How can I use the Collect() function in PySpark to retrieve data from a DataFrame?

Authored by

stats writer

June 24, 2024

The Collect() function in PySpark is a useful tool that allows users to retrieve data from a DataFrame. This function collects all the data from a DataFrame and returns it as a list, which can then be used for further analysis or manipulation. By using the Collect() function, users can easily access and work with specific data points within a DataFrame, making it a valuable function for data exploration and processing. Additionally, the Collect() function can be used to gather data from multiple DataFrames, allowing for more advanced data operations. Overall, the Collect() function is an essential feature in PySpark that enhances the data retrieval capabilities of this powerful framework.

PySpark RDD/DataFrame `collect()` is an action operation that is used to retrieve all the elements of the dataset (from all nodes) to the driver node. We should use the collect() on smaller dataset usually after filter(), group() e.t.c. Retrieving larger datasets results in `OutOfMemory` error.

In this PySpark article, I will explain the usage of `collect()` with DataFrame example, when to avoid it, and the difference between `collect()` and `select()`.

Related Articles:

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

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

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

dept =
deptColumns =
deptDF = spark.createDataFrame(data=dept, schema = deptColumns)
deptDF.show(truncate=False)

show() function on DataFrame prints the result of DataFrame in a table format. By default, it shows only 20 rows. The above snippet returns the data in a table.

```
+---------+-------+
|dept_name|dept_id|
+---------+-------+
|Finance |10 |
|Marketing|20 |
|Sales |30 |
|IT |40 |
```

```
+---------+-------+
```

Now, let's use the `collect()` to retrieve the data.

```
dataCollect = deptDF.collect()
print(dataCollect)
```

`deptDF.collect()` retrieves all elements in a DataFrame as an Array of Row type to the driver node. printing a resultant array yields the below output.

Note that `collect()` is an action hence it does not return a DataFrame instead, it returns data in an Array to the driver. Once the data is in an array, you can use `python for loop` to process it further.

```
for row in dataCollect:
print(row + "," +str(row))
```

If you wanted to get first row and first column from a DataFrame.

```
#Returns value of First Row, First Column which is "Finance"
deptDF.collect()
```

Let's understand what's happening on above statement.

In case you want to just return certain elements of a DataFrame, you should call PySpark select() transformation first.

```
dataCollect = deptDF.select("dept_name").collect()
```

When to avoid Collect()

Usually, collect() is used to retrieve the action output when you have very small result set and calling `collect()` on an RDD/DataFrame with a bigger result set causes out of memory as it returns the entire dataset (from all workers) to the driver hence we should avoid calling collect() on

a larger dataset.

collect () vs select ()

`select()` is a transformation that returns a new DataFrame and holds the columns that are selected whereas collect() is an action that returns the entire data set in an Array to the driver.

Complete Example of PySpark collect()

Below is complete PySpark example of using collect() on DataFrame, similarly you can also create a program using collect() with RDD.

```
import pyspark
from pyspark.sql import SparkSession

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

dept =
deptColumns =
deptDF = spark.createDataFrame(data=dept, schema = deptColumns)
deptDF.printSchema()
deptDF.show(truncate=False)

dataCollect = deptDF.collect()

print(dataCollect)

dataCollect2 = deptDF.select("dept_name").collect()
print(dataCollect2)

for row in dataCollect:
print(row + "," +str(row))
```

This example is also available at PySpark Github project.

Conclusion

In this PySpark article, you have learned the collect() function of the RDD/DataFrame is an action operation that returns all elements of the DataFrame to spark driver program and also learned it's not a good practice to use it on the bigger dataset.

Happy Learning !!

Related Articles