**1. Create directories in HDFS**

hadoop fs -mkdir /user/cloudera/data\_hive

hadoop fs -mkdir data\_hive/testing

hadoop fs -mkdir data\_hive/covidindia

**2. Load data into HDFS**

hadoop fs -put /home/cloudera/Desktop/shared2/StatewiseTestingDetails-201004-194827.csv data\_hive/testing

hadoop fs -put /home/cloudera/Desktop/shared2/Covid19\_india-201004-194827.csv data\_hive/covidindia

**3. Check the content of the files**

hadoop fs -cat data\_hive/testing/StatewiseTestingDetails-201004-194827.csv | head

1,4/17/2020,Andaman and Nicobar Islands,1403,1210,12

2,4/24/2020,Andaman and Nicobar Islands,2679,,27

3,4/27/2020,Andaman and Nicobar Islands,2848,,33

4,5/1/2020,Andaman and Nicobar Islands,3754,,33

5,5/16/2020,Andaman and Nicobar Islands,6677,,33

6,5/19/2020,Andaman and Nicobar Islands,6965,,33

7,5/20/2020,Andaman and Nicobar Islands,7082,,33

8,5/21/2020,Andaman and Nicobar Islands,7167,,33

9,5/22/2020,Andaman and Nicobar Islands,7263,,33

10,5/23/2020,Andaman and Nicobar Islands,7327,,33

cat: Unable to write to output stream.

hadoop fs -cat data\_hive/covidindia/Covid19\_india-201004-194827.csv | head

530,1/4/2020,Andhra Pradesh,1,0,83

531,1/4/2020,Andaman and Nicobar Islands,0,0,10

532,1/4/2020,Assam,0,0,1

533,1/4/2020,Bihar,0,1,23

534,1/4/2020,Chandigarh,0,0,16

535,1/4/2020,Chhattisgarh,2,0,9

536,1/4/2020,Delhi,6,2,152

537,1/4/2020,Goa,0,0,5

538,1/4/2020,Gujarat,5,6,82

539,1/4/2020,Haryana,21,0,43

cat: Unable to write to output stream.

**4. Count the no. of records**

hadoop fs -cat data\_hive/testing/StatewiseTestingDetails-201004-194827.csv | wc -l

1922

hadoop fs -cat data\_hive/covidindia/Covid19\_india-201004-194827.csv | wc -l

2390

**5. Create tables in MySql**

|  |  |
| --- | --- |
| **Staging Table** | **Actual Table** |
| CREAT TABLE IF NOT EXISTS State\_Testing\_Stage(  seq int not null primary key,  date varchar(30),  state varchar(50) not null,  total\_samples int,  negative int,  positive int); | CREAT TABLE IF NOT EXISTS State\_Testing(  seq int not null primary key,  date varchar(30),  state varchar(50) not null,  total\_samples int,  negative int,  positive int); |
| CREAT TABLE IF NOT EXISTS Covid\_India\_Stage(  sno int not null primary key,  date varchar(30),  state varchar(50) not null,  cured int,  deaths int,  confirmed int); | CREAT TABLE IF NOT EXISTS Covid\_India(  sno int not null primary key,  date varchar(30),  state varchar(50) not null,  cured int,  deaths int,  confirmed int); |

**5. Export data into mysql tables from HDFS**

sqoop export \

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

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

--username root \

--password-alias mysql.test\_db.password \

--table State\_Testing \

--staging-table State\_Testing\_Stage \

--clear-staging-table \

--export-dir /user/cloudera/data\_hive/testing \

--fields-terminated-by ','

sqoop export \

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

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

--username root \

--password-alias mysql.test\_db.password \

--table Covid\_India \

--staging-table Covid\_India\_Stage \

--clear-staging-table \

--export-dir /user/cloudera/data\_hive/covidindia \

--fields-terminated-by ','

**6. Import Data from Mysql to HDFS**

sqoop job \

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

--create job\_testingdetails\_inc \

-- import \

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

--username root \

--password-alias mysql.test\_db.password \

--table State\_Testing \

--warehouse-dir /user/cloudera/data\_hive/imported\_data \

--covidindia

--incremental append \

--check-column seq \

--last-value 0 \

--compress

**Run testingdetails Job:**

sqoop job --exec job\_testingdetails\_inc

sqoop job \

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

--create job\_covidindia\_inc \

-- import \

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

--username root \

--password-alias mysql.test\_db.password \

--table Covid\_India \

--warehouse-dir /user/cloudera/data\_hive/imported\_data \

--split-by sno \

--incremental append \

--check-column sno \

--last-value 0 \

--compres

**Run Covid India Job:**

sqoop job --exec job\_covidindia\_inc

**7. Create External Hive tables**

CREATE EXTERNAL TABLE IF NOT EXISTS State\_Testing(

seq INT,

date STRING,

state STRING,

total\_samples INT,

negative INT,

positive INT)

COMMENT 'Table to store state details'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE

LOCATION '/user/cloudera/data\_hive/imported\_data/State\_Testing';

CREATE EXTERNAL TABLE IF NOT EXISTS Covid\_India(

sno INT,

date STRING,

state STRING,

cured INT,

deaths INT,

confirmed INT)

COMMENT 'Table to store covid datails'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE

LOCATION '/user/cloudera/data\_hive/imported\_data/Covid\_India';

**8. Create Directories in HDFS for dynamic partitions**

Hadoop fs -mkdir /user/cloudera/data\_hive/partitions\_covidindia

Hadoop fs -mkdir /user/cloudera/data\_hive/partitions\_testing

**9. Create Optimized tables in Hive:**

CREATE EXTERNAL TABLE IF NOT EXISTS Covid\_India\_ORC(

sno INT,

date Date,

cured INT,

deaths INT,

confirmed INT)

PARTITIONED BY (state STRING)

CLUSTERED BY (date) into 4 BUCKETS

STORED AS ORC

LOCATION '/user/cloudera/data\_hive/partitions\_covidindia'

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

CREATE EXTERNAL TABLE IF NOT EXISTS State\_Testing\_ORC(

seq INT,

date Date,

total\_samples INT,

negative INT,

positive INT)

PARTITIONED BY (state STRING)

CLUSTERED BY (date) into 4 BUCKETS

STORED AS ORC

LOCATION '/user/cloudera/data\_hive/partitions\_testing'

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

**9. Insert Data into Optimized tables from normal Hive tables**

|  |  |
| --- | --- |
| INSERT OVERWRITE TABLE State\_Testing\_ORC  PARTITION(state)  SELECT  seq, from\_unixtime(unix\_timestamp(date, 'M/dd/yyyy'),'yyyy-MM-dd'),  total\_samples, negative, positive, state  FROM state\_testing; | INSERT OVERWRITE TABLE Covid\_India\_ORC  PARTITION(state)  SELECT  sno, from\_unixtime(unix\_timestamp(date, 'M/dd/yyyy'),'yyyy-MM-dd'),  cured, deaths,confirmed, state  FROM Covid\_India; |

**10. Use Query optimization technique for join operation**

SELECT /\*+MAPJOIN(S)\*/

S.state, S.date, S.total\_samples, S.negative, S.positive, C.cured, C.deaths, C.confirmed

FROM State\_Testing\_ORC S JOIN Covid\_India\_ORC C ON

(S.state = C.state) AND (S.date = C.date) limit 100;

**11. Create Table from Mapside Join**

CREATE TABLE CovidIndia\_Details AS

SELECT /\*+MAPJOIN(S)\*/

S.state, S.date, S.total\_samples, S.negative, S.positive, C.cured, C.deaths, C.confirmed

FROM State\_Testing\_ORC S JOIN Covid\_India\_ORC C ON

(S.state = C.state) AND (S.date = C.date);

**12. Find Analysis on below use cases**

1. **Find state where maximum number of covid cases reported.**
2. **State wise, calculate total number of confirmed cases and total count of positive cases.**
3. **For every state, find count of confirmed cases on latest date.**