Department of Computer Science

Big Data Analytics – Hadoop – Lab Manual

Rathinam College of Arts and Science (Autonomous)

Department of computer science

In-charge HOD Principal

Prepared by:	Approved & Reviewed by:	Issued by:	w.e.f Date:
D. Raj Balaji			

Rathinam College of Arts and Science (Autonomous)

Rathinam Tech zone, Pollachi Road, Eachanari,

Coimbatore – 641 021

DEPARTMENT OF COMPUTER SCIENCE

Lab Manual for the Academic Year 2017-18

SUBJECT	:	Big Data Analytics – Hadoop - Lab
STREAM	:	B.Sc CT / B.Sc IT

H.O.D

INDEX

LAB OBJECTIVE

Upon successful completion of this Lab the student will be able to:

- Installation and environment ready for Hadoop and HIVE
- Transfer data from Local System to HDFS
- Transfer data from HDFS to Local system
- Compile and run default .jar files from hdaoop
- Create own .jar files of JAVA programs
- Create Database and table in HIVE
- Load data into table in HIVE
- Manipulating the data in HIVE
- Retrieving the data from the HIVE (with joins and Data Aggregation with regular expressions)
- Performing database operations (like create, select, etc.,)

INTRODUCTION ABOUT LAB

There are 60 systems installed in this Lab. Their configurations are as follows:

Processor	:	intel i3 processor
RAM	:	2 GB
Hard Disk	:	500 GB
Mouse	:	Optical Mouse
Network Interface card	:	Present

Software

- All systems are configured in **DUAL BOOT** mode i.e., Students can boot from ubuntu (Linux) as per their lab requirement. Because the Hadoop shall be installed in the linux only.

- Each system will be connect with internet to connect to linux server to update the OS and Java version.

- Systems are provided for students in the 1:1 ratio.

- Systems are assigned numbers and same system is allotted for students when they do the lab.

STANDARD OPERATING PROCEDURE – SOP

a) Explanation on experiment by the concerned faculty covering the following aspects:

1) Name of the experiment/Aim
2) Software/Hardware required
3) Algorithm
4) Source Program
5) Test Data

c) Compiling and execution of the program

Writing of the experiment in the Observation Book:

The students will write the experiment in the Observation book as per the following format:

a) Name of the experiment
b) Aim of the experiment
c) Algorithm
d) Source Program
e) Test Data
f) Results for different data sets
g) Errors observed (if any) during compilation/execution
h) Signature of the Faculty

Guidelines to Students

- Equipment in the lab for the use of student community. Students need to maintain a proper decorum in the computer lab. Students must use the equipment with care. Any damage is caused is punishable.
- Students are required to carry their observation with completed exercises while entering the lab. Those who don’t have observation book, are not allowed to enter the lab
- Students are supposed to occupy the machines allotted to them and are not supposed to talk or make noise in the lab.
- Lab can be used in free time / lunch hours by the students who need to use the systems should take prior permission from the lab in-charge.
- Lab records need to be submitted on or before date of submission.
- Students are not supposed to use floppy disks / pen drives.

List of Lab Exercises

Syllabus Programs

S. No	Name of the program
1	Setup Enviornment ready for Hadoop (Hadoop Installation)
2	Create a text file with some content in the Desktop and load the text file in to the HDFS
3	Counting the words in a text file using map-reduce jar file.
4	Write a Java Program to count no of words in a text file in HDFS (Compile and Run)
5	Setup the environment ready for HIVE (Installation of HIVE)
6	Do the followings in HIVE: 1. Create Database 2. Describe and extended database describe 3. Alter Database 4. Select database 5. Create table 6. Drop database
7	Do the followings in HIVE: 1. Create Database 2. Create table 3. Describe table 4. Load data into table

8	Do the followings in HIVE: 1. Create table 2. Load data into table 3. Do the following Data Manipulation a. Select with where, Regular expression b. Group by, Sort by, having c. Limit
B 9	Do the followings in HIVE: 1. Count 2. Maximum & Maximum Distinct 3. Minimum & Minimum Distinct 4. Average & Average Distinct 5. Sum
10	Do the followings joins in HIVE: 1. Right outer join 2. Left outer join 3. Full outer join 4. Union All

Background Theory

Hadoop is an open source, Java-based programming framework that supports the processing and storage of extremely large data sets in a distributed computing environment. It is part of the Apache project sponsored by the Apache Software Foundation.

Hadoop makes it possible to run applications on systems with thousands of commodity hardware nodes, and to handle thousands of terrabytes of data. Its distributed file system facilitates rapid data transfer rates among nodes and allows the system to continue operating in case of a node failure. This approach lowers the risk of catastrophic system failure and unexpected data loss, even if a significant number of nodes become inoperative.

Apache Hive is a Data warehouse system which is built to work on Hadoop. It is used to querying and managing large datasets residing in distributed storage. The Hive query language (HiveQL) is the primary data processing method for Treasure Data. HiveQL is powered by Apache Hive. Treasure Data is a cloud data platform that allows users to collect, store, and analyze their data on the cloud.

Data Types in HIVE

- Column Types
- Literals
- Null Values
- Complex Types

Different types of commands in HiveQL:

1. Create Database

Syntax:

CREATE DATABASE|SCHEMA [IF NOT EXISTS] <database name>

2. Drop Database

Syntax:

DROP DATABASE StatementDROP (DATABASE|SCHEMA) [IF EXISTS] database_name
`[RESTRICT|CASCADE];`

3. Create Table

Syntax:

CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.] table_name

[(col_name data_type [COMMENT col_comment], ...)]

[COMMENT table_comment]

[ROW FORMAT row_format]

[STORED AS file_format]

4. Alter Table

Syntax:

ALTER TABLE name RENAME TO new_name

ALTER TABLE name ADD COLUMNS (col_spec[, col_spec ...])

ALTER TABLE name DROP [COLUMN] column_name

ALTER TABLE name CHANGE column_name new_name new_type

ALTER TABLE name REPLACE COLUMNS (col_spec[, col_spec ...])

5. Drop Table

Syntax:

DROP TABLE [IF EXISTS] table_name;

6. Select-Where

Syntax:

SELECT [ALL | DISTINCT] select_expr, select_expr, ...

FROM table_reference

[WHERE where_condition]

[GROUP BY col_list]

[HAVING having_condition]

[CLUSTER BY col_list | [DISTRIBUTE BY col_list] [SORT BY col_list]]

[LIMIT number];

7. Select-Order By

Syntax:

SELECT [ALL | DISTINCT] select_expr, select_expr, ...

FROM table_reference

```
[WHERE where_condition]
[GROUP BY col_list]
[HAVING having_condition]
[ORDER BY col_list]]
[LIMIT number];
```

8. Select-Group By

Syntax:

```
SELECT [ALL | DISTINCT] select_expr, select_expr, ...
FROM table_reference
[WHERE where_condition]
[GROUP BY col_list]
[HAVING having_condition]
[ORDER BY col_list]]
[LIMIT number];
```

9. Select-Joins

Syntax:

```
join_table:
  table_reference JOIN table_factor [join_condition]
  | table_reference {LEFT|RIGHT|FULL} [OUTER] JOIN table_reference
  join_condition
  | table_reference LEFT SEMI JOIN table_reference join_condition
  | table_reference CROSS JOIN table_reference [join_condition]
```

Solutions for programs

1. Setup Enviornment ready for Hadoop (Hadoop Installation)

Aim

To install hadoop in ubunto operating system and make ready the hadoop environment

Program & Comments to execute

Copy the hadoop installation file to the "Downloads" folder and execute the following commends.

- cd Downloads/
- sudo tar -xvzf hadoop-2.6.0.tar.gz
- sudo mkdir -p /usr/local/hadoop
- sudo chown -R myhduser:myhadoop /usr/local/hadoop/
- cd hadoop-2.6.0/
- ls
- sudo mv * /usr/local/hadoop
- ls
- sudo mv * /usr/local/hadoop
- sudo vi ~/.bashrc
- source ~/.bashrc
- hadoop version
- java -version
- sudo mkdir -p /app/myhadoop/tmp
- sudo chown -R myhduser:myhadoop /app/myhadoop/tmp/
- sudo vi /usr/local/hadoop/etc/hadoop/core-site.xml

- cd /usr/local/hadoop/etc/hadoop/
- sudo vi hadoop-env.sh
- sudo cp mapred-site.xml.template mapred-site.xml
- sudo vi mapred-site.xml
- sudo mkdir -p /usr/local/myhadoop_store/hdfs/namenode
- sudo mkdir -p /usr/local/myhadoop_store/hdfs/datanode
- sudo chown -R myhduser:myhadoop /usr/local/myhadoop_store/hdfs/namenode/
- sudo chown -R myhduser:myhadoop /usr/local/myhadoop_store/hdfs/datanode
- sudo vi hdfs-site.xml
- sudo vi yarn-site.xml
- hadoop namenode -format
- start-all.sh
- jps

Configuration File commands:

BASH Config File:

```
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
export HADOOP_INSTALL=/usr/local/hadoop-2.6.0
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
```

Core Site Configuration:

```
<property>
<name>hadoop.tmp.dir</name>
<value>/app/myhadoop/tmp</value>
</property>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
```

Mapred Configuration:

```
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
```

HDFS Configuration:

```
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:/usr/local/myhadoop_store/hdfs/namenode/</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>file:/usr/local/myhadoop_store/hdfs/datanode/</value>
</property>
```

Yarn Configuration:

```
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.nodemanager.aux-services.mapreduce_shuffle.calss</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
```

<u>Output:</u>

```
myhduser@user-Lenovo-B450:/home/user$ jps

3097 NodeManager

2226 NameNode

2692 SecondaryNameNode

3141 Jps

2872 ResourceManager

2437 DataNode

myhduser@user-Lenovo-B450:/home/user$
```

2. Create a text file with some content in the Desktop and load the text file in to the HDFS

Aim:

To Create a .txt file and put some content in that file and load that file into hadoop HDFS

Program & Comments to execute

Create folder called "inputdata" in Desktop and create a .txt file inside the inputdata folder using the following commands.

- cd Desktop
- mkdir inputdata
- cd inputdata
- echo "Rathinam college of arts and science Rathinam Institute of Management Rathinam Technical campus" >> test.txt
- cat test.txt
- cd /usr/local/hadoop/
- bin/hdfs dfs -mkdir /user
- bin/hdfs dfs -mkdir /user/myhduser
- bin/hdfs dfs -copyFromLocal /home/user/Desktop/inputdata/ test.txt
- bin/hadoop dfs -cat /user/myhduser/ test.txt

Output:

- myhduser@user-Lenovo-B450:/usr/local/hadoop$ bin/hdfs dfs -copyFromLocal /home/user/Desktop/data1/wordtest5.txt

- 16/09/19 11:23:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
- myhduser@user-Lenovo-B450:/usr/local/hadoop$

3. Counting the words in a text file using map-reduce jar file.

Aim:

To create a .txt file and put some content in that file and load that file into hadoop HDFS and count the words in that text file using word count jar file

Program & Comments to execute

Create folder called “inputdata” in Desktop and create a .txt file inside the inputdata folder using the following commands.

- cd Desktop
- mkdir inputdata
- cd inputdata
- echo “Rathinam college of arts and science Rathinam Institute of Management Rathinam Technical campus” >> test.txt
- cat test.txt

Transfer the .txt file into hadoop HDFS and count the words using the following commends.

- cd /usr/local/hadoop/
- bin/hdfs dfs -mkdir /user
- bin/hdfs dfs -mkdir /user/myhduser
- bin/hdfs dfs -copyFromLocal /home/myhduser/Desktop/data/test.txt
- bin/hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar wordcount /user/myhduser/test.txt /output
- bin/hadoop dfs -cat /output/part-r-00000
- bin/hadoop dfs -cat /output/*

Output:

```
myhduser@user-Lenovo-B450:/usr/local/hadoop$ bin/hadoop dfs -cat /outputWord/*

DEPRECATED: Use of this script to execute hdfs command is deprecated.

Instead use the hdfs command for it.

16/09/19 11:18:16 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Arts	1

College	3

Engineering	1

International	1

Management	1

Rathinam	4

School	1

myhduser@user-Lenovo-B450:/usr/local/hadoop$
```

4. Write a Java Program to count no of words in a text file in HDFS (Compile and Run)

Aim:

To write a java program to count no of words in text file in the hdfs and compile and run that java file to count the words.

Program & Comments to execute

Create folder called "inputdata" in Desktop and create a .txt file inside the inputdata folder using the following commands.

- cd Desktop
- mkdir inputdata
- cd inputdata
- echo "Rathinam college of arts and science Rathinam Institute of Management Rathinam Technical campus" >> test.txt
- cat test.txt

Transfer the .txt file into hadoop HDFS and count the words using the following commends.

- cd /usr/local/hadoop/
- bin/hdfs dfs -put '/home/user/Desktop/inputdata/ test.txt'
- /user/myhduser
- cd
- cd '/home/user/Desktop/wordcountf'

- javac -classpath /usr/local/hadoop/share/hadoop/common/hadoop-common-2.6.0.jar:/usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-core-2.6.0.jar:/usr/local/hadoop/share/hadoop/common/lib/commons-cli-1.2.jar -d /home/user/Desktop/wordcountf *.java
- jar -cvf wordcountj.jar -C /home/user/Desktop/wordcountf/wordcountc .
- cd /usr/local/hadoop/
- bin/hadoop jar /home/user/Desktop/wordcountf/wordcountj.jar WordCountpro /user/myhduser/ test.txt outputwprd
- /usr/local/hadoop/bin/hadoop fs -cat outputwprd/*

Java Program

```
import java.io.IOException;
import java.util.*;
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 WordCountpro {
 public static class Map extends 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, Context context) throws IOException,
InterruptedException {
        String line = value.toString();
        StringTokenizer tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()) {
            word.set(tokenizer.nextToken());
            context.write(word, one);
        }
    }
 }
public static class Reduce extends Reducer<Text, IntWritable, Text, IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values, Context context)
      throws IOException, InterruptedException {
        int sum = 0;
        while (values.hasNext()) {
            sum += values.next().get();
        }
        context.write(key, new IntWritable(sum));
    }
 }
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();

    Job job = new Job(conf, "wordcountjob");
    job.setJarByClass(WordCountpro.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    job.setMapperClass(Map.class);
    job.setReducerClass(Reduce.class);
    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    job.waitForCompletion(true);
 }
}
```

Output:

```
myhduser@user-Lenovo-B450:/usr/local/hadoop$ /usr/local/hadoop/bin/hadoop fs -
cat outputwprd/*

16/09/19 11:19:02 WARN util.NativeCodeLoader: Unable to load native-hadoop library
for your platform... using builtin-java classes where applicable

Arts        1

College     1

College     1

College     1

Engineering 1

International       1

Management          1
```

Rathinam	1
Rathinam	1
Rathinam	1
Rathinam	1
School	1

5. Setup the environment ready for HIVE (Installation of HIVE)

Aim

To install HIVE on the top of hadoop in ubunto operating system and make ready the HIVE environment

Program & Comments to execute

Copy the HIVE installations file to the “Downloads” folder and execute the following commends.

- cd /home/myhduser/Downloads
- sudo tar -xvzf apache-hive-1.0.1-bin.tar.gz
- sudo mv apache-hive-1.0.1-bin /usr/local
- sudo chown -R myhduser:myhadoop /usr/local/apache-hive-1.0.1-bin/
- sudo vi ~/.bashrc
- source vi ~/.bashrc
- cd $HIVE_HOME
- sudo vi bin/hive-config.sh
- start-all.sh
- jps
- hive

HIVE Configuration:

```
export HIVE_HOME="/usr/local/apache-hive-1.0.1-bin"
PATH=$PATH:$HIVE_HOME/bin
export PATH
```

<u>Output:</u>

```
myhduser@user-Lenovo-B450:~$ hive

Logging initialized using configuration in jar:file:/usr/local/apache-hive-1.0.1-
bin/lib/hive-common-1.0.1.jar!/hive-log4j.properties

SLF4J: Class path contains multiple SLF4J bindings.

SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-
log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: Found binding in [jar:file:/usr/local/apache-hive-1.0.1-bin/lib/hive-jdbc-1.0.1-
standalone.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.

SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]

hive>
```

6. Create and Manipulate Database and Create table

Aim

To create and manipulate database in HIVE and to do the following operations:

1. Create Database
2. Describe and extended database describe
3. Alter Database
4. Select database
5. Create table
6. Drop database

Program & Comments to execute

Execute the following commends to create, alter, select and drop database and to create table.

- CREATE DATABASE IF NOT EXISTS STUDENT COMMENT 'STUDENT DETAILS' WITH DBPROPERTIES ('creator' = 'DRB')
- show databases;
- describe database student;
- describe database extended student;
- alter database student set dbproperties ('edited-by' = 'Raj Balaji');
- describe database extended student;
- use student;
- create table stud3 (Reg_no INT, S_Name STRING, Class STRING, Dept STRING, Age INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

Output:

```
hive> Describe database student;

OK

student         STUDENT DETAILS hdfs://localhost:9000/user/hive/warehouse/student.db
        myhduser        USER

Time taken: 0.053 seconds, Fetched: 1 row(s)

hive> describe database extended student;

OK

student         STUDENT DETAILS hdfs://localhost:9000/user/hive/warehouse/student.db
        myhduser        USER    {edited-by=Raj Balaji, creator=DRB}

Time taken: 0.028 seconds, Fetched: 1 row(s)

hive> create table stud3 (Reg_no INT, S_Name STRING, Class STRING, Dept STRING,
Age INT) ROW FORMAT DELIMITED FIELDS TERMINATIMITED FIELDS TERMINATED BY
',';

OK

Time taken: 0.725 seconds
```

7. Create and Manipulate Table and Load data into table

Aim

To create and manipulate table in HIVE and to do the following operations:

1. Create Database
2. Create table
3. Describe table
4. Load data into table

Program & Comments to execute

Execute the following commends to create database, table and to load data into that table.

- CREATE DATABASE IF NOT EXISTS STUDENT COMMENT 'STUDENT DETAILS' WITH DBPROPERTIES ('creator' = 'DRB')
- show databases;
- describe database extended student;
- describe database extended student;
- use student;
- create table stud3 (Reg_no INT, S_Name STRING, Class STRING, Dept STRING, Age INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
- LOAD DATA LOCAL INPATH '/home/user/Desktop/StudDet.csv' OVERWRITE INTO TABLE STUD3;
- select * from stud3;
- describe stud3;

Output:

```
hive> create table stud3 (Reg_no INT, S_Name STRING, Class STRING, Dept STRING,
Age INT) ROW FORMAT DELIMITED FIELDS TERMINATIMITED FIELDS TERMINATED BY
',';

OK

Time taken: 0.725 seconds

hive> LOAD DATA LOCAL INPATH '/home/user/Desktop/StudDet.csv' OVERWRITE
INTO TABLE STUD3;

Loading data to table default.stud3

Table default.stud3 stats: [numFiles=1, numRows=0, totalSize=295, rawDataSize=0]

OK

Time taken: 0.952 seconds

hive> select * from stud3;

OK

NULL 15MCS013        Bhimarao     NULL

NULL 15MCS014        Arumugam  NULL

NULL 15MCS015        Raju   NULL

NULL 15MCS016        Karishma     NULL

NULL 15MCS017        Dhanalakshmi       NULL

Time taken: 0.639 seconds, Fetched: 5 row(s)
```

8. Data manipulation in a table

Aim

To create table and load data into that table and to do the following data manipulations:

1. Select with where, Regular expression
2. Group by, Sort by, having
3. Limit

Program & Comments to execute

Execute the following commends to create table and to load data into that table and data manipulation in that table.

- create table subject1 (subcode string, mxse int, mxca int, mnse int, mnca int, crpoint int, spart int, subtype string, subamt int) row format delimited fields terminated by ',';
- load data local inpath '/home/user/Desktop/Subject.csv' overwrite into table subject1;
- select * from subject1;
- select * from subject where subcode = '13BCA13A';
- select * from subject1 where subtype = 'T' and subamt >'350';
- select subpart, subamt where subcode like '%63%';
- select subcode, subtype, subamt from subject1 where spart = 3 group by subtype, subamt;
- selct * from subject1 subamt limit 10;
- select * from subject1 where subamt>300 gruop by spart;
- select * from subject1 group by spart having subtype = 'P';

Output:

```
hive> select * from subject1 where subcode = '13BCA13A';

OK

13BCA13A  75   25   30   0   4   3   T   80

Time taken: 1.589 seconds, Fetched: 1 row(s)

hive> select * from subject1 where subtype = 'P' and subamt >'350';

OK

13MIT23V  80   20   40   0   4   3   P   400

13MCS23V  80   20   40   0   4   3   P   400

13MJM23V  80   20   40   0   4   3   P   400

13MBA33V  80   20   40   0   4   3   P   500

13MJM43V  100  0    50   0   8   3   P   400

13MCS43V  250  0    125  0   10  3   P   400

13MIT43V  250  0    125  0   10  3   P   400

13MCM43V  160  40   80   0   10  3   P   400

13MBA43V  160  40   80   0   8   3   P   500

14MBA33V  80   20   40   0   4   3   P   500

15MCS33V  25   25   13   0   2   3   P   400

15MIT33V  25   25   13   0   2   3   P   400

14MBA43V  160  40   80   0   8   3   P   500

14MMC43V  60   40   30   0   5   3   P   400
```

```
14MCM43V  160     40      80      0       10      3       P       400

15MJC23V  80      20      40      0       4       3       P       400

Time taken: 0.097 seconds, Fetched: 16 row(s)

hive> select spart, subamt from subject1 where subcode like '%63%A';

OK

3         80

3         80

3         80

3         80

3         80

3         80

3         80

3         80

3         80

3         80

Time taken: 0.074 seconds, Fetched: 10 row(s)

hive> create table stud3 (Reg_no INT, S_Name STRING, Class STRING, Dept STRING, Age INT)
ROW FORMAT DELIMITED FIELDS TERMINAT
select * from subject1 subamt limit 10;

OK

sbjt_cod        NULL  NULL  NULL  NULL  NULL  NULL  sbjt_type       NULL

13BGE11M  75      25      30      0       4       1       T       80

13BGE11H  75      25      30      0       4       1       T       80
```

```
13BGE11F    75     25     30     0     4      1     T     80

13BGE11T    75     25     30     0     4      1     T     80

13BGE12E    75     25     30     0     4      2     T     80

13BCM13A    75     25     30     0     4      3     T     80

13BCM13B    75     25     30     0     4      3     T     80

13BCM1AA    75     25     30     0     4      3     T     80

13BGE1FA    50     0      20     0     2      4     T     80

Time taken: 0.069 seconds, Fetched: 10 row(s)

hive> select * from subject1 where subamt > 300 gruop by spart;

13MIT43V    250    0      125    0     10     3     P     400

13MCS43V    250    0      125    0     10     3     P     400

13MJM43V    100    0      50     0     8      3     P     400

13MJM23V    80     20     40     0     4      3     P     400

13MCS23V    80     20     40     0     4      3     P     400

15MIT33V    25     25     13     0     2      3     P     400

15MCS33V    25     25     13     0     2      3     P     400

14MBA33V    80     20     40     0     4      3     P     500

13MBA43V    160    40     80     0     8      3     P     500

14MBA43V    160    40     80     0     8      3     P     500

13MBA33V    80     20     40     0     4      3     P     500

Time taken: 36.535 seconds, Fetched: 163 row(s)
```

9. Data Aggregation in a table

Aim

To create table and load data into that table and to do the following data aggregation operations:

1. Count
2. Maximum & Maximum Distinct
3. Minimum & Minimum Distinct
4. Average & Average Distinct
5. Sum

Program & Comments to execute

Execute the following commends to create table and to load data into that table and to do data aggregation operation in that table.

```
create table subject1 (subcode string, mxse int, mxca int, mnse int, mnca int, crpoint int, spart int, subtype string, subamt int) row format delimited fields terminated by ',';
load data local inpath '/home/user/Desktop/Subject.csv' overwrite into table subject1;
select * from subject1;
select subcode, max(subamt) from subject1 group by subcode;
select spart, max(subamt) from subject1 group by spart;
select spart, min(subamt) from subject1 group by spart;
select spart, avg(subamt) from subject1 group by spart;
select spart, sum(subamt) from subject1 group by spart;
```

Output:

```
hive> select subcode, max(subamt) from subject1 group by subcode;
16MBA13F    350

16MBA13G    350

16MBA13P    350

16MMA13A    150

16MMA13B    150

16MMA13C    150

16MMA13D    150

16MMA1EC    150

sbjt_cod    NULL

Time taken: 28.929 seconds, Fetched: 1078 row(s)

hive> select spart, max(subamt) from subject1 group by spart;

MapReduce Jobs Launched:

Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.9 sec   HDFS Read: 31922 HDFS Write: 39 SUCCESS

Total MapReduce CPU Time Spent: 3 seconds 900 msec

OK

NULL  NULL

1     80

2     80
```

```
3       500

4       150

5       80

6       150

Time taken: 26.922 seconds, Fetched: 7 row(s)

hive> select spart, min(subamt) from subject1 group by spart;

MapReduce Jobs Launched:

Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.85 sec   HDFS Read: 31922 HDFS Write:
36 SUCCESS

Total MapReduce CPU Time Spent: 3 seconds 850 msec

OK

NULL  NULL

1       80

2       80

3       80

4       80

5       80

6       80
Time taken: 27.356 seconds, Fetched: 7 row(s)

hive> select spart, avg(subamt) from subject1 group by spart;

MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.89 sec   HDFS Read: 31922 HDFS Write:
88 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 890 msec
OK
```

```
NULL  NULL

1     80.0

2     80.0

3     150.09237875288684

4     81.70731707317073

5     80.0

6     80.75268817204301

Time taken: 26.31 seconds, Fetched: 7 row(s)

hive> select spart, sum(subamt) from subject1 group by spart;

MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.94 sec   HDFS Read: 31922 HDFS Write:
47 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 940 msec

OK

NULL  NULL

1     2240

2     560

3     129980

4     6700

5     80

6     7510

Time taken: 28.717 seconds, Fetched: 7 row(s)
```

10. Join operations

Aim

To do the following join operations in a data table in HIVE:

1. Right outer join
2. Left outer join
3. Full outer join
4. Union All

Program & Comments to execute

Execute the following commends to create table and to load data into that table and to do join operations in that table.

- create table subject1 (subcode string, mxse int, mxca int, mnse int, mnca int, crpoint int, spart int, subtype string, subamt int) row format delimited fields terminated by ',';
- load data local inpath '/home/user/Desktop/Subject.csv' overwrite into table subject1;
- select * from subject1;
- create table CrseMap (subcode string, subTitle string) row format delimited fields terminated by ',';
- load data local inpath '/home/user/Desktop/Subject.csv' overwrite into table CrseMap.csv;
- select * from CrseMap;
- select a.SubCode, b.subTitle, a.mxse, a.mnse, a.mxca, a.mnca from subject1 a Right Outer Join CorsMap b On (a.SubCode = b.SubCode);

- select a.SubCode, b.subTitle, a.mxse, a.mnse, a.mxca, a.mnca from subject1 a Left Outer Join CorsMap b On (a.SubCode = b.SubCode);
- select a.SubCode, b.subTitle, a.mxse, a.mnse, a.mxca, a.mnca from subject1 a Full Outer Join CorsMap b On (a.SubCode = b.SubCode);
- select * from subject1 union all select * from CrseMap;