How can I convert an array column to a string in PySpark?

Authored by

stats writer

June 24, 2024

In PySpark, an array column can be converted to a string by using the "concat_ws" function. This function allows you to specify a delimiter and combines the elements of the array into a single string. The resulting string can then be used in further data processing or analysis. By converting an array column to a string, it allows for easier manipulation and access of the data within the array. This conversion can be useful in various data analysis and machine learning tasks.

In this PySpark article, I will explain how to convert an array of String column on DataFrame to a String column (separated or concatenated with a comma, space, or any delimiter character) using PySpark function `concat_ws()` (translates to concat with separator), and with SQL expression using Scala example.

When curating data on DataFrame we may want to convert the Dataframe with complex struct datatypes, arrays and maps to a flat structure. here we will see how to convert array type to string type.

Before we start, first let's create a DataFrame with array of string column.

```
spark = SparkSession.builder.appName('SparkByExamples.com').getOrCreate()
```

columns =
data = ,"CA"),
("Michael,Rose,",,"NJ"),
("Robert,,Williams",,"NV")]

df = spark.createDataFrame(data=data,schema=columns)
df.printSchema()
df.show(truncate=False)

In this example "languagesAtSchool" is a column of type array. In the next section, we will convert this to a String. This example yields below schema and DataFrame.

```
root
|-- name: string (nullable = true)
|-- languagesAtSchool: array (nullable = true)
| |-- element: string (containsNull = true)
|-- currentState: string (nullable = true)
```

+----------------+-----------------+------------+
|name |languagesAtSchool |currentState|
+----------------+-----------------+------------+

James,,Smith		CA
Michael,Rose,		NJ
Robert,,Williams		NV
+----------------+-----------------+------------+

Convert an array of String to String column using concat_ws()

In order to convert array to a string, PySpark SQL provides a built-in function `concat_ws()` which takes delimiter of your choice as a first argument and array column (type Column) as the second argument.

Syntax

```
concat_ws(sep, *cols)
```

Usage

In order to use `concat_ws()` function, you need to import it using `pyspark.sql.functions.concat_ws` . Since this function takes the Column type as a second argument, you need to use `col()`.

```
from pyspark.sql.functions import col, concat_ws
df2 = df.withColumn("languagesAtSchool",
concat_ws(",",col("languagesAtSchool")))
df2.printSchema()
df2.show(truncate=False)
```

This yields below output

```
root
|-- name: string (nullable = true)
|-- languagesAtSchool: string (nullable = false)
|-- currentState: string (nullable = true)
```

+----------------+----------------+------------+
|name |languagesAtSchool|currentState|
+----------------+----------------+------------+
|James,,Smith |Java,Scala,C++ |CA |

|Michael,Rose, |Spark,Java,C++ |NJ |
|Robert,,Williams|CSharp,VB |NV |
+----------------+----------------+------------+

Using PySpark SQL expression

You can also use `concat_ws()` function with SQL expression.

```
df.createOrReplaceTempView("ARRAY_STRING")
spark.sql("select name, concat_ws(',',languagesAtSchool) as languagesAtSchool,"
+
" currentState from ARRAY_STRING")
.show(truncate=False)
```

Complete Example

Below is a complete PySpark DataFrame example of converting an array of String column to a String using a Scala example.

```
import pyspark
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName('SparkByExamples.com').getOrCreate()
```

columns =
data = ,"CA"),
("Michael,Rose,",,"NJ"),
("Robert,,Williams",,"NV")]

df = spark.createDataFrame(data=data,schema=columns)
df.printSchema()
df.show(truncate=False)

from pyspark.sql.functions import col, concat_ws
df2 = df.withColumn("languagesAtSchool",
concat_ws(",",col("languagesAtSchool")))
df2.printSchema()
df2.show(truncate=False)

df.createOrReplaceTempView("ARRAY_STRING")

```
spark.sql("select name, concat_ws(',',languagesAtSchool) as languagesAtSchool," +
" currentState from ARRAY_STRING")
.show(truncate=False)
```

This example is also available at the PySpark Github example project for reference.

Hope it helps you !! Thanks for reading.

Related Articles