Recibe una cálida:

¡Bienvenida!

Te estábamos esperando 😁

alkemy

Clase grabada

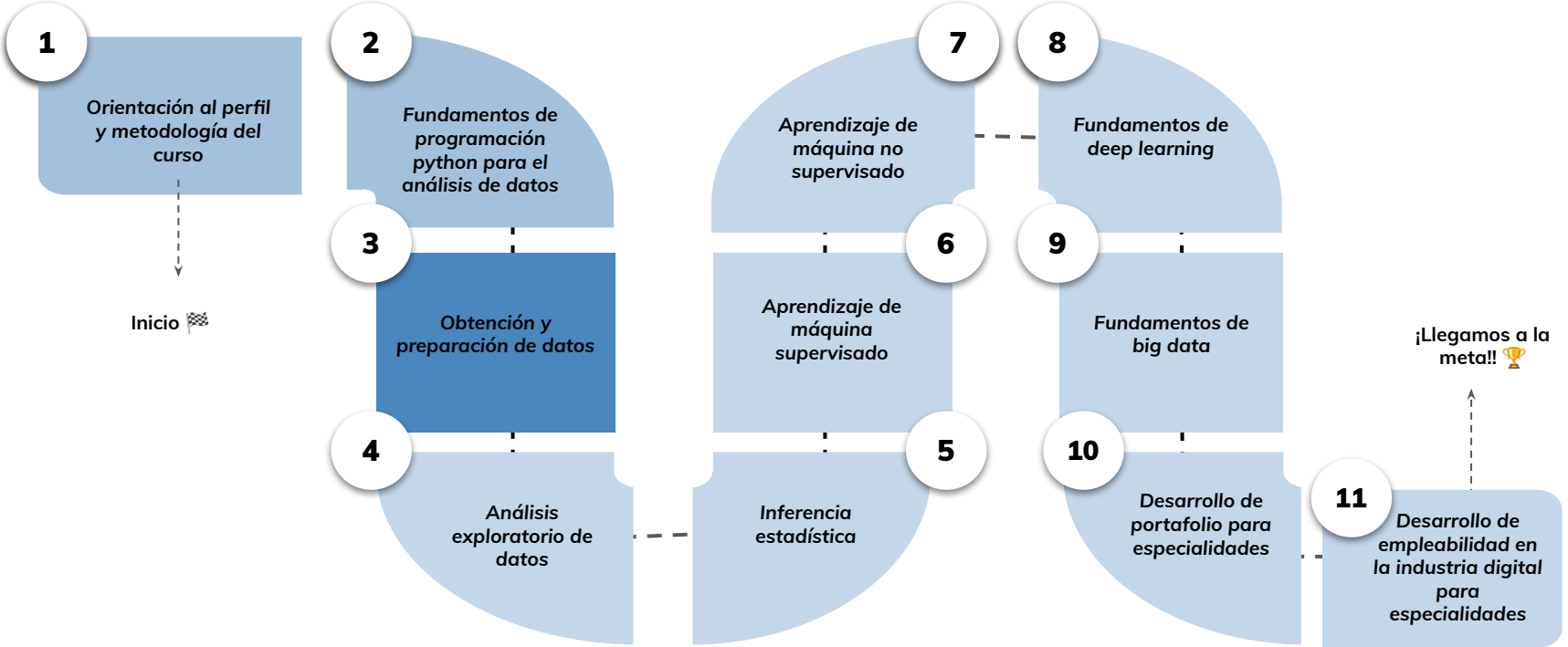
La librería pandas - Parte I

Aprendizaje Esperado 2: Utilizar métodos básicos de exploración en un set de datos utilizando estructuras de tipo serie y dataframe de la librería pandas para la selección, filtrado y sumarización de los datos para la resolución de un problema.

alkemy

Hoja de ruta

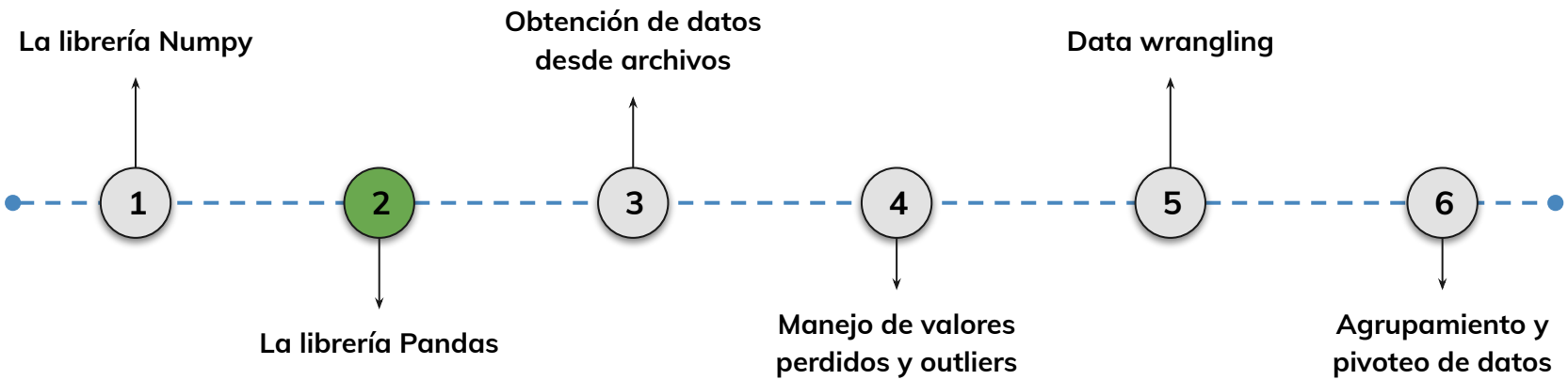
¿Cuáles skills conforman el programa? **Fundamentos de Ciencia de Datos**



alkemy

Roadmap de lecciones

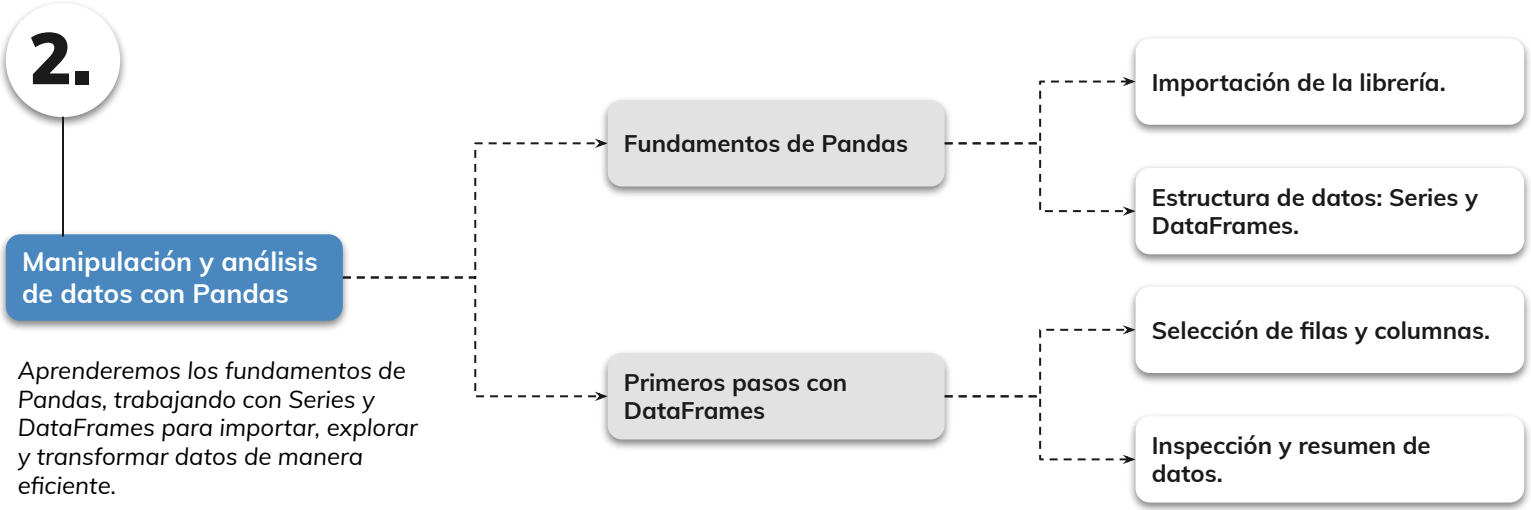
*¿Cuáles **lecciones** estaremos estudiando en este módulo?*



alkemy

Learning Path

¿Cuáles temas trabajaremos hoy?



alkemy

Objetivos de aprendizaje

¿Qué aprenderás?

- Comprender la estructura y utilidad de Series y DataFrames en Pandas.
- Crear Series y Data Frames desde distintas fuentes de datos.
- Manipular y acceder a datos usando índices y nombres de columnas.
- Realizar operaciones básicas de filtrado, selección y resumen.
- Integrar Pandas con otras librerías del ecosistema Python.

Repaso clase anterior

¿Quedó alguna duda?

En la clase anterior trabajamos 📚:

- → *Aprendimos indexación y selección en NumPy, incluyendo acceso a elementos, slicing y selección condicional.*
- → *Comprendimos la diferencia entre referencias y copias de arreglos y sus implicaciones en memoria y rendimiento.*
- → *Realizamos operaciones aritméticas, broadcasting y operaciones con escalares de manera optimizada.*
- → *Aplicamos funciones matemáticas, estadísticas y de agregación sobre arreglos.*

La librería pandas

Pandas en el Ecosistema Python

Pandas se ha convertido en una herramienta indispensable dentro del ecosistema de Python debido a su integración con bibliotecas como NumPy, Matplotlib y Scikit-learn, lo que facilita su aplicación en análisis exploratorio, modelado de datos y visualización.

Python

Lenguaje base que proporciona la flexibilidad y potencia para el desarrollo de aplicaciones de análisis de datos.

NumPy

Biblioteca fundamental para cálculos numéricos que sirve como base para Pandas.

Matplotlib

Biblioteca de visualización que se integra perfectamente con Pandas para crear gráficos a partir de datos.

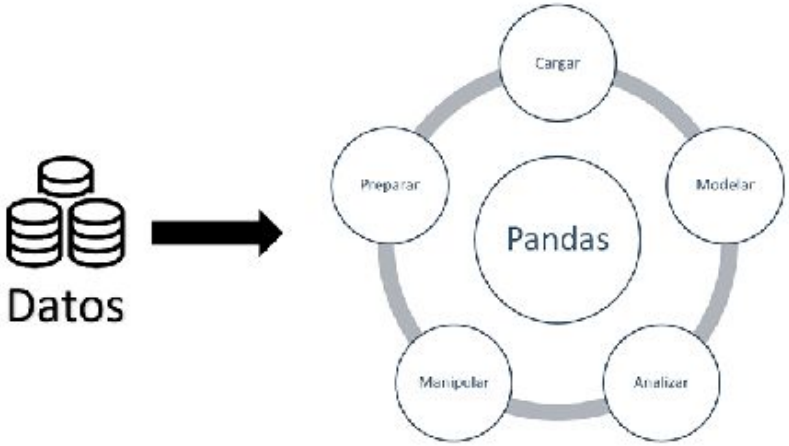
Scikit-learn

Biblioteca de machine learning que utiliza estructuras de Pandas para el modelado predictivo.

Reseña de la librería pandas

Pandas es una biblioteca de código abierto construida sobre NumPy y diseñada para facilitar el manejo y análisis de datos estructurados en Python. Fue desarrollada originalmente por Wes McKinney en 2008 y desde entonces se ha convertido en un estándar en el ámbito del análisis de datos.

A diferencia de NumPy, que se centra en arreglos numéricos multidimensionales, Pandas ofrece estructuras de datos más flexibles y expresivas, diseñadas para facilitar el manejo de datos tabulares y heterogéneos. Su principal ventaja es la capacidad de procesar grandes volúmenes de datos de manera eficiente y realizar transformaciones de manera sencilla.



Características Principales de Pandas

Pandas permite cargar, limpiar, analizar y visualizar datos de diferentes fuentes como archivos CSV, Excel, bases de datos SQL y JSON. Gracias a su integración con otras bibliotecas como Matplotlib y Seaborn, se puede complementar el análisis de datos con visualizaciones detalladas y representativas.

Estructuras de Datos

Series y DataFrames para manejar datos unidimensionales y bidimensionales

Manipulación

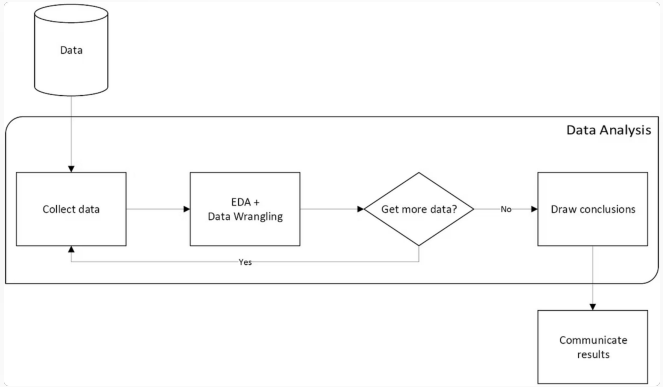
Herramientas para transformar, filtrar y agregar datos

Integración

Compatibilidad con múltiples formatos y bibliotecas

Para qué se utiliza

Pandas es ampliamente utilizada en la manipulación y análisis de datos provenientes de diferentes fuentes, como archivos CSV, Excel, bases de datos SQL y JSON.



Principales usos de Pandas

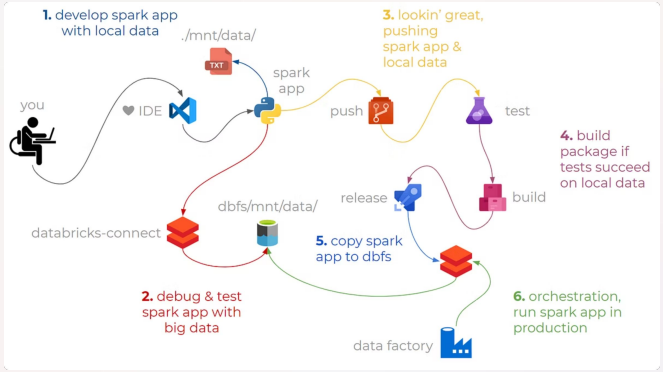
1	2	3
Carga y limpieza de datos	**Transformación de datos**	**Análisis estadístico**
Permite leer y escribir datos desde múltiples formatos, facilitando la conversión y depuración de información. Esto es clave en la etapa de preprocesamiento de datos antes de su análisis o modelado.	Ofrece herramientas para modificar la estructura de los datos, incluyendo la eliminación de valores nulos, la normalización y el manejo de datos categóricos, lo que permite adaptar la información a diferentes necesidades.	Proporciona métodos para calcular estadísticas descriptivas, identificar patrones y resumir datos mediante funciones como mean(), median(), std(), entre otras.

Más Usos de Pandas

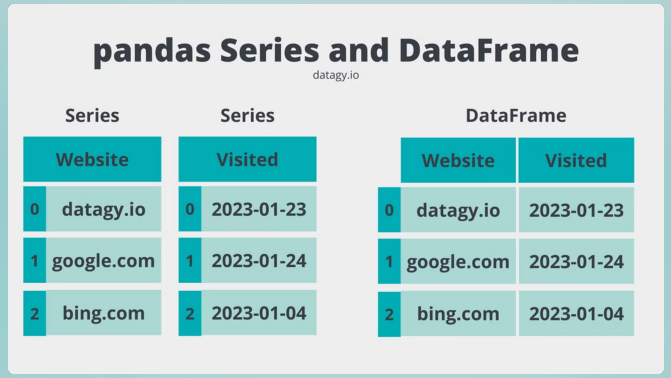
1	2	3
Filtrado y selección de datos	**Visualización de datos**	**Preparación de datos para machine learning**
Permite realizar consultas avanzadas sobre conjuntos de datos de manera rápida y eficiente, optimizando procesos en grandes volúmenes de información.	Se integra con bibliotecas de gráficos como Matplotlib y Seaborn para generar representaciones gráficas de los datos, facilitando su interpretación.	Es utilizada para la manipulación de datasets antes de aplicar modelos de aprendizaje automático, permitiendo normalizar, escalar y transformar características en los datos de entrenamiento.

Importancia de Pandas en Ciencia de Datos

Gracias a estas funcionalidades, Pandas se ha convertido en una de las herramientas más utilizadas en la ciencia de datos, proporcionando una manera eficiente y flexible de trabajar con información estructurada.



EL TIPO DE DATO SERIE



Características del tipo de dato serie

Una Serie en Pandas es una estructura de datos unidimensional que puede almacenar cualquier tipo de datos, incluyendo números, texto y fechas. Se asemeja a una columna de una hoja de cálculo o a un array de NumPy, pero con la ventaja de poseer un índice asociado a cada elemento.

Estructura Unidimensional
Similar a un array o lista, pero con índices personalizables

Flexibilidad de Datos
Puede contener diferentes tipos de datos en una misma estructura

Operaciones Vectorizadas
Permite realizar cálculos sin necesidad de bucles explícitos

Principales Características de una Serie

Índice Único

Posee un índice único que permite acceder a cada elemento.

Tipos de Datos Heterogéneos

Puede contener datos de diferentes tipos, incluyendo valores numéricos y categóricos.

Operaciones Matemáticas

Admite operaciones matemáticas y de transformación sobre sus elementos, facilitando la manipulación de datos sin necesidad de bucles.

Integración

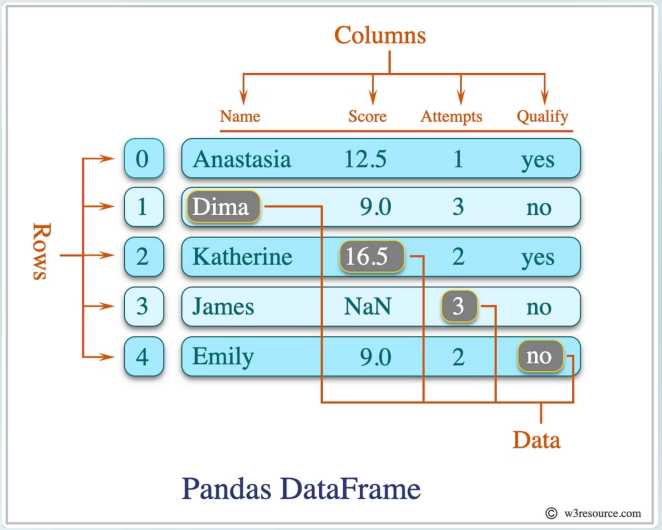
Se puede crear a partir de listas, diccionarios o arrays de NumPy, lo que permite una integración fluida con otras bibliotecas de Python.

Filtrado Eficiente

Permite filtrar y seleccionar datos de manera eficiente mediante índices y condiciones lógicas.

Importancia de las Series en Análisis de Datos

Las Series son fundamentales en el análisis de datos, ya que representan variables individuales dentro de un conjunto más grande y facilitan cálculos estadísticos y exploraciones de datos.



Creación de una serie

Se puede crear una Serie en Pandas a partir de diferentes fuentes, como listas, diccionarios o arrays de NumPy.

```
import pandas as pd

# Crear una Serie a partir de una lista
serie = pd.Series([10, 20, 30, 40, 50])
print(serie)
```

Creación de series con Índices Personalizados

También se puede asignar un índice personalizado a cada elemento:

En este caso, cada elemento de la Serie está asociado a una etiqueta en lugar de un índice numérico, lo que facilita el acceso y la manipulación de datos.

```
serie_indexada = pd.Series([100, 200, 300], index=['a', 'b', 'c'])
print(serie_indexada)
```

Obtención de datos de una serie

Una Serie en Pandas permite acceder a elementos específicos mediante su índice. Existen varias formas de extraer datos de una Serie, incluyendo el uso de índices numéricos y etiquetas.

Para obtener un solo elemento, se puede utilizar el índice correspondiente:

```
import pandas as pd
serie = pd.Series([10, 20, 30, 40], index=['a', 'b', 'c', 'd'])
print(serie['b'])  # Output: 20
```

Métodos de Acceso a Datos en Series

Selección por Índice

También es posible seleccionar múltiples elementos utilizando listas de índices.

Método .iloc[]

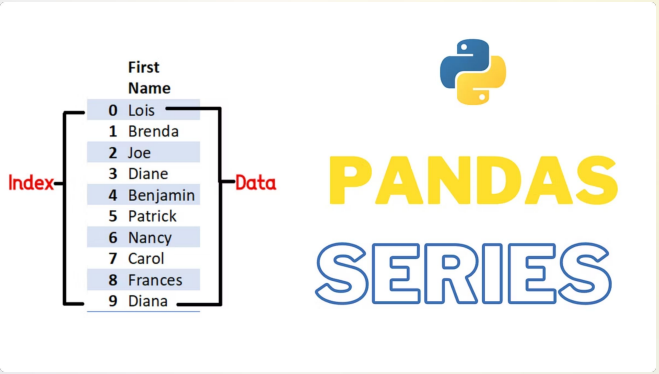
El método .iloc[] permite acceder a los elementos de la Serie mediante su posición numérica.

Slicing

El slicing se puede utilizar para obtener rangos de datos.

Filtrado Condicional

Si se desea extraer valores que cumplan ciertas condiciones, se puede aplicar filtrado condicional.



Operaciones sobre una serie

Las Series permiten realizar operaciones matemáticas y de transformación de manera eficiente. Se pueden realizar cálculos entre Series y escalares, así como aplicar funciones estadísticas y de agregación.

Operaciones Aritméticas con Escalares

Se pueden realizar operaciones directamente sobre la Serie, y los valores se modificarán elemento por elemento:

```
# Multiplicar todos los elementos por 2
serie_doble = serie * 2

# Sumar 5 a todos los elementos
serie_mas_5 = serie + 5

# Elevar al cuadrado todos los elementos
serie_cuadrado = serie ** 2
```

Operaciones entre Series

Si se realizan operaciones entre dos Series, estas se aplican a los elementos correspondientes. Si los índices no coinciden, Pandas rellena los valores faltantes con NaN.

$$series2 = pd.Series([5, 10, 15, 20], index = ['a', 'b', 'c', 'd'])$$

```
print(serie + serie2) # Suma elemento a elemento
```

Aplicación de Funciones Matemáticas

Pandas proporciona funciones matemáticas para operar sobre una Serie sin necesidad de bucles explícitos:

```
import numpy as np
# Calcular el logaritmo natural
log_serie = np.log(serie)
# Calcular el valor absoluto
abs_serie = serie.abs()
# Calcular la raíz cuadrada
sqrt_serie = np.sqrt(serie)
```

Funciones Personalizadas en Series

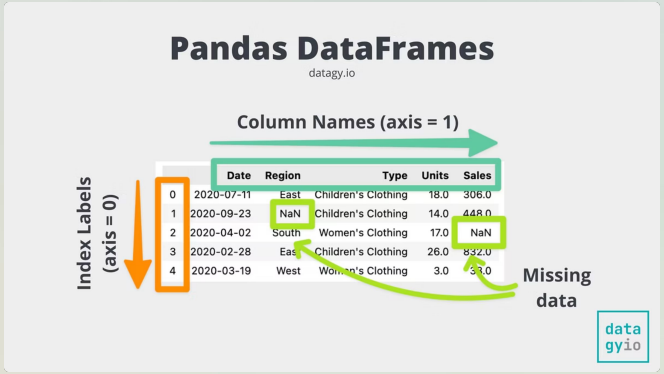
También se pueden aplicar funciones personalizadas mediante apply():

```
# Definir una función personalizada
def cuadrado_mas_uno(x):
    return x**2 + 1

# Aplicar la función a cada elemento de la Serie
resultado = serie.apply(cuadrado_mas_uno)
```

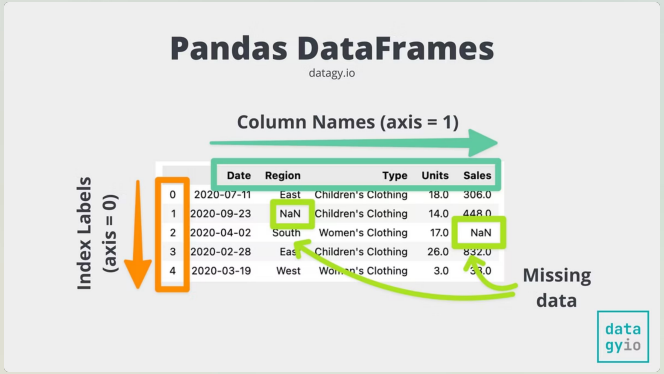
Las operaciones sobre Series permiten transformar y analizar datos de manera eficiente sin necesidad de realizar iteraciones manuales.

EL TIPO DE DATO DATAFRAME



Características del tipo de dato DATAFRAME

Un DataFrame en Pandas es una estructura de datos bidimensional, similar a una hoja de cálculo o a una tabla en una base de datos. A diferencia de una Serie, un DataFrame puede contener múltiples columnas con diferentes tipos de datos.



Principales Características de un DataFrame

Estructura Bidimensional

Posee etiquetas en las filas y columnas, facilitando la selección y manipulación de datos.

Datos Heterogéneos

Permite almacenar datos de diferentes tipos en una misma estructura, como números, texto y fechas.

Múltiples Fuentes

Se puede crear a partir de listas, diccionarios, archivos CSV, bases de datos SQL, entre otros.

Operaciones Avanzadas

Soporta operaciones avanzadas como filtrado, agrupación y transformación de datos, lo que permite analizar grandes volúmenes de información de manera eficiente.

Integración

Se integra con otras bibliotecas para análisis y visualización de datos, como NumPy, Matplotlib y Seaborn.

Importancia de los DataFrames

Los DataFrames son el tipo de dato más utilizado en Pandas debido a su flexibilidad y capacidad para manejar datos estructurados de diferentes fuentes. Su estructura tabular facilita la aplicación de operaciones matemáticas, estadísticas y de transformación de datos.

Creación de un DATAFRAME

Un DataFrame se puede crear de varias maneras, incluyendo listas de listas, diccionarios y archivos externos.

```
import pandas as pd

# Crear un DataFrame a partir de un diccionario

data = {

    'Nombre': ['Ana', 'Juan', 'María', 'Carlos'],

    'Edad': [25, 30, 22, 28],

    'Ciudad': ['Madrid', 'Barcelona', 'Sevilla', 'Valencia']

}

df = pd.DataFrame(data)
```

Creación de DataFrames desde Archivos

También se puede leer un archivo CSV para crear un DataFrame, lo que facilita la manipulación de datos provenientes de diversas fuentes:

```
df_csv = pd.read_csv('archivo.csv')
print(df_csv.head())
```

Aplicaciones de los DataFrames

Los DataFrames permiten una fácil manipulación y transformación de datos, lo que los hace fundamentales en el análisis de información estructurada. Son utilizados en múltiples aplicaciones, como análisis financiero, estudios científicos, inteligencia de negocios y aprendizaje automático.

Análisis Financiero

Procesamiento de datos de mercados, acciones y tendencias económicas.

Investigación Científica

Análisis de resultados experimentales y datos de observaciones.

Machine Learning

Preparación y transformación de datos para modelos predictivos.

Live Coding

¿En qué consistirá la Demo?

Mostraremos paso a paso cómo trabajar con Series y DataFrames en Pandas para explorar y manipular datos de forma sencilla.

1. *Importar la librería Pandas (import pandas as pd).*
2. *Crear Series desde listas y diccionarios.*
3. *Personalizar índices y acceder a elementos.*
4. *Crear DataFrames desde diccionarios y listas de listas.*
5. *Cargar un CSV y mostrar las primeras filas con head().*
6. *Seleccionar columnas y filas con .loc[] y .iloc[].*

Tiempo: 25 Minutos

Momento:

Time-out!

⏰5 -10 min.

Ejercicio N° 1

Primeros pasos con Pandas

Primeros pasos con Pandas

Contexto: 🙌

Pondrás en práctica la creación y manipulación básica de Series y DataFrames para conocer a fondo sus funcionalidades.

Consigna: ✍️

1. Crea una Serie con al menos 5 elementos y un índice personalizado.
2. Realiza una operación aritmética sobre la Serie y aplica un filtro.
3. Crea un DataFrame desde un diccionario con al menos 3 columnas y 5 filas.
4. Selecciona una columna, una fila y un subconjunto de datos.
5. Carga un archivo CSV y muestra las primeras y últimas 5 filas.

Tiempo 🕐**:** *45 Minutos*

Primeros pasos con Pandas

Paso a paso: ⚙️

1. Importar Pandas y preparar datos de prueba.
2. Crear la Serie y aplicar operaciones.
3. Construir el DataFrame y practicar selección de datos.
4. Cargar y explorar un CSV.

¿Alguna consulta?

alkemy

Resumen

¿Qué logramos en esta clase?

- ✓ ***Conocimos los fundamentos de Pandas y su rol en análisis de datos.***
- ✓ ***Aprendimos a crear y manipular Series y DataFrames.***
- ✓ ***Usamos índices para acceder y filtrar información.***
- ✓ ***Cargamos datos desde archivos externos y los exploramos.***
- ✓ ***Sentamos las bases para análisis de datos más avanzados en Python.***

¡Ponte a prueba!

Momento de ejercitación

Te invitamos a aprovechar esta última sección del espacio sincrónico para realizar de manera individual las ***actividades disponibles en la plataforma.*** *Estas propuestas son clave para afianzar lo trabajado y* ***forman parte obligatoria del recorrido de aprendizaje.***

💪 Análisis de caso ——— 💪 Selección Múltiple

💪 Comprensión lectora

Si al resolverlas surge alguna duda, compartela o tráela al próximo encuentro sincrónico.

alkemy

¡Muchas gracias!

alkemy