# How to Use PySpark `when` with OR Conditions for Conditional Logic

Authored by

**stats writer**

February 3, 2026

The PySpark "when" function is a powerful tool that allows users to apply conditional logic to their data in a Spark environment. This function can be used to create new columns or modify existing ones based on a specified condition. To use the "when" function with an OR condition, the user can simply add multiple conditions separated by the logical operator "or" within the function. This will allow the function to check for either of the conditions and perform the specified action accordingly. By utilizing the "when" function with an OR condition, users can efficiently manipulate their data and achieve their desired results.

## PySpark: Use When with OR Condition

## You can use the following syntax to use the when function in PySpark with or conditions:

## import pyspark.sql.functions as F

## df_new = df.withColumn('B10', F.when((df.team=='B') | (df.points>10), 1).otherwise(0))

## This particular example creates a new column named B10 that returns the following values:

## 1 if the team column is B or the points column is greater than 100 otherwise

## Note: The | symbol represents an "OR" operator in PySpark.

## The following example shows how to use this syntax in

practice.

### Example: How to Use When with OR Condition in PySpark

Suppose we have the following PySpark DataFrame that contains information about various basketball players:

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.getOrCreate()

#define data
data = ,
,
,
,
,
,
]

#define column names
columns =

#create dataframe using data and column names
df = spark.createDataFrame(data, columns)
```

```
#view dataframe
df.show()

+----+--------+------+
|team|position|points|
+----+--------+------+
| A| Guard| 11|
| A| Guard| 8|
| A| Forward| 22|
| A| Forward| 22|
| B| Guard| 14|
| B| Guard| 14|
| B| Forward| 13|
| B| Forward| 7|
+----+--------+------+
```

We can use the following syntax to create a new column named B10 that returns 1 if the team is A or the points is greater than 10, or 0 otherwise:

```
import pyspark.sql.functions as F

#create new DataFrame
df_new = df.withColumn('B10', F.when((df.team=='B') |
(df.points>10), 1).otherwise(0))
```

```
#view new DataFrame
df_new.show()

+----+--------+------+---+
|team|position|points|B10|
+----+--------+------+---+
| A| Guard| 11| 1|
| A| Guard| 8| 0|
| A| Forward| 22| 1|
| A| Forward| 22| 1|
| B| Guard| 14| 1|
| B| Guard| 14| 1|
| B| Forward| 13| 1|
| B| Forward| 7| 1|
+----+--------+------+---+
```

Notice that all of the values in the B10 column that meet either condition receive a value of 1.

Only one row doesn't meet either condition, which is the one row that received a value of 0.

Also note that you could return 'Yes' or 'No' instead of 1 and 0 by using the following syntax:

```
import pyspark.sql.functions as F

#create new DataFrame
df_new = df.withColumn('B10', F.when((df.team=='B') | (df.points>10), 'Yes').otherwise('No'))

#view new DataFrame
df_new.show()

+----+--------+------+---+
|team|position|points|B10|
+----+--------+------+---+
| A| Guard| 11|Yes|
| A| Guard| 8| No|
| A| Forward| 22|Yes|
| A| Forward| 22|Yes|
| B| Guard| 14|Yes|
| B| Guard| 14|Yes|
| B| Forward| 13|Yes|
| B| Forward| 7|Yes|
+----+--------+------+---+
```

The new B10 column now returns 'Yes' or 'No' instead of 1 or 0.

**Feel free to return whatever values you'd like by specifying them in the when and otherwise functions.**

**The following tutorials explain how to perform other common tasks in PySpark:**

## PySpark: How to Use When with AND Condition