Polars

Hadrien Bodin

Mines

15 octobre 2025

Table des matières

Polars

Tout comme Pandas, Polars est une bibliothèque Python de traitement de données structurées.
Cette bibliothèque récente est de plus en plus utilisée et se présente comme la concurrente directe de Pandas.

Polars

Les auteurs de la librairie citent de nombreux points d'intérêts majeurs de cette librairie.
N'ayant pas tout testé, et ayant un recul sur certaines choses plus que d'autres, je retiendrai -personnellement- ces derniers :

- Polars est rapide (bien plus que Pandas).
- La syntaxe est intuitive et agréable (après un léger temps d'adaptation pour se défaire de la syntaxe Pandas)
- Le logo est plus mignon que celui de Pandas

La rapidité

La supériorité de Polars par rapport à Pandas sur ce point s'explique par :

- Un noyau en Rust (APIfié pour Python)
- Une architecture multi-cpu
- La possibilité de transférer des calculs sur GPU

Possibilité de streamer les tableaux sans tout charger en RAM.
On peut donc théoriquement manipuler des tableaux plus gros que sa RAM. (Je n'ai jamais essayé).

La rapidité - point d'attention

De nombreux benchmark ont été réalisés et semblent unanimes quant à la supériorité en temps de calcul de Polars par rapport à Pandas.

Au demeurant :

- Le gain en performance semble dépendre étroitement du type de donnée, et de la shape du tableau
- Il est donc difficile de reproduire rapidement des benchmark pertinents avec des tableaux générés

Point d'attention

Pour nuancer : il semble que Pandas reste plus rapide que Polars sur les très petits tableaux.

Rapidité à l'ouverture

Pour un tableau de taille (33342,8) :

```
%%timeit
pl.read_csv('corona-par-pays.csv', separator=';') #polars
    -> 11.6 ms     424   s   per loop
%%timeit
pd.read_csv('corona-par-pays.csv', separator=';') #pandas
    -> 73.3 ms     2.2 ms per loop
```

Rapidité au filtrage

Pour un tableau de taille (33342,8) :

```
%%timeit
df_l.filter(pl.col('Pays')=='France')
    ->1.7 ms     68.5   s   per loop
%%timeit
df_l.filter(df_l['Pays']=='France')
    ->945   s      55.5   s   per loop
%%timeit
df_p[df_p['Pays']=='France']
    ->5.19 ms     639   s   per loop
```

Bug

La bonne manière de faire en Polars pour filtrer est la première et non la seconde. Elle est cependant anormalement lente. Ce bug est connu des maintainers.

Import

```
import polars as pl
```

Les types d'objets

Les objets sont très similaires à ceux rencontrés en Pandas :

- Les dataframe
- Les series

En plus on trouve :

- Les lazyframe : un type qui permet d'optimiser encore plus les opérations sur les tableaux.
- Les expressions : en gros, ce sont les fonctions d'agrégation

Rien de révolutionnaire par rapport à Pandas.

Les lazyframe

Cet objet est celui qui permet d'avoir la meilleure performance en polars.
Je l'utilise en réalité très peu (à tord) donc ne centrerai pas la présentation dessus.

```
q = (
    pl.scan_csv("iris.csv")
    .filter(pl.col("sepal_length") > 5)
)

df = q.collect()
```

Les lignes qui ne répondent pas au critère ne seront pas lues -> gain de temps très significatif.

Lire un fichier

Polars se targue de lire toutes les structures de données imaginables.

On utilise cependant le plus souvent le constructeur de dataframe :

```
df = pl.read_csv('data.csv')
```

Les arguments ressemblent à ceux du read_csv de Pandas.

Index

Toutes les questions sur les index et les indiçages en Pandas se résument par :

Il n'y a pas d'indexation laissée à l'utilisateur en Polars.

Modification d'une valeur dans la table

Avec Pandas on peut être tenté de modifier une valeur dans la table par l'appel de df.loc ou df.iloc ou autre.

Il est impossible d'éditer une unique valeur en Polars.
Tout doit se faire sur les colonnes.

Attention

Ces pratiques pandas sont à mon sens à proscrire.
Vous l'avez d'ailleurs vu en TD avec les nombreux warnings qui remontaient.

La syntaxe

- pl.col('nom') = accèder à la dite colonne
- df.filter() = filtrer les lignes sur une condition
- df.select() = récupérer un dataframe en sélectionnant certaines colonnes
- df.with_columns() = ajoute des colonnes

Exemples

https://docs.pola.rs/user-guide/getting-started/