Master PySpark: From Zero to Big Data Hero!!

Explode vs Explode_outer

In PySpark, explode and explode_outer are functions used to work with nested data structures, like arrays or maps, by “exploding” (flattening) each element of an array or key-value pair in a map into separate rows. The key difference between explode and explode_outer is in handling null or empty arrays, which makes them useful in different scenarios.

Here's a detailed breakdown of each function, including examples:

1. explode()

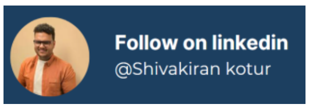
The explode() function takes a column with array or map data and creates a new row for each element in the array (or each key-value pair in the map). If the array is empty or null, explode() will drop the row entirely.

Key Characteristics

- Converts each element in an array or each entry in a map into its own row.
- **Drops rows** with null or empty arrays.

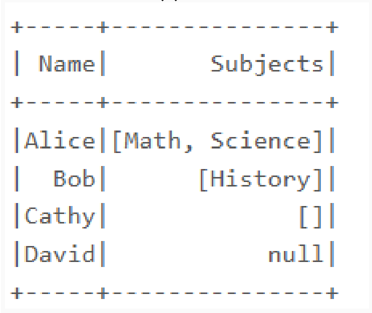
Syntax

```
from pyspark.sql.functions import explode
df.select(explode(df["column_with_array"])).show()

from pyspark.sql import SparkSession
from pyspark.sql import functions as F
from pyspark.sql.functions import explode
# Initialize Spark session
spark =
SparkSession.builder.appName("ExplodeExample").getOrCreate()
# Sample DataFrame with arrays
data = [
    ("Alice", ["Math", "Science"]),
    ("Bob", ["History"]),
    ("Cathy", []),  # Empty array
    ("David", None)  # Null array
]
```



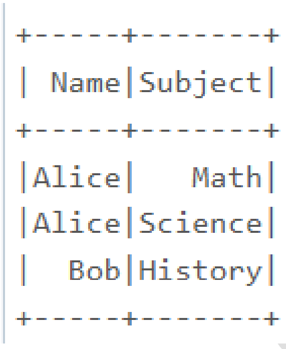
Follow me on LinkedIn – Shivakiran kotur

```
df = spark.createDataFrame(data, ["Name", "Subjects"])

df.show()
```



```
+-----+---------------+
| Name|       Subjects|
+-----+---------------+
|Alice|[Math, Science]|
|  Bob|      [History]|
|Cathy|             []|
|David|           null|
+-----+---------------+
```

```
# Use explode to flatten the array
exploded_df = df.select("Name",
explode("Subjects").alias("Subject"))

# Show the result
exploded_df.show()
```



```
+-----+-------+
| Name|Subject|
+-----+-------+
|Alice|   Math|
|Alice|Science|
|  Bob|History|
+-----+-------+
```

Explanation:

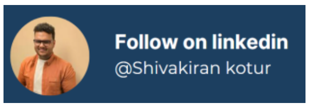
- explode() expands the Subjects array into individual rows.
- Rows with empty ([]) or null arrays (None) are removed, which is why Cathy and David do not appear in the output.

2. explode_outer()

The explode_outer() function works similarly to explode(), but it keeps rows with null or empty arrays. When explode_outer() encounters a null or empty array, it still generates a row for that entry, with null as the value in the resulting column.

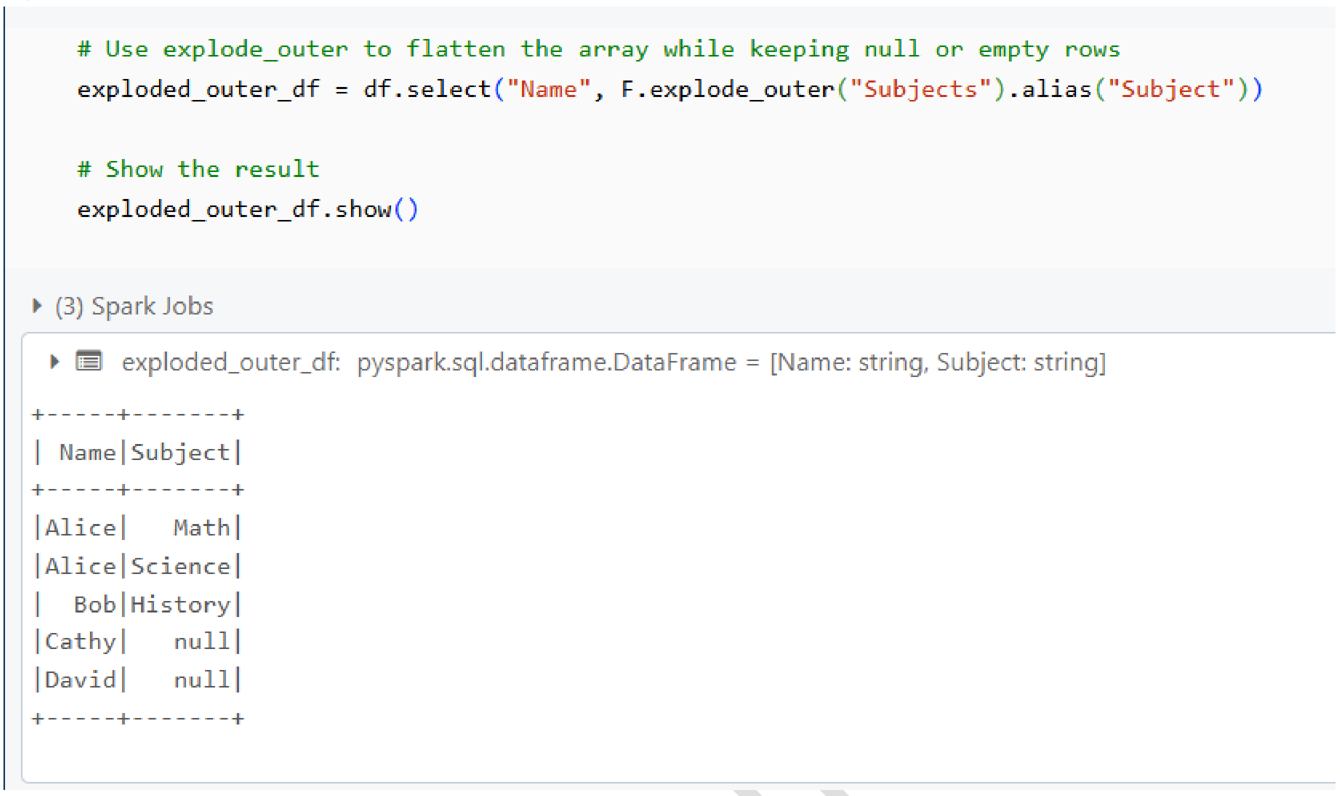
Key Characteristics

- Converts each element in an array or each entry in a map into its own row.
- **Retains rows** with null or empty arrays, using null values in the exploded column.



Follow me on LinkedIn – Shivakiran kotur

Syntax



```
# Use explode_outer to flatten the array while keeping null or empty rows
exploded_outer_df = df.select("Name", F.explode_outer("Subjects").alias("Subject"))

# Show the result
exploded_outer_df.show()
```

▸ (3) Spark Jobs

▸ exploded_outer_df: pyspark.sql.dataframe.DataFrame = [Name: string, Subject: string]

```
+-----+-------+
| Name|Subject|
+-----+-------+
|Alice|   Math|
|Alice|Science|
|  Bob|History|
|Cathy|   null|
|David|   null|
+-----+-------+
```

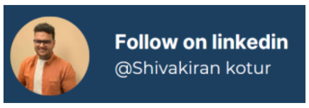
Explanation:

- explode_outer() expands the Subjects array into individual rows.
- Unlike explode(), rows with empty ([]) or null arrays (None) are kept in the result, with null values in the Subject column for these cases.

Summary Table of Differences

Function	Description	Null/Empty Arrays Behavior
explode()	Expands each element of an array or map into individual rows	Drops rows with null or empty arrays
explode_outer()	Similar to explode(), but retains rows with null or empty arrays	Keeps rows with null or empty arrays, filling with null

These functions are very useful when working with complex, nested data structures, especially when dealing with JSON or other hierarchical data.



Follow me on LinkedIn – Shivakiran kotur