CASE STUDY 3

**WORKING WITH SENSOR DATA**

there are two datasets in sensor data case study-

1. **building.csv** contains the details of the top 20 buildings all over the world and
2. **HVAC.csv** contains the target temperature and the actual temperature along with the BuildingID.

**HVAC** (heating, ventilating/ventilation, and air conditioning) is the technology of indoor and vehicularenvironmental comfort. Its goal is to provide thermal comfort and acceptable indoor air quality. Through the HVAC sensors, we will get the temperature of the buildings.

**Details:**

**Building.csv –**BuildingID, BuildingMgr, BuildingAge, HVACproduct, Country **HVAC.csv –**Date, Time, TargetTemp, ActualTemp, System, SystemAge, BuildingID.

**Objective-1**

* Load HVAC.csv file into temporary table.
* Add a new column, tempchange -set to 1, if there is a change of greater than +/-5 between actual and target temperature.

package Sensor\_Data\_Analysis

import org.apache.spark.sql.SparkSession

object Objective1

{

def main(args: Array[String]): Unit =

{

println("Sensor data analysis!!!")

// Use new SparkSession interface in Spark

val spark = SparkSession.builder().master("local").appName("Sensor Data")

.config("spark.some.config.option", "value").getOrCreate()

// load the dataset using the csvFile method

val hvac\_data = spark.read.format("com.databricks.spark.csv").option("header","true")

.option("inferSchema","true")

.load("E:\\Dataset\\HVAC.c sv")

//convert the hvac RDD into dataframe

val hvac\_data\_df = hvac\_data.toDF

//Register or load hvac dataframe into temporary table ‘hvacTemptable’ hvac\_data\_df.registerTempTable("hvacTempTable")

println("hvac dataframe is loaded in hvacTempTable !")

//use spark sql query to add extra column tempchange in hvacTempTable

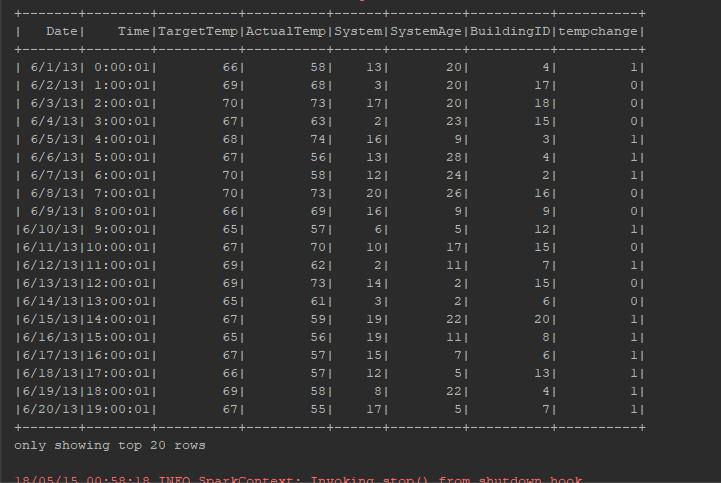
val hvac\_temp\_chnage = spark.sql("select \*,IF((targettemp - actualtemp) > 5, '1',

IF((targettemp - actualtemp) < -5, '1', 0)) AS tempchange from hvacTempTable")

hvac\_temp\_chnage.show()

} }

Below screen shot shows the ‘**hvacTempTable**’ with extra column added which shows the value 1 for temp-difference is equal to -/+5 and 0 for difference less than -/+5



**Objective-2**

* Load building.csv file into temporary table

package Sensor\_Data\_Analysis

import org.apache.spark.sql.SparkSession

object Objective2

{

def main(args: Array[String]): Unit =

{

println("Sensor data analysis!!!")

// Use new SparkSession interface in Spark

val spark = SparkSession .builder() .master("local").appName("Sensor Data")

.config("spark.some.config.option", "value") .getOrCreate()

// load the dataset using the csvFile method

val building\_data = spark .read.format("com.databricks.spark.csv") .option("header","true")

.option("inferSchema","true")

.load("E:\\ Dataset\\building

.csv")

//convert the hvac RDD into dataframe

val building\_data\_df =building\_data.toDF

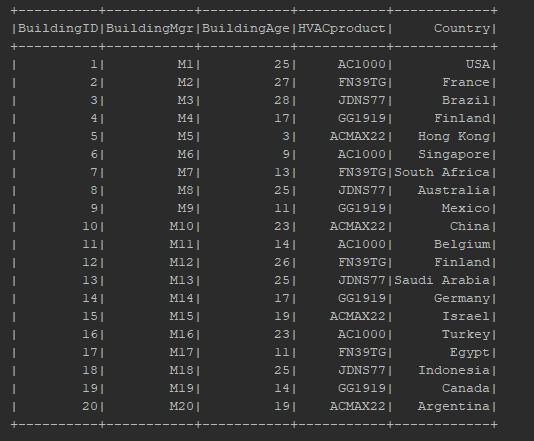
//Register or load hvac dataframe into temporary table ‘hvacTemptable’ building\_data\_df.registerTempTable("buildingTempTable")

//use spark sql to show the loaded building data in buildingTempTable

val load = spark.sql("select \* from buildingTempTable").show()

}}

Below screen shots shows the 20 rows of building data-set from ‘**bulidingTempTable**’



**Objective-3**

* Figure out the number of times, temperature has changed by 5 degrees or more for each country.

package Sensor\_Data\_Analysis

import org.apache.spark.sql.SparkSession

object Objective3

{

//declare a case class HVAC holding the dataset description of the hvac -data.

case class

HVAC(Date:String,Time:String,TargetTemp:Int,ActualTemp:Int,System:Int,SystemAge:Int,BuildingID:Int)

//declare a case class BHUILDING holding the dataset description of the building -data.

case class BUILDING(BuildingID:Int,BuildingMgr:String,BuildingAge:Int,HVAC\_Product:String,Country:String)

def main(args: Array[String]): Unit =

{

println("Sensor data analysis!!!")

* Use new SparkSession interface in spark

val spark = SparkSession .builder().master("local") .appName(“Sensor Data")

.config("spark.some.config.option", “value") .getOrCreate()

println("Spark Session is created !!!")

//load the hvac dataset using the textFile method

val hvac\_data\_with\_header =

spark.sparkContext.textFile("E:\\Dataset\\HVAC.csv")

//creating a variable header, which holds the first line of the datase t, in our data set hvac.csv the first line is a header line.

val header = hvac\_data\_with\_header.first()

//filter the header line from the dataset using the filter RDD

val hvac\_data = hvac\_data\_with\_header.filter(row => row != header)

//For implicit conversions like converting RDDs and sequences to DataFrames

import spark.implicits.\_

//preparing a structure for the data, mapping it to the case class structure, and finally converting it to a data frame.

val hvac\_data\_df = hvac\_data.map(x=>x.split(",")).map(x => HVAC(x(0),x(1),x(2).toInt,x(3).toInt,x(4).toInt,x(5).toInt,x(6).toInt)).toDF()

//Register temporary table hvacTempTable

hvac\_data\_df.registerTempTable("hvacTempTable")

//Use spark-sql queru to add extra xolum for temperature change

val hvac\_1 = spark.sql("select \*,IF((targettemp - actualtemp) > 5, '1', IF((targettemp - actualtemp) < -5, '1', 0)) AS tempchange from hvacTempTable")

// Register the new dataframe hvac1 into temporary table hvac1TempTable

hvac\_1.registerTempTable("hvac1TempTable")

println("Data Frame Registered as hvac1TempTable table !")

//load the building dataset using the textFile method

val building\_data\_with\_header =

spark.sparkContext.textFile("E:\\Dataset\\building.csv")

//creating a variable header, which holds the first line of the dataset, in our data set building.csv the first line is a header line

val header1 = building\_data\_with\_header.first()

//filter the header line from the dataset using the filter RDD

val building\_data = building\_data\_with\_header.filter(row => row != header1)

//preparing a structure for the data, mapping it to the case class structure, and finally converting it to a data frame.

val building\_data\_df = building\_data.map(x=> x.split(",")).map(x => BUILDING(x(0).toInt,x(1),x(2).toInt,x(3),x(4))).toDF

//Register temporary table

building\_data\_df.registerTempTable("buildingTempTable")

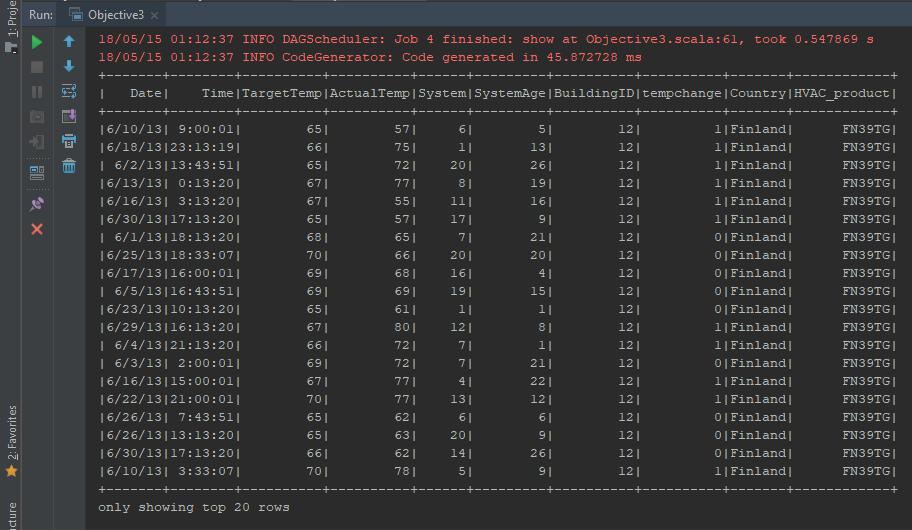
//use spark-sql query to join to tables to get columns which help to filter country and tempchange column

val join\_hvac\_building = spark.sql("select h.\*, b.Country, b.HVAC\_product from buildingTempTable b join hvac1TempTable h on b.BuildingID = h.BuildingID")

join\_hvac\_building.show()

Below two output tables shows the details for only country **Finland** as default it select the 20 rows and randomly its select the country Finland.

Below screen shot shows the join table of **hvacTempTable** and **buildingTemptable**

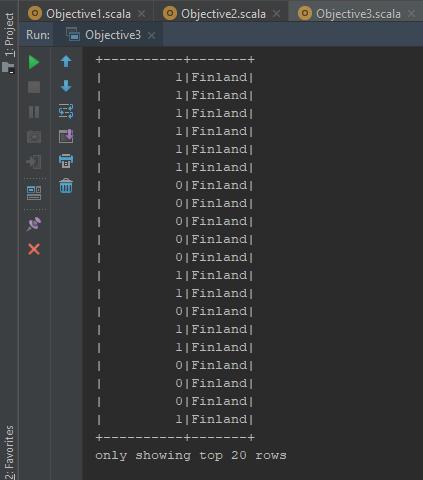


//Select temperature change and country column from above

val tempCountry = join\_hvac\_building.map(x => (new Integer(x(7).toString),x(8).toString))

tempCountry.show()

Below screen shot shows the two columns **tempchange** and **country**



//Filter the values 1 which for temperature change greater than 5 or more

val tempCountryOnes = tempCountry.filter(x=> {if(x.\_1==1) true else false})

tempCountryOnes.groupBy("\_2").count.withColumnRenamed("\_2","country" )show()

}

}

Below screen, shot shows the number of times temperature has changed by 5 degree or more for each country.

