Apache Spark,
un ecosistema poliedrico

NTT DATA

Trusted Global Innovator

Presentiamoci

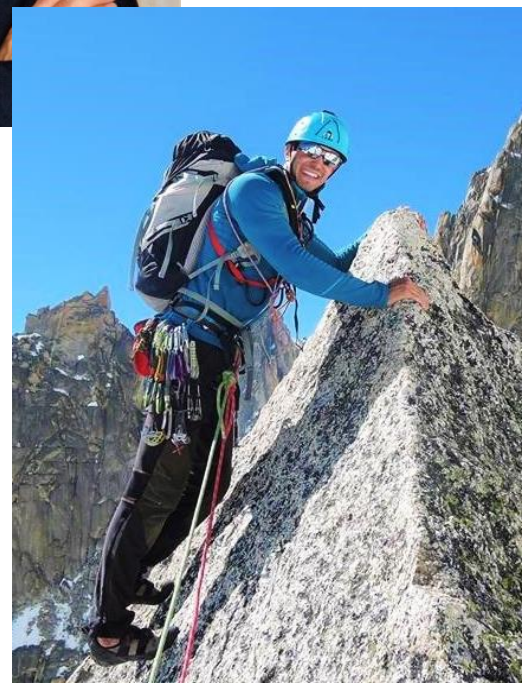
Andrea Picasso Ratto Senior Big Data Engineer @**NTTDATA**

Background accademico:

- Software Engineering con specializzazione in Big Data Architecture e Machine Learning

Focus Tecnologico:

- ETL batch e streaming processing basato su tecnologie **Spark, Flink e Kafka**
- **Natural Language Processing** verticalizzato su temi di Sentiment Analysis, Topic modelling e Text Summarization
- Design di **Architetture Big Data** a microservizi



Nel tempo libero:

- Amante dei viaggi in giro per il mondo
- Accanito alpinista del weekend

NTT DATA ITALIA: Data Intelligence Team

Market Areas

- FINANCE
- ENERGY
- MANUFACTURING & RETAIL
- PUBLIC SECTOR
- TELCO & MEDIA

Team

MORE Than 300

internal resources + 100 external

World Wide Resources
NTTDATA Global Trusted Innovator

Solution Areas

BUSINESS INTELLIGENCE

BUSINESS ANALYTICS

BIG DATA

DATA VISUALIZATION

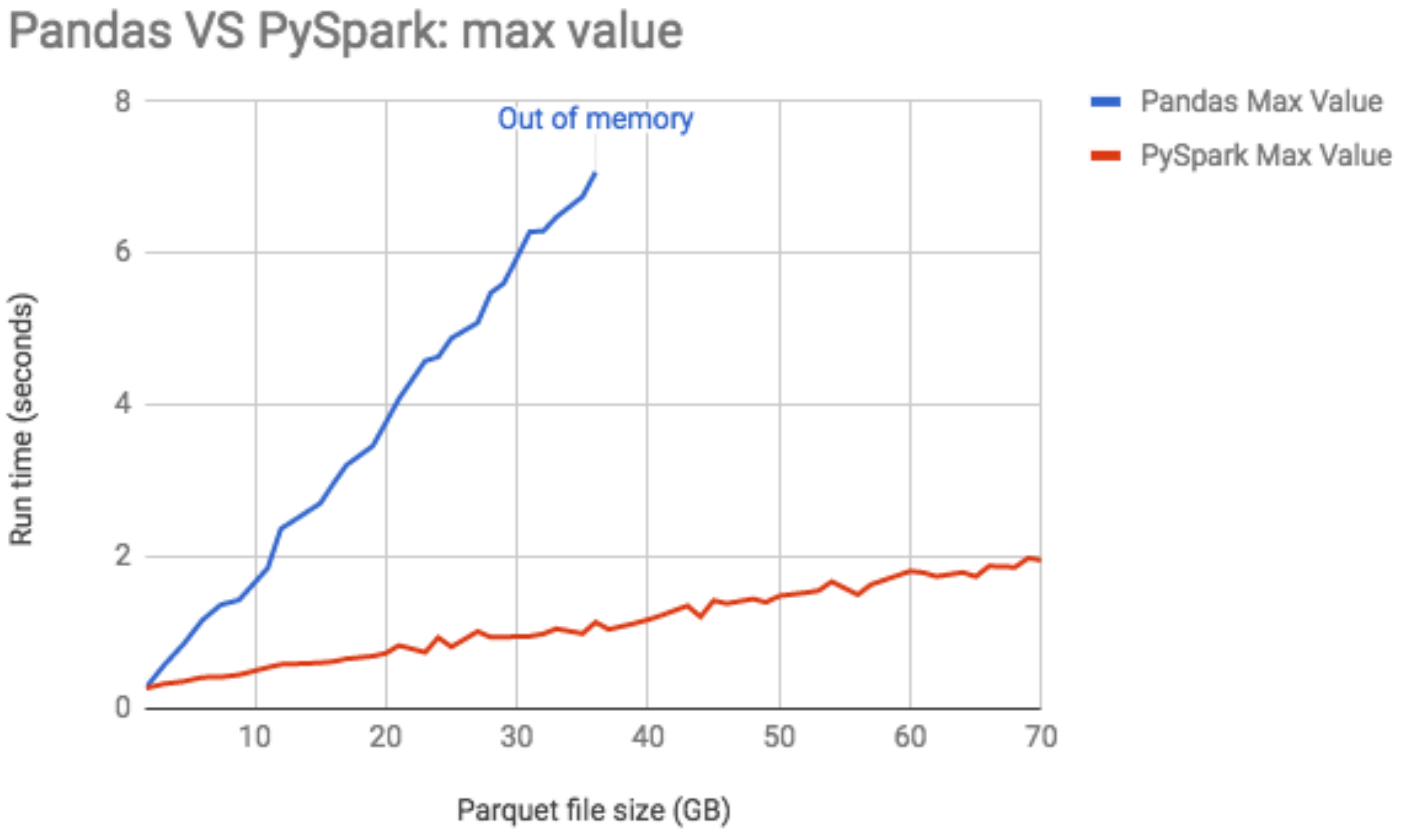
MRKT CAMPAIGN SERVICES

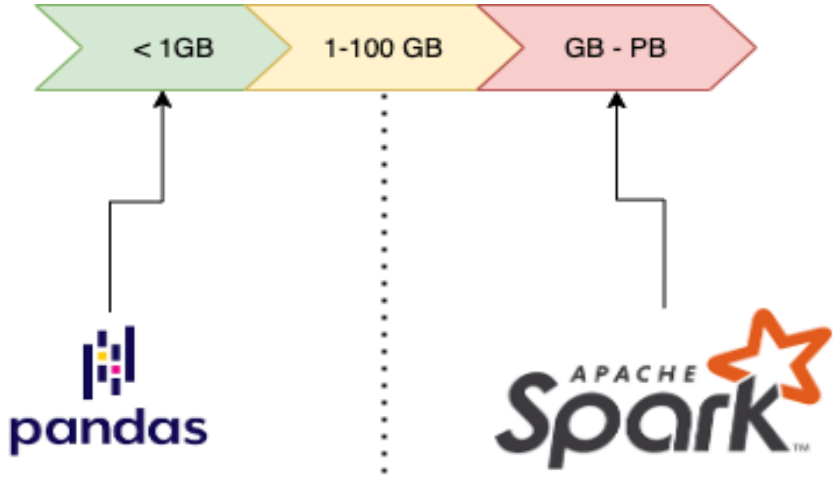
NTT DATA

```
import pandas as pd
```

```
from pyspark.sql.functions import *
```

QUANTO CONTA LA SCALABILITA‘





	Local Development	Scalable for Production
Tabular Manipulations	pandas $y_i t = \beta' x_{it} + \mu_i + \epsilon_{it}$	APACHE Spark
ML Algorithms	scikit learn	APACHE Spark ML

PARLIAMO DI NUMERI

DATO GPS

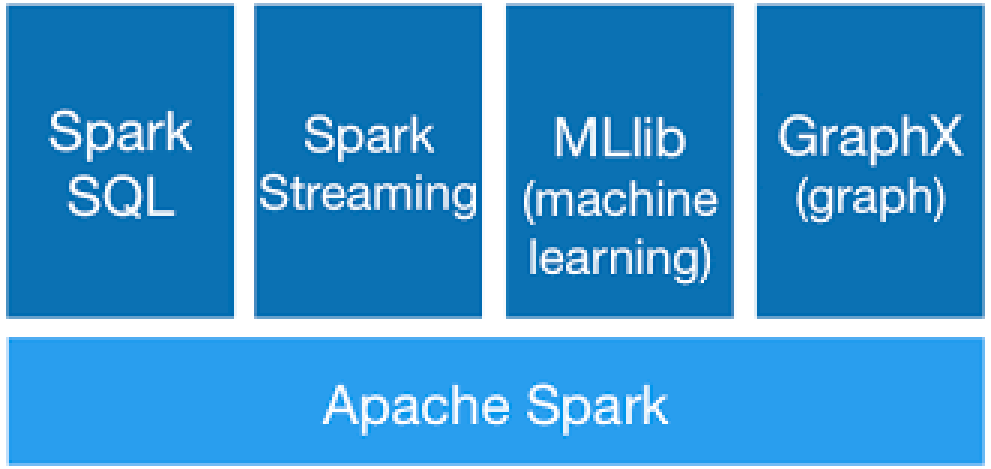
- 340 Milioni di record al giorno
- 2,7 Miliardi di record alla settimana

DATO DATALAKE & REPORTS

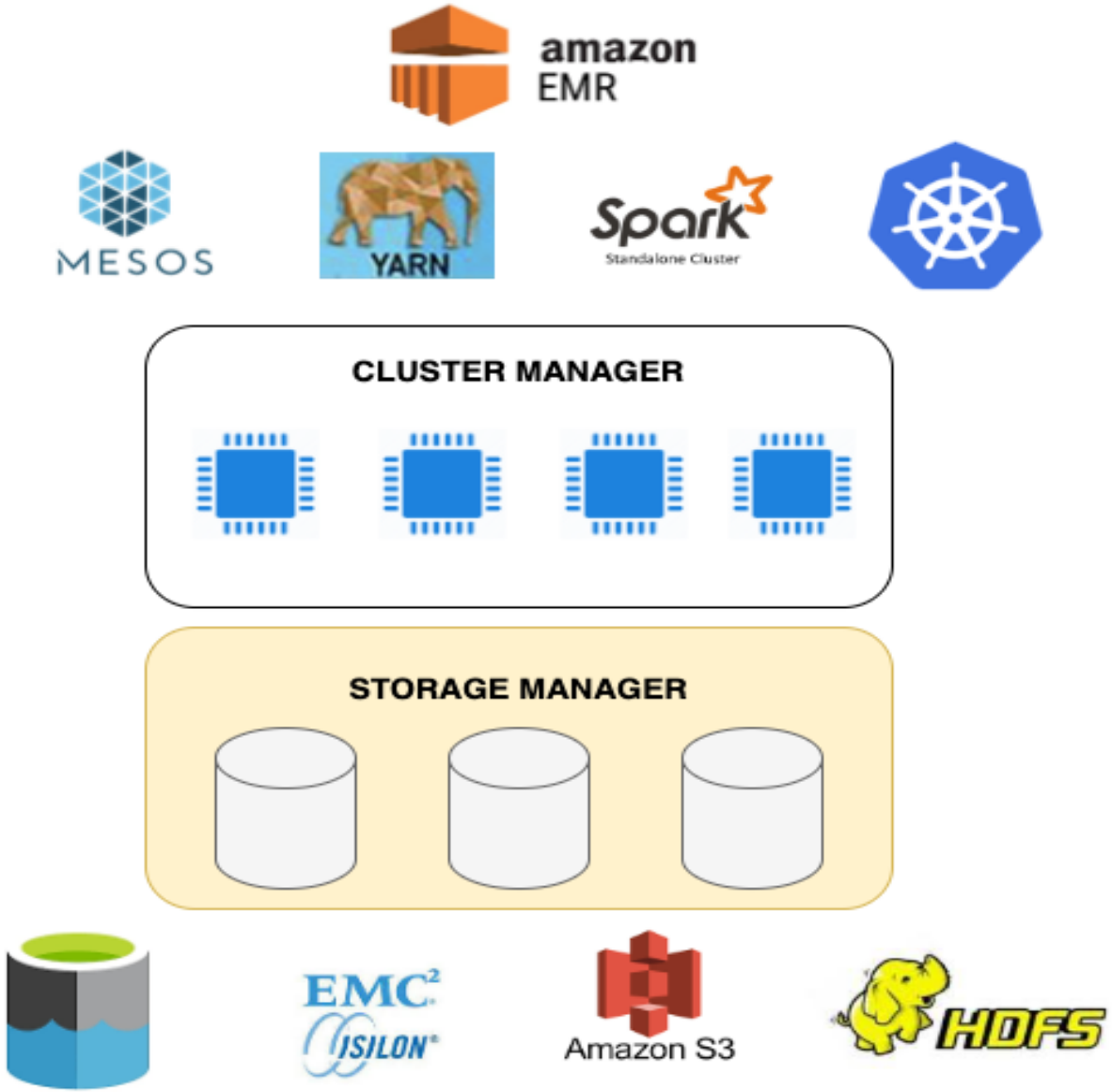
- 1400 tabelle
- 500 Milioni di record in batch notturno
- 16 Miliardi di record scansionati
- 120 vCPU ~ 600 GB Ram

APACHE SPARK

Databrick Framework per **M**assive **P**arallel **P**rocessing



APACHE SPARK ARCHITECTURE



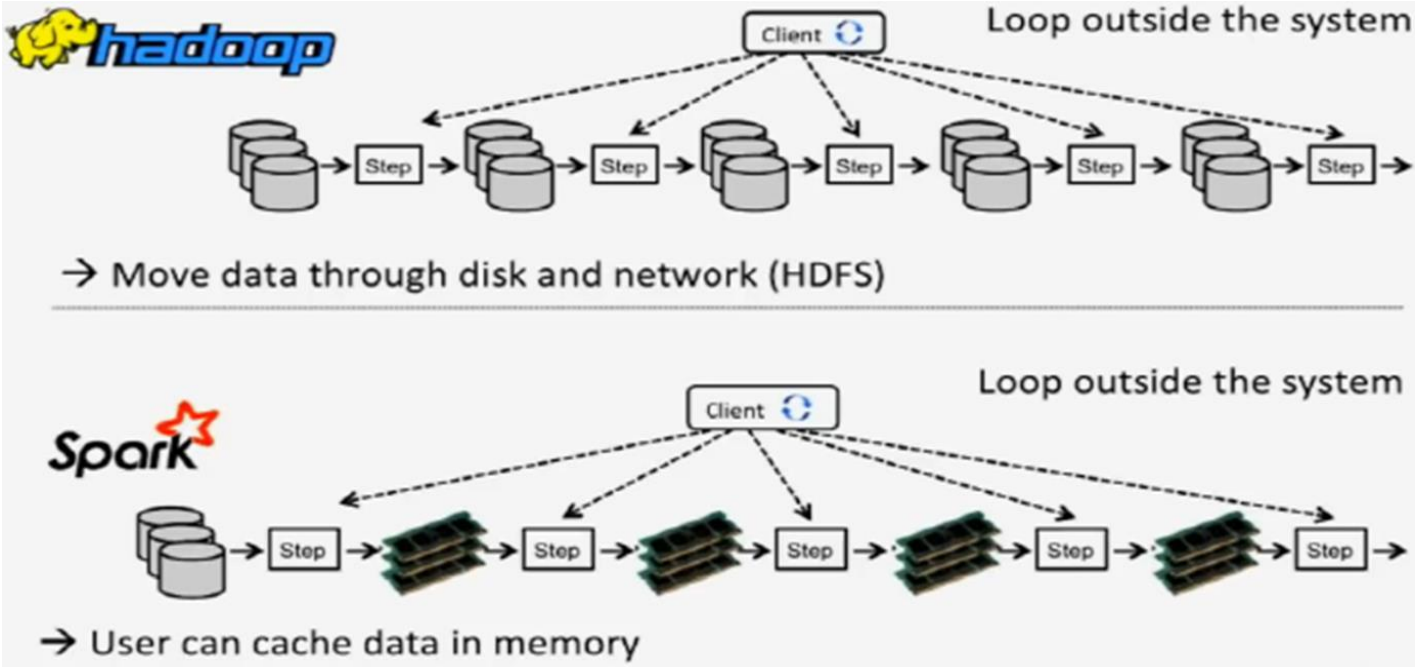
SPARK: TUTTO IN MEMORY

High level API over Map-Reduce

Resilient Distributed Dataset – **RDD**

2 CONCETTI FONDAMENTALI:

- **Lazy** evaluation
- **Narrow** vs. **wide** transformations



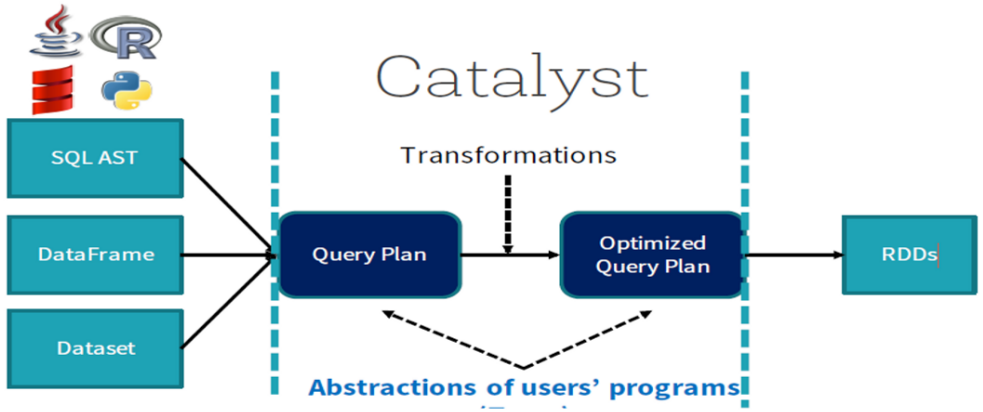
Mostafaeipour, Ali, et al. *The Journal of Supercomputing* (2020):

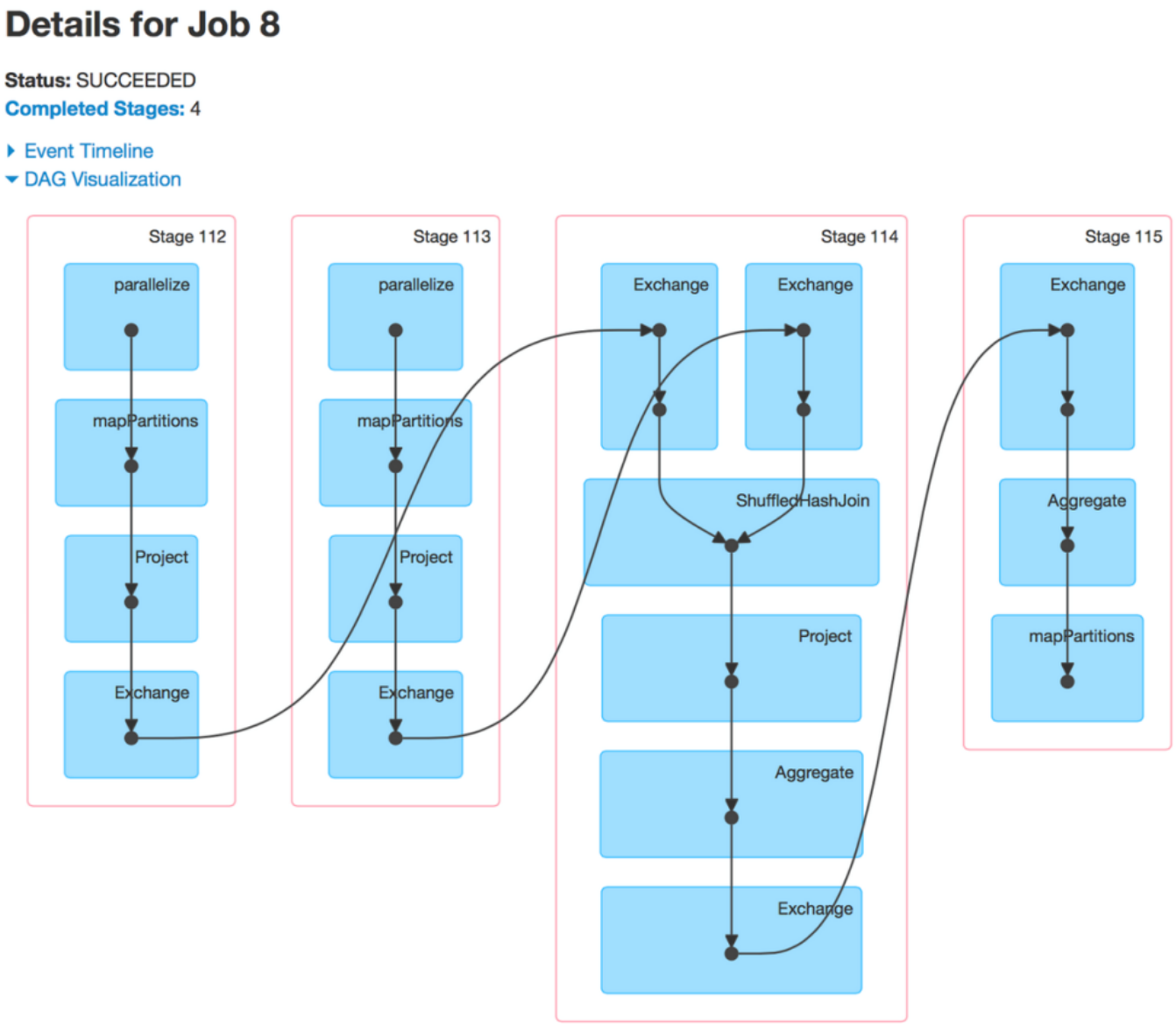
LAZY EVALUATION

Transformation

Action

DAG – Catalyst

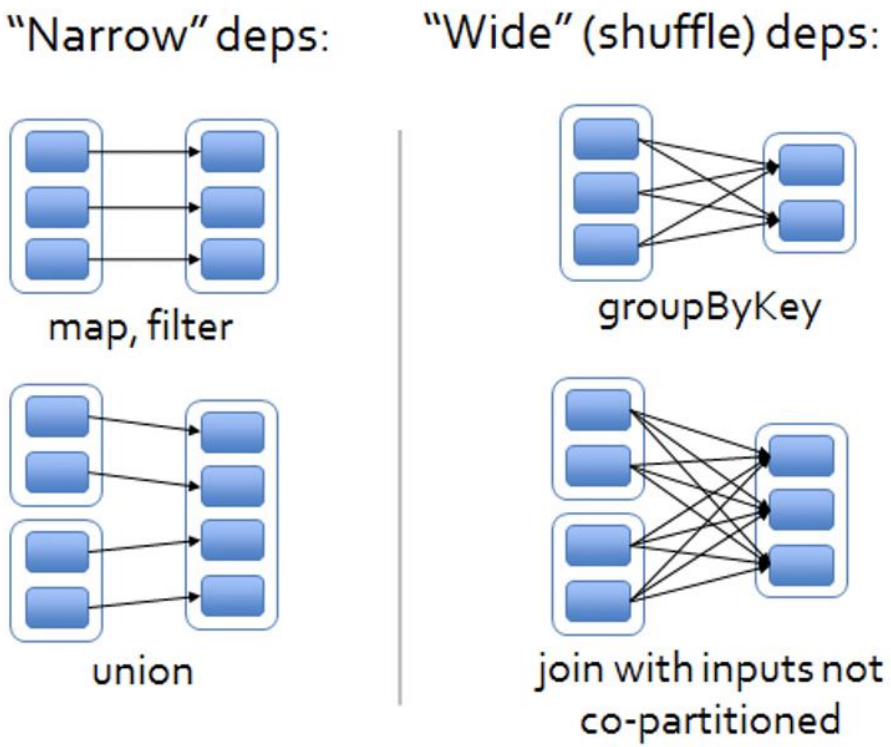




NARROW VS WIDE TRANSFORMATION

Memory hierarchy params:

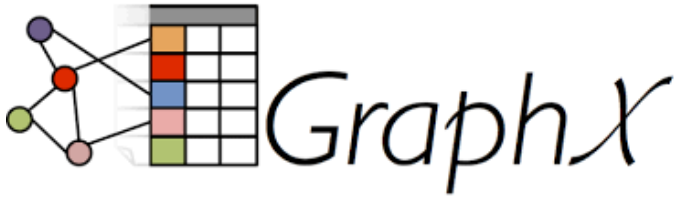
- Size
- Latency



Computer Action	Avg Latency	Normalized Human Time
3GhzCPU Clock cycle 3Ghz	0.3 ns	1 s
Level 1 cache access	0.9 ns	3 s
Level 2 cache access	2.8 ns	9 s
Level 3 cache access	12.9 ns	43 s
RAM access	70 - 100ns	3.5 to 5.5 min
NVMe SSD I/O	7-150 µs	2 hrs to 2 days
Rotational disk I/O	1-10 ms	11 days to 4 mos
Internet: SF to NYC	40 ms	1.2 years
Internet: SF to Australia	183 ms	6 years

Zaharia, Matei, et al. "Resilient distributed datasets:
A fault-tolerant abstraction for in-memory cluster computing.".

SPARK ECOSYSTEM



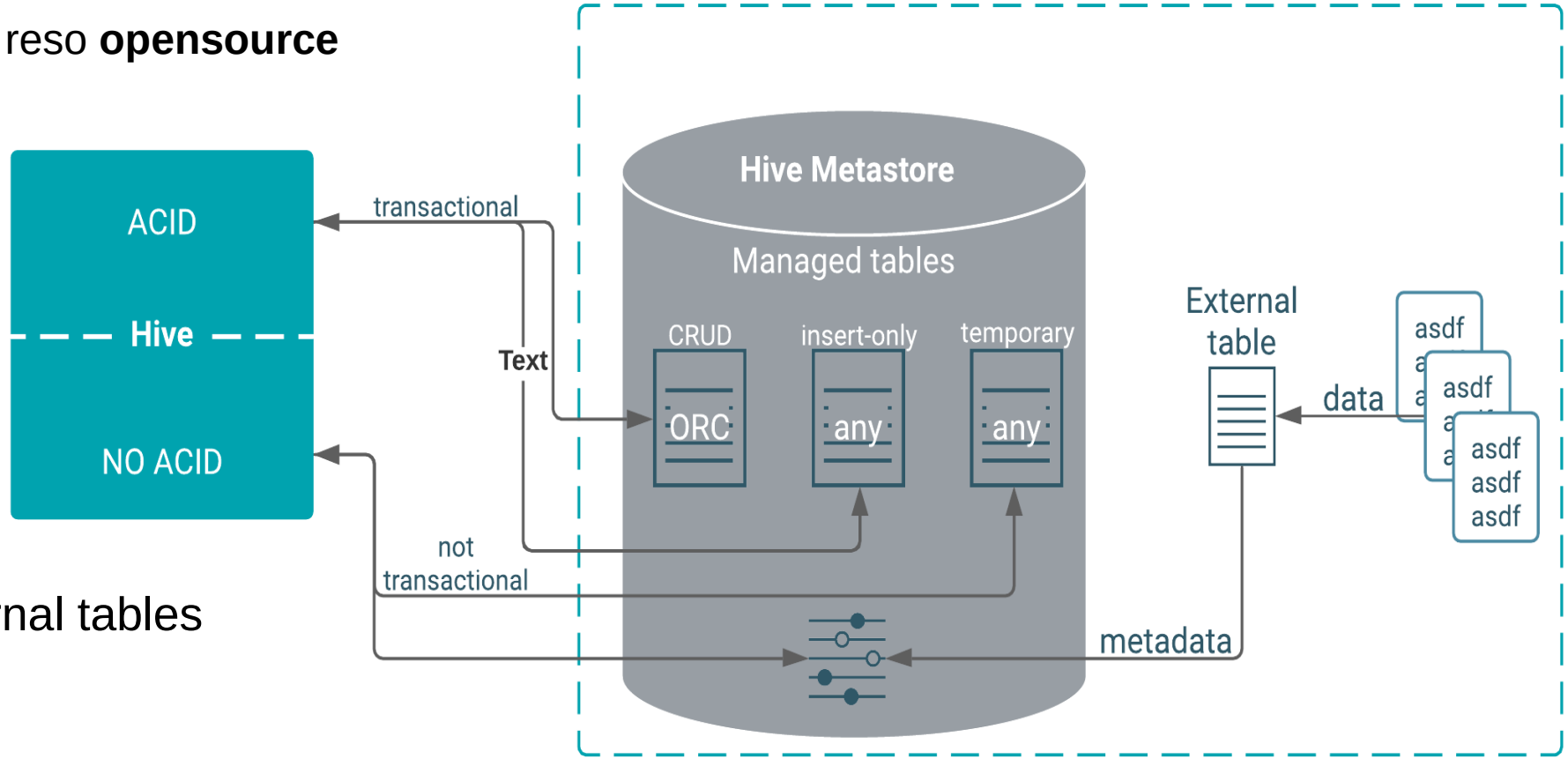
HIVE

Progetto nato in Facebook reso **opensource**

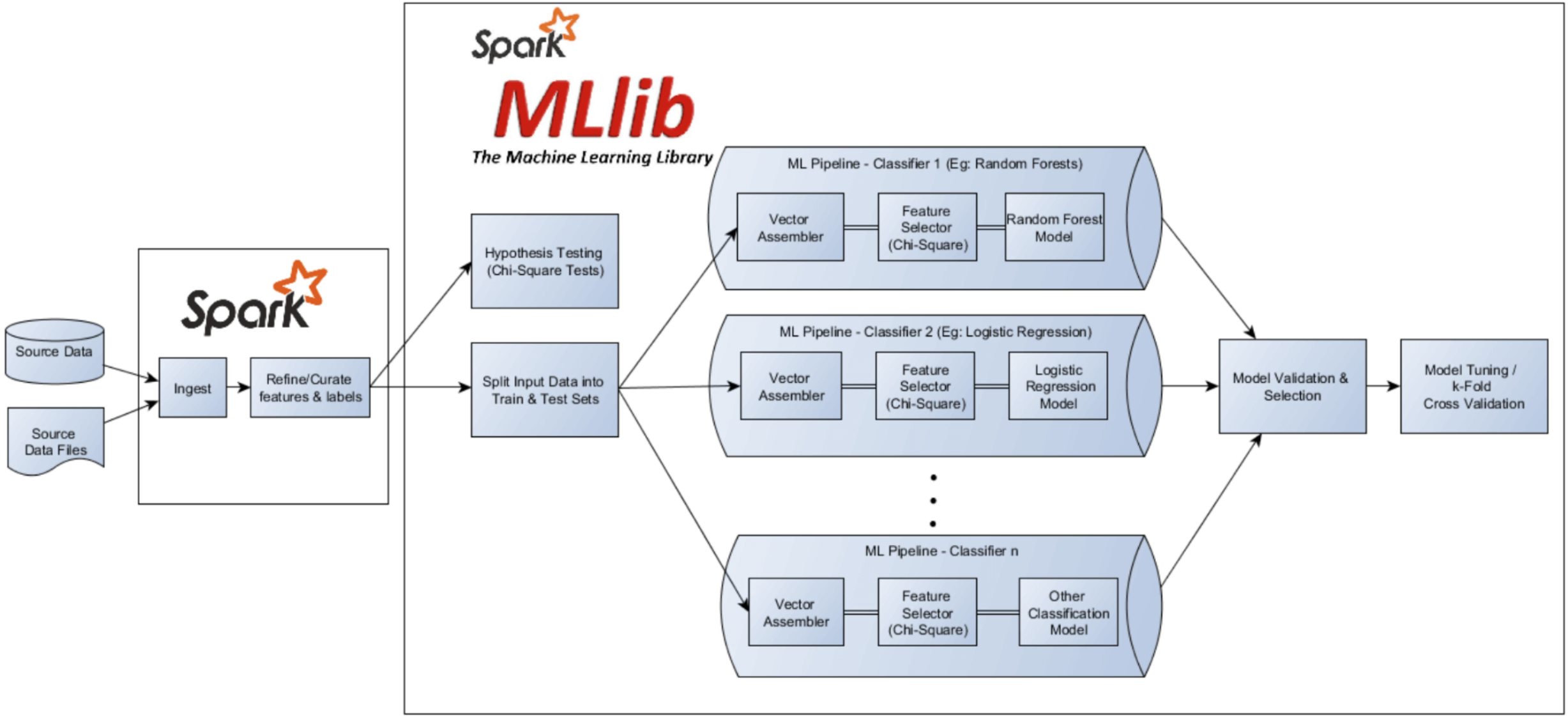
Metadata store

JDBC connector

Managed tables Vs External tables



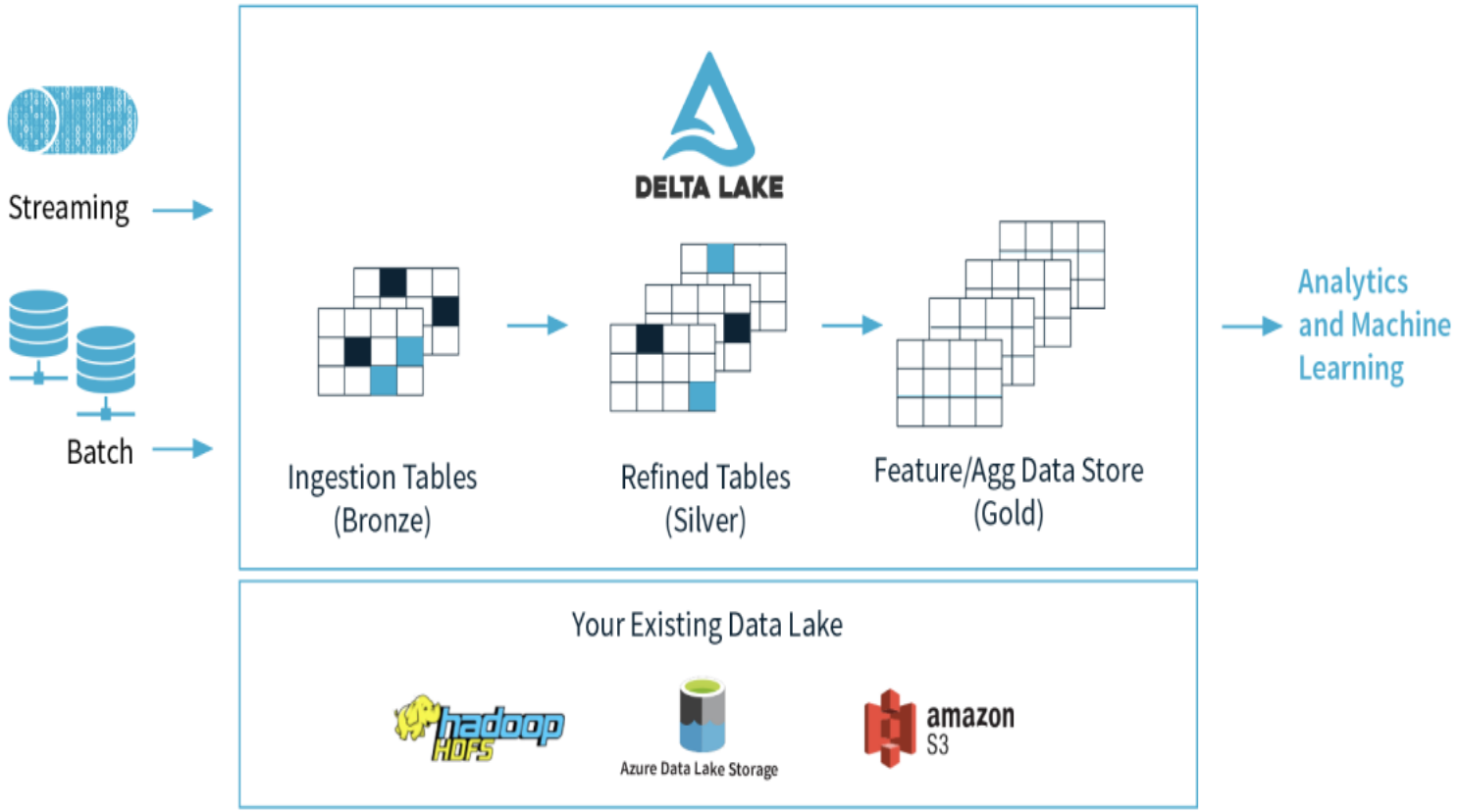
SPARK MLlib



DELTA LAKE

E' un progetto Databricks reso **opensource**.

- **ACID transaction** su file parquet: DeltaTables
- Interfaccia unificata **batch e streaming**
- **Schema evolution** grazie alla schema validation anche on write
- **Time travel** e Historical table



COME SEMBRA

- Codice semplice
- Interfaccia pulita
- Variabili parlanti

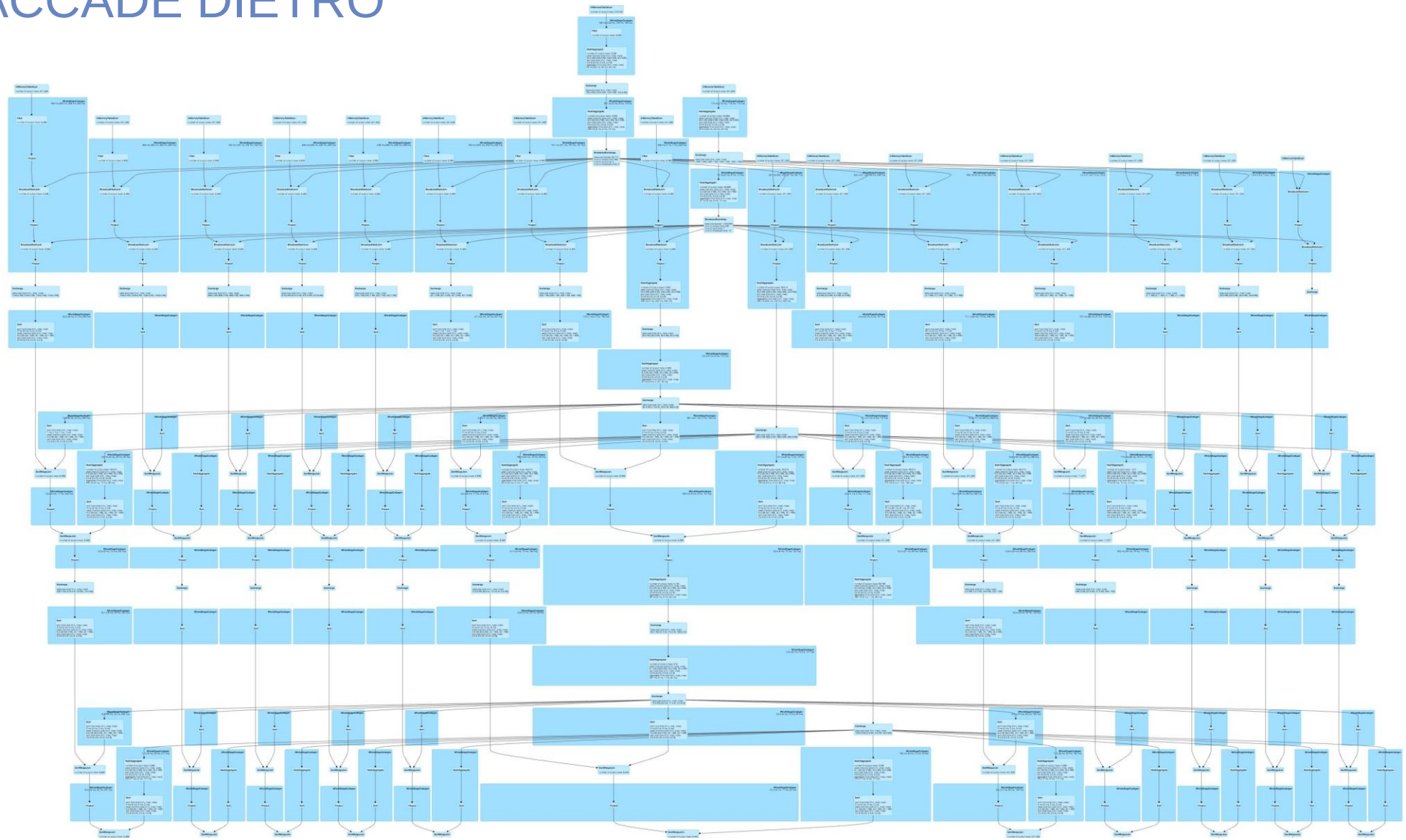
```scala
import spark.implicits._

val peopleDF = spark.read.json("examples/src/main/resources/people.json")

// DataFrames can be saved as Parquet files, maintaining the schema information
peopleDF.write.parquet("people.parquet")
val parquetFileDF = spark.read.parquet("people.parquet")

// Parquet files can also be used to create a temporary view and then used in SQL statements
parquetFileDF.createOrReplaceTempView("parquetFile")
val namesDF = spark.sql("SELECT name FROM parquetFile WHERE age BETWEEN 13 AND 19")
namesDF.map(attributes => "Name: " + attributes(0)).show()
// +------------+
// |       value|
// +------------+
// |Name: Justin|
// +------------+
```

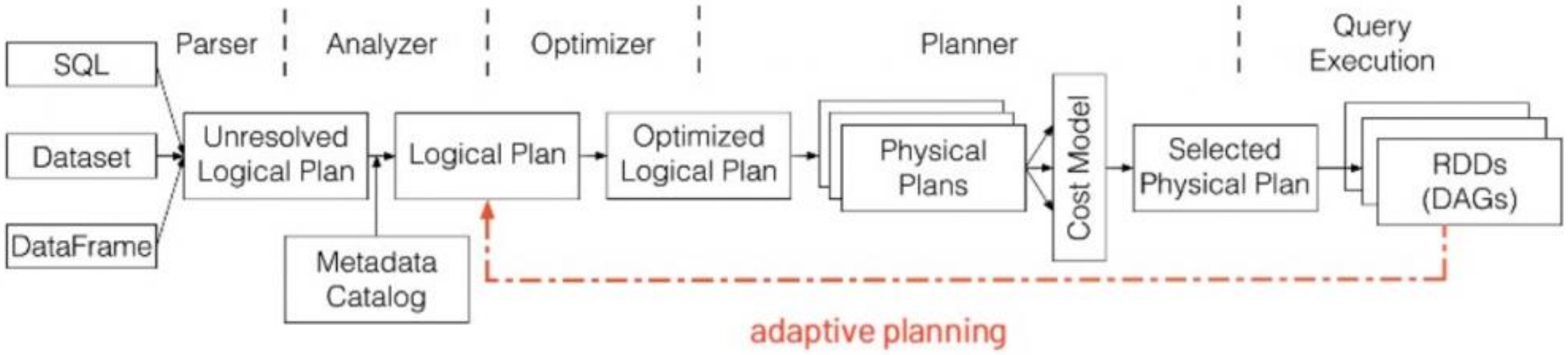
COSA ACCADE DIETRO



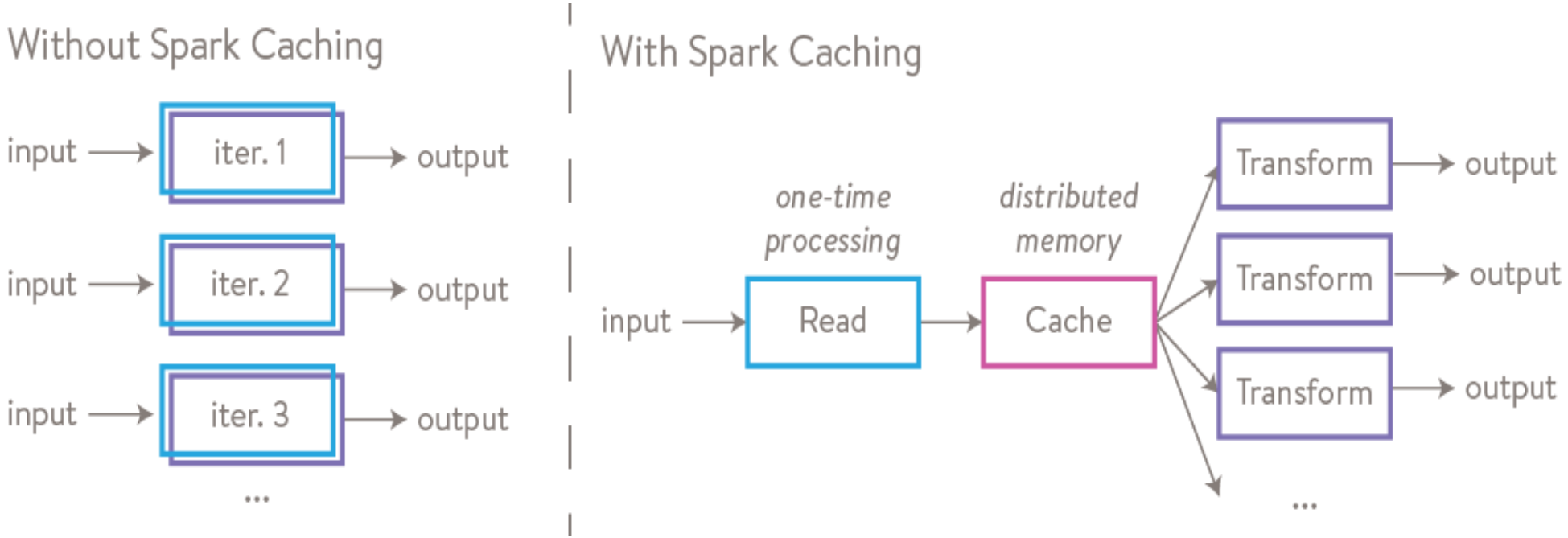
CATALYST OPTIMIZER

Ottimizzazione workloads:

- Caching
- Join strategies
- Partition Pruning



CACHING



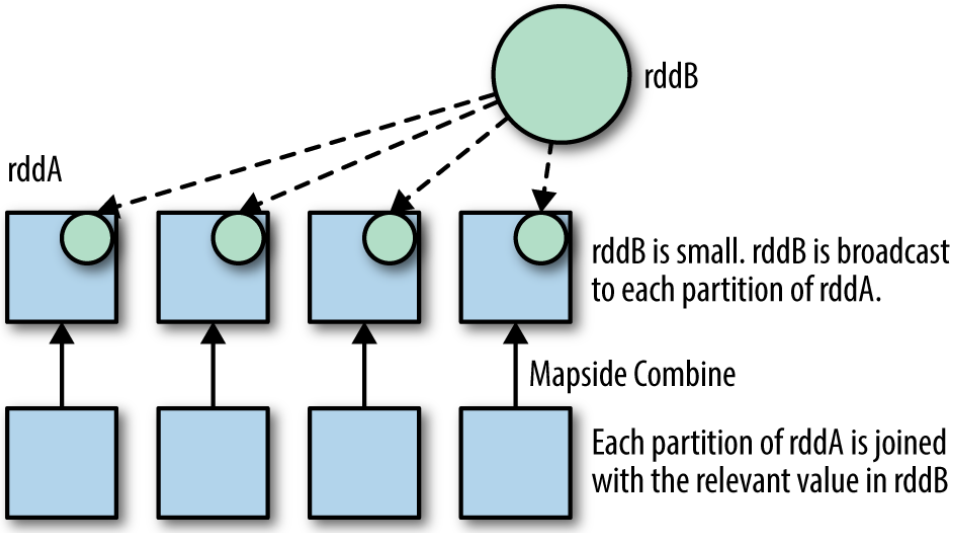
STRATEGIE DI JOIN

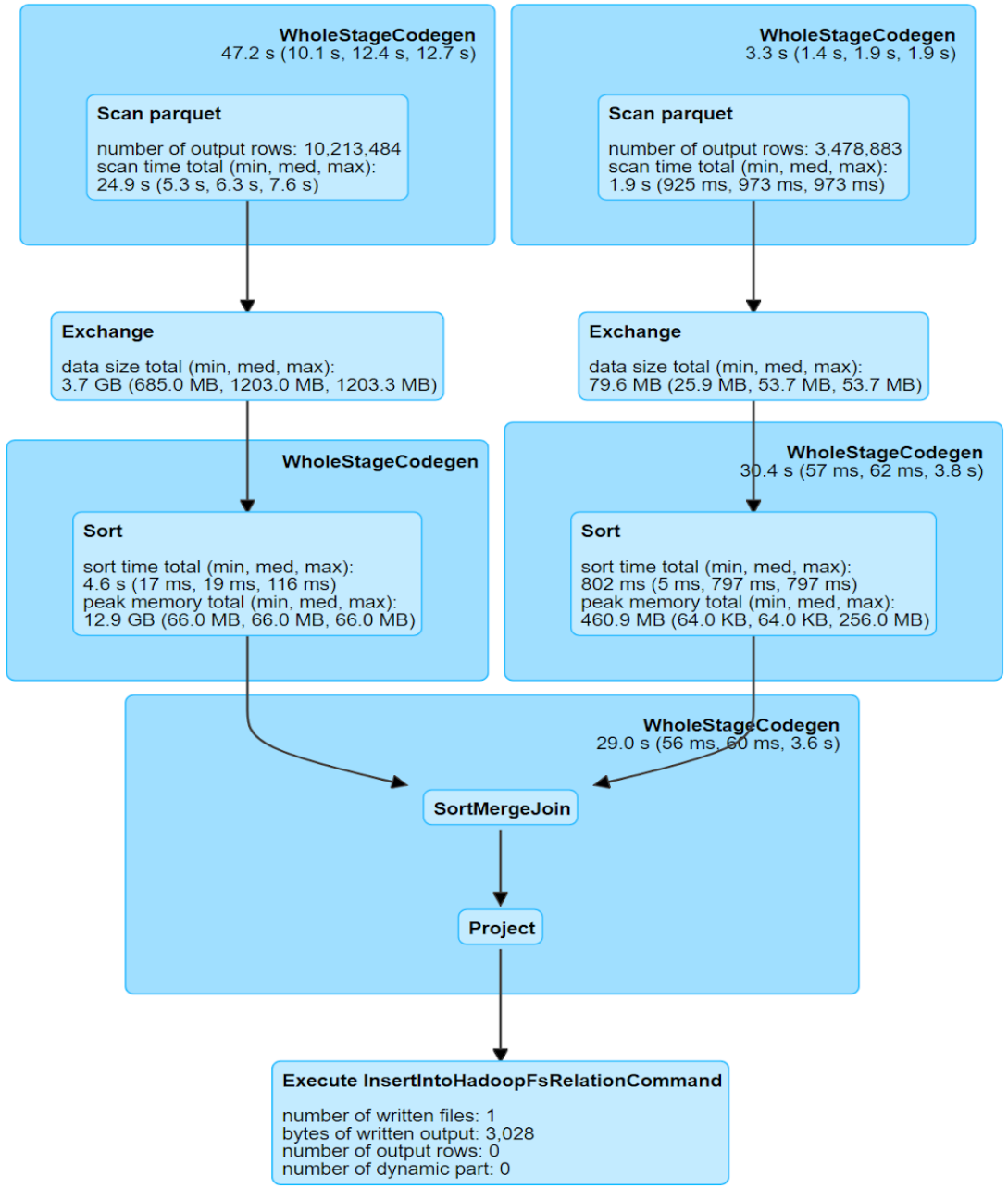
BROADCASTJOIN

SORT MERGE JOIN

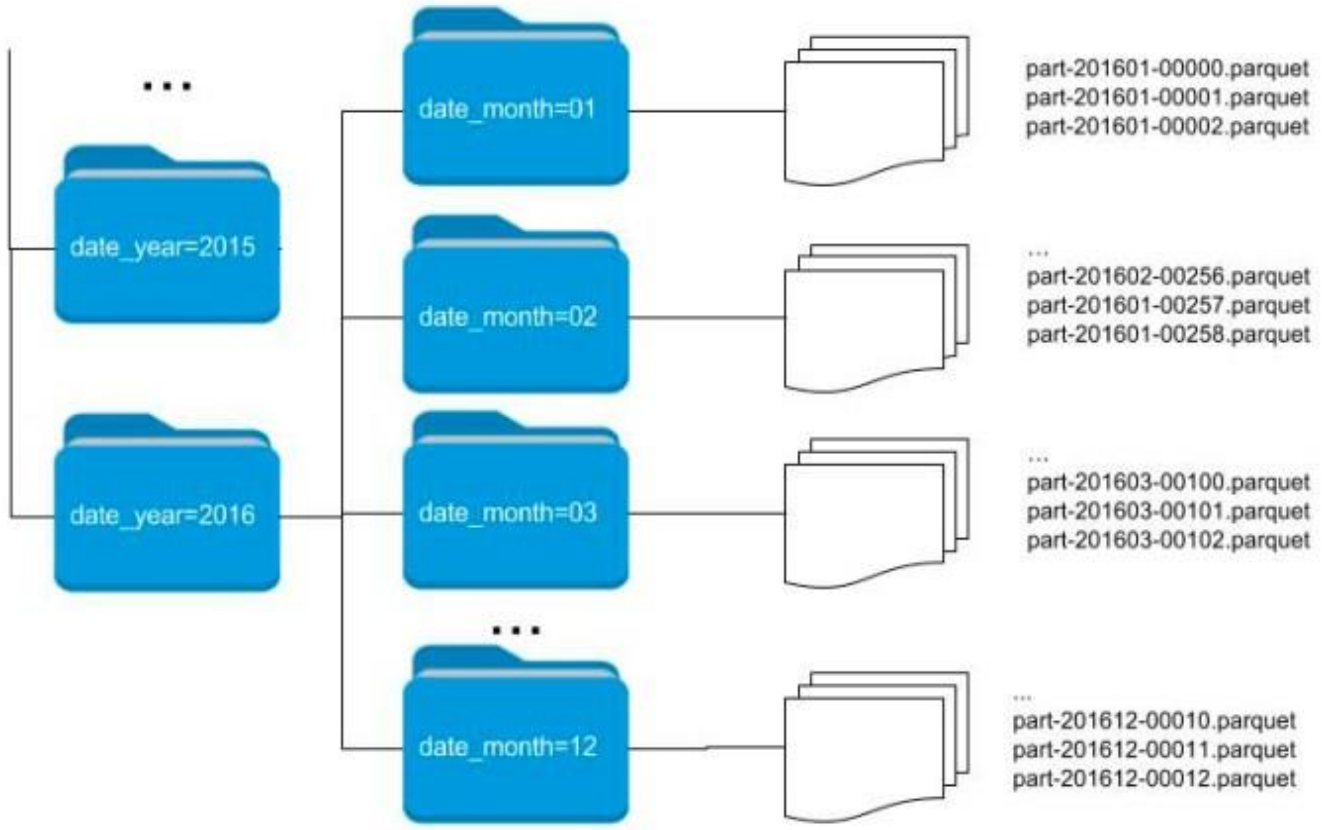
SHUFFLE HASH JOIN

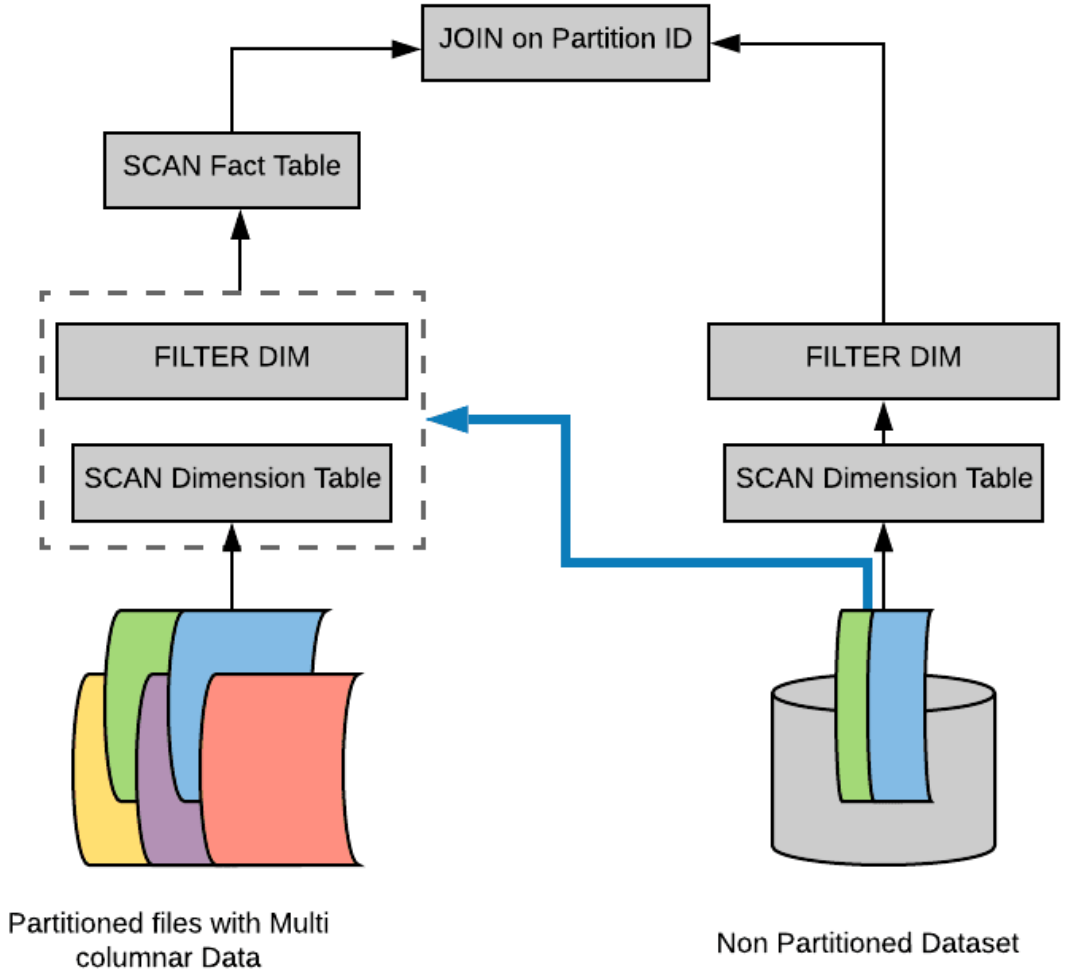
NESTED LOOP





PARTITION PRUNING





```
import pandas as pd
```

```
from pyspark.sql.functions import *
```

PANDAS VS SPARK

Dataset < 1GB

Single machine

Dataset TB o PB

Cluster di macchine

PANDAS VS KOALAS VS SPARK

Be immediately productive with Spark, with no learning curve

KOALAS

Koalas

Astrazione di Spark API

Interoperabilità con Pandas API

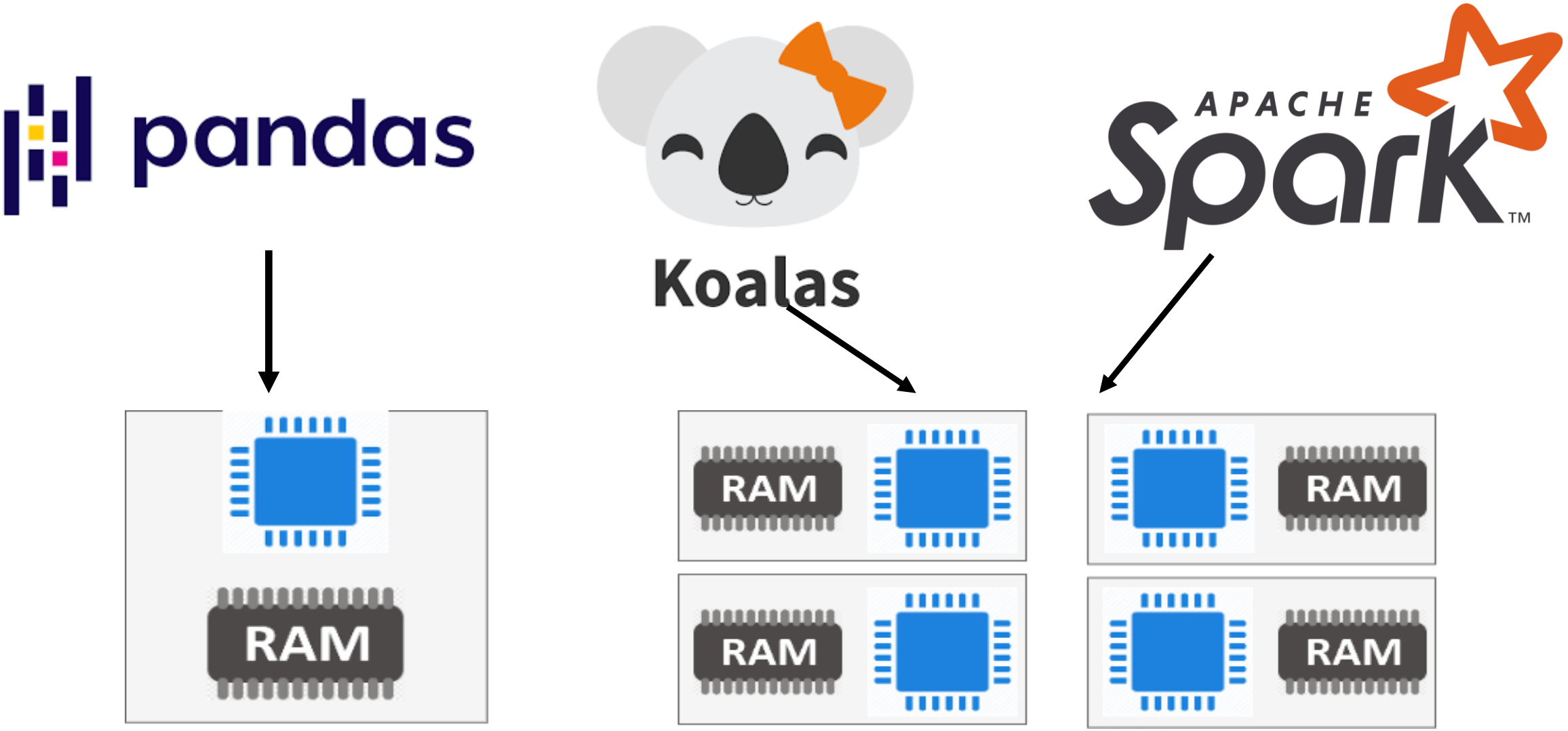
Dato organizzato in RDD

Lazy evaluation -> execution plan

```
import pandas as pd
data = pd.read_csv("fire_department_calls_sf_clean.csv", header=0)
display(pd.get_dummies(data))
```

```
import databricks.koalas as ks
data = ks.read_csv("fire_department_calls_sf_clean.csv", header=0)
display(ks.get_dummies(data))
```

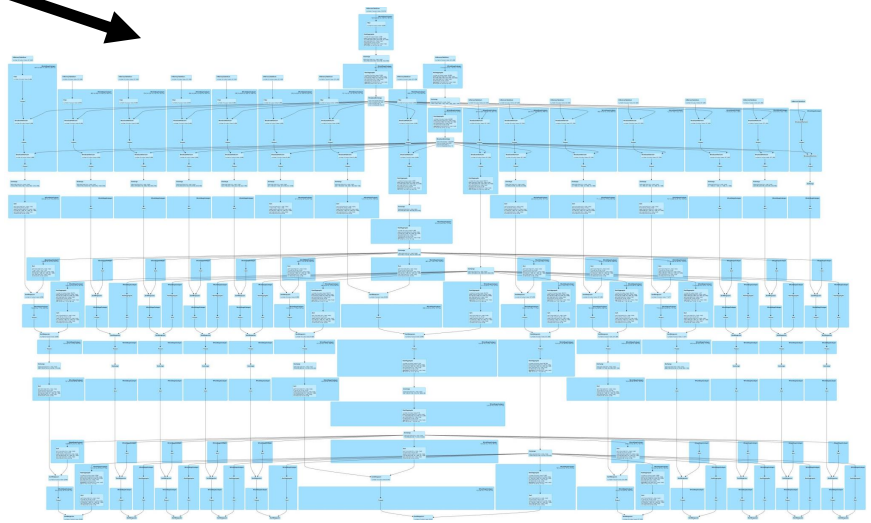
MEMORY



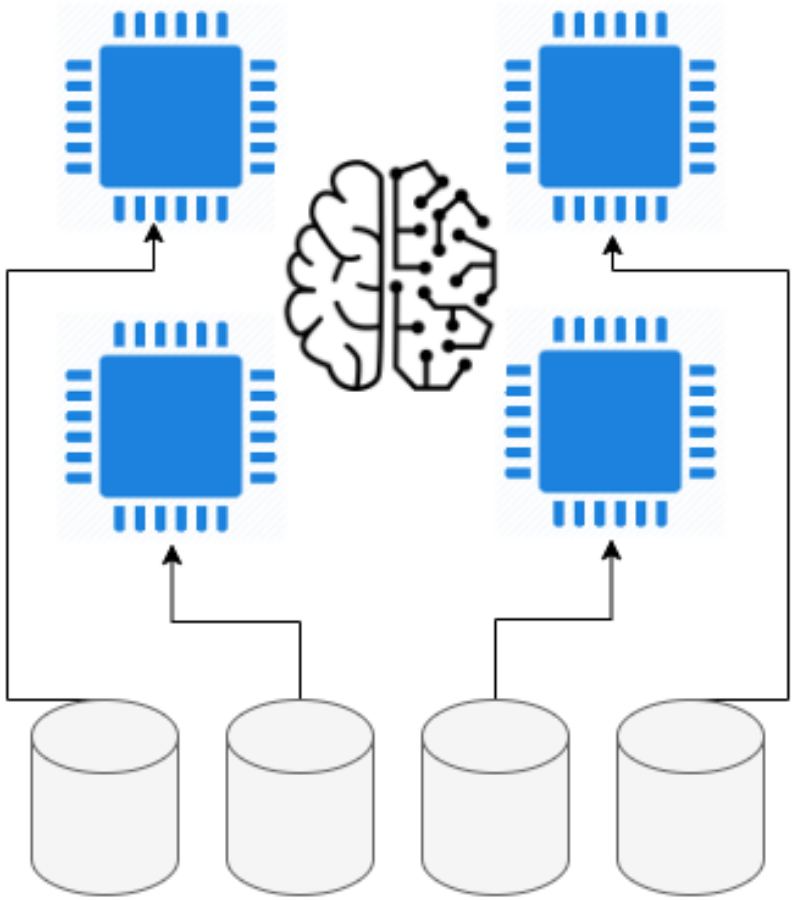
ATTENZIONE A COSA ACCADE DIETRO!

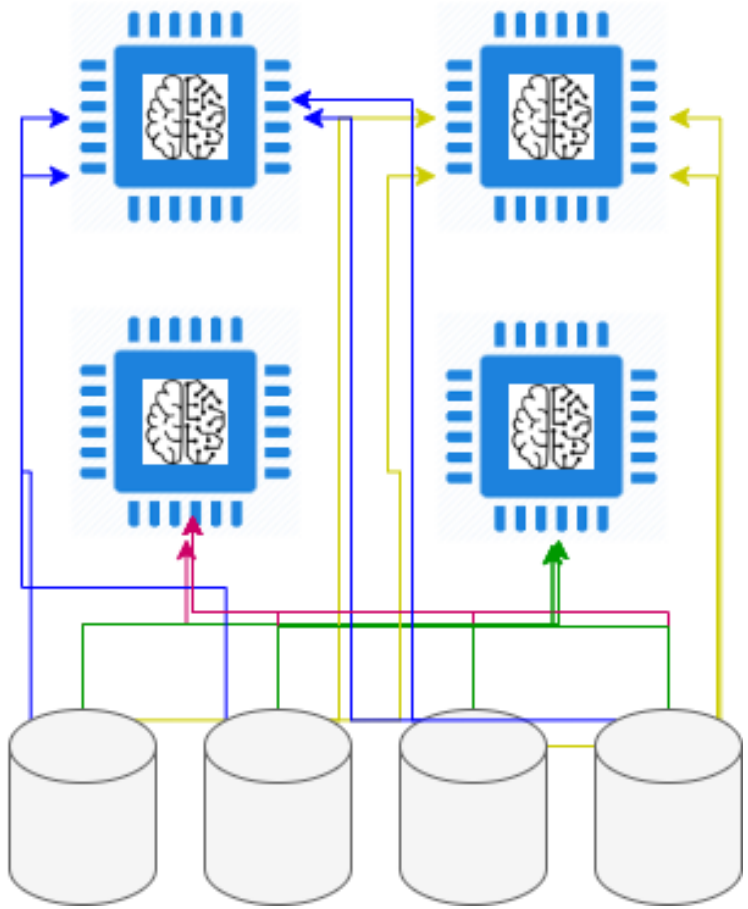
Koalas

```
import databricks.koalas as ks
data = ks.read_csv("fire_department_calls_sf_clean.csv", header=0)
display(ks.get_dummies(data))
```



DATA VS MODEL PARALLELISM





SPARK DISTRIBUTED MODELS

Classificazione e Regressione

MLP integrato + integrazione Tensorflow da

Alberi decisionli e GBT

Clustering

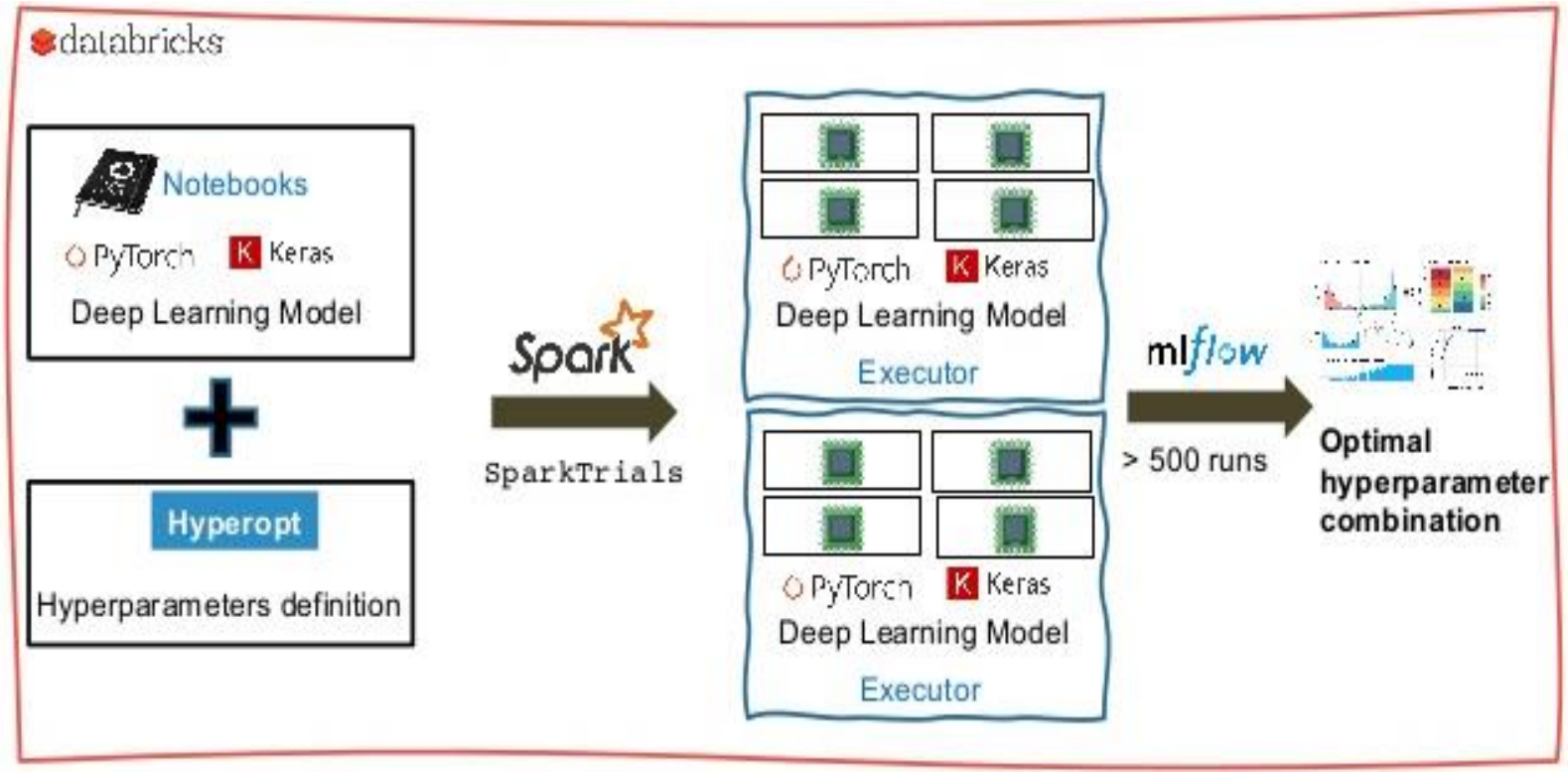
Librerie utili:

SparkNLP

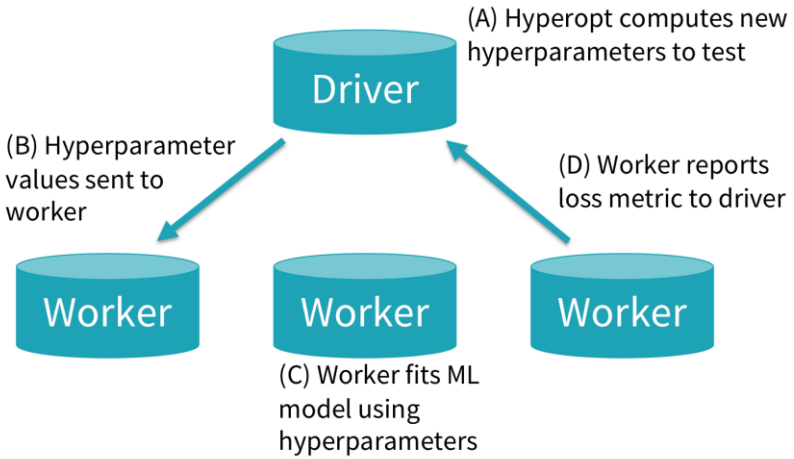
Azure mmlspark

Microsoft Machine Learning for Apache Spark

HYPEROPT

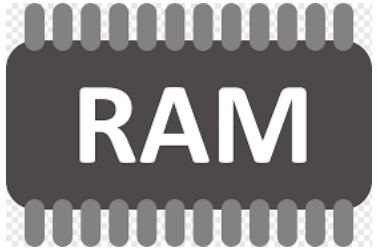


mlflow



NTT DATA

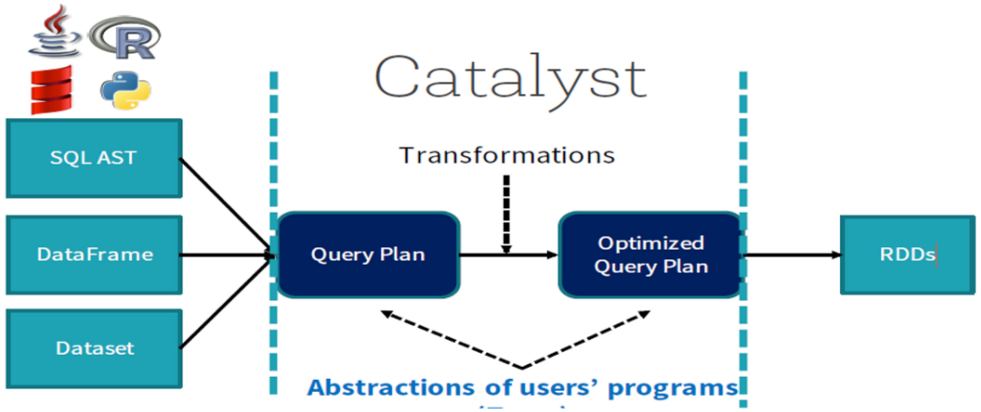
TAKEAWAYS

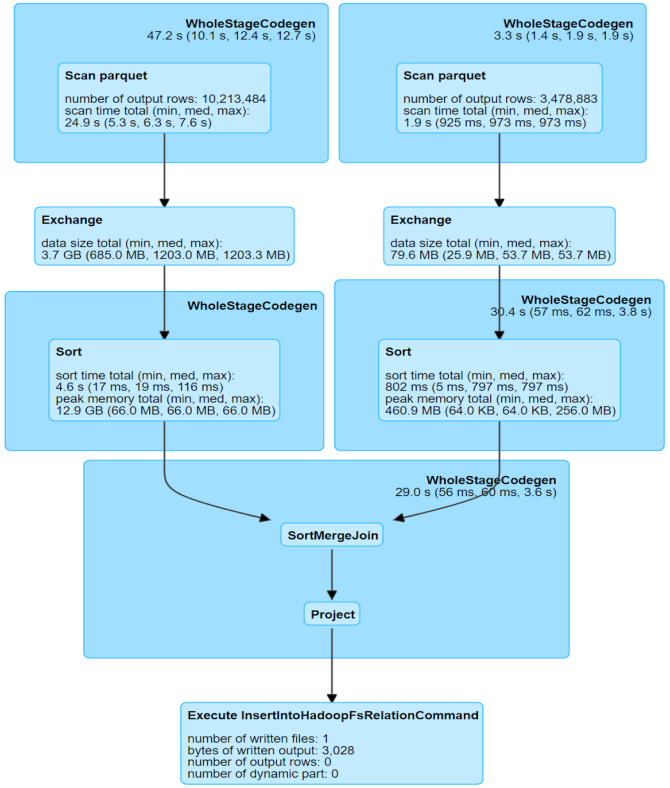


Scrivere codice Spark è facile, farlo funzionare è un altra cosa

DAG e lazy evaluation sono curciali

Modelli ML limitati ma in evoluzione





NTT DATA

NTT DATA

Global IT Innovator

andrea.picassoratto@nttdata.com