Ex.No:6	**Install and Configure Hadoop**
Date : 14/8/23	

Aim:

To Install and Configure Hadoop

Setup:

Setting up Hadoop on Windows:

Prerequisites:

- Windows 64-bit OS

- Java JDK 8+

- Administrator access

Installation:

1. Download Hadoop for Windows from [GitHub](https://github.com/cybermaggedon/hadooponazure).

2. Extract the ZIP to your chosen directory (Hadoop installation).

Configuration:

3. Set environment variables:

   - `HADOOP_HOME` (Hadoop installation path)

   - `JAVA_HOME` (Java JDK installation path)

   - Edit the "Path" variable and add:

     - `%HADOOP_HOME%\bin`

     - `%HADOOP_HOME%\sbin`

     - `%JAVA_HOME%\bin`

4. Configure Hadoop:

a. Open `hadoop-env.cmd`, set `JAVA_HOME`.

b. Create/edit `core-site.xml` with:

```
xml
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://localhost:9000</value>
</property>
```

c. Create/edit `hdfs-site.xml` with:

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

d. Create/edit `yarn-site.xml` with:

```
xml
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.env-whitelist</name>
<value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_H
```

```
OME,HADOOP_CONF_DIR,CLASSPATH_PERPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
</property>
```

Formatting HDFS:

5. Open a command prompt and run:

```
hdfs namenode -format
```

Starting Hadoop Services:

6. In a command prompt:

```
cd %HADOOP_HOME%
start-dfs.cmd (start HDFS)
start-yarn.cmd (start YARN)
```

Testing Hadoop:

7. Open a browser and visit `http://localhost:9870` to see Hadoop NameNode web interface.

Running a MapReduce Example:

8. To test, run a MapReduce example (replace the jar file):

```
hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-.jar pi 16 1000
```

Observation	20	20
Record	15	15
Viva	10	09
Total	45	44

Result:

Thus the Install and Configure Hadoop successfully.

Ex.No:7	
Date : 21/8/23	Implementation of word count programs using Map Reduce

Aim:

To Implementation of word count programs using Map Reduce.

Prepare:

1. Download MapReduceClient.jar (Link: https://github.com/MuhammadBilalYar/HADOOP- INSTALLATION-ONWINDOW-10/blob/master/MapReduceClient.jar)

2. Download Input_file.txt (Link: https://github.com/MuhammadBilalYar/HADOOPINSTALLATION-ON-WINDOW-10/blob/master/input_file.txt) Place both files in "C:/"

Hadoop Operation:

1. Open cmd in Administrative mode and move to "C:/Hadoop-2.8.0/sbin" andstart cluster

2. **Start-all.cmd**

3. Create an input directory in HDFS.

```
hadoop fs -mkdir /input_dir
```

4. Copy the input text file named input_file.txt in the input directory (input_dir)of HDFS.

```
hadoop fs -put C:/input_file.txt /input_dir
```

Verify input_file.txt available in HDFS input director

```
hadoop fs -ls /input_dir/
```

5. Run MapReduceClient.jar and also provide input and out directories.

```
hadoop jar C:/MapReduceClient.jar wordcount /input_dir /output_dir
```

6. Verify content for generated output file.

```
hadoop dfs -cat /output_dir/
```

Some Other useful commands :

8) To leave Safe mode

```
hadoop dfsadmin –safemode leave
```

9) To delete file from HDFS directory

```
hadoop fs -rm -r /iutput_dir/input_file.txt
```

10) To delete directory from HDFS directory

```
hadoop fs - rm -r /iutput_dir
```

Observation	20	20
Record	15	15
Viva	10	09
Total	45	44

Result:

Thus the word count programs using Map Reduce successfully Executed and verified.

Ex.No:8	Implement an application that stores big data in Hbase /MongoDB / Pig using Hadoop / R/Cassandra
Date : 28/8/23	

Aim:

To Implement an application that stores big data in Hbase /MongoDB / Pig using Hadoop / R/Cassandra.

Program:

Step 1: Set up the Environment

You should have Hadoop, HBase, MongoDB, Pig, R, and Cassandra installed and configured.

Step 2: Sample Dataset

Let's assume you have a simple CSV dataset named "sample_data.csv" like this:

```
Name, Age, City

John, 30, New York

Alice, 25, Los Angeles

Bob, 35, Chicago
```

Step 3: Hadoop MapReduce (Python)

Python MapReduce script to process the dataset and store it in HBase:

```
# Mapper

import sys

for line in sys.stdin:

    line = line.strip()

    fields = line.split(',')

    name, age, city = fields

    print(f"{name}\t{age}\t{city}")
```

```
# Reducer

import happybase

connection = happybase.Connection(host='localhost', port=9090)

table = connection.table('my_table')

for line in sys.stdin:

    name, age, city = line.strip().split('\t')

    table.put(name, {'info:age': age, 'info:city': city})
```

Step 4: Pig Script

A Pig script to perform some transformations and store the data in MongoDB:

```
data = LOAD 'sample_data.csv' USING PigStorage(',') AS (name:chararray,
age:int, city:chararray);

filtered_data = FILTER data BY age >= 30;

STORE filtered_data INTO 'mongodb://localhost:27017/mydb.mycollection'
USING
org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.PigMongoSt
orage();
```

Step 5: R Script

An R script to analyze the data:

```
library(rhbase)

hbase <- HBase$new(host = "localhost", port = 9090)

data <- hbase$get("my_table")

print(data)

# Your R data analysis code here
```

Step 6: Cassandra (CQL)

You would create a Cassandra keyspace, define a table schema, and insert data using CQL commands. Below is a simplified example:

cql

```
CREATE KEYSPACE mykeyspace WITH replication = {'class': 'SimpleStrategy',
'replication_factor': 1};

USE mykeyspace;

CREATE TABLE mytable (name TEXT PRIMARY KEY, age INT, city TEXT);

INSERT INTO mytable (name, age, city) VALUES ('John', 30, 'New York');
```

Step 7: Running the Code

Run the Python MapReduce script using Hadoop, execute the Pig script, run the R script, and execute the CQL commands in Cassandra.

Observation	20	20
Record	15	15
Viva	10	09
Total	45	44

Result:

Thus the Implement an application that stores big data in Hbase /MongoDB / Pig using Hadoop / R/Cassandra has been executed successfully