TRENDYTECH

UPLIFT YOUR CAREER!

Assignment Solution

Week 5-6: Apache Hive Advance - Part1 & Part2

Assignment-Solution

Steps

Step 1 -Data Preprocessing-

1.1 Create directories in hdfs to store the csv files-

hadoop fs -mkdir /user/cloudera/data1
hadoop fs -mkdir data1/Testing
hadoop fs -mkdir data1/CovidIndia

1.2 Copy the files from local to hdfs-

hadoop fs -put Downloads/StatewiseTestingDetails.csv /user/cloudera/data1/Testing

hadoop fs -put Downloads/Covid19_india.csv /user/cloudera/data1/CovidIndia

1.3 Visualize the content of the files in hdfs-

hadoop fs -cat data1/Testing/StatewiseTestingDetails.csv | head

```
(base) [cloudera@quickstart ~]$ hadoop fs -cat  data1/Testing/StatewiseTestingDe
tails.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
```

hadoop fs -cat data1/CovidIndia/Covid19_india.csv | head

```
(base) [cloudera@quickstart ~]$ hadoop fs -cat  data1/CovidIndia/Covid19_india.c
sv | 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
```

Verify number of records in two files:

hadoop fs -cat data1/Testing/StatewiseTestingDetails.csv | wc -l

```
(base) [cloudera@quickstart ~]$ hadoop fs -cat  data1/Testing/StatewiseTestingDe
tails.csv | wc -l
1922
```

hadoop fs -cat data1/CovidIndia/Covid19_india.csv | wc -l

```
(base) [cloudera@quickstart ~]$ hadoop fs -cat  data1/CovidIndia/Covid19_india.c
sv | wc -l
2390
```


1.4 Creation of Tables in Mysql-

create 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);

create 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);

```
create 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);

create 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);
```

1.5 Sqoop Export of data from hdfs to Mysql

```
sqoop export \
-Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/encryptpswd/jceks_pswdfile \
--connect jdbc:mysql://quickstart.cloudera:3306/test_db \
--username root \
--password-alias mysql.test_db.securepassword \
--table State_Testing \
--staging-table State_Testing_Stage    \
--clear-staging-table \
--export-dir /user/cloudera/data1/Testing \
--fields-terminated-by ','
```

```
> -Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/encryptp
swd/jceks_pswdfile \
> --connect jdbc:mysql://quickstart.cloudera:3306/test_db \
> --username root \
> --password-alias mysql.test_db.securepassword \
> --table State_Testing \
> --staging-table State_Testing_Stage  \
> --clear-staging-table \
> --export-dir /user/cloudera/data1/Testing \
> --fields-terminated-by ','
```

Output:

```
20/06/11 12:19:35 INFO manager.SqlManager: Migrated 1922 records from `State_Tes
ting_Stage` to `State_Testing`
```

**sqoop export **
**-Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/encryptpswd/jceks_pswdfile **
**--connect jdbc:mysql://quickstart.cloudera:3306/test_db **
**--username root **
**--password-alias mysql.test_db.securepassword **
**--table Covid_India **
**--staging-table Covid_India_Stage **
**--clear-staging-table **
**--export-dir /user/cloudera/data1/CovidIndia **
--fields-terminated-by ','

```
(base) [cloudera@quickstart ~]$ sqoop export \
> -Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/encryptpswd/jceks_pswdfile \
> --connect jdbc:mysql://quickstart.cloudera:3306/test_db \
> --username root \
> --password-alias mysql.test_db.securepassword \
> --table Covid_India \
> --staging-table Covid_India_Stage \
> --clear-staging-table \
> --export-dir /user/cloudera/data1/CovidIndia \
> --fields-terminated-by ','
```

```
20/06/12 07:43:25 INFO manager.SqlManager: Migrated 2390 records from `Covid_India_Stage` to `Covid_India`
(base) [cloudera@quickstart ~]$
```

Verify in Mysql:

```
mysql> select count(*) from State_Testing;
+----------+
| count(*) |
+----------+
|     1922 |
+----------+
1 row in set (0.02 sec)
```

```
mysql> select count(*) from Covid_India;
+----------+
| count(*) |
+----------+
|     2390 |
+----------+
1 row in set (0.01 sec)
```

Now assuming that both tables are in Mysql -This is the starting point of our problem.

Step 2:

Sqoop Import -From Mysql to HDFS:

The tables **State_Testing, Covid_India** will be sqoop imported with Incremental in append mode, as data goes on adding to the tables on daily basis. So we create a saved sqoop job for both.Also password encryption has been used, with default compression gzip.

A directory named **sqoop_imported** is created in hdfs for storing the sqoop imported data from two tables.

Sqoop job Creation for State_Testing Table:

```
sqoop job \
-Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/encryptpswd/jceks_pswdfile \
--create job_testingdetails_inc \
-- import \
--connect jdbc:mysql://quickstart.cloudera:3306/test_db \
--username root \
--password-alias mysql.test_db.securepassword \
--table State_Testing    \
--warehouse-dir /user/cloudera/data1/sqoop_imported \
--split-by seq \
--incremental append \
--check-column seq \
--last-value 0 \
--compress
```

```
(base) [cloudera@quickstart ~]$ sqoop job \
> -Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/encryptpswd/jceks_pswdfile \
> --create job_testingdetails_inc \
> -- import \
> --connect jdbc:mysql://quickstart.cloudera:3306/test_db \
> --username root \
> --password-alias mysql.test_db.securepassword \
> --table State_Testing \
> --warehouse-dir /user/cloudera/data1/sqoop_imported \
> --split-by seq \
> --incremental append \
> --check-column seq \
> --last-value 0 \
> --compress
Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
20/06/11 13:33:03 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6-cdh5.13.0
```

Execute the saved job:

sqoop job --exec job_testingdetails_inc

```
(base) [cloudera@quickstart ~]$ sqoop job --exec job_testingdetails_inc
```

Output:

```
20/06/11 13:37:50 INFO mapreduce.ImportJobBase: Transferred 24.7686 KB in 35.5348 seconds (713.7511 bytes/sec)
20/06/11 13:37:50 INFO mapreduce.ImportJobBase: Retrieved 1922 records.
```

HDFS View :

hadoop fs -ls /user/cloudera/data1/sqoop_imported/State_Testing

```
(base) [cloudera@quickstart ~]$ hadoop fs -ls  /user/cloudera/data1/sqoop_imported/State_Testing
Found 4 items
-rw-r--r--   1 cloudera cloudera       6122 2020-06-11 13:37 /user/cloudera/data1/sqoop_imported/State_Testing/part-m-00000.gz
-rw-r--r--   1 cloudera cloudera       6639 2020-06-11 13:37 /user/cloudera/data1/sqoop_imported/State_Testing/part-m-00001.gz
-rw-r--r--   1 cloudera cloudera       5823 2020-06-11 13:37 /user/cloudera/data1/sqoop_imported/State_Testing/part-m-00002.gz
-rw-r--r--   1 cloudera cloudera       6779 2020-06-11 13:37 /user/cloudera/data1/sqoop_imported/State_Testing/part-m-00003.gz
(base) [cloudera@quickstart ~]$
```

Sqoop job Creation for Covid_India Table:

**sqoop job **
**-Dhadoop.security.credential.provider.path=jceks://hdfs/user/cloudera/encryptpswd/jceks_pswdfile **
**--create job_coviddetails_inc **
**-- import **
**--connect jdbc:mysql://quickstart.cloudera:3306/test_db **

```
--username root \
--password-alias mysql.test_db.securepassword \
--table Covid_India \
--warehouse-dir /user/cloudera/data1/sqoop_imported \
--split-by sno \
--incremental append \
--check-column sno    \
--last-value 0 \
--compress
```

Execute the saved job:

sqoop job --exec job_coviddetails_inc

HDFS View:

```
(base) [cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/data1/sqoop_importe
d/
```

```
(base) [cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/data1/sqoop_imported/Covid_India
Found 4 items
-rw-r--r--   1 cloudera cloudera       4763 2020-06-12 09:43 /user/cloudera/data1/sqoop_imported/Covid_India/part-m-00000.gz
-rw-r--r--   1 cloudera cloudera       5398 2020-06-12 09:43 /user/cloudera/data1/sqoop_imported/Covid_India/part-m-00001.gz
-rw-r--r--   1 cloudera cloudera       6007 2020-06-12 09:43 /user/cloudera/data1/sqoop_imported/Covid_India/part-m-00002.gz
-rw-r--r--   1 cloudera cloudera       6392 2020-06-12 09:43 /user/cloudera/data1/sqoop_imported/Covid_India/part-m-00003.gz
```

```
Found 4 items
-rw-r--r--   1 cloudera cloudera       4763 2020-06-12 09:43 /user/cloudera/data
1/sqoop_imported/Covid_India/part-m-00000.gz
-rw-r--r--   1 cloudera cloudera       5398 2020-06-12 09:43 /user/cloudera/data
1/sqoop_imported/Covid_India/part-m-00001.gz
-rw-r--r--   1 cloudera cloudera       6007 2020-06-12 09:43 /user/cloudera/data
1/sqoop_imported/Covid_India/part-m-00002.gz
-rw-r--r--   1 cloudera cloudera       6392 2020-06-12 09:43 /user/cloudera/data
1/sqoop_imported/Covid_India/part-m-00003.gz
```

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

Commands -Database Creation in Hive

create database if not exists covid_india;

use covid_india;

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

```
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 Statewise Testing Details'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/cloudera/data1/sqoop_imported/State_Testing';
```

Mapreduce Job Outcome:

```
MapReduce Total cumulative CPU time: 3 seconds 760 msec
Ended Job = job_1591286176727_0111
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.76 sec   HDFS Read: 85110 H
DFS Write: 5 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 760 msec
OK
+-------+--+
|  _c0  |
+-------+--+
| 1922  |
+-------+--+
1 row selected (27.631 seconds)
```

Sample Output:

select * from State_Testing LIMIT 5;

```
| state_testing.seq  | state_testing.date  |     state_testing.state      | stat
e_testing.total_samples  | state_testing.negative  | state_testing.positive  |
+--------------------+---------------------+------------------------------+-----
-------------------------+-------------------------+-------------------------+--
+
| 1                  | 4/17/2020           | Andaman and Nicobar Islands  | 1403
                        | 1210                    | 12                       |
| 2                  | 4/24/2020           | Andaman and Nicobar Islands  | 2679
                        | NULL                    | 27                       |
| 3                  | 4/27/2020           | Andaman and Nicobar Islands  | 2848
                        | NULL                    | 33                       |
| 4                  | 5/1/2020            | Andaman and Nicobar Islands  | 3754
                        | NULL                    | 33                       |
| 5                  | 5/16/2020           | Andaman and Nicobar Islands  | 6677
                        | NULL                    | 33                       |
+--------------------+---------------------+------------------------------+-----
-------------------------+-------------------------+-------------------------+--
+
5 rows selected (0.188 seconds)
```

CREATE EXTERNAL TABLE IF NOT EXISTS Covid_India
(sno INT,
date STRING,
state STRING,
cured INT,
deaths INT,
confirmed INT)
COMMENT 'Table to store Statewise Covid Count Details'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/cloudera/data1/sqoop_imported/Covid_India';

```
Total MapReduce CPU Time Spent: 4 seconds 180 msec
OK
+-------+--+
|  _c0  |
+-------+--+
| 2390  |
+-------+--+
```

Verify the count :

SELECT * FROM Covid_India LIMIT 5;

```
0: jdbc:hive2://> CREATE EXTERNAL TABLE IF NOT EXISTS Covid_India
. . . . . > (sno INT,
. . . . . > date STRING,
. . . . . > state STRING,
. . . . . > cured INT,
. . . . . > deaths INT,
. . . . . > confirmed INT)
. . . . . > COMMENT 'Table to store Statewise Covid Count Details'
. . . . . > ROW FORMAT DELIMITED
. . . . . > FIELDS TERMINATED BY ','
. . . . . > STORED AS TEXTFILE
. . . . . > LOCATION '/user/cloudera/data1/sqoop_imported/Covid_India';
OK
No rows affected (0.177 seconds)
0: jdbc:hive2://> SELECT * FROM Covid_India LIMIT 5;
OK
+-----------------+-----------------+-----------------------+-----------------+-----------------+-----------------+
| covid_india.sno | covid_india.date |     covid_india.state     | covid_india.cured | covid_india.deaths | covid_india.confirmed |
+-----------------+-----------------+-----------------------+-----------------+-----------------+-----------------+
| 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              |
+-----------------+-----------------+-----------------------+-----------------+-----------------+-----------------+
5 rows selected (0.173 seconds)
```

Step 4 :Create Optimized External tables in Hive:

We will create Optimized External Tables in Hive:

Note: In the following steps, we can create hive tables with textFile format also , but for better optimization ,ORC file format has been used with compression technique Snappy. Other optimizations like partitioning and bucketing can be applied as usual.

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

There might be frequent queries on State and Date, so we chose these columns for Partitioning and Bucketing. Also to get a consolidated view

of the data,we want to perform a Map-Side Join, and for that State and Date will be used as Join columns.So both tables can be Partitioned on State and Bucketed on Date column.

CREATE DIRECTORIES IN HDFS for the Dynamically created Partitions:

hadoop fs -mkdir /user/cloudera/data1/partitions_testing
hadoop fs -mkdir /user/cloudera/data1/partitions_covidindia

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;

```
0: jdbc:hive2://> set hive.exec.dynamic.partition;
+-------------------------------------+--+
|                set                  |
+-------------------------------------+--+
| hive.exec.dynamic.partition=true    |
+-------------------------------------+--+
1 row selected (0.024 seconds)
0: jdbc:hive2://> set hive.exec.dynamic.partition.mode
. . . . . . . . > ;
+-------------------------------------------+--+
|                   set                     |
+-------------------------------------------+--+
| hive.exec.dynamic.partition.mode=strict   |
+-------------------------------------------+--+
1 row selected (0.009 seconds)
0: jdbc:hive2://> set hive.exec.dynamic.partition.mode = nonstrict;
```

```
--------------------------------+--+
              set               |
--------------------------------+--+
 hive.enforce.bucketing=false   |
--------------------------------+--+
 row selected (0.164 seconds)
): jdbc:hive2://> set hive.enforce.bucketing = true;
No rows affected (0.018 seconds)
): jdbc:hive2://> set hive.enforce.bucketing ;
--------------------------------+--+
              set               |
--------------------------------+--+
 hive.enforce.bucketing=true    |
--------------------------------+--+
```

ORC EXTERNAL TABLE CREATION IN HIVE:

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/data1/partitions_testing'
TBLPROPERTIES('orc.compress' = 'SNAPPY');

```
0: jdbc:hive2://> 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/data1/partitions_testing'
. . . . . . . . > TBLPROPERTIES('orc.compress' = 'SNAPPY');
OK
```

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

Note:Date has also to be formatted to proper hive format which is **yyyy-mm-dd** by default.

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;

```
0: jdbc:hive2://> 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;
```

Verify the data:

select * from State_Testing_ORC LIMIT 10;

state_testing_orc.seq	state_testing_orc.date	state_testing_orc.total_samples	state_testing_orc.negative	state_testing_orc.positive	state_testing_orc.state
1	2020-04-17	1403	1210	17	Andaman and Nicobar Islands
2	2020-04-24	2679	NULL	27	Andaman and Nicobar Islands
3	2020-04-27	2848	NULL	33	Andaman and Nicobar Islands
4	2020-05-01	3754	NULL	33	Andaman and Nicobar Islands
5	2020-05-16	6677	NULL	33	Andaman and Nicobar Islands
6	2020-05-19	6965	NULL	33	Andaman and Nicobar Islands
7	2020-05-20	7082	NULL	33	Andaman and Nicobar Islands
8	2020-05-21	7167	NULL	33	Andaman and Nicobar Islands
9	2020-05-22	7203	NULL	33	Andaman and Nicobar Islands
10	2020-05-23	7327	NULL	33	Andaman and Nicobar Islands

10 rows selected (0.148 seconds)

select count(*) from State_Testing_ORC;

```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.18 sec   HDFS Read: 138410
HDFS Write: 5 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 180 msec
OK
+-------+--+
|  _c0  |
+-------+--+
| 1922  |
+-------+--+
1 row selected (31.172 seconds)
```

Hdfs View for State_Testing table data : We can see the partitions being created dynamically.

hadoop fs -ls /user/cloudera/data1/partitions_testing

35 partitions are created as we have 35 unique state values in the State_Testing table.

```
(base) [cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/data1/partitions_te
sting/
Found 35 items
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:01 /user/cloudera/data
1/partitions_testing/state=Andaman and Nicobar Islands
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:01 /user/cloudera/data
1/partitions_testing/state=Andhra Pradesh
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:01 /user/cloudera/data
1/partitions_testing/state=Arunachal Pradesh
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:01 /user/cloudera/data
1/partitions_testing/state=Assam
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:01 /user/cloudera/data
1/partitions_testing/state=Bihar
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:01 /user/cloudera/data
1/partitions_testing/state=Chandigarh
```

Inside each partition 4,buckets created -

```
(base) [cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/data1/partitions_testing/state='West Bengal'
Found 4 items
-rwxr-xr-x   1 cloudera cloudera        708 2020-06-12 20:01 /user/cloudera/data1/partitions_testing/state=West Bengal/000000_0
-rwxr-xr-x   1 cloudera cloudera        678 2020-06-12 20:01 /user/cloudera/data1/partitions_testing/state=West Bengal/000001_0
-rwxr-xr-x   1 cloudera cloudera        720 2020-06-12 20:01 /user/cloudera/data1/partitions_testing/state=West Bengal/000002_0
-rwxr-xr-x   1 cloudera cloudera        720 2020-06-12 20:01 /user/cloudera/data1/partitions_testing/state=West Bengal/000003_0
```

```
(base) [cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/data1/partitions_testing/state=Karnataka
Found 4 items
-rwxr-xr-x   1 cloudera cloudera        651 2020-06-12 20:01 /user/cloudera/data1/partitions_testing/state=Karnataka/000000_0
-rwxr-xr-x   1 cloudera cloudera        653 2020-06-12 20:01 /user/cloudera/data1/partitions_testing/state=Karnataka/000001_0
-rwxr-xr-x   1 cloudera cloudera        653 2020-06-12 20:01 /user/cloudera/data1/partitions_testing/state=Karnataka/000002_0
-rwxr-xr-x   1 cloudera cloudera        675 2020-06-12 20:01 /user/cloudera/data1/partitions_testing/state=Karnataka/000003_0
```

Data stored as binary format (orc) inside buckets-

```
(base) [cloudera@quickstart ~]$ hadoop fs -cat /user/cloudera/data1/partitions_testing/state=Karnataka/000000_0
ORC
```

CREATE EXTERNAL TABLE IF NOT EXISTS Covid_India_ORC
(sno INT,
date STRING,
cured INT,
deaths INT,
confirmed INT)
PARTITIONED BY (state STRING)
CLUSTERED BY (date) into 4 BUCKETS
STORED AS ORC
LOCATION '/user/cloudera/data1/partitions_covidindia'
TBLPROPERTIES('orc.compress' = 'SNAPPY');

Create External Optimized ORC Table

```
0: jdbc:hive2://> CREATE EXTERNAL TABLE IF NOT EXISTS Covid_India_ORC
. . . . . . . . > (sno INT,
. . . . . . . . > date STRING,
. . . . . . . . > cured INT,
. . . . . . . . > deaths INT,
. . . . . . . . > confirmed INT)
. . . . . . . . > PARTITIONED BY (state STRING)
. . . . . . . . > CLUSTERED BY (date) into 4 BUCKETS
. . . . . . . . > STORED AS ORC
. . . . . . . . > LOCATION '/user/cloudera/data1/partitions_covidindia'
. . . . . . . . > TBLPROPERTIES('orc.compress' = 'SNAPPY');
OK
No rows affected (0.143 seconds)
```

Loading data to the ORC Partitioned table from normal hive table-

We take care of the proper date format also-

INSERT OVERWRITE TABLE Covid_India_ORC
PARTITION (state)
SELECT
sno,from_unixtime(unix_timestamp(date,'dd/M/yy'),'yyyy-MM-dd'),cured,deaths,confirmed,state
FROM Covid_India;

```
0: jdbc:hive2://> INSERT OVERWRITE TABLE  Covid_India_ORC
. . . . . . . . > PARTITION (state)
. . . . . . . . > SELECT sno,from_unixtime(unix_timestamp(date,'dd/M/yy'),'yyyy-
MM-dd'),cured,deaths,confirmed,state
. . . . . . . . > FROM Covid_India;
```

select count(*) from covid_india_orc;

```
Total MapReduce CPU Time Spent:
OK
+-------+--+
|  _c0  |
+-------+--+
| 2390  |
```

select * from covid_india_orc limit 10;

```
): jdbc:hive2://> select * from covid_india_orc limit 10;
OK
```

```
| covid_india_orc.sno  | covid_india_orc.date  | covid_india_orc.cured  | covid_
india_orc.deaths  | covid_india_orc.confirmed  |    covid_india_orc.state     |
+----------------------+-----------------------+------------------------+-------
-----------------+---------------------------+-----------------------------+-
-+
| 1607                 | 2020-05-05            | 32                     | 0
                 | 33                        | Andaman and Nicobar Islands  |
| 1191                 | 2020-04-22            | 11                     | 0
                 | 17                        | Andaman and Nicobar Islands  |
| 2559                 | 2020-06-02            | 33                     | 0
                 | 33                        | Andaman and Nicobar Islands  |
| 1479                 | 2020-05-01            | 16                     | 0
                 | 33                        | Andaman and Nicobar Islands  |
| 741                  | 2020-04-08            | 0                      | 0
                 | 10                        | Andaman and Nicobar Islands  |
| 1319                 | 2020-04-26            | 11                     | 0
                 | 33                        | Andaman and Nicobar Islands  |
| 2451                 | 2020-05-30            | 33                     | 0
                 | 33                        | Andaman and Nicobar Islands  |
| 2343                 | 2020-05-27            | 33                     | 0
                 | 33                        | Andaman and Nicobar Islands  |
| 2203                 | 2020-05-23            | 33                     | 0
                 | 33                        | Andaman and Nicobar Islands  |
| 834                  | 2020-04-11            | 0                      | 0
                 | 11                        | Andaman and Nicobar Islands  |
+----------------------+-----------------------+------------------------+-------
-----------------+---------------------------+-----------------------------+-
-+
10 rows selected (0.23 seconds)
```

Hdfs view of covid_india_orc table partitions-

hadoop fs -ls /user/cloudera/data1/partitions_covidindia

```
(base) [cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/data1/partitions_covidindia
Found 38 items
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Andaman and Nicobar Islands
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Andhra Pradesh
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Arunachal Pradesh
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Assam
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Bihar
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Cases being reassigned to states
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Chandigarh
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Chhattisgarh
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Dadra and Nagar Haveli
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Daman & Diu
drwxr-xr-x   - cloudera cloudera          0 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Delhi
```

View of Buckets created inside State Partitions-

hadoop fs -ls /user/cloudera/data1/partitions_covidindia/state='Jammu and Kashmir'

```
(base) [cloudera@quickstart ~]$ hadoop fs -ls /user/cloudera/data1/partitions_covidindia/state='Jammu and Kashmir'
Found 4 items
-rwxr-xr-x   1 cloudera cloudera        830 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Jammu and Kashmir/000000_0
-rwxr-xr-x   1 cloudera cloudera        845 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Jammu and Kashmir/000001_0
-rwxr-xr-x   1 cloudera cloudera        791 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Jammu and Kashmir/000002_0
-rwxr-xr-x   1 cloudera cloudera        847 2020-06-12 20:22 /user/cloudera/data1/partitions_covidindia/state=Jammu and Kashmir/000003_0
```

Each bucket has data in binary (orc) form:

hadoop fs -cat /user/cloudera/data1/partitions_covidindia/state='Jammu and Kashmir'/000003_0

```
(base) [cloudera@quickstart ~]$ hadoop fs -cat /user/cloudera/data1/partitions_covidindia/state='Jammu and Kashmir'/000003_0
ORC
2020-04-03
```

Step 6 -Inner Join two tables in Hive and get a consolidated table using Map-Side Join

Performing JOIN on two columns 'date' and 'state'.An inner join is performed, but a Map-side join for better optimization.

Map side join-Using Hints- Here it is assumed that the State_Testing table is small enough to fit in memory.

Set the below Properties-

set hive.auto.convert.join = false

```
0: jdbc:hive2://> set hive.auto.convert.join=false;
No rows affected (0.005 seconds)
0: jdbc:hive2://> set hive.auto.convert.join;
+--------------------------------+--+
|              set               |
+--------------------------------+--+
| hive.auto.convert.join=false   |
```

set hive.ignore.mapjoin.hint = false;

```
0: jdbc:hive2://> set hive.ignore.mapjoin.hint = false;
No rows affected (0.006 seconds)
0: jdbc:hive2://> set hive.ignore.mapjoin.hint;
+---------------------------------+--+
|               set               |
+---------------------------------+--+
| hive.ignore.mapjoin.hint=false  |
+---------------------------------+--+
1 row selected (0.011 seconds)
0: jdbc:hive2://>
```

Execute Inner Join as Mapside join,using Hints to indicate State_Testing is the smaller table-

SELECT /*+ MAPJOIN(T) */
T.state,T.date,T.total_samples,T.negative,T.positive,C.cured,C.deaths,C.confirmed
FROM State_Testing_ORC T JOIN Covid_India_ORC C

ON (C.state = T.state) AND (C.date = T.date) LIMIT 100;

```
0: jdbc:hive2://> SELECT /*+ MAPJOIN(T) */ T.state,T.date,T.total_samples,T.nega
tive,T.positive,C.cured,C.deaths,C.confirmed
. . . . . . . . > FROM  State_Testing_ORC T JOIN Covid_India_ORC C
. . . . . . . . >  ON (C.state = T.state) AND (C.date = T.date) LIMIT 100;
```

We can see number of reducers is set to 0.

```
Total jobs = 1
Execution log at: /tmp/cloudera/cloudera_20200615051717_356a2056-0090-4a55-b76f-2c04013b9a32.log
2020-06-15 05:17:56     Starting to launch local task to process map join;      maximum memory = 932184064
2020-06-15 05:18:00     Dump the side-table for tag: 0 with group count: 1021 into file: file:/tmp/cloudera/b9090888-f674-47eb-b5db-1bd1e67720a2/hive_2020-06-15_05-17-5
1_822_3569663929591626981-1/-local-10003/HashTable-Stage-1/MapJoin-t-10--.hashtable
2020-06-15 05:18:00     Uploaded 1 File to: file:/tmp/cloudera/b9090888-f674-47eb-b5db-1bd1e67720a2/hive_2020-06-15_05-17-51_822_3569663929591626981-1/-local-10003/Hash
Table-Stage-1/MapJoin-t-10--.hashtable (77818 bytes)
2020-06-15 05:18:00     End of local task; Time Taken: 3.788 sec.
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
```

Sample Output of INNER join as Map Side Join:

t.state	t.date	t.total_samples	t.negative	t.positive	c.cured	c.deaths	c.confirmed
Andaman and Nicobar Islands	2020-05-01	3754	NULL	33	16	0	33
Andaman and Nicobar Islands	2020-05-27	7499	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-23	7327	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-16	6677	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-24	7327	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-04-27	2848	NULL	33	11	0	33
Andaman and Nicobar Islands	2020-05-20	7082	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-28	7519	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-06-08	9341	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-29	7567	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-25	7363	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-21	7167	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-04-24	2679	NULL	27	11	0	22
Andaman and Nicobar Islands	2020-04-17	1403	1210	12	10	0	11
Andaman and Nicobar Islands	2020-06-09	9859	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-19	6965	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-22	7263	NULL	33	33	0	33
Andaman and Nicobar Islands	2020-05-26	7448	NULL	33	33	0	33
Andhra Pradesh	2020-05-05	133492	131775	1717	589	36	1717
Andhra Pradesh	2020-04-22	41512	40699	813	120	24	813
Andhra Pradesh	2020-06-02	395681	391890	3200	2378	64	3783
Andhra Pradesh	2020-05-01	102460	100997	1463	403	33	1463
Andhra Pradesh	2020-04-26	68034	66937	1097	231	31	1097
Andhra Pradesh	2020-05-30	363378	359917	2944	2226	60	3436

Without LIMIT Clause

SELECT /*+ MAPJOIN(T) */T.state,T.date,T.total_samples,T.negative,T.positive,C.cured,C.deaths,C.confirmed FROM State_Testing_ORC T JOIN Covid_India_ORC C ON (C.state = T.state) AND (C.date = T.date);

INNER JOIN returns the matching rows from both the tables.

t.state	t.date	t.total_samples	t.negative	t.positive	c.cured	c.deaths	c.confirmed
Maharashtra	2020-04-05	16008	14837	NULL	42	24	490
Maharashtra	2020-04-06	17563	15808	868	56	45	748
Maharashtra	2020-04-07	20877	19290	1018	56	48	868
Maharashtra	2020-04-09	20877	19290	868	117	72	1135
Maharashtra	2020-04-10	30000	28865	1135	125	97	1364
Maharashtra	2020-04-11	31841	30477	1761	188	110	1574
Maharashtra	2020-04-12	35668	34094	1761	208	127	1761
Maharashtra	2020-04-13	39725	37964	1996	217	149	1985
Maharashtra	2020-04-14	41071	39089	2340	229	160	2337
Maharashtra	2020-04-15	45142	42808	2690	259	178	2687
Maharashtra	2020-04-16	50882	48198	2916	295	187	2919
Maharashtra	2020-04-17	55678	52762	3204	300	194	3205
Maharashtra	2020-04-18	60166	56964	3323	331	201	3323
Maharashtra	2020-04-19	66796	63476	3651	365	211	3651
Maharashtra	2020-04-20	71321	67673	4204	507	223	4203
Maharashtra	2020-04-21	75838	71638	4676	572	232	4669
Maharashtra	2020-04-22	82304	77638	5229	722	251	5221
Maharashtra	2020-04-23	89197	83979	5218	789	269	5652
Maharashtra	2020-04-24	95210	89561	6427	840	283	6430
Maharashtra	2020-04-25	100912	94485	6817	957	301	6817
Maharashtra	2020-04-26	107979	101162	7928	1076	323	7628
Maharashtra	2020-04-27	115147	107519	8068	1188	342	8068
Maharashtra	2020-04-28	120620	112552	8590	1282	369	8590
Maharashtra	2020-04-29	128726	120136	9318	1388	400	9318
Maharashtra	2020-04-30	135694	126376	9915	1593	432	9915
Maharashtra	2020-05-01	144159	134244	10498	1773	459	10498
Maharashtra	2020-05-02	151085	140587	11506	1879	485	11506
Maharashtra	2020-05-03	159754	148248	12296	2000	521	12296
Maharashtra	2020-05-04	168374	156078	12974	2115	548	12974
Maharashtra	2020-05-05	175323	162349	14541	2465	583	14541

```
2020-06-13 01:17:05     Starting to launch local task to process map join;      maximum memory = 932184064
2020-06-13 01:17:09     Dump the side-table for tag: 0 with group count: 1921 into file: file:/tmp/cloudera/5b63d0c3-d709-49ab-a70d-1f177a5e13b2/hi
2_240_2009940658340037373-1/-local-10003/HashTable-Stage-3/MapJoin-mapfile20--.hashtable
2020-06-13 01:17:09     Uploaded 1 File to: file:/tmp/cloudera/5b63d0c3-d709-49ab-a70d-1f177a5e13b2/hive_2020-06-13_01-17-02_240_2009940658340037373
Table-Stage-3/MapJoin-mapfile20--.hashtable (77010 bytes)
2020-06-13 01:17:09     End of local task; Time Taken: 3.327 sec.
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
20/06/13 01:17:09 [HiveServer2-Background-Pool: Thread-90]: WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. I
```

t.state	t.date	t.total_samples	t.negative	t.positive	c.cured	c.deaths	c.confirmed
West Bengal	2020-05-06	30141	NULL	1456	364	140	1344
West Bengal	2020-04-01	659	568	37	6	3	37
West Bengal	2020-04-16	3811	NULL	231	42	7	231
West Bengal	2020-06-03	232225	NULL	6508	2410	335	6168
West Bengal	2020-04-12	2523	NULL	134	19	5	134
West Bengal	2020-04-09	1889	NULL	NULL	16	5	103
West Bengal	2020-05-02	20976	NULL	795	139	33	795
West Bengal	2020-04-30	16525	NULL	758	124	22	758
West Bengal	2020-05-20	111002	NULL	3103	1074	258	2961
West Bengal	2020-05-31	203751	NULL	5501	1970	309	5130
West Bengal	2020-04-27	12043	NULL	649	105	20	649
West Bengal	2020-05-13	57632	NULL	2290	612	198	2173
West Bengal	2020-04-23	7990	NULL	456	79	15	456
West Bengal	2020-06-10	297419	NULL	9320	3620	415	8905
West Bengal	2020-05-28	175769	NULL	4536	1578	289	4192
West Bengal	2020-05-17	85956	NULL	2677	872	232	2576
West Bengal	2020-06-08	280098	NULL	8613	3303	396	8187
West Bengal	2020-06-04	241831	NULL	6876	2580	345	6588
West Bengal	2020-05-29	185051	NULL	4813	1668	295	4536
West Bengal	2020-05-25	140049	NULL	3916	1339	272	3667
West Bengal	2020-05-21	115244	NULL	3197	1136	253	3183

```
1,849 rows selected (28.183 seconds)
```

We are getting only the matching records from both tables.

We can create a final consolidated table as follows:

CREATE TABLE covid_details AS
SELECT /*+ MAPJOIN(T)
***/T.state,T.date,T.total_samples,T.negative,T.positive,C.cured,C.deaths,C.confirmed FROM State_Testing_ORC T JOIN Covid_India_ORC C ON (C.state = T.state) AND (C.date = T.date);**

```
0: jdbc:hive2://> CREATE TABLE covid_details AS
. . . . . . . . > SELECT /*+ MAPJOIN(T)
. . . . . . . . > */T.state,T.date,T.total_samples,T.negative,T.positive,C.cured
,C.deaths,C.confirmed FROM State_Testing_ORC T JOIN Covid_India_ORC C ON (C.stat
e = T.state) AND (C.date = T.date);
```

See the count of records which are there in the resulting table:

select count(*) from covid_details;

```
+-------+--+
|  _c0  |
+-------+--+
| 1849  |
+-------+--+
```

Step 7:Analysis

Lets query the Maharashtra data and see the results:

SELECT * FROM covid_details WHERE state = 'Maharashtra';

```
0: jdbc:hive2://> select * from covid_details where state = 'Maharashtra';
OK
```

covid_details.state	covid_details.date	covid_details.total_samples	covid_details.negative	covid_details.positive	covid_details.cured	covid_details.deaths	covid_details.confirmed
Maharashtra	2020-04-19	66796	63476	3651	365	211	3651
Maharashtra	2020-06-02	484784	409178	70813	38108	2362	70813
Maharashtra	2020-05-05	175323	162349	14541	2465	583	14541
Maharashtra	2020-04-22	82304	77638	5220	722	251	5221
Maharashtra	2020-05-01	144158	134244	10498	1773	459	10498
Maharashtra	2020-04-26	107979	101162	7928	1876	323	7628
Maharashtra	2020-05-30	440661	380425	62228	20597	2098	62228
Maharashtra	2020-05-27	405828	345151	54758	16954	1792	54758
Maharashtra	2020-05-23	340932	299167	44582	12583	1517	44582
Maharashtra	2020-04-11	31841	30477	1761	188	110	1574
Maharashtra	2020-06-06	539099	451764	88229	35156	2849	88229
Maharashtra	2020-05-18	261815	228956	29100	6564	1068	29100
Maharashtra	2020-04-15	45142	42898	2690	259	178	
Maharashtra	2020-04-25	100912	94485	6817	957	381	6817
Maharashtra	2020-06-09	578294	485144	88528	40975	3169	88528
Maharashtra	2020-04-29	128726	120136	9318	1388	460	9318
Maharashtra	2020-05-04	168374	156078	12974	2115	548	12974
Maharashtra	2020-04-21	75838	71638	4676	572	232	4669
Maharashtra	2020-05-08	209477	183852	17974	3391	694	17974

```
66 rows selected (0.402 seconds)
```

We can see Maharashtra has very consistent data .The number of positive samples matches mostly with number of confirmed cases in Maharashtra,which should be the case ideally.So,Maharashtra did a consistent data collection with not much discrepancies between the testing data and covid cases data.

Query:

For every state,find the total number of confirmed cases reported and also total number of positive samples tested,in the entire duration of 2months, displaying the results starting with the state with the highest cases.

```
0: jdbc:hive2://> SELECT state,max(positive)as positive_cnt,max(confirmed) as confirmed_cnt
. . . . . . . . > FROM covid_details
. . . . . . . . > group by state
. . . . . . . . > order by confirmed_cnt desc;
```

Note : We are using max here as we know that data is cumulative as was mentioned in problem statement as well.

state	positive_cnt	confirmed_cnt
Maharashtra	90787	90787
Tamil Nadu	36841	34914
Delhi	32810	31309
Gujarat	21554	21014
Uttar Pradesh	11610	11335
Rajasthan	11600	11245
Madhya Pradesh	10049	9849
West Bengal	9328	8985
Karnataka	6041	5921
Bihar	5583	5459
Haryana	5438	5209
Andhra Pradesh	4126	5070
Jammu and Kashmir	4507	4346
Odisha	3250	3140
Assam	2937	2776
Punjab	2805	2719
Kerala	2162	2096
Uttarakhand	1560	1537
Telangana	1551	1454
Jharkhand	1423	1411
Chhattisgarh	1262	1240
Tripura	897	864
Himachal Pradesh	451	445
Goa	387	359
Chandigarh	328	323
Manipur	311	304
Nagaland	128	127
Puducherry	156	127
Ladakh	108	103

Arunachal Pradesh	61	57
Meghalaya	44	43
Mizoram	88	42
Andaman and Nicobar Islands	33	33
Dadra and Nagar Haveli	27	22
Sikkim	12	13

35 rows selected (61.329 seconds)

Even among all states we can see Maharashtra's data is most consistent.Their total number of positive samples and total number of confirmed cases match,as compared to other states.

More queries can be framed and executed on this consolidated table to get more insights.

****

5 Star Google Rated Big Data Course

LEARN FROM THE EXPERT

9108179578

Call for more details