```
import numpy as np
import pandas as pd


# Consideriamo tre Series aventi la stessa chiave:


data1 = {'a': 100, 'b': 200, 'c': 300}
data2 = {'a': 32, 'b': 1.3, 'c': 7}
data3 = {'a': 126, 'b': 412, 'c': 1100.32}

data1
```

```
{'a': 100, 'b': 200, 'c': 300}
```

```
data2
```

```
{'a': 32, 'b': 1.3, 'c': 7}
```

```
data3
```

```
{'a': 126, 'b': 412, 'c': 1100.32}
```

```
# Possiamo unire queste tre Series in un DataFrame!!


df = pd.DataFrame([data1, data2, data3])
df
```

```
     a      b        c
0  100  200.0   300.00
1   32    1.3     7.00
2  126  412.0  1100.32
```

```
# Possiamo anche creare un dataframe a partire da una matrice numpy:


np.random.seed(111)
matrix = np.random.randint(10,100,(3,4))
matrix
```

```
array([[94, 94, 94, 96],
       [29, 51, 76, 92],
       [50, 81, 67, 17]])
```

```
# Diamo ora un nome alle righe e un nome alle colonne


rows_name = ["cane","gatto","topo"]
col_names = ["A","B","C","D"]


df=pd.DataFrame(matrix)
df
```

```
    0   1   2   3
0  94  94  94  96
1  29  51  76  92
2  50  81  67  17
```

```
# Abbiamo così gli indici sull'asse delle y (1) e sull'asse delle x (0).
# Possiamo aggiungere le nostre liste per dare i nomi agli elementi sull'asse delle x e delle y


df=pd.DataFrame(matrix,rows_name,col_names)
df
```

```
        A   B   C   D
cane   94  94  94  96
gatto  29  51  76  92
topo   50  81  67  17
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 3 entries, cane to topo
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   A       3 non-null      int32
 1   B       3 non-null      int32
 2   C       3 non-null      int32
 3   D       3 non-null      int32
dtypes: int32(4)
memory usage: 72.0+ bytes
```

```
# entries indica il numero di righe, columns indica il nome delle
colonne.
# Questa tabella ci dice che tutti i 3 valori delle 4 colonne sono not
null e di valore int32.

# Quindi, ogni colonna presenta 3 valori. Ciascun valore è
corrispondente all'incrocio riga colonna.



# Solitamente i Df non si creano a mano come abbiamo fatto finora, ma
si impoertano fonti esterne (SQL, csv, excel etc)
# che verranno immagazzinate in python sottoforma di Df.



# IMPORTIAMO QUINDI UN FILE DI TIPO CSV. A TITOLO DI ESEMPIO,
IMPORTIAMO IL FILE



tips = pd.read_csv("tips.csv")
tips
```

```
     total_bill   tip     sex smoker   day    time  size
price_per_person  \
0         16.99  1.01  Female     No   Sun  Dinner     2
8.49
1         10.34  1.66    Male     No   Sun  Dinner     3
3.45
2         21.01  3.50    Male     No   Sun  Dinner     3
7.00
3         23.68  3.31    Male     No   Sun  Dinner     2
11.84
4         24.59  3.61  Female     No   Sun  Dinner     4
6.15
..          ...   ...     ...    ...   ...     ...   ...
...
239       29.03  5.92    Male     No   Sat  Dinner     3
9.68
240       27.18  2.00  Female    Yes   Sat  Dinner     2
13.59
241       22.67  2.00    Male    Yes   Sat  Dinner     2
11.34
242       17.82  1.75    Male     No   Sat  Dinner     2
8.91
243       18.78  3.00  Female     No  Thur  Dinner     2
9.39

             Payer Name         CC Number Payment ID
0    Christy Cunningham  3560325168603410    Sun2959
```

```
1        Douglas Tucker  4478071379779230    Sun4608
2        Travis Walters  6011812112971322    Sun4458
3      Nathaniel Harris  4676137647685994    Sun5260
4          Tonya Carter  4832732618637221    Sun2251
..                  ...                ...        ...
239       Michael Avila  5296068606052842    Sat2657
240      Monica Sanders  3506806155565404    Sat1766
241          Keith Wong  6011891618747196    Sat3880
242        Dennis Dixon     4375220550950      Sat17
243     Michelle Hardin  3511451626698139    Thur672

[244 rows x 11 columns]



# Si noti che inserire in python il csv è stato facile, perché una
copia di esso si trova già all'interno
# della cartella in cui si trova questo notebook.
# Ma se il file si fosse trovato in un'altra cartella? Ad esempio, se
il file si fosse trovato in C:\Users\jgian\Desktop\DataScience
Portilla\FilesDelCorso\03-Pandas?
# In tal caso avremmo dovuto inserire il full path



tips = pd.read_csv("C:\\Users\\jgian\\Desktop\\DataScience Portilla\\
FilesDelCorso\\03-Pandas\\tips.csv")
tips

     total_bill   tip     sex smoker   day    time  size
price_per_person  \
0         16.99  1.01  Female     No   Sun  Dinner     2
8.49
1         10.34  1.66    Male     No   Sun  Dinner     3
3.45
2         21.01  3.50    Male     No   Sun  Dinner     3
7.00
3         23.68  3.31    Male     No   Sun  Dinner     2
11.84
4         24.59  3.61  Female     No   Sun  Dinner     4
6.15
..          ...   ...     ...    ...   ...     ...   ...
...
239       29.03  5.92    Male     No   Sat  Dinner     3
9.68
240       27.18  2.00  Female    Yes   Sat  Dinner     2
13.59
241       22.67  2.00    Male    Yes   Sat  Dinner     2
11.34
242       17.82  1.75    Male     No   Sat  Dinner     2
8.91
```

```
243       18.78  3.00  Female     No  Thur  Dinner     2
9.39

              Payer Name         CC Number Payment ID
0    Christy Cunningham  3560325168603410    Sun2959
1        Douglas Tucker  4478071379779230    Sun4608
2        Travis Walters  6011812112971322    Sun4458
3      Nathaniel Harris  4676137647685994    Sun5260
4          Tonya Carter  4832732618637221    Sun2251
..                  ...                ...        ...
239       Michael Avila  5296068606052842    Sat2657
240      Monica Sanders  3506806155565404    Sat1766
241          Keith Wong  6011891618747196    Sat3880
242        Dennis Dixon     4375220550950      Sat17
243     Michelle Hardin  3511451626698139    Thur672

[244 rows x 11 columns]


# Lavoriamo quindi su questo file.


# ESTRAZIONE NOME COLONNE

tips.columns

Index(['total_bill', 'tip', 'sex', 'smoker', 'day', 'time', 'size',
       'price_per_person', 'Payer Name', 'CC Number', 'Payment ID'],
      dtype='object')


# ESTRAZIONE NOME RIGHE

tips.index

RangeIndex(start=0, stop=244, step=1)


# ESTRAZIONE PORZIONE DI UN DF


tips.head(5) # Estrai solo le prime 5 righe

   total_bill   tip     sex smoker  day    time  size
price_per_person  \
0       16.99  1.01  Female     No  Sun  Dinner     2
8.49
1       10.34  1.66    Male     No  Sun  Dinner     3
```

```
3.45
2       21.01  3.50    Male     No  Sun  Dinner     3
7.00
3       23.68  3.31    Male     No  Sun  Dinner     2
11.84
4       24.59  3.61  Female     No  Sun  Dinner     4
6.15

           Payer Name         CC Number Payment ID
0  Christy Cunningham  3560325168603410    Sun2959
1      Douglas Tucker  4478071379779230    Sun4608
2      Travis Walters  6011812112971322    Sun4458
3    Nathaniel Harris  4676137647685994    Sun5260
4        Tonya Carter  4832732618637221    Sun2251



tips.tail(5) # Estrai solo le ultime 5 righe

     total_bill   tip     sex smoker   day    time  size
price_per_person  \
239       29.03  5.92    Male     No   Sat  Dinner     3
9.68
240       27.18  2.00  Female    Yes   Sat  Dinner     2
13.59
241       22.67  2.00    Male    Yes   Sat  Dinner     2
11.34
242       17.82  1.75    Male     No   Sat  Dinner     2
8.91
243       18.78  3.00  Female     No  Thur  Dinner     2
9.39

          Payer Name         CC Number Payment ID
239    Michael Avila  5296068606052842    Sat2657
240   Monica Sanders  3506806155565404    Sat1766
241       Keith Wong  6011891618747196    Sat3880
242     Dennis Dixon     4375220550950      Sat17
243  Michelle Hardin  3511451626698139    Thur672



# INFORMAZIONI GENERALI SUL DF

tips.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   total_bill        244 non-null    float64
```

```
 1   tip               244 non-null    float64
 2   sex               244 non-null    object
 3   smoker            244 non-null    object
 4   day               244 non-null    object
 5   time              244 non-null    object
 6   size              244 non-null    int64
 7   price_per_person  244 non-null    float64
 8   Payer Name        244 non-null    object
 9   CC Number         244 non-null    int64
 10  Payment ID        244 non-null    object
dtypes: float64(3), int64(2), object(6)
memory usage: 21.1+ KB
```

```
# EFFETTUARE STATISTICHE SU VARIABILI NUMERICHE
```

```
tips.describe().round(2)
```

```
       total_bill     tip    size  price_per_person     CC Number
count      244.00  244.00  244.00            244.00  2.440000e+02
mean        19.79    3.00    2.57              7.89  2.563496e+15
std          8.90    1.38    0.95              2.91  2.369340e+15
min          3.07    1.00    1.00              2.88  6.040679e+10
25%         13.35    2.00    2.00              5.80  3.040731e+13
50%         17.80    2.90    2.00              7.26  3.525318e+15
75%         24.13    3.56    3.00              9.39  4.553675e+15
max         50.81   10.00    6.00             20.27  6.596454e+15
```

```
# Ovviamente CC Number è erroneamente salvato come int64 ==> Pandas
calcola le statistiche su di esso.
# Questo non ha senso nella realtà. Il numero di carta di credito deve
essere castato al string.
# Più avanti vediamo come funziona il casting delle variabili.
```

```
# Il nostro describe è più efficiente se trasposto.

tips.describe().transpose()
```

```
                  count          mean           std           min  \
total_bill        244.0  1.978594e+01  8.902412e+00  3.070000e+00
tip               244.0  2.998279e+00  1.383638e+00  1.000000e+00
size              244.0  2.569672e+00  9.510998e-01  1.000000e+00
price_per_person  244.0  7.888197e+00  2.914234e+00  2.880000e+00
CC Number         244.0  2.563496e+15  2.369340e+15  6.040679e+10
```

```
                              25%           50%           75%
max
total_bill         1.334750e+01  1.779500e+01  2.412750e+01
5.081000e+01
tip                2.000000e+00  2.900000e+00  3.562500e+00
1.000000e+01
size               2.000000e+00  2.000000e+00  3.000000e+00
6.000000e+00
price_per_person   5.800000e+00  7.255000e+00  9.390000e+00
2.027000e+01
CC Number          3.040731e+13  3.525318e+15  4.553675e+15
6.596454e+15
```

```
tips.head()
```

```
   total_bill   tip     sex smoker  day    time  size
price_per_person  \
0       16.99  1.01  Female     No  Sun  Dinner     2
8.49
1       10.34  1.66    Male     No  Sun  Dinner     3
3.45
2       21.01  3.50    Male     No  Sun  Dinner     3
7.00
3       23.68  3.31    Male     No  Sun  Dinner     2
11.84
4       24.59  3.61  Female     No  Sun  Dinner     4
6.15

           Payer Name         CC Number Payment ID
0  Christy Cunningham  3560325168603410    Sun2959
1      Douglas Tucker  4478071379779230    Sun4608
2      Travis Walters  6011812112971322    Sun4458
3    Nathaniel Harris  4676137647685994    Sun5260
4        Tonya Carter  4832732618637221    Sun2251
```

```
# LAVORARE CON LE COLONNE


# Per estrarre una colonna dal df si utilizza il metodo loc:

tips["day"]
```

```
0       Sun
1       Sun
2       Sun
```

```
3       Sun
4       Sun
       ...
239     Sat
240     Sat
241     Sat
242     Sat
243    Thur
Name: day, Length: 244, dtype: object
```

```
# Il risultato è una Series!! Quindi, ogni colonna di un Df è
semplicemente una pandas Series, e tutte le colonne
# condividono lo stesso indice
```

```
# Per estrarre un dataframe composto da una colonna, bisogna scrivere
nel seguente modo:
```

```
tips[["day"]]
```

```
      day
0     Sun
1     Sun
2     Sun
3     Sun
4     Sun
..    ...
239   Sat
240   Sat
241   Sat
242   Sat
243  Thur

[244 rows x 1 columns]
```

```
# Se volessimo estrarre più di una colonna, ad esempio le colonne
day,smoker e size:
```

```
tips[["day","smoker","size"]]
```

```
      day smoker  size
0     Sun     No     2
1     Sun     No     3
```

```
2     Sun     No     3
3     Sun     No     2
4     Sun     No     4
..    ...    ...   ...
239   Sat     No     3
240   Sat    Yes     2
241   Sat    Yes     2
242   Sat     No     2
243  Thur     No     2

[244 rows x 3 columns]
```

```
# Oppure
```

```
cols = ["day","smoker","size"]
tips[cols]
```

```
      day smoker  size
0     Sun     No     2
1     Sun     No     3
2     Sun     No     3
3     Sun     No     2
4     Sun     No     4
..    ...    ...   ...
239   Sat     No     3
240   Sat    Yes     2
241   Sat    Yes     2
242   Sat     No     2
243  Thur     No     2

[244 rows x 3 columns]
```

```
# NUOVO CAMPO CALCOLATO

# Possiamo creare una nuova colonna con la seguente semplice sintassi.
Ad esempio, supponiamo di voler sommare due colonne

tips["sizetimesprice"]= tips["size"] +tips["price_per_person"]
tips["sizetimesprice"]
```

```
0      10.49
1       6.45
2      10.00
3      13.84
4      10.15
       ...
```

```
239    12.68
240    15.59
241    13.34
242    10.91
243    11.39
Name: sizetimesprice, Length: 244, dtype: float64
```

```
# Il risultato è una series, e ogni riga è la somma dei corrispondenti
size e price_per_person
```

```
# Se stampiamo di nuovo il Df, noteremo che la nuova colonna
sizetimesprice è stata aggiunta
```

```
tips.columns
```

```
Index(['total_bill', 'tip', 'sex', 'smoker', 'day', 'time', 'size',
       'price_per_person', 'Payer Name', 'CC Number', 'Payment ID',
       'sizetimesprice'],
      dtype='object')
```

```
# CVD
```

```
# Possiamo anche sostituire il valore di una colonna già esistente
```

```
tips.head()
```

```
   total_bill   tip     sex smoker  day    time  size
price_per_person  \
0       16.99  1.01  Female     No  Sun  Dinner     2
8.49
1       10.34  1.66    Male     No  Sun  Dinner     3
3.45
2       21.01  3.50    Male     No  Sun  Dinner     3
7.00
3       23.68  3.31    Male     No  Sun  Dinner     2
11.84
4       24.59  3.61  Female     No  Sun  Dinner     4
6.15

           Payer Name         CC Number Payment ID  sizetimesprice
0  Christy Cunningham  3560325168603410    Sun2959           10.49
1      Douglas Tucker  4478071379779230    Sun4608            6.45
2      Travis Walters  6011812112971322    Sun4458           10.00
```

```
3    Nathaniel Harris  4676137647685994    Sun5260           13.84
4        Tonya Carter  4832732618637221    Sun2251           10.15
```

```
# Aumentando price per person, ad esempio di 4$:

tips["price_per_person"] = tips["sizetimesprice"] +4.0
tips.head()
```

```
   total_bill   tip     sex smoker  day    time  size
price_per_person  \
0       16.99  1.01  Female     No  Sun  Dinner     2
14.49
1       10.34  1.66    Male     No  Sun  Dinner     3
10.45
2       21.01  3.50    Male     No  Sun  Dinner     3
14.00
3       23.68  3.31    Male     No  Sun  Dinner     2
17.84
4       24.59  3.61  Female     No  Sun  Dinner     4
14.15

           Payer Name         CC Number Payment ID  sizetimesprice
0  Christy Cunningham  3560325168603410    Sun2959           10.49
1      Douglas Tucker  4478071379779230    Sun4608            6.45
2      Travis Walters  6011812112971322    Sun4458           10.00
3    Nathaniel Harris  4676137647685994    Sun5260           13.84
4        Tonya Carter  4832732618637221    Sun2251           10.15
```

```
# Supponiamo di voler togliere i decimali dalle nostre colonne
price_per_person e sizetimesprice.
# Dato che pandas è creato su numpy, possiamo utilizzare la funzione
round di numoy
```

```
tips["price_per_person"] =np.round(tips["price_per_person"],0)
tips["sizetimesprice"] =np.round(tips["sizetimesprice"],0)

tips.head()
```

```
   total_bill   tip     sex smoker  day    time  size
price_per_person  \
0       16.99  1.01  Female     No  Sun  Dinner     2
14.0
1       10.34  1.66    Male     No  Sun  Dinner     3
10.0
2       21.01  3.50    Male     No  Sun  Dinner     3
14.0
```

```
3       23.68  3.31    Male     No  Sun  Dinner     2
18.0
4       24.59  3.61  Female     No  Sun  Dinner     4
14.0
```

```
           Payer Name         CC Number Payment ID  sizetimesprice
0  Christy Cunningham  3560325168603410    Sun2959            10.0
1      Douglas Tucker  4478071379779230    Sun4608             6.0
2      Travis Walters  6011812112971322    Sun4458            10.0
3    Nathaniel Harris  4676137647685994    Sun5260            14.0
4        Tonya Carter  4832732618637221    Sun2251            10.0
```

```
# I decimali sono stati eliminati dalle colonne selezionate, ma i valori continuano ad essere float.
```

```
# RIMOZIONE COLONNA

# Supponiamo di voler eliminare la colonna "time". # Per farlo, si utilizza la funzione drop
```

```
tips.drop("time", axis=1)
```

```
     total_bill   tip     sex smoker   day  size  price_per_person  \
0         16.99  1.01  Female     No   Sun     2              14.0
1         10.34  1.66    Male     No   Sun     3              10.0
2         21.01  3.50    Male     No   Sun     3              14.0
3         23.68  3.31    Male     No   Sun     2              18.0
4         24.59  3.61  Female     No   Sun     4              14.0
..          ...   ...     ...    ...   ...   ...               ...
239       29.03  5.92    Male     No   Sat     3              17.0
240       27.18  2.00  Female    Yes   Sat     2              20.0
241       22.67  2.00    Male    Yes   Sat     2              17.0
242       17.82  1.75    Male     No   Sat     2              15.0
243       18.78  3.00  Female     No  Thur     2              15.0
```

```
             Payer Name         CC Number Payment ID  sizetimesprice
0    Christy Cunningham  3560325168603410    Sun2959            10.0
1        Douglas Tucker  4478071379779230    Sun4608             6.0
2        Travis Walters  6011812112971322    Sun4458            10.0
3      Nathaniel Harris  4676137647685994    Sun5260            14.0
4          Tonya Carter  4832732618637221    Sun2251            10.0
..                  ...                ...        ...             ...
239       Michael Avila  5296068606052842    Sat2657            13.0
240      Monica Sanders  3506806155565404    Sat1766            16.0
241          Keith Wong  6011891618747196    Sat3880            13.0
242        Dennis Dixon     4375220550950      Sat17            11.0
```

```
243     Michelle Hardin  3511451626698139    Thur672             11.0

[244 rows x 11 columns]



# Cosa significa axis=1? Semplice, significa che la funzione applicata
al df ha a che fare con le colonne.
# Perché? Immaginiamo un asse cartesiano. Con x=0 si intende l'asse
delle ascisse, e quindi si scorre lungo le colonne,
# con x=1 si intende l'asse delle ordinate, e quindi si scorre lungo
le righe. Se vogliamo eliminare una colonna, dovremmo
# eliminare tutti i volori della colonna stessa, e quindi dobbiamo
scorrere lungo l'asse delle y, ossia lungo le righe.
# Per questo motivo, si pone x=1. Più avanti vederemo molti esempi per
i quali bisogna scegliere se applicare una funzione
# alle colonne (x=1), oppure alle righe (x=0). Per riepilogare: Se
vogliamo scorrere lungo le righe utilizzeremo axis=1, se volessimo
# scorrere lungo le colonne avremo axis=0. Se scorriamo lungo le righe
applicheremo una funzione alle colonne.
# Se volessimo scurrere lungo le colonne applicheremo una funzione
alle righe



# Abbiamo quindi eliminato la colonna time!! Ristampiamo il nostro Df



tips.head()

   total_bill   tip     sex smoker  day    time  size
price_per_person  \
0       16.99  1.01  Female     No  Sun  Dinner     2
14.0
1       10.34  1.66    Male     No  Sun  Dinner     3
10.0
2       21.01  3.50    Male     No  Sun  Dinner     3
14.0
3       23.68  3.31    Male     No  Sun  Dinner     2
18.0
4       24.59  3.61  Female     No  Sun  Dinner     4
14.0

           Payer Name         CC Number Payment ID  sizetimesprice
0  Christy Cunningham  3560325168603410    Sun2959            10.0
1      Douglas Tucker  4478071379779230    Sun4608             6.0
2      Travis Walters  6011812112971322    Sun4458            10.0
3    Nathaniel Harris  4676137647685994    Sun5260            14.0
4        Tonya Carter  4832732618637221    Sun2251            10.0
```

```
# Non abbiamo eliminato un tubo. La colonna time è ancora presente nel
Df. Come facciamo ad eliminarla
# definitivamente? Si utilizza un parametro chiamato inplace: tale
parametro si inserisce nella nostra funzione drop.



tips.drop("time", axis=1, inplace=True)

tips.head()
```

```
   total_bill   tip     sex smoker  day  size  price_per_person  \
0       16.99  1.01  Female     No  Sun     2              14.0
1       10.34  1.66    Male     No  Sun     3              10.0
2       21.01  3.50    Male     No  Sun     3              14.0
3       23.68  3.31    Male     No  Sun     2              18.0
4       24.59  3.61  Female     No  Sun     4              14.0

           Payer Name         CC Number Payment ID  sizetimesprice
0  Christy Cunningham  3560325168603410    Sun2959            10.0
1      Douglas Tucker  4478071379779230    Sun4608             6.0
2      Travis Walters  6011812112971322    Sun4458            10.0
3    Nathaniel Harris  4676137647685994    Sun5260            14.0
4        Tonya Carter  4832732618637221    Sun2251            10.0
```

```
# La colonna time è andata a farsi benedire!!



# Il parametro inplace è tuttavia poco utilizzato. Quello che conviene
fare è inserire il valore
# di ritorno della funzione drop come indice del nostro df tips.
Vediamo come fare. Per prima cosa riapriamo il
# file contenente il nostro Df.

tips = pd.read_csv("tips.csv")
tips.head()
```

```
   total_bill   tip     sex smoker  day    time  size
price_per_person  \
0       16.99  1.01  Female     No  Sun  Dinner     2
8.49
1       10.34  1.66    Male     No  Sun  Dinner     3
3.45
2       21.01  3.50    Male     No  Sun  Dinner     3
7.00
3       23.68  3.31    Male     No  Sun  Dinner     2
11.84
4       24.59  3.61  Female     No  Sun  Dinner     4
6.15
```

```
            Payer Name         CC Number Payment ID
0  Christy Cunningham  3560325168603410    Sun2959
1      Douglas Tucker  4478071379779230    Sun4608
2      Travis Walters  6011812112971322    Sun4458
3    Nathaniel Harris  4676137647685994    Sun5260
4        Tonya Carter  4832732618637221    Sun2251
```

```
# Eliminiamo quindi la colonna time:
```

```
tips = tips.drop("time", axis=1)
tips.head()
```

```
   total_bill   tip     sex smoker  day  size  price_per_person  \
0       16.99  1.01  Female     No  Sun     2              8.49
1       10.34  1.66    Male     No  Sun     3              3.45
2       21.01  3.50    Male     No  Sun     3              7.00
3       23.68  3.31    Male     No  Sun     2             11.84
4       24.59  3.61  Female     No  Sun     4              6.15
```

```
            Payer Name         CC Number Payment ID
0  Christy Cunningham  3560325168603410    Sun2959
1      Douglas Tucker  4478071379779230    Sun4608
2      Travis Walters  6011812112971322    Sun4458
3    Nathaniel Harris  4676137647685994    Sun5260
4        Tonya Carter  4832732618637221    Sun2251
```

```
# CVD
```

```
# Per concludere la nostra trattazione sulle colonne, vediamo un modo
più semplice per spiegare
# come mai l'asse delle colonne si misura con axis=1 e l'asse delle
righe con axis=0.
# Prendiamo in considerazione la funzione shape.
```

```
tips.shape
```

```
(244, 10)
```

```
# Abbiamo 244 righe e 10 colonne. Se volessimo estrarre solo le righe:
```

```
tips.shape[0]   # axis=0
```

```
244
```

```
#  Se volessimo estrarre solo le colonne:
```

```
tips.shape[1]   # axis = 1
```

```
10
```

```
# LAVORARE CON LE RIGHE
```

```
# Riprendiamo in considerazione il nostro Df
```

```
tips = pd.read_csv("tips.csv")
tips.head()
```

```
   total_bill   tip     sex smoker  day    time  size
price_per_person  \
0       16.99  1.01  Female     No  Sun  Dinner     2
8.49
1       10.34  1.66    Male     No  Sun  Dinner     3
3.45
2       21.01  3.50    Male     No  Sun  Dinner     3
7.00
3       23.68  3.31    Male     No  Sun  Dinner     2
11.84
4       24.59  3.61  Female     No  Sun  Dinner     4
6.15

           Payer Name         CC Number Payment ID
0  Christy Cunningham  3560325168603410    Sun2959
1      Douglas Tucker  4478071379779230    Sun4608
2      Travis Walters  6011812112971322    Sun4458
3    Nathaniel Harris  4676137647685994    Sun5260
4        Tonya Carter  4832732618637221    Sun2251
```

```
# Per estrarre il nome delle righe si utilizza index
```

```
tips.index
```

```
RangeIndex(start=0, stop=244, step=1)
```

```
# Abbiamo quindi un range di indici da 0 a n-1, ossia, nel nostro caso, a 244. Tali indici NON
# possono essere ripetuti

# Di solito si lavora con gli indici quando applichiamo modelli di machine learning.
# Negli altri casi, si utilizza una colonna del Df come indice.


# Nel nostro esempio, utilizzaremo Payment ID come indice

tips.set_index("Payment ID")
```

```
            total_bill   tip     sex smoker   day    time  size  \
Payment ID
Sun2959          16.99  1.01  Female     No   Sun  Dinner     2
Sun4608          10.34  1.66    Male     No   Sun  Dinner     3
Sun4458          21.01  3.50    Male     No   Sun  Dinner     3
Sun5260          23.68  3.31    Male     No   Sun  Dinner     2
Sun2251          24.59  3.61  Female     No   Sun  Dinner     4
...                ...   ...     ...    ...   ...     ...   ...
Sat2657          29.03  5.92    Male     No   Sat  Dinner     3
Sat1766          27.18  2.00  Female    Yes   Sat  Dinner     2
Sat3880          22.67  2.00    Male    Yes   Sat  Dinner     2
Sat17            17.82  1.75    Male     No   Sat  Dinner     2
Thur672          18.78  3.00  Female     No  Thur  Dinner     2

            price_per_person          Payer Name         CC Number
Payment ID
Sun2959                 8.49  Christy Cunningham  3560325168603410
Sun4608                 3.45      Douglas Tucker  4478071379779230
Sun4458                 7.00      Travis Walters  6011812112971322
Sun5260                11.84    Nathaniel Harris  4676137647685994
Sun2251                 6.15        Tonya Carter  4832732618637221
...                      ...                 ...               ...
Sat2657                 9.68       Michael Avila  5296068606052842
Sat1766                13.59      Monica Sanders  3506806155565404
Sat3880                11.34          Keith Wong  6011891618747196
Sat17                   8.91        Dennis Dixon     4375220550950
Thur672                 9.39     Michelle Hardin  3511451626698139

[244 rows x 10 columns]
```

```
tips.head()
```

```
   total_bill   tip     sex smoker  day    time  size
price_per_person  \
0       16.99  1.01  Female     No  Sun  Dinner     2
```

```
8.49
1       10.34  1.66    Male     No  Sun  Dinner     3
3.45
2       21.01  3.50    Male     No  Sun  Dinner     3
7.00
3       23.68  3.31    Male     No  Sun  Dinner     2
11.84
4       24.59  3.61  Female     No  Sun  Dinner     4
6.15

            Payer Name         CC Number Payment ID
0  Christy Cunningham  3560325168603410    Sun2959
1      Douglas Tucker  4478071379779230    Sun4608
2      Travis Walters  6011812112971322    Sun4458
3    Nathaniel Harris  4676137647685994    Sun5260
4        Tonya Carter  4832732618637221    Sun2251


# Niente... anche in questo caso il Df di partenza non cambia. Quindi:


tips = tips.set_index("Payment ID")

tips.head()

            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun2959          16.99  1.01  Female     No  Sun  Dinner     2
Sun4608          10.34  1.66    Male     No  Sun  Dinner     3
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
Sun2251          24.59  3.61  Female     No  Sun  Dinner     4

            price_per_person          Payer Name         CC Number
Payment ID
Sun2959                 8.49  Christy Cunningham  3560325168603410
Sun4608                 3.45      Douglas Tucker  4478071379779230
Sun4458                 7.00      Travis Walters  6011812112971322
Sun5260                11.84    Nathaniel Harris  4676137647685994
Sun2251                 6.15        Tonya Carter  4832732618637221


# Ora va bene


# Si noti che Payment ID NON è una colonna, ma è il nome dell'indice.
```

```
# Supponiamo ora di voler tornare indietro, ossia di voler cancellare l'indice e farlo diventare
# di nuovo una colonna del nostro Df. Per farlo, si utilizza il metodo reset_index


tips.reset_index()
```

```
    Payment ID  total_bill   tip     sex smoker   day    time  size  \
0      Sun2959       16.99  1.01  Female     No   Sun  Dinner     2
1      Sun4608       10.34  1.66    Male     No   Sun  Dinner     3
2      Sun4458       21.01  3.50    Male     No   Sun  Dinner     3
3      Sun5260       23.68  3.31    Male     No   Sun  Dinner     2
4      Sun2251       24.59  3.61  Female     No   Sun  Dinner     4
..         ...         ...   ...     ...    ...   ...     ...   ...
239    Sat2657       29.03  5.92    Male     No   Sat  Dinner     3
240    Sat1766       27.18  2.00  Female    Yes   Sat  Dinner     2
241    Sat3880       22.67  2.00    Male    Yes   Sat  Dinner     2
242      Sat17       17.82  1.75    Male     No   Sat  Dinner     2
243    Thur672       18.78  3.00  Female     No  Thur  Dinner     2

     price_per_person          Payer Name         CC Number
0                8.49  Christy Cunningham  3560325168603410
1                3.45      Douglas Tucker  4478071379779230
2                7.00      Travis Walters  6011812112971322
3               11.84    Nathaniel Harris  4676137647685994
4                6.15        Tonya Carter  4832732618637221
..                ...                 ...               ...
239              9.68       Michael Avila  5296068606052842
240             13.59      Monica Sanders  3506806155565404
241             11.34          Keith Wong  6011891618747196
242              8.91        Dennis Dixon     4375220550950
243              9.39     Michelle Hardin  3511451626698139

[244 rows x 11 columns]
```

```
tips.head()
```

```
            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun2959          16.99  1.01  Female     No  Sun  Dinner     2
Sun4608          10.34  1.66    Male     No  Sun  Dinner     3
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
Sun2251          24.59  3.61  Female     No  Sun  Dinner     4

            price_per_person          Payer Name         CC Number
Payment ID
Sun2959                 8.49  Christy Cunningham  3560325168603410
Sun4608                 3.45      Douglas Tucker  4478071379779230
```

```
Sun4458                 7.00      Travis Walters  6011812112971322
Sun5260                11.84    Nathaniel Harris  4676137647685994
Sun2251                 6.15        Tonya Carter  4832732618637221
```

```
# Come al solito col cavolo che il Df originale viene modificato.
Quindi:
```

```
tips = tips.reset_index()
tips.head()
```

```
  Payment ID  total_bill   tip     sex smoker  day    time  size  \
0    Sun2959       16.99  1.01  Female     No  Sun  Dinner     2
1    Sun4608       10.34  1.66    Male     No  Sun  Dinner     3
2    Sun4458       21.01  3.50    Male     No  Sun  Dinner     3
3    Sun5260       23.68  3.31    Male     No  Sun  Dinner     2
4    Sun2251       24.59  3.61  Female     No  Sun  Dinner     4

   price_per_person          Payer Name         CC Number
0              8.49  Christy Cunningham  3560325168603410
1              3.45      Douglas Tucker  4478071379779230
2              7.00      Travis Walters  6011812112971322
3             11.84    Nathaniel Harris  4676137647685994
4              6.15        Tonya Carter  4832732618637221
```

```
# Perfetto.
```

```
# Per proseguire la nostra trattazione utilizziamo comunque Payment ID
come indice:
```

```
tips = tips.set_index("Payment ID")
tips.head(8)
```

```
            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun2959          16.99  1.01  Female     No  Sun  Dinner     2
Sun4608          10.34  1.66    Male     No  Sun  Dinner     3
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
Sun2251          24.59  3.61  Female     No  Sun  Dinner     4
Sun9679          25.29  4.71    Male     No  Sun  Dinner     4
Sun5985           8.77  2.00    Male     No  Sun  Dinner     2
Sun8157          26.88  3.12    Male     No  Sun  Dinner     4
```

```
            price_per_person          Payer Name         CC Number
Payment ID
Sun2959                 8.49  Christy Cunningham  3560325168603410
Sun4608                 3.45      Douglas Tucker  4478071379779230
Sun4458                 7.00      Travis Walters  6011812112971322
Sun5260                11.84    Nathaniel Harris  4676137647685994
Sun2251                 6.15         Tonya Carter  4832732618637221
Sun9679                 6.32           Erik Smith   213140353657882
Sun5985                 4.38  Kristopher Johnson  2223727524230344
Sun8157                 6.72          Robert Buck  3514785077705092
```

```
# Supponiamo ora di voler estrarre la terza riga, ossia la riga in
corrispondenza della quale l'indice è Sun4458.
# Abbiamo due modi diversi: Possiamo farlo tramite loc (si estrae la
riga in base al nome dell'indice)
# oppure iloc (si estrae la riga in base alla posizione).

# ESTRAZIONE RIGA TRAMITE loc:

tips.loc["Sun4458"]
```

```
total_bill                    21.01
tip                             3.5
sex                            Male
smoker                           No
day                             Sun
time                         Dinner
size                              3
price_per_person                7.0
Payer Name            Travis Walters
CC Number           6011812112971322
Name: Sun4458, dtype: object
```

```
# ESTRAZIONE RIGA TRAMITE iloc:

tips.iloc[2]
```

```
total_bill                    21.01
tip                             3.5
sex                            Male
smoker                           No
day                             Sun
time                         Dinner
size                              3
price_per_person                7.0
Payer Name            Travis Walters
```

```
CC Number           6011812112971322
Name: Sun4458, dtype: object
```

```
# I risultati coincidono.  CVD
```

```
# Ovviamnete, si noti che la riga estratta è una series
```

```
# ESTRAZIONE RIGHE 2,3,4

# iloc

tips.iloc[2:5]    # L'estremo superiore è escluso
```

```
            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
Sun2251          24.59  3.61  Female     No  Sun  Dinner     4

            price_per_person        Payer Name         CC Number
Payment ID
Sun4458                 7.00    Travis Walters  6011812112971322
Sun5260                11.84  Nathaniel Harris  4676137647685994
Sun2251                 6.15      Tonya Carter  4832732618637221
```

```
tips.loc["Sun4608":"Sun5260"]   # Estremo superiore incluso
```

```
            total_bill   tip   sex smoker  day    time  size  \
Payment ID
Sun4608          10.34  1.66  Male     No  Sun  Dinner     3
Sun4458          21.01  3.50  Male     No  Sun  Dinner     3
Sun5260          23.68  3.31  Male     No  Sun  Dinner     2

            price_per_person        Payer Name         CC Number
Payment ID
Sun4608                 3.45    Douglas Tucker  4478071379779230
Sun4458                 7.00    Travis Walters  6011812112971322
Sun5260                11.84  Nathaniel Harris  4676137647685994
```

```
# POSSIAMO ANCHE ESTRARRE UN SOTTOINSIEME DI RIGHE

# loc
```

```
tips.loc[["Sun4458","Sun9679","Sun8157"]]
```

```
            total_bill   tip   sex smoker  day    time  size  \
Payment ID
Sun4458          21.01  3.50  Male     No  Sun  Dinner     3
Sun9679          25.29  4.71  Male     No  Sun  Dinner     4
Sun8157          26.88  3.12  Male     No  Sun  Dinner     4

            price_per_person      Payer Name         CC Number
Payment ID
Sun4458                 7.00  Travis Walters  6011812112971322
Sun9679                 6.32      Erik Smith   213140353657882
Sun8157                 6.72     Robert Buck  3514785077705092
```

```
# iloc

tips.iloc[[2,4,7]]
```

```
            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
Sun2251          24.59  3.61  Female     No  Sun  Dinner     4
Sun8157          26.88  3.12    Male     No  Sun  Dinner     4

            price_per_person      Payer Name         CC Number
Payment ID
Sun4458                 7.00  Travis Walters  6011812112971322
Sun2251                 6.15    Tonya Carter  4832732618637221
Sun8157                 6.72     Robert Buck  3514785077705092
```

```
tips.shape
```

```
(244, 10)
```

```
# Rimozione di una riga

tips = tips.drop("Sun2251",axis=0)
tips.head(8)
```

```
            total_bill   tip     sex smoker  day    time  size  \
Payment ID
Sun2959          16.99  1.01  Female     No  Sun  Dinner     2
Sun4608          10.34  1.66    Male     No  Sun  Dinner     3
Sun4458          21.01  3.50    Male     No  Sun  Dinner     3
Sun5260          23.68  3.31    Male     No  Sun  Dinner     2
Sun9679          25.29  4.71    Male     No  Sun  Dinner     4
Sun5985           8.77  2.00    Male     No  Sun  Dinner     2
Sun8157          26.88  3.12    Male     No  Sun  Dinner     4
Sun6820          15.04  1.96    Male     No  Sun  Dinner     2

            price_per_person          Payer Name         CC Number
Payment ID
```

```
Sun2959                  8.49  Christy Cunningham  3560325168603410
Sun4608                  3.45      Douglas Tucker  4478071379779230
Sun4458                  7.00      Travis Walters  6011812112971322
Sun5260                 11.84    Nathaniel Harris  4676137647685994
Sun9679                  6.32          Erik Smith   213140353657882
Sun5985                  4.38  Kristopher Johnson  2223727524230344
Sun8157                  6.72         Robert Buck  3514785077705092
Sun6820                  7.52     Joseph Mcdonald  3522866365840377

# Per vedere se effettivamente manca una riga, usiamo la funzione
shape

tips.shape   # Prima le righe erano 244, quindi la riga è stata
eliminata correttamente

(243, 10)

# Possiamo anche inserire una riga alla fine del nostro Df

tips

            total_bill   tip     sex smoker   day    time  size  \
Payment ID
Sun2959          16.99  1.01  Female     No   Sun  Dinner     2
Sun4608          10.34  1.66    Male     No   Sun  Dinner     3
Sun4458          21.01  3.50    Male     No   Sun  Dinner     3
Sun5260          23.68  3.31    Male     No   Sun  Dinner     2
Sun9679          25.29  4.71    Male     No   Sun  Dinner     4
...                ...   ...     ...    ...   ...     ...   ...
Sat2657          29.03  5.92    Male     No   Sat  Dinner     3
Sat1766          27.18  2.00  Female    Yes   Sat  Dinner     2
Sat3880          22.67  2.00    Male    Yes   Sat  Dinner     2
Sat17            17.82  1.75    Male     No   Sat  Dinner     2
Thur672          18.78  3.00  Female     No  Thur  Dinner     2

            price_per_person          Payer Name         CC Number
Payment ID
Sun2959                 8.49  Christy Cunningham  3560325168603410
Sun4608                 3.45      Douglas Tucker  4478071379779230
Sun4458                 7.00      Travis Walters  6011812112971322
Sun5260                11.84    Nathaniel Harris  4676137647685994
Sun9679                 6.32          Erik Smith   213140353657882
...                      ...                 ...              ...
Sat2657                 9.68       Michael Avila  5296068606052842
Sat1766                13.59      Monica Sanders  3506806155565404
Sat3880                11.34          Keith Wong  6011891618747196
Sat17                   8.91        Dennis Dixon     4375220550950
Thur672                 9.39     Michelle Hardin  3511451626698139

[243 rows x 10 columns]
```

```
# Per semplicità, prendiamo una riga presente nel Df


riga = tips.loc["Sun4458"]
riga
```

```
total_bill                      21.01
tip                               3.5
sex                              Male
smoker                             No
day                               Sun
time                           Dinner
size                                3
price_per_person                  7.0
Payer Name              Travis Walters
CC Number            6011812112971322
Name: Sun4458, dtype: object
```

```
# Ed inseriamola in fondo al Df tramite la funzione concat

tips = pd.concat([tips, pd.DataFrame([riga])])
tips
```

```
         total_bill   tip     sex smoker   day    time  size  \
Sun2959       16.99  1.01  Female     No   Sun  Dinner     2
Sun4608       10.34  1.66    Male     No   Sun  Dinner     3
Sun4458       21.01  3.50    Male     No   Sun  Dinner     3
Sun5260       23.68  3.31    Male     No   Sun  Dinner     2
Sun9679       25.29  4.71    Male     No   Sun  Dinner     4
...             ...   ...     ...    ...   ...     ...   ...
Sat3880       22.67  2.00    Male    Yes   Sat  Dinner     2
Sat17         17.82  1.75    Male     No   Sat  Dinner     2
Thur672       18.78  3.00  Female     No  Thur  Dinner     2
Sun4458       21.01  3.50    Male     No   Sun  Dinner     3
Sun4458       21.01  3.50    Male     No   Sun  Dinner     3

         price_per_person          Payer Name         CC Number
Sun2959              8.49  Christy Cunningham  3560325168603410
Sun4608              3.45      Douglas Tucker  4478071379779230
Sun4458              7.00      Travis Walters  6011812112971322
Sun5260             11.84    Nathaniel Harris  4676137647685994
Sun9679              6.32          Erik Smith   213140353657882
...                   ...                 ...               ...
Sat3880             11.34         Keith Wong  6011891618747196
Sat17                8.91        Dennis Dixon     4375220550950
Thur672              9.39     Michelle Hardin  3511451626698139
```

```
Sun4458              7.00      Travis Walters  6011812112971322
Sun4458              7.00      Travis Walters  6011812112971322

[245 rows x 10 columns]
```

```
# Ovviamente quanto fatto è solo a titolo di esempio:  Nella reatà non
ha alcun senso. abbiamo infatti duplicato una riga ==> Il
# nome della riga è uguale ad uno esistente ==> I nomi delle righe
devono essere UNIVOCI.
```