Spark properties, Spark RDD, Spark Dataframe

1) Spark properties

Spark properties kiểm soát các cài đặt của ứng dụng và được cấu hình riêng cho từng ứng dụng đó, hay đơn giản nó là phương tiện điều chỉnh môi trường thực thi cho các ứng dụng Spark. Các thuộc tính này có thể được đặt trên SparkConf và được chuyển tới SparkContext để cài đặt. SparkConf cho phép cấu hình một số thuộc tính chung (URL chính, tên ứng dụng, …), cũng như các cặp key-value tùy ý thông qua phương thức set ().

```
conf = SparkConf().setMaster("local").setAppName("Word frequence")
sc = SparkContext.getOrCreate(conf=conf)
```

Các thuộc tính của Spark chủ yếu có thể được chia thành hai loại: một là liên quan đến triển khai như “spark.driver.memory”, “spark.executor.instances”, loại thuộc tính này có thể không bị ảnh hưởng khi được lập trình thông qua SparkConf trong runtime, hoặc với hành vi phụ thuộc vào trình quản lý cụm và chế độ triển khai được chọn; một cái khác chủ yếu liên quan đến kiểm soát thời gian chạy Spark như “spark.task.maxFailures”.

Giá trị mặc định của tệp tin Spark properties là **$SPARK_HOME/conf/spark-defaults.conf.** Chỉ các giá trị được chỉ định rõ ràng thông qua spark-defaults.conf, SparkConf hoặc dòng lệnh mới xuất hiện. Đối với tất cả các thuộc tính cấu hình khác, chỉ giá trị mặc định mới được sử dụng.

2) Spark RDD

RDD (Resilient Distributed Datasets - Bộ dữ liệu phân phối khả năng phục hồi) là một tập các đối tượng phân tán không thay đổi, chịu được lỗi, có khả năng hoạt động song song dùng để hỗ trợ tính toán xử lý trong bộ nhớ. Khi RDD được tạo, ta không thể thay đổi nó. Mỗi bản ghi trong RDD được chia thành các phân vùng logic, có thể được tính toán trên các node khác nhau của cụm máy chủ.

Nói cách khác, RDD là một tập hợp các đối tượng tương tự như danh sách trong Python, nhưng RDD được tính toán trên một số quy trình nằm rải rác trên nhiều phân vùng logic và xử lý chỉ trong một quy trình. Nó lưu trữ trạng thái bộ nhớ như một đối tượng trên các công việc và đối tượng có thể chia sẻ giữa các công việc đó.

Các record trong RDD có thể là đối tượng Java, Scale hay Python

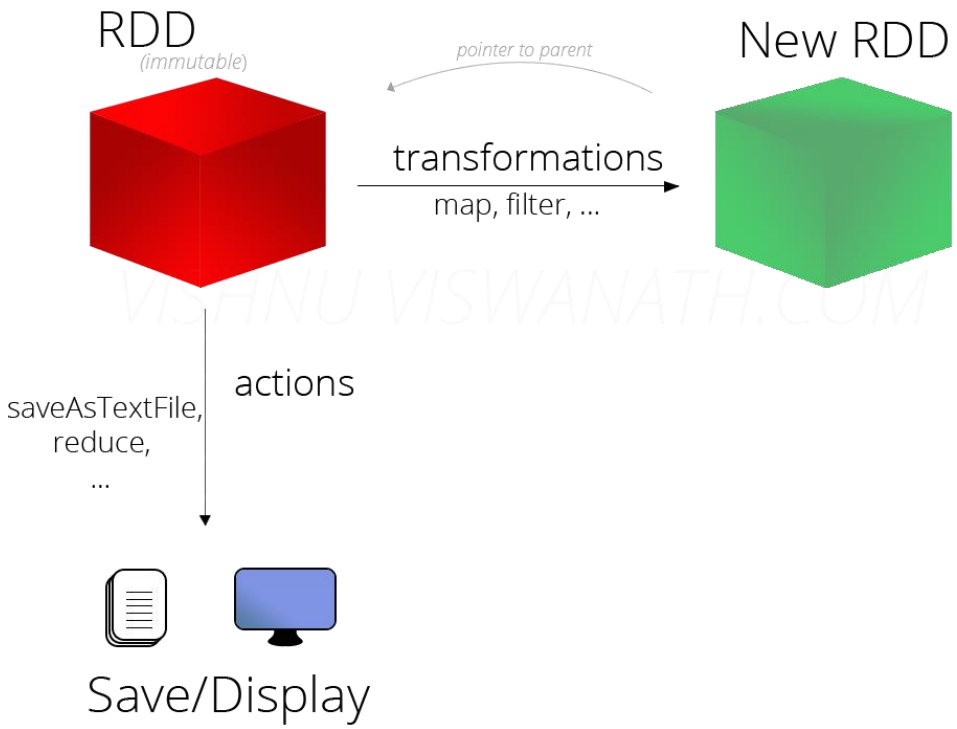
Có hai cách để tạo RDD: tạo song song một tập hợp hiện có trong trình điều khiển hoặc tham chiếu đến tập dữ liệu trong hệ thống lưu trữ bên ngoài.

Tập hợp song song được tạo bằng cách gọi phương thức parallelize của SparkContext. Các phần tử của tập hợp được sao chép để tạo thành một tập dữ liệu phân tán có thể được vận hành song song.

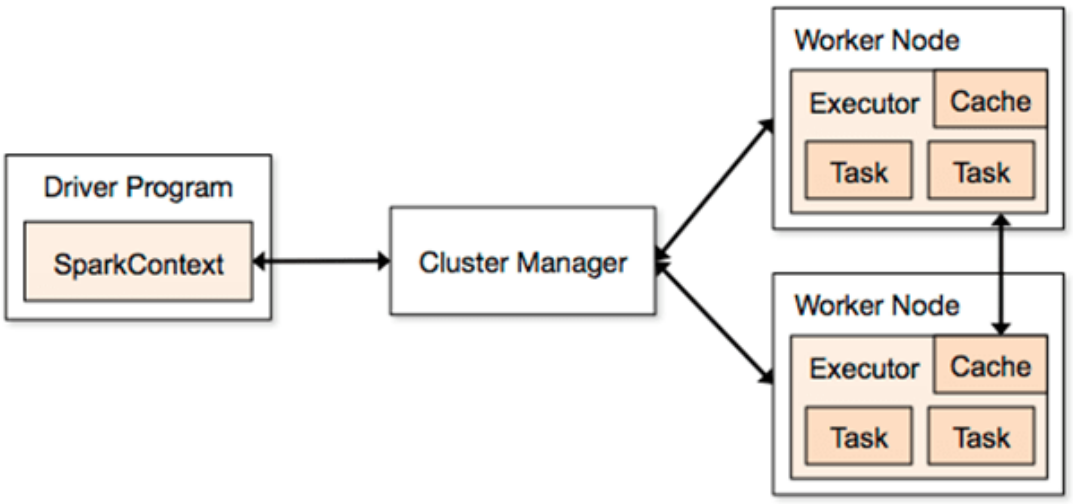
```
data = [1, 2, 3, 4, 5]
distData = sc.parallelize(data)
```

Điều quan trọng ở đây là số lượng phân vùng để chia tập dữ liệu. Spark sẽ chạy một tác vụ cho mỗi phân vùng.

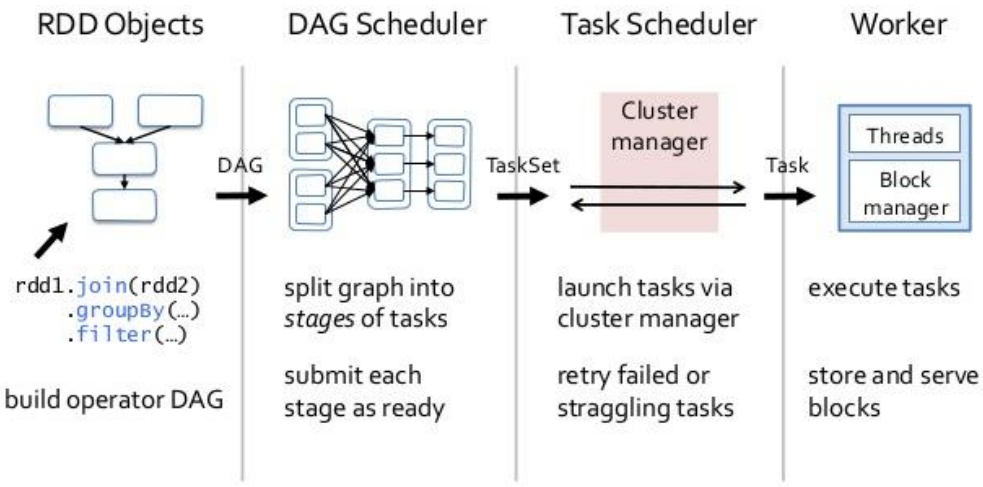
Có 2 loại RDD: transformations - biến đổi: tạo ra một tập dữ liệu mới từ một tập dữ liệu hiện có và action – hành động: trả về một giá trị sau khi chạy một tính toán trên tập dữ liệu.



Spark thực hiện đưa các thao tác RDD chuyển đổi vào DAG (Directed Acyclic Graph) và bắt đầu thực hiện. Khi một action được gọi trên RDD, Spark sẽ tạo DAG và chuyển cho DAG scheduler. DAG scheduler chia các thao tác thành các nhóm (stage) khác nhau của các task. Mỗi stage bao gồm các task dựa trên phân vùng của dữ liệu đầu vào có thể truyền thông tin với nhau và thực hiện một cách độc lập trên một máy worker.

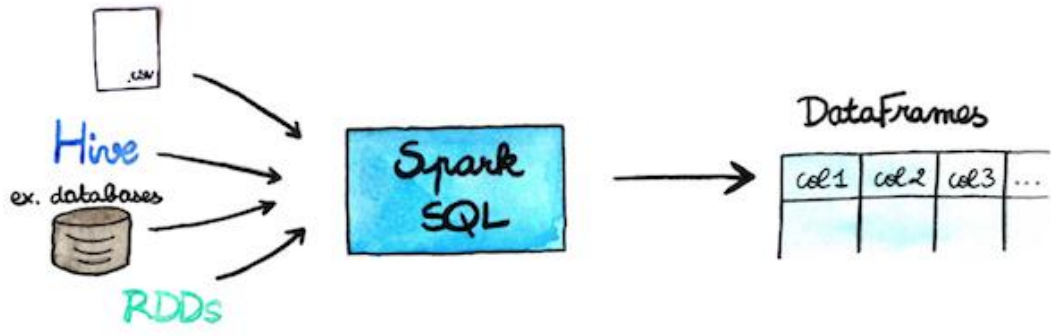


Mỗi worker bao gồm một hoặc nhiều excuter. Các excuter chịu trách nhiệm thực hiện các task trên các luồng riêng biệt. Việc chia nhỏ các task giúp đem lại hiệu năng cao hơn, giảm thiểu ảnh hưởng của dữ liệu không đối xứng.



3) Spark DataFrame

Spark Dataframe là một tập dữ liệu có tổ chức thành các cột được đặt tên cung cấp các hoạt động để lọc, nhóm hoặc tính toán tổng hợp và có thể được sử dụng với Spark SQL. Nó có thể được xây dựng từ nhiều nguồn như: tệp dữ liệu có cấu trúc, cơ sở dữ liệu bên ngoài hoặc RDD hiện có.



Cách tạo dataframe:

Để sử dụng các chức năng trong Spark, ta sử dụng SparkSession bằng cách gọi SparkSession.build.

```
from pyspark.sql import SparkSession

spark = SparkSession \
    .builder \
    .appName("Python Spark SQL basic example") \
    .config("spark.some.config.option", "some-value") \
    .getOrCreate()
```

Với SparkSession, ứng dụng có thể tạo dataframe từ RDD đã có hoặc từ Spark data sources.

Dataframe có thể được xây dựng nhiều nguồn từ nhiều cách khác nhau, do đó cách gọi dataframe là không giống nhau. Ví dụ: tạo dataframe từ tập tin JSON và CSV.

To load a JSON file you can use:

Scala Java Python R

```
df = spark.read.load("examples/src/main/resources/people.json", format="json")
df.select("name", "age").write.save("namesAndAges.parquet", format="parquet")
```

Find full example code at "examples/src/main/python/sql/datasource.py" in the Spark repo.

To load a CSV file you can use:

Scala Java Python R

```
df = spark.read.load("examples/src/main/resources/people.csv",
                     format="csv", sep=":", inferSchema="true", header="true")
```

TÀI LIỆU THAM KHẢO

1. https://spark.apache.org/docs/latest/configuration.html
2. https://spark.apache.org/docs/latest/rdd-programming-guide.html#resilient-distributed-datasets-rdds
3. https://viblo.asia/p/tong-quan-ve-apache-spark-cho-he-thong-big-data-RQqKLxR6K7z
4. https://spark.apache.org/docs/2.3.0/sql-programming-guide.html#datasets-and-dataframes