Experiment-1: Implement a simple map-reduce job that builds an inverted index on the set of input documents (Hadoop)

A) STANDALONE MODE:

- *Installation of Java*
  Command: *sudo apt update*
  Command: *sudo apt install default-jdk*

- *Download and extract Hadoop*
  Download Hadoop 2.7.0 from following link
  https://archive.apache.org/dist/hadoop/common/hadoop-2.7.0/hadoop-2.7.0.tar.gz
  Extract downloaded zip file
  Right click on file and click Extract Here
  Move Hadoop folder into /usr/lib
  Command: *sudo mv hadoop-2.7.0 /usr/lib/hadoop*

- *Set the path for java and hadoop*
  Command: *sudo gedit $HOME/.bashrc*

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export PATH=$PATH:$JAVA_HOME/bin

export HADOOP_COMMON_HOME=/usr/lib/hadoop
export HADOOP_MAPRED_HOME=/usr/lib/hadoop
export PATH=$PATH: /usr/lib/hadoop/bin
export PATH=$PATH: /usr/lib/hadoop/Sbin
```

- *Checking of java and hadoop*
  Command: *java --version*
  Command: *hadoop version*

B) PSEUDO Distributed MODE:

Hadoop single node cluster runs on single machine. The namenodes and datanodes are performing on the one machine. The installation and configuration steps as given below:

- *Installation of secured shell*
  Command: *sudo apt-get install openssh-server*

- *Create a ssh key for passwordless ssh configuration*
  Command: *ssh-keygen -t rsa –P ""*

- *Moving the key to authorized key*
  Command: *cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys*

  */**********RESTART THE COMPUTER**********/*

- *Checking of secured shell login*
  Command: *ssh localhost*

- *Add JAVA_HOME directory in hadoop-env.sh file*
  Command: *sudo gedit /usr/lib/hadoop/conf/hadoop-env.sh*

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
```

- *Creating namenode and datanode directories for hadoop*

Command: *sudo mkdir -p /usr/lib/hadoop/dfs/namenode*
Command: *sudo mkdir -p /usr/lib/hadoop/dfs/datanode*

- *Change permissions for namenode and datanode directories for hadoop*
  Command: *sudo chown -R sriindu:sriindu /usr/lib/hadoop/dfs/namenode*
  Command: *sudo chown -R sriindu:sriindu /usr/lib/hadoop/dfs/datanode*

- *Configure core-site.xml*
  Command: *sudo gedit /usr/lib/hadoop/etc/hadoop/core-site.xml*

```
<property>
        <name>fs.default.name</name>
        <value>hdfs://localhost:8020</value>
</property>
```

- *Configure hdfs-site.xml*
  Command: *sudo gedit /usr/lib/hadoop/etc/hadoop/hdfs-site.xml*

```
<property>
        <name>dfs.replication</name>
        <value>1</value>
</property>

<property>
        <name>dfs.permissions</name>
        <value>false</value>
</property>

<property>
        <name>dfs.namenode.name.dir</name>
        <value>/usr/lib/hadoop/dfs/namenode</value>
</property>

<property>
        <name>dfs.datanode.data.dir</name>
        <value>/usr/lib/hadoop/dfs/datanode</value>
</property>
```

- *Configure mapred-site.xml*
  Command: *sudo gedit /usr/lib/hadoop/etc/hadoop /mapred-site.xml*

```
<property>
        <name>mapred.job.tracker</name>
        <value>localhost:8021</value>
</property>
```

- *Format the name node*
  Command: *hadoop namenode -format*

- *Start the namenode, datanode*
  Command: *start-dfs.sh*

- *Start the Node Manager and Resource Manager*
  Command: *start-mapred.sh*

- *To check if Hadoop started correctly*

  Command: *jps*

  NameNode
  SecondaryNamenode
  DataNode
  NodeManager
  ResourceManager
  jps

C) Uploading Data into HDFS:

- Open text editor and write the following text on that editor, save the file and close

```
hai welcome to data analytics lab hai welcome
```

- Create the input directory and copy temporary content file in that input directory

Note: Now that input folder is having "file.txt"

- Put the file.txt into hdfs

  Command: `hadoop fs -mkdir /input`

  Command: `hadoop fs -put /input/file.txt  /input/`

D) Running a basic Word Count Map Reduce Java program to understand Map Reduce Paradigm:

- **WordCount MapReduce Program**

```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
public class WordCount {
 public static class TokenizerMapper  extends Mapper<Object, Text, Text, IntWritable>
{
   private final static IntWritable one = new IntWritable(1);
   private Text word = new Text();
   public void map(Object key, Text value, Context context)
           throws IOException, InterruptedException {
     StringTokenizer itr = new StringTokenizer(value.toString());
     while (itr.hasMoreTokens()) {
       word.set(itr.nextToken());
       context.write(word, one);
     }
   }
 }
public static class IntSumReducer extends Reducer<Text,IntWritable,Text,IntWritable>
```

```
{
   private IntWritable result = new IntWritable();
   public void reduce(Text key, Iterable<IntWritable> values, Context context
                   ) throws IOException, InterruptedException {
                int sum = 0;
                for (IntWritable val : values) {
                        sum += val.get();
                }
                result.set(sum);
                context.write(key, result);
              }
            }

  public static void main(String[] args) throws Exception {
              Configuration conf = new Configuration();
              Job job = Job.getInstance(conf, "word count");
              job.setJarByClass(WordCount.class);
              job.setMapperClass(TokenizerMapper.class);
              job.setCombinerClass(IntSumReducer.class);
              job.setReducerClass(IntSumReducer.class);
              job.setOutputKeyClass(Text.class);
              job.setOutputValueClass(IntWritable.class);
              FileInputFormat.addInputPath(job, new Path(args[0]));
              FileOutputFormat.setOutputPath(job, new Path(args[1]));
              System.exit(job.waitForCompletion(true) ? 0 : 1);
            }
}
```

Compile and Run wordCount Program

- Create jar file WordCount Program

  Command: `hadoop com.sun.tools.javac.Main WordCount.java`
  Command: `jar cf wc.jar WordCount*.class`
- Run WordCount jar file on input directory

  Command: `hadoop jar wc.jar WordCount input output`

- To see the output

  Command: `cat output/*`

analytics	1
data	1
hai	2
lab	1
to	1
welcome	2

Experiment-2: Process big data in HBase.

Installation of HBase:

- Download HBase from following Link:
  https://archive.apache.org/dist/hbase/2.5.8/hbase-2.5.8-bin.tar.gz

Name	Last modified	Size	Description
Parent Directory		-	
CHANGES.md	2024-03-01 00:23	1.1M	
RELEASENOTES.md	2024-03-01 00:23	600K	
api_compare_2.5.7_to_2.5.8RC0.html	2024-03-01 00:23	13K	
hbase-2.5.8-bin.tar.gz	2024-03-01 00:23	308M	
hbase-2.5.8-bin.tar.gz.asc	2024-03-01 00:23	833	
hbase-2.5.8-bin.tar.gz.sha512	2024-03-01 00:23	216	
hbase-2.5.8-client-bin.tar.gz	2024-03-01 00:23	296M	
hbase-2.5.8-client-bin.tar.gz.asc	2024-03-01 00:23	833	
hbase-2.5.8-client-bin.tar.gz.sha512	2024-03-01 00:23	268	
hbase-2.5.8-hadoop3-bin.tar.gz	2024-03-01 00:23	321M	
hbase-2.5.8-hadoop3-bin.tar.gz.asc	2024-03-01 00:23	833	
hbase-2.5.8-hadoop3-bin.tar.gz.sha512	2024-03-01 00:23	272	
hbase-2.5.8-hadoop3-client-bin.tar.gz	2024-03-01 00:23	309M	
hbase-2.5.8-hadoop3-client-bin.tar.gz.asc	2024-03-01 00:23	833	
hbase-2.5.8-hadoop3-client-bin.tar.gz.sha512	2024-03-01 00:23	300	
hbase-2.5.8-src.tar.gz	2024-03-01 00:23	36M	
hbase-2.5.8-src.tar.gz.asc	2024-03-01 00:23	833	
hbase-2.5.8-src.tar.gz.sha512	2024-03-01 00:23	216	

- Extract this file using right-click -> 'Extract here'.
- Now, move this folder to /usr/lib using following
- command:

  $ sudo mv Downloads/hbase-2.5.5 /usr/lib/hbase
- Open the bashrc file:

  $ sudo gedit ~/.bashrc
- Go to end of the file and add following lines.

  export HBASE_HOME=/usr/lib/hbase

  export PATH=$PATH:$HBASE_HOME/bin
- Close and open terminal again

```
mothi@Mothi:~$ hbase version
SLF4J: Class path contains multiple S
SLF4J: Found binding in [jar:file:/us
SLF4J: Found binding in [jar:file:/us
SLF4J: See http://www.slf4j.org/codes
SLF4J: Actual binding is of type [org
HBase 2.5.8
Source code repository git://buildbox
Compiled by apurtell on Thu Feb 29 15
From source with checksum 6610447458e
mothi@Mothi:~$
```

- **Create Directories and change permissions for "zookeeper" and "data"**

  $ sudo mkdir -p /usr/lib/hbase/zookeeper

  $ sudo mkdir -p /usr/lib/hbase/data

  $ sudo chown -R sriindu:sriindu /usr/lib/hbase/zookeeper

  $ sudo chown -R sriindu:sriindu /usr/lib/hbase/data

- **Edit hbase-env.sh file:**

  Command: *sudo gedit /usr/lib/hbase/conf/hbase-env.sh*

  export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

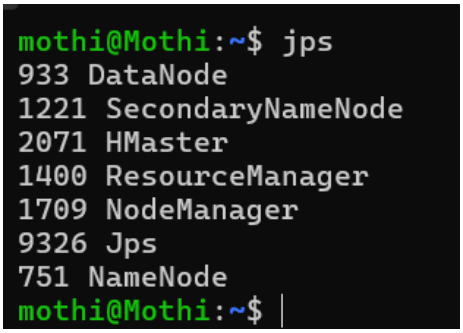
- **Edit hbase-site.xml file:**

Command: *sudo gedit /usr/lib/hbase/conf/hbase-site.xml*

```
<property>
    <name>hbase.unsafe.stream.capability.enforce</name>
    <value>false</value>
  </property>

  <property>
    <name>hbase.rootdir</name>
    <value>file:///usr/lib/hbase/data</value>
  </property>

  <property>
    <name>hbase.cluster.distributed</name>
    <value>false</value>
  </property>

  <property>
    <name>hbase.tmp.dir</name>
    <value>/usr/lib/hbase/tmp</value>
  </property>

  <property>
    <name>hbase.zookeeper.property.dataDir</name>
    <value>/usr/lib/hbase/zookeeper</value>
  </property>
```

- Start Hadoop and Hbase

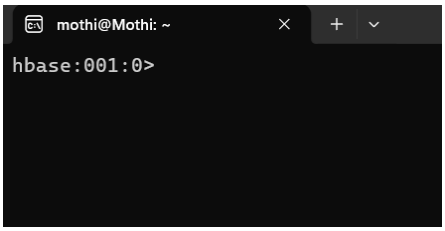
```
$ start-dfs.sh
$ start-mapred.sh
$ start-hbase.sh
$ jps
```



Store and retrieve data in HBase:

- Open Hbase using following command:

Command: *hbase shell*



- Create student table

Query: *create 'students','info';*

- Insert data into student table

Query: *put 'students', '1', 'info:roll', '101';*
Query: *put 'students', '1', 'info:name', 'mothi';*
Query: *put 'students', '1', 'info:department', 'aids';*

Query: *put 'students', '1', 'info:marks', '86';*

Query: *put 'students', '2', 'info:roll', '102';*
Query: *put 'students', '2', 'info:name', 'ravi';*
Query: *put 'students', '2', 'info:department', 'aids';*
Query: *put 'students', '2', 'info:marks', '92';*

- Retrieve data from student table
  Query: *scan 'students';*

```
hbase:021:0> scan 'students';
ROW                                   COLUMN+CELL
 1                                    column=info:department, timestamp=2026-05-09T05:05:21.168, value=aids
 1                                    column=info:marks, timestamp=2026-05-09T05:05:32.079, value=86
 1                                    column=info:name, timestamp=2026-05-09T05:05:24.524, value=mothi
 1                                    column=info:roll, timestamp=2026-05-09T05:01:58.789, value=101
 2                                    column=info:department, timestamp=2026-05-09T05:06:11.273, value=aids
 2                                    column=info:marks, timestamp=2026-05-09T05:06:22.375, value=92
 2                                    column=info:name, timestamp=2026-05-09T05:05:56.540, value=ravi
 2                                    column=info:roll, timestamp=2026-05-09T05:05:43.025, value=102
2 row(s)
Took 0.0430 seconds
hbase:022:0>
```

- Update data in student table
  Query: *put 'students', '2', 'info:marks', '95';*

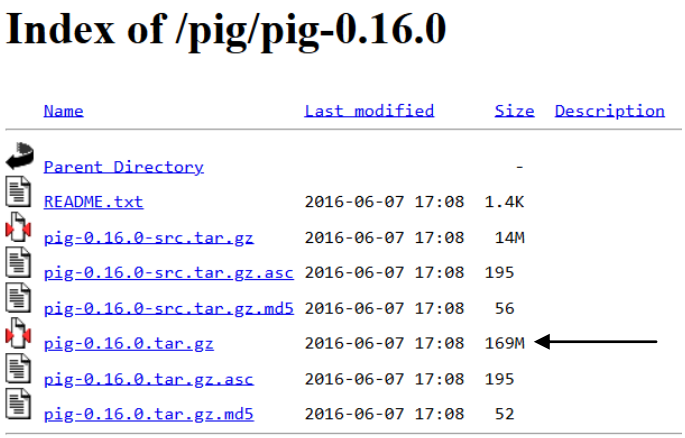
```
hbase:029:0> scan 'students';
ROW                                   COLUMN+CELL
 1                                    column=info:department, timestamp=2026-05-09T05:05:21.168, value=aids
 1                                    column=info:marks, timestamp=2026-05-09T05:05:32.079, value=86
 1                                    column=info:name, timestamp=2026-05-09T05:05:24.524, value=mothi
 1                                    column=info:roll, timestamp=2026-05-09T05:01:58.789, value=101
 2                                    column=info:department, timestamp=2026-05-09T05:06:11.273, value=aids
 2                                    column=info:marks, timestamp=2026-05-09T07:06:18.903, value=95
 2                                    column=info:name, timestamp=2026-05-09T05:05:56.540, value=ravi
 2                                    column=info:roll, timestamp=2026-05-09T05:05:43.025, value=102
2 row(s)
Took 0.0368 seconds
hbase:030:0>
```

- Drop student table
  Query: *disable 'student'*
  Query: *drop 'student'*

Experiment-3: Store and retrieve data in Pig

Installation of Pig

- Download the tar.gz file of Apache Pig from here:
  https://downloads.apache.org/pig/pig-0.16.0/pig-0.16.0.tar.gz



- Extract this file using right-click -> 'Extract here'.
- Now, move this folder to /usr/lib using following
- command:

  $ sudo mv Downloads/pig-0.16.0 /usr/lib/pig

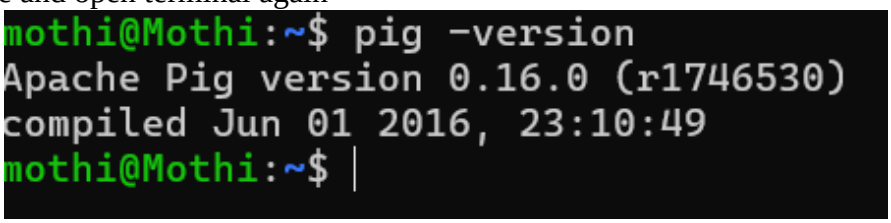
- Open the bashrc file:

  $ sudo gedit ~/.bashrc

- Go to end of the file and add following lines.

  export PIG_HOME=/usr/lib/pig
  export PATH=$PATH:$PIG_HOME/bin

- Close and open terminal again



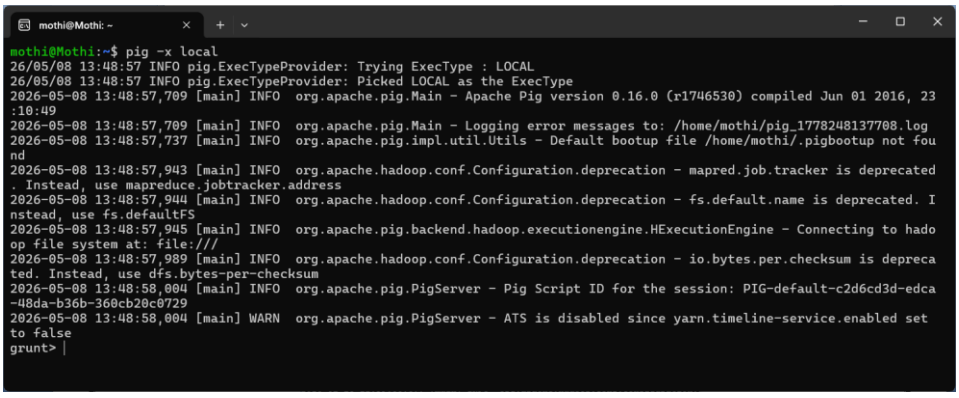
Start the Pig

- Start the pig in local mode:

  pig -x local

- Start the pig in mapreduce mode (needs hadoop datanode started):

  pig -x mapreduce



Sample Student dataset

ROLL	NAME	DEPARTMENT	MARKS
1	Ravi Rao	CIVIL	52
2	Raj Naidu	IT	74
3	Amit Verma	CSE	72
4	Raj Rao	IT	67
5	Pooja Patel	MECH	69
6	John Verma	EEE	62
7	Kiran Sharma	EEE	53
8	Geeta Naidu	AIML	54
9	Kiran Naidu	AIDS	57
10	Ravi Kumar	MECH	61

Programs:

1. **a) Load the student data using Pig.**
   b) Display all records.
   c) Store filtered results into output directory.

$gedit 1.pig

```
students = LOAD 'studentmarks.csv' USING PigStorage(',') as
(roll:int, name:chararray, department:chararray, marks:int);
dump students;
store students into 'output1';
```

$pig -x local 1.pig

```
2026-05-08 14:20:51,879 [main] INFO  org
(1,Ravi Rao,CIVIL,52)
(2,Raj Naidu,IT,74)
(3,Amit Verma,CSE,72)
(4,Raj Rao,IT,67)
(5,Pooja Patel,MECH,69)
(6,John Verma,EEE,62)
(7,Kiran Sharma,EEE,53)
(8,Geeta Naidu,AIML,54)
(9,Kiran Naidu,AIDS,57)
(10,Ravi Kumar,MECH,61)
2026-05-08 14:20:51,914 [main] INFO  org
```

```
mothi@Mothi:~/pigscripts$ cat output1/*
1       Ravi Rao        CIVIL   52
2       Raj Naidu       IT      94
3       Amit Verma      CSE     72
4       Raja Rao        IT      81
5       Pooja Patel     MECH    69
6       John Verma      EEE     82
7       Kiran Sharma    EEE     53
8       Geeta Naidu     AIML    84
9       Kiran Naidu     AIDS    95
10      Ravi Kumar      MECH    61
mothi@Mothi:~/pigscripts$
```

2. **a) Load the student data using Pig.**
   b) Filter students with marks > 80.
   c) Store filtered results into output directory.

$gedit 2.pig

```
students = LOAD 'studentmarks.csv' USING PigStorage(',') as
(roll:int, name:chararray, department:chararray, marks:int);
marks_greater_than_80 = FILTER students by marks>80;
store marks_greater_than_80 into 'output2';
```

$pig -x local 2.pig

```
mothi@Mothi:~/pigscripts$ cat output2/*
2       Raj Naidu       IT      94
4       Raja Rao        IT      81
6       John Verma      EEE     82
8       Geeta Naidu     AIML    84
9       Kiran Naidu     AIDS    95
mothi@Mothi:~/pigscripts$
```

3. a) Load the student data using Pig.
b) Find students from AIDS department.
c) Store filtered results into output directory.

$gedit 3.pig

```
students = LOAD 'studentmarks.csv' USING PigStorage(',') as
(roll:int, name:chararray, department:chararray, marks:int);
students_aids = FILTER students by department matches 'aids';
store students_aids into 'output3';
```

$pig -x local 3.pig

```
mothi@Mothi:~/pigscripts$ cat output3/*
9       Kiran Naidu     AIDS    95
mothi@Mothi:~/pigscripts$
```

4. a) Load the student data using Pig.
b) Count total number of students.
c) Store filtered results into output directory.

$gedit 4.pig

```
students = LOAD 'studentmarks.csv' USING PigStorage(',') as
(roll:int, name:chararray, department:chararray, marks:int);
grouped = GROUP students ALL;
students_count = FOREACH grouped GENERATE COUNT(students);
store students_count into 'output4';
```

$pig -x local 4.pig

```
mothi@Mothi:~/pigscripts$ cat output4/*
10
mothi@Mothi:~/pigscripts$
```

5. a) Load the student data using Pig.
b) Find average marks.
c) Store filtered results into output directory.

$gedit 5.pig

```
students = LOAD 'studentmarks.csv' USING PigStorage(',') as
(roll:int, name:chararray, department:chararray, marks:int);
grouped = GROUP students ALL;
marks_avg = FOREACH grouped GENERATE AVG(students.marks);
store marks_avg into 'output5';
```

$pig -x local 5.pig

```
mothi@Mothi:~/pigscripts$ cat output5/*
74.3
mothi@Mothi:~/pigscripts$
```

6. a) Load the student data using Pig.
b) Sort students by marks in descending order.
c) Store filtered results into output directory.

$gedit 6.pig

```
students = LOAD 'studentmarks.csv' USING PigStorage(',') as
(roll:int, name:chararray, department:chararray, marks:int);
marks_ord = ORDER students BY marks DESC;
store marks_ord into 'output6';
```

$pig -x local 6.pig

```
mothi@Mothi:~/pigscripts$ cat output6/*
9       Kiran Naidu     AIDS    95
2       Raj Naidu       IT      94
8       Geeta Naidu     AIML    84
6       John Verma      EEE     82
4       Raja Rao        IT      81
3       Amit Verma      CSE     72
5       Pooja Patel     MECH    69
10      Ravi Kumar      MECH    61
7       Kiran Sharma    EEE     53
1       Ravi Rao        CIVIL   52
mothi@Mothi:~/pigscripts$
```

7. a) Load the student data using Pig.
b) Group students by department.
c) Store filtered results into output directory.

$gedit 7.pig

```
students = LOAD 'studentmarks.csv' USING PigStorage(',') as
(roll:int, name:chararray, department:chararray, marks:int);
dept_grp = GROUP students BY department;
store dept_grp into 'output7';
```

$pig -x local 7.pig

```
mothi@Mothi:~/pigscripts$ cat output7/*
IT      {(4,Raja Rao,IT,81),(2,Raj Naidu,IT,94)}
CSE     {(3,Amit Verma,CSE,72)}
EEE     {(7,Kiran Sharma,EEE,53),(6,John Verma,EEE,82)}
AIDS    {(9,Kiran Naidu,AIDS,95)}
AIML    {(8,Geeta Naidu,AIML,84)}
MECH    {(10,Ravi Kumar,MECH,61),(5,Pooja Patel,MECH,69)}
CIVIL   {(1,Ravi Rao,CIVIL,52)}
mothi@Mothi:~/pigscripts$
```

8. a) Load the student data using Pig.
b) Find maximum marks in each department.
c) Store filtered results into output directory.

$gedit 8.pig

```
students = LOAD 'studentmarks.csv' USING PigStorage(',') as
(roll:int, name:chararray, department:chararray, marks:int);
dept_grp = GROUP students BY department;
max_marks_per_dept = FOREACH dept_grp GENERATE group AS department,
MAX(students.marks) AS max_marks;
store max_marks_per_dept into 'output8';
```

$pig -x local 8.pig

```
mothi@Mothi:~/pigscripts$ cat output8/*
IT      94
CSE     72
EEE     82
AIDS    95
AIML    84
MECH    69
CIVIL   52
mothi@Mothi:~/pigscripts$
```

Experiment-4: Perform Social media analysis using cassandra

Installing the docker image

Pull the docker image. For the latest image, use:

```
docker pull cassandra:latest
```

This `docker pull` command will get the latest version of the 'Docker Official' Apache Cassandra image available from the Dockerhub.

Start Cassandra with a `docker run` command:

```
docker run --name cass_cluster cassandra:latest
```

The `--name` option will be the name of the Cassandra cluster created.

Start the CQL shell, `cqlsh` to interact with the Cassandra node created:

```
docker exec -it cass_cluster cqlsh
```

Step-1: Create Database

```
cqlsh> CREATE KEYSPACE socialmedia
           ... WITH replication = {
           ...     'class':'SimpleStrategy',
           ...     'replication_factor':1
           ... };
cqlsh> use socialmedia;
```

Step-2: Create Tables

```
cqlsh:socialmedia> CREATE TABLE users (
                   ...     userid uuid PRIMARY KEY,
                   ...     username text,
                   ...     city text,
                   ...     followers int,
                   ...     joining_date timestamp
                   ... );

cqlsh:socialmedia> CREATE TABLE posts (
                   ...     postid uuid PRIMARY KEY,
                   ...     userid uuid,
                   ...     content text,
                   ...     hashtags list<text>,
                   ...     likes int,
                   ...     comments int,
                   ...     post_time timestamp
                   ... );
cqlsh:socialmedia> describe tables;
      posts  users
```

Step-3: Insert Data into Tables

```
cqlsh:socialmedia> INSERT INTO users (
                ...     userid,
                ...     username,
                ...     city,
                ...     followers,
                ...     joining_date
                ... )
                ... VALUES (
                ...     uuid(),
                ...     'mothi',
                ...     'Hyderabad',
                ...     2500,
                ...     toTimestamp(now())
                ... );

cqlsh:socialmedia> INSERT INTO users (
                ...     userid,
                ...     username,
                ...     city,
                ...     followers,
                ...     joining_date
                ... )
                ... VALUES (
                ...     uuid(),
                ...     'ravi',
                ...     'Delhi',
                ...     1200,
                ...     toTimestamp(now())
                ... );
cqlsh:socialmedia> select * from users;
```

userid	city	followers	joining_date	username
e10500a7-abe9-43d7-a32f-01ee317e6c06	Hyderabad	2500	2026-05-12 04:54:31.891000+0000	mothi
6f944f66-c467-4044-96f7-1e75ced0a487	Delhi	1200	2026-05-12 04:54:33.623000+0000	ravi

(2 rows)

```
cqlsh:socialmedia> INSERT INTO posts (
                ...     postid,
                ...     userid,
                ...     content,
                ...     hashtags,
                ...     likes,
                ...     comments,
                ...     post_time
                ... )
                ... VALUES (
                ...     uuid(),
                ...     uuid(),
                ...     'Learning Cassandra Big Data',
                ...     ['#cassandra','#bigdata'],
                ...     120,
                ...     15,
                ...     toTimestamp(now())
                ... );
```

```
cqlsh:socialmedia> INSERT INTO posts (
               ...     postid,
               ...     userid,
               ...     content,
               ...     hashtags,
               ...     likes,
               ...     comments,
               ...     post_time
               ... )
               ... VALUES (
               ...     uuid(),
               ...     uuid(),
               ...     'AI and Machine Learning',
               ...     ['#AI','#ML'],
               ...     450,
               ...     60,
               ...     toTimestamp(now())
               ... );
```

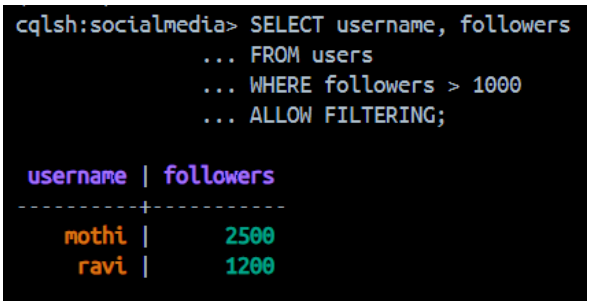
```
cqlsh:socialmedia> SELECT * FROM posts;
```

```
cqlsh:socialmedia> SELECT * FROM posts;

 postid                               | comments | content                  | hashtags                  | likes | post_time                       | userid
--------------------------------------+----------+--------------------------+---------------------------+-------+---------------------------------+--------------------------------------
 4ec9d762-6002-44b0-94aa-5ab9128067e2 |       60 |     AI and Machine Learning |           ['#AI', '#ML'] |   450 | 2026-05-12 04:55:07.234000+0000 | 71dc0287-60f9-4be6-a140-714883721985
 1809167a-010e-4d56-87c8-babca81d16c8 |       15 | Learning Cassandra Big Data | ['#cassandra', '#bigdata'] |   120 | 2026-05-12 04:55:05.915000+0000 | fec2c893-93a6-40de-9269-86b8898b7186

(2 rows)
```

```
cqlsh:socialmedia> SELECT * FROM posts
               ... WHERE likes > 100
               ... ALLOW FILTERING;
```

```
cqlsh:socialmedia> SELECT * FROM posts
               ... WHERE likes > 100
               ... ALLOW FILTERING;

 postid                               | comments | content                  | hashtags                  | likes | post_time                       | userid
--------------------------------------+----------+--------------------------+---------------------------+-------+---------------------------------+--------------------------------------
 4ec9d762-6002-44b0-94aa-5ab9128067e2 |       60 |     AI and Machine Learning |           ['#AI', '#ML'] |   450 | 2026-05-12 04:55:07.234000+0000 | 71dc0287-60f9-4be6-a140-714883721985
 1809167a-010e-4d56-87c8-babca81d16c8 |       15 | Learning Cassandra Big Data | ['#cassandra', '#bigdata'] |   120 | 2026-05-12 04:55:05.915000+0000 | fec2c893-93a6-40de-9269-86b8898b7186

(2 rows)
```

```
cqlsh:socialmedia> SELECT username, followers
               ... FROM users
               ... WHERE followers > 1000
               ... ALLOW FILTERING;
```



```
cqlsh:socialmedia> SELECT username, followers
               ... FROM users
               ... WHERE followers > 1000
               ... ALLOW FILTERING;

 username | followers
----------+-----------
    mothi |      2500
     ravi |      1200
```

Experiment-5: Buyer event analytics using Cassandra on suitable product sales data.

Step-1: Create Database

```
cqlsh> CREATE KEYSPACE ecommerce
   ... WITH replication = {
   ...     'class':'SimpleStrategy',
   ...     'replication_factor':1
   ... };
cqlsh> use ecommerce;
```

Step-2: Create Tables

```
cqlsh:ecommerce> CREATE TABLE products (
                ...     product_id uuid PRIMARY KEY,
                ...     product_name text,
                ...     category text,
                ...     price decimal,
                ...     stock int
                ... );
cqlsh:ecommerce> CREATE TABLE buyer_events (
                ...     buyer_id uuid,
                ...     event_time timestamp,
                ...     event_type text,
                ...     product_name text,
                ...     category text,
                ...     amount decimal,
                ...     device text,
                ...     city text,
                ...     PRIMARY KEY (buyer_id, event_time)
                ... ) WITH CLUSTERING ORDER BY (event_time DESC);
```

Step-3: Insert Data into Tables

```
cqlsh:ecommerce> INSERT INTO products (
                ...     product_id,
                ...     product_name,
                ...     category,
                ...     price,
                ...     stock
                ... )
                ... VALUES (
                ...     uuid(),
                ...     'iPhone 15',
                ...     'Mobiles',
                ...     79999,
                ...     50
                ... );

cqlsh:ecommerce> INSERT INTO products (
                ...     product_id,
                ...     product_name,
                ...     category,
                ...     price,
                ...     stock
                ... )
                ... VALUES (
                ...     uuid(),
                ...     'Dell Laptop',
                ...     'Computers',
```

```
                ...     65000,
                ...     30
                ... );

cqlsh:ecommerce> INSERT INTO products (
                ...     product_id,
                ...     product_name,
                ...     category,
                ...     price,
                ...     stock
                ... )
                ... VALUES (
                ...     uuid(),
                ...     'Boat Headphones',
                ...     'Accessories',
                ...     1999,
                ...     120
                ... );
cqlsh:ecommerce> INSERT INTO buyer_events (
                ...     buyer_id,
                ...     event_time,
                ...     event_type,
                ...     product_name,
                ...     category,
                ...     amount,
                ...     device,
                ...     city
                ... )
                ... VALUES (
                ...     uuid(),
                ...     toTimestamp(now()),
                ...     'VIEW',
                ...     'iPhone 15',
                ...     'Mobiles',
                ...     79999,
                ...     'Mobile',
                ...     'Hyderabad'
                ... );

cqlsh:ecommerce> INSERT INTO buyer_events (
                ...     buyer_id,
                ...     event_time,
                ...     event_type,
                ...     product_name,
                ...     category,
                ...     amount,
                ...     device,
                ...     city
                ... )
                ... VALUES (
                ...     uuid(),
                ...     toTimestamp(now()),
                ...     'PURCHASE',
                ...     'Dell Laptop',
                ...     'Computers',
                ...     65000,
                ...     'Laptop',
                ...     'Delhi'
                ... );
```

```
cqlsh:ecommerce> INSERT INTO buyer_events (
             ...     buyer_id,
             ...     event_time,
             ...     event_type,
             ...     product_name,
             ...     category,
             ...     amount,
             ...     device,
             ...     city
             ... )
             ... VALUES (
             ...     uuid(),
             ...     toTimestamp(now()),
             ...     'CART',
             ...     'Boat Headphones',
             ...     'Accessories',
             ...     1999,
             ...     'Mobile',
             ...     'Mumbai'
             ... );
```

Step-4: View Table Data

```
cqlsh:ecommerce> SELECT * FROM buyer_events;
```

buyer_id	event_time	amount	category	city	device	event_type	product_name
683dd0e0-335d-4ee1-9952-c8ab6b9a3a56	2026-05-12 05:46:44.795000+0000	1999	Accessories	Mumbai	Mobile	CART	Boat Headphones
9eb2598c-e82c-4bc3-85dc-d1efc08a7c1c	2026-05-12 05:46:44.290000+0000	79999	Mobiles	Hyderabad	Mobile	VIEW	iPhone 15
8af3a1b4-24bf-4aa0-ae24-b1ce06308d97	2026-05-12 05:46:44.299000+0000	65000	Computers	Delhi	Laptop	PURCHASE	Dell Laptop

```
cqlsh:ecommerce> SELECT * FROM products;
```

product_id	category	price	product_name	stock
7dbd2869-76f7-4740-b165-aa90147d9160	Mobiles	79999	iPhone 15	50
c2a09e0c-df5d-4d25-a175-5a06257e6d84	Computers	65000	Dell Laptop	30
319ea2e8-50e5-432a-8c1c-d07620269826	Accessories	1999	Boat Headphones	120

Step-5: Apply different queries on tables

```
cqlsh:ecommerce> SELECT *
             ... FROM buyer_events
             ... WHERE event_type = 'PURCHASE'
             ... ALLOW FILTERING;
```

buyer_id	event_time	amount	category	city	device	event_type	product_name
8af3a1b4-24bf-4aa0-ae24-b1ce06308d97	2026-05-12 05:46:44.299000+0000	65000	Computers	Delhi	Laptop	PURCHASE	Dell Laptop

```
cqlsh:ecommerce> SELECT *
             ... FROM buyer_events
             ... WHERE device = 'Mobile'
             ... ALLOW FILTERING;
```

buyer_id	event_time	amount	category	city	device	event_type	product_name
683dd0e0-335d-4ee1-9952-c8ab6b9a3a56	2026-05-12 05:46:44.795000+0000	1999	Accessories	Mumbai	Mobile	CART	Boat Headphones
9eb2598c-e82c-4bc3-85dc-d1efc08a7c1c	2026-05-12 05:46:44.290000+0000	79999	Mobiles	Hyderabad	Mobile	VIEW	iPhone 15

```
cqlsh:ecommerce> SELECT *
             ... FROM buyer_events
             ... WHERE city = 'Hyderabad'
             ... ALLOW FILTERING;
```

buyer_id	event_time	amount	category	city	device	event_type	product_name
9eb2598c-e82c-4bc3-85dc-d1efc08a7c1c	2026-05-12 05:46:44.290000+0000	79999	Mobiles	Hyderabad	Mobile	VIEW	iPhone 15

Experiment-6: Using Power Pivot (Excel) Perform the following on any dataset
a) Big Data Analytics b) Big Data Charting

For Excel 2010, you must manually download and install the Power Pivot add-in from Microsoft.

Choose Correct Version

- PowerPivot_for_Excel_x86.msi → for **32-bit Excel 2010**
- PowerPivot_for_Excel_amd64.msi → for **64-bit Excel 2010**

How to Check Excel Bit Version

1. Open Excel 2010
2. File → Help
3. Look at "About Microsoft Excel"
4. It will show:
    - 32-bit
    - or 64-bit

Installation Steps

1. Install:
    - .NET Framework 4.0 or newer
    - Office 2010 Service Pack
2. Download Power Pivot MSI
3. Run installer
4. Open Excel
5. File → Options → Add-ins
6. Manage: COM Add-ins → Go
7. Enable:
    - **Microsoft Power Pivot for Excel**

(a) Big Data Analytics using Power Pivot

Step 1: Open Excel

Open **Excel 2010/2013/2016/365**

Step 2: Enable Power Pivot

1. File → Options
2. Add-ins
3. Manage COM Add-ins → Go
4. Enable **Microsoft Power Pivot for Excel**

Step 3: Import Dataset

1. Open Power Pivot tab
2. Click **Manage**
3. Home → Get External Data
4. Select:
    - From Text/CSV
5. Choose dataset file
6. Click Finish

Step 4: Create Data Model

Dataset loads into Power Pivot Data Model.

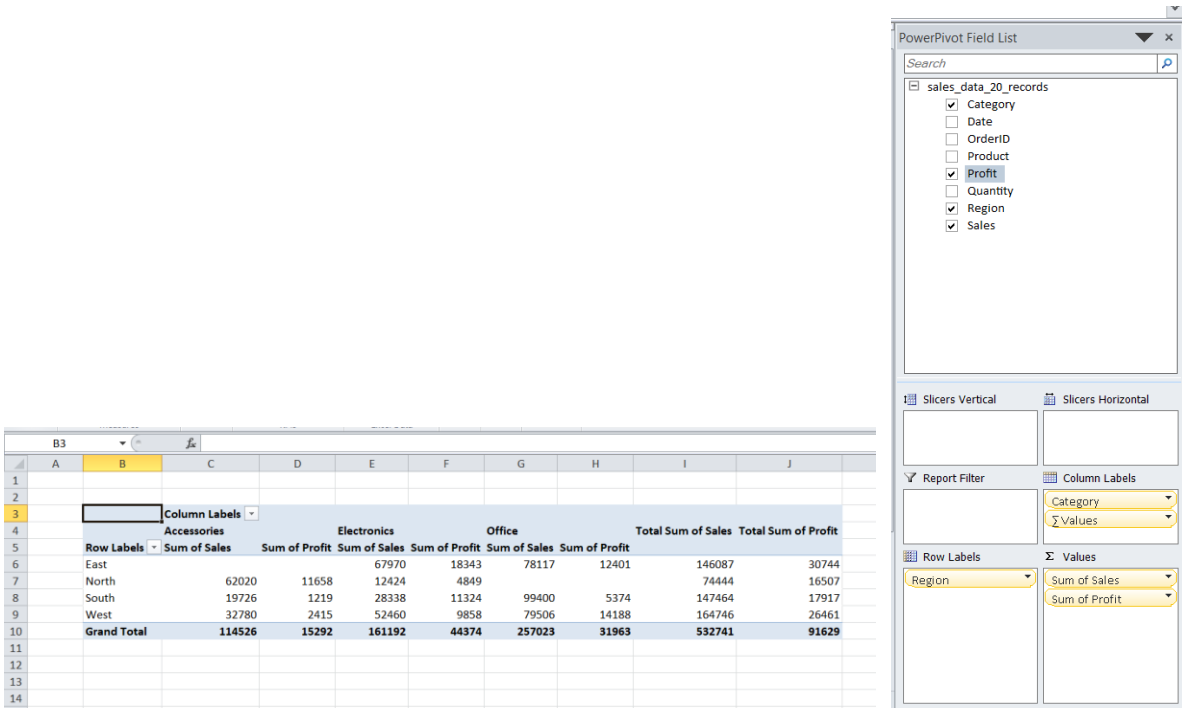
Verify columns:

- Sales
- Profit
- Quantity
- Region
- Category

Step 5: Analyze Data

Create Pivot Table:

1. Home → PivotTable
2. Add:
    - Rows → Region
    - Columns → Category
    - Values → Total Sales, Total Profit



Row Labels	Column Labels: Accessories Sum of Sales	Accessories Sum of Profit	Electronics Sum of Sales	Electronics Sum of Profit	Office Sum of Sales	Office Sum of Profit	Total Sum of Sales	Total Sum of Profit
East			67970	18343	78117	12401	146087	30744
North	62020	11658	12424	4849			74444	16507
South	19726	1219	28338	11324	99400	5374	147464	17917
West	32780	2415	52460	9858	79506	14188	164746	26461
Grand Total	**114526**	**15292**	**161192**	**44374**	**257023**	**31963**	**532741**	**91629**

(b) Big Data Charting using Power Pivot

Step 1: Create Pivot Chart

1. Select Pivot Table
2. Insert → Pivot Chart

Step 2: Generate Charts

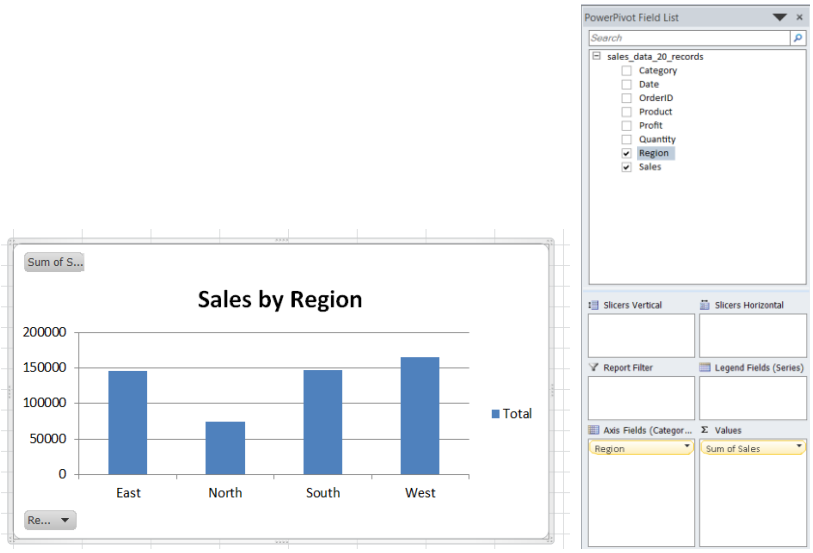
1. Sales by Region

Chart Type:

- Column Chart

Purpose:

- Compare regional sales performance



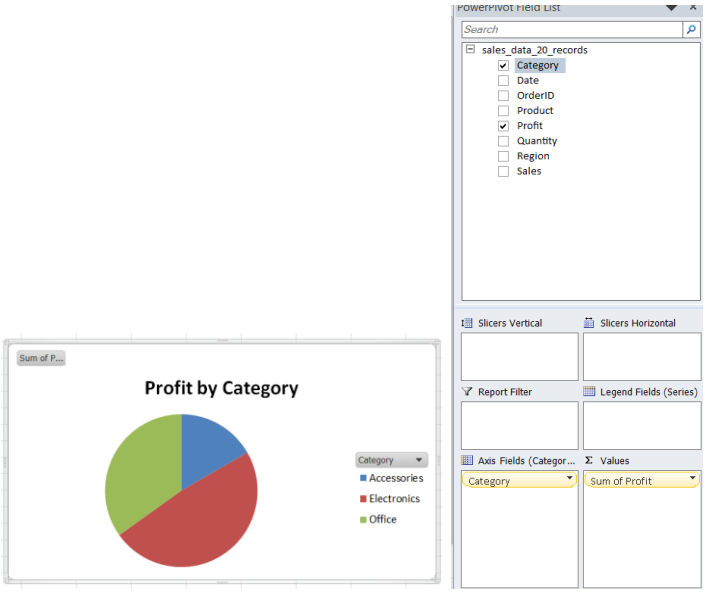
2. Profit by Category

Chart Type:

- Pie Chart

Purpose:

- Show profit contribution of each category



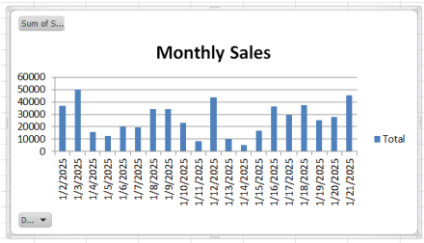
3. Monthly Sales Trend

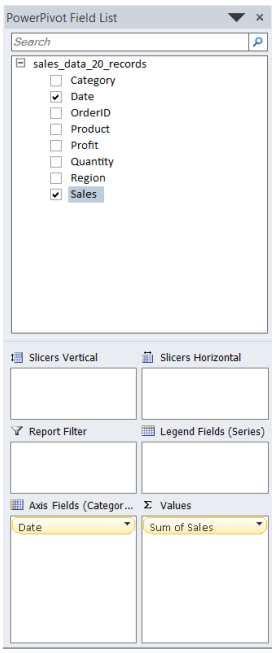
Chart Type:

- Line Chart

Purpose:

- Analyze sales growth over time





Experiment-7: Use R-Project to carry out statistical analysis of big data

Sample Dataset

Example: Student Performance Dataset

ID	Name	Branch	Marks	Attendance	Placement
1	Rahul	CSE	85	90	Yes
2	Sneha	ECE	78	88	No
3	Arjun	IT	92	95	Yes

Save as: **`students.csv`**

Procedure

Step 1: Install Required Packages

```
install.packages("ggplot2")
install.packages("dplyr")
```

Step 2: Load Libraries

```
library(ggplot2)
library(dplyr)
```

Step 3: Import Dataset

```
data <- read.csv("students.csv")
```

Step 4: View Dataset Structure

```
str(data)
summary(data)
```

```
> str(data)
'data.frame':   500 obs. of  6 variables:
 $ ID        : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Name      : chr  "Karina" "Jessica" "Mitchell" "Jacob" ...
 $ Branch    : chr  "IT" "MECH" "ECE" "ECE" ...
 $ Marks     : int  93 95 54 45 64 92 65 81 81 80 ...
 $ Attendance: int  73 68 70 61 74 92 96 81 90 99 ...
 $ Placement : chr  "Yes" "Yes" "No" "Yes" ...
> summary(data)
       ID            Name              Branch              Marks          Attendance      Placement
 Min.   :  1.0   Length:500         Length:500         Min.   : 40.00   Min.   : 60.00   Length:500
 1st Qu.:125.8   Class :character   Class :character   1st Qu.: 56.00   1st Qu.: 69.00   Class :character
 Median :250.5   Mode  :character   Mode  :character   Median : 70.00   Median : 79.00   Mode  :character
 Mean   :250.5                                         Mean   : 70.63   Mean   : 79.90
 3rd Qu.:375.2                                         3rd Qu.: 86.00   3rd Qu.: 90.25
 Max.   :500.0                                         Max.   :100.00   Max.   :100.00
```

Step 5: Statistical Analysis

```
> mean(data$Marks)
[1] 70.632
> median(data$Marks)
[1] 70
> sd(data$Marks)
[1] 17.82583
> var(data$Marks)
[1] 317.7601
> max(data$Marks)
[1] 100
> min(data$Marks)
[1] 40
```

Experiment-8: Use R-Project for data visualization of social media data

Sample Dataset

Create a CSV file named social_media.csv

PostID	Platform	Likes	Comments	Shares	Followers
1	Instagram	120	35	20	5000
2	Facebook	95	18	12	4200
3	Twitter	80	15	25	3900
4	Instagram	150	40	30	5500
5	YouTube	300	85	60	10000

Save as: **social_media.csv**

Procedure

Step 1: Install Required Packages

```
install.packages("ggplot2")
install.packages("dplyr")
```

Step 2: Load Libraries

```
library(ggplot2)
library(dplyr)
```

Step 3: Import Dataset

```
data <- read.csv("social_media.csv")
```

Step 4: View Dataset Information

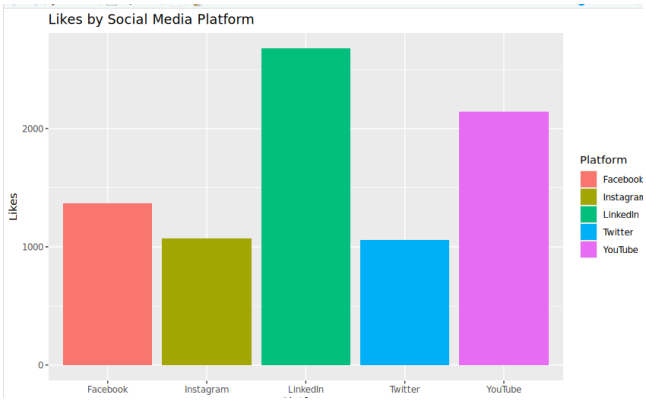
```
str(data)
summary(data)
```

```
> str(data)
'data.frame':   20 obs. of  7 variables:
 $ PostID     : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Platform   : chr  "YouTube" "Facebook" "LinkedIn" "Facebook" ...
 $ ContentType: chr  "Post" "Story" "Story" "Story" ...
 $ Likes      : int  915 463 645 907 193 303 193 52 376 658 ...
 $ Comments   : int  200 30 189 76 184 118 276 270 91 225 ...
 $ Shares     : int  46 98 57 8 35 137 97 82 52 56 ...
 $ Followers  : int  17279 47332 27765 31067 45782 11866 1197 15385 35643 22624 ...
> summary(data)
     PostID        Platform         ContentType           Likes          Comments        Shares         Followers
 Min.   : 1.00   Length:20          Length:20          Min.   : 52.0   Min.   : 24.0   Min.   :  8.00   Min.   : 1197
 1st Qu.: 5.75   Class :character   Class :character   1st Qu.:193.0   1st Qu.:111.2   1st Qu.: 37.25   1st Qu.:15278
 Median :10.50   Mode  :character   Mode  :character   Median :349.0   Median :194.5   Median : 56.50   Median :26230
 Mean   :10.50                                         Mean   :415.7   Mean   :172.8   Mean   : 69.80   Mean   :25920
 3rd Qu.:15.25                                         3rd Qu.:527.2   3rd Qu.:227.5   3rd Qu.: 97.25   3rd Qu.:36454
 Max.   :20.00                                         Max.   :942.0   Max.   :276.0   Max.   :145.00   Max.   :47563
```

Data Visualization Techniques

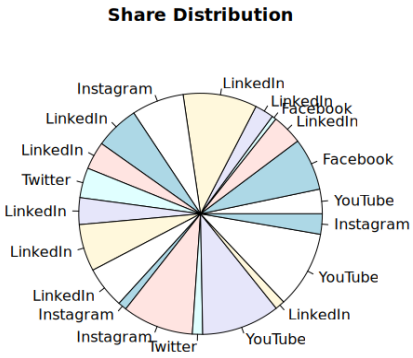
1. Bar Chart – Likes by Platform

```
ggplot(data, aes(x=Platform, y=Likes, fill=Platform)) +
  geom_bar(stat="identity") +
  ggtitle("Likes by Social Media Platform")
```



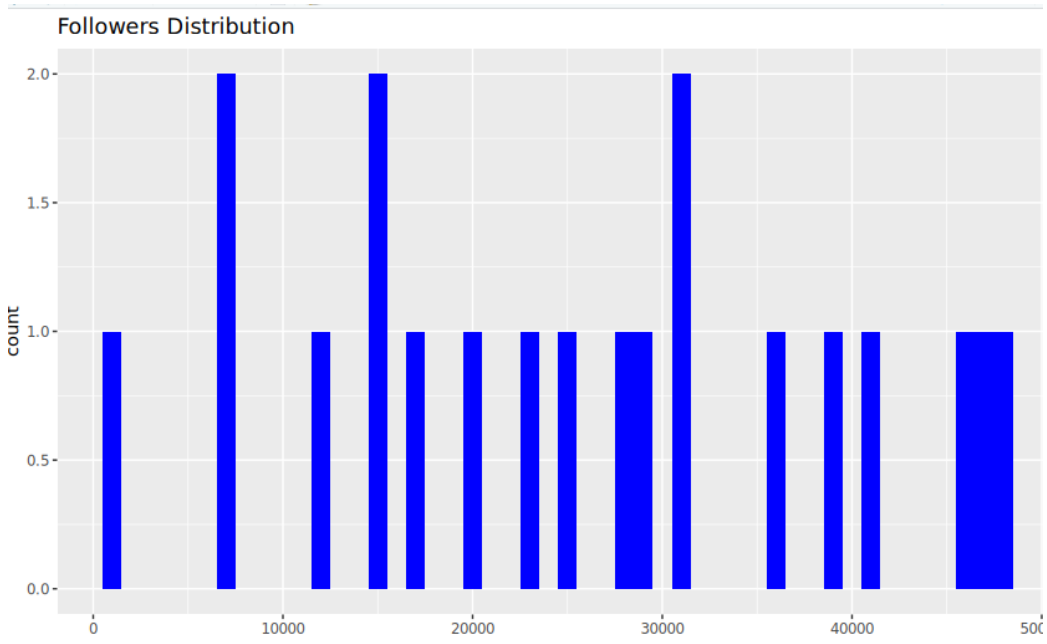
2. Pie Chart – Share Distribution

```
shares <- data$Shares
labels <- data$Platform

pie(shares,
    labels=labels,
    main="Share Distribution")
```



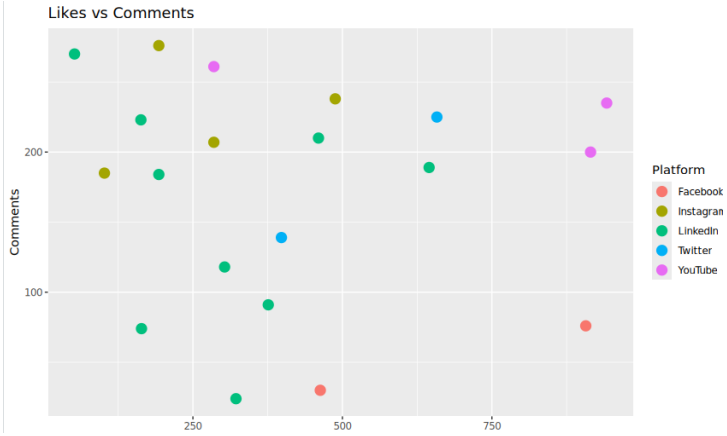
3. Histogram – Followers Count

```
ggplot(data, aes(x=Followers)) +
  geom_histogram(binwidth=1000, fill="blue") +
  ggtitle("Followers Distribution")
```



4. Scatter Plot – Likes vs Comments

```
ggplot(data,
       aes(x=Likes, y=Comments, color=Platform)) +
  geom_point(size=4) +
  ggtitle("Likes vs Comments")
```



5. Line Chart – Followers Trend

```
ggplot(data,
       aes(x=PostID, y=Followers, color=Platform)) +
  geom_line() +
  geom_point(size=3) +
  ggtitle("Followers Trend")
```