**H1B visa dataset**

**BIG DATA - HADOOP**

SANJEEV KUMAR E |Professional Diploma in Digital Transformation – Big Data with Hadoop |

24.10.2017

Tools : Apache Hadoop Framework – HDFS, MapReduce, Hive, Pig, Sqoop, MySql and MS Excel for Data Visualization

S180030100193

NIIT

**H1B visa Data set Analysis**

**A PROJECT REPORT**

***Submitted by***

***SANJEEV KUMAR E***

***in the partial fulfillment for the award of the course***

***of***

**PROFESSIONAL DIPLOMA**

***in***

**BIG DATA WITH HADOOP**

**NIIT, CHENNAI**

**JULY 2017**

**ACKNOWLEDGEMENT**

I find immense pleasure to convey my sincere and grateful thanks to **NIIT** and the management for providing necessary facilities in carrying out this project.

I greatly indebted to my Tech Mentor **Ms. Amirtha,** the batch instructor **Mr. Annu** and the SLT faculty **Mr. Sandeep** for constant support throughout the course and also for useful suggestions, constant encouragement and kind advice in bringing out this project as a success.

I express my regards and sincere thanks to the Academic Leader **Ms. Aishwarya** for providing all necessary resource to complete the project. I extend my thanks to all staff members of NIIT for their kind co-operation for the completion of the project successfully. I am grateful to thank my family and all my friends for their valuable feedback, encouragement and suggestions.

**Introduction**

**Big Data**

Big data is being generated by everything around us at all times. Every digital process and social media exchange produces it. Systems, sensors and mobile devices transmit it. Big data is arriving from multiple sources at an alarming velocity, volume and variety. To extract meaningful value from big data, you need optimal parallel processing power, analytics capabilities and skills.

While the term “big data” is relatively new, the act of gathering and storing large amounts of information for eventual analysis is ages old. The concept gained momentum in the early 2000s when industry analyst Doug Laney articulated the now-mainstream definition of big data as the four Vs:

\* **Volume-**Organizations collect data from a variety of sources, including business transactions, social media and information from sensor or machine-to-machine data. In the past, storing it would’ve been a problem – but new technologies (such as Hadoop) have eased the burden.

\* **Velocity-** Data streams in at an unprecedented speed and must be dealt with in a timely manner. RFID tags, sensors and smart metering are driving the need to deal with torrents of data in near-real time

\* **Variety-**Data comes in all types of formats – from structured, numeric data in traditional databases to unstructured text documents, email, video, audio, stock ticker data and financial transactions.

\* **veracity-**accuracy of data

4V’s of Big Data

**Apache Hadoop**

The Apache Hadoop project develops open-source software for reliable, scalable, distributed computing.

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

**History**

The genesis of Hadoop came from the Google File System paper that was published in October 2003. This paper spawned another research paper from Google – MapReduce: Simplified Data Processing on Large Clusters. Development started in the Apache Nutch project, but was moved to the new Hadoop subproject in January 2006. The first committer added to the Hadoop project was Owen O’Malley in March 2006. Hadoop 0.1.0 was released in April 2006 and continues to be evolved by the many contributors to the Apache Hadoop project. Hadoop was named after one of the founder’s toy elephant.

Hadoop Timeline

**Benefits**

Some of the reasons organizations use Hadoop is its’ ability to store, manage and analyze vast amounts of structured and unstructured data quickly, reliably, flexibly and at low-cost.

- **Scalability and Performance** – distributed processing of data local to each node in a cluster enables Hadoop to store, manage, process and analyze data at petabyte scale.

- **Reliability** – large computing clusters are prone to failure of individual nodes in the cluster. Hadoop is fundamentally resilient – when a node fails processing is re-directed to the remaining nodes in the cluster and data is automatically re-replicated in preparation for future node failures.

- **Flexibility** – unlike traditional relational database management systems, you don’t have to created structured schemas before storing data. You can store data in any format, including semi-structured or unstructured formats, and then parse and apply schema to the data when read.

- **Low Cost** – unlike proprietary software, Hadoop is open source and runs on low-cost commodity hardware.

Hadoop framework and Apache projects:

1. **Hadoop Common**: The common utilities that support the other Hadoop modules.

2. **Hadoop Distributed File System (HDFS™)**: A distributed file system that provides high-throughput access to application data.

3. **Hadoop YARN**: A framework for job scheduling and cluster resource management.

4. **Hadoop MapReduce**: A YARN-based system for parallel processing of large data sets.

Hadoop-related projects at Apache include:

5. **HBase**: A scalable, distributed database that supports structured data storage for large tables.

6. **Hive**: A data warehouse infrastructure that provides data summarization and ad hoc querying.

7. **Pig**: A high-level data-flow language and execution framework for parallel computation.

8. **Spark**: A fast and general compute engine for Hadoop data. Spark provides a simple and expressive programming model that supports a wide range of applications, including ETL, machine learning, stream processing, and graph computation.

9. **ZooKeeper**: A high-performance coordination service for distributed applications.

**Why Big Data Analytics?**

Big data analytics examines large amounts of data to uncover hidden patterns, correlations and other insights. With today’s technology, it’s possible to analyze data and get answers from it almost immediately – an effort that’s slower and less efficient with more traditional business intelligence solutions.

**Big Data – Government**

By implementing a big data platform, governments can access vast amounts of relevant information important to their daily functions. The positive effect it can have is nearly endless. It’s so important because it not only allows the government to pinpoint areas that need attention, but it also gives them that information in real time. In a society that moves so quickly from one thing to the next, real-time analysis is vital. It allows governments to make faster decisions, and it allows them to monitor those decisions and quickly enact changes if necessary.

**Uses of h1b visa dataset**

Every country needs basic information on its residents for purposes of planning, development and improvement of the residents' quality of life. Good planning is based on reliable,up-to-date, accurate and detailed information on the state of the society in the country. This information makes it possible to plan better services, improve the quality of life and solve existing problems. Statistical information, which serves as the basis for constructing planning forecasts, is essential for the democratic process since it enables the citizens to examine the decisions made by the government and local authorities, and decide whether they serve the public they are meant to help. For these reasons official statistics are collected and published in all countries, world-wide. Thus, for example,while planning a road system, the planners use information regarding the quantity of people and number of vehicles who are likely to use the road; for if not, the advantages of constructing the road may not justify its cost. Similarly, when planning a school system, there is a need for a forecast of the number of pupils who are likely to need schools, in order to ensure that they will be built in locations with an appropriate number of pupils.

**The importance of the employment details**

The h1b visa data set, wide-range activity, which takes place once a decade in the entire country. Its purpose is to gather information about the general population, in order to present a full and reliable picture of the population in the country - its housing conditions and demographic, social and economic characteristics.The information collected includes data on,

**S\_no**

**Case\_status**

**Employer\_name**

**Job\_title**

**Soc\_name**

**Soc\_code**

**Job\_title**

**Full\_time\_position**

**Previling\_wages**

**Year**

**Work\_site**

**Lon**

**Lat**

**Immigration to the United States**

Immigration to the United States is the international movement of individuals who are not natives or do not possess citizenship in order to settle, reside, study or to take-up employment in the United States. It has been a major source of population growth and cultural change throughout much of the history of the United States. The economic, social, and political aspects of immigration have caused controversy regarding ethnicity, economic benefits, jobs for non-immigrants, settlement patterns, impact on upward social mobility, crime, and voting behavior.

As for economic effects, research suggests that immigration to the United States is beneficial to the US economy. Research, with few exceptions, finds that immigration on average has positive economic effects on the native population, but is mixed as to whether low-skilled immigration adversely affects low-skilled natives. Research finds that immigration either has no impact on the crime rate or that it reduces the crime rate in the United States. Research shows that the United States excels at assimilating first- and second-generation immigrants relative to many other Western countries.

The census is extremely important for documenting the growth of immigrant communities, allocating resources for needed services, and identifying areas where civil rights enforcement may be needed. Department of Homeland Security (DHS), United States of America keeps track of all data from the Bureau of the Census, and do statistical analysis to plan its policy and funds.

**Total Data and Statistics**

total data and statistics of the h1b visa data set . that are total for 20111,2012,2013,2014,2015,2016.that consist of serial no, case status ,job title, employer name, latitude ,longitude, worksite ,prevailing wages, years.

**Benefit of h1b visa**

Everyone, including immigrants, benefits from investments in education, health care, and jobs that are distributed based on census information. And census data are also used in ways that are of special importance to immigrants, including:

1.finding the talented persons in each and every country by giving high pay.

2country economy gets improved because of those immigrants

. And the technology gets peroved

3. standary of the people living in the country gets improved

4. evaluating the effectiveness of equal opportunity employment programs and policies under the Civil Rights Act;

5. allocating funds to school districts for children with limited English language proficiency.

**Term in this Project**

The columns in the dataset include:

* CASE\_STATUS: Status associated with the last significant event or decision. Valid values include “Certified,” “Certified-Withdrawn,” Denied,” and “Withdrawn”.

Certified: Employer filed the LCA, which was approved by DOL

Certified Withdrawn: LCA was approved but later withdrawn by employer

Withdrawn: LCA was withdrawn by employer before approval

Denied: LCA was denied by DOL

* EMPLOYER\_NAME: Name of employer submitting labour condition application.
* SOC\_NAME: the Occupational name associated with the SOC\_CODE. SOC\_CODE is the occupational code associated with the job being requested for temporary labour condition, as classified by the Standard Occupational Classification (SOC) System.
* JOB\_TITLE: Title of the job
* FULL\_TIME\_POSITION: Y = Full Time Position; N = Part Time Position
* PREVAILING\_WAGE: Prevailing Wage for the job being requested for temporary labour condition. The wage is listed at annual scale in USD. The prevailing wage for a job position is defined as the average wage paid to similarly employed workers in the requested occupation in the area of intended employment. The prevailing wage is based on the employer’s minimum requirements for the position.
* YEAR: Year in which the H1B visa petition was filed
* WORKSITE: City and State information of the foreign worker’s intended area of employment
* lon: longitude of the Worksite
* lat: latitude of the Worksite

SUCCESS RATE % = (Certified + Certified Withdrawn)/Total x 100

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

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

1(b)Find top 5 job titles who are having highest avg growth in applications

--1 b) Find top 5 job titles who are having highest growth in applications.

load\_data = load '/user/hive/warehouse/project.db/h1b\_final' using PigStorage('\t') as (s\_no:int,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:long,year:chararray,worksite:chararray,longitute:double,latitute:double);

filter\_data = filter load\_data by year =='2011';

group\_data = group filter\_data by job\_title;

count\_data = foreach group\_data generate group,COUNT($1) as total;

order\_data = order count\_data by total desc ;

--dump order\_data ;

filter\_data2 = filter load\_data by year =='2012';

group\_data2 = group filter\_data2 by job\_title;

count\_data2 = foreach group\_data2 generate group,COUNT($1) as total2;

order\_data2 = order count\_data2 by total2 desc ;

--dump order\_data2 ;

filter\_data3 = filter load\_data by year =='2013';

group\_data3 = group filter\_data3 by job\_title;

count\_data3 = foreach group\_data3 generate group,COUNT($1) as total3;

order\_data3 = order count\_data3 by total3 desc ;

--dump order\_data3 ;

filter\_data4 = filter load\_data by year =='2014';

group\_data4 = group filter\_data4 by job\_title;

count\_data4 = foreach group\_data4 generate group,COUNT($1) as total4;

order\_data4 = order count\_data4 by total4 desc ;

filter\_data5 = filter load\_data by year =='2015';

group\_data5 = group filter\_data5 by job\_title;

count\_data5 = foreach group\_data5 generate group,COUNT($1) as total5;

order\_data5 = order count\_data5 by total5 desc ;

filter\_data6 = filter load\_data by year =='2016';

group\_data6 = group filter\_data6 by job\_title;

count\_data6 = foreach group\_data6 generate group,COUNT($1) as total6;

order\_data6 = order count\_data6 by total6 desc ;

join\_data = join order\_data by $0,order\_data2 by $0,order\_data3 by $0,order\_data4 by $0,order\_data5 by $0,order\_data6 by $0;

--dump join\_data ;

generate\_data = foreach join\_data generate $0,$1,$3,$5,$7,$9,$11;

--dump generate\_data ;

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

average\_growth = foreach progressive\_growth generate $0,(float)($1+$2+$3+$4+$5)/5 as avg ;

order\_average = limit(order average\_growth by avg desc) 5;

dump order\_average;

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

load\_data = load '/user/hive/warehouse/project.db/h1b\_final' using PigStorage('\t') as (s\_no:int,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:long,year:chararray,worksite:chararray,longitute:double,latitute:double);

filter\_data1 = filter load\_data by job\_title == 'DATA ENGINEER' and year == '2011';

group\_data1 = group filter\_data1 by worksite;

count\_data1 = foreach group\_data1 generate '2011',group as worksite,COUNT(filter\_data1.job\_title)as headcount;

orderby\_data1 = limit (order count\_data1 by headcount desc) 1;

filter\_data2 = filter load\_data by job\_title == 'DATA ENGINEER' and year == '2012';

group\_data2 = group filter\_data2 by worksite;

count\_data2 = foreach group\_data2 generate '2012',group as worksite,COUNT(filter\_data2.job\_title)as headcount;

orderby\_data2 = limit (order count\_data2 by headcount desc) 1;

filter\_data3 = filter load\_data by job\_title == 'DATA ENGINEER' and year == '2013';

group\_data3 = group filter\_data3 by worksite;

count\_data3 = foreach group\_data3 generate '2013',group as worksite,COUNT(filter\_data3.job\_title)as headcount;

orderby\_data3 = limit (order count\_data3 by headcount desc) 1;

filter\_data4 = filter load\_data by job\_title == 'DATA ENGINEER' and year == '2014';

group\_data4 = group filter\_data4 by worksite;

count\_data4 = foreach group\_data4 generate '2014',group as worksite,COUNT(filter\_data4.job\_title)as headcount;

orderby\_data4 = limit (order count\_data4 by headcount desc) 1;

filter\_data5 = filter load\_data by job\_title == 'DATA ENGINEER' and year == '2015';

group\_data5 = group filter\_data5 by worksite;

count\_data5 = foreach group\_data5 generate '2015',group as worksite,COUNT(filter\_data5.job\_title)as headcount;

orderby\_data5 = limit (order count\_data5 by headcount desc) 1;

filter\_data6 = filter load\_data by job\_title == 'DATA ENGINEER' and year == '2016';

group\_data6 = group filter\_data6 by worksite;

count\_data6 = foreach group\_data6 generate '2016',group as worksite,COUNT(filter\_data6.job\_title)as headcount;

orderby\_data6 = limit (order count\_data6 by headcount desc) 1;

final\_data = union orderby\_data1,orderby\_data2,orderby\_data3,orderby\_data4,orderby\_data5,orderby\_data6;

dump final\_data;

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

load\_data = load '/user/hive/warehouse/project.db/h1b\_final' using PigStorage('\t') as (s\_no:int,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:long,year:chararray,worksite:chararray,longitute:double,latitute:double);

filter\_data1 = filter load\_data by case\_status == 'CERTIFIED' and year == '2011';

group\_data1 = group filter\_data1 by worksite;

count\_data1 = foreach group\_data1 generate '2011',group as worksite,COUNT(filter\_data1.case\_status)as headcount;

orderby\_data1 = limit (order count\_data1 by headcount desc) 5;

filter\_data2 = filter load\_data by case\_status == 'CERTIFIED' and year == '2012';

group\_data2 = group filter\_data2 by worksite;

count\_data2 = foreach group\_data2 generate '2012',group as worksite,COUNT(filter\_data2.case\_status)as headcount;

orderby\_data2 = limit (order count\_data2 by headcount desc) 5;

filter\_data3 = filter load\_data by case\_status == 'CERTIFIED' and year == '2013';

group\_data3 = group filter\_data3 by worksite;

count\_data3 = foreach group\_data3 generate '2013',group as worksite,COUNT(filter\_data3.case\_status)as headcount;

orderby\_data3 = limit (order count\_data3 by headcount desc) 5;

filter\_data4 = filter load\_data by case\_status == 'CERTIFIED' and year == '2014';

group\_data4 = group filter\_data4 by worksite;

count\_data4 = foreach group\_data4 generate '2014',group as worksite,COUNT(filter\_data4.case\_status)as headcount;

orderby\_data4 = limit (order count\_data4 by headcount desc) 5;

filter\_data5 = filter load\_data by job\_title == 'DATA ENGINEER' and year == '2015';

group\_data5 = group filter\_data5 by worksite;

count\_data5 = foreach group\_data5 generate '2015',group as worksite,COUNT(filter\_data5.case\_status)as headcount;

orderby\_data5 = limit (order count\_data5 by headcount desc) 5;

filter\_data6 = filter load\_data by case\_status == 'CERTIFIED' and year == '2016';

group\_data6 = group filter\_data6 by worksite;

count\_data6 = foreach group\_data6 generate '2016',group as worksite,COUNT(filter\_data6.case\_status)as headcount;

orderby\_data6 = limit (order count\_data6 by headcount desc) 5;

final\_data = union orderby\_data1,orderby\_data2,orderby\_data3,orderby\_data4,orderby\_data5,orderby\_data6;

3) Which industry(SOC\_NAME) has the most number of Data Scientist positions?

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

//import org.apache.hadoop.io.NullWritable;

//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.Reducer.Context;

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

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

public class Project\_6 {

public static class mapclass extends Mapper<LongWritable,Text,Text,IntWritable>

{

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

{

try{

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

String soc\_name=record[3];

String job\_title=record[4];

String case\_status=record[1];

if(case\_status.equals("CERTIFIED") && job\_title.equals("DATA SCIENTIST"))

{

context.write(new Text(soc\_name),new IntWritable(1));

}

}

catch(Exception e)

{

System.out.println(e);

}

}

}

public static class reduceclass extends Reducer<Text,IntWritable,Text,IntWritable>

{

int maxcount=0;

String maxkey="";

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

{

int count=0;

for(IntWritable v:val)

{

count+=v.get();

if(count>maxcount)

{

maxcount=count;

maxkey=key.toString();

}

}

// String result = str + "," + max;

//context.write(key,new IntWritable(count));

}

protected void cleanup(Context context) throws IOException,InterruptedException

{

context.write(new Text(maxkey),new IntWritable(maxcount));

}

}

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

{

Configuration conf=new Configuration();

Job job = Job.getInstance(conf, "H1b\_Visa 6th question");

job.setJarByClass(Project\_6.class);

job.setMapperClass(mapclass.class);

job.setReducerClass(reduceclass.class);

job.setNumReduceTasks(1);

// job.setMapOutputKeyClass(Text.class);

// job.setMapOutputValueClass(Text.class);

job.setCombinerClass(reduceclass.class);

// job.setPartitionerClass(part.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

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

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

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

}

}

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

select employer\_name,count(case\_status)as total from h1b\_final where year ='2011' group by employer\_name order by total desc limit 5;

select employer\_name,count(case\_status)as total from h1b\_final where year ='2012' group by employer\_name order by total desc limit 5;

select employer\_name,count(case\_status)as total from h1b\_final where year ='2013' group by employer\_name order by total desc limit 5;

select employer\_name,count(case\_status)as total from h1b\_final where year ='2014' group by employer\_name order by total desc limit 5;

select employer\_name,count(case\_status)as total from h1b\_final where year ='2015' group by employer\_name order by total desc limit 5;

select employer\_name,count(case\_status)as total from h1b\_final where year ='2016' group by employer\_name order by total desc limit 5;

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

a) for all the applications

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.IntWritable;

//import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.NullWritable;

//import org.apache.hadoop.io.NullWritable;

//import org.apache.hadoop.io.NullWritable;

//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.Reducer.Context;

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

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

public class proj5b {

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

{

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

{

try{

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

String job\_title=record[4];

String year=record[7];

String value=1+","+year;

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

}

catch(Exception e)

{

System.out.println(e);

}

}

}

public static class part extends Partitioner<Text,Text>

{

@Override

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

{

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

String year=str[1];

if(year.equals("2011"))

{

return 0;

}

else

if(year.equals("2012"))

{

return 1;

}

else

if(year.equals("2013"))

{

return 2;

}

else

if(year.equals("2014"))

{

return 3;

}

else

if(year.equals("2015"))

{

return 4;

}

else

{

return 5;

}

}

}

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

{

TreeMap<IntWritable,Text> map=new TreeMap<IntWritable,Text>();

String year="";

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

{

int count=0;

for(Text str:val)

{

String[] ar = str.toString().split(",");

count+=Integer.parseInt(ar[0]);

year=ar[1];

}

String value1=year+","+key+","+count;

// String val1=count+","+year;

//context.write(key,new Text(count+","+year));

map.put(new IntWritable(count),new Text(value1));

if(map.size()>5)

{

map.remove(map.firstKey());

}

}

protected void cleanup(Context context) throws IOException,InterruptedException

{

for(Text value2:map.descendingMap().values())

{

context.write(NullWritable.get(),new Text(value2));

}

}

}

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

{

Configuration conf=new Configuration();

Job job = Job.getInstance(conf, "H1b\_Visa 5th question");

job.setJarByClass(proj5b.class);

job.setMapperClass(mapclass.class);

job.setReducerClass(reduceclass.class);

job.setNumReduceTasks(6);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

//job.setCombinerClass(reduceclass.class);

job.setPartitionerClass(part.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);

}

}

b) for only certified applications

. 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.IntWritable;

//import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.NullWritable;

//import org.apache.hadoop.io.NullWritable;

//import org.apache.hadoop.io.NullWritable;

//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.Reducer.Context;

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

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

public class proj5a {

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

{

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

{

try{

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

String job\_title=record[4];

String year=record[7];

String case\_status=record[1];

String value=1+","+year;

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

{

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

}

}

catch(Exception e)

{

System.out.println(e);

}

}

}

public static class part extends Partitioner<Text,Text>

{

@Override

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

{

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

String year=str[1];

if(year.equals("2011"))

{

return 0;

}

else

if(year.equals("2012"))

{

return 1;

}

else

if(year.equals("2013"))

{

return 2;

}

else

if(year.equals("2014"))

{

return 3;

}

else

if(year.equals("2015"))

{

return 4;

}

else

{

return 5;

}

}

}

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

{

TreeMap<IntWritable,Text> map=new TreeMap<IntWritable,Text>();

String year="";

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

{

int count=0;

for(Text str:val)

{

String[] ar = str.toString().split(",");

count+=Integer.parseInt(ar[0]);

year=ar[1];

}

String value1=year+","+key+","+count;

// String val1=count+","+year;

//context.write(key,new Text(count+","+year));

map.put(new IntWritable(count),new Text(value1));

if(map.size()>5)

{

map.remove(map.firstKey());

}

}

protected void cleanup(Context context) throws IOException,InterruptedException

{

for(Text value2:map.descendingMap().values())

{

context.write(NullWritable.get(),new Text(value2));

}

}

}

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

{

Configuration conf=new Configuration();

Job job = Job.getInstance(conf, "H1b\_Visa 5th question");

job.setJarByClass(proj5a.class);

job.setMapperClass(mapclass.class);

job.setReducerClass(reduceclass.class);

job.setNumReduceTasks(6);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

//job.setCombinerClass(reduceclass.class);

job.setPartitionerClass(part.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);

}

}

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

create table totalcount(total bigint)

row format delimited

fields terminated by ',';

insert overwrite table totalcount

select count(case\_status) from h1b\_final;

select a.case\_status,count(a.case\_status) as case\_count,round((count(a.case\_status)/total\*100),2) as case\_percent from h1b\_final a,totalcount b group by a.case\_status,b.total;

7) Create a bar graph to depict the number of applications for each year

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.FloatWritable;

import org.apache.hadoop.io.IntWritable;

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 proj7 {

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

{

private final static LongWritable one=new LongWritable(1);

String key1="all";

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

{

try

{

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

String case\_status=new String(str[1]);

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

}

catch(Exception e)

{

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

}

}

}

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

{

IntWritable result=new IntWritable();

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

{

long count=0;

int sum=0;

for(LongWritable val:value)

{

count+=val.get();

}

context.write(new Text(key), new LongWritable(count));

}

}

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

Configuration conf = new Configuration();

//conf.set("name", "value")

//conf.set("mapreduce.input.fileinputformat.split.minsize", "134217728");

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

job.setJarByClass(proj7.class);

job.setMapperClass(MapClass.class);

//job.setCombinerClass(ReduceClass.class);

job.setReducerClass(ReduceClass.class);

job.setNumReduceTasks(1);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(LongWritable.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);

}

}

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

select year,job\_title,round(avg(prevailing\_wage),2)as total from h1b\_final where (case\_status="CERTIFIED" or case\_status="CERTIFIED-WITHDRAWN")and full\_time\_position="N" group by year,job\_title order by total desc;

select year,job\_title,round(avg(prevailing\_wage),2)as total from h1b\_final where (case\_status="CERTIFIED" or case\_status="CERTIFIED-WITHDRAWN")and full\_time\_position="Y" group by year,job\_title order by total desc;

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) ?

load\_data = load '/user/hive/warehouse/project.db/h1b\_final' using PigStorage('\t') as (s\_no:int,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:long,year:chararray,worksite:chararray,longitute:double,latitute:double);

groupby\_emp = group load\_data by employer\_name;

count\_total = foreach groupby\_emp generate group,COUNT(load\_data.case\_status) as totalcount;

filter\_certified = filter load\_data by case\_status == 'CERTIFIED';

group\_certified\_filter\_data = group filter\_certified by employer\_name;

count\_certified = foreach group\_certified\_filter\_data generate group,COUNT(filter\_certified.case\_status) as certifiedcount;

filter\_certified\_withdrawn = filter load\_data by case\_status == 'CERTIFIED-WITHDRAWN';

group\_certified\_withdrawn\_filter\_data = group filter\_certified\_withdrawn by employer\_name;

count\_certified\_withdrawn = foreach group\_certified\_withdrawn\_filter\_data generate group,COUNT(filter\_certified\_withdrawn.case\_status) as

certified\_withdrawncount;

join\_data = join count\_total by $0,count\_certified by $0,count\_certified\_withdrawn by $0;

join\_data1 = foreach join\_data generate $0,$1,$3,$5 ;

successive\_rate = foreach join\_data1 generate $0,(float)($2+$3)/($1)\*100,$1;

final = filter successive\_rate by $1 > 70 and $2 > 1000;

final\_output = order final by $1 desc;

dump final\_output;

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)?

load\_data = load '/user/hive/warehouse/project.db/h1b\_final' using PigStorage('\t') as (s\_no:int,case\_status:chararray,employer\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_position:chararray,prevailing\_wage:long,year:chararray,worksite:chararray,longitute:double,latitute:double);

groupby\_emp = group load\_data by job\_title;

count\_total = foreach groupby\_emp generate group,COUNT(load\_data.case\_status) as totalcount;

filter\_certified = filter load\_data by case\_status == 'CERTIFIED';

group\_certified\_filter\_data = group filter\_certified by job\_title;

count\_certified = foreach group\_certified\_filter\_data generate group,COUNT(filter\_certified.case\_status) as certifiedcount;

filter\_certified\_withdrawn = filter load\_data by case\_status == 'CERTIFIED-WITHDRAWN';

group\_certified\_withdrawn\_filter\_data = group filter\_certified\_withdrawn by job\_title;

count\_certified\_withdrawn = foreach group\_certified\_withdrawn\_filter\_data generate group,COUNT(filter\_certified\_withdrawn.case\_status) as

certified\_withdrawncount;

join\_data = join count\_total by $0,count\_certified by $0,count\_certified\_withdrawn by $0;

--dump join\_data;

join\_data1 = foreach join\_data generate $0,$1,$3,$5 ;

successive\_rate = foreach join\_data1 generate $0,(float)($2+$3)/($1)\*100,$1;

final = filter successive\_rate by $1 > 70 and $2 > 1000;

final\_output = order final by $1 desc;

dump final\_output;

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

**#Create a Database in mysql and create a table in it**

**mysql -u root -p'1234' -e 'drop database project;create database if not exists project;use project;create table q10(job\_title varchar(100),success\_rate float,petitions int);';**

**sqoop export --connect jdbc:mysql://localhost/project --username root --password '1234' --table q10 --update-mode allowinsert --export-dir /user/hive/ten1/p\* --input-fields-terminated-by '\t'**

**mysql -u root -p'1234' -e 'select \* from project.q10';**

**show\_menu;suggestions:**

1) Native people standard should be improved and more jobs to be created for natives.

2) Number of denied case status is very less . Must examine the candidate before giving the visa clearance .

3) Number of part-time job is full compare to the full time job . can increase the part-time job candidate.

4) Number of job vacancies in the Florida is must be improved.

**Conclusion:**

1) H1b data set project shows that the amount of case status that are with\_drawn and denied in year wise.

2) From the H1b dataset the data engineer job title increases over time.

3) And shows that in year 2011 new jersey

Is the state that has most number of data scientists each year

4) Top job position which the H1b visa is applied and in year wise.

5) The graph shows that number of applications increases over time.

6) Most number of data scientists are available in statistics job location.