codes

April 7, 2025

[6]:
```
%pip install pandas

#import pandas as pd
from pyspark.sql import SparkSession
```

```
Requirement already satisfied: pandas in
c:\users\vikas\appdata\local\programs\python\python38\lib\site-packages (2.0.3)
Requirement already satisfied: numpy>=1.20.3 in
c:\users\vikas\appdata\local\programs\python\python38\lib\site-packages (from
pandas) (1.24.4)
Requirement already satisfied: tzdata>=2022.1 in
c:\users\vikas\appdata\local\programs\python\python38\lib\site-packages (from
pandas) (2025.1)
Requirement already satisfied: pytz>=2020.1 in
c:\users\vikas\appdata\local\programs\python\python38\lib\site-packages (from
pandas) (2025.1)
Requirement already satisfied: python-dateutil>=2.8.2 in
c:\users\vikas\appdata\roaming\python\python38\site-packages (from pandas)
(2.9.0.post0)
Requirement already satisfied: six>=1.5 in
c:\users\vikas\appdata\roaming\python\python38\site-packages (from python-
dateutil>=2.8.2->pandas) (1.17.0)
Note: you may need to restart the kernel to use updated packages.

WARNING: You are using pip version 21.1.1; however, version 25.0.1 is available.
You should consider upgrading via the
'c:\Users\vikas\AppData\Local\Programs\Python\Python38\python.exe -m pip install
--upgrade pip' command.
```

[22]:
```
# Create a Spark session
spark = SparkSession.builder.appName("MyApp").master("local[*]").config("spark.
 ↪network.timeout", "600s") \
    .config("spark.executor.heartbeatInterval", "60s") \
    .config("spark.executor.memory", "4g") \
    .config("spark.driver.memory", "2g") \
    .getOrCreate()

# here master is the URL of the cluster
```

```
# local[*] means we are running in local mode with as many worker threads as␣
 ↪logical cores on your machine
# config("spark.network.timeout", "600s") and config("spark.executor.
 ↪heartbeatInterval", "60s") are used to avoid disconnection of the spark␣
 ↪session due to inactivity for a long time
```

[23]:
```
spark
```

[23]:
```
<pyspark.sql.session.SparkSession at 0x1b81b0a0a30>
```

[9]:
```
people = spark.createDataFrame([
    {"deptId": 1, "age": 40, "name": "Hyukjin Kwon", "gender": "M", "salary":␣
 ↪50},
    {"deptId": 1, "age": None, "name": "Takuya Ueshin", "gender": "M", "salary":
 ↪ None},
    {"deptId": 2, "age": 60, "name": "Xinrong Meng", "gender": "F", "salary":␣
 ↪150},
    {"deptId": 3, "age": None, "name": "Haejoon Lee", "gender": "M", "salary":␣
 ↪200}
])

people.show()
```

```
+----+------+------+-------------+------+
| age|deptId|gender|         name|salary|
+----+------+------+-------------+------+
|  40|     1|     M| Hyukjin Kwon|    50|
|NULL|     1|     M|Takuya Ueshin|  NULL|
|  60|     2|     F| Xinrong Meng|   150|
|NULL|     3|     M|  Haejoon Lee|   200|
+----+------+------+-------------+------+
```

[24]:
```
type(people)
```

[24]:
```
pyspark.sql.dataframe.DataFrame
```

[11]:
```
people.printSchema()
```

```
root
 |-- age: long (nullable = true)
 |-- deptId: long (nullable = true)
 |-- gender: string (nullable = true)
 |-- name: string (nullable = true)
 |-- salary: long (nullable = true)
```

[12]:
```
people.dtypes
```

[12]:
```
[('age', 'bigint'),
 ('deptId', 'bigint'),
 ('gender', 'string'),
 ('name', 'string'),
 ('salary', 'bigint')]
```

[13]:
```
people.describe().show()
```

```
+-------+-----------------+-----------------+------+-----------+-------------
-----+
|summary|              age|           deptId|gender|       name|
salary|
+-------+-----------------+-----------------+------+-----------+-------------
-----+
|  count|                2|                4|     4|          4|
3|
|   mean|             50.0|             1.75|  NULL|
NULL|133.33333333333334|
| stddev|14.142135623730951|0.9574271077563381|  NULL|       NULL|
76.37626158259734|
|    min|               40|                1|     F| Haejoon Lee|
50|
|    max|               60|                3|     M|Xinrong Meng|
200|
+-------+-----------------+-----------------+------+-----------+-------------
-----+
```

[14]:
```
# if you want to rename the column name
people.withColumnRenamed("name", "full_name").show()
```

```
+----+------+------+------------+------+
| age|deptId|gender|   full_name|salary|
+----+------+------+------------+------+
|  40|     1|     M| Hyukjin Kwon|    50|
|NULL|     1|     M|Takuya Ueshin|  NULL|
|  60|     2|     F| Xinrong Meng|   150|
|NULL|     3|     M|  Haejoon Lee|   200|
+----+------+------+------------+------+
```

[15]:
```
#if you have to drop a column
people.drop("name").show()
```

```
+----+------+------+------+
| age|deptId|gender|salary|
```

```
+----+------+------+------+
|  40|     1|     M|    50|
|NULL|     1|     M|  NULL|
|  60|     2|     F|   150|
|NULL|     3|     M|   200|
+----+------+------+------+
```

[16]:
```
# if you want drop the NAN values in pyspark
people.na.drop().show()
```

```
+---+------+------+------------+------+
|age|deptId|gender|        name|salary|
+---+------+------+------------+------+
| 40|     1|     M|Hyukjin Kwon|    50|
| 60|     2|     F|Xinrong Meng|   150|
+---+------+------+------------+------+
```

[17]:
```
people.na.drop(subset=["name"]).show() # subset is used to drop the NAN values␣
↪in the specific column
```

```
+----+------+------+-------------+------+
| age|deptId|gender|         name|salary|
+----+------+------+-------------+------+
|  40|     1|     M| Hyukjin Kwon|    50|
|NULL|     1|     M|Takuya Ueshin|  NULL|
|  60|     2|     F| Xinrong Meng|   150|
|NULL|     3|     M|  Haejoon Lee|   200|
+----+------+------+-------------+------+
```

[18]:
```
people.na.drop(how="all").show() # drop the columns   if all the values are NAN
people.na.drop(how="any").show() #`drop the row if any of the values are NAN`
```

```
+----+------+------+-------------+------+
| age|deptId|gender|         name|salary|
+----+------+------+-------------+------+
|  40|     1|     M| Hyukjin Kwon|    50|
|NULL|     1|     M|Takuya Ueshin|  NULL|
|  60|     2|     F| Xinrong Meng|   150|
|NULL|     3|     M|  Haejoon Lee|   200|
+----+------+------+-------------+------+

+---+------+------+------------+------+
|age|deptId|gender|        name|salary|
+---+------+------+------------+------+
| 40|     1|     M|Hyukjin Kwon|    50|
```

```
| 60|     2|     F|Xinrong Meng|   150|
+---+------+------+------------+------+
```

[25]:
```
# fill the NAN values with the specific value
people.na.fill(0).show()
```

```
+---+------+------+-------------+------+
|age|deptId|gender|         name|salary|
+---+------+------+-------------+------+
| 40|     1|     M| Hyukjin Kwon|    50|
|  0|     1|     M|Takuya Ueshin|     0|
| 60|     2|     F| Xinrong Meng|   150|
|  0|     3|     M|  Haejoon Lee|   200|
+---+------+------+-------------+------+
```

[27]:
```
# fill the null values with mean value
from pyspark.sql.functions import mean
mean_val = people.select(mean(people.salary)).collect()
mean_val

people.na.fill(mean_val[0][0], subset=["salary"]).show()
people.show()
```

```
+----+------+------+-------------+------+
| age|deptId|gender|         name|salary|
+----+------+------+-------------+------+
|  40|     1|     M| Hyukjin Kwon|    50|
|NULL|     1|     M|Takuya Ueshin|   133|
|  60|     2|     F| Xinrong Meng|   150|
|NULL|     3|     M|  Haejoon Lee|   200|
+----+------+------+-------------+------+

+----+------+------+-------------+------+
| age|deptId|gender|         name|salary|
+----+------+------+-------------+------+
|  40|     1|     M| Hyukjin Kwon|    50|
|NULL|     1|     M|Takuya Ueshin|  NULL|
|  60|     2|     F| Xinrong Meng|   150|
|NULL|     3|     M|  Haejoon Lee|   200|
+----+------+------+-------------+------+
```

[87]:
```
people.show()
```

```
+----+------+------+-------------+------+
| age|deptId|gender|         name|salary|
```

```
+----+------+------+-------------+------+
|  40|     1|     M| Hyukjin Kwon|    50|
|NULL|     1|     M|Takuya Ueshin|  NULL|
|  60|     2|     F| Xinrong Meng|   150|
|NULL|     3|     M|  Haejoon Lee|   200|
+----+------+------+-------------+------+
```

[84]:
```
import pandas as pd
df=spark.createDataFrame(pd.DataFrame({
    'deptId':[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16],
    'age':[40,50,60,20,30,40,50,60,70,80,90,100,110,120,130,140],
    'name':['Hyukjin Kwon','Takuya Ueshin','Xinrong Meng','Haejoon␣
 ↪Lee','Hyukjin Kwon','Takuya Ueshin','Xinrong Meng','Haejoon Lee','Hyukjin␣
 ↪Kwon','Takuya Ueshin','Xinrong Meng','Haejoon Lee','Hyukjin Kwon','Takuya␣
 ↪Ueshin','Xinrong Meng','Haejoon Lee'],
}))

df.show()
```

```
+------+---+-------------+
|deptId|age|         name|
+------+---+-------------+
|     1| 40| Hyukjin Kwon|
|     2| 50|Takuya Ueshin|
|     3| 60| Xinrong Meng|
|     4| 20|  Haejoon Lee|
|     5| 30| Hyukjin Kwon|
|     6| 40|Takuya Ueshin|
|     7| 50| Xinrong Meng|
|     8| 60|  Haejoon Lee|
|     9| 70| Hyukjin Kwon|
|    10| 80|Takuya Ueshin|
|    11| 90| Xinrong Meng|
|    12|100|  Haejoon Lee|
|    13|110| Hyukjin Kwon|
|    14|120|Takuya Ueshin|
|    15|130| Xinrong Meng|
|    16|140|  Haejoon Lee|
+------+---+-------------+
```

[ ]:
```
from pyspark.sql import SparkSession
from pyspark.sql.types import StructType, StructField, IntegerType, StringType,␣
 ↪DoubleType
```

```
# Define schema for sales data
sales_schema = StructType([
    StructField("OrderID", IntegerType(), True),
    StructField("Customer", StringType(), True),
    StructField("Product", StringType(), True),
    StructField("Quantity", IntegerType(), True),
    StructField("Price", DoubleType(), True),
    StructField("TotalAmount", DoubleType(), True)
])

# Sample sales data
sales_data = [
    (101, "Alice", "Laptop", 1, 800.0, 800.0),
    (102, "Bob", "Headphones", 2, 500.0, 1000.0),
    (103, "Charlie", "Tablet", 1, 300.0, 300.0),
    (104, "David", "Headphones", 3, 50.0, 150.0),
    (105, "Emma", "Headphones", 1, 200.0, 200.0),
    (106, "Frank", "Headphones", 2, 400.0, 800.0),
    (107, "Grace", "Printer", 1, 200.0, 200.0),
    (108, "Henry", "Laptop", 1, 1200.0, 1200.0),
    (109, "Ivy", "Tablet", 2, 200.0, 400.0),
    (110, "John", "Keyboard", 1, 20.0, 20.0),
    (111, "Kate", "Printer", 1, 10.0, 10.0),
    (112, "Leo", "Laptop", 2, 100.0, 200.0),
    (113, "Mia", "Projector", 1, 500.0, 500.0),
    (114, "Nick", "Microphone", 1, 150.0, 150.0),
    (115, "Olivia", "Headphones", 1, 100.0, 100.0),
    (116, "Peter", "Router", 1, 60.0, 60.0),
    (117, "Queen", "Printer", 1, 300.0, 300.0),
    (118, "Robert", "Server", 1, 1500.0, 1500.0),
    (119, "Sophia", "Printer", 1, 200.0, 200.0),
    (120, "Tom", "Tablet", 1, 100.0, 100.0)
]

 # Create DataFrame
sales_df = spark.createDataFrame(sales_data, schema=sales_schema)

# Show DataFrame
sales_df.show()
```

```
+-------+--------+----------+--------+------+-----------+
|OrderID|Customer|   Product|Quantity| Price|TotalAmount|
+-------+--------+----------+--------+------+-----------+
|    101|   Alice|    Laptop|       1| 800.0|      800.0|
|    102|     Bob|Headphones|       2| 500.0|     1000.0|
|    103| Charlie|    Tablet|       1| 300.0|      300.0|
|    104|   David|Headphones|       3|  50.0|      150.0|
```

```
|    105|    Emma|Headphones|       1| 200.0|      200.0|
|    106|   Frank|Headphones|       2| 400.0|      800.0|
|    107|   Grace|   Printer|       1| 200.0|      200.0|
|    108|   Henry|    Laptop|       1|1200.0|     1200.0|
|    109|     Ivy|    Tablet|       2| 200.0|      400.0|
|    110|    John|  Keyboard|       1|  20.0|       20.0|
|    111|    Kate|   Printer|       1|  10.0|       10.0|
|    112|     Leo|    Laptop|       2| 100.0|      200.0|
|    113|     Mia| Projector|       1| 500.0|      500.0|
|    114|    Nick|Microphone|       1| 150.0|      150.0|
|    115|  Olivia|Headphones|       1| 100.0|      100.0|
|    116|   Peter|    Router|       1|  60.0|       60.0|
|    117|   Queen|   Printer|       1| 300.0|      300.0|
|    118|  Robert|    Server|       1|1500.0|     1500.0|
|    119|  Sophia|   Printer|       1| 200.0|      200.0|
|    120|     Tom|    Tablet|       1| 100.0|      100.0|
+-------+--------+----------+--------+------+-----------+
```

[6]:
```
# 3. Create DataFrames, perform operations like select, filter, groupBy, and aggregate
# Select columns
sales_df.select("Customer", "Product", "Price").show()
```

```
+--------+----------+------+
|Customer|   Product| Price|
+--------+----------+------+
|   Alice|    Laptop| 800.0|
|     Bob|Headphones| 500.0|
| Charlie|    Tablet| 300.0|
|   David|Headphones|  50.0|
|    Emma|Headphones| 200.0|
|   Frank|Headphones| 400.0|
|   Grace|   Printer| 200.0|
|   Henry|    Laptop|1200.0|
|     Ivy|    Tablet| 200.0|
|    John|  Keyboard|  20.0|
|    Kate|   Printer|  10.0|
|     Leo|    Laptop| 100.0|
|     Mia| Projector| 500.0|
|    Nick|Microphone| 150.0|
|  Olivia|Headphones| 100.0|
|   Peter|    Router|  60.0|
|   Queen|   Printer| 300.0|
|  Robert|    Server|1500.0|
|  Sophia|   Printer| 200.0|
|     Tom|    Tablet| 100.0|
+--------+----------+------+
```

[7]:
```
# Filter rows

sales_df.filter(sales_df["Price"] > 500).show()
```

```
+-------+--------+-------+--------+------+-----------+
|OrderID|Customer|Product|Quantity| Price|TotalAmount|
+-------+--------+-------+--------+------+-----------+
|    101|   Alice| Laptop|       1| 800.0|      800.0|
|    108|   Henry| Laptop|       1|1200.0|     1200.0|
|    118|  Robert| Server|       1|1500.0|     1500.0|
+-------+--------+-------+--------+------+-----------+
```

[8]:
```
# groupby
sales_df.groupBy("Product").count().show()
```

```
+----------+-----+
|   Product|count|
+----------+-----+
|    Laptop|    3|
|    Tablet|    3|
|Headphones|    5|
|   Printer|    4|
|  Keyboard|    1|
| Projector|    1|
|Microphone|    1|
|    Router|    1|
|    Server|    1|
+----------+-----+
```

[9]:
```
# aggregate

sales_df.groupBy("Product").agg({"Price": "sum"}).show()
```

```
+----------+----------+
|   Product|sum(Price)|
+----------+----------+
|    Laptop|    2100.0|
|    Tablet|     600.0|
|Headphones|    1250.0|
|   Printer|     710.0|
|  Keyboard|      20.0|
| Projector|     500.0|
|Microphone|     150.0|
|    Router|      60.0|
```

```
|    Server|    1500.0|
+----------+----------+
```

[14]:
```
type(sales_df)
```

[14]:
```
pyspark.sql.dataframe.DataFrame
```

[13]:
```
# mutliple aggregations
sales_df.groupBy("Product").agg({"Price": "sum", "Quantity": "sum"}).show()
```

```
+----------+----------+-------------+
|   Product|sum(Price)|sum(Quantity)|
+----------+----------+-------------+
|    Laptop|    2100.0|            4|
|    Tablet|     600.0|            4|
|Headphones|    1250.0|            9|
|   Printer|     710.0|            4|
|  Keyboard|      20.0|            1|
| Projector|     500.0|            1|
|Microphone|     150.0|            1|
|    Router|      60.0|            1|
|    Server|    1500.0|            1|
+----------+----------+-------------+
```

[11]:
```
# how to create new column in sales_df
sales_df.withColumn("DiscountedPrice", sales_df["Price"] * 0.9).show()
```

```
+-------+--------+----------+--------+------+-----------+---------------+
|OrderID|Customer|   Product|Quantity| Price|TotalAmount|DiscountedPrice|
+-------+--------+----------+--------+------+-----------+---------------+
|    101|   Alice|    Laptop|       1| 800.0|      800.0|          720.0|
|    102|     Bob|Headphones|       2| 500.0|     1000.0|          450.0|
|    103| Charlie|    Tablet|       1| 300.0|      300.0|          270.0|
|    104|   David|Headphones|       3|  50.0|      150.0|           45.0|
|    105|    Emma|Headphones|       1| 200.0|      200.0|          180.0|
|    106|   Frank|Headphones|       2| 400.0|      800.0|          360.0|
|    107|   Grace|   Printer|       1| 200.0|      200.0|          180.0|
|    108|   Henry|    Laptop|       1|1200.0|     1200.0|         1080.0|
|    109|     Ivy|    Tablet|       2| 200.0|      400.0|          180.0|
|    110|    John|  Keyboard|       1|  20.0|       20.0|           18.0|
|    111|    Kate|   Printer|       1|  10.0|       10.0|            9.0|
|    112|     Leo|    Laptop|       2| 100.0|      200.0|           90.0|
|    113|     Mia| Projector|       1| 500.0|      500.0|          450.0|
|    114|    Nick|Microphone|       1| 150.0|      150.0|          135.0|
|    115|  Olivia|Headphones|       1| 100.0|      100.0|           90.0|
|    116|   Peter|    Router|       1|  60.0|       60.0|           54.0|
```

```
|    117|   Queen|   Printer|       1| 300.0|      300.0|          270.0|
|    118|  Robert|    Server|       1|1500.0|     1500.0|         1350.0|
|    119|  Sophia|   Printer|       1| 200.0|      200.0|          180.0|
|    120|     Tom|    Tablet|       1| 100.0|      100.0|           90.0|
+-------+--------+----------+--------+------+-----------+---------------+
```

[ ]:

[ ]:

[ ]: