1. **Giới thiệu về Spark và MapReduce**
   1. **Spark**
      1. Apache Spark là "một công cụ phân tích thống nhất để xử lý dữ liệu quy mô lớn." Spark được duy trì bởi tổ chức phi lợi nhuận Apache Software Foundation, tổ chức đã phát hành hàng trăm dự án phần mềm mã nguồn mở. Hơn 1200 nhà phát triển đã đóng góp cho Spark kể từ khi dự án bắt đầu.
      2. Spark lần đầu tiên được phát hành dưới dạng một dự án mã nguồn mở vào năm 2010 và được phát triển tại AMPLab - UC Berkeley. Spark được phát triển dựa trên nền tảng Hadoop MapReduce, nhằm cải thiện một số hạn chế của MapReduce, chẳng hạn như hiệu suất và tính dễ sử dụng, trong khi vẫn duy trì nhiều ưu điểm của MapReduce.
      3. Spark bao gồm một công cụ xử lý dữ liệu cốt lõi, cũng như các thư viện cho SQL, học máy và xử lý luồng. Do cung cấp nhiều công cụ cho việc xử lý các API cho Java, Scala, Python và R, Spark nhận được sự thu hút rộng rãi của các nhà phát triển - khiến nó trở thành công cụ hữu dụng dùng để xử lý dữ liệu lớn.
   2. **MapReduce**
      1. Hadoop MapReduce là “một khung phần mềm giúp dễ dàng viết các ứng dụng xử lý lượng lớn dữ liệu (tập dữ liệu nhiều terabyte) song song trên các cụm lớn (hàng nghìn nút) của phần cứng hàng hóa theo cách thức đáng tin cậy, có khả năng chịu lỗi. "
      2. Mô hình MapReduce bao gồm hai nhiệm vụ tuần tự: Bản đồ và Rút gọn (do đó có tên). Bản đồ lọc và sắp xếp dữ liệu trong khi chuyển đổi nó thành các cặp khóa-giá trị. Sau đó, Reduce lấy đầu vào này và giảm kích thước của nó bằng cách thực hiện một số loại hoạt động tóm tắt trên tập dữ liệu.
      3. MapReduce có thể tăng tốc đáng kể các tác vụ dữ liệu lớn bằng cách chia nhỏ các tập dữ liệu lớn và xử lý chúng song song. Mô hình MapReduce lần đầu tiên được đề xuất vào năm 2004 bởi các nhân viên của Google là Jeff Dean và Sanjay Ghemawat; sau đó nó được kết hợp vào khuôn khổ Hadoop của Apache để xử lý phân tán.
2. **Sự khác biệt giữa Spark và MapReduce**

Sự khác biệt chính giữa Spark và MapReduce nằm ở cách tiếp cận xử lý: Spark có thể thực hiện việc đó trong bộ nhớ, trong khi MapReduce phải đọc và ghi vào đĩa. Do đó, tốc độ xử lý có sự khác biệt đáng kể (tốc độ xử lý của Spark có thể nhanh hơn 100 lần so với MapReduce). Tuy nhiên, khối lượng dữ liệu được xử lý cũng khác nhau: Hadoop MapReduce có thể làm việc với các tập dữ liệu lớn hơn nhiều so với Spark. Vì vậy mà chúng được sử dụng vào những mục đích phù hợp.

* 1. **Hadoop MapReduce phù hợp:** 
     1. Xử lý tuyến tính các tập dữ liệu khổng lồ: Hadoop MapReduce cho phép xử lý song song lượng dữ liệu khổng lồ. Nó chia một đoạn lớn thành các đoạn nhỏ hơn để được xử lý riêng biệt trên các nút dữ liệu khác nhau và tự động tập hợp các kết quả trên nhiều nút để trả về một kết quả duy nhất. Trong trường hợp tập dữ liệu kết quả lớn hơn RAM khả dụng, Hadoop MapReduce có thể hoạt động tốt hơn Spark.
     2. Tiết kiệm chi phí: nếu không quá quan trọng về tốc độ xử lý thì MapReduce là phải pháp hữu hiệu để tiết kiệm chi phí.
  2. **Spark phù hợp:**
     1. Xử lý dữ liệu nhanh chóng: Xử lý trong bộ nhớ làm cho Spark nhanh hơn Hadoop MapReduce - lên đến 100 lần đối với dữ liệu trong RAM và lên đến 10 lần đối với dữ liệu trong bộ nhớ.
     2. Xử lý lặp đi lặp lại: Tập dữ liệu phân tán có khả năng phục hồi (RDD) của Spark cho phép nhiều hoạt động bản đồ trong bộ nhớ, trong khi Hadoop MapReduce phải ghi kết quả tạm thời vào đĩa.
     3. Xử lý dữ liệu thời gian thực
     4. Xử lý đồ thị: Mô hình tính toán của Spark rất tốt cho các phép tính lặp lại điển hình trong xử lý đồ thị. Và Apache Spark có GraphX ​​- một API để tính toán đồ thị.
     5. Machine learning: Spark có MLlib - một thư viện học máy được tích hợp sẵn, trong khi Hadoop cần một bên thứ ba cung cấp

1. **Khả năng tương thích**

Khả năng tương thích của Spark với các loại dữ liệu và nguồn dữ liệu tương tự như của Hadoop MapReduce.Cả Hadoop và Spark đều có thể mở rộng, tuy nhiên MapReduce là lựa chọn tốt cho các doanh nghiệp cần bộ dữ liệu khổng lồ được kiểm soát bởi các hệ thống hàng hóa. Cả hai khung đều tốt theo nghĩa riêng của chúng. Hadoop có hệ thống tệp riêng mà Spark thiếu và Spark cung cấp một cách để phân tích thời gian thực mà Hadoop không sở hữu.

1. **Bảo mật**
   1. **Spark** : Bảo mật của Spark hiện đang ở giai đoạn đầu, chỉ cung cấp hỗ trợ xác thực thông qua xác thực mật khẩu. Tuy nhiên, có thể chạy Spark trên HDFS để tận dụng các ACL HDFS và quyền cấp độ tệp.
   2. **Hadoop MapReduce** : Hadoop MapReduce có các tính năng bảo mật tốt hơn Spark. Hadoop hỗ trợ xác thực Kerberos, đây là một tính năng bảo mật tốt nhưng khó quản lý. Hadoop MapReduce cũng có thể tích hợp với các dự án bảo mật của Hadoop, như Knox Gateway và Sentry. Các nhà cung cấp bên thứ ba cũng cho phép các tổ chức sử dụng Active Directory Kerberos và LDAP để xác thực. Hệ thống tệp phân tán của Hadoop tương thích với danh sách kiểm soát truy cập (ACL) và mô hình cấp phép tệp truyền thống.