

Hadoop MapReduce Tham gia & truy cập bằng ví dụ

Bởi: Prafulla Ranadive

🕒 cập nhật 13 Tháng Sáu, 2024



Tham gia Mapreduce là gì?

Mapreduce Tham gia operation được sử dụng để kết hợp hai tập dữ liệu lớn.

Tuy nhiên, quá trình này liên quan đến việc viết nhiều mã để thực hiện phép nối thực tế. Việc nối hai tập dữ liệu bắt đầu bằng cách so sánh kích thước của từng tập dữ liệu. Nếu một tập dữ liệu nhỏ hơn so với tập dữ liệu khác thì tập dữ liệu nhỏ hơn sẽ được phân phối tới mọi nút dữ liệu trong cluster.

Sau khi phân phối liên kết trong MapReduce, Mapper hoặc Giảm tốc sẽ sử dụng tập dữ liệu nhỏ hơn để thực hiện tra cứu các bản ghi trùng khớp từ tập dữ liệu lớn, sau đó kết hợp các bản ghi đó để tạo thành bản ghi đầu ra.

Mục lục:



Các loại tham gia

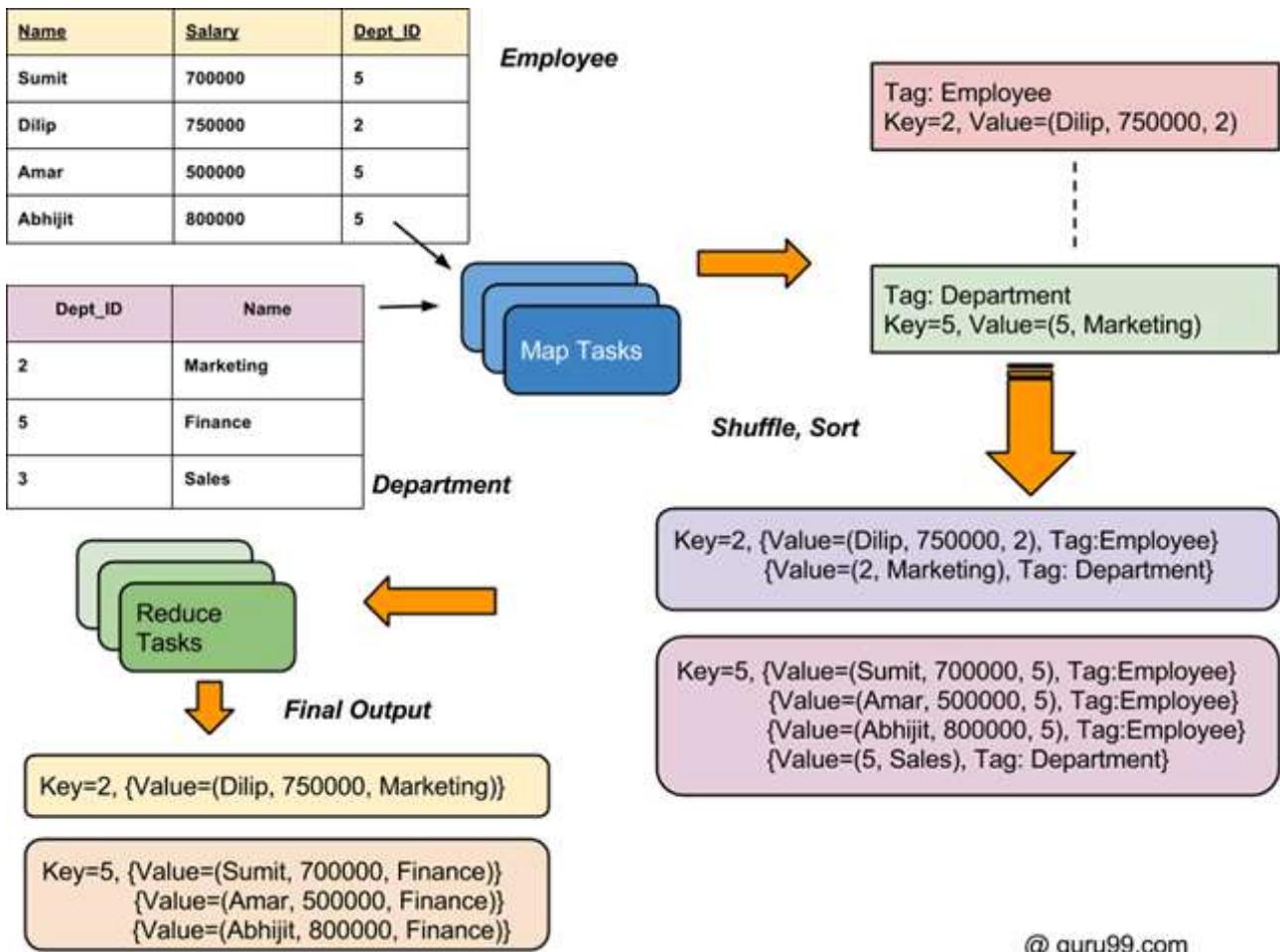
Tùy thuộc vào nơi thực hiện phép nối thực tế, các phép nối trong Hadoop được phân loại thành-

1. Tham gia từ phía bản đồ – Khi phép nối được thực hiện bởi người lập bản đồ, nó được gọi là phép nối phía bản đồ. Trong loại này, phép nối được thực hiện trước khi dữ liệu thực sự được chức năng bản đồ sử dụng. Điều bắt buộc là đầu vào của mỗi bản đồ phải ở dạng phân vùng và theo thứ tự được sắp xếp. Ngoài ra, phải có số lượng phân vùng bằng nhau và phải được sắp xếp theo khóa nối.

2. Tham gia bên giảm – Khi phép nối được thực hiện bởi bộ giảm tốc, nó được gọi là phép nối bên rút gọn. Trong phép nối này không nhất thiết phải có tập dữ liệu ở dạng có cấu trúc (hoặc được phân vùng).

Ở đây, quá trình xử lý phía bản đồ sẽ phát ra khóa nối và các bộ dữ liệu tương ứng của cả hai bảng. Do hiệu ứng của quá trình xử lý này, tất cả các bộ dữ liệu có cùng khóa nối sẽ rơi vào cùng một bộ giảm tốc, sau đó nối các bản ghi có cùng khóa nối.

Luồng quy trình tổng thể của các phép nối trong Hadoop được mô tả trong sơ đồ bên dưới.



@ guru99.com

Các loại kết nối trong Hadoop MapReduce

Cách kết hợp hai bộ dữ liệu: Ví dụ về MapReduce

Có hai Bộ dữ liệu trong hai Tập khác nhau (hiển thị bên dưới). Khóa Dept_ID là chung trong cả hai tập. Mục đích là sử dụng MapReduce Join để kết hợp các tập này

Dept_ID	Dept_Name
A11	Finance
B12	HR
C13	Manufacturing

Tập 1

Dept_ID	Total_Employee
A11	50
B12	100
C13	250

Tập 2

Đầu vào: Tập dữ liệu đầu vào là file txt, **DeptName.txt & DepStrength.txt**

[Tải xuống tập tin đầu vào từ đây](#)

Đảm bảo bạn đã cài đặt Hadoop. Trước khi bạn bắt đầu với quy trình thực tế của ví dụ MapReduce Join, hãy thay đổi người dùng thành 'hduser' (id được sử dụng trong khi cấu hình Hadoop, bạn có thể chuyển sang userid được sử dụng trong cấu hình Hadoop của mình).

```
su - hduser_
```

```
guru99@guru99-VirtualBox:~$ su - hduser_
Password:
hduser_@guru99-VirtualBox:~$
```

Bước 1) Sao chép tệp zip vào vị trí bạn chọn

```
hduser_@guru99-VirtualBox:~$ cp /home/guru99/Downloads/MapReduceJoin.tar.gz /home/hduser_/
hduser_@guru99-VirtualBox:~$ ls /home/hduser_
apache-flume-1.4.0-bin  guava-17.0.jar  MapReduceTutorial
examples.desktop        hdfs              pig_1399372687264.log
flumeTutorial           inputMapReduce    protobuf-java-2.4.1.jar
guava-10.0.1.jar        MapReduceJoin.tar.gz
```

Bước 2) Giải nén tệp Zip

```
sudo tar -xvf MapReduceJoin.tar.gz
```

```
hduser_@guru99-VirtualBox:~$ sudo tar -xvf MapReduceJoin.tar.gz
[sudo] password for hduser_:
MapReduceJoin/
MapReduceJoin/TextPair.java
MapReduceJoin/MapReduceJoin.jar
MapReduceJoin/JoinReducer.java~
MapReduceJoin/Manifest.txt
MapReduceJoin/DeptEmpStrengthMapper.java~
MapReduceJoin/JoinReducer.java
MapReduceJoin/TextPair.java~
MapReduceJoin/DeptNameMapper.java~
MapReduceJoin/JoinDriver.java
MapReduceJoin/Manifest.txt~
MapReduceJoin/DeptNameMapper.java
MapReduceJoin/DeptEmpStrength.txt
MapReduceJoin/JoinDriver.java~
MapReduceJoin/A.txt~
MapReduceJoin/B.txt~
MapReduceJoin/MapReduceJoin/
MapReduceJoin/MapReduceJoin/TextPair$FirstComparator.class
MapReduceJoin/MapReduceJoin/DeptNameMapper.class
MapReduceJoin/MapReduceJoin/JoinDriver$KeyPartitioner.class
MapReduceJoin/MapReduceJoin/TextPair.class
MapReduceJoin/MapReduceJoin/JoinDriver.class
MapReduceJoin/MapReduceJoin/TextPair$Comparator.class
MapReduceJoin/MapReduceJoin/JoinReducer.class
MapReduceJoin/MapReduceJoin/DeptEmpStrengthMapper.class
MapReduceJoin/DeptEmpStrengthMapper.java
MapReduceJoin/DeptName.txt
MapReduceJoin/DeptStrength.txt
hduser_@guru99-VirtualBox:~$
```

Bước 3) Chuyển đến thư mục MapReduceJoin/

```
cd MapReduceJoin/
```

```
hduser_@guru99-VirtualBox:~$ cd MapReduceJoin
hduser_@guru99-VirtualBox:~/MapReduceJoin$
```

Bước 4) Bắt đầu Hadoop

```
$HADOOP_HOME/sbin/start-dfs.sh
```

```
$HADOOP_HOME/sbin/start-yarn.sh
```

```
hduser_@guru99-VirtualBox:~/MapReduceJoin$ SHADOOP_HOME/sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/guru99/Downloads/hadoop/logs/hadoop-hduser_-namenode-guru99-VirtualBox.out
localhost: starting datanode, logging to /home/guru99/Downloads/hadoop/logs/hadoop-hduser_-datanode-guru99-VirtualBox.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/guru99/Downloads/hadoop/logs/hadoop-hduser_-secondarnamenode-guru99-VirtualBox.out
hduser_@guru99-VirtualBox:~/MapReduceJoin$ SHADOOP_HOME/sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/guru99/Downloads/hadoop/logs/yarn-hduser_-resourcemanager-guru99-VirtualBox.out
localhost: starting nodemanager, logging to /home/guru99/Downloads/hadoop/logs/yarn-hduser_-nodemanager-guru99-VirtualBox.out
hduser_@guru99-VirtualBox:~/MapReduceJoin$ SHADOOP_HOME/bin/hdfs dfs -copyFromLocal DeptStrength.txt DeptName.txt /
hduser_@guru99-VirtualBox:~/MapReduceJoin$
```

Bước 5) DeptStrength.txt và DeptName.txt là các tệp đầu vào được sử dụng cho chương trình ví dụ MapReduce Join này.

Những tệp này cần được sao chép sang HDFS bằng lệnh bên dưới-

```
$HADOOP_HOME/bin/hdfs dfs -copyFromLocal DeptStrength.txt
DeptName.txt /
```

```
hduser_@guru99-VirtualBox:~/MapReduceJoin$ SHADOOP_HOME/bin/hdfs dfs -copyFromLocal DeptStrength.txt DeptName.txt /
hduser_@guru99-VirtualBox:~/MapReduceJoin$
```

Bước 6) Chạy chương trình bằng lệnh bên dưới-

```
$HADOOP_HOME/bin/hadoop jar MapReduceJoin.jar
MapReduceJoin/JoinDriver/DeptStrength.txt /DeptName.txt
/output_mapreducejoin
```

```
hduser_@guru99-VirtualBox:~/MapReduceJoin$ SHADOOP_HOME/bin/hadoop jar MapReduceJoin.jar /DeptStrength.txt
/DeptName.txt /output_mapreducejoin
```

```

@guru99-VirtualBox: ~/MapReduceJoin
14/06/09 14:24:00 INFO mapreduce.Job: map 100% reduce 100%
14/06/09 14:24:00 INFO mapreduce.Job: Job job_local320013666_0001 completed successfully
14/06/09 14:24:00 INFO mapreduce.Job: Counters: 32
  File System Counters
    FILE: Number of bytes read=26013
    FILE: Number of bytes written=586340
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=277
    HDFS: Number of bytes written=85
    HDFS: Number of read operations=28
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=5
  Map-Reduce Framework
    Map input records=8
    Map output records=8
    Map output bytes=117
    Map output materialized bytes=145
    Input split bytes=417
    Combine input records=0
    Combine output records=0
    Reduce input groups=4
    Reduce shuffle bytes=0
    Reduce input records=8
    Reduce output records=4
    Spilled Records=16
    Shuffled Maps =0
    Failed Shuffles=0
    Merged Map outputs=0
    GC time elapsed (ms)=682
    CPU time spent (ms)=0
    Physical memory (bytes) snapshot=0
    Virtual memory (bytes) snapshot=0
    Total committed heap usage (bytes)=457912320
  File Input Format Counters
    Bytes Read=0
  File Output Format Counters
    Bytes Written=85

```

Execution Done!

Bước 7) Sau khi thực thi, file đầu ra (có tên 'part-00000') sẽ được lưu trữ trong thư mục /output_mapreducejoin trên HDFS

Kết quả có thể được nhìn thấy bằng giao diện dòng lệnh

```
$HADOOP_HOME/bin/hdfs dfs -cat /output_mapreducejoin/part-00000
```

```

hduser_@guru99-VirtualBox:~/MapReduceJoin$ $HADOOP_HOME/bin/hdfs dfs -cat /output_mapreducejoin/part-00000
A11      50          Finance
B12     100           HR
C13     250 Manufacturing
Dept_ID Total_Employee      Dept_Name
hduser_@guru99-VirtualBox:~/MapReduceJoin$ 

```

Kết quả cũng có thể được nhìn thấy thông qua giao diện web như-

Hadoop NameNode localhost:54310

localhost:50070/dfshealth.jsp

Google

NameNode 'localhost:54310' (active)

Started:	Fri May 02 12:33:35 IST 2014
Version:	2.2.0, 1529768
Compiled:	2013-10-07T06:28Z by hortonmu from branch-2.2.0
Cluster ID:	CID-a1832593-cb99-4642-b3a5-043b8e204dbb
Block Pool ID:	BP-657563107-127.0.1.1-1398775824455

[Browse the filesystem](#)

[NameNode Logs](#)

Cluster Summary

Security is OFF

13 files and directories, 4 blocks = 17 total.

Heap Memory used 30.93 MB is 27% of Committed Heap Memory 114.25 MB. Max Heap Memory is 966.69 MB.
Non Heap Memory used 36.84 MB is 98% of Committed Non Heap Memory 37.31 MB. Max Non Heap Memory is -1 B.

Configured Capacity	:	35.26 GB
DFS Used	:	300 KB
Non DFS Used	:	6.62 GB
DFS Remaining	:	28.64 GB
DFS Used (%)	:	0.00%

bây giờ chọn 'Duyệt hệ thống tập tin' và điều hướng tối đa
/output_mapreducejoin

localhost:50075/browseDirectory.jsp?namenodeInfoPort=50070&dir=/&nnaddr=127.0.0.1:54310

snagit

Contents of directory /

Goto : / go

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
DeptName.txt	file	39 B	1	128 MB	2014-06-09 14:22	rw-r--r--	hduser	supergroup
DeptStrength.txt	file	50 B	1	128 MB	2014-06-09 14:22	rw-r--r--	hduser	supergroup
MapReduceTutorial	dir				2014-05-06 13:59	rwxr-xr-x	hduser	supergroup
SalesJan2009.csv	file	120.74 KB	1	128 MB	2014-05-06 15:32	rw-r--r--	hduser	supergroup
inputMapReduce	dir				2014-05-08 12:10	rwxr-xr-x	hduser	supergroup
mapreduce_output_sales	dir				2014-05-08 12:11	rwxr-xr-x	hduser	supergroup
output_mapreducejoin	dir				2014-06-09 14:24	rwxr-xr-x	hduser	supergroup
user	dir				2014-05-06 16:33	rwxr-xr-x	hduser	supergroup

[Go back to DFS home](#)

Local logs

localhost:50075/browseDirectory.jsp?dir=/output_mapreducejoin&namenodeInfoPort=50070&nnaddr=127.0.0.1:54310

Mở phần-r-00000

Contents of directory /output_mapreducejoin

Goto : /output_mapreducejoin

[Go to parent directory](#)

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
SUCCESS	file	0 B	1	128 MB	2014-06-09 14:24	rw-r--r--	hduser	supergroup
part-00000	file	85 B	1	128 MB	2014-06-09 14:23	rw-r--r--	hduser	supergroup

[Go back to DFS home](#)

Local logs

[Log directory](#)

[Hadoop, 2014.](#)

Kết quả được hiển thị

File: /output_mapreducejoin/part-00000

Goto : /output_mapreducejoin

[Go back to dir listing](#)

[Advanced view/download options](#)

All	50	Finance
B12	100	HR
C13	250	Manufacturing
Dept_ID	Total_Employee	Dept_Name

[Download this file](#)

[Tail this file](#)

LƯU Ý: Xin lưu ý rằng trước khi chạy chương trình này vào lần tiếp theo, bạn sẽ cần xóa thư mục đầu ra /output_mapreducejoin

```
$HADOOP_HOME/bin/hdfs dfs -rm -r /output_mapreducejoin
```

Cách khác là sử dụng tên khác cho thư mục đầu ra.

NHỮNG BÀI VIẾT LIÊN QUAN

- Dữ liệu lớn là gì? Giới thiệu, các loại, đặc điểm, ví dụ
- Hadoop là gì? Giới thiệu, Archikiến trúc, Ecosystem, Thành phần

Bộ đếm trong MapReduce là gì?

A **Bộ đếm trong MapReduce** là một cơ chế được sử dụng để thu thập và đo lường thông tin thống kê về các công việc và sự kiện MapReduce. Bộ đếm theo dõi các số liệu thống kê công việc khác nhau trong MapReduce như số lượng

operanhững vấn đề đã xảy ra và tiến triển của operasự. Bộ đếm được sử dụng để chẩn đoán sự cố trong MapReduce.

Bộ đếm của Hadoop tương tự như việc đặt thông điệp tường trình vào mã cho bản đồ hoặc thu nhỏ. Thông tin này có thể hữu ích cho việc chẩn đoán sự cố trong quá trình xử lý công việc MapReduce.

Thông thường, các bộ đếm này trong Hadoop được xác định trong một chương trình (ánh xạ hoặc rút gọn) và được tăng lên trong quá trình thực thi khi xảy ra một sự kiện hoặc điều kiện cụ thể (cụ thể cho bộ đếm đó). Một ứng dụng rất hay của bộ đếm Hadoop là theo dõi các bản ghi hợp lệ và không hợp lệ từ tập dữ liệu đầu vào.

Các loại bộ đếm MapReduce

Về cơ bản có 2 loại [Bản đồGiảm](#) Counters

1. **Bộ đếm tích hợp của Hadoop:** Có một số bộ đếm Hadoop tích hợp tồn tại cho mỗi công việc. Dưới đây là các nhóm truy cập tích hợp-

- **Bộ đếm tác vụ MapReduce** – Thu thập thông tin cụ thể của nhiệm vụ (ví dụ: số lượng bản ghi đầu vào) trong thời gian thực hiện.
- **Bộ đếm hệ thống tập tin** – Thu thập thông tin như số byte được đọc hoặc ghi bởi một tác vụ
- **Bộ đếm định dạng đầu vào tệp** – Thu thập thông tin về một số byte được đọc qua FileInputFormat
- **Bộ đếm định dạng đầu ra tệp** – Thu thập thông tin về một số byte được ghi thông qua FileOutputFormat
- **Bộ đếm công việc** – Những bộ đếm này được JobTracker sử dụng. Số liệu thống kê được họ thu thập bao gồm số lượng nhiệm vụ được đưa ra cho một công việc.

2. **Bộ đếm do người dùng xác định**

Ngoài các bộ đếm tích hợp, người dùng có thể xác định bộ đếm của riêng mình bằng cách sử dụng các chức năng tương tự được cung cấp bởi [ngôn ngữ lập trình](#). Ví dụ, trong [Java](#) 'enum' được sử dụng để xác định các bộ đếm do người dùng xác định.

Ví dụ về bộ đếm

Một ví dụ MapClass có Bộ đếm để đếm số lượng giá trị bị thiếu và không hợp lệ. Tập dữ liệu đầu vào được sử dụng trong hướng dẫn này Tập dữ liệu đầu vào của chúng tôi là tệp CSV, [Bán hàngJan2009.csv](#)

```

public static class MapClass
    extends MapReduceBase
    implements Mapper<LongWritable, Text, Text, Text>
{
    static enum SalesCounters { MISSING, INVALID };

    public void map ( LongWritable key, Text value,
                      OutputCollector<Text, Text> output,
                      Reporter reporter) throws IOException
    {

        //Input string is split using ',' and stored in 'fields'
        array
        String fields[] = value.toString().split(",", -20);
        //Value at 4th index is country. It is stored in 'country'
        variable
        String country = fields[4];

        //Value at 8th index is sales data. It is stored in 'sales'
        variable
        String sales = fields[8];

        if (country.length() == 0) {
            reporter.incrCounter(SalesCounters.MISSING, 1);
        } else if (sales.startsWith("\\")) {
            reporter.incrCounter(SalesCounters.INVALID, 1);
        } else {
            output.collect(new Text(country), new Text(sales +
",1"));
        }
    }
}

```

Đoạn mã trên hiển thị một ví dụ về triển khai bộ đếm trong Hadoop Map Giảm.

Ở đây, **Quầy bán hàng** là một bộ đếm được xác định bằng cách sử dụng 'enum'. Nó được dùng để đếm **MISSING** và **KHÔNG HỢP LỆ** các bản ghi đầu vào.

Trong đoạn mã, nếu '**quốc gia**' trường có độ dài bằng 0 thì giá trị của nó bị thiếu và do đó bộ đếm tương ứng **Bộ đếm bán hàng.MISSING** được tăng lên.

Tiếp theo, nếu 'việc bán hàng' trường bắt đầu bằng một " thì bản ghi được coi là KHÔNG HỢP LỆ. Điều này được biểu thị bằng bộ đếm tăng dần **Bộ đếm doanh số.INVALID.**



Bạn có thể thích:



60 câu hỏi phỏng vấn Hadoop hàng đầu và...



Ví dụ về Hadoop và Mapreduce: Tạo...



Cách cài đặt Hadoop theo từng bước...



Hadoop là gì? Giới thiệu,...

← Trước

Báo cáo lỗi

Sau →



Trụ sở chính của Guru99

4023 Kennett Pike #50286,
Wilmington, Delaware,
Hoa Kỳ



Giới thiệu

Về Chúng Tôi

Quảng cáo với chúng tôi

Viết Đổi với chúng tôi

Liên lạc

Gợi ý nghề nghiệp

SAP Công cụ gợi ý nghề nghiệp

Kiểm thử phần mềm như một nghề nghiệp

Thú vị

Sách điện tử (eBook)

Nhật Ký

Bài kiểm tra

SAP Sách điện tử (eBook)

Quản lý sự riêng tư



Vietnamese

||| quyển - Gửi tin 2024 | Chính sách bảo vệ

[Thông tin](#) | [Tuyên bố từ chối liên kết](#) | [ToS](#) |

[Chính sách biên tập](#)