Programación para Analítica de Datos MODULO EN PYTHON – PANDAS

POR: Guillermo Andres De Mendoza Corrales

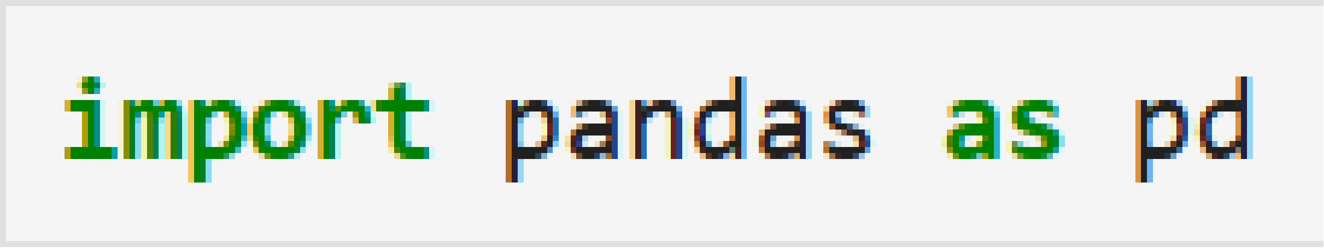
PANDAS LIBRERIA

Link: https://pandas.pydata.org/

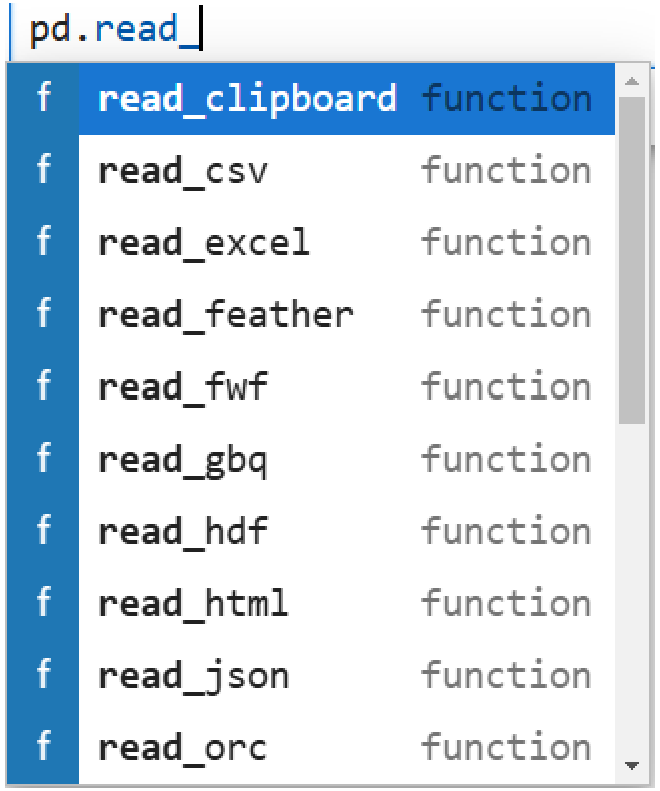
Libreria que facilita las operaciones de:

- Analisis
- Limpieza
- Exploracion
- Manipulacion

PANDAS IMPLEMETACION

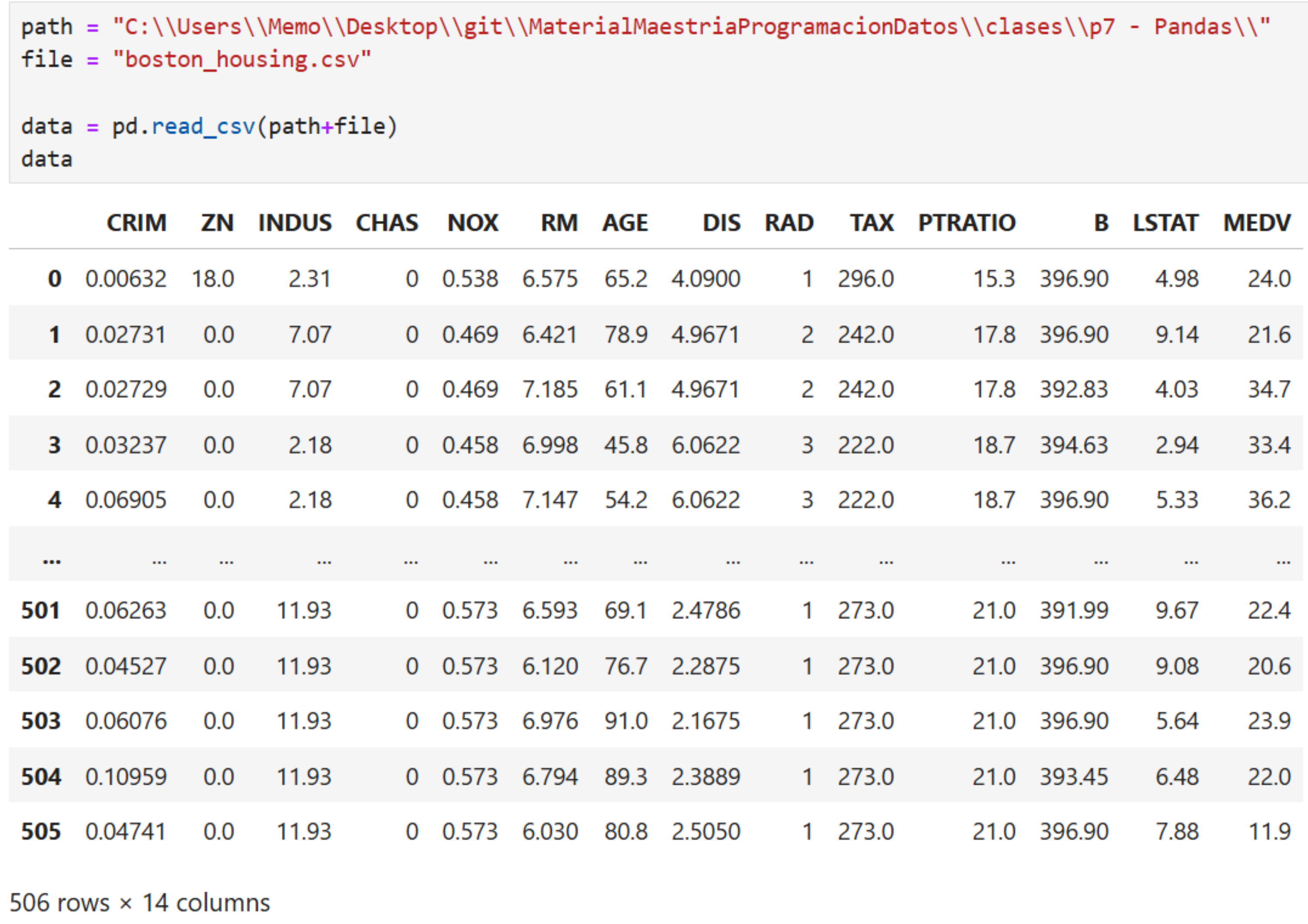


```python
import pandas as pd
```



```python
pd.read_
```

PYTHON CON PANDAS

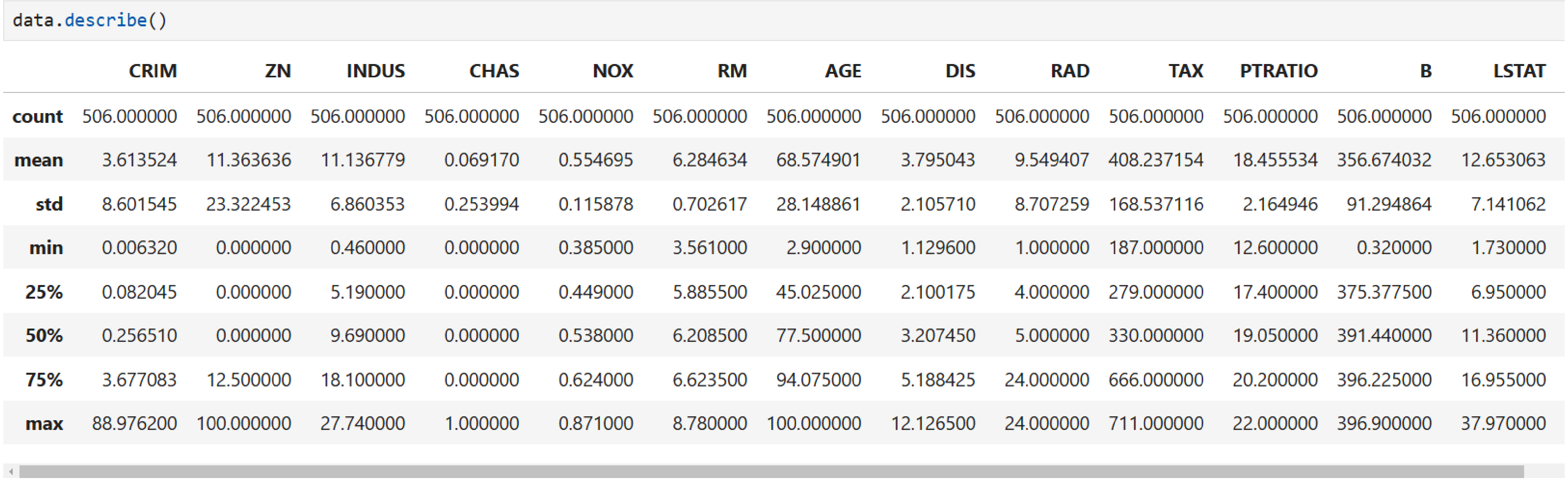


```python
path = "C:\\Users\\Memo\\Desktop\\git\\MaterialMaestriaProgramacionDatos\\clases\\p7 - Pandas\\"
file = "boston_housing.csv"

data = pd.read_csv(path+file)
data
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273.0	21.0	391.99	9.67	22.4
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273.0	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273.0	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273.0	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273.0	21.0	396.90	7.88	11.9

506 rows × 14 columns

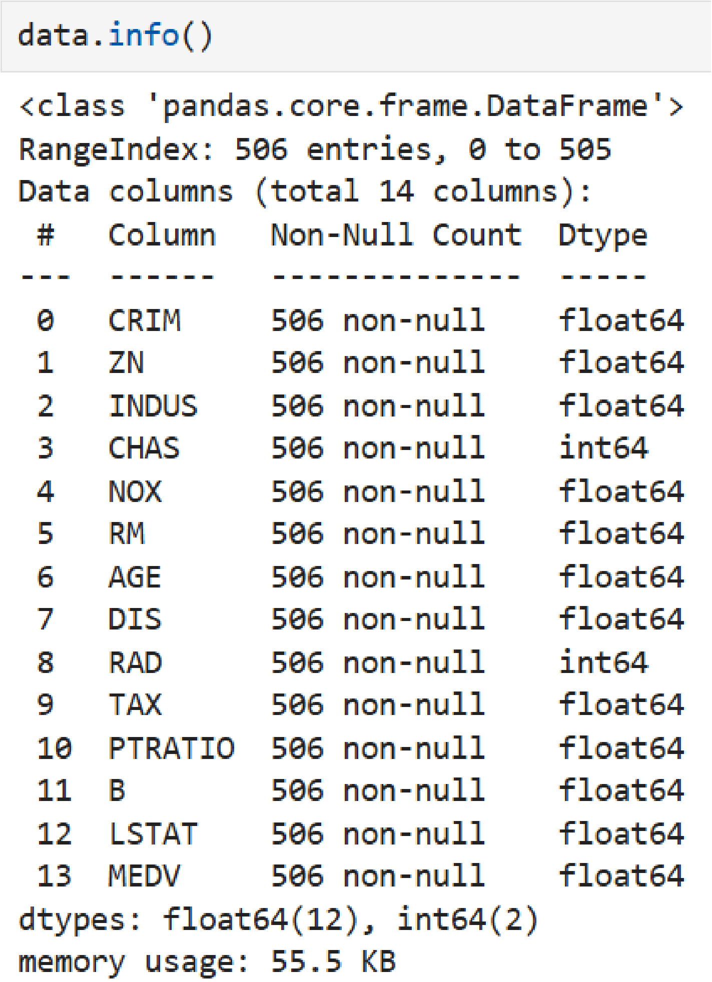
PYTHON CON PANDAS



```
data.describe()
```

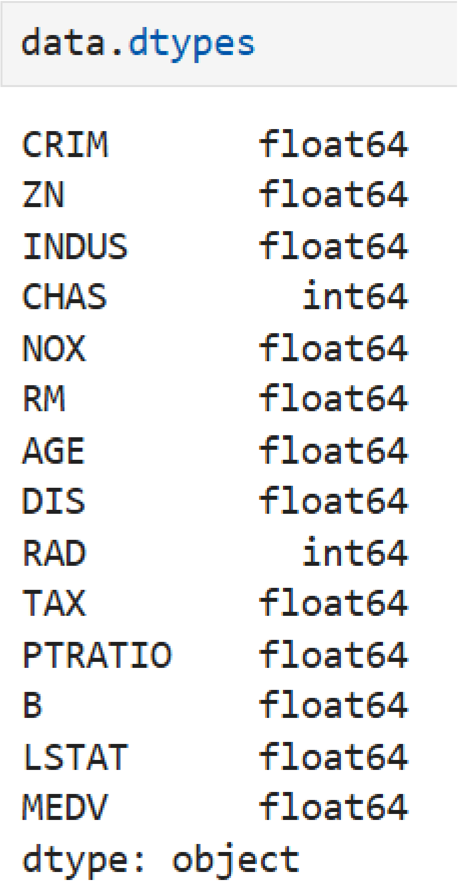
	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	68.574901	3.795043	9.549407	408.237154	18.455534	356.674032	12.653063
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861	2.105710	8.707259	168.537116	2.164946	91.294864	7.141062
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000	1.129600	1.000000	187.000000	12.600000	0.320000	1.730000
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.025000	2.100175	4.000000	279.000000	17.400000	375.377500	6.950000
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.500000	3.207450	5.000000	330.000000	19.050000	391.440000	11.360000
75%	3.677083	12.500000	18.100000	0.000000	0.624000	6.623500	94.075000	5.188425	24.000000	666.000000	20.200000	396.225000	16.955000
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000	12.126500	24.000000	711.000000	22.000000	396.900000	37.970000

PYTHON CON PANDAS - describir



```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   CRIM     506 non-null    float64
 1   ZN       506 non-null    float64
 2   INDUS    506 non-null    float64
 3   CHAS     506 non-null    int64  
 4   NOX      506 non-null    float64
 5   RM       506 non-null    float64
 6   AGE      506 non-null    float64
 7   DIS      506 non-null    float64
 8   RAD      506 non-null    int64  
 9   TAX      506 non-null    float64
 10  PTRATIO  506 non-null    float64
 11  B        506 non-null    float64
 12  LSTAT    506 non-null    float64
 13  MEDV     506 non-null    float64
dtypes: float64(12), int64(2)
memory usage: 55.5 KB
```

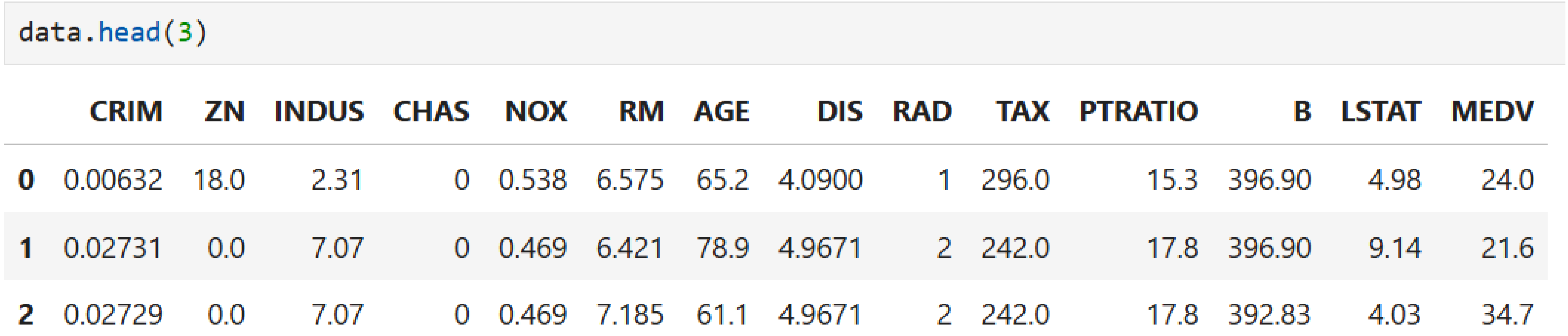


```python
data.dtypes
```

```
CRIM       float64
ZN         float64
INDUS      float64
CHAS         int64
NOX        float64
RM         float64
AGE        float64
DIS        float64
RAD          int64
TAX        float64
PTRATIO    float64
B          float64
LSTAT      float64
MEDV       float64
dtype: object
```

PYTHON CON PANDAS - describir

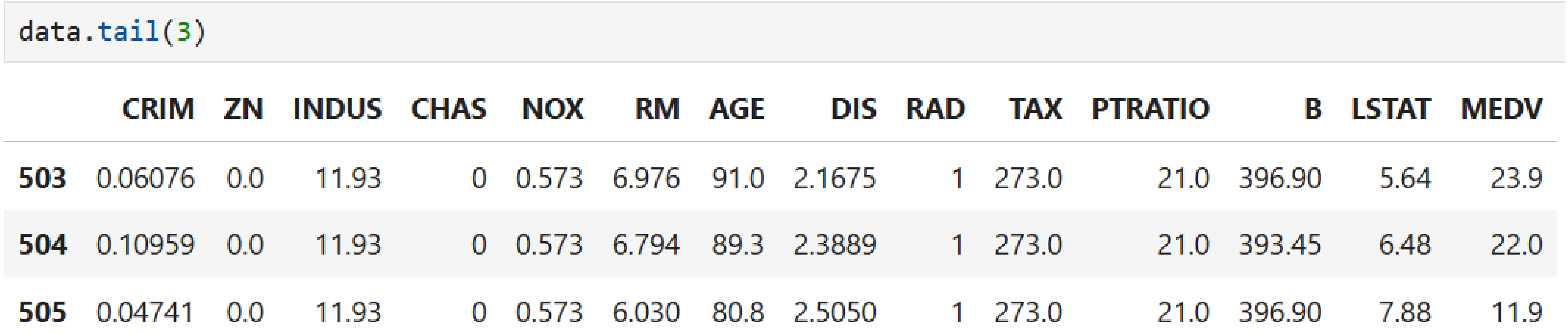
head



```
data.head(3)
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7

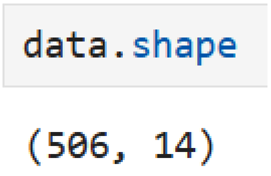
tail



```
data.tail(3)
```

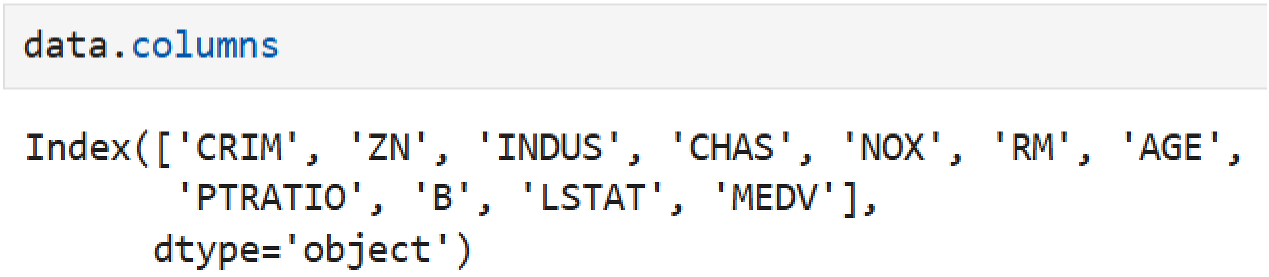
	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273.0	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273.0	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273.0	21.0	396.90	7.88	11.9

PYTHON CON PANDAS - describir



```
data.shape
```

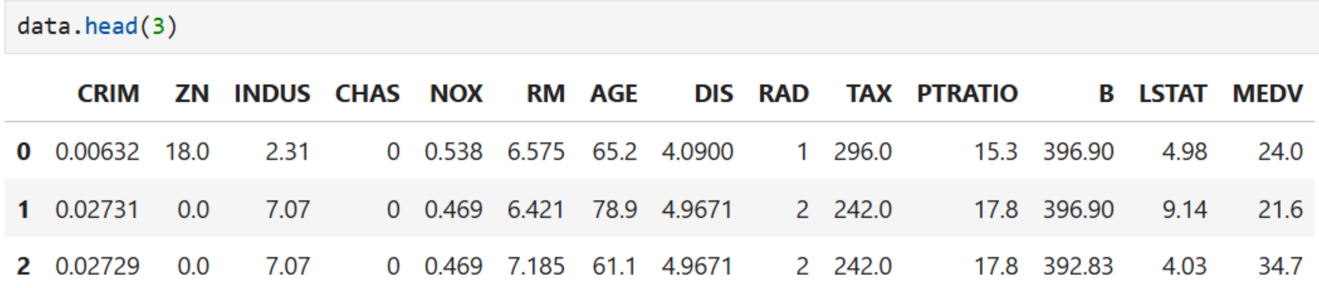
```
(506, 14)
```



```
data.columns
```

```
Index(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE',
       'PTRATIO', 'B', 'LSTAT', 'MEDV'],
      dtype='object')
```

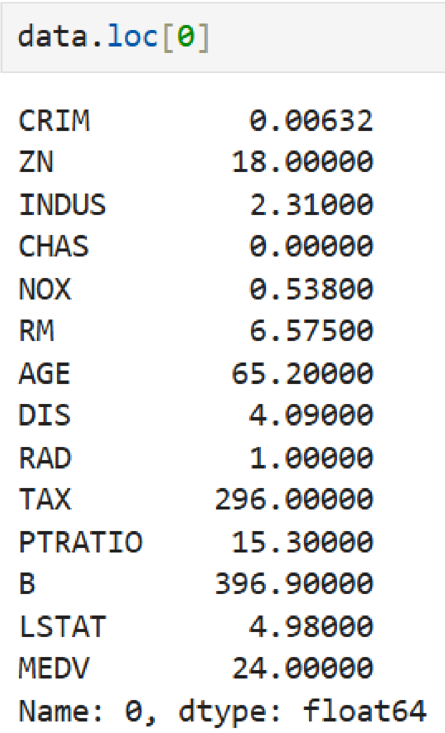
PYTHON CON PANDAS - acceso



```
data.head(3)
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7

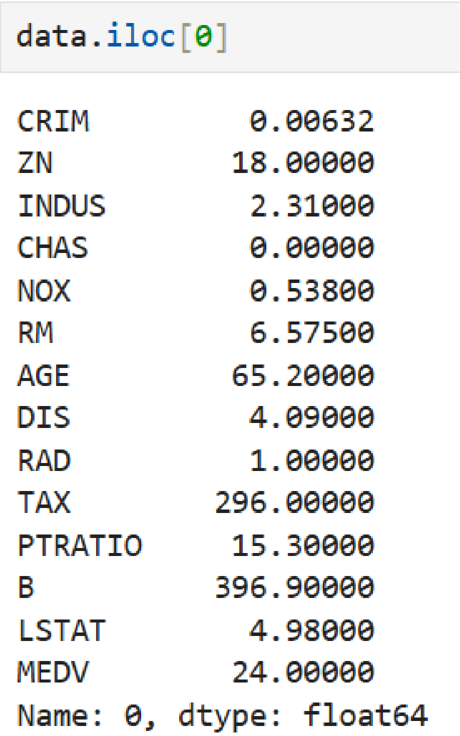
index



```
data.loc[0]
```

```
CRIM         0.00632
ZN          18.00000
INDUS        2.31000
CHAS         0.00000
NOX          0.53800
RM           6.57500
AGE         65.20000
DIS          4.09000
RAD          1.00000
TAX        296.00000
PTRATIO     15.30000
B          396.90000
LSTAT        4.98000
MEDV        24.00000
Name: 0, dtype: float64
```

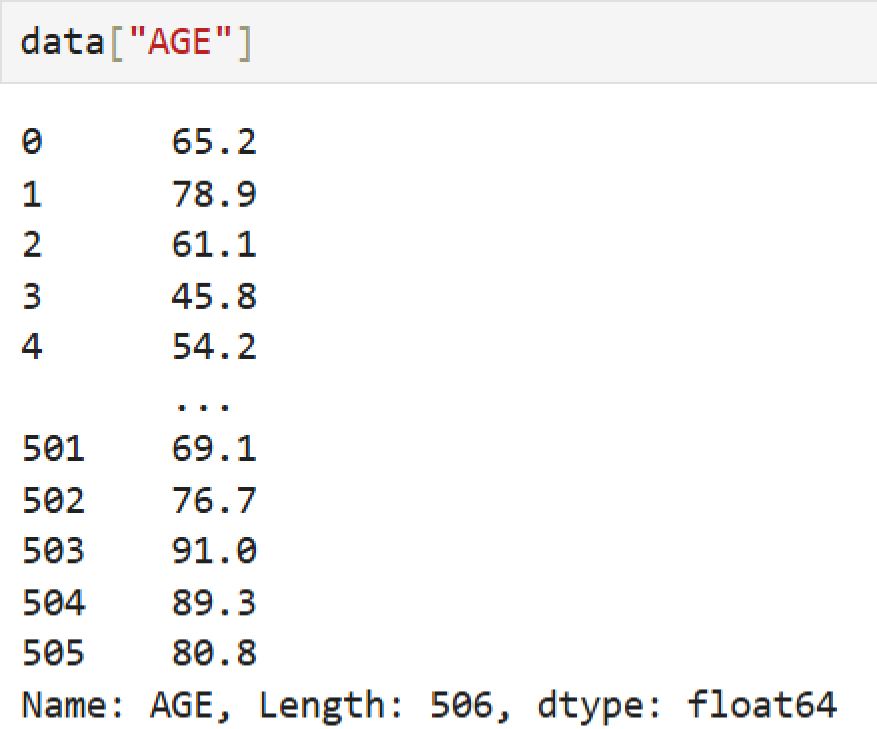
possition



```
data.iloc[0]
```

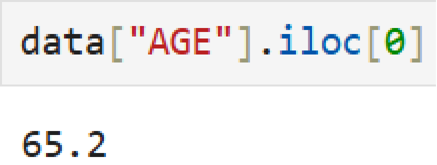
```
CRIM         0.00632
ZN          18.00000
INDUS        2.31000
CHAS         0.00000
NOX          0.53800
RM           6.57500
AGE         65.20000
DIS          4.09000
RAD          1.00000
TAX        296.00000
PTRATIO     15.30000
B          396.90000
LSTAT        4.98000
MEDV        24.00000
Name: 0, dtype: float64
```

PYTHON CON PANDAS - filtro



```
data["AGE"]
```

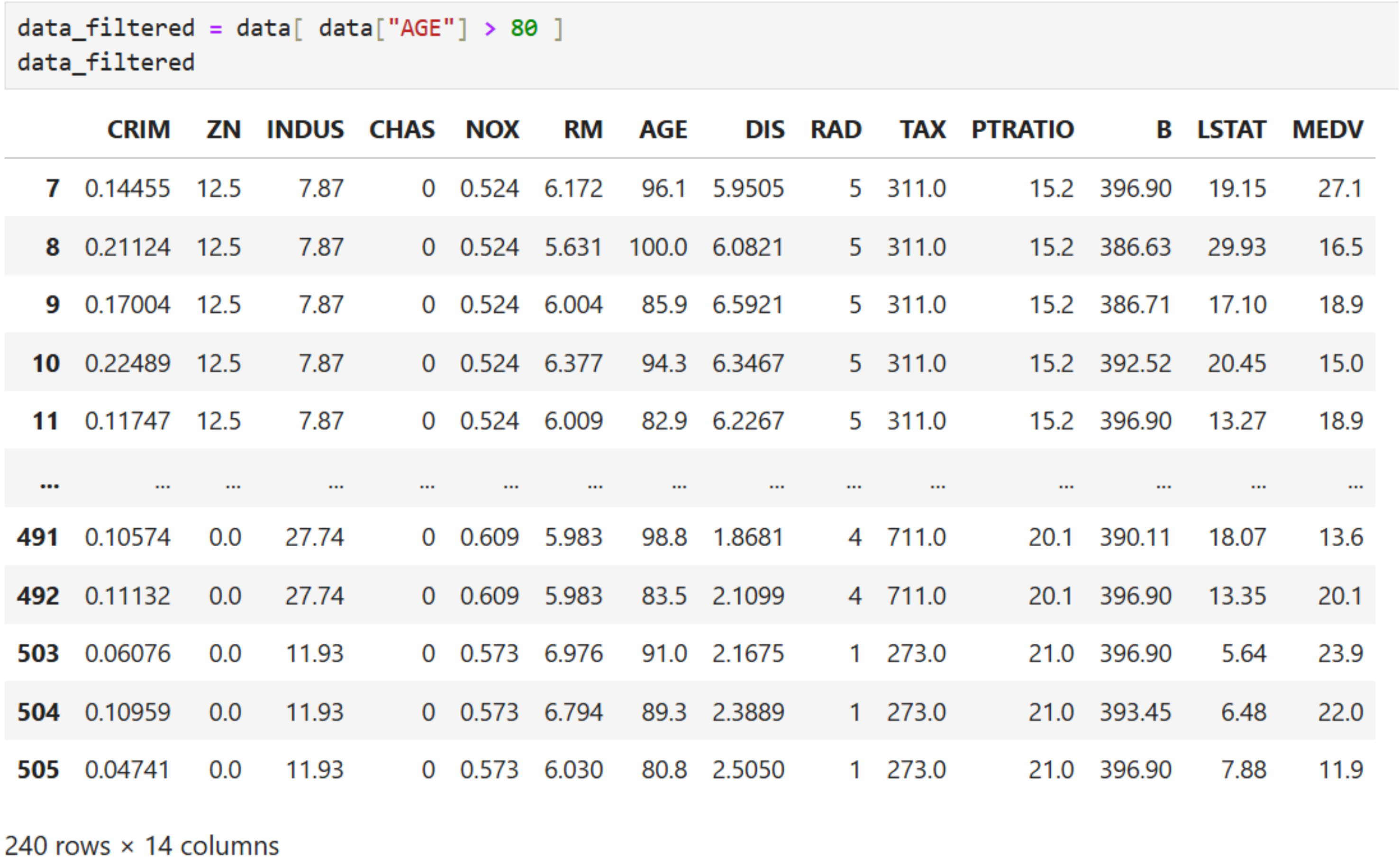
```
0       65.2
1       78.9
2       61.1
3       45.8
4       54.2
        ...
501     69.1
502     76.7
503     91.0
504     89.3
505     80.8
Name: AGE, Length: 506, dtype: float64
```



```
data["AGE"].iloc[0]
```

```
65.2
```

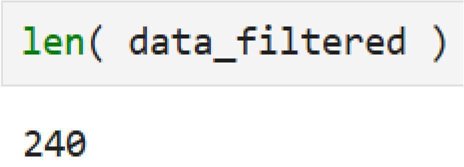
PYTHON CON PANDAS - filtros



```
data_filtered = data[ data["AGE"] > 80 ]
data_filtered
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
7	0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311.0	15.2	396.90	19.15	27.1
8	0.21124	12.5	7.87	0	0.524	5.631	100.0	6.0821	5	311.0	15.2	386.63	29.93	16.5
9	0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311.0	15.2	386.71	17.10	18.9
10	0.22489	12.5	7.87	0	0.524	6.377	94.3	6.3467	5	311.0	15.2	392.52	20.45	15.0
11	0.11747	12.5	7.87	0	0.524	6.009	82.9	6.2267	5	311.0	15.2	396.90	13.27	18.9
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
491	0.10574	0.0	27.74	0	0.609	5.983	98.8	1.8681	4	711.0	20.1	390.11	18.07	13.6
492	0.11132	0.0	27.74	0	0.609	5.983	83.5	2.1099	4	711.0	20.1	396.90	13.35	20.1
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273.0	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273.0	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273.0	21.0	396.90	7.88	11.9

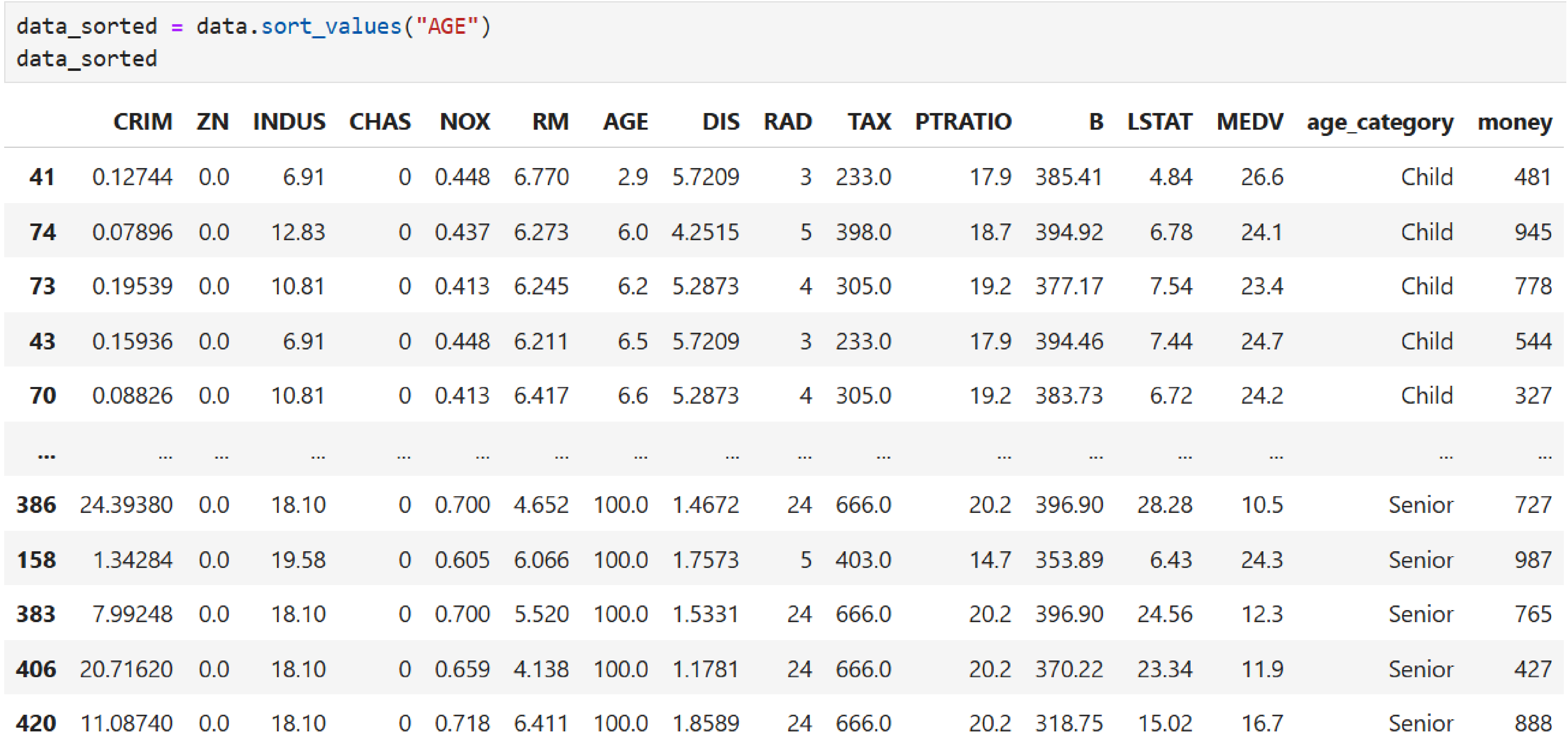
240 rows × 14 columns



```
len( data_filtered )
```

```
240
```

PYTHON CON PANDAS - filtros

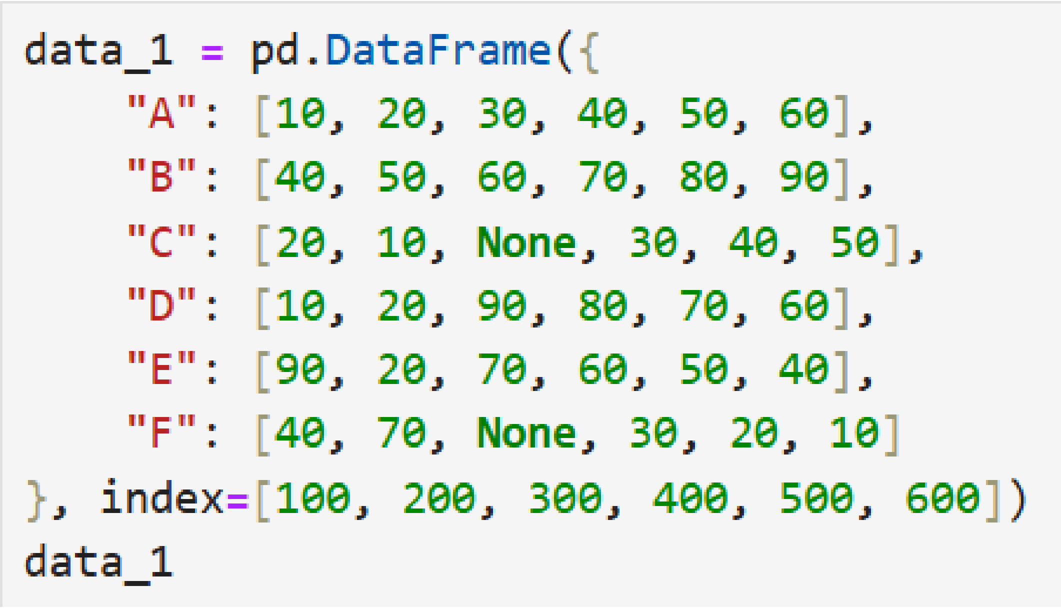


```python
data_sorted = data.sort_values("AGE")
data_sorted
```

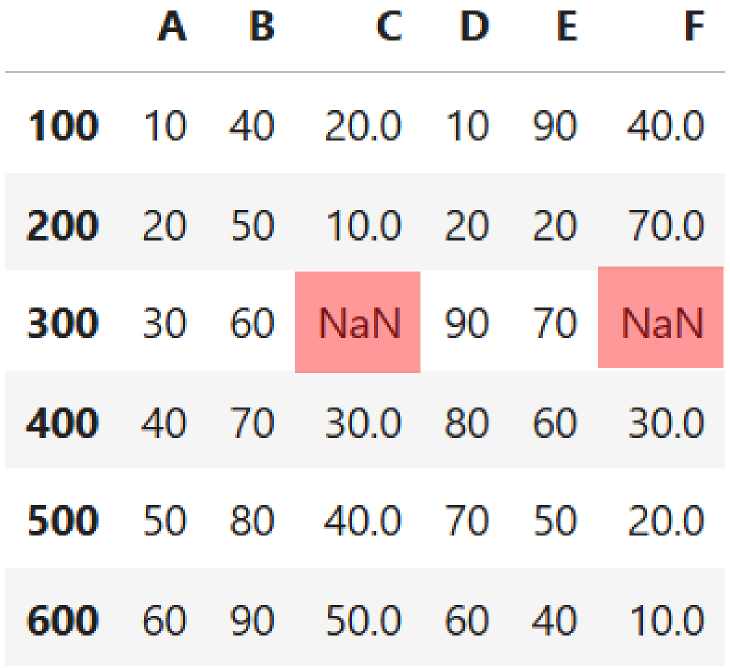
	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV	age_category	money
41	0.12744	0.0	6.91	0	0.448	6.770	2.9	5.7209	3	233.0	17.9	385.41	4.84	26.6	Child	481
74	0.07896	0.0	12.83	0	0.437	6.273	6.0	4.2515	5	398.0	18.7	394.92	6.78	24.1	Child	945
73	0.19539	0.0	10.81	0	0.413	6.245	6.2	5.2873	4	305.0	19.2	377.17	7.54	23.4	Child	778
43	0.15936	0.0	6.91	0	0.448	6.211	6.5	5.7209	3	233.0	17.9	394.46	7.44	24.7	Child	544
70	0.08826	0.0	10.81	0	0.413	6.417	6.6	5.2873	4	305.0	19.2	383.73	6.72	24.2	Child	327
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
386	24.39380	0.0	18.10	0	0.700	4.652	100.0	1.4672	24	666.0	20.2	396.90	28.28	10.5	Senior	727
158	1.34284	0.0	19.58	0	0.605	6.066	100.0	1.7573	5	403.0	14.7	353.89	6.43	24.3	Senior	987
383	7.99248	0.0	18.10	0	0.700	5.520	100.0	1.5331	24	666.0	20.2	396.90	24.56	12.3	Senior	765
406	20.71620	0.0	18.10	0	0.659	4.138	100.0	1.1781	24	666.0	20.2	370.22	23.34	11.9	Senior	427
420	11.08740	0.0	18.10	0	0.718	6.411	100.0	1.8589	24	666.0	20.2	318.75	15.02	16.7	Senior	888

PYTHON CON PANDAS – limpiar valores

Crear dataframes desde cero

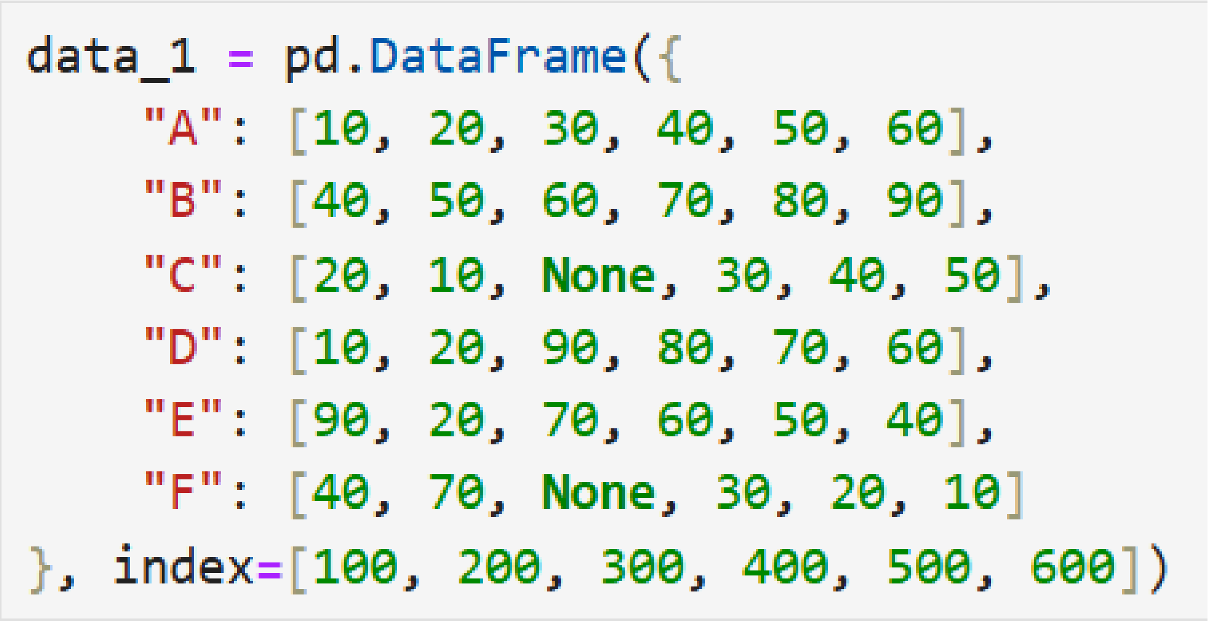


```
data_1 = pd.DataFrame({
    "A": [10, 20, 30, 40, 50, 60],
    "B": [40, 50, 60, 70, 80, 90],
    "C": [20, 10, None, 30, 40, 50],
    "D": [10, 20, 90, 80, 70, 60],
    "E": [90, 20, 70, 60, 50, 40],
    "F": [40, 70, None, 30, 20, 10]
}, index=[100, 200, 300, 400, 500, 600])
data_1
```

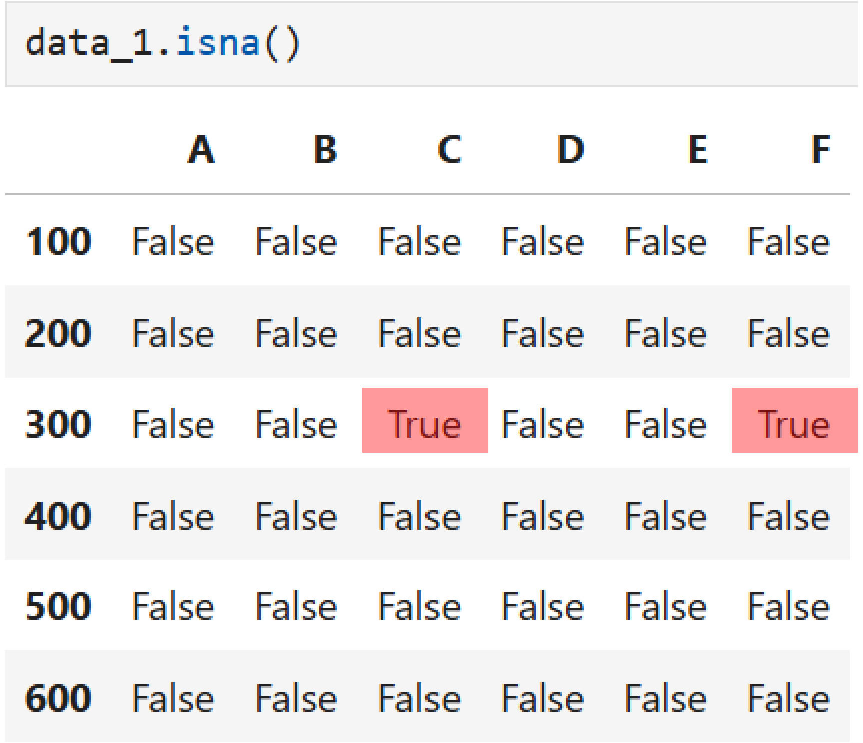


	A	B	C	D	E	F
100	10	40	20.0	10	90	40.0
200	20	50	10.0	20	20	70.0
300	30	60	NaN	90	70	NaN
400	40	70	30.0	80	60	30.0
500	50	80	40.0	70	50	20.0
600	60	90	50.0	60	40	10.0

PYTHON CON PANDAS – limpiar valores



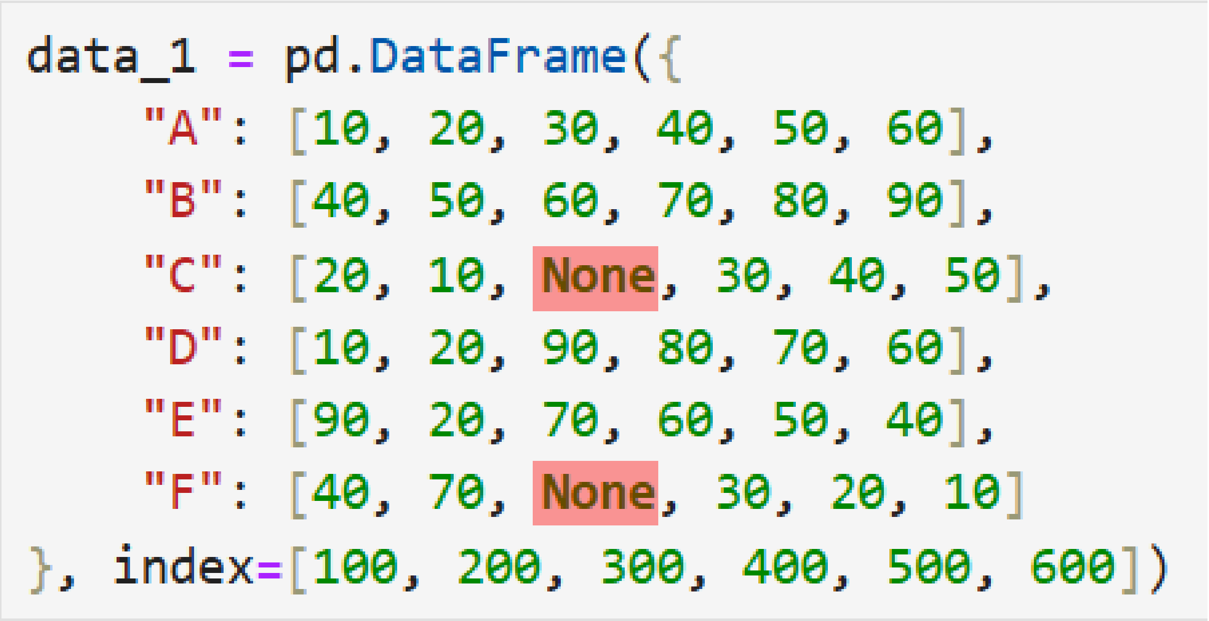
```
data_1 = pd.DataFrame({
    "A": [10, 20, 30, 40, 50, 60],
    "B": [40, 50, 60, 70, 80, 90],
    "C": [20, 10, None, 30, 40, 50],
    "D": [10, 20, 90, 80, 70, 60],
    "E": [90, 20, 70, 60, 50, 40],
    "F": [40, 70, None, 30, 20, 10]
}, index=[100, 200, 300, 400, 500, 600])
```



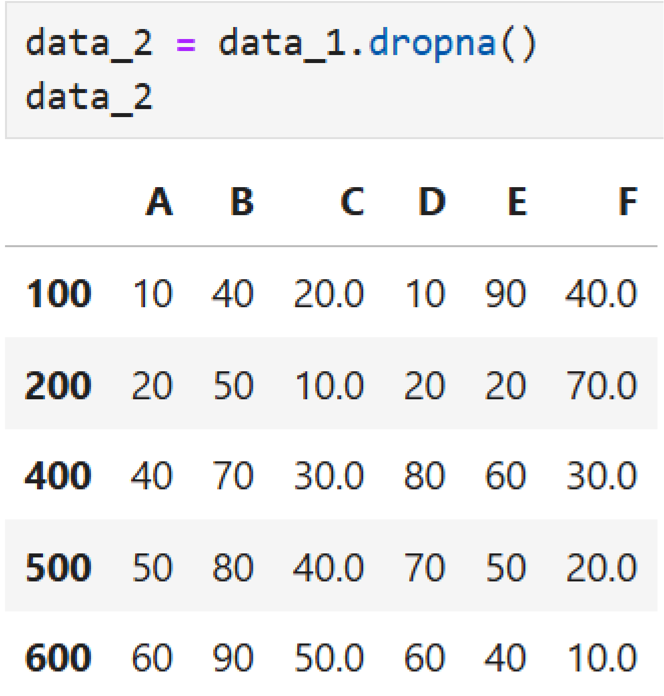
```
data_1.isna()
```

	A	B	C	D	E	F
100	False	False	False	False	False	False
200	False	False	False	False	False	False
300	False	False	True	False	False	True
400	False	False	False	False	False	False
500	False	False	False	False	False	False
600	False	False	False	False	False	False

PYTHON CON PANDAS – limpiar valores



```
data_1 = pd.DataFrame({
    "A": [10, 20, 30, 40, 50, 60],
    "B": [40, 50, 60, 70, 80, 90],
    "C": [20, 10, None, 30, 40, 50],
    "D": [10, 20, 90, 80, 70, 60],
    "E": [90, 20, 70, 60, 50, 40],
    "F": [40, 70, None, 30, 20, 10]
}, index=[100, 200, 300, 400, 500, 600])
```

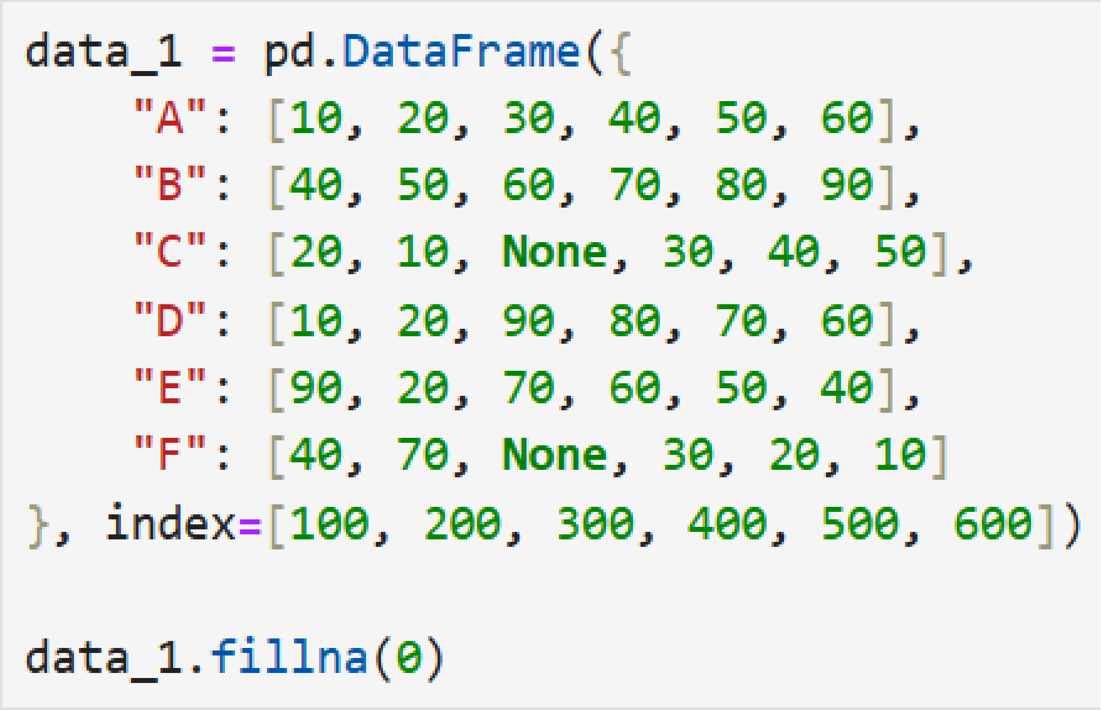


```
data_2 = data_1.dropna()
data_2
```

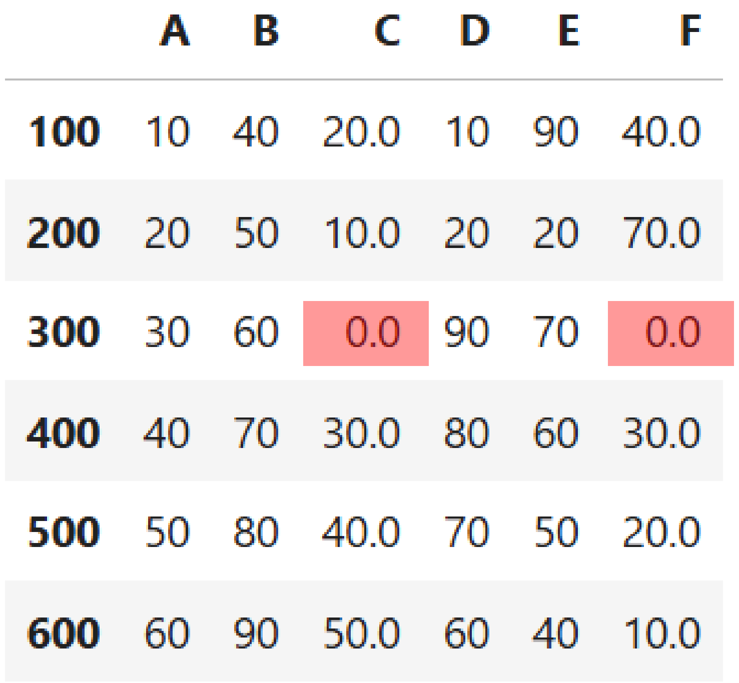
	A	B	C	D	E	F
100	10	40	20.0	10	90	40.0
200	20	50	10.0	20	20	70.0
400	40	70	30.0	80	60	30.0
500	50	80	40.0	70	50	20.0
600	60	90	50.0	60	40	10.0

Crea un nuevo dataframe sin las filas que tienen valores nulos

PYTHON CON PANDAS – limpiar valores

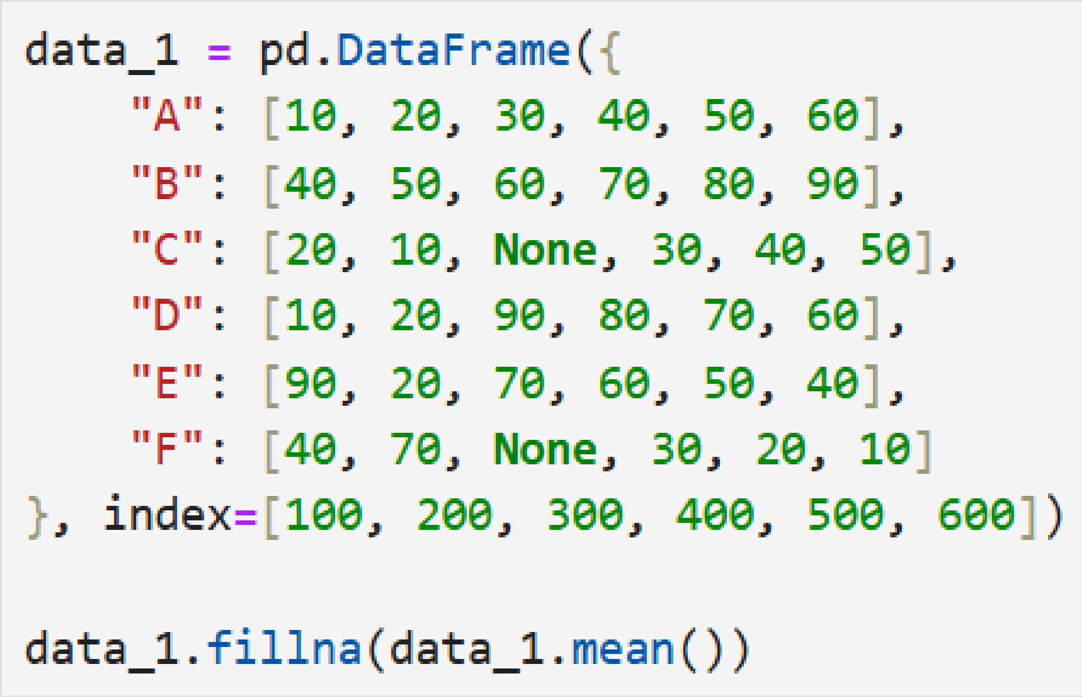


```
data_1 = pd.DataFrame({
    "A": [10, 20, 30, 40, 50, 60],
    "B": [40, 50, 60, 70, 80, 90],
    "C": [20, 10, None, 30, 40, 50],
    "D": [10, 20, 90, 80, 70, 60],
    "E": [90, 20, 70, 60, 50, 40],
    "F": [40, 70, None, 30, 20, 10]
}, index=[100, 200, 300, 400, 500, 600])

data_1.fillna(0)
```

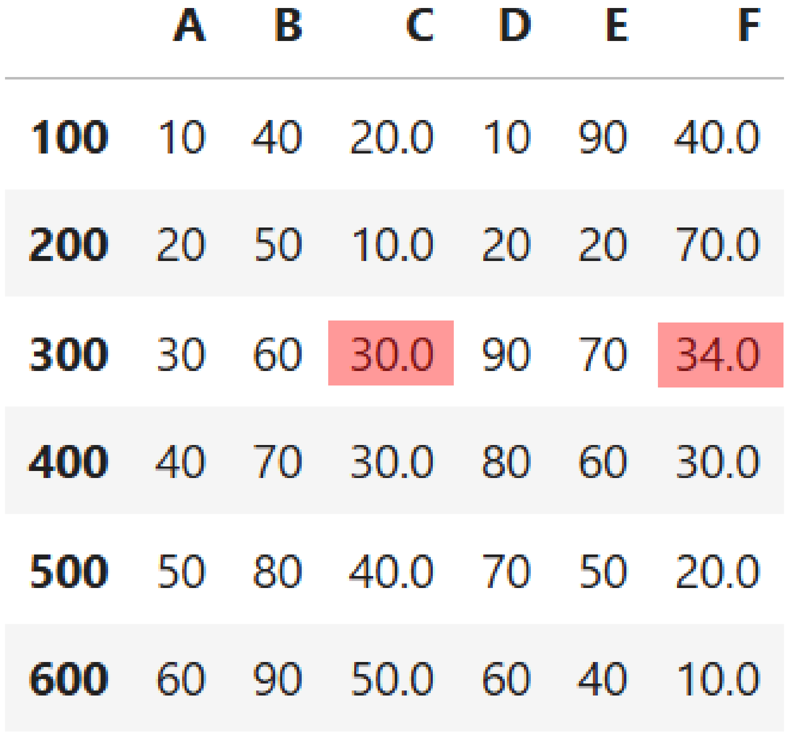


	A	B	C	D	E	F
100	10	40	20.0	10	90	40.0
200	20	50	10.0	20	20	70.0
300	30	60	0.0	90	70	0.0
400	40	70	30.0	80	60	30.0
500	50	80	40.0	70	50	20.0
600	60	90	50.0	60	40	10.0

PYTHON CON PANDAS – limpiar valores

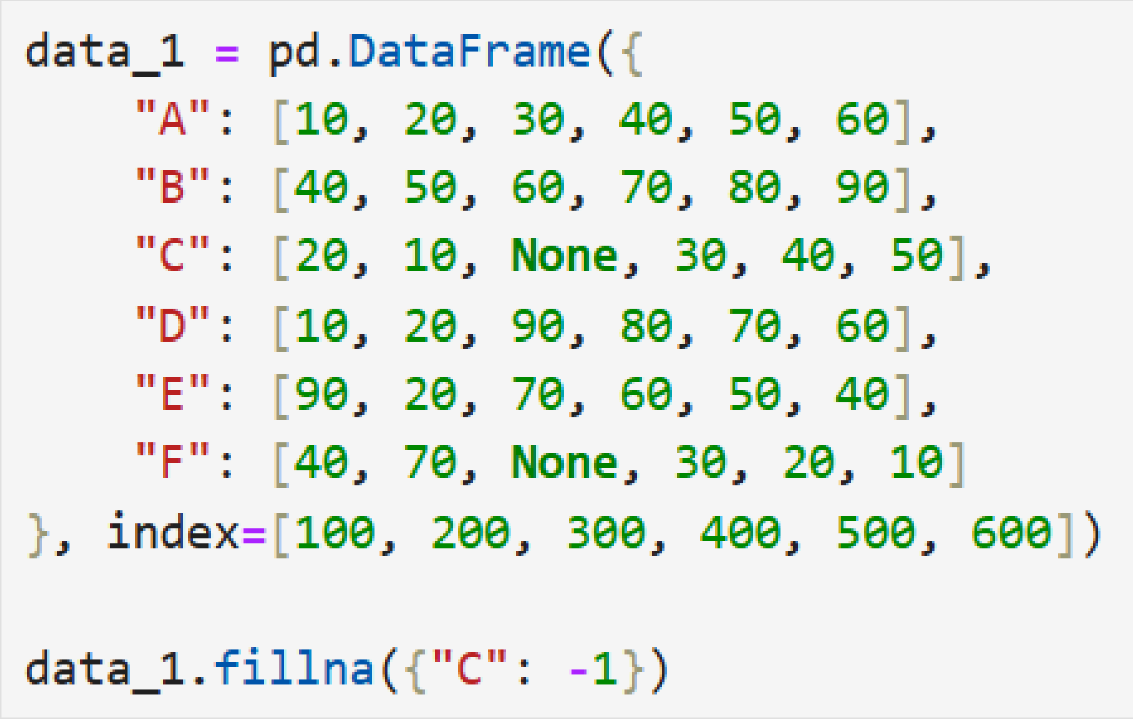


```
data_1 = pd.DataFrame({
    "A": [10, 20, 30, 40, 50, 60],
    "B": [40, 50, 60, 70, 80, 90],
    "C": [20, 10, None, 30, 40, 50],
    "D": [10, 20, 90, 80, 70, 60],
    "E": [90, 20, 70, 60, 50, 40],
    "F": [40, 70, None, 30, 20, 10]
}, index=[100, 200, 300, 400, 500, 600])

data_1.fillna(data_1.mean())
```

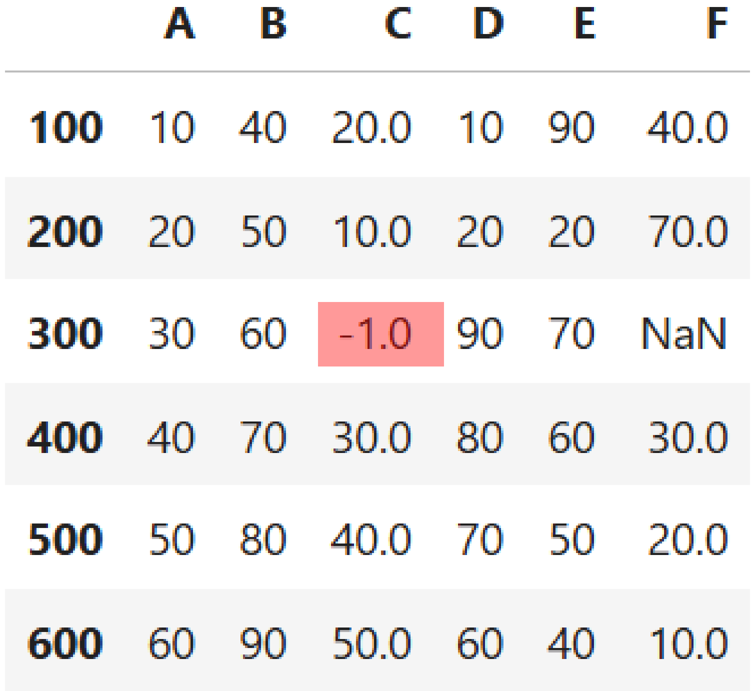


	A	B	C	D	E	F
100	10	40	20.0	10	90	40.0
200	20	50	10.0	20	20	70.0
300	30	60	30.0	90	70	34.0
400	40	70	30.0	80	60	30.0
500	50	80	40.0	70	50	20.0
600	60	90	50.0	60	40	10.0

PYTHON CON PANDAS – limpiar valores

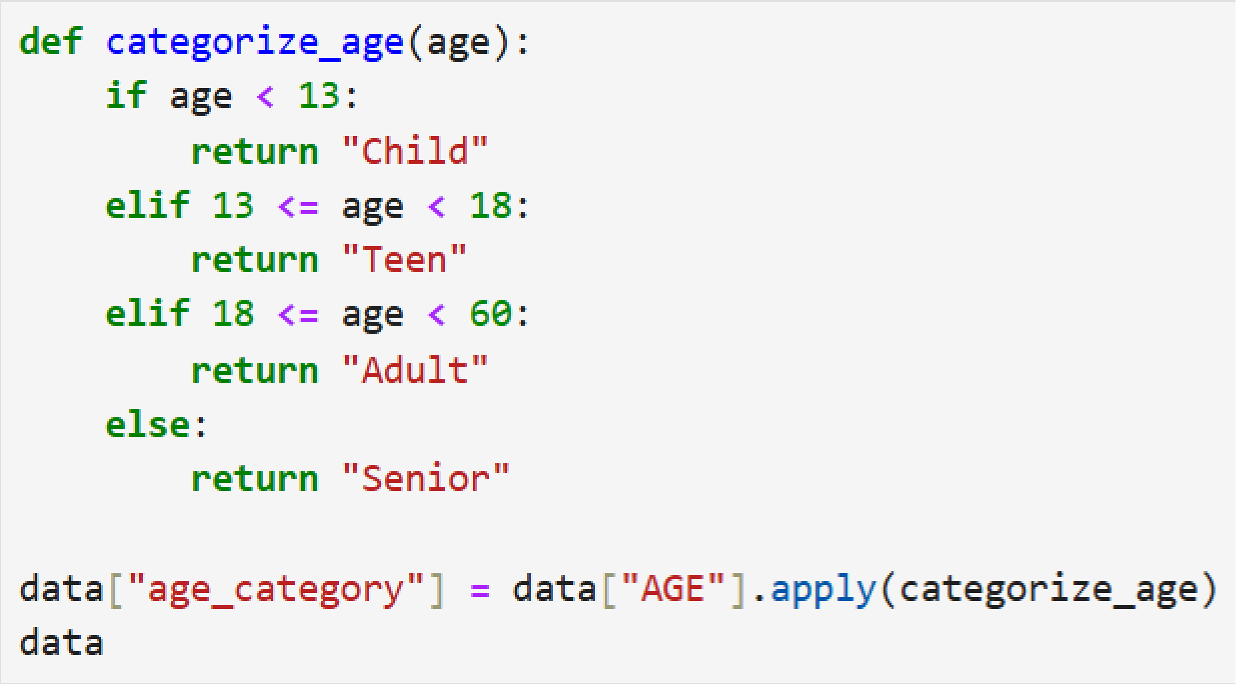


```
data_1 = pd.DataFrame({
    "A": [10, 20, 30, 40, 50, 60],
    "B": [40, 50, 60, 70, 80, 90],
    "C": [20, 10, None, 30, 40, 50],
    "D": [10, 20, 90, 80, 70, 60],
    "E": [90, 20, 70, 60, 50, 40],
    "F": [40, 70, None, 30, 20, 10]
}, index=[100, 200, 300, 400, 500, 600])

data_1.fillna({"C": -1})
```

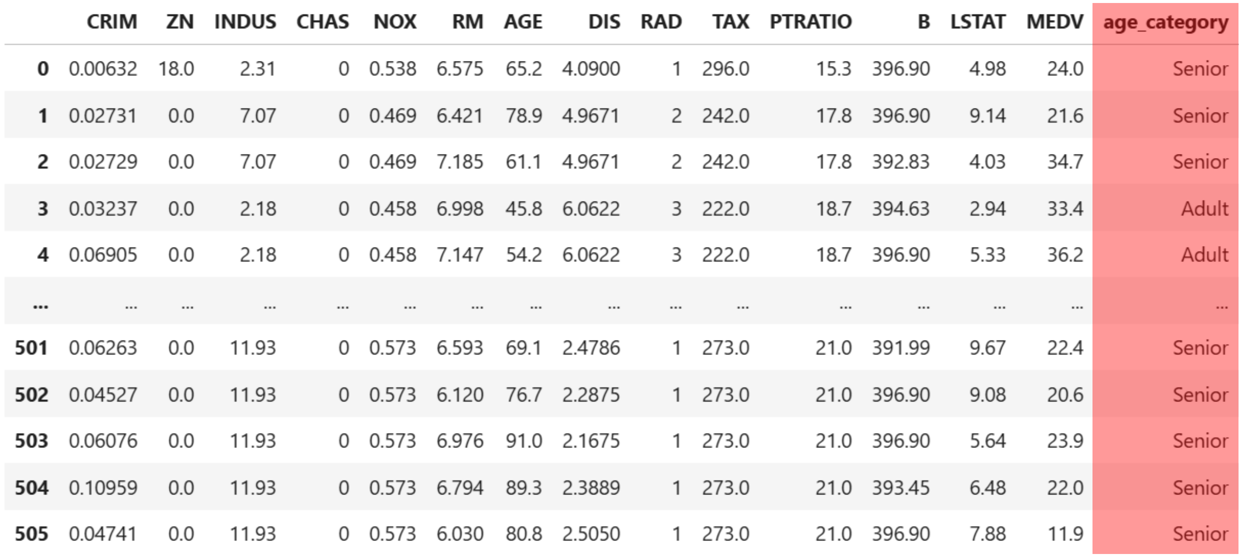


	A	B	C	D	E	F
100	10	40	20.0	10	90	40.0
200	20	50	10.0	20	20	70.0
300	30	60	-1.0	90	70	NaN
400	40	70	30.0	80	60	30.0
500	50	80	40.0	70	50	20.0
600	60	90	50.0	60	40	10.0

PYTHON CON PANDAS – crear columna

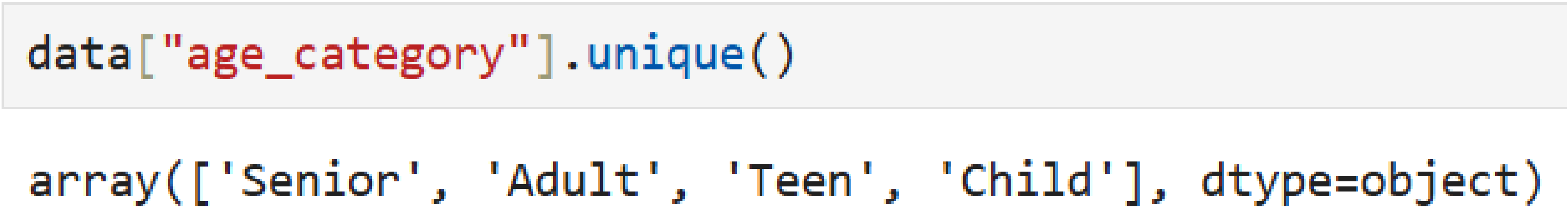


```
def categorize_age(age):
    if age < 13:
        return "Child"
    elif 13 <= age < 18:
        return "Teen"
    elif 18 <= age < 60:
        return "Adult"
    else:
        return "Senior"

data["age_category"] = data["AGE"].apply(categorize_age)
data
```



	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV	age_category
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0	Senior
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6	Senior
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7	Senior
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4	Adult
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	36.2	Adult
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273.0	21.0	391.99	9.67	22.4	Senior
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273.0	21.0	396.90	9.08	20.6	Senior
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273.0	21.0	396.90	5.64	23.9	Senior
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273.0	21.0	393.45	6.48	22.0	Senior
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273.0	21.0	396.90	7.88	11.9	Senior

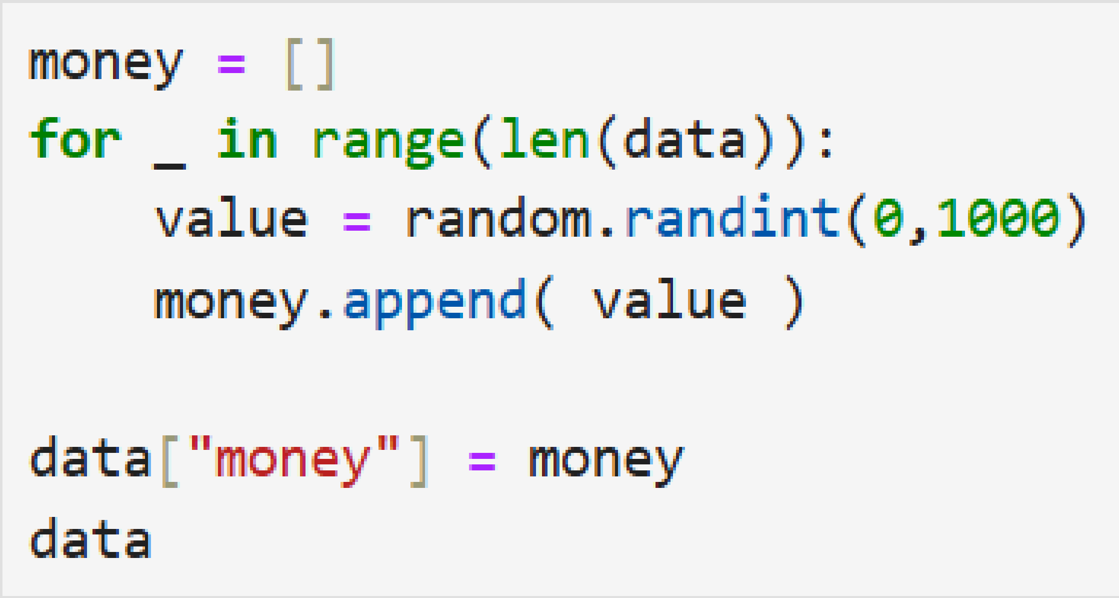
PYTHON CON PANDAS



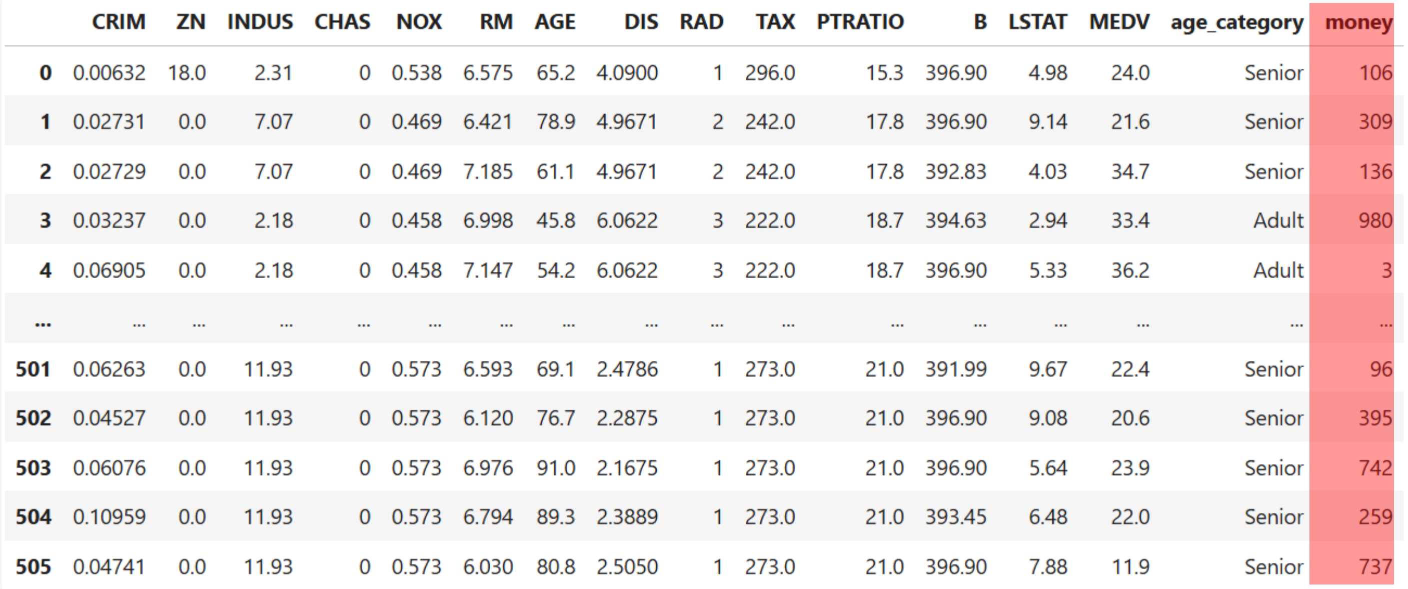
```
data["age_category"].unique()
```

```
array(['Senior', 'Adult', 'Teen', 'Child'], dtype=object)
```

PYTHON CON PANDAS – crear columna

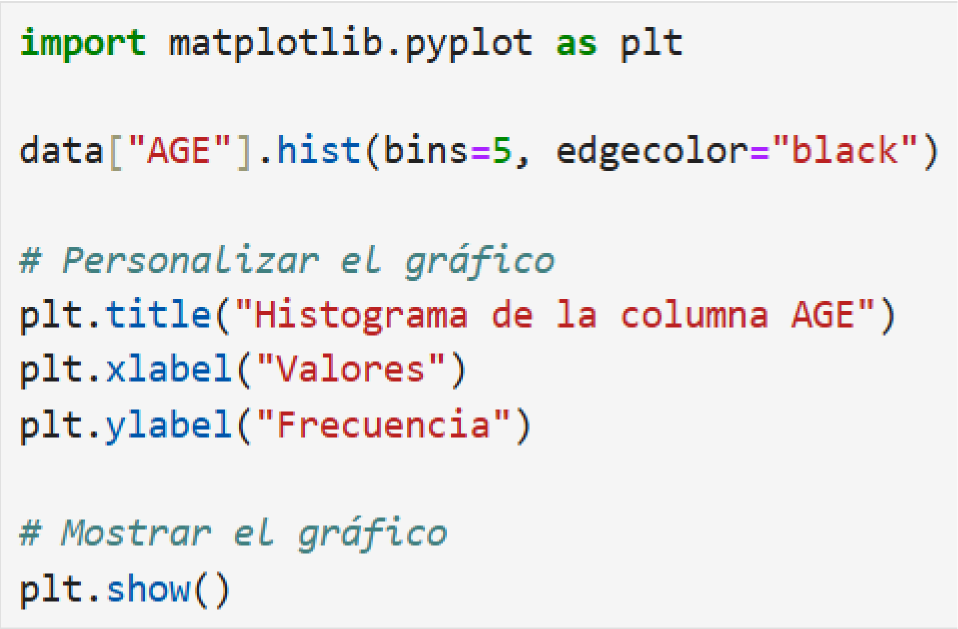


```
money = []
for _ in range(len(data)):
    value = random.randint(0,1000)
    money.append( value )

data["money"] = money
data
```

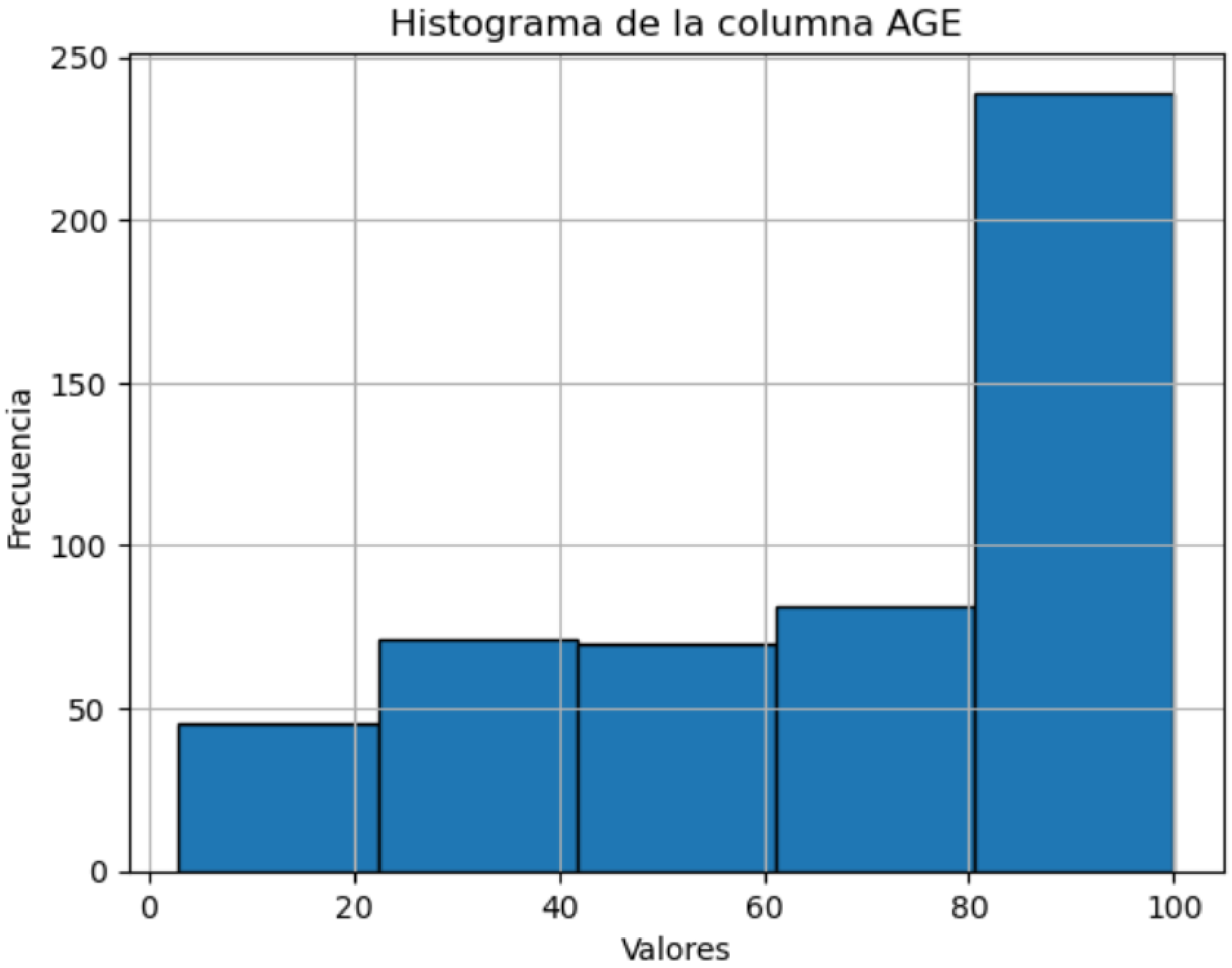


	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV	age_category	money
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0	Senior	106
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6	Senior	309
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7	Senior	136
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4	Adult	980
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	36.2	Adult	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273.0	21.0	391.99	9.67	22.4	Senior	96
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273.0	21.0	396.90	9.08	20.6	Senior	395
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273.0	21.0	396.90	5.64	23.9	Senior	742
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273.0	21.0	393.45	6.48	22.0	Senior	259
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273.0	21.0	396.90	7.88	11.9	Senior	737

PYTHON CON PANDAS – graficar

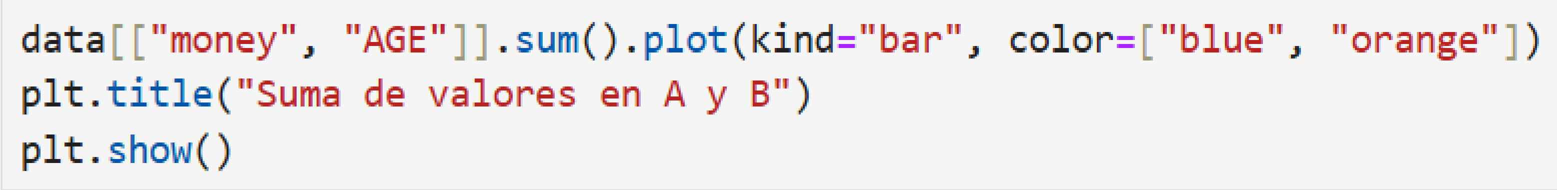


```
import matplotlib.pyplot as plt

data["AGE"].hist(bins=5, edgecolor="black")

# Personalizar el gráfico
plt.title("Histograma de la columna AGE")
plt.xlabel("Valores")
plt.ylabel("Frecuencia")

# Mostrar el gráfico
plt.show()
```

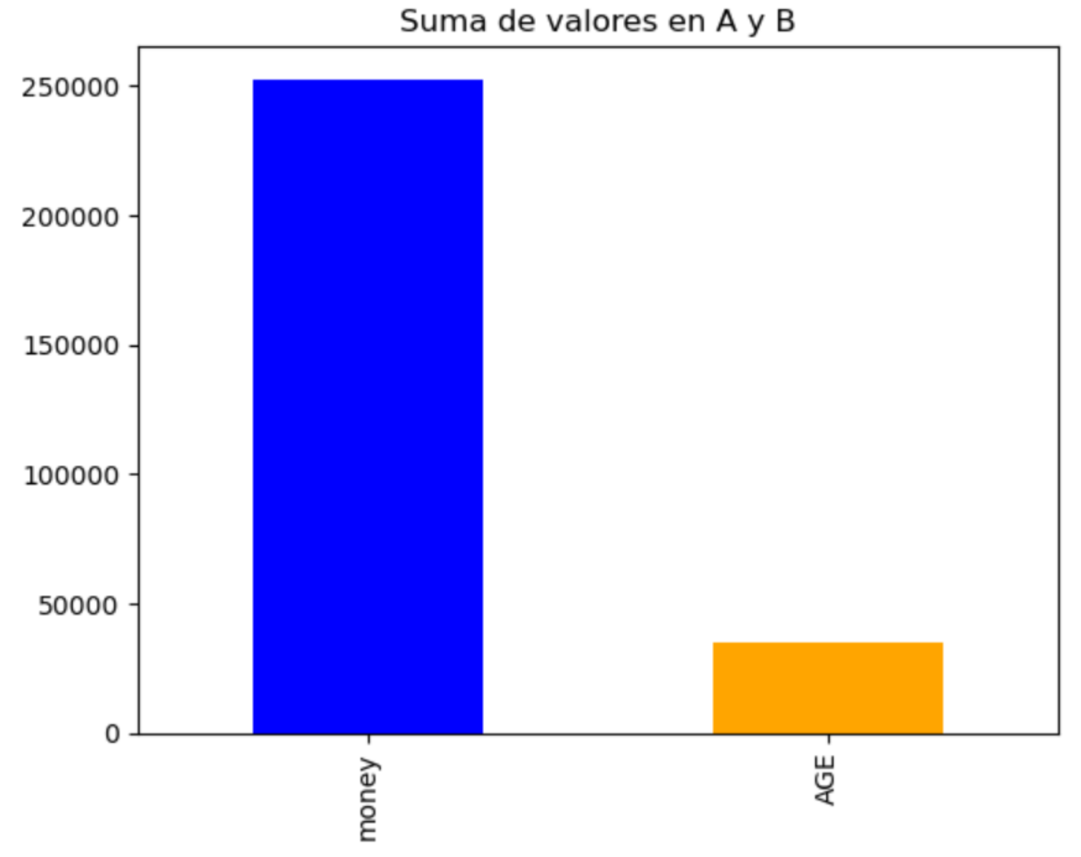


Histograma de la columna AGE

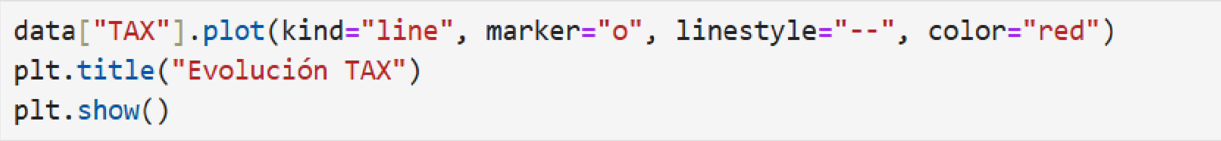
PYTHON CON PANDAS – graficar



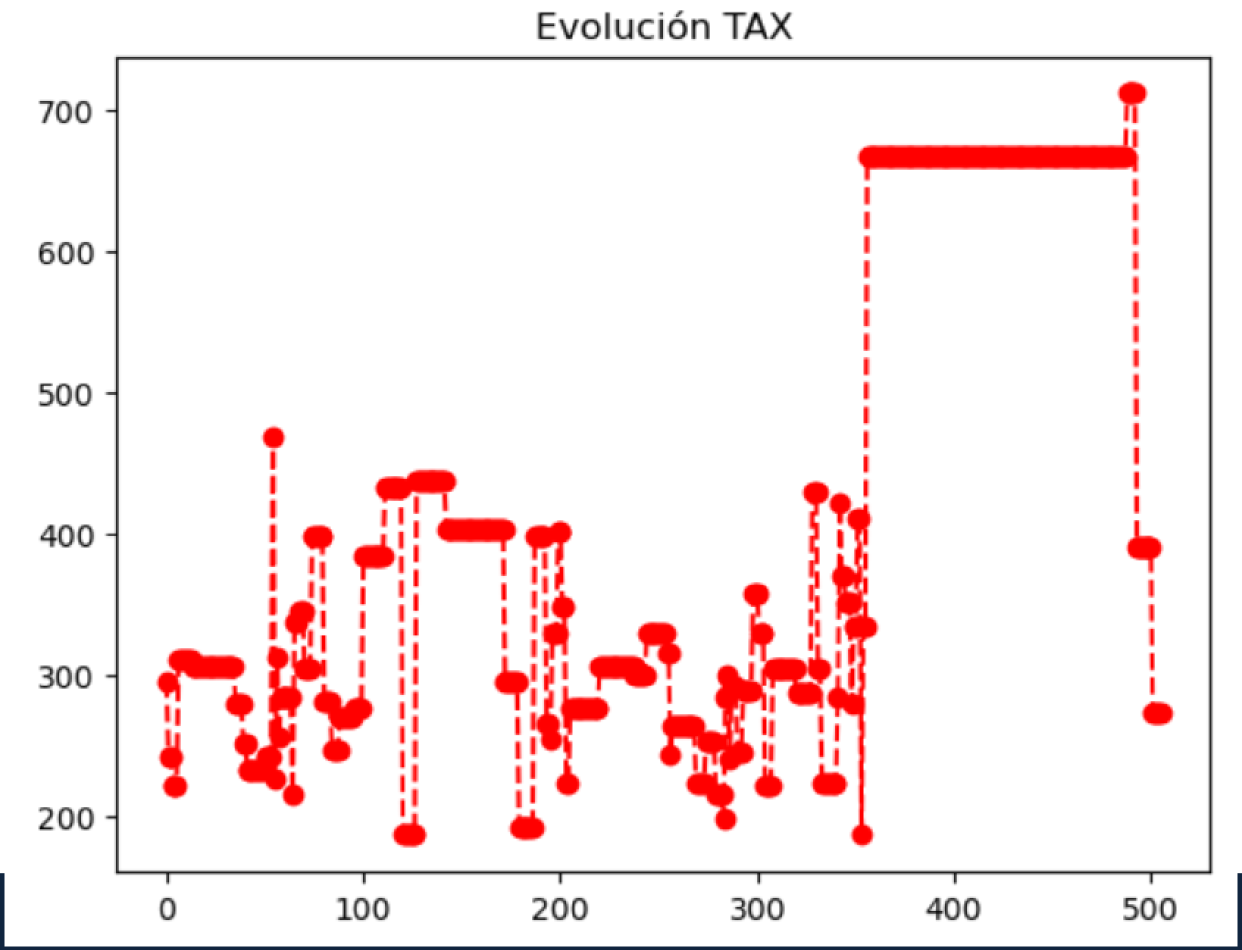
```
data[["money", "AGE"]].sum().plot(kind="bar", color=["blue", "orange"])
plt.title("Suma de valores en A y B")
plt.show()
```



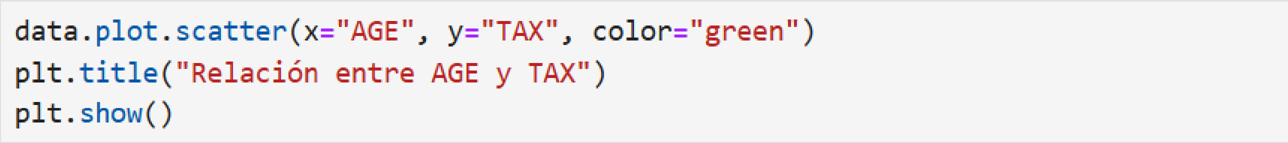
PYTHON CON PANDAS – graficar



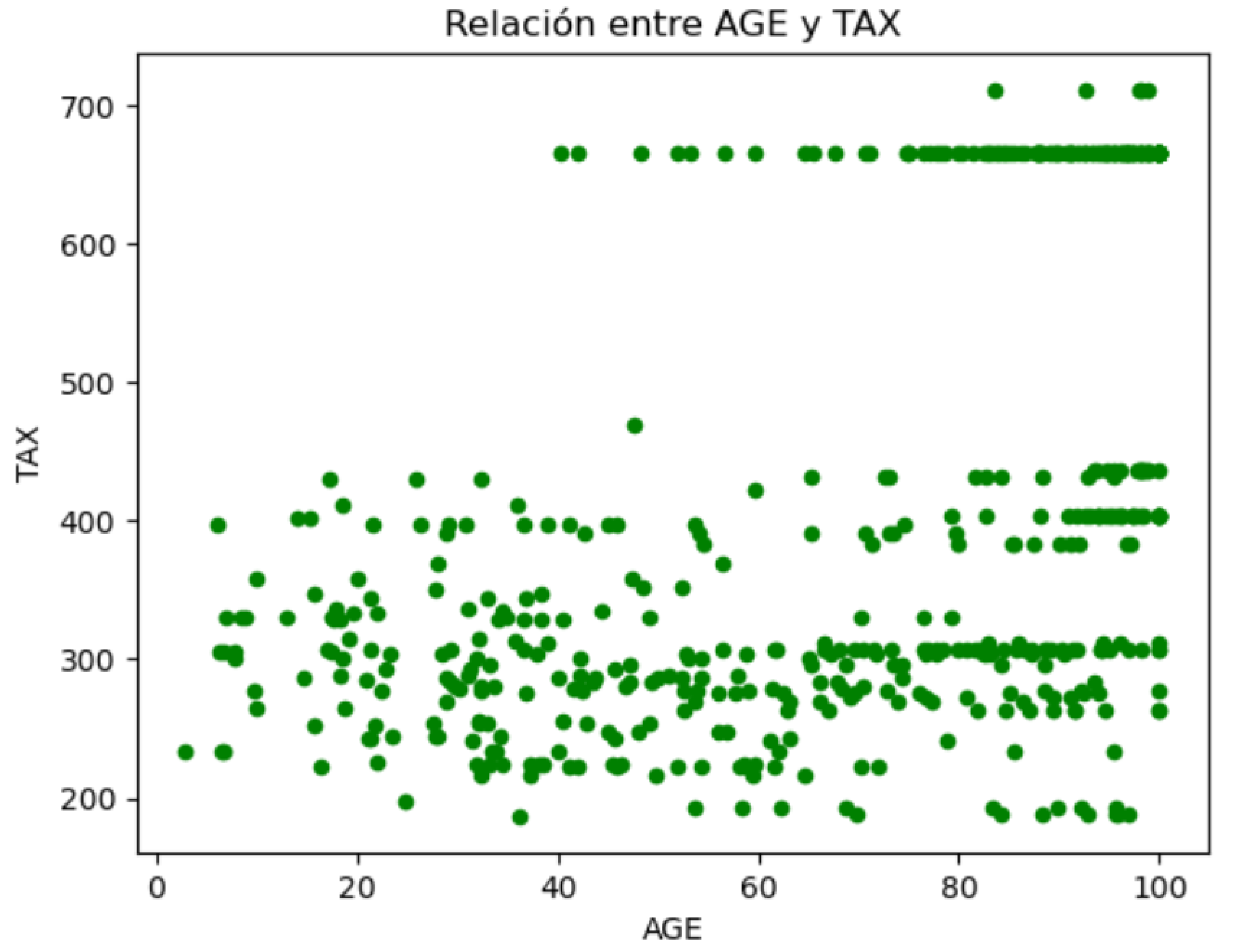
```
data["TAX"].plot(kind="line", marker="o", linestyle="--", color="red")
plt.title("Evolución TAX")
plt.show()
```



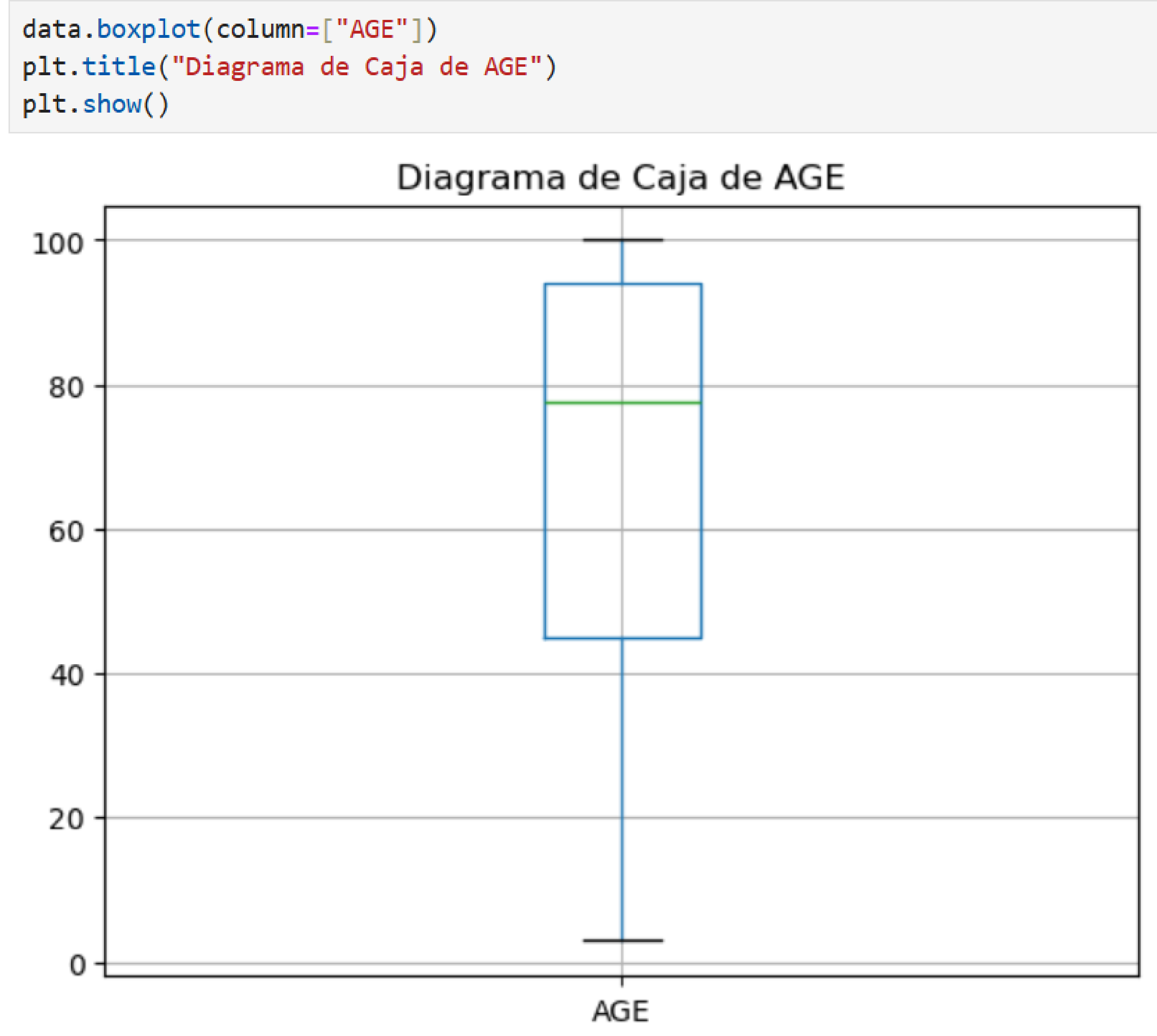
PYTHON CON PANDAS – graficar



```
data.plot.scatter(x="AGE", y="TAX", color="green")
plt.title("Relación entre AGE y TAX")
plt.show()
```

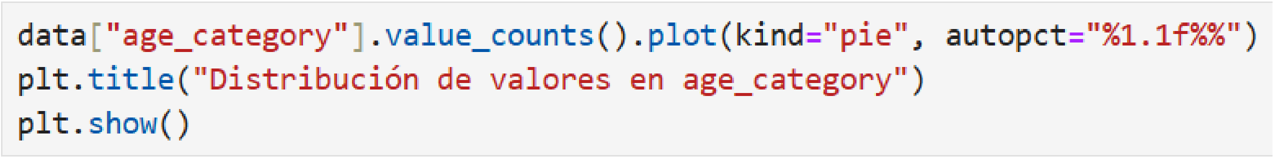


PYTHON CON PANDAS – graficar

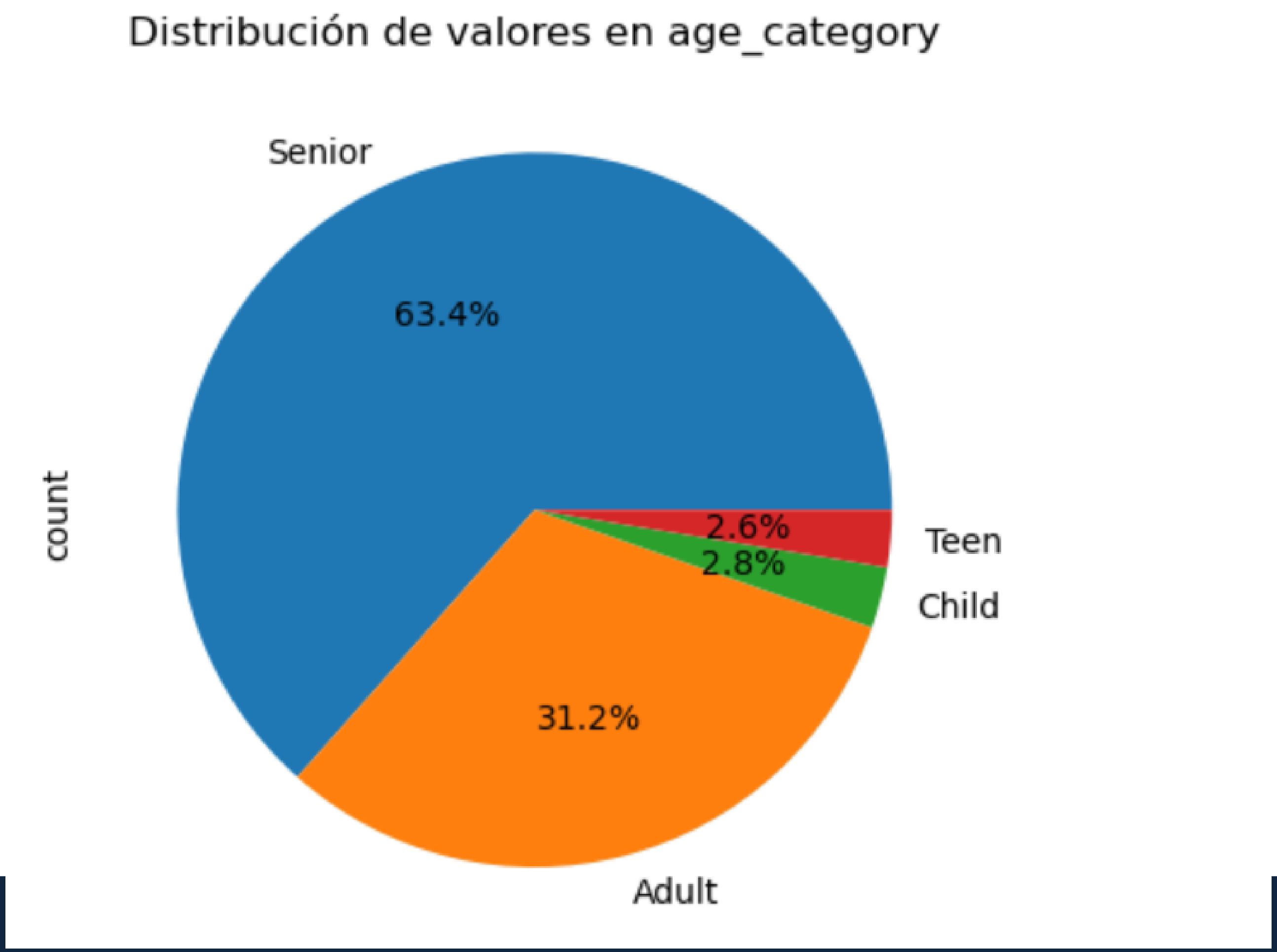


```
data.boxplot(column=["AGE"])
plt.title("Diagrama de Caja de AGE")
plt.show()
```

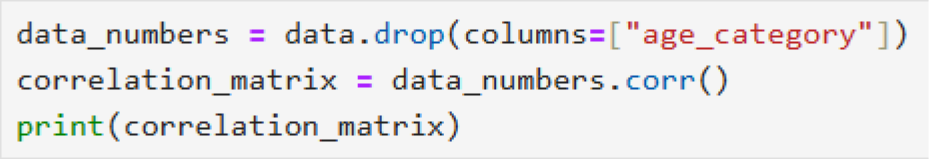
PYTHON CON PANDAS – graficar



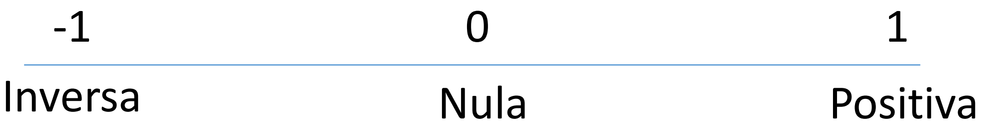
```
data["age_category"].value_counts().plot(kind="pie", autopct="%1.1f%%")
plt.title("Distribución de valores en age_category")
plt.show()
```

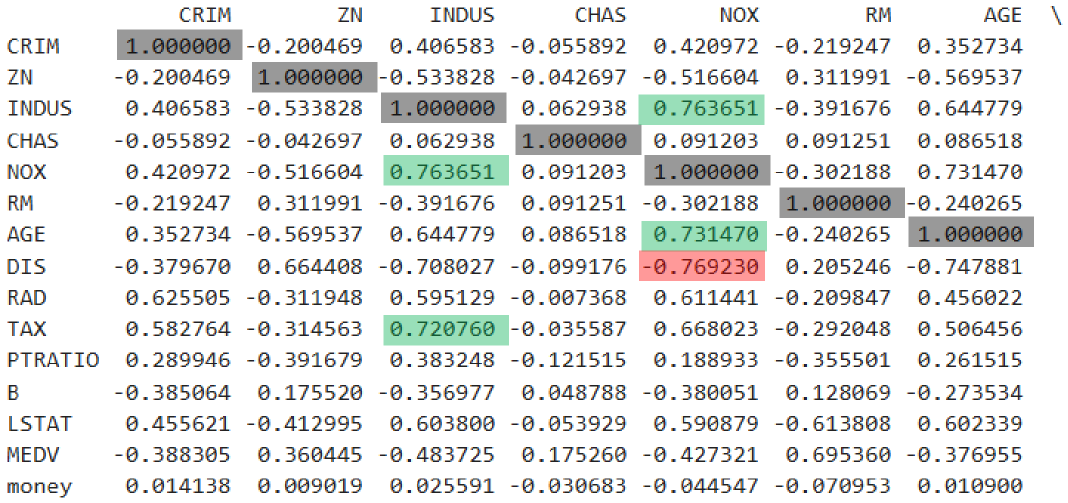


PYTHON CON PANDAS – correlaciones

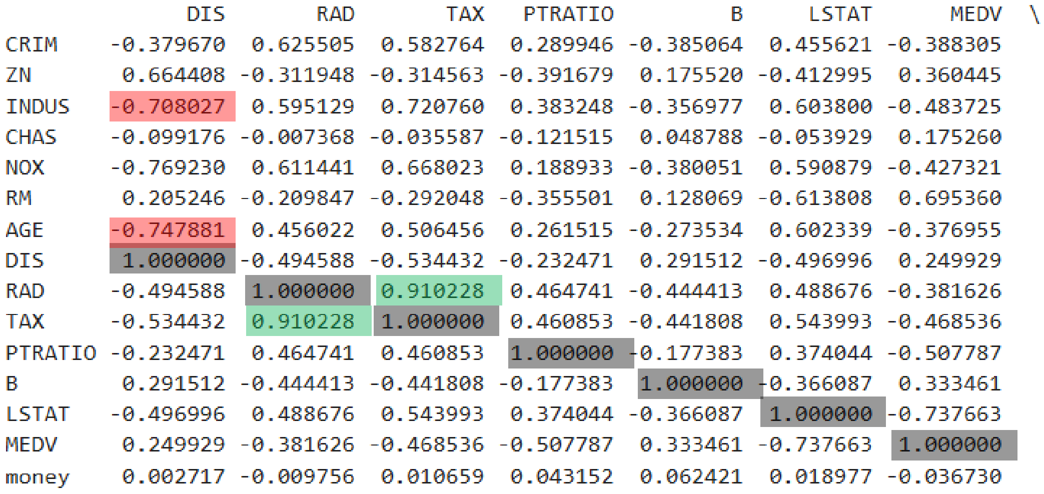


```
data_numbers = data.drop(columns=["age_category"])
correlation_matrix = data_numbers.corr()
print(correlation_matrix)
```

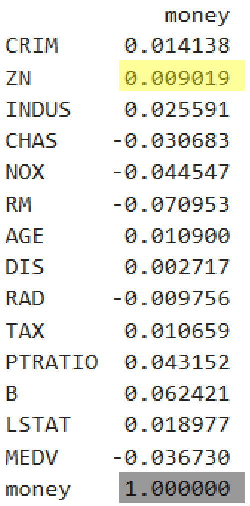




```
            CRIM        ZN     INDUS      CHAS       NOX        RM       AGE  \
CRIM    1.000000 -0.200469  0.406583 -0.055892  0.420972 -0.219247  0.352734
ZN     -0.200469  1.000000 -0.533828 -0.042697 -0.516604  0.311991 -0.569537
INDUS   0.406583 -0.533828  1.000000  0.062938  0.763651 -0.391676  0.644779
CHAS   -0.055892 -0.042697  0.062938  1.000000  0.091203  0.091251  0.086518
NOX     0.420972 -0.516604  0.763651  0.091203  1.000000 -0.302188  0.731470
RM     -0.219247  0.311991 -0.391676  0.091251 -0.302188  1.000000 -0.240265
AGE     0.352734 -0.569537  0.644779  0.086518  0.731470 -0.240265  1.000000
DIS    -0.379670  0.664408 -0.708027 -0.099176 -0.769230  0.205246 -0.747881
RAD     0.625505 -0.311948  0.595129 -0.007368  0.611441 -0.209847  0.456022
TAX     0.582764 -0.314563  0.720760 -0.035587  0.668023 -0.292048  0.506456
PTRATIO 0.289946 -0.391679  0.383248 -0.121515  0.188933 -0.355501  0.261515
B      -0.385064  0.175520 -0.356977  0.048788 -0.380051  0.128069 -0.273534
LSTAT   0.455621 -0.412995  0.603800 -0.053929  0.590879 -0.613808  0.602339
MEDV   -0.388305  0.360445 -0.483725  0.175260 -0.427321  0.695360 -0.376955
money   0.014138  0.009019  0.025591 -0.030683 -0.044547 -0.070953  0.010900
```

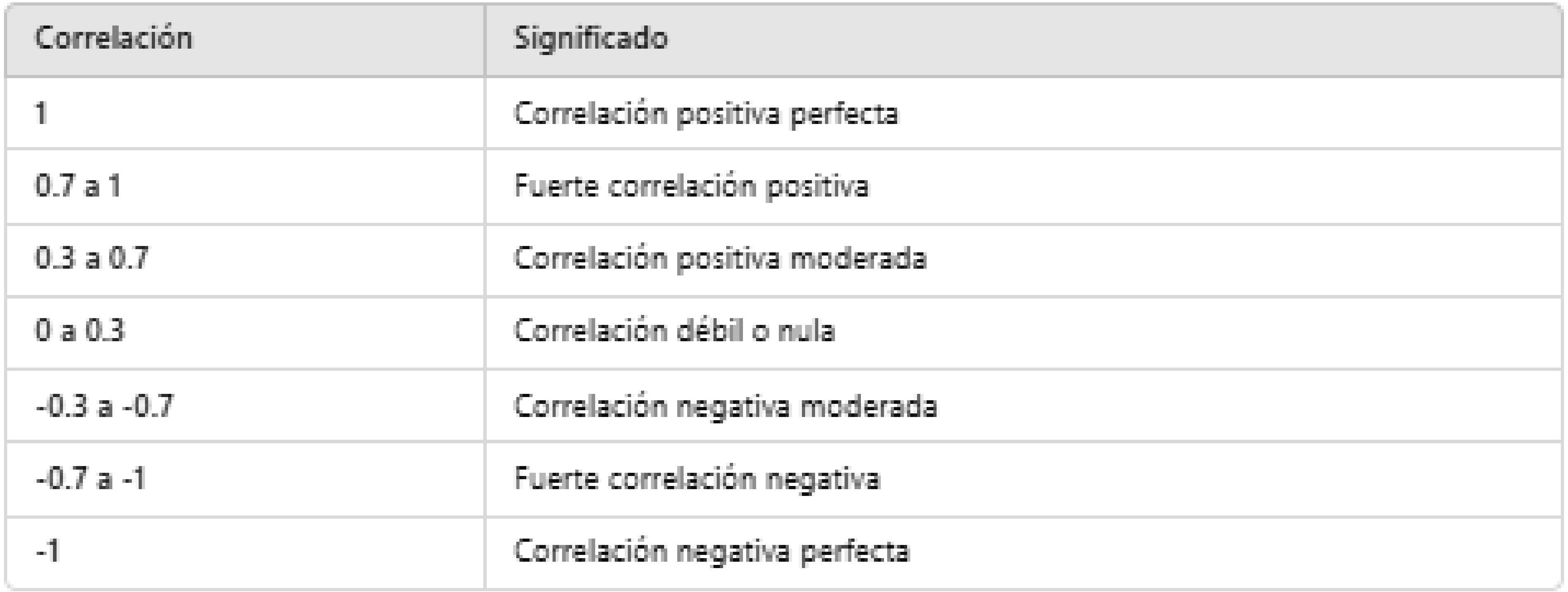


```
              DIS       RAD       TAX   PTRATIO         B     LSTAT      MEDV  \
CRIM    -0.379670  0.625505  0.582764  0.289946 -0.385064  0.455621 -0.388305
ZN       0.664408 -0.311948 -0.314563 -0.391679  0.175520 -0.412995  0.360445
INDUS   -0.708027  0.595129  0.720760  0.383248 -0.356977  0.603800 -0.483725
CHAS    -0.099176 -0.007368 -0.035587 -0.121515  0.048788 -0.053929  0.175260
NOX     -0.769230  0.611441  0.668023  0.188933 -0.380051  0.590879 -0.427321
RM       0.205246 -0.209847 -0.292048 -0.355501  0.128069 -0.613808  0.695360
AGE     -0.747881  0.456022  0.506456  0.261515 -0.273534  0.602339 -0.376955
DIS      1.000000 -0.494588 -0.534432 -0.232471  0.291512 -0.496996  0.249929
RAD     -0.494588  1.000000  0.910228  0.464741 -0.444413  0.488676 -0.381626
TAX     -0.534432  0.910228  1.000000  0.460853 -0.441808  0.543993 -0.468536
PTRATIO -0.232471  0.464741  0.460853  1.000000 -0.177383  0.374044 -0.507787
B        0.291512 -0.444413 -0.441808 -0.177383  1.000000 -0.366087  0.333461
LSTAT   -0.496996  0.488676  0.543993  0.374044 -0.366087  1.000000 -0.737663
MEDV     0.249929 -0.381626 -0.468536 -0.507787  0.333461 -0.737663  1.000000
money    0.002717 -0.009756  0.010659  0.043152  0.062421  0.018977 -0.036730
```



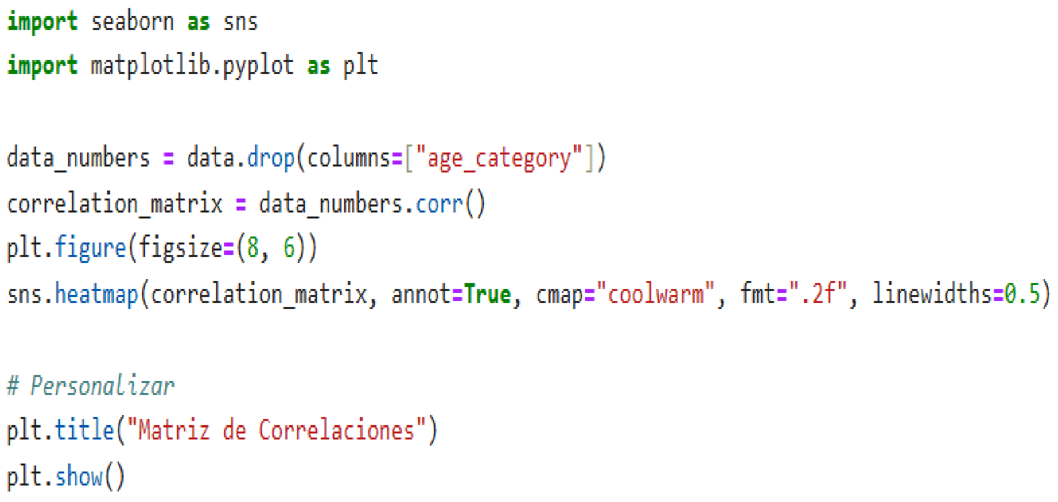
```
            money
CRIM     0.014138
ZN       0.009019
INDUS    0.025591
CHAS    -0.030683
NOX     -0.044547
RM      -0.070953
AGE      0.010900
DIS      0.002717
RAD     -0.009756
TAX      0.010659
PTRATIO  0.043152
B        0.062421
LSTAT    0.018977
MEDV    -0.036730
money    1.000000
```

PYTHON CON PANDAS – correlaciones

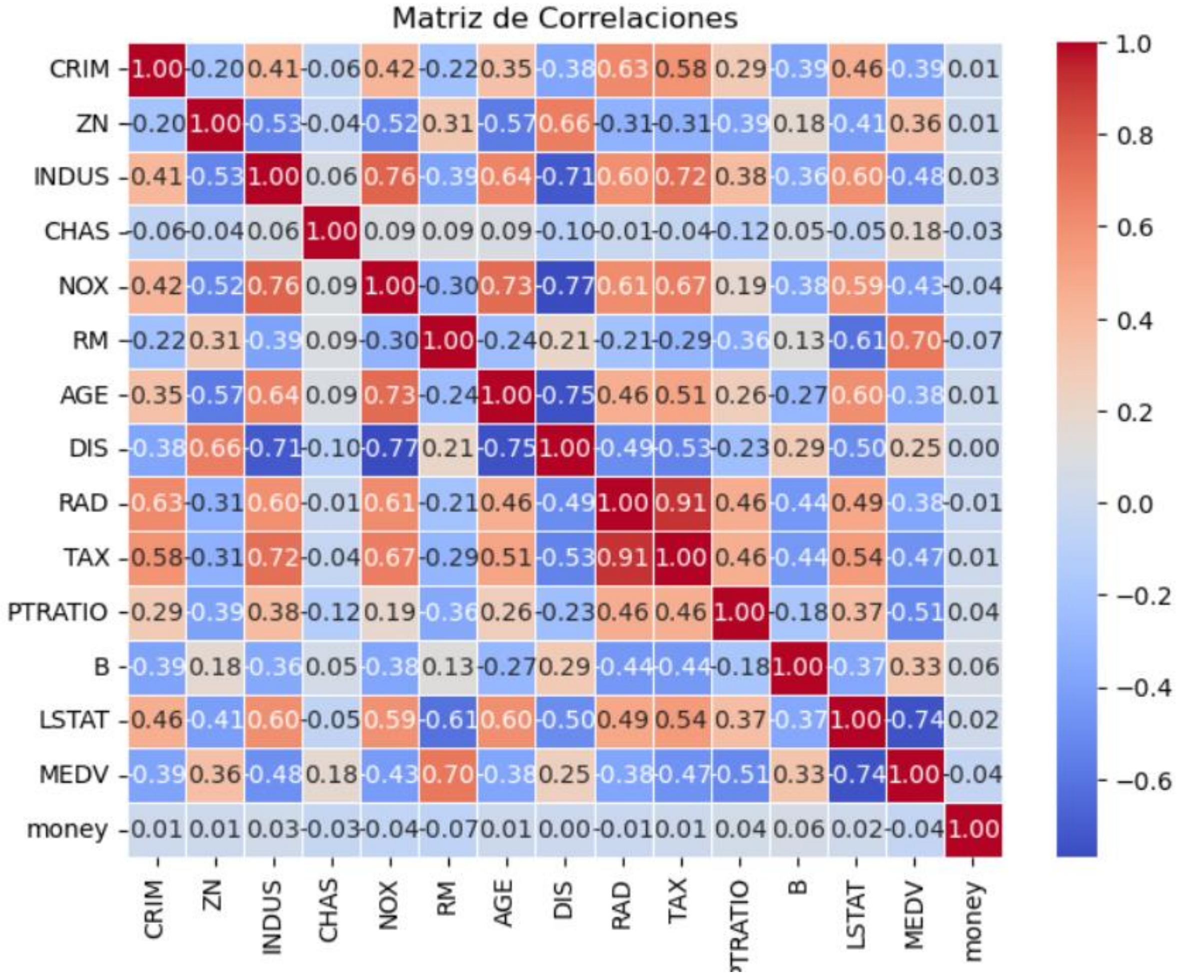


Correlación	Significado
1	Correlación positiva perfecta
0.7 a 1	Fuerte correlación positiva
0.3 a 0.7	Correlación positiva moderada
0 a 0.3	Correlación débil o nula
-0.3 a -0.7	Correlación negativa moderada
-0.7 a -1	Fuerte correlación negativa
-1	Correlación negativa perfecta

PYTHON CON PANDAS – correlaciones



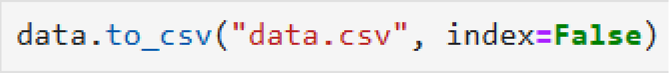
```python
import seaborn as sns
import matplotlib.pyplot as plt

data_numbers = data.drop(columns=["age_category"])
correlation_matrix = data_numbers.corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)

# Personalizar
plt.title("Matriz de Correlaciones")
plt.show()
```



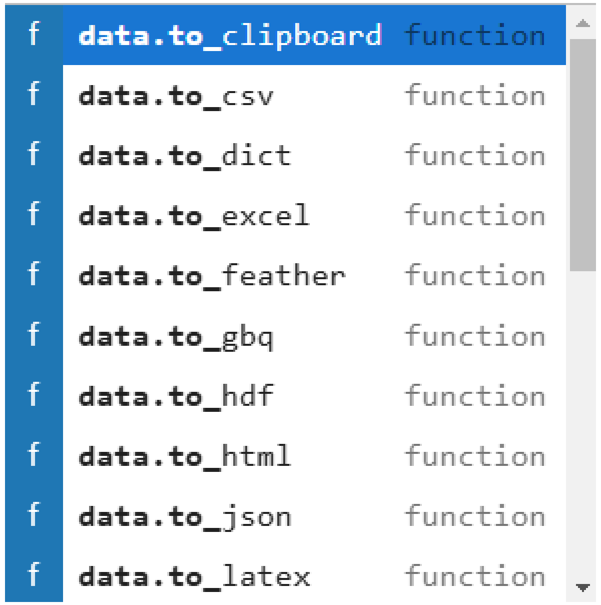
Matriz de Correlaciones

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV	money
CRIM	1.00	-0.20	0.41	-0.06	0.42	-0.22	0.35	-0.38	0.63	0.58	0.29	-0.39	0.46	-0.39	0.01
ZN	-0.20	1.00	-0.53	-0.04	-0.52	0.31	-0.57	0.66	-0.31	-0.31	-0.39	0.18	-0.41	0.36	0.01
INDUS	0.41	-0.53	1.00	0.06	0.76	-0.39	0.64	-0.71	0.60	0.72	0.38	-0.36	0.60	-0.48	0.03
CHAS	-0.06	-0.04	0.06	1.00	0.09	0.09	0.09	-0.10	-0.01	-0.04	-0.12	0.05	-0.05	0.18	-0.03
NOX	0.42	-0.52	0.76	0.09	1.00	-0.30	0.73	-0.77	0.61	0.67	0.19	-0.38	0.59	-0.43	-0.04
RM	-0.22	0.31	-0.39	0.09	-0.30	1.00	-0.24	0.21	-0.21	-0.29	-0.36	0.13	-0.61	0.70	-0.07
AGE	0.35	-0.57	0.64	0.09	0.73	-0.24	1.00	-0.75	0.46	0.51	0.26	-0.27	0.60	-0.38	0.01
DIS	-0.38	0.66	-0.71	-0.10	-0.77	0.21	-0.75	1.00	-0.49	-0.53	-0.23	0.29	-0.50	0.25	0.00
RAD	0.63	-0.31	0.60	-0.01	0.61	-0.21	0.46	-0.49	1.00	0.91	0.46	-0.44	0.49	-0.38	-0.01
TAX	0.58	-0.31	0.72	-0.04	0.67	-0.29	0.51	-0.53	0.91	1.00	0.46	-0.44	0.54	-0.47	0.01
PTRATIO	0.29	-0.39	0.38	-0.12	0.19	-0.36	0.26	-0.23	0.46	0.46	1.00	-0.18	0.37	-0.51	0.04
B	-0.39	0.18	-0.36	0.05	-0.38	0.13	-0.27	0.29	-0.44	-0.44	-0.18	1.00	-0.37	0.33	0.06
LSTAT	0.46	-0.41	0.60	-0.05	0.59	-0.61	0.60	-0.50	0.49	0.54	0.37	-0.37	1.00	-0.74	0.02
MEDV	-0.39	0.36	-0.48	0.18	-0.43	0.70	-0.38	0.25	-0.38	-0.47	-0.51	0.33	-0.74	1.00	-0.04
money	0.01	0.01	0.03	-0.03	-0.04	-0.07	0.01	0.00	-0.01	0.01	0.04	0.06	0.02	-0.04	1.00

PYTHON CON PANDAS – exportar



```
data.to_csv("data.csv", index=False)
```



```
f  data.to_clipboard  function
f  data.to_csv        function
f  data.to_dict       function
f  data.to_excel      function
f  data.to_feather    function
f  data.to_gbq        function
f  data.to_hdf        function
f  data.to_html       function
f  data.to_json       function
f  data.to_latex      function
```