Purpose

- This guide describes the setup of a standalone HBase instance running against the local filesystem/HDFS.
- This is not an appropriate configuration for a production instance of HBase, but will allow you to experiment with HBase.
- This section shows you how to create a table in HBase using the *hbase shell* CLI, insert rows into the table, perform put and scan operations against the table, enable or disable the table, and start and stop HBase.

Required Software

- Java™ HBase requires that a JDK be installed.
  http://hbase.apache.org/book.html#java

- Choose a download site from this list of Apache Download Mirrors.
  - Click on the suggested top link.
  - Click on the folder named *stable* and then download the binary file that ends in *.tar.gz* to your local filesystem.
  - Be sure to choose the version that corresponds with the version of Hadoop you are likely to use later. (*hbase-0.98.3-hadoop2-bin.tar.gz*.)

Prepare to Start the Hadoop Cluster (Cont.)

- Like HDFS, HBase has also 3 running mode:
  - **Standalone HBase** (Setup of a standalone HBase instance running against the local filesystem)
    - In standalone mode HBase runs all daemons within this **single JVM**, i.e. the HMaster, a single HRegionServer, and the ZooKeeper daemon.
    - Using HBase with a local filesystem does not guarantee *durability*.
    - Prior to HBase 0.94.x, HBase expected the loopback IP address to be 127.0.0.1. Ubuntu and some other distributions default to 127.0.1.1 and this will cause problems for you.
  - **Pseudo-Distributed Local Install**
    - HBase still runs completely on a single host, but each HBase daemon (HMaster, HRegionServer, and Zookeeper) runs as a **separate process**.
  - **Fully Distributed**

Prepare to Start the HBase-Standalone

- Unpack the downloaded HBase distribution. In the distribution, edit the file *conf/hbase-env.sh*, uncomment the line starting with JAVA_HOME, and set it to the appropriate location for your operating system:

```
# set to the root of your Java installation
  export JAVA_HOME=/usr/lib/jvm/jdk1.7.0
```

Prepare to Start the HBase-Standalone (Cont.)

- Edit *conf/hbase-site.xml*, which is the main HBase configuration file.
  - At this time, you only need to specify the directory on the local filesystem where HBase and ZooKeeper write data.
  - By default, a new directory is created under /tmp.

```
<configuration>
  <property>
    <name>hbase.rootdir</name>
    <value>file:///home/testuser/hbase</value>
  </property>
  <property>
    <name>hbase.zookeeper.property.dataDir</name>
    <value>/home/testuser/zookeeper</value>
  </property>
</configuration>
```

Prepare to Start the HBase-Standalone (Cont.)

- The *bin/start-hbase.sh* script is provided as a convenient way to start HBase.
  - Use the **jps** command to verify that you have one running process called Hmaster.
  - *Remember that in standalone mode HBase runs all daemons within this single JVM, i.e. the HMaster, a single HRegionServer, and the ZooKeeper daemon.*

Pseudo-Distributed Configuration

- You can re-configure HBase to run in pseudo-distributed mode:
  - Stop HBase if it is running.
  - Configure HBase:
    - Edit the *hbase-site.xml* configuration.
    - This directs HBase to run in distributed mode, with one JVM instance per daemon.
    - Next, change the hbase.rootdir from the local filesystem to the address of your HDFS instance, using the hdfs:// URI syntax.

```
<property>
 <name>hbase.cluster.distributed</name>
 <value>true</value>
</property>
```

```
<property>
 <name>hbase.rootdir</name>
 <value>hdfs://localhost:8020/hbase</value>
</property>
```

Lab Assignment

1. Start HBase daemon;
2. Start HBase shell;
3. Create a Book table ...
4. Add information to Book table ...
5. Count the number of rows ...
6. Retrieve an entire record with ID 1;
7. Only retrieve title and description for record with ID 3
8. Change a record ...
9. Display all the records to the screen
10. Display title and author's last name for all the records
11. Display title and description for the first 2 records
12. Explore HBase Web-based management console ...
13. Check the detailed status of your cluster via HBase shell
14. Delete a record ...
15. Drop the table Book.

1- Start HBase daemon

```
start-hbase.sh
```

2- Start HBase shell

```
hbase shell
```

3- Create a table called Book whose schema will able to house book's title, description, author's first and last names. Book's title and description should be grouped as they will saved retrieved together. Author's first and last name should also be grouped.
(hint: since title and description need to be grouped together and so do author's first and last name, it would be wise to place them into 2 families such as info and author. Then title and description will become columns of info family and first and last columns of author family).

```
create 'Book', {NAME=>'info'}, {NAME=>'author'}
```

4- Add the following information to Book table:

ID	Title	Description	First name	Last name
1	Faster than the speed love	Long book about love	Brian	Dog
2	Long day	Story about Monday	Emily	Blue
3	Flying Car	Novel about airplanes	Phil	High

```
put 'Book', '1', 'info:title', 'Faster than the speed love'
put 'Book', '1', 'info:description', 'Long book about love'
put 'Book', '1', 'author:first', 'Brian'
put 'Book', '1', 'author:last', 'Dog'
put 'Book', '2', 'info:title', 'Long day'
put 'Book', '2', 'info:description', 'Story about Monday'
put 'Book', '2', 'author:first', 'Emily'
put 'Book', '2', 'author:last', 'Blue'
put 'Book', '3', 'info:title', 'Flying Car'
put 'Book', '3', 'info:description', 'Novel about airplanes'
put 'Book', '3', 'author:first', 'Phil'
put 'Book', '3', 'author:last', 'High'
```

5- Count the number of rows. Make sure that every row is printed to the screen as it being counted.

```
count 'Book', INTERVAL => 1
```

6- Retrieve an entire record with ID 1

```
get 'Book', '1'
```

7- Only retrieve title and description for record with ID 3.

```
get 'Book', '3', {COLUMNS=>['info:title', 'info:description']}
```

8- Change the last name of an author for the record with title *Long Day* to *Happy*.

- Display the record on the screen to verify the change.
- Display both new and old value. You should be able to see both Blue and Happy. Why is that?

```
put 'Book', '2', 'author:last', 'Happy'
# to verify select the record
get 'Book', '2', {COLUMNS=>'author:last'}
# to display both versions
get 'Book', '2', {COLUMNS=>'author:last', VERSIONS=>3}get 'Book', '3',
{COLUMNS=>['info:title', 'info:description']}
```

9- Display all the records to the screen.

```
scan 'Book'
```

10- Display title and author's last name for all the records.

```
scan 'Book', {COLUMNS=>['info:title', 'author:last']}
```

11- Display title and description for the first 2 records.

```
scan 'Book', {COLUMNS=>['info:title','info:description'], LIMIT=>'2'}
or
scan 'Book', {COLUMNS=>['info:title','info:description'],
STOPROW=>'3'}
```

12- Explore HBase Web-based management console, try and learn as much as you can about your new table.

Book table is hosted via 1 Region Server and there is only 1 Region. There are no start or end keys for that region because there is only 1 region. It has 2 families info and author. There is no compression set for both families, and replication is set to 3.

13- Check the detailed status of your cluster via HBase shell.

```
status 'detailed'
```

14- Delete a record whose title is *Flying Car*, and validate the record was deleted by scanning all the records or by attempting to select the record.
delete 'Book', '3', 'info:title'

```
delete 'Book', '3', 'info:title'
delete 'Book', '3', 'info:description'
delete 'Book', '3', 'author:first'
delete 'Book', '3', 'author:last'
```

15- Drop the table Book.

```
disable 'Book'
drop 'Book'
```