**Step 1. Data Preprocessing**

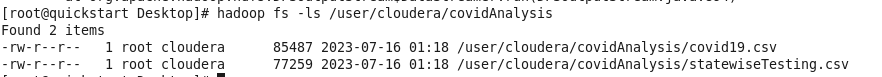
1.1 Create directories in hdfs to store the csv files

*hadoop fs -mkdir /user/cloudera/covidAnalysis*

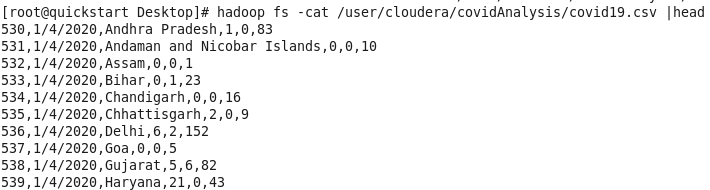
1.2 Copy the files from local to hdfs

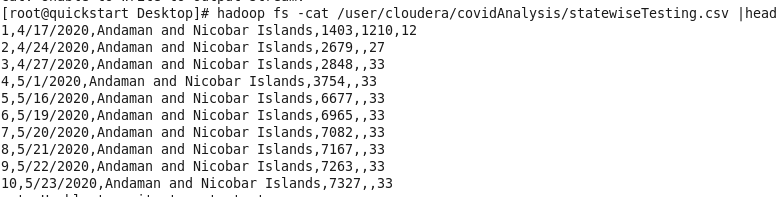
*hadoop fs -copyFromLocal covid/covid19.csv /user/cloudera/covidAnalysis*

*hadoop fs -copyFromLocal covid/statewiseTesting.csv /user/cloudera/covidAnalysis*

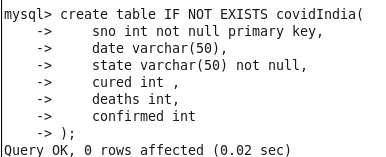
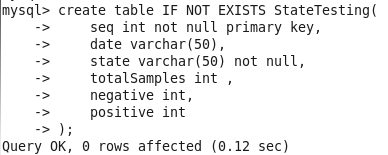


1.3 Visualize the content of the files in hdfs





1.4 Creation of Tables in Mysql



1.5 Sqoop Export of data from hdfs to Mysql

*sqoop export \*

*--connect jdbc:mysql://quickstart.cloudera:3306/test\_db \*

*--username root \*

*--password cloudera \*

*--table StateTesting \*

*--lines-terminated-by ',' \*

*--export-dir /user/cloudera/covidAnalysis/statewiseTesting.csv*



*sqoop export \*

*--connect jdbc:mysql://quickstart.cloudera:3306/test\_db \*

*--username root \*

*--password cloudera \*

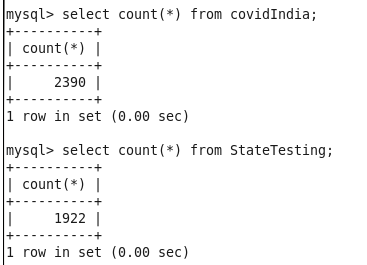
*--table covidIndia\*

*--lines-terminated-by ',' \*

*--export-dir /user/cloudera/covidAnalysis/covid19.csv*



1.6 Verify in Mysql



1.7 Delete data from hdfs

**Step 2- Sqoop Import**

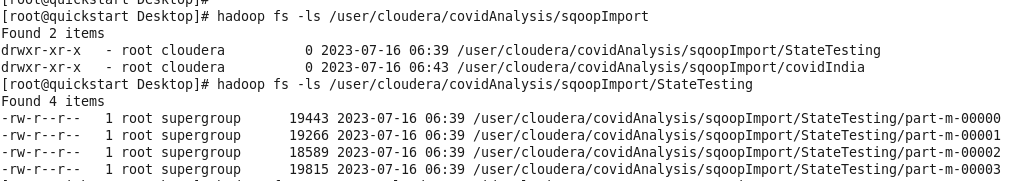
Command:



Command:

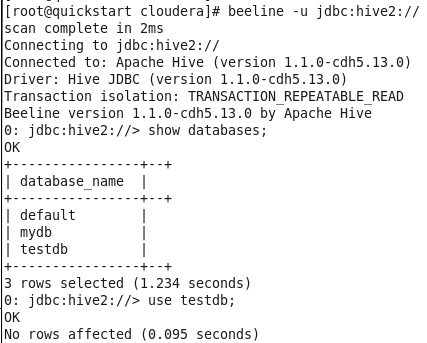


Output:



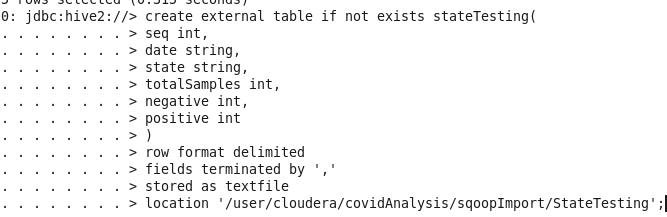
**Step 3- Create Hive External Tables on top of data in HDFS**

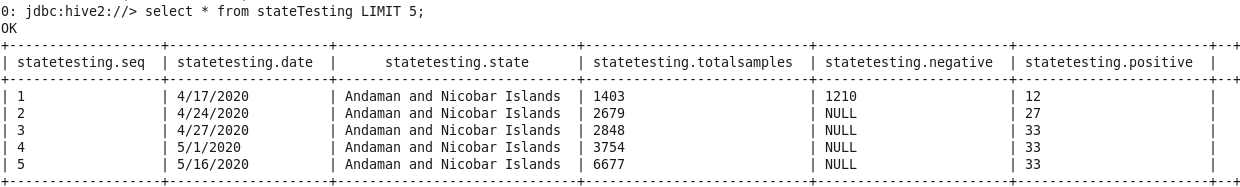
3.1 Connect to hive and select/create a database



3.2 Creation of Hive External Tables on top of data in HDFS

StateTesting table





Similarly, we do for covidIndia Table

**Step 4- Create Optimized External tables in Hive**

Optimizations Applied-

* File format used- ORC for Quicker and Efficient Reads
* Compression Codec used- Snappy for Fast Compression
* Partitioning on State Column
* Bucketing on Date Column

4.1 Create directories in hdfs for the Dynamically created Partitions:

*hadoop fs -mkdir /user/cloudera/covidAnalysis/partitions\_testing*

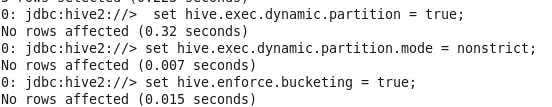
*hadoop fs -mkdir /user/cloudera/covidAnalysis/partitions\_covidIndia*

4.2 Enabling Dynamic Partitioning and Bucketing in Hive:

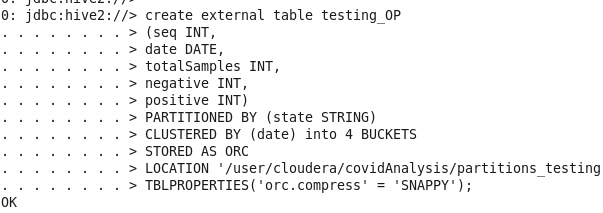
*set hive.exec.dynamic.partition = true;*

*set hive.exec.dynamic.partition.mode = nonstrict;*

*set hive.enforce.bucketing = true;*

**

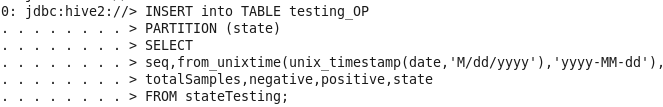
4.3 Optimized External tables creation in Hive :



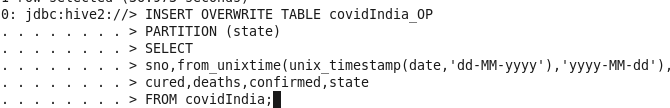


**Step 5: Load data to the optimized hive tables from normal hive tables.**

The date in stateTesting table is of the format M/dd/yyyy . We need to convert it to yyyy-MM-dd format.

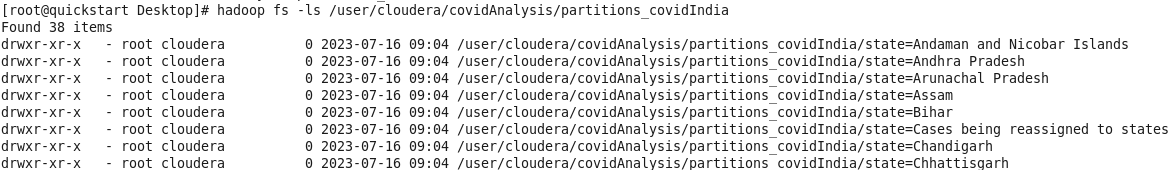


The date in covidIndia table is of the format dd-MM-yyyy . We need to convert it to yyyy-MM-dd format.

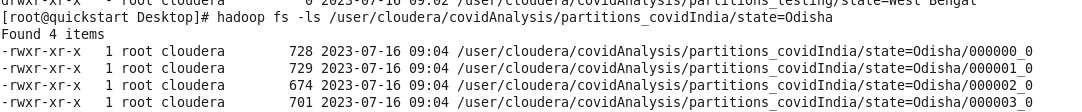


Verification:

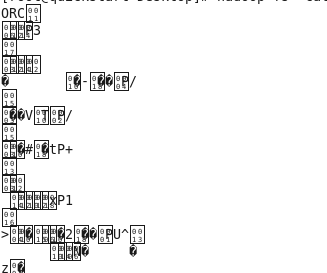
Partitions based on state



4 Buckets in each partition



Data stored in ORC format



**Step 6-Inner Join two tables in Hive and get a consolidated table.**

Performing Map side join on two columns ‘date’ and ‘state’.

Here it is assumed that the State\_Testing table is small enough to fit in memory

6.1 Set the below Properties

*set hive.auto.convert.join = false;*

*set hive.ignore.mapjoin.hint = false;*

6.2 Execute Inner Join as Mapside join (testing\_OP is the smaller table)

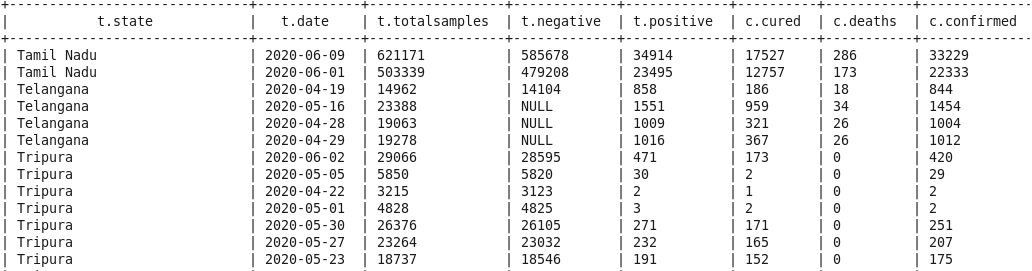
*SELECT /\*+ MAPJOIN(T)*

*\*/T.state,T.date,T.totalSamples,T.negative,T.positive,C.cured,C.deaths,*

*C.confirmed FROM testing\_OP T JOIN covidIndia\_OP C*

*ON (C.state = T.state) AND (C.date = T.date);*

Joined Table Snapshot



6.3 Create a consolidated table after join

*CREATE TABLE covid\_details AS*

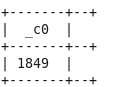
*SELECT /\*+ MAPJOIN(T)*

*\*/T.state,T.date,T.totalSamples,T.negative,T.positive,C.cured,C.deaths,*

*C.confirmed FROM testing\_OP T JOIN covidIndia\_OP C*

*ON (C.state = T.state) AND (C.date = T.date);*

Count:



**Step 7- Analysis**

1. **Ideally, the number of samples tested positive and number of covid cases**

**confirmed must be the same. See which state/states have more**

**consistent data collection like The number of positive samples (table1)**

**match mostly with number of confirmed cases(table2), for which state.**

Ideally, the number of samples tested positive and number of covid cases

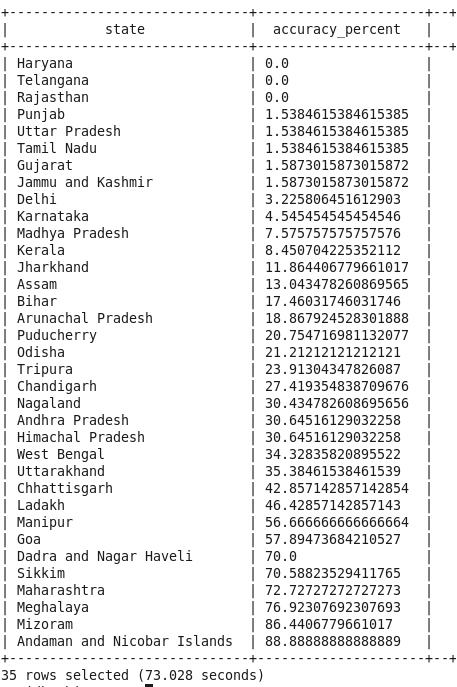
confirmed must be the same. So we find the percentage of records in each state where the data is consistent.

*select state,avg(case when positive = confirmed then 1 else 0 end)\*100 as accuracy\_percent*

*from covid\_details*

*group by state*

*order by accuracy\_percent;*

****

According to the above results, we can conclude the top 5 states with accurate covid testing for the given 2 months are **- Sikkim, Maharashtra, Meghalaya, Mizoram and Andaman and Nicobar Islands**

While the worst-performing ones are - **Haryana, Telangana, Rajasthan, Punjab, Uttar Pradesh, Tamil Nadu and Gujarat**

1. **For every state, find the total number of confirmed cases reported and**

**also the total number of positive samples tested, in the entire duration of**

**2 months, starting with the state with the highest cases.**

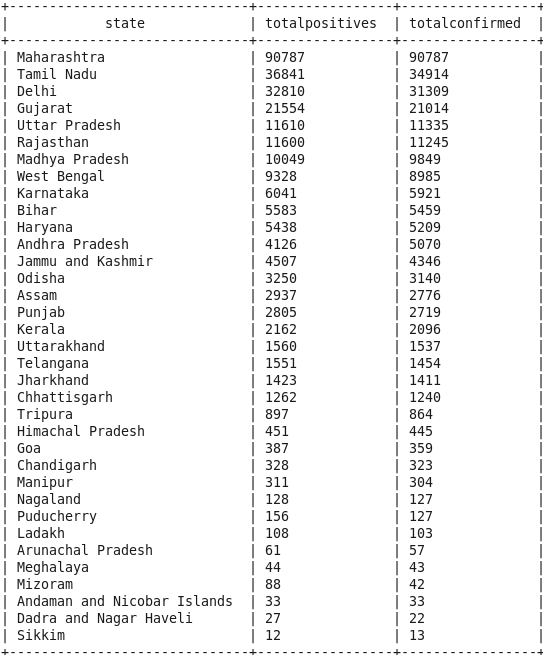
**If we scan the data we can see the data is cumulative for example TotalSamples shows cumulative total, Confirmed field shows cumulative value. So the value in each day is actually the total number till that day. Hence we only need to find the maximum number.**

*select state,max(positive) as totalPositives , max(confirmed) as totalConfirmed*

*from covid\_details*

*group by state*

*order by totalConfirmed desc;*

**