**Why do hive does not run mapreduce job while performing select \* operation?**

There are three types of operations that a hive query can perform.

A hive query can be a metadata only request. Ex: - Show tables, describe table are examples

A hive query can be an hdfs get request. Ex: - Select \* from table In this case hive can return the results by performing an hdfs operation. hadoop fs -get, more or less.

A hive query can be a Map Reduce job. Ex: - Select \* from table where id > 100

**What will be the output of partitioning and bucketing?**

Partitioning creates FOLDERS whereas Bucketing Creates FILES.

**Save the result of a hive query to HDFS**

insert overwrite directory 'sample' row format delimited fields terminated by ',' select \* from sample\_data;

**Icon Integration in CENTOS**

Create a ".desktop" file under ".local/share/applications/" with following details

Ex: eclipse-scala.desktop

#Version=1.0

Name=Eclipse-Scala

GenericName=Integrated Development Environment

Comment=Eclipse

Exec=/home/cloudera/eclipse\_scala/eclipse

Icon=/home/cloudera/eclipse\_scala/icon.xpm

Terminal=false

Type=Application

Categories=Development;IDE;

Move that file to "/usr/share/applications/"

[**Hive order by vs sort by**](https://stackoverflow.com/questions/13715044/hive-cluster-by-vs-order-by-vs-sort-by) **vs distributed by vs cluster by**

* ORDER BY x: guarantees global ordering, but does this by pushing all data through just one reducer. This is basically unacceptable for large datasets. You end up one sorted file as output.
* SORT BY x: orders data at each of N reducers, but each reducer can receive overlapping ranges of data. You end up with N or more sorted files with overlapping ranges.
* DISTRIBUTE BY x: ensures each of N reducers gets non-overlapping ranges of x, but doesn't sort the output of each reducer. You end up with N or unsorted files with non-overlapping ranges.
* CLUSTER BY x: ensures each of N reducers gets non-overlapping ranges, then sorts by those ranges at the reducers. This gives you global ordering, and is the same as doing (DISTRIBUTE BY x and SORT BY x). You end up with N or more sorted files with non-overlapping ranges.

**Multiple Joins**

select customer\_fname, customer\_lname, product\_name from customers c join orders o on(c.customer\_id = o.order\_customer\_id) join order\_items oi on(o.order\_id = oi.order\_item\_id) join products p on(p.product\_id = oi.order\_item\_product\_id);

job history

localhost:1988

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

problem with "locate" command

sudo updatedb

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

java version is not geting updated even after restart

source /etc/profile

"source" command to effect the changes we have done in the system/file

**Creating Indexing in hive.**

Ref: - https://acadgild.com/blog/indexing-in-hive/

After creation of a normal table execute the following commands

CREATE INDEX olympic\_index ON TABLE olympic (age) AS 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler' WITH DEFERRED REBUILD;

ALTER INDEX olympic\_index on olympic REBUILD;

**Note: -** you can replace org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler with BITMAP