**Polars** is a fast and memory-efficient DataFrame library, primarily written in **Rust**. It is designed to handle **large datasets** very efficiently, both in terms of speed and memory usage. Unlike Pandas, which is mostly written in Python with some C extensions, Polars is a **columnar, multi-threaded engine** from the ground up. This means it can process data in parallel automatically, making it much faster for computations on large datasets.

Some key features of Polars:

* **Columnar storage:** Polars stores data column-wise (like Apache Arrow). This allows very fast column-based operations and reduces memory usage compared to row-based storage.
* **Lazy evaluation:** You can build a chain of operations without immediately executing them. The computation is only done when you need the result. This is similar to SQL query optimization and allows for huge performance gains.
* **Multi-threading:** Polars automatically uses multiple CPU threads to perform operations in parallel. Pandas, on the other hand, is mostly single-threaded, which can slow things down on large datasets.
* **Memory efficiency:** Since it is written in Rust and designed for high performance, Polars can handle bigger datasets without running into memory issues that Pandas often encounters.

Although Polars and Pandas share many similar operations like filtering, grouping, aggregation, and joining, the **underlying engine, performance characteristics, and syntax for certain operations differ**. Polars also has seamless conversion functions to and from Pandas, so you can use both libraries if needed.

In short, Polars is a **modern alternative to Pandas**, optimized for speed and large-scale data, but it is not built on Pandas—it is entirely its own engine.

**Why Polars is faster than pandas?**

* Written in “Rust” (low-level, optimized performance).
* Uses “Apache Arrow” memory model (efficient columnar storage).
* “Lazy evaluation” supports (optimizes query plans).
* Automatic multi-threading for many operations.