***ABSTRACT***

The H1B Visa is the primary US work visa / permit made available to people from all over the world. The US Government introduced the H1B visa program to offer and enable skilled International Professionals and International Students from all over the World the opportunity to live and work in the USA.The H1B is the most popular and sought after US work visa and US Immigration requires 'every' foreign national to obtain a visa in order to legally work in America.

One of the main advantages of the H1B visa is that it is a 'dual intent' visa which means that you can apply for a Green Card (become a Legal Permanent Resident). It is important for the U.S. Department of Labor to identify which employers are sending the most number of H1B Visa applications and what is the percentage share of the annual 85,000 visa cap for the employers withmost applications. At the same time,to figure out the most common job title, examine the data science jobs, and their relative wages for which the application is coming in larger frequency. This study provides information about the employment gaps within the U.S.,which are filled by foreign workers.This gives an idea as to which areas theU.S. government should emphasize to encourage domestic students to develop more local workforce in those job domains for the future.

***INTRODUCTION***

The H1B is an employment-based, non-immigrant visa category for temporary foreign workers in the United States. For a foreign national to apply for H1B visa, an US employer must offer a job and petition for H1B visa with the US immigration department. This is the most common visa status applied for and held by international students once they complete college/ higher education (Masters, Ph.D.) and work in a full-time position. The U.S. Department of Labor (DOL) is responsible for ensuring that foreign workers do not displace or adversely affect wages or working conditions of U.S. workers. For every H-1B petition filed with the USCIS, there must be

included a Labor Condition Application (LCA), certified by the U.S. Department of Labor. The LCA is designed to ensure that the wage offered to the non-immigrant worker meets or exceeds the "prevailing wage" in the area of employment. The LCA also contains an attestation section designed to prevent the program from being used to import foreign workers to break a strike or replace U.S. citizen workers.

While an employer is not required to advertise the position before hiring an H-1B non-immigrant pursuant to the H-1B visa approval, the employer must notify the employee representative about the Labor Condition Application (LCA)—or if there is no such representation, the employer must publish the LCA at the workplace and the employer's office.

Under the regulations, LCAs are a matter of public record. Corporations hiring H-1B workers are required to make these records available to any member of the public who requests to look at them. Copies of the relevant records are also available from various web sites, including the Department of Labor.

***OBJECTIVES***

1. Data collection and production of information for government ministries and local authorities, for budgeting purposes.

2. Production of information which serves bodies, organizations and various other elements in the fields of education, the economy, business, research, etc.

3. Decision-making that facilitates the development of socio-economic policies-enhance the welfare of the population.

4. Processing and analyzing large amount of raw data by using map-reduce programming model and distributed computing on HADOOP framework to improve time and complexity.

***BIG DATA***

BIG DATA as termed in wikipedia:

Big data is data sets that are so voluminous and complex that traditional data-processing application software are inadequate to deal with them. Big data challenges include capturing data, data storage, data analysis, search, sharing, transfer, visualization, querying, updating, information privacy and data source. There are a number of concepts associated with big data: originally there were 3 concepts volume, variety, velocity.Other concepts later attributed with big data are veracity (i.e., how much noise is in the data) and value.

Volume refers to the vast amounts of data generated every second. Just think of all the emails, twitter messages, photos, video clips, sensor data etc. we produce and share every second. We are not talking Terabytes but Zettabytes or Brontobytes. On Facebook alone we send 10 billion messages per day, click the "like' button 4.5 billion times and upload 350 million new pictures each and every day.

Velocity refers to the speed at which new data is generated and the speed at which data moves around. Just think of social media messages going viral in seconds, the speed at which credit card transactions are checked for fraudulent activities, or the milliseconds it takes trading systems to analyse social media networks to pick up signals that trigger decisions to buy or sell shares.

Variety refers to the different types of data we can now use. In the past we focused on structured data that neatly fits into tables or relational databases, such as financial data (e.g. sales by product or region). In fact, 80% of the world’s data is now unstructured, and therefore can’t easily be put into tables (think of photos, video sequences or social media updates). With big data technology we can now harness differed types of data (structured and unstructured) including messages, social media conversations, photos, sensordata, video or voice recordings and bring them together with more traditional, structured data.

Veracity refers to the messiness or trustworthiness of the data. With many forms of big data, quality and accuracy are less controllable (just think of Twitter posts with hash tags, abbreviations, typos and colloquial speech as well as the reliability and accuracy of content) but big data and analytics technology now allows us to work with these type of data. The volumes often make up for the lack of quality or accuracy.

Value: Then there is another V to take into account when looking at Big Data: Value! It is all well and good having access to big data but unless we can turn it into value it is useless. So you can safely argue that 'value' is the most important V of Big Data. It is important that businesses make a business case for any attempt to collect and leverage big data.

***HADOOP ARCHITECTURE***

Apache Hadoop offers a scalable, flexible and reliable distributed computing big data framework for a cluster of systems with storage capacity and local computing power by leveraging commodity hardware. Hadoop follows a Master Slave architecture for the transformation and analysis of large datasets using Hadoop MapReduce paradigm. The 3 important hadoop components that play a vital role in the Hadoop architecture are -

1. Hadoop Distributed File System (HDFS) – Patterned after the UNIX file system
2. Hadoop MapReduce
3. Yet Another Resource Negotiator (YARN)

Haddop 1.x

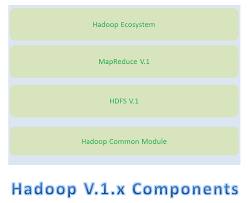
It was first version of hadoop released on 2004.

Since 2004, Hadoop has emerged as the de facto standard to store, process, and analyze hundreds of terabytes and even petabytes of data.

The major components in Hadoop 1.x are as follows:

1. HDFS V.1

2. MAPREDUCE V.1

**

Drawbacks of Hadoop 1.x

Hadoop 1.x has the following Limitations/Drawbacks:

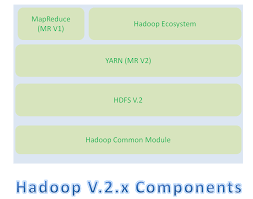
* It is only suitable for Batch Processing of Huge amount of Data, which is already in Hadoop System.
* It is not suitable for Real-time Data Processing.
* It is not suitable for Data Streaming.
* It supports upto 4000 Nodes per Cluster.
* It has a single component : JobTracker to perform many activities like Resource Management, Job Scheduling, Job Monitoring, Re-scheduling Jobs etc.
* JobTracker is the single point of failure.
* It does not support Multi-tenancy Support.
* It supports only one Name Node and One Namespace per Cluster.
* It does not support Horizontal Scalability.
* It runs only Map/Reduce jobs.
* It follows Slots concept in HDFS to allocate Resources (Memory, RAM, CPU). It has static Map and Reduce Slots. That means once it assigns resources to Map/Reduce jobs, it cannot re-use them even though some slots are idle.

Hadoop 2.x:

Apache Hadoop V.2.x has the following three major Components

1. HDFS V.2
2. YARN (MR V2)
3. MapReduce (MR V1)

In Hadoop V.2.x, these two are also know as Three Pillars of Hadoop.



Apache Hadoop 2.0 represents a generational shift in the architecture of Apache Hadoop. With YARN, Apache Hadoop is recast as a significantly more powerful platform – one that takes Hadoop beyond merely batch applications to taking its position as a ‘data operating system’ where HDFS is the file system and YARN is the operating system.

YARN is a re-architecture of Hadoop that allows multiple applications to run on the same platform.

The fundamental idea of YARN is to split up the two major responsibilities of the JobTracker and TaskTracker into separate entities.

***TOOLS USED***

**APACHE HDFS:**The Hadoop Distributed File System (HDFS) is designed to store very large data sets reliably, and to stream those data sets at high bandwidth to user applications. In a large cluster, thousands of servers both host directly attached storage and execute user application tasks. By distributing storage and computation across many servers, the resource can grow with demand while remaining economical at every size. We describe the architecture of HDFS and report on experience using HDFS to manage 40 petabytes of enterprise data at Yahoo.

**APACHE MAPREDUCE:**Hadoop MapReduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.The MapReduce framework consists of a single master JobTracker and one slave TaskTracker per cluster-node. The master is responsible for scheduling the jobs’ component tasks on the slaves, monitoring them and re-executing the failed tasks. The slaves execute the tasks as directed by the master.

**APACHE HIVE:**The Apache Hive data warehouse software facilitates querying and managing large datasets residing in distributed storage. Hive provides a mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL. At the same time this language also allows traditional map/reduce programmers to plug in their custom mappers and reducers when it is inconvenient or inefficient to express this logic in HiveQL

**APACHE PIG**:Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets. At the present time, Pig’s infrastructure layer consists of a compiler that produces sequences of Map-Reduce programs, for which large-scale parallel implementations already exist (e.g., the Hadoop subproject). Pig’s language layer currently consists of a textual language called Pig Latin.

**APACHE SCOOP:**Sqoop is a tool designed to transfer data between Hadoop and relational databases. You can use Sqoop to import data from a relational database management system (RDBMS) such as MySQL or Oracle into the Hadoop Distributed File System (HDFS), transform the data in Hadoop MapReduce, and then export the data back into an RDBMS.

**MYSQL:**A database is a separate application that stores a collection of d ata. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.

Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those type of systems.

***PROJECT OUTLINE***

|  |  |
| --- | --- |
| Title | Analysing H1B Applications Using Hadoop |
| Input | h1b.csv (h1b visa data) |
| Data Elements | S\_no, case\_status, employer\_name, soc\_name, job\_title, full\_time\_position, prevailing\_wage, year, worksite, longitute, latitute. |
| Purpose | Performing analysis on the H1B visa applicants between the years 2011-2016, to identify the growth cycle is increasing or not, which industry has granted more visa’s etc. |

**SAMPLE DATA:**

****

**Data set column description.**

1. CASE\_STATUS: Status associated with the last significant event or decision. Valid values

include.

\* "Certified": Employer filed the LCA, which was approved by DOL.

\* "Certified-Withdrawn": LCA was approved but later withdrawn by employer.

\* "Denied": LCA was denied by DOL.

\* "Withdrawn": LCA was withdrawn by employer before.

2. EMPLOYER\_NAME: Name of employer submitting labor condition application.

3. SOC\_NAME: the Occupational name associated with the SOC\_CODE. SOC\_CODE is the occupational code associated with the job being requested for temporary labor condition, as classified by the Standard Occupational Classification (SOC) System.

4. JOB\_TITLE: Title of the job: FULL\_TIME\_POSITION

\* Y = Full Time Position.

\* N = Part Time Position.

5. PREVAILING\_WAGE: Prevailing Wage for the job being requested for temporary labor 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.

6. YEAR: Year in which the H1B visa petition was filed.

7. WORKSITE: City and State information of the foreign worker’s intended area of employment.

8. lon: Longitude of the Worksite.

9. lat: Latitude of the Worksite.

***ANALYSIS***

Following are Analysis that are done:

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

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

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

4) Find top 5 locations in the US who have got certified visa for each year.

5) Which industry has the most number of Data Scientist positions?

6) Which top 5 employers file the most petitions each year?

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

8) Find the percentage and the count of each case status on total applications for each year.

Create a graph depicting the pattern of all the cases over the period of time.

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

10) find the average Prevailing Wage for each Job for each Year (take part time and full time

separate).Arrange the output in descending order.

11) Which are employers along with the number of petitions who have the success rate more

than 70% in petitions and total petitions filed more than 1000?

12) Which are the job positions along with the number of petitions which have the success rate

more than 70% in petitions and total petitions filed more than 1000?

Source code and results of analysis can be found at Github .

**START OF START.SH**

**#!/bin/bash**

**show\_menu()**

**{**

**NORMAL=`echo "\033[m"`**

**MENU=`echo "\033[36m"` #Blue**

**NUMBER=`echo "\033[33m"` #yellow**

**FGRED=`echo "\033[41m"`**

**RED\_TEXT=`echo "\033[31m"`**

**ENTER\_LINE=`echo "\033[33m"`**

**echo -e "${MENU}\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*H1B APPLICATIONS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 1)${MENU} Is the number of petitions with Data Engineer job title increasing over time? ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 2)${MENU} Find top 5 job titles who are having highest avg growth in applications.${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 3)${MENU} Which part of the US has the most Data Engineer jobs for each year? ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 4)${MENU} find top 5 locations in the US who have got certified visa for each year. ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 5)${MENU} Which industry has the most number of Data Scientist positions? ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 6)${MENU} Which top 5 employers file the most petitions each year? ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 7)${MENU}Find the most popular top 10 job positions for H1B visa applications for each year?${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 8)${MENU} 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 period of time. ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 9)${MENU} Create a bar graph to depict the number of applications for each year. ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 10)${MENU} Find the average Prevailing Wage for each Job for each Year Arrange the output in descending order. ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 11)${MENU} Which are the employers along with the number of petitions who have the success rate more than 70% in petitions? ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 12)${MENU} Which are the job positions along with the number of petitions which have the success rate more than 70% in petitions? ${NORMAL}"**

**echo -e "${MENU}#->${NUMBER} 13)${MENU} Export result for question no 10 to MySql database. ${NORMAL}"**

**echo -e "${MENU}\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*${NORMAL}"**

**echo -e "${ENTER\_LINE}Please enter a menu option and enter or ${RED\_TEXT}enter to exit. ${NORMAL}"**

**read opt**

**}**

**function option\_picked()**

**{**

**COLOR='\033[01;31m' # bold red**

**RESET='\033[00;00m' # normal white**

**MESSAGE="$1" #modified to post the correct option selected**

**echo -e "${COLOR}${MESSAGE}${RESET}"**

**}**

**clear**

**start-all.sh | zenity --progress --width 150 --title="Hadoop Services Starting" --pulsate --auto-close #--percentage**

**yad --info --title="Project" --text '<span foreground="red" font="14">\t\t\tWelcome To BigData Project\n</span><span font="12">\n<b>\tAnalysis And Summarization Of H1B Applicants</b>\n</span>' --width=450 --height=10 --button="gtk-cancel:252" --button="gtk-ok:0" --center --timeout 3**

**show\_menu**

**while [ opt != '' ]**

**do**

**if [[ $opt = "" ]]; then**

**exit;**

**else**

**case $opt in**

**1) clear;**

**option\_picked " Is the number of petitions with Data Engineer job title increasing over time?";**

**hadoop fs -rmr /h1bvisa/qns1a.out**

**hadoop jar '/home/hduser/JarFiles/qn1a.jar' Q1A /niit/h1b1visa /h1bvisa/qns1a.out**

**hadoop fs -cat /h1bvisa/qns1a.out/p\***

**sleep 5;**

**show\_menu;**

**;;**

**2) clear;**

**option\_picked " Find top 5 job titles who are having highest growth in applications. ";**

**hive -e "select \* from h1bvisa.totalavg" > /home/hduser/Desktop/1b.txt | zenity --progress --title="Hive Job Running" --pulsate --auto-close**

**cat /home/hduser/Desktop/1b.txt**

**sleep 5;**

**show\_menu;**

**;;**

**3) clear;**

**option\_picked " Which part of the US has the most Data Engineer jobs for each year?";**

**pig -x local '/home/hduser/Documents/H1bvisaOP/Pig/2aoutput.pig/2a.pig'**

**sleep 5;**

**show\_menu;**

**;;**

**4) clear;**

**option\_picked 2 " find top 5 locations in the US who have got certified visa for each year.";**

**echo -e "Enter the year (2011,2012,2013,2014,2015,2016)"**

**read var**

**hive -e " select year, worksite,count(case\_status) as total\_case\_status from h1bvisa.h1b\_final where year =$var and case\_status='CERTIFIED' group by worksite,year order by total\_case\_status desc limit 5;"**

**sleep 5;**

**show\_menu;**

**;;**

**5) clear;**

**option\_picked " Which industry has the most number of Data Scientist positions?";**

**hadoop fs -rmr /h1bvisa/qns3.out**

**hadoop jar '/home/hduser/JarFiles/3.jar' qns3 /niit/h1b1visa /h1bvisa/qns3.out | zenity --progress --title="Mapreduce Job Running" --pulsate --auto-close**

**echo "The industry which has the most number of Data Scientist positions:"**

**hadoop fs -cat /h1bvisa/qns3.out/p\***

**sleep 5**

**show\_menu;**

**;;**

**6) clear;**

**option\_picked "Which top 5 employers file the most petitions each year?";**

**echo -e "Enter the year (2011,2012,2013,2014,2015,2016)"**

**read var**

**rm -r /home/hduser/Desktop/6.txt**

**pig -x local -f '/home/hduser/Documents/H1bvisaOP/Pig/4.pig' -param s\_year=$var | zenity --progress --title="Pig Job Running" --pulsate --auto-close**

**echo -e "The top 5 employers file the most petitions each year:\n"**

**cat /home/hduser/Desktop/6.txt/p\***

**sleep 5**

**show\_menu;**

**;;**

**7) clear;**

**option\_picked "Find the most popular top 10 job positions for H1B visa applications for each year?";**

**echo -e "${MENU}\*\*${NUMBER} 1)${MENU}Top 10 job positions for all 6 years ${NORMAL}"**

**echo -e "${MENU}\*\*${NUMBER} 2)${MENU}Top 10 job positions for the choosen year ${NORMAL}"**

**echo -e "${MENU}\*\*${NUMBER} 3)${MENU}Top 10 certified job positions for all 6 years ${NORMAL}"**

**echo -e "${MENU}\*\*${NUMBER} 4)${MENU}Top 10 certified job positions for choosen year ${NORMAL}"**

**read n**

**case $n in**

**1) echo "Top 10 job positions for all 6 years :"**

**hadoop fs -rmr /h1bvisa/qns5a.out**

**hadoop jar '/home/hduser/JarFiles/5a.jar' qns5a /niit/h1b1visa /h1bvisa/qns5a.out | zenity --progress --title="Mapreduce Job Running" --pulsate --auto-close**

**echo -e "\n Resulatant most popular top 10 job positions for H1B visa applications for all 6 years?\n"**

**hadoop fs -cat /h1bvisa/qns5a.out/p\***

**sleep 5**

**;;**

**2) echo "Top 10 job positions for the choosen year:"**

**echo "Enter the year (2011,2012,2013,2014,2015,2016)"**

**read year**

**echo "You've selected ${year}"**

**rm -r /home/hduser/Desktop/5a.txt**

**hive -e "select \* from h1bvisa.5aout where year = $year limit 10;"> /home/hduser/Desktop/5a.txt | zenity --progress --title="Hive Job Running" --pulsate --auto-close**

**echo -e "\Resultant the most popular top 10 job positions for H1B visa applications for year $year?\n";**

**cat /home/hduser/Desktop/5a.txt**

**sleep 5**

**;;**

**3) echo "Top 10 certified job positions for all 6 years:"**

**hadoop fs -rmr /h1bvisa/qns5b.out**

**hadoop jar '/home/hduser/JarFiles/5b.jar' qns5b /niit/h1b1visa /h1bvisa/qns5b.out | zenity --progress --title="Mapreduce Job Running" --pulsate --auto-close**

**echo -e "\nResultant the most popular top 10 certified job positions for H1B visa applications for all 6 years?\n";**

**hadoop fs -cat /h1bvisa/qns5b.out/p\***

**sleep 5**

**;;**

**4) echo "Top 10 certified job positions for choosen year:"**

**echo "Enter the year (2011,2012,2013,2014,2015,2016)"**

**read year**

**echo "You've selected ${year}"**

**rm -r /home/hduser/Desktop/5b.txt**

**hive -e "select \* from h1bvisa.5bout where year = $year limit 10; " > /home/hduser/Desktop/5b.txt | zenity --progress --title="Hive Job Running" --pulsate --auto-close**

**echo -e "\nResultant the most popular top 10 certified job positions for H1B visa applications for year $year?\n";**

**cat /home/hduser/Desktop/5b.txt**

**sleep 5**

**;;**

**\*) echo "Please Select one among the option[1-4]";;**

**esac**

**show\_menu;**

**;;**

**8) clear;**

**option\_picked " Find the percentage and the count of each case status on total applications for each year.";**

**pig -x local '/home/hduser/Documents/H1bvisaOP/Pig/6output.pig/6.pig'**

**show\_menu;**

**;;**

**9) clear;**

**option\_picked " the number of applications for each year ";**

**hadoop fs -rmr /h1bvisa/qns7.out**

**hadoop jar '/home/hduser/JarFiles/qns7.jar' qns7 /niit/h1b1visa /h1bvisa/qns7.out | zenity --progress --title="Job Running" --pulsate --auto-close**

**echo -e "\n Resulatant most popular top 10 job positions for H1B visa applications for all 6 years?\n"**

**hadoop fs -cat /h1bvisa/qns7.out/p\***

**sleep 5**

**show\_menu;**

**;;**

**10) clear;**

**option\_picked " Find the average Prevailing Wage for each Job for each Year (take part time and full time separate) arrange output in descending order";**

**echo -e "Enter the year(2011,2012,2013,2014,2015,2016)"**

**read year**

**echo -e "Enter the choice Full time/ Part time.(Y/N)"**

**read var**

**hive -e "select job\_title,full\_time\_position,year,avg(prevailing\_wage) as average from h1bvisa.h1b\_final where full\_time\_position = '$var' and year = $year and case\_status in('CERTIFIED','CERTIFIED-WITHDRAWN') group by job\_title,full\_time\_position,year order by average desc limit 5;"**

**show\_menu;**

**;;**

**11) clear;**

**option\_picked " Which are employers who have the highest success rate in petitions more than 70% in petitions and total petions filed more than 1000?";**

**rm -r /home/hduser/Desktop/9.txt**

**hive -e "select \* from h1bvisa.total1 limit 10;"> /home/hduser/Desktop/9.txt | zenity --progress --title="Hive Job Running" --pulsate --auto-close**

**cat /home/hduser/Desktop/9.txt**

**sleep 5**

**show\_menu;**

**;;**

**12) clear;**

**option\_picked "Which are the top 10 job positions which have the success rate more than 70% in petitions and total petitions filed more than 1000";**

**rm -r /home/hduser/Desktop/10.txt**

**pig -x local '/home/hduser/Documents/H1bvisaOP/Pig/10output.pig/10.pig'> /home/hduser/Desktop/10.txt| zenity --progress --title="Pig Job Running" --pulsate --auto-close**

**cat /home/hduser/Desktop/10.txt**

**sleep 5**

**show\_menu;**

**;;**

**13) clear;**

**option\_picked " Export result for question no 10 to MySql database.";**

**hadoop fs -rm -r -f /Pig/Qns10.out**

**hadoop fs -mkdir -p /Pig/Qns10.out**

**hadoop fs -put /home/hduser/Documents/H1bvisaOP/Pig/10output.pig/p\* /Pig/Qns10.out**

**mysql -u root -p -e 'create database if not exists h1bvisa\_job\_success\_rate;use h1bvisa\_job\_success\_rate;create table if not exists job\_success\_rate(job\_title varchar(100),success\_rate float,petitions int);';**

**sqoop export --connect jdbc:mysql://localhost/h1bvisa\_job\_success\_rate --username root --password 'root' --table job\_success\_rate --update-mode allowinsert --update-key job\_title --export-dir /Pig/Qns10.out/p\* --input-fields-terminated-by '\t' ;**

**echo -e '\n\nDisplay contents from MySQL Database.\n\n'**

**echo -e '\n The top 10 job positions that have success rate more than 70% in petitions and total petitions filed more than 1000?\n\n'**

**mysql -u root -p -e 'select \* from h1bvisa\_job\_success\_rate.job\_success\_rate limit 10';**

**sleep 5;**

**show\_menu;**

**;;**

**esac**

**fi**

**done**

**END OF START.SH**

***TABLE CREATION***

Creating the table in Hive.

create database h1bvisa;

use h1bvisa;

CREATE TABLE h1b\_applications(s\_no int,case\_status string,

employer\_name string, soc\_name string, job\_title string,

full\_time\_position string,prevailing\_wage bigint,year string, worksite

string, longitute double, latitute double )

ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'

WITH SERDEPROPERTIES (

"separatorChar" = ",",

"quoteChar" = "\""

) STORED AS TEXTFILE;

load data local inpath '/home/hduser/Downloads/Project files/h1b.csv'

overwrite into table

h1b\_applications;

CREATE TABLE h1b\_app2(s\_no int,case\_status string, employer\_name

string, soc\_name string, job\_title string, full\_time\_position

string,prevailing\_wage bigint,year string, worksite string, longitute

double, latitute double )

row format delimited

fields terminated by '\t'

STORED AS TEXTFILE;

INSERT OVERWRITE TABLE h1b\_app2 SELECT regexp\_replace(s\_no, "\t", ""),

regexp\_replace(case\_status, "\t", ""), regexp\_replace(employer\_name,

"\t", ""), regexp\_replace(soc\_name, "\t", ""),

regexp\_replace(job\_title, "\t", ""),

regexp\_replace(full\_time\_position, "\t", ""), prevailing\_wage,

regexp\_replace(year, "\t", ""), regexp\_replace(worksite, "\t", ""),

regexp\_replace(longitute, "\t", ""), regexp\_replace(latitute, "\t",

"") FROM h1b\_applications where case\_status != "NA";

CREATE TABLE h1b\_final(s\_no int,case\_status string, employer\_name string,

soc\_name string, job\_title string, full\_time\_position string,prevailing\_wage

bigint,year string, worksite string, longitute double, latitute double ) row

format delimited fields terminated by '\t' STORED AS TEXTFILE;

INSERT OVERWRITE TABLE h1b\_final SELECT s\_no,case when trim(case\_status) =

"PENDING QUALITY AND COMPLIANCE REVIEW - UNASSIGNED" then "DENIED" when

trim(case\_status) = "REJECTED" then "DENIED" when trim(case\_status) =

"INVALIDATED" then "DENIED" else case\_status end, employer\_name, soc\_name,

job\_title, full\_time\_position, case when prevailing\_wage is null then 100000

else prevailing\_wage end,year, worksite, longitute, latitute FROM h1b\_app2;

***SOURCE CODE AND OUTPUT OF H1BVISA DATA ANALYSIS***

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

**Tool used:MAPREDUCE**

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.FileSystem;

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.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 Q1A {

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

{

Text myKey = new Text();

IntWritable one = new IntWritable(1);

@Override

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

{

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

String job\_title = record[4];

String year = record[7];

if(job\_title.contains("DATA ENGINEER") && job\_title != null)

{

myKey.set(year);

context.write(myKey, one);

}

}

}

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

{

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

int sum=0;

for(IntWritable val:value)

{

sum+=val.get();

}

context.write(key, new IntWritable(sum));

}

}

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

{

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

{

context.write(new Text("common"),value);

}

}

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

{

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

int count2011 = 1;

int count2012 = 1;

int count2013 = 1;

int count2014 = 1;

int count2015 = 1;

int count2016 = 1;

double growthavg = 0.00;

for(Text v:value)

{

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

if (val[0].equals("2011"))

{

count2011 = Integer.parseInt(val[1]);

}

if (val[0].equals("2012"))

{

count2012 = Integer.parseInt(val[1]);

}

if (val[0].equals("2013"))

{

count2013 = Integer.parseInt(val[1]);

}

if (val[0].equals("2014"))

{

count2014 = Integer.parseInt(val[1]);

}

if (val[0].equals("2015"))

{

count2015 = Integer.parseInt(val[1]);

}

if (val[0].equals("2016"))

{

count2016 = Integer.parseInt(val[1]);

}

}

growthavg = ((((count2012-count2011)\*100)/count2011) + (((count2013-count2012)\*100)/count2012) + (((count2014-count2013)\*100)/count2013) + (((count2015-count2014)\*100)/count2014) + (((count2016-count2015)\*100)/count2015))/5 ;

if (growthavg>0.00)

{

context.write(NullWritable.get(), new Text("The growth is positive : " + String.format("%f", growthavg)));

}

}

}

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

Configuration conf = new Configuration();

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

job.setJarByClass(Q1A.class);

job.setMapperClass(MyMapper.class);

job.setReducerClass(MyReducer.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

Path outputPath1 = new Path("FirstMapper");

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

FileOutputFormat.setOutputPath(job, outputPath1);

FileSystem.get(conf).delete(outputPath1, true);

job.waitForCompletion(true);

Job job2 = Job.getInstance(conf," ");

job2.setJarByClass(Q1A.class);

job2.setMapperClass(GrowthMapper.class);

job2.setReducerClass(GrowthReducer.class);

//job2.setNumReduceTasks(0);

job2.setMapOutputKeyClass(Text.class);

job2.setMapOutputValueClass(Text.class);

job2.setOutputKeyClass(NullWritable.class);

job2.setOutputValueClass(Text.class);

FileInputFormat.addInputPath(job2,outputPath1);

FileOutputFormat.setOutputPath(job2, new Path(args[1]));

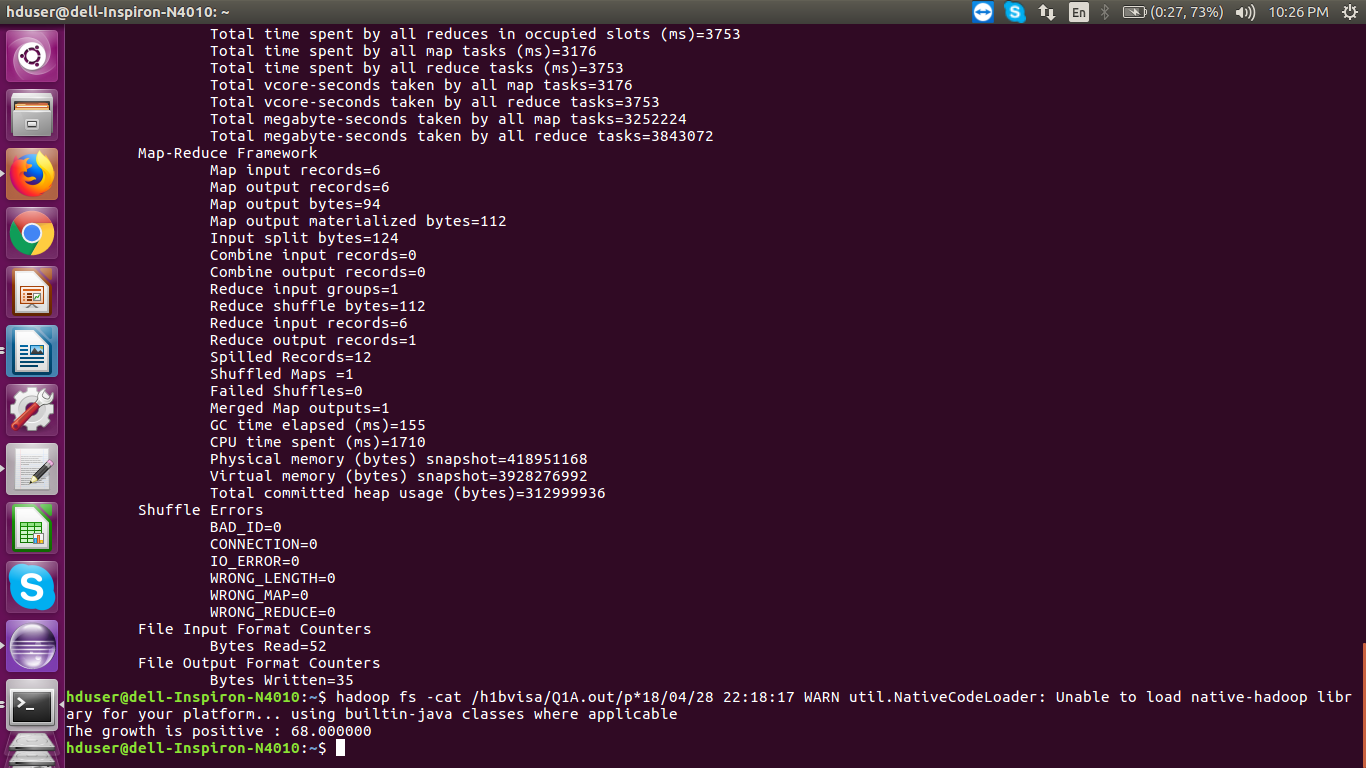
System.exit(job2.waitForCompletion(true) ? 0 :1);

}

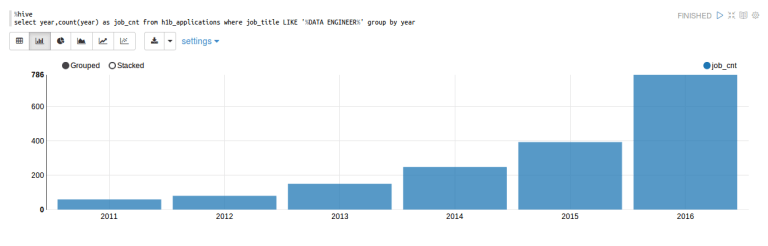
}

OUTPUT:

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



***The graph dicpicting the growth over the 6 years***



**1B.Find top 5 job titles who are having highest avg growth in applications**

**Tool used :HIVE**

fetching the petitions for each year for all job\_titles

----------------------------------------------------------------------

create table 1bavggrowth2011(year1 string,job string,p1 int)

row format delimited

fields terminated by '\t'

stored as textfile;

insert overwrite table 1bavggrowth2011 select year,LOWER(job\_title),count(\*) as p from h1b\_final where year = 2011 group by year,job\_title;

create table 1bavggrowth2012(year2 string,job string,p2 int)

row format delimited

fields terminated by '\t'

stored as textfile;

insert overwrite table 1bavggrowth2012 select year,LOWER(job\_title),count(\*) as p from h1b\_final where year = 2012 group by year,job\_title;

create table 1bavggrowth2013(year3 string,job string,p3 int)

row format delimited

fields terminated by '\t'

stored as textfile;

insert overwrite table 1bavggrowth2013 select year,LOWER(job\_title),count(\*) as p from h1b\_final where year = 2013 group by year,job\_title;

create table 1bavggrowth2014(year4 string,job string,p4 int)

row format delimited

fields terminated by '\t'

stored as textfile;

insert overwrite table 1bavggrowth2014 select year,LOWER(job\_title),count(\*) as p from h1b\_final where year = 2014 group by year,job\_title;

create table 1bavggrowth2015(year5 string,job string,p5 int)

row format delimited

fields terminated by '\t'

stored as textfile;

insert overwrite table 1bavggrowth2015 select year,LOWER(job\_title),count(\*) as p from h1b\_final where year = 2015 group by year,job\_title;

create table 1bavggrowth2016(year6 string,job string,p6 int)

row format delimited

fields terminated by '\t'

stored as textfile;

insert overwrite table 1bavggrowth2016 select year,LOWER(job\_title),count(\*) as p from h1b\_final where year = 2016 group by year,job\_title;

annual growth for each year ((current\_year-previous\_year)\*100/previous\_year)

--------------------------------------------------------------

create table a2out(year string,job string,totalavggrowth double)

row format delimited

fields terminated by '\t'

stored as textfile;

insert overwrite table a2out

select \* from(select a.year1,a.job,(sum(b.p2-a.p1)\*100/sum(a.p1)) as avg1 from 1bavggrowth2011 a,1bavggrowth2012 b where a.job=b.job group by a.year1,a.job order by avg1 desc ) q1

union

select \* from(select a.year2,a.job,(sum(b.p3-a.p2)\*100/sum(a.p2)) avg1 from 1bavggrowth2012 a,1bavggrowth2013 b where a.job=b.job group by a.year2,a.job order by avg1 desc )q2

union

select \* from(select a.year3,a.job,(sum(b.p4-a.p3)\*100/sum(a.p3)) avg1 from 1bavggrowth2013 a,1bavggrowth2014 b where a.job=b.job group by a.year3,a.job order by avg1 desc )q3

union

select \* from(select a.year4,a.job,(sum(b.p5-a.p4)\*100/sum(a.p4)) avg1 from 1bavggrowth2014 a,1bavggrowth2015 b where a.job=b.job group by a.year4,a.job order by avg1 desc )q4

union

select \* from(select a.year5,a.job,(sum(b.p6-a.p5)\*100/sum(a.p5)) avg1 from 1bavggrowth2015 a,1bavggrowth2016 b where a.job=b.job group by a.year5,a.job order by avg1 desc ) q5

adding all of them and finding the top 5 jobs

-----------------------------------------------------------------------

create table totalavg(job string,totalgrowthavgfor5years double)

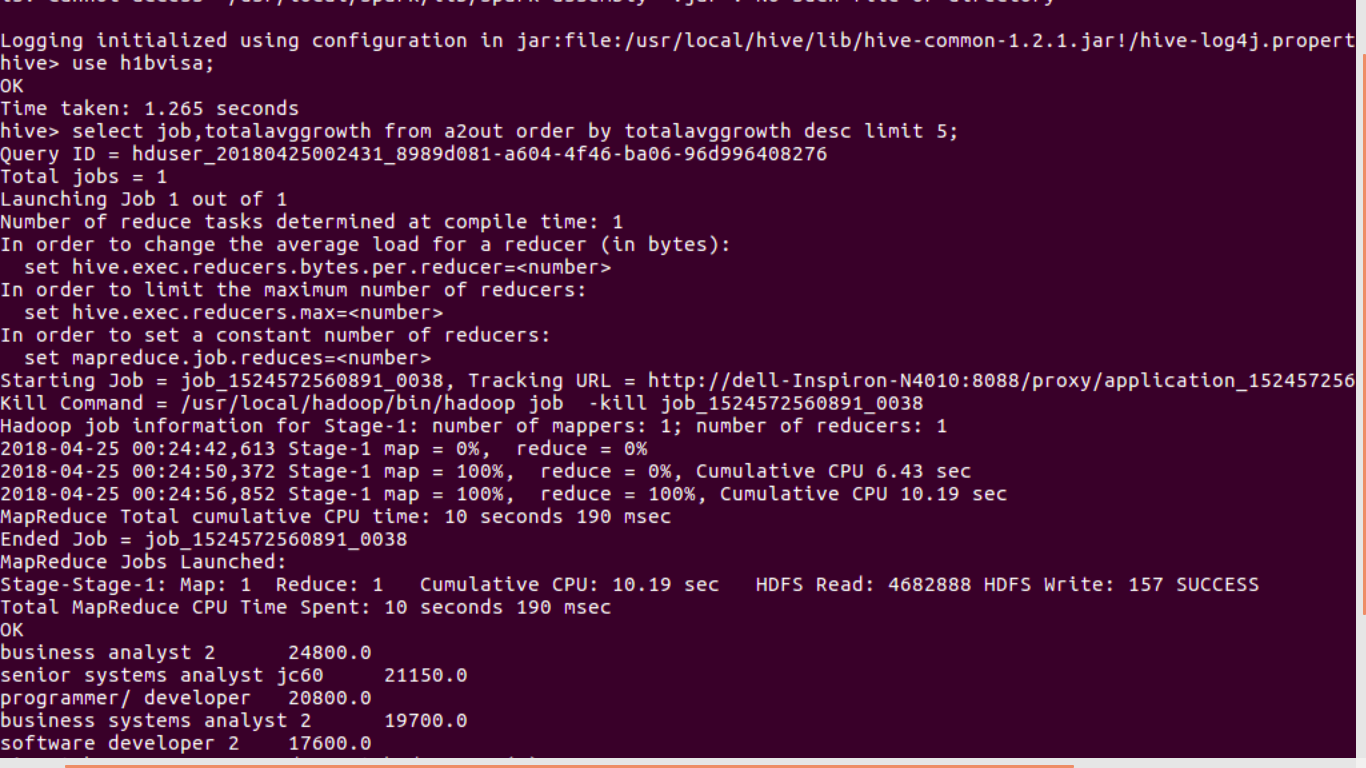
row format delimited

fields terminated by '\t'

stored as textfile;

insert overwrite table totalavg select job,sum(totalavggrowth)/5 as totalgrowthavgfor5years from a2out group by job;

select job,totalavggrowth from a2out order by totalavggrowth desc limit 5;

OUTPUT:

***Top 5 job titles who are having highest avg growth in applications***

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

**Tool used:PIG**

h1bvisa = load '/home/hduser/Documents/H1bvisa/H1b\_final' using PigStorage('\t') AS (id:long,case\_status:chararray,emp\_name:chararray,soc\_name:chararray,job\_title:chararray,full\_time\_pos:chararray,prevailing\_wage:long,year:chararray,worksite:chararray,lon:double,lat:double);

h1bvisa = foreach h1bvisa generate $4,$7,$8;

data\_eng\_2011 = filter h1bvisa by $0 matches '(.\*)DATA ENGINEER(.\*)' and $1 matches '2011';

group\_worksite\_2011 = group data\_eng\_2011 by $2;

count\_job\_2011 = foreach group\_worksite\_2011 generate '2011',group,COUNT($1);

order\_job\_2011 = order count\_job\_2011 by $2 desc;

limit\_job\_2011 = limit order\_job\_2011 1;

data\_eng\_2012 = filter h1bvisa by $0 matches '(.\*)DATA ENGINEER(.\*)' and $1 matches '2012';

group\_worksite\_2012 = group data\_eng\_2012 by $2;

count\_job\_2012 = foreach group\_worksite\_2012 generate '2012',group,COUNT($1);

order\_job\_2012 = order count\_job\_2012 by $2 desc;

limit\_job\_2012 = limit order\_job\_2012 1;

data\_eng\_2013 = filter h1bvisa by $0 matches '(.\*)DATA ENGINEER(.\*)' and $1 matches '2013';

group\_worksite\_2013 = group data\_eng\_2013 by $2;

count\_job\_2013 = foreach group\_worksite\_2013 generate '2013',group,COUNT($1);

order\_job\_2013 = order count\_job\_2013 by $2 desc;

limit\_job\_2013 = limit order\_job\_2013 1;

data\_eng\_2014 = filter h1bvisa by $0 matches '(.\*)DATA ENGINEER(.\*)' and $1 matches '2014';

group\_worksite\_2014 = group data\_eng\_2014 by $2;

count\_job\_2014 = foreach group\_worksite\_2014 generate '2014',group,COUNT($1);

order\_job\_2014 = order count\_job\_2014 by $2 desc;

limit\_job\_2014 = limit order\_job\_2014 1;

data\_eng\_2015 = filter h1bvisa by $0 matches '(.\*)DATA ENGINEER(.\*)' and $1 matches '2015';

group\_worksite\_2015 = group data\_eng\_2015 by $2;

count\_job\_2015 = foreach group\_worksite\_2015 generate '2015',group,COUNT($1);

order\_job\_2015 = order count\_job\_2015 by $2 desc;

limit\_job\_2015 = limit order\_job\_2015 1;

--dump limit\_job\_2015;

data\_eng\_2016 = filter h1bvisa by $0 matches '(.\*)DATA ENGINEER(.\*)' and $1 matches '2016';

group\_worksite\_2016 = group data\_eng\_2016 by $2;

count\_job\_2016 = foreach group\_worksite\_2016 generate '2016',group,COUNT($1);

order\_job\_2016 = order count\_job\_2016 by $2 desc;

limit\_job\_2016 = limit order\_job\_2016 1;

union\_all = union limit\_job\_2011,limit\_job\_2012,limit\_job\_2013,limit\_job\_2014,limit\_job\_2015,limit\_job\_2016;

--dump union\_all;

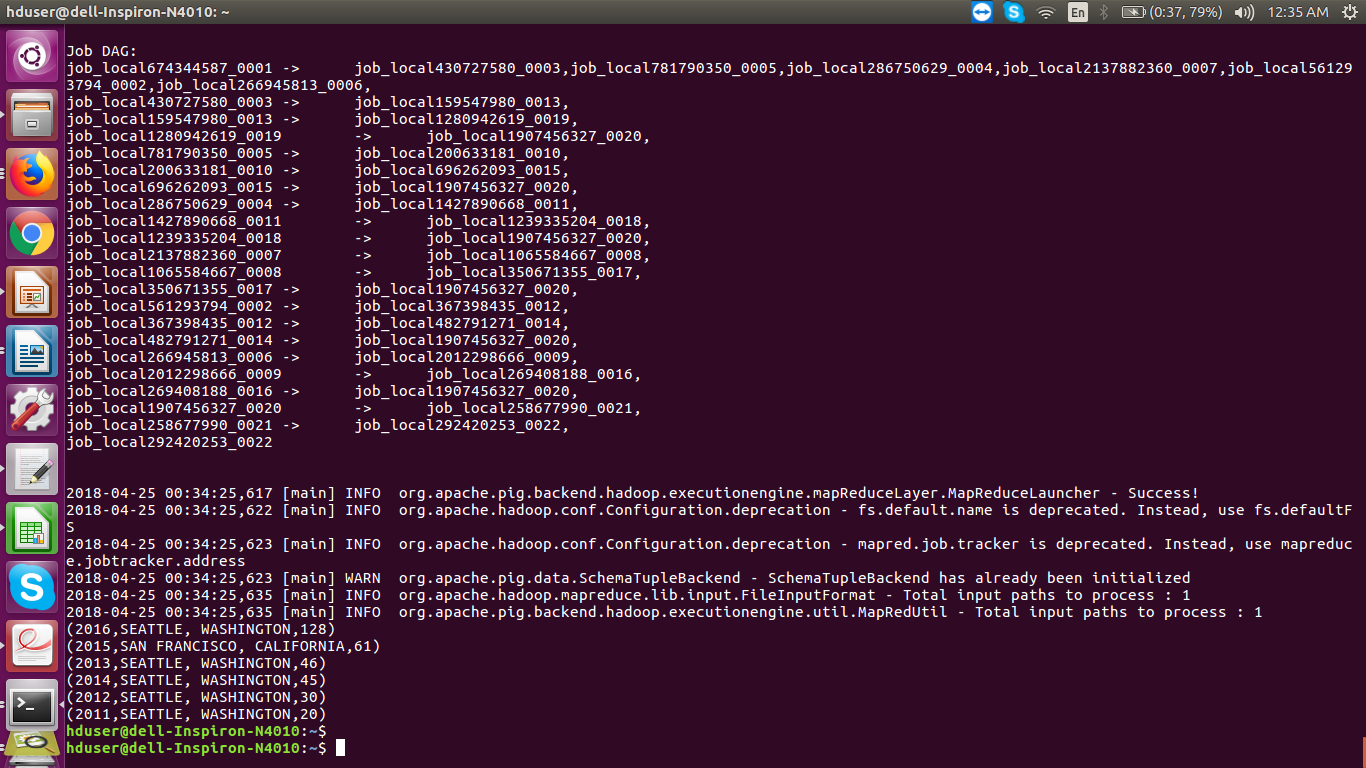
order\_union = order union\_all by $2 desc;

--dump order\_union;

store order\_union into '/home/hduser/Documents/H1bvisaOP/pig/2a.pig';

OUTPUT:

***The part of the US has the most Data Engineer jobs for each year***



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

**Tool used:HIVE**

create table 2ByYear(s\_no int,case\_status string, employer\_name string, soc\_name string, job\_title string, full\_time\_position string,prevailing\_wage bigint, worksite string, longitute double, latitute double )

partitioned by (year string)

row format delimited

fields terminated by '\t'

STORED AS TEXTFILE;

from h1b\_final h INSERT OVERWRITE TABLE 2ByYear PARTITION(year) select h.s\_no,h.case\_status,h.employer\_name,h.soc\_name, h.job\_title, h.full\_time\_position,h.prevailing\_wage,h.worksite, h.longitute, h.latitute,h.age DISTRIBUTE by year;

create table 2bout(worksite string,year string,p int)

row format delimited

fields terminated by '\t'

STORED AS TEXTFILE;

insert overwrite table 2bout

select \* from(select worksite,year,count(\*) as p from 2ByYear where case\_status = 'CERTIFIED' and year = 2011 group by year,worksite order by p desc limit 5)q1

union

select \* from(select worksite,year,count(\*) as p from 2ByYear where case\_status = 'CERTIFIED' and year = 2012 group by year,worksite order by p limit 5 )q2

union

select \* from(select worksite,year,count(\*) as p from 2ByYear where case\_status = 'CERTIFIED' and year = 2013 group by year,worksite order by p desc limit 5)q3

union

select \* from(select worksite,year,count(\*) as p from 2ByYear where case\_status = 'CERTIFIED' and year = 2014 group by year,worksite order by p desc limit 5)q4

union

select \* from(select worksite,year,count(\*) as p from 2ByYear where case\_status = 'CERTIFIED' and year = 2015 group by year,worksite order by p desc limit 5 )q5

union

select \* from(select worksite,year,count(\*) as p from 2ByYear where case\_status = 'CERTIFIED' and year = 2016 group by year,worksite order by p desc limit 5)q6

order by worksite,year desc,p desc;

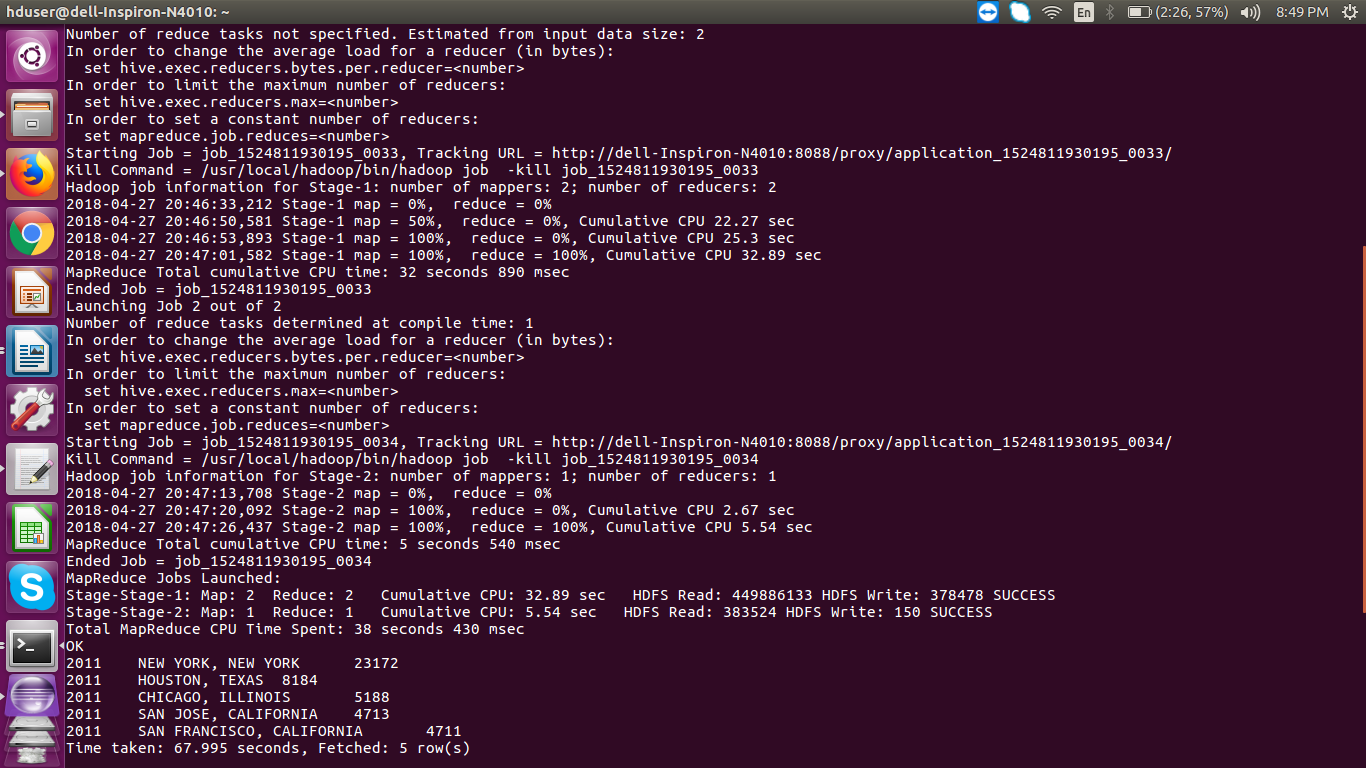
sorted for all years at once:

---------------------------------

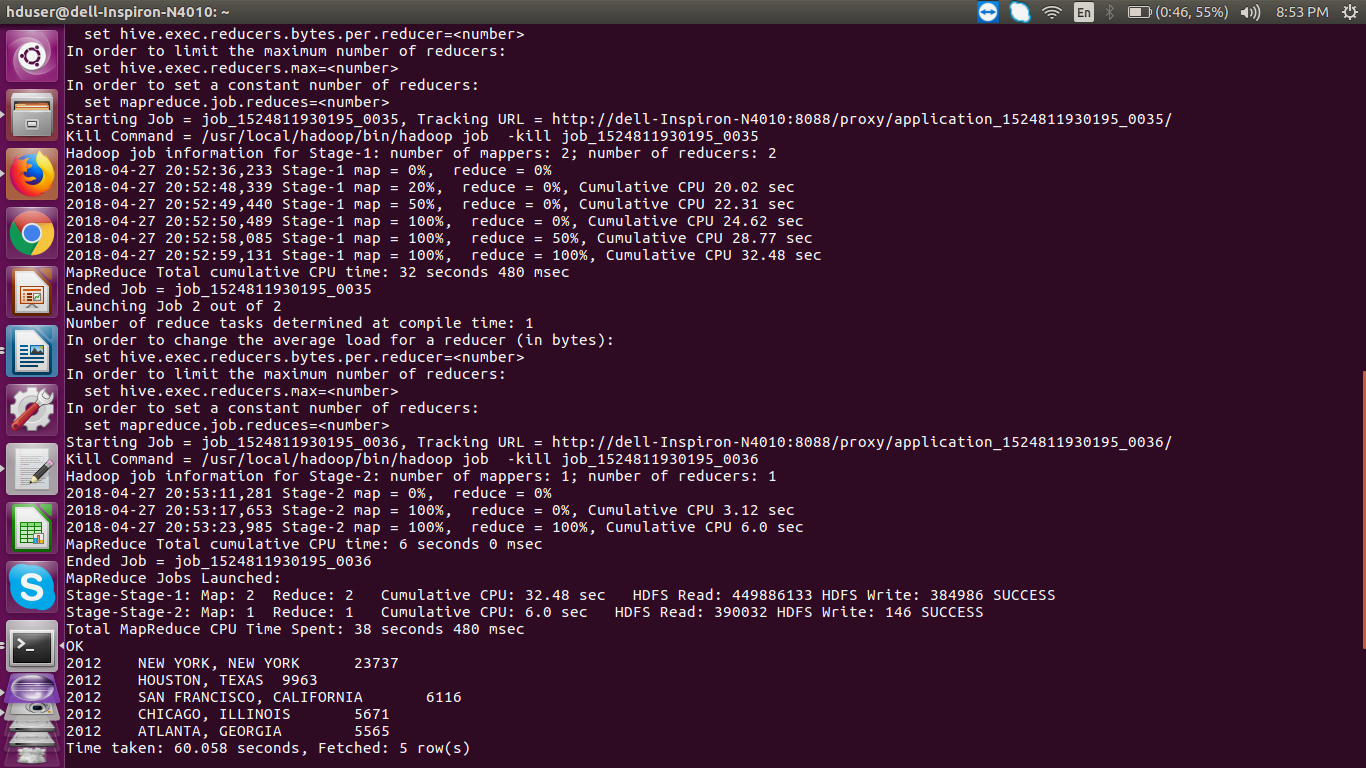
select worksite,year,p from 2bout group by year,worksite order by p desc;

OUTPUT:

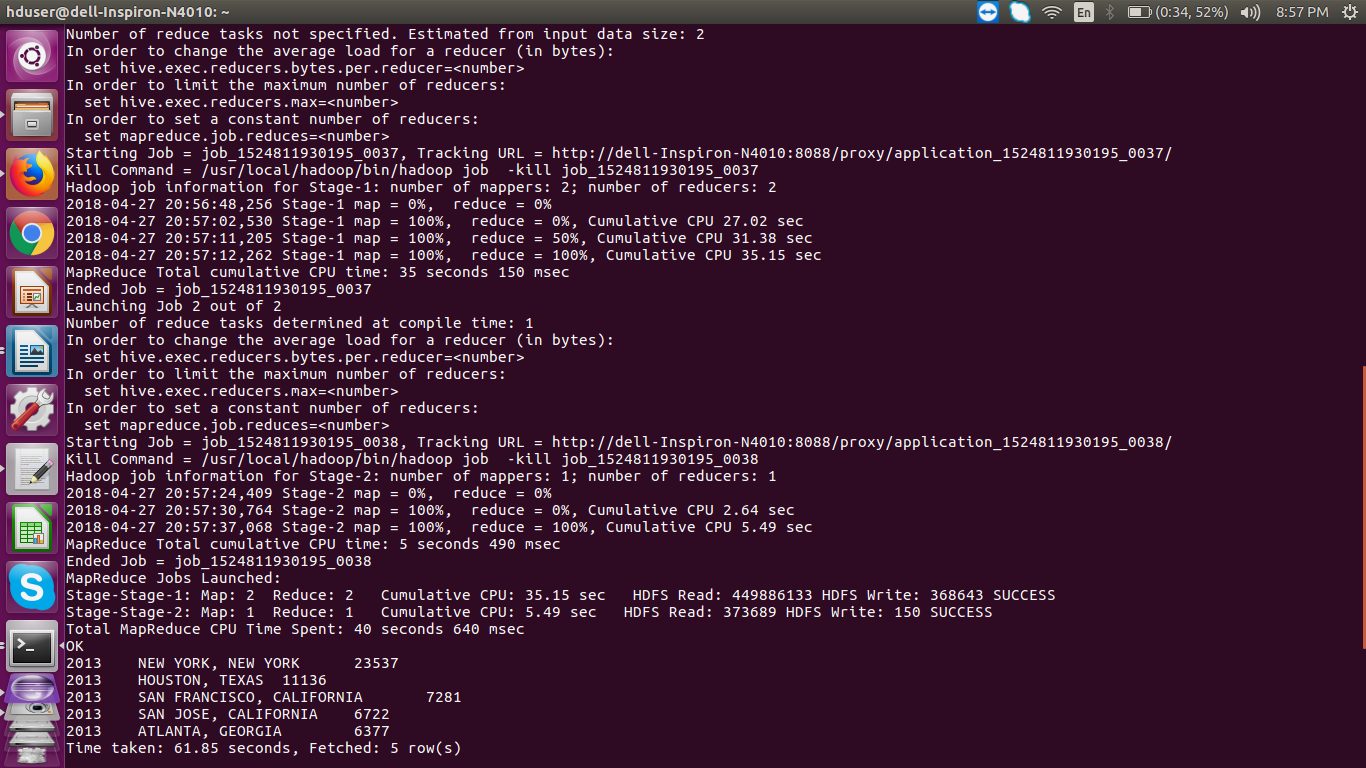
***For the year 2011:***



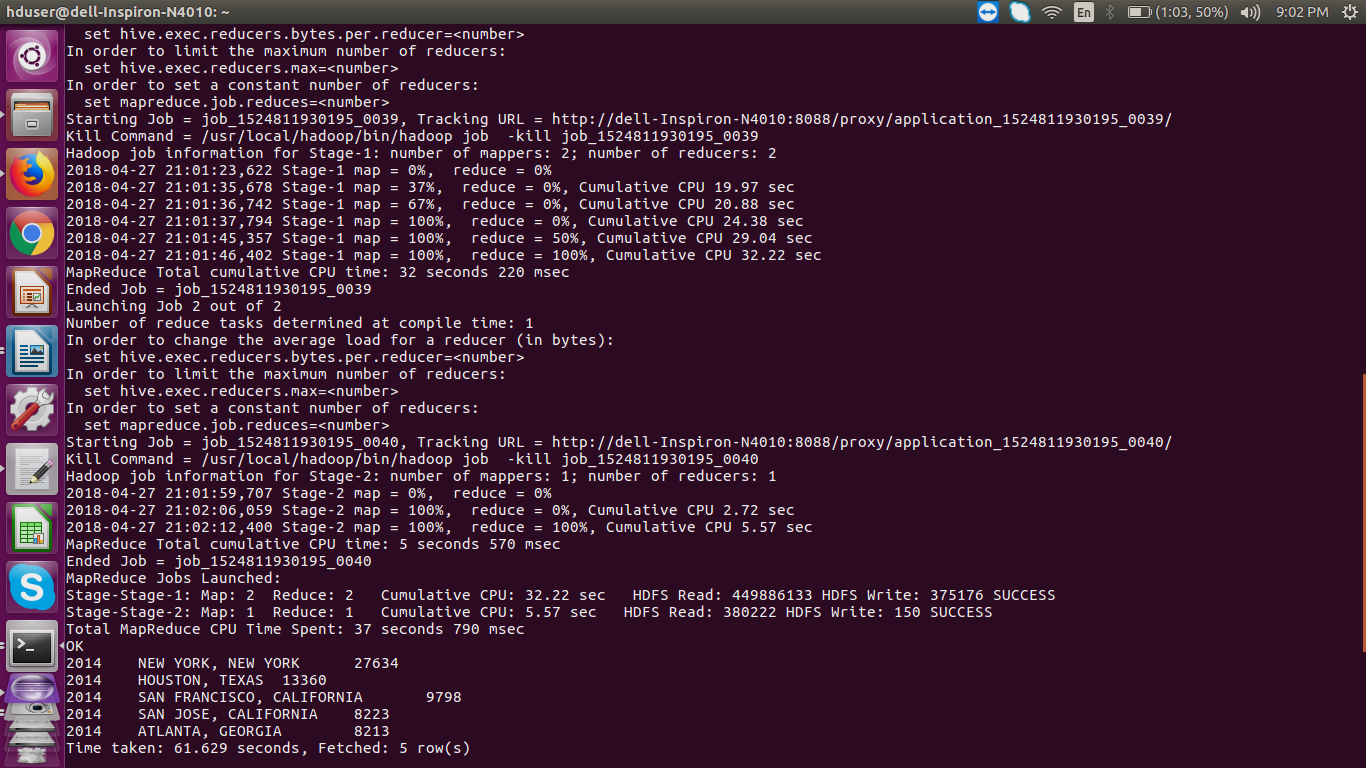
***For the year 2012:***



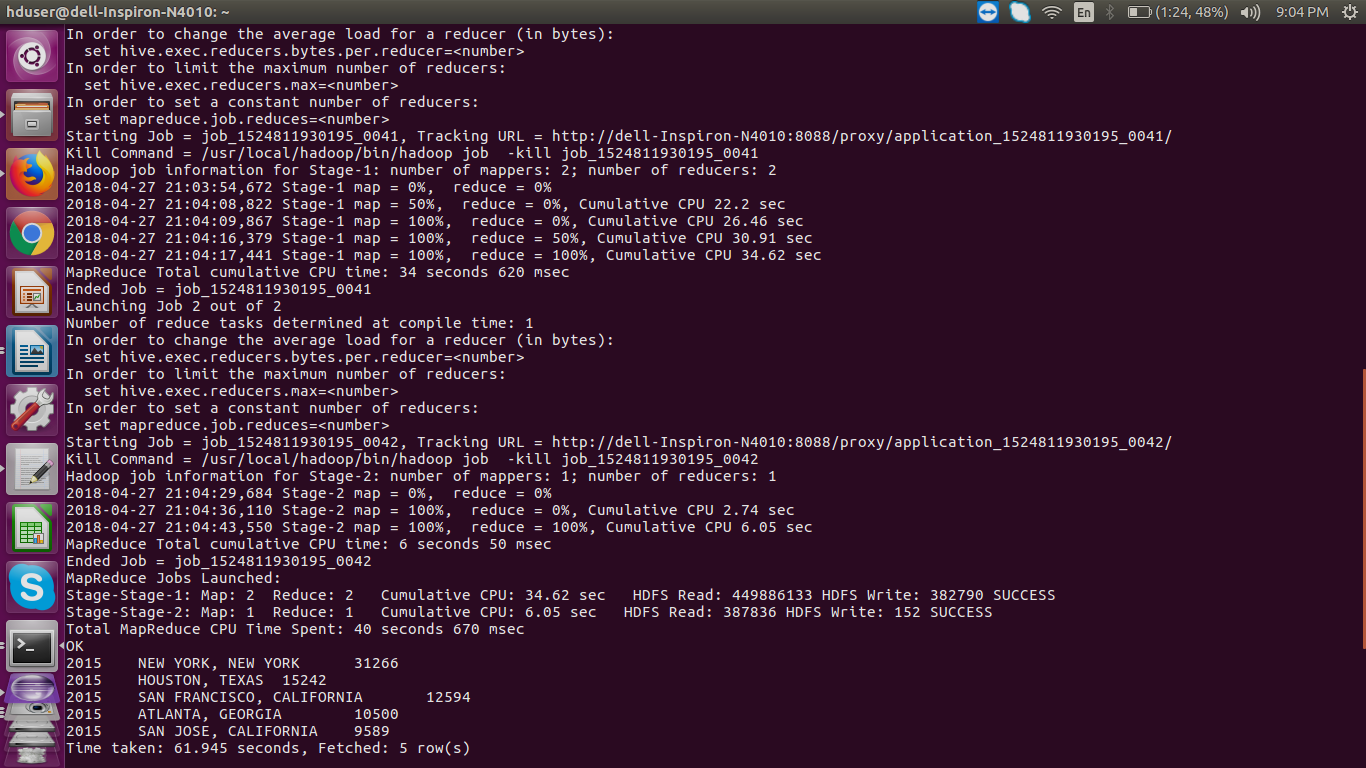
***For the year 2013:***



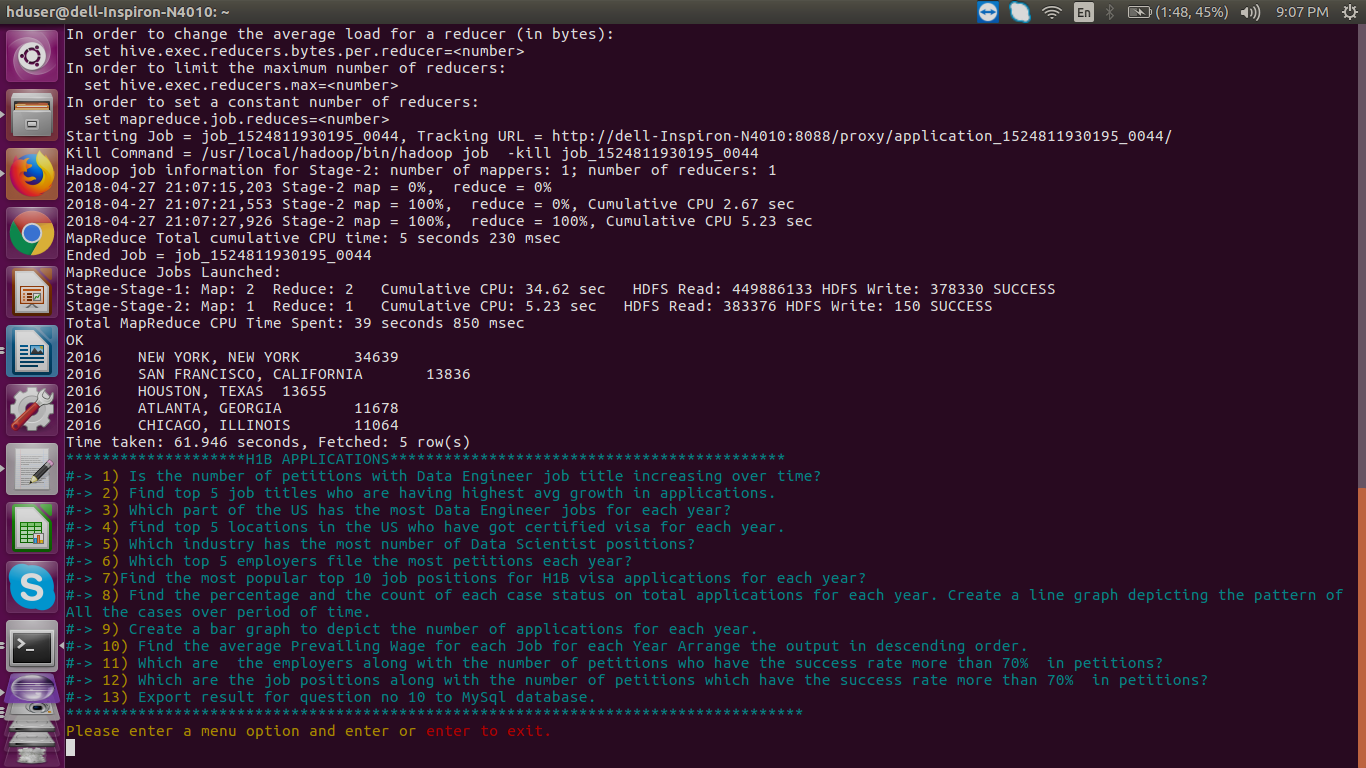
***For the year 2014:***



***For the year 2015:***



***For the year 2016:***



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

**[certified]**

**Tool used:MAPREDUCE**

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.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 qns3{

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

{

private Text key1 = new Text();

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

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

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

String industry = record[3];

String job\_title = record[4];

String case\_status = record[1];

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

{

key1.set(industry);

context.write(key1,one);

}

}

}

public static class WorksiteReducer extends Reducer<Text, IntWritable, NullWritable, Text>

{

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

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

{

int sum=0;

for(IntWritable v :values)

{

sum+=v.get();

}

String key1 = key.toString();

desccountrofDataScientists.put(new IntWritable(sum),new Text(key1));

if (desccountrofDataScientists.size() > 1 )

{

desccountrofDataScientists.remove(desccountrofDataScientists.firstKey());

}

}

protected void cleanup(Context context) throws IOException, InterruptedException {

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

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

}

}

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

Configuration conf = new Configuration();

Job job = Job.getInstance(conf,"Data Engineer Job Increasing");

job.setJarByClass(q2a.class);

job.setMapperClass(MyMapper.class);

//job.setPartitionerClass(YearPartitioner.class);

job.setReducerClass(WorksiteReducer.class);

//job.setNumReduceTasks(6);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);

job.setOutputKeyClass(NullWritable.class);

job.setOutputValueClass(Text.class);

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

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

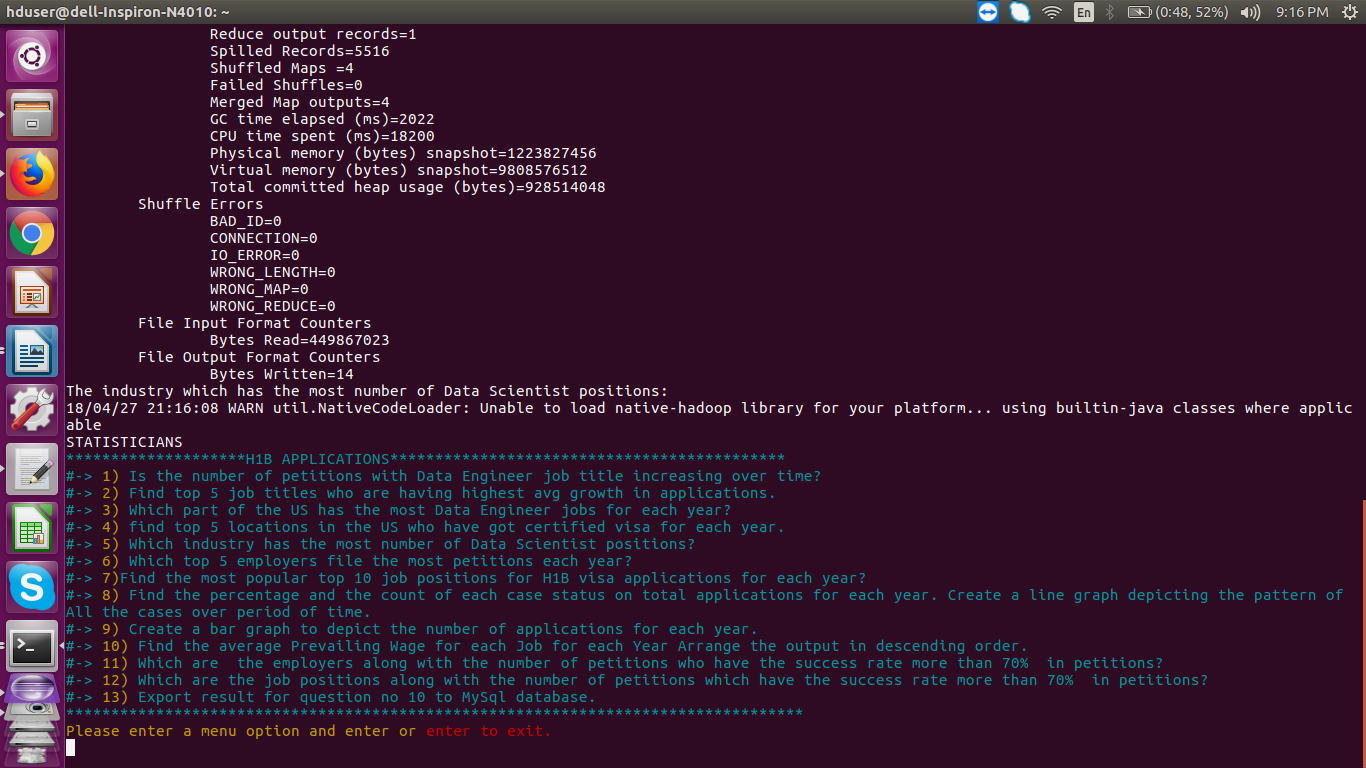
System.exit(job.waitForCompletion(true) ? 0 :1);

}

}

OUTPUT:

***Industry that has the most number of Data Scientist positions***



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

**Tool used:PIG**

data = load '/home/hduser/Documents/H1bvisa/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: int,year: chararray, worksite: chararray, longitude: double, latitute: double);

data = foreach data generate $1,$2,$7;

data\_2011 = filter data by ($2 matches '2011');

data\_2012 = filter data by ($2 matches '2012');

data\_2013 = filter data by ($2 matches '2013');

data\_2014 = filter data by ($2 matches '2014');

data\_2015 = filter data by ($2 matches '2015');

data\_2016 = filter data by ($2 matches '2016');

groupdata2011onPetitionandEmp = group data\_2011 by ($1,$2);

groupdata2012onPetitionandEmp = group data\_2012 by ($1,$2);

groupdata2013onPetitionandEmp = group data\_2013 by ($1,$2);

groupdata2014onPetitionandEmp = group data\_2014 by ($1,$2);

groupdata2015onPetitionandEmp = group data\_2015 by ($1,$2);

groupdata2016onPetitionandEmp = group data\_2016 by ($1,$2);

count2011 = foreach groupdata2011onPetitionandEmp generate FLATTEN(group),COUNT(data\_2011.$0);

count2012 = foreach groupdata2012onPetitionandEmp generate FLATTEN(group),COUNT(data\_2012.$0);

count2013 = foreach groupdata2013onPetitionandEmp generate FLATTEN(group),COUNT(data\_2013.$0);

count2014 = foreach groupdata2014onPetitionandEmp generate FLATTEN(group),COUNT(data\_2014.$0);

count2015 = foreach groupdata2015onPetitionandEmp generate FLATTEN(group),COUNT(data\_2015.$0);

count2016 = foreach groupdata2016onPetitionandEmp generate FLATTEN(group),COUNT(data\_2016.$0);

top5Employers2011 = limit(order count2011 by $2 desc) 5;

top5Employers2012 = limit(order count2012 by $2 desc) 5;

top5Employers2013 = limit(order count2013 by $2 desc) 5;

top5Employers2014 = limit(order count2014 by $2 desc) 5;

top5Employers2015 = limit(order count2015 by $2 desc) 5;

top5Employers2016 = limit(order count2016 by $2 desc) 5;

store top5Employers2011 into '/home/hduser/Documents/H1bvisaOP/pig/4a.pig/top5Employers2011';

store top5Employers2012 into '/home/hduser/Documents/H1bvisaOP/pig/4a.pig/top5Employers2012';

store top5Employers2013 into '/home/hduser/Documents/H1bvisaOP/pig/4a.pig/top5Employers2013';

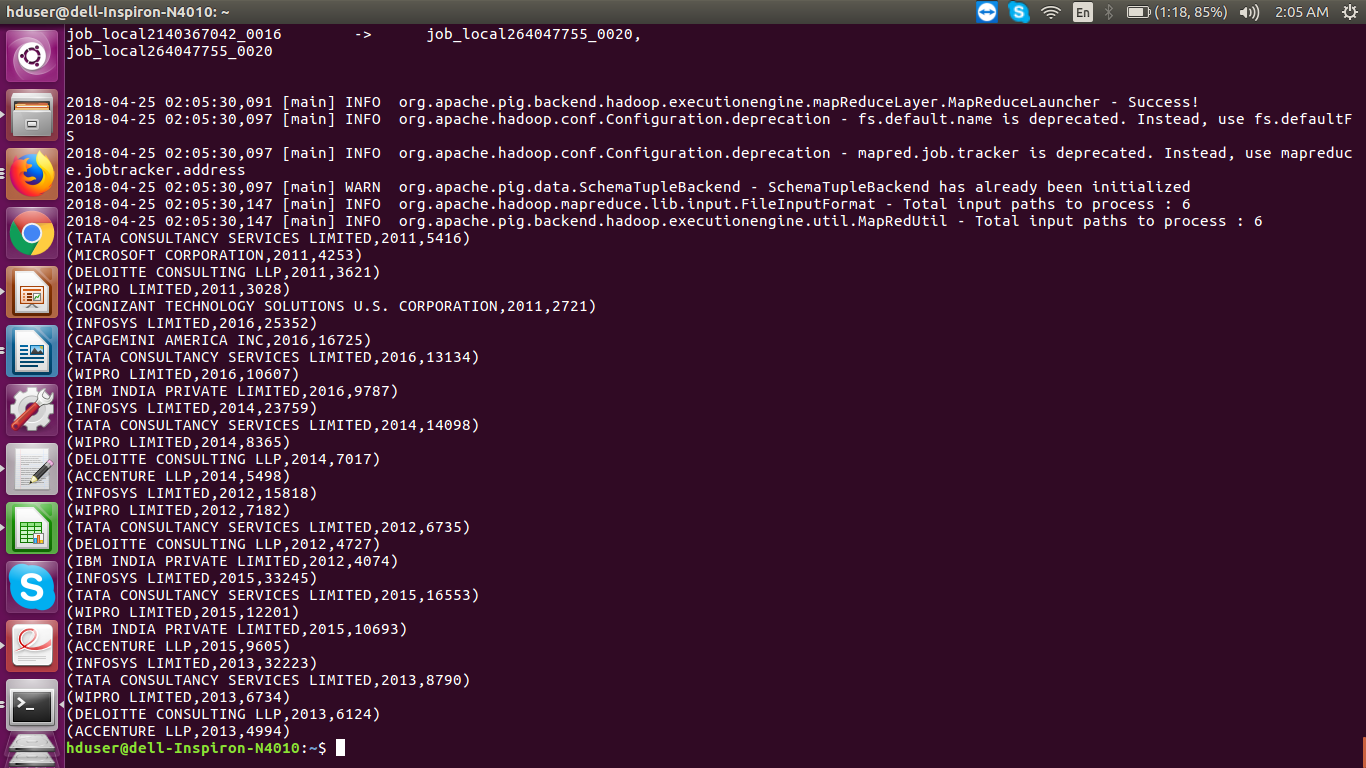
store top5Employers2014 into '/home/hduser/Documents/H1bvisaOP/pig/4a.pig/top5Employers2014';

store top5Employers2015 into '/home/hduser/Documents/H1bvisaOP/pig/4a.pig/top5Employers2015';

store top5Employers2016 into'/home/hduser/Documents/H1bvisaOP/pig/4a.pig/top5Employers2016';

OUTPUT:

***Top 5 employers file the most petitions each year***



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

**a) for all the applications.**

**b) for only certified applications.**

**Tool used:MAPREDUCE**

**5a:**

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.output.FileOutputFormat;

public class qns5a {

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

{

@Override

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

{

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

String year = record[7];

String job\_title = record[4];

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

}

}

public static class YearPartitioner extends Partitioner<Text, Text>

{

@Override

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

String year = value.toString();

if(year.equals("2011"))

{

return 0 % numReduceTasks;

}

else if(year.equals("2012"))

{

return 1 % numReduceTasks;

}

else if(year.equals("2013"))

{

return 2 % numReduceTasks;

}

else if(year.equals("2014"))

{

return 3 % numReduceTasks;

}

else if(year.equals("2015"))

{

return 4 % numReduceTasks;

}

else

{

return 5 % numReduceTasks;

}

}

}

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

{

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

@Override

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

{

int count = 0;

String year = "";

for(Text val : values)

{

year = val.toString();

count++;

}

String Job\_title = key.toString();

String myValue = year+","+Job\_title +","+count;

map.put(new Integer(count), new Text(myValue));

if(map.size() > 10)

{

map.remove(map.firstKey());

}

}

@Override

protected void cleanup( Context context)throws IOException, InterruptedException

{

for(Text top10 : map.descendingMap().values())

{

context.write(NullWritable.get(), top10);

}

}

}

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

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "Top 10 Job Positios for each Year");

job.setJarByClass(qns5a.class);

job.setMapperClass(JobMapper.class);

job.setPartitionerClass(YearPartitioner.class);

job.setNumReduceTasks(6);

job.setReducerClass(JobReducer.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]));

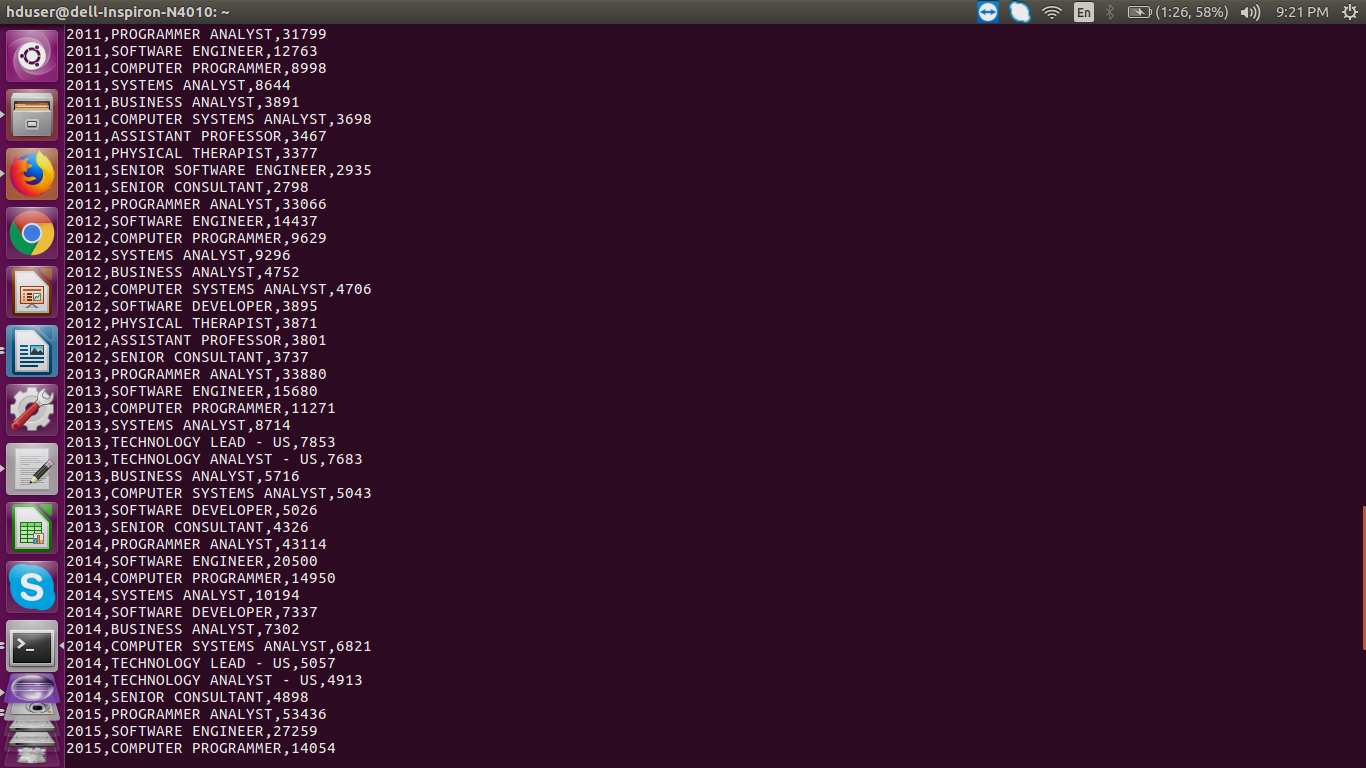
System.exit(job.waitForCompletion(true) ? 0 : 1);

}

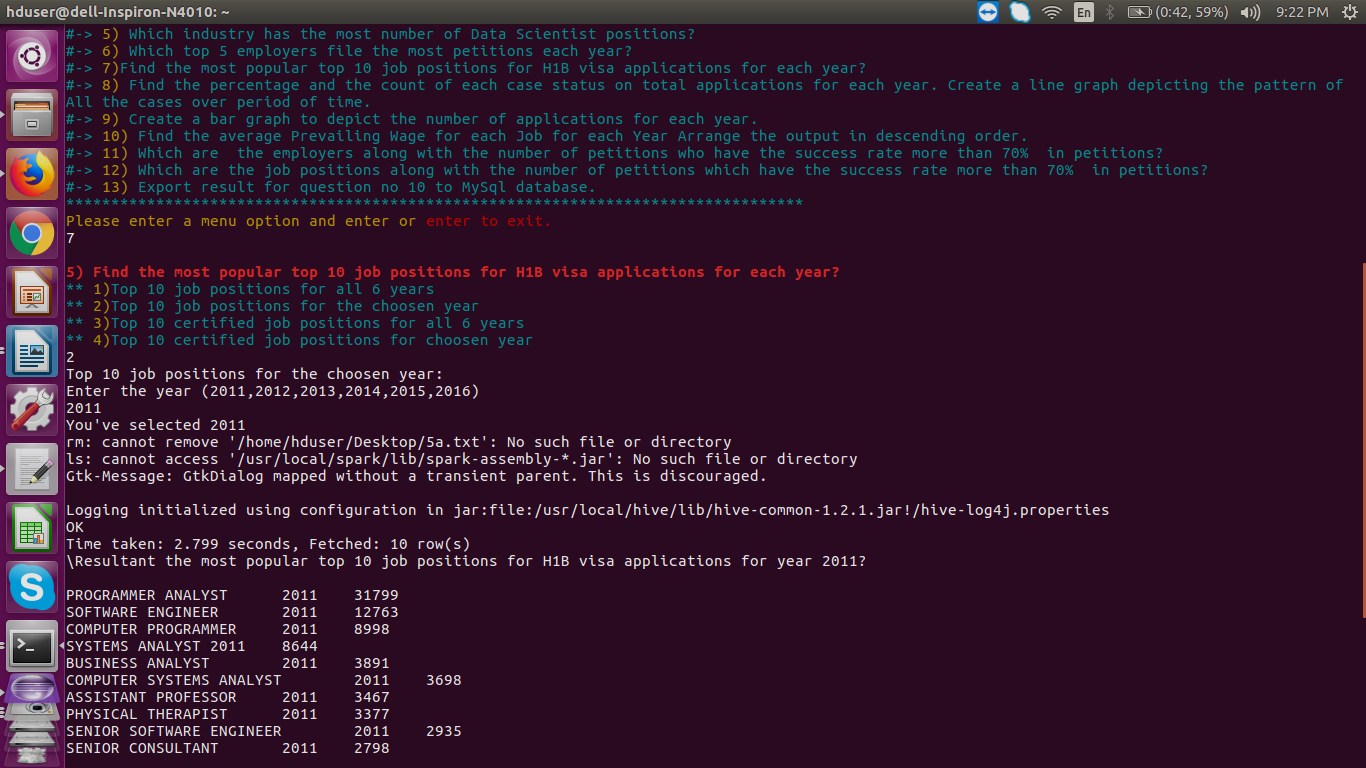
}

OUTPUT:

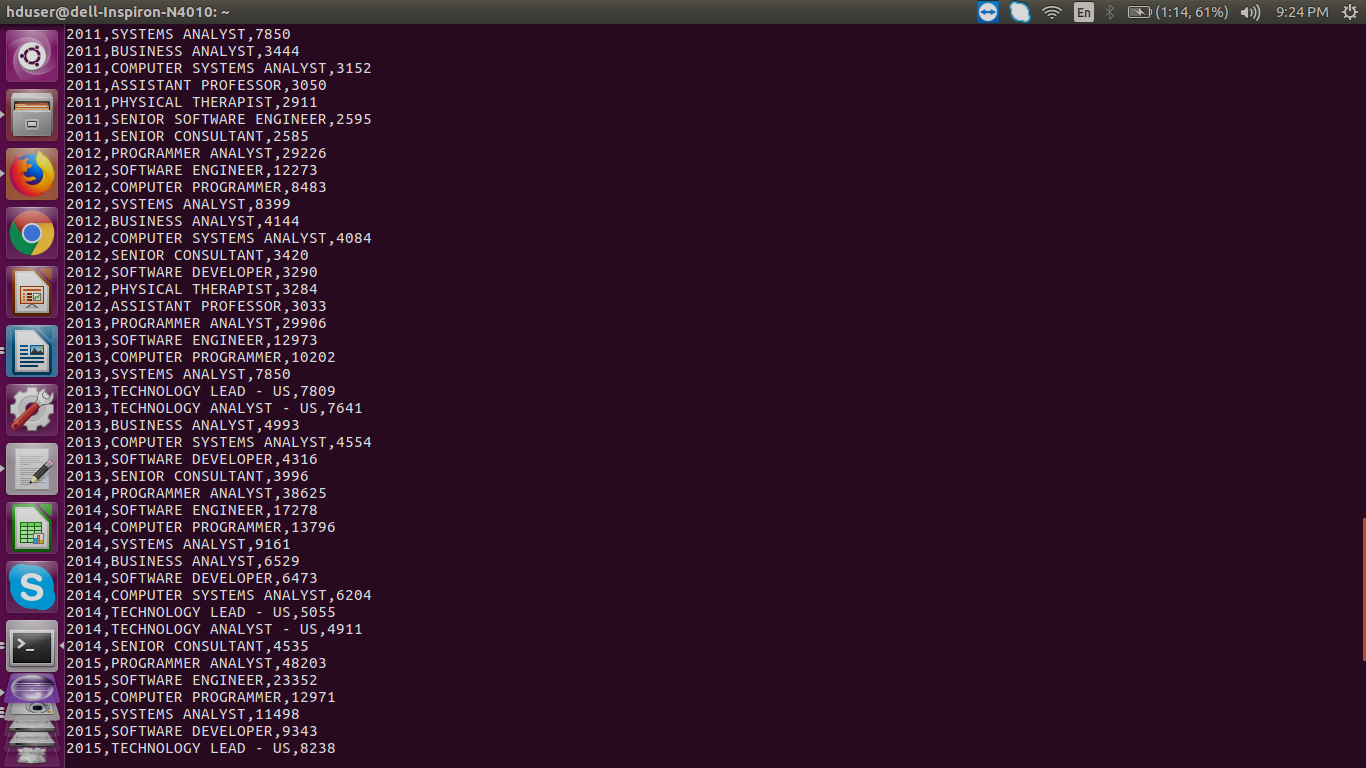
***For all 6 years:***

**

***For a specific year like 2011:***



***For a certified job positions for all years:***



***For a specific year 2011 certified job positions:***



**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.**

**Tool used:PIG**

table1 = load '/home/hduser/Documents/H1bvisa/H1b\_final' using PigStorage('\t') as (s\_no,case\_status,employer\_name,soc\_name,job\_title,full\_time\_position ,prevailing\_wage,year,worksite,longitute,latitute);

noheader = filter table1 by $0 > '0' ;

table2 = order noheader by $0;

table3 = group table2 by (year);

table4 = FOREACH table3 GENERATE FLATTEN(group) AS year,COUNT(table2.case\_status) as total\_case\_status;

table5 = group table2 by (year,case\_status);

--dump table5;

table6 = FOREACH table5 GENERATE FLATTEN(group) AS (year,case\_status),COUNT(table2.case\_status) as total\_case\_status;

join\_table = join table6 by year, table4 by year;

table7 = foreach join\_table generate $0,$1,$2,$4;

table8 = foreach table7 generate $0,$1,$2,$3,CONCAT((chararray)ROUND\_TO((float)(($2\*100)/$3),2),'%');

--describe table8;

filtcer = filter table8 by ($1 matches 'CERTIFIED');

filtden = filter table8 by ($1 matches 'DENIED');

filtcerwith = filter table8 by ($1 matches 'CERTIFIED-WITHDRAWN');

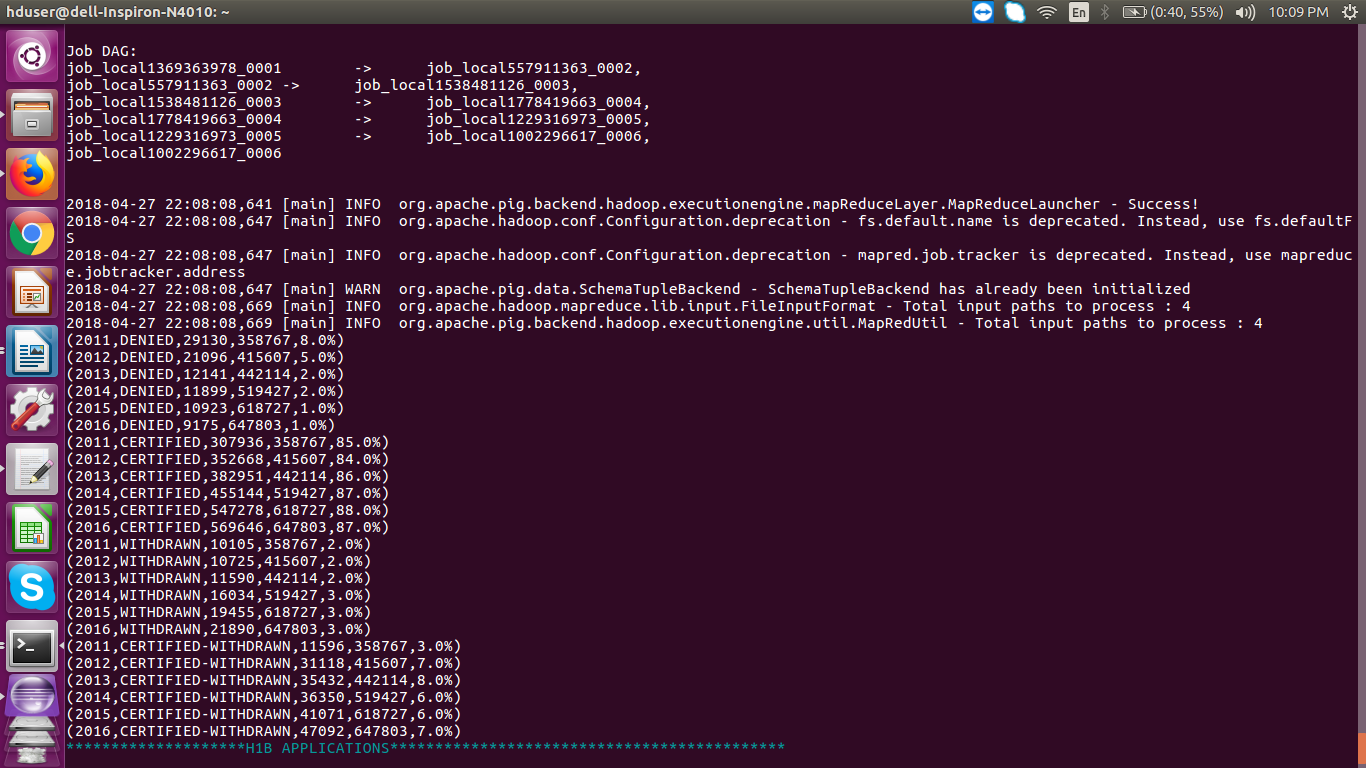
filtwith = filter table8 by ($1 matches 'WITHDRAWN');

--dump filtyr2011;

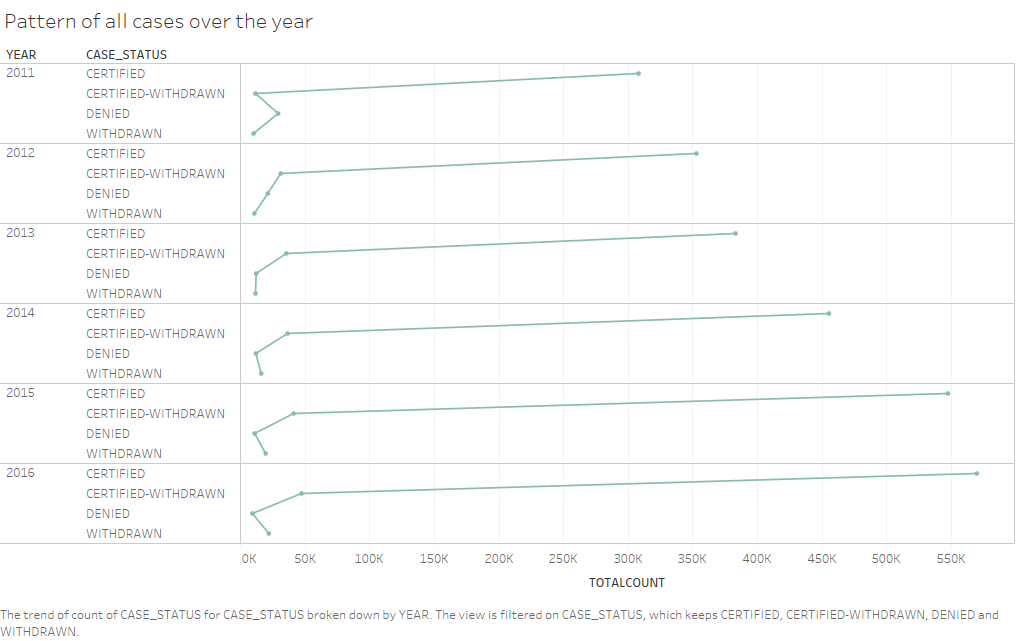
union\_all = union filtcer,filtden,filtcerwith,filtwith;

dump union\_all;

OUTPUT:

***The percentage and the count of each case status on total applications for each year.***

***The Line graph to depict the pattern of All the cases over the period of time.***



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

**Tool used:MAPREDUCE**

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.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;

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

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

public class qns7 {

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

{ private Text year = new Text();

private Text app = new Text();

@Override

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

{

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

String year1 = record[7];

String case\_status = record[1];

year.set(year1);

app.set(case\_status);

context.write(year,app);

}

}

public static class YearPartitioner extends Partitioner<Text, Text>

{

@Override

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

String year = key.toString();

if(year.equals("2011"))

{

return 0 % numReduceTasks;

}

else if(year.equals("2012"))

{

return 1 % numReduceTasks;

}

else if(year.equals("2013"))

{

return 2 % numReduceTasks;

}

else if(year.equals("2014"))

{

return 3 % numReduceTasks;

}

else if(year.equals("2015"))

{

return 4 % numReduceTasks;

}

else

{

return 5 % numReduceTasks;

}

}

}

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

{

@Override

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

{ int count=0;

String app="";

for(Text val : values)

{

app = val.toString();

count++;

}

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

}

}

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

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "Top 10 Job Positios for each Year");

job.setJarByClass(qns5a.class);

job.setMapperClass(AppMapper.class);

job.setPartitionerClass(YearPartitioner.class);

job.setNumReduceTasks(6);

job.setReducerClass(AppReducer.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

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

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

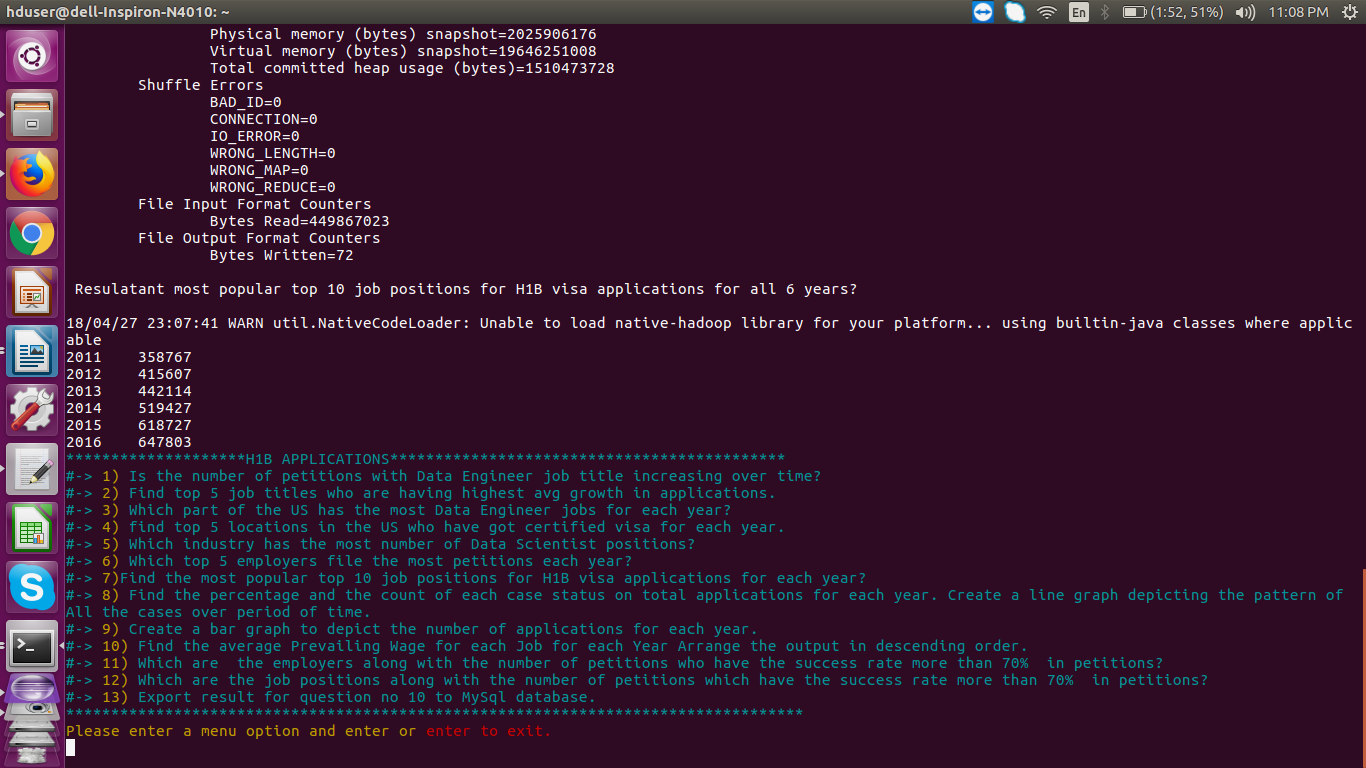
System.exit(job.waitForCompletion(true) ? 0 : 1);

}

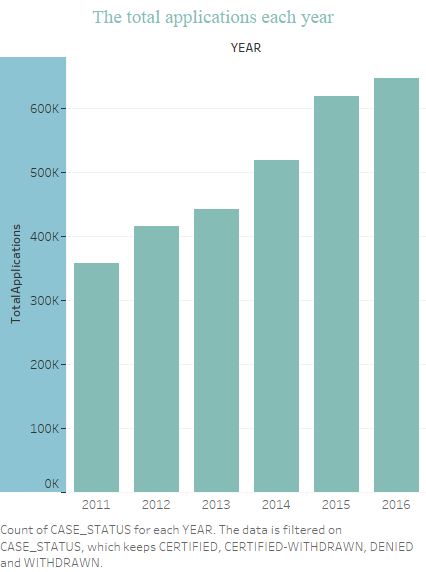
}

OUTPUT:

***The number of applications for each year***



***The bar graph depicting the number of applications for each year***



**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.**

**Tool used:HIVE**

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

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

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

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

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

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

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

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

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

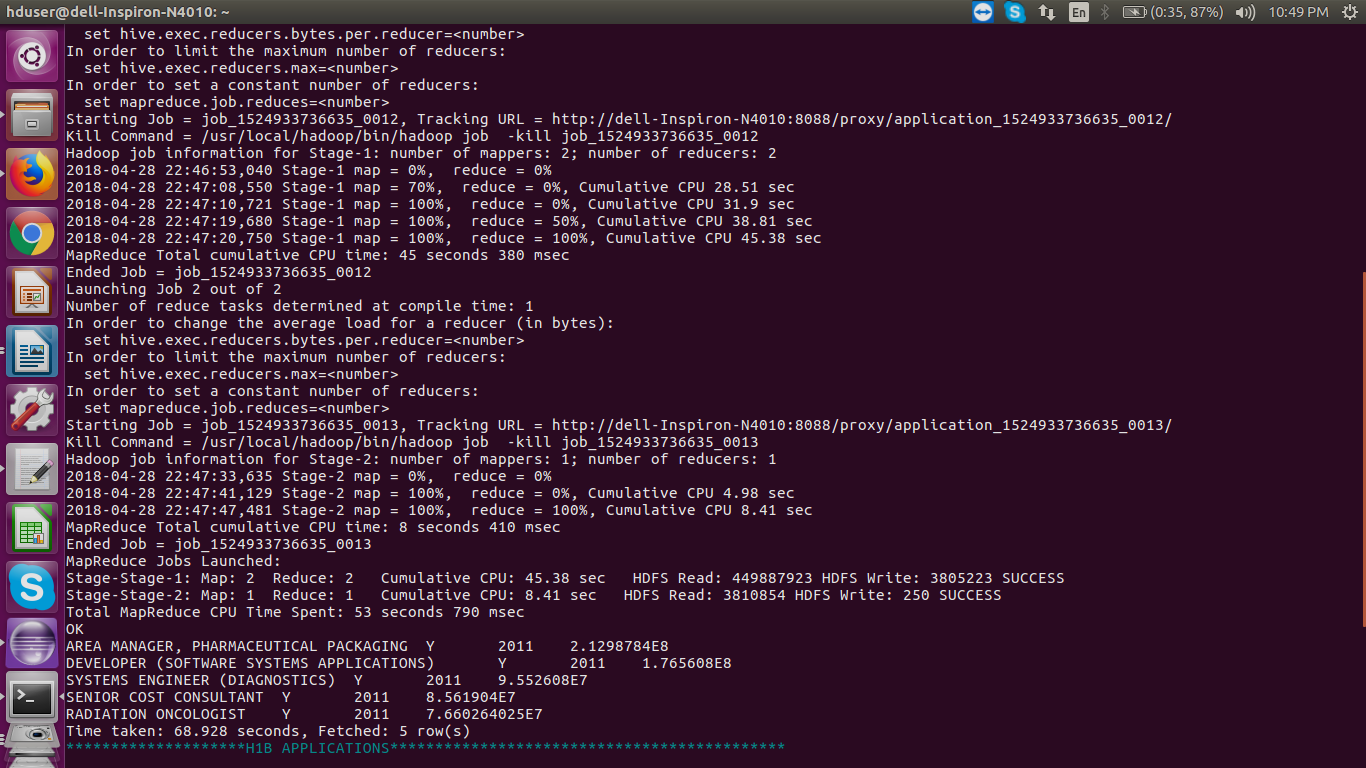
select job\_title,full\_time\_position,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='N' andyear='2015' group by job\_title,full\_time\_position,year order by average desc;

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

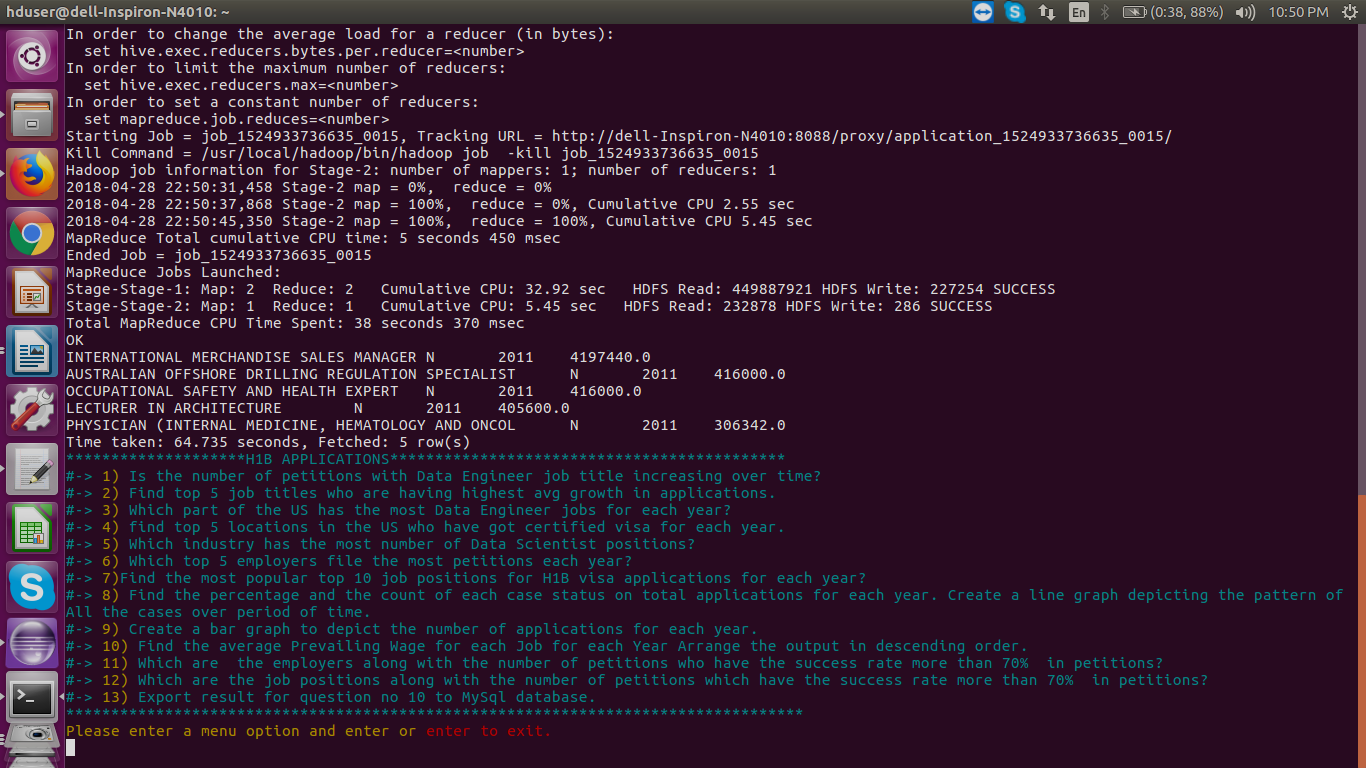
select job\_title,full\_time\_position,year,avg(prevailing\_wage) as average from h1b\_final where full\_time\_position ='N' and year='2016' group by job\_title,full\_time\_position,year order by average desc;

OUTPUT:

***The average Prevailing Wage for WITH FULL TIME POSITION job for the year 2011***



***The average Prevailing Wage for WITHOUT FULL TIME POSITION job for the year 2011***



**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)**

**Tool used:HIVE**

create table totalpetition(Employer\_name string,total int)

row format delimited

fields terminated by '\t'

STORED AS TEXTFILE;

insert overwrite table totalpetition select employer\_name,count(\*) as t from h1b\_final group by employer\_name having t>1000;

create table totalpetitionCertified(Employer\_name string,case\_status string,p int)

row format delimited

fields terminated by '\t'

STORED AS TEXTFILE;

insert overwrite table totalpetitionCertified select employer\_name ,case\_status,count(case\_status) as p from h1b\_final where case\_status = 'CERTIFIED' group by employer\_name,case\_status;

create table totalpetitionCertifiedWithdrawn(Employer\_name string,case\_status string,p int)

row format delimited

fields terminated by '\t'

STORED AS TEXTFILE;

insert overwrite table totalpetitionCertifiedWithdrawn select employer\_name ,case\_status,count(case\_status) as p from h1b\_final where case\_status = 'CERTIFIED-WITHDRAWN' group by employer\_name,case\_status;

create table total1(Employer\_name string,totalpetition int,petitionper double)

row format delimited

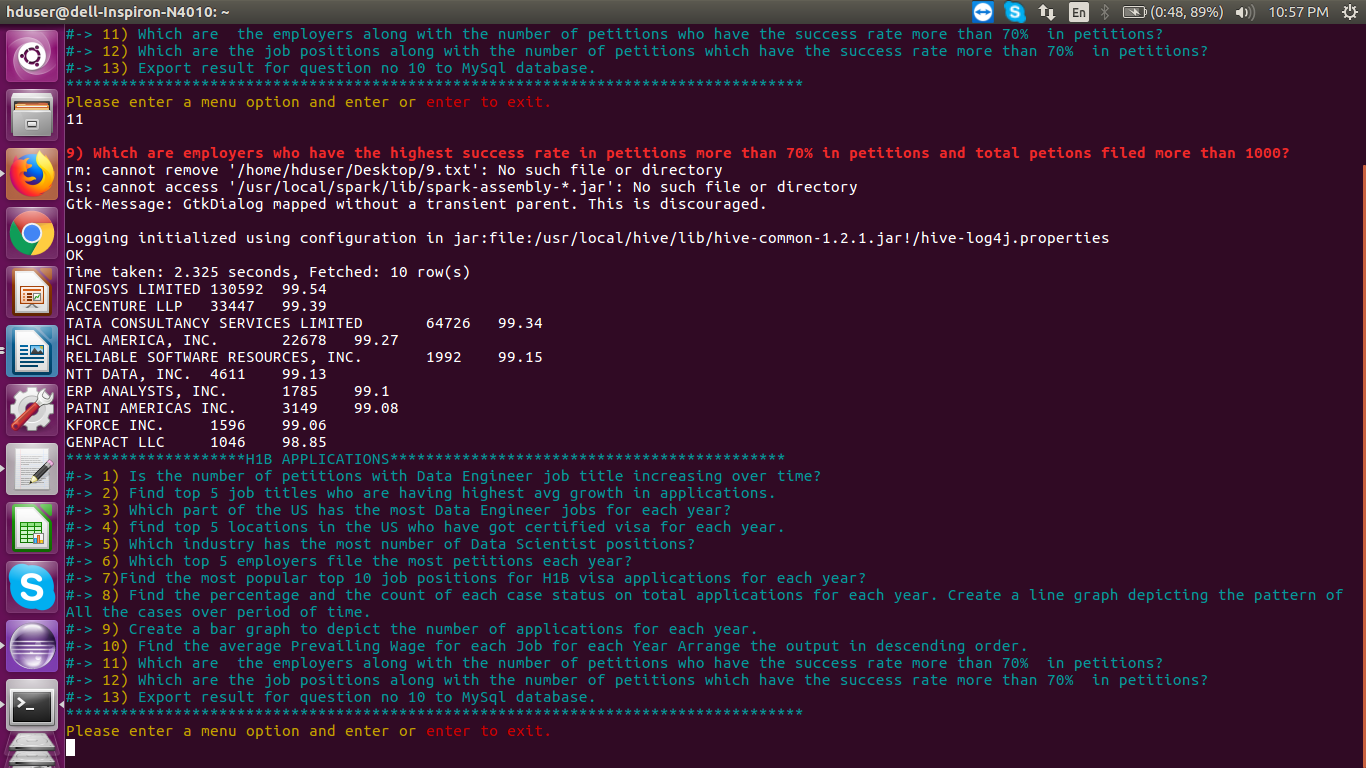
fields terminated by '\t'

STORED AS TEXTFILE;

insert overwrite table total1 select a.employer\_name,c.total,ROUND(((sum(a.p+b.p)\*100)/c.total),2) as petitionper from totalpetitionCertified a,totalpetitionCertifiedWithdrawn b,totalpetition c where a.employer\_name=b.employer\_name and a.employer\_name=b.employer\_name and b.employer\_name = c.employer\_name group by a.employer\_name,c.total order by petitionper desc;

OUTPUT:

***The employers along with the number of petitions who have the success rate more than 70% in petitions***



**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)?**

**Tool Used:PIG**

data = LOAD '/home/hduser/Documents/H1bvisa/H1b\_final/' USING PigStorage('\t') as

(s\_no:long,

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);

temp= group data by $4;

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

--dump total;

--Count Total Applications who are 'CERTIFIED'

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

temp1= group certified by $4;

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

--dump totalcertified;

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

certifiedWithdrawn= filter data by $1 == 'CERTIFIED-WITHDRAWN';

temp2= group certifiedWithdrawn by $4;

totalcertifiedwithdrawn= foreach temp2 generate group,COUNT(certifiedWithdrawn.$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;

--dump joined;

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

--dump intermediateoutput;

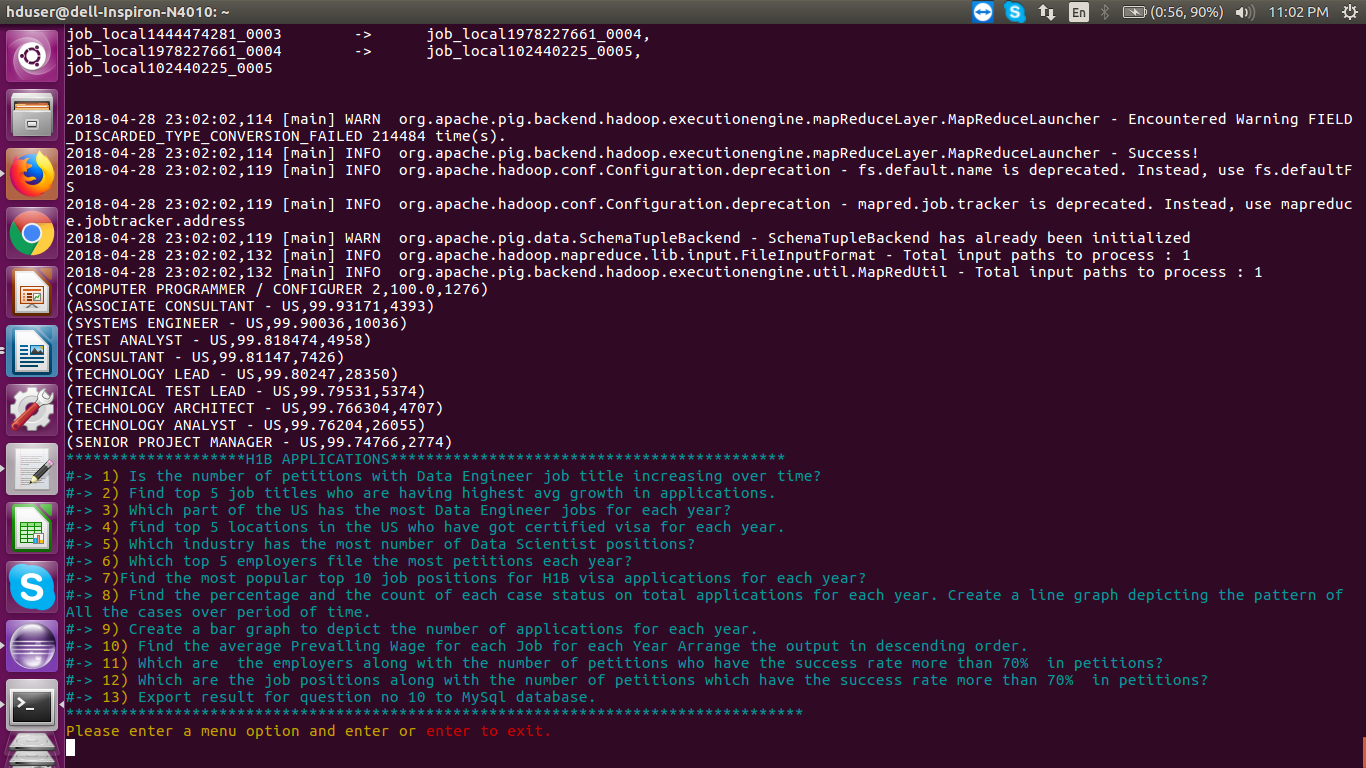
intermediateoutput2= filter intermediateoutput by $1>70 and $2>1000; --Filter by success-rate greater than 70% and petition count above 1000

finaloutput= limit(order intermediateoutput2 by $1 DESC)10;

dump finaloutput;

--store finaloutput into '/home/hduser/Desktop/10.pig';

OUTPUT:

***The job positions along with the number of petitions which have the success rate more than 70% in petitions***

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

**Tool used:SQOOP and MYSQL**

hadoop fs -rm -r -f /Pig/Qns10.out

hadoop fs -mkdir -p /Pig/Qns10.out

hadoop fs -put /home/hduser/Documents/H1bvisaOP/Pig/10output.pig/p\* /Pig/Qns10.out

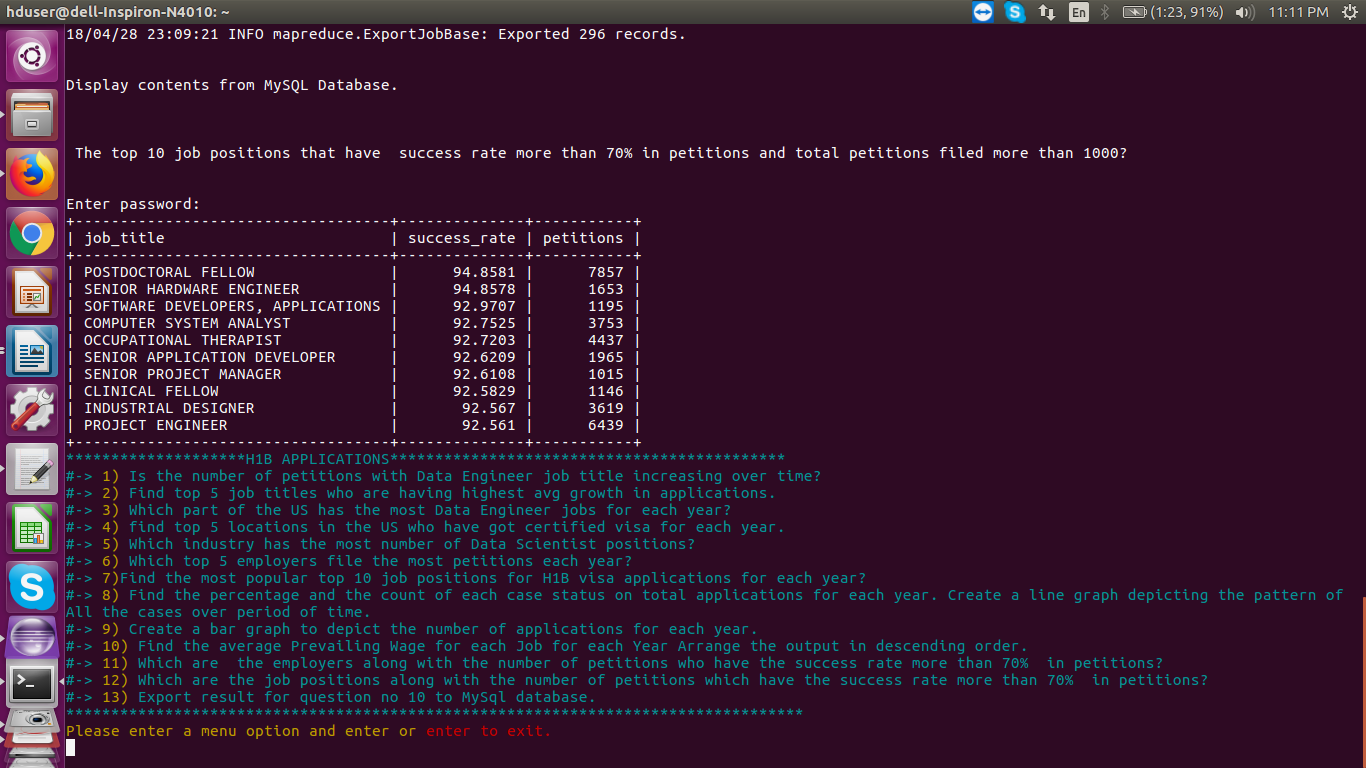
mysql -u root -p -e 'create database if not exists h1bvisa\_job\_success\_rate;use h1bvisa\_job\_success\_rate;create table if not exists job\_success\_rate(job\_title varchar(100),success\_rate float,petitions int);';

sqoop export --connect jdbc:mysql://localhost/h1bvisa\_job\_success\_rate --username root --password 'root' --table job\_success\_rate --update-mode allowinsert --update-key job\_title --export-dir /Pig/Qns10.out/p\* --input-fields-terminated-by '\t' ;

mysql -u root -p -e 'select \* from h1bvisa\_job\_success\_rate.job\_success\_rate limit 10';

OUTPUT:

***Mysql Database result:***



***CONCLUSION***

* + The H-1B visa has been the most popular long-term work visa in the United States for years, and with good reason. There is a whole host of benefits that give the H1B an edge over the other work visa categories. From its accessibility to its lengthy initial period of stay, it’s easy to see why so many foreign professionals apply to reap the H1B visa benefits each year.
  + The first H1B visa benefit, and perhaps the main reason for its popularity, is the broad requirements associated with qualifying for this visa. Another benefit of the H1B visa is the amount of time you are initially granted when you receive your visa.
  + The H1B allows holders to stay for three years initially and can

easily be extended. One of the biggest H-1B benefits is that foreign professionals from all over the worldcan apply.The H-1B is open to nationals and citizens of any country. This is a large advantage over some other visas such as the TN and J1 classifications.

According to the following analysis done with the given data it can be concluded that:

* + More job oppurtunities are available in Taxes,California and New York.
  + Consisder to select the majors in computers
  + If suitable jobs avaiable,try to apply for top 20 firms who filed the most H1Bvisa applications in the year 2014 and 2015.
  + Companies that are located in the North are offering more salaries than the others.