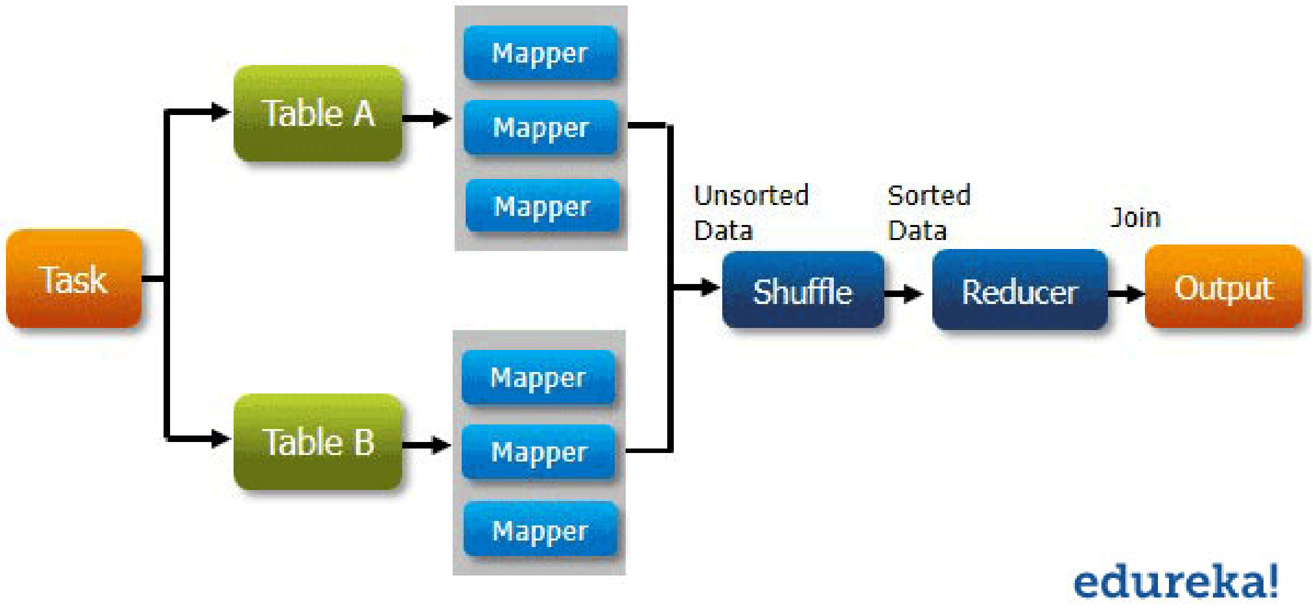
Map Side Join Vs. Join

Nov 26, 2013 | Priyanka (http... | 27,009 | 11 Comments



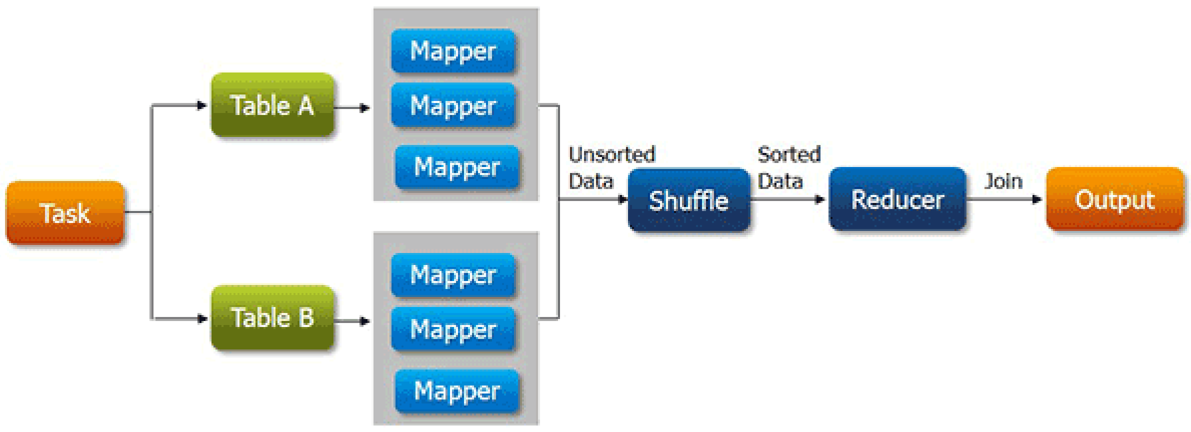
(http://www.edureka.co/big-data-and-hadoop?utm_source=blog&utm_medium=related-posts&utm_campaign=bigdata-hadoop)

In this blog we shall discuss about **Map side join** and its advantages over the normal join operation in **Hive**. But before knowing about this, we should first understand the concept of **'Join'** and what happens internally when we perform the join in **Hive**.

Join is a clause that combines the records of two tables (or Data-Sets).
Assume that we have two tables A and B. When we perform join operation on them, it will return the records which are the combination of all columns o f A and B.

Now let us understand the functionality of normal join with an example..

Whenever, we apply join operation, the job will be assigned to a Map Reduce task which consists of two stages- a ***'Map stage'*** and a '***Reduce stage***'. A mapper's job during Map Stage is to *"read"* the data from join tables and to *"return"* the **'join key'** and **'join value'** pair into an intermediate file. Further, in the shuffle stage, this intermediate file is then sorted and merged. The reducer's job during reduce stage is to take this sorted result as input and complete the task of join.



(http://cdn.edureka.co/blog/wp-content/uploads/2013/11/joins1.png)

- Map-side Join is similar to a join but all the task will be performed by the mapper alone.
- The Map-side Join will be mostly suitable for small tables to optimize the task.

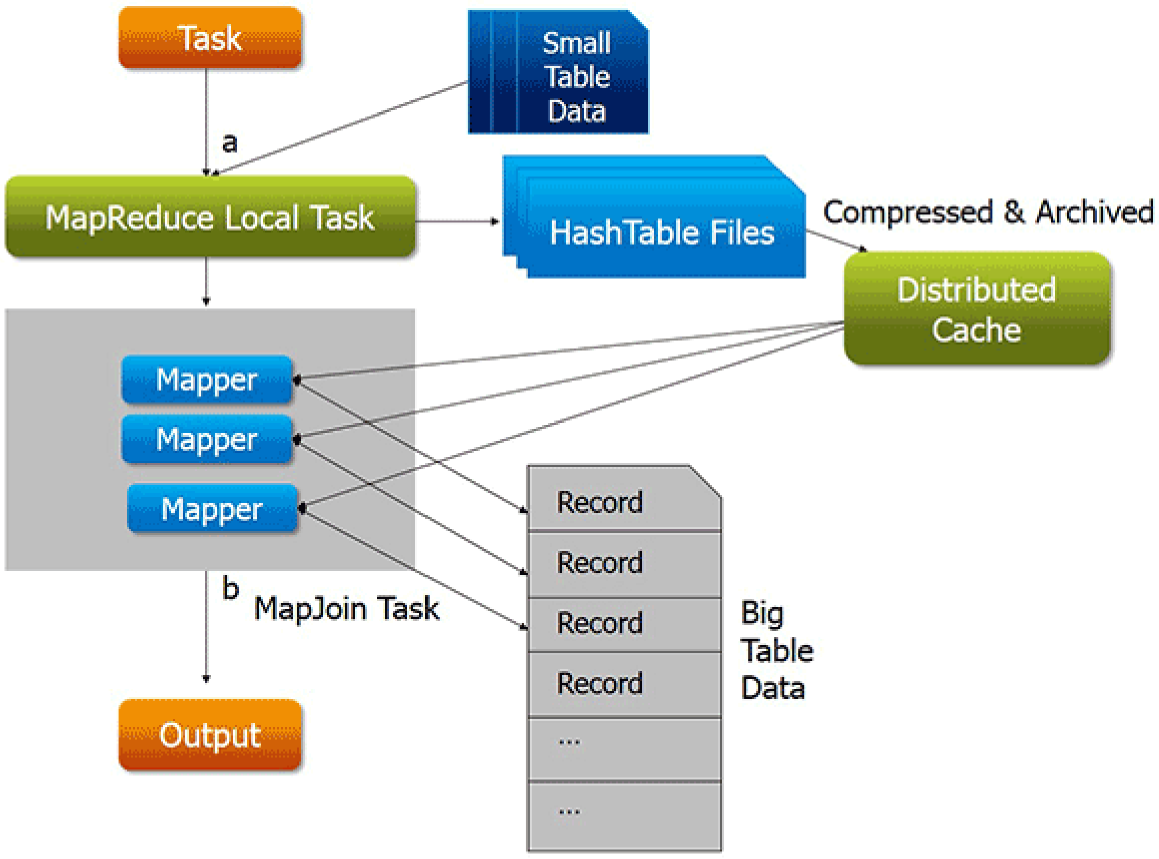
Master Map-Side Join

(http://www.edureka.co/big-data-and-hadoop?utm_source=blog&utm_medium=button&utm_campaign=map-side-join-vs-join-master-map-side-join)

How will the map-side join optimize the task?

Assume that we have two tables of which one of them is a small table. When we submit a map reduce task, a Map Reduce local task will be created before the original join Map Reduce task which will read data of the small table from HDFS and store it into an in-memory hash table. After reading, it serializes the in-memory hash table into a hash table file.

In the next stage, when the original join Map Reduce task is running, it moves the data in the hash table file to the Hadoop distributed cache, which populates these files to each mapper's local disk. So all the mappers can load this persistent hash table file back into the memory and do the join work as before. The execution flow of the optimized map join is shown in the figure below. After optimization, the small table needs to be read just once. Also if multiple mappers are running on the same machine, the distributed cache only needs to push one copy of the hash table file to this machine.



(http://cdn.edureka.co/blog/wp-content/uploads/2013/11/joins2.png)

Advantages of using map side join:

- Map-side join helps in minimizing the cost that is incurred for sorting and merging in the *shuffle* and *reduce* stages.
- Map-side join also helps in improving the performance of the task by decreasing the time to finish the task.

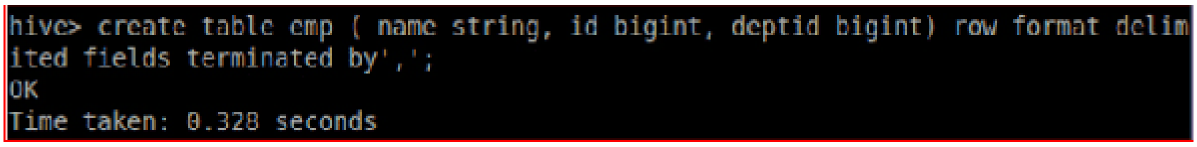
Disadvantages of Map-side join:

- Map side join is adequate only when one of the tables on which you perform map-side join operation is small enough to fit into the memory. Hence it is not suitable to perform map-side join on the tables which are huge data in both of them.

Simple Example for Map Reduce Joins:

Let us create two tables:

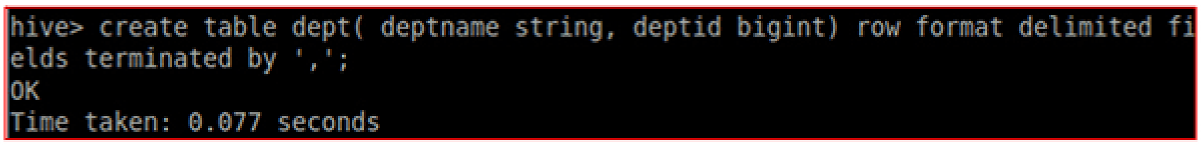
- **Emp**: contains details of an Employee such as Employee name, Employee ID and the Department she belongs to.



```
hive> create table emp ( name string, id bigint, deptid bigint) row format delim
ited fields terminated by',';
OK
Time taken: 0.328 seconds
```

(http://cdn.edureka.co/blog/wp-content/uploads/2013/11/Untitled-11.png)

- **Dept:** contains the details like the Name of the Department, Department ID and so on.

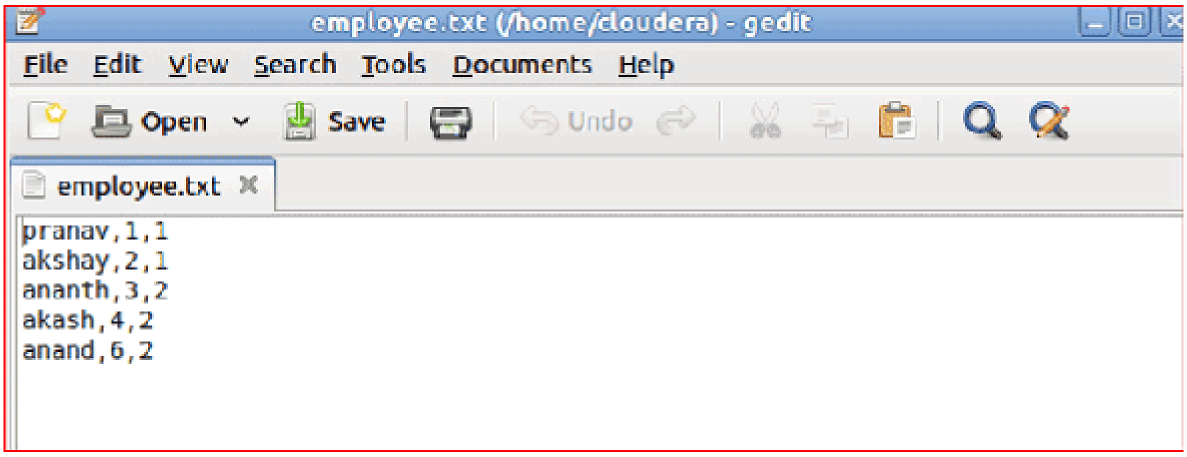


```
hive> create table dept( deptname string, deptid bigint) row format delimited fi
elds terminated by ',';
OK
Time taken: 0.077 seconds
```

(http://cdn.edureka.co/blog/wp-content/uploads/2013/11/Creating-table-using-Mapreduce-joins.png)

Create two input files as shown in the following image to load the data into the tables created.

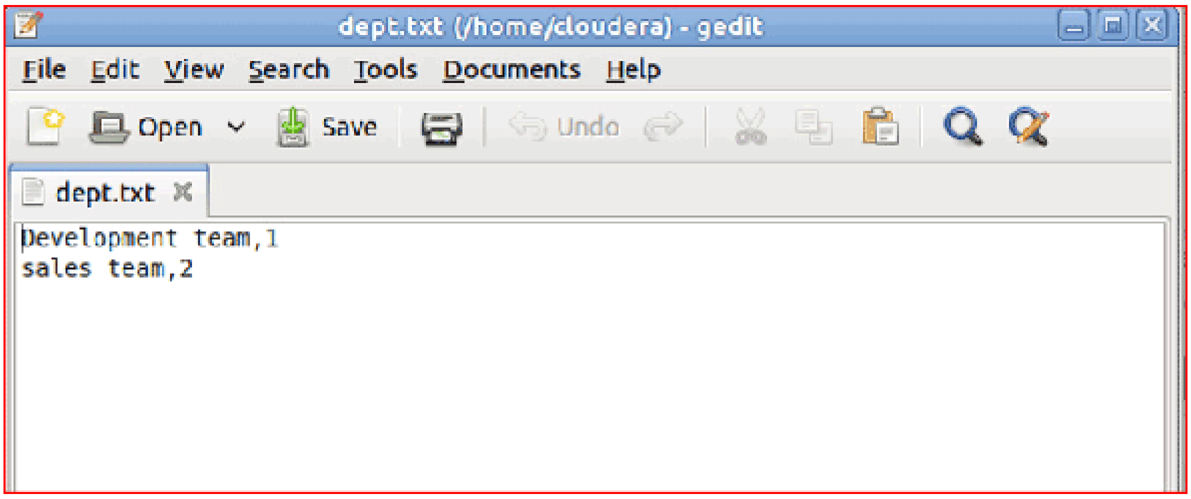
employee.txt


employee.txt (/home/cloudera) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
employee.txt

```
pranav,1,1
akshay,2,1
ananth,3,2
akash,4,2
anand,6,2
```

(http://cdn.edureka.co/blog/wp-content/uploads/2013/11/Untitled-21.png)

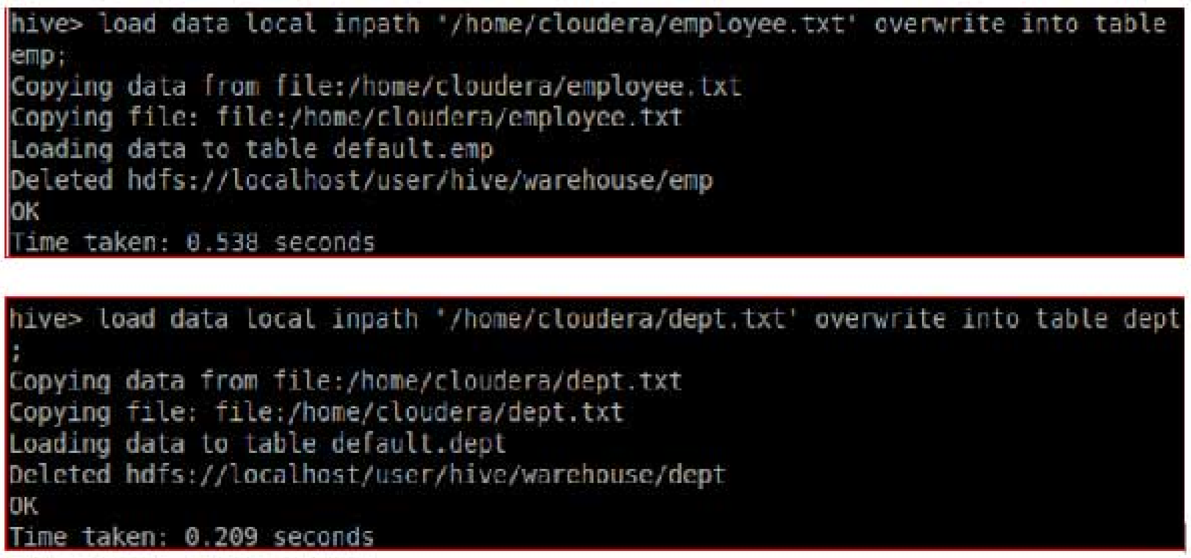
dept.txt


dept.txt (/home/cloudera) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
dept.txt

```
Development team,1
sales team,2
```

(http://cdn.edureka.co/blog/wp-content/uploads/2013/11/Untitled-31.png)

Now, let us load the data into the tables.



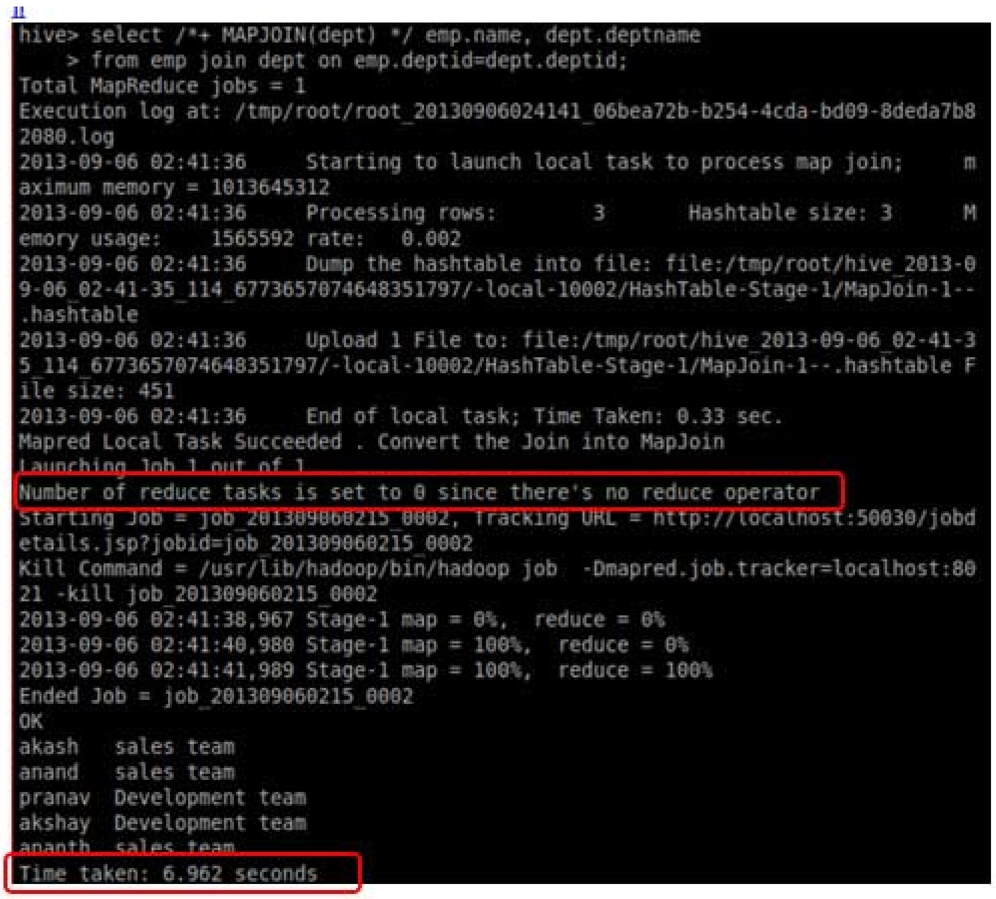
```
hive> load data local inpath '/home/cloudera/employee.txt' overwrite into table
emp;
Copying data from file:/home/cloudera/employee.txt
Copying file: file:/home/cloudera/employee.txt
Loading data to table default.emp
Deleted hdfs://localhost/user/hive/warehouse/emp
OK
Time taken: 0.538 seconds
```

```
hive> load data local inpath '/home/cloudera/dept.txt' overwrite into table dept
;
Copying data from file:/home/cloudera/dept.txt
Copying file: file:/home/cloudera/dept.txt
Loading data to table default.dept
Deleted hdfs://localhost/user/hive/warehouse/dept
OK
Time taken: 0.209 seconds
```

(http://cdn.edureka.co/blog/wp-content/uploads/2013/11/Untitled-51.jpg)

Let us perform the **Map-side Join** on the two tables to extract the list of departments in which each employee is working.

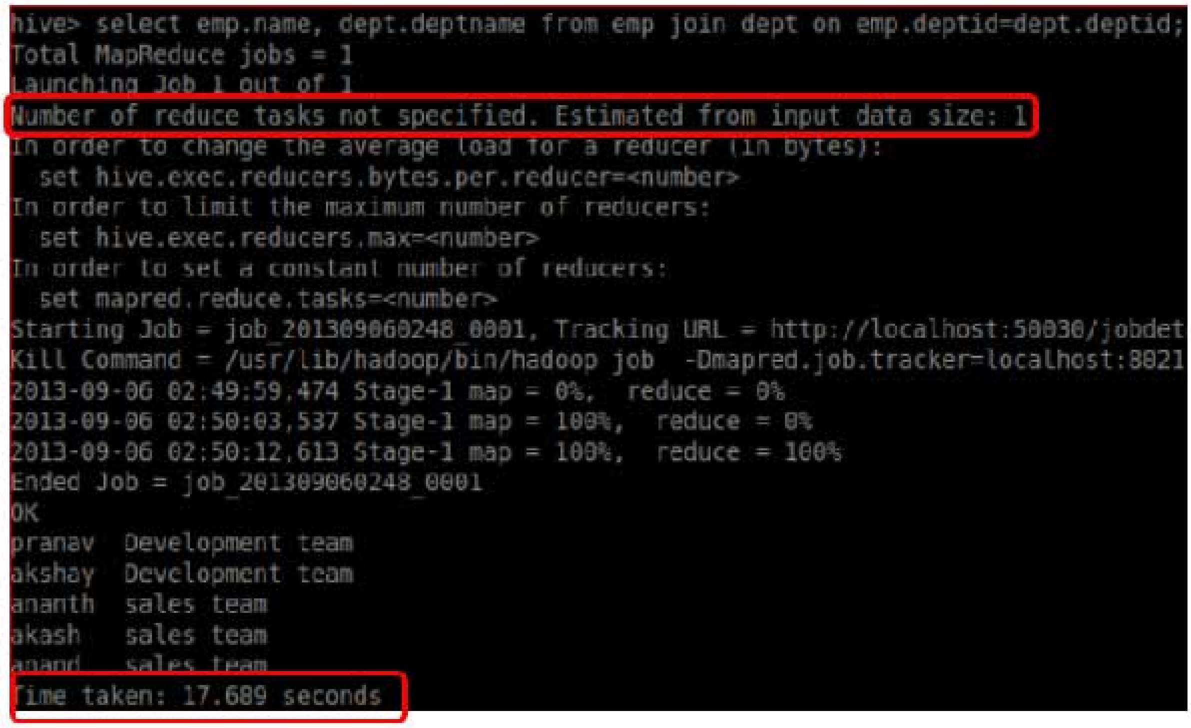
Here, the **second table dept** is a small table. Remember, always the number of department will be less than the number of employees in an organization.



```
hive> select /*+ MAPJOIN(dept) */ emp.name, dept.deptname
    > from emp join dept on emp.deptid=dept.deptid;
Total MapReduce jobs = 1
Execution log at: /tmp/root/root_20130906024141_06bea72b-b254-4cda-bd09-8deda7b8
2080.log
2013-09-06 02:41:36     Starting to launch local task to process map join;     m
aximum memory = 1013645312
2013-09-06 02:41:36     Processing rows:        3       Hashtable size: 3      M
emory usage:    1565592 rate:   0.002
2013-09-06 02:41:36     Dump the hashtable into file: file:/tmp/root/hive_2013-0
9-06_02-41-35_114_6773657074648351797/-local-10002/HashTable-Stage-1/MapJoin-1--
.hashtable
2013-09-06 02:41:36     Upload 1 File to: file:/tmp/root/hive_2013-09-06_02-41-3
5_114_6773657074648351797/-local-10002/HashTable-Stage-1/MapJoin-1--.hashtable F
ile size: 451
2013-09-06 02:41:36     End of local task; Time Taken: 0.33 sec.
Mapred Local Task Succeeded . Convert the Join into MapJoin
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_201309060215_0002, Tracking URL = http://localhost:50030/jobd
etails.jsp?jobid=job_201309060215_0002
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:80
21 -kill job_201309060215_0002
2013-09-06 02:41:38,967 Stage-1 map = 0%,  reduce = 0%
2013-09-06 02:41:40,980 Stage-1 map = 100%,  reduce = 0%
2013-09-06 02:41:41,989 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_201309060215_0002
OK
akash   sales team
anand   sales team
pranav  Development team
akshay  Development team
ananth  sales team
Time taken: 6.962 seconds
```

(http://cdn.edureka.co/blog/wp-content/uploads/2013/11/Untitled-101.jpg)

Now let's perform the same task with the help of normal Reduce-side join.



```
hive> select emp.name, dept.deptname from emp join dept on emp.deptid=dept.deptid;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_201309060248_0001, Tracking URL = http://localhost:50030/jobdet
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:8021
2013-09-06 02:49:59,474 Stage-1 map = 0%,  reduce = 0%
2013-09-06 02:50:03,537 Stage-1 map = 100%,  reduce = 0%
2013-09-06 02:50:12,613 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_201309060248_0001
OK
pranav  Development team
akshay  Development team
ananth  sales team
akash   sales team
anand   sales team
Time taken: 17.689 seconds
```

(http://cdn.edureka.co/blog/wp-content/uploads/2013/11/Untitled-81.jpg)

While executing both the joins, you can find the two differences:

- Map-reduce join has completed the job in less time when compared with the time taken in normal join.
- Map-reduce join has completed its job without the help of any reducer whereas normal join executed this job with the help of one reducer.

Hence, **Map-side Join** is your best bet when one of the tables is small enough to fit in memory to complete the job in a short span of time.

In **Real-time environment**, you will be have data-sets with huge amount of data. So performing analysis and retrieving the data will be time consuming if one of the data-sets is of a smaller size. In such cases *Map-side join* will help to complete the job in less time.

There has never been a better time to master Hadoop! Get started now with the specially curated Big Data and Hadoop course by Edureka.

Learn Hadoop from Experts

(http://www.edureka.co/big-data-and-hadoop?utm_source=blog&utm_medium=button&utm_campaign=map-side-join-vs-join-learn-hadoop-from-experts)

References:
https://www.facebook.com/notes/facebook-engineering/join-optimization-in-apache-hive/470667928919 (https://www.facebook.com/notes/facebook-engineering/join-optimization-in-apache-hive/470667928919)

Related Posts:

7 Ways Big Data Training Can Change Your Organization (http://www.edureka.co/blog/7-ways-big-data-training-can-change-your-organization/?utm_source=blog&utm_medium=related-posts&utm_campaign=mapside-vs-join)

10 Reasons Why Big Data Analytics is the Best Career Move (http://www.edureka.co/blog/10-reasons-why-big-data-analytics-is-the-best-career-move?utm_source=blog&utm_medium=related-posts&utm_campaign=mapside-vs-join)

Get started with Big Data and Hadoop (http://www.edureka.co/big-data-and-hadoop?utm_source=blog&utm_medium=related-posts&utm_campaign=bigdata-hadoop)

Get started with Comprehensive MapReduce (http://www.edureka.co/comprehensive-mapreduce-self-paced?utm_source=blog&utm_medium=related-posts&utm_campaign=mapside-vs-join)

Get started with MapReduce Design Patterns (http://www.edureka.co/mapreduce-design-patterns?utm_source=blog&utm_medium=related-posts&utm_campaign=mapside-vs-join)

Introduction to Apache Mapreduce & HDFS (http://www.edureka.co/blog/introduction-to-apache-hadoop-hdfs/?utm_source=blog&utm_medium=related-posts&utm_campaign=mapside-vs-join)

About Priyanka (9 Posts (http://www.edureka.co/blog/author/priyanka/))

(https://plus.google.com/)

Comments

11 Comments

11 Comments **http://www.edureka.co/blog/** 1 **Login**

Recommend 1 **Share** **Sort by Best**

Join the discussion…

siddesh samarth • 2 years ago

Nice one

Reply • Share ›

EdurekaSupport Mod siddesh samarth • 2 years ago

Thanks Siddesh!!

Reply • Share ›

Mahanth Kalaga • 2 years ago

Informative and clearly explained.

Can you please post an article on Reduce side joins as well.

Thanks.

Reply • Share ›

EdurekaSupport Mod Mahanth Kalaga • 2 years ago

Thanks Mahanth!! We will consider your suggestion.

Reply • Share ›

Sushobhit Rajan • 2 years ago

Great Job. nicely explained and example helps a lot. Even a small explain can do a big thing.

Thanks ..looking for more blogs like this, with examples.

Reply • Share ›

Santhosh • 2 years ago

Informative Article! Thanks a ton!

Reply • Share ›

EdurekaSupport Mod Santhosh • 2 years ago

You are welcome, Santhosh!! Feel free to browse through our other blog posts as well.

Reply • Share ›

Amit • 2 years ago

Thanks, Nice Description

Reply • Share ›

EdurekaSupport Mod Amit • 2 years ago

Thanks Amit!! Feel free to go through our other blog posts as well.

Reply • Share ›

Gopi raju • 2 years ago

Explained very clearly. Thanks for your post.........

Reply • Share ›

EdurekaSupport Mod Gopi raju • 2 years ago

Thanks Gopi! Feel free to go through our other posts as well.

Reply • Share ›

ALSO ON **HTTP://WWW.EDUREKA.CO/BLOG/**

Top Node.js Interview Questions You Must **Unicorn 2.0 has Released!**

Trending (/blog/trending) View all

10 Reasons Why Big Data Analytics is the Best Career Move (http://www.edureka.co/blog/10-reasons-why-big-data-analytics-is-the-best-career-move)

Hive Data Models (http://www.edureka.co/blog/hive-data-models/)

The Beginner's Guide to Android: Android Architecture (http://www.edureka.co/blog/beginners-guide-android-architecture/)

Demystifying Partitioning in Spark (http://www.edureka.co/blog/demystifying-partitioning-in-spark)

Apache Spark vs Hadoop MapReduce (http://www.edureka.co/blog/apache-spark-vs-hadoop-mapreduce)

Subscribe

Signup for Edureka blog updates

Enter Email for subscription

Featured Posts

(http://www.edureka.co/blog/cca-and-ccp-certifications-by-cloudera-all-you-need-to-know/)
CCA and CCP Certifications By Cloudera: All...
(http://www.edureka.co/blog/cca-and-ccp-certifications-by-cloudera-all-you-need-to-know/) Jul 22, 2016

(http://www.edureka.co/blog/top-reasons-to-learn-banking-analytics)
Top Reasons To Learn Banking Analytics in 2...
(http://www.edureka.co/blog/top-reasons-to-learn-banking-analytics)

Jun 6, 2016

(http://www.edureka.co/blog/apache-cassandra-career-opportunities)
Apache Cassandra Career Opportunities: How ...
(http://www.edureka.co/blog/apache-cassandra-career-opportunities)

May 31, 2016