In [1]:
```
!echo %JAVA_HOME%
```

```
D:\jdk-17.0.16
```

In [2]:
```
!echo %PYSPARK_PYTHON%
```

```
python
```

In [3]:
```
!echo %SPARK_HOME%
```

```
D:\spark-4.0.1-bin-hadoop3
```

In [4]:
```
!echo %HADOOP_HOME%
```

```
D:\hadoop-3.3.6
```

In [5]:
```
!echo %PATH%
```

```
C:\Users\DELL\.conda\envs\spark310;C:\Users\DELL\.conda\envs\spark310\Library\mingw-
w64\bin;C:\Users\DELL\.conda\envs\spark310\Library\usr\bin;C:\Users\DELL\.conda\envs
\spark310\Library\bin;C:\Users\DELL\.conda\envs\spark310\Scripts;C:\Users\DELL\.cond
a\envs\spark310\bin;F:\ProgramData\anaconda3\condabin;F:\ProgramData\anaconda3;F:\Pr
ogramData\anaconda3\Library\mingw-w64\bin;F:\ProgramData\anaconda3\Library\usr\bin;
F:\ProgramData\anaconda3\Library\bin;F:\ProgramData\anaconda3\Scripts;F:\Program Fil
es (x86)\VMware\VMware Workstation\bin;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\Sys
tem32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0;C:\WINDOWS\System32\OpenSSH;
C:\WINDOWS\system32\config\systemprofile\.dnx\bin;C:\Program Files\Microsoft DNX\Dnv
m;C:\Program Files (x86)\Windows Kits\8.1\Windows Performance Toolkit;C:\Program Fil
es\Microsoft SQL Server\120\Tools\Binn;C:\Program Files (x86)\Microsoft SQL Server\1
50\DTS\Binn;C:\Program Files\Azure Data Studio\bin;C:\Program Files (x86)\nodejs;
D:\jdk-17.0.16\bin;D:\hadoop-3.3.6\bin;D:\spark-4.0.1-bin-hadoop3\bin;.;C:\Program F
iles\Docker\Docker\resources\bin;C:\Users\DELL\AppData\Local\Microsoft\WindowsApp
s;.;C:\Program Files\Azure Data Studio\bin;C:\Users\DELL\AppData\Roaming\npm;C:\User
s\DELL\AppData\Local\Programs\Ollama
```

pip install pattern

In [1]:
```
from pyspark.sql import SparkSession
```

In [2]:
```
# initialization of spark context
spark = SparkSession.builder.appName("pySpWordCount").getOrCreate()
sc = spark.sparkContext
```

In [3]:
```
txtFile = sc.textFile("file:///D://DataSets//SpVsPd.txt")
```

Get Words

In [4]:
```
words = txtFile.flatMap(lambda line:line.lower().split(" "))
```

wrds = words.collect() print(len(wrds)) print(wrds)

In [5]:
```
wrds =  [row for row in words.collect()]
print(len(wrds))
print(wrds)
```

603

['1.', 'definitions', '1.1', 'what', 'is', 'pyspark?', 'pyspark', 'is', 'the', 'pyth
on', 'library', 'for', 'spark', 'programming.', 'it', 'allows', 'you', 'to', 'use',
'the', 'powerful', 'and', 'efficient', 'data', 'processing', 'capabilities', 'of',
'apache', 'spark', 'from', 'within', 'the', 'python', 'programming', 'language.', 'p
yspark', 'provides', 'a', 'high-level', 'api', 'for', 'distributed', 'data', 'proces
sing', 'that', 'can', 'be', 'used', 'to', 'perform', 'common', 'data', 'analysis',
'tasks,', 'such', 'as', 'filtering,', 'aggregation,', 'and', 'transformation', 'of',
'large', 'datasets.', '', '1.2', 'what', 'is', 'pandas?', 'pandas', 'is', 'a', 'pyth
on', 'library', 'for', 'data', 'manipulation', 'and', 'analysis.', 'it', 'provides',
'powerful', 'data', 'structures,', 'such', 'as', 'the', 'dataframe', 'and', 'serie
s,', 'that', 'are', 'designed', 'to', 'make', 'it', 'easy', 'to', 'work', 'with', 's
tructured', 'data', 'in', 'python.', 'with', 'pandas,', 'you', 'can', 'perform',
'a', 'wide', 'range', 'of', 'data', 'analysis', 'tasks,', 'such', 'as', 'filterin
g,', 'aggregation,', 'and', 'transformation', 'of', 'data,', 'as', 'well', 'as', 'da
ta', 'cleaning', 'and', 'preparation.', '', 'both', 'definitions', 'look', 'more',
'or', 'less', 'the', 'same,', 'but', 'there', 'is', 'a', 'difference', 'in', 'thei
r', 'execution', 'and', 'processing', 'architecture.', 'let’s', 'go', 'over', 'som
e', 'major', 'differences', 'between', 'these', 'two.', '', '2.', 'key', 'difference
s', 'between', 'pyspark', 'and', 'pandas', 'pyspark', 'is', 'a', 'library', 'for',
'working', 'with', 'large', 'datasets', 'in', 'a', 'distributed', 'computing', 'envi
ronment,', 'while', 'pandas', 'is', 'a', 'library', 'for', 'working', 'with', 'small
er,', 'tabular', 'datasets', 'on', 'a', 'single', 'machine.', 'pyspark', 'is', 'buil
t', 'on', 'top', 'of', 'the', 'apache', 'spark', 'framework', 'and', 'uses', 'the',
'resilient', 'distributed', 'datasets', '(rdd)', 'data', 'structure,', 'while', 'pan
das', 'uses', 'the', 'dataframe', 'data', 'structure.', 'pyspark', 'is', 'designed',
'to', 'handle', 'data', 'processing', 'tasks', 'that', 'are', 'not', 'feasible', 'wi
th', 'pandas', 'due', 'to', 'memory', 'constraints,', 'such', 'as', 'iterative', 'al
gorithms', 'and', 'machine', 'learning', 'on', 'large', 'datasets.', 'pyspark', 'all
ows', 'for', 'parallel', 'processing', 'of', 'data,', 'while', 'pandas', 'does', 'no
t.', 'pyspark', 'can', 'read', 'data', 'from', 'a', 'variety', 'of', 'sources,', 'in
cluding', 'hadoop', 'distributed', 'file', 'system', '(hdfs),', 'amazon', 's3,', 'an
d', 'local', 'file', 'systems,', 'while', 'pandas', 'is', 'limited', 'to', 'readin
g', 'data', 'from', 'local', 'file', 'systems.', 'pyspark', 'can', 'be', 'integrate
d', 'with', 'other', 'big', 'data', 'tools', 'like', 'hadoop', 'and', 'hive,', 'whil
e', 'pandas', 'is', 'not.', 'pyspark', 'is', 'written', 'in', 'scala,', 'and', 'run
s', 'on', 'the', 'java', 'virtual', 'machine', '(jvm),', 'while', 'pandas', 'is', 'w
ritten', 'in', 'python.', 'pyspark', 'has', 'a', 'steeper', 'learning', 'curve', 'th
an', 'pandas,', 'due', 'to', 'the', 'additional', 'concepts', 'and', 'technologies',
'involved', '(e.g.', 'distributed', 'computing,', 'rdds,', 'spark', 'sql,', 'spark',
'streaming,', 'etc.).', 'how', 'to', 'decide', 'which', 'library', 'to', 'use', '—',
'pyspark', 'vs', 'pandas', 'the', 'decision', 'of', 'whether', 'to', 'use', 'pyspar
k', 'or', 'pandas', 'depends', 'on', 'the', 'size', 'and', 'complexity', 'of', 'th
e', 'dataset', 'and', 'the', 'specific', 'task', 'you', 'want', 'to', 'perform.',
'', 'size', 'of', 'the', 'dataset:', 'pyspark', 'is', 'designed', 'to', 'handle', 'l
arge', 'datasets', 'that', 'are', 'not', 'feasible', 'to', 'work', 'with', 'on',
'a', 'single', 'machine', 'using', 'pandas.', 'if', 'you', 'have', 'a', 'dataset',
'that', 'is', 'too', 'large', 'to', 'fit', 'in', 'memory,', 'or', 'if', 'you', 'nee
d', 'to', 'perform', 'iterative', 'or', 'distributed', 'computations,', 'pyspark',
'is', 'the', 'better', 'choice.', 'complexity', 'of', 'the', 'task:', 'pyspark', 'i
s', 'a', 'powerful', 'tool', 'for', 'big', 'data', 'processing', 'and', 'allows', 'y
ou', 'to', 'perform', 'a', 'wide', 'range', 'of', 'data', 'processing', 'tasks,', 's
uch', 'as', 'machine', 'learning,', 'graph', 'processing,', 'and', 'stream', 'proces
sing.', 'if', 'you', 'need', 'to', 'perform', 'any', 'of', 'these', 'tasks,', 'pyspa
rk', 'is', 'the', 'better', 'choice.', 'learning', 'curve:', 'pyspark', 'has', 'a',
'steeper', 'learning', 'curve', 'than', 'pandas,', 'as', 'it', 'requires', 'knowledg

```
e', 'of', 'distributed', 'computing,', 'rdds,', 'and', 'spark', 'sql.', 'if', 'you',
'are', 'new', 'to', 'big', 'data', 'processing', 'and', 'want', 'to', 'get', 'starte
d', 'quickly,', 'pandas', 'may', 'be', 'the', 'better', 'choice.', 'resources', 'ava
ilable:', 'pyspark', 'requires', 'a', 'cluster', 'or', 'distributed', 'system', 't
o', 'run,', 'so', 'you', 'will', 'need', 'access', 'to', 'the', 'appropriate', 'infr
astructure', 'and', 'resources.', 'if', 'you', 'do', 'not', 'have', 'access', 'to',
'these', 'resources,', 'then', 'pandas', 'is', 'a', 'good', 'choice.', 'in', 'summar
y,', 'use', 'pyspark', 'for', 'large', 'datasets', 'and', 'complex', 'tasks', 'tha
t', 'are', 'not', 'feasible', 'with', 'pandas,', 'and', 'use', 'pandas', 'for', 'sma
ll', 'datasets', 'and', 'simple', 'tasks', 'that', 'can', 'be', 'handled', 'on',
'a', 'single', 'machine.']
```

for word in wrds: print(word)

Remove puctuations and special characters

Remove puctuations and special characters

In [6]:
```
from string import punctuation
```

In [8]:
```
import re
```

In [10]:
```
wordp=words.map(lambda word: re.sub(f"[{re.escape(punctuation)}]","",word))
```

import repunc = '[!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~’]' wordp= words.map(lambda word: re.sub(punc,"",word))

In [11]:
```
wrdp = wordp.collect()
print(len(wrdp))
print(wrdp)
```

```
603
['1', 'definitions', '11', 'what', 'is', 'pyspark', 'pyspark', 'is', 'the', 'pytho
n', 'library', 'for', 'spark', 'programming', 'it', 'allows', 'you', 'to', 'use', 't
he', 'powerful', 'and', 'efficient', 'data', 'processing', 'capabilities', 'of', 'ap
ache', 'spark', 'from', 'within', 'the', 'python', 'programming', 'language', 'pyspa
rk', 'provides', 'a', 'highlevel', 'api', 'for', 'distributed', 'data', 'processin
g', 'that', 'can', 'be', 'used', 'to', 'perform', 'common', 'data', 'analysis', 'tas
ks', 'such', 'as', 'filtering', 'aggregation', 'and', 'transformation', 'of', 'larg
e', 'datasets', '', '12', 'what', 'is', 'pandas', 'pandas', 'is', 'a', 'python', 'li
brary', 'for', 'data', 'manipulation', 'and', 'analysis', 'it', 'provides', 'powerfu
l', 'data', 'structures', 'such', 'as', 'the', 'dataframe', 'and', 'series', 'that',
'are', 'designed', 'to', 'make', 'it', 'easy', 'to', 'work', 'with', 'structured',
'data', 'in', 'python', 'with', 'pandas', 'you', 'can', 'perform', 'a', 'wide', 'ran
ge', 'of', 'data', 'analysis', 'tasks', 'such', 'as', 'filtering', 'aggregation', 'a
nd', 'transformation', 'of', 'data', 'as', 'well', 'as', 'data', 'cleaning', 'and',
'preparation', '', 'both', 'definitions', 'look', 'more', 'or', 'less', 'the', 'sam
e', 'but', 'there', 'is', 'a', 'difference', 'in', 'their', 'execution', 'and', 'pro
cessing', 'architecture', 'let’s', 'go', 'over', 'some', 'major', 'differences', 'be
tween', 'these', 'two', '', '2', 'key', 'differences', 'between', 'pyspark', 'and',
'pandas', 'pyspark', 'is', 'a', 'library', 'for', 'working', 'with', 'large', 'datas
ets', 'in', 'a', 'distributed', 'computing', 'environment', 'while', 'pandas', 'is',
'a', 'library', 'for', 'working', 'with', 'smaller', 'tabular', 'datasets', 'on',
'a', 'single', 'machine', 'pyspark', 'is', 'built', 'on', 'top', 'of', 'the', 'apach
e', 'spark', 'framework', 'and', 'uses', 'the', 'resilient', 'distributed', 'dataset
s', 'rdd', 'data', 'structure', 'while', 'pandas', 'uses', 'the', 'dataframe', 'dat
a', 'structure', 'pyspark', 'is', 'designed', 'to', 'handle', 'data', 'processing',
'tasks', 'that', 'are', 'not', 'feasible', 'with', 'pandas', 'due', 'to', 'memory',
'constraints', 'such', 'as', 'iterative', 'algorithms', 'and', 'machine', 'learnin
g', 'on', 'large', 'datasets', 'pyspark', 'allows', 'for', 'parallel', 'processing',
'of', 'data', 'while', 'pandas', 'does', 'not', 'pyspark', 'can', 'read', 'data', 'f
rom', 'a', 'variety', 'of', 'sources', 'including', 'hadoop', 'distributed', 'file',
'system', 'hdfs', 'amazon', 's3', 'and', 'local', 'file', 'systems', 'while', 'panda
s', 'is', 'limited', 'to', 'reading', 'data', 'from', 'local', 'file', 'systems', 'p
yspark', 'can', 'be', 'integrated', 'with', 'other', 'big', 'data', 'tools', 'like',
'hadoop', 'and', 'hive', 'while', 'pandas', 'is', 'not', 'pyspark', 'is', 'written',
'in', 'scala', 'and', 'runs', 'on', 'the', 'java', 'virtual', 'machine', 'jvm', 'whi
le', 'pandas', 'is', 'written', 'in', 'python', 'pyspark', 'has', 'a', 'steeper', 'l
earning', 'curve', 'than', 'pandas', 'due', 'to', 'the', 'additional', 'concepts',
'and', 'technologies', 'involved', 'eg', 'distributed', 'computing', 'rdds', 'spar
k', 'sql', 'spark', 'streaming', 'etc', 'how', 'to', 'decide', 'which', 'library',
'to', 'use', '—', 'pyspark', 'vs', 'pandas', 'the', 'decision', 'of', 'whether', 't
o', 'use', 'pyspark', 'or', 'pandas', 'depends', 'on', 'the', 'size', 'and', 'comple
xity', 'of', 'the', 'dataset', 'and', 'the', 'specific', 'task', 'you', 'want', 't
o', 'perform', '', 'size', 'of', 'the', 'dataset', 'pyspark', 'is', 'designed', 't
o', 'handle', 'large', 'datasets', 'that', 'are', 'not', 'feasible', 'to', 'work',
'with', 'on', 'a', 'single', 'machine', 'using', 'pandas', 'if', 'you', 'have', 'a',
'dataset', 'that', 'is', 'too', 'large', 'to', 'fit', 'in', 'memory', 'or', 'if', 'y
ou', 'need', 'to', 'perform', 'iterative', 'or', 'distributed', 'computations', 'pys
park', 'is', 'the', 'better', 'choice', 'complexity', 'of', 'the', 'task', 'pyspar
k', 'is', 'a', 'powerful', 'tool', 'for', 'big', 'data', 'processing', 'and', 'allow
s', 'you', 'to', 'perform', 'a', 'wide', 'range', 'of', 'data', 'processing', 'task
s', 'such', 'as', 'machine', 'learning', 'graph', 'processing', 'and', 'stream', 'pr
ocessing', 'if', 'you', 'need', 'to', 'perform', 'any', 'of', 'these', 'tasks', 'pys
park', 'is', 'the', 'better', 'choice', 'learning', 'curve', 'pyspark', 'has', 'a',
'steeper', 'learning', 'curve', 'than', 'pandas', 'as', 'it', 'requires', 'knowledg
e', 'of', 'distributed', 'computing', 'rdds', 'and', 'spark', 'sql', 'if', 'you', 'a
```

```
re', 'new', 'to', 'big', 'data', 'processing', 'and', 'want', 'to', 'get', 'starte
d', 'quickly', 'pandas', 'may', 'be', 'the', 'better', 'choice', 'resources', 'avail
able', 'pyspark', 'requires', 'a', 'cluster', 'or', 'distributed', 'system', 'to',
'run', 'so', 'you', 'will', 'need', 'access', 'to', 'the', 'appropriate', 'infrastru
cture', 'and', 'resources', 'if', 'you', 'do', 'not', 'have', 'access', 'to', 'thes
e', 'resources', 'then', 'pandas', 'is', 'a', 'good', 'choice', 'in', 'summary', 'us
e', 'pyspark', 'for', 'large', 'datasets', 'and', 'complex', 'tasks', 'that', 'are',
'not', 'feasible', 'with', 'pandas', 'and', 'use', 'pandas', 'for', 'small', 'datase
ts', 'and', 'simple', 'tasks', 'that', 'can', 'be', 'handled', 'on', 'a', 'single',
'machine']
```

Remove Stop Words

pip install nltk

In [14]:
```
import nltk
```

In [15]:
```
from nltk.corpus import stopwords
#nltk.download('stopwords')
#nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\DELL\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\DELL\AppData\Roaming\nltk_data...
```

Out[15]: True

In [17]:
```
stp_words = list(stopwords.words('english'))
print(len(stp_words))
print(sorted(stp_words))
```

```
198
['a', 'about', 'above', 'after', 'again', 'against', 'ain', 'all', 'am', 'an', 'an
d', 'any', 'are', 'aren', "aren't", 'as', 'at', 'be', 'because', 'been', 'before',
'being', 'below', 'between', 'both', 'but', 'by', 'can', 'couldn', "couldn't", 'd',
'did', 'didn', "didn't", 'do', 'does', 'doesn', "doesn't", 'doing', 'don', "don't",
'down', 'during', 'each', 'few', 'for', 'from', 'further', 'had', 'hadn', "hadn't",
'has', 'hasn', "hasn't", 'have', 'haven', "haven't", 'having', 'he', "he'd", "he'l
l", "he's", 'her', 'here', 'hers', 'herself', 'him', 'himself', 'his', 'how', 'i',
"i'd", "i'll", "i'm", "i've", 'if', 'in', 'into', 'is', 'isn', "isn't", 'it', "i
t'd", "it'll", "it's", 'its', 'itself', 'just', 'll', 'm', 'ma', 'me', 'mightn', "mi
ghtn't", 'more', 'most', 'mustn', "mustn't", 'my', 'myself', 'needn', "needn't", 'n
o', 'nor', 'not', 'now', 'o', 'of', 'off', 'on', 'once', 'only', 'or', 'other', 'ou
r', 'ours', 'ourselves', 'out', 'over', 'own', 're', 's', 'same', 'shan', "shan't",
'she', "she'd", "she'll", "she's", 'should', "should've", 'shouldn', "shouldn't", 's
o', 'some', 'such', 't', 'than', 'that', "that'll", 'the', 'their', 'theirs', 'the
m', 'themselves', 'then', 'there', 'these', 'they', "they'd", "they'll", "they're",
"they've", 'this', 'those', 'through', 'to', 'too', 'under', 'until', 'up', 've', 'v
ery', 'was', 'wasn', "wasn't", 'we', "we'd", "we'll", "we're", "we've", 'were', 'wer
en', "weren't", 'what', 'when', 'where', 'which', 'while', 'who', 'whom', 'why', 'wi
ll', 'with', 'won', "won't", 'wouldn', "wouldn't", 'y', 'you', "you'd", "you'll", "y
ou're", "you've", 'your', 'yours', 'yourself', 'yourselves']
```

In [18]:
```
words_wo_sw = wordp.filter(lambda x: (x not in stp_words) and (x != ''))
```

In [19]:
```
wrds_wo_sw = words_wo_sw.collect()
print(len(wrds_wo_sw))
print(wrds_wo_sw)
```

```
346
['1', 'definitions', '11', 'pyspark', 'pyspark', 'python', 'library', 'spark', 'prog
ramming', 'allows', 'use', 'powerful', 'efficient', 'data', 'processing', 'capabilit
ies', 'apache', 'spark', 'within', 'python', 'programming', 'language', 'pyspark',
'provides', 'highlevel', 'api', 'distributed', 'data', 'processing', 'used', 'perfor
m', 'common', 'data', 'analysis', 'tasks', 'filtering', 'aggregation', 'transformati
on', 'large', 'datasets', '12', 'pandas', 'pandas', 'python', 'library', 'data', 'ma
nipulation', 'analysis', 'provides', 'powerful', 'data', 'structures', 'dataframe',
'series', 'designed', 'make', 'easy', 'work', 'structured', 'data', 'python', 'panda
s', 'perform', 'wide', 'range', 'data', 'analysis', 'tasks', 'filtering', 'aggregati
on', 'transformation', 'data', 'well', 'data', 'cleaning', 'preparation', 'definitio
ns', 'look', 'less', 'difference', 'execution', 'processing', 'architecture', 'le
t’s', 'go', 'major', 'differences', 'two', '2', 'key', 'differences', 'pyspark', 'pa
ndas', 'pyspark', 'library', 'working', 'large', 'datasets', 'distributed', 'computi
ng', 'environment', 'pandas', 'library', 'working', 'smaller', 'tabular', 'dataset
s', 'single', 'machine', 'pyspark', 'built', 'top', 'apache', 'spark', 'framework',
'uses', 'resilient', 'distributed', 'datasets', 'rdd', 'data', 'structure', 'panda
s', 'uses', 'dataframe', 'data', 'structure', 'pyspark', 'designed', 'handle', 'dat
a', 'processing', 'tasks', 'feasible', 'pandas', 'due', 'memory', 'constraints', 'it
erative', 'algorithms', 'machine', 'learning', 'large', 'datasets', 'pyspark', 'allo
ws', 'parallel', 'processing', 'data', 'pandas', 'pyspark', 'read', 'data', 'variet
y', 'sources', 'including', 'hadoop', 'distributed', 'file', 'system', 'hdfs', 'amaz
on', 's3', 'local', 'file', 'systems', 'pandas', 'limited', 'reading', 'data', 'loca
l', 'file', 'systems', 'pyspark', 'integrated', 'big', 'data', 'tools', 'like', 'had
oop', 'hive', 'pandas', 'pyspark', 'written', 'scala', 'runs', 'java', 'virtual', 'm
achine', 'jvm', 'pandas', 'written', 'python', 'pyspark', 'steeper', 'learning', 'cu
rve', 'pandas', 'due', 'additional', 'concepts', 'technologies', 'involved', 'eg',
'distributed', 'computing', 'rdds', 'spark', 'sql', 'spark', 'streaming', 'etc', 'de
cide', 'library', 'use', '—', 'pyspark', 'vs', 'pandas', 'decision', 'whether', 'us
e', 'pyspark', 'pandas', 'depends', 'size', 'complexity', 'dataset', 'specific', 'ta
sk', 'want', 'perform', 'size', 'dataset', 'pyspark', 'designed', 'handle', 'large',
'datasets', 'feasible', 'work', 'single', 'machine', 'using', 'pandas', 'dataset',
'large', 'fit', 'memory', 'need', 'perform', 'iterative', 'distributed', 'computatio
ns', 'pyspark', 'better', 'choice', 'complexity', 'task', 'pyspark', 'powerful', 'to
ol', 'big', 'data', 'processing', 'allows', 'perform', 'wide', 'range', 'data', 'pro
cessing', 'tasks', 'machine', 'learning', 'graph', 'processing', 'stream', 'processi
ng', 'need', 'perform', 'tasks', 'pyspark', 'better', 'choice', 'learning', 'curve',
'pyspark', 'steeper', 'learning', 'curve', 'pandas', 'requires', 'knowledge', 'distr
ibuted', 'computing', 'rdds', 'spark', 'sql', 'new', 'big', 'data', 'processing', 'w
ant', 'get', 'started', 'quickly', 'pandas', 'may', 'better', 'choice', 'resources',
'available', 'pyspark', 'requires', 'cluster', 'distributed', 'system', 'run', 'nee
d', 'access', 'appropriate', 'infrastructure', 'resources', 'access', 'resources',
'pandas', 'good', 'choice', 'summary', 'use', 'pyspark', 'large', 'datasets', 'compl
ex', 'tasks', 'feasible', 'pandas', 'use', 'pandas', 'small', 'datasets', 'simple',
'tasks', 'handled', 'single', 'machine']
```

Convert plural to Singular

In [20]:
```
from nltk.stem import WordNetLemmatizer
```

In [21]:
```
lemmatizer = WordNetLemmatizer()
```

In [22]:
```
def singularize(word):
    lemma = lemmatizer.lemmatize(word, pos='n')  # 'n' specifies the part of speech
    return lemma
```

In [23]:
```
word_s= words_wo_sw.map(lambda word: singularize(word))
```

from pattern.en import singularize word_s= words_wo_sw.map(lambda word: singularize(word))

In [18]:
```
wrd_s = word_s.collect()
print(len(wrd_s))
print(wrd_s)
```

```
346
['1', 'definition', '11', 'pyspark', 'pyspark', 'python', 'library', 'spark', 'progr
amming', 'allows', 'use', 'powerful', 'efficient', 'data', 'processing', 'capabilit
y', 'apache', 'spark', 'within', 'python', 'programming', 'language', 'pyspark', 'pr
ovides', 'highlevel', 'api', 'distributed', 'data', 'processing', 'used', 'perform',
'common', 'data', 'analysis', 'task', 'filtering', 'aggregation', 'transformation',
'large', 'datasets', '12', 'panda', 'panda', 'python', 'library', 'data', 'manipulat
ion', 'analysis', 'provides', 'powerful', 'data', 'structure', 'dataframe', 'serie
s', 'designed', 'make', 'easy', 'work', 'structured', 'data', 'python', 'panda', 'pe
rform', 'wide', 'range', 'data', 'analysis', 'task', 'filtering', 'aggregation', 'tr
ansformation', 'data', 'well', 'data', 'cleaning', 'preparation', 'definition', 'loo
k', 'le', 'difference', 'execution', 'processing', 'architecture', 'let', 'go', 'maj
or', 'difference', 'two', '2', 'key', 'difference', 'pyspark', 'panda', 'pyspark',
'library', 'working', 'large', 'datasets', 'distributed', 'computing', 'environmen
t', 'panda', 'library', 'working', 'smaller', 'tabular', 'datasets', 'single', 'mach
ine', 'pyspark', 'built', 'top', 'apache', 'spark', 'framework', 'us', 'resilient',
'distributed', 'datasets', 'rdd', 'data', 'structure', 'panda', 'us', 'dataframe',
'data', 'structure', 'pyspark', 'designed', 'handle', 'data', 'processing', 'task',
'feasible', 'panda', 'due', 'memory', 'constraint', 'iterative', 'algorithm', 'machi
ne', 'learning', 'large', 'datasets', 'pyspark', 'allows', 'parallel', 'processing',
'data', 'panda', 'pyspark', 'read', 'data', 'variety', 'source', 'including', 'hadoo
p', 'distributed', 'file', 'system', 'hdfs', 'amazon', 's3', 'local', 'file', 'syste
m', 'panda', 'limited', 'reading', 'data', 'local', 'file', 'system', 'pyspark', 'in
tegrated', 'big', 'data', 'tool', 'like', 'hadoop', 'hive', 'panda', 'pyspark', 'wri
tten', 'scala', 'run', 'java', 'virtual', 'machine', 'jvm', 'panda', 'written', 'pyt
hon', 'pyspark', 'steeper', 'learning', 'curve', 'panda', 'due', 'additional', 'conc
ept', 'technology', 'involved', 'eg', 'distributed', 'computing', 'rdds', 'spark',
'sql', 'spark', 'streaming', 'etc', 'decide', 'library', 'use', '—', 'pyspark', 'v',
'panda', 'decision', 'whether', 'use', 'pyspark', 'panda', 'depends', 'size', 'compl
exity', 'dataset', 'specific', 'task', 'want', 'perform', 'size', 'dataset', 'pyspar
k', 'designed', 'handle', 'large', 'datasets', 'feasible', 'work', 'single', 'machin
e', 'using', 'panda', 'dataset', 'large', 'fit', 'memory', 'need', 'perform', 'itera
tive', 'distributed', 'computation', 'pyspark', 'better', 'choice', 'complexity', 't
ask', 'pyspark', 'powerful', 'tool', 'big', 'data', 'processing', 'allows', 'perfor
m', 'wide', 'range', 'data', 'processing', 'task', 'machine', 'learning', 'graph',
'processing', 'stream', 'processing', 'need', 'perform', 'task', 'pyspark', 'bette
r', 'choice', 'learning', 'curve', 'pyspark', 'steeper', 'learning', 'curve', 'pand
a', 'requires', 'knowledge', 'distributed', 'computing', 'rdds', 'spark', 'sql', 'ne
w', 'big', 'data', 'processing', 'want', 'get', 'started', 'quickly', 'panda', 'ma
y', 'better', 'choice', 'resource', 'available', 'pyspark', 'requires', 'cluster',
'distributed', 'system', 'run', 'need', 'access', 'appropriate', 'infrastructure',
'resource', 'access', 'resource', 'panda', 'good', 'choice', 'summary', 'use', 'pysp
ark', 'large', 'datasets', 'complex', 'task', 'feasible', 'panda', 'use', 'panda',
'small', 'datasets', 'simple', 'task', 'handled', 'single', 'machine']
```

Remove Numeric Values

In [19]:
```
words_wo_num = word_s.filter(lambda word: not re.search(r'\d', word))
```

In [20]:
```
wrds_wo_num = words_wo_num.collect()
print(len(wrds_wo_num))
print(wrds_wo_num)
```

```
341
['definition', 'pyspark', 'pyspark', 'python', 'library', 'spark', 'programming', 'a
llows', 'use', 'powerful', 'efficient', 'data', 'processing', 'capability', 'apach
e', 'spark', 'within', 'python', 'programming', 'language', 'pyspark', 'provides',
'highlevel', 'api', 'distributed', 'data', 'processing', 'used', 'perform', 'commo
n', 'data', 'analysis', 'task', 'filtering', 'aggregation', 'transformation', 'larg
e', 'datasets', 'panda', 'panda', 'python', 'library', 'data', 'manipulation', 'anal
ysis', 'provides', 'powerful', 'data', 'structure', 'dataframe', 'series', 'designe
d', 'make', 'easy', 'work', 'structured', 'data', 'python', 'panda', 'perform', 'wid
e', 'range', 'data', 'analysis', 'task', 'filtering', 'aggregation', 'transformatio
n', 'data', 'well', 'data', 'cleaning', 'preparation', 'definition', 'look', 'le',
'difference', 'execution', 'processing', 'architecture', 'let', 'go', 'major', 'diff
erence', 'two', 'key', 'difference', 'pyspark', 'panda', 'pyspark', 'library', 'work
ing', 'large', 'datasets', 'distributed', 'computing', 'environment', 'panda', 'libr
ary', 'working', 'smaller', 'tabular', 'datasets', 'single', 'machine', 'pyspark',
'built', 'top', 'apache', 'spark', 'framework', 'us', 'resilient', 'distributed', 'd
atasets', 'rdd', 'data', 'structure', 'panda', 'us', 'dataframe', 'data', 'structur
e', 'pyspark', 'designed', 'handle', 'data', 'processing', 'task', 'feasible', 'pand
a', 'due', 'memory', 'constraint', 'iterative', 'algorithm', 'machine', 'learning',
'large', 'datasets', 'pyspark', 'allows', 'parallel', 'processing', 'data', 'panda',
'pyspark', 'read', 'data', 'variety', 'source', 'including', 'hadoop', 'distribute
d', 'file', 'system', 'hdfs', 'amazon', 'local', 'file', 'system', 'panda', 'limite
d', 'reading', 'data', 'local', 'file', 'system', 'pyspark', 'integrated', 'big', 'd
ata', 'tool', 'like', 'hadoop', 'hive', 'panda', 'pyspark', 'written', 'scala', 'ru
n', 'java', 'virtual', 'machine', 'jvm', 'panda', 'written', 'python', 'pyspark', 's
teeper', 'learning', 'curve', 'panda', 'due', 'additional', 'concept', 'technology',
'involved', 'eg', 'distributed', 'computing', 'rdds', 'spark', 'sql', 'spark', 'stre
aming', 'etc', 'decide', 'library', 'use', '—', 'pyspark', 'v', 'panda', 'decision',
'whether', 'use', 'pyspark', 'panda', 'depends', 'size', 'complexity', 'dataset', 's
pecific', 'task', 'want', 'perform', 'size', 'dataset', 'pyspark', 'designed', 'hand
le', 'large', 'datasets', 'feasible', 'work', 'single', 'machine', 'using', 'panda',
'dataset', 'large', 'fit', 'memory', 'need', 'perform', 'iterative', 'distributed',
'computation', 'pyspark', 'better', 'choice', 'complexity', 'task', 'pyspark', 'powe
rful', 'tool', 'big', 'data', 'processing', 'allows', 'perform', 'wide', 'range', 'd
ata', 'processing', 'task', 'machine', 'learning', 'graph', 'processing', 'stream',
'processing', 'need', 'perform', 'task', 'pyspark', 'better', 'choice', 'learning',
'curve', 'pyspark', 'steeper', 'learning', 'curve', 'panda', 'requires', 'knowledg
e', 'distributed', 'computing', 'rdds', 'spark', 'sql', 'new', 'big', 'data', 'proce
ssing', 'want', 'get', 'started', 'quickly', 'panda', 'may', 'better', 'choice', 're
source', 'available', 'pyspark', 'requires', 'cluster', 'distributed', 'system', 'ru
n', 'need', 'access', 'appropriate', 'infrastructure', 'resource', 'access', 'resour
ce', 'panda', 'good', 'choice', 'summary', 'use', 'pyspark', 'large', 'datasets', 'c
omplex', 'task', 'feasible', 'panda', 'use', 'panda', 'small', 'datasets', 'simple',
'task', 'handled', 'single', 'machine']
```

Map each word

In [21]:
```
wordMap= words_wo_num.map(lambda word: (word, 1))
```

In [22]:
```
wrdMp = wordMap.collect()
print(len(wrdMp))
print(wrdMp)
```

```
341
[('definition', 1), ('pyspark', 1), ('pyspark', 1), ('python', 1), ('library', 1),
('spark', 1), ('programming', 1), ('allows', 1), ('use', 1), ('powerful', 1), ('effi
cient', 1), ('data', 1), ('processing', 1), ('capability', 1), ('apache', 1), ('spar
k', 1), ('within', 1), ('python', 1), ('programming', 1), ('language', 1), ('pyspar
k', 1), ('provides', 1), ('highlevel', 1), ('api', 1), ('distributed', 1), ('data',
1), ('processing', 1), ('used', 1), ('perform', 1), ('common', 1), ('data', 1), ('an
alysis', 1), ('task', 1), ('filtering', 1), ('aggregation', 1), ('transformation',
1), ('large', 1), ('datasets', 1), ('panda', 1), ('panda', 1), ('python', 1), ('libr
ary', 1), ('data', 1), ('manipulation', 1), ('analysis', 1), ('provides', 1), ('powe
rful', 1), ('data', 1), ('structure', 1), ('dataframe', 1), ('series', 1), ('designe
d', 1), ('make', 1), ('easy', 1), ('work', 1), ('structured', 1), ('data', 1), ('pyt
hon', 1), ('panda', 1), ('perform', 1), ('wide', 1), ('range', 1), ('data', 1), ('an
alysis', 1), ('task', 1), ('filtering', 1), ('aggregation', 1), ('transformation',
1), ('data', 1), ('well', 1), ('data', 1), ('cleaning', 1), ('preparation', 1), ('de
finition', 1), ('look', 1), ('le', 1), ('difference', 1), ('execution', 1), ('proces
sing', 1), ('architecture', 1), ('let', 1), ('go', 1), ('major', 1), ('difference',
1), ('two', 1), ('key', 1), ('difference', 1), ('pyspark', 1), ('panda', 1), ('pyspa
rk', 1), ('library', 1), ('working', 1), ('large', 1), ('datasets', 1), ('distribute
d', 1), ('computing', 1), ('environment', 1), ('panda', 1), ('library', 1), ('workin
g', 1), ('smaller', 1), ('tabular', 1), ('datasets', 1), ('single', 1), ('machine',
1), ('pyspark', 1), ('built', 1), ('top', 1), ('apache', 1), ('spark', 1), ('framewo
rk', 1), ('us', 1), ('resilient', 1), ('distributed', 1), ('datasets', 1), ('rdd',
1), ('data', 1), ('structure', 1), ('panda', 1), ('us', 1), ('dataframe', 1), ('dat
a', 1), ('structure', 1), ('pyspark', 1), ('designed', 1), ('handle', 1), ('data',
1), ('processing', 1), ('task', 1), ('feasible', 1), ('panda', 1), ('due', 1), ('mem
ory', 1), ('constraint', 1), ('iterative', 1), ('algorithm', 1), ('machine', 1), ('l
earning', 1), ('large', 1), ('datasets', 1), ('pyspark', 1), ('allows', 1), ('parall
el', 1), ('processing', 1), ('data', 1), ('panda', 1), ('pyspark', 1), ('read', 1),
('data', 1), ('variety', 1), ('source', 1), ('including', 1), ('hadoop', 1), ('distr
ibuted', 1), ('file', 1), ('system', 1), ('hdfs', 1), ('amazon', 1), ('local', 1),
('file', 1), ('system', 1), ('panda', 1), ('limited', 1), ('reading', 1), ('data',
1), ('local', 1), ('file', 1), ('system', 1), ('pyspark', 1), ('integrated', 1), ('b
ig', 1), ('data', 1), ('tool', 1), ('like', 1), ('hadoop', 1), ('hive', 1), ('pand
a', 1), ('pyspark', 1), ('written', 1), ('scala', 1), ('run', 1), ('java', 1), ('vir
tual', 1), ('machine', 1), ('jvm', 1), ('panda', 1), ('written', 1), ('python', 1),
('pyspark', 1), ('steeper', 1), ('learning', 1), ('curve', 1), ('panda', 1), ('due',
1), ('additional', 1), ('concept', 1), ('technology', 1), ('involved', 1), ('eg',
1), ('distributed', 1), ('computing', 1), ('rdds', 1), ('spark', 1), ('sql', 1), ('s
park', 1), ('streaming', 1), ('etc', 1), ('decide', 1), ('library', 1), ('use', 1),
('—', 1), ('pyspark', 1), ('v', 1), ('panda', 1), ('decision', 1), ('whether', 1),
('use', 1), ('pyspark', 1), ('panda', 1), ('depends', 1), ('size', 1), ('complexit
y', 1), ('dataset', 1), ('specific', 1), ('task', 1), ('want', 1), ('perform', 1),
('size', 1), ('dataset', 1), ('pyspark', 1), ('designed', 1), ('handle', 1), ('larg
e', 1), ('datasets', 1), ('feasible', 1), ('work', 1), ('single', 1), ('machine',
1), ('using', 1), ('panda', 1), ('dataset', 1), ('large', 1), ('fit', 1), ('memory',
1), ('need', 1), ('perform', 1), ('iterative', 1), ('distributed', 1), ('computatio
n', 1), ('pyspark', 1), ('better', 1), ('choice', 1), ('complexity', 1), ('task',
1), ('pyspark', 1), ('powerful', 1), ('tool', 1), ('big', 1), ('data', 1), ('process
ing', 1), ('allows', 1), ('perform', 1), ('wide', 1), ('range', 1), ('data', 1), ('p
rocessing', 1), ('task', 1), ('machine', 1), ('learning', 1), ('graph', 1), ('proces
sing', 1), ('stream', 1), ('processing', 1), ('need', 1), ('perform', 1), ('task',
1), ('pyspark', 1), ('better', 1), ('choice', 1), ('learning', 1), ('curve', 1), ('p
yspark', 1), ('steeper', 1), ('learning', 1), ('curve', 1), ('panda', 1), ('require
s', 1), ('knowledge', 1), ('distributed', 1), ('computing', 1), ('rdds', 1), ('spar
k', 1), ('sql', 1), ('new', 1), ('big', 1), ('data', 1), ('processing', 1), ('want',
```

```
1), ('get', 1), ('started', 1), ('quickly', 1), ('panda', 1), ('may', 1), ('better',
1), ('choice', 1), ('resource', 1), ('available', 1), ('pyspark', 1), ('requires',
1), ('cluster', 1), ('distributed', 1), ('system', 1), ('run', 1), ('need', 1), ('ac
cess', 1), ('appropriate', 1), ('infrastructure', 1), ('resource', 1), ('access',
1), ('resource', 1), ('panda', 1), ('good', 1), ('choice', 1), ('summary', 1), ('us
e', 1), ('pyspark', 1), ('large', 1), ('datasets', 1), ('complex', 1), ('task', 1),
('feasible', 1), ('panda', 1), ('use', 1), ('panda', 1), ('small', 1), ('datasets',
1), ('simple', 1), ('task', 1), ('handled', 1), ('single', 1), ('machine', 1)]
```

Reduce to count frequency

In [23]:
```
wordCount = wordMap.reduceByKey(lambda a,b: a+b)
```

In [24]:
```
wrdCount = wordCount.collect()
print(len(wrdCount))
print(wrdCount)
```

```
149
[('python', 5), ('library', 5), ('use', 5), ('powerful', 3), ('efficient', 1), ('cap
ability', 1), ('apache', 2), ('language', 1), ('api', 1), ('distributed', 8), ('use
d', 1), ('analysis', 3), ('task', 9), ('filtering', 2), ('transformation', 2), ('lar
ge', 6), ('datasets', 8), ('panda', 20), ('manipulation', 1), ('structure', 3), ('da
taframe', 2), ('easy', 1), ('work', 2), ('structured', 1), ('well', 1), ('architectu
re', 1), ('key', 1), ('working', 2), ('computing', 3), ('environment', 1), ('smalle
r', 1), ('single', 3), ('machine', 6), ('built', 1), ('framework', 1), ('us', 2),
('resilient', 1), ('rdd', 1), ('handle', 2), ('feasible', 3), ('memory', 2), ('itera
tive', 2), ('algorithm', 1), ('learning', 5), ('parallel', 1), ('read', 1), ('variet
y', 1), ('hadoop', 2), ('file', 3), ('reading', 1), ('big', 3), ('tool', 2), ('lik
e', 1), ('hive', 1), ('written', 2), ('java', 1), ('curve', 3), ('technology', 1),
('eg', 1), ('streaming', 1), ('etc', 1), ('—', 1), ('depends', 1), ('size', 2), ('da
taset', 3), ('using', 1), ('fit', 1), ('choice', 4), ('graph', 1), ('requires', 2),
('knowledge', 1), ('new', 1), ('started', 1), ('resource', 3), ('available', 1), ('a
ccess', 2), ('appropriate', 1), ('good', 1), ('summary', 1), ('small', 1), ('handle
d', 1), ('definition', 2), ('pyspark', 21), ('spark', 6), ('programming', 2), ('allo
ws', 3), ('data', 19), ('processing', 10), ('within', 1), ('provides', 2), ('highlev
el', 1), ('perform', 6), ('common', 1), ('aggregation', 2), ('series', 1), ('designe
d', 3), ('make', 1), ('wide', 2), ('range', 2), ('cleaning', 1), ('preparation', 1),
('look', 1), ('le', 1), ('difference', 3), ('execution', 1), ('let', 1), ('go', 1),
('major', 1), ('two', 1), ('tabular', 1), ('top', 1), ('due', 2), ('constraint', 1),
('source', 1), ('including', 1), ('system', 4), ('hdfs', 1), ('amazon', 1), ('loca
l', 2), ('limited', 1), ('integrated', 1), ('scala', 1), ('run', 2), ('virtual', 1),
('jvm', 1), ('steeper', 2), ('additional', 1), ('concept', 1), ('involved', 1), ('rd
ds', 2), ('sql', 2), ('decide', 1), ('v', 1), ('decision', 1), ('whether', 1), ('com
plexity', 2), ('specific', 1), ('want', 2), ('need', 3), ('computation', 1), ('bette
r', 3), ('stream', 1), ('get', 1), ('quickly', 1), ('may', 1), ('cluster', 1), ('inf
rastructure', 1), ('complex', 1), ('simple', 1)]
```

Display sorted Output

In [25]:
```
print(sorted(wrdCount))
```

```
[('access', 2), ('additional', 1), ('aggregation', 2), ('algorithm', 1), ('allows',
3), ('amazon', 1), ('analysis', 3), ('apache', 2), ('api', 1), ('appropriate', 1),
('architecture', 1), ('available', 1), ('better', 3), ('big', 3), ('built', 1), ('ca
pability', 1), ('choice', 4), ('cleaning', 1), ('cluster', 1), ('common', 1), ('comp
lex', 1), ('complexity', 2), ('computation', 1), ('computing', 3), ('concept', 1),
('constraint', 1), ('curve', 3), ('data', 19), ('dataframe', 2), ('dataset', 3), ('d
atasets', 8), ('decide', 1), ('decision', 1), ('definition', 2), ('depends', 1), ('d
esigned', 3), ('difference', 3), ('distributed', 8), ('due', 2), ('easy', 1), ('effi
cient', 1), ('eg', 1), ('environment', 1), ('etc', 1), ('execution', 1), ('feasibl
e', 3), ('file', 3), ('filtering', 2), ('fit', 1), ('framework', 1), ('get', 1), ('g
o', 1), ('good', 1), ('graph', 1), ('hadoop', 2), ('handle', 2), ('handled', 1), ('h
dfs', 1), ('highlevel', 1), ('hive', 1), ('including', 1), ('infrastructure', 1),
('integrated', 1), ('involved', 1), ('iterative', 2), ('java', 1), ('jvm', 1), ('ke
y', 1), ('knowledge', 1), ('language', 1), ('large', 6), ('le', 1), ('learning', 5),
('let', 1), ('library', 5), ('like', 1), ('limited', 1), ('local', 2), ('look', 1),
('machine', 6), ('major', 1), ('make', 1), ('manipulation', 1), ('may', 1), ('memor
y', 2), ('need', 3), ('new', 1), ('panda', 20), ('parallel', 1), ('perform', 6), ('p
owerful', 3), ('preparation', 1), ('processing', 10), ('programming', 2), ('provide
s', 2), ('pyspark', 21), ('python', 5), ('quickly', 1), ('range', 2), ('rdd', 1),
('rdds', 2), ('read', 1), ('reading', 1), ('requires', 2), ('resilient', 1), ('resou
rce', 3), ('run', 2), ('scala', 1), ('series', 1), ('simple', 1), ('single', 3), ('s
ize', 2), ('small', 1), ('smaller', 1), ('source', 1), ('spark', 6), ('specific',
1), ('sql', 2), ('started', 1), ('steeper', 2), ('stream', 1), ('streaming', 1), ('s
tructure', 3), ('structured', 1), ('summary', 1), ('system', 4), ('tabular', 1), ('t
ask', 9), ('technology', 1), ('tool', 2), ('top', 1), ('transformation', 2), ('two',
1), ('us', 2), ('use', 5), ('used', 1), ('using', 1), ('v', 1), ('variety', 1), ('vi
rtual', 1), ('want', 2), ('well', 1), ('whether', 1), ('wide', 2), ('within', 1),
('work', 2), ('working', 2), ('written', 2), ('—', 1)]
```

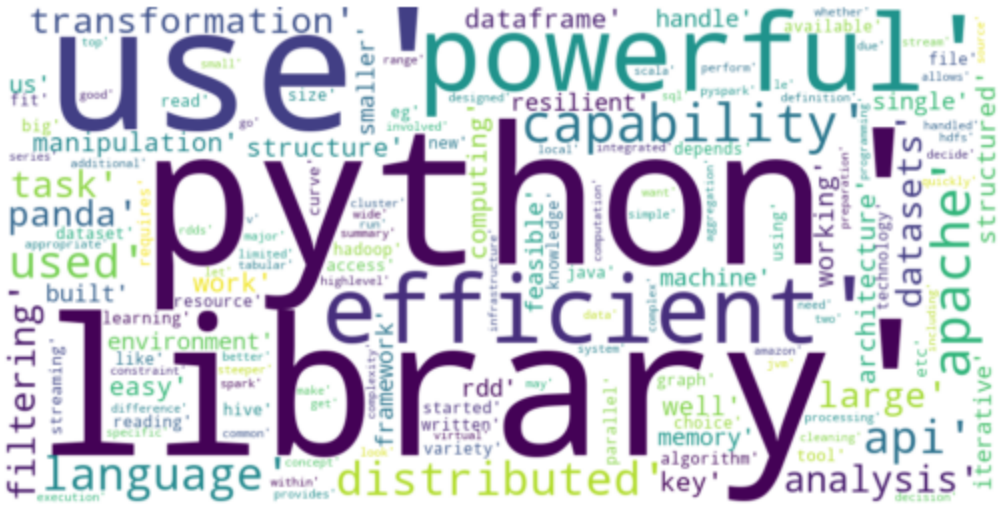
Generate Word Cloud

pip install wordcloud

In [26]:
```
import matplotlib.pyplot as plt
from wordcloud import WordCloud
```

In [27]:
```
wordcloud = WordCloud(background_color = 'white', width = 1000, height = 500).gener
```

In [28]:
```
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.rcParams['figure.figsize'] = [15, 8]
```



Create df

In [29]:
```
df = spark.createDataFrame(wrdCount, ['Word','Count'])
df.show(5)
```

```
+---------+-----+
|     Word|Count|
+---------+-----+
|   python|    5|
|  library|    5|
|      use|    5|
| powerful|    3|
|efficient|    1|
+---------+-----+
only showing top 5 rows
```

df.write.csv('D://DataSets//Output//wrdCount',header=True, mode="overwrite", sep=",")

#df.write.mode("overwrite").csv("file:///D://DataSets//Output//wrdCount")sc.stop() spark.stop()

In [ ]: