

Spark DataFrames Project Exercise

Let's get some quick practice with your new Spark DataFrame skills, you will be asked some basic questions about some stock market data, in this case Walmart Stock from the years 2012-2017. This exercise will just ask a bunch of questions, unlike the future machine learning exercises, which will be a little looser and be in the form of "Consulting Projects", but more on that later!

For now, just answer the questions and complete the tasks below.

Use the `walmart_stock.csv` file to Answer and complete the tasks below!

Start a simple Spark Session

```
In [1]: from pyspark.sql import SparkSession
from pyspark.sql.functions import avg, count, sum, max, min
from pyspark.sql.functions import month, year
#from pyspark.sql.types import *
spark = SparkSession.\
    builder.\
    appName("Sparklab_2").\
    getOrCreate()
```

Starting spark application

ID	YARN Application ID	Kind	State	Spark UI	Driver log	Current session?
2	application_1649004799305_0006	pyspark3	idle	Link	Link	✓

SparkSession available as 'spark'.

Load the Walmart Stock CSV File, have Spark infer the data types.

```
In [2]: data_file = '/HdiNotebooks/walmart_stock.csv'
df1=spark.read\
    .option("header","true")\
    .option("inferSchema","true")\
```

Spark DataFrames Project Exercise-Copy1 Last Checkpoint: in 7 hours (autosaved) [Feedback](#)

File Edit View Insert Cell Kernel Widgets Help | PySpark3

Load the Walmart Stock CSV File, have Spark infer the data types.

```
In [2]: data_file = '/HdiNotebooks/walmart_stock.csv'
df1=spark.read\
    .option("header","true")\
    .option("inferSchema","true")\
    .csv(data_file)
```

What are the column names?

```
In [3]: df1.columns

['Date', 'Open', 'High', 'Low', 'Close', 'Volume', 'Adj Close']
```

What does the Schema look like?

```
In [4]: df1.printSchema()

root
|-- Date: string (nullable = true)
|-- Open: double (nullable = true)
|-- High: double (nullable = true)
|-- Low: double (nullable = true)
|-- Close: double (nullable = true)
|-- Volume: integer (nullable = true)
|-- Adj Close: double (nullable = true)
```

Print out the first 5 columns.

Spark DataFrames Project Exercise-Copy1 Last Checkpoint: in 7 hours (autosaved) [Feedback](#)

File Edit View Insert Cell Kernel Widgets Help | PySpark3

Print out the first 5 columns.

```
In [5]: df1.head(5)

[Row(Date='2012-01-03', Open=59.970001, High=61.060001, Low=59.869999, Close=60.330002, Volume=12668800, Adj Close=52.619234999999996), Row(Date='2012-01-04', Open=60.209998999999996, High=60.349998, Low=59.470001, Close=59.709998999999996, Volume=9593300, Adj Close=52.078475), Row(Date='2012-01-05', Open=59.349998, High=59.619999, Low=58.369999, Close=59.419998, Volume=1276800, Adj Close=51.825538), Row(Date='2012-01-06', Open=59.419998, High=59.450001, Low=58.869999, Close=59.0, Volume=8069400, Adj Close=51.45922), Row(Date='2012-01-09', Open=59.029999, High=59.549999, Low=58.919998, Close=59.18, Volume=6679300, Adj Close=51.616215000000004)]
```

Use describe() to learn about the DataFrame.

```
In [6]: df1.describe().show()
```

	summary	Date	Open	High	Low	Close	Volume	Adj Clos
count	1258	1258	1258	1258	1258	1258	1258	1258
mean	null	72.35785375357709	72.83938807631165	71.9186009594594	72.38844998012726	8222093.481717011	67.2388384872814	
stddev	null	6.76809024470826	6.768186808159218	6.744075756255496	6.756859163732991	4519780.8431556	6.72260944999685	
min	2012-01-03	56.389998999999996	57.060001	56.299999	56.419998	2094900	50.36368	
max	2016-12-30	90.800003	90.970001	89.25	90.470001	80898100	84.9142160000000	

Bonus Question!

There are too many decimal places for mean and stddev in the describe() dataframe. Format the numbers to just show up to two decimal places. Pay careful attention to the datatypes that describe() returns, we didn't cover how to do this exact formatting, but we covered something very similar. [Check this link for a hint](#)

If you get stuck on this, don't worry, just view the solutions.

```
In [7]: from pyspark.sql.functions import col
        from pyspark.sql.types import DecimalType

        df1_1 = df1.describe()
        df1_1.printSchema()
```

```
root
|-- summary: string (nullable = true)
|-- Date: string (nullable = true)
|-- Open: string (nullable = true)
|-- High: string (nullable = true)
|-- Low: string (nullable = true)
|-- Close: string (nullable = true)
|-- Volume: string (nullable = true)
|-- Adj Close: string (nullable = true)
```

```
In [9]: df1.select(col("summary"),
                  col("Open").cast(DecimalType(10,2)),
                  col("High").cast(DecimalType(10,2)),
                  col("Low").cast(DecimalType(10,2)),
                  col("close").cast(DecimalType(10,2)),
                  col("volume").cast(DecimalType(10)))
        .show()
```

summary	Open	High	low	close	volume
count	1258.00	1258.00	1258.00	1258.00	1258
mean	72.36	72.84	71.92	72.39	8222893
stddev	6.77	6.77	6.74	6.76	4519781
min	56.39	57.06	56.38	56.42	2894900
max	90.80	90.97	89.25	90.47	88898100

Create a new dataframe with a column called HV Ratio that is the ratio of the High Price versus volume of stock traded for a day.

```
In [10]: df2 = df1.withColumn("HV Ratio",df1['High']/df1['Volume'])
df2.select("HV Ratio").show()
```

	HV Ratio
4.819714653321546E-6	
6.290848613094555E-6	
4.669412994783916E-6	
7.367338463826307E-6	
8.915504778943901E-6	
8.644477436914568E-6	
9.351828421515645E-6	
8.29141562102703E-6	
7.712212102001476E-6	
7.07164825329412E-6	
1.015495466386981E-5	
6.57635414663629E-6	
5.90145296180676E-6	
8.547679455011844E-6	
8.420789512685392E-6	
1.041448341728929E-5	
8.316075414862431E-6	
9.721138314992126E-6	
8.029436027787578E-6	
6.307432259386365E-6	

only showing top 20 rows

Microsoft Azure X Spark DataFrames Project Exercis X Untitled X | +

https://sparklab.azurehdinsight.net/jupyter/notebooks/Spark%20DataFrames%20Project%20... A 🔍

File Edit View Insert Cell Kernel Widgets Help

+ % 🔍 ↶ ↷ ⏮ ⏭ ⏪ ⏩ ↺ ⏹ Markdown CellToolbar 📄 📁 📺

What day had the Peak High in Price?

```
In [19]: from pyspark.sql.window import Window
from pyspark.sql.functions import col, row_number, count, rank, desc
windowSpec = Window.partitionBy().orderBy(desc("High"))
top_rec=df1.withColumn("rank",rank().over(windowSpec)).head(1)

from datetime import datetime
top_date= top_rec[0]['Date']
dt_obj = datetime.strptime(top_date, '%Y-%S-%d')
dt_obj

datetime.datetime(2015, 1, 13, 0, 0, 1)
```

What is the mean of the Close column?

```
In [11]: df1.select(avg("Close")).show()
```

```
+-----+
|      avg(Close)|
+-----+
|72.38844998012726|
+-----+
```

What is the max and min of the Volume column?

```
In [12]: df1.select(max(col("Volume")),min(col("Volume"))).show()
```

```
+-----+-----+
|max(Volume)|min(Volume)|
+-----+-----+
|  80898100|   2094900|
+-----+-----+
```

https://sparklab.azurehdinsight.net/jupyter/notebooks/Spark%20DataFrames%20Project%2...

File Edit View Insert Cell Kernel Widgets Help | PySpark

How many days was the Close lower than 60 dollars?

```
In [13]: df1.filter(df1["Close"]<60).count()
81
```

What percentage of the time was the High greater than 80 dollars ?

In other words, (Number of Days High>80)/(Total Days in the dataset)

```
In [14]: df1.filter(df1["High"]>80).count()/df1.count()*100
9.141494435612083
```

What is the Pearson correlation between High and Volume?

[Hint](#)

```
In [15]: df1.stat.corr("High", "Volume")
-0.3384326061737161
```

What is the max High per year?

```
In [16]: df2 = df1.withColumn("Year",year(df1['Date']))
df2.groupBy("Year").agg(max("High")).show()
+----+-----+
|Year|max(High)|
+----+-----+
```

https://sparklab.azurehdinsight.net/jupyter/notebooks/Spark%20DataFrames%20Project%2...

File Edit View Insert Cell Kernel Widgets Help | PySpark3

What day had the Peak High in Price?

```
In [23]: from pyspark.sql.window import Window
from pyspark.sql.functions import col, row_number ,count,rank, desc
windowSpec = Window.partitionBy().orderBy(desc("High"))
top_rec=df1.withColumn("rank",rank().over(windowSpec)).head(1)

from datetime import datetime
top_date= top_rec[0]['Date']
dt_obj = datetime.strptime(top_date, '%Y-%S-%d')
dt_obj
```

Command finished at 04-03-2022 12:48:28.933 -07:00, execution took 3s 307ms

Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
30	head	COMPLETED	2/2		a minute ago	1s

datetime.datetime(2015, 1, 13, 0, 0, 1)

https://sparklab.azurehdinsight.net/jupyter/notebooks/Spark%20DataFrames%20Project%2...

File Edit View Insert Cell Kernel Widgets Help | PySpark3

What is the max High per year?

```
In [16]: df2 = df1.withColumn("Year",year(df1['Date']))
df2.groupBy("Year").agg(max("High")).show()
```

```
+-----+
|Year|max(High)|
+-----+
|2015|90.970001|
|2013|81.370003|
|2014|88.089996|
|2012|77.599998|
|2016|75.190002|
+-----+
```

What is the average Close for each Calendar Month?

In other words, across all the years, what is the average Close price for Jan,Feb, Mar, etc... Your result will have a value for each of these months.

```
In [18]: df2= df1.withColumn("Month",month(df1['Date']))
df2.groupBy("Month").agg(avg("Close")).orderBy("Month").show()
```

```
+-----+
|Month|      avg(Close)|
+-----+
|1|71.44801958415842|
|2| 71.306804443299|
|3|71.77794377570092|
|4|72.97361900952382|
|5|72.30971688679247|
|6| 72.4953774245283|
|7|74.43971943925233|
|8|73.02981855454546|
|9|72.18411785294116|
|10|71.57854545454543|
|11| 72.1110893069307|
|12|72.84792478301885|
+-----+
```

----- Additional screenshot for some questions with running information -----

https://sparklab.azurehdinsight.net/jupyter/notebooks/Spark%20DataFrames%20Project%2...

Edit View Insert Cell Kernel Widgets Help | PySpark3

What day had the Peak High in Price?

```
23]: from pyspark.sql.window import Window
from pyspark.sql.functions import col, row_number ,count,rnk, desc
windowSpec = Window.partitionBy().orderBy(desc("High"))
top_rec=df1.withColumn("rank",rnk().over(windowSpec)).head(1)

from datetime import datetime
top_date= top_rec[0]['Date']
dt_obj = datetime.strptime(top_date, '%Y-%S-%d')
dt_obj
```

Command finished at 04-03-2022 12:46:29.933 -07:00, execution took 3s 307ms

Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
30	head	COMPLETED	2/2		9 minutes ago	1s

Jobs: 1 COMPLETED Spark: 4 EXECUTORS 3 CORES X

datetime.datetime(2015, 1, 13, 0, 0, 1)

1:


```
In [26]: df1_1.select(col("summary"),
                    col("Open").cast(DecimalType(10,2)),
                    col("High").cast(DecimalType(10,2)),
                    col("low").cast(DecimalType(10,2)),
                    col("close").cast(DecimalType(10,2)),
                    col("Volume").cast(DecimalType(10))
                    ).show()
```

Command finished at 04-03-2022 12:51:44.383 -07:00, execution took 240ms

summary	Open	High	low	close	Volume
count	1258.00	1258.00	1258.00	1258.00	1258
mean	72.36	72.84	71.92	72.39	8222093
stddev	6.77	6.77	6.74	6.76	4519781
min	56.39	57.06	56.30	56.42	2094900
max	90.80	90.97	89.25	90.47	80898100