TRƯỜNG ĐẠI HỌC HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN MÔN HỌC:
NHẬP MÔN PHÂN TÍCH DỮ LIỆU LỚN

Chủ đề:

ÁP DỤNG HADOOP CHO VIỆC XỬ LÝ DỮ LIỆU LỚN CỦA ỨNG DỤNG PHÂN TÍCH/ DỰ BÁO ĐIỂM HỌC SINH

Nhóm sinh viên thực hiện (Họ tên - Mã SV):

1. Lê Minh Tâm - 207CT28471 (Trưởng nhóm)

2. Nguyễn Quốc An – 207CT27552

3. Trần Trung Kiên – 207CT10144

4. Võ Văn Thanh Nhân – 207CT10223

TP. Hồ Chí Minh – năm 2023

NHẬN XÉT CỦA GIẢNG VIÊN

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

MỤC LỤC

PHỤ LỤC HÌNH ẢNH

LỜI NÓI ĐẦU

Viết một báo cáo đồ án môn học là một trong những việc được coi là khó mà chúng em phải hoàn thành trong quá trình học một môn học. Trong quá trình thực hiện đề tài chúng em đã gặp rất nhiều khó khăn và bỡ ngỡ. Nếu không có những sự giúp đỡ và lời động viên chân thành của nhiều người có lẽ chúng em khó có thể hoàn thành tốt tiểu luận này. Đầu tiên chúng em xin gửi lời biết ơn chân thành đến thầy Hoàng Lê Minh, người trực tiếp hướng dẫn chúng em hoàn thành tiểu luận này.

Những ý kiến đóng góp của thầy là vô cùng hữu ích, nó giúp chúng em nhận ra các khuyết điểm của đồ án và rút kinh nghiệm cho những bài học lần sau. Cảm ơn thầy và các bạn trường Đại học Văn Lang là những người đã cùng nhóm em sát cánh và trải nghiệm để hoàn thành đồ án môn học.

Nhóm xin chân thành cảm ơn.

MỞ ĐẦU

1. LÝ DO CHỌN CHỦ ĐỀ.

Xử lý lượng dữ liệu khổng lồ và Dữ liệu lớn ngày nay đã trở thành một trong những vấn đề quan trọng nhất trong khoa học máy tính. Những khó khăn với giáo dục trong lĩnh vực này phát sinh, các phương pháp và công cụ giảng dạy phù hợp là cần thiết. Việc xử lý một lượng lớn dữ liệu đến một cách nhanh chóng đòi hỏi phải lựa chọn và sắp xếp các nền tảng phần cứng mở rộng. Trong bài báo cáo, nhóm sẽ giả lập áp dụng Hadoop cho việc xử lý dữ liệu lớn của ứng dụng quản lý sinh viên. Việc sử dụng nền tảng học tập điện tử Moodle, một nền tảng dành riêng cho giảng dạy, có thể cho phép đội ngũ giảng viên và sinh viên tiếp xúc tốt hơn bằng cách tăng cường khả năng giao tiếp lẫn nhau. Giải pháp có thể cho phép đối phó với các vấn đề trong giáo dục sinh viên trong lĩnh vực Dữ liệu lớn.

2. ĐỐI TƯỢNG, PHẠM VI TÌM HIỂU CỦA CHỦ ĐỀ.

Đối tượng nghiên cứu của bài báo cáo là giáo viên, giảng viên, sinh viên, học sinh của các trường trung học và đại học của các tỉnh trên toàn nước. Chủ yếu lấy dữ liệu bao gồm các thông tin cơ bản của các học sinh và sinh viên từ hệ thống của các trường, từ đó nhóm thống kê thông tin và tạo dữ liệu chuẩn nhất.

Tạo ra một ứng dụng quản lý sinh viên bằng cách áp dụng Hadoop vào môi trường giáo dục, một nền tảng dành riêng cho các giáo viên, giảng viên từ đó dễ dàng lưu trữ và quản lý sinh viên, học sinh giúp đưa ra các quyết định và đề xuất cho tương lai.

CHƯƠNG 1. GIỚI THIỆU CHỦ ĐỀ ĐỒ ÁN

1.1 KHÁI QUÁT MỘT SỐ NỘI DUNG CƠ BẢN VỀ HADOOP VÀ NGUỒN GỐC CỦA HADOOP: [1] [2]

1.1.1 Hadoop là gì?

Một câu hỏi quan trọng là loại cơ sở hạ tầng phần cứng và phần mềm nào nên được sử dụng để xử lý các tập dữ liệu lớn? Rõ ràng là nó sẽ là quá nhiều đối với một máy tính, chúng tôi cũng biết rằng ví dụ: Các hệ thống RDBM chỉ có thể được mở rộng đến một giới hạn cố định. Vì vậy, chúng tôi cần một hệ thống có thể được thu nhỏ tuyến tính và với chi phí hợp lý. Chúng ta cần tăng sức mạnh tính toán bằng cách thêm máy tính thay vì thay thế chúng.

Có một giải pháp như vậy - Hadoop, một nền tảng phần mềm có thể mở rộng cho điện toán phân tán. Hadoop có thể lưu trữ kích thước dữ liệu thực tế không giới hạn và có thể xử lý dữ liệu này trong môi trường phân tán. Đây là một giải pháp mã nguồn mở, miễn phí và tương đối đơn giản để mở rộng quy mô. Tất nhiên chúng ta phải ghi nhớ rằng một phần cứng mở rộng sẽ tạo ra chi phí.

1.1.2 Nguồn gốc sự phát triển của Hadoop?

Hadoop là một Apache framework có mã nguồn mở được viết bằng Java. Hadoop cho phép người dùng phát triển các ứng dụng phân tán để lưu trữ, quản lý các tập dữ liệu. Một số ứng dụng có khả năng làm việc với hàng trăm node và hàng nghìn petabyte dữ liệu. Hadoop được phát triển dựa trên mô hình MapReduce, theo đó ứng dụng được chia thành nhiều phân đoạn khác nhau và chạy song song trên nhiều node khác nhau. Với cơ chế streaming, Hadoop còn cho phép phát triển các ứng dụng ở dạng phân tán dựa trên các ngôn ngữ lập trình C++, Python, Pearl, ...

Sự phát triển của Hadoop bắt đầu khi một số các kỹ sư phần mềm nhận ra rằng sẽ vô cùng hữu ích để có thể lưu trữ và phân tích các tập dữ liệu lớn hơn nhiều khả năng lưu trữ và truy cập thực tế trên một thiết bị lưu trữ vật lý (như đĩa cứng chẳng hạn). Khởi đầu của ý tưởng này có thể là do các thiết bị lưu trữ vật lý dần dần sẽ phải lớn hơn, cần nhiều thời gian hơn để thành phần đọc dữ liệu từ đĩa (nằm trong đĩa cứng, có thể là phần "head") di chuyển đến một phân đoạn cụ thể nào đó. Thay vào đó, nhiều thiết bị nhỏ hơn làm việc song song sẽ cho hiệu quả tốt hơn một thiết bị lớn.

1.2 CÁC MODULES QUAN TRỌNG CỦA NỀN TẢNG HADOOP

Nền tảng Hadoop chứa ba mô-đun quan trọng:

- HDFS – Hadoop Distributed File System;
- YARN – một khuôn khổ để lập lịch công việc và quản lý tài nguyên cụm;

- MapReduce – một hệ thống dựa trên YARN để xử lý song song các tập dữ liệu lớn.

Trong môi trường Hadoop phân tán, mỗi phần dữ liệu được lưu trữ trong một số bản sao (thường ít nhất là 3) trên các máy tính khác nhau (tức là các nút cụm). Một trong những giả định quan trọng nhất là dữ liệu được xử lý cục bộ. Điều đó có nghĩa là dữ liệu được xử lý ở nơi nó được lưu trữ (trên cùng một máy tính/nút) giúp giảm thiểu việc truyền mạng. Hơn nữa, hệ thống có khả năng chịu lỗi, tức là khi một trong các nút bị lỗi, kết quả từ nút này sẽ bị mất. Vì vậy, chỉ cần lặp lại các tính toán từ một nút bị hỏng. Hơn nữa, những tình huống như vậy được quản lý bởi chính hệ thống, vì vậy người dùng không cần thực hiện thêm hành động nào.

Bây giờ chúng tôi sẽ đưa ra mô tả ngắn gọn về các hạng mục chính của nền tảng Hadoop: HDFS và MapReduce.

1.2.1 Hadoop Distributed File System

HDFS (Hadoop Distributed File System) là một hệ thống tệp phân tán, là một phần của hệ sinh thái Apache Hadoop. Nó được thiết kế để lưu trữ và quản lý khối lượng lớn dữ liệu trên nhiều nút trong cụm Hadoop.

Trong Hadoop, mỗi tệp được lưu trữ trong một hệ thống tệp được chia thành nhiều phần (block) và mỗi phần được lưu trữ trong một số bản sao trên các vị trí khác nhau. Kích thước 1 block thông thường là 128MB, kích thước này có thể thay đổi được bằng việc cấu hình.

1.2.2 MapReduce

MapReduce là một framework (với các tool và method) để xử lý dữ liệu song song trong môi trường Hadoop.

- Một thao tác bản đồ trong đó đối với mỗi bản ghi, chúng tôi tính toán các cặp khóa-giá trị; tất cả các cặp có cùng khóa sẽ nằm trong cùng một nhóm, ví dụ: nếu chúng tôi xử lý dữ liệu thời tiết và đối với mỗi quan sát, chúng tôi tính toán một cặp nhiệt độ năm, thì tất cả các quan sát có cùng năm sẽ thuộc cùng một nhóm.
- Một hoạt động giảm trong đó đối với mỗi nhóm, chúng tôi tính toán một số tính năng từ giá trị (aggregate), ví dụ: đối với mọi nhóm quan sát được tính toán ở điểm trước, chúng tôi có thể tính toán nhiệt độ tối đa chẳng hạn. Sơ đồ này rất giống với việc nhóm và tổng hợp thông tin trong SQL.

1.3 SO SÁNH SỰ KHÁC BIỆT GIỮA HADOOP VÀ MONGODB [3] [4]

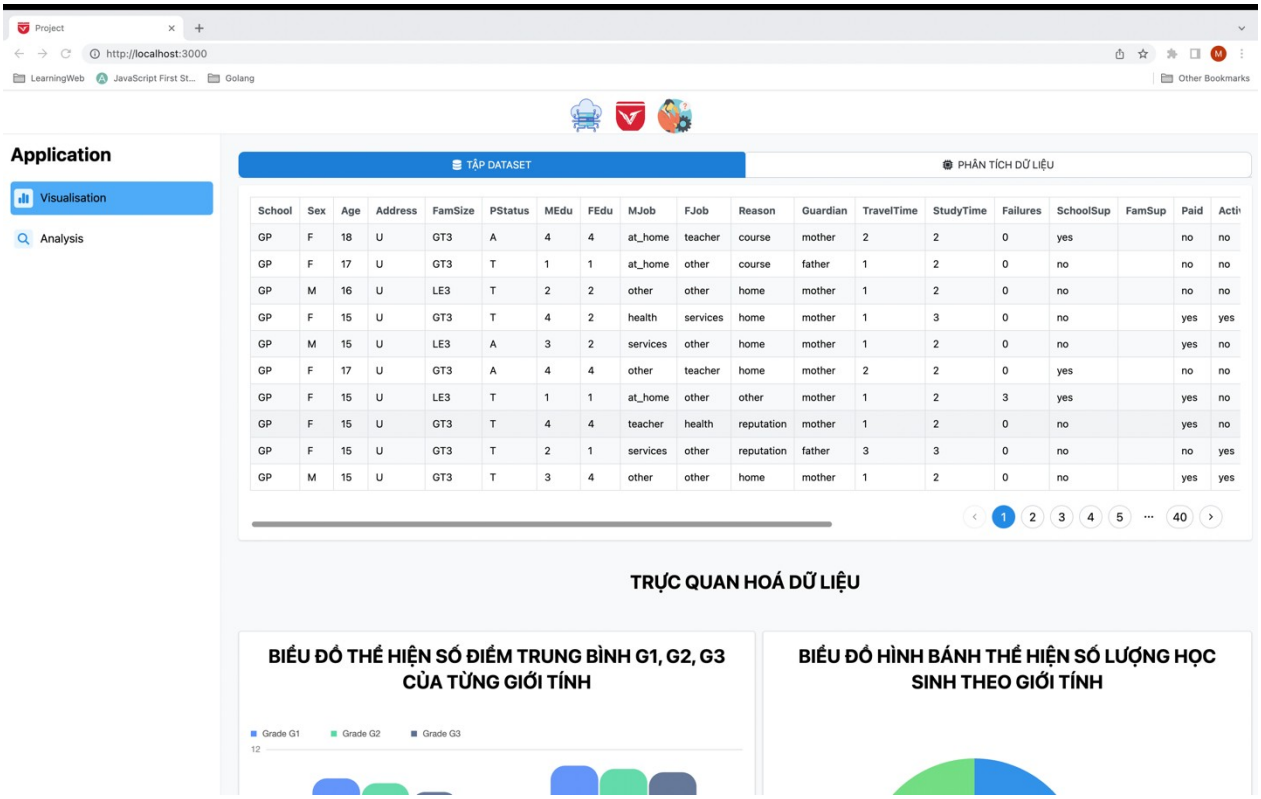
	Hadoop	MongoDB
Data Model	Hadoop là một khung xử lý dữ liệu phân tán chủ yếu tập trung vào xử lý hàng loạt các tập dữ liệu lớn. Nó sử dụng Hệ thống tệp phân tán Hadoop (HDFS) để lưu trữ và quản lý dữ liệu theo cách phân tán. Hadoop không có mô hình dữ liệu được xác định trước và có thể hoạt động với dữ liệu có cấu trúc, bán cấu trúc và phi cấu trúc.	MongoDB là một cơ sở dữ liệu định hướng tài liệu NoSQL cung cấp một lược đồ linh hoạt. Nó lưu trữ dữ liệu ở định dạng giống như JSON được gọi là BSON (JSON nhị phân). MongoDB được thiết kế để xử lý dữ liệu có cấu trúc và bán cấu trúc, đồng thời hỗ trợ các khả năng lập chỉ mục và truy vấn phong phú.
Scalability	Hadoop được thiết kế để mở rộng quy mô theo chiều ngang bằng cách thêm nhiều phần cứng hàng hóa hơn vào cụm. Nó có thể xử lý khối lượng dữ liệu lớn và phân phối khối lượng công việc xử lý trên nhiều nút, giúp nó có khả năng mở rộng cao.	MongoDB cũng có khả năng mở rộng và có thể xử lý các tập dữ liệu lớn. Nó hỗ trợ phân đoạn tự động, cho phép dữ liệu được phân phối trên nhiều nút để chia tỷ lệ theo chiều ngang.
Data Processing Paradigm	Hadoop tuân theo mô hình xử lý hàng loạt và được tối ưu hóa để xử lý khối lượng dữ liệu lớn bằng MapReduce hoặc các khung xử lý phân tán khác như Apache Spark. Nó vượt trội trong việc xử lý khối lượng công việc xử lý dữ liệu ngoại tuyến hoặc hàng loạt.	MongoDB hỗ trợ truy vấn thời gian thực và rất phù hợp với khối lượng công việc xử lý giao dịch trực tuyến (OLTP). Nó cung cấp khả năng truy vấn phong phú, lập chỉ mục và khung tổng hợp, làm cho nó phù hợp để truy cập và phân tích dữ liệu thời gian thực.
Data Model Flexibility	Hadoop có thể xử lý nhiều loại dữ liệu khác nhau, bao gồm dữ liệu có	Mô hình dữ liệu hướng tài liệu của MongoDB cung

	cấu trúc, bán cấu trúc và phi cấu trúc. Nó cung cấp sự linh hoạt trong việc xử lý và phân tích các định dạng dữ liệu khác nhau.	cấp tính linh hoạt trong việc xử lý các cấu trúc dữ liệu khác nhau. Nó cho phép lưu trữ dữ liệu không có lược đồ, làm cho nó có thể thích ứng với các yêu cầu dữ liệu thay đổi.
Data Storage	HDFS của Hadoop được tối ưu hóa để lưu trữ khối lượng lớn dữ liệu trên nhiều nút theo cách phân tán. Nó cung cấp khả năng chịu lỗi và tính sẵn sàng dữ liệu cao.	MongoDB lưu trữ dữ liệu theo cách phân tán trên nhiều nút. Nó sử dụng một định dạng lưu trữ linh hoạt có thể xử lý một loạt các kích thước và cấu trúc dữ liệu.

Tóm lại, Hadoop là khung xử lý dữ liệu phân tán được tối ưu hóa để xử lý hàng loạt và phân tích dữ liệu quy mô lớn, trong khi MongoDB là cơ sở dữ liệu định hướng tài liệu NoSQL được thiết kế để truy vấn thời gian thực và lưu trữ dữ liệu linh hoạt. Lựa chọn giữa Hadoop và MongoDB phụ thuộc vào các trường hợp sử dụng cụ thể, yêu cầu xử lý dữ liệu, nhu cầu về khả năng mở rộng và tính linh hoạt của mô hình dữ liệu mong muốn. Chúng cũng có thể được sử dụng cùng nhau trong một số tình huống nhất định, trong đó Hadoop được sử dụng để xử lý hàng loạt và MongoDB đóng vai trò là kho lưu trữ dữ liệu thời gian thực.

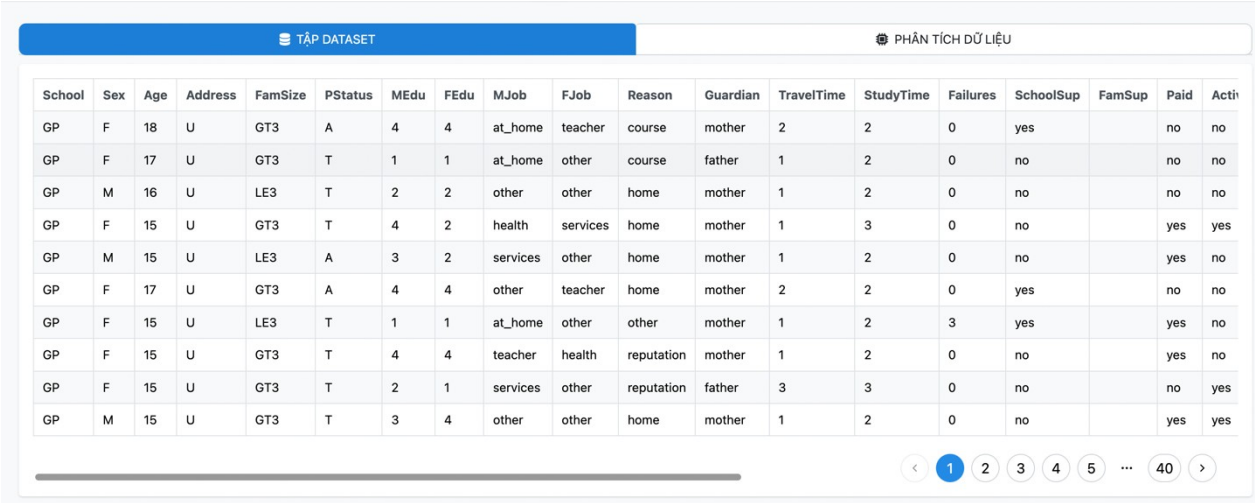
CHƯƠNG 2. KẾT QUẢ THỰC HIỆN ĐỒ ÁN

2.1 THIẾT KẾ ĐƯỢC GIAO DIỆN CƠ BẢN CỦA ỨNG DỤNG QUẢN LÝ SINH VIÊN



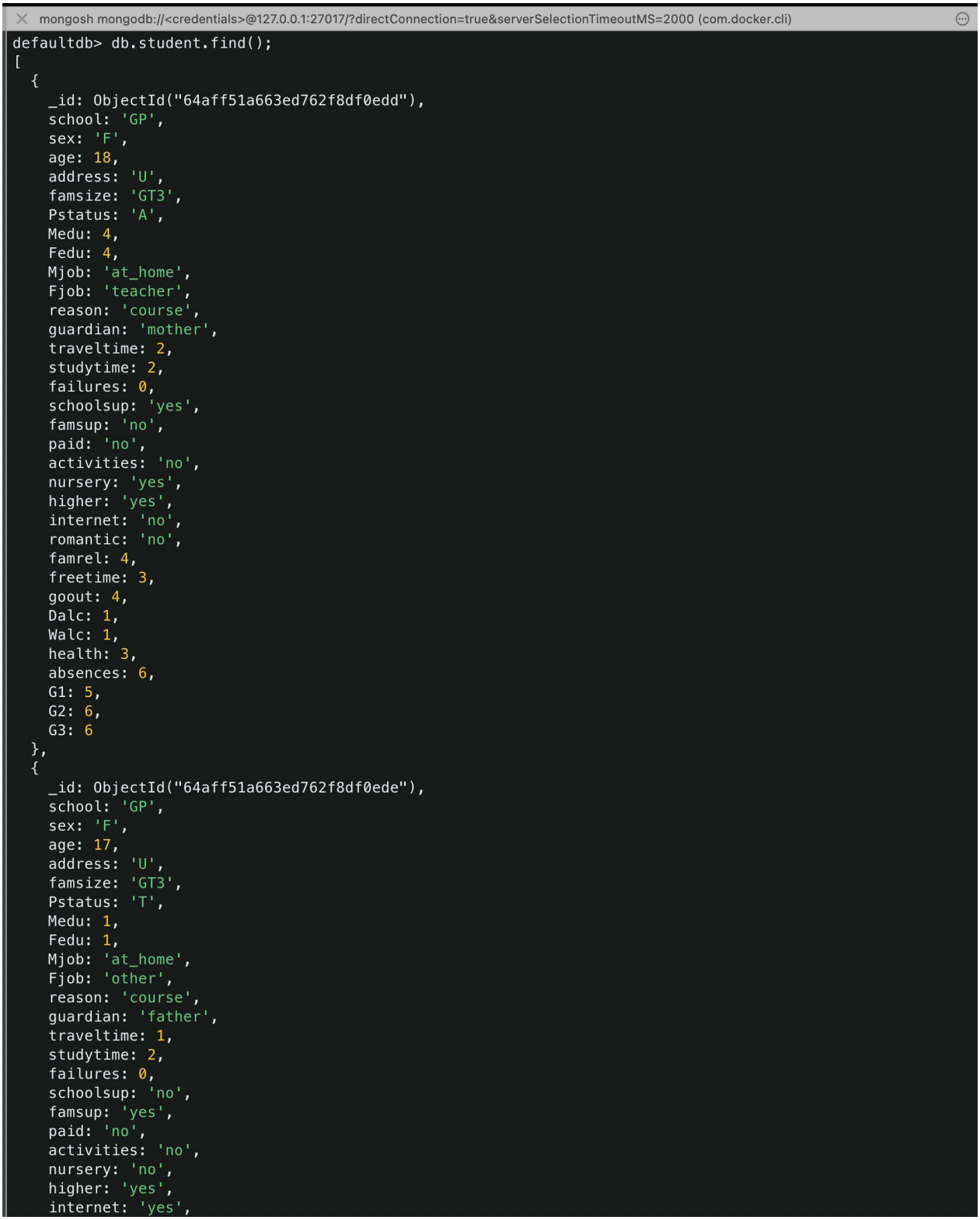
Hình 1: Giao diện hiển thị các chức năng hiển thị, hadoop ứng dụng, MapReduce

2.2 TIẾN HÀNH TRIỂN KHAI CÁC CHỨC NĂNG CHO ỨNG DỤNG



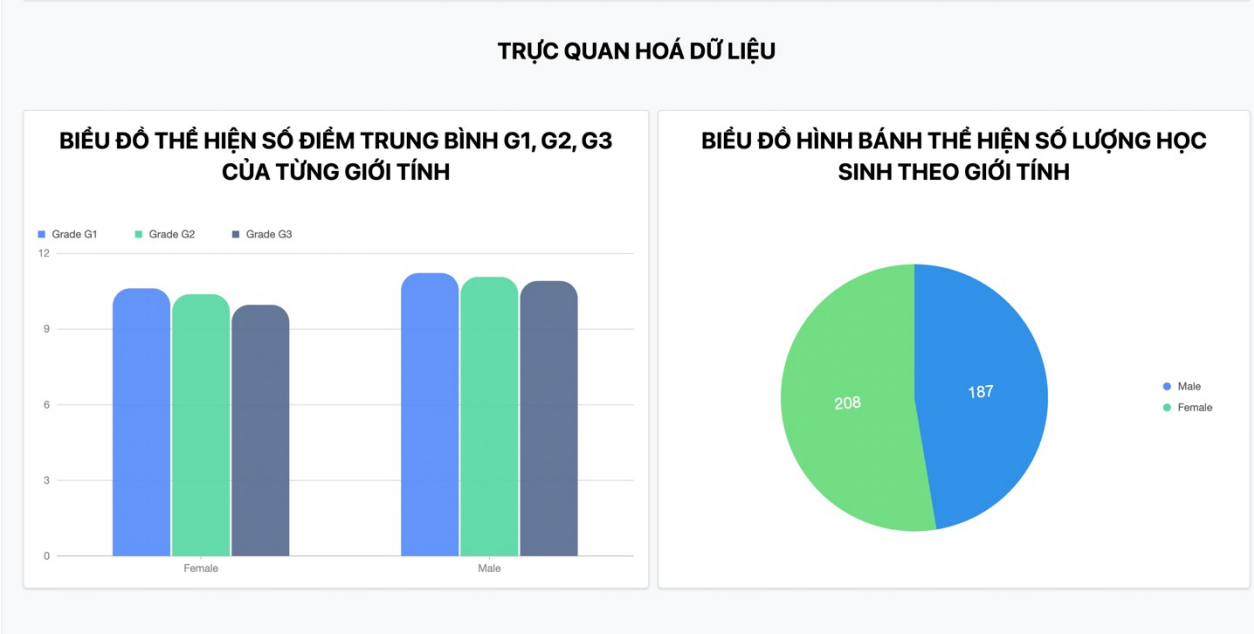
School	Sex	Age	Address	FamSize	PStatus	MEdu	FEdu	MJob	FJob	Reason	Guardian	TravelTime	StudyTime	Failures	SchoolSup	FamSup	Paid	Acti
GP	F	18	U	GT3	A	4	4	at_home	teacher	course	mother	2	2	0	yes		no	no
GP	F	17	U	GT3	T	1	1	at_home	other	course	father	1	2	0	no		no	no
GP	M	16	U	LE3	T	2	2	other	other	home	mother	1	2	0	no		no	no
GP	F	15	U	GT3	T	4	2	health	services	home	mother	1	3	0	no		yes	yes
GP	M	15	U	LE3	A	3	2	services	other	home	mother	1	2	0	no		yes	no
GP	F	17	U	GT3	A	4	4	other	teacher	home	mother	2	2	0	yes		no	no
GP	F	15	U	LE3	T	1	1	at_home	other	other	mother	1	2	3	yes		yes	no
GP	F	15	U	GT3	T	4	4	teacher	health	reputation	mother	1	2	0	no		yes	no
GP	F	15	U	GT3	T	2	1	services	other	reputation	father	3	3	0	no		no	yes
GP	M	15	U	GT3	T	3	4	other	other	home	mother	1	2	0	no		yes	yes

Hình 2: Tập dataset được hiện thị sử dụng MongoDB để lấy dữ liệu

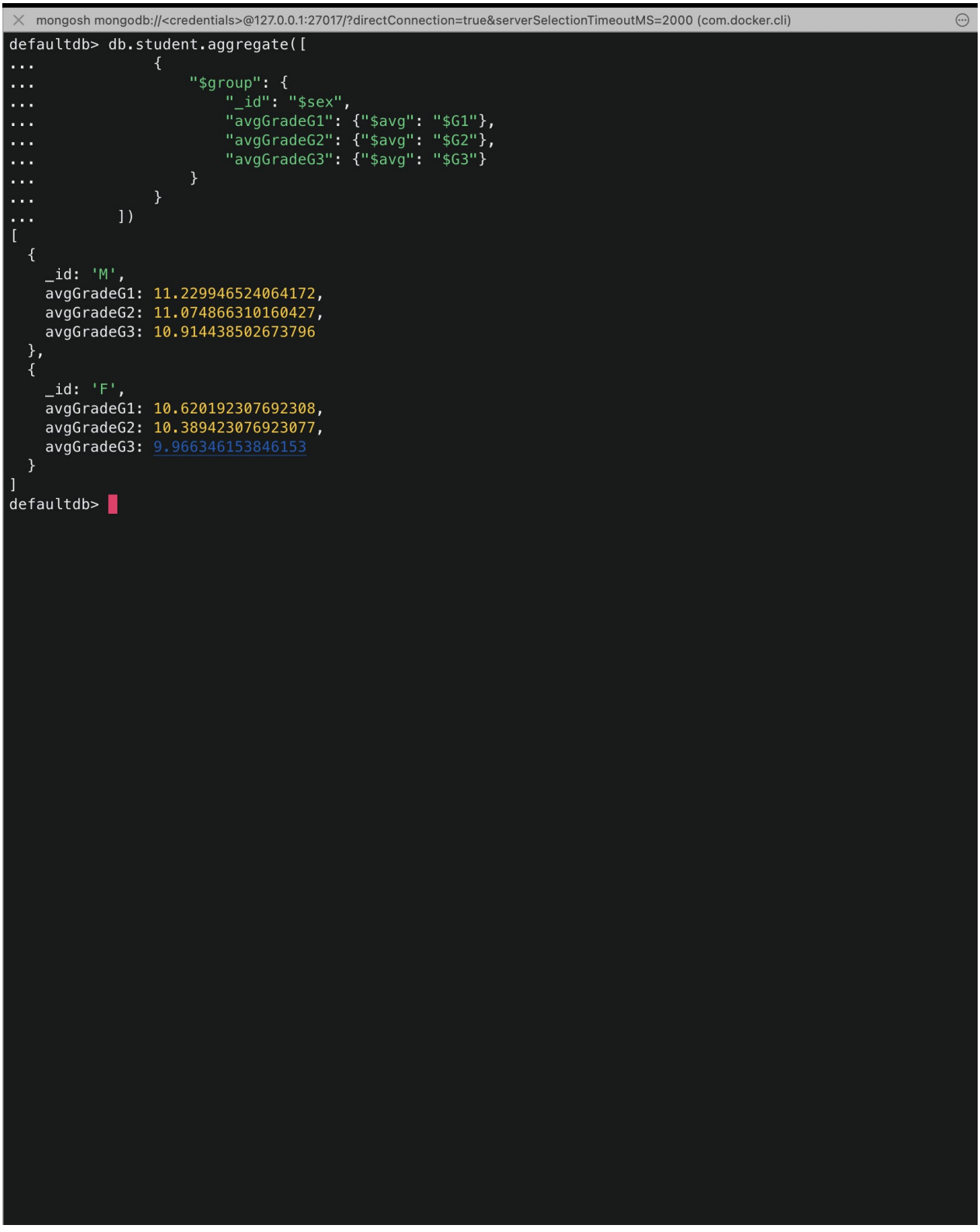


Hình 3: Dữ liệu lấy từ MongoDB

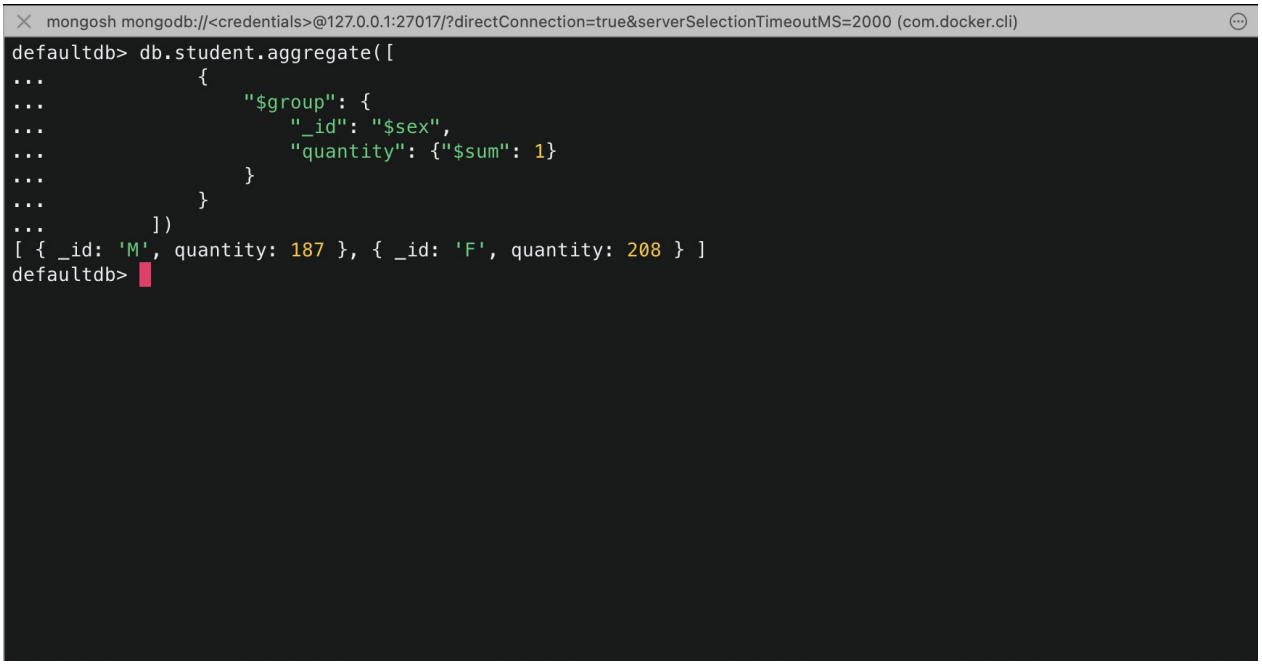
Từ dữ liệu dataset trên chúng ta có thể trực quan hóa dữ liệu để thể hiện xu hướng, tìm ra insightful của data.



Hình 4: Trực quan hóa dữ liệu

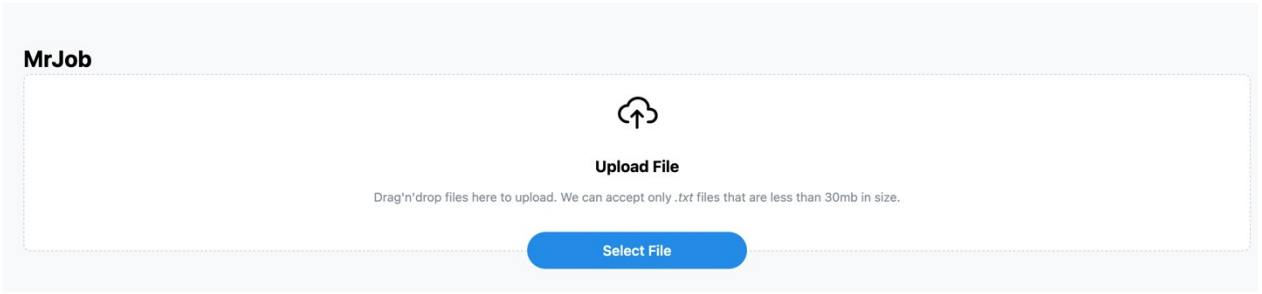


Hình 5: Dữ liệu điểm trung bình G1, G2, G3 của từng giới tính lấy từ MongoDB



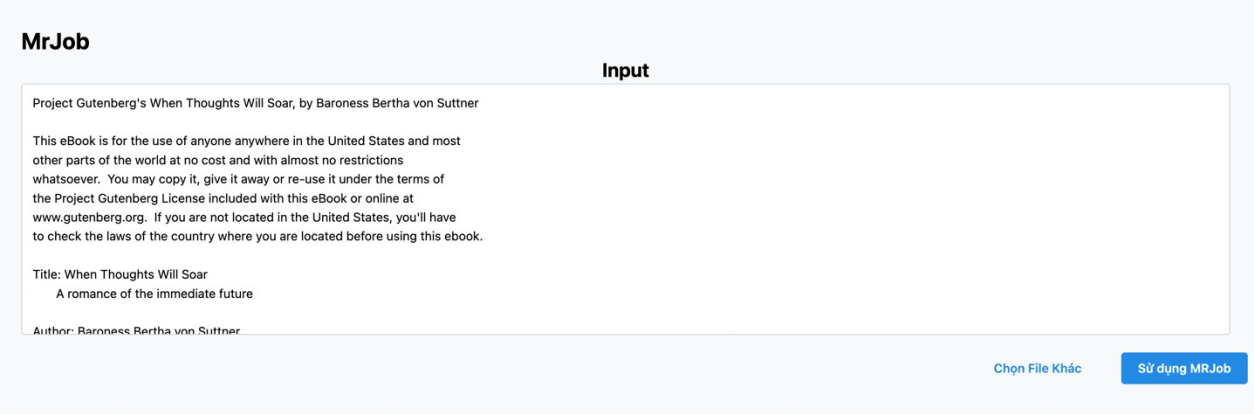
Hình 6: Dữ liệu tổng học sinh từng giới tính lấy từ MongoDB

Mô tả: MrJob là hàm thư viện sử dụng trong ngôn ngữ Python để có thể sử dụng map reduce khi không cần phải cài đặt hadoop trên máy. Ở ứng dụng này chúng ta có thể bỏ một file text có đuôi là .txt để có thể mapping và reduce dữ liệu.



Hình 7: MrJob

Mô tả: Sau khi tải file thành công, UI sẽ hiển thị thông tin chờ người dùng bấm nút chạy MrJob



Hình 8: Upload file

Mô tả: Sau khi sử dụng MrJob kết quả được hiển thị như hình bên dưới:

Hình 9: Kết quả MapReduce

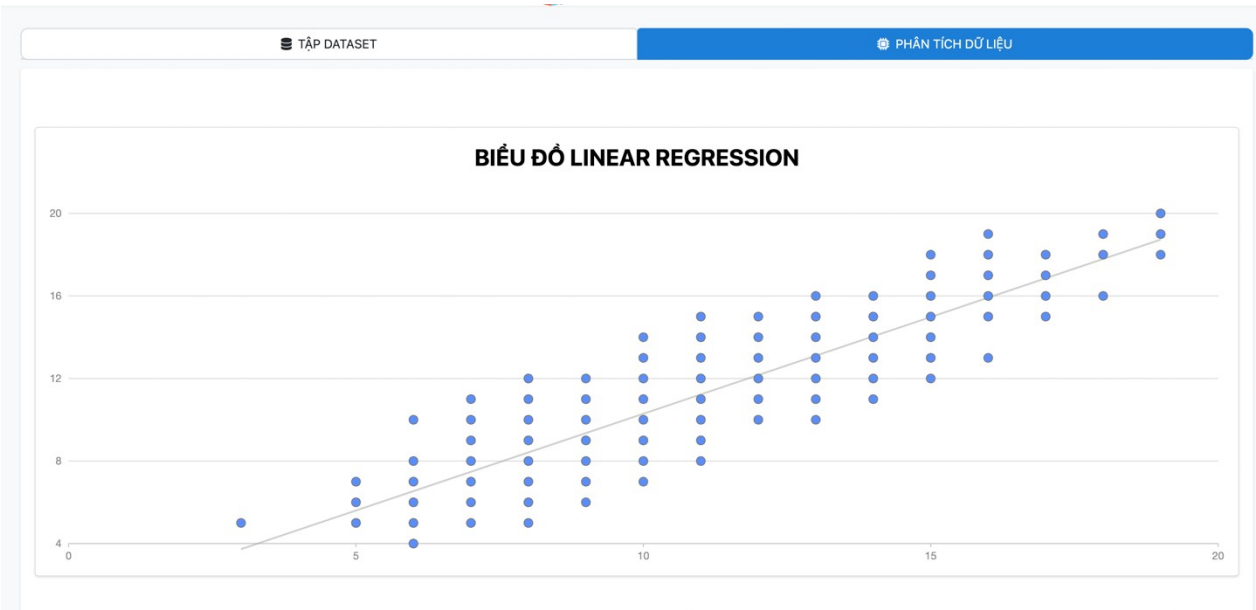
Mô tả: Mã nguồn sử dụng thư viện MrJob

```python
mapping.py > ...
from mrjob.job import MRJob

class Count(MRJob):
    """ The below mapper() function defines the mapper for MapReduce and takes
    key value argument and generates the output in tuple format .
    The mapper below is splitting the line and generating a word with its own
    count i.e. 1 """
    def mapper(self, _, line):
        for word in line.split():
            yield(word, 1)
    """ The below reducer() is aggregating the result according to their key and
    producing the output in a key-value format with its total count"""
    def reducer(self, word, counts):
        yield(word, sum(counts))

if __name__ == "__main__":
    Count.run()
```

Hình 10: Mã nguồn

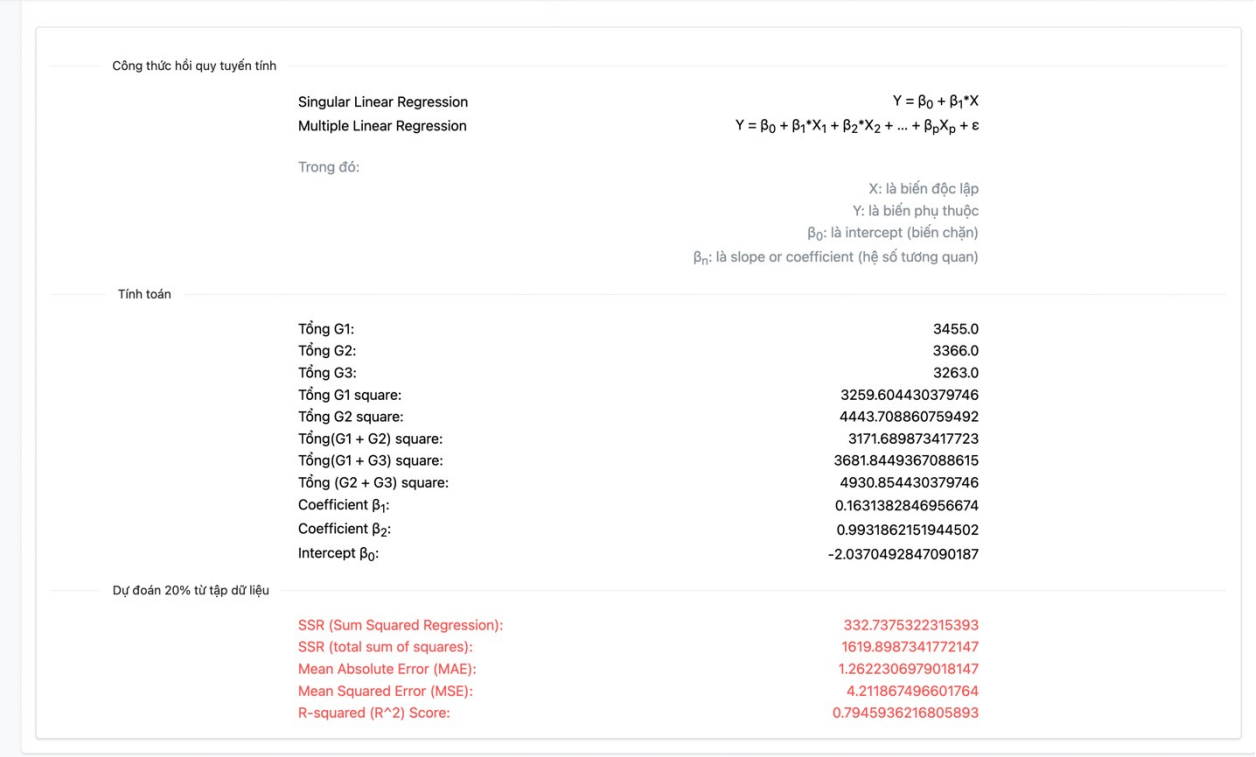
Từ tập dataset sau khi đã được sàng lọc và trực quan hóa thì chúng ta có thể áp dụng dữ liệu cho mô hình học máy (Linear Regression).

Mô tả: Trong biểu đồ scatter plot dưới đây, chúng ta có một đường hồi quy (regression line) được sử dụng để biểu diễn dữ liệu và cho thấy khả năng áp dụng Linear Regression để dự đoán điểm số của học sinh.



Hình 11: Biểu Đồ Linear Regression

Sau khi tính toán kết quả dự đoán như sau:



Hình 12: Tính toán

Mô tả: Sàng lọc dữ liệu, lấy những cột features quan trọng để áp dụng mô hình học máy. Ở trường hợp này lấy những cột G1, G2, G3 và tách train data và test data.

```python
#!/usr/bin/python3

# -*- coding: utf-8 -*-
######################

import sys
import csv
from sklearn.model_selection import train_test_split
from io import StringIO
import pandas as pd



# Read CSV data from sys.stdin
csv_data = sys.stdin.read()

# Create a StringIO object to simulate a file-like object
csv_file = StringIO(csv_data)

# Use pandas read_csv to read the CSV data
csv_reader = pd.read_csv(csv_file, delimiter=';')

g_df = csv_reader.loc[:, ["G1", "G2", "G3"]]
x_filter = g_df.dropna()

x_filter['G1'] = x_filter['G1'].apply(lambda x: str(x).replace(",", ".")).astype(float)
x_filter['G2'] = x_filter['G2'].apply(lambda x: str(x).replace(",", ".")).astype(float)
x_filter['G3'] = x_filter['G3'].apply(lambda x: str(x).replace(",", ".")).astype(float)
X = x_filter.iloc[:, :2]
y = x_filter["G3"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

X_G1_train = X_train["G1"].array
X_G2_train = X_train["G2"].array
Y_train_arr = y_train.array

X_G1_test = X_test["G1"].array
X_G2_test = X_test["G2"].array
Y_test_arr = y_test.array

with open("training_data.txt", "w") as f:
    for row in range(len(X_G1_train)):
        f.write(f"{X_G1_train[row]};{X_G2_train[row]};{Y_train_arr[row]};Train\n")
    for row in range(len(X_G1_test)):
        f.write(f"{X_G1_test[row]};{X_G2_test[row]};{Y_test_arr[row]};Test\n")
```

Hình 13: Mã nguồn

Mô tả: Mã nguồn để mapping dữ liệu trong mapred.

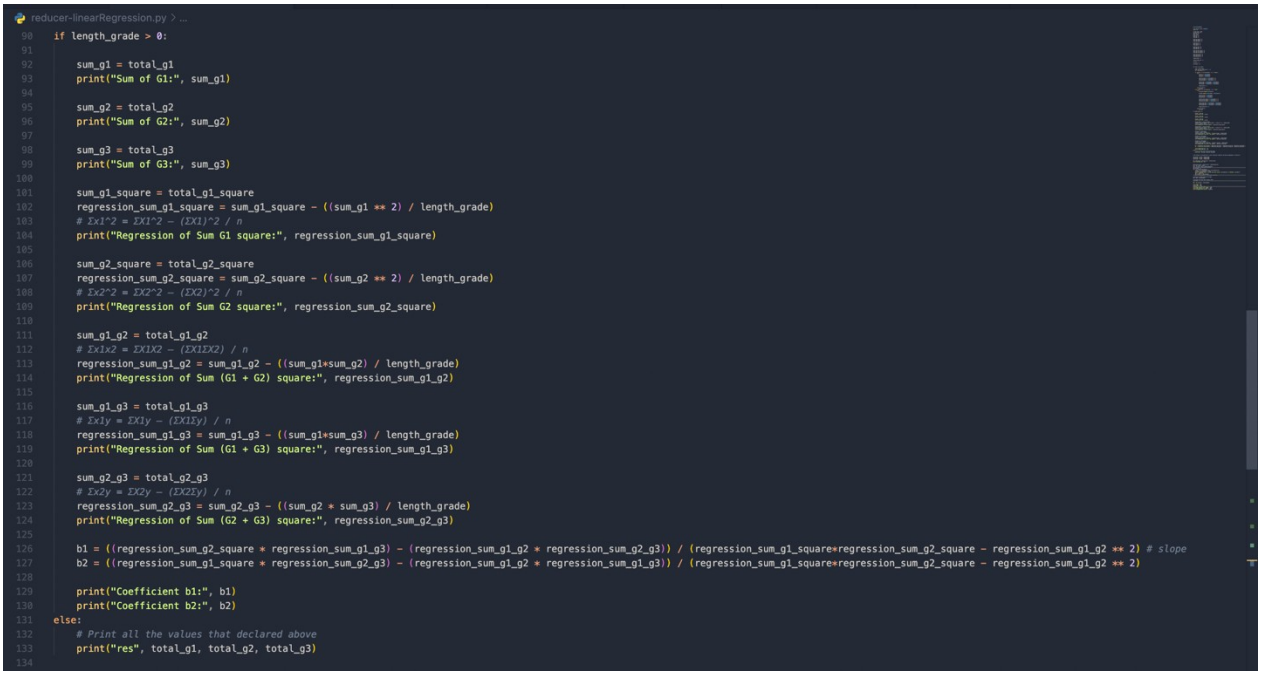
```python
#!/usr/bin/python3

import sys

for line in sys.stdin:
    line = line.strip()
    Gn = line.split(';')
    if len(Gn) == 4:
        print("%s;%s;%s;%s" %(Gn[0], Gn[1], Gn[2], Gn[3]))
```

Hình 14: Mã nguồn

Mô tả: Mã nguồn để reduce và sử dụng Linear Regression để tính toán mô hình học máy và dự đoán điểm số của học sinh.

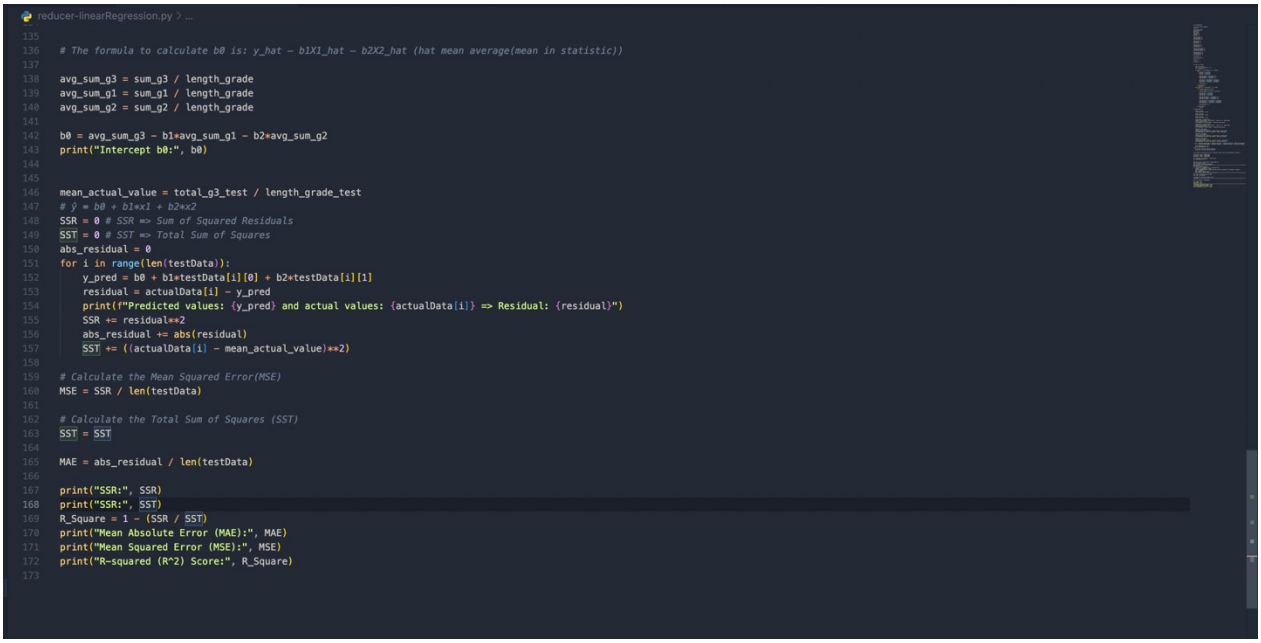
Hình 15: Mã nguồn

Hình 16: Mã nguồn



```
reducer-linearRegression.py > ...
if length_grade > 0:

    sum_g1 = total_g1
    print("Sum of G1:", sum_g1)

    sum_g2 = total_g2
    print("Sum of G2:", sum_g2)

    sum_g3 = total_g3
    print("Sum of G3:", sum_g3)

    sum_g1_square = total_g1_square
    regression_sum_g1_square = sum_g1_square - ((sum_g1 ** 2) / length_grade)
    # Σx1^2 = ΣX1^2 − (ΣX1)^2 / n
    print("Regression of Sum G1 square:", regression_sum_g1_square)

    sum_g2_square = total_g2_square
    regression_sum_g2_square = sum_g2_square - ((sum_g2 ** 2) / length_grade)
    # Σx2^2 = ΣX2^2 − (ΣX2)^2 / n
    print("Regression of Sum G2 square:", regression_sum_g2_square)

    sum_g1_g2 = total_g1_g2
    # Σx1x2 = ΣX1X2 − (ΣX1ΣX2) / n
    regression_sum_g1_g2 = sum_g1_g2 - ((sum_g1*sum_g2) / length_grade)
    print("Regression of Sum (G1 + G2) square:", regression_sum_g1_g2)

    sum_g1_g3 = total_g1_g3
    # Σx1y = ΣX1y − (ΣX1Σy) / n
    regression_sum_g1_g3 = sum_g1_g3 - ((sum_g1*sum_g3) / length_grade)
    print("Regression of Sum (G1 + G3) square:", regression_sum_g1_g3)

    sum_g2_g3 = total_g2_g3
    # Σx2y = ΣX2y − (ΣX2Σy) / n
    regression_sum_g2_g3 = sum_g2_g3 - ((sum_g2 * sum_g3) / length_grade)
    print("Regression of Sum (G2 + G3) square:", regression_sum_g2_g3)

    b1 = ((regression_sum_g2_square * regression_sum_g1_g3) - (regression_sum_g1_g2 * regression_sum_g2_g3)) / (regression_sum_g1_square*regression_sum_g2_square - regression_sum_g1_g2 ** 2) # slope
    b2 = ((regression_sum_g1_square * regression_sum_g2_g3) - (regression_sum_g1_g2 * regression_sum_g1_g3)) / (regression_sum_g1_square*regression_sum_g2_square - regression_sum_g1_g2 ** 2)

    print("Coefficient b1:", b1)
    print("Coefficient b2:", b2)
else:
    # Print all the values that declared above
    print("res", total_g1, total_g2, total_g3)
```

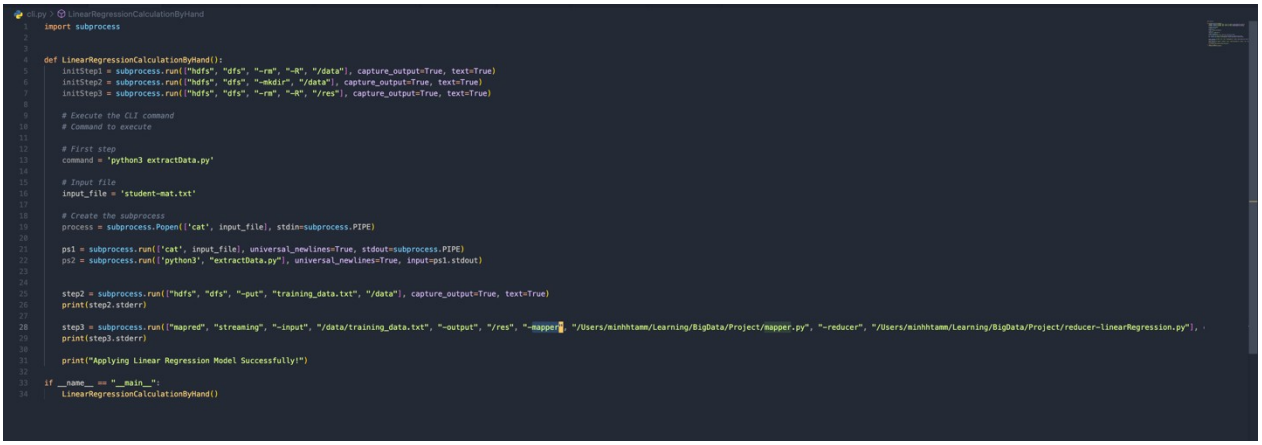
Hình 17: Mã nguồn



```
reducer-linearRegression.py > ...

# The formula to calculate b0 is: y_hat − b1X1_hat − b2X2_hat (hat mean average(mean in statistic))

avg_sum_g3 = sum_g3 / length_grade
avg_sum_g1 = sum_g1 / length_grade
avg_sum_g2 = sum_g2 / length_grade

b0 = avg_sum_g3 - b1*avg_sum_g1 - b2*avg_sum_g2
print("Intercept b0:", b0)


mean_actual_value = total_g3_test / length_grade_test
# ŷ = b0 + b1*x1 + b2*x2
SSR = 0 # SSR => Sum of Squared Residuals
SST = 0 # SST => Total Sum of Squares
abs_residual = 0
for i in range(len(testData)):
    y_pred = b0 + b1*testData[i][0] + b2*testData[i][1]
    residual = actualData[i] - y_pred
    print(f"Predicted values: {y_pred} and actual values: {actualData[i]} => Residual: {residual}")
    SSR += residual**2
    abs_residual += abs(residual)
    SST += ((actualData[i] - mean_actual_value)**2)

# Calculate the Mean Squared Error(MSE)
MSE = SSR / len(testData)

# Calculate the Total Sum of Squares (SST)
SST = SST

MAE = abs_residual / len(testData)

print("SSR:", SSR)
print("SSR:", SST)
R_Square = 1 - (SSR / SST)
print("Mean Absolute Error (MAE):", MAE)
print("Mean Squared Error (MSE):", MSE)
print("R-squared (R^2) Score:", R_Square)
```

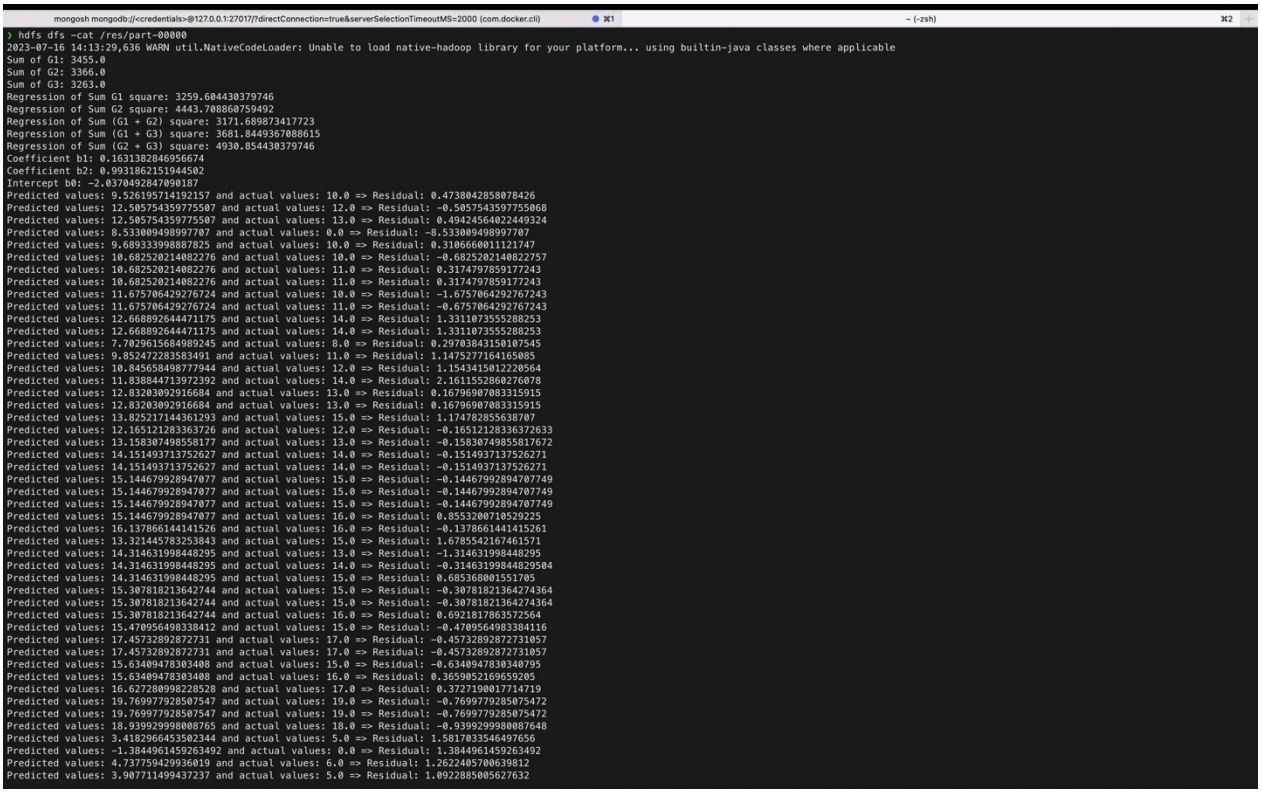
Hình 18: Mã nguồn

Mô tả: sử dụng CLI (command line interface) để sàng lọc dữ liệu và sử dụng map reduce để tính toán mô hình học máy.



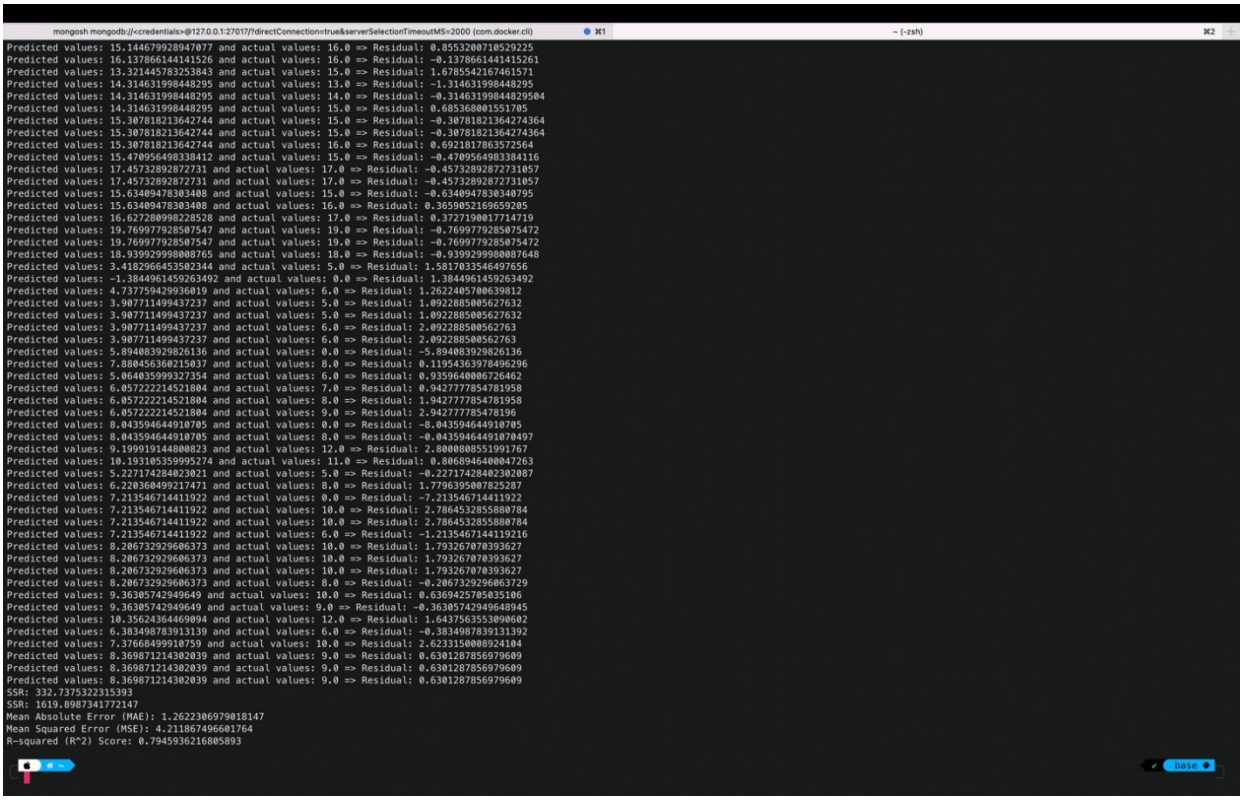
```python
import subprocess


def LinearRegressionCalculationByHand():
    initStep1 = subprocess.run(["hdfs", "dfs", "-rm", "-R", "/data"], capture_output=True, text=True)
    initStep2 = subprocess.run(["hdfs", "dfs", "-mkdir", "/data"], capture_output=True, text=True)
    initStep3 = subprocess.run(["hdfs", "dfs", "-rm", "-R", "/res"], capture_output=True, text=True)

    # Execute the CLI command
    # Command to execute

    # First step
    command = 'python3 extractData.py'

    # Input file
    input_file = 'student-mat.txt'

    # Create the subprocess
    process = subprocess.Popen(['cat', input_file], stdin=subprocess.PIPE)

    ps1 = subprocess.run(['cat', input_file], universal_newlines=True, stdout=subprocess.PIPE)
    ps2 = subprocess.run(["python3", "extractData.py"], universal_newlines=True, input=ps1.stdout)


    step2 = subprocess.run(["hdfs", "dfs", "-put", "training_data.txt", "/data"], capture_output=True, text=True)
    print(step2.stderr)

    step3 = subprocess.run(["mapred", "streaming", "-input", "/data/training_data.txt", "-output", "/res", "-mapper", "/Users/minhhtamm/Learning/BigData/Project/mapper.py", "-reducer", "/Users/minhhtamm/Learning/BigData/Project/reducer-linearRegression.py"],
    print(step3.stderr)

    print("Applying Linear Regression Model Successfully!")

if __name__ == "__main__":
    LinearRegressionCalculationByHand()
```

Hình 19: Mã nguồn



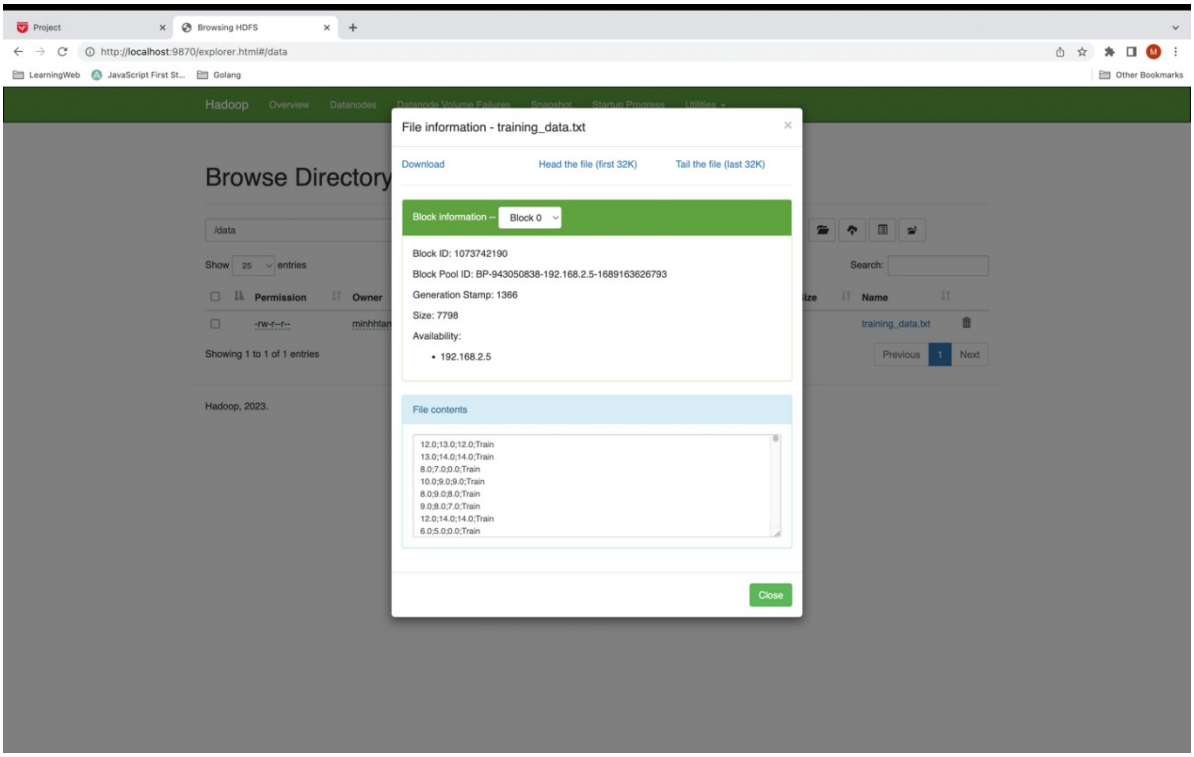
```
mongosh mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000 (com.docker.cli)   ⌘1   ~ (-zsh)   ⌘2

❯ hdfs dfs -cat /res/part-00000
2023-07-16 14:13:29,636 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Sum of G1: 3455.0
Sum of G2: 3366.0
Sum of G3: 3263.0
Regression of Sum G1 square: 3259.604430379746
Regression of Sum G2 square: 4443.708860759492
Regression of Sum (G1 + G2) square: 3171.689873417723
Regression of Sum (G1 + G3) square: 3681.8449367088615
Regression of Sum (G2 + G3) square: 4930.854430379746
Coefficient b1: 0.16313828469566674
Coefficient b2: 0.9931862151944502
Intercept b0: -2.0370492047090187
Predicted values: 9.526195714192157 and actual values: 10.0 => Residual: 0.4738042858078426
Predicted values: 12.505754359775507 and actual values: 12.0 => Residual: -0.5057543597755068
Predicted values: 12.505754359775507 and actual values: 13.0 => Residual: 0.4942456402244932
Predicted values: 8.533009498997707 and actual values: 0.0 => Residual: -8.533009498997707
Predicted values: 9.689333998887825 and actual values: 10.0 => Residual: 0.3106660011121747
Predicted values: 10.682520214082276 and actual values: 10.0 => Residual: -0.6825202140822757
Predicted values: 10.682520214082276 and actual values: 11.0 => Residual: 0.3174797859177243
Predicted values: 10.682520214082276 and actual values: 11.0 => Residual: 0.3174797859177243
Predicted values: 11.675706429276724 and actual values: 10.0 => Residual: -1.6757064292767243
Predicted values: 11.675706429276724 and actual values: 11.0 => Residual: -0.6757064292767243
Predicted values: 12.668892644471175 and actual values: 14.0 => Residual: 1.3311073555288253
Predicted values: 12.668892644471175 and actual values: 14.0 => Residual: 1.3311073555288253
Predicted values: 7.7029615684989245 and actual values: 8.0 => Residual: 0.29703843150107545
Predicted values: 9.852472283583491 and actual values: 11.0 => Residual: 1.1475277164165085
Predicted values: 10.845658498777944 and actual values: 12.0 => Residual: 1.1543415012220564
Predicted values: 11.838844713972392 and actual values: 14.0 => Residual: 2.1611552860276078
Predicted values: 12.83203092916684 and actual values: 13.0 => Residual: 0.16796907083315915
Predicted values: 12.83203092916684 and actual values: 13.0 => Residual: 0.16796907083315915
Predicted values: 13.825217144361293 and actual values: 15.0 => Residual: 1.1747828556638707
Predicted values: 12.16512128336372 6 and actual values: 12.0 => Residual: -0.16512128336372633
Predicted values: 13.158307498558177 and actual values: 13.0 => Residual: -0.15830749855817672
Predicted values: 14.151493713752627 and actual values: 14.0 => Residual: -0.1514937137526271
Predicted values: 14.151493713752627 and actual values: 14.0 => Residual: -0.1514937137526271
Predicted values: 15.144679928947077 and actual values: 15.0 => Residual: -0.14467992894707749
Predicted values: 15.144679928947077 and actual values: 15.0 => Residual: -0.14467992894707749
Predicted values: 15.144679928947077 and actual values: 15.0 => Residual: -0.14467992894707749
Predicted values: 15.144679928947077 and actual values: 16.0 => Residual: 0.8553200710529225
Predicted values: 16.137866144141526 and actual values: 16.0 => Residual: -0.1378661441415261
Predicted values: 13.321445783253843 and actual values: 15.0 => Residual: 1.6785542167461571
Predicted values: 14.314631998448295 and actual values: 13.0 => Residual: -1.314631998448295
Predicted values: 14.314631998448295 and actual values: 14.0 => Residual: -0.31463199844829504
Predicted values: 14.314631998448295 and actual values: 15.0 => Residual: 0.6853680001551705
Predicted values: 15.307818213642744 and actual values: 15.0 => Residual: -0.3078182136427436 4
Predicted values: 15.307818213642744 and actual values: 15.0 => Residual: -0.3078182136427436 4
Predicted values: 15.307818213642744 and actual values: 16.0 => Residual: 0.6921817863572564
Predicted values: 15.470956498338412 and actual values: 15.0 => Residual: -0.4709564983384116
Predicted values: 17.45732892872731 and actual values: 17.0 => Residual: -0.4573289287273105 7
Predicted values: 17.45732892872731 and actual values: 17.0 => Residual: -0.4573289287273105 7
Predicted values: 15.634094783034 08 and actual values: 15.0 => Residual: -0.6340947830340795
Predicted values: 15.634094783034 08 and actual values: 16.0 => Residual: 0.3659052169659205
Predicted values: 16.627280998228528 and actual values: 17.0 => Residual: 0.3727190017714719
Predicted values: 19.769977928507547 and actual values: 19.0 => Residual: -0.7699779285075472
Predicted values: 19.769977928507547 and actual values: 19.0 => Residual: -0.7699779285075472
Predicted values: 18.939929998008765 and actual values: 18.0 => Residual: -0.9399299980087648
Predicted values: 3.4182966453502344 and actual values: 5.0 => Residual: 1.5817033546497656
Predicted values: -1.3844961459263492 and actual values: 0.0 => Residual: 1.3844961459263492
Predicted values: 4.737759429936019 and actual values: 6.0 => Residual: 1.2622405700639812
Predicted values: 3.907711499437237 and actual values: 5.0 => Residual: 1.0922885005627632
```

Hình 20: Kết quả



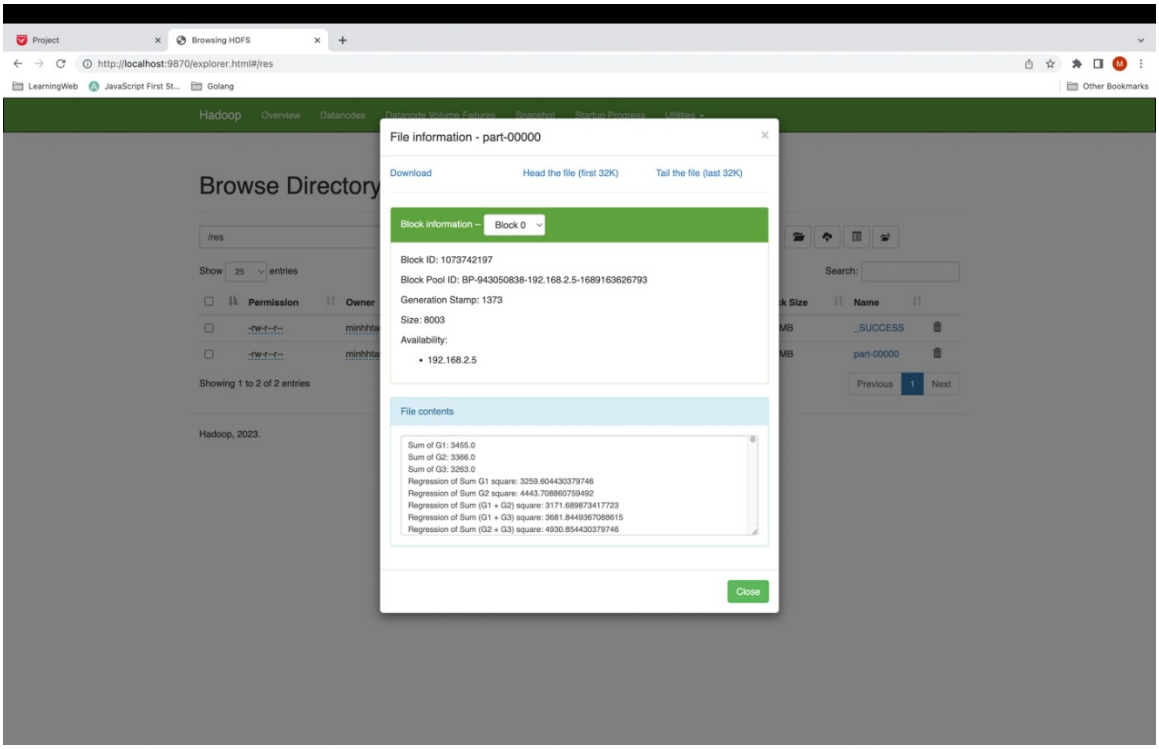
Hình 21: Kết quả

Mô tả: Kết quả trên localhost:9870 của tập dữ liệu được tách ra.



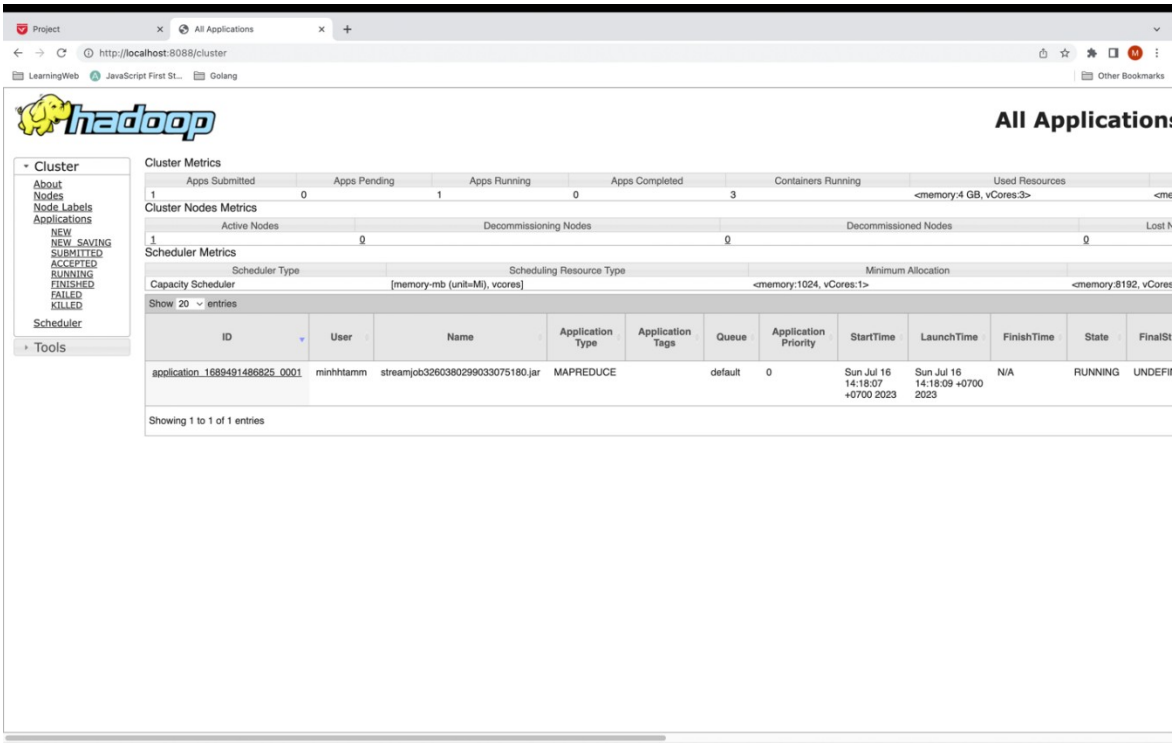
Hình 22: Kết quả

Mô tả: Kết quả được hiện thì trên localhost.



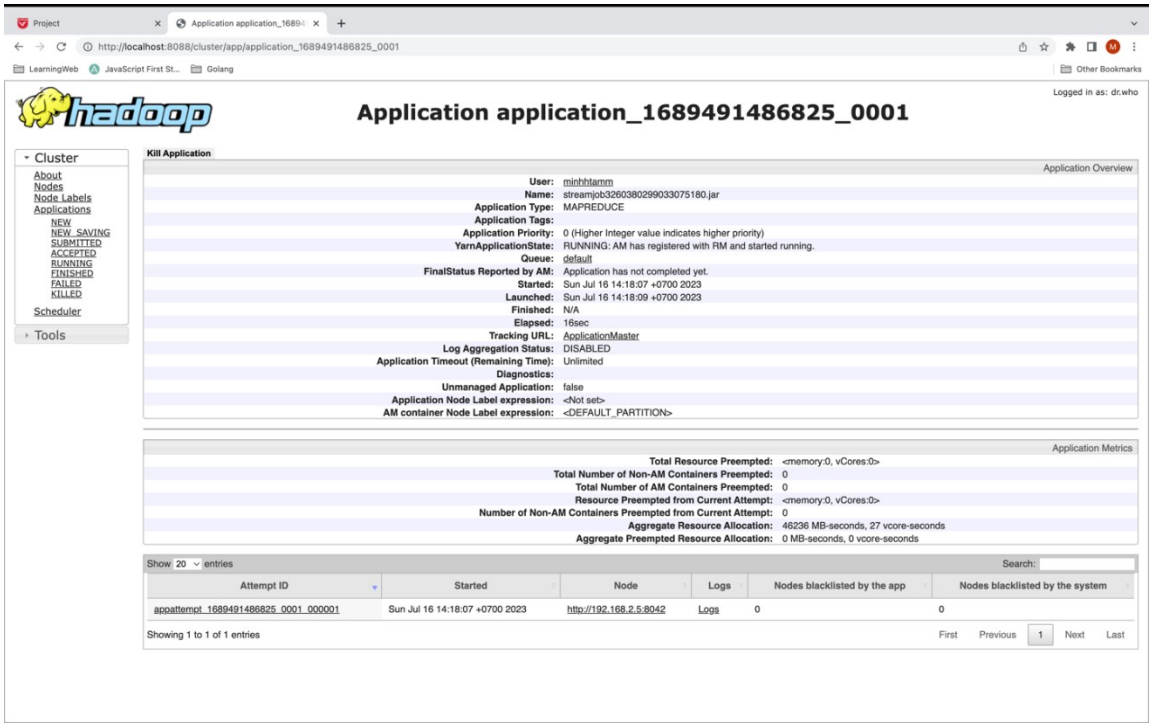
Hình 23: Kết quả

Mô tả: Job công việc của MapReduce trên localhost:8088



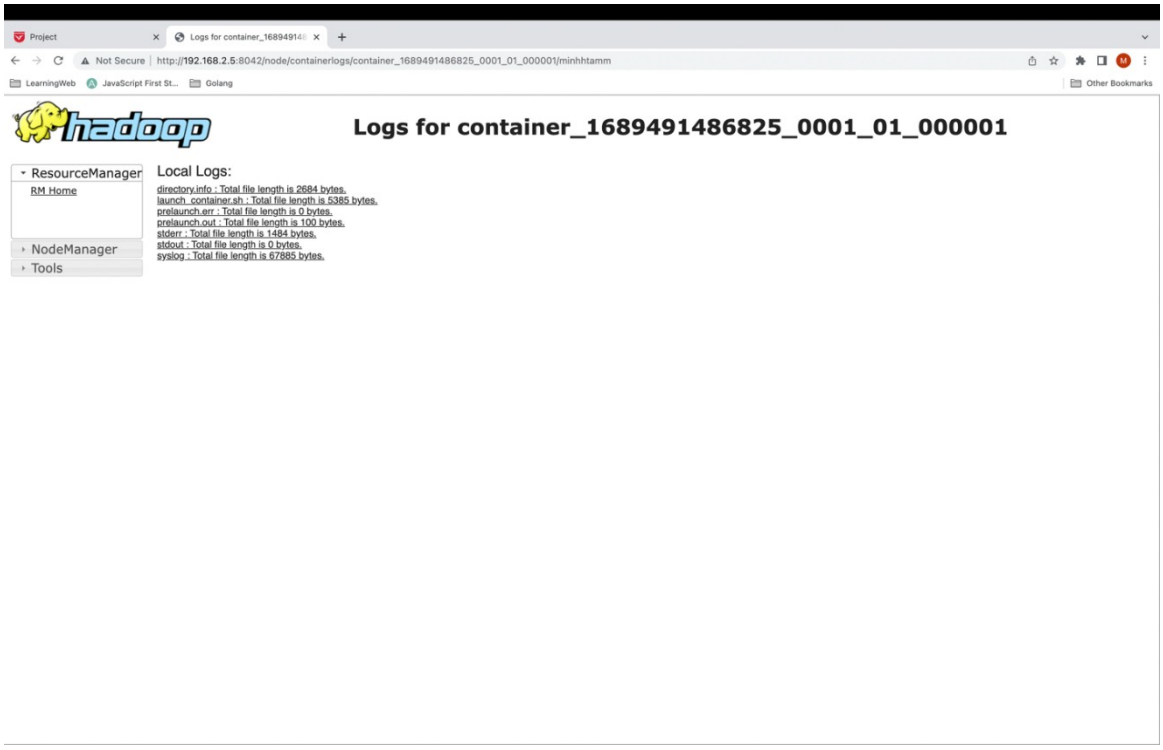
Hình 24: Kết quả

Mô tả: Theo dõi công việc trên hadoop



Hình 25: Kết quả

Mô tả: Logs để tìm ra lỗi hoặc kết quả đầu ra của Job



Hình 26: Kết quả

KẾT LUẬN VÀ ĐỀ XUẤT

1. NHẬN XÉT VỀ HADOOP VÀ NÊU ƯU ĐIỂM VÀ NHƯỢC ĐIỂM CỦA NÓ.

Hadoop là một khung điện toán phân tán mạnh mẽ được thiết kế để xử lý và phân tích khối lượng dữ liệu lớn. Nó cung cấp khả năng mở rộng, khả năng chịu lỗi, hiệu quả chi phí và tính linh hoạt, làm cho nó trở thành lựa chọn phổ biến để xử lý dữ liệu lớn. Với hệ thống tệp phân tán (HDFS), Hadoop cung cấp khả năng lưu trữ có khả năng mở rộng và chịu lỗi, trong khi hệ sinh thái các công cụ và khung mở rộng khả năng của nó cho các khối lượng công việc xử lý dữ liệu khác nhau.

Tuy nhiên, việc thiết lập và quản lý Hadoop có thể phức tạp và có thể không phù hợp để xử lý thời gian thực hoặc độ trễ thấp. Ngoài ra, nó yêu cầu đầu tư cơ sở hạ tầng và có những cân nhắc về đường cong học tập và quản lý dữ liệu. Để dạy học sinh trong lĩnh vực Dữ liệu lớn, các công cụ và phương pháp phù hợp là cần thiết.

Nhìn chung, điểm mạnh của Hadoop nằm ở khả năng mở rộng, khả năng chịu lỗi và khả năng xử lý các tác vụ xử lý dữ liệu đa dạng, khiến nó trở thành một khung có giá trị để xử lý và phân tích dữ liệu lớn.

Ưu điểm	Nhược điểm
Scalability: Hadoop có khả năng mở rộng cao và có thể xử lý khối lượng dữ liệu lớn bằng cách phân phối khối lượng công việc trên nhiều nút trong một cụm. Nó cho phép khả năng mở rộng theo chiều ngang, cho phép các tổ chức thêm nhiều nút hơn vào cụm khi dữ liệu tăng lên, đảm bảo xử lý và lưu trữ hiệu quả.	**Complexity:** Việc thiết lập và quản lý cụm Hadoop có thể phức tạp và đòi hỏi kiến thức cũng như chuyên môn chuyên môn. Nó liên quan đến việc định cấu hình nhiều thành phần, quản lý tài nguyên cụm và đảm bảo hiệu suất và độ tin cậy tối ưu.
Fault Tolerance: Hadoop cung cấp các cơ chế chịu lỗi tích hợp. Nó sao chép dữ liệu trên nhiều nút, đảm bảo tính khả dụng của dữ liệu ngay cả trong trường hợp lỗi nút. Hadoop tự động phân phối lại dữ liệu và tác vụ cho các nút khỏe mạnh, giảm thiểu tác động của lỗi đối với quá trình xử lý dữ liệu.	**Latency:** Hadoop được thiết kế chủ yếu cho khối lượng công việc xử lý hàng loạt, có nghĩa là nó có thể không phù hợp cho các yêu cầu xử lý thời gian thực hoặc độ trễ thấp. Chi phí sao chép dữ liệu, xáo trộn dữ liệu và tính toán phân tán có thể gây ra độ trễ trong kết quả xử lý.

Cost-Effective: Hadoop được thiết kế để chạy trên phần cứng hàng hóa, làm cho nó trở thành một giải pháp hiệu quả về chi phí để xử lý dữ liệu lớn. Nó loại bỏ nhu cầu về phần cứng chuyên dụng đắt tiền và cho phép các tổ chức tận dụng tài nguyên phần cứng giá cả phải chăng để xây dựng cụm Hadoop.	**Steep Learning Curve:** Hadoop và hệ sinh thái của nó có đường cong học tập dốc cho các nhà phát triển và quản trị viên. Việc hiểu các khái niệm, mô hình lập trình và công cụ liên quan đến Hadoop đòi hỏi thời gian và công sức.
Flexibility in Data Processing: Hadoop hỗ trợ nhiều tác vụ xử lý dữ liệu, bao gồm xử lý hàng loạt, xử lý thời gian thực, học máy, v.v. Hệ sinh thái của nó bao gồm các công cụ như MapReduce, Apache Spark, Hive và Pig, mang lại tính linh hoạt và khả năng mở rộng cho các yêu cầu xử lý khác nhau.	**Hardware and Infrastructure Requirements:** Hadoop yêu cầu một cụm máy, có thể liên quan đến các khoản đầu tư cơ sở hạ tầng và phần cứng trả trước. Ngoài ra, việc quản lý cụm Hadoop quy mô lớn có thể yêu cầu tài nguyên lưu trữ và tính toán đáng kể.
Distributed Storage: Hệ thống tệp phân tán Hadoop (HDFS) cung cấp một hệ thống lưu trữ phân tán và có khả năng chịu lỗi. Nó cho phép lưu trữ và xử lý các tập dữ liệu lớn trên nhiều nút, mang lại độ sẵn sàng và độ bền dữ liệu cao.	**Data Management:** Mặc dù Hadoop cung cấp một hệ thống lưu trữ phân tán và có thể mở rộng, nhưng nó có thể không phù hợp nhất với mọi loại dữ liệu. Hadoop phù hợp hơn với dữ liệu phi cấu trúc và bán cấu trúc, nhưng nó có thể không tối ưu cho dữ liệu có cấu trúc cao hoặc xử lý giao dịch.
Data Locality: Nguyên tắc định vị dữ liệu của Hadoop nhằm mục đích xử lý dữ liệu trên cùng một nút nơi dữ liệu được lưu trữ. Điều này làm giảm truyền dữ liệu qua mạng, giảm thiểu chi phí mạng và cải thiện hiệu suất xử lý tổng thể.	

Hadoop vượt trội trong việc xử lý xử lý hàng loạt quy mô lớn, lưu trữ phân tán và khả năng chịu lỗi, nhưng nó có thể yêu cầu lập kế hoạch cẩn thận, cân nhắc cơ sở hạ tầng và chuyên môn để tận dụng hết tiềm năng của nó.

2. ĐỀ XUẤT TRONG TƯƠNG LAI

Tóm lại, tương lai của Hadoop nằm ở khả năng tích hợp với điện toán đám mây, áp dụng kiến trúc kết hợp, tiến bộ xử lý luồng, vùng chứa và điều phối, khả năng phân tích nâng cao và học máy, cải thiện quản trị và bảo mật dữ liệu cũng như vai trò của nó trong việc kích hoạt kiến trúc hồ dữ liệu. Những tiến bộ và xu hướng này sẽ nâng cao hơn nữa các khả năng và ứng dụng của Hadoop trong việc giải quyết các yêu cầu xử lý dữ liệu lớn phức tạp và thúc đẩy sự đổi mới trong quá trình ra quyết định dựa trên dữ liệu.

TÀI LIỆU THAM KHẢO

[1] bigdataviet, "BIGDATAVIET," BIGDATAVIET, 8 August 2015. [Online]. Available: https://bigdataviet.wordpress.com/2015/08/08/hadoop-la-gi/.

[2] P. N. Nghia, "viblo," Viblo, 22 February 2020. [Online]. Available: https://viblo.asia/p/tim-hieu-ve-hadoop-bJzKmOBXl9N.

[3] Simplilearn, "simplilearn," Simplilearn Solutions, 30 March 2023. [Online]. Available: https://www.simplilearn.com/hadoop-vs-mongodb-article#:~:text=MongoDB%20is%20a%20C%2B%2B%20based,optimizes%20space%20better%20than%20MongoDB..

[4] mongodb, "mongodb," mongodb, [Online]. Available: https://www.mongodb.com/compare/hadoop-vs-mongodb.

LỜI CẢM ƠN

Để hoàn thành tiểu luận này, em xin gửi lời cảm ơn chân thành đến:

Xin cảm ơn giảng viên bộ môn - Thầy Hoàng Lê Minh đã giảng dạy tận tình, chi tiết để em có đủ kiến thức và vận dụng chúng vào bài tiểu luận này.

Do chưa có nhiều kinh nghiệm làm để tài cũng như những hạn chế về kiến thức, trong bài tiểu luận chắc chắn sẽ không tránh khỏi những thiếu sót. Rất mong nhận được sự nhận xét, ý kiến đóng góp, phê bình từ phía Thầy để bài tiểu luận được hoàn thiện hơn.

Lời cuối cùng, em xin kính chúc thầy nhiều sức khỏe, thành công.