ASSIGNMENT:CASE STUDY 3 - WORKING WITH SENSOR DATA

TASKS:

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

Objective - 1: Load building.csv file into temporary table

Objective - 2: Figure out the number of times, temperature has changed by 5 degrees or more for each country:

Objective - 3:

1) Join both the tables.
2) Select tempchange and country column
3) Filter the rows where tempchange is 1 and count the number of occurrence for each country

EXPLANATION: SO HERE TO DO SPARK SQL OPERATION WE ARE DEALING WITH DATAFRAME CREATION. SO FOR THAT WE NEED START "HIVE SERVICES" IN ONE TERMINAL AND IN ANOTHER TERMINAL WE START "SPARK" . THEN WE IMPORT NECESSARY PACKAGE BY ADDING HIVE CONTEXTS. SO WE ENTER AS BELOW:

```
CODE: import org.apache.spark.sql.hive._
      val sqlContext = new HiveContext(sc)
```

NOW WE NEED TO LOAD THE "HVAC.csc" DATASET FILE INTO SPARK WHICH IS IN "**PATH:/home/acadgild/HVAC.csv**". SO WE ENTER AS BELOW:

```
CODE: val data = sc.textFile("file:///home/acadgild/HVAC.csv")
```

NOW WE NEED TO REMOVE THE HEADER FROM THE DATASET WHICH IS NOT REQUIRED. SO FIRST WE LOAD THE HEADERS IN ONE VARIABLE AND FILTER IT OUT SO WE ENTER AS BELOW:

```
CODE: val header = data.first()
      val data1 = data.filter(row => row != header)
```

NOW WE CREATE A CASE CLASS FOR HOLDING THE SCHEMA FOR THE FIELDS IN THE DATASET LOADED. SO WE ENTER AS BELOW:

```
CODE: case class hvac_cls(Date:String, Time:String, TargetTemp:Int,
ActualTemp:Int, System:Int, SystemAge:Int, BuildingID:Int)
```

NOW WE NEED TO LOAD THE DATA INTO THE DATAFRAME. SO FOR THAT WE SPLIT EACH ROW OF DATASET WITH DELIMITER "," THEN WE MAP THE COLUMNS TO OUR CREATED CASE CLASS. THEN WE CONVERT IT INTO DATAFRAME. SO WE ENTER AS BELOWS:

```
CODE: val hvac = data1.map(x => x.split(",")).map(x => hvac_cls(x(0), x(1),
x(2).toInt, x(3).toInt, x(4).toInt, x(5).toInt, x(6).toInt)).toDF
```

NOW WE CREATED A TABLE FOR OUR DATAFRAME AS "HVAC" SO THAT WE CAN DO THE SQL QUERY OPERATION. AFTER CREATING A TABLE NOW WE NEED TO CREATE A NEW COLUMN WHICH WILL REPRESENT THE TEMPERATURE CHANGE AND WILL SET TO "1" IF THERE IS A CHANGE IN TEMPERATURE AS EITHER "+5 OR -5" AND THAT COLUMN WILL BE NAMED AS "tempchange". AFTER CREATING NEW COLUMN WE WILL CREATE ONE MORE TABLE AS "HVAC1" FOR THE NEW UPDATED COLUMN. SO FOR THAT WE DO FOLLOWING SQL QUERY AS BELOW:

```
CODE: hvac.registerTempTable("HVAC")
```

```
CODE: val hvac1 = sqlContext.sql("select *,IF((targettemp - actualtemp) >
5, '1', IF((targettemp - actualtemp) < -5, '1', 0)) AS tempchange from HVAC")
```

```
CODE: hvac1.registerTempTable("HVAC1")
```

NOW WE ARE DONE WITH LOADING "HVAC" DATASET AND OPERATIONS ON IT. NOW WE NEED TO LOAD THE "BUILDINGS" DATASET INTO SPARK. AND SO WE DO THE SAME OPERATIONS WHAT WE DONE BEFORE. WE ENTER FOLLOWING:

FILE PATH: /home/acadgild/building.csv

```
CODE: val data2 = sc.textFile("file:///home/acadgild/building.csv")
      val header1 = data2.first()
      val data3 = data2.filter(row => row != header1)
      case class
building(buildid:Int,buildmgr:String,buildAge:Int,hvacproduct:String,Co
untry:String)
      val build = data3.map(x=> x.split(",")).map(x =>
building(x(0).toInt,x(1),x(2).toInt,x(3),x(4))).toDF
      build.registerTempTable("building")
```

NOW FOR DOING THE "OBJECTIVE-3". WE HAVE TO JOIN THE TABLES "building" AND "HVAC1" USING "buildingId". SO WE ENTER THE AS BELOW:

```
CODE: val build1 = sqlContext.sql("select h.*, b.country, b.hvacproduct
from building b join HVAC1 h on b.buildid = h.buildingid")
```

NOW WE NEED TO TAKE COLUMNS "tempchange" AND "country" AND SAVE IT VARIABLE "test". SO WE ENTER AS BELOW:

```
CODE: val test = build1.map(x => (new
Integer(x(7).toString),x(8).toString))
```

NOW WE NEED TO FILTER OUT THE ROWS FROM "test" WHICH HAVE A TEMPERATURE CHANGE EQUAL TO 1. SO WE USE "IF-ELSE STATEMENT" TO DO SO WHICH WILL RETURN TRUE OR FALSE BASED ON THE STATMENT PROVIDED AND FILTER OUT THE ROWS WHICH IS NOT REQUIRED. SO WE ENTER AS BELOW:

```
CODE: val test1 = test.filter(x=> {if(x._1==1) true else false})
```

SO NOW WE TAKE THE COLUMN COUNTRY USING THE "GROUPBY" AND COUNT THE OCCURENCES OF EACH COUNTRY WHICH IS IDENTIFIED BY 1 AND SHOW THE RESULT.

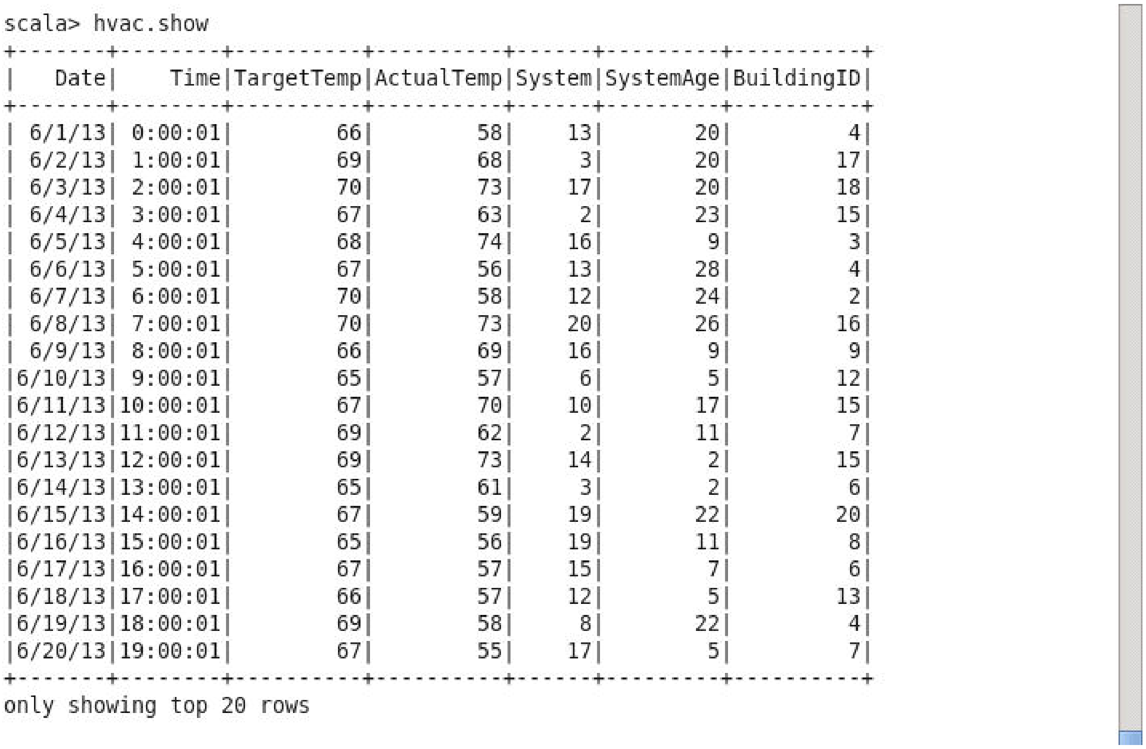
CODE: val test2 = test1.groupBy("_2").count.show

SOLUTION REPORT:

```
scala> import org.apache.spark.sql.hive._
import org.apache.spark.sql.hive._

scala> val sqlContext = new HiveContext(sc)
warning: there was one deprecation warning; re-run with -deprecation for
details
sqlContext: org.apache.spark.sql.hive.HiveContext =
org.apache.spark.sql.hive.HiveContext@4404a6b

scala> val data = sc.textFile("file:///home/acadgild/HVAC.csv")
data: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/HVAC.csv
MapPartitionsRDD[1] at textFile at <console>:31

scala> val header = data.first()
header: String = Date, Time, TargetTemp, ActualTemp, System, SystemAge,
BuildingID

scala> val data1 = data.filter(row => row != header)
data1: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at filter at
<console>:35

scala> case class hvac_cls(Date:String, Time:String, TargetTemp:Int,
ActualTemp:Int, System:Int, SystemAge:Int, BuildingID:Int)
defined class hvac_cls

scala> val hvac = data1.map(x => x.split(",")).map(x => hvac_cls(x(0),
x(1), x(2).toInt, x(3).toInt, x(4).toInt, x(5).toInt, x(6).toInt)).toDF
Thu Mar 14 12:28:49 IST 2019 WARN: Establishing SSL connection without
server's identity verification is not recommended. According to MySQL
5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established
by default if explicit option isn't set. For compliance with existing
applications not using SSL the verifyServerCertificate property is set to
'false'. You need either to explicitly disable SSL by setting useSSL=false,
or set useSSL=true and provide truststore for server certificate
verification.
Thu Mar 14 12:28:52 IST 2019 WARN: Establishing SSL connection without
server's identity verification is not recommended. According to MySQL
5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established
by default if explicit option isn't set. For compliance with existing
applications not using SSL the verifyServerCertificate property is set to
'false'. You need either to explicitly disable SSL by setting useSSL=false,
or set useSSL=true and provide truststore for server certificate
verification.
Thu Mar 14 12:28:53 IST 2019 WARN: Establishing SSL connection without
server's identity verification is not recommended. According to MySQL
5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established
by default if explicit option isn't set. For compliance with existing
applications not using SSL the verifyServerCertificate property is set to
'false'. You need either to explicitly disable SSL by setting useSSL=false,
or set useSSL=true and provide truststore for server certificate
verification.
Thu Mar 14 12:28:53 IST 2019 WARN: Establishing SSL connection without
server's identity verification is not recommended. According to MySQL
5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established
by default if explicit option isn't set. For compliance with existing
applications not using SSL the verifyServerCertificate property is set to
'false'. You need either to explicitly disable SSL by setting useSSL=false,
or set useSSL=true and provide truststore for server certificate
verification.
Thu Mar 14 12:28:54 IST 2019 WARN: Establishing SSL connection without
server's identity verification is not recommended. According to MySQL
5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established
by default if explicit option isn't set. For compliance with existing
applications not using SSL the verifyServerCertificate property is set to
'false'. You need either to explicitly disable SSL by setting useSSL=false,
or set useSSL=true and provide truststore for server certificate
verification.
Thu Mar 14 12:28:54 IST 2019 WARN: Establishing SSL connection without
server's identity verification is not recommended. According to MySQL
5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established
by default if explicit option isn't set. For compliance with existing
applications not using SSL the verifyServerCertificate property is set to
'false'. You need either to explicitly disable SSL by setting useSSL=false,
or set useSSL=true and provide truststore for server certificate
verification.
Thu Mar 14 12:28:55 IST 2019 WARN: Establishing SSL connection without
server's identity verification is not recommended. According to MySQL
5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established
by default if explicit option isn't set. For compliance with existing
applications not using SSL the verifyServerCertificate property is set to
'false'. You need either to explicitly disable SSL by setting useSSL=false,
or set useSSL=true and provide truststore for server certificate
verification.
Thu Mar 14 12:28:55 IST 2019 WARN: Establishing SSL connection without
server's identity verification is not recommended. According to MySQL
5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established
by default if explicit option isn't set. For compliance with existing
applications not using SSL the verifyServerCertificate property is set to
'false'. You need either to explicitly disable SSL by setting useSSL=false,
or set useSSL=true and provide truststore for server certificate
verification.
19/03/14 12:29:09 WARN metastore.ObjectStore: Failed to get database
global_temp, returning NoSuchObjectException
hvac: org.apache.spark.sql.DataFrame = [Date: string, Time: string ... 5
more fields]

scala> hvac.show
+-------+--------+----------+----------+------+---------+----------+
|   Date|    Time|TargetTemp|ActualTemp|System|SystemAge|BuildingID|
+-------+--------+----------+----------+------+---------+----------+
| 6/1/13| 0:00:01|        66|        58|    13|       20|         4|
| 6/2/13| 1:00:01|        69|        68|     3|       20|        17|
| 6/3/13| 2:00:01|        70|        73|    17|       20|        18|
| 6/4/13| 3:00:01|        67|        63|     2|       23|        15|
| 6/5/13| 4:00:01|        68|        74|    16|        9|         3|
| 6/6/13| 5:00:01|        67|        56|    13|       28|         4|
| 6/7/13| 6:00:01|        70|        58|    12|       24|         2|
| 6/8/13| 7:00:01|        70|        73|    20|       26|        16|
| 6/9/13| 8:00:01|        66|        69|    16|        9|         9|
|6/10/13| 9:00:01|        65|        57|     6|        5|        12|
|6/11/13|10:00:01|        67|        70|    10|       17|        15|
|6/12/13|11:00:01|        69|        62|     2|       11|         7|
|6/13/13|12:00:01|        69|        73|    14|        2|        15|
|6/14/13|13:00:01|        65|        61|     3|        2|         6|
|6/15/13|14:00:01|        67|        59|    19|       22|        20|
|6/16/13|15:00:01|        65|        56|    19|       11|         8|
|6/17/13|16:00:01|        67|        57|    15|        7|         6|
|6/18/13|17:00:01|        66|        57|    12|        5|        13|
|6/19/13|18:00:01|        69|        58|     8|       22|         4|
|6/20/13|19:00:01|        67|        55|    17|        5|         7|
+-------+--------+----------+----------+------+---------+----------+
only showing top 20 rows

scala> hvac.registerTempTable("HVAC")
warning: there was one deprecation warning; re-run with -deprecation for
details

scala> val hvac1 = sqlContext.sql("select *,IF((targettemp - actualtemp)
> 5, '1', IF((targettemp - actualtemp) < -5, '1', 0)) AS tempchange from
HVAC")
hvac1: org.apache.spark.sql.DataFrame = [Date: string, Time: string ... 6
more fields]

scala> hvac1.show
+-------+--------+----------+----------+------+---------+----------+---
-------+
|   Date|
Time|TargetTemp|ActualTemp|System|SystemAge|BuildingID|tempchange|
+-------+--------+----------+----------+------+---------+----------+---
-------+
| 6/1/13| 0:00:01|        66|        58|    13|       20|         4|
1|
| 6/2/13| 1:00:01|        69|        68|     3|       20|        17|
0|
| 6/3/13| 2:00:01|        70|        73|    17|       20|        18|
0|
| 6/4/13| 3:00:01|        67|        63|     2|       23|        15|
0|
| 6/5/13| 4:00:01|        68|        74|    16|        9|         3|
1|
| 6/6/13| 5:00:01|        67|        56|    13|       28|         4|
1|
| 6/7/13| 6:00:01|        70|        58|    12|       24|         2|
1|
| 6/8/13| 7:00:01|        70|        73|    20|       26|        16|
0|
| 6/9/13| 8:00:01|        66|        69|    16|        9|         9|
0|
|6/10/13| 9:00:01|        65|        57|     6|        5|        12|
1|
|6/11/13|10:00:01|        67|        70|    10|       17|        15|
0|
|6/12/13|11:00:01|        69|        62|     2|       11|         7|
1|
|6/13/13|12:00:01|        69|        73|    14|        2|        15|
0|
|6/14/13|13:00:01|        65|        61|     3|        2|         6|
0|
|6/15/13|14:00:01|        67|        59|    19|       22|        20|
1|
|6/16/13|15:00:01|        65|        56|    19|       11|         8|
1|
|6/17/13|16:00:01|        67|        57|    15|        7|         6|
1|
|6/18/13|17:00:01|        66|        57|    12|        5|        13|
1|
|6/19/13|18:00:01|        69|        58|     8|       22|         4|
1|
|6/20/13|19:00:01|        67|        55|    17|        5|         7|
1|
+-------+--------+----------+----------+------+---------+----------+---
-------+
only showing top 20 rows

scala> hvac1.registerTempTable("HVAC1")
warning: there was one deprecation warning; re-run with -deprecation for
details

scala> val data2 = sc.textFile("file:///home/acadgild/building.csv")
data2: org.apache.spark.rdd.RDD[String] =
file:///home/acadgild/building.csv MapPartitionsRDD[6] at textFile at
<console>:30

scala> val header1 = data2.first()
header1: String = BuildingID,BuildingMgr,BuildingAge,HVACproduct,Country

scala> val data3 = data2.filter(row => row != header1)
data3: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[7] at filter at
<console>:34

scala> case class
building(buildid:Int,buildmgr:String,buildAge:Int,hvacproduct:String,Co
untry:String)
defined class building

scala> val build = data3.map(x=> x.split(",")).map(x =>
building(x(0).toInt,x(1),x(2).toInt,x(3),x(4))).toDF
build: org.apache.spark.sql.DataFrame = [buildid: int, buildmgr: string
... 3 more fields]

scala> build.show
+-------+--------+--------+-----------+------------+
|buildid|buildmgr|buildAge|hvacproduct|     Country|
+-------+--------+--------+-----------+------------+
|      1|      M1|      25|     AC1000|         USA|
|      2|      M2|      27|     FN39TG|      France|
|      3|      M3|      28|     JDNS77|      Brazil|
|      4|      M4|      17|     GG1919|     Finland|
|      5|      M5|       3|    ACMAX22|   Hong Kong|
|      6|      M6|       9|     AC1000|   Singapore|
|      7|      M7|      13|     FN39TG|South Africa|
|      8|      M8|      25|     JDNS77|   Australia|
|      9|      M9|      11|     GG1919|      Mexico|
|     10|     M10|      23|    ACMAX22|       China|
|     11|     M11|      14|     AC1000|     Belgium|
|     12|     M12|      26|     FN39TG|     Finland|
|     13|     M13|      25|     JDNS77|Saudi Arabia|
|     14|     M14|      17|     GG1919|     Germany|
|     15|     M15|      19|    ACMAX22|      Israel|
|     16|     M16|      23|     AC1000|      Turkey|
|     17|     M17|      11|     FN39TG|       Egypt|
|     18|     M18|      25|     JDNS77|   Indonesia|
|     19|     M19|      14|     GG1919|      Canada|
|     20|     M20|      19|    ACMAX22|   Argentina|
+-------+--------+--------+-----------+------------+

scala> build.registerTempTable("building")
warning: there was one deprecation warning; re-run with -deprecation for
details

scala> val build1 = sqlContext.sql("select h.*, b.country, b.hvacproduct
from building b join HVAC1 h on b.buildid = h.buildingid")
build1: org.apache.spark.sql.DataFrame = [Date: string, Time: string ...
7 more fields]

scala> build1.show
+-------+--------+----------+----------+------+---------+----------+---
-------+-------+-----------+
|   Date|
Time|TargetTemp|ActualTemp|System|SystemAge|BuildingID|tempchange|count
ry|hvacproduct|
+-------+--------+----------+----------+------+---------+----------+---
-------+-------+-----------+
|6/10/13| 9:00:01|        65|        57|     6|        5|        12|
1|Finland|     FN39TG|
|6/18/13|23:13:19|        66|        75|     1|       13|        12|
1|Finland|     FN39TG|
| 6/2/13|13:43:51|        65|        72|    20|       26|        12|
1|Finland|     FN39TG|
|6/13/13| 0:13:20|        67|        77|     8|       19|        12|
1|Finland|     FN39TG|
|6/16/13| 3:13:20|        67|        55|    11|       16|        12|
1|Finland|     FN39TG|
|6/30/13|17:13:20|        65|        57|    17|        9|        12|
1|Finland|     FN39TG|
| 6/1/13|18:13:20|        68|        65|     7|       21|        12|
0|Finland|     FN39TG|
|6/25/13|18:33:07|        70|        66|    20|       20|        12|
0|Finland|     FN39TG|
|6/17/13|16:00:01|        69|        68|    16|        4|        12|
0|Finland|     FN39TG|
| 6/5/13|16:43:51|        69|        69|    19|       15|        12|
0|Finland|     FN39TG|
|6/23/13|10:13:20|        65|        61|     1|        1|        12|
0|Finland|     FN39TG|
|6/29/13|16:13:20|        67|        80|    12|        8|        12|
1|Finland|     FN39TG|
| 6/4/13|21:13:20|        66|        72|     7|        1|        12|
1|Finland|     FN39TG|
| 6/3/13| 2:00:01|        69|        72|     7|       21|        12|
0|Finland|     FN39TG|
|6/16/13|15:00:01|        67|        77|     4|       22|        12|
1|Finland|     FN39TG|
|6/22/13|21:00:01|        70|        77|    13|       12|        12|
1|Finland|     FN39TG|
|6/26/13| 7:43:51|        65|        62|     6|        6|        12|
0|Finland|     FN39TG|
|6/26/13|13:13:20|        65|        63|    20|        9|        12|
0|Finland|     FN39TG|
|6/30/13|17:13:20|        66|        62|    14|       26|        12|
0|Finland|     FN39TG|
|6/10/13| 3:33:07|        70|        78|     5|        9|        12|
1|Finland|     FN39TG|
+-------+--------+----------+----------+------+---------+----------+---
-------+-------+-----------+
only showing top 20 rows

scala> val test = build1.map(x => (new
Integer(x(7).toString),x(8).toString))
test: org.apache.spark.sql.Dataset[(Integer, String)] = [_1: int, _2:
string]

scala> test.show
+---+-------+
| _1|     _2|
+---+-------+
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  0|Finland|
|  0|Finland|
|  0|Finland|
|  0|Finland|
|  0|Finland|
|  1|Finland|
|  1|Finland|
|  0|Finland|
|  1|Finland|
|  1|Finland|
|  0|Finland|
|  0|Finland|
|  0|Finland|
|  1|Finland|
+---+-------+
only showing top 20 rows

scala> val test1 = test.filter(x => {if(x._1 == 1) true else false})
test1: org.apache.spark.sql.Dataset[(Integer, String)] = [_1: int, _2:
string]

scala> test1.show
+---+-------+
| _1|     _2|
+---+-------+
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
+---+-------+
only showing top 20 rows

scala> val test2 = test1.groupBy("_2").count.show
+------------+-----+
|          _2|count|
+------------+-----+
|   Singapore|  230|
|      Turkey|  243|
|     Germany|  196|
|      France|  251|
|   Argentina|  230|
|     Belgium|  199|
|     Finland|  473|
|       China|  241|
|   Hong Kong|  248|
|      Israel|  232|
|         USA|  213|
|      Mexico|  228|
|   Indonesia|  243|
|Saudi Arabia|  233|
|      Canada|  232|
|      Brazil|  226|
|   Australia|  225|
|       Egypt|  236|
|South Africa|  237|
+------------+-----+

test2: Unit = ()
```

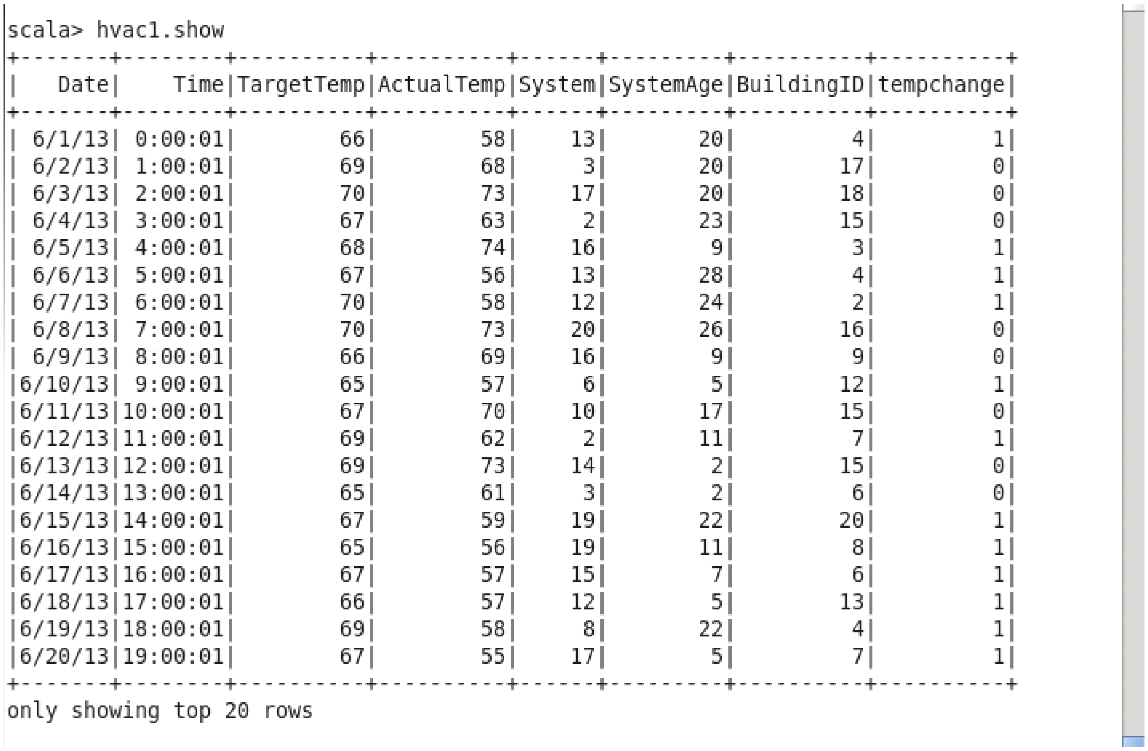
OUTPUT:

hvac.show



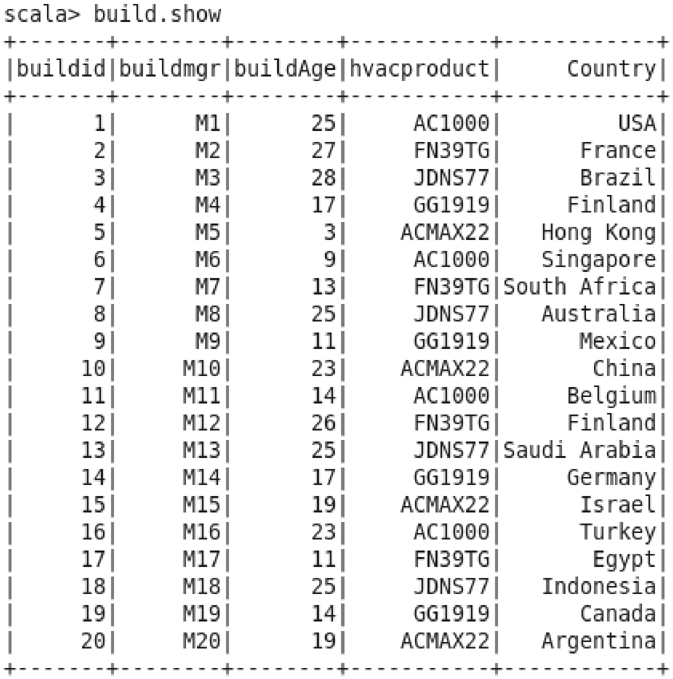
```
scala> hvac.show
+-------+--------+----------+----------+------+---------+----------+
|   Date|    Time|TargetTemp|ActualTemp|System|SystemAge|BuildingID|
+-------+--------+----------+----------+------+---------+----------+
| 6/1/13| 0:00:01|        66|        58|    13|       20|         4|
| 6/2/13| 1:00:01|        69|        68|     3|       20|        17|
| 6/3/13| 2:00:01|        70|        73|    17|       20|        18|
| 6/4/13| 3:00:01|        67|        63|     2|       23|        15|
| 6/5/13| 4:00:01|        68|        74|    16|        9|         3|
| 6/6/13| 5:00:01|        67|        56|    13|       28|         4|
| 6/7/13| 6:00:01|        70|        58|    12|       24|         2|
| 6/8/13| 7:00:01|        70|        73|    20|       26|        16|
| 6/9/13| 8:00:01|        66|        69|    16|        9|         9|
|6/10/13| 9:00:01|        65|        57|     6|        5|        12|
|6/11/13|10:00:01|        67|        70|    10|       17|        15|
|6/12/13|11:00:01|        69|        62|     2|       11|         7|
|6/13/13|12:00:01|        69|        73|    14|        2|        15|
|6/14/13|13:00:01|        65|        61|     3|        2|         6|
|6/15/13|14:00:01|        67|        59|    19|       22|        20|
|6/16/13|15:00:01|        65|        56|    19|       11|         8|
|6/17/13|16:00:01|        67|        57|    15|        7|         6|
|6/18/13|17:00:01|        66|        57|    12|        5|        13|
|6/19/13|18:00:01|        69|        58|     8|       22|         4|
|6/20/13|19:00:01|        67|        55|    17|        5|         7|
+-------+--------+----------+----------+------+---------+----------+
only showing top 20 rows
```

hvac1.show



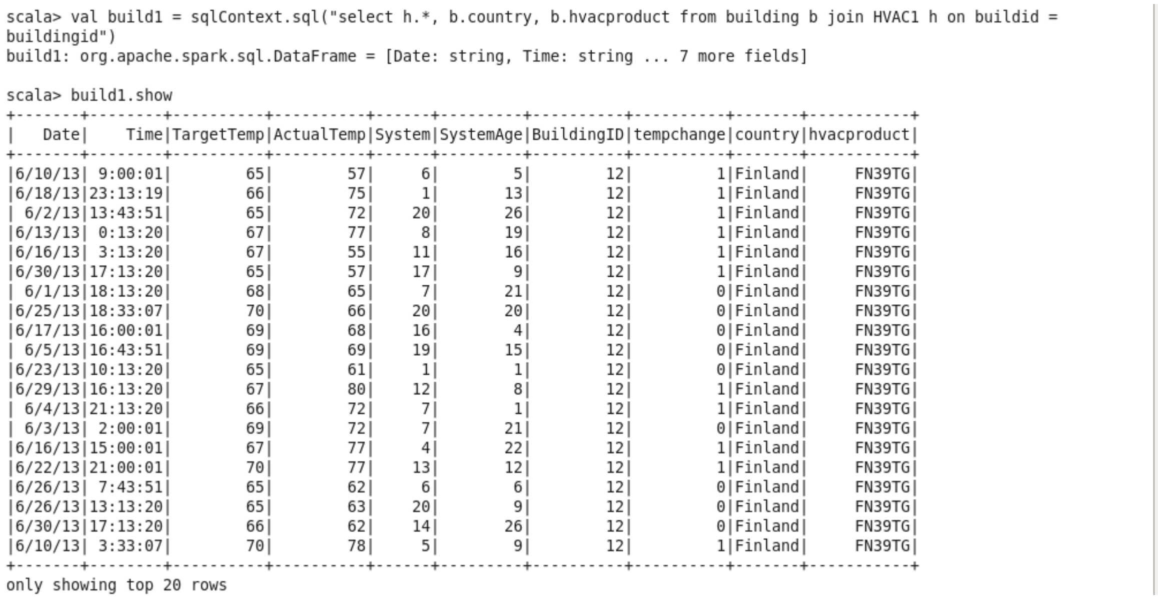
```
scala> hvac1.show
+-------+--------+----------+----------+------+---------+----------+----------+
|   Date|    Time|TargetTemp|ActualTemp|System|SystemAge|BuildingID|tempchange|
+-------+--------+----------+----------+------+---------+----------+----------+
| 6/1/13| 0:00:01|        66|        58|    13|       20|         4|         1|
| 6/2/13| 1:00:01|        69|        68|     3|       20|        17|         0|
| 6/3/13| 2:00:01|        70|        73|    17|       20|        18|         0|
| 6/4/13| 3:00:01|        67|        63|     2|       23|        15|         0|
| 6/5/13| 4:00:01|        68|        74|    16|        9|         3|         1|
| 6/6/13| 5:00:01|        67|        56|    13|       28|         4|         1|
| 6/7/13| 6:00:01|        70|        58|    12|       24|         2|         1|
| 6/8/13| 7:00:01|        70|        73|    20|       26|        16|         0|
| 6/9/13| 8:00:01|        66|        69|    16|        9|         9|         0|
|6/10/13| 9:00:01|        65|        57|     6|        5|        12|         1|
|6/11/13|10:00:01|        67|        70|    10|       17|        15|         0|
|6/12/13|11:00:01|        69|        62|     2|       11|         7|         1|
|6/13/13|12:00:01|        69|        73|    14|        2|        15|         0|
|6/14/13|13:00:01|        65|        61|     3|        2|         6|         0|
|6/15/13|14:00:01|        67|        59|    19|       22|        20|         1|
|6/16/13|15:00:01|        65|        56|    19|       11|         8|         1|
|6/17/13|16:00:01|        67|        57|    15|        7|         6|         1|
|6/18/13|17:00:01|        66|        57|    12|        5|        13|         1|
|6/19/13|18:00:01|        69|        58|     8|       22|         4|         1|
|6/20/13|19:00:01|        67|        55|    17|        5|         7|         1|
+-------+--------+----------+----------+------+---------+----------+----------+
only showing top 20 rows
```

build.show



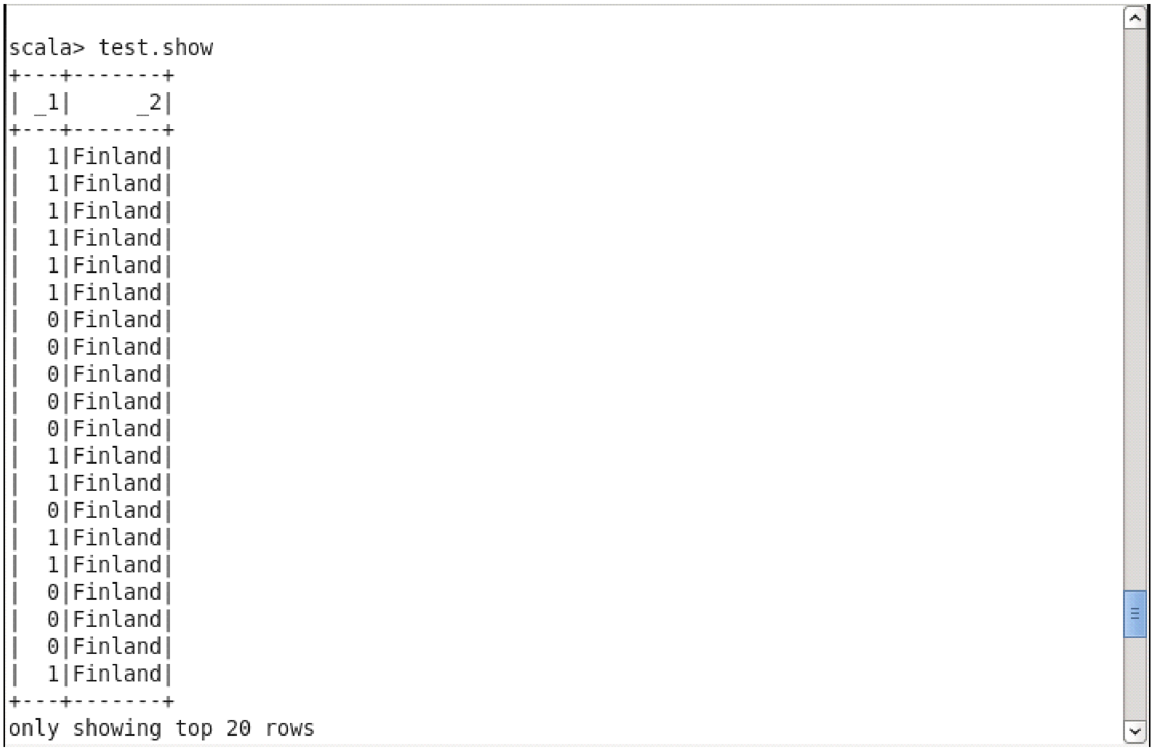
```
scala> build.show
+-------+--------+--------+-----------+------------+
|buildid|buildmgr|buildAge|hvacproduct|     Country|
+-------+--------+--------+-----------+------------+
|      1|      M1|      25|     AC1000|         USA|
|      2|      M2|      27|     FN39TG|      France|
|      3|      M3|      28|     JDNS77|      Brazil|
|      4|      M4|      17|     GG1919|     Finland|
|      5|      M5|       3|    ACMAX22|   Hong Kong|
|      6|      M6|       9|     AC1000|   Singapore|
|      7|      M7|      13|     FN39TG|South Africa|
|      8|      M8|      25|     JDNS77|   Australia|
|      9|      M9|      11|     GG1919|      Mexico|
|     10|     M10|      23|    ACMAX22|       China|
|     11|     M11|      14|     AC1000|     Belgium|
|     12|     M12|      26|     FN39TG|     Finland|
|     13|     M13|      25|     JDNS77|Saudi Arabia|
|     14|     M14|      17|     GG1919|     Germany|
|     15|     M15|      19|    ACMAX22|      Israel|
|     16|     M16|      23|     AC1000|      Turkey|
|     17|     M17|      11|     FN39TG|       Egypt|
|     18|     M18|      25|     JDNS77|   Indonesia|
|     19|     M19|      14|     GG1919|      Canada|
|     20|     M20|      19|    ACMAX22|   Argentina|
+-------+--------+--------+-----------+------------+
```

build1.show



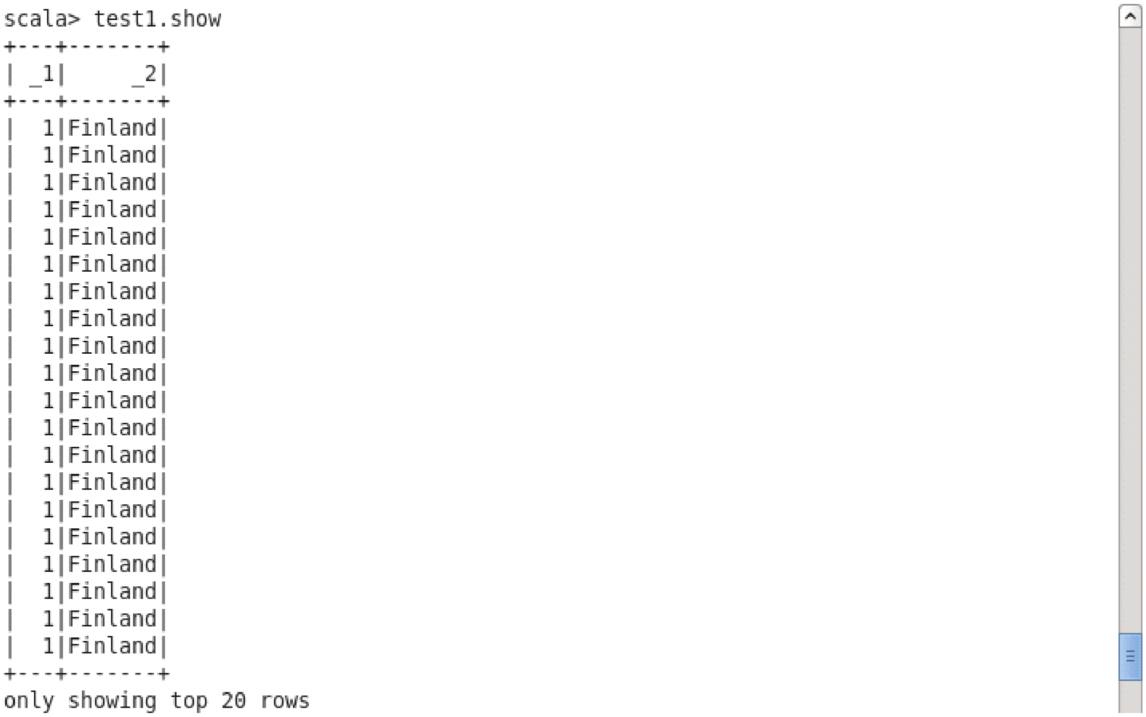
```
scala> val build1 = sqlContext.sql("select h.*, b.country, b.hvacproduct from building b join HVAC1 h on buildid =
buildingid")
build1: org.apache.spark.sql.DataFrame = [Date: string, Time: string ... 7 more fields]

scala> build1.show
+-------+--------+----------+----------+------+---------+----------+----------+-------+-----------+
|   Date|    Time|TargetTemp|ActualTemp|System|SystemAge|BuildingID|tempchange|country|hvacproduct|
+-------+--------+----------+----------+------+---------+----------+----------+-------+-----------+
|6/10/13| 9:00:01|        65|        57|     6|        5|        12|         1|Finland|     FN39TG|
|6/18/13|23:13:19|        66|        75|     1|       13|        12|         1|Finland|     FN39TG|
| 6/2/13|13:43:51|        65|        72|    20|       26|        12|         1|Finland|     FN39TG|
|6/13/13| 0:13:20|        67|        77|     8|       19|        12|         1|Finland|     FN39TG|
|6/16/13| 3:13:20|        67|        55|    11|       16|        12|         1|Finland|     FN39TG|
|6/30/13|17:13:20|        65|        57|    17|        9|        12|         1|Finland|     FN39TG|
| 6/1/13|18:13:20|        68|        65|     7|       21|        12|         0|Finland|     FN39TG|
|6/25/13|18:33:07|        70|        66|    20|       20|        12|         0|Finland|     FN39TG|
|6/17/13|16:00:01|        69|        68|    16|        4|        12|         0|Finland|     FN39TG|
| 6/5/13|16:43:51|        69|        69|    19|       15|        12|         0|Finland|     FN39TG|
|6/23/13|10:13:20|        65|        61|     1|        1|        12|         0|Finland|     FN39TG|
|6/29/13|16:13:20|        67|        80|    12|        8|        12|         1|Finland|     FN39TG|
| 6/4/13|21:13:20|        66|        72|     7|        1|        12|         1|Finland|     FN39TG|
| 6/3/13| 2:00:01|        69|        72|     7|       21|        12|         0|Finland|     FN39TG|
|6/16/13|15:00:01|        67|        77|     4|       22|        12|         1|Finland|     FN39TG|
|6/22/13|21:00:01|        70|        77|    13|       12|        12|         1|Finland|     FN39TG|
|6/26/13| 7:43:51|        65|        62|     6|        6|        12|         0|Finland|     FN39TG|
|6/26/13|13:13:20|        65|        63|    20|        9|        12|         0|Finland|     FN39TG|
|6/30/13|17:13:20|        66|        62|    14|       26|        12|         0|Finland|     FN39TG|
|6/10/13| 3:33:07|        70|        78|     5|        9|        12|         1|Finland|     FN39TG|
+-------+--------+----------+----------+------+---------+----------+----------+-------+-----------+
only showing top 20 rows
```

test.show



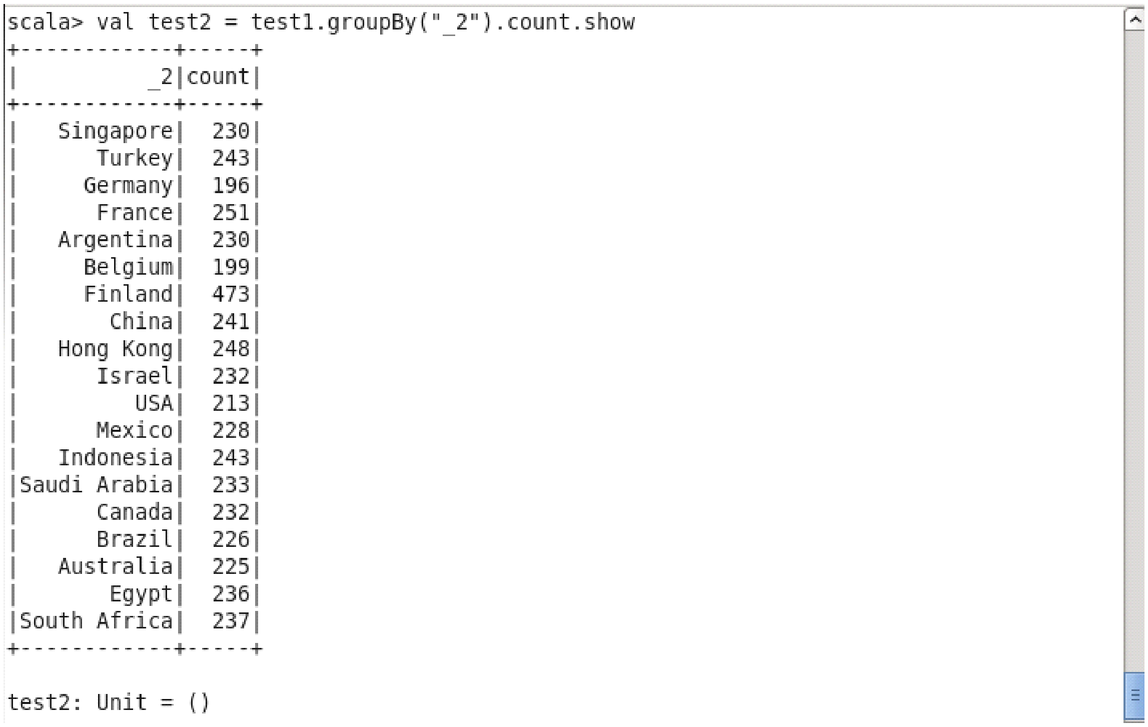
```
scala> test.show
+---+-------+
| _1|     _2|
+---+-------+
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  0|Finland|
|  0|Finland|
|  0|Finland|
|  0|Finland|
|  0|Finland|
|  1|Finland|
|  1|Finland|
|  0|Finland|
|  1|Finland|
|  1|Finland|
|  0|Finland|
|  0|Finland|
|  0|Finland|
|  1|Finland|
+---+-------+
only showing top 20 rows
```

test1.show



```
scala> test1.show
+---+-------+
| _1|     _2|
+---+-------+
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
|  1|Finland|
+---+-------+
only showing top 20 rows
```

test2.show



```
scala> val test2 = test1.groupBy("_2").count.show
+------------+-----+
|          _2|count|
+------------+-----+
|   Singapore|  230|
|      Turkey|  243|
|     Germany|  196|
|      France|  251|
|   Argentina|  230|
|     Belgium|  199|
|     Finland|  473|
|       China|  241|
|   Hong Kong|  248|
|      Israel|  232|
|         USA|  213|
|      Mexico|  228|
|   Indonesia|  243|
|Saudi Arabia|  233|
|      Canada|  232|
|      Brazil|  226|
|   Australia|  225|
|       Egypt|  236|
|South Africa|  237|
+------------+-----+

test2: Unit = ()
```

SO WE CAN SEE FROM THE OUTPUT OF "test2" THAT TEMPERATURE CHANGE IN "FINLAND" IS CHANGING MORE FREQUENTLY FOLLOWED BY "FRANCE" AND "HONG KONG"