**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

✩✩✩✩✩

****

**MÔN HỌC: BIGDATA ANALYSIS**

**BÁO CÁO CUỐI KỲ**

**ĐỀ TÀI: TÌM HIỂU NÂNG CAO VỀ APACHE PIG DEMO PIG KẾT HỢP SPARK**

**GVHD:ThS. LÊ THỊ MINH CHÂU**

**Nhóm thực hiện: Nhóm 12**

|  |  |
| --- | --- |
| **SVTH** | **MSSV** |
| Trì Hoài Lộc | 20133063 |
| Phạm Trung Nghĩa | 20133073 |
| Ngô Hoàng Khắc Tường | 20133112 |
| Nguyễn Thị Tường Vi | 20133113 |

**Lớp thứ 3\_Tiết 12-15**

**Tp.Hồ Chí Minh, tháng 5 năm 2023**

# **ĐÁNH GIÁ & CHẤM ĐIỂM**

|  |  |  |  |
| --- | --- | --- | --- |
| **MSSV** | **THÀNH VIÊN** | **ĐÁNH GIÁ** | **KÝ TÊN** |
| 20133063 | Trì Hoài Lộc |  |  |
| 20133073 | Phạm Trung Nghĩa |  |  |
| 20133112 | Ngô Hoàng Khắc Tường |  |  |
| 20133113 | Nguyễn Thị Tường Vi |  |  |

**NHẬN XÉT:**

*Ký tên*

**ThS. Lê Thị Minh Châu**

# **BẢNG PHÂN CÔNG NHIỆM VỤ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MSSV** | **Họ và tên** | **Phân công** | **Hoàn thành (%)** | **Điểm** |
| 20133063 | Trì Hoài Lộc | Phân tích và xử lý dữ liệu với PIG kết hợp SPARK | 100% |  |
| 20133073 | Phạm Trung Nghĩa | Tìm hiểu vào demo chức năng nâng cao của PIG | 100% |  |
| 20133112 | Ngô Hoàng Khắc Tường | Tìm hiểu vào demo chức năng nâng cao của PIG | 100% |  |
| 2013113 | Nguyễn Thị Tường Vi | Phân tích và xử lý dữ liệu với PIG | 100% |  |

**MỤC LỤC**

TRANG [ĐÁNH GIÁ & CHẤM ĐIỂM 1](#_Toc136364056)

[BẢNG PHÂN CÔNG NHIỆM VỤ 2](#_Toc136364057)

[MỤC LỤC 3](#_Toc136364058)

[LỜI CẢM ƠN 5](#_Toc136364059)

[LỜI MỞ ĐẦU 8](#_Toc136364060)

[Chương 1. GIỚI THIỆU VỀ KIẾN THỨC ÁP DỤNG 9](#_Toc136364061)

[1.1 Apache hadoop 9](#_Toc136364062)

[1.1.1 Khái niệm 9](#_Toc136364063)

[1.1.2 Kiến trúc Hadoop: 9](#_Toc136364064)

[1.1.3 Ưu điểm của Hadoop: 11](#_Toc136364065)

[1.2 Apache Pig 11](#_Toc136364066)

[1.2.1 Khái niệm 11](#_Toc136364067)

[1.2.2 Các chức năng của Pig 12](#_Toc136364068)

[1.2.3 Kiến trúc Apache Pig: 13](#_Toc136364069)

[1.2.4 Quá trình xử lý dữ liệu trong Apache Pig 14](#_Toc136364070)

[1.2.5 Ứng dụng và lợi ích của Apache Pig 14](#_Toc136364071)

[1.3 Apache Spark 16](#_Toc136364072)

[1.3.1 Khái niệm 16](#_Toc136364073)

[1.3.2 Resilient Distributed Datasets - RDD 16](#_Toc136364074)

[1.3.3 Chức năng của Spark 17](#_Toc136364075)

[1.3.4 Các thành phần của Spark 18](#_Toc136364076)

[1.3.5 Ứng dụng của Apache Spark 19](#_Toc136364077)

[Chương 2. TÌM HIỂU VÀ DEMO CHỨC NĂNG NÂNG CAO CỦA PIG 20](#_Toc136364078)

[2.1 Cogroup 20](#_Toc136364079)

[2.2 Rank 20](#_Toc136364080)

[2.3 Cube 21](#_Toc136364081)

[2.4 Assert 22](#_Toc136364082)

[2.5 Tokenrize 23](#_Toc136364083)

[2.6 Concat and Size 25](#_Toc136364084)

[Chương 3. PHÂN TÍCH VÀ XỬ LÝ DỮ LIỆU VỚI PIG 27](#_Toc136364085)

[3.1 Giới thiệu dataset 27](#_Toc136364086)

[3.2 Hadoop trên Window 28](#_Toc136364087)

[3.3 Phân tích và xử lý dữ liệu Beer sử dụng Pig 28](#_Toc136364088)

[3.1.1 Chuẩn bị dữ liệu 28](#_Toc136364089)

[3.1.2 Phân tích từng loại beer được người dùng đánh giá tổng quát cao nhất và nồng độ cồn người dùng thích 30](#_Toc136364090)

[3.1.3 Phân tích số lượng bia có điểm đánh giá tổng quát từ 0-5 34](#_Toc136364091)

[3.1.4 Đếm từng loại beer style có bao nhiêu beer khác nhau 36](#_Toc136364092)

[Chương 4. PHÂN TÍCH VÀ XỬ LÝ DỮ LIỆU VỚI PIG KẾT HỢP VỚI SPARK 40](#_Toc136364093)

[4.1 Xử lý dữ liệu trên PIG: 40](#_Toc136364094)

[4.2 Phân tích dữ liệu bằng SPARK: 41](#_Toc136364095)

[4.2.1 Phân tích từng loại beer được người dùng đánh giá tổng quát cao nhất và nồng độ cồn người dùng thích 41](#_Toc136364096)

[4.2.2 Phân tích số lượng bia có điểm đánh giá tổng quát từ 0-5 43](#_Toc136364097)

[4.2.3 Đếm từng loại beer có bao nhiêu beer khác nhau 44](#_Toc136364098)

[Chương 5. KẾT LUẬN 47](#_Toc136364099)

[TÀI LIỆU THAM KHẢO 48](#_Toc136364100)

# **LỜI CẢM ƠN**

*Đầu tiên, nhóm xin chân thành gửi lời cảm ơn với cô Lê Thị Minh Châu – Giảng viên phụ trách môn BigData Analysis – Trường Đại học Sư phạm kỹ thuật TP.HCM.*

*Trong thời gian học tập, các thành viên trong nhóm nhận được sự nhiều sự giúp đỡ từ Cô. Cô đã cung cấp đầy đủ kiến thức, chỉ bảo và đóng góp những ý kiến quý báu giúp nhóm có thêm hiểu biết để hoàn thành được báo cáo cuối kì cho môn học. Vì một vài sự cố không mong muốn nhưng rất may mắn Cô đã chấp nhận dạy bảo, cung cấp tài liệu và hướng dẫn tận tình trong suốt quá trình học tập. Cô luôn nhiệt tình vui vẻ chỉ dạy, chia sẻ, giúp đỡ nhóm và cả lớp trong suốt quá trình học. Nhóm rất trân quý sự tâm huyết và trách nhiệm của Cô trong công việc giảng dạy và truyền đạt kiến thức.*

*Trong quá trình thực hiện đồ án, dựa trên kiến thức được Cô cung cấp qua các buổi học lý thuyết cũng như thực hành trên lớp, kết hợp với việc tự tìm hiểu những công cụ và kiến thức mới , nhóm đã cố gắng thực hiện đồ án một cách tốt nhất .Tuy nhiên, đồ án còn chưa được hoàn hiện và có thể còn có sai sót.*

*Nhóm rất mong nhận được sự góp ý từ Cô nhằm rút ra những kinh nghiệm quý báu và hoàn thiện vốn kiến thức để nhóm có thể hoàn thành những đồ án, dự án khác trong tương lai .*

***Các thành viên trong nhóm xin chân thành cảm ơn Cô!***

# **LỜI MỞ ĐẦU**

Đề tài *"Tìm hiểu nâng cao về Apache Pig và Demo Pig kết hợp Spark*" được chọn với mục đích tìm hiểu sâu hơn về Apache Pig và khám phá sự phối hợp giữa Pig và Spark, hai công cụ quan trọng trong lĩnh vực xử lý dữ liệu lớn.

Apache Pig là một ngôn ngữ truy vấn cao cấp và một framework dùng để xử lý dữ liệu lớn. Nó cung cấp một cách tiếp cận dễ hiểu và dễ sử dụng để thực hiện các phân tích và biến đổi dữ liệu phức tạp trên các hệ thống phân tán. Việc tìm hiểu nâng cao về Apache Pig cho phép chúng ta khám phá và làm quen với các khái niệm, cú pháp và tính năng mạnh mẽ của Pig để xử lý dữ liệu một cách hiệu quả.

Kết hợp Pig và Spark trong một demo mang lại nhiều lợi ích. Spark là một framework tính toán phân tán được thiết kế để xử lý dữ liệu lớn nhanh chóng và hiệu quả. Khi kết hợp với Pig, chúng ta có thể tận dụng sự mạnh mẽ của Spark để thực hiện các phân tích và xử lý dữ liệu lớn theo cách phân tán và song song. Điều này mở ra cơ hội khám phá và áp dụng các kỹ thuật phân tích dữ liệu phức tạp và đa dạng, từ các phép biến đổi đơn giản đến các phép tính toán phức tạp và thuật toán machine learning.

Việc nghiên cứu và thực hiện demo Pig kết hợp Spark cũng mang lại sự phát triển kỹ năng và kiến thức trong lĩnh vực xử lý dữ liệu lớn. Chúng ta sẽ có cơ hội trải nghiệm và tìm hiểu về cách sử dụng Pig và Spark để xử lý dữ liệu, từ việc truy vấn dữ liệu đến phân tích và trình bày kết quả. Đồng thời, việc nắm vững Pig và Spark cũng là một lợi thế trong việc tham gia vào các dự án Big Data và phát triển ứng dụng xử lý dữ liệu lớn trong thực tế.

# **Chương 1. GIỚI THIỆU VỀ** **KIẾN THỨC ÁP DỤNG**

## **Apache hadoop**

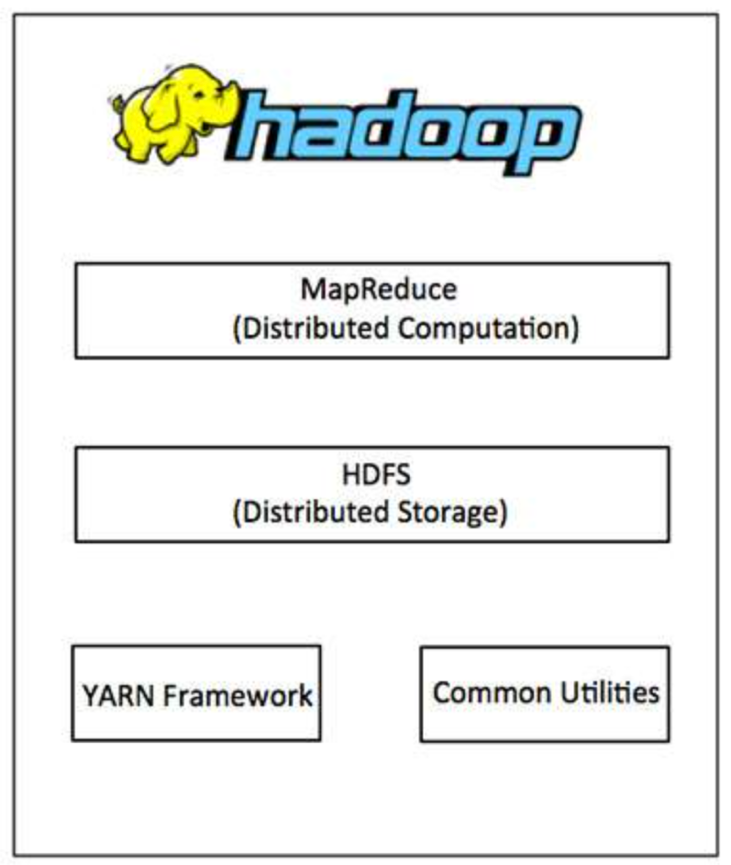
### **Khái niệm**

Hadoop là một Apache open source framework được viết bằng java cho phép xử lý phân tán các bộ dữ liệu lớn trên các cụm máy tính bằng cách sử dụng các mô hình lập trình đơn giản. Ứng dụng Hadoop framework hoạt động trong một môi trường cung cấp lưu trữ và tính toán phân tán trên các cụm máy tính. Hadoop được thiết kế để mở rộng quy mô từ một máy chủ thành hàng nghìn máy, mỗi máy cung cấp khả năng tính toán và lưu trữ cục bộ.

### **Kiến trúc Hadoop:**

Có hai lớp chính

* Processing/Computation layer (MapReduce)
* Storage layer (Hadoop Distributed File System).



**MapReduce** là một mô hình lập trình song song để viết các ứng dụng phân tán do Google phát triển nhằm xử lý hiệu quả lượng dữ liệu lớn (tập dữ liệu nhiều terabyte), trên các cụm lớn (hàng nghìn nút) của phần cứng thương mại theo cách đáng tin cậy, có khả năng chịu lỗi. Chương trình MapReduce chạy trên Hadoop, một khung mã nguồn mở của Apache.

**Hệ thống tệp phân tán Hadoop (HDFS)** dựa trên Hệ thống tệp của Google (GFS) và cung cấp một hệ thống tệp phân tán được thiết kế để chạy trên phần cứng thương mại. Nó có nhiều điểm tương đồng với các hệ thống tệp phân tán hiện có. Tuy nhiên, sự khác biệt so với các hệ thống tệp phân tán khác là rất đáng kể. Nó có khả năng chịu lỗi cao và được thiết kế để triển khai trên phần cứng giá rẻ. Nó cung cấp quyền truy cập thông lượng cao vào dữ liệu ứng dụng và phù hợp với các ứng dụng có bộ dữ liệu lớn.

Ngoài hai thành phần cốt lõi nêu trên, Hadoop framework còn bao gồm hai mô-đun sau:

* Hadoop Common: Đây là các thư viện và tiện ích Java được yêu cầu bởi các mô-đun Hadoop khác.
* Hadoop YARN: Đây là một khung để lập lịch công việc và quản lý tài nguyên cụm.

Hadoop chạy mã trên một cụm máy tính. Quá trình này bao gồm các tác vụ cốt lõi sau mà Hadoop thực hiện:

* Dữ liệu ban đầu được chia thành các thư mục và tệp. Các tệp được chia thành các khối có kích thước thống nhất là 128M và 64M (tốt nhất là 128M).
* Các tệp này sau đó được phân phối trên các nút cụm khác nhau để xử lý thêm.
* HDFS, đứng đầu hệ thống tệp cục bộ, giám sát quá trình xử lý.
* Các khối được sao chép để xử lý lỗi phần cứng.
* Kiểm tra xem mã đã được thực thi thành công chưa.
* Việc thực hiện sắp xếp diễn ra giữa các giai đoạn map và reduce.
* Gửi dữ liệu đã sắp xếp đến một máy tính nhất định.
* Viết nhật ký gỡ lỗi cho từng công việc.

### **Ưu điểm của Hadoop:**

* Hadoop cho phép người dùng viết và kiểm tra các hệ thống phân tán một cách nhanh chóng. Phân phối dữ liệu hiệu quả, tự động và có thể hoạt động trên nhiều máy khác nhau, sử dụng tính song song cơ bản của các lõi CPU.
* Hadoop không dựa vào phần cứng để cung cấp khả năng chịu lỗi và tính sẵn sàng cao (FTHA), mà chính thư viện Hadoop đã được thiết kế để phát hiện và xử lý lỗi ở lớp ứng dụng.
* Server có thể được thêm hoặc xóa khỏi cụm một cách linh hoạt và Hadoop tiếp tục hoạt động mà không bị gián đoạn.
* Một lợi thế lớn khác của Hadoop là ngoài việc là nguồn mở, nó còn tương thích trên tất cả các nền tảng vì nó dựa trên Java.

## **Apache Pig**

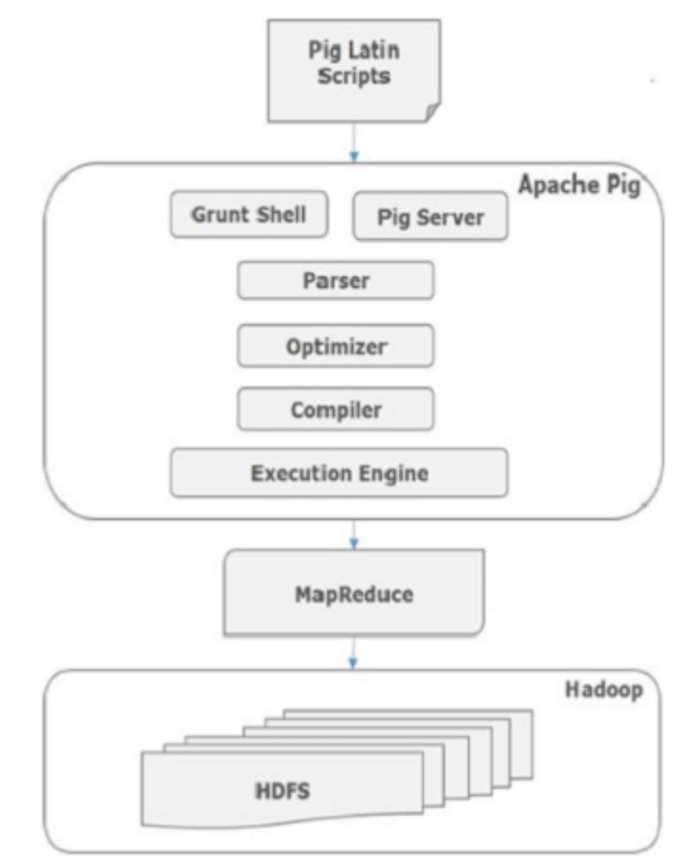
### **Khái niệm**

* Apache Pig là một sự trừu tượng hóa trên MapReduce. Nó là một công cụ/nền tảng được sử dụng để phân tích các tập dữ liệu lớn hơn biểu thị dưới dạng các luồng dữ liệu. Pig thường được sử dụng với Hadoop; chúng ta có thể thực hiện tất cả các thao tác thao tác dữ liệu trong Hadoop bằng Apache Pig.
* Để viết các chương trình phân tích dữ liệu, Pig cung cấp một ngôn ngữ cấp cao được gọi là Pig Latin. Ngôn ngữ này cung cấp nhiều toán tử khác nhau mà người lập trình có thể phát triển các hàm riêng để đọc, ghi và xử lý dữ liệu.
* Để phân tích dữ liệu bằng Apache Pig, lập trình viên cần viết script bằng ngôn ngữ Pig Latin. Tất cả các tập lệnh này được chuyển đổi nội bộ thành các tác vụ map và reduce. Apache Pig có một thành phần được gọi là Pig Engine chấp nhận các tập lệnh Pig Latin làm đầu vào và chuyển đổi các tập lệnh đó thành các công việc MapReduce.
* Sử dụng Pig Latin, các lập trình viên có thể thực hiện các tác vụ MapReduce một cách dễ dàng mà không cần phải nhập mã phức tạp trong Java.
* Apache Pig sử dụng cách tiếp cận đa truy vấn, do đó giảm độ dài của mã.
* Pig Latin là ngôn ngữ giống như SQL và thật dễ dàng để học Apache Pig khi bạn đã quen thuộc với SQL.
* Apache Pig cung cấp nhiều toán tử tích hợp để hỗ trợ các thao tác dữ liệu như nối, lọc, sắp xếp, v.v. Ngoài ra, nó cũng cung cấp các loại dữ liệu lồng nhau tuples, bags, and maps bị thiếu trong MapReduce.

### **Các chức năng của Pig**

* Tập hợp các toán tử phong phú – Nó cung cấp nhiều toán tử để thực hiện các thao tác như nối, sắp xếp, sắp xếp, v.v.
* Dễ lập trình − Pig Latin tương tự như SQL và thật dễ dàng để viết một tập lệnh Pig nếu bạn giỏi SQL.
* Cơ hội tối ưu hóa − Các tác vụ trong Apache Pig tự động tối ưu hóa quá trình thực thi của chúng, vì vậy các lập trình viên chỉ cần tập trung vào ngữ nghĩa của ngôn ngữ.
* Khả năng mở rộng − Sử dụng các toán tử hiện có, người dùng có thể phát triển các chức năng của riêng mình để đọc, xử lý và ghi dữ liệu.
* UDF's − Pig cung cấp phương tiện để tạo Hàm do người dùng xác định bằng các ngôn ngữ lập trình khác như Java và gọi hoặc nhúng chúng vào Tập lệnh Pig.
* Xử lý tất cả các loại dữ liệu − Apache Pig phân tích tất cả các loại dữ liệu, cả có cấu trúc cũng như không có cấu trúc. Nó lưu trữ kết quả trong HDFS.

### **Kiến trúc Apache Pig:**



* Trình phân tích cú pháp (Parser): Ban đầu, Pig Scripts được xử lý bởi Parser. Nó kiểm tra cú pháp của tập lệnh, kiểm tra kiểu và các kiểm tra linh tinh khác. Đầu ra của trình phân tích cú pháp sẽ là một DAG (đồ thị tuần hoàn có hướng), đại diện cho các câu lệnh Pig Latin và các toán tử logic.Trong DAG, các toán tử logic của tập lệnh được biểu diễn dưới dạng các nút và luồng dữ liệu được biểu diễn dưới dạng các cạnh.
* Trình tối ưu hóa(Optimizer):Kế hoạch logic (DAG) được chuyển đến trình tối ưu hóa logic, thực hiện các tối ưu hóa logic như phép chiếu và đẩy xuống.
* Trình biên dịch(Compiler): Trình biên dịch biên dịch kế hoạch logic được tối ưu hóa thành một loạt các công việc MapReduce.
* Công cụ thực thi(Execution engine):Cuối cùng các công việc MapReduce được gửi tới Hadoop theo thứ tự được sắp xếp;các công việc MapReduce này được thực thi trên Hadoop để tạo ra kết quả mong muốn.

### **Quá trình xử lý dữ liệu trong Apache Pig**

* Tải dữ liệu (Load Data): Dữ liệu được tải vào Apache Pig từ các nguồn khác nhau bằng cách sử dụng toán tử LOAD. Apache Pig hỗ trợ nhiều định dạng dữ liệu và có thể đọc dữ liệu từ HDFS, tệp tin thông thường, hay các hệ thống lưu trữ khác.
* Biến đổi dữ liệu (Transform Data): Dữ liệu được biến đổi và chuẩn bị cho các phép tính và phân tích tiếp theo. Apache Pig cung cấp các toán tử như FILTER, GROUP, FOREACH và JOIN để thực hiện các phép biến đổi này. Các phép biến đổi này giúp xử lý và sắp xếp dữ liệu theo yêu cầu của người dùng.
* Sử dụng các hàm do người dùng tự định nghĩa (User-Defined Functions - UDFs): Apache Pig cho phép sử dụng các hàm do người dùng tự định nghĩa để thực hiện các phép tính tùy chỉnh trên dữ liệu. Người dùng có thể tạo ra và sử dụng UDFs để mở rộng khả năng xử lý dữ liệu của Pig và thực hiện các phép tính phức tạp.
* Tối ưu hóa truy vấn (Optimize Query): Trước khi thực hiện truy vấn, Pig sử dụng bộ tối ưu hóa (Optimizer) để cải thiện hiệu suất và kế hoạch thực thi của truy vấn. Bộ tối ưu hóa áp dụng các quy tắc và phương pháp tối ưu hóa để tối đa hiệu suất và giảm thiểu dữ liệu di chuyển, giúp tăng tốc độ xử lý dữ liệu.
* Thực thi truy vấn (Execute Query): Sau quá trình tối ưu hóa, truy vấn được thực hiện bằng cách sử dụng Execution Engine, như MapReduce hoặc Tez. Execution Engine chịu trách nhiệm phân chia công việc và thực thi truy vấn trên các nút xử lý trong cụm xử lý phân tán.
* Lưu trữ kết quả (Store Results): Kết quả của truy vấn Pig Latin có thể được lưu trữ vào HDFS hoặc các hệ thống lưu trữ khác bằng toán tử STORE. Điều này cho phép lưu trữ và truy cập lại kết quả sau quá trình xử lý dữ liệu.

### **Ứng dụng và lợi ích của Apache Pig**

* Xử lý dữ liệu lớn: Apache Pig được thiết kế để xử lý dữ liệu lớn có kích thước từ gigabytes đến petabytes. Nó hoạt động trên các cụm xử lý phân tán như Hadoop, cho phép xử lý dữ liệu song song trên nhiều máy tính, giúp tăng tốc độ xử lý và hiệu quả.
* Linh hoạt và dễ sử dụng: Apache Pig sử dụng ngôn ngữ truy vấn Pig Latin, có cú pháp đơn giản và dễ hiểu. Người dùng không cần phải viết mã phức tạp như trong các ngôn ngữ lập trình khác, giúp tăng tốc độ phát triển và giảm thời gian triển khai.
* Tích hợp với hệ sinh thái Hadoop: Apache Pig tích hợp tốt với các công cụ và hệ thống khác trong hệ sinh thái Hadoop như HDFS, MapReduce, Hive và HBase. Điều này cho phép người dùng tận dụng tối đa các công cụ và khả năng của Hadoop để xử lý và phân tích dữ liệu.
* Xử lý dữ liệu không cấu trúc: Apache Pig hỗ trợ xử lý dữ liệu không cấu trúc và dữ liệu có cấu trúc. Người dùng có thể xử lý các định dạng dữ liệu như CSV, JSON, Avro và XML, cũng như dữ liệu cấu trúc như bảng và dòng cột.
* Tích hợp với ngôn ngữ lập trình khác: Apache Pig cung cấp các API để tích hợp với ngôn ngữ lập trình khác như Java, Python và JavaScript. Điều này cho phép các nhà phát triển sử dụng ngôn ngữ ưa thích của họ để tạo ra các ứng dụng phức tạp và mở rộng khả năng của Pig.
* Tối ưu hóa hiệu suất: Apache Pig cung cấp công cụ tối ưu hóa để cải thiện hiệu suất thực thi truy vấn. Các truy vấn Pig Latin có thể được tối ưu và thực thi song song trên cụm xử lý phân tán, giúp tăng tốc độ xử lý dữ liệu.
* Apache Pig cho phép mở rộng các chức năng bằng cách tạo và sử dụng các hàm do người dùng tự định nghĩa (User-Defined Functions - UDFs). Người dùng có thể viết mã tùy chỉnh để thực hiện các phép tính phức tạp và xử lý dữ liệu theo yêu cầu cụ thể của họ. Điều này giúp Apache Pig linh hoạt và mở rộng để đáp ứng các yêu cầu phân tích dữ liệu đa dạng.
* Xử lý dữ liệu đa dạng: Apache Pig hỗ trợ xử lý dữ liệu đa dạng, bao gồm cả dữ liệu có cấu trúc và không cấu trúc. Bạn có thể thực hiện các phép tính trên các tập dữ liệu có cấu trúc như bảng và dòng cột, cũng như truy vấn và phân tích dữ liệu không cấu trúc như văn bản, log hoặc JSON.
* Tích hợp với công cụ phân tích dữ liệu: Apache Pig có tích hợp với các công cụ phân tích dữ liệu phổ biến khác như Apache Hive, Apache HBase và Apache Spark. Điều này cho phép người dùng kết hợp sức mạnh của các công cụ này để thực hiện các phân tích phức tạp trên dữ liệu lớn.
* Tính khả chuyển và tái sử dụng: Apache Pig cho phép người dùng viết mã Pig Latin và lưu trữ chúng thành các kịch bản có thể thực thi lại. Điều này giúp tăng tính khả chuyển và tái sử dụng mã xử lý dữ liệu, giúp tiết kiệm thời gian và công sức trong quá trình phát triển ứng dụng.

## **Apache Spark**

### **Khái niệm**

* Apache Spark là công nghệ điện toán cụm siêu nhanh, được thiết kế để tính toán nhanh. Nó dựa trên Hadoop MapReduce và nó mở rộng mô hình MapReduce để sử dụng nó một cách hiệu quả cho nhiều loại tính toán hơn, bao gồm các truy vấn tương tác và xử lý luồng. Tính năng chính của Spark là tính toán cụm trong bộ nhớ giúp tăng tốc độ xử lý của ứng dụng.
* Spark được thiết kế để đáp ứng nhiều khối lượng công việc như ứng dụng hàng loạt, thuật toán lặp, truy vấn tương tác và phát trực tuyến. Ngoài việc hỗ trợ tất cả khối lượng công việc này trong một hệ thống tương ứng, nó còn giảm bớt gánh nặng quản lý trong việc duy trì các công cụ riêng biệt.

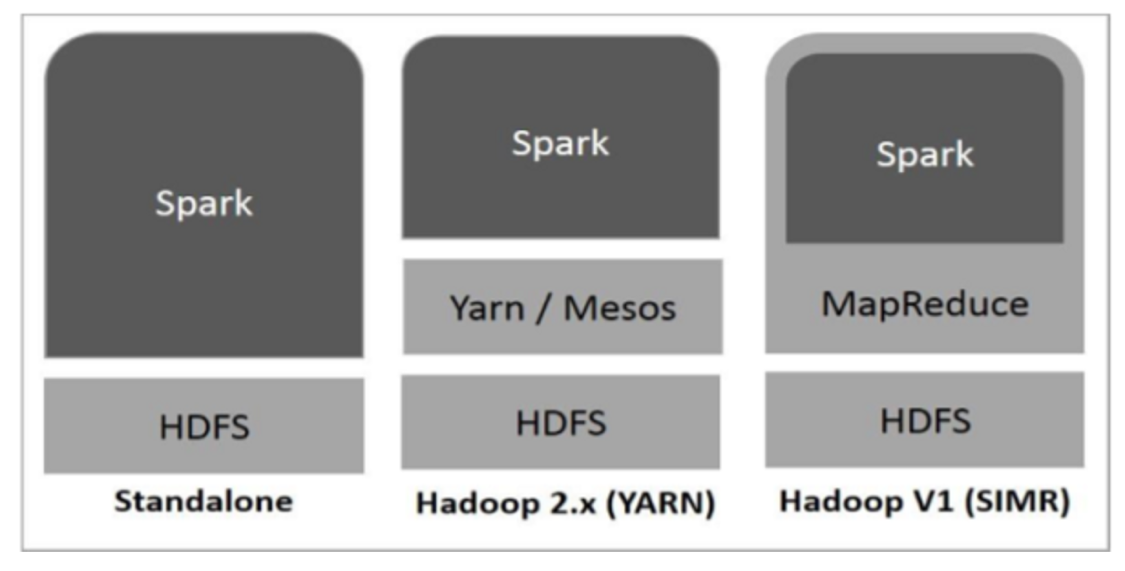
### **Resilient Distributed Datasets - RDD**

* Resilient Distributed Datasets là một cấu trúc dữ liệu cơ bản của Spark. Nó là một tập hợp bất biến phân tán của một đối tượng, hỗ trợ tínhntoán xử lý trong bộ nhớ. Mỗi nhóm dữ liệu đối tượng trong RDD được chia ra thành nhiều phần vùng vật lý. Có thể được tính toán trên các nút khác nhau của một cụm máy chủ (cluster).
* RDDs có thể chứa bất kỳ kiểu dữ liệu nào của Python, Java, hoặc đối tượng Scala, bao gồm các kiểu dữ liệu do người dùng định nghĩa. Thông thường, RDD chỉ cho phép đọc, phân mục tập hợp của các bản ghi. RDDs có thể được tạo ra qua điều khiển xác định trên dữ liệu trong bộ nhớ hoặc RDDs, RDD là một tập hợp có khả năng chịu lỗi mỗi thành phần có thể được tính toán song song.

### **Chức năng của Spark**

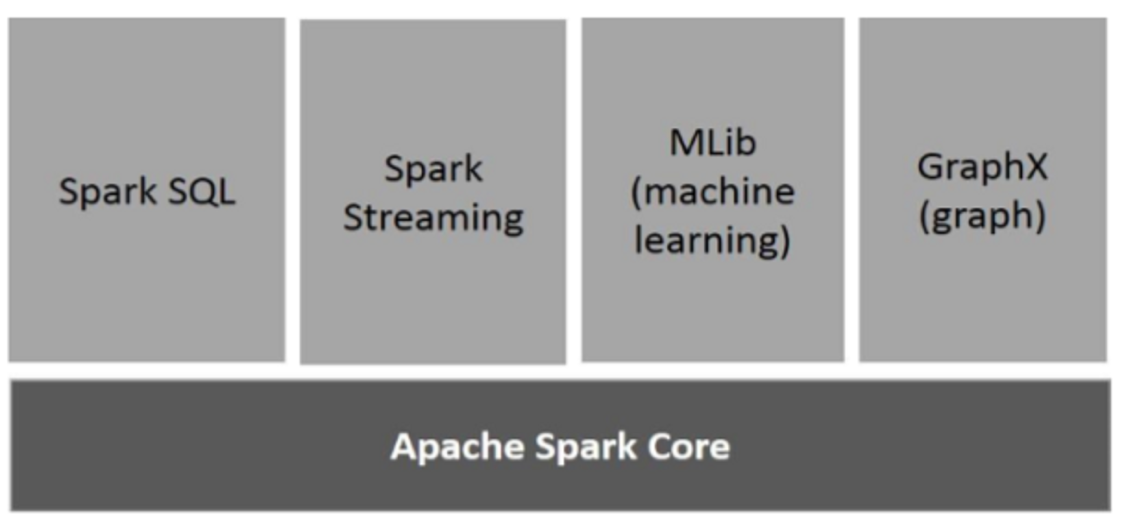
* Tốc độ - Spark giúp chạy một ứng dụng trong cụm Hadoop, nhanh hơn tới 100 lần trong bộ nhớ và nhanh hơn 10 lần khi chạy trên đĩa. Điều này có thể thực hiện được bằng cách giảm số thao tác đọc/ghi vào đĩa. Nó lưu trữ dữ liệu xử lý trung gian trong bộ nhớ.
* Hỗ trợ nhiều ngôn ngữ − Spark cung cấp API tích hợp trong Java, Scala hoặc Python. Do đó, bạn có thể viết các ứng dụng bằng các ngôn ngữ khác nhau. Spark đưa ra 80 toán tử cấp cao để truy vấn tương tác.
* Phân tích nâng cao − Spark không chỉ hỗ trợ 'map' và 'reduce'. Nó cũng hỗ trợ các truy vấn SQL, Streaming data, Học máy (ML) và thuật toán Đồ thị.

Sơ đồ sau đây cho thấy ba cách về cách Spark có thể được xây dựng với các thành phần Hadoop.

****

* Standalone – Spark: Triển khai độc lập có nghĩa là Spark chiếm vị trí trên cùng của HDFS (Hệ thống tệp phân tán Hadoop) và không gian được phân bổ rõ ràng cho HDFS. Tại đây, Spark và MapReduce sẽ chạy cạnh nhau để bao gồm tất cả các spark jobs trên cụm.
* Hadoop Yarn: Việc triển khai Hadoop Yarn có nghĩa đơn giản là spark chạy trên Yarn mà không cần cài đặt trước hoặc yêu cầu quyền truy cập root. Nó giúp tích hợp Spark vào hệ sinh thái Hadoop hoặc ngăn xếp Hadoop. Nó cho phép các thành phần khác chạy trên đỉnh ngăn xếp.
* Spark trong MapReduce (SIMR)được sử dụng để khởi chạy spark job bên cạnh việc triển khai độc lập. Với SIMR, người dùng có thể khởi động Spark và sử dụng shell của nó mà không cần bất kỳ quyền truy cập quản trị nào.

### **Các thành phần của Spark**



* Spark Core là công cụ thực thi chung cơ bản cho nền tảng spark mà tất cả các chức năng khác được xây dựng trên đó. Nó cung cấp bộ dữ liệu tham chiếu và tính toán trong bộ nhớ trong các hệ thống lưu trữ bên ngoài.
* Spark SQL là một thành phần trên Spark Core giới thiệu một bản tóm tắt dữ liệu mới có tên là SchemaRDD, cung cấp hỗ trợ cho dữ liệu có cấu trúc và bán cấu trúc.
* Spark Streaming tận dụng khả năng lập lịch nhanh của Spark Core để thực hiện phân tích phát trực tuyến. Nó nhập dữ liệu theo các lô nhỏ và thực hiện các phép biến đổi RDD (Bộ dữ liệu phân tán đàn hồi) trên các lô dữ liệu nhỏ đó.
* MLlib là một khung học máy phân tán phía trên Spark do kiến trúc Spark dựa trên bộ nhớ phân tán. Theo điểm chuẩn, nó được thực hiện bởi các nhà phát triển MLlib dựa trên các triển khai xen kẽ bình phương tối thiểu (ALS). Spark MLlib nhanh gấp 9 lần so với phiên bản Apache Mahout dựa trên đĩa Hadoop (trước khi Mahout có giao diện Spark).
* GraphX là một khung xử lý đồ thị phân tán trên Spark. Nó cung cấp một API để thể hiện tính toán biểu đồ có thể mô hình hóa các biểu đồ do người dùng xác định bằng cách sử dụng API trừu tượng hóa Pregel. Nó cũng cung cấp một thời gian chạy được tối ưu hóa cho sự trừu tượng hóa này.

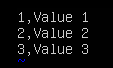
### **Ứng dụng của Apache Spark**

* Xử lý dữ liệu hàng tỷ bản ghi: Spark cho phép xử lý và phân tích dữ liệu lớn một cách hiệu quả và nhanh chóng. Với khả năng tính toán phân tán và tương thích với nhiều nguồn dữ liệu, Spark có thể xử lý hàng tỷ bản ghi trong thời gian ngắn.
* Phân tích dữ liệu thời gian thực: Spark Streaming là một thành phần của Spark cho phép xử lý dữ liệu dòng (streaming data) một cách liên tục và thời gian thực. Điều này cho phép phân tích và đưa ra quyết định trong thời gian gần như thời instant trên dữ liệu đang được truyền đến.
* Xử lý và phân tích dữ liệu đa cấu trúc: Spark hỗ trợ xử lý dữ liệu đa cấu trúc như dữ liệu JSON, dữ liệu XML và dữ liệu CSV. Điều này giúp đơn giản hóa quá trình xử lý và phân tích dữ liệu từ các nguồn đa dạng.
* Machine Learning: Spark MLlib là một thư viện machine learning tích hợp trong Spark, cung cấp các thuật toán và công cụ để thực hiện các nhiệm vụ machine learning như phân loại, gom nhóm, hồi quy và khai phá dữ liệu. Spark MLlib hỗ trợ xử lý dữ liệu lớn và tính toán phân tán, giúp tăng tốc quá trình huấn luyện và triển khai các mô hình machine learning.
* Xử lý đồ thị: Spark GraphX là một thành phần của Spark cho phép xử lý và phân tích đồ thị lớn. Điều này có thể áp dụng cho các ứng dụng như phân tích mạng xã hội, phân tích mạng giao thông, và phân tích mạng điện.
* Xử lý dữ liệu trên đám mây: Spark có tích hợp sẵn với các dịch vụ đám mây như Amazon Web Services (AWS), Microsoft Azure và Google Cloud Platform (GCP). Điều này cho phép xử lý dữ liệu trực tiếp trên các nền tảng đám mây mà không cần phải triển khai cơ sở hạ tầng riêng biệt.

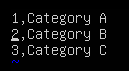
# **Chương 2. TÌM HIỂU VÀ DEMO CHỨC NĂNG NÂNG CAO CỦA PIG**

## **Cogroup**

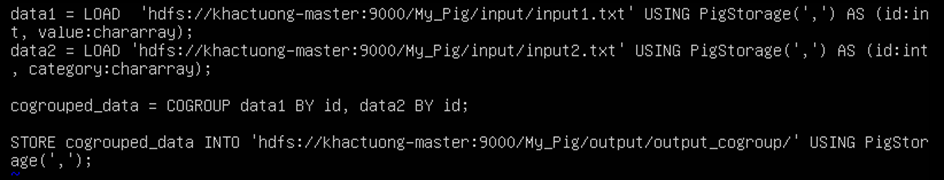
* Dùng để gom nhóm và kết hợp dữ liệu từ nhiều nguồn dữ liệu dựa trên các cột chung.
* Cú pháp : [ cogrouped\_data = COGROUP data1 BY col1, data2 BY col1;]
* Ví dụ :
  + Input
    - Input1.txt



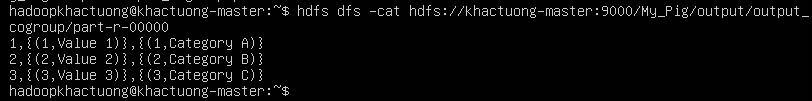
* + - Input2.txt



* + Code

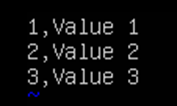


* Output

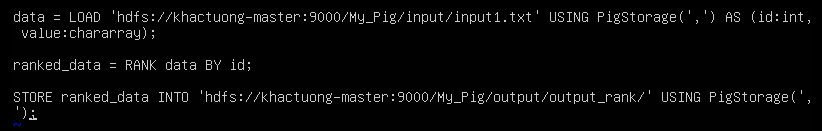


## **Rank**

* Dùng để đánh chỉ mục các bản ghi trong một tập dữ liệu dựa trên một hoặc nhiều cột.
* Cú pháp: [ranked\_data = RANK data BY col1;]
* Ví dụ:
  + Input
    - Input1.txt



* + Code

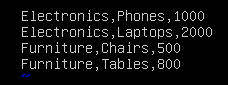


* + Output

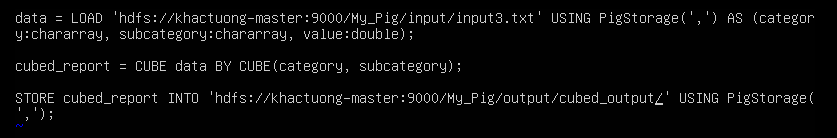


## **Cube**

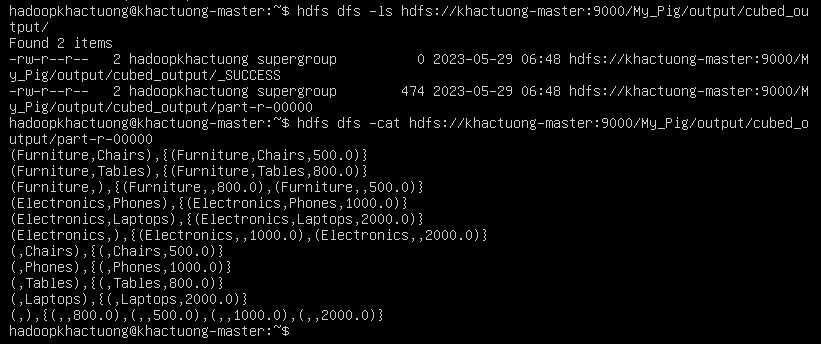
* Apache Pig hỗ trợ câu lệnh CUBE để tạo các báo cáo tóm tắt dữ liệu theo các chiều khác nhau. Câu lệnh này hữu ích khi thực hiện phân tích dữ liệu đa chiều.
* Cú pháp: [cubed\_data = CUBE data BY CUBE(col1, col2);]
* Ví dụ:
  + Input
    - Input3.txt



* + Code

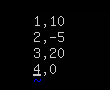


* + Output

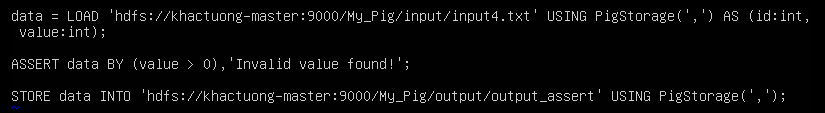


## **Assert**

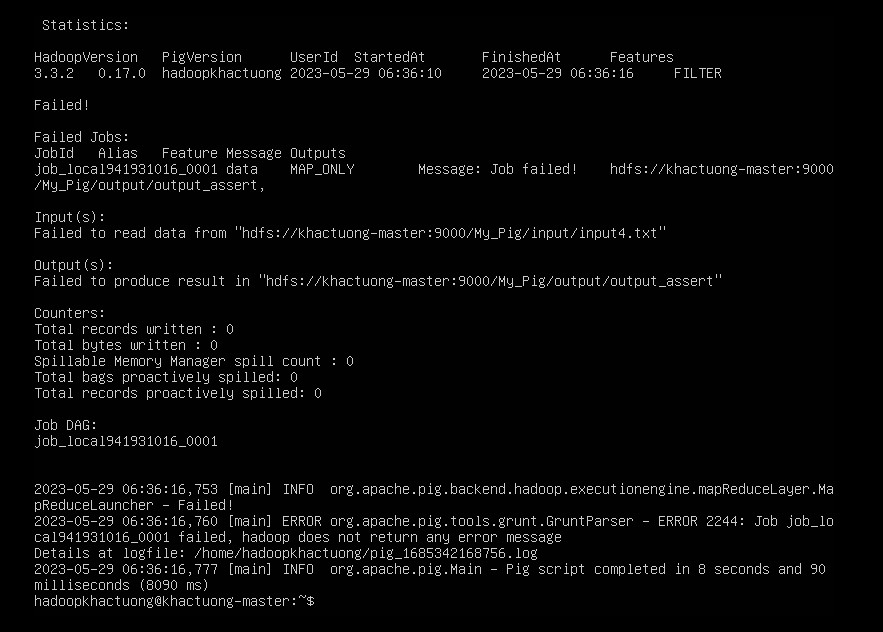
* Trong Apache Pig, câu lệnh assert được sử dụng để kiểm tra tính đúng đắn của dữ liệu và quyết định liệu một điều kiện nào đó là đúng hay sai. Nếu điều kiện được xác định trong câu lệnh assert là sai, Pig sẽ ngừng thực thi và in ra thông báo lỗi tương ứng.
* Cú pháp : [assert <condition> [error\_message];]
* Ví dụ:
  + Input
    - Input4.txt



* + Code

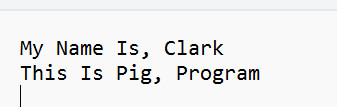


* + Output

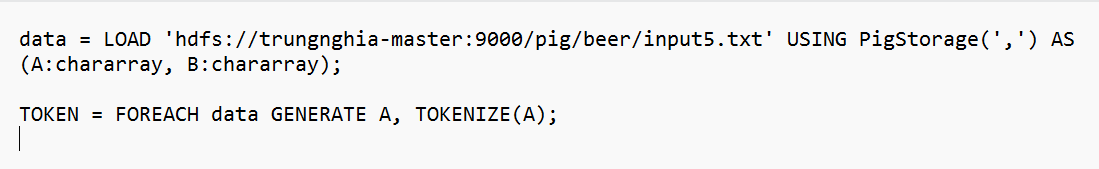


## **Tokenrize**

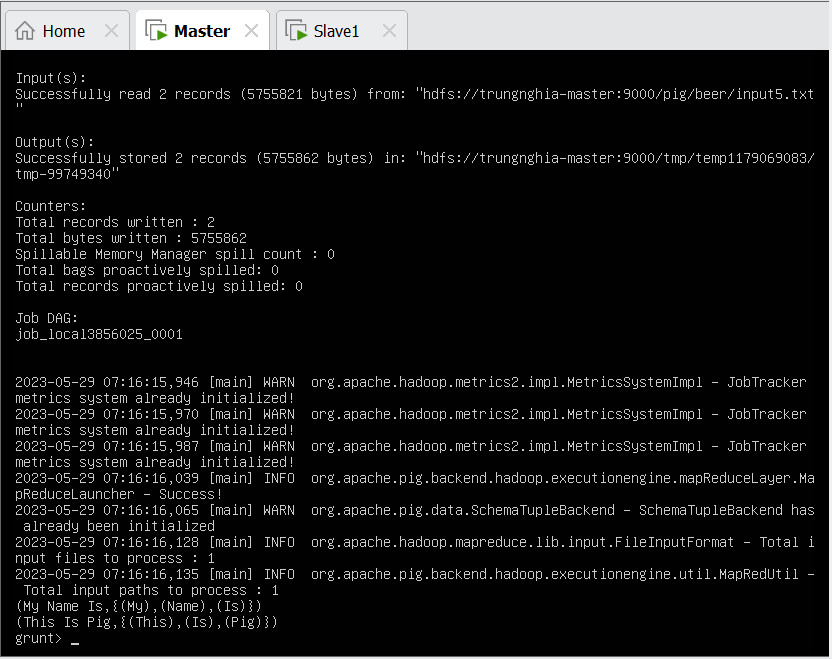
* Câu lệnh TOKENIZE() trong Pig Latin được sử dụng để phân tách một chuỗi (chứa một nhóm từ) trong một tuple duy nhất và trả về một bag chứa kết quả của phép phân tách.
* Cú pháp : [TOKENIZE(expression [, 'field\_delimiter'])]
* Ví dụ:
  + Input
    - Input5.txt



* + Code

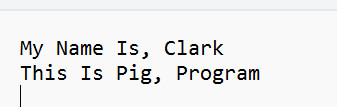


* + Output

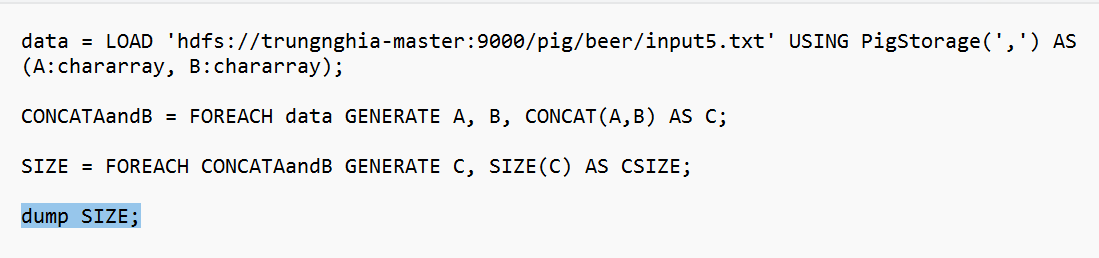


## **Concat and Size**

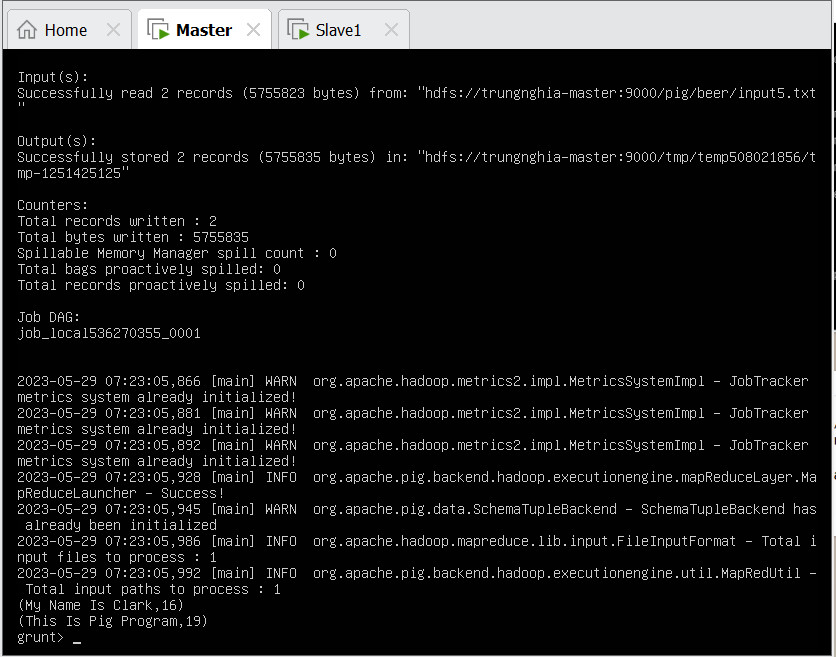
* Trong Apache Pig, câu lệnh CONCAT được sử dụng để nối các giá trị của các cột thành một chuỗi duy nhất. Cú pháp của câu lệnh CONCAT như sau
* Cú pháp: [CONCAT (expression, expression, [...expression])]
* Ví dụ:
  + Input
    - Input5.txt



* + Code



* + Output



# **Chương 3. PHÂN TÍCH VÀ XỬ LÝ DỮ LIỆU VỚI PIG**

## **Giới thiệu dataset**

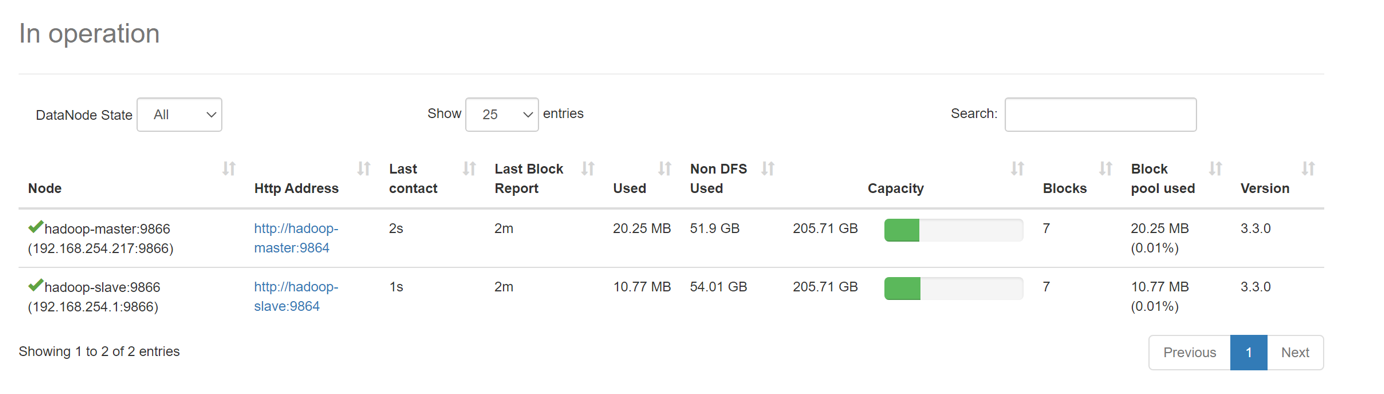
Nhóm lấy nguồn dữ liệu từ trang Kaggle: [Beer Data Analytics](https://www.kaggle.com/datasets/gauravharamkar/beer-data-analytics)

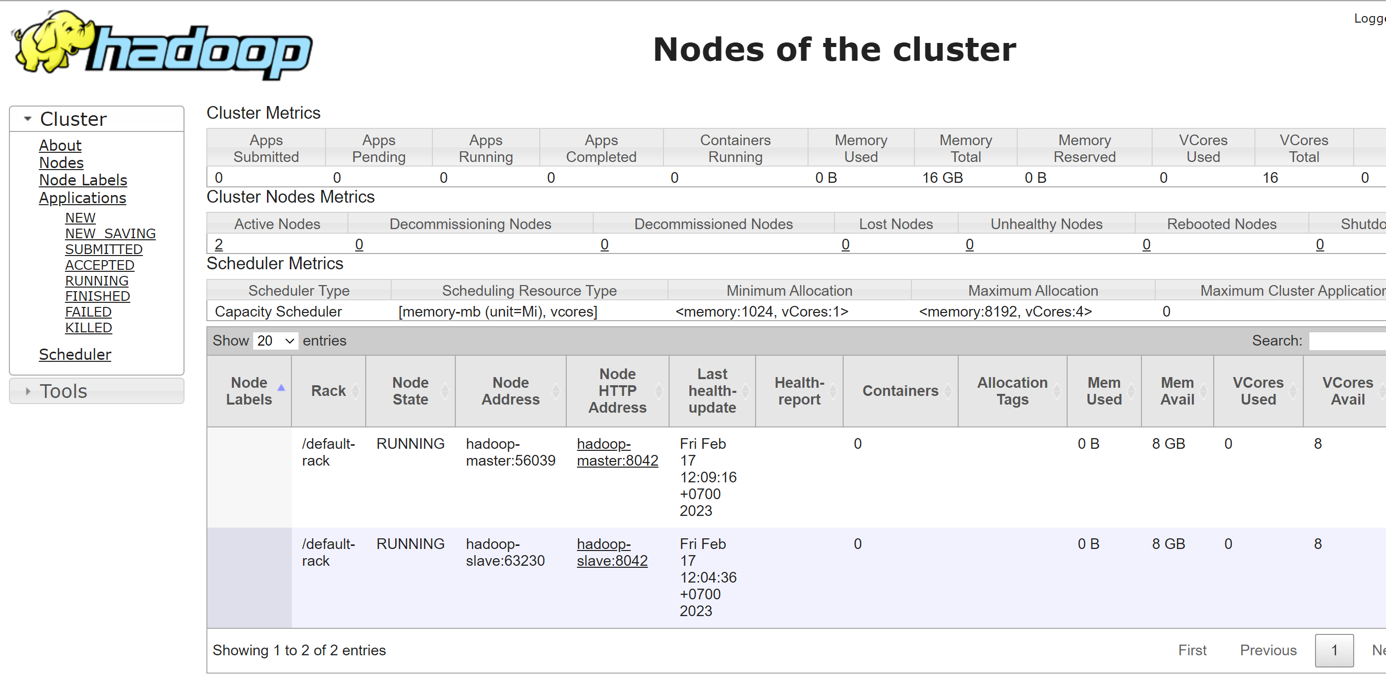
Tập dữ liệu này chứa thông tin về các loại bia khác nhau và các khía cạnh khác nhau của nó như kiểu bia, khối lượng bia tuyệt đối, tên bia, tên nhà sản xuất bia, hình thức bia, hương vị bia, mùi thơm, xếp hạng tổng thể, đánh giá, v.v. Các nhà máy bia đã và đang sử dụng dữ liệu để thúc đẩy các chiến lược tiếp thị của họ. Sử dụng các khía cạnh khác nhau của dữ liệu bia giúp các nhà máy bia hiểu được hành vi của người tiêu dùng và có thể cải thiện việc bán hàng.

Tập dữ liệu gồm: 13 cột, 528870 dòng

|  |  |
| --- | --- |
| beer\_ABV | Chỉ số cho biết nồng độ cồn trong bia (%) |
| beer\_beerId | ID của bia |
| beer\_brewerId | ID người ủ bia |
| beer\_name | Tên bia |
| beer\_style | Loại bia |
| review\_appearance | Điểm đánh giá theo vẻ bề ngoài |
| review\_palette | Điểm đánh giá theo màu sắc |
| review\_overall | Điểm đánh giá tổng thề |
| review\_taste | Điểm đánh giá theo hương vị |
| review\_profileName | tên đồ sơ đánh giá |
| review\_aroma | Điểm đánh giá theo mùi thơm |
| review\_text | Nội dung đánh giá |
| review\_time | Thời gian đánh giá |

## **Hadoop trên Window**

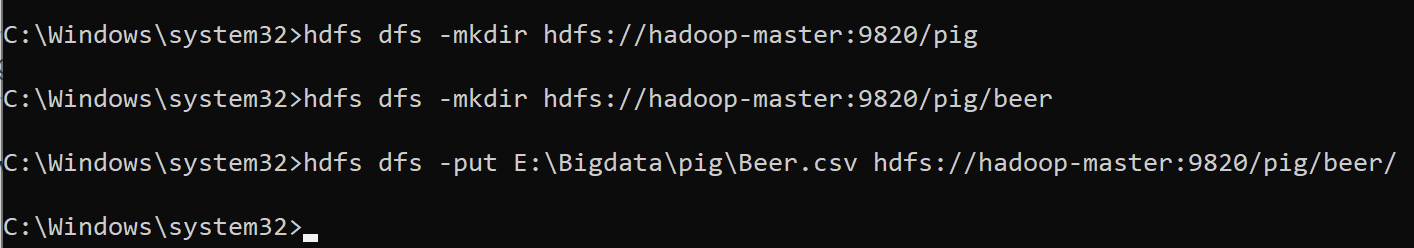


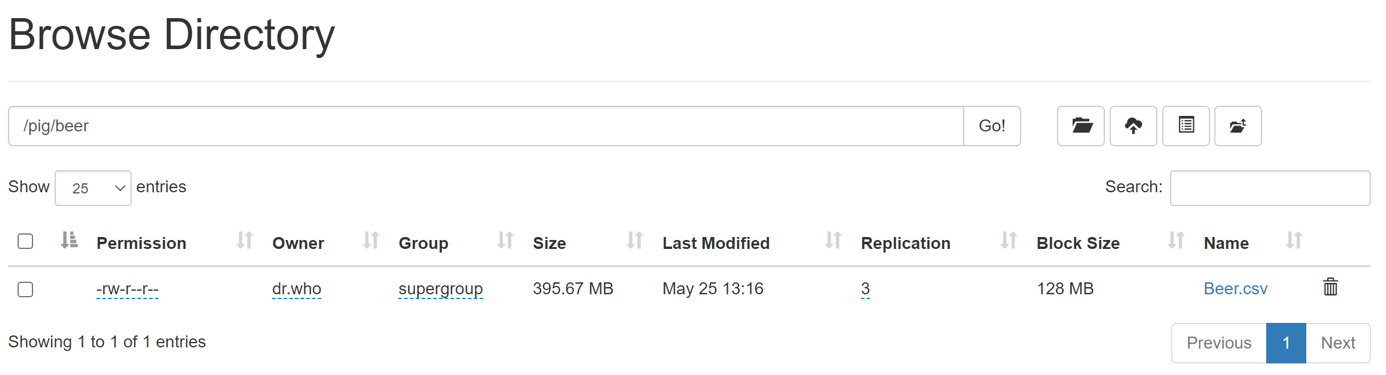


## **Phân tích và xử lý dữ liệu Beer sử dụng Pig**

### **Chuẩn bị dữ liệu**

Tạo hdfs folder pig/beer và đưa dữ liệu vào hdfs folder beer vừa tạo

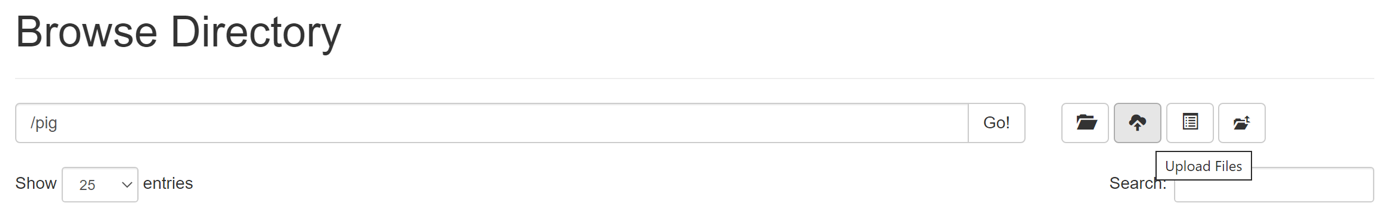


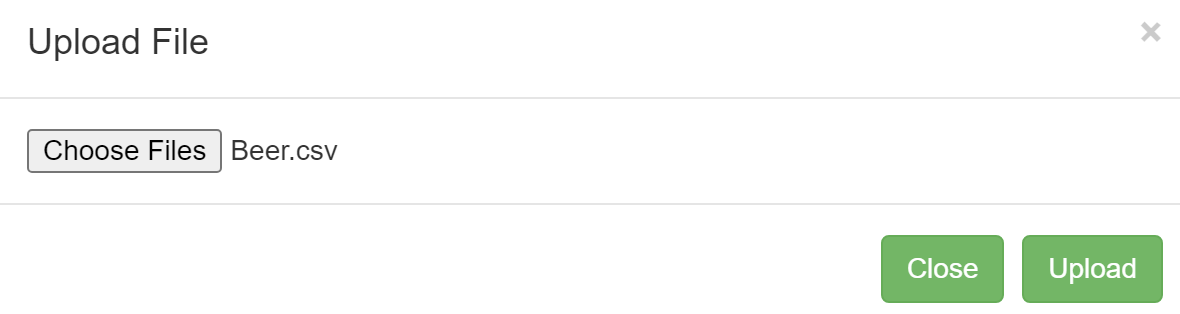


Ngoài cách trên ta còn có thể thao tác trực tiếp trên web:



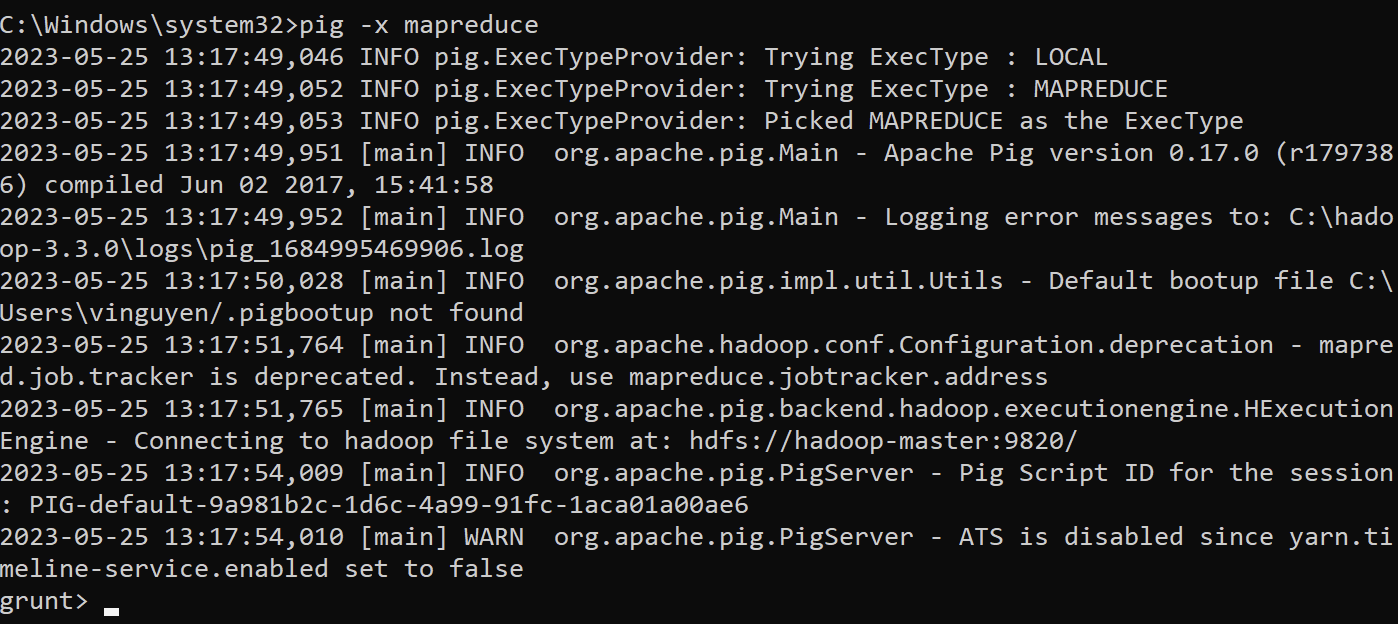






### **Phân tích từng loại beer được người dùng đánh giá tổng quát cao nhất và nồng độ cồn người dùng thích**

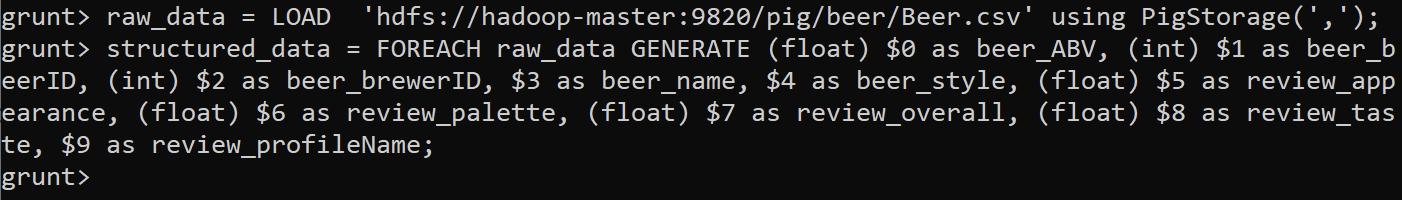
* Khởi động PIG



* Lấy dữ liệu gốc từ file csv và tạo bảng:

raw\_data = LOAD 'hdfs://hadoop-master:9820/pig/beer/Beer.csv' using PigStorage(',');

structured\_data = FOREACH raw\_data  GENERATE  (float) $0 as beer\_ABV, (int) $1 as beer\_beerID, (int) $2 as beer\_brewerID, $3 as beer\_name, $4 as beer\_style, (float) $5 as review\_appearance, (float) $6 as review\_palette, (float) $7 as review\_overall, (float) $8 as review\_taste, $9 as review\_profileName;



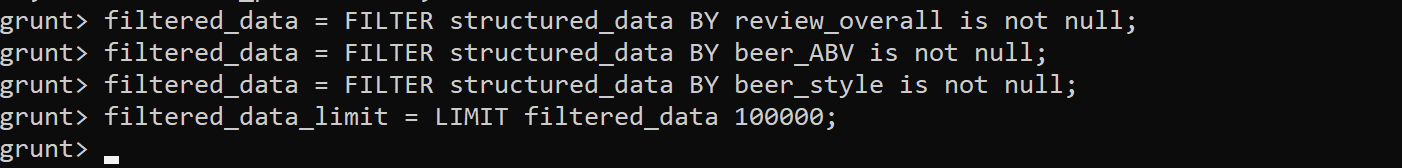
* Lọc dữ liệu chọn và chọn 100000 dòng đầu tiên trong bảng

filtered\_data = FILTER structured\_data BY review\_overall is not null;

filtered\_data = FILTER structured\_data BY beer\_ABV is not null;

filtered\_data = FILTER structured\_data BY beer\_style is not null;

filtered\_data\_limit = LIMIT filtered\_data 100000;



* Duyệt các thành phần trong filtered\_data\_limit sau đó sinh ra một bảng mới chứa các cột beer\_ABV, beer\_style, review\_overall và nhóm lại bảng theo beer\_style. đối với mỗi nhóm (beer\_style), tìm dòng có tổng thể được đánh giá cao nhất, sắp xếp theo thứ tự giảm dần. Sau đó lưu trữ kết quả vào folder.

A1 = FOREACH filtered\_data GENERATE beer\_ABV as beer\_ABV, beer\_style as beer\_style, review\_overall as review\_overall;

B1 = GROUP A1 by beer\_style;

C1 = FOREACH B1 {

I1 = ORDER A1 BY review\_overall DESC;

J1 = LIMIT I1 5;

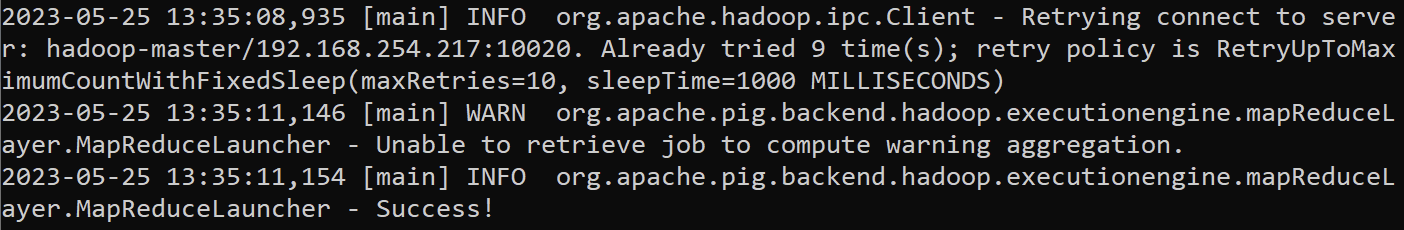
GENERATE group, J1.( beer\_ABV, beer\_style, review\_overall);

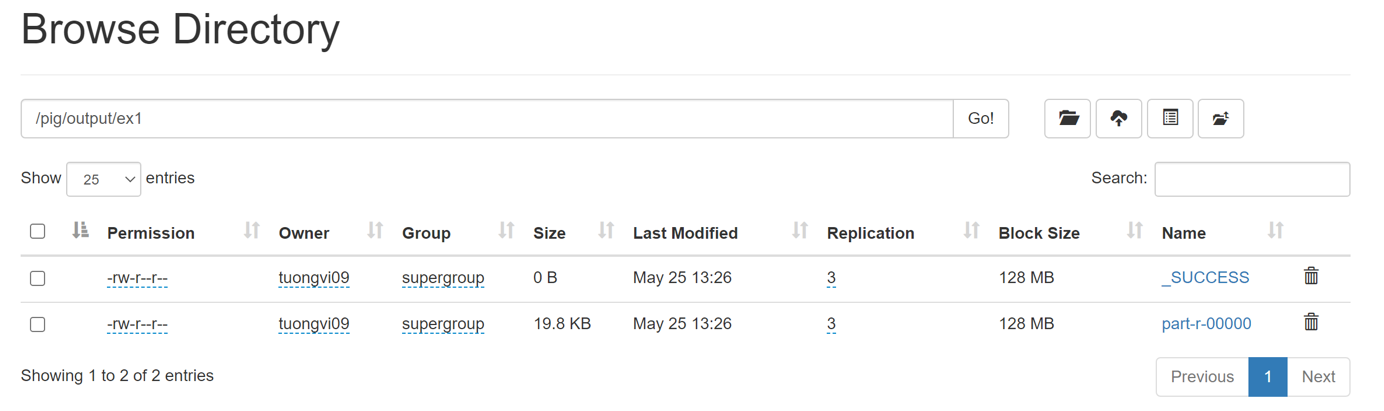
};

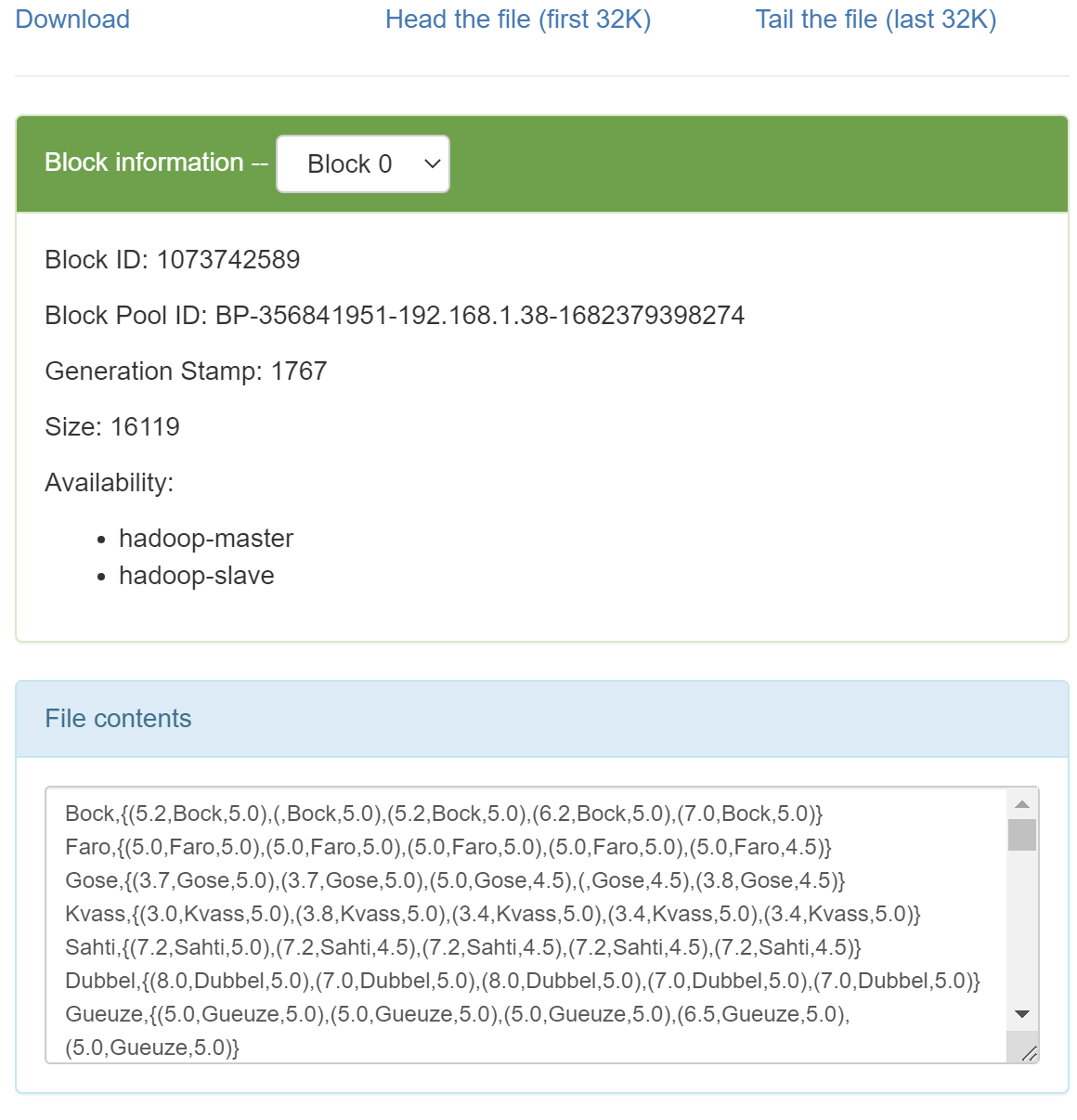
STORE C1 into 'hdfs://hadoop-master:9820/pig/output/ex1/' using PigStorage(',');



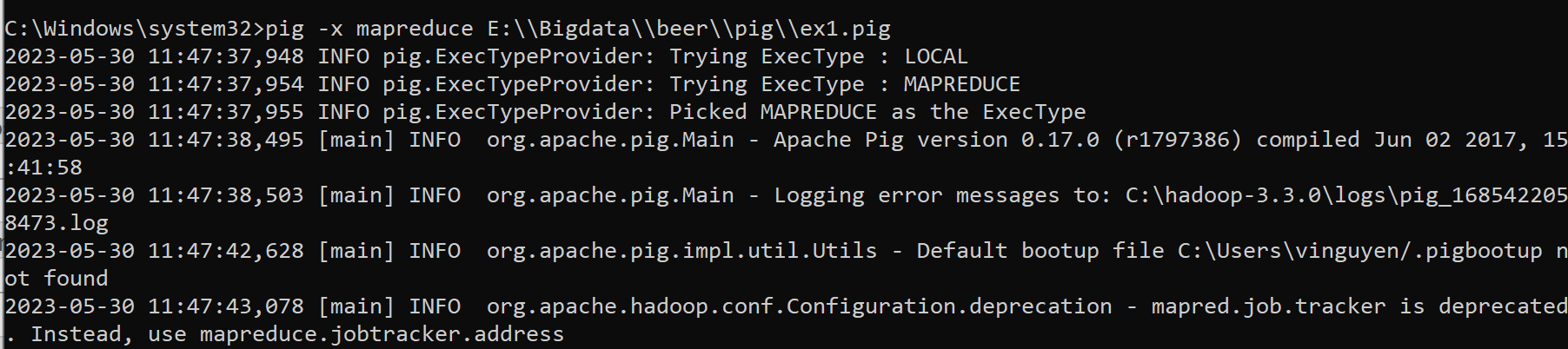
Kết quả:

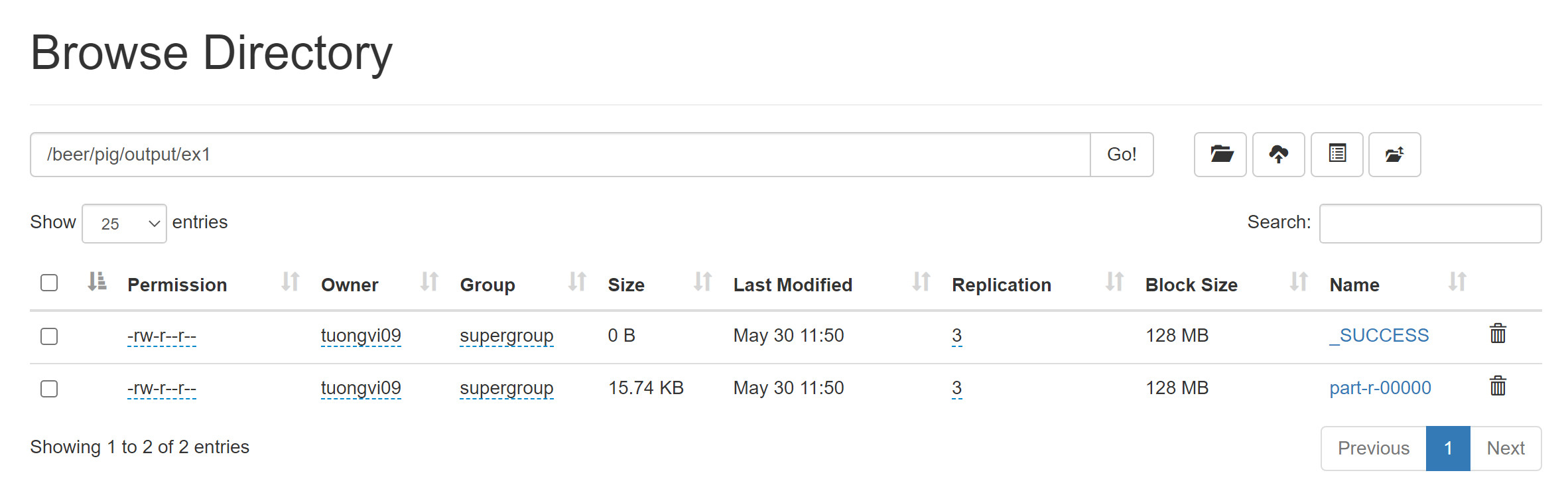


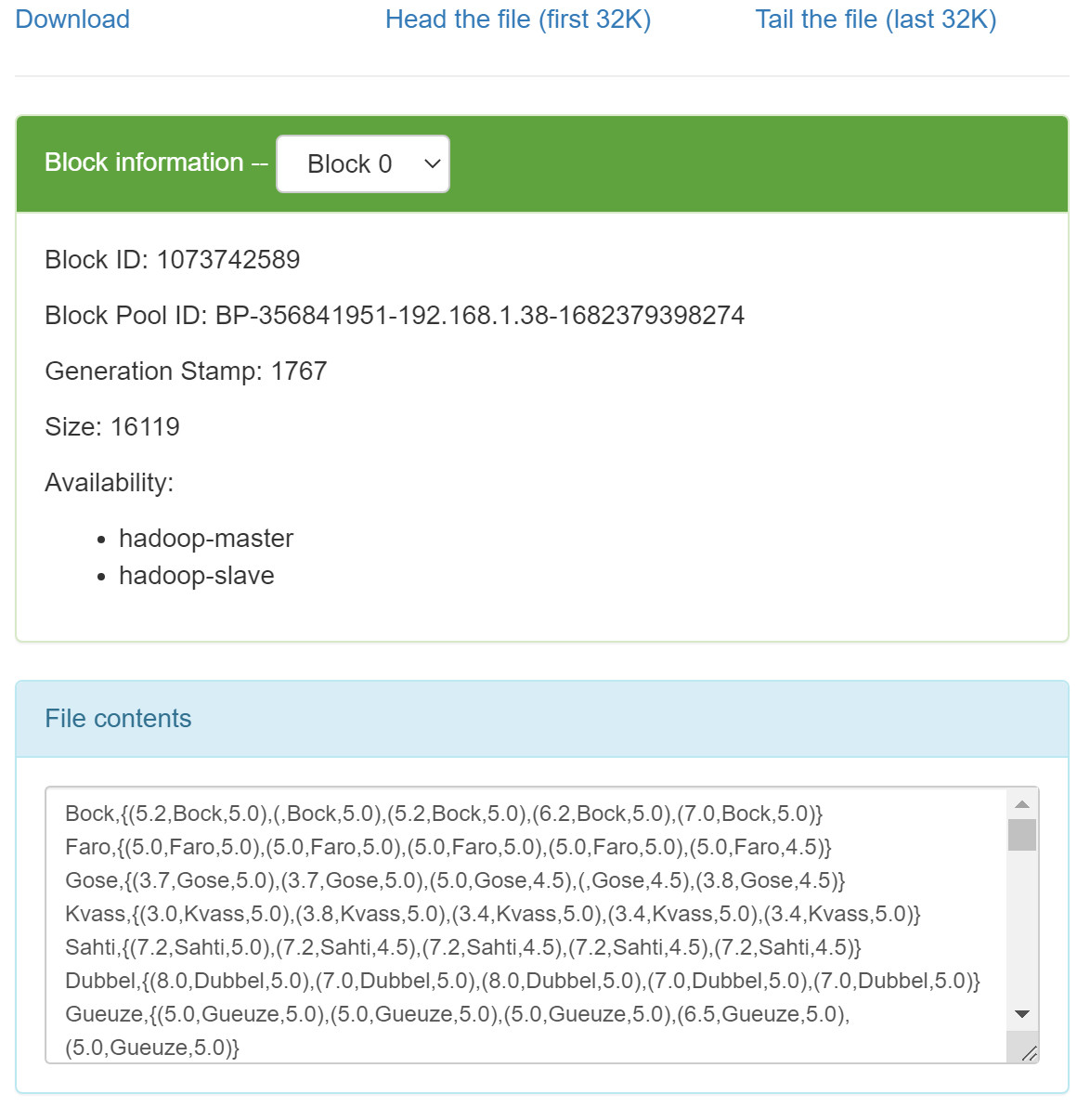




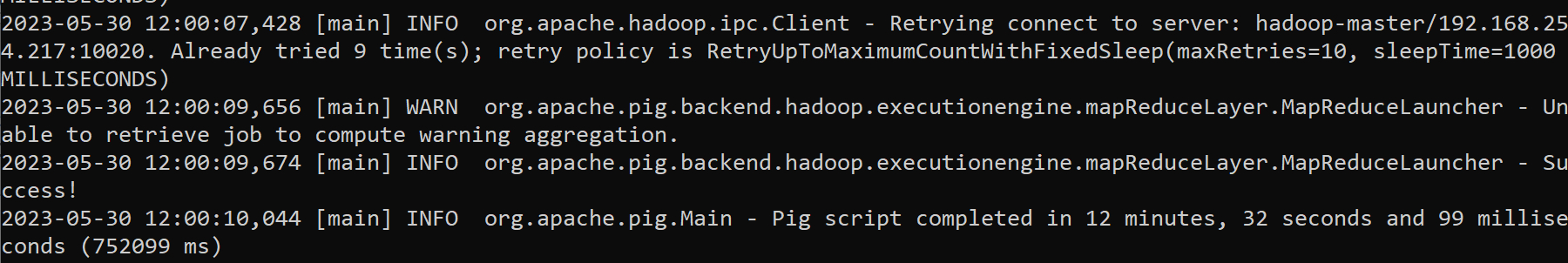
* Ngoài ra có thể chạy bằng file script:







* Thời gian thực hiện: 12 phút 32 giây



### **Phân tích số lượng bia có điểm đánh giá tổng quát từ 0-5**

File script: ex2.pig

--Lấy dữ liệu gốc từ file csv và tạo bảng:

raw\_data = LOAD 'hdfs://hadoop-master:9820/pig/beer/Beer.csv' using PigStorage(',');

structured\_data = FOREACH raw\_data GENERATE (float) $0 as beer\_ABV, (int) $1 as beer\_beerID, (int) $2 as beer\_brewerID, $3 as beer\_name, $4 as beer\_style, (float) $5 as review\_appearance, (float) $6 as review\_palette, (float) $7 as review\_overall, (float) $8 as review\_taste, $9 as review\_profileName;

--Lọc dữ liệu chọn và chọn 100000 dòng đầu tiên trong bảng

filtered\_data = FILTER structured\_data BY review\_overall is not null;

filtered\_data = FILTER structured\_data BY beer\_beerID is not null;

filtered\_data\_limit = LIMIT filtered\_data 100000;

--Duyệt các thành phần trong filtered\_data\_limit

--sau đó sinh ra một bảng mới chứa các cột review\_overall\_bin, beer\_beerID

A2 = FOREACH filtered\_data\_limit GENERATE (float) review\_overall as review\_overall\_bin;

--Tính tổng số lượng bia theo từng khoảng điểm đánh giá tổng thể “review\_overall\_bin”

grouped\_data = FOREACH (

GROUP A2 BY (

CASE

WHEN (review\_overall\_bin >= 0 AND review\_overall\_bin < 1) THEN '[0 - 1]'

WHEN (review\_overall\_bin >= 1 AND review\_overall\_bin < 2) THEN '[1 - 2]'

WHEN (review\_overall\_bin >= 2 AND review\_overall\_bin < 3) THEN '[2 - 3]'

WHEN (review\_overall\_bin >= 3 AND review\_overall\_bin < 4) THEN '[3 - 4]'

WHEN (review\_overall\_bin >= 4 AND review\_overall\_bin <= 5) THEN '[4 - 5]'

ELSE 'Unknown'

END

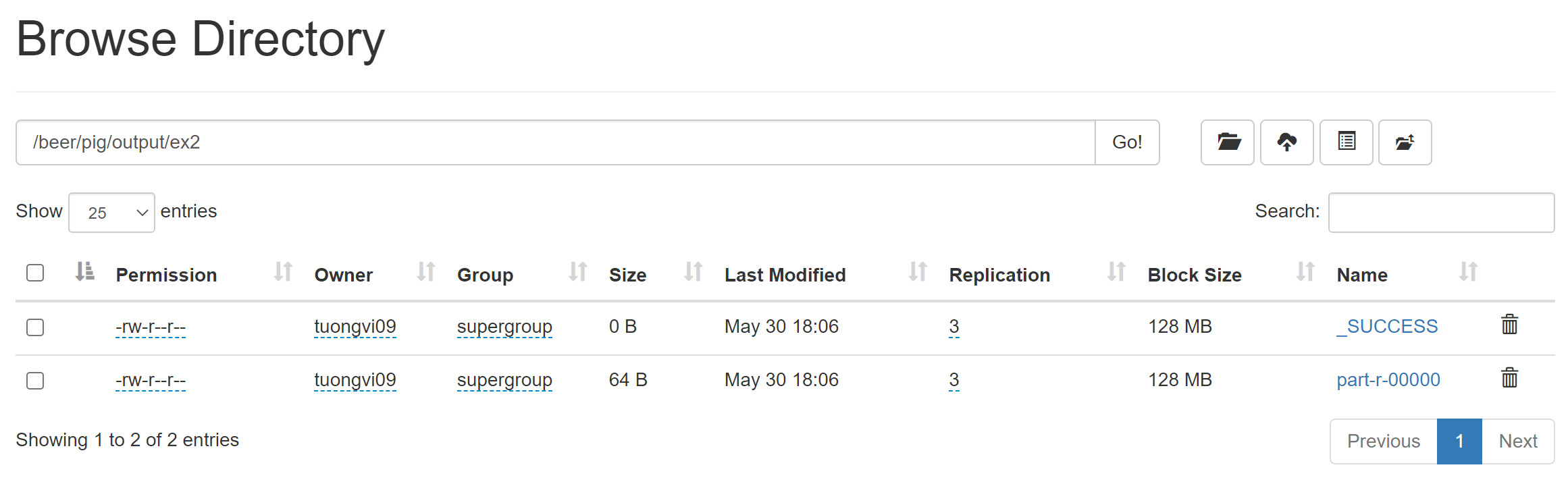
)

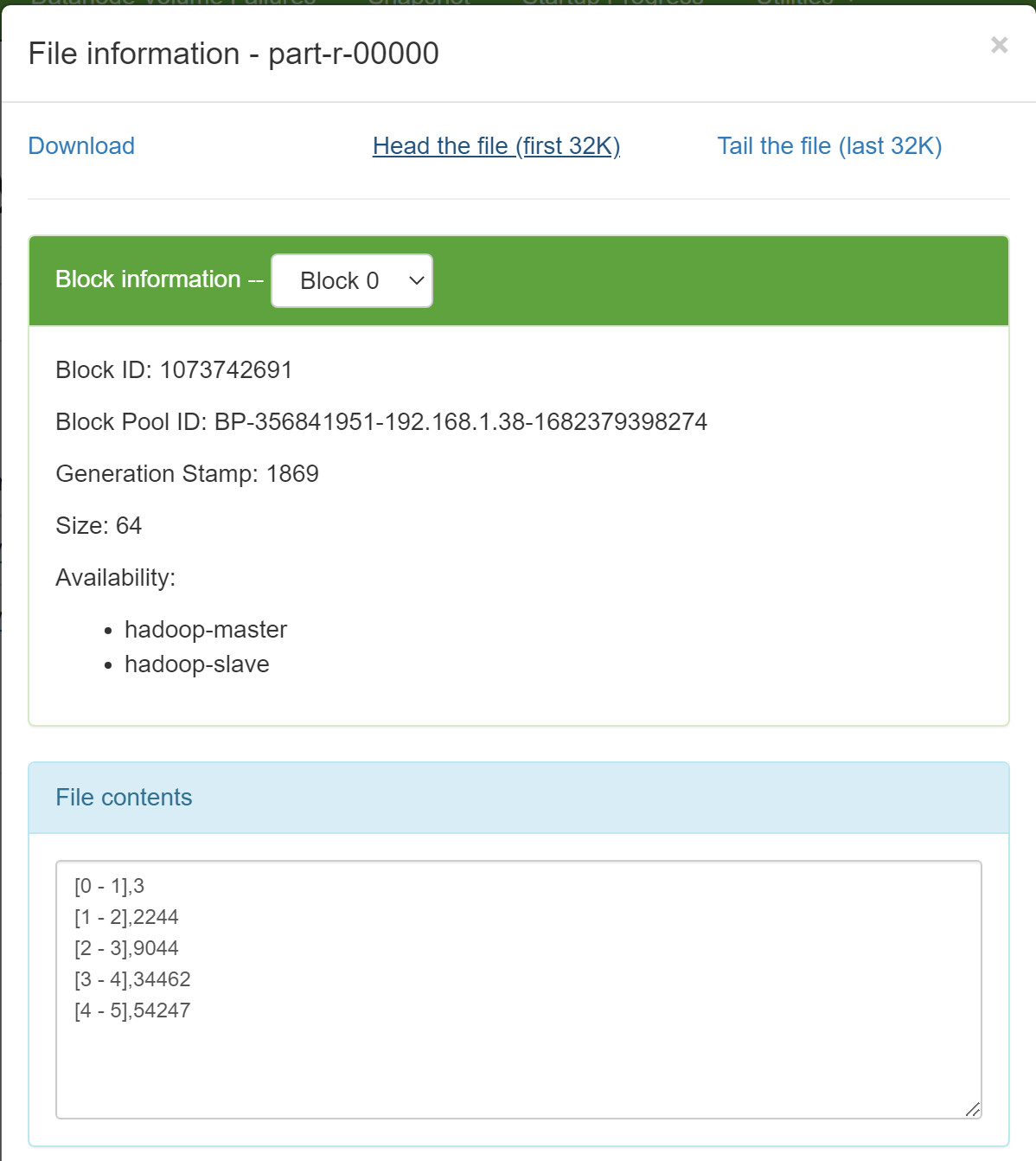
) GENERATE group AS range, COUNT(A2) AS count;

--Lưu kết quả

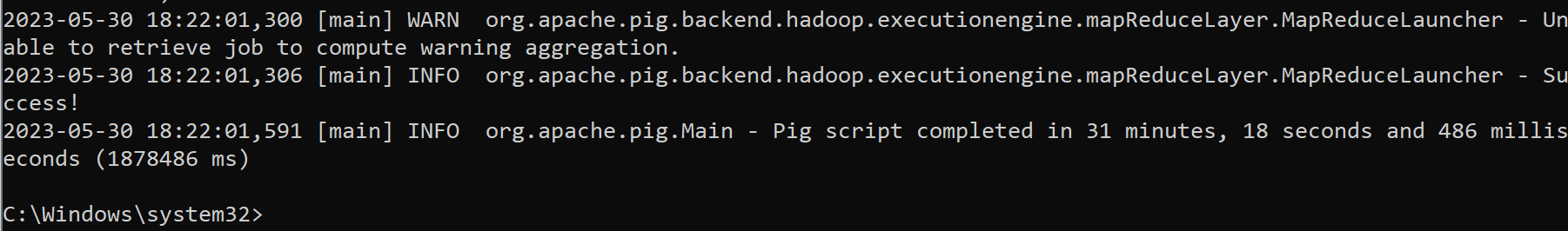
STORE grouped\_data into 'hdfs://hadoop-master:9820/pig/output/ex2/' using PigStorage(',');







* Thời gian thực hiện: 31 phút 18 giây



### **Đếm từng loại beer style có bao nhiêu beer khác nhau**

--Lấy dữ liệu gốc từ file csv và tạo bảng:

raw\_data = LOAD 'hdfs://hadoop-master:9820/pig/beer/Beer.csv' using PigStorage(',');

structured\_data = FOREACH raw\_data GENERATE (float) $0 as beer\_ABV, (int) $1 as beer\_beerID, (int) $2 as beer\_brewerID, $3 as beer\_name, $4 as beer\_style, (float) $5 as review\_appearance, (float) $6 as review\_palette, (float) $7 as review\_overall, (float) $8 as review\_taste, $9 as review\_profileName;

--Lọc dữ liệu chọn và chọn 100000 dòng đầu tiên trong bảng:

filtered\_data = FILTER structured\_data BY review\_overall is not null;

filtered\_data = FILTER structured\_data BY beer\_beerID is not null;

filtered\_data\_limit = LIMIT filtered\_data 100000;

--Duyệt các thành phần trong filtered\_data\_limit sau đó sinh ra một bảng mới chứa các cột beer\_style có kiểu dữ liệu là chararray, beer\_beerID

A3 = FOREACH filtered\_data\_limit GENERATE (chararray) beer\_style as beer\_style, beer\_beerID as beer\_beerID;

--group các dòng theo beer\_style

B3 = GROUP A3 by beer\_style;

--Với mỗi beer\_style, đếm số lượng của beerID và chỉ lấy ra các beer\_beerID có ID khác nhau

C3 = FOREACH B3 {

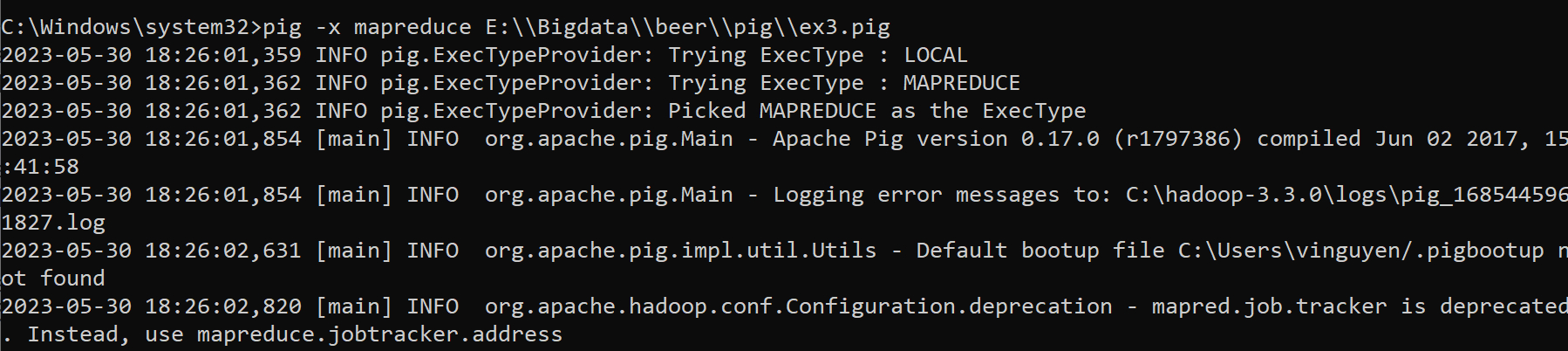
beer\_ids = DISTINCT A3. beer\_beerID;

GENERATE group, COUNT(beer\_ids) as number\_beer;

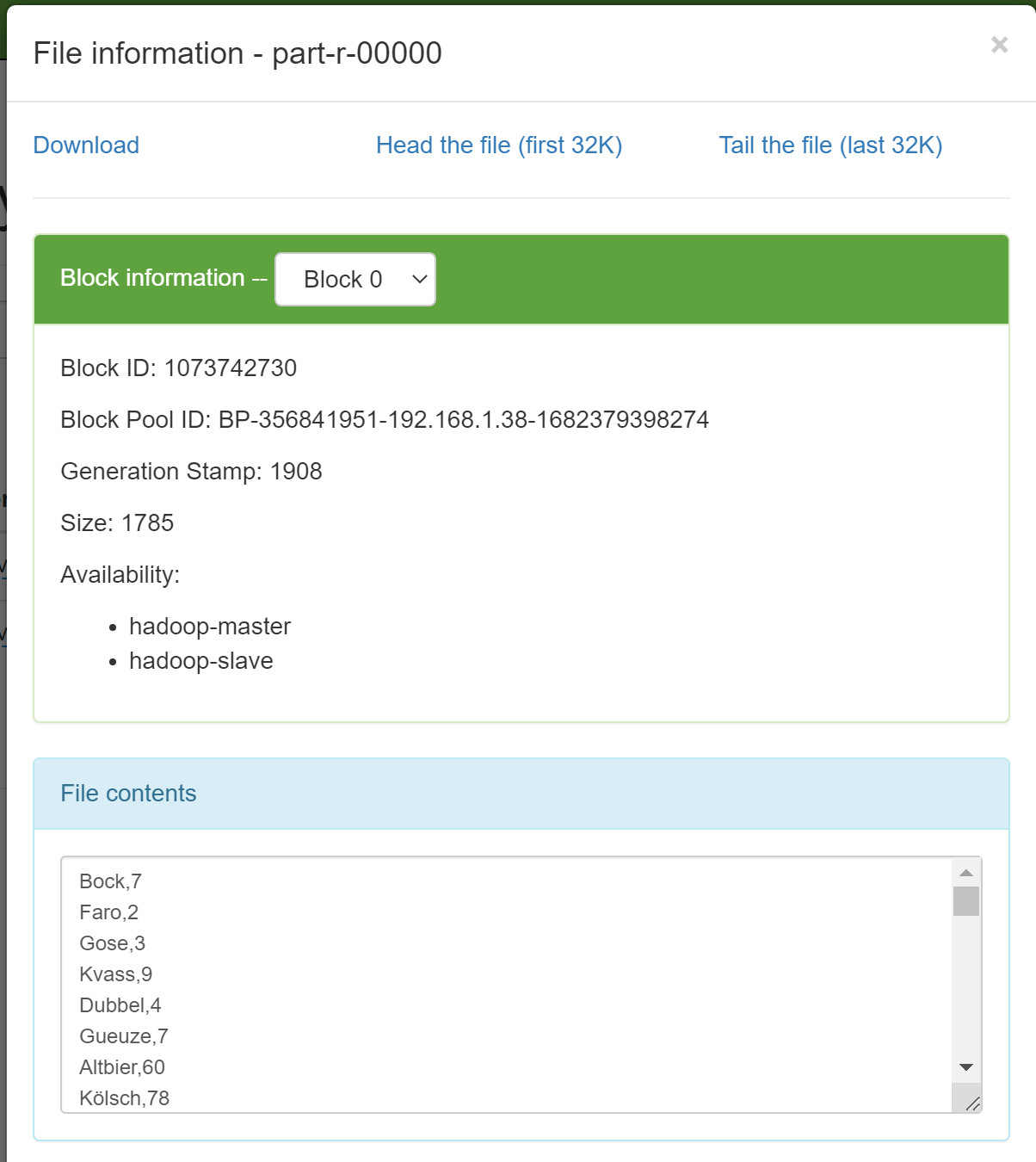
};

--Lưu kết quả:

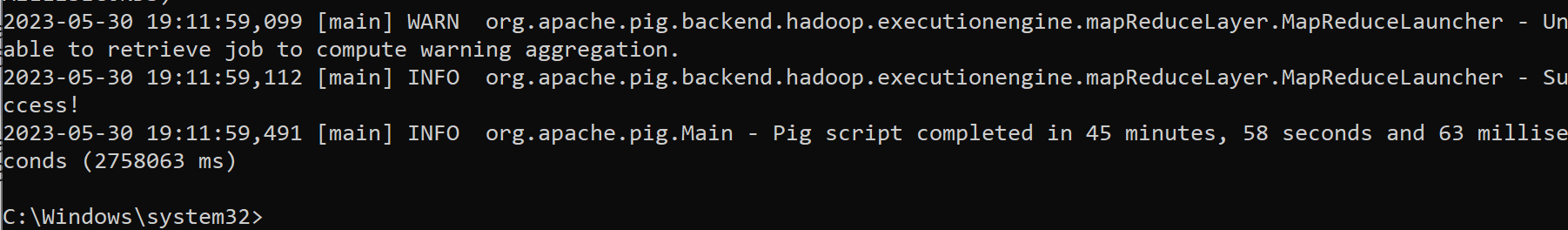
STORE C3 into 'hdfs://hadoop-master:9820/pig/output/ex3/' using PigStorage(',');







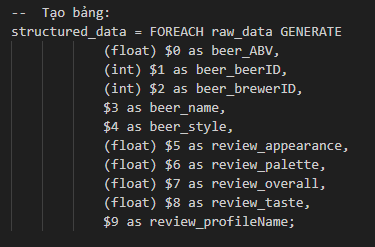
* Thời gian thực hiện: 45 phút 58 giây



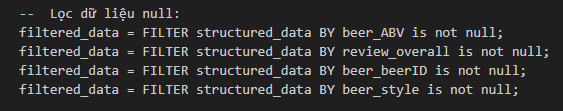
# **Chương 4. PHÂN TÍCH VÀ XỬ LÝ DỮ LIỆU VỚI PIG KẾT HỢP VỚI SPARK**

* 1. **Xử lý dữ liệu trên PIG:**
* Lấy dữ liệu gốc từ file csv và tạo bảng:

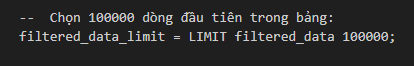




* Lọc dữ liệu null:



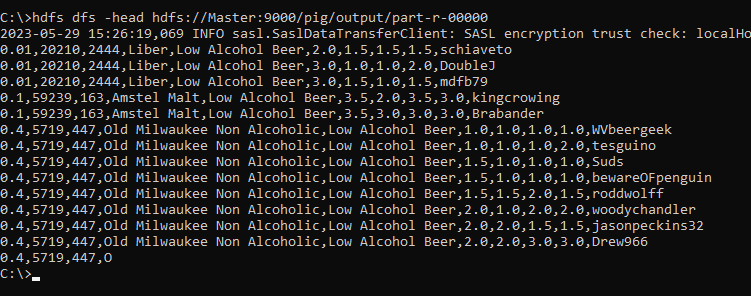
* Chọn 100k dòng đầu tiên trong bảng:



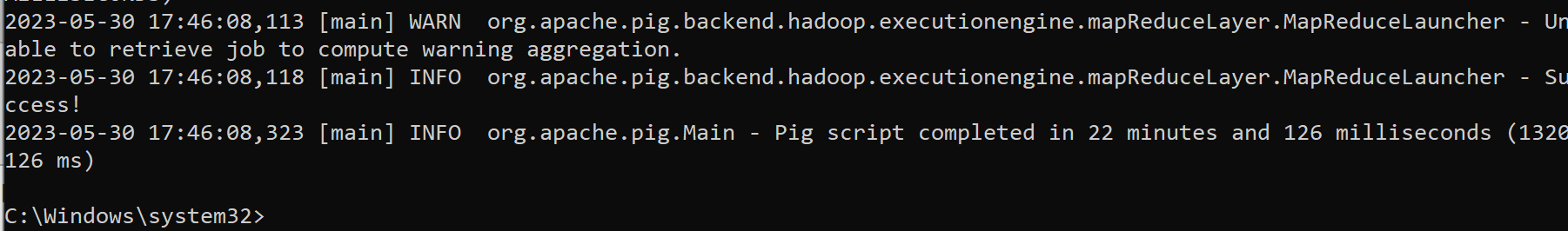
* Xuất kết quả ra file có tên là output và mỗi cột được tách bởi dấu phẩy



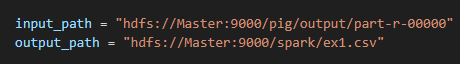
* Kết quả đạt được



* Thời gian hoàn thành: 22 phút

****

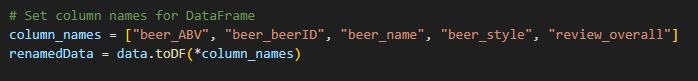
* 1. **Phân tích dữ liệu bằng SPARK:**
     1. **Phân tích từng loại beer được người dùng đánh giá tổng quát cao nhất và nồng độ cồn người dùng thích**
* Tạo đường dẫn input và output



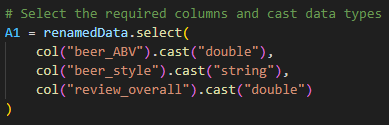
* Đọc dữ liệu vào dataframe



* Đặt tên cột cho dataframe



* Chọn các cột cần phân tích và truyền kiểu dữ liệu



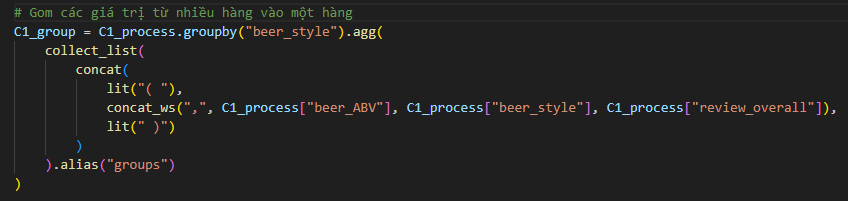
* Nhóm bảng dữ liệu theo beer\_style



* Sắp xếp điểm tổng quát theo chiều giảm dần của từng loại beer và lấy 5 hàng dầu tiên



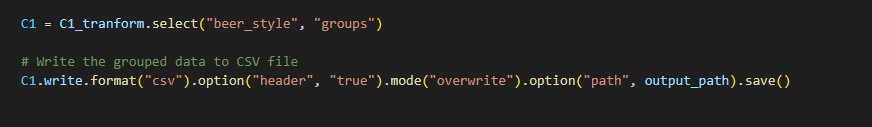
* Gom các giá trị của nhiều hàng thành một hàng nhiều giá trị theo từng loại beer



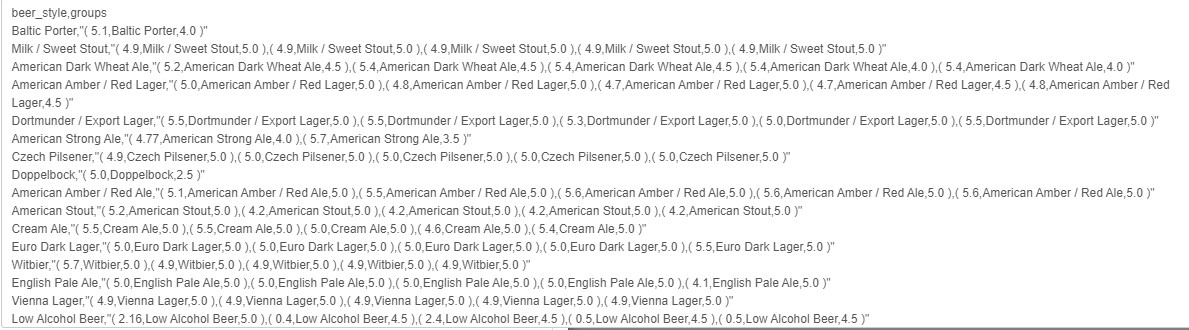
* Từng giá trị trong một hàng đó ngăn cách bằng dấu phẩy



* Chọn các cột cho kết quả và xuất ra file csv



* Kết quả đạt được



* Thời gian:



=> Thời gian thực hiện là 7 giây nhanh hơn so với chạy Pig (12 phút 32 giây)

* + 1. **Phân tích số lượng bia có điểm đánh giá tổng quát từ 0-5**
* Tạo đường dẫn input và output



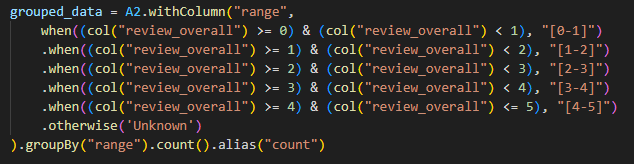
* Đọc dữ liệu vào dataframe



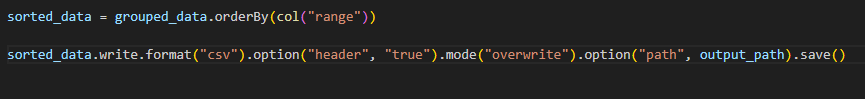
* Tạo bảng mới với cột “review\_overall” để phân tích



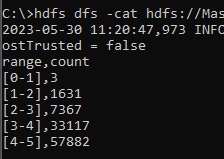
* Nhóm và đếm số lượng bia có điểm đánh giá tổng quát từ 0-5



* Sắp xếp thứ tự của cột “range” và xuất ra file csv



* Kết quả đạt được



* Thời gian



=> Thời gian thực hiện 9s nhanh hơn so với chạy pig (31 phút 18 giây)

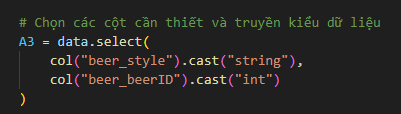
* + 1. **Đếm từng loại beer có bao nhiêu beer khác nhau**
* Tạo đường dẫn input và output



* Đọc dữ liệu vào dataframe



* Chọn các cột cần phần tích và truyền kiểu dữ liệu



* Nhóm dữ liệu theo “beer\_style”



* Đếm từng loại có bao nhiều beerID



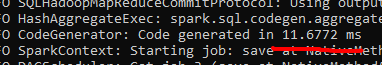
* Xuất ra file csv



* Kết quả đạt được



* Thời gian



=> Thời gian thực hiện là 11 giây nhanh hơn so với chạy pig (45 phút 58 giây)

# **Chương 5. KẾT LUẬN**

Trong môi trường Hadoop, Apache Pig cung cấp nhiều chức năng nâng cao để phân tích và xử lý dữ liệu, đồng thời kết hợp với Spark để tận dụng hiệu suất và tính linh hoạt của hai công cụ này.

Các chức năng nâng cao của Apache Pig có thể cung cấp cú pháp đơn giản và dễ hiểu, giúp người dùng tạo và thao tác dữ liệu một cách dễ dàng hơn. Pig cũng hỗ trợ xử lý dữ liệu có cấu trúc linh hoạt, cho phép làm việc với nhiều định dạng khác nhau như CSV, JSON, Avro, Parquet và nhiều loại dữ liệu khác. Tuy nhiên về thời gian xử lý và thực thi script mất nhiều thời gian trong khoảng 20 đến 30 phút, đây là một hạn chế đối với Apache Pig trong việc xử lý dữ liệu lớn

Việc kết hợp Pig với Spark trong môi trường Hadoop lại là một giải pháp cho hạn chế trên, khi thời gian xử lý chỉ trong khoảng vài giây. Pig có thể chạy trên cụm Hadoop và sử dụng các tính năng của Spark để tăng tốc độ xử lý dữ liệu lớn và phân tán. Spark cung cấp hiệu suất xử lý cao và khả năng tích hợp với hệ sinh thái dữ liệu lớn, giúp tối ưu hóa quá trình xử lý và tăng cường hiệu suất.

Khi kết hợp Pig và Spark, người dùng có thể thực hiện phân tích và xử lý dữ liệu một cách hiệu quả và linh hoạt. Pig cung cấp các toán tử và hàm tích hợp để phân tích dữ liệu, trong khi Spark đảm bảo hiệu suất xử lý cao. Điều này giúp người dùng tạo ra các luồng xử lý phức tạp và xử lý dữ liệu lớn một cách hiệu quả trong môi trường Hadoop.

Tổng kết lại, Apache Pig cung cấp nhiều chức năng nâng cao để phân tích và xử lý dữ liệu trong môi trường Hadoop. Kết hợp với Spark, Pig tận dụng sự mạnh mẽ của hai công cụ này để mang lại hiệu suất và tính linh hoạt trong việc phân tích và xử lý dữ liệu lớn trong môi trường Hadoop.

# **TÀI LIỆU THAM KHẢO**

|  |  |
| --- | --- |
| [1] | A. Spark, "Spark SQL Guide," https://spark.apache.org/, [Online]. Available: https://spark.apache.org/docs/latest/sql-getting-started.html. [Accessed 05 2023]. |
| [2] | H. A. DŨNG, "LV\_Hoang Anh Dung.pdf," https://uet.vnu.edu.vn/, 2020. [Online]. Available: http://lib.uet.vnu.edu.vn/bitstream/123456789/1139/1/LV\_Hoang%20Anh%20Dung.pdf. [Accessed 05 2023]. |
| [3] | S. B. Examples, "Spark By Examples | Learn Spark Tutorial with Examples," https://sparkbyexamples.com/, [Online]. Available: https://sparkbyexamples.com/. [Accessed 05 2023]. |
| [4] | X. Gravity, "Setting up Hadoop 3.1.3 multi node cluster with problems and solution in Windows 10," https://www.youtube.com/, 2020. [Online]. Available: https://www.youtube.com/watch?v=FAly8HaYkbQ. [Accessed 05 2023]. |
| [5] | X. Gravity, "Setting up Hadoop Datanode on Windows 10," https://www.youtube.com/, 2020. [Online]. Available: https://www.youtube.com/watch?v=HFa5wFGuS08. [Accessed 05 2023]. |
| [6] | Tutorialspoint, "Apache Spark Tutorial," https://www.tutorialspoint.com/, [Online]. Available: https://www.tutorialspoint.com/apache\_spark/index.htm. [Accessed 05 2023]. |
| [7] | Tutorialspoint, "Apache Pig Tutorial," https://www.tutorialspoint.com/, [Online]. Available: https://www.tutorialspoint.com/apache\_pig/index.htm. [Accessed 05 2023]. |
| [8] | H. Lê, “PHÂN TÍCH DỮ LIỆU LỚN VỚI APACHE PIG TRÊN AWS ELASTIC MAPREDUCE,” https://renovacloud.com/, [Trực tuyến]. Available: https://renovacloud.com/phan-tich-du-lieu-lon-voi-apache-pig-tren-aws-elastic-mapreduce/. [Đã truy cập 05 2023]. |