Demonstrate word count on an input file using MapReduce Program.

**Procedure:**

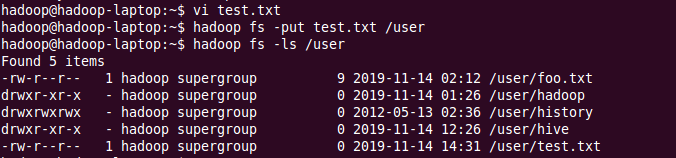
1.First create a file ‘test.txt’ and add any data into it.

Copy the file into HDFS by using the following command

Cmd: hadoop fs -put test.txt /user

Verify if the file is copied by using the following command

Cmd: hadoop fs -ls /user



2. Now create 3 .java files for Mapper, Reducer and main program wordcount .java



import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class Reduce extends Reducer<Text,IntWritable,Text,IntWritable>

{

private IntWritable result=new IntWritable();

public void reduce(Text key,Iterable<IntWritable>values,Context context)throws IOException,InterruptedException

{

int sum=0;

for(IntWritable val : values)

{

sum+=val.get();

}

result.set(sum);

context.write(key,result);

}

}



import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class Map extends Mapper<Object,Text,Text,IntWritable>{

private final IntWritable one = new IntWritable(1);

private Text word=new Text();

public void map(Object key,Text value,Context context)throws IOException,InterruptedException{

StringTokenizer itr=new StringTokenizer(value.toString());

while(itr.hasMoreTokens())

{

word.set(itr.nextToken());

context.write(word,one);

}

}

}



import java.io.IOException;

import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class wordcount

{

public static void main(String[] args)throws Exception

{

Configuration conf=new Configuration();

Job job=Job.getInstance(conf,"wordcount");

job.setJarByClass(wordcount.class);

job.setMapperClass(Map.class);

job.setCombinerClass(Reduce.class);

job.setReducerClass(Reduce.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

FileInputFormat.addInputPath(job,new Path(args[0]));

FileOutputFormat.setOutputPath(job,new Path(args[1]));

System.exit(job.waitForCompletion(true)? 0 : 1);

}

}

3. Now compile the above .java files by using the command:

Cmd :javac wordcount.java -cp $(hadoop classpath)

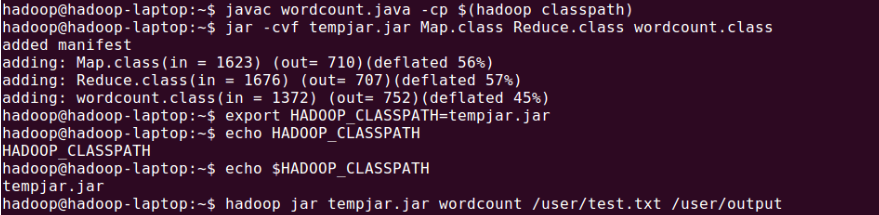
4. Create a jar file from the java files using the command:

Cmd: jar cvf temp.jar Map.class Reduce.class wordcount.class

5. Use the below commands to export and execute

export HADOOP\_CLASSPATH=tempjar.jar

echo $HADOOP\_CLASSPATH



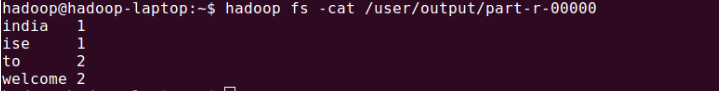
6. Use this command to start MapReduce execution:

hadoop jar tempjar.jar wordcount /user/test.txt /user/output



7. Command to view the output:

Hadoop fs -cat /user/output/part-r-00000



6. Create an external table(example: movieapp\_log). An external table is created for the conversion text file to binary file which can then be sent to HSFS through the operating system. Avro system is used for conversion of text file to binary file.

The command used is:

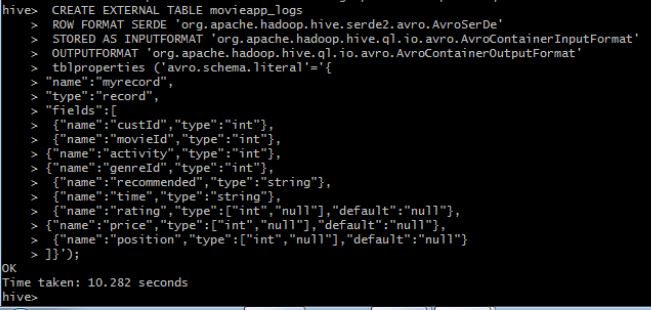
CREATE EXTERNAL TABLE movieapp\_logs

ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe'

STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.avro.AvroContainerInputFormat'

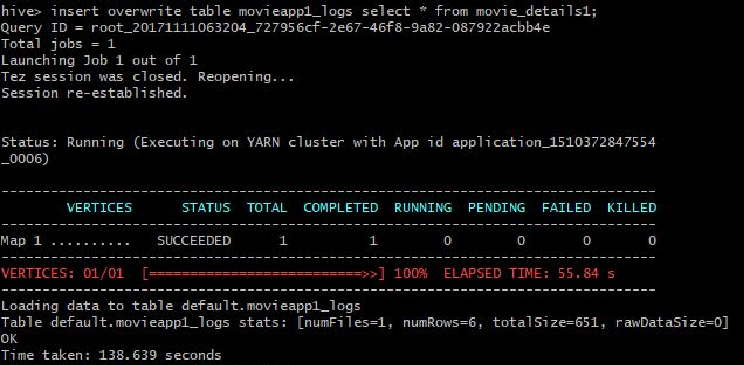
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.avro.AvroContainerOutputFormat'

tblproperties ('avro.schema.literal'='{"name": "my\_record", "type": "record", "fields": [ {"name":"custId", "type":"int"}, {"name":"movieId", "type":"int"}, {"name":"activity", "type":"int"}, {"name":"genereId", "type":"int"}, {"name":"recommended", "type":"string"}, {"name":"time", "type":"string"}, {"name":"rating","type":["int","null"],"default":"null"}, {"name":"price","type":["int","null"],"default":"null"}, {"name":"position","type":["int","null"],"default":"null"} ]}');



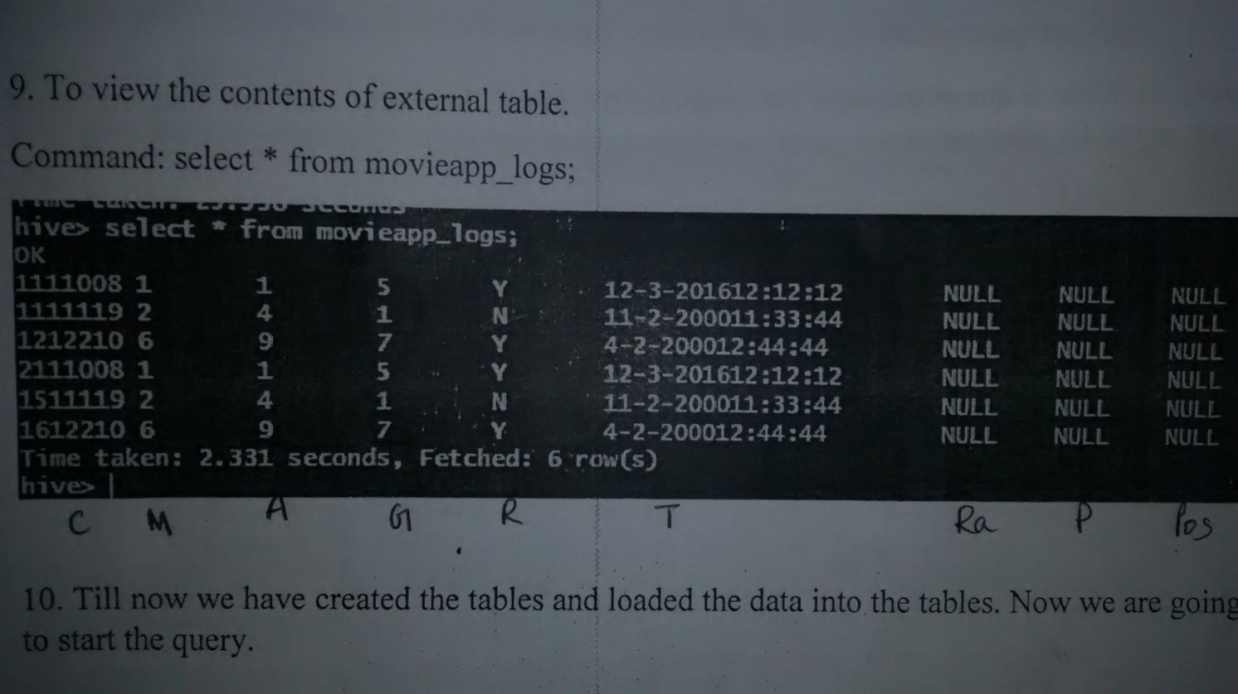
7. Now insert the details of internally created table(cmovie\_details) using insert overwrite command to the external table (loading the external table with the data)

Command: insert overwrite table movieapp\_logs select \* from cmovie\_details;



To view the contents of external table.

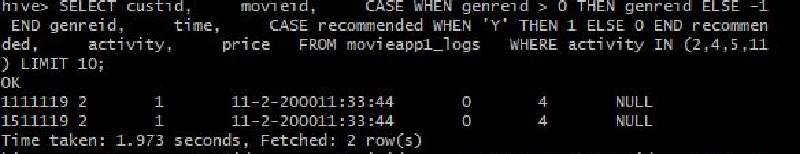
Command: select \* from movieapp\_logs;



Till now we have created the tables and loaded the data into the tables. Now we are going to start the query.

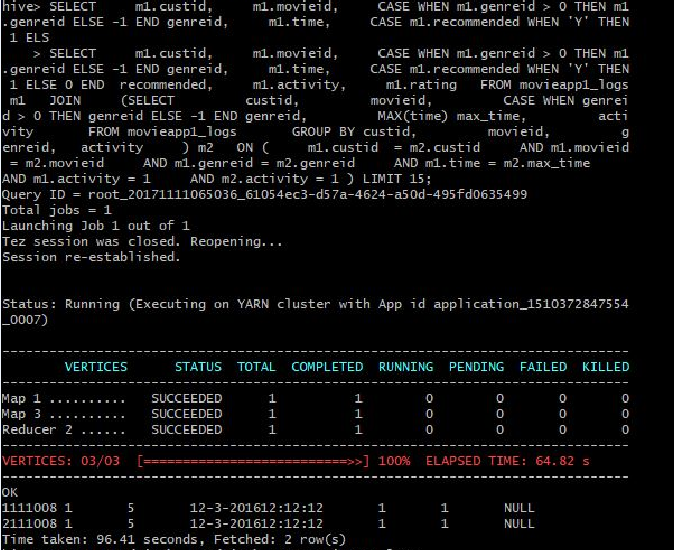
**Query 1:** Write a query to select only those clicks which correspond to starting, browsing, completing, or purchasing movies. Use a CASE statement to transform the RECOMMENDED column into integers where ‘Y’ is 1 and ‘N’ is 0. Also, ensure GENREID is not null. Only include the first 10 rows.

Command: SELECT custid, movieid, CASE WHEN genreid> 0 THEN genreid ELSE -1 END genreid, time, CASE recommended WHEN 'Y' THEN 1 ELSE 0 END recommended, activity, price FROM movieapp\_log WHERE activity IN (2,4,5,11) LIMIT 10;



**Query 2:** Write a query to select the customer ID, movie ID, recommended state and most recent rating for each movie.

Command: SELECT m1.custid, m1.movieid, CASE WHEN m1.genreid > 0 THEN m1.genreid ELSE -1 END genreid, m1.time, CASE m1.recommended WHEN 'Y' THEN 1 ELSE 0 END recommended, m1.activity, m1.rating FROM movieapp\_logs m1 JOIN (SELECT custid, movieid, CASE WHEN genreid> 0 THEN genreid ELSE -1 END genreid, MAX(time) max\_time, activity FROM movieapp\_logs GROUP BY custid, movieid, genreid, activity ) m2 ON ( m1.custid = m2.custid AND m1.movieid = m2.movieid AND m1.genreid = m2.genreid AND m1.time = m2.max\_time AND m1.activity = 1 AND m2.activity = 1 ) LIMIT 15;



**Program 6**

The moveapp\_log\_json table contains an activity column. Activity states are as follows:

* RATE\_MOVIE
* COMPLETED\_MOVIE
* PAUSE\_MOVIE
* START\_MOVIE
* BROWSE\_MOVIE
* LIST\_MOVIE
* SEARCH\_MOVIE
* LOGIN
* LOGOUT
* INCOMPLETE\_MOVIE

a. Load the results of the previous two queries into a staging table. First create the staging

table:

b. Next, load the results of the queries into the staging table.

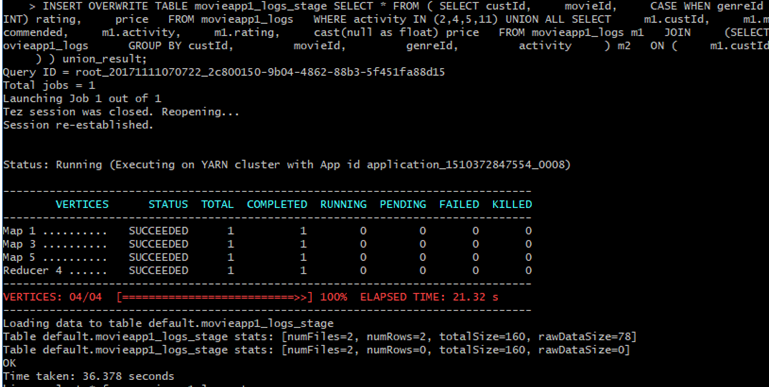
11. Load the results of the previous two queries into a staging table.First, create the staging table:

Command: CREATE TABLE movieapp\_logs\_stage ( custId INT, movieId INT, genreId INT, time STRING, recommended INT, activity INT, rating INT, sales FLOAT ) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

****

Next, load the results of the queries into the staging table:

Command**:** INSERT OVERWRITE TABLE movieapp1\_logs\_stage SELECT \* FROM ( SELECT custId, movieId, CASE WHEN genreId > 0 THEN genreId ELSE -1 END genreId, time, CAST((CASE recommended WHEN 'Y' THEN 1 ELSE 0 END) AS INT) recommended, activity, cast(null AS INT) rating, price FROM movieapp1\_logs WHERE activity IN (2,4,5,11) UNION ALL SELECT m1.custId, m1.movieId, CASE WHEN m1.genreId > 0 THEN m1.genreId ELSE -1 END genreId, m1.time, CAST((CASE m1.recommended WHEN 'Y' THEN 1 ELSE 0 END) AS INT) recommended, m1.activity, m1.rating, cast(null as float) price FROM movieapp1\_logs m1 JOIN (SELECT custId, movieId, CASE WHEN genreId > 0 THEN genreId ELSE -1 END genreid, MAX(time) max\_time, activity FROM movieapp1\_logs GROUP BY custId, movieId, genreId, activity ) m2 ON ( m1.custId = m2.custId AND m1.movieId = m2.movieId AND m1.genreId = m2.genreId AND m1.time = m2.max\_time AND m1.activity = 1 AND m2.activity = 1 ) ) union\_result;

****

12. Now view the staging table.

Command: select \* from movieapp1\_logs\_stage;

