

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

🙠🟅🙢



**BÁO CÁO CUỐI KÌ**

**ĐIỆN TOÁN ĐÁM MÂY**

**TÌM HIỂU PARQUET VÀ**

**VIẾT ỨNG DỤNG DEMO**

**GVDH: HUỲNH XUÂN PHỤNG**

**SVTH:**

**Trần Kiện Khang 19110375**

**Lê Thị Minh Nguyệt 19110413**

**Trần Đông Thịnh 19110466**

*Thành phố Hồ Chí Minh – Tháng 12/2021*

|  |  |
| --- | --- |
| **ĐH SƯ PHẠM KỸ THUẬT TP. HCM**  **KHOA CNTT**  \*\*\*\*\*\*\* | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh Phúc**  \*\*\*\*\*\*\* |

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Họ và tên Sinh viên 1: **Lê Thị Minh Nguyệt** MSSV 1: **19110413**

Họ và tên Sinh viên 2: **Trần Kiện Khang** MSSV 2: **19110375**

Họ và tên Sinh viên 2: **Trần Đông Thịnh** MSSV 2: **19110466**

Ngành: **Công nghệ Thông tin**

Tên đề tài: **Tìm hiểu Parquet và viết ứng dụng demo**

Họ và tên Giáo viên hướng dẫn: **TS. Huỳnh Xuân Phụng**

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện:

2. Ưu điểm:

3. Khuyết điểm:

4. Đề nghị cho bảo vệ hay không?

5. Đánh giá loại:

6. Điểm:

|  |  |  |  |
| --- | --- | --- | --- |
| *Tp. Hồ Chí Minh, ngày tháng năm 2021* | | | |
|  | | Giáo viên hướng dẫn  *(Ký & ghi rõ họ tên)* |
| **ĐH SƯ PHẠM KỸ THUẬT TP. HCM**  **KHOA CNTT**  \*\*\*\*\*\*\* | | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh Phúc**  \*\*\*\*\*\*\* | | | |

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Họ và tên Sinh viên 1: **Lê Thị Minh Nguyệt** MSSV 1: **19110413**

Họ và tên Sinh viên 2: **Trần Kiện Khang** MSSV 2: **19110375**

Họ và tên Sinh viên 2: **Trần Đông Thịnh** MSSV 2: **19110466**

Ngành: **Công nghệ Thông tin**

Tên đề tài: **Tìm hiểu Parquet và viết ứng dụng demo**

Họ và tên Giáo viên hướng dẫn: **TS. Huỳnh Xuân Phụng**

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện:

2. Ưu điểm:

3. Khuyết điểm:

4. Đề nghị cho bảo vệ hay không?

5. Đánh giá loại:

6. Điểm:

*Tp. Hồ Chí Minh, ngày tháng năm 2021*

Giáo viên phản biện

(Ký & ghi rõ họ tên)

1. LỜI CẢM ƠN

*Lời đầu tiên, nhóm chúng em xin gửi lời cám ơn sâu sắc nhất đến thầy Huỳnh Xuân Phụng.*

*Trong quá trình tìm hiểu và học tập môn Điện toán đám mây chúng em đã nhận được sự giảng dạy và hướng dẫn tận tình, tâm huyết từ thầy, giúp chúng em tích lũy thêm rất nhiều kiến thức. Từ những kiến thức thầy đã truyền đạt chúng em xin trình bày lại những gì mình đã làm được trong bài tiểu luận môn học này.*

*Tuy nhiên, kiến thức về môn học của chúng em vẫn còn những hạn chế nhất định và không tránh khỏi những thiếu sót trong quá trình hoàn thành bài báo cáo này. Chúng em mong nhận được lời góp ý, nhận xét đến từ thầy để đề tài báo cáo của chúng em được hoàn thiện hơn.*

*Chúc em kính chúc thầy sức khỏe, hạnh phúc và thành công hơn nữa trong sự nghiệp trồng người để tiếp tục dìu dắt nhiều thế hệ sinh viên đến những bến bờ tri thức.*

*Chúng em xin chân thành cảm ơn thầy!*

***Nhóm thực hiện***

*Lê Thị Minh Nguyệt – 19110413*

*Trần Kiện Khang – 19110375*

*Trần Đông Thịnh – 19110466*

**MỤC LỤC**

[LỜI CẢM ƠN 4](#_Toc91033169)

[MỤC LỤC 5](#_Toc91033170)

[DANH MỤC HÌNH ẢNH 7](#_Toc91033171)

[DANH MỤC HÌNH 8](#_Toc91033172)

[PHẦN MỞ ĐẦU 9](#_Toc91033173)

[1. Tính cấp thiết của đề tài 9](#_Toc91033174)

[2. Đối tượng nghiên cứu 9](#_Toc91033175)

[3. Phạm vi nghiên cứu 9](#_Toc91033176)

[4. Kết quả dự kiến đạt được 9](#_Toc91033177)

[PHẦN NỘI DUNG 10](#_Toc91033178)

[CHƯƠNG 1: LÝ THUYẾT VỀ APACHE PARQUET 10](#_Toc91033179)

[1. Apache Parquet 10](#_Toc91033180)

[2. Lịch sử 10](#_Toc91033181)

[3. Đặc trưng 11](#_Toc91033182)

[4. Ưu điểm 11](#_Toc91033183)

[4.1. Nén 11](#_Toc91033184)

[4.2. Hiệu suất 11](#_Toc91033185)

[4.3. Hỗ trợ mã nguồn mở 11](#_Toc91033186)

[4.4. Định hướng cột với việc lưu trữ dạng cột để truy vấn phân tích 12](#_Toc91033187)

[5. So sánh định dạng CSV và Parquet 12](#_Toc91033188)

[CHƯƠNG 2: MÔ HÌNH MAPREDUCE 14](#_Toc91033189)

[1. MapReduce là gì? 14](#_Toc91033190)

[2. Các hàm chính của MapReduce 14](#_Toc91033191)

[3. Các ưu điểm nổi bật của MapReduce 15](#_Toc91033192)

[4. Hoạt động của MapReduce 15](#_Toc91033193)

[CHƯƠNG 3: CÀI ĐẶT VÀ DEMO 17](#_Toc91033194)

[1. Tạo máy ảo EC2 trên AWS 17](#_Toc91033195)

[2. Cài đặt Hadoop phiên bản 2.7.7 18](#_Toc91033196)

[2.1. Tổng quan về Hadoop 18](#_Toc91033197)

[2.2. Cài đặt hadoop 18](#_Toc91033198)

[3. Chuyển file CSV sang Parquet 27](#_Toc91033199)

[4. Đọc và ghi file Parquet bằng MapReduce 28](#_Toc91033200)

[4.1. Tạo nơi lưu trữ data 28](#_Toc91033201)

[4.2. Code MapReduce 28](#_Toc91033202)

[PHẦN KẾT LUẬN 33](#_Toc91033203)

[1. Kết quả đạt được 33](#_Toc91033204)

[1.1. Kiến thức tìm hiểu được 33](#_Toc91033205)

[1.2. Chương trình đã làm được 33](#_Toc91033206)

[2. Ưu điểm 33](#_Toc91033207)

[3. Nhược điểm 33](#_Toc91033208)

[4. Hướng phát triển 33](#_Toc91033209)

[TÀI LIỆU THAM KHẢO 34](#_Toc91033210)

1. DANH MỤC HÌNH ẢNH

[Hình 1: Lưu trữ dạng cột và hàng 10](#_Toc91020682)

1. DANH MỤC HÌNH

[Bảng 1: So sánh dữ liệu định dạng theo CSV và Parquet 13](#_Toc91020704)

1. PHẦN MỞ ĐẦU
   1. Tính cấp thiết của đề tài

Vài năm trở lại đây, cụm từ “big data” được nhắc rất nhiều trong các bài báo dự đoán về xu hướng công nghệ cũng như các cuộc hội thảo. Nó bắt đầu xuất hiện khi có sự ra đời của điện toán đám mây. “Big Data” tạm dịch là dữ liệu khổng lồ.

Dung lượng dữ liệu tạo ra nhanh chóng tới mức hơn 90% dung lượng dữ liệu từ trước tới nay. Dữ liệu này được tạo ra mọi lúc mọi nơi: từ những chiếc cảm biến để thu thập thông tin về thời tiết, những thông tin được đưa lên các trang mạng xã hội, diễn đàn, báo chí, … Dữ liệu được đưa ra dưới mọi hình thức khác nhau: văn bản, hình ảnh, video, audio….

Big Data được xử lý thông qua 4 giai đoạn: thu thập (acquire), tổ chức (organize), phân tích (analyze), quyết định (decide). Đề tài chúng em nghiên cứu là một phần nhỏ trong giai đoạn tổ chức. Đó là lưu trữ dữ liệu ở dưới dạng file Parquet, đọc ghi sử dụng Hadoop/MapReduce.

* 1. Đối tượng nghiên cứu

Đối với đề tài này, đối tượng nghiên cứu là Big Data. Đồng thời kèm theo đó là các công nghệ áp dụng, cụ thể như:

* JDK và JRE: Bộ công cụ phát triển Java.
* Hadoop: một framwork giúp lưu trữ và xử lý Big Data áp dụng MapReduce.
* Apache Parquet: Apache Parquet là một định dạng lưu trữ dữ liệu theo cột miễn phí và nguồn mở của hệ sinh thái Apache Hadoop.
  1. Phạm vi nghiên cứu

Đề tài chúng em chủ yếu tập trung vào sử dụng máy ảo EC2 để thực hiện các vấn đề chuyển dữ liệu lưu trữ ở dạng hàng (cụ thể là CSV) thành dữ liệu lưu trữ ở dạng cột (Parquet File), đọc và ghi dữ liệu trong Hadoop sử dụng mô hình MapReduce.

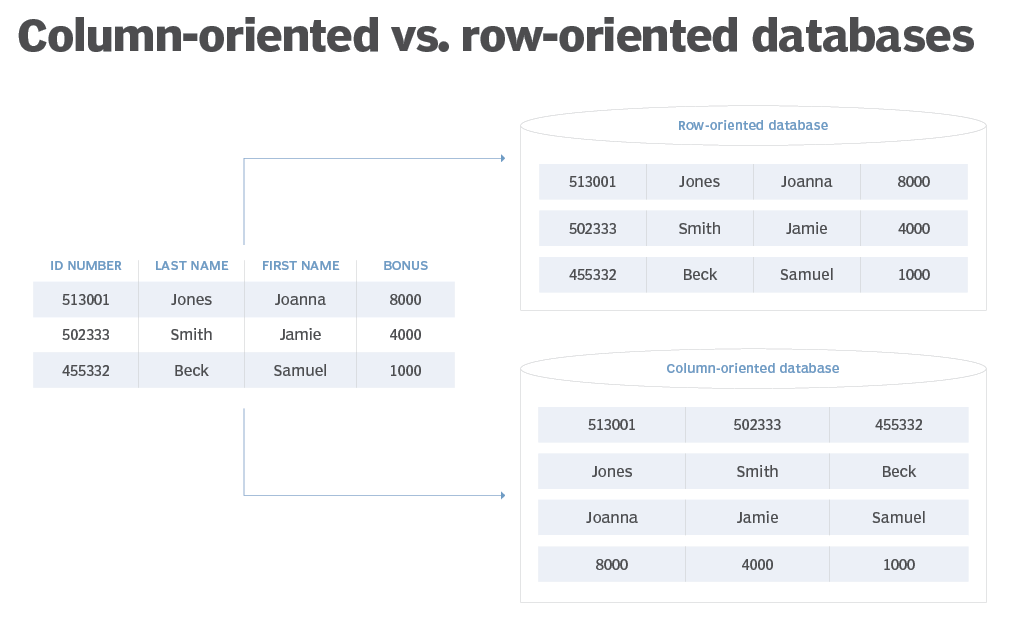
* 1. Kết quả dự kiến đạt được
* Cài đặt được Hadoop
* Chuyển file CSV thành Parquet
* Đọc và ghi file Parquet

1. PHẦN NỘI DUNG
2. CHƯƠNG 1: LÝ THUYẾT VỀ APACHE PARQUET
   1. Apache Parquet

Apache Parquet là một định dạng lưu trữ dạng cột có sẵn cho bất kỳ dự án nào trong hệ sinh thái Hadoop, bất kể lựa chọn khung xử lý dữ liệu, mô hình dữ liệu hoặc ngôn ngữ lập trình.

Parquet là một định dạng tệp mã nguồn mở có sẵn cho bất kỳ dự án nào trong hệ sinh thái Hadoop. Apache Parquet được thiết kế cho định dạng lưu trữ dữ liệu dạng cột hiệu quả và hiệu quả so với các tệp hàng như tệp CSV hoặc TSV.

Cột: Không giống như các định dạng dựa trên hàng như CSV hoặc Avro, Apache Parquet hướng theo cột - nghĩa là các giá trị của mỗi cột trong bảng được lưu trữ bên cạnh nhau, thay vì các giá trị của mỗi bản ghi.



Hình 1: Lưu trữ dạng cột và hàng

Mã nguồn mở: Parquet được sử dụng miễn phí và mã nguồn mở theo giấy phép Apache Hadoop và tương thích với hầu hết data framework Hadoop.

Tự mô tả: Trong Parquet, siêu dữ liệu bao gồm lược đồ và cấu trúc được nhúng trong mỗi tệp, làm cho nó trở thành định dạng tệp tự mô tả.

* 1. Lịch sử

Dự án mã nguồn mở để xây dựng Apache Parquet bắt đầu như một nỗ lực chung giữa Twitter và Cloudera. Parquet được thiết kế như một sự cải tiến dựa trên định dạng lưu trữ dạng cột Trevni do người sáng tạo Doug Cutting của Hadoop tạo ra. Phiên bản đầu tiên, Apache Parquet 1.0, được phát hành vào tháng 7 năm 2013. Kể từ ngày 27 tháng 4 năm 2015, Apache Parquet là một dự án cấp cao nhất được Apache Software Foundation (ASF) kiểm duyệt.

* 1. Đặc trưng

Parquet sử dụng thuật toán cắt nhỏ bản ghi và thuật toán lắp ráp, đáp ứng các cấu trúc dữ liệu phức tạp có thể được sử dụng để lưu trữ dữ liệu. Parquet được tối ưu hóa để làm việc với dữ liệu phức tạp với số lượng lớn và có các cách khác nhau để nén, mã hóa dữ liệu hiệu quả. Các giá trị trong mỗi cột được lưu trữ vật lý trong các vị trí bộ nhớ liền kề và việc lưu trữ theo cột này cung cấp các lợi ích sau:

* Nén theo cột hiệu quả và tiết kiệm không gian lưu trữ
* Những câu truy vấn tìm các giá trị cột cụ thể không cần đọc toàn bộ dữ liệu hàng, do đó cải thiện hiệu suất
* Các kỹ thuật mã hóa khác nhau có thể được áp dụng cho các cột khác nhau
  1. Ưu điểm
     1. Nén

Nén là việc lấy một tệp làm cho nó có kích thước nhỏ hơn. Trong Parquet, quá trình nén được thực hiện theo từng cột và nó được xây dựng để hỗ trợ các tùy chọn nén linh hoạt với các lược đồ mã hóa có thể mở rộng cho mỗi loại dữ liệu - ví dụ: có thể sử dụng các mã hóa khác nhau để nén dữ liệu số nguyên và chuỗi.

Dữ liệu parquet có thể được nén bằng cách sử dụng các phương pháp mã hóa sau:

* Dictionary: mã hoá này được thực hiện một cách tự động đối với dữ liệu là số nhỏ và có giá trị duy nhất
* Bit Packing: Lưu trữ hiệu quả với các số nguyên nhỏ
* Run length encoding (RLE): khi cùng một giá trị xuất hiện nhiều lần, một giá trị duy nhất được lưu trữ một lần cùng với số lần xuất hiện. Parquet triển khai một phiên bản kết hợp của Bit Packing và RLE, trong đó bộ mã hóa chuyển mạch dựa trên đó tạo ra kết quả nén tốt nhất.
  + 1. Hiệu suất

Parquet được tối ưu hóa cho hiệu suất. Khi chạy các truy vấn trên hệ thống tệp dựa trên Parquet của bạn, bạn có thể chỉ tập trung vào dữ liệu có liên quan rất nhanh chóng. Hơn nữa, lượng dữ liệu được quét sẽ nhỏ hơn và dẫn đến việc giảm thiểu dữ liệu được quét.

* + 1. Hỗ trợ mã nguồn mở

Do Parquet là một phần của hệ sinh thái Hadoop, có mã nguồn mở, đang được cải tiến và hỗ trợ bởi một cộng đồng và nhà phát triển một cách mạnh mẽ. Lưu trữ dữ liệu ở định dạng mở tránh việc bị nhà sản xuất khoá và tăng tính linh hoạt của mình.

* + 1. Định hướng cột với việc lưu trữ dạng cột để truy vấn phân tích

Dữ liệu phức tạp như logs và event streams sẽ cần được biểu diễn dưới dạng một bảng có hàng trăm hoặc hàng nghìn cột và nhiều triệu hàng. Lưu trữ bảng này ở định dạng dựa trên hàng, chẳng hạn như CSV có nghĩa là:

* Các truy vấn sẽ mất nhiều thời gian hơn để chạy vì nhiều dữ liệu hơn cần được quét, thay vì chỉ truy vấn tập hợp con của các cột mà chúng ta cần
* Lưu trữ tốn kém hơn vì CSV không được nén như Parquet

Định dạng cột cung cấp khả năng nén tốt hơn và cải thiện hiệu suất ngay lập tức, đồng thời cho phép bạn truy vấn dữ liệu theo chiều dọc - từng cột.

* 1. So sánh định dạng CSV và Parquet

CSV là một định dạng đơn giản và phổ biến rộng rãi được sử dụng bởi nhiều công cụ như Excel, Google Trang tính và nhiều công cụ khác có thể tạo tệp CSV. Mặc dù các tệp CSV là định dạng mặc định cho các đường ống xử lý dữ liệu, nó có một số nhược điểm:

* Amazon Athena và Spectrum sẽ tính phí dựa trên lượng dữ liệu được quét trên mỗi truy vấn.
* Google và Amazon sẽ tính phí bạn theo lượng dữ liệu được lưu trữ trên GS / S3.
* Các khoản phí Dataproc của Google dựa trên thời gian.

Parquet đã giúp người dùng giảm ít nhất một phần ba yêu cầu lưu trữ trên các bộ dữ liệu lớn, ngoài ra, nó còn cải thiện đáng kể thời gian quét và giải mã, do đó làm tăng chi phí tổng thể.

Bảng sau đây so sánh số tiền tiết kiệm được cũng như tốc độ tăng tốc thu được khi chuyển đổi dữ liệu thành Parquet từ CSV.

Bảng 1: So sánh dữ liệu định dạng theo CSV và Parquet

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset | Kích thước trên Amazon S3 | Thời gian chạy truy vấn | Dữ liệu được quét | Trị giá |
| Dữ liệu được lưu trữ dưới dạng tệp CSV | 1 TB | 236 giây | 1,15 TB | $ 5,75 |
| Dữ liệu được lưu trữ ở định dạng Apache Parquet | 130 GB | 6,78 giây | 2,51 GB | $ 0,01 |
| Tiết kiệm | Giảm 87% khi sử dụng Parquet | Nhanh hơn 34 lần | Dữ liệu được quét ít hơn 99% | Tiết kiệm 99,7% |

1. CHƯƠNG 2: MÔ HÌNH MAPREDUCE
   1. MapReduce là gì?

MapReduce là mô hình được thiết kế độc quyền bởi Google, nó là một mô hình hoặc mẫu lập trình trong khuôn khổ Hadoop được sử dụng để truy cập dữ liệu lớn được lưu trữ trong Hệ thống tệp Hadoop (HDFS). Nó là một thành phần cốt lõi, không thể thiếu đối với hoạt động của khuôn khổ Hadoop.

MapReduce hỗ trợ xử lý đồng thời bằng cách chia nhỏ petabyte dữ liệu thành các phần nhỏ hơn và xử lý chúng song song trên các máy chủ bình thường trên Hadoop. Cuối cùng, nó tổng hợp tất cả dữ liệu từ nhiều máy chủ để trả lại đầu ra tổng hợp cho ứng dụng.

Ví dụ: một cụm Hadoop với 20.000 máy chủ bình thường không quá đắt đỏ và khối dữ liệu 256 MB trong mỗi cụm, có thể xử lý khoảng 5TB dữ liệu cùng một lúc. Điều này làm giảm thời gian xử lý so với xử lý tuần tự một tập dữ liệu lớn.

MapReduce đã từng là phương pháp duy nhất mà qua đó dữ liệu được lưu trữ trong HDFS có thể được truy xuất, nhưng hiện nay không còn như vậy nữa. Ngày nay, có các hệ thống dựa trên truy vấn khác như Hive và Pig được sử dụng để truy xuất dữ liệu từ HDFS bằng cách sử dụng các câu lệnh giống SQL. Tuy nhiên, chúng thường chạy cùng với các công việc được viết bằng mô hình MapReduce. Đó là bởi vì MapReduce có những lợi thế riêng.

* 1. Các hàm chính của MapReduce

MapReduce có 2 hàm chính là Map () và Reduce (), đây là 2 hàm đã được định nghĩa bởi người dùng và nó cũng chính là 2 giai đoạn liên tiếp trong quá trình xử lý dữ liệu của MapReduce. Nhiệm vụ cụ thể của từng hàm như sau:

* Hàm Map (): có nhiệm vụ nhận Input cho các cặp giá trị/ khóa và output chính là tập những cặp giá trị/khóa trung gian. Sau đó, chỉ cần ghi xuống đĩa cứng và tiến hành thông báo cho các hàm Reduce () để trực tiếp nhận dữ liệu.
* Hàm Reduce (): có nhiệm vụ tiếp nhận từ khóa trung gian và những giá trị tương ứng với lượng từ khóa đó. Sau đó, tiến hành ghép chúng lại để có thể tạo thành một tập khóa khác nhau. Các cặp khóa/giá trị này thường sẽ thông qua một con trỏ vị trí để đưa vào các hàm reduce. Quá trình này sẽ giúp cho lập trình viên quản lý dễ dàng hơn một lượng danh sách cũng như phân bổ giá trị sao cho phù hợp nhất với bộ nhớ hệ thống.
* Ở giữa Map và Reduce thì còn 1 bước trung gian đó chính là Shuffle. Sau khi Map hoàn thành xong công việc của mình thì Shuffle sẽ làm nhiệm vụ chính là thu thập cũng như tổng hợp từ khóa/giá trị trung gian đã được map sinh ra trước đó rồi chuyển qua cho Reduce tiếp tục xử lý.
  1. Các ưu điểm nổi bật của MapReduce

Mapreduce được ưa chuộng sử dụng như vậy bởi nó sở hữu nhiều ưu điểm vượt trội như sau:

* MapReduce có khả năng xử lý dễ dàng mọi bài toán có lượng dữ liệu lớn nhờ khả năng tác vụ phân tích và tính toán phức tạp. Nó có thể xử lý nhanh chóng cho ra kết quả dễ dàng chỉ trong khoảng thời gian ngắn.
* Mapreduce có khả năng chạy song song trên các máy có sự phân tán khác nhau. Với khả năng hoạt động độc lập kết hợp phân tán, xử lý các lỗi kỹ thuật để mang lại nhiều hiệu quả cho toàn hệ thống.
* MapRedue có khả năng thực hiện trên nhiều nguồn ngôn ngữ lập trình khác nhau như: Java, C/ C++, Python, Perl, Ruby, ... tương ứng với nó là những thư viện hỗ trợ.
* Như bạn đã biết, mã độc trên internet ngày càng nhiều hơn nên việc xử lý những đoạn mã độc này cũng trở nên rất phức tạp và tốn kém nhiều thời gian. Chính vì vậy, các ứng dụng MapReduce dần hướng đến quan tâm nhiều hơn cho việc phát hiện các mã độc để có thể xử lý chúng. Nhờ vậy, hệ thống mới có thể vận hành trơn tru và được bảo mật nhất.
  1. Hoạt động của MapReduce

*Nguyên tắc hoạt động*

Mapreduce hoạt động dựa vào nguyên tắc chính là "Chia để trị", như sau:

* Phân chia các dữ liệu cần xử lý thành nhiều phần nhỏ trước khi thực hiện.
* Xử lý các vấn đề nhỏ theo phương thức song song trên các máy tính rồi phân tán hoạt động theo hướng độc lập.
* Tiến hành tổng hợp những kết quả thu được để đề ra được kết quả sau cùng.

*Các bước hoạt động của MapReduce*

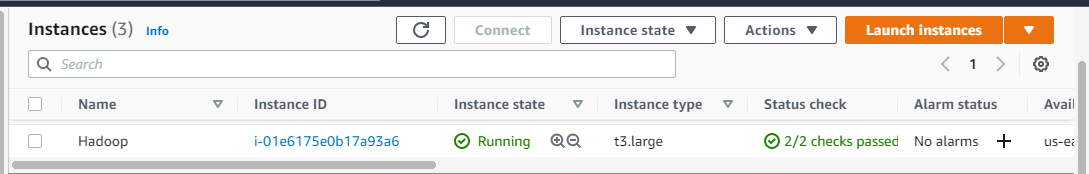
* Bước 1: Tiến hành chuẩn bị các dữ liệu đầu vào để cho Map () có thể xử lý.
* Bước 2: Lập trình viên thực thi các mã Map () để xử lý.
* Bước 3: Tiến hành trộn lẫn các dữ liệu được xuất ra bởi Map () vào trong Reduce Processor
* Bước 4: Tiến hành thực thi tiếp mã Reduce () để có thể xử lý tiếp các dữ liệu cần thiết.
* Bước 5: Thực hiện tạo các dữ liệu xuất ra cuối cùng.

*Luồng dữ liệu nền tảng của Mapreduce*

* Input Reader
* Map Function
* Partition Function
* Compare Function
* Reduce Function
* Output Writer

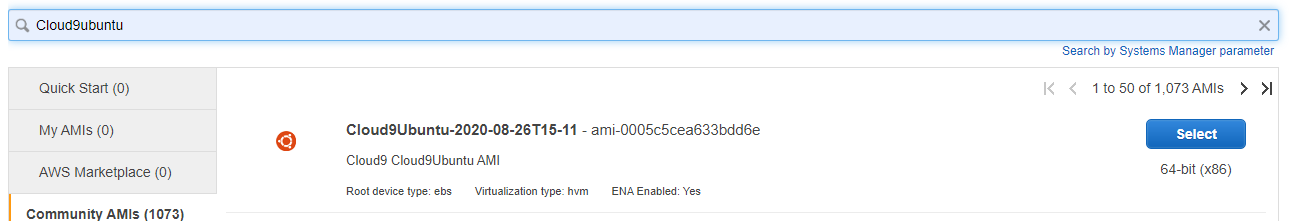
1. CHƯƠNG 3: CÀI ĐẶT VÀ DEMO
   1. Tạo máy ảo EC2 trên AWS

Tạo 1 máy ảo phiên bản Cloud9Ubuntu-2020-08-26T15-11 trên Amazon EC2.



Máy ảo EC2 trên AWS được tạo như thông thường và có các đặc tính đặc biệt sau:

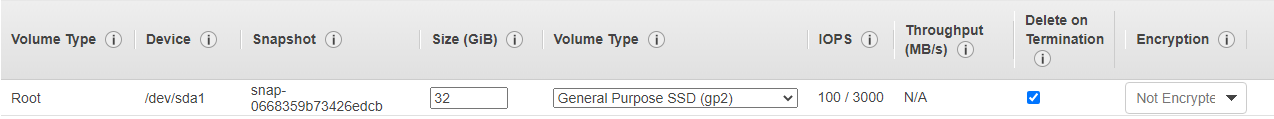
+ Chọn máy Cloud9Ubuntu-2020-08-26T15-11



+ Chọn phiên bản của máy: 2 vCPUs và 8 Memory (GiB)



+ Bộ nhớ máy là 32 GB



+ Cấu hình bảo mật: Mình sẽ bật tất cả các port và chỗ Source chọn Anywhere



+ Mỗi lần sử dụng máy ảo trên EC2 chúng ta phải kết nối với nó



* 1. Cài đặt Hadoop phiên bản 2.7.7
     1. Tổng quan về Hadoop

Hadoop là một khuôn khổ để chạy các chương trình máy tính phân tán. Nó bao gồm HDFS và Map Reduce (Khung lập trình).

Người dùng chỉ có thể chạy chương trình MapReduce trong các phiên bản trước của Hadoop. Do đó, nó phù hợp cho các tính toán xử lý hàng loạt.

Các sợi cung cấp API để yêu cầu và phân bổ tài nguyên trong cluster. Vì vậy, YARN có sẵn trong các phiên bản sau của Hadoop 2. Do đó, API tạo điều kiện cho chương trình ứng dụng xử lý dữ liệu quy mô lớn của HDFS.

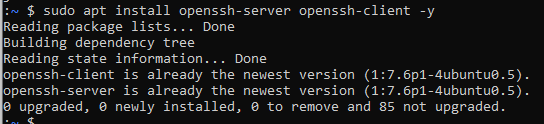
* + 1. Cài đặt hadoop

**2.2.1. Thiết lập ssh không mật khẩu**

*a) Cài đặt Máy chủ SSH Mở và Máy khách SSH Mở*

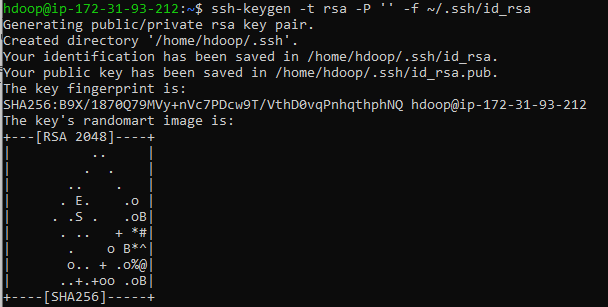
Thiết lập ứng dụng ssh không mật khẩu bằng lệnh sau.

|  |
| --- |
| [php] sudo apt-get install openssh-server openssh-client [/ php] |



*b) Tạo các cặp khóa công khai và khóa riêng tư*

|  |
| --- |
| [php] ssh-keygen -t rsa -P “” [/ php] |

Terminal sẽ nhắc người dùng nhập tên tệp. Nhấn enter và tiếp tục. Vị trí của một tệp sẽ nằm trong thư mục chính. Hơn nữa, phần mở rộng sẽ là tệp .ssh.

*c) Định cấu hình SSH không cần mật khẩu*

Lệnh dưới đây sẽ thêm khóa ssh công khai vào các khóa được ủy quyền. Hơn nữa, nó sẽ cấu hình ssh không mật khẩu.

|  |
| --- |
| [php] cat $ HOME / .ssh / id\_rsa.pub >> $ HOME / .ssh / allow\_keys [/ php] |

*d) Tiếp theo xác minh hoạt động của ssh không có mật khẩu*

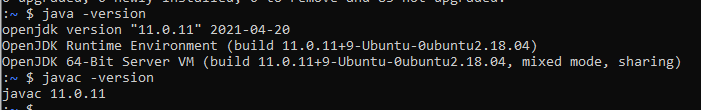
Khi chúng ta nhập “ssh localhost”, nó sẽ nhắc chúng ta kết nối với nó. Nhập 'yes' và nhấn enter để tiếp tục.

*e) Cài đặt rsync bằng lệnh*

|  |
| --- |
| [php] $ sudo apt-get install rsync [/ php] |

Kiểm tra phiên bản java hiện tại

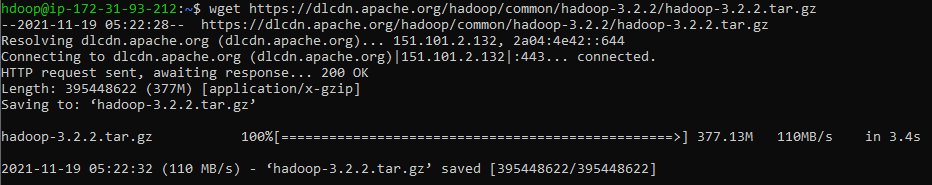
|  |
| --- |
| [php] $ java -version [/ php] |



**2.2.2. Cấu hình và thiết lập hadoop**

*a) Tải xuống gói hadoop 2.7.7*

Sử dụng liên kết nhân bản được cung cấp và tải xuống gói Hadoop bằng **wget**lệnh:

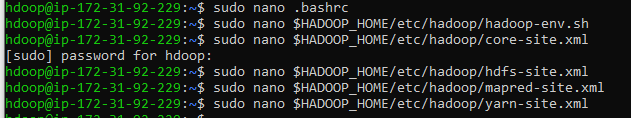


*b) Mở Tarball*

Sau khi quá trình tải xuống hoàn tất, hãy giải nén các tệp để bắt đầu cài đặt Hadoop:

|  |
| --- |
| tar xzf hadoop-2.7.7.tar.gz |

**2.2.3. Thiết lập cấu hình**



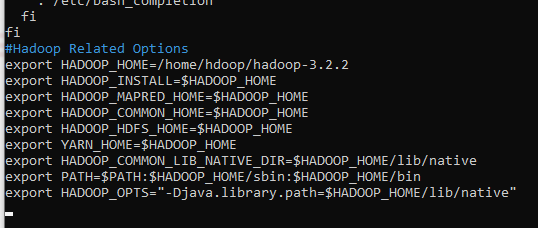
*a) Thiết lập các biến môi trường*

Chỉnh sửa .bashrc- Chỉnh sửa bashrc và do đó thêm hadoop vào một đường dẫn:

|  |
| --- |
| [php] nano bash.bashrc [/ php] |

Và thêm các biến đường dẫn sau vào đó

|  |
| --- |
| [php]export HADOOP\_HOME=/home/hduser/hadoop-2.7.7 export HADOOP\_INSTALL=$HADOOP\_HOME export HADOOP\_MAPRED\_HOME=$HADOOP\_HOME export HADOOP\_COMMON\_HOME=$HADOOP\_HOME export HADOOP\_HDFS\_HOME=$HADOOP\_HOME export YARN\_HOME=$HADOOP\_HOME export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_HOME/lib/native export PATH=$PATH:$HADOOP\_HOME/sbin:$HADOOP\_HOME/bin[/php] |



Điều quan trọng là phải áp dụng các thay đổi cho môi trường đang chạy hiện tại bằng cách sử dụng lệnh sau:

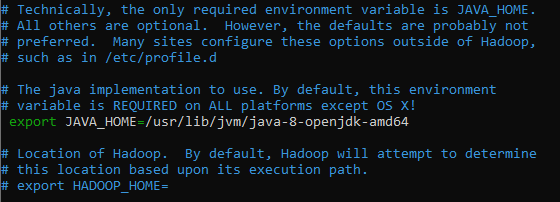
|  |
| --- |
| source ~/.bashrc |

*b) Thay đổi tệp cấu hình Hadoop*

* **Chỉnh sửa hadoop-env.sh**

Chỉnh sửa tệp hadoop-env.sh nằm trong etc / hadoop bên trong thư mục cài đặt Hadoop. Người dùng có thể đặt JAVA\_HOME:

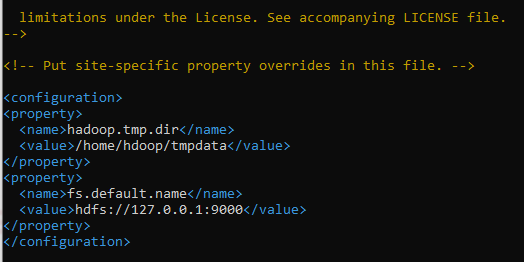
|  |
| --- |
| [php] export JAVA\_HOME = /usr/lib/jvm/java-8-openjdk-amd64 |



* **Chỉnh sửa core-site.xml**

Chỉnh sửa core-site.xml bằng “nano core-site.xml”. Tệp nằm trong etc / hadoop bên trong thư mục Hadoop. Sau đó, Thêm cấu hình sau để ghi đè các giá trị mặc định cho thư mục tạm thời và thêm URL HDFS để thay thế cài đặt hệ thống tệp cục bộ mặc định:

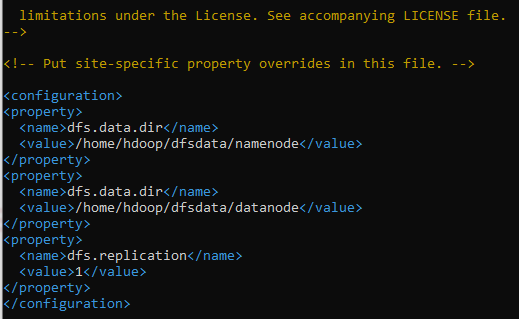
|  |
| --- |
| [php] <configuration>  <property>  <name>hadoop.tmp.dir</name>  <value>/home/hdoop/tmpdata</value>  </property>  <property>  <name>fs.default.name</name>  <value>hdfs://127.0.0.1:9000</value>  </property>  </configuration> |



* **Chỉnh sửa hdfs-site.xml**

 Chỉnh sửa hdfs-site.xml với “nano hdfs-site.xml”. Tệp này thực sự nằm trong etc / hadoop bên trong thư mục cài đặt Hadoop và thêm cấu hình sau:

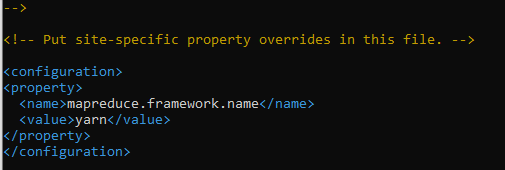
|  |
| --- |
| <configuration>  <property>  <name>dfs.data.dir</name>  <value>/home/hdoop/dfsdata/namenode</value>  </property>  <property>  <name>dfs.data.dir</name>  <value>/home/hdoop/dfsdata/datanode</value>  </property>  <property>  <name>dfs.replication</name>  <value>1</value>  </property>  </configuration> |



* **Chỉnh sửa mapred-site.xml**

Thêm cấu hình sau để thay đổi giá trị MapReduce framework mặc định thành yarn:

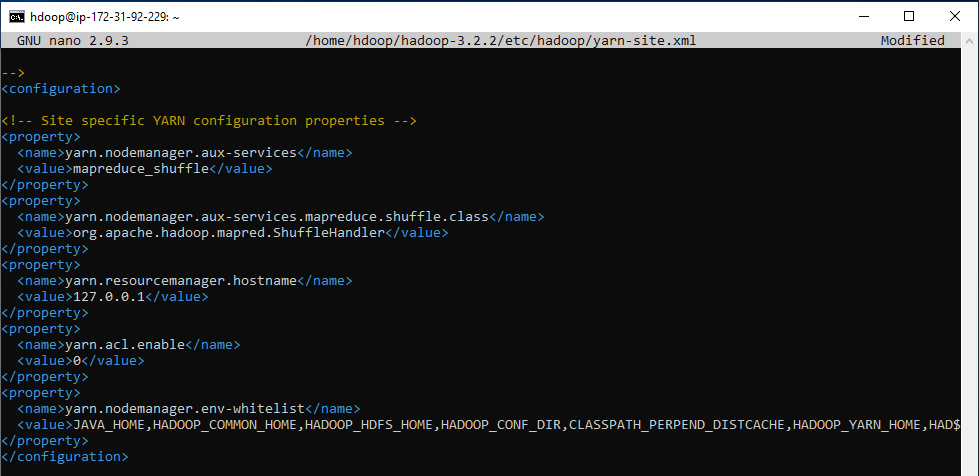
|  |
| --- |
| <configuration>  <property>  <name>mapreduce.framework.name</name>  <value>yarn</value>  </property>  </configuration> |



* **Chỉnh sửa  yarn-site.xml**

Chỉnh sửa yarn-site.xml với “nano yarn-site.xml”. Nó nằm trong etc / hadoop bên trong thư mục cài đặt Hadoop. Cuối cùng, chúng tôi thêm các mục sau:

|  |
| --- |
| <configuration>  <property>  <name>yarn.nodemanager.aux-services</name>  <value>mapreduce\_shuffle</value>  </property>  <property>  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>  <value>org.apache.hadoop.mapred.ShuffleHandler</value>  </property>  <property>  <name>yarn.resourcemanager.hostname</name>  <value>127.0.0.1</value>  </property>  <property>  <name>yarn.acl.enable</name>  <value>0</value>  </property>  <property>  <name>yarn.nodemanager.env-whitelist</name>  <value>JAVA\_HOME,HADOOP\_COMMON\_HOME,HADOOP\_HDFS\_HOME,HADOOP\_CONF\_DIR,CLASSPATH\_PERPEND\_DISTCACHE,HADOOP\_YARN\_HOME,HADOOP\_MAPRED\_HOME</value>  </property>  </configuration> |



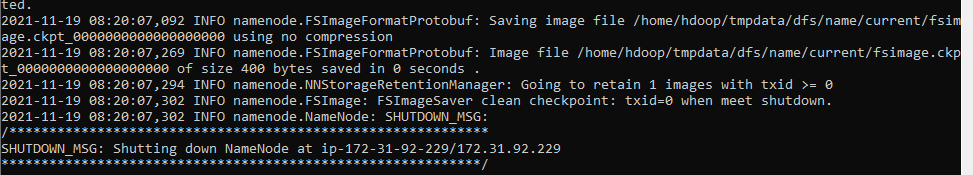
**2.2.4 Start the cluster (cụm)**

*a) Định dạng namenode*

Định dạng namenode trước khi sử dụng nó lần đầu tiên.

|  |
| --- |
| [php] hdfs namenode -format [/ php] |

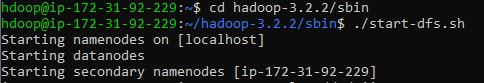
Khi thấy thông báo tắt máy báo hiệu sự kết thúc của quá trình định dạng NameNode.



*b) Khởi động HDFS*

Bắt đầu cụm hadoop bằng cách sử dụng tập lệnh khởi động hadoop.

|  |
| --- |
| [php] ./start-dfs.sh [/ php] |



*c) Khởi động các dịch vụ YARN*

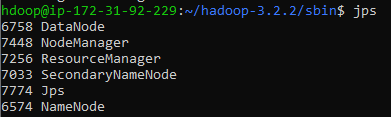
Để khởi động YARN, sử dụng

|  |
| --- |
| [php] ./start-yarn.sh [/ php] |



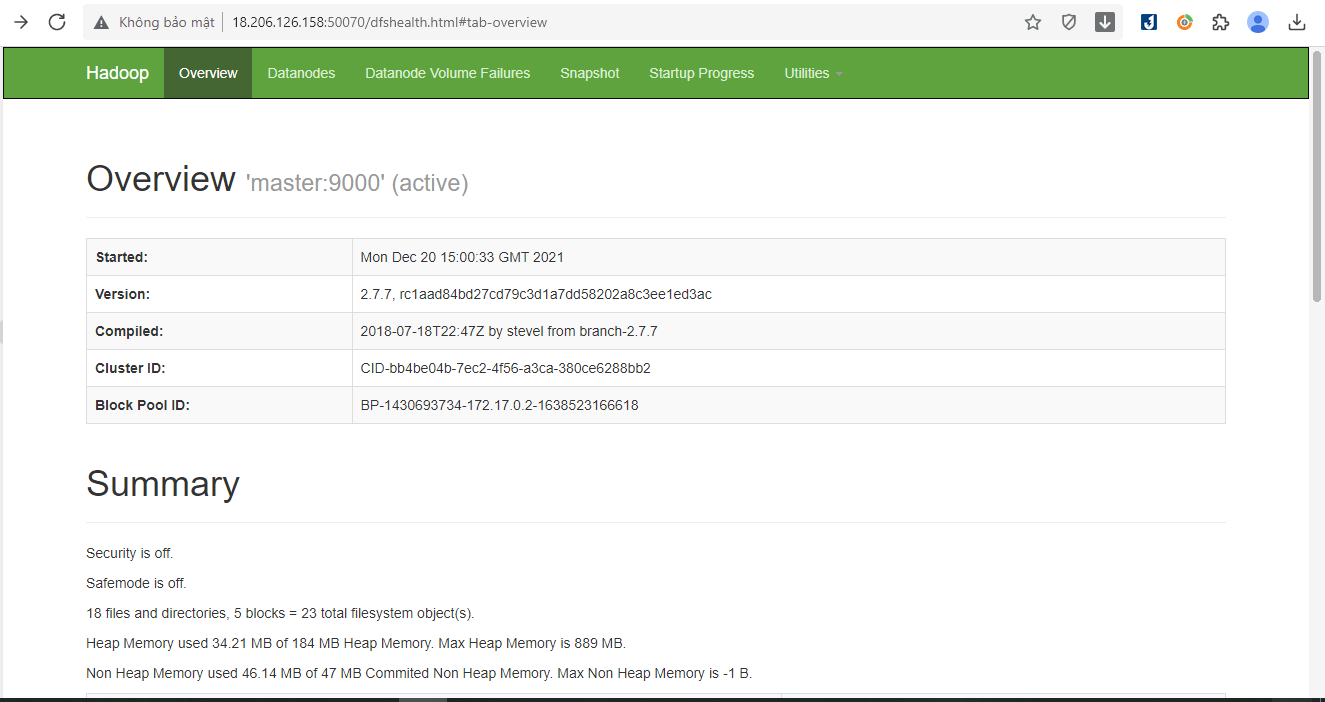
*d) Xác minh nếu tất cả quá trình đã bắt đầu*

|  |
| --- |
| [php] jps |



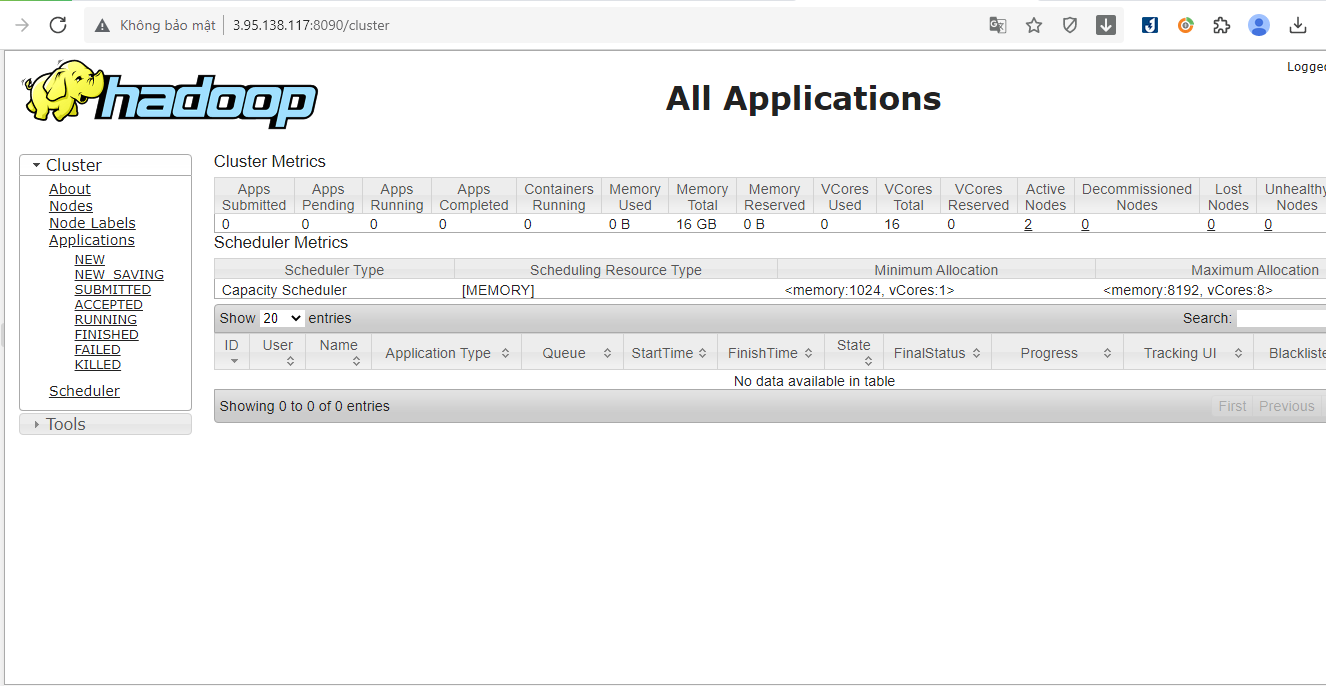
*e) Giao diện web-Để xem giao diện người dùng Web của NameNode*

Truy cập: (http: // localhost: 50070)



*f) Giao diện người dùng quản lý tài nguyên (http: // localhost: 8090)*

Giao diện web sẽ hiển thị tất cả các công việc đang chạy trên thông tin cụm. Do đó, điều này sẽ giúp theo dõi báo cáo tiến độ.



*g) Ngừng các cụm (clusters)*

Để dừng Dịch vụ HDFS, sử dụng

|  |
| --- |
| stop-dfs.sh |

Để ngừng dịch vụ YARN, sử dụng

|  |
| --- |
| [php] stop-yarn.sh [/ php] |

* 1. Chuyển file CSV sang Parquet

Tạo 1 file code python tên convert2.py

|  |
| --- |
| vi convert2.py |



Sau đó thêm code vào file và lưu lại

|  |
| --- |
| import dask.dataframe as dd  df = dd.read\_csv('dataset.csv')  df.to\_parquet('dataset\_parquet', write\_index=False) |

Thực hiện chạy code bằng câu lệnh:

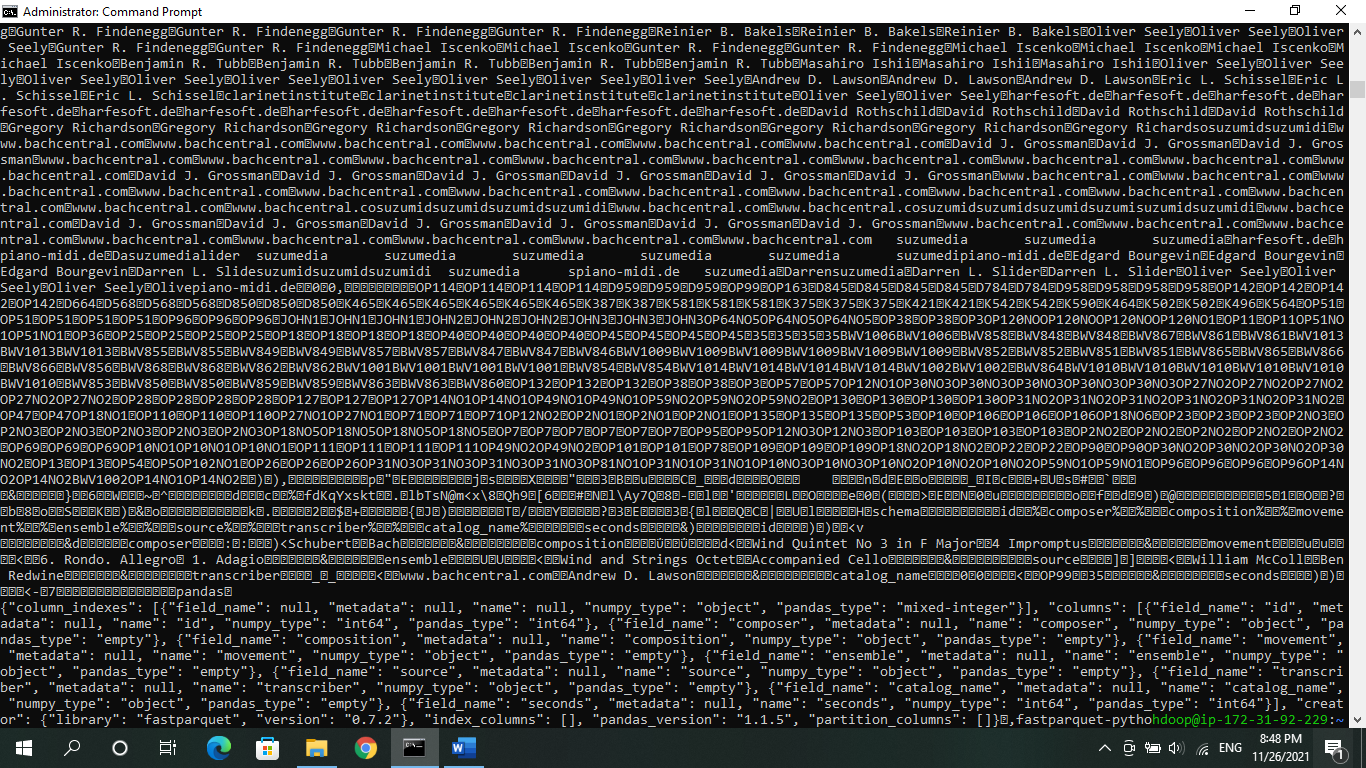
|  |
| --- |
| python3 convert2.py |

Sau đó xem lại tất cả các thư mục thì thấy file csv đã chuyển sang parquet trong thư mục dataset\_parquet

Đặt data vào hdfs và xem file parquet

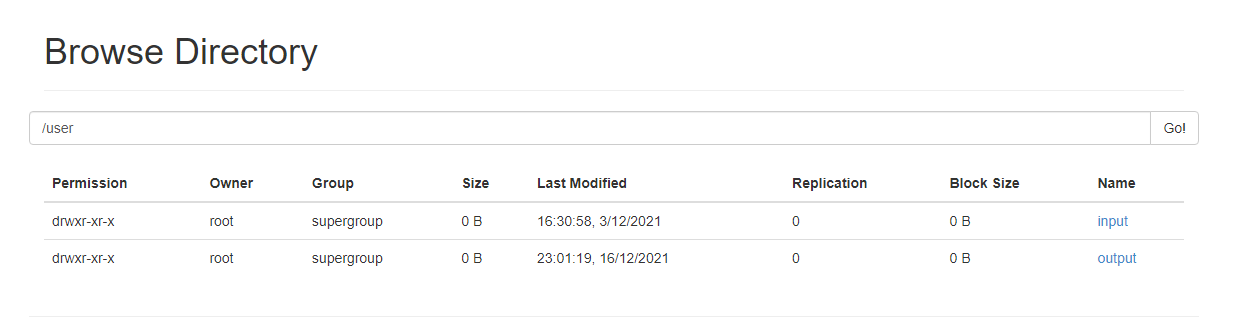
|  |
| --- |
| hdfs dfs -put dataset\_parquet/ ./  hdfs dfs -cat /user/hdoop/dataset\_parquet/part.0.parquet |





* 1. Đọc và ghi file Parquet bằng MapReduce
     1. Tạo nơi lưu trữ data

Tạo 1 thư mục user, trong đó tạo thêm thư mục input và output để chứa dữ liệu đầu vào và đầu ra của bài toán



Sao chép dataset.csv vào thư mục input



Cài một số môi trường cần thiết

|  |
| --- |
| sudo apt update  sudo apt install python3-pip python3-dev  sudo apt install python  sudo apt install jupyter  pip3 install --upgrade --force-reinstall --no-cache-dir jupyter  jupyter notebook password |

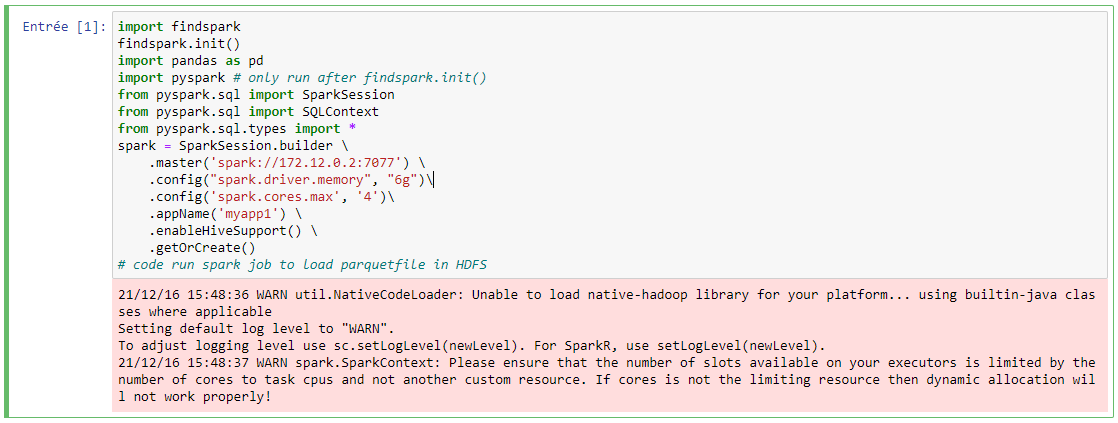
* + 1. Code MapReduce

Truy cập vào http: // localhost: 8070 để vào Jupyter



Code chạy lệnh spark để tải file parquet trong HDFS

|  |
| --- |
| import findspark  findspark.init()  import pandas as pd  import pyspark # only run after findspark.init()  from pyspark.sql import SparkSession  from pyspark.sql import SQLContext  from pyspark.sql.types import \*  spark = SparkSession.builder \  .master('spark://172.12.0.2:7077') \  .config("spark.driver.memory", "6g")\  .config('spark.cores.max', '4')\  .appName('myapp1') \  .enableHiveSupport() \  .getOrCreate()  # code run spark job to load parquetfile in HDFS |

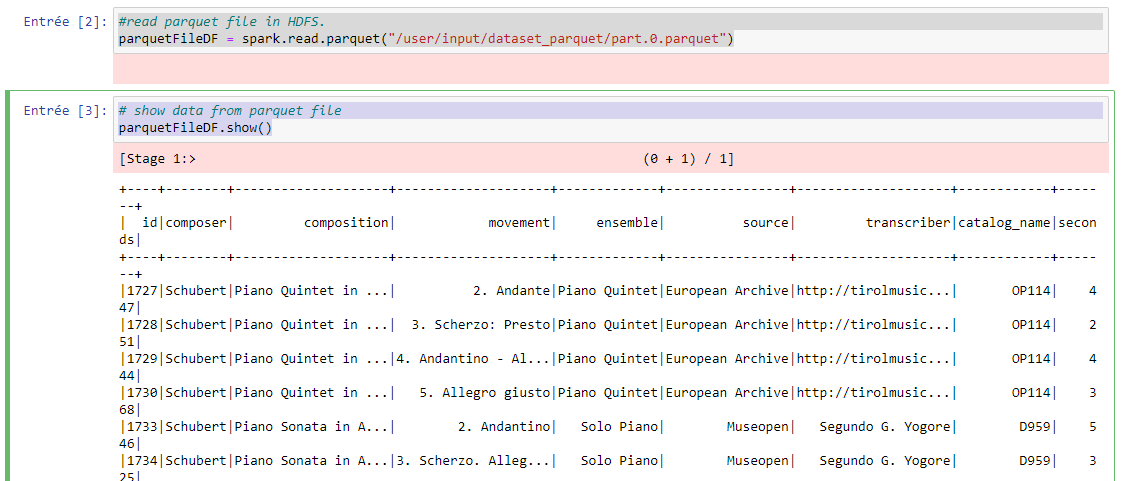


Đọc file parquet trong HDFS

|  |
| --- |
| #read parquet file in HDFS.  parquetFileDF = spark.read.parquet("/user/input/dataset\_parquet/part.0.parquet") |

Xem file parquet

|  |
| --- |
| # show data from parquet file  parquetFileDF.show() |



Hàm map sẽ lấy data của parquet tạo nên 1 cặp key – value, kết quả của hàm sẽ tạo ra 1 dataframe có key là từng tên của composer và value là 1

Hàm reduce sẽ đếm nhưng key nào giống nhau thì sẽ tăng value lên

|  |
| --- |
| # function map : tạo lên 1 cặp key - value: key = 1, value = composer  def map(parquetFileDF):  df=parquetFileDF.toPandas()  map\_data = pd.DataFrame()  for line in range(df.count()[0]):  map\_data.at[line,'composer'] = df.at[line,'composer']  map\_data.at[line,'value'] = 1  return map\_data    def reducer(mapdata):  current\_composer = None  current\_count = 0  composer = None  df\_output = pd.DataFrame()  line=None  for line in range(mapdata.count()[0]):  composer = mapdata.at[line,'composer']  count = mapdata.at[line,'value']  try:  count = int(count)  except ValueError:  # count was not a number, so silently  # ignore/discard this line  continue  if current\_composer == composer:  current\_count += count  else:  if current\_composer:  df\_output.at[line,'composer'] = current\_composer  df\_output.at[line,'count'] = current\_count  current\_count = count  current\_composer = composer  if current\_composer == composer:  df\_output.at[line+1,'composer'] = current\_composer  df\_output.at[line+1,'count'] = current\_count  return df\_output  mapdata\_ = map(parquetFileDF)  mapdata\_ = mapdata\_.sort\_values(by='composer', ascending=True).reset\_index()  print(mapdata\_)  output = reducer(mapdata\_)  print(output.values) |

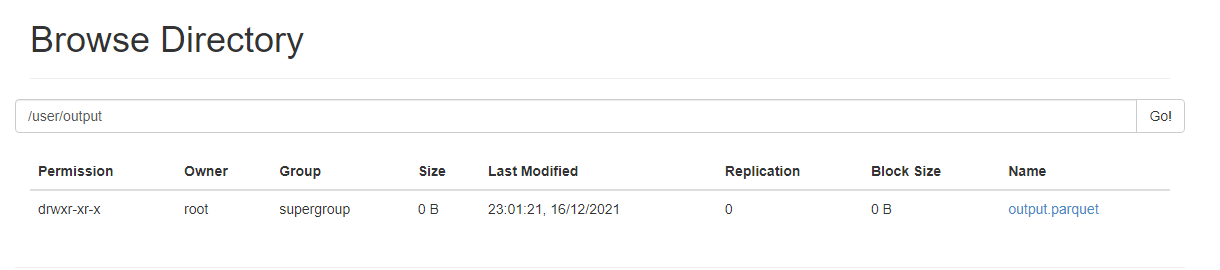


Sau khi xử lý xong sẽ ghi xuống file parquet, mỗi lần ghi phải xóa file cũ để tránh file đã tồn tại

|  |
| --- |
| df1 = spark.createDataFrame(output.astype(str))  df1.write.parquet("/user/output/output.parquet")  # read output  output = spark.read.parquet("/user/output/output.parquet")  output.show() |



Khi chương trình chạy thành công thì file parquet sẽ tự động ghi vào thư mục output



1. PHẦN KẾT LUẬN
   1. Kết quả đạt được

Sau một thời gian nghiên cứu và thực hiện đề tài “***Tìm hiểu Parquet và viết ứng dụng demo***”, nhóm chúng em đã đạt được những kết quả như sau:

* + 1. Kiến thức tìm hiểu được

Những kiến thức quan trọng về Apache Parquet, MapReduce, và tổng quan về Hadoop.

* + 1. Chương trình đã làm được
* Cài đặp Hadoop và kết nối với giao diện người dùng của HDFS.
* Chuyển file CSV sang Parquet bằng python.
* Đọc và Ghi file Parquet bằng MapReduce.
  1. Ưu điểm
* Khi cài đặt xong thì mở các giao diện lên không bị lỗi
* Code chuyển file CSV sang Parquet chính xác
* Đọc ghi file Parquet bằng MapReduce được và trả về đúng thư mục trên giao diện
  1. Nhược điểm
* Để đọc được file Parquet bằng MapReduce thì với mỗi file Parquet phải viết một cái code riêng (giống cấu trúc với nhau) chứ vẫn chưa viết code tổng quát đề xử lý được tất cả các file Parquet
  1. Hướng phát triển
* Tiếp tục nghiên cứu hoàn thiện code MapReduce.
* Xử lý mọi thao tác trên giao diện không cần code.
* Tìm hiểu cách cài đặt chương trình để người khác có thể sử dụng một cách tiện lợi nhất.

1. TÀI LIỆU THAM KHẢO

[1]. SearchDataManagement. 2021. “*What is a Columnar Database?”,* truy cập tại: <https://searchdatamanagement.techtarget.com/definition/columnar-database>, ngày truy cập 20/12/2021.

[2]. Blake Barnhill, Matt David, 2021, “*Row vs Column Oriented Databases”,* truy cập tại:

<https://dataschool.com/data-modeling-101/row-vs-column-oriented-databases/> , ngày truy cập 20/12/2021.

[3]. Databricks, 2021, “*What Is Apache Parquet”,* truy cập tại: <https://databricks.com/glossary/what-is-parquet>, ngày truy cập 20/12/2021.

[4]. Databricks, 2021, “*What Is Apache Parquet*”, truy cập tại: <https://databricks.com/glossary/what-is-parquet> , ngày truy cập 20/12/2021.

[5]. Databricks, 2021, “*What Is Apache Parquet*”, truy cập tại: <https://databricks.com/glossary/what-is-parquet> , ngày truy cập 20/12/2021.

[6]. Talend.com, 2021, “*Mapreduce là gì? Tổng quan mô hình lập trình Mapreduce”,* truy cập tại: <https://www.talend.com/resources/what-is-mapreduce/>, ngày truy cập 20/12/2021.

[7]. P., support, c. and </form>, 2021, “*How to Install Hadoop on Ubuntu 18.04 or 20.04*”, truy cập tại: <https://phoenixnap.com/kb/install-hadoop-ubuntu#ftoc-heading-7> , ngày truy cập 15/12/2021.