**<https://shrikantbang.wordpress.com/2014/01/14/apache-pig-group-by-nested-foreach-join-example/>**

**Apache Pig : Group By, Nested Foreach, Join Example**

**Input Data:**

For experimental purpose we have generated dummy test data.  
We have following dataset for operations.

1. **CUSTOMER’S ORDER HISTORY DATA :**

This data contains the orders placed by customer. For example customer with id ‘A’ had ordered item ‘I’. Order date in milliseconds was ‘1391230800000’ and Delivery date in milliseconds was ‘1391317200000’.  
Schema : ( customerId:int, itemId:int, orderDate:long, deliveryDate:long)  
You can generate the dummy data using following script:

|  |  |
| --- | --- |
|  | #!/usr/bin/perl |
|  | use Time::localtime; |
|  |  |
|  | for($i=0 ; $i<100000 ; $i++ ){ |
|  | $customer\_id=int(rand(100)); |
|  | $item\_id=int(rand(1000)); |
|  | $epoc = time(); |
|  | $one\_day\_sec=86400; |
|  | $one\_hour\_sec=3600; |
|  | $random\_time=int(rand(15)\* $one\_day\_sec +$one\_hour\_sec); |
|  | $date1=int(($epoc-$random\_time)\*1000+rand(1234)); |
|  | $date2=int(($epoc+$random\_time)\*1000+rand(1234)); |
|  |  |
|  | print "$customer\_id,$item\_id,$date1,$date2\n"; |
|  | } |

[**view raw**](https://gist.github.com/bshrikant28/8405586/raw/6cfb1bc4c4da388d8d3f0a2f7b1b4ec76367c79c/generate_order_data)[**generate\_order\_data**](https://gist.github.com/bshrikant28/8405586#file-generate_order_data) hosted with  by [**GitHub**](https://github.com/)

1. **CUSTOMER’S BASIC INFORMATION DATA :**

This data contains the basic information of customer. For example customer Id, name and city.  
Schema : ( custoerId:int, name:chararray, city:chararry)  
You can generate the dummy test data using following script:

|  |  |
| --- | --- |
|  | #!/usr/bin/perl |
|  | use String::Random; |
|  |  |
|  | @city = ("CityA", "CityB", "CityC", "CityD", "CityE", "CityF", "CityG", "CityH", "CityI", "CityJ"); |
|  |  |
|  | for($cnt=0;$cnt<100;$cnt++){ |
|  | $r1 = int(rand(10)); |
|  | $random = new String::Random; |
|  | $name = $random->randpattern(CcccCccc); |
|  | print "$cnt,$name,$city[$r1]\n"; |
|  | } |

[**view raw**](https://gist.github.com/bshrikant28/8405618/raw/9ffab253d8d1cce96d738a590aefa80bc96c5fdf/customer_info)[**customer\_info**](https://gist.github.com/bshrikant28/8405618#file-customer_info) hosted with  by [**GitHub**](https://github.com/)

**Examples:**

1. GROUP BY:  
   Lets start with ‘group by’ operation.  
   **Problem Statement:**  
   Find the number of items bought by each customer.  
   **Input:**  
   Customer’s order history data.  
   **Output:**  
   Output should contain the total number of items bought by each customer.  
   Schema of output should be : (customerId:int , itmCnt:int );

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | -- Problem Stmt : find the number of items bought by each customer.  -- load the input data :: Schema ( customerId , itemId , order Date, delivery Date );  orders = load '/testData100k' using PigStorage(',') as (cstrId:int, itmId:int, orderDate: long, deliveryDate: long );  -- group input by customer id  grpd = group orders by cstrId;  -- count number of items bought by each customer  items\_by\_customer = foreach grpd generate group as cstrId, COUNT(orders) as itemCnt;  describe items\_by\_customer;  --items\_by\_customer: {cstrId: int,itemCnt: long} |

1. NESTED FOREACH:  
   Now lets discuss about Nested Foreach Operation. We can use foreach operator to iterate over input records, but we can also apply different relational operations to each record in same data processing pipeline.

**Problem Statement:**  
Find the total number of items bought by each customer and which item he/she bought higest times. For example Customer A has bought items ‘i1’ 10 times , ‘i2’ 5 times and ‘i3’ 7 times. So total number of items taken by Customer A is 10+5+7=22, out of which item ‘i1’ is taken higest times.  
**Input:**  
Customer’s order history data.  
**Output:**  
Output should contain total number of items bought by customer and which of item he/she bought higest time.  
Schema of output should be :  
(customerId:int, itemId:int , higestBoughtItemCnt:long, totalItemCnt:long)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | -- Problem Stmt : find the number of items bought by each customer  -- which item he/she bought highest time.  -- load the input data :: Schema ( customerId , itemId , order Date, delivery Date );  orders = load '/testData100k' using PigStorage(',') as (cstrId:int, itmId:int, orderDate: long, deliveryDate: long );  -- group by  custorer-id and item-id  grpd\_cstr\_itm = group orders by (cstrId,itmId);  grpd\_cstr\_itm\_cnt = foreach grpd\_cstr\_itm generate group.cstrId as cstrId, group.itmId as itmId, COUNT(orders) as itmCnt;  -- group by cstrId  grpd\_cstr = group grpd\_cstr\_itm\_cnt by cstrId ;  describe grpd\_cstr;  -- grpd\_cstr: {group: int,grpd\_cstr\_itm\_cnt: {cstrId: int,itmId: int,itmCnt: long}}  -- iterate over grpd\_cstr\_itm and find total number of items bought by customer and which item he/or she bought higest time.  result = foreach grpd\_cstr{      total\_orders = SUM(grpd\_cstr\_itm\_cnt.itmCnt);      srtd\_orders = order grpd\_cstr\_itm\_cnt by itmCnt desc;      higest\_bought = limit srtd\_orders 1;      generate FLATTEN(higest\_bought),total\_orders as totalCnt;  }; |

|  |
| --- |
| -- result will contains ( customer\_id , itm\_id\_bought\_higest\_times, number\_of\_times\_it\_bought, total\_items);  describe result;  -- result: {higest\_bought::cstrId: int,higest\_bought::itmId: int,higest\_bought::itmCnt: long,totalCnt: long} |

1. JOIN:  
   Like in RDBMS, join operator in Pig join datasets based on values common to each dataset.  
   We will join customer order history dataset with customer information dataset to get interesting result.  
   **Problem Statement:**  
   Find the total number of items ordered by customers by city.  
   For example customers from CityA has ordered item ‘i1’ 300 times. Customer from CityB ordered item ‘i4’ 129 times.  
   **Input:**  
   Customer’s order history data.  
   Customer’s basic information data.  
   **Output:**  
   Output should contain total number of items ordered by customers by city.  
   Schema of output should be :  
   (itemId:int, city:chararray, totalItemCnt:long)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | -- Problem Stmt : find the a perticular items ordered by customers by city.  -- For e.g. Customers from CityA has ordered the item 'item1' 350 times.  -- load the input data :: Schema ( customerId , itemId , order Date, delivery Date );  orders = load '/testData100k' using PigStorage(',') as (cstrId:int, itmId:int, orderDate: long, deliveryDate: long );  -- load the customer information data :: (customerId , name , city )  cstr\_info = load '/customerInformation' using PigStorage(',') as (cstrId:int, name:chararray, city:chararray);  -- join orders and customer\_info by cstrId;  jnd = join orders by cstrId, cstr\_info by cstrId;  describe jnd;  -- jnd: {orders::cstrId: int,orders::itmId: int,orders::orderDate: long,orders::deliveryDate: long,cstr\_info::cstrId: int,cstr\_info::name: chararray,cstr\_info::city: chararray}  -- group  by itemId and city  jnd\_grp = group jnd by (orders::itmId, cstr\_info::city);  describe jnd\_grp;  -- jnd\_grp: {group: (orders::itmId: int,cstr\_info::city: chararray),jnd: {orders::cstrId: int,orders::itmId: int,orders::orderDate: long,orders::deliveryDate: long,cstr\_info::cstrId: int,cstr\_info::name: chararray,cstr\_info::city: chararray}}  -- lets count and generate the result.  result = foreach jnd\_grp generate FLATTEN(group) , COUNT(jnd) as cnt;  describe result;  --result: {group::orders::itmId: int,group::cstr\_info::city: chararray,cnt: long} |

Advertisements