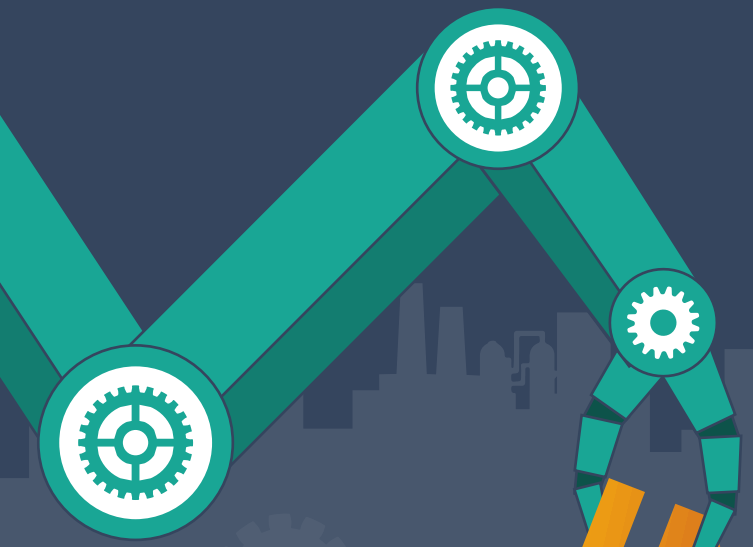
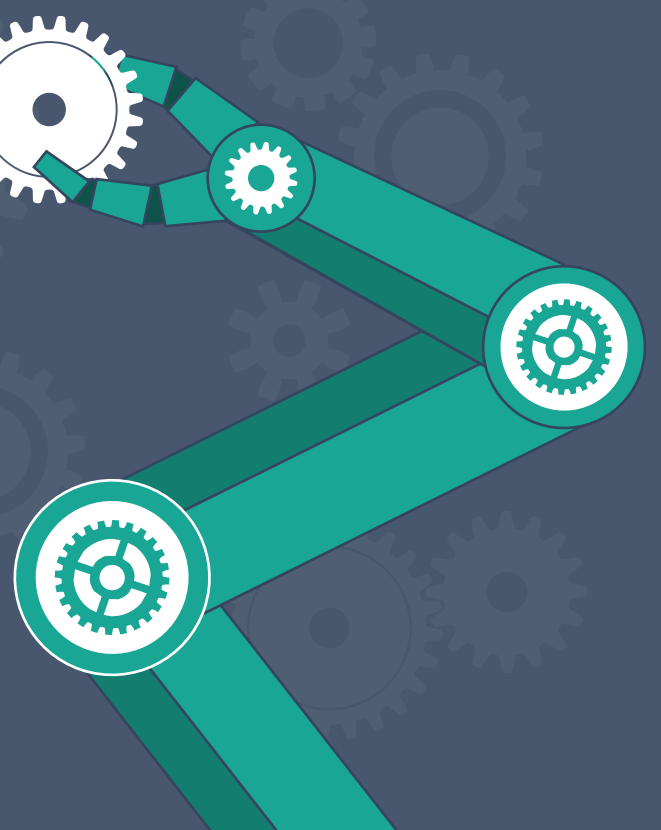
SAGIM

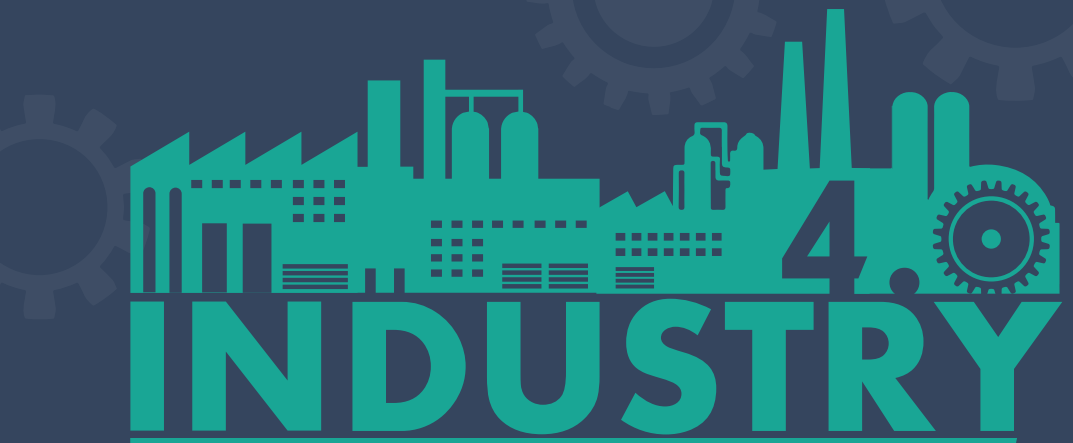


INDUSTRY 4.0

# PANDAS

## INTRODUCTION A LA BIBLIOTHEQUE PANDAS

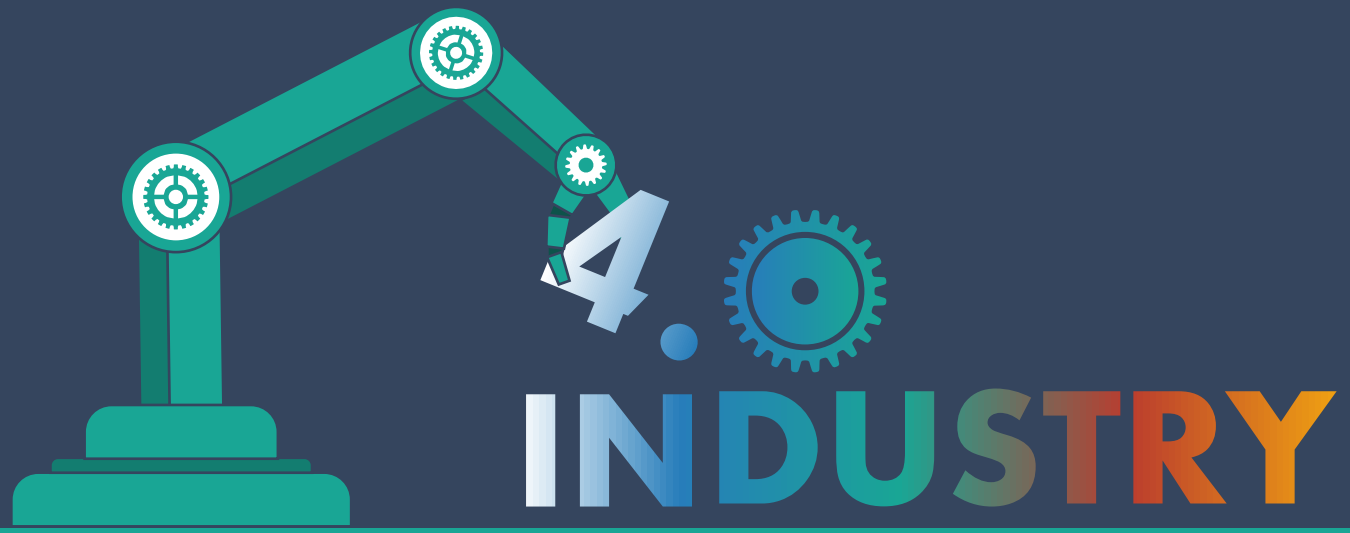





# Thank You

Insert the Sub Title of
Your Presentation

SAGIM




# PANDAS

ANALYSE ET LE TRAITEMENT DES DONNÉES

SAGIM

# INTRODUCTION

Pandas est une bibliothèque open-source pour le langage Python, conçue pour faciliter la manipulation, l'analyse et le traitement des données.

Très populaire dans le domaine de la science des données, elle permet de travailler facilement avec des ensembles de données tabulaires et structurées, comme les fichiers CSV, Excel, bases de données SQL, ou des données issues d'APIs.

SAGIM

# Caractéristiques principales de Pandas

## Structures de données puissantes :

**Series** : Une structure unidimensionnelle (comme une colonne dans un tableau ou une liste).

**DataFrame** : Une structure bidimensionnelle, comme une table Excel, avec des lignes et des colonnes.

SAGIM

# Caractéristiques principales de Pandas

## Fonctionnalités avancées :

- ✓ Lecture/écriture de différents formats de fichiers (CSV, Excel, JSON, SQL, etc.).
- ✓ Nettoyage et préparation des données : gestion des valeurs manquantes, duplication, etc.
- ✓ Opérations rapides sur les données, comme le tri, le filtrage, et l'agrégation.
- ✓ Groupement des données pour des analyses avancées.
- ✓ Fusion et jointure des ensembles de données.

SAGIM

# Installation de Pandas

Avant tout, assure-toi que Pandas est installé.
Pour l'installer, utilise la commande suivante dans ton terminal ou environnement Python :

```
pip install pandas
```

SAGIM

# Concepts de base

## Les objets principaux :

**Series** : Une série est une structure de données unidimensionnelle, comme une liste ou une colonne.

**DataFrame** : Un tableau bidimensionnel avec des lignes et colonnes (semblable à un tableau Excel ou une table SQL).

SAGIM

# Concepts de base

```
import pandas as pd

# Créer une Series
serie = pd.Series([10, 20, 30, 40])
print(serie)
# Créer un DataFrame
data = {'Nom': [‘said’,’ahmed', ‘khadija'],
        'Âge': [25, 30, 35],
        'Ville': [‘casa', ‘rabat', ‘meknes’]}
df = pd.DataFrame(data)
print(df)
```

SAGIM

# Les DataFrame (s)

Un DataFrame est une structure de données tabulaire similaire à une feuille Excel ou un tableau SQL.

Syntaxe :

```
import pandas as pd
data = {'Nom': ['Alice', 'Bob', 'Charlie’],
         'Age': [25, 30, 35]
        }
df = pd.DataFrame(data)
print(df)
```

SAGIM

# Les DataFrame (s)

## Exercice 1 :

Créez un DataFrame contenant les colonnes suivantes : Prénom, Ville, et Salaire.
Les valeurs doivent être :

- Prénom : ['John', 'Anna', 'Peter']
- Ville : ['Paris', 'Lyon', 'Marseille']
- Salaire : [50000, 60000, 70000]

SAGIM

# Les DataFrame (s)

Solution 1 :

```
import pandas as pd

data = {'Prénom': ['John', 'Anna', 'Peter'],
        'Ville': ['Paris', 'Lyon', 'Marseille'],
        'Salaire': [50000, 60000, 70000]}
df = pd.DataFrame(data)
print(df)
```

# Les DataFrame (s)

## Exercice 2 :

Ajoutez une colonne Prime au DataFrame précédent avec les valeurs [5000, 6000, 7000].

SAGIM

# Les DataFrame (s)

Solution 2 :

```
import pandas as pd

data = {'Prénom': ['John', 'Anna', 'Peter'],
        'Ville': ['Paris', 'Lyon', 'Marseille'],
        'Salaire': [50000, 60000, 70000]}
df = pd.DataFrame(data)
df['Prime'] = [5000, 6000, 7000]
print(df)
```

SAGIM

# Lecture d'un fichier CSV

La fonction pd.read_csv() permet de charger un fichier CSV dans un DataFrame.

## Syntaxe :

```
import pandas as pd
df = pd.read_csv('fichier.csv')
print(df.head())  # Affiche les premières lignes du DataFrame
```

SAGIM

# Lecture d'un fichier CSV

## Exercice 1 :

Chargez le fichier employees.csv (disponible localement) et affichez les 5 premières lignes.

SAGIM

# Lecture d'un fichier CSV

## Solution 1 :

```
import pandas as pd
df = pd.read_csv('employees.csv')
print(df.head())
```

## Exercice 2 :

Affichez les 10 dernières lignes du même fichier.

SAGIM

# Lecture d'un fichier CSV

## Solution 2 :

```
import pandas as pd
df = pd.read_csv('employees.csv')
print(df.tail(10))
```

SAGIM

# Sélection de colonnes

Vous pouvez sélectionner une ou plusieurs colonnes spécifiques d'un DataFrame.

Syntaxe :

```
# Sélection d'une seule colonne
colonne = df['Nom']

# Sélection de plusieurs colonnes
colonnes = df[['Nom', 'Age']]
```

SAGIM

# Sélection de colonnes

## Exercice 1 :

À partir du DataFrame précédent (employees.csv), sélectionnez uniquement la colonne Nom.

SAGIM

# Sélection de colonnes

## Solution 1:

```
import pandas as pd
df = pd.read_csv('employees.csv')
noms = df['Nom']
print(noms)
```

SAGIM

# Sélection de colonnes

## Exercice 2 :

Sélectionnez les colonnes Nom et Salaire du DataFrame.

SAGIM

# Sélection de colonnes

## Solution :

```
import pandas as pd
df = pd.read_csv('employees.csv')
nom_salaire = df[['Nom', 'Salaire']]
print(nom_salaire)
```

SAGIM

# Filtrage de données

Le **filtrage** permet de sélectionner des lignes basées sur certaines conditions.

SAGIM

# Filtrage de données

## Exemple :

```
# Filtrer les employés dont le salaire est supérieur
# à 50000
import pandas as pd
df = pd.read_csv('employees.csv')
filtered_df = df[df['Salaire'] > 50000]
```

SAGIM

# Filtrage de données

## Exercice 1 :

Filtrez les employés dont le salaire est supérieur à 60000.

## Solution :

```
import pandas as pd
df = pd.read_csv('employees.csv')
high_salary = df[df['Salaire'] > 60000]
print(high_salary)
```

SAGIM

# Filtrage de données

## Exercice 2 :

Filtrez les employés vivant à Paris.

## Solution :

```
import pandas as pd
df = pd.read_csv('employees.csv')
paris_employees = df[df['Ville'] == ‘Casa']
print(paris_employees)
```

SAGIM

# Ajout de colonnes calculées

Vous pouvez ajouter des colonnes basées sur des calculs existants.

Syntaxe :

```
df[‘NewCol'] = df[‘Col1'] + df[‘Col2']
```

SAGIM

# Ajout de colonnes calculées

**Exercice 1 :**

Ajoutez une colonne Total_Salaire qui correspond à la somme de Salaire et Prime.

**Solution :**

```
import pandas as pd
df = pd.read_csv('employees.csv')
df['Total_Salaire'] = df['Salaire'] + df['Prime']
print(df)
```