TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO GIỮA KỲ MÔN
XỬ LÝ DỮ LIỆU LỚN

Người hướng dẫn: **ThS. NGUYỄN THÀNH AN**

Người thực hiện: **TRƯƠNG BỈNH THUẬN - 52100322**

NGUYỄN ĐÌNH DANH - 52100878

NGUYỄN THANH TÚ - 52100349

ĐẶNG VIẾT TRUNG - 52100342

TRẦN THỊ VẸN - 52100674

Lớp: **21050301**

Khoá: **25**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO GIỮA KỲ MÔN
XỬ LÝ DỮ LIỆU LỚN

Người hướng dẫn: **ThS. NGUYỄN THÀNH AN**

Người thực hiện: **TRƯƠNG BỈNH THUẬN - 52100322**

NGUYỄN ĐÌNH DANH - 52100878

NGUYỄN THANH TÚ - 52100349

ĐẶNG VIẾT TRUNG - 52100342

TRẦN THỊ VẸN - 52100674

Lớp: **21050301**

Khoá: **25**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

LỜI CẢM ƠN

Trong suốt quá trình học tập và rèn luyện, chúng em đã nhận được rất nhiều sự giúp đỡ tận tình, sự quan tâm, chăm sóc của thầy Nguyễn Thành An. Ngoài ra, chúng em còn được thầy truyền đạt những kiến thức về xử lý ảnh hay ho và thú vị, thầy cô còn giúp sinh viên có được nhiều niềm vui trong việc học và cảm thấy thoải mái, … Chúng em xin chân thành cảm ơn các thầy cô rất nhiều trong suốt quá trình học tập này!

Bởi lượng kiến thức của chúng em còn hạn hẹp và gặp nhiều vấn đề trong quá trình học nên báo cáo này sẽ còn nhiều thiếu sót và cần được học hỏi thêm. Chúng em rất mong em sẽ nhận được sự góp ý của quý thầy cô về bài báo cáo này để chúng em rút kinh nghiệm trong những môn học sắp tới. Cuối cùng, chúng em xin chân thành cảm ơn quý thầy cô.

TP Hồ Chí Minh, ngày 05 tháng 03 năm 2024

Sinh viên:

TRƯƠNG BỈNH THUẬN - 52100322

NGUYỄN ĐÌNH DANH - 52100878

NGUYỄN THANH TÚ - 52100349

ĐẶNG VIẾT TRUNG - 52100342

TRẦN THỊ VẸN - 52100674

BÁO CÁO CUỐI KÌ ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là sản phẩm của riêng tôi/chúng tôi và được sự hướng dẫn của thầy Phạm Văn Huy. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 05 tháng 03 năm 2024

Tác giả

(ký tên và ghi rõ họ tên)

Trần Thị Vẹn	*Nguyễn Thanh Tú*
Trương Bỉnh Thuận	*Nguyễn Đình Danh*
Đặng Viết Trung	

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

Phần đánh giá của GV chấm bài

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

TÓM TẮT

Bài báo cáo về xử lý dữ liệu lớn yêu cầu thực hiện một loạt các yêu cầu sử dụng dữ liệu từ tập tin baskets.csv. Dưới đây là mô tả cơ bản và kết quả đạt được khi thực hiện các yêu cầu đó:

Câu 1: Sử dụng RDD

Yêu cầu: Sử dụng RDD trong thư viện PySpark để đọc tập tin baskets.csv và thực hiện các nhiệm vụ.

Kết quả:

- f1: Đọc tập tin baskets.csv và in ra danh sách các món hàng phân biệt theo thứ tự tăng dần và hiển thị 10 món hàng đầu tiên và 10 món hàng cuối cùng.
- f2: Tìm danh sách các món hàng phân biệt và số lần mỗi món hàng được mua, sau đó chọn ra 100 món hàng được mua nhiều nhất và vẽ biểu đồ cột để trực quan hóa số lần mua.
- f3: Tìm số lượng giỏ hàng mà mỗi người dùng đã mua và chọn ra 100 người dùng mua nhiều giỏ hàng nhất, sau đó vẽ biểu đồ cột để trực quan hoá số lượng giỏ hàng.
- f4: Tìm người dùng mua nhiều món hàng phân biệt nhất và món hàng được mua bởi nhiều người dùng nhất.

Câu 2: Sử dụng DataFrame

Yêu cầu: Sử dụng DataFrame (PySpark) để tìm ra danh sách giỏ hàng và số lượng giỏ hàng được mua trong một ngày (date). Vẽ biểu đồ đường để trực quan hoá số lượng giỏ hàng được mua theo ngày.

Câu 3: Sử dụng PCY

Yêu cầu: Sử dụng PySpark để cài đặt lớp đối tượng PCY để thực hiện thuật toán PCY. Tạo ra các cặp phổ biến và luật kết hợp từ danh sách giỏ hàng được xác định từ câu 2.

Kết quả: Lưu DataFrame kết quả chứa các cặp phổ biến và danh sách các luật kết hợp xuống các tệp pcy_frequent_pairs.csv và pcy_association_rules.csv.

MỤC LỤC

DANH MỤC HÌNH ẢNH, BẢNG BIỂU VÀ ĐỒ THỊ

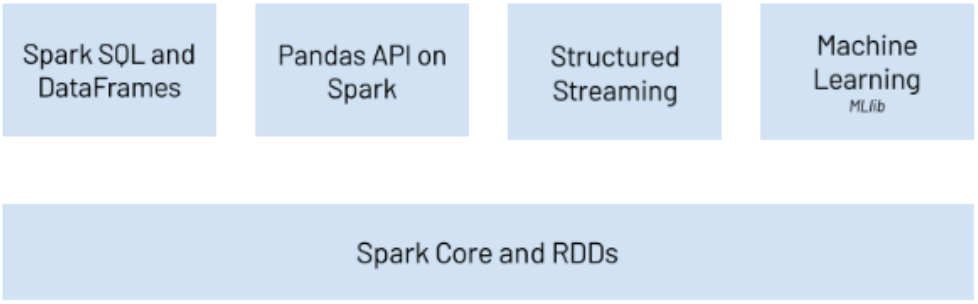
CHƯƠNG 1 - CƠ SỞ LÝ THUYẾT

1.1 Tổng quan về PySpark

PySpark là API Python cho Apache Spark. Nó cho phép bạn thực hiện thời gian thực, xử lý dữ liệu quy mô lớn trong môi trường phân tán bằng Python. Nó cũng cung cấp một PySpark Shell để phân tích tương tác dữ liệu của bạn.

PySpark kết hợp khả năng học hỏi và dễ sử dụng của Python với sức mạnh của Apache Spark để cho phép xử lý và phân tích dữ liệu ở mọi kích thước cho mọi người quen thuộc với Python.

PySpark hỗ trợ tất cả các tính năng của Spark như Spark SQL, DataFrames, Structured Streaming, Machine Learning (MLlib) và Spark Core.



Hình 1.1: Tính năng của Spark

1.2 PySpark – RDD

RDD (Resilient Distributed Dataset) là một cấu trúc dữ liệu cơ bản trong thư viện PySpark, được thiết kế để xử lý dữ liệu trên các cụm dữ liệu phân tán (distributed data clusters). Dưới đây là một số thông tin chi tiết về RDD trong thư viện PySpark:

Khái niệm cơ bản: RDD là một tập hợp các phần tử không thay đổi, được phân tán trên nhiều nút trong cụm dữ liệu. RDD có thể được tạo ra từ các tập dữ liệu có sẵn trong bộ nhớ hoặc từ các phần tử trong hệ thống tệp.

```python
from pyspark import SparkContext

# Khởi tạo SparkContext
sc = SparkContext("local", "example")

# Tạo RDD từ một danh sách Python
data = [1, 2, 3, 4, 5]
rdd = sc.parallelize(data)

# In các phần tử của RDD
print(rdd.collect())  # Output: [1, 2, 3, 4, 5]
```

Hình 1.2: Tạo RDD từ dữ liệu có sẵn trong bộ nhớ

- **Bền vững (Resilient):** Tính bền vững của RDD đề cập đến khả năng tái tạo dữ liệu trong trường hợp một phần của cụm dữ liệu bị mất do lỗi. RDD có thể tái tạo dữ liệu mất bằng cách sử dụng thông tin gốc và các phép biến đổi.
- **Phân tán (Distributed):** RDD được phân tán trên nhiều nút trong cụm dữ liệu, cho phép xử lý song song trên dữ liệu lớn.
- **Khả năng biến đổi (Transformations):** RDD hỗ trợ các phép biến đổi, như map(), filter(), flatMap(), reduceByKey(),... để thực hiện các thao tác xử lý dữ liệu trên toàn bộ RDD hoặc một phần của RDD.

```python
# Áp dụng phép nhân cho mỗi phần tử của RDD
rdd_mapped = rdd.map(lambda x: x * 2)

# In các phần tử đã biến đổi
print(rdd_mapped.collect())  # Output: [2, 4, 6, 8, 10]
```

Hình 1.3: Biến đổi bằng map()

- **Khả năng hành động (Actions):** RDD hỗ trợ các phép hành động như collect(), count(), reduce(), saveAsTextFile(),... để tính toán và trả về kết quả từ dữ liệu.

```
# Sử dụng reduce để tính tổng các phần tử
rdd_sum = rdd.reduce(lambda x, y: x + y)

# In tổng
print(rdd_sum)  # Output: 15
```

Hình 1.4: Tính tổng bằng reduce()

- **Immutability:** RDD là không thay đổi (immutable), có nghĩa là một khi đã tạo, bạn không thể thay đổi nó. Thay vào đó, bạn có thể tạo một RDD mới từ RDD hiện có thông qua các phép biến đổi.
- **Lười biến đổi (Lazy evaluation):** PySpark sử dụng lười biến đổi, nghĩa là các phép biến đổi trên RDD không được thực hiện ngay lập tức khi gọi, mà chỉ khi một phép hành động được gọi. Điều này giúp tối ưu hóa hiệu suất bằng cách chỉ tính toán khi cần thiết.
- **Caching:** RDD có khả năng lưu trữ cache, giúp tăng tốc độ tính toán bằng cách lưu trữ dữ liệu trung gian trong bộ nhớ hoặc đĩa.

1.3 Dataframe

Trong Apache Spark, DataFrame là một cấu trúc dữ liệu phân tán giống như một bảng trong SQL hoặc một bảng Excel, có thể được sử dụng để xử lý và phân tích dữ liệu phân tán. DataFrame cung cấp một giao diện lập trình cao cấp hơn so với RDD, làm cho việc xử lý và truy vấn dữ liệu dễ dàng hơn. Dưới đây là một số khái niệm và tính năng của DataFrame trong PySpark:

- Cấu trúc dữ liệu phân tán: DataFrame là một tập hợp các dữ liệu phân tán, được tổ chức thành các cột có tên.
- Khả năng xử lý dữ liệu có cấu trúc: DataFrame hỗ trợ xử lý dữ liệu có cấu trúc, giúp cho việc làm việc với dữ liệu dạng bảng trở nên dễ dàng.
- Tích hợp với các ngôn ngữ khác nhau: DataFrame có thể được sử dụng trong các ngôn ngữ lập trình khác nhau như Python, Scala hoặc Java.

- Tính tương tự với SQL: DataFrame cung cấp các phương thức cho phép thực hiện các thao tác dữ liệu tương tự như trong SQL như lọc (filter), nhóm (group by), kết hợp (join),...

Hỗ trợ tối ưu hóa hiệu suất: DataFrame có khả năng tối ưu hóa hiệu suất trong các tác vụ xử lý dữ liệu phức tạp thông qua Catalyst Optimizer.

Hỗ trợ đa nguồn dữ liệu: DataFrame có thể đọc dữ liệu từ nhiều nguồn khác nhau như HDFS, S3, Hive, JSON, CSV, Parquet, JDBC, và nhiều nguồn dữ liệu khác.

1.4 PCY

PCY (Park-Chen-Yu) là một thuật toán khai thác luật kết hợp trong mạng nén tập dữ liệu lớn (Frequent Pattern Mining) được phát triển bởi Jian Pei, Jiawei Han và Wei Wang vào năm 2000. Thuật toán này được thiết kế để tìm các luật kết hợp phổ biến giữa các mặt hàng trong tập dữ liệu lớn mà không cần quét lại toàn bộ tập dữ liệu nhiều lần, từ đó giảm thiểu thời gian tính toán và không gian lưu trữ.

Cơ bản, thuật toán PCY hoạt động như sau:

Tính toán bảng băm thứ nhất (Hash Table 1):

- Tạo một bảng băm với kích thước đủ lớn để chứa số lượng ô độc lập tương ứng với tất cả các cặp mặt hàng có thể có.
- Quét tất cả các giao dịch trong tập dữ liệu và tăng giá trị của ô băm tương ứng với cặp mặt hàng được ghép cặp.

Tính toán bảng băm thứ hai (Hash Table 2):

- Tạo một bảng băm thứ hai với kích thước tương tự như bảng băm thứ nhất.
- Quét lại tất cả các giao dịch trong tập dữ liệu, và cho phép ô băm thứ hai tăng giá trị nếu mặt hàng trong cặp đã được tính toán trong bảng băm thứ nhất.

Lọc và trích xuất luật kết hợp:

- Với mỗi cặp mặt hàng có thể, kiểm tra xem số lần xuất hiện của nó vượt qua một ngưỡng tần suất (min_support) đã định trước.

- Kiểm tra lại các mặt hàng trong cặp bằng cách sử dụng bảng băm thứ hai để đảm bảo rằng chúng đã được tính toán trong một số ô băm vượt qua một ngưỡng nhất định (ngưỡng PCY).

CHƯƠNG 2 - ÁP DỤNG

2.1 Câu 1

2.1.1 Yêu cầu 1

Hàm f1(path):

Đọc file và loại bỏ header:

- rdd ← Đọc file tại đường dẫn
- header ← Dòng đầu tiên của rdd
- rdd ← Lọc rdd để loại bỏ header

Tách dữ liệu:

- items_rdd ← Chia mỗi dòng trong rdd theo dấu phẩy (,)

Lấy dữ liệu cần thiết:

- unique_items ← Lấy phần tử thứ 3 (chỉ mục 2 - itemDescription) từ mỗi item trong items_rdd
- unique_items ← Loại bỏ trùng lặp và sắp xếp unique_items (không phân biệt hoa thường)

Ghi dữ liệu:

- đường dẫn kết quả ← "f1"
- Nếu đường dẫn kết quả đã tồn tại:
- Xóa đường dẫn kết quả
- Ghi unique_items vào một file duy nhất tại đường dẫn kết quả

In dữ liệu:

- first_10_name← 10 phần tử đầu tiên của unique_items

- In từng phần tử trong first_10_name trên một dòng riêng
- In dấu phân cách (ví dụ: "--------------------------")
- total_count ← Số lượng phần tử trong unique_items
- last_10_name ← 10 phần tử cuối cùng của unique_items (sử dụng cắt lát)
- In từng phần tử trong last_10_name trên một dòng riêng

Hình 2.1: Sơ đồ thực hiện yêu cầu 1

```
f1(path)
```

```
abrasive cleaner
artif. sweetener
baby cosmetics
bags
baking powder
bathroom cleaner
beef
berries
beverages
bottled beer
--------------------------
UHT-milk
vinegar
waffles
whipped/sour cream
whisky
white bread
white wine
whole milk
yogurt
zwieback
```

Hình 2.2: Kết quả của yêu cầu 1.1

2.1.2 Yêu cầu 2

Hàm f2(path)

Đọc file và loại bỏ header:

- rdd ← Đọc file tại đường dẫn
- header ← Dòng đầu tiên của rdd
- rdd ← Lọc rdd để loại bỏ header

Tách dữ liệu và đếm số lần xuất hiện:

- items_rdd ← Chia mỗi dòng trong rdd theo dấu phẩy (,)
- item_counts ← Với mỗi item trong items_rdd:
- Lấy phần tử thứ ba (chỉ mục 2 - itemDescription) làm key, đặt số lần đếm là 1
- ReduceByKey: Kết hợp số lần đếm cho mỗi key duy nhất, cộng dồn chúng lại

Sắp xếp các item:

- sorted_items ← Sắp xếp item_counts theo số lần đếm (giảm dần)

Ghi dữ liệu đã sắp xếp:

- đường dẫn kết quả ← "f2"
- Nếu đường dẫn kết quả tồn tại: Xóa nó
- output_rdd ← Đặt output_rdd thành sorted_items
- Ghi output_rdd vào một file duy nhất tại đường dẫn kết quả

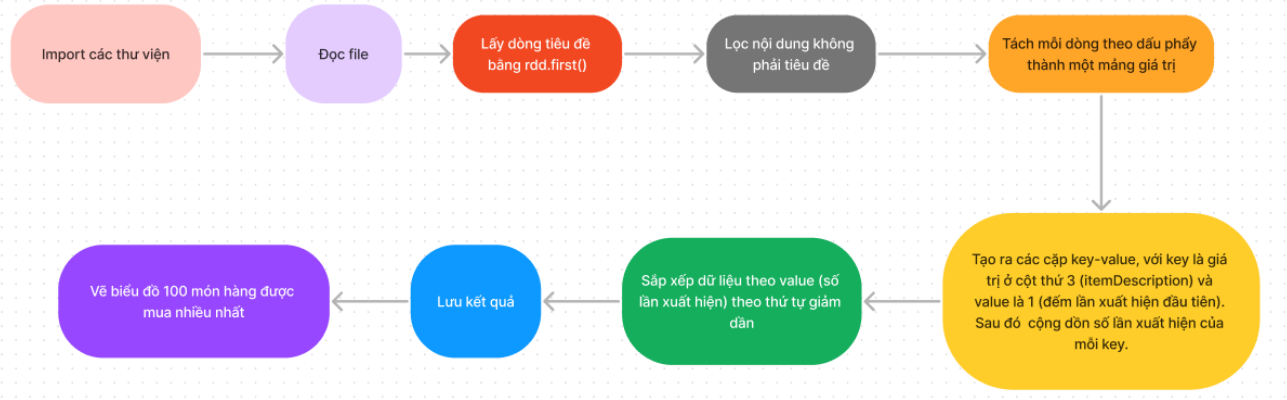
Trích xuất dữ liệu để trực quan hóa:

- top_items ← Lấy 100 phần tử đầu tiên của sorted_items
- item_names ← Trích xuất tên item (phần tử đầu tiên) từ top_items
- item_counts ← Trích xuất số lần xuất hiện của item (phần tử thứ hai) từ top_items

Tạo biểu đồ thanh:

- Tạo một figure với kích thước phù hợp
- Vẽ biểu đồ thanh với item_names trên trục x và item_counts trên trục y
- Đặt nhãn cho các trục và tiêu đề
- Xoay nhãn trục x để dễ đọc hơn

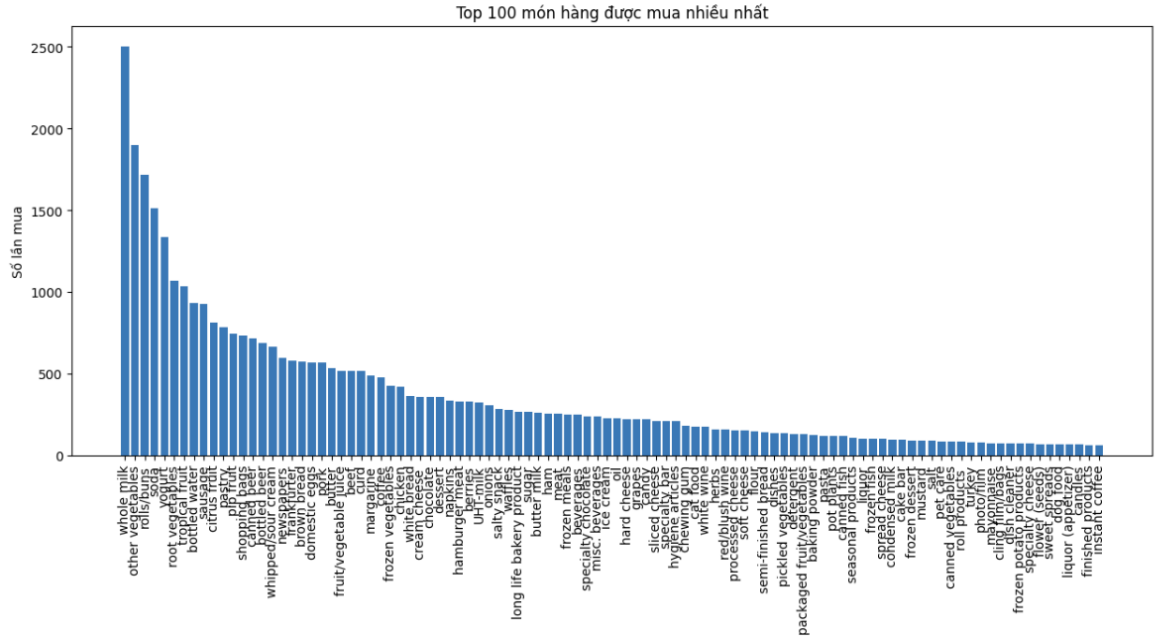
- Hiển thị biểu đồ



Hình 2.3: Sơ đồ thực hiện yêu cầu 2

```
f2(path)
```

```
('whole milk', 2502)
('other vegetables', 1898)
('rolls/buns', 1716)
('soda', 1514)
('yogurt', 1334)
('root vegetables', 1071)
('tropical fruit', 1032)
('bottled water', 933)
('sausage', 924)
('citrus fruit', 812)
```



Hình 2.4: Kết quả của yêu cầu 1.2

2.1.3 Yêu cầu 3

Hàm f3(path)

Đọc file và loại bỏ header:

- rdd ← Đọc file tại đường dẫn
- header ← Dòng đầu tiên của rdd
- rdd ← Lọc rdd để loại bỏ header

Tách dữ liệu và chuẩn bị số lượng giỏ hàng cho mỗi người dùng:

- data_rdd ← Chia mỗi dòng trong rdd theo dấu phẩy (,)
- user_counts ← Với mỗi item trong data_rdd:
- Lấy hai phần tử đầu tiên (ID người dùng, ID giỏ hàng) làm key dạng tuple, đặt số lần đếm là 1
- ReduceByKey (bước 1): Giữ lại ID người dùng làm key, loại bỏ ID giỏ hàng (giả sử ID người dùng là chỉ số quan tâm)
- ReduceByKey (bước 2): Đếm số lần xuất hiện của mỗi ID người dùng, cộng dồn chúng lại

Sắp xếp người dùng:

- sorted_users ← Sắp xếp user_counts theo số lượng giỏ hàng (giảm dần)

Ghi dữ liệu đã sắp xếp:

- đường dẫn kết quả ← "f3"
- Nếu đường dẫn kết quả tồn tại: Xóa nó
- Ghi sorted_users vào một file duy nhất tại đường dẫn kết quả

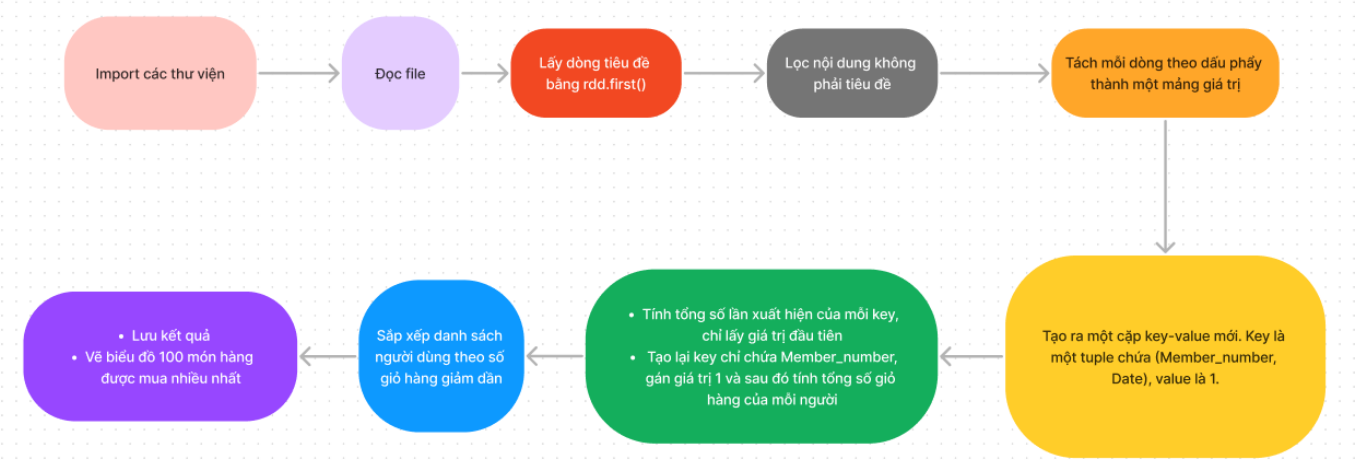
Trích xuất dữ liệu để trực quan hóa:

- top_users ← Lấy 100 phần tử đầu tiên của sorted_users
- user_names ← Trích xuất ID người dùng (phần tử đầu tiên) từ top_users
- basket_counts ← Trích xuất số lượng giỏ hàng (phần tử thứ hai) từ top_users

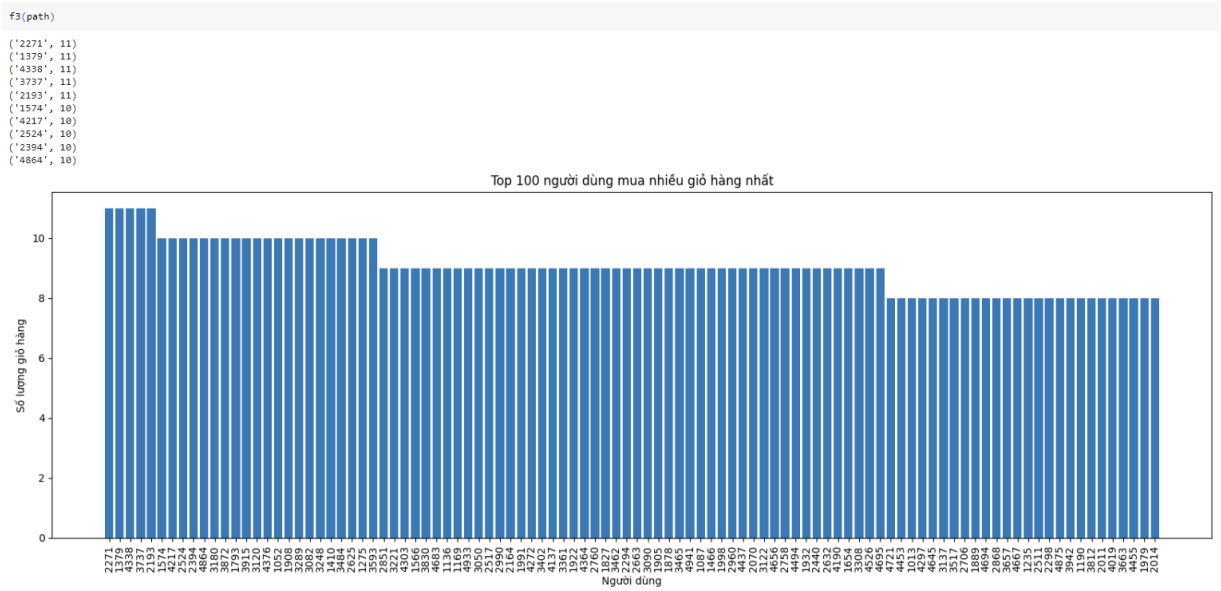
Tạo biểu đồ thanh:

- Tạo một figure với kích thước phù hợp
- Vẽ biểu đồ thanh với user_names trên trục x và basket_counts trên trục y

- Đặt nhãn cho các trục và tiêu đề
- Xoay nhãn trục x để dễ đọc hơn
- Hiển thị biểu đồ



Hình 2.5: Sơ đồ thực hiện yêu cầu 3



Hình 2.6: Kết quả của yêu cầu 1.3

2.1.4 Yêu cầu 4

Hàm f4(path)

Đọc file và loại bỏ header:

rdd ← Đọc file tại đường dẫn

header ← Dòng đầu tiên của rdd

rdd ← Lọc rdd để loại bỏ header

Tìm người dùng mua nhiều món hàng phân biệt nhất:

- data_rdd ← Chia mỗi dòng trong rdd theo dấu phẩy (,)
- user_item_counts ← Với mỗi item trong data_rdd:
- Lấy ID người dùng (phần tử đầu tiên) và tên món hàng (phần tử thứ ba) làm key dạng tuple
- user_item_counts ← Loại bỏ trùng lặp (chỉ giữ lại các món hàng riêng biệt mỗi người dùng mua)
- user_item_counts ← Đặt số lần đếm là 1 cho mỗi cặp người dùng-món hàng
- ReduceByKey: Kết hợp số lần đếm cho mỗi người dùng, cộng dồn số lượng món hàng riêng biệt đã mua

In kết quả cho người dùng mua nhiều món hàng phân biệt nhất:

- user_most_items ← Lấy phần tử đầu tiên (ID người dùng, số lượng) từ số lượng món hàng của người dùng đã được sắp xếp (giảm dần theo số lượng)
- In ID người dùng và số lượng món hàng riêng biệt đã mua

Tìm món hàng được mua bởi nhiều người dùng nhất:

- item_user_counts ← Logic tương tự bước 2, nhưng đổi vị trí ID người dùng và tên món hàng làm key
- item_user_counts ← Loại bỏ trùng lặp (chỉ giữ lại các người dùng riêng biệt mua mỗi món hàng)
- item_user_counts ← Đặt số lần đếm là 1 cho mỗi cặp món hàng-người dùng
- ReduceByKey: Kết hợp số lần đếm cho mỗi món hàng, cộng dồn số lượng người dùng đã mua

In kết quả cho món hàng được mua bởi nhiều người dùng nhất:

- item_most_users ← Lấy phần tử đầu tiên (tên món hàng, số lượng) từ số lượng người dùng mua món hàng đã được sắp xếp (giảm dần theo số lượng)
- In tên món hàng và số lượng người dùng đã mua

Ghi kết quả vào file:

- đường dẫn kết quả ← "f4"
- Nếu đường dẫn kết quả tồn tại: Xóa nó
- output_rdd ← Tạo RDD với kết quả của cả người dùng và món hàng kết hợp
- Ghi output_rdd vào một file duy nhất tại đường dẫn kết quả

Hình 2.7: Sơ đồ thực hiện yêu cầu 4

```
f4(path)
```

```
Người dùng mua nhiều món hàng phân biệt nhất:
Mã người dùng: 2051
Số lượng món hàng: 26
Món hàng được mua bởi nhiều người dùng nhất:
Tên món hàng: whole milk
Số lượng người mua: 1786
```

Hình 2.8: Kết quả của yêu cầu 1.4

2.2 Câu 2

Đọc dữ liệu và tạo DataFrame:

- data ← Đọc file CSV tại đường dẫn (giả sử có hàng tiêu đề)
- Với mỗi hàng trong data:

Trích xuất số thành viên, ngày tháng, và chia ngày tháng thành năm, tháng, ngày

Tạo một bản ghi với các thành phần này

Nhóm dữ liệu theo thành viên, thành phần ngày tháng và tổng hợp các mục:

- Nhóm các bản ghi trong data theo số thành viên, năm, tháng, ngày
- Với mỗi nhóm, thu thập tất cả các mô tả mục duy nhất vào một tập hợp và lưu trữ nó dưới dạng "Giỏ hàng"

Chuyển đổi các thành phần ngày tháng sang kiểu số nguyên:

- Chuyển đổi các cột năm, tháng, ngày trong data sang kiểu dữ liệu số nguyên

Sắp xếp dữ liệu theo ngày tháng:

- Sắp xếp data theo năm, tháng, ngày theo thứ tự tăng dần

Chuẩn bị và hiển thị kết quả:

- Chọn chỉ cột "Giỏ hàng" từ data
- Ghép nối tất cả các mục trong mỗi giỏ hàng (trong cột "Giỏ hàng") bằng dấu phẩy (,) và khoảng trắng làm dấu phân cách, lưu trữ nó dưới dạng "BasketString"
- In tất cả các mục trong data

Lưu giỏ hàng dưới dạng CSV:

- Hợp nhất tất cả các phân vùng dữ liệu thành một phân vùng duy nhất (tối ưu hóa tùy chọn cho tập dữ liệu nhỏ)
- Ghi cột "BasketString" vào một file CSV có tên "baskets"

Vẽ biểu đồ

- ***Đếm số lượng giỏ hàng theo ngày:***

Nhóm dữ liệu trong df theo năm, tháng, ngày

Với mỗi nhóm, đếm số lượng bản ghi (giỏ hàng)

- ***Sắp xếp dữ liệu theo ngày:***

Sắp xếp dữ liệu kết quả theo năm, tháng, ngày theo thứ tự tăng dần

- ***Chuẩn bị dữ liệu để vẽ biểu đồ:***

Chuyển đổi Spark DataFrame df_count sang Pandas DataFrame (pandas_df) để tương thích với việc vẽ biểu đồ

- ***Tạo biểu đồ đường:***

Tạo một figure với kích thước phù hợp

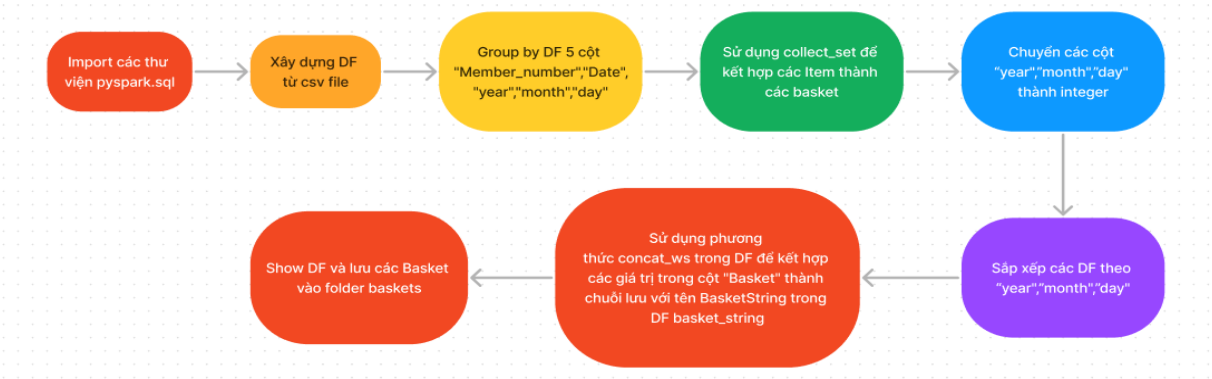
Trích xuất các cột "Ngày" và "Số lượng" từ pandas_df để vẽ biểu đồ

Vẽ biểu đồ đường với "Ngày" trên trục x và "Số lượng" trên trục y

Đặt nhãn cho các trục và tiêu đề

Xoay nhãn trục x để dễ đọc hơn

Hiển thị biểu đồ



Hình 2.9: Sơ đồ thực hiện yêu cầu câu 2

Member_number	Date	year	month	day	Basket
1789	01/01/2014	2014	1	1	[candles, hamburger meat]
2709	01/01/2014	2014	1	1	[yogurt, frozen vegetables]
2943	01/01/2014	2014	1	1	[whole milk, flower (seeds)]
1440	01/01/2014	2014	1	1	[yogurt, other vegetables]
4260	01/01/2014	2014	1	1	[soda, brown bread]
3956	01/01/2014	2014	1	1	[yogurt, shopping bags, waffles, chocolate]
1249	01/01/2014	2014	1	1	[citrus fruit, coffee]
3797	01/01/2014	2014	1	1	[whole milk, waffles]
2974	01/01/2014	2014	1	1	[bottled water, berries, whipped/sour cream]
4942	01/01/2014	2014	1	1	[butter, frozen vegetables]
1381	01/01/2014	2014	1	1	[curd, soda]
1659	01/01/2014	2014	1	1	[specialty chocolate, frozen vegetables]
1922	01/01/2014	2014	1	1	[tropical fruit, other vegetables]
2237	01/01/2014	2014	1	1	[Instant food products, bottled water]
2542	01/01/2014	2014	1	1	[bottled water, sliced cheese]
3681	01/01/2014	2014	1	1	[dishes, onions, whipped/sour cream]
2610	01/01/2014	2014	1	1	[domestic eggs, bottled beer, hamburger meat]
2351	01/01/2014	2014	1	1	[shopping bags, cleaner]
2226	01/01/2014	2014	1	1	[sausage, bottled water]
2727	01/01/2014	2014	1	1	[hamburger meat, frozen potato products]

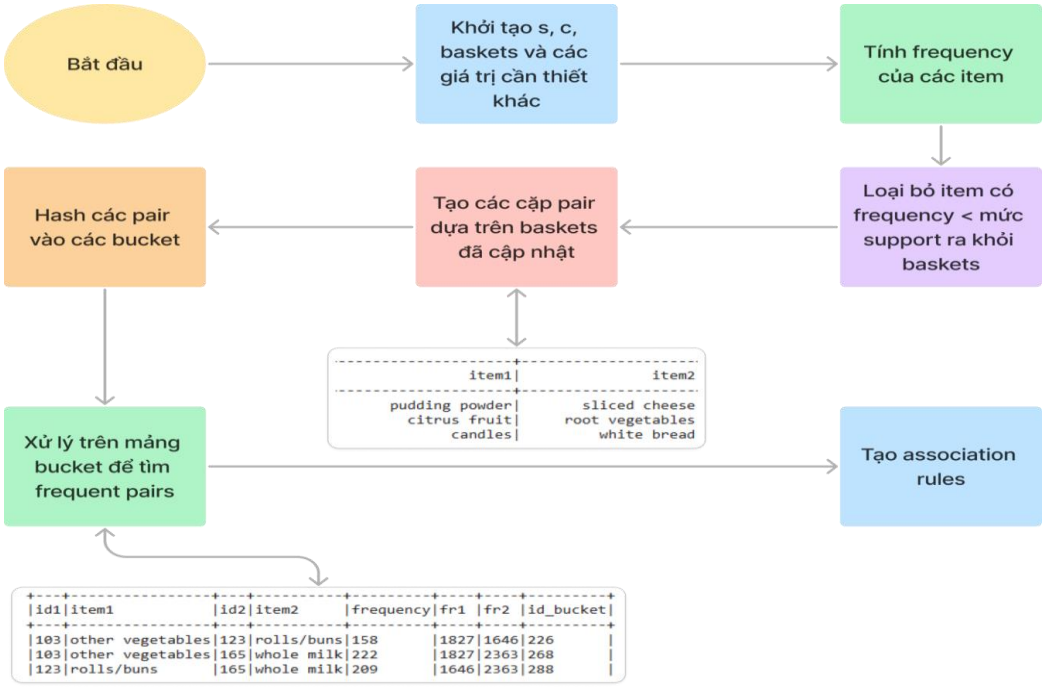
only showing top 20 rows

Hình 2.10: Kết quả các giỏ hàng được xếp theo thứ tự tăng dần năm, tháng, ngày

2.3 Câu 3

2.3.1 Tổng quan thuật toán

Sơ đồ các bước xây dựng class thuật toán PCY:



Hình 2.11: Sơ đồ tổng quát Class PCY

- Hàm __init()__ khởi tạo các giá trị cần thiết như ngưỡng support, confident, link đến output basket của câu 2,...
- Tính toán frequency của từng phần tư riêng biệt, sau đó loại bỏ ra khỏi các basket phần tử nào có:

$$\textbf{support(item)} < \textbf{support threshold}$$

$$\Leftrightarrow \frac{Frequency(item)}{Total\ basket} < Support\ threshold$$

$$\Leftrightarrow Frequency(item) < Support\ threshold \times Total\ basket$$

- Tạo tổ hợp chập 2 các phần tử trong mỗi basket - candidate pair.
- Hash các pair vừa tạo vào các bucket.
- Xử lí trên bucket để tìm các frequent pair:
- Loại bỏ các bucket nào có tổng số frequency của các cặp nhỏ hơn ngưỡng support

$$\textbf{Frequency bucket} < (\textbf{s} \times \textbf{Total basket})$$

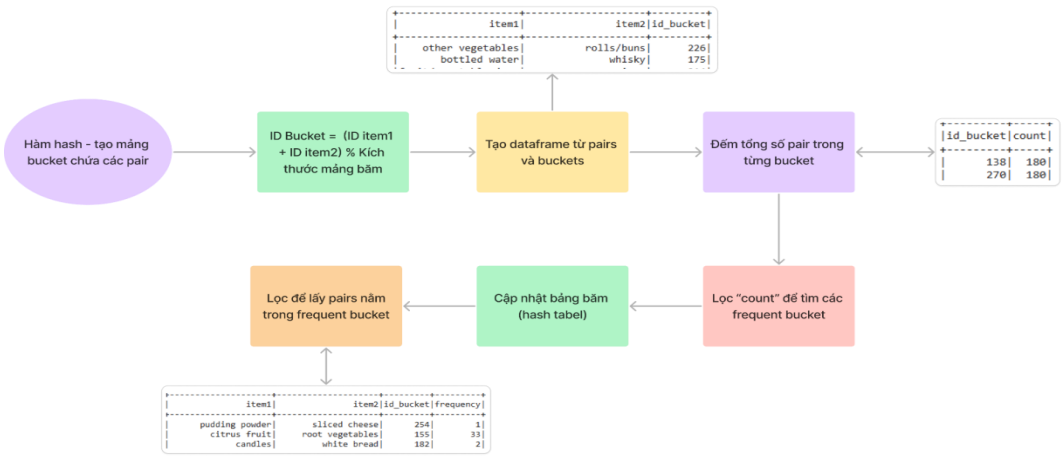
- Lọc trên các bucket còn lại, tìm frequent pair - các cặp thỏa ngưỡng support và confidence.

id1	item1	id2	item2	frequency	fr1	fr2	id_bucket
104	other vegetables	167	whole milk	222	1827	2363	1
125	rolls/buns	167	whole milk	209	1646	2363	2

- Tạo association rule.

2.3.2 Hash functions

Sơ đồ các bước của xử lý bằng Hash fucntions:



Hình 2.12: Sơ đồ Hash functions

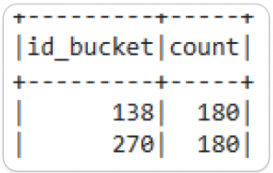
- Hash functions nhận vào 2 tham số là danh sách pairs và danh sách items_id để tạo ra bảng bucket chứa các pair.

$$\boldsymbol{ID\ Bucket = (ID\ Item1 + ID\ Item2)\ mod\ SizeHashTable}$$

Ví dụ: Ta có cặp ID (11,10) và SizeHashTable = 10

=> ID Bucket = (11 + 3) mod 10 = 14 mod 10 = 4

- Sau khi tính toán xong ID Bucket cho từng pair, chúng ta thêm cột ID Bucket đó vào DataFrame.
- Đếm tổng số pair trong từng bucket, sau đó loại bỏ bucket có tổng pair nhỏ hơn ngưỡng support.
    + Một Bucket có thể chứa nhiều pair.
    + Một frequent pair chắc chắc nằm trong một frequent bucket, tuy nhiên một frequent bucket có thể không có frequent pair.
    + Ta có thể tìm frequent bucket dựa trên tổng số lượng pair trong bucket, nếu tổng pair nhỏ hơn ngưỡng support thì chắc chắn trong đó không có frequent pair.



id_bucket	count
138	180
270	180

Hình 2.13: Dataframe của bucket và số pair trong bucket

- Khi thực hiện việc lọc dữ liệu, bảng băm được biểu diễn dưới dạng một vector một chiều với các giá trị có thể là 0 hoặc 1. Ban đầu, bảng băm được khởi tạo với

tất cả các phần tử có giá trị là 0. Sau khi hoàn thành quá trình lọc, bảng băm được cập nhật để phản ánh các bucket thoả mãn và không thoả mãn.

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```

Hình 2.14: Hash table

- Dựa vào bảng băm đã được cập nhật, thực hiện lọc để chọn ra các cặp pair nằm trong bucket lớn hơn ngưỡng support.

item1	item2	id_bucket
other vegetables	rolls/buns	226
bottled water	whisky	175

Hình 2.15: Dataframe pairs trong frequent bucket

2.3.3 Association Rule

Từ 1 cặp (X, Y) ta được 2 rule: $X \rightarrow Y$, $Y \rightarrow X$

Lần lượt tính các giá trị trong association rules với rule $X \rightarrow Y$:

+ Support (Hỗ trợ):

$$\boldsymbol{Support(X,Y) = \frac{Frequency(X,Y)}{Total\ basket}}$$

+ Confidence (Độ tin cậy):

$$\boldsymbol{Support(X,Y) = \frac{Frequency(X,Y)}{Frequency(X)}}$$

+ Lift:

$$Lift(X,Y) = \frac{Support(X,Y)}{Support(X) \times Support(Y)} = \frac{Frequency(X,Y) \times Total\ basket}{Frequency(X) \times Frequency(Y)}$$

Tương tự với rule Y → X

Loại bỏ các rule có:

$Support < Support\ Threshold$ và $Confidence < Confidence\ Threshold$.

Ta được kết quả:

antecedent	consequent	confidence	lift	support
other vegetables	whole milk	0.12151067323481117	0.7694304712706219	0.014836596939116486
rolls/buns	whole milk	0.12697448359659783	0.8040284376030018	0.013967787208447505
whole milk	other vegetables	0.09394837071519255	0.7694304712706219	0.014836596939116486
whole milk	rolls/buns	0.08844688954718578	0.8040284376030018	0.013967787208447505

Hình 2.16: Association rules

CHƯƠNG 3 - TỔNG KẾT

3.1 Đánh giá mức độ hoàn thành yêu cầu

Yêu cầu	Mô tả kết quả làm được	Mức độ hoàn thành
Câu 1	Trực quan hóa dữ liệu, kết quả thành công, in được kết quả đúng với yêu cầu đề bài.	100%
Câu 2	Trực quan hóa dữ liệu, kết quả thành công, in được kết quả đúng với yêu cầu đề bài.	100%
Câu 3	Xây dựng thành công thuật toán PCY để tìm các frequent pairs trong tập dữ liệu. Áp dụng thuật toán PCY để tạo ra association rules từ các frequent pairs. In ra được kết quả theo yêu cầu.	100%
Câu 4	Không biết rõ form báo cáo nên còn nhiều thiếu sót.	80%

3.2 Thuận lợi và khó khăn

- Thuận lợi:

+ Có kinh nghiệm làm việc với Spark từ trước, giúp giảm thiểu thời gian làm quen với môi trường làm việc.

+ Có kiến thức nền vững về các thuật toán khai phá dữ liệu giúp dễ dàng tiếp cận và hiện thực thuật toán PCY.

- Khó khăn:

+ Thiếu tài liệu cụ thể và chi tiết về thuật toán PCY, dẫn đến việc phải dựa vào tài liệu tổng quát và kiến thức tự học để hiện thực.

+ Các tài liệu có sự khác biệt về cách giải thích và thực hiện thuật toán PCY, gây ra sự nhầm lẫn và khó khăn trong quá trình hiện thực và đánh giá kết quả.

3.3 Danh sách thành viên

HỌ VÀ TÊN	MSSV	EMAIL
Đặng Viết Trung	52100342	52100342@student.tdtu.edu.vn
Nguyễn Thanh Tú	52100349	52100349@student.tdtu.edu.vn
Trương Bỉnh Thuận	52100322	52100322@student.tdtu.edu.vn
Trần Thị Vẹn	52100674	52100674@student.tdtu.edu.vn
Nguyễn Đình Danh	52100878	52100878@student.tdtu.edu.vn

TÀI LIỆU THAM KHẢO

[1] Tutorialspoint, “PySpark Tutorial,” Tutorialspoint.
Available: https://www.tutorialspoint.com/pyspark/index.htm.

[2] Apache Software Foundation, “Apache Spark Python API Documentation,” Apache Spark Documentation.
Available: https://spark.apache.org/docs/latest/api/python/index.html

[3] Apache Software Foundation, “RDD: Resilient Distributed Datasets,” Apache Spark Documentation.
Available:https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.RDD.html.