# Pandas

Cleaning and Finding Relationships in Data Frame

# Selecting columns and filtering

- We can show only selected columns in a DataFrame

```
print(df["Pulse"])   #df.Pulse
print(df["Pulse"].dtype)
print(df[["Pulse", "Calories"]])
```

- We can subset our data by applying Boolean indexing (filter). For example if we want to subset the rows in which Pulse < 100

```
print(df["Pulse"] < 100)
print(df[df["Pulse"] < 100])
```

https://www.w3schools.com/python/pandas/pandas_csv.asp

# Cleaning Data

- If your data is large and some part of that is not necessary or has bad data, then you can fix it by using pandas functions.
- Bad data could be: empty cells, data in wrong format, duplicates, etc.
- For example, the data set in data.csv contains some bad data

| | | | | |
|---|---|---|---|---|
| 15 | 60 | 98 | 120 | 215.2 |
| 16 | 60 | 100 | 120 | 300.0 |
| 17 | 45 | 90 | 112 | NaN |
| 18 | 60 | 103 | 123 | 323.0 |
| 19 | 45 | 97 | 125 | 243.0 |
| 20 | 60 | 108 | 131 | 364.2 |
| 21 | 45 | 100 | 119 | 282.0 |
| 22 | 60 | 130 | 101 | 300.0 |
| 23 | 45 | 105 | 132 | 246.0 |
| 24 | 60 | 102 | 126 | 334.5 |
| 25 | 60 | 100 | 120 | 250.0 |
| 26 | 60 | 92 | 118 | 241.0 |
| 27 | 60 | 103 | 132 | NaN |

https://www.w3schools.com/python/pandas/pandas_csv.asp

# Cleaning Data

- We can use the dropna() method to remove/drop all NULL values.

```
import pandas as pd
df = pd.read_csv(“data.csv”)
newDf = df.dropna()
print(newDf.to_string())
```

- By default, the dropna() method does not change the original DataFrame. If you want to change the original DataFrame, use the inplace = True argument

```
import pandas as pd
df = pd.read_csv('data.csv')
df.dropna(inplace = True)
print(df.to_string())
```

# Cleaning Data

- Instead of removing entire row to get rid of NULL values, we can insert new values.
- The fillna() method allows us to replace empty cells with a value.
- For example, let's replace NULL values with the number 555:

```
import pandas as pd
df = pd.read_csv('data.csv')
df.fillna(555, inplace = True)
```

# Cleaning Data

- If we want to replace not all empty cells but only a selected column’s empty cells, then we can specify the column name.

```
import pandas as pd
df = pd.read_csv('data.csv')
df["Calories"].fillna(555, inplace = True)
```

# Cleaning Data

- We can use the mean() and median() methods to calculate the respective values and replace the empty cells with those values.

```
import pandas as pd
df = pd.read_csv('data.csv')
x = df["Calories"].mean()                    # .median()
df["Calories"].fillna(x, inplace = True)
```

# Cleaning Data

```
import pandas as pd
df = pd.read_csv("data.csv")

df.loc[1,"Duration"] = 111

print(df.to_string())
```

# Drop Columns

```
to_drop = ["Pulse", "Maxpulse“]
df.drop(to_drop, inplace=True, axis=1)
print(df.to_string())
```

# Finding Relationships

- The corr() method calculates the relationship between each column in your data set.

```
import pandas as pd
df = {
        "ar1": [10, 20, 80],
        "ar2": [25.3, 69.2, 100.5]
}
data = pd.DataFrame(df)
print(data.corr())
```

# Finding Relationships

```
import pandas as pd
df = pd.read_csv('data.csv')

df.corr()
```

- The corr() method ignores any NULL, non-numeric data type or columns in the Dataframe.