Venkatesan Thirumalaisamy

t.venk8@gmail.com

Certification project

Airlines Analysis

Contents

[Solution to Problem Statement A 2](#_Toc523496270)

[Brief Description of the Solution 2](#_Toc523496271)

[MapReduce Code 2](#_Toc523496272)

[Command to execute the MapReduce Code 3](#_Toc523496273)

[Comparison of data shuffled with and without Compression enabled 3](#_Toc523496274)

[Solution to Problem Statement B 4](#_Toc523496275)

[Create Hive table for airline routes with Clustering 4](#_Toc523496276)

[Create Hive table for Airports with Clustering 5](#_Toc523496277)

[Load data into airlines and airline\_routes tables 6](#_Toc523496278)

[Find list of airports with zero stops with HiveQL 6](#_Toc523496279)

[Code Output (part of the data) in Hive Table 6](#_Toc523496280)

[Solution to Problem Statement C 6](#_Toc523496281)

[Spark SQL Code 6](#_Toc523496282)

[Code Output in HDFS (part of data in JSON Format) 7](#_Toc523496283)

[Solution to Problem Statement D 7](#_Toc523496284)

[Create Hive table for airports 7](#_Toc523496285)

[Load data into airports table 8](#_Toc523496286)

[Scala Spark Code 8](#_Toc523496287)

[Code Output 8](#_Toc523496288)

[Solution to Problem Statement E 8](#_Toc523496289)

[Pig Script to save the result to HDFS 8](#_Toc523496290)

[Command to execute the Pig script 9](#_Toc523496291)

[Code Output (part of data) in HDFS 9](#_Toc523496292)

# Solution to Problem Statement A

Find list of Airports operating in the Country India

## Brief Description of the Solution

For this solution I’m thinking of using Map Reduce Framework, as the approach doesn’t involve datasets joins and the logic is simple enough to be implemented in a Map Reduce program.

This would be a map only Map Reduce Program. Hence, no Reducer class is not specified in the Map Reduce Driver program.

Input Key Class – LongWritable (Offset of line from beginning of file)

Input Value Class – Text (line of text/record from the input file)

Output Key Class – VIntWritable (Airport ID)

Output Value Class – Text (Airport Name, City and Country – including City and Country for clarity)

I’m thinking of reducing the data transferred during the Shuffle phase by configuring Map output data compression. I’ve included snapshots below with and without the compression configured.

## MapReduce Code

**package** com.venkat.mapreduce;

**import** java.io.IOException;

**import** org.apache.hadoop.io.LongWritable;

**import** org.apache.hadoop.conf.Configuration;

**import** org.apache.hadoop.fs.Path;

**import** org.apache.hadoop.io.Text;

**import** org.apache.hadoop.io.VIntWritable;

**import** org.apache.hadoop.io.compress.CompressionCodec;

**import** org.apache.hadoop.io.compress.GzipCodec;

**import** org.apache.hadoop.mapreduce.Job;

**import** org.apache.hadoop.mapreduce.Mapper;

**import** org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

**import** org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

**public** **class** AirportsList {

**public** **static** **class** AirportsMapper **extends** Mapper<LongWritable, Text, VIntWritable, Text>

{

VIntWritable apId = **new** VIntWritable();

Text apNm = **new** Text();

String formattedAirportName;

**public** **void** map(LongWritable key, Text value, Context context) **throws** IOException, InterruptedException

{

//write dp logic

String[] tokens = value.toString().split(",");

**if**(tokens.length == 12 && tokens[3].trim().equalsIgnoreCase("INDIA"))

{

formattedAirportName = tokens[1].trim() + "," +

tokens[2].trim() + "," + tokens[3].trim();

apId.set(Integer.*parseInt*(tokens[0]));

apNm.set(formattedAirportName);

context.write(apId, apNm);

}

}

}

**public** **static** **void** main(String[] args) **throws** IOException, ClassNotFoundException, InterruptedException {

Configuration conf = **new** Configuration();

conf.setBoolean(Job.***MAP\_OUTPUT\_COMPRESS***, **true**);

conf.setClass(Job.***MAP\_OUTPUT\_COMPRESS\_CODEC***, GzipCodec.**class**, CompressionCodec.**class**);

Job job = Job.*getInstance*(conf, "Airports Count");

job.setMapperClass(AirportsMapper.**class**);

job.setJarByClass(AirportsList.**class**);

job.setMapOutputKeyClass(VIntWritable.**class**);

job.setMapOutputValueClass(Text.**class**);

job.setInputFormatClass(TextInputFormat.**class**);

job.setOutputFormatClass(TextOutputFormat.**class**);

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

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

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

}

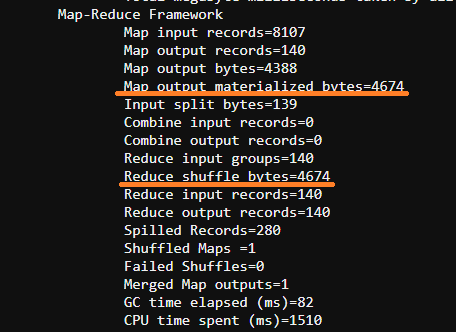
}

## Command to execute the MapReduce Code

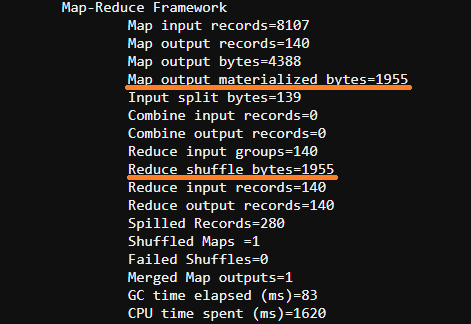
hadoop jar airlines/airportsList.jar com.venkat.mapreduce.AirportsList /user/edureka\_409125/datasets/airports /user/edureka\_409125/assignments/output/airports\_india\_list

## Comparison of data shuffled with and without Compression enabled

Without Compression



With Compression



# Solution to Problem Statement B

Find the list of Airlines having zero stops

## Create Hive table for airline routes with Clustering

CREATE TABLE airline\_routes

(

code string,

id int,

scode string,

sid int,

dcode string,

did int,

codeshare string,

stops int,

equipment string

)

CLUSTERED BY (id)

INTO 2 BUCKETS

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ',' ;

## Create Hive table for Airports with Clustering

CREATE TABLE airlines

(

id int,

name string,

alias string,

iata string,

icao string,

callsign string,

country string,

active string

)

CLUSTERED BY (id)

INTO 2 BUCKETS

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ',' ;

## Load data into airlines and airline\_routes tables

LOAD DATA LOCAL INPATH ' Final\_airlines' INTO TABLE airlines;

LOAD DATA LOCAL INPATH ' routes.dat' INTO TABLE airline\_routes;

## Find list of airports with zero stops with HiveQL

SELECT a.id, a.name FROM airline\_routes r

INNER JOIN airlines a ON a.id = r.id

WHERE r.stops = 0 GROUP BY a.id, a.name;

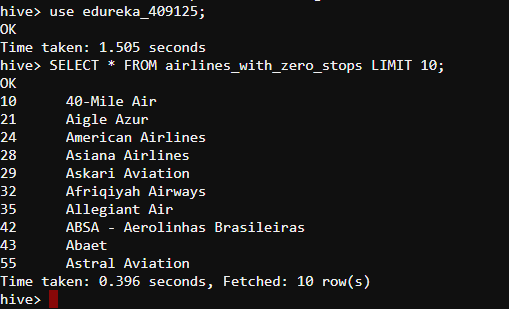
If we want to store the query result into a Hive table named *airlines\_with\_zero\_stops*, we can use the command below.

CREATE TABLE airlines\_with\_zero\_stops AS SELECT a.id, a.name FROM airline\_routes r

INNER JOIN airlines a ON a.id = r.id

WHERE r.stops = 0 GROUP BY a.id, a.name;

## Code Output (part of the data) in Hive Table



# Solution to Problem Statement C

List of Airlines operating with code share

We can use the Hive tables created for solving Problem Statement B above

## Spark SQL Code

//get id and name of the airline with code share using Spark SQL

val airlineWithCodeShare = spark.sql("SELECT a.id, a.name FROM edureka\_409125.airline\_routes r INNER JOIN edureka\_409125.airlines a ON a.id = r.id WHERE r.codeshare = 'Y' GROUP BY a.id, a.name")

//write the output data in parquet format in HDFS

airlineWithCodeShare.write.format("parquet").save("/user/edureka\_409125/assignments/output/airline\_with\_codeshare")

val codeshareresult = spark.read.format("parquet").load("/user/edureka\_409125/assignments/output/airline\_with\_codeshare")

codeshareresult.show()

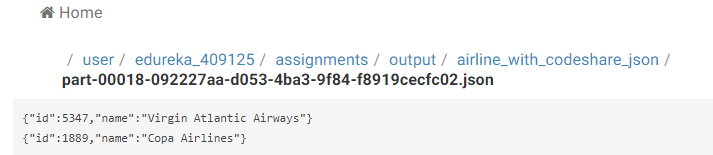
//if we want to write the data into json format in HDFS

airlineWithCodeShare.write.format("json").save("/user/edureka\_409125/assignments/output/airline\_with\_codeshare\_json")

val codeshareresult = spark.read.format("json").load("/user/edureka\_409125/assignments/output/airline\_with\_codeshare\_json")

codeshareresult.show

## Code Output in HDFS (part of data in JSON Format)



# Solution to Problem Statement D

Which country (or) territory having highest Airports

## Create Hive table for airports

CREATE TABLE airports

(

id int,

name string,

city string,

country string,

iata string,

icao string,

lat double,

lon double,

alt int,

tz float,

dst char(1),

dbtz string

)

CLUSTERED BY (id)

INTO 2 BUCKETS

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ',' ;

## Load data into airports table

LOAD DATA LOCAL INPATH 'airports\_mod.dat' INTO TABLE airports;

## Scala Spark Code

val ap = spark.table("edureka\_409125.airports")

val countryWithMaxAirports = ap.groupBy("country").count().sort($"count".desc).selectExpr("country").head()

## Code Output

scala> countryWithMaxAirports

res23: org.apache.spark.sql.Row = [United States]

# Solution to Problem Statement E

Find the list of Active Airlines in United state

## Pig Script to save the result to HDFS

airlines = LOAD 'datasets/airlines' USING PigStorage(',') AS (id:int, name:chararray, alias:chararray, iata:chararray, icao:chararray, callsign:chararray, country:chararray, active:chararray);

active\_airlines\_us = FILTER airlines BY UPPER(TRIM(country)) == 'UNITED STATES' AND UPPER(TRIM(active)) == 'Y';

airline\_names = FOREACH active\_airlines\_us GENERATE id, name;

STORE airline\_names INTO '/user/edureka\_409125/assignments/output/active\_airlines\_in\_us' USING PigStorage (',');

## Command to execute the Pig script

pig -f active\_airlines\_in\_us.pig

## Code Output (part of data) in HDFS

