**H1B DATA ANALYSIS:**

**1.  
a) Is the number of petitions with Data Engineer job title increasing over time?**

* Technology used : HIVE
* hive query:

select year,count(year) from h1b\_final where job\_title=='DATA ENGINEER'group by year;

**OUTPUT:**

YEAR NO OF PETITION

2011 18

2012 32

2013 41

2014 89

2015 160

2016 251

**1.  
b) Find top 5 job titles who are having highest avg growth in applications.[ALL]**

* TECHNOLOGY USED: PIG
* PIG PROGRAM

register /usr/local/hive/lib/hive-exec-1.2.1.jar

register /usr/local/hive/lib/hive-common-1.2.1.jar

data1 = LOAD '/home/hduser/h1b\_final\_mapreduce/\*' USING PigStorage('\t') as (s\_no:double,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:double,year:chararray,worksite:chararray,longitude,latitude);

--dump data1;

part = foreach data1 generate $1,$4,$7,$8;

year1 = filter data1 by year=='2011';

A = group year1 by $4;

count\_A = foreach A generate group,COUNT($1);

part = foreach data1 generate $1,$4,$7,$8;

year1 = filter data1 by year=='2012';

A = group year1 by $4;

count\_B = foreach A generate group,COUNT($1);

part = foreach data1 generate $1,$4,$7,$8;

year1 = filter data1 by year=='2013';

A = group year1 by $4;

count\_C = foreach A generate group,COUNT($1);

part = foreach data1 generate $1,$4,$7,$8;

year1 = filter data1 by year=='2014';

A = group year1 by $4;

count\_D = foreach A generate group,COUNT($1);

part = foreach data1 generate $1,$4,$7,$8;

year1 = filter data1 by year=='2015';

A = group year1 by $4;

count\_E = foreach A generate group,COUNT($1);

part = foreach data1 generate $1,$4,$7,$8;

year1 = filter data1 by year=='2016';

A = group year1 by $4;

count\_F = foreach A generate group,COUNT($1);

joined = join count\_A by $0, count\_B by $0, count\_C by $0, count\_D by $0, count\_E by $0, count\_F by $0;

application = foreach joined generate $0,$1,$3,$5,$7,$9,$11;

growth = foreach application generate $0, (float) ($6-$5)\*100/$5, (float)($5-$4)\*100/$4, (float)($4-$3)\*100/$3, (float)($3-$2)\*100/$2,(float)($2-$1)\*100/$1;

Avg\_growth = foreach growth generate $0,($1+$2+$3+$4+$5)/5;

order\_Avg\_growth = order Avg\_growth by $1 desc;

final\_output = limit order\_Avg\_growth 5;

dump final\_output;

OUTPUT:

(SENIOR SYSTEMS ANALYST JC60,4255.4)

(SOFTWARE DEVELOPER 2,3480.8)

(PROJECT MANAGER 3,3233.4)

(SYSTEMS ANALYST JC65,2985.0)

(MODULE LEAD,2917.2)

**2.  
a) Which part of the US has the most Data Engineer jobs for each year?**

* TECHNOLOGY USED: PIG
* PIG PROGRAM

register /usr/local/hive/lib/hive-exec-1.2.1.jar

register /usr/local/hive/lib/hive-common-1.2.1.jar

data1 = LOAD '/home/hduser/h1b\_final\_mapreduce/\*' USING PigStorage('\t') as (s\_no:double,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:double,year:chararray,worksite:chararray,longitude,latitude);

dump data1;

part = foreach data1 generate $4,$7,$8;

year1 = filter data1 by year=='2011';

data\_engg\_job1 = filter year1 by job\_title matches '.\*DATA ENGINEER.\*';

A =group data\_engg\_job1 by worksite;

count\_A =foreach A generate group as area,COUNT(data\_engg\_job1)as no\_of\_job;

final\_11= order count\_A by no\_of\_job desc;

dump final\_11;

part = foreach data1 generate $4,$7,$8;

year1 = filter data1 by year=='2012';

data\_engg\_job1 = filter year1 by job\_title matches '.\*DATA ENGINEER.\*';

A =group data\_engg\_job1 by worksite;

count\_A =foreach A generate group as area,COUNT(data\_engg\_job1)as no\_of\_job;

final\_12= order count\_A by no\_of\_job desc;

dump final\_12;

**OUTPUT:**

SAN FRANCISCO, CALIFORNIA 2011 20

SAN FRANCISCO, CALIFORNIA 2012 30

MENLO PARK, CALIFORNIA 2013 45

MENLO PARK, CALIFORNIA 2014 46

SAN FRANCISCO, CALIFORNIA 2015 61

MENLO PARK, CALIFORNIA 2016 128

**2.  
b) find top 5 locations in the US who have got certified visa for each year.[certified**]

* TECHNOLOGY USED: HIVE
* HIVE QUERY

select worksite,count(case\_status) as A , year from h1b\_final where year ='2011' AND case\_status = 'CERTIFIED' group by worksite,year order by A desc limit 5;

select worksite,count(case\_status) as A , year from h1b\_final where year ='2012' AND case\_status = 'CERTIFIED' group by worksite,year order by A desc limit 5;

select worksite,count(case\_status) as A , year from h1b\_final where year ='2013' AND case\_status = 'CERTIFIED' group by worksite,year order by A desc limit 5;

select worksite,count(case\_status) as A , year from h1b\_final where year ='2014' AND case\_status = 'CERTIFIED' group by worksite,year order by A desc limit 5;

select worksite,count(case\_status) as A , year from h1b\_final where year ='2015' AND case\_status = 'CERTIFIED' group by worksite,year order by A desc limit 5;

select worksite,count(case\_status) as A , year from h1b\_final where year ='2016' AND case\_status = 'CERTIFIED' group by worksite,year order by A desc limit 5;

**OUTPUT:**

NEW YORK, NEW YORK 2011 23172

HOUSTON, TEXAS 2011 8184

CHICAGO, ILLINOIS 2011 5188

SAN JOSE, CALIFORNIA 2011 4713

SAN FRANCISCO, CALIFORNIA 2011 4711

NEW YORK, NEW YORK 2012 23737

HOUSTON, TEXAS 2012 9963

SAN FRANCISCO, CALIFORNIA 2012 6116

CHICAGO, ILLINOIS 2012 5671

ATLANTA, GEORGIA 2012 5565

NEW YORK, NEW YORK 2013 23537

HOUSTON, TEXAS 2013 11136

SAN FRANCISCO, CALIFORNIA 2013 7281

SAN JOSE, CALIFORNIA 2013 6722

ATLANTA, GEORGIA 2013 6377

NEW YORK, NEW YORK 2014 27634

HOUSTON, TEXAS 2014 13360

SAN FRANCISCO, CALIFORNISA 2014 9798

SAN JOSE, CALIFORNIA 2014 8223

ATLANTA, GEORGIA 2014 8213

NEW YORK, NEW YORK 2015 31266

HOUSTON, TEXAS 2015 15242

SAN FRANCISCO, CALIFORNIA 2015 12594

ATLANTA, GEORGIA 2015 10500

SAN JOSE, CALIFORNIA 2015 9589

NEW YORK, NEW YORK 2016 34639

SAN FRANCISCO, CALIFORNIA 2016 13836

HOUSTON, TEXAS 2016 13655

ATLANTA, GEORGIA 2016 11678

CHICAGO, ILLINOIS 2016 11064

**3.Which industry(SOC\_NAME) has the most number of Data Scientist positions.[certified]**

* TECHNOLOGY USED: MAPREDUCE
* MAPREDUCE PROGRAM:

package question3;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.NullWritable;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import java.io.IOException;

import java.util.TreeMap;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

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

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

public class DESOC

{

public static class DESOCMapper extends Mapper < LongWritable, Text, Text, LongWritable >

{

LongWritable one = new LongWritable(1);

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

{

if (key.get() > 0)

{

String[] token = values.toString().split("\t");

if (token[4].contains("DATA SCIENTIST") && token[1].equals("CERTIFIED"))

{

Text answer = new Text(token[3]);

context.write(answer, one);

}

}

}

}

public static class DESOCReducer extends Reducer < Text, LongWritable, NullWritable, Text >

{

private TreeMap < LongWritable,

Text > DataScientistJobs = new TreeMap < LongWritable,Text > ();

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

{

long sum = 0;

for (LongWritable val: values)

sum += val.get();

DataScientistJobs.put(new LongWritable(sum), new Text(key.toString().replaceAll("\"", "") + "," + sum));

if (DataScientistJobs.size() > 5)

{

DataScientistJobs.remove(DataScientistJobs.firstKey());

}

}

protected void cleanup(Context context) throws IOException,

InterruptedException {

for (Text t: DataScientistJobs.descendingMap().values())

{

context.write(NullWritable.get(), t);

}

}

}

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

Configuration conf = new Configuration();

Job job = new Job(conf, "Data Scientist jobs");

job.setJarByClass(DESOC.class);

job.setMapperClass(DESOCMapper.class);

job.setReducerClass(DESOCReducer.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(LongWritable.class);

job.setOutputKeyClass(NullWritable.class);

job.setOutputValueClass(Text.class);

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

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

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

}

}

**OUTPUT:**

STATISTICIANS 572

COMPUTER AND INFORMATION RESEARCH SCIENTISTS 419

OPERATIONS RESEARCH ANALYSTS 380

COMPUTER AND INFORMATION RESEARCH SCIENTISTS 181

COMPUTER OCCUPATIONS ALL OTHER 160

**4.Which top 5 employers file the most petitions each year? - Case Status – ALL.**

* TECHNOLOGY USED: MAPREDUCE
* MAPREDUCE PROGRAM:

package query4;

import java.io.IOException;

import java.util.TreeMap;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.NullWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Partitioner;

import org.apache.hadoop.mapreduce.Reducer;

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

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

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

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

public class DataEngineerGrowth

{

public static class DataEngineerMapper extends Mapper<LongWritable,Text,Text,LongWritable>

{

LongWritable one =new LongWritable(1);

public void map(LongWritable key,Text values,Context context) throws IOException, InterruptedException,ArrayIndexOutOfBoundsException

{

try

{

if(key.get()> 0)

{

String[] token=values.toString().split("\t");

if (!token[1].equals("NA") && token[1] != null && !token[2].equals("NA") && token[2] != null && !token[7].equals("NA") && token[7] != null)

{ Text answer= new Text(token[2]+'\t'+token[7]);

context.write(answer,one);

}

}

}

catch (ArrayIndexOutOfBoundsException e)

{}

}

}

public static class DataEngineerReducer extends Reducer < Text, LongWritable, NullWritable, Text >

{

private TreeMap < LongWritable,Text > Top5Employers = new TreeMap < LongWritable,

Text > ();

long sum = 0;

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

InterruptedException {

sum = 0;

for (LongWritable val: values) {

sum += val.get();

}

Top5Employers.put(new LongWritable(sum), new Text(key + "," + sum));

if (Top5Employers.size() > 5)

Top5Employers.remove(Top5Employers.firstKey());

}

protected void cleanup(Context context) throws IOException,

InterruptedException {

for (Text t: Top5Employers.descendingMap().values())

context.write(NullWritable.get(), t);

}

}

public static class DataEngineerPartitioner extends Partitioner < Text, LongWritable >

{

public int getPartition(Text key, LongWritable value, int numReduceTasks)

{

String[] str = key.toString().split("\t");

if (str[1].equals("2011"))

{

return 0;

}

else if (str[1].equals("2012"))

{

return 1;

}

else if (str[1].equals("2013"))

{

return 2;

}

else if (str[1].equals("2014"))

{

return 3;

}

else if (str[1].equals("2015"))

{

return 4;

}

else if (str[1].equals("2016"))

{

return 5;

}

else

{

return 6;

}

}

}

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

{

Configuration conf= new Configuration();

Job job= new Job(conf,"Question 4");

job.setJarByClass(DataEngineerGrowth.class);

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

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

job.setMapperClass(DataEngineerMapper.class);

job.setPartitionerClass(DataEngineerPartitioner.class);

job.setReducerClass(DataEngineerReducer.class);

job.setNumReduceTasks(7);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(LongWritable.class);

job.setOutputKeyClass(NullWritable.class);

job.setOutputValueClass(Text.class);

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

}

}

**OUTPUT:**

TATA CONSULTANCY SERVICES LIMITED 2011 5416

MICROSOFT CORPORATION 2011 4253

DELOITTE CONSULTING LLP 2011 3621

WIPRO LIMITED 2011 3028

COGNIZANT TECHNOLOGY SOLUTIONS U.S. CORPORATION 2011 2721

INFOSYS LIMITED 2012 15818

WIPRO LIMITED 2012 7182

TATA CONSULTANCY SERVICES LIMITED 2012 6735

DELOITTE CONSULTING LLP 2012 4727

IBM INDIA PRIVATE LIMITED 2012 4074

INFOSYS LIMITED 2013 32223

TATA CONSULTANCY SERVICES LIMITED 2013 8790

WIPRO LIMITED 2013 6734

DELOITTE CONSULTING LLP 2013 6124

ACCENTURE LLP 2013 4994

INFOSYS LIMITED 2014 23759

TATA CONSULTANCY SERVICES LIMITED 2014 14098

WIPRO LIMITED 2014 8365

DELOITTE CONSULTING LLP 2014 7017

ACCENTURE LLP 2014 5498

INFOSYS LIMITED 2015 33245

TATA CONSULTANCY SERVICES LIMITED 2015 16553

WIPRO LIMITED 2015 12201

IBM INDIA PRIVATE LIMITED 2015 10693

ACCENTURE LLP 2015 9605

INFOSYS LIMITED 2016 25352

CAPGEMINI AMERICA INC 2016 16725

TATA CONSULTANCY SERVICES LIMITED 2016 13134

WIPRO LIMITED 2016 10607

IBM INDIA PRIVATE LIMITED 2016 9787

**5.Find the most popular top 10 job positions for H1B visa applications for each year?  
 a) for all the applications.**

* TECHNOLOGY USED: HIVE
* HIVE QUERY:

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2011'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2012'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2013'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2014'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2015'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2016'group by year,job\_title order by total desc limit 10;

**OUTPUT:**

PROGRAMMER ANALYST 2011 31799

SOFTWARE ENGINEER 2011 12763

COMPUTER PROGRAMMER 2011 8998

SYSTEMS ANALYST 2011 8644

BUSINESS ANALYST 2011 3891

COMPUTER SYSTEMS ANALYST 2011 3698

ASSISTANT PROFESSOR 2011 3467

PHYSICAL THERAPIST 2011 3377

SENIOR SOFTWARE ENGINEER 2011 2935

SENIOR CONSULTANT 2011 2798

PROGRAMMER ANALYST 2012 33066

SOFTWARE ENGINEER 2012 14437

COMPUTER PROGRAMMER 2012 9629

SYSTEMS ANALYST 2012 9296

BUSINESS ANALYST 2012 4752

COMPUTER SYSTEMS ANALYST 2012 4706

SOFTWARE DEVELOPER 2012 3895

PHYSICAL THERAPIST 2012 3871

ASSISTANT PROFESSOR 2012 3801

PROGRAMMER ANALYST 2013 33880

SOFTWARE ENGINEER 2013 15680

COMPUTER PROGRAMMER 2013 11271

SYSTEMS ANALYST 2013 8714

TECHNOLOGY LEAD - US 2013 7853

TECHNOLOGY ANALYST – US 2013 7683

BUSINESS ANALYST 2013 5716

COMPUTER SYSTEMS ANALYST 2013 5043

SOFTWARE DEVELOPER 2013 5026

SENIOR CONSULTANT 2013 4326

PROGRAMMER ANALYST 2014 43114

SOFTWARE ENGINEER 2014 20500

COMPUTER PROGRAMMER 2014 14950

SYSTEMS ANALYST 2014 10194

SOFTWARE DEVELOPER 2014 7337

BUSINESS ANALYST 2014 7302

COMPUTER SYSTEMS ANALYST 2014 6821

TECHNOLOGY LEAD - US 2014 5057

TECHNOLOGY ANALYST – US 2014 4913

SENIOR CONSULTANT 2014 4898

PROGRAMMER ANALYST 2015 53436

SOFTWARE ENGINEER 2015 27259

COMPUTER PROGRAMMER 2015 14054

SYSTEMS ANALYST 2015 12803

SOFTWARE DEVELOPER 2015 10441

BUSINESS ANALYST 2015 8853

TECHNOLOGY LEAD - US 2015 8242

COMPUTER SYSTEMS ANALYST 2015 7918

TECHNOLOGY ANALYST – US 2015 7014

SENIOR SOFTWARE ENGINEER 2015 6013

PROGRAMMER ANALYST 2016 53743

SOFTWARE ENGINEER 2016 30668

SOFTWARE DEVELOPER 2016 14041

SYSTEMS ANALYST 2016 12314

COMPUTER PROGRAMMER 2016 11668

BUSINESS ANALYST 2016 9167

COMPUTER SYSTEMS ANALYST 2016 6900

SENIOR SOFTWARE ENGINEER 2016 6439

DEVELOPER 2016 6084

TECHNOLOGY LEAD - US 2016 5410

**5.Find the most popular top 10 job positions for H1B visa applications for each year?  
 b) for only certified applications.**

* TECHNOLOGY USED: HIVE
* HIVE QUERY:

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2011'AND case\_status='CERTIFIED'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2012'AND case\_status='CERTIFIED'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2013'AND case\_status='CERTIFIED'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2014'AND case\_status='CERTIFIED'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2015'AND case\_status='CERTIFIED'group by year,job\_title order by total desc limit 10;

select job\_title,year,count(job\_title) as total from h1b\_final where year = '2016'AND case\_status='CERTIFIED'group by year,job\_title order by total desc limit 10;

**OUTPUT:**

PROGRAMMER ANALYST 2011 28806

SOFTWARE ENGINEER 2011 11224

COMPUTER PROGRAMMER 2011 8038

SYSTEMS ANALYST 2011 7850

BUSINESS ANALYST 2011 3444

COMPUTER SYSTEMS ANALYST 2011 3152

ASSISTANT PROFESSOR 2011 3050

PHYSICAL THERAPIST 2011 2911

SENIOR SOFTWARE ENGINEER 2011 2595

SENIOR CONSULTANT 2011 2585

PROGRAMMER ANALYST 2012 29226

SOFTWARE ENGINEER 2012 12273

COMPUTER PROGRAMMER 2012 8483

SYSTEMS ANALYST 2012 8399

BUSINESS ANALYST 2012 4144

COMPUTER SYSTEMS ANALYST 2012 4084

SENIOR CONSULTANT 2012 3420

SOFTWARE DEVELOPER 2012 3290

PHYSICAL THERAPIST 2012 3284

ASSISTANT PROFESSOR 2012 3033

PROGRAMMER ANALYST 2013 29906

SOFTWARE ENGINEER 2013 12973

COMPUTER PROGRAMMER 2013 10202

SYSTEMS ANALYST 2013 7850

TECHNOLOGY LEAD - US 2013 7809

TECHNOLOGY ANALYST – US 2013 7641

BUSINESS ANALYST 2013 4993

COMPUTER SYSTEMS ANALYST 2013 4554

SOFTWARE DEVELOPER 2013 4316

PROGRAMMER ANALYST 2014 38625

SOFTWARE ENGINEER 2014 17278

COMPUTER PROGRAMMER 2014 13796

SYSTEMS ANALYST 2014 9161

BUSINESS ANALYST 2014 6529

SOFTWARE DEVELOPER 2014 6473

COMPUTER SYSTEMS ANALYST 2014 6204

TECHNOLOGY LEAD - US 2014 5055

TECHNOLOGY ANALYST – US 2014 4911

SENIOR CONSULTANT 2014 4535

PROGRAMMER ANALYST 2015 48203

SOFTWARE ENGINEER 2015 23352

COMPUTER PROGRAMMER 2015 12971

SYSTEMS ANALYST 2015 11498

SOFTWARE DEVELOPER 2015 9343

TECHNOLOGY LEAD - US 2015 8238

BUSINESS ANALYST 2015 7919

COMPUTER SYSTEMS ANALYST 2015 7234

TECHNOLOGY ANALYST – US 2015 7009

SENIOR SOFTWARE ENGINEER 2015 5324

PROGRAMMER ANALYST 2016 47964

SOFTWARE ENGINEER 2016 25890

SOFTWARE DEVELOPER 2016 12474

SYSTEMS ANALYST 2016 10986

COMPUTER PROGRAMMER 2016 10528

BUSINESS ANALYST 2016 8175

COMPUTER SYSTEMS ANALYST 2016 6205

DEVELOPER 2016 5912

SENIOR SOFTWARE ENGINEER 2016 5630

TECHNOLOGY LEAD - US 2016 5405

**6.Find the percentage and the count of each case status on total applications for each year. Create a line graph depicting the pattern of All the cases over the period of time.**

* TECHNOLOGY USED: MAPREDUCE
* MAPREDUCE PROGRAM:

import java.io.\*;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

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

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

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

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

public class MyQuery6

{

public static class MapClass extends Mapper<LongWritable,Text,Text,Text>

{

public void map(LongWritable key, Text value, Context context)

{

try{

String[] str = value.toString().split("\t");

String year = str[7];

String case\_status = str[1];

context.write(new Text(year),new Text(case\_status));

}

catch(Exception e)

{

System.*out*.println(e.getMessage());

}

}

}

public static class ReduceClass extends Reducer<Text,Text,Text,Text>

{

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

long totalcount= 0,certified\_count=0,certified\_withdrawn\_count=0,denied\_count=0,withdrawn\_count=0;

double certified\_AvgPerc=0,certified\_withdrawn\_AvgPerc=0,denied\_AvgPerc=0,withdrawn\_AvgPerc=0;

for (Text T : values)

{

totalcount++;

String case\_status=T.toString();

if(case\_status.equals("CERTIFIED"))

{

certified\_count++;

}

else if(case\_status.equals("CERTIFIED-WITHDRAWN"))

{

certified\_withdrawn\_count++;

}

else if(case\_status.equals("WITHDRAWN"))

{

withdrawn\_count++;

}

else

{

denied\_count++;

}

}

certified\_AvgPerc = ((double)certified\_count/(double)totalcount)\*100;

certified\_withdrawn\_AvgPerc = ((double)certified\_withdrawn\_count/(double)totalcount)\*100;

withdrawn\_AvgPerc = ((double)withdrawn\_count/(double)totalcount)\*100;

denied\_AvgPerc = ((double)denied\_count/(double)totalcount)\*100;

String COUNT=totalcount+"\t"+certified\_count+"\t"+certified\_AvgPerc+"\t"+certified\_withdrawn\_count+"\t"

+certified\_withdrawn\_AvgPerc+"\t"+withdrawn\_count+"\t"+withdrawn\_AvgPerc+"\t"+denied\_count+"\t"+ denied\_AvgPerc;

context.write(key,new Text(COUNT));

}

}

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

Configuration conf = new Configuration();

Job job = new Job(conf, "H1B DATA");

job.setJarByClass(MyQuery6.class);

job.setMapperClass(MapClass.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

job.setReducerClass(ReduceClass.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(Text.class);

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

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

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

}

}

**OUTPUT:**

2011 358767.0 307936.0 85.83175152675692 11596.0 3.2321813321738064 10105.0 2.816591269542628 29130.0 8.119475871526646

2012 415607.0 352668.0 84.85612609989725 31118.0 7.487361858678993 10725.0 2.5805628875355806 21096.0 5.0759491538881685

2013 442114.0 382951.0 86.61815730784367 35432.0 8.014222576077664 11590.0 2.621495813297023 12141.0 2.7461243027816353

2014 519427.0 455144.0 87.62424748809747 36350.0 6.9980959788382195 16034.0 3.086863024063054 11899.0 2.290793509001265

2015 618727.0 547278.0 88.45225761927313 41071.0 6.637984119005636 19455.0 3.144359305477214 10923.0 1.76539895624403

2016 647803.0 569646.0 87.93506667922192 47092.0 7.269493966529948 21890.0 3.3791137120389996 9175.0 1.4163256422091284

**GRAPH:**

**7.Create a bar graph to depict the number of applications for each year [All]**

* TECHNOLOGY USED: MAPREDUCE
* MAPREDUCE PROGRAM:

import java.io.\*;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.Reducer.Context;

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

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

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

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

public class MyQuery7

{

public static class mymap extends Mapper<LongWritable,Text, LongWritable,Text>

{

public void map (LongWritable inkey,Text val,Context context) throws InterruptedException,IOException

{

String[] str= val.toString().split("\t");

String year = str[7];

long s\_no = Long.*parseLong*(str[0]);

//LongWritable outkey = new LongWritable(1);//Text outkey=new Text("1");

//long year = Long.parseLong(str[2]);

context.write(new LongWritable(s\_no),new Text( str[7]));

}

}

public static class myreduce extends Reducer<LongWritable,Text,Text,LongWritable>

{

public void reduce (Text inkey,Iterable<LongWritable>val,Context context)throws InterruptedException,IOException

{

long count=0;

for(LongWritable v:val)

{

count++;

}

context.write(inkey,new LongWritable(count));

}

}

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

{

Configuration conf = new Configuration();

Job job = new Job(conf, "H1B\_data");

job.setJarByClass(MyQuery7.class);

job.setMapperClass(mymap.class);

job.setReducerClass(myreduce.class);

job.setNumReduceTasks(1);

job.setMapOutputKeyClass(LongWritable.class);

job.setMapOutputValueClass(Text.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(LongWritable.class);

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

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

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

}

}

**OUTPUT:**

2011 358767

2012 415607

2013 442114

2014 519427

2015 618727

2016 647803

**8. Find the average Prevailing Wage for each Job for each Year (take part time and full time separate).Arrange the output in descending order - [Certified and Certified Withdrawn]**

* TECHNOLOGY USED: HIVE
* HIVE QUERY:

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='Y' and year='2011' and case\_status = 'CERTIFIED' group by job\_title,full\_time\_position,case\_status,year order by average desc;

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='N' and year='2011' and case\_status = 'CERTIFIED\_WITHDRAWN' group by job\_title,full\_time\_position,case\_status,year order by average desc;

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='Y' and year='2012' and case\_status = 'CERTIFIED' group by job\_title,full\_time\_position,case\_status,year order by average desc;

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='N' and year='2012' and case\_status = 'CERTIFIED\_WITHDRAWN' group by job\_title,full\_time\_position,case\_status,year order by average desc;

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='Y' and year='2013' and case\_status = 'CERTIFIED' group by job\_title,full\_time\_position,case\_status,year order by average desc;

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='N' and year='2013' and case\_status = 'CERTIFIED\_WITHDRAWN' group by job\_title,full\_time\_position,case\_status,year order by average desc;

4)select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='Y' and year='2014' and case\_status = 'CERTIFIED' group by job\_title,full\_time\_position,case\_status,year order by average desc;

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='N' and year='2014' and case\_status = 'CERTIFIED\_WITHDRAWN' group by job\_title,full\_time\_position,case\_status,year order by average desc;

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='Y' and year='2015' and case\_status = 'CERTIFIED' group by job\_title,full\_time\_position,case\_status,year order by average desc;

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='N' and year='2015' and case\_status = 'CERTIFIED\_WITHDRAWN' group by job\_title,full\_time\_position,case\_status,year order by average desc;

select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='Y' and year='2016' and case\_status = 'CERTIFIED' group by job\_title,full\_time\_position,case\_status,year order by average desc;

Select job\_title,full\_time\_position,case\_status,year,avg(prevailing\_wage)as average from h1b\_final where full\_time\_position ='N' and year='2016' and case\_status = 'CERTIFIED\_WITHDRAWN' group by job\_title,full\_time\_position,case\_status,year order by average desc;

**OUTPUT:**

(JAPANESE TEACHING ASSISTANT,2013,N),17076.0

(INSTRUCTOR, PSYCHOLOGY,2013,N),16972.0

(ASSISTANT VOLLEYBALL COACH,2013,N),16889.0

(DIRECTOR OF RELIGIOUS ACTIVITIES AND EDUCATION,2013,N),16764.0

(COSTUMER,2013,N),16640.0

(ASSISTANT WOMEN'S SOCCER COACH,2013,N),16120.0

(BILINGUAL SPECIAL EDUCATION TEACHER ASSISTANT,2013,N),15080.0

**9.Which are the employers along with the number of petitions who have the success rate more than 70% in petitions. (total petitions filed 1000 OR more than 1000) ?**

* TECHNOLOGY USED: PIG
* PIG PROGRAM:

register /usr/local/hive/lib/hive-exec-1.2.1.jar

register /usr/local/hive/lib/hive-common-1.2.1.jar

data1 = LOAD '/home/hduser/h1b\_final\_mapreduce/\*' USING PigStorage('\t') as (s\_no:double,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:double,year:chararray,worksite:chararray,longitude,latitude);

--dump data1;

clean = filter data1 by $1!='NA';

temp= group clean by $2;

total= foreach temp generate group,COUNT(clean.$1); --Group by employername and count the case statuses as a whole

certified= filter data1 by $1 == 'CERTIFIED';

temp1= group certified by $2;

totalcertified= foreach temp1 generate group,COUNT(certified.$1); --Group by employername and count the case status = 'CERTIFIED'

certified= filter data1 by $1 == 'CERTIFIED-WITHDRAWN';

temp2= group certified by $2;

totalcertifiedwithdrawn= foreach temp2 generate group,COUNT(certified.$1); --Group by employername and count the case status = 'CERTIFIED-WITHDRAWN'

joined= join totalcertified by $0,totalcertifiedwithdrawn by $0,total by $0;

--dump joined;

joined= foreach joined generate $0,$1,$3,$5;

intermediateoutput= foreach joined generate $0,(float)($1+$2)\*100/($3),$3;

intermediateoutput2= filter intermediateoutput by $1>70 and $2>1000; --Filter by success-rate greater than 70% and petition count above 1000

finaloutput= order intermediateoutput2 by $1 DESC;

finaloutput1 = limit finaloutput 10;

dump finaloutput1;

**output:**

(INFOSYS LIMITED,99.54055,130592)

(ACCENTURE LLP,99.39307,33447)

(TATA CONSULTANCY SERVICES LIMITED,99.337204,64726)

(HCL AMERICA, INC.,99.26801,22678)

(RELIABLE SOFTWARE RESOURCES, INC.,99.14658,1992)

(NTT DATA, INC.,99.13251,4611)

(ERP ANALYSTS, INC.,99.10364,1785)

(PATNI AMERICAS INC.,99.07907,3149)

(KFORCE INC.,99.06015,1596)

(GENPACT LLC,98.852776,1046)

**10. Which are the job positions along with the number of petitions which have the success rate more than 70% in petitions (total petitions filed 1000 OR more than 1000)?**

* TECHNOLOGY USED: PIG
* PIG PROGRAM:

register /usr/local/hive/lib/hive-exec-1.2.1.jar

register /usr/local/hive/lib/hive-common-1.2.1.jar

data1 = LOAD '/home/hduser/h1b\_final\_mapreduce/\*' USING PigStorage('\t') as (s\_no:double,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:double,year:chararray,worksite:chararray,longitude,latitude);

--dump data1;

cleansed= filter data1 by $1 is not null and $1!='NA';

temp= group cleansed by $4;

total= foreach temp generate group,COUNT(cleansed.$1);

--Count Total Applications who are 'CERTIFIED'

certified= filter data1 by $1 == 'CERTIFIED';

temp1= group certified by $4;

totalcertified= foreach temp1 generate group,COUNT(certified.$1);

--Count Total Applications who are 'CERTIFIED-WITHDRAWN'

certified= filter data1 by $1 == 'CERTIFIED-WITHDRAWN';

temp2= group certified by $4;

totalcertifiedwithdrawn= foreach temp2 generate group,COUNT(certified.$1);

--SUCCESS\_RATE=(CERTIFIED+CERTIFIED-WITHDRAWN)/TOTAL X 100

joined= join totalcertified by $0,totalcertifiedwithdrawn by $0,total by $0;

joined= foreach joined generate $0,$1,$3,$5;

intermediateoutput= foreach joined generate $0,(float)($1+$2)\*100/($3),$3;

intermediateoutput2= filter intermediateoutput by $1>70 and $2>1000; --Filter by success-rate greater than 70% and petition count above 1000

intermediateoutput3= order intermediateoutput2 by $1 DESC;

finaloutput= limit intermediateoutput3 10;

dump finaloutput; --ANSWER

--STORE DATA INTO TEXT FILE

--store finaloutput into '/home/mohith/Pig/question10' using PigStorage('\t');

**Output:**

(COMPUTER PROGRAMMER / CONFIGURER 2,100.0,1276)

(ASSOCIATE CONSULTANT - US,99.93171,4393)

(SYSTEMS ENGINEER - US,99.90036,10036)

(TEST ANALYST - US,99.818474,4958)

(CONSULTANT - US,99.81147,7426)

(TECHNOLOGY LEAD - US,99.80247,28350)

(TECHNICAL TEST LEAD - US,99.79531,5374)

(TECHNOLOGY ARCHITECT - US,99.766304,4707)

(TECHNOLOGY ANALYST - US,99.76204,26055)

(SENIOR PROJECT MANAGER - US,99.74766,2774)

**11. Export result for question no 10 to MySql database**

* TECHNOLOGY USED: SQOOP

1.mysql –u root –p

2. create database h1b\_FINAL;

3. use h1b\_FINAL;

4.CREATE TABLE suc\_rate(job\_title varchar(100)NOT NULL,total\_no\_of\_appl INT NOT NULL,certifiedANDcertified\_withdrwan\_count INT NOT NULL,

5.desc suc\_rate;

+---------------------------------------+--------------+------+-----+---------+-------+

| field | type | null | key | default | extra |

+---------------------------------------+--------------+------+-----+---------+-------+

| job\_title | varchar(100) | no | | null | |

| total\_no\_of\_appl | int(11) | no | | null | |

| certifiedandcertified\_withdrwan\_count | int(11) | no | | null | |

| success\_rate | float | no | | null | |

+---------------------------------------+--------------+------+-----+---------+-------+

6.Start sqoop…..connect to msql>h1b database

sqoop list-tables --connect jdbc:mysql://localhost/h1b --username root --password 'ROOT'

7. export data from hdfs to msql >success\_rate table

sqoop export --connect jdbc:mysql://localhost/h1b --username 'root' --password 'hduser' --table success\_rate --export-dir /Query10 --input-fields-terminated-by ',' --mysql-delimiters -m 1**;**

**OUTPUT :**

**mysql> select \*from suc\_rate;**

+------------------------------------------------------------+------------------+---------------------------------------+--------------+

| job\_title | total\_no\_of\_appl | certifiedANDcertified\_withdrwan\_count | Success\_rate |

+------------------------------------------------------------+------------------+---------------------------------------+--------------+

| PRODUCTION SUPPORT LEAD - US | 1301 | 1301 | 100 |

| ASSOCIATE CONSULTANT - US | 4393 | 4390 | 99.9317 |

| SYSTEMS ENGINEER - US | 10036 | 10026 | 99.9004 |

| TEST ENGINEER - US | 2198 | 2195 | 99.8635 |

| PRODUCTION SUPPORT ANALYST - US | 1451 | 1449 | 99.8622 |

| TEST ANALYST - US | 4958 | 4949 | 99.8185 |

| CONSULTANT - US | 7426 | 7412 | 99.8115 |

| TECHNOLOGY LEAD - US | 28350 | 28294 | 99.8025 |

| TECHNICAL TEST LEAD - US | 5374 | 5363 | 99.7953 |

| SENIOR TECHNOLOGY ARCHITECT - US | 1417 | 1414 | 99.7883 |

| TECHNOLOGY ARCHITECT - US | 4707 | 4696 | 99.7663 |

| TECHNOLOGY ANALYST - US | 26055 | 25993 | 99.762 |

| SENIOR PROJECT MANAGER - US | 2774 | 2767 | 99.7477 |

| DEVELOPER USER INTERFACE | 5247 | 5232 | 99.7141 |

| COMPUTER SYSTEMS ANALYST 2 | 4031 | 4019 | 99.7023 |

| SYSTEMS ANALYST - II | 1339 | 1335 | 99.7013 |

| PROJECT MANAGER - III | 1651 | 1646 | 99.6972 |

| PROJECT MANAGER - US | 7046 | 7024 | 99.6878 |

| PROGRAMMER ANALYST - II | 3588 | 3576 | 99.6656 |

| LEAD CONSULTANT - US | 3402 | 3390 | 99.6473 |

| COMPUTER SYSTEMS ANALYST 3 | 2170 | 2161 | 99.5853 |

| COMPUTER PROGRAMMER/CONFIGURER 2 | 6729 | 6700 | 99.569 |

| PROGRAMMER ANALYST - I | 1432 | 1425 | 99.5112 |

| SYSTEMS ANALYST - III | 1006 | 1001 | 99.503 |

| PRINCIPAL CONSULTANT - US | 1352 | 1345 | 99.4822 |

| COMPUTER SPECIALIST/TESTING AND QUALITY ANALYST 2 | 3998 | 3975 | 99.4247 |

| COMPUTER PROGRAMMER/CONFIGURER 3 | 1145 | 1138 | 99.3886 |

| COMPUTER SPECIALIST/SYSTEM SUPPORT AND DEVELOPMENT | 1339 | 1330 | 99.3279 |

| COMPUTER SPECIALIST/SYSTEM SUPPORT AND DEVELOPMENT ADMIN 2 | 1085 | 1077 | 99.2627 |

| DATA WAREHOUSE SPECIALIST | 1631 | 1618 | 99.2029 |