DIGITAL INNOVATION ONE

Aceleração Global Dev #4 everis

Criando pipelines de dados eficientes - Parte 1
Spark Streaming

Marco Antonio Pereira
Expert Data Architect

DIGITAL INNOVATION ONE

Objetivos da Aula

1. Apresentar Apache Spark

2. StreamContext

3. Casos de Uso

DIGITAL INNOVATION ONE

Requisitos Básicos

- ✓ Máquina Virtual com Apache Spark 2.4
- ✓ Conhecimentos básicos em Python ou Scala
- ✓ Paciência

DIGITAL
INNOVATION
ONE

Parte 1: Sobre

Criando pipelines de dados eficientes - Parte 1
Spark Streaming

DIGITAL INNOVATION ONE

O que é?

- Apache Spark é um projeto de processamento de dados distribuído de código aberto que foi iniciado em 2009
- Spark foi originalmente de código aberto
- O projeto Spark tem mais de 400 indivíduos contribuidores e committers de empresas como Facebook, Yahoo, Intel, Netflix, Databricks e outras
- Spark é escrito em Scala, que é construído em cima da Java Virtual Machine (JVM) e Java runtime
- Capaz de ser executado no Windows e também no Linux
- Mais de 500 organizações utilizam Spark

DIGITAL INNOVATION ONE

O que é?

Spark como uma abstração

O Spark permite que os desenvolvedores criem rotinas de processamento de dados complexas e de vários estágios, fornecendo uma API de alto nível e uma estrutura tolerante a falhas que permite aos programadores se concentrar na lógica em vez de questões ambientais ou de infraestrutura, como falha de hardware.

Spark é rápido, eficiente e escalonável

O Spark implementa uma estrutura distribuída e tolerante a falhas na memória chamada Resilient Distributed Dataset (RDD) . O Spark maximiza o uso de memória em várias máquinas, melhorando o desempenho geral em ordens de magnitude. A reutilização dessas estruturas na memória pelo Spark o torna adequado para operações iterativas de aprendizado de máquina, bem como para consultas interativas.

DIGITAL INNOVATION ONE

Aplicabilidade

- Operações de extração-transformação-carregamento (load) (ETL)
- Análise preditiva e aprendizado de máquina
- Operações de acesso a dados (como consultas e visualizações SQL)
- Mineração e processamento de texto
- Processamento de eventos em tempo real
- Aplicativos gráficos
- Reconhecimento de padrões
- Mecanismos de recomendação

DIGITAL INNOVATION ONE

Programação

- Scala
- Python
- Java
- SQL
- R

DIGITAL INNOVATION ONE

Programação

Exemplos RDD:

Python

```
rddArquivo = sc.textFile ("hdfs://meucluster/data/arquivo.txt")
```

Scala

```
val rddArquivo = sc.textFile("hdfs:// meucluster/data/arquivo.txt")
```

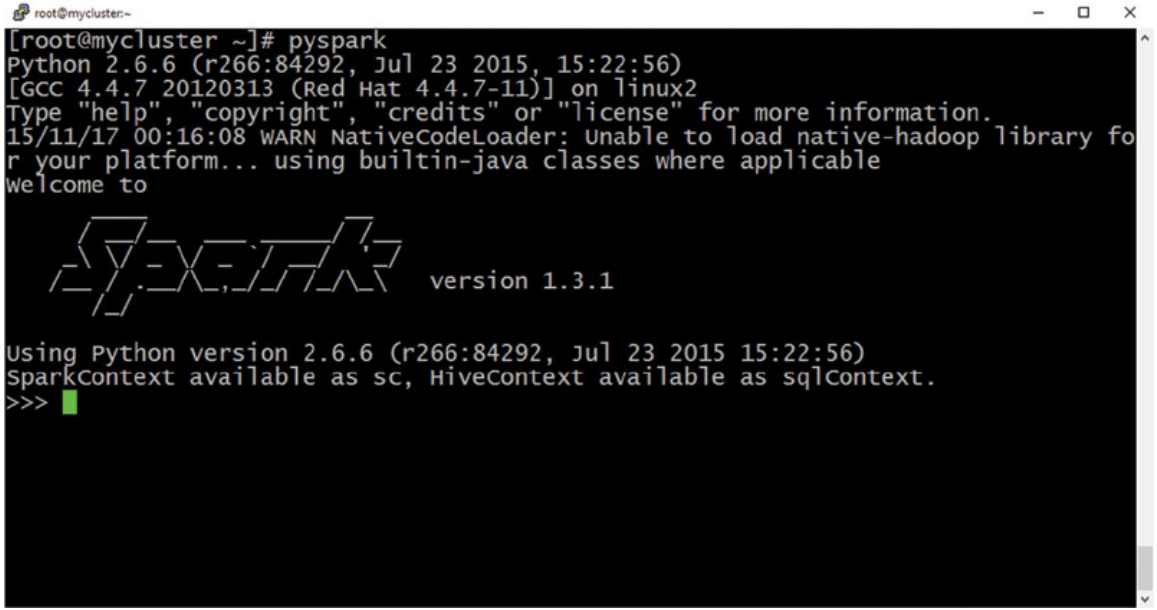
Java

```
JavaRDD<String> rddArquivo = sc.textFile("hdfs://meucluster/data/arquivo.txt");
```

DIGITAL INNOVATION ONE

Uso interativo

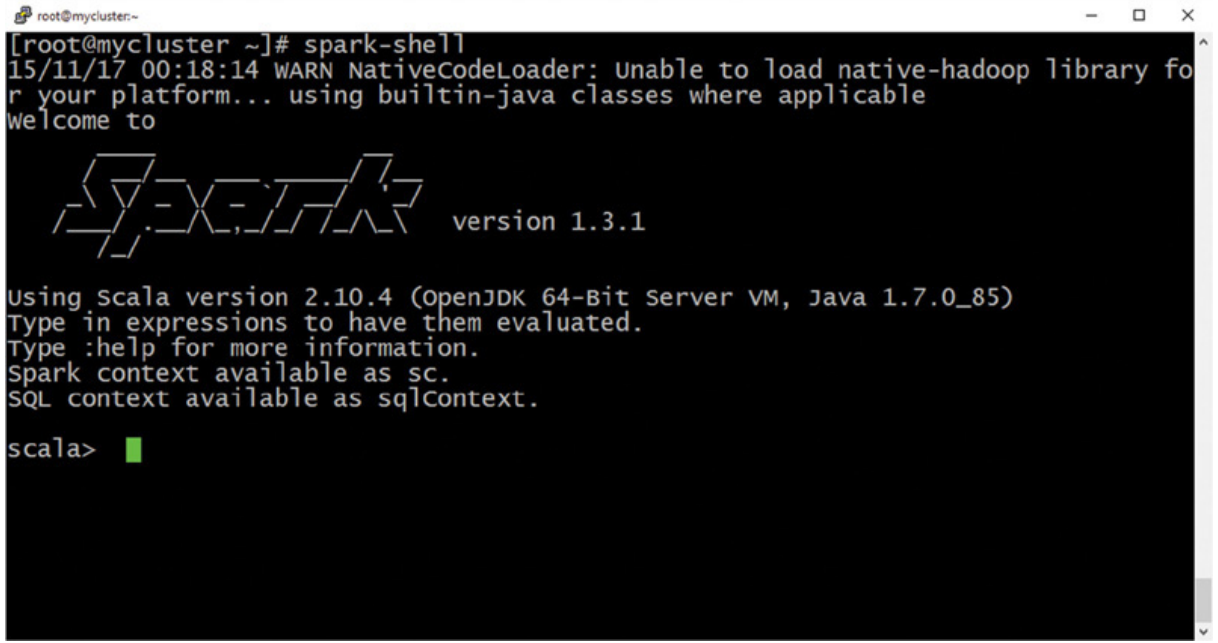
PySpark:



DIGITAL INNOVATION ONE

Uso interativo

Scala:



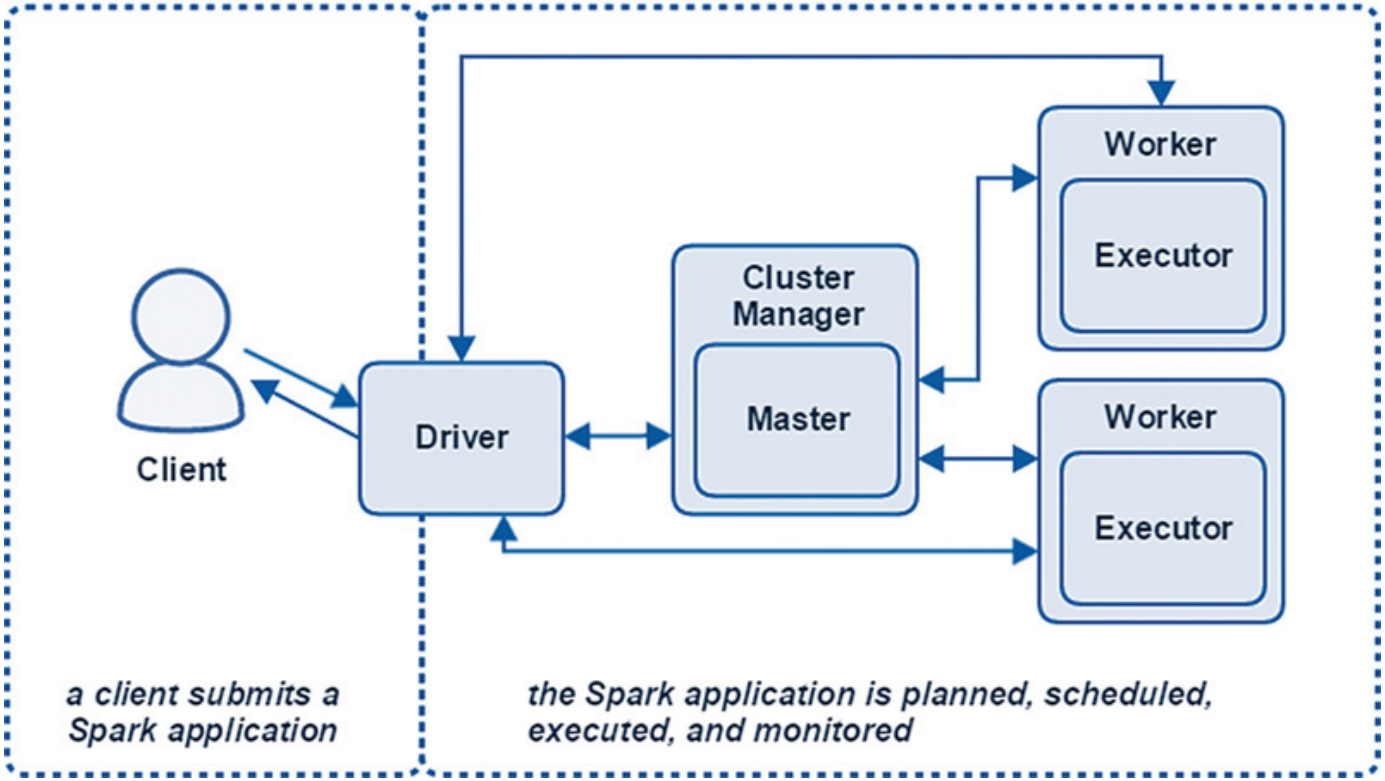
Uso não-interativo

spark-submit:

```
[linux@user ~]$ spark-submit \
--master yarn \
--queue "SquadFI" \
--name "[Programa Spark 123] ETL 999" \
--driver-memory 2G \
--executor-memory 2G \
--executor-cores 1 \
--proxy-user hive \
--conf "spark.driver.maxResultSize=16g" \
--conf "spark.dynamicAllocation.enabled=true" \
--conf "spark.shuffle.service.enabled=true" \
--conf "spark.shuffle.service.port=7337" \
--conf "spark.dynamicAllocation.initialExecutors=10" \
--conf "spark.dynamicAllocation.minExecutors=10" \
--conf "spark.dynamicAllocation.maxExecutors=80" \
--conf "spark.yarn.driver.memoryOverhead=2000" \
--conf "spark.yarn.executor.memoryOverhead=2000" \
--driver-java-options "-Djavax.security.auth.useSubjectCredsOnly=false" \
--jars commons-csv-1.2.jar,spark-csv_2.11-1.5.0.jar \
Main.py <parametro 1> <parametro 2> <parametro N>
```

DIGITAL INNOVATION ONE

Anatomia



1. Driver
2. Cluster Manager
3. Spark Master
4. Executors

DIGITAL INNOVATION ONE

Anatomia

DRIVER

A vida útil do aplicativo Spark começa (e termina) com o driver Spark. O driver Spark é o processo que os clientes usam para enviar aplicativos no Spark. O driver também é responsável por planejar e coordenar a execução do programa Spark e retornar o status e/ou resultados (dados) ao cliente.

O driver Spark é responsável por criar o SparkContext. O SparkContext, é referido como sc, é a instância do aplicativo que representa a conexão com o Master do Spark (e os Workers do Spark). O SparkContext é instanciado no início de um aplicativo Spark (incluindo os shells interativos) e é usado para todo o programa.

- Uma das principais funções do driver é planejar o aplicativo.
- O driver obtém a entrada de processamento do aplicativo e planeja a execução do programa.
- O driver pega todas as transformações solicitadas (operações de manipulação de dados) e ações (solicitações de saída ou um prompt para executar o programa) e cria um Gráfico Acíclico Direcionado (Directed Acyclic Graph - DAG).

DIGITAL INNOVATION ONE

Anatomia

DAG

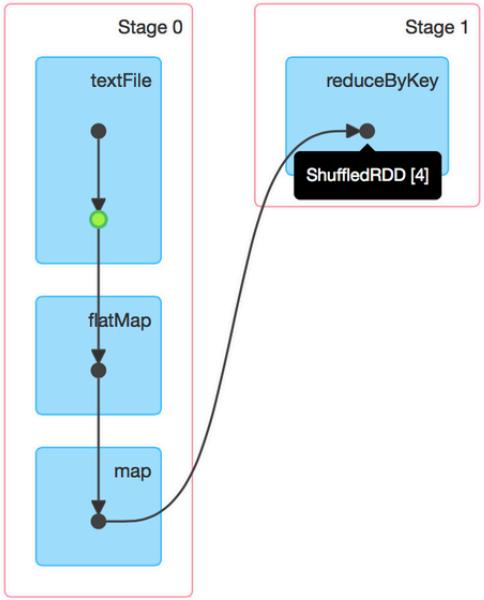
Details for Job 0

Status: SUCCEEDED

Completed Stages: 2

▸ Event Timeline

▾ DAG Visualization



DIGITAL INNOVATION ONE

Anatomia

CLUSTER MANAGER

O Cluster Manager é o processo responsável por monitorar os nós Workers e reservar recursos nesses nós mediante solicitação do Spark Master. O Spark Master, por sua vez, disponibiliza esses recursos de cluster para o Driver na forma de executores.

O Cluster Manager pode ser separado do processo Master. Esse é o caso ao executar o Spark no Mesos ou YARN.

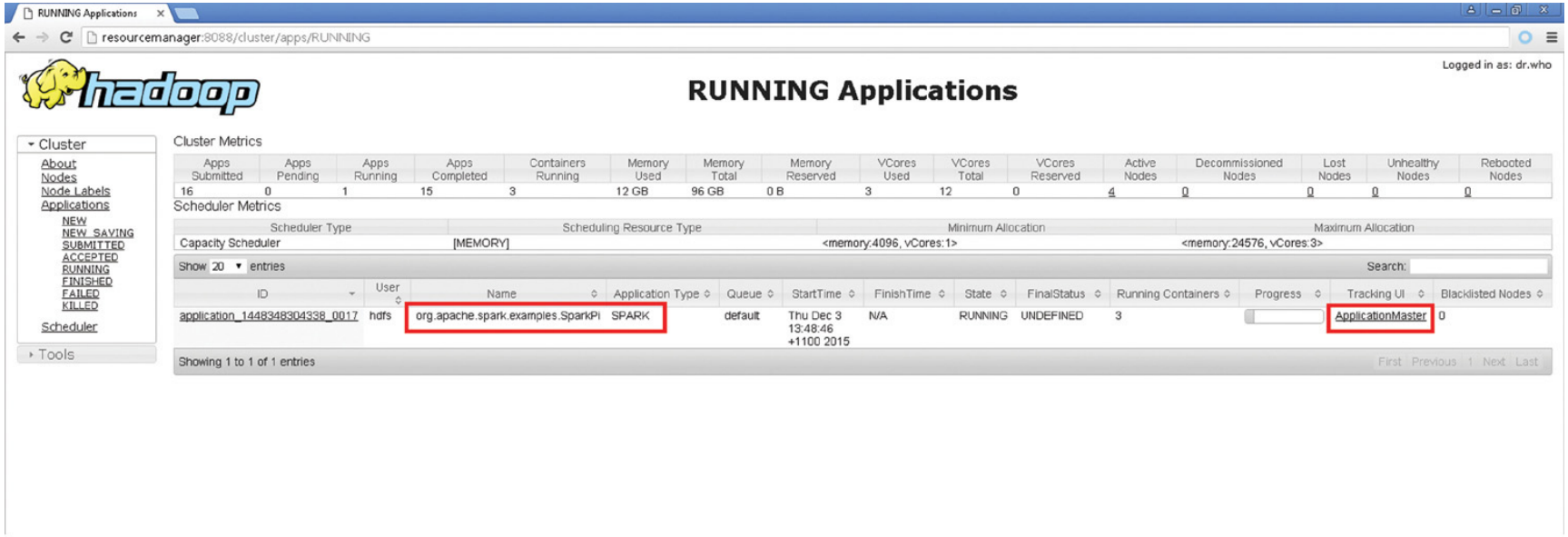
No caso do Spark rodando no modo Standalone, o processo Master também executa as funções do Cluster Manager. Efetivamente, ele atua como seu próprio gerenciador de cluster.

- O Cluster Manager em um aplicativo Spark distribuído é o processo que governa, monitora e reserva recursos na forma de contêineres em nós de worker de cluster (ou escravos). Esses contêineres são reservados mediante solicitação do Spark Master. O Cluster Manager no caso do Spark no YARN é o YARN ResourceManager.
- Um Spark Driver em execução no modo YARN envia um aplicativo ao ResourceManager e, em seguida, o ResourceManager designa um ApplicationsMaster para o aplicativo Spark.

Anatomia

CLUSTER MANAGER

Exemplo de um aplicativo Spark gerenciado pelo YARN mostrado no ResourceManager UI, normalmente disponível em http://<resource_manager>:8088



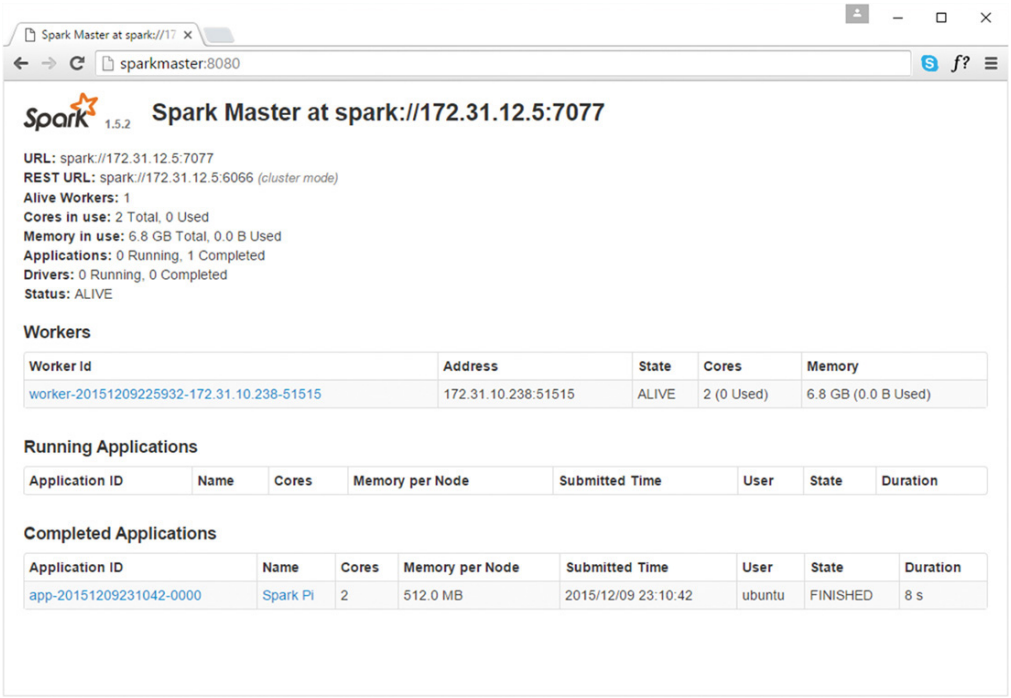
DIGITAL INNOVATION ONE

Anatomia

SPARK MASTER

O Spark master é o processo que solicita recursos no cluster e os disponibiliza para o driver Spark. Em qualquer modo de implantação, o mestre negocia recursos ou contêineres com nós de trabalho ou nós escravos e rastreia seu status e monitora seu progresso.

O processo mestre do Spark atende a uma interface de usuário da web na porta 8080 no host mestre:



DIGITAL INNOVATION ONE

Anatomia

SPARK MASTER

- O processo ApplicationsMaster que é instanciado pelo ResourceManager no envio do aplicativo Spark atua como o Spark Master. O Driver informa o ApplicationsMaster sobre os requisitos de seu executor para o aplicativo. O ApplicationsMaster, por sua vez, solicita containers (que são hospedados em NodeManagers) do ResourceManager para hospedar esses executores.

- O ApplicationsMaster é responsável por gerenciar esses containers (executors) durante o ciclo de vida do aplicativo Spark.

- O Driver coordena o estado do aplicativo e as transições de estágio de processamento.

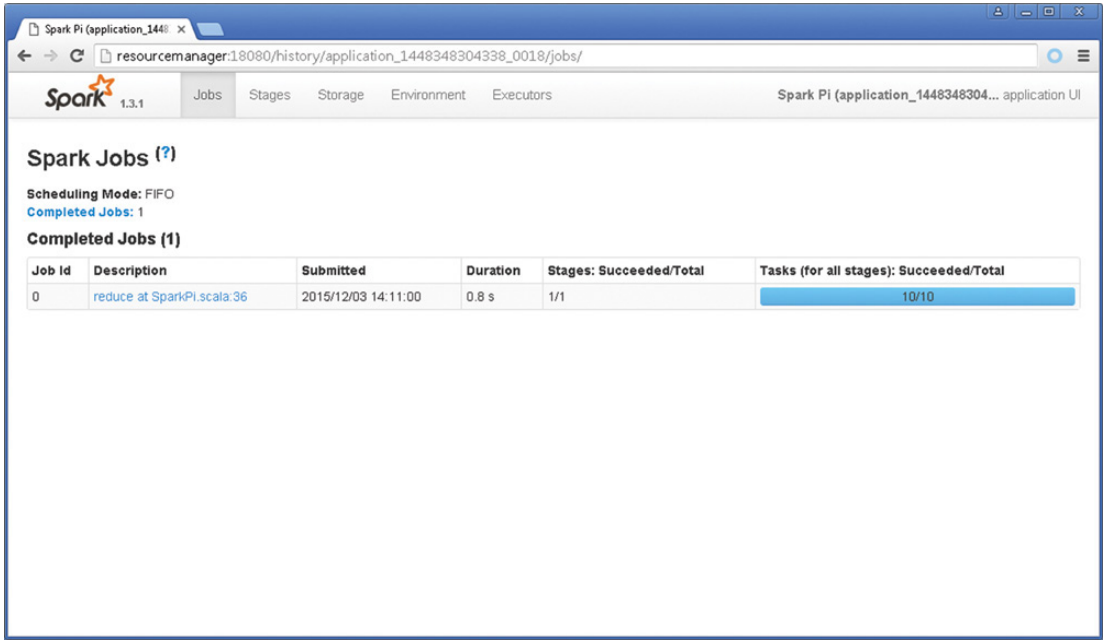
- O próprio ApplicationsMaster é hospedado em uma JVM em um nó slave ou worker no cluster e é, na verdade, o primeiro recurso alocado para qualquer aplicativo Spark em execução no YARN.

DIGITAL INNOVATION ONE

Anatomia

SPARK MASTER

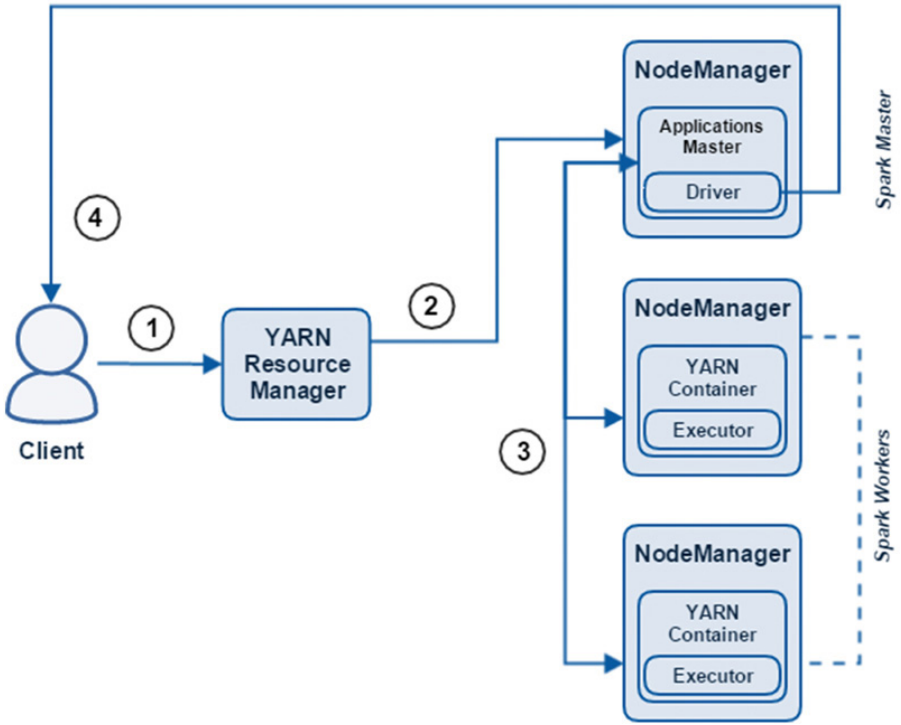
A UI do ApplicationsMaster para um aplicativo Spark é a UI do Spark Master. Isso está disponível como um link na UI do ResourceManager:



DIGITAL INNOVATION ONE

Anatomia

Modo Cluster



DIGITAL INNOVATION ONE

Anatomia

Modo Cluster (Yarn-Cluster)

1. O Client (uma chamada de processo do usuário via spark-submit) envia um aplicativo Spark para o Cluster Manager (o YARN ResourceManager).

2. O ResourceManager atribui um ApplicationsMaster (o Spark Master) para o aplicativo. O processo do Driver é criado no mesmo nó.

3. O ApplicationsMaster solicita containers para os Executors do ResourceManager. Os contêineres são atribuídos e os executors são gerados. O Driver então se comunica com os executors para organizar o processamento de tarefas e estágios do programa Spark.

4. O driver retorna o progresso, resultados e status para o cliente.

Iniciando

[linux@user ~]$ pyspark --master local

Option	Description
`Local`	Runs the application in local mode using *one* core.
`local[n]`	Runs the application in local mode using *n* cores. An asterisk (*) can be used as *n* to enable *all cores* on the system to be used.
`local[n,m]`	Runs the application in local mode using *n* cores, retrying failed tasks *m* times. An asterisk can be used as *n* to enable *all cores* on the system to be used.

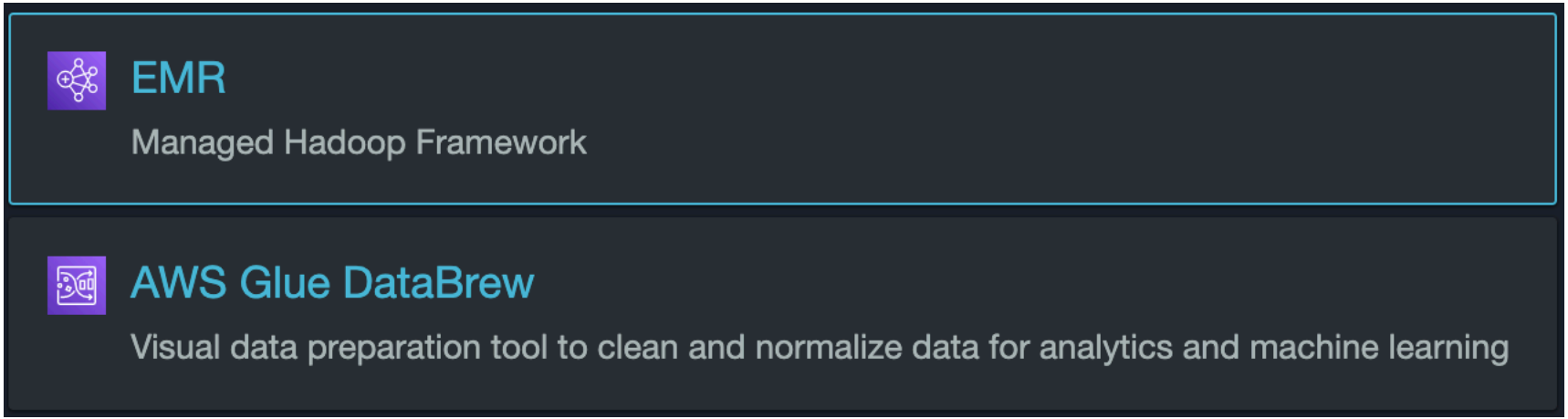
Modo não interativo (aplicativo em Spark-submit)

```
from pyspark import SparkContext
sc = SparkContext ("local [*]")
#o código do seu aplicativo ...
```

DIGITAL INNOVATION ONE

Cloud

AWS



DIGITAL INNOVATION ONE

Cloud

AWS-EMR (Elastic MapReduce)

Configuração de software

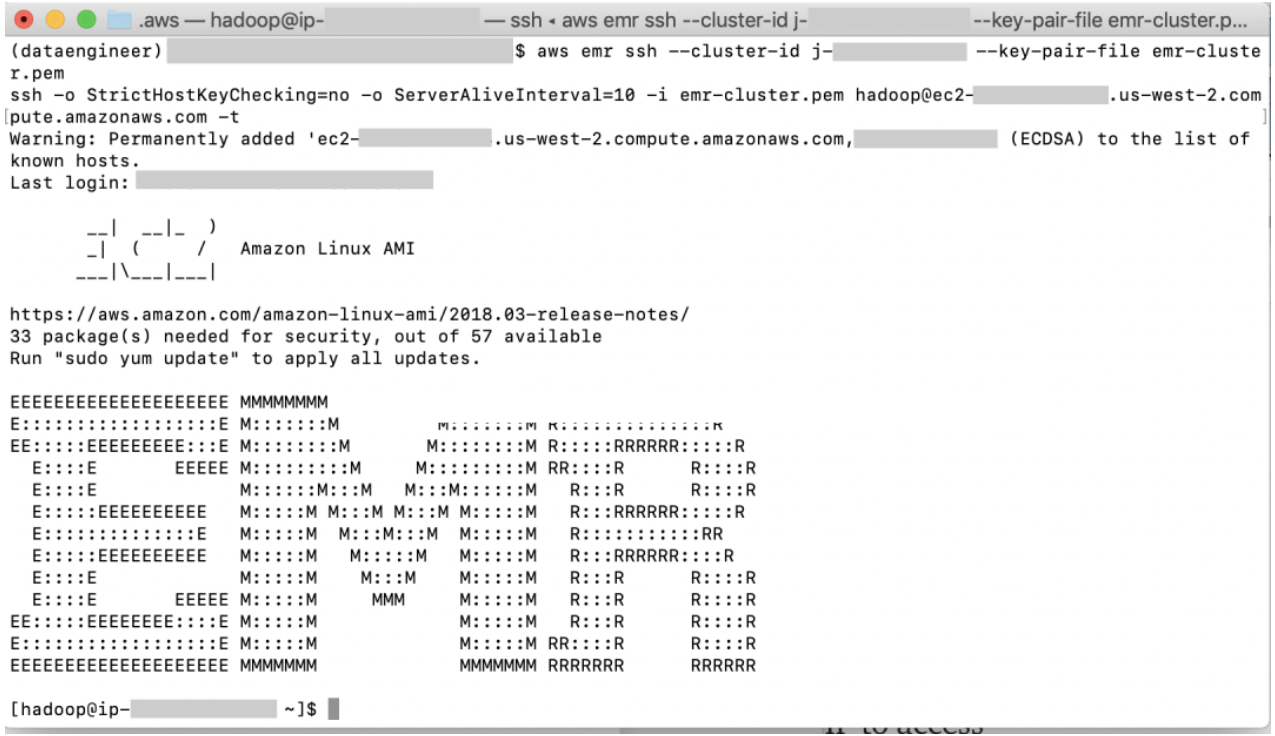
Versão emr-6.2.0

- [x] Hadoop 3.2.1
- [ ] JupyterHub 1.1.0
- [ ] Ganglia 3.7.2
- [x] Hive 3.1.2
- [ ] ZooKeeper 3.4.14
- [ ] Sqoop 1.4.7
- [ ] Oozie 5.2.0
- [ ] TensorFlow 2.3.1
- [ ] Zeppelin 0.9.0
- [ ] Tez 0.9.2
- [ ] HBase 2.2.6-amzn-0
- [ ] Presto 0.238.3
- [ ] JupyterEnterpriseGateway 2.1.0
- [x] Hue 4.8.0
- [x] Spark 3.0.1
- [ ] Livy 0.7.0
- [ ] Flink 1.11.2
- [ ] Pig 0.17.0
- [ ] PrestoSQL 343
- [ ] MXNet 1.7.0
- [ ] Phoenix 5.0.0
- [ ] HCatalog 3.1.2

DIGITAL INNOVATION ONE

Cloud

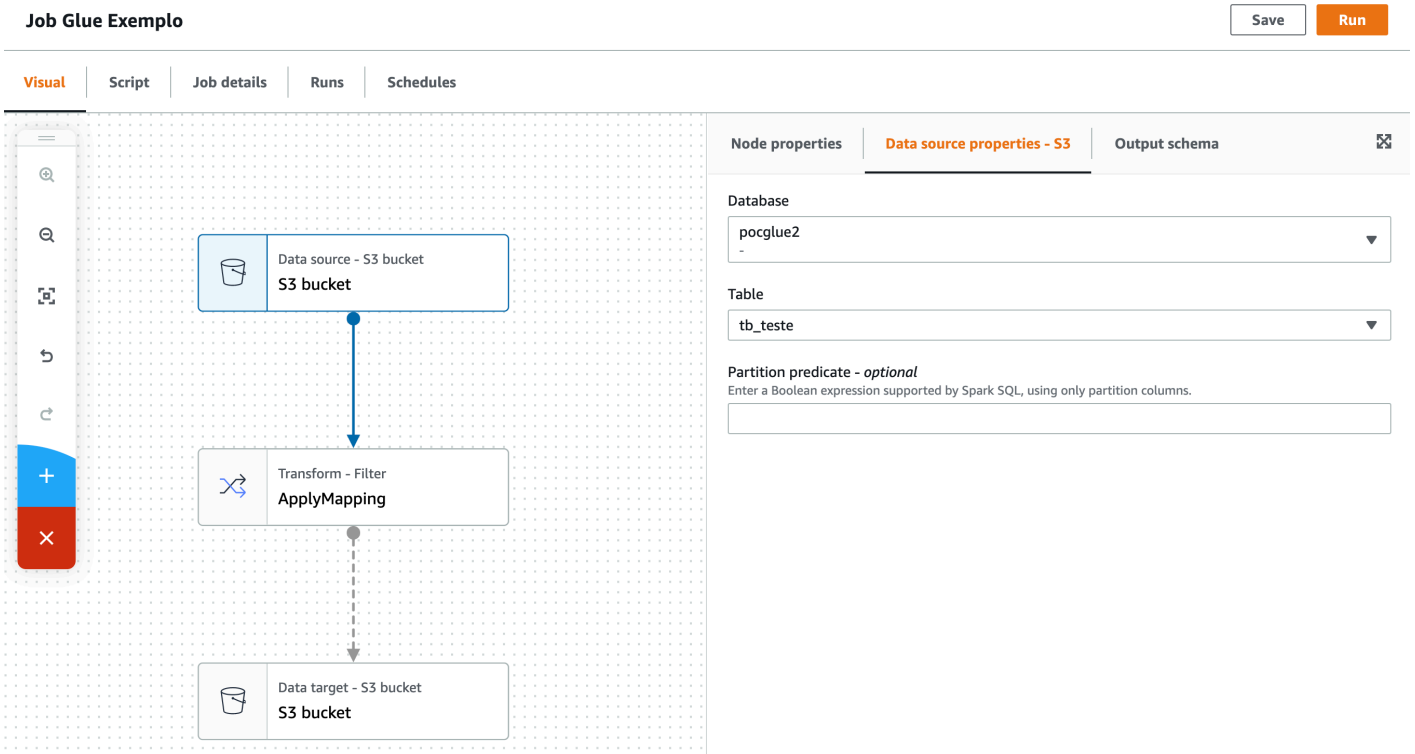
AWS-EMR (Elastic MapReduce)



DIGITAL INNOVATION ONE

Cloud

AWS-GLUE



DIGITAL INNOVATION ONE

Cloud

AWS-GLUE

Job Glue Exemplo

Save | Run

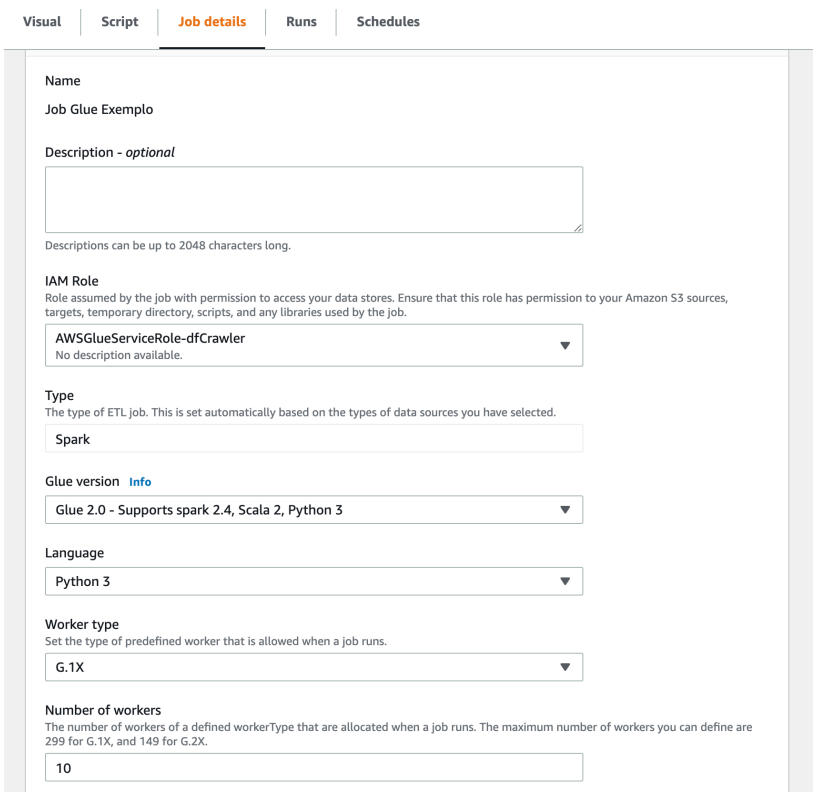
Visual | Script | Job details | Runs | Schedules

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
import re

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
## @type: DataSource
## @args: [database = "pocglue2", table_name = "tb_teste", transformation_ctx = "DataSource0"]
## @return: DataSource0
## @inputs: []
DataSource0 = glueContext.create_dynamic_frame.from_catalog(database = "pocglue2", table_name = "tb_teste", transformation_ctx = "DataSource0")
## @type: Filter
## @args: [f = lambda row : (row["altura"] < 40), transformation_ctx = "Transform0"]
## @return: Transform0
## @inputs: [frame = DataSource0]
Transform0 = Filter.apply(frame = DataSource0, f = lambda row : (row["altura"] < 40), transformation_ctx = "Transform0")
## @type: DataSink
## @args: [connection_type = "s3", format = "json", connection_options = {"path": "s3://saida11/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
## @return: DataSink0
## @inputs: [frame = Transform0]
DataSink0 = glueContext.write_dynamic_frame.from_options(frame = Transform0, connection_type = "s3", format = "json", connection_options = {"path": "s3://s
job.commit()
```

DIGITAL INNOVATION ONE

Cloud

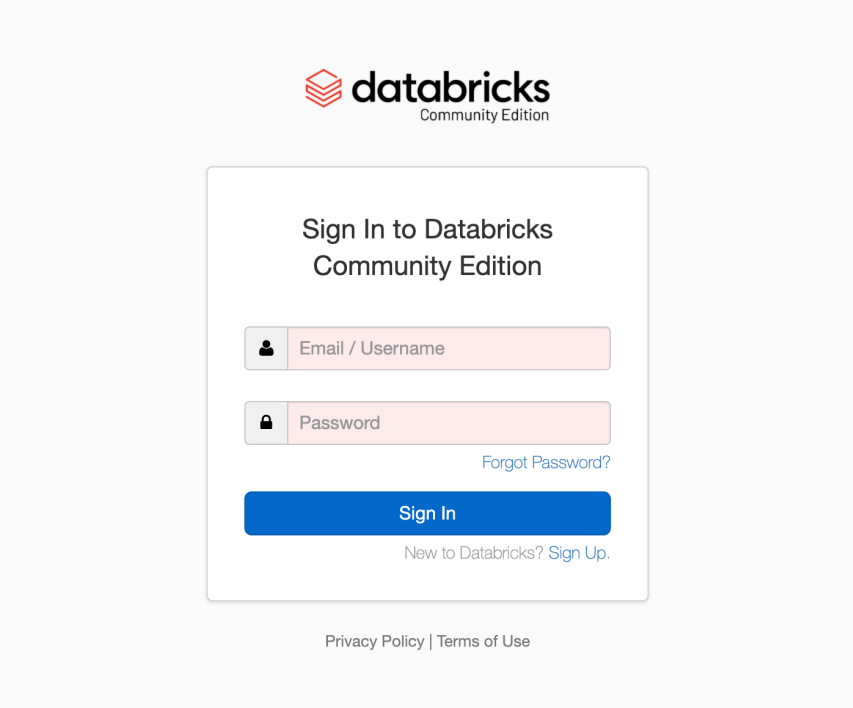
AWS-GLUE



DIGITAL INNOVATION ONE

Cloud

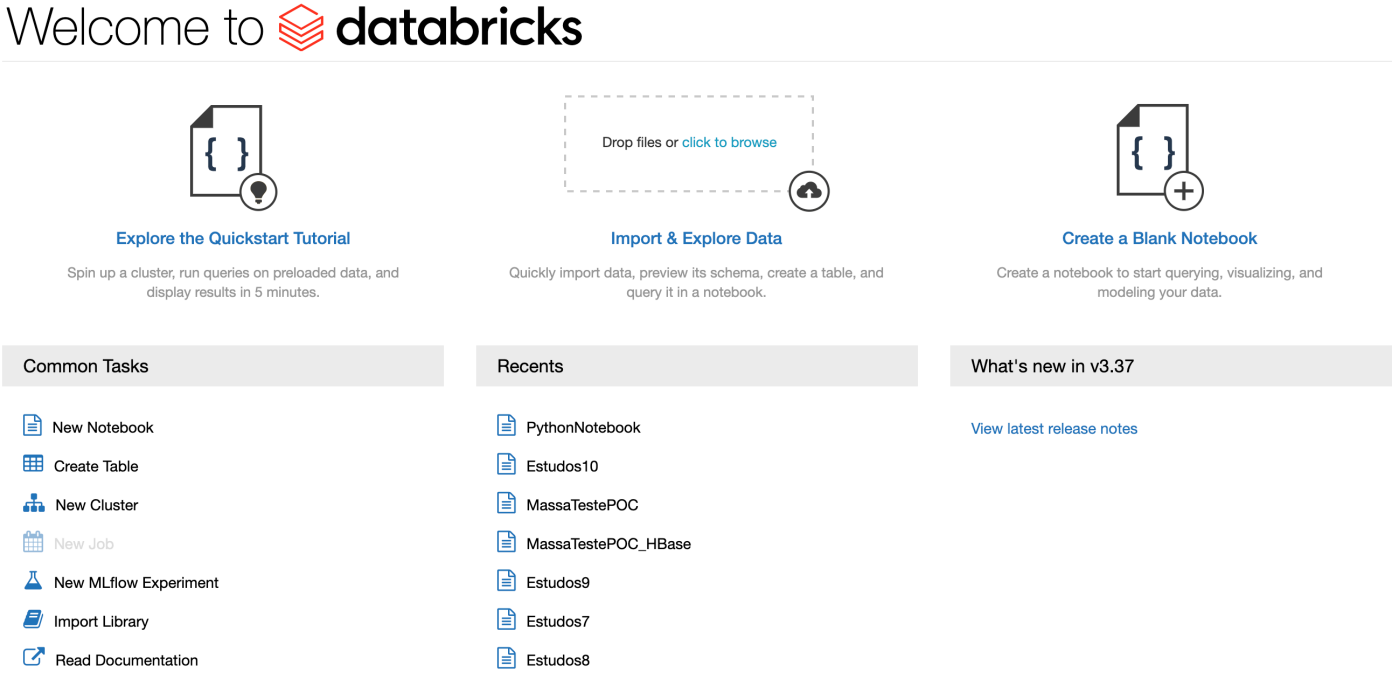
Databricks Community (https://community.cloud.databricks.com/)



DIGITAL INNOVATION ONE

Cloud

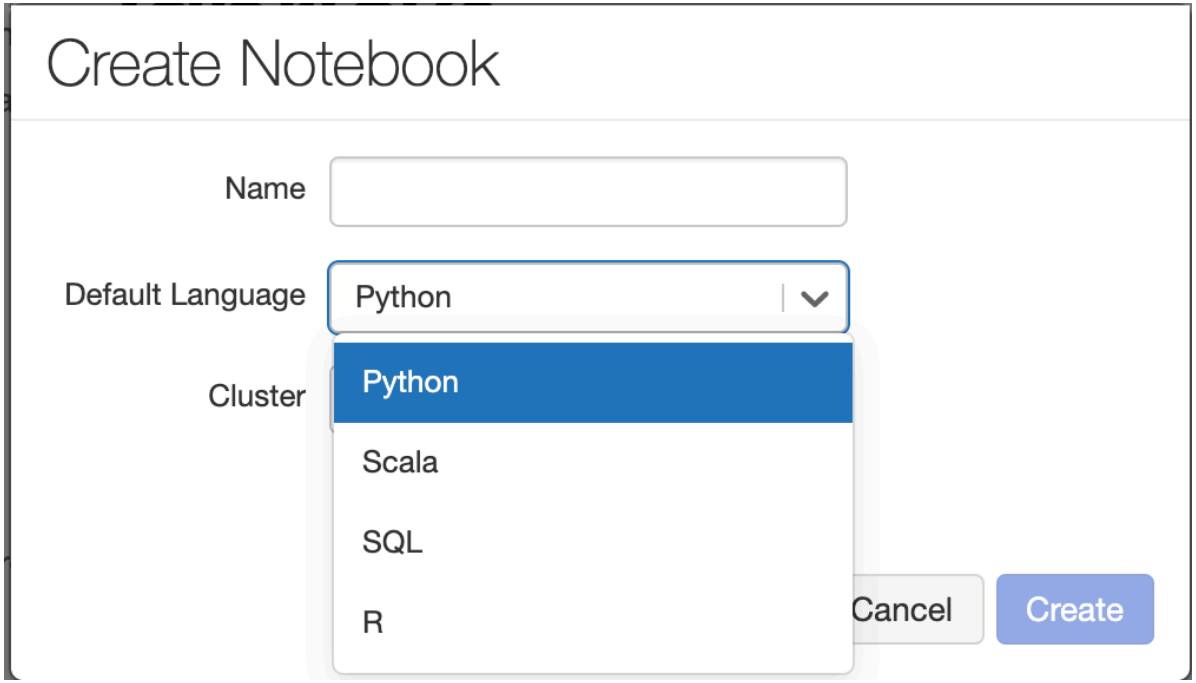
Databricks Community (https://community.cloud.databricks.com/)



DIGITAL INNOVATION ONE

Cloud

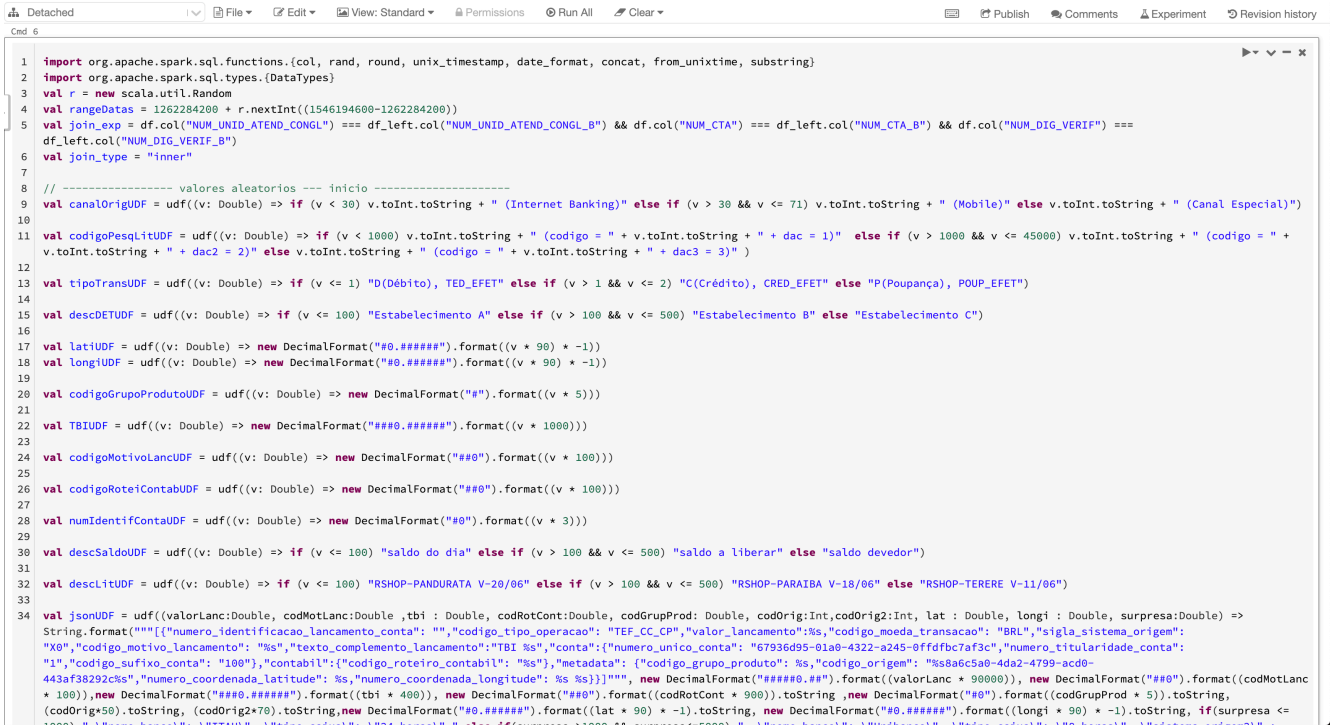
Databricks Community (https://community.cloud.databricks.com/)



DIGITAL INNOVATION ONE

Cloud

Databricks Community (https://community.cloud.databricks.com/)

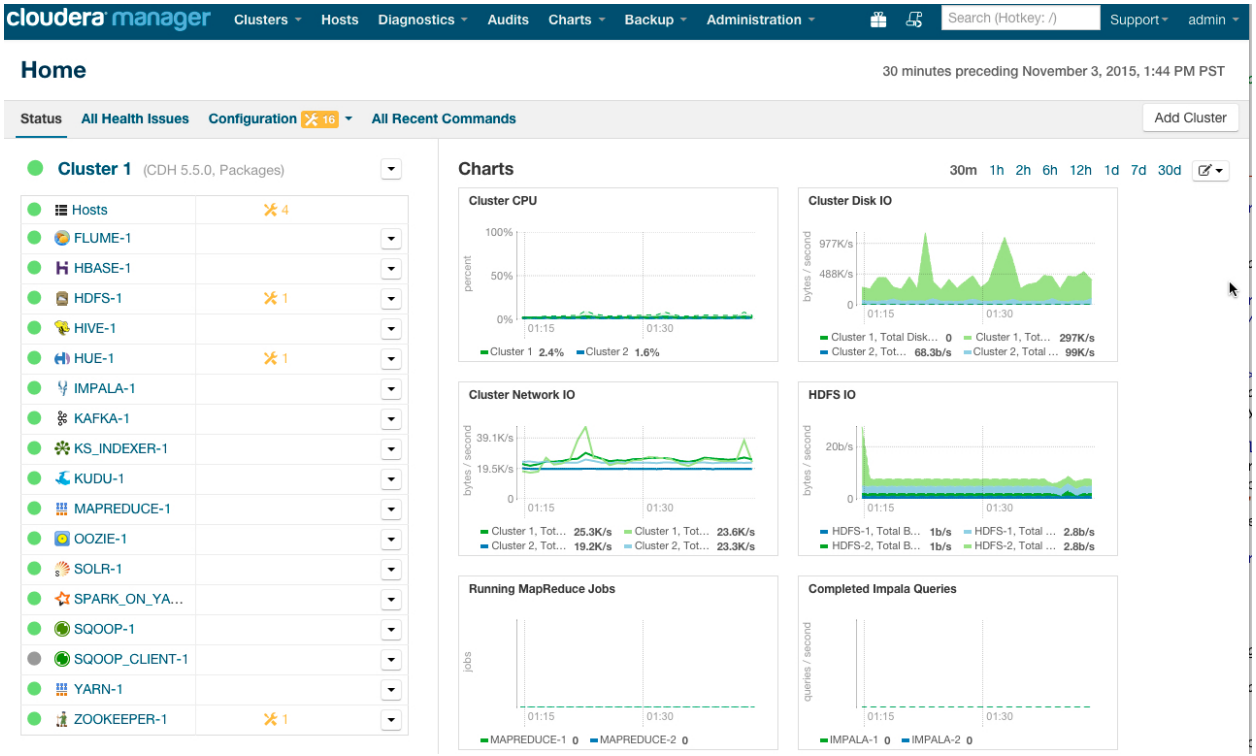



```
import org.apache.spark.sql.functions.{col, rand, round, unix_timestamp, date_format, concat, from_unixtime, substring}
import org.apache.spark.sql.types.{DataTypes}
val r = new scala.util.Random
val rangeDatas = 1262284200 + r.nextInt((1546194600-1262284200))
val join_exp = df.col("NUM_UNID_ATEND_CONGL") === df_left.col("NUM_UNID_ATEND_CONGL_B") && df.col("NUM_CTA") === df_left.col("NUM_CTA_B") && df.col("NUM_DIG_VERIF") === df_left.col("NUM_DIG_VERIF_B")
val join_type = "inner"

// ----------------- valores aleatorios --- inicio ---------------------
val canalOrigUDF = udf((v: Double) => if (v < 30) v.toInt.toString + " (Internet Banking)" else if (v > 30 && v <= 71) v.toInt.toString + " (Mobile)" else v.toInt.toString + " (Canal Especial)")

val codigoPesqLitUDF = udf((v: Double) => if (v < 1000) v.toInt.toString + " (codigo = " + v.toInt.toString + " + dac = 1)"  else if (v > 1000 && v <= 45000) v.toInt.toString + " (codigo = " + v.toInt.toString + " + dac2 = 2)" else v.toInt.toString + " (codigo = " + v.toInt.toString + " + dac3 = 3)" )

val tipoTransUDF = udf((v: Double) => if (v <= 1) "D(Débito), TED_EFET" else if (v > 1 && v <= 2) "C(Crédito), CRED_EFET" else "P(Poupança), POUP_EFET")

val descDETUDF = udf((v: Double) => if (v <= 100) "Estabelecimento A" else if (v > 100 && v <= 500) "Estabelecimento B" else "Estabelecimento C")

val latiUDF = udf((v: Double) => new DecimalFormat("#0.######").format((v * 90) * -1))
val longiUDF = udf((v: Double) => new DecimalFormat("#0.######").format((v * 90) * -1))

val codigoGrupoProdutoUDF = udf((v: Double) => new DecimalFormat("#").format((v * 5)))

val TBIUDF = udf((v: Double) => new DecimalFormat("###0.######").format((v * 1000)))

val codigoMotivoLancUDF = udf((v: Double) => new DecimalFormat("##0").format((v * 100)))

val codigoRoteiContabUDF = udf((v: Double) => new DecimalFormat("##0").format((v * 100)))

val numIdentifContaUDF = udf((v: Double) => new DecimalFormat("#0").format((v * 3)))

val descSaldoUDF = udf((v: Double) => if (v <= 100) "saldo do dia" else if (v > 100 && v <= 500) "saldo a liberar" else "saldo devedor")

val descLitUDF = udf((v: Double) => if (v <= 100) "RSHOP-PANDURATA V-20/06" else if (v > 100 && v <= 500) "RSHOP-PARAIBA V-18/06" else "RSHOP-TERERE V-11/06")

val jsonUDF = udf((valorLanc:Double, codMotLanc:Double ,tbi : Double, codRotCont:Double, codGrupProd: Double, codOrig:Int,codOrig2:Int, lat : Double, longi : Double, surpresa:Double) =>
String.format("""[{"numero_identificacao_lancamento_conta": "","codigo_tipo_operacao": "TEF_CC_CP","valor_lancamento":%s,"codigo_moeda_transacao": "BRL","sigla_sistema_origem":
"X0","codigo_motivo_lancamento": "%s","texto_complemento_lancamento":"TBI %s","conta":{"numero_unico_conta": "67936d95-01a0-4322-a245-0ffdfbc7af3c","numero_titularidade_conta":
"1","codigo_sufixo_conta": "100"},"contabil":{"codigo_roteiro_contabil": "%s"},"metadata": {"codigo_grupo_produto": %s,"codigo_origem": "%s8a6c5a0-4da2-4799-acd0-
443af38292c%s","numero_coordenada_latitude": %s,"numero_coordenada_longitude": %s %s}}]""", new DecimalFormat("#####0.##").format((valorLanc * 90000)), new DecimalFormat("##0").format((codMotLanc
* 100)),new DecimalFormat("###0.######").format((tbi * 400)), new DecimalFormat("##0").format((codRotCont * 900)).toString ,new DecimalFormat("#0").format((codGrupProd * 5)).toString,
(codOrig*50).toString, (codOrig2*70).toString,new DecimalFormat("#0.######").format((lat * 90) * -1).toString, new DecimalFormat("#0.######").format((longi * 90) * -1).toString, if(surpresa <=
```

DIGITAL INNOVATION ONE

On Premises

Cloudera



DIGITAL
INNOVATION
ONE

Parte 2: Streaming

Criando pipelines de dados eficientes - Parte 1

Spark Streaming

DIGITAL INNOVATION ONE

Por que?

- Processamento de eventos em tempo real em sistemas de big data
- De sensores e processamento de dados de rede à detecção de fraudes e monitoramento de sites e muito mais
- Capacidade de consumir, processar e obter insights de fontes de dados de streaming

Os objetivos para Spark Streaming incluíam:

- Latência baixa (segunda escala)
- Processamento de evento único (e apenas uma vez)
- Escalabilidade linear
- Integração com Spark Core API

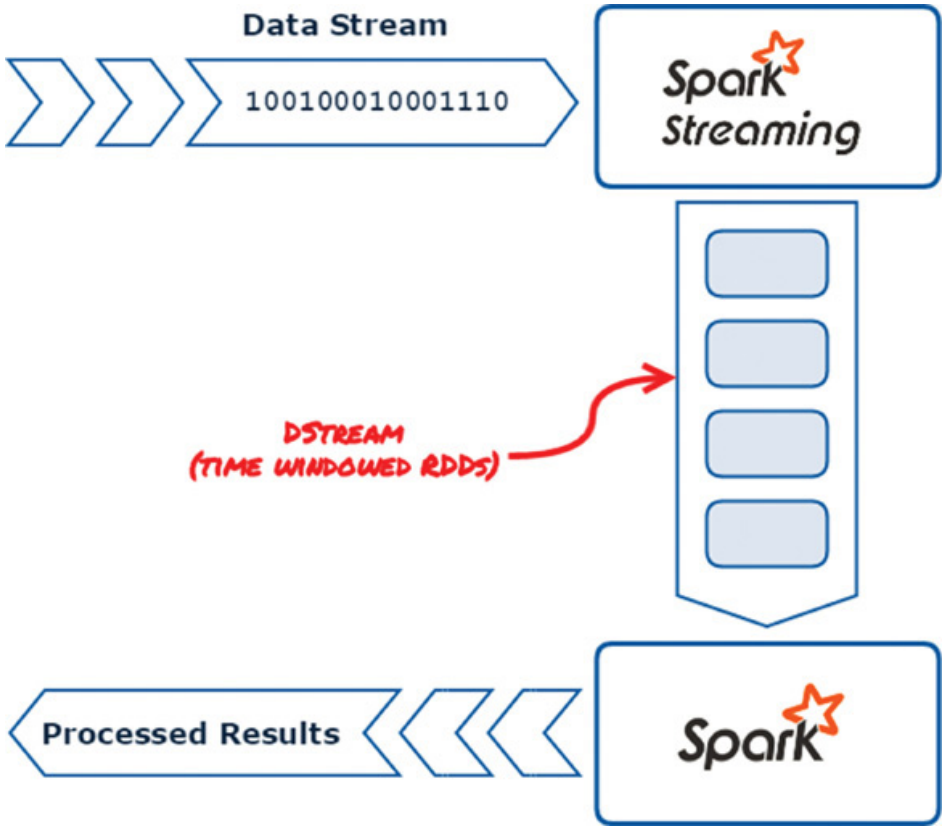
DIGITAL INNOVATION ONE

Por que?

- Necessidade do processamento de eventos/fluxos como um componente-chave
- Processamento de eventos integrado com sua estrutura de lote baseada em RDD
- A abordagem Spark Streaming
- Spark Streaming apresenta o conceito de "discretized streams" (ou DStreams)
- DStreams são essencialmente lotes de dados armazenados em vários RDDs, cada lote representando uma janela de tempo (normalmente em segundos)
- Os RDDs resultantes podem ser processados usando a API Spark RDD principal e todas as transformações disponíveis

DIGITAL INNOVATION ONE

Macro arq.



DIGITAL INNOVATION ONE

Revisão

Assim como acontece com os pontos de entrada do programa SparkContext, SQLContext e HiveContext que discuti antes, os aplicativos Spark Streaming também têm um ponto de entrada chamado StreamingContext

- Aplicação Spark = Driver + grupo de Executors
- Driver executa o processo principal e cria um SparkContext que serve para coordenar a execução do seu job.
- Os executors são processos em execução nos work nodes responsável por executar tasks que o Driver atribuiu a ele.
- O cluster manager (Yarn, Mesos) é responsável pela alocação de recursos para seu aplicativo Spark.

O SparkContext é usado pelo processo do Spark Driver do seu aplicativo Spark para estabelecer uma comunicação com o cluster e o cluster manager (Yarn) para então coordenar e executar jobs.

SparkContext também permite o acesso a outros dois contextos, ou seja, SQLContext e HiveContext.

Para criar um SparkContext, primeiro você precisa criar uma configuração, chamada de SparkConf.

DIGITAL INNOVATION ONE

Revisão

- SQLContext é o ponto de entrada para SparkSQL, que é um módulo Spark para processamento de dados estruturados.
- Depois que o SQLContext é inicializado, o usuário pode usá-lo para realizar várias operações "semelhantes a sql" em conjuntos de dados e dataframes.
- Para criar um SQLContext, primeiro você precisa instanciar um SparkContext.
- Se o seu aplicativo Spark precisa se comunicar com o Hive e você está usando o Spark <2.0, provavelmente precisará de um HiveContext.

DIGITAL INNOVATION ONE

Revisão

```
# PySpark

from pyspark import SparkContext, HiveContext

conf = SparkConf().setAppName ('app').setMaster (master)
sc = SparkContext (conf)
hive_context = HiveContext (sc)
hive_context.sql ("select * from tableName limit 0")
```

DIGITAL INNOVATION ONE

Revisão

```
// Scala – Acessando o Context

import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.sql.hive.HiveContext

val sparkConf = new SparkConf().setAppName("app").setMaster("yarn")
val sc = new SparkContext(sparkConf)
val hiveContext = new HiveContext(sc)
hiveContext.sql("select * from tableName limit 0")
```

DIGITAL INNOVATION ONE

Revisão

O Spark 2.0 introduziu o SparkSession que substituiu essencialmente SQLContext e HiveContext e concede acesso imediato ao SparkContext.

Criar um Spark Session com suporte a Hive:

```
# PySpark

from pyspark.sql import SparkSession
spark_session = SparkSession.builder.enableHiveSupport().getOrCreate()
# Duas maneiras de acessar o contexto do spark a partir da sessão do spark
spark_context = spark_session._sc
spark_context = spark_session.sparkContext
```

DIGITAL INNOVATION ONE

StreamContext

- O StreamingContext representa uma conexão a uma plataforma ou cluster Spark usando um SparkContext existente
- O StreamingContext é usado para criar os datasources de DStreams controlar a computação de streaming e as transformações de DStream.
- O StreamingContext também especifica o argumento batchDuration, que é um intervalo de tempo em segundos pelo qual os dados de streaming serão divididos em lotes.
- Depois de instanciar um StreamingContext, você criaria uma conexão com um fluxo de dados e definiria uma série de transformações a serem realizadas.
- O método start() ou ssc.start() é usado para acionar a os dados de entrada depois que um StreamingContext é estabelecido.
- O StreamingContext pode ser interrompido programaticamente usando ssc.stop()ou ssc.awaitTermination().

DIGITAL INNOVATION ONE

StreamContext

- Discretized streams (DStreams) são o objeto de programação básico na API Spark Streaming
- Streams representam uma sequência contínua de RDDs que são criados a partir de um fluxo contínuo de dados
- DStreams podem ser criados a partir de fontes de dados de streaming, como soquetes TCP, sistemas de mensagens, APIs de streaming (como a API de streaming do Twitter) e muito mais.
- DStreams (como uma abstracção RDD) também pode ser produzido a partir de transformações realizadas na DStreams existentes (tais como map, flatMap e outras operações).

DStreams oferece suporte a dois tipos de operações:

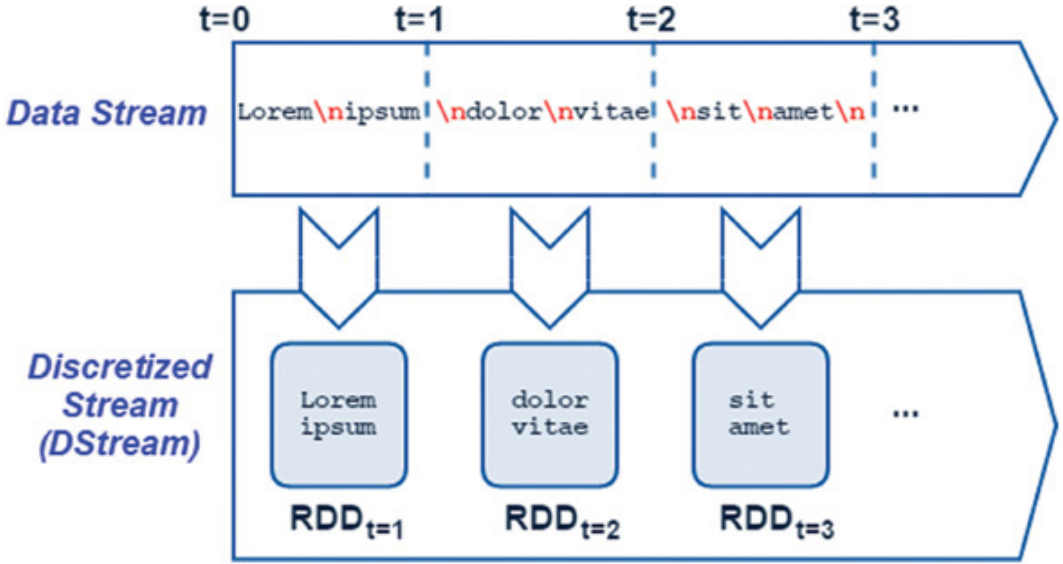
1. Imagem Transformações

2. Imagem Operações de saída

StreamContext

DStreams são Lazy Evaluation assim como Spark RDD

Representação de um discretized stream, com cada intervalo t representando uma janela de tempo especificada pelo batchDurationargumento na instanciação de StreamingContext:



DIGITAL INNOVATION ONE

StreamContext

- DStreams Source são definidos em um StreamingContext para um fluxo de dados de entrada especificado, da mesma forma que os RDDs são criados para uma fonte de dados de entrada em um SparkContext
- Muitas fontes de entrada de streaming comuns estão incluídas na API de streaming, como fontes para ler dados de um soquete TCP ou para ler dados enquanto eles estão sendo gravados no HDFS
- As fontes básicas de dados de entrada para a criação de DStreams são descritas aqui:
  socketTextStream()
- StreamingContext.socketTextStream (hostname, port, storageLevel = StorageLevel (True, True, False, False, 2))
- O método socketTextStream é usado para criar um DStream a partir de uma fonte TCP de entrada definida pelos argumentos hostnamee e port.
- Os dados recebidos são interpretados usando a enconding UTF8, com terminação de nova linha usada para definir novos registros.
- O storageLevelargumento que define o storage level padrão MEMORY_AND_DISK_SER_2

StreamContext

```
from pyspark.streaming import StreamingContext

ssc = StreamingContext(sc, 1)
lines = ssc.socketTextStream('localhost', 9999)
counts = lines.flatMap(lambda line: line.split(" ")).map(lambda word: (word, 1)).reduceByKey(lambda a, b: a+b)
counts.pprint()
ssc.start()
ssc.awaitTermination()
```

DIGITAL INNOVATION ONE

Parte 3: Casos de Uso

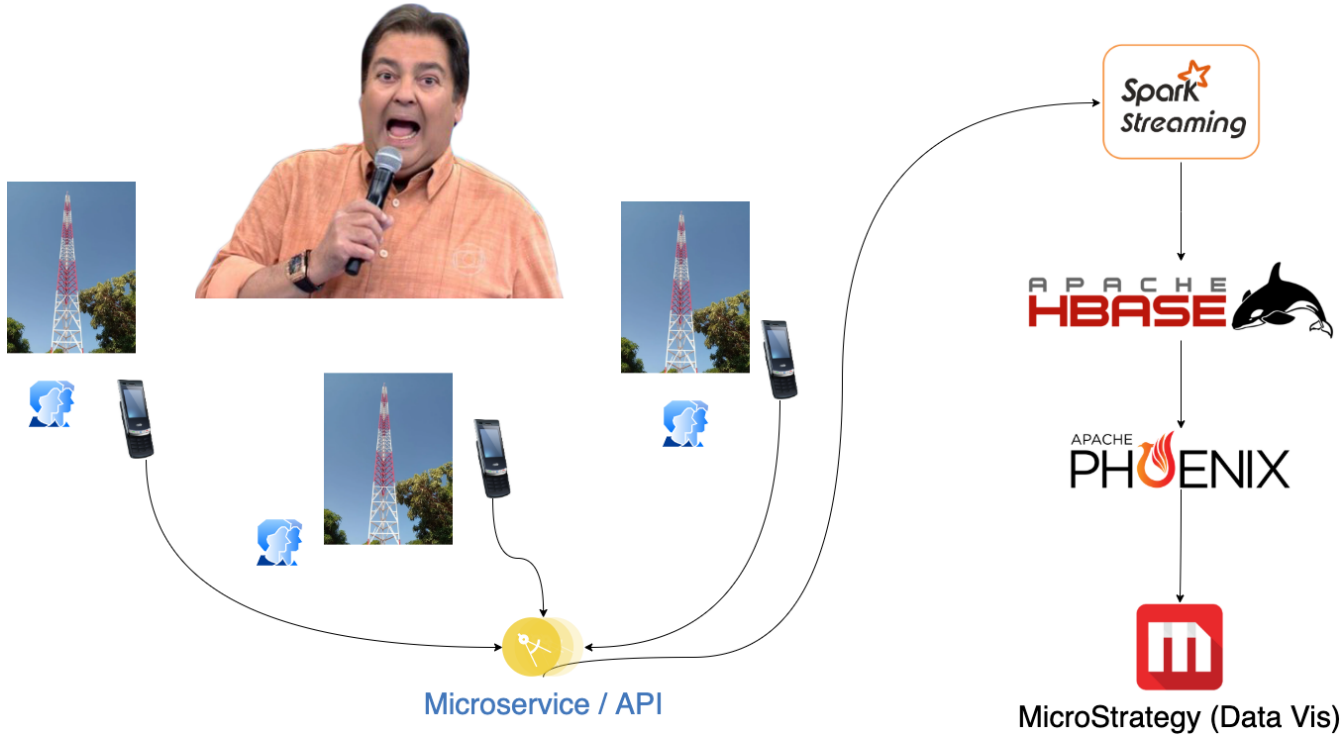
Criando pipelines de dados eficientes - Parte 1

Spark Streaming

DIGITAL INNOVATION ONE

Telecom

"Torpedão do Faustão"

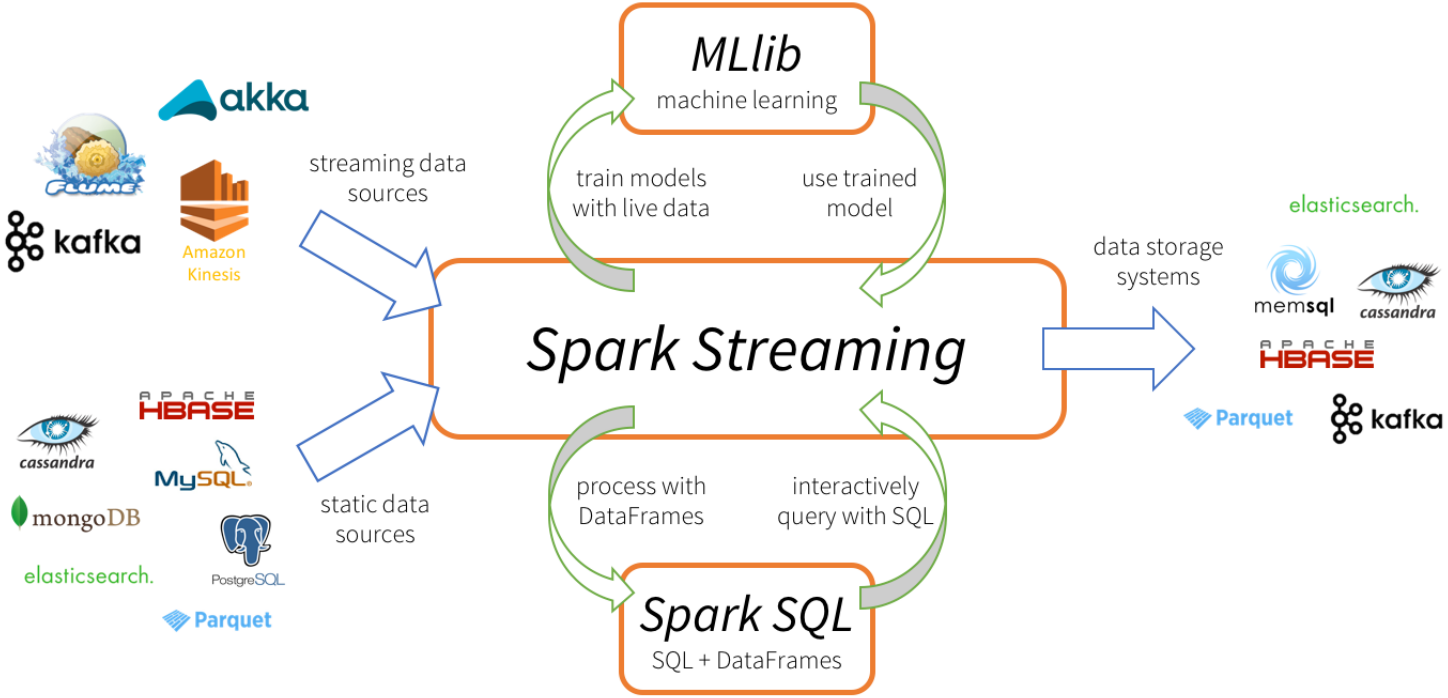


DIGITAL INNOVATION ONE

Compatibilidade

DIGITAL INNOVATION ONE

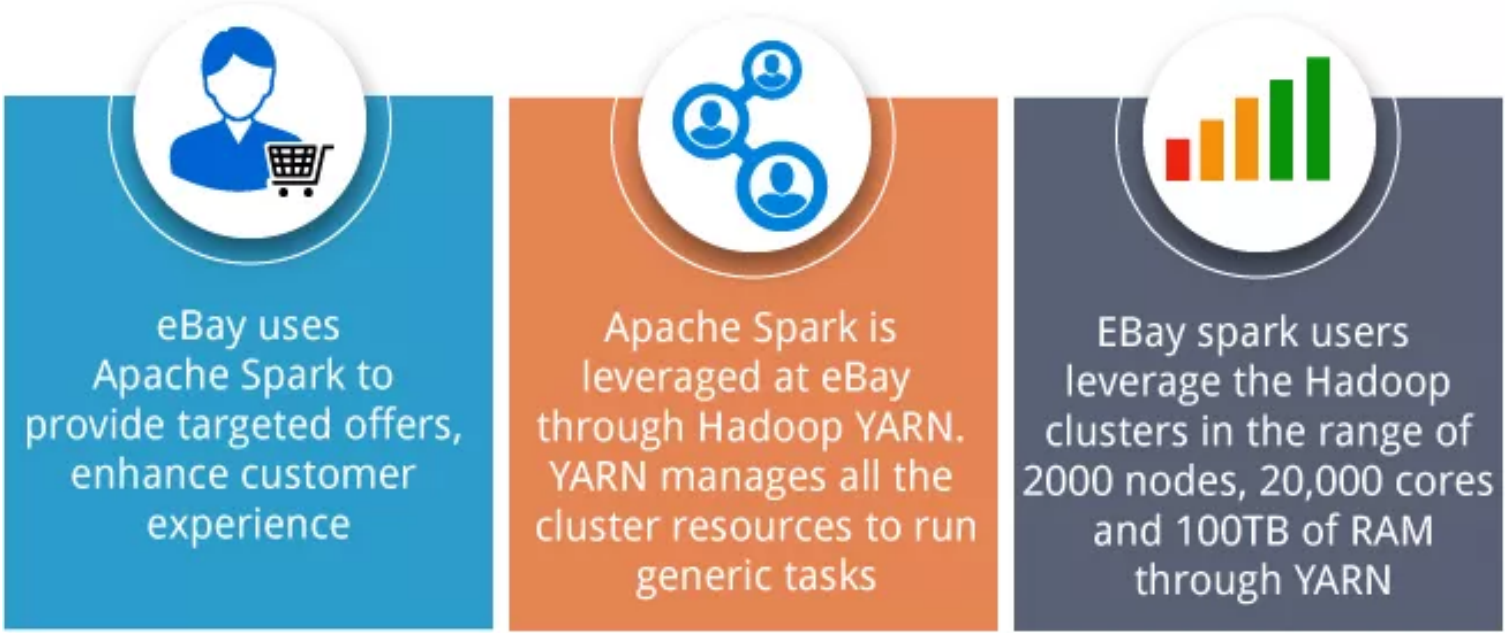
Compatibilidade



DIGITAL INNOVATION ONE

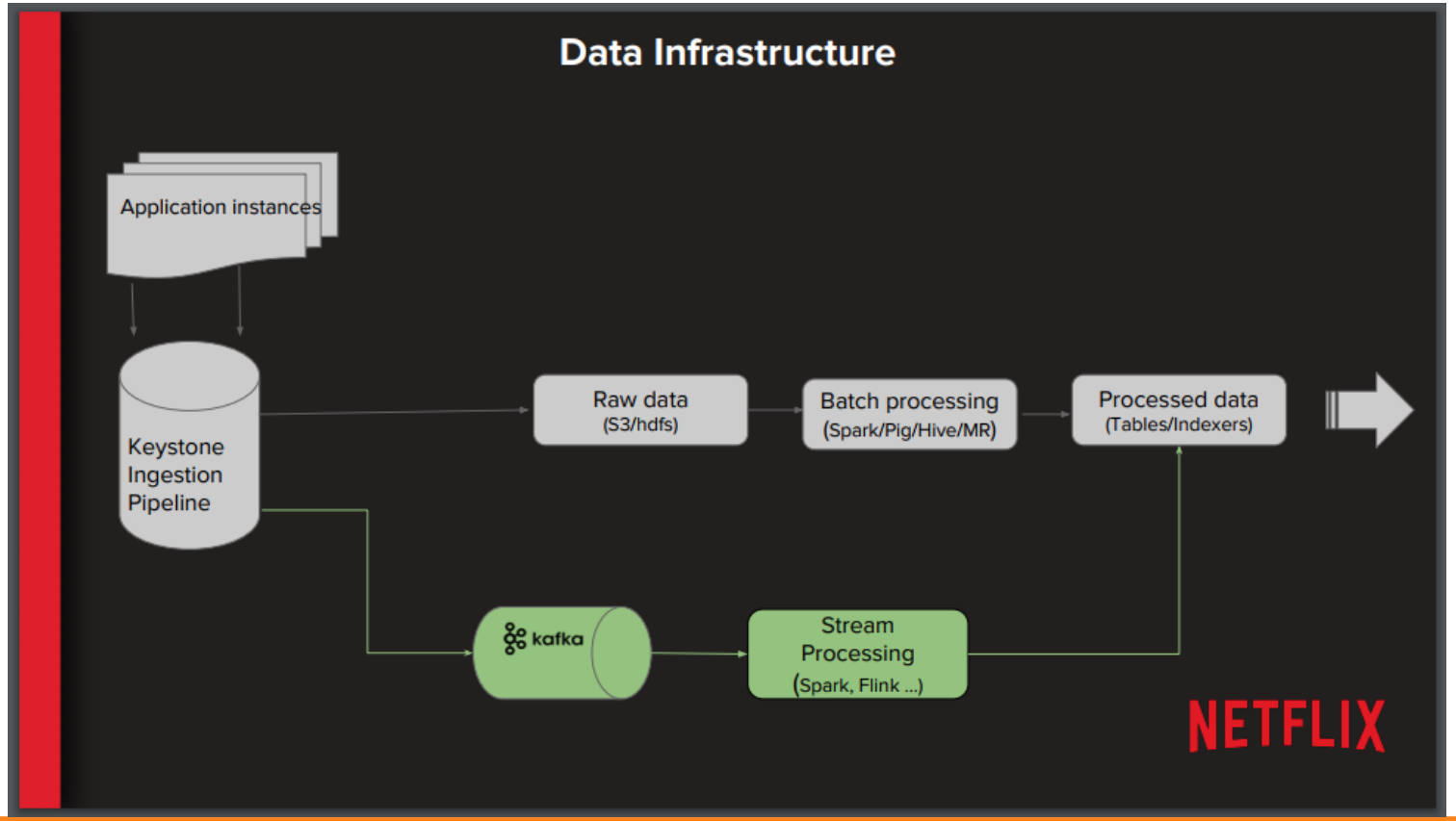
eBay

Apache Spark at ebay



DIGITAL INNOVATION ONE

eBay



DIGITAL INNOVATION ONE

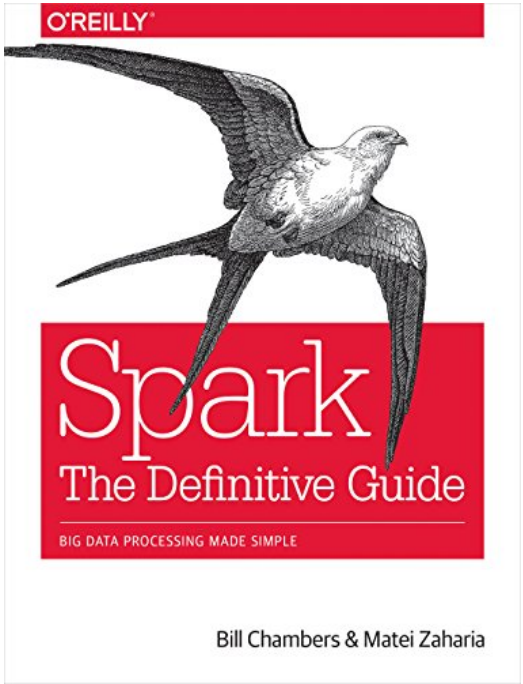
Parte 4: Encerramento

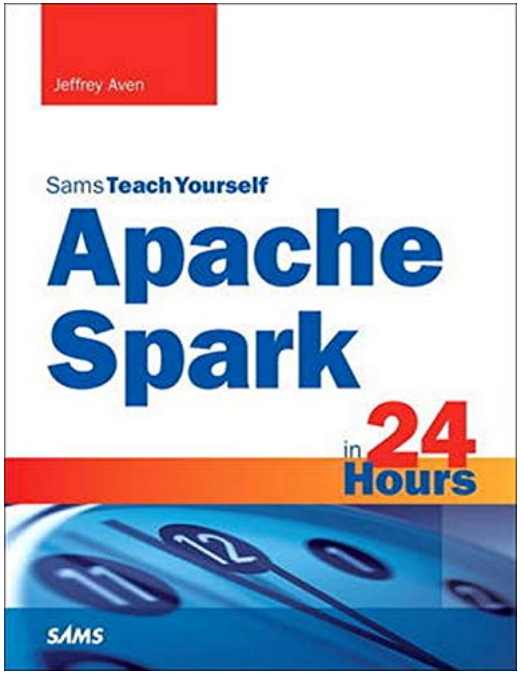
Criando pipelines de dados eficientes - Parte 1

Spark Streaming

DIGITAL INNOVATION ONE

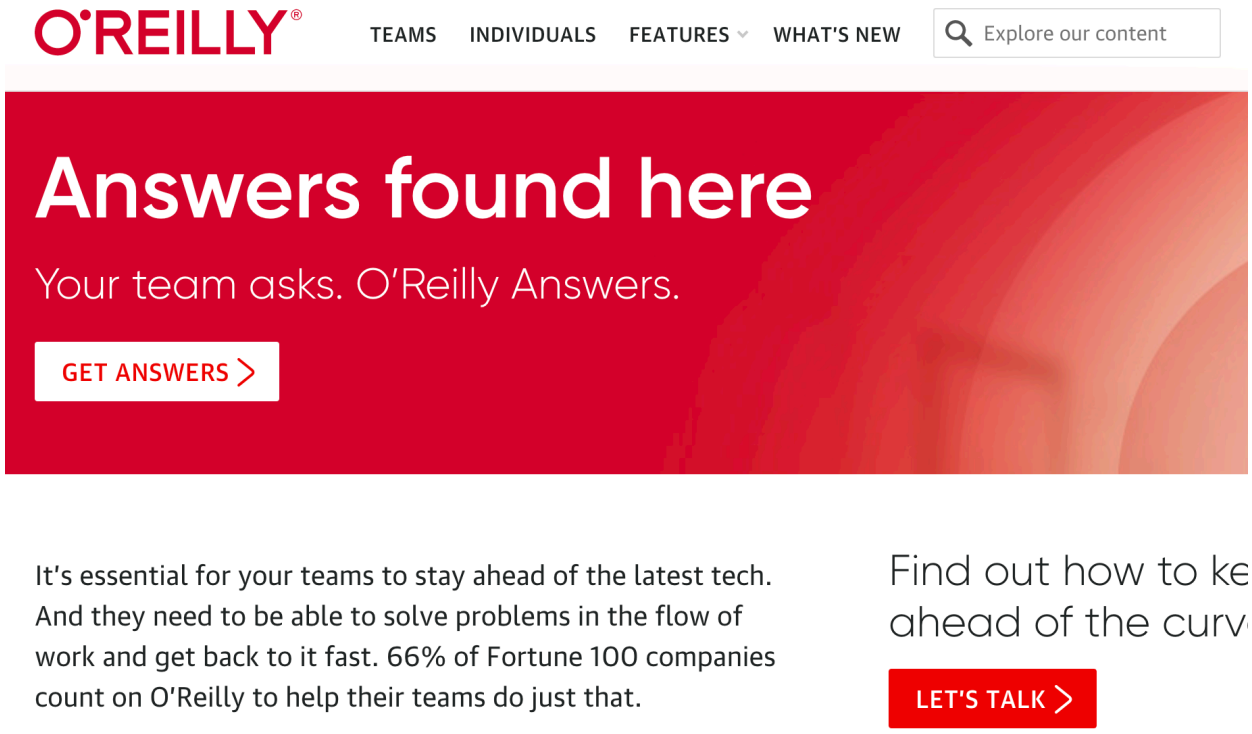
Referências





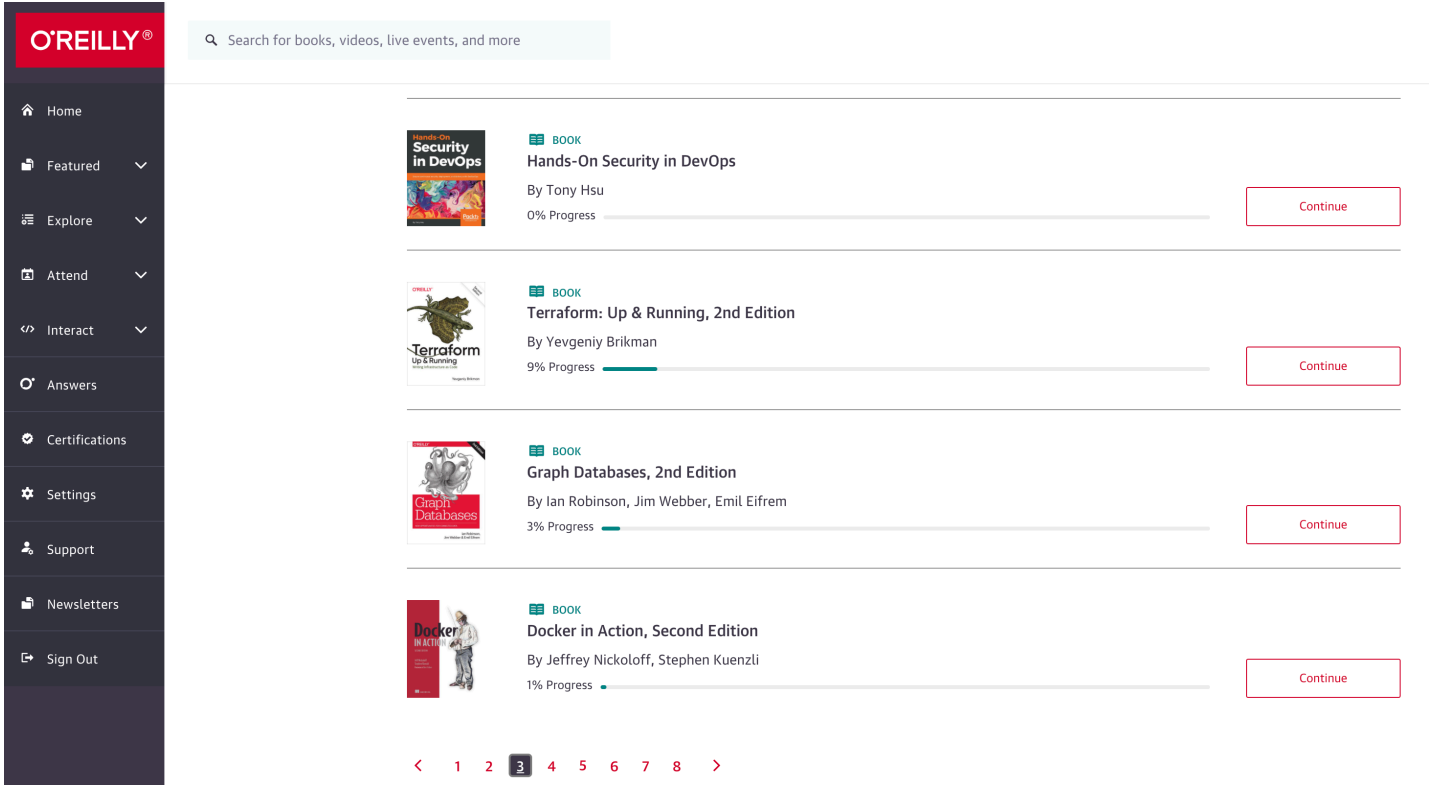
Referências

Safari Books: https://www.safaribooksonline.com/



Referências

Safari Books: https://www.safaribooksonline.com/



DIGITAL INNOVATION ONE

Referências

https://academy.databricks.com/exam/INT-ADAS-v2-CT

Preparation

The following Databricks courses should help you prepare for this exam:

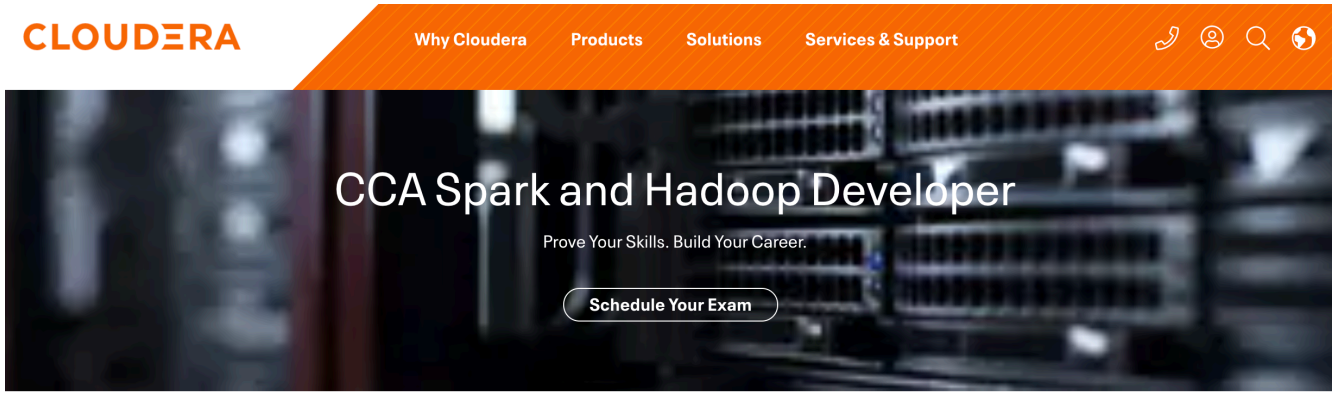
- DB 105 - Apache Spark Programming
- Quick Reference: Spark Architecture
- Future self-paced course on the Spark DataFrames API

In addition, Sections I, II, and IV of *Spark: The Definitive Guide* should also be helpful in preparation.

DIGITAL INNOVATION ONE

Certificações

Cloudera: https://www.cloudera.com/about/training/certification/cca-spark.html



CCA Spark and Hadoop Developer

CCA Spark and Hadoop Developer Exam (CCA175)

- **Number of Questions:** 8–12 performance-based (hands-on) tasks on Cloudera Enterprise cluster. See below for full cluster configuration
- **Time Limit:** 120 minutes
- **Passing Score:** 70%
- **Language:** English
- **Price:** USD $295

Purchase

Watch a free **OnDemand course** to help prepare for your certification

Have questions? Read our **Certification FAQ**

Verify a certification

Contact us at **certification@cloudera.com**

DIGITAL INNOVATION ONE

Certificações

Cloudera: https://www.cloudera.com/about/training/certification/ccp-data-engineer.html

CCP Data Engineer

CCP Data Engineer Exam (DE575)

- **Number of Questions:** 5–10 performance-based (hands-on) tasks on pre-configured Cloudera Enterprise cluster.
- **Time Limit:** 240 minutes
- **Passing Score:** 70%
- **Language:** English
- **Price:** USD $400

Purchase

Have questions? Read our **Certification FAQ**

Verify a Certification

Contact us at **certification@cloudera.com**

DIGITAL INNOVATION ONE

Certificações

Databricks: https://academy.databricks.com/category/certifications

Assessments

Databricks Certified Associate Developer for Apache Spark 3.0 – *Assessment*

$ 200.00 USD

The Databricks Certified Associate Developer for Apache Spark 3.0 certification exam assesses an understanding of the basics of the Spark architecture and the ability to apply the Spark DataFrame API to complete individual data manipulation tasks.

☆ VIEW

Databricks Certified Associate Developer for Apache Spark 2.4 – *Assessment*

$ 200.00 USD

The Databricks Certified Associate Developer for Apache Spark 2.4 certification exam assesses an understanding of the basics of the Spark architecture and the ability to apply the Spark DataFrame API to complete individual data manipulation tasks.

☆ VIEW

Databricks Certified Associate ML Practitioner for Apache Spark 2.4 – *Assessment*

$ 200.00 USD

The Databricks Certified Associate ML Practitioner for Apache Spark 2.4 certification exam assesses the understanding of and ability to apply machine techniques using the Spark ML library.

☆ VIEW

Databricks Certified Professional Data Scientist – *Assessment*

$ 200.00 USD

The Databricks Certified Professional Data Scientist certification exam assesses the understanding of the basics of machine learning, the steps in the machine learning lifecycle, the understanding of basic machine learning algorithms and techniques, and the understanding of the basics of machine learning model management.

☆ VIEW

Databricks Certified Professional Data Engineer – *Assessment*

$ 200.00 USD

Coming soon (early 2021).

☆ VIEW

DIGITAL
INNOVATION
ONE

Dúvidas?

Criando pipelines de dados eficientes - Parte 1
Spark Streaming

DIGITAL INNOVATION ONE

Aceleração Global Dev #4 everis

Criando pipelines de dados eficientes - Parte 2
Spark SQL - PySpark

Marco Antonio Pereira
Expert Data Architect

DIGITAL INNOVATION ONE

Objetivos da Aula

1. Apresentar Spark SQL

2. Usabilidade

3. Casos de Uso

DIGITAL INNOVATION ONE

Requisitos Básicos

- ✓ Máquina Virtual com Apache Spark 2.4
- ✓ Conta na Community Databricks
- ✓ Conhecimentos básicos em Python ou Scala
- ✓ Persistência

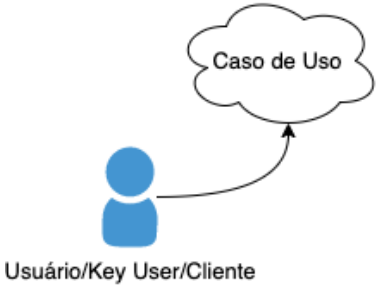
DIGITAL INNOVATION ONE

Parte 1: Falando sobre Dados

Criando pipelines de dados eficientes - Parte 2
Spark SQL

DIGITAL INNOVATION ONE

Dados



Quero criar um painel de movimentação de vendas por produto.

Neste painel, quero poder ver:
- O produto
- Preço do produto
- Quantidade vendida por produto
- Impostos/Taxas

Para comparar com um período anterior em até 10 anos em uma visão diária, mensal e anual

O balanço do dia atual é fechado às 7h do dia posterior.

DIGITAL INNOVATION ONE

Dados

Que tipo de informação podemos extrair?

Lógico:

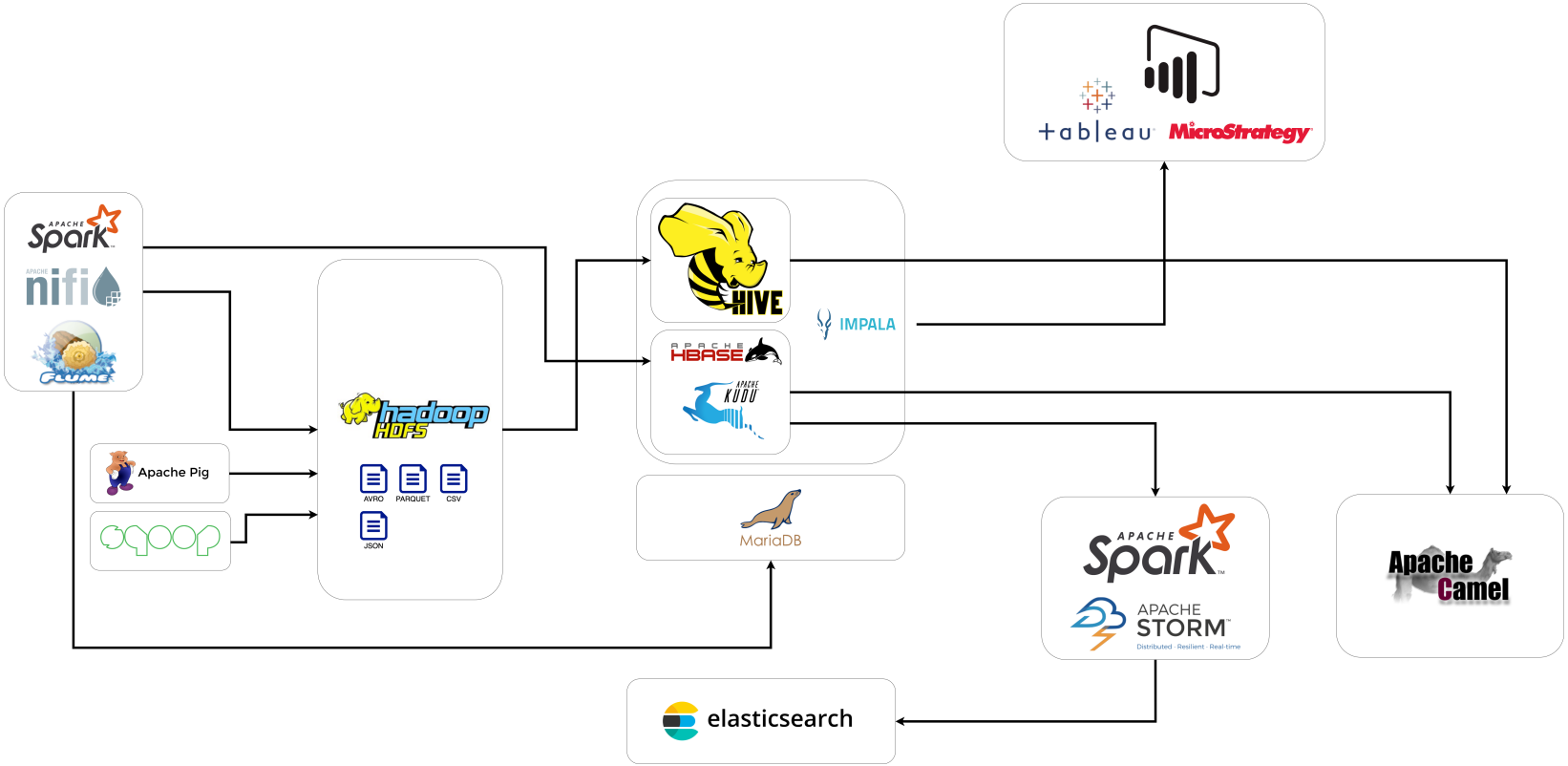
1. Procurar/mapear informações de produtos em alguma base de dados de produto
2. Criar histórico/base vegetativa de até 10 anos
3. Possível ingestão de dados Batch (dia -1 = dados de ontém até -10 anos)

Físico:

1. Qual a melhor estratégia de particionamento?
2. Qual banco de dados devo utilizar?
3. Qual tipo de arquivo de arquivo devo utilizar?
4. Qual tipo de compressão devo utilizar?

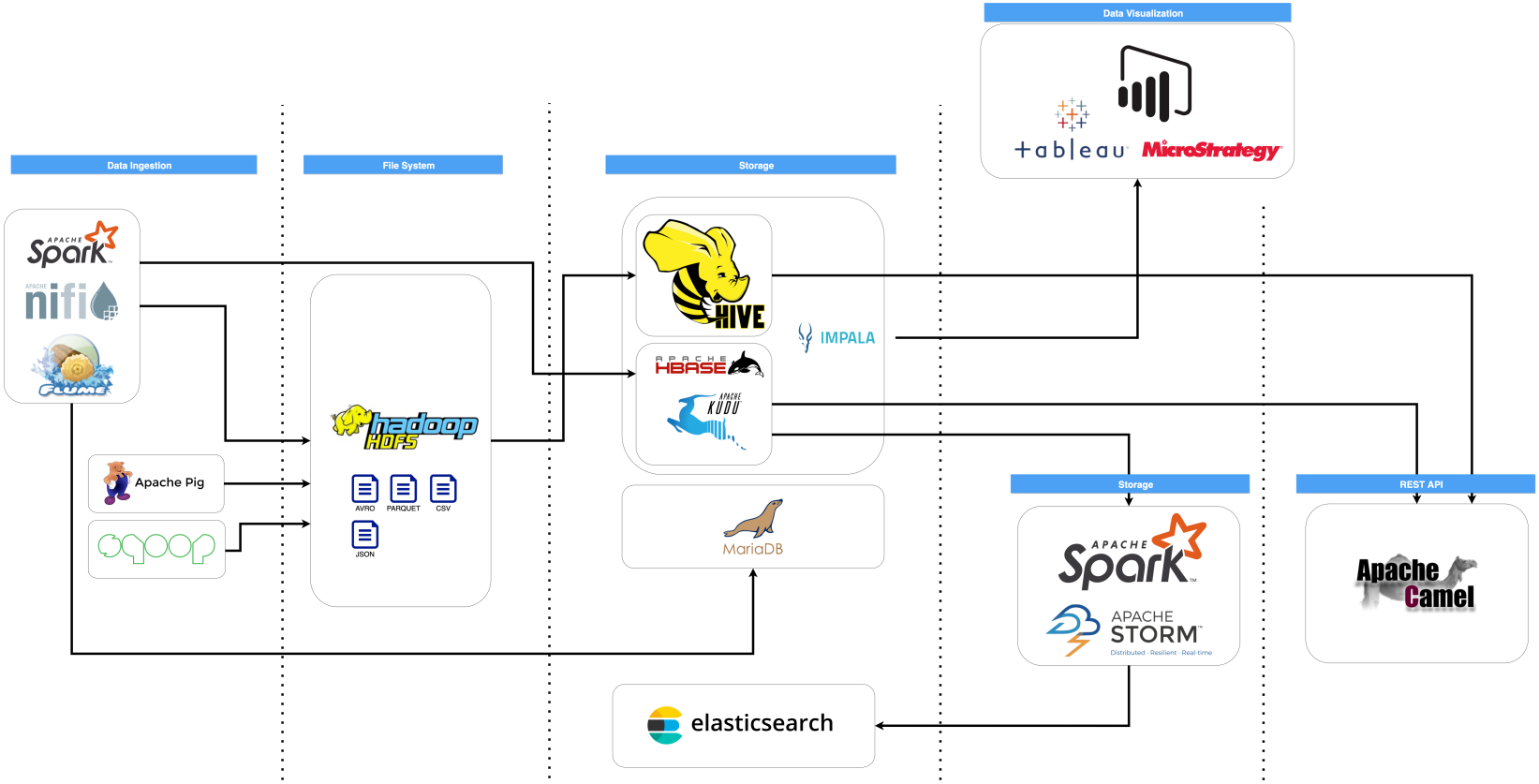
DIGITAL INNOVATION ONE

Arquitetura



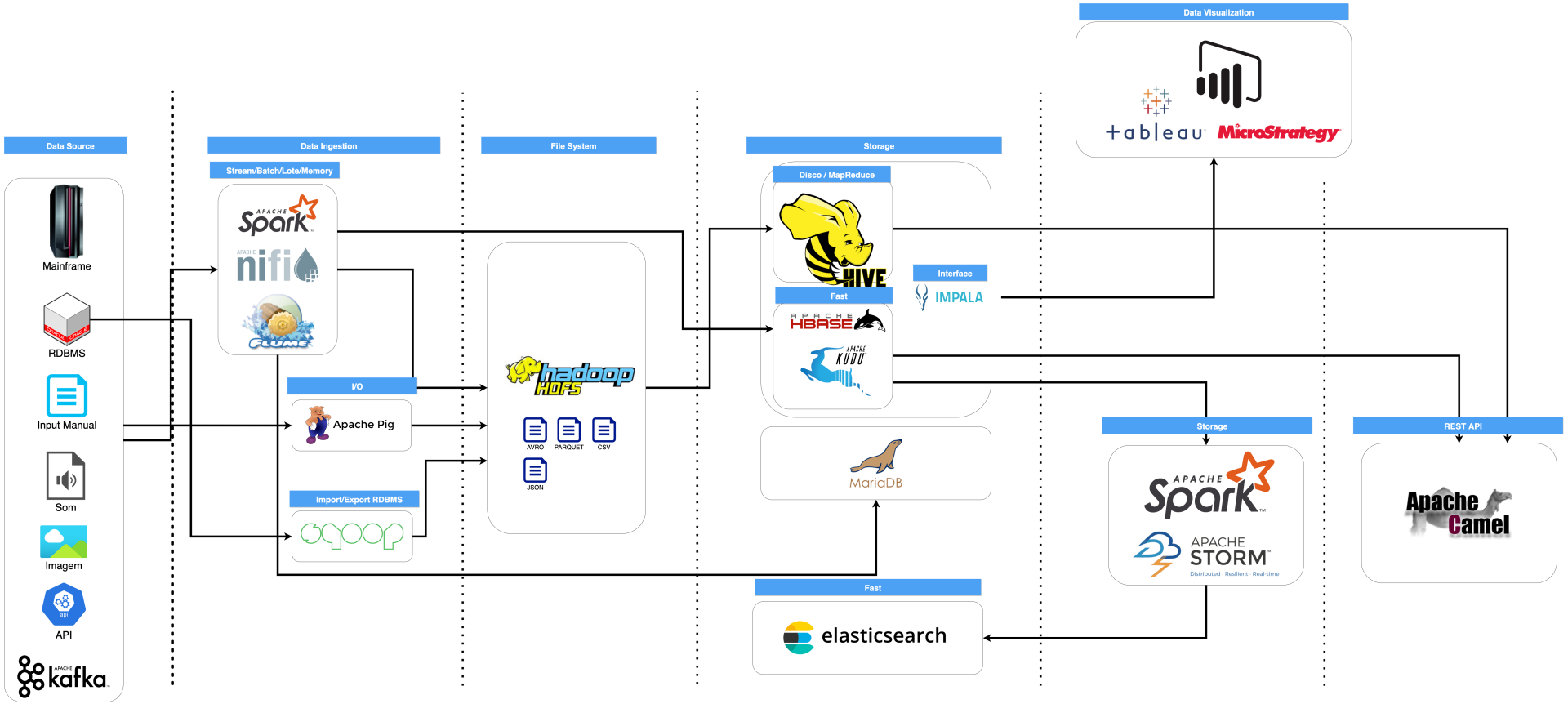
DIGITAL INNOVATION ONE

Arquitetura



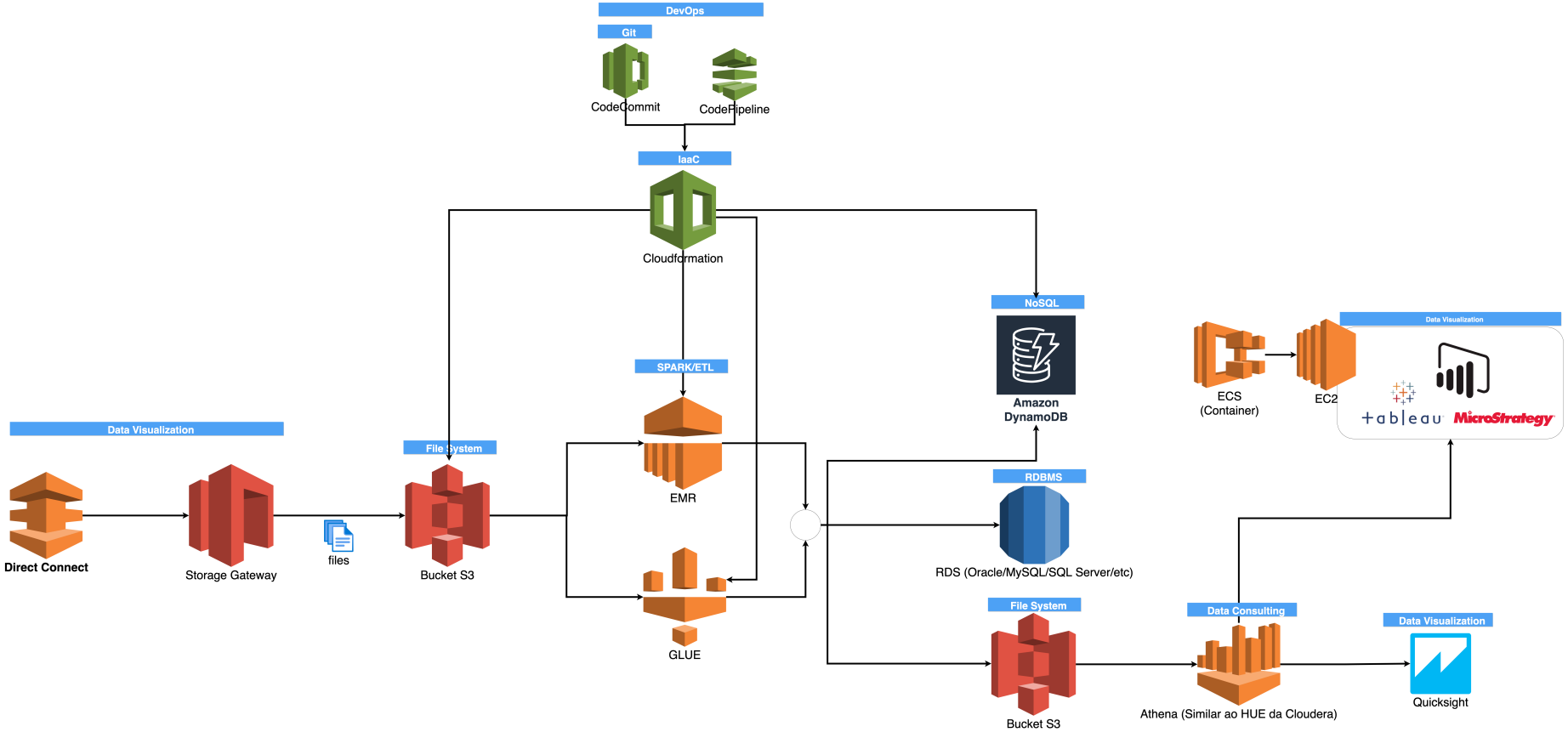
DIGITAL INNOVATION ONE

On Premises



DIGITAL INNOVATION ONE

Cloud (AWS)



DIGITAL INNOVATION ONE

Cloud (AWS)

Guia de custo e configuração de máquinas: https://aws.amazon.com/pt/ec2/pricing/on-demand/

Spark:

Node	Máquina	CPU	Memória	Armazenamento	Custo/Hora
Namenode	m5d.4xlarge	16	64GB	2 x 300 SSD NVMe	1,44 USD
Datanode 1	m5d.4xlarge	16	64GB	2 x 300 SSD NVMe	1,44 USD
Datanode 2	m5d.4xlarge	16	64GB	2 x 300 SSD NVMe	1,44 USD
Datanode 3	m5d.4xlarge	16	64GB	2 x 300 SSD NVMe	1,44 USD
Datanode 4	m5d.4xlarge	16	64GB	2 x 300 SSD NVMe	1,44 USD

DIGITAL INNOVATION ONE

Cloud (AWS)

RDS

Tipo	Máquina	Custo/Hora
RDS/Oracle	db.r5.2xlarge	3,1152 USD

DynamoDB: https://aws.amazon.com/pt/dynamodb/pricing/on-demand/

Tipo	Custo
gravação	1,875 USD por milhão de unidades de solicitação de gravação
leitura	0,375 USD por milhão de unidades de solicitação de gravação

CloudWatch (Logs): https://aws.amazon.com/pt/cloudwatch/pricing/

Tipo	Primeiras Primeiras 10.000 métricas
CloudWatch	0,30 USD / mês

DIGITAL INNOVATION ONE

Camadas

RAW ZONE

Todo dado recebido pelos sistemas origens classificamos como Raw Zone. Realizamos ingestões nesta camada com todos os tipos STRING. O tipo STRING é o tipo mais genérico de todos. Ao realizarmos ingestões com o tipo String, garantimos que o dado AS-IS é armazenado. Caso arquivos provenientes do sistema origem cheguem com algum tipo de anormalidade tal como caracteres especiais, formatos inválidos, entre outros, o tipo STRING é flexível o suficiente para armazenar o dado como ele vem da origem.

TRUSTED ZONE

Na camada Trusted realizamos ingestões mais direcionada a regras de negócio. Após o cruzamento de tabelas do tipo Raw, realizamos transformações de CAST, por exemplo, realizamos JOIN's e aplicamos regras de negócio. A ingestão final deve contemplar os data-types corretos, ou seja, um campo valorado decimal em Raw é STRING porém aqui ele é Decimal.

REFINED ZONE

Na ultima camada, Refined, criamos visões para relatórios. Criamos tabelas do tipo Refined selecionando apenas campos específicos para as visões de relatório.Então se o resultado de uma tabela agregada e tipada, Trusted possuir várias informações sobre um determinado tema e precisamos mostrar no relatório um agrupamento menor de campos, criarmos Refined a partir de Trusted. Caso todos os campos precisem ser mostrados no Dashboard, entendemos que a "Trusted" é a "Refined".

Quando você está trabalhando em um ambiente de Big Data, existem vários formatos de dados. Os dados podem ser formados em um formato legível como arquivo JSON ou CSV, mas isso não significa que essa é a melhor maneira de realmente armazenar os dados.

Existem três formatos de arquivo otimizados para uso em clusters Hadoop:

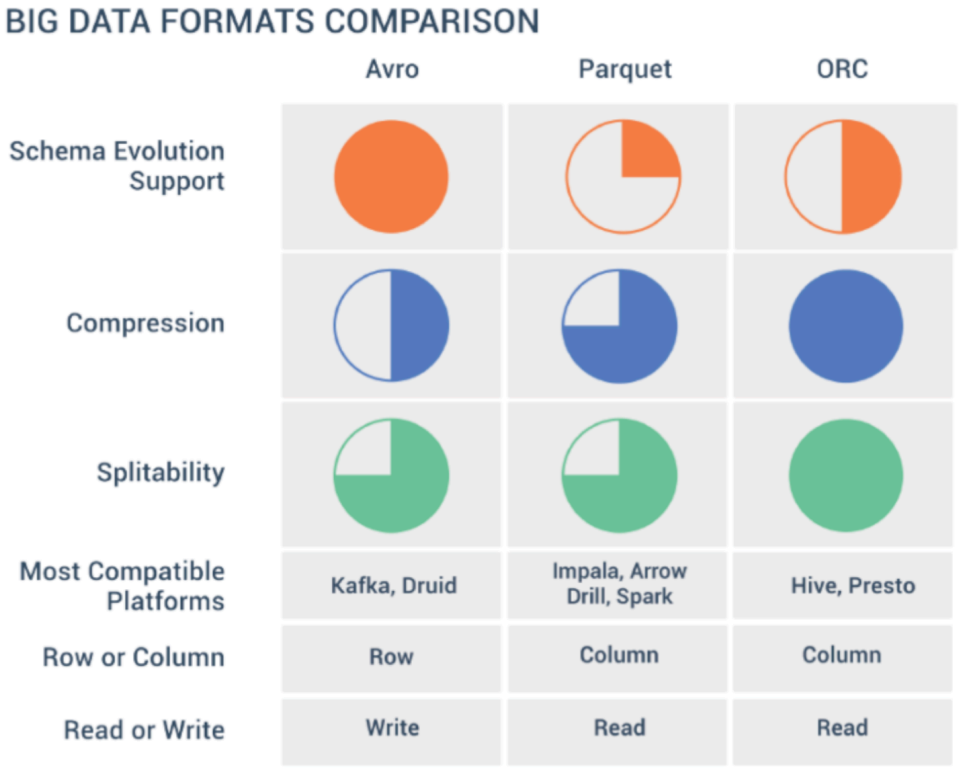
Optimized Row Columnar (ORC)

Avro

Parquet

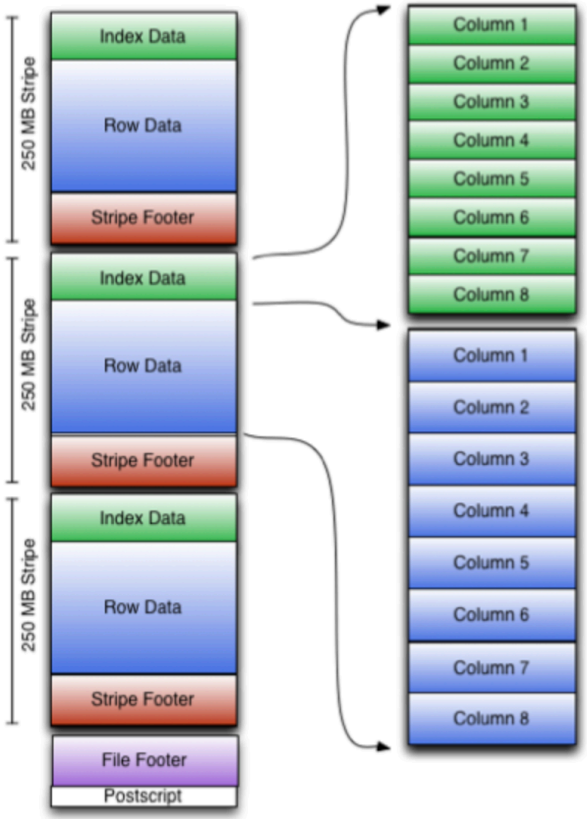
DIGITAL INNOVATION ONE

Tipos de armaz.



DIGITAL INNOVATION ONE

Parquet



DIGITAL INNOVATION ONE

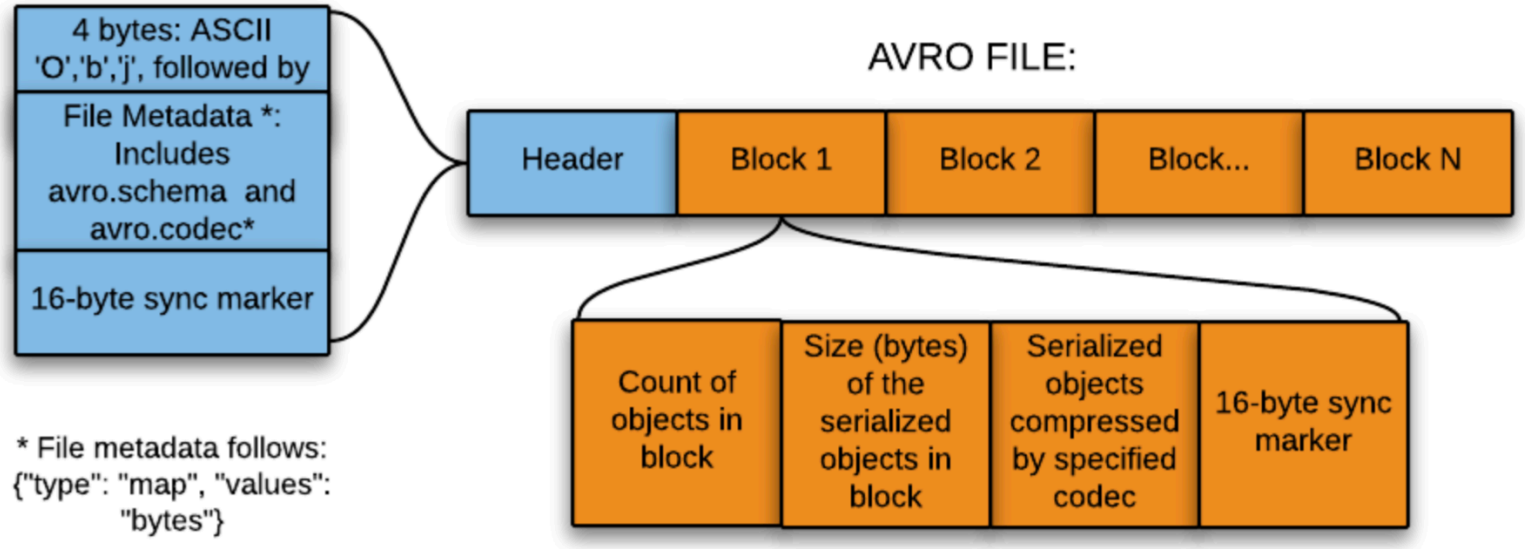
Parquet

- Orientado por coluna (armazenar dados em colunas): os armazenamentos de dados orientados por coluna são otimizados para cargas de trabalho analíticas pesadas em leitura
- Altas taxas de compressão (até 75% com compressão Snappy)
- Apenas as colunas necessárias seriam buscadas / lidas (reduzindo a E / S do disco)
- Pode ser lido e escrito usando Avro API e Avro Schema

Referência: http://parquet.apache.org/documentation/latest/

DIGITAL INNOVATION ONE

Avro



DIGITAL INNOVATION ONE

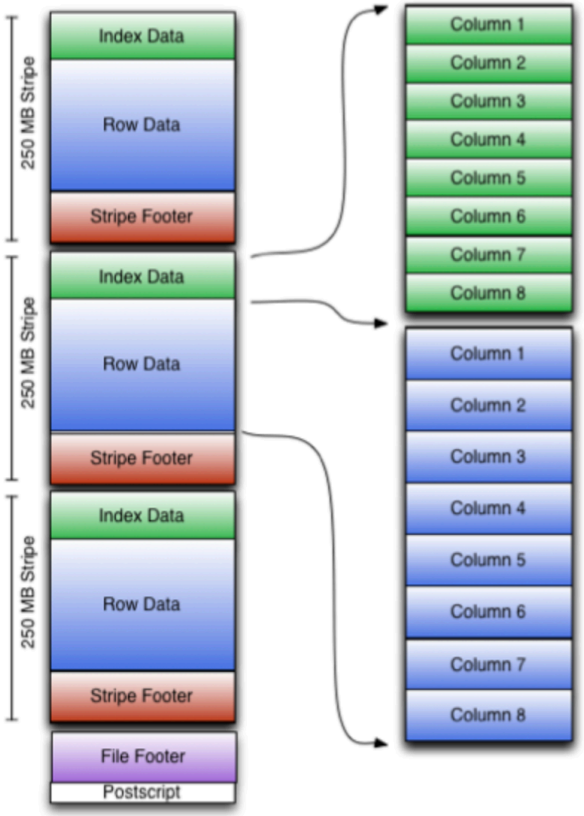
Avro

- Com base em linha (armazenar dados em linhas): bancos de dados baseados em linha são melhores para cargas de trabalho transacionais pesadas de gravação
- Serialização de suporte
- Formato binário rápido
- Suporta compressão de bloco e divisível
- Evolução do esquema de suporte (o uso de JSON para descrever os dados, enquanto usa o formato binário para otimizar o tamanho do armazenamento)
- Armazena o esquema no cabeçalho do arquivo para que os dados sejam autodescritivos

Referência: https://avro.apache.org/docs/1.10.1/

DIGITAL INNOVATION ONE

ORC



DIGITAL INNOVATION ONE

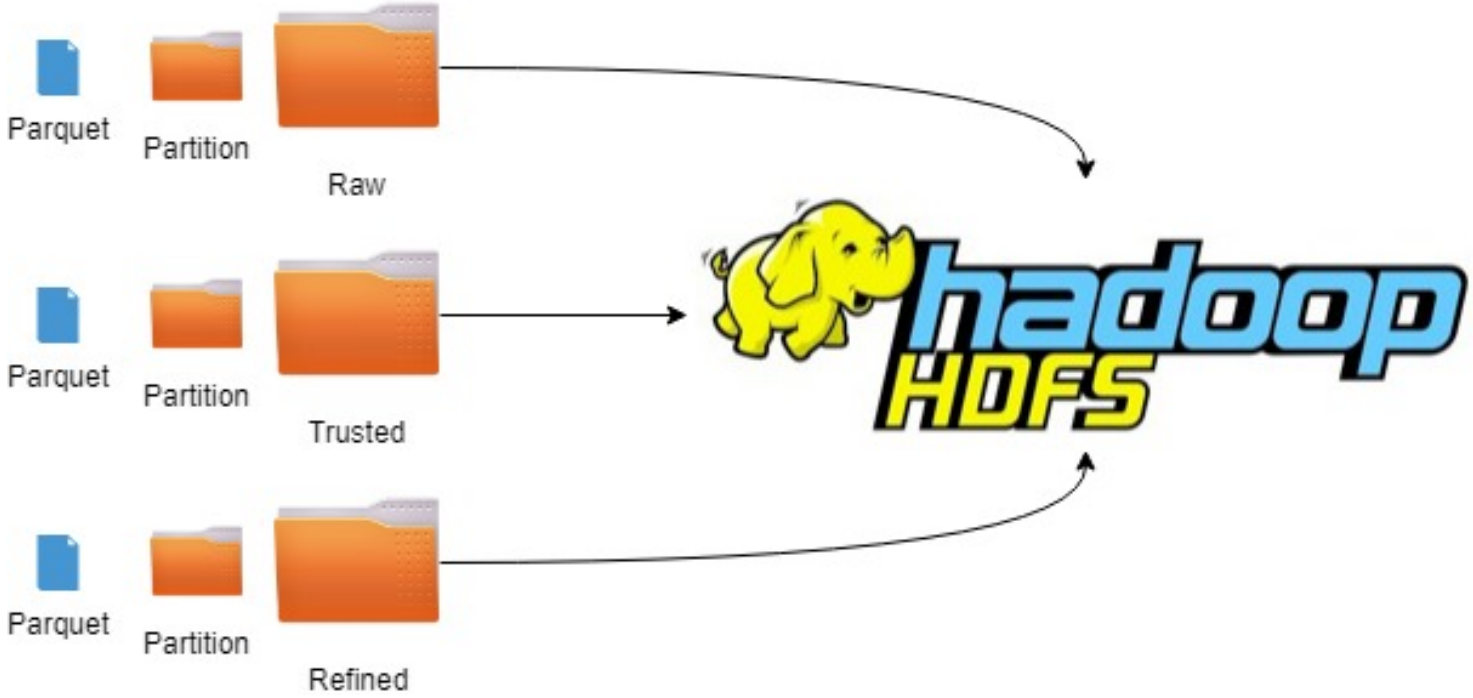
ORC

- Orientado por coluna (armazenar dados em colunas): os armazenamentos de dados orientados por coluna são otimizados para cargas de trabalho analíticas pesadas em leitura
- Altas taxas de compressão (ZLIB)
- Suporte ao tipo Hive (datetime, decimal e os tipos complexos como struct, list, map e union)
- Metadados armazenados usando buffers de protocolo, que permitem adição e remoção de campos
- Compatível com HiveQL
- Suporte a Serialização

Referência: https://cwiki.apache.org/confluence/display/Hive/LanguageManual+ORC#LanguageManualORC-ORCFiles

DIGITAL INNOVATION ONE

Camadas



Cada uma das 3 camadas se tratam de subdiretórios dentro do sistema de arquivos distribuídos, HDFS, e podem ser mapeados através da propriedade LOCATION.

DIGITAL INNOVATION ONE

HDFS

Referência dos Schemas, incluindo local/nodes com os arquivos

File 1

File 1 File 1 File 1

$ hdfs dfs -du -h /<path>

Size * Replication Factory

Arquivos físicos

Apache Derby

MariaDB

DIGITAL INNOVATION ONE

Parte 2: Spark SQL

Criando pipelines de dados eficientes - Parte 1

Spark Streaming

DIGITAL INNOVATION ONE

DataFrame

```
# Leitura de Dataframe

## Opção 1
df1 = spark.read.format("csv").option("header","true").load(path_dataset1)

## Opção 2
df1 = spark.read.csv(path_dataset1)
df1 = spark.read.option("header","true").option("inferSchema","true").csv(path_dataset1)

## Exibindo dataframe
df1.show()
```

DIGITAL INNOVATION ONE

DataFrame

```
## Outras formas de leitura de arquivos com PySpark

path = "/../../arquivoXPTO"

# Criando um dataframe a partir de um JSON
dataframe = spark.read.json(path)

# Criando um dataframe a partir de um ORC
dataframe = spark.read.orc(path)

# Criando um dataframe a partir de um PARQUET
dataframe = spark.read.parquet(path)
```

DataFrame

```
# Leitura de um RDD

rdd = sc.textFile(path_rdd)
#rdd.show() = Errado, não é possível exibir um SHOW() de um RDD, somente um Dataframe
rdd.collect()
```

DataFrame

```
# Criando uma tabela temporária

nome_tabela_temporiaria = "tempTableDataFrame1"
df1.createOrReplaceTempView(nome_tabela_temporiaria)
```

DIGITAL INNOVATION ONE

DataFrame

```
nome_tabela_temporiaria = "tempTableDataFrame1"
df1.createOrReplaceTempView(nome_tabela_temporiaria)

# Lendo a tabela temporaria opcao 1
spark.read.table(nome_tabela_temporiaria).show()

# Lendo a tabela temporaria opcao 2
spark.sql("SELECT * FROM tempTableDataFrame1").show()
```

DIGITAL INNOVATION ONE

DataFrame

```
# Visualização do Databricks
display(spark.sql("SELECT * FROM tempTableDataFrame1"))
```

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hundred	people_vaccinated_per_hur
1	Argentina	ARG	2020-12-29	700	null	null	null	null	0	null
2	Argentina	ARG	2020-12-30	null	null	null	null	15656	null	null
3	Argentina	ARG	2020-12-31	32013	null	null	null	15656	0.07	null
4	Argentina	ARG	2021-01-01	null	null	null	null	11070	null	null
5	Argentina	ARG	2021-01-02	null	null	null	null	8776	null	null
6	Argentina	ARG	2021-01-03	null	null	null	null	7400	null	null
7	Argentina	ARG	2021-01-04	39599	null	null	null	6483	0.09	null
8	Argentina	ARG	2021-01-05	null	null	null	null	7084	null	null

Showing the first 1000 rows.

DataFrame

```
# Scala
#import org.apache.spark.sql.functions._

# Python
from pyspark.sql.functions import col, column

# Usando function col ou column
df1.select(col("country"), col("date"), column("iso_code")).show()

# Usando selectExpr
df1.selectExpr("country", "date", "iso_code").show()
```

DIGITAL INNOVATION ONE

DataFrame

```
# Scala import
# org.apache.spark.sql.types._

# Criando um Schema manualmente no PySpark
from pyspark.sql.types import *

dataframe_ficticio = StructType([
                    StructField("col_String_1", StringType()),
                    StructField("col_Integer_2", IntegerType()),
                    StructField("col_Decimal_3", DecimalType())
                        ])
dataframe_ficticio
```

DIGITAL INNOVATION ONE

DataFrame

```
# Função para gerar Schema (campos/colunas/nomes de colunas)
'''
# Scala
org.apache.spark.sql.types._
def getSchema(fields : Array[StructField]) : StructType = {
  new StructType(fields)
}
'''
# PySpark
def getSchema(fields):
  return StructType(fields)

schema = getSchema([StructField("coluna1", StringType()), StructField("coluna2", StringType()), StructField("coluna3", StringType())])
```

DataFrame

```
#Show
df1.show(2)

#Take
df1.take(2)
```

DataFrame

```
# Gravando um novo CSV

path_destino="/FileStore/tables/CSV/"
nome_arquivo="arquivo.csv"
path_geral= path_destino + nome_arquivo
df1.write.format("csv").mode("overwrite").option("sep", "\t").save(path_geral)
```

DIGITAL INNOVATION ONE

DataFrame

```
# Gravando um novo JSON

path_destino="/FileStore/tables/JSON/"
nome_arquivo="arquivo.json"
path_geral= path_destino + nome_arquivo
df1.write.format("json").mode("overwrite").save(path_geral)
```

DataFrame

```
# Gravando um novo PARQUET

path_destino="/FileStore/tables/PARQUET/"
nome_arquivo="arquivo.parquet"
path_geral= path_destino + nome_arquivo
df1.write.format("parquet").mode("overwrite").save(path_geral)
```

DIGITAL INNOVATION ONE

DataFrame

```
# Gravando um novo ORC

path_destino="/FileStore/tables/ORC/"
nome_arquivo="arquivo.orc"
path_geral= path_destino + nome_arquivo
df1.write.format("orc").mode("overwrite").save(path_geral)
```

DIGITAL INNOVATION ONE

DataFrame

```
# Outros tipos de SELECT

#Diferentes formas de selecionar uma coluna

from pyspark.sql.functions import *

df1.select("country").show(5)
df1.select('country').show(5)
df1.select(col("country")).show(5)
df1.select(column("country")).show(5)
df1.select(expr("country")).show(5)
```

DIGITAL INNOVATION ONE

DataFrame

```
# Define uma nova coluna com um valor constante
df2 = df1.withColumn("nova_coluna", lit(1))

# Adicionar coluna
teste = expr("total_vaccinations < 40")
df1.select("country", "total_vaccinations").withColumn("teste", teste).show(5)

# Renomear uma coluna
df1.select(expr("total_vaccinations as total_de_vacinados")).show(5)
df1.select(col("country").alias("pais")).show(5)
df1.select("country").withColumnRenamed("country", "pais").show(5)

# Remover uma coluna
df3 = df1.drop("country")
df3.columns
```

DIGITAL INNOVATION ONE

DataFrame

```
# Filtrando dados e ordenando
# where() é um alias para filter().

# Seleciona apenas os primeiros registros da coluna "total_vaccinations"
df1.filter(df1.total_vaccinations > 55).orderBy(df1.total_vaccinations).show(2)

# Filtra por país igual Argentina
df1.select(df1.total_vaccinations, df1.country).filter(df1.country == "Argentina").show(5)

# Filtra por país diferente Argentina
df1.select(df1.total_vaccinations, df1.country).where(df1.country != "Argentina").show(5) # python type
```

DIGITAL INNOVATION ONE

DataFrame

```
# Filtrando dados e ordenando

# Mostra valores únicos
df1.select("country").distinct().show()

# Especificando vários filtros em comando separados
filtro_vacinas = df1.total_vaccinations < 100
filtro_pais = df1.country.contains("Argentina")
df1.select(df1.total_vaccinations, df1.country, df1.vaccines).where(df1.vaccines.isin("Sputnik V", "Sinovac")).filter(filtro_vacinas).show(5)
df1.select(df1.total_vaccinations, df1.country, df1.vaccines).where(df1.vaccines.isin("Sputnik V",
"Sinovac")).filter(filtro_vacinas).withColumn("filtro_pais", filtro_pais).show(5)
```

DataFrame

```
'''################################################################################################################################
Convertendo dados
################################################################################################################################'''

df5 = df1.withColumn("PAIS", col("country").cast("string").alias("PAIS"))
df5.select(df5.PAIS).show(2)

'''################################################################################################################################
Trabalhando com funções
################################################################################################################################'''

# Usando funções
df1.select(upper(df1.country)).show(3)
df1.select(lower(df1.country)).show(4)
```

DIGITAL
INNOVATION
ONE

DataFrame

JOINS

DIGITAL INNOVATION ONE

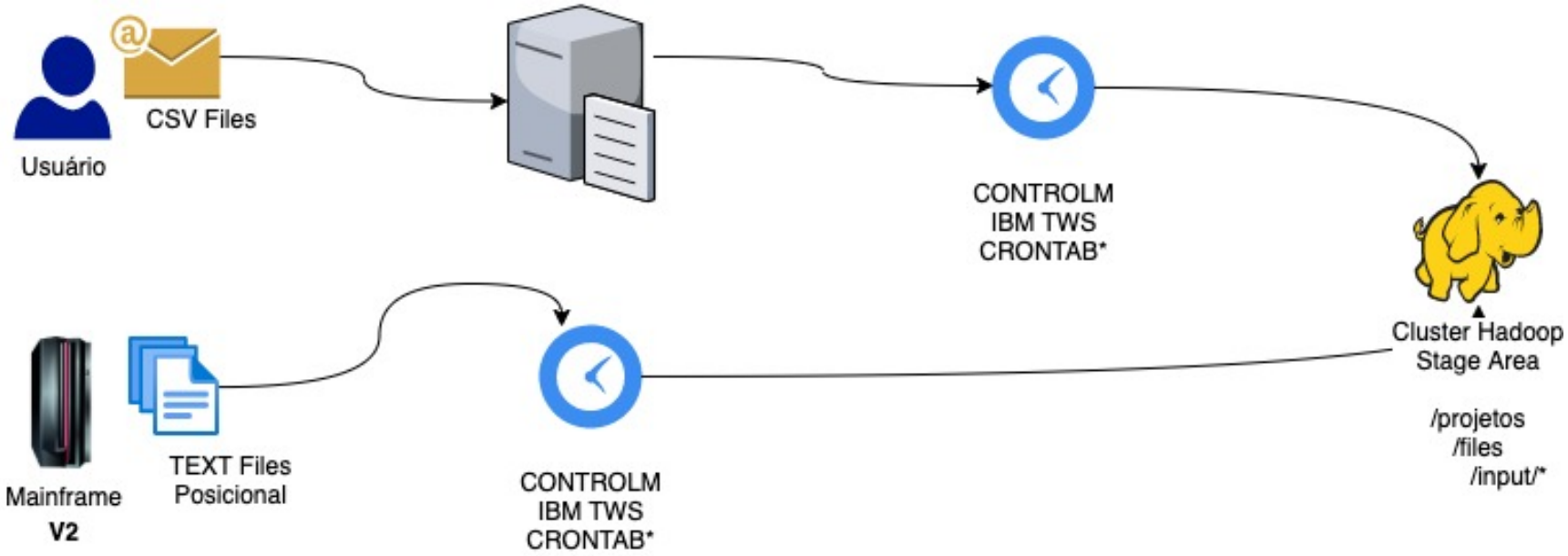
Parte 3: Casos de Uso

Criando pipelines de dados eficientes - Parte 1

Spark Streaming

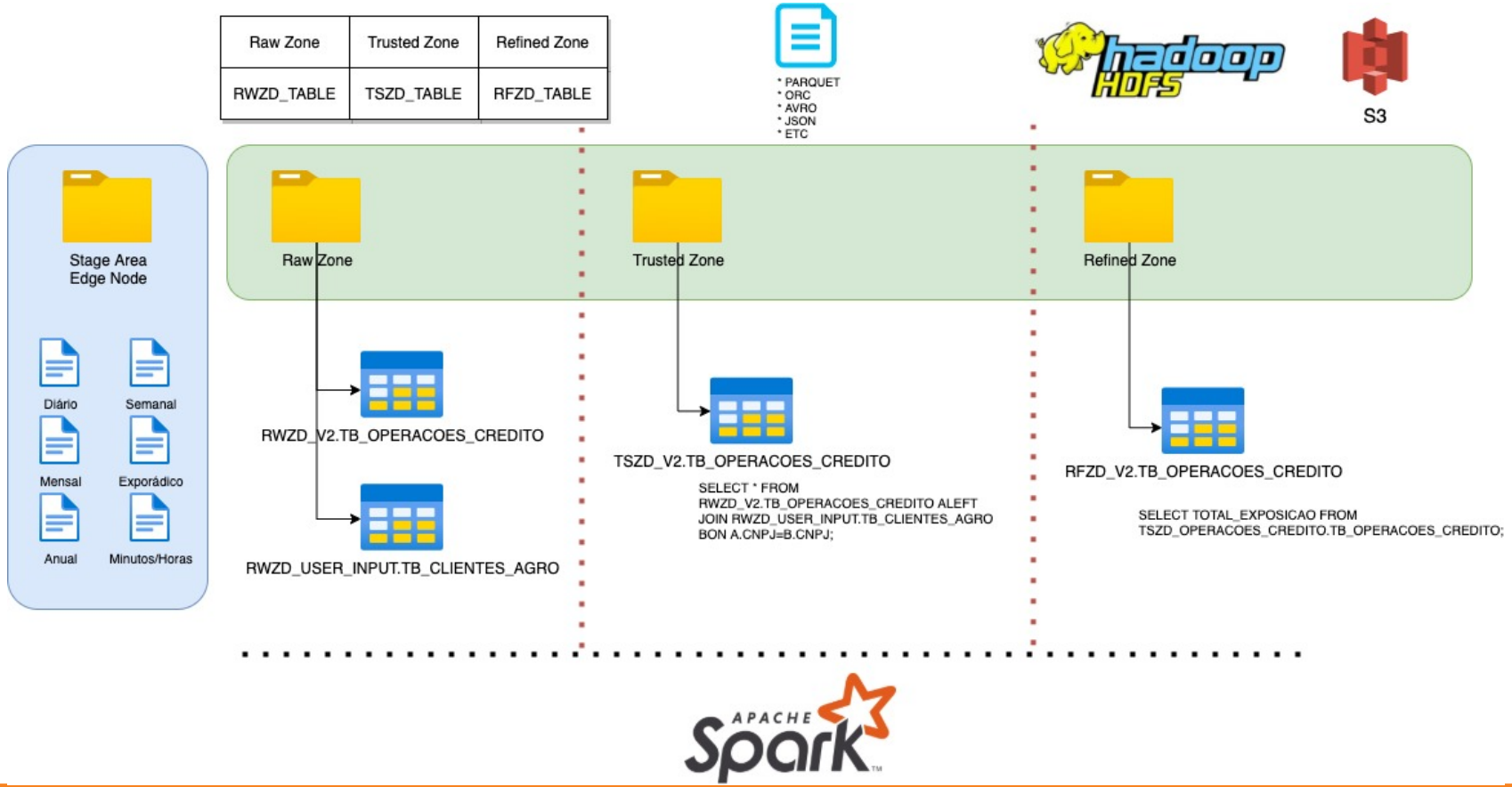
DIGITAL INNOVATION ONE

Caso de Uso 1



DIGITAL INNOVATION ONE

Caso de Uso 2



DIGITAL INNOVATION ONE

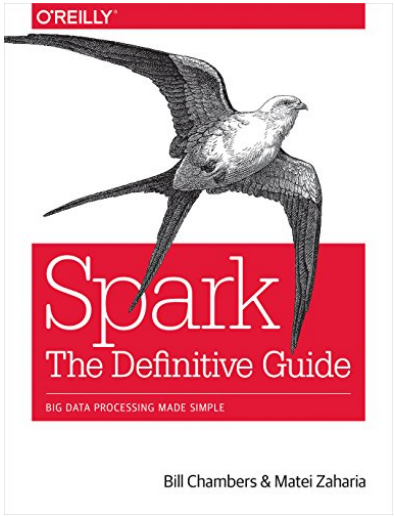
Parte 4: Encerramento

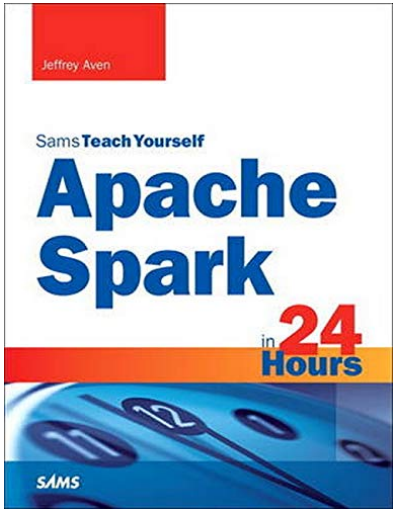
Criando pipelines de dados eficientes - Parte 1

Spark Streaming

DIGITAL INNOVATION ONE

Referências

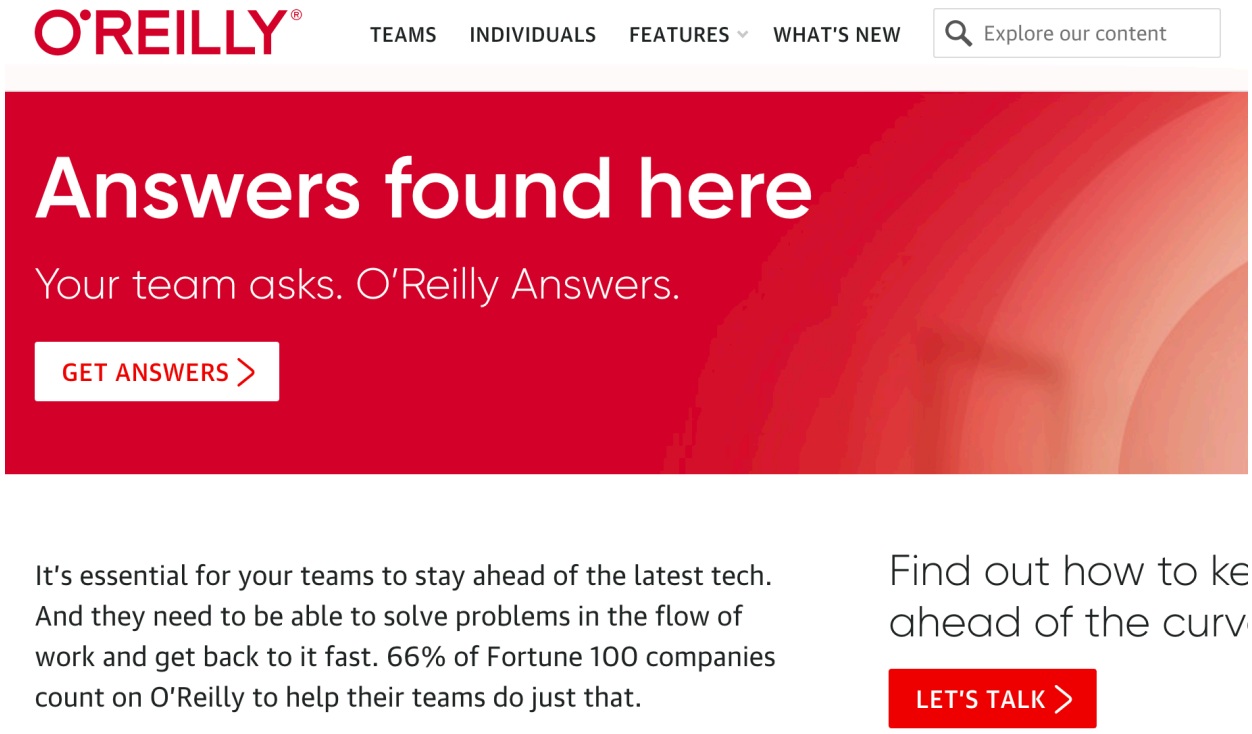




Documentação: https://spark.apache.org/docs/2.4.0/api/python/pyspark.sql.html

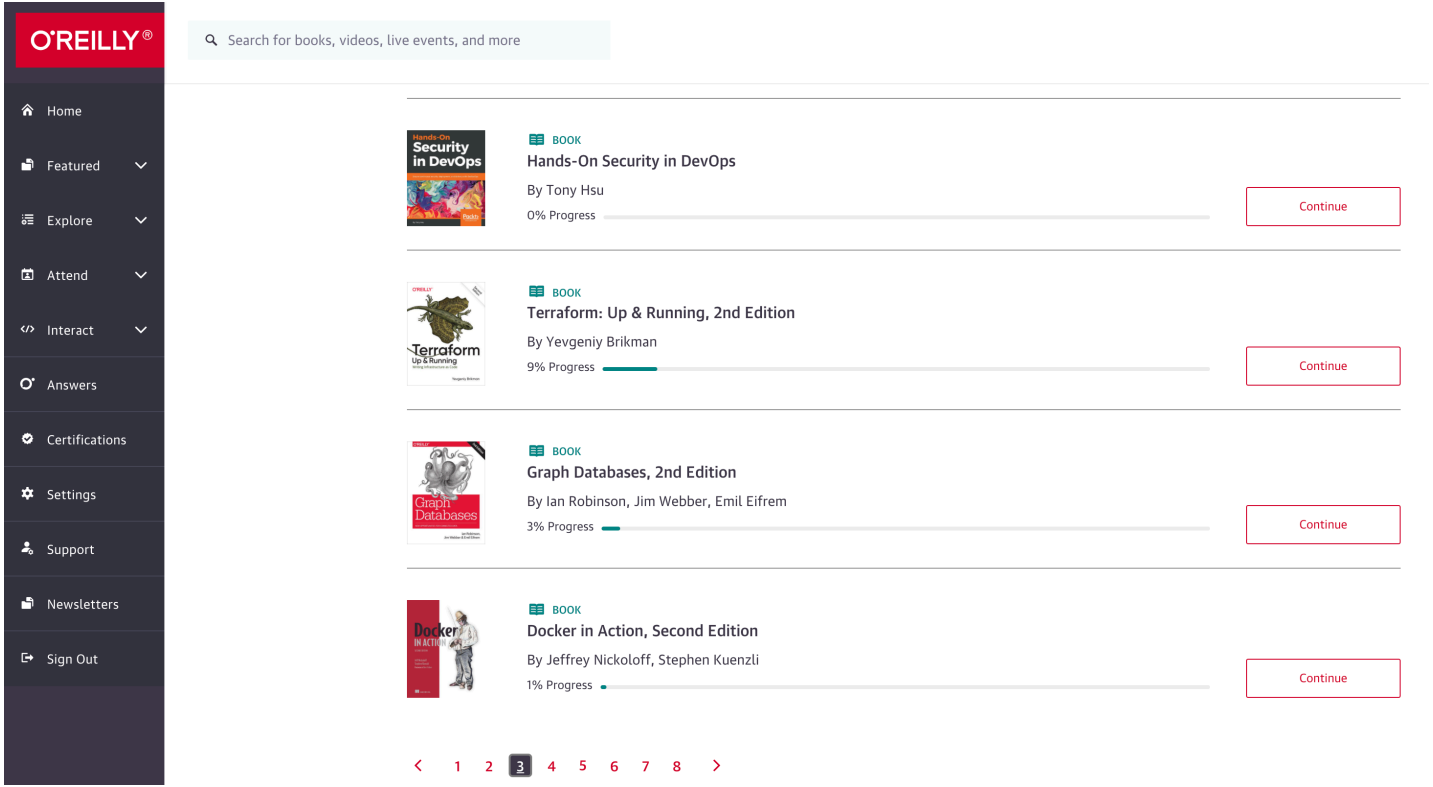
Referências

Safari Books: https://www.safaribooksonline.com/



Referências

Safari Books: https://www.safaribooksonline.com/



DIGITAL INNOVATION ONE

Referências

https://academy.databricks.com/exam/INT-ADAS-v2-CT

Preparation

The following Databricks courses should help you prepare for this exam:

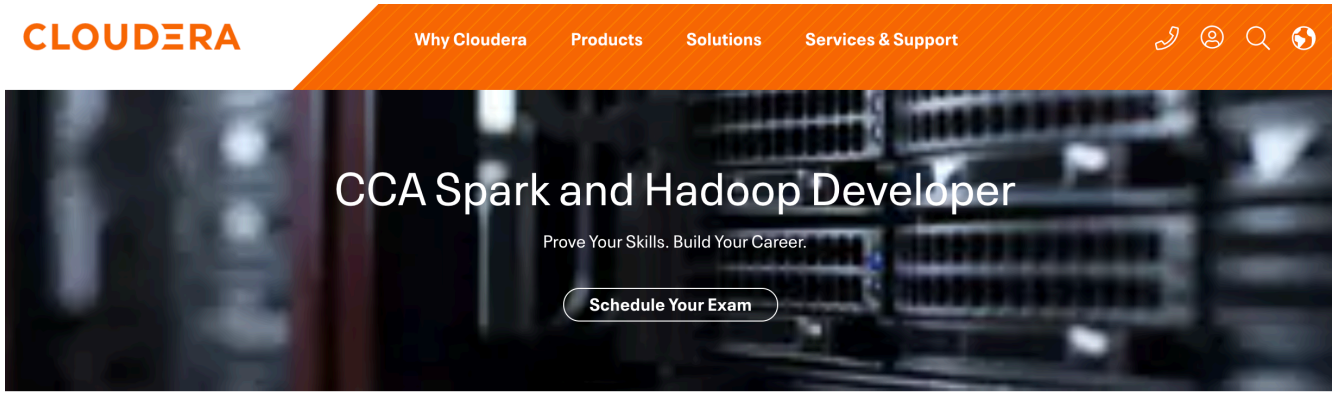
- DB 105 - Apache Spark Programming
- Quick Reference: Spark Architecture
- Future self-paced course on the Spark DataFrames API

In addition, Sections I, II, and IV of *Spark: The Definitive Guide* should also be helpful in preparation.

DIGITAL INNOVATION ONE

Certificações

Cloudera: https://www.cloudera.com/about/training/certification/cca-spark.html



CCA Spark and Hadoop Developer

CCA Spark and Hadoop Developer Exam (CCA175)

- **Number of Questions:** 8–12 performance-based (hands-on) tasks on Cloudera Enterprise cluster. See below for full cluster configuration
- **Time Limit:** 120 minutes
- **Passing Score:** 70%
- **Language:** English
- **Price:** USD $295

Purchase

Watch a free **OnDemand course** to help prepare for your certification

Have questions? Read our **Certification FAQ**

Verify a certification

Contact us at **certification@cloudera.com**

DIGITAL INNOVATION ONE

Certificações

Cloudera: https://www.cloudera.com/about/training/certification/ccp-data-engineer.html

CCP Data Engineer

CCP Data Engineer Exam (DE575)

- **Number of Questions:** 5–10 performance-based (hands-on) tasks on pre-configured Cloudera Enterprise cluster.
- **Time Limit:** 240 minutes
- **Passing Score:** 70%
- **Language:** English
- **Price:** USD $400

Purchase

Have questions? Read our **Certification FAQ**

Verify a Certification

Contact us at **certification@cloudera.com**

DIGITAL INNOVATION ONE

Certificações

Databricks: https://academy.databricks.com/category/certifications

Assessments

Databricks Certified Associate Developer for Apache Spark 3.0 – *Assessment*

$ 200.00 USD

The Databricks Certified Associate Developer for Apache Spark 3.0 certification exam assesses an understanding of the basics of the Spark architecture and the ability to apply the Spark DataFrame API to complete individual data manipulation tasks.

☆ VIEW

Databricks Certified Associate Developer for Apache Spark 2.4 – *Assessment*

$ 200.00 USD

The Databricks Certified Associate Developer for Apache Spark 2.4 certification exam assesses an understanding of the basics of the Spark architecture and the ability to apply the Spark DataFrame API to complete individual data manipulation tasks.

☆ VIEW

Databricks Certified Associate ML Practitioner for Apache Spark 2.4 – *Assessment*

$ 200.00 USD

The Databricks Certified Associate ML Practitioner for Apache Spark 2.4 certification exam assesses the understanding of and ability to apply machine techniques using the Spark ML library.

☆ VIEW

Databricks Certified Professional Data Scientist – *Assessment*

$ 200.00 USD

The Databricks Certified Professional Data Scientist certification exam assesses the understanding of the basics of machine learning, the steps in the machine learning lifecycle, the understanding of basic machine learning algorithms and techniques, and the understanding of the basics of machine learning model management.

☆ VIEW

Databricks Certified Professional Data Engineer – *Assessment*

$ 200.00 USD

Coming soon (early 2021).

☆ VIEW

DIGITAL
INNOVATION
ONE

Dúvidas?

Criando pipelines de dados eficientes - Parte 1
Spark Streaming