Polars vs. Dask: Analyzing Large Datasets

-Shivam Pawar

Polars

Key Features of Polars

- Optimized for **parallel processing** using multi-threading.
- **Lazy evaluation**: Deferred execution until results are needed.
- Uses a **columnar memory layout** for faster computation.

Execution Model:

- Combines operations intelligently using query optimization.
- Executes in-memory when possible and streams data for large files.

Dask

Key Features of Dask

- Parallel processing with task graphs.
- Processes data in chunks for **out-of-core computation**.
- Integrates well with other Python libraries like Pandas and NumPy.

Execution Model:

- Dask divides data into smaller **"partitions" (chunks)**.
- Operations are applied on each partition, then results are combined.

Execution Models of Dask and Polars

Dask Execution

- **Parallel Execution**: Splits the data into smaller **chunks** (partitions).
- Operations are applied to partitions in parallel.
- Trigger: `.compute()` to execute the task

Polars Execution

- **Lazy Execution**: Builds a **query plan** for operations and executes only when triggered.
- Combines tasks using query optimization.
- Trigger: `.collect()` to compute the result.

Pros and Cons

Dask

✅ Handles distributed computation on clusters.
✅ Integrates well with Pandas and other libraries.
❌ Chunked processing can add overhead for small datasets.

Polars

✅ Faster execution with lazy evaluation and multi-threading.
✅ Memory-efficient for large datasets.
❌ Does not support distributed clusters (yet).