**Andrew Stanford**

**Due Date: 1/22/2018**

**Python Exercise**

**PART 1**

# Import the datadotworld module as dw

import datadotworld as dw

# Import the city council votes dataset

dataset = dw.load\_dataset('https://data.world/stephen-hoover/chicago-city-council-votes')

**PART 2**

# Import the city council votes dataset

dataset = dw.load\_dataset('https://data.world/stephen-hoover/chicago-city-council-votes')

# Use describe() to review all the metadata that is downloaded with the dataset.

# Print it to the screen using pp.pprint().

dataset.describe()

pp.pprint(dataset.describe())

# Use describe() again to get a description of a specific resource: alderman\_votes. Print it to the screen.

dataset.describe('alderman\_votes')

pp.pprint(dataset.describe('alderman\_votes'))

**PART 2**

In [1]: alderman\_votes = dataset.describe('alderman\_votes')

In [2]: alderman\_votes

**PART 2**

# The datadotworld module and dataset have already been loaded for you:

import datadotworld as dw

dataset = dw.load\_dataset('https://data.world/stephen-hoover/chicago-city-council-votes')

# Use the dataframes property to assign the alderman\_votes table to the variable votes\_dataframe.

votes\_dataframe = dataset.dataframes['alderman\_votes']

# Use the pandas shape property to get rows/columns size for the `votes\_dataframe` dataframe.

pp.pprint(votes\_dataframe.shape)

# Use the pandas head function to print the first 3 rows of the `votes\_dataframe` dataframe.

pp.pprint(votes\_dataframe.head(3))

**PART 3**

# datadotworld module has been imported as dw

import datadotworld as dw

# We've loaded two datasets to use

int\_dataset = dw.load\_dataset('https://data.world/jonloyens/intermediate-data-world')

fipsCodes\_dataset = dw.load\_dataset('https://data.world/uscensusbureau/fips-state-codes')

# Create two dataframes: police\_shootings from the 'fatal\_police\_shootings\_data' table of int\_dataset and state\_abbrvs, from the 'statesfipscodes' table of fipsCodes\_dataset

police\_shootings = int\_dataset.dataframes['fatal\_police\_shootings\_data']

state\_abbrvs = fipsCodes\_dataset.dataframes['statesfipscodes']

## Merge the two datasets together on the state and stusab fields. Assign to a merged\_dataframe variable.

merged\_dataframe = police\_shootings.merge(state\_abbrvs, how = 'left', left\_on = 'state', right\_on='stusab')

## Add a 'citystate' column to the merged\_dataframe dataframe, populating it with the concatinated values from the 'city' and 'state\_name' columns, separated by ', '.

merged\_dataframe["citystate"] = merged\_dataframe["city"] + ", " + merged\_dataframe["state\_name"]

## Print head of merged\_dataframe

pp.pprint(merged\_dataframe.head(5))

**PART 4**

# datadotworld module has been imported as dw

import datadotworld as dw

## Complete the SQL query to select all rows from the `unhcr\_all` table where `Year` equals 2010. Assign the query string to a `sql\_query` variable.

sql\_query = "SELECT \* FROM `unhcr\_all` WHERE Year = 2010"

## Use the `query` method of the datadotworld module to run the `sql\_query` against the `https://data.world/nrippner/refugee-host-nations` dataset. Assign the results to a `query2010` variable.

query2010 = dw.query('https://data.world/nrippner/refugee-host-nations', sql\_query)

## Use the dataframe property of the resulting query to create a dataframe variable named `unhcr2010`

unhcr2010 = query2010.dataframe

## Print the first 5 rows using the head method.

pp.pprint(unhcr2010.head(5))

**PART 4**

# datadotworld module has been imported as dw

import datadotworld as dw

## Complete the SQL query to select state, the count of farmers markets (fmid), and average obesity rate from agriculture.`national-farmers-markets`.export, LEFT JOINED against health.`obesity-by-state-2014`.adult\_obese on state and location

sql\_query = "SELECT state, count(fmid) as count, Avg(obesity.Value) as obesityAvg FROM Export LEFT JOIN health.`obesity-by-state-2014`.`adult\_obese` as obesity ON state = obesity.location GROUP BY state ORDER BY count desc"

## Use the `query` method of the datadotworld module to run the `sql\_query` against the `https://data.world/agriculture/national-farmers-markets` dataset. Assign the results to a `queryResults` variable.

queryResults = dw.query('https://data.world/agriculture/national-farmers-markets', sql\_query)

## Use the dataframes property of the resulting query to create a dataframe variable named `stateStats`

stateStats = queryResults.dataframe

## Plot the stateStats results using state as the x-axis (matplotlib is already imported)

stateStats.plot(x='state')

plt.show()

**PART 4**

# datadotworld module has been imported as dw

import datadotworld as dw

# We've written a SPARQL query for you and assigned it to the `sparql\_query` variable:

sparql\_query = "PREFIX GOT: <https://tutorial.linked.data.world/d/sparqltutorial/> SELECT ?FName ?LName WHERE {?person GOT:col-got-house \"Stark\" . ?person GOT:col-got-fname ?FName . ?person GOT:col-got-lname ?LName .}"

# Use the pre-defined SPARQL query to query dataset http://data.world/tutorial/sparqltutorial and return the results to a queryResults variable

queryResults = dw.query('http://data.world/tutorial/sparqltutorial', sparql\_query, query\_type='sparql')

# Use the dataframe property of the resulting query to create a dataframe variable named `houseStark`

houseStark = queryResults.dataframe

# Use pp.pprint() to print the dataframe to the screen.

pp.pprint(houseStark)

PART 5 – No work…

# Import the datadotworld module as dw and the sys module

import datadotworld as dw

import sys

# Import a dataset

refugee\_dataset = dw.load\_dataset('nrippner/refugee-host-nations')

# Get the size of the dataset:

sys.getsizeof(refugee\_dataset)

# List all of the data files:

dataframes = refugee\_dataset.dataframes

for df in dataframes:

pp.pprint(df)

# print all of the files in a dataset:

resources = refugee\_dataset.describe()['resources']

pp.pprint('name:')

for r in resources:

pp.pprint(r['name'])

pp.pprint('\ntype of file:')

for r in resources:

pp.pprint(r['format'])