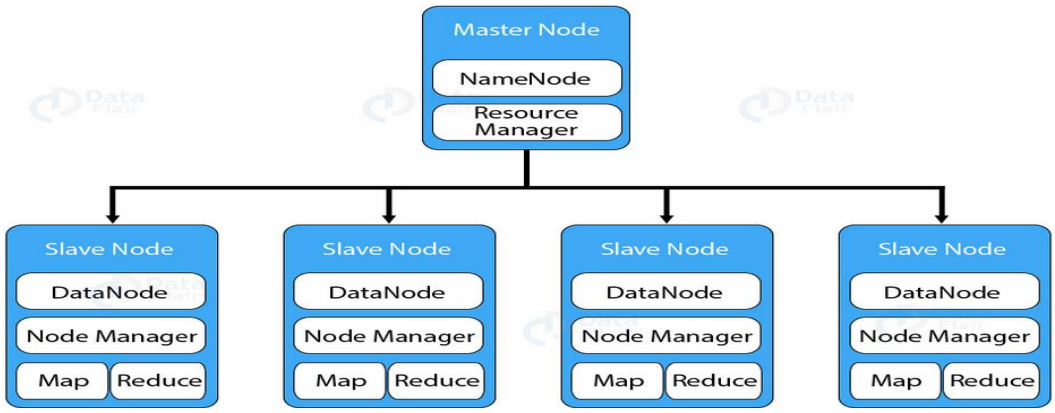
I. Kiến trúc của hadoop

❖ **HDFS(Hadoop Distributed File System)**



(a) Hệ thống tệp phân tán tuân theo **Master - Slave.** Mỗi cụm bao gồm 1 nút chính duy nhất và các nút phụ .Bên trong các tệp được chia thành một hoặc nhiều khối và mỗi khối được lưu trữ trên các máy phụ khác nhau .

– Nút **Master** lưu trữ và quản lý không gian tên hệ thống tệp, đó là thông tin về các khối tệp như vị trí khối, quyền, v.v. Các **Slave** lưu trữ các khối dữ liệu của tệp.

– Nút chính là NameNode và DataNodes là các nút phụ.

– NameNode là trung tâm của Hệ thống tệp phân tán Hadoop. Nó duy trì và quản lý không gian tên hệ thống tệp và cung cấp quyền truy cập phù hợp cho các máy khách. Nó lưu trữ dưới dạng 2 tệp : Fsimage và Edit log.

– **Chức năng của NameNode** :

+ Quản lý duy trì datanode.
+ Thực hiện các hoạt động của không gian tên hệ thống tệp như mở, đổi tên và đóng các tệp và thư mục.
+ Ghi lại từng thay đổi được thực hiện đối với không gian tên hệ thống tệp.
+ DataNode không thành công, NameNode sẽ chọn DataNodes mới cho các bản sao mới.

– Data node là các nút chia nhỏ theo nhánh trong HDFS hadoop

– **Chức năng của Datanode**:

+ Chịu trách nhiệm phục vụ các yêu cầu đọc / ghi của máy khách
+ DataNodes thực hiện tạo khối, sao chép và xóa
+ DataNodes gửi một nhịp tim đến NameNode để báo cáo tình trạng của HDFS
+ Gửi báo cáo khối đến NameNode để báo cáo danh sách các khối mà nó chứa.

(b) Block in HDFS: Bên trong, HDFS chia tệp thành các phần có kích thước khối được gọi là khối. Kích thước của khối là 128 Mb theo mặc định

– Tệp ở HDFS, nhỏ hơn một khối đơn lẻ không chiếm dung lượng kích thước khối đầy đủ của bộ nhớ bên dưới.Mỗi tệp được lưu trữ trong HDFS không cần phải là bội số chính xác của kích thước khối đã định cấu hình.

– Ưu điểm : + Không giới hạn về kích thước tệp ,tính đơn giản của hệ thống con lưu trữ

+ Phù hợp tốt với nhân rộng để cung cấp Khả năng chịu lỗi và Tính sẵn sàng cao
+ Loại bỏ các mối quan tâm về siêu dữ liệu

(c) Replication Management :

– Đối với hệ thống phân tán, dữ liệu phải được dự phòng ở nhiều nơi để nếu một máy bị lỗi, dữ liệu có thể truy cập từ các máy khác.

– HDFS lưu trữ bản sao của một khối trên nhiều Mã dữ liệu dựa trên yếu tố sao chép.

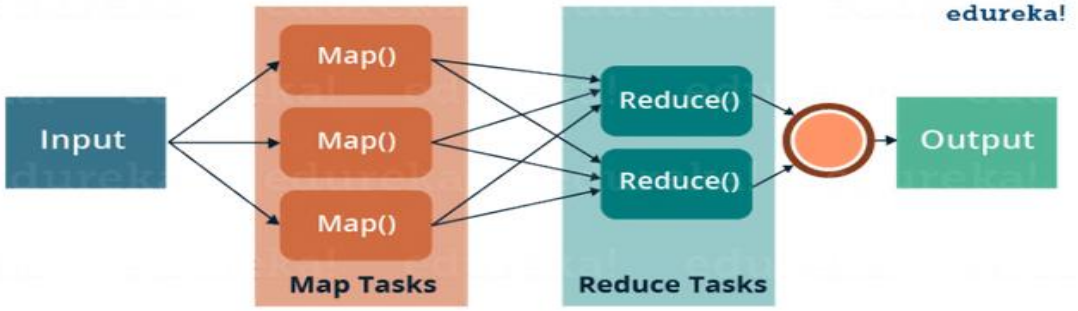
– Cơ chế sao chép này làm cho HDFS có khả năng chịu lỗi.

(d) Rack Awareness

- **Rack** là tập hợp của khoảng 40-50 máy được kết nối bằng cách sử dụng cùng một Network switch. Nếu network gặp sự cố, toàn bộ Racks sẽ không khả dụng.
- Trong một cụm Hadoop lớn, có nhiều racks. Mỗi rack bao gồm các DataNodes. Giao tiếp giữa các DataNodes trên cùng rack hiệu quả hơn so với giao tiếp giữa các DataNodes nằm trên racks khác nhau.
- Để giảm lưu lượng network trong quá trình tệp đọc/ghi, NameNode chọn DataNode gần nhất để phục vụ yêu cầu đọc / ghi của máy khách. NameNode duy trì **id rack** của mỗi DataNode để đạt được thông tin rack này.
- Ưu điểm : Ngăn ngừ mất dữ liệu khi giá đỡ bị lỗi, giảm thiểu chi phí ghivà tối đa hóa tốc độ đọc.Tối đa hóa băng thông và độ trễ thấp

❖ **Mapreduce**

- Mapreduce là lớp xử lý của Hadoop . Mô hình lập trình MapReduce được thiết kế để xử lý song song khối lượng lớn dữ liệu bằng cách chia công việc thành một tập hợp các nhiệm vụ độc lập.
- Luồng dữ liệu và cách hoạt động Mapreduce

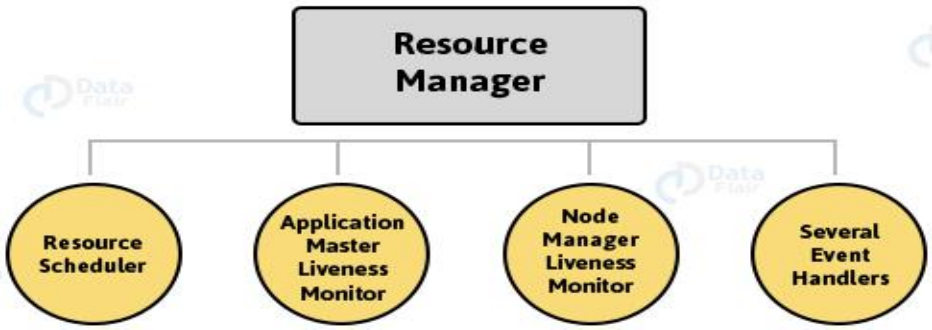


+ Input files: Dữ liệu cho một tác vụ MapReduce được lưu trữ trong các tệp đầu vào và các tệp đầu vào thường nằm trong **HDFS** . Định dạng của các tệp này là tùy ý, trong khi các tệp nhật ký dựa trên dòng và định dạng nhị phân cũng có thể được sử dụng.
+ InputFormat: Xác định cách các tệp đầu vào này được phân chia và đọc. Nó chọn các tệp hoặc các đối tượng khác được sử dụng để nhập liệu. InputFormat tạo InputSplit.
+ InputSplits: Nó được tạo bởi InputFormat, đại diện một cách logic cho dữ liệu sẽ được xử lý bởi một riêng lẻ Mapper**.** Một nhiệm vụ Mapper được tạo cho mỗi lần tách; do đó số lượng nhiệm vụ Mapper sẽ bằng số lượng InputSplits.
+ RecordReader : Chuyển đổi dữ liệu thành các cặp key-value phù hợp để đọc. Theo mặc định, nó sử dụng TextInputFormat để chuyển đổi dữ liệu thành một cặp khóa-giá trị. RecordReader giao tiếp với InputSplit cho đến khi việc đọc tệp không được hoàn thành.
+ Mapper: Nó xử lý từng bản ghi đầu vào (từ RecordReader) và tạo ra cặp key-value mới và cặp key-value được tạo bởi Mapper này hoàn toàn khác với cặp đầu vào. Đầu ra của Mapper còn được gọi là đầu ra trung gian được ghi vào đĩa cục bộ. Đầu ra của Mapper không được lưu trữ trên HDFS vì đây là dữ liệu tạm thời và việc ghi trên HDFS sẽ tạo ra các bản sao không cần thiết.
+ Combiner: Hadoop MapReduce Combiner thực hiện tổng hợp cục bộ trên đầu ra của người lập map, giúp giảm thiểu việc truyền dữ liệu giữa trình ánh xạ. Khi chức năng bộ kết hợp được thực thi, kết quả đầu ra sau đó sẽ được chuyển cho trình phân vùng để làm việc tiếp theo.
+ Partitioner: Trình phân vùng lấy đầu ra từ các bộ kết hợp và thực hiện phân vùng. Phân vùng đầu ra diễn ra trên cơ sở khóa và sau đó được sắp xếp.

+ Shuffling and Sorting : Khi tất cả các trình ánh xạ đã hoàn thành và đầu ra của chúng được xáo trộn trên các nút giảm tốc, thì đầu ra trung gian này được hợp nhất và sắp xếp, sau đó được cung cấp làm đầu vào để giảm giai đoạn.
+ Reduce: Nó lấy tập hợp các cặp key-value trung gian do người lập map tạo ra làm đầu vào và sau đó chạy một hàm giảm thiểu trên mỗi cặp để tạo ra đầu ra. Đầu ra của bộ giảm tốc là đầu ra cuối cùng, được lưu trữ trong HDFS.
+ RecordWriter: Nó ghi cặp key-value đầu ra này từ pha Giảm tốc vào các tệp đầu ra
+ Output Format: Cách các cặp khóa-giá trị đầu ra này được ghi trong tệp đầu ra bởi RecordWriter được xác định bởi OutputFormat. Đầu ra cuối cùng của bộ giảm tốc được ghi trên HDFS bởi các cá thể OutputFormat

❖ **Yarn(Yet Another Resource Locator)**

1) Resource manager



– Resource manager chạy trên nút chính.
– Nó biết vị trí của nút ở đâu (Rack Awareness). Nó biết mỗi nút có bao nhiêu tài nguyên.
– Resource Scheduler là một trong những dịch vụ quan trọng được chạy bởi Resource Manager.
– Trình lập lịch tài nguyên quyết định cách các tài nguyên được giao cho các nhiệm vụ khác nhau.
– Application Manager là một dịch vụ nữa do Resource Manager điều hành.
– Application Manager dụng thương lượng vùng chứa đầu tiên cho một ứng dụng.
– Resource manager theo dõi thông tin từ Node manager
– ResourceManger có hai thành phần quan trọng đó là Scheduler và ApplicationManager
+ **Scheduler** : có trách nhiệm phân bổ tài nguyên cho các ứng dụng khác nhau. Đây là Scheduler thuần túy vì nó không thực hiện theo dõi trạng thái cho ứng dụng. Nó cũng không sắp xếp lại các tác vụ bị lỗi do lỗi phần cứng hoặc phần mềm. Bộ lập lịch phân bổ các tài nguyên dựa trên các yêu cầu của ứng dụng.
+**ApplicationManager** : Chấp nhận nộp công việc. Đàm phán container đầu tiên để thực thi ApplicationMaster. Một nơi chứa kết hợp các yếu tố như CPU, bộ nhớ, đĩa và mạng. Khởi động lại container ApplicationMaster khi không thành công.

2) Node manager

– Nó chạy trên máy từng nút
– Nó quản lý các thùng chứa. Các vùng chứa không là gì khác ngoài một phần nhỏ dung lượng tài nguyên của Node Manager
– Node manager nút giám sát việc sử dụng tài nguyên của mỗi vùng chứa.
– Nó gửi nhịp tim đến Resource manager

3) Job submitter

– Khách hàng gửi công việc cho Resource manager
– Resource Manager liên hệ với Resource Scheduler và phân bổ vùng chứa.
– Bây giờ Resource manager liên hệ với Node manager có liên quan để khởi chạy vùng chứa.
– Container chạy Application Master.