**Problem Statement:**

* Exporting the Data from the Local File System to the HDFS using Flume
* Performing Analysis on the data (in xml form) using PIG to get results for the below problem statements:
* Find out the districts who achieved 100 percent objective in BPL cards

Export the results to MySQL using Sqoop

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

Export the results to MySQL using Sqoop.

**Dataset:**

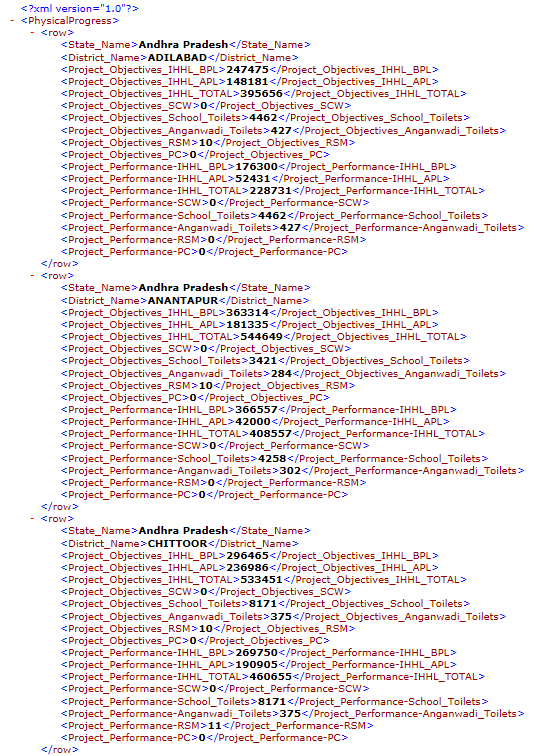
The dataset is an xml file that contains the State-Wise Development data for India

**Google Drive Link:**

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

**Screenshot:**

A sample view of the data in the xml file.



**Exporting the Data from the Local File System to the HDFS using Flume**

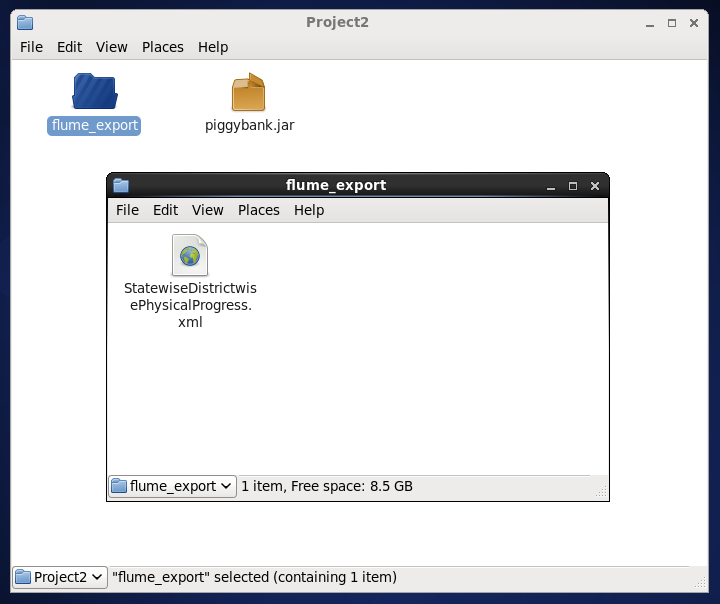
To perform this task we have to execute the following steps:

* Download Apache Flume for the Acadgild VM and extract it

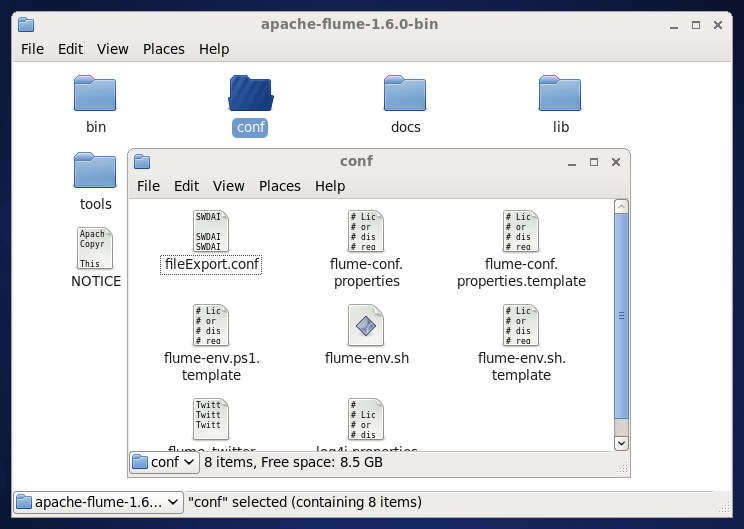
Update the location of Apache Flume in the .bashrc file

* Create the spool directory from where Flume will retrieve the data to be stored in the HDFS

Here my spool directory is flume\_export and my State and District Progression Log File for India is StatewiseDistrictwisePhysicalProgress.xml



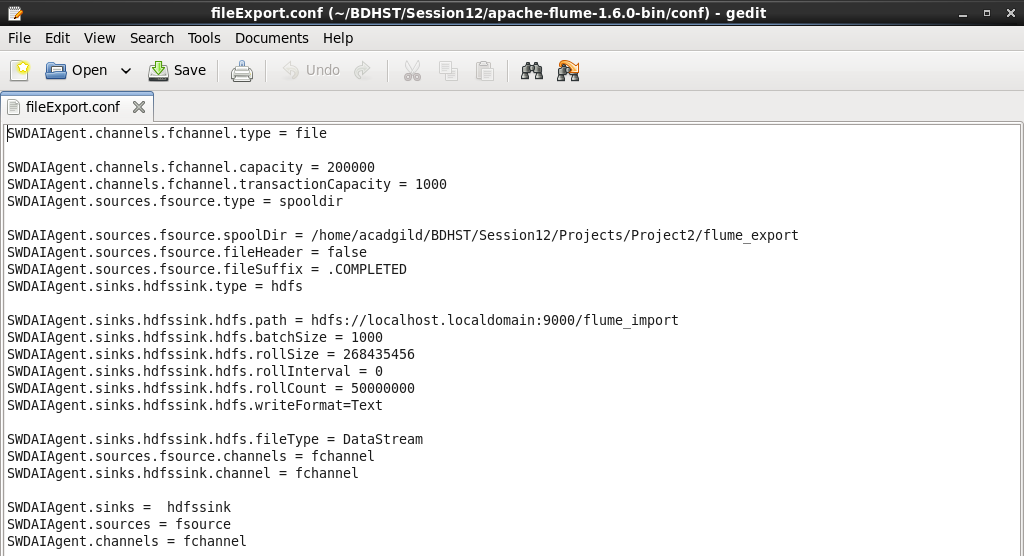
* Create the configuration document for the flume job. This will contain the necessary information for **fetching log files from spool directory** and **storing these files in the HDFS**



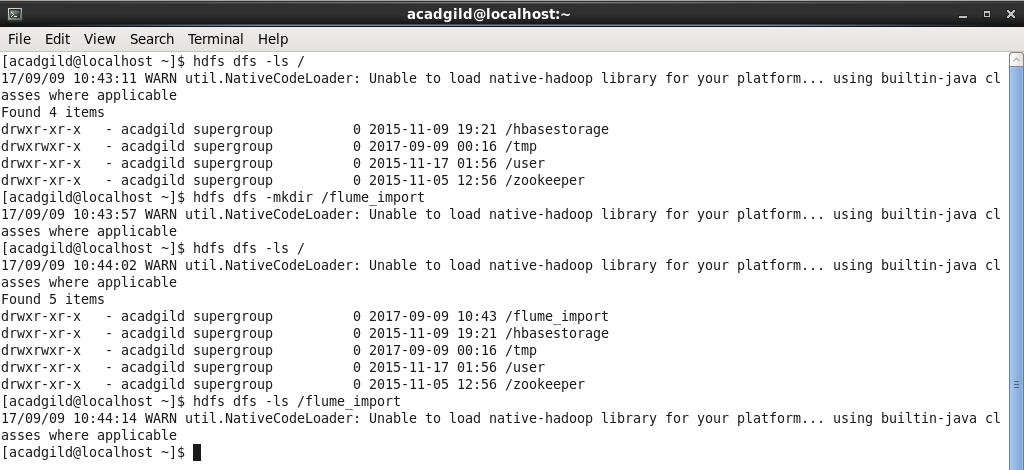
My configuration file fileExport.conf is stored in the **conf** directory of Apache Flume directory

Below is the configuration file fileExport.conf, some important configurations are:

* + Specifying the type of structure the file is coming in the channel: **file**
  + Specifying the capacity of the transmission channel
  + Specifying the type of source: **spool directory source**
  + Specifying the path of the spool directory
  + Specifying the suffix to be added to the name of the file in the spool directory
  + Specifying the path in the HDFS to store the data



* Create the folder flume\_import in the HDFS that will hold the data from Flume Agent/Job



* Execute the flume command that will create the flume job fetching data from the Local File System to the HDFS:

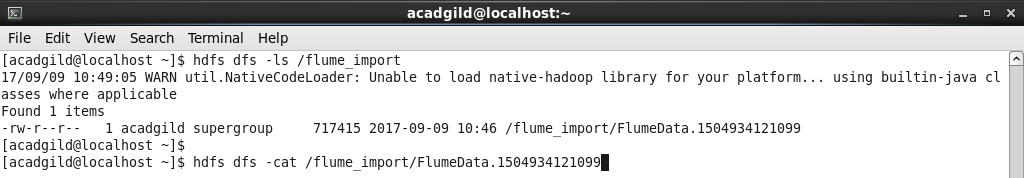
flume-ng agent –n <agentName> –f <path to fileExport.conf>

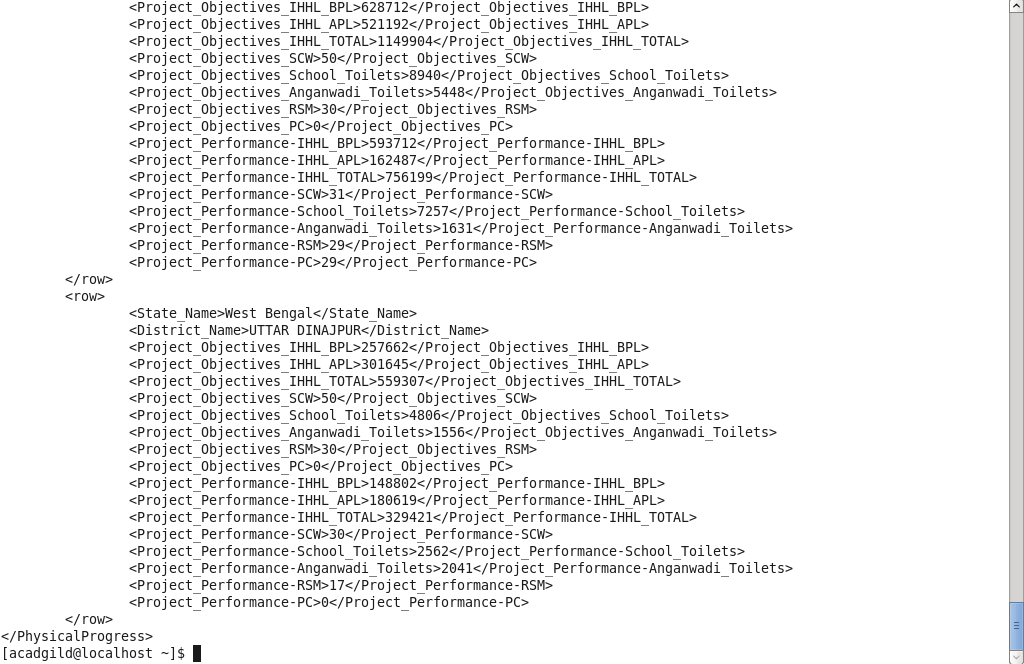




This will now check the spool directory flume\_export for the log file to export and then export/store it in the HDFS directory flume\_import as given in the configuration file.

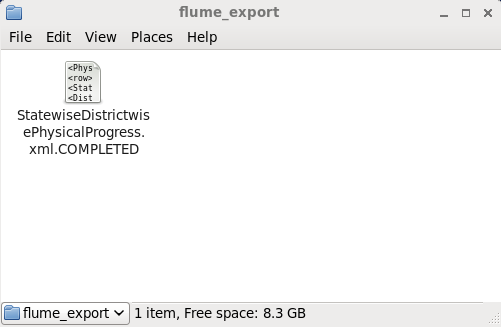
* Checking the HDFS import directory flume\_import to see if the data has been exported successfully





The xml file has been successfully exported

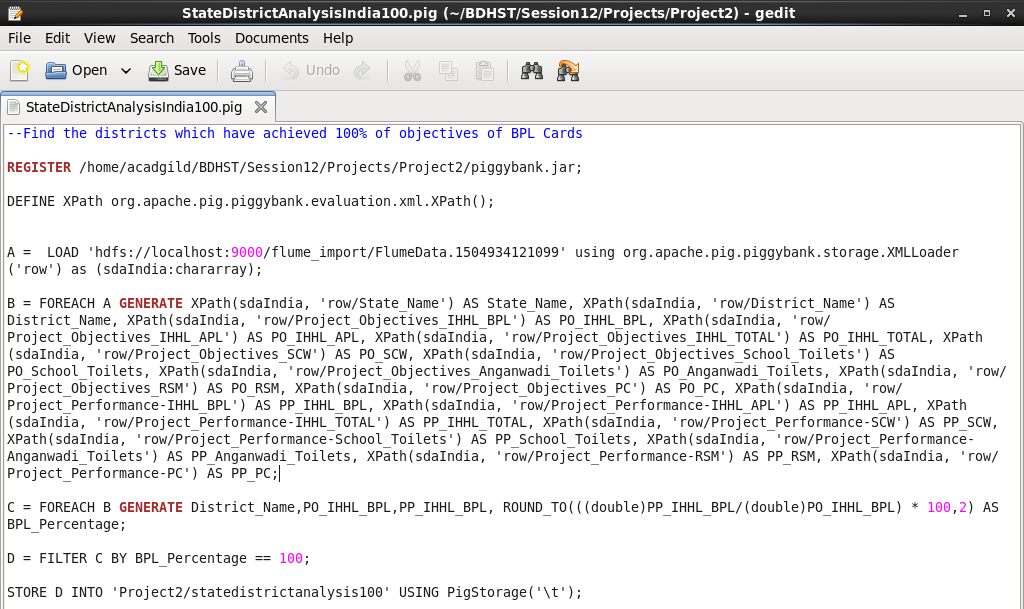
Also, below is the xml data file in the spool directory. As mentioned in the configuration file, the flume job has added the **COMPLETED** as suffix addition to the file name. This shows us that the file has been successfully read from the spool directory.



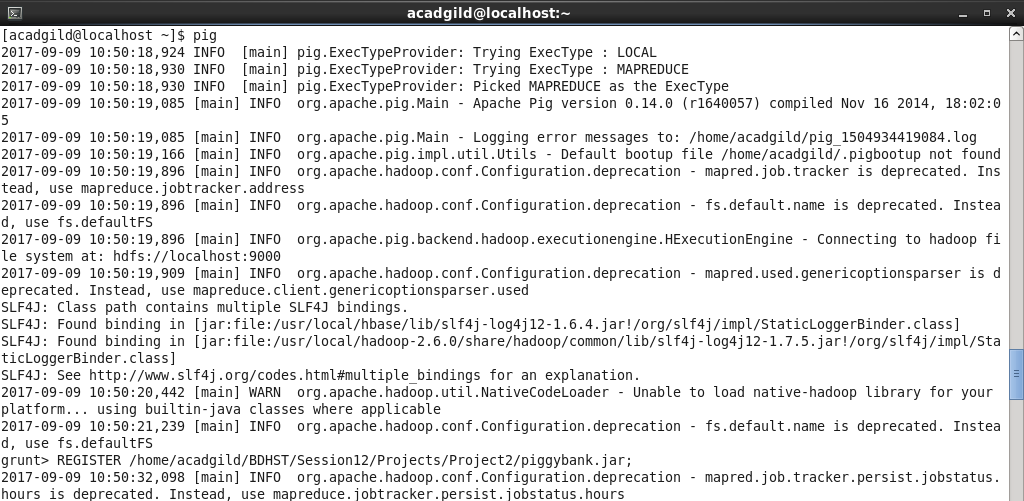
**Performing Analysis on the data (in xml form) using PIG**

**Find out the districts who achieved 100 percent objective in BPL cards. Export the results to MySQL using Sqoop**

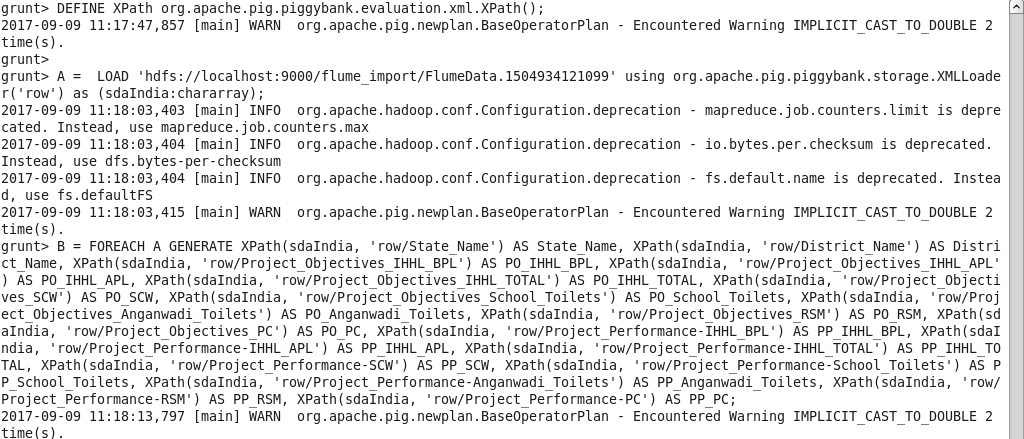
This is a summary of the commands used to execute the above problem statement



* Starting the Pig Shell using the command **pig** (not local so we can access the HDFS)
* Registering the **piggybank** jar that contains the executables for various pig functions. Ex: Parse XML (Used in this assignment)



* Defining the XML Parse function as **XPath** (name used to call the function)
* Loading the data in the HDFS (that was exported using Flume) and using the XML Loader function to load the data into the relation **A** with every starting tag ‘row’ as one line of type: chararray with the name **sdaIndia**
* Generating the rows (sdaIndia) in relation A by using the XML Parser **XPath.** Every tag under the main tag **row** will be separated by the tag name and given a pseudo name in the relation.

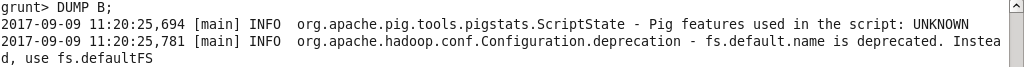


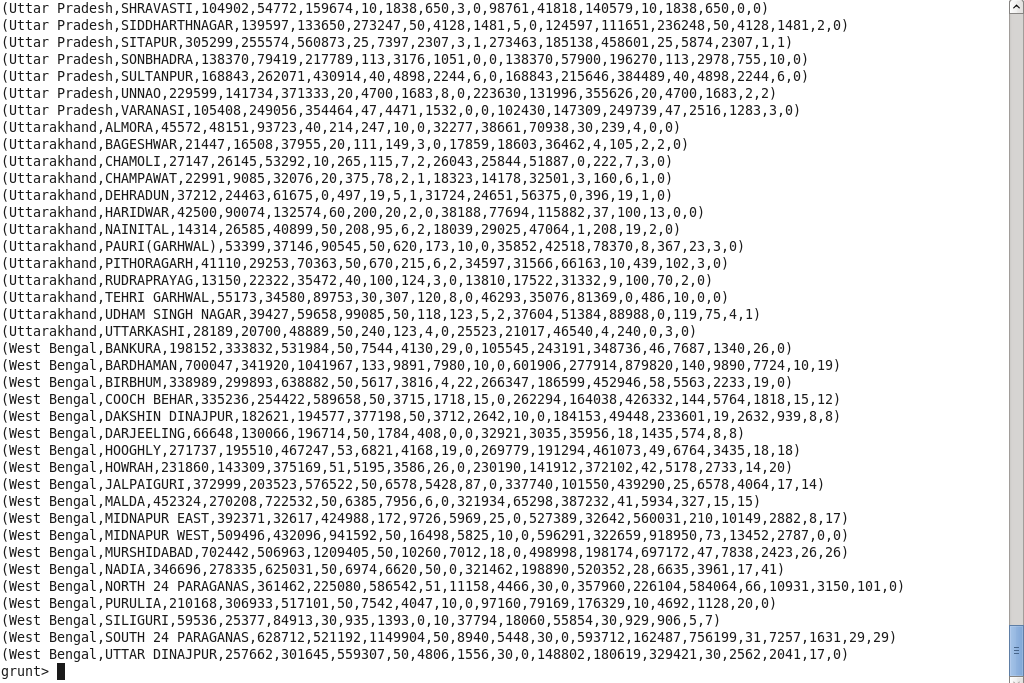
* Displaying the results of the Load statement

D:\MICHELLE\BIG DATA & HADOOP Development\Session 12 - PROJECT - 1 & INTRODUCTION TO SCALA - SESSION I\Projects\Screenshots\B11.png

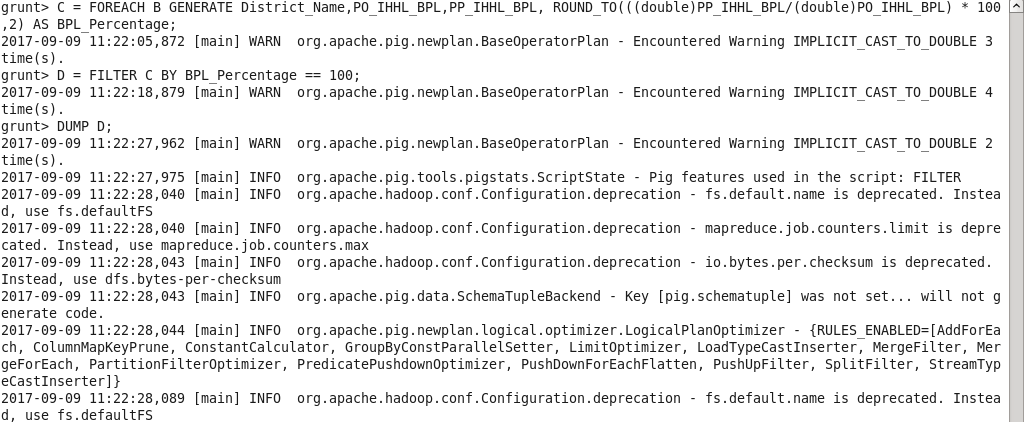


* Displaying the result of the Row Generating statement. All the data has been separated by tag name and formatted into a tuple of multiple fields.

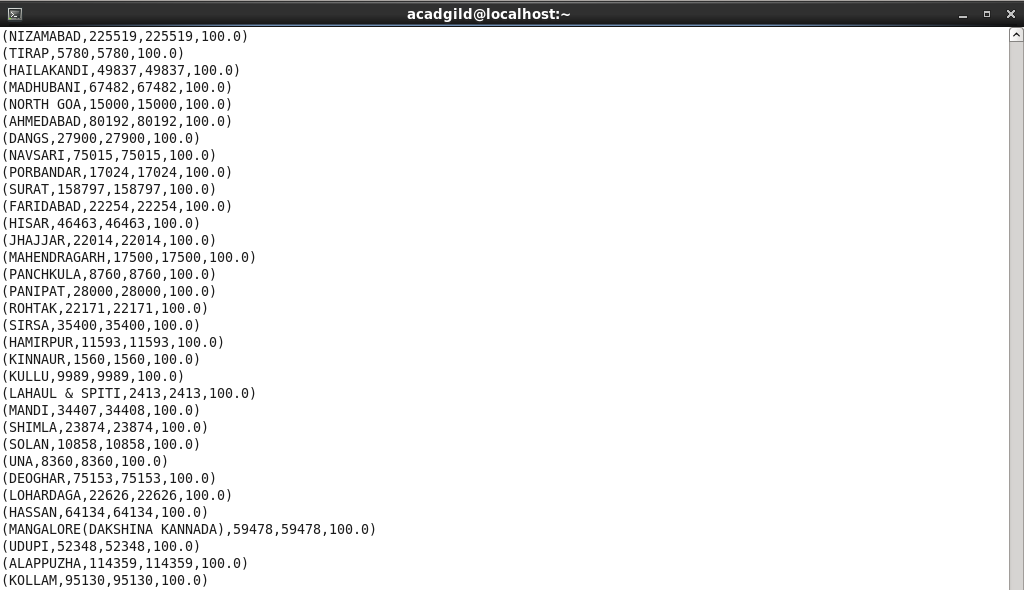


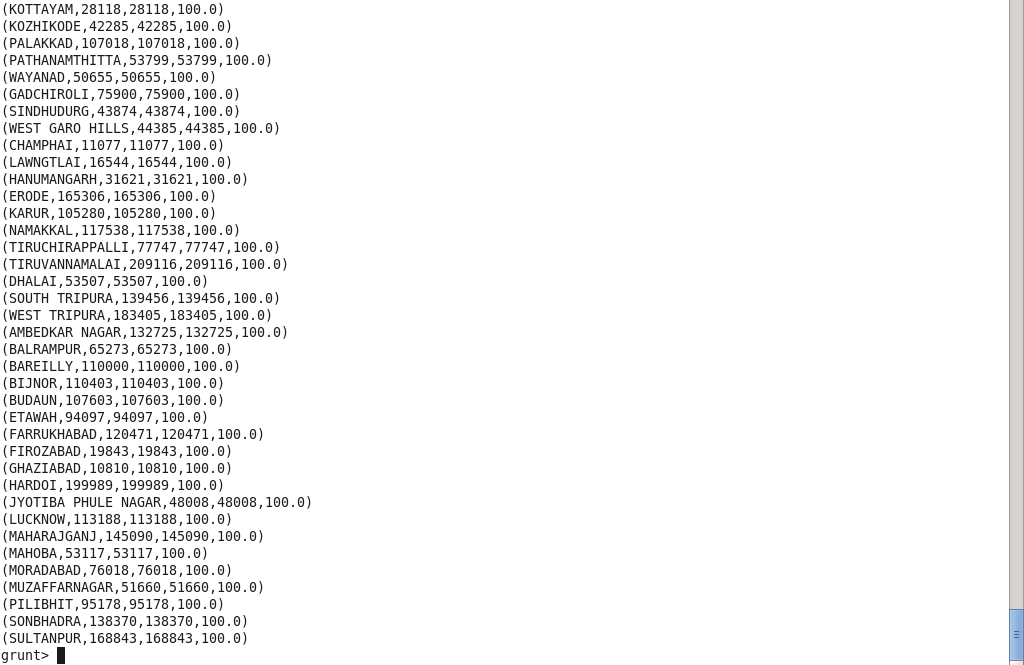


* Generating column names pertaining to **District** and **BPL information** and finding the **Percentage** of performance achieved for the objective that was set for BPL Cards in India.
* Filtering the above result for those records where 100% objective has been met and displaying the result.

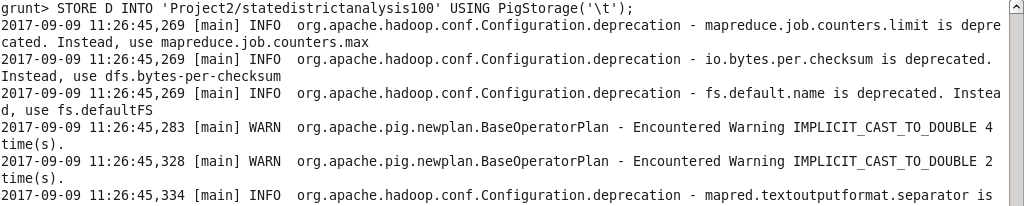


* The result of the above procedure:



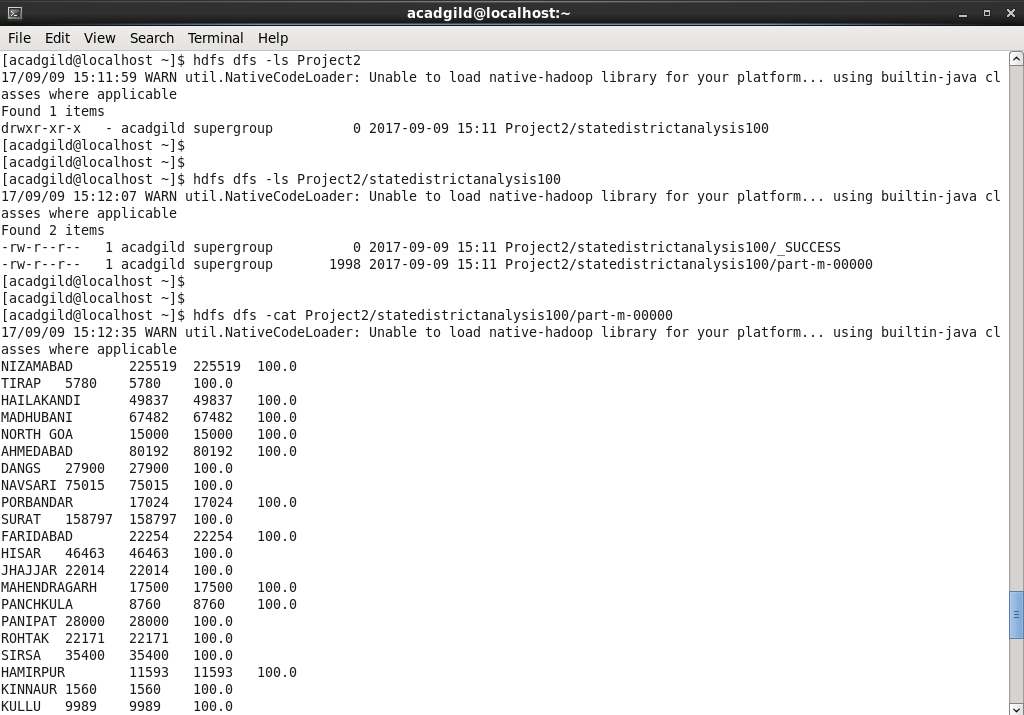


* Now we store the result in the HDFS for the **Sqoop job to export the data to a MySQL database**
* Storing the data in the HDFS under the path given below and separating the fields by tab space



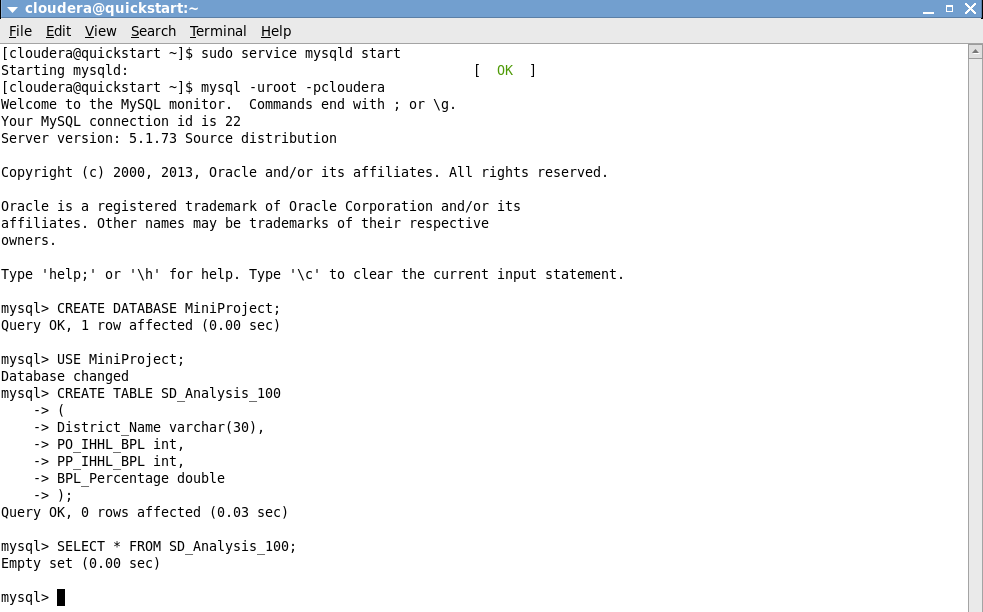
* To check if the file has been successfully stored in the HDFS, we check the output folder of its contents.

The data has been stored successfully as seen by the file named **part-m-00000** that hold the output of the MapReduce job

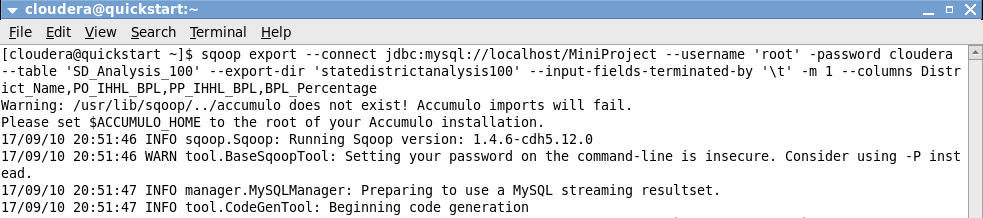


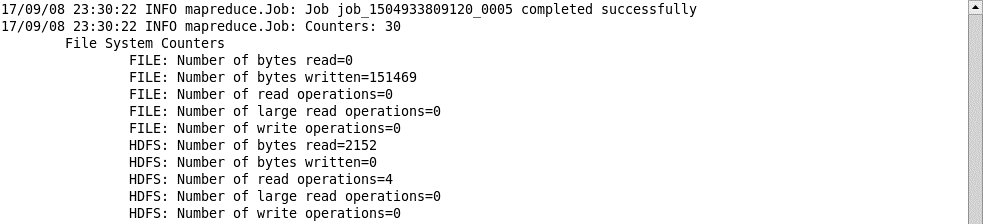
* Now we export the data in the HDFS to a Table in MySQL by the following steps:
  + Start the MySQL service and terminal and create the database and table to hold the data

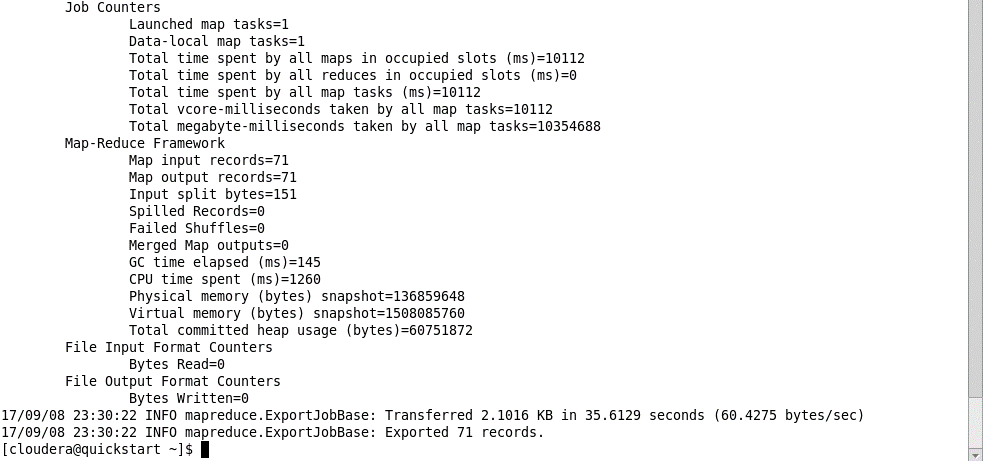
Here my table is named **SD\_Analysis\_100**



* + Using the Sqoop command given below:
    - Specifying the name of the database to hold the data
    - Specifying the password of the VM (Can also be manually entered or got from a password file)
    - Specifying the name of the table to hold the data
    - Specifying the directory in the HDFS that holds the data
    - Specifying how the fields are terminated
    - Specifying the number of MapReduce jobs :1
    - Specifying the column names to import to the MySQL table



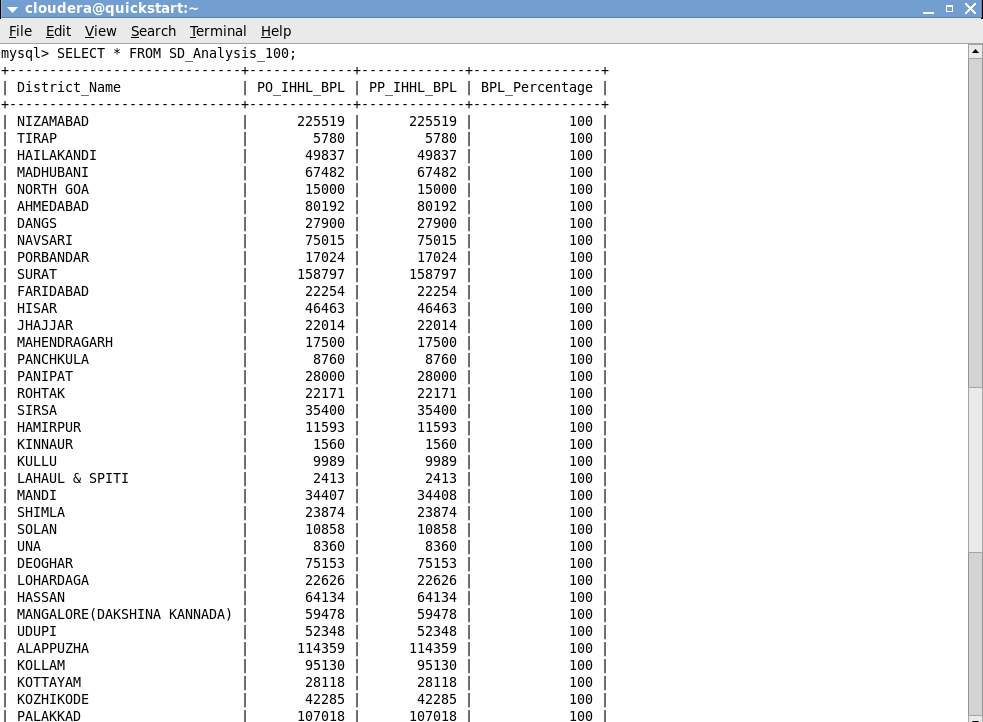


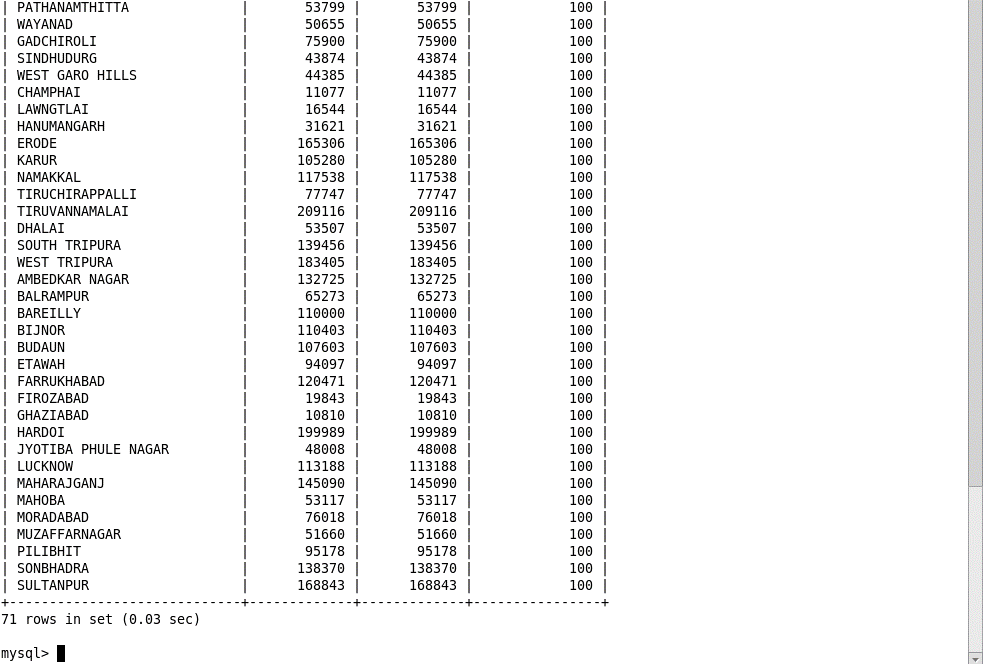


The file has been successfully written to the MySQL table **SD\_Analysis\_100**

**OUTPUT:**

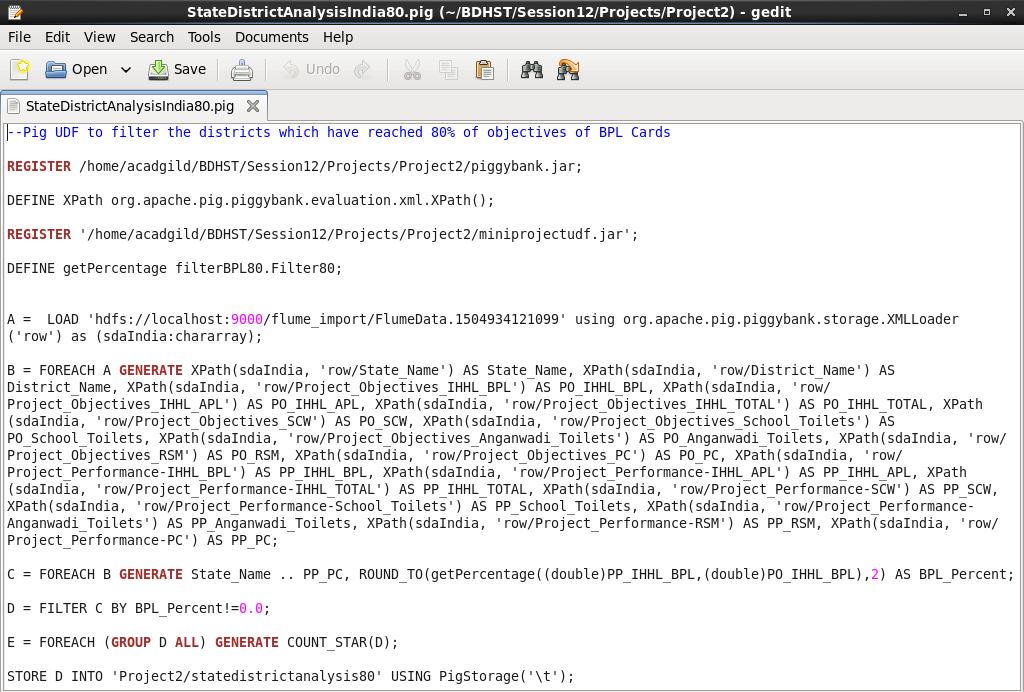
* To check the contents of the MySQL table **SD\_Analysis\_100** use the **SELECT \*** command





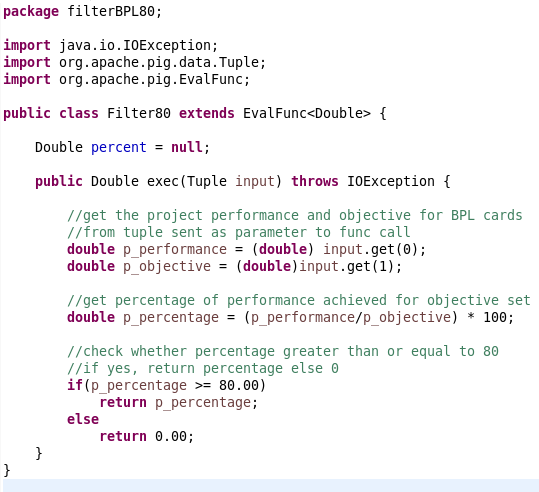
**Write a Pig UDF to filter the districts which have reached 80% of objectives of BPL cards. Export the results to MySQL using Sqoop.**

* To filter the districts that have reached 80% of their objectives in BPL Cards, I have created a Pig Script(with commands similar to the problem before) and executed it via the pig MapReduce shell

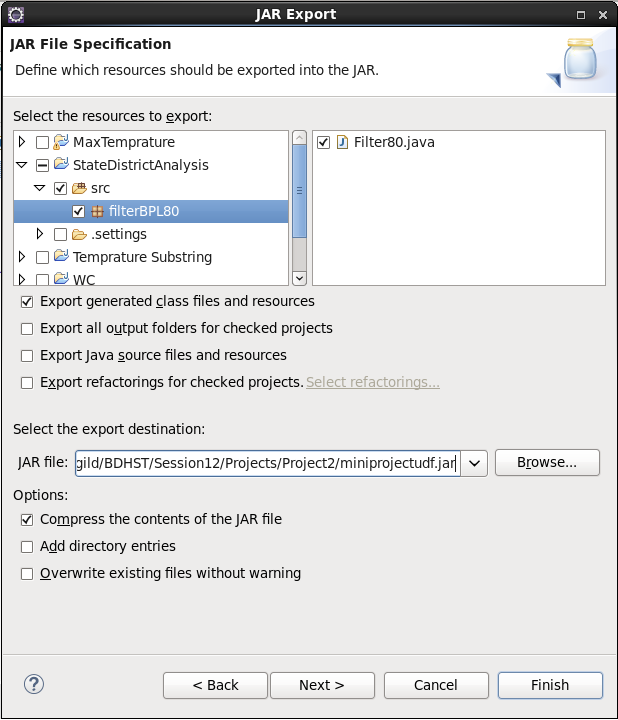


The steps followed are explained as below:

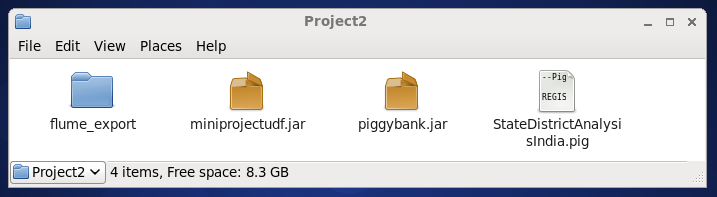
* Registering the piggybank jar that contains the executables for various pig functions. Ex: Parse XML (Used in this assignment)
* Defining the XML Parse function as **XPath** (name used to call the function)
* Registering the Pig UDF miniprojectudf created to filter the districts which have reached 80% of objectives of BPL cards. (Written in Java)
* Defining getPercentage as the function to be used to execute the UDF in package **filterBPL80** and class **Filter80**
* Loading the data in the HDFS (that was exported using Flume) and using the XML Loader function to load the data into the relation **A** with every starting tag ‘row’ as one line of type: chararray with the name **sdaIndia**
* Generating the rows (sdaIndia) in relation A by using the XML Parser **XPath.** Every tag under the main tag **row** will be separated by the tag name and given a pseudo name in the relation.
* Generating all column and finding the **Percentage** of performance achieved, for the objective that was set for BPL Cards in India, by using a Pig UDF written in java and exported as a jar as below:
* Below is an image of the Pig UDF



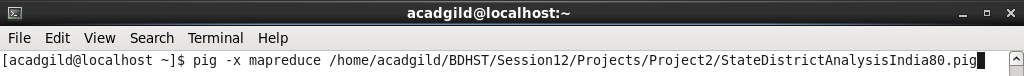
* The UDF exported as a jar

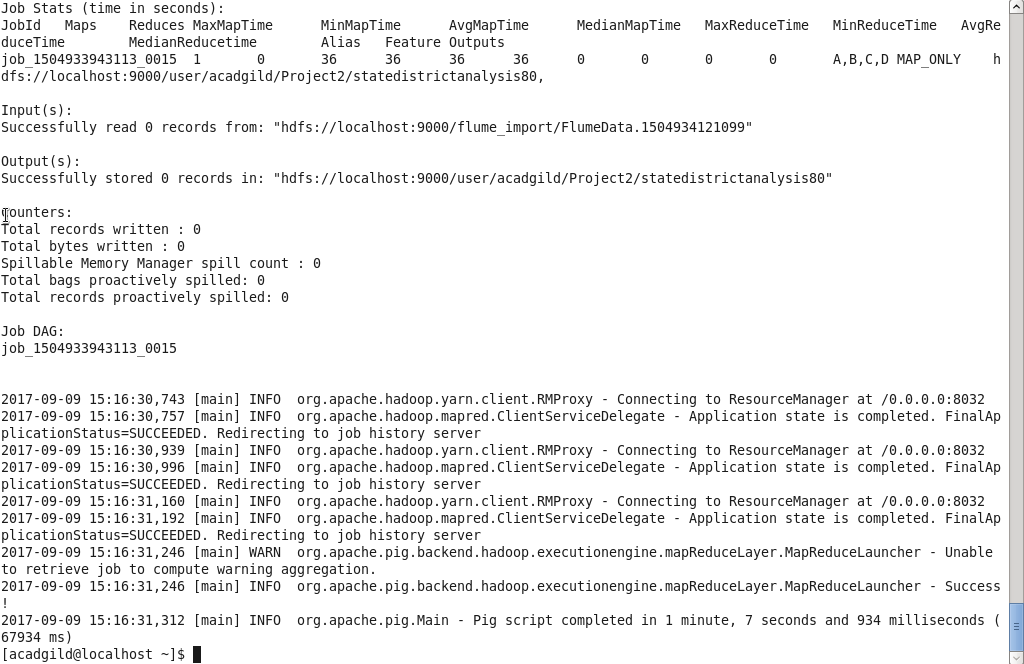


* The UDF in the directory from where it is accessed



* Filtering the above result for those records where percentage is 0.0% (The records that do not meet the 80% objective). Therefore giving us the records that have received 80% and above in BPL cards
* Getting the count of the filtered records
* Storing the results, i.e. the filter records into a directory in the HDFS and separating the fields by tab space
* Executing the Pig Script in MapReduce mode (can access HDFS) as below:

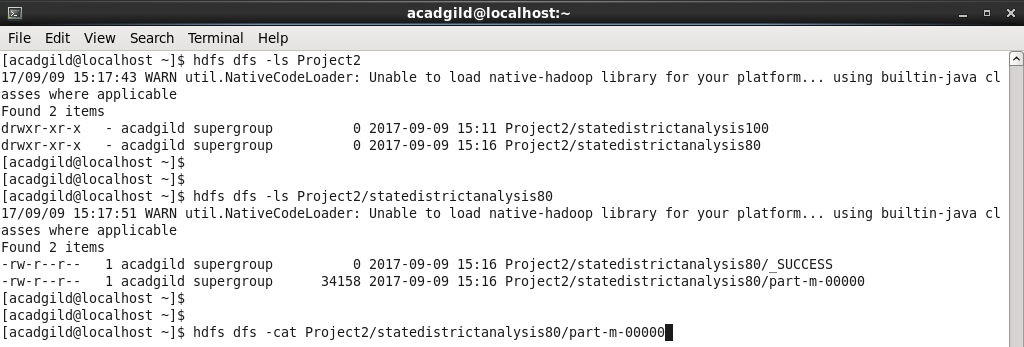


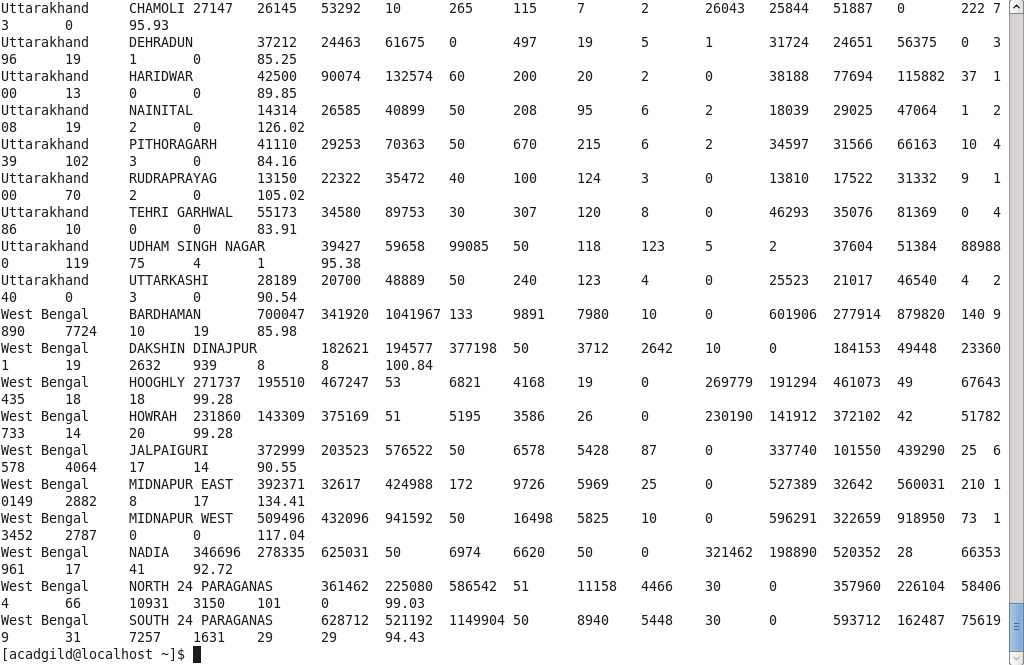


The execution is successful.

* Checking the contents of the folder statedistrictanalysis80 in HDFS that contains the filtered data

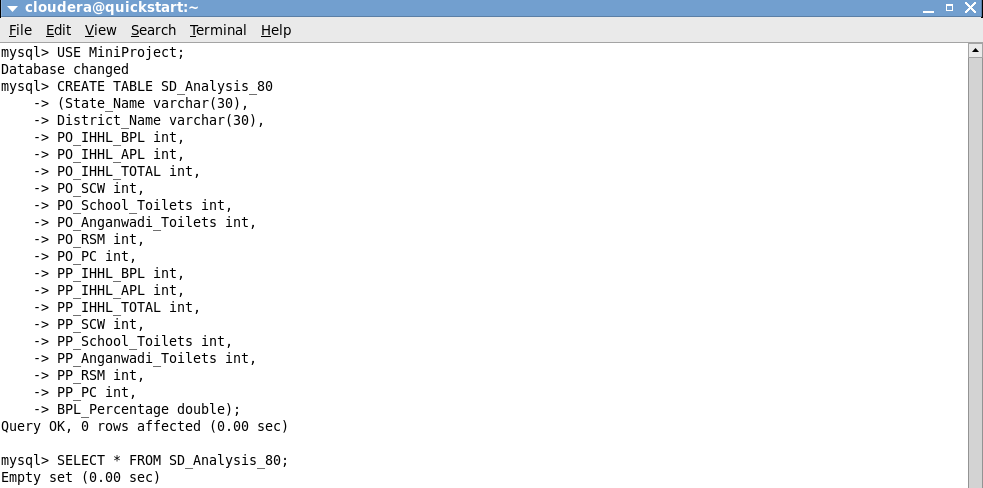
The data has been stored successfully as seen by the file named **part-m-00000** that hold the output of the MapReduce job.



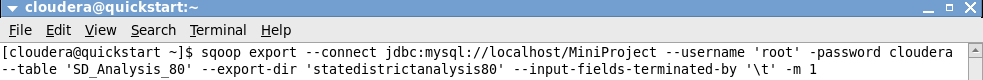


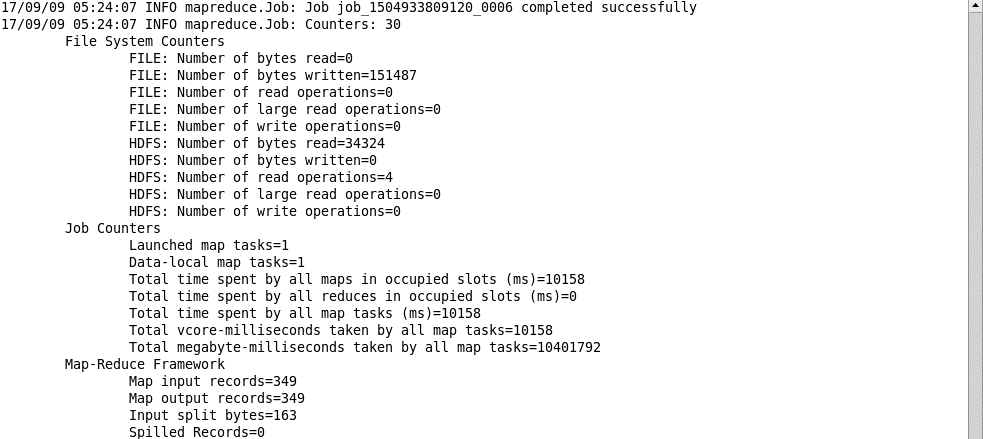
* Now we export the data in the HDFS to a Table in MySQL by the following steps:
  + Start the MySQL service and terminal and create the database and table to hold the data

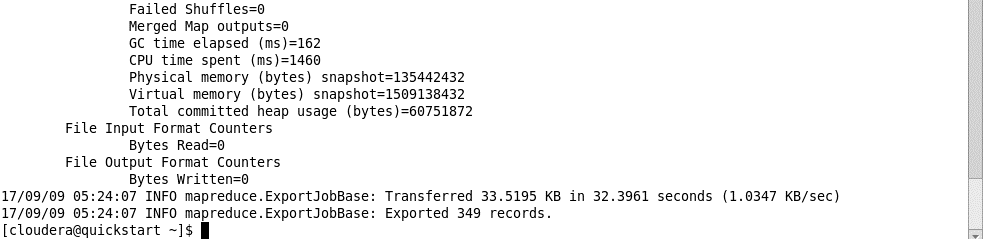
Here my table is named **SD\_Analysis\_80**



* + Using the Sqoop command given below:
    - Specifying the name of the database to hold the data
    - Specifying the password of the VM (Can also be manually entered or got from a password file)
    - Specifying the name of the table to hold the data
    - Specifying the directory in the HDFS that holds the data
    - Specifying how the fields are terminated (tab separated)
    - Specifying the number of MapReduce jobs :1
    - Specifying the column names to import to the MySQL table (Only some of all the columns that are present in the HDFS are exported)



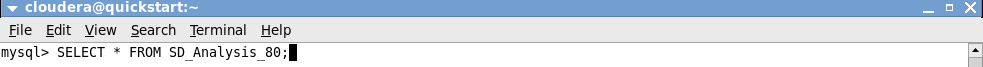


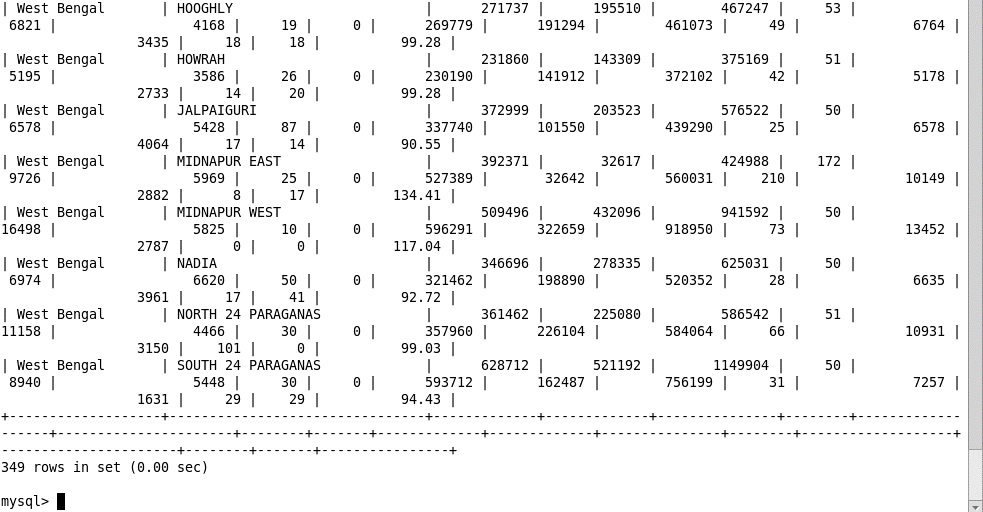


The file has been successfully written to the MySQL table **SD\_Analysis\_80**

**OUTPUT:**

* To check the contents of the MySQL table **SD\_Analysis\_80** use the **SELECT \*** command





* Using the below command you can check for specific columns in the table

