NANDHA ENGINEERING COLLEGE

(AUTONOMOUS)
Erode

RECORD NOTE BOOK

17AIP04 BIG DATA ANALYTICS LABORATORY
VI - Semester

B.TECH (ARTIFICIAL INTELLIGENCE AND DATA SCIENCE)

RECORD NOTE BOOK

Name : …….

Reg.No : ……. Year : …….

Branch : ……. Semester : ……

NANDHA ENGINEERING COLLEGE

(AUTONOMOUS)

Erode

RECORD NOTE BOOK

17AIP04 BIG DATA ANALYTICS LABORATORY

VI - Semester

B.TECH (ARTIFICIAL INTELLIGENCE AND DATA SCIENCE)

NANDHA ENGINEERING COLLEGE

(AUTONOMOUS)

Erode

Bonafide Certificate

Register Number:

Certified that this is the Bonafide Record of work done by……
of the **sixth Semester B.TECH – ARTIFICIAL INTELLIGENCE AND DATA SCIENCE** branch during the academic year **2023-2024** in the 17AIP04 BIG DATA ANALYTICS LABORATORY**.**

Staff-in-charge **Head of the Department**

Submitted for the End Semester Practical Examination held on……

Internal Examiner **External Examiner**

17AIP04 BIG DATA ANALYTICS LABORATORY

VI- Semester

B.TECH (ARTIFICIAL INTELLIGENCE AND DATA SCIENCE)

SYLLABUS

1. Downloading and installing Hadoop; Understanding different Hadoop modes. Startup scripts, Configuration files.
2. Hadoop Implementation of file management tasks, such as Adding files and directories, retrieving files and Deleting files
3. Develop a MapReduce program to find the grades of student's.
4. Implement of Matrix Multiplication with Hadoop Map Reduce
5. Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm.
6. Installation of Hive along with practice examples.
7. Installation of HBase, Installing thrift along with Practice examples
8. Practice importing and exporting data from various databases.

INDEX

EX.No.	DATE	NAME OF THE EXPERIMENTS	PAGE No.	MARKS	SIGN
1					
2					
3					
4					
5					
6					
7					
8					
CONTENT BEYOND SYLLABUS					
AVERAGE MARKS AWARDED					

Downloading and installing Hadoop

EX. NO:

DATE:

AIM:

Downloading and installing Hadoop; Understanding different Hadoop modes. Startup scripts, Configuration files.

Prerequisites

Have Git and Docker installed

1.Clone docker-hadoop repo from github

Go to this link https://github.com/big-data-europe/docker-hadoop and clone the repository, if you're using console, you can type the next command:

git clone https://github.com/big-data-europe/docker-hadoop

And you will get the next folder structure

2. Start the necessary containers using docker-compose

Run the next command:

docker-compose up -d

This will start the five necessary containers (if it is the first time you're doing this, you'll have to wait until the download finish)

Docker Compose allows us to run multi-container Docker applications and use multiple commands using only a YAML file.
So calling the previous command will run all the lines in our docker-compose.yml file, and will download the necessary images, if needed.

```
PS C:\Users\TE502801\Desktop\Uni\big data\docker-hadoop> docker-compose up -d
[+] Running 6/6
 - Network docker-hadoop_default  Created                                    0.6s
 - Container namenode             Started                                    4.1s
 - Container nodemanager          Started                                    3.6s
 - Container resourcemanager      Started                                    2.8s
 - Container historyserver        Started                                    3.6s
 - Container datanode             Started                                    3.0s
```

You can check if all the five containers are running typing "docker ps" command.

3. Get in our master node "namenode"

docker exec -it namenode bash

We're getting in our namenode container, which is the master node of our Hadoop cluster. It is basically a mini-linux. It will allow us to manage our HDFS file system.

```
PS C:\Users\TE502801\Desktop\Uni\big data\docker-hadoop> docker exec -it namenode bash
root@55f096a5f729:/#
```

4. Create folder structure to allocate input files

First, you can list all the files in our HDFS system

hdfs dfs -l /

Now, we have to create a /user/root/ file, since hadoop works with this defined structure

hdfs dfs -mkdir -p /user/root

We can verify if it was created correctly

hdfs dfs -ls /user/

5. Download MapReduce script

We will use a .jar file containing the classes needed to execute MapReduce algorithm. You can do this manually, compiling the .java files and zipping them. But we will download the .jar file ready to go

Go to: https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-mapreduce-examples/2.7.1/ and download the one named **hadoop-mapreduce-examples-2.7.1-sources.jar**

6. Download or create the .txt file that you want to process

Most of the time we will use Hadoop to process a fairly large file, maybe several gigabytes, but for this occasion we will use a 1MB file, the classic book "Don Quijote de La Mancha" as plain text, you can use any text you want. You can download the text file I used here: https://gist.github.com/jsdario/6d6c69398cb0c73111e49f1218960f79#file-el_quijote-txt

7. Move our .jar & .txt file into the container

First, move these from where you downloaded them and put them in the cloned repository folder. Then type the next command (you have to be in your normal terminal, not inside the namenode container, you can use "exit" command).

docker cp hadoop-mapreduce-examples-2.7.1-sources.jar namenode:/tmp
Do the same for the .txt file
docker cp el_quijote.txt namenode:/tmp

8. Create the input folder inside our namenode container

Get in the namenode container again
docker exec -it namenode bash

Then
hdfs dfs -mkdir /user/root/input

9. Copy your .txt file from /tmp to the input file

Firts cd to /tmp (cd /tmp)
hdfs dfs -put el_quijote.txt /user/root/input

9.1 Visualize the dashboard in your localhost

Acces to localhost:9870 in your browser

In this case there is just one node (our computer) in later posts we will se how to add more nodes.

10. Run MapReduce

hadoop jar hadoop-mapreduce-examples-2.7.1-sources.jar org.apache.hadoop.examples.WordCount input output

You'll see a large input, but if you have done the past steps right everything should be fine.

11. See the results

hdfs dfs -cat /user/root/output/*

Depending on your text file, youll see something like this

You can check the results accesing to the output folder
hdfs dfs -ls /user/root/putput

12. Turn down the containers

docker-compose down
OUTPUT:

VIVA QUESTIONS:

1. What are the different types of Hadoop configuration files?
2. What are the different modes in which we can configure install Hadoop?
3. How do I download and install Apache Hadoop?
4. How can you set up an HDFS environment discuss the steps involved in configuring and deploying HDFS on a cluster of machines?

MARK ALLOCATION

CONTINUOUS INTERNAL ASSESSMENT		
Preparation & Conduct of experiments	**(50)**	
Observation &Result	**(30)**	
Record	**(10)**	
Viva-voce	**(10)**	
Total	**(100)**	

RESULT:

SIGNATURE OF FACULTY

Hadoop Implementation of file management tasks

EX. NO:

DATE:

AIM:

Hadoop Implementation of file management tasks, such as Adding files and directories, retrieving files and Deleting files.

PROCEDURE:

Before you can run Hadoop programs on data stored in HDFS, you'll need to put the data into HDFS first. Let's create a directory and put a file in it. HDFS has a default working directory of /user/$USER, where $USER is your login user name. This directory isn't automatically created for you, though, so let's create it with the mkdir command. For the purpose of illustration, we use chuck. You should substitute your user name in the example commands.

Step-1 hadoop fs -mkdir /user/chuck hadoop fs -put example.txt hadoop fs -put example.txt /user/chuck

Step-2 Retrieving Files from HDFS The Hadoop command get copies files from HDFS back to the local filesystem. To retrieve example.txt, we can run the following command:

hadoop fs -cat example.txt

Step-3 Deleting Files from HDFS hadoop fs -rm example.txt • Command for creating a directory in hdfs is "hdfs dfs –mkdir /lendicse". • Adding directory is done through the command "hdfs dfs –put lendi_english /". Step-4 Copying Data from NFS to HDFS

Copying from directory command is "hdfs dfs –copyFromLocal /home/lendi/Desktop/shakes/glossary /lendicse/"

- View the file by using the command "hdfs dfs –cat /lendi_english/glossary"
- Command for listing of items in Hadoop is "hdfs dfs –ls hdfs://localhost:9000/".
- Command for Deleting files is "hdfs dfs –rm r /kartheek".

SAMPLE INPUT: Input as any data format of type structured, Unstructured or Semi Structured

OUTPUT:

VIVA-VOCE Questions

1) What is the command used to copy the data from local to hdfs

2) What is the command used to run the hadoop jar file

3) What command is used to remove directory from Hadoop recursively?

4) What is command used to list out directories of Data Node through web tool

MARK ALLOCATION

CONTINUOUS INTERNAL ASSESSMENT		
Preparation & Conduct of experiments	**(50)**	
Observation &Result	**(30)**	
Record	**(10)**	
Viva-voce	**(10)**	
Total	**(100)**	

RESULT:

SIGNATURE OF FACULTY

Develop a MapReduce program

EX. NO:

DATE:

AIM:
To Develop a MapReduce program to find the grades of student's.

PROGRAM:

```
import java.util.Scanner;
public class JavaExample
{
public static void main(String args[])
{
/* This program assumes that the student has 6 subjects,
* thats why I have created the array of size 6. You can
* change this as per the requirement.
*/
int marks[] = new int[6];
int i;
float total=0, avg;
Scanner scanner = new Scanner(System.in);
for(i=0; i<6; i++) {
System.out.print("Enter Marks of Subject"+(i+1)+":");
marks[i] = scanner.nextInt();
total = total + marks[i];
}
scanner.close();
//Calculating average
here avg = total/6;
System.out.print("The student Grade is: ");
if(avg>=80)
{
System.out.print("A");
}
else if(avg>=60 && avg<80)
{
System.out.print("B");
}
else if(avg>=40 && avg<60)
{
System.out.print("C");
}
 else
{
System.out.print("D");
```

```
}
}
}
```

OUTPUT:

1. How do you write a MapReduce program?
2. How is mapping and reducing done in input data?
3. Can MapReduce program be written in any language other than Java justify your answer with sufficient coding examples?
4. Define mapreduce term.

MARK ALLOCATION

CONTINUOUS INTERNAL ASSESSMENT		
Preparation & Conduct of experiments	**(50)**	
Observation &Result	**(30)**	
Record	**(10)**	
Viva-voce	**(10)**	
Total	**(100)**	

RESULT:

SIGNATURE OF FACULTY

Matrix Multiplication with Hadoop Map Reduce

EX.NO:

DATE:

AIM:
Implement of Matrix Multiplication with Hadoop Map Reduce

PROGRAM:

Mapper Logic :

```
import java.io.IOException;

import org.apache.hadoop.conf.*;

import org.apache.hadoop.io.*;

import org.apache.hadoop.mapreduce.*;

public class MatrixMapper extends Mapper<LongWritable, Text, Text, Text>

{

public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException

{

Configuration conf = context.getConfiguration();

int m = Integer.parseInt(conf.get("m"));

int p = Integer.parseInt(conf.get("p"));

String line = value.toString();

String[] indicesAndValue = line.split(",");

Text outputKey = new Text();

Text outputValue = new Text();

if (indicesAndValue[0].equals("M"))

{

        for (int k = 0; k < p; k++)
```

```
{
outputKey.set(indicesAndValue[1] + "," + k);
outputValue.set("M," + indicesAndValue[2] + "," + indicesAndValue[3]);
context.write(outputKey, outputValue);
}}
else {
for (int i = 0; i < m; i++)
{
outputKey.set(i + "," + indicesAndValue[2]);
outputValue.set("N," + indicesAndValue[1] + "," + indicesAndValue[3]); context.write(outputKey,
outputValue);
}}}}
```

Reducer Logic :

```
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
public class MatrixReducer extends Reducer<Text, Text, Text, Text>
{
public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException
{
String[] value;
HashMap<Integer, Float> hashA = new HashMap<Integer, Float>();
HashMap<Integer, Float> hashB = new HashMap<Integer, Float>();
for (Text val : values)
{
```

```
value = val.toString().split(",");
if (value[0].equals("M"))
{
hashA.put(Integer.parseInt(value[1]), Float.parseFloat(value[2]));
}
else
{
hashB.put(Integer.parseInt(value[1]), Float.parseFloat(value[2]));
} }
int n = Integer.parseInt(context.getConfiguration().get("n")); float result = 0.0f;
float a_ij; float b_jk;
for (int j = 0; j < n; j++)
{
a_ij = hashA.containsKey(j) ? hashA.get(j) : 0.0f;
b_jk = hashB.containsKey(j) ? hashB.get(j) : 0.0f;
result += a_ij * b_jk;
}
if (result != 0.0f)
{
context.write(null, new Text(key.toString() + "," + Float.toString(result)));
} } }
```

Driver Logic :

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
```

```
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

public class MatrixDriver
{
	public static void main(String[] args) throws Exception
	{
		Configuration conf = new Configuration();
		// M is an m-by-n matrix; N is an n-by-p matrix.
		conf.set("m", "2");
		conf.set("n", "2");
		conf.set("p", "2");
		Job job = Job.getInstance(conf, "MatrixMultiplication");
		job.setJarByClass(MatrixDriver.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(Text.class);
		job.setMapperClass(MatrixMapper.class);
		job.setReducerClass(MatrixReducer.class);
		job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);
		FileInputFormat.addInputPath(job, new Path(args[0]));
		FileOutputFormat.setOutputPath(job, new Path(args[1]));
		job.submit();
```

```
}}
```

OUTPUT:

VIVA QUESTION:

1. How to do matrix multiplication in Hadoop?
2. How do you reduce matrix multiplication?
3. What is Hadoop MapReduce?
4. What is the implementation language of the Hadoop MapReduce framework?

MARK ALLOCATION

CONTINUOUS INTERNAL ASSESSMENT		
Preparation & Conduct of experiments	**(50)**	
Observation &Result	**(30)**	
Record	**(10)**	
Viva-voce	**(10)**	
Total	**(100)**	

RESULT:

SIGNATURE OF FACULTY

Word Count Map Reduce program to understand Map Reduce Paradigm

EX.NO:

DATE:

AIM:

Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm

PROCEDURE:

Steps to execute MapReduce word count example

- Create a text file in your local machine and write some text into it.
  $ nano data.txt
- Check the text written in the data.txt file.
  $ cat data.txt

In this example, we find out the frequency of each word exists in this text file. Create a directory in HDFS, where to kept text file.
$ hdfs dfs -mkdir /test
Upload the data.txt file on HDFS in the specific directory.
$ hdfs dfs -put /home/codegyani/data.txt /test

Write the code.
Create the jar file of this program and name it countworddemo.jar.
Run the jar file
hadoop jar /home/codegyani/wordcountdemo.jar com.javatpoint.WC_Runner /test/data.txt /r_output
The output is stored in /r_output/part-00000
now execute the command to see the output.
hdfs dfs -cat /r_output/part-00000

WC_Mapper.java:

```
package com.javatpoint;

import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class WC_Mapper extends MapReduceBase implements
Mapper<LongWritable,Text,Text,IntWritable>{
   private final static IntWritable one = new IntWritable(1);
   private Text word = new Text();
   public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
         Reporter reporter) throws IOException{
      String line = value.toString();
      StringTokenizer  tokenizer = new StringTokenizer(line);
      while (tokenizer.hasMoreTokens()){
         word.set(tokenizer.nextToken());
         output.collect(word, one);
      }
   }

}
```

WC_Reducer.java

```
package com.javatpoint;
   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 WC_Reducer  extends MapReduceBase implements
Reducer<Text,IntWritable,Text,IntWritable> {
   public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,IntWritable> output,
    Reporter reporter) throws IOException {
   int sum=0;
   while (values.hasNext()) {
   sum+=values.next().get();
   }
   output.collect(key,new IntWritable(sum));
   }
   }
```

WC_Runner.java

```
package com.javatpoint;

   import java.io.IOException;
   import org.apache.hadoop.fs.Path;
   import org.apache.hadoop.io.IntWritable;
   import org.apache.hadoop.io.Text;
   import org.apache.hadoop.mapred.FileInputFormat;
   import org.apache.hadoop.mapred.FileOutputFormat;
   import org.apache.hadoop.mapred.JobClient;
   import org.apache.hadoop.mapred.JobConf;
   import org.apache.hadoop.mapred.TextInputFormat;
   import org.apache.hadoop.mapred.TextOutputFormat;
   public class WC_Runner {
      public static void main(String[] args) throws IOException{
         JobConf conf = new JobConf(WC_Runner.class);
         conf.setJobName("WordCount");
         conf.setOutputKeyClass(Text.class);
         conf.setOutputValueClass(IntWritable.class);
         conf.setMapperClass(WC_Mapper.class);
         conf.setCombinerClass(WC_Reducer.class);
         conf.setReducerClass(WC_Reducer.class);
         conf.setInputFormat(TextInputFormat.class);
         conf.setOutputFormat(TextOutputFormat.class);
         FileInputFormat.setInputPaths(conf,new Path(args[0]));
         FileOutputFormat.setOutputPath(conf,new Path(args[1]));
         JobClient.runJob(conf);
      }
   }
```

OUTPUT:

VIVA QUESTIONS:

1. What is the MapReduce programming paradigm?
2. What is MapReduce for word count?
3. How do I run a MapReduce program?
4. How do I run a WordCount program in Cloudera?

MARK ALLOCATION

CONTINUOUS INTERNAL ASSESSMENT		
Preparation & Conduct of experiments	**(50)**	
Observation &Result	**(30)**	
Record	**(10)**	
Viva-voce	**(10)**	
Total	**(100)**	

RESULT:

SIGNATURE OF FACULTY

Installation of Hive along with practice examples

EX.NO:

DATE:

AIM:
Installation of Hive along with practice examples

PROCEDURE:
Steps to install Apache Hive

Download the Apache Hive tar file.

http://mirrors.estointernet.in/apache/hive/hive-1.2.2/

DUnzip the downloaded tar file.

tar -xvf apache-hive-1.2.2-bin.tar.gz

DOpen the bashrc file.

$ sudo nano ~/.bashrc

DNow, provide the following HIVE_HOME path.

export HIVE_HOME=/home/codegyani/apache-hive-1.2.2-bin

export PATH=$PATH:/home/codegyani/apache-hive-1.2.2-bin/bin

DUpdate the environment variable.

$ source ~/.bashrc

DLet's start the hive by providing the following command.

$ hive

Hive - Create Database

In Hive, the database is considered as a catalog or namespace of tables. So, we can maintain multiple tables within a database where a unique name is assigned to each table. Hive also provides a default database with a name default.

- Initially, we check the default database provided by Hive. So, to check the list of existing databases, follow the below command: -

1. hive> show databases;

Here, we can see the existence of a default database provided by Hive.

- Let's create a new database by using the following command: -

1. hive> create database demo;

So, a new database is created.

- Let's check the existence of a newly created database.

1. hive> show databases;
2. If we want to suppress the warning generated by Hive on creating the database with the same name, follow the below command: -
3. hive> create a database if not exists demo;

- Hive also allows assigning properties with the database in the form of key-value pair.

1. hive>create the database demo
   >WITH DBPROPERTIES ('creator' = 'NANDHA', 'date' = '2019-06-03');

2. Let's retrieve the information associated with the database.
3. hive> describe database extended demo;

Hive - Drop Database

In this section, we will see various ways to drop the existing database.

1. hive> show databases;

Now, drop the database by using the following command.

2. hive> drop database demo;
3. Let's check whether the database is dropped or not.
4. hive> show databases;
5. hive> drop database if exists demo;
6. hive> drop database if exists demo cascade;

OUTPUT:

VIVA QUESTIONS:

1. How to install Hive with Spark?
2. Why install Hive?
3. How much is Hive installation?
4. Can you install Hive without hub?

MARK ALLOCATION

CONTINUOUS INTERNAL ASSESSMENT		
Preparation & Conduct of experiments	**(50)**	
Observation &Result	**(30)**	
Record	**(10)**	
Viva-voce	**(10)**	
Total	**(100)**	

RESULT:

SIGNATURE OF FACULTY

Installation of HBase, Installing thrift along with Practice examples

EX.NO:

DATE:

AIM:
Installation of HBase, Installing thrift along with Practice examples .

PROCEDURE:
The prerequisite for HBase installation are Java and Hadoop installed on your Linux machine.

Hbase can be installed in three modes: standalone, Pseudo Distributed mode and Fully Distributed mode.

Download the Hbase package from http://www.interior-dsgn.com/apache/hbase/stable/ and unzip it with the below commands.

$cd usr/local/$wget http://www.interior-dsgn.com/apache/hbase/stable/hbase-0.98.8-hadoop2-bin.tar.gz
$tar -zxvf hbase-0.98.8-hadoop2-bin.tar.gz
Login as super user as shown below

1. $su
2. $password: enter your password here
3. mv hbase-0.99.1/* Hbase/

Configuring HBase in Standalone Mode
Set the java Home for HBase and open hbase-env.sh file from the conf folder. Edit JAVA_HOME environment variable and change the existing path to your current JAVA_HOME variable as shown below.

cd /usr/local/Hbase/conf
gedit hbase-env.sh
Replace the existing JAVA_HOME value with your current value as shown below.

export JAVA_HOME=/usr/lib/jvm/java-1.7.0
Inside /usr/local/Hbase you will find hbase-site.xml. Open it and within configuration add the below code.

```
<configuration>
  //Here you have to set the path where you want HBase to store its files.
<property>
<name>hbase.rootdir</name>
<value>file:/home/hadoop/HBase/HFiles</value>
</property>

  //Here you have to set the path where you want HBase to store its built in zookeeper  files.
<property>
<name>hbase.zookeeper.property.dataDir</name>
<value>/home/hadoop/zookeeper</value>
</property>
</configuration>
```

Now start the Hbase by running the start-hbase.sh present in the bin folder of Hbase.

$cd /usr/local/HBase/bin
$./start-hbase.sh

Cloudera VM is recommended as it has Hbase preinstalled on it.

Starting Hbase: Type Hbase shell in terminal to start the hbase.

PROGRAM1:

We have to import data present in the file into an HBase table by creating it through Java API.

Data_file.txt contains the below data

1. 1,India,Bihar,Champaran,2009,April,P1,1,5
2. 2,India, Bihar,Patna,2009,May,P1,2,10
3. 3,India, Bihar,Bhagalpur,2010,June,P2,3,15
4. 4,United States,California,Fresno,2009,April,P2,2,5
5. 5,United States,California,Long Beach,2010,July,P2,4,10
6. 6,United States,California,San Francisco,2011,August,P1,6,20

The Java code is shown below

This data has to be inputted into a new HBase table to be created through JAVA API. Following column families have to be created

"sample,region,time.product,sale,profit".
Column family region has three column qualifiers: country, state, city

Column family Time has two column qualifiers: year, month

Jar Files
Make sure that the following jars are present while writing the code as they are required by the HBase.

commons-loging-1.0.4
commons-loging-api-1.0.4
hadoop-core-0.20.2-cdh3u2
hbase-0.90.4-cdh3u2
log4j-1.2.15
zookeper-3.3.3-cdh3u0
Program Code

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.HColumnDescriptor;
import org.apache.hadoop.hbase.HTableDescriptor;
import org.apache.hadoop.hbase.client.HBaseAdmin;
import org.apache.hadoop.hbase.client.HTable;
```

```
import org.apache.hadoop.hbase.client.Put;
import org.apache.hadoop.hbase.util.Bytes;


public class readFromFile {
   public static void main(String[] args) throws IOException{
      if(args.length==1)
         {
         Configuration conf = HBaseConfiguration.create(new Configuration());
         HBaseAdmin hba = new HBaseAdmin(conf);
         if(!hba.tableExists(args[0])){
            HTableDescriptor ht = new HTableDescriptor(args[0]);
            ht.addFamily(new HColumnDescriptor("sample"));
            ht.addFamily(new HColumnDescriptor("region"));
            ht.addFamily(new HColumnDescriptor("time"));
            ht.addFamily(new HColumnDescriptor("product"));
            ht.addFamily(new HColumnDescriptor("sale"));
            ht.addFamily(new HColumnDescriptor("profit"));
            hba.createTable(ht);
            System.out.println("New Table Created");

            HTable table = new HTable(conf,args[0]);

            File f = new File("/home/training/Desktop/data");
            BufferedReader br = new BufferedReader(new FileReader(f));
            String line = br.readLine();
            int i =1;
            String rowname="row";
            while(line!=null && line.length()!=0){
               System.out.println("Ok till here");
               StringTokenizer tokens = new StringTokenizer(line,",");
               rowname = "row"+i;
               Put p = new Put(Bytes.toBytes(rowname));
               p.add(Bytes.toBytes("sample"),Bytes.toBytes("sampleNo."),
Bytes.toBytes(Integer.parseInt(tokens.nextToken())));
               p.add(Bytes.toBytes("region"),Bytes.toBytes("country"),Bytes.toBytes(tokens.nextToken()));
               p.add(Bytes.toBytes("region"),Bytes.toBytes("state"),Bytes.toBytes(tokens.nextToken()));
               p.add(Bytes.toBytes("region"),Bytes.toBytes("city"),Bytes.toBytes(tokens.nextToken()));
               p.add(Bytes.toBytes("time"),Bytes.toBytes("year"),Bytes.toBytes(Integer.parseInt(tokens.nextToken())));
               p.add(Bytes.toBytes("time"),Bytes.toBytes("month"),Bytes.toBytes(tokens.nextToken()));
               p.add(Bytes.toBytes("product"),Bytes.toBytes("productNo."),Bytes.toBytes(tokens.nextToken()));
               p.add(Bytes.toBytes("sale"),Bytes.toBytes("quantity"),Bytes.toBytes(Integer.parseInt(tokens.nextToken())));
               p.add(Bytes.toBytes("profit"),Bytes.toBytes("earnings"),Bytes.toBytes(tokens.nextToken()));
               i++;
               table.put(p);
               line = br.readLine();
            }
               br.close();
               table.close();
            }
         else
            System.out.println("Table Already exists.Please enter another table name");
      }
      else
         System.out.println("Please Enter the table name through command line");
```

```
    }
}
```

OUTPUT:

VIVA QUESTIONS:

1. What are the HBase installation modes?
2. How do I download HBase on Windows 10?
3. How do I start a HBase server?
4. Can we install HBase without Hadoop?

MARK ALLOCATION

CONTINUOUS INTERNAL ASSESSMENT		
Preparation & Conduct of experiments	**(50)**	
Observation &Result	**(30)**	
Record	**(10)**	
Viva-voce	**(10)**	
Total	**(100)**	

RESULT:

SIGNATURE OF FACULTY

Practice importing and exporting data from various databases

EX.NO:

DATE:

AIM:

To Practice importing and exporting data from various databases.

PROGRAM:

```
library
(ggplot2)
library (dplyr)
library (maps)
library (ggmap)
library
(mongolite)
library
(lubridate)
library
(gridExtra)
crimes=data.table::fread("Crimes_2001_to_present.csv")
names (crimes)
```

Output:

ID' 'Case Number' 'Date' 'Block' 'IUCR' 'Primary Type' 'Description' 'Location Description' 'Arrest''Domestic' 'Beat' 'District' 'Ward' 'Community Area' 'FBI Code'

'XCoordinate' 'Y Coordinate' 'Year' 'Updated On' 'Latitude' 'Longitude' 'Location'

1) Let’s remove spaces in the column names to avoid any problems when we query it from MongoDB.

```
names(crimes) = gsub("
","",names(crimes))names(crimes)
```

```
'ID' 'CaseNumber' 'Date' 'Block' 'IUCR' 'PrimaryType' 'Description'
'LocationDescription' 'Arrest' 'Domestic' 'Beat' 'District' 'Ward' 'CommunityArea'
'FBICode' 'XCoordinate' 'YCoordinate' 'Year' 'UpdatedOn' 'Latitude' 'Longitude'
'Location'
```

2) Let’s use the insert function from the mongolite package to insert rows to a collection in MongoDB.Let’s create a database called Chicago and call the collection crimes.

```
my_collection = mongo(collection = "crimes", db = "Chicago") #createconnection, database and collection
```

```
my_collection$insert(crimes)
```

3) Let’s check if we have inserted the “crimes” data.

my_collection$count()

```
6261148
```

We see that the collection has 6261148 records.

4) First, let’s look what the data looks like by displaying one record:

my_collection$iterate()$one()

```
$ID
1454164
$Case Number
 G185744'
$Date
 04/01/2001 06:00:00 PM'
$Block
 049XX N MENARD AV'
$IUCR
```

0910'

$Primary Type

MOTOR VEHICLE THEFT'

$Description

AUTOMOBILE'

$Location Description

STREET'

$Arrest

false'

$Domestic

false'

$Beat

1622

$District

16

$FBICode

07'

$XCoordinate

1136545

$YCoordinate

1932203

$Year

2001

$Updated On

08/17/2015 03:03:40 PM'

$Latitude

41.970129962

$Longitude

87.773302309

$Location

(41.970129962, -87.773302309)'

5) How many distinct “Primary Type” do we have?

length(my_collection$distinct("PrimaryType"))

35

As shown above, there are 35 different crime primary types in the database. We will see the patterns of the most common crime types below.

6) Now, let’s see how many domestic assaults there are in the collection.

```
my_collection$count('{"PrimaryType":"ASSAULT", "Domestic" : "true" }')
```

```
82470
```

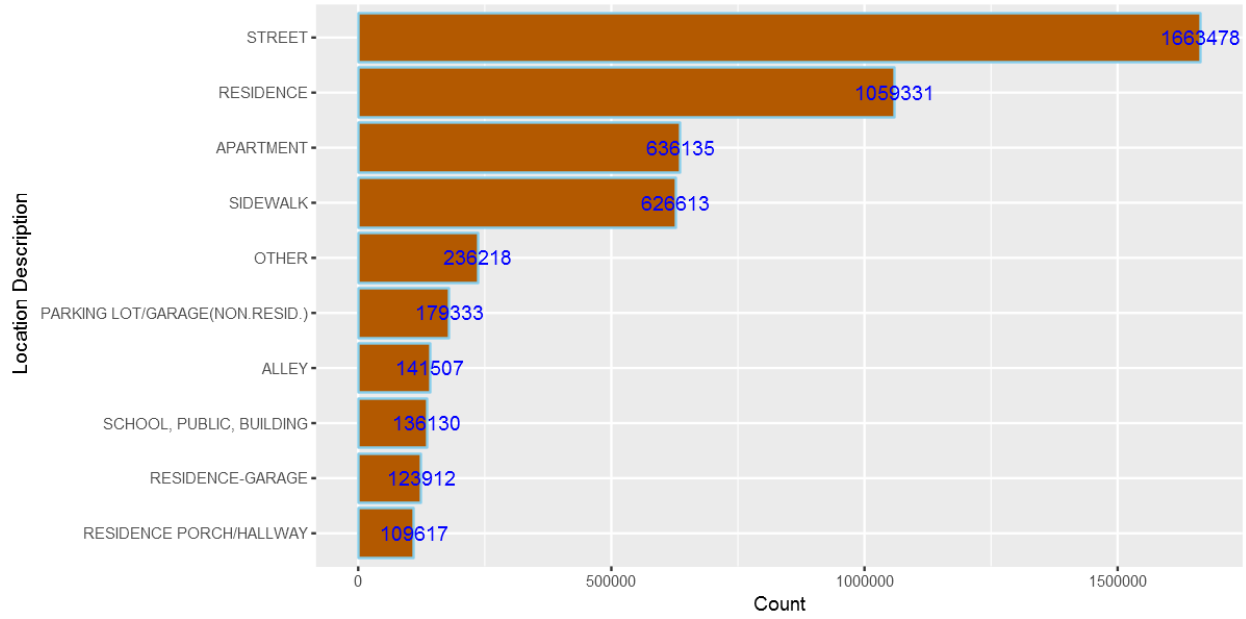
7) To get the filtered data and we can also retrieve only the columns of interest.

```
query1= my_collection$find('{"PrimaryType" : "ASSAULT", "Domestic" :"true" }')
query2= my_collection$find('{"PrimaryType" : "ASSAULT", "Domestic" :"true" }',
 fields = '{"_id":0, "PrimaryType":1, "Domestic":1}')ncol(query1) # with all the columns
ncol(query2) # only the selected columns
```

```
22
2
```

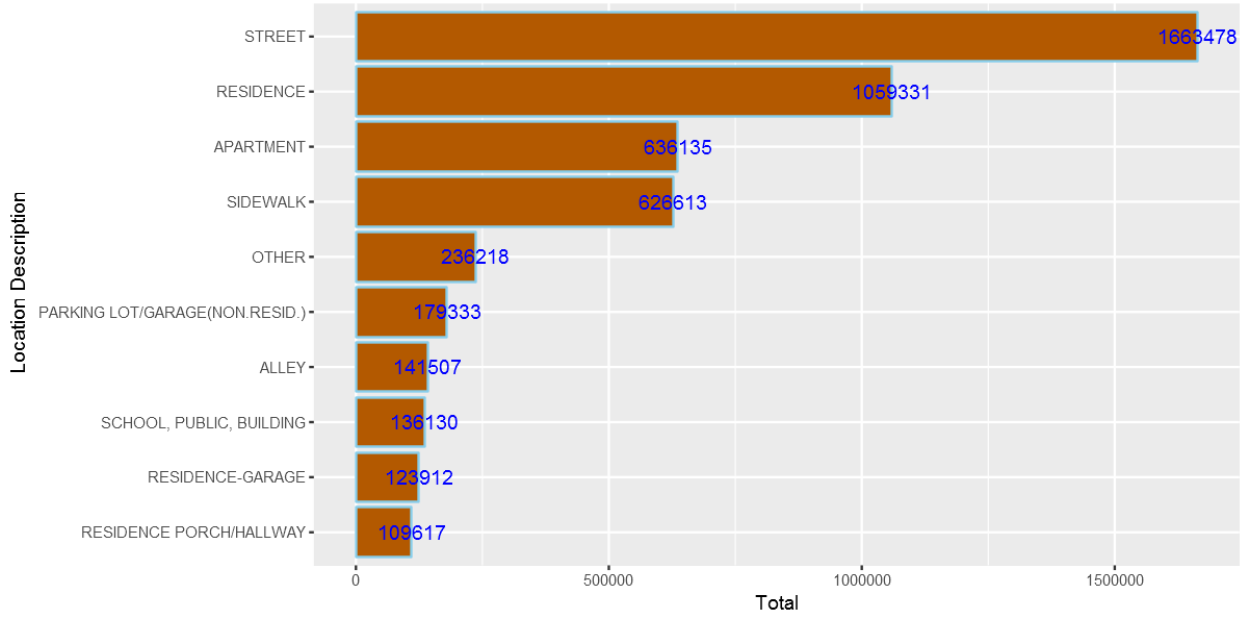
8) To find out “Where do most crimes take place?” use the following command.

```
my_collection$aggregate('[{"$group":{"_id":"$LocationDescription", "Count": {"$sum":1}}}]')%>%na.omit()%>%

arrange(desc(Count))%>%head(10)%>%

ggplot(aes(x=reorder(`_id`,Count),y=Count))+

geom_bar(stat="identity",color='skyblue',fill='#b35900')+geom_text(aes(label = Count), color = "blue") +coord_flip()+xlab("Location Description")
```



9) If loading the entire dataset we are working with does not slow down our analysis, we can use data.table or dplyr but when dealing with big data, using MongoDB can give us performance boost as the whole data will not be loaded into memory. We can reproduce the above plot without using MongoDB, like so:

```
crimes%>%group_by(`LocationDescription`)%>%summarise(Total=n())%>%
arrange(desc(Total))%>%head(10)%>%

ggplot(aes(x=reorder(`LocationDescription`,Total),y=Total))+

geom_bar(stat="identity",color='skyblue',fill='#b35900')+geom_text(aes(label =
Total), color = "blue") +coord_flip()+xlab("Location Description")
```



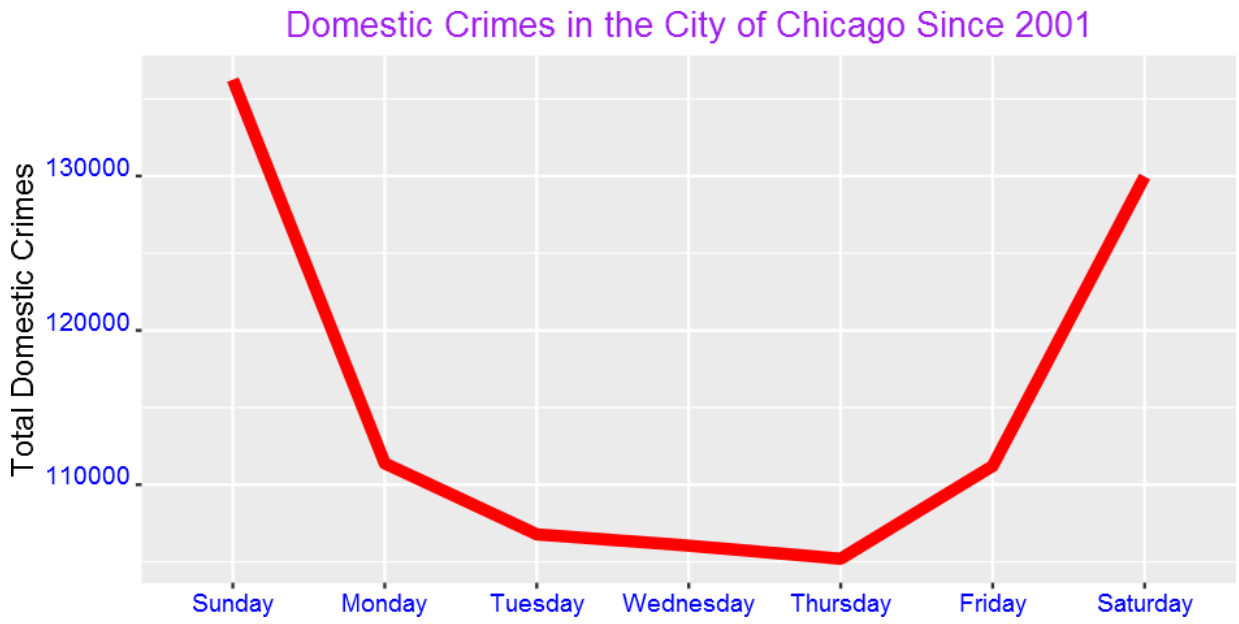
10) What if we want to query all records for certain columns only? This helps us to load only the columns we want and to save memory for our analysis.

```
query3= my_collection$find('{}', fields = '{"_id":0, "Latitude":1, "Longitude":1,"Year":1}')
```

11) We can explore any patterns of domestic crimes. For example, are they common in certain days/hours/months?

```
domestic=my_collection$find('{"Domestic":"true"}', fields = '{"_id":0,
"Domestic":1,"Date":1}')

domestic$Date= mdy_hms(domestic$Date)

domestic$Weekday = weekdays(domestic$Date)

domestic$Hour = hour(domestic$Date) domestic$month

= month(domestic$Date,label=TRUE)

WeekdayCounts = as.data.frame(table(domestic$Weekday))

WeekdayCounts$Var1 = factor(WeekdayCounts$Var1, ordered=TRUE, levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday","Saturday"))

ggplot(WeekdayCounts, aes(x=Var1, y=Freq)) +
geom_line(aes(group=1),size=2,color="red") + xlab("Day of the Week") +
ylab("Total Domestic Crimes")+

ggtitle("Domestic Crimes in the City of Chicago Since 2001")+
```

```
  theme(axis.title.x=element_blank(),axis.text.y =
  element_text(color="blue",size=11,angle=0,hjust=1,vjust=0),
 axis.text.x = element_text(color="blue",size=11,angle=0,hjust=.5,vjust=.5),
 axis.title.y = element_text(size=14),
 plot.title=element_text(size=16,color="purple",hjust=0.5))
```

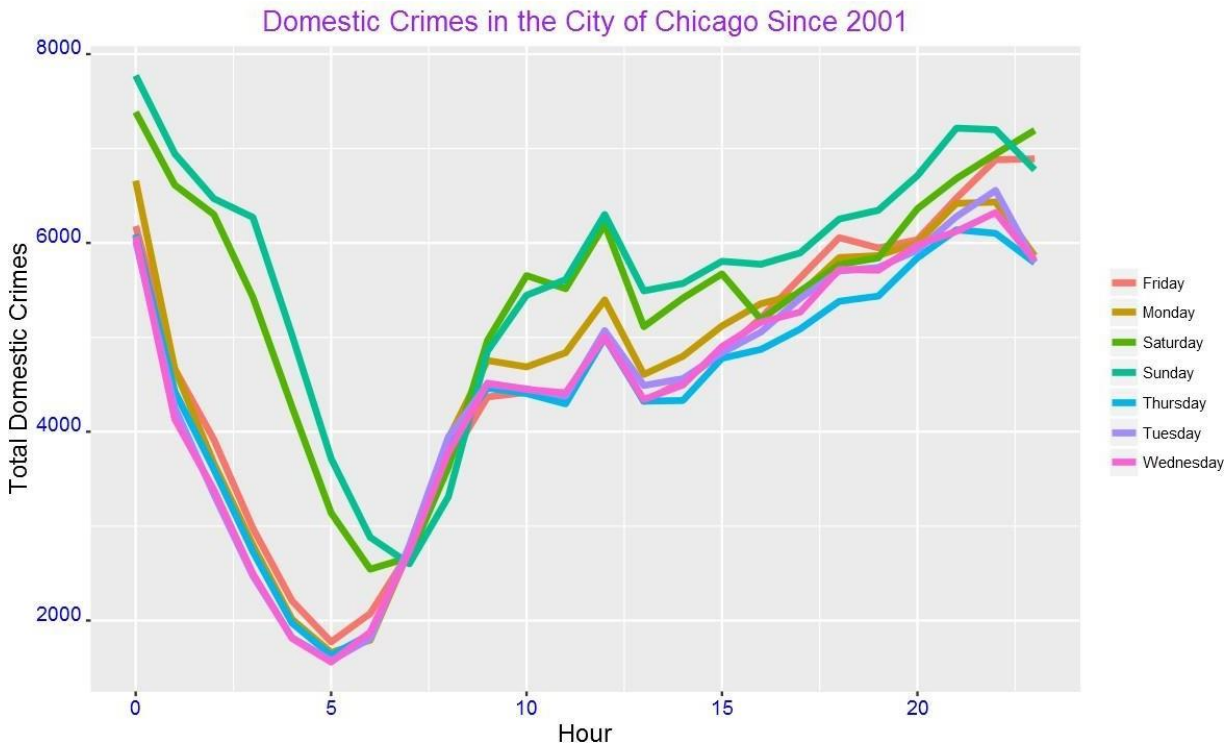


12) Domestic crimes are common over the weekend than in weekdays? What could be the reason?

We can also see the pattern for each day by hour:

```
  DayHourCounts = as.data.frame(table(domestic$Weekday,
  domestic$Hour))DayHourCounts$Hour =
  as.numeric(as.character(DayHourCounts$Var2))
```

```
ggplot(DayHourCounts, aes(x=Hour, y=Freq)) + geom_line(aes(group=Var1,color=Var1),
size=1.4)+ylab("Count")+
    ylab("Total Domestic Crimes")+ggtitle("Domestic Crimes in the City ofChicagoSince
    2001")+
theme(axis.title.x=element_text(size=14),axis.text.y =
element_text(color="blue",size=11,angle=0,hjust=1,vjust=0),
 axis.text.x = element_text(color="blue",size=11,angle=0,hjust=.5,vjust=.5),
 axis.title.y = element_text(size=14),
   legend.title=element_blank(),
   plot.title=element_text(size=16,color="purple",hjust=0.5))
```

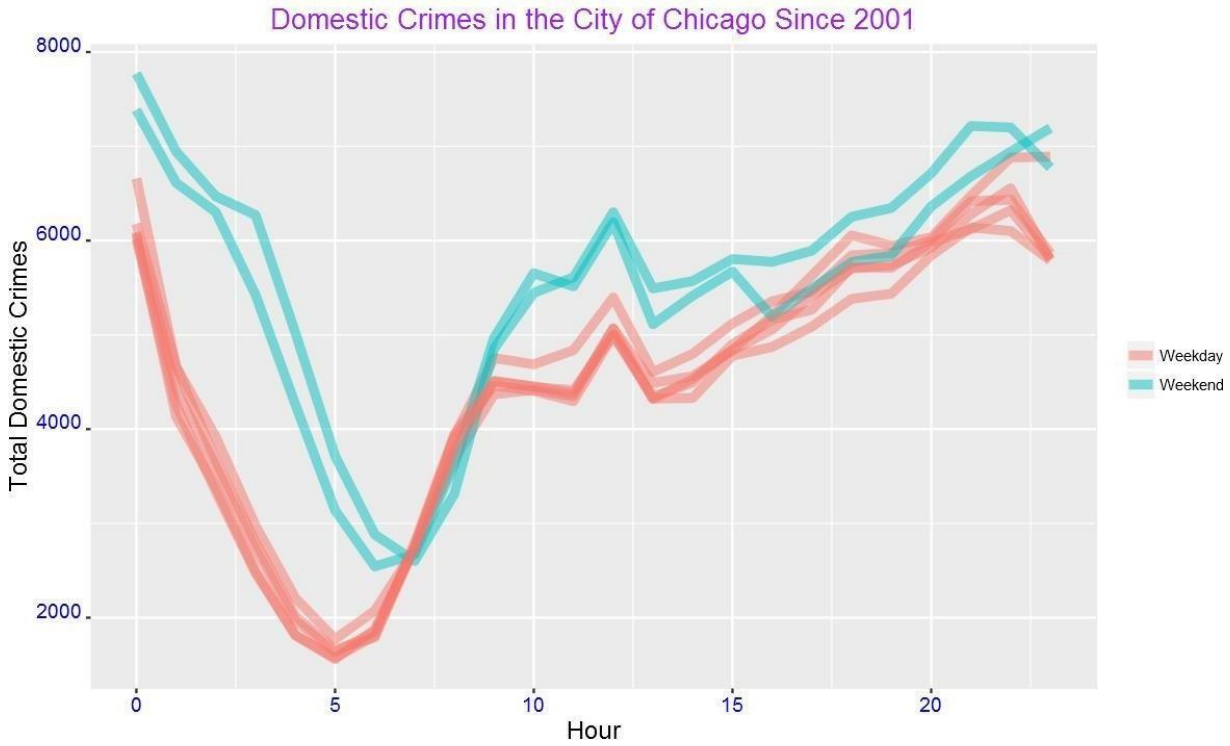


13) The crimes peak mainly around mid-night. We can also use one color for weekdays and another color for weekend as shown below.

```
DayHourCounts$Type = ifelse((DayHourCounts$Var1 == "Sunday")
|(DayHourCounts$Var1 == "Saturday"), "Weekend","Weekday")

ggplot(DayHourCounts, aes(x=Hour, y=Freq)) + geom_line(aes(group=Var1,color=Type),
size=2, alpha=0.5) +

ylab("Total Domestic Crimes")+ggtitle("Domestic Crimes in the City ofChicagoSince
2001")+

theme(axis.title.x=element_text(size=14),axis.text.y =
element_text(color="blue",size=11,angle=0,hjust=1,vjust=0),

axis.text.x = element_text(color="blue",size=11,angle=0,hjust=.5,vjust=.5),
```

legend.title=element_blank(),

plot.title=element_text(size=16,color="purple",hjust=0.5))



14) The difference between weekend and weekdays are clearer from this figure than from the previous plot. We can also see the above pattern from a heat map.

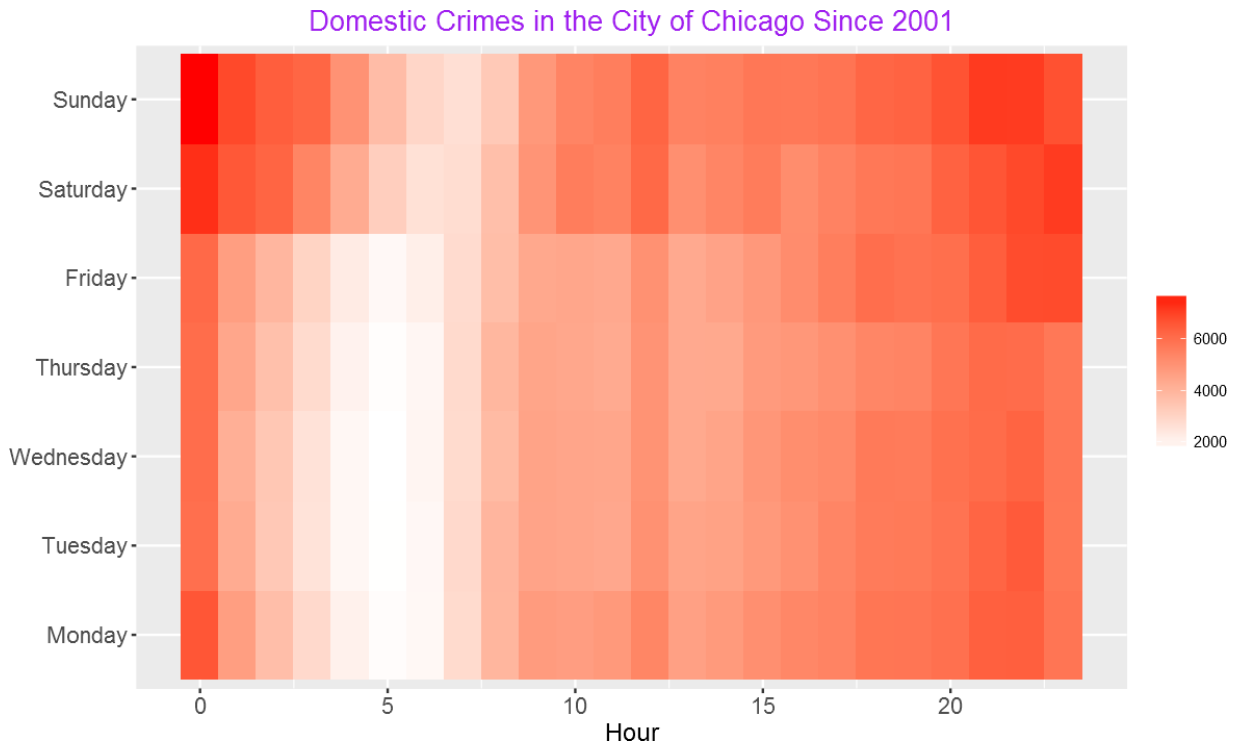
DayHourCounts$Var1 = factor(DayHourCounts$Var1, ordered=TRUE, levels=c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"))

ggplot(DayHourCounts, aes(x = Hour, y = Var1)) + geom_tile(aes(fill = Freq)) + scale_fill_gradient(name="Total MV Thefts", low="white", high="red") +

ggtitle("Domestic Crimes in the City of Chicago Since 2001")+theme(axis.title.y = element_blank())+ylab("")+theme(axis.title.x=element_text(size=14),axis.text.y =element_text(size=13),axis.text.x = element_text(size=13), axis.title.y =

element_text(size=14),

legend.title=element_blank(),plot.title=element_text(size=16,color="purple",hjus t=0.5))



15) Let's see the pattern of other crime types. Let's focus on four of the most common ones.

```
crimes=my_collection$find('{}', fields = '{"_id":0,
"PrimaryType":1,"Year":1}')
```

crimes%>%group_by(PrimaryType)%>%summarize(Count=n())%>%arrange (desc(Count))%>%head(4)

Imported 6261148 records. Simplifying into dataframe...

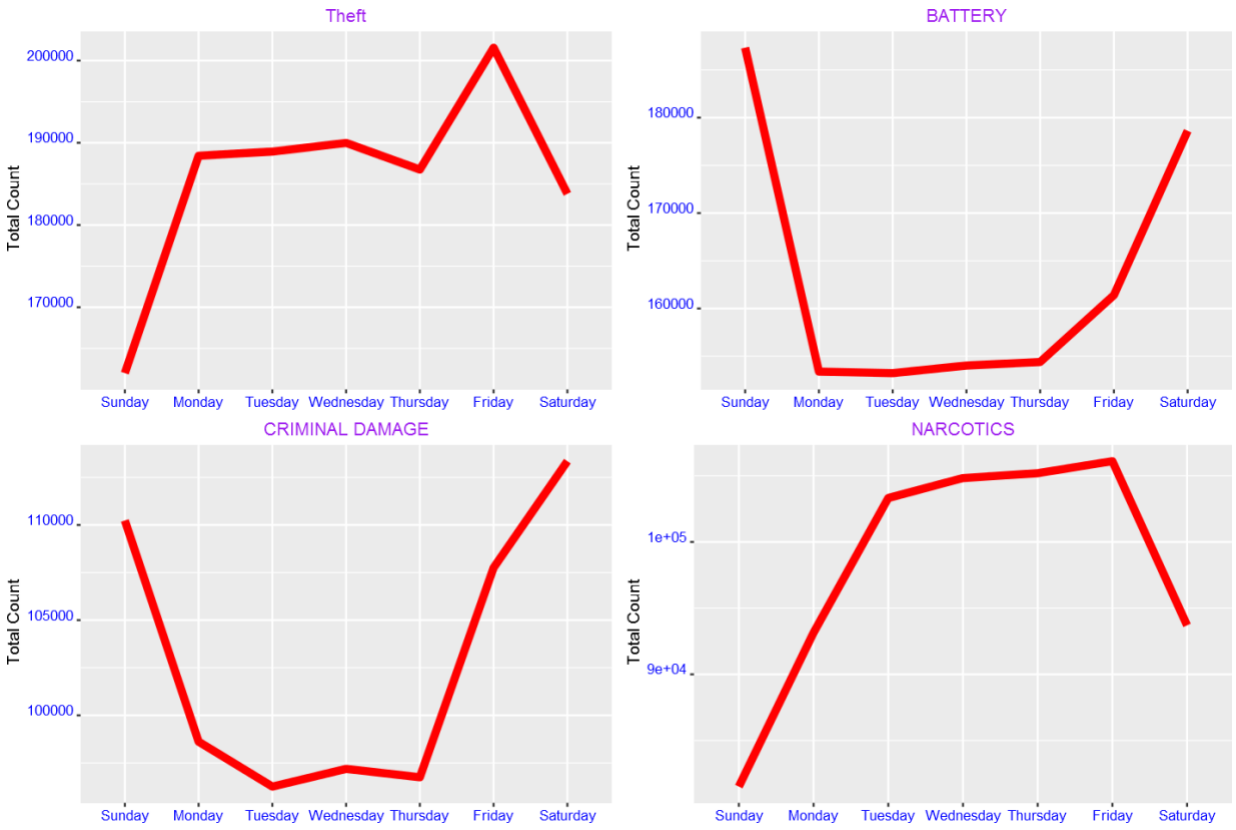
PrimaryType	**Count**

THEFT	**1301434**
BATTERY	**1142377**
CRIMINAL DAMAGE	**720143**
NARCOTICS	**687790**

16) As shown in the table above, the most common crime type is theft followed by battery. Narcotics is fourth most common while criminal damage is the third most common crime type in the city of Chicago. Now, let's generate plots by day and hour.

```
four_most_common=crimes%>%group_by(PrimaryType)%>%summarize(C
ount=n())%>%arrange(desc(Count))%>%head(4)

four_most_common=four_most_common$PrimaryType

crimes=my_collection$find('{}', fields = '{"_id":0, "PrimaryType":1,"Date":1}')

crimes=filter(crimes,PrimaryType %in%four_most_common)

crimes$Date= mdy_hms(crimes$Date)

crimes$Weekday = weekdays(crimes$Date)

crimes$Hour = hour(crimes$Date)

crimes$month=month(crimes$Date,label =

TRUE)


g = function(data){WeekdayCounts = as.data.frame(table(data$Weekday))
```

```
WeekdayCounts$Var1 = factor(WeekdayCounts$Var1, ordered=TRUE,
levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
"Friday","Saturday"))

ggplot(WeekdayCounts, aes(x=Var1, y=Freq)) +
geom_line(aes(group=1),size=2,color="red") + xlab("Day of the Week") +
theme(axis.title.x=element_blank(),axis.text.y =
element_text(color="blue",size=10,angle=0,hjust=1,vjust=0),

 axis.text.x = element_text(color="blue",size=10,angle=0,hjust=.5,vjust=.5),

 axis.title.y = element_text(size=11),

 plot.title=element_text(size=12,color="purple",hjust=0.5)) }

g1=g(filter(crimes,PrimaryType=="THEFT"))+ggtitle("Theft")+ylab("Total
Count")

g2=g(filter(crimes,PrimaryType=="BATTERY"))+ggtitle("BATTERY")+yl ab("
Total Count")

g3=g(filter(crimes,PrimaryType=="CRIMINAL
DAMAGE"))+ggtitle("CRIMINAL DAMAGE")+ylab("Total Count")

g4=g(filter(crimes,PrimaryType=="NARCOTICS"))+ggtitle("NARCOTICS ")+ylab("Total
Count")

grid.arrange(g1,g2,g3,g4,ncol=2)
```



From the plots above, we see that theft is most common on Friday. Battery and criminal damage, on the other hand, are highest at weekend. We also observe that narcotics decreases over weekend.

We can also see the pattern of the above four crime types by hour:

17) We can also see a map for domestic crimes only:

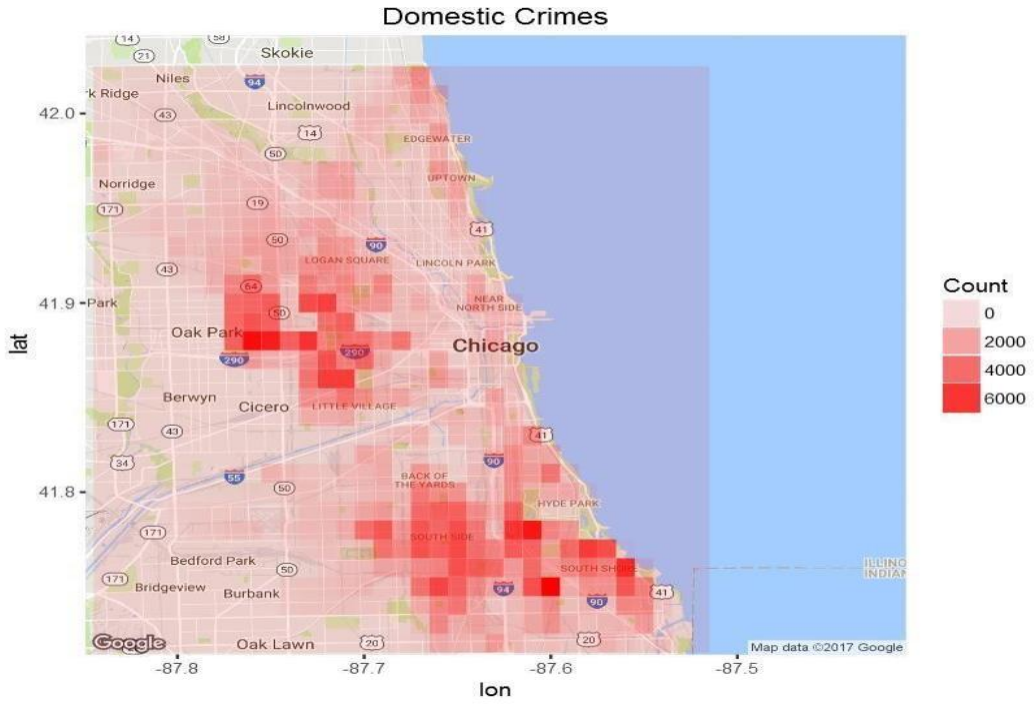
```
domestic=my_collection$find('{"Domestic":"true"}', fields ='{"_id":0,"Latitude":1,
"Longitude":1,"Year":1}')
```

```
LatLonCounts=as.data.frame(table(round(domestic$Longitude,2),round(domest ic$Latitude,2)))
```

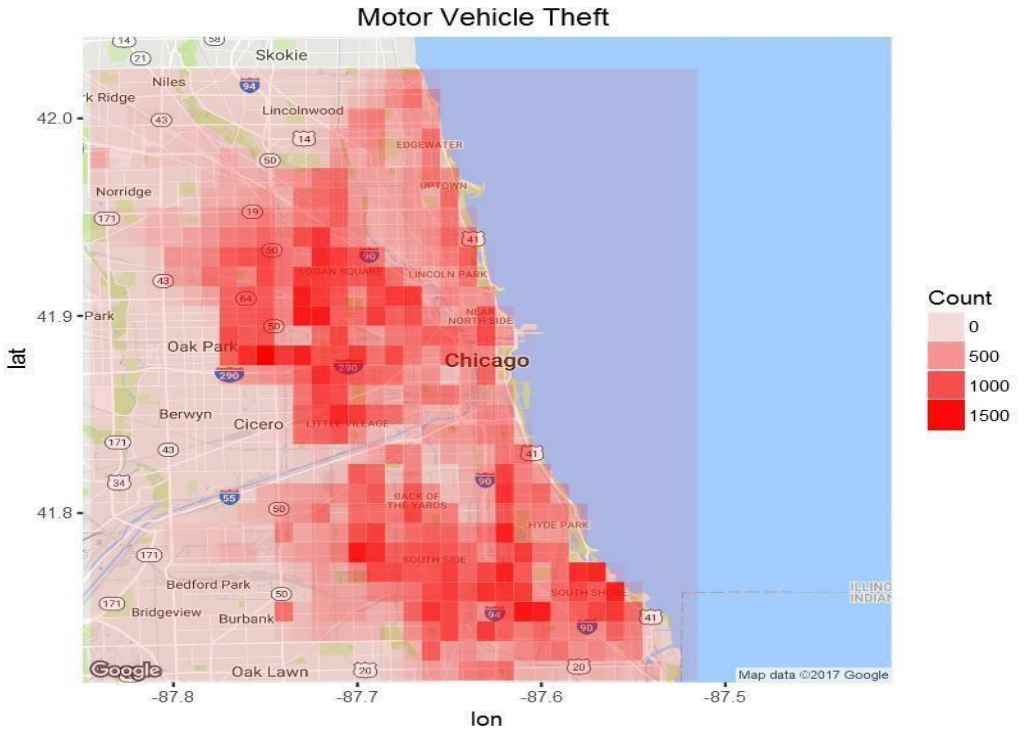
```
LatLonCounts$Long = as.numeric(as.character(LatLonCounts$Var1))LatLonCounts$Lat =
as.numeric(as.character(LatLonCounts$Var2))
```

```
ggmap(chicago) + geom_tile(data = LatLonCounts, aes(x = Long, y = Lat, alpha
= Freq), fill="red")+
```

```
ggtitle("Domestic Crimes")+labs(alpha="Count")+theme(plot.title = element_text(hjust=0.5))
```



18) Let's see where motor vehicle theft is common:



Domestic crimes show concentration over two areas whereas motor vehicle theft iswide spread over large part of the city of Chicago.

VIVA QUESTIONS:

1. Which technology is used to import and export data in Hadoop?
2. What is importing and exporting to and from Hadoop using Sqoop?
3. How do I export and import data from one database to another?
4. Which Hadoop ecosystem component is used to import and export data into RDBMS?
5. What is importing and exporting data?
6. Which technology in big data is used to import and export data from RDBMS to HDFS and vice versa?

MARK ALLOCATION

CONTINUOUS INTERNAL ASSESSMENT		
Preparation & Conduct of experiments	**(50)**	
Observation &Result	**(30)**	
Record	**(10)**	
Viva-voce	**(10)**	
Total	**(100)**	

RESULT:

SIGNATURE OF FACULTY

CONTENT BEYOND SYLLABUS

1. Implement an MR Program that processes a weather dataset
2. Develop and execute the partitions and Buckets partitioning program in Hive