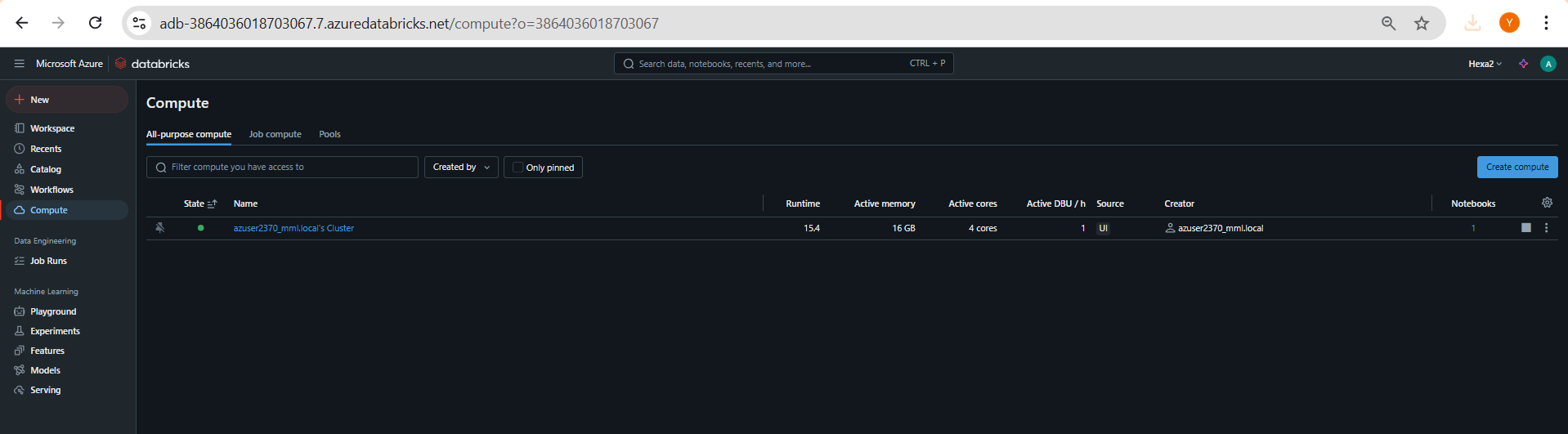
Azure Databricks Coding Challenge (10-12-2024)

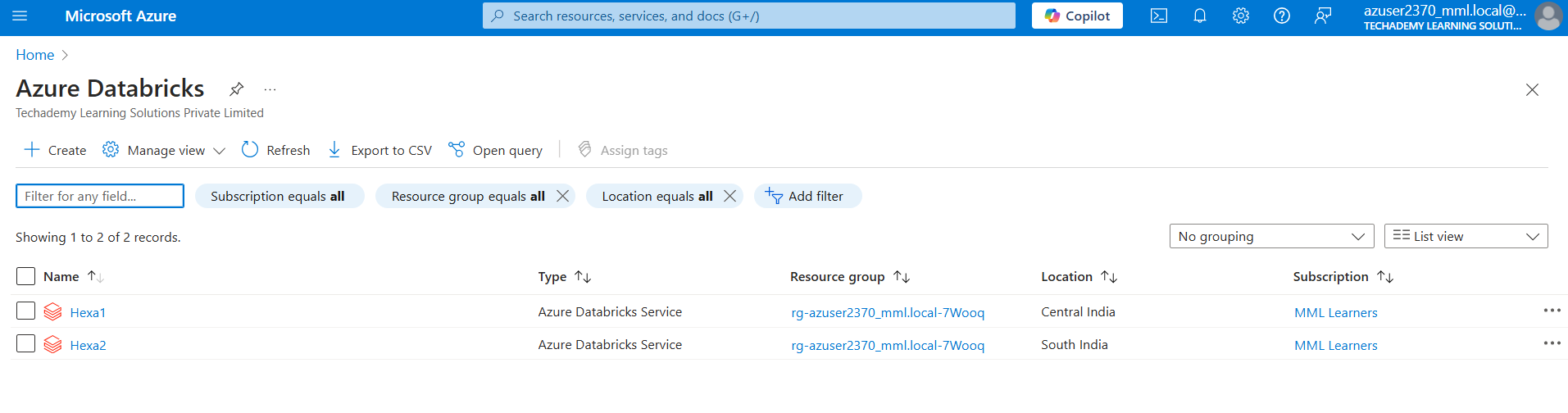
Done by: S.YAZHINI

1. Create a cluster & Attach the notebook to the cluster and run all commands in the notebook & creates a DataFrame from a Databricks dataset& Create a Visualizations in Databricks notebooks

**Cluster created:**



**Databricks Workspace Created:**

****

**Datasets added to the Databricks Catalog (inside hive metastore):**

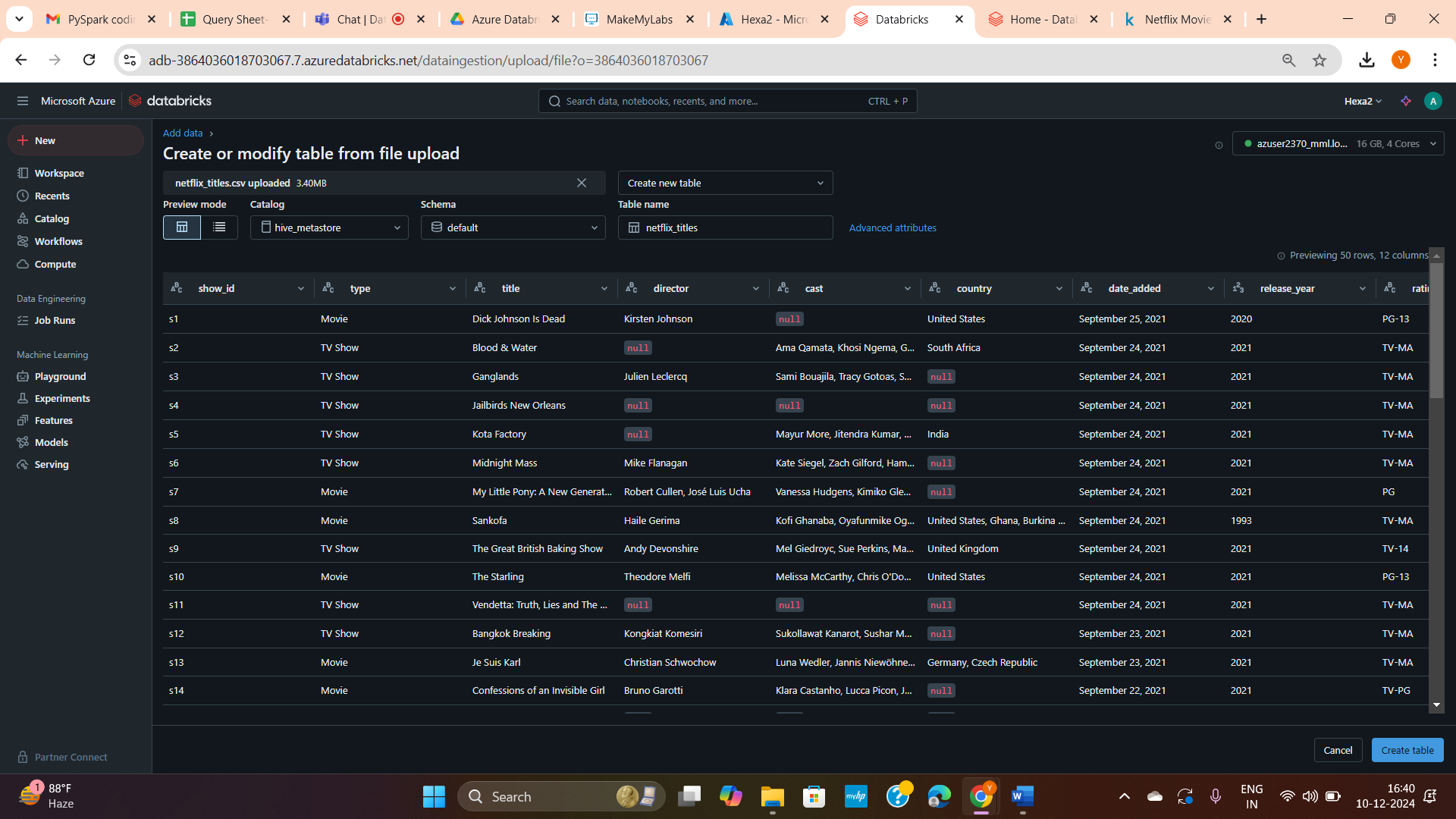
Dataset used (from Kaggle):

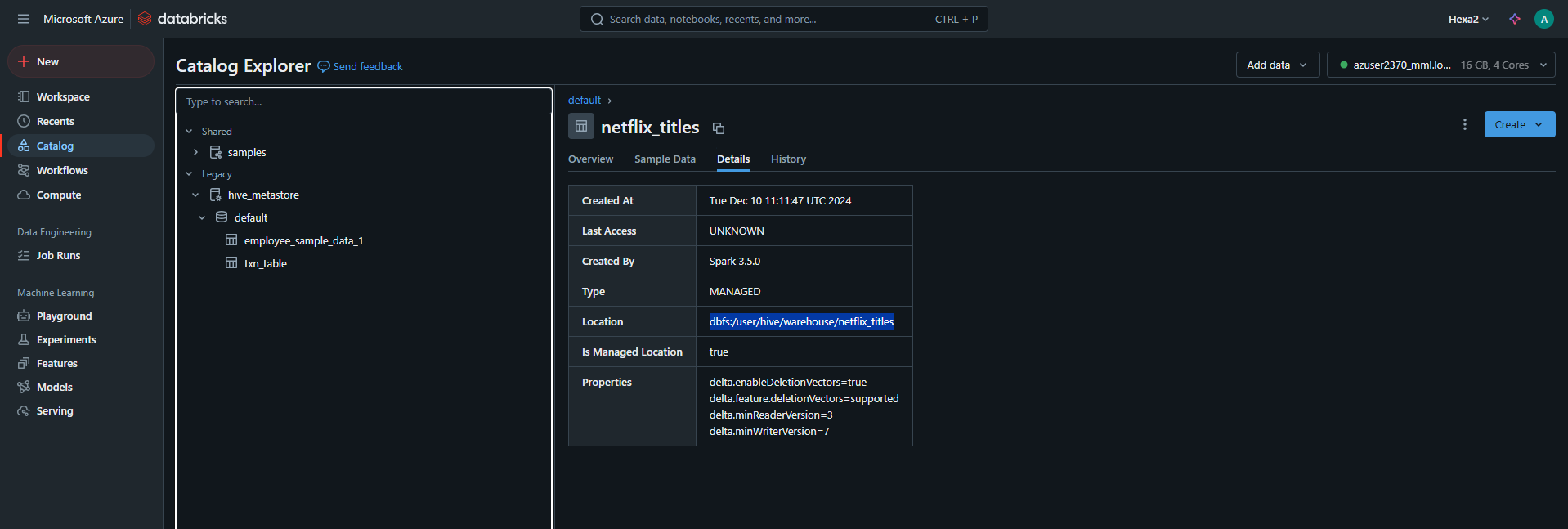
<https://www.kaggle.com/datasets/shivamb/netflix-shows>

You can also download the above dataset from my drive link given below:

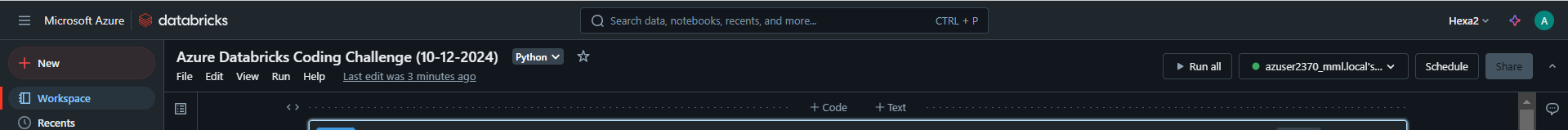
<https://drive.google.com/file/d/1ZmKmbjTEWCF89IiwGBsF3xRWKR_sA58l/view?usp=drive_link>

Here are the uploaded dataset steps screenshots:





**Notebook attached to the cluster:**

****

**PySpark Code/commands:**

**Initialize Spark Session:**

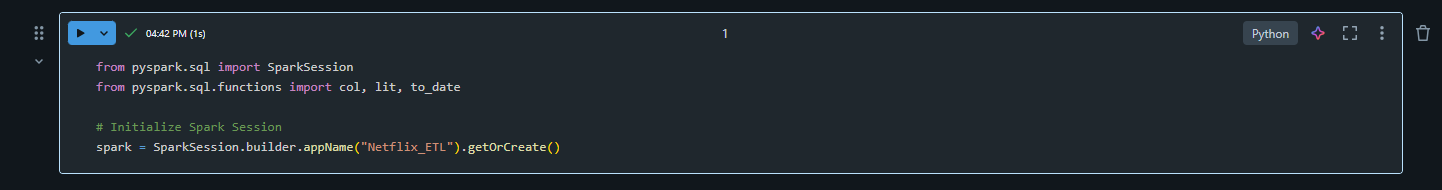
from pyspark.sql import SparkSession

from pyspark.sql.functions import col, lit, to\_date

# Initialize Spark Session

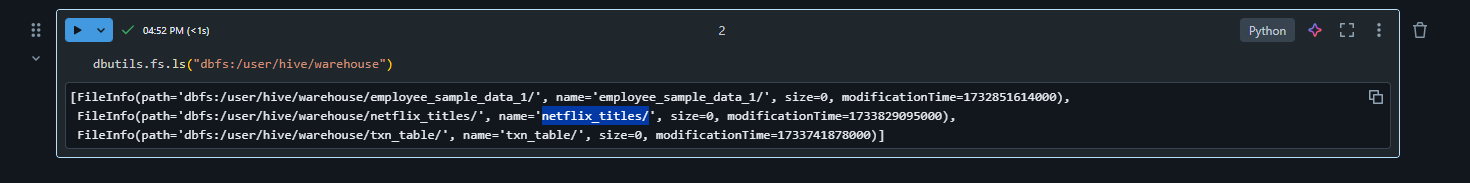
spark = SparkSession.builder.appName("Netflix\_ETL").getOrCreate()

where Netflix\_ETL is a general app name given by me.



**Command to check whether our table is available in the desired directory:**

Syntax: dbutils.fs.ls("dbfs:/directory\_path")

****

**Ingestion: Load data into DataFrame**

# Load data into DataFrame

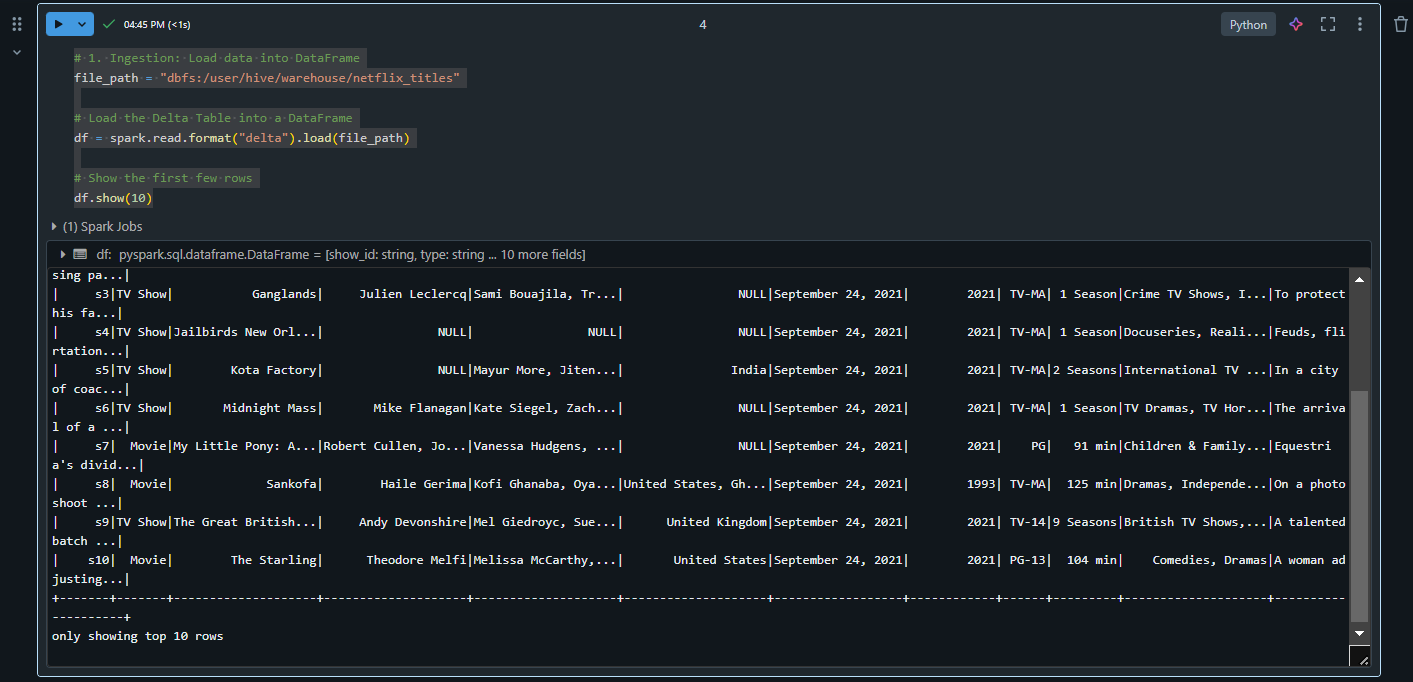
file\_path = "dbfs:/user/hive/warehouse/netflix\_titles"

# Load the Delta Table into a DataFrame

df = spark.read.format("delta").load(file\_path)

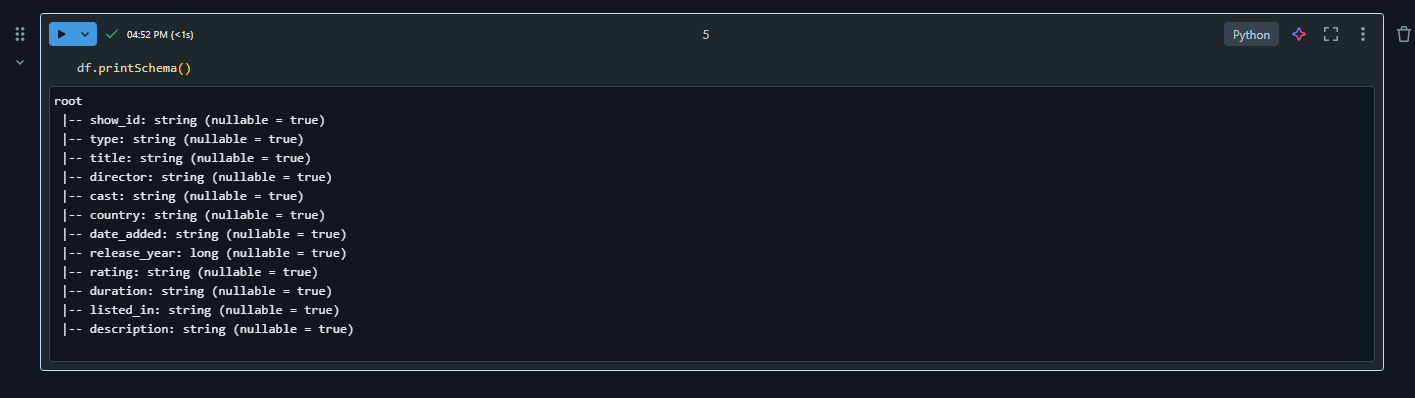
# Show the first few rows

df.show(10)



**For displaying schema of our delta table:**

Syntax: df.printSchema()



**Transformation: Data Cleaning and Filtering codes:**

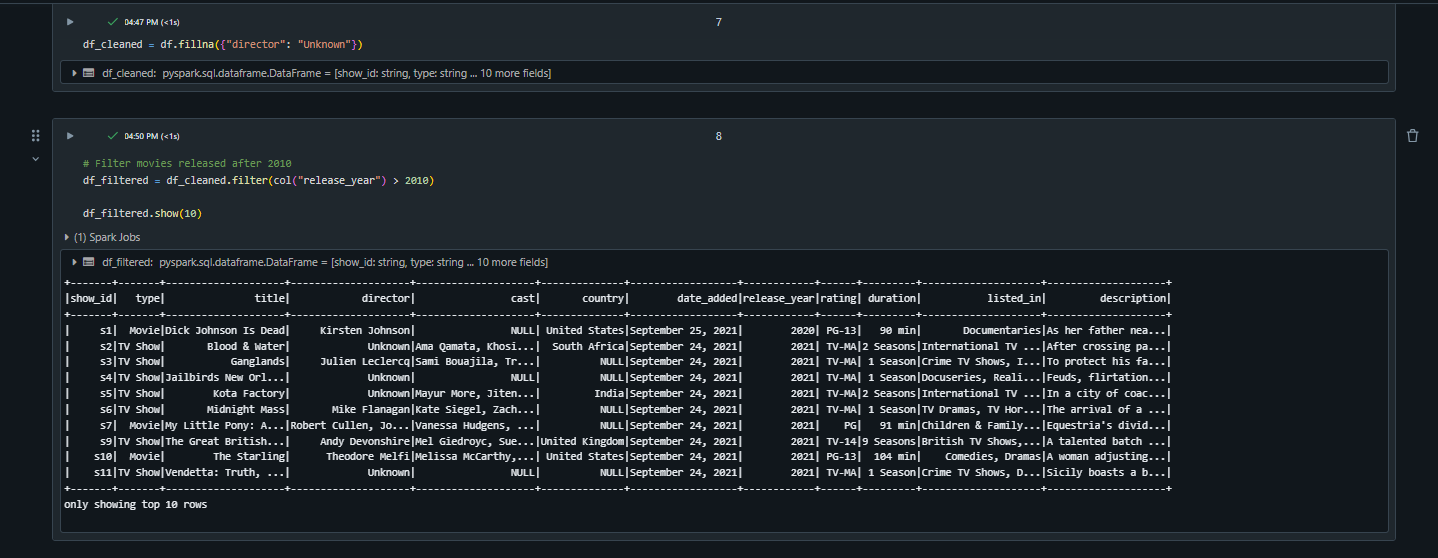
**Replace nulls in 'director' column with 'Unknown'**

df\_cleaned = df.fillna({"director": "Unknown"})

**Filter movies released after 2010**

df\_filtered = df\_cleaned.filter(col("release\_year") > 2010)

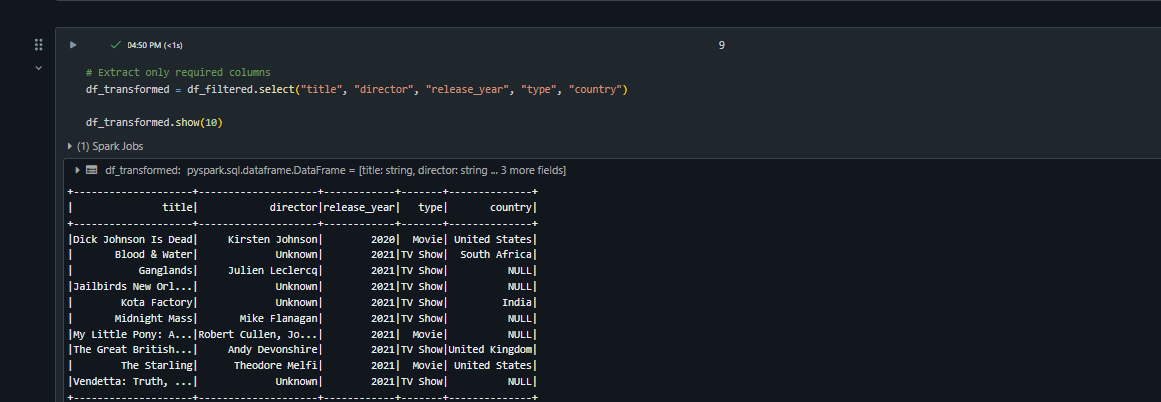
df\_filtered.show(10)



**# Extract only required columns**

df\_transformed = df\_filtered.select("title", "director", "release\_year", "type", "country")

df\_transformed.show(10)



**Getting data with a specific condition:**

from pyspark.sql.functions import when, col

# Add a column to categorize content as "New" or "Old"

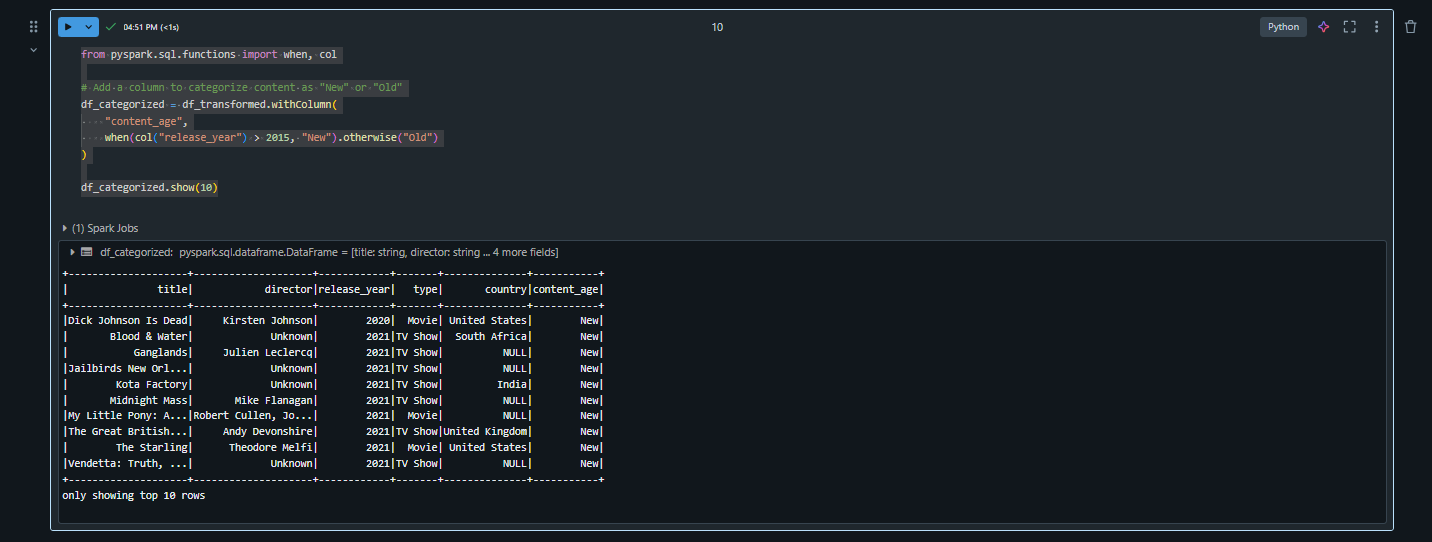
df\_categorized = df\_transformed.withColumn(

    "content\_age",

    when(col("release\_year") > 2015, "New").otherwise("Old")

)

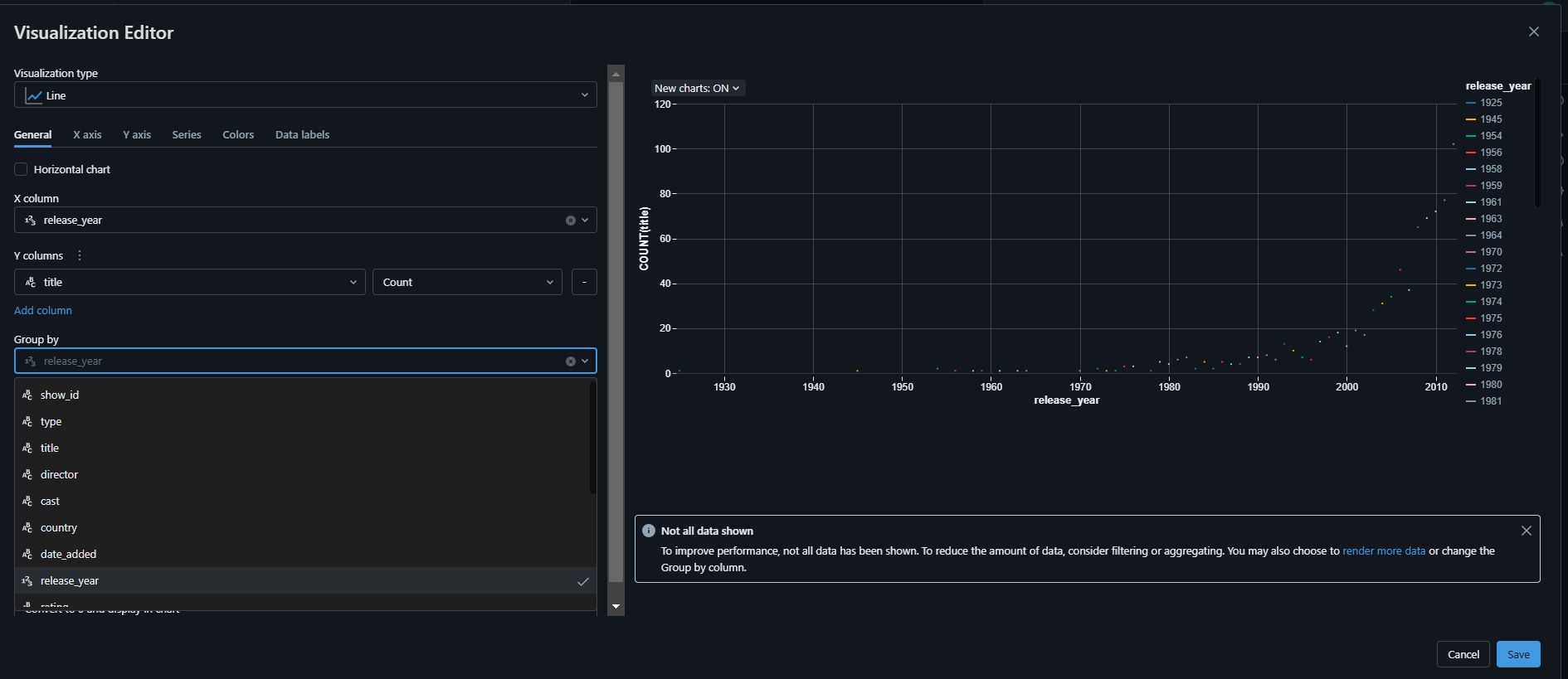
df\_categorized.show(10)



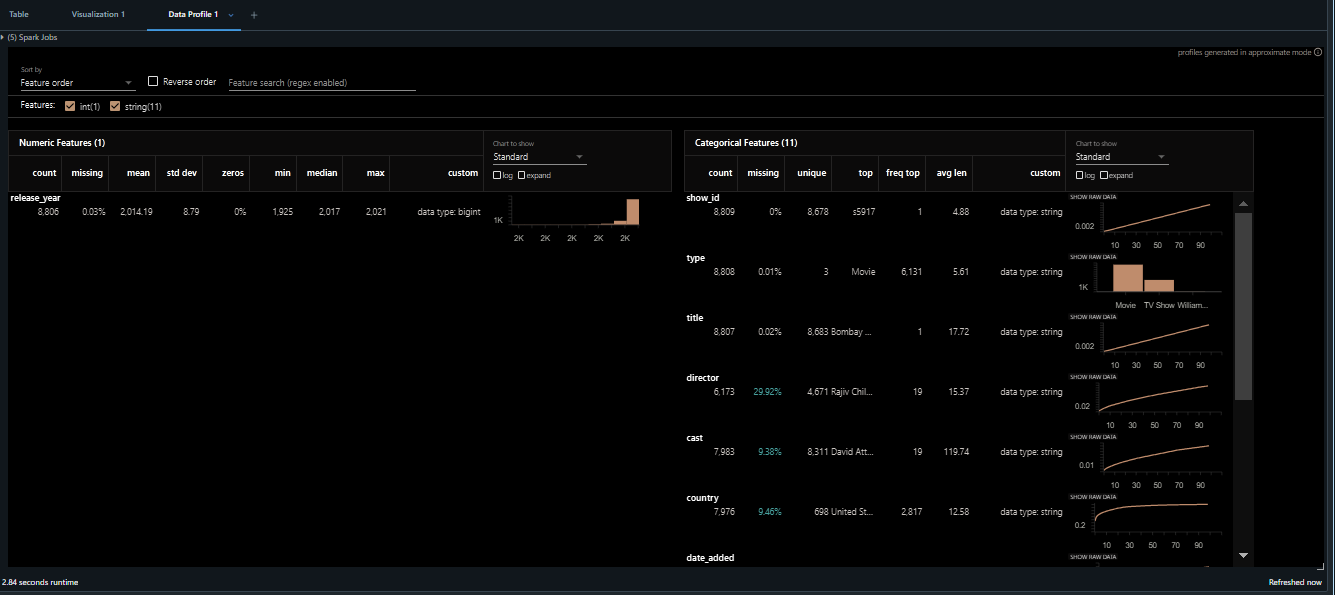
**Visualization:**

X-axis (sample): release\_year

Y-axis (sample): count(title)



**Data Profile:**

****

