How to Group a PySpark DataFrame by Month

Authored by

stats writer

January 2, 2026

In PySpark DataFrame, you can use the groupBy() method and pass it a list of columns to group by. For example, to group by month, you can pass in the month column as an argument. This will group the DataFrame by each month and create a new DataFrame with the grouped data. You can then use the groupBy() method to perform aggregations like sum, count, avg, etc. on the grouped DataFrame to get the desired results.

You can use the following syntax to group rows by month in a PySpark DataFrame:

from pyspark.sql.functions import month, sum

df.groupBy(month('date').alias('month')).agg(sum('sales').alias('sum_sales')).show()

This particular example groups the rows of the DataFrame by month based on the date in the **date** column and then calculates the sum of the values in the **sales** column by month.

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

Example: How to Group by Month in PySpark

Suppose we have the following PySpark DataFrame that contains information about the sales made on various days at some company:

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()

+----------+-----+
| date|sales|
+----------+-----+
|2023-04-11| 22|
|2023-04-15| 14|
|2023-04-17| 12|
|2023-05-21| 15|
|2023-05-23| 30|
|2023-10-26| 45|
|2023-10-28| 32|
|2023-10-29| 47|
+----------+-----+
```

Suppose we would like to calculate the sum of the sales, grouped by month.

We can use the following syntax to do so:

```
from pyspark.sql.functions import month, sum

#calculate sum of sales by month
df.groupBy(month('date').alias('month')).agg(sum('sales').alias('sum_sales')).show()

+-----+---------+
|month|sum_sales|
+-----+---------+
| 4| 48|
| 5| 45|
| 10| 124|
+-----+---------+
```

The resulting DataFrame shows the sum of sales by month.

For example, we can see:

The sum of sales for dates in April (month 4) is **48**.
The sum of sales for dates in May (month 5) is **45**.
The sum of sales for dates in October (month 10) is **124**.

Note that you can also aggregate sales by a different metric if you'd like.

For example, you could use the following syntax to calculate the total count of sales, grouped by month:

from pyspark.sql.functions import month, count

#calculate count of sales by month
df.groupBy(month('date').alias('month')).agg(count('sales').alias('cnt_sales')).show()

```
+-----+---------+
|month|cnt_sales|
+-----+---------+
|    4|        3|
|    5|        2|
|   10|        3|
+-----+---------+
```

The resulting DataFrame now shows the count of sales by month.

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