
```

Number of New Grants from the National Cancer Institute

In [37]:
```
# Calculate how many new NCI Projects each PI has
NCI_new_PI = NCI_new.groupby(['PI_NAME'])['APPLICATION_ID'].nunique().sort_valu

# Convert into a dataframe and reset index
NCI_new_PI = NCI_new_PI.to_frame().reset_index()

# Let's correct the columns names, this shouldn't be APPLICATION_ID but Total_N
NCI_new_PI.rename(columns={'APPLICATION_ID':'Total_New_NCI_Projects_2015'}, inp

# View the first 3 observations
NCI_new_PI.head(3)
```

Out[37]:

	PI_NAME	Total_New_NCI_Projects_2015
0	AIFANTIS, IANNIS	1
1	BARCELLOS-HOFF, MARY HELEN	1
2	CARUCCI, JOHN A	1

In [38]:
```
# Look at how many rows and columns there are before merging
NCI_new_PI.shape
```

Out[38]:
```
(14, 2)
```

In [39]:
```
DF_2015 = pd.merge(DF_2015, NCI_new_PI, on='PI_NAME', how = 'outer')

# Look at how many rows and columns there are after merging
DF_2015.shape
```

Out[39]:
```
(508, 6)
```

In [40]:
```
# Look at the first 3 observations
DF_2015.head(3)
```

Out[40]:

	PI_NAME	Total_Projects_2015	Total_Cost_All_2015	Total_NCI_Projects_2015	NCI_Total_
0	CARROLL, WILLIAM L.	27	432951.0	27.0	
1	COSTA, MAX	12	2563657.0	NaN	
2	OGEDEGBE, OLUGBENGA G.	7	3222047.0	NaN	

Check point 1: (Assignment 3 due March 10th)

Add a column to the `DF_2015` dataframe that has the total value of New National Cancer Instute Projects (3points) Due

In [41]:

```
# Calculate the sum of the total new NCI costs for each PI
NCI_new_cost = NCI_new.groupby('PI_NAME')['TOTAL_COST'].sum().sort_values(ascer

# Convert into a dataframe and reset index
NCI_new_cost = NCI_new_cost.to_frame().reset_index()
NCI_new_cost

# Rename the colums NCI_New_TOTAL_COST
NCI_new_cost.rename(columns={'TOTAL_COST':'NCI_New_Total_Cost_2015'}, inplace=T
NCI_new_cost

# Outer merge DF2015 and NCI_new_cost (https://pandas.pydata.org/docs/reference

DF_2015 = pd.merge(NCI_new_cost,DF_2015, on='PI_NAME', how = 'outer')

DF_2015.head(15)
```

Out[41]:

	PI_NAME	NCI_New_Total_Cost_2015	Total_Projects_2015	Total_Cost_All_2015	Total_I
0	ZELENIUCH-JAQUOTTE, ANNE	788279.0	1	788279.0	
1	KIRCHHOFF, TOMAS	478926.0	2	663257.0	
2	AIFANTIS, IANNIS	457533.0	4	1535056.0	
3	CARUCCI, JOHN A	444422.0	1	444422.0	
4	GOLD, HEATHER TAFFET	424833.0	1	424833.0	
5	SCHNEIDER, ROBERT	419016.0	1	419016.0	
6	ROTHENBERG, ELI	253893.0	2	580181.0	
7	MOSCATELLI, DAVID A	221198.0	1	221198.0	
8	POSSEMATO, RICHARD LEWIS	221198.0	2	470197.0	
9	SKOK, JANE AMANDA	220825.0	2	745991.0	
10	OSMAN, IMAN	203400.0	1	203400.0	
11	POLSKY, DAVID	184331.0	1	184331.0	
12	MILLER, GEORGE	149955.0	2	501668.0	
13	BARCELLOS-HOFF, MARY HELEN	30244.0	2	242119.0	
14	CARROLL, WILLIAM L.	NaN	27	432951.0	

What Is the Most Common CI for each PI?

In [42]:

```
# Calculate how many projects each PI has with each CI
temp = df_NYU3.groupby(['PI_NAME', 'IC_NAME'])['APPLICATION_ID'].nunique().sort

# Convert into a dataframe and reset index
temp = temp.to_frame().reset_index()

# Rename APPLICATION_ID Count
temp.rename(columns={'APPLICATION_ID':'Count'}, inplace = True)

# Group by PI_NAME and select the resultant rows with the max value in the (nu
# Sorted in ascending order of Count
temp = temp.groupby(['PI_NAME'],sort=True)[['IC_NAME', 'Count']].max()
```

```
# Rename IC_NAME Most_common_IC
temp.rename(columns={'IC_NAME':'Most_common_IC_2015', 'Count': 'Most_common_IC_
temp = temp.reset_index()

# View the first 3 observations
temp.head(3)
```

Out[42]:

	PI_NAME	Most_common_IC_2015	Most_common_IC_Count_2015
0	BINKA, FRED NEWTON	FOGARTY INTERNATIONAL CENTER	1
1	BRAITHWAITE, RONALD SCOTT	NATIONAL INSTITUTE OF BIOMEDICAL IMAGING AND BIOENGINEERING	1
2	CHANDARANA, HERSH	NATIONAL INSTITUTE OF BIOMEDICAL IMAGING AND BIOENGINEERING	1

In [43]:
```
# Merge the temp dataframe into the DF_2015 data frame
# Use an outer merge
DF_2015= pd.merge(DF_2015, temp, on='PI_NAME', how = 'outer')

# Look at how many rows and columns there are
DF_2015.shape
```

Out[43]:
```
(508, 9)
```

In [44]:
```
# View the first 3 observations
DF_2015.head(3)
```

Out[44]:

	PI_NAME	NCI_New_Total_Cost_2015	Total_Projects_2015	Total_Cost_All_2015	Total_NCI
0	ZELENIUCH-JAQUOTTE, ANNE	788279.0	1	788279.0	
1	KIRCHHOFF, TOMAS	478926.0	2	663257.0	
2	AIFANTIS, IANNIS	457533.0	4	1535056.0	

Link Data on PI Name

Now we will merge together the `DF_2015` Date we just created with the pre processed `DF_2017` we read in at the beginning of this notebook. We will use an `outer` joint, so that we capture PI's the had grant(s) in only 2015, both 2015 and 2017, and only 2017

In [45]:
```
# Create a new dataframe by using an outer merge to merge the DF_2015 and DF_2(
# merging on PI_NAME
DF = pd.merge(DF_2015, DF_2017, on='PI_NAME', how = 'outer')
DF.shape
```

Out[45]:
```
(758, 17)
```

In [46]:
```
# Look at the first 5 observations
DF.head(5)
```

Out[46]:

	PI_NAME	NCI_New_Total_Cost_2015	Total_Projects_2015	Total_Cost_All_2015	Total_NCI
0	ZELENIUCH-JAQUOTTE, ANNE	788279.0	1.0	788279.0	
1	KIRCHHOFF, TOMAS	478926.0	2.0	663257.0	
2	AIFANTIS, IANNIS	457533.0	4.0	1535056.0	
3	CARUCCI, JOHN A	444422.0	1.0	444422.0	
4	GOLD, HEATHER TAFFET	424833.0	1.0	424833.0	

In [47]:
```
# Look at the last 5 observations
DF.tail(5)
```

Out[47]:

	PI_NAME	NCI_New_Total_Cost_2015	Total_Projects_2015	Total_Cost_All_2015	Total_NC
753	FALKNER, ANNEGRET LEA	NaN	NaN	NaN	
754	DOUTHIT, JESSICA P	NaN	NaN	NaN	
755	DORAN, KELLY	NaN	NaN	NaN	
756	DODSON, JOHN A	NaN	NaN	NaN	
757	ZOTTER, BRENDAN CARL	NaN	NaN	NaN	

Fill in the Missing values with 0

Caution! it is often not the case that you should fill missing values with Zero. It is appropriate in this case because we know for example, FALKNER ANNEGRET LEA, was not a PI on an NIH project for NYU in 2015. Note that this narrows the scope of the question, though, because FALKNER ANNEGRET LEA could have worked on non-NIH projects in 2015, or she could have worked on NIH projects at a different institution in 2015. You have to be very mindful of what assumptions you are making and how you are narrowing the scope of your research when you fill in missing values with zero for your own project.

In [48]:
```
# use the fillma function to fill all of the missing values with Zero
DF= DF.fillna(0)
```

```
# view the first 5 observations
DF.head()
```

Out[48]:

	PI_NAME	NCI_New_Total_Cost_2015	Total_Projects_2015	Total_Cost_All_2015	Total_NCI
0	ZELENIUCH-JAQUOTTE, ANNE	788279.0	1.0	788279.0	
1	KIRCHHOFF, TOMAS	478926.0	2.0	663257.0	
2	AIFANTIS, IANNIS	457533.0	4.0	1535056.0	
3	CARUCCI, JOHN A	444422.0	1.0	444422.0	
4	GOLD, HEATHER TAFFET	424833.0	1.0	424833.0	

Develop measures for change over time

Now that we have brought together the 2015 and 2017 data, we can create measures for the change in outcomes. We will develop two measures

1. Change in the number of NCI Projects
2. Change in the Total cost of NCI Projects

Change in the Number of National Cancer Institute Projects

In [49]:
```
# Create a new column (Change_in_NCI_Projects) that is equal to the difference
DF['Change_in_NCI_Projects'] = DF['Total_NCI_Projects_2017'] - DF['Total_NCI_Pr

# Look at the first 5 observations for select variables
DF[['PI_NAME','Total_NCI_Projects_2017', 'Total_NCI_Projects_2015', 'Change_in_
```

Out[49]:

	PI_NAME	Total_NCI_Projects_2017	Total_NCI_Projects_2015	Change_in_NCI_Projects
0	ZELENIUCH-JAQUOTTE, ANNE	1.0	1.0	0.0
1	KIRCHHOFF, TOMAS	1.0	2.0	-1.0
2	AIFANTIS, IANNIS	6.0	4.0	2.0
3	CARUCCI, JOHN A	1.0	1.0	0.0
4	GOLD, HEATHER TAFFET	1.0	1.0	0.0

Change in the Cost of National Cancer Institute Projects

In [50]:
```
# Create a new column (Change_in_NCI_Cost) that is equal to the difference in N
DF['Change_in_NCI_Cost'] = DF['NCI_Total_Cost_2017'] - DF['NCI_Total_Cost_2015'

# Look at the first 5 observations for select variables
DF[['PI_NAME','NCI_Total_Cost_2017', 'NCI_Total_Cost_2015', 'Change_in_NCI_Cost
```

Out[50]:

	PI_NAME	NCI_Total_Cost_2017	NCI_Total_Cost_2015	Change_in_NCI_Cost
0	ZELENIUCH-JAQUOTTE, ANNE	694206.0	788279.0	-94073.0
1	KIRCHHOFF, TOMAS	484247.0	663257.0	-179010.0
2	AIFANTIS, IANNIS	2654279.0	1535056.0	1119223.0
3	CARUCCI, JOHN A	206473.0	444422.0	-237949.0
4	GOLD, HEATHER TAFFET	406860.0	424833.0	-17973.0

Results

Now that the data is prepaired, we have answer some questions about PIs in 2015 and 2017

In [51]:
```
DF.columns
```

Out[51]:
```
Index(['PI_NAME', 'NCI_New_Total_Cost_2015', 'Total_Projects_2015',
       'Total_Cost_All_2015', 'Total_NCI_Projects_2015', 'NCI_Total_Cost_201
5',
       'Total_New_NCI_Projects_2015', 'Most_common_IC_2015',
       'Most_common_IC_Count_2015', 'Total_Projects_2017',
       'Total_Cost_All_2017', 'Total_NCI_Projects_2017', 'NCI_Total_Cost_201
7',
       'Total_New_NCI_Projects_2017', 'NCI_New_Total_Cost_2017',
       'Most_common_IC_2017', 'Most_common_IC_Count_2017',
       'Change_in_NCI_Projects', 'Change_in_NCI_Cost'],
      dtype='object')
```

How many PIs form NYU were there in 2015?

In [52]:
```
# Look at the first 2 observations for select variables
DF.head(2)
```

Out[52]:

	PI_NAME	NCI_New_Total_Cost_2015	Total_Projects_2015	Total_Cost_All_2015	Total_NCI
0	ZELENIUCH-JAQUOTTE, ANNE	788279.0	1.0	788279.0	
1	KIRCHHOFF, TOMAS	478926.0	2.0	663257.0	

In [53]:
```
# Subset dataframe for PIs who had more than zero total projects in 2015
temp = DF[DF['Total_Projects_2015'] > 0]
```

```
# Count the number of unique PI names
temp["PI_NAME"].nunique()
```

Out[53]: 508

How many PIs form NYU were there in 2017?

In [54]:
```
# Subset dataframe for PIs who had more than zero total projects in 2017
temp = DF[DF['Total_Projects_2017'] > 0]

# Count the number of unique PI names
temp["PI_NAME"].nunique()
```

Out[54]: 590

How many PIs from NYU had NCI projects in 2015?

In [55]:
```
# Subset dataframe for PIs who had more than zero NCI projects in 2015
temp = DF[DF['Total_NCI_Projects_2015'] > 0]

# Count the number of unique PI names
temp["PI_NAME"].nunique()
```

Out[55]: 68

How many PIs from NYU had NCI projects in 2017?

In [56]:
```
# Subset dataframe for PIs who had more than zero NCI projects in 2017
temp = DF[DF['Total_NCI_Projects_2017'] > 0]

# Count the number of unique PI names
temp["PI_NAME"].nunique()
```

Out[56]: 80

How many PIs from NYU had new NCI projects in 2015?

In [57]:
```
# Subset dataframe for PIs who had more than zero new NCI projects in 2015
temp = DF[DF['Total_New_NCI_Projects_2015'] > 0]

# Count the number of unique PI names
temp["PI_NAME"].nunique()
```

Out[57]: 14

How many PIs from NYU had new NCI projects in 2017?

In [58]:
```
# Subset dataframe for PIs who had more than zero new NCI projects in 2017
temp = DF[DF['Total_New_NCI_Projects_2017'] > 0]

# Count the number of unique PI names
temp["PI_NAME"].nunique()
```

Out[58]: 26

How many PIs only had non NCI Project(s) in 2015 but had at least one NCI project in 2017?

In [59]:
```
# Subset dataframe for PIs who had zero NCI projects in 2015
# and more than zero projects in 2015
# and more than zero NCI projects in 2017
Converted_PIs = DF[(DF['Total_NCI_Projects_2015'] == 0) &
           (DF['Total_Projects_2015'] > 0) &
           (DF['Total_NCI_Projects_2017'] > 0)]

# Count the number of unique PI names
Converted_PIs["PI_NAME"].nunique()
```

Out[59]: 2

In [60]:
```
Converted_PIs.head()
```

Out[60]:

	PI_NAME	NCI_New_Total_Cost_2015	Total_Projects_2015	Total_Cost_All_2015	Total_N
73	DYNLACHT, BRIAN D	0.0	2.0	688410.0	
284	SCHMIDT, BRIAN L	0.0	1.0	396250.0	

In [61]:
```
# look at the first 5 observations for select variables
Converted_PIs[['PI_NAME', 'Total_Projects_2015', 'Total_Cost_All_2015', 'Total_
      'Total_NCI_Projects_2017', 'NCI_Total_Cost_2017', 'Most_common_IC_2015']]
```

Out[61]:

	PI_NAME	Total_Projects_2015	Total_Cost_All_2015	Total_NCI_Projects_2015	Total_NCI
73	DYNLACHT, BRIAN D	2.0	688410.0	0.0	
284	SCHMIDT, BRIAN L	1.0	396250.0	0.0	

Checkpoint 2 (Assignment 3 due March 10th)

How many PIs had a NCI funded projects in 2017 but not 2015? (1point)

For those who have at least one NCI funded project in 2015, how many NCI funded projects do they have on average? (1point)

For those who have at least one NCI funded project in 2017, how many NCI funded projects do they have on average? (1point)

For the PIs who only had non-NCI Project(s) in 2015 but had at least one NCI project in 2017, what are the 5 most common ICs they are moving from (in total, not per PI)? (2points)

Can we sum the `Total_Cost_All_2015` column in the `DF` dataframe to get the total cost of projects in 2015? Why or why not? (2points)

- This is not a question you need code to answer, instead, create a Markdown cell, and answer in a complete sentence.

1. How many PIs had a NCI funded projects in 2017 but not 2015? (1point)

In [62]:
```
Converted_PIs = DF[(DF['Total_NCI_Projects_2017'] > 0) &
           (DF['Total_NCI_Projects_2015'] == 0)]

# Count the number of unique PI names
Converted_PIs["PI_NAME"].nunique()
```

Out[62]:
```
41
```

2. For those who have at least one NCI funded project in 2015, how many NCI funded projects do they have on average? (1point)

In [63]:
```
# Subset dataframe for PIs who had more than zero NCI projects in 2015
temp = DF[DF['Total_NCI_Projects_2015'] > 0]

temp["PI_NAME"].nunique()
temp

temp2 = temp ["Total_NCI_Projects_2015"].mean()
temp2
```

Out[63]:
```
1.75
```

3. For those who have at least one NCI funded project in 2017, how many NCI funded projects do they have on average? (1point)

In [64]:
```
# Subset dataframe for PIs who had more than zero NCI projects in 2017
temp = DF[DF['Total_NCI_Projects_2017'] > 0]
temp
temp3=temp["Total_NCI_Projects_2017"].mean()
temp3
```

Out[64]:
```
1.75
```

4. For the PIs who only had non-NCI Project(s) in 2015 but had at least one NCI project in 2017, what are the 5 most common ICs they are moving from (in total, not per PI)? (2points)

In [65]:
```
# Subset dataframe for PIs who had zero NCI projects in 2015
# and more than zero projects in 2015
# and more than zero NCI projects in 2017
Converted_PIs = DF[(DF['Total_NCI_Projects_2015'] == 0) &
           (DF['Total_Projects_2015'] > 0) &
```

```
            (DF['Total_NCI_Projects_2017'] > 0)]
Converted_PIs
```

Out[65]:

	PI_NAME	NCI_New_Total_Cost_2015	Total_Projects_2015	Total_Cost_All_2015	Total_N
73	DYNLACHT, BRIAN D	0.0	2.0	688410.0	
284	SCHMIDT, BRIAN L	0.0	1.0	396250.0	

There are only 2 PI who only had non-NCI Project(s) in 2015 but had at least one NCI project in 2017. They are DYNLACHT, BRIAN D, and SCHMIDT, BRIAN L.

5. Can we sum the `Total_Cost_All_2015` column in the `DF` dataframe to get the total cost of projects in 2015? Why or why not? (2points)

- This is not a question you need code to answer, instead, create a Markdown cell, and answer in a complete sentence.

No, because each project can have more than one PI, but each PI has one value of Total_Costs_All_2015 in the DF dataframe which in fact equals to the total cost of each project. If we sum them up, we will overcalculate the total costs of projects in 2015.