
🟢 1. Spark Session & Data Loading

```
from pyspark.sql import SparkSession


spark = SparkSession.builder.appName("Practice").getOrCreate()
```

Read CSV

```
df = spark.read.csv(
    "/FileStore/tables/ecommerce.csv",
    header=True,
    inferSchema=True
)
```

🟢 2. Data Inspection

```
df.show()
df.show(10, truncate=False)
df.printSchema()
df.columns
df.count()
df.describe().show()
```

🟢 3. Column Selection & Renaming

```
df.select("order_id", "price").show()
df.select(df.product, df.category).show()
df.withColumnRenamed("order_date", "date")
```

🟢 4. Filtering Rows

```
df.filter(df.price > 5000).show()
df.where("category = 'Electronics'").show()
```

```
df.filter((df.price > 5000) & (df.quantity >= 2)).show()
```

🟢 5. Create / Modify Columns

```
from pyspark.sql.functions import col

df.withColumn("total_price", col("price") * col("quantity"))
df.drop("customer_id")
```

🟢 6. Aggregations & GroupBy

```
from pyspark.sql.functions import sum, avg, max, min, count

df.groupBy("category").count().show()

df.groupBy("category").agg(
    sum("quantity").alias("total_qty"),
    avg("price").alias("avg_price")
).show()
```

🟢 7. Sorting & Limiting

```
df.orderBy("price", ascending=False).show()
df.orderBy(col("price").desc()).limit(5).show()
```

🟢 8. Distinct & Duplicates

```
df.select("category").distinct().show()
df.dropDuplicates().show()
df.dropDuplicates(["order_id"])
```

🟢 9. Handling Missing Values

```
df.dropna()

df.fillna(0)

df.fillna({"price": 0, "category": "Unknown"})
```

🟢 10. Date & Time Functions

```
from pyspark.sql.functions import to_date, year, month


df.withColumn("order_date", to_date("order_date"))

df.filter(month("order_date") == 1)
```

🟢 11. Cache & Persist

```
df.cache()

df.persist()

df.unpersist()
```

🟢 12. Joins

```
df1.join(df2, "product", "inner")

df1.join(df2, ["product", "category"], "left")
```

Broadcast Join

```
from pyspark.sql.functions import broadcast


df_large.join(broadcast(df_small), "id")
```

🟢 13. Repartition & Coalesce

```
df.repartition(10)

df.coalesce(5)
```

🟢 14. Spark SQL Commands

df.createOrReplaceTempView("ecommerce")

%sql

SELECT category, SUM(quantity) AS total_qty

FROM ecommerce

GROUP BY category

ORDER BY total_qty DESC

🟢 15. Query Plan & Debugging

df.explain()

df.explain(True)

🟢 16. Export Data

CSV

df.write.mode("overwrite").option("header", True).csv("/FileStore/output")

Parquet (Recommended)

df.write.mode("overwrite").parquet("/FileStore/output_parquet")

🟢 1. Spark Session & Data Loading

```
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("Practice").getOrCreate()
```

Read CSV

```
df = spark.read.csv(
    "/FileStore/tables/ecommerce.csv",
    header=True,
    inferSchema=True
)
```

🟢 2. Data Inspection

```
df.show()
df.show(10, truncate=False)
df.printSchema()
df.columns
df.count()
df.describe().show()
```

🟢 3. Column Selection & Renaming

```
df.select("order_id", "price").show()
df.select(df.product, df.category).show()
df.withColumnRenamed("order_date", "date")
```

🟢 4. Filtering Rows

```
df.filter(df.price > 5000).show()
df.where("category = 'Electronics'").show()
df.filter((df.price > 5000) & (df.quantity >= 2)).show()
```

🟢 5. Create / Modify Columns

```
from pyspark.sql.functions import col

df.withColumn("total_price", col("price") * col("quantity"))
df.drop("customer_id")
```

🟢 6. Aggregations & GroupBy

```
from pyspark.sql.functions import sum, avg, max, min, count

df.groupBy("category").count().show()

df.groupBy("category").agg(
    sum("quantity").alias("total_qty"),
    avg("price").alias("avg_price")
).show()
```

🟢 7. Sorting & Limiting

```
df.orderBy("price", ascending=False).show()
df.orderBy(col("price").desc()).limit(5).show()
```

🟢 8. Distinct & Duplicates

```
df.select("category").distinct().show()
df.dropDuplicates().show()
df.dropDuplicates(["order_id"])
```

🟢 9. Handling Missing Values

```
df.dropna()
df.fillna(0)
df.fillna({"price": 0, "category": "Unknown"})
```

🟢 10. Date & Time Functions

```
from pyspark.sql.functions import to_date, year, month

df.withColumn("order_date", to_date("order_date"))
df.filter(month("order_date") == 1)
```

🟢 11. Cache & Persist

```
df.cache()
df.persist()
df.unpersist()
```

🟢 12. Joins

```
df1.join(df2, "product", "inner")
df1.join(df2, ["product", "category"], "left")
```

Broadcast Join

```
from pyspark.sql.functions import broadcast

df_large.join(broadcast(df_small), "id")
```

🟢 13. Repartition & Coalesce

```
df.repartition(10)
df.coalesce(5)
```

🟢 14. Spark SQL Commands

```
df.createOrReplaceTempView("ecommerce")
%sql
```

```
SELECT category, SUM(quantity) AS total_qty
FROM ecommerce
GROUP BY category
ORDER BY total_qty DESC
```

🟢 15. Query Plan & Debugging

```
df.explain()
df.explain(True)
```

🟢 16. Export Data

CSV

```
df.write.mode("overwrite").option("header", True).csv("/FileStore/output")
```

Parquet (Recommended)

```
df.write.mode("overwrite").parquet("/FileStore/output_parquet")
```

🧠 Interview-Critical Commands (Must Remember)

Command	Why Important
`select()`	Column projection
`filter()`	Row filtering
`groupBy()`	Aggregation (shuffle)
`orderBy()`	Global sort
`cache()`	Performance
`broadcast()`	Avoid shuffle
`repartition()`	Control parallelism
`explain()`	Optimization insight

⚡ Databricks Magic Commands

```
%sql
SHOW TABLES;
%python
df.show()
%fs ls /FileStore
```

⚡ Databricks Magic Commands

```
%sql

SHOW TABLES;

%python

df.show()

%fs ls /FileStore
```