# Building-Hvac Dataset Analysis

There are two datasets; building.csv contains the details of the top 20 buildings all over the world and HVAC.csv contains the target temperature and the actual temperature along with the building Id.

HVAC (heating, ventilating/ventilation, and air conditioning) is the technology of indoor and vehicular environmental 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.

## Understanding Data

The data format is comma separated values. There are 8000 observations.

**Schema Description**

**Building.txt** – BuildingID, BuildingMgr, BuildingAge, HVACproduct, Country

**HVAC.txt** – Date, Time, TargetTemp, ActualTemp, System, SystemAge, BuildingID

## Problem Statements

**1) Load both the files (building and hvac) from HDFS and make it an RDD.**

```
scala> val sqlContext = new org.apache.spark.sql.SQLContext(sc)

scala> import sqlContext.implicits._

scala> val data =
sc.textFile("file:///home/cloudera/khasimbabu/Spark_Excercise/attachment_hvac.txt")

scala> val data2 =
sc.textFile("file:///home/cloudera/khasimbabu/Spark_Excercise/attachment_building.txt")
```

**2) Remove the headers from both the files so that only actual data should be present in both RDD.**

```
scala> val header = data.first()

scala> val data1 = data.filter(row => row != header)

scala> val data3 = data2.filter(row => row != header1)
```

**3) Convert both the RDDs into dataframes.**

```
scala> case class
hvac_cls(Date:String,Time:String,TargetTemp:Int,ActualTemp:Int,System:Int,SystemAge:Int,BuildingI
d:Int)

scala> val hvac_data = 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()

scala> case class
build_data(buildid:Int,buildmgr:String,buildAge:Int,hvacproduct:String,country:String)

scala> case class build_cls(buildid:Int,buildmgr:String,buildAge:Int,hvacproduct:String,country:String)
scala> val build_data = data3.map(x => x.split(",")).map(x =>
build_cls(x(0).toInt,x(1),x(2).toInt,x(3),x(4))).toDF()
```

```
scala> hvac_data.registerTempTable("HVAC")
```

```
scala> val hvac1 = sqlContext.sql("select *,IF((TargetTemp-ActualTemp)>5,'1',IF((TargetTemp-
ActualTemp)<-5,'1',0)) as tempchange from HVAC")
```

**4) Register the temp table for both the dataframes.**

```
scala> hvac1.registerTempTable("HVAC1")
```

```
scala> build_data.registerTempTable("BUILDING")
```

**5) Determine which country's temperature is changing more frequently than others.**

```
scala> val build1 = sqlContext.sql("select h.*,b.country,b.hvacproduct from BUILDING b join HVAC1 h
on buildid = BuildingId")
```

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

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

```
scala> val test2 = test1.map(x => (x._2,1)).reduceByKey(_+_).map(item =>
item.swap).sortByKey(false).collect
```