```python
#1. Start a Simple Spark Session
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("WalmartStock").getOrCreate()

#2. Load the Walmart Stock CSV File and Infer Data Types
df = spark.read.csv("18_6_walmart_stock.csv", inferSchema=True,
header=True)

#3. Get Column Names
df.columns
```

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

```python
#4. Print the Schema
df.printSchema()
```

```
root
 |-- Date: date (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)
```

```python
#5. Print First 5 Rows
df.show(5)
```

```
+----------+------------------+---------+---------+------------------
+--------+------------------+
|      Date|              Open|     High|      Low|             Close|
Volume|         Adj Close|
+----------+------------------+---------+---------+------------------
+--------+------------------+
|2012-01-03|         59.970001|61.060001|59.869999|         60.330002|
12668800|52.619234999999996|
|2012-01-04|60.209998999999996|60.349998|59.470001|59.709998999999996|
9593300|         52.078475|
|2012-01-05|         59.349998|59.619999|58.369999|         59.419998|
12768200|         51.825539|
|2012-01-06|         59.419998|59.450001|58.869999|              59.0|
8069400|          51.45922|
|2012-01-09|         59.029999|59.549999|58.919998|             59.18|
6679300|51.616215000000004|
+----------+------------------+---------+---------+------------------
+--------+------------------+
only showing top 5 rows
```

```
#6. Use describe() to Learn About the DataFrame
df.describe().show()
```

```
+-------+------------------+-----------------+-----------------
+-----------------+-----------------+-----------------+
|summary|              Open|             High|              Low|
Close|           Volume|        Adj Close|
+-------+------------------+-----------------+-----------------
+-----------------+-----------------+-----------------+
|  count|              1258|             1258|             1258|
1258|             1258|             1258|
|   mean| 72.35785375357709|72.83938807631165| 71.9186009594594|
72.38844998012726|8222093.481717011|67.23883848728146|
| stddev|  6.76809024470826|6.768186808159218|6.744075756255496|
6.756859163732991|  4519780.8431556|6.722609449996857|
|    min|56.389998999999996|        57.060001|        56.299999|
56.419998|          2094900|        50.363689|
|    max|         90.800003|        90.970001|            89.25|
90.470001|         80898100|84.91421600000001|
+-------+------------------+-----------------+-----------------
+-----------------+-----------------+-----------------+
```

```
#7. Bonus Question - Format Numbers to Two Decimal Places
from pyspark.sql.functions import format_number
desc_df = df.describe()
for col_name in ["Open", "High", "Low", "Close", "Volume", "Adj
Close"]:
    desc_df = desc_df.withColumn(col_name,
format_number(desc_df[col_name].cast("float"), 2))
    desc_df.show()
```

```
+-------+--------+-----------------+-----------------
+-----------------+-----------------+-----------------+
|summary|    Open|             High|              Low|
Close|           Volume|        Adj Close|
+-------+--------+-----------------+-----------------
+-----------------+-----------------+-----------------+
|  count|1,258.00|             1258|             1258|
1258|             1258|             1258|
|   mean|   72.36|72.83938807631165| 71.9186009594594|
72.38844998012726|8222093.481717011|67.23883848728146|
| stddev|    6.77|6.768186808159218|6.744075756255496|
6.756859163732991|  4519780.8431556|6.722609449996857|
|    min|   56.39|        57.060001|        56.299999|
56.419998|          2094900|        50.363689|
|    max|   90.80|        90.970001|            89.25|
90.470001|         80898100|84.91421600000001|
+-------+--------+-----------------+-----------------
+-----------------+-----------------+-----------------+
```

```
+-------+--------+--------+-----------------+-----------------
+-----------------+-----------------+
|summary|    Open|    High|              Low|            Close|
Volume|        Adj Close|
+-------+--------+--------+-----------------+-----------------
+-----------------+-----------------+
|  count|1,258.00|1,258.00|             1258|             1258|
1258|             1258|
|   mean|   72.36|   72.84| 71.9186009594594|72.38844998012726|
8222093.481717011|67.23883848728146|
| stddev|    6.77|    6.77|6.744075756255496|6.756859163732991|
4519780.8431556|6.722609449996857|
|    min|   56.39|   57.06|        56.299999|        56.419998|
2094900|        50.363689|
|    max|   90.80|   90.97|            89.25|        90.470001|
80898100|84.91421600000001|
+-------+--------+--------+-----------------+-----------------
+-----------------+-----------------+

+-------+--------+--------+--------+-----------------
+-----------------+-----------------+
|summary|    Open|    High|     Low|            Close|
Volume|        Adj Close|
+-------+--------+--------+--------+-----------------
+-----------------+-----------------+
|  count|1,258.00|1,258.00|1,258.00|             1258|
1258|             1258|
|   mean|   72.36|   72.84|   71.92|72.38844998012726|
8222093.481717011|67.23883848728146|
| stddev|    6.77|    6.77|    6.74|6.756859163732991|
4519780.8431556|6.722609449996857|
|    min|   56.39|   57.06|   56.30|        56.419998|
2094900|        50.363689|
|    max|   90.80|   90.97|   89.25|        90.470001|
80898100|84.91421600000001|
+-------+--------+--------+--------+-----------------
+-----------------+-----------------+

+-------+--------+--------+--------+--------+-----------------
+-----------------+
|summary|    Open|    High|     Low|   Close|           Volume|
Adj Close|
+-------+--------+--------+--------+--------+-----------------
+-----------------+
|  count|1,258.00|1,258.00|1,258.00|1,258.00|             1258|
1258|
|   mean|   72.36|   72.84|   71.92|   72.39|8222093.481717011|
67.23883848728146|
| stddev|    6.77|    6.77|    6.74|    6.76|  4519780.8431556|
```

```
6.722609449996857|
|    min|   56.39|   57.06|   56.30|   56.42|          2094900|
50.363689|
|    max|   90.80|   90.97|   89.25|   90.47|         80898100|
84.91421600000001|
+-------+--------+--------+--------+--------+-----------------
+-----------------+

+-------+--------+--------+--------+--------+-------------
+-----------------+
|summary|    Open|    High|     Low|   Close|       Volume|         Adj
Close|
+-------+--------+--------+--------+--------+-------------
+-----------------+
|  count|1,258.00|1,258.00|1,258.00|1,258.00|     1,258.00|
1258|
|   mean|   72.36|   72.84|   71.92|   72.39| 8,222,093.50|
67.23883848728146|
| stddev|    6.77|    6.77|    6.74|    6.76| 4,519,781.00|
6.722609449996857|
|    min|   56.39|   57.06|   56.30|   56.42| 2,094,900.00|
50.363689|
|    max|   90.80|   90.97|   89.25|   90.47|80,898,096.00|
84.91421600000001|
+-------+--------+--------+--------+--------+-------------
+-----------------+

+-------+--------+--------+--------+--------+-------------+---------+
|summary|    Open|    High|     Low|   Close|       Volume|Adj Close|
+-------+--------+--------+--------+--------+-------------+---------+
|  count|1,258.00|1,258.00|1,258.00|1,258.00|     1,258.00| 1,258.00|
|   mean|   72.36|   72.84|   71.92|   72.39| 8,222,093.50|    67.24|
| stddev|    6.77|    6.77|    6.74|    6.76| 4,519,781.00|     6.72|
|    min|   56.39|   57.06|   56.30|   56.42| 2,094,900.00|    50.36|
|    max|   90.80|   90.97|   89.25|   90.47|80,898,096.00|    84.91|
+-------+--------+--------+--------+--------+-------------+---------+


#8. Create a New Column HV Ratio
from pyspark.sql.functions import col
df = df.withColumn("HV Ratio", col("High") / col("Volume"))
df.show(5)

+----------+------------------+---------+---------+-----------------
+--------+------------------+--------------------+
|      Date|              Open|     High|      Low|            Close|
Volume|         Adj Close|            HV Ratio|
+----------+------------------+---------+---------+-----------------
+--------+------------------+--------------------+
|2012-01-03|         59.970001|61.060001|59.869999|         60.330002|
```

```
12668800|52.619234999999996|4.819714653321546E-6|
|2012-01-04|60.209998999999996|60.349998|59.470001|59.709998999999996|
9593300|         52.078475|6.290848613094555E-6|
|2012-01-05|         59.349998|59.619999|58.369999|         59.419998|
12768200|         51.825539|4.669412994783916E-6|
|2012-01-06|         59.419998|59.450001|58.869999|              59.0|
8069400|          51.45922|7.367338463826307E-6|
|2012-01-09|         59.029999|59.549999|58.919998|             59.18|
6679300|51.616215000000004|8.915604778943901E-6|
+----------+------------------+---------+---------+-----------------
+--------+------------------+--------------------+
only showing top 5 rows
```

```
#9. Find the Day With Peak High Price
df.orderBy(col("High").desc()).select("Date").show(1)
```

```
+----------+
|      Date|
+----------+
|2015-01-13|
+----------+
only showing top 1 row
```

```
#10. Calculate the Mean of the Close Column
df.selectExpr("avg(Close)").show()
```

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

```
#11. Find Max and Min of Volume Column
df.selectExpr("max(Volume)", "min(Volume)","avg(Volume)").show()
```

```
+-----------+-----------+-----------------+
|max(Volume)|min(Volume)|      avg(Volume)|
+-----------+-----------+-----------------+
|   80898100|    2094900|8222093.481717011|
+-----------+-----------+-----------------+
```

```
#12. Count Days Where Close Was Lower Than 60 Dollars
df.filter(col("Close") < 60).count()
```

```
81
```

```
#13. Percentage of Days Where High Was Greater Than 80 Dollars
(df.filter(col("High") > 80).count() / df.count()) * 100
```

```
9.141494435612083
```

```
#14. Compute Pearson Correlation Between High and Volume
df.selectExpr("corr(High, Volume)").show()
```

```
+------------------+
| corr(High, Volume)|
+------------------+
|-0.3384326061737161|
+------------------+
```

```
#15. Find the Max High Per Year
from pyspark.sql.functions import year
df.withColumn("Year", year(col("Date"))).groupBy("Year").agg({"High":
"max"}).orderBy("Year").show()
```

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

```
#16. Compute Average Close Per Calendar Month
from pyspark.sql.functions import month
df.withColumn("Month",
month(col("Date"))).groupBy("Month").agg({"Close":
"avg"}).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|
+-----+-----------------+
```