**Apache spark (Spark)**

Là 1 hệ thống được dùng để xử lý các vấn đề về dữ liệu lớn. Matei Zaharia, cha đẻ của Spark, sử dụng Hadoop từ những ngày đầu. Đến năm 2009 ông viết Apache Spark để giải quyết những bài toán học máy ở đại học UC Berkely vì Hadoop MapReduce hoạt động không hiệu quả cho những bài toán này. Rất sớm sau đó ông nhận ra rằng Spark không chỉ hữu ích cho học máy mà còn cho cả việc xử lý luồng dữ liệu hoàn chỉnh.

**Apache Spark** gồm có 5 thành phần chính : Spark Core, Spark Luồnging, Spark SQL, MLlib và GraphX, trong đó:

* **Spark Core** Spark Core đảm nhận vai trò thực hiện công việc tính toán và xử lý trong bộ nhớ (In-memory computing) đồng thời nó cũng tham chiếu các dữ liệu được lưu trữ tại các hệ thống lưu trữ bên ngoài.
* **Spark SQL** cung cấp một kiểu data abstraction mới (SchemaRDD) nhằm hỗ trợ cho cả kiểu dữ liệu có cấu trúc (structured data) và dữ liệu nửa cấu trúc (semi-structured data – thường là dữ liệu dữ liệu có cấu trúc nhưng không đồng nhất và cấu trúc của dữ liệu phụ thuộc vào chính nội dung của dữ liệu ấy).
* **Spark Luồnging** được sử dụng để thực hiện việc phân tích luồng bằng việc coi các luồng là các mini-batches và thực hiệc kỹ thuật RDD transformation đối với các dữ liệu mini-batches này. Qua đó cho phép các đoạn code được viết cho xử lý batch có thể được tận dụng lại vào trong việc xử lý luồng, làm cho việc phát triển lambda architecture được dễ dàng hơn. Tuy nhiên điều này lại tạo ra độ trễ trong xử lý dữ liệu (độ trễ chính bằng mini-batch duration)
* **MLlib** (Machine Learning Library): là một nền tảng học máy phân tán bên trên Spark do kiến trúc phân tán dựa trên bộ nhớ.
* **GrapX**: Grapx là nền tảng xử lý đồ thị dựa trên Spark. Nó cung cấp các Api để diễn tả các tính toán trong đồ thị bằng cách sử dụng Pregel Api.

# **Những điểm nổi bật của Spark**

* Xử lý dữ liệu theo lô và thời gian thực
* Có thể tích hợp với tất cả các nguồn dữ liệu và định dạng tệp được hỗ trợ bởi cụm Hadoop.
* Hỗ trợ nhiều ngôn ngữ: Java, Scala, Python và R
* Phân tích thời gian thực tức là dữ liệu đến từ các luồng sự kiện thời gian thực với tốc độ hàng triệu sự kiện mỗi giây.

**Mapreduce**

Mapreduce gồm 2 thủ tục là map và reduce. 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.

Trong map() thì bao gồm có lọc (filter) và phân loại (sort) trên dữ liệu khi thủ tục Reduce() thực hiện quá trình tổng hợp dữ liệu.

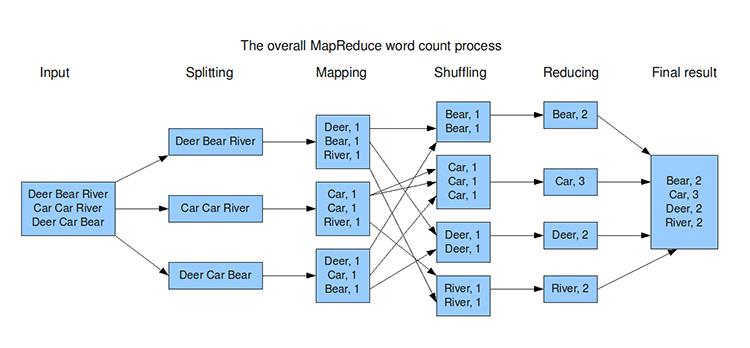
Các thủ tục map() và reduce() được viết bằng nhiều loại ngôn ngữ khác nhau. Được cài đặt miễn phí và sử dụng phổ biên trong Apache hadoop

**Công dụng chính của các hàm là:**

* **Hàm Map():** có nhiệm vụ nhận đầu vào 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.
* **Hàm 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. Các cặp khóa/giá trị này thường sẽ thông qua một con trỏ vị trí để đưa vào các hàm reduce.
* Ở giữa Map và Reduce thì còn 1 bước trung gian đó chính là **Shuffle**. Sau khi Map hoàn thành  xong công việc của mình thì Shuffle sẽ làm nhiệm vụ chính là 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 preduce**

* Phân chia các dữ liệu cần xử lý thành nhiều phần nhỏ.
* Xử lý các vấn đề nhỏ theo phương thức song song trên các máy tính rồi phân tán hoạt động độc lập.
* Tiến hành tổng hợp những kết quả thu được để đề ra được kết quả sau cùng.



## Ưu điểm của MapReduce

* MapReduce có khả năng tác vụ phân tích và tính toán phức tạp. Có thể xử lý nhanh chóng cho ra kết quả dễ dàng chỉ trong khoảng thời gian ngắn.
* Mapreduce có khả năng chạy song song trên các máy có sự phân tán khác nhau.
* MapRedue có khả năng thực hiện trên nhiều nguồn ngôn ngữ lập trình khác nhau

**Các bước hoạt động của MapReduce khi lập trình**

* B1: Tiến hành chuẩn bị các dữ liệu đầu vào
* B2: Thực thi hàm Map() để xử lý dữ liệu.
* B3: Sắp xếp và trộn các kết quả thu từ các máy tính phân tán thích hợp nhất.
* B4: Dùng hàm Reduce() để có thể xử lý tiếp các dữ liệu cần thiết.

Vd về code

