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()
```