# 1. RDD Operations ¶

#### Import findspark to use Spark in jupyter notebook

In [1]:
```
import findspark
findspark.init()

from pyspark import SparkContext
```

#### Initialize SparkCOntext

In [2]:
```
sc=SparkContext.getOrCreate()
```

#### Create Resilient Distributed Datasets (RDDs)

In [3]:
```
data=sc.parallelize([['France','50M','3T'],['India','30M','30T'],['Kenya','70M','25T'],
                     ['Nigeria','90M','60T'],['China','20M','2T'],['USA','80M','30T'],
                     ['UK','70M','25T'],['USA','20M','30T'],['China','70M','25T'],
                     ['France', '50M', '3T'],['China','70M','25T'] ])
data2=sc.parallelize([23,42,78,90,12,9,42,89,42,12,50,60])
a=sc.parallelize([('USA',35),('Canada',24),('Mexico',27),('Kenya',23)])
b=sc.parallelize([('Kenya',30),('USA',35),('South Africa',23),('Rwanda',23)])
print('data\n',data,'\ndata2\n',data2,'\na\n',a,'\nb\n',b)
```

```
data
 ParallelCollectionRDD[0] at readRDDFromFile at PythonRDD.scala:262
data2
 ParallelCollectionRDD[1] at readRDDFromFile at PythonRDD.scala:262
a
 ParallelCollectionRDD[2] at readRDDFromFile at PythonRDD.scala:262
b
 ParallelCollectionRDD[3] at readRDDFromFile at PythonRDD.scala:262
```

**Show data from RDD**

In [4]:
```
data.collect()
```

Out[4]:
```
[['France', '50M', '3T'],
 ['India', '30M', '30T'],
 ['Kenya', '70M', '25T'],
 ['Nigeria', '90M', '60T'],
 ['China', '20M', '2T'],
 ['USA', '80M', '30T'],
 ['UK', '70M', '25T'],
 ['USA', '20M', '30T'],
 ['China', '70M', '25T'],
 ['France', '50M', '3T'],
 ['China', '70M', '25T']]
```

# i. Actions

Sparks action functions produce a value back to the Spark driver program

Check the RDD Persistence

```
In [5]: data.persist().is_cached
```

```
Out[5]: True
```

Get number of partitions

```
In [6]: data.getNumPartitions()
```

```
Out[6]: 8
```

Show all data

```
In [7]: data.collect()
```

```
Out[7]: [['France', '50M', '3T'],
         ['India', '30M', '30T'],
         ['Kenya', '70M', '25T'],
         ['Nigeria', '90M', '60T'],
         ['China', '20M', '2T'],
         ['USA', '80M', '30T'],
         ['UK', '70M', '25T'],
         ['USA', '20M', '30T'],
         ['China', '70M', '25T'],
         ['France', '50M', '3T'],
         ['China', '70M', '25T']]
```

Get number of items

```
In [8]: len(data.collect())
Out[8]: 11
```

Get distinct Records

```
In [9]: sorted(data2.collect())
Out[9]: [9, 12, 12, 23, 42, 42, 42, 50, 60, 78, 89, 90]
```

```
In [10]: sorted(data2.distinct().collect())
Out[10]: [9, 12, 23, 42, 50, 60, 78, 89, 90]
```

Get max value from the list

```
In [11]: data2.max()
Out[11]: 90
```

Get min value from the list

```
In [12]: data2.min()
Out[12]: 9
```

Get sum value from the list

```
In [13]: data2.sum()
Out[13]: 549
```

Get average

```
In [14]: data2.mean()
```

```
Out[14]: 45.75
```

Get variance value from the list

```
In [15]: data2.variance()
```

```
Out[15]: 773.1875000000001
```

Get standard deviation value from the list

```
In [16]: data2.stdev()
```

```
Out[16]: 27.80624929759496
```

Get mean, std deviation, max and min values

```
In [17]: data2.stats()
```

```
Out[17]: (count: 12, mean: 45.75, stdev: 27.80624929759496, max: 90.0, min: 9.0)
```

Show first column

```
In [18]: data.first()
```

```
Out[18]: ['France', '50M', '3T']
```

Get random record

```
In [19]: data.takeSample(1,True)
```

```
Out[19]: [['Nigeria', '90M', '60T']]
```

Count records

```
In [20]: data.count()
```

```
Out[20]: 11
```

Count occurence of the items

```
In [21]: data.countByKey().items()
```

```
Out[21]: dict_items([('France', 2), ('India', 1), ('Kenya', 1), ('Nigeria', 1), ('China', 3), ('USA', 2), ('UK', 1)])
```

# ii. Sparks Transformation Functions

Sparks transformation functions produce a new Resilient Distributed Dataset (RDD)

```
In [22]: tran_data=sc.parallelize([4,6,8,2,2,6])
         tran_data.collect()
```

```
Out[22]: [4, 6, 8, 2, 2, 6]
```

Select two random items

```
In [23]: tran_data.sample(2,True).collect()
```

```
Out[23]: [4, 4, 4, 2, 2, 6, 6]
```

Use map to multiply each item with 2

```
In [24]: tran_data.map(lambda x : x*2).collect()
Out[24]: [8, 12, 16, 4, 4, 12]
```

Use filterMap to duplicate the items

```
In [25]: tran_data.flatMap(lambda x : [x,x]).collect()
Out[25]: [4, 4, 6, 6, 8, 8, 2, 2, 2, 2, 6, 6]
```

Filter from data where item is China

```
In [26]: data.filter(lambda x : "China" in x).collect()
Out[26]: [['China', '20M', '2T'], ['China', '70M', '25T'], ['China', '70M', '25T']]
```

Return unique items

```
In [27]: tran_data.distinct().collect()
Out[27]: [8, 2, 4, 6]
```

Sorting

In [28]:
```
data.sortByKey(1, True).collect()
```

Out[28]:
```
[['China', '20M', '2T'],
 ['China', '70M', '25T'],
 ['China', '70M', '25T'],
 ['France', '50M', '3T'],
 ['France', '50M', '3T'],
 ['India', '30M', '30T'],
 ['Kenya', '70M', '25T'],
 ['Nigeria', '90M', '60T'],
 ['UK', '70M', '25T'],
 ['USA', '80M', '30T'],
 ['USA', '20M', '30T']]
```

Join two RDDs

In [29]:
```
print("a : ",a.collect())
print("b : ",b.collect())
```

```
a :  [('USA', 35), ('Canada', 24), ('Mexico', 27), ('Kenya', 23)]
b :  [('Kenya', 30), ('USA', 35), ('South Africa', 23), ('Rwanda', 23)]
```

In [30]:
```
a.join(b).collect()
```

Out[30]:
```
[('USA', (35, 35)), ('Kenya', (23, 30))]
```

Left outer join

In [31]:
```
a.leftOuterJoin(b).collect()
```

Out[31]:
```
[('USA', (35, 35)),
 ('Mexico', (27, None)),
 ('Canada', (24, None)),
 ('Kenya', (23, 30))]
```

Right outer join

```
In [32]: a.rightOuterJoin(b).collect()
```

```
Out[32]: [('USA', (35, 35)),
          ('South Africa', (None, 23)),
          ('Kenya', (23, 30)),
          ('Rwanda', (None, 23))]
```

Union

```
In [33]: a.union(b).collect()
```

```
Out[33]: [('USA', 35),
          ('Canada', 24),
          ('Mexico', 27),
          ('Kenya', 23),
          ('Kenya', 30),
          ('USA', 35),
          ('South Africa', 23),
          ('Rwanda', 23)]
```

Difference

```
In [34]: a.subtract(b).collect()
```

```
Out[34]: [('Kenya', 23), ('Canada', 24), ('Mexico', 27)]
```

Intersection

```
In [35]: a.intersection(b).collect()
```

```
Out[35]: [('USA', 35)]
```

Cartesian

In [36]:
```
a.cartesian(b).collect()
```

Out[36]:
```
[(('USA', 35), ('Kenya', 30)),
 (('USA', 35), ('USA', 35)),
 (('USA', 35), ('South Africa', 23)),
 (('USA', 35), ('Rwanda', 23)),
 (('Canada', 24), ('Kenya', 30)),
 (('Canada', 24), ('USA', 35)),
 (('Canada', 24), ('South Africa', 23)),
 (('Canada', 24), ('Rwanda', 23)),
 (('Mexico', 27), ('Kenya', 30)),
 (('Mexico', 27), ('USA', 35)),
 (('Mexico', 27), ('South Africa', 23)),
 (('Mexico', 27), ('Rwanda', 23)),
 (('Kenya', 23), ('Kenya', 30)),
 (('Kenya', 23), ('USA', 35)),
 (('Kenya', 23), ('South Africa', 23)),
 (('Kenya', 23), ('Rwanda', 23))]
```

# 2. DataFrames in PySpark

***Creating Sparks DataFrames***

In [37]:
```
from pyspark.sql import SparkSession
import pandas as pd

spark = SparkSession.builder.getOrCreate()
```

Create Spark DataFrame from List

In [38]:
```
spark_df_from_list = spark.createDataFrame([['France','50M','3T'],['India','30M','30T'],['Kenya','70M','25T'],
                     ['Nigeria','90M','60T'],['China','20M','2T'],['USA','80M','30T'],
                     ['UK','70M','25T'],['USA','20M','30T'],['China','70M','25T'],
                     ['France', '50M', '3T'],['China','70M','25T'] ],
                          schema='Country string,Population string,GDP string')
spark_df_from_list.show()
```

```
+-------+----------+---+
|Country|Population|GDP|
+-------+----------+---+
| France|       50M| 3T|
|  India|       30M|30T|
|  Kenya|       70M|25T|
|Nigeria|       90M|60T|
|  China|       20M| 2T|
|    USA|       80M|30T|
|     UK|       70M|25T|
|    USA|       20M|30T|
|  China|       70M|25T|
| France|       50M| 3T|
|  China|       70M|25T|
+-------+----------+---+
```

Create Spark DataFrame from Pandas DataFrame

In [39]:
```
# Pandas DataFrame
pandas_df=pd.DataFrame([['France','50M','3T'],['India','30M','30T'],['Kenya','70M','25T'],
                        ['Nigeria','90M','60T'],['China','20M','2T'],['USA','80M','30T'],
                        ['UK','70M','25T'],['USA','20M','30T'],['China','70M','25T'],
                        ['France', '50M', '3T'],['China','70M','25T'] ],
                             columns=['Country','Population','GDP'])
pandas_df
```

Out[39]:

| | Country | Population | GDP |
|---|---|---|---|
| **0** | France | 50M | 3T |
| **1** | India | 30M | 30T |
| **2** | Kenya | 70M | 25T |
| **3** | Nigeria | 90M | 60T |
| **4** | China | 20M | 2T |
| **5** | USA | 80M | 30T |
| **6** | UK | 70M | 25T |
| **7** | USA | 20M | 30T |
| **8** | China | 70M | 25T |
| **9** | France | 50M | 3T |
| **10** | China | 70M | 25T |

In [40]:
```
# Sparks DataFrame
spark_df=spark.createDataFrame(pandas_df)
spark_df.show()
```

```
+-------+----------+---+
|Country|Population|GDP|
+-------+----------+---+
| France|       50M| 3T|
|  India|       30M|30T|
|  Kenya|       70M|25T|
|Nigeria|       90M|60T|
|  China|       20M| 2T|
|    USA|       80M|30T|
|     UK|       70M|25T|
|    USA|       20M|30T|
|  China|       70M|25T|
| France|       50M| 3T|
|  China|       70M|25T|
+-------+----------+---+
```

Create a PySpark DataFrame from an RDD

In [41]:
```
# RDD Data
rdd_data=sc.parallelize([['France','50M','3T'],['India','30M','30T'],['Kenya','70M','25T'],
                     ['Nigeria','90M','60T'],['China','20M','2T'],['USA','80M','30T'],
                     ['UK','70M','25T'],['USA','20M','30T'],['China','70M','25T'],
                     ['France', '50M', '3T'],['China','70M','25T'] ])
rdd_data.collect()
```

Out[41]:
```
[['France', '50M', '3T'],
 ['India', '30M', '30T'],
 ['Kenya', '70M', '25T'],
 ['Nigeria', '90M', '60T'],
 ['China', '20M', '2T'],
 ['USA', '80M', '30T'],
 ['UK', '70M', '25T'],
 ['USA', '20M', '30T'],
 ['China', '70M', '25T'],
 ['France', '50M', '3T'],
 ['China', '70M', '25T']]
```

In [42]:
```
# Spark DataFrame from RDD
spark_df_from_rdd=spark.createDataFrame(rdd_data, schema=['Country','Population','GDP'])
spark_df_from_rdd.show()
```

```
+-------+----------+---+
|Country|Population|GDP|
+-------+----------+---+
| France|       50M| 3T|
|  India|       30M|30T|
|  Kenya|       70M|25T|
|Nigeria|       90M|60T|
|  China|       20M| 2T|
|    USA|       80M|30T|
|     UK|       70M|25T|
|    USA|       20M|30T|
|  China|       70M|25T|
| France|       50M| 3T|
|  China|       70M|25T|
+-------+----------+---+
```

Display Spark DataFrame with Pandas DataFrame Jupyter Look and Feel

In [43]:
```
spark.conf.set('spark.sql.repl.eagerEval.enabled', True)
spark_df_from_rdd
```

Out[43]:

| Country | Population | GDP |
|---|---|---|
| France | 50M | 3T |
| India | 30M | 30T |
| Kenya | 70M | 25T |
| Nigeria | 90M | 60T |
| China | 20M | 2T |
| USA | 80M | 30T |
| UK | 70M | 25T |
| USA | 20M | 30T |
| China | 70M | 25T |
| France | 50M | 3T |
| China | 70M | 25T |

Import Data to Spark from CSV file

In [44]:
```
df=spark.read.csv("titanic.csv", inferSchema=True, header=True)
df.show()
```

```
+--------+------+--------------------+------+----+----------------------+----------------------+-------+
|Survived|Pclass|                Name|   Sex| Age|Siblings/Spouses Aboard|Parents/Children Aboard|   Fare|
+--------+------+--------------------+------+----+----------------------+----------------------+-------+
|       0|     3|Mr. Owen Harris B...|  male|22.0|                     1|                     0|   7.25|
|       1|     1|Mrs. John Bradley...|female|38.0|                     1|                     0|71.2833|
|       1|     3|Miss. Laina Heikk...|female|26.0|                     0|                     0|  7.925|
|       1|     1|Mrs. Jacques Heat...|female|35.0|                     1|                     0|   53.1|
|       0|     3|Mr. William Henry...|  male|35.0|                     0|                     0|   8.05|
|       0|     3|     Mr. James Moran|  male|27.0|                     0|                     0| 8.4583|
|       0|     1|Mr. Timothy J McC...|  male|54.0|                     0|                     0|51.8625|
|       0|     3|Master. Gosta Leo...|  male| 2.0|                     3|                     1| 21.075|
|       1|     3|Mrs. Oscar W (Eli...|female|27.0|                     0|                     2|11.1333|
|       1|     2|Mrs. Nicholas (Ad...|female|14.0|                     1|                     0|30.0708|
|       1|     3|Miss. Marguerite ...|female| 4.0|                     1|                     1|   16.7|
|       1|     1|Miss. Elizabeth B...|female|58.0|                     0|                     0|  26.55|
|       0|     3|Mr. William Henry...|  male|20.0|                     0|                     0|   8.05|
|       0|     3|Mr. Anders Johan ...|  male|39.0|                     1|                     5| 31.275|
|       0|     3|Miss. Hulda Amand...|female|14.0|                     0|                     0| 7.8542|
|       1|     2|Mrs. (Mary D King...|female|55.0|                     0|                     0|   16.0|
|       0|     3| Master. Eugene Rice|  male| 2.0|                     4|                     1| 29.125|
|       1|     2|Mr. Charles Eugen...|  male|23.0|                     0|                     0|   13.0|
|       0|     3|Mrs. Julius (Emel...|female|31.0|                     1|                     0|   18.0|
|       1|     3|Mrs. Fatima Masse...|female|22.0|                     0|                     0|  7.225|
+--------+------+--------------------+------+----+----------------------+----------------------+-------+
only showing top 20 rows
```

Show data schema

In [45]:
```
df.printSchema()
```

```
root
 |-- Survived: integer (nullable = true)
 |-- Pclass: integer (nullable = true)
 |-- Name: string (nullable = true)
 |-- Sex: string (nullable = true)
 |-- Age: double (nullable = true)
 |-- Siblings/Spouses Aboard: integer (nullable = true)
 |-- Parents/Children Aboard: integer (nullable = true)
 |-- Fare: double (nullable = true)
```

Show data types

In [46]:
```
df.dtypes
```

Out[46]:
```
[('Survived', 'int'),
 ('Pclass', 'int'),
 ('Name', 'string'),
 ('Sex', 'string'),
 ('Age', 'double'),
 ('Siblings/Spouses Aboard', 'int'),
 ('Parents/Children Aboard', 'int'),
 ('Fare', 'double')]
```

Show logical and physical structure of the DataFrame

In [47]:
```
df.explain()
```

```
== Physical Plan ==
FileScan csv [Survived#99,Pclass#100,Name#101,Sex#102,Age#103,Siblings/Spouses Aboard#104,Parents/Children Aboard#10
5,Fare#106] Batched: false, DataFilters: [], Format: CSV, Location: InMemoryFileIndex[file:/C:/Users/soongaya/Documen
ts/Data Science/data2ml/git/data2ml/titanic.csv], PartitionFilters: [], PushedFilters: [], ReadSchema: struct<Survive
d:int,Pclass:int,Name:string,Sex:string,Age:double,Siblings/Spouses Aboard:int,Pare...
```

show columns

```
In [48]: df.columns
```

```
Out[48]: ['Survived',
          'Pclass',
          'Name',
          'Sex',
          'Age',
          'Siblings/Spouses Aboard',
          'Parents/Children Aboard',
          'Fare']
```

Show top 5 records with head function

```
In [49]: df.head(5)
```

```
Out[49]: [Row(Survived=0, Pclass=3, Name='Mr. Owen Harris Braund', Sex='male', Age=22.0, Siblings/Spouses Aboard=1, Parents/Ch
         ildren Aboard=0, Fare=7.25),
          Row(Survived=1, Pclass=1, Name='Mrs. John Bradley (Florence Briggs Thayer) Cumings', Sex='female', Age=38.0, Sibling
         s/Spouses Aboard=1, Parents/Children Aboard=0, Fare=71.2833),
          Row(Survived=1, Pclass=3, Name='Miss. Laina Heikkinen', Sex='female', Age=26.0, Siblings/Spouses Aboard=0, Parents/C
         hildren Aboard=0, Fare=7.925),
          Row(Survived=1, Pclass=1, Name='Mrs. Jacques Heath (Lily May Peel) Futrelle', Sex='female', Age=35.0, Siblings/Spous
         es Aboard=1, Parents/Children Aboard=0, Fare=53.1),
          Row(Survived=0, Pclass=3, Name='Mr. William Henry Allen', Sex='male', Age=35.0, Siblings/Spouses Aboard=0, Parents/C
         hildren Aboard=0, Fare=8.05)]
```

Show top 5 records in a tabular format with show function

In [50]:
```
df.show(5)
```

```
+--------+------+--------------------+------+----+----------------------+----------------------+-------+
|Survived|Pclass|                Name|   Sex| Age|Siblings/Spouses Aboard|Parents/Children Aboard|   Fare|
+--------+------+--------------------+------+----+----------------------+----------------------+-------+
|       0|     3|Mr. Owen Harris B...|  male|22.0|                     1|                     0|   7.25|
|       1|     1|Mrs. John Bradley...|female|38.0|                     1|                     0|71.2833|
|       1|     3|Miss. Laina Heikk...|female|26.0|                     0|                     0|  7.925|
|       1|     1|Mrs. Jacques Heat...|female|35.0|                     1|                     0|   53.1|
|       0|     3|Mr. William Henry...|  male|35.0|                     0|                     0|   8.05|
+--------+------+--------------------+------+----+----------------------+----------------------+-------+
only showing top 5 rows
```

Show top 5 records with take function

In [51]:
```
df.take(5)
```

Out[51]:
```
[Row(Survived=0, Pclass=3, Name='Mr. Owen Harris Braund', Sex='male', Age=22.0, Siblings/Spouses Aboard=1, Parents/Ch
ildren Aboard=0, Fare=7.25),
 Row(Survived=1, Pclass=1, Name='Mrs. John Bradley (Florence Briggs Thayer) Cumings', Sex='female', Age=38.0, Sibling
s/Spouses Aboard=1, Parents/Children Aboard=0, Fare=71.2833),
 Row(Survived=1, Pclass=3, Name='Miss. Laina Heikkinen', Sex='female', Age=26.0, Siblings/Spouses Aboard=0, Parents/C
hildren Aboard=0, Fare=7.925),
 Row(Survived=1, Pclass=1, Name='Mrs. Jacques Heath (Lily May Peel) Futrelle', Sex='female', Age=35.0, Siblings/Spous
es Aboard=1, Parents/Children Aboard=0, Fare=53.1),
 Row(Survived=0, Pclass=3, Name='Mr. William Henry Allen', Sex='male', Age=35.0, Siblings/Spouses Aboard=0, Parents/C
hildren Aboard=0, Fare=8.05)]
```

Show data with three features/columns only

In [52]:
```
df.select('name','age','survived').show(5)
```

```
+--------------------+----+--------+
|                name| age|survived|
+--------------------+----+--------+
|Mr. Owen Harris B...|22.0|       0|
|Mrs. John Bradley...|38.0|       1|
|Miss. Laina Heikk...|26.0|       1|
|Mrs. Jacques Heat...|35.0|       1|
|Mr. William Henry...|35.0|       0|
+--------------------+----+--------+
only showing top 5 rows
```

Show distinct records

In [53]:
```
df.select('sex').distinct().show()
```

```
+------+
|   sex|
+------+
|female|
|  male|
+------+
```

Count distinct records

In [54]:
```
df.select('sex').distinct().count()
```

Out[54]: 2

Aggregate data with groupby

In [55]:
```
df.select('sex').groupby('sex').count().show()
```

```
+------+-----+
|   sex|count|
+------+-----+
|female|  314|
|  male|  573|
+------+-----+
```

Show the number of columns

In [56]:
```
len(df.columns)
```

Out[56]: 8

Describe Dataframe for statistical analysis

In [57]:
```
df.select('Sex','Pclass','age','survived','Fare').describe()
```

Out[57]:

| summary | Sex | Pclass | age | survived | Fare |
|---|---|---|---|---|---|
| count | 887 | 887 | 887 | 887 | 887 |
| mean | null | 2.305524239007892 | 29.471443066516347 | 0.3855693348365276 | 32.30542018038328 |
| stddev | null | 0.8366620036697728 | 14.121908405462552 | 0.48700411775101266 | 49.78204040017391 |
| min | female | 1 | 0.42 | 0 | 0.0 |
| max | male | 3 | 80.0 | 1 | 512.3292 |

Describe single feature for statistical analysis

In [58]:
```
df.describe('sex').show()
```

```
+-------+------+
|summary|   sex|
+-------+------+
|  count|   887|
|   mean|  null|
| stddev|  null|
|    min|female|
|    max|  male|
+-------+------+
```

Adding a new column

In [59]:
```
df=df.withColumn('new_fare',df['fare']+100) # adding 100 on the base fare to get new fare
df.select('name','pclass','fare','new_fare').show(5)
```

```
+--------------------+------+-------+--------+
|                name|pclass|   fare|new_fare|
+--------------------+------+-------+--------+
|Mr. Owen Harris B...|     3|   7.25|  107.25|
|Mrs. John Bradley...|     1|71.2833|171.2833|
|Miss. Laina Heikk...|     3|  7.925| 107.925|
|Mrs. Jacques Heat...|     1|   53.1|   153.1|
|Mr. William Henry...|     3|   8.05|  108.05|
+--------------------+------+-------+--------+
only showing top 5 rows
```

Rename a column

In [60]:
```
df=df.withColumnRenamed('pclass','Passenger_Class') # Rename pclass to Passenger_Class
df.columns
```

Out[60]:
```
['Survived',
 'Passenger_Class',
 'Name',
 'Sex',
 'Age',
 'Siblings/Spouses Aboard',
 'Parents/Children Aboard',
 'Fare',
 'new_fare']
```

Aggregate with groupby sum in a specific column

In [61]:
```
df.groupby('sex').sum().show()
```

```
+------+------------+-------------------+-----------------+--------------------------+------------------------
--+----------------+----------------+
|   sex|sum(Survived)|sum(Passenger_Class)|         sum(Age)|sum(Siblings/Spouses Aboard)|sum(Parents/Children Aboar
d)|        sum(Fare)|    sum(new_fare)|
+------+------------+-------------------+-----------------+--------------------------+------------------------
--+----------------+----------------+
|female|         233|                678|           8704.0|                       218|                       2
04|13966.66279999999|45366.66280000006|
|  male|         109|               1367|17437.170000000002|                       248|                       1
36|14688.24489999999|71988.24490000005|
+------+------------+-------------------+-----------------+--------------------------+------------------------
--+----------------+----------------+
```

Aggregate with groupby sum in a specific column

In [62]:
```
df.select('sex','fare','Passenger_Class','Survived').groupby('sex').sum().show()
```

```
+------+-----------------+-------------------+-------------+
|   sex|        sum(fare)|sum(Passenger_Class)|sum(Survived)|
+------+-----------------+-------------------+-------------+
|female|13966.66279999999|                678|          233|
|  male|14688.24489999999|               1367|          109|
+------+-----------------+-------------------+-------------+
```

Pivot

In [63]:
```
df.select('sex','Passenger_Class').groupby('sex').pivot('Passenger_Class').sum().show()
```

```
+------+---+---+----+
|   sex|  1|  2|   3|
+------+---+---+----+
|female| 94|152| 432|
|  male|122|216|1029|
+------+---+---+----+
```

Drop columns

In [64]:
```
# Lets new_fare and home.dest columns
df=df.drop('new_fare','home.dest')
df.columns
```

Out[64]:
```
['Survived',
 'Passenger_Class',
 'Name',
 'Sex',
 'Age',
 'Siblings/Spouses Aboard',
 'Parents/Children Aboard',
 'Fare']
```

Show null values for in fare column

In [65]:
```
df.select('fare').where('fare is null').show()
```

```
+----+
|fare|
+----+
+----+
```

Drop null values

In [66]:
```
drop_null_df=df.dropna()
drop_null_df.select('fare').where('fare is null').show()
```

```
+----+
|fare|
+----+
+----+
```

Replace null with specific value

In [67]:
```
replace_null_df=df.fillna(0.00) # Replace fare null values with 0.00
replace_null_df.filter(replace_null_df['fare']==0.00).select('name','fare').show()
```

```
+--------------------+----+
|                name|fare|
+--------------------+----+
|  Mr. Lionel Leonard| 0.0|
|Mr. William Harrison| 0.0|
|Mr. William Henry...| 0.0|
|  Mr. Francis Parkes| 0.0|
|Mr. William Cahoo...| 0.0|
|Mr. Alfred Flemin...| 0.0|
|Mr. William Campbell| 0.0|
|Mr. Anthony Wood ...| 0.0|
|  Mr. Alfred Johnson| 0.0|
|Mr. William Henry...| 0.0|
|Mr. Ennis Hasting...| 0.0|
| Mr. Robert J Knight| 0.0|
|Mr. Thomas Jr And...| 0.0|
|     Mr. Richard Fry| 0.0|
|Jonkheer. John Ge...| 0.0|
+--------------------+----+
```

Select data with "like"

In [68]:
```
df.select("name",df['name'].like("Allen")).show(5)
```

```
+--------------------+---------------+
|                name|name LIKE Allen|
+--------------------+---------------+
|Mr. Owen Harris B...|          false|
|Mrs. John Bradley...|          false|
|Miss. Laina Heikk...|          false|
|Mrs. Jacques Heat...|          false|
|Mr. William Henry...|          false|
+--------------------+---------------+
only showing top 5 rows
```

Select data using "startswith"

```
In [69]: df.select("name",df['name'].startswith("All")).show(5)
```

```
+------------------+--------------------+
|              name|startswith(name, All)|
+------------------+--------------------+
|Mr. Owen Harris B...|               false|
|Mrs. John Bradley...|               false|
|Miss. Laina Heikk...|               false|
|Mrs. Jacques Heat...|               false|
|Mr. William Henry...|               false|
+------------------+--------------------+
only showing top 5 rows
```

Select data using "endswith"

```
In [70]: df.select('name',df['name'].endswith('n')).show(5)
```

```
+------------------+----------------+
|              name|endswith(name, n)|
+------------------+----------------+
|Mr. Owen Harris B...|           false|
|Mrs. John Bradley...|           false|
|Miss. Laina Heikk...|            true|
|Mrs. Jacques Heat...|           false|
|Mr. William Henry...|            true|
+------------------+----------------+
only showing top 5 rows
```

Select data using "between"

In [71]:
```
df.select('age',df['age'].between(30,50)).show(10)
```

```
+----+-----------------------------+
| age|((age >= 30) AND (age <= 50))|
+----+-----------------------------+
|22.0|                        false|
|38.0|                         true|
|26.0|                        false|
|35.0|                         true|
|35.0|                         true|
|27.0|                        false|
|54.0|                        false|
| 2.0|                        false|
|27.0|                        false|
|14.0|                        false|
+----+-----------------------------+
only showing top 10 rows
```

Select data using "contains"

In [72]:
```
df.select('name',df['name'].contains('All')).show(5)
```

```
+--------------------+------------------+
|                name|contains(name, All)|
+--------------------+------------------+
|Mr. Owen Harris B...|             false|
|Mrs. John Bradley...|             false|
|Miss. Laina Heikk...|             false|
|Mrs. Jacques Heat...|             false|
|Mr. William Henry...|              true|
+--------------------+------------------+
only showing top 5 rows
```

Select data using "substr"

In [73]:
```
df.select('name',df['name'].substr(0,5)).show(10)
```

```
+--------------------+---------------------+
|                name|substring(name, 0, 5)|
+--------------------+---------------------+
|Mr. Owen Harris B...|                Mr. O|
|Mrs. John Bradley...|                Mrs. |
|Miss. Laina Heikk...|                Miss.|
|Mrs. Jacques Heat...|                Mrs. |
|Mr. William Henry...|                Mr. W|
|     Mr. James Moran|                Mr. J|
|Mr. Timothy J McC...|                Mr. T|
|Master. Gosta Leo...|                Maste|
|Mrs. Oscar W (Eli...|                Mrs. |
|Mrs. Nicholas (Ad...|                Mrs. |
+--------------------+---------------------+
only showing top 10 rows
```

Select data using substring and creating a new alias

In [74]:
```
df.select('name',df['name'].substr(0,5).alias('First 5 String Characters')).show(10)
```

```
+--------------------+------------------------+
|                name|First 5 String Characters|
+--------------------+------------------------+
|Mr. Owen Harris B...|                    Mr. O|
|Mrs. John Bradley...|                    Mrs. |
|Miss. Laina Heikk...|                    Miss.|
|Mrs. Jacques Heat...|                    Mrs. |
|Mr. William Henry...|                    Mr. W|
|     Mr. James Moran|                    Mr. J|
|Mr. Timothy J McC...|                    Mr. T|
|Master. Gosta Leo...|                    Maste|
|Mrs. Oscar W (Eli...|                    Mrs. |
|Mrs. Nicholas (Ad...|                    Mrs. |
+--------------------+------------------------+
only showing top 10 rows
```

Select data using conditional operators in filter

```
In [75]: df.filter((df['age']>=40) & (df['age']<=50)).select('name','age').show(5)
```

```
+--------------------+----+
|                name| age|
+--------------------+----+
|Don. Manuel E Uru...|40.0|
|Mrs. William Augu...|48.0|
|Mr. Alexander Osk...|42.0|
|Mrs. Johan (Johan...|40.0|
|Mrs. Henry Sleepe...|49.0|
+--------------------+----+
only showing top 5 rows
```

Select data using "startswith" in filter

```
In [76]: df.filter(df['name'].startswith("Don")).select('name').show()
```

```
+--------------------+
|                name|
+--------------------+
|Don. Manuel E Uru...|
+--------------------+
```

Select data using "contains" in filter

In [77]:
```
df.filter(df['name'].contains('Master')).select('name').show(5)
```

```
+--------------------+
|                name|
+--------------------+
|Master. Gosta Leo...|
| Master. Eugene Rice|
|Master. Juha Niil...|
|Master. William F...|
|Master. Harald Skoog|
+--------------------+
only showing top 5 rows
```

# 3. Spark SQL with PySpark

Read data

In [78]:
```
data=spark.read.csv("titanic.csv", inferSchema=True, header=True)
data.show(5)
```

```
+--------+------+--------------------+------+----+----------------------+----------------------+-------+
|Survived|Pclass|                Name|   Sex| Age|Siblings/Spouses Aboard|Parents/Children Aboard|   Fare|
+--------+------+--------------------+------+----+----------------------+----------------------+-------+
|       0|     3|Mr. Owen Harris B...|  male|22.0|                     1|                     0|   7.25|
|       1|     1|Mrs. John Bradley...|female|38.0|                     1|                     0|71.2833|
|       1|     3|Miss. Laina Heikk...|female|26.0|                     0|                     0|  7.925|
|       1|     1|Mrs. Jacques Heat...|female|35.0|                     1|                     0|   53.1|
|       0|     3|Mr. William Henry...|  male|35.0|                     0|                     0|   8.05|
+--------+------+--------------------+------+----+----------------------+----------------------+-------+
only showing top 5 rows
```

Show Data Structure

In [79]:
```
data.printSchema()
```

```
root
 |-- Survived: integer (nullable = true)
 |-- Pclass: integer (nullable = true)
 |-- Name: string (nullable = true)
 |-- Sex: string (nullable = true)
 |-- Age: double (nullable = true)
 |-- Siblings/Spouses Aboard: integer (nullable = true)
 |-- Parents/Children Aboard: integer (nullable = true)
 |-- Fare: double (nullable = true)
```

Register a DataFrame as an SQL Temporary View

In [80]:
```
data.createOrReplaceTempView('titanic')
```

Select 5 records

In [81]:
```
spark.sql("SELECT * FROM titanic limit 5").show()
```

```
+--------+------+--------------------+------+----+-----------------------+-----------------------+-------+
|Survived|Pclass|                Name|   Sex| Age|Siblings/Spouses Aboard|Parents/Children Aboard|   Fare|
+--------+------+--------------------+------+----+-----------------------+-----------------------+-------+
|       0|     3|Mr. Owen Harris B...|  male|22.0|                      1|                      0|   7.25|
|       1|     1|Mrs. John Bradley...|female|38.0|                      1|                      0|71.2833|
|       1|     3|Miss. Laina Heikk...|female|26.0|                      0|                      0|  7.925|
|       1|     1|Mrs. Jacques Heat...|female|35.0|                      1|                      0|   53.1|
|       0|     3|Mr. William Henry...|  male|35.0|                      0|                      0|   8.05|
+--------+------+--------------------+------+----+-----------------------+-----------------------+-------+
```

Get number of Passengers in Each Class

In [82]:
```
spark.sql("SELECT pclass AS Passenger_Class, COUNT(*) AS Number_of_Passengers FROM titanic GROUP BY pclass").show()
```

```
+---------------+--------------------+
|Passenger_Class|Number_of_Passengers|
+---------------+--------------------+
|              1|                 216|
|              3|                 487|
|              2|                 184|
+---------------+--------------------+
```

***SparkSQL Mathematical Functions***

Get First, Last, Min, Max, Mean and Total fare of Passenger

In [83]:
```
spark.sql("SELECT FIRST(Fare) AS First_Record, LAST(Fare) AS Last_Record,MIN(Fare) AS Min_Fare, "+
          "MAX(Fare) AS Max_Fare,AVG(Fare) AS Average_fare, SUM(Fare) AS Total_Fare FROM titanic").show()
```

```
+------------+-----------+--------+--------+-----------------+------------------+
|First_Record|Last_Record|Min_Fare|Max_Fare|     Average_fare|        Total_Fare|
+------------+-----------+--------+--------+-----------------+------------------+
|        7.25|       7.75|     0.0|512.3292|32.30542018038328|28654.907699999967|
+------------+-----------+--------+--------+-----------------+------------------+
```

Get Standard Deviation of the Fare for passengers

In [84]:
```
spark.sql("SELECT stddev(Fare) AS Fare_Standard_Deviation FROM titanic").show()
```

```
+----------------------+
|Fare_Standard_Deviation|
+----------------------+
|      49.78204040017391|
+----------------------+
```

Get Variance of Fare distribution for the passengers

In [85]:
```
spark.sql("SELECT variance(Fare) AS Fare_Variance FROM titanic").show()
```

```
+------------------+
|     Fare_Variance|
+------------------+
|2478.2515464045478|
+------------------+
```

Get Skewness of Fare distribution for the passenger

In [86]:
```
spark.sql("SELECT skewness(Fare) FROM titanic").show()
```

```
+-----------------+
|   skewness(Fare)|
+-----------------+
|4.769588111295382|
+-----------------+
```

Get kurtosis of the Fare distribution for the passengers

In [87]:
```
spark.sql("SELECT kurtosis(Fare) AS Kurtosis FROM titanic").show()
```

```
+-----------------+
|         Kurtosis|
+-----------------+
|33.07063200139631|
+-----------------+
```

### *Analytical Functions*

In [88]:
```
from pyspark.sql.window import Window
from pyspark.sql.functions import row_number
```

In [89]:
```
df=data['pclass','name','survived','fare']
df
```

Out[89]:

| pclass | name | survived | fare |
|---|---|---|---|
| 3 | Mr. Owen Harris B... | 0 | 7.25 |
| 1 | Mrs. John Bradley... | 1 | 71.2833 |
| 3 | Miss. Laina Heikk... | 1 | 7.925 |
| 1 | Mrs. Jacques Heat... | 1 | 53.1 |
| 3 | Mr. William Henry... | 0 | 8.05 |
| 3 | Mr. James Moran | 0 | 8.4583 |
| 1 | Mr. Timothy J McC... | 0 | 51.8625 |
| 3 | Master. Gosta Leo... | 0 | 21.075 |
| 3 | Mrs. Oscar W (Eli... | 1 | 11.1333 |
| 2 | Mrs. Nicholas (Ad... | 1 | 30.0708 |
| 3 | Miss. Marguerite ... | 1 | 16.7 |
| 1 | Miss. Elizabeth B... | 1 | 26.55 |
| 3 | Mr. William Henry... | 0 | 8.05 |
| 3 | Mr. Anders Johan ... | 0 | 31.275 |
| 3 | Miss. Hulda Amand... | 0 | 7.8542 |
| 2 | Mrs. (Mary D King... | 1 | 16.0 |
| 3 | Master. Eugene Rice | 0 | 29.125 |
| 2 | Mr. Charles Eugen... | 1 | 13.0 |
| 3 | Mrs. Julius (Emel... | 0 | 18.0 |
| 3 | Mrs. Fatima Masse... | 1 | 7.225 |

only showing top 20 rows

row number Window Function.
This function returns sequential row number starting from 1 to the result of each window partition

In [90]:
```
win_function=Window.partitionBy("pclass").orderBy("Fare")
df.withColumn("Row_Number",row_number().over(win_function)).show(truncate=False)
```

```
+------+-----------------------------------------------+--------+-------+----------+
|pclass|name                                           |survived|fare   |Row_Number|
+------+-----------------------------------------------+--------+-------+----------+
|1     |Mr. William Harrison                           |0       |0.0    |1         |
|1     |Mr. William Henry Marsh Parr                   |0       |0.0    |2         |
|1     |Mr. Thomas Jr Andrews                          |0       |0.0    |3         |
|1     |Mr. Richard Fry                                |0       |0.0    |4         |
|1     |Jonkheer. John George Reuchlin                 |0       |0.0    |5         |
|1     |Mr. Frans Olof Carlsson                        |0       |5.0    |6         |
|1     |Mr. Edward Pomeroy Colley                      |0       |25.5875|7         |
|1     |Mr. John D Baumann                             |0       |25.925 |8         |
|1     |Dr. Alice (Farnham) Leader                     |1       |25.9292|9         |
|1     |Mrs. Frederick Joel (Margaret Welles Barron) Swift|1       |25.9292|10        |
|1     |Mr. Richard William Smith                      |0       |26.0   |11        |
|1     |Mr. Arthur Ernest Nicholson                    |0       |26.0   |12        |
|1     |Miss. Helen Monypeny Newsom                    |1       |26.2833|13        |
|1     |Mr. James Robert McGough                       |1       |26.2875|14        |
|1     |Mr. Spencer Victor Silverthorne                |1       |26.2875|15        |
|1     |Mr. Edward Pennington Calderhead               |1       |26.2875|16        |
|1     |Mr. John Irwin Flynn                           |1       |26.3875|17        |
|1     |Miss. Elizabeth Bonnell                        |1       |26.55  |18        |
|1     |Mr. Charles Hallace Romaine                    |1       |26.55  |19        |
|1     |Mr. William Thomas Stead                       |0       |26.55  |20        |
+------+-----------------------------------------------+--------+-------+----------+
only showing top 20 rows
```

rank() window function.
This function returns a rank to the result within a window partition while leaving gaps where there is duplicates

In [91]:
```
from pyspark.sql.functions import rank

df.withColumn("Rank", rank().over(win_function)).show(truncate=False)
```

```
+------+-----------------------------------------------+--------+-------+----+
|pclass|name                                           |survived|fare   |Rank|
+------+-----------------------------------------------+--------+-------+----+
|1     |Mr. William Harrison                           |0       |0.0    |1   |
|1     |Mr. William Henry Marsh Parr                   |0       |0.0    |1   |
|1     |Mr. Thomas Jr Andrews                          |0       |0.0    |1   |
|1     |Mr. Richard Fry                                |0       |0.0    |1   |
|1     |Jonkheer. John George Reuchlin                 |0       |0.0    |1   |
|1     |Mr. Frans Olof Carlsson                        |0       |5.0    |6   |
|1     |Mr. Edward Pomeroy Colley                      |0       |25.5875|7   |
|1     |Mr. John D Baumann                             |0       |25.925 |8   |
|1     |Dr. Alice (Farnham) Leader                     |1       |25.9292|9   |
|1     |Mrs. Frederick Joel (Margaret Welles Barron) Swift|1       |25.9292|9   |
|1     |Mr. Richard William Smith                      |0       |26.0   |11  |
|1     |Mr. Arthur Ernest Nicholson                    |0       |26.0   |11  |
|1     |Miss. Helen Monypeny Newsom                    |1       |26.2833|13  |
|1     |Mr. James Robert McGough                       |1       |26.2875|14  |
|1     |Mr. Spencer Victor Silverthorne                |1       |26.2875|14  |
|1     |Mr. Edward Pennington Calderhead               |1       |26.2875|14  |
|1     |Mr. John Irwin Flynn                           |1       |26.3875|17  |
|1     |Miss. Elizabeth Bonnell                        |1       |26.55  |18  |
|1     |Mr. Charles Hallace Romaine                    |1       |26.55  |18  |
|1     |Mr. William Thomas Stead                       |0       |26.55  |18  |
+------+-----------------------------------------------+--------+-------+----+
only showing top 20 rows
```

dense_rank() window function.
This function returns a rank to the result within a window partition without leaving gaps where there is duplicates

In [92]:
```
from pyspark.sql.functions import dense_rank

df.withColumn("Dense_Rank",dense_rank().over(win_function)).show(truncate=False)
```

```
+------+-----------------------------------------------+--------+-------+----------+
|pclass|name                                           |survived|fare   |Dense_Rank|
+------+-----------------------------------------------+--------+-------+----------+
|1     |Mr. William Harrison                           |0       |0.0    |1         |
|1     |Mr. William Henry Marsh Parr                   |0       |0.0    |1         |
|1     |Mr. Thomas Jr Andrews                          |0       |0.0    |1         |
|1     |Mr. Richard Fry                                |0       |0.0    |1         |
|1     |Jonkheer. John George Reuchlin                 |0       |0.0    |1         |
|1     |Mr. Frans Olof Carlsson                        |0       |5.0    |2         |
|1     |Mr. Edward Pomeroy Colley                      |0       |25.5875|3         |
|1     |Mr. John D Baumann                             |0       |25.925 |4         |
|1     |Dr. Alice (Farnham) Leader                     |1       |25.9292|5         |
|1     |Mrs. Frederick Joel (Margaret Welles Barron) Swift|1       |25.9292|5         |
|1     |Mr. Richard William Smith                      |0       |26.0   |6         |
|1     |Mr. Arthur Ernest Nicholson                    |0       |26.0   |6         |
|1     |Miss. Helen Monypeny Newsom                    |1       |26.2833|7         |
|1     |Mr. James Robert McGough                       |1       |26.2875|8         |
|1     |Mr. Spencer Victor Silverthorne                |1       |26.2875|8         |
|1     |Mr. Edward Pennington Calderhead               |1       |26.2875|8         |
|1     |Mr. John Irwin Flynn                           |1       |26.3875|9         |
|1     |Miss. Elizabeth Bonnell                        |1       |26.55  |10        |
|1     |Mr. Charles Hallace Romaine                    |1       |26.55  |10        |
|1     |Mr. William Thomas Stead                       |0       |26.55  |10        |
+------+-----------------------------------------------+--------+-------+----------+
only showing top 20 rows
```

percent_rank Window Function.
The PERCENT_RANK function computes the rank of the passenger's fare within a passenger class (pclass) as a percentage

In [93]:
```
from pyspark.sql.functions import percent_rank
df.withColumn("Percent_Rank",percent_rank().over(win_function)).show(truncate=False)
```

```
+------+-------------------------------------------------+--------+-------+--------------------+
|pclass|name                                             |survived|fare   |Percent_Rank        |
+------+-------------------------------------------------+--------+-------+--------------------+
|1     |Mr. William Harrison                             |0       |0.0    |0.0                 |
|1     |Mr. William Henry Marsh Parr                     |0       |0.0    |0.0                 |
|1     |Mr. Thomas Jr Andrews                            |0       |0.0    |0.0                 |
|1     |Mr. Richard Fry                                  |0       |0.0    |0.0                 |
|1     |Jonkheer. John George Reuchlin                   |0       |0.0    |0.0                 |
|1     |Mr. Frans Olof Carlsson                          |0       |5.0    |0.023255813953488372|
|1     |Mr. Edward Pomeroy Colley                        |0       |25.5875|0.027906976744186046|
|1     |Mr. John D Baumann                               |0       |25.925 |0.03255813953488372 |
|1     |Dr. Alice (Farnham) Leader                       |1       |25.9292|0.037209302325581395|
|1     |Mrs. Frederick Joel (Margaret Welles Barron) Swift|1       |25.9292|0.037209302325581395|
|1     |Mr. Richard William Smith                        |0       |26.0   |0.046511627906976744|
|1     |Mr. Arthur Ernest Nicholson                      |0       |26.0   |0.046511627906976744|
|1     |Miss. Helen Monypeny Newsom                      |1       |26.2833|0.05581395348837209 |
|1     |Mr. James Robert McGough                         |1       |26.2875|0.06046511627906977 |
|1     |Mr. Spencer Victor Silverthorne                  |1       |26.2875|0.06046511627906977 |
|1     |Mr. Edward Pennington Calderhead                 |1       |26.2875|0.06046511627906977 |
|1     |Mr. John Irwin Flynn                             |1       |26.3875|0.07441860465116279 |
|1     |Miss. Elizabeth Bonnell                          |1       |26.55  |0.07906976744186046 |
|1     |Mr. Charles Hallace Romaine                      |1       |26.55  |0.07906976744186046 |
|1     |Mr. William Thomas Stead                         |0       |26.55  |0.07906976744186046 |
+------+-------------------------------------------------+--------+-------+--------------------+
only showing top 20 rows
```

cume_dist Window Function.
This function returns the cumulative distribution of values within a window partition

In [94]:
```
from pyspark.sql.functions import cume_dist
df.withColumn("Cummulative_Dist",cume_dist().over(win_function)).show(truncate=False)
```

```
+------+-------------------------------------------------+--------+-------+--------------------+
|pclass|name                                             |survived|fare   |Cummulative_Dist    |
+------+-------------------------------------------------+--------+-------+--------------------+
|1     |Mr. William Harrison                             |0       |0.0    |0.023148148148148147|
|1     |Mr. William Henry Marsh Parr                     |0       |0.0    |0.023148148148148147|
|1     |Mr. Thomas Jr Andrews                            |0       |0.0    |0.023148148148148147|
|1     |Mr. Richard Fry                                  |0       |0.0    |0.023148148148148147|
|1     |Jonkheer. John George Reuchlin                   |0       |0.0    |0.023148148148148147|
|1     |Mr. Frans Olof Carlsson                          |0       |5.0    |0.027777777777777776|
|1     |Mr. Edward Pomeroy Colley                        |0       |25.5875|0.032407407407407406|
|1     |Mr. John D Baumann                               |0       |25.925 |0.037037037037037035|
|1     |Dr. Alice (Farnham) Leader                       |1       |25.9292|0.046296296296296294|
|1     |Mrs. Frederick Joel (Margaret Welles Barron) Swift|1       |25.9292|0.046296296296296294|
|1     |Mr. Richard William Smith                        |0       |26.0   |0.05555555555555555 |
|1     |Mr. Arthur Ernest Nicholson                      |0       |26.0   |0.05555555555555555 |
|1     |Miss. Helen Monypeny Newsom                      |1       |26.2833|0.06018518518518518 |
|1     |Mr. James Robert McGough                         |1       |26.2875|0.07407407407407407 |
|1     |Mr. Spencer Victor Silverthorne                  |1       |26.2875|0.07407407407407407 |
|1     |Mr. Edward Pennington Calderhead                 |1       |26.2875|0.07407407407407407 |
|1     |Mr. John Irwin Flynn                             |1       |26.3875|0.0787037037037037  |
|1     |Miss. Elizabeth Bonnell                          |1       |26.55  |0.14814814814814814 |
|1     |Mr. Charles Hallace Romaine                      |1       |26.55  |0.14814814814814814 |
|1     |Mr. William Thomas Stead                         |0       |26.55  |0.14814814814814814 |
+------+-------------------------------------------------+--------+-------+--------------------+
only showing top 20 rows
```

lag Window Function.
The lag function returns the previous value

In [95]:
```
from pyspark.sql.functions import lag

df.withColumn("Previous_Fare",lag("Fare",1).over(win_function)).show()
```

```
+------+--------------------+--------+-------+-------------+
|pclass|                name|survived|   fare|Previous_Fare|
+------+--------------------+--------+-------+-------------+
|     1|Mr. William Harrison|       0|    0.0|         null|
|     1|Mr. William Henry...|       0|    0.0|          0.0|
|     1|Mr. Thomas Jr And...|       0|    0.0|          0.0|
|     1|     Mr. Richard Fry|       0|    0.0|          0.0|
|     1|Jonkheer. John Ge...|       0|    0.0|          0.0|
|     1|Mr. Frans Olof Ca...|       0|    5.0|          0.0|
|     1|Mr. Edward Pomero...|       0|25.5875|          5.0|
|     1|  Mr. John D Baumann|       0| 25.925|      25.5875|
|     1|Dr. Alice (Farnha...|       1|25.9292|       25.925|
|     1|Mrs. Frederick Jo...|       1|25.9292|      25.9292|
|     1|Mr. Richard Willi...|       0|   26.0|      25.9292|
|     1|Mr. Arthur Ernest...|       0|   26.0|         26.0|
|     1|Miss. Helen Monyp...|       1|26.2833|         26.0|
|     1|Mr. James Robert ...|       1|26.2875|      26.2833|
|     1|Mr. Spencer Victo...|       1|26.2875|      26.2875|
|     1|Mr. Edward Pennin...|       1|26.2875|      26.2875|
|     1|Mr. John Irwin Flynn|       1|26.3875|      26.2875|
|     1|Miss. Elizabeth B...|       1|  26.55|      26.3875|
|     1|Mr. Charles Halla...|       1|  26.55|        26.55|
|     1|Mr. William Thoma...|       0|  26.55|        26.55|
+------+--------------------+--------+-------+-------------+
only showing top 20 rows
```

lead Window Function.
The lead function returns the next value

In [96]:
```
from pyspark.sql.functions import lead

df.withColumn("Next_Fare",lead("Fare",1).over(win_function)).show(truncate=False)
```

```
+------+-------------------------------------------------+--------+-------+---------+
|pclass|name                                             |survived|fare   |Next_Fare|
+------+-------------------------------------------------+--------+-------+---------+
|1     |Mr. William Harrison                             |0       |0.0    |0.0      |
|1     |Mr. William Henry Marsh Parr                     |0       |0.0    |0.0      |
|1     |Mr. Thomas Jr Andrews                            |0       |0.0    |0.0      |
|1     |Mr. Richard Fry                                  |0       |0.0    |0.0      |
|1     |Jonkheer. John George Reuchlin                   |0       |0.0    |5.0      |
|1     |Mr. Frans Olof Carlsson                          |0       |5.0    |25.5875  |
|1     |Mr. Edward Pomeroy Colley                        |0       |25.5875|25.925   |
|1     |Mr. John D Baumann                               |0       |25.925 |25.9292  |
|1     |Dr. Alice (Farnham) Leader                       |1       |25.9292|25.9292  |
|1     |Mrs. Frederick Joel (Margaret Welles Barron) Swift|1       |25.9292|26.0     |
|1     |Mr. Richard William Smith                        |0       |26.0   |26.0     |
|1     |Mr. Arthur Ernest Nicholson                      |0       |26.0   |26.2833  |
|1     |Miss. Helen Monypeny Newsom                      |1       |26.2833|26.2875  |
|1     |Mr. James Robert McGough                         |1       |26.2875|26.2875  |
|1     |Mr. Spencer Victor Silverthorne                  |1       |26.2875|26.2875  |
|1     |Mr. Edward Pennington Calderhead                 |1       |26.2875|26.3875  |
|1     |Mr. John Irwin Flynn                             |1       |26.3875|26.55    |
|1     |Miss. Elizabeth Bonnell                          |1       |26.55  |26.55    |
|1     |Mr. Charles Hallace Romaine                      |1       |26.55  |26.55    |
|1     |Mr. William Thomas Stead                         |0       |26.55  |26.55    |
+------+-------------------------------------------------+--------+-------+---------+
only showing top 20 rows
```