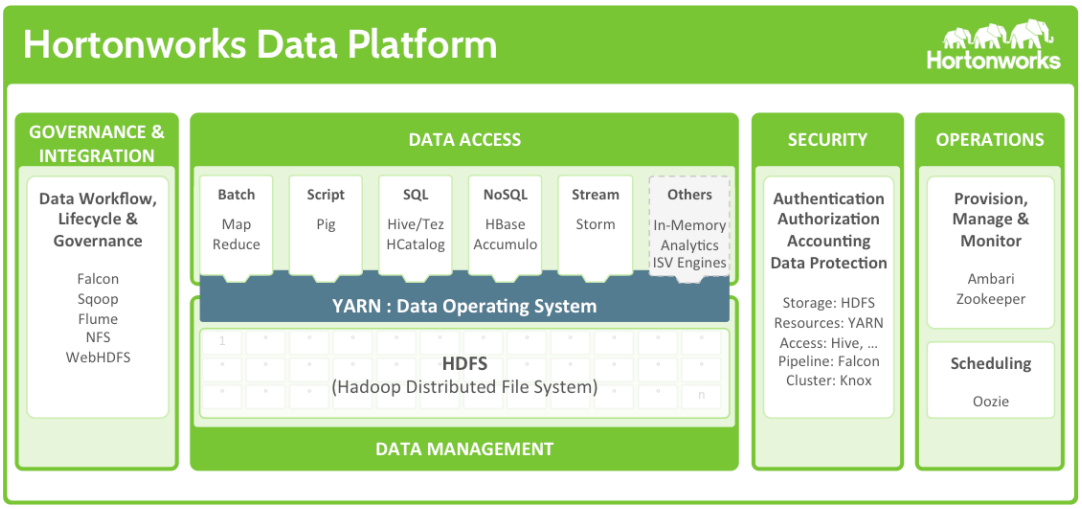


**Figure 1:** Architectural layout of the Hortonworks Sandbox 2.1

```
RAW_DATA = LOAD $Data$ ;
DATA = FOREACH RAW_DATA GENERATE
            FLATTEN(
             ( tuple (LONG,LONG,CHARARRAY,
            CHARARRAY,CHARARRAY, float )
                REGEX_EXTRACT_ALL( line ,          '^(\\d+)
            \\s+(\\d+)\\s+(\\d{4})(\\d{ 2 } ) (\\d{2})\\ s+(\\d+\\.\\d) . * $ ' )
            )
             as   (
             STN:     int ,
            WBAN:     int ,
             YEAR:     int ,
            MONTH:        int , DAY:          int ,
             TEMP:     float
    ) ;
    GroupedData = group DATA by STN . .MONTH;
    avgTemp = FOREACH GroupedData GENERATE AVG( GroupedData.TEMP) ;
    dump avgTemp;
```

**Code example 1**: Apache Pig – Calculating monthly averages

```
def updateMean(value: Double, weeklyMean: Array[Double]): Array[Double] {
val NumEntries: Double = weeklyMean(0)
val Mean: Double = weeklyMeans(1)
// -1 means: no measurement
if (value != -1) (NumEntries*Mean+values)/(NumEntries+1.) else Mean
}
```

**Code example 2:** Scala code for Spark, calculating averages

```
var pMeans: Array[Array[Float]] = Array.tabulate(52, 2)((x,y) => 0f )
var tMeans: Array[Array[Float]] = Array.tabulate(52, 2)((x,y) => 0f )
```

**Code example 3:** Scala code for Spark, creating vectors

```
val tdat = data.map(p => (findWeek(p(2).toInt()), p(3)))
.filter(p => (p(1)>0))
val pdat = data.map(p => (findWeek(p(2).toInt()), p(13)))
.filter(p => (p(1)>=0))

val tMean = tdat.reduceByKey(v => v.foldLeft(0)(_+_) / v.size)
val pMean = pdat.reduceByKey(v => v.foldLeft(0)(_+_) / v.size)
```

**Code example 4:** Scala code for Spark, calculating averages as a reducer