Pandas

pandas *is* a data analysis library that allows us to easily read *in and* work *with* different types of data so we can use this to analyze CSV files Excel files *and* other similar formats so *if* you're getting into the data science field then this library *is* going to be essential to learn it's one of the most downloaded packages *for* Python *and* that's for a great reason so not only does it allow us to easily read *in and* analyze data but it also has great performance since it built on top of numpy.

Jupiter notebooks allows us to actually see our data more easily by using the browser to print out our data and tables that make it year to visualize so I'm gonna use it in the series

Loading Data:

```
import pandas as pd

# data frame
df = pd.read_csv('/home/saabir/Desktop/pandas/data/survey_results_public.csv')

# when it reads this in it's going to read it in as a data frame so data frames
# frame is basically just rows and columns of data we can see what a data frame
# looks like but just by printing it out

df

  Respondent  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize
0   1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not loo
1   2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not emplo
2   3   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full
3   4   I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Employed full-time  United States
4   5   I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
88878   88377   NaN   Yes   Less than once a month but more than once per ...   The quality of OSS and closed source software ...   Not emp
88879   88601   NaN   No  Never   The quality of OSS and closed source software ...   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
88880   88802   NaN   No  Never   NaN   Employed full-time  NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   N
88881   88816   NaN   No  Never   OSS is, on average, of HIGHER quality than pro...   Independent contractor, freelancer, or self-em...   N
88882   88863   NaN   Yes   Less than once per year   OSS is, on average, of HIGHER quality than pro...   Not employed, and not looking for

88883 rows × 85 columns

# The df.shape attribute in Python typically refers to the shape or dimensions of a DataFrame
df.shape

(88883, 85)

# The df.info() method provides information about the number of rows and columns in the DataFrame,
# as well as the data types of all the columns.
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 88883 entries, 0 to 88882
Data columns (total 85 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Respondent             88883 non-null  int64
 1   MainBranch             88331 non-null  object
 2   Hobbyist               88883 non-null  object
 3   OpenSourcer            88883 non-null  object
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
 82  Dependents             83059 non-null  object
 83  SurveyLength           86984 non-null  object
 84  SurveyEase             87081 non-null  object
dtypes: float64(5), int64(1), object(79)
memory usage: 57.6+ MB

# we have 88883 enteries (rows) 85 columns , most parts Objects mean string

pd.set_option('display.max_columns',85) # set the maximum columns to see (displayed ) one into 85
# remeber to go the first pd df variable you run case if you run under here you won't see any data immideitly
#  the survey results schema CSV file that was included in our download gives the matching questions for all of these
# column names here so if we wanted to see what these column names here mean for this data then we can load schema CSV file
# (here) means the titles of the  data survey like => Respondent  MainBranch  Hobbyist  OpenSourcer OpenSour

df_schema = pd.read_csv('/home/saabir/Desktop/pandas/data/survey_results_schema.csv')

df_schema

  Column  QuestionText
0   Respondent  Randomized respondent ID number (not in order ...
1   MainBranch  Which of the following options best describes ...
2   Hobbyist  Do you code as a hobby?
3   OpenSourcer   How often do you contribute to open source?
4   OpenSource  How do you feel about the quality of open sour...
...   ...   ...
80  Sexuality   Which of the following do you currently identi...
81  Ethnicity   Which of the following do you identify as? Ple...
82  Dependents  Do you have any dependents (e.g., children, el...
83  SurveyLength  How do you feel about the length of the survey...
84  SurveyEase  How easy or difficult was this survey to compl...

85 rows × 2 columns

# let's set this up so that we can view 85 rows and then reprint
pd.set_option('display.max_row',85)

df_schema

  Column  QuestionText
0   Respondent  Randomized respondent ID number (not in order ...
1   MainBranch  Which of the following options best describes ...
2   Hobbyist  Do you code as a hobby?
3   OpenSourcer   How often do you contribute to open source?
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
80  Sexuality   Which of the following do you currently identi...
81  Ethnicity   Which of the following do you identify as? Ple...
82  Dependents  Do you have any dependents (e.g., children, el...
83  SurveyLength  How do you feel about the length of the survey...
84  SurveyEase  How easy or difficult was this survey to compl...

 # to display the first few rows of a DataFrame use .head() method By default, it returns the first 5 rows of the DataFrame
# but If you want to display a different number of rows, you can pass the desired number as an argument to df.head(n)
df.head()

  Respondent  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize
0   1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not loo
1   2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not emplo
2   3   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full
3   4   I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Employed full-time  United States
4   5   I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full

df.head(10)

  Respondent  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize
0   1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not loo
1   2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not emplo
2   3   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full
3   4   I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Employed full-time  United States
4   5   I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full
5   6   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full
6   7   I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Independent contractor, freelancer,
7   8   I code primarily as a hobby   Yes   Less than once per year   OSS is, on average, of HIGHER quality than pro...   Not employed, but
8   9   I am a developer by profession  Yes   Once a month or more often  The quality of OSS and closed source software ...   Employed full
9   10  I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full

# to display the last few rows of a DataFrame use .tail() By default, it returns the last 5 rows of the DataFrame,
# but you can specify a different number of rows to display .tail(number)
df.tail()

  Respondent  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize
88878   88377   NaN   Yes   Less than once a month but more than once per ...   The quality of OSS and closed source software ...   Not emp
88879   88601   NaN   No  Never   The quality of OSS and closed source software ...   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
88880   88802   NaN   No  Never   NaN   Employed full-time  NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   N
88881   88816   NaN   No  Never   OSS is, on average, of HIGHER quality than pro...   Independent contractor, freelancer, or self-em...   N
88882   88863   NaN   Yes   Less than once per year   OSS is, on average, of HIGHER quality than pro...   Not employed, and not looking for
```

we're going to be learning about the data frame and series data types so like I said in the last video these are basically the backbone of pandas and are the two primary data types that you'll likely be using the most, we're gonna go over how we can think of data frames and series data types in a different way and then we'll look at the basics of getting information from these data types

data frame is made up of multiple rows here and we also have multiple columns so in the case of this data these are survey results but your data can be you know whatever your data is but it's most likely going to be in rows and columns kind of like table

Data Frame and Series Basics

```
# explain the table rows & columns take example like Hobbyist (and see what that question was ) you can also look Employment

# data frame is just rows and columns but now let me explain how I like to think of data frames using native Python
# so if we were only using Python and not using pandas to store information in rows and columns then how would we do this
# well for those of you familiar with dictionaries you might think that it's a good idea to store information

# dictionaries  we have keys and values so if I'm representing some data for a person we represent like this
person = {
    "first": "Corey",
    "last": "Schafer",
    "email": "CoreyMSchafer@gmail.com"
}

# this dictionary was represents data for a single person but how would we represent data for multiple people
# so to represent this we make all of our values and our dictionaries in a list
people = {
    "first": ["Corey"],
    "last": ["Schafer"],
    "email": ["CoreyMSchafer@gmail.com"]
}

#  now instead of just a single string here for the values I instead have a list and our list of person
# (now we have single person but since this is a list we can add more first names and information in here  )

people = {
    "first": ["Corey", 'Jane', 'John'],
    "last": ["Schafer", 'Doe', 'Doe'],
    "email": ["CoreyMSchafer@gmail.com", 'JaneDoe@email.com', 'JohnDoe@email.com']
}
# the keys will be our columns firt , last , email
# first value will be the first person ,  second vaue will be the second person # third value will be third person etc..

# if you look up the definition of a panda's data frame online then you'll see a lot of definitions that just say
# something like it's a two dimensional data structure now that might sound a little confusing but in layman's terms
# that basically just means rows and columns

# the key for email here would be our email column and contain all of the emails values
# if we wanted to see the email column then we can just access that key
people['email']

['CoreyMSchafer@gmail.com', 'JaneDoe@email.com', 'JohnDoe@email.com']

# we can actually create a dataframe from this dictionary and see what this looks like so let's do that and
# look at some basic data frame functionality
import pandas as pd

df = pd.DataFrame(people)

df

  first   last  email
0   Corey   Schafer   CoreyMSchafer@gmail.com
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

# we have these over here to the far left that don't have column names this 0 1 & 2 now this is an index now I'm not
# going to go too much into indexes right now because that's what the next video is going to cover but basically
# it's a unique value for our rows now it doesn't need to be unique but again we'll talk more about later

# let's just access the values of a single column so just like we did with the dictionary we can access a single
# column just like we were accessing the key of a dictionary
df['email']

0    CoreyMSchafer@gmail.com
1          JaneDoe@email.com
2          JohnDoe@email.com
Name: email, dtype: object

# I do want to emphasize that I only use the pure Python example so that we could get an idea of how to think about a
# data frame but like I said a data frame is much much more than just a dictionary of Lists so for example we can see
# that when we displayed the email column here it doesn't look the same as when we displayed the list of values from
# that dictionary and that's because this is actually returning a series and we can see this if we check the type
# so if I check the type of this email column

type(df['email'])  # this is a series of object

pandas.core.series.Series

# what is a series ?
# series is  basically a list of data if you look up the definition of a series online then you'll see a lot of definitions
# says it's a one-dimensional array  but in layman's terms that basically just means that it's rows of data

# you can think of a data frame as being rows and columns and series being rows of a single column
# other way you can say data frame is basically a container for multiple of these series objects

# one more time so we can see that data frame here is two-dimensional because it has rows and columns so we can see here
# that it has you know first name, last name, and email,  now whenever we access just the email then we can see that we get
# all these emails here now this is a series and I said that a data frame basically contains is a container
# for multiple series objects so we can think of this email column here as a series this last column here is a series
# and this first column as a series and also we can see where we printed out this series here for the emails we can
# see that this series also has an index as well just like our data frame did so this index is over here on the left the
# 0 1 - okay so we can access a single column of a data frame like we're accessing in here df['email']

# but you might also see some people use dot notation to do the same thing
df.email

0    CoreyMSchafer@gmail.com
1          JaneDoe@email.com
2          JohnDoe@email.com
Name: email, dtype: object

# compare b/w the two ways of accessing
df['email']

 # the way that you want to do this is really just a personal preference

0    CoreyMSchafer@gmail.com
1          JaneDoe@email.com
2          JohnDoe@email.com
Name: email, dtype: object

# access multiple columns
# we can use the bracket notation and pass in a list of the columns that we want

df[['first', 'email']]

  first   email
0   Corey   CoreyMSchafer@gmail.com
1   Jane  JaneDoe@email.com
2   John  JohnDoe@email.com

# if you have a lot of columns and want to see all of them easily then we can use .columns

df.columns

Index(['first', 'last', 'email'], dtype='object')

# in order to get rows ( single person data line ) we can use the loc and iLoc indexers

# first let's take a look at iloc allows us to access rows by integer location hence the name iloc integer location
df.iloc[0]
# compare the actaul data frame to see how it grapped the first row

first                      Corey
last                     Schafer
email    CoreyMSchafer@gmail.com
Name: 0, dtype: object

df

  first   last  email
```

```
0   Corey   Schafer   CoreyMSchafer@gmail.com
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

# we can select multiple rows as well by passing in a list of integers

df.iloc[[0,1]]

  first   last  email
0   Corey   Schafer   CoreyMSchafer@gmail.com
1   Jane  Doe   JaneDoe@email.com

# with these iloc and loc we can also select the row and then special columns as the second argument

# iloc[['first arguments rows 0,1,2'], ['the second arguments is the columns you want first last, email']]

# so remember our first name is the first column ( which will be index 0 )  the  last name is the second column (index 1)
# and the email is the third column (index 2)

df.iloc[[0,1], [2]]  # select the first two rows and then select columns email but using their index

  email
0   CoreyMSchafer@gmail.com
1   JaneDoe@email.com

 #loc we're going to be searching by label
# real quick let's look at our entire data frame
df

  first   last  email
0   Corey   Schafer   CoreyMSchafer@gmail.com
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

# when we're talking about labels that rows has is  these  indexes and now we don't have custom indexes (labels) right now
# so this index is just a default range of integers so at the moment this will somewhat be similar witt iloc indexers
# but we'll look  use cases loc with actual labels in the next chapter the

# so on the left you see 0,1,2 are the indexers and that are the labels of that row, loc uses that labels to get the values

df.loc[0]

first                      Corey
last                     Schafer
email    CoreyMSchafer@gmail.com
Name: 0, dtype: object

#  same as iloc we can access multipe rows in loc by passing list of rows we want using indexers (labels in the next )

df.loc[[1,2]] # it select the first two rows (persons lines ) so don't think of columns

  first   last  email
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

# with te same logic loc we can also select the row and then special columns as the second argument just like we did
# in iloc but this time we not using iloc (integer loc) so we can't use integers instead we gonna use string

df.loc[[0,1],['first', 'email']]

  first   email
0   Corey   CoreyMSchafer@gmail.com
1   Jane  JaneDoe@email.com

# let's go back to our gaint data and mess around little bit
import pandas as pd

df = pd.read_csv('/home/saabir/Desktop/pandas/data/survey_results_public.csv')
df_schema = pd.read_csv('/home/saabir/Desktop/pandas/data/survey_results_schema.csv')
pd.set_option('display.max_columns', 85)
pd.set_option('display.max_rows', 85)

# let's see how many rows and columns that we have in this data frame
df.shape

(88883, 85)

df

  Respondent  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize
0   1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not loo
```

```
1   2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not emplo
2   3   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full
3   4   I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Employed full-time  United States
4   5   I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
88878   88377   NaN   Yes   Less than once a month but more than once per ...   The quality of OSS and closed source software ...   Not emp
88879   88601   NaN   No  Never   The quality of OSS and closed source software ...   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
88880   88802   NaN   No  Never   NaN   Employed full-time  NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   N
88881   88816   NaN   No  Never   OSS is, on average, of HIGHER quality than pro...   Independent contractor, freelancer, or self-em...   N
88882   88863   NaN   Yes   Less than once per year   OSS is, on average, of HIGHER quality than pro...   Not employed, and not looking for

88883 rows × 85 columns

df.columns

Index(['Respondent', 'MainBranch', 'Hobbyist', 'OpenSourcer', 'OpenSource',
       'Employment', 'Country', 'Student', 'EdLevel', 'UndergradMajor',
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
       'SOComm', 'WelcomeChange', 'SONewContent', 'Age', 'Gender', 'Trans',
       'Sexuality', 'Ethnicity', 'Dependents', 'SurveyLength', 'SurveyEase'],
      dtype='object')

# lets grap all the answers for hoppies column
df['Hobbyist']

0        Yes
1         No
2        Yes
3         No
4        Yes
        ...
88878    Yes
88879     No
88880     No
88881     No
88882    Yes
Name: Hobbyist, Length: 88883, dtype: object

# real quick let me show you something that will cover more of further into the series but I want to give you an idea
# of how powerful something like pandas is so let's say that we wanted to know how many of these responses were answered
# yes and how many were answered no

df['Hobbyist'].value_counts()

Yes    71257
No     17626
Name: Hobbyist, dtype: int64

df['Age'].value_counts()

25.0    4669
24.0    4428
27.0    4418
26.0    4397
28.0    4387
        ...
24.8       1
90.0       1
61.3       1
4.0        1
87.0       1
Name: Age, Length: 126, dtype: int64

df['Age'].value_counts().head(10)

25.0    4669
24.0    4428
27.0    4418
26.0    4397
28.0    4387
23.0    4109
30.0    4050
29.0    3991
22.0    3358
31.0    3063
Name: Age, dtype: int64

# we'll cover more of this and future videos when we learn more about analyzing data in depth but I wanted to give you a
# quick taste as to why it's beneficial to even learn pandaslike we're doing here it makes this type of stuff really easy

# so now let's grab a specific row and a specific column so let's grab the first row and we'll also grab that same
# hobbyist column for that row
df.loc[0]

Respondent                                                        1
MainBranch                           I am a student who is learning to code
Hobbyist                                                        Yes
OpenSourcer                                                   Never
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
Dependents                                                       No
SurveyLength                                      Appropriate in length
SurveyEase                                    Neither easy nor difficult
Name: 0, dtype: object

# now let's grab the specific column in this row
df.loc[0,"Hobbyist"]

'Yes'

# and if we use iloc it will be the same but litle bit diffrent
df.iloc[0,2]

'Yes'

 # get the first three responses for the hobbiest column
df.loc[[0,1,2],'Hobbyist']

0    Yes
1     No
2    Yes
Name: Hobbyist, dtype: object

 # we can also use slicing to grab multiple rows and columns as well
df.loc[0:2, 'Hobbyist']

0    Yes
1     No
2    Yes
Name: Hobbyist, dtype: object

# we can do this with the columns as well so right now we're only getting two hobbiest column but let's go back and
# look at our columns and see what columns come afterthe hobbiest column
df.columns

Index(['Respondent', 'MainBranch', 'Hobbyist', 'OpenSourcer', 'OpenSource',
       'Employment', 'Country', 'Student', 'EdLevel', 'UndergradMajor',
       'EduOther', 'OrgSize', 'DevType', 'YearsCode', 'Age1stCode',
       'YearsCodePro', 'CareerSat', 'JobSat', 'MgrIdiot', 'MgrMoney',
       ...  ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
       'SOHowMuchTime', 'SOAccount', 'SOPartFreq', 'SOJobs', 'EntTeams',
       'SOComm', 'WelcomeChange', 'SONewContent', 'Age', 'Gender', 'Trans',
       'Sexuality', 'Ethnicity', 'Dependents', 'SurveyLength', 'SurveyEase'],
      dtype='object')

df.loc[0:5, 'Hobbyist': 'Employment']
# the reason that slicing is inclusive forthese values because imagine how much of a pain it would be if we wanted all of
# the columns from hobbyist to employment but the last value here wasn't inclusive and we had to come up here and say well
# if I want from hobbyists  to employment then I really need to pass in you know hobbyist to country and country's not
# inclusive that would just be way too confusing so it's so much easier for this to be inclusive here

  Hobbyist  OpenSourcer   OpenSource  Employment
0   Yes   Never   The quality of OSS and closed source software ...   Not employed, and not looking for work
1   No  Less than once per year   The quality of OSS and closed source software ...   Not employed, but looking for work
2   Yes   Never   The quality of OSS and closed source software ...   Employed full-time
3   No  Never   The quality of OSS and closed source software ...   Employed full-time
4   Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full-time
5   Yes   Never   The quality of OSS and closed source software ...   Employed full-time

# lets see what question ws in these columns we displayed
df_schema

  Column  QuestionText
0   Respondent  Randomized respondent ID number (not in order ...
1   MainBranch  Which of the following options best describes ...
2   Hobbyist  Do you code as a hobby?
3   OpenSourcer   How often do you contribute to open source?
4   OpenSource  How do you feel about the quality of open sour...
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
80  Sexuality   Which of the following do you currently identi...
81  Ethnicity   Which of the following do you identify as? Ple...
82  Dependents  Do you have any dependents (e.g., children, el...
83  SurveyLength  How do you feel about the length of the survey...
```

```
84  SurveyEase  How easy or difficult was this survey to compl...

df_schema.loc[2:5]

  Column  QuestionText
2   Hobbyist  Do you code as a hobby?
3   OpenSourcer   How often do you contribute to open source?
4   OpenSource  How do you feel about the quality of open sour...
5   Employment  Which of the following best describes your cur...
```

How to set , Reset Indexes

```
# in this video we'll learn how to set custom indexes and the benefits of doing this now

# we can look at indexes using a simple data frame with a little bit of data and then we'll see how to use these with our
# larger survey data set that we've been using so far in the series

people = {
    "first": ["Corey", 'Jane', 'John'],
    "last": ["Schafer", 'Doe', 'Doe'],
    "email": ["CoreyMSchafer@gmail.com", 'JaneDoe@email.com', 'JohnDoe@email.com']
}

import pandas as pd

df = pd.DataFrame(people)

df

  first   last  email
0   Corey   Schafer   CoreyMSchafer@gmail.com
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

 # our data frames have this thing on the far left over here that looks like a column without a name and this is an index

# sometimes you may need to have a different identifier for each row and that will basically be the label for that row so
# it's usually unique now  but pandas it doesn't actually enforce indexes being unique and sometimes it won't be but most
# of the time these  will be unique values

# so what might be a better index for our sample data here well maybe the email address would be a good index for this data
#  since that's usually a unique value for most people but first let's see our email columns
df['email']

0    CoreyMSchafer@gmail.com
1          JaneDoe@email.com
2          JohnDoe@email.com
Name: email, dtype: object

# what if we wanted to set these email addresses as the index for this data frame
df.set_index('email')

  first   last
email
CoreyMSchafer@gmail.com   Corey   Schafer
JaneDoe@email.com   Jane  Doe
JohnDoe@email.com   John  Doe

# now we can see that the email is on the far left and it's bold
# but if I look at my data frame again
df

  first   last  email
0   Corey   Schafer   CoreyMSchafer@gmail.com
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

# we can see that our data frame didn't actually change it still has the default index over here on the left and that's
# because pandas doesn't do a lot of these changes in place unless we specifically tell it to do so and this is actually
# nice because it allows us to  experiment without worrying about modifying our data frame in unexpected ways

# let's say that we actually did want to set our index to the email column and have those changes
df.set_index('email', inplace=True)
```

```
df
# rerun the data frame again then now we can see that it actually dead did set that index and modified that data frame

  first   last
email
CoreyMSchafer@gmail.com   Corey   Schafer
JaneDoe@email.com   Jane  Doe
JohnDoe@email.com   John  Doe

 # we can actually look specifically at that index
df.index
# why would this actually be useful ? the email address as the index gives us a nice unique identifier for our rows and we
# no longer need need to access our row using interer index

Index(['CoreyMSchafer@gmail.com', 'JaneDoe@email.com', 'JohnDoe@email.com'], dtype='object', name='email')

# we used loc to search our data frame rows by using labels, these indexes were the labels for the rows , before we were
# using the default ranged index but now we can find a specific row by passing loc into one of that labels of emails we
# just set to be our indexers
df.loc['CoreyMSchafer@gmail.com']

first      Corey
last     Schafer
Name: CoreyMSchafer@gmail.com, dtype: object

# we can still pass in values for the specific columns as well so if we wanted the last name
df.loc['JaneDoe@email.com', 'last']

'Doe'

# now we actually no longer have those default integers as our index because now it's using the email so if I try to use
# those integers  that we use before so if I say if I want:
# df.loc[0]
# then we're going to get a type error and I get an error because it no longer has an index with that label if you want to
# use integer  location instead of labels then you still have the iloc index that is available to you
df['iloc']

# if you accidentally set the index and want to reset it then we can do that with the reset index method
df.reset_index()
# you have to add in-place equal to true so that those changes carry over
df.reset_index(inplace=True)

df

  email   first   last
0   CoreyMSchafer@gmail.com   Corey   Schafer
1   JaneDoe@email.com   Jane  Doe
2   JohnDoe@email.com   John  Doe

# switch over to our other notebookhere with the stack overflow data that we've been using so far throughout the series
# and we'll take a look at some real-worldexamples of why using indexes is useful

import pandas as pd

# df = pd.read_csv('data/survey_results_public.csv', index_col='Respondent')
df = pd.read_csv('data/survey_results_public.csv', )

pd.set_option('display.max_columns', 85)
pd.set_option('display.max_rows', 85)

df.head()

  Respondent  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize
0   1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not loo
1   2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not emplo
2   3   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full
3   4   I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Employed full-time  United States
4   5   I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full

# if we look at this respondent column here this respondent column is actually a unique ID so its respondent one respondent
# two and three and so on so really we should probably clean this up a bit and just use that respondent ID as our data
#  frame index

df.set_index('Respondent', inplace=True)

# let's see how our data index changed
df.head()

  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize   DevType   Y
Respondent
1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not looking
```

```
2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not employed,
3   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full-tim
4   I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Employed full-time  United States   No
5   I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full-tim

# we can set index in that way or we can do this while we're actually reading in the data by passing in an additional
# argument to the read CSV method

# but before doing that let's reset what we did before
df.reset_index(inplace=True)

# and see the chnages we did
df.head(3)

  Respondent  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize
0   1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not loo
1   2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not emplo
2   3   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full

 # let's just add another argument to the read csv method that set's automatically the column index
df = pd.read_csv('data/survey_results_public.csv', index_col='Respondent')

df.head(3)

  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize   DevType   Y
Respondent
1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not looking
2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not employed,
3   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full-tim

# if you wanted the all answers of first respondent
# (remember this time first respondent has indexer , you changed defualt index and set the respondent as index and they
#  start from 1)
df.loc[1]

MainBranch                           I am a student who is learning to code
Hobbyist                                                                Yes
OpenSourcer                                                           Never
OpenSource                 The quality of OSS and closed source software ...
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
Sexuality                                              Straight / Heterosexual
Ethnicity                                                               NaN
Dependents                                                               No
SurveyLength                                           Appropriate in length
SurveyEase                                        Neither easy nor difficult
Name: 1, dtype: object

# so if we use real word example

# suppose if wanted to know what hobbiest  meant then we can look at our schema data frame
df_schema = pd.read_csv('/home/saabir/Desktop/pandas/data/survey_results_schema.csv')

df_schema

  Column  QuestionText
0   Respondent  Randomized respondent ID number (not in order ...
1   MainBranch  Which of the following options best describes ...
2   Hobbyist  Do you code as a hobby?
3   OpenSourcer   How often do you contribute to open source?
4   OpenSource  How do you feel about the quality of open sour...
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
80  Sexuality   Which of the following do you currently identi...
81  Ethnicity   Which of the following do you identify as? Ple...
82  Dependents  Do you have any dependents (e.g., children, el...
83  SurveyLength  How do you feel about the length of the survey...
84  SurveyEase  How easy or difficult was this survey to compl...

# here and we can see that the answer that they are the answer on the survey or the question on the survey for hobbiest was :
#  do you code as a hobby so we can see why the answers is yes or no but what if I wanted to see what a specific
# column(question) meant without  needing to search through this entire data frame manually well in this case we can
# simply set the column name as the index

df_schema.set_index('Column', inplace=True)

# check if the changes we implemented and column become the index of our schema data
df_schema.head(4) # it did and it's bold as you can see

  QuestionText
Column
Respondent  Randomized respondent ID number (not in order ...
MainBranch  Which of the following options best describes ...
```

```
Hobbyist  Do you code as a hobby?
OpenSourcer   How often do you contribute to open source?

# so now if i wanted to see again what hoppiest was meant in the surver i just can use loc to find the row of that
df_schema.loc['Hobbyist']

QuestionText    Do you code as a hobby?
Name: Hobbyist, dtype: object

# let's go back to our survey data here and see if we can find a column that doesn't make much sense to us
df_schema.loc['MgrIdiot']

QuestionText    How confident are you that your manager knows ...
Name: MgrIdiot, dtype: object

# the text is actually truncated in Jupiter notebooks by default but we can we can change that setting if we'd like to see
#  this entire question text but I kind of want it on since we have so much data to display but instead if you want to see
#  the full text for that question then you can just access the data in that row and column directly by also passing in the column name into
df_schema.loc['MgrIdiot', 'QuestionText'] # now we can see that the full question text for what MGR idiot means

'How confident are you that your manager knows what they’re doing?'

# this is one nice example here of when setting these indexes is useful because it allows us to Search these rows by labels very easily

# let me show you one more thing before we finish up here , so it might make it a bit easier to read this schema data frame
#  if the indexes were sorted alphabetically

df_schema.sort_index()

  QuestionText
Column
Age   What is your age (in years)? If you prefer not...
Age1stCode  At what age did you write your first line of c...
BetterLife  Do you think people born today will have a bet...
BlockchainIs  Blockchain / cryptocurrency technology is prim...
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
WorkRemote  How often do you work remotely?
WorkWeekHrs   On average, how many hours per week do you work?
YearsCode   Including any education, how many years have y...
YearsCodePro  How many years have you coded professionally (...

# we can see that now these indexes are sorted alphabetically so if we knew that we wanted to know get to employment or
# something it's going to be in the ease

# if you wanted to sort this in descending order instead then we could just say ascending is equal to false

df_schema.sort_index(ascending=False)

  QuestionText
Column
YearsCodePro  How many years have you coded professionally (...
YearsCode   Including any education, how many years have y...
WorkWeekHrs   On average, how many hours per week do you work?
WorkRemote  How often do you work remotely?
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
BlockchainOrg   How is your organization thinking about or imp...
BlockchainIs  Blockchain / cryptocurrency technology is prim...
BetterLife  Do you think people born today will have a bet...
Age1stCode  At what age did you write your first line of c...
Age   What is your age (in years)? If you prefer not...

# but as told you before pandas doesn't change the settings unless you specify to change it , the ascending and descending
# order we did was allowing us to you know see what things would look like without actually affecting the data frame itself
df_schema

  QuestionText
Column
Respondent  Randomized respondent ID number (not in order ...
MainBranch  Which of the following options best describes ...
Hobbyist  Do you code as a hobby?
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
Dependents  Do you have any dependents (e.g., children, el...
SurveyLength  How do you feel about the length of the survey...
SurveyEase  How easy or difficult was this survey to compl...

#  so if you wanted to the sort to be permanent and procceed the changes you can add inplace to be true to make the changes
df_schema.sort_index(inplace=True)

df_schema

  QuestionText
```

```
Column
Age   What is your age (in years)? If you prefer not...
Age1stCode  At what age did you write your first line of c...
BetterLife  Do you think people born today will have a bet...
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
YearsCode   Including any education, how many years have y...
YearsCodePro  How many years have you coded professionally (...
```

Filtering - Using Conditionals to Filter Rows and Columns

```
people = {
    "first": ["Corey", 'Jane', 'John'],
    "last": ["Schafer", 'Doe', 'Doe'],
    "email": ["CoreyMSchafer@gmail.com", 'JaneDoe@email.com', 'JohnDoe@email.com']
}

import pandas as pd
df = pd.DataFrame(people)

# filtering is one of the main things to learn with pandas because it's basically how we begin every project by filtering
# the data that we want from the data that we don't now

# let's say that I wanted everyone that has the last name of DOE let's do comparision

df['last'] == 'doe'

0    False
1    False
2    False
Name: last, dtype: bool

# we get back is a series object and this might not be what you expected so maybe you thought we would just get a data
# frame back with all of the values that met our criteria but what we got back is a series with a bunch of true false
# values now these true false values actually correspond to our original data frame and the true values are the rows
#  that met our filter criteria and the false values are the rows that didn't meet our filter criteria

# so we can see here up here in our simple data frame the last name of Schaefer is false and these two last names here with
#  Doe are true

# so now let's apply this filter to our data frame so first I'm going to assign this this return a series here to a variable
#  and I'm just  going to call this variable filt so I'll say filt is equal to and then this comparison here now filter is
# a built-in Python keyword so be sure to use something else anytime

filt = df['last'] == 'Doe'

filt

0    False
1     True
2     True
Name: last, dtype: bool

filt

0    False
1     True
2     True
Name: last, dtype: bool

df[filt]

# Pandas takes the DataFrame df and filters it based on the filt condition. It keeps the rows where the condition is True
# and  removes the rows where it's False.

  first   last  email
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

# we get a data frame back where it returned all of the rows that have the last name of doh now we only assigned the filter
# on a different line because I think that's easier to read but you might see some people put these comparisons directly
# in the brackets for the data frame

df[ df['last'] == 'Doe' ]
```

```
  first   last  email
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

# another way that you can do this and the way that I prefer to do it is to use the .loc indexer that we've seen a few times

df.loc[filt]

  first   last  email
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

# this is one reason why pandas can be a bit confusing to people because there are multiple things that you can pass into
# these different brackets to get different results

# the reason that I like using dot loc for this is because we can still grab these specific columns that we want as well

df.loc[filt, ['email', 'first']]

  email   first
1   JaneDoe@email.com   Jane
2   JohnDoe@email.com   John

# first let's go over the [and] and [or] operators now we can't use the Python built-in  and + or keywords for our filters
# so we're going to be using some other symbols [and => &]  and [or => | ]

filt = (df['first'] == 'John') & (df['last'] == 'Doe')

df.loc[filt]

  first   last  email
2   John  Doe   JohnDoe@email.com

df

  first   last  email
0   Corey   Schafer   CoreyMSchafer@gmail.com
1   Jane  Doe   JaneDoe@email.com
2   John  Doe   JohnDoe@email.com

# let's say that I wanted to get the complete opposite of this filter where we want all of the rows where the last name
# isn't Schaefer and the first name isn't John

# negate that filter and give me the opposite of those results
df.loc[~filt]

  first   last  email
0   Corey   Schafer   CoreyMSchafer@gmail.com
1   Jane  Doe   JaneDoe@email.com



# now Let's go Back to our big data and see what we can do in filtering

import pandas as pd

df = pd.read_csv('data/survey_results_public.csv', index_col='Respondent')
df_schema = pd.read_csv('data/survey_results_schema.csv', index_col='Column')

pd.set_option('display.max_columns', 85)
pd.set_option('display.max_rows', 85)

df.head()

  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize   DevType   Y
Respondent
1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not looking
2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not employed,
3   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full-tim
4   I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Employed full-time  United States   No
5   I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full-tim

# let's say that we wanted to look at the data for people who are making a salary over a certain amount now maybe we want to want to take
# a look at what languages are earning the higher salaries or something like that so in order to do that I'm going to first create a filter
# now if you don't know which column and the data frame it gives the salary then you can always find that using the schema data frame

# if you don't know which column and the data frame it gives the salary then you can always find that using the schema data
#  frame that we've seen throughout the series that tells us with what each of these columns here  means but for the sake
# of time here I'll  just tell you that the column for salary
df_schema
```

```
  QuestionText
Column
Respondent  Randomized respondent ID number (not in order ...
MainBranch  Which of the following options best describes ...
Hobbyist  Do you code as a hobby?
OpenSourcer   How often do you contribute to open source?
OpenSource  How do you feel about the quality of open sour...
---------------------------------------------------------------------------
Sexuality   Which of the following do you currently identi...
Ethnicity   Which of the following do you identify as? Ple...
Dependents  Do you have any dependents (e.g., children, el...
SurveyLength  How do you feel about the length of the survey...
SurveyEase  How easy or difficult was this survey to compl...

# let's check if it's true and see the full text of ConvertedComp column
df_schema.loc['ConvertedComp', 'QuestionText']

'Salary converted to annual USD salaries using the exchange rate on 2019-02-01,assuming 12 working months & 50 working weeks.'

# so if we want those who take salar  over let's say a high salary is over 100,000
filt = (df['ConvertedComp'] > 100000)

filt

Respondent
1        False
2        False
3        False
4        False
5        False
         ...
88377    False
88601    False
88802    False
88816    False
88863    False
Name: ConvertedComp, Length: 88883, dtype: bool

df.loc[filt]
# if i scroll over converted comp column then all of these salaries here should be over 100,000 and it looks like they are

  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize   DevType   Y
Respondent
6   I am not primarily a developer, but I write co...   Yes   Never   The quality of OSS and closed source software ...   Employed full-tim
16  I am a developer by profession  Yes   Never   The quality of OSS and closed source software ...   Employed full-time  United Kingdom  N
22  I am a developer by profession  Yes   Less than once per year   OSS is, on average, of HIGHER quality than pro...   Employed full-time
26  I am a developer by profession  Yes   Less than once per year   The quality of OSS and closed source software ...   Employed full-time
32  I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Employed full-time  United States   No
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...
88874   I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of LOWER quality than prop...   Employed full
88876   I am a developer by profession  Yes   Never   OSS is, on average, of HIGHER quality than pro...   Employed full-time  United States
88877   I am a developer by profession  Yes   Less than once per year   OSS is, on average, of HIGHER quality than pro...   Employed full-t
88878   I am a developer by profession  Yes   Less than once per year   The quality of OSS and closed source software ...   Employed full-t
88882   I am a developer by profession  Yes   Never   OSS is, on average, of LOWER quality than prop...   Employed full-time  Netherlands

13777 rows × 84 columns

# to narrow things down let's just select the column of salary
df.loc[filt, 'ConvertedComp']

Respondent
6         366420.0
16        455352.0
22        103000.0
26        114000.0
32       1100000.0
           ...
88874    2000000.0
88876     180000.0
88877    2000000.0
88878     130000.0
88882     588012.0
Name: ConvertedComp, Length: 13777, dtype: float64

# in order to narrow these columns down a bit let's just grab a certain number of columns so I'll grab the country the
# programming languages and the salary

df.loc[filt, ['Country', 'LanguageWorkedWith', 'ConvertedComp' , 'Age']].head(5)

  Country   LanguageWorkedWith  ConvertedComp   Age
```

```
Respondent
6   Canada  Java;R;SQL  366420.0  28.0
16  United Kingdom  Bash/Shell/PowerShell;C#;HTML/CSS;JavaScript;T...   455352.0  26.0
22  United States   Bash/Shell/PowerShell;C++;HTML/CSS;JavaScript;...   103000.0  47.0
26  United States   Bash/Shell/PowerShell;C++;C#;HTML/CSS;JavaScri...   114000.0  34.0
32  United States   Bash/Shell/PowerShell;HTML/CSS;JavaScript;PHP;...   1100000.0   21.0

# I wanted to filter out the survey results here so that I only see the results from 'Somalia'
filter_by_country = (df['ConvertedComp'] > 100000) & (df['Country'] == 'Somalia')

df.loc[filter_by_country]

  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize   DevType   Y
Respondent
6246  I am not primarily a developer, but I write co...   Yes   Less than once a month but more than once per ...   The quality of OSS and 

# When you have many countries to deal with, writing conditions for each one can be a real pain. so instead of thtat what we can do is ,
# so now let's select five countries Kenya, Austria, Chine, Russia , and Somalia what we can do is make a list of that countires and then
# filter out once all
list_countries = ['Somalia' , 'Kenya' , 'Austria', 'Chine','Russia' ]
filter_by_country = (df['ConvertedComp'] > 100000) & (df['Country'].isin(list_countries))

df.loc[filter_by_country]

  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize   DevType   Y
Respondent
1365  I am a developer by profession  Yes   Less than once per year   The quality of OSS and closed source software ...   Employed full-tim
3658  I am a developer by profession  Yes   Less than once per year   OSS is, on average, of LOWER quality than prop...   Independent contr
4080  I am a developer by profession  Yes   Never   OSS is, on average, of HIGHER quality than pro...   Employed full-time  Austria   Yes, 
6246  I am not primarily a developer, but I write co...   Yes   Less than once a month but more than once per ...   The quality of OSS and 
9438  I am a developer by profession  Yes   Once a month or more often  The quality of OSS and closed source software ...   Employed full-t
---------------------------------------------------------------------------
80150   I am a developer by profession  Yes   Less than once a month but more than once per ...   The quality of OSS and closed source soft
81311   I am not primarily a developer, but I write co...   Yes   Less than once per year   The quality of OSS and closed source software .
81871   I am a developer by profession  Yes   Once a month or more often  The quality of OSS and closed source software ...   Independent c
82020   I am not primarily a developer, but I write co...   Yes   Less than once a month but more than once per ...   OSS is, on average, o
86717   I am a developer by profession  Yes   Less than once a month but more than once per ...   The quality of OSS and closed source soft

# just to narrow down these results to make sure to where we can see the country right off
df.loc[filter_by_country, 'Country']

Respondent
1365     Austria
3658     Austria
4080     Austria
6246     Somalia
9438     Austria
15570    Austria
15933    Austria
16093    Austria
16931    Austria
18697      Kenya
19581    Austria
---------------------------------------------------------------------------
82020    Austria
86717    Austria
Name: Country, dtype: object



# let me show you one more common filter operation that you'll probably use a lot so we can actually use string methods
#  within pandas  as well to do some alterations to our data frame

# let's say that we only want to look at people who answered that they knew Python as a programming language, first of all
#  the column that lists the programming languages that each person said that they know is that [language worked with]
# column so let's select that column

df['LanguageWorkedWith']

Respondent
1                          HTML/CSS;Java;JavaScript;Python
2                                     C++;HTML/CSS;Python
3                                                HTML/CSS
4                                     C;C++;C#;Python;SQL
5              C++;HTML/CSS;Java;JavaScript;Python;SQL;VBA
                               ...
88377                        HTML/CSS;JavaScript;Other(s):
88601                                                  NaN
88802                                                  NaN
88816                                                  NaN
88863    Bash/Shell/PowerShell;HTML/CSS;Java;JavaScript...
```

```
Name: LanguageWorkedWith, Length: 88883, dtype: object

# we can see that we get some programming languages here and each different language is separated by a semicolon

filter_python = df['LanguageWorkedWith'].str.contains('Python')
# in this query we going to get error but let's see what the error actually to undresntand and then solve it

df.loc[filter_python]  #

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[41], line 1
----> 1 df.loc[filter_python]
---------------------------------------------------------------------------
    136     return False
    137 return True

ValueError: Cannot mask with non-boolean array containing NA / NaN values

# as yu can see the error we using .contain method into a column that has non (which is not string ) so to solve this what
#  we can do is to say don't check the column if its NAN
filter_python = df['LanguageWorkedWith'].str.contains('Python', na=False)

df.loc[filter_python]

  MainBranch  Hobbyist  OpenSourcer   OpenSource  Employment  Country   Student   EdLevel   UndergradMajor  EduOther  OrgSize   DevType   Ye
Respondent
1   I am a student who is learning to code  Yes   Never   The quality of OSS and closed source software ...   Not employed, and not looking
2   I am a student who is learning to code  No  Less than once per year   The quality of OSS and closed source software ...   Not employed,
4   I am a developer by profession  No  Never   The quality of OSS and closed source software ...   Employed full-time  United States   No
5   I am a developer by profession  Yes   Once a month or more often  OSS is, on average, of HIGHER quality than pro...   Employed full-tim
8   I code primarily as a hobby   Yes   Less than once per year   OSS is, on average, of HIGHER quality than pro...   Not employed, but loo
...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   ...   .
84539   NaN   Yes   Less than once a month but more than once per ...   The quality of OSS and closed source software ...   Employed full-t
85738   NaN   Yes   Never   The quality of OSS and closed source software ...   Not employed, and not looking for work  Brazil  Yes, full-t
86566   NaN   Yes   Less than once a month but more than once per ...   OSS is, on average, of HIGHER quality than pro...   Retired   Switz
87739   NaN   Yes   Less than once per year   OSS is, on average, of HIGHER quality than pro...   Employed part-time  Czech Republic  Yes,
88212   NaN   No  Less than once per year   OSS is, on average, of HIGHER quality than pro...   Employed full-time  Spain   No  Secondary s

36443 rows × 84 columns

# just to narrow down these results to make sure, let's just grap language worked with colums
df.loc[filter_python, 'LanguageWorkedWith']

Respondent
1                           HTML/CSS;Java;JavaScript;Python
2                                       C++;HTML/CSS;Python
4                                       C;C++;C#;Python;SQL
5               C++;HTML/CSS;Java;JavaScript;Python;SQL;VBA
8         Bash/Shell/PowerShell;C;C++;HTML/CSS;Java;Java...
                               ...
84539     Bash/Shell/PowerShell;C;C++;HTML/CSS;Java;Java...
85738       Bash/Shell/PowerShell;C++;Python;Ruby;Other(s):
86566       Bash/Shell/PowerShell;HTML/CSS;Python;Other(s):
87739              C;C++;HTML/CSS;JavaScript;PHP;Python;SQL
88212                            HTML/CSS;JavaScript;Python
Name: LanguageWorkedWith, Length: 36443, dtype: object
```