Table of Contents

Version Supported: hbase 2.4 and Phoenix 5.4

1. Integrating Hbase with Phoenix – 90 Minutes

Expand the latest phoenix-hbase-[hbase.version][phoenix.version]-bin.tar.gz for your HBase version.

```
#tar -xvf phoenix-hbase-2.4.0-5.1.2-bin.tar.gz -C /opt

#mv phoenix-hbase-2.4.0-5.1.2-bin/ phoenix
```

Add the phoenix-server-hbase-[hbase.version]-[phoenix.version].jar to the classpath of all HBase region servers and masters and remove any previous version.

- An easy way to do this is to copy it into the HBase lib directory

```
#cp /opt/phoenix/phoenix-server-hbase-2.4.0-5.1.2.jar /opt/hbase/lib/

#cp /opt/phoenix/phoenix-pherf-5.1.2.jar  /opt/hbase/lib/
```

Export some variables using the shell scripts.

Open the script editor as shown below and add the following variables in it.

```
#vi ~/.bashrc

export HADOOP_HOME=/opt/hadoop
export HBASE_HOME=/opt/hbase
export PHOENIX_HOME=/opt/phoenix
export PATH=$PATH:/opt/jdk/bin:/opt/hbase/bin:/opt/hadoop/bin:$PHOENIX_HOME/bin
export JAVA_HOME=/opt/jdk
```

- (Not required for this lab only for information)Add the phoenix-client-hbase-[hbase.version]-[phoenix.version].jar to the classpath of any JDBC client.
  Do not copy the above client library in the hbase/lib folder.

  Add the following property in the hbase-site.xml

```
<property>
<name>hbase.regionserver.wal.codec</name>
  <value>org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec</value>
</property>
```

Restart HBase.

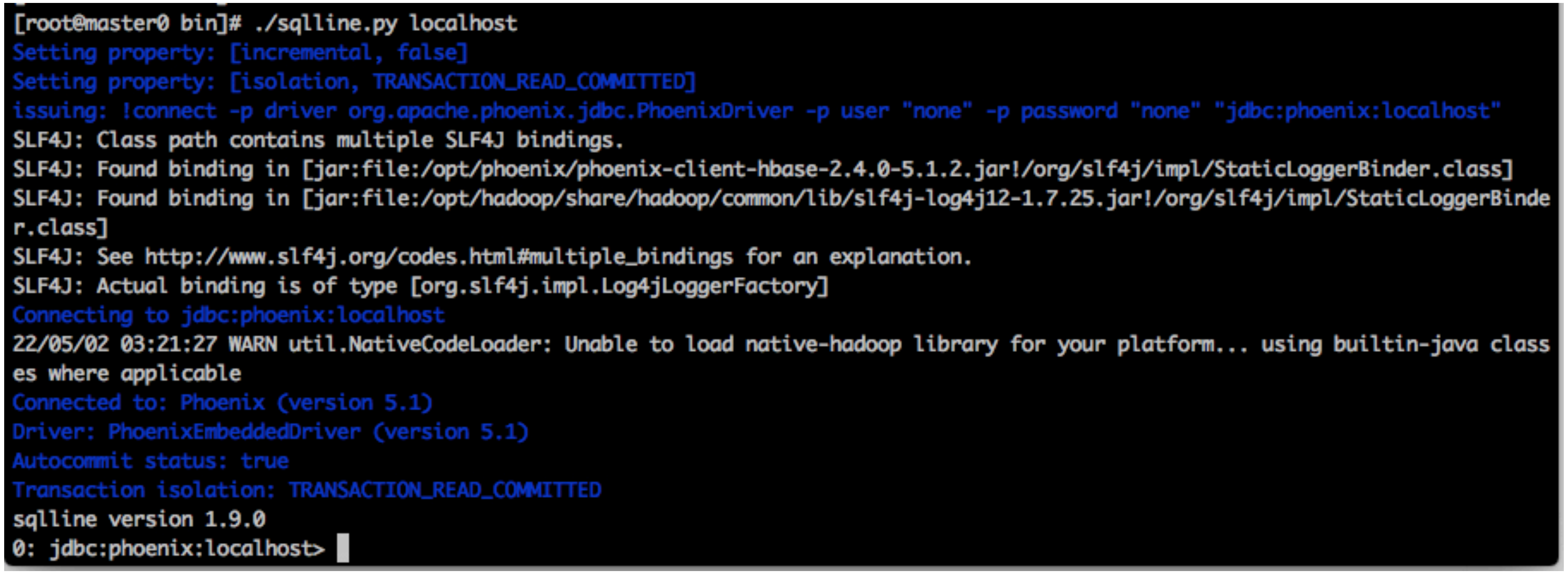
```
#cd /opt/hbase

#/opt/hbase/bin/stop-hbase.sh

#/opt/hbase/bin/start-hbase.sh
```

To connect Hbase using Phoenix Command Line perform the following.

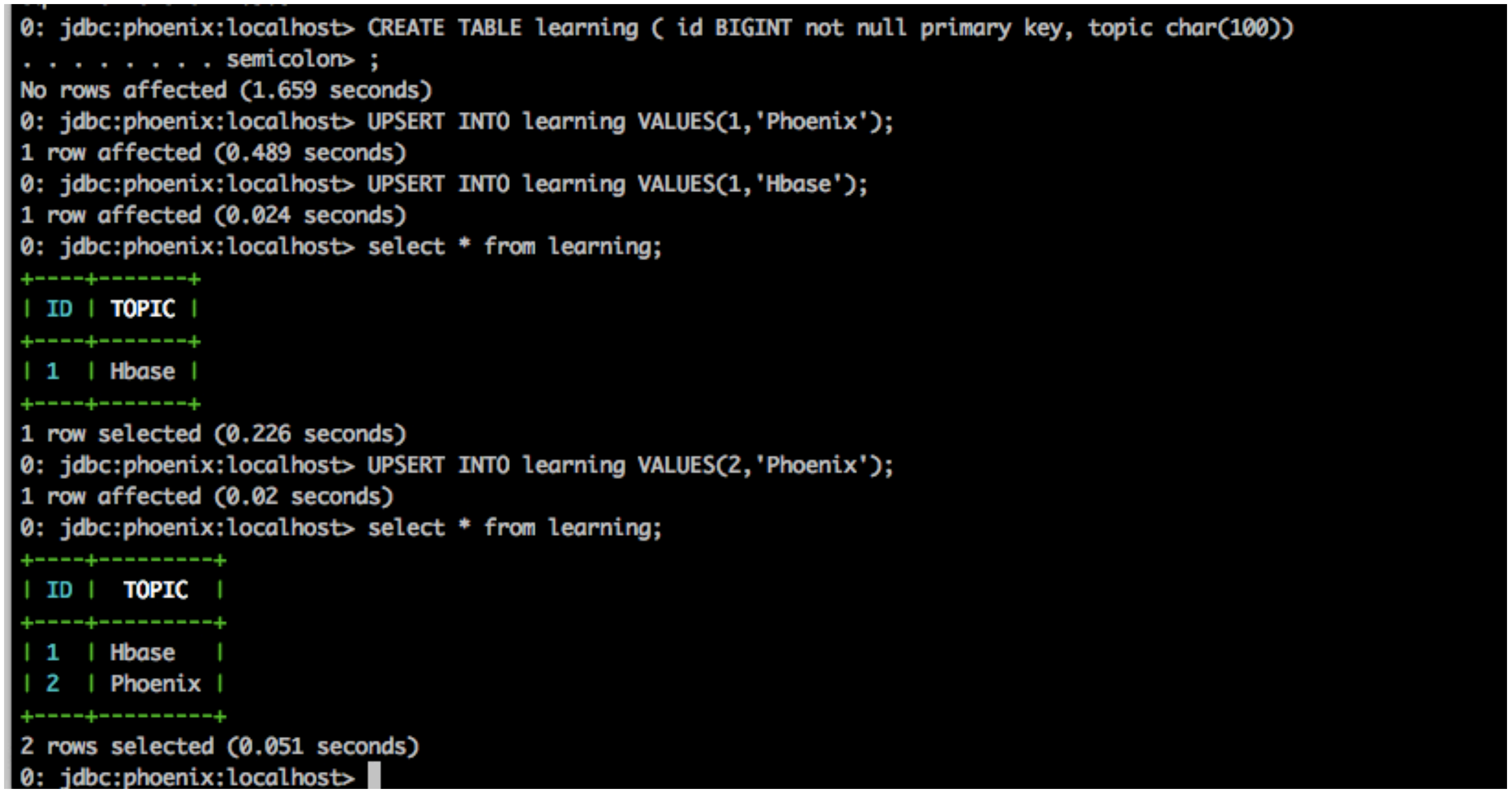
- A terminal interface to execute SQL from the command line is now bundled with Phoenix. To start it, execute the following from the bin directory of phoenix folder:
  # /opt/phoenix/bin
  $ ./sqlline.py localhost



```
[root@master0 bin]# ./sqlline.py localhost
Setting property: [incremental, false]
Setting property: [isolation, TRANSACTION_READ_COMMITTED]
issuing: !connect -p driver org.apache.phoenix.jdbc.PhoenixDriver -p user "none" -p password "none" "jdbc:phoenix:localhost"
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/phoenix/phoenix-client-hbase-2.4.0-5.1.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinde
r.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Connecting to jdbc:phoenix:localhost
22/05/02 03:21:27 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java class
es where applicable
Connected to: Phoenix (version 5.1)
Driver: PhoenixEmbeddedDriver (version 5.1)
Autocommit status: true
Transaction isolation: TRANSACTION_READ_COMMITTED
sqlline version 1.9.0
0: jdbc:phoenix:localhost>
```

Create a table and insert 2 records:

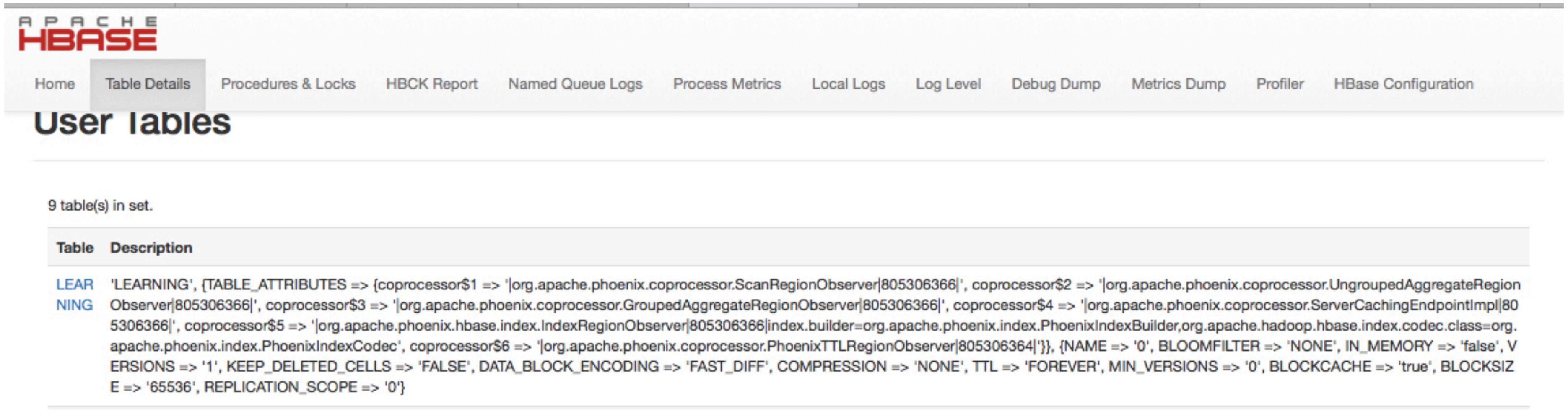
```
#CREATE TABLE learning ( id BIGINT not null primary key, topic char(100));
#UPSERT INTO learning VALUES(1,'Phoenix');
#UPSERT INTO learning VALUES(1,'Hbase');
#UPSERT INTO learning VALUES(2,'Phoenix');
#select * from learning;
```



```
0: jdbc:phoenix:localhost> CREATE TABLE learning ( id BIGINT not null primary key, topic char(100))
. . . . . . . . semicolon> ;
No rows affected (1.659 seconds)
0: jdbc:phoenix:localhost> UPSERT INTO learning VALUES(1,'Phoenix');
1 row affected (0.489 seconds)
0: jdbc:phoenix:localhost> UPSERT INTO learning VALUES(1,'Hbase');
1 row affected (0.024 seconds)
0: jdbc:phoenix:localhost> select * from learning;
+----+--------+
| ID | TOPIC |
+----+--------+
| 1  | Hbase |
+----+--------+
1 row selected (0.226 seconds)
0: jdbc:phoenix:localhost> UPSERT INTO learning VALUES(2,'Phoenix');
1 row affected (0.02 seconds)
0: jdbc:phoenix:localhost> select * from learning;
+----+----------+
| ID |  TOPIC   |
+----+----------+
| 1  | Hbase    |
| 2  | Phoenix |
+----+----------+
2 rows selected (0.051 seconds)
0: jdbc:phoenix:localhost>
```

You can also verify from the Hbase web console:

#http://localhost:16010/tablesDetailed.jsp



User Tables

9 table(s) in set.

Table	Description
LEARNING	'LEARNING', {TABLE_ATTRIBUTES => {coprocessor$1 => '|org.apache.phoenix.coprocessor.ScanRegionObserver|805306366|', coprocessor$2 => '|org.apache.phoenix.coprocessor.UngroupedAggregateRegionObserver|805306366|', coprocessor$3 => '|org.apache.phoenix.coprocessor.GroupedAggregateRegionObserver|805306366|', coprocessor$4 => '|org.apache.phoenix.coprocessor.ServerCachingEndpointImpl|805306366|', coprocessor$5 => '|org.apache.phoenix.hbase.index.IndexRegionObserver|805306366|index.builder=org.apache.phoenix.index.PhoenixIndexBuilder,org.apache.hadoop.hbase.index.codec.class=org.apache.phoenix.index.PhoenixIndexCodec', coprocessor$6 => '|org.apache.phoenix.coprocessor.PhoenixTTLRegionObserver|805306364|'}}, {NAME => '0', BLOOMFILTER => 'NONE', IN_MEMORY => 'false', VERSIONS => '1', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'FAST_DIFF', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}

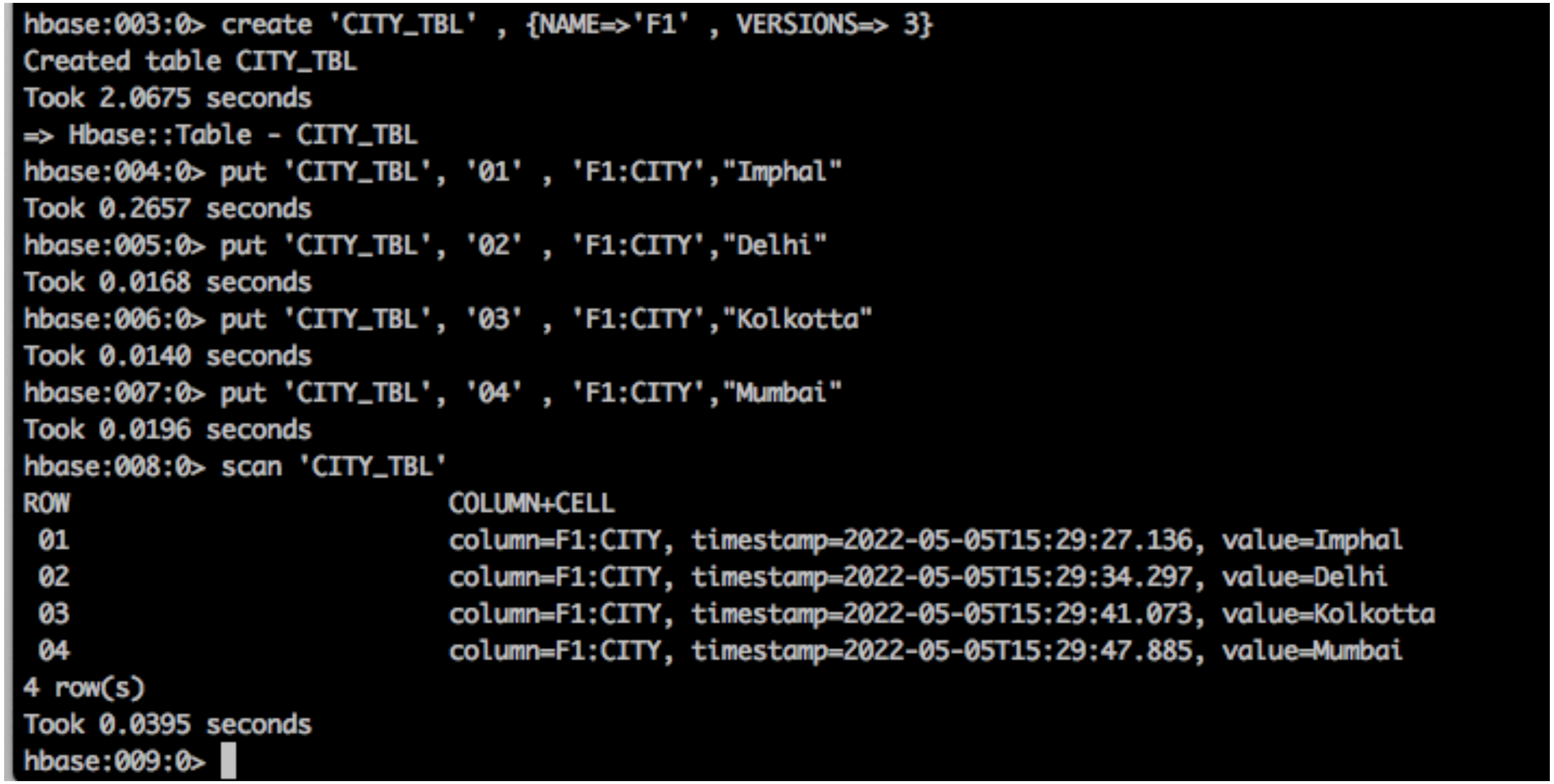
Reading Existing Hbase Table using Phoenix.

Create the following table using Hbase Shell and insert four records.

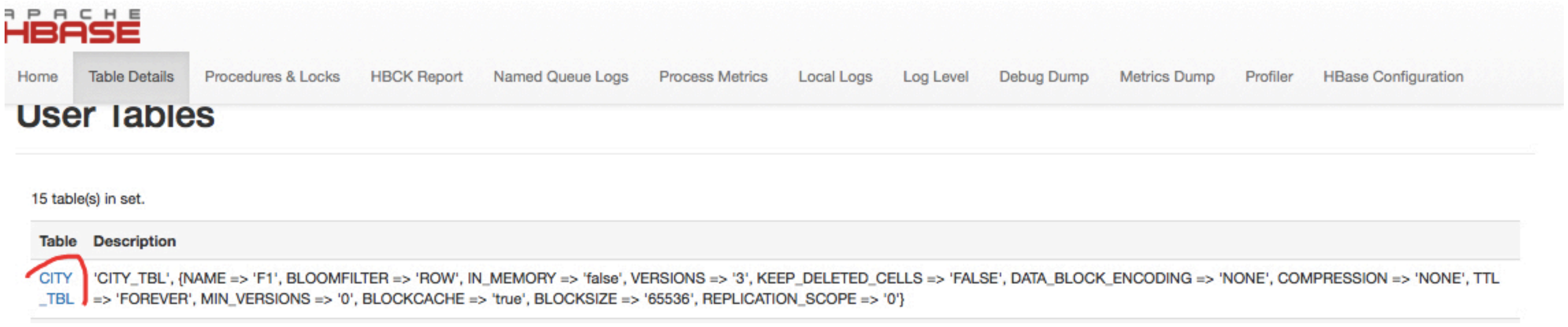
```
/* DML Queries  Begin*/

create 'CITY_TBL' , {NAME=>'F1' , VERSIONS=> 3}
put 'CITY_TBL', '01' , 'F1:CITY',"Imphal"
put 'CITY_TBL', '02' , 'F1:CITY',"Delhi"
put 'CITY_TBL', '03' , 'F1:CITY',"Kolkotta"
put 'CITY_TBL', '04' , 'F1:CITY',"Mumbai"
scan 'CITY_TBL'

/* DML Queries  Ends*/
```


```
hbase:003:0> create 'CITY_TBL' , {NAME=>'F1' , VERSIONS=> 3}
Created table CITY_TBL
Took 2.0675 seconds
=> Hbase::Table - CITY_TBL
hbase:004:0> put 'CITY_TBL', '01' , 'F1:CITY',"Imphal"
Took 0.2657 seconds
hbase:005:0> put 'CITY_TBL', '02' , 'F1:CITY',"Delhi"
Took 0.0168 seconds
hbase:006:0> put 'CITY_TBL', '03' , 'F1:CITY',"Kolkotta"
Took 0.0140 seconds
hbase:007:0> put 'CITY_TBL', '04' , 'F1:CITY',"Mumbai"
Took 0.0196 seconds
hbase:008:0> scan 'CITY_TBL'
ROW                           COLUMN+CELL
 01                           column=F1:CITY, timestamp=2022-05-05T15:29:27.136, value=Imphal
 02                           column=F1:CITY, timestamp=2022-05-05T15:29:34.297, value=Delhi
 03                           column=F1:CITY, timestamp=2022-05-05T15:29:41.073, value=Kolkotta
 04                           column=F1:CITY, timestamp=2022-05-05T15:29:47.885, value=Mumbai
4 row(s)
Took 0.0395 seconds
hbase:009:0>
```

Verify the table using the web console.



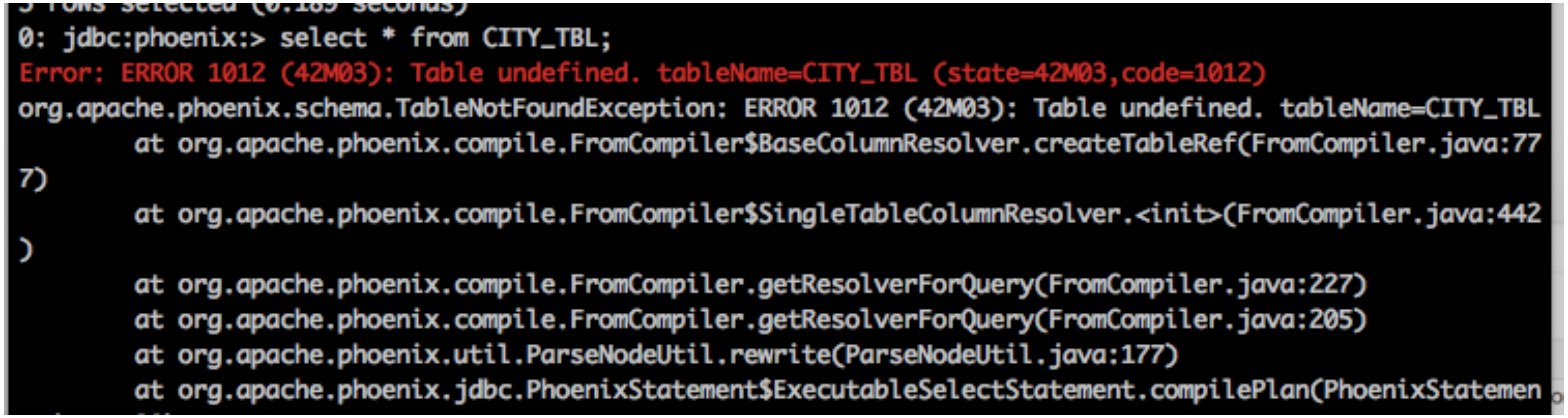
User Tables

15 table(s) in set.

Table	Description
CITY_TBL	'CITY_TBL', {NAME => 'F1', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '3', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}

Fetch Record from the Phoenix shell.

#select * from CITY_TBL;



```
3 rows selected (0.189 seconds)
0: jdbc:phoenix:> select * from CITY_TBL;
Error: ERROR 1012 (42M03): Table undefined. tableName=CITY_TBL (state=42M03,code=1012)
org.apache.phoenix.schema.TableNotFoundException: ERROR 1012 (42M03): Table undefined. tableName=CITY_TBL
	at org.apache.phoenix.compile.FromCompiler$BaseColumnResolver.createTableRef(FromCompiler.java:77
7)
	at org.apache.phoenix.compile.FromCompiler$SingleTableColumnResolver.<init>(FromCompiler.java:442
)
	at org.apache.phoenix.compile.FromCompiler.getResolverForQuery(FromCompiler.java:227)
	at org.apache.phoenix.compile.FromCompiler.getResolverForQuery(FromCompiler.java:205)
	at org.apache.phoenix.util.ParseNodeUtil.rewrite(ParseNodeUtil.java:177)
	at org.apache.phoenix.jdbc.PhoenixStatement$ExecutableSelectStatement.compilePlan(PhoenixStatemen
```

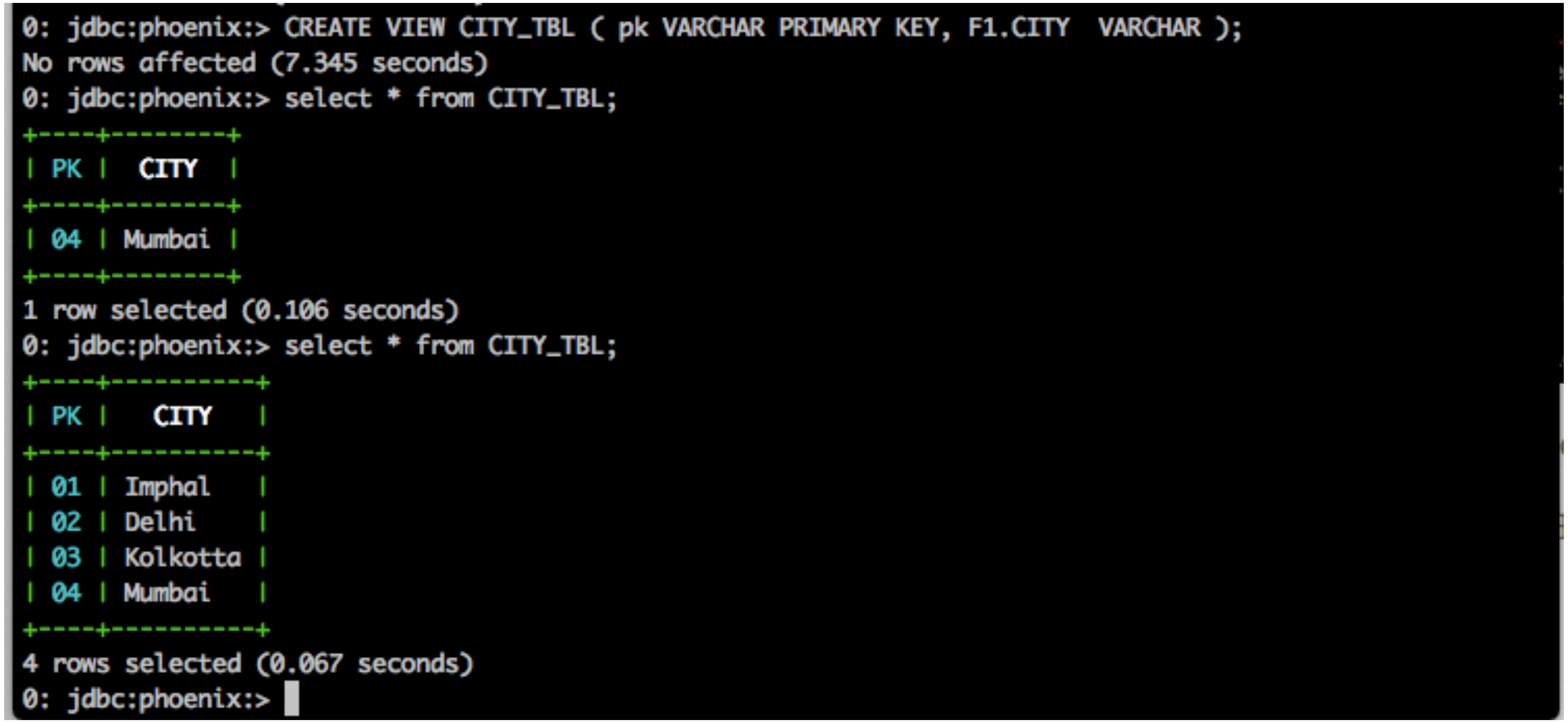
This error is because the Phoenix can't read the Meta data from the Hbase directly.
You can create view to fetch the metadata from the phoenix.

Execute the following view command in the Phoenix CLI.

#CREATE VIEW CITY_TBL (pk VARCHAR PRIMARY KEY, F1.CITY VARCHAR);

Try fetching records again from the Phoenix CLI

# select * from CITY_TBL;



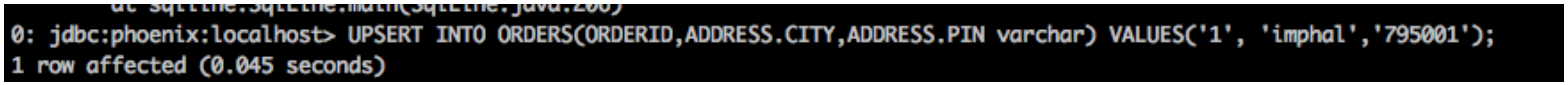
```
0: jdbc:phoenix:> CREATE VIEW CITY_TBL ( pk VARCHAR PRIMARY KEY, F1.CITY  VARCHAR );
No rows affected (7.345 seconds)
0: jdbc:phoenix:> select * from CITY_TBL;
+----+--------+
| PK |  CITY  |
+----+--------+
| 04 | Mumbai |
+----+--------+
1 row selected (0.106 seconds)
0: jdbc:phoenix:> select * from CITY_TBL;
+----+----------+
| PK |   CITY   |
+----+----------+
| 01 | Imphal   |
| 02 | Delhi    |
| 03 | Kolkotta |
| 04 | Mumbai   |
+----+----------+
4 rows selected (0.067 seconds)
0: jdbc:phoenix:>
```

Dynamic Columns Using Phoenix CLI:

#CREATE TABLE ORDERS (ORDERID VARCHAR PRIMARY KEY, ADDRESS.CITY VARCHAR);
#UPSERT INTO ORDERS(ORDERID,ADDRESS.CITY) VALUES('1', 'imphal');

Let us add a column, Address.pincode dynamically.

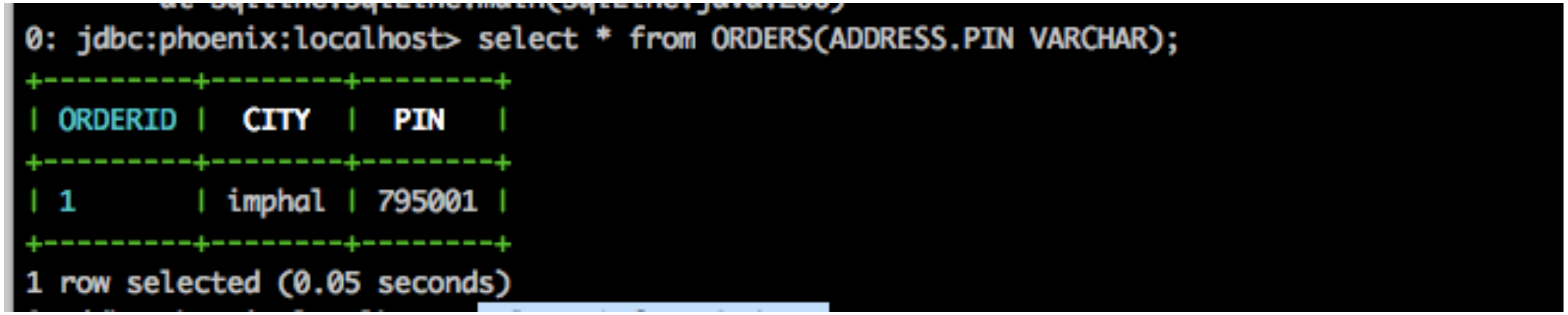
# UPSERT INTO ORDERS(ORDERID,ADDRESS.CITY,ADDRESS.PIN varchar) VALUES('1', 'imphal','795001');



```
0: jdbc:phoenix:localhost> UPSERT INTO ORDERS(ORDERID,ADDRESS.CITY,ADDRESS.PIN varchar) VALUES('1', 'imphal','795001');
1 row affected (0.045 seconds)
```

Fetch records with the Dynamic Column

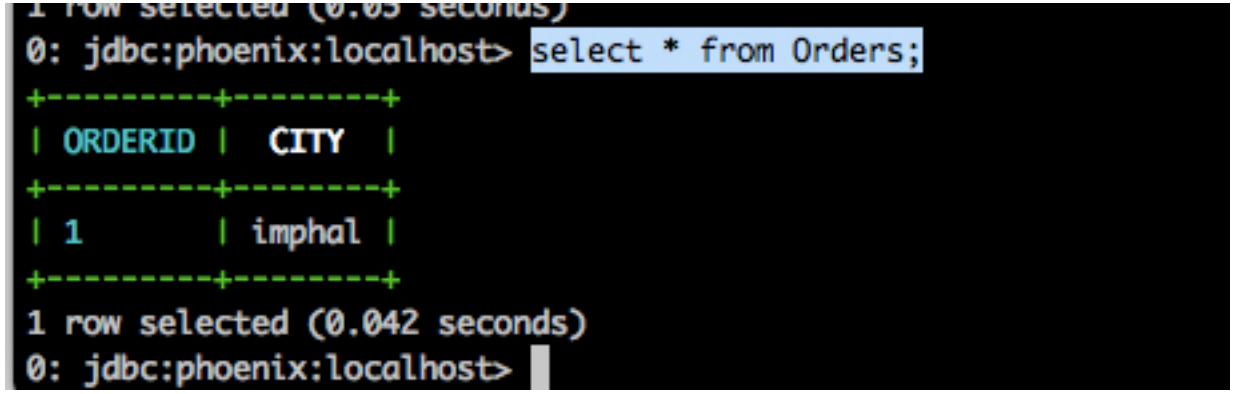
#select * from ORDERS(ADDRESS.PIN VARCHAR);



```
0: jdbc:phoenix:localhost> select * from ORDERS(ADDRESS.PIN VARCHAR);
+---------+--------+--------+
| ORDERID |  CITY  |  PIN   |
+---------+--------+--------+
| 1       | imphal | 795001 |
+---------+--------+--------+
1 row selected (0.05 seconds)
```

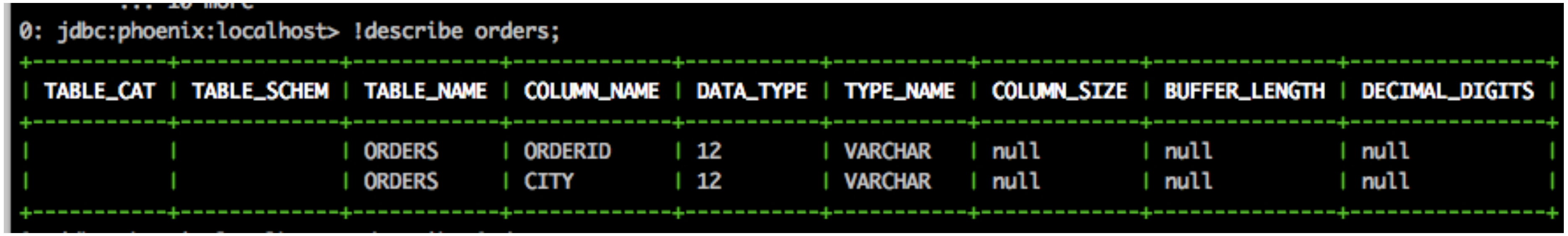
Fetch records without the Dynamic Column

# select * from Orders;



```
1 row selected (0.05 seconds)
0: jdbc:phoenix:localhost> select * from Orders;
+---------+--------+
| ORDERID |  CITY  |
+---------+--------+
| 1       | imphal |
+---------+--------+
1 row selected (0.042 seconds)
0: jdbc:phoenix:localhost> 
```

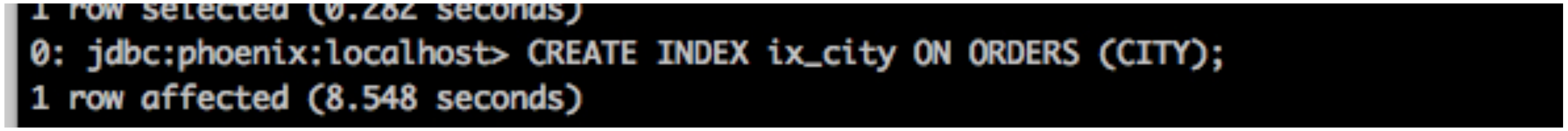
Let us get the definition of the table.
# !describe orders;



```
    ... 10 more
0: jdbc:phoenix:localhost> !describe orders;
+-----------+-------------+------------+-------------+-----------+-----------+-------------+---------------+----------------+
| TABLE_CAT | TABLE_SCHEM | TABLE_NAME | COLUMN_NAME | DATA_TYPE | TYPE_NAME | COLUMN_SIZE | BUFFER_LENGTH | DECIMAL_DIGITS |
+-----------+-------------+------------+-------------+-----------+-----------+-------------+---------------+----------------+
|           |             | ORDERS     | ORDERID     | 12        | VARCHAR   | null        | null          | null           |
|           |             | ORDERS     | CITY        | 12        | VARCHAR   | null        | null          | null           |
+-----------+-------------+------------+-------------+-----------+-----------+-------------+---------------+----------------+
```

Index

#CREATE INDEX ix_city ON ORDERS (CITY);



```
1 row selected (0.282 seconds)
0: jdbc:phoenix:localhost> CREATE INDEX ix_city ON ORDERS (CITY);
1 row affected (8.548 seconds)
```

--------------------------------- Lab Ends Here ---

Error:

```
2022-04-30T16:00:17,884 ERROR [RS_OPEN_PRIORITY_REGION-regionserver/master0:16020-0]
coprocessor.CoprocessorHost: The coprocessor org.apache.phoenix.coprocessor.MetaDataRegionObserver threw
java.lang.NoClassDefFoundError: org/apache/hadoop/hbase/filter/CompareFilter$CompareOp
java.lang.NoClassDefFoundError: org/apache/hadoop/hbase/filter/CompareFilter$CompareOp
	at org.apache.phoenix.expression.BaseExpression.<clinit>(BaseExpression.java:66) ~[phoenix-server-hbase-2.4.0-
5.1.2.jar:5.1.2]
	at org.apache.phoenix.jdbc.PhoenixDatabaseMetaData.<clinit>(PhoenixDatabaseMetaData.java:604) ~[phoenix-server-hbase-
2.4.0-5.1.2.jar:5.1.2]
	at org.apache.phoenix.coprocessor.MetaDataRegionObserver.<clinit>(MetaDataRegionObserver.java:107) ~[phoenix-server-
hbase-2.4.0-5.1.2.jar:5.1.2]
	at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method) ~[?:1.8.0_45]
	at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:62) ~[?:1.8.0_45]
	at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45) ~[?:1.8.0_45]
	at java.lang.reflect.Constructor.newInstance(Constructor.java:422) ~[?:1.8.0_45]
	at org.apache.hadoop.hbase.regionserver.RegionCoprocessorHost.checkAndGetInstance(RegionCoprocessorHost.java:437)
~[hbase-server-3.0.0-alpha-1.jar:3.0.0-alpha-1]
	at org.apache.hadoop.hbase.regionserver.RegionCoprocessorHost.checkAndGetInstance(RegionCoprocessorHost.java:90)
~[hbase-server-3.0.0-alpha-1.jar:3.0.0-alpha-1]
	at org.apache.hadoop.hbase.coprocessor.CoprocessorHost.checkAndLoadInstance(CoprocessorHost.java:272) ~[hbase-server-
3.0.0-alpha-1.jar:3.0.0-alpha-1]
	at org.apache.hadoop.hbase.coprocessor.CoprocessorHost.load(CoprocessorHost.java:247) ~[hbase-server-3.0.0-alpha-
1.jar:3.0.0-alpha-1]
```

Solution: User hbase 2.4.1 and Phoenix. 5.1.2

Error:

```
coprocessor.CoprocessorHost: The coprocessor org.apache.phoenix.coprocessor.ServerCachingEndpointImpl threw
java.lang.ClassCastException: org.apache.phoenix.coprocessor.ServerCachingEndpointImpl cannot be cast to
com.google.protobuf.Service
```

Solution: Remove phoenix-client-hbase-2.4.0-5.1.2.jar from the /opt/hbase/jar folder.

Error:
ERROR: org.apache.hadoop.hbase.PleaseHoldException: Master is initializing

Install zookeeper and access it to remove the following entry.

```
#bin/zkCli.sh
[zk: localhost:2181(CONNECTED) 3] get /hbase/meta-region-server
?master:16000?m l5???PBUF

master0.hp.com?}?????0

#delete /hbase/meta-region-server
```

Restart the hbase.

Errors:

Mutable secondary indexes must have the hbase.regionserver.wal.codec property set to org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec in the hbase-sites.xml of every region server. tableName=IX_CITY

Solution: (hbase-site.xml)
`hbase.regionserver.wal.codecorg.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec`

Set hbase.regionserver.wal.codec to enable custom write-ahead log ("WAL") edits to be written as follows:

```
<property>
  <name>hbase.regionserver.wal.codec</name>
  <value>org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec</value>
</property>
```