**Project 1:**

1. Write a mapreduce program to calculate the number of cases investigated under each FBI code

2. Write a mapreduce program to calculate the number of cases investigated under FBI code 32.

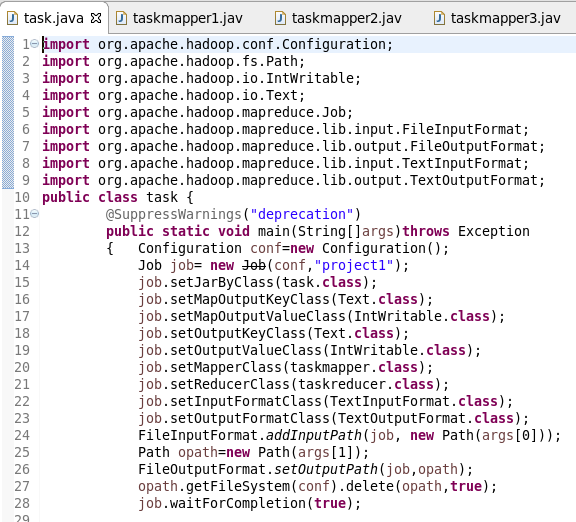
3. Write a mapreduce program to calculate the number of arrests in theft district wise.

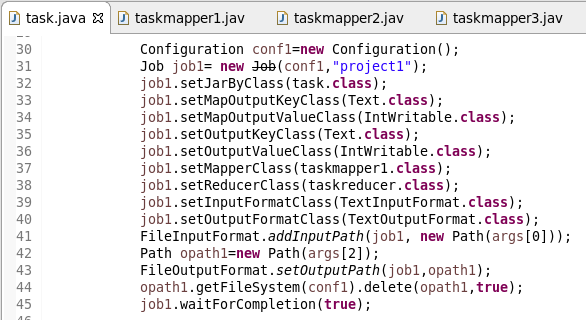
4. Write a mapreduce program to calculate the number of arrests done between October 2014 and October 2015.

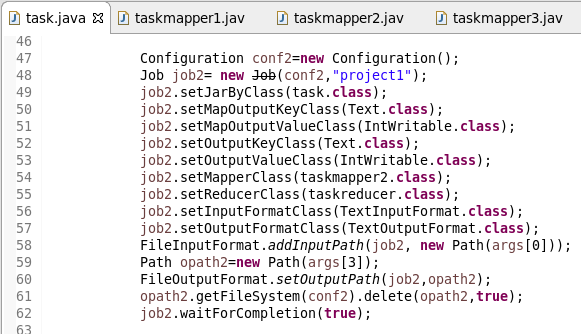
Mapreduce approach:

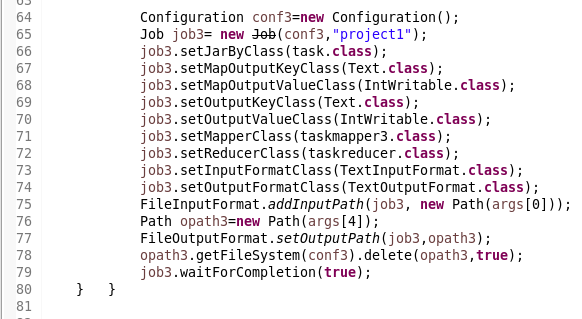
Codes:

Task class:





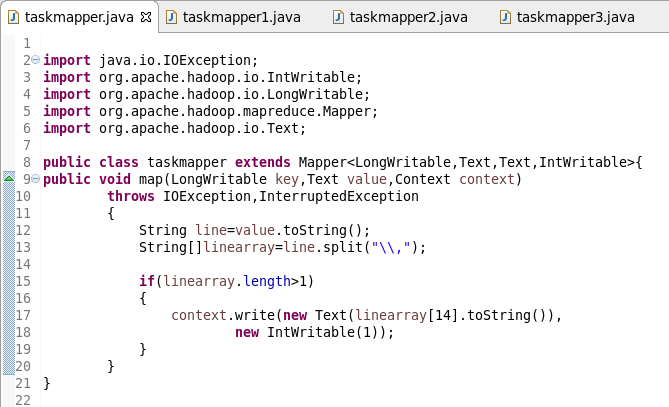




In this driver class (task class) we are creating 4 objects of job dedicated to each problem statement. Respective mapper class and reducer class are set. All the required configurations are set such as input path, output path etc.

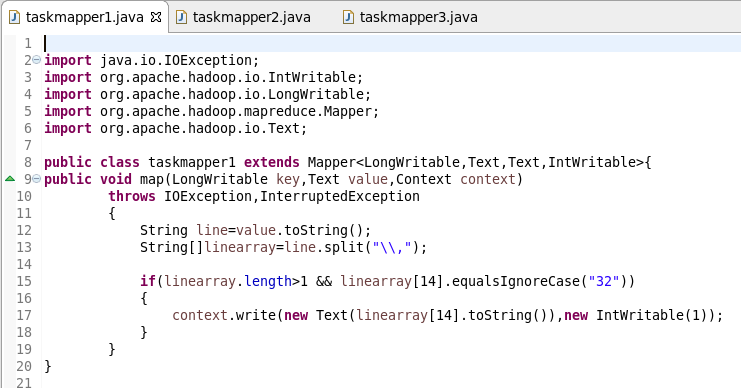
All four jobs are included in same driver class.

Mapper class for problem 1:



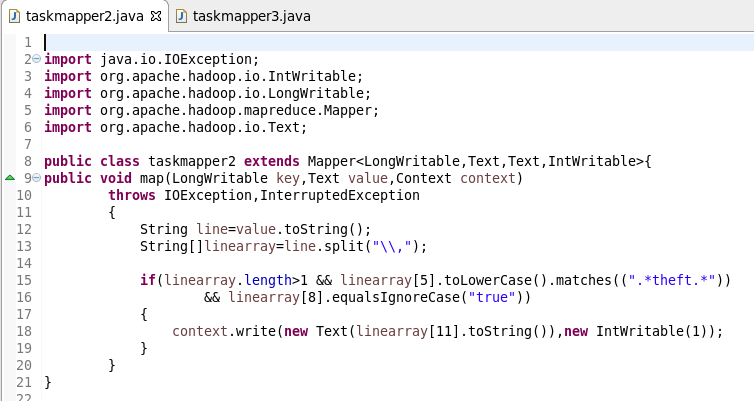
In this class, we are generating key and value pair where key is FBI code and value is intwritable value ‘1’ for taking count.

Mapper class for problem 2:



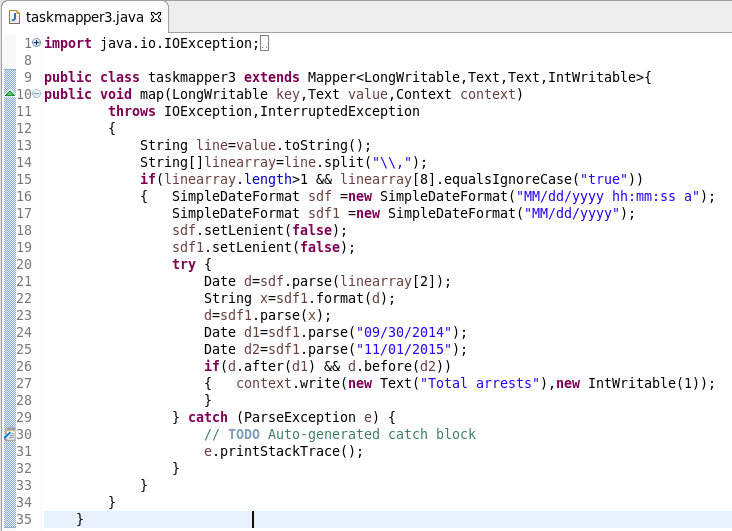
In this mapper we are generating key and value pair where key is FBI code ‘32’ and value is intwritable value ‘1’ to take count of respective cases under the specified code i.e. ‘32’.

Mapper class for problem 3:



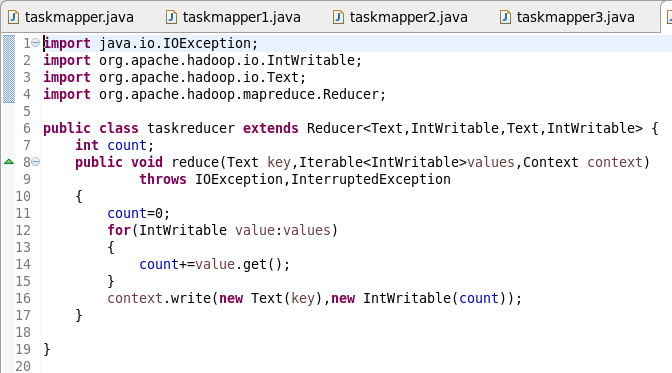
In this class we are generating key and value pair where key is the district id and value is intwritable value ‘1’ to take count of total arrests in theft category in each district. So we are first filtering the records based on arrests and the primary type theft.

Mapper class for problem 4:



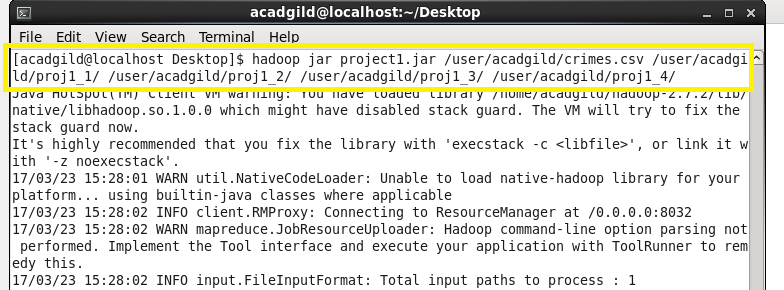
In this mapper we are generating key and value pair where key is kept common as ‘total arrests’ and value is intwritable value ‘1’. Here we have to take count of arrests in particular period. So first we are filtering records based on whether they fall in given range or not.

Reducer class:



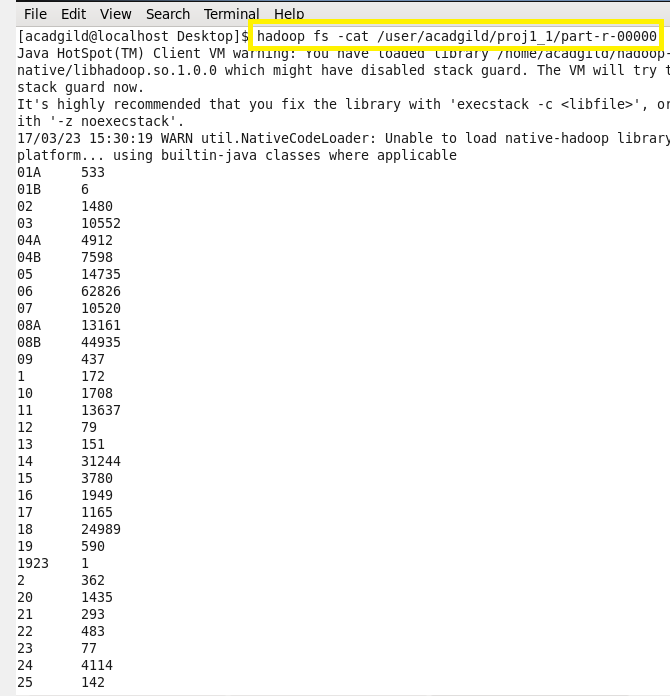
Reducer class is common for all the problem statements. Here we are taking count of each keys coming from the mapper. Reduce method gets invoked for each unique key and the count is calculated by iterating values.

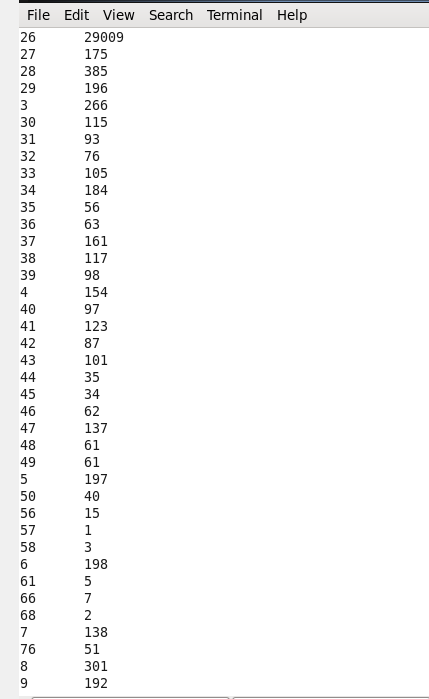
Running jar:



Project1.jar is the jar exported and input and output paths are given.

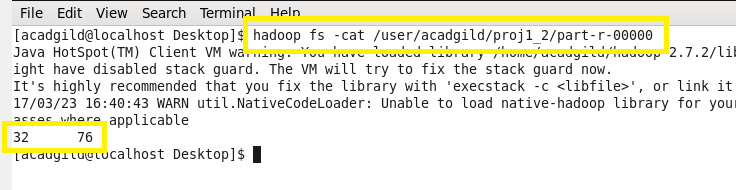
Output 1:





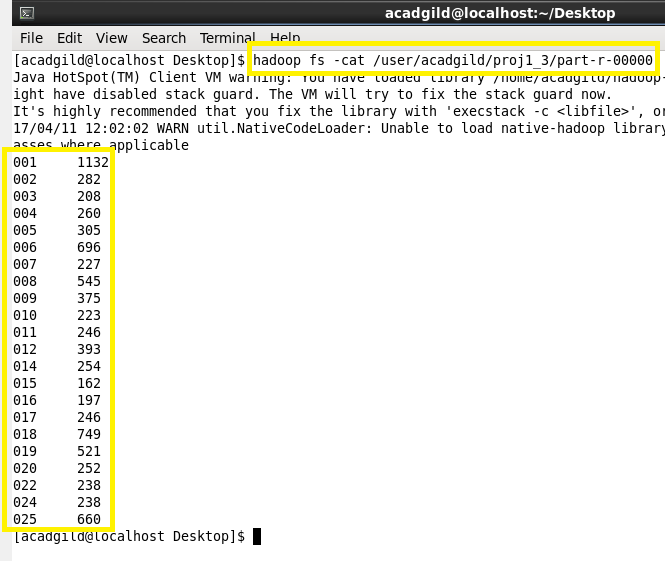
These are the total number of cases under each FBI code.

Output 2:



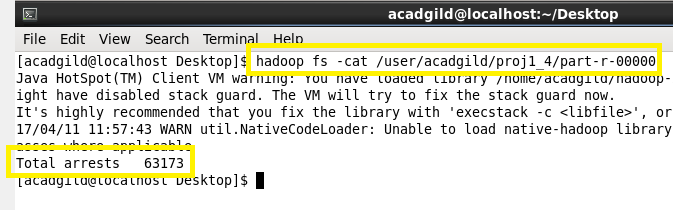
Total cases handled by FBI code ‘32’ are 76.

Output 3:



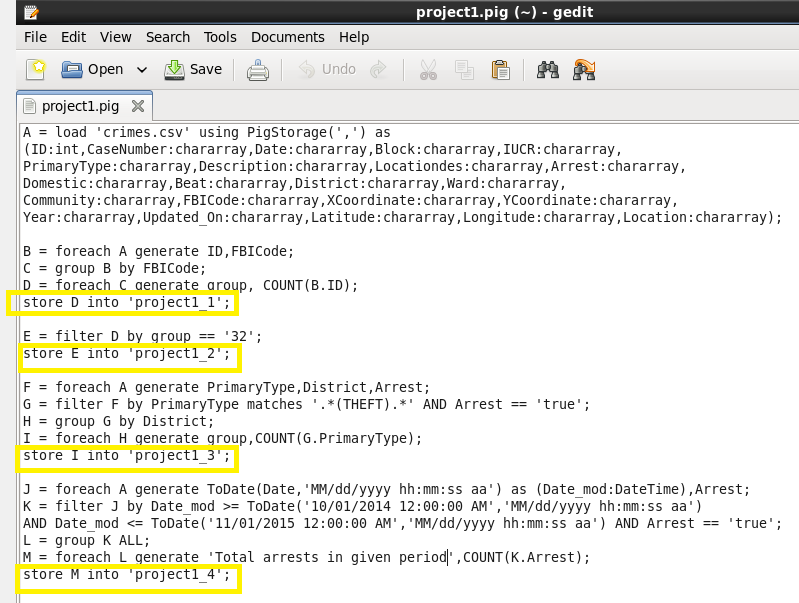
These are the total arrests in theft category in each state.

Output 4:



These are the total number of cases during October 2014 and October 2015.

Pig approach:



Problem 1:

First we loaded all the data in A. Generate B such as it contains only case ID and FBICode. To take count of cases under each FBICode first we grouped B by FBICode and then took count of IDs.

Problem 2:

For problem 2, we filtered D such that it contains data related to FBICode ‘32’.

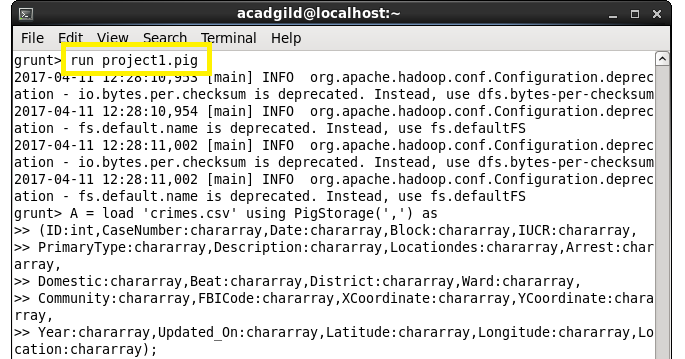
Problem 3:

First we filtered F to get all the arrests in theft category. Then group H by district and then take count of primary\_type which will give count of arrests in theft category for each district.

Problem 4:

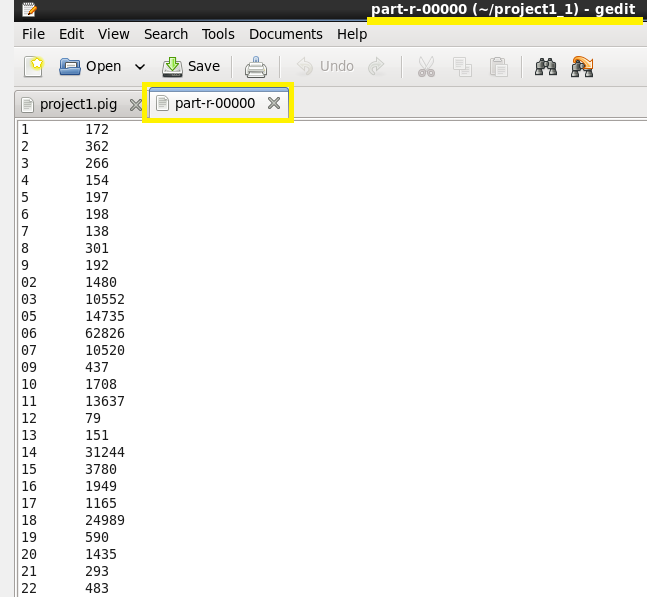
First we filtered the records so that we will get all the records in particular time period and the take count of all arrests.

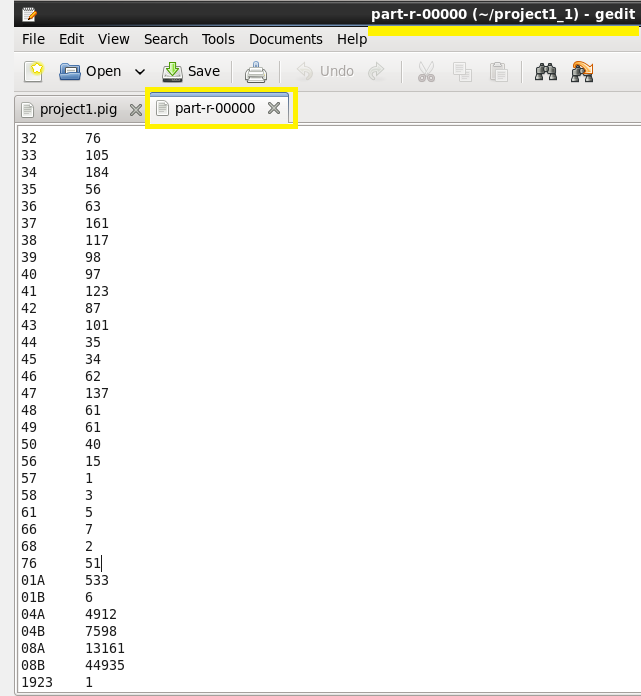
Running pig script:



The pig script is written in a project1.pig file. Run this .pig file on grunt shell. Respective results will be stored in respective files.

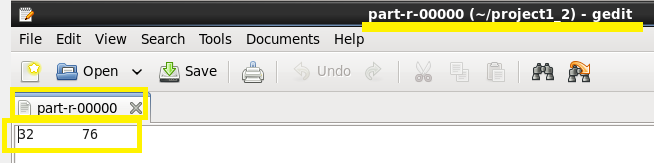
Output of problem 1:





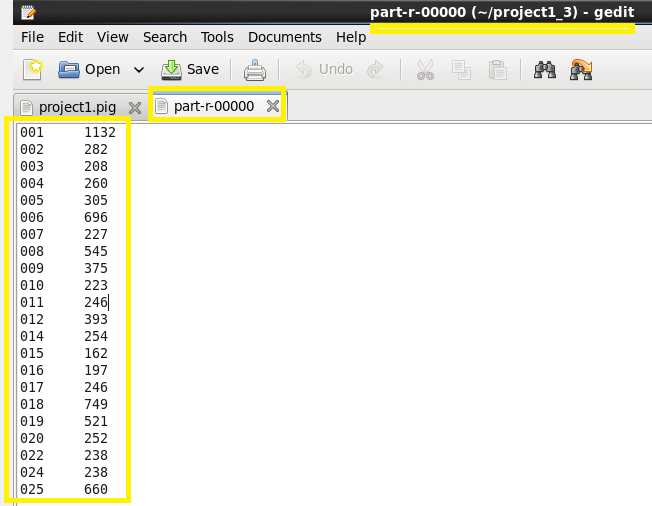
These are the total number of cases under each FBI code.

Output of problem 2:



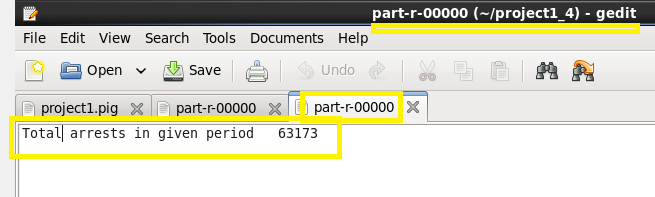
Total cases handled by FBI code ‘32’ are 76.

Output of problem 3:



These are the total arrests in theft category in each state.

Output of problem 4:



These are the total number of cases during October 2014 and October 2015.

Conclusion:

Results obtained from pig and mapreduce are same. Pig is abstraction over mapreduce. Only the difference is by using pig we can process data with minimum number of lines and it is more readable.