How to Sort PySpark DataFrames in Descending Order Using Window.orderBy()

Authored by

stats writer

January 18, 2026

You can use the following syntax to use **Window.orderBy()** and sort in descending order in PySpark:

from pyspark.sql.functions import row_number, desc
from pyspark.sql.window import Window

#specify window
w = Window.partitionBy('team').orderBy(desc('points'))

#add column called 'id' that contains row numbers
df = df.withColumn('id', row_number().over(w))

This particular example adds a new column to the DataFrame called **id** that contains row numbers for each row, grouped by **team** and sorted by the values in the **points** column in descending order.

The following example shows how to use this syntax in practice.

Example: How to Use Window.orderBy() Descending in PySpark

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

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| 21|
| A| Forward| 22|
| A| Forward| 30|
| B| Guard| 14|
| B| Guard| 14|
| B| Forward| 13|
| B| Forward| 7|
+----+--------+------+
```

Suppose we would like to add a new column named **id** that contains row numbers for each row in the DataFrame, grouped by the **team** column and sorted based on the values in the **points** column in descending order.

We can use the following syntax to do so:

```
from pyspark.sql.functions import row_number, desc
from pyspark.sql.window import Window

#specify window
w = Window.partitionBy('team').orderBy(desc('points'))

#add column called 'id' that contains row numbers
df = df.withColumn('id', row_number().over(w))

#view DataFrame
df.show()

+----+--------+------+---+
|team|position|points| id|
+----+--------+------+---+
| A| Forward| 30| 1|
| A| Forward| 22| 2|
| A| Forward| 21| 3|
```

A	Guard	11	4
A	Guard	8	5
B	Guard	14	1
B	Guard	14	2
B	Forward	13	3
B	Forward	7	4
+----+--------+------+---+

The new column named **id** contains row numbers for each row in the DataFrame, grouped by the **team** column and sorted based on the values in the **points** column in descending order.

For example:

The row with the largest points value for team A receives an **id** value of **1**.
The row with the next largest points value for team A receives an **id** value of **2**.

And so on.

Note #1: If we didn't use the **desc** function within the **orderBy** function, the row numbers would have been assigned based on the values in the **points** column in ascending order instead.

Note #2: You can find the complete documentation for the PySpark **Window.orderBy** function .