MapReduce Exercises

April 4, 2016

Q.1

Consider the following table snippet:

Author	Paper Title	Citations
Claudio Gutierrez	Semantics and Complexity of SPARQL	320
Claudio Gutierrez	Survey of graph database models	315
Claudio Gutierrez	Foundations of semantic web databases	232
Claudio Gutierrez	The expressive power of SPARQL	157
Claudio Gutierrez	Minimal deductive systems for RDF	137
...	...	...
Jorge Perez	Semantics and Complexity of SPARQL	320
Jorge Perez	Minimal deductive systems for RDF	137
Jorge Perez	The recovery of a schema mapping	66
...	...	...
Renzo Angles	Survey of graph database models	315
Renzo Angles	The expressive power of SPARQL	157
Renzo Angles	Current graph database models	20
...	...	...

The table is a large tab-separated values (TSV) file contains millions of records about authors, their papers, and the citations of their papers. Multiple authors may write a single paper (as seen above). Paper titles and author names can be assumed to be unique.

From this table, you wish to compute a new table with pairs of co-authors and the sum of the number of citations of those papers they have co-authored together. Based on the partial data input above, the result would look like the following (avoiding duplicates by ensuring that Author 1 is alphabetically lower than Author 2):

Author 1	Author 2	Citations
Claudio Gutierrez	Jorge Perez	457
Claudio Gutierrez	Renzo Angles	472
...	...	...

You then wish to sort the results in descending order by total citations.

Given this input and desired output, design a series of MapReduce jobs to perform the required processing. In particular, detail the sequence of map/reduce phases of your algorithm: what are the map keys, what are the map values, what are the reduce keys, what are the reduce values, what does the map function do, what does the reduce function do. Also indicate if there is a possibility to use a combiner at each step. You can use natural language, diagrams, examples AND/OR pseudo-code to describe the algorithm, as you prefer (so long as it is readable).

Q.2

You are working for a large supermarket chain. They are interested in how much money their customers spend, on average, at different hours of the day. They give you three large tab-separated values (TSV) files containing millions of records as follows:

1: ReceiptItems.tsv

Receipt ID	Item ID
R1401	I306
R1401	I306
R1401	I504
R1402	I007
R1402	I306
R1403	I306
R1403	I504
...	...

2: ReceiptTimes.tsv

Receipt ID	Time
R1403	19:00
R1401	18:59
R1402	19:01
...	...

3: ItemDetails.tsv

Item ID	Name	Price ($)
I306	Zanahoria 500g	500
I504	CocaCola 3L	1400
I007	Comfort	1200
...	...	

In these tables, the Receipt ID column corresponds to an individual transaction, where a customer pays for their items. The Item ID corresponds to a unique identifier for each type of item. The same item may appear multiple times in a transaction. So in the table above, in transaction R1401, a customer buys 2 × Zanahoria 500g ($500) and 1 × CocaCola 3L ($1400), spending a total of $2400 at time 18:59. Likewise transaction R1402 spends $1700 at time 19:00 and transaction R1403 spends $1900 at time 19:01.

Given this input, your manager wants you to compute the total spent by customers of the supermarket chain each hour of the day. For example, just considering the three transactions above, the answer would be:

Output

Hour	Total
...	...
18:00–18:59	$2400
19:00–19:59	$3600
...	...

The output should then be sorted in descending order by total value.

Given this input and desired output, design a MapReduce process (consisting of multiple jobs) to complete the required processing. In particular, detail the sequence of map/reduce jobs of your algorithm: what are the map keys, what are the map values, what are the reduce keys, what are the reduce values, what does the map function do, what does the reduce function do. Also indicate if there is a possibility to use a combiner. Give an example of the output you would expect for any intermediate map–reduce phase(s). You can use natural language, diagrams AND/OR pseudo-code to describe the algorithm, as you prefer (so long as it is readable).