Everything is Better with Friends

Using SAS in Python Applications with SASPy and Open-Source Tooling (Beyond the Basics)

Setup for Part 2

Getting setup to use Google Colab with SAS OnDemand for Academics (ODA)

1. To execute code cells, you'll need credentials for the following:
    - Google. (If you're not already signed in, you should see a **Sign In** button in the upper right corner. You can also visit https://accounts.google.com/signup to create an account for free.)
2. We recommend enabling line numbers using the Tools menu: **Tools** -> **Settings** -> **Editor** -> **Show line numbers** -> **Save**
3. We also recommend enabling the Table of Contents using the View menu: **View** -> **Table of contents**
4. To save a copy of this notebook, along with any edits you make, please use the File menu: **File** -> **Save a copy in Drive**
5. Looking for "extra credit"? Please let us know if you spot any typos!

Install and import packages

```
# Install the rich module for colorful printing
!pip install rich

# We'll use IPython to display DataFrames or HTML content
from IPython.display import display, HTML

# We'll use the pandas package to create and manipulate DataFrame objects
```

```
import pandas

# We'll use the requests package to call a web API
import requests

# We're overwriting the default print function with rich.print
from rich import print

# We're also setting the maximum line width of rich.print to be a bit wider (to avoid line wrapping)
from rich import get_console
console = get_console()
console.width = 165
```

```
Collecting rich
  Downloading rich-12.4.1-py3-none-any.whl (231 kB)
     |████████████████████████████████| 231 kB 26.3 MB/s
Requirement already satisfied: typing-extensions<5.0,>=4.0.0 in /usr/local/lib/python3.7/dist-packages (from
Collecting commonmark<0.10.0,>=0.9.0
  Downloading commonmark-0.9.1-py2.py3-none-any.whl (51 kB)
     |████████████████████████████████| 51 kB 8.6 MB/s
Requirement already satisfied: pygments<3.0.0,>=2.6.0 in /usr/local/lib/python3.7/dist-packages (from rich) (
Installing collected packages: commonmark, rich
Successfully installed commonmark-0.9.1 rich-12.4.1
```

Part 2. Rectangularizing unstructured data in Python applications

Section 2.1. Create pharm_class_response

```
# Let's explore one of the many endpoints of the openFDA API!

# Use an open web API to get the number of drugs available by pharmacologic class.
# Note: By default, only the first 100 results are provided, sorted in descending order by count.
# To retrieve more than the first 100 results, a combination of limit and skip parameters can be
# used, as described at https://open.fda.gov/apis/paging/
pharm_class_response = requests.get('https://api.fda.gov/drug/ndc.json?count=pharm_class.exact')
```

```
8
9 # Check the resulting status code to make sure the API call was successful, with 200 = "OK".
10 http_status = pharm_class_response.status_code
11 http_status_info = f'https://httpstatuses.com/{http_status}'
12 if http_status == 200:
13     print('API call successful!\n')
14 print(f'See {http_status_info} for more information about HTTP status code {http_status}.')
```

```
API call successful!

See https://httpstatuses.com/200 for more information about HTTP status coc
```

Concept Check 2.1

1. True or False: Changing Line 12 to a single-equals (`=`) would have the same effect.
2. True or False: Removing the indentation on Line 13 would have the same effect.

- Fun Fact: The FDA provides many open APIs. Examples for the APIs related specifically to the National Drug Code (NDC) database, including the API used above, can be found at https://open.fda.gov/apis/drug/ndc/example-api-queries/

Solution: False! Single-equals (`=`) is only used for variable assignment, and double-equals (`==`) is only used to test for equality.

Section 2.2. Create pharm_class_json

```
1 # Extract and print the JSON-formatted list of counts of drugs by pharmacologic class.
2 pharm_class_json = pharm_class_response.json()
3 print(pharm_class_json)
```

```
{
    'meta': {
        'disclaimer': 'Do not rely on openFDA to make decisions regarding
all results are unvalidated. We may limit or otherwise restrict your acce
        'terms': 'https://open.fda.gov/terms/',
        'license': 'https://open.fda.gov/license/',
        'last_updated': '2022-05-20'
    },
    'results': [
        {'term': 'Cell-mediated Immunity [PE]', 'count': 6296},
        {'term': 'Increased Histamine Release [PE]', 'count': 6275},
        {'term': 'Allergens [CS]', 'count': 6266},
        {'term': 'Increased IgG Production [PE]', 'count': 4114},
        {'term': 'Anti-Inflammatory Agents', 'count': 4095},
        {'term': 'Cyclooxygenase Inhibitors [MoA]', 'count': 4095},
        {'term': 'Non-Steroidal [CS]', 'count': 4095},
        {'term': 'Nonsteroidal Anti-inflammatory Drug [EPC]', 'count': 40
        {'term': 'Corticosteroid Hormone Receptor Agonists [MoA]', 'count
        {'term': 'Corticosteroid [EPC]', 'count': 2688},
        {'term': 'Histamine H1 Receptor Antagonists [MoA]', 'count': 2653
        {'term': 'Histamine-1 Receptor Antagonist [EPC]', 'count': 2534},
        {'term': 'Osmotic Activity [MoA]', 'count': 2224},
        {'term': 'Increased Large Intestinal Motility [PE]', 'count': 221
        {'term': 'Sigma-1 Agonist [EPC]', 'count': 2182},
        {'term': 'Sigma-1 Receptor Agonists [MoA]', 'count': 2182},
        {'term': 'Uncompetitive N-methyl-D-aspartate Receptor Antagonist
        {'term': 'Uncompetitive NMDA Receptor Antagonists [MoA]', 'count'
        {'term': 'Osmotic Laxative [EPC]', 'count': 2143},
        {'term': 'Decreased Central Nervous System Disorganized Electrica
        {'term': 'Inhibition Large Intestine Fluid/Electrolyte Absorption
        {'term': 'Adrenergic alpha1-Agonists [MoA]', 'count': 2028},
        {'term': 'alpha-1 Adrenergic Agonist [EPC]', 'count': 2028},
        {'term': 'Opioid Agonist [EPC]', 'count': 1816},
        {'term': 'Pollen [CS]', 'count': 1727},
        {'term': 'Non-Standardized Pollen Allergenic Extract [EPC]', 'cou
        {'term': 'Plant Proteins [CS]', 'count': 1618},
        {'term': 'Dietary Proteins [CS]', 'count': 1576},
        {'term': 'Non-Standardized Food Allergenic Extract [EPC]', 'count
        {'term': 'Antiarrhythmic [EPC]', 'count': 1548},
        {'term': 'Serotonin Uptake Inhibitors [MoA]', 'count': 1540},
        {'term': 'Non-Standardized Plant Allergenic Extract [EPC]', 'coun
        {'term': 'beta-Adrenergic Blocker [EPC]', 'count': 1510},
        {'term': 'Local Anesthesia [PE]', 'count': 1429},
        {'term': 'Stimulation Large Intestine Fluid/Electrolyte Secretion
        {'term': 'Amide Local Anesthetic [EPC]', 'count': 1379},
        {'term': 'Amides [CS]', 'count': 1379},
```

```
{'term': 'Adrenergic beta-Antagonists [MoA]', 'count': 1363},
{'term': 'Calculi Dissolution Agent [EPC]', 'count': 1360},
{'term': 'Anti-epileptic Agent [EPC]', 'count': 1344},
{'term': 'Atypical Antipsychotic [EPC]', 'count': 1341},
{'term': 'Central Nervous System Stimulation [PE]', 'count': 1339
{'term': 'Copper Absorption Inhibitor [EPC]', 'count': 1291},
{'term': 'Decreased Copper Ion Absorption [PE]', 'count': 1291},
{'term': 'Decreased Prostaglandin Production [PE]', 'count': 1291
{'term': 'Central Nervous System Stimulant [EPC]', 'count': 1290}
{'term': 'Full Opioid Agonists [MoA]', 'count': 1286},
{'term': 'Angiotensin 2 Receptor Blocker [EPC]', 'count': 1189},
{'term': 'Angiotensin 2 Receptor Antagonists [MoA]', 'count': 118
{'term': 'HMG-CoA Reductase Inhibitor [EPC]', 'count': 1183},
{'term': 'Hydroxymethylglutaryl-CoA Reductase Inhibitors [MoA]',
{'term': 'Cytochrome P450 2C19 Inhibitors [MoA]', 'count': 1177},
{'term': 'Calcium Channel Antagonists [MoA]', 'count': 1161},
{'term': 'Increased Diuresis [PE]', 'count': 1104},
{'term': 'Inhibition Small Intestine Fluid/Electrolyte Absorption
{'term': 'Magnesium Ion Exchange Activity [MoA]', 'count': 1099},
{'term': 'Decreased Platelet Aggregation [PE]', 'count': 1091},
{'term': 'Serotonin Reuptake Inhibitor [EPC]', 'count': 1081},
{'term': 'Norepinephrine Uptake Inhibitors [MoA]', 'count': 1055}
{'term': 'Standardized Chemical Allergen [EPC]', 'count': 1028},
{'term': 'Angiotensin Converting Enzyme Inhibitor [EPC]', 'count'
{'term': 'Angiotensin-converting Enzyme Inhibitors [MoA]', 'count
{'term': 'Calcium [CS]', 'count': 997},
{'term': 'Cytochrome P450 3A4 Inhibitors [MoA]', 'count': 992},
{'term': 'Platelet Aggregation Inhibitor [EPC]', 'count': 991},
{'term': 'Increased Coagulation Factor Activity [PE]', 'count': 9
{'term': 'Proton Pump Inhibitor [EPC]', 'count': 978},
{'term': 'Proton Pump Inhibitors [MoA]', 'count': 978},
{'term': 'Blood Coagulation Factor [EPC]', 'count': 971},
{'term': 'Cations', 'count': 971},
{'term': 'Divalent [CS]', 'count': 971},
{'term': 'Thiazide Diuretic [EPC]', 'count': 920},
{'term': 'Thiazides [CS]', 'count': 920},
{'term': 'Benzodiazepine [EPC]', 'count': 912},
{'term': 'Benzodiazepines [CS]', 'count': 912},
{'term': 'Seed Storage Proteins [CS]', 'count': 899},
{'term': 'Adrenergic alpha-Agonists [MoA]', 'count': 844},
{'term': 'alpha-Adrenergic Agonist [EPC]', 'count': 844},
{'term': 'Skin Barrier Activity [PE]', 'count': 835},
{'term': 'Azole Antifungal [EPC]', 'count': 822},
{'term': 'Azoles [CS]', 'count': 822},
{'term': 'Penicillin-class Antibacterial [EPC]', 'count': 814},
{'term': 'Penicillins [CS]', 'count': 814},
```

```
{'term': 'Mood Stabilizer [EPC]', 'count': 785},
{'term': 'Non-Standardized Fungal Allergenic Extract [EPC]', 'cou
{'term': 'Antihistamine [EPC]', 'count': 766},
{'term': 'Histamine Receptor Antagonists [MoA]', 'count': 766},
{'term': 'Fungal Proteins [CS]', 'count': 765},
{'term': 'Dihydropyridine Calcium Channel Blocker [EPC]', 'count'
{'term': 'Dihydropyridines [CS]', 'count': 762},
{'term': 'Potassium Compounds [CS]', 'count': 662},
{'term': 'Potassium Salt [EPC]', 'count': 662},
{'term': 'P-Glycoprotein Inhibitors [MoA]', 'count': 657},
{'term': 'Tricyclic Antidepressant [EPC]', 'count': 650},
{'term': 'Centrally-mediated Muscle Relaxation [PE]', 'count': 64
```

Concept Check 2.2

- Short Answer: What types of standard Python objects appear in the output of `pharm_class_json`?
- Fun Fact: In Python, it's common to work with deeply nested objects (like a Russian nested doll, or a Turducken).

Solution: We see instances of `int`, `str`, `dict`, and `list`.

Section 2.3. Create pharm_class_list

```
# When an API returns a nested collection of dicts and lists like this, we need to match the
# structure recursively using
# (a) dict-indexing to get values corresponding to specific keys and
# (b) for-loops to loop over lists.

# Accumulate pharmacologic classes and counts in a list of lists called pharm_class_list.
pharm_class_list = []
for pharm_class_count in pharm_class_json['results']:
    pharm_class_list.append(
        [
            pharm_class_count['term'],
            pharm_class_count['count'],
        ]
    )

```

```
# In case we want to track when these API results were obtained, let's also extract the date.
pharm_class_date = pharm_class_json['meta']['last_updated']

# Now let's print the date.
print(f'Date of API results: {pharm_class_date}')
print('\n')

# And then let's print pharm_class_list.
print(pharm_class_list)
```

Date of API results: 2022-05-20

```
[
    ['Cell-mediated Immunity [PE]', 6296],
    ['Increased Histamine Release [PE]', 6275],
    ['Allergens [CS]', 6266],
    ['Increased IgG Production [PE]', 4114],
    ['Anti-Inflammatory Agents', 4095],
    ['Cyclooxygenase Inhibitors [MoA]', 4095],
    ['Non-Steroidal [CS]', 4095],
    ['Nonsteroidal Anti-inflammatory Drug [EPC]', 4095],
    ['Corticosteroid Hormone Receptor Agonists [MoA]', 2688],
    ['Corticosteroid [EPC]', 2688],
    ['Histamine H1 Receptor Antagonists [MoA]', 2653],
    ['Histamine-1 Receptor Antagonist [EPC]', 2534],
    ['Osmotic Activity [MoA]', 2224],
    ['Increased Large Intestinal Motility [PE]', 2210],
    ['Sigma-1 Agonist [EPC]', 2182],
    ['Sigma-1 Receptor Agonists [MoA]', 2182],
    ['Uncompetitive N-methyl-D-aspartate Receptor Antagonist [EPC]', 2182],
    ['Uncompetitive NMDA Receptor Antagonists [MoA]', 2182],
    ['Osmotic Laxative [EPC]', 2143],
    ['Decreased Central Nervous System Disorganized Electrical Activity [PE]', 2055],
    ['Inhibition Large Intestine Fluid/Electrolyte Absorption [PE]', 2040],
    ['Adrenergic alpha1-Agonists [MoA]', 2028],
    ['alpha-1 Adrenergic Agonist [EPC]', 2028],
    ['Opioid Agonist [EPC]', 1816],
    ['Pollen [CS]', 1727],
    ['Non-Standardized Pollen Allergenic Extract [EPC]', 1645],
    ['Plant Proteins [CS]', 1618],
    ['Dietary Proteins [CS]', 1576],
    ['Non-Standardized Food Allergenic Extract [EPC]', 1574],
    ['Antiarrhythmic [EPC]', 1548],
    ['Serotonin Uptake Inhibitors [MoA]', 1540],
    ['Non-Standardized Plant Allergenic Extract [EPC]', 1535],
    ['beta-Adrenergic Blocker [EPC]', 1510],
    ['Local Anesthesia [PE]', 1429],
    ['Stimulation Large Intestine Fluid/Electrolyte Secretion [PE]', 1390],
    ['Amide Local Anesthetic [EPC]', 1379],
    ['Amides [CS]', 1379],
    ['Adrenergic beta-Antagonists [MoA]', 1363],
    ['Calculi Dissolution Agent [EPC]', 1360],
    ['Anti-epileptic Agent [EPC]', 1344],
    ['Atypical Antipsychotic [EPC]', 1341],
    ['Central Nervous System Stimulation [PE]', 1339],
    ['Copper Absorption Inhibitor [EPC]', 1291],
```

```
['Decreased Copper Ion Absorption [PE]', 1291],
['Decreased Prostaglandin Production [PE]', 1291],
['Central Nervous System Stimulant [EPC]', 1290],
['Full Opioid Agonists [MoA]', 1286],
['Angiotensin 2 Receptor Blocker [EPC]', 1189],
['Angiotensin 2 Receptor Antagonists [MoA]', 1185],
['HMG-CoA Reductase Inhibitor [EPC]', 1183],
['Hydroxymethylglutaryl-CoA Reductase Inhibitors [MoA]', 1183],
['Cytochrome P450 2C19 Inhibitors [MoA]', 1177],
['Calcium Channel Antagonists [MoA]', 1161],
['Increased Diuresis [PE]', 1104],
['Inhibition Small Intestine Fluid/Electrolyte Absorption [PE]', 1099],
['Magnesium Ion Exchange Activity [MoA]', 1099],
['Decreased Platelet Aggregation [PE]', 1091],
['Serotonin Reuptake Inhibitor [EPC]', 1081],
['Norepinephrine Uptake Inhibitors [MoA]', 1055],
['Standardized Chemical Allergen [EPC]', 1028],
['Angiotensin Converting Enzyme Inhibitor [EPC]', 1023],
['Angiotensin-converting Enzyme Inhibitors [MoA]', 1023],
['Calcium [CS]', 997],
['Cytochrome P450 3A4 Inhibitors [MoA]', 992],
['Platelet Aggregation Inhibitor [EPC]', 991],
['Increased Coagulation Factor Activity [PE]', 981],
['Proton Pump Inhibitor [EPC]', 978],
```

Concept Check 2.3

1. True or False: Changing Line 8 to `pharm_class_json['RESULTS']` (i.e., changing the dictionary key to all caps) would have the same effect.
2. Short Answer: What types of standard Python objects appear in the definition of `pharm_class_list`?

- Fun Fact: Instead of bothering with a list of lists, we could have instead built a DataFrame row-by-row inside the for-loop. However, DataFrame operations inside a for-loop tend to be slow.

```
['Azole Antifungal [EPC]', 822],
```

Solution: False! Dictionary keys are case-sensitive in Python, just like variable names.

```
['Mood Stabilizer [EPC]', 785],
```

Section 2.4. Create pharm_class_df

```
['Fungal Proteins [CS]', 765],
```

```
# Now that we've finish looping, we can put the definitions in a DataFrame called pharm_class_df.
pharm_class_df = pandas.DataFrame(pharm_class_list, columns = ['term', 'count'])

# We can also inspect the size of pharm_class_df.
print(f'The size of pharm_class_df: {pharm_class_df.shape}')
print('\n')

# In addition, we can get a sense of the average size pharmacologic class.
print(f"The median size pharmacologic class in pharm_class_df: {pharm_class_df['count'].median()}")
print('\n')

# Finally, we can display pharm_class_df.
print(f'The contents of pharm_class_df:')
display(pharm class df)
```

```
The size of pharm_class_df: (100, 2)
```

Concept Check 2.4

- Short Answer: Other than the median, what are some descriptive statistics we might consider using to better understand the contents of `pharm_class_df`?
- Fun Fact: JSON-formatted data is useful because of how flexibly information can be nested. However, to actually work with the information inside, it's common to first rectangularize the JSON object.

Solution: The following section of the "Getting Started" guide for `pandas` gives a good overview: https://pandas.pydata.org/docs/getting_started/intro_tutorials/06_calculate_statistics.html

Section 2.5. Additional Exercises

For practice, we recommend the following:

- Run the code cell below.
- Repeat the steps in Sections 2.3-4 with the following two changes:
  - Form a DataFrame whose first column is `count`.
  - Calculate a statistic other than `median`.

```
# Let's try a different openFDA endpoint.
generic_name_response = requests.get('https://api.fda.gov/drug/ndc.json?count=generic_name.exact')

# Check the resulting status code to make sure the API call was successful, with 200 = "OK".
if generic_name_response.status_code == 200:
    print('API call successful!\n')

# Finally, let's extract and print the JSON-formatted return value.
generic_name_json = generic_name_response.json()
print('Here\'s the resulting data structure:')
```

```
print(generic_name_json)

# Accumulate generic names and counts in a list of lists called generic_name_list.
generic_name_list = []
for generic_name_count in generic_name_json['results']:
    generic_name_list.append(
        [
            generic_name_count['count'],
            generic_name_count['term'],
        ]
    )

# In case we want to track when these API results were obtained, let's also extract the date.
generic_name_date = generic_name_json['meta']['last_updated']

# Now let's print the date.
print(f'Date of API results: {generic_name_date}')
print('\n')

# And then let's print generic_name_list.
print(generic_name_list)

# Now that we've finish looping, we can put the definitions in a DataFrame called generic_name_df.
generic_name_df = pandas.DataFrame(generic_name_list, columns = ['count','term'])

# We can also inspect the size of generic_name_df.
print(f'The size of generic_name_df: {generic_name_df.shape}')
print('\n')

# In addition, we can get a sense of the average size generic type.
print(f"The mean generics count in generic_name_df: {generic_name_df['count'].mean()}")
print('\n')

# Finally, we can display generic_name_df.
print(f'The contents of generic_name_df:')
display(generic_name_df)
```

```
API call successful!

Here's the resulting data structure:
{
    'meta': {
        'disclaimer': 'Do not rely on openFDA to make decisions regarding
all results are unvalidated. We may limit or otherwise restrict your acce
        'terms': 'https://open.fda.gov/terms/',
        'license': 'https://open.fda.gov/license/',
        'last_updated': '2022-05-20'
    },
    'results': [
        {'term': 'ALCOHOL', 'count': 3032},
        {'term': 'Alcohol', 'count': 1445},
        {'term': 'Ethyl Alcohol', 'count': 1046},
        {'term': 'Ibuprofen', 'count': 1017},
        {'term': 'Acetaminophen', 'count': 1005},
        {'term': 'Benzalkonium Chloride', 'count': 898},
        {'term': 'BENZALKONIUM CHLORIDE', 'count': 779},
        {'term': 'Zinc Oxide', 'count': 723},
        {'term': 'Menthol', 'count': 636},
        {'term': 'Isopropyl Alcohol', 'count': 586},
        {'term': 'Sodium Fluoride', 'count': 567},
        {'term': 'Salicylic Acid', 'count': 508},
        {'term': 'Oxygen', 'count': 448},
        {'term': 'ETHYL ALCOHOL', 'count': 435},
        {'term': 'Benzocaine', 'count': 417},
        {'term': 'Aspirin', 'count': 403},
        {'term': 'Gabapentin', 'count': 395},
        {'term': 'Nicotine Polacrilex', 'count': 377},
        {'term': 'ZINC OXIDE', 'count': 366},
        {'term': 'Diphenhydramine HCl', 'count': 347},
        {'term': 'SALICYLIC ACID', 'count': 327},
        {'term': 'Avobenzone, Homosalate, Octisalate, Octocrylene', 'coun
        {'term': 'Lisinopril', 'count': 311},
        {'term': 'MENTHOL', 'count': 306},
        {'term': 'Levothyroxine Sodium', 'count': 294},
        {'term': 'Hydrocortisone', 'count': 285},
        {'term': 'Naproxen Sodium', 'count': 284},
        {'term': 'TITANIUM DIOXIDE', 'count': 279},
        {'term': 'Loratadine', 'count': 267},
        {'term': 'Benzalkonium chloride', 'count': 265},
        {'term': 'Simethicone', 'count': 262},
        {'term': 'Aripiprazole', 'count': 261},
        {'term': 'Famotidine', 'count': 258},
        {'term': 'OCTINOXATE, TITANIUM DIOXIDE', 'count': 258},
```

```
{'term': 'OCTINOXATE and TITANIUM DIOXIDE', 'count': 250},
{'term': 'Titanium Dioxide and Zinc Oxide', 'count': 249},
{'term': 'Hand Sanitizer', 'count': 247},
{'term': 'Lamotrigine', 'count': 247},
{'term': 'Prednisone', 'count': 244},
{'term': 'Diphenhydramine Hydrochloride', 'count': 240},
{'term': 'Omeprazole', 'count': 240},
{'term': 'PREGABALIN', 'count': 234},
{'term': 'Guaifenesin', 'count': 233},
{'term': 'Levetiracetam', 'count': 229},
{'term': 'Amoxicillin', 'count': 228},
{'term': 'Calcium Carbonate', 'count': 225},
{'term': 'ISOPROPYL ALCOHOL', 'count': 225},
{'term': 'Titanium Dioxide, Zinc Oxide', 'count': 225},
{'term': 'alcohol', 'count': 224},
{'term': 'Pyrithione Zinc', 'count': 223},
{'term': 'Titanium Dioxide', 'count': 222},
{'term': 'Lidocaine', 'count': 218},
{'term': 'Metformin Hydrochloride', 'count': 217},
{'term': 'Tadalafil', 'count': 216},
{'term': 'Cetirizine Hydrochloride', 'count': 213},
{'term': 'Ethyl alcohol', 'count': 213},
{'term': 'Chloroxylenol', 'count': 211},
{'term': 'TITANIUM DIOXIDE, ZINC OXIDE', 'count': 209},
{'term': 'Bismuth subsalicylate', 'count': 206},
{'term': 'Olanzapine', 'count': 205},
{'term': 'Losartan Potassium', 'count': 202},
{'term': 'SODIUM FLUORIDE', 'count': 202},
{'term': 'Pregabalin', 'count': 200},
{'term': 'levothyroxine sodium', 'count': 190},
{'term': 'Hydrocodone Bitartrate and Acetaminophen', 'count': 188
{'term': 'Aluminum Chlorohydrate', 'count': 183},
{'term': 'Dimethicone', 'count': 181},
{'term': 'Carvedilol', 'count': 177},
{'term': 'Fluconazole', 'count': 173},
{'term': 'ATORVASTATIN CALCIUM', 'count': 172},
{'term': 'Avobenzone, Homosalate, Octisalate, Octocrylene, Oxyben
{'term': 'Lidocaine Hydrochloride', 'count': 169},
{'term': 'Potassium Chloride', 'count': 169},
{'term': 'Methylphenidate Hydrochloride', 'count': 168},
{'term': 'Topiramate', 'count': 165},
{'term': 'Diclofenac Sodium', 'count': 164},
{'term': 'Hydrochlorothiazide', 'count': 164},
{'term': 'Quetiapine Fumarate', 'count': 160},
{'term': 'DOCUSATE SODIUM', 'count': 159},
{'term': 'Celecoxib', 'count': 156},
```

```
        {'term': 'Diltiazem Hydrochloride', 'count': 155},
        {'term': 'Naproxen', 'count': 155},
        {'term': 'ACETAMINOPHEN', 'count': 154},
        {'term': 'Alprazolam', 'count': 154},
        {'term': 'Rosuvastatin Calcium', 'count': 154},
        {'term': 'Amoxicillin and Clavulanate Potassium', 'count': 153},
        {'term': 'Levofloxacin', 'count': 153},
        {'term': 'Acyclovir', 'count': 152},
        {'term': 'Clotrimazole', 'count': 151},
        {'term': 'Lorazepam', 'count': 149},
        {'term': 'Buspirone Hydrochloride', 'count': 148},
        {'term': 'DIPHENHYDRAMINE HYDROCHLORIDE', 'count': 146},
        {'term': 'Glipizide', 'count': 146},
        {'term': 'Trazodone Hydrochloride', 'count': 146},
        {'term': 'CHLOROXYLENOL', 'count': 142},
        {'term': 'Fenofibrate', 'count': 142},
        {'term': 'Methocarbamol', 'count': 141},
        {'term': 'Metoprolol Tartrate', 'count': 139},
        {'term': 'Venlafaxine Hydrochloride', 'count': 138},
        {'term': 'Bisacodyl', 'count': 137}
    ]
}
```


Date of API results: **2022-05-20**


```
[
    [3032, 'ALCOHOL'],
    [1445, 'Alcohol'],
    [1046, 'Ethyl Alcohol'],
    [1017, 'Ibuprofen'],
    [1005, 'Acetaminophen'],
    [898, 'Benzalkonium Chloride'],
    [779, 'BENZALKONIUM CHLORIDE'],
    [723, 'Zinc Oxide'],
    [636, 'Menthol'],
    [586, 'Isopropyl Alcohol'],
    [567, 'Sodium Fluoride'],
    [508, 'Salicylic Acid'],
    [448, 'Oxygen'],
    [435, 'ETHYL ALCOHOL'],
    [417, 'Benzocaine'],
    [403, 'Aspirin'],
    [395, 'Gabapentin'],
    [377, 'Nicotine Polacrilex'],
    [366, 'ZINC OXIDE'],
    [347, 'Diphenhydramine HCl'],
    [327, 'SALICYLIC ACID'],
```

```
[313, 'Avobenzone, Homosalate, Octisalate, Octocrylene'],
[311, 'Lisinopril'],
[306, 'MENTHOL'],
[294, 'Levothyroxine Sodium'],
[285, 'Hydrocortisone'],
[284, 'Naproxen Sodium'],
[279, 'TITANIUM DIOXIDE'],
[267, 'Loratadine'],
[265, 'Benzalkonium chloride'],
[262, 'Simethicone'],
[261, 'Aripiprazole'],
[258, 'Famotidine'],
[258, 'OCTINOXATE, TITANIUM DIOXIDE'],
[250, 'OCTINOXATE and TITANIUM DIOXIDE'],
[249, 'Titanium Dioxide and Zinc Oxide'],
[247, 'Hand Sanitizer'],
[247, 'Lamotrigine'],
[244, 'Prednisone'],
[240, 'Diphenhydramine Hydrochloride'],
[240, 'Omeprazole'],
[234, 'PREGABALIN'],
[233, 'Guaifenesin'],
[229, 'Levetiracetam'],
[228, 'Amoxicillin'],
[225, 'Calcium Carbonate'],
[225, 'ISOPROPYL ALCOHOL'],
[225, 'Titanium Dioxide, Zinc Oxide'],
[224, 'alcohol'],
[223, 'Pyrithione Zinc'],
[222, 'Titanium Dioxide'],
[218, 'Lidocaine'],
[217, 'Metformin Hydrochloride'],
[216, 'Tadalafil'],
[213, 'Cetirizine Hydrochloride'],
[213, 'Ethyl alcohol'],
[211, 'Chloroxylenol'],
[209, 'TITANIUM DIOXIDE, ZINC OXIDE'],
[206, 'Bismuth subsalicylate'],
[205, 'Olanzapine'],
[202, 'Losartan Potassium'],
[202, 'SODIUM FLUORIDE'],
[200, 'Pregabalin'],
[190, 'levothyroxine sodium'],
[188, 'Hydrocodone Bitartrate and Acetaminophen'],
[183, 'Aluminum Chlorohydrate'],
[181, 'Dimethicone'],
```

```
    [177, 'Carvedilol'],
    [173, 'Fluconazole'],
    [172, 'ATORVASTATIN CALCIUM'],
    [171, 'Avobenzone, Homosalate, Octisalate, Octocrylene, Oxybenzone'],
    [169, 'Lidocaine Hydrochloride'],
    [169, 'Potassium Chloride'],
    [168, 'Methylphenidate Hydrochloride'],
    [165, 'Topiramate'],
    [164, 'Diclofenac Sodium'],
    [164, 'Hydrochlorothiazide'],
    [160, 'Quetiapine Fumarate'],
    [159, 'DOCUSATE SODIUM'],
    [156, 'Celecoxib'],
    [155, 'Diltiazem Hydrochloride'],
    [155, 'Naproxen'],
    [154, 'ACETAMINOPHEN'],
    [154, 'Alprazolam'],
    [154, 'Rosuvastatin Calcium'],
    [153, 'Amoxicillin and Clavulanate Potassium'],
    [153, 'Levofloxacin'],
    [152, 'Acyclovir'],
    [151, 'Clotrimazole'],
    [149, 'Lorazepam'],
    [148, 'Buspirone Hydrochloride'],
    [146, 'DIPHENHYDRAMINE HYDROCHLORIDE'],
    [146, 'Glipizide'],
    [146, 'Trazodone Hydrochloride'],
    [142, 'CHLOROXYLENOL'],
    [142, 'Fenofibrate'],
    [141, 'Methocarbamol'],
    [139, 'Metoprolol Tartrate'],
    [138, 'Venlafaxine Hydrochloride'],
    [137, 'Bisacodyl']
]
```

The size of generic_name_df: (100, 2)

The mean generics count in generic_name_df: 317.61

The contents of generic_name_df:

	count	term
0	3032	ALCOHOL
1	1445	Alcohol

2	1046	Ethyl Alcohol
3	1017	Ibuprofen
4	1005	Acetaminophen
...	...	...
95	142	Fenofibrate
96	141	Methocarbamol
97	139	Metoprolol Tartrate
98	138	Venlafaxine Hydrochloride

Notes and Resources

Want some ideas for what to do next? Here are our suggestions:

1. For more about the `pandas` package, including the methods used above, see the following:
   - https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.shape.html
   - https://pandas.pydata.org/docs/reference/api/pandas.Series.median.html
   - https://pandas.pydata.org/docs/reference/api/pandas.Series.value_counts.html
2. For more about the `requests` package, see https://docs.python-requests.org/
3. For more about the `rich` package, see https://rich.readthedocs.io/
4. For more about some of the Python features used, such as dictionaries, lists, and control flow with if-then-else conditionals and for-loops, we recommend the following chapters of A Whirlwind Tour of Python:
   - https://jakevdp.github.io/WhirlwindTourOfPython/06-built-in-data-structures.html
   - https://jakevdp.github.io/WhirlwindTourOfPython/07-control-flow-statements.html

5. For more information on f-strings (i.e., Python strings like `f'https://httpstatuses.com/{http_status}'`), see https://realpython.com/python-f-strings/.
6. For background on the HTTP Request/Response Cycle, we recommend the following:
    - Brief Overview: https://backend.turing.edu/module2/lessons/how_the_web_works_http
    - Deeper Overview: https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview
    - Summary of HTTP Status Codes: https://httpstatuses.com/
    - Google's Implementation of HTTP Status Code 418: https://www.google.com/teapot
7. For more practice with open web APIs, we recommend looking through https://github.com/public-apis/public-apis and trying to parse the output from http://deckofcardsapi.com/
8. For more about the complexity of parsing JSON in SAS, see https://blogs.sas.com/content/sasdummy/2016/12/02/json-libname-engine-sas/
9. We welcome follow-up conversations. You can connect with us on LinkedIn or email us at isaiah.lankham@gmail.com and matthew.t.slaughter@gmail.com
10. If you have a GitHub account (or don't mind creating one), you can also chat with us on Gitter at https://gitter.im/saspy-bffs/community