```
sc.stop()
```

a) Create a new Spark Session with new SparkConfig

```
from pyspark import SparkConf, SparkContext
config = SparkConf().setMaster("local[4]").setAppName("Sales-
Analysis")
sc = SparkContext(conf=config)
```

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

```
from pyspark.sql import SparkSession
spark =
SparkSession.builder.appName("AssignmentSession").getOrCreate()

spark
```

```
<pyspark.sql.session.SparkSession at 0x7f42873fa128>
```

```
sales_data =
spark.read.csv('file:///home/hadoop/Downloads/sales_data_sample.csv',
                           header=True,
                           inferSchema=True)
```

c) Find the shape of DataFrame.

```
print("Number of rows:", sales_data.count())
print("Number of cols:", len(sales_data.columns))
```

```
Number of rows: 2823
Number of cols: 25
```

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

```
sales_data.schema.fields
```

```
[StructField(ORDERNUMBER,IntegerType,true),
 StructField(QUANTITYORDERED,IntegerType,true),
 StructField(PRICEEACH,DoubleType,true),
 StructField(ORDERLINENUMBER,IntegerType,true),
 StructField(SALES,DoubleType,true),
 StructField(ORDERDATE,StringType,true),
 StructField(STATUS,StringType,true),
 StructField(QTR_ID,IntegerType,true),
 StructField(MONTH_ID,IntegerType,true),
 StructField(YEAR_ID,IntegerType,true),
```

```
 StructField(PRODUCTLINE,StringType,true),
 StructField(MSRP,IntegerType,true),
 StructField(PRODUCTCODE,StringType,true),
 StructField(CUSTOMERNAME,StringType,true),
 StructField(PHONE,StringType,true),
 StructField(ADDRESSLINE1,StringType,true),
 StructField(ADDRESSLINE2,StringType,true),
 StructField(CITY,StringType,true),
 StructField(STATE,StringType,true),
 StructField(POSTALCODE,StringType,true),
 StructField(COUNTRY,StringType,true),
 StructField(TERRITORY,StringType,true),
 StructField(CONTACTLASTNAME,StringType,true),
 StructField(CONTACTFIRSTNAME,StringType,true),
 StructField(DEALSIZE,StringType,true)]
```

```
from pyspark.sql.types import StringType

numerical_columns = [field.name for field in
                     sales_data.schema.fields
                     if not isinstance(field.dataType,StringType)]

numerical_columns
```

```
['ORDERNUMBER',
 'QUANTITYORDERED',
 'PRICEEACH',
 'ORDERLINENUMBER',
 'SALES',
 'QTR_ID',
 'MONTH_ID',
 'YEAR_ID',
 'MSRP']
```

```
sales_data.select(numerical_columns).describe().show()
```

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

```
7.0924548352816155|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|
|    max|             10425|               97|             100.0|
18|           14082.8|                4|               12|
2005|              214|
+-------+------------------+-----------------+------------------
+-----------------+------------------+------------------
+------------------+------------------+------------------+
```

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

```
from pyspark.sql.functions import col, sum

sales_data.select([sum(col(c).isNull().cast("int"))\
                                .alias(c) for c in
sales_data.columns]).collect()


[Row(ORDERNUMBER=0, QUANTITYORDERED=0, PRICEEACH=0, ORDERLINENUMBER=0,
SALES=0, ORDERDATE=0, STATUS=0, QTR_ID=0, MONTH_ID=0, YEAR_ID=0,
PRODUCTLINE=0, MSRP=0, PRODUCTCODE=0, CUSTOMERNAME=0, PHONE=0,
ADDRESSLINE1=0, ADDRESSLINE2=2521, CITY=0, STATE=1486, POSTALCODE=76,
COUNTRY=0, TERRITORY=0, CONTACTLASTNAME=0, CONTACTFIRSTNAME=0,
DEALSIZE=0)]

cleaned_df=sales_data.fillna('')

from pyspark.sql.functions import when,isnull,count
cleaned_df.select([count(when(isnull(column),column)).alias(column)
for column in cleaned_df.columns]).collect()

[Row(ORDERNUMBER=0, QUANTITYORDERED=0, PRICEEACH=0, ORDERLINENUMBER=0,
SALES=0, ORDERDATE=0, STATUS=0, QTR_ID=0, MONTH_ID=0, YEAR_ID=0,
PRODUCTLINE=0, MSRP=0, PRODUCTCODE=0, CUSTOMERNAME=0, PHONE=0,
ADDRESSLINE1=0, ADDRESSLINE2=0, CITY=0, STATE=0, POSTALCODE=0,
COUNTRY=0, TERRITORY=0, CONTACTLASTNAME=0, CONTACTFIRSTNAME=0,
DEALSIZE=0)]
```

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

```
cleaned_df.withColumn("TOTAL_REVENUE",col("QUANTITYORDERED") *
col("PRICEEACH")) \
```

```
   .groupBy("COUNTRY") \
   .agg(sum("TOTAL_REVENUE").alias("TotalRevenue"))\
   .orderBy('TotalRevenue',ascending=False)\
   .show()
```

```
+-----------+------------------+
|    COUNTRY|      TotalRevenue|
+-----------+------------------+
|        USA|2986425.2099999995|
|      Spain|1021705.9700000002|
|     France| 919257.8499999997|
|  Australia|521598.45999999985|
|         UK|413203.33999999997|
|      Italy| 309402.8699999999|
|    Finland|268714.70000000007|
|     Norway| 246115.8000000001|
|  Singapore| 227985.5000000001|
|     Canada|193504.34000000003|
|    Denmark|         192747.63|
|    Germany|         178689.08|
|     Sweden|174264.10000000006|
|    Austria|172793.05000000002|
|      Japan|153076.68999999994|
|    Belgium|          94528.88|
|Switzerland| 93344.90999999999|
|Philippines| 80291.16999999998|
|    Ireland|          43237.24|
+-----------+------------------+
```

Insight: USA has the highest revenue

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

```
cleaned_df.select('PRODUCTCODE', 'SALES') \
    .groupBy('PRODUCTCODE') \
    .agg(round(sum('SALES'),3).alias('SALES')) \
    .orderBy('SALES', ascending=False) \
    .limit(5) \
    .show()
```

```
+-----------+---------+
|PRODUCTCODE|    SALES|
+-----------+---------+
|   S18_3232|288245.42|
|   S10_1949|191073.03|
|   S10_4698|170401.07|
|   S12_1108|168585.32|
```

```
|   S18_2238|154623.95|
+-----------+---------+
```

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

```
from pyspark.sql.functions import *
cleaned_df.groupBy('PRODUCTCODE')\
.agg(round(avg('QUANTITYORDERED'),3).alias('AVG_QTY'))\
.orderBy('AVG_QTY',ascending=False)\
.show()
```

```
+-----------+-------+
|PRODUCTCODE|AVG_QTY|
+-----------+-------+
|   S24_3856| 38.963|
|   S24_2766| 38.696|
|   S50_1341| 38.423|
|   S18_1342| 38.346|
|   S18_3856| 38.346|
|   S18_2319| 38.192|
|   S18_4600| 38.185|
|  S700_4002| 38.111|
|  S700_2610| 38.077|
|   S12_4473| 37.926|
|   S18_3685|  37.92|
|  S700_3167|  37.52|
|   S12_1108| 37.423|
|   S24_3949| 37.333|
|   S18_4721|  37.25|
|   S12_4675| 37.077|
|   S12_2823| 37.077|
|   S24_2011| 36.923|
|   S24_2300| 36.889|
|   S24_2887| 36.818|
+-----------+-------+
only showing top 20 rows
```

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

```
cleaned_df.withColumn('ORDERDATE',to_timestamp(col('ORDERDATE'),'MM/
dd/yyyy H:mm'))\
     .filter(col('SALES').cast('float') > 10000) \
     .orderBy('ORDERDATE') \
     .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|
+-----------+--------------+---------+---------------+-------
+-------------------+----------+------+--------+-------+------------
+----+-----------+--------------------+--------------
+--------------------+------------+-------------+-----+----------
+---------+---------+---------------+----------------+--------+
|      10127|             46|    100.0|              2|11279.2|2003-
06-03 00:00:00|   Shipped|     2|       6|   2003|Classic Cars| 207|
S12_1108|  Muscle Machine Inc|     2125557413|   4092 Furth Circle|
Suite 400|          NYC|   NY|     10022|      USA|       NA|
Young|           Jeff|   Large|
|      10150|             45|    100.0|              8|10993.5|2003-
09-19 00:00:00|   Shipped|     3|       9|   2003|Classic Cars| 214|
S10_1949|Dragon Souveniers...|   +65 221 7555|Bronz Sok., Bronz...|
|    Singapore|     |     79903|Singapore|    Japan|      Natividad|
Eric|   Large|
|      10247|             44|    100.0|              2|10606.2|2004-
05-05 00:00:00|   Shipped|     2|       5|   2004|Classic Cars| 207|
S12_1108| Suominen Souveniers|+358 9 8045 555|Software Engineer...|
|        Espoo|     | FIN-02271|  Finland|     EMEA|       Suominen|
Kalle|   Large|
|      10304|             47|    100.0|              6|10172.7|2004-
10-11 00:00:00|   Shipped|     4|      10|   2004|Classic Cars| 214|
S10_1949|  Auto Assoc. & Cie.|     30.59.8555|67, avenue de l'E...|
|   Versailles|     |     78000|   France|     EMEA|         Tonini|
Daniel|   Large|
|      10312|             48|    100.0|              3|11623.7|2004-
10-21 00:00:00|   Shipped|     4|      10|   2004|Classic Cars| 214|
S10_1949|Mini Gifts Distri...|     4155551450|     5677 Strong St.|
|   San Rafael|   CA|     97562|      USA|       NA|         Nelson|
Valarie|   Large|
|      10322|             50|    100.0|              6|12536.5|2004-
11-04 00:00:00|   Shipped|     4|      11|   2004|Vintage Cars| 127|
S18_2325|Online Diecast Cr...|     6035558647|2304 Long Airport...|
|       Nashua|   NH|     62005|      USA|       NA|          Young|
Valarie|   Large|
|      10333|             46|    100.0|              2|11336.7|2004-
11-18 00:00:00|   Shipped|     4|      11|   2004|Vintage Cars|  99|
S18_3320|     Mini Wheels Co.|     6505555787|5557 North Pendal...|
|San Francisco|   CA|          |      USA|       NA|         Murphy|
Julie|   Large|
```

```
|      10339|             55|    100.0|             13|10758.0|2004-
11-23 00:00:00|   Shipped|     4|      11|   2004|Vintage Cars|  88|
S24_3151|Tokyo Collectable...|+81 3 3584 0555|      2-2-8 Roppongi|
|    Minato-ku|Tokyo|  106-0032|    Japan|    Japan|      Shimamura|
Akiko|   Large|
|      10375|             43|    100.0|              2|10039.6|2005-
02-03 00:00:00|   Shipped|     1|       2|   2005|      Planes|  72|
S24_4278|   La Rochelle Gifts|     40.67.8555|67, rue des Cinqu...|
|       Nantes|      |     44000|   France|     EMEA|        Labrune|
Janine|   Large|
|      10388|             46|    100.0|              2|10066.6|2005-
03-03 00:00:00|   Shipped|     1|       3|   2005|      Planes|  91|
S700_1691|    FunGiftIdeas.com|     5085552555|   1785 First Street|
|  New Bedford|   MA|     50553|      USA|       NA|        Benitez|
Violeta|   Large|
|      10403|             66|    100.0|              9|11886.6|2005-
04-08 00:00:00|   Shipped|     2|       4|   2005| Motorcycles| 193|
S10_4698|UK Collectables, ...| (171) 555-2282|Berkeley Gardens ...|
|    Liverpool|      |   WX1 6LT|       UK|     EMEA|          Devon|
Elizabeth|   Large|
|      10405|             76|    100.0|              3|11739.7|2005-
04-14 00:00:00|   Shipped|     2|       4|   2005|Classic Cars| 140|
S24_3856|         Mini Caravy|     88.60.1555|    24, place Kluber|
|   Strasbourg|      |     67000|   France|     EMEA|        Citeaux|
Frederique|   Large|
|      10406|             65|    100.0|              1|10468.9|2005-
04-15 00:00:00|  Disputed|     2|       4|   2005|Classic Cars| 141|
S18_3685|Danish Wholesale ...|     31 12 3555|        Vinb'ltet 34|
|    Kobenhavn|      |      1734|  Denmark|     EMEA|       Petersen|
Jytte|   Large|
|      10407|             76|    100.0|              2|14082.8|2005-
04-22 00:00:00|   On Hold|     2|       4|   2005|Vintage Cars| 170|
S18_1749|The Sharp Gifts W...|     4085553659|      3086 Ingle Ln.|
|     San Jose|   CA|     94217|      USA|       NA|          Frick|
Sue|   Large|
|      10412|             60|    100.0|              9|11887.8|2005-
05-03 00:00:00|   Shipped|     2|       5|   2005|Classic Cars| 169|
S18_3232|Euro Shopping Cha...| (91) 555 94 44|  C/ Moralzarzal, 86|
|       Madrid|      |     28034|    Spain|     EMEA|         Freyre|
Diego|   Large|
|      10424|             50|    100.0|              6|12001.0|2005-
05-31 00:00:00|In Process|     2|       5|   2005|Classic Cars| 214|
S10_1949|Euro Shopping Cha...| (91) 555 94 44|  C/ Moralzarzal, 86|
|       Madrid|      |     28034|    Spain|     EMEA|         Freyre|
Diego|   Large|
+-----------+--------------+---------+---------------+-------
+-------------------+----------+------+--------+-------+------------
+----+-----------+--------------------+---------------
+--------------------+------------+-------------+-----+----------
```

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

j) Filter out rows where the STATUS is Cancelled; and calculate the total sales from the remaining orders?

```
cleaned_df\
.filter(col('STATUS') != 'Cancelled')\
.agg(sum('SALES').alias('TOTAL_SALES'))\
.show()
```

```
+-----------------+
|      TOTAL_SALES|
+-----------------+
|9838141.370000018|
+-----------------+
```

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

```
from pyspark.sql.functions import to_date, year, to_timestamp

yearly_data = cleaned_df\
.withColumn("YEAR", year(to_timestamp(col("ORDERDATE"), 'M/d/yyyy
H:mm')))

yearly_data.groupBy(['CUSTOMERNAME', 'YEAR'])\
.agg(sum('SALES').alias('TOTAL_SALES')).orderBy('CUSTOMERNAME','YEAR')
.show()
```

```
+--------------------+----+------------------+
|        CUSTOMERNAME|YEAR|       TOTAL_SALES|
+--------------------+----+------------------+
|      AV Stores, Co.|2003| 51017.91999999999|
|      AV Stores, Co.|2004|         106789.89|
|        Alpha Cognac|2003| 55349.31999999999|
|        Alpha Cognac|2005|15139.119999999999|
|  Amica Models & Co.|2004| 94117.26000000002|
|Anna's Decoration...|2003| 88983.70999999999|
|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|60135.840000000004|
|Australian Collec...|2004|140859.56999999998|
|Australian Gift N...|2003|37739.090000000004|
```

```
|Australian Gift N...|2005|          21730.03|
|  Auto Assoc. & Cie.|2004| 64834.32000000001|
|    Auto Canal Petit|2004| 79103.85999999999|
|    Auto Canal Petit|2005|           14066.8|
|Auto-Moto Classic...|2003|           7277.35|
+--------------------+----+------------------+
only showing top 20 rows
```

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

```
#from pyspark.sql.functions import *

percentile_33, percentile_67 = cleaned_df\
.approxQuantile("SALES", [0.33, 0.66], 0.01)

sales_data_with_category = cleaned_df.withColumn(
    "CATEGORY",
    when(col("SALES") > percentile_67, "High")\
    .when(col("SALES") > percentile_33, "Medium")
    .otherwise("Low")
)


sales_data_with_category.select(['CUSTOMERNAME','SALES',
'CATEGORY']).show(20)
```

```
+--------------------+-------+--------+
|        CUSTOMERNAME|  SALES|CATEGORY|
+--------------------+-------+--------+
|   Land of Toys Inc.| 2871.0|  Medium|
|  Reims Collectables| 2765.9|  Medium|
|     Lyon Souveniers|3884.34|  Medium|
|   Toys4GrownUps.com| 3746.7|  Medium|
|Corporate Gift Id...|5205.27|    High|
|Technics Stores Inc.|3479.76|  Medium|
|Daedalus Designs ...|2497.77|  Medium|
|        Herkku Gifts|5512.32|    High|
|     Mini Wheels Co.|2168.54|     Low|
|    Auto Canal Petit|4708.44|    High|
|Australian Collec...|3965.66|    High|
|     Vitachrome Inc.|2333.12|     Low|
|Tekni Collectable...|3188.64|  Medium|
|     Gift Depot Inc.|3676.76|  Medium|
|   La Rochelle Gifts|4177.35|    High|
|Marta's Replicas Co.|4099.68|    High|
|Toys of Finland, Co.|2597.39|  Medium|
|  Baane Mini Imports|4394.38|    High|
|Diecast Classics ...|4358.04|    High|
```

```
|   Land of Toys Inc.|4396.14|    High|
+--------------------+-------+--------+
only showing top 20 rows
```

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?

```
cleaned_df.select('CUSTOMERNAME','COUNTRY').distinct().show(20)
```

```
+--------------------+-------+
|        CUSTOMERNAME|COUNTRY|
+--------------------+-------+
|Toms Spezialitten...|Germany|
|Oulu Toy Supplies...|Finland|
|          Petit Auto|Belgium|
|Corporate Gift Id...|    USA|
|Cambridge Collect...|    USA|
|Toys of Finland, Co.|Finland|
|  Enaco Distributors|  Spain|
|       Rovelli Gifts|  Italy|
|Tekni Collectable...|    USA|
|     Mini Auto Werke|Austria|
|      AV Stores, Co.|     UK|
|Bavarian Collecta...|Germany|
| L'ordine Souveniers|  Italy|
|        Alpha Cognac| France|
|Salzburg Collecta...|Austria|
|Blauer See Auto, Co.|Germany|
|Iberia Gift Impor...|  Spain|
|Technics Stores Inc.|    USA|
|       Mini Classics|    USA|
|  Baane Mini Imports| Norway|
+--------------------+-------+
only showing top 20 rows
```

```
demographic_data = [

    ("Toms Spezialitten, Ltd", "Germany", 83000000),
    ("Oulu Toy Supplies, Inc.", "Finland", 5500000),
    ("Corporate Gift Ideas Co.", "USA", 331000000),
    ("Tokyo Collectables Limited, Ltd.", "Japan", 126000000),
    ("Royal Canadian Collectables, Ltd.", "Canada", 38000000)

]
```

```
demographic_df = spark.createDataFrame(demographic_data,
["CUSTOMERNAME", "COUNTRY", "POPULATION"])

demographic_df.show(truncate=False)

from pyspark.sql.functions import sum


joined_df = cleaned_df.join(demographic_df,
on=["CUSTOMERNAME",'COUNTRY'], how="inner")

joined_df.select('CUSTOMERNAME','SALES','COUNTRY','POPULATION').show()
sales_by_country =
joined_df.groupBy("COUNTRY","CUSTOMERNAME",'POPULATION').agg(sum("SALE
S").alias("TOTAL_SALES"))


sales_by_country.show()
```

```
+---------------------------------+-------+----------+
|CUSTOMERNAME                     |COUNTRY|POPULATION|
+---------------------------------+-------+----------+
|Toms Spezialitten, Ltd           |Germany|83000000  |
|Oulu Toy Supplies, Inc.          |Finland|5500000   |
|Corporate Gift Ideas Co.         |USA    |331000000 |
|Tokyo Collectables Limited, Ltd. |Japan  |126000000 |
|Royal Canadian Collectables, Ltd.|Canada |38000000  |
+---------------------------------+-------+----------+
```

```
+--------------------+-------+-------+----------+
|        CUSTOMERNAME|  SALES|COUNTRY|POPULATION|
+--------------------+-------+-------+----------+
|Toms Spezialitten...| 1549.8|Germany|  83000000|
|Toms Spezialitten...|4753.49|Germany|  83000000|
|Toms Spezialitten...|2916.76|Germany|  83000000|
|Toms Spezialitten...|5497.65|Germany|  83000000|
|Toms Spezialitten...|2163.72|Germany|  83000000|
|Toms Spezialitten...|2880.48|Germany|  83000000|
|Toms Spezialitten...|1383.03|Germany|  83000000|
|Toms Spezialitten...| 5356.8|Germany|  83000000|
|Toms Spezialitten...|4237.76|Germany|  83000000|
|Toms Spezialitten...|2819.88|Germany|  83000000|
|Toms Spezialitten...|3127.82|Germany|  83000000|
|Toms Spezialitten...|6266.12|Germany|  83000000|
|Toms Spezialitten...|3414.58|Germany|  83000000|
|Toms Spezialitten...|2171.07|Germany|  83000000|
|Toms Spezialitten...|3415.68|Germany|  83000000|
|Toms Spezialitten...|8940.96|Germany|  83000000|
|Toms Spezialitten...|3448.08|Germany|  83000000|
```

```
|Toms Spezialitten...| 1832.6|Germany|  83000000|
|Toms Spezialitten...|6231.91|Germany|  83000000|
|Toms Spezialitten...|2504.75|Germany|  83000000|
+--------------------+-------+-------+----------+
only showing top 20 rows

+-------+--------------------+----------+------------------+
|COUNTRY|        CUSTOMERNAME|POPULATION|       TOTAL_SALES|
+-------+--------------------+----------+------------------+
|    USA|Corporate Gift Id...| 331000000|149882.49999999997|
| Canada|Royal Canadian Co...|  38000000| 74634.84999999999|
|Germany|Toms Spezialitten...|  83000000|         100306.58|
|Finland|Oulu Toy Supplies...|   5500000|         104370.38|
+-------+--------------------+----------+------------------+
```

Insight:Increase in Population has positive influence on Total_SALES

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?

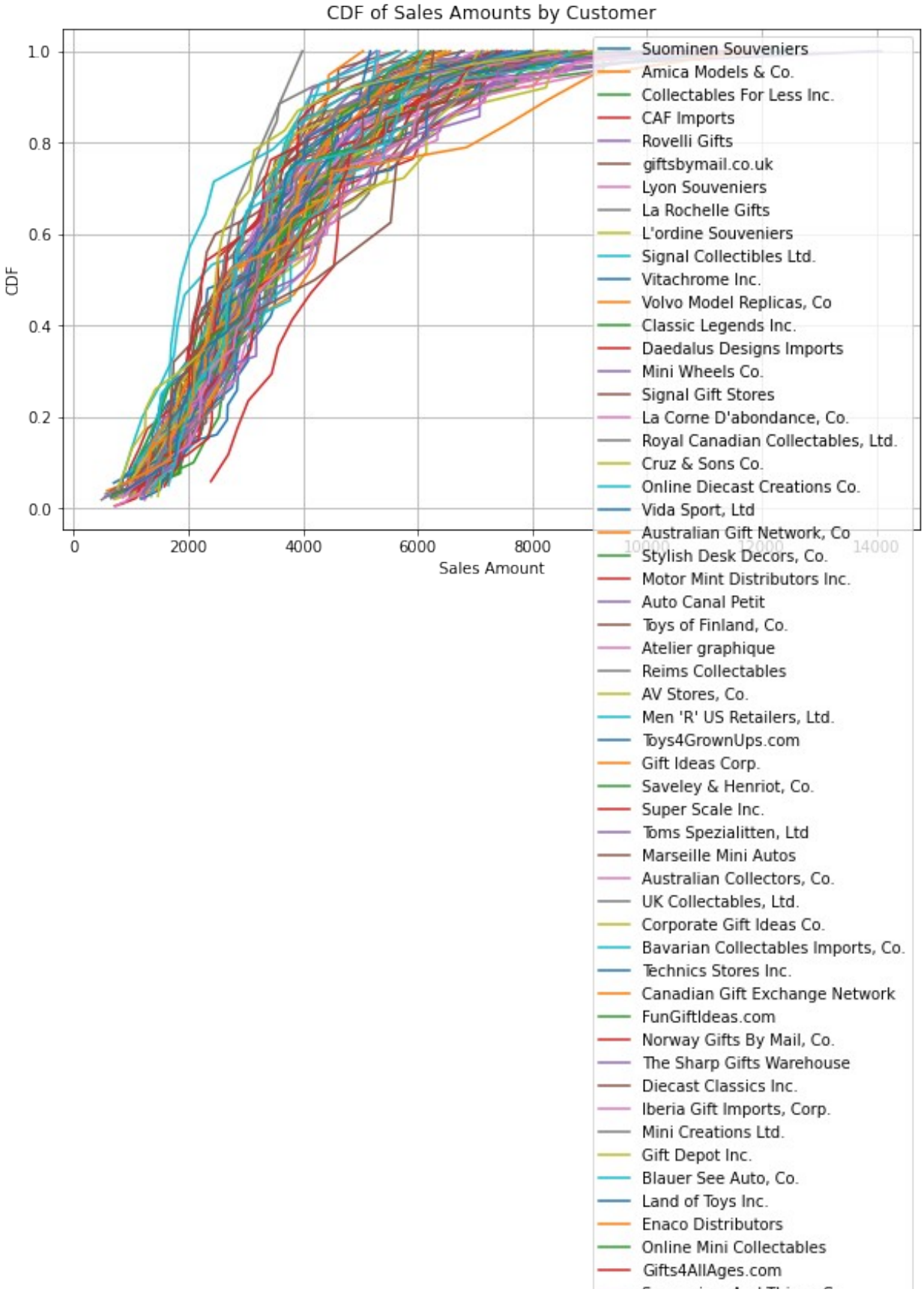
```
from pyspark.sql.window import Window
from pyspark.sql.functions import cume_dist

window = Window.partitionBy("CUSTOMERNAME").orderBy("SALES")
df_cdf=cleaned_df.withColumn("CDF",
cume_dist().over(window)).select("CUSTOMERNAME","SALES","CDF")
df_cdf.show()
```

```
+------------------+-------+------------------+
|      CUSTOMERNAME|  SALES|               CDF|
+------------------+-------+------------------+
|Suominen Souveniers| 891.03|0.03333333333333333|
|Suominen Souveniers| 1086.6|0.06666666666666667|
|Suominen Souveniers|1103.76|               0.1|
|Suominen Souveniers|1629.04|0.13333333333333333|
|Suominen Souveniers| 1988.4|0.16666666666666666|
|Suominen Souveniers|2140.11|               0.2|
|Suominen Souveniers|2447.76|0.23333333333333334|
|Suominen Souveniers|2632.89|0.26666666666666666|
|Suominen Souveniers| 2773.8|               0.3|
|Suominen Souveniers|2775.08| 0.3333333333333333|
|Suominen Souveniers|2817.87|0.36666666666666664|
|Suominen Souveniers|2851.84|               0.4|
|Suominen Souveniers|2931.98|0.43333333333333335|
|Suominen Souveniers|3128.65| 0.4666666666666667|
|Suominen Souveniers|3288.82|               0.5|
|Suominen Souveniers|3595.62| 0.5333333333333333|
|Suominen Souveniers|3686.54| 0.5666666666666667|
```

```
|Suominen Souveniers| 3784.8|                0.6|
|Suominen Souveniers| 4068.7| 0.6333333333333333|
|Suominen Souveniers|4142.64| 0.6666666666666666|
+-------------------+-------+-------------------+
only showing top 20 rows
```

Insight:Using cumulative distribution, we can see that, 50% of Suominen Souveniers sales are below 3288.82.s

```
df_pandas = df_cdf.toPandas()
import matplotlib.pyplot as plt


plt.figure(figsize=(10, 6))

for customer in df_pandas['CUSTOMERNAME'].unique():
    customer_data = df_pandas[df_pandas['CUSTOMERNAME'] == customer]
    plt.plot(customer_data['SALES'], customer_data['CDF'],
label=customer)


plt.xlabel('Sales Amount')
plt.ylabel('CDF')
plt.title('CDF of Sales Amounts by Customer')


plt.grid(True)
plt.legend()


plt.show()

# 50% of sales for majority of customers are below 2500
#The max sale price goes upto 14000
```


CDF of Sales Amounts by Customer
CDF
Sales Amount
0.0
0.2
0.4
0.6
0.8
1.0
0
2000
4000
6000
8000
10000
12000
14000
Suominen Souveniers
Amica Models & Co.
Collectables For Less Inc.
CAF Imports
Rovelli Gifts
giftsbymail.co.uk
Lyon Souveniers
La Rochelle Gifts
L'ordine Souveniers
Signal Collectibles Ltd.
Vitachrome Inc.
Volvo Model Replicas, Co
Classic Legends Inc.
Daedalus Designs Imports
Mini Wheels Co.
Signal Gift Stores
La Corne D'abondance, Co.
Royal Canadian Collectables, Ltd.
Cruz & Sons Co.
Online Diecast Creations Co.
Vida Sport, Ltd
Australian Gift Network, Co
Stylish Desk Decors, Co.
Motor Mint Distributors Inc.
Auto Canal Petit
Toys of Finland, Co.
Atelier graphique
Reims Collectables
AV Stores, Co.
Men 'R' US Retailers, Ltd.
Toys4GrownUps.com
Gift Ideas Corp.
Saveley & Henriot, Co.
Super Scale Inc.
Toms Spezialitten, Ltd
Marseille Mini Autos
Australian Collectors, Co.
UK Collectables, Ltd.
Corporate Gift Ideas Co.
Bavarian Collectables Imports, Co.
Technics Stores Inc.
Canadian Gift Exchange Network
FunGiftIdeas.com
Norway Gifts By Mail, Co.
The Sharp Gifts Warehouse
Diecast Classics Inc.
Iberia Gift Imports, Corp.
Mini Creations Ltd.
Gift Depot Inc.
Blauer See Auto, Co.
Land of Toys Inc.
Enaco Distributors
Online Mini Collectables
Gifts4AllAges.com

Insight:

1. **Distribution of Sales Amounts**: Most customers have small sales amounts (0-4000) around 60% with fewer high-value customers as the sales amounts increase.
2. **Variation in Customer Behavior**: Sales patterns vary widely, with some customers contributing quickly to total sales while others spread purchases over time.
3. **Sales Saturation**: Many customers' total sales reach a cap around 8000-10000.
4. **Outliers**: A few high-value customers extend sales amounts beyond 14,000, deviating from the general trend.
5. **Customer Segmentation**: Customers can be segmented into low, mid, and high-volume groups based on the variation in sales behavior.

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

```
rank = Window.partitionBy('PRODUCTLINE').orderBy(desc('TOTAL_SALES'))
group =
cleaned_df.select('PRODUCTLINE','COUNTRY','SALES').groupBy('PRODUCTLIN
E','COUNTRY')
group = group.agg(round(sum('SALES'),2).alias('TOTAL_SALES'))
group.withColumn("Rank",dense_rank().over(rank)).show()
```

```
+------------+-----------+-----------+----+
| PRODUCTLINE|    COUNTRY|TOTAL_SALES|Rank|
+------------+-----------+-----------+----+
| Motorcycles|        USA|   520371.7|   1|
| Motorcycles|     France|  226390.31|   2|
| Motorcycles|  Australia|   89968.76|   3|
| Motorcycles|      Spain|   74634.82|   4|
| Motorcycles|     Norway|   51768.63|   5|
| Motorcycles|    Finland|   47866.72|   6|
| Motorcycles|         UK|   40802.81|   7|
| Motorcycles|      Japan|   26536.41|   8|
| Motorcycles|    Austria|   26047.66|   9|
| Motorcycles|Philippines|   18061.68|  10|
| Motorcycles|     Sweden|   15567.25|  11|
| Motorcycles|      Italy|     7567.8|  12|
| Motorcycles|    Germany|     7497.5|  13|
| Motorcycles|    Ireland|     4953.2|  14|
| Motorcycles|     Canada|    4177.49|  15|
| Motorcycles|  Singapore|     4175.6|  16|
|Vintage Cars|        USA|   757755.9|   1|
|Vintage Cars|      Spain|  229514.51|   2|
|Vintage Cars|  Australia|  189555.32|   3|
|Vintage Cars|     France|  176609.81|   4|
+------------+-----------+-----------+----+
only showing top 20 rows


Insight: For Motorcycles, USA has highest Total Sales
```

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

```
from pyspark.sql.functions import sum as sum


window_spec =
Window.partitionBy("CUSTOMERNAME").orderBy("ORDERDATE").rowsBetween(Wi
ndow.unboundedPreceding, Window.currentRow)


df_running_total = cleaned_df.withColumn("RUNNING_TOTAL",
sum("SALES").over(window_spec))


df_total_sales =
df_running_total.groupBy("CUSTOMERNAME").agg(sum("RUNNING_TOTAL").alia
s("TOTAL_SALES"))


df_top_5_customers = df_total_sales.orderBy("TOTAL_SALES",
ascending=False).limit(5)


df_top_5_customers.show()
```

```
+--------------------+--------------------+
|        CUSTOMERNAME|         TOTAL_SALES|
+--------------------+--------------------+
|Euro Shopping Cha...|1.1734027052999999E8|
|Mini Gifts Distri...|5.786273786000003E7|
|Australian Collec...| 5886507.8199999975|
|   La Rochelle Gifts|  5040347.829999999|
|  Muscle Machine Inc|          5001260.5|
+--------------------+--------------------+
```

```
Insight: Cumulative sum of Total Sales is highest for Euro Shopping
Channel
```

q) Identify and handle Outliers in DataFrame.

```
# UDF for handling outliers
def handle_outliers(df, column):

    quantiles = df.approxQuantile(column, [0.25, 0.75], 0.01)
    Q1, Q3 = quantiles[0], quantiles[1]

    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
```

```
    upper_bound = Q3 + 1.5 * IQR


    outlier_count = df.filter((col(column) < lower_bound) |
(col(column) > upper_bound)).count()

    capped_df = df.withColumn(
        column,
        when(col(column) < lower_bound, lower_bound)
        .when(col(column) > upper_bound, upper_bound)
        .otherwise(col(column))
    )

    return outlier_count, capped_df



columns = ["SALES", "MSRP", "QUANTITYORDERED", "PRICEEACH"]


capped_sales_data=cleaned_df
for col_name in columns:
    count, capped_sales_data = handle_outliers(capped_sales_data,
col_name)

    print(f"Outliers in {col_name}: {count}")


capped_sales_data.show()
```

```
Outliers in SALES: 88
Outliers in MSRP: 28
Outliers in QUANTITYORDERED: 8
Outliers in PRICEEACH: 0
+-----------+--------------+---------+--------------+-------
+---------------+-------+------+--------+-------+-----------+----
+-----------+--------------------+----------------
+--------------------+------------+-------------+--------+----------
+---------+---------+---------------+-----------------+--------+
|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|           30.0|     95.7|              2| 2871.0|
2/24/2003 0:00|Shipped|     1|       2|   2003|Motorcycles|95.0|
S10_1678|   Land of Toys Inc.|      2125557818|897 Long Airport ...|
|          NYC|      NY|     10022|      USA|       NA|
Yu|            Kwai|   Small|
|      10121|           34.0|    81.35|              5| 2765.9|
5/7/2003 0:00|Shipped|     2|       5|   2003|Motorcycles|95.0|
S10_1678|  Reims Collectables|      26.47.1555|  59 rue de l'Abbaye|
|        Reims|        |     51100|   France|     EMEA|
Henriot|            Paul|   Small|
|      10134|           41.0|    94.74|              2|3884.34|
7/1/2003 0:00|Shipped|     3|       7|   2003|Motorcycles|95.0|
S10_1678|     Lyon Souveniers|+33 1 46 62 7555|27 rue du Colonel...|
|        Paris|        |     75508|   France|     EMEA|       Da
Cunha|          Daniel|  Medium|
|      10145|           45.0|    83.26|              6| 3746.7|
8/25/2003 0:00|Shipped|     3|       8|   2003|Motorcycles|95.0|
S10_1678|   Toys4GrownUps.com|      6265557265|  78934 Hillside Dr.|
|     Pasadena|      CA|     90003|      USA|       NA|
Young|           Julie|  Medium|
|      10159|           49.0|    100.0|             14|5205.27|
10/10/2003 0:00|Shipped|     4|      10|   2003|Motorcycles|95.0|
S10_1678|Corporate Gift Id...|      6505551386|     7734 Strong St.|
|San Francisco|      CA|        |      USA|       NA|
Brown|           Julie|  Medium|
|      10168|           36.0|    96.66|              1|3479.76|
10/28/2003 0:00|Shipped|     4|      10|   2003|Motorcycles|95.0|
S10_1678|Technics Stores Inc.|      6505556809|   9408 Furth Circle|
|   Burlingame|      CA|     94217|      USA|       NA|
Hirano|            Juri|  Medium|
|      10180|           29.0|    86.13|              9|2497.77|
11/11/2003 0:00|Shipped|     4|      11|   2003|Motorcycles|95.0|
S10_1678|Daedalus Designs ...|      20.16.1555|184, chausse de T...|
|        Lille|        |     59000|   France|     EMEA|
Rance|         Martine|   Small|
|      10188|           48.0|    100.0|              1|5512.32|
11/18/2003 0:00|Shipped|     4|      11|   2003|Motorcycles|95.0|
S10_1678|        Herkku Gifts|   +47 2267 3215|Drammen 121, PR 7...|
|       Bergen|        |    N 5804|   Norway|     EMEA|
Oeztan|          Veysel|  Medium|
|      10201|           22.0|    98.57|              2|2168.54|
12/1/2003 0:00|Shipped|     4|      12|   2003|Motorcycles|95.0|
S10_1678|     Mini Wheels Co.|      6505555787|5557 North Pendal...|
|San Francisco|      CA|        |      USA|       NA|
Murphy|           Julie|   Small|
|      10211|           41.0|    100.0|             14|4708.44|
1/15/2004 0:00|Shipped|     1|       1|   2004|Motorcycles|95.0|
S10_1678|    Auto Canal Petit|  (1) 47.55.6555|   25, rue Lauriston|
|        Paris|        |     75016|   France|     EMEA|
```

```
Perrier|       Dominique|  Medium|
|      10223|           37.0|    100.0|              1|3965.66|
2/20/2004 0:00|Shipped|     1|       2|   2004|Motorcycles|95.0|
S10_1678|Australian Collec...|    03 9520 4555|   636 St Kilda Road|
Level 3|    Melbourne|Victoria|      3004|Australia|     APAC|
Ferguson|           Peter|  Medium|
|      10237|           23.0|    100.0|              7|2333.12|
4/5/2004 0:00|Shipped|     2|       4|   2004|Motorcycles|95.0|
S10_1678|     Vitachrome Inc.|      2125551500|   2678 Kingston Rd.|
Suite 101|          NYC|      NY|     10022|      USA|       NA|
Frick|         Michael|   Small|
|      10251|           28.0|    100.0|              2|3188.64|
5/18/2004 0:00|Shipped|     2|       5|   2004|Motorcycles|95.0|
S10_1678|Tekni Collectable...|      2015559350|       7476 Moss Rd.|
|       Newark|      NJ|     94019|      USA|       NA|
Brown|         William|  Medium|
|      10263|           34.0|    100.0|              2|3676.76|
6/28/2004 0:00|Shipped|     2|       6|   2004|Motorcycles|95.0|
S10_1678|     Gift Depot Inc.|      2035552570| 25593 South Bay Ln.|
|  Bridgewater|      CT|     97562|      USA|       NA|
King|           Julie|  Medium|
|      10275|           45.0|    92.83|              1|4177.35|
7/23/2004 0:00|Shipped|     3|       7|   2004|Motorcycles|95.0|
S10_1678|   La Rochelle Gifts|      40.67.8555|67, rue des Cinqu...|
|       Nantes|        |     44000|   France|     EMEA|
Labrune|          Janine|  Medium|
|      10285|           36.0|    100.0|              6|4099.68|
8/27/2004 0:00|Shipped|     3|       8|   2004|Motorcycles|95.0|
S10_1678|Marta's Replicas Co.|      6175558555| 39323 Spinnaker Dr.|
|    Cambridge|      MA|     51247|      USA|       NA|
Hernandez|           Marta|  Medium|
|      10299|           23.0|    100.0|              9|2597.39|
9/30/2004 0:00|Shipped|     3|       9|   2004|Motorcycles|95.0|
S10_1678|Toys of Finland, Co.|     90-224 8555|       Keskuskatu 45|
|     Helsinki|        |     21240|  Finland|     EMEA|
Karttunen|           Matti|   Small|
|      10309|           41.0|    100.0|              5|4394.38|
10/15/2004 0:00|Shipped|     4|      10|   2004|Motorcycles|95.0|
S10_1678|  Baane Mini Imports|      07-98 9555|Erling Skakkes ga...|
|      Stavern|        |      4110|   Norway|     EMEA|
Bergulfsen|           Jonas|  Medium|
|      10318|           46.0|    94.74|              1|4358.04|
11/2/2004 0:00|Shipped|     4|      11|   2004|Motorcycles|95.0|
S10_1678|Diecast Classics ...|      2155551555|    7586 Pompton St.|
|    Allentown|      PA|     70267|      USA|       NA|
Yu|           Kyung|  Medium|
|      10329|           42.0|    100.0|              1|4396.14|
11/15/2004 0:00|Shipped|     4|      11|   2004|Motorcycles|95.0|
S10_1678|   Land of Toys Inc.|      2125557818|897 Long Airport ...|
```

```
|          NYC|      NY|     10022|      USA|       NA|
Yu|           Kwai|  Medium|
+-----------+--------------+---------+--------------+-------
+---------------+-------+------+--------+-------+-----------+----
+-----------+--------------------+---------------
+--------------------+------------+-------------+--------+----------
+---------+---------+---------------+----------------+--------+
only showing top 20 rows
```

Insight: The outliers were in SALES, QUANTITYORDERED and MSRP column and have been removed

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?

```
top_10_countries_df =
cleaned_df.select('COUNTRY','SALES').groupBy("COUNTRY") \
    .agg(sum("SALES").alias("TOTAL_SALES")) \
    .orderBy('TOTAL_SALES',ascending=False) \
    .limit(10) \

top_10_countries_df.cache()
top_10_countries_df.persist(StorageLevel.MEMORY_ONLY)
top_10_countries_df.show()
# Choose MEMORY_AND_DISK for larger datasets to avoid data loss if
memory is insufficient
```

```
+---------+------------------+
|  COUNTRY|       TOTAL_SALES|
+---------+------------------+
|      USA|        3627982.83|
|    Spain|1215686.9200000009|
|   France|1110916.5199999993|
|Australia| 630623.1000000001|
|       UK| 478880.4600000001|
|    Italy|374674.30999999976|
|  Finland| 329581.9100000001|
|   Norway| 307463.7000000001|
|Singapore|288488.41000000003|
|  Denmark|         245637.15|
+---------+------------------+
```

Insight: We choose MEMORY_ONLY for caching in this case because our dataset is relatively small,and it can completely fit into memory without any risk of exceeding available resources. This allows for the fastest access to the data in subsequent transformations, avoiding any disk I/O overhead that would come with other persistence levels like MEMORY_AND_DISK.

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?

```
pivot_df = cleaned_df.groupBy("ORDERDATE") \
    .pivot("PRODUCTLINE") \
    .agg(sum("SALES")).alias('TOTAL_SALES')
pivot_df.show()
```

```
+--------------+-----------------+-----------------+-------
+-----------------+-------+-----------------+-----------------+
|     ORDERDATE|      Classic Cars|       Motorcycles| Planes|
Ships| Trains|  Trucks and Buses|      Vintage Cars|
+--------------+-----------------+-----------------+-------
+-----------------+-------+-----------------+-----------------+
|3/29/2004 0:00|              NULL|              NULL|   NULL|
4933.719999999999|   NULL|              NULL|           3788.4|
|5/30/2005 0:00|              NULL|              NULL|   NULL|
NULL|   NULL|             NULL|          14578.75|
|3/19/2004 0:00|           15330.7|              NULL|   NULL|
NULL|   NULL|             NULL|              NULL|
| 9/7/2004 0:00|              NULL|              NULL|   NULL|
NULL|   NULL|             NULL| 7673.379999999999|
| 5/4/2004 0:00|          15340.86|              NULL|   NULL|
NULL|   NULL|          18938.3|           2694.15|
|11/9/2004 0:00|              NULL|              NULL|   NULL|
6673.29|3807.68|              NULL|           9665.35|
|11/4/2003 0:00|          37852.99|          18877.11|   NULL|
NULL|   NULL|             NULL|              NULL|
| 7/1/2003 0:00|              NULL|25624.880000000005|   NULL|
NULL|   NULL|             NULL|              NULL|
|12/1/2003 0:00|              NULL|25431.879999999997|7120.96|
NULL|   NULL|             NULL|            1113.6|
| 7/2/2004 0:00|          12334.82|              NULL|   NULL|
NULL|   NULL|             NULL|              NULL|
|1/29/2003 0:00|           15263.7|              NULL|   NULL|
NULL|   NULL|          23041.1|16397.199999999997|
| 9/3/2003 0:00|           4444.54|           3155.58|   NULL|
NULL|   NULL|             NULL|              NULL|
|3/23/2005 0:00|           6109.29|              NULL|   NULL|
8320.880000000001| 2154.0|             NULL|15610.619999999999|
|10/4/2003 0:00|          27257.79|              NULL|   NULL|
NULL|   NULL|             NULL|              NULL|
|7/24/2003 0:00|          14677.02|              NULL|   NULL|
NULL|   NULL|26099.979999999996|           1254.83|
| 3/3/2003 0:00|          42605.87|          12639.15|   NULL|
NULL|   NULL|             NULL|              NULL|
|1/23/2005 0:00|19850.739999999998|              NULL|   NULL|
NULL| 2986.5|13287.029999999999|              NULL|
|4/26/2004 0:00|              NULL|              NULL|   NULL|
NULL|   NULL|             NULL|            7129.0|
```

```
|2/16/2005 0:00|18356.399999999998|              NULL|   NULL|
NULL|   NULL|              NULL|           20004.0|
|4/21/2003 0:00|              NULL|              NULL|   NULL|
4219.2|   NULL|              NULL|              NULL|
+--------------+------------------+------------------+-------
+-----------------+-------+------------------+------------------+
only showing top 20 rows
```

Insight: Pivoting can expnd dataframes, increasing memory usage. In this case, many positions are null due to the absence of values, yet they still occupy memory space. This leads to additional memory overhead, even though these positions don't contribute meaningful data.

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

```
df = cleaned_df.withColumn("ORDERDATE_TS",
to_timestamp(col("ORDERDATE"), 'M/d/yyyy H:mm')) \
               .withColumn("MONTH", month(col("ORDERDATE_TS"))) \
               .withColumn("YEAR", year(col("ORDERDATE_TS")))

monthly_sales = df.groupBy("YEAR", "MONTH", "PRODUCTLINE") \
                  .agg(sum("SALES").alias("TOTAL_SALES"))


windowSpec = Window.partitionBy("PRODUCTLINE").orderBy("YEAR",
"MONTH")

monthly_growth = monthly_sales.withColumn("PREV_MONTH_TOTAL_SALES",
lag("TOTAL_SALES").over(windowSpec)) \
                              .withColumn("PERCENTAGE_GROWTH",
                                          (col("TOTAL_SALES") -
col("PREV_MONTH_TOTAL_SALES")) / col("PREV_MONTH_TOTAL_SALES") * 100)

monthly_growth.show(50)

+----+-----+-----------+------------------+---------------------
+------------------+
|YEAR|MONTH| PRODUCTLINE|       TOTAL_SALES|PREV_MONTH_TOTAL_SALES|
PERCENTAGE_GROWTH|
+----+-----+-----------+------------------+---------------------
+------------------+
|2003|    2| Motorcycles|25783.760000000002|                  null|
null|
|2003|    3| Motorcycles|          12639.15|    25783.760000000002| -
50.98019063162239|
|2003|    4| Motorcycles|23475.590000000004|               12639.15|
85.7370946622202|
|2003|    5| Motorcycles|          22097.32|    23475.590000000004|-
```

```
5.8710771486467594|
|2003|    6| Motorcycles|           2642.01|              22097.32| -
88.04375372217082|
|2003|    7| Motorcycles| 37924.23000000001|               2642.01|
1335.430978686682|
|2003|    8| Motorcycles|44164.909999999996|     37924.23000000001|
16.455653812878953|
|2003|    9| Motorcycles|           3155.58|    44164.909999999996| -
92.85500638402749|
|2003|   10| Motorcycles| 64235.65000000001|               3155.58|
1935.6210268793695|
|2003|   11| Motorcycles|          109345.5|     64235.65000000001|
70.22556788948191|
|2003|   12| Motorcycles|25431.879999999997|              109345.5| -
76.74172233882508|
|2004|    1| Motorcycles|          41200.52|    25431.879999999997|
62.00343820433252|
|2004|    2| Motorcycles|           49066.5|              41200.52|
19.091943499742246|
|2004|    4| Motorcycles| 36269.07000000001|               49066.5| -
26.08180734309558|
|2004|    5| Motorcycles|46848.950000000004|     36269.07000000001|
29.17053015144859|
|2004|    6| Motorcycles|          47237.41|    46848.950000000004|
0.8291754671129217|
|2004|    7| Motorcycles|           22774.0|              47237.41|-
51.788211927791984|
|2004|    8| Motorcycles|          62704.93|               22774.0|
175.3356020022833|
|2004|    9| Motorcycles| 42471.04999999999|              62704.93| -
32.26840377622623|
|2004|   10| Motorcycles|          39413.96|     42471.04999999999|-
7.1980560876173065|
|2004|   11| Motorcycles|151711.85999999996|              39413.96|
284.9190997301463|
|2004|   12| Motorcycles|          20846.98|    151711.85999999996| -
86.25883302729265|
|2005|    1| Motorcycles| 39913.35999999999|              20846.98|
91.45871488340275|
|2005|    2| Motorcycles|          47951.42|     39913.35999999999|
20.138770577069952|
|2005|    3| Motorcycles|47830.829999999994|              47951.42|-
0.2514836891170351|
|2005|    4| Motorcycles|          59862.22|    47830.829999999994|
25.15404813171757|
|2005|    5| Motorcycles|           39389.7|              59862.22|-
34.199399888610884|
|2003|    1|Vintage Cars|          46826.84|                  null|
null|
```

```
|2003|    2|Vintage Cars|          24002.23|              46826.84|-
48.742580110039455|
|2003|    3|Vintage Cars|46931.009999999995|              24002.23|
95.52770721720438|
|2003|    4|Vintage Cars|          20750.34|    46931.009999999995|-
55.785439094534716|
|2003|    5|Vintage Cars| 47033.58000000001|              20750.34|
126.66414140683963|
|2003|    6|Vintage Cars|26582.510000000006|     47033.58000000001| -
43.48184850058192|
|2003|    7|Vintage Cars|          29652.09|    26582.510000000006|
11.547367046979362|
|2003|    8|Vintage Cars|          23285.46|              29652.09| -
21.47110035076786|
|2003|    9|Vintage Cars| 53045.90000000001|              23285.46|
127.80696623558225|
|2003|   10|Vintage Cars|         100603.25|     53045.90000000001|
89.65320599707043|
|2003|   11|Vintage Cars|184673.39999999997|             100603.25|
83.56603787650992|
|2003|   12|Vintage Cars|          47601.15|    184673.39999999997| -
74.22414381280682|
|2004|    1|Vintage Cars|           74917.4|              47601.15|
57.38569341286921|
|2004|    2|Vintage Cars|          25041.84|               74917.4| -
66.57406690568546|
|2004|    3|Vintage Cars|55981.220000000016|              25041.84|
123.5507454723775|
|2004|    4|Vintage Cars|          55030.32|    55981.220000000016|-
1.6986053537240091|
|2004|    5|Vintage Cars|          26749.18|              55030.32| -
51.39192357958303|
|2004|    6|Vintage Cars|47985.840000000004|              26749.18|
79.39181687064801|
|2004|    7|Vintage Cars|40712.490000000005|    47985.840000000004|-
15.157283898750126|
|2004|    8|Vintage Cars|          78139.05|    40712.490000000005|
91.92893876056216|
|2004|    9|Vintage Cars| 73100.31999999999|              78139.05|-
6.4484147170972905|
|2004|   10|Vintage Cars|116160.29000000002|     73100.31999999999|
58.9053098536368|
|2004|   11|Vintage Cars|233990.33999999997|    116160.29000000002|
101.43746197603323|
+----+-----+------------+------------------+----------------------
+-------------------+
only showing top 50 rows
```

Insight: Highest growth for Classic Cars was in March 2003 etc

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

```
repartitionDF = cleaned_df.repartition("COUNTRY")
print("Number of partitions after repartitioning:",
repartitionDF.rdd.getNumPartitions())
```

```
Number of partitions after repartitioning: 200
```

Insight :By repartitioning the data, we ensure that it is evenly distributed across partitions, which prevents situations where some partitions might have less data than others.

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?

```
from pyspark.sql.functions import broadcast

customer_details = spark.createDataFrame([
    (1, "Land of Toys Inc.", "john.doe@example.com"),
    (2, "Reims Collectables", "jane.smith@example.com"),
    (3, "Lyon Souveniers", "alice.johnson@example.com")
], ["customer_id", "customer_name", "email"])

joined_df = cleaned_df.join(
    broadcast(customer_details),
    cleaned_df["CUSTOMERNAME"] == customer_details["customer_name"],

)

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|customer_id|     customer_name|
email|
+-----------+---------------+---------+---------------+-------
+---------------+---------+------+--------+-------+------------+----
+-----------+------------------+----------------+--------------------
+------------+-----+-----+----------+-------+---------+---------------
+----------------+--------+-----------+------------------
+--------------------+
```

```
|      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|          1| Land of Toys Inc.|john.doe@example.com|
|      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|          2|Reims Collectables|jane.smith@exampl...|
|      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|          3|   Lyon Souveniers|alice.johnson@exa...|
|      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|          1| Land of Toys Inc.|john.doe@example.com|
|      10107|             39|    99.91|              5|3896.49|
2/24/2003 0:00|  Shipped|     1|       2|   2003| Motorcycles| 118|
S10_2016| Land of Toys Inc.|      2125557818|897 Long Airport ...|
|  NYC|   NY|     10022|    USA|       NA|             Yu|
Kwai|  Medium|          1| Land of Toys Inc.|john.doe@example.com|
|      10134|             27|    100.0|              5|3307.77|
7/1/2003 0:00|  Shipped|     3|       7|   2003| Motorcycles| 118|
S10_2016|   Lyon Souveniers|+33 1 46 62 7555|27 rue du Colonel...|
|Paris|     |     75508| France|     EMEA|       Da Cunha|
Daniel|  Medium|          3|   Lyon Souveniers|alice.johnson@exa...|
|      10329|             20|    100.0|              2| 3176.0|
11/15/2004 0:00|  Shipped|     4|      11|   2004| Motorcycles| 118|
S10_2016| Land of Toys Inc.|      2125557818|897 Long Airport ...|
|  NYC|   NY|     10022|    USA|       NA|             Yu|
Kwai|  Medium|          1| Land of Toys Inc.|john.doe@example.com|
|      10107|             27|    100.0|              4|6065.55|
2/24/2003 0:00|  Shipped|     1|       2|   2003| Motorcycles| 193|
S10_4698| Land of Toys Inc.|      2125557818|897 Long Airport ...|
|  NYC|   NY|     10022|    USA|       NA|             Yu|
Kwai|  Medium|          1| Land of Toys Inc.|john.doe@example.com|
|      10134|             31|    100.0|              4|7023.98|
7/1/2003 0:00|  Shipped|     3|       7|   2003| Motorcycles| 193|
S10_4698|   Lyon Souveniers|+33 1 46 62 7555|27 rue du Colonel...|
|Paris|     |     75508| France|     EMEA|       Da Cunha|
Daniel|   Large|          3|   Lyon Souveniers|alice.johnson@exa...|
|      10329|             26|    100.0|              3| 5868.2|
11/15/2004 0:00|  Shipped|     4|      11|   2004| Motorcycles| 193|
S10_4698| Land of Toys Inc.|      2125557818|897 Long Airport ...|
|  NYC|   NY|     10022|    USA|       NA|             Yu|
```

```
Kwai|  Medium|          1| Land of Toys Inc.|john.doe@example.com|
|      10248|            20|    100.0|             3| 2910.4|
5/7/2004 0:00|Cancelled|     2|       5|   2004|Classic Cars| 136|
S10_4757| Land of Toys Inc.|      2125557818|897 Long Airport ...|
|  NYC|   NY|     10022|    USA|       NA|            Yu|
Kwai|   Small|          1| Land of Toys Inc.|john.doe@example.com|
|      10359|            48|    54.68|             6|2624.64|
12/15/2004 0:00|  Shipped|     4|      12|   2004|Classic Cars| 136|
S10_4757|Reims Collectables|      26.47.1555|  59 rue de l'Abbaye|
|Reims|     |     51100| France|     EMEA|        Henriot|
Paul|   Small|          2|Reims Collectables|jane.smith@exampl...|
|      10395|            32|    100.0|             2|3370.56|
3/17/2005 0:00|  Shipped|     1|       3|   2005|Classic Cars| 136|
S10_4757|   Lyon Souveniers|+33 1 46 62 7555|27 rue du Colonel...|
|Paris|     |     75508| France|     EMEA|       Da Cunha|
Daniel|  Medium|          3|   Lyon Souveniers|alice.johnson@exa...|
|      10329|            41|    71.47|             5|2930.27|
11/15/2004 0:00|  Shipped|     4|      11|   2004|Classic Cars| 194|
S12_1099| Land of Toys Inc.|      2125557818|897 Long Airport ...|
|  NYC|   NY|     10022|    USA|       NA|            Yu|
Kwai|   Small|          1| Land of Toys Inc.|john.doe@example.com|
|      10359|            42|    100.0|             8|4764.48|
12/15/2004 0:00|  Shipped|     4|      12|   2004|Classic Cars| 207|
S12_1108|Reims Collectables|      26.47.1555|  59 rue de l'Abbaye|
|Reims|     |     51100| France|     EMEA|        Henriot|
Paul|  Medium|          2|Reims Collectables|jane.smith@exampl...|
|      10395|            33|    69.12|             1|2280.96|
3/17/2005 0:00|  Shipped|     1|       3|   2005|Classic Cars| 207|
S12_1108|   Lyon Souveniers|+33 1 46 62 7555|27 rue du Colonel...|
|Paris|     |     75508| France|     EMEA|       Da Cunha|
Daniel|   Small|          3|   Lyon Souveniers|alice.johnson@exa...|
|      10107|            21|    100.0|             1| 3036.6|
2/24/2003 0:00|  Shipped|     1|       2|   2003| Motorcycles| 150|
S12_2823| Land of Toys Inc.|      2125557818|897 Long Airport ...|
|  NYC|   NY|     10022|    USA|       NA|            Yu|
Kwai|  Medium|          1| Land of Toys Inc.|john.doe@example.com|
|      10121|            50|    100.0|             4| 8284.0|
5/7/2003 0:00|  Shipped|     2|       5|   2003| Motorcycles| 150|
S12_2823|Reims Collectables|      26.47.1555|  59 rue de l'Abbaye|
|Reims|     |     51100| France|     EMEA|        Henriot|
Paul|   Large|          2|Reims Collectables|jane.smith@exampl...|
|      10134|            20|    100.0|             1| 2711.2|
7/1/2003 0:00|  Shipped|     3|       7|   2003| Motorcycles| 150|
S12_2823|   Lyon Souveniers|+33 1 46 62 7555|27 rue du Colonel...|
|Paris|     |     75508| France|     EMEA|       Da Cunha|
Daniel|   Small|          3|   Lyon Souveniers|alice.johnson@exa...|
|      10329|            24|    100.0|             6|3542.64|
11/15/2004 0:00|  Shipped|     4|      11|   2004| Motorcycles| 150|
S12_2823| Land of Toys Inc.|      2125557818|897 Long Airport ...|
```

```
|  NYC|   NY|     10022|    USA|       NA|             Yu|
Kwai|  Medium|          1| Land of Toys Inc.|john.doe@example.com|
+----------+--------------+---------+--------------+-------
+--------------+---------+------+--------+-------+-----------+----
+----------+-----------------+---------------+-------------------
+-----------+-----+-----+----------+-------+---------+--------------
+----------------+--------+-----------+------------------
+--------------------+
only showing top 20 rows
```

Insight: Size of DataFrame:Ensure the DataFrame being broadcasted is small to fit into memory on all nodes. Resource Overheads:Broadcasting large DataFrames increases network traffic and memory usage. Data Skew:Check for even data distribution to avoid inefficiencies with large, skewed DataFrames.

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.

```
from pyspark.sql.functions import udf, col, when
from pyspark.sql.types import StringType

def categorize_sales(sales_amount):
    if sales_amount > percentile_67:
        return "High"
    elif sales_amount > percentile_33:
        return "Medium"
    else:
        return "Low"

categorize_sales_udf = udf(categorize_sales, StringType())


cleaned_df = cleaned_df.withColumn("SALES",
col("SALES").cast("float"))



df = cleaned_df.withColumn("SALES_CATEGORY", \
                           categorize_sales_udf(col("SALES")))

df.groupBy("COUNTRY", "SALES_CATEGORY").count().show()

+-----------+-------------+-----+
|    COUNTRY|SALES_CATEGORY|count|
+-----------+-------------+-----+
|Philippines|          Low|    7|
|     Norway|       Medium|   21|
|        USA|          Low|  312|
```

```
|    Austria|           Low|   15|
|     Canada|          High|   18|
|    Denmark|           Low|   18|
|     Canada|        Medium|   24|
|     Canada|           Low|   28|
|      Italy|          High|   32|
|Switzerland|        Medium|   13|
|    Ireland|          High|    8|
|Philippines|          High|    9|
|    Ireland|           Low|    7|
|  Australia|          High|   63|
|    Finland|        Medium|   36|
|  Singapore|           Low|   28|
|     Norway|          High|   34|
|        USA|        Medium|  332|
|      Italy|           Low|   38|
|        USA|          High|  360|
+-----------+--------------+-----+
only showing top 20 rows
```

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

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import udf, col
from pyspark.sql.types import FloatType


def apply_discount(productline, sales):
    if productline == "Classic Cars":
        discount = 0.10
    elif productline == "Motorcycles":
        discount = 0.05
    else:
        discount = 0.0

    discounted_sales = sales * (1 - discount)
    return discounted_sales


apply_discount_udf = udf(apply_discount, FloatType())



discounted_df = cleaned_df.withColumn(
    "DISCOUNTED_SALES",
    apply_discount_udf(col("PRODUCTLINE"), col("SALES"))
)
```

```
discounted_df.select("PRODUCTLINE","SALES","DISCOUNTED_SALES").show()
```

```
+-----------+-------+----------------+
|PRODUCTLINE|  SALES|DISCOUNTED_SALES|
+-----------+-------+----------------+
|Motorcycles| 2871.0|          2727.45|
|Motorcycles| 2765.9|         2627.605|
|Motorcycles|3884.34|         3690.123|
|Motorcycles| 3746.7|         3559.365|
|Motorcycles|5205.27|        4945.0063|
|Motorcycles|3479.76|         3305.772|
|Motorcycles|2497.77|        2372.8816|
|Motorcycles|5512.32|        5236.7036|
|Motorcycles|2168.54|         2060.113|
|Motorcycles|4708.44|         4473.018|
|Motorcycles|3965.66|         3767.377|
|Motorcycles|2333.12|         2216.464|
|Motorcycles|3188.64|         3029.208|
|Motorcycles|3676.76|         3492.922|
|Motorcycles|4177.35|        3968.4827|
|Motorcycles|4099.68|        3894.6963|
|Motorcycles|2597.39|        2467.5205|
|Motorcycles|4394.38|         4174.661|
|Motorcycles|4358.04|         4140.138|
|Motorcycles|4396.14|         4176.333|
+-----------+-------+----------------+
only showing top 20 rows
```

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?

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

window = Window.partitionBy("CUSTOMERNAME").orderBy("SALES")
df_cdf=cleaned_df.withColumn("CDF",
cume_dist().over(window)).select("CUSTOMERNAME","SALES","CDF")
df_cdf.show(100)
```

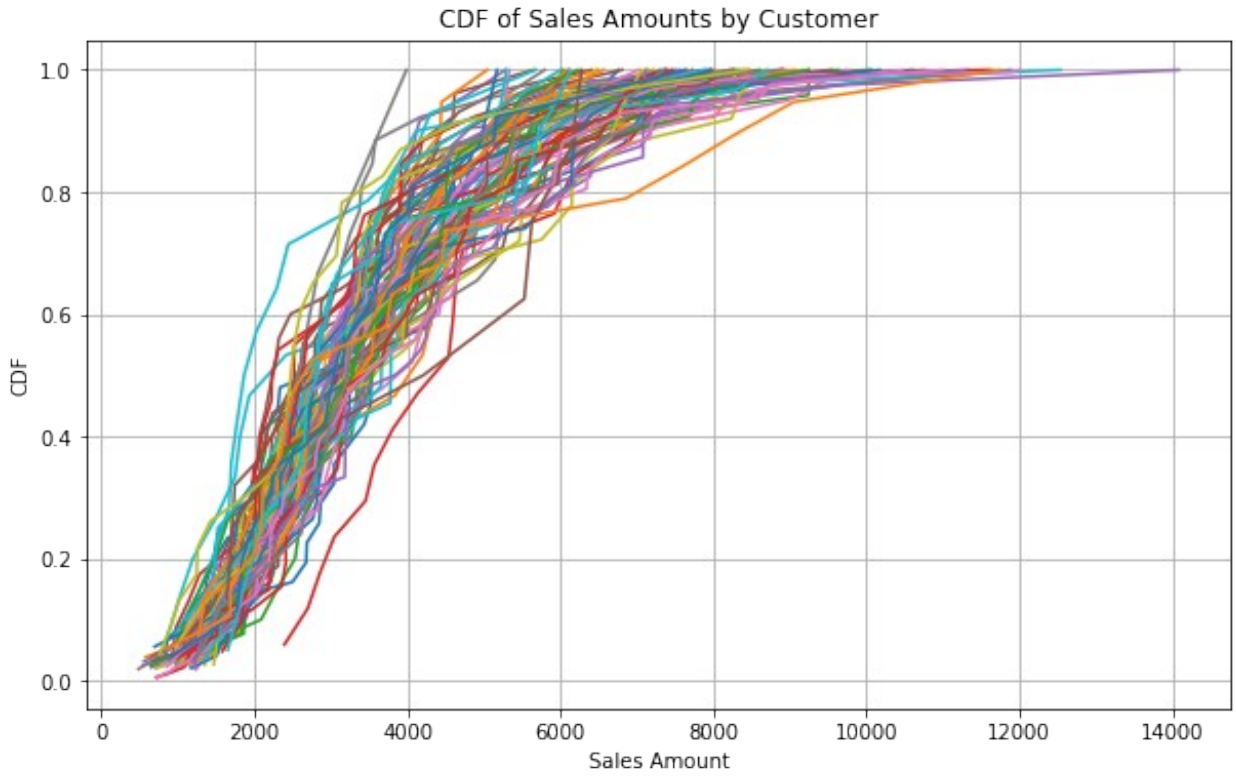
```
+--------------------+-------+--------------------+
|        CUSTOMERNAME|  SALES|                 CDF|
+--------------------+-------+--------------------+
| Suominen Souveniers| 891.03| 0.03333333333333333|
```

Suominen Souveniers	1086.6	0.06666666666666667
Suominen Souveniers	1103.76	0.1
Suominen Souveniers	1629.04	0.13333333333333333
Suominen Souveniers	1988.4	0.16666666666666666
Suominen Souveniers	2140.11	0.2
Suominen Souveniers	2447.76	0.23333333333333334
Suominen Souveniers	2632.89	0.26666666666666666
Suominen Souveniers	2773.8	0.3
Suominen Souveniers	2775.08	0.3333333333333333
Suominen Souveniers	2817.87	0.36666666666666664
Suominen Souveniers	2851.84	0.4
Suominen Souveniers	2931.98	0.43333333333333335
Suominen Souveniers	3128.65	0.4666666666666667
Suominen Souveniers	3288.82	0.5
Suominen Souveniers	3595.62	0.5333333333333333
Suominen Souveniers	3686.54	0.5666666666666667
Suominen Souveniers	3784.8	0.6
Suominen Souveniers	4068.7	0.6333333333333333
Suominen Souveniers	4142.64	0.6666666666666666
Suominen Souveniers	4157.73	0.7
Suominen Souveniers	4381.25	0.7333333333333333
Suominen Souveniers	4836.5	0.7666666666666667
Suominen Souveniers	5154.41	0.8
Suominen Souveniers	5500.44	0.8333333333333334
Suominen Souveniers	5938.53	0.8666666666666667
Suominen Souveniers	6287.66	0.9
Suominen Souveniers	6576.5	0.9333333333333333
Suominen Souveniers	6756.0	0.9666666666666667
Suominen Souveniers	10606.2	1.0
Amica Models & Co.	577.6	0.038461538461538464
Amica Models & Co.	1381.05	0.07692307692307693
Amica Models & Co.	1557.36	0.11538461538461539
Amica Models & Co.	1574.0	0.15384615384615385
Amica Models & Co.	1656.69	0.19230769230769232
Amica Models & Co.	1921.92	0.23076923076923078
Amica Models & Co.	2084.81	0.2692307692307692
Amica Models & Co.	2137.05	0.3076923076923077
Amica Models & Co.	2418.24	0.34615384615384615
Amica Models & Co.	2800.08	0.38461538461538464
Amica Models & Co.	2819.28	0.4230769230769231
Amica Models & Co.	2941.89	0.46153846153846156
Amica Models & Co.	2954.53	0.5
Amica Models & Co.	3006.43	0.5384615384615384
Amica Models & Co.	3474.46	0.5769230769230769
Amica Models & Co.	3668.6	0.6153846153846154
Amica Models & Co.	3704.05	0.6538461538461539
Amica Models & Co.	4242.24	0.6923076923076923
Amica Models & Co.	4455.0	0.7307692307692307
Amica Models & Co.	4750.8	0.7692307692307693

```
|  Amica Models & Co.|4946.06|  0.8076923076923077|
|  Amica Models & Co.|5126.24|  0.8461538461538461|
|  Amica Models & Co.| 5239.5|  0.8846153846153846|
|  Amica Models & Co.|8014.82|  0.9230769230769231|
|  Amica Models & Co.| 8253.0|  0.9615384615384616|
|  Amica Models & Co.|8411.56|                 1.0|
|Collectables For ...|1066.75|0.041666666666666664|
|Collectables For ...|1237.88| 0.08333333333333333|
|Collectables For ...|1340.64|               0.125|
|Collectables For ...|1735.92| 0.16666666666666666|
|Collectables For ...| 1875.2| 0.20833333333333334|
|Collectables For ...|1978.62|                0.25|
|Collectables For ...|2004.77|  0.2916666666666667|
|Collectables For ...| 2057.4|  0.3333333333333333|
|Collectables For ...|2142.14|               0.375|
|Collectables For ...|2172.48|  0.4166666666666667|
|Collectables For ...| 2603.2|  0.4583333333333333|
|Collectables For ...|2906.97|                 0.5|
|Collectables For ...|3097.44|  0.5416666666666666|
|Collectables For ...|3156.16|  0.5833333333333334|
|Collectables For ...|3230.37|               0.625|
|Collectables For ...| 3288.6|  0.6666666666666666|
|Collectables For ...| 3476.8|  0.7083333333333334|
|Collectables For ...|3494.94|                0.75|
|Collectables For ...|4405.22|  0.7916666666666666|
|Collectables For ...|4514.92|  0.8333333333333334|
|Collectables For ...|4581.36|               0.875|
|Collectables For ...| 6724.0|  0.9166666666666666|
|Collectables For ...|9240.44|  0.9583333333333334|
|Collectables For ...|9245.76|                 1.0|
|         CAF Imports| 1824.0| 0.07692307692307693|
|         CAF Imports| 2009.2| 0.15384615384615385|
|         CAF Imports|2451.84| 0.23076923076923078|
|         CAF Imports|2526.48|  0.3076923076923077|
|         CAF Imports| 2611.0| 0.38461538461538464|
|         CAF Imports|2738.54| 0.46153846153846156|
|         CAF Imports|3070.52|  0.5384615384615384|
|         CAF Imports| 3266.1|  0.6153846153846154|
|         CAF Imports|3675.63|  0.6923076923076923|
|         CAF Imports|4055.04|  0.7692307692307693|
|         CAF Imports| 6083.0|  0.8461538461538461|
|         CAF Imports|6952.12|  0.9230769230769231|
|         CAF Imports|8378.58|                 1.0|
|       Rovelli Gifts| 977.43|0.020833333333333332|
|       Rovelli Gifts|1112.94|0.041666666666666664|
|       Rovelli Gifts|1163.05|              0.0625|
|       Rovelli Gifts| 1459.6| 0.08333333333333333|
|       Rovelli Gifts|1495.26| 0.10416666666666667|
|       Rovelli Gifts|1504.16|               0.125|
```

```
|       Rovelli Gifts|1565.85| 0.14583333333333334|
+--------------------+-------+--------------------+
only showing top 100 rows
```

```
plt.figure(figsize=(10, 6))

for customer in df_pandas['CUSTOMERNAME'].unique():
    customer_data = df_pandas[df_pandas['CUSTOMERNAME'] == customer]
    plt.plot(customer_data['SALES'], customer_data['CDF'],
label=customer)


plt.xlabel('Sales Amount')
plt.ylabel('CDF')
plt.title('CDF of Sales Amounts by Customer')


plt.grid(True)


plt.show()
```



Insight: # 50% of sales for majority of customers are below 2500