UNIVERSIDAD DIEGO PORTALES

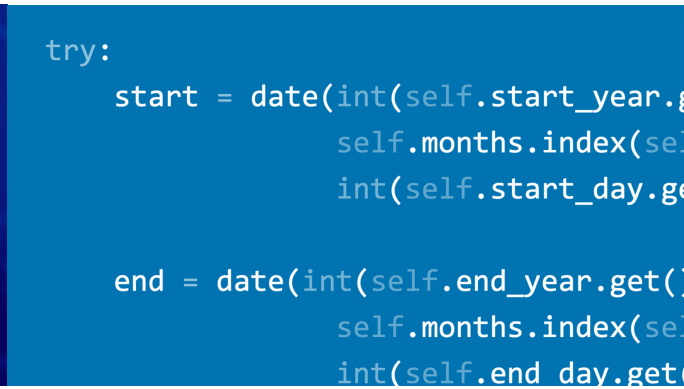
Introducción a Python

Introducción

2026


MY CODE IS RUNNING SLOW
PYTHON?
imgflip.com

- Introducción a Pandas
  - Operaciones básicas de datos
  - Agrupaciones de datos
  - Gráfica matplotlib
  - Gráfica Plotly



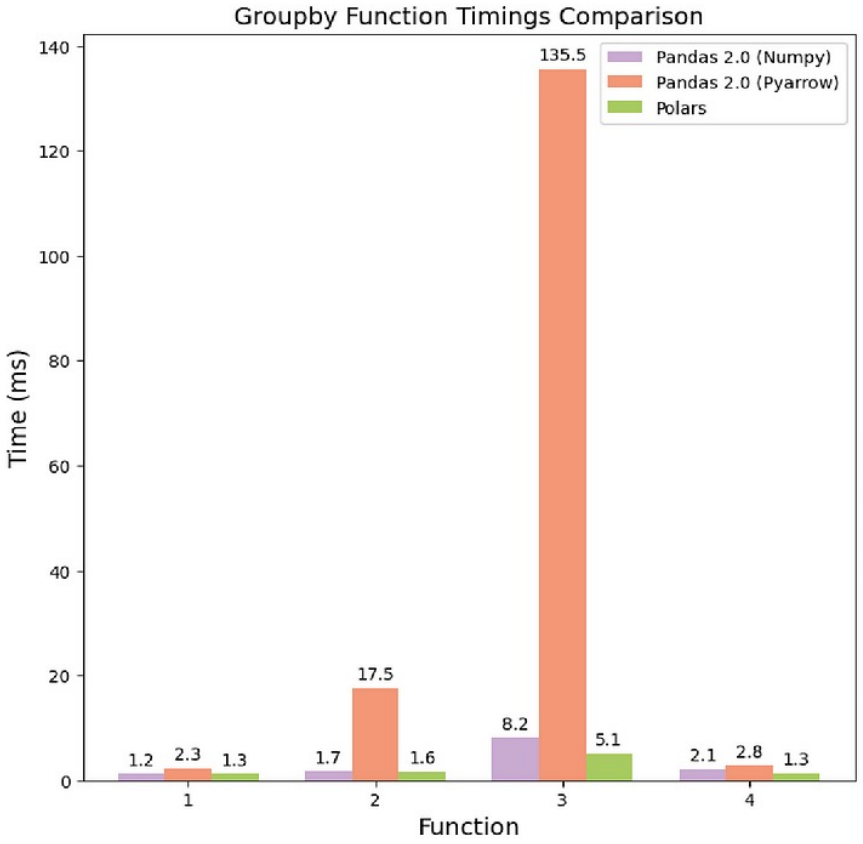
udp UNIVERSIDAD DIEGO PORTALES

- El análisis de datos es el proceso de evaluar datos utilizando herramientas analíticas y estadísticas para descubrir información útil y ayudar en la toma de decisiones de negocios.

Pandas es un módulo (de código abierto) que proporciona:

- estructuras de datos de alto rendimiento y
- herramientas de análisis de datos para el lenguaje de programación Python.

```
#importar pandas
import pandas as pd
```



fuente: https://medium.com/cuenex/pandas-2-0-vs-polars-the-ultimate-battle-a378eb75d6d1

Pandas está basado en otros módulos, tales como Numpy y matplotlib. Usa principalmente dos estructuras de datos:

- Series
- DataFrames

Nos enfocaremos en el último.

Un DataFrame es una matriz (arreglo bidimensional) etiquetada con columnas de tipos potencialmente diferentes.

	ID	Name	Sex	Age	Height	Weight	Team
2	17492	Eugne Henri Callot	M	20	NA	NA	France
3	26280	Georges Daviel de la Nzire	M	17	NA	NA	France
4	26315	Henri de Laborde	M	NA	NA	NA	France
5	26984	J. Defert	M	NA	NA	NA	France
6	12563	Conrad Helmut Fritz Bcker	M	25	NA	NA	Germany
7	12563	Conrad Helmut Fritz Bcker	M	25	NA	NA	Germany
8	12563	Conrad Helmut Fritz Bcker	M	25	NA	NA	Germany
9	12563	Conrad Helmut Fritz Bcker	M	25	NA	NA	Germany
10	12563	Conrad Helmut Fritz Bcker	M	25	NA	NA	Germany
11	12563	Conrad Helmut Fritz Bcker	M	25	NA	NA	Germany

¿Para qué sirve?

Un dataframe nos permite:

- Cargar datos de otra fuente de datos (como un CSV)
- Mantener datos en forma ordenada
- Acceder por nombre o por índice (posición)
- Realizar operaciones sobre datos (como Numpy)
- Entre otras posibilidades

Ejemplo 1

- Para aprender Pandas de forma más práctica, iremos paso a paso usando los siguientes datos almacenados en un archivo CSV

- Lo que haremos es:

  - Cargar los datos
  - Acceder a datos
  - Seleccionar columnas
  - Borrar los datos faltantes
  - Borrar columnas

Carga de datos

```
import pandas as pd

df = pd.read_csv('simple_data.csv')

print(df.describe(include='all'))

print(df.head(10))

print(df.tail(10))
```

read_csv() lee un archivo csv y lo carga en un dataframe

con la opción include

head(n) nos muestra los primeros n registros

tail(n) nos muestra los últimos n registros

	Nombre	Altura	Edad
count	14	12.000000	14.000000
unique	12	NaN	NaN
top	Pedro	NaN	NaN
freq	2	NaN	NaN
mean	NaN	19.000000	32.714286
std	NaN	17.689237	24.602778
min	NaN	3.000000	2.000000
25%	NaN	5.500000	14.750000
50%	NaN	12.500000	28.000000
75%	NaN	26.250000	53.250000
max	NaN	56.000000	78.000000

	Nombre	Altura	Edad
0	Pedro	12.0	23
1	Juan	13.0	12
2	Diego	45.0	23
3	Arturo	24.0	45
4	Felipe	23.0	24
5	Andrés	56.0	56
6	Juan	6.0	2
7	Sofía	NaN	34
8	Antonia	6.0	56
9	Bernardita	4.0	67

Como puede observar los datos de distinto tipo tienen diferentes propiedades estadísticas

Columnas

```
import pandas as pd

df = pd.read_csv('simple_data.csv')

full_data = df[['Nombre','Altura']]

columnas  = df.columns
```

Asignamos las columnas que deseamos a una variable

También podemos almacenar las columnas en una variable

full_data

	Nombre	Altura
0	Pedro	12.2
1	Juan	13.0
2	Diego	45.0
3	Arturo	24.0
4	Felipe	23.0
5	Andrés	56.0
6	Juan	6.0
7	Sofía	23.0
8	Antonia	6.0
9	Bernardita	4.0
10	Camila	3.0
11	Pedro	3.0
12	Alvaro	34.0
13	Magnolia	33.0

hint: *considere emplear pandas_profiling* `[pip3 install pandas_profiling –upgrade]`

Eliminar columna

```
import pandas as pd
df = pd.read_csv('simple_data.csv')
less_data = df.drop(columns=["Edad"])
```

También es possible eliminar columnas que no sean empleadas con el commando .drop

df

	Nombre	Altura	Edad
0	Pedro	12.2	23.0
1	Juan	13.0	12.0
2	Diego	45.0	23.0
3	Arturo	24.0	45.0
4	Felipe	23.0	24.0
5	Andrés	56.0	45.0
6	Juan	6.0	2.0
7	Sofía	23.0	NaN
8	Antonia	6.0	23.0
9	Bernardita	4.0	67.0
10	Camila	3.0	32.0
11	Pedro	3.0	23.0
12	Alvaro	34.0	NaN
13	Magnolia	33.0	45.0

drop

less_data

	Nombre	Altura
0	Pedro	12.2
1	Juan	13.0
2	Diego	45.0
3	Arturo	24.0
4	Felipe	23.0
5	Andrés	56.0
6	Juan	6.0
7	Sofía	23.0
8	Antonia	6.0
9	Bernardita	4.0
10	Camila	3.0
11	Pedro	3.0
12	Alvaro	34.0
13	Magnolia	33.0

Eliminar columna

```
import pandas as pd
df = pd.read_csv('simple_data.csv')
df.drop(columns=["Edad"], inplace=True)
```

Con la opción inplace=True eliminamos los datos en la misma variable

df

	Nombre	Altura	Edad
0	Pedro	12.2	23.0
1	Juan	13.0	12.0
2	Diego	45.0	23.0
3	Arturo	24.0	45.0
4	Felipe	23.0	24.0
5	Andrés	56.0	45.0
6	Juan	6.0	2.0
7	Sofía	23.0	NaN
8	Antonia	6.0	23.0
9	Bernardita	4.0	67.0
10	Camila	3.0	32.0
11	Pedro	3.0	23.0
12	Alvaro	34.0	NaN
13	Magnolia	33.0	45.0

drop

inplace = True

df

	Nombre	Altura
0	Pedro	12.2
1	Juan	13.0
2	Diego	45.0
3	Arturo	24.0
4	Felipe	23.0
5	Andrés	56.0
6	Juan	6.0
7	Sofía	23.0
8	Antonia	6.0
9	Bernardita	4.0
10	Camila	3.0
11	Pedro	3.0
12	Alvaro	34.0
13	Magnolia	33.0

Datos faltantes

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

También es posible eliminar filas que tengan datos faltantes

dropna

df

	Nombre	Altura	Edad
0	Pedro	12.2	23.0
1	Juan	13.0	12.0
2	Diego	45.0	23.0
3	Arturo	24.0	45.0
4	Felipe	23.0	24.0
5	Andrés	56.0	45.0
6	Juan	6.0	2.0
7	Sofía	23.0	NaN
8	Antonia	6.0	23.0
9	Bernardita	4.0	67.0
10	Camila	3.0	32.0
11	Pedro	3.0	23.0
12	Alvaro	34.0	NaN
13	Magnolia	33.0	45.0

df

	Nombre	Altura	Edad
0	Pedro	12.2	23.0
1	Juan	13.0	12.0
2	Diego	45.0	23.0
3	Arturo	24.0	45.0
4	Felipe	23.0	24.0
5	Andrés	56.0	45.0
6	Juan	6.0	2.0
8	Antonia	6.0	23.0
9	Bernardita	4.0	67.0
10	Camila	3.0	32.0
11	Pedro	3.0	23.0
13	Magnolia	33.0	45.0

Ejemplo 2

- Para aprender Pandas de forma más práctica, iremos paso a paso usando datos de las crisis bancarias que han sufrido países del continente Africano a lo largo de los años 1860 a 2014. Una copia de los datos se encuentra en webcursos para que los bajen

- Lo que haremos es:
  - Cargar los datos
  - Analizar un dataframe
  - Acceder a datos
  - Realizar filtros
  - Obtener estadísticas

Carga de datos

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

print(df.describe())

print(df.head(10))
```

read_csv() lee un archivo '.csv' y lo carga en un dataframe

describe() nos permite conocer datos generales del dataframe

head(n) nos muestra los primeros n registros

	case	year	systemic_crisis	exch_usd	domestic_debt_in_default	sovereign_external_debt_default
count	1059.000000	1059.000000	1059.000000	1059.000000	1059.000000	1059.000000
mean	35.613787	1967.767705	0.077432	43.140831	0.039660	0.152975
std	23.692402	33.530632	0.267401	111.475380	0.195251	0.360133
min	1.000000	1860.000000	0.000000	0.000000	0.000000	0.000000
25%	15.000000	1951.000000	0.000000	0.195350	0.000000	0.000000
50%	38.000000	1973.000000	0.000000	0.868400	0.000000	0.000000
75%	56.000000	1994.000000	0.000000	8.462750	0.000000	0.000000
max	70.000000	2014.000000	1.000000	744.306139	1.000000	1.000000

UNIVERSIDAD DIEGO PORTALES

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

print(df.iloc[0])
print(df.iloc[0,2])
```

Muestra el primer registro del listado de países

Muestra el campo #3 del primer registro ('country')

```
case                                1
cc3                               DZA
country                       Algeria
year                             1870
systemic_crisis                     1
exch_usd                     0.052264
domestic_debt_in_default            0
sovereign_external_debt_default     0
gdp_weighted_default                0
inflation_annual_cpi          3.44146
independence                        0
currency_crises                     0
inflation_crises                    0
banking_crisis                 crisis
Name: 0, dtype: object
```

df.iloc[0,2])

df.iloc[0,5])

UNIVERSIDAD DIEGO PORTALES

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

print(df['country'])

print(df['country '][0])
```

Podemos obtener todos los elementos de una columna usando el nombre de la columna

Muestra el primer registro del listado anterior

```
0          Algeria
1          Algeria
2          Algeria
3          Algeria
4          Algeria
           ...
1054      Zimbabwe
1055      Zimbabwe
1056      Zimbabwe
1057      Zimbabwe
1058      Zimbabwe
Name: country, Length: 1059, dtype: object
```

UNIVERSIDAD DIEGO PORTALES

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

data = df[['country', 'year']]

print(data.iloc[0])
```

Podemos obtener más de una columna

Muestra el primer registro del listado anterior

```
country Algeria
year 1870
Name: 0, dtype: object
```

Observación

¿Por qué cuando realizamos la consulta de 1 columna pudimos ir al primer registro haciendo [0] directamente, en cambio en la consulta con 2 columnas tuvimos que usar .iloc[0]?

Observación

¿Por qué cuando realizamos la consulta de 1 columna pudimos ir al primer registro haciendo [0] directamente, en cambio en la consulta con 2 columnas tuvimos que usar .iloc[0]?

Cuando consultamos por 1 columna obtenemos una serie, el equivalente a un arreglo de 1 dimensión → funciona como una lista

```
import pandas as pd
df = pd.read_csv('african_crises.csv')
print(df['country'])
print(df['country '][0])
```

En cambio, al usar 2 o más columnas, obtenemos un dataframe. Por ello debemos usarlo como dataframe

```
import pandas as pd
df = pd.read_csv('african_crises.csv')
data = df[['country', 'year']]
print(data.iloc[0])
```

Estadísticas simples

```python
import pandas as pd

df = pd.read_csv('african_crises.csv')

print(df.shape[0])

unique_country = df['country'].unique()


print(list(unique_country))


n = df['country'].nunique()

print(f'Numero de paises: {n}')
```

shape() nos permite conocer las dimensiones…¡igual que en Numpy!

unique() nos permite eliminar los elementos duplicados

Para mostrar todos los datos en una colección una dimension podemos usar list()

nunique() retorna el número entero único de valores sin repetición.

Filtros

```
import pandas as pd

df = pd.read_csv('african_crises.csv')

bank_crisis= df[df['banking_crisis'] == 'crisis']



unique_country_crisis = bank_crisis['country'].unique()

print(unique_country_crisis)
```

Podemos colocar condiciones para buscar en vez de usar posiciones

```
['Algeria' 'Angola' 'Central African Republic' 'Ivory Coast' 'Egypt'
 'Kenya' 'Mauritius' 'Morocco' 'Nigeria' 'South Africa' 'Tunisia' 'Zambia'
 'Zimbabwe']
```

Filtros

Podemos emplear múltiples condiciones usando & (and), | (or) y separando **con paréntesis** las condiciones

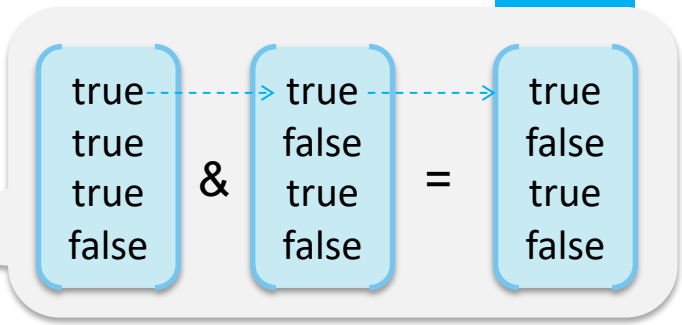
```
import pandas as pd

df = pd.read_csv('african_crises.csv')

crisis_1996 = df[(df['banking_crisis'] == 'crisis') &
                 (df['year'] == 1996)]


unique_country_crisis = crisis_1996['country'].unique()
print(unique_country_crisis)
```

ejemplo



El operador & significa bitwise and, y el operador | realiza bitwise or.

UNIVERSIDAD DIEGO PORTALES

Podemos usar la función fillna() para darle un valor a los que no tienen medalla y luego comparar contra ese valor

```
import pandas as pd

df = pd.read_csv('african_crises.csv')


filter = df[df['inflation_crises'].fillna('none') !='none']


unique_country = filter['country'].unique()
print(unique_country)
```

```
['Algeria' 'Angola' 'Central African Republic' 'Ivory Coast' 'Egypt'
 'Kenya' 'Mauritius' 'Morocco' 'Nigeria' 'South Africa' 'Tunisia' 'Zambia'
 'Zimbabwe']
```

¿Es lo mismo?

```
import pandas as pd

df = pd.read_csv('african_crises.csv')


option_1 = df[df['inflation_crises'].fillna('none') !='none']

option_2 = df[(df['inflation_crises']==0) |
              (df['inflation_crises']==1) ]

print(option_1.equals(option_2))
```

La función equals() permite ver si 2 dataframes tienen los mismos datos

Description

Context

This dataset is a derivative of Reinhart et. al's Global Financial Stability dataset which can be found online at: https://www.hbs.edu/behavioral-finance-and-financial-stability/data/Pages/global.aspx

The dataset will be valuable to those who seek to understand the dynamics of financial stability within the African context.

Content

The dataset specifically focuses on the Banking, Debt, Financial, Inflation and Systemic Crises that occurred, from 1860 to 2014, in 13 African countries, including: Algeria, Angola, Central African Republic, Ivory Coast, Egypt, Kenya, Mauritius, Morocco, Nigeria, South Africa, Tunisia, Zambia and Zimbabwe.

Acknowledgements

Reinhart, C., Rogoff, K., Trebesch, C. and Reinhart, V. (2019) Global Crises Data by Country.
[online] https://www.hbs.edu/behavioral-finance-and-financial-stability/data. Available at: https://www.hbs.edu/behavioral-finance-and-financial-stability/data/Pages/global.aspx [Accessed: 17 July 2019].

fuente: https://www.kaggle.com/chirin/africa-economic-banking-and-systemic-crisis-data

Tiempo : 5 minutos

Muestre los nombres (sin repetir) de los países que hayan estado en crisis y sean independientes con una fecha posterior a 1980

- country - Nombre de países
- independence (0, 1)
- Year (Año de la muestra)

¿qué funciones debo conocer?

```
df['banking_crisis']=='crisis'
df['independece']==1
df['country'].unique()
```

filtros

Tiempo : 5 minutos

```python
import pandas as pd

df = pd.read_csv('african_crises.csv')

filter = df[ (df['banking_crisis']=='crisis') &
             (df['independence']==1 ) &
             (df['year']>1980 )]

countries = filter['country'].unique()
print(countries)
```

¿qué funciones debo conocer?

```python
df['banking_crisis']=='crisis'
df['independece']==1
df['country'].unique()
```

filtros

Description

Context

This dataset is a derivative of Reinhart et. al's Global Financial Stability dataset which can be found online at: https://www.hbs.edu/behavioral-finance-and-financial-stability/data/Pages/global.aspx

The dataset will be valuable to those who seek to understand the dynamics of financial stability within the African context.

Content

The dataset specifically focuses on the Banking, Debt, Financial, Inflation and Systemic Crises that occurred, from 1860 to 2014, in 13 African countries, including: Algeria, Angola, Central African Republic, Ivory Coast, Egypt, Kenya, Mauritius, Morocco, Nigeria, South Africa, Tunisia, Zambia and Zimbabwe.

Acknowledgements

Reinhart, C., Rogoff, K., Trebesch, C. and Reinhart, V. (2019) Global Crises Data by Country.
[online] https://www.hbs.edu/behavioral-finance-and-financial-stability/data. Available at: https://www.hbs.edu/behavioral-finance-and-financial-stability/data/Pages/global.aspx [Accessed: 17 July 2019].

fuente: https://www.kaggle.com/chirin/africa-economic-banking-and-systemic-crisis-data

Tiempo : 10 minutos

Muestre el país del continente africano con el mayor número de crisis en la historia

- country - Nombre de países
- banking_crisis (0, 1)
- Year (Año de la muestra)

¿qué funciones debo conocer?

```
df['banking_crisis']==1
filter['country'].unique()
```

} *filtros*

Tiempo : 10 minutos

```
import pandas as pd
import numpy as np

df = pd.read_csv('african_crises.csv')
filter = df[ (df['banking_crisis']=='crisis')]
countries_list = filter['country'].unique()

numero_crisis = []

for country in countries_list:
  id = filter['country']== country
  numero_crisis.append(sum(id))

id_max= np.argmax(numero_crisis)
print(f'{countries_list[id_max]} posee el mayor numero de crisis con
{numero_crisis[id_max]} eventos.')
```

Recorremos la lista de paises que cumplen la condición anterior

la variable id solo contiene valores verdaderos y falsos.