In [1]:
```
from pyspark.sql import SparkSession
import pyspark.sql.functions as F
from pyspark.sql.types import *

spark = SparkSession\
    .builder\
    .appName("chapter-03-tour")\
    .getOrCreate()

import os
SPARK_BOOK_DATA_PATH = os.environ['SPARK_BOOK_DATA_PATH']
```

Spark SQL

In [2]:
```
file_path = SPARK_BOOK_DATA_PATH + "/data/retail-data/by-day/*.csv"
retail_df = spark.read.format("csv")\
  .option("header", "true")\
  .option("inferSchema", "true")\
  .load(file_path)
```

In [3]:
```
retail_df.count()
```

Out[3]:
```
541909
```

In [4]:
```
retail_df.createOrReplaceTempView("retail_table")
```

In [5]:
```
staticSchema = retail_df.schema
```

In [6]:
```
print(staticSchema)
```

```
StructType(List(StructField(InvoiceNo,StringType,true),StructField(Sto
ckCode,StringType,true),StructField(Description,StringType,true),Struc
tField(Quantity,IntegerType,true),StructField(InvoiceDate,TimestampTyp
e,true),StructField(UnitPrice,DoubleType,true),StructField(CustomerID,
DoubleType,true),StructField(Country,StringType,true)))
```

In [7]:
```
retail_df.printSchema()
```

```
root
 |-- InvoiceNo: string (nullable = true)
 |-- StockCode: string (nullable = true)
 |-- Description: string (nullable = true)
 |-- Quantity: integer (nullable = true)
 |-- InvoiceDate: timestamp (nullable = true)
 |-- UnitPrice: double (nullable = true)
 |-- CustomerID: double (nullable = true)
 |-- Country: string (nullable = true)
```

In [8]:
```
retail_df.describe().show()
```

```
+-------+-----------------+-----------------+--------------------+--
---------------+-----------------+------------------+-----------+
|summary|         InvoiceNo|        StockCode|         Description|
Quantity|        UnitPrice|        CustomerID|    Country|
+-------+-----------------+-----------------+--------------------+--
---------------+-----------------+------------------+-----------+
|  count|           541909|           541909|              540455|
541909|           541909|            406829|     541909|
|   mean|  559965.752026781|27623.240210938104|             20713.0|
9.55224954743324|4.611113626089641|15287.690570239585|       null|
| stddev|13428.417280796697|16799.737628427683|                NaN|21
8.0811578502335|96.75985306117963| 1713.600303321597|       null|
|    min|           536365|            10002| 4 PURPLE FLOCK D...|
-80995|        -11062.06|           12346.0|  Australia|
|    max|          C581569|                m|   wrongly sold sets|
80995|          38970.0|           18287.0|Unspecified|
+-------+-----------------+-----------------+--------------------+--
---------------+-----------------+------------------+-----------+
```

In [9]:
```
df = spark.sql("select * from retail_table limit 5")
```

In [10]:
```
df.show()
```

```
+---------+---------+--------------------+--------+------------------
+---------+----------+--------------+
|InvoiceNo|StockCode|         Description|Quantity|        InvoiceDate
|UnitPrice|CustomerID|       Country|
+---------+---------+--------------------+--------+------------------
+---------+----------+--------------+
|   580538|    23084|  RABBIT NIGHT LIGHT|      48|2011-12-05 08:38:00
|     1.79|   14075.0|United Kingdom|
|   580538|    23077| DOUGHNUT LIP GLOSS |      20|2011-12-05 08:38:00
|     1.25|   14075.0|United Kingdom|
|   580538|    22906|12 MESSAGE CARDS ...|      24|2011-12-05 08:38:00
|     1.65|   14075.0|United Kingdom|
|   580538|    21914|BLUE HARMONICA IN...|      24|2011-12-05 08:38:00
|     1.25|   14075.0|United Kingdom|
|   580538|    22467|   GUMBALL COAT RACK|       6|2011-12-05 08:38:00
|     2.55|   14075.0|United Kingdom|
+---------+---------+--------------------+--------+------------------
+---------+----------+--------------+
```

In [11]:
```
# COMMAND ----------

from pyspark.sql.functions import window, column, desc, col

retail_df.selectExpr(
    "CustomerId",
    "(UnitPrice * Quantity) as total_cost",
    "InvoiceDate")\
  .groupBy(col("CustomerId"), window(col("InvoiceDate"), "1 day"))\
  .sum("total_cost")\
  .sort(desc("sum(total_cost)"))\
  .show(5)
```

```
+----------+--------------------+------------------+
|CustomerId|              window|   sum(total_cost)|
+----------+--------------------+------------------+
|   17450.0|[2011-09-19 20:00...|          71601.44|
|      null|[2011-11-13 19:00...|          55316.08|
|      null|[2011-11-06 19:00...|          42939.17|
|      null|[2011-03-28 20:00...| 33521.39999999998|
|      null|[2011-12-07 19:00...|31975.590000000007|
+----------+--------------------+------------------+
only showing top 5 rows
```

Spark Streaming

In [12]:
```
# COMMAND ----------

streamingDataFrame = spark\
    .readStream\
    .format("csv")\
    .schema(staticSchema)\
    .option("maxFilesPerTrigger", 1)\
    .option("header", "true")\
    .load(SPARK_BOOK_DATA_PATH + "/data/retail-data/by-day/*.csv")
```

In [13]:
```
# COMMAND ----------

purchaseByCustomerPerHour = streamingDataFrame\
  .selectExpr(
    "CustomerId",
    "(UnitPrice * Quantity) as total_cost",
    "InvoiceDate")\
  .groupBy(col("CustomerId"), window(col("InvoiceDate"), "1 day"))\
  .sum("total_cost")
```

In [14]:
```
# COMMAND ----------

purchaseByCustomerPerHour\
    .writeStream\
    .format("memory")\
    .queryName("customer_purchases")\
    .outputMode("complete")\
    .start()
```

Out[14]:
```
<pyspark.sql.streaming.StreamingQuery at 0x7f7ff5f67630>
```

use `Ctrl-Enter` to execute below cell repeatly to see streaming result

In [33]:
```
# COMMAND ----------

spark.sql("""
  SELECT *
  FROM customer_purchases
  ORDER BY `sum(total_cost)` DESC
  """)\
  .show(5)
```

```
+----------+--------------------+------------------+
|CustomerId|              window|   sum(total_cost)|
+----------+--------------------+------------------+
|   17450.0|[2011-09-19 20:00...|          71601.44|
|      null|[2011-11-13 19:00...|          55316.08|
|      null|[2011-08-29 20:00...| 23032.59999999993|
|   12931.0|[2011-08-03 20:00...|19045.480000000003|
|      null|[2011-05-09 20:00...| 17949.28000000001|
+----------+--------------------+------------------+
only showing top 5 rows
```

In [54]:
```
# COMMAND ----------

spark.sql("""
  SELECT *
  FROM customer_purchases
  ORDER BY `sum(total_cost)` DESC
  """)\
  .show(5)
```

```
+----------+--------------------+------------------+
|CustomerId|              window|   sum(total_cost)|
+----------+--------------------+------------------+
|   17450.0|[2011-09-19 20:00...|          71601.44|
|      null|[2011-11-13 19:00...|          55316.08|
|      null|[2011-11-06 19:00...|          42939.17|
|      null|[2011-03-28 20:00...| 33521.39999999998|
|      null|[2011-12-07 19:00...|31975.590000000007|
+----------+--------------------+------------------+
only showing top 5 rows
```

Spark ML Pipeline

In [35]:
```
# COMMAND ----------

from pyspark.sql.functions import date_format, col

preppedDataFrame = retail_df\
  .na.fill(0)\
  .withColumn("day_of_week", date_format(col("InvoiceDate"), "EEEE"))\
  .coalesce(5)

preppedDataFrame.show(3)
```

```
+---------+---------+--------------------+--------+-------------------
+---------+----------+--------------+-----------+
|InvoiceNo|StockCode|         Description|Quantity|        InvoiceDate
|UnitPrice|CustomerID|       Country|day_of_week|
+---------+---------+--------------------+--------+-------------------
+---------+----------+--------------+-----------+
|   580538|    23084|  RABBIT NIGHT LIGHT|      48|2011-12-05 08:38:00
|     1.79|   14075.0|United Kingdom|     Monday|
|   580538|    23077| DOUGHNUT LIP GLOSS |      20|2011-12-05 08:38:00
|     1.25|   14075.0|United Kingdom|     Monday|
|   580538|    22906|12 MESSAGE CARDS ...|      24|2011-12-05 08:38:00
|     1.65|   14075.0|United Kingdom|     Monday|
+---------+---------+--------------------+--------+-------------------
+---------+----------+--------------+-----------+
only showing top 3 rows
```

In [36]:
```
# COMMAND ----------

trainDataFrame = preppedDataFrame\
  .where("InvoiceDate < '2011-07-01'")

testDataFrame = preppedDataFrame\
  .where("InvoiceDate >= '2011-07-01'")

trainDataFrame.show(3)
```

```
+---------+---------+--------------------+--------+--------------------
+---------+----------+--------------+-----------+
|InvoiceNo|StockCode|         Description|Quantity|        InvoiceDate
|UnitPrice|CustomerID|       Country|day_of_week|
+---------+---------+--------------------+--------+--------------------
+---------+----------+--------------+-----------+
|   537226|    22811|SET OF 6 T-LIGHTS...|       6|2010-12-06 08:34:00
|     2.95|   15987.0|United Kingdom|     Monday|
|   537226|    21713|CITRONELLA CANDLE...|       8|2010-12-06 08:34:00
|      2.1|   15987.0|United Kingdom|     Monday|
|   537226|    22927|GREEN GIANT GARDE...|       2|2010-12-06 08:34:00
|     5.95|   15987.0|United Kingdom|     Monday|
+---------+---------+--------------------+--------+--------------------
+---------+----------+--------------+-----------+
only showing top 3 rows
```

In [37]:
```
# COMMAND ----------

from pyspark.ml.feature import StringIndexer

indexer = StringIndexer()\
  .setInputCol("day_of_week")\
  .setOutputCol("day_of_week_index")
```

In [38]:
```
# COMMAND ----------

from pyspark.ml.feature import OneHotEncoder

encoder = OneHotEncoder()\
  .setInputCol("day_of_week_index")\
  .setOutputCol("day_of_week_encoded")
```

In [39]:
```
# COMMAND ----------

from pyspark.ml.feature import VectorAssembler

vectorAssembler = VectorAssembler()\
  .setInputCols(["UnitPrice", "Quantity", "day_of_week_encoded"])\
  .setOutputCol("features")
```

In [40]:
```
# COMMAND ----------

from pyspark.ml import Pipeline

transformationPipeline = Pipeline()\
  .setStages([indexer, encoder, vectorAssembler])
```

In [41]:
```
# COMMAND ----------

fittedPipeline = transformationPipeline.fit(trainDataFrame)
```

In [45]:
```
# COMMAND ----------

transformedTraining = fittedPipeline.transform(trainDataFrame)
```

In [46]:
```
transformedTraining.show(5)
```

```
+---------+---------+--------------------+--------+--------------------
+---------+----------+--------------+-----------+-----------------+---
----------------+--------------------+
|InvoiceNo|StockCode|         Description|Quantity|        InvoiceDate
|UnitPrice|CustomerID|       Country|day_of_week|day_of_week_index|day
_of_week_encoded|            features|
+---------+---------+--------------------+--------+--------------------
+---------+----------+--------------+-----------+-----------------+---
----------------+--------------------+
|   537226|    22811|SET OF 6 T-LIGHTS...|       6|2010-12-06 08:34:00
|     2.95|   15987.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[2.95,...|
|   537226|    21713|CITRONELLA CANDLE...|       8|2010-12-06 08:34:00
|      2.1|   15987.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[2.1,8...|
|   537226|    22927|GREEN GIANT GARDE...|       2|2010-12-06 08:34:00
|     5.95|   15987.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[5.95,...|
|   537226|    20802|SMALL GLASS SUNDA...|       6|2010-12-06 08:34:00
|     1.65|   15987.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[1.65,...|
|   537226|    22052|VINTAGE CARAVAN G...|      25|2010-12-06 08:34:00
|     0.42|   15987.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[0.42,...|
+---------+---------+--------------------+--------+--------------------
+---------+----------+--------------+-----------+-----------------+---
----------------+--------------------+
only showing top 5 rows
```

Spark ML Clustering

In [47]:
```
# COMMAND ----------

from pyspark.ml.clustering import KMeans

kmeans = KMeans()\
  .setK(20)\
  .setSeed(10)
```

In [48]:
```
# COMMAND ----------

kmModel = kmeans.fit(transformedTraining)
```

In [49]:
```
type(kmModel)
```

Out[49]:
```
pyspark.ml.clustering.KMeansModel
```

In [50]:
```
kmModel.summary
```

Out[50]:
```
<pyspark.ml.clustering.KMeansSummary at 0x7f7ff583f5c0>
```

In [51]:
```
# COMMAND ----------

transformedTest = fittedPipeline.transform(testDataFrame)
```

In [52]:
```
transformedTest.show(5)
```

```
+---------+---------+--------------------+--------+------------------
+---------+----------+--------------+-----------+-----------------+---
----------------+--------------------+
|InvoiceNo|StockCode|         Description|Quantity|        InvoiceDate
|UnitPrice|CustomerID|       Country|day_of_week|day_of_week_index|day
_of_week_encoded|            features|
+---------+---------+--------------------+--------+------------------
+---------+----------+--------------+-----------+-----------------+---
----------------+--------------------+
|   580538|    23084|  RABBIT NIGHT LIGHT|      48|2011-12-05 08:38:00
|     1.79|   14075.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[1.79,...|
|   580538|    23077| DOUGHNUT LIP GLOSS |      20|2011-12-05 08:38:00
|     1.25|   14075.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[1.25,...|
|   580538|    22906|12 MESSAGE CARDS ...|      24|2011-12-05 08:38:00
|     1.65|   14075.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[1.65,...|
|   580538|    21914|BLUE HARMONICA IN...|      24|2011-12-05 08:38:00
|     1.25|   14075.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[1.25,...|
|   580538|    22467|   GUMBALL COAT RACK|       6|2011-12-05 08:38:00
|     2.55|   14075.0|United Kingdom|     Monday|              2.0|
(5,[2],[1.0])|(7,[0,1,4],[2.55,...|
+---------+---------+--------------------+--------+------------------
+---------+----------+--------------+-----------+-----------------+---
----------------+--------------------+
only showing top 5 rows
```

In [ ]:
```

```