Phan Thanh Vinh @vinhphan812 Theo dõi

83 3 1

Đã đăng vào thg 8 27, 2023 2:54 CH - 26 phút đọc

6.3K 1 6

Tìm Hiểu Về Hadoop, HDFS, Hadoop MapReduce (Lý Thuyết)

...

Mở Đầu

Hiện nay, dữ liệu được tạo ra một cách nhanh chóng và liên tục, từ các ứng dụng trực tuyến, thiết bị di động, máy tính cá nhân, các bộ cảm biến và nhiều nguồn khác. Vơi sự gia tăng vượt trội của dữ liệu, các công cụ truyền thống không còn đủ để xử lý nó (**Big Data**). Để giải quyết vấn đề này, **Hadoop** đã được phát triển như một giải pháp xử lý giữ liệu lớn và phân tán.

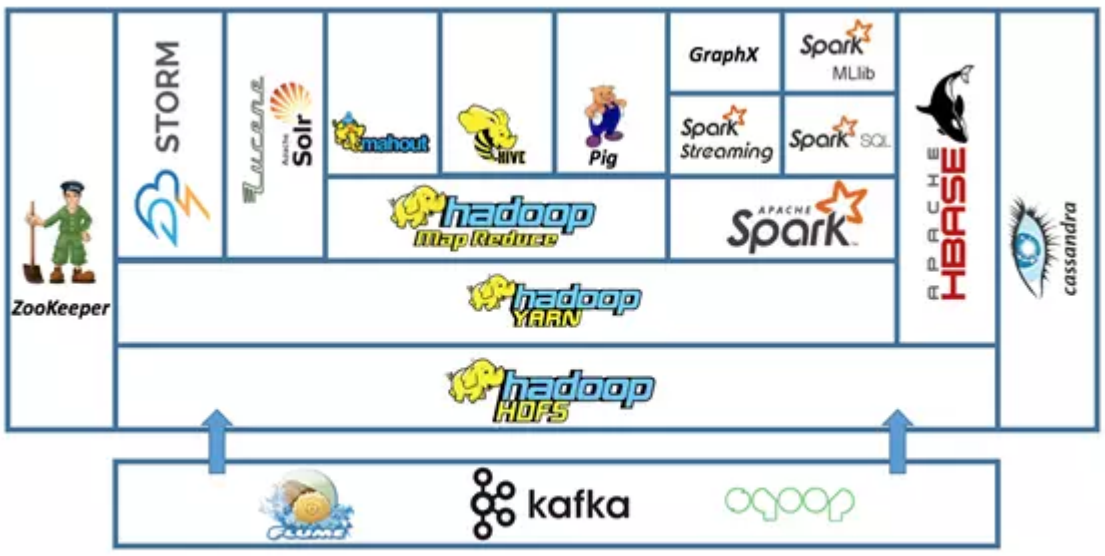
- **Walmart** xử lý hơn 1 triệu giao dịch của khách hàng mỗi giờ, dữ liệu nhập vào ước tính hơn 2,5 PB.
- **Facebook** có hơn 1,9 tỷ người dùng đồng thời, có hàng trăm server xử lý và lưu trữ dữ liệu.
- **Twitter** là hệ thống mạng xã hội với 1,3 tỷ người dùng đang hoạt động và trong giai đoạn đầu.

Vậy dữ liệu lớn là gì? - Dữ liệu lớn hay con gọi là **Big Data** là thuật ngữ mô tả quá trình xử lý dữ liệu trên một tập dữ liệu lớn bao gồm cả dữ liệu *có cấu trúc*, phi cấu trúc hay *bán cấu trúc*. **Big Data** rất quan trọng với các tổ chức, doanh nghiệp thì dữ liệu ngày một lớn và càng nhiều dữ liệu sẽ giúp các phân tích càng chính xác hơn. Việc phân tích chính xác giúp doanh nghiệp đưa ra các quyết định giúp tăng hiệu quả sản xuất, giảm rủi ro và chi phí. Chính vì vậy chúng ta nên sử dụng **Hadoop**.

Bài viết này, sẽ cho các bạn thấy được cái nhìn tổng quan và khái quát hơn về **Hadoop** và một số công cụ hổ trợ, một trong các công nghệ cốt lõi trong xử lý và lưu trữ dữ liệu phân tán.

Hadoop là gì?

Hadoop là một công nghệ phân tán và mã nguồn mở được sử dụng phổ biến để xử lý và lưu trữ khối dữ liệu lớn trên các cụm máy tính phân tán, được thiết kế để xử lý và lưu trữ dữ liệu lớn một cách hiệu quả. **Hadoop** được tạo ra bởi **Doug Cutting** và **Mike Cafarella** và năm *2005*, và được phát triển bởi **Apache Software Foundation** dựa trên công nghệ **Google File System** và **MapReduce**. Cung cấp khả năng xử lý dữ liệu lớn, mở rộng linh hoạt và chi phí thấp, làm cho nó trở thành một công nghệ không thể thiếu trong các hệ thống xử lý dữ liệu hiện đại. Nó được phát triển để giải quyết các thách thức trong lĩnh vực **Big Data** mà các công nghệ cũ không thể đáp ứng.



Hadoop sử dụng mô hình phân tán để lưu trữ và xử lý dữ liệu trên các máy tính thông thường. Thay vì lưu trữ tất cả dữ liệu trên một máy chủ duy nhất, **Hadoop** phân chia dữ liệu thành các khối và lưu chúng trên nhiều máy tính. Điều này giúp tăng khả năng mở rộng, độ tin cậy và hiệu năng. **Hadoop** phân tán dữ liệu trên nhiều nút máy tính và xử lý nó *song song* trên các nút. Điều này giúp giảm thời gian xử lý và tăng hiệu suất. **Hadoop** được sử dụng rộng rãi trong các lĩnh vực như khoa học dữ liệu, trí tuệ nhân tạo, tài chính, ý tế, và nhiều lĩnh vực khác.

Hadoop được xây dựng dựa trên ba phần chính là **Hadoop Distributes FileSystem (HDFS)**, **YARN** và **MapReduce**. Trong đó:

- **HDFS (Hadoop Distributed File System)**: **HDFS** là hệ thống file phân tán được sử dụng để lưu trữ dữ liệu. Nó phân chia dữ liệu thành các khối và phân tán chúng trên nhiều máy tính. **HDFS** có độ tin cậy cao và thích hợp cho việc lưu trữ khối lượng dữ liệu lớn.
- **YARN (Yet Another Resource Negotiator)**: **YARN** là một khung phần mềm quản lý và phân bổ tài nguyên để vận hành các ứng dụng trên **Hadoop**. Nó cung cấp các dịch vụ như quản lý tài nguyên, lên lịch và giám sát cho các ứng dụng chạy trên **Hadoop**.
- **MapReduce**: **MapReduce** là khung mô hình lập trình để xử lý và tính toán trên số lượng dữ liệu lớn trong môi trường phân tán. Nó hoạt động dựa trên 2 pha chính là **Map** và **Reduce**. Pha **Map** sẽ *chia nhỏ dữ liệu* thành các cặp key-value, sau đó pha **Reduce** sẽ *nhóm các cặp* key-value lại với nhau để tính toán kết quả cuối cùng.

Hadoop sử dụng thuật toán **MapReduce** để xử lý dữ liệu một cách song song trên nhiều máy. **MapReduce** chia tác vụ xử lý dữ liệu thành các tác vụ nhỏ hơn (**map**) rồi tổng hợp kết quả (**reduce**). Điều này cho phép xử lý dữ liệu một cách nhanh chóng và hiệu quả.

Hadoop cho phép xử lý dữ liệu theo lô và có khả năng xử lý khối lượng dữ liệu cực lớn. **Hadoop** sử dụng một cụm các máy tính (*server*) thông thường để lưu trữ, tính toán. Việc tính toán này trên **HDFS** được thực hiện một cách song song, đồng thời và trừu tượng với các lập trình viên giúp họ tránh được việc lập trình mạng và xử lý bài toán đồng bộ phức tạp. Không giống như nhiều hệ thống phân tán khác, **Hadoop** cung cấp việc xử lý *logic* trên nơi lưu trữ dữ liệu mà không phải lấy dữ liệu từ các máy khác giúp tăng hiệu năng một cách mạnh mẽ.

Hadoop bao gồm nhiều *module* như:

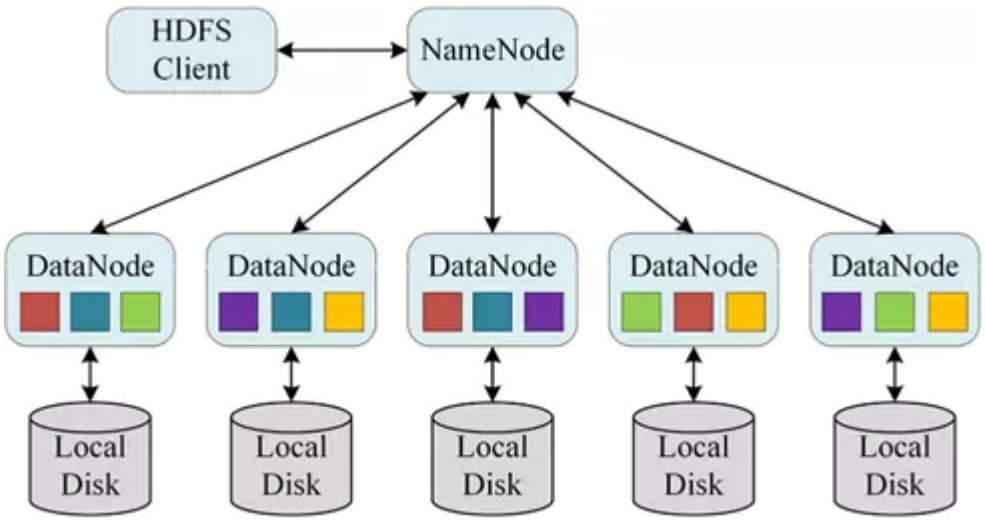
- **Hadoop Common**: các tiện ích cơ bản hỗ trợ **Hadoop**.
- **Hadoop Distributes File System (HDFS)**: Hệ thống file phân tán cung cấp khả năng truy vấn song song tối đa hóa theo đường truyền truy cập bởi ứng dụng.
- **Hadoop YARN**: **Framework** quản lý lập lịch tác vụ và quản lý các tài nguyên trên cụm.
- **Hadoop MapReduce**: Hệ thống **YARN-based** để xử lý tập dữ liệu lớn.

Hadoop đang được sử dụng rộng rãi trong thực tế để giải quyết các bài toán liên quan tới dữ liệu lớn như phân tích dữ liệu khách hàng, mô hình hóa dữ liệu, xử lý log, phân tích mạng xã hội, tìm kiếm web, v.v...

Để triển khai **Hadoop** trên một cụm máy tính phân tán, người dùng cần cài đặt và cấu hình các thành phần của **Hadoop**, bao gồm **HDFS**, **Hadoop MapReduce** và các công nghệ khác trong hệ sinh thái **Hadoop**. Các công cụ hỗ trợ như **Ansible**, **Puppet**, **Docker** và **Kubernetes** được sử dụng để giảm thiểu thời gian và công sức cài đặt **Hadoop**.

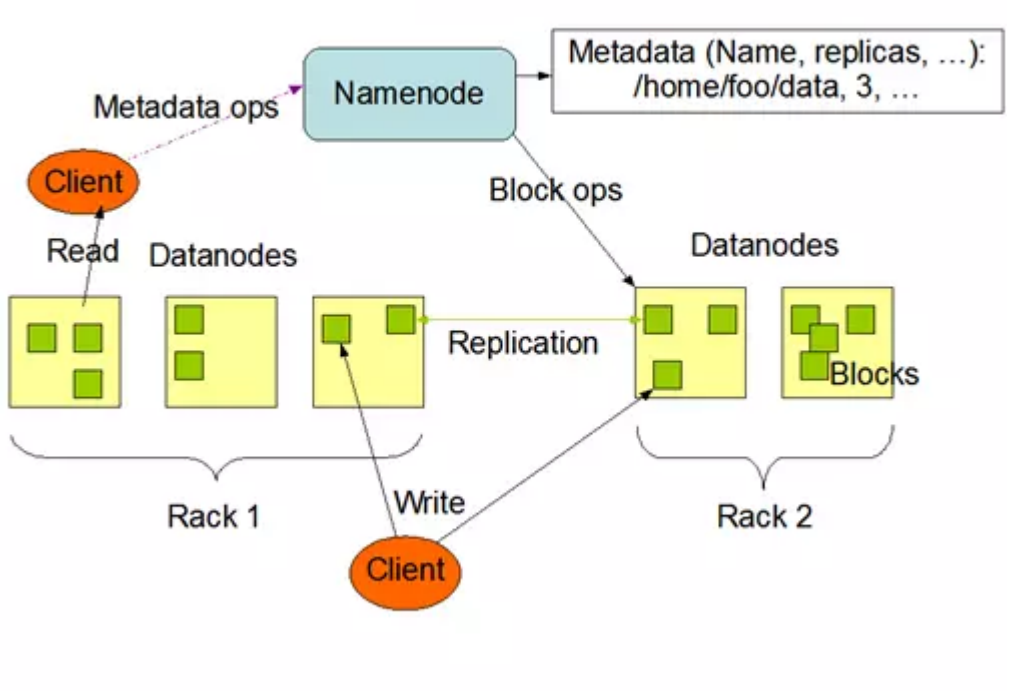
HDFS (Hadoop Distributes File System):

HDFS là từ viết tắt của **Hadoop Distributes File System**, là một hệ thống lưu trữ dữ liệu phân tán được thiết kế chạy trên phần cứng thông thường để lưu trữ các tệp dữ liệu có kích thước lớn trên nhiều nút máy tính trong một mạng **Hadoop**. **HDFS** cũng tương tự như những hệ thống file phân tán khác. Tuy nhiên, sự khác biệt ở đây là **HDFS** có khả năng chịu lỗi cao (**fault-tolerant**) và được thiết kế để *deploy* trên các phần cứng rẻ tiền. HDFS cũng cung cấp khả năng truy cập **high throughput** từ ứng dụng và thích hợp với các ứng dụng có tập dữ liệu lớn. **HDFS** là một phần của nền tảng **Hadoop** và là một phần quan trọng của việc xử lý dữ liệu lớn.

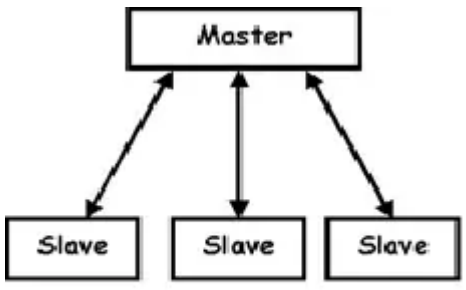


HDFS được thiết kế để chịu được sự cố và có thể xử lý các tệp dữ liệu có kích thước rất lớn. **HDFS** lưu trữ dữ liệu dưới dạng các khối, mỗi khối có kích thước mặc định là **128MB**, nhưng có thể tùy chỉnh kích thước theo nhu cầu sử dụng. Dữ liệu được phân tán trên nhiều nút máy tính trong một mạng **Hadoop**, giúp tăng tốc độ truy xuất dữ liệu và giảm thời gian xử lý.

HDFS Architecture



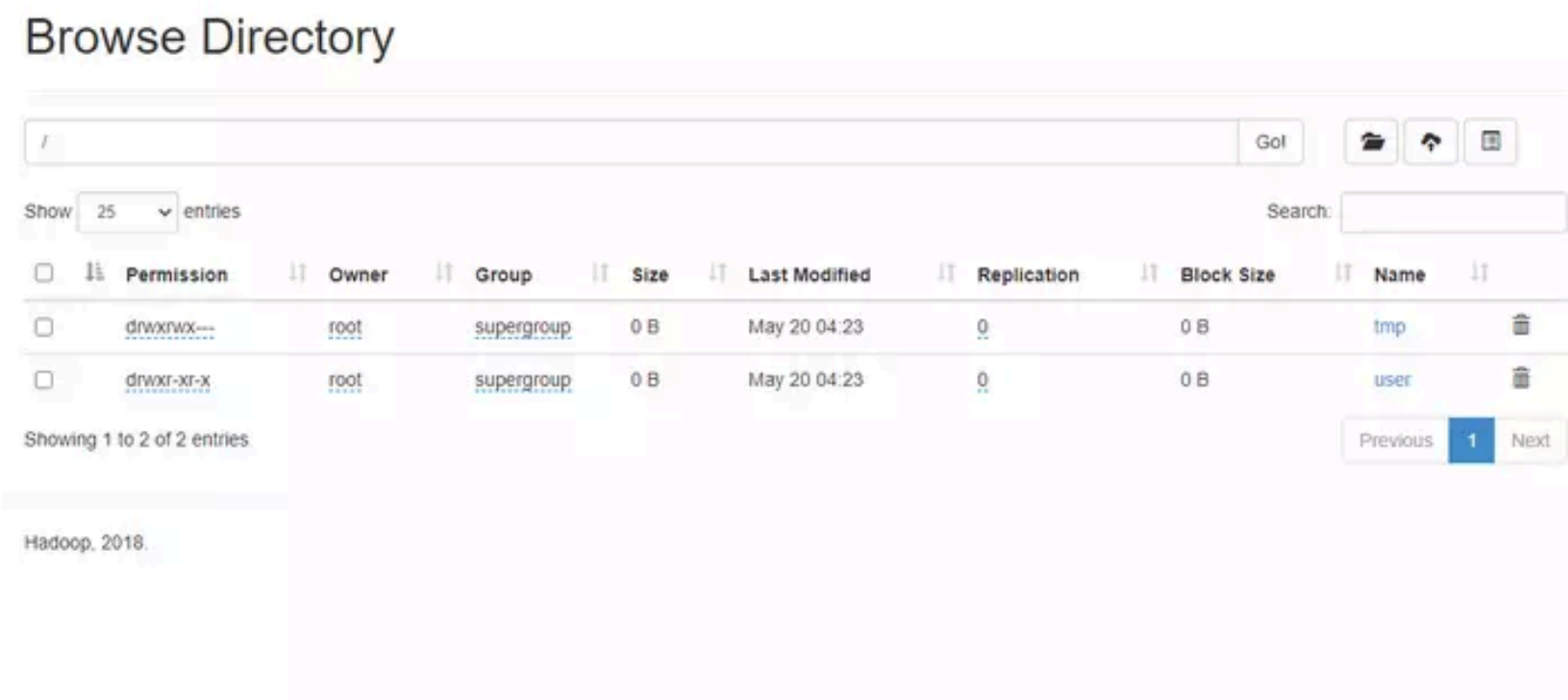
HDFS sử dụng kiến trúc **Master/Slave**, bao gồm hai thành phần chính là **NameNode** và **DataNode**. **NameNode** là trung tâm điều khiển của **HDFS** và lưu trữ thông tin về vị trí và trạng thái các khối dữ liệu. **DataNode** là các nút lưu trữ dữ liệu thực sự và phân tán dữ liệu trên các nút khác nhau.



Một cụm **HDFS** bao gồm hai loại nút (**Node**) hoạt động theo mô hình nút chủ (**Master**) - nút thợ (**Worker**):

- Một cụm **HDFS** có 1 **NameNode** (**Master**).
- Một cụm **HDFS** có một hoặc nhiều các **DataNode** (**Worker**).

NameNode quản lý các **Namespace Filesystem**. Nó quản lý một **Filesystem Tree** và các **metadata** cho tất cả *file* và thư mục trên **tree**. Thông tin này được lưu trữ trên đĩa vật lý dưới dạng không gian tên ảnh và nhật ký (*edit log*). NameNode còn quản lý thông tin các khối (**block**) của một tập tin được lưu trên những **DataNodes** nào.



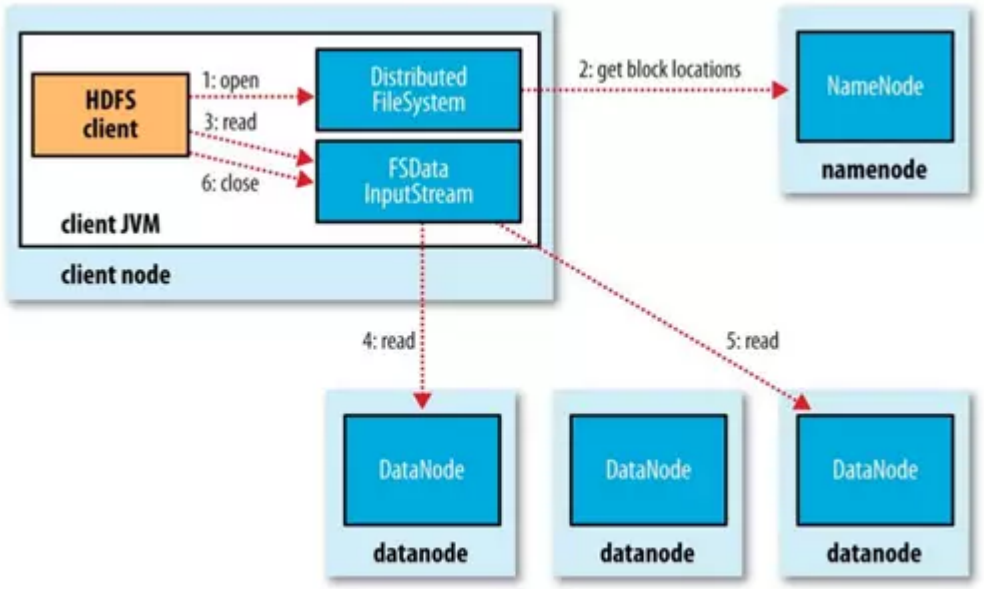
HDFS đưa ra một không gian tên cho phép dữ liệu được lưu trên tập tin. Trong đó một tập tin được chia ra thanh một hay nhiều khối (**block**) và các **block** được lưu trên một tập các **DataNode**. **NameNode** thực thi các hoạt động trên hệ thống quản trị không gian tên tập tin như mở, đóng đổi tên tập tin và thư mục. Các **DataNode** có tính năng xử lý các yêu cầu về đọc ghi từ máy khách. Ngoài ra các **DataNode** còn thực hiện việc tạo, xóa, lặp các khối theo sự hướng dẫn của **DataNode**.

HDFS đưa ra một không gian tên cho phép dữ liệu được lưu trên tập tin. Trong đó một tập tin được chia ra thanh một hay nhiều khối (**block**) và các block được lưu trên một tập các **DataNode**. **NameNode** thực thi các hoạt động trên hệ thống quản trị không gian tên tập tin như mở, đóng đổi tên tập tin và thư mục. Các **DataNode** có tính năng xử lý các yêu cầu về đọc ghi từ máy khách. Ngoài ra các **DataNode** còn thực hiện việc tạo, xóa, lặp các khối theo sự hướng dẫn của **DataNode**.

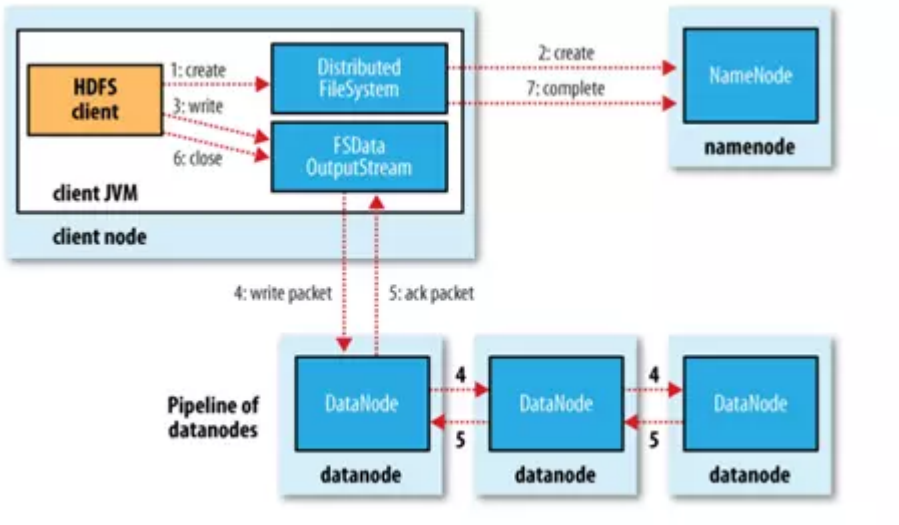
HDFS cũng cung cấp các cơ chế tối ưu hóa hiệu suất để cải thiện hiệu suất của hệ thống. Nó cung cấp các tính năng như tối ưu hóa băng thông, cơ chế lưu trữ đa cấp, cơ chế lưu trữ đệm, đồng bộ hóa dữ liệu và các cơ chế lưu trữ lớn. **HDFS** cũng cung cấp các cơ chế quản lý bảo mật, bảo vệ dữ liệu và các cơ chế phân tán để bảo vệ dữ liệu trước các cuộc tấn công bên ngoài. Ngoài ra, **HDFS** cũng cung cấp các cơ chế để tối ưu hóa và bảo trì hệ thống, bao gồm các cơ chế như tự động phân phối dữ liệu, cấu trúc dữ liệu và các cơ chế lưu trữ lớn.

HDFS cung cấp các tính năng như sao lưu dữ liệu, phục hồi dữ liệu, xử lý song song và truy vấn dữ liệu từ xa. **HDFS** cũng cung cấp các API để cho các ứng dụng khác truy cập vào xử lý dữ liệu trên **HDFS**. **HDFS** cũng cung cấp các cơ chế để quản lý dữ liệu, bao gồm các cơ chế như tự động phân phối dữ liệu, cấu trúc dữ liệu và các cơ chế lưu trữ lớn.

- Đọc dữ liệu trên **HDFS**:
    - Với khối dữ liệu (**block**) *ID* và địa chỉ *IP* đích máy chủ (*host*) của **Datanode**, máy khách (*client*) có thể liên lạc với các **DataNode** còn lại để đọc các khối (**block**) cần thiết. Quá trình này lặp lại cho đến khi tất cả các khối trong *file* được đọc và máy khách đóng luồng đọc *file* trực tuyến.



- Ghi dữ liệu trên **HDFS**:
    - Việc ghi dữ liệu sẽ phức tạp hơn việc đọc dữ liệu đối với hệ thống **HDFS**. Trong bên dưới, ban đầu, máy khách gửi yêu cầu đến tạo một file bằng việc sử dụng **Hadoop FileSystem APIs**. Một yêu cầu được gửi đến **NameNode** để tạo tập tin *metadata* nếu *user* có quyền tạo. Thông tin *metadata* cho tập tin mới đã được tạo; tuy nhiên lúc này chưa có một **block** nào liên kết với tập tin này. Một tiến trình trả về kết quả được gửi lại cho máy khách xác nhận yêu cầu tạo *file* đã hoàn thành và bắt dầu có thể ghi dữ liệu. Ở mức **API**, một đối tượng **JAVA** là *stream* sẽ trả về. Dữ liệu của máy khách sẽ ghi vào luồng này và được chia ra thành các gói, lưu trong *queue* của bộ nhớ. Một tiến trình riêng biệt sẽ liên hệ với **NameNode** để yêu câu một tập **DataNode** phục vụ cho việc sao lưu dữ liệu vào các khối (**block**). Máy khách sẽ tạo ra một kết nối trực tiếp đến **DataNode** đầu tiên trong danh sách. **DataNode** đầu tiên đó sẽ kết nối lần lượt đến các **DataNode** khác. Các gói dữ liệu được ghi dần vào các **DataNode**. Mỗi **DataNode** sẽ phản hồi dữ liệu ghi thành công hay không. Quá trình này kết thúc khi toàn bộ các gói dữ liệu đã được lưu tại các khối (**block**) của **DataNode**.
    - Tuy nhiên, **HDFS** cũng có một số hạn chế. Vì mỗi tệp dữ liệu được chia thành các khối dữ liệu và lưu trữ trên các nút khác nhau, việc truy xuất dữ liệu trở nên chậm hơn so với việc truy xuất dữ trên đĩa cục bộ. Hơn nữa, **HDFS** không phù hợp cho các tác vụ yêu cầu truy xuất một phần của tệp dữ liệu, vì việc truy xuất phải được thực hiện trên toàn bộ khối dữ liệu.

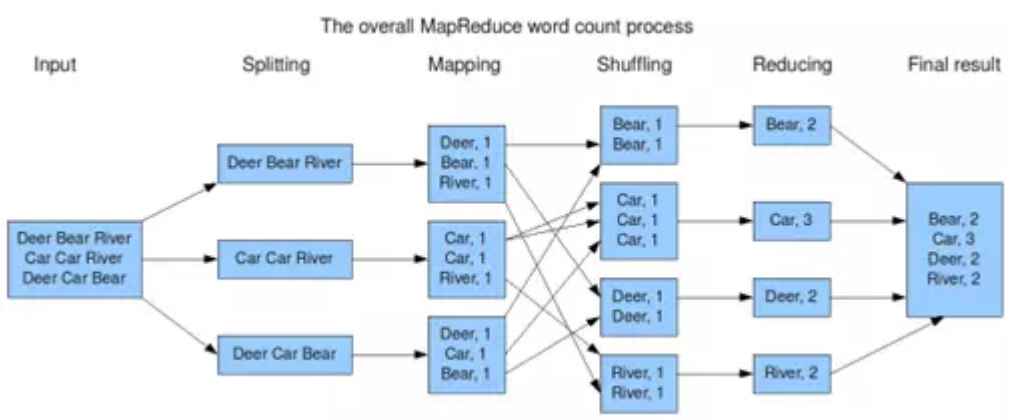


Tóm lại, **Hadoop Distributes FileSystem (HDFS)** là một hệ thống lưu trữ dữ liệu phân tán được thiết kế lưu trữ và quản lý các tệp dữ liệu lớn trên các cụm máy tính phân tán trong một mạng **Hadoop**. Nó sử dụng kiến trúc **Master/Slave**, Với các tính năng như sao lưu dữ liệu, phục hồi dữ liệu, xử lý song song và truy cập dữ liệu từ xa, **HDFS** là một phần quan trọng của việc xử lý dữ liệu của **Hadoop**.

MapReduce

MapReduce là một khung làm việc xử lý dữ liệu phân tán được sử dụng để xử lý dữ liệu lớn trên **Hadoop**. **MapReduce** được phát triển bởi **Google** và sau đó được **Apache Software Foundation** phát triển và phát hành dưới dạng một phần của hệ sinh thái **Hadoop**.

MapReduce thực hiện xử lý dữ liệu bằng cách phân tách dữ liệu thành các phần nhỏ hơn và xử lý chúng song song trên các nút máy tính khác nhau trong cùng một mạng **Hadoop**. Khung làm việc **MapReduce** được thiết kế để hoạt động trên các phần dữ liệu độc lập, do đó, các phần dữ liệu có thể được xử lý độc lập và đồng thời tăng tốc độ xử lý.



MapReduce bao gồm hai pha chính là pha **Map** và pha **Reduce**. Trong pha **Map**, dữ liệu được xử lý và phân tích bằng các hàm **Map** để tạo ra các cặp *key-value*. *Key-value* này sau đó được chuyển đến pha **Reduce** để được tổng hợp và xử lý tiếp theo. Trong pha **Reduce**, các cặp *key-value* được tổng hợp và xử lý bằng các hàm **Reduce** để tạo ra kết quả cuối cùng.

Trong quá trình "**Map**", các giá trị đầu vào được chia thành các phần nhỏ hơn và được xử lý song song trên các nút trong cụm máy tính phân tán. Mỗi nút xử lý một phần nhỏ của dữ liệu đầu vào và tạo ra các cặp *key-value*. Sau khi các cặp *key-value* được tạo ra, chúng được sắp xếp và gom nhóm lại theo khóa và truyền đến các tác vụ "**Reduce**".

Trong quá trình "**Reduce**", các cặp *key-value* được xử lý để tạo ra kết quả cuối cùng. Tác vụ "**Reduce**" sẽ nhận các cặp *key-value* được gom nhóm theo khóa và xử lý chúng để tạo ra các kết quả cuối cùng. Kết quả này có thể được lưu trữ lại hoặc truyền đến các ứng dụng khác để tiếp tục xử lý.

MapReduce có thể xử lý nhiều loại dữ liệu khác nhau, bao gồm văn bản, hình ảnh, âm thanh và video. Nó cũng hỗ trợ các tính năng xử lý lỗi, sao lưu và khôi phục dữ liệu.

MapReduce là một công nghệ quan trọng trong việc xử lý dữ liệu lớn và được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau.

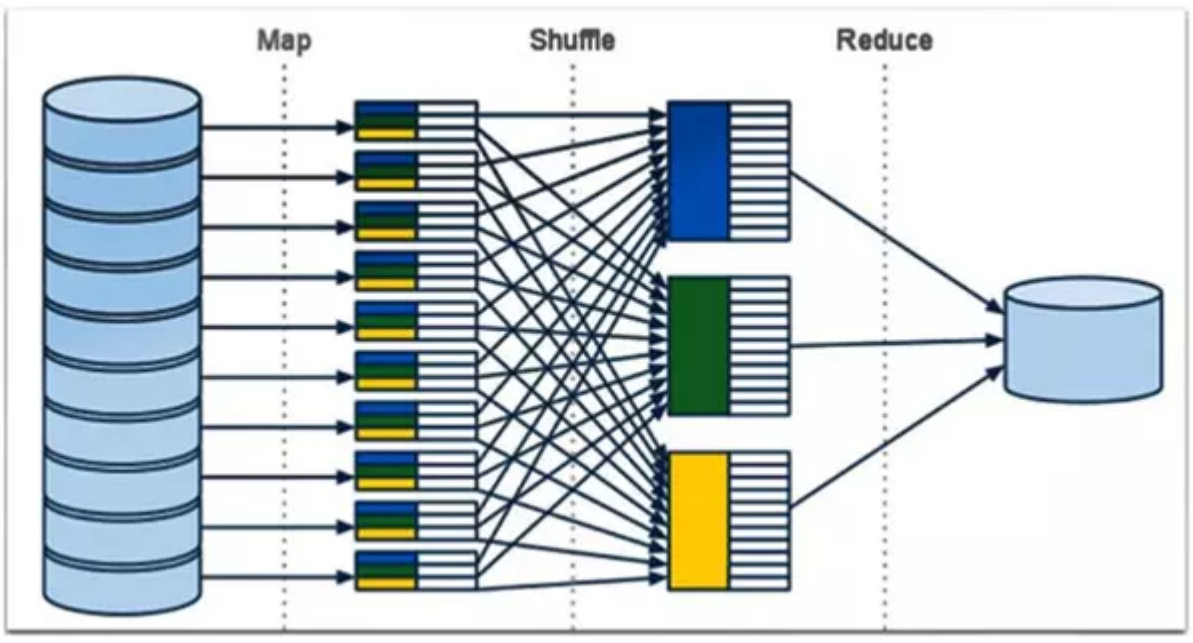
MapReduce đã trở thành một phần quan trọng của nền tảng **Hadoop** và được sử dụng rộng rãi trong các dự án xử lý dữ liệu lớn và phân tích dữ liệu trên thế giới.

MapReduce cung cấp khả năng mở rộng linh hoạt và độ tin cậy cao, làm cho nó trở thành một công nghệ không thể thiếu trong các hệ thống xử lý dữ liệu lớn. Nó cho phép xử lý dữ liệu lớn trên các cụm máy tính phân tán và có thể mở rộng lên đến hàng nghìn nút. Nó cũng cung cấp tính năng phân tán và song song, giúp tăng tốc độ xử lý dữ liệu và giảm thời gian xử lý.

Tuy nhiên, **MapReduce** cũng có một số hạn chế. Việc xử lý dữ liệu trong **MapReduce** có thể trở nên chậm nếu dữ liệu không được chia đều giữa các nút trong cụm máy tính phân tán. Ngoài ra, MapReduce cũng không phù hợp cho các tác vụ xử lý dữ liệu phức tạp và phân tích thời gian thực.

MapReduce là một kỹ thuật xử lý dữ liệu phân tán được sử dụng trong **Hadoop** để phân tích và xử lý các tập dữ liệu lớn. Kỹ thuật này hoạt động bằng cách phân tách các tác vụ xử lý thành các công việc nhỏ hơn được thực hiện trên các nút trong một cụm **Hadoop**. Dưới đây là một số thông tin về **MapReduce** và cách nó hoạt động trong **Hadoop**.

1. Các bước của **MapReduce**:
    - Bước **Map**: Dữ liệu vào được chia thành các phần nhỏ hơn và được xử lý độc lập trên từng nút trong cụm **Hadoop**.
    - Bước **Shuffle**: Dữ liệu đầu ra từ bước **Map** được sắp xếp và gom nhóm để chuẩn bị cho bước **Reduce**.
    - Bước **Reduce**: Dữ liệu được xử lý lại trên các nút trong cụm Hadoop và kết quả cuối cùng được trả về.



2. Lợi ích của **MapReduce**:
    - Có thể xử lý các tập dữ liệu lớn và phức tạp trên nhiều máy chủ trong một cụm.
    - Đạt được hiệu suất và tốc độ xử lý cao bằng cách phân tách các tác vụ xử lý thành các công việc nhỏ hơn và thực hiện trên nhiều máy chủ đồng thời.
    - Đảm bảo tính độc lập và bất biến của các công việc xử lý, do đó giảm thiểu tác động của lỗi trong quá trình xử lý dữ liệu.
    - Có thể mở rộng hệ thống để xử lý tập dữ liệu lớn hơn hoặc tăng tốc độ xử lý bằng cách thêm nhiều máy chủ vào cụm Hadoop.

3. Cách sử dụng **MapReduce** trong **Hadoop**:

    - Viết mã **Java** hoặc các ngôn ngữ lập trình khác để thực hiện các bước **MapReduce**.
    - Sử dụng các công cụ và kỹ thuật liên quan đến **Hadoop** để thiết lập và quản lý một cụm **Hadoop** để xử lý dữ liệu.
    - Sử dụng các công cụ và kỹ thuật phân tích dữ liệu, chẳng hạn như **Hive**, **Pig**, **Spark**, ... để thực hiện các tác vụ phân tích dữ liệu và truy vấn dữ liệu trên nền tảng **Hadoop**.

- Trong quá trình sử dụng **MapReduce**, người dùng cần phải nắm được các kiến thức liên quan đến lập trình và xử lý dữ liệu trên **Hadoop**, cũng như hiểu rõ về cấu trúc và hoạt động của một cụm **Hadoop**. Ngoài ra, cần phải tối ưu hóa các tác vụ **MapReduce** để đạt được hiệu suất tốt nhất trong quá trình xử lý dữ liệu. Việc tìm hiểu và áp dụng **MapReduce** trong **Hadoop** sẽ giúp người dùng có thể xử lý các tập dữ liệu lớn và phức tạp một cách hiệu quả và đáng tin cậy.
- Tóm lại, **MapReduce** là một khung làm việc xử lý dữ liệu phân tán được sử dụng để xử lý dữ liệu lớn trên **Hadoop**. Với khả năng xử lý dữ liệu độc lập và song song trên nhiều máy tính khác nhau, **MapReduce** là một công nghệ quan trọng trong việc xử lý dữ liệu lớn và được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau. .

Apache Hadoop | Big Data

Bài viết liên quan

Chương 10: SORTING - 1.Lý thuyết cơ bản
Đặng An
20 phút đọc
693 3 0 5

Giải pháp Big Data - Hadoop
Nguyen Thi Thanh Ly
5 phút đọc
5.2K 2 0 1

Chương 11: SEARCHING - Lý thuy...
Đặng An
11 phút đọc
232 2 0 1

Một số lý thuyết khác trong Swift
Bui Manh Tri
8 phút đọc
899 1 0 0

Chương 10: SORTING - 1.Lý thuyết cơ bản
Đặng An
20 phút đọc
693 3 0 5

Giải pháp Big Data - Hadoop
Nguyen Thi Thanh Ly
5 phút đọc
5.2K 2 0 1

Chương 11: SEARCHING - Lý thuyết cơ bản
Đặng An
11 phút đọc
232 2 0 1