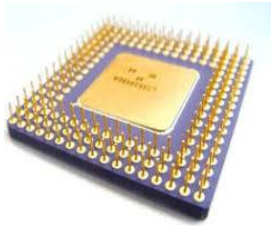
Spark

Guilherme
Gustavo
Victor

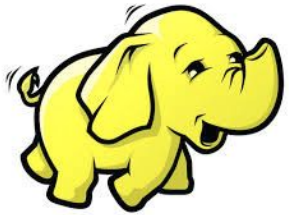
O que é Spark ?

- Apache Spark é uma engine para computação unificada e um conjunto de bibliotecas para processamento paralelo de dados em clusters.
- Spark foi programado em sua grande parte com Scala e tem suporte a várias linguagens de programação como Python, Java, Scala e R.



Mas porquê ele existe ?

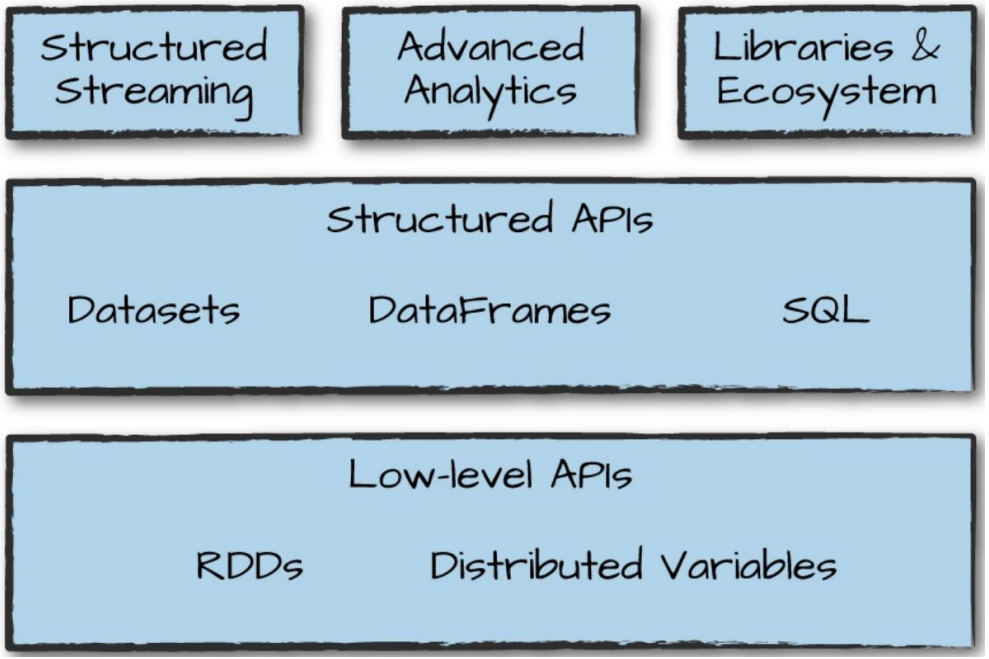
- Limitação da lei de moore (2005)
- Dificuldade de tratar concorrência em sistemas distribuídos
- Barateamento do custo de armazenamento (Big Data)
- Necessidade de um framework de uso geral
- Necessidade de resolver problemas do Hadoop.



O nascimento do spark

- Universidade de Berkley (Califórnia) 2009
- Equipe da pesquisa: **Matei Zaharia**, Mosharaf Chowdhury, Michael Franklin, Scott Shenker
- Criadores: Grupo de Pesquisa do AMPlab
- Entrevista com usuários do Haddop
- Tentativa de criar um framework generalista para tratar concorrência em sistemas distribuídos
- Projeto torna-se propriedade da Apache Foundation (2013)
- Databricks é fundada por membros da AMPlab

Núcleo do Spark



Filosofia

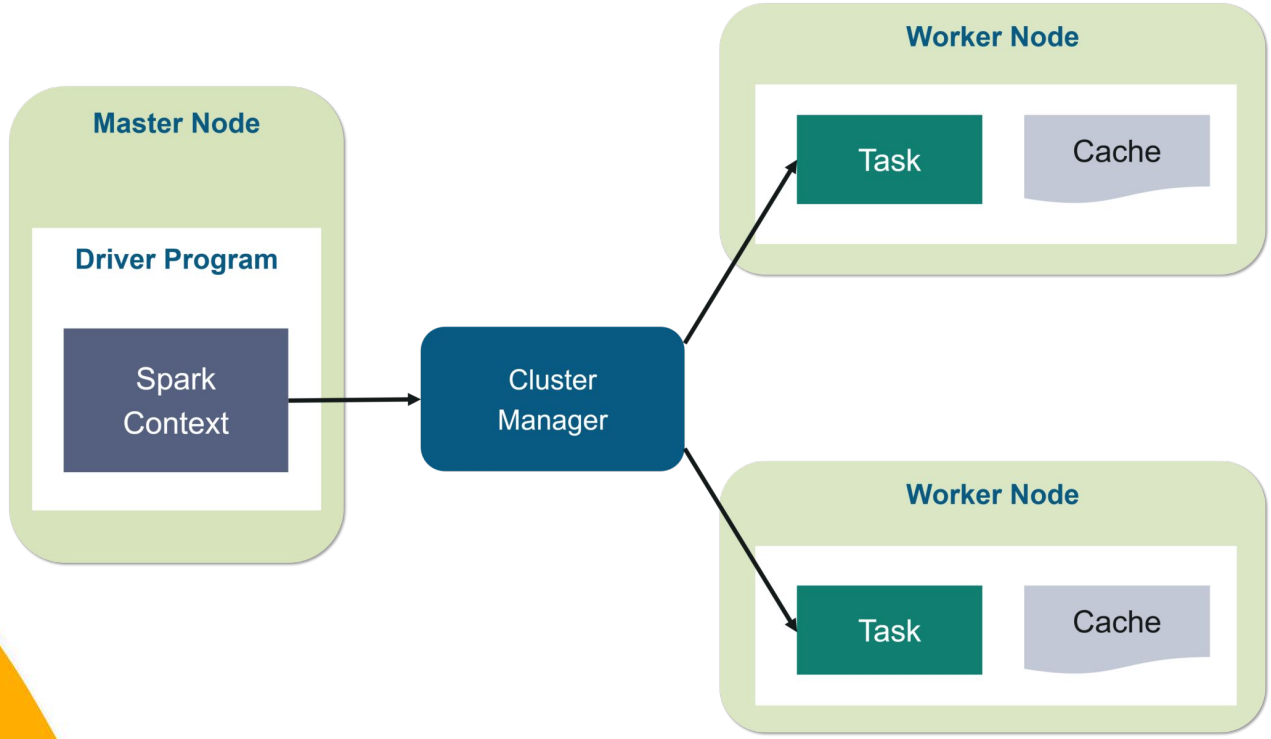
Spark SQL - módulo para trabalhar com dados estruturados

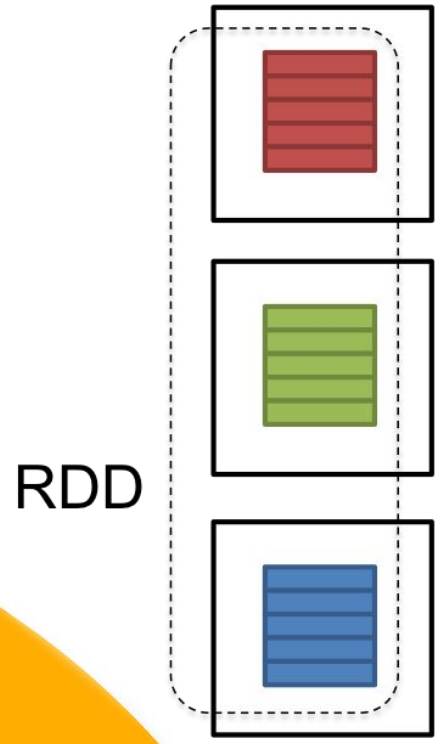
Spark MLlib - biblioteca escalável de Machine Learning

Spark GraphX - API para computação paralela de grafos.

Spark Streaming - simplifica a construção de aplicações de streaming escaláveis e tolerantes à falha.

Operação


Master Node
Driver Program
Spark Context
Cluster Manager
Worker Node
Task
Cache
Worker Node
Task
Cache

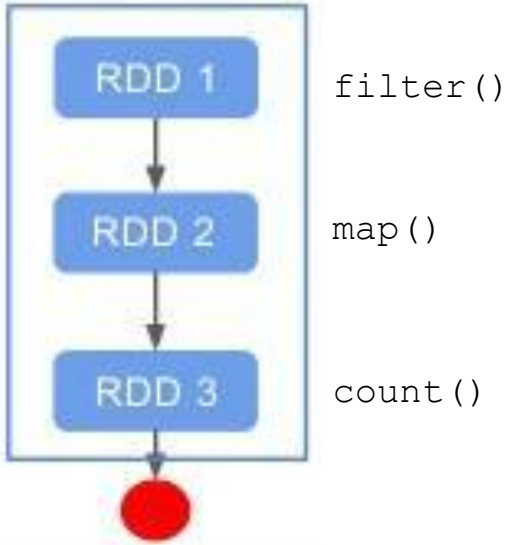


Resilient Distributed Dataset

- In-memory Computation
- Immutability
- Persistence
- Coarse-grained Operations
- Lazy Evaluations
- Partitioning
- Location-Stickiness
- Fault Tolerance

RDD - Resilient Distributed Dataset

```
RDD1   = sc.textFile("hdfs://…);
RDD2   = RDD1.filter(someFunction);
RDD3   = RDD2.map(someFunction);
result = RDD3.count();
```



Casos de uso

Miscelânia de dados

Extract, **T**ransform, **L**oad

Consulta

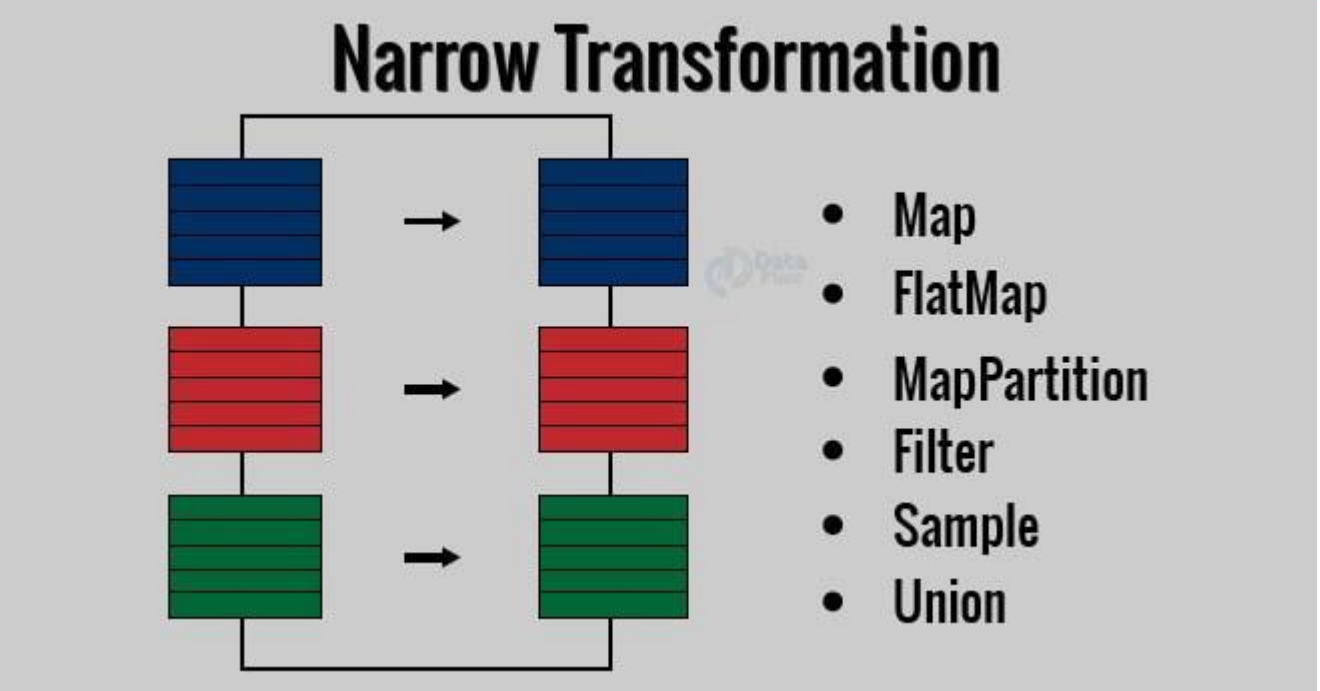
Extract

- Eventos, csv, .xls, logs, estruturados, parceiros, pdf
- Definir um Schema
- Junior limpando dados

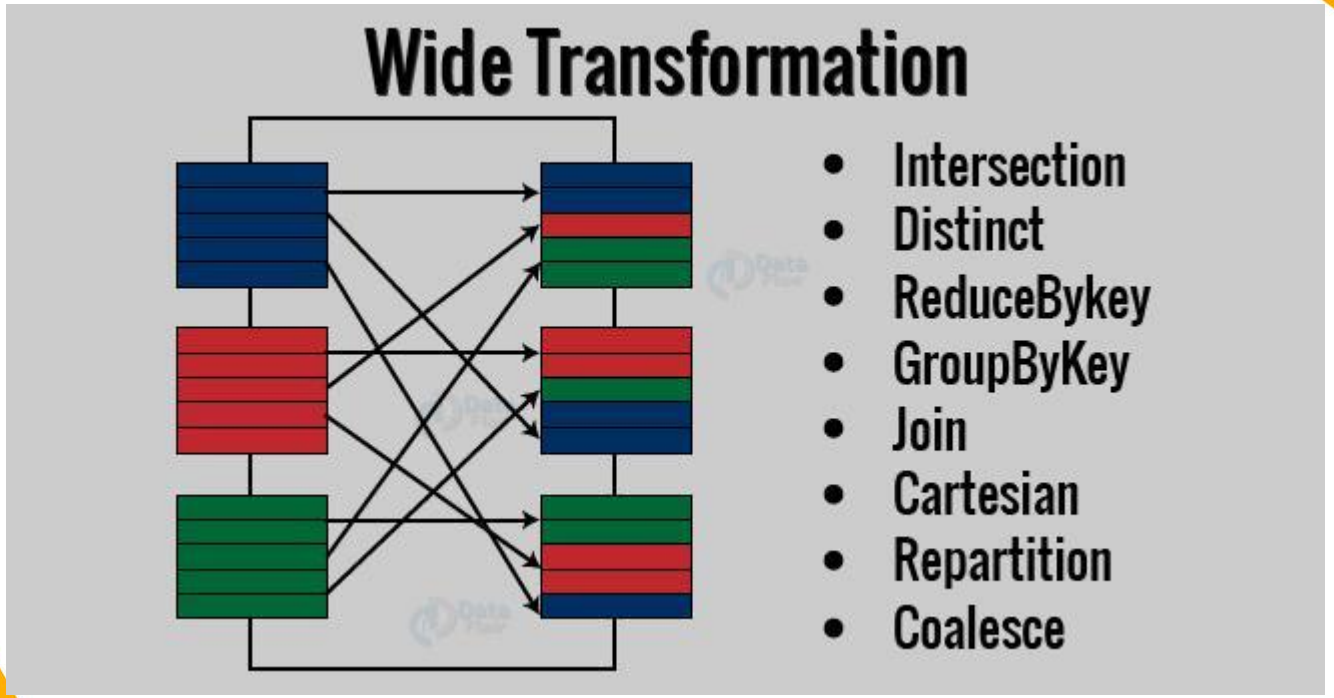
Transform

- Lógica de negócio (medidas, dimensões)
- Limpeza (Male->M, Female->F)
- Filtrar apenas algumas colunas
- Juntar dados de diferentes fontes (lookup, merge)
- Transpor linhas e colunas
- Validações simples ou complexas

Transform


Narrow Transformation
Map
FlatMap
MapPartition
Filter
Sample
Union

Transform


Wide Transformation
Intersection
Distinct
ReduceBykey
GroupByKey
Join
Cartesian
Repartition
Coalesce

Transform - actions

- count
- collect
- take
- top
- countByValue

- reduce
- fold
- aggregate
- foreach

Load

- Data Warehouse
- Dados tabulados prontos para consulta
- Fonte de dados para criar visualizações
- Fonte de dados para treinar modelos
- Cobrir requisitos de recuperação de dados em geral

Quando não usar

- Os dados cabem em uma máquina
- Processamento em tempo real

IoT - Fog computing

Goals: Descentralizar/distribuir armazenamento e processamento

Necessidade:

- Latência baixa
- Processamento paralelo massivo para machine learning
- Algoritmos de analytics para grafos complexos

IoT - Fog computing

Solução: bibliotecas nativas do Apache Spark

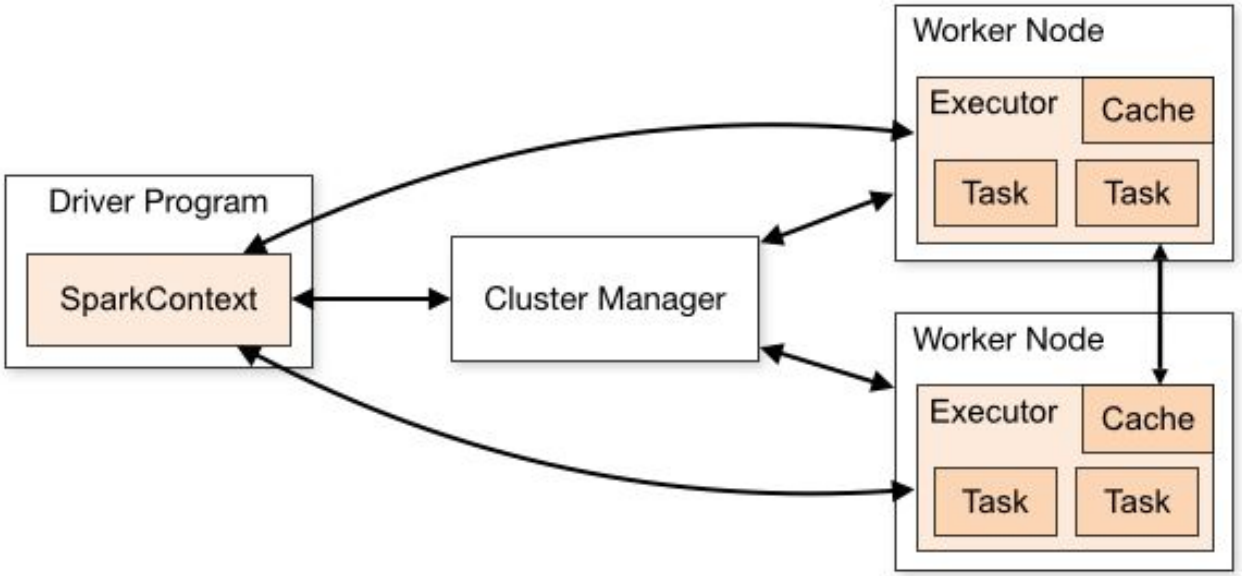
- Spark Streaming
- Spark MLlib
- Spark GraphX

A arquitetura do spark

- Spark não é apenas rápido por conta da utilização de RAM (recorde mundial de ordenação em disco 2014)

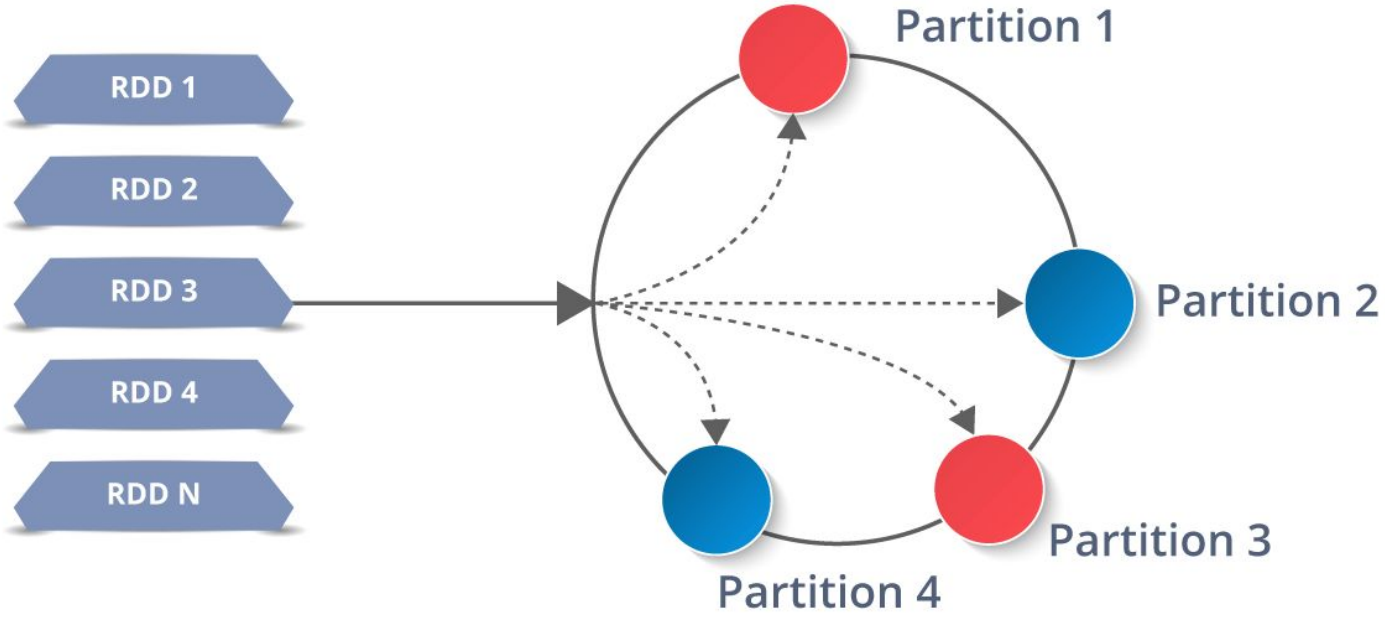
Mas como o Spark consegue fazer tudo isso

Panorama Geral

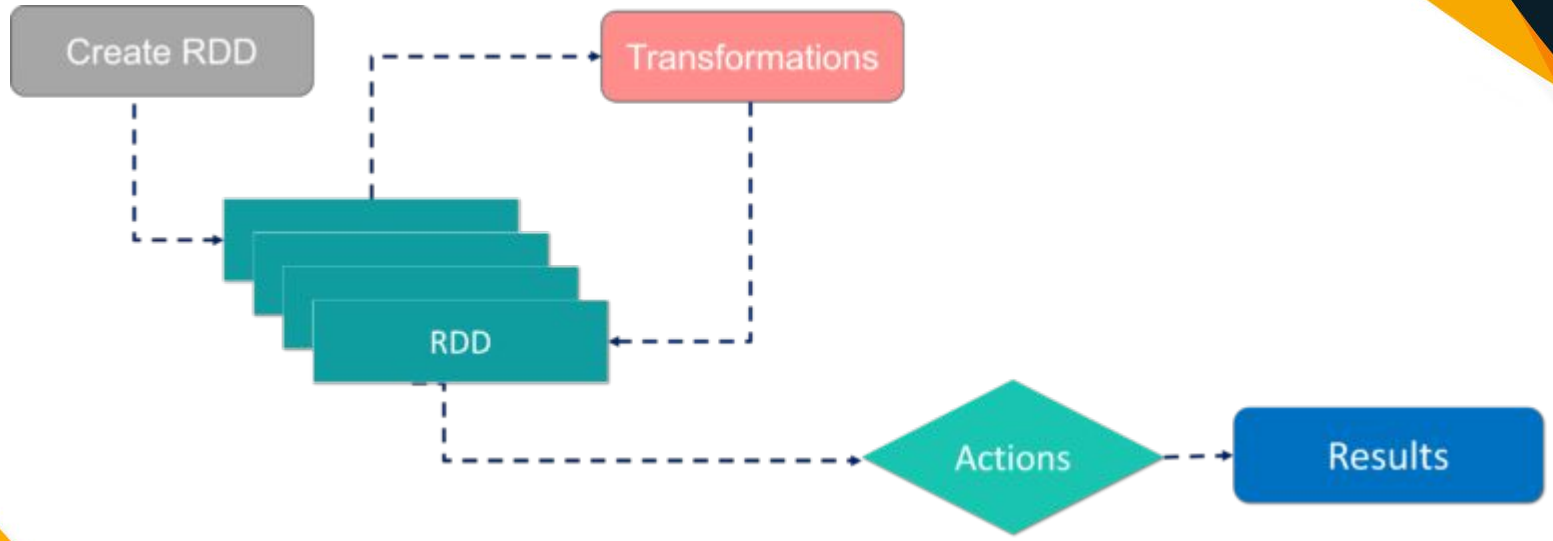


Panorama Geral

Representação dos dados distribuídos RDD



Como RDD permite paralelização



Certo, mas como operações são representadas em termos de RDD ?

- Cada solicitação é modelada em baixo nível como **transformations e actions.**
- Supondo que temos uma tabela **(DES_COUNTRY_NAME, ORIGIN_COUNTRY_NAME,COUNT)**
- Queremos consultar os 5 países mais visitados

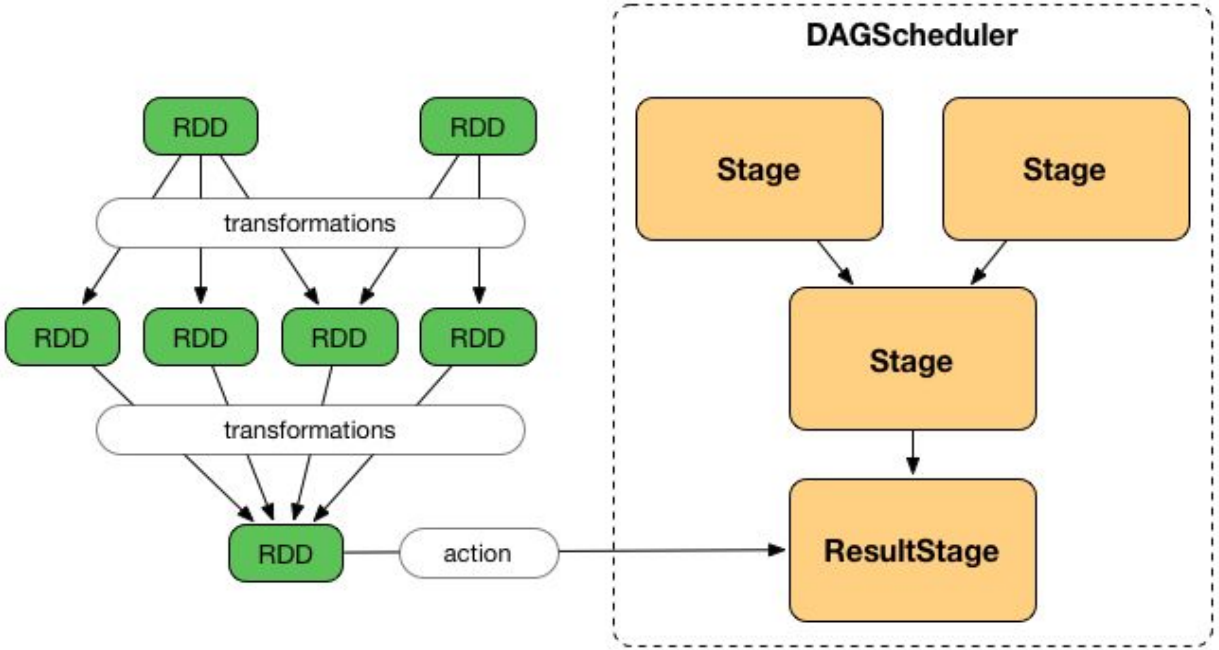
```
flightData2015 = spark\
  .read\
  .option("inferSchema", "true")\
  .option("header", "true")\
  .csv("/data/flight-data/csv/2015-summary.csv")
```

Queremos consultar os 5 países mais visitados

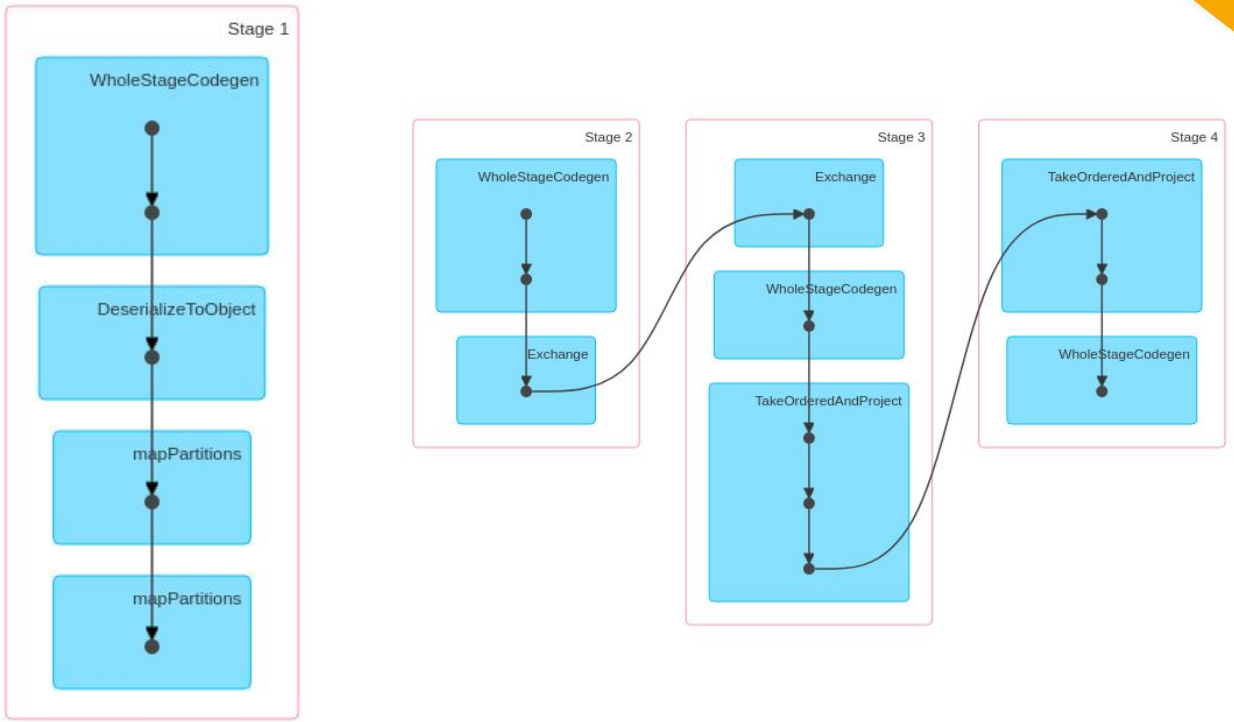
```
flightData2015\
  .groupBy("DEST_COUNTRY_NAME")\
  .sum("count")\
  .withColumnRenamed("sum(count)", "destination_total")\
  .sort(desc("destination_total"))\
  .limit(5)\
  .explain()
```

```
== Physical Plan ==
TakeOrderedAndProject(limit=5, orderBy=[destination_total#16194L DESC], outpu...
+- *HashAggregate(keys=[DEST_COUNTRY_NAME#7323], functions=[sum(count#7325L)])
   +- Exchange hashpartitioning(DEST_COUNTRY_NAME#7323, 5)
      +- *HashAggregate(keys=[DEST_COUNTRY_NAME#7323], functions=[partial_sum...
         +- InMemoryTableScan [DEST_COUNTRY_NAME#7323, count#7325L]
               +- InMemoryRelation [DEST_COUNTRY_NAME#7323, ORIGIN_COUNTRY_NA...
                     +- *Scan csv [DEST_COUNTRY_NAME#7578,ORIGIN_COUNTRY_NAME...
```

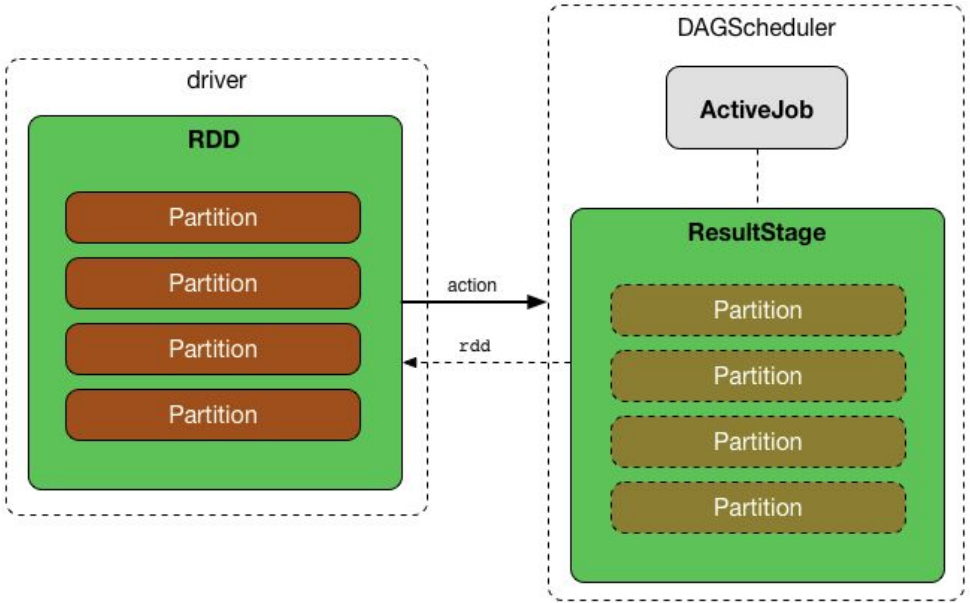
Execução por Etapas e pipelining



DAG gerado pelo comando



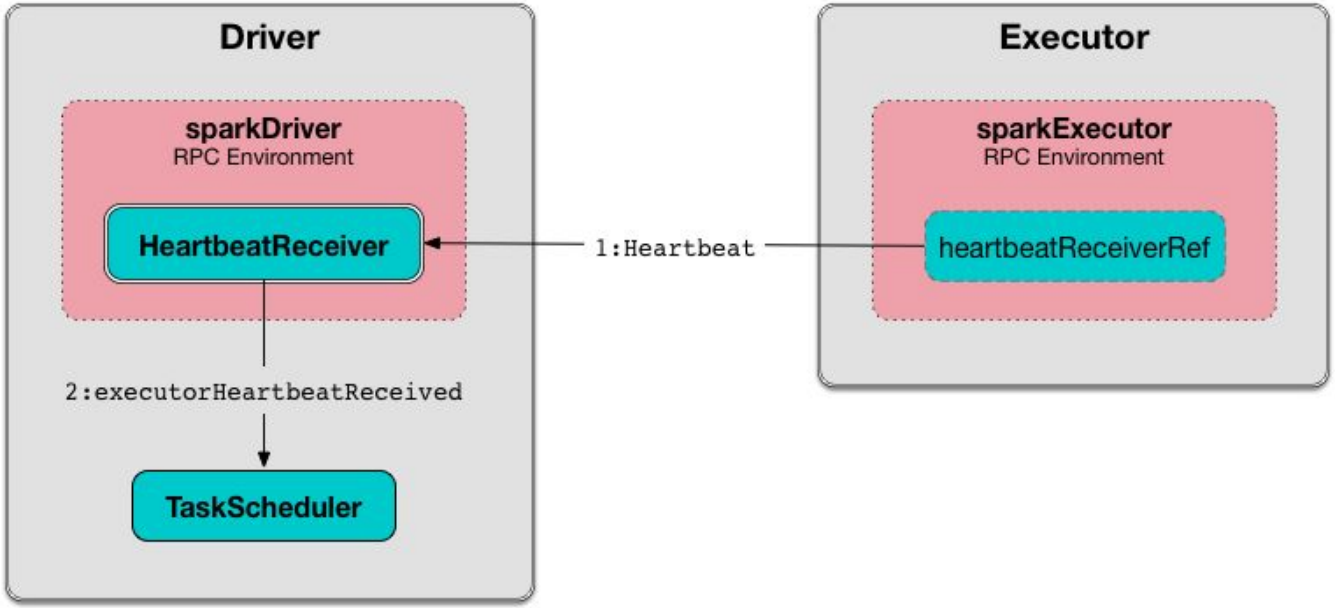
Simplificação do processo



Tolerância a falha a partir do DAG

- Cada RDD tem uma referência ao DAG que o gerou
- Se um executor falha, o cluster manager sabe onde estão as réplicas daquele RDD
- Cluster manager dispara um executor com o RDD anterior àquele no DAG ou com a réplica daquele RDD

Como saber que um executor está off ?

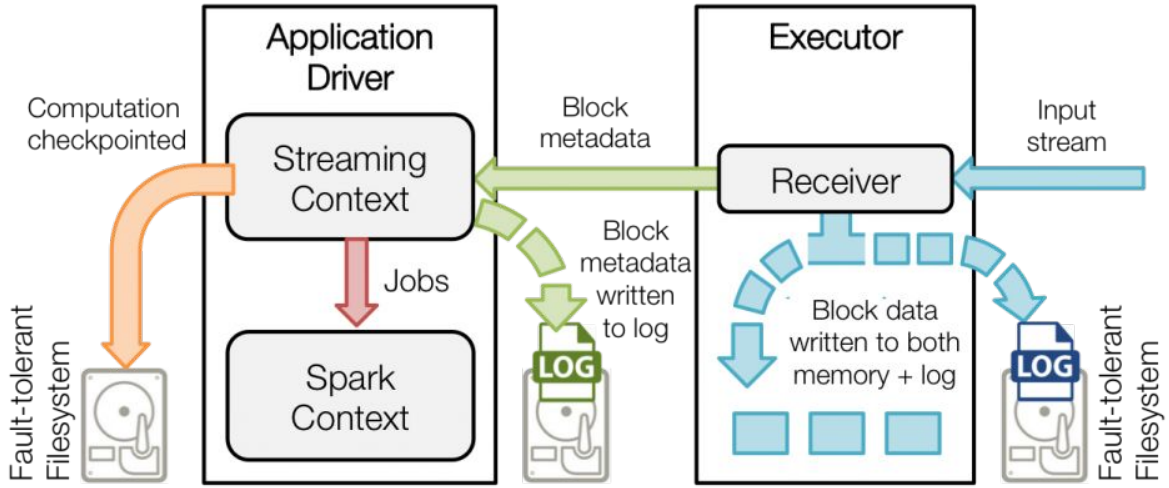


E no caso de falhas nos nós driver e cluster manager ?

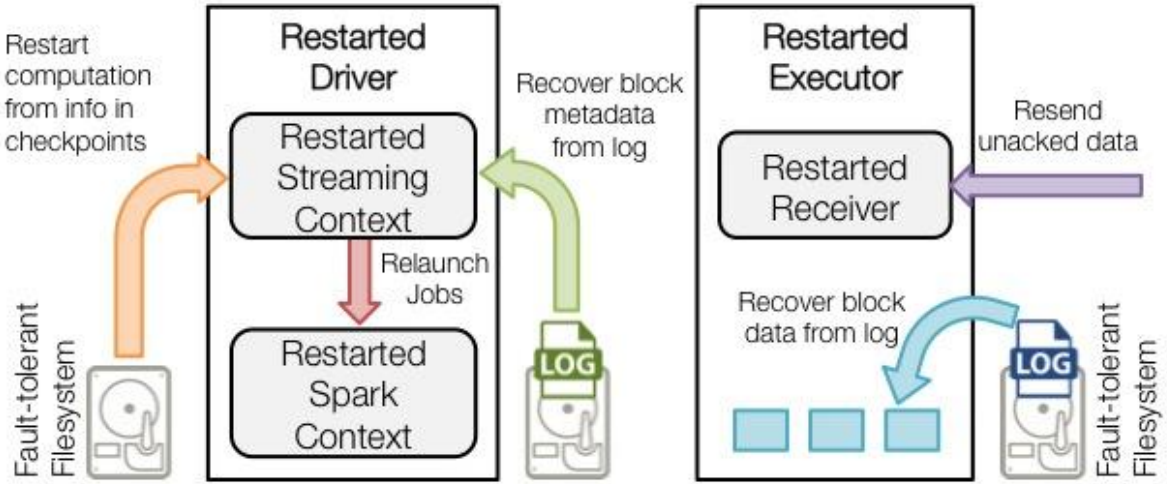
- (CM) Há a possibilidade de efetuar uma eleição de líder por Quórum (Zoo keeper)
- Driver - o driver é a camada da aplicação que está usando o serviço, a falha não é tratada

Como tolerância a falha em dados é tratada em streaming

Tolerância a falha em streaming



Tolerância a falha em streaming



Casos de uso em empresas

Casos de uso em empresas

- Spark usado estrategicamente desde o startup
- Batch data processing offline e consultas
- Dados processados alimentarão sistemas de tempo real
- Data lake para análise, extração de conhecimento e IA
- Treinamento de modelos de recomendação
- Treinamento de modelos de IA para análise de fraude
- Diminui o processamento de 1 mês para 2 horas

Casos de uso em empresas

Nubank e Dafiti early adopters

Início do nubank: queries no Datomic

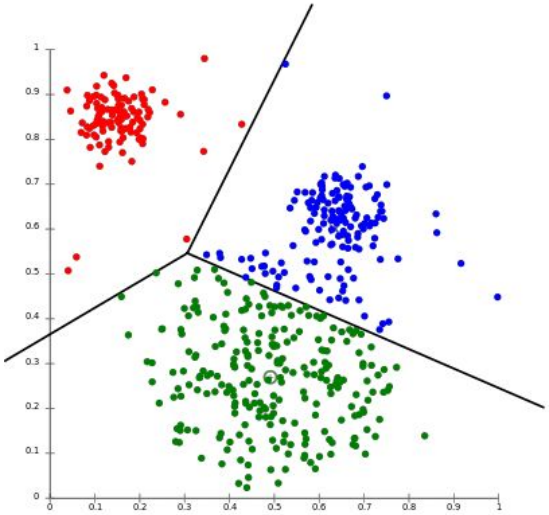
- Visualizações em terceiros (Tableau, D3, ...)
- Ambiente não amigável
- Sem suporte à linguagens

Casos de uso em empresas

Início do time de dados no nubank: Apache Spark

- Microserviço manda o streaming do Datomic para a S3
- Spark (mesos) e storage (S3) na AWS mínima
- DSL em scala para aplicações Spark
- ETL diária com output no data lake
- Subscribe das tabelas no metastore do Databricks
- Interface Databricks pra consulta consumindo S3

Um pequeno Exemplo de K-Means no databricks



Referências

https://www.datawarehouse4u.info/ETL-process.html/
https://www.splicemachine.com/future-data-integration-is-no-etl/
https://www.qubole.com/blog/apache-spark-use-cases/
https://www.dezyre.com/article/top-5-apache-spark-use-cases/271
https://data-flair.training/blogs/apache-spark-ecosystem-components/
https://www.npntraining.com/blog/apache-spark-use-cases/
http://shop.oreilly.com/product/0636920034957.do
https://techvidvan.com/tutorials/fault-tolerance-in-spark/

https://br.linkedin.com/in/fabianofernandes
https://br.linkedin.com/in/caiquelima