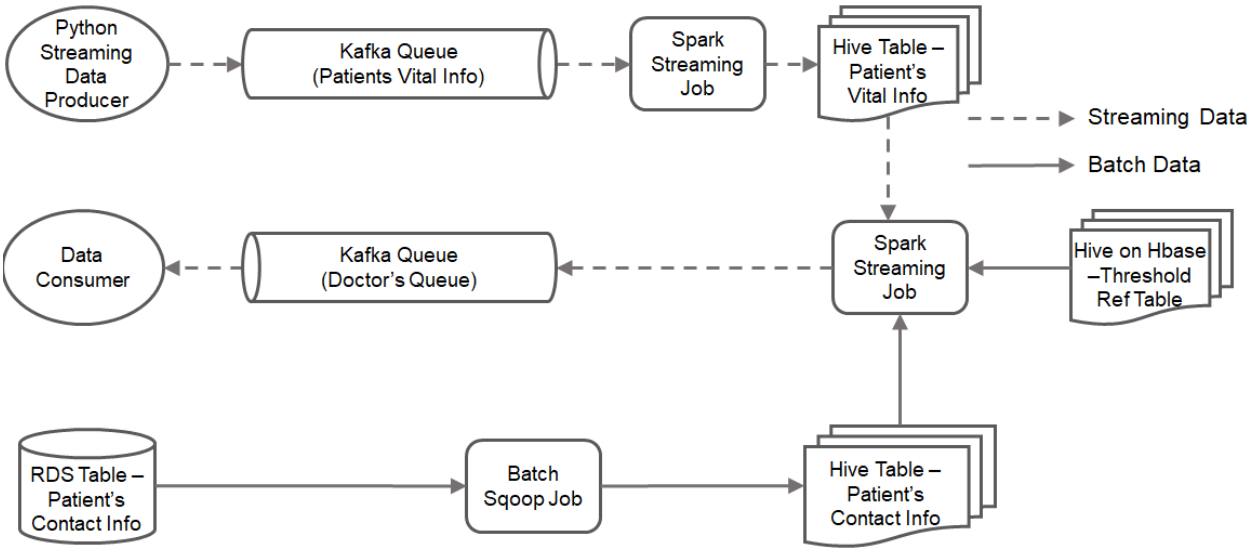
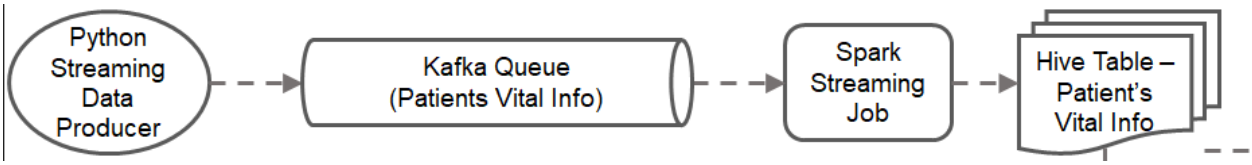
Complete Project Workflow:





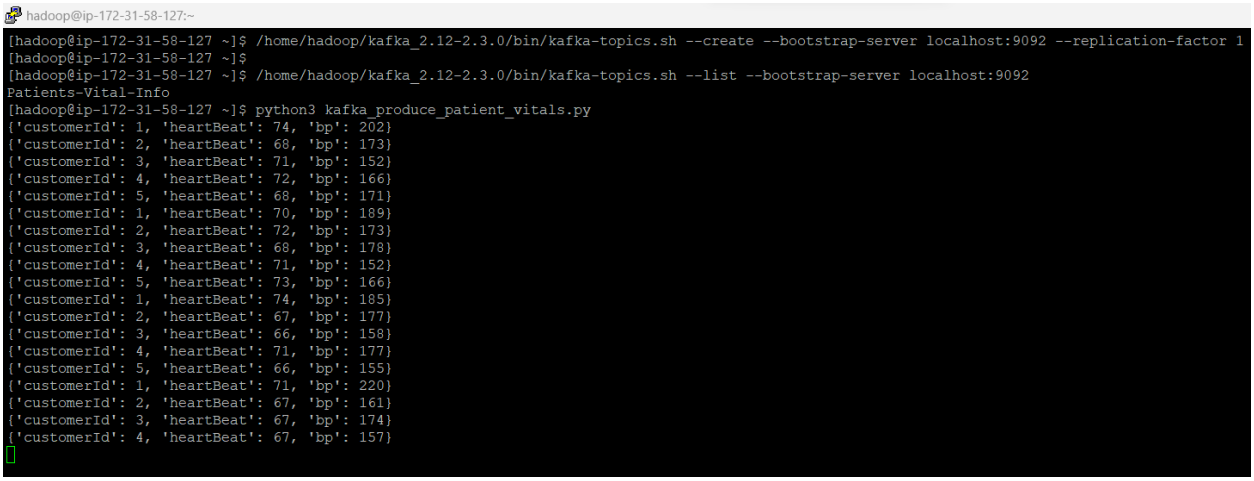
1. First start Kafka Server on default cluster master node

2. Create a Kafka topic **Patients-Vital-Info** on master node with default settings and check topic creation

   ***** Kafka server starting and topic creation command to be executed from **kafka.pdf**

3. Execute python script to push patient vital info into Kafka queue. The script will read from rds database table **patients_vital_info** and publish 1 row at interval of 1 second to above named kafka topic which will simulate a kafka stream.

   python3 kafka_produce_patient_vitals.py



4. Next, Spark-Streaming Job will be submitted to consume message from the topic **Patients-Vital-Info**
   It will initiate a spark session, subscribe and read raw stream from the kafka topic and convert to structured dataframe using defined schema and then stored into hdfs in **parquet** format in defined location **/user/hadoop/health-alert/patients-vital-info/**

   export SPARK_KAFKA_VERSION=0.10

   spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.11:2.4.5 kafka_spark_patient_vitals.py

   In each batch, a parquet file will be storing in the location. To check the storing of the streaming data to file:

   hadoop fs -ls /user/hadoop/health-alert/patients-vital-info/

```
+-----------+---------+---+-----------------------+

-------------------------------------------
Batch: 3
-------------------------------------------
+-----------+---------+---+-----------------------+
|customerId|heartBeat|bp |message_time           |
+-----------+---------+---+-----------------------+
|1          |70       |211|2023-02-02 15:20:30.003|
|2          |72       |163|2023-02-02 15:20:30.003|
|3          |56       |171|2023-02-02 15:20:30.003|
|4          |66       |157|2023-02-02 15:20:30.003|
|5          |73       |166|2023-02-02 15:20:30.003|
|1          |71       |210|2023-02-02 15:20:30.003|
|2          |67       |161|2023-02-02 15:20:30.003|
|3          |71       |160|2023-02-02 15:20:30.003|
|4          |71       |157|2023-02-02 15:20:30.003|
|5          |67       |171|2023-02-02 15:20:30.003|
+-----------+---------+---+-----------------------+

-------------------------------------------
Batch: 4
-------------------------------------------
+-----------+---------+---+-----------------------+
|customerId|heartBeat|bp |message_time           |
+-----------+---------+---+-----------------------+
|1          |77       |183|2023-02-02 15:20:40.003|
|2          |69       |151|2023-02-02 15:20:40.003|
|3          |68       |175|2023-02-02 15:20:40.003|
|4          |69       |176|2023-02-02 15:20:40.003|
|5          |71       |166|2023-02-02 15:20:40.003|
|1          |71       |177|2023-02-02 15:20:40.003|
|2          |72       |162|2023-02-02 15:20:40.003|
|3          |67       |159|2023-02-02 15:20:40.003|
|4          |69       |171|2023-02-02 15:20:40.003|
|5          |67       |169|2023-02-02 15:20:40.003|
+-----------+---------+---+-----------------------+
```

```
hadoop@ip-172-31-58-127:~
[hadoop@ip-172-31-58-127 ~]$ hadoop fs -ls /user/hadoop/health-alert/patients-vital-info/
Found 13 items
drwxr-xr-x   - hadoop hadoop          0 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/_spark_metadata
-rw-r--r--   1 hadoop hadoop       1294 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-051f5d92-f290-4b20-b5c1-01e2c2a78dde-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1294 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-103e2202-4787-409c-93cc-5f8ff5de8f39-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1589 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-2c8e442f-f16d-4955-a44c-71aa54d89b3c-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1290 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-6c5b680c-6028-4e85-bad8-102cf771bb48-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1290 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-89525018-6a24-4676-9031-c74adc634c85-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1282 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-9476b70e-db58-4103-a78b-693ebe003083-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1290 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-95d72dd9-93cc-47d8-8dc4-c8392c024686-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1290 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-a3d716d5-9995-43e6-a79a-4730655f5f8f-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1286 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-c1984a6f-07e2-44a0-867a-143f7c5f7e9b-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1211 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-dc6a81de-4eeb-4f9d-b972-ccdd31a9b2b2-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1286 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-efb0bc2c-4b4b-441b-bb4e-afba61e1db95-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1294 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-f544ebe4-d9ae-44d4-afdc-efe059d0f1d6-c000.snappy.parquet
[hadoop@ip-172-31-58-127 ~]$ hadoop fs -ls /user/hadoop/health-alert/patients-vital-info/
Found 15 items
drwxr-xr-x   - hadoop hadoop          0 2023-02-02 15:22 /user/hadoop/health-alert/patients-vital-info/_spark_metadata
-rw-r--r--   1 hadoop hadoop       1294 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-051f5d92-f290-4b20-b5c1-01e2c2a78dde-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1294 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-103e2202-4787-409c-93cc-5f8ff5de8f39-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1589 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-2c8e442f-f16d-4955-a44c-71aa54d89b3c-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1286 2023-02-02 15:22 /user/hadoop/health-alert/patients-vital-info/part-00000-510b4c54-20e9-4c1c-9a62-12db6f3bb027-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1290 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-6c5b680c-6028-4e85-bad8-102cf771bb48-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1290 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-89525018-6a24-4676-9031-c74adc634c85-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1282 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-9476b70e-db58-4103-a78b-693ebe003083-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1290 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-95d72dd9-93cc-47d8-8dc4-c8392c024686-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1290 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-a3d716d5-9995-43e6-a79a-4730655f5f8f-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1286 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-c1984a6f-07e2-44a0-867a-143f7c5f7e9b-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1290 2023-02-02 15:22 /user/hadoop/health-alert/patients-vital-info/part-00000-cb9f25c6-6deb-4a96-8155-e44a4a488470-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1211 2023-02-02 15:20 /user/hadoop/health-alert/patients-vital-info/part-00000-dc6a81de-4eeb-4f9d-b972-ccdd31a9b2b2-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1286 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-efb0bc2c-4b4b-441b-bb4e-afba61e1db95-c000.snappy.parquet
-rw-r--r--   1 hadoop hadoop       1294 2023-02-02 15:21 /user/hadoop/health-alert/patients-vital-info/part-00000-f544ebe4-d9ae-44d4-afdc-efe059d0f1d6-c000.snappy.parquet
[hadoop@ip-172-31-58-127 ~]$
```

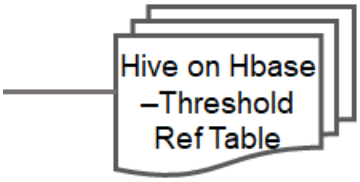
5. Next, we will create external hive table **patients_vital_info** and load data from the above hdfs locaton: **/user/hadoop/health-alert/patients-vital-info/**

   a. Creation of **healthdb**
   b. Creation of table **patients_vital_info** on that database
   c. Select and view the data.

   > To execute script **from hive1.pdf** in hive shell

   hive table creation and view data

```
Time taken: 0.081 seconds
hive (healthdb)>
              > CREATE EXTERNAL TABLE healthdb.patients_vital_info (
              > customerId int,
              > heartBeat int,
              > bp int,
              > message_time timestamp)
              > STORED AS PARQUET
              > LOCATION '/user/hadoop/health-alert/patients-vital-info/'
              > TBLPROPERTIES ('parquet.compress'='SNAPPY');
OK
Time taken: 0.694 seconds
hive (healthdb)>
              > select * from healthdb.patients_vital_info limit;
OK
limit.customerid        limit.heartbeat limit.bp        limit.message_time
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
1       78      185     2023-02-02 15:23:10.002
2       73      162     2023-02-02 15:23:10.002
3       69      169     2023-02-02 15:23:10.002
4       70      168     2023-02-02 15:23:10.002
5       73      179     2023-02-02 15:23:10.002
1       74      201     2023-02-02 15:23:10.002
2       70      160     2023-02-02 15:23:10.002
3       67      177     2023-02-02 15:23:10.002
4       67      179     2023-02-02 15:23:10.002
1       70      211     2023-02-02 15:20:30.003
2       72      163     2023-02-02 15:20:30.003
3       56      171     2023-02-02 15:20:30.003
4       66      157     2023-02-02 15:20:30.003
5       73      166     2023-02-02 15:20:30.003
1       71      210     2023-02-02 15:20:30.003
2       67      161     2023-02-02 15:20:30.003
3       71      160     2023-02-02 15:20:30.003
4       71      157     2023-02-02 15:20:30.003
5       67      171     2023-02-02 15:20:30.003
```



6. Next, provided threshold data for comparison to be stored in a hbase table **threshold_ref_hbase** and then create a hbase on hive table **threshold_ref_hive** which will be required to be access from spark.

   a. create a hbase table **threshold_ref_hbase**
   b. insert threshold data in the table
   c. query to verity the inserted data

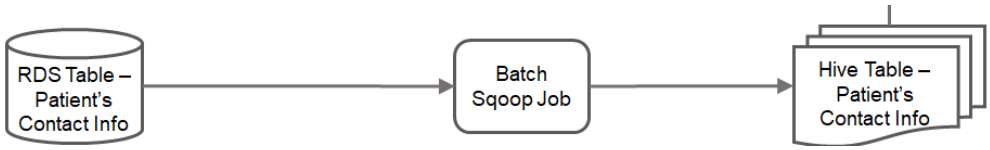
   > run script from **hbase.pdf** in hive shell

   hbase table creation and view inserted data

```
hbase(main):099:0> put 'threshold_ref_hbase', '12', 'msg:alert_flag', '1'
0 row(s) in 0.0030 seconds

hbase(main):100:0> put 'threshold_ref_hbase', '12', 'msg:alert_message', 'Higher BP than Normal'
0 row(s) in 0.0030 seconds

hbase(main):101:0>
hbase(main):102:0* scan 'threshold_ref_hbase'
ROW                                             COLUMN+CELL
 1                                              column=attr:attribute, timestamp=1675351589567, value=heartBeat
 1                                              column=limit:high_age_limit, timestamp=1675351589653, value=40
 1                                              column=limit:high_value, timestamp=1675351589704, value=69
 1                                              column=limit:low_age_limit, timestamp=1675351589616, value=0
 1                                              column=limit:low_value, timestamp=1675351589683, value=0
 1                                              column=msg:alert_flag, timestamp=1675351589726, value=1
 1                                              column=msg:alert_message, timestamp=1675351589753, value=Low Heart Rate than Normal
 10                                             column=attr:attribute, timestamp=1675351591464, value=bp
 10                                             column=limit:high_age_limit, timestamp=1675351591492, value=100
 10                                             column=limit:high_value, timestamp=1675351591517, value=150
 10                                             column=limit:low_age_limit, timestamp=1675351591480, value=41
 10                                             column=limit:low_value, timestamp=1675351591505, value=0
 10                                             column=msg:alert_flag, timestamp=1675351591532, value=1
 10                                             column=msg:alert_message, timestamp=1675351591546, value=Low BP than Normal
 11                                             column=attr:attribute, timestamp=1675351591572, value=bp
 11                                             column=limit:high_age_limit, timestamp=1675351591603, value=100
```



7. Next, we will run a sqoop import job which will import from rds table **patients_information** and store in hdfs **locatiion /user/hadoop/health-alert/patients-contact-info** and subsequently load it in external hive table **patients_contact_info**

> To import, run script from **sqoop.pdf** in command shell

```
        Map-Reduce Framework
                Map input records=5
                Map output records=5
                Input split bytes=87
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=103
                CPU time spent (ms)=2150
                Physical memory (bytes) snapshot=267186176
                Virtual memory (bytes) snapshot=3280048128
                Total committed heap usage (bytes)=239075328
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=220
23/02/02 15:29:36 INFO mapreduce.ImportJobBase: Transferred 220 bytes in 17.8755 seconds (12.3074 bytes/sec)
23/02/02 15:29:36 INFO mapreduce.ImportJobBase: Retrieved 5 records.
23/02/02 15:29:36 INFO util.AppendUtils: Creating missing output directory - patients-contact-info
23/02/02 15:29:36 INFO tool.ImportTool: Saving incremental import state to the metastore
23/02/02 15:29:36 INFO tool.ImportTool: Updated data for job: patientcontactimport
[hadoop@ip-172-31-58-127 ~]$
[hadoop@ip-172-31-58-127 ~]$ hadoop fs -ls /user/hadoop/health-alert/patients-contact-info/
Found 1 items
-rw-r--r--   1 hadoop hadoop        220 2023-02-02 15:29 /user/hadoop/health-alert/patients-contact-info/part-m-00000
[hadoop@ip-172-31-58-127 ~]$
```

8. Next, to load **patients_contact_info** from **/user/hadoop/health-alert/patients-contact-info/**
and create hbase on hive for **threshold_ref_hbase - threshold_ref_hive**

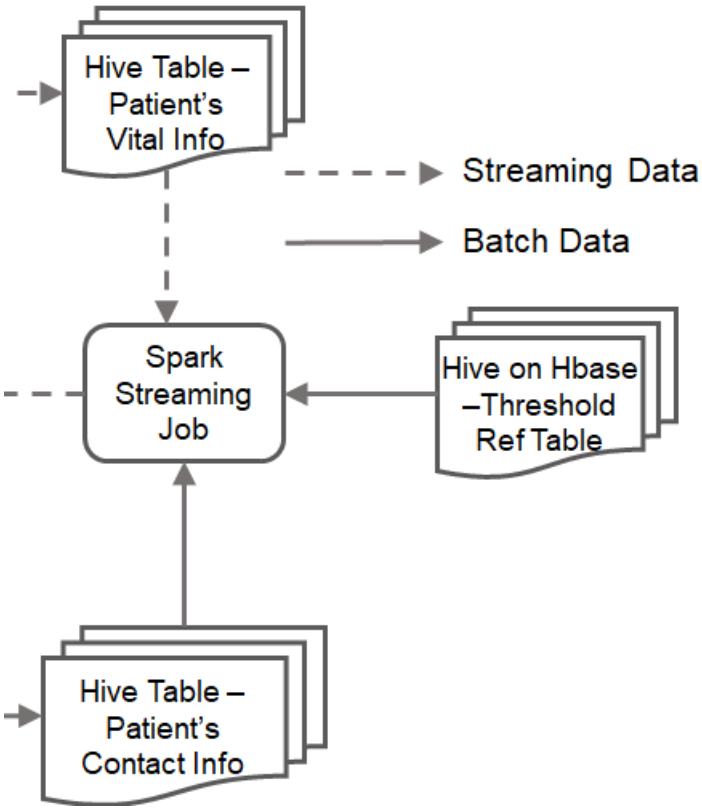
> run script from **hive2.pdf** in hive shell

Hive table for contact info and threshold table hbase on hive creation and view data

```
hive>
    > use healthdb;
OK
Time taken: 0.613 seconds
hive>
    > CREATE EXTERNAL TABLE healthdb.patients_contact_info(
    >    patientid int,
    >    patientname varchar(255),
    >    patientaddress varchar(255),
    >    phone_number varchar(255),
    >    admitted_ward int,
    >    age int,
    >    other_details varchar(255)
    >  )
    > ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
    > LINES TERMINATED BY '\n'
    > STORED AS TEXTFILE
    > LOCATION '/user/hadoop/health-alert/patients-contact-info/' ;
OK
Time taken: 0.459 seconds
hive>
    > CREATE EXTERNAL TABLE healthdb.threshold_ref_hive(ref_id int, attribute varchar(20), low_age_limit int, high_age_limit int,
alert_message varchar(255))
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,attr:attribute,limit:low_age_limit,limit:high_age_limit,limit:low_valu
    > TBLPROPERTIES ("hbase.table.name" = "threshold_ref_hbase");
OK
Time taken: 1.913 seconds
hive>
```

```
hive>
    > select * from healthdb.patients_contact_info;
OK
1       Alex S  XDC test Address        8982739282      1       23      NULL
2       Sammy A New Building Address    2382739282      2       45      NULL
3       Karan C Aws Address     8923739282      3       56      NULL
4       Dara M  India Address   2182739282      4       67      NULL
5       Pam     ABC test Address        4982739282      5       72      NULL
Time taken: 2.418 seconds, Fetched: 5 row(s)
hive>
    > select * from healthdb.threshold_ref_hive;
OK
1       heartBeat       0       40      0       69      1       Low Heart Rate than Normal
10      bp      41      100     0       150     1       Low BP than Normal
11      bp      41      100     151     180     0       Normal
12      bp      41      100     181     9999    1       Higher BP than Normal
2       heartBeat       0       40      70      78      0       Normal
3       heartBeat       0       40      79      9999    1       Higher Heart Rate than Normal
4       bp      0       40      0       160     1       Low BP than Normal
5       bp      0       40      161     220     0       Normal
6       bp      0       40      221     9999    1       Higher BP than Normal
7       heartBeat       41      100     0       65      1       Low Heart Rate than Normal
8       heartBeat       41      100     66      73      0       Normal
9       heartBeat       41      100     74      9999    1       Higher Heart Rate than Normal
Time taken: 0.474 seconds, Fetched: 12 row(s)
hive>
    >
```

Comparison of vital info steam and threshold data and generate stream for alert notification



9. Next we will run a spark job to which will first get patient information and then compare with reference data with vital info to get status of the health reading. If comparison return status flag 1 which indicates there is abnormality and only those streaming data will be pushed to another topic as alert message.

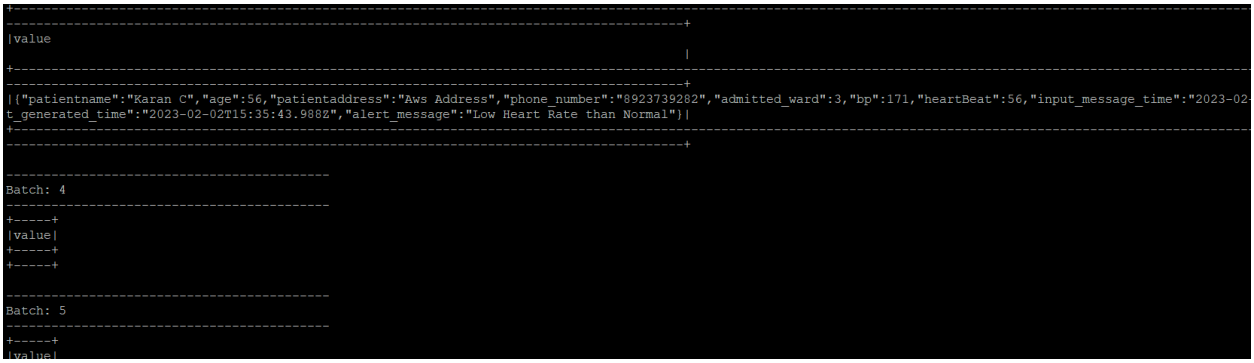
export SPARK_KAFKA_VERSION=0.10

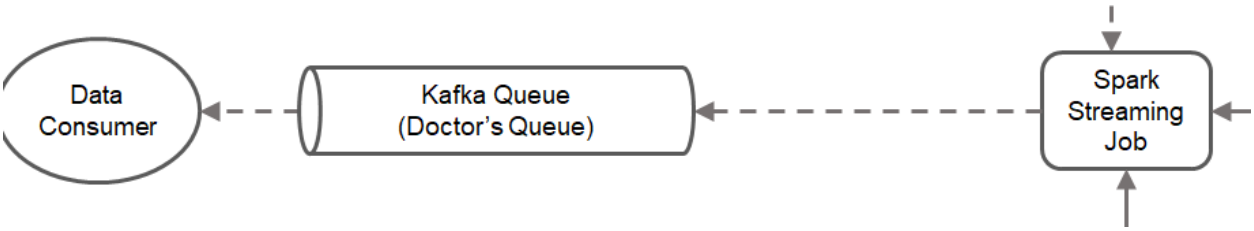
spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.11:2.4.5 kafka_spark_generate_alerts.py

The spark-streaming job script will create spark session, read stream **/user/hadoop/health-alert/patients-vital-info/** next the steam will be joined with static dataframe obtained from hive table **patients_contact_info** using spark sql to get patient contact information and then compared with static dataframe obtained from hbase on hive table

threshold_ref_hive and pick alert stream based on status flag. The stream then converted json format to publish to topic **Health-Alert-Messages.**

Publishing of json message to topic Health-Alert-Messages





10. Next and final task will be consuming from the topic **Health-Alert-Messages** and publish it SNS topic subscriber to send alert message through email endpoint.

* SNS topic **Health-Alert-Notification** and subscriber to be created and configured for email endpoint before publishing to SNS.

* SNS topic and subscriber creation and configuration process described in **sns.pdf**

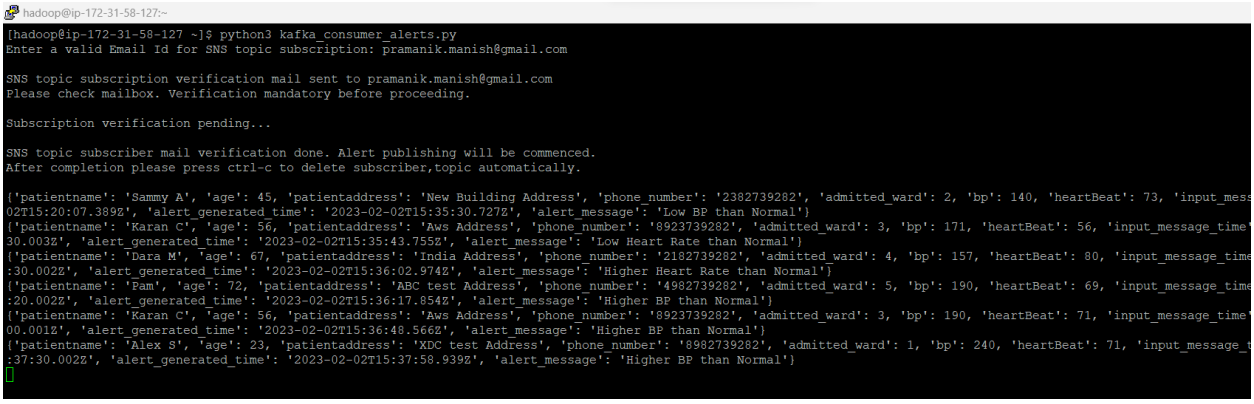
** Alternatively, here we will not create topic and subscriber manually. Instead, will initiate topic and subscriber programmatically and when proceeding done, force exit (Ctrl+C) which will also delete subscriber and topic programmatically which will avoid any manual SNS activity.

To publish Alert Messages to SNS topic-subscriber, execute script

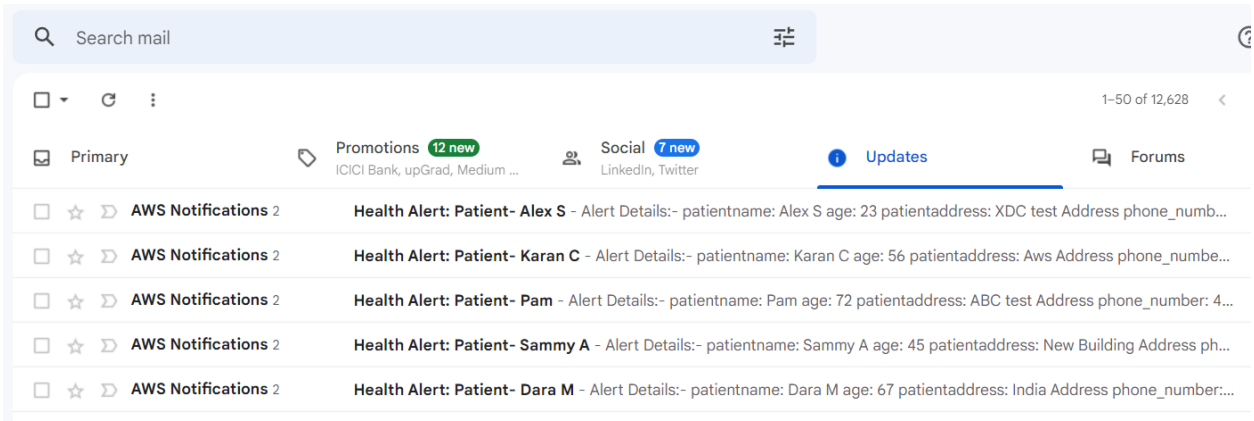
python3 kafka_consumer_alerts.py

N.B- When prompt, a valid and active Email ID advised to be entered.

Consumption of alert message and publish to SNS topic subscriber



Health Alert Notification mail in inbox:



Health Alert Notification mail format: