**Task 4**

**Multiple ways of creating dataframe using sqlContext**

**Notebook:** <https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/1092176685531650/3530701261005487/6776489139542437/latest.html>

txF = sc.textFile("<File Path>")

tx1 = txF.map(lambda x: x.split(","))

1. Ambiguous column names

sqlContext.createDataFrame(tx1)

1. Specified column names - Approach 1

sqlContext.createDataFrame(tx1, ['account\_id', 'balance'])

1. Specified column names - Approach 2

from pyspark.sql import Row

account = Row(‘account\_id’, ‘balance’)

tx2 = tx1.map(lambda x: account(\*x)) #: Pointer. Only pointer is allowed here

sqlContext.createDataFrame(tx2)

1. Applying schema more sophisticated way

from pyspark.sql.types import \*

tx2 = tx1.map(lambda x: (int(x[0]),int(x[1])))

schema = StructType([StructField("account\_id", IntegerType(), True), StructField("balance", IntegerType(), True)])

sqlContext.createDataFrame(tx2, schema)

**NOTE:** try creating df directly from tx1 and see the result

1. We can also create dataframe from native python pandas dataframe but that has been kept as an exercise for learners