How do I add a new column to spark data frame (Pyspark)?

I have a Spark data frame (using Pyspark 1.5.1) and would like to add a new column.

Tried the following without any success:

```
type(randomed_hours) # => list

#Create in Python and transform to RDD

new_col = pd.DataFrame(randomed_hours,columns=['new_col'])

spark_new_col = sqlContext.createDataFrame(new_col)

my_df_spark.withColumn("hours",  spark_new_col["new_col"])
```

Also got an error using this:

```
my_df_spark.withColumn("hours",  sc.parallelize(randomed_hours))
```

So how do I add a new column (based on Python vector) to existing Data frame with PySpark ?

Thanks ! Boris

python apache-spark apache-spark-sql pyspark

edited Nov 12 '15 at 23:17

zero323
66.4k 16 78 138

asked Nov 12 '15 at 21:14

Boris
53 1 6

Please, add highlight to the code – Alberto Bonsanto Nov 12 '15 at 21:39

2 Answers

You cannot add an arbitrary column to a `DataFrame` in Spark. New columns can be created only by using literals:

```
from pyspark.sql.functions import lit

df = sqlContext.createDataFrame(
    [(1, "a", 23.0), (3, "B", -23.0)], ("x1", "x2", "x3"))

df_with_x4 = df.withColumn("x4", lit(0))
df_with_x4.show()

## +---+---+-----+---+
## | x1| x2|   x3| x4|
## +---+---+-----+---+
## |  1|  a| 23.0|  0|
## |  3|  B|-23.0|  0|
## +---+---+-----+---+
```

transforming an existing column:

```
from pyspark.sql.functions import exp
```

```
df_with_x5 = df_with_x4.withColumn("x5", exp("x3"))
df_with_x5.show()

## +---+---+-----+---+--------------------+
## | x1| x2|   x3| x4|                  x5|
## +---+---+-----+---+--------------------+
## |  1|  a| 23.0|  0| 9.744803446248903E9|
## |  3|  B|-23.0|  0|1.026187963170189...|
## +---+---+-----+---+--------------------+
```

included using `join`:

```
from pyspark.sql.functions import exp

lookup = sqlContext.createDataFrame([(1, "foo"), (2, "bar")], ("k", "v"))
df_with_x6 = (df_with_x5
    .join(lookup, col("x1") == col("k"), "leftouter")
    .drop("k")
    .withColumnRenamed("v", "x6"))

## +---+---+-----+---+--------------------+----+
## | x1| x2|   x3| x4|                  x5|  x6|
## +---+---+-----+---+--------------------+----+
## |  1|  a| 23.0|  0| 9.744803446248903E9| foo|
## |  3|  B|-23.0|  0|1.026187963170189...|null|
## +---+---+-----+---+--------------------+----+
```

or generated with function / udf:

```
from pyspark.sql.functions import rand

df_with_x7 = df_with_x6.withColumn("x7", rand())
df_with_x7.show()

## +---+---+-----+---+--------------------+----+-------------------+
## | x1| x2|   x3| x4|                  x5|  x6|                 x7|
## +---+---+-----+---+--------------------+----+-------------------+
## |  1|  a| 23.0|  0| 9.744803446248903E9| foo|0.41930610446846617|
## |  3|  B|-23.0|  0|1.026187963170189...|null|0.37801881545497873|
## +---+---+-----+---+--------------------+----+-------------------+
```

If you want to add content of an arbitrary RDD as a column you can

- add row numbers to existing data frame
- call `zipWithIndex` on RDD and convert it to data frame

- join both using index as a join key

edited Nov 13 '15 at 0:31

answered Nov 12 '15 at 23:37

zero323
66.4k 16 78 138

To add a column using a UDF:

```
df = sqlContext.createDataFrame(
    [(1, "a", 23.0), (3, "B", -23.0)], ("x1", "x2", "x3"))

from pyspark.sql.functions import udf
from pyspark.sql.types import *

def valueToCategory(value):
   if   value == 1: return 'cat1'
   elif value == 2: return 'cat2'
   ...
   else: return 'n/a'

# NOTE: it seems that calls to udf() must be after SparkContext() is called
udfValueToCategory = udf(valueToCategory, StringType())
df_with_cat = df.withColumn("category", udfValueToCagetory("x1"))
df_with_cat.show()

## +---+---+-----+---------+
## | x1| x2|   x3| category|
## +---+---+-----+---------+
## |  1|  a| 23.0|     cat1|
## |  3|  B|-23.0|      n/a|
## +---+---+-----+---------+
```

answered May 16 at 22:04

Mark Rajcok
196k 56 312 326