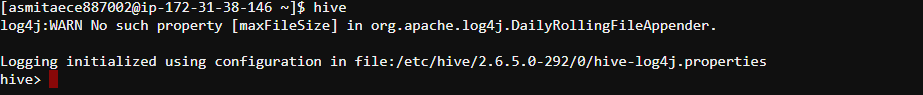
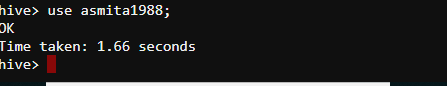
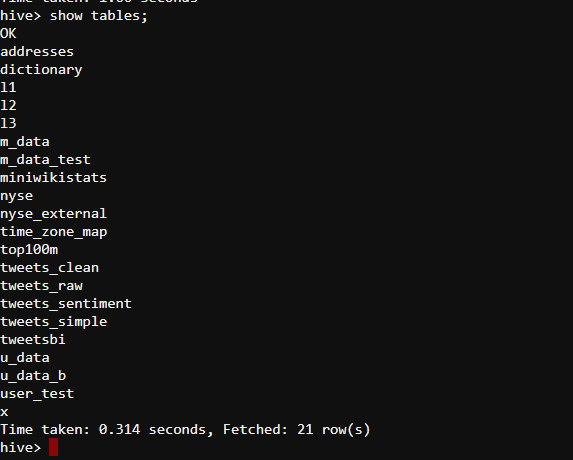
Example1:

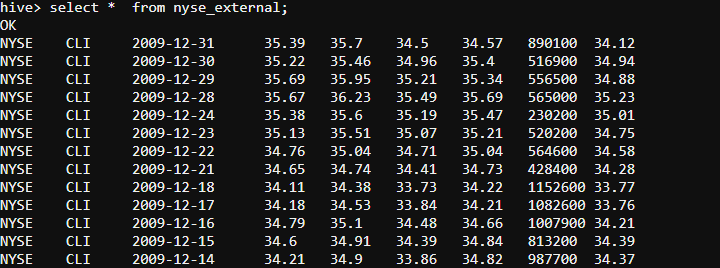
Login to Hive console :

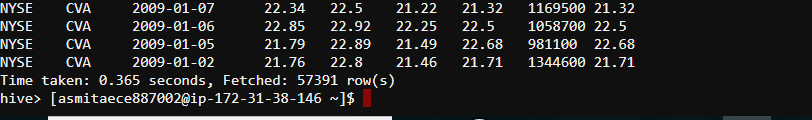


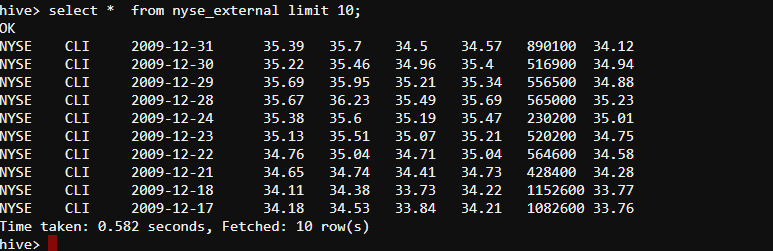
Use my database



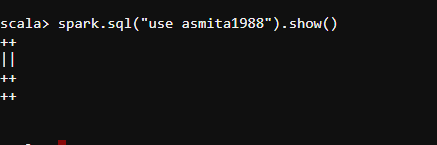


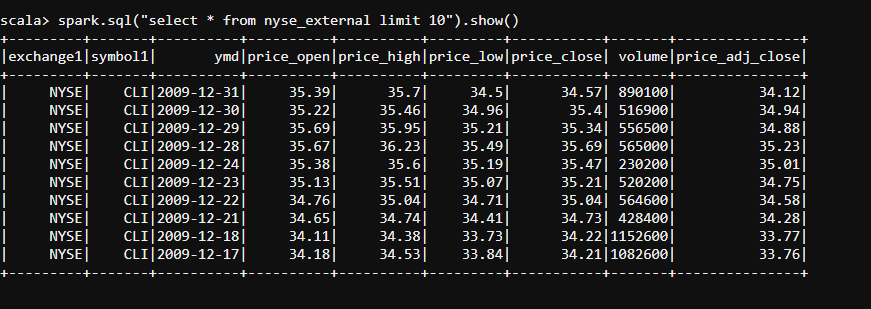






Now in another CLI , we open the scalaa prompt

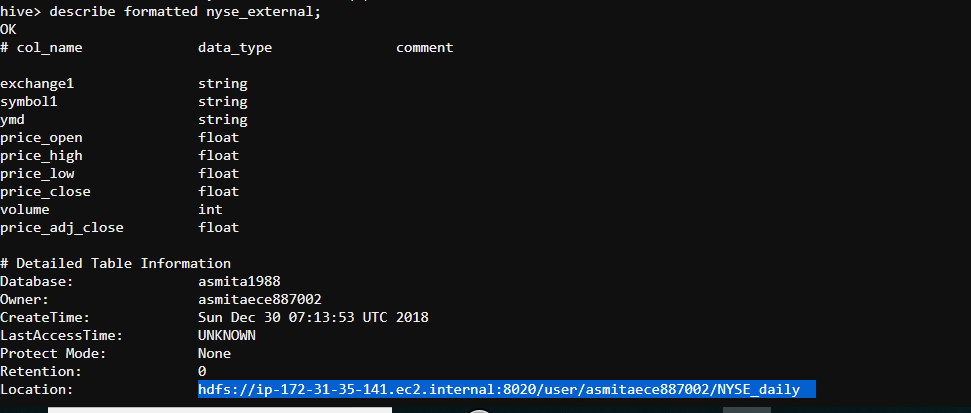




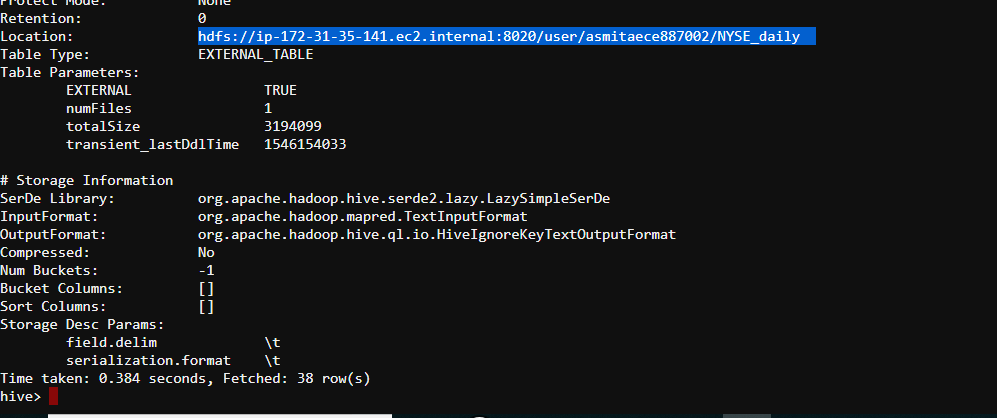
Hence the spark sql is not talking to hadoop at all , it directly finds out the query result from metastore

IF we check from HIVE -

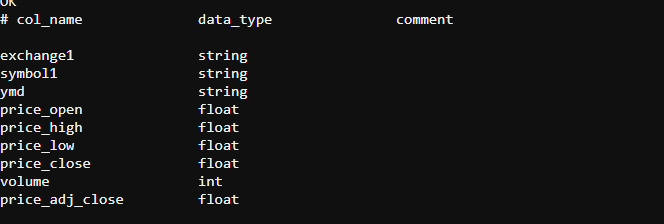
Describe formatted nyse\_external;



The location of the table is as follows :

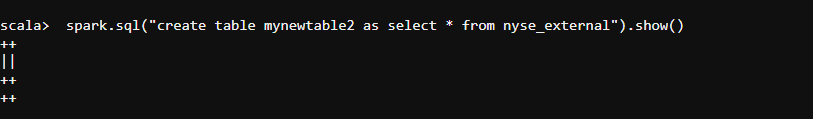


Below are the various columns formats :

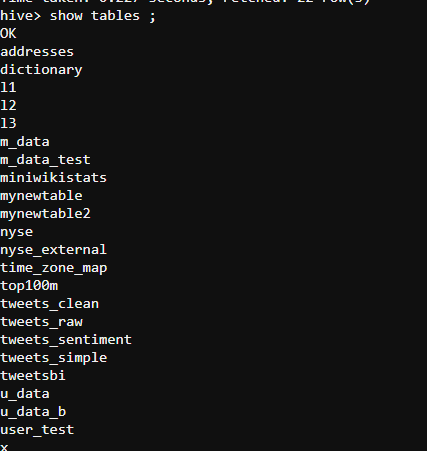


Now based on the column type , the dataframe was created in spark shell .

Now create a new table in spark sql --

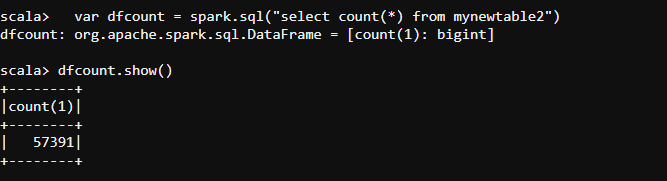


Now check in hive command

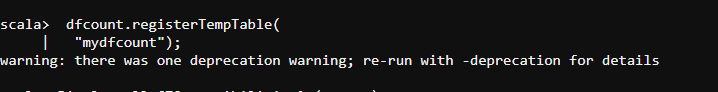


The table mynewtables2 is created

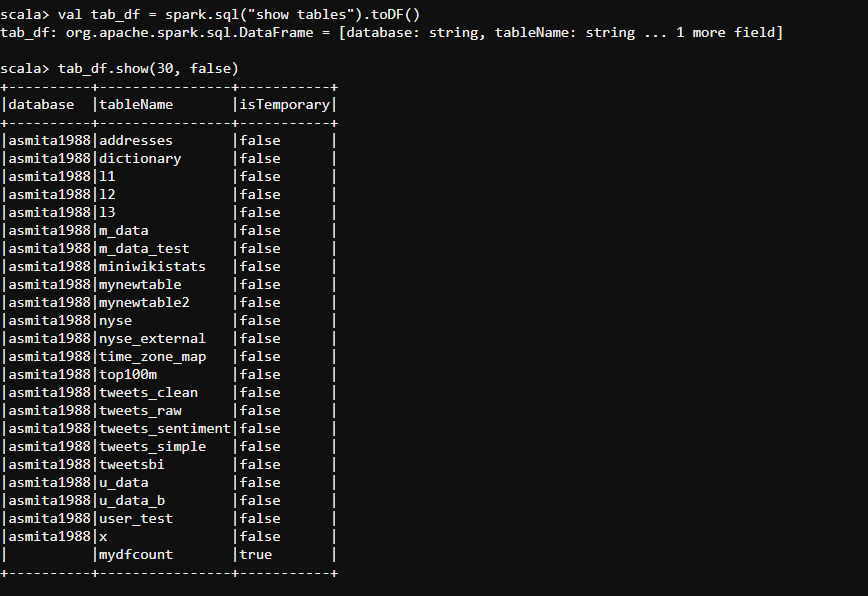
From spark – shell , check the count



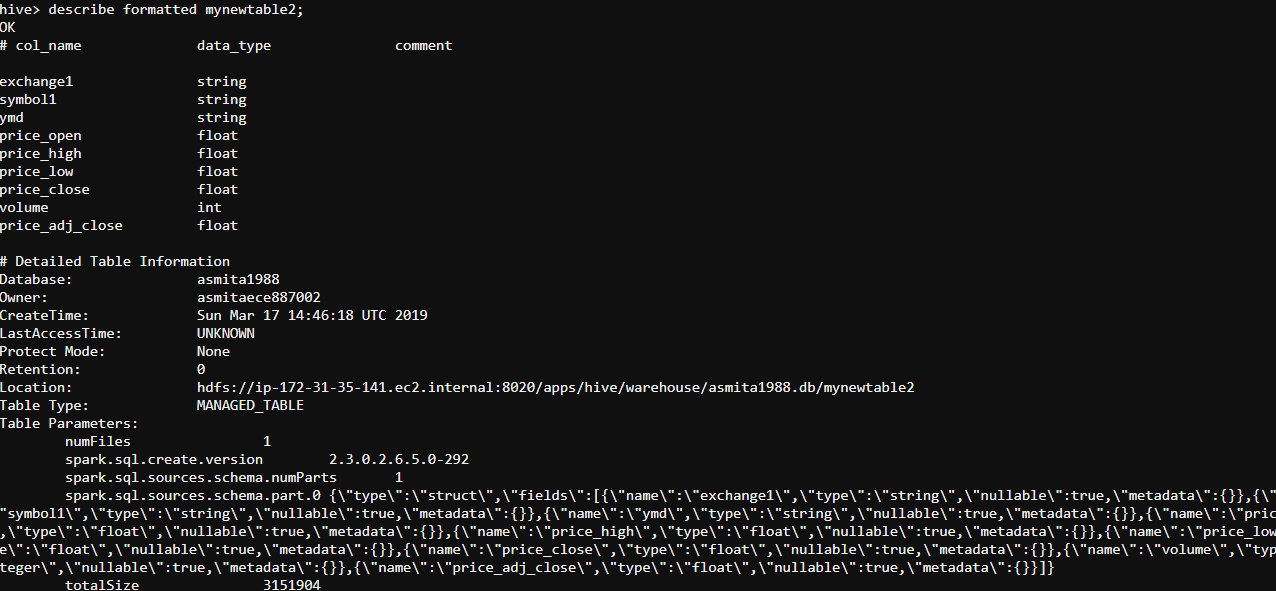
The dataframe is registered as a temporary view

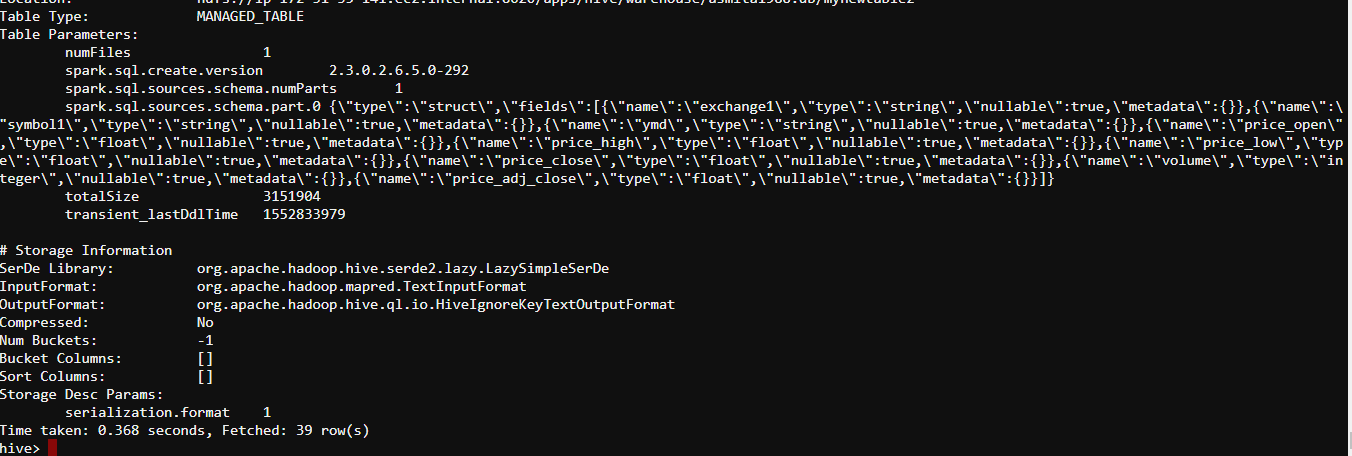


Now checking the mydfcount , its uts a cached temporary data in the memory



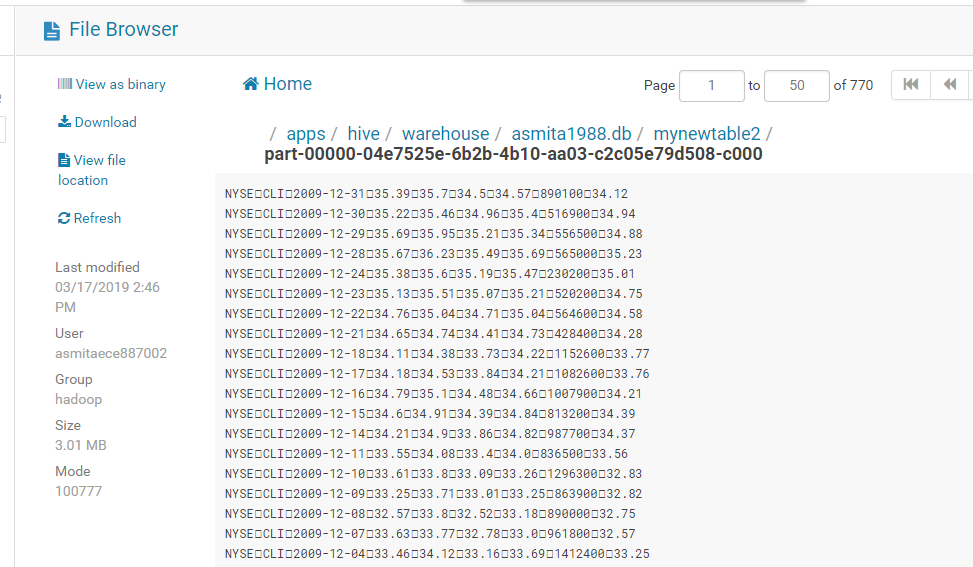
Now checking the new table metadata from hive





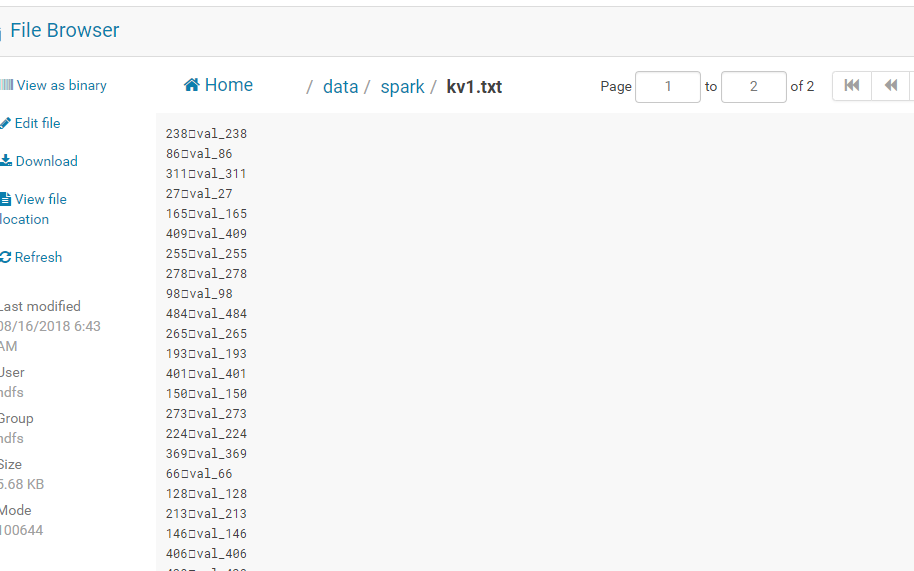
Location of the file :

hdfs://ip-172-31-35-141.ec2.internal:8020/apps/hive/warehouse/asmita1988.db/mynewtable2



Example2: Creating and loading key-value paired data in a hive table , by creating the table from spark sql

The data set attached is :



TO achieve the purpose ,

1)we need to import the Row class into spark-shell

Row class is used for mapping RDD schema

2) We need to Create a class ‘Record’ with attributes Int and String

3) We now build a Spark Session ‘spark’ to demonstrate Hive example in Spark SQL.

4)I mporting Implicits class into the shell

5) Importing SQL library into the Spark Shell.

6) Creating a table ‘src’ with columns to store key and value

import org.apache.spark.sql.Row

import org.apache.spark.sql.SparkSession

///declare the case class ///

case class Record(key: Int, value: String)

///declare a spark session for the hive example

val spark = SparkSession.builder().appName("Spark Hive Example").enableHiveSupport().getOrCreate()

////import the implicits

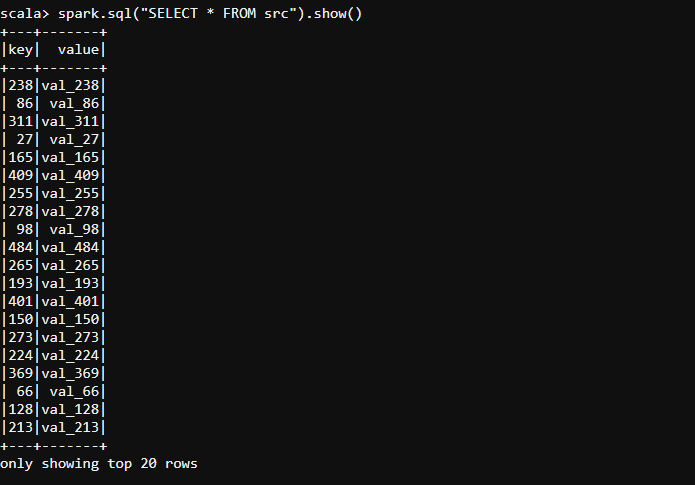
import spark.implicits.\_

import spark.sql

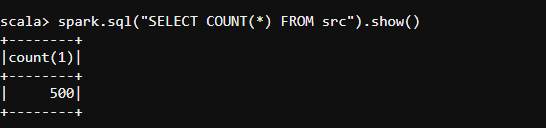
spark.sql("CREATE TABLE IF NOT EXISTS src (key INT, value STRING)")

spark.sql("LOAD DATA INPATH '/user/asmitaece887002/kv1.txt'OVERWRITE INTO TABLE src")

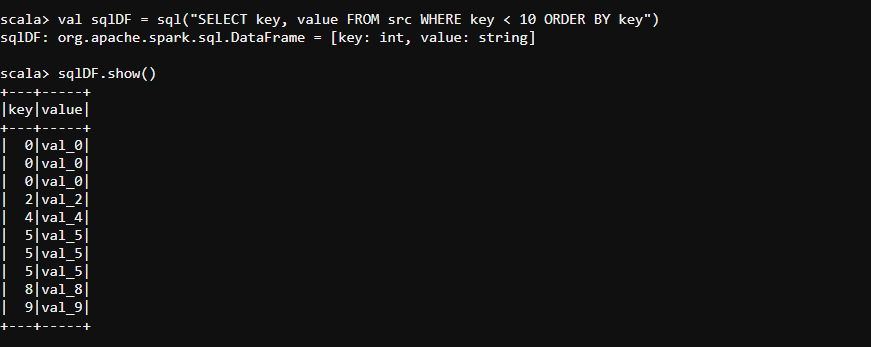
spark.sql("SELECT \* FROM src").show()



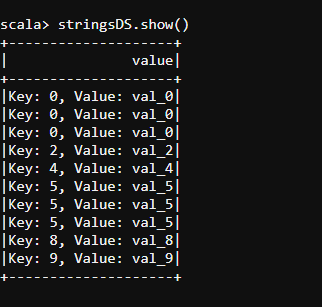
1. perform the ‘count’ operation to select the number of keys in ‘src’ table.



1. now select all the records with ‘key’ value less than 10 and store it in the ‘sqlDF’ DataFrame



c)Creating a Dataset ‘stringDS’ from ‘sqlDF’.  
Displaying the contents of ‘stringDS’ Dataset.



d) We create a DataFrame ‘recordsDF’ and store all the records with key values 1 to 100.

Create a temporary view ‘records’ of ‘recordsDF’ DataFrame.

Displaying the contents of the join of tables ‘records’ and ‘src’ with ‘key’ as the primary key

