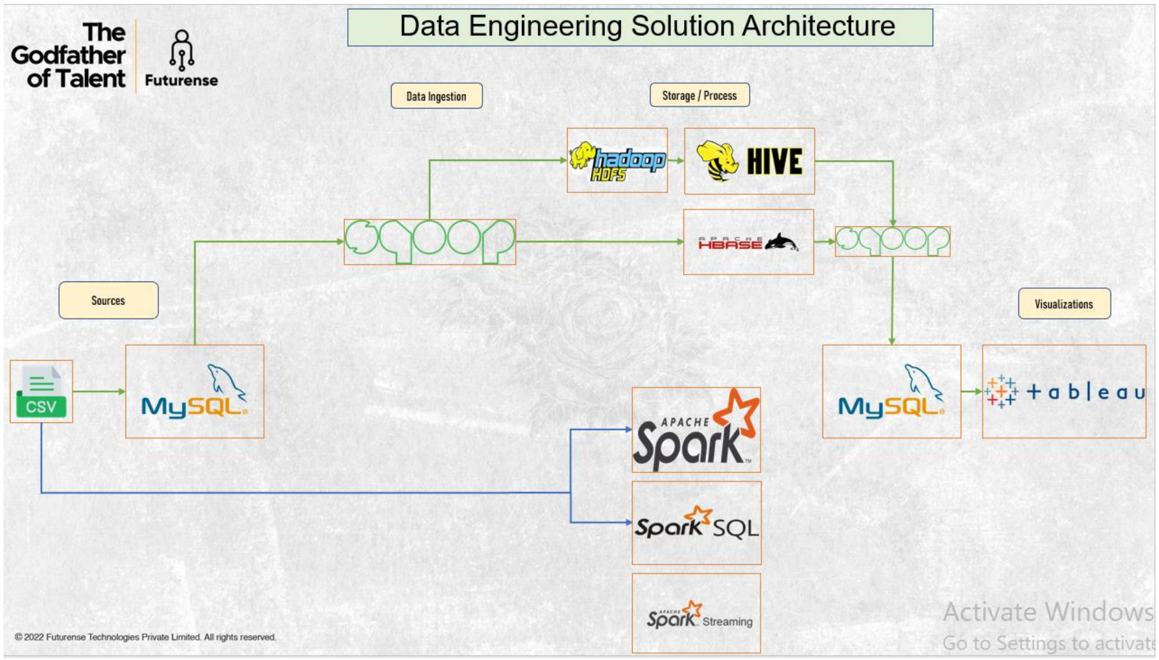


Walmart Stock Analysis

HIVE | PYSPARK | SPARKSQL | MYSQL | TABLEAU

Shubham Ghate | Futurense Technologies | Jul -2023

HIVE - Analysis

MYSQL

Create Database , Tables & load data-

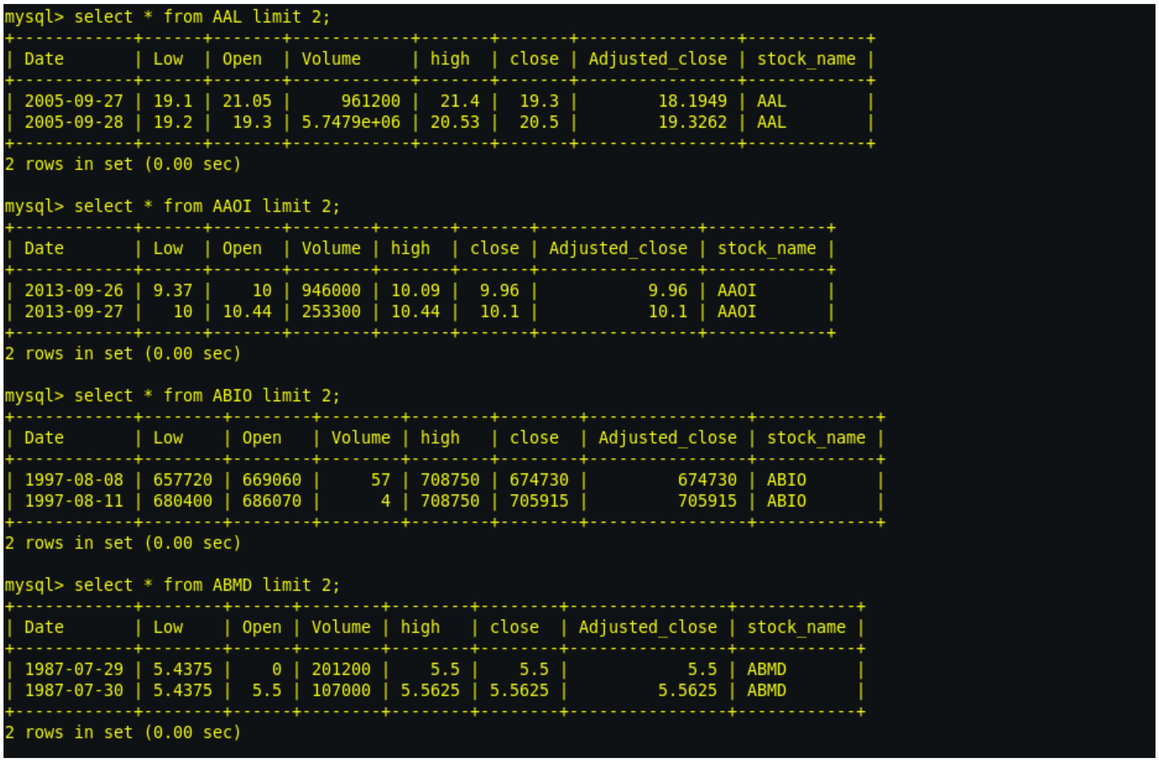
- mysql> create database Project;
- mysql> create table AAL(Date date,Low float,Open float,Volume float,high float,close float,Adjusted_close float);
- mysql> LOAD DATA INFILE '/home/cloudera/project/AAL.csv' INTO TABLE AAL FIELDS TERMINATED BY ',' (@var1,@var2,@var3,@var4,@var5,@var6,@var7) SET Date=STR_TO_DATE(@var1,'%d-%m-%Y'),Low=@var2,Open=@var3,Volume=@var4,high=@var5,Close=@var6,Adjusted_close=@var7;

Add Stock_Name column in each file -

- mysql> alter table AAL add stock_name varchar(20) default 'AAL';

Stock_Data - Merge all four stock files-

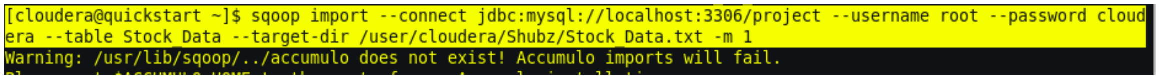
- mysql> create table Stock_Data(Date date,Low float,Open float,Volume float,high float,close float,Adjusted_close float,Stock_Name varchar(20));
- mysql> insert into Stock_Data select * from AAL union all select * from AAOI union all select * from ABIO union all select * from ABMD;



```
mysql> select * from AAL limit 2;
+------------+------+-------+------------+-------+-------+----------------+------------+
| Date       | Low  | Open  | Volume     | high  | close | Adjusted_close | stock_name |
+------------+------+-------+------------+-------+-------+----------------+------------+
| 2005-09-27 | 19.1 | 21.05 |     961200 |  21.4 |  19.3 |        18.1949 | AAL        |
| 2005-09-28 | 19.2 |  19.3 | 5.7479e+06 | 20.53 |  20.5 |        19.3262 | AAL        |
+------------+------+-------+------------+-------+-------+----------------+------------+
2 rows in set (0.00 sec)

mysql> select * from AAOI limit 2;
+------------+------+-------+--------+-------+-------+----------------+------------+
| Date       | Low  | Open  | Volume | high  | close | Adjusted_close | stock_name |
+------------+------+-------+--------+-------+-------+----------------+------------+
| 2013-09-26 | 9.37 |    10 | 946000 | 10.09 |  9.96 |           9.96 | AAOI       |
| 2013-09-27 |   10 | 10.44 | 253300 | 10.44 |  10.1 |           10.1 | AAOI       |
+------------+------+-------+--------+-------+-------+----------------+------------+
2 rows in set (0.00 sec)

mysql> select * from ABIO limit 2;
+------------+--------+--------+--------+--------+--------+----------------+------------+
| Date       | Low    | Open   | Volume | high   | close  | Adjusted_close | stock_name |
+------------+--------+--------+--------+--------+--------+----------------+------------+
| 1997-08-08 | 657720 | 669060 |     57 | 708750 | 674730 |         674730 | ABIO       |
| 1997-08-11 | 680400 | 686070 |      4 | 708750 | 705915 |         705915 | ABIO       |
+------------+--------+--------+--------+--------+--------+----------------+------------+
2 rows in set (0.00 sec)

mysql> select * from ABMD limit 2;
+------------+--------+------+--------+--------+--------+----------------+------------+
| Date       | Low    | Open | Volume | high   | close  | Adjusted_close | stock_name |
+------------+--------+------+--------+--------+--------+----------------+------------+
| 1987-07-29 | 5.4375 |    0 | 201200 |    5.5 |    5.5 |            5.5 | ABMD       |
| 1987-07-30 | 5.4375 |  5.5 | 107000 | 5.5625 | 5.5625 |         5.5625 | ABMD       |
+------------+--------+------+--------+--------+--------+----------------+------------+
2 rows in set (0.00 sec)
```

SQOOP Pipeline for MYSQL to HDFS -

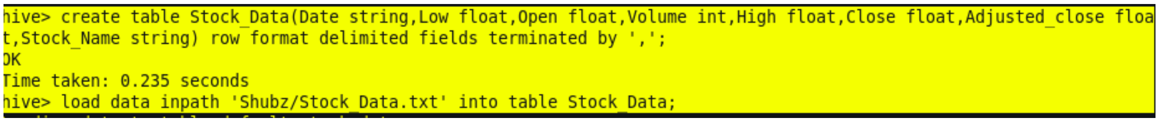
[cloudera@quickstart ~]$ sqoop import --connect jdbc:mysql://localhost:3306/project --username root --password cloudera --table Stock_Data --target-dir /user/cloudera/Shubz/Stock_Data.txt -m 1



```
[cloudera@quickstart ~]$ sqoop import --connect jdbc:mysql://localhost:3306/project --username root --password cloud
era --table Stock_Data --target-dir /user/cloudera/Shubz/Stock_Data.txt -m 1
Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.
```

Creating Schema & Loading data into HIVE –

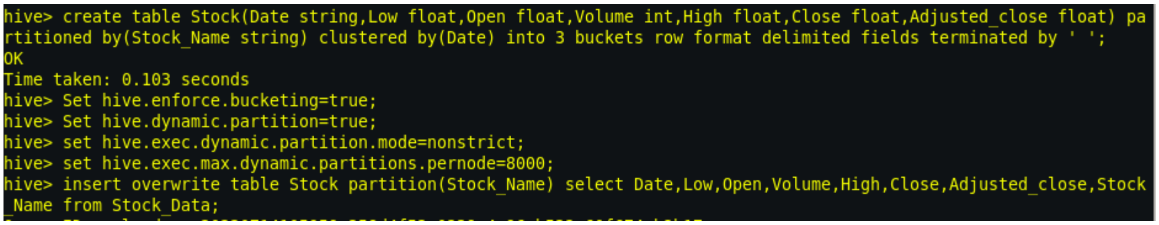
- ✓ hive> create table Stock_Data(Date string,Low float,Open float,Volume int,High float,Close float,Adjusted_close float,Stock_Name string) row format delimited fields terminated by ',';
- ✓ hive> load data inpath 'Shubz/Stock_Data.txt' into table Stock_Data;



```
hive> create table Stock_Data(Date string,Low float,Open float,Volume int,High float,Close float,Adjusted_close float,Stock_Name string) row format delimited fields terminated by ',';
OK
Time taken: 0.235 seconds
hive> load data inpath 'Shubz/Stock_Data.txt' into table Stock_Data;
```

Implementing Partitioning and Bucketing in Hive: Optimizing Data Storage and Query Performance

- ✓ hive> create table Stock(Date string,Low float,Open float,Volume int,High float,Close float,Adjusted_close float) partitioned by(Stock_Name string) clustered by(Date) into 3 buckets row format delimited fields terminated by ' ';
- ✓ hive> Set hive.enforce.bucketing=true;
- ✓ hive> Set hive.dynamic.partition=true;
- ✓ hive> set hive.exec.dynamic.partition.mode=nonstrict;
- ✓ hive> set hive.exec.max.dynamic.partitions.pernode=8000;
- ✓ hive> insert overwrite table Stock partition(Stock_Name) select Date,Low,Open,Volume,High,Close,Adjusted_close,Stock_Name from Stock_Data;



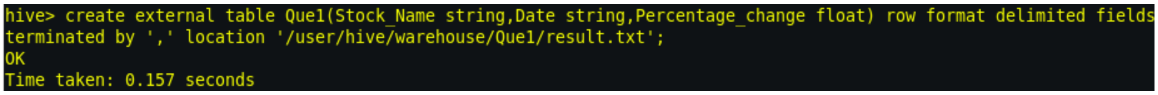
```
hive> create table Stock(Date string,Low float,Open float,Volume int,High float,Close float,Adjusted_close float) partitioned by(Stock_Name string) clustered by(Date) into 3 buckets row format delimited fields terminated by ' ';
OK
Time taken: 0.103 seconds
hive> Set hive.enforce.bucketing=true;
hive> Set hive.dynamic.partition=true;
hive> set hive.exec.dynamic.partition.mode=nonstrict;
hive> set hive.exec.max.dynamic.partitions.pernode=8000;
hive> insert overwrite table Stock partition(Stock_Name) select Date,Low,Open,Volume,High,Close,Adjusted_close,Stock_Name from Stock_Data;
```

Problem Statements : -

1. Write a Hive query to identify the top three dates that experienced the largest percentage change in stock price (from open to close) for every stock.

------> EXTERNAL TABLE CREATION:

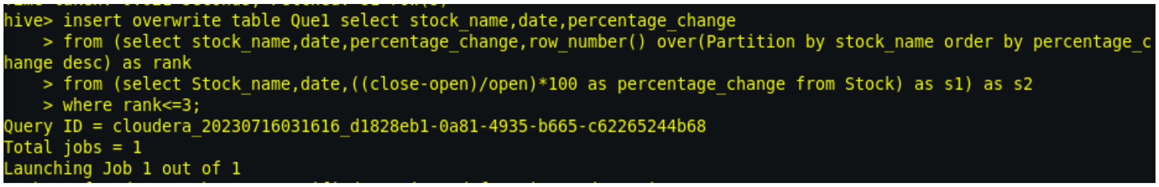
Hive> create external table Que1(Stock_Name string,Date string,Percentage_change float) row format delimited fields terminated by ',' location '/user/hive/warehouse/Que1/result.txt';



```
hive> create external table Que1(Stock_Name string,Date string,Percentage_change float) row format delimited fields terminated by ',' location '/user/hive/warehouse/Que1/result.txt';
OK
Time taken: 0.157 seconds
```

------>TRANSFERRING OUTPUT DATA TO EXTERNAL TABLE

hive> insert overwrite table Que1
SELECT stock_name,date,percentage_change
FROM (select stock_name,date,percentage_change,row_number() over(Partition by stock_name order by percentage_change desc) as rank
FROM (select Stock_name,date,((close-open)/open)*100 as percentage_change from Stock) as s1) as s2 where rank<=3;



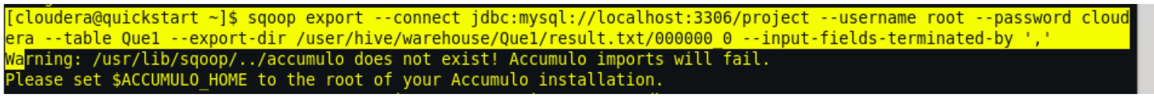
```
hive> insert overwrite table Que1 select stock_name,date,percentage_change
    > from (select stock_name,date,percentage_change,row_number() over(Partition by stock_name order by percentage_c
hange desc) as rank
    > from (select Stock_name,date,((close-open)/open)*100 as percentage_change from Stock) as s1) as s2
    > where rank<=3;
Query ID = cloudera_20230716031616_d1828eb1-0a81-4935-b665-c62265244b68
Total jobs = 1
Launching Job 1 out of 1
```

------>MYSQL TABLE CREATED (CLIENT DATABASE)

mysql> create table Que1(Stock_name varchar(20),Stock_date date,Percentage_change float);

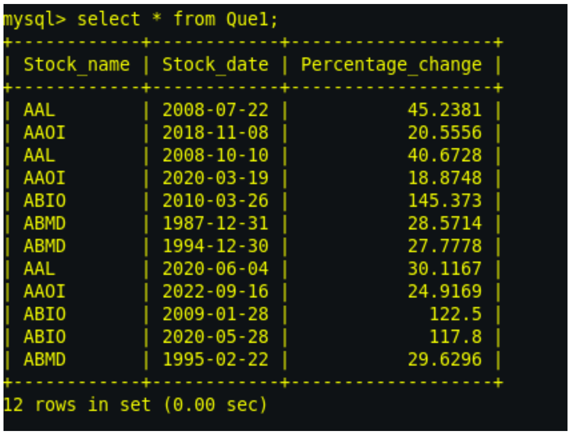
------>SQOOP COMMAND TO TRANSFER O/P TABLE TO MYSQL

[cloudera@quickstart ~]$ sqoop export --connect jdbc:mysql://localhost:3306/project --username root --password cloudera --table Que1 --export-dir /user/hive/warehouse/Que1/result.txt/000000_0 --input-fields-terminated-by ','



```
[cloudera@quickstart ~]$ sqoop export --connect jdbc:mysql://localhost:3306/project --username root --password cloud
era --table Que1 --export-dir /user/hive/warehouse/Que1/result.txt/000000_0 --input-fields-terminated-by ','
Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
```

------>FINAL OUTPUT DATA -



```
mysql> select * from Que1;
+------------+------------+-------------------+
| Stock_name | Stock_date | Percentage_change |
+------------+------------+-------------------+
| AAL        | 2008-07-22 |           45.2381 |
| AAOI       | 2018-11-08 |           20.5556 |
| AAL        | 2008-10-10 |           40.6728 |
| AAOI       | 2020-03-19 |           18.8748 |
| ABIO       | 2010-03-26 |           145.373 |
| ABMD       | 1987-12-31 |           28.5714 |
| ABMD       | 1994-12-30 |           27.7778 |
| AAL        | 2020-06-04 |           30.1167 |
| AAOI       | 2022-09-16 |           24.9169 |
| ABIO       | 2009-01-28 |             122.5 |
| ABIO       | 2020-05-28 |             117.8 |
| ABMD       | 1995-02-22 |           29.6296 |
+------------+------------+-------------------+
12 rows in set (0.00 sec)
```

2. write a Hive query to identify the dates where Low is less than average month low for every stock.

------> EXTERNAL TABLE CREATION:

```
hive> create external table Que2(Stock_Name string,Date string) row format delimited
fields terminated by ',' location '/user/hive/warehouse/Que2/result.txt';
```

------>TRANSFERRING OUTPUT DATA TO EXTERNAL TABLE

```
hive> insert overwrite table Que2
select stock_name,date from (select stock_name,date, low,avg(low) over(partition by
stock_name,year(date),month(date)) as avg_monthly_low from stock) as s1 where
low<avg_monthly_low;
```

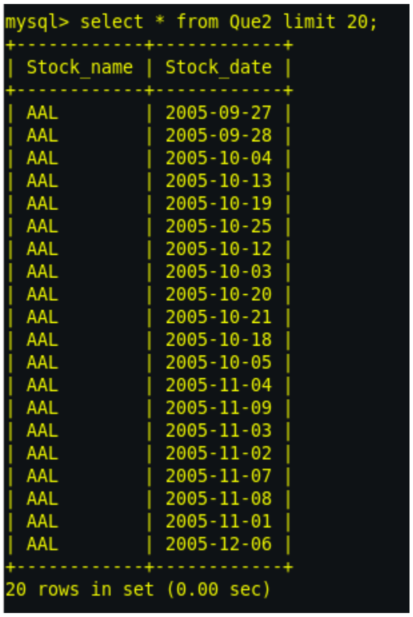
------>MYSQL TABLE CREATED (CLIENT DATABASE)

```
mysql> create table Que2(Stock_name varchar(20),Stock_date date);
```

------>SQOOP COMMAND TO TRANSFER O/P TABLE TO MYSQL

```
[cloudera@quickstart ~]$ sqoop export --connect jdbc:mysql://localhost:3306/project --
username root --password cloudera --table Que2 --export-dir
/user/hive/warehouse/Que2/result.txt/000000_0 --input-fields-terminated-by ','
```

------>FINAL OUTPUT DATA –



```
mysql> select * from Que2 limit 20;
+------------+------------+
| Stock_name | Stock_date |
+------------+------------+
| AAL        | 2005-09-27 |
| AAL        | 2005-09-28 |
| AAL        | 2005-10-04 |
| AAL        | 2005-10-13 |
| AAL        | 2005-10-19 |
| AAL        | 2005-10-25 |
| AAL        | 2005-10-12 |
| AAL        | 2005-10-03 |
| AAL        | 2005-10-20 |
| AAL        | 2005-10-21 |
| AAL        | 2005-10-18 |
| AAL        | 2005-10-05 |
| AAL        | 2005-11-04 |
| AAL        | 2005-11-09 |
| AAL        | 2005-11-03 |
| AAL        | 2005-11-02 |
| AAL        | 2005-11-07 |
| AAL        | 2005-11-08 |
| AAL        | 2005-11-01 |
| AAL        | 2005-12-06 |
+------------+------------+
20 rows in set (0.00 sec)
```

3. Write a Hive query to find the date with the longest consecutive streak of increasing closing prices for every stock.

------> EXTERNAL TABLE CREATION:

hive> create external table Que3(Stock_Name string,streak_date string) row format delimited fields terminated by ',' location '/user/hive/warehouse/Que3/result.txt';

------>TRANSFERRING OUTPUT DATA TO EXTERNAL TABLE

hive> insert overwrite table Que3
SELECT stock_name, streak_date FROM (SELECT stock_name, streak_date,ROW_NUMBER() OVER (PARTITION BY stock_name ORDER BY streak_length DESC) AS rn
FROM (SELECT stock_name, date, close,SUM(is_increasing) OVER (PARTITION BY stock_name, grp ORDER BY date) AS streak_length,FIRST_VALUE(date) OVER (PARTITION BY stock_name, grp ORDER BY date) AS streak_date

FROM (SELECT stock_name, date, close,CASE WHEN LAG(close, 1) OVER (PARTITION BY stock_name ORDER BY date) < close THEN 1 ELSE 0 END AS is_increasing,DATE_SUB(date, ROW_NUMBER() OVER (PARTITION BY stock_name ORDER BY date)) AS grp
FROM Stock) as sub) as sub2) as sub3
WHERE rn = 1;

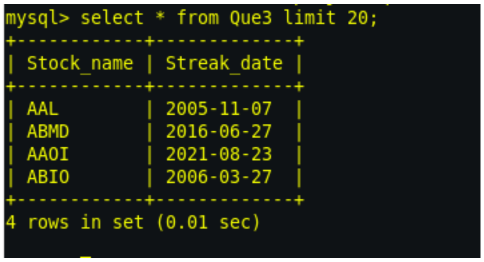
------>MYSQL TABLE CREATED (CLIENT DATABASE)

```
mysql> create table Que3(Stock_name varchar(20),Streak_date date);
```

------>SQOOP COMMAND TO TRANSFER O/P TABLE TO MYSQL

```
[cloudera@quickstart ~]$ sqoop export --connect jdbc:mysql://localhost:3306/project --
username root --password cloudera --table Que3 --export-dir
/user/hive/warehouse/Que3/result.txt/000000_0 --input-fields-terminated-by ','
```

------>FINAL OUTPUT DATA –



```
mysql> select * from Que3 limit 20;
+------------+-------------+
| Stock_name | Streak_date |
+------------+-------------+
| AAL        | 2005-11-07  |
| ABMD       | 2016-06-27  |
| AAOI       | 2021-08-23  |
| ABIO       | 2006-03-27  |
+------------+-------------+
4 rows in set (0.01 sec)
```

4. write a Hive query to find the dates where AAL open price is higher than AAOI open price OR AAL volume greater than AMBD (write your query in an optimised way)

------> EXTERNAL TABLE CREATION:

```
hive> create external table Que4(Date string) row format delimited fields terminated by ','
location '/user/hive/warehouse/Que4/result.txt';
```

------>TRANSFERRING OUTPUT DATA TO EXTERNAL TABLE

```
hive> insert overwrite table Que4
SELECT s1.date FROM Stock s1 JOIN Stock s2 ON s1.date = s2.date

JOIN Stock s3 ON s1.date = s3.date
WHERE (s1.stock_name = 'AAL' AND s1.open > s2.open AND s2.stock_name = 'AAOI')
OR (s1.stock_name = 'AAL' AND s1.volume > s3.volume AND s3.stock_name = 'AMBD');
```

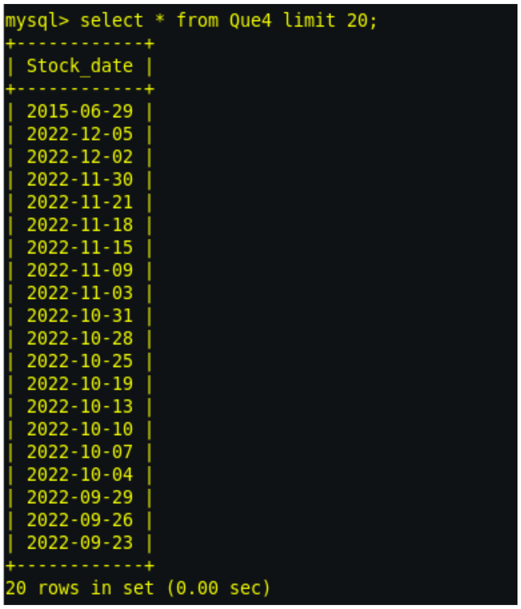
------>MYSQL TABLE CREATED (CLIENT DATABASE)

```
mysql> create table Que4(Stock_date date);
```

------>SQOOP COMMAND TO TRANSFER O/P TABLE TO MYSQL

```
[cloudera@quickstart ~]$ sqoop export --connect jdbc:mysql://localhost:3306/project --
username root --password cloudera --table Que4 --export-dir
/user/hive/warehouse/Que4/result.txt/000000_0 --input-fields-terminated-by ','
```

------>FINAL OUTPUT DATA –



```
mysql> select * from Que4 limit 20;
+------------+
| Stock_date |
+------------+
| 2015-06-29 |
| 2022-12-05 |
| 2022-12-02 |
| 2022-11-30 |
| 2022-11-21 |
| 2022-11-18 |
| 2022-11-15 |
| 2022-11-09 |
| 2022-11-03 |
| 2022-10-31 |
| 2022-10-28 |
| 2022-10-25 |
| 2022-10-19 |
| 2022-10-13 |
| 2022-10-10 |
| 2022-10-07 |
| 2022-10-04 |
| 2022-09-29 |
| 2022-09-26 |
| 2022-09-23 |
+------------+
20 rows in set (0.00 sec)
```

5. write a Hive query to calculate VH ratio(volume to hive ratio).

------> EXTERNAL TABLE CREATION:

hive> create external table Que5(Stock_Name string,Volume int,High float,VH_Ratio float) row format delimited fields terminated by ',' location '/user/hive/warehouse/Que5/result.txt';

------>TRANSFERRING OUTPUT DATA TO EXTERNAL TABLE:

hive> insert overwrite table Que5 SELECT Stock_Name,Volume,High, volume / adjusted_close AS vh_ratio FROM Stock;

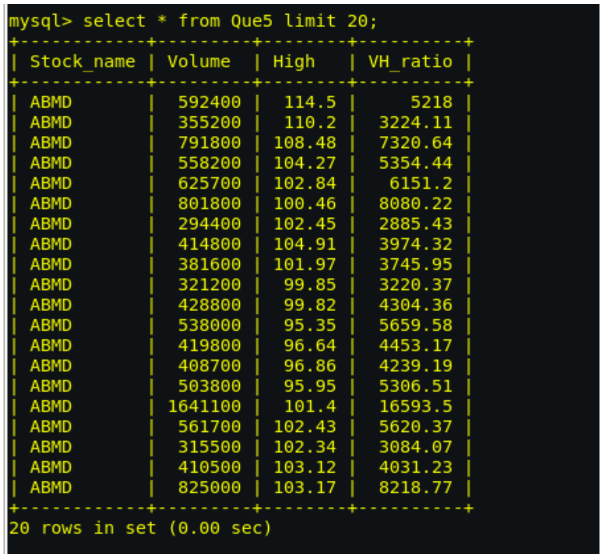
------>MYSQL TABLE CREATED (CLIENT DATABASE)

mysql> create table Que5(Stock_name varchar(20),Volume int,High float,VH_ratio float);

------>SQOOP COMMAND TO TRANSFER O/P TABLE TO MYSQL

[cloudera@quickstart ~]$ sqoop export --connect jdbc:mysql://localhost:3306/project --username root --password cloudera --table Que5 --export-dir /user/hive/warehouse/Que5/result.txt/000000_0 --input-fields-terminated-by ','

------>FINAL OUTPUT DATA –



```
mysql> select * from Que5 limit 20;
+------------+---------+--------+----------+
| Stock_name | Volume  | High   | VH_ratio |
+------------+---------+--------+----------+
| ABMD       |  592400 |  114.5 |     5218 |
| ABMD       |  355200 |  110.2 |  3224.11 |
| ABMD       |  791800 | 108.48 |  7320.64 |
| ABMD       |  558200 | 104.27 |  5354.44 |
| ABMD       |  625700 | 102.84 |   6151.2 |
| ABMD       |  801800 | 100.46 |  8080.22 |
| ABMD       |  294400 | 102.45 |  2885.43 |
| ABMD       |  414800 | 104.91 |  3974.32 |
| ABMD       |  381600 | 101.97 |  3745.95 |
| ABMD       |  321200 |  99.85 |  3220.37 |
| ABMD       |  428800 |  99.82 |  4304.36 |
| ABMD       |  538000 |  95.35 |  5659.58 |
| ABMD       |  419800 |  96.64 |  4453.17 |
| ABMD       |  408700 |  96.86 |  4239.19 |
| ABMD       |  503800 |  95.95 |  5306.51 |
| ABMD       | 1641100 |  101.4 |  16593.5 |
| ABMD       |  561700 | 102.43 |  5620.37 |
| ABMD       |  315500 | 102.34 |  3084.07 |
| ABMD       |  410500 | 103.12 |  4031.23 |
| ABMD       |  825000 | 103.17 |  8218.77 |
+------------+---------+--------+----------+
20 rows in set (0.00 sec)
```

6. Write a Hive query to find the dates where previous day close and current day open difference is greater than 0 for each stock.

------> EXTERNAL TABLE CREATION:

```
hive> create external table Que6(Stock_Name string,Date string) row format delimited
fields terminated by ',' location '/user/hive/warehouse/Que6/result.txt';
```

------>TRANSFERRING OUTPUT DATA TO EXTERNAL TABLE:

```
hive> insert overwrite table Que6
SELECT stock_name, date
FROM (SELECT stock_name, date,lag(close,1) OVER (PARTITION BY stock_name
ORDER BY date) AS prev_close,open
FROM Stock) sub
WHERE (open - prev_close) > 0;
```

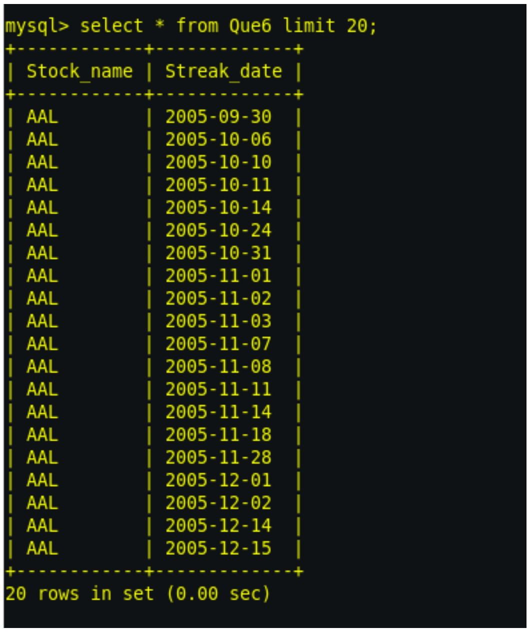
------>MYSQL TABLE CREATED (CLIENT DATABASE)

```
mysql> create table Que6(Stock_name varchar(20),Streak_date date);
```

------>SQOOP COMMAND TO TRANSFER O/P TABLE TO MYSQL

```
[cloudera@quickstart ~]$ sqoop export --connect jdbc:mysql://localhost:3306/project --
username root --password cloudera --table Que6 --export-dir
/user/hive/warehouse/Que6/result.txt/000000_0 --input-fields-terminated-by ','
```

------>FINAL OUTPUT DATA –



```
mysql> select * from Que6 limit 20;
+------------+------------+
| Stock_name | Streak_date |
+------------+------------+
| AAL        | 2005-09-30 |
| AAL        | 2005-10-06 |
| AAL        | 2005-10-10 |
| AAL        | 2005-10-11 |
| AAL        | 2005-10-14 |
| AAL        | 2005-10-24 |
| AAL        | 2005-10-31 |
| AAL        | 2005-11-01 |
| AAL        | 2005-11-02 |
| AAL        | 2005-11-03 |
| AAL        | 2005-11-07 |
| AAL        | 2005-11-08 |
| AAL        | 2005-11-11 |
| AAL        | 2005-11-14 |
| AAL        | 2005-11-18 |
| AAL        | 2005-11-28 |
| AAL        | 2005-12-01 |
| AAL        | 2005-12-02 |
| AAL        | 2005-12-14 |
| AAL        | 2005-12-15 |
+------------+------------+
20 rows in set (0.00 sec)
```

7. Find median of volume for ABIO.

------> EXTERNAL TABLE CREATION:

```
hive> create external table Que7(median_volume float) row format delimited fields terminated by ',' location '/user/hive/warehouse/Que7/result.txt';
```

------>TRANSFERRING OUTPUT DATA TO EXTERNAL TABLE:

```
hive> insert overwrite table Que7
SELECT percentile(volume, 0.5) AS median_volume
FROM Stock WHERE stock_name = 'ABIO';
```

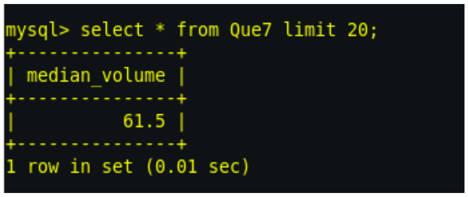
------>MYSQL TABLE CREATED (CLIENT DATABASE)

```
mysql> create table Que7(median_volume float);
```

------>SQOOP COMMAND TO TRANSFER O/P TABLE TO MYSQL

[cloudera@quickstart ~]$ sqoop export --connect jdbc:mysql://localhost:3306/project --username root --password cloudera --table Que7 --export-dir /user/hive/warehouse/Que7/result.txt/000000_0 --input-fields-terminated-by ','

------>FINAL OUTPUT DATA –



```
mysql> select * from Que7 limit 20;
+---------------+
| median_volume |
+---------------+
|          61.5 |
+---------------+
1 row in set (0.01 sec)
```

PYSPARK – Analysis

IMPORT LIBRARIES

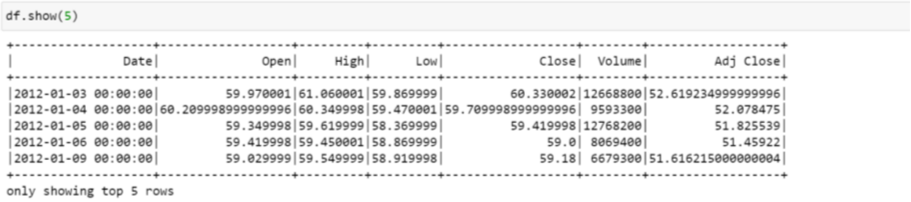
- ✓ import findspark
- ✓ import pyspark
- ✓ from pyspark.sql import SparkSession
- ✓ from pyspark.sql.functions import *

READ DATAFILE

spark1 = SparkSession.builder.appName('Walmart_project').getOrCreate()

df=spark1.read.csv("walmart_stock.csv",inferSchema=True,header=True)

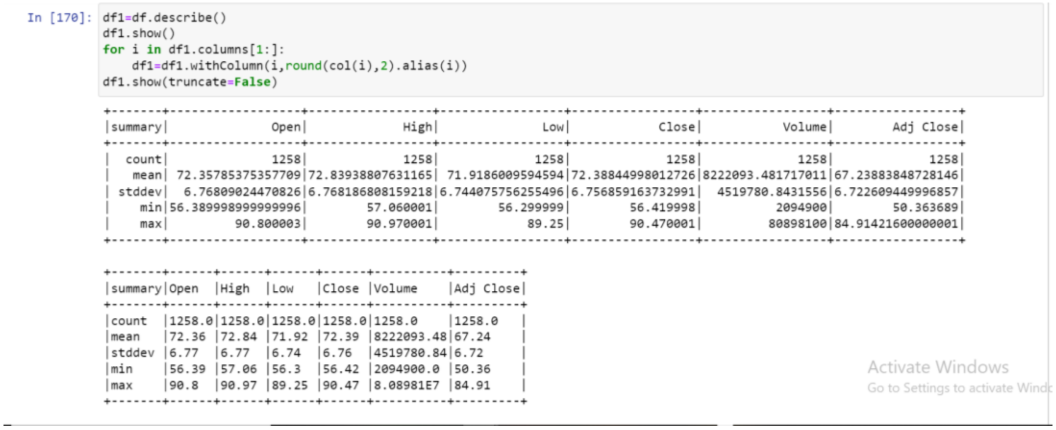
scenario 1: print out first 5 columns



```
df.show(5)
```

```
+-------------------+------------------+---------+---------+------------------+--------+------------------+
|               Date|              Open|     High|      Low|             Close|  Volume|         Adj Close|
+-------------------+------------------+---------+---------+------------------+--------+------------------+
|2012-01-03 00:00:00|         59.970001|61.060001|59.869999|         60.330002|12668800|52.619234999999996|
|2012-01-04 00:00:00|60.209998999999996|60.349998|59.470001|59.709998999999996| 9593300|         52.078475|
|2012-01-05 00:00:00|         59.349998|59.619999|58.369999|         59.419998|12768200|         51.825539|
|2012-01-06 00:00:00|         59.419998|59.450001|58.869999|              59.0| 8069400|          51.45922|
|2012-01-09 00:00:00|         59.029999|59.549999|58.919998|             59.18| 6679300|51.616215000000004|
+-------------------+------------------+---------+---------+------------------+--------+------------------+
only showing top 5 rows
```

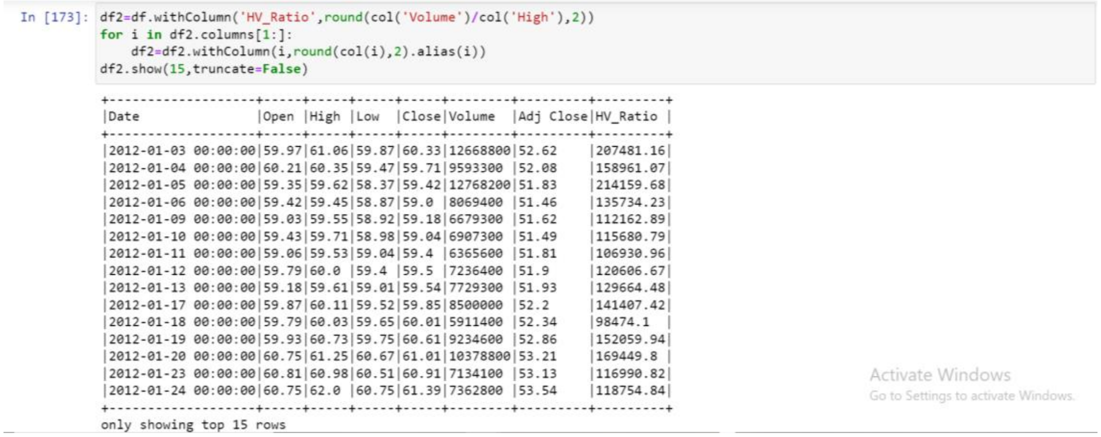
scenario 2: There are too many decimal places for mean and stddev in the describe() dataframe. Format the numbers to just show up to two decimal places.Pay careful attention to the datatypes that .describe() returns, we didn't cover how to do this exact formatting,but we covered something very similar.



```
In [170]: df1=df.describe()
          df1.show()
          for i in df1.columns[1:]:
              df1=df1.withColumn(i,round(col(i),2).alias(i))
          df1.show(truncate=False)
```

```
+-------+------------------+-----------------+-----------------+-----------------+-----------------+-----------------+
|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|
+-------+------------------+-----------------+-----------------+-----------------+-----------------+-----------------+

+-------+------+------+------+------+----------+---------+
|summary|Open  |High  |Low   |Close |Volume    |Adj Close|
+-------+------+------+------+------+----------+---------+
|count  |1258.0|1258.0|1258.0|1258.0|1258.0    |1258.0   |
|mean   |72.36 |72.84 |71.92 |72.39 |8222093.48|67.24    |
|stddev |6.77  |6.77  |6.74  |6.76  |4519780.84|6.72     |
|min    |56.39 |57.06 |56.3  |56.42 |2094900.0 |50.36    |
|max    |90.8  |90.97 |89.25 |90.47 |8.08981E7 |84.91    |
+-------+------+------+------+------+----------+---------+
```

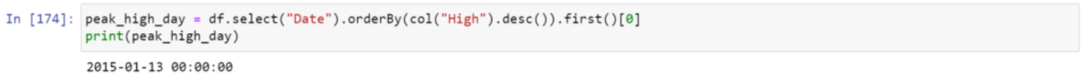
scenario3: Create a new dataframe with a column called HV Ratio that is the ratio of the High Price versus volume of stock traded for a day.?



```
In [173]: df2=df.withColumn('HV_Ratio',round(col('Volume')/col('High'),2))
          for i in df2.columns[1:]:
              df2=df2.withColumn(i,round(col(i),2).alias(i))
          df2.show(15,truncate=False)
```

```
+-------------------+-----+-----+-----+-----+--------+---------+---------+
|Date               |Open |High |Low  |Close|Volume  |Adj Close|HV_Ratio |
+-------------------+-----+-----+-----+-----+--------+---------+---------+
|2012-01-03 00:00:00|59.97|61.06|59.87|60.33|12668800|52.62    |207481.16|
|2012-01-04 00:00:00|60.21|60.35|59.47|59.71|9593300 |52.08    |158961.07|
|2012-01-05 00:00:00|59.35|59.62|58.37|59.42|12768200|51.83    |214159.68|
|2012-01-06 00:00:00|59.42|59.45|58.87|59.0 |8069400 |51.46    |135734.23|
|2012-01-09 00:00:00|59.03|59.55|58.92|59.18|6679300 |51.62    |112162.89|
|2012-01-10 00:00:00|59.43|59.71|58.98|59.04|6907300 |51.49    |115680.79|
|2012-01-11 00:00:00|59.06|59.53|59.04|59.4 |6365600 |51.81    |106930.96|
|2012-01-12 00:00:00|59.79|60.0 |59.4 |59.5 |7236400 |51.9     |120606.67|
|2012-01-13 00:00:00|59.18|59.61|59.01|59.54|7729300 |51.93    |129664.48|
|2012-01-17 00:00:00|59.87|60.11|59.52|59.85|8500000 |52.2     |141407.42|
|2012-01-18 00:00:00|59.79|60.03|59.65|60.01|5911400 |52.34    |98474.1  |
|2012-01-19 00:00:00|59.93|60.73|59.75|60.61|9234600 |52.86    |152059.94|
|2012-01-20 00:00:00|60.75|61.25|60.67|61.01|10378800|53.21    |169449.8 |
|2012-01-23 00:00:00|60.81|60.98|60.51|60.91|7134100 |53.13    |116990.82|
|2012-01-24 00:00:00|60.75|62.0 |60.75|61.39|7362800 |53.54    |118754.84|
+-------------------+-----+-----+-----+-----+--------+---------+---------+
only showing top 15 rows
```

scenario4: What day had the Peak High in Price?

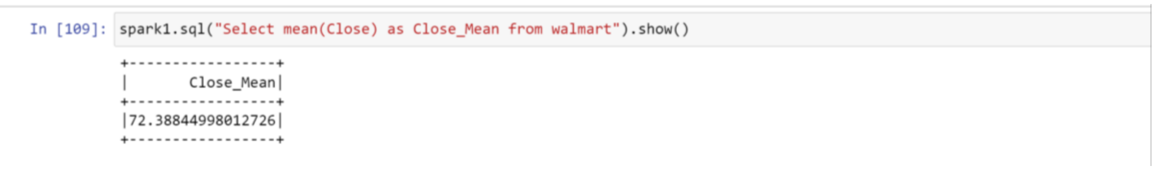


```
In [174]: peak_high_day = df.select("Date").orderBy(col("High").desc()).first()[0]
          print(peak_high_day)

          2015-01-13 00:00:00
```

SPARK-SQL – Analysis

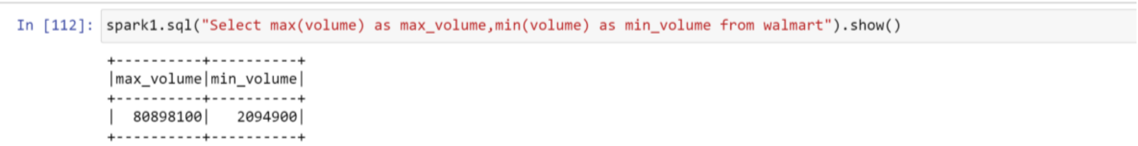
- df.createOrReplaceTempView("walmart")

scenario5: What is the mean of the Close column?



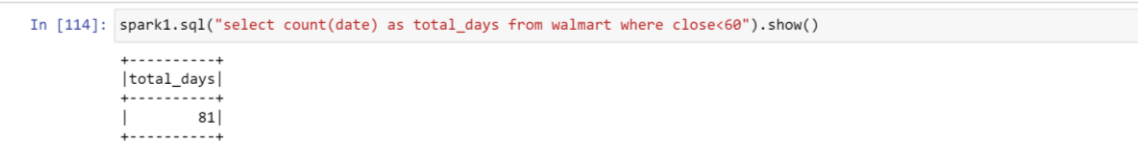
```
In [109]: spark1.sql("Select mean(Close) as Close_Mean from walmart").show()

          +-----------------+
          |       Close_Mean|
          +-----------------+
          |72.38844998012726|
          +-----------------+
```

scenario6: What is the max and min of the Volume column?



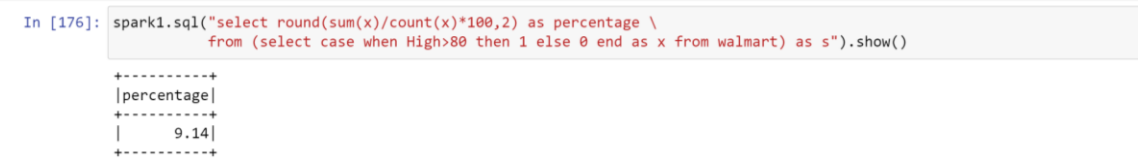
```
In [112]: spark1.sql("Select max(volume) as max_volume,min(volume) as min_volume from walmart").show()

          +----------+----------+
          |max_volume|min_volume|
          +----------+----------+
          |  80898100|   2094900|
          +----------+----------+
```

scenario7: How many days was the Close lower than 60 dollars?



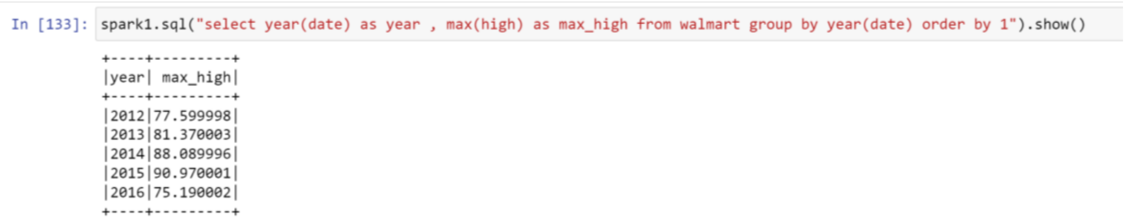
```
In [114]: spark1.sql("select count(date) as total_days from walmart where close<60").show()

          +----------+
          |total_days|
          +----------+
          |        81|
          +----------+
```

Scenario8: 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)



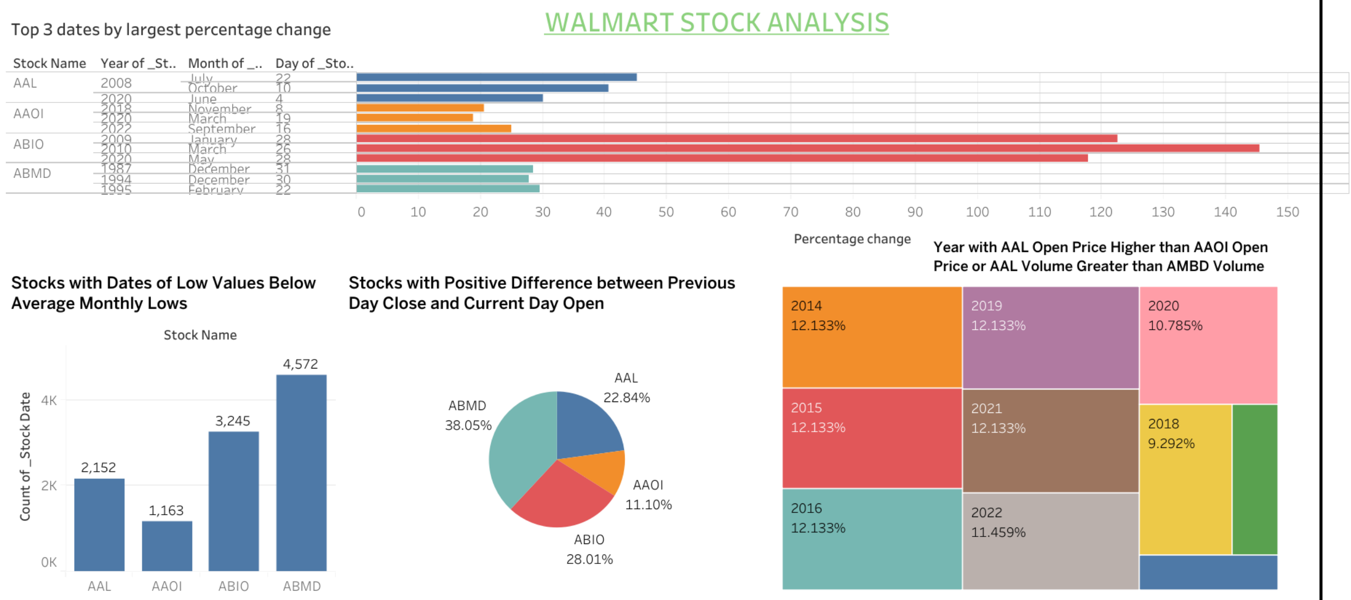
```
In [176]: spark1.sql("select round(sum(x)/count(x)*100,2) as percentage \
                    from (select case when High>80 then 1 else 0 end as x from walmart) as s").show()

          +----------+
          |percentage|
          +----------+
          |      9.14|
          +----------+
```

Scenario9: What is the max High per year?



```
In [133]: spark1.sql("select year(date) as year , max(high) as max_high from walmart group by year(date) order by 1").show()

          +----+---------+
          |year| max_high|
          +----+---------+
          |2012|77.599998|
          |2013|81.370003|
          |2014|88.089996|
          |2015|90.970001|
          |2016|75.190002|
          +----+---------+
```

Visualization



WALMART STOCK ANALYSIS

Top 3 dates by largest percentage change

Stock Name	Year of _St..	Month of _..	Day of _Sto..
AAL	2008	July	22
		October	10
	2020	June	4
AAOI	2018	November	8
	2020	March	19
	2022	September	16
ABIO	2009	January	28
	2010	March	26
	2020	May	28
ABMD	1987	December	31
	1994	December	30
	1995	February	22

Stocks with Dates of Low Values Below Average Monthly Lows

Stocks with Positive Difference between Previous Day Close and Current Day Open

Year with AAL Open Price Higher than AAOI Open Price or AAL Volume Greater than AMBD Volume