```
Microsoft Windows [Version 10.0.22631.3447]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
setLogLevel(newLevel).
24/04/19 16:15:16 WARN NativeCodeLoader: Unable to load native-hadoop
library for your platform... using builtin-java classes where applicable
Spark context Web UI available at http://Siddharth:4040
Spark context available as 'sc' (master = local[*], app id = local-
1713523517726).
Spark session available as 'spark'.
Welcome to

      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.5.1
      /_/

Using Scala version 2.12.18 (OpenJDK 64-Bit Server VM, Java 11.0.22)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val numbersDF = Seq(
     |       (1, "Alex", 50.0),
     |       (2, "Bob", 20.0),
     |       (3, "Cara", 30.0),
     |       (4, "Devin", 45.0),
     |       (5, "Euro", 75.0)
     |     ).toDF("Sr", "Name", "Marks")
numbersDF: org.apache.spark.sql.DataFrame = [Sr: int, Name: string ... 1
more field]

scala> println("Original DataFrame:")
Original DataFrame:

scala>     numbersDF.show()
+---+-----+-----+
| Sr| Name|Marks|
+---+-----+-----+
|  1| Alex| 50.0|
|  2|  Bob| 20.0|
|  3| Cara| 30.0|
|  4|Devin| 45.0|
|  5| Euro| 75.0|
+---+-----+-----+


scala>     val filteredDF = numbersDF.filter($"Marks" > 20)
filteredDF: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Sr:
int, Name: string ... 1 more field]

scala>     println("Filtered DataFrame:")
```

```
Filtered DataFrame:

scala>     filteredDF.show()
+---+-----+-----+
| Sr| Name|Marks|
+---+-----+-----+
|  1| Alex| 50.0|
|  3| Cara| 30.0|
|  4|Devin| 45.0|
|  5| Euro| 75.0|
+---+-----+-----+

scala> val sumDF =
numbersDF.groupBy("Name").agg(sum("Marks").alias("total_number"),
sum("Sr").alias("total_value"))
sumDF: org.apache.spark.sql.DataFrame = [Name: string, total_number:
double ... 1 more field]

scala> println("Sum of numbers and values for each letter:")
Sum of numbers and values for each letter:

scala>     sumDF.show()
+-----+------------+-----------+
| Name|total_number|total_value|
+-----+------------+-----------+
| Alex|        50.0|          1|
|  Bob|        20.0|          2|
| Cara|        30.0|          3|
|Devin|        45.0|          4|
| Euro|        75.0|          5|
+-----+------------+-----------+


scala> val avgValue = numbersDF.agg(avg("Marks")).first().getDouble(0)
avgValue: Double = 44.0

scala> println("Average value: " + avgValue)
Average value: 44.0

scala> val stdDevValue =
numbersDF.agg(stddev("Marks")).first().getDouble(0)
stdDevValue: Double = 21.03568396796263

scala> println("Standard deviation of value: " + stdDevValue)
Standard deviation of value: 21.03568396796263

scala>  val sortedDF = numbersDF.sort($"Sr".desc)
sortedDF: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Sr:
int, Name: string ... 1 more field]

scala>  println("Sorted DataFrame:")
Sorted DataFrame:

scala>  sortedDF.show()
```

```
+---+-----+-----+
| Sr| Name|Marks|
+---+-----+-----+
|  5| Euro| 75.0|
|  4|Devin| 45.0|
|  3| Cara| 30.0|
|  2|  Bob| 20.0|
|  1| Alex| 50.0|
+---+-----+-----+


scala> val minValue = numbersDF.agg(min("Marks")).first().getDouble(0)
minValue: Double = 20.0

scala> println("Minimum value: " + minValue)
Minimum value: 20.0

scala> val maxValue = numbersDF.agg(max("Marks")).first().getDouble(0)
maxValue: Double = 75.0

scala> println("Maximum value: " + maxValue)
Maximum value: 75.0
```