Polars

Hermilo

4 de Septiembre 2025

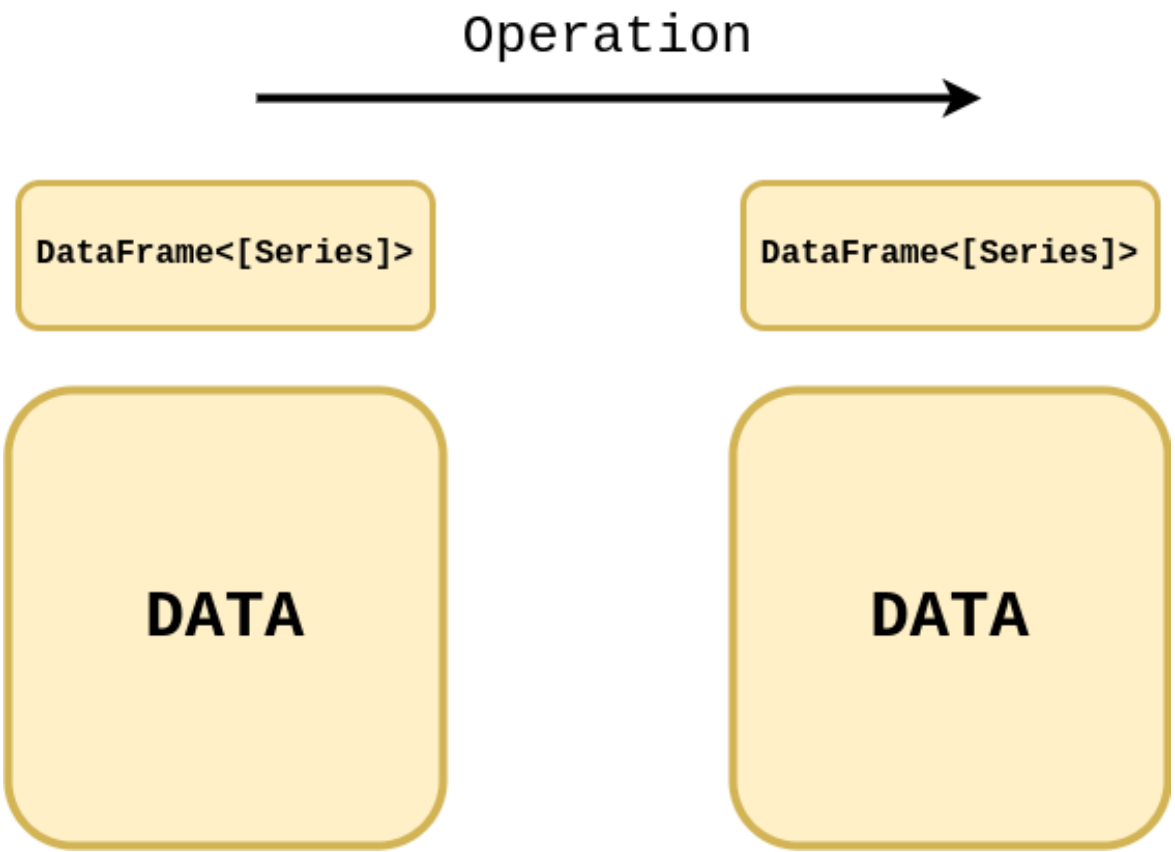
Polars : Query Engine for DataFrames

Polars : Query Engine for DataFrames

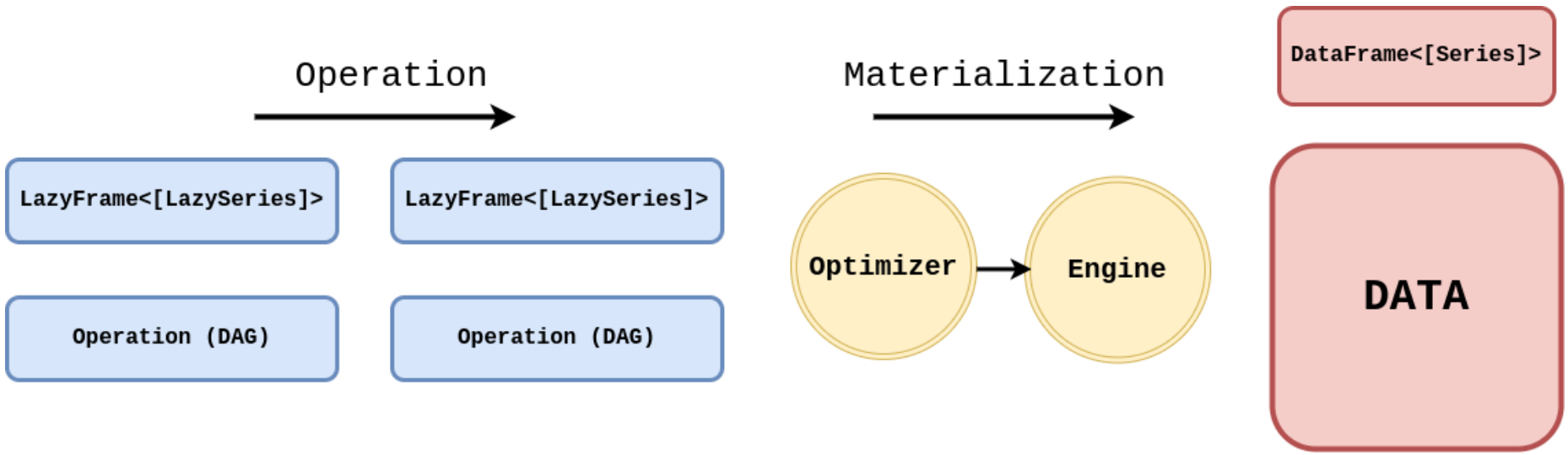
Polars es una biblioteca desarrollada en Rust que integra las siguientes características por diseño:

- **Velocidad** : Rust es un lenguaje de system programmig conocido por su desempeño y seguridad.
- **Paralelismo** : Aprovecha arquitecturas multicore he implementa algoritmos paralelos de work stealing.
- **Eficiencia de memoria**:
  - Polars utiliza evaluaciones lazy, lo que significa que una operación no es realizada hasta que esta es necesitada.
  - Las consultas pueden ser encadenadas y optimizadas antes de su ejecución, lo que se traduce en ejecuciones más eficientes de queries.
- **Almacenamiento eficiente de datos** : Polars utiliza Apache Arrow como modelo de almacenamiento en memoria. Es decir, utiliza un formato columnar del almacenamiento de datos, lo cual resulta más eficiente que el tradicional almacenamiento basado en filas (como el utilizado por Pandas).
- Diseñado para **out-of-core processing** (más grande que la RAM).

Polars : Query Engine for DataFrames



Polars : Query Engine for DataFrames



Aceleración impulsada por NVIDIA-RAPIDS

Consideraciones de diseño :

- Misma semática.
- Optimizador.
- Parte de la biblioteca de Polars (no es necesario `import cudf_polars as pl`)

```python
query = (
  transaction.group_by("cust_id").agg(
    pl.cum("amount")
  ).sort(by = "amount", descending = True)
  .head()
)

# Run on the CPU
result_cpu = query.collect()

# Run on the GPU
result_gpu = query.collect(engine="gpu")

# assert both result are equal
pl.testing.assert_frame_equal(result_cpu, result_gpu)
```

Optimizaciones

Vectorización ofrecida por NumPy:

- SI : Performance de C.
- NO : Optimizaciones.

```python
import polars as pl

q = (
    pl.scan_csv('../datos/flights.csv')
    .select(['MONTH', 'ORIGIN_AIRPORT','DESTINATION_AIRPORT'])
    .filter(
        (pl.col('MONTH') == 5) &
        (pl.col('ORIGIN_AIRPORT') == 'SFO') &
        (pl.col('DESTINATION_AIRPORT') == 'SEA'))
)
%time
df = q.collect(no_optimization=True)
CPU times: user 3 µs, sys: 0 ns, total: 3 µs
Wall time: 5.48 µs
%time
df = q.collect()
CPU times: user 5 µs, sys: 0 ns, total: 5 µs
Wall time: 15.7 µs
```

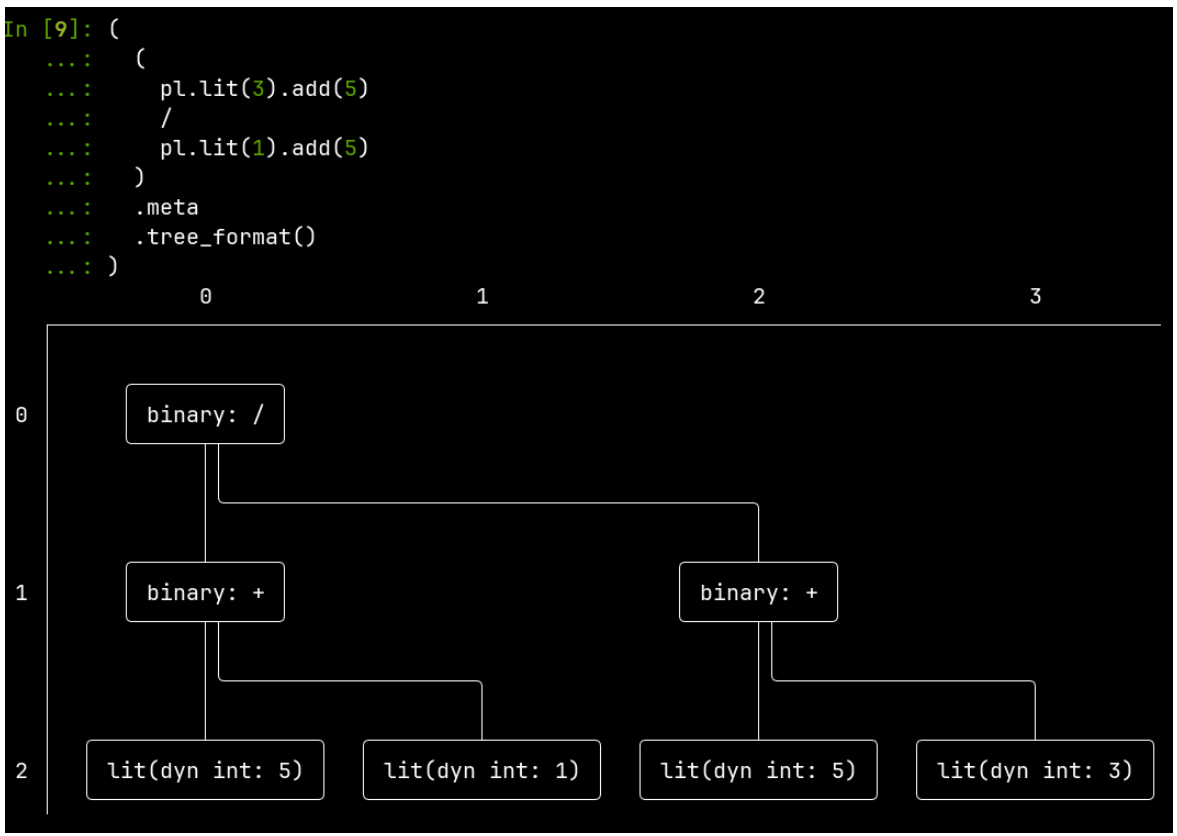
Polars Expressions

¿Qué es una expresión?

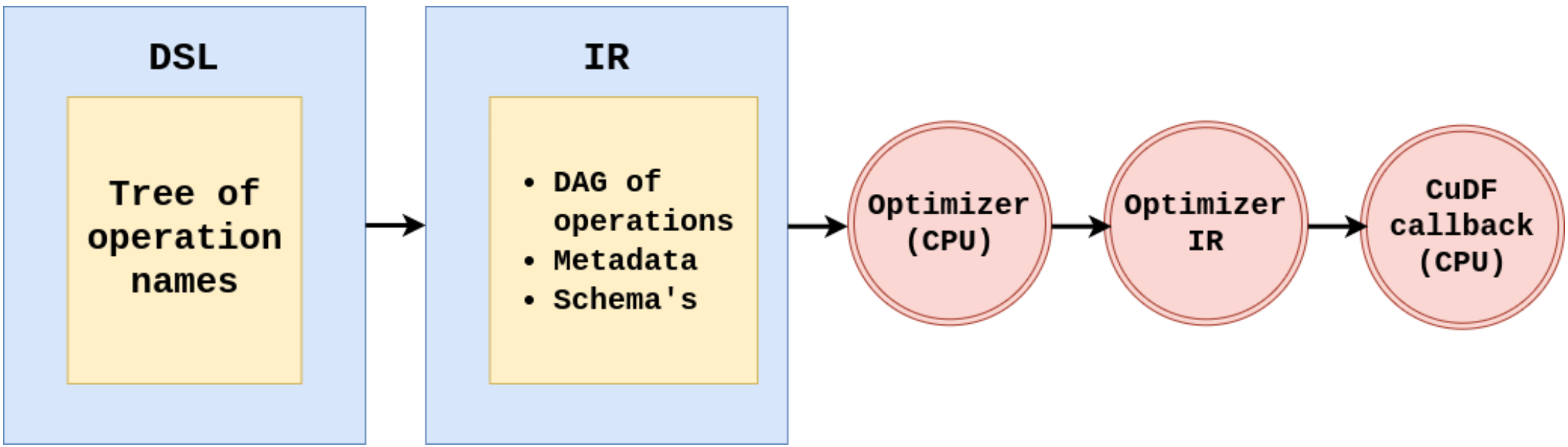
Una expresión es un árbol de operaciones que describen cómo construir ona o más Series.

Polars Expressions

```
In [9]: (
   ...:   (
   ...:     pl.lit(3).add(5)
   ...:     /
   ...:     pl.lit(1).add(5)
   ...:   )
   ...:   .meta
   ...:   .tree_format()
   ...: )
```



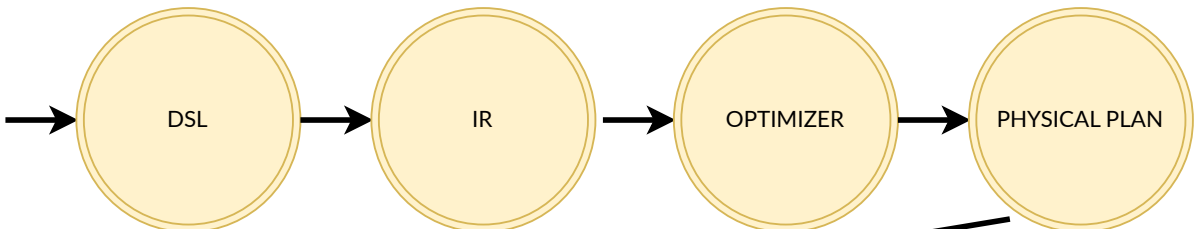
Fase Lógica


DSL
Tree of operation names
IR
• DAG of operations
• Metadata
• Schema's
Optimizer (CPU)
Optimizer IR
CuDF callback (CPU)

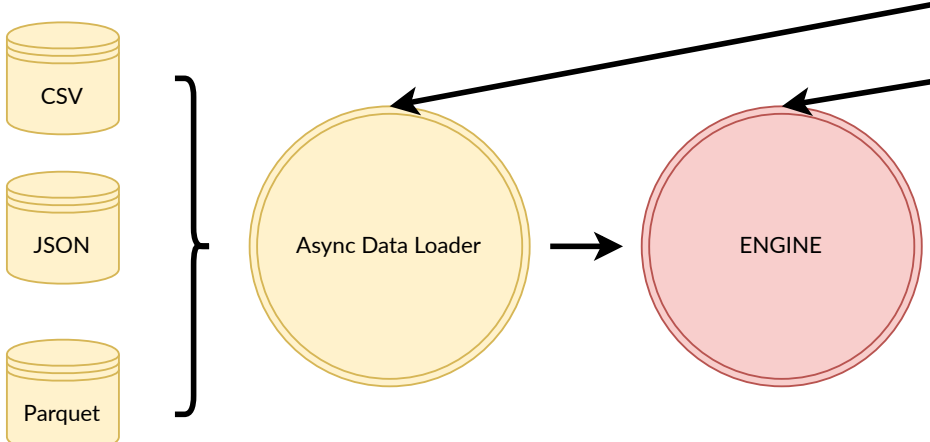
Roadmap de un query

Logical

```python
(
  pl.scan_parquet("my_file.parquet")
  .filter(pl.col("dogs")=="Retriever")
  .select("dogs", "origins","weight")
  .head(10)
  .collect(engine="gpu")
)
```



Physical



Paralelismo : Work Stealing