In [1]:
```python
from IPython.display import display, clear_output
```

In [2]:
```python
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 [4]:
```python
file_path = SPARK_BOOK_DATA_PATH + "/data/retail-data/by-day/*.csv"

retail_df = spark.read.csv(file_path, header=True, inferSchema=True)
```

```python
retail_df = spark.read\
  .format("csv")\
  .option("header", "true")\
  .option("inferSchema", "true")\
  .load(file_path)
```

In [5]:
```python
retail_df.count()
```

Out[5]: 541909

In [6]:
```python
retail_df.show(5,False)
```

```
+---------+---------+-------------------------------+--------+-------------------+---------+-------
---+--------------+
|InvoiceNo|StockCode|Description                    |Quantity|InvoiceDate        |UnitPrice|Custome
rID|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 WITH ENVELOPES|24      |2011-12-05 08:38:00|1.65     |14075.0
|United Kingdom|
|580538   |21914    |BLUE HARMONICA IN BOX          |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|
+---------+---------+-------------------------------+--------+-------------------+---------+-------
---+--------------+
only showing top 5 rows
```

In [7]:
```
display(retail_df.toPandas())
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	580538	23084	RABBIT NIGHT LIGHT	48	2011-12-05 08:38:00	1.79	14075.0	United Kingdom
1	580538	23077	DOUGHNUT LIP GLOSS	20	2011-12-05 08:38:00	1.25	14075.0	United Kingdom
2	580538	22906	12 MESSAGE CARDS WITH ENVELOPES	24	2011-12-05 08:38:00	1.65	14075.0	United Kingdom
3	580538	21914	BLUE HARMONICA IN BOX	24	2011-12-05 08:38:00	1.25	14075.0	United Kingdom
4	580538	22467	GUMBALL COAT RACK	6	2011-12-05 08:38:00	2.55	14075.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	543282	22849	BREAD BIN DINER STYLE MINT	1	2011-02-06 16:08:00	16.95	12956.0	United Kingdom
541905	543282	84879	ASSORTED COLOUR BIRD ORNAMENT	8	2011-02-06 16:08:00	1.69	12956.0	United Kingdom
541906	543282	84659A	WHITE TRAVEL ALARM CLOCK	1	2011-02-06 16:08:00	2.55	12956.0	United Kingdom
541907	543282	82484	WOOD BLACK BOARD ANT WHITE FINISH	1	2011-02-06 16:08:00	7.95	12956.0	United Kingdom
541908	543282	22168	ORGANISER WOOD ANTIQUE WHITE	1	2011-02-06 16:08:00	8.50	12956.0	United Kingdom

541909 rows × 8 columns

In [8]:
```
retail_df.createOrReplaceTempView("retail_table")
# retail_table is a SQL table for query
```

In [9]:
```
staticSchema = retail_df.schema
```

In [10]:
```
print(staticSchema)
```

```
StructType(List(StructField(InvoiceNo,StringType,true),StructField(StockCode,StringType,true),Struc
tField(Description,StringType,true),StructField(Quantity,IntegerType,true),StructField(InvoiceDate,
StringType,true),StructField(UnitPrice,DoubleType,true),StructField(CustomerID,DoubleType,true),Str
uctField(Country,StringType,true)))
```

In [11]:
```
retail_df.printSchema()
```

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

In [12]:
```
display(retail_df.describe().toPandas())
```

	summary	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	Custom
0	count	541909	541909	540455	541909	541909	541909	40
1	mean	559965.752026781	27623.240210938104	20713.0	9.55224954743324	None	4.611113626089641	15287.69057023
2	stddev	13428.417280796697	16799.737628427683	NaN	218.0811578502335	None	96.75985306117963	1713.60030332
3	min	536365	10002	4 PURPLE FLOCK DINNER CANDLES	-80995	2010-12-01 08:26:00	-11062.06	123
4	max	C581569	m	wrongly sold sets	80995	2011-12-09 12:50:00	38970.0	182

In [13]:
```
df = spark.sql("select * from retail_table limit 5")
```

In [14]:

```
df.show(truncate=False)   # disable truncate to show description in full
```

```
+---------+---------+-------------------------------+--------+-------------------+---------+-------
---+--------------+
|InvoiceNo|StockCode|Description                    |Quantity|InvoiceDate        |UnitPrice|Custome
rID|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 WITH ENVELOPES|24      |2011-12-05 08:38:00|1.65     |14075.0
|United Kingdom|
|580538   |21914    |BLUE HARMONICA IN BOX          |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 [15]:

```python
# 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, False)
)
```

```
+----------+-----------------------------------------+------------------+
|CustomerId|window                                   |sum(total_cost)   |
+----------+-----------------------------------------+------------------+
|17450.0   |[2011-09-19 20:00:00, 2011-09-20 20:00:00]|71601.44          |
|null      |[2011-11-13 19:00:00, 2011-11-14 19:00:00]|55316.08          |
|null      |[2011-11-06 19:00:00, 2011-11-07 19:00:00]|42939.17          |
|null      |[2011-03-28 20:00:00, 2011-03-29 20:00:00]|33521.39999999998 |
|null      |[2011-12-07 19:00:00, 2011-12-08 19:00:00]|31975.590000000007|
+----------+-----------------------------------------+------------------+
only showing top 5 rows
```

In [16]:

```python
# col() can be omitted
(
retail_df.selectExpr(
    "CustomerId",
    "(UnitPrice * Quantity) as total_cost",
    "InvoiceDate")
  .groupBy("CustomerId", window("InvoiceDate", "1 day"))
  .sum("total_cost")
  .withColumnRenamed("sum(total_cost)", "sum_total_cost")
  .withColumnRenamed("window", "InvoiceDateWindow")
  .sort(desc("sum_total_cost"))
  .withColumn("sum_total_cost", F.round("sum_total_cost",2))
  .show(5, truncate=False)
)
```

```
+----------+-----------------------------------------+--------------+
|CustomerId|InvoiceDateWindow                        |sum_total_cost|
+----------+-----------------------------------------+--------------+
|17450.0   |[2011-09-19 20:00:00, 2011-09-20 20:00:00]|71601.44      |
|null      |[2011-11-13 19:00:00, 2011-11-14 19:00:00]|55316.08      |
|null      |[2011-11-06 19:00:00, 2011-11-07 19:00:00]|42939.17      |
|null      |[2011-03-28 20:00:00, 2011-03-29 20:00:00]|33521.4       |
|null      |[2011-12-07 19:00:00, 2011-12-08 19:00:00]|31975.59      |
+----------+-----------------------------------------+--------------+
only showing top 5 rows
```

Spark Streaming

In [17]:

```python
## Extract
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 [18]:

```python
## Transform
# cost per day
purchaseByCustomerPerDay = (streamingDataFrame
  .selectExpr(
    "CustomerId",
    "(UnitPrice * Quantity) as total_cost",
    "InvoiceDate")
  .groupBy(col("CustomerId"), window(col("InvoiceDate"), "1 day"))
  .sum("total_cost")
)
```

In [19]:

```python
## Load
# store result into a SQL table `customer_purchases` is specified by `queryName`
(
purchaseByCustomerPerDay
    .writeStream
    .queryName("customer_purchases")
    .format("memory")
    .outputMode("complete")
    .start()
)
```

Out[19]:

```
<pyspark.sql.streaming.StreamingQuery at 0x7f452f78dd90>
```

In [26]:

```
## Monitor stream
# use `Ctrl-Enter` to execute below cell repeatly to see streaming result as more data are read
spark.sql("""
  SELECT
      *
  FROM customer_purchases
  ORDER BY `sum(total_cost)` DESC
  """).show(5, False)
```

```
+----------+-----------------------------------------+------------------+
|CustomerId|window                                   |sum(total_cost)   |
+----------+-----------------------------------------+------------------+
|null      |[2011-11-06 19:00:00, 2011-11-07 19:00:00]|42939.17          |
|null      |[2011-07-03 20:00:00, 2011-07-04 20:00:00]|13667.65999999993 |
|18102.0   |[2011-07-03 20:00:00, 2011-07-04 20:00:00]|13282.0           |
|null      |[2011-11-21 19:00:00, 2011-11-22 19:00:00]|13216.889999999894|
|null      |[2010-11-30 19:00:00, 2010-12-01 19:00:00]|12584.299999999988|
+----------+-----------------------------------------+------------------+
only showing top 5 rows
```

In [21]:

```
spark.sql("""
  SELECT *
  FROM customer_purchases
  ORDER BY `sum(total_cost)` DESC
  """).show(5,truncate=False)
```

```
+----------+-----------------------------------------+------------------+
|CustomerId|window                                   |sum(total_cost)   |
+----------+-----------------------------------------+------------------+
|12678.0   |[2011-10-27 20:00:00, 2011-10-28 20:00:00]|8947.960000000005 |
|null      |[2011-05-18 20:00:00, 2011-05-19 20:00:00]|4012.6600000000067|
|13694.0   |[2011-10-27 20:00:00, 2011-10-28 20:00:00]|3304.030000000001 |
|null      |[2011-10-27 20:00:00, 2011-10-28 20:00:00]|3270.980000000003 |
|13199.0   |[2011-10-27 20:00:00, 2011-10-28 20:00:00]|1912.7999999999997|
+----------+-----------------------------------------+------------------+
only showing top 5 rows
```

Spark ML Pipeline

In [30]:

```python
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, truncate=False)
```

```
+---------+---------+-------------------------------+--------+-------------------+---------+-------
---+--------------+-----------+
|InvoiceNo|StockCode|Description                    |Quantity|InvoiceDate        |UnitPrice|Custome
rID|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 WITH ENVELOPES|24      |2011-12-05 08:38:00|1.65     |14075.0
|United Kingdom|Monday     |
+---------+---------+-------------------------------+--------+-------------------+---------+-------
---+--------------+-----------+
only showing top 3 rows
```

In [32]:

```python
preppedDataFrame.where(F.isnull(F.col("InvoiceDate"))).show(4)
```

```
+---------+---------+-----------+--------+-----------+---------+----------+-------+-----------+
|InvoiceNo|StockCode|Description|Quantity|InvoiceDate|UnitPrice|CustomerID|Country|day_of_week|
+---------+---------+-----------+--------+-----------+---------+----------+-------+-----------+
+---------+---------+-----------+--------+-----------+---------+----------+-------+-----------+
```

In [33]:

```python
# split data into (train,test)
trainDataFrame = preppedDataFrame.where("InvoiceDate < '2011-07-01'")

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

In [34]:

```
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 [36]:
```
display(testDataFrame.toPandas())
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	day_of_week
0	580538	23084	RABBIT NIGHT LIGHT	48	2011-12-05 08:38:00	1.79	14075.0	United Kingdom	Monday
1	580538	23077	DOUGHNUT LIP GLOSS	20	2011-12-05 08:38:00	1.25	14075.0	United Kingdom	Monday
2	580538	22906	12 MESSAGE CARDS WITH ENVELOPES	24	2011-12-05 08:38:00	1.65	14075.0	United Kingdom	Monday
3	580538	21914	BLUE HARMONICA IN BOX	24	2011-12-05 08:38:00	1.25	14075.0	United Kingdom	Monday
4	580538	22467	GUMBALL COAT RACK	6	2011-12-05 08:38:00	2.55	14075.0	United Kingdom	Monday
...	...	...	...	...	...	...	...	...	...
296001	562595	84818	DANISH ROSE PHOTO FRAME	24	2011-08-07 15:52:00	0.79	17602.0	United Kingdom	Sunday
296002	562595	47343A	FUSCHIA FLOWER PURSE WITH BEADS	12	2011-08-07 15:52:00	0.83	17602.0	United Kingdom	Sunday
296003	562595	15044C	PURPLE PAPER PARASOL	6	2011-08-07 15:52:00	2.95	17602.0	United Kingdom	Sunday
296004	562595	15044D	RED PAPER PARASOL	12	2011-08-07 15:52:00	2.95	17602.0	United Kingdom	Sunday
296005	562595	15044A	PINK PAPER PARASOL	6	2011-08-07 15:52:00	2.95	17602.0	United Kingdom	Sunday

296006 rows × 9 columns

In [37]:

```python
# pre-processing features

from pyspark.ml.feature import StringIndexer
from pyspark.ml.feature import OneHotEncoder
from pyspark.ml.feature import VectorAssembler

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

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

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

In [38]:

```python
# setup pipeline

from pyspark.ml import Pipeline

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

In [39]:

```python
# run pipeline on train data

fittedPipeline = transformationPipeline.fit(trainDataFrame)
```

In [40]:

```python
# verify on train data

transformedTraining = fittedPipeline.transform(trainDataFrame)
```

In [41]:
```
transformedTraining.show(3, truncate=False, vertical=True)
```

```
-RECORD 0-----------------------------------------
 InvoiceNo           | 537226
 StockCode           | 22811
 Description         | SET OF 6 T-LIGHTS CACTI
 Quantity            | 6
 InvoiceDate         | 2010-12-06 08:34:00
 UnitPrice           | 2.95
 CustomerID          | 15987.0
 Country             | United Kingdom
 day_of_week         | Monday
 day_of_week_index   | 2.0
 day_of_week_encoded | (5,[2],[1.0])
 features            | (7,[0,1,4],[2.95,6.0,1.0])
-RECORD 1-----------------------------------------
 InvoiceNo           | 537226
 StockCode           | 21713
 Description         | CITRONELLA CANDLE FLOWERPOT
 Quantity            | 8
 InvoiceDate         | 2010-12-06 08:34:00
```

Spark ML Clustering

In [42]:
```
# COMMAND ----------

from pyspark.ml.clustering import KMeans

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

In [43]:
```
# COMMAND ----------

kmModel = kmeans.fit(transformedTraining)
```

In [44]:
```
type(kmModel)
```

Out[44]:
```
pyspark.ml.clustering.KMeansModel
```

In [45]:
```
kmModel.summary
```

Out[45]:
```
<pyspark.ml.clustering.KMeansSummary at 0x7f452f764a60>
```

In [46]:
```
spark.stop()
```

In [ ]:
```

```