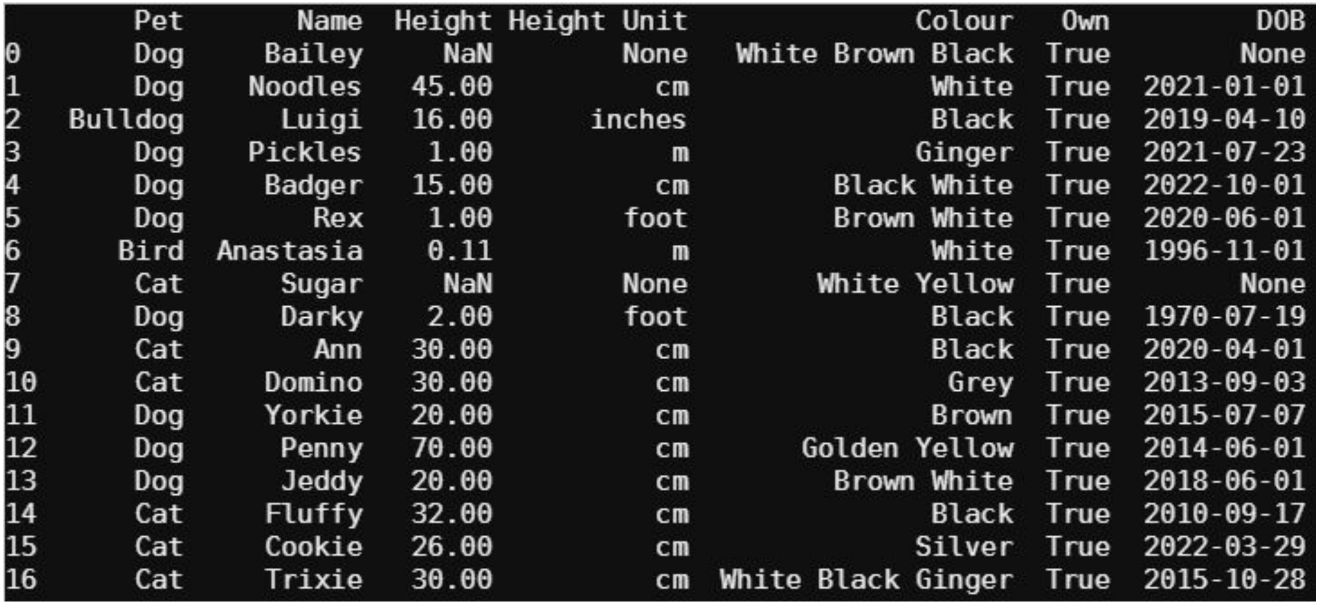
Data Science with Python [DSP 2023-JAN]

Rev 0.1 - 25th January 2023	First Class - Initial file creation
Rev 0.2 - 1st February 2023	Second Class
Rev 0.3 - 8th February 2023	Third Class
Rev 0.4 - 1st March 2023	Fourth Class
Rev 0.5 - 8st March 2023	Fifth Class

pandas’s DataFrame Python Object

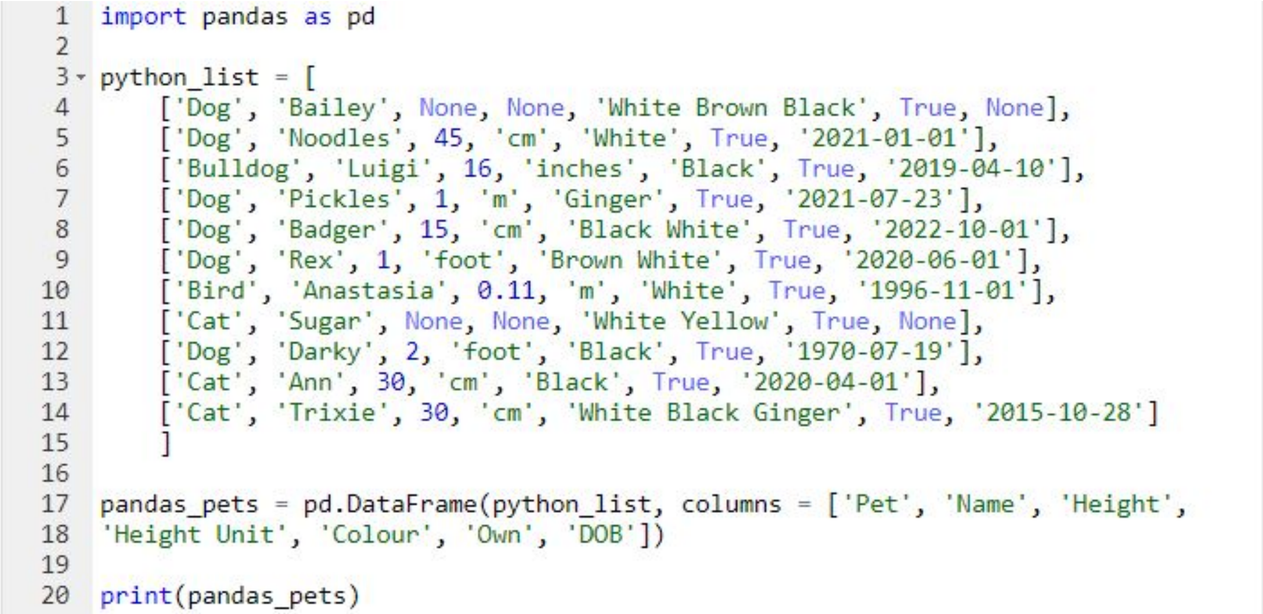
- The pandas Python module (library) was started by Wes McKinney in 2008 and has now grown into one of the most popular Python modules.
- DataFrames are at the core of the pandas Python module and are used to represent rectangular data, aka tabular data.
- Rectangular data is represented by rows and columns, similar to a Spreadsheet or SQL database table.
- Rows are indexed with a number and columns can be named.



```
        Pet       Name  Height Height Unit              Colour   Own         DOB
0       Dog     Bailey     NaN        None   White Brown Black  True        None
1       Dog    Noodles   45.00          cm               White  True  2021-01-01
2   Bulldog      Luigi   16.00      inches               Black  True  2019-04-10
3       Dog    Pickles    1.00           m              Ginger  True  2021-07-23
4       Dog     Badger   15.00          cm         Black White  True  2022-10-01
5       Dog        Rex    1.00        foot         Brown White  True  2020-06-01
6      Bird  Anastasia    0.11           m               White  True  1996-11-01
7       Cat      Sugar     NaN        None        White Yellow  True        None
8       Dog      Darky    2.00        foot               Black  True  1970-07-19
9       Cat        Ann   30.00          cm               Black  True  2020-04-01
10      Cat     Domino   30.00          cm                Grey  True  2013-09-03
11      Dog     Yorkie   20.00          cm               Brown  True  2015-07-07
12      Dog      Penny   70.00          cm       Golden Yellow  True  2014-06-01
13      Dog      Jeddy   20.00          cm         Brown White  True  2018-06-01
14      Cat     Fluffy   32.00          cm               Black  True  2010-09-17
15      Cat     Cookie   26.00          cm              Silver  True  2022-03-29
16      Cat     Trixie   30.00          cm  White Black Ginger  True  2015-10-28
```

DataFrame Class

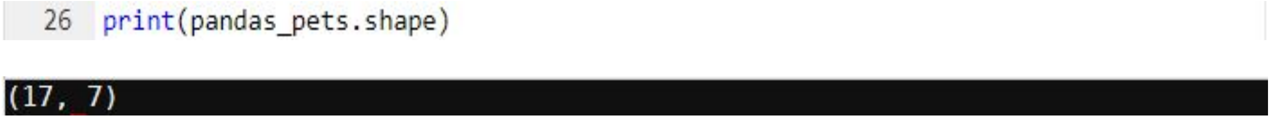
- The pandas DataFrame class is the blueprint or the template for a DataFrame object that is based on the pandas DataFrame class's common attributes and methods. The pandas DataFrame constructor is a way to define a new DataFrame object that contains both data and functionality.
- DataFrame objects functionality is defined by the pandas DataFrame class.
- Dataframe objects data can be added from a Python list, a data file, another DataFrame object, etc.
- **`pd`** is the standard alias for the pandas module.



```python
import pandas as pd

python_list = [
    ['Dog', 'Bailey', None, None, 'White Brown Black', True, None],
    ['Dog', 'Noodles', 45, 'cm', 'White', True, '2021-01-01'],
    ['Bulldog', 'Luigi', 16, 'inches', 'Black', True, '2019-04-10'],
    ['Dog', 'Pickles', 1, 'm', 'Ginger', True, '2021-07-23'],
    ['Dog', 'Badger', 15, 'cm', 'Black White', True, '2022-10-01'],
    ['Dog', 'Rex', 1, 'foot', 'Brown White', True, '2020-06-01'],
    ['Bird', 'Anastasia', 0.11, 'm', 'White', True, '1996-11-01'],
    ['Cat', 'Sugar', None, None, 'White Yellow', True, None],
    ['Dog', 'Darky', 2, 'foot', 'Black', True, '1970-07-19'],
    ['Cat', 'Ann', 30, 'cm', 'Black', True, '2020-04-01'],
    ['Cat', 'Trixie', 30, 'cm', 'White Black Ginger', True, '2015-10-28']
    ]

pandas_pets = pd.DataFrame(python_list, columns = ['Pet', 'Name', 'Height',
'Height Unit', 'Colour', 'Own', 'DOB'])

print(pandas_pets)
```

DataFrame Object `shape` Property

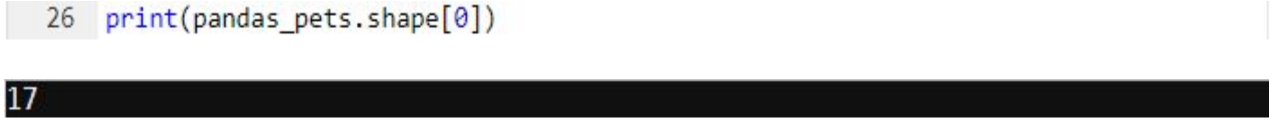
- The **`shape`** property holds the number of rows and columns of the DataFrame.



```
print(pandas_pets.shape)
```

```
(17, 7)
```

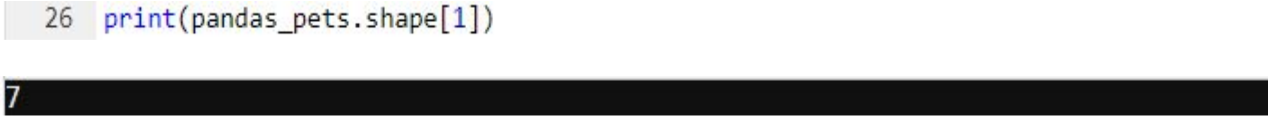
- The first index [0] of the **`shape`** property holds the row count.



```
print(pandas_pets.shape[0])
```

```
17
```

- The second index [1] of the **`shape`** property holds the column count.



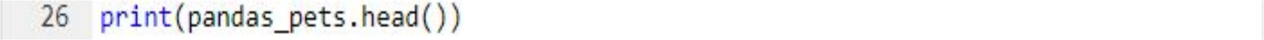
```
print(pandas_pets.shape[1])
```

```
7
```

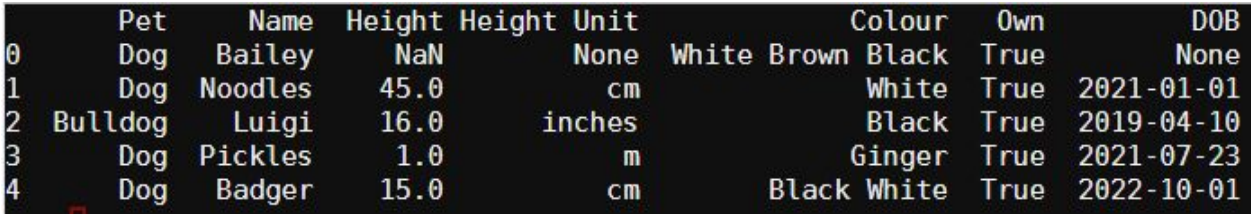
Note: You may need to restart your eu.pythonanywhere.com console for correctly returned data.

DataFrame Object `head()` Method

- The **`head()`** method returns the first 5 rows of the dataframe.



```
print(pandas_pets.head())
```

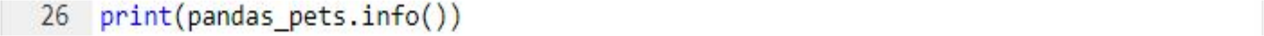


```
       Pet     Name  Height Height Unit              Colour  Own         DOB
0      Dog   Bailey     NaN        None  White Brown Black  True        None
1      Dog  Noodles    45.0          cm              White  True  2021-01-01
2  Bulldog    Luigi    16.0      inches              Black  True  2019-04-10
3      Dog  Pickles     1.0           m             Ginger  True  2021-07-23
4      Dog   Badger    15.0          cm        Black White  True  2022-10-01
```

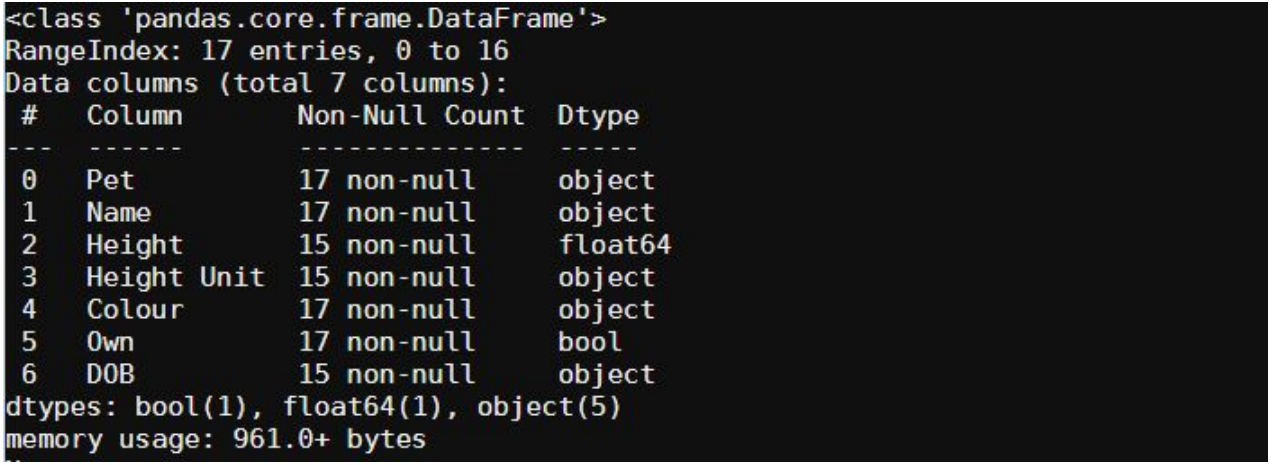
Note: You may need to restart your eu.pythonanywhere.com console for correctly returned data.

DataFrame Object `info()` Method

- The **`info()`** method returns information on each of the columns.



```
print(pandas_pets.info())
```



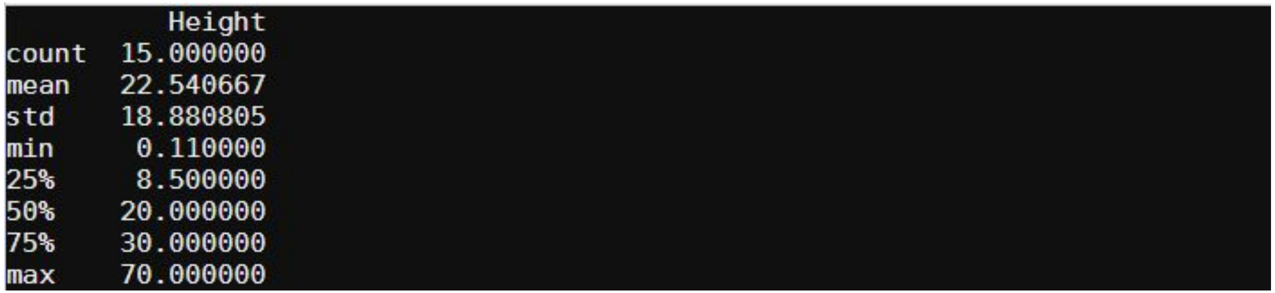
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17 entries, 0 to 16
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Pet          17 non-null     object
 1   Name         17 non-null     object
 2   Height       15 non-null     float64
 3   Height Unit  15 non-null     object
 4   Colour       17 non-null     object
 5   Own          17 non-null     bool
 6   DOB          15 non-null     object
dtypes: bool(1), float64(1), object(5)
memory usage: 961.0+ bytes
```

Note: You may need to restart your eu.pythonanywhere.com console for correctly returned data.

DataFrame Object `describe()` Method

- The **`describe()`** method returns calculated summary statistics for each column.

```
print(pandas_pets.describe())
```



```
           Height
count   15.000000
mean    22.540667
std     18.880805
min      0.110000
25%      8.500000
50%     20.000000
75%     30.000000
max     70.000000
```

Note: You may need to restart your eu.pythonanywhere.com console for correctly returned data.

Dealing With Missing Data: NaN & None

1. Impute Missing Data

- Fill in missing data with average values or estimated values.

2. Drop Missing Data

- If the missing data is limited in amount, dropping it might be the simplest option.

3. Create an Indicator Variable

- If missing data is a frequent occurrence, creating an indicator variable that represents whether data is missing or not can be helpful.

4. Use Machine Learning Algorithms

- Machine learning algorithms such as decision trees and random forests are capable of handling missing data.

Missing Data in pandas_pets DataFrame

- The data in our pandas_pets DataFrame is limited in amount, therefore we decided to drop it.

Additional Resources

University of North Florida - Pseudocode Examples:
https://www.unf.edu/~broggio/cop2221/2221pseu.htm#:~:text=Pseudocode%20is%20an%20artificial%20and,%2C%20for%2C%20if%2C%20switch.