

College SCOREBoard

**Engineering of Big-Data Systems**

**Summer 2020**

**INFO 7250 CRN 41798**

**Professor: Yusuf Ozbek**

**Suryanarayana Malladi**

**August 10, 2020**

**Summary:**

The dataset is from US government’s Department of Education. It is called College Scorecard.

College Scorecard contains data about institutions and their field of study.

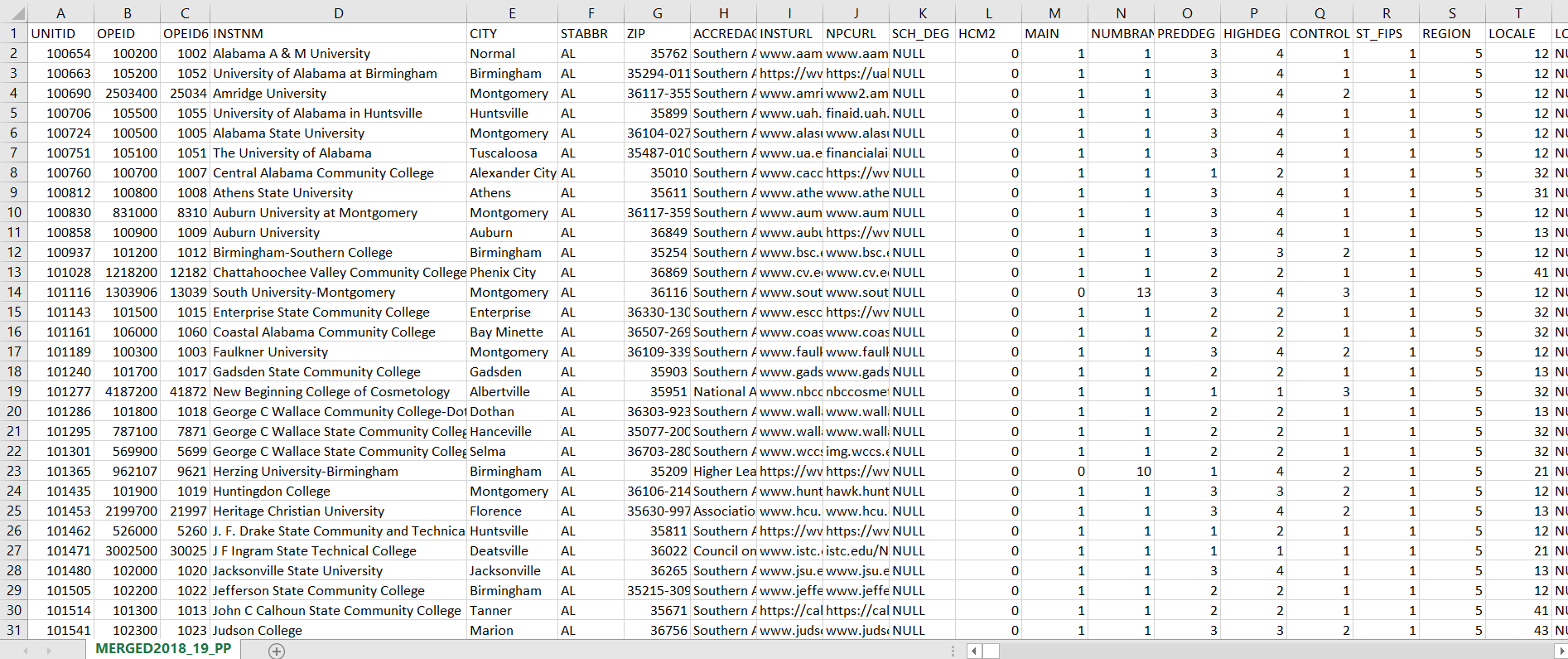
There are two sets of csv files:

* The first set focuses on high level institute data - almost 2000 rows of data
* The second set focuses on field of study for each institute - 18 rows of data

The data is from 1996 to 2019.

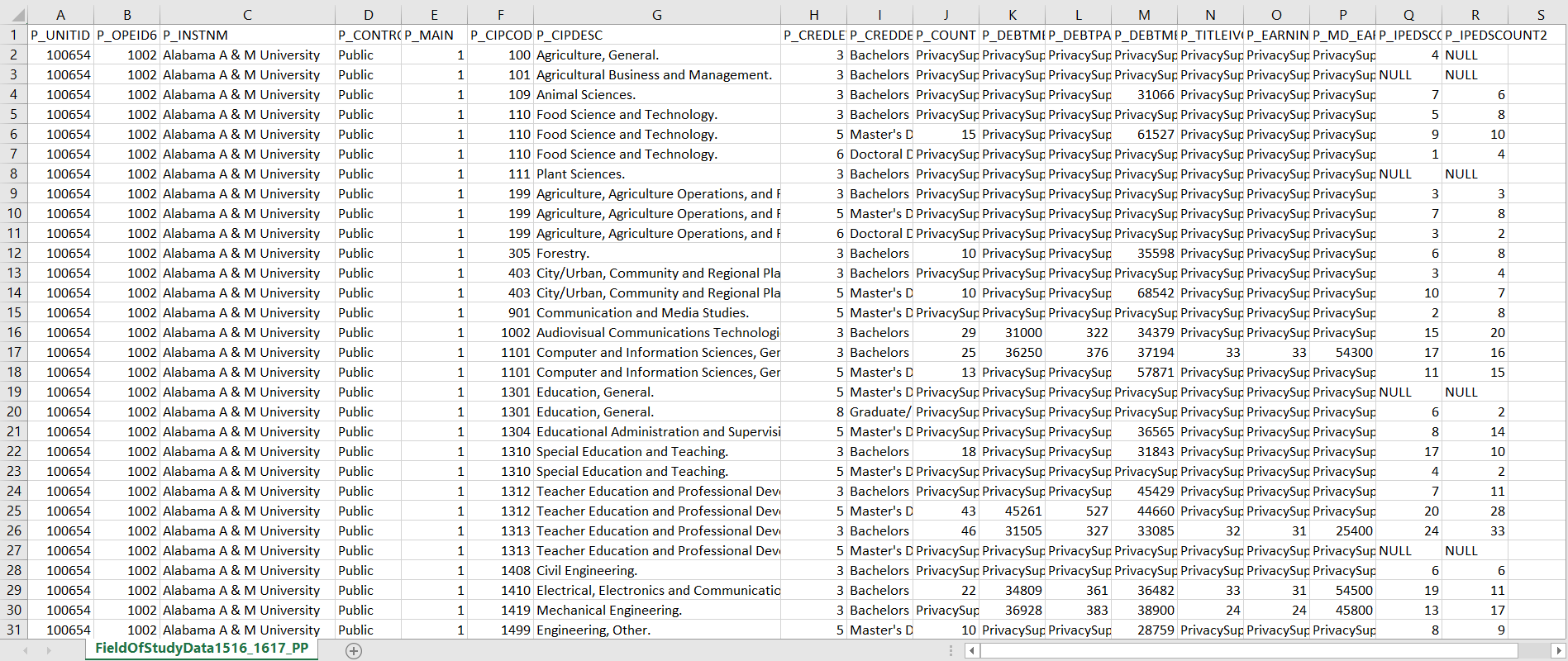
The attributes used from MERGED2018\_19\_PP.csv are

1. ZIP



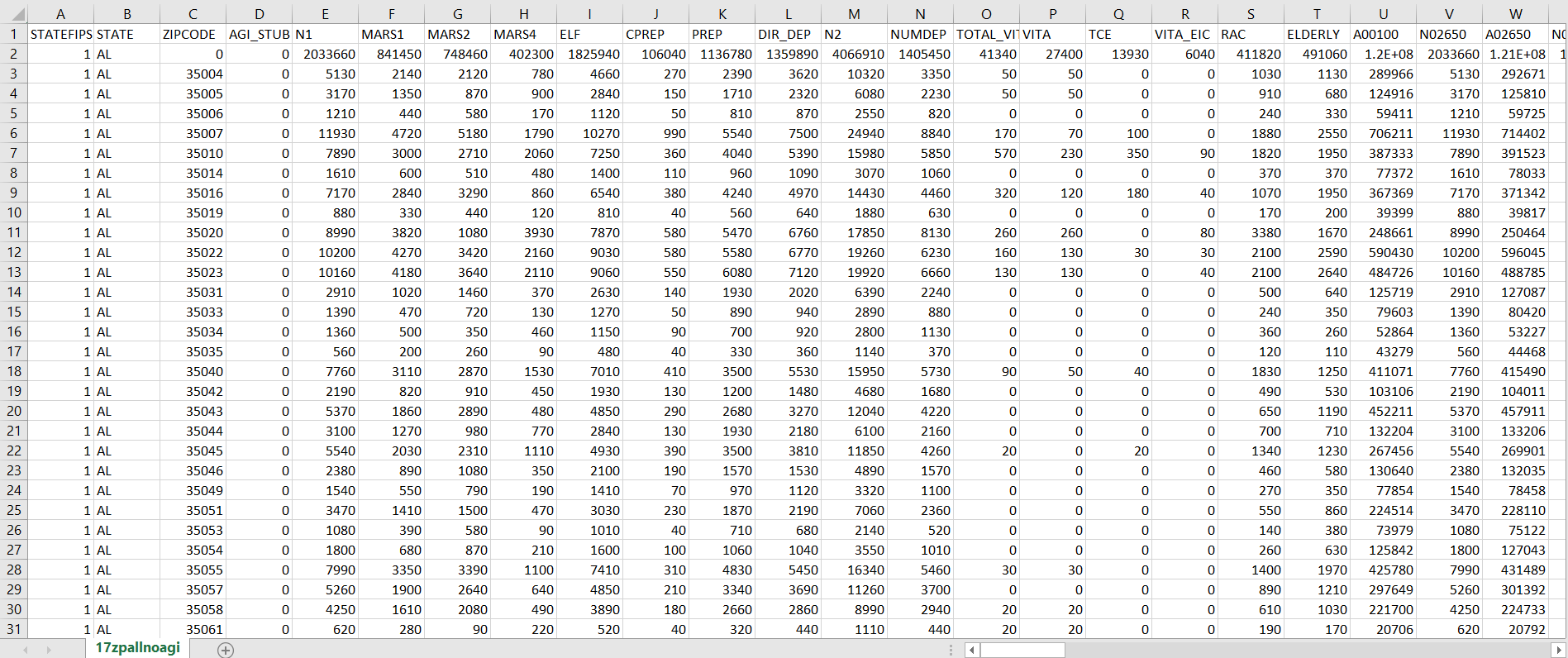
The attributes used from FieldOfStudyData1516\_1617\_PP are





The attributes used from IRS data are

1. ZIPCODE
2. A00100 – Aggregate Gross Income



**Summary of analysis conducted:**

|  |  |  |
| --- | --- | --- |
| S.No | Description | Implementation Details |
| 1 | Number of schools in a zip code (Descending order) | Mongo Import utility |
| 2 | Find Pearson’s coefficient of correlation between the number of schools in a zip code and wealth of the residents | Pearson Coefficient Algorithm, MR Chaining, Custom Writable Class, Multiple Maps in Same MR Job, SequenceFileInputFormat |
| 3 | Find number of students enrolled in each state and display the state in descending order of number of students | Pseudo Secondary Sorting, Clean up function in Reduce |
| 4 | Find minimum, maximum, average, standard deviation of SAT scores for all schools by state | Numerical Summarization |
| 5 | Find all degrees offered by an institute | Pig, Replicated Join |
| 6 | Find top 2 selective colleges by each state | Pig, Pig Latin operators |
| 7 |  |  |
| 8 |  |  |
| 9 |  |  |
| 10 |  |  |

**Getting Started**

Starting Hadoop

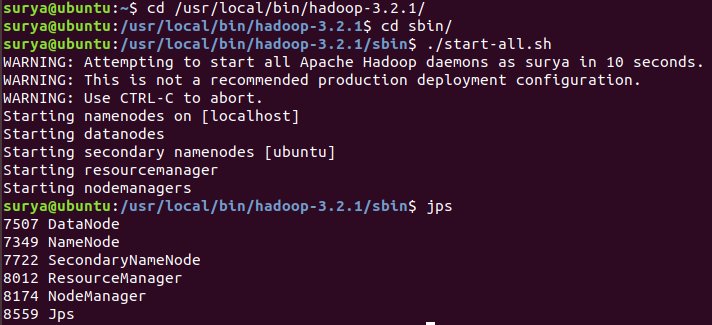
1. Navigate to directory where Hadoop is installed

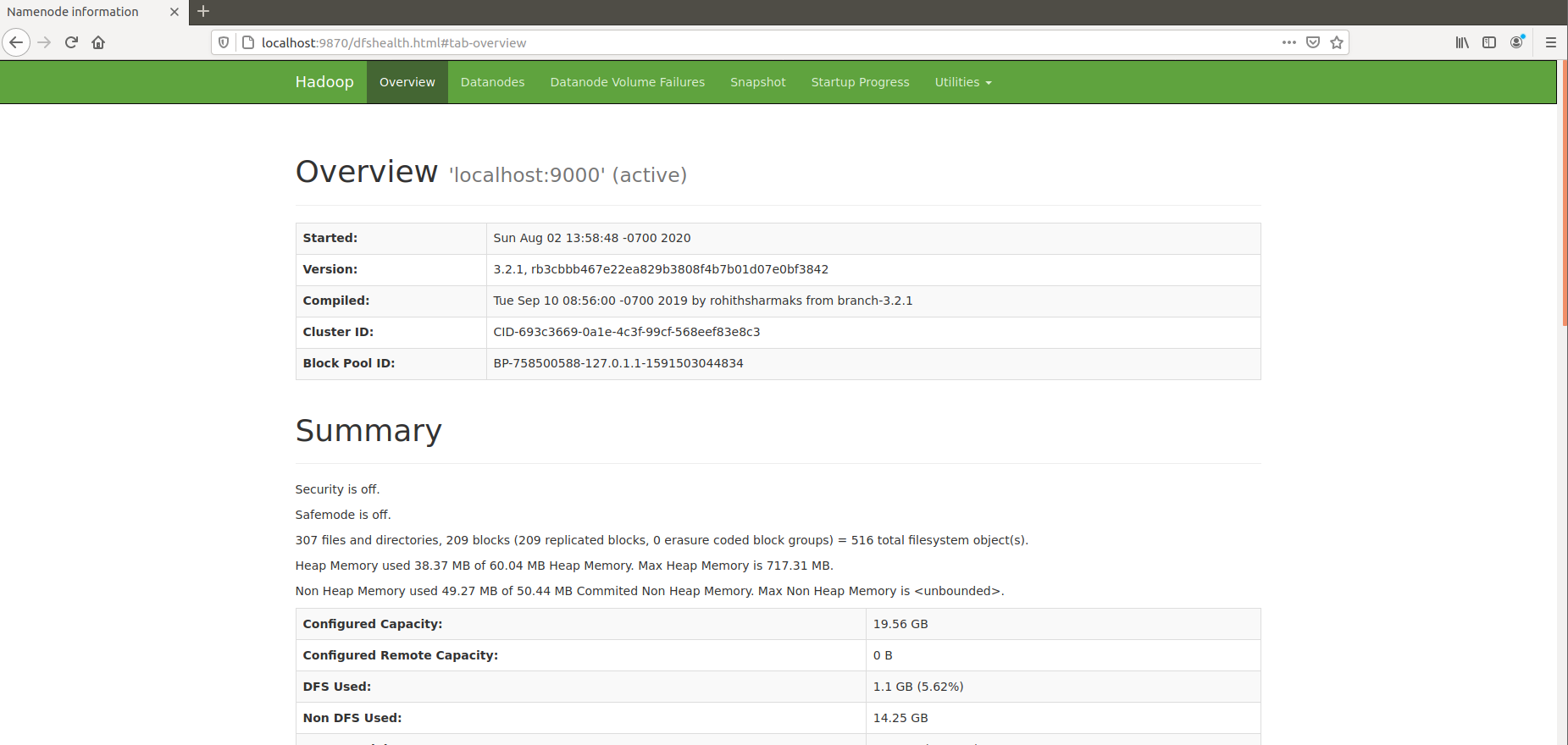
**cd /usr/local/bin/hadoop-3.2.1**

1. Enter into the sbin directory where all hadoop shell scripts are present

**cd /sbin**

1. Start the hadoop using the command **./start-all.sh**
2. Check if hadoop has started using the command **jps**





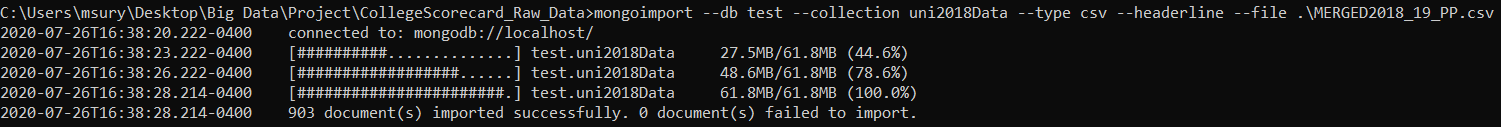
**Importing data into HDFS**

**./hadoop fs -copyFromLocal '/home/surya/Desktop/CollegeScorecard\_Raw\_Data/MERGED2017\_18\_PP.csv' /project**

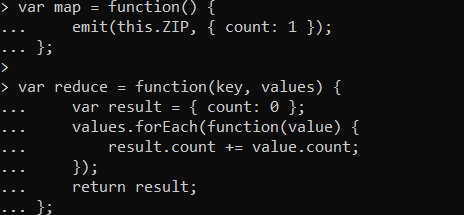
**./hadoop fs -copyFromLocal '/home/surya/Desktop/17zpallnoagi.csv' /project**

**Analysis # 01 : Number of schools in a zip code (Descending order)**

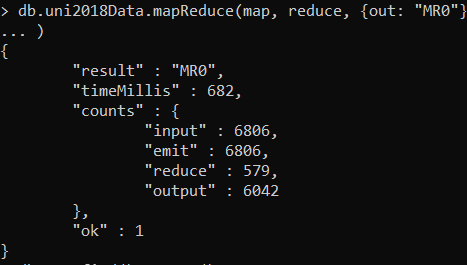
1. Import data from csv file into the MongoDB database.



2. Define map, reduce functions

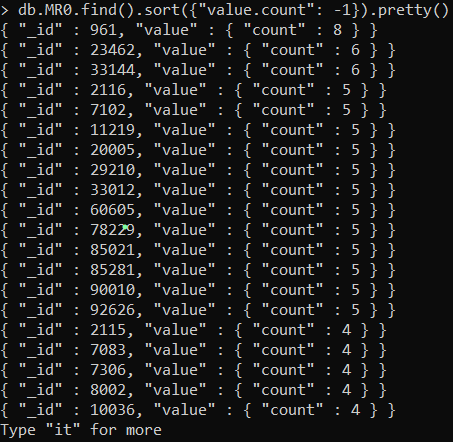


3. Call the mapReduce function on the collection and save the result into another collection



4. Check the MR0 collection to check the result and display in descending order of counts.

Zip code 961 has the highest number of educational institutes



**Analysis # 02: Find Pearson’s coefficient of correlation between the number of schools in a zip code and wealth of the residents**

[Pearson coefficient](https://en.wikipedia.org/wiki/Pearson_correlation_coefficient) also called as correlation coefficient, shows how two sets of data are related to each other. The coefficient will range from +1 to -1.

A coefficient of +1 shows that there is a linear positive coefficient. That means as one set of data increases the other also increases.

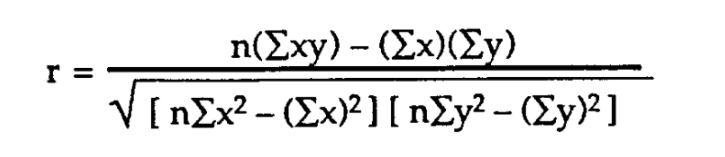
A coefficient of -1 shows that there is a linear negative coefficient. That means as one set of data increases the other decreases.

A coefficient of 0 means that there is no relation between the two data sets.

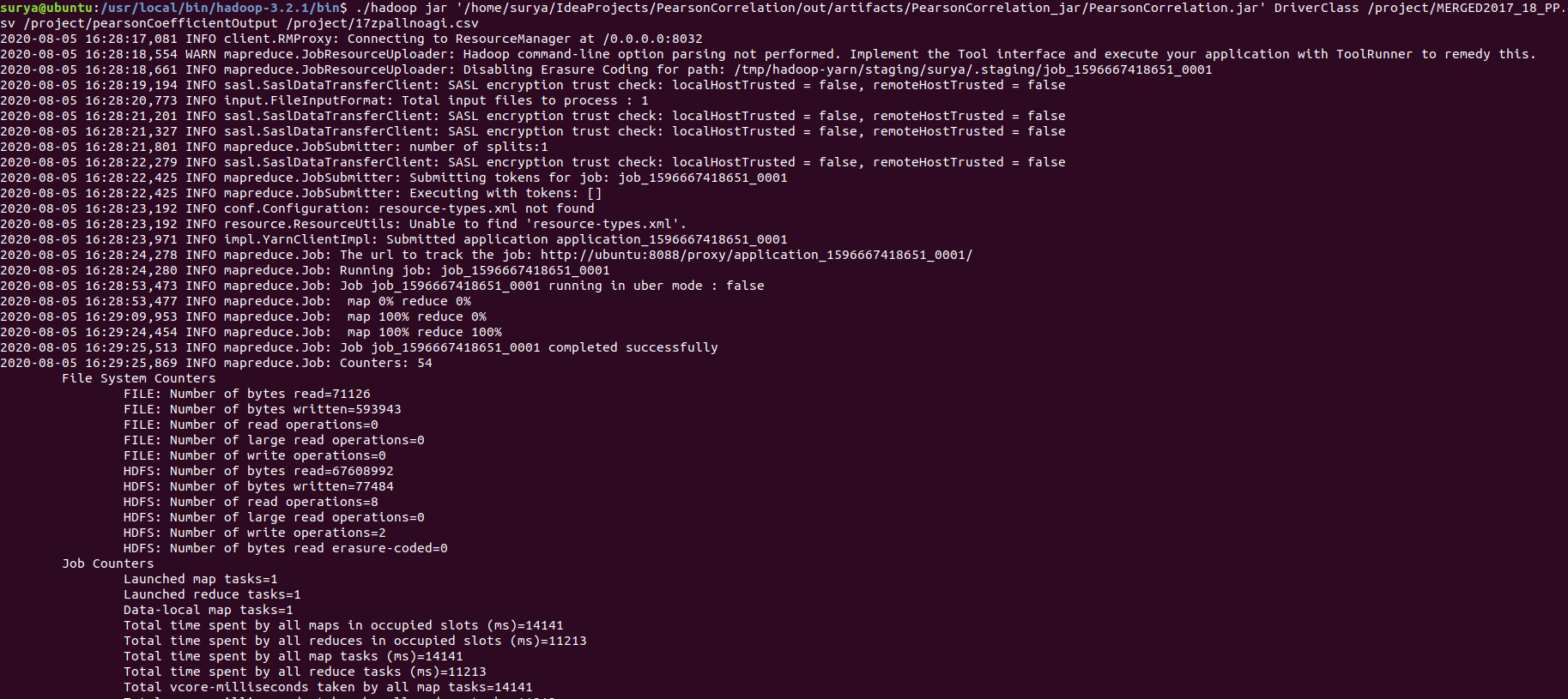
Three map reduce jobs were chained to calculate the correlation coefficient.

1. Calculated the number of universities in every zip code.
2. Calculated the aggregated gross income from each zip code.
3. The outputs of the first two MR jobs are joined and the Pearson correlation coefficient is calculated.

The formula is as follows:

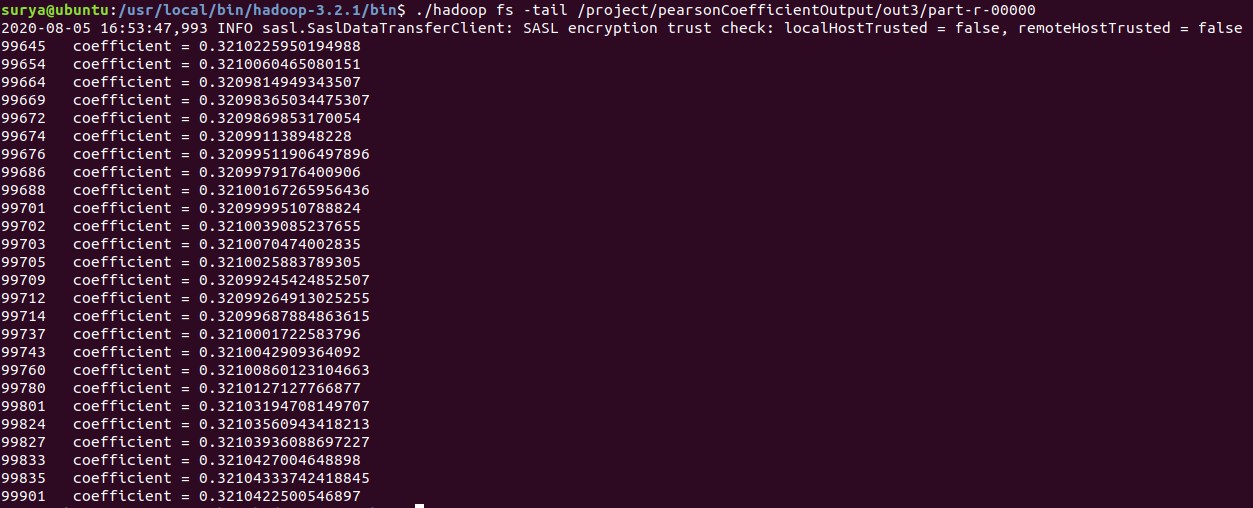


**./hadoop jar '/home/surya/IdeaProjects/PearsonCorrelation/out/artifacts/PearsonCorrelation\_jar/PearsonCorrelation.jar' DriverClass /project/MERGED2017\_18\_PP.csv /project/pearsonCoefficientOutput /project/17zpallnoagi.csv**



The output of the Map Reduce jobs can be seen with the following command

**./hadoop fs -tail /project/pearsonCoefficientOutput/out3/part-r-00000**

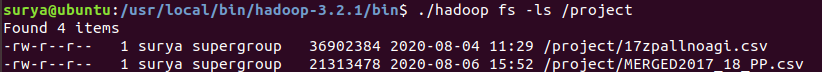


The correlation coefficient is 0.321, which means between the number of schools to the wealth of residents in a zip code, the relationship is moderately strong.

**Analysis # 03: Find number of students enrolled in each state and display the state in descending order of number of students**

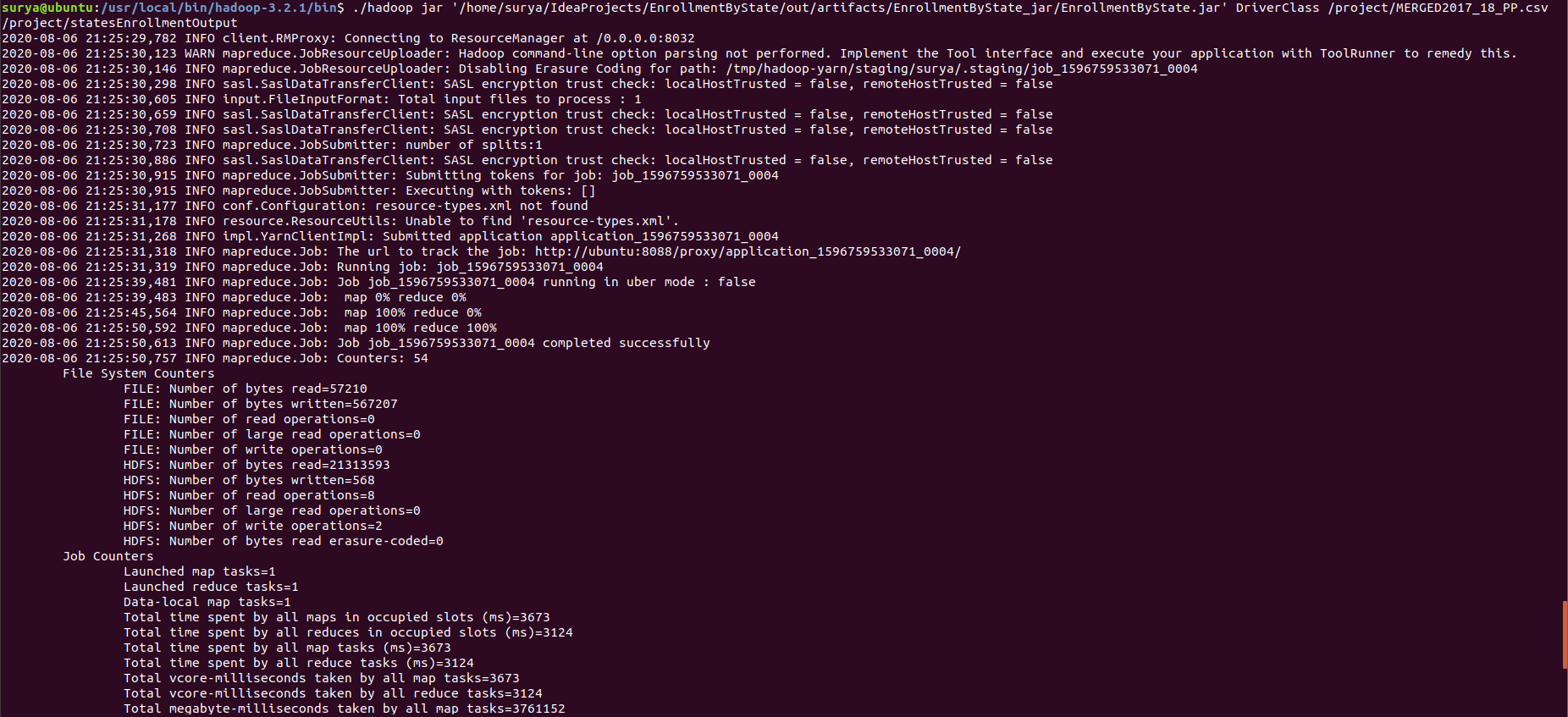
1. Write Map Reduce jobs in Hadoop and create a jar.
2. Make sure the input files are in HDFS with the following command

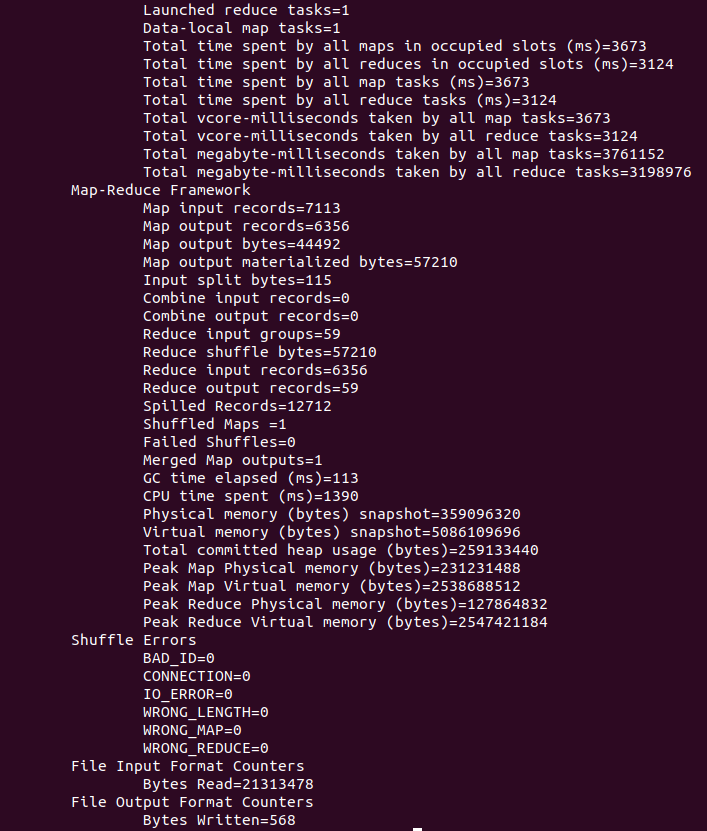
**./hadoop fs -ls /project**



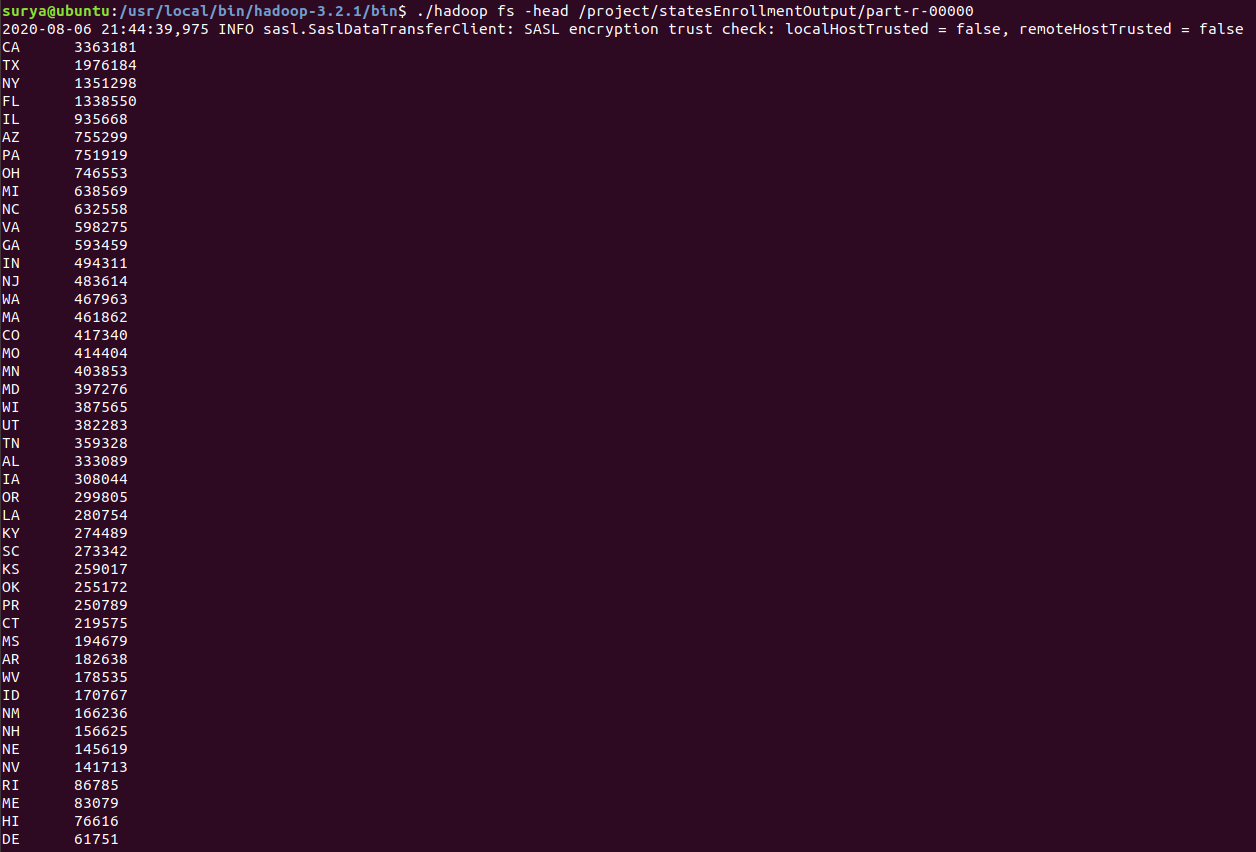
1. Run the jar in Hadoop with the following command.

**./hadoop jar '/home/surya/IdeaProjects/EnrollmentByState/out/artifacts/EnrollmentByState\_jar/EnrollmentByState.jar' DriverClass /project/MERGED2017\_18\_PP.csv /project/statesEnrollmentOutput**

****



1. Check result using the following command. The result will be in the folder specified as the second argument in HDFS

****

In the above screenshot we can see that the states are sorted by the number of students enrolled in their universities.

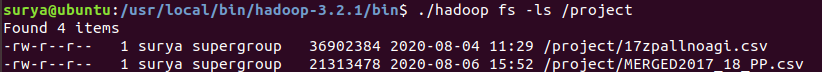
**Note:**

Secondary Sorting is not necessary, since Secondary sorting requires both key and value to be the same composite Writable type and that would be a waste of space. Since number of states will always be less than 55 including territories, we can sort in the clean-up function of reduce function which will be called only once after map reduce functions are complete.

**Analysis # 04: Find minimum, maximum, average, standard deviation of SAT scores for all schools by state**

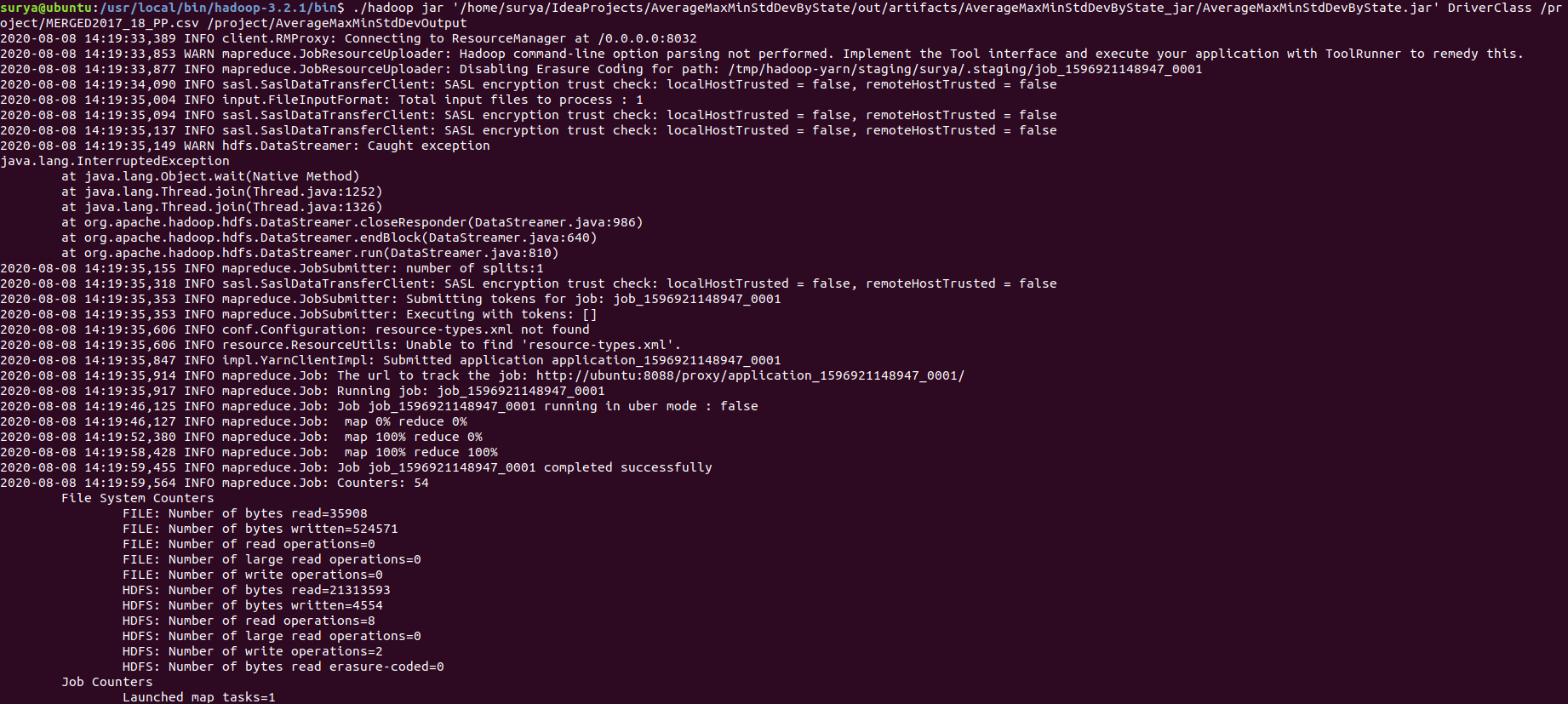
1. Make sure the inputs files are in HDFS with the following command.

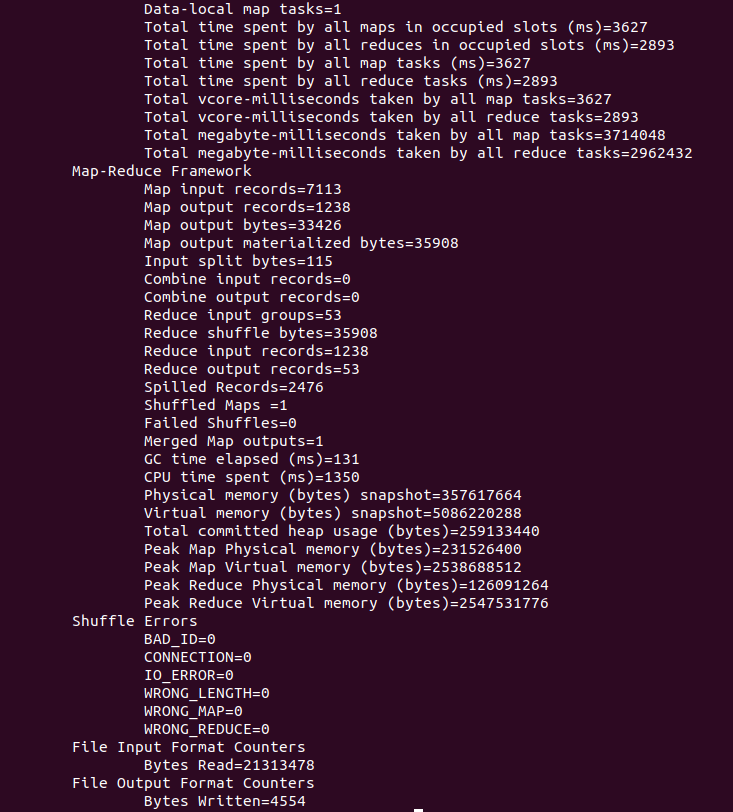
**./hadoop fs -ls /project**



1. Run the jar in Hadoop with the following command.

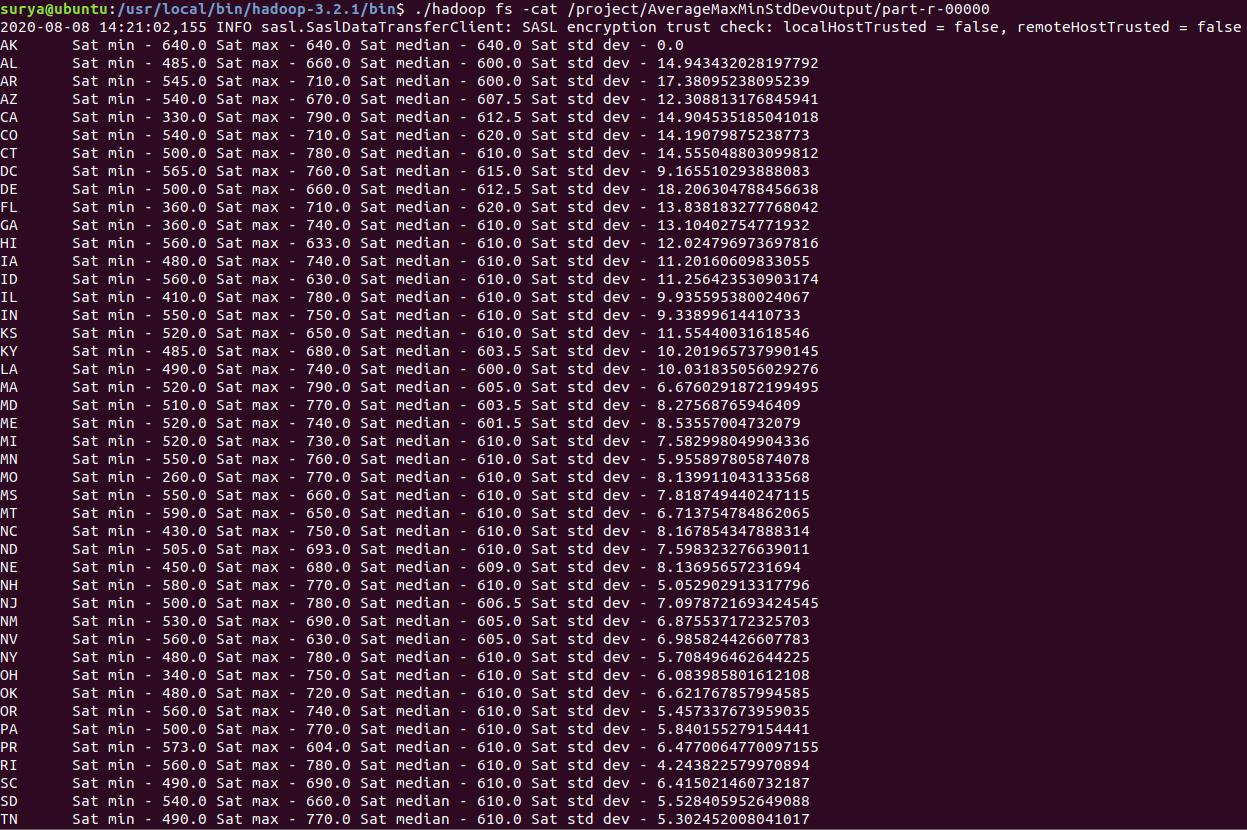
**./hadoop jar '/home/surya/IdeaProjects/AverageMaxMinStdDevByState/out/artifacts/AverageMaxMinStdDevByState\_jar/AverageMaxMinStdDevByState.jar' DriverClass /project/MERGED2017\_18\_PP.csv /project/AverageMaxMinStdDevOutput**

****

****

1. Check result using the following command. The result will be in the folder specified as the second argument in HDFS.

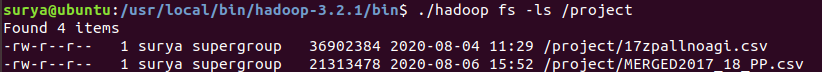
**./hadoop fs -cat /project/AverageMaxMinStdDevOutput/part-r-00000**

****

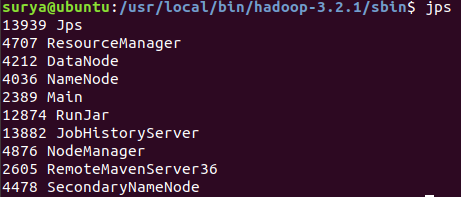
**Getting started with Apache Pig**

1. Make sure the inputs files are in HDFS with the following command and all hadoop services are running

**./hadoop fs -ls /project**

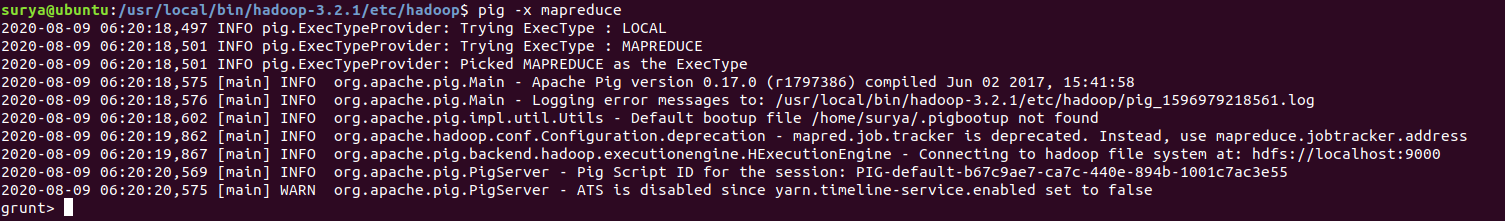


**jsp**

****

1. Make sure pig starts properly

**pig -x hadoop**

3. Load the two data sets into pig

**fieldOfStudy = LOAD 'hdfs://localhost:9000/project/FieldOfStudyData1516\_1617\_PP.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',', 'NO\_MULTILINE', 'UNIX', 'SKIP\_INPUT\_HEADER');**



**collegesList = LOAD 'hdfs://localhost:9000/project/MERGED2017\_18\_PP.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',', 'NO\_MULTILINE', 'UNIX', 'SKIP\_INPUT\_HEADER');**



**Analysis # 05: Find all degrees offered by an institute**

1. Get required fields from first file

**filteredColleges = FOREACH collegesList GENERATE $0 as uniId, $2 as opeId;**

1. Get required fields from second file

**filteredFieldOfStudy = FOREACH fieldOfStudy GENERATE $1 as opeId, $7 as degree;**

1. Apply replicated Join as first file is much smaller than the other when the id is same.

**joined = Join filteredColleges by opeId, filteredFieldOfStudy by opeId USING 'replicated';**

****

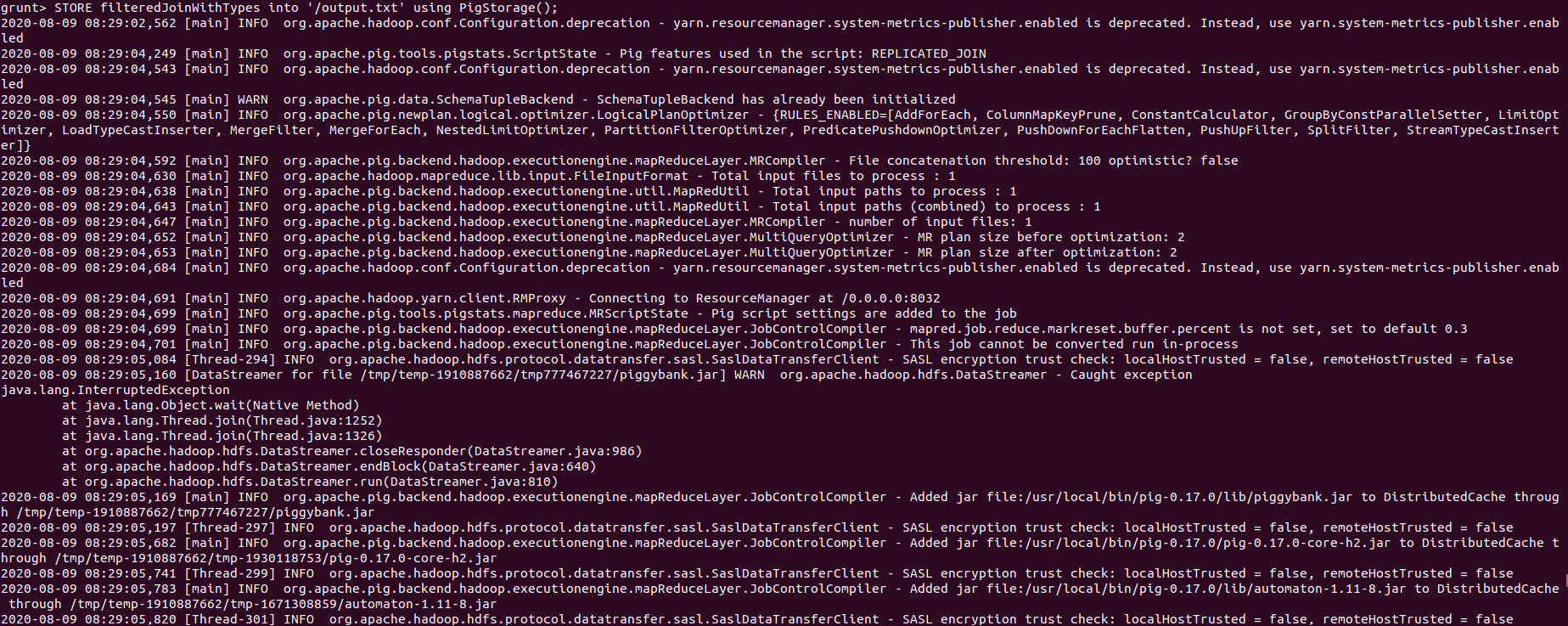
1. Get the id and degree name from the joined view.

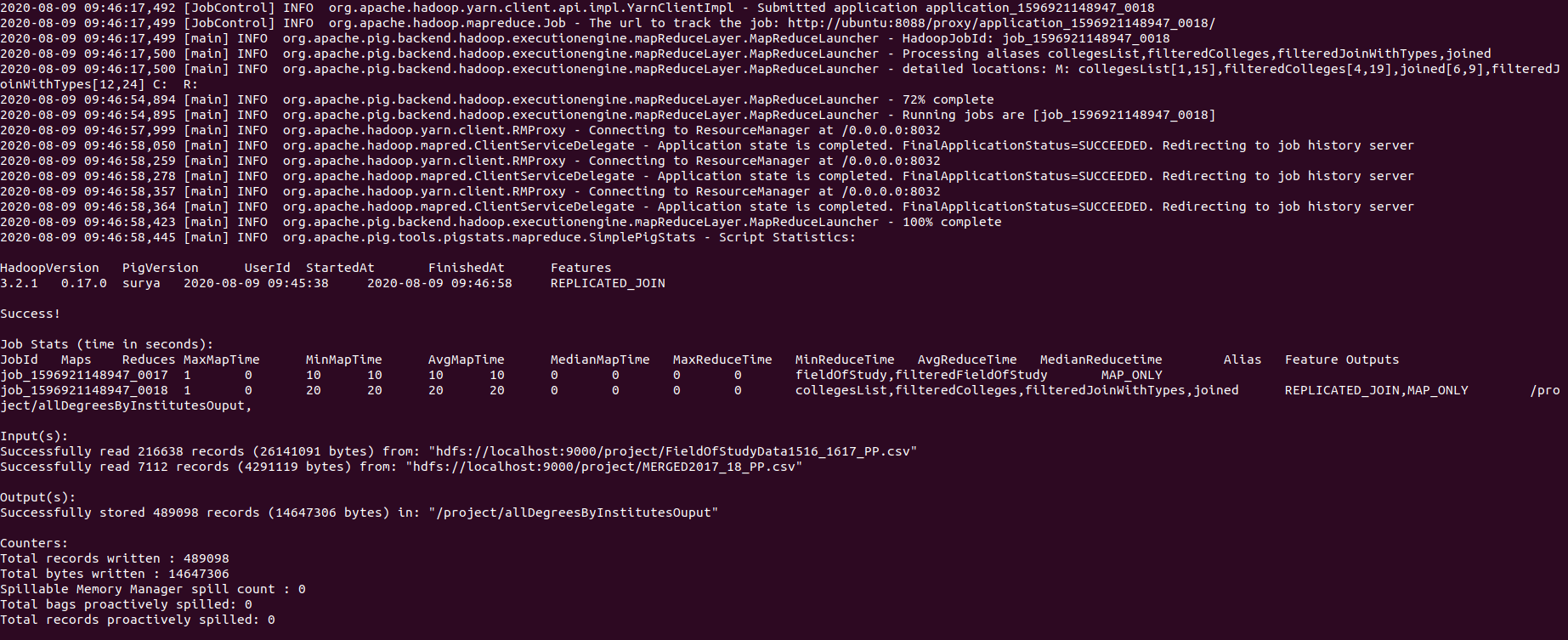
**filteredJoin = FOREACH joined $0 as id, $3 as degreeName;**

****

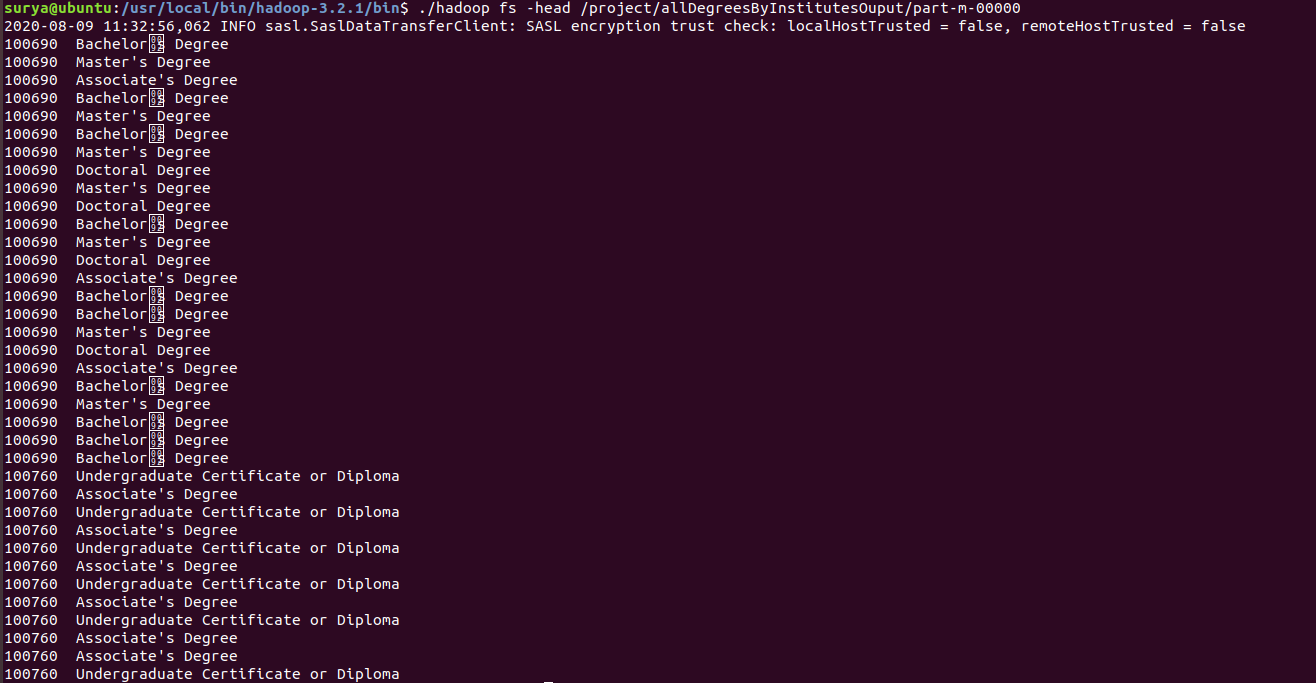
1. Store the filteredJoing variable into a file in HDFS

**STORE filteredJoin into '/project/allDegreesByInstitutesOuput' using PigStorage();**

****



1. Check the file in HDFS



**Analysis # 06: Find top 2 selective colleges by each state**

1. Get required fields from first file

**filteredColleges = FOREACH collegesList GENERATE $0 as uniId, $2 as opeId;**

1. Get state name, college name and admission rate from each row.

**colleges = FOREACH collegesList GENERATE (chararray) $5 as state, (chararray) $3 as collegeName, (float) $14 as admissionRate;**

1. Delete all records which have NULL values

**filteredCollegesByState = FILTER colleges BY ((state IS NOT NULL) AND (collegeName IS NOT NULL) AND (admissionRate IS NOT NULL));**

1. Group all colleges by state

**groupedFilteredByState = GROUP filteredCollegesByState by state;**

1. Get top 2 colleges by admission rate using limit and order operators

**top2FilteredGroupedCollegesByState = FOREACH groupedFilteredByState {**

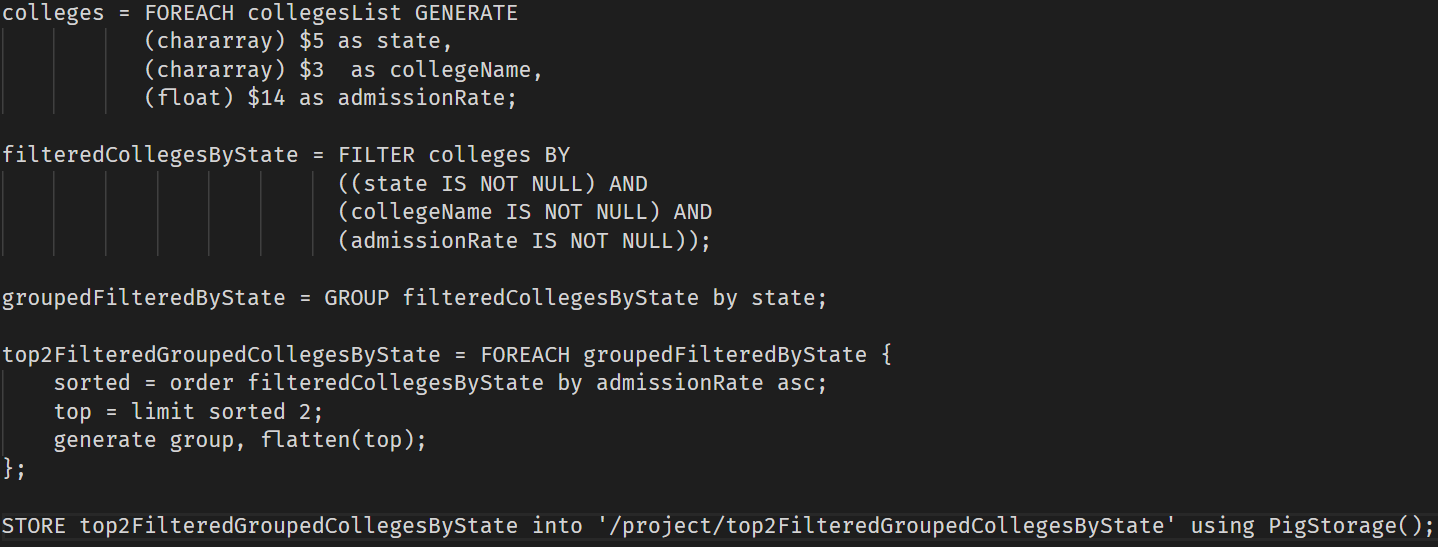
**sorted = order filteredCollegesByState by admissionRate asc;**

**top = limit sorted 2;**

**generate group, flatten(top);**

**};**

All the commands in order:



Check HDFS for output

