qwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnm

|  |
| --- |
| PROJECT-1  US-CRIME-ANALYSIS  1/29/2018  ABINAYA RAJAGOPAL |

***CONTENTS***

1. calculate the number of cases investigated under each FBI code .............................................. 3
2. calculate the number of cases investigated under FBI code 32.................................................. 8
3. calculate the number of arrests in theft district wise................................................................. 12
4. calculate the number of arrests done between October 2014 and October 2015..................... 16
5. Single script................................................................................................................................. 20

**PROJECT-1**

**(US-CRIME-ANALYSIS)**

1. Write a MapReduce/Pig program to calculate the number of cases investigated under each

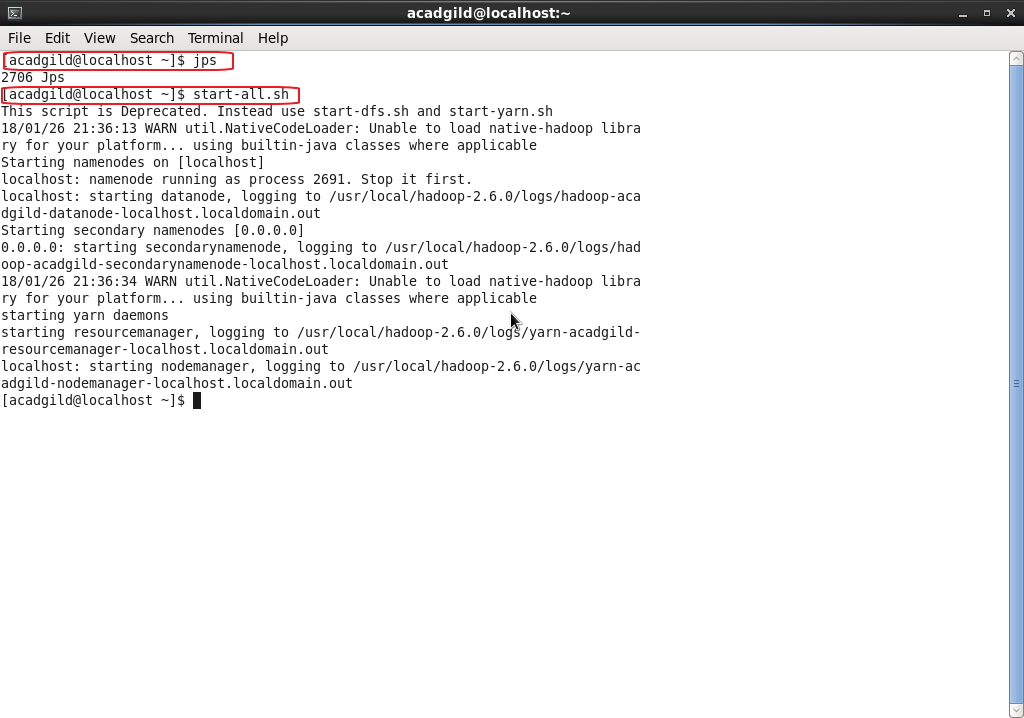
FBI code

**STEP-BY-STEP-EXECUTION:**

* The required Hadoop threads are started by firing the below command:

start-all.sh

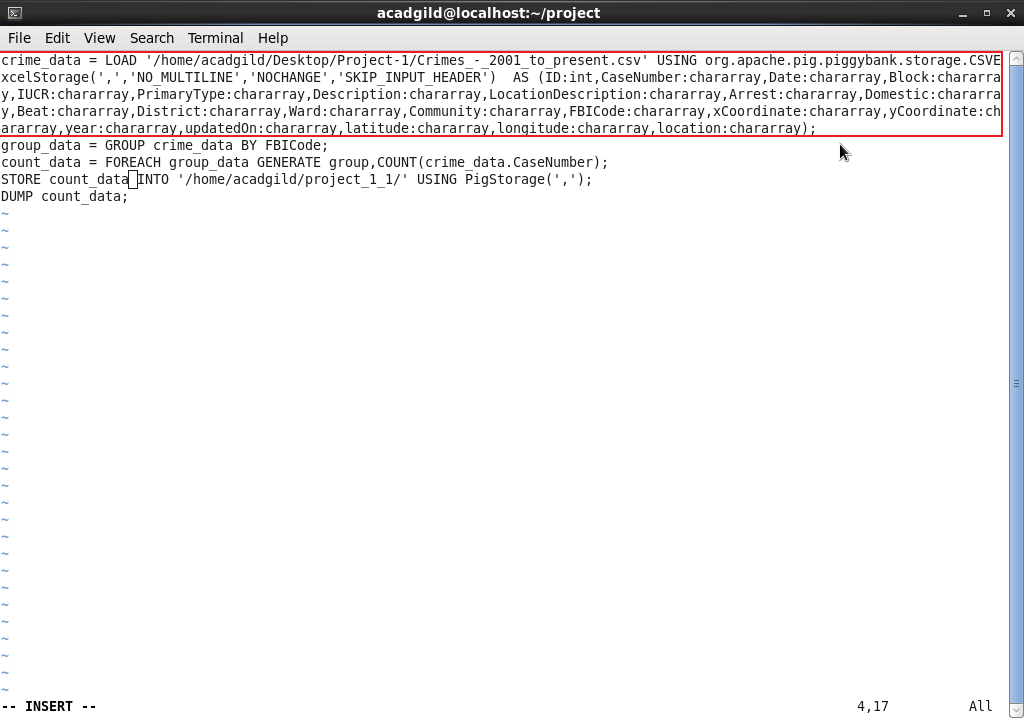
The running daemons can be checked using command: - **jps**

****

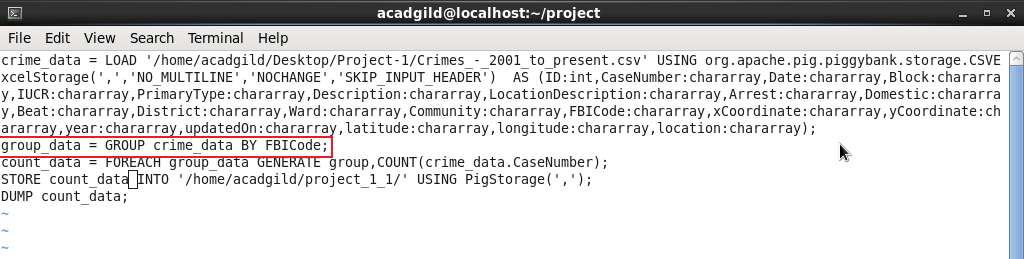
* loadFile.pig is created in path /home/acadgild/Project\_1 and the loading and filteration/Data Enrichment is done via the script.

The input file provided is in csv format and is placed in /home/acadgild/Desktop/Project\_1 folder.

Hence, the whole path needs to be provided along with the filename, the file will be loaded using org.apache.pig.piggybank.storage.CSVExcelStorage and the corresponding column names to be loaded are passed along with the datatype.

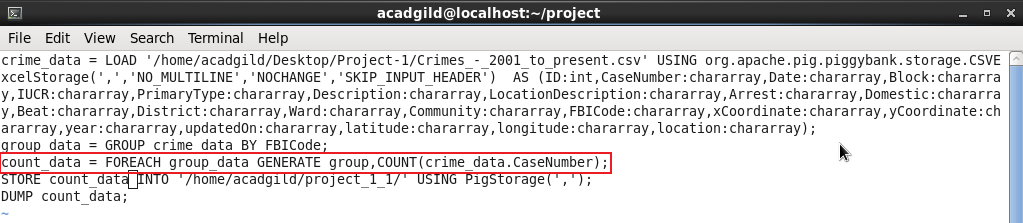


* The contents loaded from the csv file are grouped based on FBICode in the screenshot below.

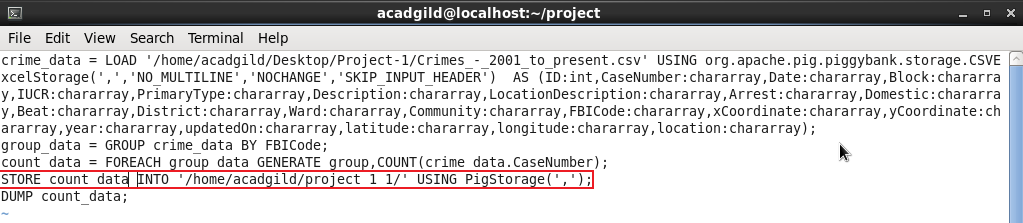
****

* Count of number of cases as per the FBICode needs to be calculated.

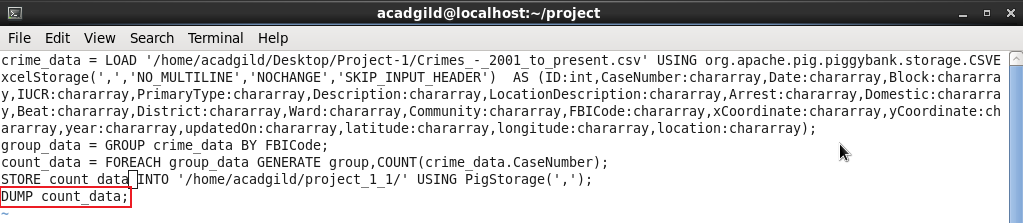
Hence, in the below screenshot, on the grouped data – count of cases based on FBICode is calculated.

****

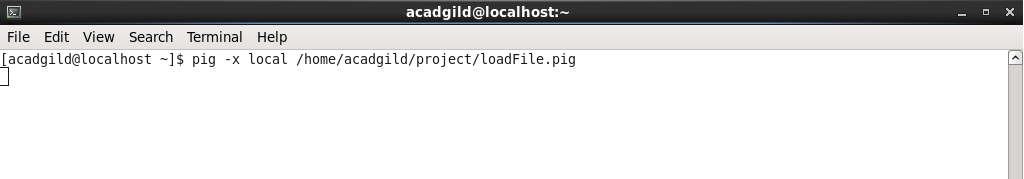
* The final output will be stored in a file delimited with commas and below is the highlighted screenshot for the same.

****

* DUMP command dumps the final output on to the console after execution.

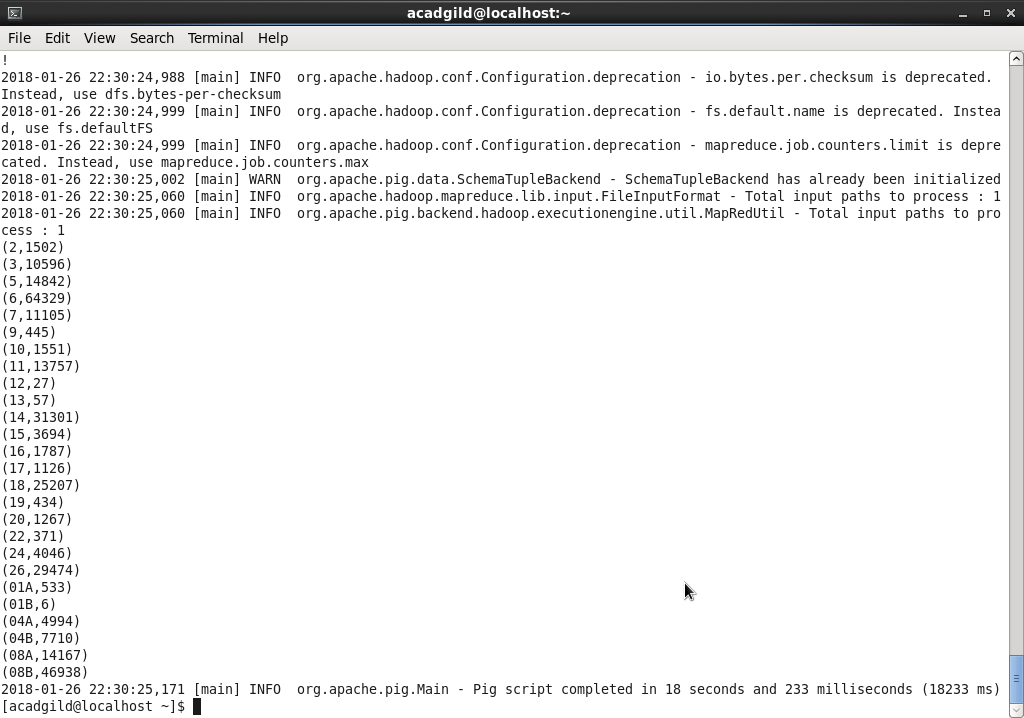
****

* Once the script is saved, this will be executed in pig in local mode as shown below.

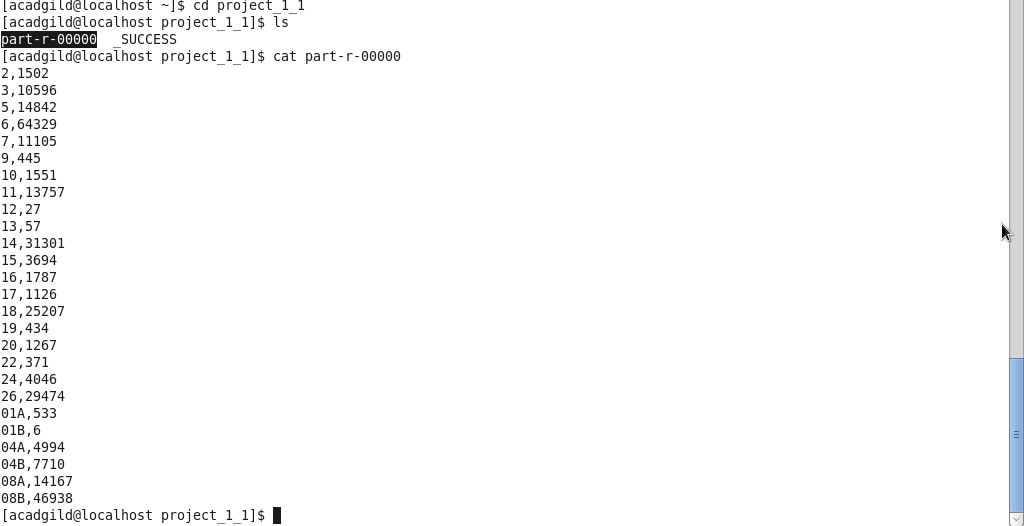
****

* Below is the result shown on the console

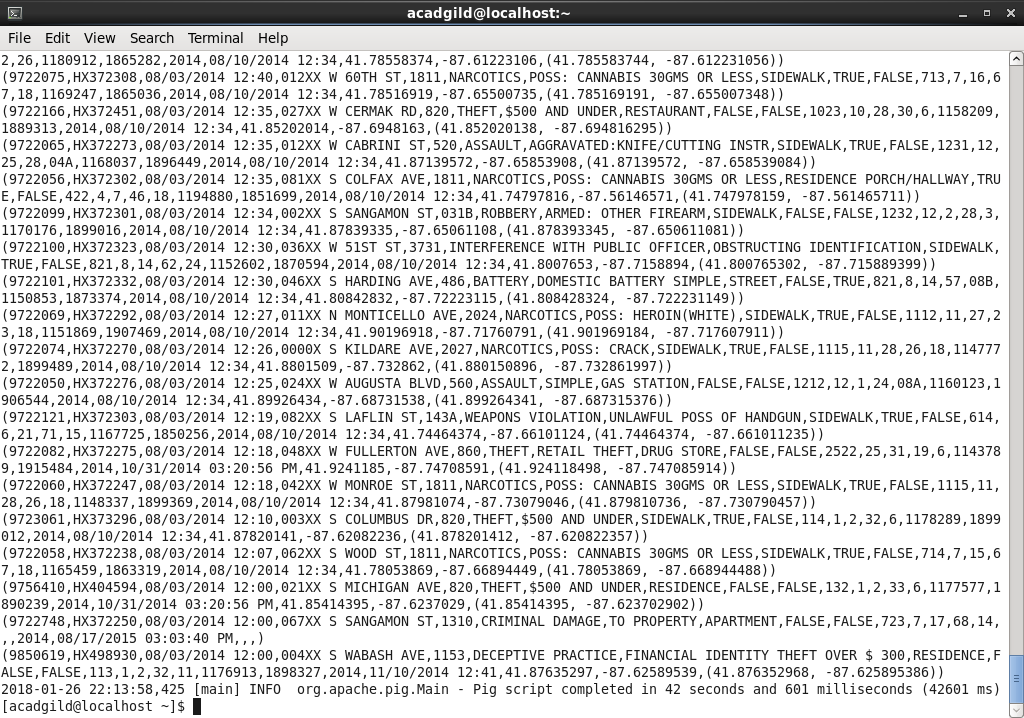
***DUMP Command:***

******

***STORE COMMAND:*** *Results stored in the file will be viewed using cat command.*

**

* Once the csv initial file is loaded, using DUMP command the results can be verified as below



Script file below is attached for reference:



Output files:



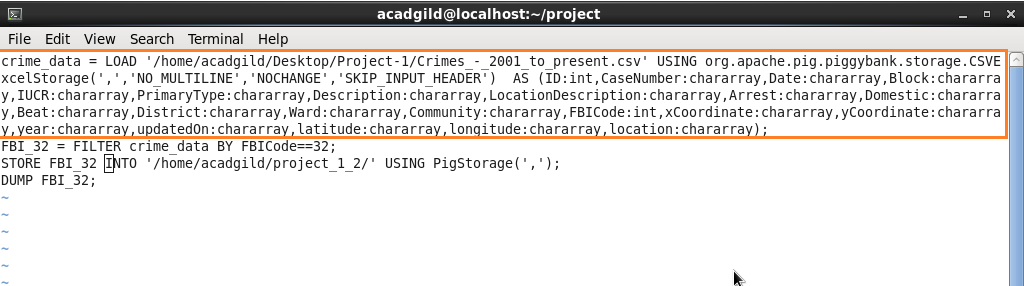
1. Write a MapReduce/Pig program to calculate the number of cases investigated under FBI code 32.

**STEP BY STEP EXECUTION:**

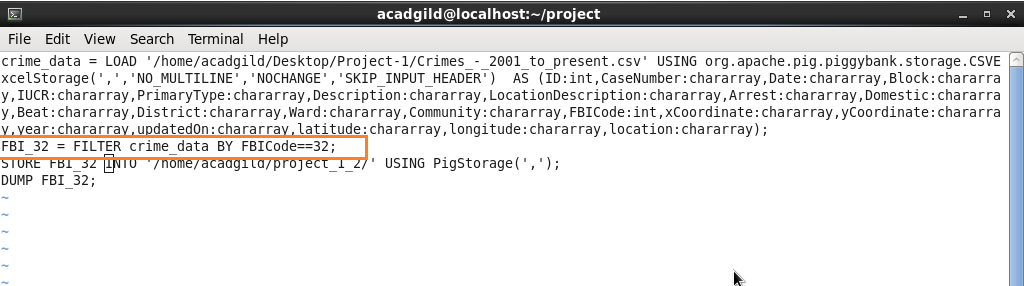
* All the daemon threads are already started as part of Program 1. If not, the same step 1 would be repeated from Program 1 to start Hadoop cluster.
* **Fbicode\_32.pig** is created in path /home/acadgild/Project\_1 and the loading and filteration/Data Enrichment is done via the script. Each time the input file is loaded.

The **input file** provided is in csv format and is placed in ***/home/acadgild/Desktop/Project\_1*** folder.

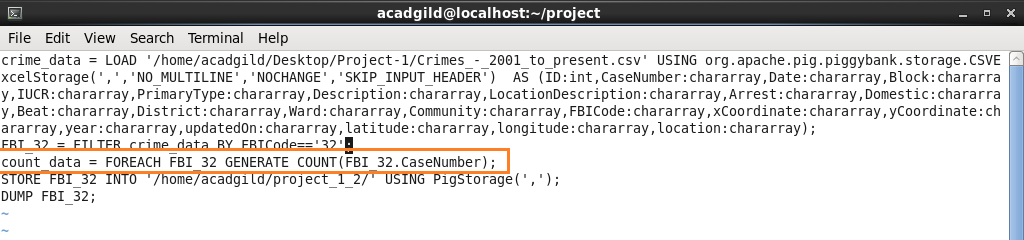
Hence, the whole path needs to be provided along with the filename, the file will be loaded using org.apache.pig.piggybank.storage.CSVExcelStorage and the corresponding column names to be loaded are passed along with the datatype.



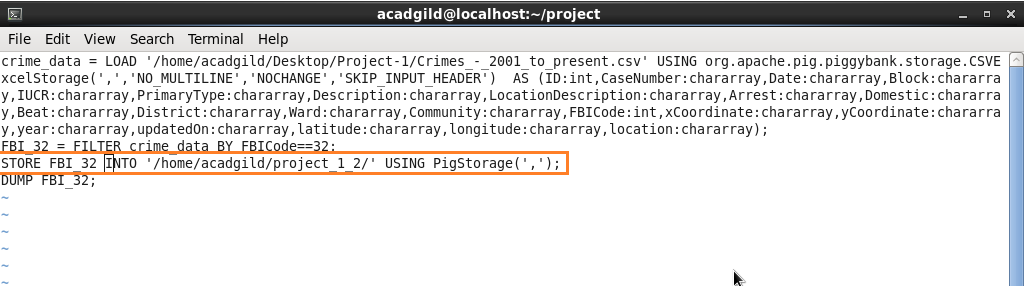
* The loaded data is filtered by FBICode. The records with FBICode==32 needs to be pulled out.



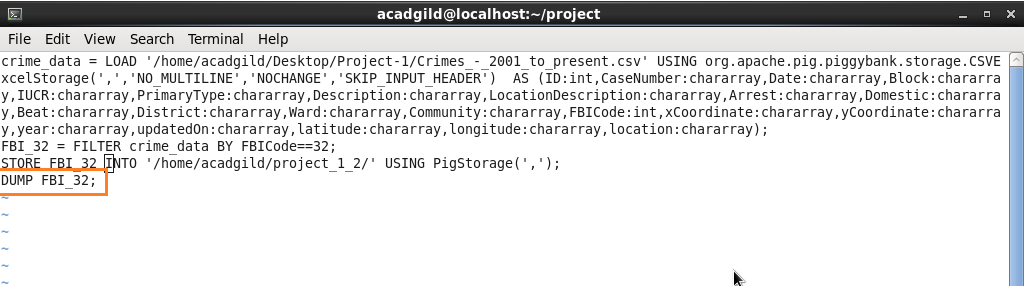
* Final count of loaded data is calculated using FOREACH command and COUNT function.



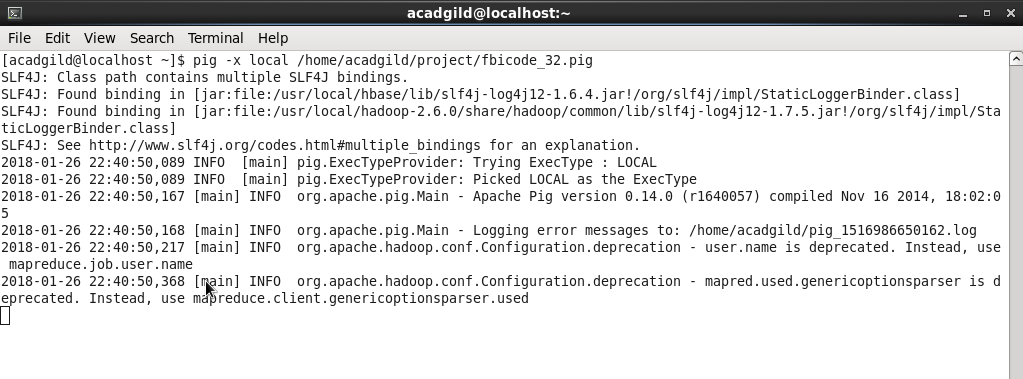
* The filtered data will be stored in the file path specified - /home/acadgild/Project\_1\_2 delimited with commas.



* DUMP command dumps the final output on to the console after execution.

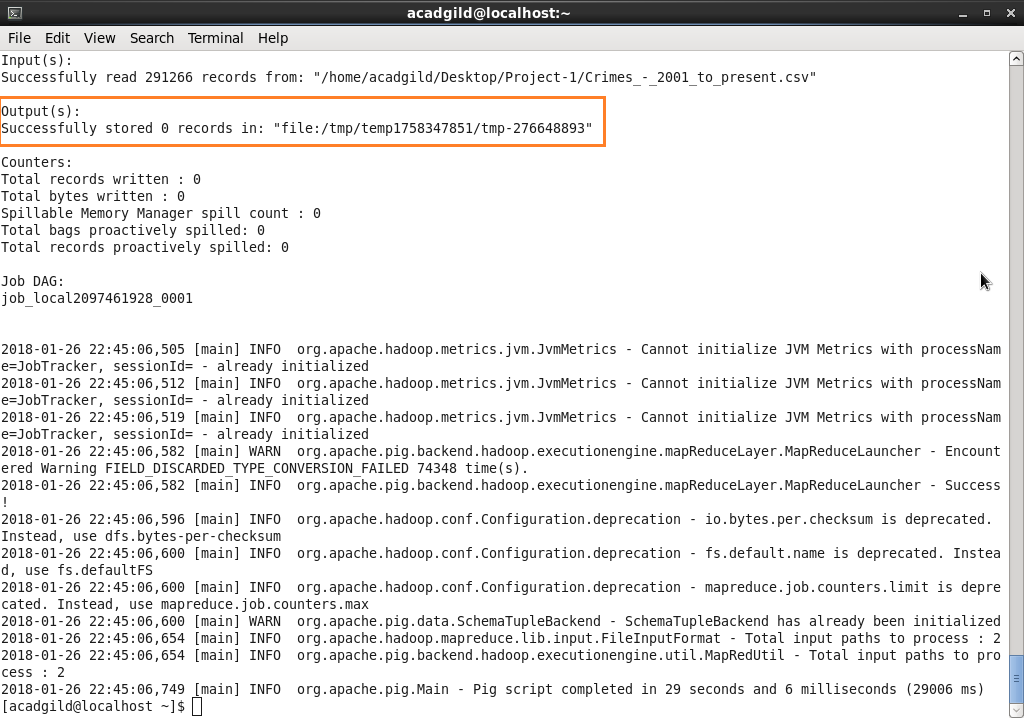


* Once the script is saved, this will be executed in pig in local mode as shown below.

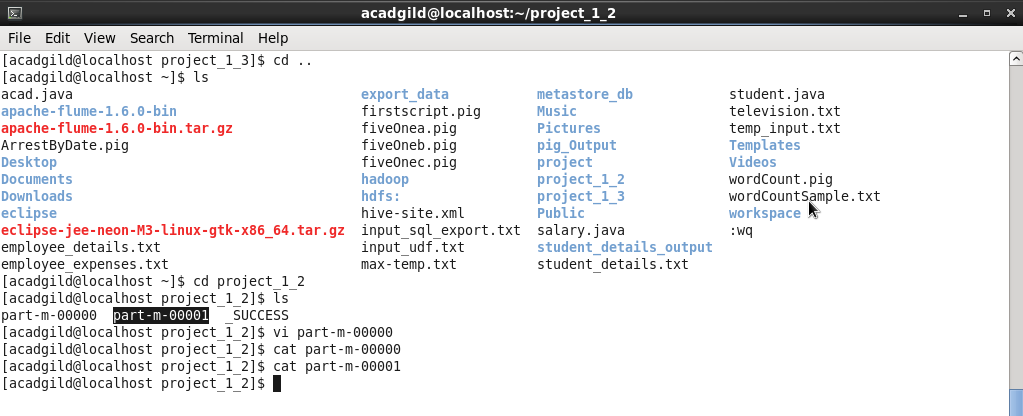


* Below is the result shown on the console

***DUMP Command: -*** As there are no records with fbicode==32, no records have been displayed



***STORE COMMAND: -*** *Results stored in the file will be viewed using cat command (*NO records to be shown)



*Script file is attached below for reference:*

**

*Output files below:*

**

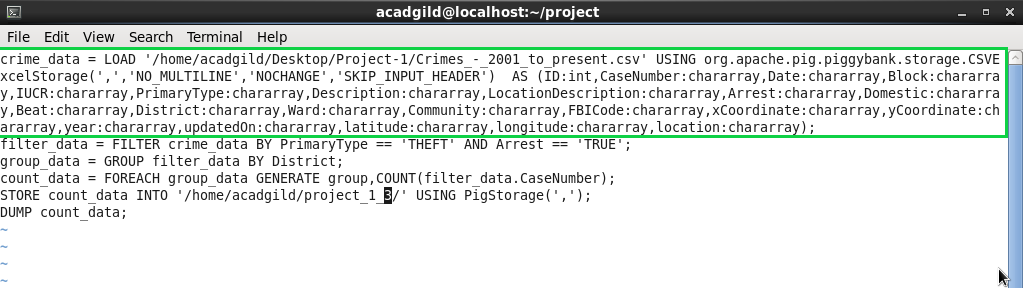
1. Write a MapReduce/Pig program to calculate the number of arrests in theft district wise.

**STEP BY STEP EXECUTION:**

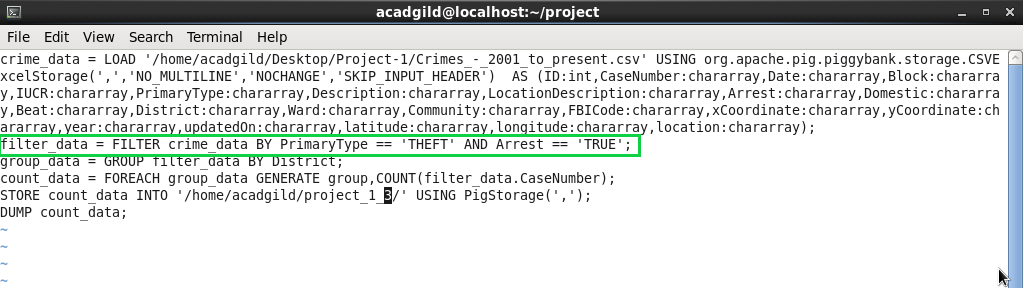
* All the daemon threads are already started as part of Program 1. If not, the same step 1 would be repeated from Program 1 to start Hadoop cluster.
* **theft\_district.pig** is created in path /home/acadgild/Project\_1 and the loading and filteration/Data Enrichment is done via the script. Each time the input file is loaded.

The **input file** provided is in csv format and is placed in ***/home/acadgild/Desktop/Project\_1*** folder.

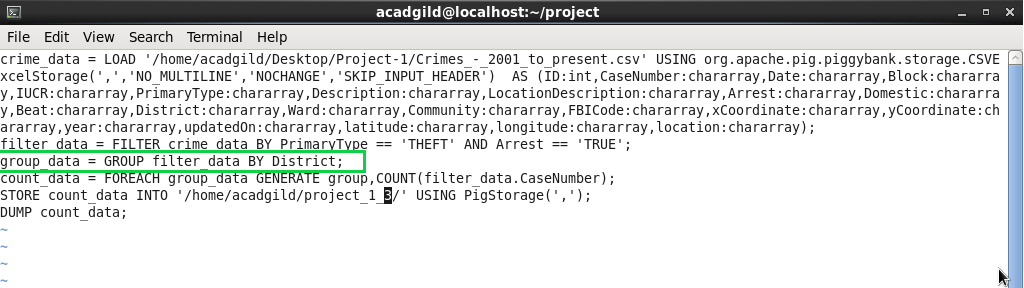
Hence, the whole path needs to be provided along with the filename, the file will be loaded using org.apache.pig.piggybank.storage.CSVExcelStorage and the corresponding column names to be loaded are passed along with the datatype.



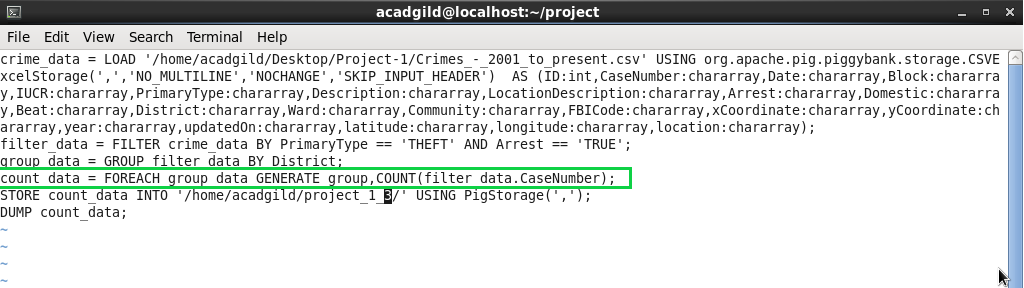
* The loaded data will be filtered based on column PrimaryType having Theft and column Arrest having True.



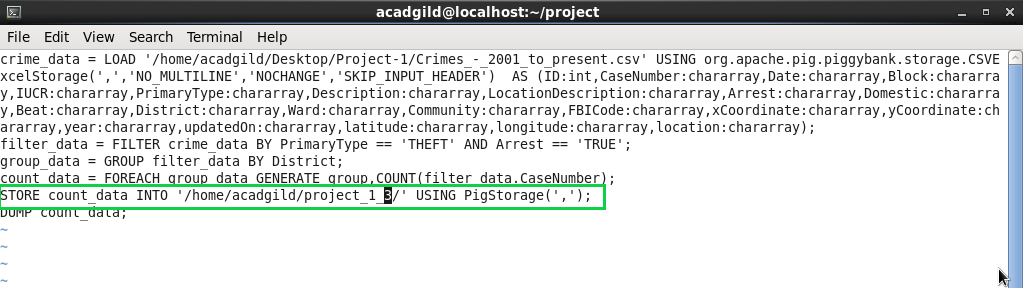
* Next, the filtered data is grouped by district.



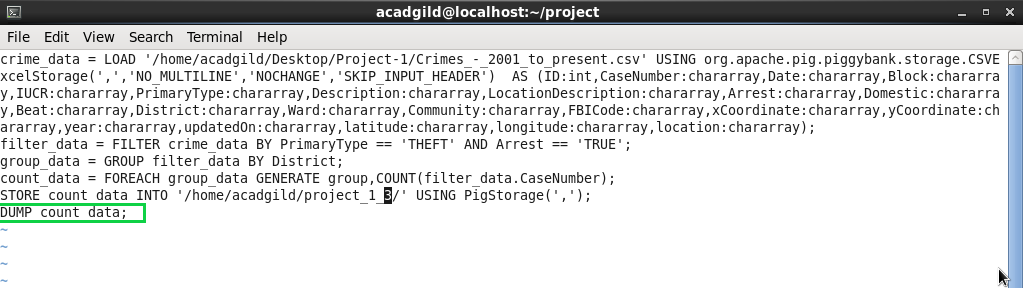
* Total count of arrests in theft district wise is calculated using **FOREACH** command. For each of the grouped data, based on the group, count of case numbers is calculated which would inturn return the number of arrests in theft district wise.



* The final output can be stored in a file and the below command is an example to store the results in an external file. The results will be stored in the path specified. Here, it will be stored in ***/home/acadgild/Project\_1\_3*** delimited with commas.



* Using Dump command the results can be dumped on the console as soon as script execution completes.

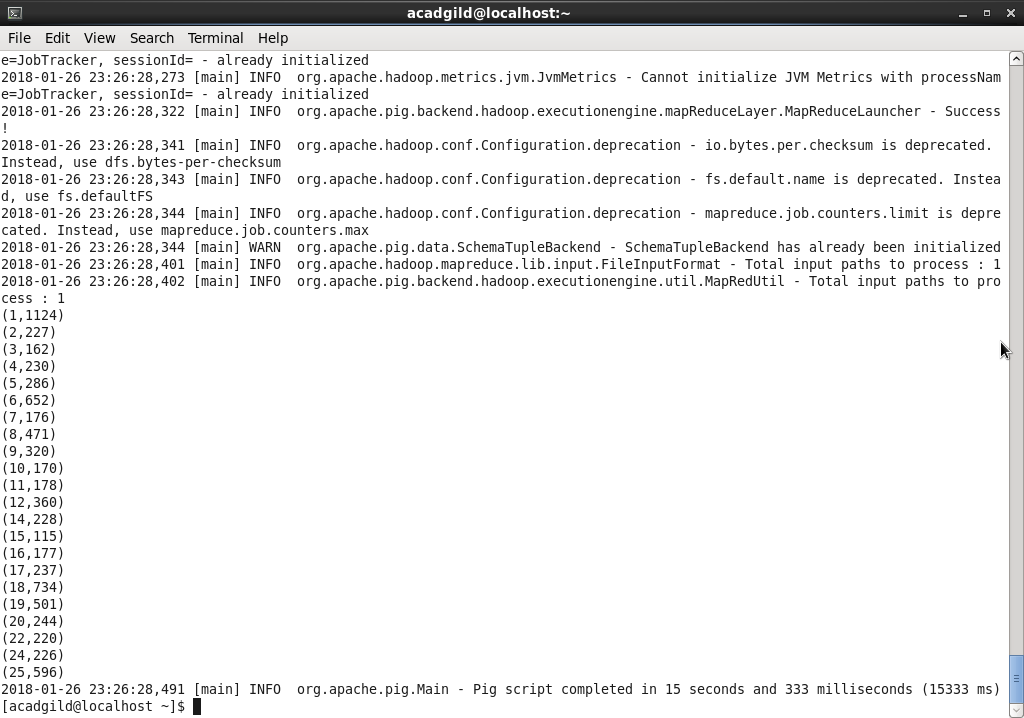


* Once the script is saved, this will be executed in pig in local mode as shown below.

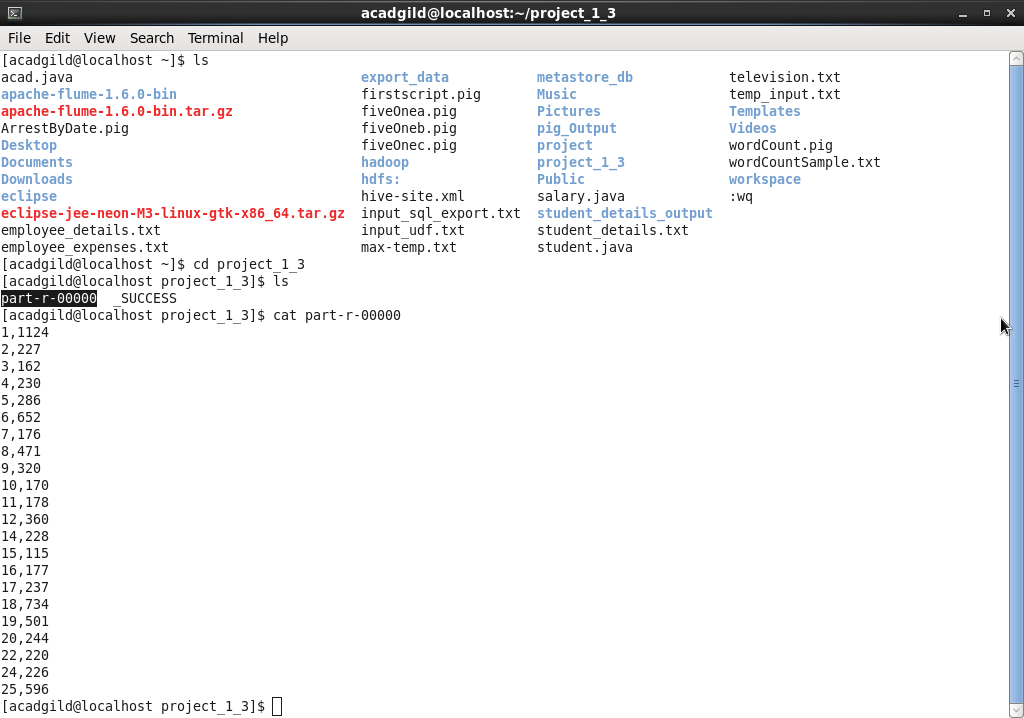


* Below is the result shown on the console

***DUMP Command***



***STORE COMMAND: -*** *Results stored in the file will be viewed using cat command*



*Script file is attached below for reference:*



*Output files below:*

**

1. Write a MapReduce/Pig program to calculate the number of arrests done between October

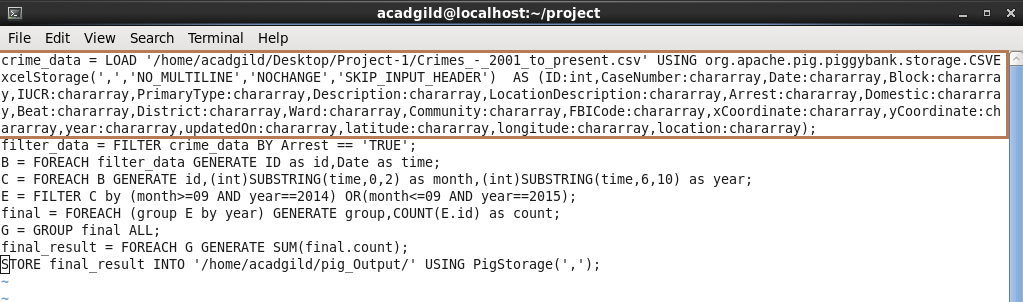
2014 and October 2015.

**STEP BY STEP EXECUTION:**

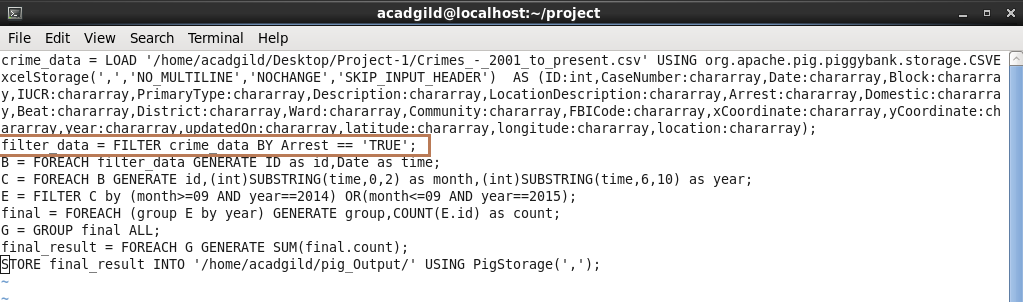
* All the daemon threads are already started as part of Program 1. If not, the same step 1 would be repeated from Program 1 to start Hadoop cluster.
* **ArrestByDate.pig** is created in path /home/acadgild/Project\_1 and the loading and filteration/Data Enrichment is done via the script. Each time the input file is loaded.

The **input file** provided is in csv format and is placed in ***/home/acadgild/Desktop/Project\_1*** folder.

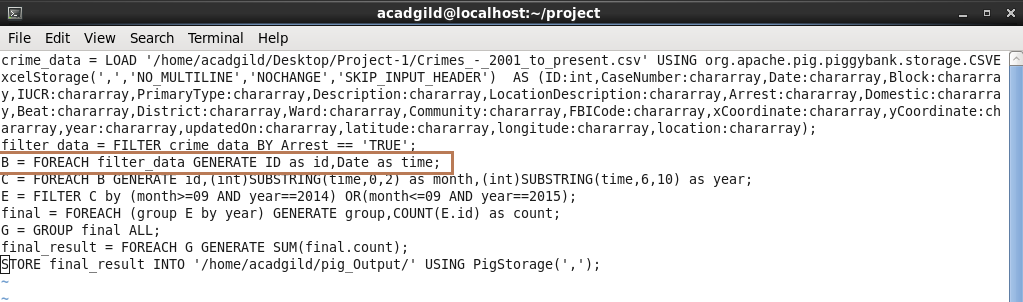
Hence, the whole path needs to be provided along with the filename, the file will be loaded using org.apache.pig.piggybank.storage.CSVExcelStorage and the corresponding column names to be loaded are passed along with the datatype.



* Loaded data is filtered based on column Arrest having value as True.



* From the filtered data, columns ID and Date are pulled and given an alias name as id and time correspondingly.

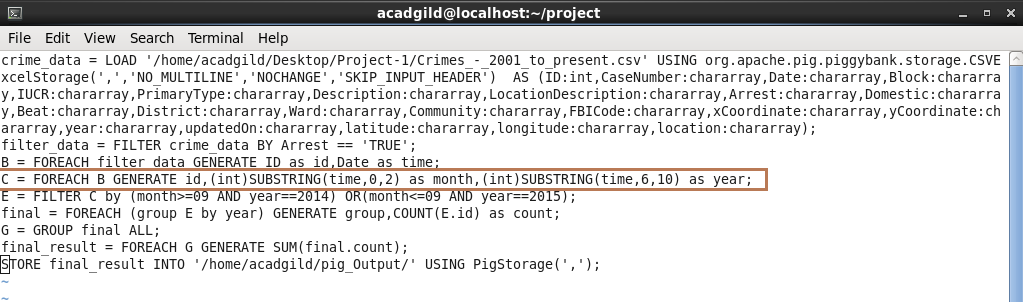


* From the pulled out columns having *id and time as alias names* , now from the time column month and year needs to be pulled. This can be achieved using ***substring*** function.

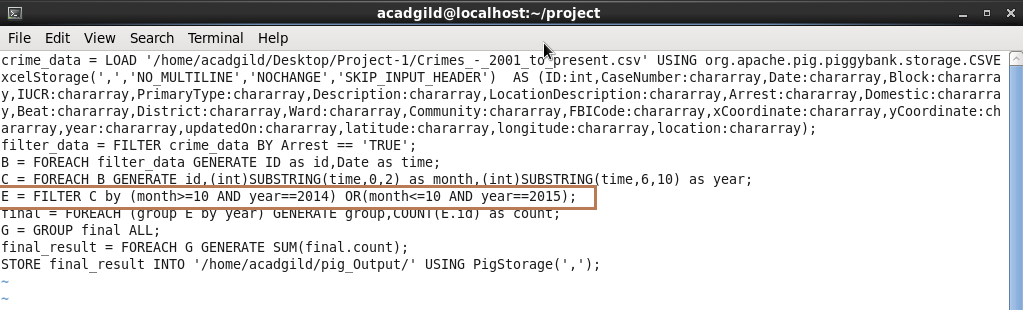
SUBSTRING(time,0,2) as month -> From the time column the values from position 0 to 2 will be pulled and will be stored as month. Further references to this can be made using this alias name, say month.

SUBSTRING(time,6,10) as year-> From the time column the values from position 6 to 10 will be pulled and will be stored as year. Further references to this can be made using this alias name, say year.

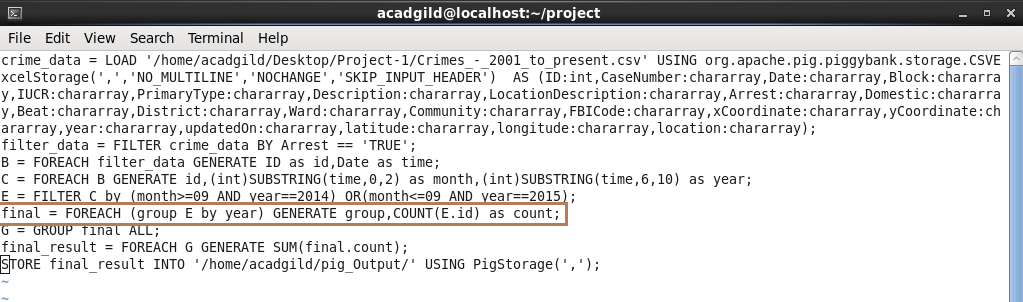
Below is the screenshot depicting the above explained lines:



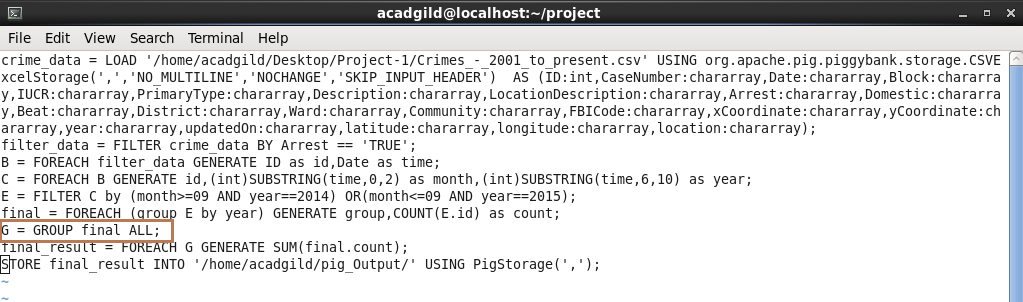
* On the newly generated values month and year in the above step, now the data needs to be filtered and the records need to be pulled, say those having month>=10 and year ==2014 and those having month<=10 and year ==2015, which means the data between October 2014 and October 2015.



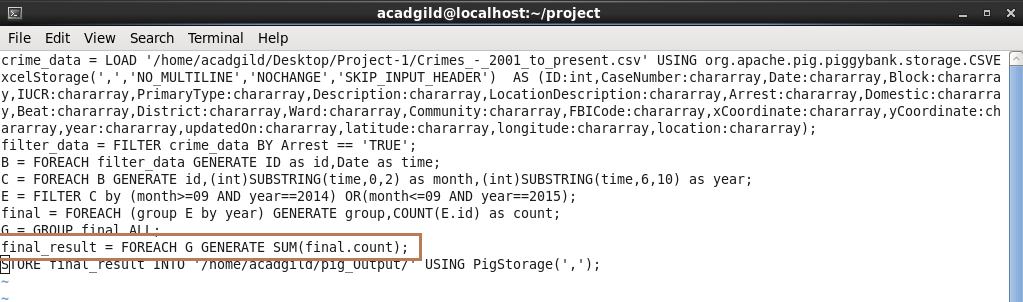
* For each of the filtered data in the above step it is now grouped by year and the count of cases is calculated.



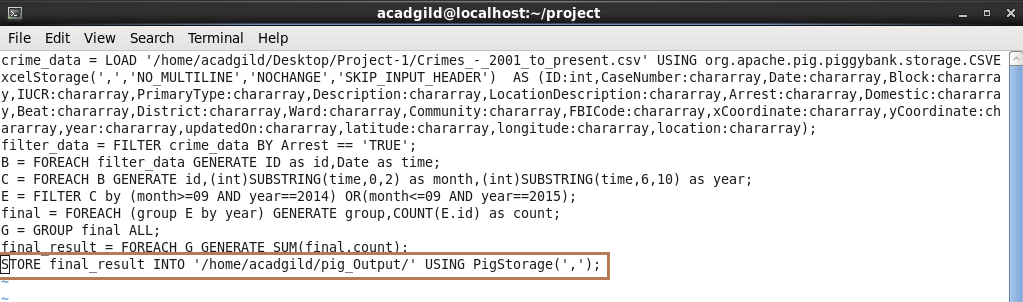
* Finally, after all the transformation and aggregations, values are grouped.



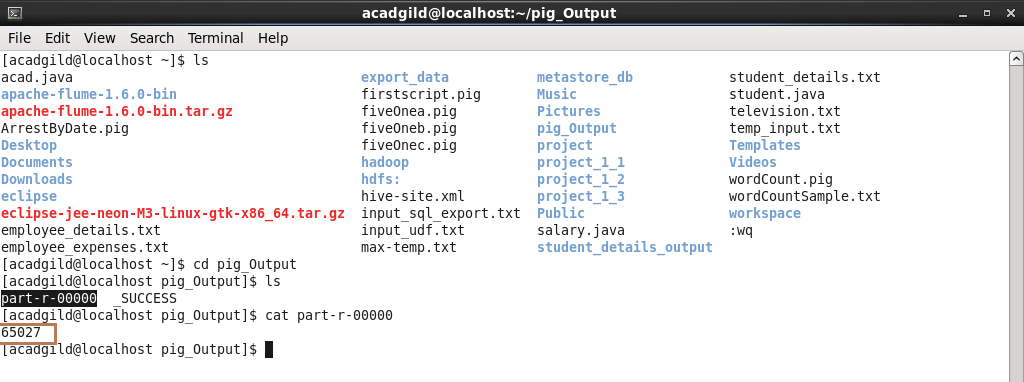
* After grouping the values on the above step, SUM(final.count) is calculated.



* Final result will be stored in an external file in the path specified delimited with commas.



* Below is the result from the stored file which can be viewed using **CAT** command. - ***65027***



*Script file is attached below for reference:*

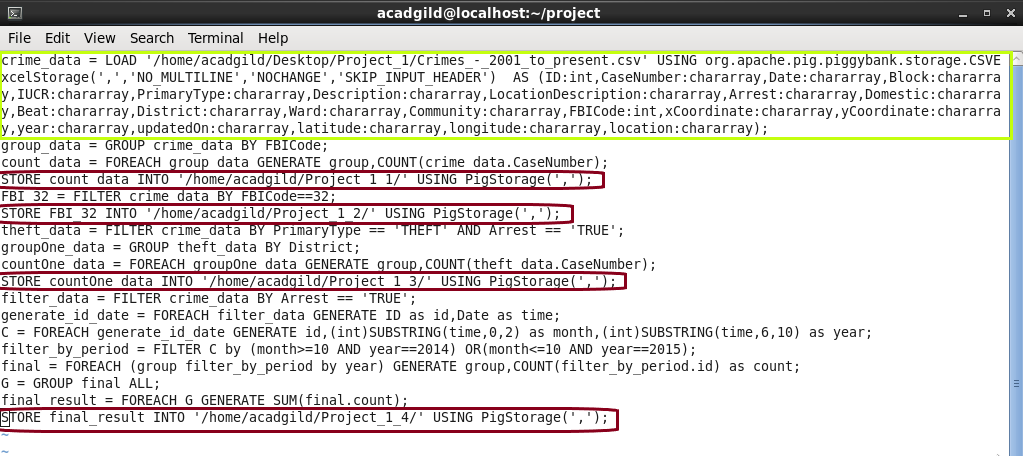


*Output files below:*

**

1. Whole of the above operations can be done in a single file by which we can avoid multiple times loading a same file.

Below is the screenshot of the script.

**

In the above screenshot,

***GREEN –*** shows the file is loaded only once.

***BROWN –*** shows the results are totally stored 4 times for 4 different operations in a single script.