Analysing H1B Visa

By using Hadoop

**Name : Ambe Kunal Badade**

**Student Id : S171107500138**

**Abstract**

What is Big Data?

Big data is a term that describes the large volume of data – both structured, unstructured and semi-structured data thatoverrun a business on a day-to-day basis. But it’s not the amount of data that’s important. It’s what organizations do with the data that matters. Big data can be analysed for insights that lead to better decisions and strategic business moves.

Hadoop

Hadoop is an open-source software framework for storing data and running applications on clusters of commodity hardware. It provides massive storage for any kind of data, enormous processing power and the ability to handle virtually limitless concurrent tasks or jobs.

**Why Hadoop is important:**

* **Ability to store and process huge amounts of any kind of data, quickly.** With data volumes and varieties constantly increasing, especially from social media and the Internet of Things (IoT), that's a key consideration.
* **Computing power.** Hadoop's distributed computing model processes big data fast. The more computing nodes you use, the more processing power you have.
* **Fault tolerance.** Data and application processing are protected against hardware failure. If a node goes down, jobs are automatically redirected to other nodes to make sure the distributed computing does not fail. Multiple copies of all data are stored automatically.
* **Flexibility.** Unlike traditional relational databases, you don’t have to pre-process data before storing it. You can store as much data as you want and decide how to use it later. That includes unstructured data like text, images and videos.
* **Low cost.** The open-source framework is free and uses commodity hardware to store large quantities of data.
* **Scalability.** You can easily grow your system to handle more data simply by adding nodes. Little administration is required.

**Acknowledgement**

I wish to thank my master trainer Mr.SandeepAgarwal and my Tech mentor Ms.Jyoti Mittal for providing complete learning on Big Data and Hadoop and guiding me in accomplishing the objectives of my project.

**Project Outline**

|  |  |
| --- | --- |
| Title | Big Data Analysis in Hadoop on H1B Data |
| Inputs | H1b Data |
| Data Elements | Sr.no, case status, employer name, soc name, job position, full time position,prevailing wage, year, worksite, longitude, latitude. |
| Purpose | To provide analyzed report to H1B Sponsers to help them to make plans for minimum wage provision and to get skilled workers from respective worksite. |
| Methodology | Agile |

**Project Implementation**

**Assumption:**

1. Hadoop cluster is running.

2. Ecosystem products (Pig, hive, Sqoop) are installed.

3. H1b data is available on HDFS in CSV format.

**Prerequisite for all the jobs:**

H1B data is in CSV format hence need to be converted into TextFile format in HadoopFileSystem.

**Steps for conversion:**

1. Create a table in hive to read entire record as as on single row.
2. Create a table to convert and store aentire record into different tab separated fields which will create a file on HadoopfileSystem in TextFile format.

**Note:** Look up supporting tables are created based on analysis.

**H1b analysis:**

**Task 1:**

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

Technology : Pig.

h1bvisa = load '/home/hduser/h1b\_final/' using PigStorage() as (sr\_no:chararray,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:double,year:chararray,work\_site:chararray,longitude:chararray,latitude:chararray);

data = foreach h1bvisa generate $4,$7;

dengg= filter data by job\_title matches'.\*DATA ENGINEER.\*';

--dump data;.

year11 = filter dengg by year == '2011';

year12 = filter dengg by year == '2012';

year13 = filter dengg by year == '2013';

year14 = filter dengg by year == '2014';

year15 = filter dengg by year == '2015';

year16 = filter dengg by year == '2016';

data11 = group year11 by year;

total11 = foreach data11 generate group as year,COUNT(year11.job\_title);

--dump total11;

data12 = group year12 by year;

total12 = foreach data12 generate group as year,COUNT(year12.job\_title);

data13 = group year13 by year;

total13 = foreach data13 generate group as year,COUNT(year13.job\_title);

data14 = group year14 by year;

total14 = foreach data14 generate group as year,COUNT(year14.job\_title);

data15 = group year15 by year;

total15 = foreach data15 generate group as year,COUNT(year15.job\_title);

data16 = group year16 by year;

total16 = foreach data16 generate group as year,COUNT(year16.job\_title);

result = UNION total11,total12,total13,total14,total15,total16;

final = order result by year asc;

--RESULT WILL BE SHOWN AS YEAR,TOTAL APPLICATION OF DATAENGG

store final into '/home/hduser/project/output1a' using PigStorage();

OUTPUT

2011 60

2012 81

2013 151

2014 249

2015 394

2016 786

b)**Find top 5 job titles who are having highest growth in applications.**

Technology : Pig

h1bvisa = load '/home/hduser/h1b\_final/' using PigStorage() as (sr\_no:chararray,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:double,year:chararray,work\_site:chararray,longitude:chararray,latitude:chararray);

data = foreach h1bvisa generate $4,$7;

year11 = filter data by year == '2011';

year12 = filter data by year == '2012';

year13 = filter data by year == '2013';

year14 = filter data by year == '2014';

year15 = filter data by year == '2015';

year16 = filter data by year == '2016';

data11 = group year11 by job\_title;

total11 = foreach data11 generate group as job\_title,COUNT(year11.$1)as no;

--dump total11;

data12 = group year12 by job\_title;

total12 = foreach data12 generate group as job\_title,COUNT(year12.$1)as no;

data13 = group year13 by job\_title;

total13 = foreach data13 generate group as job\_title,COUNT(year13.$1) as no;

data14 = group year14 by job\_title;

total14 = foreach data14 generate group as job\_title,COUNT(year14.$1)as no;

data15 = group year15 by job\_title;

total15 = foreach data15 generate group as job\_title,COUNT(year15.$1)as no;

data16 = group year16 by job\_title;

total16 = foreach data16 generate group as job\_title,COUNT(year16.$1)as no;

join\_data1 = join total11 by job\_title ,total12 by job\_title;

join\_data2 = join total12 by job\_title ,total13 by job\_title;

join\_data3 = join total13 by job\_title ,total14 by job\_title;

join\_data4 = join total14 by job\_title ,total15 by job\_title;

join\_data5 = join total15 by job\_title ,total16 by job\_title;

filterdata1 = foreach join\_data1 generate $0,$1,$3;

growth1 = foreach filterdata1 generate $0,(double)($2-$1)\*100/$1 as grwoth11;

filterdata2 = foreach join\_data2 generate $0,$1,$3;

growth2 = foreach filterdata2 generate $0,(double)($2-$1)\*100/$1 as grwoth12;

filterdata3 = foreach join\_data3 generate $0,$1,$3;

growth3 = foreach filterdata3 generate $0,(double)($2-$1)\*100/$1 as grwoth13;

filterdata4 = foreach join\_data4 generate $0,$1,$3;

growth4 = foreach filterdata4 generate $0,(double)($2-$1)\*100/$1 as grwoth14;

filterdata5 = foreach join\_data5 generate $0,$1,$3;

growth5 = foreach filterdata5 generate $0,(double)($2-$1)\*100/$1 as grwoth15;

growth = join growth1 by $0,growth2 by $0,growth3 by $0,growth4 by $0,growth5 by $0;

maindata = foreach growth generate $0,$1,$3,$5,$7,$9;

avggrowth = foreach maindata generate $0,(double)(($1+$2+$3+$4+$5)/5) as average;

final = order avggrowth by $1 desc;

result = limit final 5;

store result into '/home/hduser/project/h1b\_1b' using PigStorage();

Ouptut

SENIOR SYSTEMS ANALYST JC60 4255.464472401351

SOFTWARE DEVELOPER 2 3480.5924860255664

PROJECT MANAGER 3 3233.333333333333

SYSTEMS ANALYST JC65 2984.8808859629844

MODULE LEAD 2917.112044407748

**Task 2:**

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

**Technology used: Pig**

MapReduce : problem2a

package p2a;

import java.io.\*;

import java.util.TreeMap;

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

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

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

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

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

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

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

public class Partwise extends Configured implements Tool

{

//Map class

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 str1 = str[4]; //job title column

String str2 = str[7].trim(); //year column

String str3 =str[8]; //worksite

if (str1.contains("DATA ENGINEER"))

{

context.write(new Text(str3), new Text(str2));

}

}

catch(Exception e)

{

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

}

}

}

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

{

TreeMap<Integer,Text> tm=new TreeMap<Integer,Text>();

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

{

int count=0;

String year=null;

for(Text val:values)

{

year=val.toString();

count++;

}

String total=String.format("%d", count);

String mykey=key.toString();

String myvalue = mykey+","+","+year+","+total;

tm.put(new Integer(count),new Text(myvalue));

}

public void cleanup(Context context) throws IOException, InterruptedException

{

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

{

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

}

}

}

public static class MyPartitioner extends

Partitioner < Text, Text >

{

@Override

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

{

int year=Integer.parseInt(value.toString());

if(year==2011)

{

return 0 % numReduceTasks;

}

else if(year==2012)

{

return 1 % numReduceTasks;

}

else if(year==2013)

{

return 2 % numReduceTasks;

}

else if(year==2014)

{

return 3 % numReduceTasks;

}

else if(year==2015)

{

return 4 % numReduceTasks;

}

else if(year==2016)

{

return 5 % numReduceTasks;

}

else

{

return 6 % numReduceTasks;

}

}

}

@Override

public int run(String[] arg) throws Exception

{

Configuration conf = getConf();

Job job = new Job(conf);

job.setJarByClass(Partwise.class);

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

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

job.setMapperClass(MapClass.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

//set partitioner statement

job.setPartitionerClass(MyPartitioner.class);

job.setReducerClass(ReduceClass.class);

job.setNumReduceTasks(7);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

job.setOutputKeyClass(NullWritable.class);

job.setOutputValueClass(Text.class);

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

return 0;

}

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

{

int res = ToolRunner.run(new Configuration(), new Partwise(),ar);

System.exit(0);

}

}

Output

SEATTLE, WASHINGTON,,2011,20

SAN FRANCISCO, CALIFORNIA,,2011,4

SAN MATEO, CALIFORNIA,,2011,3

WALTHAM, MASSACHUSETTS,,2011,2

TALLAHASSEE, FLORIDA,,2011,1

2012

SEATTLE, WASHINGTON,,2012,30

SAN FRANCISCO, CALIFORNIA,,2012,10

PONTIAC, MICHIGAN,,2012,3

SAN MATEO, CALIFORNIA,,2012,2

WOODLAND HILLS, CALIFORNIA,,2012,1

2013

SEATTLE, WASHINGTON,,2013,46

SAN FRANCISCO, CALIFORNIA,,2013,17

MENLO PARK, CALIFORNIA,,2013,12

NEW YORK, NEW YORK,,2013,6

ATLANTA, GEORGIA,,2013,5

MOUNTAIN VIEW, CALIFORNIA,,2013,3

THOUSAND OAKS, CALIFORNIA,,2013,2

WOODLAND HILLS, CALIFORNIA,,2013,1

2014

SEATTLE, WASHINGTON,,2014,45

SAN FRANCISCO, CALIFORNIA,,2014,34

MENLO PARK, CALIFORNIA,,2014,21

NEW YORK, NEW YORK,,2014,18

MOUNTAIN VIEW, CALIFORNIA,,2014,13

SAN MATEO, CALIFORNIA,,2014,8

IRVINE, CALIFORNIA,,2014,7

REDWOOD CITY, CALIFORNIA,,2014,5

SUNNYVALE, CALIFORNIA,,2014,4

ST. PETERSBURG, FLORIDA,,2014,3

WINSTON-SALEM, NORTH CAROLINA,,2014,2

YONKERS, NEW YORK,,2014,1

2015

SEATTLE, WASHINGTON,,2015,61

NEW YORK, NEW YORK,,2015,41

MENLO PARK, CALIFORNIA,,2015,23

MOUNTAIN VIEW, CALIFORNIA,,2015,18

SAN MATEO, CALIFORNIA,,2015,15

SANTA MONICA, CALIFORNIA,,2015,13

SAN RAMON, CALIFORNIA,,2015,8

SUNNYVALE, CALIFORNIA,,2015,7

SAN JOSE, CALIFORNIA,,2015,6

REDWOOD CITY, CALIFORNIA,,2015,5

CHICAGO, ILLINOIS,,2015,4

TROY, MICHIGAN,,2015,3

WESTBOROUGH, MASSACHUSETTS,,2015,2

WOODLAND HILLS, CALIFORNIA,,2015,1

2016

SEATTLE, WASHINGTON,,2016,128

SAN FRANCISCO, CALIFORNIA,,2016,90

NEW YORK, NEW YORK,,2016,70

MENLO PARK, CALIFORNIA,,2016,39

IRVINE, CALIFORNIA,,2016,18

SUNNYVALE, CALIFORNIA,,2016,16

SAN MATEO, CALIFORNIA,,2016,14

CHICAGO, ILLINOIS,,2016,13

SANTA CLARA, CALIFORNIA,,2016,12

MOUNTAIN VIEW, CALIFORNIA,,2016,11

SAN JOSE, CALIFORNIA,,2016,10

PLANO, TEXAS,,2016,9

SANTA MONICA, CALIFORNIA,,2016,8

WALTHAM, MASSACHUSETTS,,2016,7

BURLINGTON, MASSACHUSETTS,,2016,6

SAN BRUNO, CALIFORNIA,,2016,5

VIENNA, VIRGINIA,,2016,4

VENICE, CALIFORNIA,,2016,3

WILMINGTON, DELAWARE,,2016,2

YORKTOWN HEIGHTS, NEW YORK,,2016,1

**Task:2**

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

package p2b;

import java.io.\*;

import java.util.TreeMap;

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

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

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

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

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

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

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

public class Driver extends Configured implements Tool

{

//Map class

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 str1 = str[1]; //case status.

String str2 = str[7].trim(); //year column

String str3 =str[8]; //worksite

if (str1.contains("CERTIFIED"))

{

context.write(new Text(str3), new Text(str2));

}

}

catch(Exception e)

{

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

}

}

}

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

{

TreeMap<Integer,Text> tm=new TreeMap<Integer,Text>();

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

{

int count=0;

String year=null;

for(Text val:values)

{

year=val.toString();

count++;

}

String total=String.format("%d", count);

String mykey=key.toString();

String myvalue = mykey+","+","+year+","+total;

tm.put(new Integer(count),new Text(myvalue));

if (tm.size() > 5)

{

tm.remove(tm.firstKey());

}

}

public void cleanup(Context context) throws IOException, InterruptedException

{

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

{

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

}

}

}

public static class MyPartitioner extends

Partitioner < Text, Text >

{

@Override

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

{

int year=Integer.parseInt(value.toString());

if(year==2011)

{

return 0 % numReduceTasks;

}

else if(year==2012)

{

return 1 % numReduceTasks;

}

else if(year==2013)

{

return 2 % numReduceTasks;

}

else if(year==2014)

{

return 3 % numReduceTasks;

}

else if(year==2015)

{

return 4 % numReduceTasks;

}

else if(year==2016)

{

return 5 % numReduceTasks;

}

else

{

return 6 % numReduceTasks;

}

}

}

@Override

public int run(String[] arg) throws Exception

{

Configuration conf = getConf();

Job job = new Job(conf, "topsal");

job.setJarByClass(Driver.class);

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

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

job.setMapperClass(MapClass.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

//set partitioner statement

job.setPartitionerClass(MyPartitioner.class);

job.setReducerClass(ReduceClass.class);

job.setNumReduceTasks(7);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

job.setOutputKeyClass(NullWritable.class);

job.setOutputValueClass(Text.class);

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

return 0;

}

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

{

int res = ToolRunner.run(new Configuration(), new Driver(),ar);

System.exit(0);

}

}

OUTPUT

2011

NEW YORK, NEW YORK,,2011,24024

2012

NEW YORK, NEW YORK,,2012,26161

2013

NEW YORK, NEW YORK,,2013,25888

2014.

NEW YORK, NEW YORK,,2014,30132

2015

NEW YORK, NEW YORK,,2015,34216

2016

NEW YORK, NEW YORK,,2016,37746

**Task 3:**

**Which industry has the most number of Data Scientist positions?**

**Technology Used: Hive**

use h1b\_project;

insert overwrite directory '/h1b/problem3' row format delimited fields terminated by ',' stored as textfile select soc\_name,count(job\_title) as total from h1b\_final where job\_title ="DATA SCIENTIST" group by soc\_name order by total desc limit 10;

OutPut

STATISTICIANS,415

COMPUTER AND INFORMATION RESEARCH SCIENTISTS,333

OPERATIONS RESEARCH ANALYSTS,253

Computer and Information Research Scientists,134

COMPUTER OCCUPATIONS, ALL OTHER,121

MATHEMATICIANS,108

Statisticians,104

SOFTWARE DEVELOPERS, APPLICATIONS,79

COMPUTER SYSTEMS ANALYSTS,66

Operations Research Analysts,64

**Task 4:**

**Which top 5 employers file the most petitions each year?**

**Technology Used: Hive**

use h1b\_project;

insert overwrite directory '/h1b/problem4/2015' row format delimited fields terminated by ',' stored as textfile select employer\_name,year,count(case\_status) as total from h1b\_final where year=2015 group by employer\_name,year order by total desc limit 5;

insert overwrite directory '/h1b/problem4/2011' row format delimited fields terminated by ',' stored as textfile select employer\_name,year,count(case\_status) as total from h1b\_final where year=2011 group by employer\_name,year order by total desc limit 5;

insert overwrite directory '/h1b/problem4/2012' row format delimited fields terminated by ',' stored as textfile select employer\_name,year,count(case\_status) as total from h1b\_final where year=2012 group by employer\_name,year order by total desc limit 5;

insert overwrite directory '/h1b/problem4/2013' row format delimited fields terminated by ',' stored as textfile select employer\_name,year,count(case\_status) as total from h1b\_final where year=2013 group by employer\_name,year order by total desc limit 5;

insert overwrite directory '/h1b/problem4/2014' row format delimited fields terminated by ',' stored as textfile select employer\_name,year,count(case\_status) as total from h1b\_final where year=2014 group by employer\_name,year order by total desc limit 5;

insert overwrite directory '/h1b/problem4/2016' row format delimited fields terminated by ',' stored as textfile select employer\_name,year,count(case\_status) as total from h1b\_final where year=2016 group by employer\_name,year order by total desc limit 5;

OUTPUT  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2011

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

2012

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

2013

INFOSYS LIMITED,2013,32223

TATA CONSULTANCY SERVICES LIMITED,2013,8790

WIPRO LIMITED,2013,6734

DELOITTE CONSULTING LLP,2013,6124

ACCENTURE LLP,2013,4994

2014

INFOSYS LIMITED,2014,23759

TATA CONSULTANCY SERVICES LIMITED,2014,14098

WIPRO LIMITED,2014,8365

DELOITTE CONSULTING LLP,2014,7017

ACCENTURE LLP,2014,5498

2015

INFOSYS LIMITED,2015,33245

TATA CONSULTANCY SERVICES LIMITED,2015,16553

WIPRO LIMITED,2015,12201

IBM INDIA PRIVATE LIMITED,2015,10693

ACCENTURE LLP,2015,9605

2016

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

**Task 5:**

**Find the most popular top 10 job positions for H1B visa applications for each year?**

**Technology Used:Hive**

use h1b\_project;

insert overwrite directory '/h1b/problem5a/2011' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2011 group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5a/2012' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2012 group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5a/2013' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2013 group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5a/2014' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2014 group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5a/2015' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2015 group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5a/2016' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2016 group by job\_title,year order by total desc limit 10;

OUTPUT

2011

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

2012

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

SENIOR CONSULTANT,2012,3737

2013

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

2014

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

2015

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

2016

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 : Hive

use h1b\_project;

insert overwrite directory '/h1b/problem5b/2011' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2011 AND case\_status ="CERTIFIED" group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5b/2012' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2012 AND case\_status ="CERTIFIED" group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5b/2013' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2013 AND case\_status ="CERTIFIED" group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5b/2014' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2014 AND case\_status ="CERTIFIED" group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5b/2015' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2015 AND case\_status ="CERTIFIED" group by job\_title,year order by total desc limit 10;

insert overwrite directory '/h1b/problem5b/2016' row format delimited fields terminated by ',' stored as textfile select job\_title,year,count(case\_status) as total from h1b\_final where year=2016 AND case\_status ="CERTIFIED" group by job\_title,year order by total desc limit 10;

2011

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

2012

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

2013

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

SENIOR CONSULTANT,2013,3996

2014

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

2015

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

2016

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 : pig

--certified

h1b = load '/home/hduser/h1b\_final/' using PigStorage() as(sno:int, case\_status:chararray, employer\_name:chararray, soc\_name:chararray, job\_title:chararray,full\_time\_position:chararray,prevailining\_wage:double, year:chararray, worksite:chararray, longitude:int, latitude:int);

new = foreach h1b generate $7, $1;

groupbyyear = group new by year;

newgroup = foreach groupbyyear generate group as year, COUNT(new)as total;

filterbycase= filter new by case\_status=='CERTIFIED';

filteryear= group filterbycase by year;

certified = foreach filteryear generate group as year, COUNT(filterbycase) as certified;

joindata = join newgroup by $0,certified by $0;

data = foreach joindata generate $0,$1,$3,((double)$3\*100/(double)$1) as percent;

Store data into '/home/hduser/project/output6/certified' using PigStorage();

--certified-with

newgroup1= foreach groupbyyear generate group as year, COUNT(new)as total;

filterbycase1= filter new by case\_status=='CERTIFIED-WITHDRAWN';

filteryear1= group filterbycase1 by year;

certified1 = foreach filteryear1 generate group as year, COUNT(filterbycase1) as certified;

joindata1 = join newgroup1 by $0,certified1 by $0;

data1 = foreach joindata1 generate $0,$1,$3,((double)$3\*100/(double)$1) as percent;

Store data1 into '/home/hduser/project/output6/certified-with' using PigStorage();

--withdrawn

newgroup2 = foreach groupbyyear generate group as year, COUNT(new)as with;

filterbycase2= filter new by case\_status=='WITHDRAWN';

filteryear2= group filterbycase2 by year;

certified2 = foreach filteryear2 generate group as year, COUNT(filterbycase2) as certified;

joindata2 = join newgroup2 by $0,certified by $0;

data2 = foreach joindata2 generate $0,$1,$3,((double)$3\*100/(double)$1) as percent;

Store data1 into '/home/hduser/project/output6/withdrawn' using PigStorage();

--Denied

newgroup3 = foreach groupbyyear generate group as year, COUNT(new)as total;

filterbycase3= filter new by case\_status=='DENIED';

filteryear3= group filterbycase3 by year;

denied3= foreach filteryear3 generate group as year, COUNT(filterbycase3) as certified;

joindata3 = join newgroup3 by $0,certified by $0;

data3 = foreach joindata3 generate $0,$1,$3,((double)$3\*100/(double)$1) as percent;

Store data1 into '/home/hduser/project/output6/denied' using PigStorage();

Output

Certified

2011 358767 307936 85.83175152675692

2012 415607 352668 84.85612609989725

2013 442114 382951 86.61815730784367

2014 519427 455144 87.62424748809747

2015 618727 547278 88.45225761927313

2016 647803 569646 87.93506667922192

Cerified-withdrawn

2011 358767 11596 3.2321813321738064

2012 415607 31118 7.487361858678993

2013 442114 35432 8.014222576077664

2014 519427 36350 6.9980959788382195

2015 618727 41071 6.637984119005636

2016 647803 47092 7.269493966529948

withdrawn

2011 358767 11596 3.2321813321738064

2012 415607 31118 7.487361858678993

2013 442114 35432 8.014222576077664

2014 519427 36350 6.9980959788382195

2015 618727 41071 6.637984119005636

2016 647803 47092 7.269493966529948

Denied

2011 358767 11596 3.2321813321738064

2012 415607 31118 7.487361858678993

2013 442114 35432 8.014222576077664

2014 519427 36350 6.9980959788382195

2015 618727 41071 6.637984119005636

2016 647803 47092 7.269493966529948

**Task 7:**

**Create a bar graph to depict the number of applications for each year ?**

**Technology Used: pig**

h1b = load '/h1b\_final' using PigStorage() as(sno:int, case\_status:chararray, employer\_name:chararray, soc\_name:chararray, job\_title:chararray,full\_time\_position:chararray,prevailining\_wage:double, year:chararray, worksite:chararray, longitude:int, latitude:int);

data = foreach h1b generate year,sno;

yeardata = group data by year;

total = foreach yeardata generate group as year,COUNT(data.sno) as application;

result = order total by year desc;

store result into '/h1b/output7' using PigStorage();

**output**

2016 647803

2015 618727

2014 519427

2013 442114

2012 415607

2011 358767

**Find the average Prevailing Wage for each Job for each Year (take part time and full time separate) ?**

Technology :MapReduce.

Package h1b\_prob8;

import java.io.\*;

import java.util.TreeMap;

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

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

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

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

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

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

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

public class Driver extends Configured implements Tool

{

//Map class

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 str1 = str[4]; //job title column

String str2 = str[7]; //year column

String str3 =str[5]; //full or part time

String str4 = str[6]; //prevaling wage

context.write(new Text(str1+","+str3), new Text(str4+","+str2));

}

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 total=0;

String year="";

int count=0;

for(Text val:values)

{

int wage=0;

String data[]=val.toString().split(",");

if(data[0].contains("N"))

{

wage=0;

}

else wage=Integer.parseInt(data[0]);

total+=wage;

count++;

year=data[1];

}

double avg=(double)total\*100/(double)count;

String average=String.format("%.2f",avg);

String newValue=year+","+average;

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

}

}

public static class MyPartitioner extends

Partitioner < Text, Text >

{

@Override

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

{

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

int year=Integer.parseInt(record[1]);

if(year==2011)

{

return 0 % numReduceTasks;

}

else if(year==2012)

{

return 1 % numReduceTasks;

}

else if(year==2013)

{

return 2 % numReduceTasks;

}

else if(year==2014)

{

return 3 % numReduceTasks;

}

else if(year==2015)

{

return 4 % numReduceTasks;

}

else if(year==2016)

{

return 5 % numReduceTasks;

}

else

{

return 6 % numReduceTasks;

}

}

}

@Override

public int run(String[] arg) throws Exception

{

Configuration conf = getConf();

Job job = new Job(conf, "topsal");

job.setJarByClass(Driver.class);

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

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

job.setMapperClass(MapClass.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

//set partitioner statement

job.setPartitionerClass(MyPartitioner.class);

job.setReducerClass(ReduceClass.class);

job.setNumReduceTasks(7);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(Text.class);

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

return 0;

}

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

{

int res = ToolRunner.run(new Configuration(), new Driver(),ar);

System.exit(0);

}

}

Output

2011 (BILINGUAL) PROGRAMMER,Y 2011,4103800.00

2012 "SALES MANAGER,N 2012,4180800.00

2013 "TEST" SENIOR SCIENTIST,Y 2013,8332500.00

2014 'COMPUTER SYSTEMS ANALYSTS,Y 2014,4000000.00

2015 (FASHION) SHOE DESIGNER,Y 2015,7142700.00

2016 "BUSINESS SYSTEM ANALYST,N 2016,4942100.00

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

Technology :MapReduce

package h1b\_prob9;

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.Reducer;

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

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

public class Driver

{

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

{

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

{

try

{

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

String case\_status=str[1];

String Employer=str[2];

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

}

catch(Exception e)

{

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

}

}

}

public static class SuccessReducer extends Reducer<Text,Text,NullWritable,Text>

{

TreeMap<Double,Text> topMap=new TreeMap<Double,Text>();

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

{

double success\_rate=0;

int petitions=0,valid\_app=0;

int certified=0,certified\_withdrawn=0;

String myvalue=null;

String mykey=key.toString();

for(Text val:values)

{

petitions++;

String case\_status=val.toString();

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

{

certified++;

}

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

{

certified\_withdrawn++;

}

}

if(petitions>10000)

{

valid\_app=certified+certified\_withdrawn;

success\_rate=(double)valid\_app\*100/(double)petitions;

String mysuccessRate=String.format("%.2f",success\_rate);

String mypetitions=String.format("%d", petitions);

myvalue=mykey+"\t"+mypetitions+"\t"+mysuccessRate;

}

if(success\_rate >70.00)

{

topMap.put(new Double(success\_rate),new Text(myvalue));

if(topMap.size() >10)

{

topMap.remove(topMap.firstKey());

}

}

}

public void cleanup(Context context) throws IOException, InterruptedException

{

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

{

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

}

}

}

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

{

Configuration conf= new Configuration();

Job job=new Job(conf);

job.setJarByClass(Driver.class);

job.setMapperClass(SuccessMapper.class);

job.setReducerClass(SuccessReducer.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

job.setOutputKeyClass(NullWritable.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

INFOSYS LIMITED 130592 99.48

TECH MAHINDRA (AMERICAS),INC. 10732 99.34

TATA CONSULTANCY SERVICES LIMITED 64726 99.24

ACCENTURE LLP 33447 98.40

DELOITTE CONSULTING LLP 36742 98.31

HCL AMERICA, INC. 22678 98.04

ERNST & YOUNG U.S. LLP 18232 98.04

IGATE TECHNOLOGIES INC. 12564 95.91

CAPGEMINI AMERICA INC 16725 95.41

LARSEN & TOUBRO INFOTECH LIMITED 17457 95.39

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 more than 1000)?

Technology:MapReduce.

package h1b\_10\_10;

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.Reducer;

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

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

public class Driver

{

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

{

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

{

try

{

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

String case\_status=str[1];

String job\_title=str[4];

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

}

catch(Exception e)

{

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

}

}

}

public static class SuccessReducer extends Reducer<Text,Text,NullWritable,Text>

{

TreeMap<Double,Text> topMap=new TreeMap<Double,Text>();

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

{

double success\_rate=0;

int petitions=0,valid\_app=0;

int certified=0,certified\_withdrawn=0;

String myvalue=null;

String mykey=key.toString();

for(Text val:values)

{

petitions++;

String case\_status=val.toString();

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

{

certified++;

}

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

{

certified\_withdrawn++;

}

}

if(petitions>10000)

{

valid\_app=certified+certified\_withdrawn;

success\_rate=(double)valid\_app\*100/(double)petitions;

String mysuccessRate=String.format("%.2f",success\_rate);

String mypetitions=String.format("%d", petitions);

myvalue=mykey+"\t"+mypetitions+"\t"+mysuccessRate;

}

if(success\_rate >70.00)

{

topMap.put(new Double(success\_rate),new Text(myvalue));

if(topMap.size() >10)

{

topMap.remove(topMap.firstKey());

}

}

}

public void cleanup(Context context) throws IOException, InterruptedException

{

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

{

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

}

}

}

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

{

Configuration conf= new Configuration();

Job job=new Job(conf);

job.setJarByClass(Driver.class);

job.setMapperClass(SuccessMapper.class);

job.setReducerClass(SuccessReducer.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

job.setOutputKeyClass(NullWritable.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

SYSTEMS ENGINEER - US 10036 99.84

TECHNOLOGY LEAD - US 28350 99.75

TECHNOLOGY ANALYST - US 26055 99.73

DEVELOPER 12909 95.67

COMPUTER SYSTEMS ENGINEER 11090 94.95

CONSULTANT 23081 92.97

SENIOR CONSULTANT 24904 92.33

LEAD ENGINEER 11157 91.37

COMPUTER PROGRAMMER 70570 90.72

SYSTEMS ANALYST 61965 89.96

Scope

YARN

The fundamental idea of YARN is to split up the functionalities of resource management and job scheduling/monitoring into separate daemons. The idea is to have a global ResourceManager (*RM*) and per-application ApplicationMaster (*AM*). An application is either a single job or a DAG of jobs.

The ResourceManager and the NodeManager form the data-computation framework. The ResourceManager is the ultimate authority that arbitrates resources among all the applications in the system. The NodeManager is the per-machine framework agent who is responsible for containers, monitoring their resource usage (cpu, memory, disk, network) and reporting the same to the ResourceManager/Scheduler.

The per-application ApplicationMaster is, in effect, a framework specific library and is tasked with negotiating resources from the ResourceManager and working with the NodeManager(s) to execute and monitor the tasks.

**HBase:** HBase is a non-relational database that allows for low-latency, quick lookups in Hadoop. It adds transactional capabilities to Hadoop, allowing users to conduct updates, inserts and deletes. EBay and Facebook use HBase heavily.

**Flume:** Flume is a framework for populating Hadoop with data. Agents are populated throughout ones IT infrastructure – inside web servers, application servers and mobile devices, for example – to collect data and integrate it into Hadoop.

**Oozie:** Oozie is a workflow processing system that lets users define a series of jobs written in multiple languages – such as Map Reduce, Pig and Hive -- then intelligently link them to one another. Oozie allows users to specify, for example, that a particular query is only to be initiated after specified previous jobs on which it relies for data are completed.