Master PySpark: From Zero to Big Data Hero!!

Here's an example of a PySpark DataFrame with data and corresponding notes that explain the various transformations, sorting, and string functions:

Sample Data Creation

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, desc, asc, concat,
concat_ws, initcap, lower, upper, instr, length, lit

# Create a Spark session
spark =
SparkSession.builder.appName("SortingAndStringFunctions").getOrCrea
te()

# Sample data
data = [
    ("USA", "North America", 100, 50.5),
    ("India", "Asia", 300, 20.0),
    ("Germany", "Europe", 200, 30.5),
    ("Australia", "Oceania", 150, 60.0),
    ("Japan", "Asia", 120, 45.0),
    ("Brazil", "South America", 180, 25.0)
]

# Define the schema
columns = ["Country", "Region", "UnitsSold", "UnitPrice"]

# Create DataFrame
df = spark.createDataFrame(data, columns)

# Display the original DataFrame
df.show()
```

Notes with Examples

Sorting the DataFrame

1. Sort by a single column (ascending order):

```
df.orderBy("Country").show(5)
```

▸ (1) Spark Jobs

```
+---------+-------------+---------+---------+
|  Country|       Region|UnitsSold|UnitPrice|
+---------+-------------+---------+---------+
|Australia|      Oceania|      150|     60.0|
|   Brazil|South America|      180|     25.0|
|  Germany|       Europe|      200|     30.5|
|    India|         Asia|      300|     20.0|
|    Japan|         Asia|      120|     45.0|
+---------+-------------+---------+---------+
only showing top 5 rows
```

Note: By default, the sorting is in ascending order. This shows the top 5 countries in alphabetical order.

2. Sort by multiple columns:

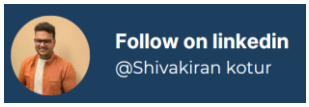
```
df.orderBy("Country", "UnitsSold").show(5)
```

▸ (1) Spark Jobs

```
+---------+-------------+---------+---------+
|  Country|       Region|UnitsSold|UnitPrice|
+---------+-------------+---------+---------+
|Australia|      Oceania|      150|     60.0|
|   Brazil|South America|      180|     25.0|
|  Germany|       Europe|      200|     30.5|
|    India|         Asia|      300|     20.0|
|    Japan|         Asia|      120|     45.0|
+---------+-------------+---------+---------+
only showing top 5 rows
```

Note: Here, the DataFrame is sorted first by Country (ascending), and within the same country, it is sorted by UnitsSold in ascending order.

3. Sort by a column in descending order and limit:



```
sorted_df = df.orderBy(desc("Country")).limit(3)
sorted_df.show()
```

(1) Spark Jobs

sorted_df: pyspark.sql.dataframe.DataFrame = [Country: string, Region: string ... 2 more fields]

```
+-------+-------------+---------+---------+
|Country|       Region|UnitsSold|UnitPrice|
+-------+-------------+---------+---------+
|    USA|North America|      100|     50.5|
|  Japan|         Asia|      120|     45.0|
|  India|         Asia|      300|     20.0|
+-------+-------------+---------+---------+
```

Note: This sorts the DataFrame by Country in descending order and limits the output to the top 3 rows.

4. Sorting with null values last:

```
sorted_df = df.orderBy(col("Country").desc(), nulls_last=True).show(5)
```

(1) Spark Jobs

```
+-------+-------------+---------+---------+
|Country|       Region|UnitsSold|UnitPrice|
+-------+-------------+---------+---------+
|    USA|North America|      100|     50.5|
|  Japan|         Asia|      120|     45.0|
|  India|         Asia|      300|     20.0|
|Germany|       Europe|      200|     30.5|
| Brazil|South America|      180|     25.0|
+-------+-------------+---------+---------+
only showing top 5 rows
```

Note: This ensures that null values (if present) are placed at the end when sorting by Country.

Summary of Key Functions:

- Sorting: You can sort a DataFrame by one or more columns using .orderBy() or .sort(). By default, sorting is ascending, but you can change it using asc() or desc().

These functions and transformations are common in PySpark for manipulating and querying data effectively!