PySpark Aggregate Functions with Examples

NNK (https://sparkbyexamples.com/author/admin/)

PySpark (https://sparkbyexamples.com/category/pyspark/)

PySpark provides built-in standard Aggregate functions defines in DataFrame API, these come in handy when we need to make aggregate operations on DataFrame columns. Aggregate functions operate on a group of rows and calculate a single return value for every group.

All these aggregate functions accept input as, Column type or column name in a string and several other arguments based on the function and return Column type.

When possible try to leverage standard library as they are little bit more compile-time safety, handles null and perform better when compared to UDF's. If your application is critical on performance try to avoid using custom UDF at all costs as these are not guarantee on performance.

PySpark Aggregate Functions

PySpark SQL Aggregate functions are grouped as "agg_funcs" in Pyspark. Below is a list of functions defined under this group. Click on each link to learn with example.

- approx_count_distinct
- avg
- collect_list
- collect_set
- countDistinct
- count
- grouping
- first
- last
- kurtosis
- max
- min
- mean
- skewness
- stddev
- stddev_samp
- stddev_pop
- sum
- sumDistinct
- variance, var_samp, var_pop

PySpark Aggregate Functions Examples

First, let's create a DataFrame (https://sparkbyexamples.com/pyspark/different-ways-to-create-dataframe-in-pyspark/) to work with PySpark aggregate functions. All examples provided here are also available at PySpark Examples GitHub (https://github.com/spark-

examples/pyspark-examples/blob/master/pyspark-aggregate.py)project.


```
simpleData = [("James", "Sales"
    ("Michael", "Sales", 4600),
    ("Robert", "Sales", 4100),
    ("Maria", "Finance", 3000),
    ("James", "Sales", 3000),
    ("Scott", "Finance", 3300),
    ("Jen", "Finance", 3900),
    ("Jeff", "Marketing", 3000)
    ("Kumar", "Marketing", 2000
    ("Saif", "Sales", 4100)
  ]
schema = ["employee_name", "dep
df = spark.createDataFrame(data
df.printSchema()
df.show(truncate=False)
```

Yields below output.

```
+-------------+----------+-----
|employee_name|department|salar
+-------------+----------+-----
|        James|     Sales|   300
|      Michael|     Sales|   460
|       Robert|     Sales|   410
|        Maria|   Finance|   300
|        James|     Sales|   300
|        Scott|   Finance|   330
|          Jen|   Finance|   390
|         Jeff| Marketing|   300
|        Kumar| Marketing|   200
|         Saif|     Sales|   410
+-------------+----------+-----
```

Now let's see how to aggregate data in PySpark.

approx_count_distinct Aggregate Function

In PySpark `approx_count_distinct()` function returns the count of distinct items in a group.

```
//approx_count_distinct()
print("approx_count_distinct: "
      str(df.select(approx_coun

//Prints approx_count_distinct:
```

avg (average) Aggregate Function

`avg()` function returns the average of values in the input column.

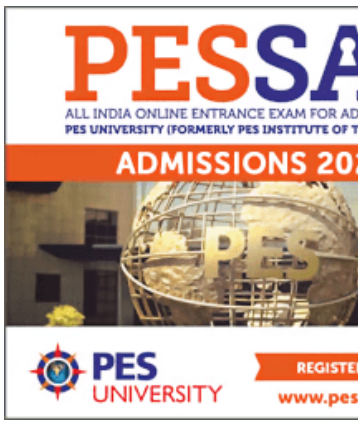
```
//avg
print("avg: " + str(df.select(a

//Prints avg: 3400.0
```

collect_list Aggregate Function

`collect_list()` function returns all values from an input column with duplicates.

pyspark/pyspark-read-and-write-parquet-file/)

PySpark – Read & Write JSON file (https://sparkbyexamples.com/pyspark/pyspark-read-json-file-into-dataframe/)



PySpark Built-In Functions

PySpark – when() (https://sparkbyexamples.com/pyspark/pyspark-when-otherwise/)

PySpark – expr() (https://sparkbyexamples.com/pyspark/pyspark-sql-expr-expression-function/)

PySpark – lit() (https://sparkbyexamples.com/pyspark/pyspark-lit-add-literal-constant/)

PySpark – split() (https://sparkbyexamples.com/pyspark/pyspark-convert-string-to-array-column/)

PySpark – concat_ws() (https://sparkbyexamples.com/pyspark/pyspark-convert-array-column-to-string-column/)

Pyspark – substring() (https://sparkbyexamples.com/pyspark/pyspark-substring-from-a-column/)

PySpark – translate() (https://sparkbyexamples.com/pyspark/pyspark-replace-column-values/#translate-replace-character-by-character)

PySpark – regexp_replace() (https://sparkbyexamples.com/pyspark/pyspark-replace-column-

```
//collect_list
df.select(collect_list("salary"

+-----------------------------
|collect_list(salary)
+-----------------------------
|[3000, 4600, 4100, 3000, 3000,
-----------------------------
```

llect_set Aggregate nction

lect_set() function returns all es from an input column with duplicate values eliminated.

```
//collect_set
df.select(collect_set("salary")

+-----------------------------
|collect_set(salary)
+-----------------------------
|[4600, 3000, 3900, 4100, 3300,
+-----------------------------
```

countDistinct Aggregate Function

countDistinct() function returns the number of distinct elements in a columns

```
//countDistinct
df2 = df.select(countDistinct("
df2.show(truncate=False)
print("Distinct Count of Departu
```

count function

count() function returns number of elements in a column.

```
print("count: "+str(df.select(c

Prints county: 10
```

grouping function

`grouping()` Indicates whether a given input column is aggregated or not. returns 1 for aggregated or 0 for not aggregated in the result. If you try grouping directly on the salary column you will get below error.

```
Exception in thread "main" org.
  // grouping() can only be use
```

first function

`first()` function returns the first element in a column when ignoreNulls is set to true, it returns the first non-null element.

```
//first
df.select(first("salary")).show

+--------------------+
|first(salary, false)|
+--------------------+
|3000                |
+--------------------+
```

last function

`last()` function returns the last element in a column. when ignoreNulls is set to true, it returns the last non-null element.

```
//last
df.select(last("salary")).show(

+-------------------+
|last(salary, false)|
+-------------------+
|4100               |
+-------------------+
```

kurtosis function

`kurtosis()` function returns the kurtosis of the values in a group.

```
df.select(kurtosis("salary")).s

+-------------------+
|kurtosis(salary)   |
+-------------------+
|-0.6467803030303032|
+-------------------+
```

max function

`max()` function returns the maximum value in a column.

```
df.select(max("salary")).show(t

+-----------+
|max(salary)|
+-----------+
|4600       |
+-----------+
```

min function

`min()` function

```
df.select(min("salary")).show(t

+-----------+
|min(salary)|
+-----------+
|2000       |
+-----------+
```

mean function

`mean()` function returns the average of the values in a column. Alias for Avg

```
df.select(mean("salary")).show(

+-----------+
|avg(salary)|
+-----------+
|3400.0     |
+-----------+
```

skewness function

`skewness()` function returns the skewness of the values in a group.

```
df.select(skewness("salary")).s

+--------------------+
|skewness(salary)    |
+--------------------+
|-0.12041791181069571|
+--------------------+
```

stddev(), stddev_samp() and stddev_pop()

`stddev()` alias for `stddev_samp`.

`stddev_samp()` function returns the sample standard deviation of values in a column.

`stddev_pop()` function returns the population standard deviation of the values in a column.

```
df.select(stddev("salary"), std
    stddev_pop("salary")).show(

+------------------+----------
|stddev_samp(salary)|stddev_sam
+------------------+----------
|765.9416862050705  |765.941686
+------------------+----------
```

sum function

`sum()` function Returns the sum of all values in a column.

```
df.select(sum("salary")).show(t

+-----------+
|sum(salary)|
+-----------+
|34000      |
+-----------+
```

sumDistinct function

`sumDistinct()` function returns the sum of all distinct values in a column.

```
df.select(sumDistinct("salary")

+--------------------+
|sum(DISTINCT salary)|
+--------------------+
|20900               |
+--------------------+
```

variance(), var_samp(), var_pop()

`variance()` alias for `var_samp`

`var_samp()` function returns the unbiased variance of the values in a column.

`var_pop()` function returns the population variance of the values in a column.

```
df.select(variance("salary"),va
  .show(truncate=False)

+-----------------+------------
|var_samp(salary) |var_samp(sal
+-----------------+------------
|586666.6666666666|586666.66666
+-----------------+------------
```

Source code of PySpark Aggregate examples

```
import pyspark
from pyspark.sql import SparkSe
from pyspark.sql.functions impo
from pyspark.sql.functions impo
from pyspark.sql.functions impo
from pyspark.sql.functions impo
from pyspark.sql.functions impo

spark = SparkSession.builder.ap

simpleData = [("James", "Sales"
    ("Michael", "Sales", 4600),
    ("Robert", "Sales", 4100),
    ("Maria", "Finance", 3000),
    ("James", "Sales", 3000),
    ("Scott", "Finance", 3300),
    ("Jen", "Finance", 3900),
    ("Jeff", "Marketing", 3000)
    ("Kumar", "Marketing", 2000
    ("Saif", "Sales", 4100)
  ]
schema = ["employee_name", "dep

df = spark.createDataFrame(data
df.printSchema()
df.show(truncate=False)

print("approx_count_distinct: "
      str(df.select(approx_coun

print("avg: " + str(df.select(a

df.select(collect_list("salary"

df.select(collect_set("salary")

df2 = df.select(countDistinct("
df2.show(truncate=False)
print("Distinct Count of Depart

print("count: "+str(df.select(c
df.select(first("salary")).show
df.select(last("salary")).show(
df.select(kurtosis("salary")).s
df.select(max("salary")).show(t
df.select(min("salary")).show(t
df.select(mean("salary")).show(
df.select(skewness("salary")).s
df.select(stddev("salary"), std
    stddev_pop("salary")).show(
df.select(sum("salary")).show(t
df.select(sumDistinct("salary")
df.select(variance("salary"),va
  .show(truncate=False)
```

Conclusion

In this article, I've consolidated and listed all PySpark Aggregate functions with scala examples and also learned the benefits of using PySpark SQL functions.

Happy Learning !!

Share this:

(https://sparkbyexamples.com/pyspark/pyspark-aggregate-functions/?share=facebook&nb=1)

(https://sparkbyexamples.com/pyspark/pyspark-aggregate-functions/?share=reddit&nb=1)

(https://sparkbyexamples.com/pyspark/pyspark-aggregate-functions/?share=pinterest&nb=1) 2

(https://sparkbyexamples.com/pyspark/pyspark-aggregate-functions/?share=tumblr&nb=1)

(https://sparkbyexamples.com/pyspark/pyspark-aggregate-functions/?share=pocket&nb=1)

(https://sparkbyexamples.com/pyspark/pyspark-aggregate-functions/?share=linkedin&nb=1)

(https://sparkbyexamples.com/pyspark/pyspark-aggregate-functions/?share=twitter&nb=1)

TAGS: AGGREGATE FUNCTIONS (HTTPS://SPARKBYEXAMPLES.COM/TAG/AGGREGATE-FUNCTIONS/)

(https://sparkbyexamples.com/author/admin/)

NNK

(Https://Sparkbyexamples.Com/Author/Admin/)

SparkByExamples.com is a Big Data and Spark examples community page, all examples are simple and easy to understand and well tested in our development environment Read more .. (https://sparkbyexamples.com/about-sparkbyexamples/)

THIS POST HAS ONE COMMENT

Tw 30 JAN 2021 REPLY

Thank you for all your efforts in putting PySpark operations together. It is very helpful.

Leave a Reply

Apache HBase (https://sparkbyexamples.com/category/hbase/)

Apache Cassandra (https://sparkbyexamples.com/category/cassandra/)

Snowflake Database (https://sparkbyexamples.com/category/snowflake/)

H2O Sparkling Water (https://sparkbyexamples.com/category/h2o-sparkling-water/)

PySpark (https://sparkbyexamples.com/category/pyspark/)

Spark isin() & IS NOT IN Operator Example (https://sparkbyexamples.com/spark/spark-isin-is-not-in-operator-example/)

Spark – Get Size/Length of Array & Map Column (https://sparkbyexamples.com/spark/spark-get-size-length-of-array-map-column/)

Spark Using Length/Size Of a DataFrame Column (https://sparkbyexamples.com/spark/spark-using-length-size-of-a-dataframe-column/)

Spark rlike() Working with Regex Matching Examples (https://sparkbyexamples.com/spark/spark-rlike-regex-matching-examples/)

Spark Check String Column Has Numeric Values (https://sparkbyexamples.com/spark/spark-check-string-column-has-numeric-values/)

Spark Check Column Data Type is Integer or String (https://sparkbyexamples.com/spark/spark-check-column-data-type-is-integer-or-string/)

(https://twitter.com/sparkbyexamples)
(https://www.facebook.com/sparkbyexamples/)
(https://www.linkedin.com/in/n-nk-b860a8193/)
(https://github.com/spark-examples/)