PROTONX

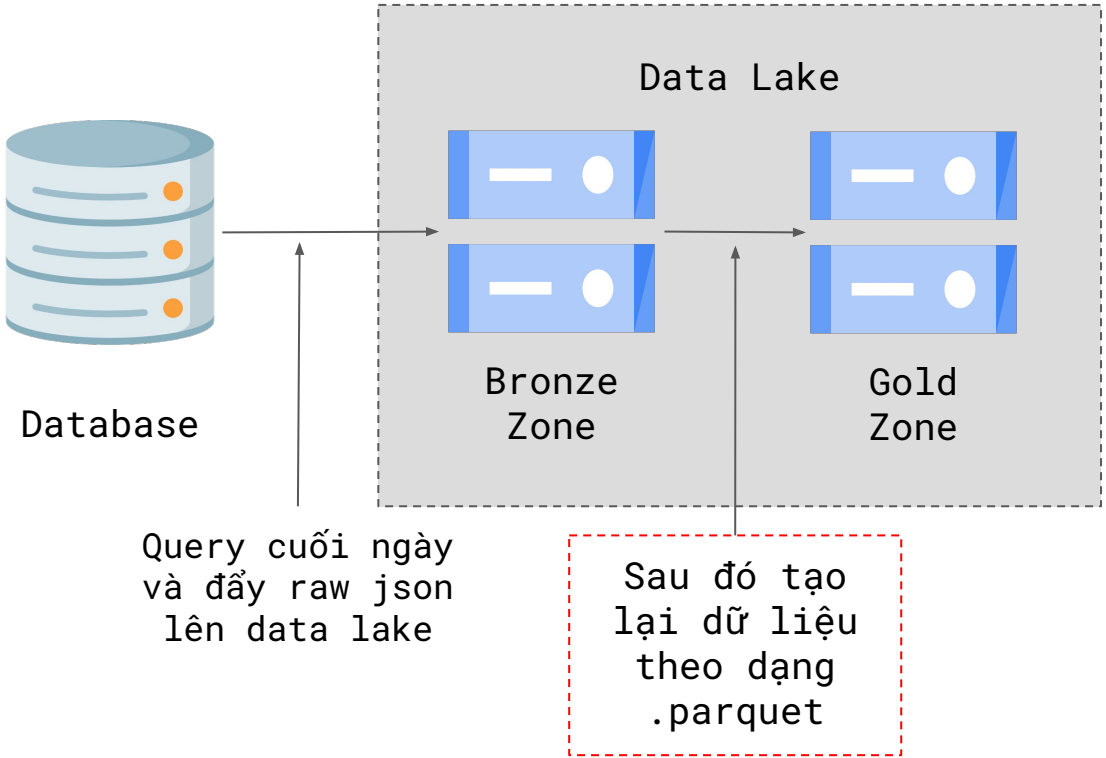
MapReduce và Spark

Introduction to MapReduce and Spark

PROTON X

Hệ thống xử lý dữ liệu

Data Pipeline



PROTON X

Vấn đề

Problems

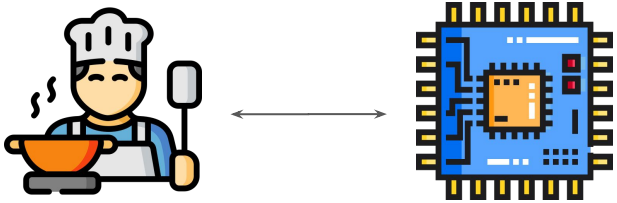
Data cần xử lý nhiều

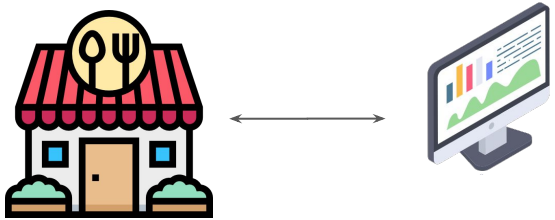
- **Tốc độ tính toán**
- **Khả năng sử dụng bộ nhớ trong (RAM)**

PROTON X

Ví dụ

Example





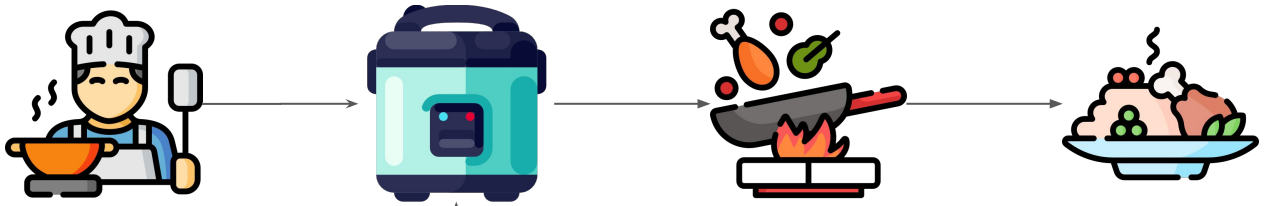
PROTON X

Tăng tốc độ xử lý

Speed up

Lập trình tuyến tính

Single flow Programming



Mất thời gian chờ

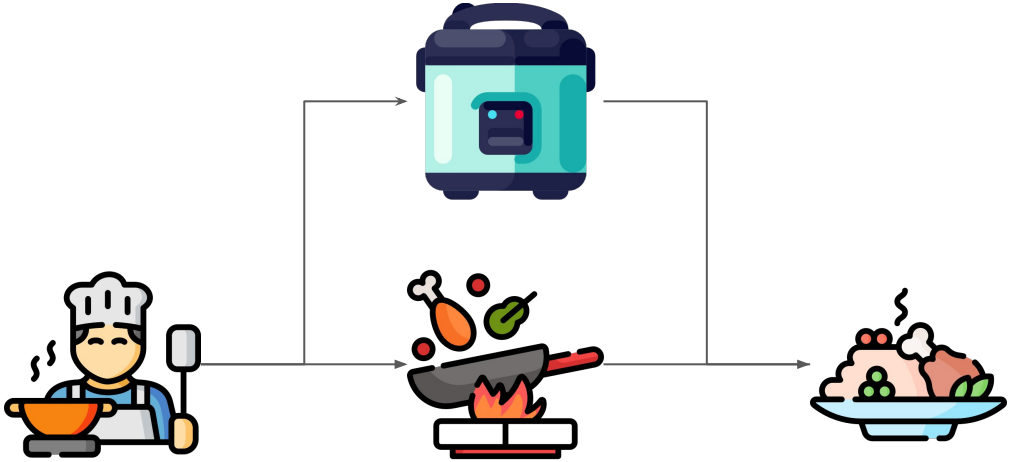
PROTON X

Tăng tốc độ xử lý

Speed up

Lập trình bất đồng bộ

Asynchronous Programming



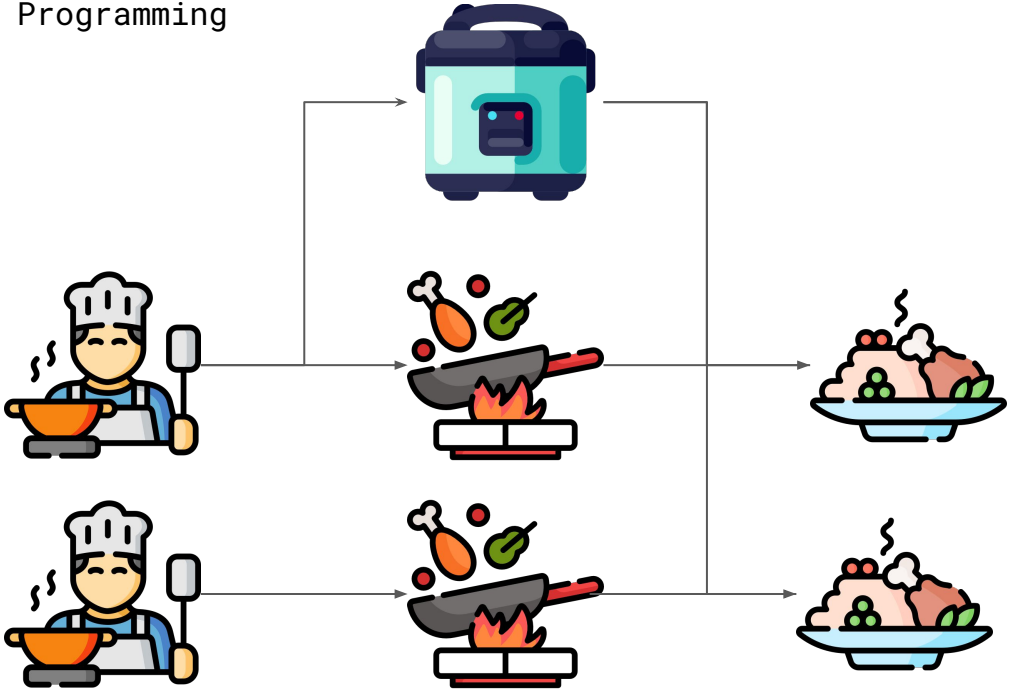
PROTON X

Tăng tốc độ xử lý

Speed up

Lập trình song song

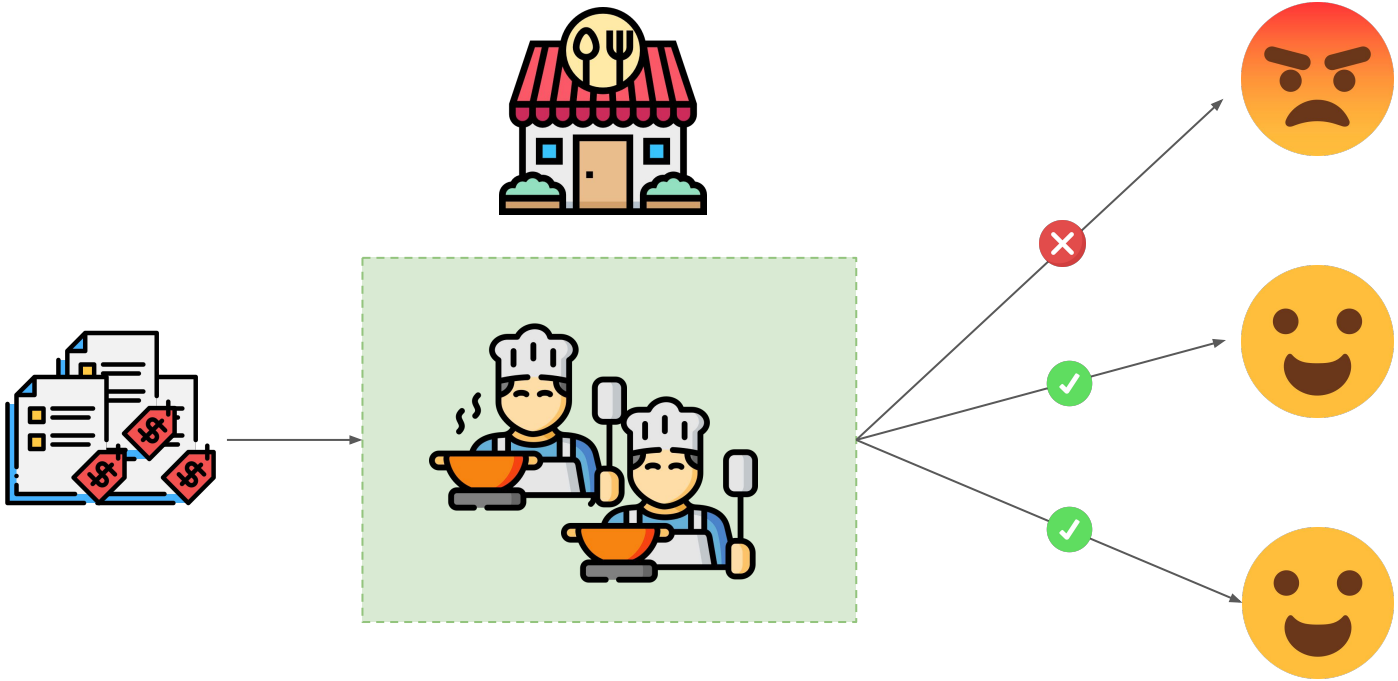
Parallel Programming



PROTON X

Mở rộng?

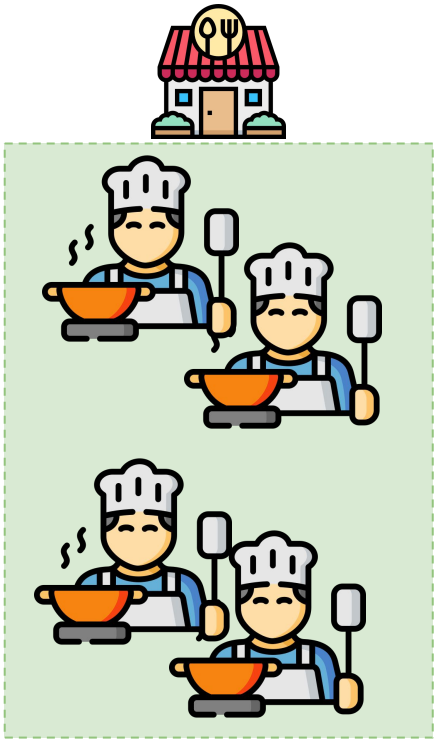
Scaling ?



PROTON X

Mở rộng?

Scaling ?



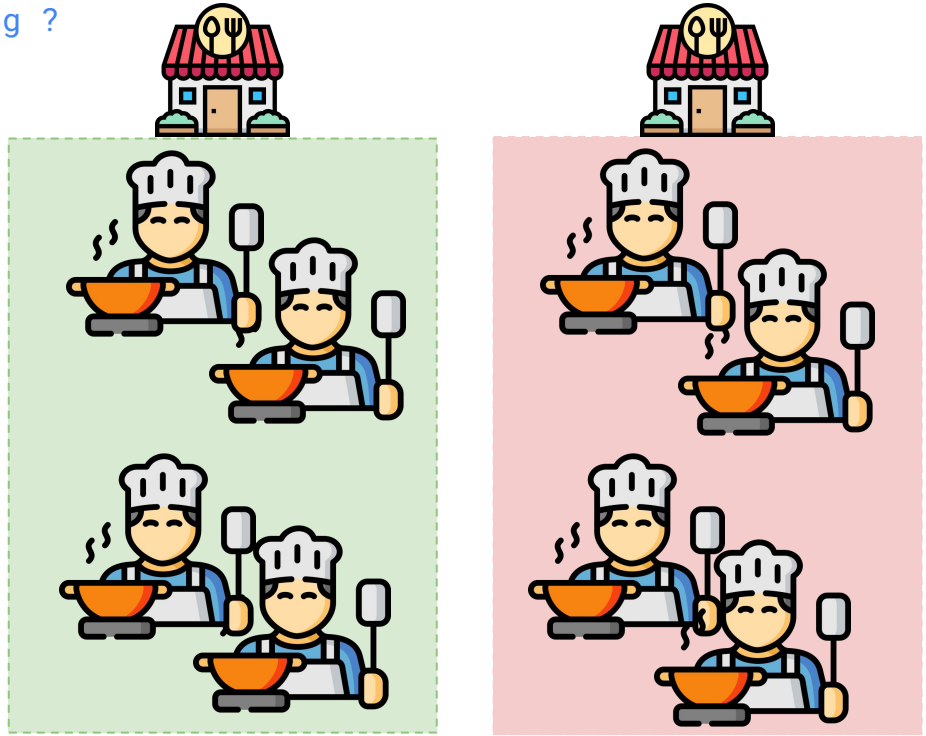
Giới hạn về mặt bằng / phần cứng

Mở rộng theo chiều dọc. Nhiều đầu bếp hơn

PROTON X

Mở rộng?

Scaling ?



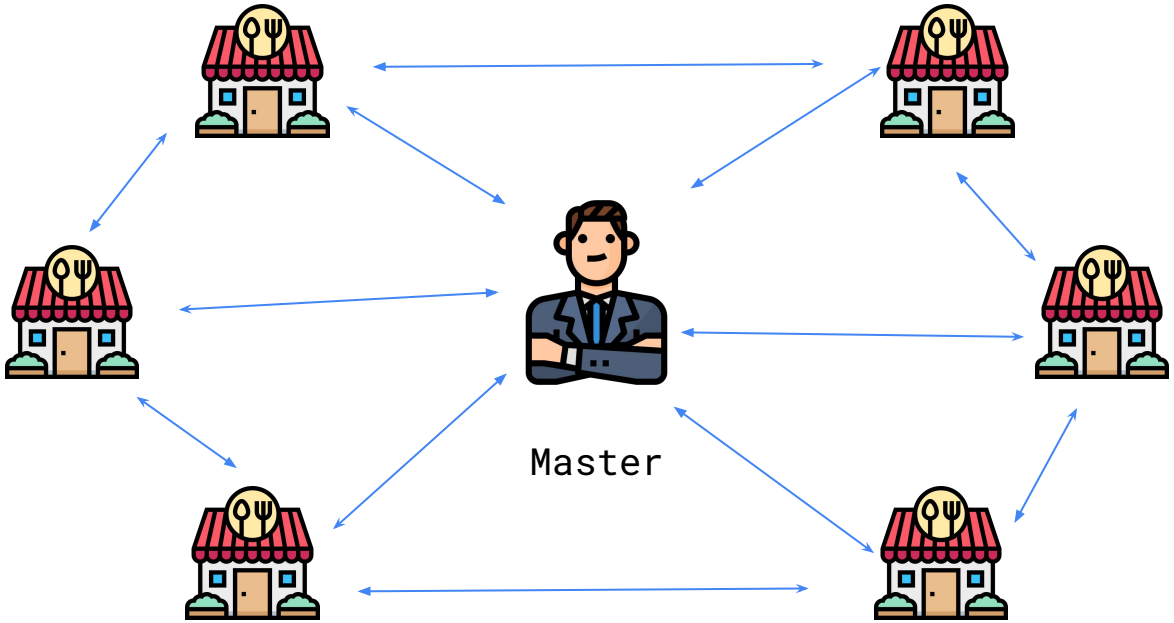
Mở rộng theo chiều dọc. Nhiều đầu bếp hơn

Mở rộng theo chiều ngang. Nhiều chi nhánh hơn

PROTON X

“Cụm” nhà hàng

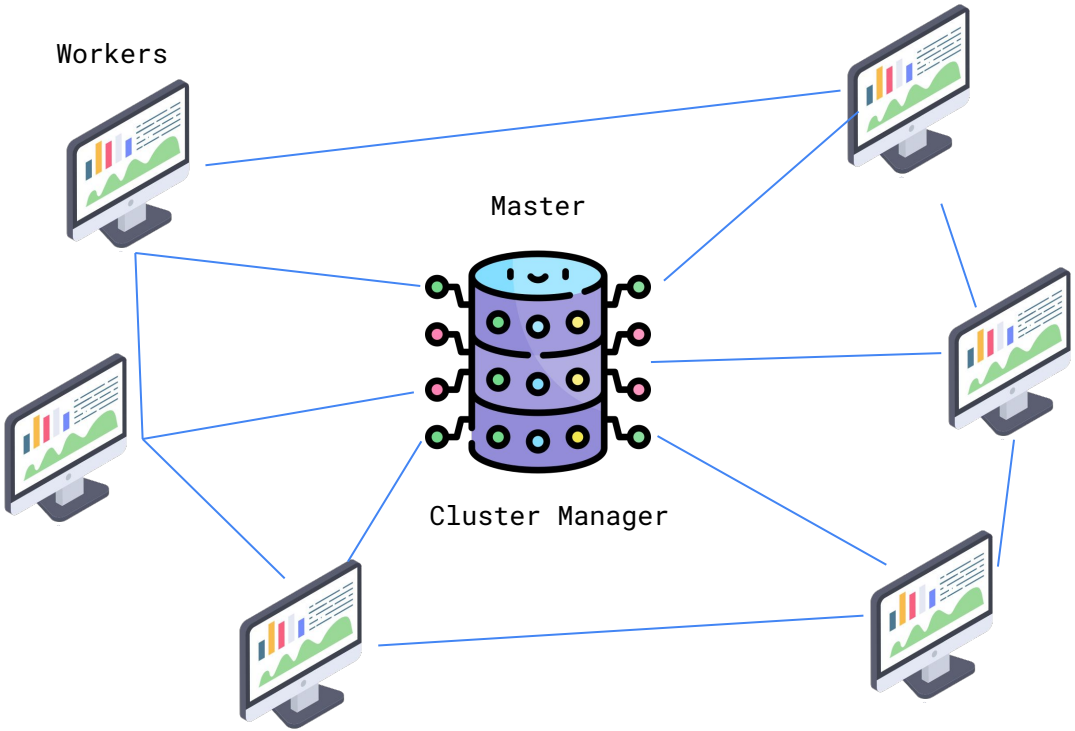
“Cluster” of Restaurant



PROTON X

Distributed Systems

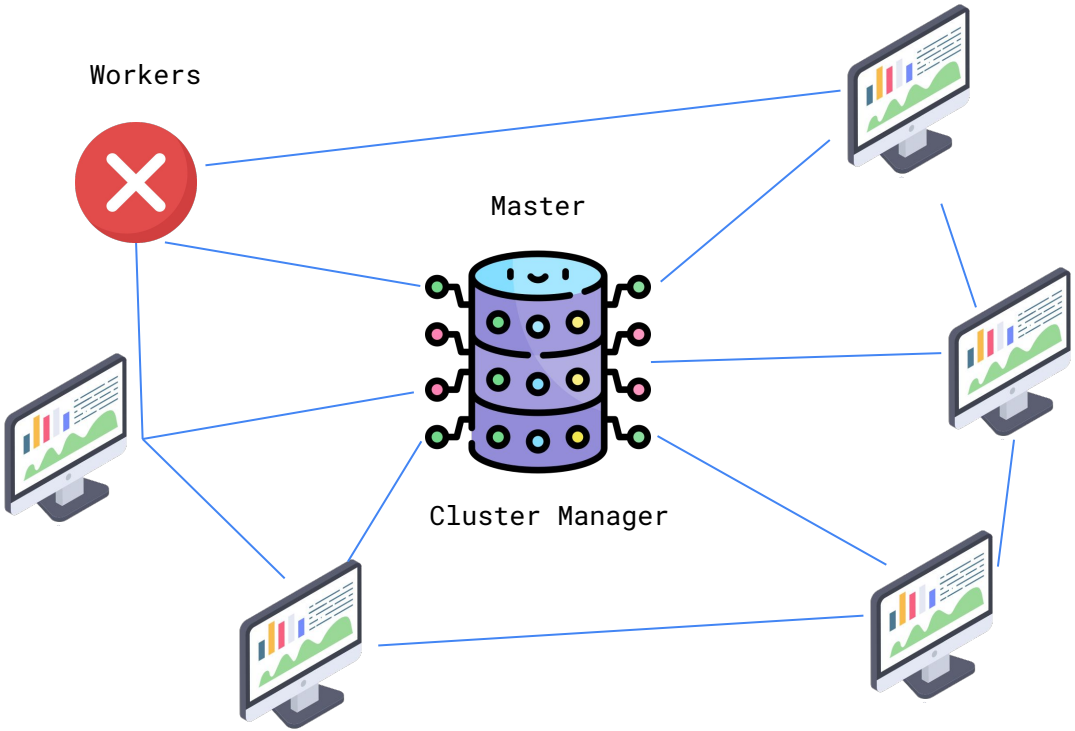
Hệ Thống Phân Tán



PROTON X

Distributed Systems

Hệ Thống Phân Tán



PROTON X

Trình quản lý Cluster

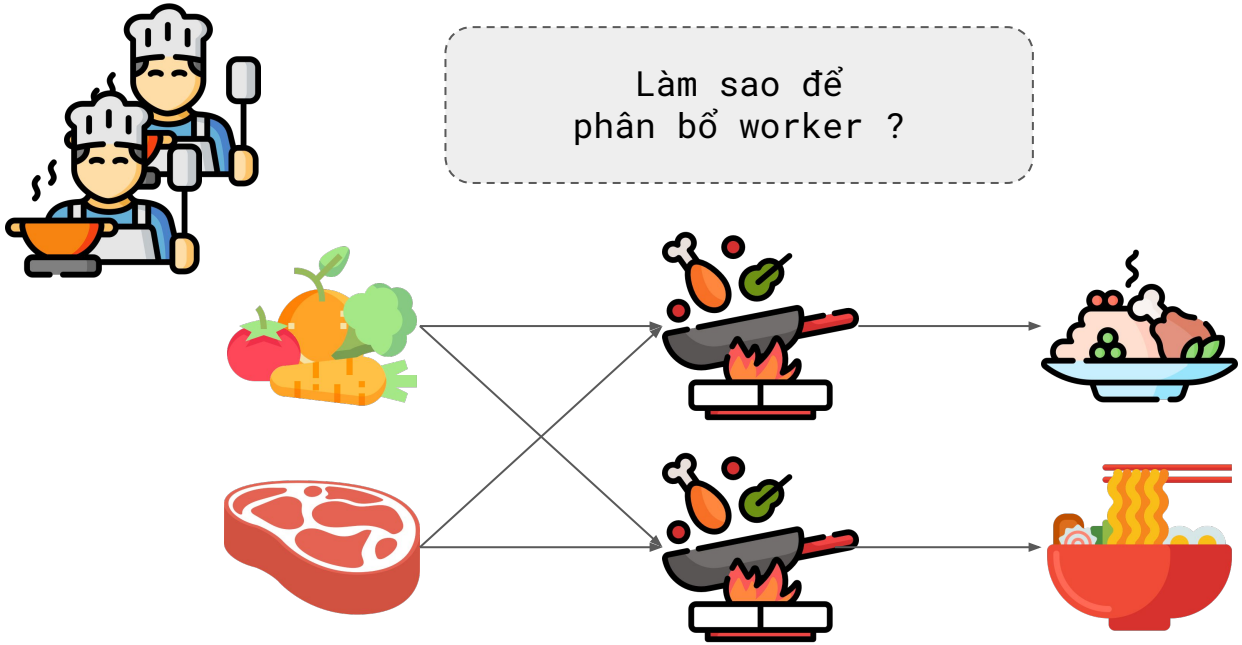
Cluster Manager Software

kubernetes

PROTON X

Thuật toán tính toán phân tán ?

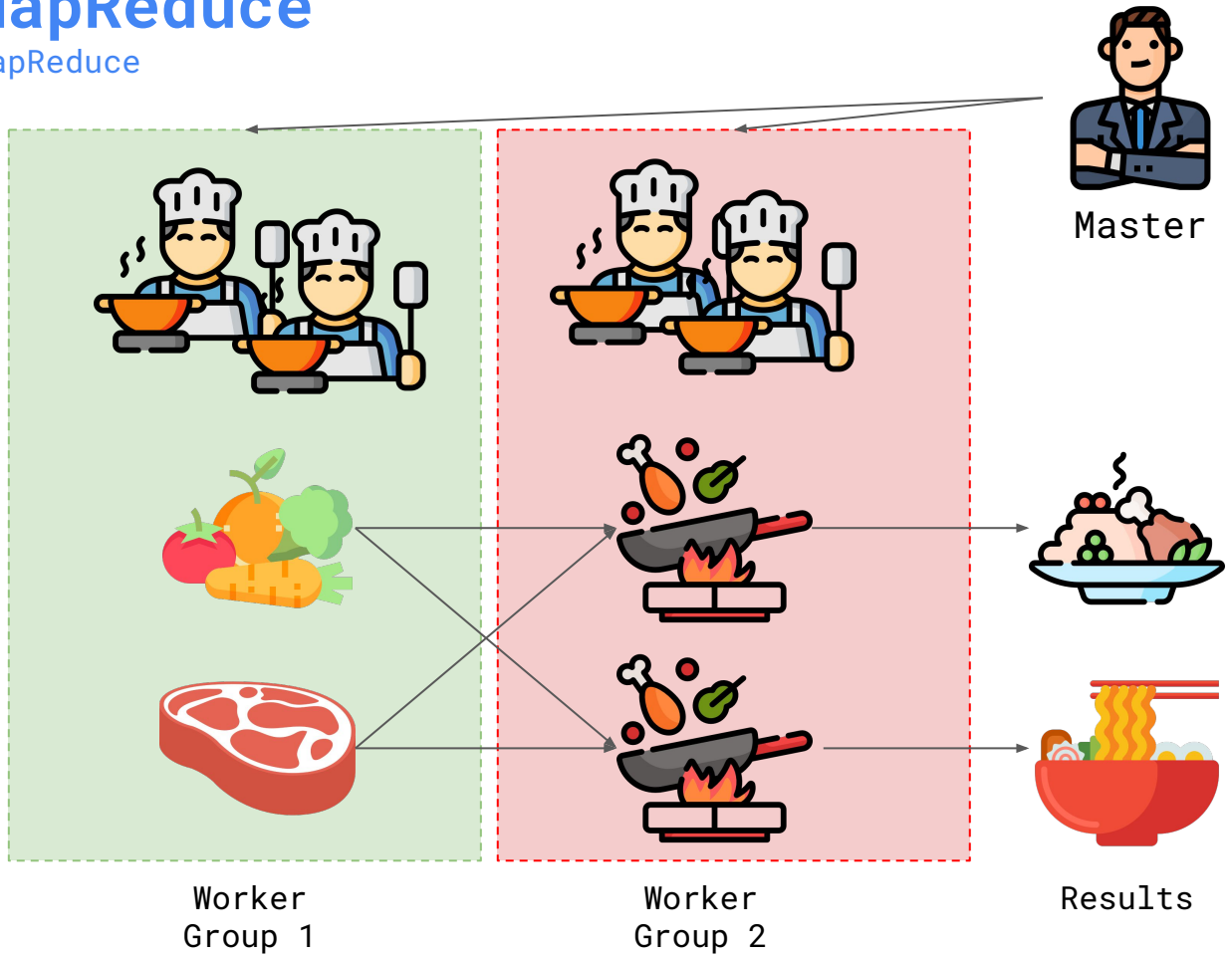
Algorithm for Distributed Computing



PROTON X

MapReduce

MapReduce



PROTON X

Lịch sử MapReduce

History of MapReduce

2004 **MapReduce**

Được tạo ra bởi Jeffery Dean và Sanjay Ghemawat tại Google.

2007 **Hadoop**

Được tạo ra bởi Doug Cutting tại Yahoo, bao gồm: Hadoop Distributed File System (*HDFS*), và *MapReduce* framework

2009 - 2014 **Apache Spark**

Cú pháp đơn giản hơn, xử lý nhanh hơn. Apache Spark chính thức soán ngôi MapReduce trong việc xử lý phân tán.

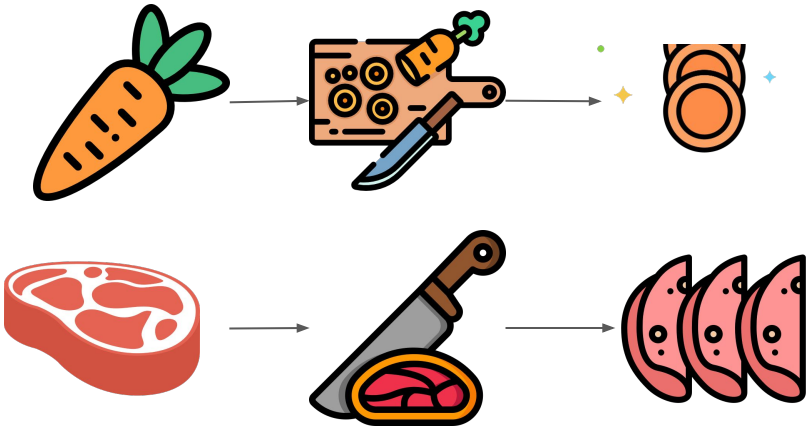
2014 - 2016 **Google Dataflow - Apache Beam**

Google triển khai dự án Dataflow và publish SDK, nền tảng cho Apache Beam. Apache Beam có thể xử lý dữ liệu theo batch và streaming với cùng một cấu trúc câu lệnh.

PROTON X

Khái niệm MapReduce

MapReduce Concept

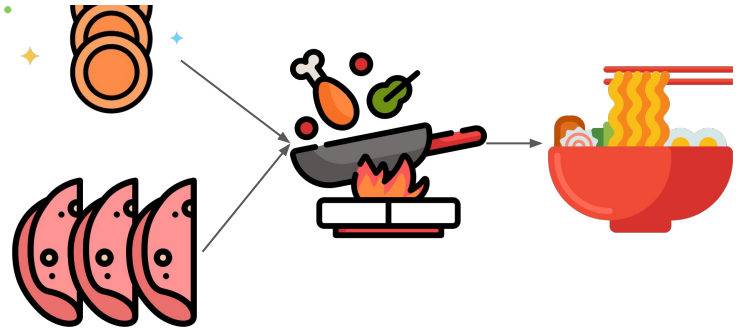


Mỗi nguyên liệu đầu vào sẽ được sơ chế và ra thành phẩm tương ứng

Map

Reduce

Các nguyên liệu sẽ được tổng hợp lại thành món ăn



PROTON X

MapReduce và Python

MapReduce and Python

```
def plus_one(n):
      return n + 1

numbers = [1,2,3,4]
results = map(plus_one, numbers)
list(results)
# > [2, 3, 4, 5]
```

Lấy n điểm dữ liệu đầu vào và output ra n điểm tương ứng

Map

Reduce

Lấy n điểm dữ liệu đầu vào và output ra 1 điểm

```
from functools import reduce
def addition(a,b):
      return a + b

numbers = [1,2,3,4]
results = reduce(addition, numbers)
results
# > 10
```

PROTON X

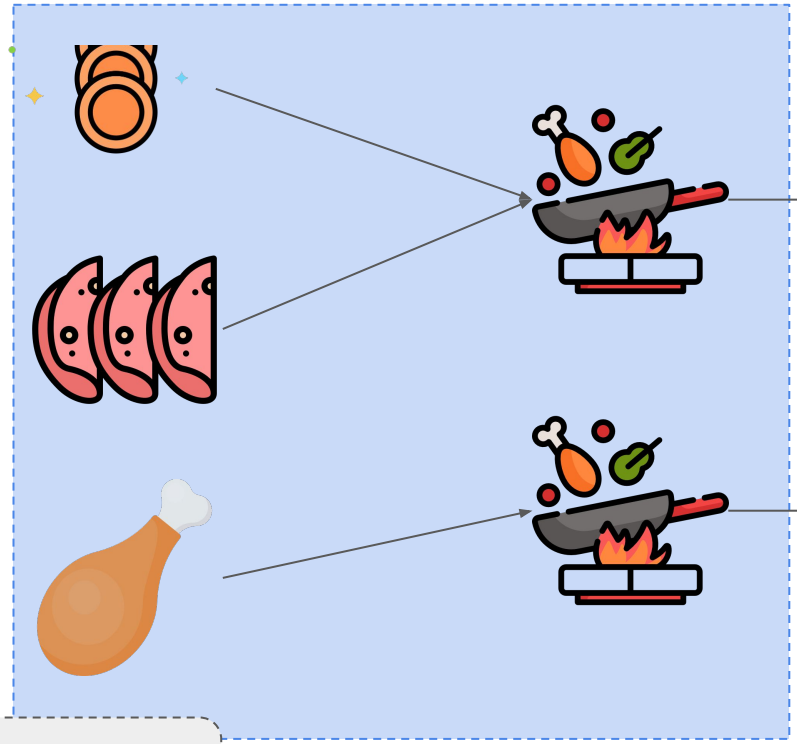
Trộn và Sắp Xếp

Shuffle and Sort

Nguyên liệu mì

Nguyên liệu mì

Nguyên liệu cà ri



Các nguyên liệu làm mỗi món sẽ được đưa đến bếp làm món đó

Nhóm các nguyên liệu vào 1 nhóm

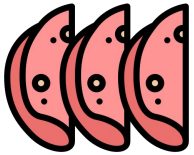
PROTON X

Cặp Key và Value

Pair Key Value

Nguyên liệu mì

,



Key

Value

PROTON X

Key Value trong Python

Key Value In Python

```
def plus_one_with_key(n):
      key="odd"
      result = n + 1
      if result == 0:
            key="zero"
      if abs(result) % 2 == 0:
            key="even"
      return (key,n + 1)

numbers = [1,2,3,4]
results = map(plus_one, numbers)
list(results)
# > [("even",2),("odd",3),
      ("even", 4), ("odd",5)]
```

Map

Reduce

```
from functools import reduce
from itertools import groupby

def addition(tupple_a,tupple_b):
      key, value_a = tupple_a
      key, value_b = tupple_b
      return key,value_a + value_b

numbers = [("even",2),("odd",3),
("even", 4), ("odd",5)]
numbers = sorted(numbers)
results = []
for key, group in groupby(numbers,
                  lambda x: x[0]):
      results.append(
                  reduce(addition,
                  list(group)))
results
# [('even', 6), ('odd', 8)]
```

PROTON X

Trộn và Sắp Xếp

Shuffle and Sort

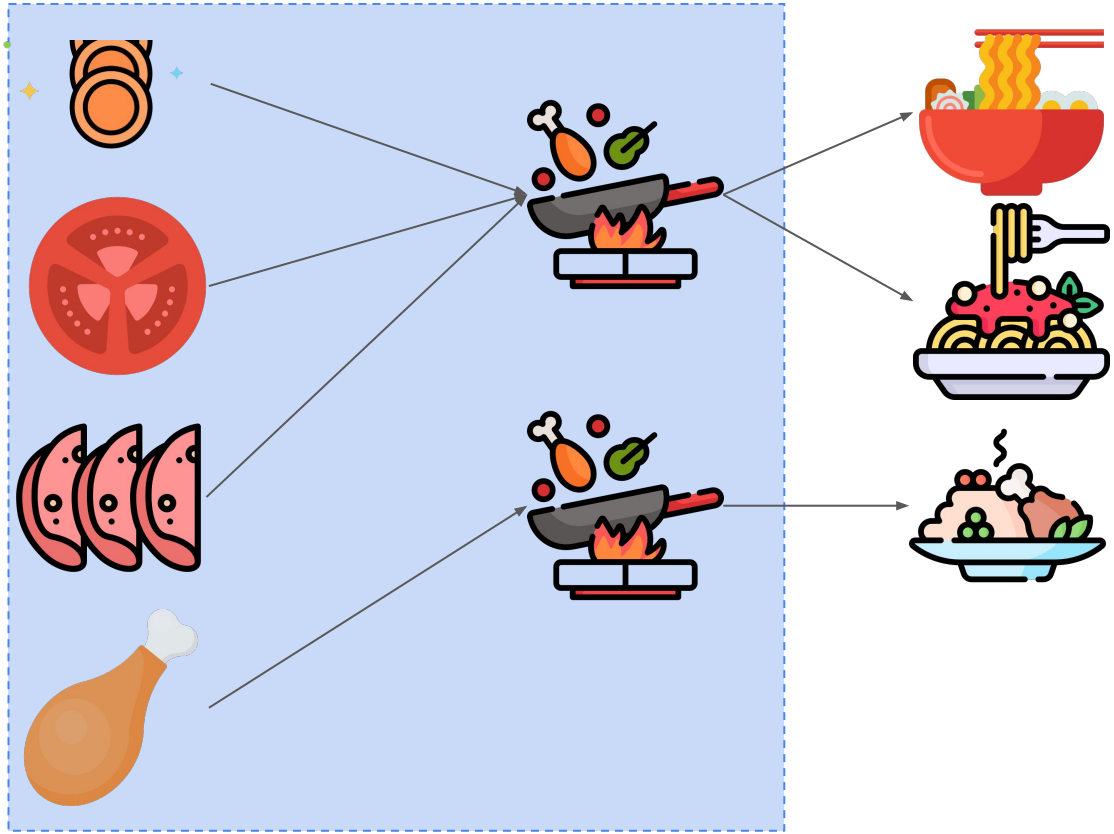
Nếu mỗi bếp được làm 2 món khác nhau ?

Nguyên liệu mì nước

Nguyên liệu mì ý

Nguyên liệu mì nước

Nguyên liệu cà ri



PROTON X

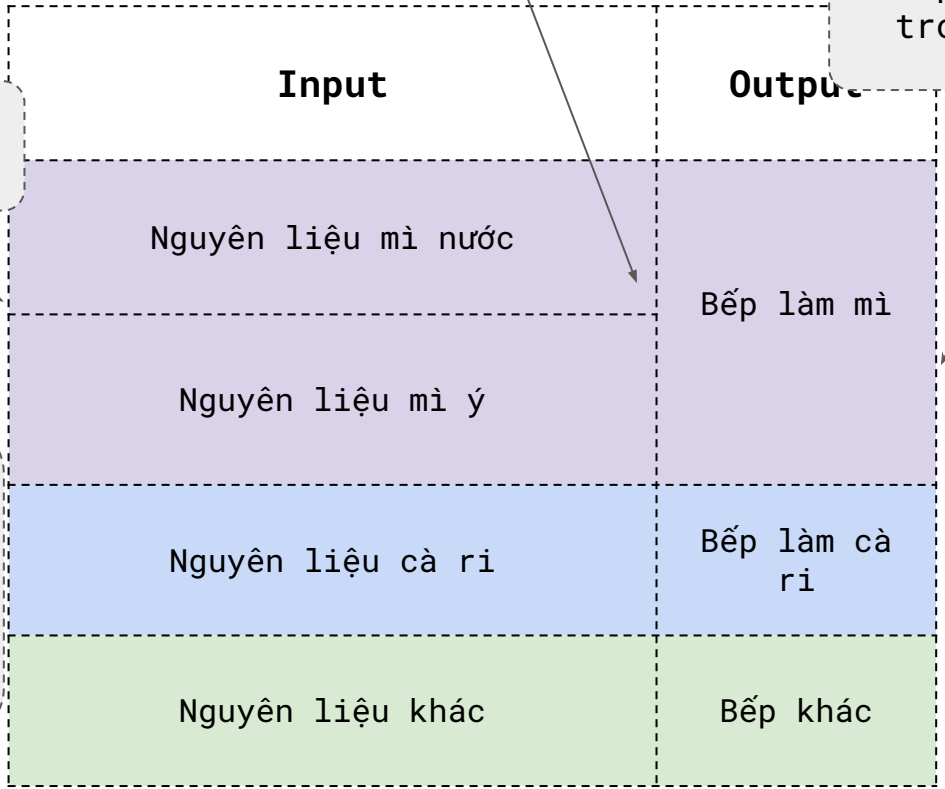
Hàm băm

Hash Function

Xử lý input đó

Output là một giá trị trong tập M < N cho trước

Hàm băm nhận một input có trong tập hợp có N giá trị



Trong ví dụ ở đây hàm băm tìm nguyên liệu làm món mì / cà ri để output ra bếp + mì hoặc cà ri

PROTON X

Hàm băm

Hash Function

Input	Output
1	1
2	2
3	0
4	1

f(x) = x mod 3

Giả sử chúng ta chỉ có 3 worker. Hàm băm sẽ trả về giá trị trong tập hợp gồm 3 phần tử (1 , 2 , 0)

PROTON X

Hàm băm

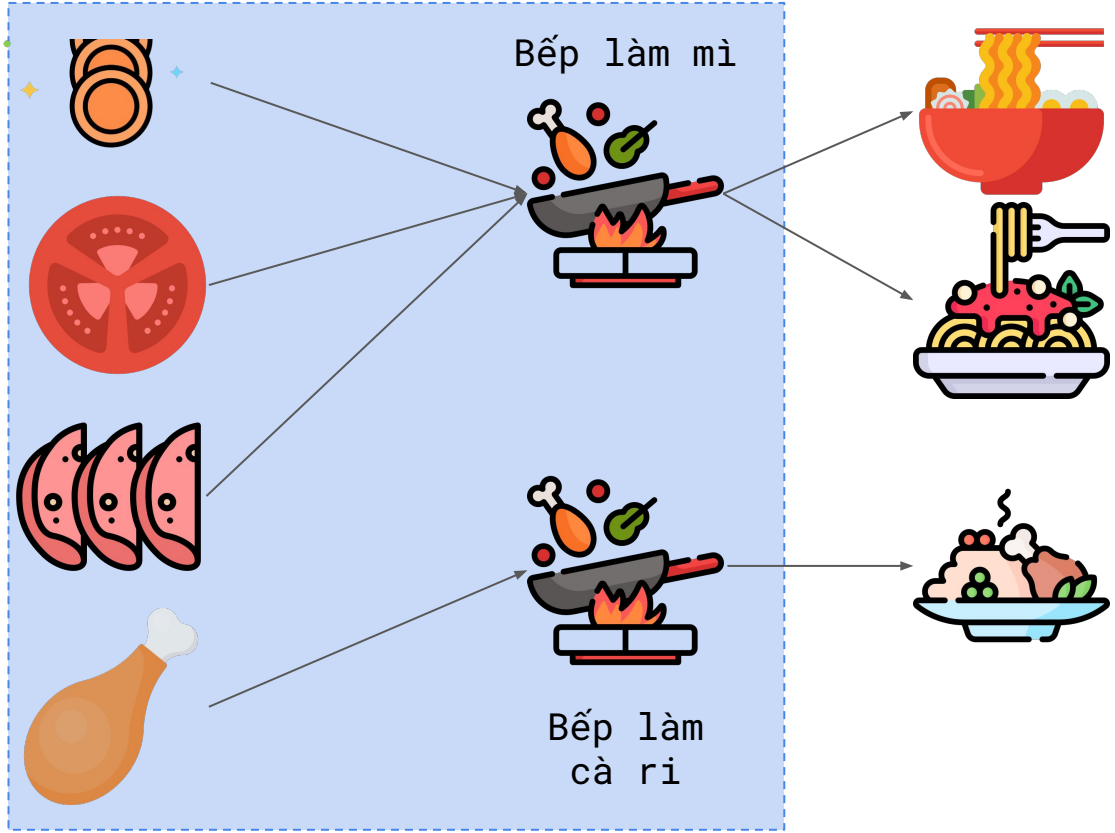
Hash Function

Nguyên liệu mì nước

Nguyên liệu mì ý

Nguyên liệu mì nước

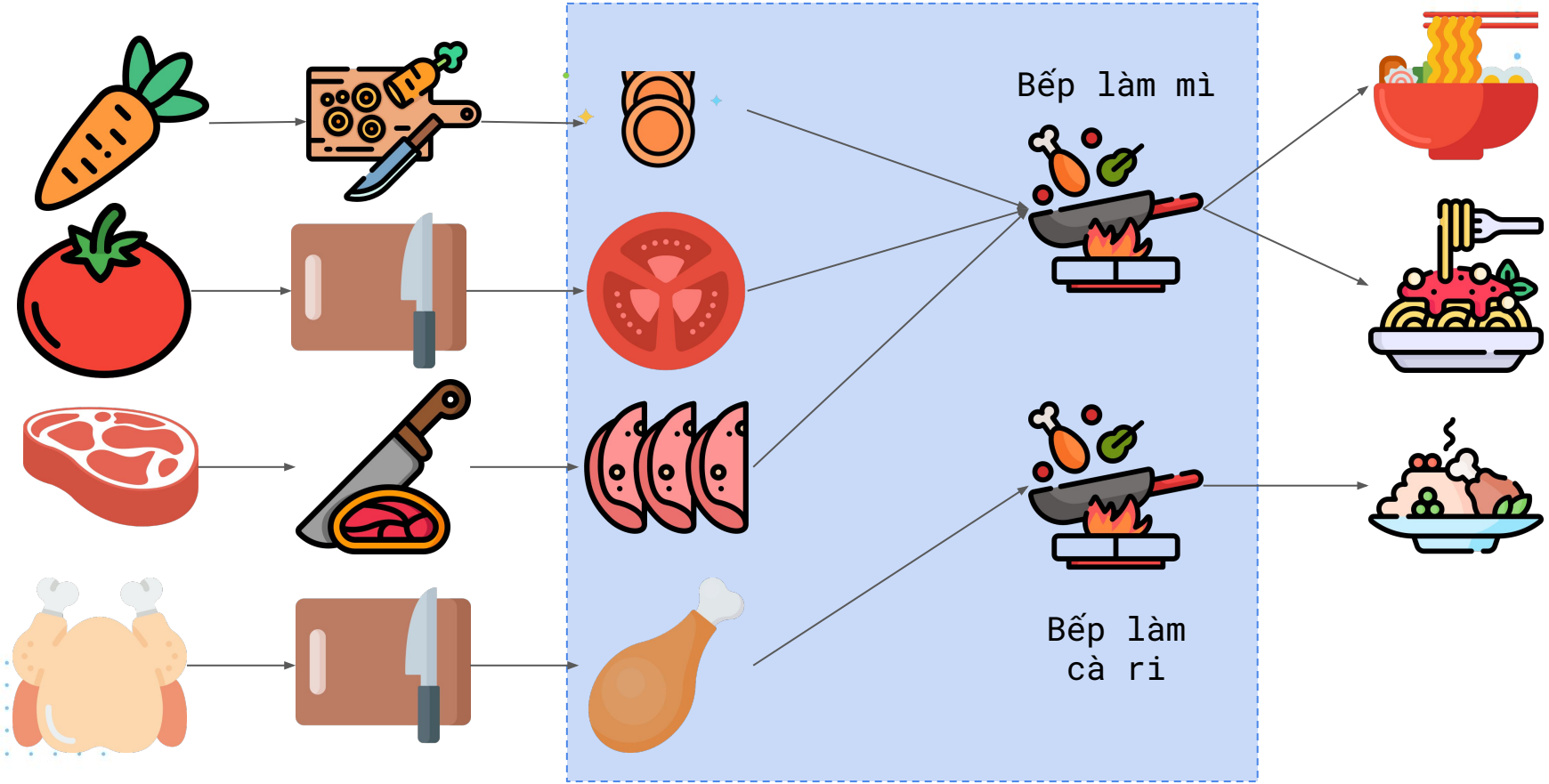
Nguyên liệu cà ri



PROTON X

Tổng thể MapReduce

MapReduce



PROTON X

MapReduce và BigQuery

MapReduce and BigQuery

```
SELECT event_type, e.float_value + 1 as plus_one
FROM `adventure_mmo_game.event_log`,
      UNNEST(event_attribute) as e
WHERE event_type = 'purchase'
AND e.key = 'revenue'
LIMIT 1000;
```

Lấy n điểm dữ liệu đầu vào và output ra n điểm tương ứng

Map

Reduce

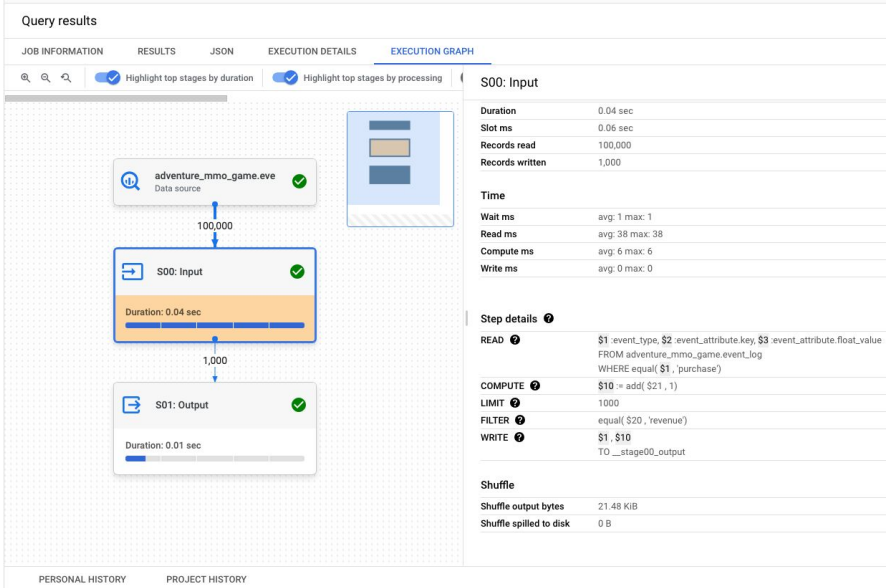
Lấy n điểm dữ liệu đầu vào và output ra 1 điểm

```
SELECT event_type, SUM(e.float_value) as
sum
FROM `adventure_mmo_game.event_log`,
      UNNEST(event_attribute) as e
WHERE event_type = 'purchase'
AND e.key = 'revenue'
GROUP BY event_type;
```

PROTON X

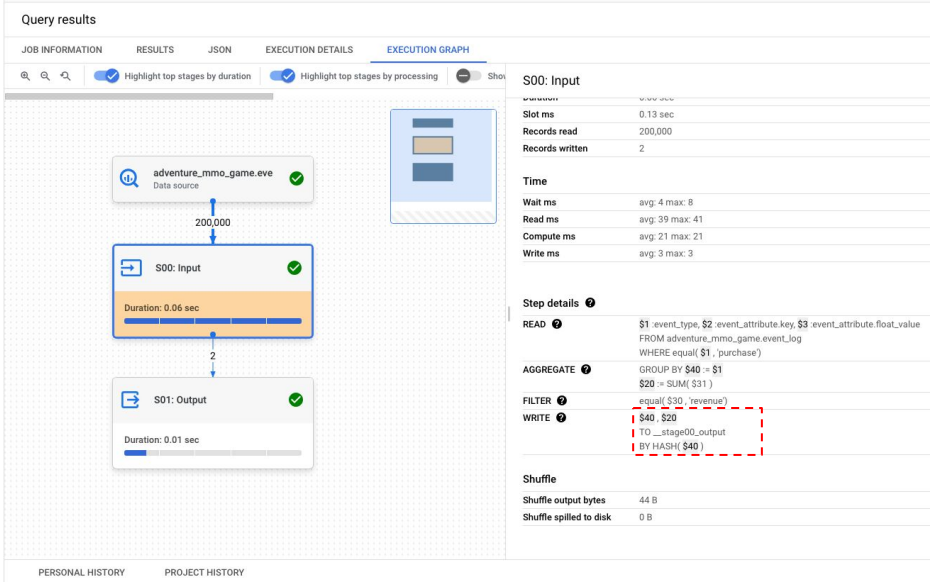
MapReduce và BigQuery

MapReduce and BigQuery



Map

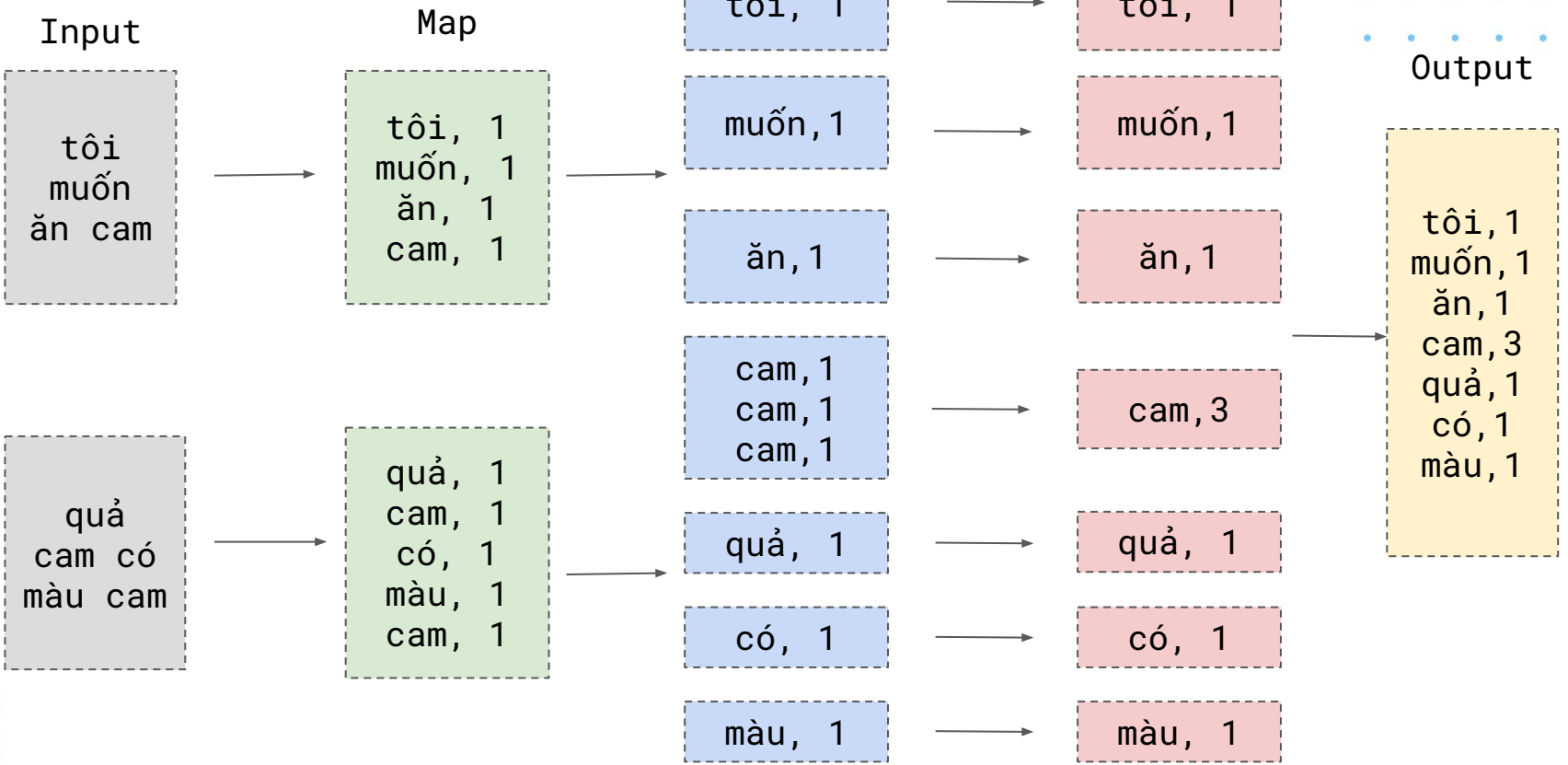
Reduce



PROTON X

Word Count

Word Count



PROTON X

Các công cụ MapReduce

MapReduce Tools

PROTON X

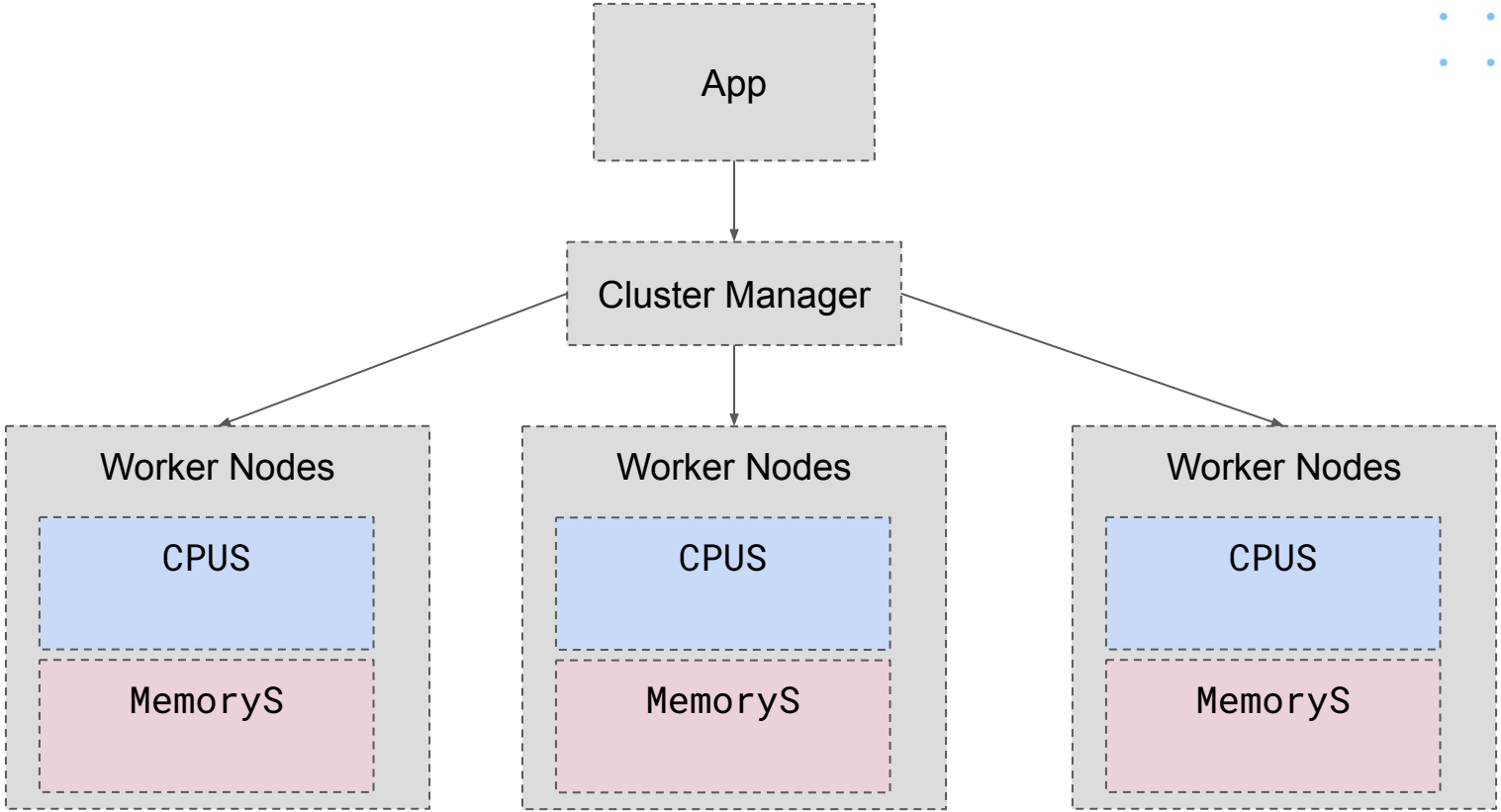
Spark

Spark

PROTON X

Kiến trúc Spark

Spark Architecture



PROTON X

Cài đặt Spark

Installation Spark

Python

```
pip install pyspark
```

Dataproc

https://console.cloud.google.com/dataproc/

Databricks

https://docs.databricks.com/getting-started/community-edition.html

PROTON X

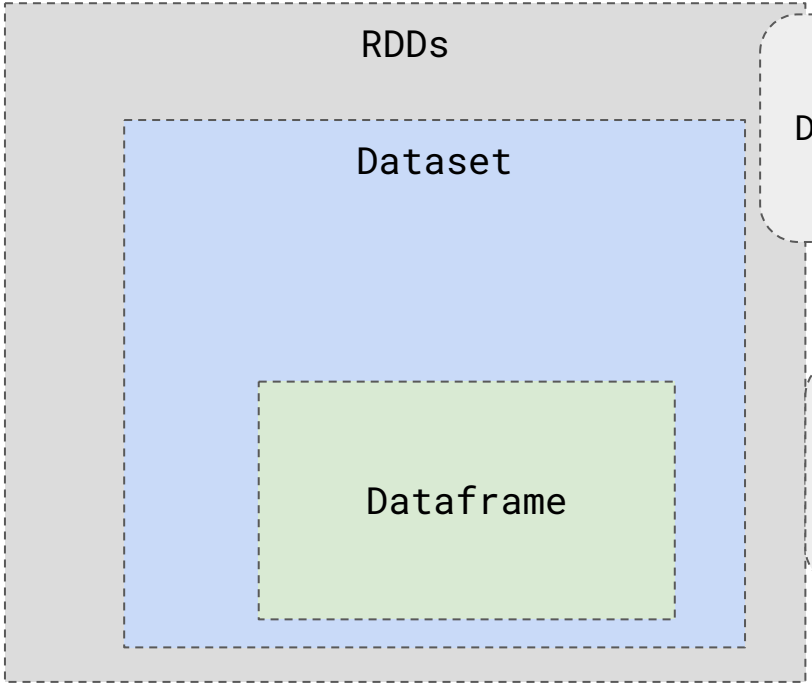
Cấu trúc dữ liệu trong Spark

Spark Data Structure

RDD là nền móng của Spark

Có thể dùng Spark SQL API để xử lý Dataframe

Dataframe là cấu trúc dữ liệu thường dùng và thân thiện với người dùng nhất



Có thể định nghĩa Dataset theo định dạng của một class bất kỳ

Dataframe là Dataset của class Row.

PROTON X

Spark Dataframe

Spark Dataframe

```
# Đọc file csv
df = ( spark.read
      .format("csv")
      .option("header",true)
      .option("sep",",")
      .load("gs://data.csv")
)

# Đọc file parquet
df = spark.read.parquet("gs://data.parquet")
```

PROTON X

Spark Dataframe

Spark Dataframe

Phép toán	Code
Thêm cột	`new_df = df.withColumn("new_col",df.col * 2)`
Xoá cột	`new_df = df.drop("col_name")`
Sửa tên cột	`renamed_df = df.withColumnRenamed("old_col", "new_col")`

PROTON X

Spark và SQL

Spark and SQL

```
df.select(
      F.col("product_id"),
      F.col("review_date"),
      F.col("star_rating"),
      F.col("helpful_votes")
)
.where(
      (F.col("star_rating") >= 4 ) &
      (F.col("helpful_votes") >= 10 )
)
.groupBy(["product_id","review_date"])
.count()
```

```
SELECT product_id,review_date,COUNT(star_rating)
FROM df
WHERE star_rating >=4 and helpful_votes >= 10
GROUP BY product_id
```

PROTON X

Spark và SQL

Spark and SQL

Phép toán	Code
SELECT	df.select(F.col("col")) df.select(F.col("col")).distinct()
WHERE	df.filter(F.col("col") > 5) df.where(F.col("col") > 5)
GROUP BY	df.groupBy("col").count() df.groupBy("col").agg({"col": "func"})
ORDER BY	df.orderBy(F.desc("col")) df.orderBy(F.asc("col"))
JOIN	df1.join(df2, on=F.col("key_col"), how="inner")

https://spark.apache.org/docs/latest/api/python/reference/pyspark.sql/functions.html

PROTON X

UDF (User Defined Function)

Spark SQL

```
from pyspark.sql import functions as F
from pyspark.sql.types import IntegerType

def convert_case(string):
     return string.upper()

convert_case_udf = F.udf(convert_case,returnType=types.StringType())
df.withColumn('review_headline_uppercase',
               convert_case_udf(F.col("review_headline"))
               )
```

Định nghĩa Function

Kiểu dữ liệu function sẽ nhận

PROTON X

Spark SQL

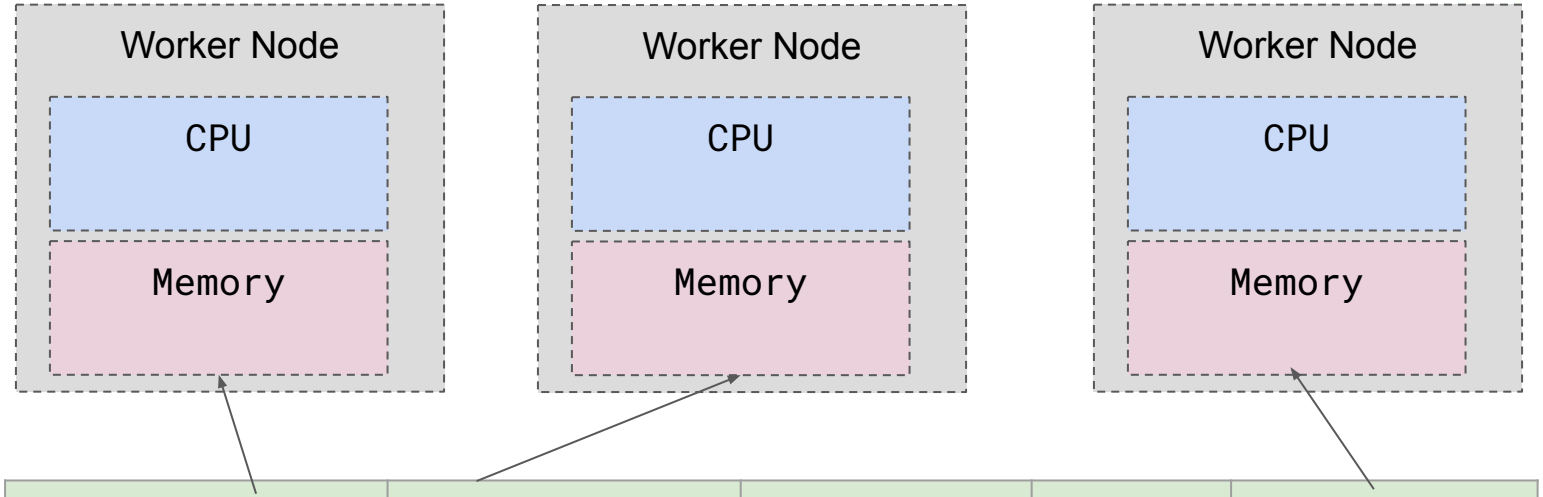
Spark SQL

```
# Tạo temp View để query
df.createOrReplaceTempView("df")

query = """
SELECT product_id,review_date,COUNT(star_rating)
FROM df
WHERE star_rating >=4 and helpful_votes >= 10
GROUP BY product_id
"""

spark.sql(query).show()
```

PROTON X

RDD

Resilient Distributed Datasets



Data source được chia làm nhiều partition và load vào RAM của worker

Data của từng partition có thể có schema / định dạng khác nhau

Lazy load. Chỉ load 1 vài partition vào RAM

PROTON X

RDD

Resilient Distributed Datasets

```
# Lấy rdd từ dataframe
df_rdd = df.rdd

# Tạo spark context
sc = spark.sparkContext()

# Các cách đọc lên rdd
my_rdd_1 = sc.textFile("gs://data.txt")
my_rdd_2 = sc.wholeTextFiles(gs://folder/)
my_rdd_3 = sc.binaryFiles("gs://data.gz",
                          use_unicode=False,
                          compression="gzip",
                          bufferSize=65536)
```

PROTON X

Các hàm Map trong RDD

Map Function in RDD

Phép toán	Tác dụng
map	Thực hiện phép toán trên từng rdd
mapPartition	Thực hiện phép toán trên toàn bộ partition. 1 partition gồm nhiều rdd
flatMap	Thực hiện phép toán cho từng rdd và output ra một list với mỗi rdd đó
flatMapValues	Thực hiện phép toán cho từng key,value rdd. Key sẽ được giữ nguyên trong quá trình biến đổi

PROTON X

Các hàm Reduce trong RDD

Reduce Function in RDD

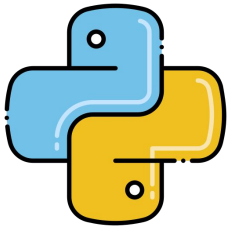
Phép toán	Tác dụng
reduce	Thực hiện các phép tổng hợp theo value cho một tập hợp các rdd
reduceByKey	Thực hiện các phép tổng hợp theo value cho một tập hợp các key,value rdd có cùng 1 key

PROTON X

Spark Job

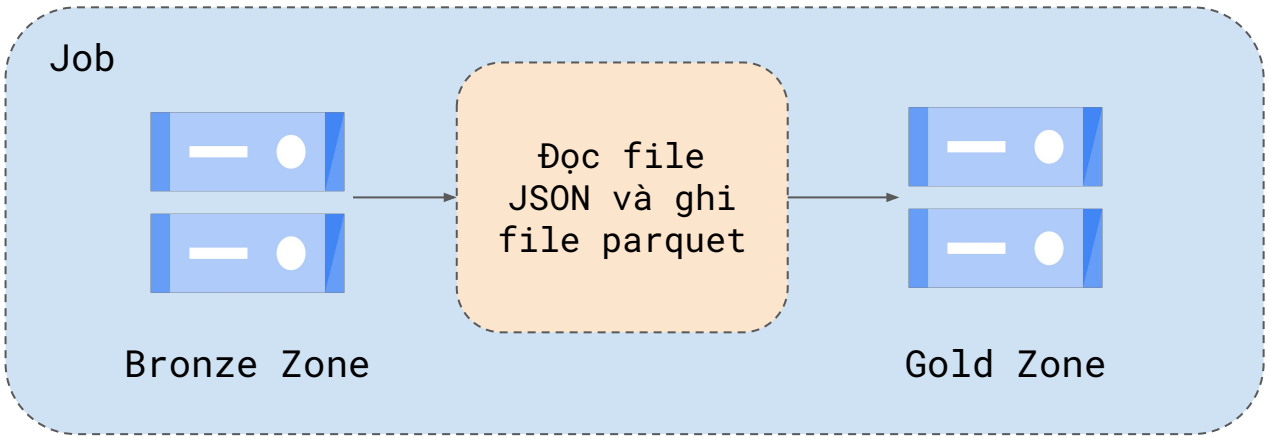
Spark Job

Script



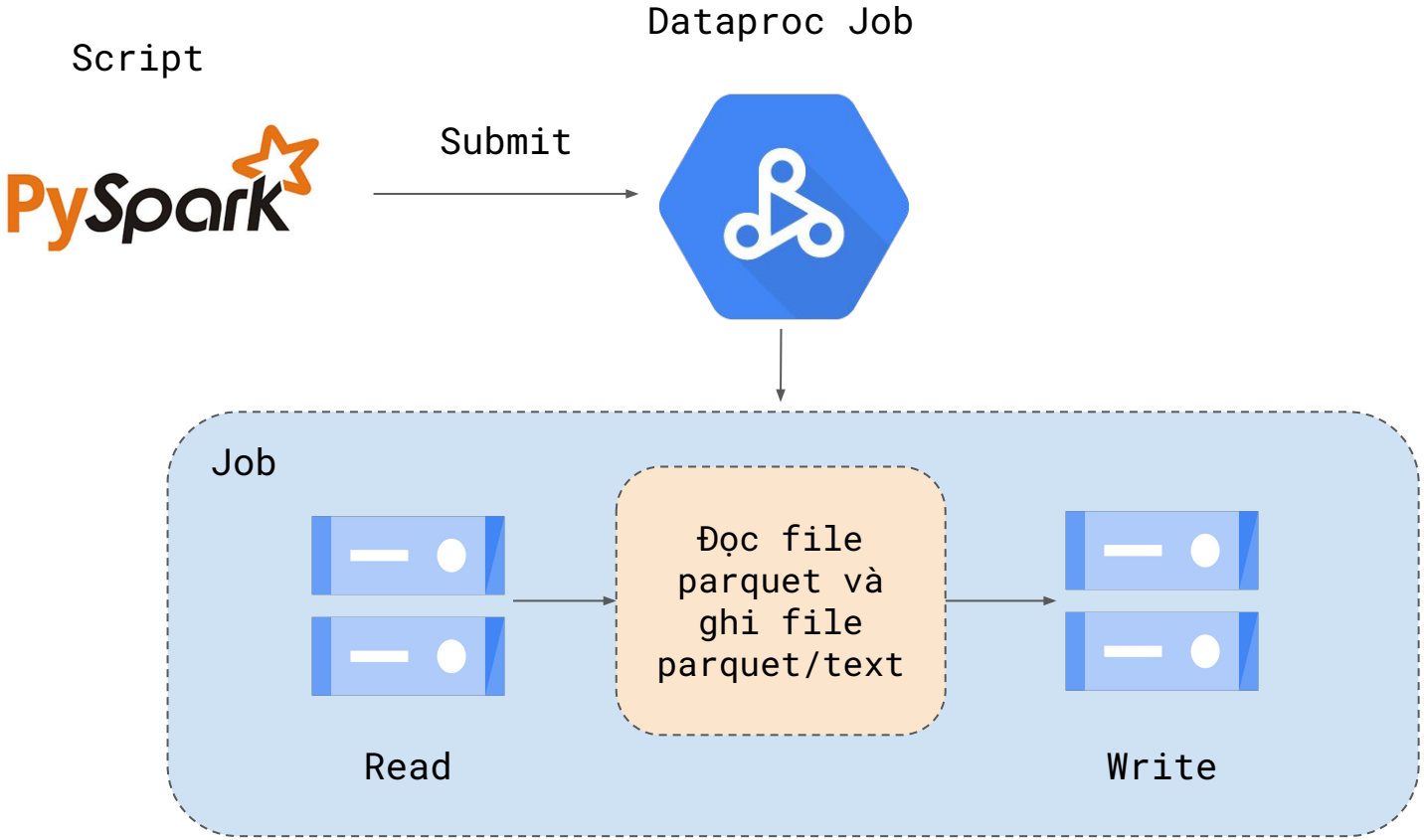
Submit

Cloud Run Job



Spark Job

Spark Job



PROTON X

Submit Spark Job

Submit Spark Job

```
CLUSTER_NAME=<CLUSTER_NAME>
REGION=<REGION>
INPUT=<INPUT_PATH>
OUTPUT=<OUTPUT_PATH>

gcloud dataproc jobs submit pyspark \
    --cluster=${CLUSTER_NAME} \
    --region=${REGION} \
    spark_job.py \
    -- \
        --input=${INPUT} \
        --output=${OUTPUT}
```