http_analysis

Analise de logs HTTP com PySpark

READY

Autor: Bruno F. Bessa

```
%spark.pyspark
# Importacao de pacotes necessarios:

import re
import pyspark.sql.functions
from pyspark.sql.types import *
```

FINISHED

```
%spark.pyspark
# Assegurando-se de que ha uma conexao com Spark:
sc
```

FINISHED

```
<SparkContext master=local[*] appName=Zeppelin>
```

```
%spark.pyspark
# Carga dos arquivos de requisicoes HTTP via Spark Context na forma de RDD.

textFile1 = sc.textFile("NASA_access_log_Jul95")
textFile2 = sc.textFile("NASA_access_log_Aug95")

logLines = textFile1.union(textFile2)
logLines.cache()
```

FINISHED

```
UnionRDD[85] at union at NativeMethodAccessorImpl.java:0
```

A descricao dos registros dos logs e a seguinte:

READY

- host responsavel pela requisicao
- timestamo da data
- requisicao
- codigo de retorno HTTP
- total de bytes retornados

Com estas informacoes, farei a quebra em colunas do arquivo de texto de uma forma mais intuitiva do que utilizando expressoes regulares. Vemos que o formato padrao dos registros segue a forma abaixo:

‘199.72.81.55 - - [01/Jul/1995:00:00:01 -0400] “GET /history/apollo/ HTTP/1.0” 200 6245’

Entao usarei alguns dos caracteres que delimitam os campos para separa-los.

FINISHED

```
%spark.pyspark
# Identificando alguns caracteres que podem ser substituidos por um unico identificador
temp_var = logLines.map(lambda k: k.replace(" - - [", ";"))
temp_var2 = temp_var.map(lambda k: k.replace('] "', ";"))
temp_var3 = temp_var2.map(lambda k: k.replace('" ', ";"))

# Os dois ultimos campos podem ser separados posteriormente pois têâem o formato mais s
# Com o caracter ; podemos separar as variaveis e criar um data frame.
temp_var4 = temp_var3.map(lambda k: k.split(";"))
logLinesDF = temp_var4.toDF()
logLinesDF.show(2, truncate = False)
```

```
+--------------------+--------------------------+--------------------------------+------
--+
|_1                  |_2                        |_3                              |_4
|
+--------------------+--------------------------+--------------------------------+------
--+
|199.72.81.55        |01/Jul/1995:00:00:01 -0400|GET /history/apollo/ HTTP/1.0   |200 62
45|
|unicomp6.unicomp.net|01/Jul/1995:00:00:06 -0400|GET /shuttle/countdown/ HTTP/1.0|200 39
85|
+--------------------+--------------------------+--------------------------------+------
--+
only showing top 2 rows
```

FINISHED

```
%spark.pyspark
#temp_var4.map(lambda x: (1, len(x))).countByValue()
temp_var4.first()
```

```
['199.72.81.55', '01/Jul/1995:00:00:01 -0400', 'GET /history/apollo/ HTTP/1.0', '200 624
5']
```

FINISHED

```
%spark.pyspark
# Identificando alguns caracteres que podem ser substituidos por um unico identificador
temp_var = logLines.map(lambda k: k.replace(";", ","))
temp_var1 = temp_var.map(lambda k: k.replace(" - - [", ";"))
temp_var2 = temp_var1.map(lambda k: k.replace('] "', ";"))
temp_var3 = temp_var2.map(lambda k: k.replace('" ', ";"))

# Os dois ultimos campos podem ser separados posteriormente pois têâem o formato mais s
# Com o caracter ; podemos separar as variaveis e criar um data frame.
temp_var4 = temp_var3.map(lambda k: k.split(";"))
logLinesDF = temp_var4.toDF()
logLinesDF.show(2, truncate = False)
```

```
+------------------+--------------------------+--------------------------------+------
--+
|_1                |_2                        |_3                              |_4
|
+------------------+--------------------------+--------------------------------+------
--+
|199.72.81.55      |01/Jul/1995:00:00:01 -0400|GET /history/apollo/ HTTP/1.0   |200 62
45|
|unicomp6.unicomp.net|01/Jul/1995:00:00:06 -0400|GET /shuttle/countdown/ HTTP/1.0|200 39
85|
+------------------+--------------------------+--------------------------------+------
--+
only showing top 2 rows
```

FINISHED

```
%spark.pyspark
# Verificando como foi feito o parsing:
temp_var4.map(lambda x: (1, len(x))).countByValue()
```

```
defaultdict(<class 'int'>, {(1, 5): 30, (1, 1): 1, (1, 4): 3461582})
```

FINISHED

```
%spark.pyspark
# Tratando a separacao das duas ultimas colunas:
split_col = pyspark.sql.functions.split(logLinesDF["_4"], " ")
logLinesDF = logLinesDF.withColumn("codigo_http", split_col.getItem(0))
logLinesDF = logLinesDF.withColumn("total_bytes", split_col.getItem(1))

# Da mesma forma, removerei o timezone da data-hora:
split_col = pyspark.sql.functions.split(logLinesDF["_2"], " -")
logLinesDF = logLinesDF.withColumn("data_hora_string", split_col.getItem(0))
logLinesDF = logLinesDF.withColumn("timezone", split_col.getItem(1))

logLinesDF.show()
```

```
+--------------------+--------------------+--------------------+---------+-----------+
-----------+--------------------+--------+
|                  _1|                  _2|                  _3|       _4|codigo_http|
total_bytes|    data_hora_string|timezone|
+--------------------+--------------------+--------------------+---------+-----------+
-----------+--------------------+--------+
|        199.72.81.55|01/Jul/1995:00:00...|GET /history/apol...| 200 6245|        200|
6245|01/Jul/1995:00:00:01|    0400|
|unicomp6.unicomp.net|01/Jul/1995:00:00...|GET /shuttle/coun...| 200 3985|        200|
3985|01/Jul/1995:00:00:06|    0400|
|      199.120.110.21|01/Jul/1995:00:00...|GET /shuttle/miss...| 200 4085|        200|
4085|01/Jul/1995:00:00:09|    0400|
|  burger.letters.com|01/Jul/1995:00:00...|GET /shuttle/coun...|    304 0|        304|
0|01/Jul/1995:00:00:11|    0400|
|      199.120.110.21|01/Jul/1995:00:00...|GET /shuttle/miss...| 200 4179|        200|
4179|01/Jul/1995:00:00:11|    0400|
|  burger.letters.com|01/Jul/1995:00:00...|GET /images/NASA-...|    304 0|        304|
0|01/Jul/1995:00:00:12|    0400|
```

FINISHED

```
%spark.pyspark
# Renomeando as colunas do dataframe
logLinesDF = logLinesDF.select(pyspark.sql.functions.col("_1").alias("host"),
                                pyspark.sql.functions.col("data_hora_string").substr(1,
                                pyspark.sql.functions.col("_3").alias("requisicao"),
                                pyspark.sql.functions.col("codigo_http").alias("codigo_
                                pyspark.sql.functions.col("codigo_http").alias("total_by

logLinesDF.show()
```

```
+--------------------+-----------+--------------------+-----------+-----------+
|                host|data_string|          requisicao|codigo_http|total_bytes|
+--------------------+-----------+--------------------+-----------+-----------+
|        199.72.81.55|01/Jul/1995|GET /history/apol...|        200|        200|
|unicomp6.unicomp.net|01/Jul/1995|GET /shuttle/coun...|        200|        200|
|      199.120.110.21|01/Jul/1995|GET /shuttle/miss...|        200|        200|
|  burger.letters.com|01/Jul/1995|GET /shuttle/coun...|        304|        304|
|      199.120.110.21|01/Jul/1995|GET /shuttle/miss...|        200|        200|
|  burger.letters.com|01/Jul/1995|GET /images/NASA-...|        304|        304|
|  burger.letters.com|01/Jul/1995|GET /shuttle/coun...|        200|        200|
|     205.212.115.106|01/Jul/1995|GET /shuttle/coun...|        200|        200|
|         d104.aa.net|01/Jul/1995|GET /shuttle/coun...|        200|        200|
|      129.94.144.152|01/Jul/1995|      GET / HTTP/1.0|        200|        200|
|unicomp6.unicomp.net|01/Jul/1995|GET /shuttle/coun...|        200|        200|
|unicomp6.unicomp.net|01/Jul/1995|GET /images/NASA-...|        200|        200|
|unicomp6.unicomp.net|01/Jul/1995|GET /images/KSC-l...|        200|        200|
|         d104.aa.net|01/Jul/1995|GET /shuttle/coun...|        200|        200|
|         d104.aa.net|01/Jul/1995|GET /images/NASA-...|        200|        200|
```

FINISHED

Respondendo as questoes do desafio (observacao importante):

Como houve linhas no RDD que foram escritas com numero diferente de colunas do que o esperado (4) conforme exibido no passo de verificacao do parsing, o dataframe e corrompido, impedindo a visualizacao de dados. Este desafio esta reproduzido na internet em diferentes sites e decidi nao alterar minha solucao utilizando codigos de terceiros pois nao ha valor neste tipo de pratica. Prefiro exibir minhas proprias habilidades, mesmo que com alguma limitacao.

Os metodos spark utilizados a seguir sao aqueles que acredito que trariam a resposta correta para as perguntas do questionario, porem a execucao retorna erros devido ao erro exposto acima.

```
%spark.pyspark
# Qual o numero de hosts unicos?
logLinesDF.groupBy("host").count().filter("count = 1").select("host").show()
```

ERROR

```
Traceback (most recent call last):
  File "/tmp/zeppelin_pyspark-7194404941686918311.py", line 360, in <module>
    exec(code, _zcUserQueryNameSpace)
  File "<stdin>", line 1, in <module>
  File "/usr/spark-2.2.0/python/pyspark/sql/dataframe.py", line 336, in show
    print(self._jdf.showString(n, 20))
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/java_gateway
.py", line 1160, in __call__
    answer, self.gateway_client, self.target_id, self.name)
  File "/usr/spark-2.2.0/python/pyspark/sql/utils.py", line 63, in deco
    return f(*a, **kw)
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/protocol.py"
, line 320, in get_return_value
    format(target_id, ".", name), value)
py4j.protocol.Py4JJavaError: An error occurred while calling o1203.showString.
: org.apache.spark.SparkException: Job aborted due to stage failure: Task 0 in stage 7
8.0 failed 1 times, most recent failure: Lost task 0.0 in stage 78.0 (TID 194, localho
st, executor driver): java.lang.IllegalStateException: Input row doesn't have expected
```

```
%spark.pyspark
# Qual o total de erros 404?
logLinesDF.groupBy("codigo_http").count().filter("codigo_http = '404'").show()
```

ERROR

```
Traceback (most recent call last):
  File "/tmp/zeppelin_pyspark-7194404941686918311.py", line 360, in <module>
    exec(code, _zcUserQueryNameSpace)
  File "<stdin>", line 1, in <module>
  File "/usr/spark-2.2.0/python/pyspark/sql/dataframe.py", line 336, in show
    print(self._jdf.showString(n, 20))
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/java_gateway
.py", line 1160, in __call__
    answer, self.gateway_client, self.target_id, self.name)
  File "/usr/spark-2.2.0/python/pyspark/sql/utils.py", line 63, in deco
    return f(*a, **kw)
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/protocol.py"
, line 320, in get_return_value
    format(target_id, ".", name), value)
py4j.protocol.Py4JJavaError: An error occurred while calling o1251.showString.
: org.apache.spark.SparkException: Job aborted due to stage failure: Task 0 in stage 8
0.0 failed 1 times, most recent failure: Lost task 0.0 in stage 80.0 (TID 197, localho
st, executor driver): java.lang.IllegalStateException: Input row doesn't have expected
```

```
%spark.pyspark
# Quais os 5 URLs que mais causaram erro 404?
logLinesDF.filter("codigo_http = '404'").groupBy("requisicao").count().sort(pyspark.sql
```

ERROR

```
Traceback (most recent call last):
  File "/tmp/zeppelin_pyspark-7194404941686918311.py", line 360, in <module>
    exec(code, _zcUserQueryNameSpace)
  File "<stdin>", line 1, in <module>
  File "/usr/spark-2.2.0/python/pyspark/sql/dataframe.py", line 338, in show
    print(self._jdf.showString(n, int(truncate)))
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/java_gateway
.py", line 1160, in __call__
    answer, self.gateway_client, self.target_id, self.name)
  File "/usr/spark-2.2.0/python/pyspark/sql/utils.py", line 63, in deco
    return f(*a, **kw)
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/protocol.py"
, line 320, in get_return_value
    format(target_id, ".", name), value)
py4j.protocol.Py4JJavaError: An error occurred while calling o1312.showString.
: org.apache.spark.SparkException: Job aborted due to stage failure: Task 0 in stage 8
2.0 failed 1 times, most recent failure: Lost task 0.0 in stage 82.0 (TID 200, localho
st, executor driver): java.lang.IllegalStateException: Input row doesn't have expected
```

```
%spark.pyspark
# Qual a quantidade de erros 404 por dia?
logLinesDF.filter("codigo_http = '404'").groupBy("data_string").count().show()
```

ERROR

```
Traceback (most recent call last):
  File "/tmp/zeppelin_pyspark-7194404941686918311.py", line 360, in <module>
    exec(code, _zcUserQueryNameSpace)
  File "<stdin>", line 1, in <module>
  File "/usr/spark-2.2.0/python/pyspark/sql/dataframe.py", line 336, in show
    print(self._jdf.showString(n, 20))
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/java_gateway
.py", line 1160, in __call__
    answer, self.gateway_client, self.target_id, self.name)
  File "/usr/spark-2.2.0/python/pyspark/sql/utils.py", line 63, in deco
    return f(*a, **kw)
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/protocol.py"
, line 320, in get_return_value
    format(target_id, ".", name), value)
py4j.protocol.Py4JJavaError: An error occurred while calling o1370.showString.
: org.apache.spark.SparkException: Job aborted due to stage failure: Task 0 in stage 8
4.0 failed 1 times, most recent failure: Lost task 0.0 in stage 84.0 (TID 203, localho
st, executor driver): java.lang.IllegalStateException: Input row doesn't have expected
```

ERROR

```
%spark.pyspark
# Qual o total de bytes retornados?
logLinesDF.select("total_bytes").groupBy().sum().show()
```

```
Traceback (most recent call last):
  File "/tmp/zeppelin_pyspark-7194404941686918311.py", line 360, in <module>
    exec(code, _zcUserQueryNameSpace)
  File "<stdin>", line 1, in <module>
  File "/usr/spark-2.2.0/python/pyspark/sql/dataframe.py", line 336, in show
    print(self._jdf.showString(n, 20))
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/java_gateway
.py", line 1160, in __call__
    answer, self.gateway_client, self.target_id, self.name)
  File "/usr/spark-2.2.0/python/pyspark/sql/utils.py", line 63, in deco
    return f(*a, **kw)
  File "/usr/local/lib/python3.4/dist-packages/py4j-0.10.6-py3.4.egg/py4j/protocol.py"
, line 320, in get_return_value
    format(target_id, ".", name), value)
py4j.protocol.Py4JJavaError: An error occurred while calling o1422.showString.
: org.apache.spark.SparkException: Job aborted due to stage failure: Task 0 in stage 8
6.0 failed 1 times, most recent failure: Lost task 0.0 in stage 86.0 (TID 206, localho
st, executor driver): java.lang.IllegalStateException: Input row doesn't have expected
```

Conclusao

FINISHED

Analises quantitativas com Spark requer um bom trabalho de limpeza e normalizacao dos dados. As particularidades relacionadas as caracteristicas de programacao funcional podem ser superadas pois a API para Python e muito robusta e prove bom grau de abstracao para criacao de aplicacoes.

```
%md
```

FINISHED