# Testes Spark Lab Semantix

**Perguntas sobre Spark - Teste Semantix**

## Qual o objetivo do comando cache em Spark?

Persiste(grava) dados em memória, é uma técnica de otimização de performance, dados parciais podem ser persistidos em memória para acelerar computações ou reutizados nos processamentos para evitar repetições(actions, ações) desnecessárias e assim aumentar a performance de um longo processamento.

## O mesmo código implementado em Spark é normalmente mais rápido que a implementação equivalente em MapReduce. Por quê?

O modelo de aplicativo do Spark através do uso de RDDs(coleções distribuídas), “lazy evaluation”, e DAG(Graphs de dependência entre as operações) permite processamento em paralelo(MPP) muito mais simplificado e eficiente que o modelo map/reduce Java no Hadoop, um aplicativo desenvolvido em Spark(Scala ou Python) também é muito mais simples se comparado a mesma implementação em Java map/reduce.

## Qual é a função do SparkContext ?

É a criação de uma instância de aplicativo no Spark, é uma analogia a uma conexão a um banco de dados relacional onde você especifica os parâmetros de configuração, memória, e recursos, nele você cria a conexão e o aplicativo, depois é utilizado para criar sessions(SparkSessions, SQLContext, HIVEContext), e outras instâncias do Spark Core

## Explique com suas palavras o que é Resilient Distributed Datasets (RDD).

É a abstração de dados básica do Spark, imutável, particionado e distribuído, esta abstração é a base da computação paralela do Spark.

## GroupByKey é menos eficiente que reduceByKey em grandes dataset. Por quê?

GroupByKey e reduceByKey são “Wide transformations”, ou seja, produzem multiplas referências de dependências das partições filhas, mas na transformação reduceByKey o Spark combina a saída em uma chave comum em cada partição antes de fazer “shuffle” nos dados, o mesmo não acontece com GroupBykey onde todos os pares chave-valor fazem “shuffle” por entre os executores, causando muito mais processamento para o agrupamento, e uma função .sum deve ser aplicada e só é executada após todo o agrupamento ser concluído.

## Explique o que o código Scala abaixo faz.

```
val textFile = sc.textFile ( "hdfs://..." )
val counts = textFile . flatMap ( line => line . split ( " " ))
    .map ( word => ( word , 1 ))
    .reduceByKey ( _ + _ )
counts.saveAsTextFile ( "hdfs://..." )
```

***Geral***: Este código é uma implementação simples de Word Count(contar palavras em um texto).
***Passo a passo***:
1 - Lê um ou mais arquivos de texto do armazenamento HDFS do Hadoop.
2 - Serializa o texto com a função flatMap, lê uma linha por vez.
3 - Para cada linha faz as transformações ->separa as palavras por espaço-> para cada palavra faz um map e cria uma tupla com o número 1->para cada tupla agrupa pela chave palavra e soma o número( + )->atribui a saída a counts.
4 - Grava o conteúdo de counts em um arquivo de texto em um armazenamento HDFS no Hadoop.

**Autor**: Ricardo Kurz dos Santos

## Efetua o download dos arquivos de log da Nasa

```
%sh
rm -f NASA_access_log*
wget ftp://ita.ee.lbl.gov/traces/NASA_access_log_Jul95.gz -q
wget ftp://ita.ee.lbl.gov/traces/NASA_access_log_Aug95.gz -q

gunzip NASA_access_log_Jul95.gz
gunzip NASA_access_log_Aug95.gz
ls
```

```
NASA_access_log_Aug95
NASA_access_log_Jul95
```

## Carga dos arquivos no storage HDFS

```
%sh
hdfs dfs -rm -f /tmp/NASA_access_log*
hdfs dfs -put NASA_access_log* /tmp/
hdfs dfs -ls /tmp/NASA_access_log*
```

```
18/06/04 11:12:33 INFO fs.TrashPolicyDefault: Moved: 'hdfs://master01.blackdragon.local:8020/tmp/NASA_access_log_Jul95.gz' to trash at: hdfs://master01.blackdragon.local:8020/use
r/zeppelin/.Trash/Current/tmp/NASA_access_log_Jul95.gz
-rw-r--r--   3 zeppelin hdfs  167813770 2018-06-04 11:12 /tmp/NASA_access_log_Aug95
-rw-r--r--   3 zeppelin hdfs  205242368 2018-06-04 11:12 /tmp/NASA_access_log_Jul95
```

## Lê os arquivos no HDFS e grava em um dataframe

```
%spark2
val log_file = spark.read.text("/tmp/NASA_access_log*")
```

```
log_file: org.apache.spark.sql.DataFrame = [value: string]
```

## Pré-visualização dos textos de log

```
%spark2
log_file.show(5, truncate=false)
```

```
+-----------------------------------------------------------------------------------------------------------------------+
|value                                                                                                                  |
+-----------------------------------------------------------------------------------------------------------------------+
|in24.inetnebr.com - - [01/Aug/1995:00:00:01 -0400] "GET /shuttle/missions/sts-68/news/sts-68-mcc-05.txt HTTP/1.0" 200 1839|
|uplherc.upl.com - - [01/Aug/1995:00:00:07 -0400] "GET / HTTP/1.0" 304 0                                                |
|uplherc.upl.com - - [01/Aug/1995:00:00:08 -0400] "GET /images/ksclogo-medium.gif HTTP/1.0" 304 0                       |
|uplherc.upl.com - - [01/Aug/1995:00:00:08 -0400] "GET /images/MOSAIC-logosmall.gif HTTP/1.0" 304 0                     |
|uplherc.upl.com - - [01/Aug/1995:00:00:08 -0400] "GET /images/USA-logosmall.gif HTTP/1.0" 304 0                        |
+-----------------------------------------------------------------------------------------------------------------------+
only showing top 5 rows
```

## Extração dos campos usando Regex, cria um novo dataframe com as colunas correspondentes às informações, faz cache em memória para acesso rápido

```
%spark2
import org.apache.spark.sql.functions._

val log_file_df = log_file.select(regexp_extract($"value", "^([^\\s]+\\s)", 1).as("host"),
    regexp_extract($"value", "^.*\\[(\\d\\d/\\w{3}/\\d{4}:\\d{2}:\\d{2}:\\d{2} -\\d{4})]", 1).as("timestamp"),
    regexp_extract($"value", "^.*\"\\w+\\s+([^\\s]+)\\s+HTTP.*\"", 1).as("requisicao"),
    regexp_extract($"value", "^.*\"\\s+([^\\s]+)", 1).cast("integer").as("codigo_http"),
    regexp_extract($"value", "^.*\\s+(\\d+)$", 1).cast("integer").as("bytes_retorno")
    )
log_file_df.cache()
```

```
import org.apache.spark.sql.functions._
log_file_df: org.apache.spark.sql.DataFrame = [host: string, timestamp: string ... 3 more fields]
res35: log_file_df.type = [host: string, timestamp: string ... 3 more fields]
```

## Pré-visualização e validação do esquema

```
%spark2
log_file_df.columns
log_file_df.printSchema
```

```
res36: Array[String] = Array(host, timestamp, requisicao, codigo_http, bytes_retorno)
root
 |-- host: string (nullable = true)
 |-- timestamp: string (nullable = true)
 |-- requisicao: string (nullable = true)
 |-- codigo_http: integer (nullable = true)
```

```
 |-- bytes_retorno: integer (nullable = true)
```

## Validação das colunas numéricas quanto aos nulos

```
%spark2
log_file.filter($"value".isNull).count()
log_file.filter(!$"value".rlike("\\d+$")).show(5, truncate=false)
log_file_df.filter($"codigo_http".isNull).count()
log_file_df.filter($"bytes_retorno".isNull).count()
log_file_df.filter($"bytes_retorno".isNull).groupBy($"codigo_http").count().show()
```

```
|tia1.eskimo.com - - [01/Aug/1995:00:28:41 -0400] "GET /pub/winvn/release.txt HTTP/1.0" 404 -                                |
|itws.info.eng.niigata-u.ac.jp - - [01/Aug/1995:00:38:01 -0400] "GET /ksc.html/facts/about_ksc.html HTTP/1.0" 403 -          |
|grimnet23.idirect.com - - [01/Aug/1995:00:50:12 -0400] "GET /www/software/winvn/winvn.html HTTP/1.0" 404 -                  |
+-------------------------------------------------------------------------------------------------------------------------+
only showing top 5 rows
res40: Long = 1
res41: Long = 33905
+-----------+-----+
|codigo_http|count|
+-----------+-----+
|        501|   41|
|       null|    1|
|        400|   15|
|        403|  225|
|        404|20900|
|        200|  161|
|        302|12562|
+-----------+-----+
```

## Existem 33905 ocorrências de nulo em "bytes_retorno", usando o método "na.fill" para preencher com zeros, atribui em um novo dataframe

```
%spark2
val log_file_df_clean = log_file_df.na.fill(0)
log_file_df_clean.filter($"bytes_retorno".isNull).groupBy($"codigo_http").count().show()
```

```
log_file_df_clean: org.apache.spark.sql.DataFrame = [host: string, timestamp: string ... 3 more fields]
+-----------+-----+
|codigo_http|count|
+-----------+-----+
+-----------+-----+
```

## Cria visão temporária para execução de consultas SQL

```
%spark2
log_file_df_clean.createOrReplaceTempView("log_file_view")
```

## Pré-visualização dos dados no Matrix usando SQL

```
%spark2.sql
select * from log_file_view LIMIT 5
```

| host | timestamp | requisicao |
|---|---|---|
| in24.inetnebr.com | 01/Aug/1995:00:00:01 -0400 | /shuttle/missions/sts-68/news/sts-68-mcc-05.txt |
| uplherc.upl.com | 01/Aug/1995:00:00:07 -0400 | / |
| uplherc.upl.com | 01/Aug/1995:00:00:08 -0400 | /images/ksclogo-medium.gif |
| uplherc.upl.com | 01/Aug/1995:00:00:08 -0400 | /images/MOSAIC-logosmall.gif |
| uplherc.upl.com | 01/Aug/1995:00:00:08 -0400 | /images/USA-logosmall.gif |

## Número de hosts únicos

```
%spark2.sql
SELECT COUNT(*) as total_distinct_hosts FROM (SELECT DISTINCT host FROM log_file_view)
```

| total_distinct_hosts |
|---|
| 137979 |

## O total de erros 404

```
%spark2.sql
SELECT COUNT(*) as total_erro_404 FROM log_file_view where codigo_http = 404
```

| total_erro_404 |
|---|
| 20901 |

## Os 5 URLs que mais causaram erro 404

```
%spark2.sql
SELECT requisicao, codigo_http, COUNT(*) as total_erro_404_by_url FROM log_file_view
WHERE codigo_http = 404 GROUP BY requisicao, codigo_http ORDER BY COUNT(*) DESC LIMIT 5
```

| requisicao | codigo_http |
|---|---|
| /pub/winvn/readme.txt | 404 |
| /pub/winvn/release.txt | 404 |
| /shuttle/missions/STS-69/mission-STS-69.html | 404 |
| /shuttle/missions/sts-68/ksc-upclose.gif | 404 |
| /history/apollo/a-001/a-001-patch-small.gif | 404 |

## Quantidade de erros 404 por dia

```
%spark2.sql
SELECT to_date(substr(timestamp, 1, 11),'dd/MMM/yyyy') as data, codigo_http, COUNT(*) as total_erro_404_by_dia from log_file_view
WHERE codigo_http = 404 GROUP BY data, codigo_http ORDER BY data
```

| data | codigo_http | total_erro_404_by_dia |
|---|---|---|
| 1995-07-01 | 404 | 316 |
| 1995-07-02 | 404 | 291 |
| 1995-07-03 | 404 | 474 |
| 1995-07-04 | 404 | 359 |
| 1995-07-05 | 404 | 497 |
| 1995-07-06 | 404 | 640 |
| 1995-07-07 | 404 | 570 |
| 1995-07-08 | 404 | 302 |
| 1995-07-09 | 404 | 348 |

## O total de bytes retornados

```
%spark2.sql
SELECT sum(bytes_retorno) as total_bytes_retorno from log_file_view
```

| total_bytes_retorno |
|---|
| 65524314915 |