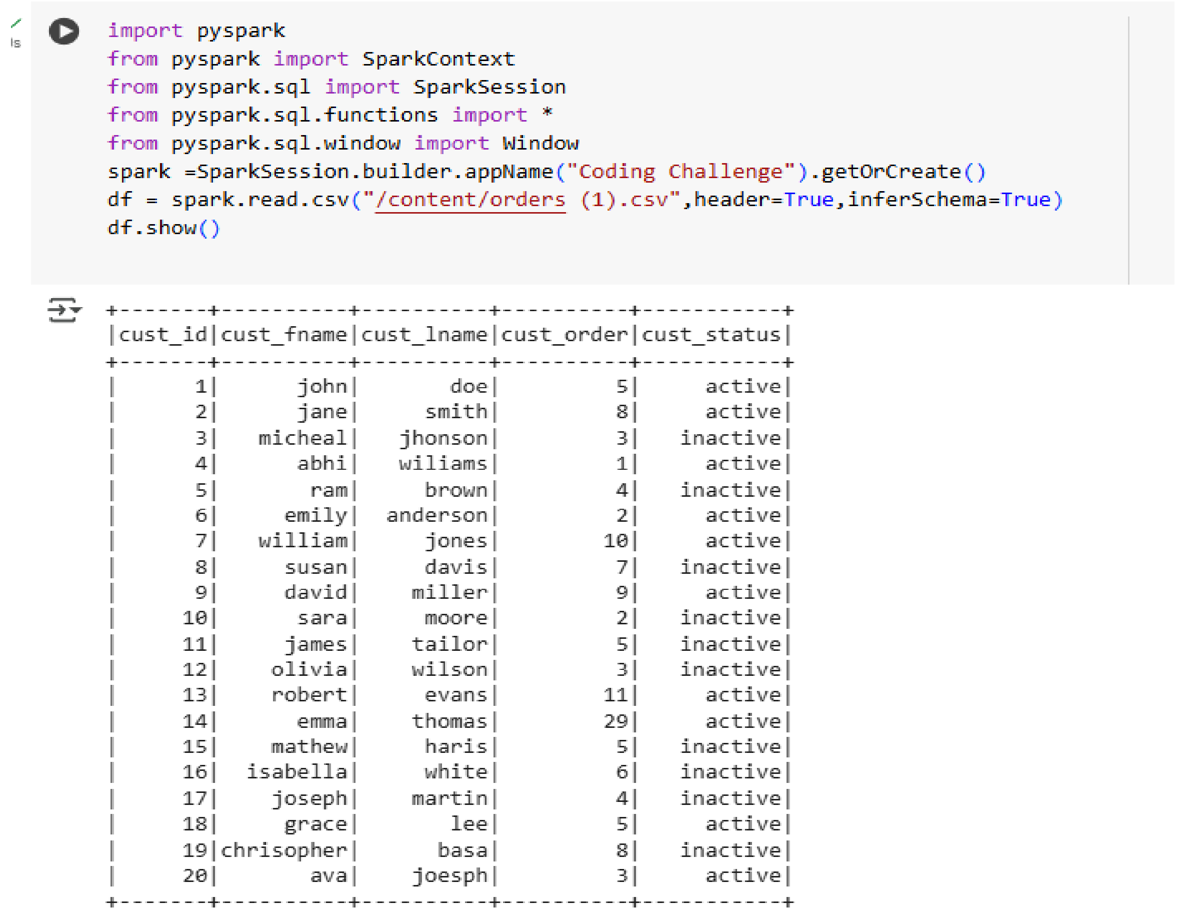
# Pyspark and Spark SQL Coding Challenge

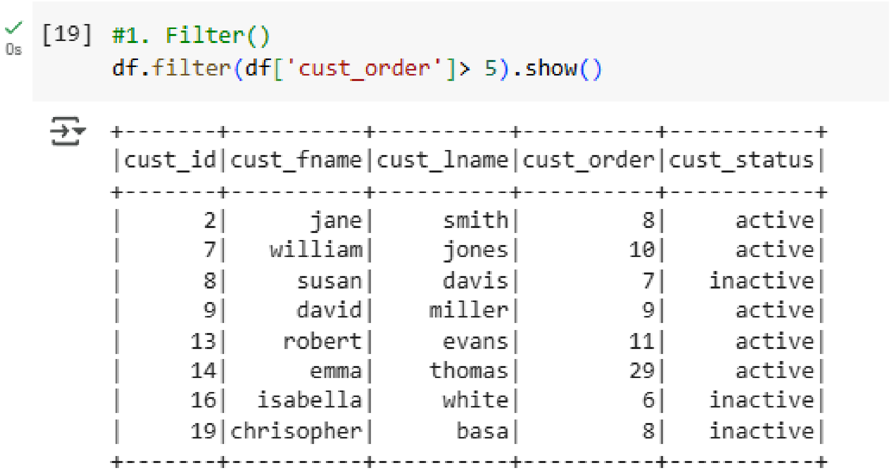
## Applying Transformations  in pyspark:



```
import pyspark
from pyspark import SparkContext
from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.window import Window
spark =SparkSession.builder.appName("Coding Challenge").getOrCreate()
df = spark.read.csv("/content/orders (1).csv",header=True,inferSchema=True)
df.show()
```

```
+-------+----------+----------+----------+-----------+
|cust_id|cust_fname|cust_lname|cust_order|cust_status|
+-------+----------+----------+----------+-----------+
|      1|      john|       doe|         5|     active|
|      2|      jane|     smith|         8|     active|
|      3|   micheal|   jhonson|         3|   inactive|
|      4|      abhi|   wiliams|         1|     active|
|      5|       ram|     brown|         4|   inactive|
|      6|     emily|  anderson|         2|     active|
|      7|   william|     jones|        10|     active|
|      8|     susan|     davis|         7|   inactive|
|      9|     david|    miller|         9|     active|
|     10|      sara|     moore|         2|   inactive|
|     11|     james|    tailor|         5|   inactive|
|     12|    olivia|    wilson|         3|   inactive|
|     13|    robert|     evans|        11|     active|
|     14|      emma|    thomas|        29|     active|
|     15|    mathew|     haris|         5|   inactive|
|     16|  isabella|     white|         6|   inactive|
|     17|    joseph|    martin|         4|   inactive|
|     18|     grace|       lee|         5|     active|
|     19|chrisopher|      basa|         8|   inactive|
|     20|       ava|    joesph|         3|     active|
+-------+----------+----------+----------+-----------+
```

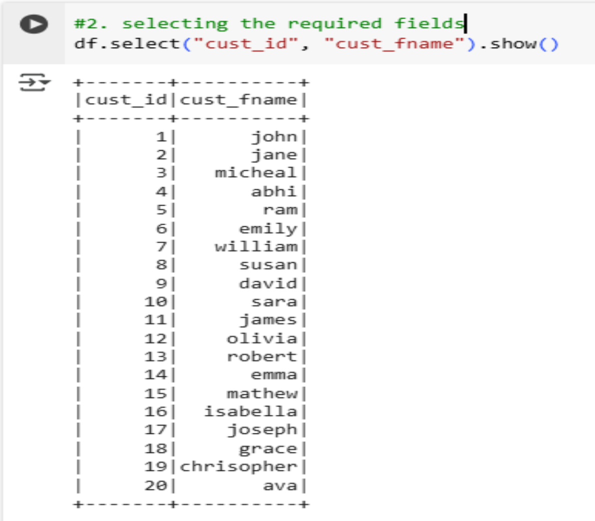
1. **Filter()**



```
[19] #1. Filter()
     df.filter(df['cust_order']> 5).show()
```

```
+-------+----------+----------+----------+-----------+
|cust_id|cust_fname|cust_lname|cust_order|cust_status|
+-------+----------+----------+----------+-----------+
|      2|      jane|     smith|         8|     active|
|      7|   william|     jones|        10|     active|
|      8|     susan|     davis|         7|   inactive|
|      9|     david|    miller|         9|     active|
|     13|    robert|     evans|        11|     active|
|     14|      emma|    thomas|        29|     active|
|     16|  isabella|     white|         6|   inactive|
|     19|chrisopher|      basa|         8|   inactive|
+-------+----------+----------+----------+-----------+
```

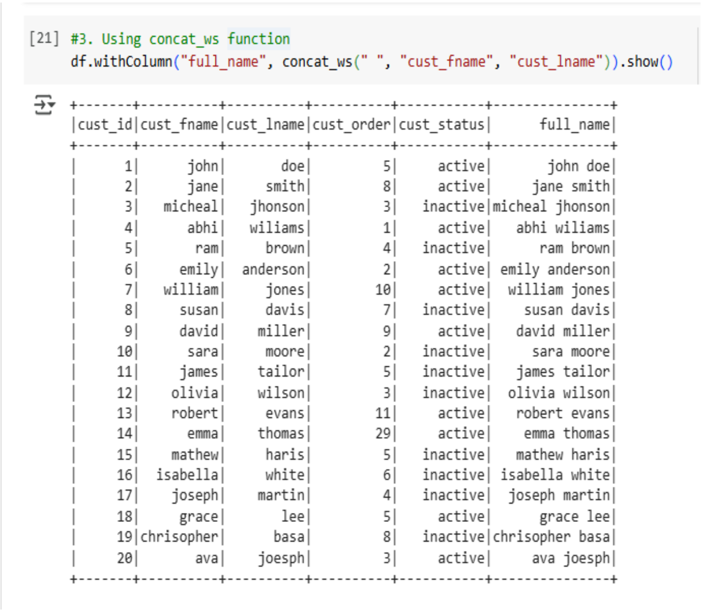
## 2. Select()



```
#2. selecting the required fields
df.select("cust_id", "cust_fname").show()
```

```
+-------+----------+
|cust_id|cust_fname|
+-------+----------+
|      1|      john|
|      2|      jane|
|      3|   micheal|
|      4|      abhi|
|      5|       ram|
|      6|     emily|
|      7|   william|
|      8|     susan|
|      9|     david|
|     10|      sara|
|     11|     james|
|     12|    olivia|
|     13|    robert|
|     14|      emma|
|     15|    mathew|
|     16|  isabella|
|     17|    joseph|
|     18|     grace|
|     19|chrisopher|
|     20|       ava|
+-------+----------+
```

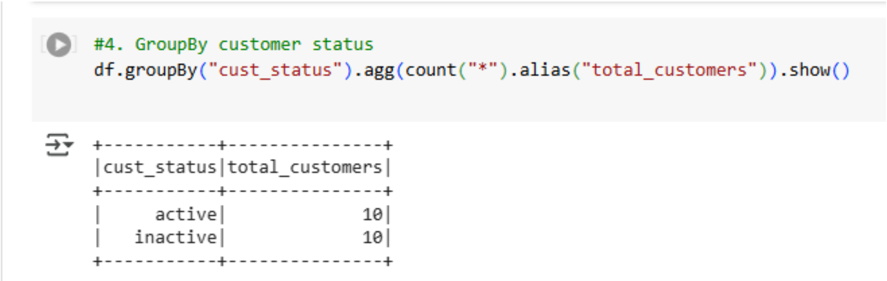
## 3. Using sql functions for joining 2 columns and adding a new column



```
[21] #3. Using concat_ws function
     df.withColumn("full_name", concat_ws(" ", "cust_fname", "cust_lname")).show()
```

```
+-------+----------+----------+----------+-----------+---------------+
|cust_id|cust_fname|cust_lname|cust_order|cust_status|      full_name|
+-------+----------+----------+----------+-----------+---------------+
|      1|      john|       doe|         5|     active|       john doe|
|      2|      jane|     smith|         8|     active|     jane smith|
|      3|   micheal|   jhonson|         3|   inactive|micheal jhonson|
|      4|      abhi|   wiliams|         1|     active|   abhi wiliams|
|      5|       ram|     brown|         4|   inactive|      ram brown|
|      6|     emily|  anderson|         2|     active| emily anderson|
|      7|   william|     jones|        10|     active|  william jones|
|      8|     susan|     davis|         7|   inactive|    susan davis|
|      9|     david|    miller|         9|     active|   david miller|
|     10|      sara|     moore|         2|   inactive|     sara moore|
|     11|     james|    tailor|         5|   inactive|   james tailor|
|     12|    olivia|    wilson|         3|   inactive|  olivia wilson|
|     13|    robert|     evans|        11|     active|   robert evans|
|     14|      emma|    thomas|        29|     active|    emma thomas|
|     15|    mathew|     haris|         5|   inactive|   mathew haris|
|     16|  isabella|     white|         6|   inactive| isabella white|
|     17|    joseph|    martin|         4|   inactive|  joseph martin|
|     18|     grace|       lee|         5|     active|      grace lee|
|     19|chrisopher|      basa|         8|   inactive|chrisopher basa|
|     20|       ava|    joesph|         3|     active|     ava joesph|
+-------+----------+----------+----------+-----------+---------------+
```

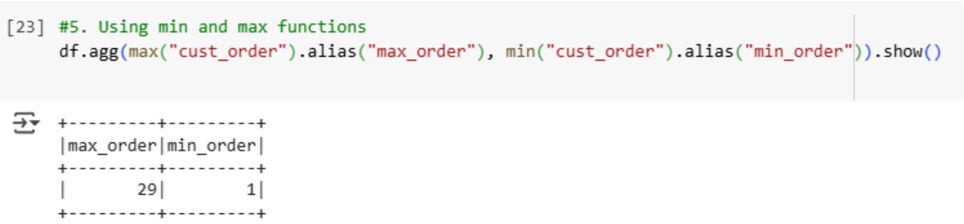
## 4. Group By()



```
#4. GroupBy customer status
df.groupBy("cust_status").agg(count("*").alias("total_customers")).show()
```

```
+-----------+---------------+
|cust_status|total_customers|
+-----------+---------------+
|     active|             10|
|   inactive|             10|
+-----------+---------------+
```

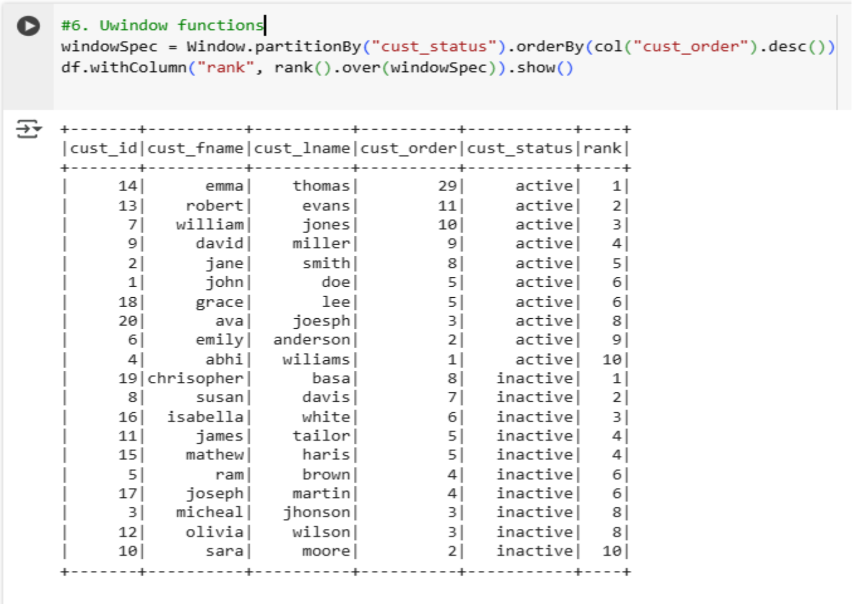
## 5. Simple Aggregations like Min and Max functions



```
[23] #5. Using min and max functions
     df.agg(max("cust_order").alias("max_order"), min("cust_order").alias("min_order")).show()
```

```
+---------+---------+
|max_order|min_order|
+---------+---------+
|       29|        1|
+---------+---------+
```

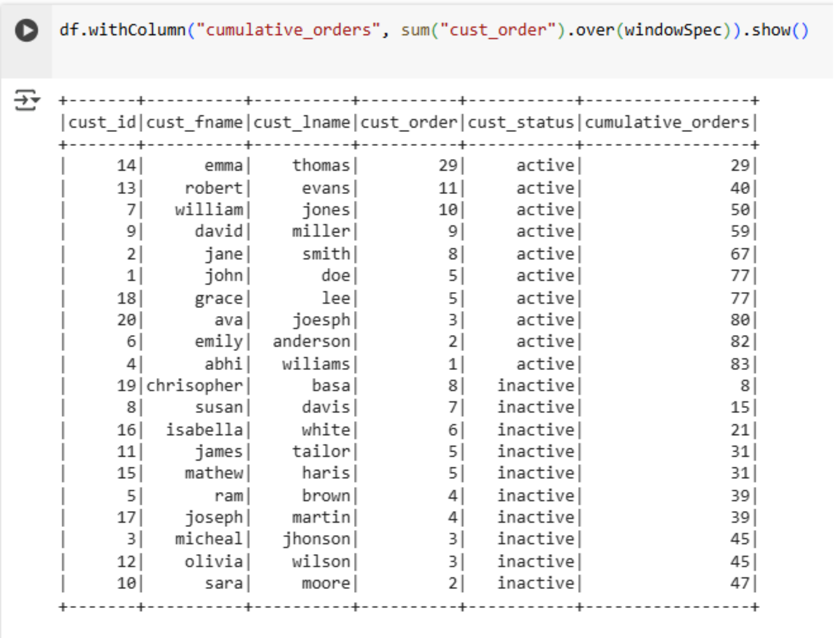
## 6. Window Function like rank()



```
#6. Uwindow functions
windowSpec = Window.partitionBy("cust_status").orderBy(col("cust_order").desc())
df.withColumn("rank", rank().over(windowSpec)).show()
```

```
+-------+----------+----------+----------+-----------+----+
|cust_id|cust_fname|cust_lname|cust_order|cust_status|rank|
+-------+----------+----------+----------+-----------+----+
|     14|      emma|    thomas|        29|     active|   1|
|     13|    robert|     evans|        11|     active|   2|
|      7|   william|     jones|        10|     active|   3|
|      9|     david|    miller|         9|     active|   4|
|      2|      jane|     smith|         8|     active|   5|
|      1|      john|       doe|         5|     active|   6|
|     18|     grace|       lee|         5|     active|   6|
|     20|       ava|    joesph|         3|     active|   8|
|      6|     emily|  anderson|         2|     active|   9|
|      4|      abhi|   wiliams|         1|     active|  10|
|     19|chrisopher|      basa|         8|   inactive|   1|
|      8|     susan|     davis|         7|   inactive|   2|
|     16|  isabella|     white|         6|   inactive|   3|
|     11|     james|    tailor|         5|   inactive|   4|
|     15|    mathew|     haris|         5|   inactive|   4|
|      5|       ram|     brown|         4|   inactive|   6|
|     17|    joseph|    martin|         4|   inactive|   6|
|      3|   micheal|   jhonson|         3|   inactive|   8|
|     12|    olivia|    wilson|         3|   inactive|   8|
|     10|      sara|     moore|         2|   inactive|  10|
+-------+----------+----------+----------+-----------+----+
```

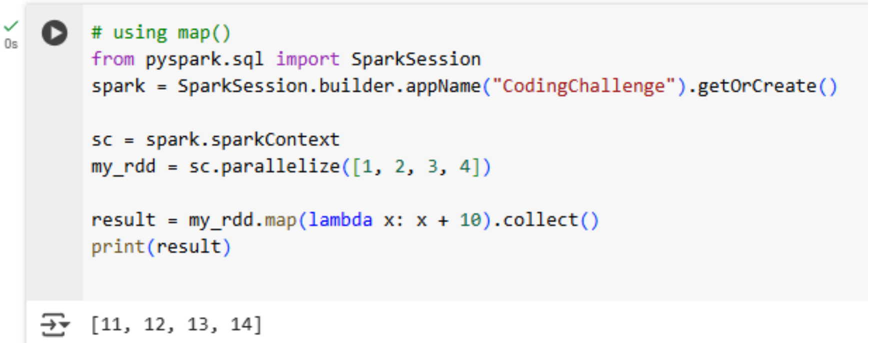
## 7. Sum() in window function



```
df.withColumn("cumulative_orders", sum("cust_order").over(windowSpec)).show()
```

```
+-------+----------+----------+----------+-----------+-----------------+
|cust_id|cust_fname|cust_lname|cust_order|cust_status|cumulative_orders|
+-------+----------+----------+----------+-----------+-----------------+
|     14|      emma|    thomas|        29|     active|               29|
|     13|    robert|     evans|        11|     active|               40|
|      7|   william|     jones|        10|     active|               50|
|      9|     david|    miller|         9|     active|               59|
|      2|      jane|     smith|         8|     active|               67|
|      1|      john|       doe|         5|     active|               77|
|     18|     grace|       lee|         5|     active|               77|
|     20|       ava|    joesph|         3|     active|               80|
|      6|     emily|  anderson|         2|     active|               82|
|      4|      abhi|   wiliams|         1|     active|               83|
|     19|chrisopher|      basa|         8|   inactive|                8|
|      8|     susan|     davis|         7|   inactive|               15|
|     16|  isabella|     white|         6|   inactive|               21|
|     11|     james|    tailor|         5|   inactive|               31|
|     15|    mathew|     haris|         5|   inactive|               31|
|      5|       ram|     brown|         4|   inactive|               39|
|     17|    joseph|    martin|         4|   inactive|               39|
|      3|   micheal|   jhonson|         3|   inactive|               45|
|     12|    olivia|    wilson|         3|   inactive|               45|
|     10|      sara|     moore|         2|   inactive|               47|
+-------+----------+----------+----------+-----------+-----------------+
```

### 8. Using Map() in rdd



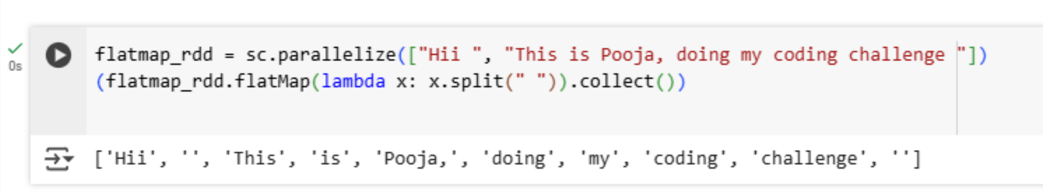
```
# using map()
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("CodingChallenge").getOrCreate()

sc = spark.sparkContext
my_rdd = sc.parallelize([1, 2, 3, 4])

result = my_rdd.map(lambda x: x + 10).collect()
print(result)
```

```
[11, 12, 13, 14]
```

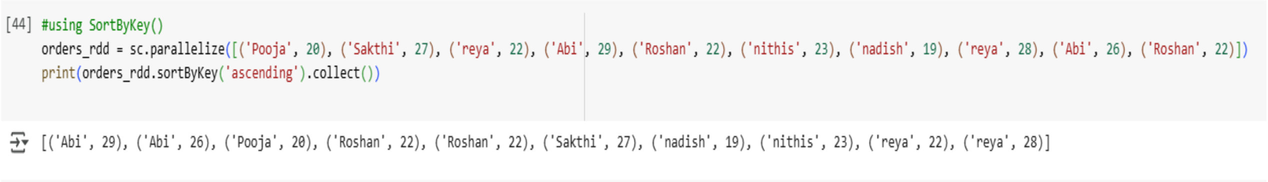
### 9. Using flatMap()



```
flatmap_rdd = sc.parallelize(["Hii ", "This is Pooja, doing my coding challenge "])
(flatmap_rdd.flatMap(lambda x: x.split(" ")).collect())
```

```
['Hii', '', 'This', 'is', 'Pooja,', 'doing', 'my', 'coding', 'challenge', '']
```

### 10. Using sortByKey()

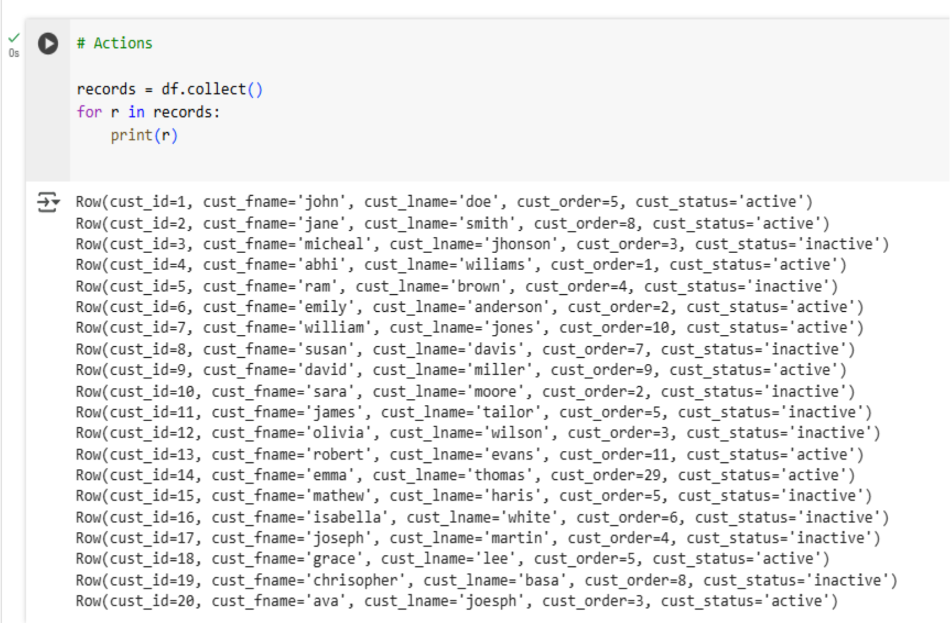


```
[44] #using SortByKey()
     orders_rdd = sc.parallelize([('Pooja', 20), ('Sakthi', 27), ('reya', 22), ('Abi', 29), ('Roshan', 22), ('nithis', 23), ('nadish', 19), ('reya', 28), ('Abi', 26), ('Roshan', 22)])
     print(orders_rdd.sortByKey('ascending').collect())
```

```
[('Abi', 29), ('Abi', 26), ('Pooja', 20), ('Roshan', 22), ('Roshan', 22), ('Sakthi', 27), ('nadish', 19), ('nithis', 23), ('reya', 22), ('reya', 28)]
```

## Applying Actions in pyspark:

### 1. Using collect()

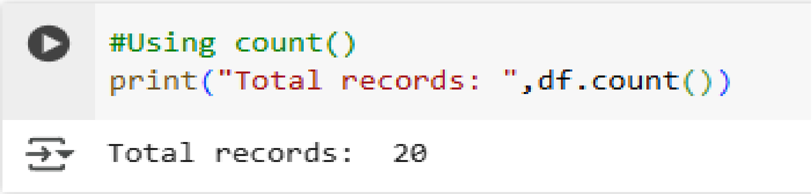


```
# Actions

records = df.collect()
for r in records:
    print(r)
```

```
Row(cust_id=1, cust_fname='john', cust_lname='doe', cust_order=5, cust_status='active')
Row(cust_id=2, cust_fname='jane', cust_lname='smith', cust_order=8, cust_status='active')
Row(cust_id=3, cust_fname='micheal', cust_lname='jhonson', cust_order=3, cust_status='inactive')
Row(cust_id=4, cust_fname='abhi', cust_lname='wiliams', cust_order=1, cust_status='active')
Row(cust_id=5, cust_fname='ram', cust_lname='brown', cust_order=4, cust_status='inactive')
Row(cust_id=6, cust_fname='emily', cust_lname='anderson', cust_order=2, cust_status='active')
Row(cust_id=7, cust_fname='william', cust_lname='jones', cust_order=10, cust_status='active')
Row(cust_id=8, cust_fname='susan', cust_lname='davis', cust_order=7, cust_status='inactive')
Row(cust_id=9, cust_fname='david', cust_lname='miller', cust_order=9, cust_status='active')
Row(cust_id=10, cust_fname='sara', cust_lname='moore', cust_order=2, cust_status='inactive')
Row(cust_id=11, cust_fname='james', cust_lname='tailor', cust_order=5, cust_status='inactive')
Row(cust_id=12, cust_fname='olivia', cust_lname='wilson', cust_order=3, cust_status='inactive')
Row(cust_id=13, cust_fname='robert', cust_lname='evans', cust_order=11, cust_status='active')
Row(cust_id=14, cust_fname='emma', cust_lname='thomas', cust_order=29, cust_status='active')
Row(cust_id=15, cust_fname='mathew', cust_lname='haris', cust_order=5, cust_status='inactive')
Row(cust_id=16, cust_fname='isabella', cust_lname='white', cust_order=6, cust_status='inactive')
Row(cust_id=17, cust_fname='joseph', cust_lname='martin', cust_order=4, cust_status='inactive')
Row(cust_id=18, cust_fname='grace', cust_lname='lee', cust_order=5, cust_status='active')
Row(cust_id=19, cust_fname='chrisopher', cust_lname='basa', cust_order=8, cust_status='inactive')
Row(cust_id=20, cust_fname='ava', cust_lname='joesph', cust_order=3, cust_status='active')
```

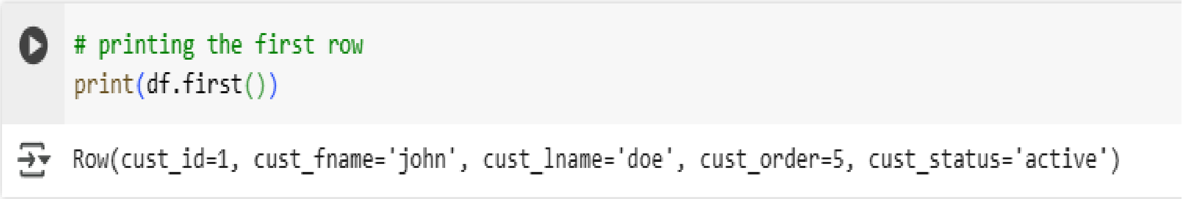
## 2. Using count()



```
#Using count()
print("Total records: ",df.count())
```

```
Total records:  20
```

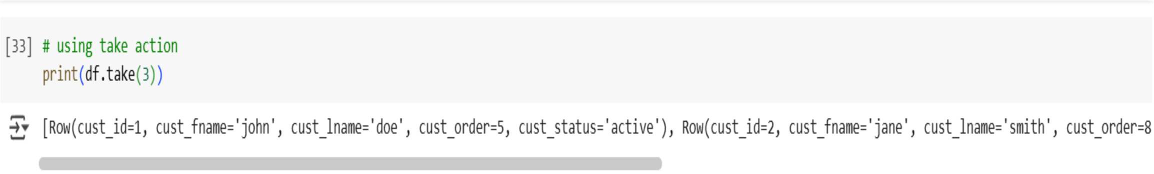
## 3. Using first()



```
# printing the first row
print(df.first())
```

```
Row(cust_id=1, cust_fname='john', cust_lname='doe', cust_order=5, cust_status='active')
```

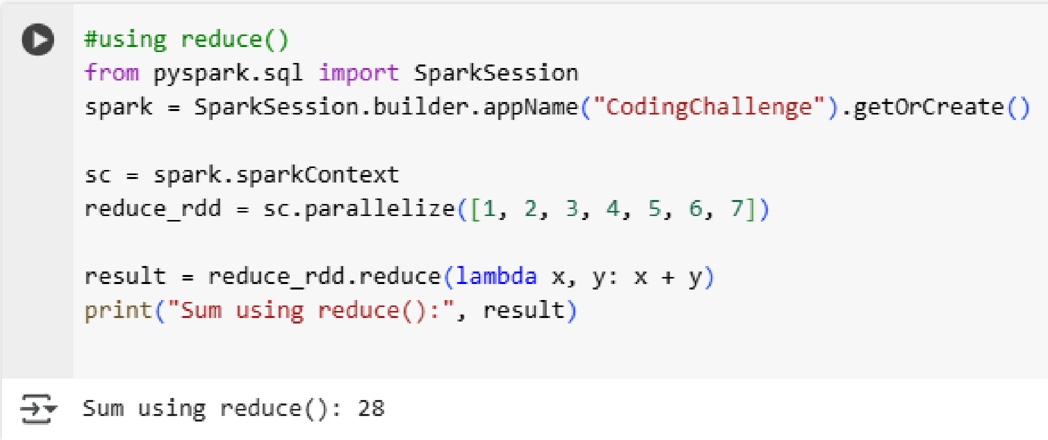
## 4. Using take()



```
[33] # using take action
     print(df.take(3))
```

```
[Row(cust_id=1, cust_fname='john', cust_lname='doe', cust_order=5, cust_status='active'), Row(cust_id=2, cust_fname='jane', cust_lname='smith', cust_order=8
```

## 5. Using reduce()



```
#using reduce()
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("CodingChallenge").getOrCreate()

sc = spark.sparkContext
reduce_rdd = sc.parallelize([1, 2, 3, 4, 5, 6, 7])

result = reduce_rdd.reduce(lambda x, y: x + y)
print("Sum using reduce():", result)
```

```
Sum using reduce(): 28
```