How can I use the groupBy function in PySpark DataFrame to sort the data in descending order?

Authored by

stats writer

June 24, 2024

The groupBy function in PySpark DataFrame allows users to group data based on a specified column or multiple columns. By using this function, users can then apply aggregate functions, such as sum or count, to the grouped data. In order to sort the data in descending order, users can use the orderBy function in conjunction with the groupBy function. This will allow them to specify the column to sort by and the desired sorting order, in this case, descending. This can be useful for organizing and analyzing large datasets in a more organized and meaningful manner.

PySpark DataFrame `groupBy(), filter(),` and `sort()` - In this PySpark example, let's see how to do the following operations in sequence 1) DataFrame group by using aggregate function sum(), 2) filter() the group by result, and 3) sort() or orderBy() to do descending or ascending order.

In order to demonstrate all these operations together, let's create a PySpark DataFrame.

```
simpleData =

schema =
df = spark.createDataFrame(data=simpleData, schema = schema)
df.printSchema()
df.show(truncate=False)
```

Using DataFrame groupBy(), filter() and sort()

Below is a complete PySpark DataFrame example of how to do group by, filter, and sort by descending order.

```
from pyspark.sql.functions import sum, col, desc
df.groupBy("state")
.agg(sum("salary").alias("sum_salary"))
.filter(col("sum_salary") > 100000)
.sort(desc("sum_salary"))
.show()

#+-----+----------+
#|state|sum_salary|
#+-----+----------+
#| NY| 252000|
#| CA| 171000|
#| NV| 166000|
#+-----+----------+
```

Alternatively, you can use the following SQL expression to achieve the same result.

```
df.createOrReplaceTempView("EMP")
spark.sql("select state, sum(salary) as sum_salary from EMP " +
"group by state having sum_salary > 100000 " +
"order by sum_salary desc").show()
```

Below is an Explanation of First DataFrame Example.

First, let's do a PySpark groupBy() on Dataframe by using an aggregate function `sum("salary")`, groupBy() returns `GroupedData` object which contains aggregate functions like `sum()`, `max()`, `min()`, `avg()`, `mean()`, `count()`.

```
df.groupBy("state").sum("salary").show()
#+-----+-----------+
#|state|sum(salary)|
#+-----+-----------+
#| NJ| 91000|
#| NV| 166000|
#| CA| 171000|
#| DE| 99000|
#| NY| 252000|
#+-----+-----------+
```

In the above example, you cannot give an alias name to the aggregate column and the column name would be defaults to agg function name and column name `(sum("salary"))`. This default name is not user friendly hence let's see how to provide an alias by using `agg()` function. Also, to do filter and sort it's better if you know the exact column name.

```
# Group by using by giving alias name.
from pyspark.sql.functions import sum
dfGroup=df.groupBy("department")
.agg(sum("bonus").alias("sum_salary"))
```

#+-----+----------+
#|state|sum_salary|
#+-----+----------+
#|NJ |91000 |
#|NV |166000 |

#|CA |171000 |
#|DE |99000 |
#|NY |252000 |
#+-----+----------+

Here, I have used the PySpark SQL function `sum()` that returns a Column type and uses `alias()` of this class. Now let's see how to filter the group by data. The below example selects the data that has a total sum of a salary greater than 100,000.

```
# Filter after group by
dfFilter=dfGroup.filter(dfGroup.sum_salary > 100000)
dfFilter.show()
```

#+-----+----------+
#|state|sum_salary|
#+-----+----------+
#| NV| 166000|
#| CA| 171000|
#| NY| 252000|
#+-----+----------+

Now, let's use DataFrame sort() transformation to sort on group by column, by default it does sort by ascending order.

```
# Sory by on group by column
from pyspark.sql.functions import asc
dfFilter.sort("sum_salary").show()
```

#+-----+----------+
#|state|sum_salary|
#+-----+----------+
#| NV| 166000|
#| CA| 171000|
#| NY| 252000|
#+-----+----------+

In order to descending order, use Spark SQL function `desc()`.

```
# Sort by descending order.
from pyspark.sql.functions import desc
dfFilter.sort(desc("sum_salary")).show()
```

#+-----+----------+
#|state|sum_salary|
#+-----+----------+
#| NY| 252000|
#| CA| 171000|
#| NV| 166000|
#+-----+----------+

Conclusion

In this PySpark example I have explained how to do DataFrame groupby(), filter() and sort() by descending order. hope you like it.

Happy Learning !!

Related Articles