Databricks - Spark Scripts

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, lit, avg, count, when, round

#□1 Create SparkSession
spark = SparkSession.builder.appName("PopulationPriceExample").getOrCreate()

# 2 Load CSV file
df = spark.read.option("header", True).option("inferSchema",
True).csv("/databricks-datasets/samples/population-vs-price/data_geo.csv")

# Show schema
df.printSchema()

# ======== TRANSFORMATIONS ========

# 1. Select specific columns
df1 = df.select("State", "County", "Population", "MedianHomeValue")

# 2. Filter rows
df2 = df1.filter(col("Population") > 50000)

# 3. Drop duplicates
df3 = df2.dropDuplicates(["State", "County"])

# 4. Add new column
df4 = df3.withColumn("Population_in_Lakhs", round(col("Population") / 100000, 2))

# 5. Rename column
df5 = df4.withColumnRenamed("MedianHomeValue", "Median_House_Value")

# 6. Replace null values
df6 = df5.fillna({"Median_House_Value": 0})

# 7. GroupBy & aggregate
df7 = df6.groupBy("State").agg(
   avg("Median_House_Value").alias("Avg_House_Value"),
   count("*").alias("Total_Counties")
)

# 8. Sort results
df8 = df7.orderBy(col("Avg_House_Value").desc())

# 9. Conditional column
```

```python
df9 = df8.withColumn("High_Value_State", when(col("Avg_House_Value") > 300000,
lit("Yes")).otherwise(lit("No")))

# 10. Limit rows
df10 = df9.limit(10)

# ======== ACTIONS ========

# 11. Show top records
df10.show(truncate=False)

# 12. Collect results
collected_data = df10.collect()

# 13. Count rows
row_count = df10.count()
print(f"Row Count: {row_count}")

# 14. Save to CSV (overwrite mode)
df10.write.mode("overwrite").option("header", True).csv("/tmp/output_population_price")

# 15. Take first 3 rows
sample_data = df10.take(3)
print("Sample Data:", sample_data)

# Stop SparkSession
spark.stop()
```

Breakdown of Operations

#	Type	Operation
1	Transformation	select() – Choose specific columns
2	Transformation	filter() – Keep rows matching condition
3	Transformation	dropDuplicates() – Remove duplicate records
4	Transformation	withColumn() – Create a calculated column
5	Transformation	withColumnRenamed() – Rename column
6	Transformation	fillna() – Replace null values
7	Transformation	groupBy().agg() – Aggregate data
8	Transformation	orderBy() – Sort results
9	Transformation	withColumn() + when() – Conditional values
10	Transformation	limit() – Limit number of rows
11	Action	show() – Display data
12	Action	collect() – Retrieve all rows as Python list
13	Action	count() – Count number of rows
14	Action	write.csv() – Save DataFrame to file
15	Action	take() – Retrieve specific number of rows