

## Lab: Interactive Queries on Phoenix using sqlline.py

### 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 sqlline utility to query data from Hbase.  
Before you begin: Start and connect to your classroom lab environment, executed lab 1  
Related lesson: Phoenix Data Model

### Query Phoenix using sqlline.py

1. login to Phoenix shell using ./sqlline.py

```
# /usr/hdp/current/phoenix-client/bin/sqlline.py localhost:2181:/hbase-unsecure
```

```
[root@sandbox datasets]# /usr/hdp/current/phoenix-client/bin/sqlline.py
localhost:2181:/hbase-unsecure
Setting property: [incremental, false]
Setting property: [isolation, TRANSACTION_READ_COMMITTED]
issuing: !connect jdbc:phoenix:localhost:2181:/hbase-unsecure none none
org.apache.phoenix.jdbc.PhoenixDriver
Connecting to jdbc:phoenix:localhost:2181:/hbase-unsecure
Connected to: Phoenix (version 4.7)
Driver: PhoenixEmbeddedDriver (version 4.7)
Autocommit status: true
Transaction isolation: TRANSACTION_READ_COMMITTED
Building list of tables and columns for tab-completion (set fastconnect to true
to skip)...
91/91 (100%) Done
Done
sqlline version 1.1.8
0: jdbc:phoenix:localhost:2181:/hbase-unsecur>
```

1. And execute the below command:

```
select * from WEB_STAT;
```

```
0: jdbc:phoenix:node1:2181:/hbase-unsecure> select * from WEB_STAT;
```

| HOST | DOMAIN         | FEATURE   | DATE       | CORE |
|------|----------------|-----------|------------|------|
| EU   | Apple.com      | Mac       | 2012-12-31 | 35   |
| EU   | Apple.com      | Store     | 2013-01-02 | 345  |
| EU   | Google.com     | Analytics | 2013-01-13 | 25   |
| EU   | Google.com     | Search    | 2013-01-08 | 395  |
| EU   | Salesforce.com | Dashboard | 2013-01-06 | 12   |
| EU   | Salesforce.com | Login     | 2013-01-11 | 5    |
| EU   | Salesforce.com | Reports   | 2013-01-02 | 25   |
| EU   | Salesforce.com | Reports   | 2013-01-02 | 125  |
| EU   | Salesforce.com | Reports   | 2013-01-04 | 75   |
| EU   | Salesforce.com | Reports   | 2013-01-04 | 475  |
| EU   | Salesforce.com | Reports   | 2013-01-13 | 355  |
| NA   | Apple.com      | Login     | 2012-12-31 | 35   |
| NA   | Apple.com      | Login     | 2013-01-03 | 135  |
| NA   | Apple.com      | Mac       | 2013-01-01 | 345  |
| NA   | Apple.com      | Mac       | 2013-01-07 | 3    |
| NA   | Apple.com      | iPad      | 2013-01-04 | 85   |
| NA   | Apple.com      | iPad      | 2013-01-05 | 35   |
| NA   | Apple.com      | iPad      | 2013-01-05 | 9    |

You can see the entire rows in the table WEB\_STAT which we just created with sql files in phoenix in previous lab.

3. Now run the below query to obtain Average CPU and DB usage by domain:

```
SELECT DOMAIN, AVG(CORE) Average_CPU_Usage, AVG(DB) Average_DB_Usage
FROM WEB_STAT
GROUP BY DOMAIN
ORDER BY DOMAIN DESC;
```

```
0: jdbc:phoenix:node1:2181:/hbase-unsecure> SELECT DOMAIN, AVG(CORE) Average_CPU_Usage, AVG(DB) Average_DB_Usage
. . . . . > FROM WEB_STAT
. . . . . > GROUP BY DOMAIN
. . . . . > ORDER BY DOMAIN DESC;
```

| DOMAIN         | AVERAGE_CPU_USAGE | AVERAGE_DB_USAGE |
|----------------|-------------------|------------------|
| Salesforce.com | 260.7272          | 257.6363         |
| Google.com     | 212.875           | 213.75           |
| Apple.com      | 114.1111          | 119.5555         |

```
3 rows selected (0.131 seconds)
0: jdbc:phoenix:node1:2181:/hbase-unsecure>
```

4. Now run the below query to Sum, Min and Max CPU usage by Salesforce grouped by day

```
SELECT TRUNC(DATE,'DAY') DAY, SUM(CORE) TOTAL_CPU_Usage, MIN(CORE)
MIN_CPU_Usage, MAX(CORE) MAX_CPU_Usage
FROM WEB_STAT
WHERE DOMAIN LIKE 'Salesforce%'
GROUP BY TRUNC(DATE,'DAY');
```

```
0: jdbc:phoenix:node1:2181:/hbase-unsecure> SELECT TRUNC(DATE,'DAY') DAY, SUM(CORE) TOTAL_CPU_Usage, MIN(CORE) MIN_CPU_Usage, MAX(CORE) MAX_CPU_Usage
. . . . . > FROM WEB_STAT
. . . . . > WHERE DOMAIN LIKE 'Salesforce%'
. . . . . > GROUP BY TRUNC(DATE,'DAY');
```

| DAY        | TOTAL_CPU_USAGE | MIN_CPU_USAGE | MAX_CPU_USAGE |
|------------|-----------------|---------------|---------------|
| 2012-12-31 | 35              | 35            | 35            |
| 2013-01-01 | 150             | 25            | 125           |
| 2013-01-02 | 88              | 88            | 88            |
| 2013-01-03 | 26              | 3             | 23            |
| 2013-01-04 | 550             | 75            | 475           |
| 2013-01-05 | 12              | 12            | 12            |
| 2013-01-07 | 345             | 345           | 345           |
| 2013-01-08 | 390             | 35            | 355           |
| 2013-01-09 | 345             | 345           | 345           |
| 2013-01-10 | 335             | 335           | 335           |
| 2013-01-11 | 5               | 5             | 5             |
| 2013-01-12 | 355             | 355           | 355           |
| 2013-01-13 | 5               | 5             | 5             |
| 2013-01-14 | 720             | 65            | 655           |
| 2013-01-15 | 785             | 785           | 785           |
| 2013-01-16 | 1590            | 355           | 1235          |

```
16 rows selected (0.094 seconds)
0: jdbc:phoenix:node1:2181:/hbase-unsecure>
```

5. Now run the below query to list host and total active users when core CPU usage is 10X greater than DB usage:

```
SELECT HOST, SUM(ACTIVE_VISITOR) TOTAL_ACTIVE_VISITORS
FROM WEB_STAT
WHERE DB > (CORE * 10)
GROUP BY HOST;
```

```

0: jdbc:phoenix:node1:2181:/hbase-unsecure> SELECT HOST, SUM(ACTIVE_VISITOR) TOTAL_ACTIVE_VISITORS
. . . . . > FROM WEB_STAT
. . . . . > WHERE DB > (CORE * 10)
. . . . . > GROUP BY HOST;
+-----+-----+
| HOST | TOTAL_ACTIVE_VISITORS |
+-----+-----+
| EU   | 150                   |
| NA   | 1                     |
+-----+-----+
2 rows selected (0.073 seconds)
0: jdbc:phoenix:node1:2181:/hbase-unsecure>

```

6. Exit from the Phoenix client:

```

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

```

## Result

Executed a series of queries on the table created in HBase using Phoenix.

