**Experiment No.: 6**

**Compute Average Salary and Total Salary by Gender for an Enterprise**

To compute the **average salary** and **total salary by gender** for an enterprise using **Hadoop MapReduce**, we need to design a MapReduce program where:

1. The **Mapper** will process the input data (employee salary and gender), emitting key-value pairs, where the key is the gender, and the value is the salary of the employee.
2. The **Reducer** will aggregate the data based on gender, calculating the total salary and average salary for each gender.

**Problem Setup:**

Assume that the input data contains the following information for each employee:

* Employee ID
* Gender (Male/Female)
* Salary

For example:

EID Gender Salary

1 Male 5000

2 Female 6000

3 Male 7000

4 Female 8000

5 Male 5500

We want to calculate:

1. **Total salary by gender** (sum of all salaries for males and females).
2. **Average salary by gender** (average of all salaries for males and females).

### **Steps to Implement**

#### **1. Mapper (SalaryByGenderMapper.java)**

The mapper will emit key-value pairs where the key is the gender and the value is the salary of an employee.

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

import java.io.IOException;

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

private Text gender = new Text();

private IntWritable salary = new IntWritable();

@Override

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

// Parse the input line: assuming space-separated values [EID Gender Salary]

String[] tokens = value.toString().split("\\s+");

// Extract Gender and Salary

String empGender = tokens[1];

int empSalary = Integer.parseInt(tokens[2]);

// Emit the gender as the key and salary as the value

gender.set(empGender);

salary.set(empSalary);

context.write(gender, salary); // Emit (Gender, Salary)

}

}

**Explanation**:

* The **key** is the gender of the employee.
* The **value** is the salary of the employee.

#### **2. Reducer (SalaryByGenderReducer.java)**

The reducer will receive all the values (salaries) associated with each gender and compute the total salary and average salary.

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

import java.io.IOException;

public class SalaryByGenderReducer extends Reducer<Text, IntWritable, Text, Text> {

private Text result = new Text();

@Override

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

int totalSalary = 0;

int count = 0;

// Sum all the salaries for the given gender

for (IntWritable val : values) {

totalSalary += val.get();

count++;

}

// Calculate average salary

double averageSalary = (double) totalSalary / count;

// Format the result to show both total and average salary

result.set("Total Salary: " + totalSalary + ", Average Salary: " + String.format("%.2f", averageSalary));

context.write(key, result); // Emit (Gender, "Total Salary, Average Salary")

}

}

**Explanation**:

* The reducer processes each gender group, summing up all the salaries and counting the number of employees.
* It computes the **average salary** by dividing the **total salary** by the number of employees.
* It then emits the **total salary** and **average salary** for each gender.

#### **3. Driver Code (SalaryByGenderDriver.java)**

The driver code will set up the job by specifying the input and output paths, as well as the mapper and reducer classes.

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.lib.input.FileInputFormat;

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

public class SalaryByGenderDriver {

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

if (args.length != 2) {

System.err.println("Usage: SalaryByGender <input path> <output path>");

System.exit(-1);

}

Configuration conf = new Configuration();

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

// Set the jar class

job.setJarByClass(SalaryByGenderDriver.class);

// Set the Mapper and Reducer classes

job.setMapperClass(SalaryByGenderMapper.class);

job.setReducerClass(SalaryByGenderReducer.class);

// Set the output types of the job

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

// Set the input and output paths

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

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

// Wait for the job to complete

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

}

}

**Explanation**:

* The driver code specifies the input and output paths, as well as the mapper and reducer classes.
* It runs the MapReduce job to compute the total and average salaries by gender.

### **Input File Format**

The input file should contain space-separated values, each line representing an employee's ID, gender, and salary. For example:

1 Male 5000

2 Female 6000

3 Male 7000

4 Female 8000

5 Male 5500

### **Expected Output**

After running the job, the output will be written in the following format:

Female Total Salary: 14000, Average Salary: 7000.00

Male Total Salary: 17500, Average Salary: 5833.33

This output represents the total and average salaries for each gender (Male and Female).

### **Running the Program on Hadoop**

1. Prepare Input: Upload your input file to HDFS:

hdfs dfs -put employees.txt /user/hadoop/input/

1. Compile the Program: Compile the Java code into a JAR file:

javac -classpath `hadoop classpath` -d /path/to/compiled/classes SalaryByGenderMapper.java SalaryByGenderReducer.java SalaryByGenderDriver.java jar -cvf salarybygender.jar -C /path/to/compiled/classes .

1. Run the Job: Run the MapReduce job with the input and output directories:

hadoop jar salarybygender.jar SalaryByGenderDriver /user/hadoop/input /user/hadoop/output

1. View the Output: After the job completes, you can view the output:

hdfs dfs -cat /user/hadoop/output/part-r-00000

**Summary**

* The **Mapper** processes each line of the input file and emits the gender as the key and salary as the value.
* The **Reducer** aggregates the salaries by gender, computes the total salary and the average salary, and then writes the result.
* This MapReduce program allows you to efficiently calculate the total and average salaries by gender in a large dataset.