

Librería de análisis de datos: Pandas



Pandas - Guía de referencia

Importar la librería

```
import pandas as pd
```

Series

Índice Valores

Las **Series** son una estructura de datos de 1 dimensión con índice asociado basado en etiquetas.

```
s = pd.Series(data, index, dtype, name, ...)
```

index= Valores tomados como índice de la serie.

dtype= Tipo de dato de la serie.

name= Nombre de la serie.

data= Contenido de la serie creada.

Con listas

```
pd.Series([1, 2, 3], index = ['a', 'b', 'c'])
```

Con diccionarios

```
pd.Series({'a': 1, 'b': 2, 'c': 3})
```

	a	b	c
dtype:	int64	int64	int64

DataFrame

Filas **Columnas**

Los **DataFrame** son una estructura de datos de 2 dimensión con índices asociados a filas y columnas basado en etiquetas.

```
df = pd.DataFrame(data, index, columns, ...)
```

index= Valores tomados como índice de las filas.

columns= Valores tomados como índice de las columnas.

data= Contenido del **DataFrame** creado.

Con arreglos

```
pd.DataFrame([[1, 2], [3, 4]], columns= ['a', 'b'])
```

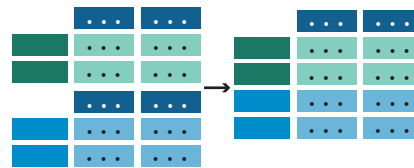
Con diccionarios

```
pd.DataFrame({'a': [1, 3], 'b': [2, 4]})
```

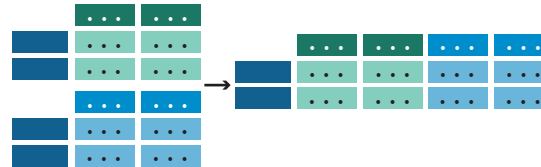
	a	b
0	1	2
1	3	4

Combinación de datos

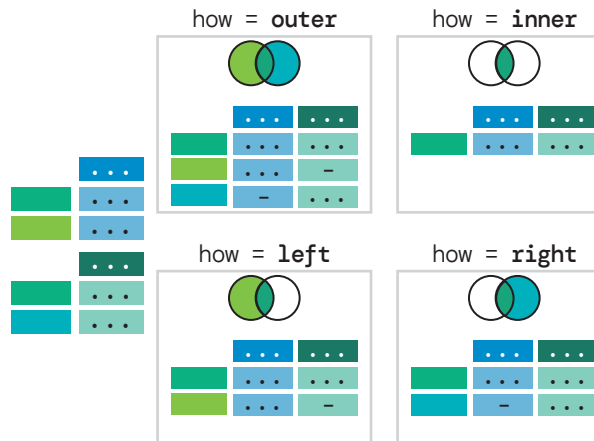
```
pd.concat([df_a, df_b])
```



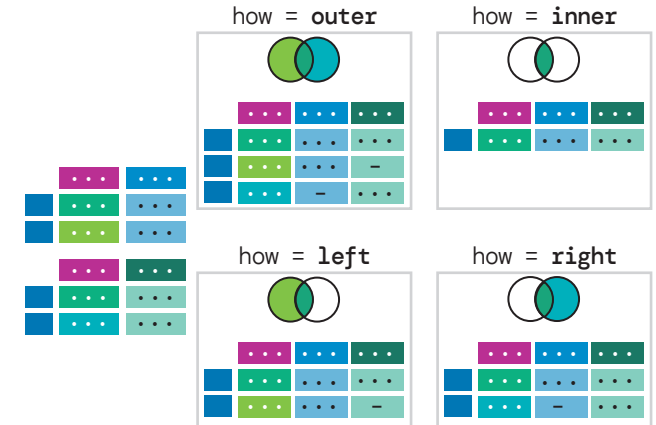
```
pd.concat([df_a, df_b], axis = 1)
```



```
pd.join(df_a, df_b, how = ___)
```

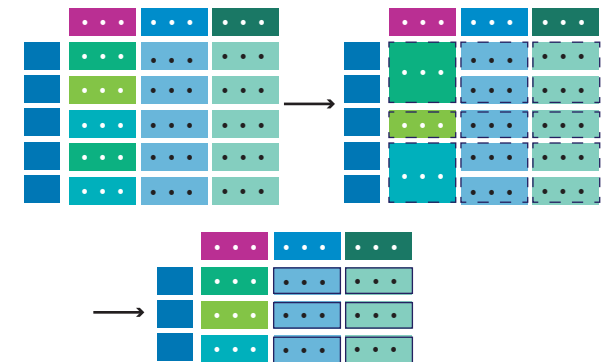


```
pd.merge(df_a, df_b, on = 'col', how = _)
```



Agrupar datos

```
df.groupby('A').agg('B': function, 'C': function, ...)
```



```
df.groupby('A').*
```

Funciones de agregación predefinidas.

- .first()
- .last()
- .sum()
- .prod()
- .size()
- .mean()

Importar datos

pd.read*

```
pd.read_csv(ruta, ...)
pd.read_excel(ruta, ...)
pd.read_table(ruta, ...)
pd.read_json(ruta, ...)
pd.read_sql(query, conector, ...)
...
```

Exportar datos

df.to_*

```
df.to_csv(...)
df.to_excel(...)
df.to_latex(...)
df.to_json(...)
df.to_clipboard(...)
...
```

Selección por etiquetas

df.loc['id'] Filas con etiqueta 'id'.

df.loc['a' : 'f'] Elementos en las filas entre a y f.

df.loc[['a', 'c']] Elementos en las filas a y c.

df.loc['i', 'col'] Elemento en la fila i y columna col.

df.loc[:, 'col'] Elementos en columna 'col'.

df.at['x', 'y'] Elemento en la celda (x, y).

Visualizar datos

df.plot.*

```
df.plot.line(...) Gráfica de líneas.
df.plot.bar(...) Gráfica de barras.
df.plot.pie(...) Gráfica circular.
df.plot.hist(...) Histograma.
df.plot.box(...) Diagrama de cajas.
df.plot.scatter(...) Diagrama de dispersión.
df.plot.area(...) Gráfica de áreas.
df.plot.hexbin(...) Gráfica hexagonal.
```

Selección por posición

df.iloc[0] Fila en la posición 0.

df.iloc[0 : 5] Elementos en las filas entre 0 y 10.

df.iloc[[0, 1]] Elementos en las filas 0 y 1.

df.iloc[0, 3] Elemento en la fila 0 y columna 3.

df.iloc[:, 2] Elementos en la columna 2.

df.iat[1, 2] Elemento en la celda (1, 2).

Selección condicional

Series (<, >, >=, <=, ==, !=) valor
Arreglo de booleanos usado para indexar.

df[df['col'] > 0]
Selección de filas que cumplan una condición.

df[(df['col'] > 0) & (df['col'] < 100)]
Operaciones a nivel de bits (&, |, ~, ^).

df.where(df < 0, -df)
Reemplazar filas que cumplan la condición con el valor de la celda de otro DataFrame.

df.mask(df > 0, -df)
Reemplazar filas que NO cumplan la condición con el valor de la celda de otro DataFrame.

df.query('(a < b) & (b < c)')
Selección condicional con una sintaxis especial basada en nombres de columnas.

Selección de filas

```
df.head(n) Selecciona las primeras n filas.
df.tail(n) Selecciona las últimas n filas.
df.sample(n) Selecciona n filas aleatorias.
df.nsmallest(n, 'col') Selecciona las n filas menores.
df.nlargest(n, 'col') Selecciona las n filas mayores.
```

Describir datos

```
df['col'].min() Valor mínimo.
df['col'].max() Valor máximo.
df['col'].mean() Media aritmética.
df['col'].mode() Moda.
df['col'].median() Mediana.
df['col'].std() Desviación estándar.
df['col'].var() Varianza.
df['col'].quantile(q) Cuantil en posición q.
df['col'].skew() Asimetría.
df['col'].kurt() Curtosis.
df['col'].corr() Correlación.
df.pivot_table(...) Tablas de pivote.
pd.crosstab(...) Tabla de contingencia.
```

Operaciones en objetos

```
df['col'] = 0 Asignar valores escalares.
df['c'] = df['a'] + df['b'] Operadores entre objetos.
df['col'] *= 1000 Asignación con operación.
```

```
df.loc['max'] = df.max() Asignación de filas.
df.iloc[0:5, 0:5] = 3.5 Asignación de rangos.
df[ df['a'] < 0 ] = 0 Asignación condicional.
df.at[0,0] = np.NaN Asignación de celdas.
```

Limpieza de datos

```
df.replace(...) Reemplazar valores.
df.drop_duplicates() Eliminar filas repetidas.
df.dropna() Eliminar valores faltantes.
df.fillna(valor) Imputar valores faltantes.
df.isna() Filas con valores faltantes.
df.notna() Filas sin valores faltantes.
df['col'].unique() Valores únicos de la fila.
df['col'].value_counts() Conteo de valores por columna.
```

Utilidades generales

```
df['col'].idxmin() Etiqueta del valor mínimo.
df['col'].idxmax() Etiqueta del valor máximo.
df.set_index(index) Asignar un nuevo índice.
df.reset_index() Reiniciar el índice.
df.astype(dtype) Modificar tipo de dato.
df.sort_values(by = "col") Reordenar por valores de columna.
```