import random

from pyspark.sql.functions import udf, col, when, row\_number

from pyspark.sql.window import Window

from pyspark.sql.types import StringType

# Function to introduce typos

def introduce\_typos(word, change\_type):

"""Randomly introduce typos or completely change the word."""

if not word or len(word) < 2:

return word

word = list(word)

if change\_type == "same":

return "".join(word) # No change

elif change\_type == "similar":

idx = random.randint(0, len(word) - 2) # Pick a random index to swap

word[idx], word[idx + 1] = word[idx + 1], word[idx] # Swap adjacent letters

return "".join(word)

elif change\_type == "dissimilar":

return "RANDOM\_" + str(random.randint(100, 999)) # Completely change word

return "".join(word)

# Register as PySpark UDF

introduce\_typos\_udf = udf(introduce\_typos, StringType())

# Assign a row number to shuffle records equally

window\_spec = Window.orderBy("Source") # Shuffle records

df = df.withColumn("row\_number", row\_number().over(window\_spec))

# Introduce typos based on random selection

df = df.withColumn(

"Destination",

when((col("row\_number") % 3 == 0), introduce\_typos\_udf(col("Destination"), "same"))

.when((col("row\_number") % 3 == 1), introduce\_typos\_udf(col("Destination"), "similar"))

.otherwise(introduce\_typos\_udf(col("Destination"), "dissimilar"))

)

# Assign classification labels based on changes

df = df.withColumn(

"classification\_final",

when(col("Destination") == col("Source"), "Same")

.when(col("Destination").contains("RANDOM\_"), "Dissimilar")

.otherwise("Similar")

)

# Drop temporary row\_number column

df = df.drop("row\_number")

# Show results

df.select("Source", "Destination", "classification\_final").show(truncate=False)