How can I use the parallelize() function in PySpark to create an RDD from a list of data?

Authored by

stats writer

June 24, 2024

The parallelize() function in PySpark is used to create a Resilient Distributed Dataset (RDD) from a list of data. This function distributes the data across multiple nodes in a cluster, allowing for parallel processing and efficient execution of operations. By calling the parallelize() function and passing in a list of data, the list is partitioned and stored in memory on the cluster, creating an RDD. This allows for easy and efficient manipulation of large datasets using PySpark's parallel processing capabilities.

PySpark `parallelize()` is a function in SparkContext and is used to create an RDD from a list collection. In this article, I will explain the usage of parallelize to create RDD and how to create an empty RDD with PySpark example.

Before we start let me explain what is RDD, Resilient Distributed Datasets (**RDD**) is a fundamental data structure of PySpark, It is an immutable distributed collection of objects. Each dataset in **RDD** is divided into logical partitions, which may be computed on different nodes of the cluster.

Below is an example of how to create an RDD using a parallelize method from Sparkcontext. `sparkContext.parallelize()` creates an RDD with a list of Integers.

Using sc.parallelize on PySpark Shell or REPL

PySpark shell provides SparkContext variable "sc", use `sc.parallelize()` to create an RDD.

```
rdd = sc.parallelize()
```

Using PySpark sparkContext.parallelize() in application

Since PySpark 2.0, First, you need to create a SparkSession which internally creates a SparkContext for you.

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

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

Now, use `sparkContext.parallelize()` to create rdd from a list or collection.

```
rdd=sparkContext.parallelize()
```

```
rddCollect = rdd.collect()
print("Number of Partitions: "+str(rdd.getNumPartitions()))
print("Action: First element: "+str(rdd.first()))
print(rddCollect)
```

By executing the above program you should see below output.

```
Number of Partitions: 4
Action: First element: 1
```

parallelize() function also has another signature which additionally takes integer argument to specifies the number of partitions. Partitions are basic units of parallelism in PySpark.

Remember, RDDs in PySpark are a collection of partitions.

create empty RDD by using sparkContext.parallelize

Some times we may need to create empty RDD and you can also use parallelize() in order to create it.

```
emptyRDD = sparkContext.emptyRDD()
emptyRDD2 = rdd=sparkContext.parallelize()
```

print("is Empty RDD : "+str(emptyRDD2.isEmpty()))

The complete code can be downloaded from GitHub - PySpark Examples project

Related Articles