**Problem Statement:**

**Explain the following in brief with an example.**

**● Map side Join**

**● Reduce side Join**

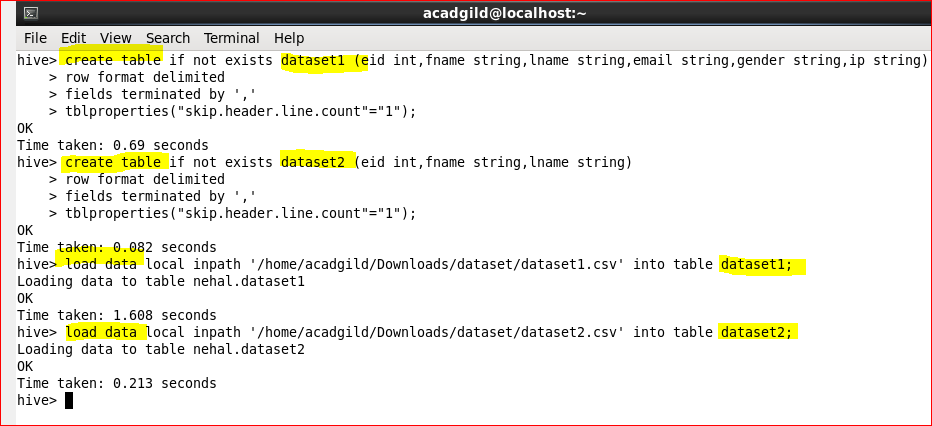
**● Bucket Map Join**

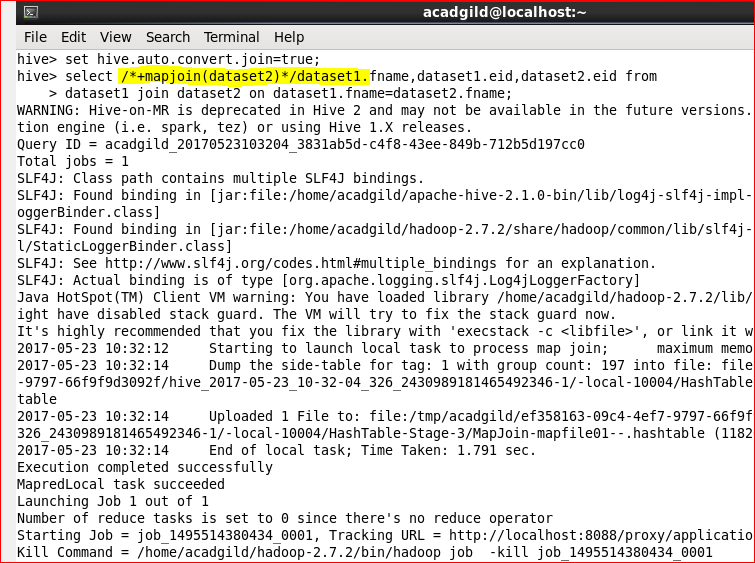
**● SMBM Join**

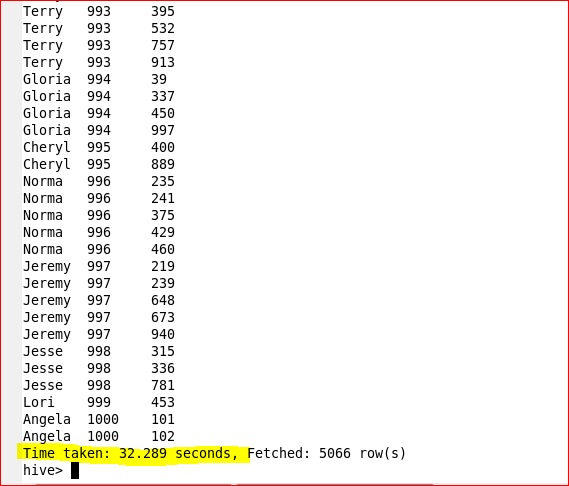
**1. Map Side Join:**

**Also known as replicated join, a map-side join is a special type of join where a smaller table is loaded in memory and join is performed in map phase of MapReduce job. Since there is no reducer involved in the map-side join, it is much faster when compared to regular join.**

**An important point to note is, one table must be small enough to fit into memory. It is recommended to have a proper configuration so that Hive automatically attempt to convert Joins into the map-side join. Below is a Hive join operation which is not a map-side join.**

****

****

****

**In the image above, note the highlighted part. You can see that “number of reducer” is 1 which slows down the join operation.**

**Now, to perform map-side join, set few configurations either into *hive-site.xml* OR directly from Hive shell. Below are the configurations which I have set from Hive shell.**

**hive> set hive.auto.convert.join=true;**

**hive> set hive.auto.convert.join.noconditionaltask=true;**

**Once you are done with the configuration, execute the same join operation as we performed above.**

**Note the highlighted part again! You will find that there is no reducer phase performed in this join operation. Hence, the map-side join is faster than regular join operation.**

**2. Reduce-Side Joins:-**

**Reduce-Side joins are simpler than Map-Side joins since the input datasets need not to be structured. But it is less efficient as both datasets have  
to go through the MapReduce shuffle phase. the records with the same key are brought together in the reducer. We can also use the Secondary Sort technique  
to control the order of the records.**

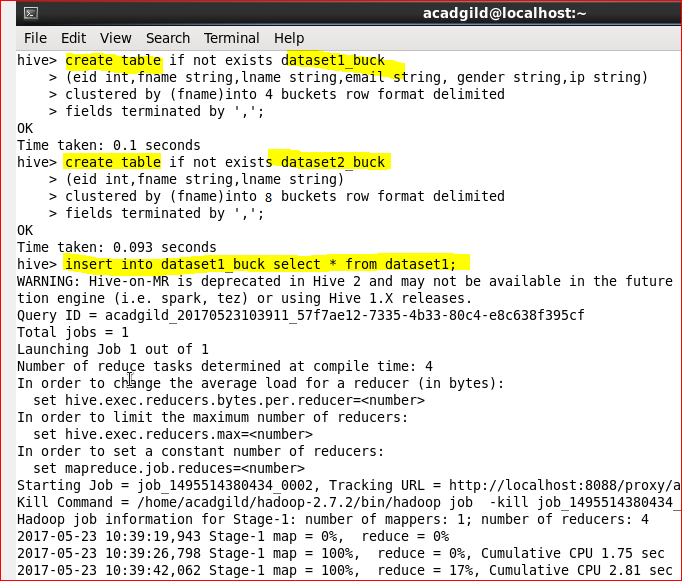
**3. Bucket-Map join:-**

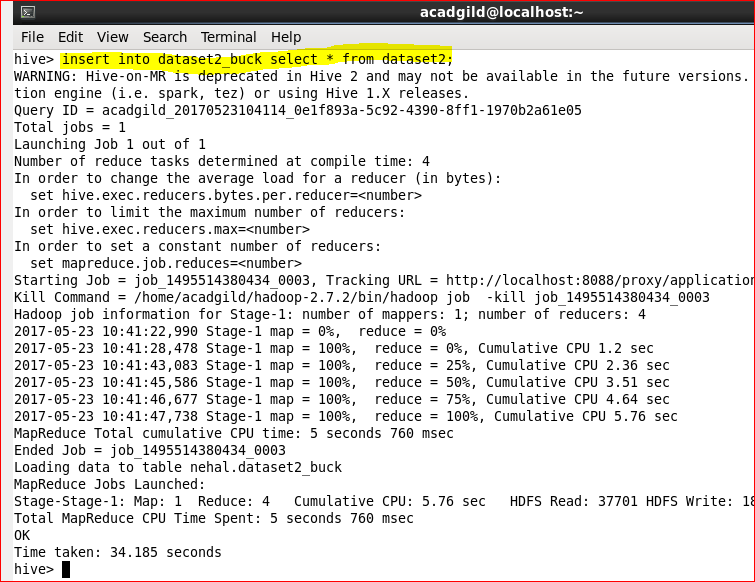
**The constraint for performing Bucket-Map join is:**

**If tables being joined are bucketed on the join columns, and the number of buckets in one table is a multiple of the number of buckets in the other table, the buckets can be joined with each other.**

**To perform bucketing, we need to have bucketed tables. Let’s create them.**

|  |
| --- |
|  |
|  |

****

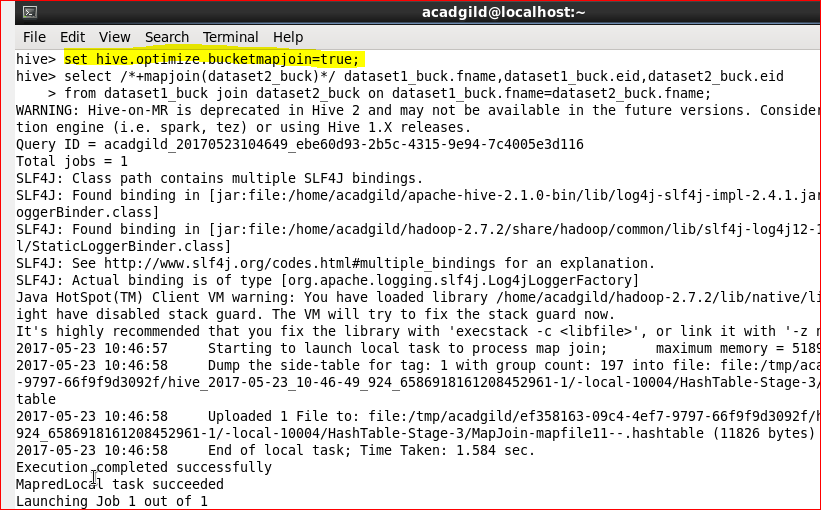
****

**Here, we have two tables that are bucketed. We can now perform Bucket-map join between these two datasets.**

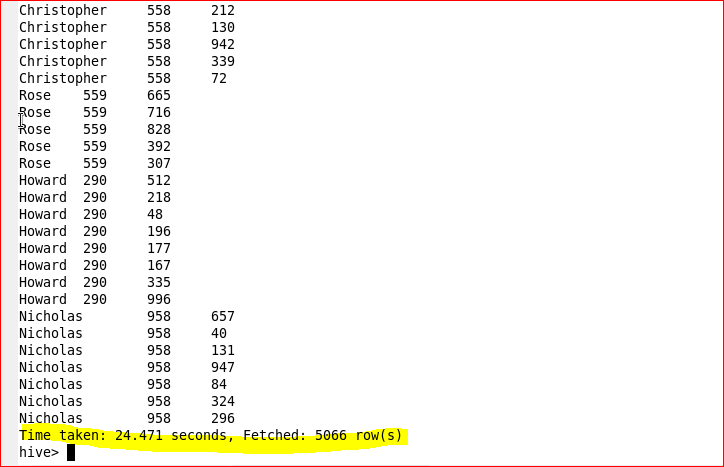
**Here, for the first table we have created 4 buckets and for the second table we have created 8 buckets on the same column. Now, we can perform Bucket-map join on these two tables.>**

**For performing Bucket-Map join, we need to set this property in the Hive shell.**

**set hive.optimize.bucketmapjoin = true**

****

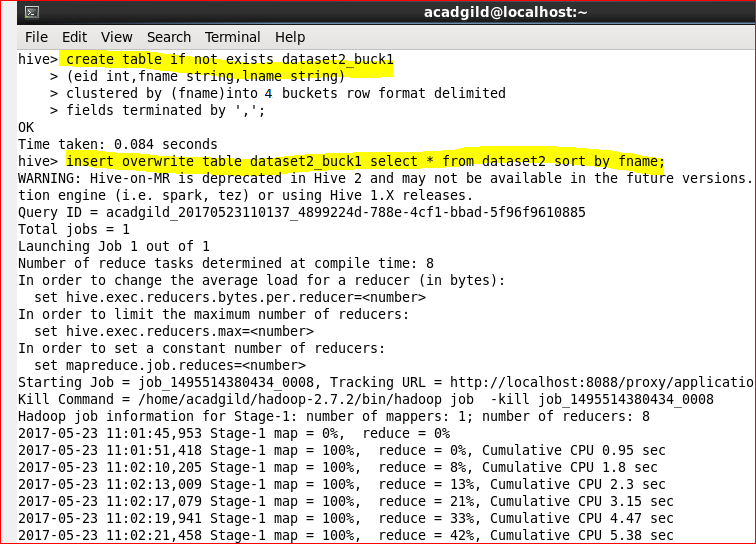
**This query took 24.471 seconds to complete. You can see the same in the screenshot shown below:**

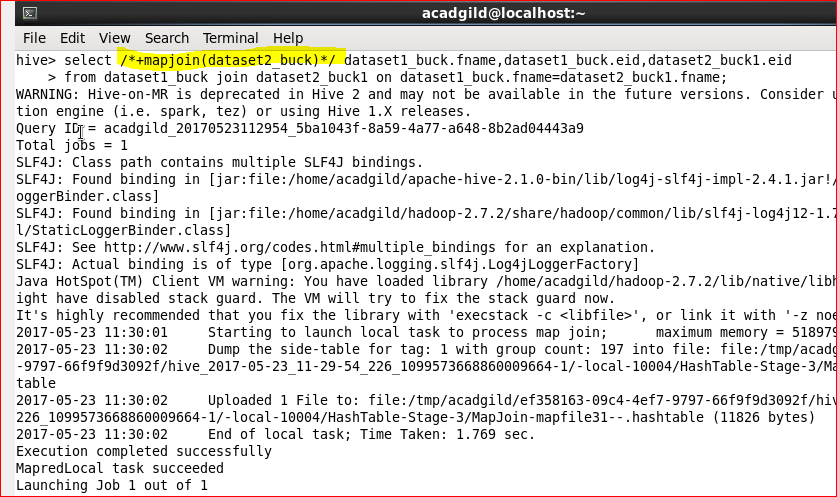
****

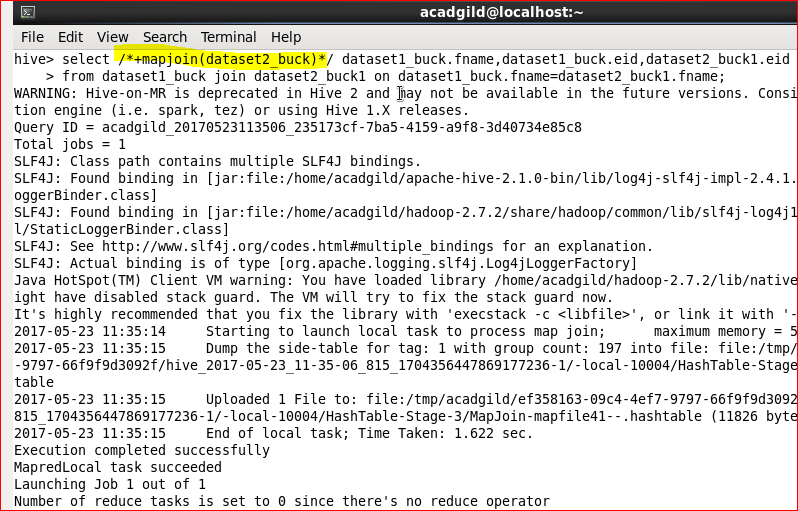
**4. SMBM Join:**

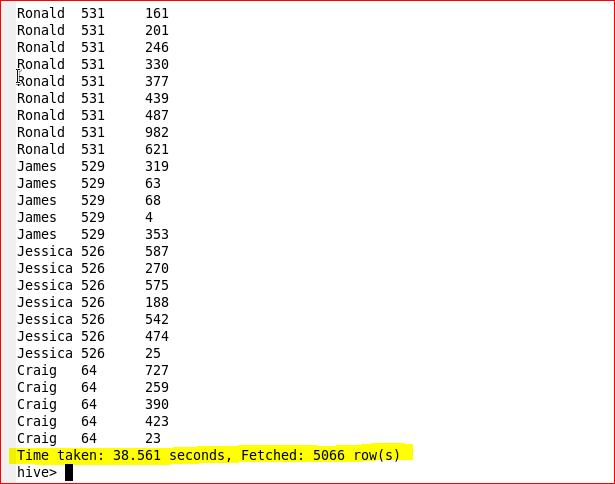
**It is another Hive join optimization technique where all the tables need to be bucketed and sorted. In this case joins are very efficient because they require a simple merge of the presorted tables.**

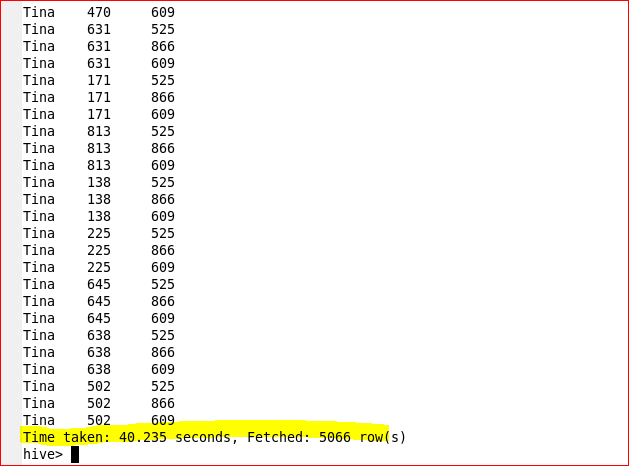
**Let us create bucketed tables from our existing tables i.e.; emp and dept. Before creating bucketed table, you need to set below properties.**

****

****

**Now the stage is set to perform SMB Map Join to optimize Hive joining. Again, make some changes in properties to perform SMB Map join.**

****

****