```
import org.apache.spark.sql.functions._
import org.apache.spark.sql.expressions.Window
import org.apache.hadoop.fs.{FileSystem, Path}
import java.nio.file.{Files, Paths, StandardCopyOption}
import java.io.PrintWriter
```

FINISHED

```
import org.apache.spark.sql.functions._
import org.apache.spark.sql.expressions.Window
import org.apache.hadoop.fs.{FileSystem, Path}
import java.nio.file.{Files, Paths, StandardCopyOption}
import java.io.PrintWriter
```

Took 0 sec. Last updated by sc10648_nyu_edu at November 22 2024, 8:32:34 PM.

```
val directoryPath = "/user/sc10648_nyu_edu/stocks/"

val fs = FileSystem.get(spark.sparkContext.hadoopConfiguration)

val csvFiles = fs.listStatus(new Path(directoryPath)).filter(_.getPath.getName.endsWith(".csv
```

FINISHED

```
directoryPath: String = /user/sc10648_nyu_edu/stocks/
fs: org.apache.hadoop.fs.FileSystem = DFS[DFSClient[clientName=DFSClient_NONMAPREDUCE_-1639038
098_1, ugi=sc10648_nyu_edu (auth:SIMPLE)]]
csvFiles: Array[String] = Array(hdfs://nyu-dataproc-m/user/sc10648_nyu_edu/stocks/0001.HK.csv,
hdfs://nyu-dataproc-m/user/sc10648_nyu_edu/stocks/0002.HK.csv, hdfs://nyu-dataproc-m/user/sc10
648_nyu_edu/stocks/0003.HK.csv, hdfs://nyu-dataproc-m/user/sc10648_nyu_edu/stocks/0004.HK.csv,
hdfs://nyu-dataproc-m/user/sc10648_nyu_edu/stocks/0005.HK.csv, hdfs://nyu-dataproc-m/user/sc10
648_nyu_edu/stocks/0006.HK.csv, hdfs://nyu-dataproc-m/user/sc10648_nyu_edu/stocks/0007.HK.csv,
hdfs://nyu-dataproc-m/user/sc10648_nyu_edu/stocks/0008....
```

Took 0 sec. Last updated by sc10648_nyu_edu at November 22 2024, 8:32:36 PM.

```
var allMetricsData = Seq.empty[(String, String, Double, Double, Double, Double, Long, Long)]
```

FINISHED

```
allMetricsData: Seq[(String, String, Double, Double, Double, Double, Long, Long)] = List()
```

Took 1 sec. Last updated by sc10648_nyu_edu at November 22 2024, 8:32:39 PM.

```
def processAndWriteToNewFolder(filePath: String, outputDir: String): Unit = {

    val rawDf = spark.read.option("header", false).option("inferSchema", "true").csv(filePath

    val filteredRDD = rawDf.rdd.zipWithIndex().filter { case (_, idx) => idx >= 3 }.map(_._1)
    val filteredDf = spark.createDataFrame(filteredRDD, rawDf.schema)

    val columnNames = Seq("Date", "AdjClose", "Close", "Open", "High", "Low", "Volume")
    val finalDf = filteredDf.toDF(columnNames: _*)

    val selectedDf = finalDf.select($"Date", $"Close")
    val formattedDf = selectedDf.withColumn("Close", $"Close".cast("double")).withColumn("Dat

    val startDate = formattedDf.select(min($"Date")).collect()(0)(0)
    val minClose = formattedDf.select(min($"Close")).collect()(0)(0).asInstanceOf[Double]
```

FINISHED

```
    val maxClose = formattedDf.select(max($"Close")).collect()(0)(0).asInstanceOf[Double]
    val avgClose = formattedDf.select(avg($"Close")).collect()(0)(0).asInstanceOf[Double]
    val stdDevClose = formattedDf.select(stddev($"Close")).collect()(0)(0).asInstanceOf[Doubl
    val nullCount = formattedDf.filter($"Close".isNull).count()
    val zeroCount = formattedDf.filter($"Close" === 0).count()


    val fileNameWithoutExtension = filePath.split("/").last.replace(".csv", "")

    allMetricsData :+= (fileNameWithoutExtension, startDate.toString, minClose, maxClose, avg

    val deduplicatedDf = formattedDf.dropDuplicates("Date")

    val adjustedDf = deduplicatedDf.withColumn("Close", when($"Close" === 0, lit(null)).other

    val windowSpec = Window.orderBy("Date")
    val filledDf = adjustedDf.withColumn("Close",coalesce(last($"Close", ignoreNulls = true).

    val fileName2 = filePath.split("/").last.replace(".csv", "")
    val outputPath = s"$outputDir/$fileName2"

    filledDf.write.mode("overwrite").option("header", "false").csv(outputPath)

}
```

```
processAndWriteToNewFolder: (filePath: String, outputDir: String)Unit
```

Took 1 sec. Last updated by sc10648_nyu_edu at November 22 2024, 8:32:41 PM.

```
val outputDir = "/user/sc10648_nyu_edu/stocks_processed"

val outputPath = new Path(outputDir)
if (!fs.exists(outputPath)) {
  fs.mkdirs(outputPath)
}
```

FINISHED

```
outputDir: String = /user/sc10648_nyu_edu/stocks_processed
outputPath: org.apache.hadoop.fs.Path = /user/sc10648_nyu_edu/stocks_processed
res35: AnyVal = ()
```

Took 1 sec. Last updated by sc10648_nyu_edu at November 22 2024, 8:32:46 PM.

```
csvFiles.foreach(file => processAndWriteToNewFolder(file, outputDir))
```

SPARK JOB FINISHED

Took 24 sec. Last updated by sc10648_nyu_edu at November 22 2024, 8:33:12 PM.

```
val metricsDf = spark.createDataFrame(allMetricsData).toDF("File", "StartDate", "MinClose", "M

metricsDf.show(false)
```

FINISHED

```
+-------+----------+-------------------+------------------+-------------------+-------------
-----+---------+---------+
|File   |StartDate |MinClose           |MaxClose          |AvgClose           |StdDevClose
|NullCount|ZeroCount|
+-------+----------+-------------------+------------------+-------------------+-------------
```

```
-----+---------+---------+
|0001.HK|2000-01-04|28.946083068847656  |123.074462890625  |68.49653999412385  |20.38463950043
869  |0        |0        |
|0002.HK|2000-01-04|28.950000762939453  |96.94999694824219 |59.30261901290009  |16.48112598479
8634 |0        |0        |
|0003.HK|2000-01-04|2.442112922668457   |16.310039520263672|7.44519474690014   |3.378552668670
7246 |0        |0        |
|0004.HK|2000-01-04|3.452552080154419   |33.349998474121094|14.300606976066675 |7.077720022842
44   |0        |0        |
|0005.HK|2000-01-03|28.200000762939453  |152.8000030517578 |83.16123467243061  |28.50901956372
3243 |0        |0        |
|0006.HK|2000-01-04|22.799999237060547  |82.25             |48.15528824201178  |14.86866499774
```

Took 0 sec. Last updated by sc10648_nyu_edu at November 22 2024, 8:33:32 PM.

READY