1. Spark

Spark hay Apache Spark là một trong những khung được sử dụng rộng rãi nhất khi xử lý và làm việc với Big Data, và Python là một trong những ngôn ngữ lập trình được sử dụng rộng rãi nhất cho việc phân tích data. Do đó, Spark và Python được sử dụng cùng nhau, được gọi là PySpark. Apache Spark là một khung công tác điện toán cụm nguồn mở để xử lý thời gian thực được phát triển bởi Quỹ phần mềm Apache. Spark cung cấp một giao diện để lập trình toàn bộ các cụm với sự song song dữ liệu ngầm và khả năng chịu lỗi.

Một số tính năng của Apache Spark mang lại lợi thế cho các khung công tác khác:

- **Tốc độ:** Nó nhanh hơn 100 lần so với các khung xử lý dữ liệu quy mô lớn truyền thống.
- **Bộ nhớ đệm mạnh mẽ:** Lớp lập trình đơn giản cung cấp khả năng lưu trữ bộ nhớ cache và ổ đĩa mạnh mẽ.
- **Triển khai:** Có thể được triển khai thông qua Mesos, Hadoop thông qua Sợi hoặc trình quản lý cụm riêng của Spark.
- **Thời gian thực:** Tính toán thời gian thực và độ trễ thấp vì tính toán trong bộ nhớ.
- **Polygot:** Đây là một trong những tính năng quan trọng nhất của khung này vì nó có thể được lập trình bằng Scala, Java, Python và R.

Về bộ nhớ, Spark giải quyết các vấn đề vấn đề xung quanh định nghĩa Resilient Distributed Datasets (RDDs). RDDs hỗ trợ hai kiểu thao tác thao tác: transformations và action. Thao tác chuyển đổi(tranformation) tạo ra dataset từ dữ liệu có sẵn. Thao tác actions trả về giá trị cho chương trình điều khiển (driver program) sau khi thực hiện tính toán trên dataset. 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

Quá trình Spark xây dựng DAG: Có hai kiểu chuyển đổi có thể áp dụng trên các RDDs đó là chuyển đổi hẹp và chuyển đổi rộng:

- Chuyển đổi rộng: yêu cầu dữ liệu phải xáo trộn. Ví dụ: reduceByKey(), sortByKey(), groupByKey().
- Chuyển đổi hẹp: không yêu cầu xáo trộn dữ liệ vượt qua các phân vùng (partition). Ví dụ: map(), filter().

2. MapReduce

MapReduce là mô hình được thiết kế độc quyền bởi Google, nó có khả năng lập trình xử lý các tập dữ liệu lớn song song và phân tán thuật toán trên 1 cụm máy tính. MapReduce trở thành một trong những thành ngữ tổng quát hóa trong thời gian gần đây.

MapReduce bao gồm một thủ tục map() và một thủ tục reduce(). Thủ tục map() bao gồm lọc (filter) và phân loại (sort) trên dữ liệu khi thủ tục Reduce() thực hiện quá trình tổng hợp dữ liệu. Đây là mô hình dựa vào các khái niệm biến đối của bản đồ và reduce

những chức năng lập trình theo hướng chức năng. Thư viện của thủ tục Map() và Reduce() sẽ được viết bằng nhiều loại ngôn ngữ khác nhau. Thủ tục được cài đặt miễn phí và được sử dụng phổ biến nhất là là Apache Hadoop.

MapReduce có hai hàm chính là Map() và Reduce(), đây là hai hàm được đỉnh nghĩa bởi người dùng và cũng là hai giai đoạn liên tiếp trong quá trình xử lý dữ liệu của MapReduce. Nhiệm vụ cụ thể của từng hàm:

- **Map():** có nhiệm vụ nhận Input cho các cặp giá trị/ khóa và output chính là tập những cặp giá trị/khóa trung gian. Sau đó, chỉ cần ghi xuống đĩa cứng và tiến hành thông báo cho các hàm Reduce() để trực tiếp nhận dữ liệu.
- **Reduce():** có nhiệm vụ tiếp nhận từ khóa trung gian và những giá trị tương ứng với lượng từ khóa đó. Sau đó, tiến hành ghép chúng lại để có thể tạo thành một tập khóa khác nhau.
- Ở giữa Map() và Reduce() còn một bước trung gian là Shuffle, bước này có nhiệm vụ thu thập cũng như tổng hợp từ khóa/giá trị trung gian đã được map sinh ra trước đó rồi chuyển qua cho Reduce tiếp tục xử lý.

Nguyên tắc hoạt động của MapReduce:

- Phân chia dữ liệu cần xử lý thành các thành phần nhỏ.
- Xử lý các vấn đề của thành phần nhỏ đó theo phương thức song song rồi phân tán độc lập.
- Tổng hợp các kết quả vừa thu được.