Mini Project

# Concepts Using

1. HDFS
2. MYSQL
3. PASSWORD ALIS IN SQOOP
4. SQOOP
5. HIVE
6. HIVE FILE FORMATS AND COMPRESSION TECHNIQUES
7. HBASE

We have 2 data sets:

**Covid19\_india**

Columns: Sno, Date, State, Cured, Deaths, Confirmed

**StatewiseTestingDetails**

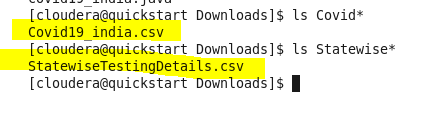
Columns: Seq, Date, State, Total\_Samples, Negative, Positive

# 1. HDFS

🡺 Here we are storing data in hdfs from local system

## Storing data in hdfs

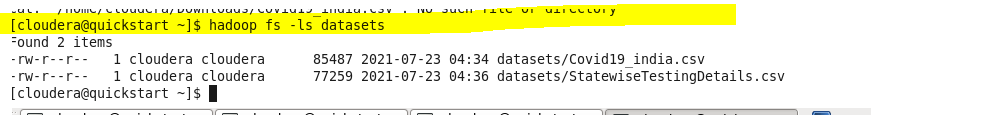
**Step1: Store that two files in local machine (/home/cloudera/Downloads) using WinScp**



**Step2: Create directory in HDFS**

hadoop fs -mkdir /user/cloudera/datasets

**Step3: Put these two files from local to hdfs**



hadoop fs -put '/home/cloudera/Downloads/Covid19\_india.csv' '/user/cloudera/datasets'

hadoop fs -put '/home/cloudera/Downloads/StatewiseTestingDetails.csv' '/user/cloudera/datasets'

# 2. MYSQL

🡺Here We Are Creating Target Tables and Staging Tables

create database assigniment;

**Step1: Create target tables**

create table covid19india (

Sno int NOT NULL PRIMARY KEY,

Date varchar(225),

State varchar(225),

Cured int,

Deaths int,

Confirmed int);

create table statewisetesting (

Seq int NOT NULL PRIMARY KEY,

Date varchar(225),

State varchar(225),

Total\_Samples int,

Negative int,

Positive int

);

**Step2: Create staging tables from target tables**

create table covid19india\_staging like covid19india;

create table statewisetesting\_staging like statewisetesting;

# 3. PASSWORD ALIS IN SQOOP

hadoop credential create mysql.cloudera.password -provider jceks://hdfs/user/cloudera/mysql.password.jceks

# 4. SQOOP

## Sqoop export

Here we are exporting data to the tables in MYSQL database using password alias

sqoop export \

-Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/mysql.password.jceks \

--connect jdbc:mysql://localhost:3306/assigniment \

--username root \

--password-alias mysql.cloudera.password \

--table covid19india \

--staging-table covid19india\_staging \

--clear-staging-table \

--export-dir '/user/cloudera/datasets/Covid19\_india.csv' \

--fields-terminated-by ','

sqoop export \

-Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/mysql.password.jceks \

--connect jdbc:mysql://localhost:3306/assigniment \

--username root \

--password-alias mysql.cloudera.password \

--table statewisetesting \

--staging-table statewisetesting\_staging \

--clear-staging-table \

--export-dir '/user/cloudera/datasets/ StatewiseTestingDetails.csv' \

--fields-terminated-by ','

## Sqoop Import

hadoop fs -mkdir /user/cloudera/datasets2

## Create sqoop job for there two tables in MySql

sqoop job \

-Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/mysql.password.jceks \

--create job\_covid19\_india \

-- import \

--connect jdbc:mysql://localhost:3306/assigniment \

--username root \

--password-alias mysql.cloudera.password \

--table covid19india \

--warehouse-dir '/user/cloudera/datasets2' \

--check-column Sno \

--incremental append \

--last-value 0

sqoop job \

-Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/mysql.password.jceks \

--create job\_statewisetesting\_india \

-- import \

--connect jdbc:mysql://localhost:3306/assigniment \

--username root \

--password-alias mysql.cloudera.password \

--table statewisetesting \

--warehouse-dir '/user/cloudera/datasets2' \

--check-column Seq \

--incremental append \

--last-value 0

## Execute sqoop jobs

sqoop job –exec job\_covid19\_india

sqoop job –exec job\_statewisetesting\_india

# 5. Hive

## Create optimized tables

Step1: Create hive table (External or Managed)

Step2: Load data into hive table -- For external table we are skipping this process bz we are giving file directory in ‘location’

Step3: Create optimized table

Step4: Load data form hive table to hive optimized table

## Step1: Create hive external tables on top of the data in HDFS

create external table covid\_india (

Sno int,

Date string,

State string,

Cured int,

Deaths int,

Confirmed int)

row format delimited

fields terminated by ','

location '/user/cloudera/datasets2/covid19india/';

create external table state\_testing (

Seq int,

Date string,

State string,

Total\_Samples int,

Negative int,

Positive int

)

row format delimited

fields terminated by ','

location '/user/cloudera/datasets2/statewisetesting'

## Step3: Create optimized table

hadoop fs -mkdir datasets3

hadoop fs -mkdir datasets3/covid\_india\_partition

hadoop fs -mkdir datasets3/statewise\_partition

### Set below properties

set hive.exec.dynamic.partition=true;

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

set hive.enforce.bucketing=true;

create external table covid\_india\_ORC (

Date date,

State string,

Cured int,

Deaths int,

Confirmed int)

partitioned by (Sno int)

clustered by (Date) into 4 buckets

stored as ORC

location '/user/cloudera/datasets3/covid\_india\_partition/'

tblproperties('orc.compress'='sanppy');

create external table state\_testing\_ORC (

Date date,

State string,

Total\_Samples int,

Negative int,

Positive int

)

partitioned by (Seq int)

clustered by (Date) into 4 buckets

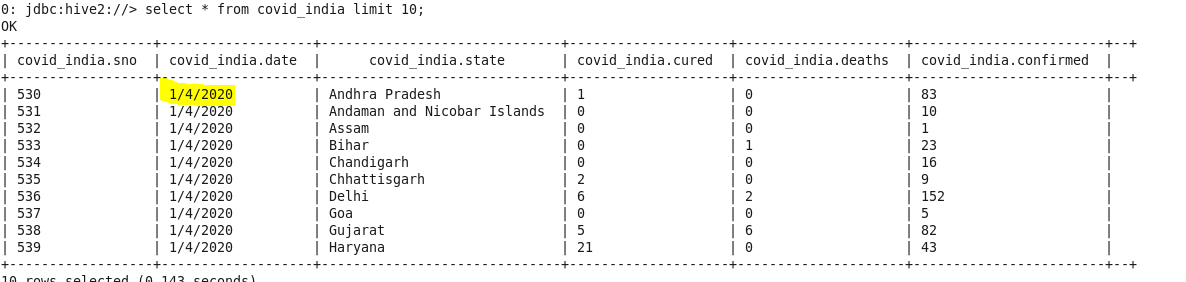
stored as ORC

location '/user/cloudera/datasets3/statewise\_partition'

tblproperties('orc.compress'='snappy');

## Step4: Load data from normal hive tables to optimized tables

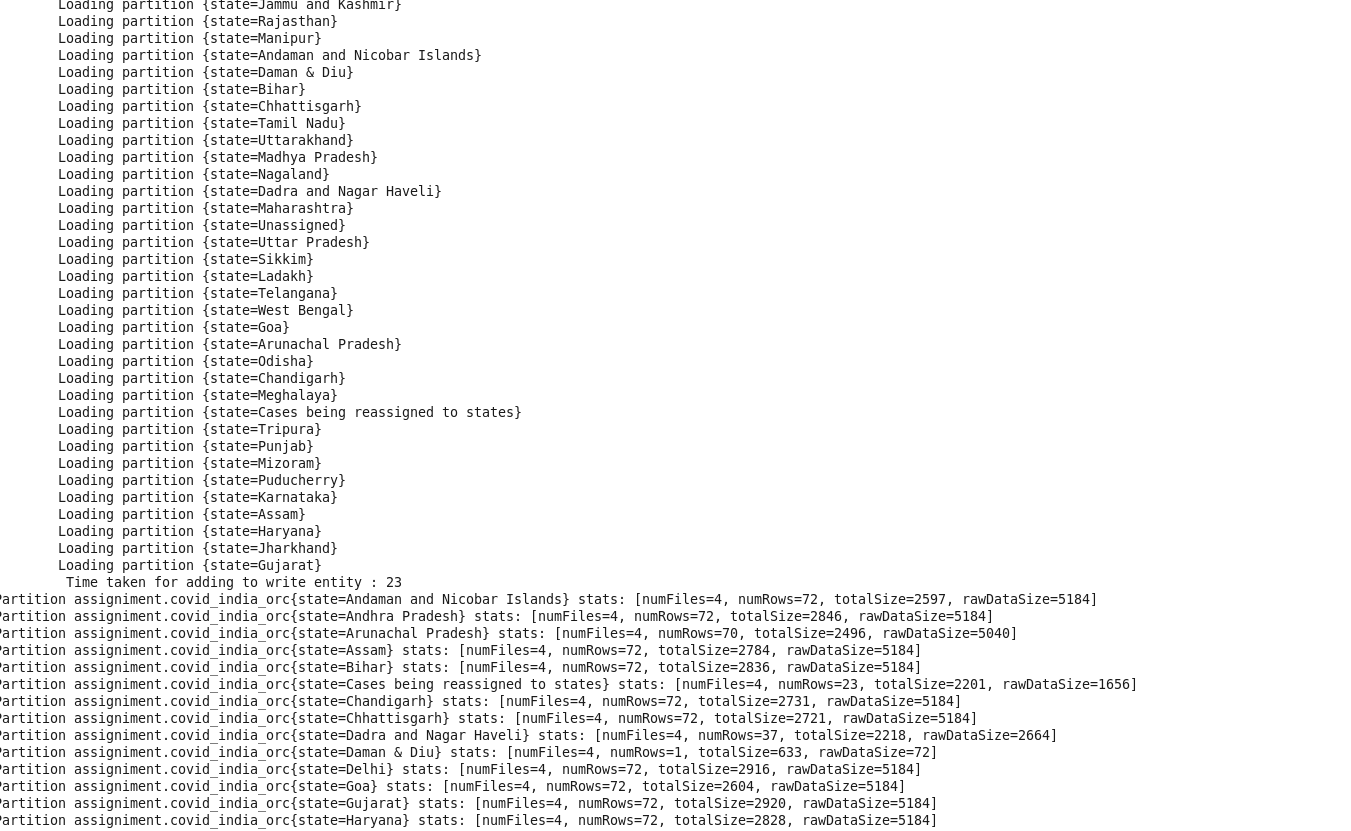
Note: While loading data we have to make sure the **Date column** is in date format



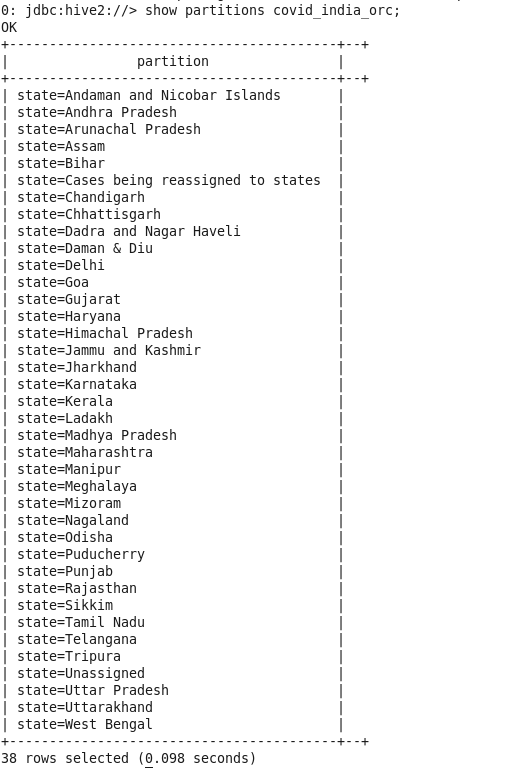
insert overwrite table covid\_india\_ORC

partition (State)

select Sno,from\_unixtime(unix\_timestamp(Date,'dd/M/yyyy'),'yyyy-MM-dd'), Cured, Deaths, Confirmed, State from covid\_india;



show partitions covid\_india\_orc;



insert overwrite table state\_testing\_ORC

partition (State)

select Seq ,from\_unixtime(unix\_timestamp(Date,'M/dd/yyyy'),'yyyy-MM-dd') date , Total\_Samples, Negative , Positive, State from state\_testing;

set hive.auto.convert.join=true;

### Joining two tables

select c.State,c.Date,s.Total\_Samples, s.negative, s.positive, c.cured, c.deaths, c.confirmed from covid\_india\_orc c

join state\_testing\_orc s on s.Date = c.Date and s.State = c.State;

### Creating consolidated table

create table covid\_details as

select c.State,c.Date,s.Total\_Samples, s.negative, s.positive, c.cured, c.deaths, c.confirmed from covid\_india\_orc c

join state\_testing\_orc s on s.Date = c.Date and s.State = c.State;

Finding consistence data

positive samples

select state,min(positive-confirmed) MostConsistency,max(positive-confirmed) LeastConsistency from covid\_details

group by state;

# 7. Hbase

## Creating Hive-Hbase table

CREATE TABLE cov(rkey string, state string, Date date, total\_samples int, negative int, positive int, cured int, deaths int, confirmed int)

STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH

SERDEPROPERTIES ("hbase.columns.mapping"=":key, testing: state, testing: date, testing: total\_samples, testing: negative, testing: positive,

covidcases: cured,covidcases: deaths,covidcases: confirmed")TBLPROPERTIES("hbase.table.name"="cov\_hbase");

insert overwrite table cov select concat(state,'',date) rkey,state, date, total\_samples, negative, positive, cured, deaths, confirmed

from covid\_details;