assignment7

September 18, 2024

[1]:
```
sc
```

[1]:
```
<SparkContext master=local[*] appName=PySparkShell>
```

[2]:
```
spark
```

[2]:
```
<pyspark.sql.session.SparkSession at 0x7ff5000e73c8>
```

0.0.1 a) Create a new Spark Session with new SparkConfig

[3]:
```
sc.stop()
spark.stop()
```

[4]:
```
from pyspark import SparkConf, SparkContext
# setMaster sets spark ContextManager which is loca[cpu cores]
config = SparkConf().setMaster('local[4]').setAppName("PySparkSession")
sc = SparkContext(conf=config)
```

[5]:
```
sc
```

[5]:
```
<SparkContext master=local[4] appName=PySparkSession>
```

0.0.2 b) Create new instance of Spark SQL session and define new DataFrame using sales_data_sample.csv dataset.

[6]:
```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("SQLSession").getOrCreate()
```

[7]:
```
spark
```

[7]:
```
<pyspark.sql.session.SparkSession at 0x7ff500065f28>
```

[8]:
```
sales_df = spark.read.csv('file:///home/hadoop/Downloads/sales_data_sample.
 ↪csv', header=True, inferSchema=True)
sales_df.show(5)
```

```
+----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+-----------+----+-----------+--------------------
+---------------+-------------------+-----------+------------+-----+--------
--+-------+---------+--------------+----------------+--------+
|ORDERNUMBER|QUANTITYORDERED|PRICEEACH|ORDERLINENUMBER|  SALES|      ORDERDATE|
STATUS|QTR_ID|MONTH_ID|YEAR_ID|PRODUCTLINE|MSRP|PRODUCTCODE|
CUSTOMERNAME|           PHONE|        ADDRESSLINE1|ADDRESSLINE2|         CITY|ST
ATE|POSTALCODE|COUNTRY|TERRITORY|CONTACTLASTNAME|CONTACTFIRSTNAME|DEALSIZE|
+----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+-----------+----+-----------+--------------------
+---------------+-------------------+-----------+------------+-----+--------
--+-------+---------+--------------+----------------+--------+
|      10107|             30|     95.7|             2| 2871.0| 2/24/2003
0:00|Shipped|     1|       2|   2003|Motorcycles|  95|   S10_1678|   Land of
Toys Inc.|      2125557818|897 Long Airport …|        null|          NYC|
NY|     10022|    USA|       NA|             Yu|            Kwai|   Small|
|      10121|             34|    81.35|             5| 2765.9|  5/7/2003
0:00|Shipped|     2|       5|   2003|Motorcycles|  95|   S10_1678|  Reims
Collectables|      26.47.1555|  59 rue de l'Abbaye|        null|        Reims|
null|     51100| France|     EMEA|        Henriot|            Paul|   Small|
|      10134|             41|    94.74|             2|3884.34|  7/1/2003
0:00|Shipped|     3|       7|   2003|Motorcycles|  95|   S10_1678|     Lyon
Souveniers|+33 1 46 62 7555|27 rue du Colonel…|        null|        Paris|
null|     75508| France|     EMEA|       Da Cunha|          Daniel|  Medium|
|      10145|             45|    83.26|             6| 3746.7| 8/25/2003
0:00|Shipped|     3|       8|   2003|Motorcycles|  95|   S10_1678|
Toys4GrownUps.com|      6265557265|  78934 Hillside Dr.|        null|
Pasadena|   CA|     90003|    USA|       NA|          Young|           Julie|
Medium|
|      10159|             49|    100.0|            14|5205.27|10/10/2003
0:00|Shipped|     4|      10|   2003|Motorcycles|  95|   S10_1678|Corporate Gift
Id…|      6505551386|     7734 Strong St.|        null|San Francisco|   CA|
null|    USA|       NA|          Brown|           Julie|  Medium|
+----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+-----------+----+-----------+--------------------
+---------------+-------------------+-----------+------------+-----+--------
--+-------+---------+--------------+----------------+--------+
only showing top 5 rows
```

[ ]:

[9]:
```
sales_df.printSchema()
```

```
root
 |-- ORDERNUMBER: integer (nullable = true)
 |-- QUANTITYORDERED: integer (nullable = true)
 |-- PRICEEACH: double (nullable = true)
```

```
 |-- ORDERLINENUMBER: integer (nullable = true)
 |-- SALES: double (nullable = true)
 |-- ORDERDATE: string (nullable = true)
 |-- STATUS: string (nullable = true)
 |-- QTR_ID: integer (nullable = true)
 |-- MONTH_ID: integer (nullable = true)
 |-- YEAR_ID: integer (nullable = true)
 |-- PRODUCTLINE: string (nullable = true)
 |-- MSRP: integer (nullable = true)
 |-- PRODUCTCODE: string (nullable = true)
 |-- CUSTOMERNAME: string (nullable = true)
 |-- PHONE: string (nullable = true)
 |-- ADDRESSLINE1: string (nullable = true)
 |-- ADDRESSLINE2: string (nullable = true)
 |-- CITY: string (nullable = true)
 |-- STATE: string (nullable = true)
 |-- POSTALCODE: string (nullable = true)
 |-- COUNTRY: string (nullable = true)
 |-- TERRITORY: string (nullable = true)
 |-- CONTACTLASTNAME: string (nullable = true)
 |-- CONTACTFIRSTNAME: string (nullable = true)
 |-- DEALSIZE: string (nullable = true)
```

0.0.3 c) Find the shape of DataFrame.

[10]:
```
rows = sales_df.count()
columns = len(sales_df.columns)
print(f'Shape of dataframe is: ({rows}, {columns})')
```

```
Shape of dataframe is: (2823, 25)
```

0.0.4 d) Find the Summary of DataFrame for all numerical data columns.

[42]:
```
from pyspark.sql.types import StringType
from pyspark.sql.functions import *

numerical_columns = [field.name for field in sales_df.schema.fields if not␣
 ↪isinstance(field.dataType, StringType)]
sales_df.select(numerical_columns).summary().show()

# INSIGHT: summary of the DataFrame for numerical columns shows count, min␣
 ↪vlaue, max value, and quartile values
```

```
+-------+-----------------+----------------+-----------------+--------------
--+-----------------+-----------------+-----------------+-----------------+-
```

```
-----------------+
|summary|       ORDERNUMBER|  QUANTITYORDERED|         PRICEEACH|
ORDERLINENUMBER|             SALES|            QTR_ID|          MONTH_ID|
YEAR_ID|              MSRP|
+-------+------------------+-----------------+------------------+--------------
--+------------------+-----------------+-----------------+------------------+-
-----------------+
|  count|              2823|             2823|              2823|
2823|              2823|             2823|              2823|
2823|              2823|
|   mean|10258.725115125753|35.09280906836698|
83.65854410201929|6.466170740347148|  3553.88907190932|2.7176762309599716|7.0924
548352816155|2003.8150903294368|100.71555083244775|
| stddev|  92.0854775957196| 9.74144273706958|20.174276527840536|
4.22584096469094|1841.8651057401842| 1.203878088001756|
3.656633307661765|0.6996701541300869| 40.18791167720266|
|    min|             10100|                6|             26.88|
1|            482.13|                1|                1|              2003|
33|
|    25%|             10180|               27|              68.8|
3|           2203.11|                2|                4|              2003|
68|
|    50%|             10262|               35|              95.7|
6|            3184.8|                3|                8|              2004|
99|
|    75%|             10334|               43|             100.0|
9|            4508.0|                4|               11|              2004|
124|
|    max|             10425|               97|             100.0|
18|           14082.8|                4|               12|              2005|
214|
+-------+------------------+-----------------+------------------+--------------
--+------------------+-----------------+-----------------+------------------+-
-----------------+
```

0.0.5 e) Identify and handle missing or null values in the columns.

[12]:
```
sales_df.select([count(when(isnull(col), col)).alias(col) for col in sales_df.
 ↪columns]).show()
# ADDRESSLINE2, STATE, POSTALCODE - all have NULL values in them - this might␣
 ↪need to be handled [dont drop]
```

```
+-----------+--------------+---------+---------------+-----+---------+------+--
----+--------+-------+-----------+----+-----------+------------+-----+----------
--+------------+----+-----+----------+-------+---------+--------------+--------
--------+---------+
```

```
|ORDERNUMBER|QUANTITYORDERED|PRICEEACH|ORDERLINENUMBER|SALES|ORDERDATE|STATUS|QT
R_ID|MONTH_ID|YEAR_ID|PRODUCTLINE|MSRP|PRODUCTCODE|CUSTOMERNAME|PHONE|ADDRESSLIN
E1|ADDRESSLINE2|CITY|STATE|POSTALCODE|COUNTRY|TERRITORY|CONTACTLASTNAME|CONTACTF
IRSTNAME|DEALSIZE|
+-----------+--------------+---------+--------------+-----+---------+------+--
----+--------+-------+----------+----+----------+------------+-----+----------
--+------------+----+-----+----------+-------+---------+--------------+--------
--------+---------+
|          0|             0|        0|             0|    0|        0|     0|
0|       0|      0|          0|   0|          0|           0|    0|           0|
2521|   0| 1486|        76|      0|        0|             0|               0|
0|
+-----------+--------------+---------+--------------+-----+---------+------+--
----+--------+-------+----------+----+----------+------------+-----+----------
--+------------+----+-----+----------+-------+---------+--------------+--------
--------+---------+
```

[13]:
```
# handling null values with empty string
sales_df1 = sales_df.fillna('')
```

0.0.6 f) Calculate the total revenue generated per country by combining the columns QUANTITYORDERED and PRICEEACH using Spark DataFrame operations?

[14]:
```
queryResult = sales_df1.select(
                col('COUNTRY'), (col('QUANTITYORDERED')*col('PRICEEACH')).
 ↪alias('REVENUESUM')
             ).groupBy(['COUNTRY']).agg(round(sum('REVENUESUM'), 3).
 ↪alias('TOTALREVENUE'))
queryResult.show()
```

```
+----------+------------+
|    COUNTRY|TOTALREVENUE|
+----------+------------+
|     Sweden|    174264.1|
|Philippines|    80291.17|
|  Singapore|    227985.5|
|    Germany|   178689.08|
|     France|   919257.85|
|    Belgium|    94528.88|
|    Finland|    268714.7|
|      Italy|   309402.87|
|     Norway|    246115.8|
|      Spain|  1021705.97|
|    Denmark|   192747.63|
```

```
|    Ireland|    43237.24|
|        USA|  2986425.21|
|         UK|   413203.34|
|Switzerland|    93344.91|
|     Canada|   193504.34|
|      Japan|   153076.69|
|  Australia|   521598.46|
|    Austria|   172793.05|
+-----------+------------+
```

0.0.7 g) Determine the top 5 products with the highest total sales revenue using Spark DataFrame?

[15]:
```
# from pyspark.sql.functions import *
# from pyspark.sql.types import *
```

[16]:
```
queryResult = sales_df1.select(
                col('PRODUCTLINE'), (col('SALES')).alias('TOTALREVENUE')
              ).groupBy(['PRODUCTLINE']).agg(round(sum('TOTALREVENUE'), 3).
 ↪alias('TOTALREVENUE'))\
              .orderBy(['TOTALREVENUE'], ascending=False)
queryResult.limit(5).show()
```

```
+---------------+------------+
|     PRODUCTLINE|TOTALREVENUE|
+---------------+------------+
|    Classic Cars|  3919615.66|
|    Vintage Cars|  1903150.84|
|     Motorcycles|  1166388.34|
|Trucks and Buses|  1127789.84|
|          Planes|   975003.57|
+---------------+------------+
```

0.0.8 h) Find the average order quantity for each product using groupBy and agg operations?

[17]:
```
queryResult = sales_df1.select(
                col('PRODUCTLINE'), col('QUANTITYORDERED')
              ).groupBy(['PRODUCTLINE']).\
              agg(round(avg('QUANTITYORDERED'), 3).alias('AVG_QUANTITYORDERED'))

queryResult.show()
```

```
+---------------+------------------+
|     PRODUCTLINE|AVG_QUANTITYORDERED|
+---------------+------------------+
|     Motorcycles|            35.236|
|    Vintage Cars|             34.71|
|           Ships|            34.731|
|Trucks and Buses|            35.804|
|    Classic Cars|            35.152|
|          Trains|            35.221|
|          Planes|            35.056|
+---------------+------------------+
```

0.0.9 i) Using Spark DataFrame, filter orders where the SALES value exceeds $10,000 and sort the results by the ORDERDATE column?

[18]:
```
#sales_df1.filter(col('SALES')>10000).show()
from pyspark.sql.functions import to_timestamp
sales_df1.withColumn('ORDERDATE', to_timestamp(col('ORDERDATE'), 'MM/dd/yyyy H:
 ↪mm'))\
            .select(col('ORDERNUMBER'), col('SALES'), col('ORDERDATE'))\
            .filter(col('SALES')>10000)\
            .orderBy('ORDERDATE').show()
```

```
+-----------+-------+-------------------+
|ORDERNUMBER|  SALES|          ORDERDATE|
+-----------+-------+-------------------+
|      10127|11279.2|2003-06-03 00:00:00|
|      10150|10993.5|2003-09-19 00:00:00|
|      10247|10606.2|2004-05-05 00:00:00|
|      10304|10172.7|2004-10-11 00:00:00|
|      10312|11623.7|2004-10-21 00:00:00|
|      10322|12536.5|2004-11-04 00:00:00|
|      10333|11336.7|2004-11-18 00:00:00|
|      10339|10758.0|2004-11-23 00:00:00|
|      10375|10039.6|2005-02-03 00:00:00|
|      10388|10066.6|2005-03-03 00:00:00|
|      10403|11886.6|2005-04-08 00:00:00|
|      10405|11739.7|2005-04-14 00:00:00|
|      10406|10468.9|2005-04-15 00:00:00|
|      10407|14082.8|2005-04-22 00:00:00|
|      10412|11887.8|2005-05-03 00:00:00|
|      10424|12001.0|2005-05-31 00:00:00|
+-----------+-------+-------------------+
```

0.0.10 j) Filter out rows where the STATUS is 'Cancelled' and calculate the total sales from the

remaining orders?

[19]:
```
sales_df1.filter(col('STATUS') != 'Cancelled').agg(round(sum('SALES'), 3).
↪alias('TOTAL SALES')).show()
```

```
+-----------+
|TOTAL SALES|
+-----------+
| 9838141.37|
+-----------+
```

0.0.11 k) Use Spark Data Frame transformations to derive the yearly sales for each customer (CUSTOMERNAME) based on the ORDERDATE column?

[20]:
```
queryResult = sales_df1.withColumn('ORDERDATE', to_timestamp(col('ORDERDATE'),␣
↪'MM/dd/yyyy H:mm'))\
              .select(col('CUSTOMERNAME'), year('ORDERDATE').alias('YEAR'),␣
↪col('SALES'))\
              .groupBy(['CUSTOMERNAME', 'YEAR'])\
              .agg(round(sum('SALES'), 3).alias('AVG_QUANTITYORDERED'))\
              .orderBy(['CUSTOMERNAME', 'YEAR'])

queryResult.show()
```

```
+--------------------+----+-------------------+
|        CUSTOMERNAME|YEAR|AVG_QUANTITYORDERED|
+--------------------+----+-------------------+
|      AV Stores, Co.|2003|           51017.92|
|      AV Stores, Co.|2004|          106789.89|
|        Alpha Cognac|2003|           55349.32|
|        Alpha Cognac|2005|           15139.12|
|  Amica Models & Co.|2004|           94117.26|
|Anna's Decoration…|2003|          88983.71|
|Anna's Decoration…|2005|          65012.42|
|   Atelier graphique|2003|            16560.3|
|   Atelier graphique|2004|            7619.66|
|Australian Collec…|2003|          37878.55|
|Australian Collec…|2004|          12334.82|
|Australian Collec…|2005|          14378.09|
|Australian Collec…|2003|          60135.84|
|Australian Collec…|2004|         140859.57|
|Australian Gift N…|2003|          37739.09|
|Australian Gift N…|2005|          21730.03|
```

```
|  Auto Assoc. & Cie.|2004|           64834.32|
|    Auto Canal Petit|2004|           79103.86|
|    Auto Canal Petit|2005|            14066.8|
|Auto-Moto Classic…|2003|            7277.35|
+--------------------+----+--------------------+
only showing top 20 rows
```

0.0.12 l) Add a new column to the DataFrame that categorizes orders as 'High', 'Medium', or 'Low' sales based on the SALES value?

[21]:
```
from pyspark.sql.window import Window
from pyspark.sql.functions import ntile

windowSpec = Window.orderBy('SALES')
sales_df1.withColumn('quantile',ntile(3).over(windowSpec))\
         .select(col('ORDERNUMBER'), col('SALES'), col('quantile'))\
         .withColumn(
             'SALES_CATEGORY',
             when(col('quantile')==1, 'Low')\
             .when(col('quantile')==2, 'Medium')\
             .when(col('quantile')==3, 'High')
         )\
         .select(col('ORDERNUMBER'), col('SALES'), col('SALES_CATEGORY'))\
         .show()
```

```
+-----------+------+--------------+
|ORDERNUMBER| SALES|SALES_CATEGORY|
+-----------+------+--------------+
|      10425|482.13|           Low|
|      10407|541.14|           Low|
|      10408|553.95|           Low|
|      10280| 577.6|           Low|
|      10419|640.05|           Low|
|      10264| 651.8|           Low|
|      10420|652.35|           Low|
|      10214| 683.8|           Low|
|      10304| 694.6|           Low|
|      10344| 703.6|           Low|
|      10110| 710.2|           Low|
|      10135| 717.4|           Low|
|      10114|721.44|           Low|
|      10358| 728.4|           Low|
|      10375|733.11|           Low|
|      10193|759.46|           Low|
|      10203| 777.0|           Low|
|      10409|785.64|           Low|
```

```
|      10281| 813.2|           Low|
|      10156| 820.4|           Low|
+-----------+------+--------------+
only showing top 20 rows
```

0.0.13 m) Assume , If you have another DataFrame with customer demographic data, how would you perform a join to compute the total sales per demographic group?

[22]:
```
customer_demographic_data = spark.read.csv('file:///home/hadoop/Downloads/
 ↪sales_data_customer_demographic.csv', header=True, inferSchema=True)
joined_df = sales_df1.join(customer_demographic_data, sales_df1.
 ↪CUSTOMERNAME==customer_demographic_data.CUSTOMERNAME, "inner")
joined_df.show()
```

```
+-----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+-----------+----+-----------+--------------------
+---------------+--------------------+-----------+-------------+--------+-----
-----+---------+---------+---------------+---------------+--------+-----------
--------+---------+-------------+
|ORDERNUMBER|QUANTITYORDERED|PRICEEACH|ORDERLINENUMBER|  SALES|      ORDERDATE|
STATUS|QTR_ID|MONTH_ID|YEAR_ID|PRODUCTLINE|MSRP|PRODUCTCODE|
CUSTOMERNAME|           PHONE|        ADDRESSLINE1|ADDRESSLINE2|         CITY|
STATE|POSTALCODE|  COUNTRY|TERRITORY|CONTACTLASTNAME|CONTACTFIRSTNAME|DEALSIZE|
CUSTOMERNAME|ESTD_YEAR|NUM_EMPLOYEES|
+-----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+-----------+----+-----------+--------------------
+---------------+--------------------+-----------+-------------+--------+-----
-----+---------+---------+---------------+---------------+--------+-----------
--------+---------+-------------+
|      10107|             30|     95.7|             2| 2871.0| 2/24/2003
0:00|Shipped|     1|       2|   2003|Motorcycles|  95|   S10_1678|   Land of
Toys Inc.|      2125557818|897 Long Airport …|            |          NYC|
NY|     10022|      USA|       NA|             Yu|            Kwai|   Small|
Land of Toys Inc.|     2009|           75|
|      10121|             34|    81.35|             5| 2765.9|  5/7/2003
0:00|Shipped|     2|       5|   2003|Motorcycles|  95|   S10_1678|  Reims
Collectables|      26.47.1555|  59 rue de l'Abbaye|            |        Reims|
|     51100|   France|     EMEA|        Henriot|            Paul|   Small|
Reims Collectables|     2015|           70|
|      10134|             41|    94.74|             2|3884.34|  7/1/2003
0:00|Shipped|     3|       7|   2003|Motorcycles|  95|   S10_1678|     Lyon
Souveniers|+33 1 46 62 7555|27 rue du Colonel…|            |        Paris|
|     75508|   France|     EMEA|       Da Cunha|          Daniel|  Medium|
Lyon Souveniers|     1995|           60|
|      10145|             45|    83.26|             6| 3746.7| 8/25/2003
```

```
0:00|Shipped|     3|       8|   2003|Motorcycles|  95|   S10_1678|
Toys4GrownUps.com|      6265557265|  78934 Hillside Dr.|            |
Pasadena|      CA|     90003|      USA|       NA|          Young|
Julie|  Medium|   Toys4GrownUps.com|     2008|          120|
|      10159|             49|    100.0|             14|5205.27|10/10/2003
0:00|Shipped|     4|      10|   2003|Motorcycles|  95|   S10_1678|Corporate Gift
Id…|      6505551386|     7734 Strong St.|            |San Francisco|      CA|
|      USA|       NA|          Brown|           Julie|  Medium|Corporate Gift
Id…|     1999|           50|
|      10168|             36|    96.66|              1|3479.76|10/28/2003
0:00|Shipped|     4|      10|   2003|Motorcycles|  95|   S10_1678|Technics
Stores Inc.|      6505556809|   9408 Furth Circle|            |   Burlingame|
CA|     94217|      USA|       NA|         Hirano|            Juri|
Medium|Technics Stores Inc.|     2010|           80|
|      10180|             29|    86.13|              9|2497.77|11/11/2003
0:00|Shipped|     4|      11|   2003|Motorcycles|  95|   S10_1678|Daedalus
Designs …|      20.16.1555|184, chausse de T…|            |        Lille|
|     59000|   France|     EMEA|          Rance|         Martine|
Small|Daedalus Designs …|     2002|          100|
|      10188|             48|    100.0|             1|5512.32|11/18/2003
0:00|Shipped|     4|      11|   2003|Motorcycles|  95|   S10_1678|        Herkku
Gifts|   +47 2267 3215|Drammen 121, PR 7…|            |       Bergen|        |
N 5804|   Norway|     EMEA|         Oeztan|          Veysel|  Medium|
Herkku Gifts|     2000|           90|
|      10201|             22|    98.57|              2|2168.54| 12/1/2003
0:00|Shipped|     4|      12|   2003|Motorcycles|  95|   S10_1678|     Mini
Wheels Co.|      6505555787|5557 North Pendal…|           |San Francisco|
CA|          |      USA|       NA|         Murphy|           Julie|   Small|
Mini Wheels Co.|     2009|           35|
|      10211|             41|    100.0|             14|4708.44| 1/15/2004
0:00|Shipped|     1|       1|   2004|Motorcycles|  95|   S10_1678|    Auto Canal
Petit|  (1) 47.55.6555|   25, rue Lauriston|            |        Paris|        |
75016|   France|     EMEA|        Perrier|       Dominique|  Medium|    Auto
Canal Petit|     2002|           55|
|      10223|             37|    100.0|              1|3965.66| 2/20/2004
0:00|Shipped|     1|       2|   2004|Motorcycles|  95|   S10_1678|Australian
Collec…|    03 9520 4555|   636 St Kilda Road|     Level 3|
Melbourne|Victoria|      3004|Australia|     APAC|       Ferguson|
Peter|  Medium|Australian Collec…|     2007|           60|
|      10237|             23|    100.0|              7|2333.12|  4/5/2004
0:00|Shipped|     2|       4|   2004|Motorcycles|  95|   S10_1678|
Vitachrome Inc.|      2125551500|   2678 Kingston Rd.|   Suite 101|
NYC|      NY|     10022|      USA|       NA|          Frick|         Michael|
Small|     Vitachrome Inc.|     2015|           25|
|      10251|             28|    100.0|              2|3188.64| 5/18/2004
0:00|Shipped|     2|       5|   2004|Motorcycles|  95|   S10_1678|Tekni
Collectable…|      2015559350|       7476 Moss Rd.|            |       Newark|
NJ|     94019|      USA|       NA|          Brown|         William|
```

```
Medium|Tekni Collectable…|     2007|           85|
|      10263|             34|    100.0|              2|3676.76| 6/28/2004
0:00|Shipped|     2|       6|   2004|Motorcycles|  95|   S10_1678|     Gift
Depot Inc.|      2035552570| 25593 South Bay Ln.|            |  Bridgewater|
CT|     97562|      USA|       NA|           King|           Julie|  Medium|
Gift Depot Inc.|     2002|           80|
|      10275|             45|    92.83|              1|4177.35| 7/23/2004
0:00|Shipped|     3|       7|   2004|Motorcycles|  95|   S10_1678|   La Rochelle
Gifts|      40.67.8555|67, rue des Cinqu…|            |       Nantes|        |
44000|   France|     EMEA|        Labrune|          Janine|  Medium|   La
Rochelle Gifts|     1999|           55|
|      10285|             36|    100.0|              6|4099.68| 8/27/2004
0:00|Shipped|     3|       8|   2004|Motorcycles|  95|   S10_1678|Marta's
Replicas Co.|      6175558555| 39323 Spinnaker Dr.|            |    Cambridge|
MA|     51247|      USA|       NA|      Hernandez|           Marta|
Medium|Marta's Replicas Co.|     2009|           45|
|      10299|             23|    100.0|              9|2597.39| 9/30/2004
0:00|Shipped|     3|       9|   2004|Motorcycles|  95|   S10_1678|Toys of
Finland, Co.|     90-224 8555|       Keskuskatu 45|            |     Helsinki|
|     21240|  Finland|     EMEA|      Karttunen|           Matti|   Small|Toys
of Finland, Co.|     2014|           50|
|      10309|             41|    100.0|              5|4394.38|10/15/2004
0:00|Shipped|     4|      10|   2004|Motorcycles|  95|   S10_1678|  Baane Mini
Imports|      07-98 9555|Erling Skakkes ga…|            |      Stavern|
|      4110|   Norway|     EMEA|     Bergulfsen|           Jonas|  Medium|
Baane Mini Imports|     1997|           60|
|      10318|             46|    94.74|              1|4358.04| 11/2/2004
0:00|Shipped|     4|      11|   2004|Motorcycles|  95|   S10_1678|Diecast
Classics …|      2155551555|    7586 Pompton St.|            |    Allentown|
PA|     70267|      USA|       NA|             Yu|           Kyung|
Medium|Diecast Classics …|     2006|           65|
|      10329|             42|    100.0|              1|4396.14|11/15/2004
0:00|Shipped|     4|      11|   2004|Motorcycles|  95|   S10_1678|   Land of
Toys Inc.|      2125557818|897 Long Airport …|            |          NYC|
NY|     10022|      USA|       NA|             Yu|            Kwai|  Medium|
Land of Toys Inc.|     2009|           75|
+----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+-----------+----+-----------+--------------------
+---------------+-------------------+------------+-------------+--------+-----
-----+---------+---------+---------------+---------------+--------+------------
--------+---------+-------------+
only showing top 20 rows
```

0.0.14 n) Can you implement a cumulative distribution function (CDF) over the SALES value for each CUSTOMERNAME? What insights can you gather from analyzing the CDF distribution for each customer?

[40]:
```
#from pyspark.sql.functions

#sales_df1.groupBy(['CUSTOMERNAME'])\
#          .agg(
#             round(sum('SALES'), 3).alias('SUM'),
#             round(avg('SALES'), 3).alias('MEAN'),
#             round(expr(f"percentile_approx({'SALES'}, 0.5)"), 3).
 ↪alias('MEDIAN'),
#             round(stddev('SALES'), 3).alias('STDDEV'),
#             round(count('SALES'), 3).alias('COUNT'),
#             round(max('SALES'), 3).alias('MAX'),
#             round(min('SALEs'), 3).alias('MIN')
#          )\
#          .show()

from pyspark.sql.functions import cume_dist
from pyspark.sql.window import Window

sales_df1\
.withColumn("cume_dist", round(cume_dist().over(Window.
 ↪partitionBy('CUSTOMERNAME').orderBy("SALES")), 5))\
.select("ORDERNUMBER", "CUSTOMERNAME", "SALES", "cume_dist")\
.show(50)

# INSIGHT: for the first row, cume_dist is 0.033 means that for the particular␣
 ↪customer,
# SALES value of that order is higher than 3.3% of all SALES values of the␣
 ↪customer
```

```
+-----------+------------------+-------+---------+
|ORDERNUMBER|       CUSTOMERNAME|  SALES|cume_dist|
+-----------+------------------+-------+---------+
|      10141|Suominen Souveniers| 891.03|  0.03333|
|      10141|Suominen Souveniers| 1086.6|  0.06667|
|      10141|Suominen Souveniers|1103.76|      0.1|
|      10363|Suominen Souveniers|1629.04|  0.13333|
|      10247|Suominen Souveniers| 1988.4|  0.16667|
|      10141|Suominen Souveniers|2140.11|      0.2|
|      10363|Suominen Souveniers|2447.76|  0.23333|
|      10363|Suominen Souveniers|2632.89|  0.26667|
|      10363|Suominen Souveniers| 2773.8|      0.3|
|      10363|Suominen Souveniers|2775.08|  0.33333|
|      10363|Suominen Souveniers|2817.87|  0.36667|
```

```
|      10363|Suominen Souveniers|2851.84|      0.4|
|      10363|Suominen Souveniers|2931.98|  0.43333|
|      10247|Suominen Souveniers|3128.65|  0.46667|
|      10363|Suominen Souveniers|3288.82|      0.5|
|      10363|Suominen Souveniers|3595.62|  0.53333|
|      10363|Suominen Souveniers|3686.54|  0.56667|
|      10141|Suominen Souveniers| 3784.8|      0.6|
|      10363|Suominen Souveniers| 4068.7|  0.63333|
|      10363|Suominen Souveniers|4142.64|  0.66667|
|      10247|Suominen Souveniers|4157.73|      0.7|
|      10247|Suominen Souveniers|4381.25|  0.73333|
|      10141|Suominen Souveniers| 4836.5|  0.76667|
|      10363|Suominen Souveniers|5154.41|      0.8|
|      10141|Suominen Souveniers|5500.44|  0.83333|
|      10141|Suominen Souveniers|5938.53|  0.86667|
|      10141|Suominen Souveniers|6287.66|      0.9|
|      10363|Suominen Souveniers| 6576.5|  0.93333|
|      10247|Suominen Souveniers| 6756.0|  0.96667|
|      10247|Suominen Souveniers|10606.2|      1.0|
|      10280| Amica Models & Co.|  577.6|  0.03846|
|      10280| Amica Models & Co.|1381.05|  0.07692|
|      10280| Amica Models & Co.|1557.36|  0.11538|
|      10280| Amica Models & Co.| 1574.0|  0.15385|
|      10280| Amica Models & Co.|1656.69|  0.19231|
|      10293| Amica Models & Co.|1921.92|  0.23077|
|      10293| Amica Models & Co.|2084.81|  0.26923|
|      10280| Amica Models & Co.|2137.05|  0.30769|
|      10293| Amica Models & Co.|2418.24|  0.34615|
|      10280| Amica Models & Co.|2800.08|  0.38462|
|      10293| Amica Models & Co.|2819.28|  0.42308|
|      10293| Amica Models & Co.|2941.89|  0.46154|
|      10280| Amica Models & Co.|2954.53|      0.5|
|      10280| Amica Models & Co.|3006.43|  0.53846|
|      10280| Amica Models & Co.|3474.46|  0.57692|
|      10280| Amica Models & Co.| 3668.6|  0.61538|
|      10280| Amica Models & Co.|3704.05|  0.65385|
|      10293| Amica Models & Co.|4242.24|  0.69231|
|      10280| Amica Models & Co.| 4455.0|  0.73077|
|      10280| Amica Models & Co.| 4750.8|  0.76923|
+-----------+-------------------+-------+---------+
only showing top 50 rows
```

0.0.15 o) Write spark dataframe code to rank products by total revenue within each country (COUNTRY)?

[24]:
```
from pyspark.sql.window import Window
from pyspark.sql.functions import dense_rank


sales_df1.withColumn(
    'REVENUESUM',col('QUANTITYORDERED')*col('PRICEEACH')
)\
.groupBy(['COUNTRY', 'PRODUCTLINE']).agg(round(sum('REVENUESUM'), 3).
 ↪alias('TOTALREVENUE'))\
.withColumn('rank', dense_rank().over(Window.partitionBy('COUNTRY').
 ↪orderBy(col('TOTALREVENUE').desc())))\
.show()
```

```
+-----------+---------------+------------+----+
|    COUNTRY|     PRODUCTLINE|TOTALREVENUE|rank|
+-----------+---------------+------------+----+
|     Sweden|    Classic Cars|    50377.62|   1|
|     Sweden|Trucks and Buses|    39562.44|   2|
|     Sweden|    Vintage Cars|    31784.94|   3|
|     Sweden|           Ships|    29514.62|   4|
|     Sweden|     Motorcycles|     12388.6|   5|
|     Sweden|          Planes|     7435.88|   6|
|     Sweden|          Trains|      3200.0|   7|
|Philippines|    Classic Cars|    43815.85|   1|
|Philippines|     Motorcycles|     17491.9|   2|
|Philippines|          Planes|    17048.33|   3|
|Philippines|    Vintage Cars|     1935.09|   4|
|  Singapore|    Classic Cars|    91791.76|   1|
|  Singapore|Trucks and Buses|    75797.19|   2|
|  Singapore|    Vintage Cars|     30221.0|   3|
|  Singapore|           Ships|    13065.74|   4|
|  Singapore|          Trains|    12934.21|   5|
|  Singapore|     Motorcycles|      4175.6|   6|
|    Germany|    Classic Cars|   113357.71|   1|
|    Germany|          Planes|    20786.78|   2|
|    Germany|    Vintage Cars|    18480.53|   3|
+-----------+---------------+------------+----+
only showing top 20 rows
```

0.0.16 p) Calculate a running total of SALES for each customer and show the top 5 customers by this cumulative total?

[25]:
```
from pyspark.sql.window import Window

sales_df1\
.groupBy(['CUSTOMERNAME'])\
.agg(round(sum('SALES'), 3).alias('TOTALSALES'))\
.withColumn(
        'cumulative_sum',
        sum('TOTALSALES')\
            .over(Window.orderBy('TOTALSALES')\
                .rowsBetween(Window.unboundedPreceding, Window.currentRow))
).orderBy('cumulative_sum', ascending=False).show(5)
```

```
+--------------------+----------+------------------+
|        CUSTOMERNAME|TOTALSALES|    cumulative_sum|
+--------------------+----------+------------------+
|Euro Shopping Cha…| 912294.11|     1.003262885E7|
|Mini Gifts Distri…| 654858.06|        9120334.74|
|Australian Collec…| 200995.41|        8465476.68|
|  Muscle Machine Inc| 197736.94|8264481.2700000005|
|   La Rochelle Gifts|  180124.9|        8066744.33|
+--------------------+----------+------------------+
only showing top 5 rows
```

0.0.17 q) Find and handle Invalid and Outliers values in entire DataFrame. (Check for only continuous dataset).

[26]:
```
# "ORDERLINENUMBER", "ORDERNUMBER", should be included? [coz not necessarily␣
 ↪continuous]
col_list = ["QUANTITYORDERED", "PRICEEACH", "SALES", "QTR_ID", "MONTH_ID",␣
 ↪"YEAR_ID", "MSRP"]


# creating a deepcopy for us to work on
new_dataframe = sales_df1.alias("new_dataframe")

# only from the columns we determined as continuous, drop NA values
new_dataframe = new_dataframe.dropna(subset=col_list)

# for each filed, filter out values outside threshold
for fieldName in col_list:
    q1 = new_dataframe.approxQuantile(fieldName, [0.25], relativeError=0.
 ↪0001)[0]
```

```
    q3 = new_dataframe.approxQuantile(fieldName, [0.75], relativeError=0.
↪0001)[0]

    minThreshold = q1 - 1.5*(q3-q1)
    maxThreshold = q3 + 1.5*(q3-q1)
    #print(f"{fieldName}, {minThreshold}, {maxThreshold}")

    new_dataframe = new_dataframe.filter((col(fieldName)>=minThreshold) &␣
↪(col(fieldName)<=maxThreshold))

new_dataframe.show()
```

```
+----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+-----------+----+-----------+--------------------
+---------------+-------------------+-----------+-------------+--------+-----
-----+---------+---------+---------------+----------------+--------+
|ORDERNUMBER|QUANTITYORDERED|PRICEEACH|ORDERLINENUMBER|  SALES|      ORDERDATE|
STATUS|QTR_ID|MONTH_ID|YEAR_ID|PRODUCTLINE|MSRP|PRODUCTCODE|
CUSTOMERNAME|           PHONE|        ADDRESSLINE1|ADDRESSLINE2|         CITY|
STATE|POSTALCODE|  COUNTRY|TERRITORY|CONTACTLASTNAME|CONTACTFIRSTNAME|DEALSIZE|
+----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+-----------+----+-----------+--------------------
+---------------+-------------------+-----------+-------------+--------+-----
-----+---------+---------+---------------+----------------+--------+
|      10107|             30|     95.7|             2| 2871.0| 2/24/2003
0:00|Shipped|     1|       2|   2003|Motorcycles|  95|   S10_1678|   Land of
Toys Inc.|      2125557818|897 Long Airport …|            |          NYC|
NY|     10022|      USA|       NA|             Yu|            Kwai|   Small|
|      10121|             34|    81.35|             5| 2765.9|  5/7/2003
0:00|Shipped|     2|       5|   2003|Motorcycles|  95|   S10_1678|  Reims
Collectables|      26.47.1555|  59 rue de l'Abbaye|            |        Reims|
|     51100|   France|     EMEA|        Henriot|            Paul|   Small|
|      10134|             41|    94.74|             2|3884.34|  7/1/2003
0:00|Shipped|     3|       7|   2003|Motorcycles|  95|   S10_1678|     Lyon
Souveniers|+33 1 46 62 7555|27 rue du Colonel…|            |        Paris|
|     75508|   France|     EMEA|       Da Cunha|          Daniel|  Medium|
|      10145|             45|    83.26|             6| 3746.7| 8/25/2003
0:00|Shipped|     3|       8|   2003|Motorcycles|  95|   S10_1678|
Toys4GrownUps.com|      6265557265|  78934 Hillside Dr.|            |
Pasadena|      CA|     90003|      USA|       NA|          Young|
Julie|  Medium|
|      10159|             49|    100.0|            14|5205.27|10/10/2003
0:00|Shipped|     4|      10|   2003|Motorcycles|  95|   S10_1678|Corporate Gift
Id…|      6505551386|     7734 Strong St.|            |San Francisco|      CA|
|      USA|       NA|          Brown|           Julie|  Medium|
|      10168|             36|    96.66|             1|3479.76|10/28/2003
0:00|Shipped|     4|      10|   2003|Motorcycles|  95|   S10_1678|Technics
```

```
Stores Inc.|      6505556809|   9408 Furth Circle|            |   Burlingame|
CA|     94217|      USA|       NA|         Hirano|            Juri|  Medium|
|      10180|             29|    86.13|             9|2497.77|11/11/2003
0:00|Shipped|     4|      11|   2003|Motorcycles|  95|   S10_1678|Daedalus
Designs …|      20.16.1555|184, chausse de T…|            |        Lille|
|     59000|   France|     EMEA|          Rance|         Martine|   Small|
|      10188|             48|    100.0|             1|5512.32|11/18/2003
0:00|Shipped|     4|      11|   2003|Motorcycles|  95|   S10_1678|        Herkku
Gifts|   +47 2267 3215|Drammen 121, PR 7…|            |       Bergen|        |
N 5804|   Norway|     EMEA|         Oeztan|          Veysel|  Medium|
|      10201|             22|    98.57|             2|2168.54| 12/1/2003
0:00|Shipped|     4|      12|   2003|Motorcycles|  95|   S10_1678|     Mini
Wheels Co.|      6505555787|5557 North Pendal…|           |San Francisco|
CA|          |      USA|       NA|         Murphy|           Julie|   Small|
|      10211|             41|    100.0|            14|4708.44| 1/15/2004
0:00|Shipped|     1|       1|   2004|Motorcycles|  95|   S10_1678|    Auto Canal
Petit|  (1) 47.55.6555|   25, rue Lauriston|            |        Paris|        |
75016|   France|     EMEA|        Perrier|       Dominique|  Medium|
|      10223|             37|    100.0|             1|3965.66| 2/20/2004
0:00|Shipped|     1|       2|   2004|Motorcycles|  95|   S10_1678|Australian
Collec…|    03 9520 4555|   636 St Kilda Road|     Level 3|
Melbourne|Victoria|      3004|Australia|     APAC|       Ferguson|
Peter|  Medium|
|      10237|             23|    100.0|             7|2333.12|  4/5/2004
0:00|Shipped|     2|       4|   2004|Motorcycles|  95|   S10_1678|
Vitachrome Inc.|      2125551500|   2678 Kingston Rd.|   Suite 101|
NYC|      NY|     10022|      USA|       NA|          Frick|         Michael|
Small|
|      10251|             28|    100.0|             2|3188.64| 5/18/2004
0:00|Shipped|     2|       5|   2004|Motorcycles|  95|   S10_1678|Tekni
Collectable…|      2015559350|       7476 Moss Rd.|            |       Newark|
NJ|     94019|      USA|       NA|          Brown|         William|  Medium|
|      10263|             34|    100.0|             2|3676.76| 6/28/2004
0:00|Shipped|     2|       6|   2004|Motorcycles|  95|   S10_1678|     Gift
Depot Inc.|      2035552570| 25593 South Bay Ln.|            |  Bridgewater|
CT|     97562|      USA|       NA|           King|           Julie|  Medium|
|      10275|             45|    92.83|             1|4177.35| 7/23/2004
0:00|Shipped|     3|       7|   2004|Motorcycles|  95|   S10_1678|   La Rochelle
Gifts|      40.67.8555|67, rue des Cinqu…|            |       Nantes|        |
44000|   France|     EMEA|        Labrune|          Janine|  Medium|
|      10285|             36|    100.0|             6|4099.68| 8/27/2004
0:00|Shipped|     3|       8|   2004|Motorcycles|  95|   S10_1678|Marta's
Replicas Co.|      6175558555| 39323 Spinnaker Dr.|            |    Cambridge|
MA|     51247|      USA|       NA|      Hernandez|           Marta|  Medium|
|      10299|             23|    100.0|             9|2597.39| 9/30/2004
0:00|Shipped|     3|       9|   2004|Motorcycles|  95|   S10_1678|Toys of
Finland, Co.|     90-224 8555|       Keskuskatu 45|            |     Helsinki|
|     21240|  Finland|     EMEA|      Karttunen|           Matti|   Small|
```

```
|      10309|             41|    100.0|             5|4394.38|10/15/2004
0:00|Shipped|     4|      10|   2004|Motorcycles|  95|   S10_1678|  Baane Mini
Imports|      07-98 9555|Erling Skakkes ga…|           |      Stavern|
|      4110|   Norway|     EMEA|     Bergulfsen|           Jonas|  Medium|
|      10318|             46|    94.74|             1|4358.04| 11/2/2004
0:00|Shipped|     4|      11|   2004|Motorcycles|  95|   S10_1678|Diecast
Classics …|      2155551555|    7586 Pompton St.|            |    Allentown|
PA|     70267|      USA|       NA|             Yu|           Kyung|  Medium|
|      10329|             42|    100.0|             1|4396.14|11/15/2004
0:00|Shipped|     4|      11|   2004|Motorcycles|  95|   S10_1678|   Land of
Toys Inc.|      2125557818|897 Long Airport …|            |          NYC|
NY|     10022|      USA|       NA|             Yu|            Kwai|  Medium|
+----------+--------------+---------+--------------+-------+---------------+-
------+------+--------+-------+-----------+----+-----------+-------------------
+---------------+-------------------+-----------+-------------+--------+-----
-----+---------+---------+---------------+----------------+--------+
only showing top 20 rows
```

[27]:
```
#sales_df.printSchema()
```

0.0.18 r) How would you cache a DataFrame containing sales data from the top 10 countries by sales to avoid recomputation in subsequent transformations? What persistence level (e.g. MEMORY_ONLY, MEMORY_AND_DISK) would you choose and why?

[28]:
```
countries_top10 = sales_df1.groupBy(['COUNTRY'])\
.agg(round(sum('SALES'), 3).alias("TOTALSALES"))\
.orderBy(col('TOTALSALES').desc())\
.limit(10)

countries_top10.show()
```

```
+---------+----------+
|  COUNTRY|TOTALSALES|
+---------+----------+
|      USA|3627982.83|
|    Spain|1215686.92|
|   France|1110916.52|
|Australia|  630623.1|
|       UK| 478880.46|
|    Italy| 374674.31|
|  Finland| 329581.91|
|   Norway|  307463.7|
|Singapore| 288488.41|
|  Denmark| 245637.15|
```

```
+---------+----------+
```

[29]:
```
# caching
countries_top10_cached = countries_top10.cache()
```

[30]:
```
# (persistence)[https://sparkbyexamples.com/pyspark/pyspark-persist-in-detail/]
from pyspark import StorageLevel
countries_top10_persisted = countries_top10.persist(StorageLevel.MEMORY_ONLY)

# the size of data that we are smalling here is small [10 rows and 2 columns],␣
 ↪which can sit in the memory
# MEMORY_ONLY_SER is not used as serialization and deserialization requires␣
 ↪overhead in processing
# MEMORY_AND_DISK need to be used only when storage required is larger than␣
 ↪available memory
# replication to more cluster notes is also not required
```

0.0.19 s) How would you pivot the data to show PRODUCTLINE as columns and the total SALES for each ORDERDATE as the values? What are the implications of pivoting large datasets in Spark?

[31]:
```
sales_df1.withColumn('ORDERDATE', to_timestamp(col('ORDERDATE'), 'MM/dd/yyyy H:
 ↪mm'))\
.withColumn("ORDERDATE", col("ORDERDATE").cast('date'))\
.groupBy("ORDERDATE")\
.pivot("PRODUCTLINE")\
.agg(round(sum("SALES"), 3))\
.orderBy("ORDERDATE")\
.show()

# performing pivot operations on lagre datasets can cause increased memory␣
 ↪consumption
# and higher processing time.
# number of new columns in the new dataframe will depend on unique values in␣
 ↪the pivot column
```

```
+----------+------------+-----------+--------+--------+-------+----------------+
------------+
| ORDERDATE|Classic Cars|Motorcycles|  Planes|   Ships| Trains|Trucks and
Buses|Vintage Cars|
+----------+------------+-----------+--------+--------+-------+----------------+
------------+
|2003-01-06|        null|       null|    null|    null|   null|            null|
12133.25|
|2003-01-09|        null|       null|    null|    null|   null|            null|
```

```
11432.34|
|2003-01-10|        null|       null|    null|    null|   null|            null|
6864.05|
|2003-01-29|     15263.7|       null|    null|    null|   null|          23041.1|
16397.2|
|2003-01-31|    25928.08|       null|    null|    null|4933.55|         13760.33|
null|
|2003-02-11|    20464.41|       null|    null|20452.04| 4330.1|            null|
13624.56|
|2003-02-17|        null|       null|39205.31| 6598.34|   null|            null|
10377.67|
|2003-02-24|        null|   25783.76|    null|    null|   null|            null|
null|
|2003-03-03|    42605.87|   12639.15|    null|    null|   null|            null|
null|
|2003-03-10|    27398.82|       null|    null|    null|   null|            null|
null|
|2003-03-18|    27812.88|       null|    null|    null|   null|            null|
23205.04|
|2003-03-24|     7209.11|       null|    null|    null|   null|            null|
2539.89|
|2003-03-25|        null|       null|    null|    null|   null|            null|
18695.58|
|2003-03-26|        null|       null|    null|    null|   null|          9908.06|
2490.5|
|2003-04-01|    17994.34|       null|    null|    null|   null|         20223.07|
null|
|2003-04-04|    16209.72|       null|    null|    null|   null|          8567.69|
null|
|2003-04-11|        null|       null|    null|    null|1711.26|            null|
null|
|2003-04-16|    20664.74|       null|    null|14155.52|3045.21|            null|
5792.0|
|2003-04-21|        null|       null|    null|  4219.2|   null|            null|
null|
|2003-04-28|      5004.8|       null|14216.32| 9024.73|   null|            null|
10383.29|
+----------+------------+-----------+--------+--------+-------+----------------+
------------+
only showing top 20 rows
```

0.0.20 t) How would you calculate the percentage growth of total sales month over month for each PRODUCTLINE using Spark DataFrame?

[32]:
```
from pyspark.sql.window import Window
from pyspark.sql.functions import lag

sales_df1\
.withColumn('ORDERDATE', to_timestamp(col('ORDERDATE'), 'MM/dd/yyyy H:mm'))\
.select(
    col('PRODUCTLINE'), year('ORDERDATE').alias('YEAR'), month('ORDERDATE').
 ↪alias('MONTH'), col('SALES'))\
.groupBy(["PRODUCTLINE", "YEAR", "MONTH"]).agg(round(sum("SALES"), 3).
 ↪alias("SALES"))\
.withColumn(
    'LAG',
    lag('SALES', 1).over(Window.partitionBy("PRODUCTLINE").
 ↪orderBy("PRODUCTLINE", "YEAR", "MONTH")))\
.withColumn(
    'PERCENTAGE_GROWTH',
    round((col("SALES")-col("LAG"))*100/col("LAG"), 3).
 ↪alias('PERCENTAGE_GROWTH'))\
.show(35)
```

```
+------------+----+-----+---------+---------+-----------------+
| PRODUCTLINE|YEAR|MONTH|    SALES|      LAG|PERCENTAGE_GROWTH|
+------------+----+-----+---------+---------+-----------------+
| Motorcycles|2003|    2| 25783.76|     null|             null|
| Motorcycles|2003|    3| 12639.15| 25783.76|           -50.98|
| Motorcycles|2003|    4| 23475.59| 12639.15|           85.737|
| Motorcycles|2003|    5| 22097.32| 23475.59|           -5.871|
| Motorcycles|2003|    6|  2642.01| 22097.32|          -88.044|
| Motorcycles|2003|    7| 37924.23|  2642.01|         1335.431|
| Motorcycles|2003|    8| 44164.91| 37924.23|           16.456|
| Motorcycles|2003|    9|  3155.58| 44164.91|          -92.855|
| Motorcycles|2003|   10| 64235.65|  3155.58|         1935.621|
| Motorcycles|2003|   11| 109345.5| 64235.65|           70.226|
| Motorcycles|2003|   12| 25431.88| 109345.5|          -76.742|
| Motorcycles|2004|    1| 41200.52| 25431.88|           62.003|
| Motorcycles|2004|    2|  49066.5| 41200.52|           19.092|
| Motorcycles|2004|    4| 36269.07|  49066.5|          -26.082|
| Motorcycles|2004|    5| 46848.95| 36269.07|           29.171|
| Motorcycles|2004|    6| 47237.41| 46848.95|            0.829|
| Motorcycles|2004|    7|  22774.0| 47237.41|          -51.788|
| Motorcycles|2004|    8| 62704.93|  22774.0|          175.336|
| Motorcycles|2004|    9| 42471.05| 62704.93|          -32.268|
| Motorcycles|2004|   10| 39413.96| 42471.05|           -7.198|
| Motorcycles|2004|   11|151711.86| 39413.96|          284.919|
```

```
| Motorcycles|2004|   12| 20846.98|151711.86|          -86.259|
| Motorcycles|2005|    1| 39913.36| 20846.98|           91.459|
| Motorcycles|2005|    2| 47951.42| 39913.36|           20.139|
| Motorcycles|2005|    3| 47830.83| 47951.42|           -0.251|
| Motorcycles|2005|    4| 59862.22| 47830.83|           25.154|
| Motorcycles|2005|    5|  39389.7| 59862.22|          -34.199|
|Vintage Cars|2003|    1| 46826.84|     null|             null|
|Vintage Cars|2003|    2| 24002.23| 46826.84|          -48.743|
|Vintage Cars|2003|    3| 46931.01| 24002.23|           95.528|
|Vintage Cars|2003|    4| 20750.34| 46931.01|          -55.785|
|Vintage Cars|2003|    5| 47033.58| 20750.34|          126.664|
|Vintage Cars|2003|    6| 26582.51| 47033.58|          -43.482|
|Vintage Cars|2003|    7| 29652.09| 26582.51|           11.547|
|Vintage Cars|2003|    8| 23285.46| 29652.09|          -21.471|
+------------+----+-----+---------+---------+-----------------+
only showing top 35 rows
```

0.0.21 u) How can you rebalance the data by partioning based on the COUNTRY column to ensure that large data partitions are avoided?

[33]:
```
sales_df1_repartitioned = sales_df1.repartition(col("COUNTRY"))
sales_df1_repartitioned.rdd.getNumPartitions()

# repartition() helps distribute the dataframe across different partitions,␣
 ↪thereby balancing
# the distribution of data. This helps improve parallel processing
```

[33]: 200

0.0.22 v) Suppose you have a smaller lookup table with customer details. How would you perform a broadcast join with the large sales_data_sample dataset to improve join performance? What are the key considerations when using broadcast joins?

[34]:
```
from pyspark.sql.functions import broadcast


customer_lookup_table = spark.read.csv('file:///home/hadoop/Downloads/
 ↪sales_data_customer_demographic.csv', header=True, inferSchema=True)

joined_df = sales_df1.join(broadcast(customer_lookup_table), sales_df1.
 ↪CUSTOMERNAME==customer_lookup_table.CUSTOMERNAME, "inner")
joined_df.show()
```

```
# key considerations when using broadcast join
# - one of the dataframes must be small enough to fit in the memory of each␣
 ↪worker node
# it makes the join operation much less resource intensive
```

```
+----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+----------+----+----------+--------------------
+----------------+--------------------+-----------+-------------+--------+-----
-----+---------+---------+---------------+---------------+--------+-----------
--------+---------+-------------+
|ORDERNUMBER|QUANTITYORDERED|PRICEEACH|ORDERLINENUMBER|  SALES|      ORDERDATE|
STATUS|QTR_ID|MONTH_ID|YEAR_ID|PRODUCTLINE|MSRP|PRODUCTCODE|
CUSTOMERNAME|           PHONE|        ADDRESSLINE1|ADDRESSLINE2|         CITY|
STATE|POSTALCODE|  COUNTRY|TERRITORY|CONTACTLASTNAME|CONTACTFIRSTNAME|DEALSIZE|
CUSTOMERNAME|ESTD_YEAR|NUM_EMPLOYEES|
+----------+--------------+---------+--------------+-------+--------------+-
------+------+--------+-------+----------+----+----------+--------------------
+----------------+--------------------+-----------+-------------+--------+-----
-----+---------+---------+---------------+---------------+--------+-----------
--------+---------+-------------+
|      10107|             30|     95.7|             2| 2871.0| 2/24/2003
0:00|Shipped|     1|       2|   2003|Motorcycles|  95|   S10_1678|   Land of
Toys Inc.|      2125557818|897 Long Airport …|            |          NYC|
NY|     10022|      USA|       NA|             Yu|            Kwai|   Small|
Land of Toys Inc.|     2009|           75|
|      10121|             34|    81.35|             5| 2765.9|  5/7/2003
0:00|Shipped|     2|       5|   2003|Motorcycles|  95|   S10_1678|  Reims
Collectables|      26.47.1555|  59 rue de l'Abbaye|            |        Reims|
|     51100|   France|     EMEA|        Henriot|            Paul|   Small|
Reims Collectables|     2015|           70|
|      10134|             41|    94.74|             2|3884.34|  7/1/2003
0:00|Shipped|     3|       7|   2003|Motorcycles|  95|   S10_1678|     Lyon
Souveniers|+33 1 46 62 7555|27 rue du Colonel…|            |        Paris|
|     75508|   France|     EMEA|       Da Cunha|          Daniel|  Medium|
Lyon Souveniers|     1995|           60|
|      10145|             45|    83.26|             6| 3746.7| 8/25/2003
0:00|Shipped|     3|       8|   2003|Motorcycles|  95|   S10_1678|
Toys4GrownUps.com|      6265557265|  78934 Hillside Dr.|            |
Pasadena|      CA|     90003|      USA|       NA|          Young|
Julie|  Medium|   Toys4GrownUps.com|     2008|          120|
|      10159|             49|    100.0|            14|5205.27|10/10/2003
0:00|Shipped|     4|      10|   2003|Motorcycles|  95|   S10_1678|Corporate Gift
Id…|      6505551386|     7734 Strong St.|            |San Francisco|      CA|
|      USA|       NA|          Brown|           Julie|  Medium|Corporate Gift
Id…|     1999|           50|
|      10168|             36|    96.66|             1|3479.76|10/28/2003
```

0:00|Shipped| 4| 10| 2003|Motorcycles| 95| S10_1678|Technics
Stores Inc.| 6505556809| 9408 Furth Circle| | Burlingame|
CA| 94217| USA| NA| Hirano| Juri|
Medium|Technics Stores Inc.| 2010| 80|
| 10180| 29| 86.13| 9|2497.77|11/11/2003
0:00|Shipped| 4| 11| 2003|Motorcycles| 95| S10_1678|Daedalus
Designs …| 20.16.1555|184, chausse de T…| | Lille|
| 59000| France| EMEA| Rance| Martine|
Small|Daedalus Designs …| 2002| 100|
| 10188| 48| 100.0| 1|5512.32|11/18/2003
0:00|Shipped| 4| 11| 2003|Motorcycles| 95| S10_1678| Herkku
Gifts| +47 2267 3215|Drammen 121, PR 7…| | Bergen| |
N 5804| Norway| EMEA| Oeztan| Veysel| Medium|
Herkku Gifts| 2000| 90|
| 10201| 22| 98.57| 2|2168.54| 12/1/2003
0:00|Shipped| 4| 12| 2003|Motorcycles| 95| S10_1678| Mini
Wheels Co.| 6505555787|5557 North Pendal…| |San Francisco|
CA| | USA| NA| Murphy| Julie| Small|
Mini Wheels Co.| 2009| 35|
| 10211| 41| 100.0| 14|4708.44| 1/15/2004
0:00|Shipped| 1| 1| 2004|Motorcycles| 95| S10_1678| Auto Canal
Petit| (1) 47.55.6555| 25, rue Lauriston| | Paris| |
75016| France| EMEA| Perrier| Dominique| Medium| Auto
Canal Petit| 2002| 55|
| 10223| 37| 100.0| 1|3965.66| 2/20/2004
0:00|Shipped| 1| 2| 2004|Motorcycles| 95| S10_1678|Australian
Collec…| 03 9520 4555| 636 St Kilda Road| Level 3|
Melbourne|Victoria| 3004|Australia| APAC| Ferguson|
Peter| Medium|Australian Collec…| 2007| 60|
| 10237| 23| 100.0| 7|2333.12| 4/5/2004
0:00|Shipped| 2| 4| 2004|Motorcycles| 95| S10_1678|
Vitachrome Inc.| 2125551500| 2678 Kingston Rd.| Suite 101|
NYC| NY| 10022| USA| NA| Frick| Michael|
Small| Vitachrome Inc.| 2015| 25|
| 10251| 28| 100.0| 2|3188.64| 5/18/2004
0:00|Shipped| 2| 5| 2004|Motorcycles| 95| S10_1678|Tekni
Collectable…| 2015559350| 7476 Moss Rd.| | Newark|
NJ| 94019| USA| NA| Brown| William|
Medium|Tekni Collectable…| 2007| 85|
| 10263| 34| 100.0| 2|3676.76| 6/28/2004
0:00|Shipped| 2| 6| 2004|Motorcycles| 95| S10_1678| Gift
Depot Inc.| 2035552570| 25593 South Bay Ln.| | Bridgewater|
CT| 97562| USA| NA| King| Julie| Medium|
Gift Depot Inc.| 2002| 80|
| 10275| 45| 92.83| 1|4177.35| 7/23/2004
0:00|Shipped| 3| 7| 2004|Motorcycles| 95| S10_1678| La Rochelle
Gifts| 40.67.8555|67, rue des Cinqu…| | Nantes| |
44000| France| EMEA| Labrune| Janine| Medium| La

```
Rochelle Gifts|     1999|           55|
|      10285|             36|    100.0|              6|4099.68| 8/27/2004
0:00|Shipped|     3|       8|   2004|Motorcycles|  95|   S10_1678|Marta's
Replicas Co.|      6175558555| 39323 Spinnaker Dr.|            |    Cambridge|
MA|     51247|      USA|       NA|      Hernandez|           Marta|
Medium|Marta's Replicas Co.|     2009|           45|
|      10299|             23|    100.0|              9|2597.39| 9/30/2004
0:00|Shipped|     3|       9|   2004|Motorcycles|  95|   S10_1678|Toys of
Finland, Co.|     90-224 8555|       Keskuskatu 45|            |     Helsinki|
|     21240|  Finland|     EMEA|      Karttunen|           Matti|   Small|Toys
of Finland, Co.|     2014|           50|
|      10309|             41|    100.0|              5|4394.38|10/15/2004
0:00|Shipped|     4|      10|   2004|Motorcycles|  95|   S10_1678|  Baane Mini
Imports|      07-98 9555|Erling Skakkes ga…|            |      Stavern|
|      4110|   Norway|     EMEA|     Bergulfsen|           Jonas|  Medium|
Baane Mini Imports|     1997|           60|
|      10318|             46|    94.74|              1|4358.04| 11/2/2004
0:00|Shipped|     4|      11|   2004|Motorcycles|  95|   S10_1678|Diecast
Classics …|      2155551555|    7586 Pompton St.|            |    Allentown|
PA|     70267|      USA|       NA|             Yu|           Kyung|
Medium|Diecast Classics …|     2006|           65|
|      10329|             42|    100.0|              1|4396.14|11/15/2004
0:00|Shipped|     4|      11|   2004|Motorcycles|  95|   S10_1678|   Land of
Toys Inc.|      2125557818|897 Long Airport …|            |          NYC|
NY|     10022|      USA|       NA|             Yu|            Kwai|  Medium|
Land of Toys Inc.|     2009|           75|
+----------+--------------+---------+---------------+-------+---------------+-
------+------+--------+-------+-----------+----+-----------+--------------------
+----------------+--------------------+------------+-------------+--------+-----
-----+---------+---------+---------------+----------------+--------+------------
--------+---------+-------------+
only showing top 20 rows
```

0.0.23 w) Create a UDF that categorizes the sales values (SALES) into custom buckets like “Low”, “Medium”, “High”. Apply this UDF to the DataFrame and calculate the count of orders in each category per COUNTRY.

[35]:
```
from pyspark.sql.functions import col, udf
from pyspark.sql.types import StringType


q33 = sales_df1.approxQuantile("SALES", [0.33], relativeError=0.0001)[0]
q66 = sales_df1.approxQuantile("SALES", [0.66], relativeError=0.0001)[0]

def categorizeSales(sales):
```

```
    if sales <= q33:
        return 'Low'
    elif sales <= q66:
        return 'Medium'
    else:
        return 'High'

# register UDF
udf_categorizeSales = udf(categorizeSales, StringType())

sales_df1.withColumn("SALES_CATEGORY", udf_categorizeSales(col("SALES")))\
.groupBy("COUNTRY")\
.pivot("SALES_CATEGORY")\
.agg(count("SALES_CATEGORY"))\
.orderBy("COUNTRY")\
.show()
```

```
+-----------+----+---+------+
|    COUNTRY|High|Low|Medium|
+-----------+----+---+------+
|  Australia|  61| 67|    57|
|    Austria|  20| 16|    19|
|    Belgium|  11| 14|     8|
|     Canada|  18| 28|    24|
|    Denmark|  23| 18|    22|
|    Finland|  32| 25|    35|
|     France| 102|112|   100|
|    Germany|  21| 22|    19|
|    Ireland|   8|  7|     1|
|      Italy|  32| 38|    43|
|      Japan|  15| 16|    21|
|     Norway|  34| 30|    21|
|Philippines|   9|  7|    10|
|  Singapore|  30| 28|    21|
|      Spain| 119|112|   111|
|     Sweden|  20| 16|    21|
|Switzerland|  14|  5|    12|
|         UK|  36| 53|    55|
|        USA| 354|318|   332|
+-----------+----+---+------+
```

0.0.24 x) Create a Python UDF to calculate discounts for specific product lines. For example, give a 10% discount for Classic Cars and 5% for Motorcycles. Apply this UDF to derive new discounted sales values.

[39]:
```
from pyspark.sql.functions import col, udf
from pyspark.sql.types import DoubleType


def discountSales(productLine):
    productLine = productLine.lower()
    if productLine == 'classic cars': return 0.10
    elif productLine == 'motorcycles': return 0.05
    elif productLine == 'planes': return 0.20
    elif productLine == 'ships': return 0.25
    elif productLine == 'trains': return 0.30
    elif productLine == 'trucks and buses': return 0.35
    elif productLine == 'vintage cars': return 0.40
    else: return 0.0

# register UDF
udf_discountSales = udf(discountSales, DoubleType())

sales_df1.withColumn("SALES_DISCOUNT", udf_discountSales(col("PRODUCTLINE")))\
.select("ORDERNUMBER", "CUSTOMERNAME", "PRODUCTLINE", "SALES",␣
 ↪"SALES_DISCOUNT")\
.withColumn("DISCOUNTED_SALES", round(col("SALES")*(1-col("SALES_DISCOUNT")),␣
 ↪3))\
.show()
```

```
+----------+--------------------+----------+-------+--------------+----------
-----+
|ORDERNUMBER|        CUSTOMERNAME|PRODUCTLINE|
SALES|SALES_DISCOUNT|DISCOUNTED_SALES|
+----------+--------------------+----------+-------+--------------+----------
-----+
|      10107|   Land of Toys Inc.|Motorcycles| 2871.0|          0.05|
2727.45|
|      10121|  Reims Collectables|Motorcycles| 2765.9|          0.05|
2627.605|
|      10134|     Lyon Souveniers|Motorcycles|3884.34|          0.05|
3690.123|
|      10145|   Toys4GrownUps.com|Motorcycles| 3746.7|          0.05|
3559.365|
|      10159|Corporate Gift Id…|Motorcycles|5205.27|          0.05|
4945.007|
|      10168|Technics Stores Inc.|Motorcycles|3479.76|          0.05|
3305.772|
```

```
|      10180|Daedalus Designs …|Motorcycles|2497.77|          0.05|
2372.882|
|      10188|        Herkku Gifts|Motorcycles|5512.32|          0.05|
5236.704|
|      10201|     Mini Wheels Co.|Motorcycles|2168.54|          0.05|
2060.113|
|      10211|    Auto Canal Petit|Motorcycles|4708.44|          0.05|
4473.018|
|      10223|Australian Collec…|Motorcycles|3965.66|          0.05|
3767.377|
|      10237|     Vitachrome Inc.|Motorcycles|2333.12|          0.05|
2216.464|
|      10251|Tekni Collectable…|Motorcycles|3188.64|          0.05|
3029.208|
|      10263|     Gift Depot Inc.|Motorcycles|3676.76|          0.05|
3492.922|
|      10275|   La Rochelle Gifts|Motorcycles|4177.35|          0.05|
3968.483|
|      10285|Marta's Replicas Co.|Motorcycles|4099.68|          0.05|
3894.696|
|      10299|Toys of Finland, Co.|Motorcycles|2597.39|          0.05|
2467.52|
|      10309|  Baane Mini Imports|Motorcycles|4394.38|          0.05|
4174.661|
|      10318|Diecast Classics …|Motorcycles|4358.04|          0.05|
4140.138|
|      10329|   Land of Toys Inc.|Motorcycles|4396.14|          0.05|
4176.333|
+-----------+--------------------+-----------+-------+--------------+-----------
-----+
only showing top 20 rows
```

0.0.25 y) How would you set up an incremental loading mechanism for orders placed daily based on the ORDERDATE column? How can Spark checkpointing can be used with incremental load to ensure no data loss occurs during failures?

inorder to use incremental loading mechanism for processing orders placed daily (assuming data is sorted in order of ORDERDATE), we first need to keep track of the date till which order details have already been loaded. when there is new data with ORDERDATE greater than the tracked date, they will also need to be loaded and tracked date updated. before an update is done, it is necessary to checkpoint the previous state. this is done inorder to ensure that a recovery operation can be performed incase of any issue with the newly updated data

0.0.26 z) How do you implement a cumulative distribution function (CDF) over the SALES value for each CUSTOMERNAME? What insights can you gather from analyzing the CDF distribution for each customer?

[38]:
```
from pyspark.sql.functions import cume_dist
from pyspark.sql.window import Window

sales_df1\
.withColumn("cume_dist", round(cume_dist().over(Window.
 ↪partitionBy('CUSTOMERNAME').orderBy("SALES")), 5))\
.select("ORDERNUMBER", "CUSTOMERNAME", "SALES", "cume_dist")\
.show(50)

# INSIGHT: for the first row, cume_dist is 0.033 means that for the particular␣
 ↪customer,
# SALES value of that order is higher than 3.3% of all SALES values of the␣
 ↪customer
```

```
+-----------+------------------+-------+---------+
|ORDERNUMBER|       CUSTOMERNAME|  SALES|cume_dist|
+-----------+------------------+-------+---------+
|      10141|Suominen Souveniers| 891.03|  0.03333|
|      10141|Suominen Souveniers| 1086.6|  0.06667|
|      10141|Suominen Souveniers|1103.76|      0.1|
|      10363|Suominen Souveniers|1629.04|  0.13333|
|      10247|Suominen Souveniers| 1988.4|  0.16667|
|      10141|Suominen Souveniers|2140.11|      0.2|
|      10363|Suominen Souveniers|2447.76|  0.23333|
|      10363|Suominen Souveniers|2632.89|  0.26667|
|      10363|Suominen Souveniers| 2773.8|      0.3|
|      10363|Suominen Souveniers|2775.08|  0.33333|
|      10363|Suominen Souveniers|2817.87|  0.36667|
|      10363|Suominen Souveniers|2851.84|      0.4|
|      10363|Suominen Souveniers|2931.98|  0.43333|
|      10247|Suominen Souveniers|3128.65|  0.46667|
|      10363|Suominen Souveniers|3288.82|      0.5|
|      10363|Suominen Souveniers|3595.62|  0.53333|
|      10363|Suominen Souveniers|3686.54|  0.56667|
|      10141|Suominen Souveniers| 3784.8|      0.6|
|      10363|Suominen Souveniers| 4068.7|  0.63333|
|      10363|Suominen Souveniers|4142.64|  0.66667|
|      10247|Suominen Souveniers|4157.73|      0.7|
|      10247|Suominen Souveniers|4381.25|  0.73333|
|      10141|Suominen Souveniers| 4836.5|  0.76667|
|      10363|Suominen Souveniers|5154.41|      0.8|
|      10141|Suominen Souveniers|5500.44|  0.83333|
|      10141|Suominen Souveniers|5938.53|  0.86667|
```

```
|      10141|Suominen Souveniers|6287.66|      0.9|
|      10363|Suominen Souveniers| 6576.5|  0.93333|
|      10247|Suominen Souveniers| 6756.0|  0.96667|
|      10247|Suominen Souveniers|10606.2|      1.0|
|      10280| Amica Models & Co.|  577.6|  0.03846|
|      10280| Amica Models & Co.|1381.05|  0.07692|
|      10280| Amica Models & Co.|1557.36|  0.11538|
|      10280| Amica Models & Co.| 1574.0|  0.15385|
|      10280| Amica Models & Co.|1656.69|  0.19231|
|      10293| Amica Models & Co.|1921.92|  0.23077|
|      10293| Amica Models & Co.|2084.81|  0.26923|
|      10280| Amica Models & Co.|2137.05|  0.30769|
|      10293| Amica Models & Co.|2418.24|  0.34615|
|      10280| Amica Models & Co.|2800.08|  0.38462|
|      10293| Amica Models & Co.|2819.28|  0.42308|
|      10293| Amica Models & Co.|2941.89|  0.46154|
|      10280| Amica Models & Co.|2954.53|      0.5|
|      10280| Amica Models & Co.|3006.43|  0.53846|
|      10280| Amica Models & Co.|3474.46|  0.57692|
|      10280| Amica Models & Co.| 3668.6|  0.61538|
|      10280| Amica Models & Co.|3704.05|  0.65385|
|      10293| Amica Models & Co.|4242.24|  0.69231|
|      10280| Amica Models & Co.| 4455.0|  0.73077|
|      10280| Amica Models & Co.| 4750.8|  0.76923|
+-----------+-------------------+-------+---------+
only showing top 50 rows
```