How to Join PySpark DataFrames on Columns with Different Names

Authored by

stats writer

February 7, 2026

PySpark is a data processing framework that allows users to perform advanced operations on large datasets. One common operation is the "join" operation, which combines data from multiple tables or datasets based on a common key. However, a question arises if this operation can be performed on columns with different names. The answer is yes, PySpark offers the flexibility to perform join operations on columns with different names by specifying the appropriate arguments and renaming the columns if necessary. This allows for a seamless integration of data from disparate sources and enables users to efficiently analyze and extract insights from their data.

PySpark: Join on Different Column Names

You can use the following syntax to join two DataFrames together based on different column names in PySpark:

df3 = df1.withColumn('id', col('team_id')).join(df2.withColumn('id', col('team_name')), on='id')

Here is what this syntax does:

First, it renames the team_id column from df1 to id.Then, it renames the team_name column from df2 to id.Lastly, it joins together df1 and df2 based on values in the id columns.

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

Example: How to Join on Different Column Names in PySpark

Suppose we have the following DataFrame named df1:

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

#define data
data = ,
,
,
,
,
]

#define column names
columns =

#create dataframe using data and column names
df1 = spark.createDataFrame(data, columns)

#view dataframe
df1.show()

+-------+------+
```

|team_ID|points|

+-------+------+

| Mavs| 18|

| Nets| 33|

| Lakers| 12|

| Kings| 15|

| Hawks| 19|

|Wizards| 24|

| Magic| 28|

+-------+------+

And suppose we have another DataFrame named df2:

#define data

data = ,

,

,

,

,

]

#define column names

columns =

```
#create dataframe using data and column names
df2 = spark.createDataFrame(data, columns)

#view dataframe
df2.show()

+---------+-------+
|team_name|assists|
+---------+-------+
| Hawks| 4|
| Wizards| 5|
| Raptors| 5|
| Kings| 12|
| Mavs| 7|
| Nets| 11|
| Magic| 3|
+---------+-------+
```


We can use the following syntax to perform an inner join between these two DataFrames by renaming the team columns from each DataFrame to id and then by joining on values from the id column:

```
#join df1 and df2 on different column names
df3 = df1.withColumn('id',
```

```
col('team_id')).join(df2.withColumn('id',
col('team_name')), on='id')
#view resulting DataFrame
df3.show()

+-------+-------+------+---------+-------+
| id|team_ID|points|team_name|assists|
+-------+-------+------+---------+-------+
| Hawks| Hawks| 19| Hawks| 4|
| Kings| Kings| 15| Kings| 12|
| Magic| Magic| 28| Magic| 3|
| Mavs| Mavs| 18| Mavs| 7|
| Nets| Nets| 33| Nets| 11|
|Wizards|Wizards| 24| Wizards| 5|
+-------+-------+------+---------+-------+
```

We have successfully joined the two DataFrames into one DataFrame based on matching values in the new id column.

Note that you can also use the select function to only display certain columns in the resulting joined DataFrame.

For example, we can use the following syntax to only

display the id, points and assists columns in the joined DataFrame:

```
#join df1 and df2 on different column names
df3 = df1.withColumn('id', col('team_id')).join(df2.withColumn('id', col('team_name')), on='id')
.select('id', 'points', 'assists')
#view resulting DataFrame
df3.show()
```

```
+-------+------+-------+
| id|points|assists|
+-------+------+-------+
| Hawks| 19| 4|
| Kings| 15| 12|
| Magic| 28| 3|
| Mavs| 18| 7|
| Nets| 33| 11|
|Wizards| 24| 5|
+-------+------+-------+
```

Notice that only the id, points and assists columns are shown in the joined DataFrame.

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