# **Group Operator**

Cú pháp

Được đưa ra dưới đây là cú pháp của toán tử **nhóm**.

grunt> Group\_data = GROUP Relation\_name BY age;

Ví dụ

Giả sử rằng chúng ta có một tệp có tên **student\_details.txt** trong thư mục HDFS **/ pig\_data /** như được hiển thị dưới đây.

**student\_details.txt**

001,Rajiv,Reddy,21,9848022337,Hyderabad

002,siddarth,Battacharya,22,9848022338,Kolkata

003,Rajesh,Khanna,22,9848022339,Delhi

004,Preethi,Agarwal,21,9848022330,Pune

005,Trupthi,Mohanthy,23,9848022336,Bhuwaneshwar

006,Archana,Mishra,23,9848022335,Chennai

007,Komal,Nayak,24,9848022334,trivendram

008,Bharathi,Nambiayar,24,9848022333,Chennai

Và chúng ta đã tải tệp này vào Apache Pig với tên quan hệ **student\_details** như hình dưới đây.

grunt> student\_details = LOAD 'hdfs://localhost:9000/pig\_data/student\_details.txt' USING PigStorage(',')

as (id:int, firstname:chararray, lastname:chararray, age:int, phone:chararray, city:chararray);

Bây giờ, chúng ta hãy nhóm các bản ghi / tuples trong mối quan hệ theo độ tuổi như được hiển thị dưới đây.

grunt> group\_data = GROUP student\_details by age;

Xác minh

Verify the relation **group\_data** using the **DUMP** operator as shown below.

grunt> Dump group\_data;

Output

Sau đó, bạn sẽ nhận được đầu ra hiển thị nội dung của mối quan hệ được đặt tên **group\_data** như hình dưới đây. Ở đây bạn có thể quan sát thấy rằng lược đồ kết quả có hai cột −

* Một là **tuổi** tác, theo đó chúng ta đã nhóm mối quan hệ.

1. Cái còn lại là một **chiếc túi**, trong đó có nhóm tuples, hồ sơ sinh viên với độ tuổi tương ứng.

(21,{(4,Preethi,Agarwal,21,9848022330,Pune),(1,Rajiv,Reddy,21,9848022337,Hydera bad)})

(22,{(3,Rajesh,Khanna,22,9848022339,Delhi),(2,siddarth,Battacharya,22,984802233 8,Kolkata)})

(23,{(6,Archana,Mishra,23,9848022335,Chennai),(5,Trupthi,Mohanthy,23,9848022336 ,Bhuwaneshwar)})

(24,{(8,Bharathi,Nambiayar,24,9848022333,Chennai),(7,Komal,Nayak,24,9848022334, trivendram)})

Bạn có thể xem lược đồ của bảng sau khi nhóm dữ liệu bằng cách sử dụng lệnh **mô tả** như hình dưới đây.

**grunt> Describe group\_data;**

group\_data: {group: int,student\_details: {(id: int,firstname: chararray,

lastname: chararray,age: int,phone: chararray,city: chararray)}}

Theo cách tương tự, bạn có thể nhận được hình minh họa mẫu của lược đồ bằng cách sử dụng lệnh **minh họa** như hình dưới đây.

$ Illustrate group\_data;

Nó sẽ tạo ra đầu ra sau đây −

-------------------------------------------------------------------------------------------------

|group\_data| group:int | student\_details:bag{:tuple(id:int,firstname:chararray,lastname:chararray,age:int,phone:chararray,city:chararray)}|

-------------------------------------------------------------------------------------------------

| | 21 | { 4, Preethi, Agarwal, 21, 9848022330, Pune), (1, Rajiv, Reddy, 21, 9848022337, Hyderabad)}|

| | 2 | {(2,siddarth,Battacharya,22,9848022338,Kolkata),(003,Rajesh,Khanna,22,9848022339,Delhi)}|

-------------------------------------------------------------------------------------------------

Nhóm theo Nhiều Cột

Chúng ta hãy nhóm mối quan hệ theo độ tuổi và thành phố như hình dưới đây.

grunt> group\_multiple = GROUP student\_details by (age, city);

Bạn có thể xác minh nội dung của mối quan hệ được đặt tên **group\_multiple** bằng cách sử dụng toán tử Dump như hình dưới đây.

**grunt> Dump group\_multiple;**

((21,Pune),{(4,Preethi,Agarwal,21,9848022330,Pune)})

((21,Hyderabad),{(1,Rajiv,Reddy,21,9848022337,Hyderabad)})

((22,Delhi),{(3,Rajesh,Khanna,22,9848022339,Delhi)})

((22,Kolkata),{(2,siddarth,Battacharya,22,9848022338,Kolkata)})

((23,Chennai),{(6,Archana,Mishra,23,9848022335,Chennai)})

((23,Bhuwaneshwar),{(5,Trupthi,Mohanthy,23,9848022336,Bhuwaneshwar)})

((24,Chennai),{(8,Bharathi,Nambiayar,24,9848022333,Chennai)})

(24,trivendram),{(7,Komal,Nayak,24,9848022334,trivendram)})

Nhóm Tất cả

Bạn có thể nhóm một mối quan hệ theo tất cả các cột như hình dưới đây.

grunt> **group\_all** = GROUP **student\_details** All;

Bây giờ, hãy xác minh nội dung của mối quan hệ **group\_all** như hình dưới đây.

**grunt> Dump group\_all;**

(all,{(8,Bharathi,Nambiayar,24,9848022333,Chennai),(7,Komal,Nayak,24,9848022334 ,trivendram),

(6,Archana,Mishra,23,9848022335,Chennai),(5,Trupthi,Mohanthy,23,9848022336,Bhuw aneshwar),

(4,Preethi,Agarwal,21,9848022330,Pune),(3,Rajesh,Khanna,22,9848022339,Delhi),

(2,siddarth,Battacharya,22,9848022338,Kolkata),(1,Rajiv,Reddy,21,9848022337

# **Cogroup Operator**

## Nhóm hai quan hệ bằng cogroup

Giả sử rằng chúng ta có hai tệp **là student\_details.txt** và **employee\_details.txt** trong thư mục HDFS **/ pig\_data /** như được hiển thị dưới đây.

**student\_details.txt**

001,Rajiv,Reddy,21,9848022337,Hyderabad

002,siddarth,Battacharya,22,9848022338,Kolkata

003,Rajesh,Khanna,22,9848022339,Delhi

004,Preethi,Agarwal,21,9848022330,Pune

005,Trupthi,Mohanthy,23,9848022336,Bhuwaneshwar

006,Archana,Mishra,23,9848022335,Chennai

007,Komal,Nayak,24,9848022334,trivendram

008,Bharathi,Nambiayar,24,9848022333,Chennai

**employee\_details.txt**

001,Robin,22,newyork

002,BOB,23,Kolkata

003,Maya,23,Tokyo

004,Sara,25,London

005,David,23,Bhuwaneshwar

006,Maggy,22,Chennai

Và chúng ta đã tải các tệp này vào Pig với tên mối quan hệ **student\_details** và **employee\_details** tương ứng, như được hiển thị dưới đây.

grunt> student\_details = LOAD 'hdfs://localhost:9000/pig\_data/student\_details.txt' USING PigStorage(',')

as (id:int, firstname:chararray, lastname:chararray, age:int, phone:chararray, city:chararray);

grunt> employee\_details = LOAD 'hdfs://localhost:9000/pig\_data/employee\_details.txt' USING PigStorage(',')

as (id:int, name:chararray, age:int, city:chararray);

Bây giờ, chúng ta hãy nhóm các hồ sơ / tuples của các mối quan hệ **student\_details** và **employee\_details** với độ tuổi chính, như được hiển thị dưới đây.

grunt> cogroup\_data = COGROUP student\_details by age, employee\_details by age;

### **Xác minh**

Xác minh mối quan hệ **cogroup\_data** bằng cách sử dụng toán tử **DUMP** như hình dưới đây.

grunt> Dump cogroup\_data;

### **Output**

Nó sẽ tạo ra đầu ra sau đây, hiển thị nội dung của mối quan hệ được đặt tên **cogroup\_data** như hình dưới đây.

(21,{(4,Preethi,Agarwal,21,9848022330,Pune), (1,Rajiv,Reddy,21,9848022337,Hyderabad)},

{ })

(22,{ (3,Rajesh,Khanna,22,9848022339,Delhi), (2,siddarth,Battacharya,22,9848022338,Kolkata) },

{ (6,Maggy,22,Chennai),(1,Robin,22,newyork) })

(23,{(6,Archana,Mishra,23,9848022335,Chennai),(5,Trupthi,Mohanthy,23,9848022336 ,Bhuwaneshwar)},

{(5,David,23,Bhuwaneshwar),(3,Maya,23,Tokyo),(2,BOB,23,Kolkata)})

(24,{(8,Bharathi,Nambiayar,24,9848022333,Chennai),(7,Komal,Nayak,24,9848022334, trivendram)},

{ })

(25,{ },

{(4,Sara,25,London)})

Nhà điều hành **cogroup** nhóm các tuples từ mỗi mối quan hệ theo độ tuổi mà mỗi nhóm mô tả một giá trị tuổi cụ thể.

# **Join Operator**

Toán **tử JOIN** được sử dụng để kết hợp các bản ghi từ hai hoặc nhiều quan hệ. Trong khi thực hiện một thao tác nối, chúng tôi tuyên bố một (hoặc một nhóm) tuple từ mỗi mối quan hệ, là chìa khóa. Khi các phím này khớp, hai tuples cụ thể được khớp, nếu không các bản ghi bị bỏ. Tham gia có thể thuộc các loại sau đây −

* Self-join
* Inner-join
* Outer-join − left join, right join, and full join

Chương này giải thích với các ví dụ về cách sử dụng toán tử tham gia trong Tiếng Latinh lợn. Giả sử rằng chúng ta có hai tệp cụ thể là **customers.txt** and **orders.txt** in the **/pig\_data/** thư mục của HDFS như hình dưới đây.

**customers.txt**

1,Ramesh,32,Ahmedabad,2000.00

2,Khilan,25,Delhi,1500.00

3,kaushik,23,Kota,2000.00

4,Chaitali,25,Mumbai,6500.00

5,Hardik,27,Bhopal,8500.00

6,Komal,22,MP,4500.00

7,Muffy,24,Indore,10000.00

**orders.txt**

102,2009-10-08 00:00:00,3,3000

100,2009-10-08 00:00:00,3,1500

101,2009-11-20 00:00:00,2,1560

103,2008-05-20 00:00:00,4,2060

Và chúng tôi đã tải hai tập tin này vào Pig với các mối quan hệ **customers** và **orders** như hình dưới đây.

grunt> customers = LOAD 'hdfs://localhost:9000/pig\_data/customers.txt' USING PigStorage(',')

as (id:int, name:chararray, age:int, address:chararray, salary:int);

grunt> orders = LOAD 'hdfs://localhost:9000/pig\_data/orders.txt' USING PigStorage(',')

as (oid:int, date:chararray, customer\_id:int, amount:int);

Bây giờ chúng ta hãy thực hiện các hoạt động tham gia khác nhau về hai mối quan hệ này.

## Self - join

**Tự tham gia** được sử dụng để tham gia một bảng với chính nó như thể bảng là hai mối quan hệ, tạm thời đổi tên ít nhất một mối quan hệ.

Nói chung, trong Apache Pig, để thực hiện tự tham gia, chúng tôi sẽ tải cùng một dữ liệu nhiều lần, dưới các bí danh khác nhau (tên). Do đó, hãy để chúng tôi tải nội dung của **khách hàng tệp.txt** như hai bảng như hình dưới đây.

grunt> customers1 = LOAD 'hdfs://localhost:9000/pig\_data/customers.txt' USING PigStorage(',')

as (id:int, name:chararray, age:int, address:chararray, salary:int);

grunt> customers2 = LOAD 'hdfs://localhost:9000/pig\_data/customers.txt' USING PigStorage(',')

as (id:int, name:chararray, age:int, address:chararray, salary:int);

### **Cú pháp**

Được đưa ra dưới đây là cú pháp thực hiện **self-join** thao tác sử dụng toán tử **JOIN**

grunt> Relation3\_name = JOIN Relation1\_name BY key, Relation2\_name BY key ;

### **Ví dụ**

Hãy để chúng tôi thực hiện hoạt động **self-join** vào mối quan hệ **custormer**, bằng cách tham gia hai mối quan hệ **custormer1** và **custormer**2 như hình dưới đây.

grunt> customers3 = JOIN customers1 BY id, customers2 BY id;

### **Xác minh**

Xác minh mối quan hệ **khách hàng3** bằng cách sử dụng nhà điều hành DUMP như hình dưới đây.

grunt> Dump customers3;

### **Output**

Nó sẽ tạo ra đầu ra sau đây, hiển thị nội dung của mối quan hệ **customers**.

(1,Ramesh,32,Ahmedabad,2000,1,Ramesh,32,Ahmedabad,2000)

(2,Khilan,25,Delhi,1500,2,Khilan,25,Delhi,1500)

(3,kaushik,23,Kota,2000,3,kaushik,23,Kota,2000)

(4,Chaitali,25,Mumbai,6500,4,Chaitali,25,Mumbai,6500)

(5,Hardik,27,Bhopal,8500,5,Hardik,27,Bhopal,8500)

(6,Komal,22,MP,4500,6,Komal,22,MP,4500)

(7,Muffy,24,Indore,10000,7,Muffy,24,Indore,10000)

## Inner Join

**Inner Join** được sử dụng khá thường xuyên; nó còn được gọi là **equijoin**. Một nối bên trong trả về các hàng khi có một trận đấu trong cả hai bảng.

Nó tạo ra một mối quan hệ mới bằng cách kết hợp các giá trị cột của hai mối quan hệ (nói A và B) dựa trên vị ngữ nối. Truy vấn so sánh từng hàng A với mỗi hàng B để tìm tất cả các cặp hàng đáp ứng vị ngữ nối. Khi vị ngữ nối được thỏa mãn, các giá trị cột cho mỗi cặp hàng A và B khớp được kết hợp thành một hàng kết quả.

### **Cú pháp**

Dưới đây là cú pháp thực hiện thao tác **innerjoin** bằng cách sử dụng toán tử **JOIN**.

grunt> result = JOIN relation1 BY columnname, relation2 BY columnname;

### **Ví dụ**

grunt> coustomer\_orders = JOIN customers BY id, orders BY customer\_id;

### **Xác minh**

Xác minh mối quan hệ **coustomer\_orders** bằng cách sử dụng toán tử **DUMP** như hình dưới đây.

grunt> Dump coustomer\_orders;

### **Output**

Bạn sẽ nhận được đầu ra sau đây sẽ là nội dung của mối quan hệ được đặt tên **coustomer\_orders**.

(2,Khilan,25,Delhi,1500,101,2009-11-20 00:00:00,2,1560)

(3,kaushik,23,Kota,2000,100,2009-10-08 00:00:00,3,1500)

(3,kaushik,23,Kota,2000,102,2009-10-08 00:00:00,3,3000)

(4,Chaitali,25,Mumbai,6500,103,2008-05-20 00:00:00,4,2060)

## Left Outer Join

**left outer Join** thao tác trả về tất cả các hàng từ bảng bên trái, ngay cả khi không có khớp trong mối quan hệ bên phải.

### **Cú pháp**

cú pháp thực hiện thao tác **left outer Join** bằng cách sử dụng toán tử **JOIN**.

grunt> Relation3\_name = JOIN Relation1\_name BY id LEFT OUTER, Relation2\_name BY customer\_id;

### **Ví dụ**

grunt> outer\_left = JOIN customers BY id LEFT OUTER, orders BY customer\_id;

### **Xác minh**

Xác minh mối quan hệ **outer\_left** sử dụng toán tử **DUMP** như hình dưới đây.

grunt> Dump outer\_left;

### **Output**

Nó sẽ tạo ra đầu ra sau đây, hiển thị nội dung của mối quan hệ **outer\_left**.

(1,Ramesh,32,Ahmedabad,2000,,,,)

(2,Khilan,25,Delhi,1500,101,2009-11-20 00:00:00,2,1560)

(3,kaushik,23,Kota,2000,100,2009-10-08 00:00:00,3,1500)

(3,kaushik,23,Kota,2000,102,2009-10-08 00:00:00,3,3000)

(4,Chaitali,25,Mumbai,6500,103,2008-05-20 00:00:00,4,2060)

(5,Hardik,27,Bhopal,8500,,,,)

(6,Komal,22,MP,4500,,,,)

(7,Muffy,24,Indore,10000,,,,)

## Right Outer Join

**right outer join** thao tác trả về tất cả các hàng từ bảng bên phải, ngay cả khi không có khớp trong bảng bên trái.

### **Cú pháp**

Được đưa ra dưới đây là cú pháp thực hiện thao tác **right outer join**  bằng cách sử dụng toán tử **JOIN**.

grunt> outer\_right = JOIN customers BY id RIGHT, orders BY customer\_id;

### **Ví dụ**

grunt> outer\_right = JOIN customers BY id RIGHT, orders BY customer\_id;

### **Xác minh**

Xác minh mối quan hệ **outer\_right** sử dụng toán tử **DUMP** như hình dưới đây.

grunt> Dump outer\_right

### **Output**

Nó sẽ tạo ra đầu ra sau đây, hiển thị nội dung của mối quan hệ **outer\_right**.

(2,Khilan,25,Delhi,1500,101,2009-11-20 00:00:00,2,1560)

(3,kaushik,23,Kota,2000,100,2009-10-08 00:00:00,3,1500)

(3,kaushik,23,Kota,2000,102,2009-10-08 00:00:00,3,3000)

(4,Chaitali,25,Mumbai,6500,103,2008-05-20 00:00:00,4,2060)

## Full Outer Join

**full outer join** trả về các hàng khi có một trận đấu trong một trong các mối quan hệ.

### **Cú pháp**

grunt> outer\_full = JOIN customers BY id FULL OUTER, orders BY customer\_id;

### **Ví dụ**

grunt> outer\_full = JOIN customers BY id FULL OUTER, orders BY customer\_id;

### **Xác minh**

Xác minh mối quan hệ **outer\_full** bằng cách sử dụng toán tử **DUMP** như hình dưới đây.

grun> Dump outer\_full;

### **Output**

Nó sẽ tạo ra đầu ra sau đây, hiển thị nội dung của mối quan hệ **outer\_full**.

(1,Ramesh,32,Ahmedabad,2000,,,,)

(2,Khilan,25,Delhi,1500,101,2009-11-20 00:00:00,2,1560)

(3,kaushik,23,Kota,2000,100,2009-10-08 00:00:00,3,1500)

(3,kaushik,23,Kota,2000,102,2009-10-08 00:00:00,3,3000)

(4,Chaitali,25,Mumbai,6500,103,2008-05-20 00:00:00,4,2060)

(5,Hardik,27,Bhopal,8500,,,,)

(6,Komal,22,MP,4500,,,,)

(7,Muffy,24,Indore,10000,,,,)

## Using Multiple Keys

Chúng ta có thể thực hiện thao tác JOIN bằng cách sử dụng multiple keys.

### **Cú pháp**

Dưới đây là cách bạn có thể thực hiện thao tác JOIN trên hai bảng bằng cách sử dụng multiple keys.

grunt> Relation3\_name = JOIN Relation2\_name BY (key1, key2), Relation3\_name BY (key1, key2);

Giả sử rằng chúng tôi có hai tệp **employee.txt** và **employee\_contact.txt** trong thư **mục / pig\_data /** của HDFS như được hiển thị dưới đây.

**employee.txt**

001,Rajiv,Reddy,21,programmer,003

002,siddarth,Battacharya,22,programmer,003

003,Rajesh,Khanna,22,programmer,003

004,Preethi,Agarwal,21,programmer,003

005,Trupthi,Mohanthy,23,programmer,003

006,Archana,Mishra,23,programmer,003

007,Komal,Nayak,24,teamlead,002

008,Bharathi,Nambiayar,24,manager,001

**employee\_contact.txt**

001,9848022337,Rajiv@gmail.com,Hyderabad,003

002,9848022338,siddarth@gmail.com,Kolkata,003

003,9848022339,Rajesh@gmail.com,Delhi,003

004,9848022330,Preethi@gmail.com,Pune,003

005,9848022336,Trupthi@gmail.com,Bhuwaneshwar,003

006,9848022335,Archana@gmail.com,Chennai,003

007,9848022334,Komal@gmail.com,trivendram,002

008,9848022333,Bharathi@gmail.com,Chennai,001

Và chúng tôi đã tải hai tập tin này vào Pig với **employee** quan hệ và **employee\_contact** như hình dưới đây.

grunt> employee = LOAD 'hdfs://localhost:9000/pig\_data/employee.txt' USING PigStorage(',')

as (id:int, firstname:chararray, lastname:chararray, age:int, designation:chararray, jobid:int);

grunt> employee\_contact = LOAD 'hdfs://localhost:9000/pig\_data/employee\_contact.txt' USING PigStorage(',')

as (id:int, phone:chararray, email:chararray, city:chararray, jobid:int);

Bây giờ, chúng ta hãy tham gia nội dung của hai mối quan hệ này bằng cách sử dụng toán tử JOIN như hình dưới đây.

grunt> emp = JOIN employee BY (id,jobid), employee\_contact BY (id,jobid);

### **Xác minh**

Xác minh **emp** mối quan hệ bằng cách sử dụng toán tử **DUMP** như hình dưới đây.

grunt> Dump emp;

### **Output**

Nó sẽ tạo ra đầu ra sau đây, hiển thị nội dung của mối quan hệ được đặt tên **là EMP** như hình dưới đây.

(1,Rajiv,Reddy,21,programmer,113,1,9848022337,Rajiv@gmail.com,Hyderabad,113)

(2,siddarth,Battacharya,22,programmer,113,2,9848022338,siddarth@gmail.com,Kolka ta,113)

(3,Rajesh,Khanna,22,programmer,113,3,9848022339,Rajesh@gmail.com,Delhi,113)

(4,Preethi,Agarwal,21,programmer,113,4,9848022330,Preethi@gmail.com,Pune,113)

(5,Trupthi,Mohanthy,23,programmer,113,5,9848022336,Trupthi@gmail.com,Bhuwaneshw ar,113)

(6,Archana,Mishra,23,programmer,113,6,9848022335,Archana@gmail.com,Chennai,113)

(7,Komal,Nayak,24,teamlead,112,7,9848022334,Komal@gmail.com,trivendram,112)

(8,Bharathi,Nambiayar,24,manager,111,8,9848022333,Bharathi@gmail.com,Chennai,1