

# Pandas Introduction

Python for Data Analysis

Press Space for next page →



# Reza Rizky



[rezarzky](#)



[rezarzky](#)

# Bakhtiar A.



[maziyank](#)



[maziyank](#)

# Pandas ?

- Library yang menyediakan struktur data dan analisis data.
- Digunakan untuk memanipulasi data, mengubah dimensi data, mengecek data, dan lain sebagainya.
- Panda diambil dari istilah "**panel data**", istilah ekonometrik untuk kumpulan data.

Versi terakhir 1.2.4 / 12 April 2021

Website: [pandas.pydata.org](https://pandas.pydata.org)



# Topik

- Data Frame and Series
- Reading Data
- Selecting and Filtering Data
- Sorting Data
- Grouping Data
- Handling Missing Data
- Handling Duplicates
- Apply Function
- Brief Plotting
- Saving Data

# Menggunakan Pandas

Secara default sudah tersedia di paket anaconda. Sehingga kita bisa langsung melakukan impor pustaka pandas.

```
import pandas as pd  
  
# your code here ....
```



# Data Frame dan Series

**DataFrame** adalah struktur data 2-dimensi yang berbentuk tabular (mempunyai baris dan kolom).  
**Series** adalah struktur data 1-dimensi yang berbentuk tabular (mempunyai baris dan kolom).

**Series**

	apples
0	3
1	2
2	0
3	1

**Series**

	oranges
0	0
1	3
2	7
3	2

+

=

**DataFrame**

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

# Data Frame Structure

Axis 0 / "index"      Axis 1 / "columns" →

↓

Index Labels

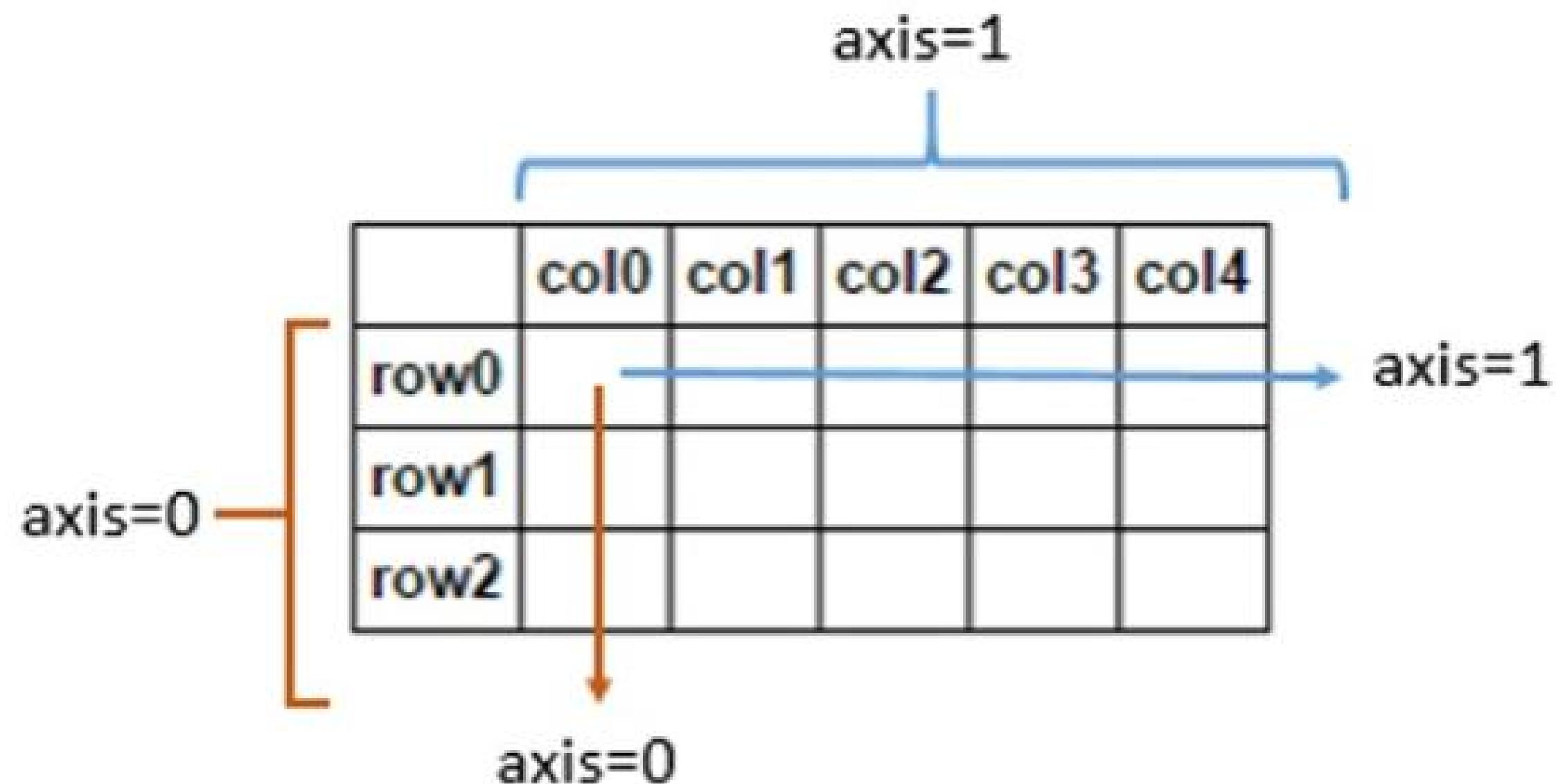
	Column Labels						
	color	director_name	num_critic_for_reviews	...	imdb_score	aspect_ratio	movie_facebook_likes
0	Color	James Cameron	723.0	...	7.9	1.78	33000
1	Color	Gore Verbinski	302.0	...	7.1	2.35	0
2	Color	Sam Mendes	602.0	...	6.8	2.35	85000
3	Color	Christopher Nolan	813.0	...	8.5	2.35	164000
4	NaN	Doug Walker	NaN	...	7.1	NaN	0
...	...	...	...	...	...	...	...
4911	Color	Scott Smith	1.0	...	7.7	NaN	84
4912	Color	NaN	43.0	...	7.5	16.00	32000
4913	Color	Benjamin Roberds	13.0	...	6.3	NaN	16
4914	Color	Daniel Hsia	14.0	...	6.3	2.35	660
4915	Color	Jon Gunn	43.0	...	6.6	1.85	456

4916 rows × 28 columns

Missing Values      Truncated Data      Data / Values

# Data Frame Axis (1 atau 0 ?)

- Axis 0 merepresentasikan rows dan axis 1 merepresentasikan columns.
- Series hanya mempunyai 1 axis, axis 0.





# Reading Data

``pd.read_csv()```

DataFrame dari file CSV

``pd.read_excel()```

DataFrame dari file Excel

``pd.read_html()```

DataFrame dari file HTML

Contoh:

```
import pandas  
  
# membuat dataframe dari file titanic.csv  
df = pd.read_csv('titanic.csv')
```

Check out the guides for more.

# Peek Data

Beberapa fungsi umum untuk melihat data secara sekilas:

```
# Membaca beberapa n baris pertama  
df.head(n)
```

```
# Membaca beberapa n baris terakhir  
df.tail(n)
```

```
# Membaca n baris data secara sampling  
df.sample(n)
```

```
# Membaca n baris data terbesar sesuai kolom a  
df.nlargest(a, n)
```

```
# Membaca n baris data terkecil sesuai kolom a  
df.nsmallest(a, n)
```

# Informasi Data Frame

Beberapa cara untuk mengetahui profil DataFrame

```
# mengetahui dimensi DataFrame  
df.shape  
  
# jumlah data (baris x kolom) pada dataframe  
df.size  
  
# list kolom pada dataframe  
df.columns  
  
# list index pada dataframe  
df.index  
  
# informasi dataframe, kolom tipe data dsb.  
df.info()  
  
#informasi tipe data pada dataframe  
df.dtypes
```



# Selecting Data

Terdapat beberapa macam cara untuk mengeksplorasi data pada DataFrame

## Mengakses data dengan nama kolom

```
# mengakses satu kolom  
df.nama_kolom  
df["age"]  
  
# mengakses lebih dari satu kolom  
df[["sex", "age"]]
```

## Memilih data dengan label tertentu dari posisi index (.iloc).

```
# memilih data pada baris ke 2 dan kolom ke 3  
df.iloc[1,2]  
  
# misal: memilih data pada baris ke 2 hingga ke 5 dan kolom ke 3  
df.iloc[1:5, 2]  
  
# misal: memilih data pada seluruh baris ke untuk kolom ke 3  
df.iloc[:, 2]
```

# Selecting Data

Terdapat beberapa macam cara untuk mengeksplorasi data pada DataFrame

Memilih data dengan nama index / kolom (.loc).

```
# menjadikan kolom Name sebagai index
df.set_index("Name", inplace = True)

# memilih baris dengan nama index "Allen, Mr. William Henry" dan kolom "sex"
df.loc["Allen, Mr. William Henry", "Sex"]

# memilih baris dengan index mulai dari "Allen, Mr. William Henry"
# sampai "Graham, Miss. Margaret Edith" dan kolom "fare" sampai "class"
df.loc["Allen, Mr. William Henry":"Graham, Miss. Margaret Edith", "Fare":"Pclass"]
```

Memilih single value data dengan nama index / kolom (.at).

```
df.at["Allen, Mr. William Henry", "sex"]
# memilih baris dengan nama index "Allen, Mr. William Henry" dan kolom "sex"
```

# Filtering Data

Menggunakan conditional.

```
# memilih data berdasarkan kondisi tertentu  
df[boolean/conditional]  
  
# misal: memilih data dengan usia lebih dari 50  
df[df.Age > 50]  
  
# misal: memilih data dengan usia lebih dari 50 dan kurang dari 70  
df[df.Age > 50 & df.Age < 79]
```

Menggunakan query.

```
# memilih data berdasarkan kondisi tertentu  
df.query(expression)  
  
# misal: memilih data dengan usia lebih dari 50  
df.query("Age > 50")  
  
# misal: memilih data dengan usia lebih dari 50 dan kurang dari 70  
df.query("Age > 50 & Age < 70")
```

# Sorting Data

Untuk melakukan sorting data pada DataFrame:

```
df.sort_values(by, axis=0, ascending=True, inplace=False,  
kind='quicksort', na_position='last', ignore_index=False, key=None)
```

Contoh:

```
# Hanya satu kolom:  
dataframe.sort_values(by='Age', ascending=False)  
  
# Beberapa kolom:  
df.sort_values(by=["Age", "Pclass"], ascending = (False, True))
```

See the guides.  
-----

# Grouping Data

```
df.groupby(by=None, axis=0, level=None, as_index=True,  
sort=True, group_keys=True, squeeze=<object object>, observed=False, dropna=True)
```

Contoh:

```
# Kelompokan berdasarkan embark_town  
df_grouped = df.groupby(['Embark_town']).mean()  
  
# Aggregate functions: count(), mean(), sum(), min(), max(), median(), quantile(), var(), std()  
  
# Melihat value pada group embark_town  
df_grouped.get_group('Embark_town')
```

See the guides.  
-----

# Handling Missing Value

```
DataFrame.dropna(axis=0, how='any', thresh=None, subset=None, inplace=False)  
DataFrame.fillna(value=None, method=None, axis=None, inplace=False, limit=None, downcast=None)
```

Contoh:

```
# Mengecek missing value  
df.isnull()  
# atau  
df.isna()  
  
# Menghapus missing value  
df.dropna(inplace=True)  
  
# Mengisi NaN value  
df.age.fillna(value=df.age.mean(), inplace=True)
```

See [the guides](#).

# Handling Duplicates

```
df.drop_duplicates(subset=None, keep='first', inplace=False, ignore_index=False)
```

Contoh:

```
# Menghapus duplikat untuk seluruh kolom
```

```
df.drop_duplicates()
```

```
# Menghapus duplikat untuk spesifik kolom
```

```
df.drop_duplicates(subset=['Name'], keep='last')
```

See [the guides](#).

# Apply Function

Fungsi ini dapat diaplikasikan ke DataFrame dan Series

```
df.apply(func, axis=0, raw=False, result_type=None, args=(), **kwds)
```

Contoh:

```
#using function
def makedouble(val):
    return val*2

df.fare.apply(makedouble)

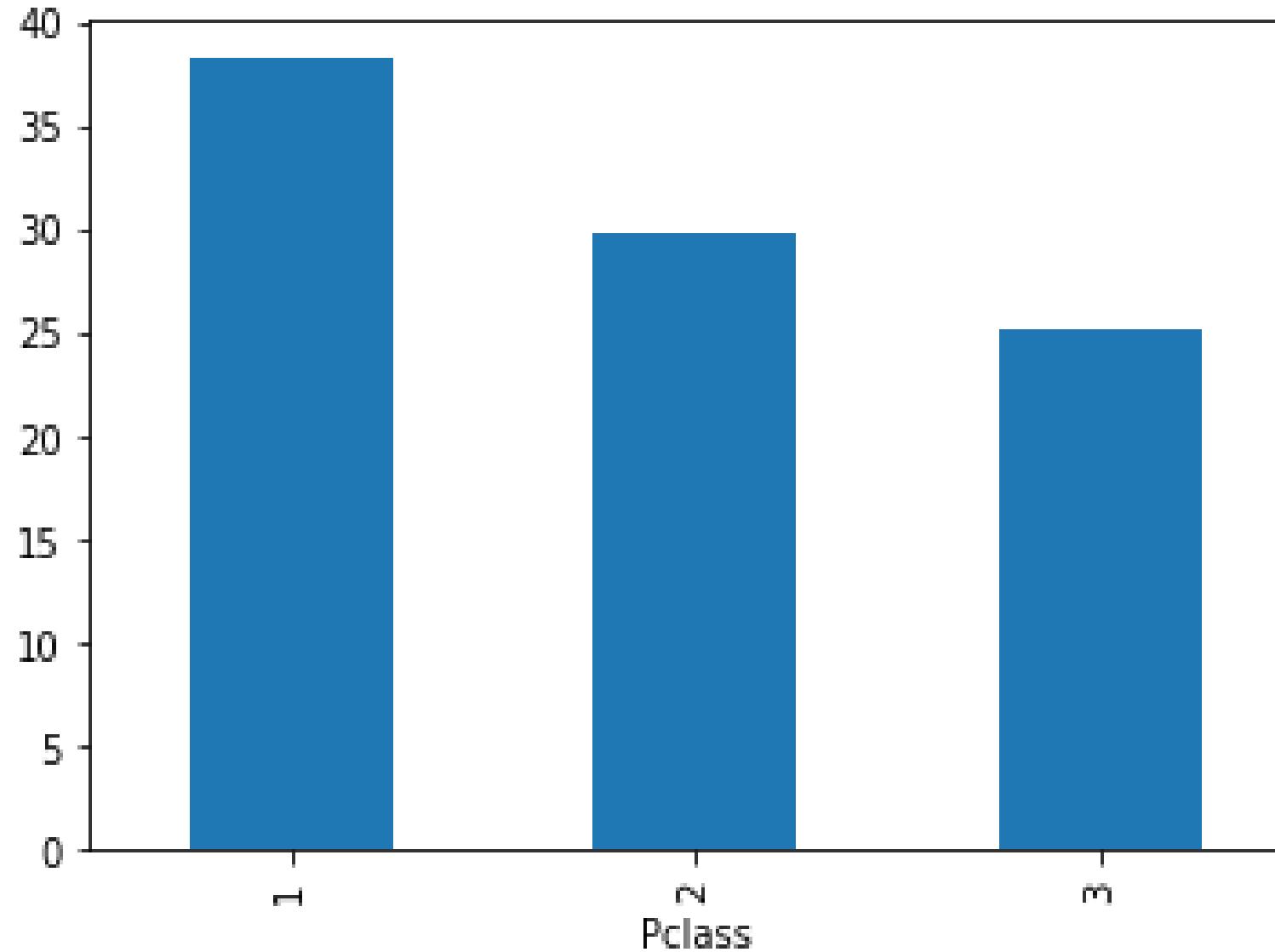
#using lambda
df.fare.apply(lambda x: x*2)
```

See the guides.  
-----

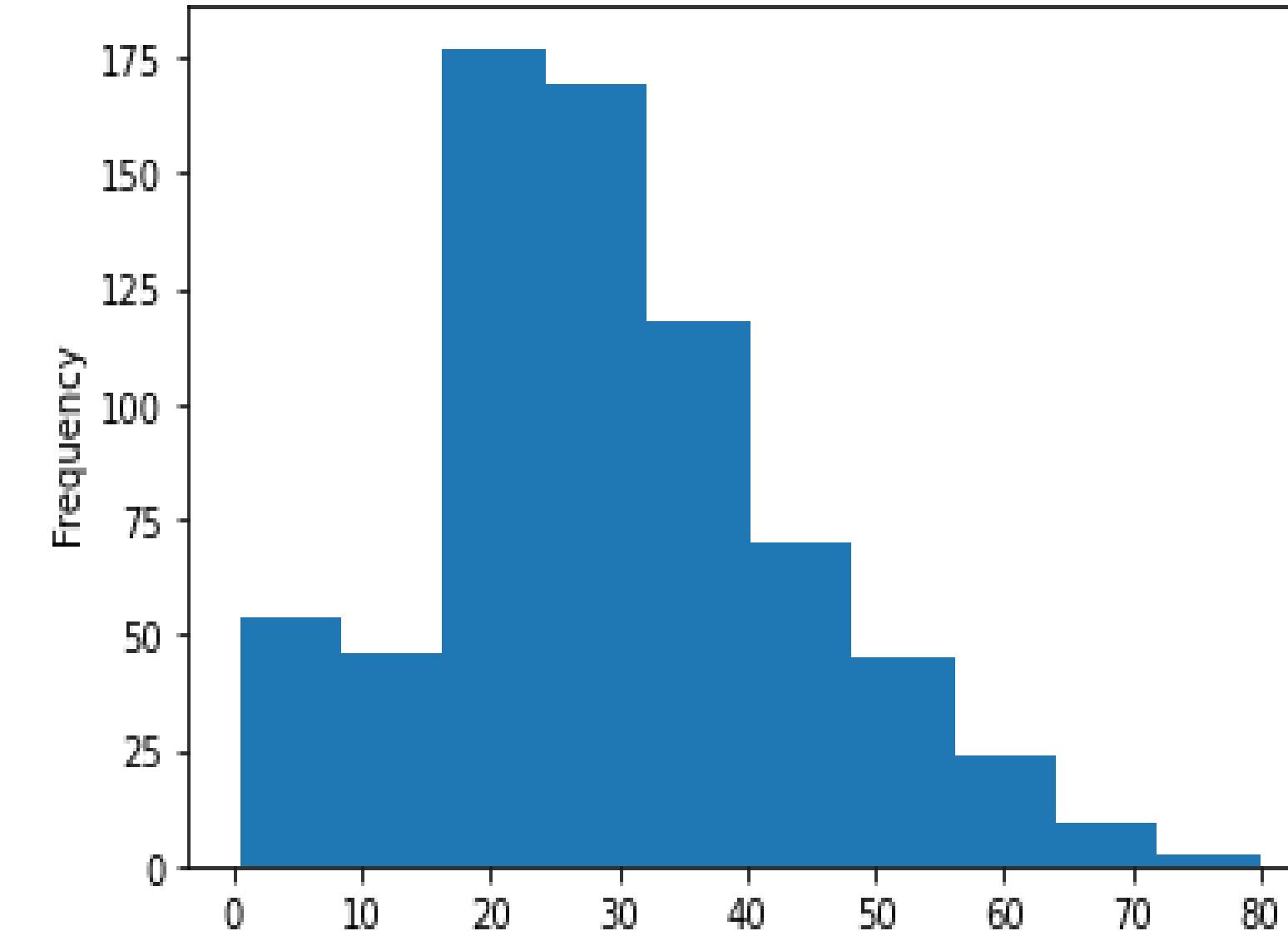
# Brief Plotting - Bar and Hist

Pandas terintegrasi dengan matplotlib sehingga kita bisa menggunakan secara langsung.

```
# Barplot  
df_grouped = df.groupby(by="Pclass").mean()  
df_grouped.Age.plot.bar()
```



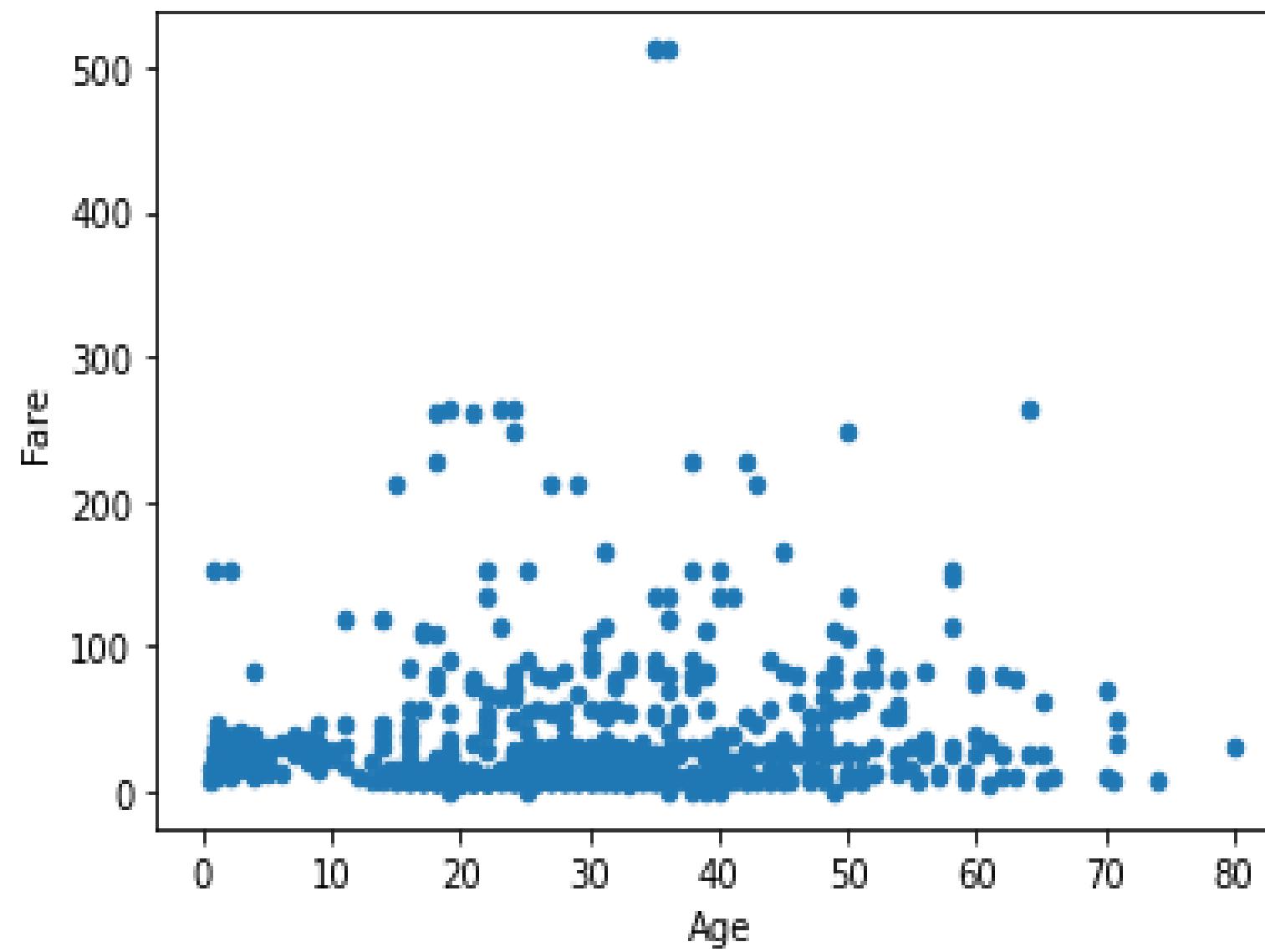
```
# Histogram  
df.Age.plot.hist()
```



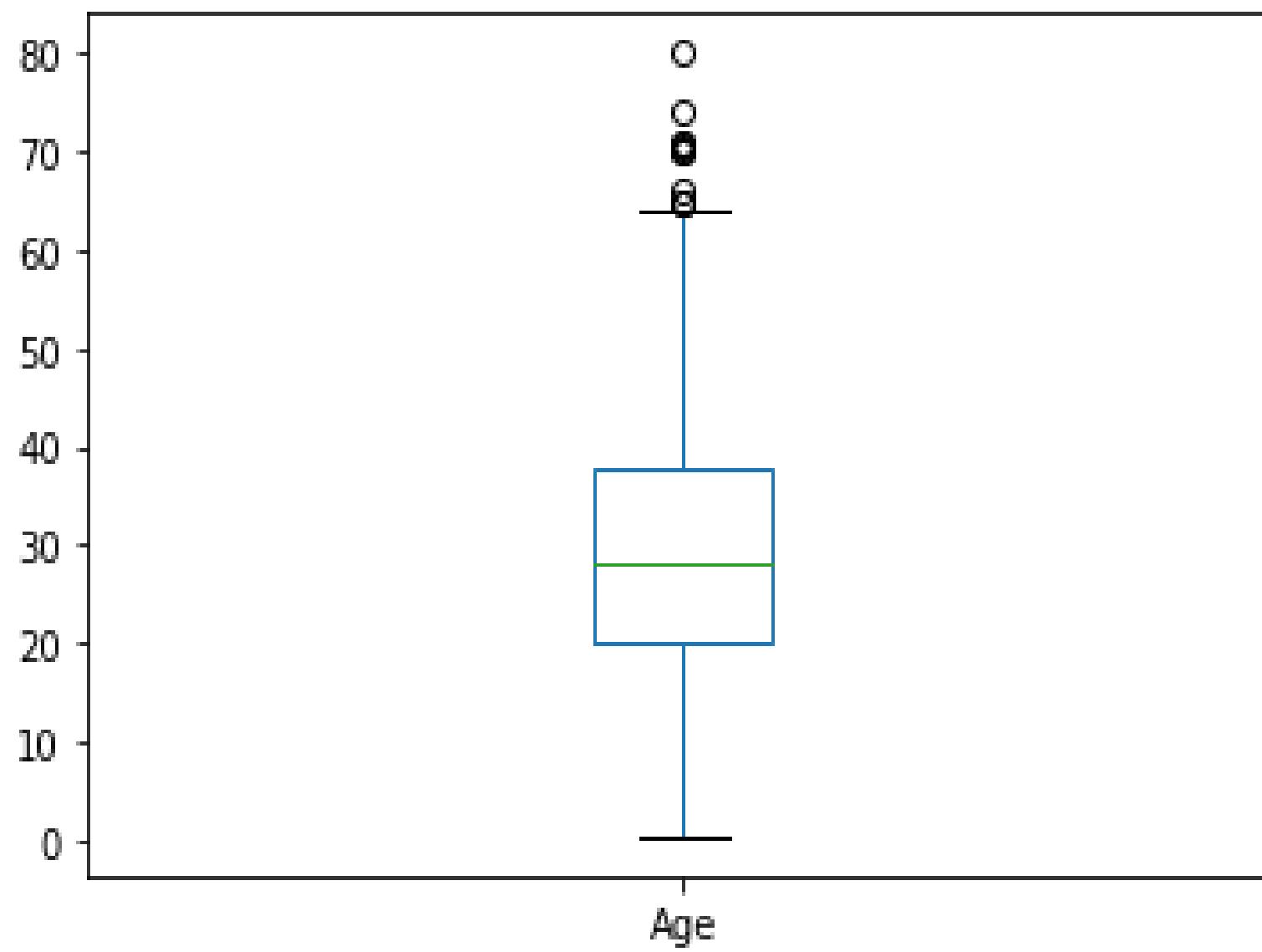
# Brief Plotting - Box and Scatter

Pandas terintegrasi dengan matplotlib sehingga kita bisa menggunakan secara langsung.

```
# Scatterplot  
df.plot.scatter(x="Age", y="Fare")
```



```
# Box Plot  
df.Age.plot.box()
```



# Thank You!

Semoga bermanfaat.