# Big Data Hadoop Training

Project: State-wise Development Analysis Solution

**1. Executive Summary**

**1.1 Project Overview**

To develop the System to analyze the log data (In XML format) of government progress of various development activities.

**1.2 Purpose and Scope of this Specification**

The purpose of this project is to capture the data for analyzing the progress of various activities.

**In scope**

This following requirement will be addressed in phase 1 of Project:

* Developing system to handle the incoming log feed and store the information in Hadoop Cluster(Flume)
* Analyze the data and understand the progress
* Store the results in hbase/RDBMS

**2. Product/Service Description**

**2.1 Assumptions**

Log will be generated in XML format and stored in a server

**2.2 Constraints**

Describe any items that will constrain the design options, including

* This system may not be used for searching for now. But it will be used for analysis and saving the relevant information as of now
* System will be using mysql as a database

**3. Requirements**

* The FLUME job which will place format the data and place the data to HDFS
* Pig/Mapreduce job for parsing the XML data.
* Create pig scripts/Mapreduce jobs to analyze the data
* Create the sqoop job to store the data in database

**Priority Definitions**

The following definitions are intended as a guideline to prioritize requirements.

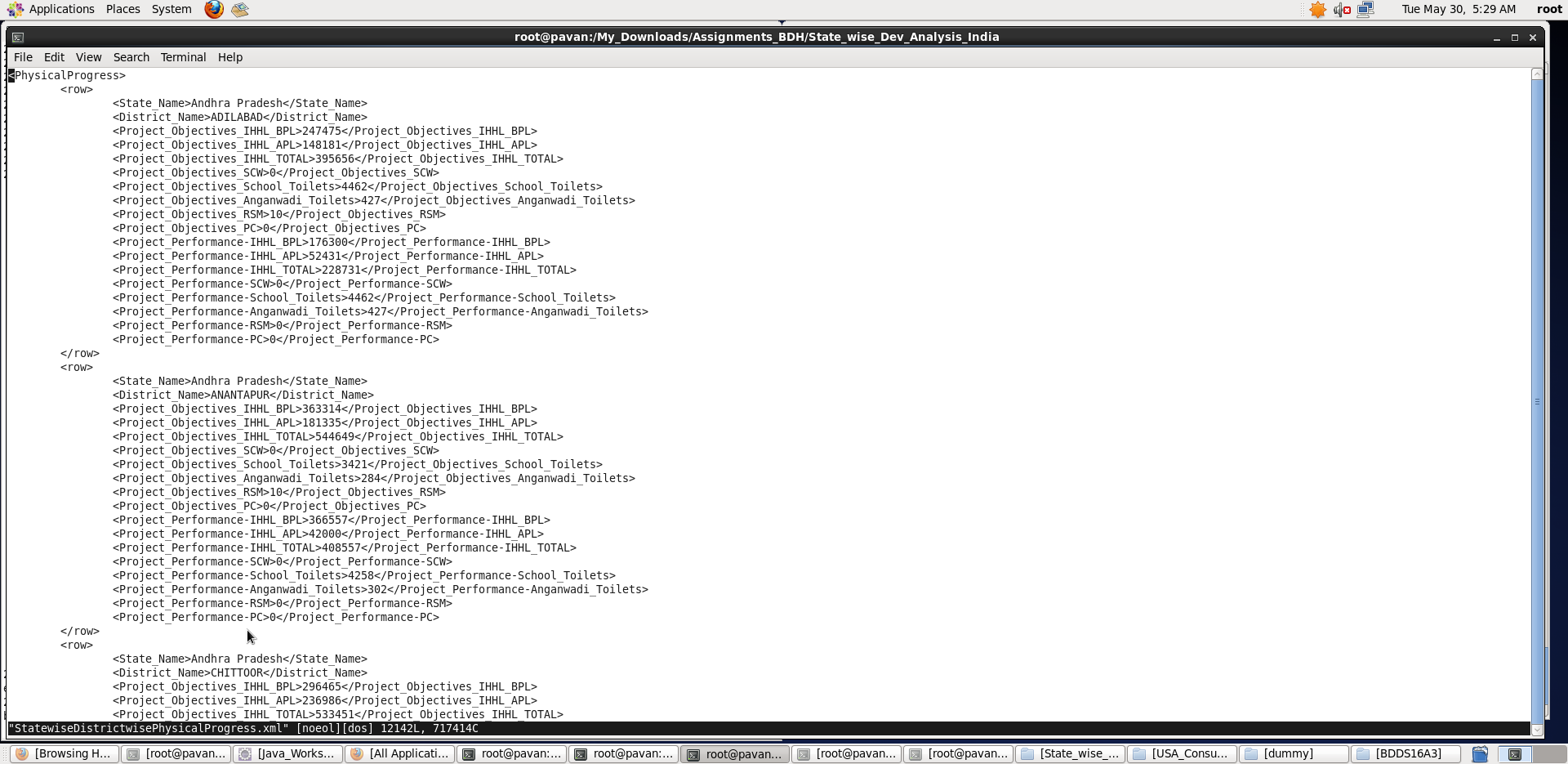
Priority 1 – Create FLUME job for fetching log files from spool directory into HDFS

Priority 2 – Mapreduce/pig job to preprocess

**Download the dataset using the below link:**

**Link:** <https://drive.google.com/file/d/0Bxr27gVaXO5sUjd2RWFQS3hQQUE/view?usp=sharing>

Data kept in **/My\_Downloads/Assignments\_BDH/State\_wise\_Dev\_Analysis\_India local FS**

****

**Steps followed to create the above project.**

**Step 1:**

**Copy dataset from local file system to HDFS using flume.**

1. We will be using spool directory as our source and HDFS as destination.  
   Here we will create an agent for flume (here data is fetched from Local File System).

**Flume\_data\_transfer\_State\_Dev\_Analysis.conf file: (put it in /etc/flume-ng/conf directory)**

#Flume Configuration Starts

dev\_analysis.sources = localsrc

dev\_analysis.channels = fileChannel

dev\_analysis.sinks = hdfsdest

# Define a source for dev\_analysis on linux FS

dev\_analysis.sources.localsrc.type = spooldir

#Spooldir in my case is /My\_Downloads/Assignments\_BDH/State\_wise\_Dev\_Analysis\_India

dev\_analysis.sources.localsrc.spoolDir = /My\_Downloads/Assignments\_BDH/State\_wise\_Dev\_Analysis\_India

dev\_analysis.sources.localsrc.fileHeader = false

dev\_analysis.sources.localsrc.fileSuffix = .COMPLETED

# Define a file channel called fileChannel on dev\_analysis on linux FS

dev\_analysis.channels.fileChannel.type = memory

dev\_analysis.channels.fileChannel.capacity = 200000

dev\_analysis.channels.fileChannel.transactionCapacity = 1000

# Define a sink for dev\_analysis on HDFS

dev\_analysis.sinks.hdfsdest.type = hdfs

#Sink is /tmp/State\_wise\_Dev\_Analysis\_India/flume\_import under hdfs

dev\_analysis.sinks.hdfsdest.hdfs.path = hdfs://pavan:8020/tmp/State\_wise\_Dev\_Analysis\_India/flume\_import

dev\_analysis.sinks.hdfsdest.hdfs.batchSize = 1000

dev\_analysis.sinks.hdfsdest.hdfs.rollSize = 268435456

dev\_analysis.sinks.hdfsdest.hdfs.rollInterval = 0

dev\_analysis.sinks.hdfsdest.hdfs.rollCount = 50000000

dev\_analysis.sinks.hdfsdest.hdfs.writeFormat=Text

dev\_analysis.sinks.hdfsdest.hdfs.fileType = DataStream

# Associating Channel with both Source and Sink

dev\_analysis.sources.localsrc.channels = fileChannel

dev\_analysis.sinks.hdfsdest.channel = fileChannel

Note: This is a basic conf file configuring the source, channel and sink.

**dev\_analysis** is the flume agent name

1) dev\_analysis.sources.source1\_1.spoolDir is set with input path as in local file system path.

Here <input\_path> is < /My\_Downloads/Assignments\_BDH/State\_wise\_Dev\_Analysis\_India>

2) dev\_analysis.sinks.hdfs-sink1\_1.hdfs.path is set with output path as in HDFS path

Here <output\_path> is <hdfs://pavan:8020/tmp/State\_wise\_Dev\_Analysis\_India/flume\_import>

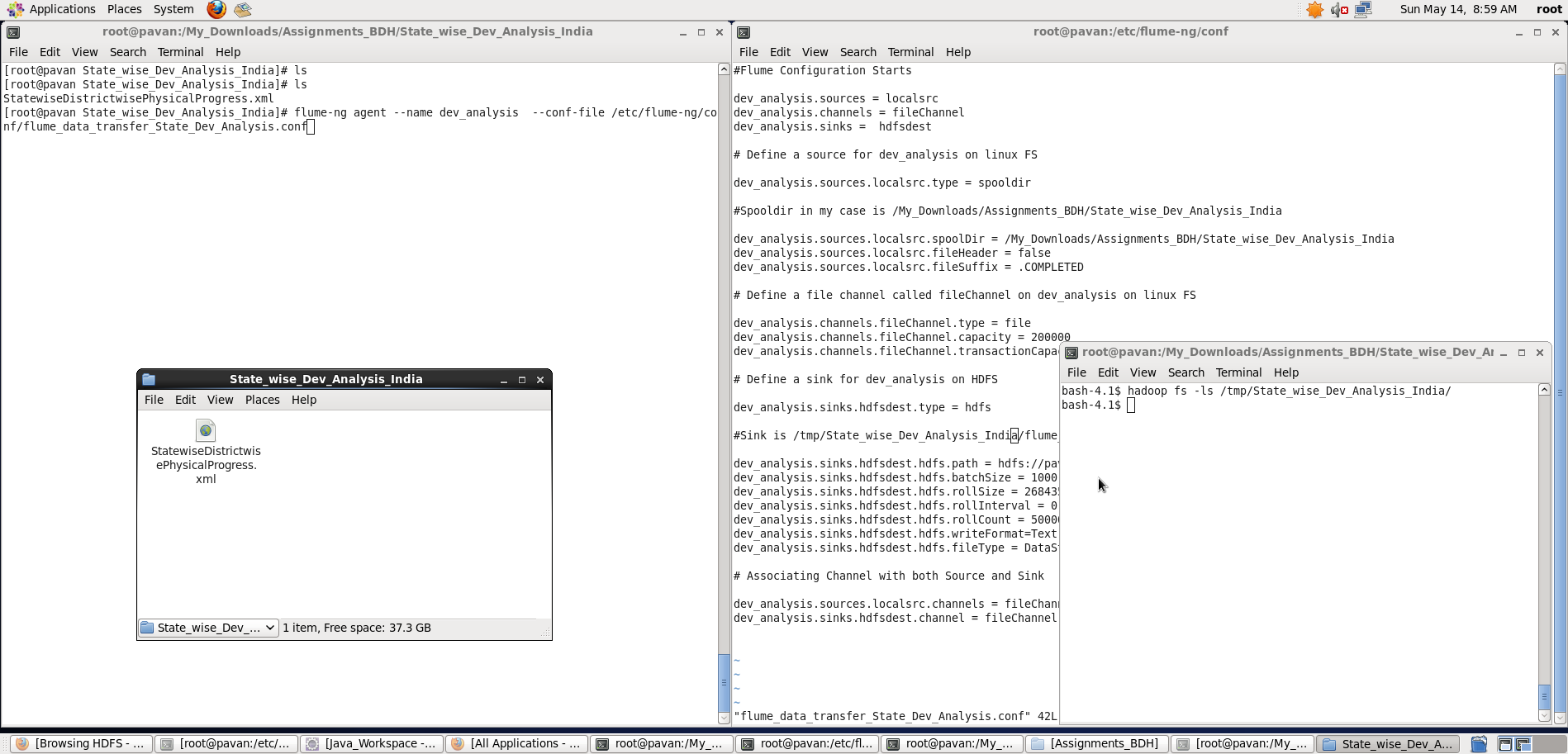
Note: Hostname : pavan , PortNo: 8020 to connect to my HDFS.

**Create specified input and output paths and give read/write permissions accordingly.**

**We will now open another terminal and start flume agent by the following**

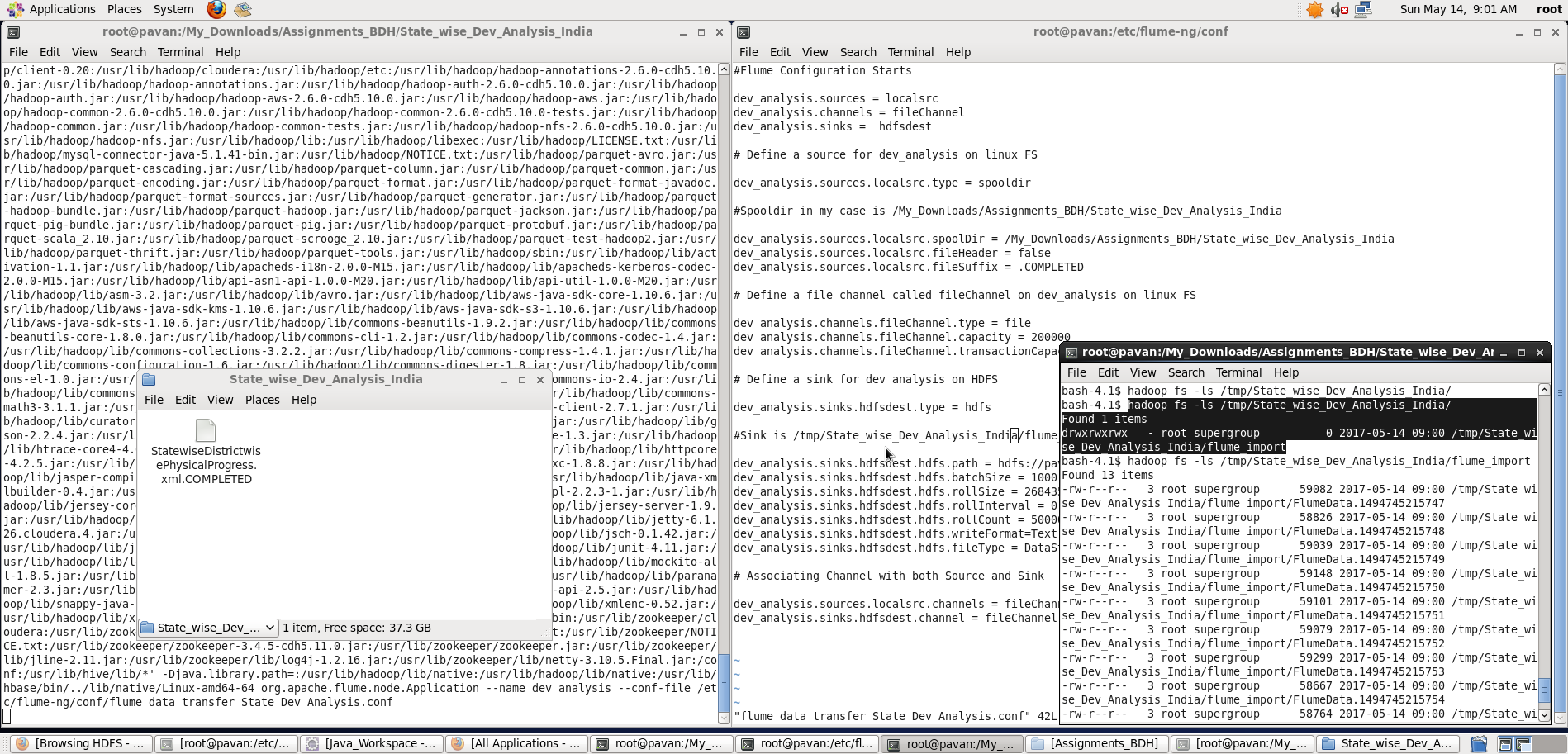
**Command:**

flume-ng agent --name dev\_analysis --conf-file /etc/flume-ng/conf/flume\_data\_transfer\_State\_Dev\_Analysis.conf



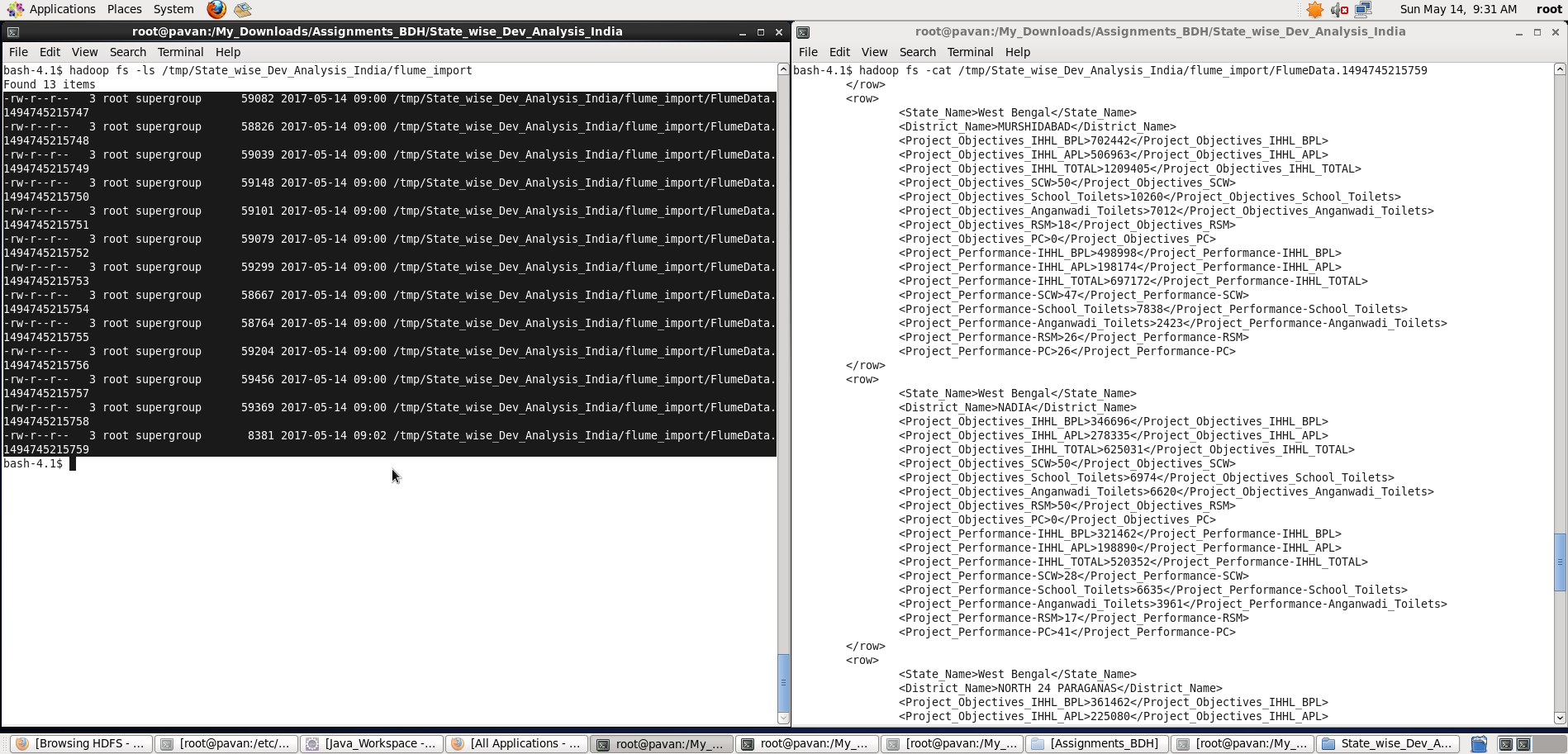
**Once the flume starts running, the dataset we placed in the SpoolDir gets copied into HDFS.**

**Once, the copying is done, it marks the input files as .COMPLETED (we specified in flume\_data\_transfer\_State\_wise\_Dev\_Analysis.conf file)**



**We can see the dataset being copied into specified sink directory in HDFS using the command:**

**$ hadoop fs –cat /tmp/State\_wise\_Dev\_Analysis/flume\_import/FlumeData.1494745215759**



**Now, we work on problem statements in which I am loading the data from HDFS.**

**Step 2:** Input file is in the XML format use Map reduce or pig to parse the data and get the results for the below problem statements.

**Note: I have used Pig Scripts to get the results for the below problem statements.**

**Problem statement 1:**

1. Find out the districts who achieved 100 percent objective in BPL cards

A. **Pig Latin Script:**

register '/usr/lib/pig/piggybank.jar';

xmldata\_from\_hdfs = LOAD 'hdfs://pavan:8020/tmp/State\_wise\_Dev\_Analysis\_India/flume\_import/' using org.apache.pig.piggybank.storage.XMLLoader('row') as (doc:chararray);

xmldata\_parsed = foreach xmldata\_from\_hdfs GENERATE FLATTEN(REGEX\_EXTRACT\_ALL(doc,'<row>\\s\*<State\_Name>(.\*)</State\_Name>\\s\*<District\_Name>(.\*)</District\_Name>\\s\*<Project\_Objectives\_IHHL\_BPL>(.\*)</Project\_Objectives\_IHHL\_BPL>\\s\*<Project\_Objectives\_IHHL\_APL>(.\*)</Project\_Objectives\_IHHL\_APL>\\s\*<Project\_Objectives\_IHHL\_TOTAL>(.\*)</Project\_Objectives\_IHHL\_TOTAL>\\s\*<Project\_Objectives\_SCW>(.\*)</Project\_Objectives\_SCW>\\s\*<Project\_Objectives\_School\_Toilets>(.\*)</Project\_Objectives\_School\_Toilets>\\s\*<Project\_Objectives\_Anganwadi\_Toilets>(.\*)</Project\_Objectives\_Anganwadi\_Toilets>\\s\*<Project\_Objectives\_RSM>(.\*)</Project\_Objectives\_RSM>\\s\*<Project\_Objectives\_PC>(.\*)</Project\_Objectives\_PC>\\s\*<Project\_Performance-IHHL\_BPL>(.\*)</Project\_Performance-IHHL\_BPL>\\s\*<Project\_Performance-IHHL\_APL>(.\*)</Project\_Performance-IHHL\_APL>\\s\*<Project\_Performance-IHHL\_TOTAL>(.\*)</Project\_Performance-IHHL\_TOTAL>\\s\*<Project\_Performance-SCW>(.\*)</Project\_Performance-SCW>\\s\*<Project\_Performance-School\_Toilets>(.\*)</Project\_Performance-School\_Toilets>\\s\*<Project\_Performance-Anganwadi\_Toilets>(.\*)</Project\_Performance-Anganwadi\_Toilets>\\s\*<Project\_Performance-RSM>(.\*)</Project\_Performance-RSM>\\s\*<Project\_Performance-PC>(.\*)</Project\_Performance-PC>\\s\*</row>')) AS (state\_name:chararray, district\_name:chararray, po\_ihhl\_bpl:int, po\_ihhl\_apl:int, po\_ihhl\_total:int, po\_scw:int, po\_sch\_toilets:int, po\_angan\_toilets:int, po\_rsm:int, po\_pc:int, pp\_ihhl\_bpl:int, pp\_ihhl\_apl:int, pp\_ihhl\_total:int, pp\_scw:int, pp\_sch\_toilets:int, pp\_angan\_toilets:int, pp\_rsm:int, pp\_pc:int);

data\_with\_full\_percent\_bpl\_cards = filter xmldata\_parsed by po\_ihhl\_bpl==pp\_ihhl\_bpl;

districts\_with\_full\_percent\_bpl\_cards = foreach data\_with\_full\_percent\_bpl\_cards generate district\_name,po\_ihhl\_bpl,pp\_ihhl\_bpl;

--dump districts\_with\_full\_percent\_bpl\_cards;

store districts\_with\_full\_percent\_bpl\_cards into 'hdfs://pavan:8020/tmp/project\_2/State\_wise\_Dev\_Analysis\_results' using PigStorage (',');

**Explanation :**

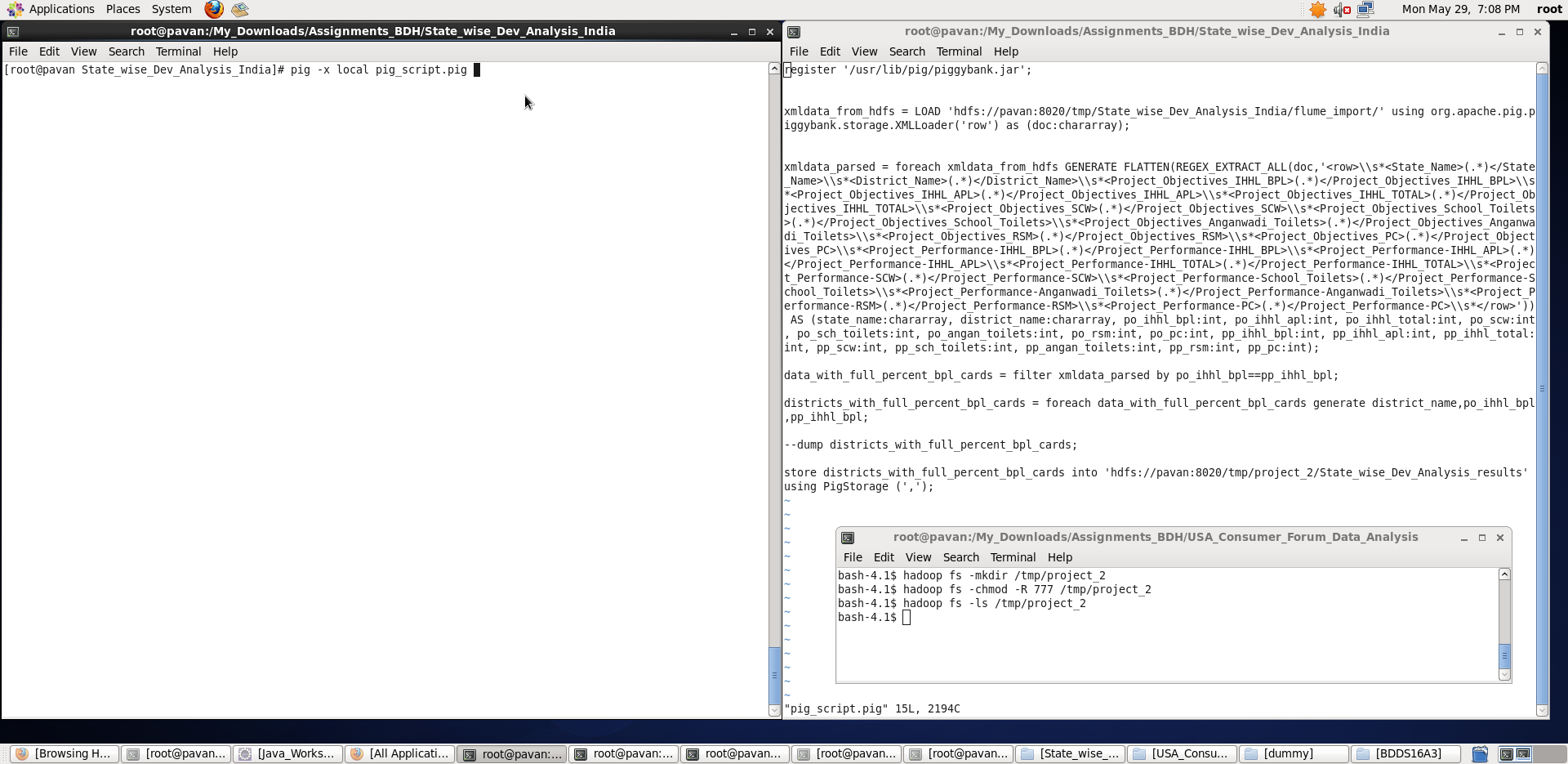
* Loaded data from HDFS directory location: **'hdfs://pavan:8020/tmp/State\_wise\_Dev\_Analysis\_India/flume\_import/'** into “xmldata\_from\_hdfs” relation using XMLLoader and schema is provided accordingly.

**Note: In order to load the XML data into the Pig structures, one needs to parse the XML so that Pig can understand it. Thus we shall use the XMLLoader() function which is present in PiggyBank of Apache. PiggyBank is a repository of Java user defined functions.**

* Then, all records are filtered based on 100 percent objective in BPL cards i.e. (po\_ihl\_bpl==pp\_ihhl\_bpl)
* Now, foreach data\_with\_full\_percent\_bpl\_cards , I have generated the district\_name, po\_ihhl\_bpl and pp\_ihhl\_bpl records.
* **Storing the result relation into a HDFS directory** 'hdfs://pavan:8020/tmp/project\_2/State\_wise\_Dev\_Analysis\_results’**.**

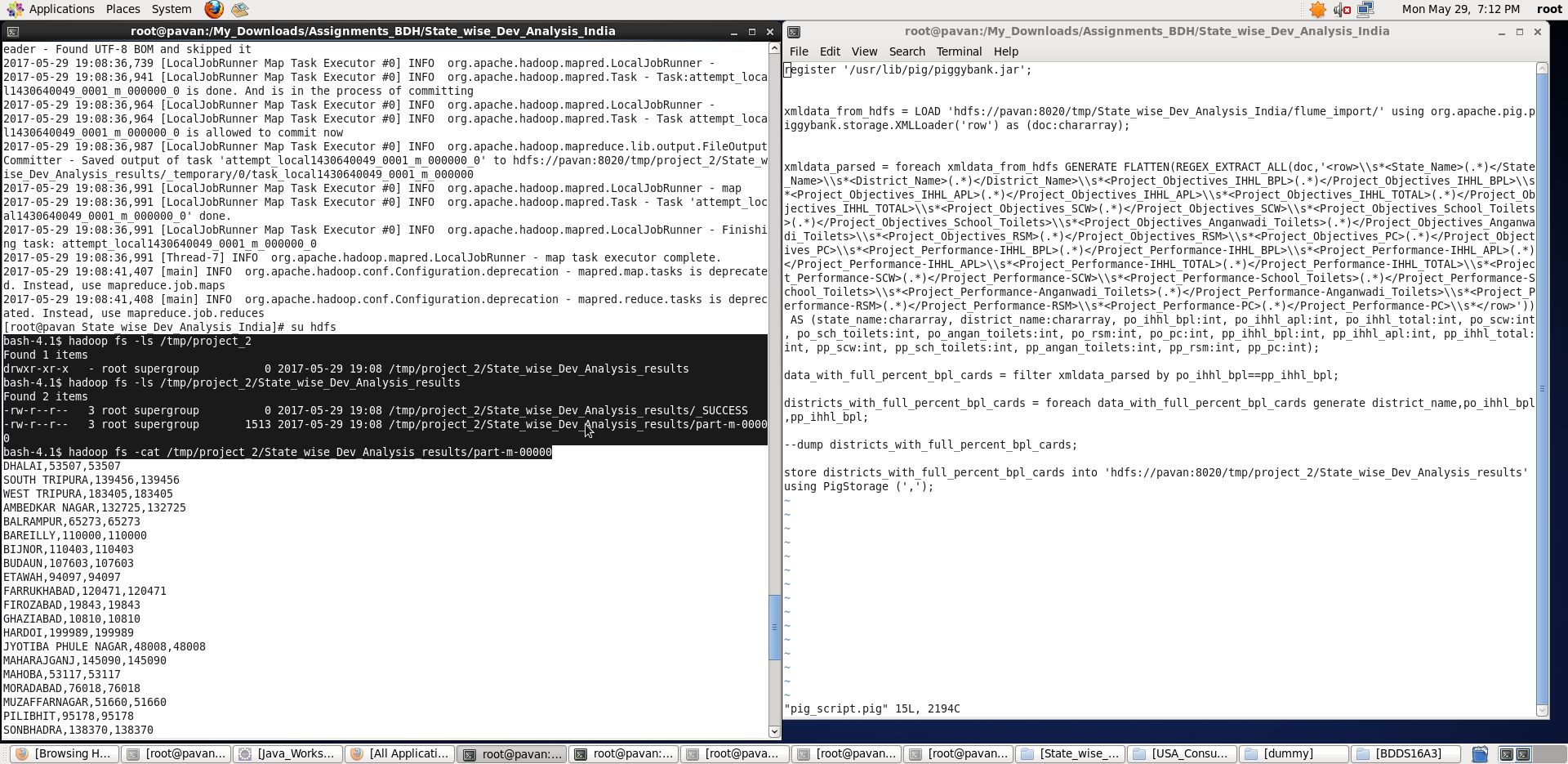
Screenshots

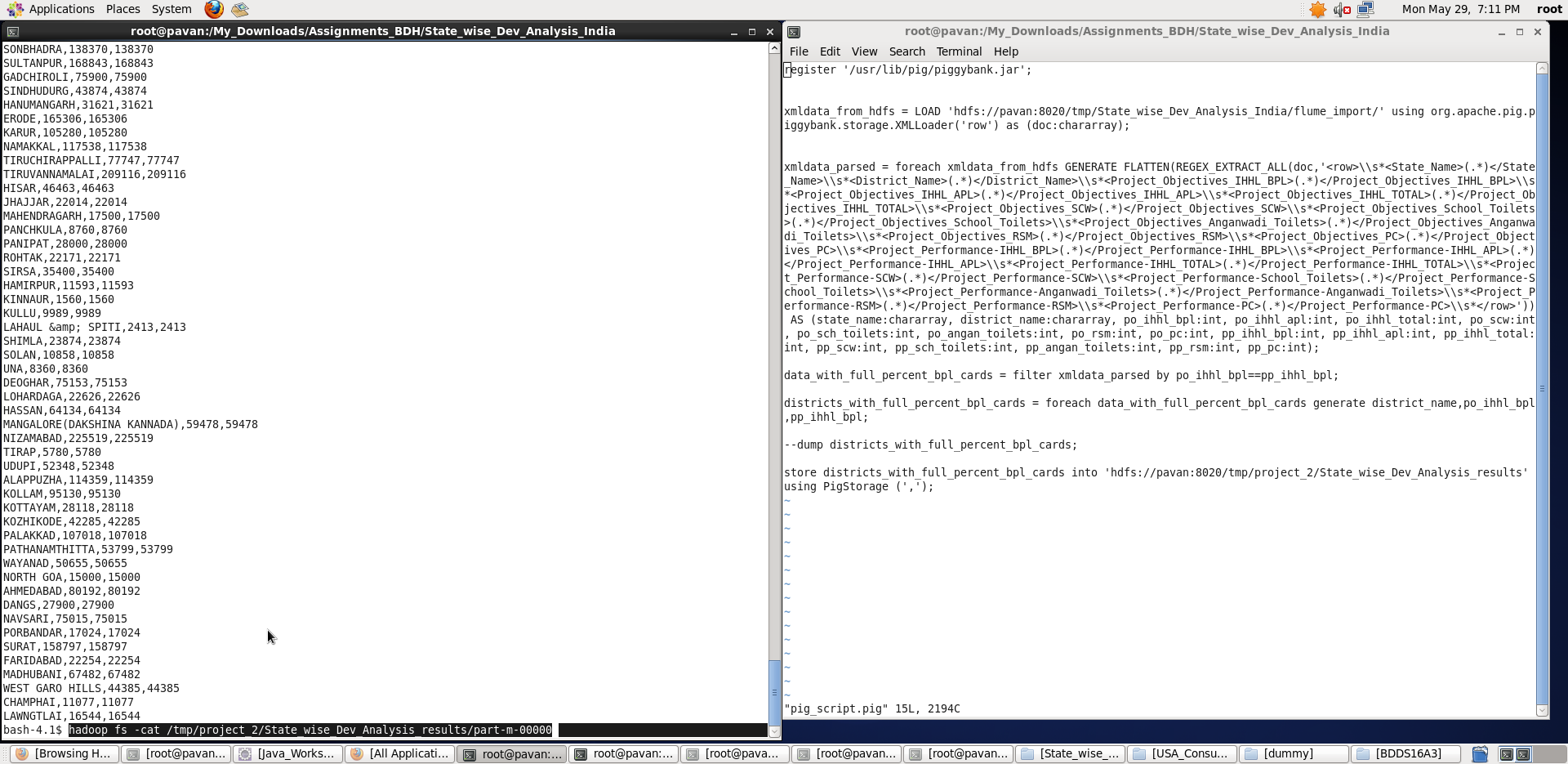
Running **Pig Latin Script in Local Mode** inside /My\_Downloads/Assignments\_BDH/State\_wise\_Dev\_Analysis\_India/ directory



**Output stored in the format**: **<District\_Name>,<po\_ihhl\_bpl>,<pp\_ihhl\_bpl>**

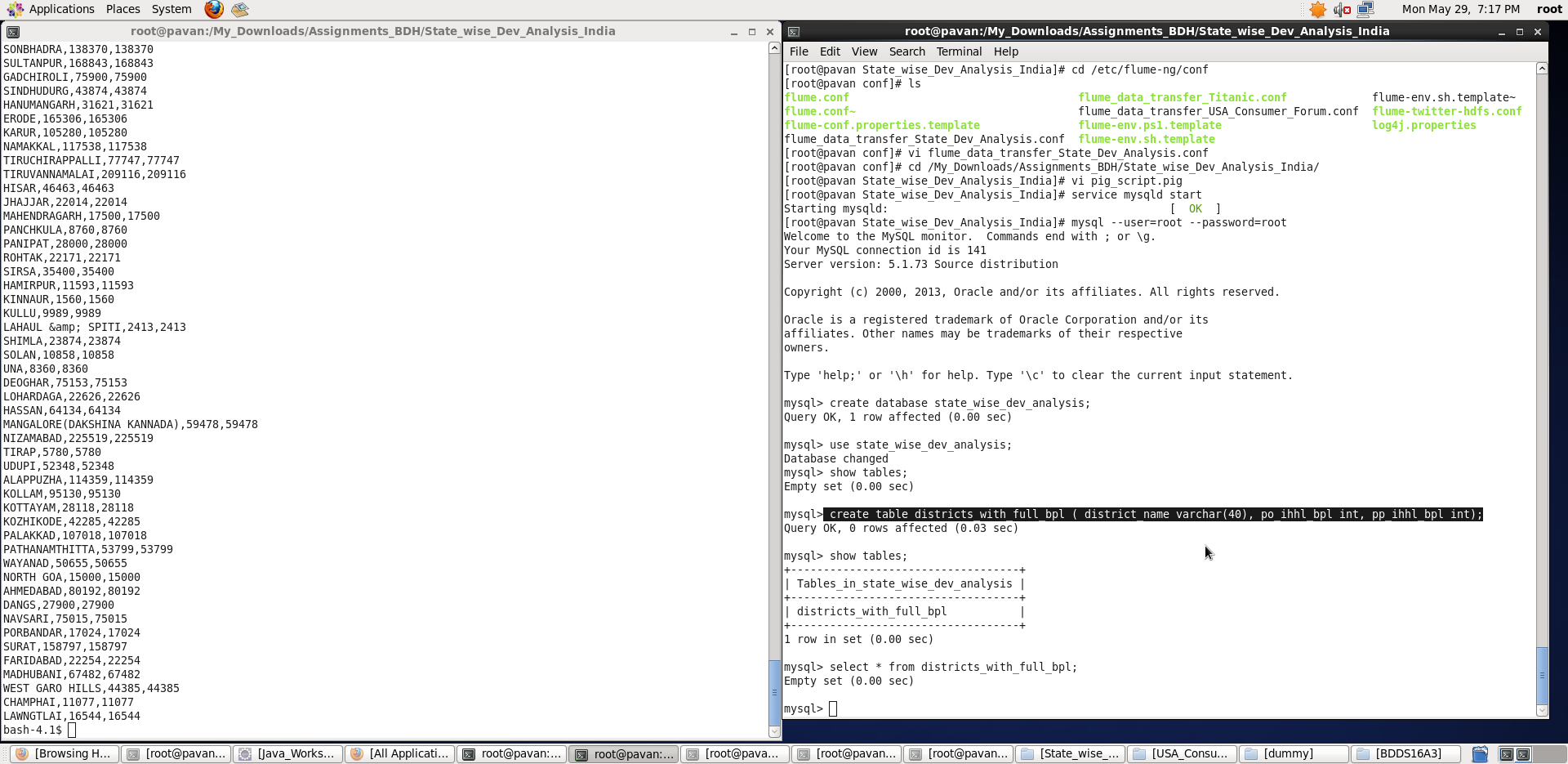
**We can see the output stored in the specified HDFS directory as below:**



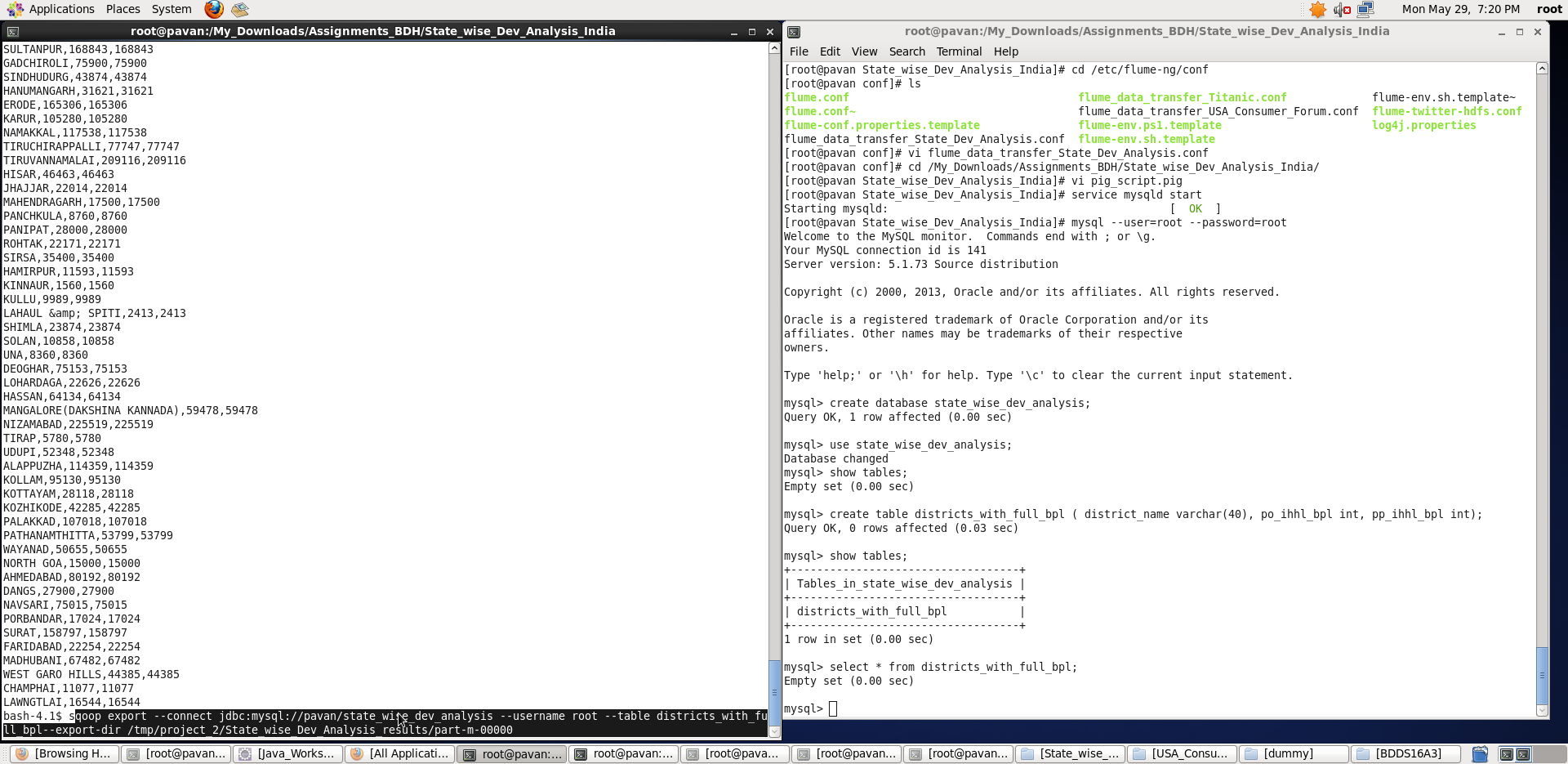


**Export the results to mysql using sqoop :**

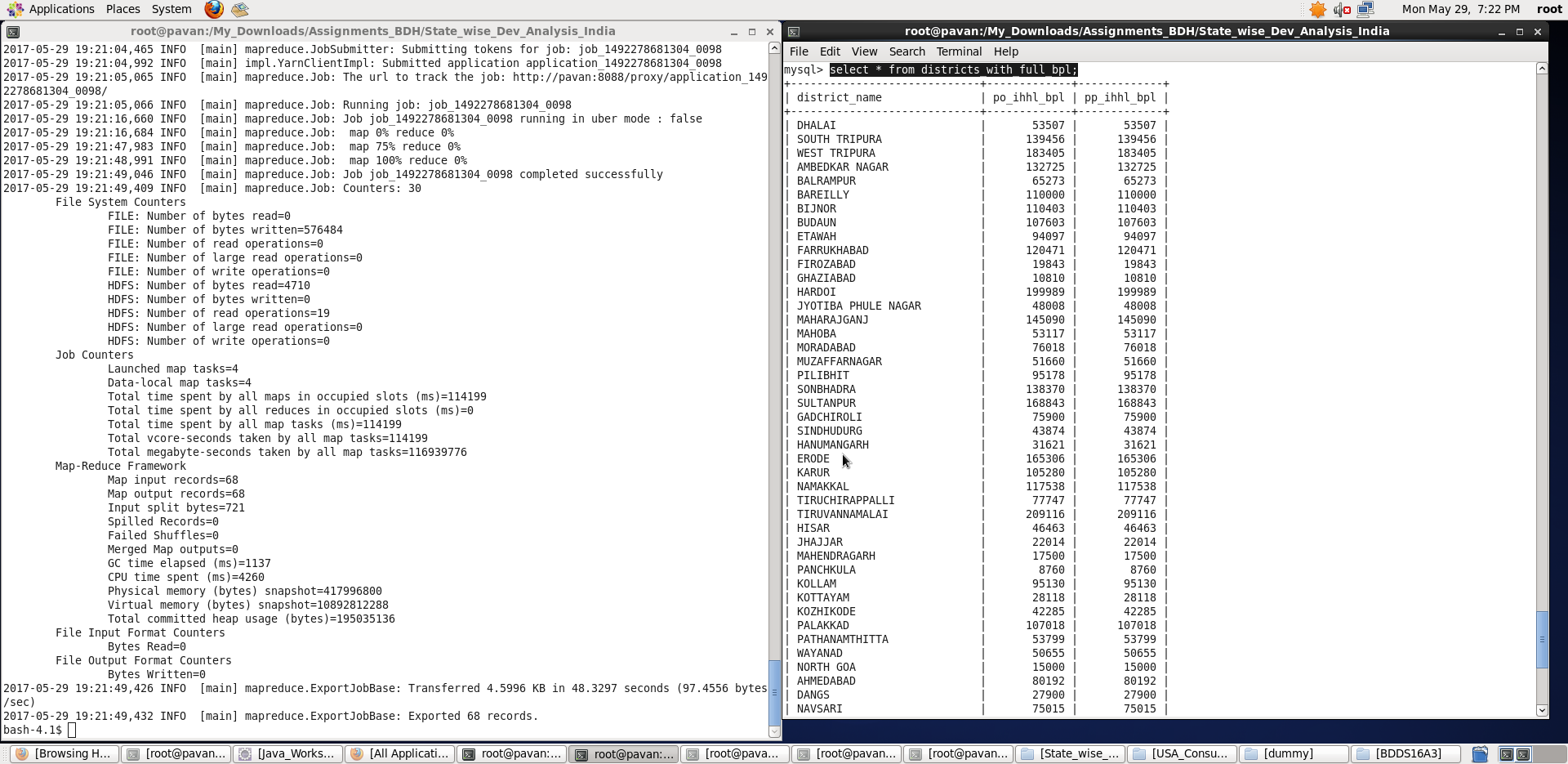
**For this, the table should be already present in mysql before we export the result from HDFS**

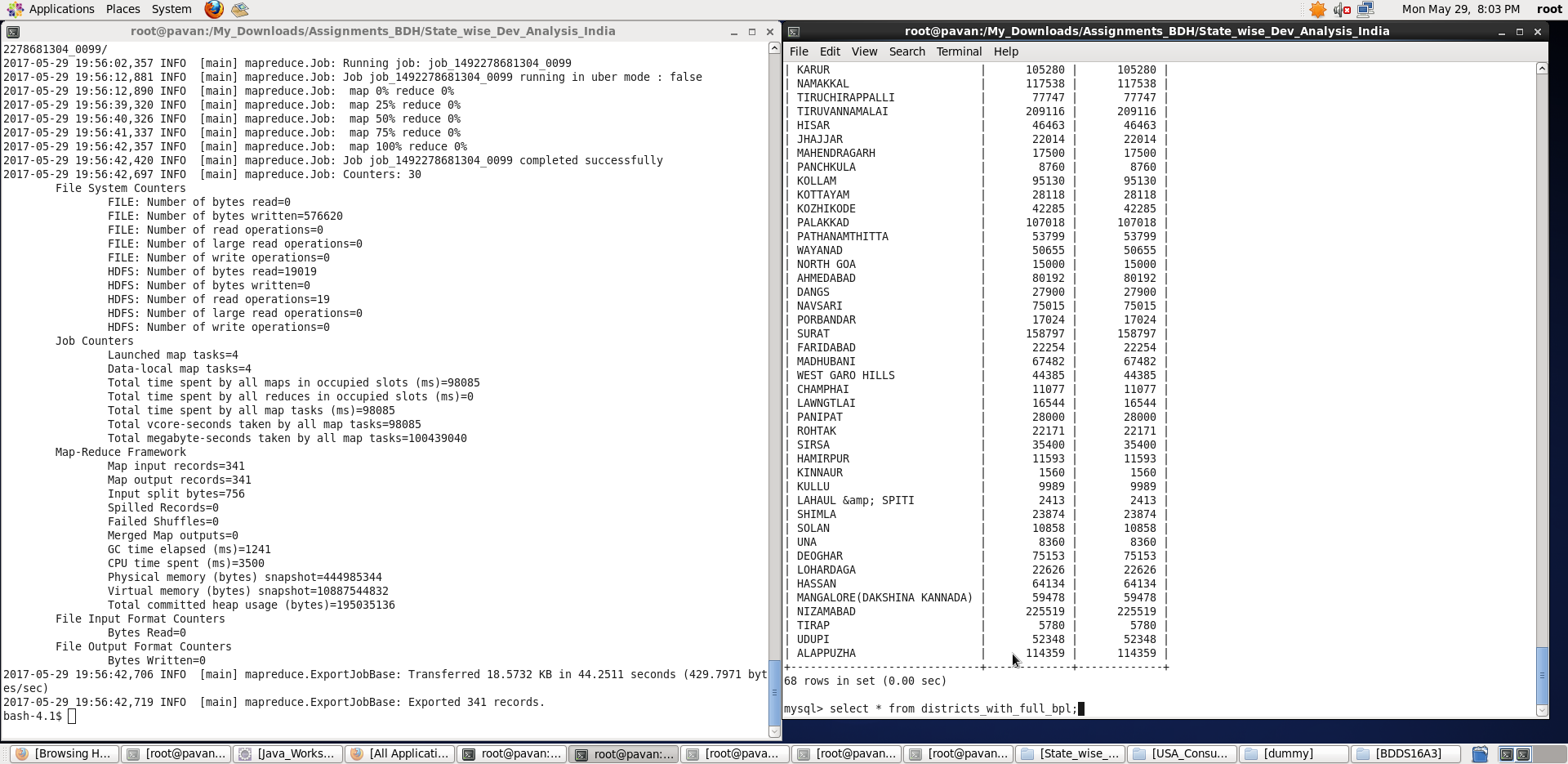


**Once, the table is created, use the sqoop export command as shown below to export the results into mysql.**



**Once, the export is done, we can see results getting stored in the districts\_with\_full\_bpl table inside state\_wise\_dev\_analysis database using “*select \* from districts\_with\_full\_bpl”* command.**





**Thus, we found the districts who achieved 100 percent objective in BPL cards and exported the results to mysql using sqoop.**

**Note: I have considered exact 100 percent objective achievement in BPL cards**

**Problem statement 2:**

2. Write a Pig UDF to filter the districts who have reached 80% of objectives of BPL cards.

1. **Steps involved in writing a Pig UDF:**
2. Create Java Class for a User Defined Function which extends org.apache.pig.FilterFunc.

**Java code:**

import java.io.IOException;

import org.apache.pig.FilterFunc;

import org.apache.pig.data.Tuple;

public class pig\_udf\_filter\_districts extends FilterFunc {

@Override

public Boolean exec(Tuple input) throws IOException {

// TODO Auto-generated method stub

try {

int po\_ihhl\_bpl = Integer.parseInt(input.get(0).toString());

int pp\_ihhl\_bpl = Integer.parseInt(input.get(1).toString());

if (pp\_ihhl\_bpl >= (0.8)\*po\_ihhl\_bpl)

return true;

else

return false;

}

catch(ArrayIndexOutOfBoundsException e){

System.out.println("FAILED: SemanticException : Arguments length mismatch");

System.out.println("Input Tuple should consist of two arguments");

}

return null;

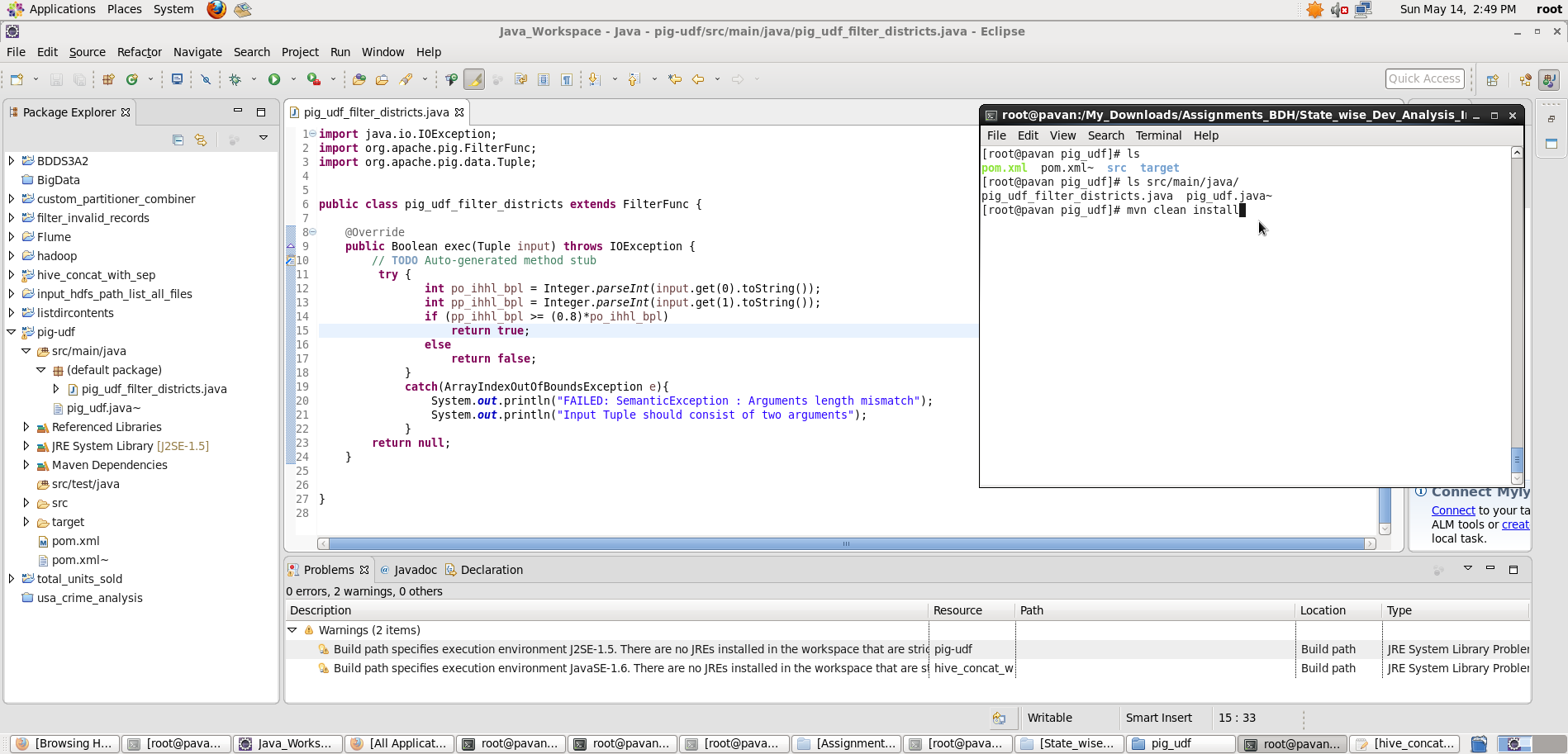
}

}

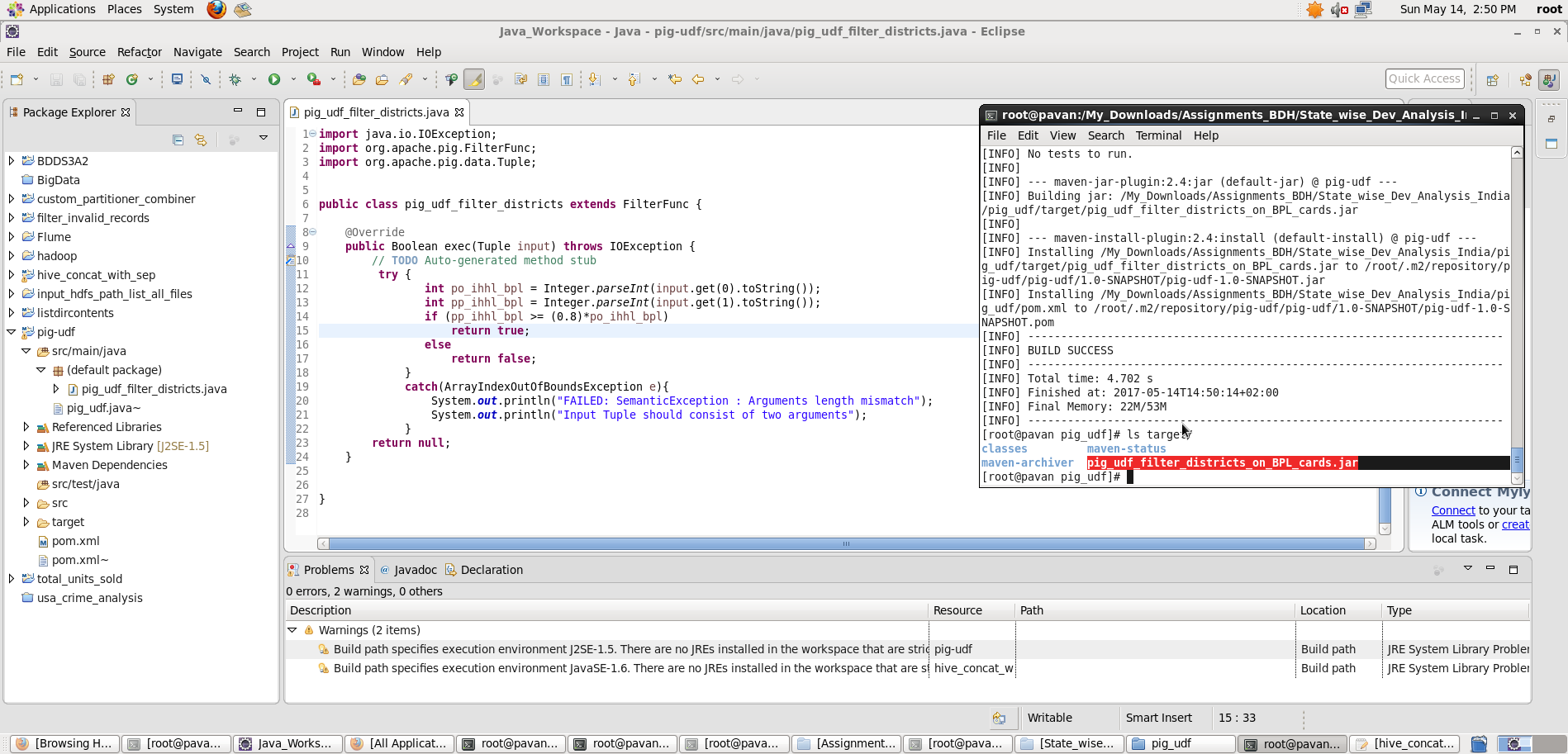
1. Packaging the Java class into a JAR file (I am using Maven)

Make the folder structure pig\_udf/src/main/java/pig\_udf\_filter\_districts.java. Place the pom.xml file in pig\_udf/ and run the maven build command

**# mvn clean install**



Once, the build is successful, we can see the **pig\_udf\_filter\_districts\_on\_BPL\_cards.jar** being created inside the target directory.



1. Now, simply, register the jar inside Pig Latin script and use the UDF with class name - **pig\_udf\_filter\_districts**

and **pass the required arguments** as follows:

**Pig Latin Script:**

register ./pig\_udf/target/pig\_udf\_filter\_districts\_on\_BPL\_cards.jar;

register '/usr/lib/pig/piggybank.jar';

xmldata\_from\_hdfs = LOAD 'hdfs://pavan:8020/tmp/State\_wise\_Dev\_Analysis\_India/flume\_import/' using org.apache.pig.piggybank.storage.XMLLoader('row') as (doc:chararray);

xmldata\_parsed = foreach xmldata\_from\_hdfs GENERATE FLATTEN(REGEX\_EXTRACT\_ALL(doc,'<row>\\s\*<State\_Name>(.\*)</State\_Name>\\s\*<District\_Name>(.\*)</District\_Name>\\s\*<Project\_Objectives\_IHHL\_BPL>(.\*)</Project\_Objectives\_IHHL\_BPL>\\s\*<Project\_Objectives\_IHHL\_APL>(.\*)</Project\_Objectives\_IHHL\_APL>\\s\*<Project\_Objectives\_IHHL\_TOTAL>(.\*)</Project\_Objectives\_IHHL\_TOTAL>\\s\*<Project\_Objectives\_SCW>(.\*)</Project\_Objectives\_SCW>\\s\*<Project\_Objectives\_School\_Toilets>(.\*)</Project\_Objectives\_School\_Toilets>\\s\*<Project\_Objectives\_Anganwadi\_Toilets>(.\*)</Project\_Objectives\_Anganwadi\_Toilets>\\s\*<Project\_Objectives\_RSM>(.\*)</Project\_Objectives\_RSM>\\s\*<Project\_Objectives\_PC>(.\*)</Project\_Objectives\_PC>\\s\*<Project\_Performance-IHHL\_BPL>(.\*)</Project\_Performance-IHHL\_BPL>\\s\*<Project\_Performance-IHHL\_APL>(.\*)</Project\_Performance-IHHL\_APL>\\s\*<Project\_Performance-IHHL\_TOTAL>(.\*)</Project\_Performance-IHHL\_TOTAL>\\s\*<Project\_Performance-SCW>(.\*)</Project\_Performance-SCW>\\s\*<Project\_Performance-School\_Toilets>(.\*)</Project\_Performance-School\_Toilets>\\s\*<Project\_Performance-Anganwadi\_Toilets>(.\*)</Project\_Performance-Anganwadi\_Toilets>\\s\*<Project\_Performance-RSM>(.\*)</Project\_Performance-RSM>\\s\*<Project\_Performance-PC>(.\*)</Project\_Performance-PC>\\s\*</row>')) AS (state\_name:chararray, district\_name:chararray, po\_ihhl\_bpl:int, po\_ihhl\_apl:int, po\_ihhl\_total:int, po\_scw:int, po\_sch\_toilets:int, po\_angan\_toilets:int, po\_rsm:int, po\_pc:int, pp\_ihhl\_bpl:int, pp\_ihhl\_apl:int, pp\_ihhl\_total:int, pp\_scw:int, pp\_sch\_toilets:int, pp\_angan\_toilets:int, pp\_rsm:int, pp\_pc:int);

data\_with\_eighty\_percent\_bpl\_cards = filter xmldata\_parsed by pig\_udf\_filter\_districts(po\_ihhl\_bpl,pp\_ihhl\_bpl);

districts\_with\_eighty\_percent\_bpl\_cards = foreach data\_with\_eighty\_percent\_bpl\_cards generate district\_name,po\_ihhl\_bpl,pp\_ihhl\_bpl;

--dump districts\_with\_eighty\_percent\_bpl\_cards;

store districts\_with\_eighty\_percent\_bpl\_cards into 'hdfs://pavan:8020/tmp/project\_2/state\_wise\_dev\_analysis\_results\_task\_b' using PigStorage(',');

**Explanation :**

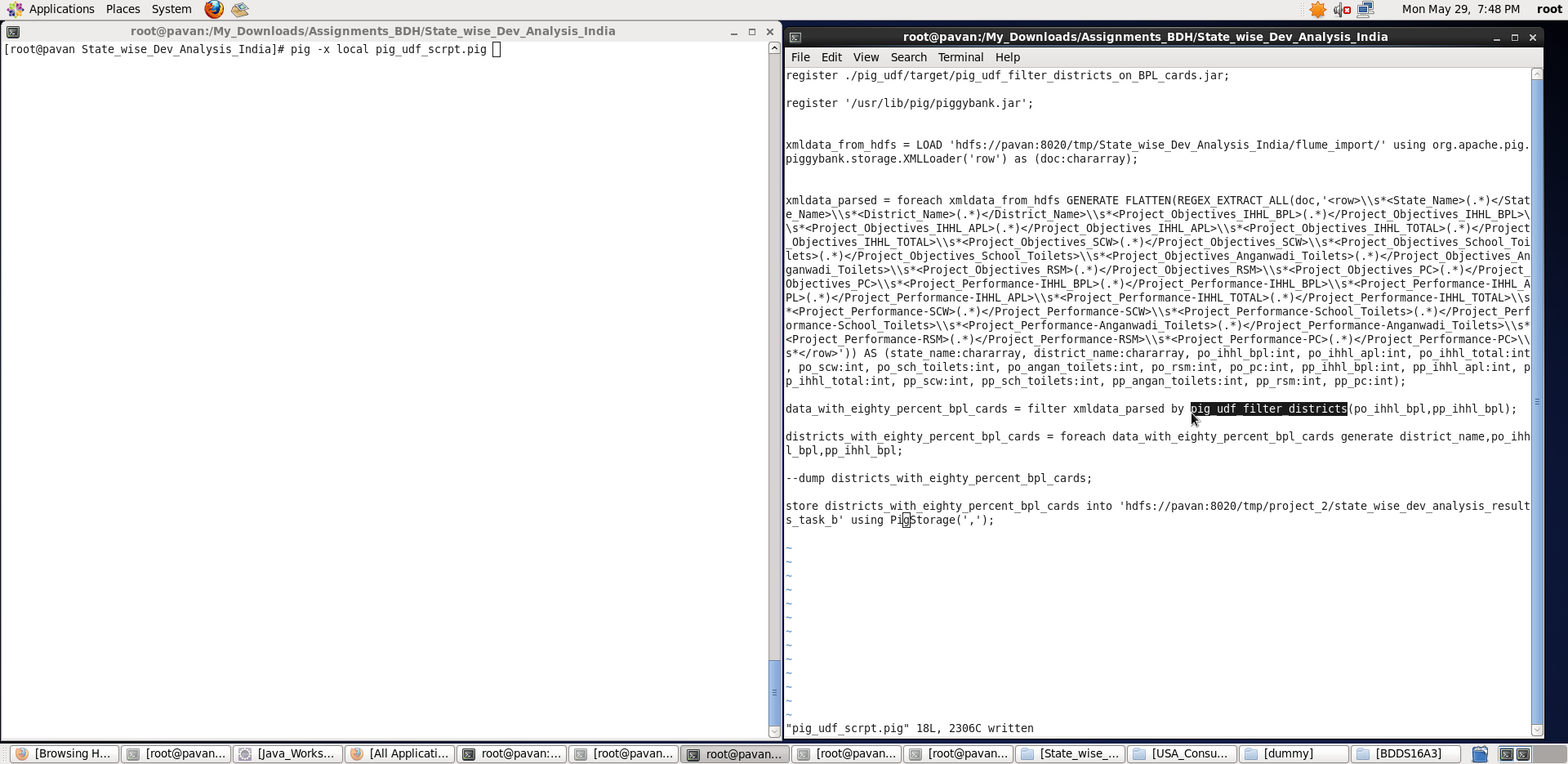
* Loaded data from HDFS directory location: **'hdfs://pavan:8020/tmp/State\_wise\_Dev\_Analysis\_India/flume\_import/'** into “xmldata\_from\_hdfs” relation using XMLLoader and schema is provided accordingly.

**Note: In order to load the XML data into the Pig structures, one needs to parse the XML so that Pig can understand it. Thus we shall use the XMLLoader() function which is present in PiggyBank of Apache. PiggyBank is a repository of Java user defined functions.**

* Then, all records are filtered with the help of Pig UDF (that we have created previously)
* Now, foreach data\_with\_eighty\_percent\_bpl\_cards , I have generated the district\_name, po\_ihhl\_bpl and pp\_ihhl\_bpl records.
* **Storing the result relation into a HDFS directory** 'hdfs://pavan:8020/tmp/project\_2/state\_wise\_dev\_analysis\_results\_task\_b’**.**

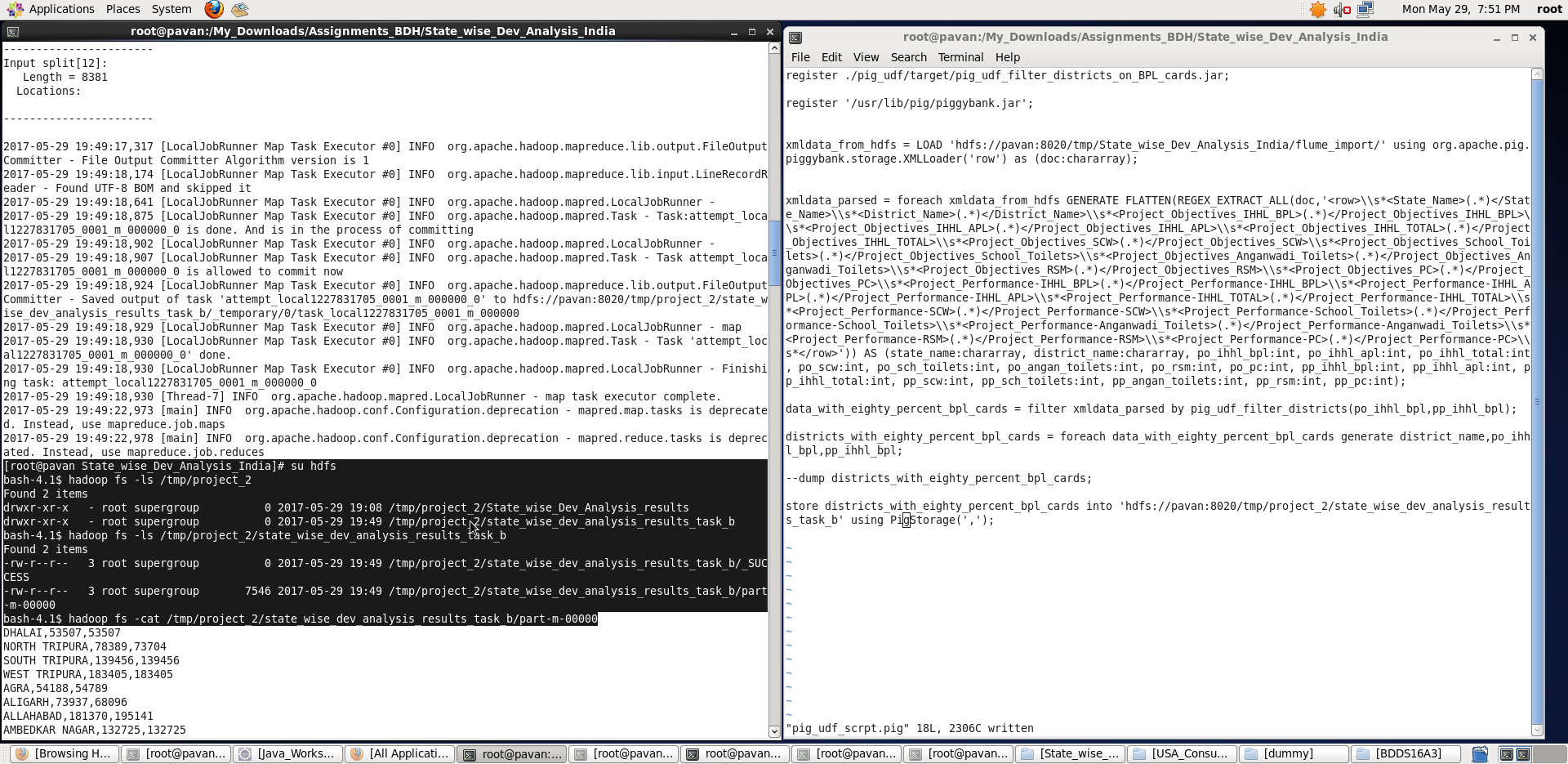
Screenshots

Running **Pig Latin Script in Local Mode** inside /My\_Downloads/Assignments\_BDH/State\_wise\_Dev\_Analysis\_India/ directory



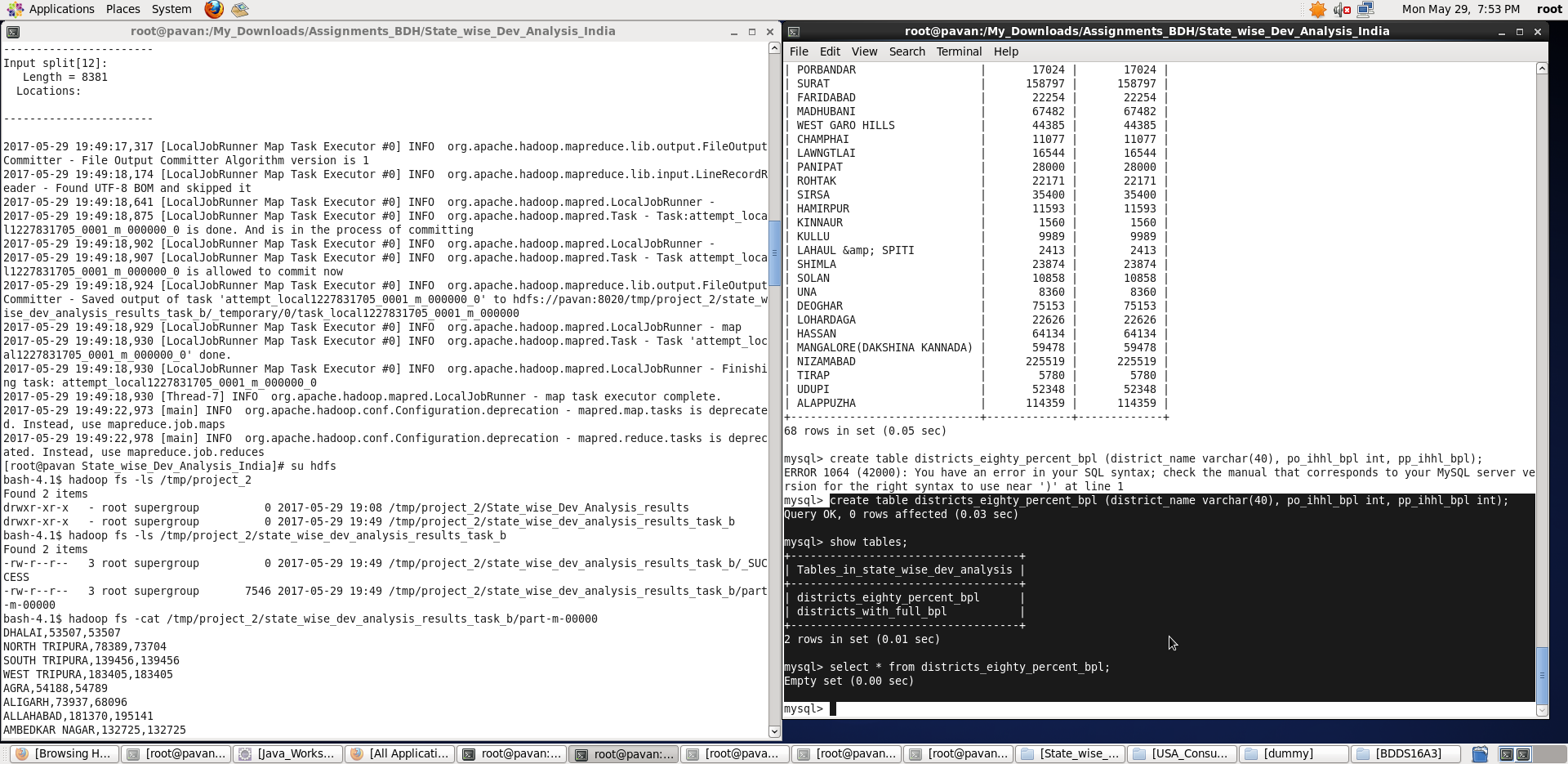
**Output stored in the format**: **<District\_Name>,<po\_ihhl\_bpl>,<pp\_ihhl\_bpl>**

**We can see the output stored in the specified HDFS directory as below:**

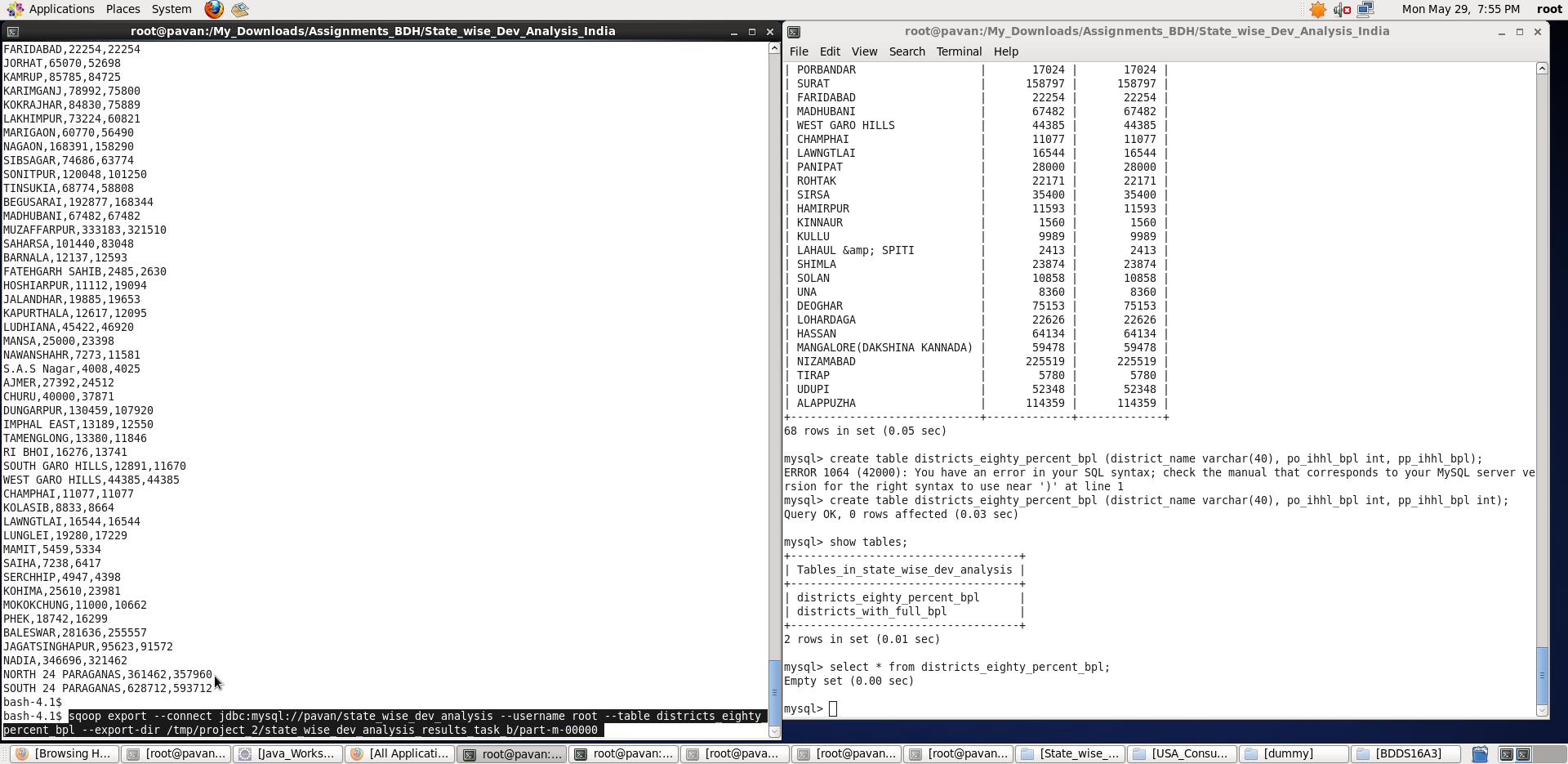


**Export the results to mysql using sqoop.**

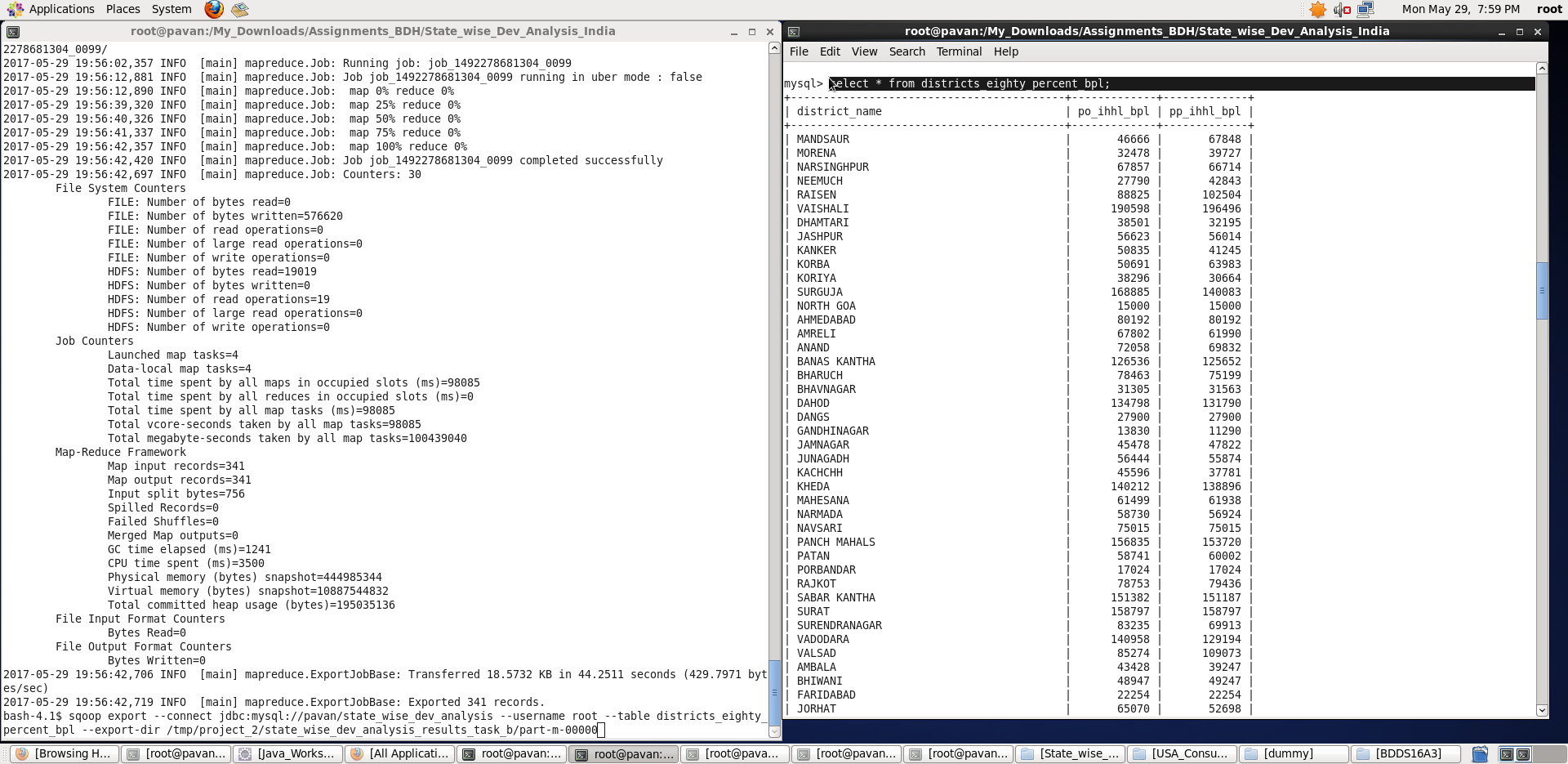
**For this, the table should be already present in mysql before we export the result from HDFS**

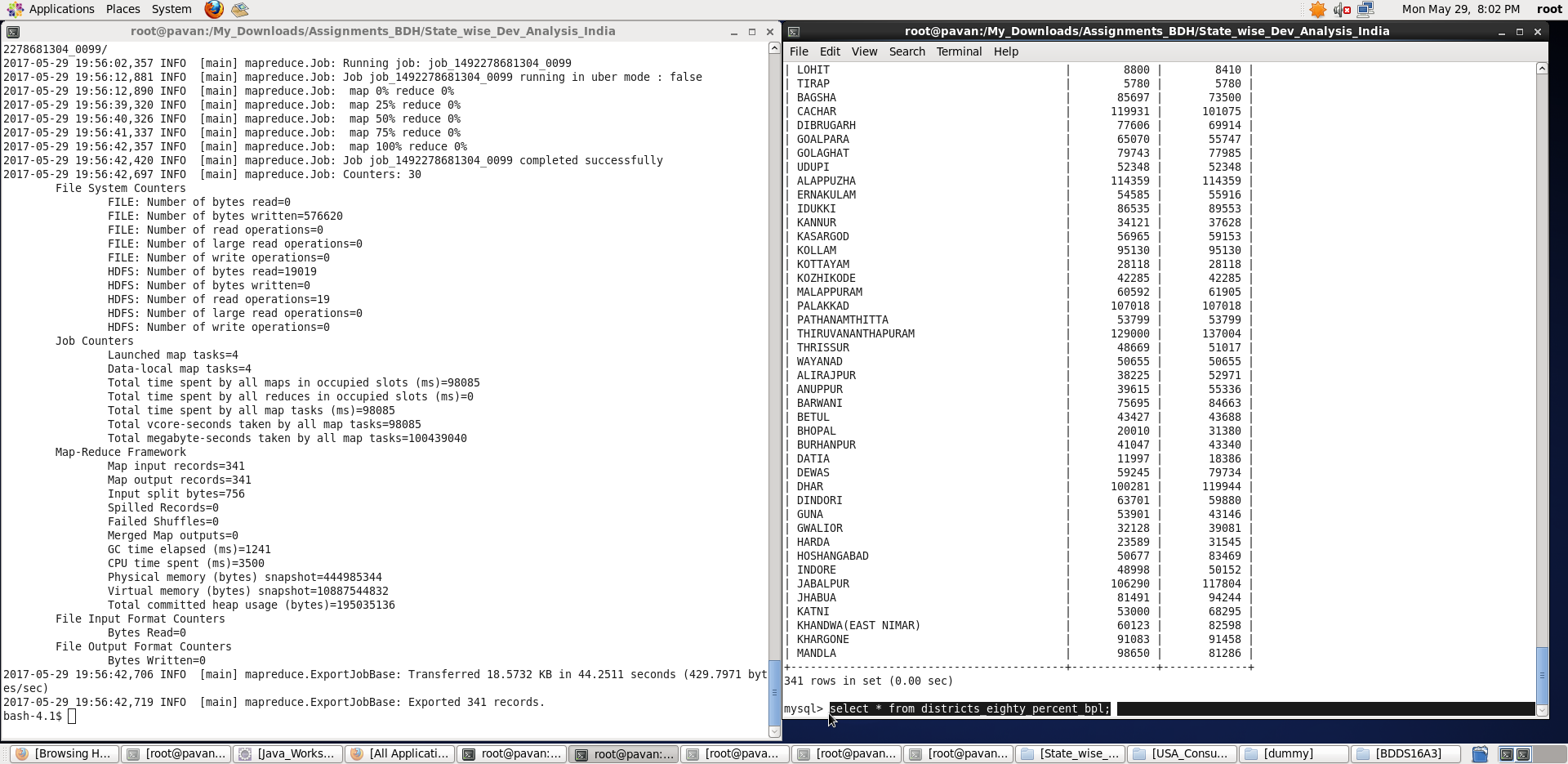


**Once, the table is created, use the sqoop export command as shown below to export the results into mysql.**



**Once, the export is done, we can see results getting stored in the districts\_with\_eighty\_percent\_bpl table inside state\_wise\_dev\_analysis database using “*select \* from districts\_with\_eighty\_percent\_bpl”* command.**



****

**Thus, we found the districts who have reached 80 percent objective in BPL cards and exported the results to mysql using sqoop.**

**Note: I have considered more than 80 percent objective achievement in BPL cards.**