Kalyan Big Data Projects – Project 1
How To Stream CSV Data Into Phoenix Using Apache Kafka

Pre-Requisites of Flume Project:

hadoop-2.6.0
kafka-0.9.0
hbase-1.1.2
phoenix-4.7.0
java-1.7

NOTE: Make sure that install all the above components

Flume Project Download Links:

`hadoop-2.6.0.tar.gz` ==> link
(https://archive.apache.org/dist/hadoop/core/hadoop-2.6.0/hadoop-2.6.0.tar.gz)

`kafka_2.11-0.9.0.0.tgz` ==> link
(https://archive.apache.org/dist/kafka/0.9.0.0/kafka_2.11-0.9.0.0.tgz)

`hbase-1.1.2-bin.tar.gz` ==> link
(https://archive.apache.org/dist/hbase/1.1.2/hbase-1.1.2-bin.tar.gz)

`phoenix-4.7.0-HBase-1.1-bin.tar.gz` ==> link
(https://archive.apache.org/dist/phoenix/phoenix-4.7.0-HBase-1.1/bin/phoenix-4.7.0-HBase-1.1-bin.tar.gz)

`kalyan-kafka-consumer-csv.properties` ==> link
(https://github.com/kalyanhadooptraining/kalyan-bigdata-realtime-projects/blob/master/kafka/project1-phoenix-csv/kalyan-kafka-consumer-csv.properties)

`kalyan-bigdata-examples.jar` ==> link
(https://github.com/kalyanhadooptraining/kalyan-bigdata-realtime-projects/blob/master/kalyan/kalyan-bigdata-examples.jar)

`kalyan-phoenix-kafka-4.7.0-HBase-1.1-minimal.jar` ==> link
(https://github.com/kalyanhadooptraining/kalyan-bigdata-realtime-projects/blob/master/kalyan/kalyan-phoenix-kafka-4.7.0-HBase-1.1-minimal.jar)

`json-path-2.2.0.jar` ==> link
(http://central.maven.org/maven2/com/jayway/jsonpath/json-path/2.2.0/json-path-2.2.0.jar)

`commons-io-2.4.jar` ==> link
(http://central.maven.org/maven2/commons-io/commons-io/2.4/commons-io-2.4.jar)

Learnings of this Project:

- We will learn Kafka Configurations and Commands
- Kafka Information
  1. Kalyan Util (CSV data generator)
  2. Kafka Producer (Listen on CSV data)
  3. Kafka Consumer (Recieves the data from Kafka Producer)
  4. Phoenix Consumer (Write the data into Phoenix Table)
- Major project in Real Time `Product Log Analysis`
  1. We are extracting the data from server logs
  2. This data will be useful to do analysis on product views
  3. CSV is the output format
- We can use phoenix to analyze this data

---

1. create "**kalyan-kafka-consumer-csv.properties**" file with below content in "**/home/orienit**" folder

```
serializer=org.apache.phoenix.flume.serializer.CsvEventSerializer
serializer.delimiter=,
serializer.columns=userid,username,password,email,country,state,city,dt

jdbcUrl=jdbc:phoenix:localhost
table=users4
ddl=CREATE TABLE IF NOT EXISTS users4 (userid BIGINT NOT NULL, username
VARCHAR, password VARCHAR, email VARCHAR, country VARCHAR, state VARCHAR, city
VARCHAR, dt VARCHAR NOT NULL CONSTRAINT PK PRIMARY KEY (userid, dt))

bootstrap.servers=localhost:9092
topics=csv-topic1,csv-topic2
poll.timeout.ms=100
```

2. Copy "**kalyan-kafka-consumer-csv.properties**" file into hdfs location is "**/kalyan/kafka**"

- Create `**/kalyan/kafka**` folder in hdfs using below command
  - *hadoop fs -mkdir -p /kalyan/kafka*

- Put the properties file into hdfs using below command
  - *hadoop fs -put /home/orienit/kalyan-kafka-consumer-csv.properties /kalyan/kafka/kalyan-kafka-consumer-csv.properties*

3. Copy "**kalyan-phoenix-kafka-4.7.0-Hbase-1.1-minimal.jar**" file into "**$PHONIEX_HOME**" folder

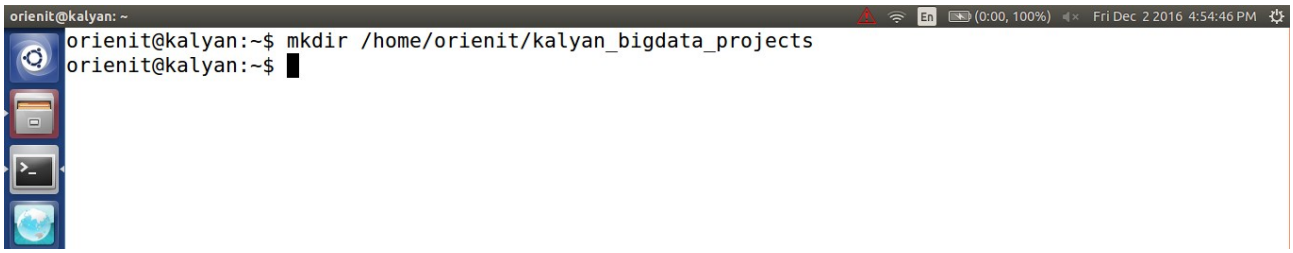
4. Generate Large Amount of Sample CSV data follow this article.

(http://kalyanbigdatatraining.blogspot.com/2016/12/how-to-generate-large-amount-of-sample.html)

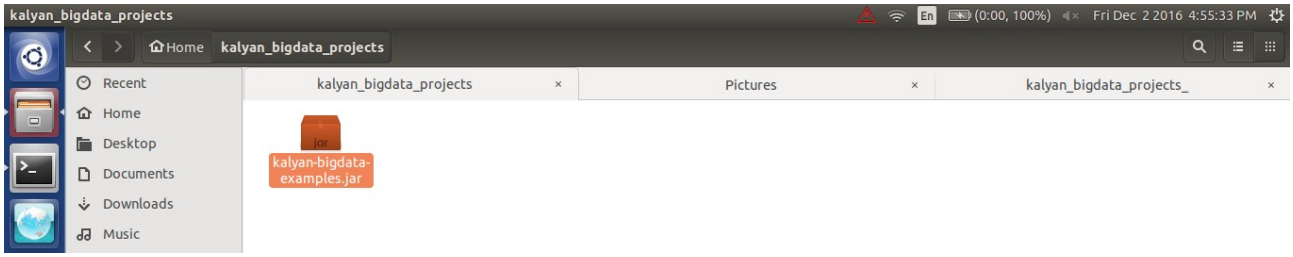
5. Follow below steps...

i) Create '**kalyan_bigdata_projects**' folder in **user home** (i.e **/home/orienit**)

Command: *mkdir /home/orienit/kalyan_bigdata_projects*

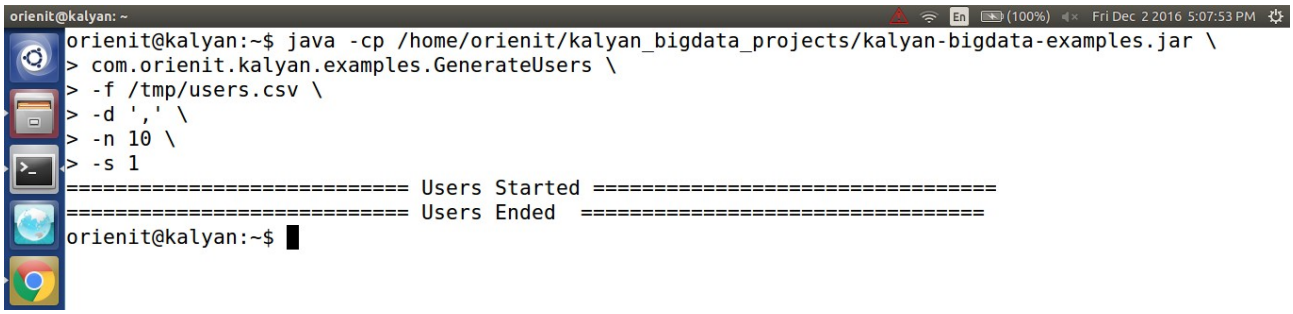


ii) Copy '**kalyan-bigdata-examples.jar**' jar file into '**/home/orienit/kalyan_bigdata_projects**' folder



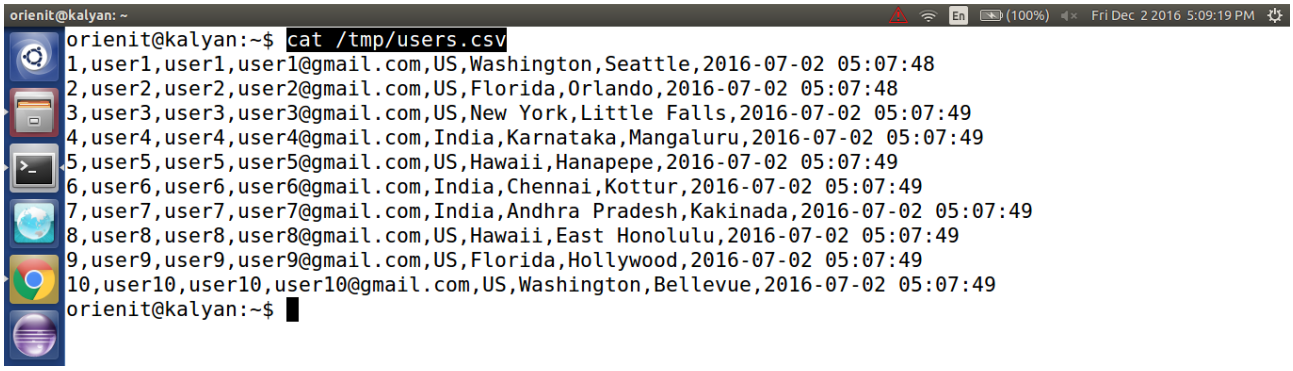
iii) Execute Below Command to Generate Sample CSV data with 10 lines. Increase this number to get more data ...

```
java -cp /home/orienit/kalyan_bigdata_projects/kalyan-bigdata-examples.jar \
com.orienit.kalyan.examples.GenerateUsers \
-f /tmp/users.csv \
-d ',' \
-n 10 \
-s 1
```



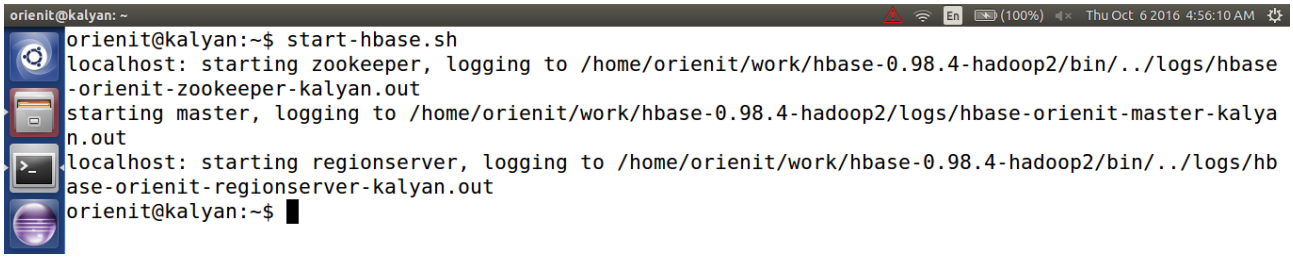
6. Verify the Sample CSV data in Console, using below command

cat /tmp/users.csv

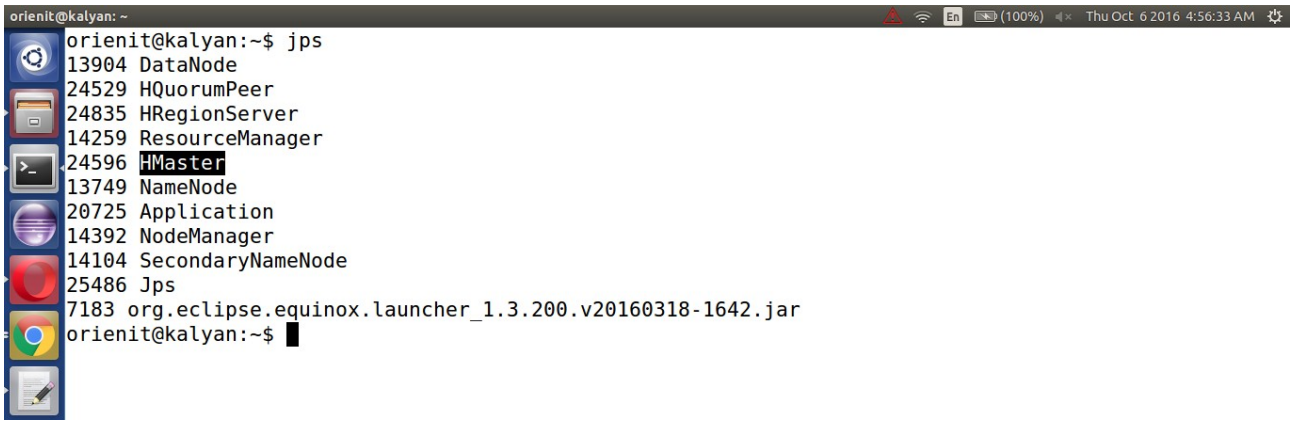


7. To work with **Kafka + Phoenix Integration,** Follow the below steps

i. Start the hbase using below '**start-hbase.sh**' command.



ii. verify the hbase is running or not with "**jps**" command



iii. Start the phoenix using below '**sqlline.py localhost**' command.

```
orienit@kalyan: ~                                                    Thu Oct 6 2016 5:25:26 PM
orienit@kalyan:~$ sqlline.py localhost
Setting property: [incremental, false]
Setting property: [isolation, TRANSACTION_READ_COMMITTED]
issuing: !connect jdbc:phoenix:localhost none none org.apache.phoenix.jdbc.PhoenixDriver
Connecting to jdbc:phoenix:localhost
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/orienit/work/phoenix-4.7.0-HBase-1.1-bin/phoenix-4.7.0-HBas
e-1.1-client.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/orienit/work/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log
4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
16/10/06 17:25:05 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platfor
m... using builtin-java classes where applicable
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)...
83/83 (100%) Done
Done
sqlline version 1.1.8
0: jdbc:phoenix:localhost>
```

iv. List out all the tables in phoenix using '**!tables**' command

```
orienit@kalyan: ~                                                    Thu Oct 6 2016 5:26:31 PM
0: jdbc:phoenix:localhost> !tables
+------------+-------------+-------------+---------------+----------+------------+----------------+
| TABLE_CAT  | TABLE_SCHEM | TABLE_NAME  |  TABLE_TYPE   | REMARKS  | TYPE_NAME  | SELF_REFERENC  |
+------------+-------------+-------------+---------------+----------+------------+----------------+
|            | SYSTEM      | CATALOG     | SYSTEM TABLE  |          |            |                |
|            | SYSTEM      | FUNCTION    | SYSTEM TABLE  |          |            |                |
|            | SYSTEM      | SEQUENCE    | SYSTEM TABLE  |          |            |                |
|            | SYSTEM      | STATS       | SYSTEM TABLE  |          |            |                |
+------------+-------------+-------------+---------------+----------+------------+----------------+
0: jdbc:phoenix:localhost>
```

v. Start the `**kafka server**` using below command (New Terminal)
$KAFKA_HOME/bin/kafka-server-start.sh $KAFKA_HOME/config/server.properties

```
orienit@kalyan: ~                                                    Tue Jan 3 2017 4:43:25 PM
orienit@kalyan: ~                                  orienit@kalyan: ~
orienit@kalyan:~$ $KAFKA_HOME/bin/kafka-server-start.sh $KAFKA_HOME/config/server.properties
[2017-01-03 16:43:14,430] INFO KafkaConfig values:
        advertised.host.name = null
        metric.reporters = []
        quota.producer.default = 9223372036854775807
        offsets.topic.num.partitions = 50
        log.flush.interval.messages = 9223372036854775807
        auto.create.topics.enable = true
        controller.socket.timeout.ms = 30000
        log.flush.interval.ms = null
        principal.builder.class = class org.apache.kafka.common.security.auth.DefaultPrincipalBuild
er
        replica.socket.receive.buffer.bytes = 65536
        min.insync.replicas = 1
        replica.fetch.wait.max.ms = 500
        num.recovery.threads.per.data.dir = 1
        ssl.keystore.type = JKS
        default.replication.factor = 1
        ssl.truststore.password = null
        log.preallocate = false
```

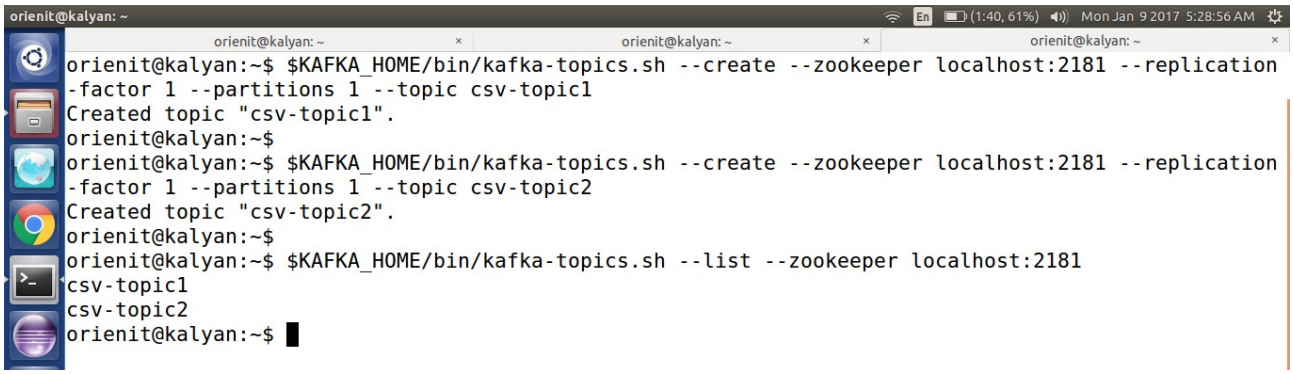
vi. Create a `**csv-topic1 & csv-topic2**` topics using below command (New Terminal)

$KAFKA_HOME/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic csv-topic1

$KAFKA_HOME/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic csv-topic2

vii. List out all the topics

$KAFKA_HOME/bin/kafka-topics.sh --list --zookeeper localhost:2181

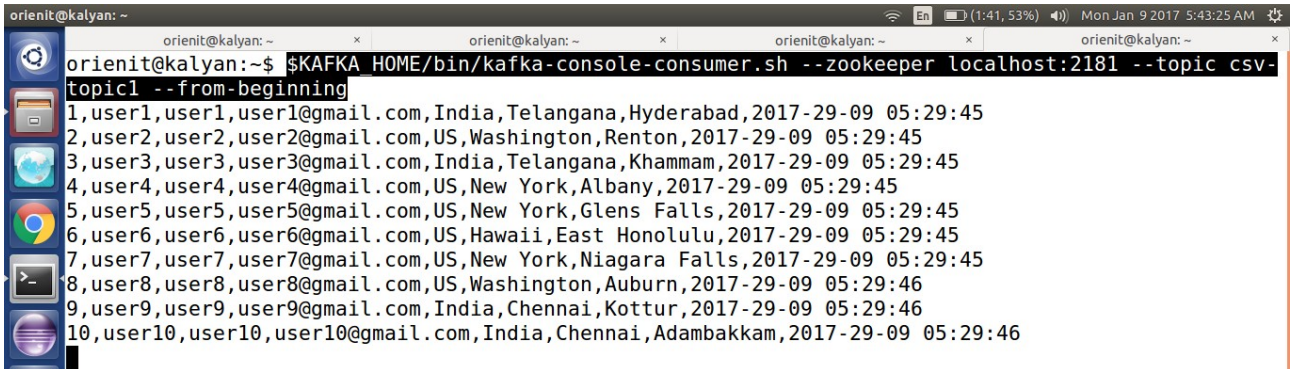


viii. Start the `**kafka producer**` using below command (New Terminal)

tail -f /tmp/users.csv | $KAFKA_HOME/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic csv-topic1

ix. Start the `**kafka consumer**` using below command (New Terminal)

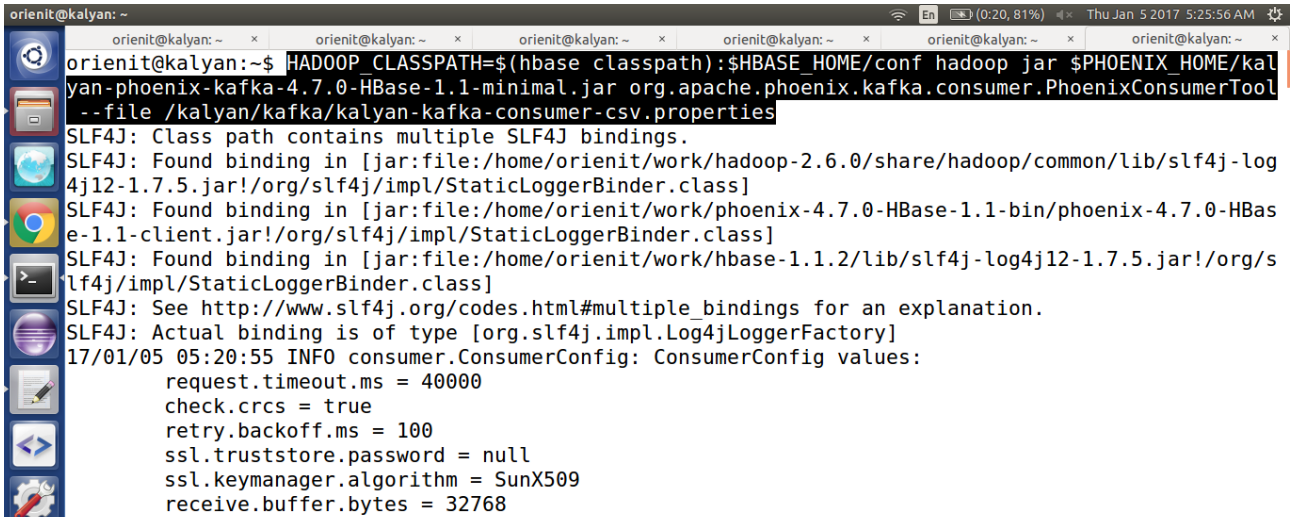
$KAFKA_HOME/bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic csv-topic1 --from-beginning



8. Execute the below command to **`Extract data from CSV data into Phoenix using Kafka`**

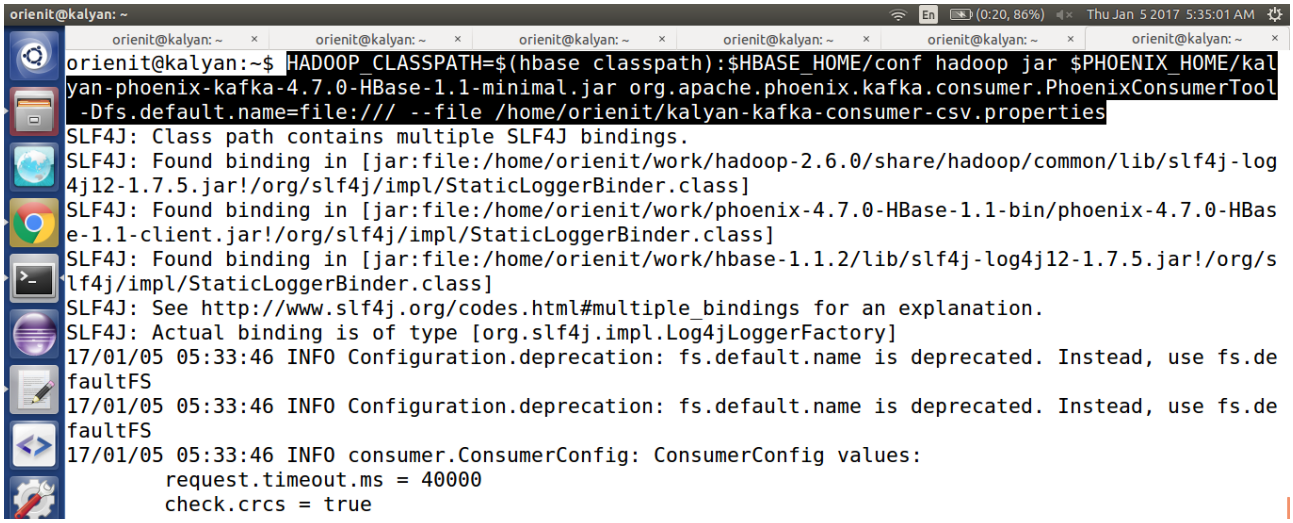
Properties file from HDFS:

HADOOP_CLASSPATH=$(hbase classpath):$HBASE_HOME/conf hadoop jar $PHOENIX_HOME/kalyan-phoenix-kafka-4.7.0-HBase-1.1-minimal.jar org.apache.phoenix.kafka.consumer.PhoenixConsumerTool --file /kalyan/kafka/kalyan-kafka-consumer-csv.properties



Properties file from Local Fs:

HADOOP_CLASSPATH=$(hbase classpath):$HBASE_HOME/conf hadoop jar $PHOENIX_HOME/kalyan-phoenix-kafka-4.7.0-HBase-1.1-minimal.jar org.apache.phoenix.kafka.consumer.PhoenixConsumerTool -Dfs.default.name=file:/// --file /home/orienit/kalyan-kafka-consumer-csv.properties



9. Verify the data in console

```
17/01/05 05:20:56 INFO metrics.Metrics: Initializing metrics system: phoenix
17/01/05 05:20:56 INFO impl.MetricsConfig: loaded properties from hadoop-metrics2.properties
17/01/05 05:20:56 INFO impl.MetricsSystemImpl: Scheduled snapshot period at 10 second(s).
17/01/05 05:20:56 INFO impl.MetricsSystemImpl: phoenix metrics system started
17/01/05 05:21:00 INFO client.HBaseAdmin: Created USERS4
17/01/05 05:21:00 INFO serializer.BaseEventSerializer:  the upsert statement is UPSERT  INTO users4
 ("USERID", "USERNAME", "PASSWORD", "EMAIL", "COUNTRY", "STATE", "CITY", "DT") VALUES (?, ?, ?, ?,
?, ?, ?, ?)
Got 10 records after 886 timeouts
```

10. Verify the data in Phoenix , `**users4**` table is created or not using below command

!tables

```
orienit@kalyan: ~                                                    Thu Jan 5 2017 5:23:27 AM
0: jdbc:phoenix:localhost> !tables
+------------+-------------+------------+---------------+----------+------------+----------------+
| TABLE_CAT  | TABLE_SCHEM | TABLE_NAME |  TABLE_TYPE   | REMARKS  | TYPE_NAME  | SELF_REFERENC  |
+------------+-------------+------------+---------------+----------+------------+----------------+
|            | SYSTEM      | CATALOG    | SYSTEM TABLE  |          |            |                |
|            | SYSTEM      | FUNCTION   | SYSTEM TABLE  |          |            |                |
|            | SYSTEM      | SEQUENCE   | SYSTEM TABLE  |          |            |                |
|            | SYSTEM      | STATS      | SYSTEM TABLE  |          |            |                |
|            |             | USERS4     | TABLE         |          |            |                |
+------------+-------------+------------+---------------+----------+------------+----------------+
0: jdbc:phoenix:localhost>
```

11. Execute below command to get the data from phoenix table '**users4**'

select count(*) from users4;

select * from users4;

```
orienit@kalyan: ~                                                    Thu Jan 5 2017 5:23:46 AM
0: jdbc:phoenix:localhost> select count(*) from users4;
+-----------+
| COUNT(1)  |
+-----------+
| 10        |
+-----------+
1 row selected (0.021 seconds)
0: jdbc:phoenix:localhost> select * from users4;
+---------+-----------+-----------+-------------------+---------+----------------+--------------+
| USERID  | USERNAME  | PASSWORD  |       EMAIL       | COUNTRY |     STATE      |     CITY     |
+---------+-----------+-----------+-------------------+---------+----------------+--------------+
| 1       | user1     | user1     | user1@gmail.com   | India   | Andhra Pradesh | Kakinada     |
| 2       | user2     | user2     | user2@gmail.com   | US      | Florida        | South Beach  |
| 3       | user3     | user3     | user3@gmail.com   | India   | Telangana      | Hyderabad    |
| 4       | user4     | user4     | user4@gmail.com   | India   | Andhra Pradesh | Rajahmundry  |
| 5       | user5     | user5     | user5@gmail.com   | India   | Telangana      | Nalgonda     |
| 6       | user6     | user6     | user6@gmail.com   | India   | Chennai        | Tambaram     |
| 7       | user7     | user7     | user7@gmail.com   | India   | Telangana      | Karimnagar   |
| 8       | user8     | user8     | user8@gmail.com   | India   | Karnataka      | Mangaluru    |
| 9       | user9     | user9     | user9@gmail.com   | US      | Hawaii         | Pupukea      |
| 10      | user10    | user10    | user10@gmail.com  | US      | Washington     | Bellingham   |
+---------+-----------+-----------+-------------------+---------+----------------+--------------+
10 rows selected (0.099 seconds)
0: jdbc:phoenix:localhost>
```