
SIX WEEKS SUMMER TRAINING REPORT

ON

# BIG DATA

**TOPIC:** Crime Recorded (2001-2014)

Submitted by

**Name:** *Vinayak Vedant Srivastava*

**Course:** *B.Tech (Computer Science Engineering)*

Under the guidance of

**Ms. Gurpreet Kaur**

**SHRI RAMSWAROOP MEMORIAL UNIVERSITY, UTTAR PRADESH**

# DECLARATION

I hereby declare that I have completed my six-week summer training at IBM under the guidance of Ms. Gurpreet Kaur. I hereby undertake that the project undertaken by me is the genuine work of mine.

**Name of Student:** Vinayak Vedant Srivastava

**Date:……**

# ACKNOWLEDGEMENT

We would like to express our deepest appreciation to all those who provided us the possibility to complete this report. A special gratitude we give to our 6th semester B.D.E. project supervisor, Ms. Gurpreet Kaur, whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project and especially in writing this report. Furthermore, we could also like to acknowledge with much appreciation her crucial role, in giving the permission to use all required equipment and the necessary materials to complete the task ‘Analysis of AMCAT data’ using Apache Hadoop (Pig and Hive), IBM Cognos insight and R Programming, and gave suggestions about the task.

- **Vinayak Vedant Srivastava**

# TABLE OF CONTENTS

# INTRODUCTION

## What is Big Data?

Big Data is also data but with a huge size. Big Data is a term used to describe a collection of data that is huge in volume and yet growing exponentially with time. In short such data is so large and complex that none of the traditional data management tools are able to store it or process it efficiently.

## Types of Big Data

Big Data could be found in three forms:

1. Structured
2. Unstructured
3. Semi-structured

### Structured

Any data that can be stored, accessed and processed in the form of fixed format is termed as a 'structured' data.

### Unstructured

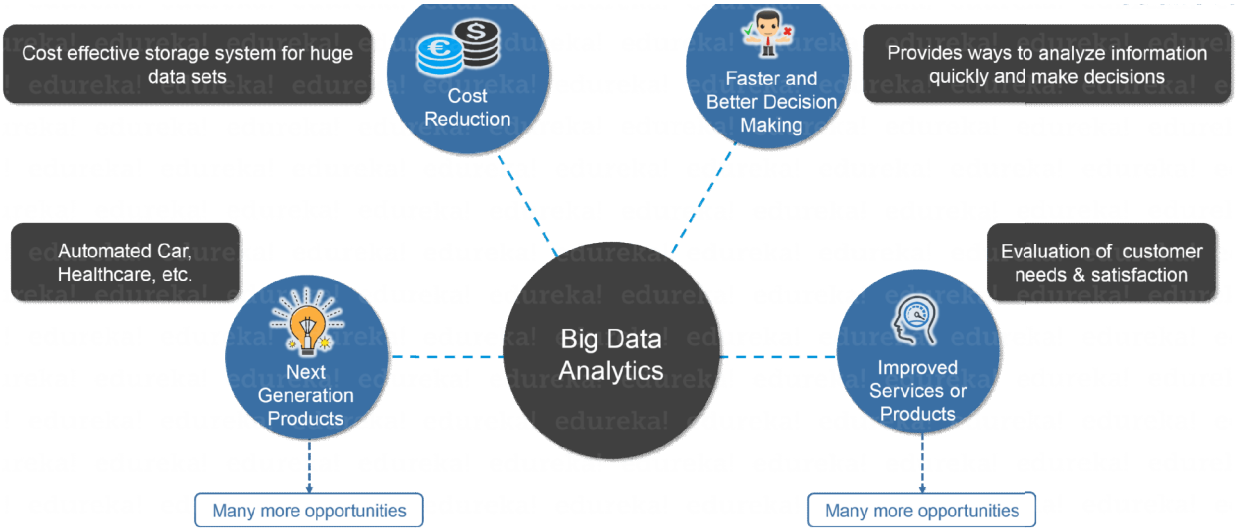
Any data with unknown form or the structure is classified as unstructured data.

### Semi-structured

Semi-structured data can contain both the forms of data. We can see semi-structured data as a structured in form but it is actually not defined with e.g. a table definition in relational DBMS.

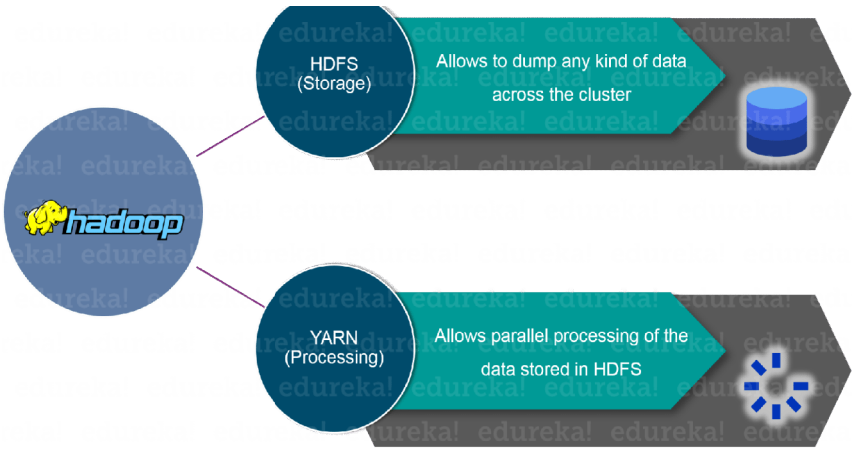
# Characteristics of Big Data

(i) *Volume*
(ii) *Variety*
(iii) *Velocity*
(iv) *Variability*



# WHAT IS HADOOP?

Hadoop is a framework that allows you to first store Big Data in a distributed environment, so that, you can process it parallelly.




## Hadoop Tools:

1. **Hadoop Distributed File System**

The Hadoop Distributed File System (HDFS) is designed to store very large data sets reliably, and to stream those data sets at high bandwidth to user applications. In a large cluster, thousands of servers both host directly attached storage and execute user application tasks. By distributing storage and computation across many servers, the resource can grow with demand while remaining economical at every size. We describe the architecture of HDFS and report on experience using HDFS to manage 40 petabytes of enterprise data at Yahoo.

2. **Hbase**

HBase is a column-oriented database management system that runs on top of HDFS. It is well suited for sparse data sets, which are common in many big data use cases. Unlike relational database systems, HBase does not support a structured query language like SQL; in fact, HBase isn't a relational data store at all. HBase applications are written in Java much like a typical MapReduce application.

3. **HIVE**

The Apache Hive data warehouse software facilitates querying and managing large datasets residing in distributed storage. Hive provides a mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL.

4. **Sqoop**

Sqoop is a tool designed to transfer data between Hadoop and relational databases. You can use Sqoop to import data from a relational database management system (RDBMS) such as MySQL or Oracle into the Hadoop Distributed File System (HDFS), transform the data in Hadoop MapReduce, and then export the data back into an RDBMS.

5. **Pig**

Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The

salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

6. **Map reduce**

Hadoop MapReduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.

# PROJECT: RECORDED CRIME (2001-2014)

This project is to analyze a dataset related to *CRIME happened* in India from 2001-2014. This dataset contains complete information about various aspects of crimes committed in India in different regions along with the types of crime and total number of casualties.

## Purpose

The purpose of this project is to analyze a dataset related to crime data along with casualties happened with the police officers. This dataset contains information about various offence committed in INDIA. It gives detailed analysis of crime and its growth in the country.

## Project Scope

- The objective of this project is to analyze the rate of crimes happened in different regions of the country.
- It is valid for predictive analysis as well as descriptive analysis too.

## Dataset

The dataset contains information about crimes happened in different regions of India. It also includes information about police officer killed/injured in various categories.  This POC contains

Hive, Pig, Map – Reduce and Sqoop queries to analyze the data. The dataset contains 10677 entries.

## Details

This dataset contains complete information about various aspects of crimes happened in India from 2001-2014. Attributes are divided by ‘,’ i.e. they are comma separated (csv). The dataset contain:

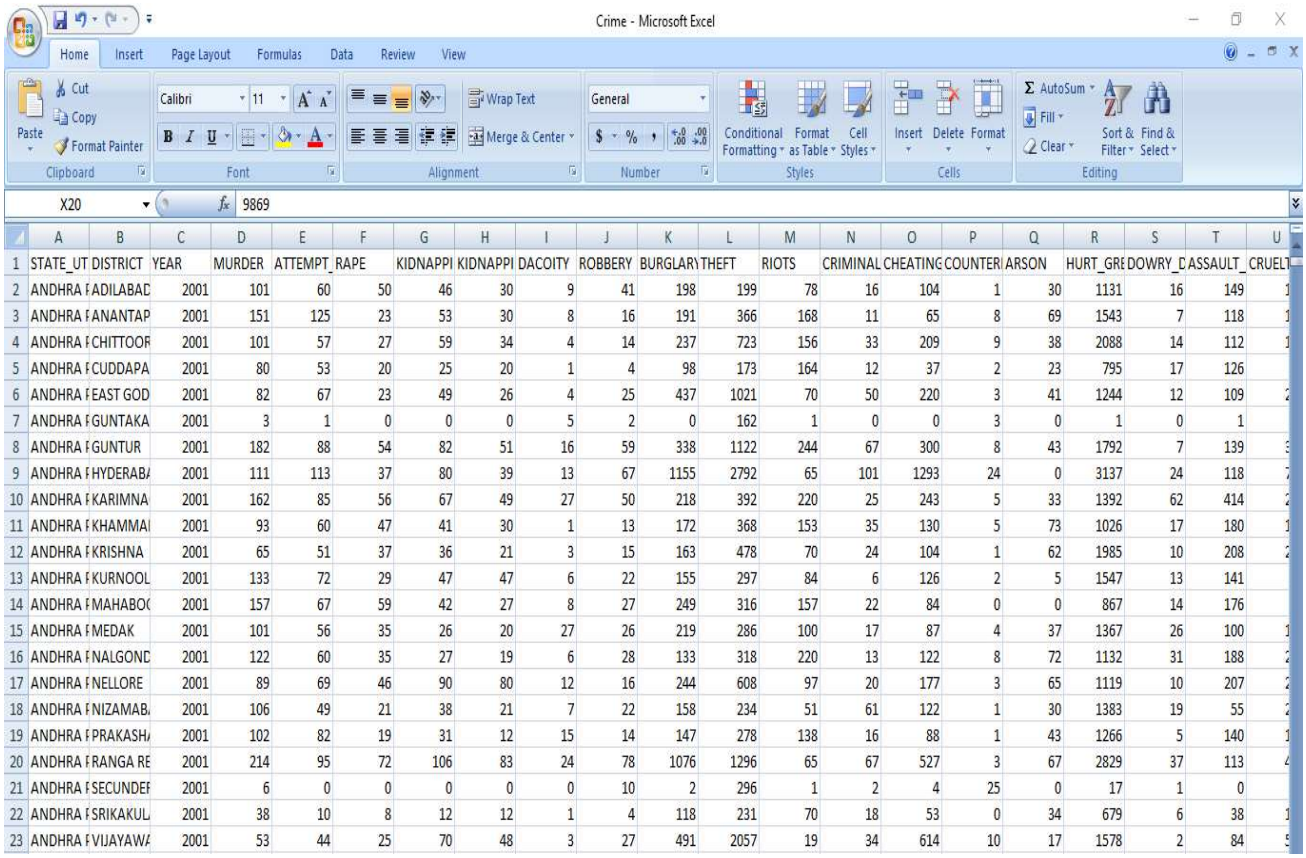
STATE_UT,DISTRICT,YEAR,MURDER,ATTEMPT_TO_MURDER,RAPE,KIDNAPPING,KIDNAPPING_AND_ABDUCTION_OF_WOMEN_AND_GIRLS,DACOITY,ROBBERY,BURGLARY,THEFT,RIOTS,CRIMINAL_BREACH_OF_TRUST,CHEATING,COUNTERFEITING,ARSON,HURT_GREVIOUS_HURT,DOWRY_DEATHS,ASSAULT_ON_WOMEN,CRUELTY_BY_HUSBAND_OR_HIS_RELATIVES,CAUSING_DEATH_BY_NEGLIGENCE,OTHER_IPC_CRIMES,TOTAL_IPC_CRIMES etc…

## Attributes

- STATE_UT - Name of State and Union Territories
- DISTRICT - Name of District
- YEAR - Year in which crime was committed
- MURDER - Number of Murder Victims
- ATTEMPT_TO_MURDER - Number of Attempt to Murder Victims
- RAPE - Number of Rape Victims
- KIDNAPPING - Number of Kidnapping Victims

- KIDNAPPING_AND_ABDUCTION OF_WOMEN_AND_GIRLS - Number of Kidnapping Victims of women and girls
- DACOITY - Number of Dacoity Victims
- ROBBERY - Number of Robbery Victims
- BURGLARY - Number of Burglary Victims
- THEFT - Number of Theft Victims
- RIOTS - Number of Riots Victims
- CRIMINAL_BREACH_OF_TRUST - Victims of Criminal Breach of Trust
- CHEATING - Victims of Cheating
- COUNTERFEITING - Number of Counterfeiting Victims
- ARSON - Number of ARSON Victims
- HURT_GREVIOUS_HURT - Victims of Hurt
- DOWRY_DEATHS - Victims of Dowry deaths
- ASSAULT_ON_WOMEN - Number of Assault happened on Women
- CRUELTY_BY_HUSBAND OR_HIS_RELATIVES - Victims of Cruelty by husband and his family
- CAUSING_DEATH_BY_NEGLIGENCE - Victims of death cause due to Negligence
- OTHER_IPC_CRIMES - Number of Victims of Other IPC Crimes
- TOTAL_IPC_CRIMES - Total Number of Crimes
- AUTO_THEFT - Crimes related to theft of Automotive Vehicles
- GENDER - Male or Female Victim

# Dataset (Excel View):




| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | STATE_UT | DISTRICT | YEAR | MURDER | ATTEMPT_ | RAPE | KIDNAPPI | KIDNAPPI | DACOITY | ROBBERY | BURGLAR\ | THEFT | RIOTS | CRIMINAL | CHEATING | COUNTERI | ARSON | HURT_GRE | DOWRY_D | ASSAULT_ | CRUELT |
| 2 | ANDHRA F | ADILABAD | 2001 | 101 | 60 | 50 | 46 | 30 | 9 | 41 | 198 | 199 | 78 | 16 | 104 | 1 | 30 | 1131 | 16 | 149 | 1 |
| 3 | ANDHRA F | ANANTAP | 2001 | 151 | 125 | 23 | 53 | 30 | 8 | 16 | 191 | 366 | 168 | 11 | 65 | 8 | 69 | 1543 | 7 | 118 | 1 |
| 4 | ANDHRA F | CHITTOOR | 2001 | 101 | 57 | 27 | 59 | 34 | 4 | 14 | 237 | 723 | 156 | 33 | 209 | 9 | 38 | 2088 | 14 | 112 | 1 |
| 5 | ANDHRA F | CUDDAPA | 2001 | 80 | 53 | 20 | 25 | 20 | 1 | 4 | 98 | 173 | 164 | 12 | 37 | 2 | 23 | 795 | 17 | 126 | |
| 6 | ANDHRA F | EAST GOD | 2001 | 82 | 67 | 23 | 49 | 26 | 4 | 25 | 437 | 1021 | 70 | 50 | 220 | 3 | 41 | 1244 | 12 | 109 | 2 |
| 7 | ANDHRA F | GUNTAKA | 2001 | 3 | 1 | 0 | 0 | 0 | 5 | 2 | 0 | 162 | 1 | 0 | 0 | 3 | 0 | 1 | 0 | 1 | |
| 8 | ANDHRA F | GUNTUR | 2001 | 182 | 88 | 54 | 82 | 51 | 16 | 59 | 338 | 1122 | 244 | 67 | 300 | 8 | 43 | 1792 | 7 | 139 | 3 |
| 9 | ANDHRA F | HYDERABA | 2001 | 111 | 113 | 37 | 80 | 39 | 13 | 67 | 1155 | 2792 | 65 | 101 | 1293 | 24 | 0 | 3137 | 24 | 118 | 7 |
| 10 | ANDHRA F | KARIMNA | 2001 | 162 | 85 | 56 | 67 | 49 | 27 | 50 | 218 | 392 | 220 | 25 | 243 | 5 | 33 | 1392 | 62 | 414 | 2 |
| 11 | ANDHRA F | KHAMMAI | 2001 | 93 | 60 | 47 | 41 | 30 | 1 | 13 | 172 | 368 | 153 | 35 | 130 | 5 | 73 | 1026 | 17 | 180 | 1 |
| 12 | ANDHRA F | KRISHNA | 2001 | 65 | 51 | 37 | 36 | 21 | 3 | 15 | 163 | 478 | 70 | 24 | 104 | 1 | 62 | 1985 | 10 | 208 | 2 |
| 13 | ANDHRA F | KURNOOL | 2001 | 133 | 72 | 29 | 47 | 47 | 6 | 22 | 155 | 297 | 84 | 6 | 126 | 2 | 5 | 1547 | 13 | 141 | |
| 14 | ANDHRA F | MAHABO( | 2001 | 157 | 67 | 59 | 42 | 27 | 8 | 27 | 249 | 316 | 157 | 22 | 84 | 0 | 0 | 867 | 14 | 176 | |
| 15 | ANDHRA F | MEDAK | 2001 | 101 | 56 | 35 | 26 | 20 | 27 | 26 | 219 | 286 | 100 | 17 | 87 | 4 | 37 | 1367 | 26 | 100 | 1 |
| 16 | ANDHRA F | NALGONC | 2001 | 122 | 60 | 35 | 27 | 19 | 6 | 28 | 133 | 318 | 220 | 13 | 122 | 8 | 72 | 1132 | 31 | 188 | 2 |
| 17 | ANDHRA F | NELLORE | 2001 | 89 | 69 | 46 | 90 | 80 | 12 | 16 | 244 | 608 | 97 | 20 | 177 | 3 | 65 | 1119 | 10 | 207 | 2 |
| 18 | ANDHRA F | NIZAMAB | 2001 | 106 | 49 | 21 | 38 | 21 | 7 | 22 | 158 | 234 | 51 | 61 | 122 | 1 | 30 | 1383 | 19 | 55 | 2 |
| 19 | ANDHRA F | PRAKASH | 2001 | 102 | 82 | 19 | 31 | 12 | 15 | 14 | 147 | 278 | 138 | 16 | 88 | 1 | 43 | 1266 | 5 | 140 | 1 |
| 20 | ANDHRA F | RANGA RE | 2001 | 214 | 95 | 72 | 106 | 83 | 24 | 78 | 1076 | 1296 | 65 | 67 | 527 | 3 | 67 | 2829 | 37 | 113 | 4 |
| 21 | ANDHRA F | SECUNDEI | 2001 | 6 | 0 | 0 | 0 | 0 | 0 | 10 | 2 | 296 | 1 | 2 | 4 | 25 | 0 | 17 | 1 | 0 | |
| 22 | ANDHRA F | SRIKAKUL | 2001 | 38 | 10 | 8 | 12 | 12 | 1 | 4 | 118 | 231 | 70 | 18 | 53 | 0 | 34 | 679 | 6 | 38 | 1 |
| 23 | ANDHRA F | VIJAYAWA | 2001 | 53 | 44 | 25 | 70 | 48 | 3 | 27 | 491 | 2057 | 19 | 34 | 614 | 10 | 17 | 1578 | 2 | 84 | 5 |

# PROBLEM STATEMENT

1. Total Murder being recorded in Uttar Pradesh during the year 2013.
2. Number of Rape victims found in various age groups.
3. Top 3 categories of Policemen killed or injured.
4. Categorization of Murder victims on the basis of Gender.
5. Year having maximum number of casualties (Policemen killed).
6. Total number of Dacoity happened in each year.
7. Top 5 states of Dowry Deaths during the year 2014.
8. Top 3 category of Auto Theft during year 2011.
9. Year wise Total Rape case reported v/s Total rape victims.
10. State wise percentage of IPC crimes recorded.

## Query 1:- Total Murder being recorded in Uttar Pradesh during the year 2013.

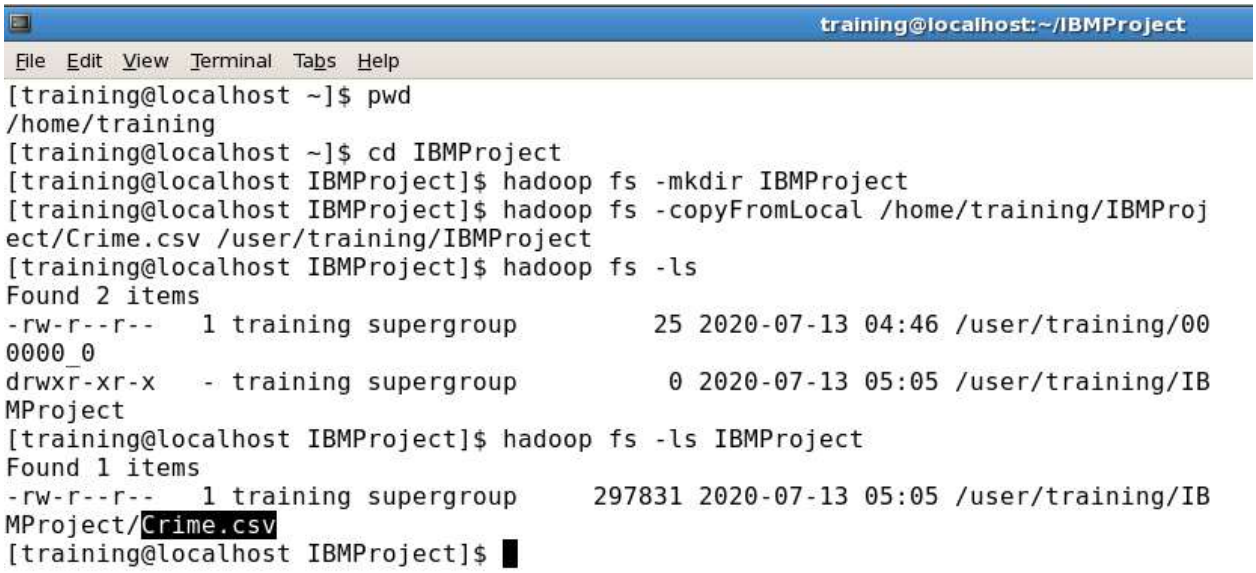
### CODING

Step i. Create a new directory with name *IBMProject*

Step ii. Transfer data into Hadoop Distributed File System(HDFS)

- ~ In order to perform queries on HIVE we need to transfer the data to HDFS

```
[training@localhost ~]$ pwd
[training@localhost ~]$ cd IBMProject
[training@localhost ~]$ hadoop fs -mkdir IBMProject
[training@localhost ~]$ hadoop fs -copyFromLocal
/home/training/IBMProject/Crime.csv /user/training/IBMProject
```




Step iii. Open Hive shell

Step iv. Create a new database with name *crimedata*

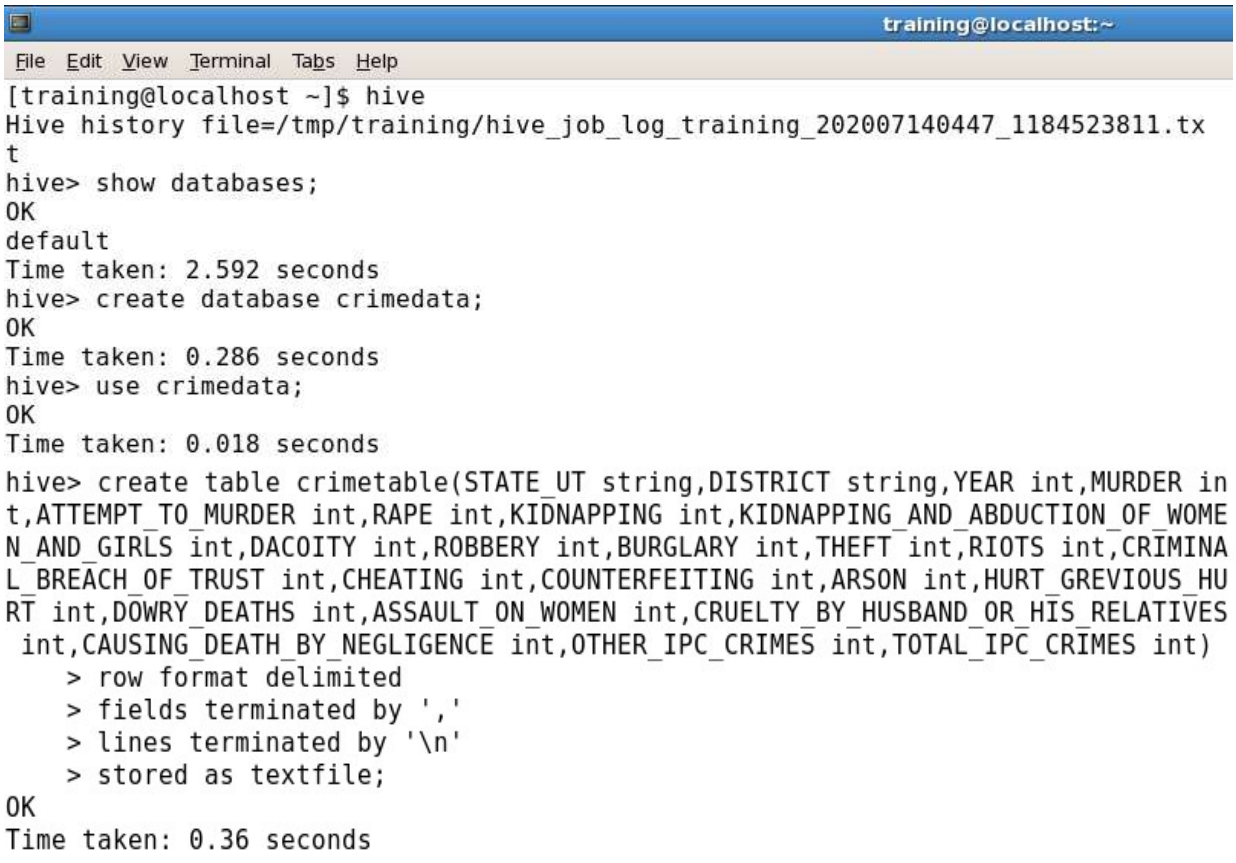
Step v. Create a new table name *crimetable*

- ~ Use the above database i.e. crimedata for creation of table.

Step vi. Load the data into table crimetable

```
[training@localhost ~]$ hive
hive> show databases;
hive> create database crimedata;
hive> use crimedata;
hive> create table crimetable(STATE_UT string,DISTRICT
string,YEAR int,MURDER int,ATTEMPT_TO_MURDER
int,RAPE int,KIDNAPPING
int,KIDNAPPING_AND_ABDUCTION_OF_WOMEN_AND_
GIRLS int,DACOITY int,ROBBERY int,BURGLARY
int,THEFT int,RIOTS int,CRIMINAL_BREACH_OF_TRUST
int,CHEATING int,COUNTERFEITING int,ARSON
```

```
int,HURT_GREVIOUS_HURT int,DOWRY_DEATHS
int,ASSAULT_ON_WOMEN
int,CRUELTY_BY_HUSBAND_OR_HIS_RELATIVES
int,CAUSING_DEATH_BY_NEGLIGENCE
int,OTHER_IPC_CRIMES int,TOTAL_IPC_CRIMES int)
   > row format delimited
   > fields terminated by ','
   > lines terminated by '\n'
   > stored as textfile;
hive> load data inpath '/user/training/IBMProject/Crime.csv'
into table crimetable;
hive> select * from crimetable;
```



```
hive> load data inpath '/user/training/IBMProject/Crime.csv' into table crimetable;
Loading data to table crimedata.crimetable
OK
Time taken: 0.442 seconds
hive> select * from crimetable;
```

Step vii. Again create a new table with name *murdertable*

~ Filtering of data is done in this phase

Step viii. Load data into murdertable by accessing data from crimedata

```
hive> create table murdertable(STATE_UT string,MURDER int)
   > row format delimited
   > fields terminated by ',’
   > lines terminated by '\n'
   > stored as textfile;
hive> insert overwrite table murdertable select
STATE_UT,MURDER from crimetable where YEAR='2013';
hive> select sum(MURDER) from murdertable where
STATE_UT='UTTAR PRADESH';
```

```
LAKSHADWEEP     Lakshadweep     2014    0       0       1       0       0       0       0
5       1       0       1       0       0       30      74
LAKSHADWEEP     Total   2014    0       0       1       0       0       0       0       10
1       0       1       0       0       30      74
PUDUCHERRY      Karaikal        2014    6       1       3       1       0       1       1
0       8       0       12      1       44      322     461
PUDUCHERRY      Puducherry      2014    19      14      7       12      1       2       5
1       97      1       20      3       165     1014    1890
PUDUCHERRY      Total   2014    25      15      10      13      1       3       6       61
105     1       32      4       209     1336    2351
Time taken: 0.582 seconds
hive> create table murdertable(STATE_UT string,MURDER int)
    > row format delimited
    > fields terminated by ','
    > lines terminated by '\n'
    > stored as textfile;
OK
Time taken: 0.042 seconds
```

```
hive> insert overwrite table murdertable select STATE_UT,MURDER from crimetable where YEAR='2013';
Total MapReduce jobs = 2
Launching Job 1 out of 2
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_202007140415_0005, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_202007140415_0005
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:8021 -kill job_202007140415_0005
2020-07-14 05:46:01,568 Stage-1 map = 0%,  reduce = 0%
2020-07-14 05:46:03,586 Stage-1 map = 100%,  reduce = 0%
Ended Job = job_202007140415_0005
Ended Job = -230666345, job is filtered out (removed at runtime).
Moving data to: hdfs://localhost/tmp/hive-training/hive_2020-07-14_05-45-59_133_6982992251250274676/-ext-10000
Loading data to table crimedata.murdertable
Deleted hdfs://localhost/user/hive/warehouse/crimedata.db/murdertable
Table crimedata.murdertable stats: [num_partitions: 0, num_files: 1, num_rows: 0, total_size: 11338]
822 Rows loaded to murdertable
OK
Time taken: 5.864 seconds
```

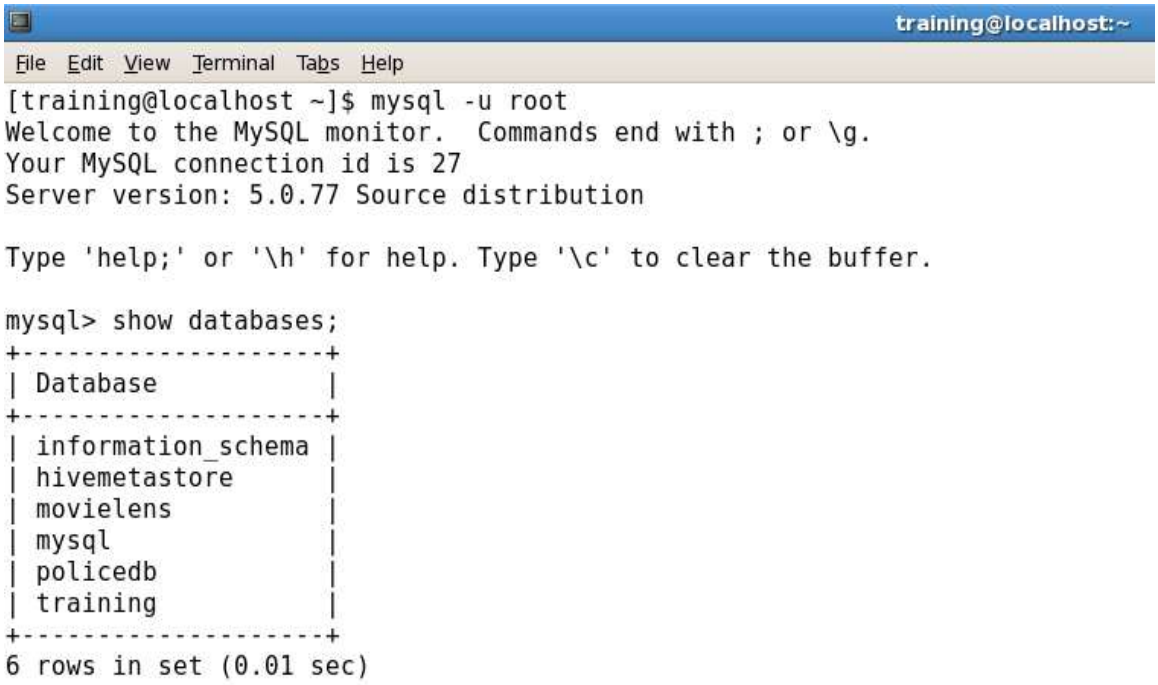
```
hive> select sum(MURDER) from murdertable where STATE_UT='UTTAR PRADESH';
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202007140415_0006, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_202007140415_0006
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:8021 -kill job_202007140415_0006
2020-07-14 05:47:43,900 Stage-1 map = 0%,  reduce = 0%
2020-07-14 05:47:46,014 Stage-1 map = 100%,  reduce = 0%
2020-07-14 05:47:54,343 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_202007140415_0006
OK
10094
Time taken: 14.989 seconds
```

## Query 2:- Number of Rape victims found in various age groups.

### CODING

Step i. Open mysql

```
[training@localhost ~]$ mysql -u root
mysql> show databases;
```




Step ii. Create a new database with name *rapedb*

Step iii. Create an empty table with name rapetb

~ Use the previous created database i.e. rapetb

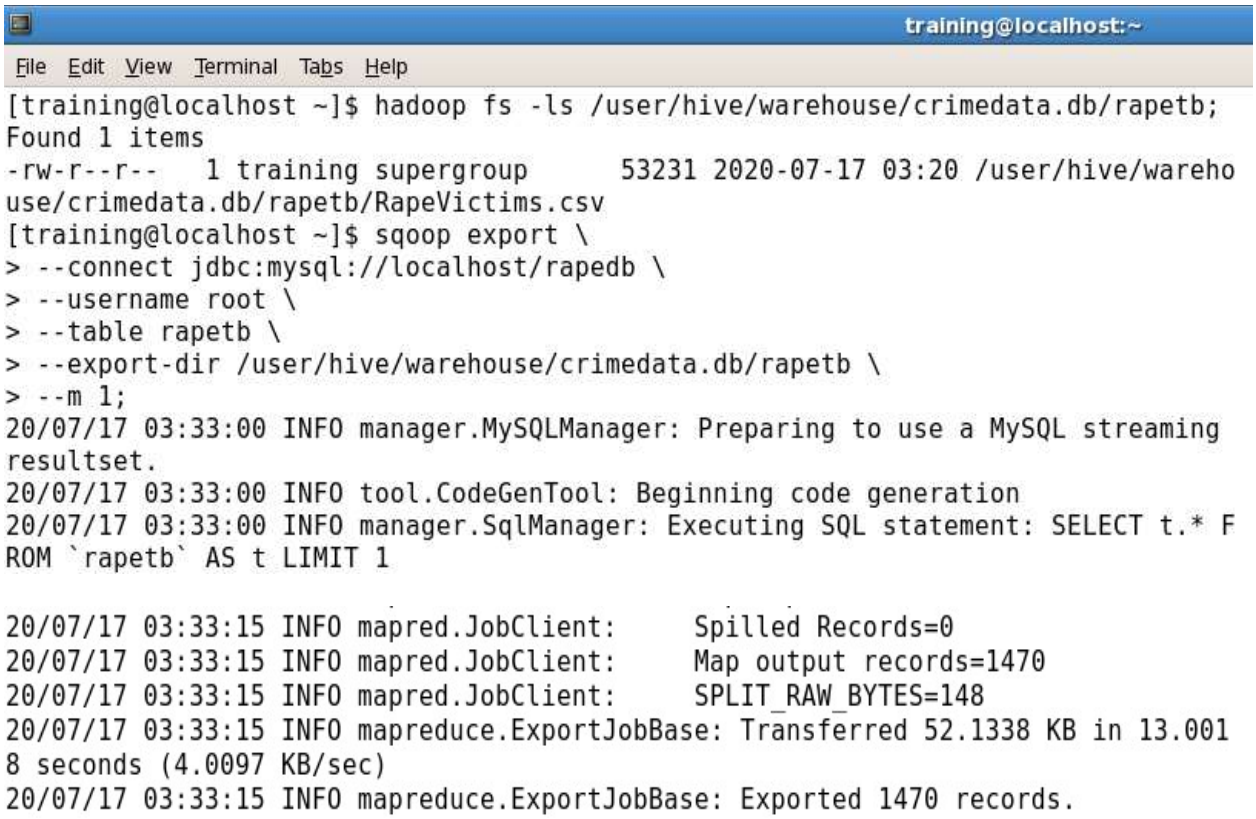
```
mysql> create database rapedb;
mysql> use rapedb;
mysql> create table rapetb(AREA_NAME
varchar(30),YEAR int,RAPE_CASES_REPORTED
```

```
int,VICTIMS_BETWEEN_10_14_YRS
int,VICTIMS_BETWEEN_14_18_YRS
int,VICTIMS_BETWEEN_18_30_YRS
int,VICTIMS_BETWEEN_30_50_YRS
int,VICTIMS_ABOVE_50_YRS
int,VICTIMS_OF_RAPE_TOTAL int);
mysql> quit
```

```
mysql> create database rapedb;
Query OK, 1 row affected (0.00 sec)

mysql> use rapedb;
Database changed
mysql> create table rapetb(AREA_NAME varchar(30),YEAR int,RAPE_CASES_REPORTED in
t,VICTIMS_BETWEEN_10_14_YRS int,VICTIMS_BETWEEN_14_18_YRS int,VICTIMS_BETWEEN_18
_30_YRS int,VICTIMS_BETWEEN_30_50_YRS int,VICTIMS_ABOVE_50_YRS int,VICTIMS_OF_RA
PE_TOTAL int);
Query OK, 0 rows affected (0.06 sec)

mysql> quit
Bye
```

Step iv. Now exit mysql. Export the data present on hadoop(dataset) using SQOOP into MYSQL in table *rapetb*

```
[training@localhost ~]$ hadoop fs -ls
/user/hive/warehouse/crimedata.db/rapetb;
[training@localhost ~]$ sqoop export \
> --connect jdbc:mysql://localhost/rapedb \
> --username root \
> --table rapetb \
> --export-dir /user/hive/warehouse/crimedata.db/rapetb \
> --m 1;
```




Step v. Open mysql and use same database i.e. *rapedb*

Step vi. Perform SUM function over the data present in *rapetb*

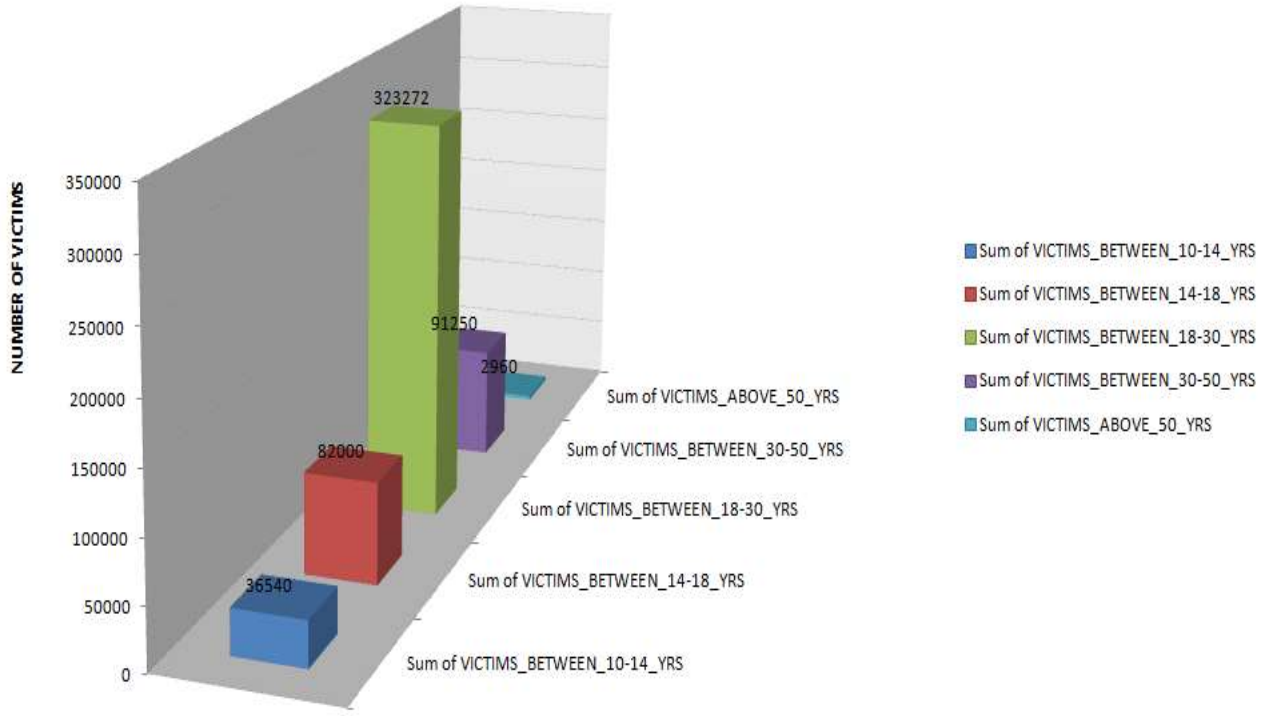
```
[training@localhost ~]$ mysql -u root
mysql> use rapedb;
mysql> select * from rapetb limit 10;
mysql> select sum(VICTIMS_BETWEEN_10_14_YRS),
sum(VICTIMS_BETWEEN_14_18_YRS),
sum(VICTIMS_BETWEEN_18_30_YRS),
sum(VICTIMS_BETWEEN_30_50_YRS),
sum(VICTIMS_ABOVE_50_YRS) from rapetb;
```

```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
[training@localhost ~]$ mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 30
Server version: 5.0.77 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use rapedb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from rapetb limit 10;
+---------------------------+------+---------------------+----------------------
-----+---------------------------+--------------------------+-----------------
---------+----------------------+----------------------+
| AREA_NAME                 | YEAR | RAPE_CASES_REPORTED | VICTIMS_BETWEEN_10_14
_YRS | VICTIMS_BETWEEN_14_18_YRS | VICTIMS_BETWEEN_18_30_YRS | VICTIMS_BETWEEN_3
0_50_YRS | VICTIMS_ABOVE_50_YRS | VICTIMS_OF_RAPE_TOTAL |
mysql> select sum(VICTIMS_BETWEEN_10_14_YRS), sum(VICTIMS_BETWEEN_14_18_YRS), su
m(VICTIMS_BETWEEN_18_30_YRS), sum(VICTIMS_BETWEEN_30_50_YRS), sum(VICTIMS_ABOVE_
50_YRS) from rapetb;
+-------------------------------+-------------------------------+-------------
------------------+--------------------------------+---------------------------
+
| sum(VICTIMS_BETWEEN_10_14_YRS) | sum(VICTIMS_BETWEEN_14_18_YRS) | sum(VICTIMS_
BETWEEN_18_30_YRS) | sum(VICTIMS_BETWEEN_30_50_YRS) | sum(VICTIMS_ABOVE_50_YRS)
|
+-------------------------------+-------------------------------+-------------
------------------+--------------------------------+---------------------------
+
|                         36540 |                              82000 |
           323272 |                             91250 |                         2960
|
+-------------------------------+-------------------------------+-------------
------------------+--------------------------------+---------------------------
+
1 row in set (0.01 sec)
```

## VISUALIZATION

This column graph shows the graph plotted between VICTIMS OF VARIOUS AGE GROUPS to NUMBER OF VICTIMS.

# AGE WISE VICTIMS CHART GRAPH



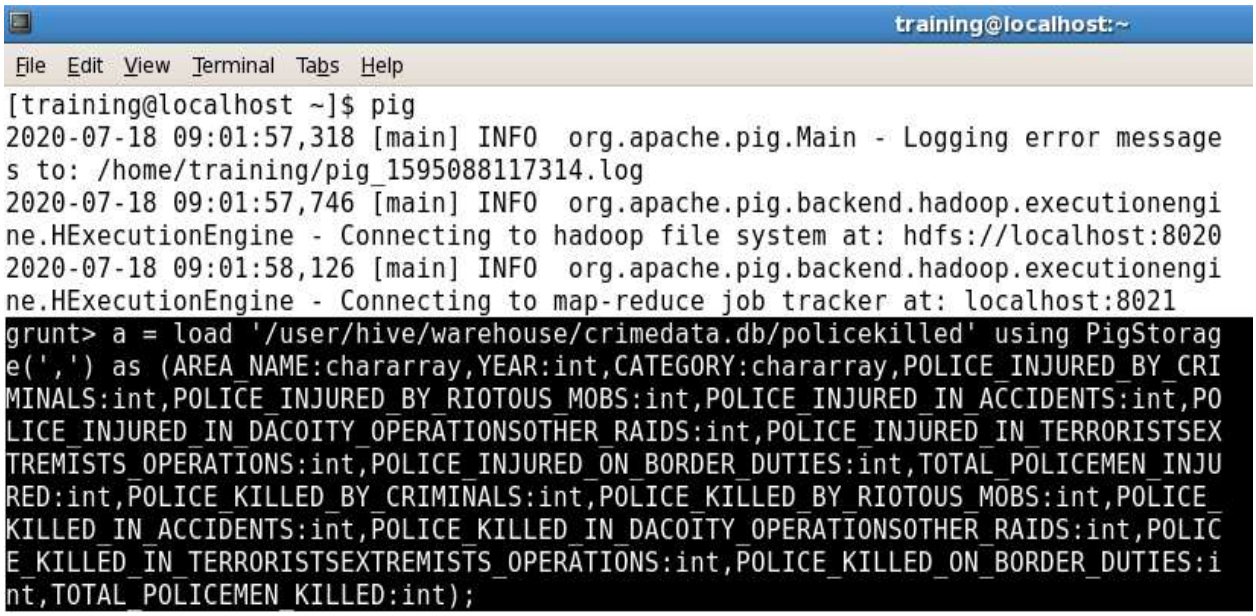
## Query 3:- Top 3 categories of Policemen killed or injured.

**CODING**

Step i. Open Pig in MapReduce mode

Step ii. Create a bag with name ***'a'*** and load the data present on Hadoop (dataset) into bag ***'a'***

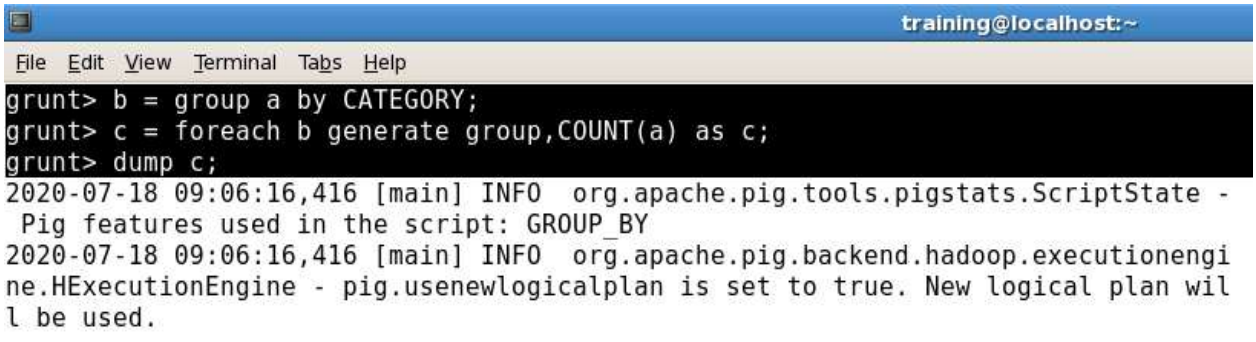
```
[training@localhost ~]$ pig
grunt> a = load
'/user/hive/warehouse/crimedata.db/policekilled' using
PigStorage(',') as
(AREA_NAME:chararray,YEAR:int,CATEGORY:chararray,P
OLICE_INJURED_BY_CRIMINALS:int,POLICE_INJURED_
BY_RIOTOUS_MOBS:int,POLICE_INJURED_IN_ACCIDEN
TS:int,POLICE_INJURED_IN_DACOITY_OPERATIONSOT
HER_RAIDS:int,POLICE_INJURED_IN_TERRORISTSEXT
REMISTS_OPERATIONS:int,POLICE_INJURED_ON_BOR
DER_DUTIES:int,TOTAL_POLICEMEN_INJURED:int,POLI
CE_KILLED_BY_CRIMINALS:int,POLICE_KILLED_BY_RIO
TOUS_MOBS:int,POLICE_KILLED_IN_ACCIDENTS:int,PO
LICE_KILLED_IN_DACOITY_OPERATIONSOTHER_RAID
S:int,POLICE_KILLED_IN_TERRORISTSEXTREMISTS_OP
ERATIONS:int,POLICE_KILLED_ON_BORDER_DUTIES:int
,TOTAL_POLICEMEN_KILLED:int);
```




Step iii. Now perform GROUP operation by referencing with a column CATEGORY in a bag ***'b'***

Step iv. Perform COUNT operation on ***'a'*** in bag ***'c'***

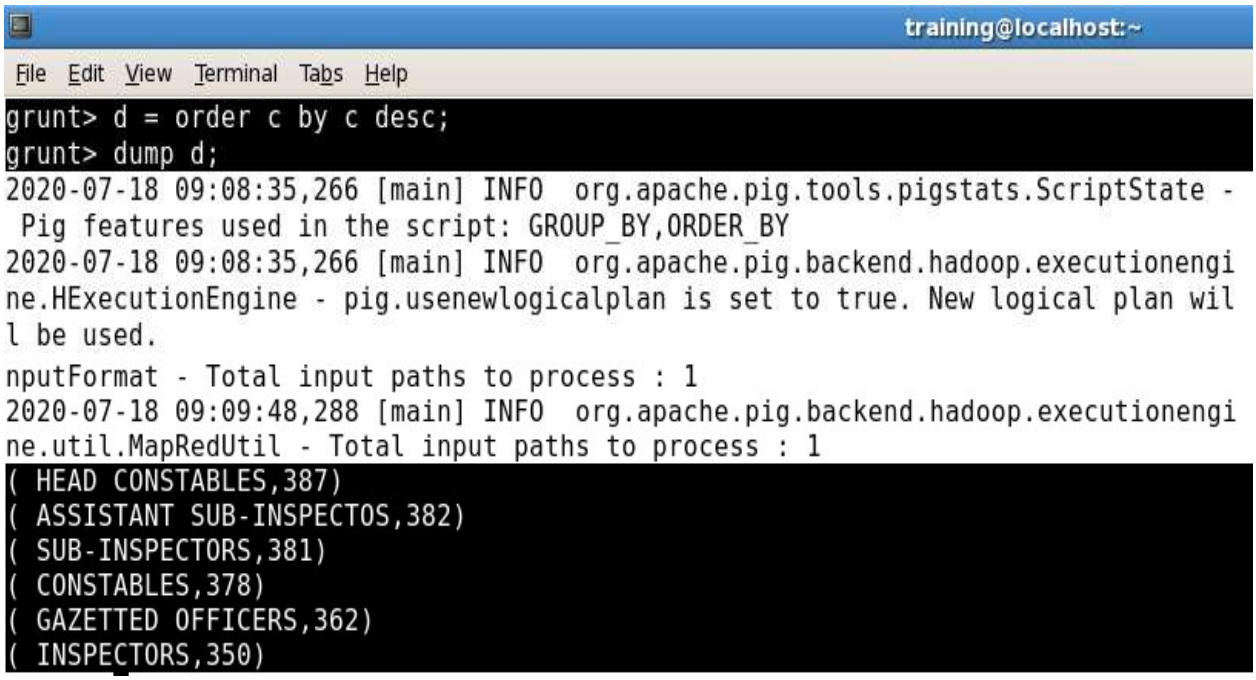
```
grunt> b = group a by CATEGORY;
grunt> c = foreach b generate group,COUNT(a) as c;
grunt> dump c;
```



```
2020-07-18 09:06:41,665 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2020-07-18 09:06:41,665 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
( CONSTABLES,378)
( INSPECTORS,350)
( SUB-INSPECTORS,381)
( HEAD CONSTABLES,387)
( GAZETTED OFFICERS,362)
( ASSISTANT SUB-INSPECTOS,382)
```

Step v. Order bag ***'c'*** as Descending Order

```
grunt> d = order c by c desc;
grunt> dump d;
```



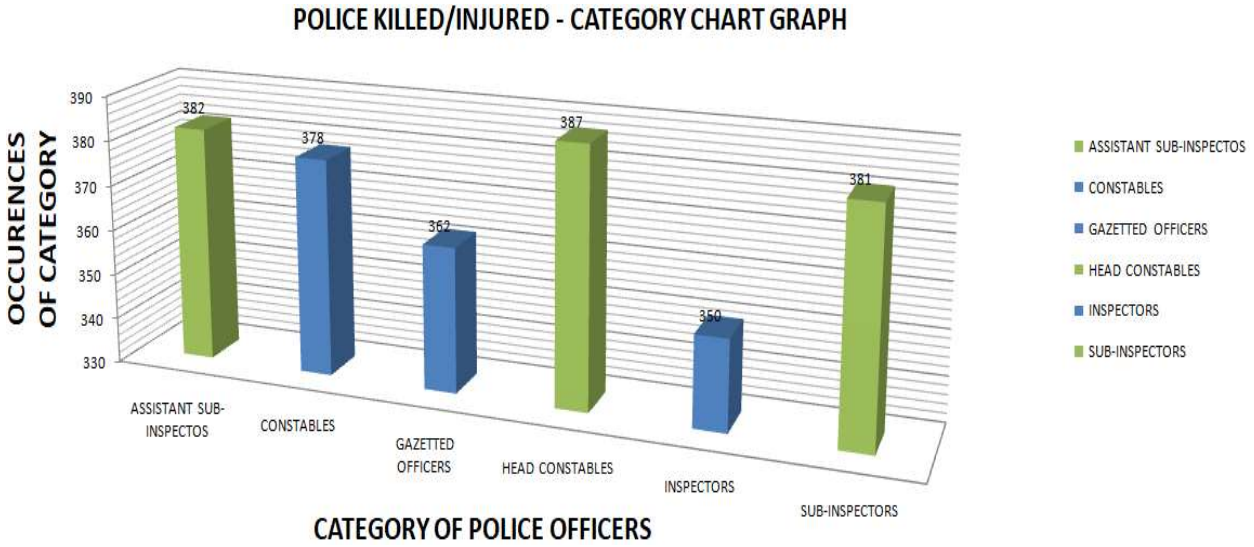

Step vi. Apply LIMIT operation as limit 3.

```
grunt> e = limit d 3;
grunt> dump e;
```

```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
grunt> e = limit d 3;
grunt> dump e;
2020-07-18 09:11:34,367 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: GROUP_BY,ORDER_BY,LIMIT
2020-07-18 09:11:34,367 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - pig.usenewlogicalplan is set to true. New logical plan wil
ne.mapReduceLayer.MapReduceLauncher - Success!
2020-07-18 09:12:42,707 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2020-07-18 09:12:42,707 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
( HEAD CONSTABLES,387)
( ASSISTANT SUB-INSPECTOS,382)
( SUB-INSPECTORS,381)
```

## VISUALIZATION

This column graph shows the graph plotted between CATEGORY OF POLICE OFFICERS and their OCCURENCES. The ***THREE BAR*** with ***GREEN COLOR*** shows the top 3 category in the graph.



## Query 4:- Categorization of Murder victims on the basis of Gender.

### CODING

Step i. Open Hive and use previously created database i.e. *crimedata*

Step ii. Create a new table with name *murdervictim*

```
[training@localhost ~]$ hive
hive> use crimedata;
hive> create table murdervictimtb(AREA_NAME string,YEAR
int,GENDER string,VICTIMS_UPTO_10_YRS
int,VICTIMS_UPTO_10_15_YRS
int,VICTIMS_UPTO_15_18_YRS
int,VICTIMS_UPTO_18_30_YRS
int,VICTIMS_UPTO_30_50_YRS
int,VICTIMS_ABOVE_50_YRS int,VICTIMS_TOTAL int)
    > row format delimited
    > fields terminated by ','
    > lines terminated by '\n'
    > stored as textfile;
```

```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
[training@localhost ~]$ hive
Hive history file=/tmp/training/hive_job_log_training_202007172232_1758519624.tx
t
hive> show databases;
OK
crimedata
default
Time taken: 2.982 seconds
hive> use crimedata;
OK
Time taken: 0.025 seconds
hive> create table murdervictimtb(AREA_NAME string,YEAR int,GENDER string,VICTIM
S_UPTO_10_YRS int,VICTIMS_UPTO_10_15_YRS int,VICTIMS_UPTO_15_18_YRS int,VICTIMS_
UPTO_18_30_YRS int,VICTIMS_UPTO_30_50_YRS int,VICTIMS_ABOVE_50_YRS int,VICTIMS_T
OTAL int)
    > row format delimited
    > fields terminated by ','
    > lines terminated by '\n'
    > stored as textfile;
OK
Time taken: 0.747 seconds
```

Step iii. Load data into the table from local file system (location of dataset)

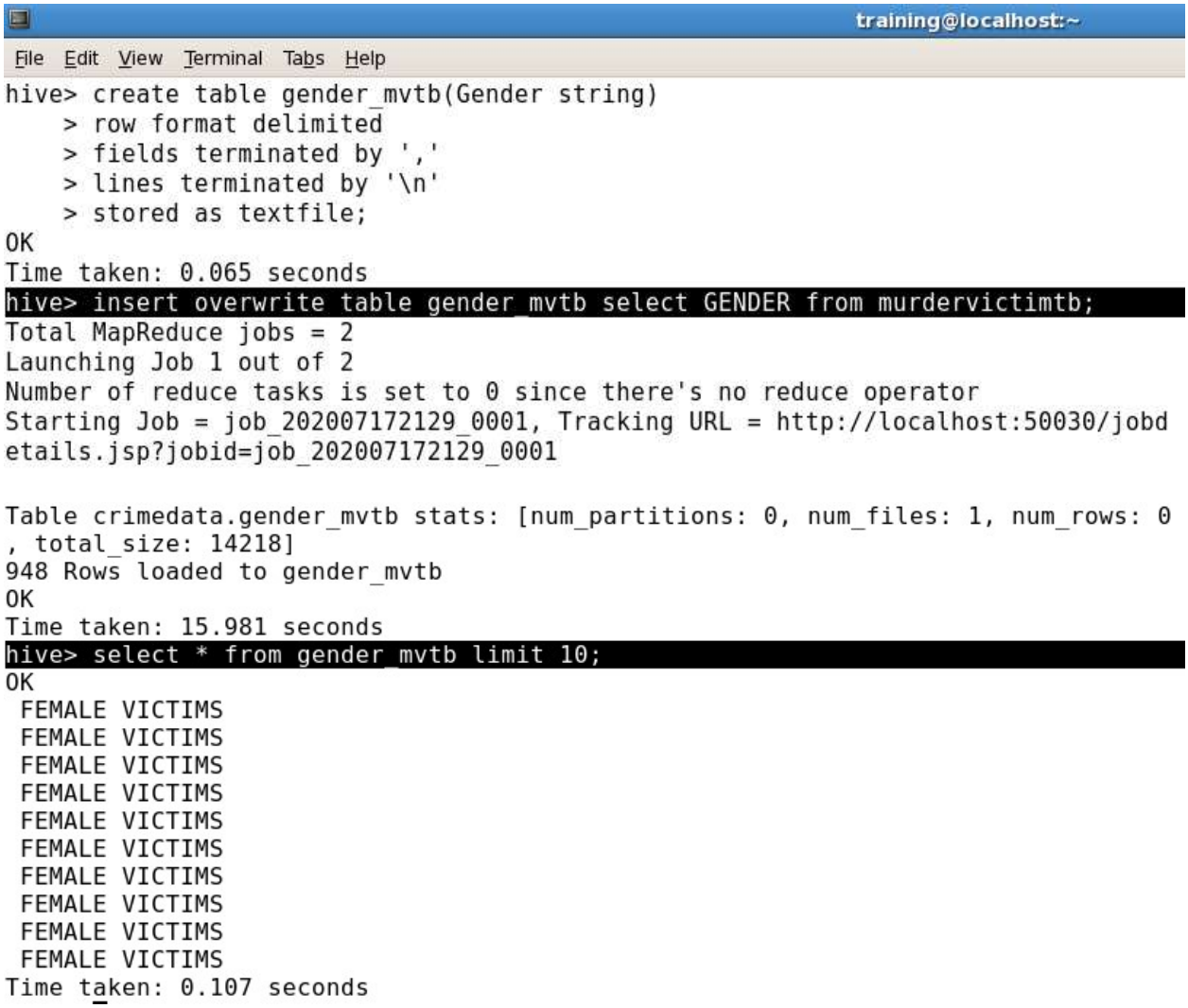
hive> load data local inpath '/home/training/IBMProject/MurderVictim.csv' into table murdervictimtb;

```
hive> load data local inpath '/home/training/IBMProject/MurderVictim.csv' into t
able murdervictimtb;
Copying data from file:/home/training/IBMProject/MurderVictim.csv
Copying file: file:/home/training/IBMProject/MurderVictim.csv
Loading data to table crimedata.murdervictimtb
OK
Time taken: 0.541 seconds
```

Step iv. Create a new table with name *gender_mvtb*

Step v. Load data of only GENDER column from table *murdervictim*

```
hive> create table gender_mvtb(Gender string)
    > row format delimited
    > fields terminated by ','
    > lines terminated by '\n'
    > stored as textfile;
hive> insert overwrite table gender_mvtb select GENDER
from murdervictimtb;
```

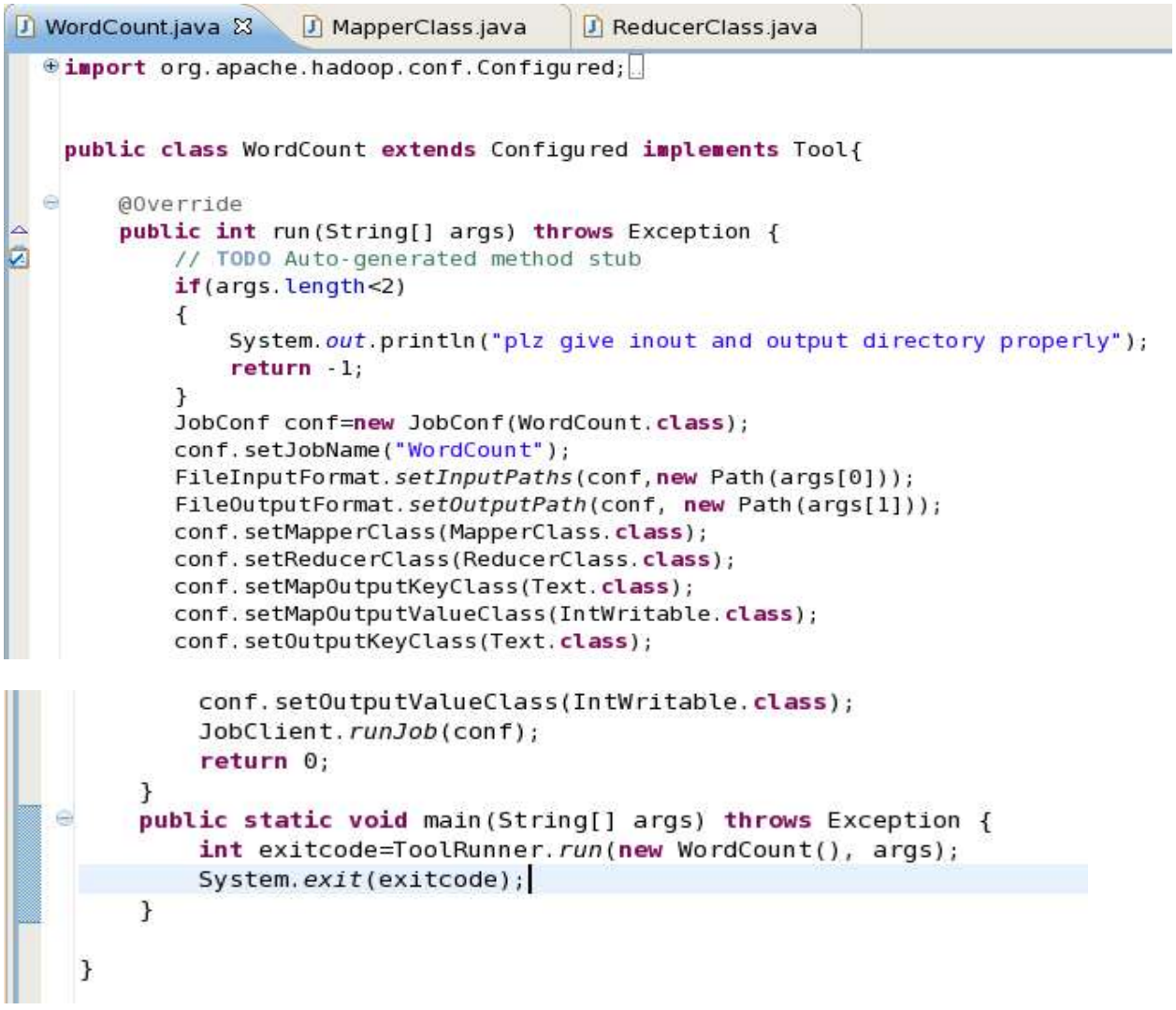



Step vi. Now minimize the terminal and open ECLIPSE

Step vii. Create a new java project with name *WordCount.*

~ The project contains three classes i.e. WordCount, Mapper, Reducer Class along with codes for MapReduce.

## Screenshot: WordCount Class



```
WordCount.java   MapperClass.java   ReducerClass.java

import org.apache.hadoop.conf.Configured;


public class WordCount extends Configured implements Tool{

    @Override
    public int run(String[] args) throws Exception {
        // TODO Auto-generated method stub
        if(args.length<2)
        {
            System.out.println("plz give inout and output directory properly");
            return -1;
        }
        JobConf conf=new JobConf(WordCount.class);
        conf.setJobName("WordCount");
        FileInputFormat.setInputPaths(conf,new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        conf.setMapperClass(MapperClass.class);
        conf.setReducerClass(ReducerClass.class);
        conf.setMapOutputKeyClass(Text.class);
        conf.setMapOutputValueClass(IntWritable.class);
        conf.setOutputKeyClass(Text.class);

          conf.setOutputValueClass(IntWritable.class);
          JobClient.runJob(conf);
          return 0;
    }
    public static void main(String[] args) throws Exception {
        int exitcode=ToolRunner.run(new WordCount(), args);
        System.exit(exitcode);
    }

}
```

## Screenshot: Mapper Class

WordCount.java | MapperClass.java | ReducerClass.java

```
public class MapperClass extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWritable>{

    @Override
    public void map(LongWritable key, Text value,
            OutputCollector<Text, IntWritable> output, Reporter arg3)
            throws IOException {
        // TODO Auto-generated method stub
        String s=value.toString();
        for(String word:s.split(" "))
        {
            if(word.length()>0)
            {
                output.collect(new Text(word), new IntWritable(1));
            }
        }

    }


}
```

## Screenshot: Reducer Class

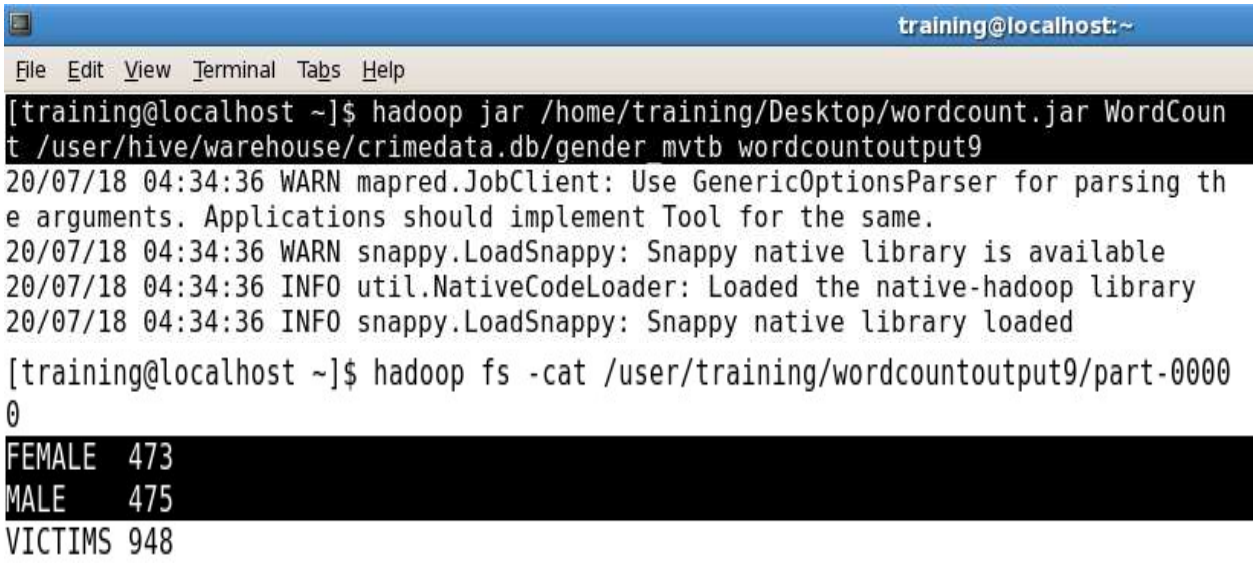
WordCount.java | MapperClass.java | ReducerClass.java

```
import java.io.IOException;
import java.util.Iterator;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;


public class ReducerClass extends MapReduceBase implements Reducer<Text,IntWritable,Text,IntWritable>{

    @Override
    public void reduce(Text key, Iterator<IntWritable> value,
            OutputCollector<Text, IntWritable> output, Reporter arg3)
            throws IOException {
        // TODO Auto-generated method stub
        int count=0;
        while(value.hasNext())
        {
            IntWritable i=value.next();

            count+=i.get();
        }
        output.collect(key,new IntWritable(count));
    }


}
```

Step viii. Export that project and create a jar file with name *wordcount.jar*

Step ix. Use MapReduce commands in terminal to perform wordcount over the data present in gender_mvtb

```
[training@localhost ~]$ hadoop jar
/home/training/Desktop/wordcount.jar WordCount
/user/hive/warehouse/crimedata.db/gender_mvtb
wordcountoutput9
[training@localhost ~]$ hadoop fs -cat
/user/training/wordcountoutput9/part-00000
```




## VISUALIZATION

This pie-chart graph shows the graph plotted between numbers of MALE &FEMALE VICTIMS in murder crime.

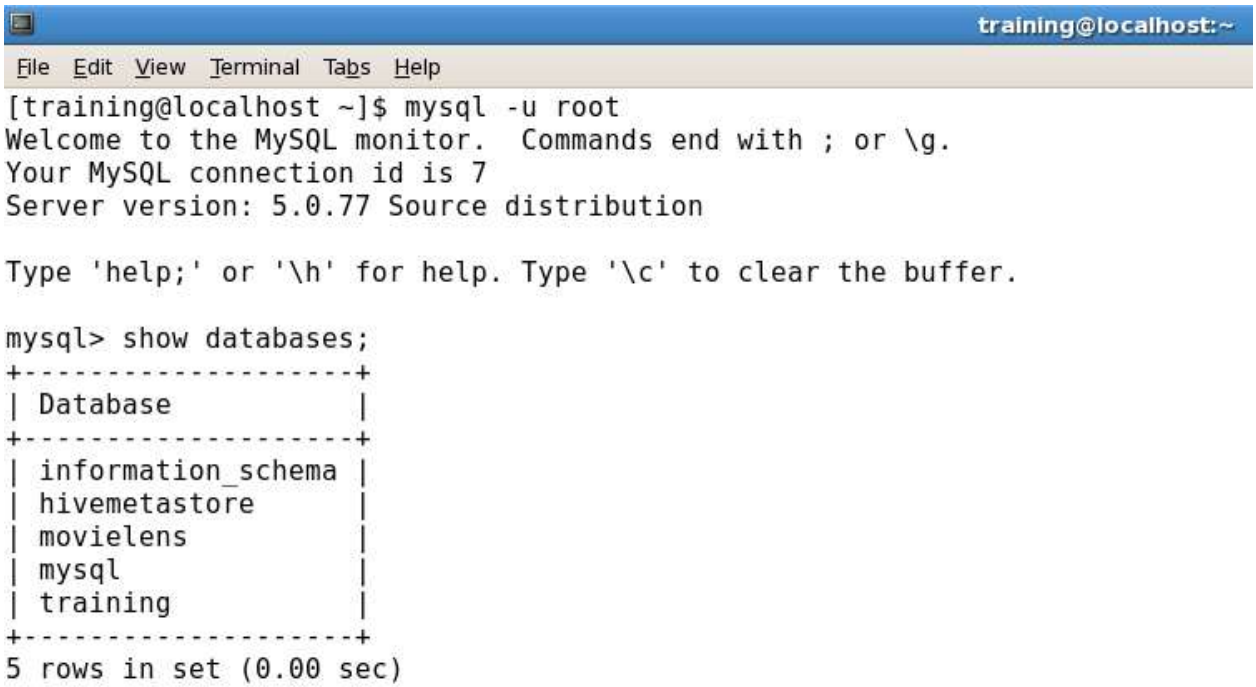
## MALE - FEMALE CHART GRAPH OF MURDER VICTIMS

## Query 5:- Year having maximum number of casualties (Policemen killed).

**CODING**

Step i. Open Mysql

```
[training@localhost ~]$ mysql -u root
mysql> show databases;
```




Step ii. Create a new database with name *policedb*

Step iii. Create an empty new table with name *policetb* by using previously created database i.e. policedb

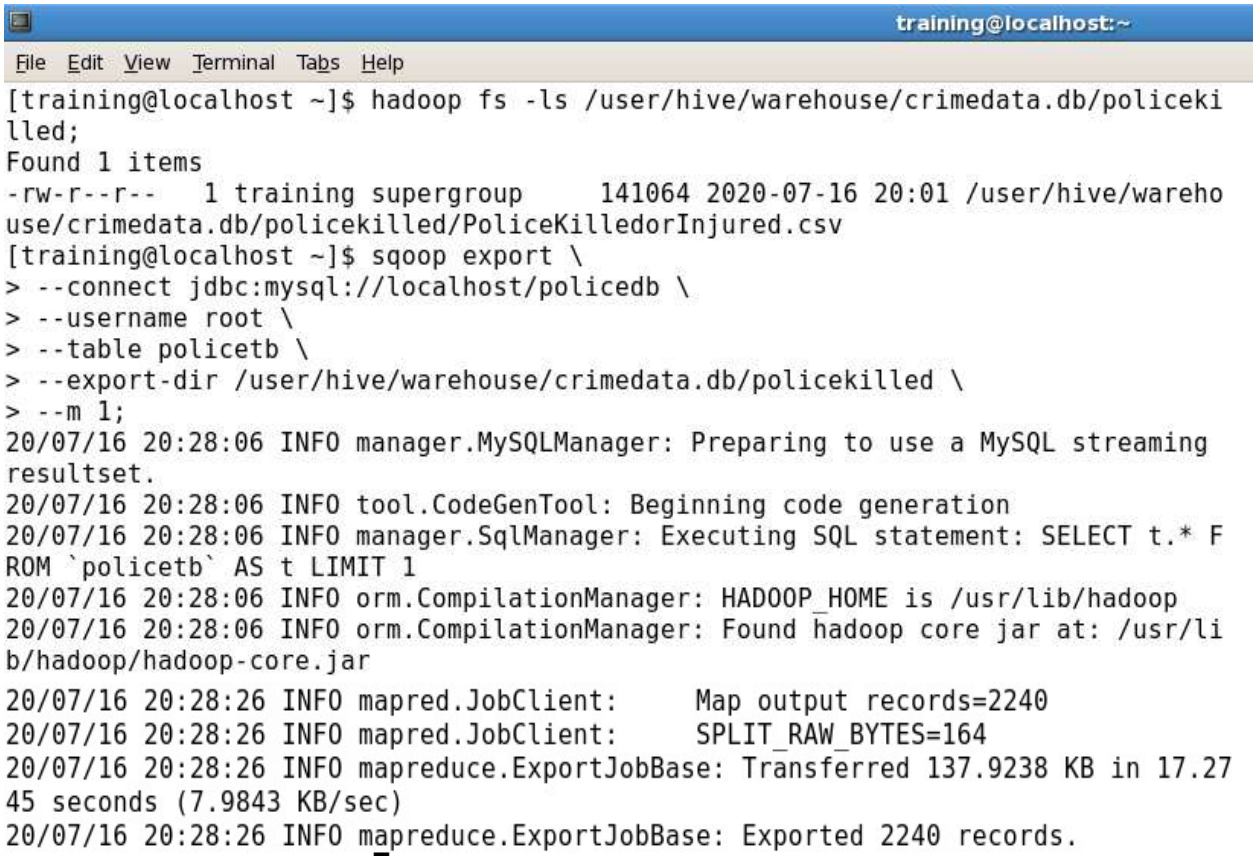
```
mysql> create database policedb;
mysql> use policedb;
mysql> create table policetb(AREA_NAME
varchar(30),YEAR int,CATEGORY
varchar(30),POLICE_INJURED_BY_CRIMINALS
```

int,POLICE_INJURED_BY_RIOTOUS_MOBS
int,POLICE_INJURED_IN_ACCIDENTS
int,POLICE_INJURED_IN_DACOITY_OPERATIONSOTHE
R_RAIDS
int,POLICE_INJURED_IN_TERRORISTSEXTREMISTS_OP
ERATIONS int,POLICE_INJURED_ON_BORDER_DUTIES
int,TOTAL_POLICEMEN_INJURED
int,POLICE_KILLED_BY_CRIMINALS
int,POLICE_KILLED_BY_RIOTOUS_MOBS
int,POLICE_KILLED_IN_ACCIDENTS
int,POLICE_KILLED_IN_DACOITY_OPERATIONSOTHER_
RAIDS
int,POLICE_KILLED_IN_TERRORISTSEXTREMISTS_OPE
RATIONS int,POLICE_KILLED_ON_BORDER_DUTIES
int,TOTAL_POLICEMEN_KILLED int);
mysql> quit

```
mysql> create database policedb;
Query OK, 1 row affected (0.00 sec)

mysql> use policedb;
Database changed
mysql> create table policetb(AREA_NAME varchar(30),YEAR int,CATEGORY varchar(30)
,POLICE_INJURED_BY_CRIMINALS int,POLICE_INJURED_BY_RIOTOUS_MOBS int,POLICE_INJUR
ED_IN_ACCIDENTS int,POLICE_INJURED_IN_DACOITY_OPERATIONSOTHER_RAIDS int,POLICE_I
NJURED_IN_TERRORISTSEXTREMISTS_OPERATIONS int,POLICE_INJURED_ON_BORDER_DUTIES in
t,TOTAL_POLICEMEN_INJURED int,POLICE_KILLED_BY_CRIMINALS int,POLICE_KILLED_BY_RI
OTOUS_MOBS int,POLICE_KILLED_IN_ACCIDENTS int,POLICE_KILLED_IN_DACOITY_OPERATION
SOTHER_RAIDS int,POLICE_KILLED_IN_TERRORISTSEXTREMISTS_OPERATIONS int,POLICE_KIL
LED_ON_BORDER_DUTIES int,TOTAL_POLICEMEN_KILLED int);
Query OK, 0 rows affected (0.00 sec)

mysql> quit
Bye
```

Step iv. Now exit mysql. Export the data present on hadoop(dataset) using SQOOP into MYSQL in table *policetb*

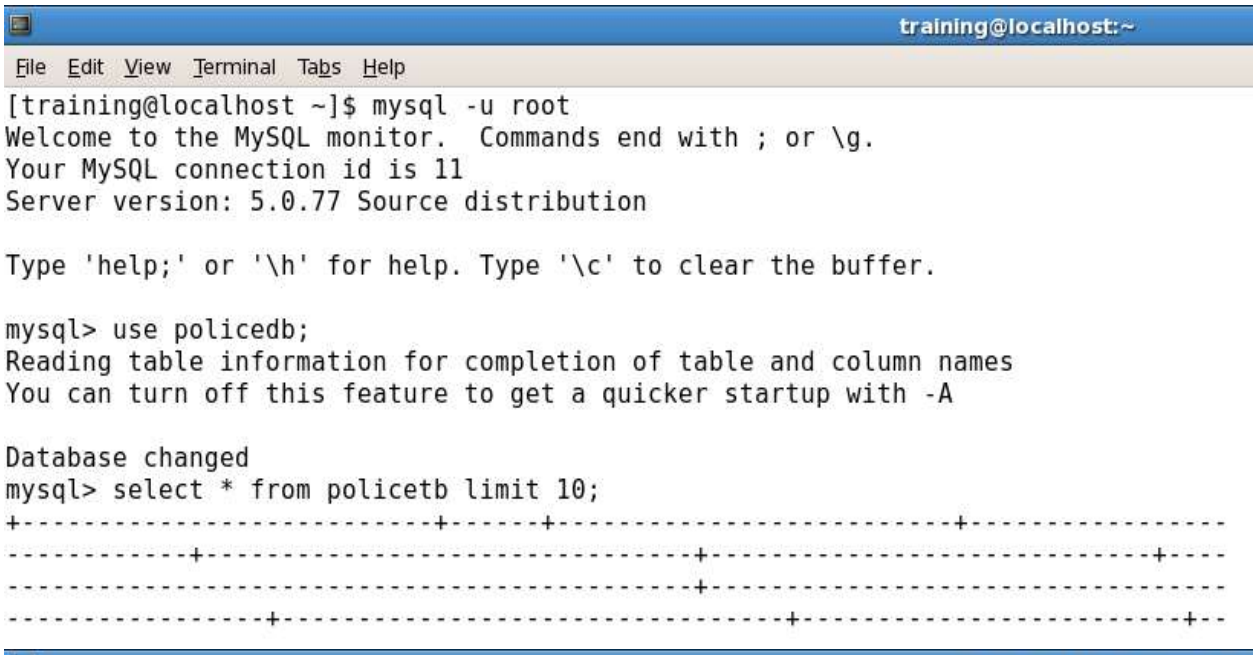
```
[training@localhost ~]$ hadoop fs -ls
/user/hive/warehouse/crimedata.db/policekilled;
[training@localhost ~]$ sqoop export \
> --connect jdbc:mysql://localhost/policedb \
> --username root \
> --table policetb \
> --export-dir /user/hive/warehouse/crimedata.db/policekilled
\
> --m 1;
```



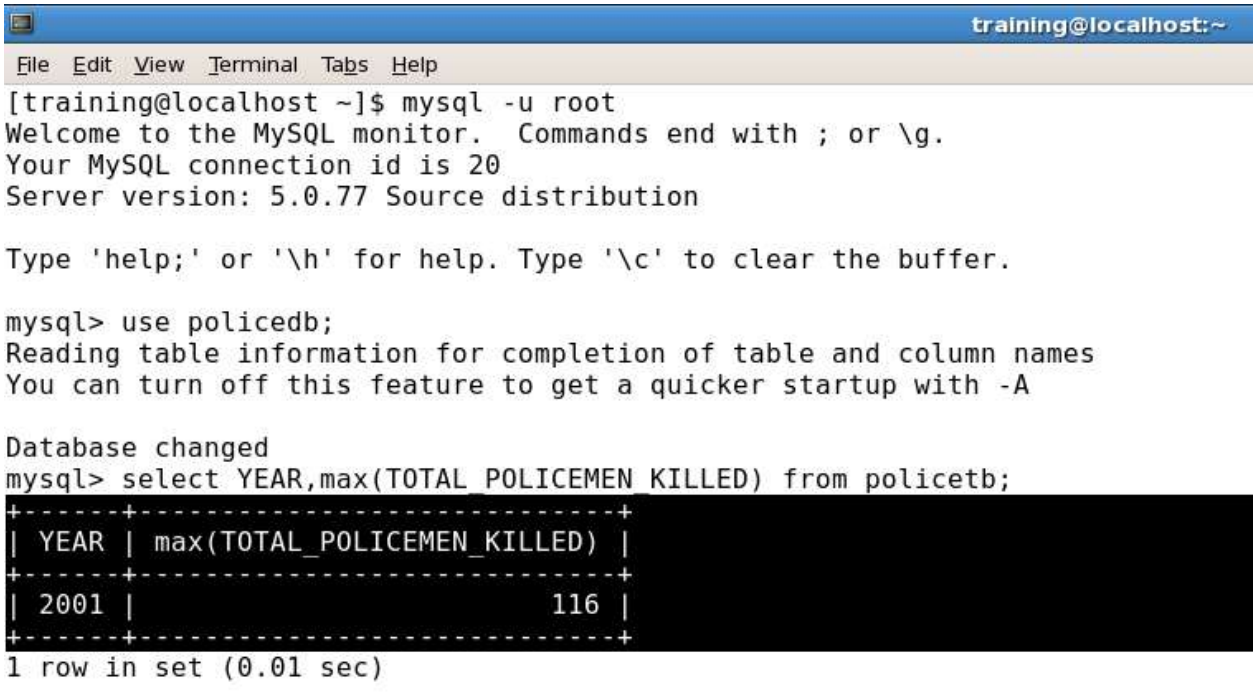

Step v. Open mysql and use same database i.e. *policedb*

Step vi. Perform MAX function over the data present in *policetb*

```
[training@localhost ~]$ mysql -u root
mysql> use policedb;
mysql> select * from policetb limit 10;
mysql> select YEAR,MAX(TOTAL_POLICEMEN_KILLED)
from policetb;
```



```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
[training@localhost ~]$ mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 5.0.77 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use policedb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from policetb limit 10;
+---------------------------+------+--------------------------+-----------------
------------+--------------------------------+----------------------------+----
-------------------------------------------+----------------------------------
-----------------+--------------------------------+--------------------------+--
```



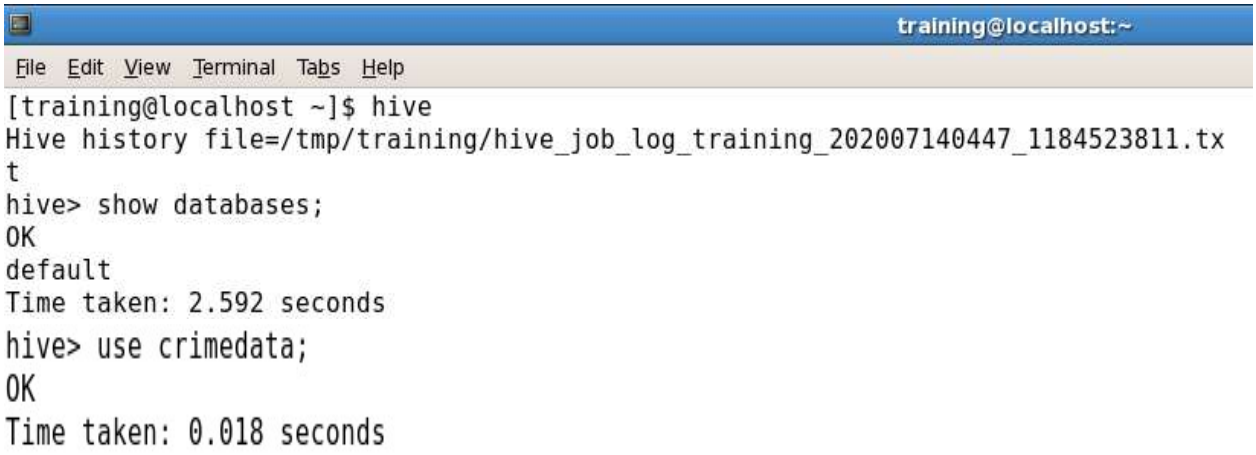
```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
[training@localhost ~]$ mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 20
Server version: 5.0.77 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use policedb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select YEAR,max(TOTAL_POLICEMEN_KILLED) from policetb;
+------+------------------------------+
| YEAR | max(TOTAL_POLICEMEN_KILLED) |
+------+------------------------------+
| 2001 |                          116 |
+------+------------------------------+
1 row in set (0.01 sec)
```

## Query 6:- Total number of Dacoity happened in each year.

**CODING**

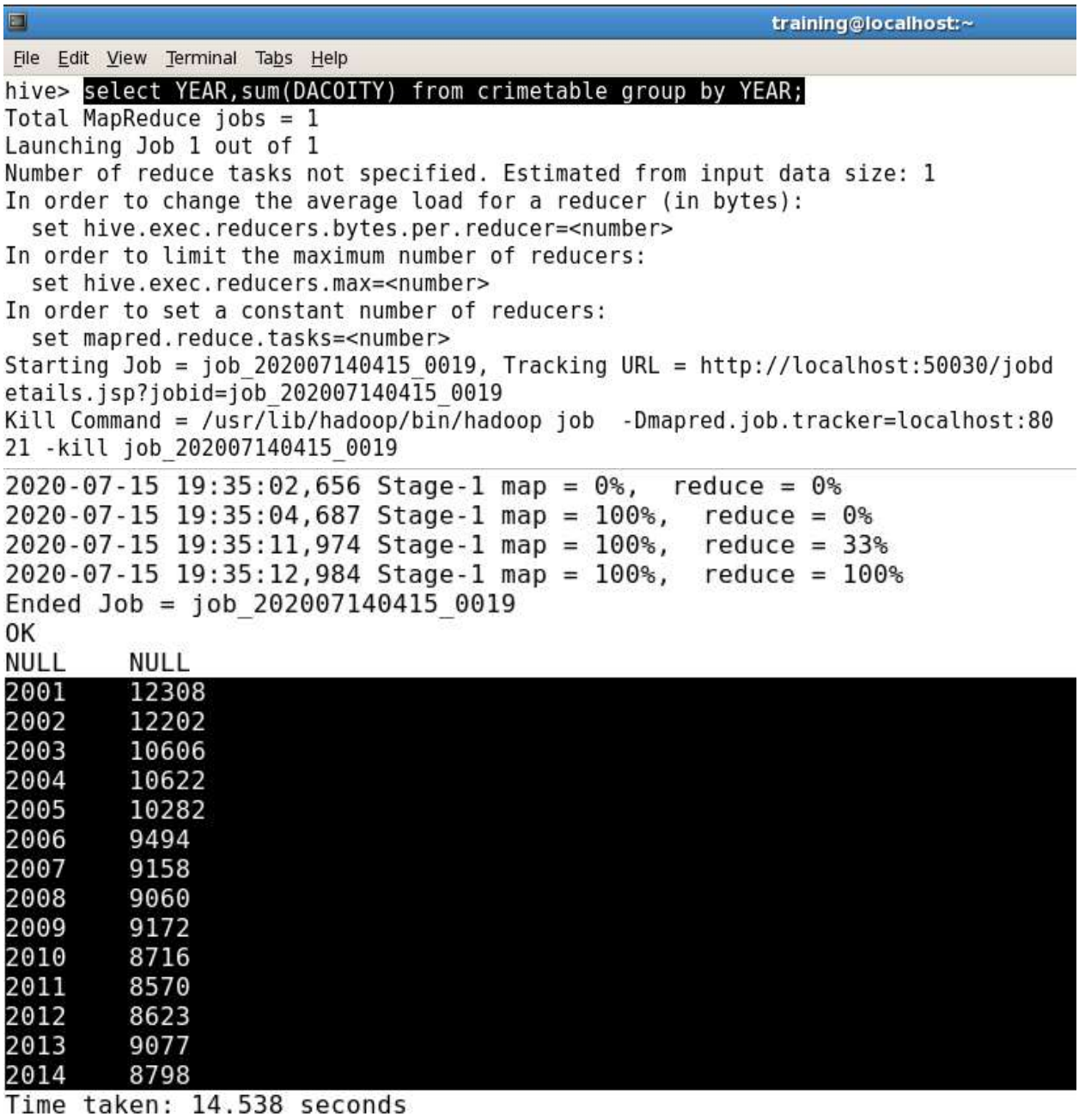
Step i. Open hive and use previously created database i.e. crimedata

```
[training@localhost ~]$ hive
hive> show databases;
hive> use crimedata;
```




Step ii. Perform SUM function over the column DACOITY by grouping YEAR

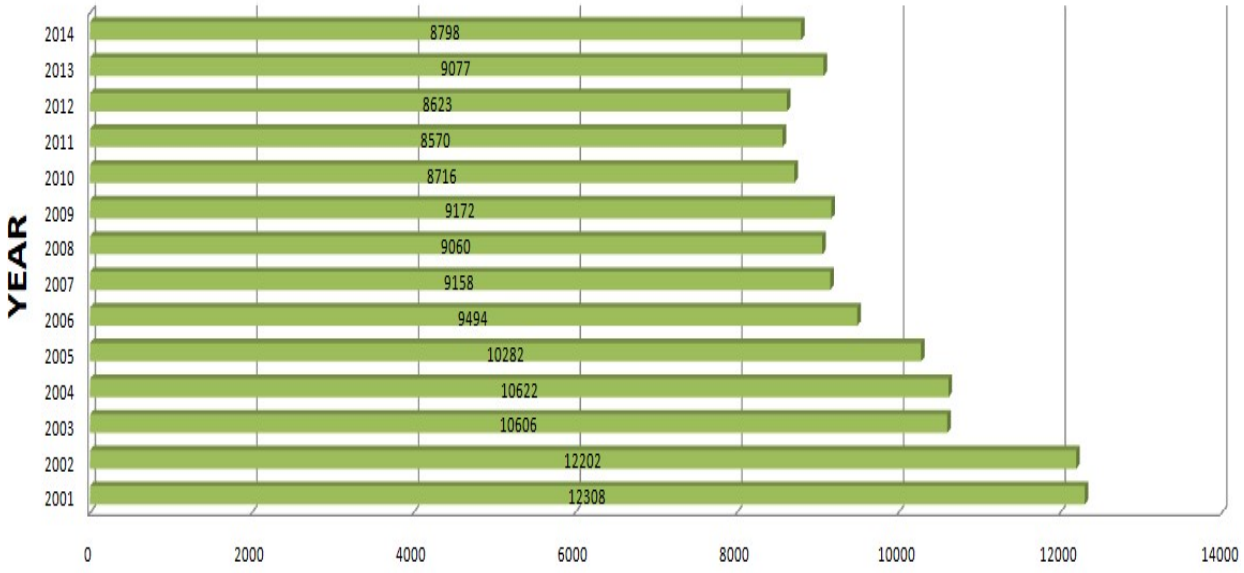
```
hive> select YEAR,sum(DACOITY) from crimetable group by
YEAR;
```




## VISUALIZATION

This bar graph shows the graph plotted between DACOITY happened in each YEAR.

## DACOITY CHART GRAPH



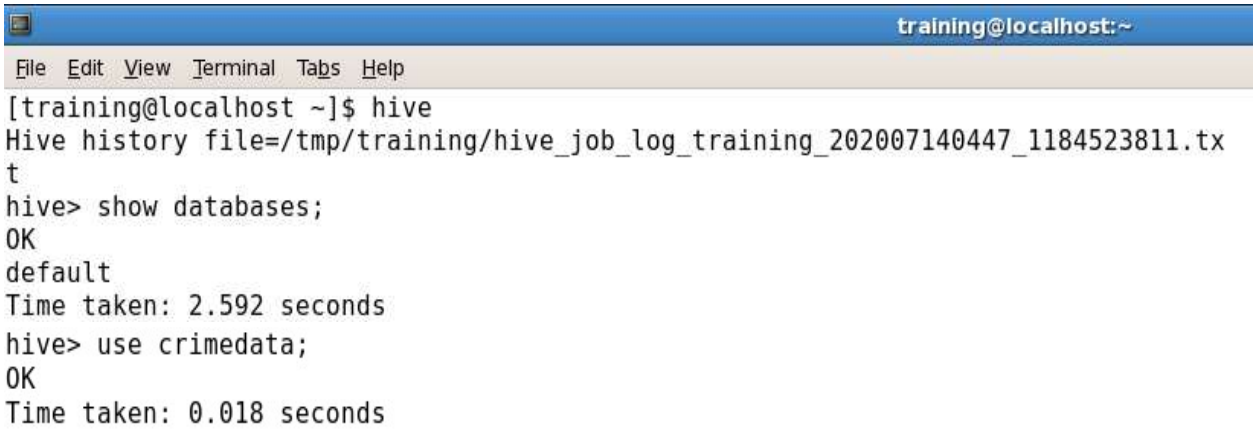

| | 2001 | 2002 | 2003 | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | 2014 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Total | 12308 | 12202 | 10606 | 10622 | 10282 | 9494 | 9158 | 9060 | 9172 | 8716 | 8570 | 8623 | 9077 | 8798 |

## Query 7:- Top 5 states of Dowry Deaths during the year 2014.

**CODING**

Step i. Open hive and use previously created database i.e. crimedata

```
[training@localhost ~]$ hive
hive> show databases;
hive> use crimedata;
```




Step ii. Create a new table with name *dowrydeaths*

Step iii. Load the data from previously created table i.e. *crimetable*

```
hive> create table dowrydeaths(STATE_UT
string,DOWRY_DEATHS int) row format delimited fields
terminated by ',' lines terminated by '\n' stored as textfile;
hive> insert overwrite table dowrydeaths select
STATE_UT,sum(DOWRY_DEATHS) from crimetable where
YEAR='2014' group by STATE_UT;
```

```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
hive> create table dowrydeaths(STATE_UT string,DOWRY_DEATHS int) row format deli
mited fields terminated by ',' lines terminated by '\n' stored as textfile;
OK
Time taken: 0.258 seconds
```

```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
hive> insert overwrite table dowrydeaths select STATE_UT,sum(DOWRY_DEATHS) from
crimetable where YEAR='2014' group by STATE_UT;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202007140415_0014, Tracking URL = http://localhost:50030/jobd
etails.jsp?jobid=job_202007140415_0014
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:80
21 -kill job_202007140415_0014
2020-07-15 04:44:10,533 Stage-1 map = 0%,  reduce = 0%
2020-07-15 04:44:13,570 Stage-1 map = 100%,  reduce = 0%
2020-07-15 04:44:20,634 Stage-1 map = 100%,  reduce = 33%
2020-07-15 04:44:21,646 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_202007140415_0014
Loading data to table crimedata.dowrydeaths
Deleted hdfs://localhost/user/hive/warehouse/crimedata.db/dowrydeaths
Table crimedata.dowrydeaths stats: [num_partitions: 0, num_files: 1, num_rows: 0
, total_size: 491]
36 Rows loaded to dowrydeaths
OK
Time taken: 20.656 seconds
```

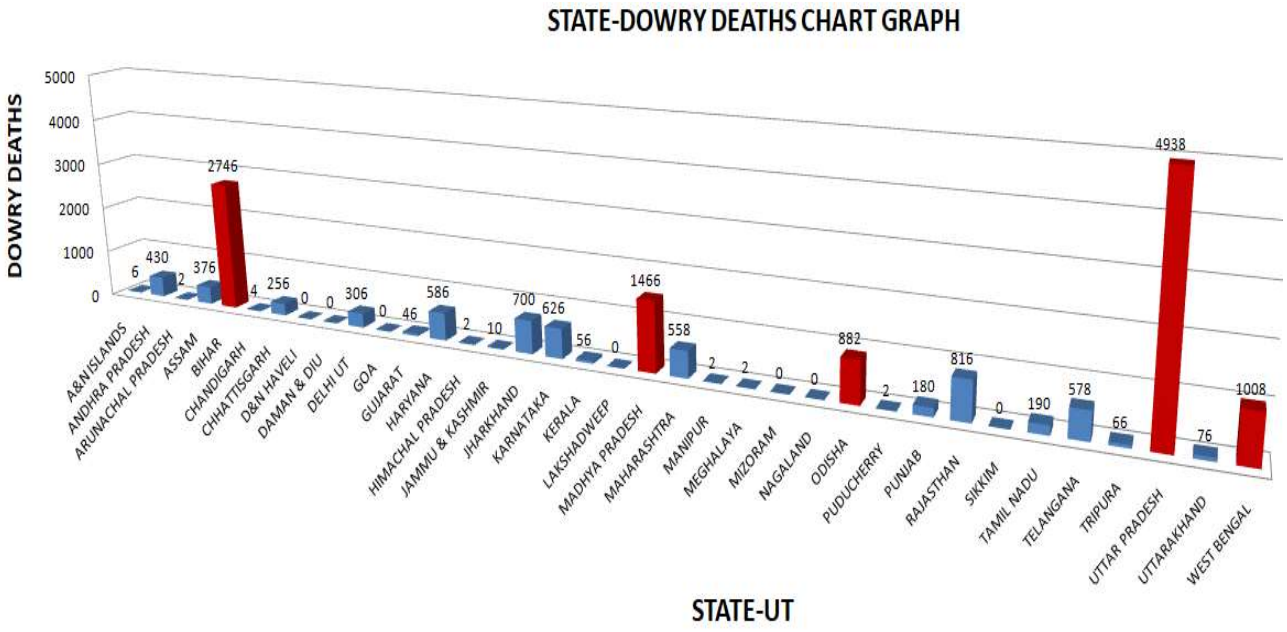
Step iv.Perform ***desc*** function with ***limit 5*** over the data present in dowrydeaths

```
hive> select * from dowrydeaths sort by DOWRY_DEATHS desc limit 5;
```

```
hive> select * from dowrydeaths sort by DOWRY_DEATHS desc limit 5;
Total MapReduce jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202007140415_0016, Tracking URL = http://localhost:50030/jobd
etails.jsp?jobid=job_202007140415_0016
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:80
21 -kill job_202007140415_0016
2020-07-15 04:46:14,780 Stage-1 map = 0%,  reduce = 0%
2020-07-15 04:46:16,798 Stage-1 map = 100%,  reduce = 0%
2020-07-15 04:46:23,895 Stage-1 map = 100%,  reduce = 33%
2020-07-15 04:46:24,905 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_202007140415_0016
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202007140415_0017, Tracking URL = http://localhost:50030/jobd
etails.jsp?jobid=job_202007140415_0017
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:80
21 -kill job_202007140415_0017
2020-07-15 04:46:28,168 Stage-2 map = 0%,  reduce = 0%
2020-07-15 04:46:30,221 Stage-2 map = 100%,  reduce = 0%
2020-07-15 04:46:37,283 Stage-2 map = 100%,  reduce = 33%
2020-07-15 04:46:38,293 Stage-2 map = 100%,  reduce = 100%
Ended Job = job_202007140415_0017
OK
UTTAR PRADESH   4938
BIHAR   2746
MADHYA PRADESH  1466
WEST BENGAL     1008
ODISHA  882
Time taken: 26.895 seconds
```

## VISUALIZATION

This column graph shows the graph plotted between STATE/UT and DOWRY DEATHS. The ***FIVE BAR*** with ***RED COLOR*** shows the top 5 category in the graph.


STATE-DOWRY DEATHS CHART GRAPH
DOWRY DEATHS
5000
4000
3000
2000
1000
0
6
430
2
376
2746
4
256
0
0
306
0
46
586
2
10
700
626
56
0
1466
558
2
2
0
0
882
2
180
816
0
190
578
66
4938
76
1008
A&N ISLANDS
ANDHRA PRADESH
ARUNACHAL PRADESH
ASSAM
BIHAR
CHANDIGARH
CHHATTISGARH
D&N HAVELI
DAMAN & DIU
DELHI UT
GOA
GUJARAT
HARYANA
HIMACHAL PRADESH
JAMMU & KASHMIR
JHARKHAND
KARNATAKA
KERALA
LAKSHADWEEP
MADHYA PRADESH
MAHARASHTRA
MANIPUR
MEGHALAYA
MIZORAM
NAGALAND
ODISHA
PUDUCHERRY
PUNJAB
RAJASTHAN
SIKKIM
TAMIL NADU
TELANGANA
TRIPURA
UTTAR PRADESH
UTTARAKHAND
WEST BENGAL
STATE-UT

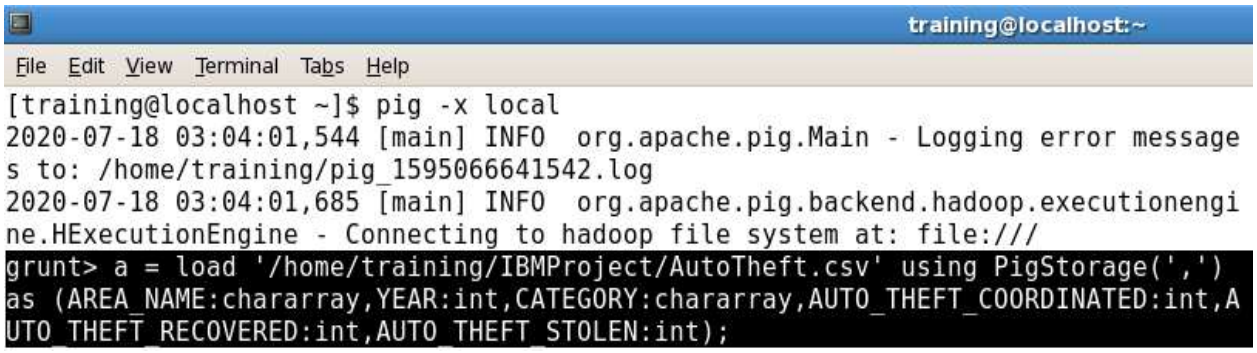
## Query 8:- Top 3 category of Auto Theft during year 2011.

**CODING**

Step i. Open Pig in local mode

Step ii. Create a bag with name ***'a'*** and load the data present on Hadoop (dataset) into bag ***'a'***

```
[training@localhost ~]$ pig -x local
grunt> a = load
'/home/training/IBMProject/AutoTheft.csv' using
PigStorage(',') as
(AREA_NAME:chararray,YEAR:int,CATEGORY:chararr
ay,AUTO_THEFT_COORDINATED:int,AUTO_THEFT_
RECOVERED:int,AUTO_THEFT_STOLEN:int);
```




Step iii. Now perform GROUP operation by referencing with a column CATEGORY in a bag ***'b'***

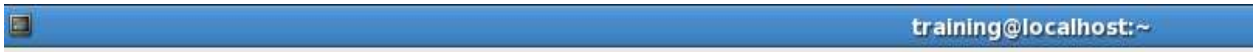
Step iv. Perform COUNT operation on ***'a'*** in bag ***'c'***

```
grunt> b = group a by CATEGORY;
grunt> c = foreach b generate group,COUNT(a) as c;
grunt> dump c;
```

```
grunt> b = group a by CATEGORY;
grunt> c = foreach b generate group,COUNT(a) as c;
grunt> dump c;
2020-07-18 03:15:30,940 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: GROUP_BY
2020-07-18 03:15:30,941 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - pig.usenewlogicalplan is set to true. New logical plan wil
l be used.
2020-07-18 03:15:31,166 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - (Name: c: Store(file:/tmp/temp216009587/tmp1421318702:org.
apache.pig.impl.io.InterStorage) - scope-30 Operator Key: scope-30)
nputFormat - Total input paths to process : 1
2020-07-18 03:15:40,781 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(BUSES,103)
( BUSES,254)
(TRUCKS/TEMPO ,124)
( CAR/TAXI/JEEP,465)
(OTHER  VEHICLES,122)
( CYCLES/ SCOOTERS,480)
( OTHER MOTOR VEHICLES,303)
( GOODS CARRYING VEHICLES (TRUCKS/TEMPO ETC),292)
```

Step v. Order bag ***'c'*** as Descending Order

```
grunt> d = order c by c desc;
grunt> dump d;
```




```
grunt> d = order c by c desc;
grunt> dump d;
2020-07-18 03:23:42,076 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: GROUP_BY,ORDER_BY
2020-07-18 03:23:42,076 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - pig.usenewlogicalplan is set to true. New logical plan wil
l be used.
2020-07-18 03:23:42,124 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - (Name: d: Store(file:/tmp/temp216009587/tmp1195855272:org.
```

```
nputFormat - Total input paths to process : 1
2020-07-18 03:24:29,553 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
( CYCLES/ SCOOTERS,480)
( CAR/TAXI/JEEP,465)
( OTHER MOTOR VEHICLES,303)
( GOODS CARRYING VEHICLES (TRUCKS/TEMPO ETC),292)
( BUSES,254)
(TRUCKS/TEMPO ,124)
(OTHER  VEHICLES,122)
(BUSES,103)
```

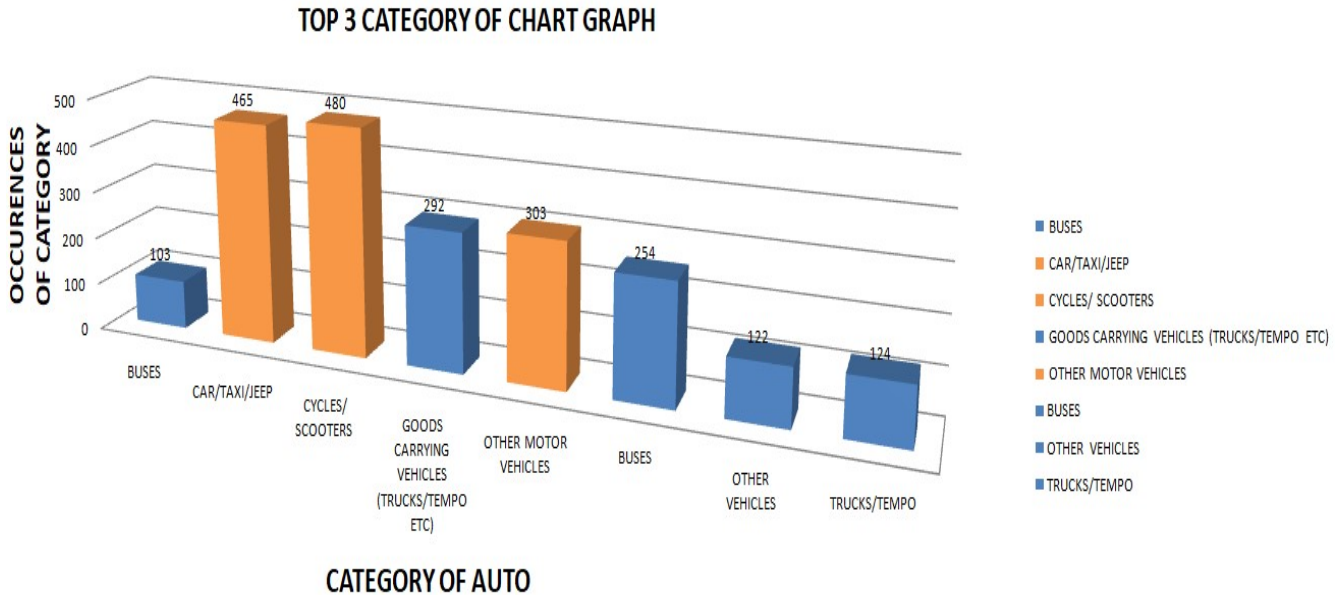
Step vi. Apply LIMIT operation as limit 3.

```
grunt> e = limit d 3;
grunt> dump e;
```

```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
grunt> e = limit d 3;
grunt> dump e;
2020-07-18 03:28:55,273 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: GROUP_BY,ORDER_BY,LIMIT
2020-07-18 03:28:55,273 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - pig.usenewlogicalplan is set to true. New logical plan wil
l be used.
2020-07-18 03:29:35,258 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Success!
2020-07-18 03:29:35,260 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2020-07-18 03:29:35,260 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
( CYCLES/ SCOOTERS,480)
( CAR/TAXI/JEEP,465)
( OTHER MOTOR VEHICLES,303)
```

## VISUALIZATION

This column graph shows the graph plotted between CATEGORY OF AUTO and their OCCURENCES. The ***THREE BAR*** with ***ORANGE COLOR*** shows the top 3 category in the graph.


TOP 3 CATEGORY OF CHART GRAPH
OCCURENCES OF CATEGORY
500
400
300
200
100
0
103
465
480
292
303
254
122
124
BUSES
CAR/TAXI/JEEP
CYCLES/ SCOOTERS
GOODS CARRYING VEHICLES (TRUCKS/TEMPO ETC)
OTHER MOTOR VEHICLES
BUSES
OTHER VEHICLES
TRUCKS/TEMPO
CATEGORY OF AUTO
BUSES
CAR/TAXI/JEEP
CYCLES/ SCOOTERS
GOODS CARRYING VEHICLES (TRUCKS/TEMPO ETC)
OTHER MOTOR VEHICLES
BUSES
OTHER VEHICLES
TRUCKS/TEMPO

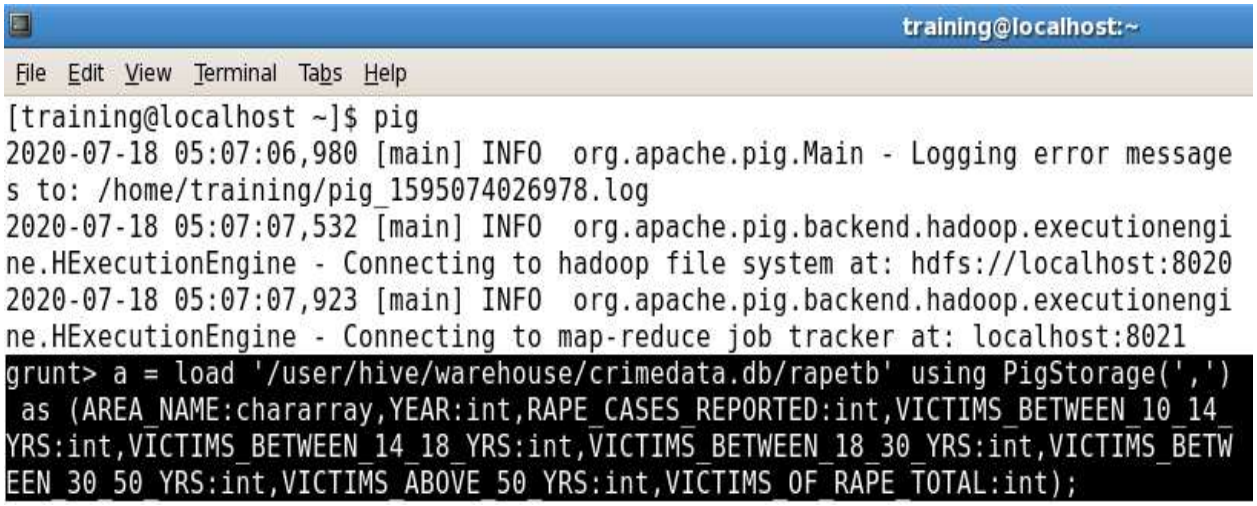
## Query 9:- Year wise Total Rape case reported v/s Total rape victims.

**CODING**

Step i. Open Pig in MapReduce mode

Step ii. Create a bag with name ***'a'*** and load the data present on Hadoop (dataset) into bag ***'a'***

```
[training@localhost ~]$ pig
grunt> a = load
'/user/hive/warehouse/crimedata.db/rapetb' using
PigStorage(',') as
(AREA_NAME:chararray,YEAR:int,RAPE_CASES_RE
PORTED:int,VICTIMS_BETWEEN_10_14_YRS:int,VIC
TIMS_BETWEEN_14_18_YRS:int,VICTIMS_BETWEE
N_18_30_YRS:int,VICTIMS_BETWEEN_30_50_YRS:i
nt,VICTIMS_ABOVE_50_YRS:int,VICTIMS_OF_RAPE
_TOTAL:int);
```




Step iii. Now perform GROUP operation by referencing with a column YEAR in a bag ***'b'***

```
grunt> b = group a by YEAR;
grunt> dump b;
```

```
grunt> b = group a by YEAR;
grunt> dump b;
,(TAMIL NADU,2014,686,50,139,411,66,0,666),(TAMIL NADU,2014,0,0,0,0,0,0,0),(TAMI
L NADU,2014,686,50,139,411,66,0,666),(TRIPURA,2014,238,27,61,93,38,0,219),(TRIPU
RA,2014,0,0,0,0,0,0,0),(TRIPURA,2014,238,27,61,93,38,0,219),(UTTAR PRADESH,2014,
1563,162,246,862,250,0,1520),(UTTAR PRADESH,2014,4,1,2,1,0,0,4),(UTTAR PRADESH,2
014,1559,161,244,861,250,0,1516),(UTTARAKHAND,2014,121,3,5,92,19,0,119),(UTTARAK
HAND,2014,11,2,2,5,1,0,10),(UTTARAKHAND,2014,110,1,3,87,18,0,109),(WEST BENGAL,2
014,2311,22,51,1779,459,0,2311),(WEST BENGAL,2014,4,1,0,2,1,0,4),(WEST BENGAL,20
14,2307,21,51,1777,458,0,2307)})
```

Step iv.Perform SUM operation on ***'a'*** in bag ***'c'***

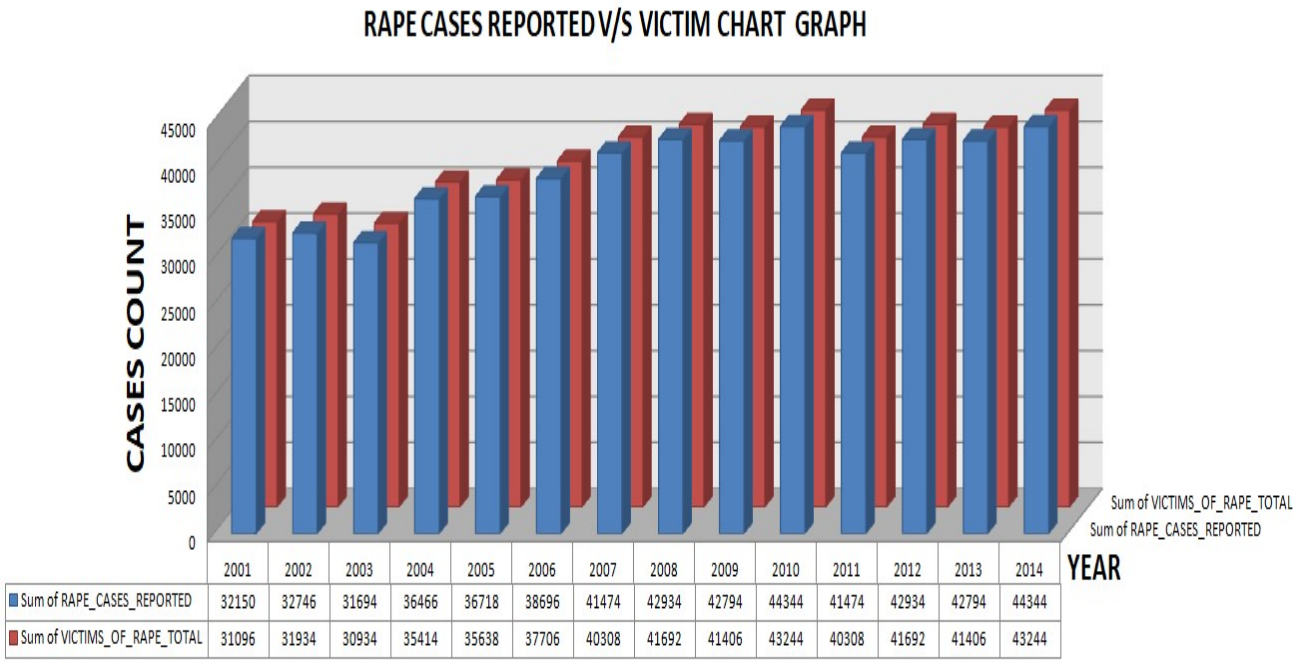
```
grunt> c = foreach b generate
group,SUM(a.RAPE_CASES_REPORTED),SUM(a.VIC
TIMS_OF_RAPE_TOTAL);
grunt> dump c;
```

```
training@localhost:~
File  Edit  View  Terminal  Tabs  Help
grunt> c = foreach b generate group,SUM(a.RAPE_CASES_REPORTED),SUM(a.VICTIMS_OF_
RAPE_TOTAL);
grunt> dump c;
2020-07-18 05:22:27,441 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: GROUP_BY
2020-07-18 05:22:27,441 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - pig.usenewlogicalplan is set to true. New logical plan wil
l be used.
2020-07-18 05:22:27,468 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - (Name: c: Store(hdfs://localhost/tmp/temp-1173743573/tmp-2
03161532:org.apache.pig.impl.io.InterStorage) - scope-114 Operator Key: scope-11
```

```
nputFormat - Total input paths to process : 1
2020-07-18 05:22:50,740 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(2001,32150,31096)
(2002,32746,31934)
(2003,31694,30934)
(2004,36466,35414)
(2005,36718,35638)
(2006,38696,37706)
(2007,41474,40308)
(2008,42934,41692)
(2009,42794,41406)
(2010,44344,43244)
(2011,41474,40308)
(2012,42934,41692)
(2013,42794,41406)
(2014,44344,43244)
```

## VISUALIZATION

This column graph shows the graphical comparison between RAPE CASES REPORTED and VICTIMS OF RAPE CASES.




| | 2001 | 2002 | 2003 | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | 2014 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Sum of RAPE_CASES_REPORTED | 32150 | 32746 | 31694 | 36466 | 36718 | 38696 | 41474 | 42934 | 42794 | 44344 | 41474 | 42934 | 42794 | 44344 |
| Sum of VICTIMS_OF_RAPE_TOTAL | 31096 | 31934 | 30934 | 35414 | 35638 | 37706 | 40308 | 41692 | 41406 | 43244 | 40308 | 41692 | 41406 | 43244 |

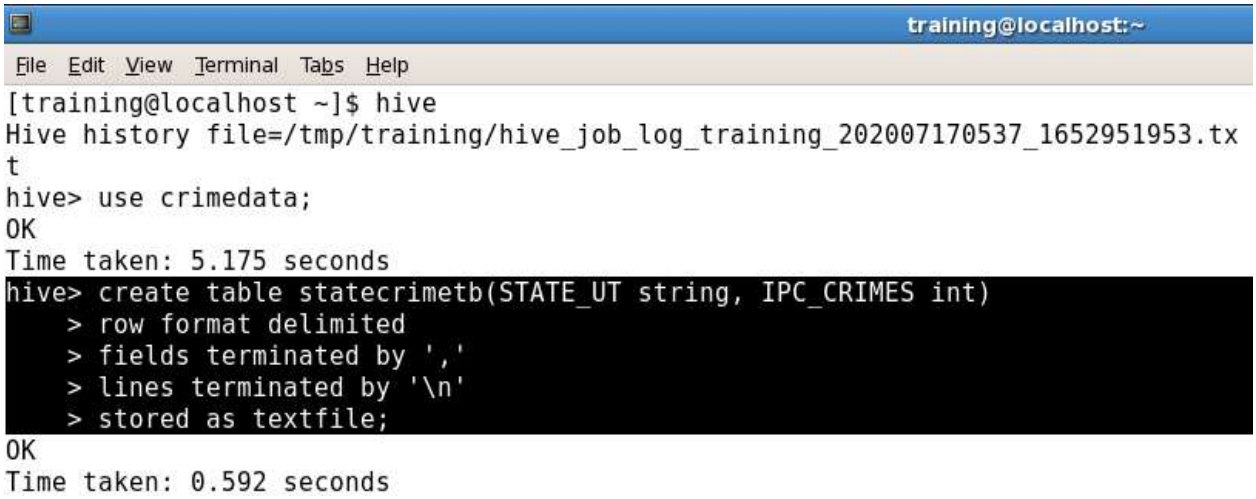
## Query 10:- State wise percentage of IPC crimes recorded.

**CODING**

Step i. Open Hive

Step ii. Create a new table with name statecrimetb into previously created database i.e. crimedata

```
[training@localhost ~]$ hive
hive> use crimedata;
hive> create table statecrimetb(STATE_UT string,
IPC_CRIMES int)
    > row format delimited
    > fields terminated by ','
    > lines terminated by '\n'
    > stored as textfile;
```




Step iii. Insert data into statecrimetb from previously created crimetable

hive> insert overwrite table statecrimetb select STATE_UT,sum(TOTAL_IPC_CRIMES) from crimetable group by STATE_UT;
hive> select * from statecrimetb;

```
hive> insert overwrite table statecrimetb select STATE_UT,sum(TOTAL_IPC_CRIMES)
from crimetable group by STATE_UT;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202007161753_0012, Tracking URL = http://localhost:50030/jobd
etails.jsp?jobid=job_202007161753_0012
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:80
21 -kill job_202007161753_0012
2020-07-17 05:40:05,807 Stage-1 map = 0%,  reduce = 0%
2020-07-17 05:40:07,830 Stage-1 map = 100%,  reduce = 0%
2020-07-17 05:40:18,978 Stage-1 map = 100%,  reduce = 33%
hive> select * from statecrimetb;
OK
A & N ISLANDS   18230
A&N ISLANDS     2288
ANDHRA PRADESH  4614898
ARUNACHAL PRADESH       67222
ASSAM   1606674
BIHAR   3413366
CHANDIGARH      96520
CHHATTISGARH    1340494
D & N HAVELI    9476
D&N HAVELI      1106
DAMAN & DIU     6808
DELHI UT        1742776
GOA     80142
GUJARAT 3316696
HARYANA 1472820
HIMACHAL PRADESH        363354
JAMMU & KASHMIR 616670
JHARKHAND       1033274
```

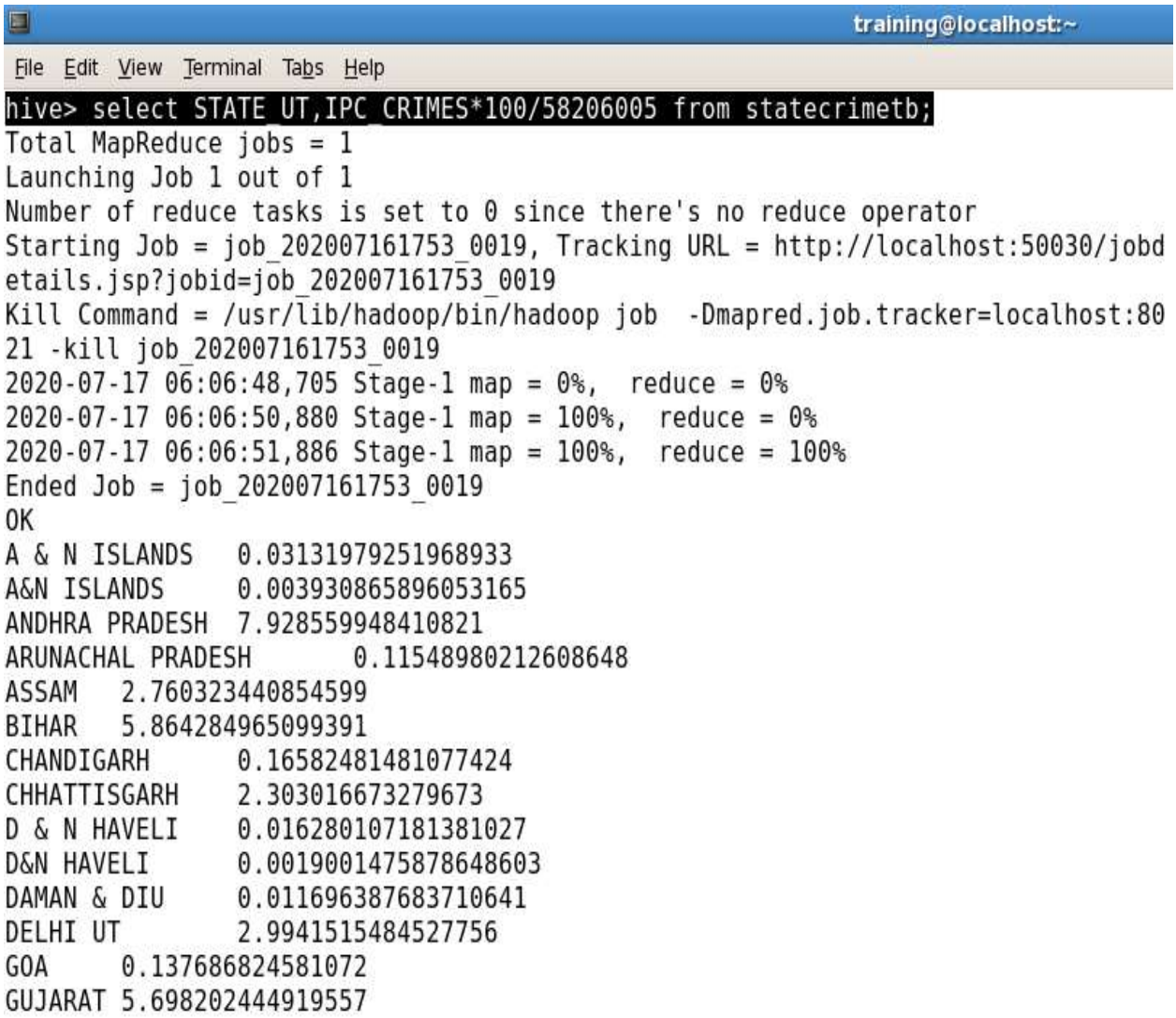
Step iv. Perform SUM operation over the column IPC_CRIMES of table statecrimetb

```
hive> select sum(IPC_CRIMES) from statecrimetb;
```

```
hive> select sum(IPC_CRIMES) from statecrimetb;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202007161753_0020, Tracking URL = http://localhost:50030/jobd
etails.jsp?jobid=job_202007161753_0020
Kill Command = /usr/lib/hadoop/bin/hadoop job  -Dmapred.job.tracker=localhost:80
21 -kill job_202007161753_0020
2020-07-17 06:07:27,350 Stage-1 map = 0%,  reduce = 0%
2020-07-17 06:07:29,614 Stage-1 map = 100%,  reduce = 0%
2020-07-17 06:07:39,276 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_202007161753_0020
OK
58206005
Time taken: 17.112 seconds
```

Step v. Apply PERCENTAGE over the column IPC_CRIMES of statecrimetb

```
hive> select STATE_UT,IPC_CRIMES*100/58206005 from statecrimetb;
```

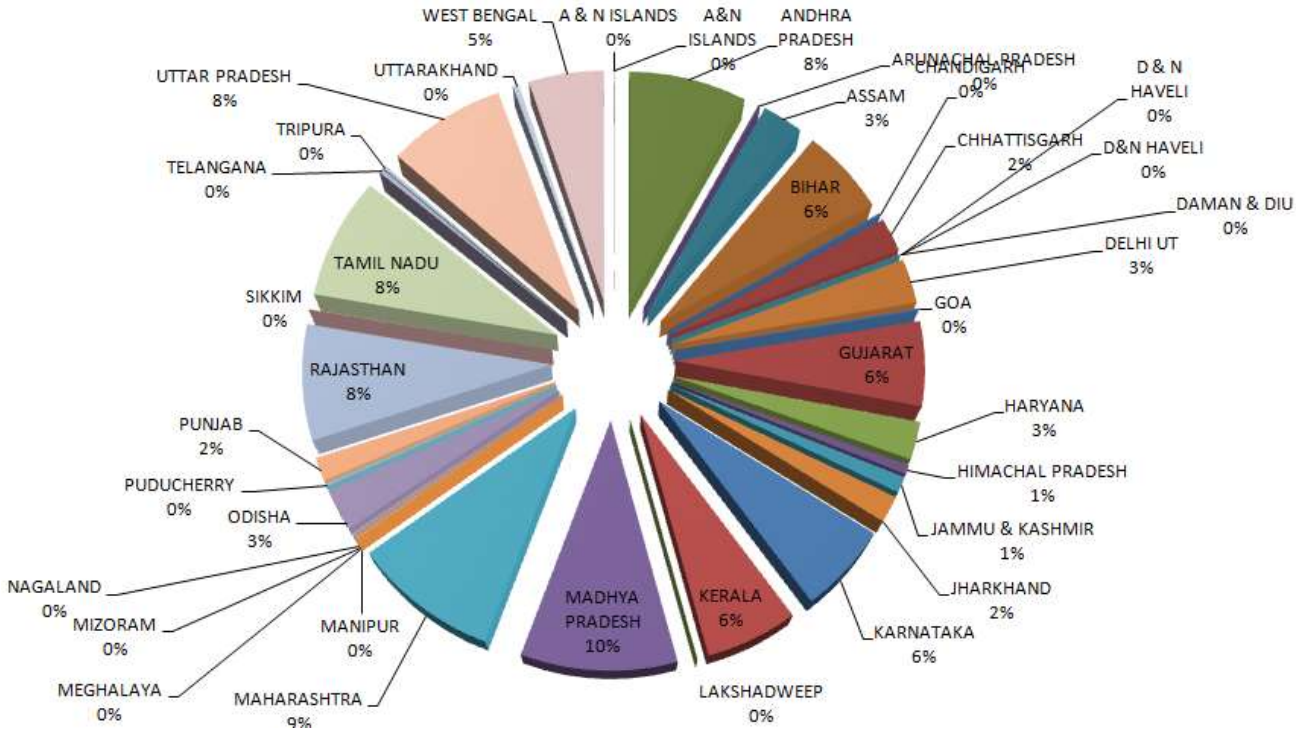

training@localhost:~
File Edit View Terminal Tabs Help
hive> select STATE_UT,IPC_CRIMES*100/58206005 from statecrimetb;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_202007161753_0019, Tracking URL = http://localhost:50030/jobd
etails.jsp?jobid=job_202007161753_0019
Kill Command = /usr/lib/hadoop/bin/hadoop job -Dmapred.job.tracker=localhost:80
21 -kill job_202007161753_0019
2020-07-17 06:06:48,705 Stage-1 map = 0%, reduce = 0%
2020-07-17 06:06:50,880 Stage-1 map = 100%, reduce = 0%
2020-07-17 06:06:51,886 Stage-1 map = 100%, reduce = 100%
Ended Job = job_202007161753_0019
OK
A & N ISLANDS 0.03131979251968933
A&N ISLANDS 0.003930865896053165
ANDHRA PRADESH 7.928559948410821
ARUNACHAL PRADESH 0.11548980212608648
ASSAM 2.760323440854599
BIHAR 5.864284965099391
CHANDIGARH 0.16582481481077424
CHHATTISGARH 2.303016673279673
D & N HAVELI 0.016280107181381027
D&N HAVELI 0.0019001475878648603
DAMAN & DIU 0.011696387683710641
DELHI UT 2.9941515484527756
GOA 0.137686824581072
GUJARAT 5.698202444919557

```
HARYANA 2.5303574777207265
HIMACHAL PRADESH        0.6242551777947997
JAMMU & KASHMIR 1.0594611329191894
JHARKHAND       1.7752017167300864
KARNATAKA       5.920468171625934
KERALA  5.852282767044397
LAKSHADWEEP     0.0029481494220398737
MADHYA PRADESH  9.874149583019827
MAHARASHTRA     9.345070152126057
MANIPUR 0.14130157189107895
MEGHALAYA       0.1098477725794787
MIZORAM 0.10253237616977835
NAGALAND        0.05254097064383649
ODISHA  2.733810712485765
PUDUCHERRY      0.19217432977920404
PUNJAB  1.5766105232613026
RAJASTHAN       7.789436845906192
SIKKIM  0.030069062461854236
TAMIL NADU      8.268875350575941
TELANGANA       0.31029444470549045
TRIPURA 0.2226505667241722
UTTAR PRADESH   7.983990655259711
UTTARAKHAND     0.4188021493658601
WEST BENGAL     5.050200576383828
Time taken: 9.616 seconds
```

## VISUALIZATION

This Pie-chart graph shows percentage wise representation of IPC CRIMES in STATE/UT.

# STATE/UT - CRIMES PERCENTAGE CHART GRAPGH




A & N ISLANDS | A&N ISLANDS | ANDHRA PRADESH | ARUNACHAL PRADESH | ASSAM | BIHAR | CHANDIGARH

CHHATTISGARH | D & N HAVELI | D&N HAVELI | DAMAN & DIU | DELHI UT | GOA | GUJARAT

HARYANA | HIMACHAL PRADESH | JAMMU & KASHMIR | JHARKHAND | KARNATAKA | KERALA | LAKSHADWEEP

MADHYA PRADESH | MAHARASHTRA | MANIPUR | MEGHALAYA | MIZORAM | NAGALAND | ODISHA

PUDUCHERRY | PUNJAB | RAJASTHAN | SIKKIM | TAMIL NADU | TELANGANA | TRIPURA

UTTAR PRADESH | UTTARAKHAND | WEST BENGAL