# Apache Spark là gì

Apache Spark là một framework dùng cho cluster-computing với mã nguồn mở. Spark cung cấp một giao diện để lập trình toàn bộ các cụm với tính song song dữ liệu ngầm và khả năng chịu lỗi. Ban đầu được phát triển tại Đại học California, AMPLab của Berkeley, cơ sở mã Spark sau đó được tặng cho Apache Software Foundation, tổ chức này đã duy trì nó kể từ đó.

Apache Spark có nền tảng kiến trúc là resilient distributed dataset (RDD), một tập hợp đa mục dữ liệu và chỉ có thể đọc được phân phối trên một cụm máy, được duy trì theo cách có thể chịu lỗi. API khung dữ liệu đã được phát hành dưới dạng trừu tượng trên RDD, tiếp theo là API tập dữ liệu. Trong Spark 1.x, RDD là giao diện lập trình ứng dụng chính (API), nhưng kể từ Spark 2.x, việc sử dụng API tập dữ liệu được khuyến khích mặc dù API RDD không bị phản đối. Công nghệ RDD vẫn làm nền tảng cho Dataset API.

# Những tính năng nổi bật

* “Spark as a Service”: Giao diện REST để quản lí (submit, start, stop, xem trạng thái) spark job, spark context
* Tăng tốc, giảm độ trễ thực thi job xuống mức chỉ tính bằng giây bằng cách tạo sẵn spark context cho các job dùng chung.
* Stop job đang chạy bằng cách stop spark context
* Bỏ bước upload gói jar lúc start job làm cho job được start nhanh hơn.
* Cung cấp hai cơ chế chạy job đồng bộ và bất đồng bộ
* Cho phép cache RDD theo tên , tăng tính chia sẻ và sử dụng lại RDD giữa các job
* Hỗ trợ viết spark job bằng cú pháp SQL
* Dễ dàng tích hợp với các công cụ báo cáo như: Business Intelligence, Analytics, Data Integration Tools

# Thành phần của Apache Spark

Spark Core là nền tảng cho cẩ Spark: cung cấp những chức năng cơ bản nhất của Spark như lập lịch cho các tác vụ, quản lý bộ nhớ, fault recovery, tương tác với các hệ thống lưu trữ, tiếp xúc thông qua giao diện lập trình ứng dụng… Đặc biệt, Spark Core cung cấp API để định nghĩa RDD (Resilient Distributed DataSet) là tập hợp của các item được phân tán trên các node của cluster và có thể được xử lý song song. Khả năng chịu lỗi đạt được bằng cách theo dõi "con kế thừa" của từng RDD (trình tự hoạt động tạo ra nó) để có thể tái tạo lại nó trong trường hợp mất dữ liệu. RDD có thể chứa bất kỳ loại đối tượng Python, .NET, Java hoặc Scala nào.

Bên cạnh phong cách lập trình chức năng theo hướng RDD, Spark cung cấp hai hình thức hạn chế của các biến được chia sẻ: dữ liệu broadcast chỉ đọc tham chiếu cần có sẵn trên tất cả các nút, trong khi bộ tích lũy có thể được sử dụng để lập trình sự tối giảm theo kiểu mệnh lệnh.

Spark có thể chạy trên nhiều loại Cluster Managers như Hadoop YARN, Apache Mesos hoặc trên chính cluster manager được cung cấp bởi Spark được gọi là Standalone Scheduler.

* Spark SQL : Spark SQL là một thành phần nằm bên trên Spark Core đã giới thiệu một phần trừu tượng hóa dữ liệu được gọi là DataFrames, để cung cấp hỗ trợ cho dữ liệu có cấu trúc và bán cấu trúc. Spark SQL cung cấp một ngôn ngữ dành riêng cho miền (DSL) để thao tác DataFrames. Nó cũng cung cấp hỗ trợ ngôn ngữ SQL, với các giao diện dòng lệnh và máy chủ ODBC / JDBC. Mặc dù DataFrames thiếu kiểm tra kiểu thời gian biên dịch được cung cấp bởi các RDD, kể từ Spark 2.0, DataSet được đánh mạnh cũng được Spark SQL hỗ trợ đầy đủ.
* Spark Streaming: Spark Streaming sử dụng khả năng tạo lịch nhanh của Spark Core để thực hiện phân tích luồng. Nó nhập dữ liệu trong các lô nhỏ và thực hiện các phép biến đổi RDD trên các lô dữ liệu nhỏ đó. Thiết kế này cho phép cùng một bộ mã ứng dụng được viết cho phân tích hàng loạt được sử dụng trong phân tích luồng, do đó tạo điều kiện dễ dàng triển khai kiến trúc lambda. Trong Spark 2.x, một công nghệ riêng biệt dựa trên Tập dữ liệu, được gọi là Truyền trực tuyến có cấu trúc, có giao diện cấp cao hơn cũng được cung cấp để hỗ trợ phát trực tuyến.
* Spark MLlib: Spark MLlib là một framework học máy phân tán nằm trên Spark Core, phần lớn là do kiến trúc Spark dựa trên bộ nhớ phân tán, nhanh gấp 9 lần so với việc triển khai dựa trên đĩa được Apache Mahout sử dụng (theo điểm chuẩn được thực hiện bởi các nhà phát triển MLlib so với việc triển khai bình phương nhỏ nhất (ALS) xen kẽ và trước khi bản thân Mahout đạt được giao diện Spark) và quy mô tốt hơn Vowpal Wabbit. MLlib Cung cấp rất nhiều thuật toán của học máy như: classification, regression, clustering, collaborative filtering…

GraphX là thư viện để xử lý đồ thị.

Trong các thư viện mà Spark cung cấp thì có 69% người dùng Spark SQL, 62% sử dụng DataFrames, Spark Streaming và MLlib + GraphX là 58%

# Quản lý bộ nhớ của Apache Spark

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.

# MapReduce là gì?

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 sẽ  bao gồm những thủ tục sau: thủ tục 1 Map() và 1 Reduce(). Thủ tục Map() bao gồm lọc (filter) và phân loại (sort) trên dữ liệu khi thủ tục khi thủ tục Reduce() thực hiện quá trình tổng hợp dữ liệu. Đây là mô hình dựa vào các khái niệm biển 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.

# Các hàm chính của MapReduce là gì?

MapReduce có 2 hàm chính là Map() và Reduce(), đây là 2 hàm đã được định nghĩa bởi người dùng và nó cũng chính là 2 giai đoạn liên tiếp trong quá trình xử lý dữ liệu của MapReduce. Nhiệm vụ cụ thể của từng hàm như sau:

* **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.
* **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. Quá trình này sẽ giúp cho lập trình viên quản lý dễ dàng hơn một lượng danh sách cũng như  phân bổ giá trị sao cho  phù hợp nhất với bộ nhớ hệ thống.
* 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.

## Các ưu điểm nổi bật của MapReduce

Mapreduce được ưa chuộng sử dụng như vậy bởi nó sở hữu nhiều ưu điểm vượt trội như sau:

* MapReduce có khả năng xử lý dễ dàng mọi bài toán có lượng dữ liệu lớn nhờ khả năng tác vụ phân tích và tính toán phức tạp. Nó 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. Với khả năng hoạt động độc lập kết hợp  phân tán, xử lý các lỗi kỹ thuật để mang lại nhiều hiệu quả cho toàn hệ thống.
* MapRedue có khả năng thực hiện trên nhiều nguồn ngôn ngữ lập trình khác nhau như: [Java,](https://blog.itnavi.com.vn/java-la-gi-nhung-dieu-can-biet-truoc-khi-lap-trinh-java/) [C/ C++](https://blog.itnavi.com.vn/ngon-ngu-lap-trinh-c-va-c/), [Python](https://blog.itnavi.com.vn/python-la-gi-nhung-dieu-ban-can-biet-de-hoc-ngon-ngu-python/), Perl, Ruby,… tương ứng với nó là những thư viện hỗ trợ.
* Như bạn đã biết, mã độc trên internet ngày càng nhiều hơn nên việc xử lý những đoạn mã độc này cũng trở nên rất phức tạp và tốn kém nhiều thời gian. Chính vì vậy, các ứng dụng MapReduce dần hướng đến quan tâm nhiều hơn cho việc phát hiện các mã độc để có thể xử lý chúng. Nhờ vậy, hệ thống mới có thể vận hành trơn tru và được bảo mật nhất.