HADOOP COMMANDS

To Start Hadoop (Present Working Directory should be Hadoop Folder)

bin/start-all.sh

If you are inside **BIN** folder of Hadoop Folder

./start-all.sh

To Start Hadoop (Present Working Directory should be Hadoop Folder)

bin/stop-all.sh

./stop-all.sh

To start **HDFS** Daemon process – **start-dfs.sh** -> To Stop: **stop-dfs.sh**
To start **MR** Daemon process – **start-mapred .sh** -> To Stop: **stop-mapred.sh**

Hadoop allows to **start/stop** each and every daemons by below commands.

Daemon Process	Command
Name Node	hadoop-daemon.sh start namenode
Data Node	hadoop-daemon.sh start datanode
Job Tracker	hadoop-daemon.sh start jobtracker
Task Tracker	hadoop-daemon.sh start tasktracker
Secondary Name Node	hadoop-daemon.sh start secondarynamenode
To stop the respective daemons, use stop instead of start. Ex hadoop-daemon.sh **stop** secondarynamenode	

Note:
HDFS Daemons – NN, SNN, JT
MR Daemons – DN, TT

To Delete Name Node

bin/hadoop namenode -format

Note: This command should be EXECUTED only when the Hadoop is getting installed. It is advised not to use this command once data are inserted into HDFS

To create Secure Key

ssh-keygen -t rsa

To Create Directory

hadoop dfs -mkdir /NewDirectory/Dir1

To Copy File

From Local System to HDFS

bin/hadoop dfs -copyFromLocal /home/administrator/R3.jar /ClientJavaF

From HDFS to Local

bin/hadoop dfs -copyToLocal /ClientJavaFolder/HadoopAddFile.txt /home/administrator/Desktop/Hadoop/NewFileCopied.txt

From HDFS to HDFS

bin/hadoop dfs -cp /ClientJavaFolder/HadoopAddFile.txt /myNewDir/hello

Note: Directory name should not have SPACE or SPECIAL characters [**(** ,) , **&** , **:** , < , >]

To Move : File will be copied and file from copied path will be deleted.

From Local System to HDFS

bin/hadoop dfs -moveFromLocal /home/administrator/Desktop/Hadoop/NewFileCopied.txt /NewFile/MovedFile.txt

From HDFS to HDFS

/hadoop dfs -mv /NewFile/MovedFile.txt /a.txt

To list the files available

bin/hadoop dfs -ls /

Note: between ls and / - space must be provided.

To Delete from HDFS

To Delete File inside Dir

bin/hadoop dfs -rm /ClientJavaFolder/NewRenameFile.txt

To Delete Directory

bin/hadoop dfs -rm /ClientJavaFolder

To Delete Directory which start with same name and ends with different character.

bin/hadoop dfs -rm /ClientJavaFolder/File*

Ex: If dir ClientJavaFolder have 3 files. If you want to delete all the files which starts with "**File"** – then you need to use above command. i.e. * will be used.

File1.txt
File2.txt
HelloFile.txt

To Rename the File name

bin/hadoop dfs -mv /ClientJavaFolder/HadoopAddFile.txt /ClientJavaFolder/RenamedFile.txt

To View Directory View

bin/hadoop dfs -du /

Ex:
bin/hadoop dfs -du /
Found 2 items
238925722 hdfs://localhost:50000/ClientJavaFolder
119462861 hdfs://localhost:50000/NewFile

To set Access permission

To set permission for File

bin/hadoop dfs -chmod 771 /ClientJavaFolder/NewRenameFile.txt

To set permission for Directory

bin/hadoop dfs -chmod 771 /ClientJavaFolder/

Note: Only Directories will be given execute permission and for all files READ WRITE can be given.

To run JAR file

Use below CMD if the JAR is NOT EXPORTED along with class name

bin/hadoop jar /home/administrator/Print.jar org.samples.mapreduce.training.patientrxMR /NewJavaFolder/NewRenameFile.txt /hello.txt

Format:

bin/hadoop jar [jarpath] [classPath] [inputpath] [outputpath]

Use below CMD if the JAR is EXPORTED along with class name

bin/hadoop jar /home/administrator/Print.jar /NewJavaFolder/NewRenameFile.txt /hello.txt

To run Streaming-API or other program files apart from JAVA.

hadoop jar /home/administrator/hadoop-1.2.1/contrib/streaming/hadoop-*streaming*.jar \-file /home/administrator/hadoop-1.2.1/streaming_api/mapper.py -mapper /home/administrator/hadoop-1.2.1/streaming_api/mapper.py \-file /home/administrator/hadoop-1.2.1/streaming_api/reducer.py -reducer /home/administrator/hadoop-1.2.1/streaming_api/reducer.py \-input /my30Mb_file -output /my30Mb_file_output_1

Format:

bin/hadoop jar [streamingjar path]
\-file
[mapper type file format]
-mapper
[mapper type file]
\-file
[reducer type file format]
- reducer
[reducer type file]
-input
[input file path]
-output

[output file path]

Note: Spaces are not allowed in above command for all BACKWARD SLASH
Ex: **\ -file is wrong**
\-file is corrent
The whole comment should be ONE LINE command

Generic Parserer Used via PRE-DEFINED Command (Using LOCAL HOST method)

hadoop jar /home/administrator/genericparserfiles.jar -files hdfs://localhost:50000/GenericParsers/input2 /file /GenericParsers/OutputwithFiles2

Generic Parserer Used via HADOOP File Command

hadoop jar /home/administrator/genericparser.jar /hello.xt /GenericParsers/input2 /GenericParsers/Output3

Hadoop Safe Mode Command

hadoop dfsadmin -safemode leave

HIVE

Without Partition;

```
create table patient(pid INT,pname STRING,drug STRING,gender STRING,tot_amt INT)
row format delimited fields terminated by ',' stored as textfile;
```

WithPartition;

```
create table country(cid INT, name STRING,cntry STRING, joindate STRING) partitioned
by(country STRING, jdate STRING) row format delimited fields terminated by ',' stored as
textfile;
```

To load data in HIVE;

From HDFS:

```
load data inpath '/my30Mb_file' into table patient;
```

From Local

```
load data local inpath '/home/administrator/Desktop/US' into table patient;
```

Partitioned by:

```
load data local inpath '/home/administrator/Desktop/US' into table country partition
(country='US',jdate='01-05-2010');
```

Note: All partition variables must be used when trying to load data. Ex country and jdate are partition variables. If we try to use country along, it will not work. We need to use all varaibles.

To write output in HDFS

To HDFS:

```
insert overwrite  directory '/hive/output' select * from patient;
```

To Local

```
insert overwrite local directory '/home/administrator/h' select * from patient;
```

(Use above query when you need only one column to be stored. For more columns storage, see below note section)

Note: Format problem will be available if we try to store in local drive. i.e. data will be stored without spaces or any delimiters. We can use below query to put our own delimiter.

```
insert overwrite local directory '/home/administrator/Desktop/Show' row format
delimited fields terminated by ',' select * from country;
```

This delimiter can be any thing irrespective to default delimiter which was given in CREATE statement.

Partitions in HIVE

From HDFS:

```
load data inpath '/home/administrator/Desktop/US' into table country partition
(country='US',jdate='01-05-2010');
```

From Local

```
load data local inpath '/home/administrator/Desktop/US' into table country partition
(country='US',jdate='01-05-2010');
```

To Run User Defined Functions

To Add Jar:

```
add jar file:///home/administrator/hive.jar
```

To Add Jar:

```
create temporary function myfn as 'org.samples.hive.training.hiveUDAF';
```

To start HWI service

```
bin/hive --service hwi
```

Note: Hive will allow only one instance of HWI and in mulitple terminals if you try to execute this query. You will end up with error.

To Start Hive Server

```
bin/hive --service hiveserver
```

Buckets:

```
create table bucket (cid INT, cname STRING) CLUSTERED BY(cid) 3 buckets row format
delimited fields terminated by ',' stored as textfile;
```

Note: When you try to load the same data again to same directory, it will create separate instance. Whereas in normal method without bucketing, it will replace with the loaded data.

To load data in Bucket:

```
insert overwrite table bkt_table_name select * from table_name;
```

HBASE

To Start Hbase (Present Working Directory should be Hbase Folder)

bin/start-hbase.sh

If you are inside BIN folder of HBase Folder

./start-hbase .sh

To Start Hbase (Present Working Directory should be Hbase Folder)

bin/stop-hbase .sh

If you are inside BIN folder of Hbase Folder

./stop-hbase .sh

To enter HBASE Prompt

bin/hbase shell

To show tables use below comment

list

To create table

create 'mercury','Personal','Medical'

Note: Here QUERY –
mercury is TABLE name
Personal & Medical – Family

Below are some other methods to create table

1. create 'table1', {NAME => 'f1'}
    1. NAME Attirbute contributes as Family.
    2. This follows KEY VALUE Pair method.
    3. This attribute name must be in UPPER CASE.
2. create 't1', 'f1', {SPLITS => ['10', '20', '30', '40']}

To insert in table

put 'mercury','001','Personal:Name','Bala'

Note:

1. By Default, we will have 3 versions of data. It is configurable via code itself. Below example shows how versions will be changed.
    create 'sam1', {NAME => 'f1'},{NAME =>'f2',VERSIONS => 5}
2. If the table is disabled, put statement will not work. To find whether the table is enabled or disabled. Use below command
    is_disabled 'table_name'

To retrive values

Using GET Method

get 'mercury','001'

To get specific column

Ex: get 'mercury','001' ,'Personal'

Ex: get 'mercury','001' ,'Personal:Age'

To get all columns available in table1

scan 'mercury'

Note: This will display latest version. To display all versions of data, use below command.

To get versions of tables

scan 'mercury',{VERSIONS =>3}

To delete values

delete 'dummy','002','f1:name' (specific family wise)

delete 'dummy','002' (specific row key wise)

To drop table

drop 'mercury'

Note: To delete we need to disble the table. Use **disable 'mercury'** to disable the table.

Enable/Disable/Drop/Exist

Description	Command & Desc	
	Cmd	Description
Enable Table	enable 'table_name'	Only after enabling the table, it is allowed to write data.
To Enable more than one table	enable_all 'tab.*'	
To check whether the table is disabled	is_enabled 'table_name'	Returns True if it is enabled else False.
Disable Table	disable 'table_name'	Data Insertion **NOT POSSIBLE**
To Disable more than one table	disable_all 'tab.*'	
To check whether the table is disabled	is_disabled 'table_name'	Returns True if it is disabled else False.
To check whether table exist in system	exists 'table_name'	Returns True if table exist else False.
To drop all tables	drop_all 'tab.*'	Note: All the table must be disabled.

PIG

To Start PIG Service

To load data from HDFS

bin/pig

To load data from Local System

bin/pig -x local

Note:

1. If the mode is HDFS,
   1. The output will be available in /tmp directory
2. If the mode is LOCAL,
   1. Output will be available in /tmp folder. (click COMPUTER link in left pane, to see this folder.

To Create & Load Data without Schema;

Login as

bin/pig -x local

To Load Data

```
A = load '/data10' using PigStorage(',');
A = load '/data10' using PigStorage(',') as (pid:int, pname:chararray, drug:chararray,
gender:chararray, amt:int);
```

To Filter Data

```
B = filter A by $2 =='avil';
```

To Create & Load Data with Schema(via HDFS);

Login as

bin/pig

To Load Data

```
C= load '/data10' using PigStorage(',') as (pid:int, pname:chararray,
drug:chararray, gender:chararray, tot_amt:int);
```

To Filter Data

```
D = filter C by drug=='avil'; (we use column values rather than $ value.)
X = filter C by (f1==8) OR (NOT (f2+f3 > f1));
```

To Fetch Specific Column Data

```
A = load '/data10' using PigStorage(',') as (pid:int, pname:chararray, drug:chararray,
gender:chararray, amt:int);
B = foreach A generate pid,drug;   (using column values.)
C = foreach A generate $0,$3;      (using $ variable.)
```

To Display Data

```
dump A
```

To Describe Bag

```
illustrate A
```

To take Procedural Plan Data

```
explain A (procedureal Plan will be displayed.)
```

To Display Data

```
illustrate A
```

To Store Data in HDFS

```
Store F into '/Pig_Results' using PigStorage(',');
```

To Use Group Fn

```
G = GROUP F by drug;
sm = foreach G generate group,SUM(C.tot_amt) as S;
```

To Use CoGroup Fn

```
A = LOAD '/data10' using PigStorage(',');
B = LOAD '/drug' using PigStorage(',');
C = COGROUP A by $2, B by $1;
```

To use NOT NULL Condition:

```
B = filter A by $2 is not null;
```

To Use Distinct

```
A = LOAD '/tuple2' using PigStorage(',') as (f1:int, f2:int, f3:int);
B = GROUP A by f1;
C = foreach B generate group,SUM(A.f2) as S;
dump C
```

Split Fn

```
A = LOAD '/patientHR-10' using PigStorage(',') ;
SPLIT A into x if $0%2==0, z if $0==1;
dump x
dump y
```

Note: This can be used to do mulitple operations at a single query. Like Male records in one Variable and Female records in another varaible.

To Use MACRO

```
DEFINE myMacro1(fTable,fCol,fValue) returns returnVariable
{ $returnVariable = FILTER $fTable BY $fCol == '$fValue';};
```

To Run the Macro

```
D = myMacro1(C,'drug','metacin');
```

To Use Joins

```
A = load '/data10' using PigStorage(',');
B = load '/drug' using PigStorage(',');
C = join A by $2, B by $1;
```

To Use Inner/outer/left/right joins use below query

```
C = join A by $2 left outer, B by $1;
```

(left outer/right outer//full outer)

Project-Range Expressions (..) is used.

```
A = LOAD '/data10' using PigStorage(',');

B = foreach A generate $0..$4;
```

Note: .. refers from $0th Column to $4th Column, the collumn

To Sort Data

```
A = LOAD '/data10' using PigStorage(',');
B = foreach A generate $0..$4;
C = order B by $4;
C = order B by $4..$6;
C = order B by $4..$6,$7;
```

To run jar File

```
register /home/administrator/myFile.jar
B = foreach C generate com.Upper(pname);
```

Note: A, B, C & D are temporary variables to handle the table.

To Execute jar file instead of REGISTER method

```
A = LOAD '/WordcountInput.txt';
B = MAPREDUCE 'wordcount.jar' STORE A INTO 'inputDir' LOAD 'outputDir'
AS (word:chararray, count: int) `org.myorg.WordCo
```

Tuples/Bag/Maps

Tuple:

```
A = LOAD '/tuple' using PigStorage(' ') AS (t1:tuple(t1a:int,
t1b:int,t1c:int),t2:tuple(t2a:int,t2b:int,t2c:int));
```

(Refer **PIG.txt**, search above query for more explaination)

To Use Tuple

```
B = FOREACH A GENERATE t1.t1a,t2.t2a;
```

Bag:

```
A = LOAD '/tuple' using PigStorage(' ') AS (t1:tuple(t1a:int,
t1b:int,t1c:int),t2:tuple(t2a:int,t2b:int,t2c:int));
```

(Refer **PIG.txt**, search above query for more explaination)

To Use Tuple

```
B = FOREACH A GENERATE t1.t1a,t2.t2a;
```

Conventions of TUPLE/BAG/MAP:

(1,2,3) is tuple
{1,2} is bag
{ (1,3),(3,4) } is tuple inside bag
(123,10,{ (1,3),(3,4) })is bag inside tuple
[key#value] is MAP
Ex: ['name' # 'John', 'ext' # 5555]

Bag:

```
A = LOAD '/tuple' using PigStorage(' ') AS (t1:tuple(t1a:int,
t1b:int,t1c:int),t2:tuple(t2a:int,t2b:int,t2c:int));
```

SQOOP

To check SQOOP & DB Connectivitiy

```
bin/sqoop
eval --connect jdbc:mysql://localhost/test
        -username root
        -password root
--query
        "select * from patient";
```

To List Databases

```
bin/sqoop list-databases --connect jdbc:mysql://localhost/information_schema -username
root -password root
```

To Move Data from RDBMS to HDFS (Default Path: /user/username/table_name)

```
bin/sqoop
        import
                --connect jdbc:mysql://localhost/test
                -username root
                -password root
        --table
                patient
        -m 1
```

-m – Mapper

To Move Data from RDBMS to Customized HDFS Directory

```
bin/sqoop
        import
                --connect jdbc:mysql://localhost/test
                -username root
                -password root
        --table patient
        --split-by pid
        -m 1  OR –num-mappers          (-num-mappers can be used instead of -m)
        --target-dir /Sqoop            (to store in HDFS dir)
        --where  “pid < 10”            (to use where clause condiitons)
```

Commands	Descriptions
--query	This command is used to accept the sql query. If you use this command, --table should not be used.
--columns “<col>”	If you need to fetch specific columns of table. Use this command. Ex: --columns “col1,col2,col3”
--num-mappers	It is as same as like -m option which we will provide to introduce how many mappers will be running.
--warehouse-dir	As same like –target-dir, But the difference is – it will create a directory of

<table>
<tr><td></td><td>TABLE name and will be storing all details inside the newly created dir.

Ex: Consider PatientMR is the table which we are going to store in HDFS

<table>
<tr><th>Command</th><th>Input Path</th><th>HDFS Directory</th></tr>
<tr><td>--target-dir</td><td>/My_Target</td><td>/My_Target</td></tr>
<tr><td>--warehouse-dir</td><td>/My_Target</td><td>/My_Target/PatientMR</td></tr>
</table>
</td></tr>
<tr><td>--where “condition”</td><td>To give condiitons to filter the required records.</td></tr>
<tr><td>--append</td><td>This command is used to append the records in same directory. If you do NOT use this command, you cannot write the data in same path. If you use this command, the data will written in same path with different file name and NOT OVERWRITTING the data.

If the destination directory already exists in HDFS, Sqoop will refuse to import and overwrite that directory’s contents. If you use the --append argument, Sqoop will import data to a temporary directory and then rename the files into the normal target directory which will not disturb existing directory files.</td></tr>
<tr><td>-z or -compress</td><td>To compress the data and store it in HDFS as .gz file format.</td></tr>
<tr><td>--fields-terminated-by [character]</td><td>To store in HDFS with field needs to be delimiter.
--fields-terminated-by \\t - Tab Space
--fields-terminated-by \\b – Back Space
\\n – newline\

You can use any character you need.
This will take only first character even if you give as word.
Ex: --fields-terminated-by XYZ</td></tr>
<tr><td>--hadoop-home</td><td>To Override Hadoop Home</td></tr>
<tr><td>Generic Options</td><td>Like Hadoop Commands, you can use generic options in SQOOP.

<table>
<tr><td>-D <property=value></td><td>use value for given property</td></tr>
<tr><td>-fs <local|namenode:port></td><td>specify a namenode</td></tr>
<tr><td>-jt <local|jobtracker:port></td><td>specify a job tracker</td></tr>
<tr><td>-files <comma separated list of files></td><td>specify comma separated files to be copied to the map reduce cluster</td></tr>
<tr><td>-libjars <comma separated list of jars></td><td>specify comma separated jar files to include in the classpath.</td></tr>
<tr><td>-archives <comma separated list of archives></td><td>specify comma separated archives to be unarchived on the compute</td></tr>
</table>
</td></tr>
</table>

	machines.
-conf <configuration file>	specify an application configuration file

To Import All Tables

```
bin/sqoop
        import-all-tables
        --connect jdbc:mysql://localhost/test
        -username root
        -password root
        -m 1;
```

To Import Data to HIVE

```
bin/sqoop
        import
                --connect jdbc:mysql://localhost/test
        -username root
        -password root
--table patient
--hive-table patient_hive
--create-hive-table
--hive-import -m 1;
```

Import Commands

Commands	Descriptions
--input-escaped-by <char>	Sets the input escape character
--input-fields-terminated-by <char>	Sets the input field separator
--input-lines-terminated-by <char>	Sets the input end-of-line character
--input-optionally-enclosed-by <char>	Sets a field enclosing character

To Import Data to HBASE

```
bin/sqoop import
        --connect jdbc:mysql://localhost/test
        -username root
        -password root
                --table patient
                        --hbase-table patientsqp
                                --column-family MyFamily
                                --hbase-row-key pid
                                --hbase-create-table  -m 1
```

To Export Data from HDFS to SQL

```
bin/sqoop
        export
```

```
--connect jdbc:mysql://localhost/test -username root -password root
--table patient1
--export-dir /Sqoop/AsTextFile/patient/part-m-00000
-m 1;
```

Export & Other Commands

Commands	Descriptions
--update-mode allowinsert	By adding this command, allows user to insert the new records which are not available in
--update-key <column name>	This is used to update the table based on column name that is given.
--verbose	Print more information while working

To Create Scoop JOB

```
bin/sqoop job
        --create myjob
        -- import
                –connect jdbc:mysql://localhost/test
                -username root -password root
        --table patient
        -m 1 --target-dir /Sqoop/MRJobLastValue
--last-value pid
```

Commands	Descriptions
bin/sqoop job --list	To display all the Sqoop JOB that are created.
bin/sqoop job –exec <jobname>	To execute the job created.

SQOOP Main Commands

codegen	**Generate code to interact with database records**
create-hive-table	**Import a table definition into Hive**
eval	**Evaluate a SQL statement and display the results**
export	**Export an HDFS directory to a database table**
help	**List available commands**
import	**Import a table from a database to HDFS**
import-all-tables	**Import tables from a database to HDFS**
job	**Work with saved jobs**
list-databases	**List available databases on a server**
list-tables	**List available tables in a database**
merge	**Merge results of incremental imports**
metastore	**Run a standalone Sqoop metastore**
version	**Display version information**

FLUME

To RUN Flume

bin/flume-ng
agent
--conf-file netcat_flume.conf
--name a1
-Dflume.root.logger=INFO,console

--conf-file – To specify the file name we need to run in FLUME. **--name** -- To specific the agent name. There are more than one agent can be available in conf file. Thats why we specify Agent Name.

Sources ***-mandatory attributes** **type, bind, port, interceptor**	
Attributes	**Description**
type*	What type of transfer method the flume uses is mentioned. Ex: **a1.sources.r1.type = netcat (Terminal based)** **a1.sources.r1.type = exec (Log File based)** **To mention Log File, below code will be used. Also need to give batchsize.** **a1.sources.r1.command = tail - F /home/administration/data.log** **a1.sources.r1.batchSize = 2** (Batchsize is max no of lines to read and send to the channel at a time) **a1.sources.r1.type = spooldir (Directory based)**
bind*	IP Address of Source machine or if the local machine is used. Then below method is followed. **a1.sources.r1.bind = localhost**
port*	Port number is mentioned here.
Interceptors	Interceptors will be intermediater between Source and Channels.We need to specify type and header of the interceptor. **a1.sources.r1.interceptors = i1** **a1.sources.r1.interceptors.i1.type = host** **a1.sources.r1.interceptors.i1.hostheader = hostname**
TBD	

Multi Node Cluster Setup

Steps for Multi Node Cluster

1. Choose number of nodes in the cluster and assign the NN, SNN, JT, DN, TT respectively.
2. Generate SSH Key in NN. Copy the ssh key to all DN.
    1. ssh-keygen -t rsa
    2. Go to .ssh directory
    3. cat id_rsa.pub >> authorized_keys
    4. Now use below command to copy into DN.
        1. Ssh-copy-id -i /home/adminstrator/.ssh/id_rsa.pub systemname/username
3. In all nodes, specify NN's IP Address in Master file and DN's IP Address in Slave file which is avialable in below path.
    1. Go to directory of HDFS. Say hadoop-1.2.1/conf. File called masters and slaves will be avaible.
    2. Masters file will have default value localhost. We need to give IP address of NN.
    3. Masters file will have default value localhost. We need to give IP address of DN.
4. In DN, open core-site.xml which is inside conf folder. Configure its NN value as NN Node IP address and port number to access NN.
5. In DN, open mapred-site.xmlwhich is inside conf folder. Configure its JT value as JP Node IP address and port number to access NN.
6. In DN, open hdfs-site.xml which is inside conf folder.
    1. Set name dir, data dir, block size, replication factor.
7. Start Cluster and check whether the cluster is up.

Hadoop Admin Access Details

When do we need to go towards HADOOP?

If we have High Scalability & High Computation, then we need to implement HADOOP.

Steps for Admin

1. Design the Cluster
    1. How many DN nodes needs to be used. Replication factors and others.
2. DesignHardware configuration
    1. This should be in ratio of 1:2:4 i.e. **1 TB data : 2 Quad Core Processor : 4 GB RAM**
    2. RAM Size should be LARGE.
    3. RAID Hard Drives will be used.
3. OS Selection
    1. Ubuntu, Fedora Core, CentOS, RHEL and Solaris.
4. Java Version needs to selected.
    1. Hadoop accepts all versions of JAVA expect Version 7 and Version 6 - 1.6.0u18.
5. Configure Hadoop details and Check the health of the cluster. Run below commands for checking.
    1. bin/hadoop fsck /
    2. bin/hadoop fsck / -files -blocks -racks

Adding a node in LIVE Cluster.

1. In hdfs-site.xml and mapred-site.xml below changes needs to be made.
    1. Below configuration needs to be done. i.e. setting include file

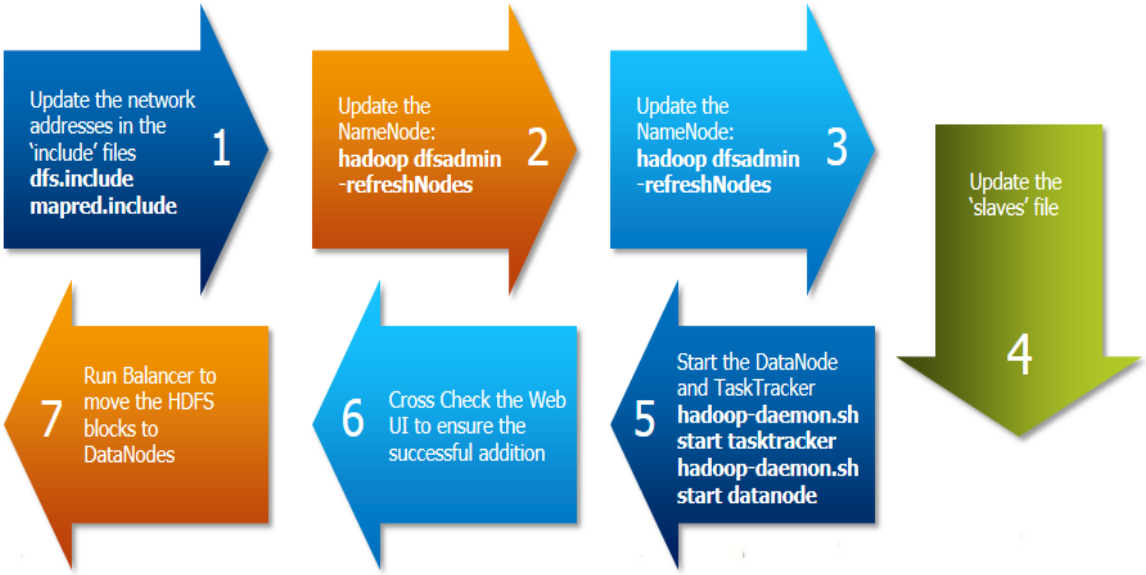
hdfs-site.xml

```
<property>
    <name>dfs.hosts.include </name>
    <value>/home/hadoop/includes</value>
    <final>true</final>
    </property>
```

mapred-site.xml

```
<property>
    <name>mapred.hosts.include </name>
    <value>/home/hadoop/includes</value>
    <final>true</final>
    </property>
```

    2. Need to Update Name Node:
        1. **bin/hadoop dfsadmin -refreshNodes** in DN machine.
        2. Now in Commisionning Link, we will get the DN related details.
        3. Again run same command to finaize.
    3. In slaves file, update the new node IP address.
    4. Start DN and TT in newly added node.
        1. hadoop-daemon.sh start tasktracker
        2. hadoop-daemon.sh start datanode
    5. Cross check UI.
    6. Run Balancer using this command start-balancer.sh
    7. Refer below image



1. **Deleting a node in LIVE Cluster.**
    1. In hdfs-site.xml and mapred-site.xml below changes needs to be made.

2. Below configuration needs to be done. i.e. setting exclude file

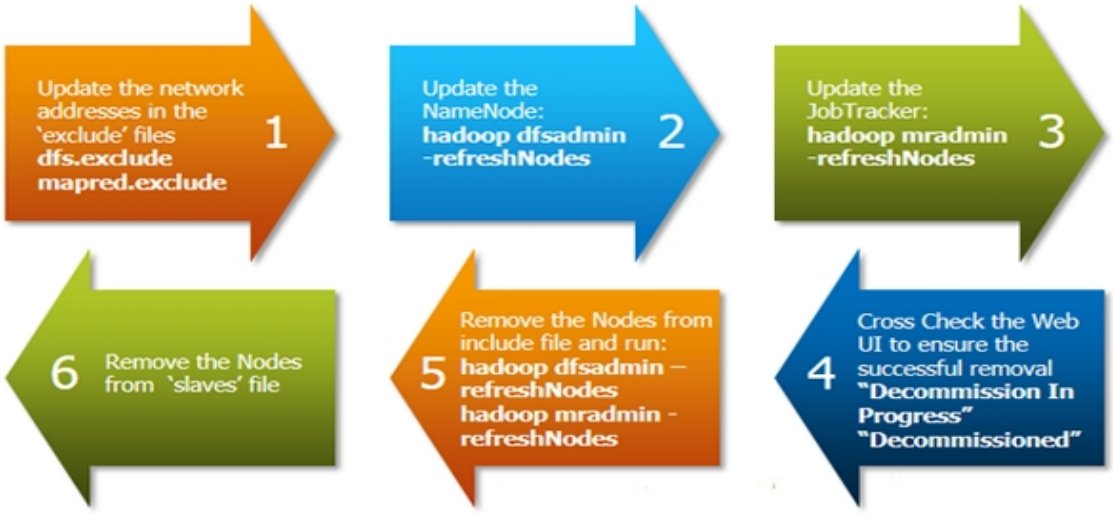
hdfs-site.xml

```
<property>
<name>dfs.hosts.exclude </name>
<value>/home/hadoop/excludes</value>
<final>true</final>
</property>
```

mapred-site.xml

```
<property>
<name>mapred.hosts. exclude </name>
<value>/home/hadoop/excludes</value>
<final>true</final>
</property>
```

3. Need to Update Name Node:
    1. **bin/hadoop dfsadmin -refreshNodes** in DN machine.
4. In slaves file, update the new node IP address.
5. Start DN and TT in newly added node.
    1. hadoop-daemon.sh stop tasktracker
    2. hadoop-daemon.sh stop datanode
6. Cross check UI.
7. Run Balancer using this command **start-balancer.sh**
8. Refer below image



Port Numbers

Daemon Process	Web Port	RPC Port
Namenode	50070	8020 (50000)
Secondarynamenode	50090	-
Data Node	50075	50010
Job Tracker	50030	8021(50001)
Task Tracker	50060	50020
HMaster	60000	60010
HRegion Server	60010	60030
HQuorumPeer	2181	
Sqoop Metastore	16000	

If you have any concerns or need some changes in this doc, please contact us via below mail address.

ballicon@gmail.com
rbharath4u@gmail.com
saikiran.isk@gmail.com
email2anandraj@gmail.com
silambarasan909@gmail.com