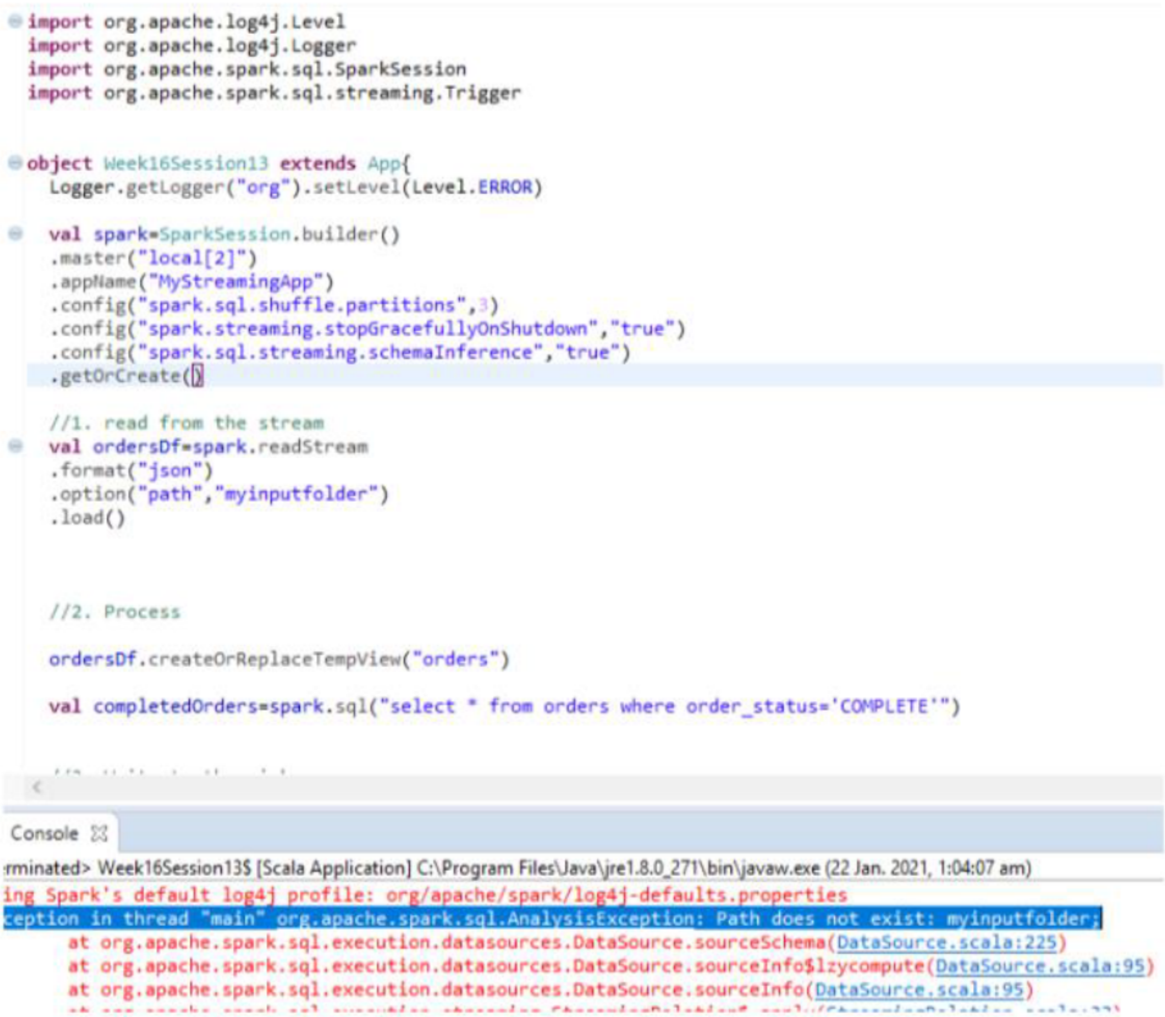
Week16 FAQS

W16:1 Erich are getting path does not exists error

Exception in thread "main" org.apache.spark.sql.AnalysisException: Path does not exist: myinputfolder;



```
import org.apache.log4j.Level
import org.apache.log4j.Logger
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.streaming.Trigger


object Week16Session13 extends App{
  Logger.getLogger("org").setLevel(Level.ERROR)

  val spark=SparkSession.builder()
  .master("local[2]")
  .appName("MyStreamingApp")
  .config("spark.sql.shuffle.partitions",3)
  .config("spark.streaming.stopGracefullyOnShutdown","true")
  .config("spark.sql.streaming.schemaInference","true")
  .getOrCreate()

  //1. read from the stream
  val ordersDf=spark.readStream
  .format("json")
  .option("path","myinputfolder")
  .load()



  //2. Process

  ordersDf.createOrReplaceTempView("orders")

  val completedOrders=spark.sql("select * from orders where order_status='COMPLETE'")
```

```
rminated> Week16Session13$ [Scala Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (22 Jan. 2021, 1:04:07 am)
ing Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
ception in thread "main" org.apache.spark.sql.AnalysisException: Path does not exist: myinputfolder;
	at org.apache.spark.sql.execution.datasources.DataSource.sourceSchema(DataSource.scala:225)
	at org.apache.spark.sql.execution.datasources.DataSource.sourceInfo$lzycompute(DataSource.scala:95)
	at org.apache.spark.sql.execution.datasources.DataSource.sourceInfo(DataSource.scala:95)
```

Ans: Give absolute path for myinputfolder

Week16 FAQS

W16:3 Why do we need to change the checkpoint location after every run?

Ans: There is no need to change checkpoint location always. In fact it must be same for actual jobs. In video, we just changed to avoid confusion
Otherwise it will run based on the previous state store in the checkpoint. we are not restarting the jobs here, so it is better to delete or provide the new checkpoint locations for our job execution

W16:4 Nichlolas practicing the streaming join with another stream using file source but I'm not getting the proper response (joined records)..could someone please help Nichlolas on it where he is doing the wrong, (Nichlolas using impression and click json records as input streams here),..Nichlolas placing these input records in the respective folders for generating the i/p stream,

here is the code snippet:

```
import org.apache.spark.SparkContext
import org.apache.spark.streaming.StreamingContext
import org.apache.spark.streaming.Seconds
import org.apache.log4j.Level
import org.apache.log4j.Logger
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.types.IntegerType
import org.apache.spark.sql.types.StructType
import org.apache.spark.sql.types.StringType
import org.apache.spark.sql.types.StructField
import org.apache.spark.sql.streaming.Trigger
import org.apache.spark.sql.functions._
import org.apache.spark.sql.functions
import org.apache.spark.sql.expressions.Window
import org.apache.spark.sql.types.TimestampType
import org.apache.spark.sql.types.LongType
object MySparkWeek16StructuredStreamingJoinStreamWithStream2 extends App{

 Logger.getLogger("org").setLevel(Level.ERROR)
 //create spark session
 println("Welcome to Structured Streaming")
 val spark = SparkSession.builder()
             .master("local[2]")

.appName("MySparkWeek16StructuredStreamingJoinStreamWithStream2")
             .config("spark.sql.shuffle.partitions",2)
             .config("spark.streaming.stopGracefullyOnShutdown","true")
          //   .config("spark.sql.streaming.schemaInference","true")
             .getOrCreate()
```

Week16 FAQS

```
 val impressionsSchema = StructType(List(StructField("impressionId",
IntegerType),
                                    StructField("impressionTime",
TimestampType),
                                    StructField("CampaignName",
StringType)))

 val impressionsDf = spark.readStream
                     .format("json")

.option("path","C:\\Users\\posiva\\BigData\\16Week_ApacheSparkStreamingPart
-2\\ImpressionsInputStream")
                    // .option("inferSchema",true)
                     .option("maxFilesPerTrigger",1)
                    .schema(impressionsSchema)
                    .load()

 //when we use socket as datasource
 //val valueDf =
impressionsDf.select(from_json(col("value"),impressionsSchema).alias("value
"))

 impressionsDf.printSchema()

 val impressionsDfNew =  impressionsDf.withWatermark("impressionTime","30
minutes")


 val clicksSchema = StructType(List(StructField("clickId", IntegerType),
                                    StructField("clickTime",
TimestampType)))

 val clicksDf = spark.readStream
                     .format("json")

.option("path","C:\\Users\\posiva\\BigData\\16Week_ApacheSparkStreamingPart
-2\\ClicksInputStream")
                    // .option("inferSchema",true)
                     .option("maxFilesPerTrigger",1)
                    .schema(clicksSchema)
                    .load()

  clicksDf.printSchema()
  val clicksDfNew =  clicksDf.withWatermark("clickTime","30 minutes")

 //joining stream with static
 val joinCondition = expr("impressionId == clickId AND clickTime BETWEEN
impressionTime AND impressionTime + interval 15 minute")
 val joinType       = "leftOuter"

 val enrichedDf = impressionsDfNew.join(clicksDfNew,joinCondition,joinType)
                                 // .drop(clicksDfNew.col("clickId"))
```

Week16 FAQS

```
 //write output to the sink
 val resultQuery = enrichedDf.writeStream
                    .format("console")
                    .outputMode("append") //update, append
                    .option("checkpointLocation","checkpointLocation_1")
                  //.option("cleanSource","delete")
                    .option("cleanSource","archive")

.option("sourceArchiveDir","C:\\Users\\posiva\\BigData\\16Week_ApacheSparkStreamingPart-2\\OrdersArchiveStream")
                    .trigger(Trigger.ProcessingTime("10 Second"))
                    .start()
 resultQuery.awaitTermination()


}
```

input click message:
{"clickID": "100001", "ClickTime": "2020-11-01 10:15:00"}
{"clickID": "100001", "ClickTime": "2020-11-01 10:16:00"}
input impression message:
{"impressionID": "100001", "ImpressionTime": "2020-11-01 10:00:00", "CampaignName": "Trendytech India"}
{"impressionID": "100002", "ImpressionTime": "2020-11-01 10:00:00", "CampaignName": "Trendytech India"}
but I'm not receiving the matched record in the output

Ans: You are reading from files, so check if the data in the files is correct, because some time if you **manually create files** then it won't be able to get data.

W16:5 for the left-outer join example between impressions & clicks streams, watermark boundary is calculate as :

impressions = max(event time) - watermark - (max interval time b/w left & right streams)
clicks = max(event time) - watermark
Why is the max interval time not considered for clicks stream watermark boundary ?
Is it because the joinExpr has constraint on ClicksTime depending on the ImpressionTime ??
Is my understanding right, or am I missing something ?

Ans: Yes, the basic understanding in the left outer join example is, you want to reduce more and more records from the right table so that you have to lookup less records otherwise it could result in out-of-memory exception. That's why the joinExpr has a constraint on the right table (clicks) to be within the left table(impression) time range. I hope it's clear

Week16 FAQS

W16:6 Why we can not use output mode = update, while doing inner join on both streaming dataframes. (In week16 & Structured Streaming - Session13) When I change output mode from append to update I'm getting the following error.

Exception in thread "main" org.apache.spark.sql.AnalysisException: Inner join between two streaming DataFrames/Datasets is not supported in Update output mode, only in Append output mode;

Ans: Append is used when there is no aggregation and update is used with aggregation.
In join , new records need to be joined hence append
If you are doing aggregation the Updates also needs to be considered

W16:7 Please help with the error:

```
val spark = SparkSession.builder()
   .appName("Structured Streaming Test")
   .config("spark.master", "local[*]")
   .config("spark.sql.shuffle.partitions", 2)
   .config("spark.sql.streaming.schemaInference", "true")
   .getOrCreate()
 //1. read from file source
 val orderDf = spark.readStream
   .format("json")
   .option("path", "src/main/data_week16/inputdata/JsonMessages.txt")
   .load()
 //2. process
 orderDf.createOrReplaceTempView("order")
 val completedOrders = spark.sql("select * from order where order_status = 'COMPLETE'")
//  val completedOrders = orderDf.filter(col("order_status") === "COMPLETE")
 //3. write to the sink
 val ordersQuery = completedOrders.writeStream
   .format("json")
   .outputMode("append")
   .option("path", "src/main/data_week16/output")
   .option("checkpointLocation", "checkpoint-loc2")
   .trigger(Trigger.ProcessingTime("30 seconds"))
   .start()
 ordersQuery.awaitTermination()
```

`Ans:` Path in Structured Streaming has to be a directory not a file.

Week16 FAQS

EX:-

.option("path", "G:/TRENDY~TECH/WEEK-16/JsonMessages.txt") //WRONG WAY

option("path", "G:/TRENDY~TECH/WEEK-16/MyMessages/")   //RIGHT WAY✅

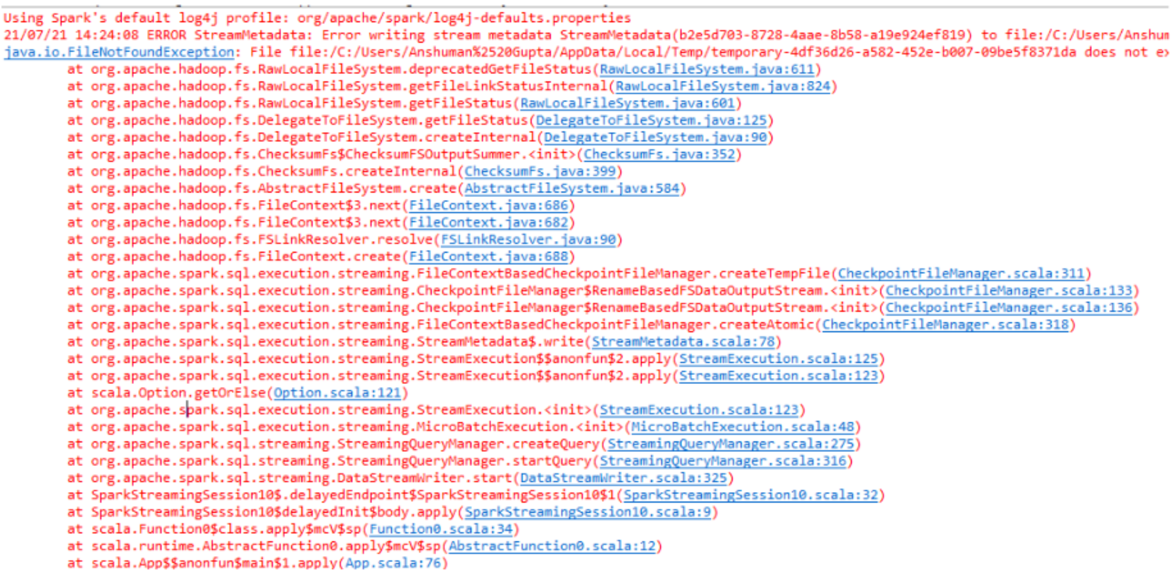
& for WRITE :-

.option("path", "G:/TRENDY~TECH/WEEK-16/OutoutOfMyMessages/")

W16:8 Can someone please let me know how to do hive,Hbase and spark integrations while coding any spark program

Ans: you need to add Hive and hbase jar, as we added for spark. and then we can write code(explore on internet for exact steps). Hive and spark integraton we have done in previous week.

W16:9 I am not able to resolve below error please help in this => [21/07/21 14:24:08 ERROR StreamMetadata: Error writing stream metadata StreamMetadata(b2e5d703-8728-4aae-8b58-a19e924ef819) to file:/C:/Users/Anshuman%2520Gupta/AppData/Local/Temp/temporary-4df36d26-a582-452e-b007-09be5f8371da/metadata java.io.FileNotFoundException: File file:/C:/Users/Anshuman%2520Gupta/AppData/Local/Temp/temporary-4df36d26-a582-452e-b007-09be5f8371da does not exist]

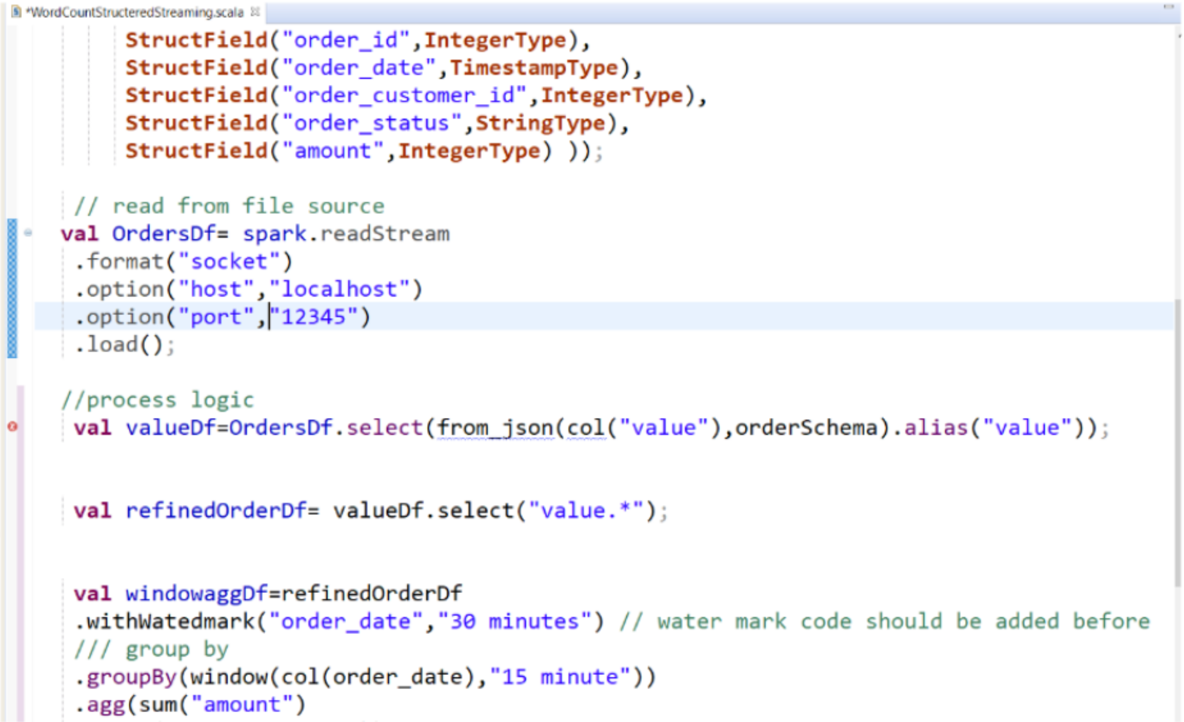


```
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
21/07/21 14:24:08 ERROR StreamMetadata: Error writing stream metadata StreamMetadata(b2e5d703-8728-4aae-8b58-a19e924ef819) to file:/C:/Users/Anshum
java.io.FileNotFoundException: File file:/C:/Users/Anshuman%2520Gupta/AppData/Local/Temp/temporary-4df36d26-a582-452e-b007-09be5f8371da does not e
	at org.apache.hadoop.fs.RawLocalFileSystem.deprecatedGetFileStatus(RawLocalFileSystem.java:611)
	at org.apache.hadoop.fs.RawLocalFileSystem.getFileLinkStatusInternal(RawLocalFileSystem.java:824)
	at org.apache.hadoop.fs.RawLocalFileSystem.getFileStatus(RawLocalFileSystem.java:601)
	at org.apache.hadoop.fs.DelegateToFileSystem.getFileStatus(DelegateToFileSystem.java:125)
	at org.apache.hadoop.fs.DelegateToFileSystem.createInternal(DelegateToFileSystem.java:90)
	at org.apache.hadoop.fs.ChecksumFs$ChecksumFSOutputSummer.<init>(ChecksumFs.java:352)
	at org.apache.hadoop.fs.ChecksumFs.createInternal(ChecksumFs.java:399)
	at org.apache.hadoop.fs.AbstractFileSystem.create(AbstractFileSystem.java:584)
	at org.apache.hadoop.fs.FileContext$3.next(FileContext.java:686)
	at org.apache.hadoop.fs.FileContext$3.next(FileContext.java:682)
	at org.apache.hadoop.fs.FSLinkResolver.resolve(FSLinkResolver.java:90)
	at org.apache.hadoop.fs.FileContext.create(FileContext.java:688)
	at org.apache.spark.sql.execution.streaming.FileContextBasedCheckpointFileManager.createTempFile(CheckpointFileManager.scala:311)
	at org.apache.spark.sql.execution.streaming.CheckpointFileManager$RenameBasedFSDataOutputStream.<init>(CheckpointFileManager.scala:133)
	at org.apache.spark.sql.execution.streaming.CheckpointFileManager$RenameBasedFSDataOutputStream.<init>(CheckpointFileManager.scala:136)
	at org.apache.spark.sql.execution.streaming.FileContextBasedCheckpointFileManager.createAtomic(CheckpointFileManager.scala:318)
	at org.apache.spark.sql.execution.streaming.StreamMetadata$.write(StreamMetadata.scala:78)
	at org.apache.spark.sql.execution.streaming.StreamExecution$$anonfun$2.apply(StreamExecution.scala:125)
	at org.apache.spark.sql.execution.streaming.StreamExecution$$anonfun$2.apply(StreamExecution.scala:123)
	at scala.Option.getOrElse(Option.scala:121)
	at org.apache.spark.sql.execution.streaming.StreamExecution.<init>(StreamExecution.scala:123)
	at org.apache.spark.sql.execution.streaming.MicroBatchExecution.<init>(MicroBatchExecution.scala:48)
	at org.apache.spark.sql.streaming.StreamingQueryManager.createQuery(StreamingQueryManager.scala:275)
	at org.apache.spark.sql.streaming.StreamingQueryManager.startQuery(StreamingQueryManager.scala:316)
	at org.apache.spark.sql.streaming.DataStreamWriter.start(DataStreamWriter.scala:325)
	at SparkStreamingSession10$.delayedEndpoint$SparkStreamingSession10$1(SparkStreamingSession10.scala:32)
	at SparkStreamingSession10$delayedInit$body.apply(SparkStreamingSession10.scala:9)
	at scala.Function0$class.apply$mcV$sp(Function0.scala:34)
	at scala.runtime.AbstractFunction0.apply$mcV$sp(AbstractFunction0.scala:12)
	at scala.App$$anonfun$main$1.apply(App.scala:76)
```

Ans : Make sure there is no space in your path and also give a checkpoint, if given already then delete it and run again.

Week16 FAQS

W16:10 Sam is getting below error



```
            StructField("order_id",IntegerType),
            StructField("order_date",TimestampType),
            StructField("order_customer_id",IntegerType),
            StructField("order_status",StringType),
            StructField("amount",IntegerType) ));

  // read from file source
  val OrdersDf= spark.readStream
   .format("socket")
   .option("host","localhost")
   .option("port","12345")
   .load();

  //process logic
   val valueDf=OrdersDf.select(from_json(col("value"),orderSchema).alias("value"));


   val refinedOrderDf= valueDf.select("value.*");


   val windowaggDf=refinedOrderDf
   .withWatedmark("order_date","30 minutes") // water mark code should be added before
   /// group by
   .groupBy(window(col(order_date),"15 minute"))
   .agg(sum("amount")
```

Ans : Add import → import org.apache.spark.sql.functions._