**Hadoop**

**1.Hdfs**

Là hệ thống phân tán của Hadoop

* Cho phép phân tán dữ liệu
* Cho phép tính toán và phân tán song song

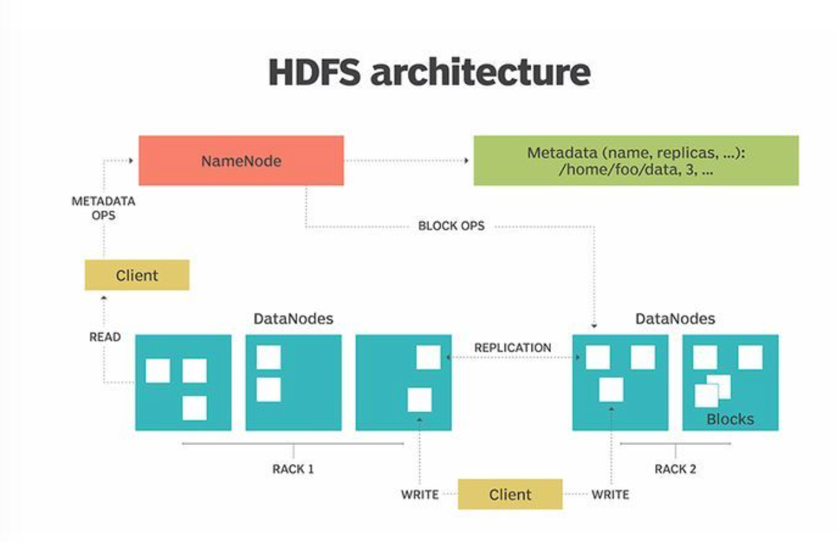
Cho phép nhiều máy tính sử lý chung công việc

* Có khả năng mở rộng theo chiều ngang

(scale out , horizontal scaling) :

* Có khả năng mở rộng theo chiều dọc

**HDFS Architecture**



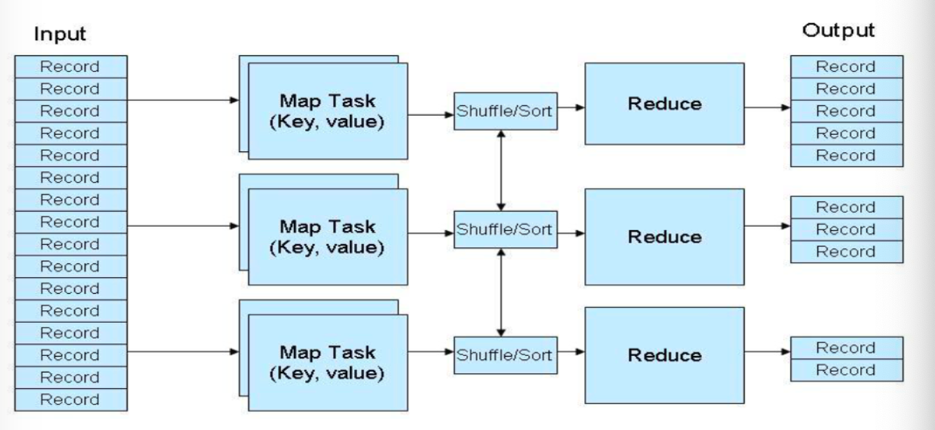
* **NameNode** : chỉ có duy nhất 1 namenode (master) thực hiện nhiệm vị :
  + Lưu chữ metadât của dữ liệu thực tế
  + Quản lý không gian tên của hệ thống file (ánh xạ các file name với các blocks, ánh xạ các block vào các datanode)
  + Quản lý cấu hình của cụm
  + Chỉ định các công việc của Datanode
* **Datanode :**
  + Lưu giữ dữ liệu thực tế
  + Trực tiếp thực hiện các công việc xử lý
* **SecondaryNameNode**
  + Là 1 node phụ chạy cùng với namenode:
  + Đọc ghi các file, Các metadata được lưu trên ram của datanode và ghi nó vào ổ cúng
  + Liện tục dọc nội dung trong Editlog và cập nhập vào Fslmage, để chuẩn bị cho lần khởi dộng tiếp theo của namenode
  + Liên tục kiêm tra tính chính xác của các tin lưu trên các datanode
* **Cơ chế Hearbeat:**
  + Là cách để dâtnode cho nemanode biết nó vẫn còn sống. Định kỳ datanode sẽ gửi một hearbeat về cho namenode để namenode biết là datanode còn đnag hoạt động, nếu không gửi -> chết.
* **Rack:**
  + Là 1 cụm datanode có cùng đầu mạng, bao gồm các máy vật lý ( tương đương 1 server hay 1 node ) cùng kết nối chung 1 swich
* **Blocks**
  + Là đơn vị lưu chữ của HDFS, các data được đưa vào HDFS được chia thành các block có kích thước cố định ( mạc định là 128)

**2.MapReduece**

- MapReduce là một kỹ thuật xử lý và là một mô hình lập trình cho tính toán phân tán để triển khai và xử lý dữ liệu lớn.

- **Các thành phần của mapreduce**

- Maper, Shffle and sorting, Reduce



* Máy master phân phối M Map task cho các máy vào theo dõi tiến trình của chúng
* Các Map task đọc dữ liệu và tiến hành xử lý, kết quả được lưu trữ tại local buffer
* Pha Shuffle chỉ định các reducers tới các vùng nhớ đệm nơi mà chúng được đọc từ xa và xử lý bởi các reducers
* Reducer xuất kết quả và lưu trữ trên HDFS

- **Maper**: Đây là pha đầu tiên của chương trình. Có hai bước trong pha này: splitting and mapping. Một tập dữ liệu được chia thành các đơn vị bằng nhau được gọi là chunks trong bước phân tách (splitting). Hadoop bao gồm một RecordReader sử dụng TextInputFormat để chuyển đổi các phân tách đầu vào thành các cặp key-value.

- **Shuffe**: Đây là pha thứ hai diễn ra sau khi hoàn thành Mapper. Nó bao gồm hai bước chính: sắp xếp và hợp nhất (sorting and merging). Trong pha này, các cặp key-value được sắp xếp bằng cách sử dụng các key. Việc hợp nhất đảm bảo rằng các cặp key-value được kết hợp.

-**Reduce**: Trong pha Reduce, đầu ra của pha Shuffle được sử dụng làm đầu vào. Reducer xử lý đầu vào này hơn nữa để giảm các giá trị trung gian thành các giá trị nhỏ hơn. Nó cung cấp một bản tóm tắt của toàn bộ tập dữ liệu, ví dụ như là tính tổng, tìm max, min,… . Đầu ra của pha này được lưu trữ trong HDFS.

- **ưu diểm:**

- Hỗ trợ xử lý tính toán song song : Trong MapReduce, công việc được phân chia giữa nhiều node và mỗi node hoạt động đồng thời với một phần công việc. Mô hình MapReduce cho phép công việc được phân chia ra thành các công việc nhỏ hơn hoàn toàn riêng biệt.

**3.Yarn(Yet Another Resource negotiator)**

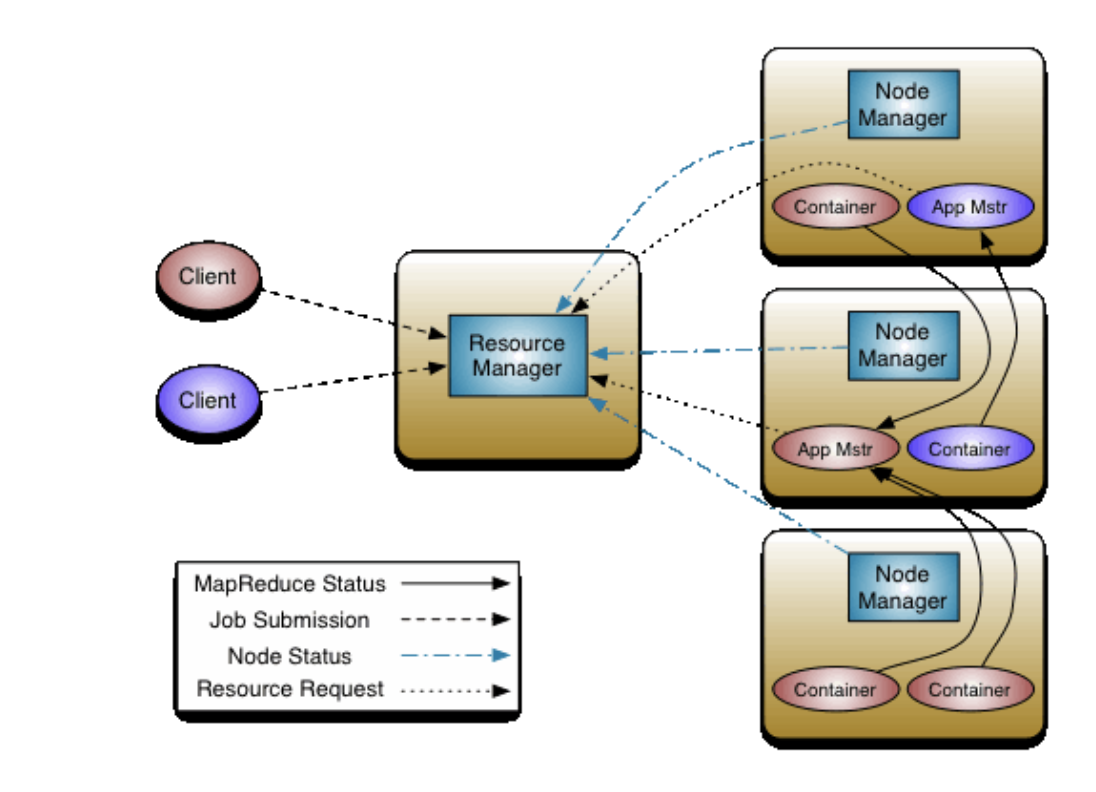
- **Resource manager(RM):** quản lý toàn bộ tài nguyên của cluster

- **Application Master(AM)** đơn vị trên 1 ứng dụng và quản lý vòng đời của Job

Với Yarn , mapreduce sec là 1 ứng dụng chạy trên Yarn, Sử dụng tài nguyên của RM cấp phát. Các node tính toán trong cluster bây giờ sẽ chạy NodeManager quản lý các tiến trình chạy trên máy đó. Resource Manager và Node Manager trở thành xương sống của tính toán phân tán trong YARN. Việc mỗi ứng dụng được tách ra riêng cho phép các process chạy lâu (long running process) cũng có thể được khởi động trên YARN.

ApplicationMaster trên 1 ứng dụng là một thư viện cho phép yêu càu tài nguyên từ Resource Manager và giao tiếp với Node Manager để chạy và thực thi các tasks. Trong YARN, MapReduce2 là thay vì là linh hồn của hadoop như ở hadoop 1 thì chỉ là một ứng dụng. Application Master cho phép xây dựng các ứng dụng khác MR chạy trên YARN.

* **Chạy ứng dụng trên Yarn**



Quá trình 1 ứng dụng chạy trên YARN được mô tả bằng sơ đồ trên. Cụ thể các bước khởi động 1 ứng dụng như sau:

* Client giao 1 task cho Resource Manager
* Resource Manager tính toán tài nguyên cần thiết theo yêu cầu của ứng dụng và tạo 1 App Master (App Mstr). Application Master được chuyển đến chạy 1 một node tính toán. Application Master sẽ liên lạc với các NodeManager ở các node khác để ra yêu cầu công việc cho node này.
* Node Manager nhận yêu cầu và chạy các task trên container
* Các thông tin trạng thái thay vì được gửi đến JobTracker sẽ được gửi đến App Master.

Các ứng dụng khác nhau sẽ có các AppMstr khác nhau và sẽ được khởi động ở các máy trạm khác nhau. Điều này làm việc khi một máy trạm bị lỗi phần cứng và sập, các ứng dụng khác không bị ảnh hưởng, chỉ ứng dụng có AppMstr chạy trên máy đó bị ảnh hưởng.

* **Lập lịch (schedulers)**
  + CapacityScheduler
  + FairScheduler

Scheduler mặc định của YARN là FairScheduler ở đó các Job sẽ được cho vào hàng đợi.

**-Tài Nguyên**

Ở YARN, tài nguyên không đơn thuần là CPU cores nữa mà bao gồm cả bộ nhớ, DiskIO, và GPUs... Một ứng dụng khi khởi động sẽ yêu cầu

* Priority: đặc quyền?
* Bộ nhớ MB: chương trình ứng dụng dự định sẽ sử dụng bao nhiêu MB bộ nhớ
* CPU: ứng dụng dự định sử dụng bao nhiêu cores.
* Số lượng containers.

Containers là số lượng tasks có thể được triển khai song song trên 1 node tính toán. Giới hạn tài nguyên sử dụng bởi 1 containers được thực hiện thông qua tính năng [cgroups](http://en.wikipedia.org/wiki/Cgroups) của Linux kernels.