Bigdata và Hadoop Ecosystem

Giới thiệu Hadoop

Hadoop là một Apache framework mã nguồn mở cho phép phát triển các ứng dụng phân tán (distributed processing) để lưu trữ và quản lý các tập dữ liệu lớn. Hadoop hiện thực mô hình MapReduce, mô hình mà ứng dụng sẽ được chia nhỏ ra thành nhiều phân đoạn khác nhau được chạy song song trên nhiều node khác nhau. Hadoop được viết bằng Java tuy nhiên vẫn hỗ trợ C++, Python, Perl bằng cơ chế streaming.

Tìm hiểu chức năng chính của Hadoop

Hadoop sẽ thực hiện các chức năng sau:

- Xử lý khối dữ liệu lớn (khổng lồ), đơn vị tính dữ liệu mà Hadoop xử lý được tính bằng Petabyte.
- Xử lý những lỗi thường xuyên xuất hiện.
- Xử lý vấn đề trong các môi trường có sự phân tán, nguồn dữ liệu được lưu trữ ở các phần cứng khác nhau và đòi hỏi có sự xử lý đồng bộ.
- Xây dựng băng thông giữa những phần cứng mang nguồn data bị phân tán giới hạn

Kiến trúc cơ bản của Hadoop

Hadoop có một cấu trúc liên kết master-slave. Chức năng của node master là gán một tác vụ cho các node slave khác nhau và quản lý tài nguyên. Các node slave lưu trữ dữ liệu thực trong khi trên master chứa metadata.

Trong một cụm, Hadoop gồm có 2 lớp là HDFS Layer và MapReduce. Trong từng lớp sẽ có những thành phần liên quan riêng. Cụ thể trong Master node sẽ có TaskTracker, JobTracker và Name Node, Data Node. Slave/Worker node sẽ gồm Task Tracker, Data Node hoặc phần này chỉ đơn giản là các dữ liệu hoặc chỉ là node phục vụ nhiệm vụ tính toán. Trong một Hadoop sẽ có 4 module như sau:

- **Hadoop Common**: Đây là các thư viện và tiện ích cần thiết của Java để các module khác sử dụng. Những thư viện này cung cấp hệ thống file và lớp OS trừu tượng, đồng thời chứa các mã lệnh Java để khởi động Hadoop.
- **Hadoop YARN**: Đây là framework để quản lý lập lịch các job và tài nguyên các node.
- **Hadoop Distributed File System (HDFS)**: Đây là hệ thống lưu trữ và quản lý file phân tán, đồng thời cung cấp truy cập thông lượng cao cho ứng dụng khai thác dữ liệu.
- **Hadoop MapReduce**: Đây là hệ thống dựa trên YARN dùng để xử lý song song các tập dữ liệu lớn.

Ưu điểm của Hadoop:

- Hadoop có khả năng thêm nhiều node mới và thay đổi được các cấu hình của chúng một cách dễ dàng.
- Các doanh nghiệp không cần phải đầu tư quá nhiều vào phần cứng quá mạnh và đặc biệt khi chạy Hadoop. Nhờ vậy, có thể tiết kiệm được tối đa các chi phí đầu tư ban đầu.
- Hadoop có khả năng xử lý được hầu hết dữ liệu có cấu trúc hoặc không có cấu trúc một cách dễ dàng.
- Trong suốt quá trình hoạt động thì 1 node trên hệ thống nếu bị lỗi thì nền tảng của Hadoop sẽ có thể tự động di chuyển sang dạng node dự phòng khác. Nhờ vậy mà hệ thống sẽ có thể hoạt động xuyên suốt ổn định hơn.
- Hadoop là mã nguồn mở, điều này giúp nó tương thích rất nhiều cấu hình và platform khác nhau.

Ứng dụng Hadoop:

1. Finance sectors
2. Security and Law Enforcement
3. Companies use Hadoop for understanding customers requirements
4. Hadoop Applications in Retail industry
5. Real-time analysis of customers data
6. Uses of Hadoop in Government sectors
7. Hadoop Uses in Advertisements Targeting Platforms
8. Businesses use Hadoop for sentiment analysis

9. Hadoop Applications in Financial Trading and Forecasting
10. Hadoop applications in improving Personal Quantification
11. Healthcare sectors
12. Optimizing machine performance

Tìm hiểu sâu về HDFS

HDFS là viết tắt của **Hệ thống tệp phân tán Hadoop (Hadoop Distributed File System)** . Nó cung cấp cho việc lưu trữ dữ liệu của Hadoop. HDFS chia đơn vị dữ liệu thành các đơn vị nhỏ hơn gọi là các block và lưu trữ chúng theo cách phân tán.

a. NameNode và DataNode

HDFS có kiến trúc Master-Slave . Trình nền có tên NameNode chạy trên máy master chính. Nó chịu trách nhiệm quản lý Namespace và điều chỉnh truy cập tệp của client. DataNode chạy trên các nút slave. Nó có trách nhiệm lưu trữ business data thực tế. Bên trong đó, một tệp được chia thành một số block và được lưu trữ trên một nhóm các máy phụ thuộc. Namenode quản lý sửa đổi namespace tập tin. Chính vì vậy là những hành động như mở, đóng và đổi tên tập tin hoặc thư mục đều thông qua Namenode. NameNode cũng theo dõi ánh xạ các block tới DataNodes. DataNodes này phục vụ yêu cầu đọc / ghi từ client của file system. DataNode cũng tạo, xóa và sao chép các khối theo yêu cầu từ NameNode.

HDFS dược viết bằng ngôn ngữ chính là java. Do đó, người ta có thể deploy DataNode và NameNode trên các máy đã cài đặt Java. Thông thường sẽ cần có một máy chuyên dụng chạy NameNode. Và tất cả các nút khác trong cụm chạy DataNode. NameNode chứa siêu dữ liệu giống như vị trí của các khối trên DataNodes. Và phân chia tài nguyên giữa các DataNodes khác nhau.

b. Block trong HDFS

Block là đơn vị lưu trữ nhỏ nhất trên hệ thống máy tính. Đây là bộ lưu trữ liền kề nhỏ nhất được phân bổ cho một tệp. Trong Hadoop, **kích thước khối mặc định là 128 MB hoặc 256 MB** .

c. Quản lý Replication(nhân bản)

Để cung cấp **khả năng chịu lỗi HDFS** sử dụng kỹ thuật replication. Trong đó, nó tạo các bản sao của các block và lưu trữ trên các DataNodes khác nhau. Hệ số sao chép quyết định số lượng bản sao của các block được lưu trữ. Nó là 3 theo mặc định nhưng có thể cấu hình thành bất kỳ giá trị nào. Giả sử chúng ta có một tệp 1GB thì với hệ số sao chép là 3, nó sẽ yêu cầu 3GB tổng dung lượng lưu trữ.

Để duy trì hệ số replication NameNode thu thập báo cáo khối từ mỗi DataNode. Bất cứ khi nào một khối được sao chép dưới mức hoặc sao chép quá mức, NameNode sẽ thêm hoặc xóa các bản sao tương ứng.

d. Rack Awareness

Một rack chứa nhiều máy DataNode và có nhiều rack như vậy trong sản xuất. **HDFS tuân theo thuật toán Rack Awareness** để đặt các bản sao của các block theo kiểu phân tán. Thuật toán Rack Awareness này cung cấp độ trễ thấp và khả năng chịu lỗi. Giả sử hệ số replication được cấu hình là 3. Bây giờ thuật toán rack awareness sẽ đặt block đầu tiên lên một rack. Nó sẽ giữ hai block còn lại trên một rack khác. Nó không lưu trữ nhiều hơn hai khối trong cùng một rack nếu có thể.

Làm như vậy sẽ giải quyết được các vấn đề sau:

- Để cải thiện độ sẵn sàng cao và độ tin cậy của dữ liệu.
- Cải thiện hiệu suất của cụm.
- Để cải thiện băng thông mạng.
- Tránh mất dữ liệu nếu toàn bộ rack bị hỏng mặc dù khả năng lỗi rack thấp hơn rất nhiều so với lỗi node.
- Để giữ dữ liệu số lượng lớn trong giá khi có thể.
- Băng thông cao hơn nên độ trễ thấp hơn.