

Spark – Big Data Processing

Aula 4



Semantix[®]

All about data

Quem sou eu?



Rodrigo Augusto Rebouças

Engenheiro de dados da Semantix
Instrutor do Semantix Academy

Contatos

rodrigo.augusto@semantix.com.br
[linkedin.com/in/rodrigo-reboucas](https://www.linkedin.com/in/rodrigo-reboucas)



DataFrame

- Criação de Schemas

Tipagem de Dados

- Spark SQL e DataFrame

- ByteType
- ShortType
- IntegerType
- LongType
- FloatType
- DoubleType
- DecimalType
- BigDecimal
- StringType
- BinaryType
- BooleanType
- TimestampType
- DateType
- ArrayType
- MapType
- StructType

- <https://spark.apache.org/docs/latest/sql-ref-datatypes.html>

Esquemas - Definição

Scala

- Inferir esquema manualmente em dados com cabeçalho

```
scala>import org.apache.spark.sql.types._
```

```
scala> val columnsList = List(  
    StructField("id", IntegerType),  
    StructField("setor", StringType) )
```

```
scala> val setorSchema = StructType(columnsList)
```

```
scala> val setorDF = spark.read.option("header","true"). \  
schema(setorSchema).csv("setor.csv")
```

setor.csv

```
id, setor  
1, vendas  
2, TI  
3, RH
```

Esquemas - Definição

Python

- Inferir esquema manualmente em dados com cabeçalho

```
from pyspark.sql.types import *

columns_list = [
    StructField("id", StringType() ),
    StructField("setor", StringType() )]

setor_schema = StructType(columns_list)

setor_df = spark.read.option("header","true"). \
    schema(setor_schema).csv("setor.csv")
```

setor.csv

```
id, setor
1, vendas
2, TI
3, RH
```

DataFrame

- Testar Schemas

Esquemas - Testar

Python

```
from pyspark.sql.types import *

columns_list = [
    StructField("id", StringType() ),
    StructField("setor", StringType() )]

setor_schema = StructType(columns_list)

dados_teste = [Row(1, "vendas"), Row(2,"TI"), Row(3,'RH') ]

setor_df = spark.createDataFrame(data= dados_teste, schema=setor_schema)
```


Esquemas - Testar

Python

- Row class

```
from pyspark.sql import Row

Schema = Row("id","setor")
dados_teste = [ Schema(1, "vendas"), Schema(2,"TI"), Schema(3,'RH') ]

teste_df = spark.createDataFrame(data= dados_teste)

teste_df.printSchema()
```

Exercício – Spark Schemas

1. Criar o DataFrame `names_us_sem_schema` para ler os arquivos no HDFS `“/user/rodrigo/data/names”`
2. Visualizar o Schema e os 5 primeiros registos do `names_us_sem_schema`
3. Criar o DataFrame `names_us` para ler os arquivos no HDFS `“/user/<nome>/data/names”` com o seguinte schema:
 - `nome: String`
 - `sexo: String`
 - `quantidade: Inteiro`
4. Visualizar o Schema e os 5 primeiros registos do `names_us`
5. Salvar o DataFrame `names_us` no formato `orc` no `hdfs` `“/user/<nome>/names_us_orc”`

Dataset

- **Conceitos**

Dataset

- Coleção distribuída de objetos de tipagem forte
 - Tipos primitivos: Int ou String
 - Tipos Complexos: Array ou listas
 - Product objects
 - Scala – case classes
 - Java – JavaBen objects
 - Row objects
- Mapeado para um schema relacional
 - Schema é definido por um encoder
 - Schema mapea objeto de propriedades para tipos de colunas
- Otimizada pelo Catalyst
- Implementado apenas em Java e Scala

<https://spark.apache.org/docs/latest/api/scala/org/apache/spark/sql/Dataset.html>

<https://spark.apache.org/docs/latest/api/java/index.html?org/apache/spark/sql/Dataset.html>

DataFrame x Dataset

- DataFrames (Conjuntos de dados de Row objects)
 - Representam dados tabulares
 - Transformações são não tipadas
 - Linhas podem conter elementos de qualquer tipo
 - Schemas definidos não são aplicados os tipos de coluna até o tempo de execução
- Datasets
 - Representam dados tipados e orientados a objeto
 - Transformações são tipadas
 - Propriedades do objeto são tipadas em tempo de compilação
- Representação dos dados
 - RDD - 2011
 - Dataframe - 2013
 - Dataset - 2015

Criação de Dataset



Criação de Dataset

- Criar Dataset com case classes (Recomendada)
 - case class <NomeClasse>(<Atributo>:<Tipo>, ..., <Atributo>:<Tipo>)
- Exemplo:

```
case class Name(id: Integer, name: String)
reg = Seq(Name(1,"Rodrigo"),Name(2,"Augusto"))
regDS = spark.createDataset(reg)
regDS.show

//Imprimir para cada linha só o name
regDS.foreach(n => println(n.name))
```

```
+---+-----+
| id|   name|
+---+-----+
|  1|Rodrigo|
|  2|Augusto|
+---+-----+
```

Criação de Dataset de DataFrame

- Ler dados estruturados para um DataFrame
- Criar um Dataset para ler os dados do DataFrame
- Forçar para inserir um schema com o Encoder

```
case class Name(id: Integer, name: String)
```

```
val regDF = spark.read.json("registros.json")  
val regDS = regDF.as[Name]  
regDS.show
```

```
import org.apache.spark.sql.Encoders  
val schema = Encoders.product[Name].schema  
val regDS = spark.read.schema(Name).json("registros.json")  
.as[Name]
```

```
+---+-----+  
| id|   name|  
+---+-----+  
|  1|Rodrigo|  
|  2|Augusto|  
+---+-----+
```


Criação de Dataset de RDD

- Ler dados não estruturados ou semi estruturados para um RDD
- Criar um Dataset para ler os dados do RDD

```
case class PcodeLatLon(pcode: String,  
latlon: Tuple2[Double,Double])  
  
val pLatLonRDD = sc.textFile("latlon.tsv").map(_.split('\t')) \  
.map(fields =>(PcodeLatLon(fields(0),(fields(1).toFloat,fields(2).toFloat))))  
  
val pLatLonDS = spark.createDataset(pLatLonRDD)  
  
pLatLonDS.printSchema  
  
println(pLatLonDS.first)
```

Transformações de Dataset



Transformações

- Transformações tipadas criam um novo Dataset
 - Filter
 - Limit
 - Sort
 - flatMap
 - Map
 - orderBy
- Transformações não tipadas retornam DataFrames ou colunas não tipadas
 - Join
 - groupBy
 - Col
 - Drop
 - Select
 - withColumn

Exemplo de transformações

- Tipadas: Dataset
- Não tipadas: DataFrame

```
scala> val sortedDS = regDS.sort("name")
sortedDS: org.apache.spark.sql.Dataset[Name] = [id: int, name: string]
```

```
scala> val nameDF = regDS.select("name")
nameDF: org.apache.spark.sql.DataFrame = [name: string]
```

```
scala> val combineDF = regDS.sort("name").
  where("id > 10").select("name")
combineDF: org.apache.spark.sql.DataFrame = [name: string]
```

```
+---+-----+
| id|   name|
+---+-----+
|  1|Rodrigo|
|  2|Augusto|
+---+-----+
```

Salvar Dataset

- São salvos como DataFrames
 - Dataset.write
 - Retorna um DataFrameWriter
 - Exemplo

```
regDS.write.save("hdfs://localhost/user/cloudera/registros/")
```

```
regDS.write.json("registros")
```

Exercícios - Dataset com DataFrame

1. Criar o DataFrame names_us para ler os arquivos no HDFS “/user/<nome>/data/names”
2. Visualizar o Schema do names_us
3. Mostras os 5 primeiros registros do names_us
4. Criar um case class Nascimento para os dados do names_us
5. Criar o Dataset names_ds para ler os dados do HDFS “/user/<nome>/data/names”, com uso do case class Nascimento
6. Visualizar o Schema do names_ds
7. Mostras os 5 primeiros registros do names_ds
8. Salvar o Dataset names_ds no hdfs “/user/<nome>/ names_us_parquet” no formato parquet com compressão snappy



Semantix[®]

All about data

contato@semantix.com.br

www.semantix.com.br