

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



**NGUYỄN THỊ HỒNG GẮM - 52100018  
NGUYỄN PHÚC SĨ LUÂN - 52100061  
NGUYỄN TRUNG DŨNG - 52100783  
LÊ ĐÀO DUY TÂN - 52100104  
TRẦN TẤN THÀNH - 52100110**

**BÁO CÁO GIỮA KỲ  
MÔN XỬ LÝ DỮ LIỆU LỚN**

**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**



**NGUYỄN THỊ HỒNG GẮM - 52100018  
NGUYỄN PHÚC SĨ LUÂN - 52100061  
NGUYỄN TRUNG DŨNG - 52100783  
LÊ ĐÀO DUY TÂN - 52100104  
TRẦN TÂN THÀNH - 52100110**

## **BÁO CÁO GIỮA KỲ MÔN XỬ LÝ DỮ LIỆU LỚN**

Người hướng dẫn  
**ThS. Nguyễn Thành An**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

## LỜI CẢM ƠN

Trước hết, chúng em muốn bày tỏ lòng biết ơn sâu sắc đến các thầy cô giáo của trường Đại học Tôn Đức Thắng, người đã luôn dành sự quan tâm và hỗ trợ cho chúng em trong quá trình hoàn thành dự án này. Đặc biệt, chúng em muốn gửi lời cảm ơn đặc biệt đến thầy Nguyễn Thành An vì sự tận tâm và sự hướng dẫn chi tiết giúp đỡ của thầy, đã giúp chúng em hoàn thành dự án này.

Chúng em hiểu rằng với sự hạn chế về thời gian và kinh nghiệm của sinh viên, bài báo cáo này có thể không tránh khỏi những thiếu sót. Chúng em rất mong nhận được sự chỉ dẫn và đóng góp ý kiến từ các thầy cô để chúng em có thể cải thiện và nâng cao kiến thức của mình, từ đó phục vụ tốt hơn cho công việc thực tế trong tương lai.

*TP. Hồ Chí Minh, ngày 15 tháng 03 năm 2024*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

*Nguyễn Thị Hồng Gấm*

*Nguyễn Phúc Sĩ Luân*

*Nguyễn Trung Dũng*

*Lê Đào Duy Tân*

*Trần Tấn Thành*

## CÔNG TRÌNH ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi và được sự hướng dẫn khoa học của ThS. Nguyễn Thành An. 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.

Ngoài ra, trong Dự án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung Dự 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 15 tháng 03 năm 2024*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

*Nguyễn Thị Hồng Gấm*

*Nguyễn Phúc Sĩ Luân*

*Nguyễn Trung Dũng*

*Lê Đào Duy Tân*

*Trần Tấn Thành*

## TÓM TẮT

Báo cáo sẽ trình bày các nội dung như sau:

Câu 1: Sử dụng RDD trong thư viện PySpark:

- Load dữ liệu từ tập tin baskets.csv và đảm bảo đúng cấu trúc.
- Tìm ra các giỏ hàng từ dữ liệu và lưu kết quả theo đúng cấu trúc.

Câu 2: Sử dụng DataFrame trong thư viện PySpark:

- Cài đặt lớp đối tượng với đủ các phương thức và thuộc tính cần thiết.
- Sử dụng DataFrame để tìm ra danh sách giỏ hàng từ dữ liệu đã được load từ tập tin baskets.csv trong câu 1.
- Vẽ biểu đồ đường để trực quan hóa.
- Lưu kết quả theo đúng cấu trúc.

Câu 3: Sử dụng PySpark để cài đặt lớp đối tượng PCY:

- Cài đặt lớp đối tượng PCY với các phương thức và thuộc tính cần thiết.
- Cài đặt thuật toán PCY
- Lưu kết quả theo đúng cấu trúc.

## MỤC LỤC

<b>DANH MỤC HÌNH VẼ .....</b>	<b>vi</b>
<b>DANH MỤC BẢNG BIỂU .....</b>	<b>vii</b>
<b>DANH MỤC CÁC CHỮ VIẾT TẮT .....</b>	<b>viii</b>
<b>CHƯƠNG 1. RDD .....</b>	<b>1</b>
1.1 RDD trong PySpark .....	1
1.2 Phương pháp thực thi bài 1 .....	1
1.2.1 Hàm <i>f1(file_csv)</i> : .....	1
1.2.2 Hàm <i>f2(file_path)</i> : .....	3
1.2.3 Hàm <i>f3(file_path)</i> : .....	4
1.2.4 Hàm <i>f4(file_path)</i> : .....	5
<b>CHƯƠNG 2. DATAFRAME .....</b>	<b>8</b>
2.1 DataFrame trong PySpark .....	8
2.2 Phương pháp thực thi bài 2 .....	8
2.2.1 Đọc dữ liệu và tìm danh sách giỏ hàng: .....	8
2.2.2 Tìm ra số lượng giỏ hàng được mua trong một ngày .....	9
<b>CHƯƠNG 3. PCY .....</b>	<b>11</b>
3.1 Thuật toán PCY là gì? .....	11
3.2 Các bước thực hiện .....	11
3.3 Phương pháp thực thi bài 3 .....	12
3.3.1 Phương thức khởi tạo trong class <i>PCY</i> .....	12
3.3.2 Phương thức chạy các bước <i>PCY</i> .....	12
<b>CHƯƠNG 4. PHÂN CÔNG VÀ ĐÁNH GIÁ .....</b>	<b>15</b>

<b>TÀI LIỆU THAM KHẢO .....</b>	<b>16</b>
---------------------------------	-----------

## DANH MỤC HÌNH VẼ

Hình 1.1	In ra 10 phần tử đầu và cuối .....	2
Hình 1.2	Các items theo số lần xuất hiện giảm dần .....	3
Hình 1.3	Biểu đồ 100 món hàng được mua nhiều nhất .....	4
Hình 1.4	Các member number theo số lượng items giảm dần .....	5
Hình 1.5	Biểu đồ 100 người dùng mua nhiều giỏ hàng nhất .....	5
Hình 1.6	Thông tin về người dùng và món hàng .....	7
Hình 2.1	Kết quả danh sách giỏ hàng .....	9
Hình 2.2	Số lượng giỏ hàng mỗi ngày .....	10
Hình 2.3	Biểu đồ đường ra số lượng giỏ hàng được mua trong mỗi ngày .....	10
Hình 3.1	Kết quả PCY .....	14



**DANH MỤC BẢNG BIỂU**

Bảng 4.1	Bảng phân công và đánh giá .....	15
----------	----------------------------------	----

## **DANH MỤC CÁC CHỮ VIẾT TẮT**

CSV	Comma-separated Values
RDD	Resilient Distributed Dataset
PCY	Park-Chen-Yu

## CHƯƠNG 1. RDD

### 1.1 RDD trong PySpark

- RDD là viết tắt của Resilient Distributed Dataset, là một tập dữ liệu phân tán được xử lý song song trong Spark.
- RDD có thể được tạo từ nhiều nguồn dữ liệu khác nhau như file, database, hay streaming data. Tạo RDD từ tập tin CSV bằng `SparkContext.textFile()`.
- Các thao tác cơ bản trên RDD:
  - `map()`: Áp dụng hàm lambda cho từng phần tử RDD.
  - `filter()`: Lọc các phần tử RDD theo điều kiện.
  - `reduceByKey()`: Hợp nhất các giá trị theo khóa.
  - `sortByKey()`: Sắp xếp RDD theo khóa.
  - `take()`: Lấy một số phần tử đầu tiên hoặc cuối cùng của RDD.

### 1.2 Phương pháp thực thi bài 1

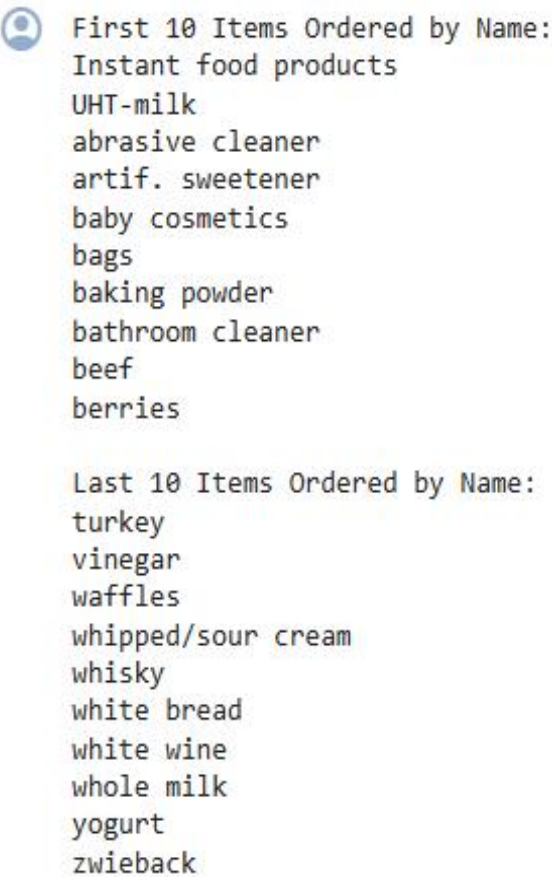
- Tạo RDD từ dữ liệu trong tệp văn bản được chỉ định bởi một đường dẫn “file\_path”.
- “`first()`” dùng để trích xuất dòng đầu tiên từ RDD (dòng đầu tiên là tiêu đề). Từ đó giúp loại bỏ tiêu đề khỏi dữ liệu chính.
- “`filter`” được sử dụng trong RDD để loại bỏ bất kỳ dòng nào mà bằng với tiêu đề.
- “`map`” chuyển đổi mỗi dòng thành một danh sách các giá trị được tách bằng dấu phẩy.
- Kết quả là “data” là một RDD chứa các danh sách các giá trị từ các dòng trong tệp văn bản, mà không bao gồm tiêu đề.

#### 1.2.1 Hàm *f1(file\_csv)*:

- Hàm `f1` nhận đầu vào là một đường dẫn tới một tệp CSV.
- Thực hiện các phương thức xử lý dữ liệu trên RDD:
  - “`map(lambda x: x[2]....)`” lấy phần tử thứ ba của mỗi phần tử trong RDD.

- “distinct()” lọc ra các phần tử duy nhất và sử dụng “sortBy” đã sắp xếp theo tên.
  - “take(10)” lấy 10 phần tử đầu tiên từ RDD đã được xử lý.
  - “top(10)” lấy 10 phần tử lớn nhất từ RDD đã được xử lý. Hàm top lấy từ dưới lên nên phải sắp xếp lại (sorted) để có thứ tự từ trên xuống.
- Thực hiện ghi dữ liệu vào một tệp CSV mới:
- Sử dụng “writer” và “writerow” để ghi kết quả và lưu xuống thư mục fl.
  - Nếu thư mục chứa tệp đầu ra không tồn tại, nó sẽ được tạo mới.
  - Dữ liệu được ghi vào tệp CSV với cấu trúc: “First Item”, “Last Item”, trong đó mỗi hàng chứa một cặp giá trị tương ứng từ hai danh sách “first\_10\_items” và “last\_10\_items”.

```
f1("/content/baskets.csv")
```



```

First 10 Items Ordered by Name:
Instant food products
UHT-milk
abrasive cleaner
artif. sweetener
baby cosmetics
bags
baking powder
bathroom cleaner
beef
berries


Last 10 Items Ordered by Name:
turkey
vinegar
waffles
whipped/sour cream
whisky
white bread
white wine
whole milk
yogurt
zwieback

```

Hình 1.1 In ra 10 phần tử đầu và cuối

### 1.2.2 Hàm *f2(file\_path)*:

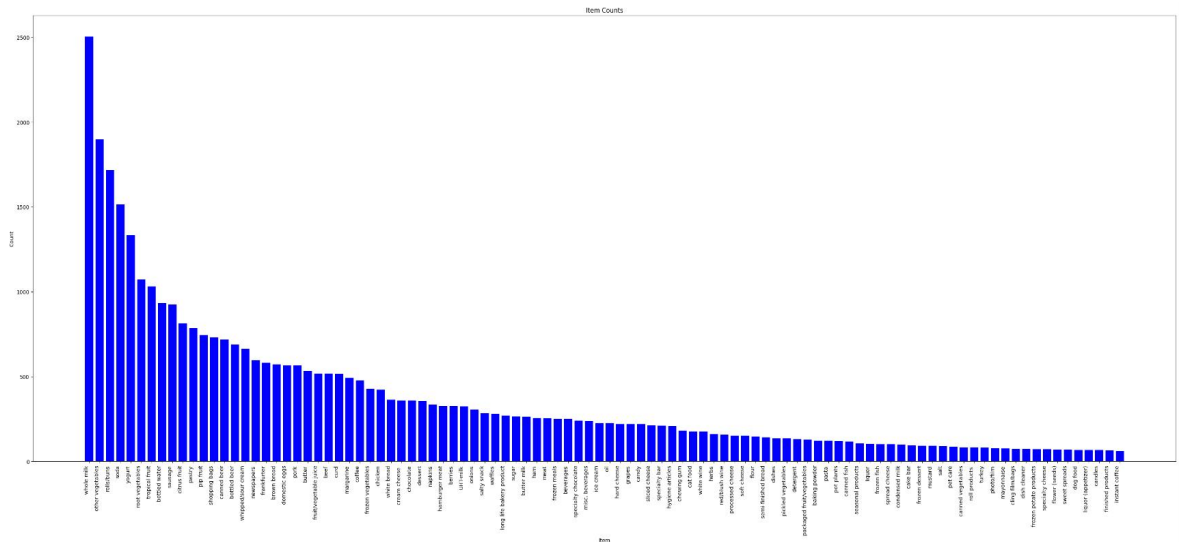
- Hàm *f2* nhận một đường dẫn tới một tệp dữ liệu.
- Thực hiện các phương thức xử lý dữ liệu trên RDD:
  - “*map(lambda x: x[2],1)*” tạo cặp (*itemDescription*, 1) từ mỗi phần tử trong RDD.
  - “*reduceByKey(lambda a, b: a + b)*” tính tổng số lần xuất hiện của mỗi mặt hàng.
  - “*sortBy(...)*” sắp xếp các cặp theo số lần xuất hiện giảm dần bằng “*ascending=False*”
  - “*collect()*” thu thập kết quả vào một danh sách.
  - “*take(100)*” lấy 100 cặp (*itemDescription*, *count*) có số lần xuất hiện nhiều nhất.
- Sử dụng thư viện Matplotlib để tạo và hiển thị biểu đồ với *x* đại diện items và *y* đại diện cho số lần items đó được mua.
- Thực hiện ghi dữ liệu vào một tệp CSV mới trong thư mục *f2* tương tự như hàm *f1* nhưng với dữ liệu được ghi vào tệp CSV là cấu trúc “*Item*” và “*Count*”, trong đó mỗi hàng chứa một cặp giá trị tương ứng từ danh sách các mặt hàng và số lần xuất hiện đã được sắp xếp theo thứ tự giảm dần.



```

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
pastry: 785
pip fruit: 744
  
```

Hình 1.2 Các items theo số lần xuất hiện giảm dần



Hình 1.3 Biểu đồ 100 món hàng được mua nhiều nhất

### 1.2.3 Hàm *f3(file\_path)*:

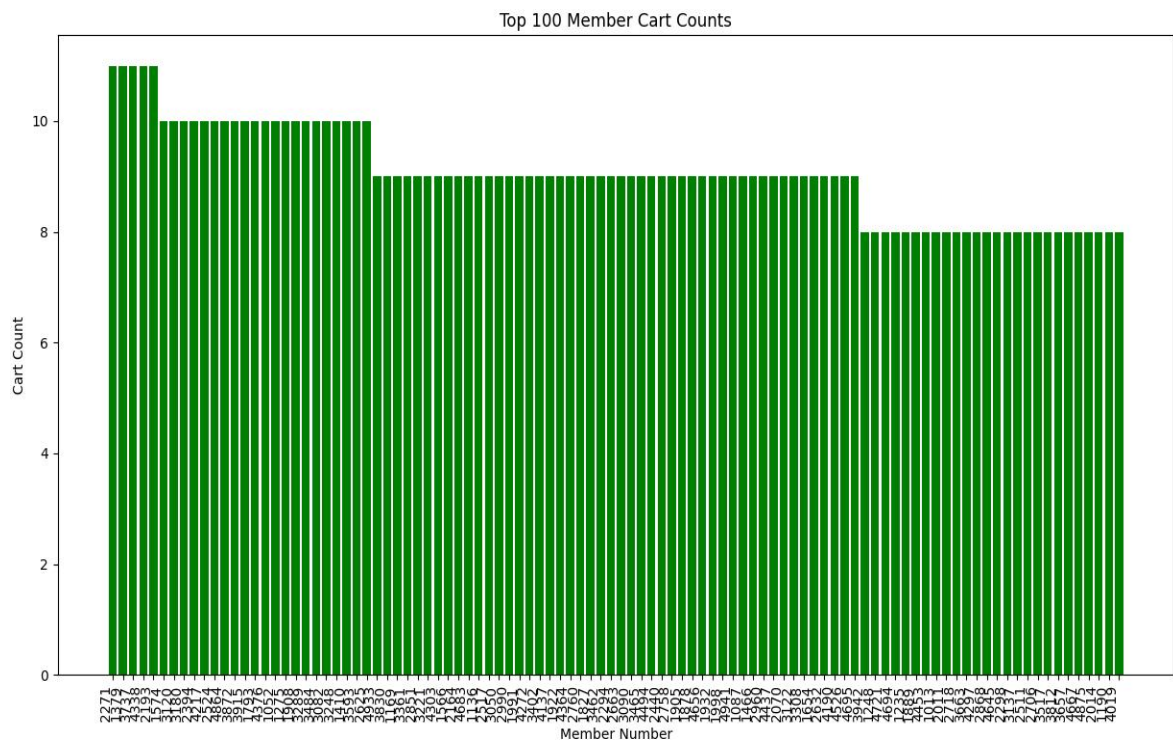
- Hàm f3 nhận một đường dẫn tới một tệp dữ liệu.
- Thực hiện các phương thức xử lý dữ liệu trên RDD:
  - “map(lambda x: (x[0], x[1]))” tạo cặp (member\_number, date) từ mỗi phần tử trong RDD.
  - “groupByKey(...)” nhóm theo member\_number (x[0]) và đếm số lượng ngày khác nhau (len(set(x[1]))).
  - “sortBy()” sắp xếp kết quả theo số lượng giỏ hàng giảm dần bằng “ascending=False”.
  - “take(100)” lấy các giá trị thành viên và số lượng giỏ hàng từ 100 phần tử đầu tiên của RDD.
  - “zip” chuyển các cặp (member\_number, cart\_count) thành hai danh sách riêng biệt, một danh sách chứa các giá trị (member\_number) và một danh sách chứa các giá trị (cart\_count).
- Sử dụng thư viện Matplotlib để tạo và hiển thị biểu đồ với x đại diện cho member\_number và y đại diện cho số lượng giỏ hàng họ mua.
- Thực hiện ghi dữ liệu vào một tệp CSV mới trong thư mục f3 tương tự như hàm f1 nhưng với dữ liệu được ghi vào tệp CSV có cấu trúc "Member Number" và

"Cart Count", trong đó mỗi hàng chứa một cặp giá trị tương ứng từ danh sách các số thành viên và số lượng giỏ hàng đã được sắp xếp theo thứ tự giảm dần.

```
f3("/content/baskets.csv")
```

2271:	11
1379:	11
3737:	11
4338:	11
2193:	11
1574:	10
3120:	10
3180:	10
2394:	10
4217:	10

Hình 1.4 Các member number theo số lượng items giảm dần



Hình 1.5 Biểu đồ 100 người dùng mua nhiều giỏ hàng nhất


#### 1.2.4 Hàm $f4(file\_path)$ :

- Hàm  $f4$  nhận một đường dẫn tới một tệp dữ liệu.
- Tìm ra người dùng mua nhiều món hàng phân biệt nhất:

- “map(lambda x: (x[0], x[2]))” tạo cặp (member, item) từ mỗi phần tử trong RDD.
  - “groupByKey(...)” nhóm theo member (x[0]) và đếm số lượng item duy (len(set(x[1]))).
  - “collect()” đưa dữ liệu thành một danh sách Python.
  - “sorted” sắp xếp theo phần tử thứ 2 (x[1]) trong mỗi phần tử của danh sách (tức là số lượng items mỗi member đã mua)
    - + “reverse=True” để sắp xếp theo thứ tự giảm dần.
    - + “[0][1]” sau khi danh sách đã được sắp xếp, lấy phần tử đầu tiên của danh sách để được số lượng items của member mua nhiều nhất.
  - Dùng vòng “for” để tìm member có số lượng mua hàng bằng với số lượng mua lớn nhất.
- Tìm ra món hàng được mua nhiều nhất:
- “map(lambda x: (x[2], 1))” lấy phần tử thứ 3 (x[2]) chứa tên của item rồi sau đó cho mỗi item ở lần xuất hiện đầu tiên là 1.
  - “reduceByKey(lambda a, b: a + b)” tính tổng số lượng người mua cho mỗi mặt hàng.
  - “sorted” sắp xếp theo phần tử thứ 2 (x[1]) trong mỗi phần tử của danh sách (tức là số lượng đã được mua của mỗi item )
    - + “reverse=True” để sắp xếp theo thứ tự giảm dần.
    - + “[0][1]” sau khi danh sách đã được sắp xếp, lấy phần tử đầu tiên của danh sách để được item có số lượng mua lớn nhất.
  - Dùng vòng “for” để tìm item có số lượng được mua bằng với số lượng của item được mua nhiều nhất.
- Thực hiện ghi dữ liệu vào 2 tệp CSV mới f4\_1.cs và f4\_2.cs tương tự như các câu trên.



```
f4("/content/baskets.csv")
```

 Users with the most distinct items (26 items):  
User 2051: 26 items  
User 1379: 26 items

Items bought by the most users (2502 users):  
Item whole milk: 2502 users

Hình 1.6 Thông tin về người dùng và món hàng

## CHƯƠNG 2. DATAFRAME

### 2.1 DataFrame trong PySpark

- DataFrame là một cấu trúc dữ liệu bảng với các cột và các hàng trong Spark.
- DataFrame được tạo từ RDD, list, dictionary, hoặc từ database.
- Tạo DataFrame:
  - Từ RDD: Sử dụng toDF() method.
  - Từ list: Sử dụng spark.createDataFrame() method.
  - Từ dictionary: Sử dụng spark.createDataFrame() method.
  - Từ database: Sử dụng spark.read.jdbc() method.
- Thực hiện thao tác trên DataFrame:
  - select(): Chọn các cột mong muốn.
  - filter(): Lọc các dòng theo điều kiện.
  - groupBy(): Nhóm các dòng theo một hoặc nhiều cột.
  - agg(): Hợp nhất các giá trị theo nhóm.
  - orderBy(): Sắp xếp DataFrame theo một hoặc nhiều cột.
  - show(): Hiển thị DataFrame ra màn hình.

### 2.2 Phương pháp thực thi bài 2

#### 2.2.1 Đọc dữ liệu và tìm danh sách giỏ hàng:

- Các bước thực hiện:
  - Đọc file từ đường dẫn '/content/baskets.csv' bằng “.read.csv” vào DataFrame.
  - “F\_to\_date()” để chuyển đổi cột “Date” sang định dạng “dd/MM/yyyy”.
  - “orderBy()” sắp xếp DataFrame theo cột “Date” theo thứ tự tăng dần.
  - “groupby()” nhóm các mặt hàng theo “Member\_number” và “Date”, sau đó thu thập danh sách các mặt hàng vào cột “items”
  - “array\_distinct()” được áp dụng trên cột "items" để loại bỏ các phần tử trùng lặp từ mỗi mảng trong cột "items".

- “concat\_ws()” được áp dụng trên cột (“items”) để nối các phần tử trong mảng thành một chuỗi với dấu phân tách là (“, ”).
- Lưu dữ liệu:
  - Ghi DataFrame "Baskets" vào một thư mục với tên "Baskets".
  - “coalesce(1)” đảm bảo rằng chỉ có một tệp kết quả được tạo ra.
  - “write.csv()” để ghi dữ liệu ra file CSV, với “header=True” bao gồm tiêu đề và mode="overwrite" để ghi đè lên các tệp đã tồn tại.

```
Baskets.show(truncate=False)
```

Member_number	Date	items
1249	2014-01-01	citrus fruit, coffee
1381	2014-01-01	curd, soda
1440	2014-01-01	other vegetables, yogurt
1659	2014-01-01	specialty chocolate, frozen vegetables
1789	2014-01-01	hamburger meat, candles
1922	2014-01-01	tropical fruit, other vegetables
2226	2014-01-01	sausage, bottled water
2237	2014-01-01	bottled water, Instant food products
2351	2014-01-01	cleaner, shopping bags
2542	2014-01-01	sliced cheese, bottled water
2610	2014-01-01	hamburger meat, bottled beer, domestic eggs
2709	2014-01-01	yogurt, frozen vegetables
2727	2014-01-01	hamburger meat, frozen potato products
2943	2014-01-01	whole milk, flower (seeds)
2974	2014-01-01	berries, whipped/sour cream, bottled water
3681	2014-01-01	onions, whipped/sour cream, dishes
3797	2014-01-01	waffles, whole milk
3942	2014-01-01	other vegetables, yogurt, Instant food products
3956	2014-01-01	yogurt, shopping bags, waffles, chocolate
4260	2014-01-01	soda, brown bread

only showing top 20 rows

Hình 2.1 Kết quả danh sách giỏ hàng

### 2.2.2 Tìm ra số lượng giỏ hàng được mua trong một ngày

- Các bước xử lý:
  - “withColumn("Date", to\_date("Date", "dd/MM/yyyy"))” được sử dụng để chuyển đổi cột "Date" sang định dạng ngày tháng.
  - “groupBy()” để nhóm dữ liệu theo cột "Date".
  - “count()” đếm số lượng dòng trong mỗi nhóm.

- “orderBy()” sắp xếp tăng dần kết quả theo cột "Date".
  - “collect()” thu thập kết quả thành một danh sách các hàng.
  - “dict()” được sử dụng để chuyển đổi danh sách các hàng thành một từ điển.
- Sử dụng thư viện Matplotlib để tạo và hiển thị biểu đồ với x đại diện Date và y đại diện cho Number of Baskets.

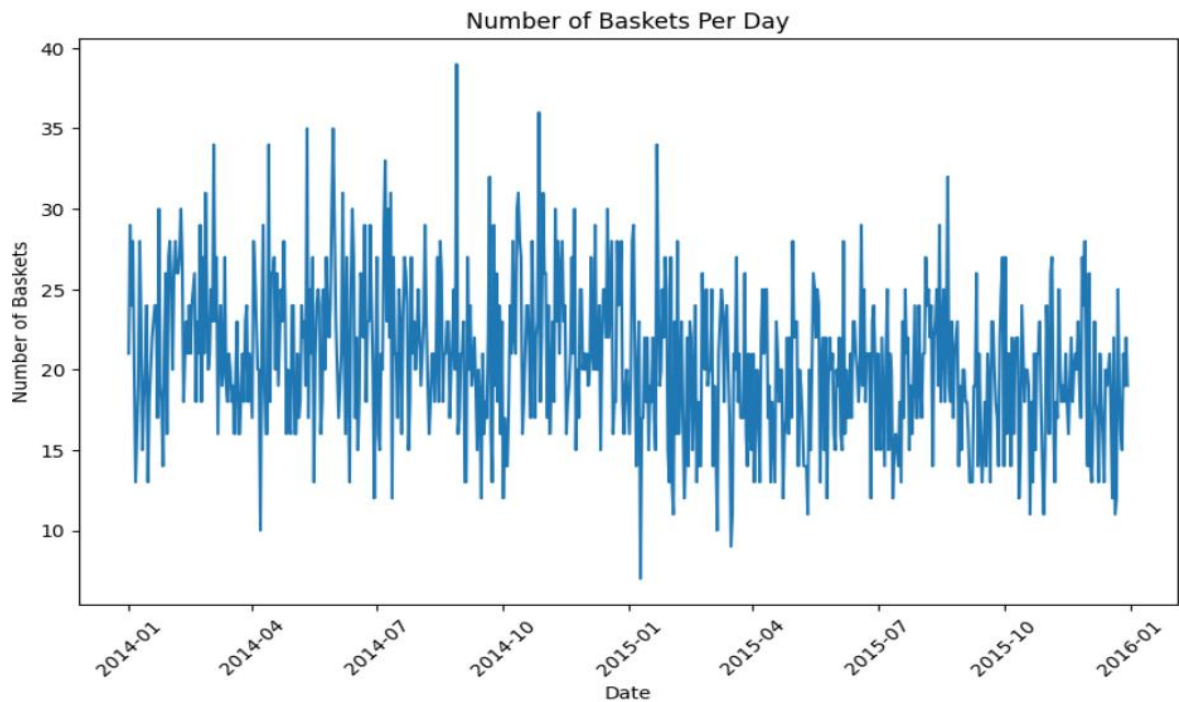
```

Count
2014-01-01    21
2014-01-02    29
2014-01-03    24
2014-01-04    28
2014-01-05    20
...
2015-12-26    15
2015-12-27    21
2015-12-28    19
2015-12-29    22
2015-12-30    19

[728 rows x 1 columns]

```

Hình 2.2 Số lượng giỏ hàng mỗi ngày



Hình 2.3 Biểu đồ đường ra số lượng giỏ hàng được mua trong mỗi ngày

## CHƯƠNG 3. PCY

### 3.1 Thuật toán PCY là gì?

Thuật toán PCY (Park-Chen-Yu) là một phương pháp để tìm kiếm các tập phổ biến trong dữ liệu tập hợp (frequent itemsets) trong việc khai phá dữ liệu. Đây là một phương pháp tối ưu hóa của thuật toán Apriori, một trong những thuật toán đầu tiên và phổ biến nhất trong lĩnh vực này.

### 3.2 Các bước thực hiện

- Tìm các phần tử phổ biến (Vòng 1):
  - Quét qua toàn bộ tập dữ liệu để đếm số lần xuất hiện của từng phần tử.
  - Tính support cho từng phần tử dựa trên số lần xuất hiện chia tổng số bản ghi của tập dữ liệu.
  - Loại bỏ những phần tử không phổ biến (có số lần xuất hiện dưới một ngưỡng được đặt trước gọi là minimum support).
- Tạo và lọc các itemsets phổ biến (Vòng 2):
  - Quét lại toàn bộ tập dữ liệu để đếm số lần xuất hiện của các tập dữ liệu (itemsets) đồng thời tính support cho mỗi tập dữ liệu.
  - Tiến hành băm các tập dữ liệu vào bảng băm (gọi là các bucket) và tính tổng support cho các tập dữ liệu có cùng thuộc một bucket.
  - Loại bỏ đi các bucket có tổng support nhỏ hơn minimum support.
- Tìm cặp phổ biến: sau khi thực hiện pass 2, nhận được danh sách các itemsets phổ biến. Để lấy ra danh sách các cặp phổ biến chỉ cần lọc trong danh sách các itemsets phổ biến có số lượng phần tử của mảng itemset là 2.
- Tính toán quy luật kết hợp:
  - Lấy danh sách các itemsets phổ biến và thực hiện tích Descartes.
  - Điều kiện itemsets bên trái là tập con của itemsets bên phải.
  - Tính confidence cho mỗi quy tắc bằng cách lấy support của tập cha chia support của tập con.
  - Lọc các quy tắc thỏa điều kiện confidence  $\geq$  minimum confidence.

### 3.3 Phương pháp thực thi bài 3

#### 3.3.1 Phương thức khởi tạo trong class PCY

Hàm “\_\_init\_\_(self, file\_path, s, c)”: sử dụng “read.csv” đọc dữ liệu từ tệp csv lưu vào một DataFrame theo đường dẫn “file\_path”, đồng thời lưu lại “s” là ngưỡng support và “c” là ngưỡng confidence.

#### 3.3.2 Phương thức chạy các bước PCY

- Hàm “get\_frequent\_items(self, df\_count)”: nhận số lượng dòng của DataFrame làm đối số đầu vào. Phương thức này tính toán số lần xuất hiện của mỗi item và lọc ra các item phổ biến dựa trên ngưỡng support “s”.
  - Hàm “split” chuyển đổi cột "items" từ chuỗi thành một mảng các phần tử.
  - Chia mảng các phần tử thành các dòng độc lập sử dụng hàm “explode”, mỗi dòng chứa một phần tử từ mảng.
  - Nhóm các dòng theo giá trị của phần tử (item) và đếm số lần xuất hiện của mỗi item trong cột "item".
  - Tính toán support cho mỗi item bằng cách chia số lần xuất hiện của mỗi item cho tổng số dòng của DataFrame.
  - Lọc ra các item phổ biến dựa trên ngưỡng support “s”.
  - Trả về DataFrame chứa các item phổ biến sau khi đã tính toán support và lọc ra các item có support lớn hơn hoặc bằng ngưỡng “s”.
- Hàm “generate\_frequent\_itemsets(self, frequent\_items\_df, df\_count)”:
  - Tạo một RDD từ cột "items" của DataFrame self.df, mỗi phần tử trong RDD là một set chứa các item trong một giao dịch.
  - Phương thức “flatMap” để tạo các itemset có kích thước từ 2 đến độ dài của mỗi giao dịch bằng cách kết hợp các item trong giao dịch.
  - Gán giá trị 1 cho mỗi itemset và sau đó sử dụng phương thức “reduceByKey” để đếm số lần xuất hiện của mỗi itemset.
  - Chuyển đổi kết quả thành DataFrame “itemsets\_df”, trong đó cột "itemset" là một chuỗi các item và cột "support" là số lần xuất hiện của mỗi itemset.

- Tính toán support cho mỗi itemset bằng cách chia số lần xuất hiện của mỗi itemset cho tổng số giao dịch (`df_count`).
  - Sử dụng một hàm hash để tạo các bucket cho mỗi itemset.
  - Tính tổng support cho mỗi bucket bằng “groupBy” và “agg”.
  - “filter” lọc ra các bucket có tổng support lớn hơn hoặc bằng ngưỡng “s”.
  - Lấy ra các itemset từ các bucket đã lọc và loại bỏ các itemset có support nhỏ hơn ngưỡng “s”.
  - Trả về DataFrame chứa các itemset phổ biến khi đã lọc và tính toán support.
- Hàm “*find association rules(self, frequent itemsets df, frequent items df)*”:
- Gộp tất cả các tập và item phổ biến vào một DataFrame.
  - Đổi tên cột để phân biệt giữa các tập bên trái và bên phải trong quy luật.
  - Chuyển các cột chứa chuỗi (string) thành danh sách (list) các phần tử.
  - Kết hợp mỗi itemset bên trái với mỗi itemset bên phải bằng “join”. Các itemset bên trái có ít phần tử (subset) hơn bên phải.
  - Tính tỷ lệ confidence cho mỗi cặp itemset và lọc ra các cặp thỏa điều kiện tỷ lệ confidence lớn hơn hoặc bằng ngưỡng confidence “c”.
  - Tìm consequence bằng cách lọc ra các phần tử bên tập phải không nằm bên tập trái.
  - Đổi tên các cột của DataFrame (`left_itemset`) và (`right_itemset`) lần lượt là “antecedent” và “consequent” để thể hiện rõ ràng ý nghĩa của từng cột.
  - Trả về DataFrame chứa các luật kết hợp thỏa mãn điều kiện.
- Hàm “*save to csv(self, save df, name save)*”:
- Kiểm tra file csv “name\_save.csv” đã tồn tại trong thư mục hiện tại hay chưa.
  - Nếu file CSV đã tồn tại, gọi “subprocess.call” thực hiện xóa file CSV đó. Lệnh này sử dụng lệnh hệ thống rm để xóa file CSV.
  - “toPandas()” để chuyển DataFrame thành DataFrame của thư viện Pandas.
  - “to\_csv” của Pandas để lưu DataFrame thành file CSV với tên đã chỉ định.

- Hàm “run(self)”:

- Dùng “count” đếm số lượng dòng trong DataFrame lưu vào “df\_count”.
- “get\_frequent\_items” tạo DataFrame chứa các item phổ biến.
- Gọi phương thức “generate\_frequent\_itemsets” để tạo DataFrame chứa các tập phổ biến dựa trên các item phổ biến đã tìm thấy và df\_count.
- Tách các itemset từ chuỗi thành danh sách.
- Lọc ra các cặp phổ biến từ DataFrame chứa các itemsets.
- In ra màn hình số lượng và danh sách các cặp phổ biến.
- Gọi phương thức “find\_association\_rules” để tạo DataFrame chứa các luật kết hợp dựa trên các tập phổ biến và các item phổ biến đã tìm thấy.
- Sử dụng “ThreadPoolExecutor” để chạy song song hai nhiệm vụ lưu file:
  - Lưu DataFrame chứa các cặp phổ biến vào “pcy\_frequent\_pairs.csv”.
  - Lưu DataFrame chứa các luật kết hợp vào pcy\_association\_rules.csv.

```
file_path = '/content/Baskets/part-00000-9315cbe8-6655-46'
s = 0.001
c = .2
pcy = PCY(file_path, s, c)
pcy.run()
```

Có 588 cặp phổ biến

Danh sách các cặp phổ biến

pair	support
citrus fruit, coffee	0.0013366303548753592
canned beer, pip fruit	0.001603956425850431
frozen vegetables, tropical fruit	0.0010693042839002875
chocolate, citrus fruit	0.0010693042839002875
pork, tropical fruit	0.0015371249081066632
fruit/vegetable juice, pastry	0.0014702933903628951
citrus fruit, tropical fruit	0.0028737552629820224
newspapers, whipped/sour cream	0.0015371249081066632
coffee, shopping bags	0.001403461872619127
ice cream, soda	0.0012029673193878234
domestic eggs, pastry	0.0013366303548753592
margarine, soda	0.0026064291920069507
curd, whole milk	0.004143554100113613
canned beer, yogurt	0.0038762280291385416
napkins, tropical fruit	0.0012697988371315912
shopping bags, soda	0.004410880171088686
curd, margarine	0.0010693042839002875
brown bread, curd	0.0012029673193878234
candy, yogurt	0.0012029673193878234
citrus fruit, pip fruit	0.0014702933903628951

only showing top 20 rows

Danh sách association rules

antecedent	consequent	confidence
[sausage, yogurt]	[whole milk]	0.2558139534883721
[rolls/buns, sausage]	[whole milk]	0.2125

Hình 3.1 Kết quả PCY



## CHƯƠNG 4. PHÂN CÔNG VÀ ĐÁNH GIÁ

Bảng 4.1 Bảng phân công và đánh giá

STT	MSSV	Họ và tên	Công việc	Đánh giá
SV1	52100018	Nguyễn Thị Hồng Gấm	Câu 1	Hoàn thành
SV2	52100061	Nguyễn Phúc Sĩ Luân	Câu 3	Hoàn thành
SV3	52100783	Nguyễn Trung Dũng	Câu 1	Hoàn thành
SV4	52100104	Lê Đào Duy Tân	Câu 2	Hoàn thành
SV5	52100110	Trần Tấn Thành	Câu 3	Hoàn thành

## TÀI LIỆU THAM KHẢO

### Tiếng Anh

[1] pandas Development Team. (2022). "pandas.DataFrame." pandas 1.4.0 documentation.

[2] The Apache Software Foundation. (2022). "RDD." Apache Spark 0.6.2 Documentation - RDD.

[3] Tutorialspoint. (2022). "PySpark Tutorial."

[4] Slama, J. (2022). "Mining Massive Datasets."