**Ecommerce Data Set:**

from pyspark.sql import SparkSession

from pyspark.sql.functions import col

from pyspark.sql import functions as F

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

ecommerce\_df = spark.read.format("csv").option("header","true").option("inferSchema","true").load("/content/sample\_data/ecommerce\_data.csv")

**# 1. Calculate the Total Revenue per Category**

total\_revenue\_df = ecommerce\_df.withColumn("revenue", (col("price") \* col("quantity")) \* (1-col("discount\_percentage")/100)).groupBy("category").agg(F.sum("revenue").alias("total\_revenue"))

total\_revenue\_df.show()

**# 2. Filter Transactions with a Discount Greater Than 10%**

high\_discount\_df = ecommerce\_df.filter(col("discount\_percentage") > 10)

high\_discount\_df.show()

**# 3. Find the Most Expensive Product Sold**

expensive\_df = ecommerce\_df.orderBy(col("price").desc()).limit(1)

expensive\_df.show()

**# 4. Calculate the Average Quantity of Products Sold per Category**

avg\_quantity\_df = ecommerce\_df.groupBy("category").agg(F.avg("quantity").alias("average\_quantity"))

avg\_quantity\_df.show()

**# 5. Identify Customers Who Purchased More Than One Product in single transaction**

high\_buy\_df = ecommerce\_df.filter(col("quantity")>1)

high\_buy\_df.show()

**# 6. Find the Top 3 Highest Revenue Transactions**

top\_3\_highest\_df = ecommerce\_df.withColumn("revenue", (col("price") \* col("quantity")) \* (1-col("discount\_percentage")/100)).orderBy(col("revenue").desc()).limit(3)

top\_3\_highest\_df.show()

**# 7. Calculate the Total Number of Transactions per Day**

transaction\_per\_day = ecommerce\_df.groupBy("transaction\_date").agg(F.count("\*").alias("transaction\_count"))

transaction\_per\_day.show()

**# 8. Find the Customer Who Spent the Most Money**

high\_customer\_df = ecommerce\_df.withColumn("total\_spent",(col("price") \* col("quantity")) \* (1-col("discount\_percentage")/100)).groupBy("customer\_id").agg(F.sum("total\_spent").alias("total\_spent")) \

.orderBy(col("total\_spent").desc()).limit(1)

high\_customer\_df.show()

**# 9. Calculate the Average Discount Given per Product Category**

avg\_discount\_df = ecommerce\_df.groupBy("category").agg(F.avg("discount\_percentage").alias("average\_discount"))

avg\_discount\_df.show()

**# 10. Create a New Column for Final Price After Discount**

ecommerce\_df = ecommerce\_df.withColumn("final\_price", col("price") - (col("price") \* col("discount\_percentage") / 100))

ecommerce\_df.show()