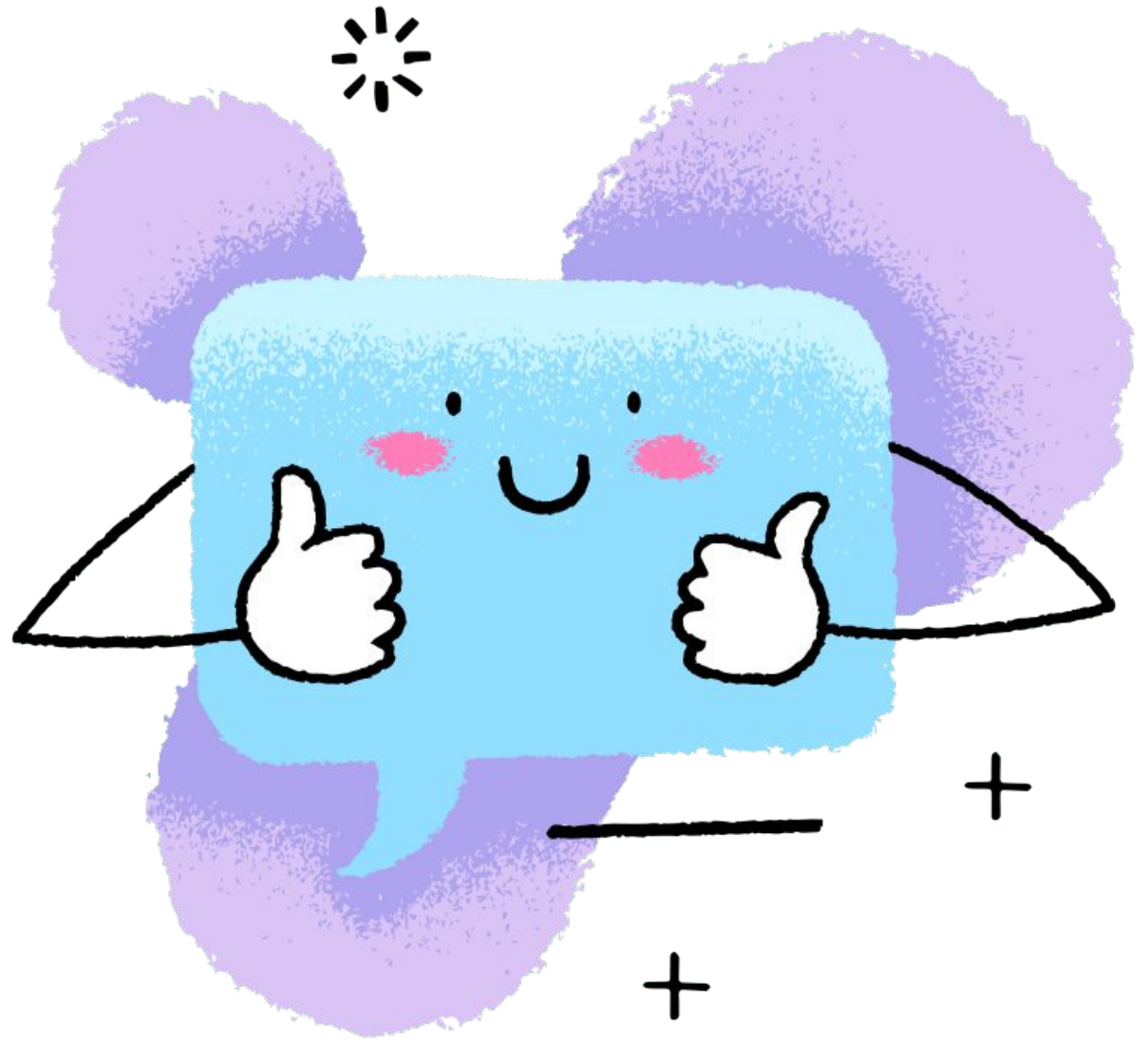


PySpark



Знакомство

Котельников Егор Александрович

сейчас в **goods.ru** (ведущий разработчик): создание распределенного хранилища, реализация потоковой обработки данных из BigQuery, Google Analytics

Платформа ОФД (ведущий разработчик): разработка хранилища, разработка аналитических продуктов на данных продаж не крупного бизнеса (50% всех чеков РФ)

МТС (инженер данных): создание витрин данных для DS, поиск закономерностей в геолокациях

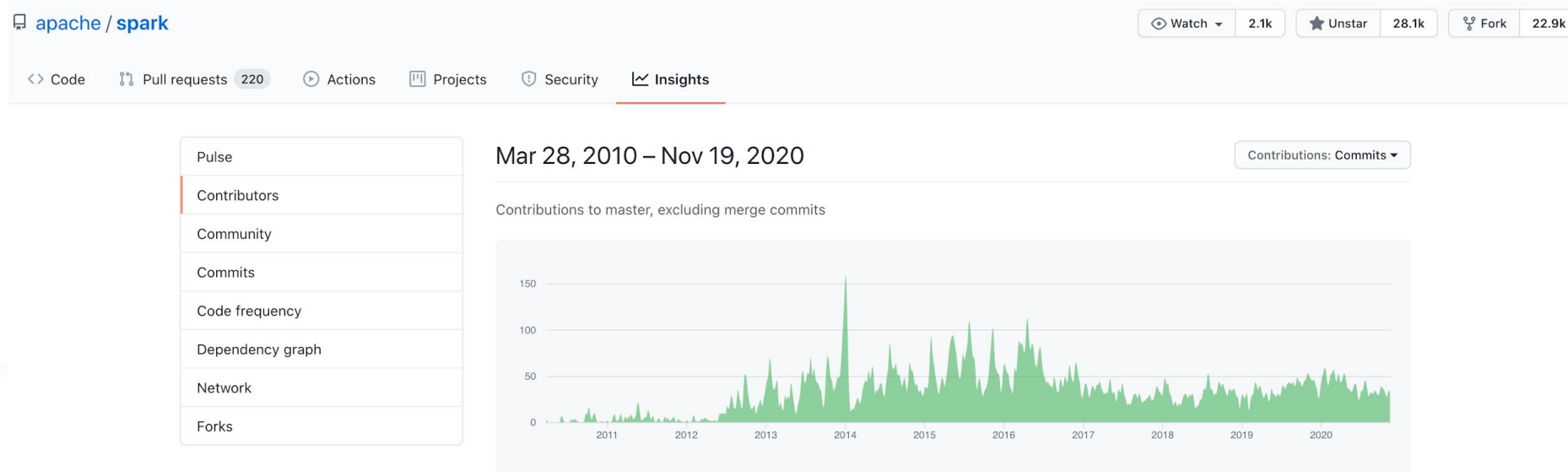
Яндекс (стажер): стажировка разработчиком в Яндекс.Таланты

Образование: физфак МГУ, ШАД (один курс)



Почему Spark

1. Spark есть везде: банки (масштабируемость), средний и маленький бизнес (бесплатно)
2. Молодая технология, следовательно, мало специалистов
3. Универсальный фреймворк: создание хранилища, запросы к данным, потоковая обработка, ML.
4. Активное развитие



What is Apache Spark?

Apache Spark is a **general purpose platform** for quickly processing **large scale data** that is developed in **Scala** programming language.

- A framework for distributed computing (100s of nodes)
- In-memory, fault tolerant data structures
- API that supports Scala, Java, Python, R, SQL
- Open source

Apache Spark is a **unified computing engine** and a set of **libraries** for parallel data processing on computer clusters.

План курса

1. Введение. Используемые инструменты. Обзор интерфейса
2. Архитектура Spark. Планы выполнения запросов. Оптимизация запросов.
3. Типы данных, сложные запросы. Пользовательские функции
4. Машинное обучение

Архитектура Spark

Apache Spark

Что будет на уроке

1. Что такое большие данные.
2. Spark API: python, java, scala
3. SparkSession. Параметры приложения spark
4. Форматы хранения данных parquet, avro. Другие источники данных: jdbc.

Откуда пошло

Google File System (2003): распределенная, устойчивая к отказу единичных серверов

Bigtable: scalable storage of structured data across GFS

MapReduce (MR)

Hadoop Modules:

Others (For Data Processing)	MapReduce (For Data Processing)
YARN (Resource Management For Cluster)	
HDFS (A Reliable & Redundant Storage)	

Недостаток Hadoop MapReduce:
промежуточные результаты последовательных операций
записываются на диск

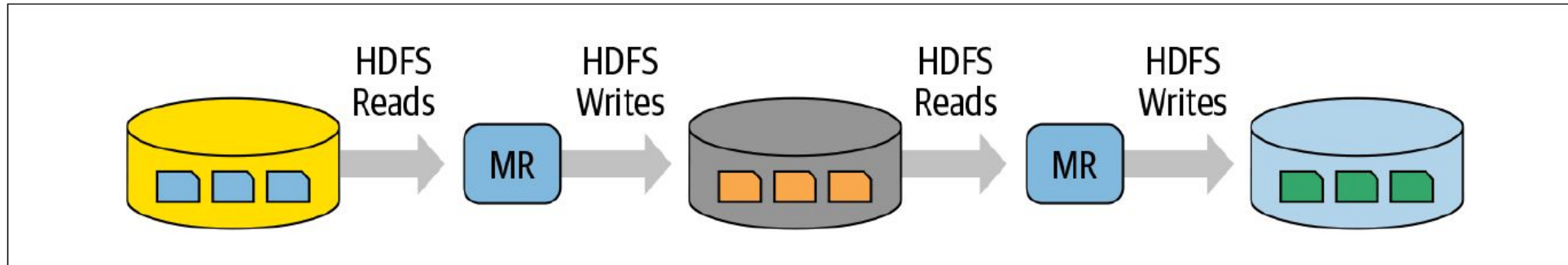
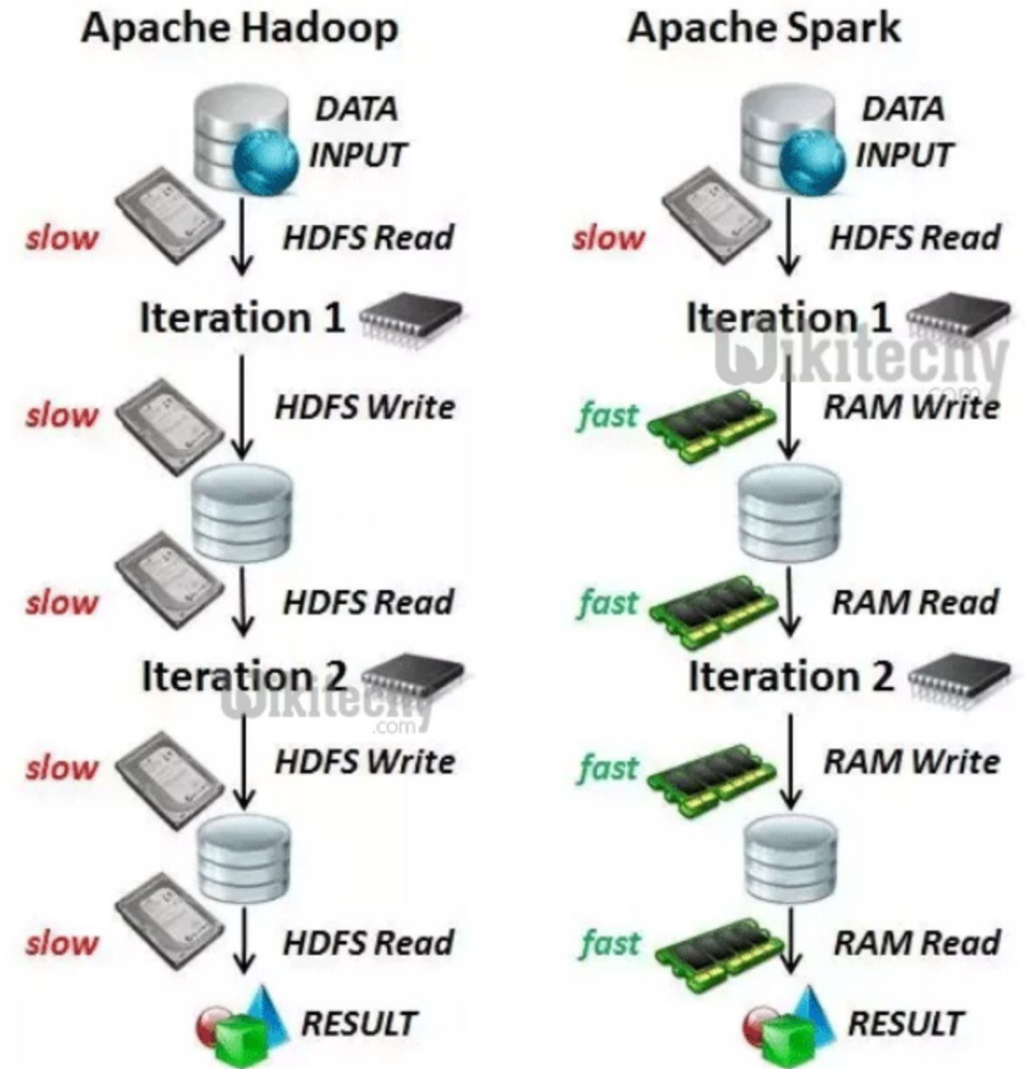


Figure 1-1. Intermittent iteration of reads and writes between map and reduce computations

Apache Hive (Tez), Apache Impala

Spark (2009) vs Hadoop MapReduce:

1. in-memory storage for intermediate results between iterative and interactive map and reduce computations
2. offer easy and composable APIs in multiple languages as a programming model

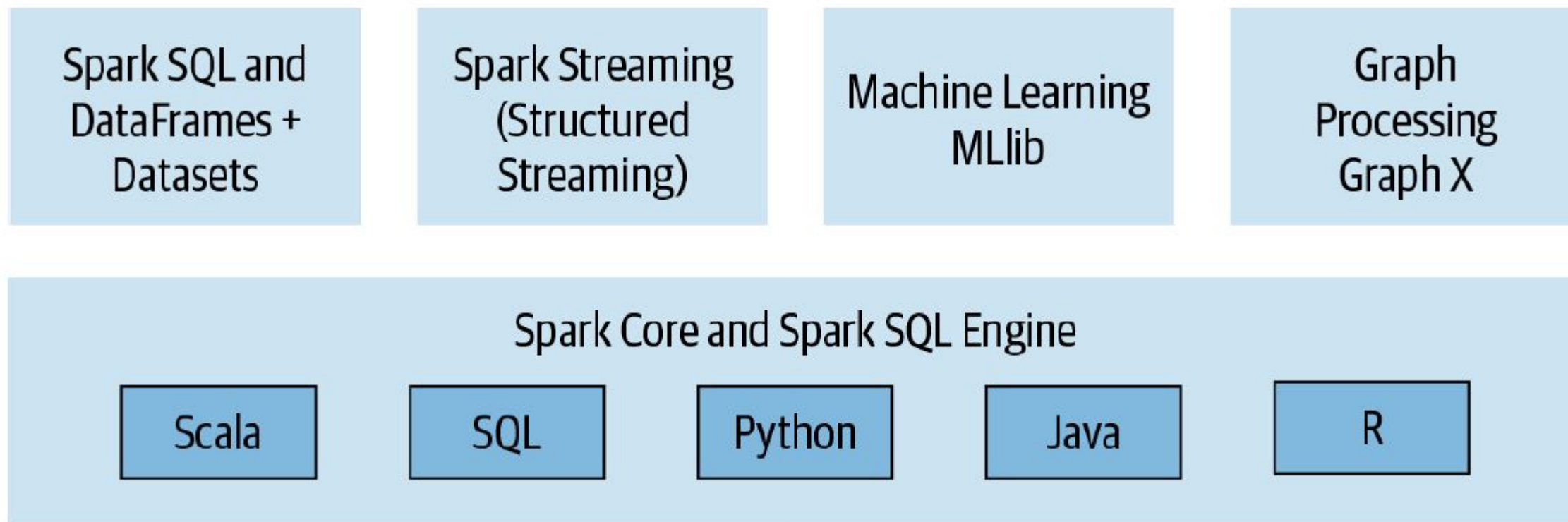


Предпосылки Spark:

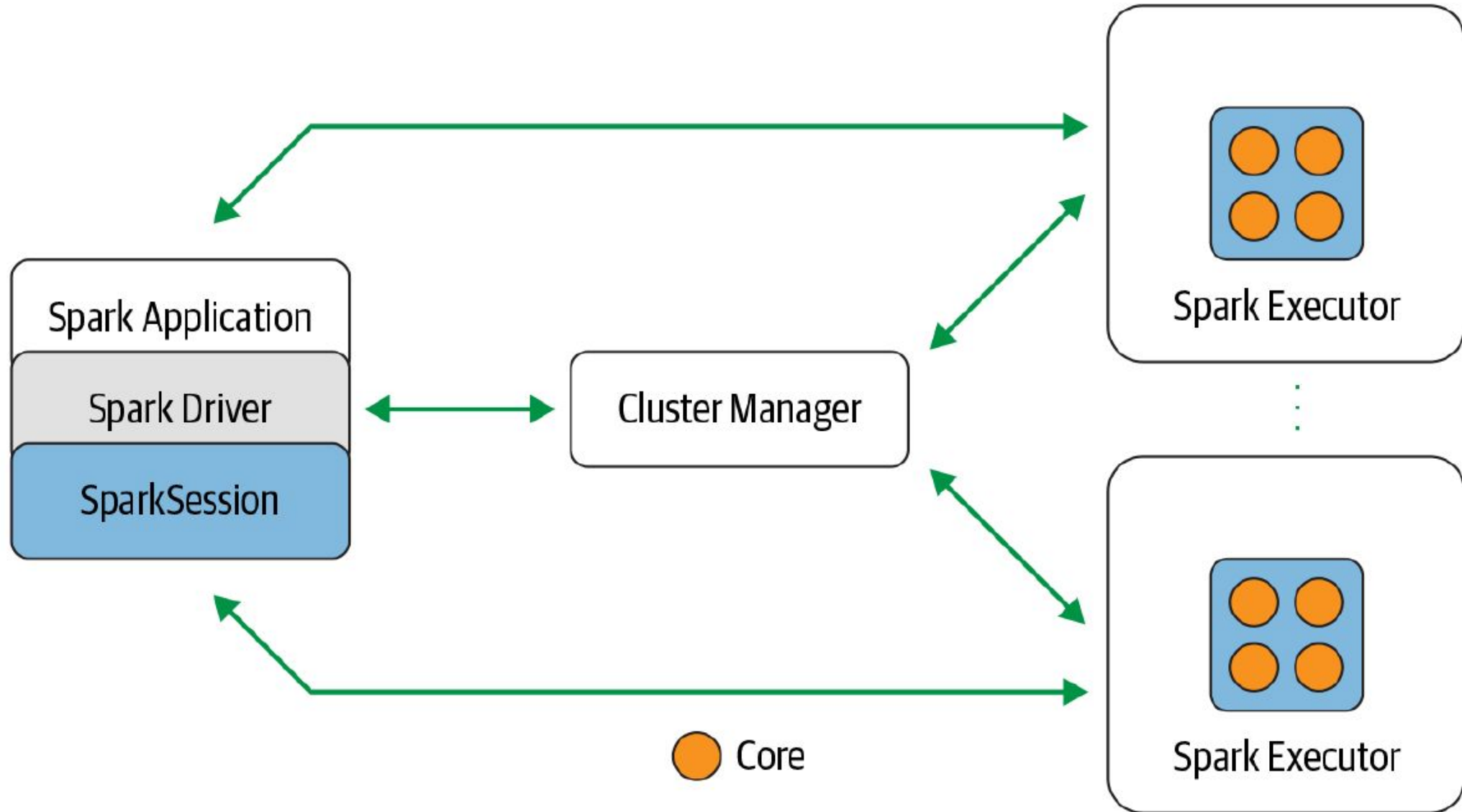
Со времен появления Hadoop MapReduce железо стало лучше по числу ядер CPU и памяти. Вычисления строятся не последовательно, а в DAG и оптимизируются.

Потребность в разделении хранилища от вычислений: Spark может принимать данные из любых источников и обрабатывать их в памяти, пакетами (jdbc for SQL) или потоком (Kafka)

Spark -- Unified Engine for Big Data Processing -- объединил в себе возможности пакетной обработки, работы с графами, потоками и SQL



Архитектура приложения Spark



Zeppelin

- Встроенная визуализация
- Динамические формы
- Каждый параграф в своем интерпретере
- Плагины



FINISHED ▶ ⌵ ⌵ ⌵ ⌵

%md Hello \${name=sun}

name

moon

Hello moon

FINISHED ▶ ⌵ ⌵ ⌵ ⌵

%spark

```
println("Today is "+z.select("day", Seq(("Monday", "1"), ("Tuesday", "2"), ("Wednesday", "3"), ("Thursday", "4"), ("Friday", "5"), ("Saturday", "6"), ("Sunday", "7"))))
```

day

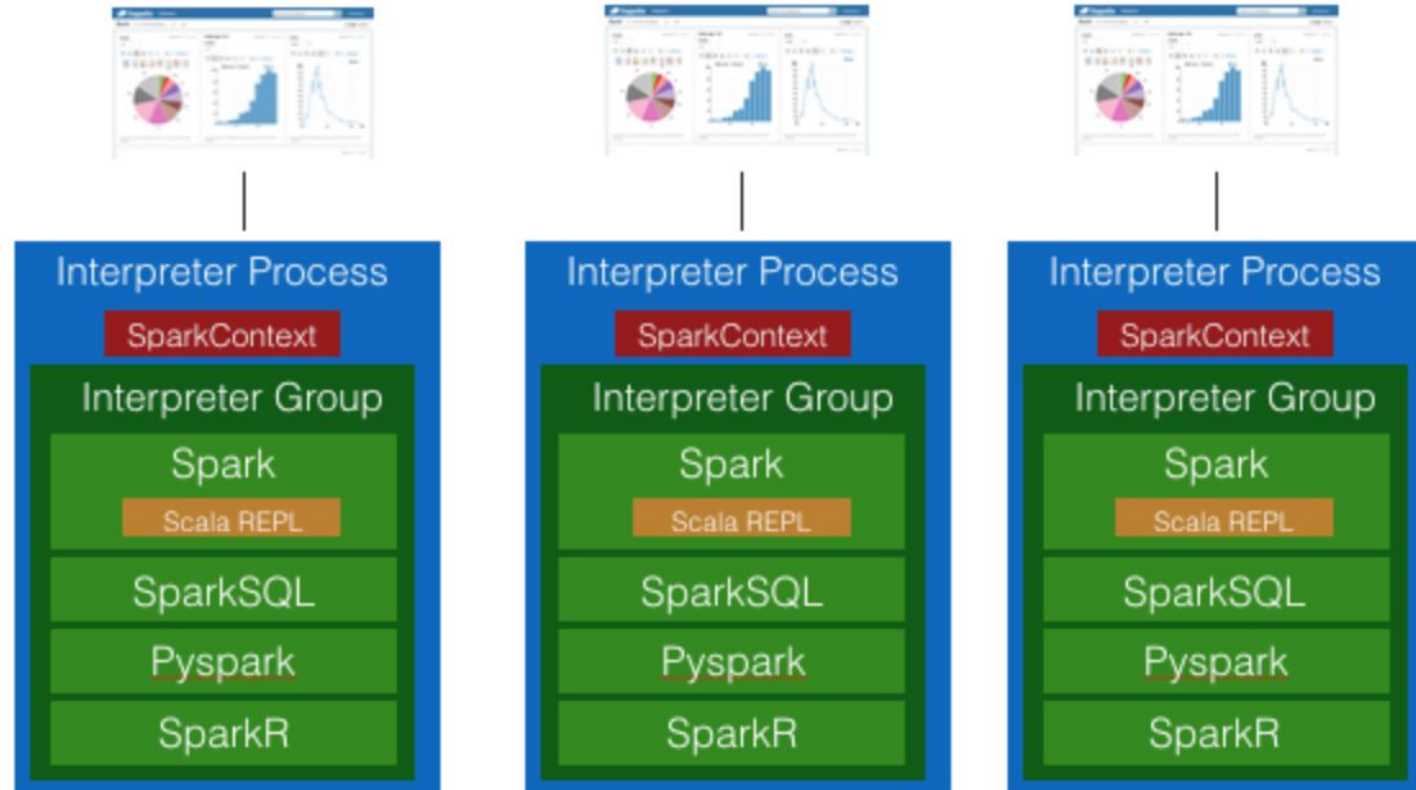
5

Today is Friday

Zeppelin

Interpreter

In *Isolated* mode, each Note has its own SparkContext and Scala REPL.



Cluster manager

Распределяет ресурсы между spark приложениями

standalone cluster manager

FIFO исполнение приложений

Apache Hadoop YARN

стандартное решение

распределяет, освобождает ресурсы между различными приложениями

Часть Hadoop

Apache Mesos

YARN-like, but большая изолированность процессов

поддерживает не-hadoop приложения

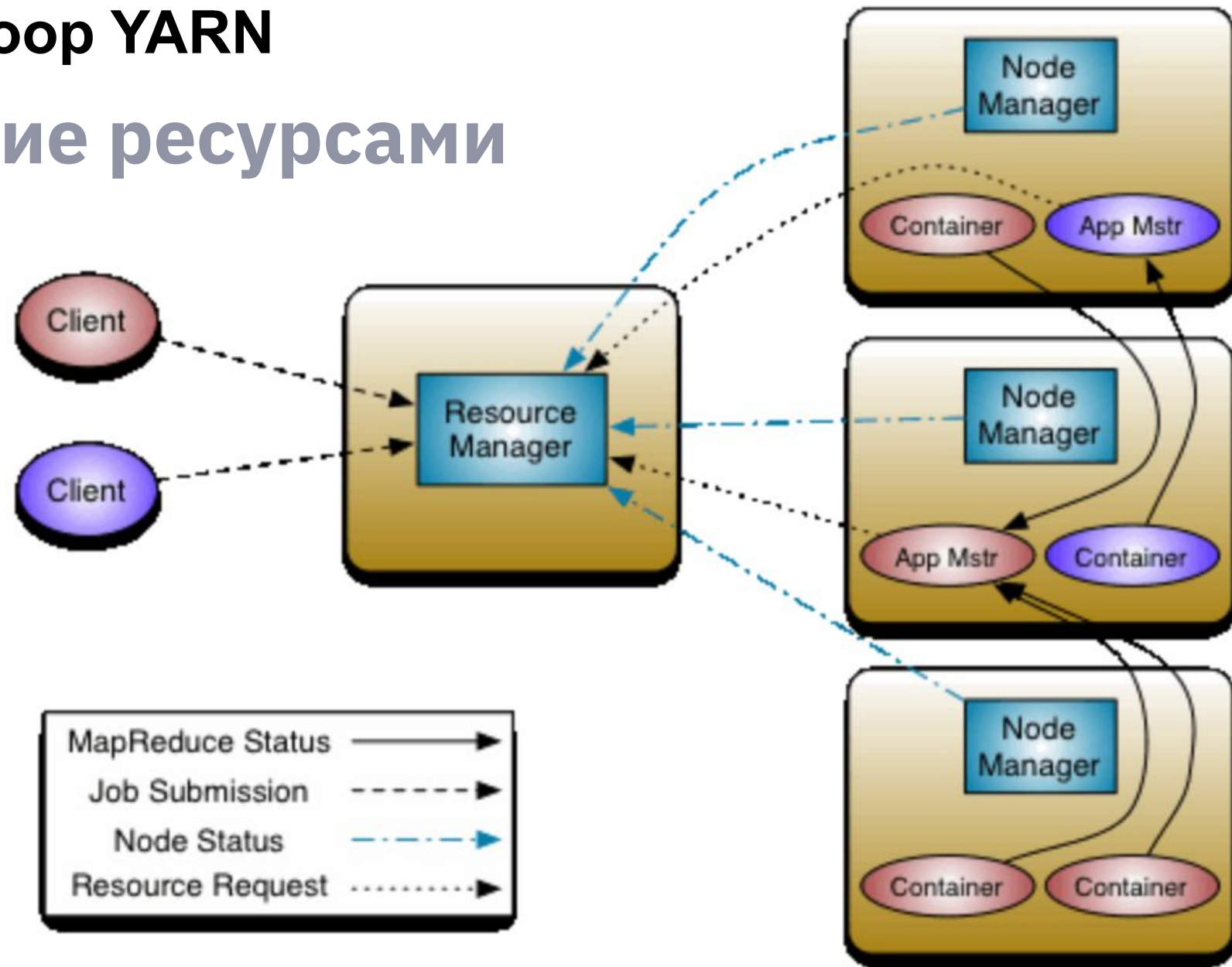
Kubernetes

запуск в контейнерах = абсолютная изолированность

существенно сложнее в поддержке

Apache Hadoop YARN

Управление ресурсами



Spark driver

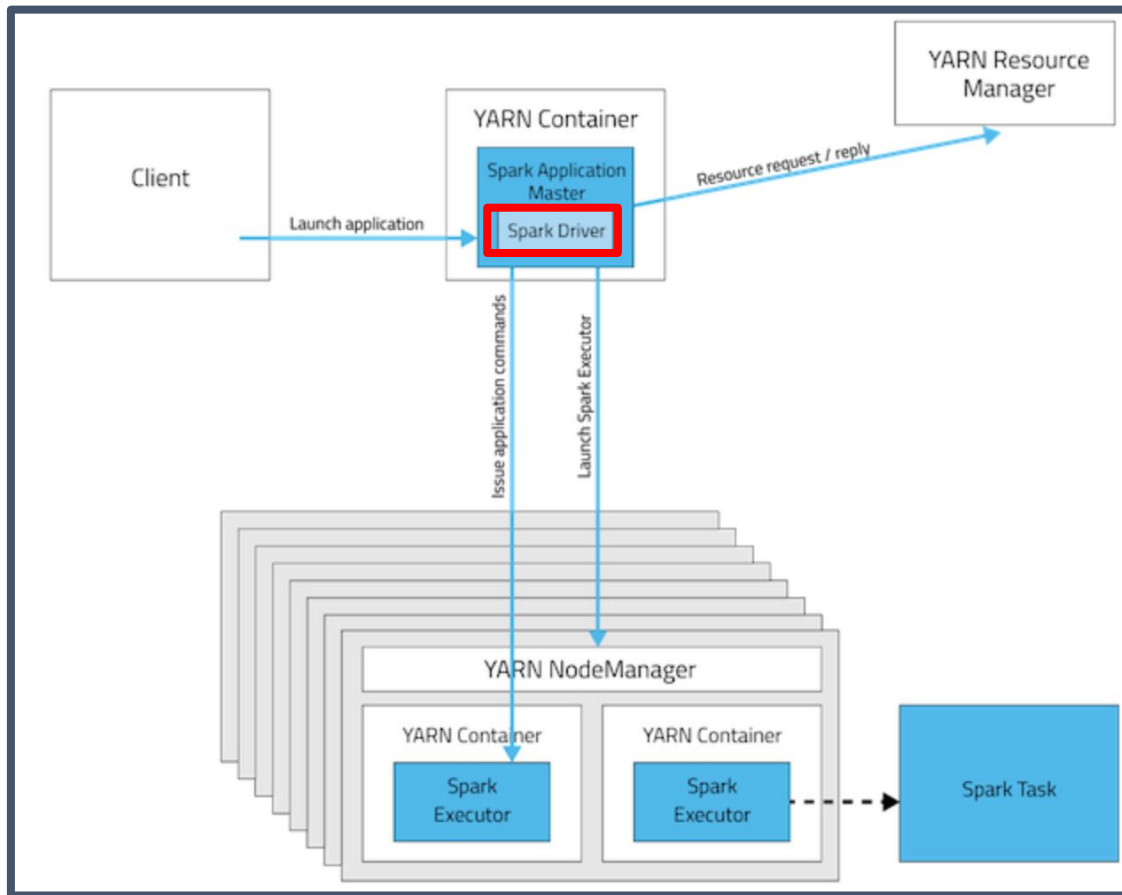
Может

находиться вне
кластера

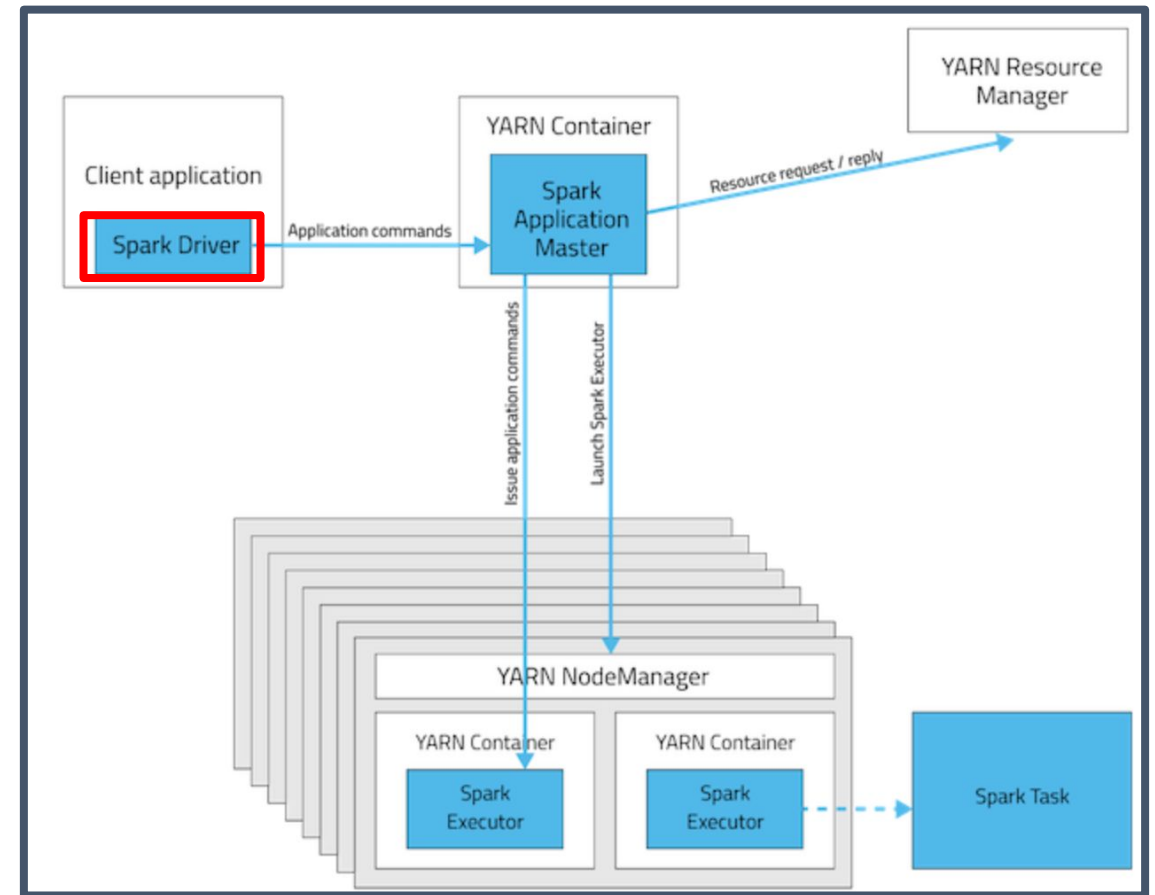
- it communicates with the cluster manager
- it requests resources (CPU, memory, etc.) from the cluster manager for Spark's executors (JVMs)
- it transforms all the Spark operations into DAG computations
- it distributes their execution as tasks across the Spark executors

Spark on YARN deployment modes

cluster mode



client mode



SparkSession

Точка входа в приложение

С его помощью можно:

- Управлять параметрами spark приложения
- читать и записывать данные в spark DataFrame
- исполнять SQL

```
1 from pyspark.sql import SparkSession
2
3 spark = SparkSession \
4     .builder \
5     .appName("Python Spark SQL basic example") \
6     .config("spark.some.config.option", "some-value") \
7     .getOrCreate()
```

Компоненты Spark

Structured
Streaming

Advanced
Analytics

Libraries &
Ecosystem

Structured APIs

Datasets

DataFrames

SQL

Low-level APIs

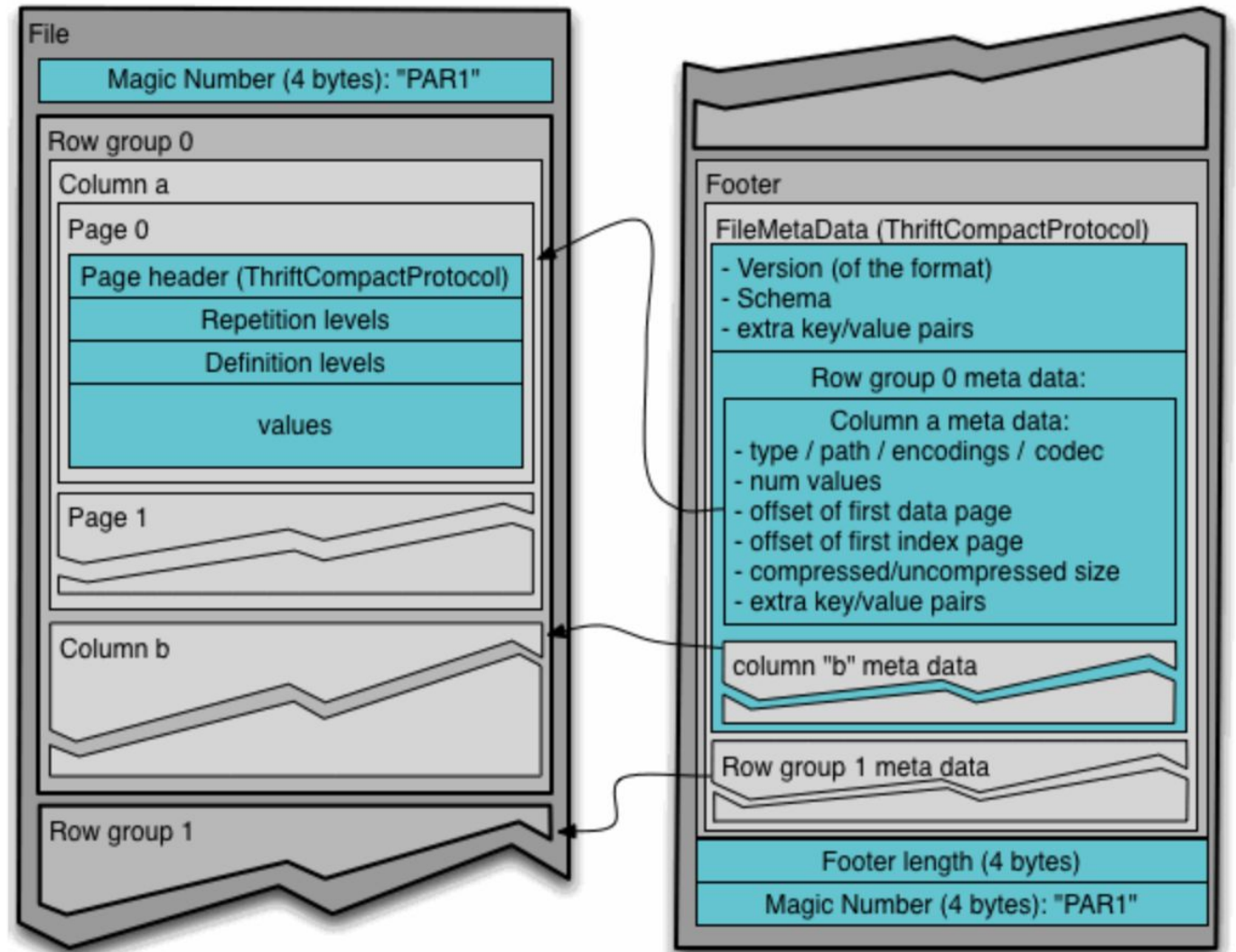
RDDs

Distributed Variables

Parquet file format

Формат хранения данных для hadoop


- Колончатый формат
- Разное сжатие для разных типов данных



Parquet vs CSV

Меньше размер, быстрее чтение

The following table compares the savings as well as the speedup obtained by converting data into Parquet from CSV.

Dataset	Size on Amazon S3	Query Run Time	Data Scanned	Cost
Data stored as CSV files	1 TB	236 seconds	1.15 TB	\$5.75
Data stored in Apache Parquet Format	130 GB	6.78 seconds	2.51 GB	\$0.01
 Savings	87% less when using Parquet	34x faster	99% less data scanned	99.7% savings

Практика

Работа с DataFrame

тетрадки

Lecture 1

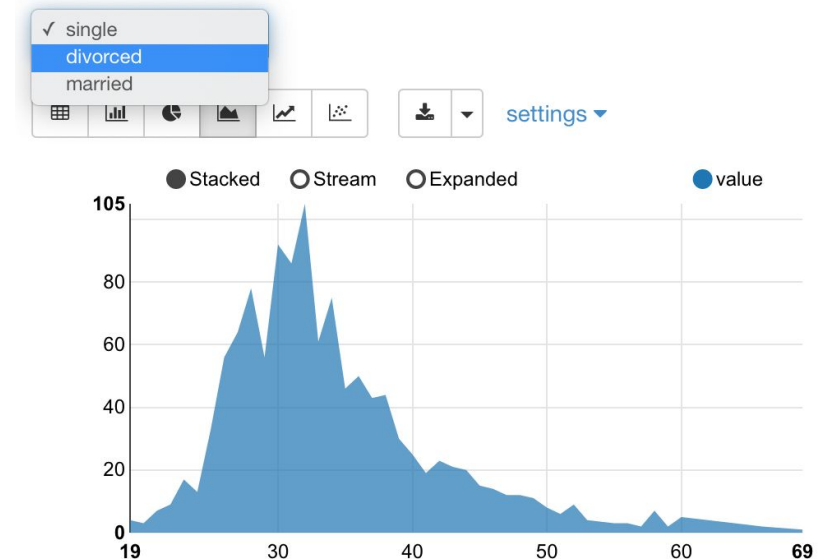
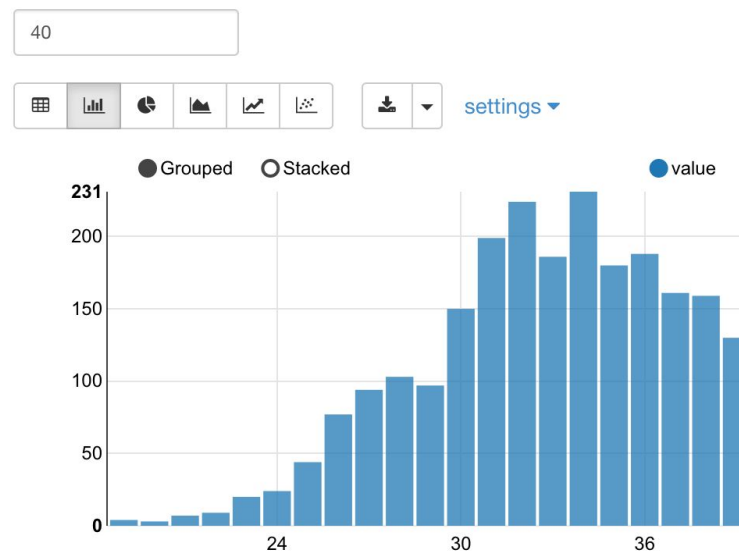
Spark SQL (PySpark)

Домашнее задание 1

Визуализация

<https://s3.amazonaws.com/apache-zepplin/tutorial/bank/bank.csv>

1. Построить распределения клиентов по возрастам
2. Распределение по возрасту с динамическим численным параметром `max_age`
3. Распределение по возрасту с динамическим параметром “marital”



Домашнее задание 2

Fire Station onboarding

```
/user/admin/sf-fire-calls.csv
```

- What were all the different types of fire calls in 2018?
- What months within the year 2018 saw the highest number of fire calls?
- Which neighborhood in San Francisco generated the most fire calls in 2018?
- Which neighborhoods had the worst response times to fire calls in 2018?
- Which week in the year in 2018 had the most fire calls?
- Is there a correlation between neighborhood, zip code, and number of fire calls?
- How can we use Parquet files or SQL tables to store this data and read it back?

Файлы для упражнений

<https://github.com/apache/spark/blob/master/examples/src/main/resources/people.json>

Полезные ресурсы

<https://sparkbyexamples.com>

[parquet doc](#)

[A Neanderthal's Guide to Apache Spark in Python](#)

[Matei Zaharia. Презентация от автора Spark](#)

Спасибо!
Каждый день
вы становитесь
лучше :)

