Timezone Conversion with Daylight Saving in Spark

This document describes how to implement timezone conversion from UTC to local time zones with and without daylight saving using PySpark. The approach includes creating a table of European time zones and determining whether daylight saving is active based on the day of the year.

# PySpark Code for Timezone Conversion

from pyspark.sql import SparkSession  
from pyspark.sql.functions import col, to\_utc\_timestamp, from\_utc\_timestamp, lit, dayofyear, when  
  
# Initialize the Spark session  
spark = SparkSession.builder.appName("Timezone Conversion").getOrCreate()  
  
# Schema  
schema = """   
 SalesOrderID INT,  
 SalesOrderDetailID INT,  
 CarrierTrackingNumber STRING,  
 OrderQty INT,  
 ProductID INT,  
 SpecialOfferID INT,  
 UnitPrice DOUBLE,  
 UnitPriceDiscount DOUBLE,  
 LineTotal Double,  
 rowguid STRING,  
 ModifiedDate STRING  
"""  
  
# Load the CSV data  
df = spark.read.format("csv").option("header", "true").schema(schema).load("/FileStore/tables/Sales\_SalesOrderDetail.csv")  
  
# Convert 'ModifiedDate' to timestamp  
df = df.withColumn("ModifiedDate", col("ModifiedDate").cast("timestamp"))  
  
# Static list of European time zones with daylight saving details  
time\_zone\_table = [  
 {"zone\_id": "Europe/London", "local\_time\_with\_dst": "BST", "local\_time\_without\_dst": "GMT"},  
 {"zone\_id": "Europe/Berlin", "local\_time\_with\_dst": "CEST", "local\_time\_without\_dst": "CET"},  
 {"zone\_id": "Europe/Paris", "local\_time\_with\_dst": "CEST", "local\_time\_without\_dst": "CET"},  
 {"zone\_id": "Europe/Madrid", "local\_time\_with\_dst": "CEST", "local\_time\_without\_dst": "CET"},  
 {"zone\_id": "Europe/Rome", "local\_time\_with\_dst": "CEST", "local\_time\_without\_dst": "CET"},  
 # Add other relevant zones here  
]  
  
# Create a DataFrame for the time zones  
zone\_df = spark.createDataFrame(time\_zone\_table)  
  
# Define the conversion logic and calculate local time  
df\_with\_timezone = df.withColumn("UTC", to\_utc\_timestamp(col("ModifiedDate"), "UTC")) .withColumn("Local\_Time\_Without\_DST", from\_utc\_timestamp(col("ModifiedDate"), "Europe/London")) .withColumn("Local\_Time\_With\_DST", from\_utc\_timestamp(col("ModifiedDate"), "Europe/Berlin"))  
  
# Determine if daylight saving is on or off (assuming DST between day 60 and day 300 of the year for Europe)  
df\_with\_timezone = df\_with\_timezone.withColumn("Day\_Of\_Year", dayofyear(col("ModifiedDate"))) .withColumn("Is\_DST\_On", when((col("Day\_Of\_Year") >= 60) & (col("Day\_Of\_Year") <= 300), lit("Yes")).otherwise(lit("No")))  
  
# Show the final DataFrame  
df\_with\_timezone.show()