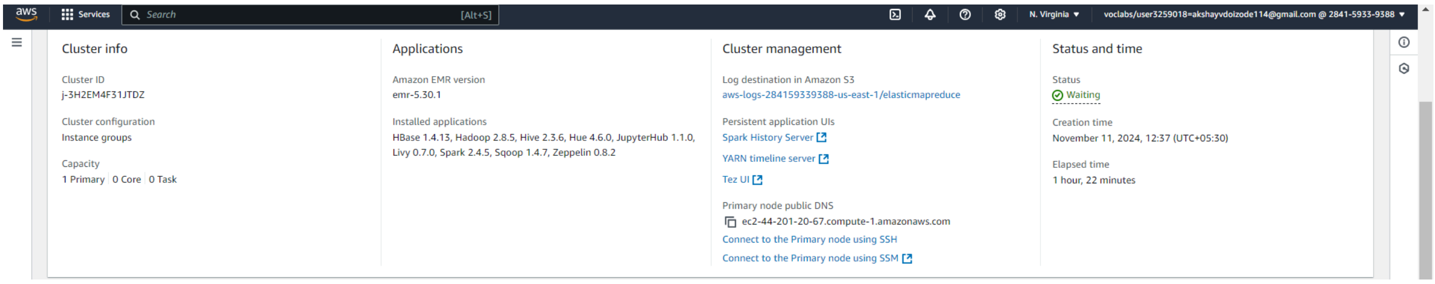
Capstone Project: Instant Health Alert System - Final-Project Submission

EMR instances

- **EMR cluster** *(with Spark, Hive, Sqoop,Hbase)*



aws
Services
Search [Alt+S]
N. Virginia
voclabs/user3259018=akshayvdoizode114@gmail.com @ 2841-5933-9388

Cluster info

Cluster ID
j-3H2EM4F31JTDZ

Cluster configuration
Instance groups

Capacity
1 Primary | 0 Core | 0 Task

Applications

Amazon EMR version
emr-5.30.1

Installed applications
HBase 1.4.13, Hadoop 2.8.5, Hive 2.3.6, Hue 4.6.0, JupyterHub 1.1.0, Livy 0.7.0, Spark 2.4.5, Sqoop 1.4.7, Zeppelin 0.8.2

Cluster management

Log destination in Amazon S3
aws-logs-284159339388-us-east-1/elasticmapreduce

Persistent application UIs
Spark History Server
YARN timeline server
Tez UI

Primary node public DNS
ec2-44-201-20-67.compute-1.amazonaws.com
Connect to the Primary node using SSH
Connect to the Primary node using SSM

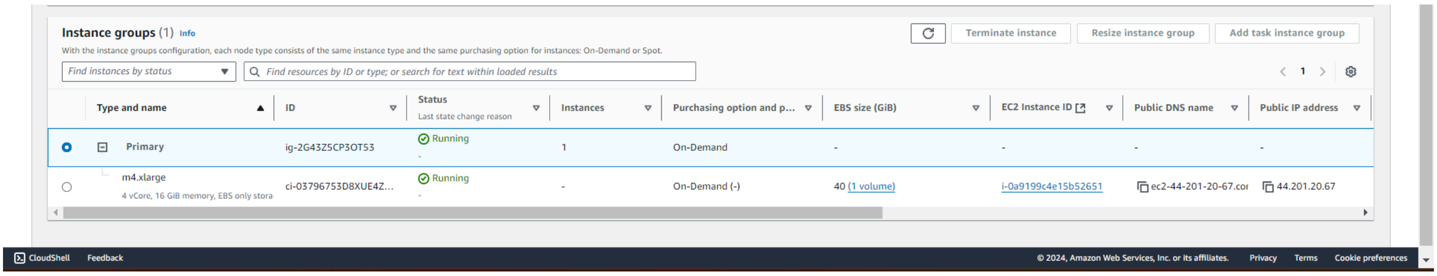
Status and time

Status
Waiting

Creation time
November 11, 2024, 12:37 (UTC+05:30)

Elapsed time
1 hour, 22 minutes

EMR Hardware Configuration *(with 1 master node)*



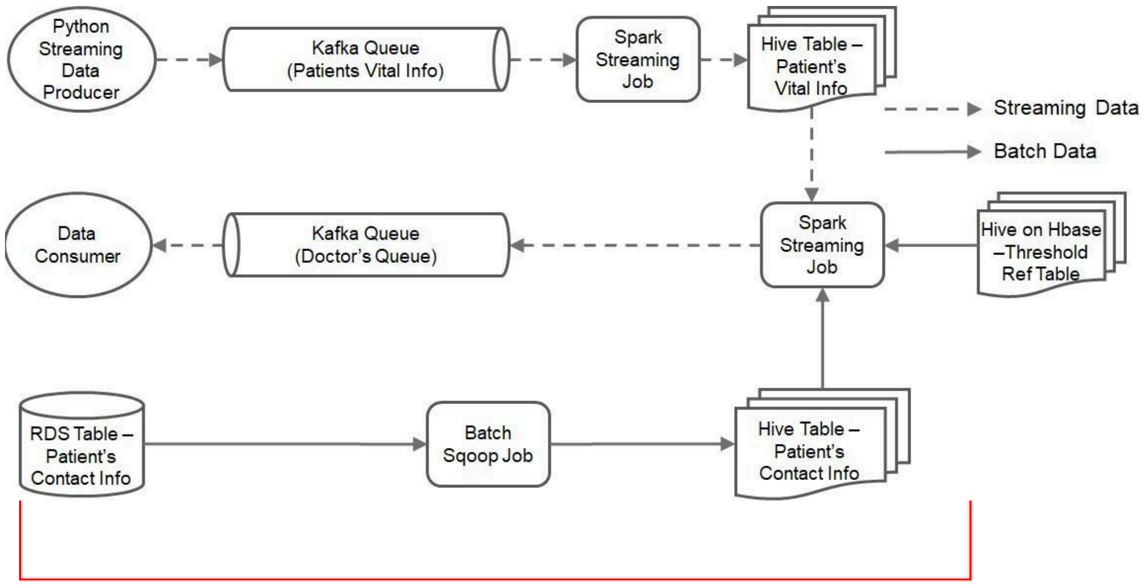
Instance groups (1) Info

With the instance groups configuration, each node type consists of the same instance type and the same purchasing option for instances: On-Demand or Spot.

	Type and name	ID	Status Last state change reason	Instances	Purchasing option and p...	EBS size (GiB)	EC2 Instance ID	Public DNS name	Public IP address
●	Primary	ig-2G43Z5CP3OT53	Running -	1	On-Demand	-	-	-	-
○	m4.xlarge 4 vCore, 16 GiB memory, EBS only stora	ci-03796753D8XUE4Z...	Running -	-	On-Demand (-)	40 (1 volume)	i-0a9199c4e15b52651	ec2-44-201-20-67.coı	44.201.20.67

PART 1:

We will import patient contact information from the RDS using Sqoop and load the data into a Hive table for further processing.



Python Streaming Data Producer
Kafka Queue (Patients Vital Info)
Spark Streaming Job
Hive Table – Patient's Vital Info
Streaming Data
Batch Data
Data Consumer
Kafka Queue (Doctor's Queue)
Spark Streaming Job
Hive on Hbase –Threshold Ref Table
RDS Table – Patient's Contact Info
Batch Sqoop Job
Hive Table – Patient's Contact Info

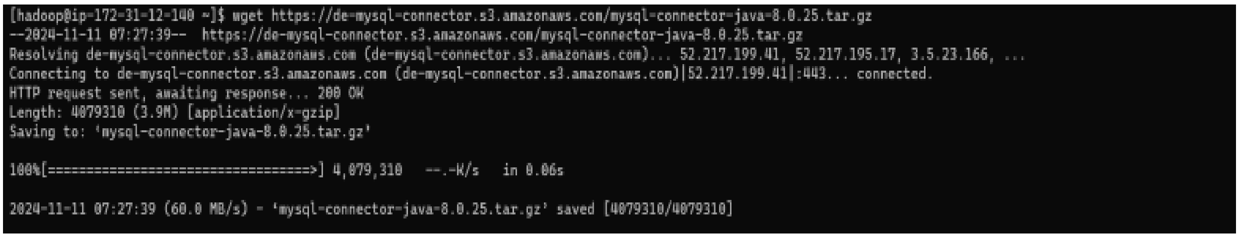
1. Import Patient Contact Info records to HDFS using Sqoop
2. Creating an external Hive table for storing Patient's Contact Info

SQOOP SETUP

Following steps are followed to setup Sqoop on EMR Cluster

1. To install the MySQL connector jar file.

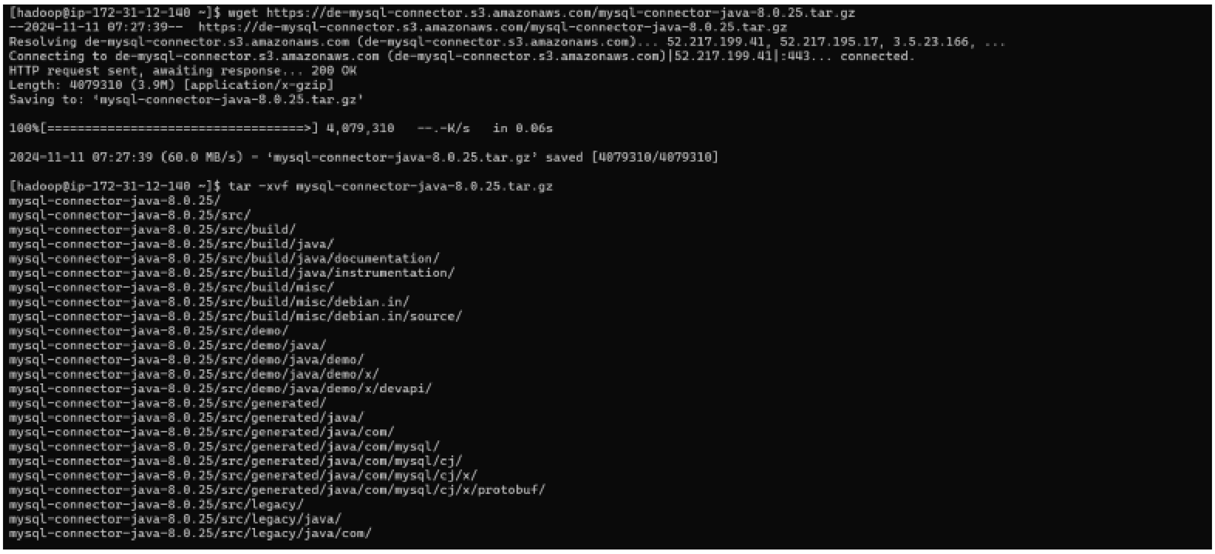
```
wget https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
```



```
[hadoop@ip-172-31-12-140 ~]$ wget https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
--2024-11-11 07:27:39--  https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
Resolving de-mysql-connector.s3.amazonaws.com (de-mysql-connector.s3.amazonaws.com)... 52.217.199.41, 52.217.195.17, 3.5.23.166, ...
Connecting to de-mysql-connector.s3.amazonaws.com (de-mysql-connector.s3.amazonaws.com)|52.217.199.41|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4079310 (3.9M) [application/x-gzip]
Saving to: 'mysql-connector-java-8.0.25.tar.gz'

100%[==================================>] 4,079,310   --.-K/s   in 0.06s

2024-11-11 07:27:39 (60.0 MB/s) - 'mysql-connector-java-8.0.25.tar.gz' saved [4079310/4079310]
```

2. Extract the MySQL connector tar file

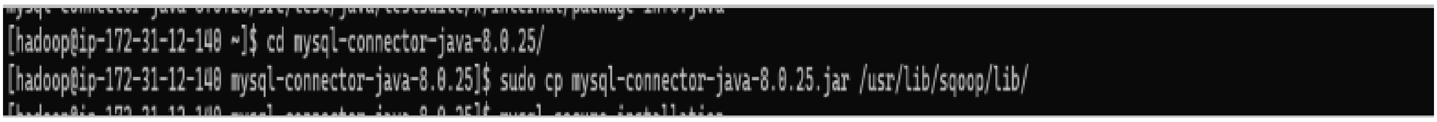
```
tar -xvf mysql-connector-java-8.0.25.tar.gz
```



```
[hadoop@ip-172-31-12-140 ~]$ wget https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
--2024-11-11 07:27:39--  https://de-mysql-connector.s3.amazonaws.com/mysql-connector-java-8.0.25.tar.gz
Resolving de-mysql-connector.s3.amazonaws.com (de-mysql-connector.s3.amazonaws.com)... 52.217.199.41, 52.217.195.17, 3.5.23.166, ...
Connecting to de-mysql-connector.s3.amazonaws.com (de-mysql-connector.s3.amazonaws.com)|52.217.199.41|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4079310 (3.9M) [application/x-gzip]
Saving to: 'mysql-connector-java-8.0.25.tar.gz'

100%[==================================>] 4,079,310   --.-K/s   in 0.06s

2024-11-11 07:27:39 (60.0 MB/s) - 'mysql-connector-java-8.0.25.tar.gz' saved [4079310/4079310]

[hadoop@ip-172-31-12-140 ~]$ tar -xvf mysql-connector-java-8.0.25.tar.gz
mysql-connector-java-8.0.25/
mysql-connector-java-8.0.25/src/
mysql-connector-java-8.0.25/src/build/
mysql-connector-java-8.0.25/src/build/java/
mysql-connector-java-8.0.25/src/build/java/documentation/
mysql-connector-java-8.0.25/src/build/java/instrumentation/
mysql-connector-java-8.0.25/src/build/misc/
mysql-connector-java-8.0.25/src/build/misc/debian.in/
mysql-connector-java-8.0.25/src/build/misc/debian.in/source/
mysql-connector-java-8.0.25/src/demo/
mysql-connector-java-8.0.25/src/demo/java/
mysql-connector-java-8.0.25/src/demo/java/demo/
mysql-connector-java-8.0.25/src/demo/java/demo/x/
mysql-connector-java-8.0.25/src/demo/java/demo/x/devapi/
mysql-connector-java-8.0.25/src/generated/
mysql-connector-java-8.0.25/src/generated/java/
mysql-connector-java-8.0.25/src/generated/java/com/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/cj/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/cj/x/
mysql-connector-java-8.0.25/src/generated/java/com/mysql/cj/x/protobuf/
mysql-connector-java-8.0.25/src/legacy/
mysql-connector-java-8.0.25/src/legacy/java/
mysql-connector-java-8.0.25/src/legacy/java/com/
```

3. Navigate to the MySQL Connector directory created in the previous step, and copy it to the Sqoop library to complete the installation.

```
cd mysql-connector-java-8.0.25/
sudo cp mysql-connector-java-8.0.25.jar /usr/lib/sqoop/lib/
```

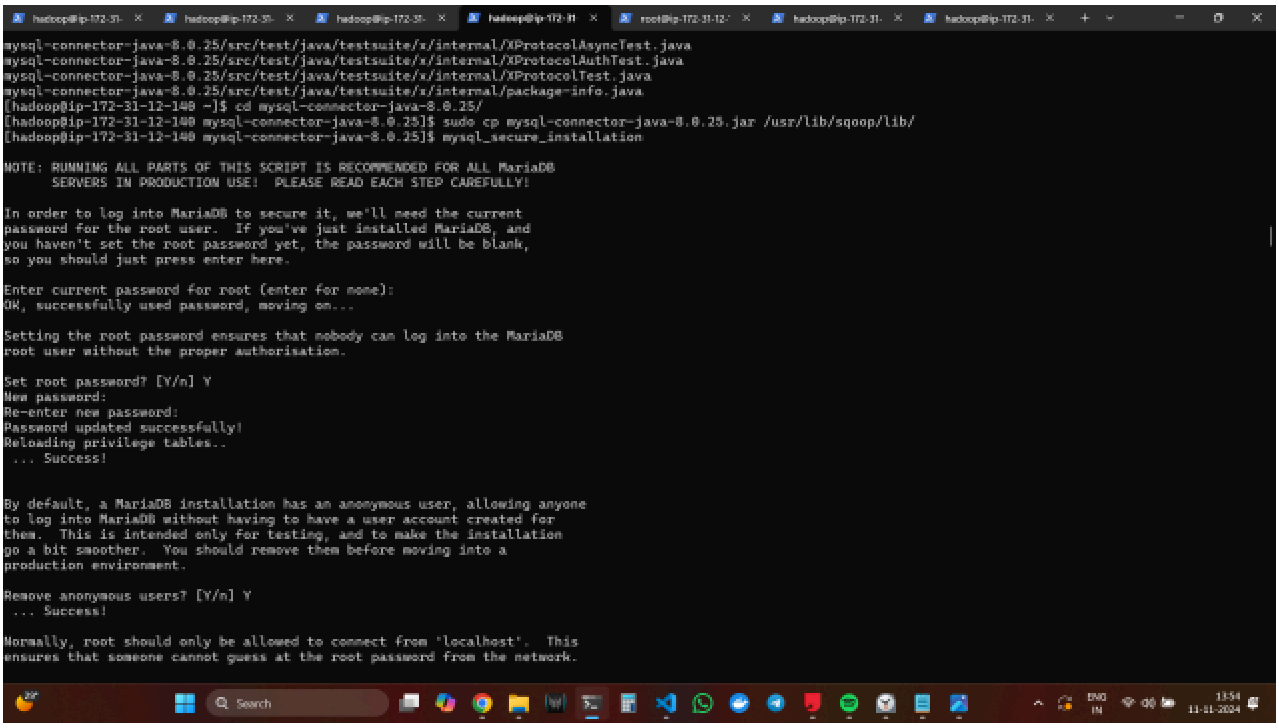


```
[hadoop@ip-172-31-12-140 ~]$ cd mysql-connector-java-8.0.25/
[hadoop@ip-172-31-12-140 mysql-connector-java-8.0.25]$ sudo cp mysql-connector-java-8.0.25.jar /usr/lib/sqoop/lib/
```

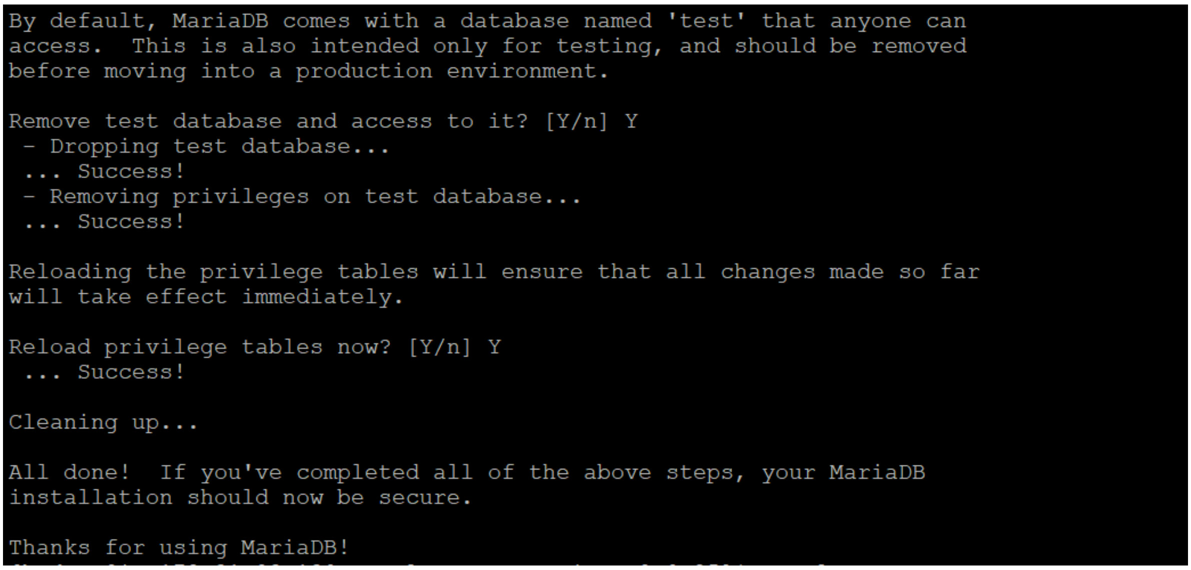
4. Set up MySQL on your EMR cluster (Inside this folder mysql-connector-java-8.0.25)

```
mysql_secure_installation
```

Enter current password for root (enter for none): ENTER
Set root password [Y/n] Y New password: 123
Re-enter password: 123
Remove anonymous users [Y/n] Y
Disallow root login remotely [Y/n] n
Remove test database and access to it [Y/n] Y
Reload privilege tables now [Y/n] Y



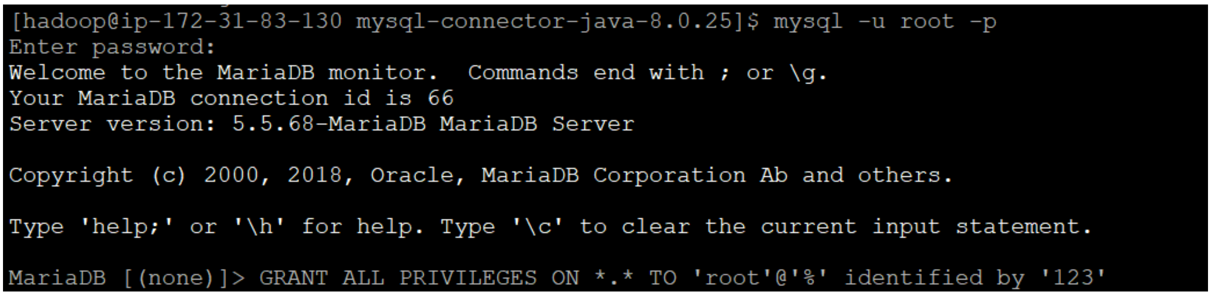
```
mysql-connector-java-8.0.25/src/test/java/testsuite/x/internal/XProtocolAsyncTest.java
mysql-connector-java-8.0.25/src/test/java/testsuite/x/internal/XProtocolAuthTest.java
mysql-connector-java-8.0.25/src/test/java/testsuite/x/internal/XProtocolTest.java
mysql-connector-java-8.0.25/src/test/java/testsuite/x/internal/package-info.java
[hadoop@ip-172-31-12-140 ~]$ cd mysql-connector-java-8.0.25/
[hadoop@ip-172-31-12-140 mysql-connector-java-8.0.25]$ sudo cp mysql-connector-java-8.0.25.jar /usr/lib/sqoop/lib/
[hadoop@ip-172-31-12-140 mysql-connector-java-8.0.25]$ mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] Y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] Y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.
```



```
By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] Y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] Y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

6. After this a prompt for password will be shown hit 123 and grant root privileges to the other user and restart mariaDB

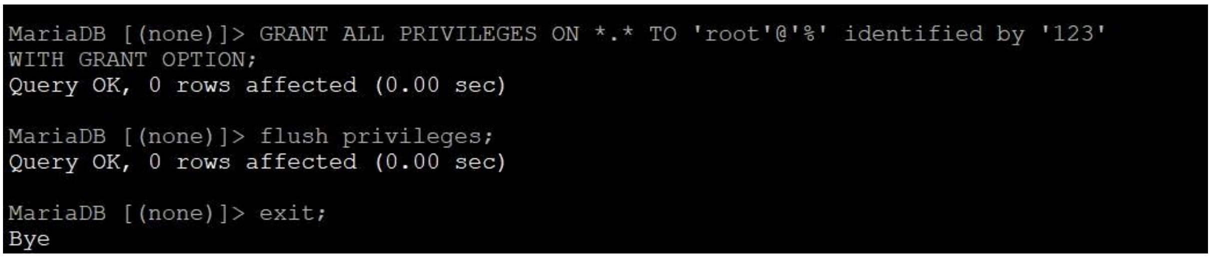
```
mysql -u root -p
```



```
[hadoop@ip-172-31-83-130 mysql-connector-java-8.0.25]$ mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 66
Server version: 5.5.68-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' identified by '123'
```

7. Inside MariaDB (MariaDB >)
Following queries need to be run for granting all privileges to the root user.

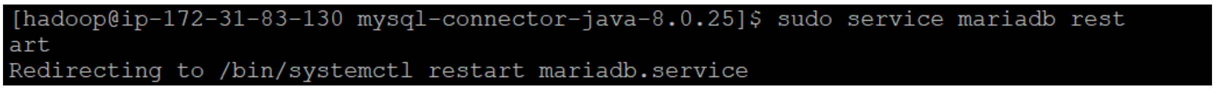
```
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' identified by '123' WITH GRANT OPTION;
flush privileges;
  exit;
```



```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' identified by '123'
WITH GRANT OPTION;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> exit;
Bye
```

8. Restart the MySQL service to finish setting up MySQL. (Inside this folder mysql-connector-java-8.0.25)

```
sudo service mariadb restart
```

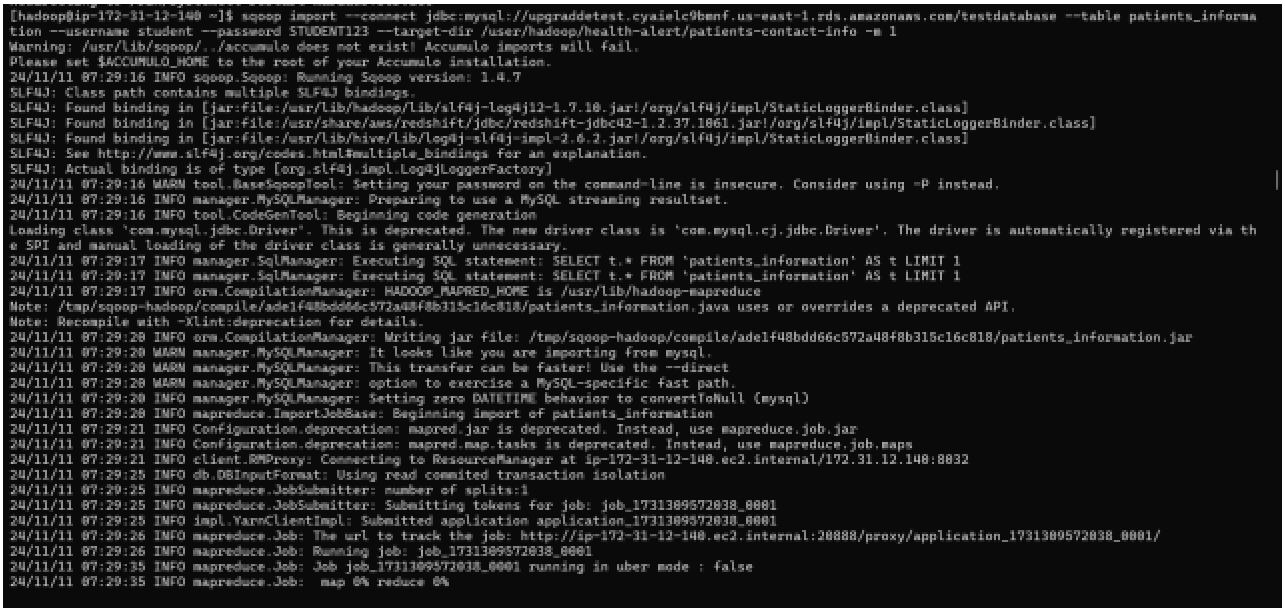


```
[hadoop@ip-172-31-83-130 mysql-connector-java-8.0.25]$ sudo service mariadb rest
art
Redirecting to /bin/systemctl restart mariadb.service
```

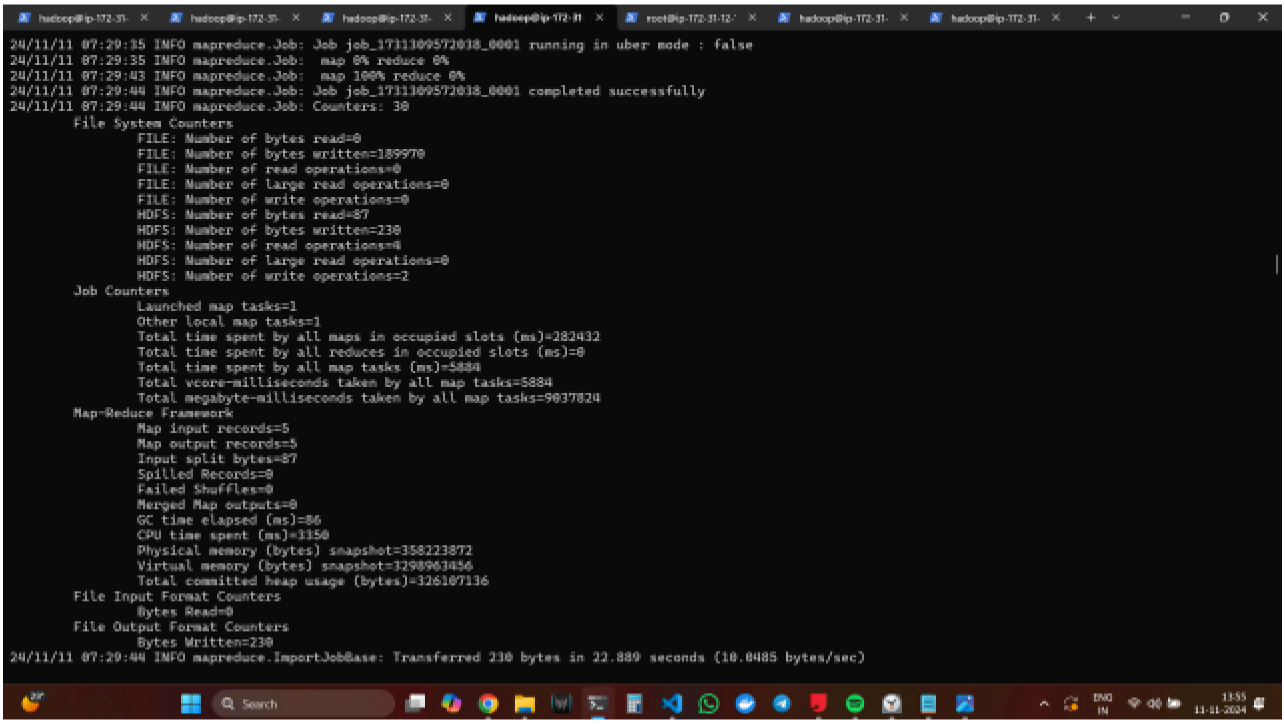
Sqoop Commands

1. Import data to HDFS

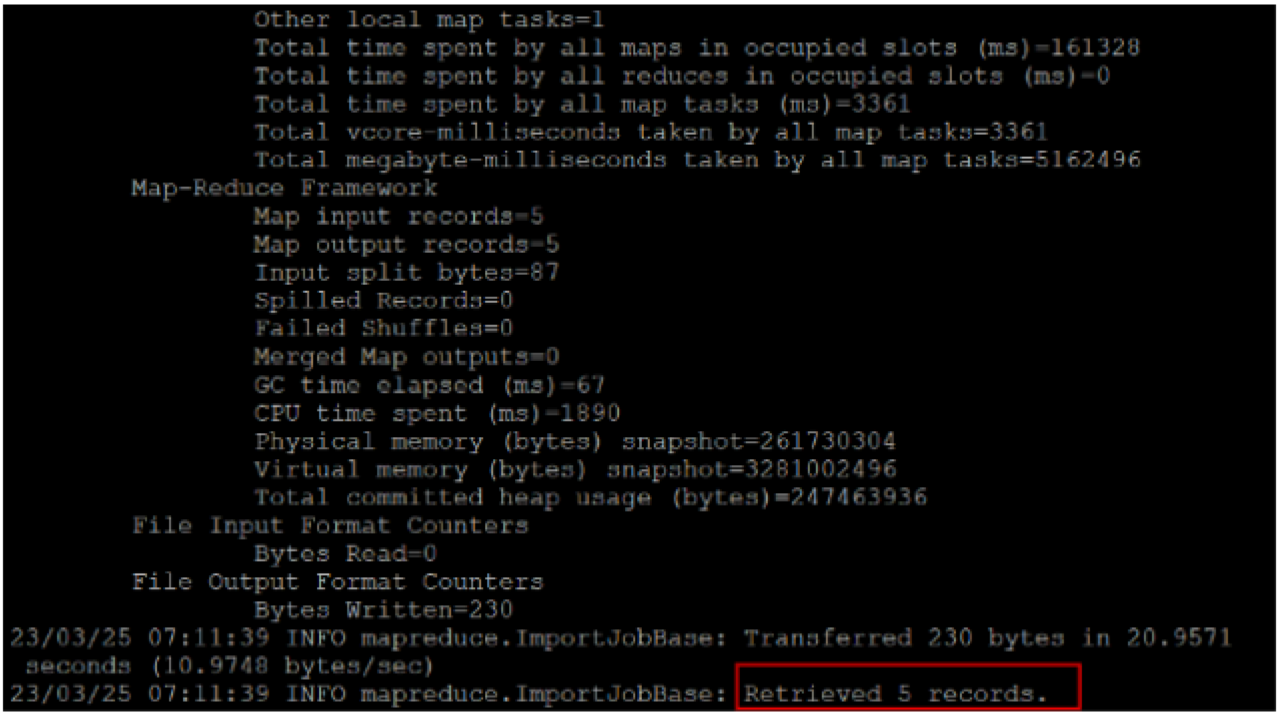
```
sqoop import --connect
  jdbc:mysql://upgraddetest.cyaielc9bmnf.us-east-1.rds.amazonaws.com/testdatabase --table
  patients_information --username student --password STUDENT123 --target-dir
  /user/hadoop/health-alert/patients-contact-info -m 1
```



```
[hadoop@ip-172-31-12-140 ~]$ sqoop import --connect jdbc:mysql://upgraddetest.cyaielc9bmnf.us-east-1.rds.amazonaws.com/testdatabase --table patients_informa
tion --username student --password STUDENT123 --target-dir /user/hadoop/health-alert/patients-contact-info -m 1
Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
24/11/11 07:29:16 INFO sqoop.Sqoop: Running Sqoop version: 1.4.7
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/share/aws/redshift/jdbc/redshift-jdbc42-1.2.37.1061.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
24/11/11 07:29:16 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
24/11/11 07:29:16 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset.
24/11/11 07:29:16 INFO tool.CodeGenTool: Beginning code generation
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered via th
e SPI and manual loading of the driver class is generally unnecessary.
24/11/11 07:29:17 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `patients_information` AS t LIMIT 1
24/11/11 07:29:17 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `patients_information` AS t LIMIT 1
24/11/11 07:29:17 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /usr/lib/hadoop-mapreduce
Note: /tmp/sqoop-hadoop/compile/ade1f48bdd66c572a48f8b315c16c818/patients_information.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
24/11/11 07:29:20 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-hadoop/compile/ade1f48bdd66c572a48f8b315c16c818/patients_information.jar
24/11/11 07:29:20 WARN manager.MySQLManager: It looks like you are importing from mysql.
24/11/11 07:29:20 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
24/11/11 07:29:20 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
24/11/11 07:29:20 INFO manager.MySQLManager: Setting zero DATETIME behavior to convertToNull (mysql)
24/11/11 07:29:20 INFO mapreduce.ImportJobBase: Beginning import of patients_information
24/11/11 07:29:21 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
24/11/11 07:29:21 INFO Configuration.deprecation: mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
24/11/11 07:29:21 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-12-140.ec2.internal/172.31.12.140:8032
24/11/11 07:29:25 INFO db.DBInputFormat: Using read commited transaction isolation
24/11/11 07:29:25 INFO mapreduce.JobSubmitter: number of splits:1
24/11/11 07:29:25 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1731309572038_0001
24/11/11 07:29:25 INFO impl.YarnClientImpl: Submitted application application_1731309572038_0001
24/11/11 07:29:26 INFO mapreduce.Job: The url to track the job: http://ip-172-31-12-140.ec2.internal:20888/proxy/application_1731309572038_0001/
24/11/11 07:29:26 INFO mapreduce.Job: Running job: job_1731309572038_0001
24/11/11 07:29:35 INFO mapreduce.Job: Job job_1731309572038_0001 running in uber mode : false
24/11/11 07:29:35 INFO mapreduce.Job:  map 0% reduce 0%
```



```
24/11/11 07:29:35 INFO mapreduce.Job: Job job_1731309572038_0001 running in uber mode : false
24/11/11 07:29:35 INFO mapreduce.Job:  map 0% reduce 0%
24/11/11 07:29:43 INFO mapreduce.Job:  map 100% reduce 0%
24/11/11 07:29:44 INFO mapreduce.Job: Job job_1731309572038_0001 completed successfully
24/11/11 07:29:44 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=189970
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=87
                HDFS: Number of bytes written=230
                HDFS: Number of read operations=4
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Other local map tasks=1
                Total time spent by all maps in occupied slots (ms)=282432
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=5884
                Total vcore-milliseconds taken by all map tasks=5884
                Total megabyte-milliseconds taken by all map tasks=9037824
        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)=86
                CPU time spent (ms)=3350
                Physical memory (bytes) snapshot=358223872
                Virtual memory (bytes) snapshot=3298963456
                Total committed heap usage (bytes)=326107136
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=230
24/11/11 07:29:44 INFO mapreduce.ImportJobBase: Transferred 230 bytes in 22.889 seconds (10.0485 bytes/sec)
```



```
                Other local map tasks=1
                Total time spent by all maps in occupied slots (ms)=161328
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=3361
                Total vcore-milliseconds taken by all map tasks=3361
                Total megabyte-milliseconds taken by all map tasks=5162496
        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)=67
                CPU time spent (ms)=1890
                Physical memory (bytes) snapshot=261730304
                Virtual memory (bytes) snapshot=3281002496
                Total committed heap usage (bytes)=247463936
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=230
23/03/25 07:11:39 INFO mapreduce.ImportJobBase: Transferred 230 bytes in 20.9571
 seconds (10.9748 bytes/sec)
23/03/25 07:11:39 INFO mapreduce.ImportJobBase: Retrieved 5 records.
```

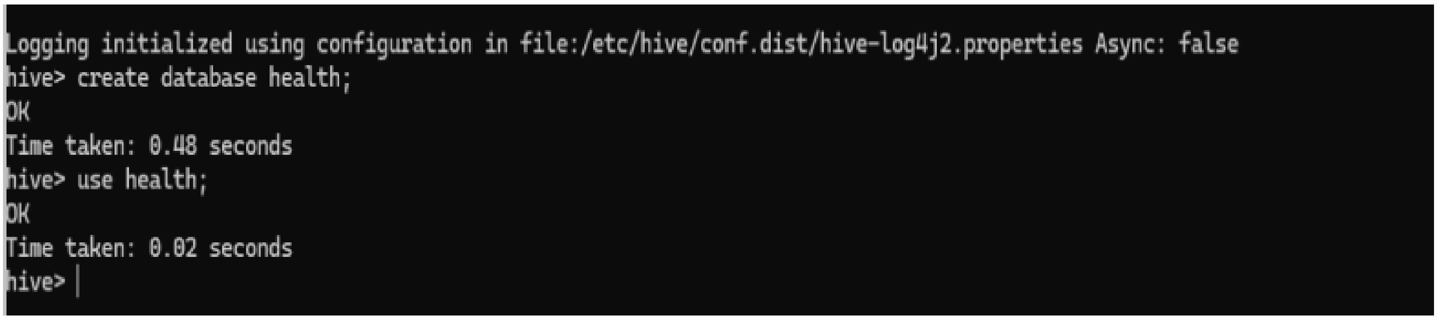
Hive table creation (for Patients_Contact_Info)

- Open Hive shell.
- Create a database ***health***

```
create database health;
```

- Use database patient_health_care

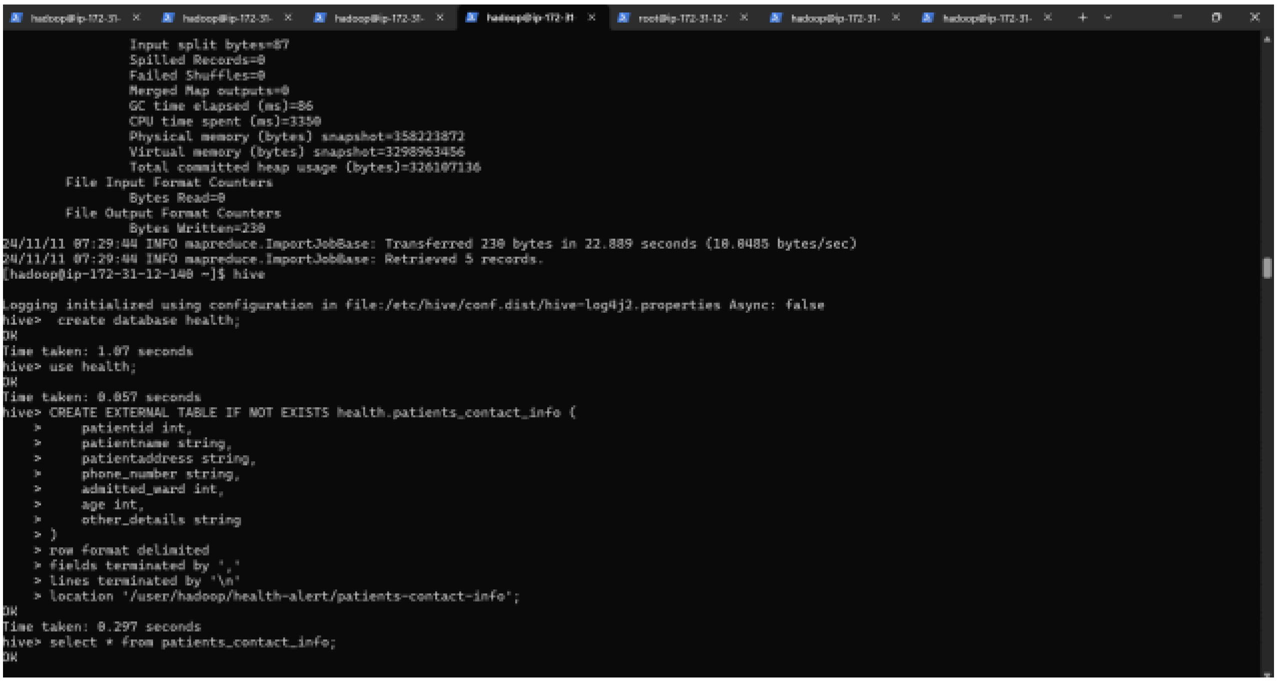
```
use health;
```



```
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> create database health;
OK
Time taken: 0.48 seconds
hive> use health;
OK
Time taken: 0.02 seconds
hive> |
```

- Create external table named ***Patients_Contact_Info***

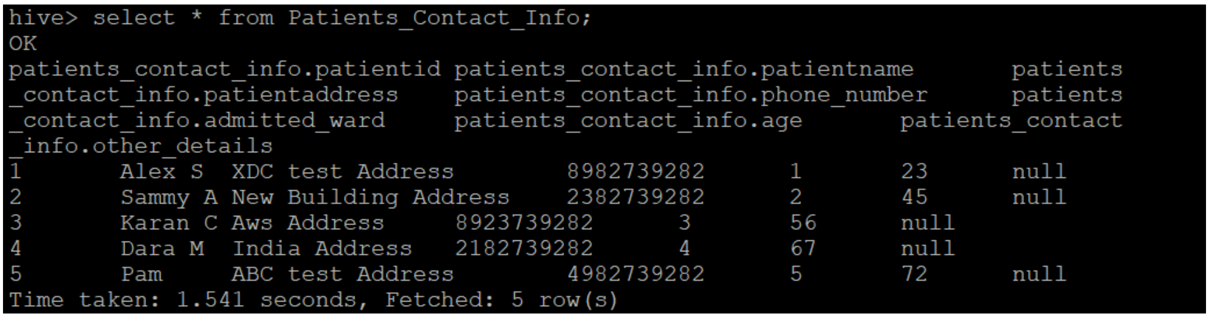
```
CREATE EXTERNAL TABLE IF NOT EXISTS health.patients_contact_info (
   patientid int,
   patientname string,
   patientaddress string,
   phone_number string,
   admitted_ward int,
   age int,
   other_details string
)
row format delimited
fields terminated by ','
lines terminated by '\n'
location '/user/hadoop/health-alert/patients-contact-info';
```



```
                Input split bytes=87
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=86
                CPU time spent (ms)=3350
                Physical memory (bytes) snapshot=358223872
                Virtual memory (bytes) snapshot=3298963456
                Total committed heap usage (bytes)=326107136
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=230
24/11/11 07:29:44 INFO mapreduce.ImportJobBase: Transferred 230 bytes in 22.889 seconds (10.0485 bytes/sec)
24/11/11 07:29:44 INFO mapreduce.ImportJobBase: Retrieved 5 records.
[hadoop@ip-172-31-12-140 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive>  create database health;
OK
Time taken: 1.07 seconds
hive> use health;
OK
Time taken: 0.057 seconds
hive> CREATE EXTERNAL TABLE IF NOT EXISTS health.patients_contact_info (
    >     patientid int,
    >     patientname string,
    >     patientaddress string,
    >     phone_number string,
    >     admitted_ward int,
    >     age int,
    >     other_details string
    > )
    > row format delimited
    > fields terminated by ','
    > lines terminated by '\n'
    > location '/user/hadoop/health-alert/patients-contact-info';
OK
Time taken: 0.297 seconds
hive> select * from patients_contact_info;
OK
```

```
select * from Patients_Contact_Info;
```

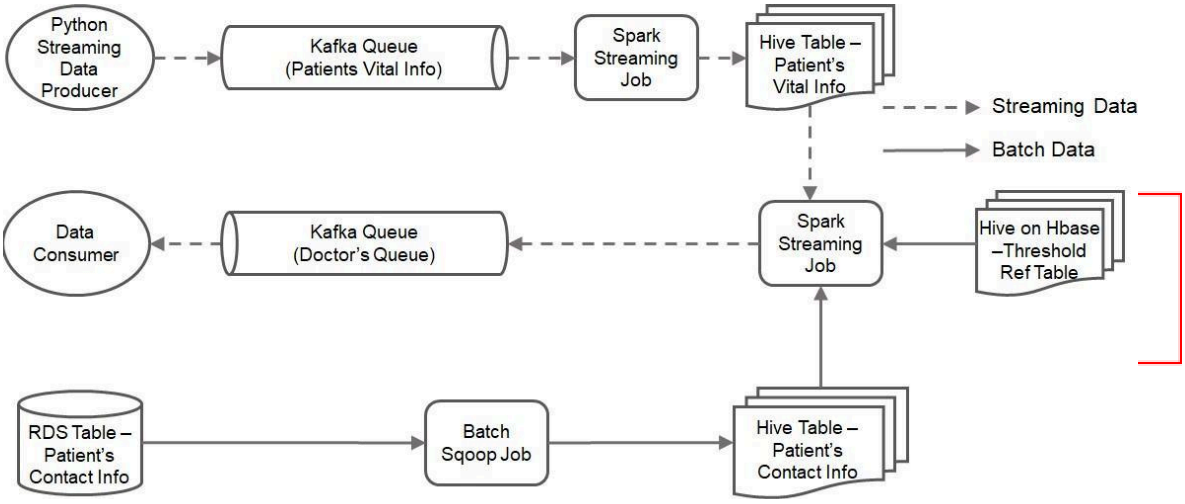
- View the records in ***Patients_Contact_Info*** table



```
hive> select * from Patients_Contact_Info;
OK
patients_contact_info.patientid patients_contact_info.patientname       patients
_contact_info.patientaddress    patients_contact_info.phone_number      patients
_contact_info.admitted_ward     patients_contact_info.age       patients_contact
_info.other_details
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: 1.541 seconds, Fetched: 5 row(s)
```

PART 2:

Create an HBase table to store threshold reference information and create a hive external table on top of this HBase table

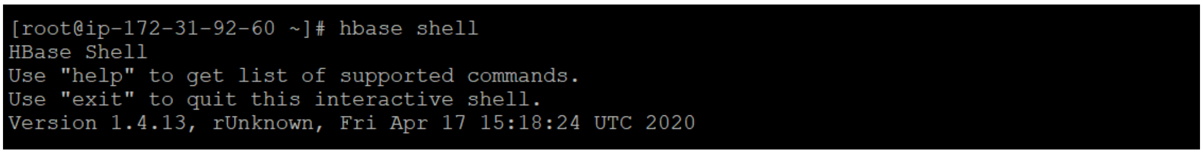


Python Streaming Data Producer
Kafka Queue (Patients Vital Info)
Spark Streaming Job
Hive Table – Patient's Vital Info
Streaming Data
Batch Data
Data Consumer
Kafka Queue (Doctor's Queue)
Spark Streaming Job
Hive on Hbase –Threshold Ref Table
RDS Table – Patient's Contact Info
Batch Sqoop Job
Hive Table – Patient's Contact Info

1. Create a HBase table named **threshold_ref** with 3 column families: attribute, limit, alert
2. Insert 12 records in this HBase table
3. Set up Hive-HBase integration (since HBase and Hive are on separate clusters)
4. Create a Hive external table named **Threshold_Reference_Table** on top of HBase table

Navigate to HBase shell using below commands:

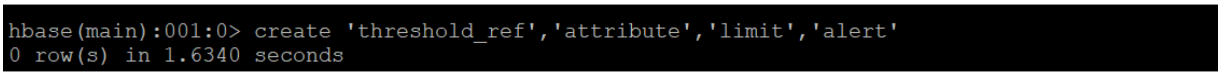
```
sudo -i
hbase shell
```



```
[root@ip-172-31-92-60 ~]# hbase shell
HBase Shell
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
Version 1.4.13, rUnknown, Fri Apr 17 15:18:24 UTC 2020
```

Create threshold_ref table in HBase

```
create 'threshold_ref','attribute','limit','alert'
```

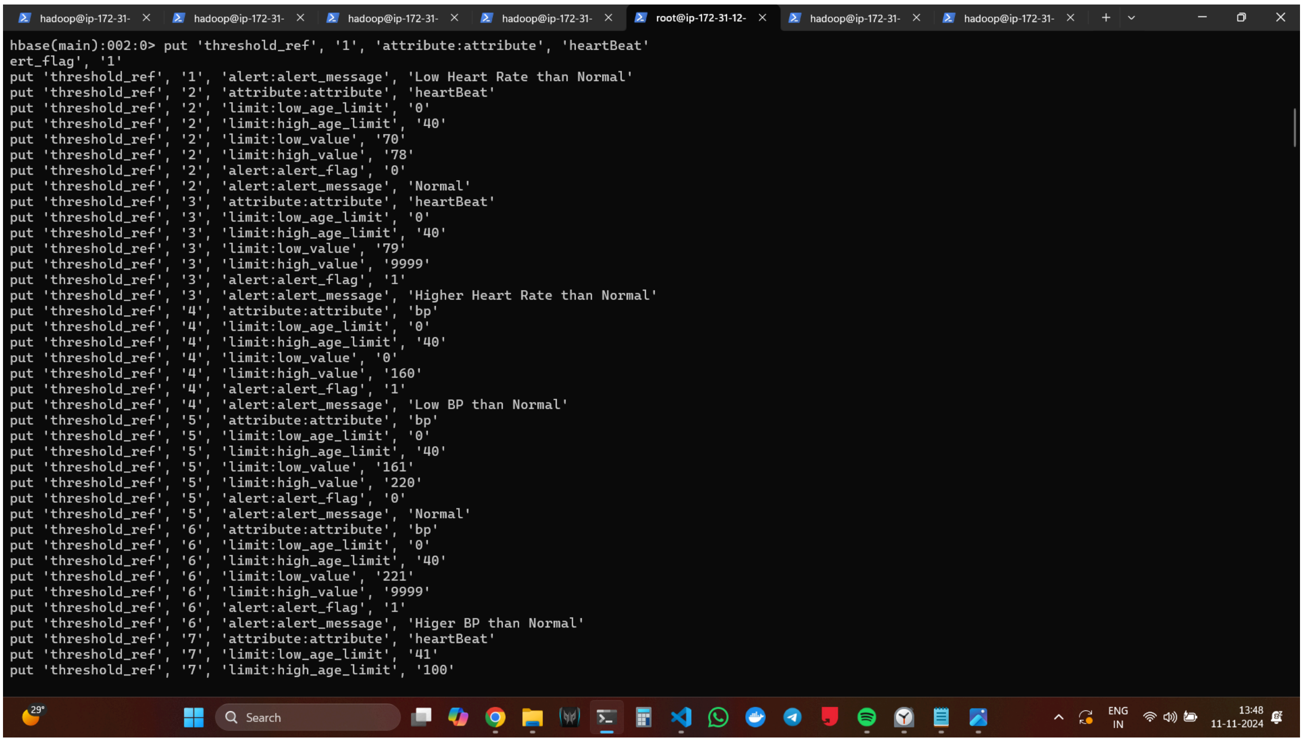


```
hbase(main):001:0> create 'threshold_ref','attribute','limit','alert'
0 row(s) in 1.6340 seconds
```

Insert records into HBase table

```
put 'threshold_ref', '1', 'attribute:attribute', 'heartBeat'
put 'threshold_ref', '1', 'limit:low_age_limit', '0'
put 'threshold_ref', '1', 'limit:high_age_limit', '40'
put 'threshold_ref', '1', 'limit:low_value', '0'
put 'threshold_ref', '1', 'limit:high_value', '69'
put 'threshold_ref', '1', 'alert:alert_flag', '1'
put 'threshold_ref', '1', 'alert:alert_message', 'Low Heart Rate than Normal'

put 'threshold_ref', '2', 'attribute:attribute', 'heartBeat'
put 'threshold_ref', '2', 'limit:low_age_limit', '0'
put 'threshold_ref', '2', 'limit:high_age_limit', '40'
put 'threshold_ref', '2', 'limit:low_value', '70'
put 'threshold_ref', '2', 'limit:high_value', '78'
put 'threshold_ref', '2', 'alert:alert_flag', '0'
put 'threshold_ref', '2', 'alert:alert_message', 'Normal'

put 'threshold_ref', '3', 'attribute:attribute', 'heartBeat'
put 'threshold_ref', '3', 'limit:low_age_limit', '0'
put 'threshold_ref', '3', 'limit:high_age_limit', '40'
put 'threshold_ref', '3', 'limit:low_value', '79'
put 'threshold_ref', '3', 'limit:high_value', '9999'
put 'threshold_ref', '3', 'alert:alert_flag', '1'
put 'threshold_ref', '3', 'alert:alert_message', 'Higher Heart Rate than Normal'

put 'threshold_ref', '4', 'attribute:attribute', 'bp'
put 'threshold_ref', '4', 'limit:low_age_limit', '0'
put 'threshold_ref', '4', 'limit:high_age_limit', '40'
put 'threshold_ref', '4', 'limit:low_value', '0'
put 'threshold_ref', '4', 'limit:high_value', '160'
put 'threshold_ref', '4', 'alert:alert_flag', '1'
put 'threshold_ref', '4', 'alert:alert_message', 'Low BP than Normal'

put 'threshold_ref', '5', 'attribute:attribute', 'bp'
put 'threshold_ref', '5', 'limit:low_age_limit', '0'
put 'threshold_ref', '5', 'limit:high_age_limit', '40'
put 'threshold_ref', '5', 'limit:low_value', '161'
put 'threshold_ref', '5', 'limit:high_value', '220'
put 'threshold_ref', '5', 'alert:alert_flag', '0'
put 'threshold_ref', '5', 'alert:alert_message', 'Normal'
put 'threshold_ref', '6', 'attribute:attribute', 'bp'
put 'threshold_ref', '6', 'limit:low_age_limit', '0'
put 'threshold_ref', '6', 'limit:high_age_limit', '40'
put 'threshold_ref', '6', 'limit:low_value', '221'
put 'threshold_ref', '6', 'limit:high_value', '9999'
put 'threshold_ref', '6', 'alert:alert_flag', '1'
put 'threshold_ref', '6', 'alert:alert_message', 'Higer BP than Normal'

put 'threshold_ref', '7', 'attribute:attribute', 'heartBeat'
put 'threshold_ref', '7', 'limit:low_age_limit', '41'
put 'threshold_ref', '7', 'limit:high_age_limit', '100'
put 'threshold_ref', '7', 'limit:low_value', '0'
put 'threshold_ref', '7', 'limit:high_value', '65'
put 'threshold_ref', '7', 'alert:alert_flag', '1'
put 'threshold_ref', '7', 'alert:alert_message', 'Low Heart Rate than Normal'

put 'threshold_ref', '8', 'attribute:attribute', 'heartBeat'
put 'threshold_ref', '8', 'limit:low_age_limit', '41'
put 'threshold_ref', '8', 'limit:high_age_limit', '100'
put 'threshold_ref', '8', 'limit:low_value', '66'
put 'threshold_ref', '8', 'limit:high_value', '73'
put 'threshold_ref', '8', 'alert:alert_flag', '0'
put 'threshold_ref', '8', 'alert:alert_message', 'Normal'

put 'threshold_ref', '9', 'attribute:attribute', 'heartBeat'
put 'threshold_ref', '9', 'limit:low_age_limit', '41'
put 'threshold_ref', '9', 'limit:high_age_limit', '100'
put 'threshold_ref', '9', 'limit:low_value', '74'
put 'threshold_ref', '9', 'limit:high_value', '9999'
put 'threshold_ref', '9', 'alert:alert_flag', '1'
put 'threshold_ref', '9', 'alert:alert_message', 'Higher Heart Rate than Normal'

put 'threshold_ref', '10', 'attribute:attribute', 'bp'
put 'threshold_ref', '10', 'limit:low_age_limit', '41'
put 'threshold_ref', '10', 'limit:high_age_limit', '100'
put 'threshold_ref', '10', 'limit:low_value', '0'
put 'threshold_ref', '10', 'limit:high_value', '150'
put 'threshold_ref', '10', 'alert:alert_flag', '1'
put 'threshold_ref', '10', 'alert:alert_message', 'Low BP than Normal'

put 'threshold_ref', '11', 'attribute:attribute', 'bp'
put 'threshold_ref', '11', 'limit:low_age_limit', '41'
put 'threshold_ref', '11', 'limit:high_age_limit', '100'
put 'threshold_ref', '11', 'limit:low_value', '151'
put 'threshold_ref', '11', 'limit:high_value', '180'
put 'threshold_ref', '11', 'alert:alert_flag', '0'
put 'threshold_ref', '11', 'alert:alert_message', 'Normal';

put 'threshold_ref', '12', 'attribute:attribute', 'bp'
put 'threshold_ref', '12', 'limit:low_age_limit', '41'
put 'threshold_ref', '12', 'limit:high_age_limit', '100'
put 'threshold_ref', '12', 'limit:low_value', '181'
put 'threshold_ref', '12', 'limit:high_value', '9999'
put 'threshold_ref', '12', 'alert:alert_flag', '1'
put 'threshold_ref', '12', 'alert:alert_message', 'Higher BP than Normal'
```

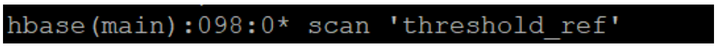
Screenshot for insertion of records



```
hbase(main):002:0> put 'threshold_ref', '1', 'attribute:attribute', 'heartBeat'
ert_flag', '1'
put 'threshold_ref', '1', 'alert:alert_message', 'Low Heart Rate than Normal'
put 'threshold_ref', '2', 'attribute:attribute', 'heartBeat'
put 'threshold_ref', '2', 'limit:low_age_limit', '0'
put 'threshold_ref', '2', 'limit:high_age_limit', '40'
put 'threshold_ref', '2', 'limit:low_value', '70'
put 'threshold_ref', '2', 'limit:high_value', '78'
put 'threshold_ref', '2', 'alert:alert_flag', '0'
put 'threshold_ref', '2', 'alert:alert_message', 'Normal'
put 'threshold_ref', '3', 'attribute:attribute', 'heartBeat'
put 'threshold_ref', '3', 'limit:low_age_limit', '0'
put 'threshold_ref', '3', 'limit:high_age_limit', '40'
put 'threshold_ref', '3', 'limit:low_value', '79'
put 'threshold_ref', '3', 'limit:high_value', '9999'
put 'threshold_ref', '3', 'alert:alert_flag', '1'
put 'threshold_ref', '3', 'alert:alert_message', 'Higher Heart Rate than Normal'
put 'threshold_ref', '4', 'attribute:attribute', 'bp'
put 'threshold_ref', '4', 'limit:low_age_limit', '0'
put 'threshold_ref', '4', 'limit:high_age_limit', '40'
put 'threshold_ref', '4', 'limit:low_value', '0'
put 'threshold_ref', '4', 'limit:high_value', '160'
put 'threshold_ref', '4', 'alert:alert_flag', '1'
put 'threshold_ref', '4', 'alert:alert_message', 'Low BP than Normal'
put 'threshold_ref', '5', 'attribute:attribute', 'bp'
put 'threshold_ref', '5', 'limit:low_age_limit', '0'
put 'threshold_ref', '5', 'limit:high_age_limit', '40'
put 'threshold_ref', '5', 'limit:low_value', '161'
put 'threshold_ref', '5', 'limit:high_value', '220'
put 'threshold_ref', '5', 'alert:alert_flag', '0'
put 'threshold_ref', '5', 'alert:alert_message', 'Normal'
put 'threshold_ref', '6', 'attribute:attribute', 'bp'
put 'threshold_ref', '6', 'limit:low_age_limit', '0'
put 'threshold_ref', '6', 'limit:high_age_limit', '40'
put 'threshold_ref', '6', 'limit:low_value', '221'
put 'threshold_ref', '6', 'limit:high_value', '9999'
put 'threshold_ref', '6', 'alert:alert_flag', '1'
put 'threshold_ref', '6', 'alert:alert_message', 'Higer BP than Normal'
put 'threshold_ref', '7', 'attribute:attribute', 'heartBeat'
put 'threshold_ref', '7', 'limit:low_age_limit', '41'
put 'threshold_ref', '7', 'limit:high_age_limit', '100'
```

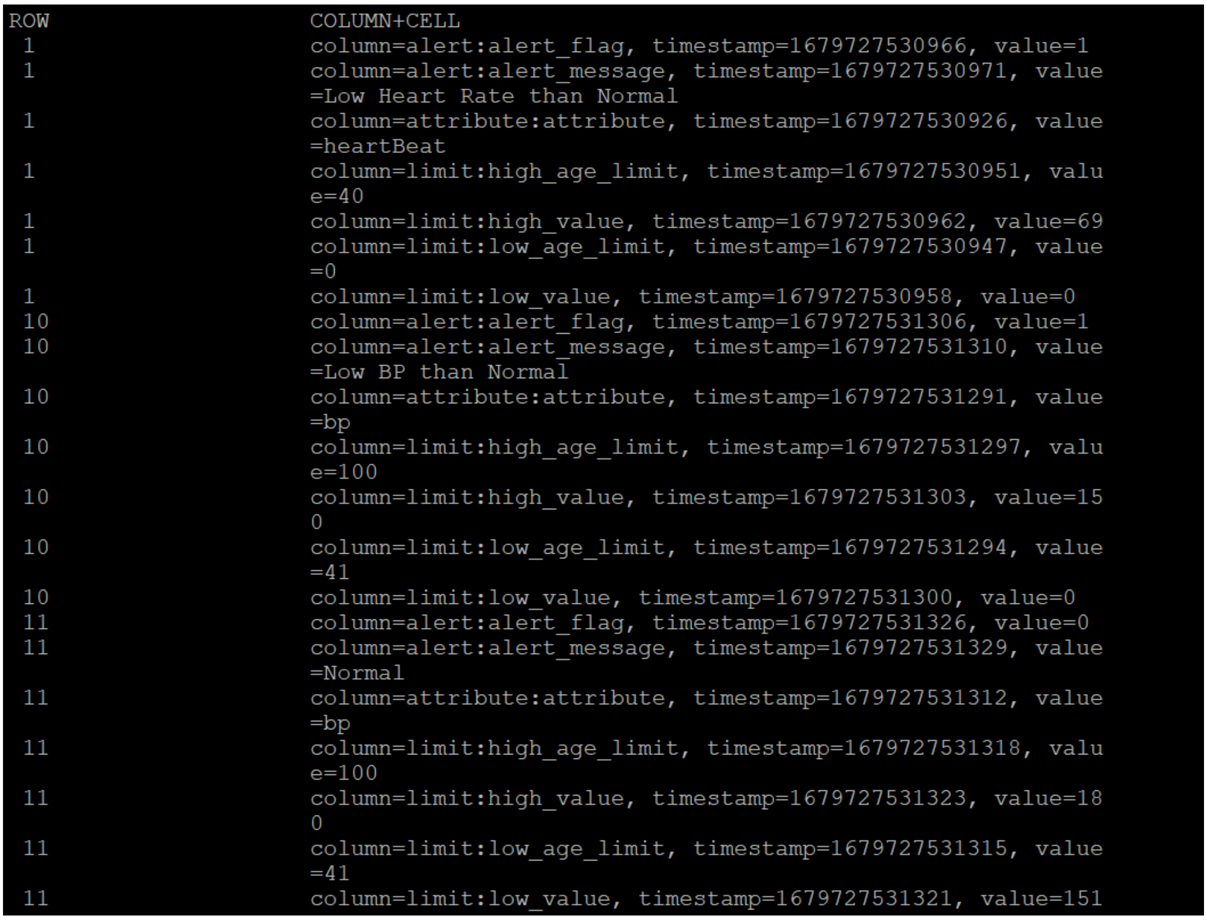
View records in HBase table

```
scan 'threshold_ref'
```

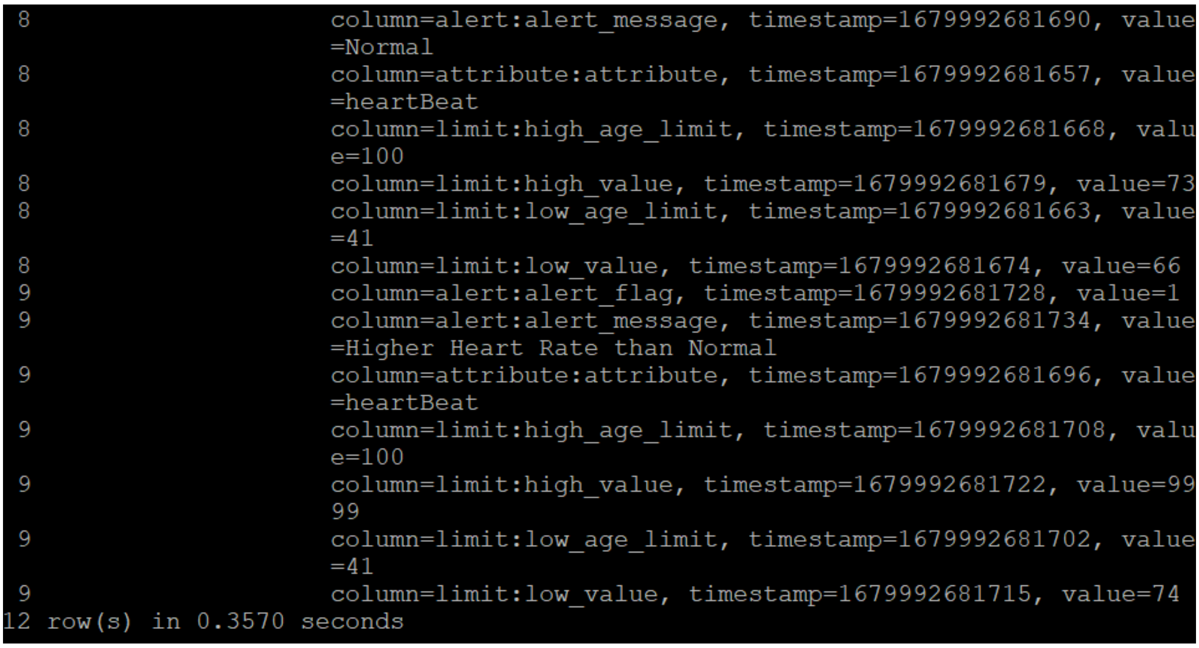


```
hbase(main):098:0* scan 'threshold_ref'
```

Screenshots of records



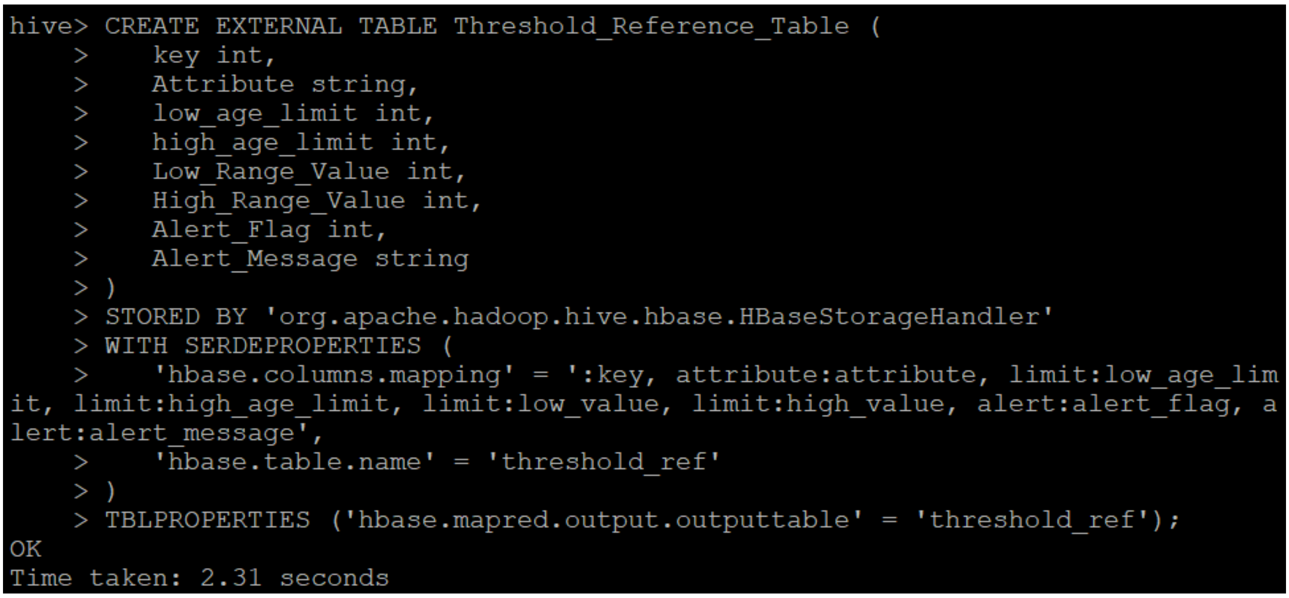
```
ROW                    COLUMN+CELL
 1                     column=alert:alert_flag, timestamp=1679727530966, value=1
 1                     column=alert:alert_message, timestamp=1679727530971, value
                       =Low Heart Rate than Normal
 1                     column=attribute:attribute, timestamp=1679727530926, value
                       =heartBeat
 1                     column=limit:high_age_limit, timestamp=1679727530951, valu
                       e=40
 1                     column=limit:high_value, timestamp=1679727530962, value=69
 1                     column=limit:low_age_limit, timestamp=1679727530947, value
                       =0
 1                     column=limit:low_value, timestamp=1679727530958, value=0
 10                    column=alert:alert_flag, timestamp=1679727531306, value=1
 10                    column=alert:alert_message, timestamp=1679727531310, value
                       =Low BP than Normal
 10                    column=attribute:attribute, timestamp=1679727531291, value
                       =bp
 10                    column=limit:high_age_limit, timestamp=1679727531297, valu
                       e=100
 10                    column=limit:high_value, timestamp=1679727531303, value=15
                       0
 10                    column=limit:low_age_limit, timestamp=1679727531294, value
                       =41
 10                    column=limit:low_value, timestamp=1679727531300, value=0
 11                    column=alert:alert_flag, timestamp=1679727531326, value=0
 11                    column=alert:alert_message, timestamp=1679727531329, value
                       =Normal
 11                    column=attribute:attribute, timestamp=1679727531312, value
                       =bp
 11                    column=limit:high_age_limit, timestamp=1679727531318, valu
                       e=100
 11                    column=limit:high_value, timestamp=1679727531323, value=18
                       0
 11                    column=limit:low_age_limit, timestamp=1679727531315, value
                       =41
 11                    column=limit:low_value, timestamp=1679727531321, value=151
```



```
 8                    column=alert:alert_message, timestamp=1679992681690, value
                      =Normal
 8                    column=attribute:attribute, timestamp=1679992681657, value
                      =heartBeat
 8                    column=limit:high_age_limit, timestamp=1679992681668, valu
                      e=100
 8                    column=limit:high_value, timestamp=1679992681679, value=73
 8                    column=limit:low_age_limit, timestamp=1679992681663, value
                      =41
 8                    column=limit:low_value, timestamp=1679992681674, value=66
 9                    column=alert:alert_flag, timestamp=1679992681728, value=1
 9                    column=alert:alert_message, timestamp=1679992681734, value
                      =Higher Heart Rate than Normal
 9                    column=attribute:attribute, timestamp=1679992681696, value
                      =heartBeat
 9                    column=limit:high_age_limit, timestamp=1679992681708, valu
                      e=100
 9                    column=limit:high_value, timestamp=1679992681722, value=99
                      99
 9                    column=limit:low_age_limit, timestamp=1679992681702, value
                      =41
 9                    column=limit:low_value, timestamp=1679992681715, value=74
12 row(s) in 0.3570 seconds
```

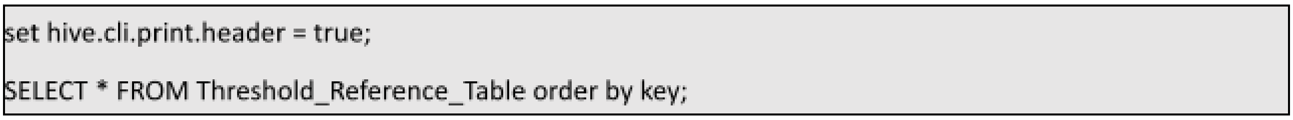
Threshold_Reference_Table in Hive

```
 CREATE EXTERNAL TABLE Threshold_Reference_Table (
   key int,
Attribute string,
   low_age_limit int,
   high_age_limit int,
   Low_Range_Value int,
   High_Range_Value int,
   Alert_Flag int,
   Alert_Message string
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
 WITH SERDEPROPERTIES (
   'hbase.columns.mapping' = ':key, attribute:attribute, limit:low_age_limit, limit:high_age_limit, limit:low_value,
 limit:high_value, alert:alert_flag, alert:alert_message',
'hbase.table.name' = 'threshold_ref'
)
TBLPROPERTIES ('hbase.mapred.output.outputtable' = 'threshold_ref');
```

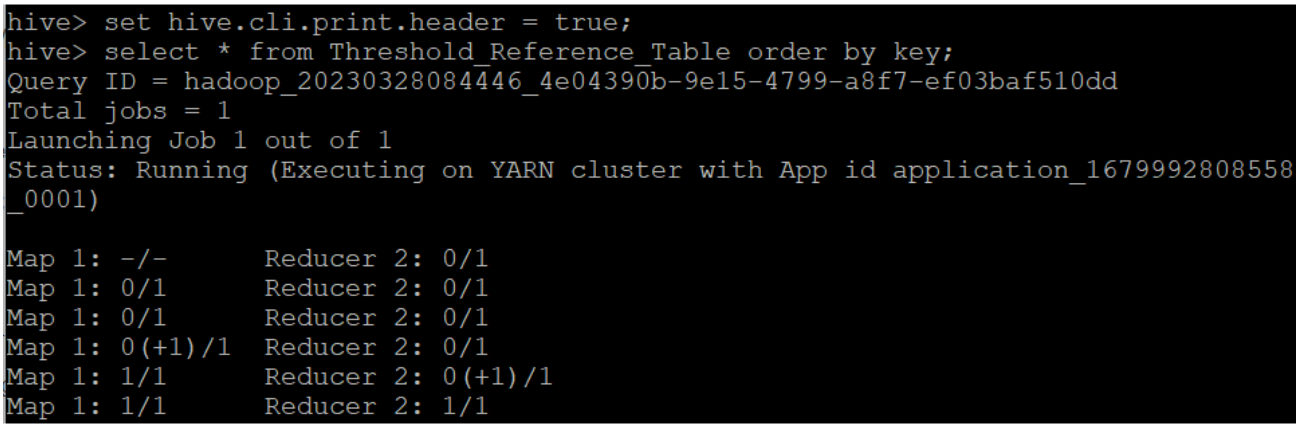


```
hive> CREATE EXTERNAL TABLE Threshold_Reference_Table (
    >     key int,
    >     Attribute string,
    >     low_age_limit int,
    >     high_age_limit int,
    >     Low_Range_Value int,
    >     High_Range_Value int,
    >     Alert_Flag int,
    >     Alert_Message string
    > )
    > STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
    > WITH SERDEPROPERTIES (
    >     'hbase.columns.mapping' = ':key, attribute:attribute, limit:low_age_lim
it, limit:high_age_limit, limit:low_value, limit:high_value, alert:alert_flag, a
lert:alert_message',
    >     'hbase.table.name' = 'threshold_ref'
    > )
    > TBLPROPERTIES ('hbase.mapred.output.outputtable' = 'threshold_ref');
OK
Time taken: 2.31 seconds
```

- View the contents of Threshold_Reference_Table

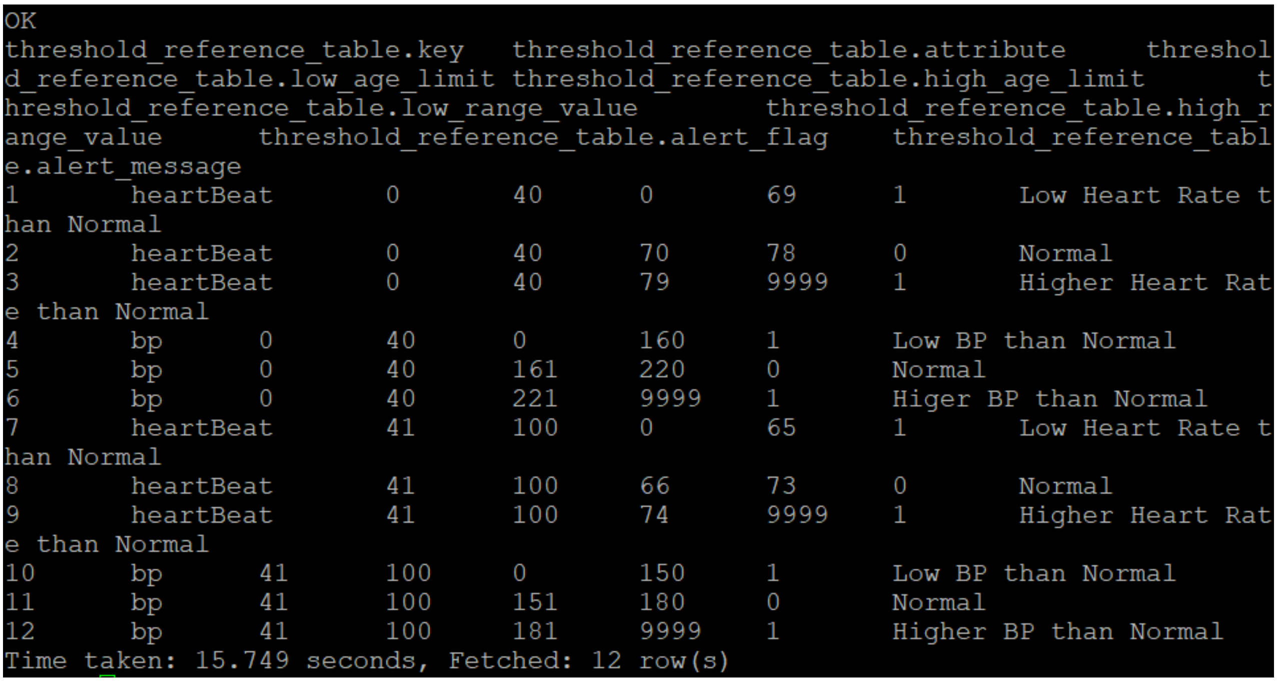


```
set hive.cli.print.header = true;

SELECT * FROM Threshold_Reference_Table order by key;
```



```
hive> set hive.cli.print.header = true;
hive> select * from Threshold_Reference_Table order by key;
Query ID = hadoop_20230328084446_4e04390b-9e15-4799-a8f7-ef03baf510dd
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1679992808558
_0001)

Map 1: -/-      Reducer 2: 0/1
Map 1: 0/1      Reducer 2: 0/1
Map 1: 0/1      Reducer 2: 0/1
Map 1: 0(+1)/1  Reducer 2: 0/1
Map 1: 1/1      Reducer 2: 0(+1)/1
Map 1: 1/1      Reducer 2: 1/1
```

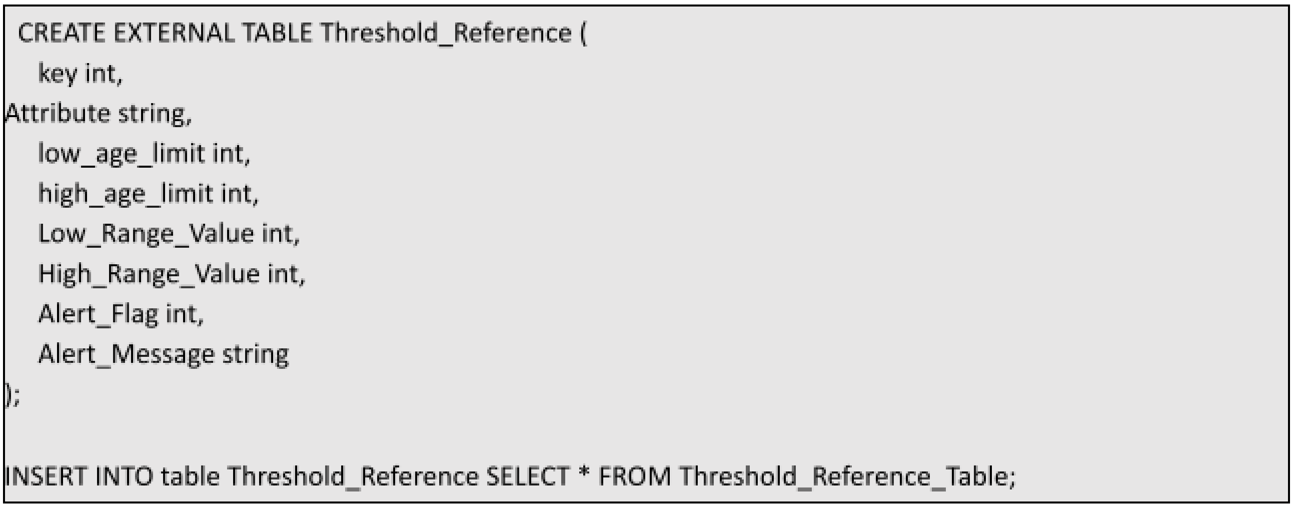
Screenshot of Threshold_Reference_Table records:



```
OK
threshold_reference_table.key   threshold_reference_table.attribute     threshol
d_reference_table.low_age_limit threshold_reference_table.high_age_limit        t
hreshold_reference_table.low_range_value        threshold_reference_table.high_r
ange_value      threshold_reference_table.alert_flag    threshold_reference_tabl
e.alert_message
1       heartBeat       0       40      0       69      1       Low Heart Rate t
han Normal
2       heartBeat       0       40      70      78      0       Normal
3       heartBeat       0       40      79      9999    1       Higher Heart Rat
e 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       Higer BP than Normal
7       heartBeat       41      100     0       65      1       Low Heart Rate t
han Normal
8       heartBeat       41      100     66      73      0       Normal
9       heartBeat       41      100     74      9999    1       Higher Heart Rat
e 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
Time taken: 15.749 seconds, Fetched: 12 row(s)
```

- Create copy of threshold table ***Threshold_Reference*** in Hive and insert the records from Threshold_Reference_Table to Threshold_Reference

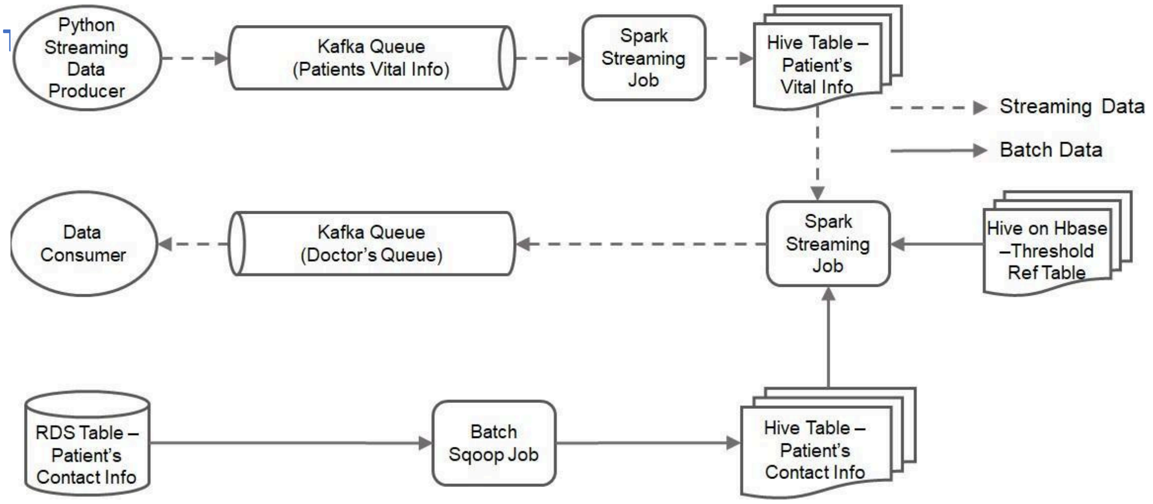
This table can be accessed by spark streaming application 2 for Part 4 mentioned below)



```
 CREATE EXTERNAL TABLE Threshold_Reference (
   key int,
Attribute string,
   low_age_limit int,
   high_age_limit int,
   Low_Range_Value int,
   High_Range_Value int,
   Alert_Flag int,
   Alert_Message string
);

INSERT INTO table Threshold_Reference SELECT * FROM Threshold_Reference_Table;
```

PART 3:

Dealing with Kafka part in EMR Cluster


Python Streaming Data Producer
Kafka Queue (Patients Vital Info)
Spark Streaming Job
Hive Table – Patient's Vital Info
Streaming Data
Batch Data
Data Consumer
Kafka Queue (Doctor's Queue)
Spark Streaming Job
Hive on Hbase –Threshold Ref Table
RDS Table – Patient's Contact Info
Batch Sqoop Job
Hive Table – Patient's Contact Info

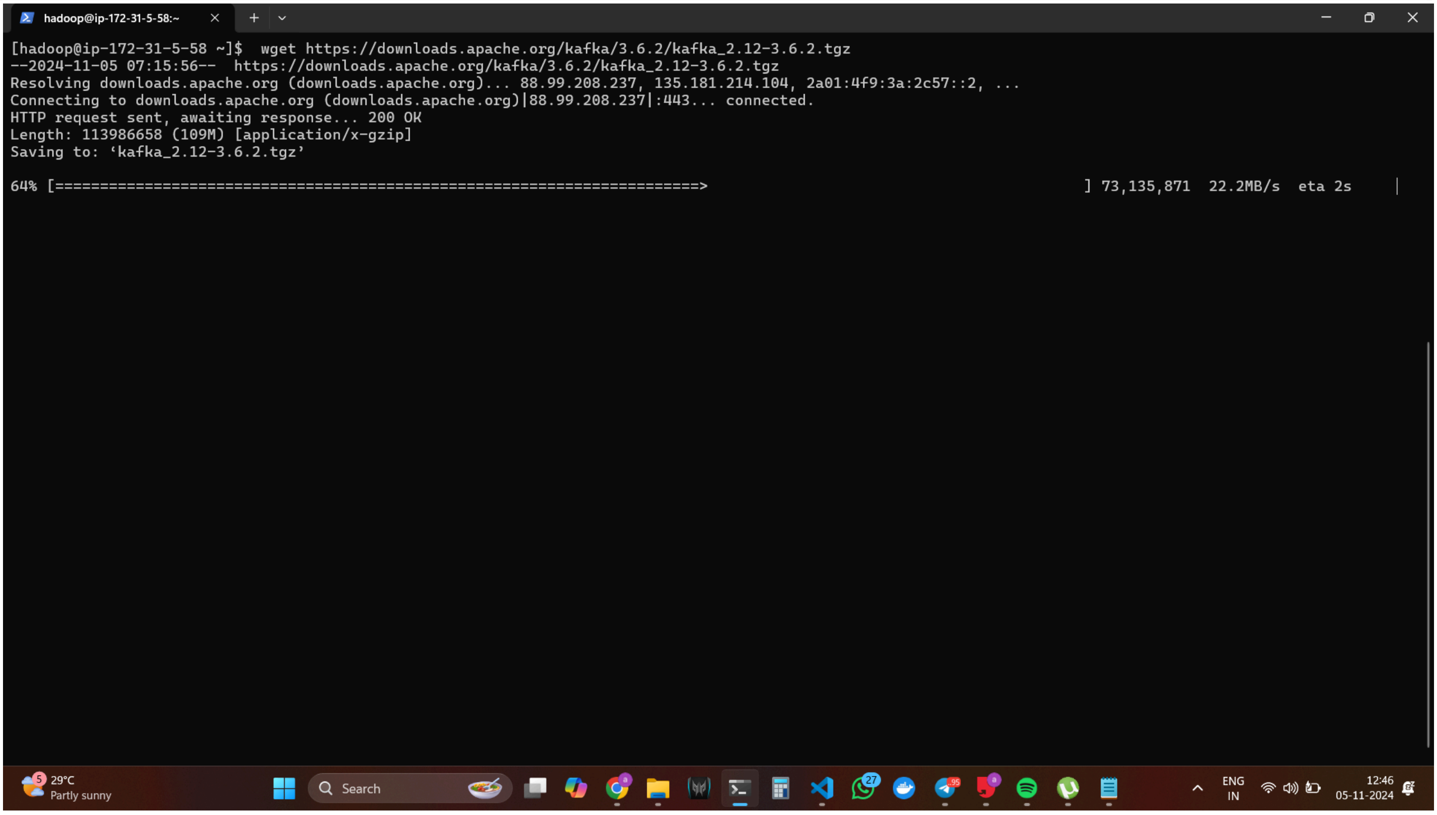
This involves following tasks:

1. Build a Kafka Producer application in Python to simulate streaming data by reading from RDS every second.
2. Push patient vitals data to the Kafka topic `patients_vital_info`.
3. Create a Spark streaming job to:
4. Read data from the Kafka topic.
5. Add a timestamp column.
6. Store the data in HDFS in Parquet format.
7. Set up an external Hive table, `patients_vital_info`, to read streaming data from the HDFS location.
8. The producer will process all 1800 records within 30 minutes

Setting Up Apache Kafka on an EMR Cluster

Downloading Kafka files

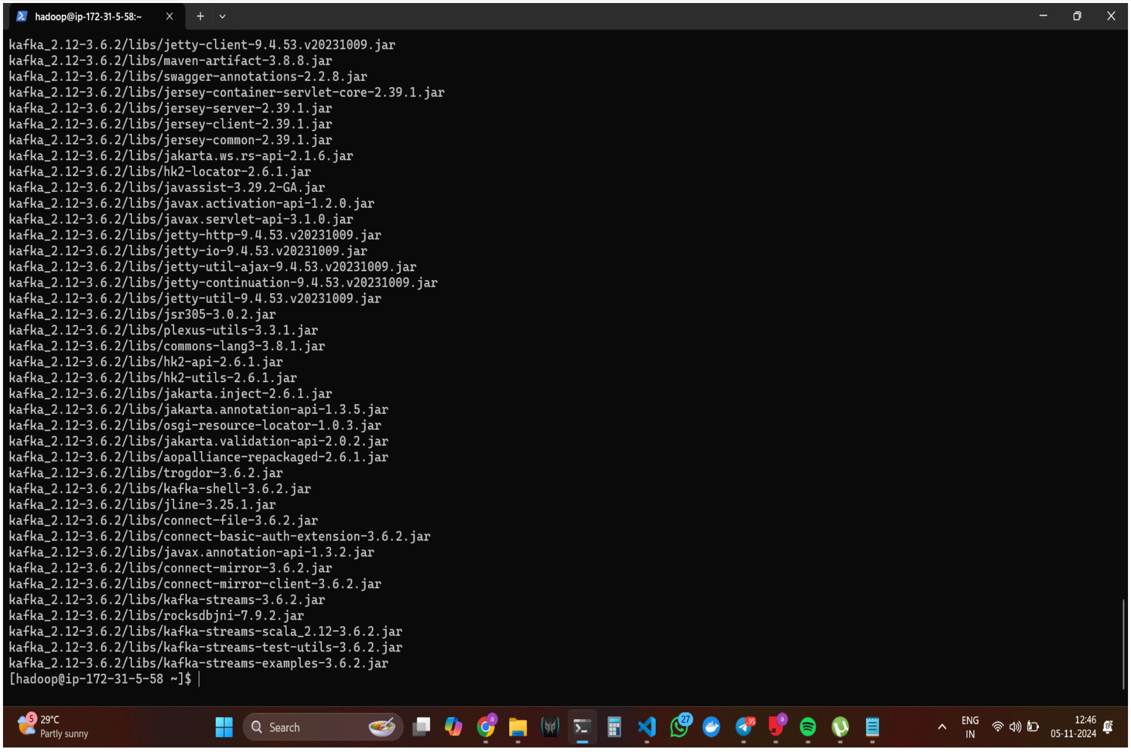
```
wget https://downloads.apache.org/kafka/3.6.2/kafka_2.12-3.6.2.tgz
```



```
[hadoop@ip-172-31-5-58 ~]$  wget https://downloads.apache.org/kafka/3.6.2/kafka_2.12-3.6.2.tgz
--2024-11-05 07:15:56--  https://downloads.apache.org/kafka/3.6.2/kafka_2.12-3.6.2.tgz
Resolving downloads.apache.org (downloads.apache.org)... 88.99.208.237, 135.181.214.104, 2a01:4f9:3a:2c57::2, ...
Connecting to downloads.apache.org (downloads.apache.org)|88.99.208.237|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 113986658 (109M) [application/x-gzip]
Saving to: ‘kafka_2.12-3.6.2.tgz’

64% [=======================================================================>                                       ] 73,135,871  22.2MB/s  eta 2s
```

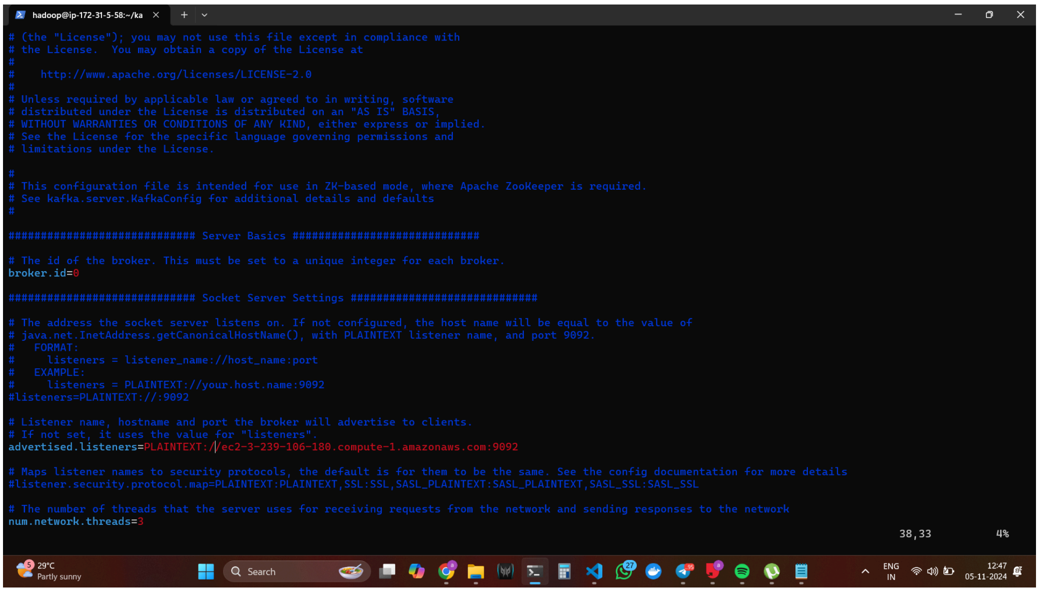
Extract Kafka files

```
tar -xvf kafka_2.12-3.6.2.tgz
```



```
kafka_2.12-3.6.2/libs/jetty-client-9.4.53.v20231009.jar
kafka_2.12-3.6.2/libs/maven-artifact-3.8.8.jar
kafka_2.12-3.6.2/libs/swagger-annotations-2.2.8.jar
kafka_2.12-3.6.2/libs/jersey-container-servlet-core-2.39.1.jar
kafka_2.12-3.6.2/libs/jersey-server-2.39.1.jar
kafka_2.12-3.6.2/libs/jersey-client-2.39.1.jar
kafka_2.12-3.6.2/libs/jersey-common-2.39.1.jar
kafka_2.12-3.6.2/libs/jakarta.ws.rs-api-2.1.6.jar
kafka_2.12-3.6.2/libs/hk2-locator-2.6.1.jar
kafka_2.12-3.6.2/libs/javassist-3.29.2-GA.jar
kafka_2.12-3.6.2/libs/javax.activation-api-1.2.0.jar
kafka_2.12-3.6.2/libs/javax.servlet-api-3.1.0.jar
kafka_2.12-3.6.2/libs/jetty-http-9.4.53.v20231009.jar
kafka_2.12-3.6.2/libs/jetty-io-9.4.53.v20231009.jar
kafka_2.12-3.6.2/libs/jetty-util-ajax-9.4.53.v20231009.jar
kafka_2.12-3.6.2/libs/jetty-continuation-9.4.53.v20231009.jar
kafka_2.12-3.6.2/libs/jetty-util-9.4.53.v20231009.jar
kafka_2.12-3.6.2/libs/jsr305-3.0.2.jar
kafka_2.12-3.6.2/libs/plexus-utils-3.3.1.jar
kafka_2.12-3.6.2/libs/commons-lang3-3.8.1.jar
kafka_2.12-3.6.2/libs/hk2-api-2.6.1.jar
kafka_2.12-3.6.2/libs/hk2-utils-2.6.1.jar
kafka_2.12-3.6.2/libs/jakarta.inject-2.6.1.jar
kafka_2.12-3.6.2/libs/jakarta.annotation-api-1.3.5.jar
kafka_2.12-3.6.2/libs/osgi-resource-locator-1.0.3.jar
kafka_2.12-3.6.2/libs/jakarta.validation-api-2.0.2.jar
kafka_2.12-3.6.2/libs/aopalliance-repackaged-2.6.1.jar
kafka_2.12-3.6.2/libs/trogdor-3.6.2.jar
kafka_2.12-3.6.2/libs/kafka-shell-3.6.2.jar
kafka_2.12-3.6.2/libs/jline-3.25.1.jar
kafka_2.12-3.6.2/libs/connect-file-3.6.2.jar
kafka_2.12-3.6.2/libs/connect-basic-auth-extension-3.6.2.jar
kafka_2.12-3.6.2/libs/javax.annotation-api-1.3.2.jar
kafka_2.12-3.6.2/libs/connect-mirror-3.6.2.jar
kafka_2.12-3.6.2/libs/connect-mirror-client-3.6.2.jar
kafka_2.12-3.6.2/libs/kafka-streams-3.6.2.jar
kafka_2.12-3.6.2/libs/rocksdbjni-7.9.2.jar
kafka_2.12-3.6.2/libs/kafka-streams-scala_2.12-3.6.2.jar
kafka_2.12-3.6.2/libs/kafka-streams-test-utils-3.6.2.jar
kafka_2.12-3.6.2/libs/kafka-streams-examples-3.6.2.jar
[hadoop@ip-172-31-5-58 ~]$
```

Update the advertised listeners to use our EMR IP address



```
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

#
# This configuration file is intended for use in ZK-based mode, where Apache ZooKeeper is required.
# See kafka.server.KafkaConfig for additional details and defaults
#

############################# Server Basics #############################

# The id of the broker. This must be set to a unique integer for each broker.
broker.id=0

############################# Socket Server Settings #############################

# The address the socket server listens on. If not configured, the host name will be equal to the value of
# java.net.InetAddress.getCanonicalHostName(), with PLAINTEXT listener name, and port 9092.
#   FORMAT:
#     listeners = listener_name://host_name:port
#   EXAMPLE:
#     listeners = PLAINTEXT://your.host.name:9092
#listeners=PLAINTEXT://:9092

# Listener name, hostname and port the broker will advertise to clients.
# If not set, it uses the value for "listeners".
advertised.listeners=PLAINTEXT://ec2-3-239-106-180.compute-1.amazonaws.com:9092

# Maps listener names to security protocols, the default is for them to be the same. See the config documentation for more details
#listener.security.protocol.map=PLAINTEXT:PLAINTEXT,SSL:SSL,SASL_PLAINTEXT:SASL_PLAINTEXT,SASL_SSL:SASL_SSL

# The number of threads that the server uses for receiving requests from the network and sending responses to the network
num.network.threads=3
                                                                                  38,33          4%
```

- Start Zookeeper server

```
cd kafka_2.12-3.6.2
bin/zookeeper-server-start.sh config/zookeeper.properties
```

- Start Kafka server

```
cd downloads/kafka_2.12-3.6.2
bin/kafka-server-start.sh config/server.properties
```

```
cd kafka_2.12-3.6.2

bin/kafka-topics.sh --bootstrap-server localhost9092 --delete -- topic patients_vital_info
```

- Delete the topic if it already exists and create again

```
bin/kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1
--topic patients_vital_info
```

- Transfer the Python producer file and Install required packages

```
WINSCP transfer file to emr cluster

pip install mysql-connector-python  kafka-python
```

- Run the producer script

```
python kafka_produce_patient_vitals.py
```

Spark Streaming Job 1 (kafka_spark_patient_vitals.py)

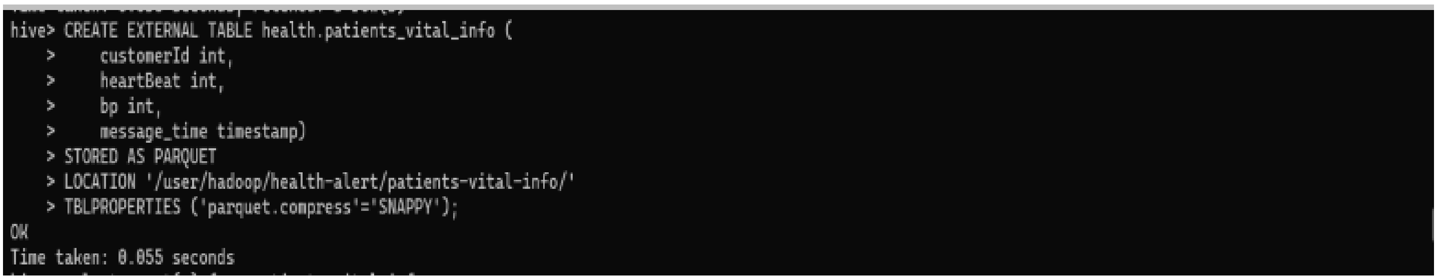
It reads data from Kakfa topic and add the **timestamp column** and store in HDFS location in parquet format.

```
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
```

```
hive

use health;

CREATE EXTERNAL TABLE health.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');
```

In the Hive Shell

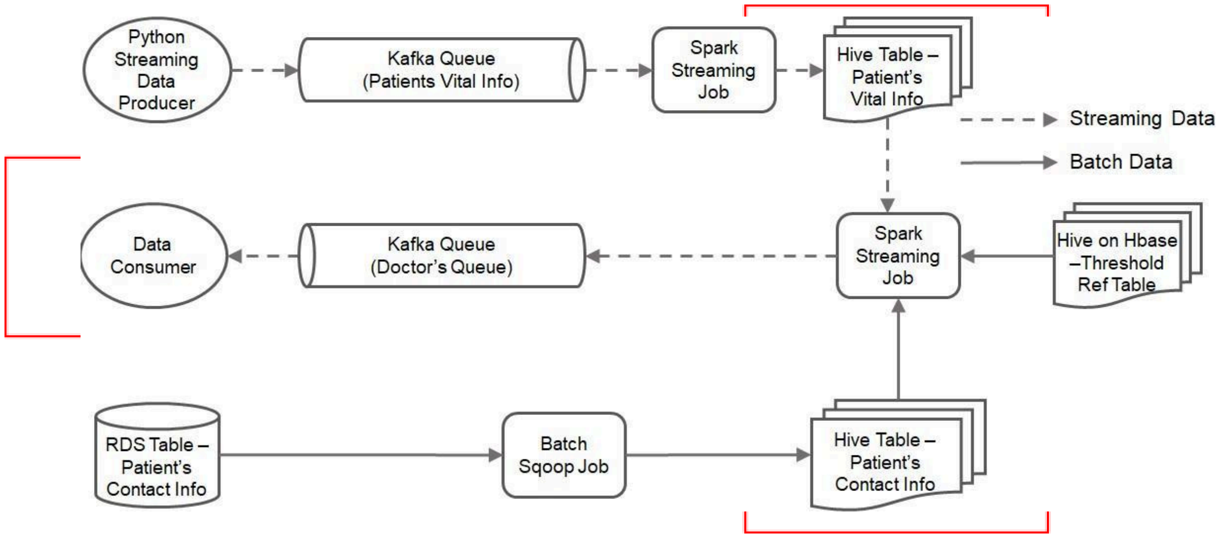
- Create external table ‘patients_vital_info’



```
hive> CREATE EXTERNAL TABLE health.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: 0.055 seconds
```

PART 4:

Compare vital information with threshold values, analyze the results, and send notifications if the data exceeds threshold limits.


Python Streaming Data Producer
Kafka Queue (Patients Vital Info)
Spark Streaming Job
Hive Table – Patient's Vital Info
Streaming Data
Batch Data
Data Consumer
Kafka Queue (Doctor's Queue)
Spark Streaming Job
Hive on Hbase –Threshold Ref Table
RDS Table – Patient's Contact Info
Batch Sqoop Job
Hive Table – Patient's Contact Info

This involves following tasks:

1. Create a Kafka topic called `alerts_message` to store abnormal patient vital signs.
2. Develop a Spark streaming job to:
3. Read data from three Hive tables.
4. Analyze patient vitals and, if irregular, push them to the `alerts_message`Kafka topic.
5. Set up a Kafka consumer to read messages from the `alerts_message` topic.
6. Use SNS to send email notifications for messages received by the consumer.

Create a Kafka topic (alerts_message)

- Delete the topic if it already exists

```
bin/kafka-topics.sh --bootstrap-server localhost:9092 --delete -- topic alerts_message
```

- Create the topic again

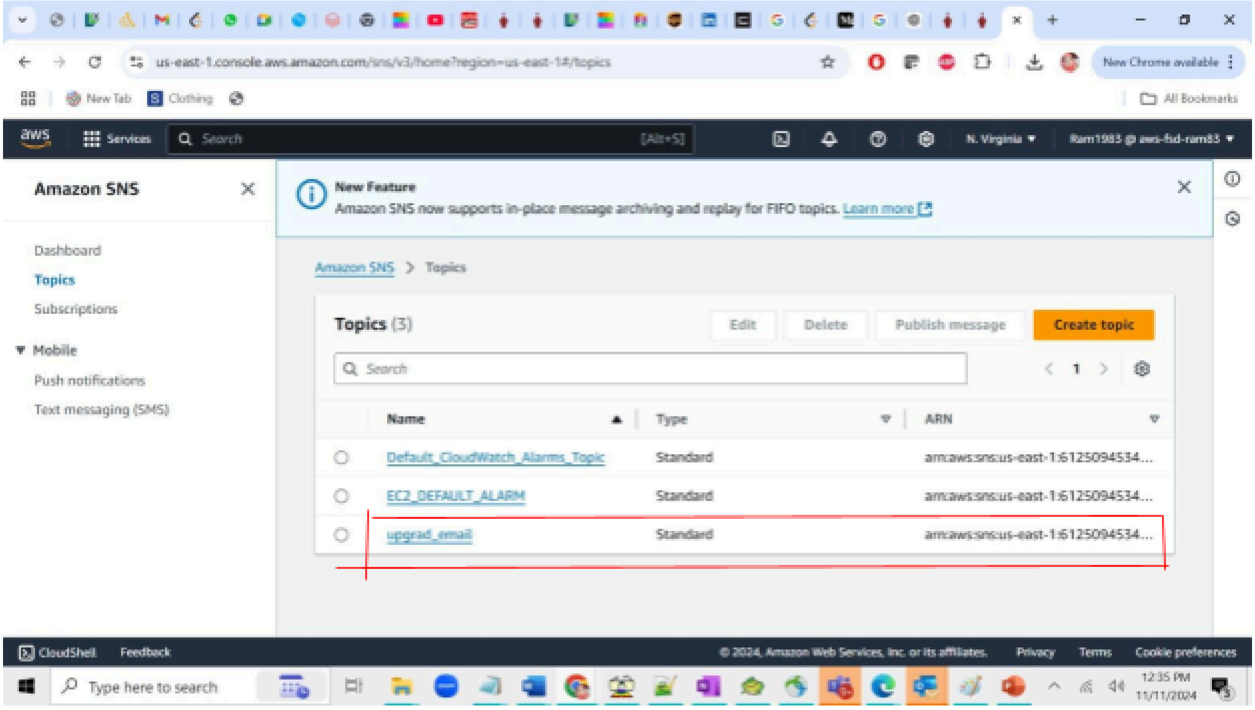
```
bin/kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic
alerts_message
```

Spark Streaming Job to push irregular patient vitals to alerts_message Kafka topic

```
export SPARK_KAFKA_VERSION=0.10
spark-submit --executor-memory 4G --num-executors 4 --packages
org.apache.spark:spark-sql-kafka-0-10_2.11:2.4.5
kafka_spark_generate_alerts.py
```

Configure SNS

Create SNS topic (Health_Alerts)



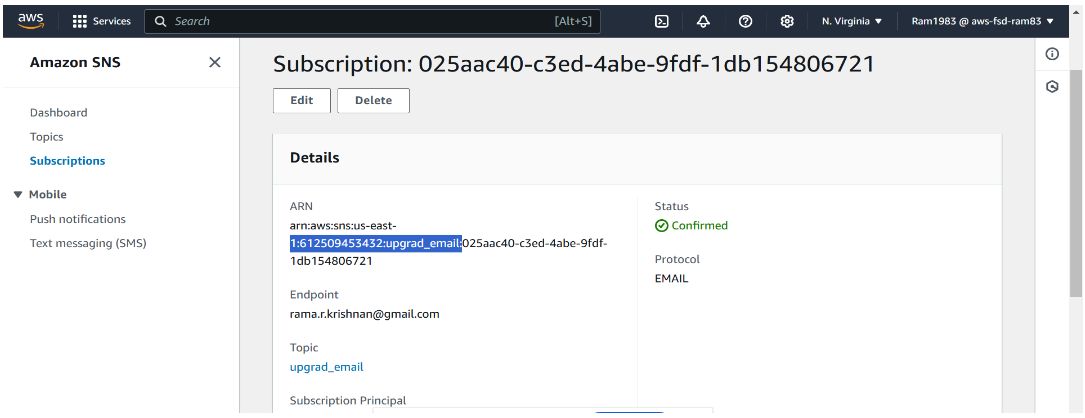
New Feature
Amazon SNS now supports in-place message archiving and replay for FIFO topics. Learn more

Amazon SNS > Topics

Topics (3)

	Name	Type	ARN
○	Default_CloudWatch_Alarms_Topic	Standard	arn:aws:sns:us-east-1:6125094534...
○	EC2_DEFAULT_ALARM	Standard	arn:aws:sns:us-east-1:6125094534...
○	upgrad_email	Standard	arn:aws:sns:us-east-1:6125094534...

Subscribe to SNS topic



Subscription: 025aac40-c3ed-4abe-9fdf-1db154806721

Details

ARN
arn:aws:sns:us-east-1:612509453432:upgrad_email:025aac40-c3ed-4abe-9fdf-1db154806721

Endpoint
rama.r.krishnan@gmail.com

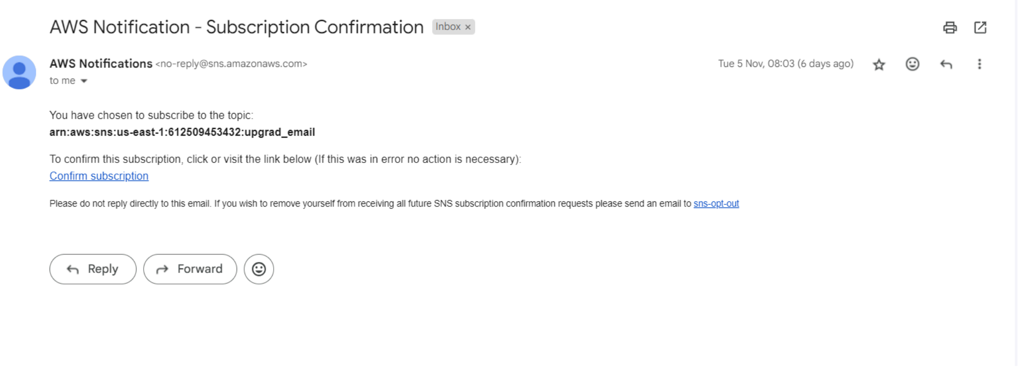
Topic
upgrad_email

Subscription Principal

Status
Confirmed

Protocol
EMAIL

Subscription Confirmation Email



AWS Notification - Subscription Confirmation

AWS Notifications <no-reply@sns.amazonaws.com>
to me

Tue 5 Nov, 08:03 (6 days ago)

You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:612509453432:upgrad_email

To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
Confirm subscription

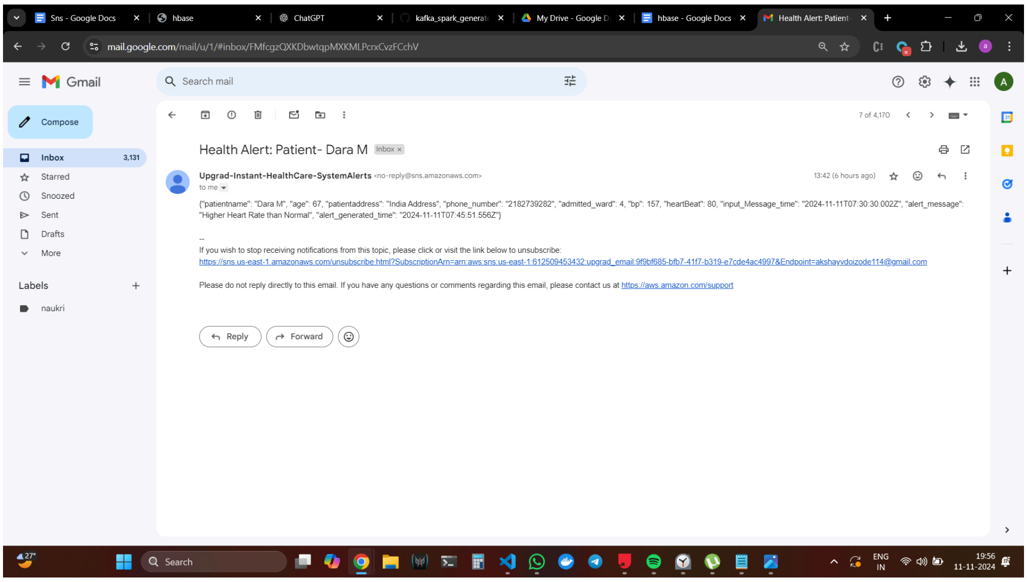
Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to sns-opt-out

Kafka consumer

```
pip install kafka-python
pip install boto3
python kafka_consume_alerts.py
```

Final Output Screenshot:

Patient health Alert email - using SNS



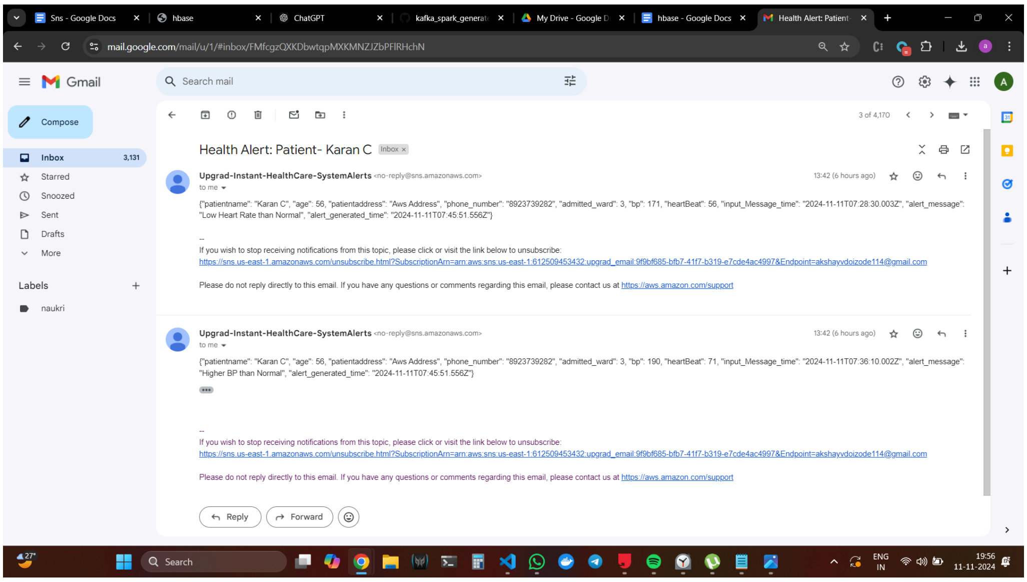
Health Alert: Patient- Dara M Inbox

Upgrad-Instant-HealthCare-SystemAlerts <no-reply@sns.amazonaws.com> 13:42 (6 hours ago)
to me

{"patientname": "Dara M", "age": 67, "patientaddress": "India Address", "phone_number": "2182739282", "admitted_ward": 4, "bp": 157, "heartBeat": 80, "input_Message_time": "2024-11-11T07:30:30.002Z", "alert_message": "Higher Heart Rate than Normal", "alert_generated_time": "2024-11-11T07:45:51.556Z"}

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:612509453432:upgrad_email:9f9bf685-bfb7-41f7-b319-e7cde4ac4997&Endpoint=akshayvdoizode114@gmail.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support



Health Alert: Patient- Karan C Inbox

Upgrad-Instant-HealthCare-SystemAlerts <no-reply@sns.amazonaws.com> 13:42 (6 hours ago)
to me

{"patientname": "Karan C", "age": 56, "patientaddress": "Aws Address", "phone_number": "8923739282", "admitted_ward": 3, "bp": 171, "heartBeat": 56, "input_Message_time": "2024-11-11T07:28:30.003Z", "alert_message": "Low Heart Rate than Normal", "alert_generated_time": "2024-11-11T07:45:51.556Z"}

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:612509453432:upgrad_email:9f9bf685-bfb7-41f7-b319-e7cde4ac4997&Endpoint=akshayvdoizode114@gmail.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support

Upgrad-Instant-HealthCare-SystemAlerts <no-reply@sns.amazonaws.com> 13:42 (6 hours ago)
to me

{"patientname": "Karan C", "age": 56, "patientaddress": "Aws Address", "phone_number": "8923739282", "admitted_ward": 3, "bp": 190, "heartBeat": 71, "input_Message_time": "2024-11-11T07:36:10.002Z", "alert_message": "Higher BP than Normal", "alert_generated_time": "2024-11-11T07:45:51.556Z"}

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:612509453432:upgrad_email:9f9bf685-bfb7-41f7-b319-e7cde4ac4997&Endpoint=akshayvdoizode114@gmail.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support

Summary

1. Setup and Configuration:

- Kafka Installation: Download and extract Kafka, configure the server, and start both Zookeeper and Kafka servers. Create two Kafka topics: `patients_vital_info` (for patient data) and `alerts_message` (for alerts).
- Python and Kafka Libraries: Install required Python libraries (`mysql-connector-python` and `kafka-python`) for data processing and connectivity.
- MySQL Setup: Install MySQL Connector, configure MySQL, set up user privileges, and enable access for data import.

2. Data Import with Sqoop and Storage in Hive:

- MySQL Data Import: Import patient contact information from MySQL RDS into HDFS using Sqoop.
- Hive Table Creation: Create a Hive database and external table (`patients_contact_info`) to store imported patient data

3. Threshold Configuration with HBase and Hive Integration:

- HBase Table Creation: Create `threshold_ref` in HBase to store threshold values for patient vitals.
- Hive-HBase Integration: Create an external Hive table (`threshold_reference_table`) linked to `threshold_ref` in HBase, allowing Hive to access threshold data for analysis.

4. Streaming and Real-Time Processing with Kafka and Spark:

- Kafka Producer and Spark Job: A Python Kafka producer simulates streaming data by sending patient vitals to `patients_vital_info`. A Spark streaming job then reads data from Kafka, adds timestamps, and stores it in HDFS in Parquet format.
- Hive Table for Streaming Data: Create an external Hive table (`patients_vital_info`) to read the streaming data from HDFS.

5. Alert Generation and Notification:

- Spark Streaming for Alerts: Create a Spark job to analyze the patient vitals against threshold values from Hive tables. Any irregularities are pushed to the `alerts_message` Kafka topic.
- Kafka Consumer and SNS Notifications: Set up a Kafka consumer to read from `alerts_message` and send email notifications using AWS SNS for any alert messages.