### \*\*Exercise: PySpark Data Transformations on Movie Data\*\*

from pyspark.sql import SparkSession

from pyspark.sql.functions import col, year, to\_date, avg

# Initialize Spark Session

spark = SparkSession.builder.appName("MovieDataTransformations").getOrCreate()

### \*\*Tasks\*\*:

#1. \*\*Load the Dataset\*\*:

file\_path = "/content/sample\_data/movies.csv.txt"

df = spark.read.csv(file\_path, header=True, inferSchema=True)

# Show the DataFrame

print("Initial DataFrame:")

df.show()

# \*2. Filter Movies by Genre\*

sci\_fi\_movies = df.filter(df.genre == "Sci-Fi")

print("Movies in the 'Sci-Fi' genre:")

sci\_fi\_movies.show()

# \*3. Top-Rated Movies\*

top\_rated\_movies = df.orderBy(col("rating").desc()).limit(3)

print("Top 3 highest-rated movies:")

top\_rated\_movies.show()

# \*4. Movies Released After 2010\*

df = df.withColumn("date", to\_date(col("date"), "yyyy-MM-dd"))

movies\_after\_2010 = df.filter(year(col("date")) > 2010)

print("Movies released after 2010:")

movies\_after\_2010.show()

# \*5. Calculate Average Box Office Collection by Genre\*

avg\_box\_office\_by\_genre = df.groupBy("genre").agg(avg("box\_office").alias("average\_box\_office"))

print("Average box office collection by genre:")

avg\_box\_office\_by\_genre.show()

# \*6. Add a New Column for Box Office in Billions\*

df = df.withColumn("box\_office\_in\_billions", col("box\_office") / 1e9)

print("DataFrame with box office in billions:")

df.show()

# \*7. Sort Movies by Box Office Collection\*

sorted\_movies\_by\_box\_office = df.orderBy(col("box\_office").desc())

print("Movies sorted by box office collection (descending):")

sorted\_movies\_by\_box\_office.show()

# \*8. Count the Number of Movies per Genre\*

count\_movies\_per\_genre = df.groupBy("genre").count()

print("Number of movies per genre:")

count\_movies\_per\_genre.show()

# Stop Spark Session

spark.stop()