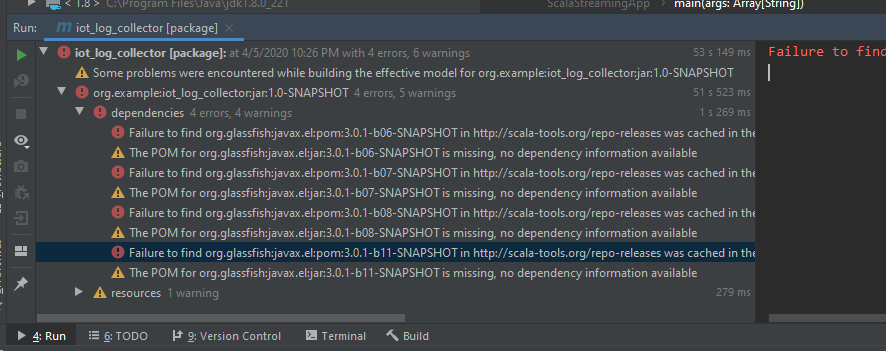
# IoT Data Collector Application Demo

## Prerequisites

* Cloudera Quickstart VM 5.13 has been downloaded and started up.
* Cloudera VM jdk version is set with 1.8.0\_202 before the Apache Spark 2 parcel upgrade
* Apache Spark 2 parcel upgrade has been downloaded and distributed to node.

## Installing and Running

When first compiling of the project, maven gives below errors, due to wrong dependency definition (-SNAPSHOT) of the **org.apache.hbase.connectors.spark:hbase-spark:1.0.0**.



* org.glassfish:javax.el:pom:3.0.1-b06-SNAPSHOT
* org.glassfish:javax.el:pom:3.0.1-b07-SNAPSHOT
* org.glassfish:javax.el:pom:3.0.1-b08-SNAPSHOT
* org.glassfish:javax.el:pom:3.0.1-b11-SNAPSHOT

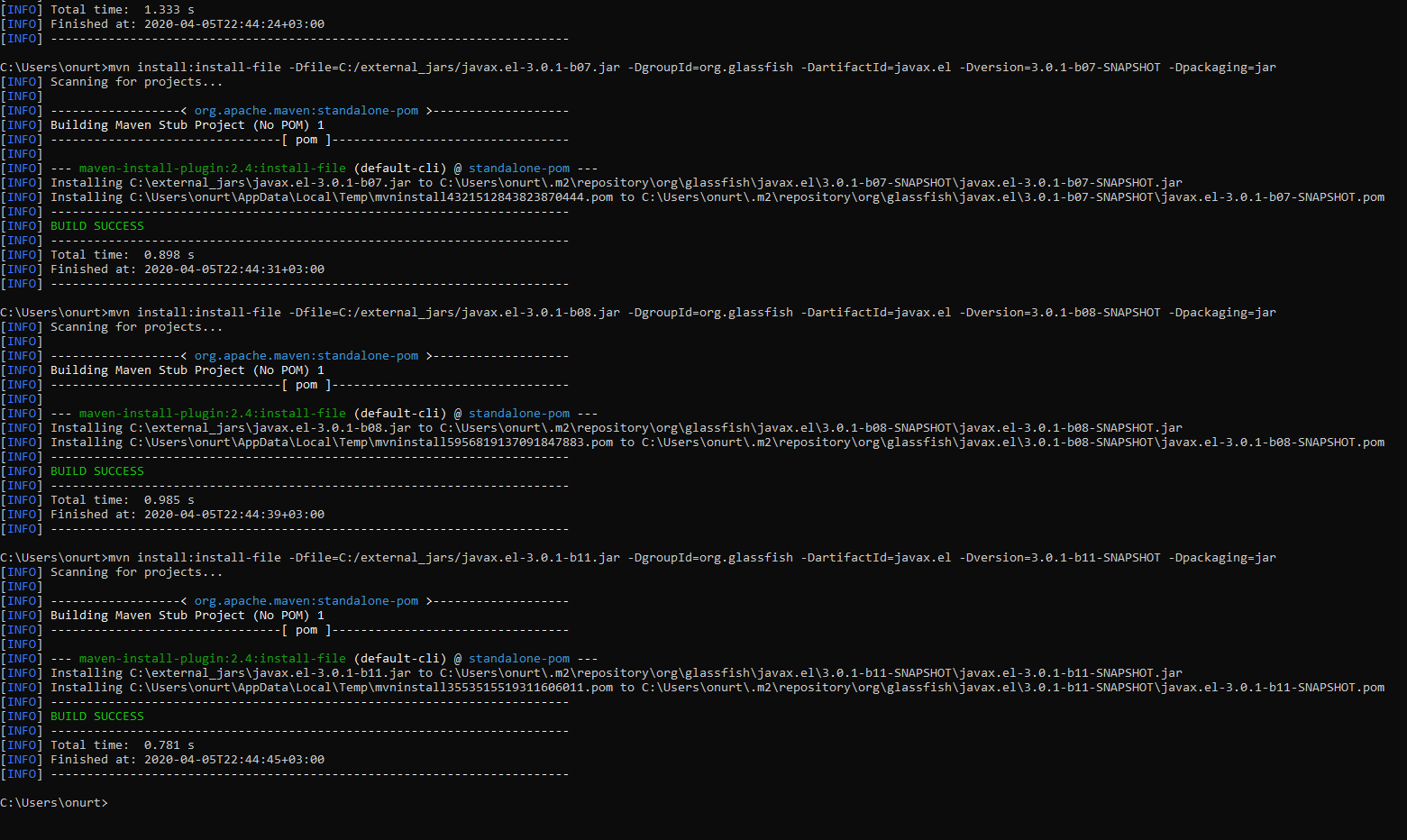
These jars have been downloaded (<https://mvnrepository.com/artifact/org.glassfish/javax.el>) and loaded to local maven manually using below maven command.

*mvn install:install-file -Dfile=C:/external\_jars/javax.el-3.0.1-b06.jar -DgroupId=org.glassfish -DartifactId=javax.el -Dversion=3.0.1-b06-SNAPSHOT -Dpackaging=jar*

*mvn install:install-file -Dfile=C:/external\_jars/javax.el-3.0.1-b07.jar -DgroupId=org.glassfish -DartifactId=javax.el -Dversion=3.0.1-b07-SNAPSHOT -Dpackaging=jar*

*mvn install:install-file -Dfile=C:/external\_jars/javax.el-3.0.1-b08.jar -DgroupId=org.glassfish -DartifactId=javax.el -Dversion=3.0.1-b08-SNAPSHOT -Dpackaging=jar*

*mvn install:install-file -Dfile=C:/external\_jars/javax.el-3.0.1-b11.jar -DgroupId=org.glassfish -DartifactId=javax.el -Dversion=3.0.1-b11-SNAPSHOT -Dpackaging=jar*



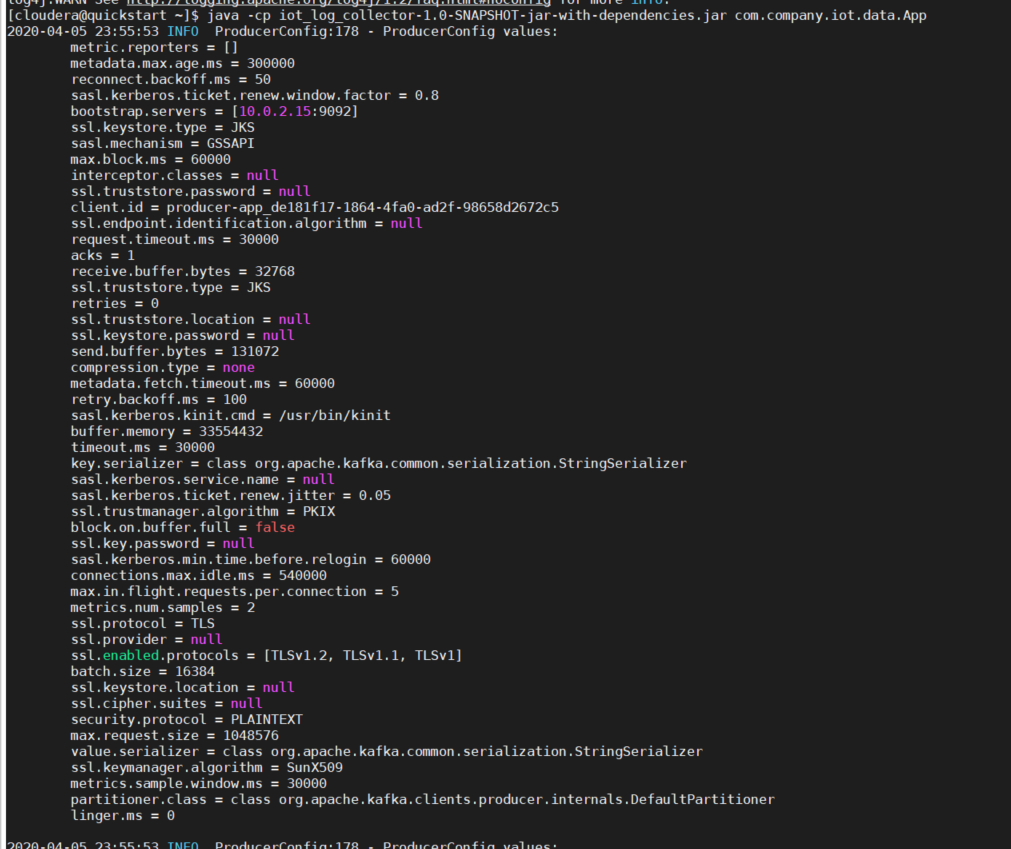
After creating executable jar (mvn clean package), iot\_log\_collector-1.0-SNAPSHOT-jar-with-dependencies.jar file will be used for data pipelines.

Project have two executable parts. First part is Kafka data ingestion application, the other part is Spark data processing application. Both of them have been written by Java.

#### Kafka Data Ingestion Application Part

In order to run data ingestion application, below command will be used

*java -cp iot\_log\_collector-1.0-SNAPSHOT-jar-with-dependencies.jar com.company.iot.data.App*



#### Spark Data Processing Application Part

Before the running Spark application (spark-submit), SPARK\_KAFKA\_VERSION variable should be set as below

*export SPARK\_KAFKA\_VERSION=0.10*

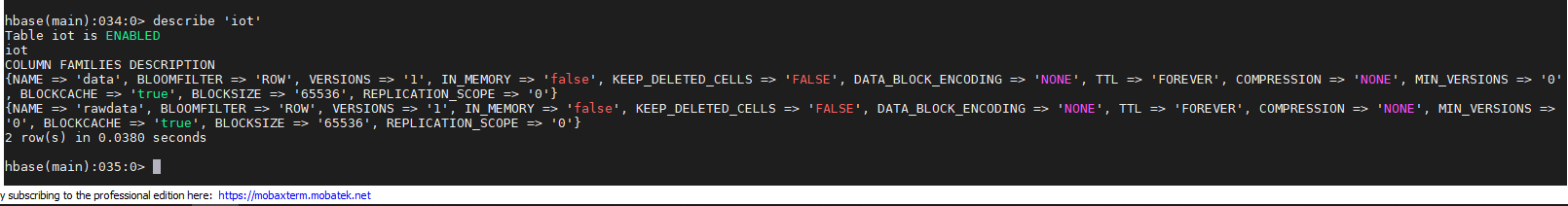
and

*spark-submit --master local --class com.company.iot.data.streaming.JavaHBaseStreaming iot\_log\_collector-1.0-SNAPSHOT-jar-with-dependencies.jar*

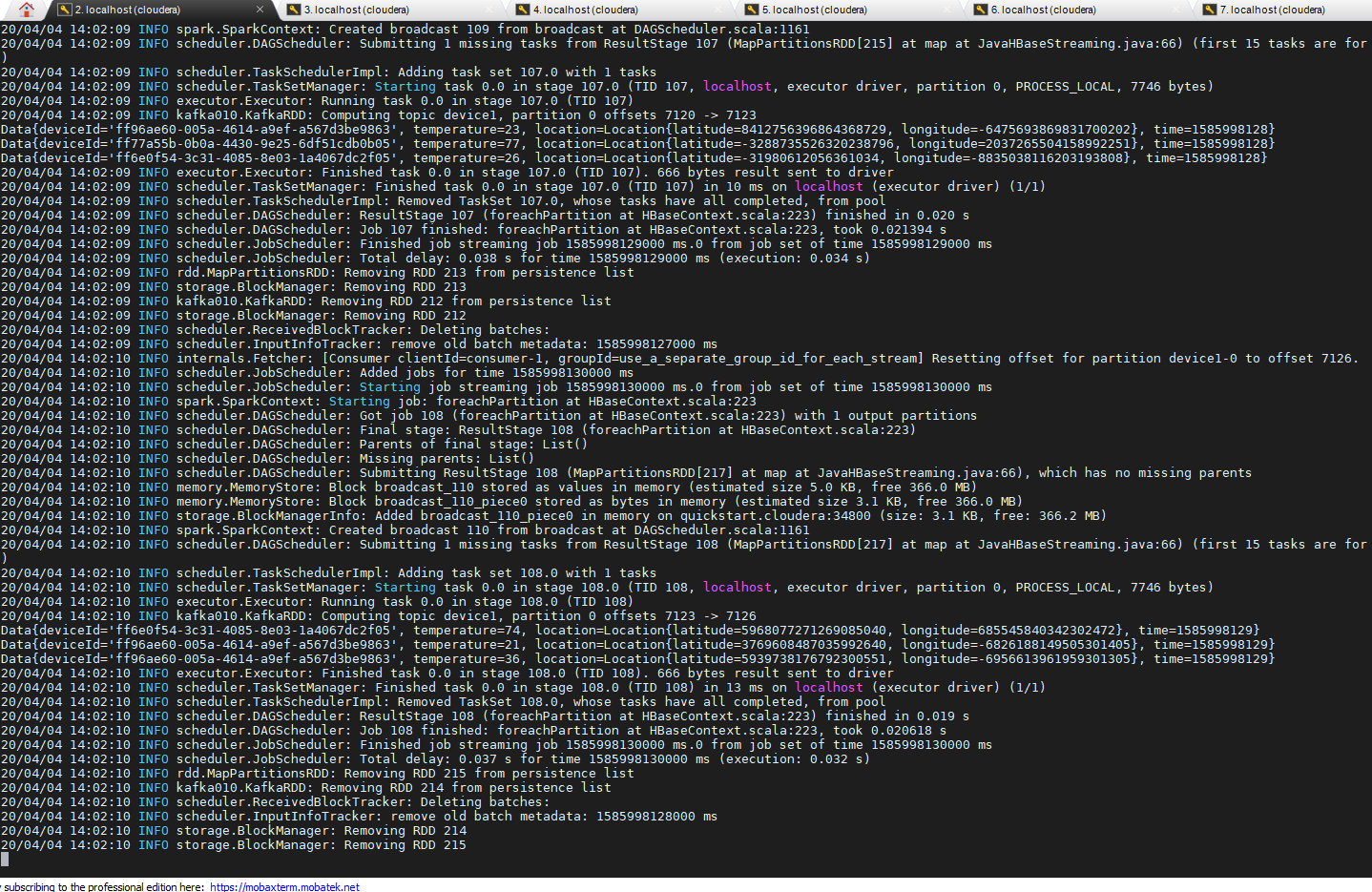
## Starting

Before the data ingestion and data processing start, Hbase table created.

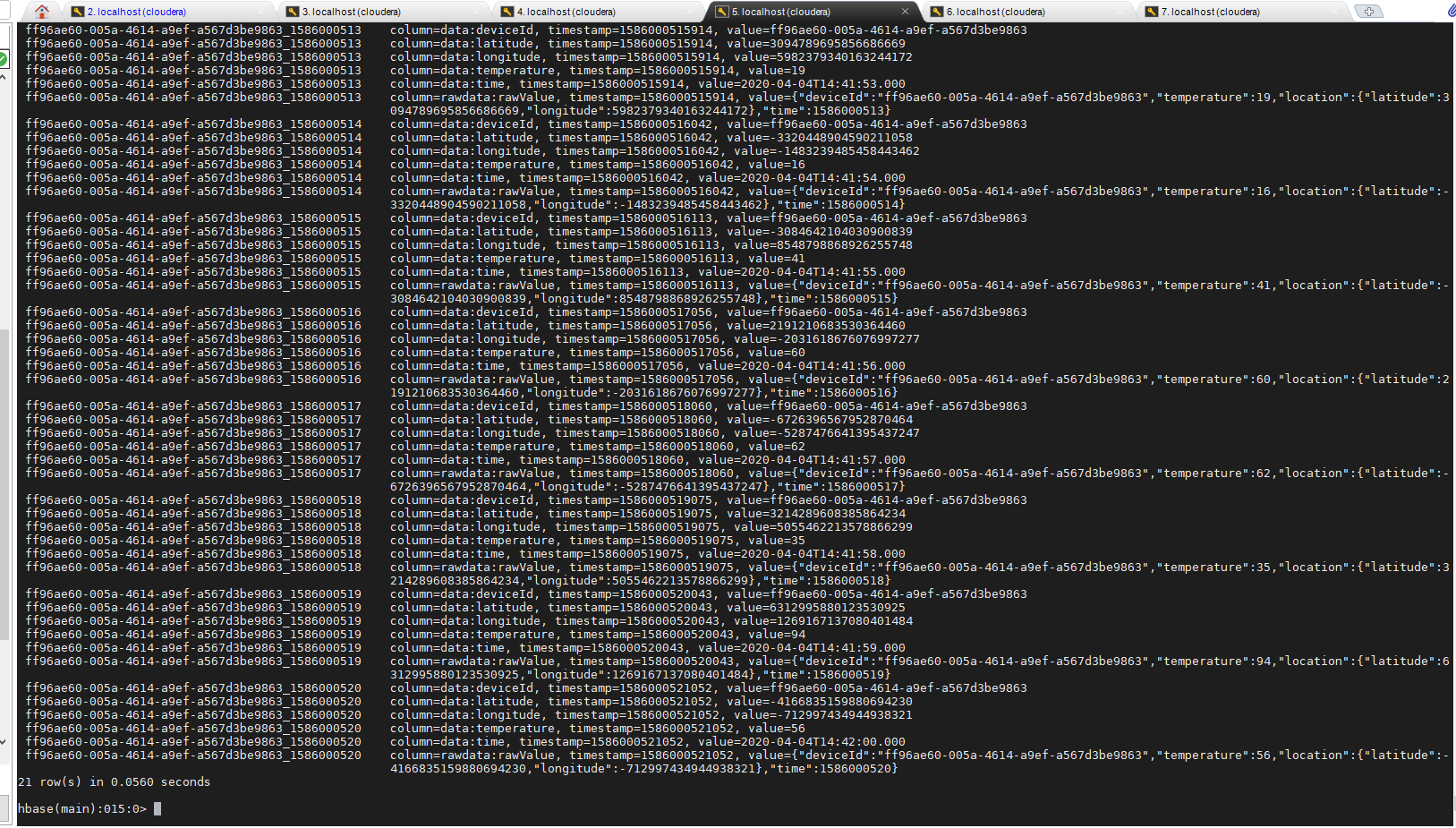
Create ‘iot’,’rawdata’,’data’



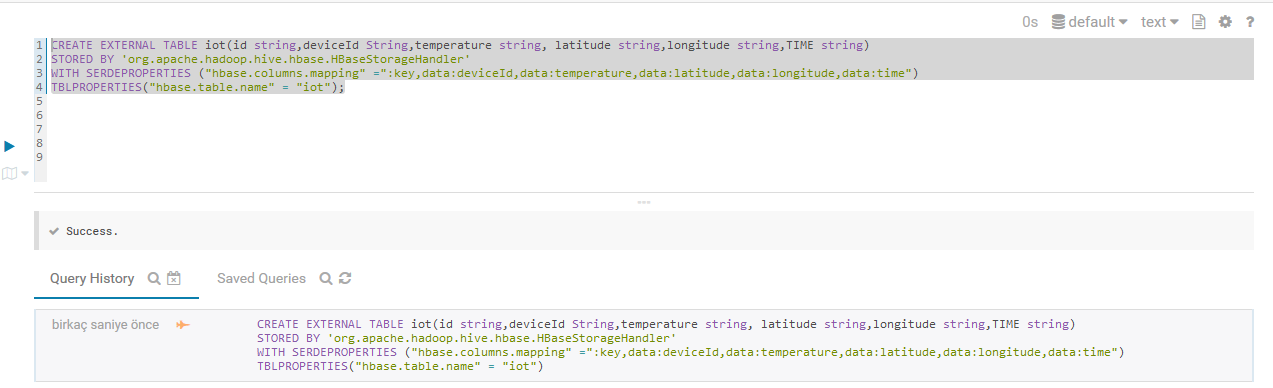
First, Kafka application has been started and then, Spark streaming application has been started. Spark application console output can be seen as below



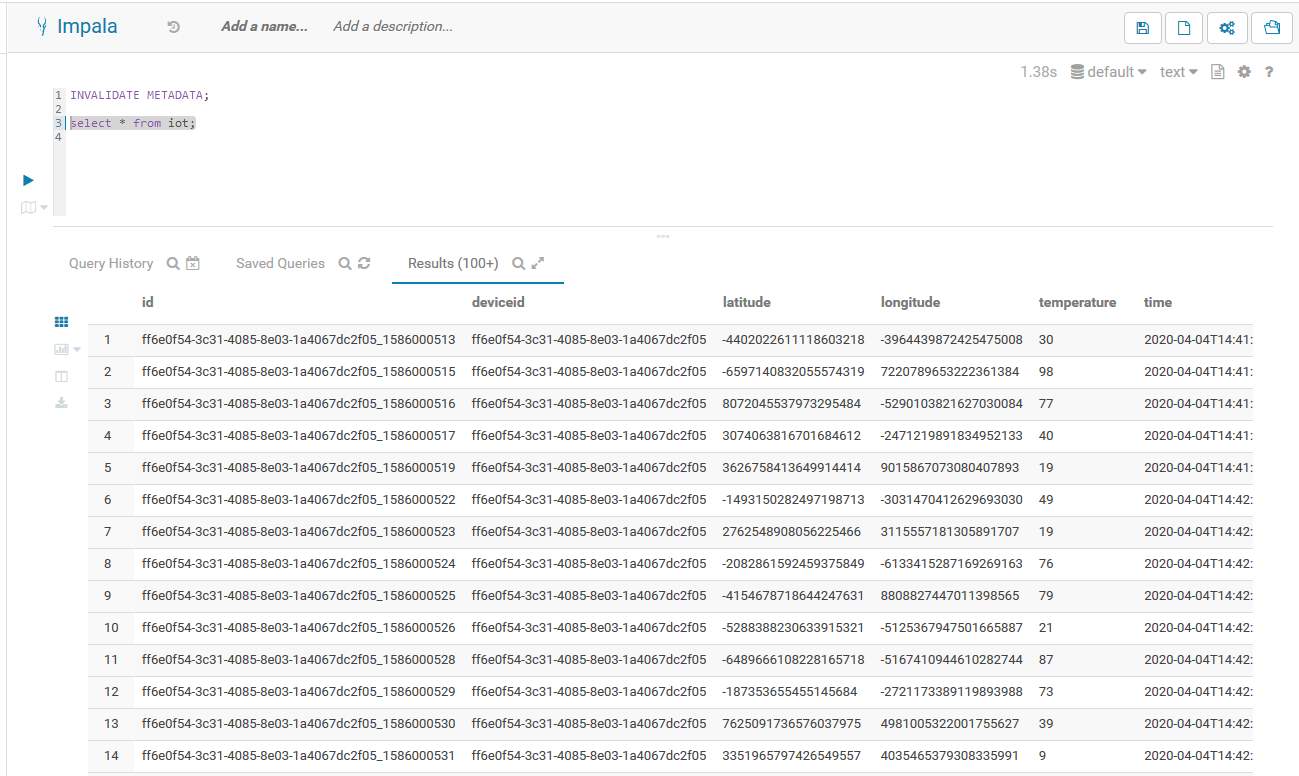
At the same time, data which was inserted into Hbase can be seen as below (After insert into HBase).



Metastore Table has been created on hive



Select test on Impala

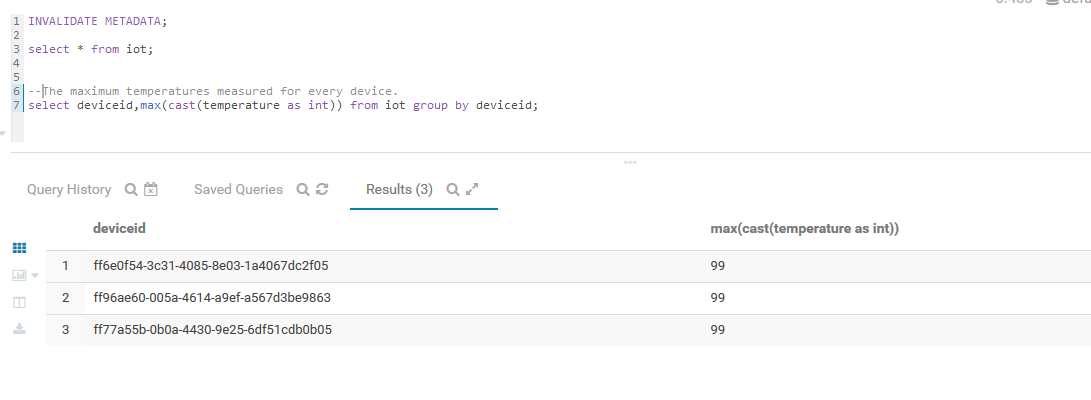


## Expected Data Analysis Queries

1. The maximum temperatures measured for every device

Impala Query:

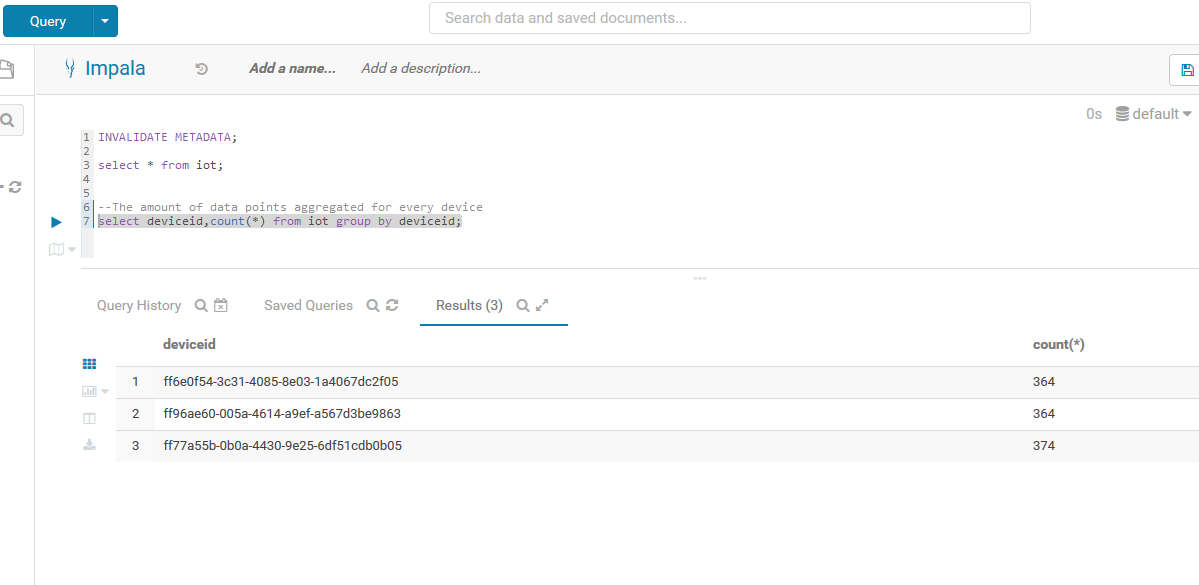
Select deviceid, max(cast(temperature as int)) from iot group by deviceid;



1. The amount of data points aggregated for every device

Impala Query:

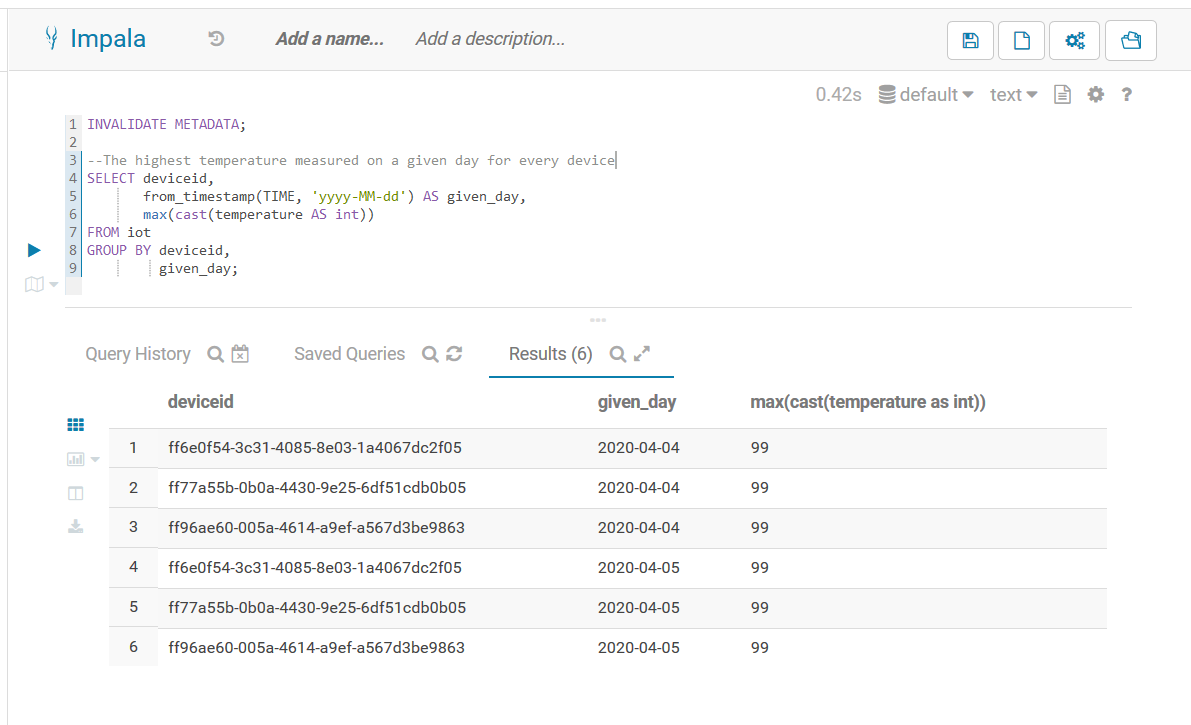
Select deviceid, count(\*) from iot group by deviceid;



1. The highest temperature measured on a given day for every device

Impala Query:

Select deviceid, from\_timestamp(time,’yyyy-MM-dd’) as given\_day, max(cast(temperature as int)) from iot group by deviceid,given\_day;



select deviceid,max(cast(temperature as int)) from iot where time like '2020-04-04%' group by deviceid

