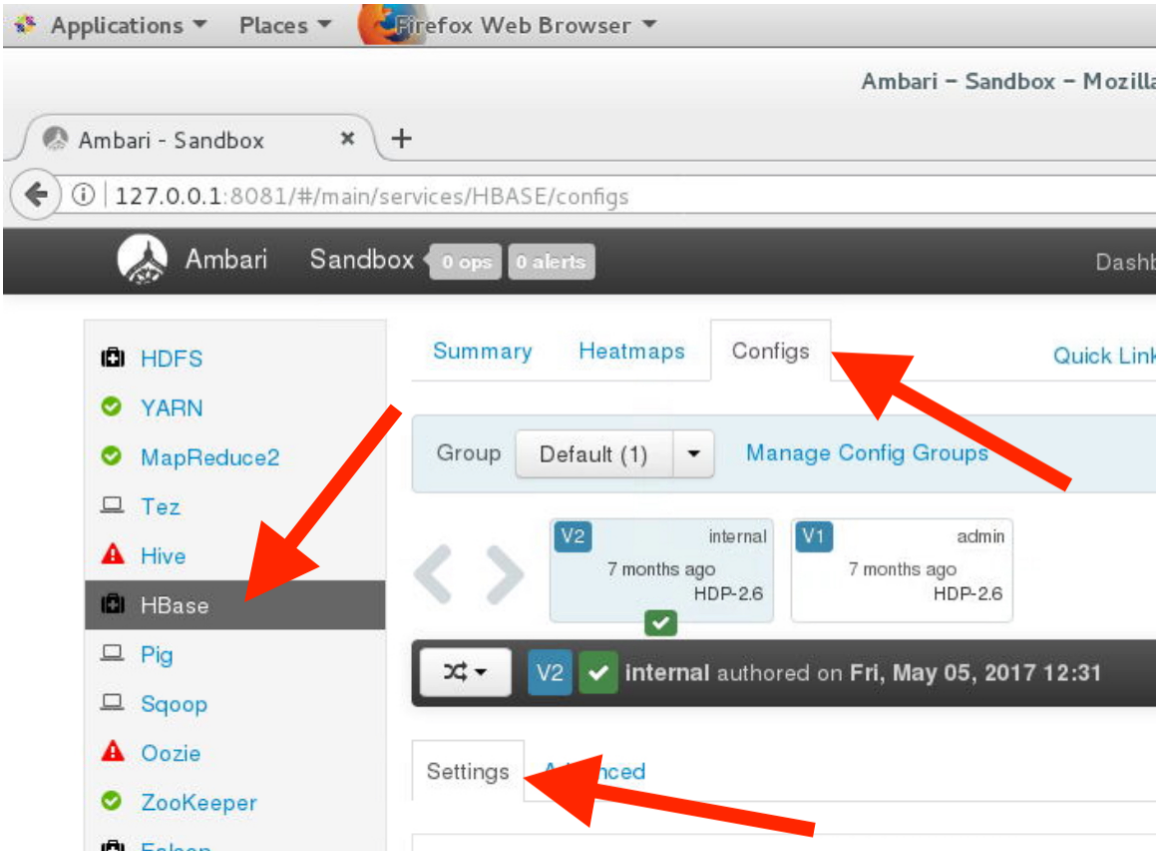
Lab: Getting started with Apache Phoenix

About This Lab

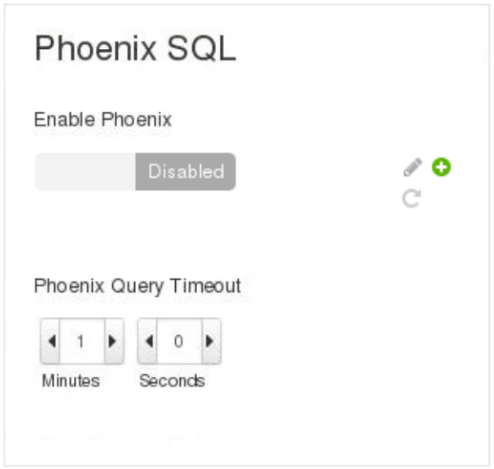
Objective:	To Connect and upsert data into HBase via Phoenix psql utility.
File locations:	N/A
Successful outcome:	You will: Connect to your lab environment; log in and verify HBase access and then use Phoenix psql utility to load data from sql, csv files into HBase.
Before you begin:	Start and connect to your classroom lab environment
Related lesson:	Apache Phoenix Architecture

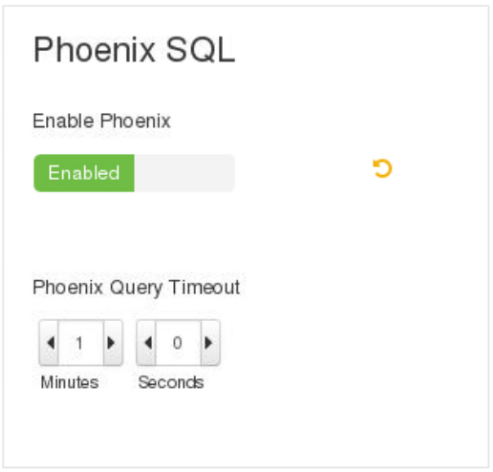
Install Apache Phoenix

1. Via the Firefox browser within the lab environment, log into Ambari at http://127.0.0.1:8081 using `raj_ops` for the username and `raj_ops` as the password. Once there select the *Configs* tab of the *HBase* service as shown below.



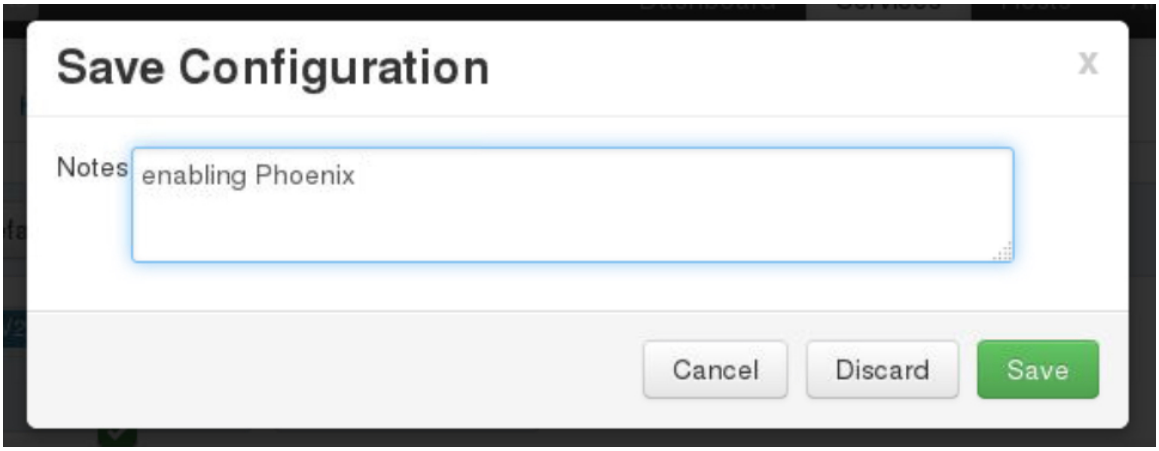
2. Scroll down to the bottom of this web page to find the *Phoenix SQL* visual control as shown on left below. *Enable* the option which will then look like seen on the right below.



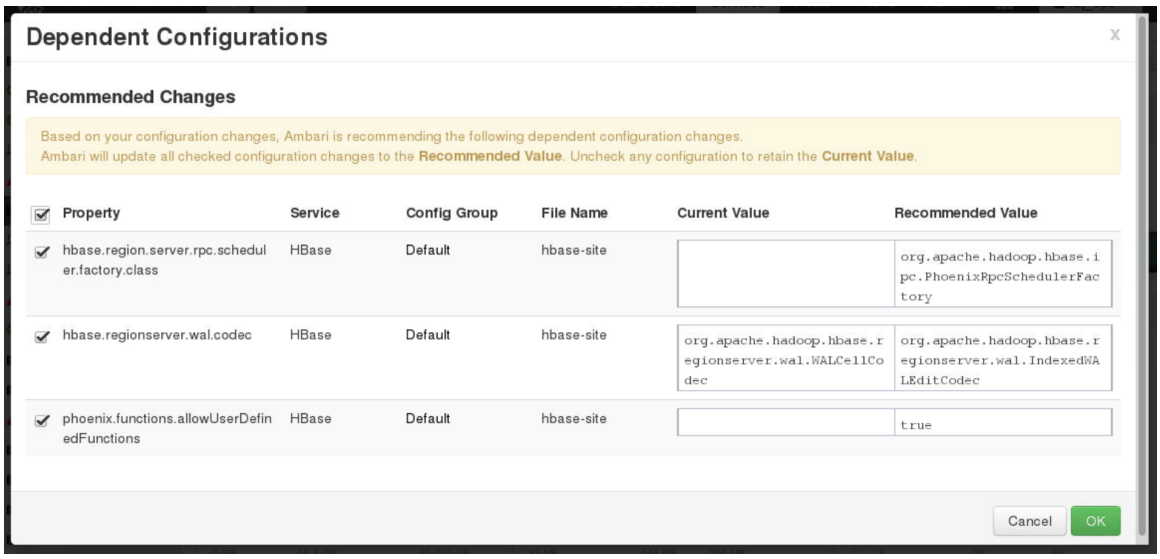


3. Click the green *Save* button.

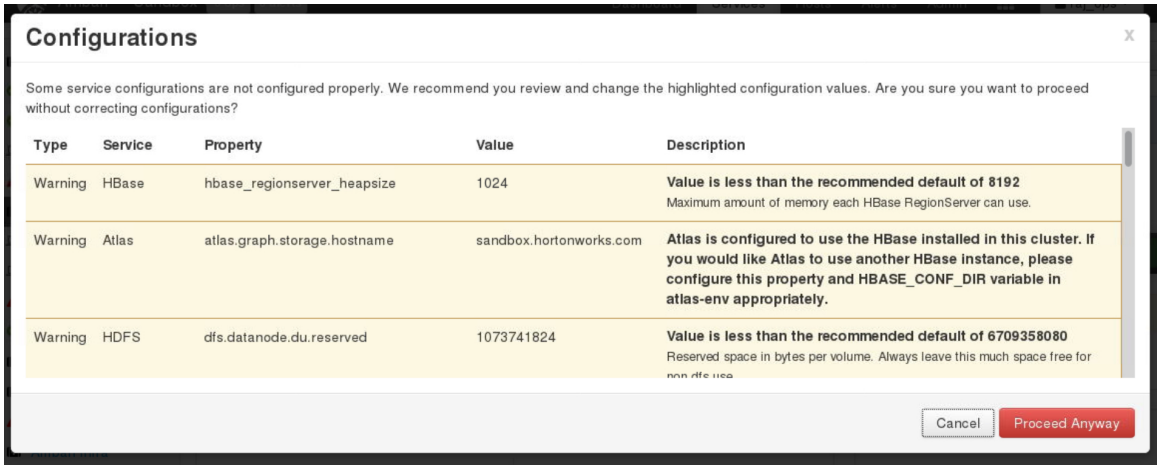
4. Provide appropriate *Notes* and click on *Save* again.



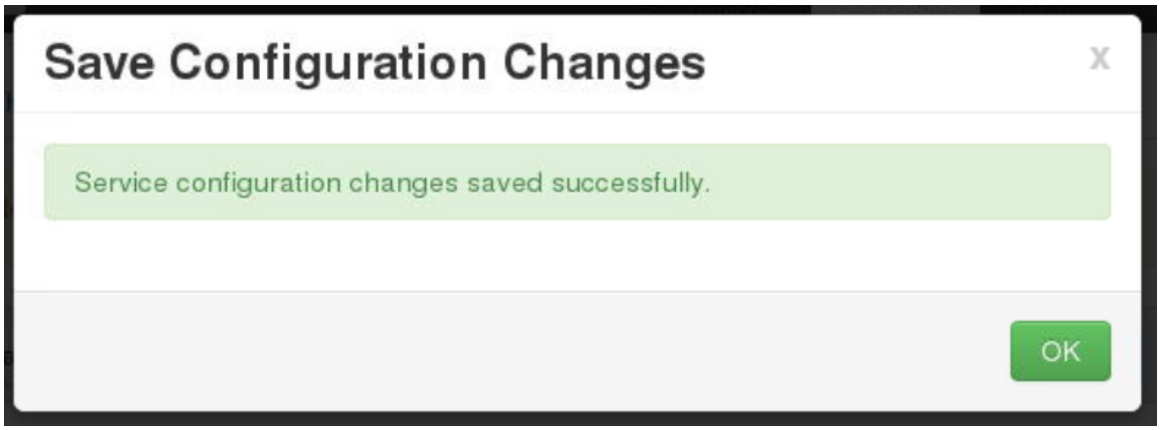
5. Click on the *OK* button when presented with *Dependent Configurations*.



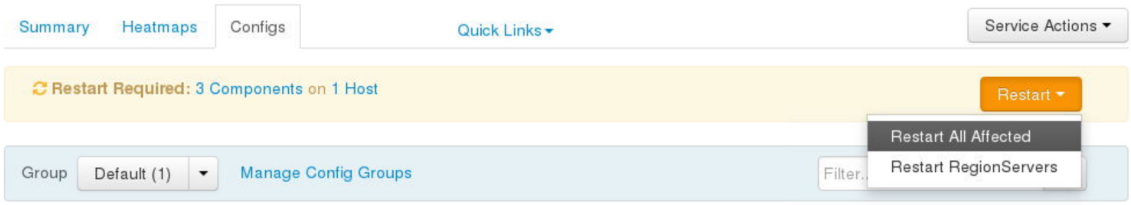
6. Click *Proceed Anyway* in the *Configurations* pop-up.



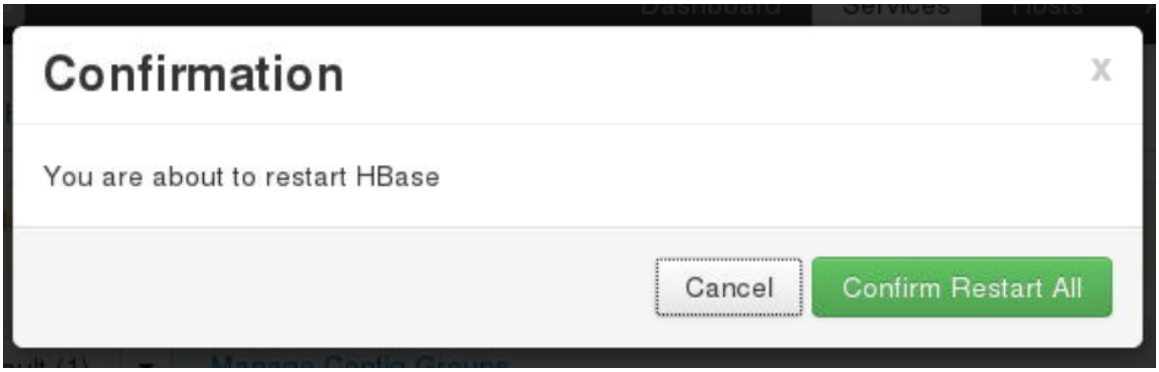
7. Click *OK* to *Save Configuration Changes*.



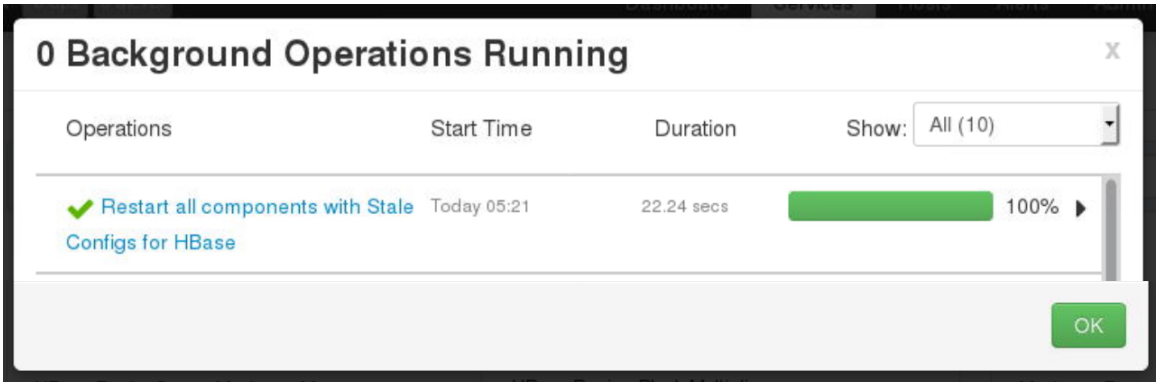
8. *Restart All Affected* components as suggested.



9. *Confirm Restart All*.



10. Wait until the restarts complete and then dismiss the pop-up with the *OK* button.



Upsert Data into HBase via Phoenix with psql

1. If you are not already, log into the sandbox, remain as user `root`, and download needed files.

```
[root@ip-172-30-0-164 ~]# ssh -p 2222 root@127.0.0.1
root@127.0.0.1's password:
Last login: Thu Jun  1 20:58:25 2017 from 172.17.0.1
[root@sandbox ~]# cd rtlabs/datasets/
[root@sandbox datasets]# wget
https://raw.githubusercontent.com/HortonworksUniversity/RealTime_Labs/master/da
tasets/phoenix/WEB_STAT.sql
[root@sandbox datasets]# wget
https://raw.githubusercontent.com/HortonworksUniversity/RealTime_Labs/master/da
tasets/phoenix/WEB_STAT.csv
[root@sandbox datasets]#
```

2. Login to hbase shell and execute the following commands:

 # hbase shell
hbase(main):> list
hbase(main):> exit

Make a note of the list of existing tables which should be similar to those shown below.

```
[root@sandbox ~]$ hbase shell
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.1.2.2.6.0.3-8, r3307790b5a22cf93100cad0951760718dee5dec7, Sat
Apr  1 21:41:47 UTC 2017

hbase(main):001:0> list
TABLE
ATLAS_ENTITY_AUDIT_EVENTS
atlas_titan
iemployee
3 row(s) in 0.1900 seconds

=> ["ATLAS_ENTITY_AUDIT_EVENTS", "atlas_titan", "iemployee"]
hbase(main):002:0> exit
[root@sandbox ~]$
```

3. Execute the below command to BulkLoad a CSV file to Phoenix using psql.py:

```
[root@sandbox datasets]# /usr/hdp/current/phoenix-client/bin/psql.py
localhost:2181:/hbase-unsecure WEB_STAT.sql WEB_STAT.csv

no rows upserted
Time: 1.255 sec(s)

csv columns from database.
CSV Upsert complete. 39 rows upserted
Time: 0.09 sec(s)

[root@sandbox datasets]#
```

4. Now login to **hbase shell** and list the tables again:

```
[root@node1 examples]# hbase shell
2017-07-21 11:43:12,946 INFO  [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.98.4.2.2.4.2-2-hadoop2, rdd8a499345afc1ac49dc5ef212ba64b23abfe110, Tue Mar 31 16:18:12 EDT 2015

hbase(main):001:0> list
TABLE
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/hdp/2.2.4.2-2/hadoop/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/hdp/2.2.4.2-2/zookeeper/lib/slf4j-log4j12-1.6.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SYSTEM.CATALOG
SYSTEM.SEQUENCE
SYSTEM.STATS
WEB_STAT
ambarismoketest
5 row(s) in 1.5490 seconds

=> ["SYSTEM.CATALOG", "SYSTEM.SEQUENCE", "SYSTEM.STATS", "WEB_STAT", "ambarismoketest"]
hbase(main):002:0>
```

Now you can see that the new WEB_STAT table is populated in Hbase.

5. Perform a scan on the table to see the data populated:

   hbase(main):> **scan 'WEB_STAT'**

```
hbase(main):002:0> scan 'WEB_STAT'
ROW                                COLUMN+CELL
 EUApple.com\x00Mac\x00\x80\x00\x01; column=STATS:ACTIVE_VISITOR, timestamp=1500651645278, value=\x80\x00\x00"
 \xF3\xA04\xC8
 EUApple.com\x00Mac\x00\x80\x00\x01; column=USAGE:CORE, timestamp=1500651645278, value=\x80\x00\x00\x00\x00\x00\x00#
 \xF3\xA04\xC8
 EUApple.com\x00Mac\x00\x80\x00\x01; column=USAGE:DB, timestamp=1500651645278, value=\x80\x00\x00\x00\x00\x00\x00\x16
 \xF3\xA04\xC8
 EUApple.com\x00Mac\x00\x80\x00\x01; column=USAGE:_0, timestamp=1500651645278, value=
 \xF3\xA04\xC8
 EUApple.com\x00Store\x00\x80\x00\x0 column=STATS:ACTIVE_VISITOR, timestamp=1500651645278, value=\x80\x00\x00\xAA
 1;\xFD\xEC\xEC\xC8
 EUApple.com\x00Store\x00\x80\x00\x0 column=USAGE:CORE, timestamp=1500651645278, value=\x80\x00\x00\x00\x00\x00\x01Y
 1;\xFD\xEC\xEC\xC8
 EUApple.com\x00Store\x00\x80\x00\x0 column=USAGE:DB, timestamp=1500651645278, value=\x80\x00\x00\x00\x00\x00\x02\xD2
 1;\xFD\xEC\xEC\xC8
 EUApple.com\x00Store\x00\x80\x00\x0 column=USAGE:_0, timestamp=1500651645278, value=
```

6. Exit from the Phoenix client:

```
0: jdbc:phoenix:localhost:2181:/hbase-unsecur> !q
Closing: org.apache.phoenix.jdbc.PhoenixConnection
[root@sandbox datasets]#
```

Result

Apache Phoenix is installed on your Cluster and now it can create tables in Hbase from its interface. Created a table from Phoenix and verified the data from Hbase.