Basic Operations

This lecture will cover some basic operations with Spark DataFrames.

We will play around with some stock data from Apple.

In [1]:
```
from pyspark.sql import SparkSession
```

In [2]:
```
# May take awhile locally
spark = SparkSession.builder.appName("Operations").getOrCreate()
```

In [26]:
```
# Let Spark know about the header and infer the Schema types!
df = spark.read.csv('appl_stock.csv',inferSchema=True,header=True)
```

In [28]:
```
df.printSchema()
```

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

Filtering Data

A large part of working with DataFrames is the ability to quickly filter out data based on conditions. Spark DataFrames are built on top of the Spark SQL platform, which means that is you already know SQL, you can quickly and easily grab that data using SQL commands, or using the DataFram methods (which is what we focus on in this course).

In [31]:
```
# Using SQL
df.filter("Close<500").show()
```

```
+--------------------+------------------+------------------+------------------+------------------+--------
-+------------------+
|                Date|              Open|              High|               Low|             Close|   Volum
e|         Adj Close|
+--------------------+------------------+------------------+------------------+------------------+--------
-+------------------+
|2010-01-04 00:00:...|        213.429998|        214.499996|212.38000099999996|        214.009998|12343240
0|         27.727039|
|2010-01-05 00:00:...|        214.599998|        215.589994|        213.249994|        214.379993|15047620
0|27.774976000000002|
|2010-01-06 00:00:...|        214.379993|            215.23|        210.750004|        210.969995|13804000
0|27.333178000000004|
|2010-01-07 00:00:...|            211.75|        212.000006|        209.050005|            210.58|11928280
0|          27.28265|
|2010-01-08 00:00:...|        210.299994|        212.000006|209.06000500000002|211.98000499999998|11190270
0|         27.464034|
|2010-01-11 00:00:...|212.79999700000002|        213.000002|        208.450005|210.11000299999998|11555740
0|         27.221758|
|2010-01-12 00:00:...|209.18999499999998|209.76999500000002|        206.419998|        207.720001|14861490
0|          26.91211|
|2010-01-13 00:00:...|        207.870005|210.92999500000002|        204.099998|        210.650002|15147300
0|          27.29172|
|2010-01-14 00:00:...|210.11000299999998|210.45999700000002|        209.020004|            209.43|10822350
0|         27.133657|
|2010-01-15 00:00:...|210.92999500000002|211.59999700000003|        205.869999|            205.93|14851690
0|26.680197999999997|
|2010-01-19 00:00:...|        208.330002|215.18999900000003|        207.240004|        215.039995|18250190
0|27.860484999999997|
|2010-01-20 00:00:...|        214.910006|        215.549994|        209.500002|            211.73|15303820
0|         27.431644|
|2010-01-21 00:00:...|        212.079994|213.30999599999998|        207.210003|        208.069996|15203860
0|         26.957455|
|2010-01-22 00:00:...|206.78000600000001|        207.499996|            197.16|            197.75|22044190
0|         25.620401|
|2010-01-25 00:00:...|202.51000200000001|        204.699999|        200.190002|        203.070002|26642490
0|26.309658000000002|
|2010-01-26 00:00:...|205.95000100000001|        213.710005|        202.580004|        205.940001|46677750
0|         26.681494|
|2010-01-27 00:00:...|        206.849995|            210.58|        199.530001|        207.880005|43064210
0|26.932840000000002|
|2010-01-28 00:00:...|        204.930004|        205.500004|        198.699995|        199.289995|29337560
0|25.819922000000002|
|2010-01-29 00:00:...|        201.079996|        202.199995|        190.250002|        192.060003|31148810
0|         24.883208|
|2010-02-01 00:00:...|192.36999699999998|             196.0|191.29999899999999|        194.729998|18746910
0|         25.229131|
+--------------------+------------------+------------------+------------------+------------------+--------
-+------------------+
only showing top 20 rows
```

In [35]:
```
# Using SQL with .select()
df.filter("Close<500").select('Open').show()
```

```
+------------------+
|              Open|
+------------------+
|        213.429998|
|        214.599998|
|        214.379993|
|            211.75|
|        210.299994|
|212.79999700000002|
|209.18999499999998|
|        207.870005|
|210.11000299999998|
|210.92999500000002|
|        208.330002|
|        214.910006|
|        212.079994|
|206.78000600000001|
|202.51000200000001|
|205.95000100000001|
|        206.849995|
|        204.930004|
|        201.079996|
|192.36999699999998|
+------------------+
only showing top 20 rows
```

In [36]:
```
# Using SQL with .select()
df.filter("Close<500").select(['Open','Close']).show()
```

```
+------------------+------------------+
|              Open|             Close|
+------------------+------------------+
|        213.429998|        214.009998|
|        214.599998|        214.379993|
|        214.379993|        210.969995|
|            211.75|            210.58|
|        210.299994|211.98000499999998|
|212.79999700000002|210.11000299999998|
|209.18999499999998|        207.720001|
|        207.870005|        210.650002|
|210.11000299999998|            209.43|
|210.92999500000002|            205.93|
|        208.330002|        215.039995|
|        214.910006|            211.73|
|        212.079994|        208.069996|
|206.78000600000001|            197.75|
|202.51000200000001|        203.070002|
|205.95000100000001|        205.940001|
|        206.849995|        207.880005|
|        204.930004|        199.289995|
|        201.079996|        192.060003|
|192.36999699999998|        194.729998|
+------------------+------------------+
only showing top 20 rows
```

Using normal python comparison operators is another way to do this, they will look very similar to SQL operators, except you need to make sure you are calling the entire column within the dataframe, using the format: df["column name"]

Let's see some examples:

In [38]:
```
df.filter(df["Close"] < 200).show()
```

In [38]:
```
df.filter(df['Close'] < 200).show()
```

```
+-------------------+------------------+------------------+------------------+------------------+--------
-+------------------+
|               Date|              Open|              High|               Low|             Close|   Volum
e|         Adj Close|
+-------------------+------------------+------------------+------------------+------------------+--------
-+------------------+
|2010-01-22 00:00:...|206.78000600000001|        207.499996|            197.16|            197.75|22044190
0|         25.620401|
|2010-01-28 00:00:...|        204.930004|        205.500004|        198.699995|        199.289995|29337560
0|25.819922000000002|
|2010-01-29 00:00:...|        201.079996|        202.199995|        190.250002|        192.060003|31148810
0|         24.883208|
|2010-02-01 00:00:...|192.36999699999998|             196.0|191.29999899999999|        194.729998|18746910
0|         25.229131|
|2010-02-02 00:00:...|        195.909998|        196.319994|193.37999299999998|        195.859997|17458560
0|25.375532999999997|
|2010-02-03 00:00:...|        195.169994|        200.200003|        194.420004|        199.229994|15383200
0|25.812148999999998|
|2010-02-04 00:00:...|        196.730003|        198.370001|        191.570005|        192.050003|18941300
0|         24.881912|
|2010-02-05 00:00:...|192.63000300000002|             196.0|        190.850002|        195.460001|21257670
0|25.323710000000002|
|2010-02-08 00:00:...|        195.690006|197.88000300000002|        193.999994|194.11999699999998|11956770
0|           25.1501|
|2010-02-09 00:00:...|        196.419996|        197.499994|        194.749998|196.19000400000002|15822170
0|         25.418289|
|2010-02-10 00:00:...|        195.889997|             196.6|            194.26|195.12000700000002| 9259040
0|          25.27966|
|2010-02-11 00:00:...|        194.880001|        199.750006|194.05999599999998|        198.669994|13758640
0|         25.739595|
|2010-02-23 00:00:...|        199.999998|        201.330002|        195.709993|        197.059998|14377370
0|         25.531005|
|2014-06-09 00:00:...|         92.699997|         93.879997|             91.75|         93.699997| 7541500
0|         88.906324|
|2014-06-10 00:00:...|         94.730003|         95.050003|             93.57|             94.25| 6277700
0|         89.428189|
|2014-06-11 00:00:...|         94.129997|         94.760002|         93.470001|         93.860001| 4568100
0|         89.058142|
|2014-06-12 00:00:...|         94.040001|         94.120003|         91.900002|         92.290001| 5474900
0|         87.568463|
|2014-06-13 00:00:...|         92.199997|         92.440002|         90.879997|         91.279999| 5452500
0|         86.610132|
|2014-06-16 00:00:...|         91.510002|             92.75|         91.449997|         92.199997| 3556100
0|         87.483064|
|2014-06-17 00:00:...|         92.309998|         92.699997|         91.800003| 92.08000200000001| 2972600
0| 87.36920699999999|
+-------------------+------------------+------------------+------------------+------------------+--------
-+------------------+
only showing top 20 rows
```

In [39]:
```
# Will produce an error, make sure to read the error!
df.filter(df["Close"] < 200 and df['Open'] > 200).show()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-39-df4ea1e41a0f> in <module>()
----> 1 df.filter(df["Close"] < 200 and df['Open'] > 200).show()

/usr/local/Cellar/apache-spark/2.0.1/libexec/python/pyspark/sql/column.py in __nonzero__(self)
    425 
    426     def __nonzero__(self):
--> 427         raise ValueError("Cannot convert column into bool: please use '&' for 'and', '|' for 'or',
 "
    428                          "'~' for 'not' when building DataFrame boolean expressions.")
    429     __bool__ = __nonzero__

ValueError: Cannot convert column into bool: please use '&' for 'and', '|' for 'or', '~' for 'not' when bu
ilding DataFrame boolean expressions.
```

In [47]:
```
# Make sure to add in the parenthesis separating the statements!
df.filter( (df["Close"] < 200) & (df['Open'] > 200) ).show()
```

```
+-------------------+------------------+----------+----------+----------+---------+------------------+
|               Date|              Open|      High|       Low|     Close|   Volume|         Adj Close|
+-------------------+------------------+----------+----------+----------+---------+------------------+
|2010-01-22 00:00:...|206.78000600000001|207.499996|    197.16|    197.75|220441900|         25.620401|
|2010-01-28 00:00:...|        204.930004|205.500004|198.699995|199.289995|293375600|25.819922000000002|
|2010-01-29 00:00:...|        201.079996|202.199995|190.250002|192.060003|311488100|         24.883208|
|2010-01-29 00:00:...|        201.079996|202.199995|190.250002|192.060003|311488100|         24.883208|
+--------------------+------------------+----------+----------+----------+---------+------------------+
```

In [49]:
```
# Make sure to add in the parenthesis separating the statements!
df.filter( (df["Close"] < 200) | (df['Open'] > 200) ).show()
```

```
+--------------------+------------------+------------------+------------------+------------------+--------
-+------------------+
|                Date|              Open|              High|               Low|             Close|   Volum
e|         Adj Close|
+--------------------+------------------+------------------+------------------+------------------+--------
-+------------------+
|2010-01-04 00:00:...|        213.429998|        214.499996|212.38000099999996|        214.009998|12343240
0|         27.727039|
|2010-01-05 00:00:...|        214.599998|        215.589994|        213.249994|        214.379993|15047620
0|27.774976000000002|
|2010-01-06 00:00:...|        214.379993|            215.23|        210.750004|        210.969995|13804000
0|27.333178000000004|
|2010-01-07 00:00:...|            211.75|        212.000006|        209.050005|            210.58|11928280
0|          27.28265|
|2010-01-08 00:00:...|        210.299994|        212.000006|209.06000500000002|211.98000499999998|11190270
0|         27.464034|
|2010-01-11 00:00:...|212.79999700000002|        213.000002|        208.450005|210.11000299999998|11555740
0|         27.221758|
|2010-01-12 00:00:...|209.18999499999998|209.76999500000002|        206.419998|        207.720001|14861490
0|          26.91211|
|2010-01-13 00:00:...|        207.870005|210.92999500000002|        204.099998|        210.650002|15147300
0|          27.29172|
|2010-01-14 00:00:...|210.11000299999998|210.45999700000002|        209.020004|            209.43|10822350
0|         27.133657|
|2010-01-15 00:00:...|210.92999500000002|211.59999700000003|        205.869999|            205.93|14851690
0|26.680197999999997|
|2010-01-19 00:00:...|        208.330002|215.18999900000003|        207.240004|        215.039995|18250190
0|27.860484999999997|
|2010-01-20 00:00:...|        214.910006|        215.549994|        209.500002|            211.73|15303820
0|         27.431644|
|2010-01-21 00:00:...|        212.079994|213.30999599999998|        207.210003|        208.069996|15203860
0|         26.957455|
|2010-01-22 00:00:...|206.78000600000001|        207.499996|            197.16|            197.75|22044190
0|         25.620401|
|2010-01-25 00:00:...|202.51000200000001|        204.699999|        200.190002|        203.070002|26642490
0|26.309658000000002|
|2010-01-26 00:00:...|205.95000100000001|        213.710005|        202.580004|        205.940001|46677750
0|         26.681494|
|2010-01-27 00:00:...|        206.849995|            210.58|        199.530001|        207.880005|43064210
0|26.932840000000002|
|2010-01-28 00:00:...|        204.930004|        205.500004|        198.699995|        199.289995|29337560
0|25.819922000000002|
|2010-01-29 00:00:...|        201.079996|        202.199995|        190.250002|        192.060003|31148810
0|         24.883208|
|2010-02-01 00:00:...|192.36999699999998|             196.0|191.29999899999999|        194.729998|18746910
0|         25.229131|
+--------------------+------------------+------------------+------------------+------------------+--------
-+------------------+
only showing top 20 rows
```

In [51]:
```
# Make sure to add in the parenthesis separating the statements!
df.filter( (df["Close"] < 200) & ~(df['Open'] < 200) ).show()
```

```
+--------------------+------------------+----------+----------+----------+---------+------------------+
|                Date|              Open|      High|       Low|     Close|   Volume|         Adj Close|
+--------------------+------------------+----------+----------+----------+---------+------------------+
|2010-01-22 00:00:...|206.78000600000001|207.499996|    197.16|    197.75|220441900|         25.620401|
|2010-01-28 00:00:...|        204.930004|205.500004|198.699995|199.289995|293375600|25.819922000000002|
|2010-01-29 00:00:...|        201.079996|202.199995|190.250002|192.060003|311488100|         24.883208|
+--------------------+------------------+----------+----------+----------+---------+------------------+
```

In [46]:
```
df.filter(df["Low"] == 197.16).show()
```

```
+--------------------+------------------+----------+------+------+---------+---------+
|                Date|              Open|      High|   Low| Close|   Volume|Adj Close|
+--------------------+------------------+----------+------+------+---------+---------+
|2010-01-22 00:00:...|206.78000600000001|207.499996|197.16|197.75|220441900|25.620401|
+--------------------+------------------+----------+------+------+---------+---------+
```

In [52]:
```
# Collecting results as Python objects
df.filter(df["Low"] == 197.16).collect()
df.filter(df["Low"] == 197.16).collect()
```

Out[52]:
```
[Row(Date=datetime.datetime(2010, 1, 22, 0, 0), Open=206.78000600000001, High=207.499996, Low=197.16, Clos
e=197.75, Volume=220441900, Adj Close=25.620401)]
```

In [53]:
```
result = df.filter(df["Low"] == 197.16).collect()
```

In [62]:
```
# Note the nested structure returns a nested row object
type(result[0])
```

Out[62]:
```
pyspark.sql.types.Row
```

In [65]:
```
row = result[0]
```

Rows can be called to turn into dictionaries

In [64]:
```
row.asDict()
```

Out[64]:
```
{'Adj Close': 25.620401,
```