How can I convert a PySpark DataFrame to a Pandas DataFrame?

Authored by

stats writer

June 24, 2024

Converting a PySpark DataFrame to a Pandas DataFrame allows for easier data manipulation and analysis in a familiar Python environment. To do so, the PySpark DataFrame must first be converted into a Pandas DataFrame using the "toPandas()" function. This function collects all the data from the PySpark DataFrame and transforms it into a Pandas DataFrame, which can then be used for further data processing and analysis. This conversion is particularly useful for users who are more comfortable working with Pandas and its various libraries for data manipulation and visualization.

(Spark with Python) PySpark DataFrame can be converted to Python pandas DataFrame using a function `toPandas()`, In this article, I will explain how to create Pandas DataFrame from PySpark (Spark) DataFrame with examples.

Before we start first understand the main differences between the Pandas & PySpark, operations on Pyspark run faster than Pandas due to its distributed nature and parallel execution on multiple cores and machines.

In other words, pandas run operations on a single node whereas PySpark runs on multiple machines. If you are working on a Machine Learning application where you are dealing with larger datasets, PySpark processes operations many times faster than pandas. Refer to pandas DataFrame Tutorial beginners guide with examples

After processing data in PySpark we would need to convert it back to Pandas DataFrame for a further procession with Machine Learning application or any Python applications.

Key Points -

Prepare PySpark DataFrame

In order to explain with an example first let's create a PySpark DataFrame.

```
import pyspark
from pyspark.sql import SparkSession
```

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

data =

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

This yields below schema and result of the DataFrame.

```
root
|-- first_name: string (nullable = true)
|-- middle_name: string (nullable = true)
|-- last_name: string (nullable = true)
|-- dob: string (nullable = true)
|-- gender: string (nullable = true)
|-- salary: long (nullable = true)
```

```
+----------+-----------+---------+-----+------+------+
|first_name|middle_name|last_name|dob |gender|salary|
+----------+-----------+---------+-----+------+------+
|James | |Smith |36636|M |60000 |
|Michael |Rose | |40288|M |70000 |
|Robert | |Williams |42114| |400000|
|Maria |Anne |Jones |39192|F |500000|
|Jen |Mary |Brown | |F |0 |
+----------+-----------+---------+-----+------+------+
```

Convert PySpark Dataframe to Pandas DataFrame

PySpark DataFrame provides a method `toPandas()` to convert it to Python Pandas DataFrame.

`toPandas()` results in the collection of all records in the PySpark DataFrame to the driver program and should be done only on a small subset of the data. running on larger dataset's results in memory error and crashes the application. To deal with a larger dataset, you can also try increasing memory on the driver.

```
pandasDF = pysparkDF.toPandas()
print(pandasDF)
```

This yields the below panda's DataFrame. Note that pandas add a sequence number to the result as a row Index. You can rename pandas columns by using `rename()` function.

```
first_name middle_name last_name dob gender salary
0 James Smith 36636 M 60000
1 Michael Rose 40288 M 70000
```

```
2 Robert Williams 42114 400000
3 Maria Anne Jones 39192 F 500000
4 Jen Mary Brown F 0
```

I have dedicated Python pandas Tutorial with Examples where I explained pandas concepts in detail.

Convert Spark Nested Struct DataFrame to Pandas

Most of the time data in PySpark DataFrame will be in a structured format meaning one column contains other columns so let's see how it convert to Pandas. Here is an example with nested struct where we have `firstname`, `middlename` and `lastname` are part of the `name` column.

```
# Nested structure elements
from pyspark.sql.types import StructType, StructField, StringType,IntegerType
dataStruct =

schemaStruct = StructType()),
StructField('dob', StringType(), True),
StructField('gender', StringType(), True),
StructField('salary', StringType(), True)
])
df = spark.createDataFrame(data=dataStruct, schema = schemaStruct)
df.printSchema()

pandasDF2 = df.toPandas()
print(pandasDF2)
```

Converting structured DataFrame to Pandas DataFrame results below output.

```
name dob gender salary
0 (James, , Smith) 36636 M 3000
1 (Michael, Rose, ) 40288 M 4000
2 (Robert, , Williams) 42114 M 4000
3 (Maria, Anne, Jones) 39192 F 4000
4 (Jen, Mary, Brown) F -1
```

FAQ on Convert PySpark DataFrame to Pandas

Why would I want to convert a PySpark DataFrame to a pandasDataFrame?

Converting PySpark DataFrames to Pandas allows you to leverage the extensive functionality and ease of use offered by the Pandas library for data manipulation, analysis, and visualization.

Are there any limitations or considerations when converting PySpark DataFrames to Pandas?

It's essential to consider memory constraints, especially when dealing with large datasets. Converting large PySpark DataFrames to DataFrames may lead to out-of-memory errors if the data cannot fit into memory.

Can I convert any PySpark DataFrame to a Pandas DataFrame?

You can convert any PySpark DataFrame to a DataFrame using the `toPandas()` method. However, keep in mind the potential performance implications and ensure compatibility between PySpark and Pandas data types and structures.

How can I optimize the conversion process from PySpark to Pandas for better performance?

Optimizations can include selecting relevant columns before conversion, filtering out unnecessary data, and using appropriate data types to minimize memory usage. Additionally, consider using partitioning and caching in PySpark to optimize performance before converting to Pandas.

Conclusion

In this simple article, you have learned to convert Spark DataFrame to pandas using `toPandas()` function of the Spark DataFrame. Also have seen a similar example with complex nested structure elements. `toPandas()` results in the collection of all records in the DataFrame to the driver program and should be done on a small subset of the data.

Happy Learning !!

Related Articles

References