from pyspark.sql import SparkSession

from pyspark.sql.functions import col

**# Initialize SparkSession**

spark = SparkSession.builder \

.appName("Product Sales Analysis") \

.getOrCreate()

**# Sample data for products**

products = [

(1, "Laptop", "Electronics", 50000),

(2, "Smartphone", "Electronics", 30000),

(3, "Table", "Furniture", 15000),

(4, "Chair", "Furniture", 5000),

(5, "Headphones", "Electronics", 2000),

]

**# Sample data for sales transactions**

sales = [

(1, 1, 2),

(2, 2, 1),

(3, 3, 3),

(4, 1, 1),

(5, 4, 5),

(6, 2, 2),

(7, 5, 10),

(8, 3, 1),

]

**# Define schema for DataFrames**

product\_columns = ["ProductID", "ProductName", "Category", "Price"]

sales\_columns = ["SaleID", "ProductID", "Quantity"]

**# Create DataFrames**

product\_df = spark.createDataFrame(products, schema=product\_columns)

sales\_df = spark.createDataFrame(sales, schema=sales\_columns)

**# Show the DataFrames**

print("Products DataFrame:")

product\_df.show()

print("Sales DataFrame:")

sales\_df.show()

**# Join the DataFrames:**

**# - Join the `product\_df` and `sales\_df` DataFrames on `ProductID` to create a**

**# combined DataFrame with product and sales data**.

joined\_df = product\_df.join(sales\_df, on="ProductID", how="inner")

**# Show the joined DataFrame**

print("Joined DataFrame:")

joined\_df.show()

**# 2. Calculate Total Sales Value:**

**# - For each product, calculate the total sales value by multiplying the price**

**# by the quantity sold.**

total\_sales\_df = joined\_df.withColumn("TotalSalesValue", col("Price") \* col("Quantity"))

# Show the DataFrame with total sales value

print("DataFrame with Total Sales Value:")

total\_sales\_df.show()

**# 3. Find the Total Sales for Each Product Category:**

**# - Group the data by the `Category` column and calculate the total sales value**

**# for each product category.**

total\_sales\_by\_category = total\_sales\_df.groupBy("Category").agg({"TotalSalesValue": "sum"})

**# Show the DataFrame with total sales by category**

print("DataFrame with Total Sales by Category:")

total\_sales\_by\_category.show()

**# 4. Identify the Top-Selling Product:**

**# - Find the product that generated the highest total sales value.**

top\_selling\_product = total\_sales\_df.groupBy("ProductName").agg({"TotalSalesValue": "sum"}).orderBy("sum(TotalSalesValue)", ascending=False).limit(1).withColumnRenamed("sum(TotalSalesValue)", "TotalSalesValue")

# Show the top-selling product

print("Top-Selling Product:")

print(top\_selling\_product.show())

**# 5. \*\*Sort the Products by Total Sales Value:\*\***

**# - Sort the products by total sales value in descending order.**

sorted\_products\_df = total\_sales\_df.orderBy("TotalSalesValue", ascending=False)

# Show the DataFrame with products sorted by total sales value

print("DataFrame with Products Sorted by Total Sales Value:")

sorted\_products\_df.show()

**# 6. \*\*Count the Number of Sales for Each Product:\*\***

**# - Count the number of sales transactions for each product.**

sales\_count\_df = total\_sales\_df.groupBy("ProductName").agg({"SaleID": "count"}).withColumnRenamed("count(SaleID)", "SalesCount")

# Show the DataFrame with sales count for each product

print("DataFrame with Sales Count for Each Product:")

sales\_count\_df.show()

**# 7. \*\*Filter the Products with Total Sales Value Greater Than ₹50,000:\*\***

**# - Filter out the products that have a total sales value greater than ₹50,000.**

filtered\_products\_df = total\_sales\_by\_category.filter(col("sum(TotalSalesValue)") > 50000)

# Show the DataFrame with filtered products

print("DataFrame with Filtered Products:")

filtered\_products\_df.show()