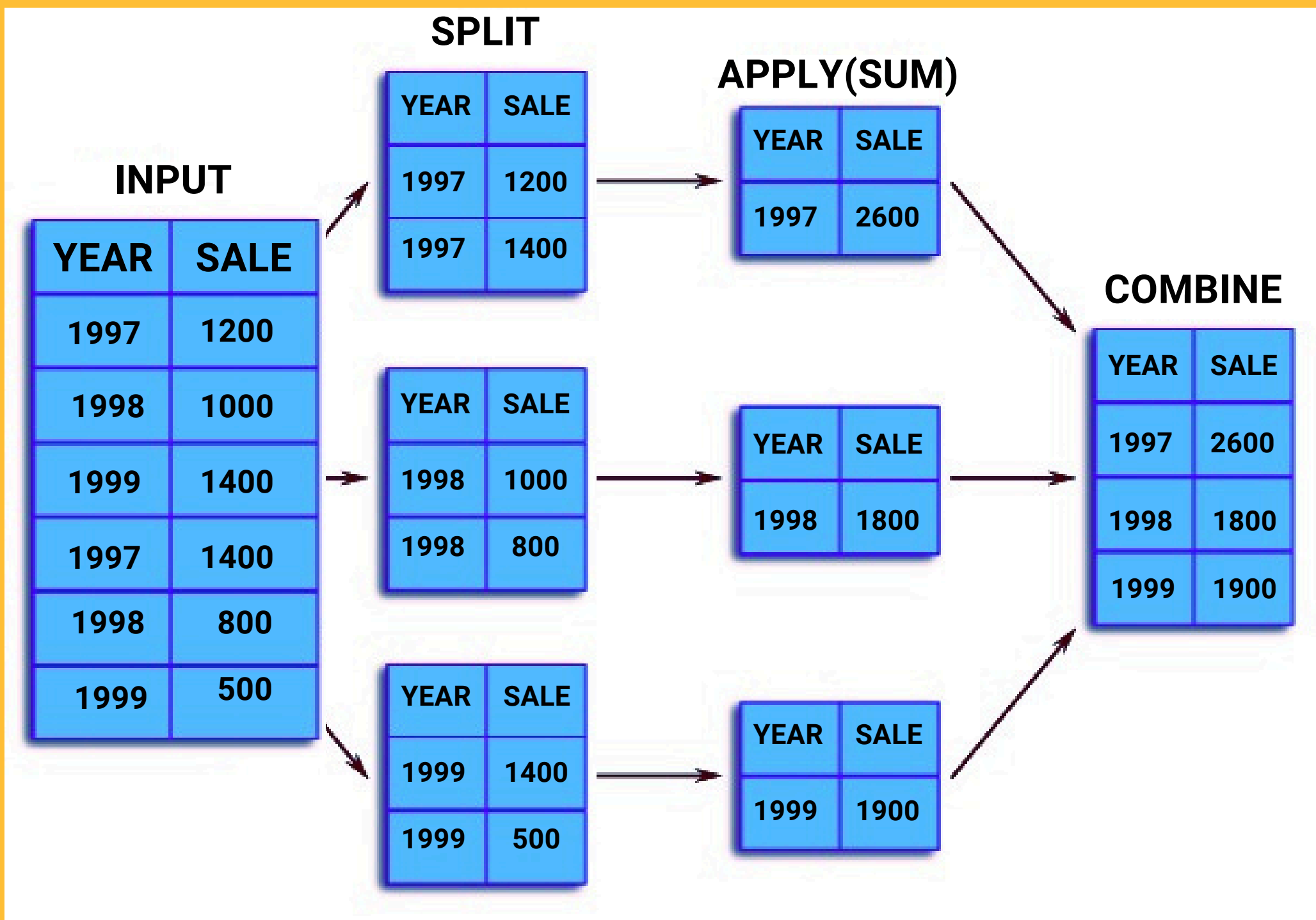
GROUPBY IN PANDAS vs PYSPARK

groupby()

DataFrame


INPUT
YEAR
SALE
1997
1200
1998
1000
1999
1400
1997
1400
1998
800
1999
500
SPLIT
YEAR
SALE
1997
1200
1997
1400
YEAR
SALE
1998
1000
1998
800
YEAR
SALE
1999
1400
1999
500
APPLY(SUM)
YEAR
SALE
1997
2600
YEAR
SALE
1998
1800
YEAR
SALE
1999
1900
COMBINE
YEAR
SALE
1997
2600
1998
1800
1999
1900

WHAT IS GROUPBY FUNCTION?

- Pandas GroupBy is a powerful and versatile function in Python. It allows you to split your
- data into separate groups to perform computations for better analysis.

```
from pyspark.sql import SparkSession
# Create Spark session
spark = SparkSession.builder.appName("GroupByExample").getOrCreate()
# Sample data
data = [ (1997, 1200), (1998, 1000), (1999, 1400),
(1997, 1400), (1998, 800), (1999, 500) ]
# Define schema (column names)
columns = ["YEAR", "SALE"]
# Create DataFrame
df = spark.createDataFrame(data, columns)
df.display()
```

	YEAR	SALE
0	1997	1200
1	1998	1000
2	1999	1400
3	1997	1400
4	1998	800
5	1999	500

```
# importing the essential library
import pandas as pd

# Creating a dummy DataFrame
df = pd.DataFrame({'YEAR':[1997,1998,1999,1997,1998,1999],
                   'SALE':[1200,1000,1400,1400,800,500]})

print(df)
```

	YEAR	SALE
0	1997	1200
1	1998	1000
2	1999	1400
3	1997	1400
4	1998	800
5	1999	500

Let’s group the dataset based on the YEAR using GroupBy:

```
df.groupby(by = 'YEAR')
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f57130babe0>
```

- GroupBy has conveniently returned a **DataFrameGroupBy** object.
- It has split the data into separate groups.
- However, it **won’t do anything** unless it is being told explicitly to do so.
- So, let’s find the count of different YEAR

Let’s group the dataset based on the YEAR using GroupBy:

```
df.groupby(by = 'YEAR')
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f57130babe0>
```

- GroupBy has conveniently returned a **DataFrameGroupBy** object.
- It has split the data into separate groups.
- However, it **won’t do anything** unless it is being told explicitly to do so.
- So, let’s find the count of different YEAR

```
df.groupby(by = 'YEAR').count()
```

YEAR	SALE
1997	2
1998	2
1999	2

We did not tell GroupBy which column we wanted it to apply the aggregation function on, so it applied it to all the relevant columns and returned the output.

GroupBy object supports column indexing just like a DataFrame! So let's find out the total sales for each location type:

```
df.groupby(by = 'YEAR').sum()
```

YEAR	SALE
1997	2600
1998	1800
1999	1900

GroupBy object supports column indexing just like a DataFrame! So let’s find out the total sales for each location type:

```
df_grouped =df.groupBy("YEAR").agg(sum("SALE").
alias("TOTAL_SALES"))
```

Display the results using .display().

```
df_grouped.display()
```

YEAR	SALE
1997	2600
1998	1800
1999	1900

Thanks for reading !

Follow my profile for more coding related contents

Like Comment Share

@rganesh0203