Unit 3

1) Explain HDFS architecture.

he Hadoop Distributed File System (HDFS) is designed to provide a fault-tolerant file system designed to run on commodity hardware. The primary objective of HDFS is to store data reliably even in the presence of failures including Name Node failures, Data Node failures and network partitions.

HDFS uses a master/slave architecture in which one device (the master) controls one or more other devices (the slaves). The HDFS cluster consists of a single Name Node and a master server manages the file system namespace and regulates access to files.

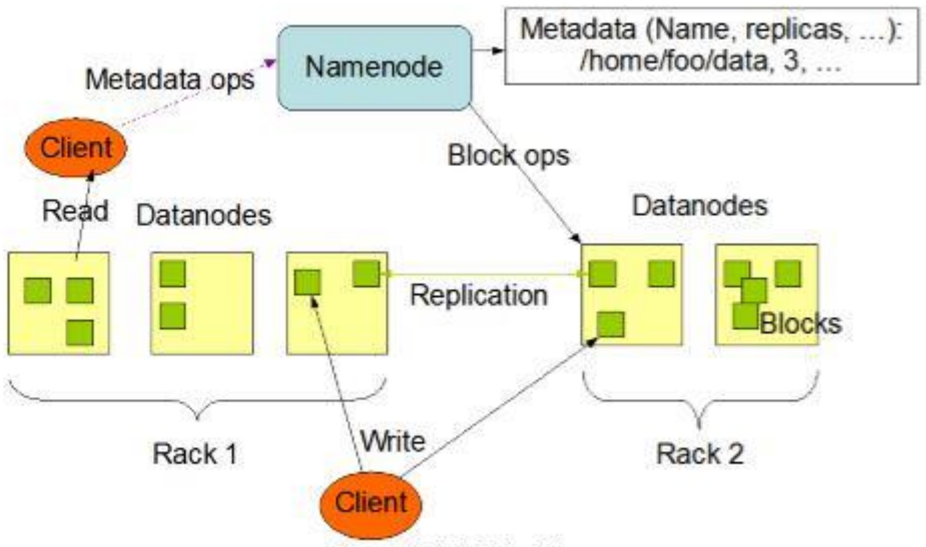

Figure: HDFS Architecture

The main components of HDFS are as described below:

NameNode and DataNodes:

HDFS has a master/slave architecture. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system’s clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

The NameNode and DataNode are pieces of software designed to run on commodity machines. These machines typically run a GNU/Linux operating system (OS). HDFS is built using the Java language; any machine that supports Java can run the NameNode or the DataNode software. Usage of the highly portable Java language means that HDFS can be deployed on a wide range of machines. A typical deployment has a dedicated machine that runs only the NameNode software. Each of the other machines in the cluster runs one instance of the DataNode software. The architecture does not preclude running multiple DataNodes on the same machine but in a real deployment that is rarely the case. The existence of a single NameNode in a cluster greatly simplifies the architecture of the system. The NameNode is the arbitrator and repository for all HDFS metadata. The system is designed in such a way that user data never flows through the NameNode.

The File System Namespace:

HDFS supports a traditional hierarchical file organization. A user or an application can create directories and store files inside these directories. The file system namespace hierarchy is similar to most other existing file systems; one can create and remove files, move a file from one directory to another, or rename a file. HDFS does not yet implement user quotas. HDFS does not support hard links or soft links. However, the HDFS architecture does not preclude implementing these features.

The NameNode maintains the file system namespace. Any change to the file system namespace or its properties is recorded by the NameNode. An application can specify the number of replicas of a file that should be maintained by HDFS. The number of copies of a file is called the replication factor of that file. This information is stored by the NameNode.

Data Replication:

HDFS is designed to reliably store very large files across machines in a large cluster. It stores each file as a sequence of blocks; all blocks in a file except the last block are the same size. The blocks of a file are replicated for fault tolerance. The block size and replication factor are configurable per file. An application can specify the number of replicas of a file. The replication factor can be specified at file creation time and can be changed later. Files in HDFS are write-once and have strictly one writer at any time.

The NameNode makes all decisions regarding replication of blocks. It periodically receives a Heartbeat and a Blockreport from each of the DataNodes in the cluster. Receipt of a Heartbeat implies that the DataNode is functioning properly. A Blockreport contains a list of all blocks on a DataNode.

2) Explain different HDFS commands.

3.1. version

Hadoop HDFS version Command Usage

```
version
```

Hadoop HDFS version Command Example

```
hdfs dfs version
```

Hadoop HDFS version Command Description

This Hadoop command prints the **Hadoop** version

3.2. mkdir

Hadoop HDFS mkdir Command Usage

```
mkdir <path>
```

Hadoop HDFS mkdir Command Example

```
hdfs dfs -mkdir /user/dataflair/dir1
```

Hadoop HDFS mkdir Command Description

This HDFS command takes path URI's as an argument and creates directories. Creates any parent directories in path that are missing (e.g., mkdir -p in Linux).

Learn various features of Hadoop HDFS from this **HDFS features guide**.

3.3. ls

Hadoop HDFS ls Command Usage

```
ls <path>
```

Hadoop HDFS ls Command Example

```
hdfs dfs -ls /user/dataflair/dir1
```

Hadoop HDFS ls Commnad Description

This Hadoop HDFS ls command displays a list of the contents of a directory specified by path provided by the user, showing the names, permissions, owner, size and modification date for each entry.

Hadoop HDFS ls Command Example

```
hdfs dfs -ls -R
```

Hadoop HDFS ls Description

This Hadoop fs command behaves like -ls, but recursively displays entries in all subdirectories of a path.

3.4. put

Hadoop HDFS put Command Usage

```
put <localSrc> <dest>
```

Hadoop HDFS put Command Example

```
hdfs dfs -put /home/dataflair/Desktop/sample /user/dataflair/dir1
```

Hadoop HDFS put Command Description

This hadoop basic command copies the file or directory from the local file system to the destination within the DFS.

Learn Internals of **HDFS Data Write Pipeline and File write execution flow**.

3.5. copyFromLocal

Hadoop HDFS copyFromLocal Command Usage

```
copyFromLocal <localSrc> <dest>
```

Hadoop HDFS copyFromLocal Command Example

```
hdfs dfs -copyFromLocal /home/dataflair/Desktop/sample /user/dataflair/dir1
```

Hadoop HDFS copyFromLocal Command Description

This hadoop shell command is similar to put command, but the source is restricted to a local file reference.

Learn **Internals of HDFS Data Read Operation, How Data flows in HDFS while reading the file.**

Any Doubt yet in Hadoop HDFS Commands? Please Comment.

3.6. get

Hadoop HDFS get Command Usage

```
get [-crc] <src> <localDest>
```

Hadoop HDFS get Command Example

```
hdfs dfs -get /user/dataflair/dir2/sample /home/dataflair/Desktop
```

Hadoop HDFS get Command Description

This HDFS fs command copies the file or directory in HDFS identified by the source to the local file system path identified by local destination.

Hadoop HDFS get Command Example

```
hdfs dfs -getmerge /user/dataflair/dir2/sample /home/dataflair/Desktop
```

Hadoop HDFS get Command Description

This HDFS basic command retrieves all files that match to the source path entered by the user in HDFS, and creates a copy of them to one single, merged file in the local file system identified by local destination.

Hadoop HDFS get Command Example

```
hadoop fs -getfacl /user/dataflair/dir1/sample
hadoop fs -getfacl -R /user/dataflair/dir1
```

Hadoop HDFS get Command Description

This Apache Hadoop command shows the Access Control Lists (ACLs) of files and directories. If a directory contains a default ACL, then getfacl also displays the default ACL.

Options :

-R: It displays a list of all the ACLs of all files and directories recursively.

path: File or directory to list.

Hadoop HDFS get Command Example

```
hadoop fs -getfattr -d /user/dataflair/dir1/sample
```

Hadoop HDFS get Command Description

This HDFS file system command displays if there is any extended attribute names and values for a file or directory.

Options:

-R: It recursively lists the attributes for all files and directories.

-n name: It displays the named extended attribute value.

-d: It displays all the extended attribute values associated with the pathname.

-e encoding: Encodes values after extracting them. The valid converted coded forms are "text", "hex", and "base64". All the values encoded as text strings are with double quotes (" "), and prefix 0x and 0s are used for all the values which are converted and coded as hexadecimal and base64.

path: The file or directory.

Learn: **Rack Awareness**, **High Availability**

3.7. copyToLocal

Hadoop HDFS copyToLocal Command Usage

```
copyToLocal <src> <localDest>
```

Hadoop HDFS copyToLocal Command Example

```
hdfs dfs -copyToLocal /user/dataflair/dir1/sample /home/dataflair/Desktop
```

Hadoop HDFS copyToLocal Description

Similar to get command, only the difference is that in this the destination is restricted to a local file reference.

3.8. cat

Hadoop HDFS cat Command Usage

```
cat <file-name>
```

Hadoop HDFS cat Command Example

```
hdfs dfs -cat /user/dataflair/dir1/sample
```

Hadoop HDFS cat Command Description

This Hadoop fs shell command displays the contents of the filename on console or stdout.

3.9. mv

Hadoop HDFS mv Command Usage

```
mv <src> <dest>
```

Hadoop HDFS mv Command Example

```
hadoop fs -mv /user/dataflair/dir1/purchases.txt /user/dataflair/dir2
```

Hadoop HDFS mv Command Description

This basic HDFS command moves the file or directory indicated by the source

to destination, within HDFS.
Learn: HDFS Disk Balancer and Erasure coding

3.10. cp

Hadoop HDFS cp Command Usage

```
cp <src> <dest>
```

Hadoop HDFS cp Command Example

```
hadoop fs -cp /user/dataflair/dir2/purchases.txt /user/dataflair/dir1
```

Hadoop HDFS cp Command Description

This Hadoop File system shell command copies the file or directory identified by the source to destination, within HDFS.

3) What is HIVE? Explain architecture of HIVE.

4) Explain working of HIVE

https://www.tutorialspoint.com/hive/hive_introduction.htm
(refer prajval also for architecture)

5) Explain HIVE data types.

https://www.tutorialspoint.com/hive/hive_data_types.htm

6) Explain HIVEQL for order by, group by, join

https://www.tutorialspoint.com/hive/hiveql_select_order_by.htm

7) **Explain HBase architecture.**

https://www.tutorialspoint.com/hbase/hbase_architecture.htm

8) **Explain Pig.**

https://www.tutorialspoint.com/apache_pig/apache_pig_architecture.htm

9) **ExplainZookeeper.**

https://www.tutorialspoint.com/zookeeper/zookeeper_overview.htm