Master PySpark: From Zero to Big Data Hero!!

Joins Part 2

```
from pyspark.sql import SparkSession
from pyspark.sql import Row
from pyspark.sql.functions import broadcast

# Initialize Spark session
spark = SparkSession.builder.appName("JoinsExample").getOrCreate()

# Sample DataFrames
data1 = [Row(id=0), Row(id=1), Row(id=1), Row(id=None),
Row(id=None)]
data2 = [Row(id=1), Row(id=0), Row(id=None)]
df1 = spark.createDataFrame(data1)
df2 = spark.createDataFrame(data2)

# Inner Join
inner_join = df1.join(df2, on="id", how="inner")
print("Inner Join:")
inner_join.show()
```

```
Inner Join:
+---+
| id|
+---+
|  0|
|  1|
|  1|
+---+
```

```
# Left Join
left_join = df1.join(df2, on="id", how="left")
print("Left Join:")
left_join.show()
```

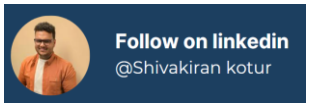
```
Left Join
+----+
|  id|
+----+
|   0|
|   1|
|   1|
|null|
|null|
+----+
```

```
# Right Join
right_join = df1.join(df2, on="id", how="right")
print("Right Join:")
right_join.show()
```

```
Right Join:
+----+
|  id|
+----+
|   1|
|   1|
|   0|
|null|
+----+
```

```
# Full (Outer) Join
full_join = df1.join(df2, on="id", how="outer")
print("Full (Outer) Join:")
full_join.show()
```

```
Full (Outer) Join:
+----+
|  id|
+----+
|null|
|null|
|null|
|   0|
|   1|
|   1|
+----+
```



Follow me on LinkedIn – Shivakiran kotur

```python
# Left Anti Join
left_anti_join = df1.join(df2, on="id", how="left_anti")
print("Left Anti Join:")
left_anti_join.show()
```

```
Left Anti Join:
+----+
|  id|
+----+
|null|
|null|
+----+
```

```python
# Right Anti Join (Equivalent to swapping DataFrames and performing Left Anti Join)
right_anti_join = df2.join(df1, on="id", how="left_anti")
print("Right Anti Join:")
right_anti_join.show()
```

```
Right Anti Join:
+----+
|  id|
+----+
|null|
+----+
```

```python
# Broadcast Join (Optimizing a join with a smaller DataFrame)
broadcast_join = df1.join(broadcast(df2), on="id", how="inner")
print("Broadcast Join:")
broadcast_join.show()
```

```
Broadcast Join:
+---+
| id|
+---+
|  0|
|  1|
|  1|
+---+
```

Comparison: Left Join vs. Left Anti Join

- **Left Join**:
  - Keeps all rows from the left DataFrame and includes matching rows from the right, filling in null for unmatched rows.
- **Left Anti Join**:
  - Keeps only rows from the left DataFrame that do not have a match in the right DataFrame.
- **Summary**: Choose a left join to combine data and keep all rows from the left DataFrame. Use a left anti join to identify entries unique to the left DataFrame.

Broadcast Joins in PySpark

- **Definition**: A broadcast join optimizes joins when one DataFrame is small enough to fit into memory by broadcasting it to all nodes. This eliminates the need to shuffle data across the cluster, significantly improving performance for large datasets.
- **Usage**: Recommended for joining a large DataFrame with a small DataFrame (that can fit into memory). You can force a broadcast join using broadcast(df_small).

- **Advantages**:
  - Avoids data shuffling, which can speed up processing for suitable cases.
  - Reduces memory consumption and network I/O for specific types of joins.

In summary:

- **Left Anti Join** is useful for identifying non-matching rows from one DataFrame against another.
- **Broadcast Join** is a performance optimization technique ideal for joining a large DataFrame with a small one efficiently, reducing shuffle costs.

These concepts help you manage and optimize your data processing tasks efficiently in PySpark.