

CS 378 – Big Data Programming

Lecture 15

Join Patterns

Join Patterns

- It is almost always the case that our “big data” is coming from multiple sources
 - Web logs (of different types)
 - Databases (RDBMS, column-oriented DB, NoSQL, ...)
 - Key/value store (redis, ...)
 - ...
- And we need to combine/integrate this data for
 - Reporting, analysis, BI, ETL, ...

Join Patterns

- We're familiar with "join" in RDBMS (using SQL)
- A join combines data from two or more data sets
 - Based on a field or set of fields – the *foreign key*
 - In RDBMS, the *foreign key* field matches values in the column of another table
 - Effectively a cross-reference between two or more tables

Join Example

Tables 5-1, 5-2 from MapReduce Design Patterns

Table 5-1. Table A

User ID	Reputation	Location
3	3738	New York, NY
4	12946	New York, NY
5	17556	San Diego, CA
9	3443	Oakland, CA

Table 5-2. Table B

User ID	Post ID	Text
3	35314	Not sure why this is getting downvoted.
3	48002	Hehe, of course, it's all true!
5	44921	Please see my post below.
5	44920	Thank you very much for your reply.
8	48675	HTML is not a subset of XML!

Join Example – Inner Join

Table 5-3 from MapReduce Design Patterns

A.User ID	A.Reputation	A.Location	B.User ID	B.Post ID	B.Text
3	3738	New York, NY	3	35314	Not sure why this is getting downvoted.
3	3738	New York, NY	3	48002	Hehe, of course, it's all true!
5	17556	San Diego, CA	5	44921	Please see my post below.
5	17556	San Diego, CA	5	44920	Thank you very much for your reply.

Join Example – Left Outer Join

Table 5-4 from MapReduce Design Patterns

A.User ID	A.Reputation	A.Location	B.User ID	B.Post ID	B.Text
3	3738	New York, NY	3	35314	Not sure why this is getting downvoted.
3	3738	New York, NY	3	48002	Hehe, of course, it's all true!
4	12946	New York, NY	null	null	null
5	17556	San Diego, CA	5	44921	Please see my post below.
5	17556	San Diego, CA	5	44920	Thank you very much for your reply.
9	3443	Oakland, CA	null	null	null

Reduce Side Join

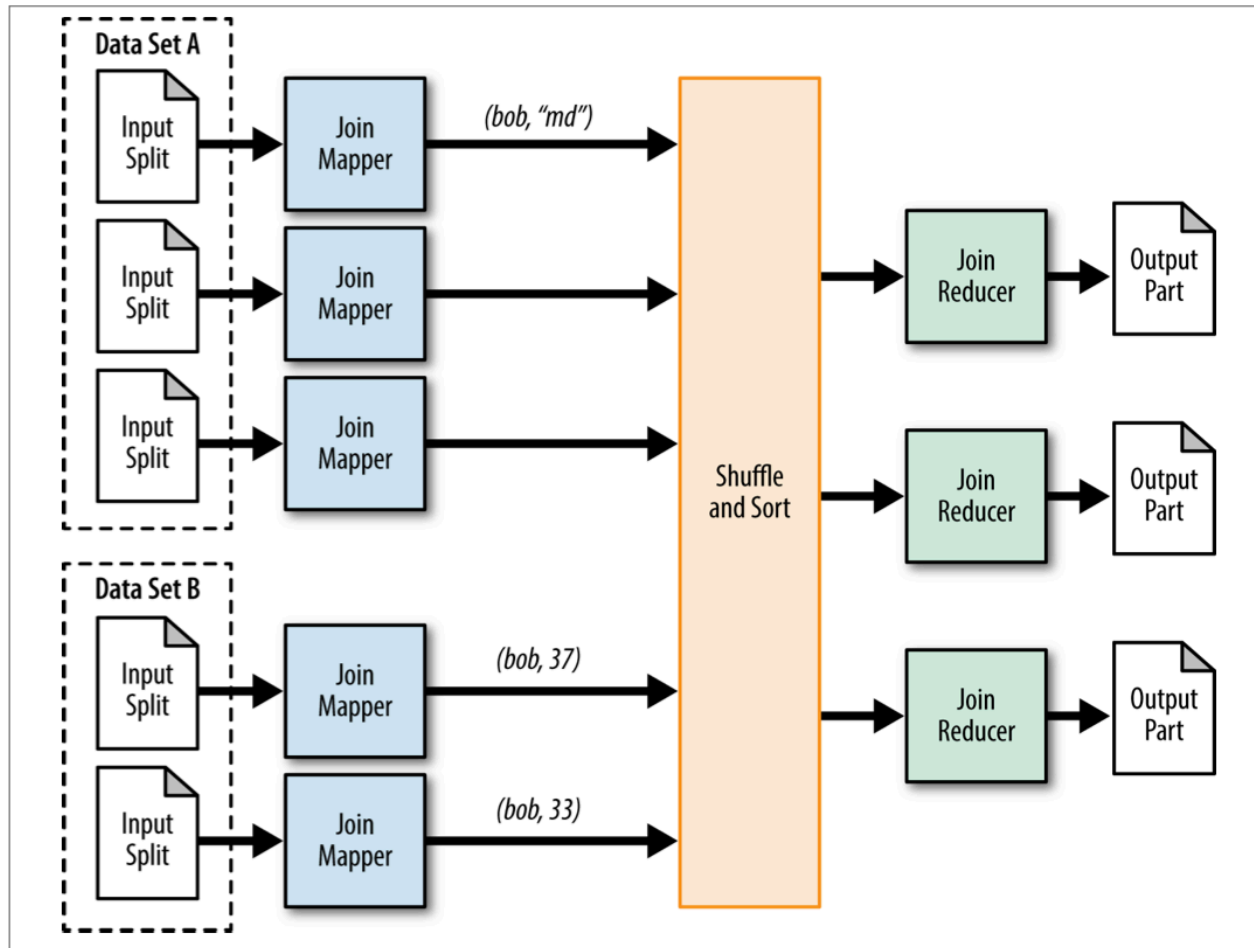
- The simplest implementation of mapReduce join is the *reduce side join*
- Reduce side join can accomplish any of the joins we discussed
 - Inner join, full outer join, left outer join, right outer join
 - Anti-join, full Cartesian product
- Requires that all data be sent over the network to reducers
- Can join as many data sets as you need (needs common key)

Reduce Side Join - Steps

- Mapper reads its data source, extracts *foreign key*
 - Foreign key output as the key, input record as value
- A partitioner can be used
 - If you understand the key distribution
 - Want to equally distribute keys across reducers
- Reducer performs the join operation
 - Inner join, outer join,

Reduce Side Join - Data Flow

Figure 5-1 from MapReduce Design Patterns



Reduce Side Join - Example

- Join multiple sources of data
 - User sessions
 - Other impression data
- Key is VIN (vehicle identification number)
 - Combine VIN impression data from multiple sources
- Join output is VIN impression stats with
 - Stats from user sessions
 - Stats from another source

MultipleInputs

- The `MultipleInputs` class allows different mappers to handle different input data
- One mapper will read user sessions
- Another mapper will read impression counts
- We'll do this in assignment 7