Data DAy México

Analiza datos como los gurús

Ciencia de Datos con Apache Spark y Optimus

Favio Vázquez

Data Scientist

March 15th, 2018

@faviovaz

https://github.com/faviovazquez

https://www.linkedin.com/in/faviovazquez/

Esquema

1. Introducción a Apache Spark y Optimus

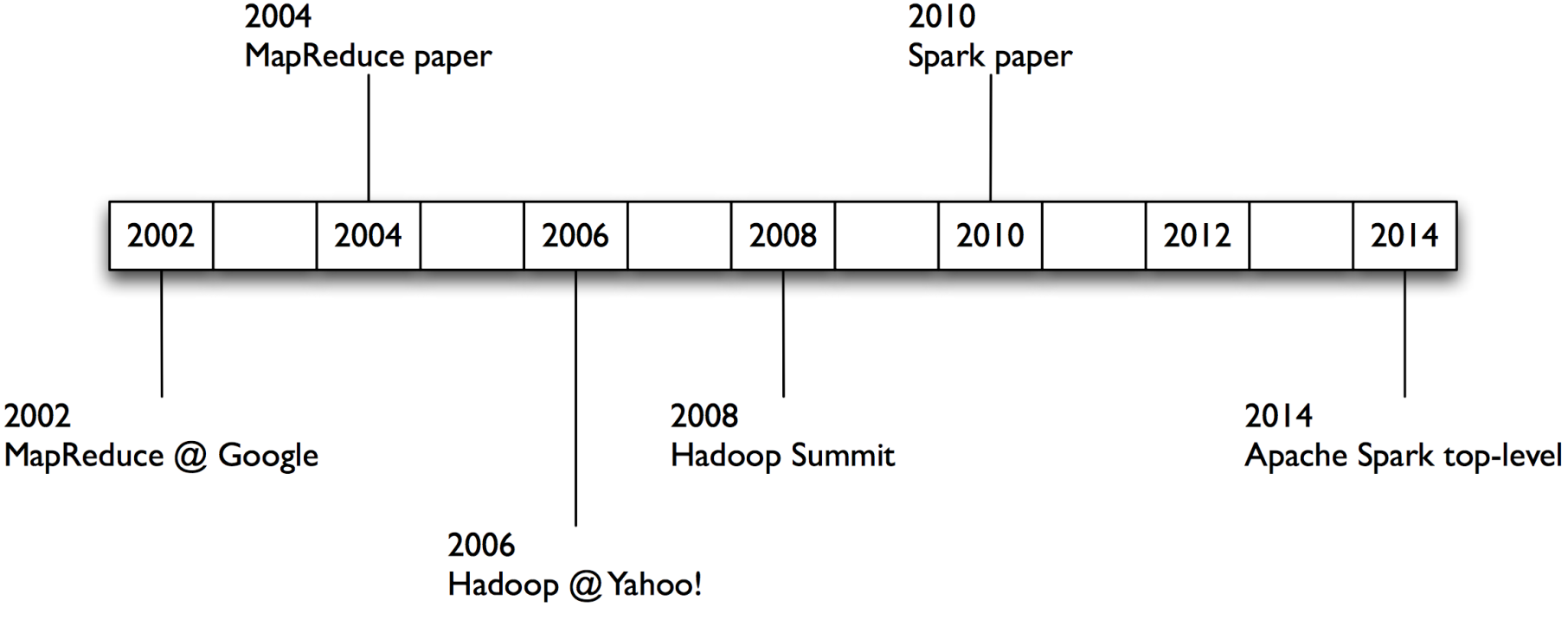
- Fundamentos de Spark
- Fundamentos de Optimus

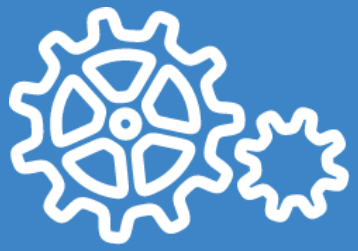
2. Data Science con Spark y Optimus

- Limpieza, Exploración y Preparación de Datos con Spark y Optimus
- Introducción a Machine Learning con Spark y Optimus.

http://spark.apache.org/

Historia


2004
MapReduce paper
2010
Spark paper
2002
2004
2006
2008
2010
2012
2014
2002
MapReduce @ Google
2006
Hadoop @ Yahoo!
2008
Hadoop Summit
2014
Apache Spark top-level



2004 – Google

MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat

research.google.com/archive/mapreduce.html

2006 – Apache

Hadoop, originating from the Nutch Project

Doug Cutting **research.yahoo.com/files/cutting.pdf**

2008 – Yahoo

web scale search indexing

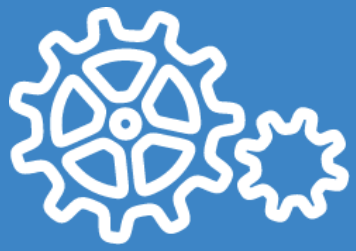
Hadoop Summit, HUG, etc.

developer.yahoo.com/hadoop/

2009 – Amazon AWS

Elastic MapReduce

Hadoop modified for EC2/S3, plus support for Hive, Pig, Cascading, etc. **aws.amazon.com/elasticmapreduce/**

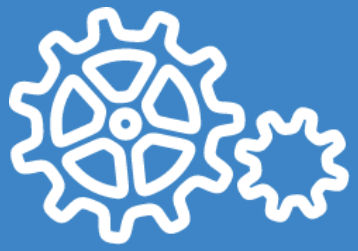


¿Qué es complicado en Big Data?

La combinación compleja de hilos de ejecución, sistemas de almacenamiento y modos de trabajo.

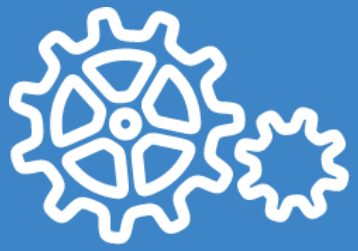
- ETL, agregaciones, machine learning, streaming, etc.

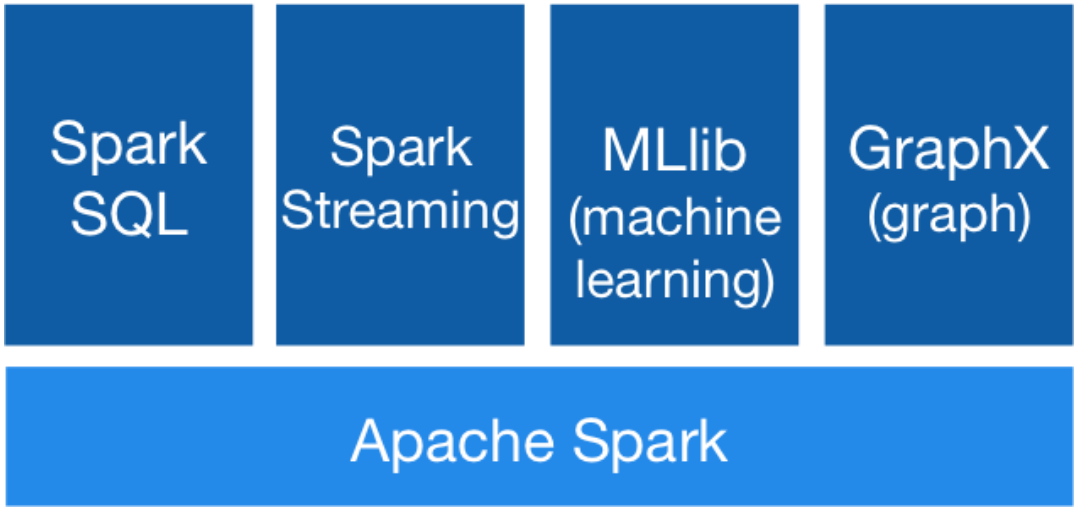
Muy complicado obtener **productividad** y **performance**



¿Qué es?

Es un motor general y muy rápido para el procesamiento en paralelo de datos en gran escala.





Motor Unificado

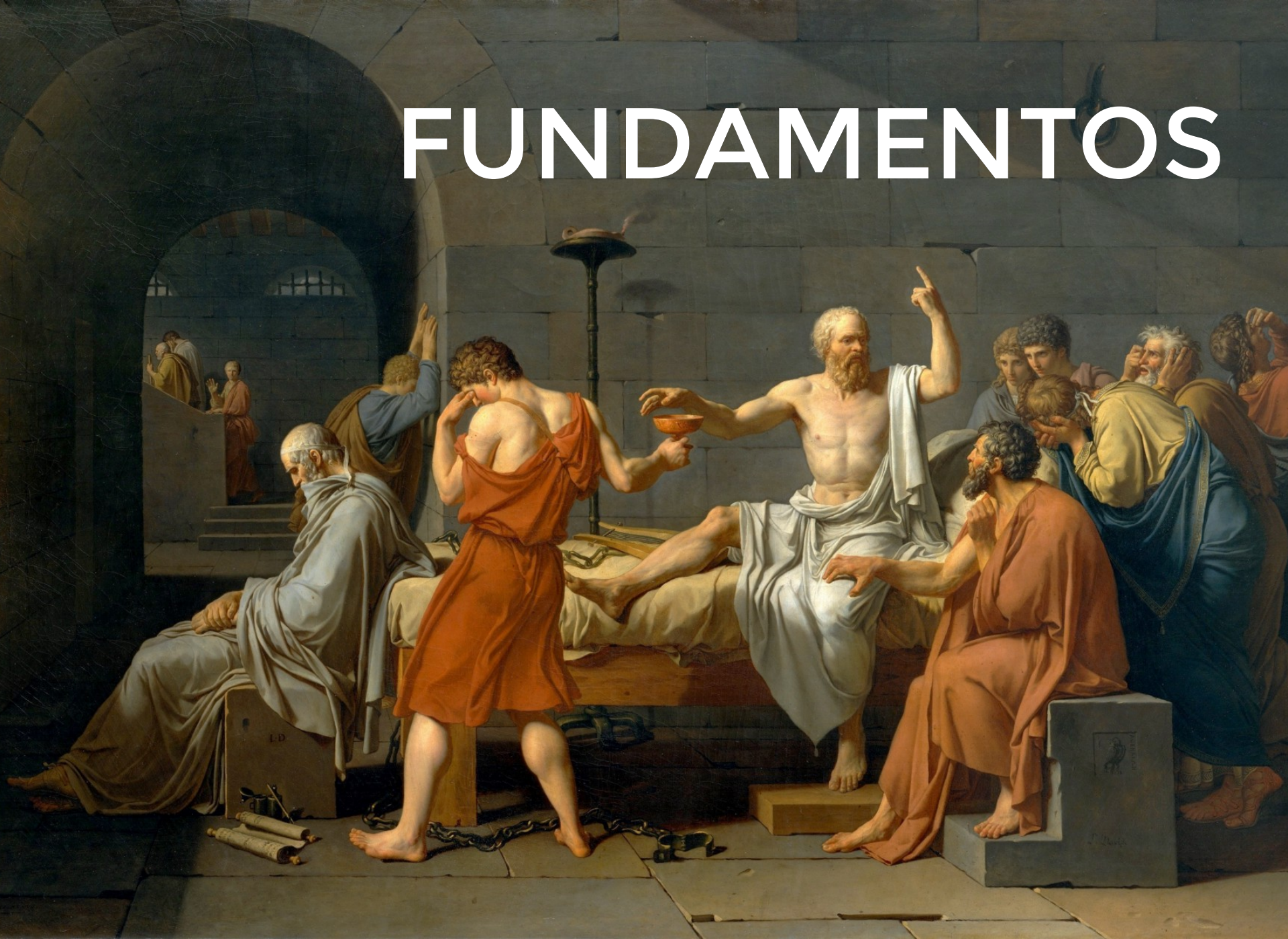
- Expresa todo el workflow con una API
- Conecta librerías existentes y sistemas de almacenamiento

APIs de alto nivel con espacio para optimizar

RDD → Transformaciones
RDD → Acciones
RDD → Caché

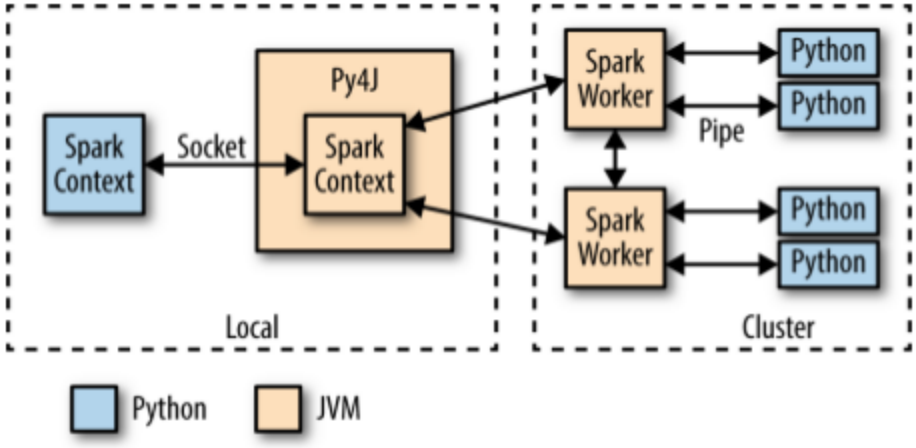
Dataset → Tipado
Dataset → Scala y Java
Dataset → Beneficios de RDD

Dataframe → Dataset[Row]
Dataframe → Optimizado
Dataframe → Versátil


FUNDAMENTOS

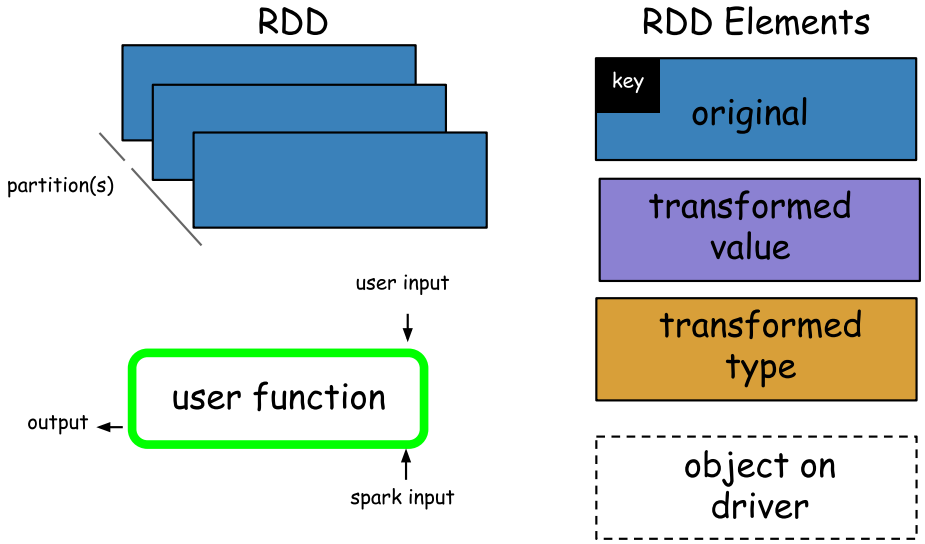
SparkContext

- Un programa Spark crea primero un objeto SparkContext que le indica a Spark cómo y dónde acceder a un clúster.
- El shell PySpark y Optimus crean automáticamente la variable sc.
- iPython y los programas deben usar un constructor para crear un nuevo SparkContext

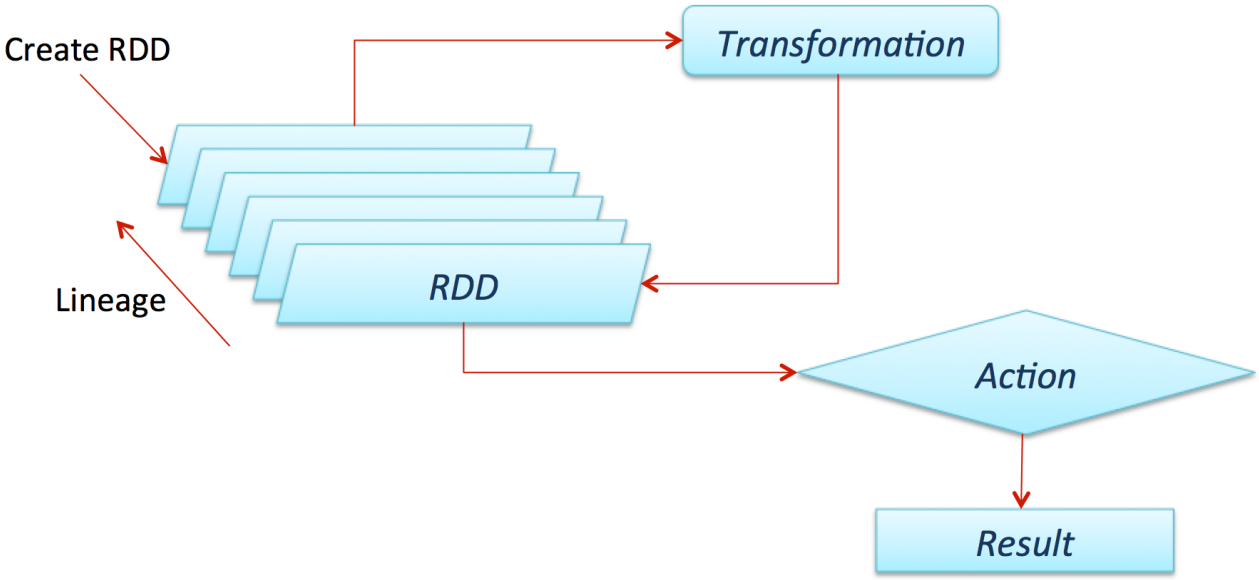


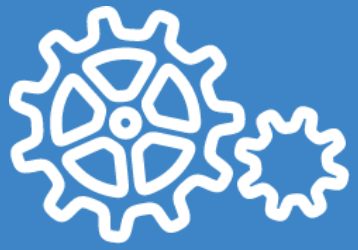
RDDs

Los Resilient Distributed Datasets (RDD) son una colección distribuida de objetos JVM inmutables que permiten realizar cálculos muy rápidamente, y son la columna vertebral de Apache Spark.



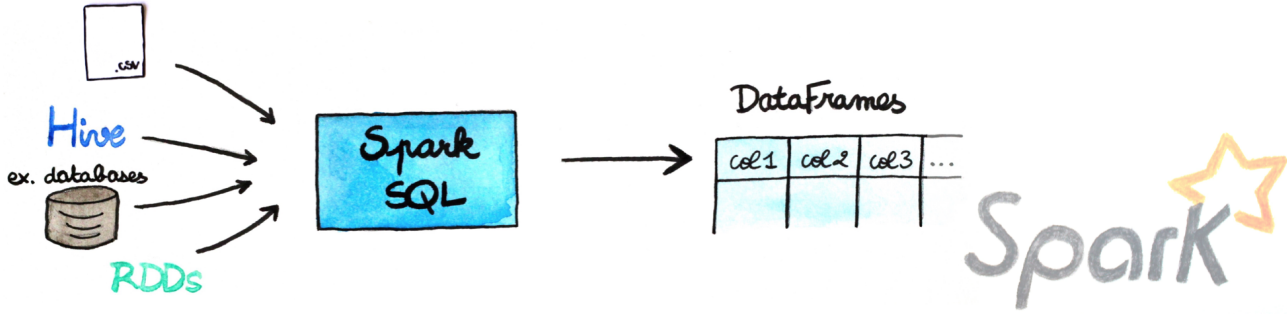
RDDs



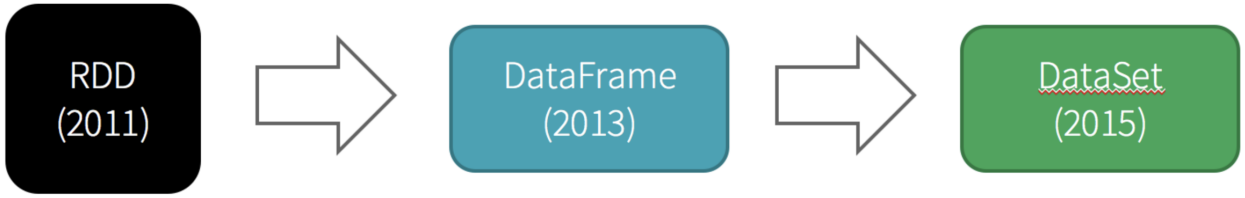


DataFrames

	0	1	2	3	4	5
0	Rank (1-10)	University	Rank (11-20)	University	Rank (21-30)	University
1	1	University of Cambridge	11	University of Lancaster	21	University of Edinburgh
2	2	University of Oxford	12	University of Surrey	22	University of Kent
3	3	London School of Economics	13	Loughborough University	23	Cardiff University
4	4	University of St Andrews	14	University of York	23	University of Leeds
5	5	Durham University	15	University of East Anglia	23	University of Nottingham
6	6	Imperial College	16	University of Southampton	26	University of Sheffield
7	7	University of Warwick	17	University of Birmingham	27	Aston University
8	8	University of Bath	18	University of Bristol	28	King's College London
9	9	University College London	19	University of Leicester	29	University of Manchester
10	10	University of Exeter	20	Newcastle University	30	University of Glasgow



DataFrames



Distribute collection of JVM objects

Functional Operators (map, filter, etc.)

Distribute collection of Row objects

Expression-based operations and UDFs

Logical plans and optimizer

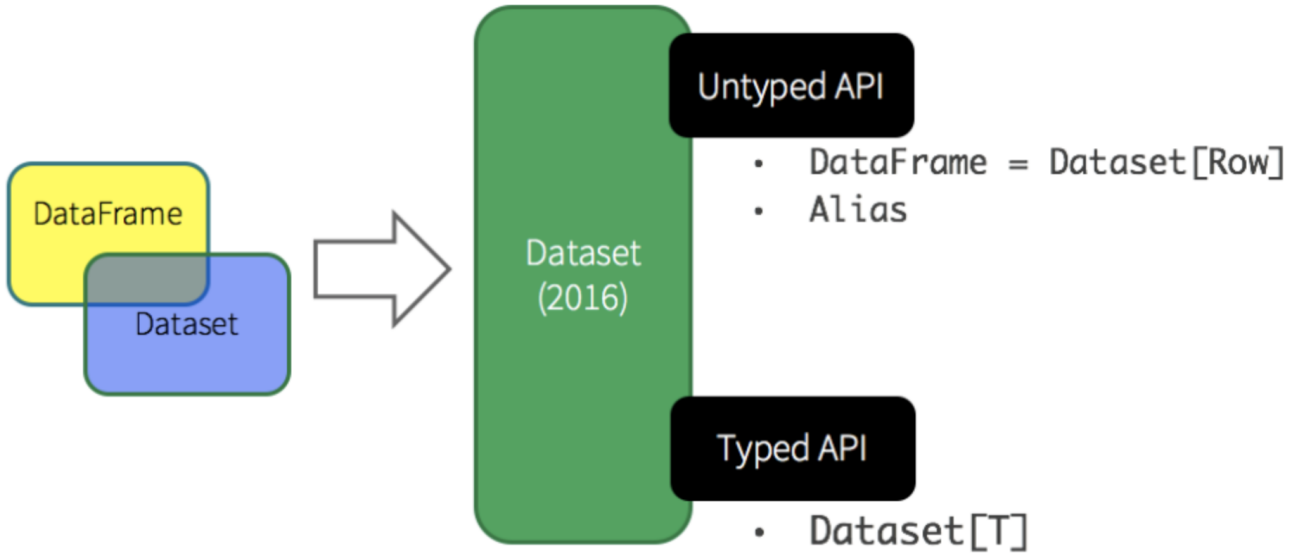
Fast/efficient internal representations

Internally rows, externally JVM objects

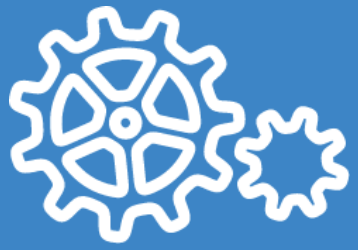
Almost the "Best of both worlds": type safe + fast

But slower than DF
Not as good for interactive analysis, especially Python

DataFrames


DataFrame
Dataset
Dataset
(2016)
Untyped API
• DataFrame = Dataset[Row]
• Alias
Typed API
• Dataset[T]

DataFrames



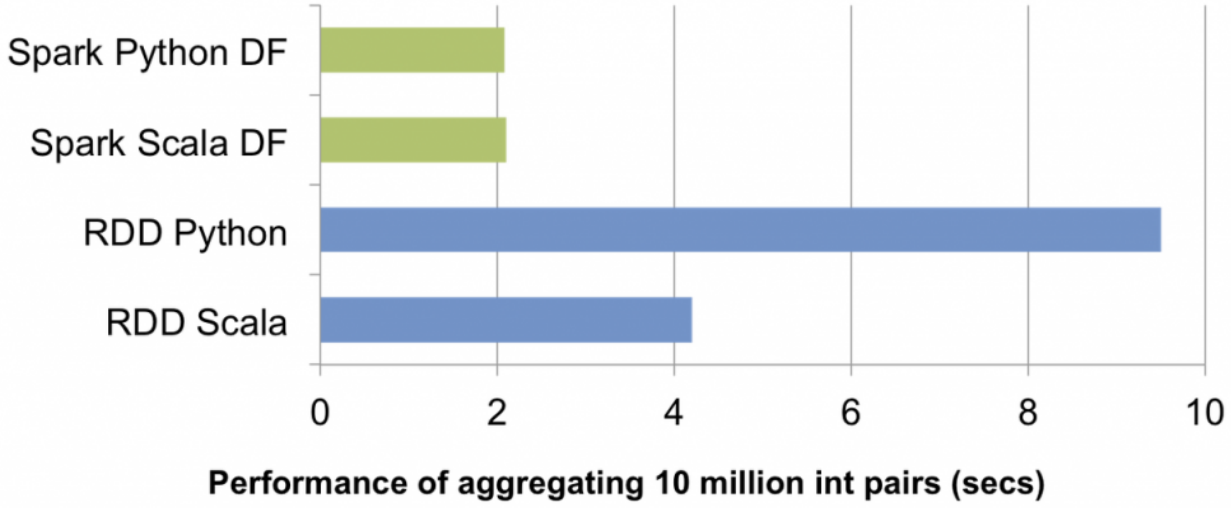
- GroupBy y AVG con RDDs

```
data.map(lambda x: (x.rooms, [x.age,1]) \
    .reduceByKey(lambda x, y: [x[0] + y[0], x[1] + y[1]]) \
    .map(lambda x: x[0], x[1][0] / x[1][1]) \
    .collect()
```

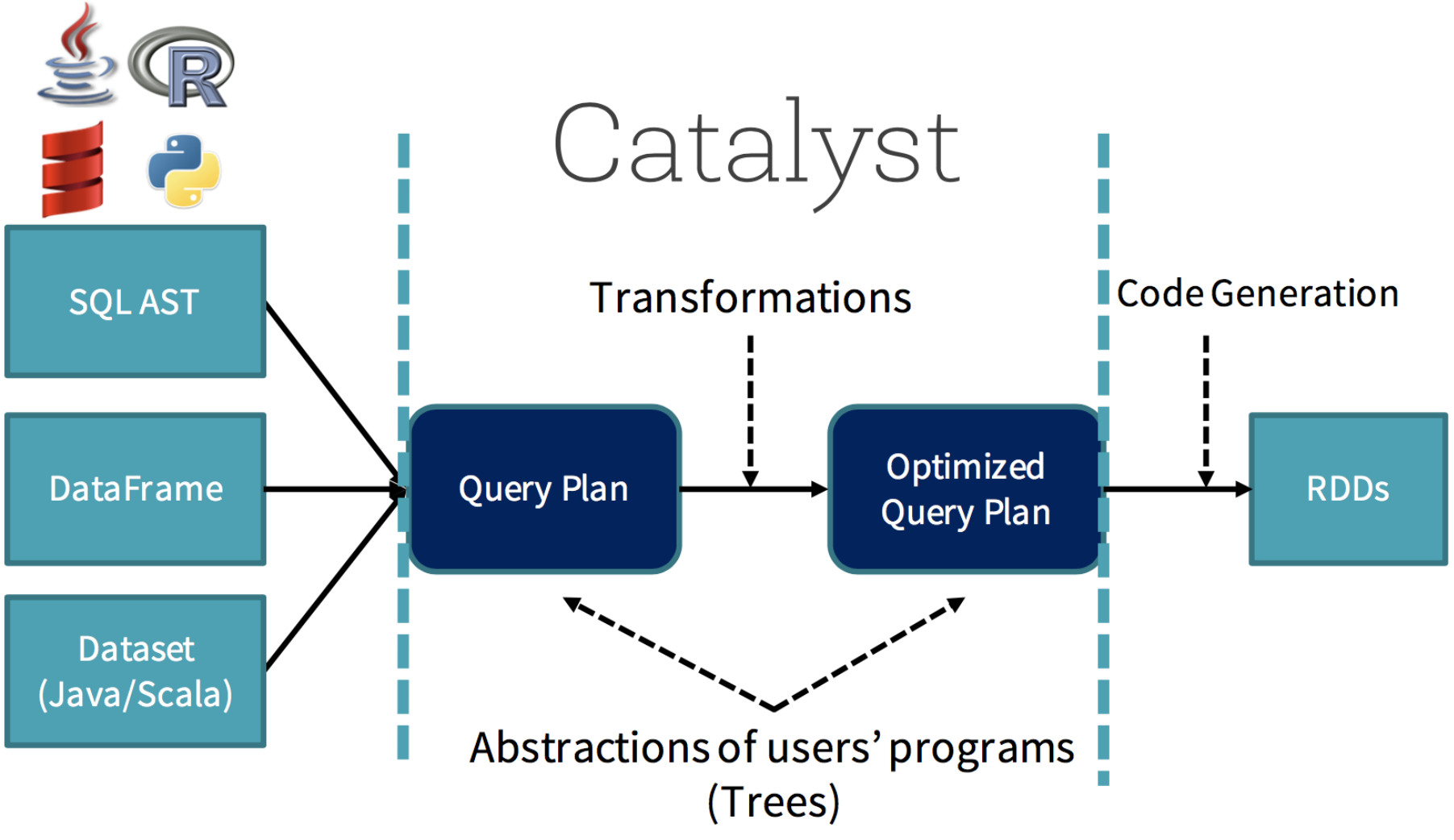
- GroupBy y AVG con DataFrames

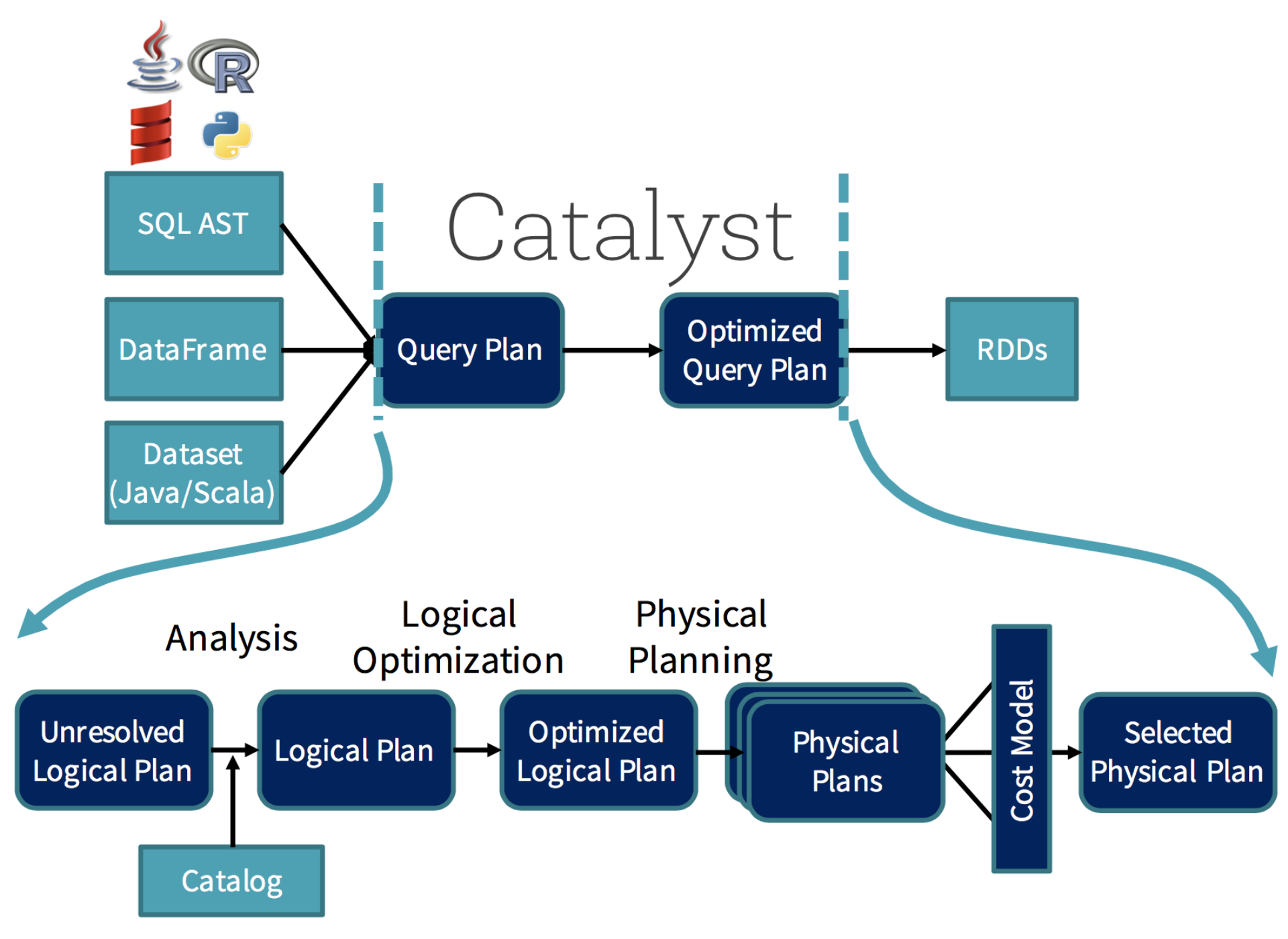
```
data.groupBy("rooms").avg("age")
```

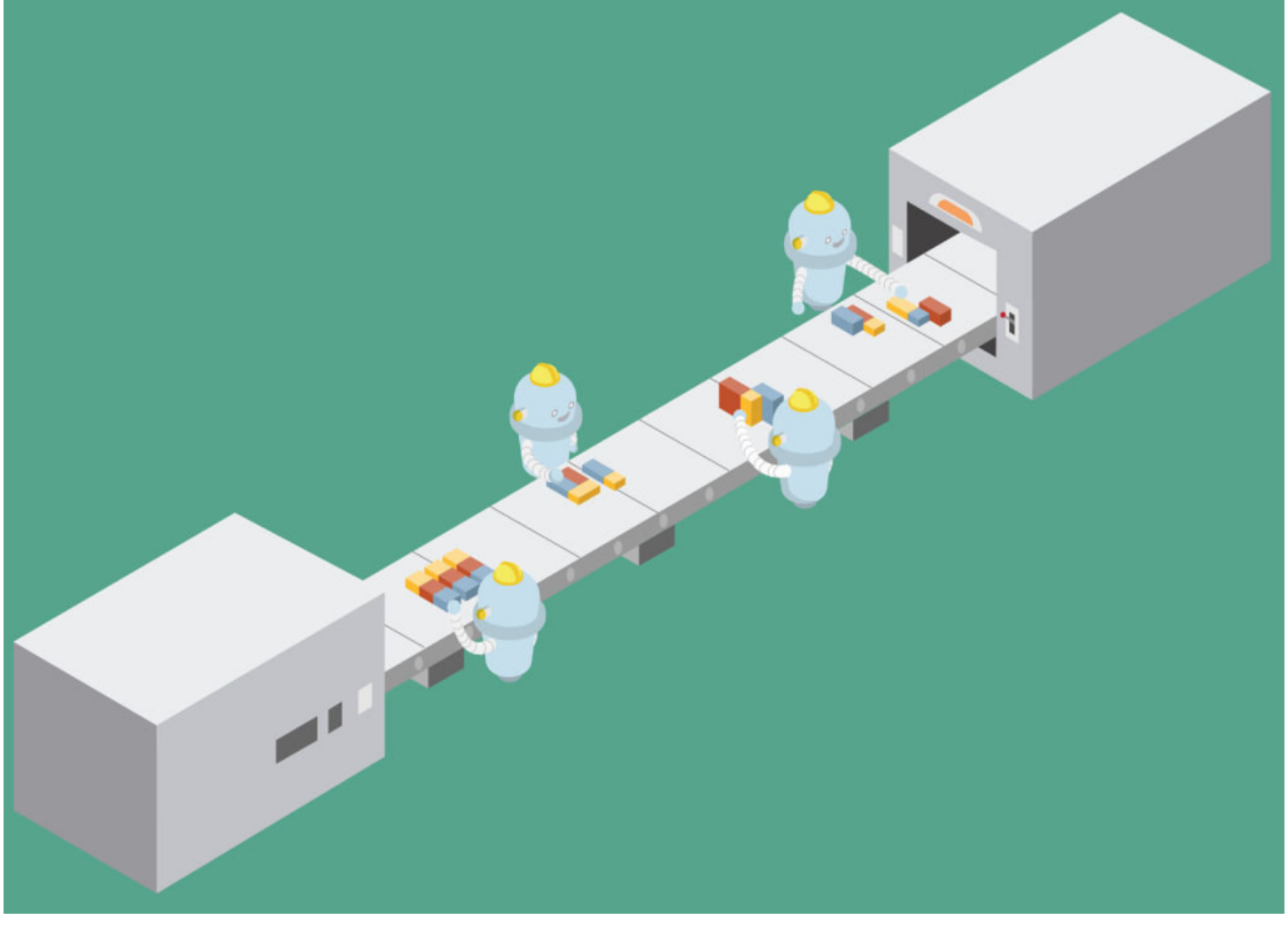
DataFrames



Optimización


Catalyst
SQL AST
DataFrame
Dataset
(Java/Scala)
Query Plan
Transformations
Optimized
Query Plan
Code Generation
RDDs
Abstractions of users' programs
(Trees)


SQL AST
DataFrame
Dataset
(Java/Scala)
Catalyst
Query Plan
Optimized
Query Plan
RDDs
Analysis
Logical
Optimization
Physical
Planning
Unresolved
Logical Plan
Logical Plan
Optimized
Logical Plan
Physical
Plans
Cost Model
Selected
Physical Plan
Catalog

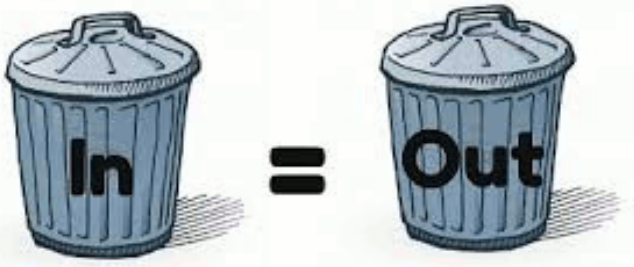


OPTIMUS

¿Por qué?

Los datos sucios cuestan $3 trilliones+ por año

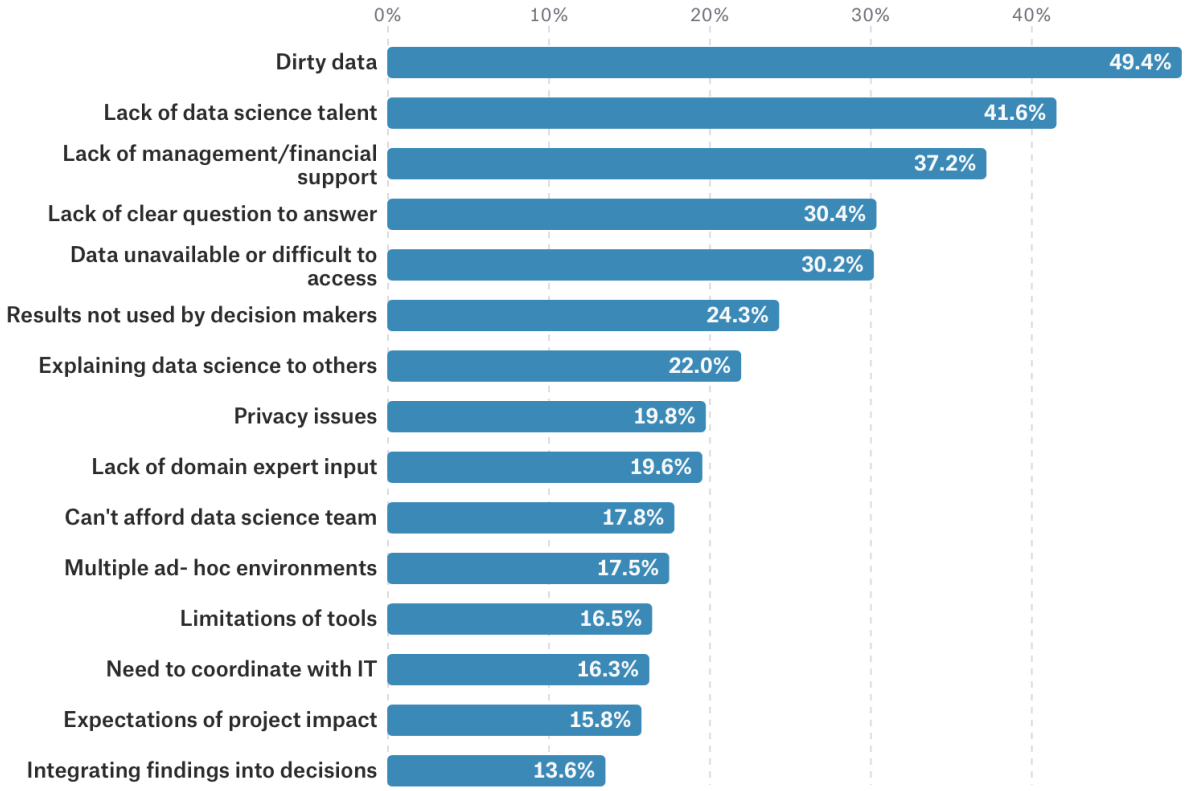
Malos Modelos



Las compañías pierden 12% de revenue

¿Por qué?

Encuesta Kaggle 16000 DS. Problemas.



Problemas

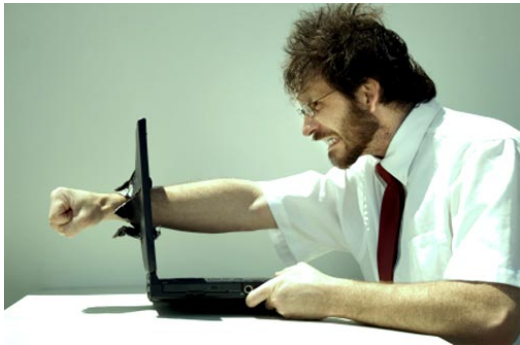
Fact of the day: **Los Datos están sucios**

	City	Country	Population	Area	Density
r_1	New York	USA	8734520	6400	26403
r_2	Philadelphia	United States	“1,204,542”	“3,231”	NaN
r_3	New York City	USA	8734520	6400	26403

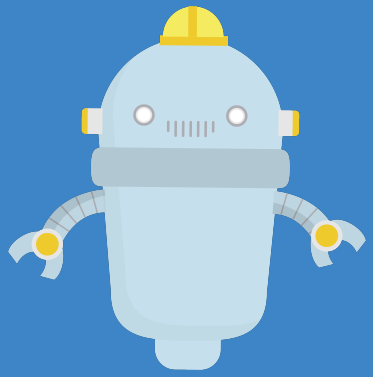
Problems

Soluciones actuales para limpiar datos

- Difíciles de usar y entender.
- La mayoría de ellos tienen su propio lenguaje.
- No son adecuados para Big Data.
- **SampleClean** no continuado y nada fácil de usar.
- Mala integración con Machine Learning.
- Atascados en la limpieza.
- Mal documentado.
- Un poco feos.



Optimus



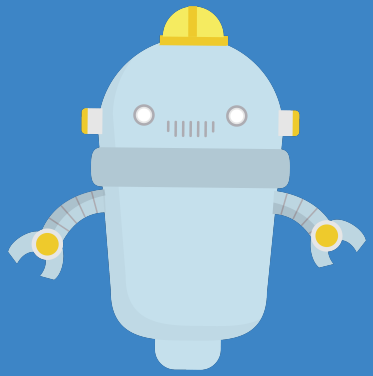
OPTIMUS

Optimus is the missing framework for cleaning and pre-processing data in a distributed fashion. It uses all the power of Apache Spark (optimized via Catalyst) to do so. It implements several handy tools for data wrangling and munging that will make your life much easier. The first obvious advantage over any other public data cleaning library or framework is that it will work on your laptop or your big cluster, and second, it is amazingly easy to install, use and understand.

https://github.com/ironmussa/Optimus

https://hioptimus.com

Optimus



Características:

- Fácil de instalar, usar e implementar.
- Utiliza Python y PySpark.
- Funciona en tu computadora portátil o clúster.
- DFTransformer con excelentes métodos para la limpieza de datos y disputas.
- DFAnalyzer con excelentes métodos para visualizar datos y comprender.
- Métodos para feature engineering (realmente fáciles de usar).
- Machine Learning con Spark (fácil e igualmente rápido). Trabajo en progreso.

Demo