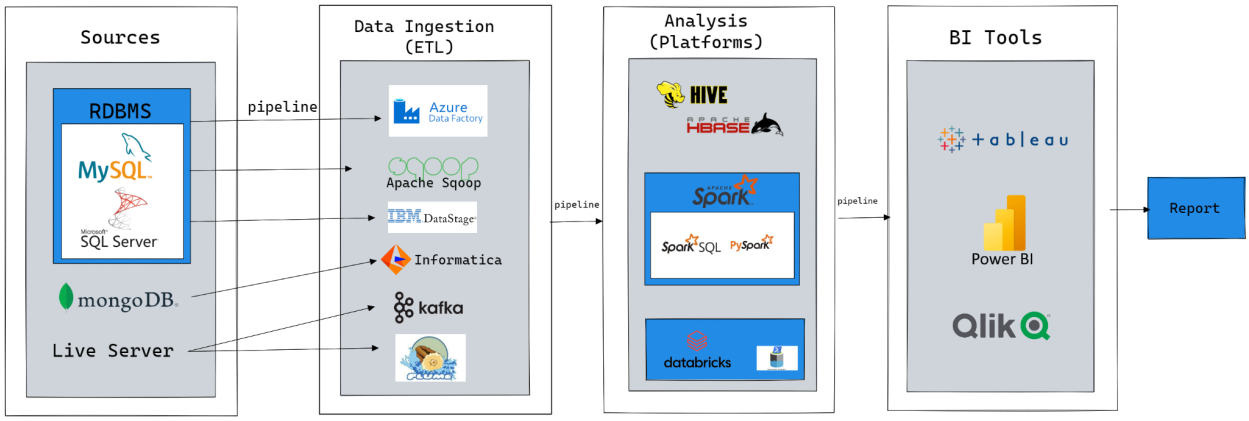
Walmart Stock Analysis

Group 06: Roshan, Gangothri, Narendra, Chandan, Faryar

Big Data Architecture



Analysis: Hive, PySpark, Spark SQL

Scenario 01: Print out first 5 columns

```
In [22]: first5Column = spark.sql("select Date, Open, High, Low, Close from walmart_stock")
```

```
In [23]: first5Column.show(4)
```

```
+-------------------+-----------------+---------+---------+-----------------+
|               Date|             Open|     High|      Low|            Close|
+-------------------+-----------------+---------+---------+-----------------+
|2012-01-03 00:00:00|        59.970001|61.060001|59.869999|        60.330002|
|2012-01-04 00:00:00|60.209998999999996|60.349998|59.470001|59.709998999999996|
|2012-01-05 00:00:00|        59.349998|59.619999|58.369999|        59.419998|
|2012-01-06 00:00:00|        59.419998|59.450001|58.869999|             59.0|
+-------------------+-----------------+---------+---------+-----------------+
only showing top 4 rows
```

Scenario 02: Describe function, formatted to two decimal place

```
describeDF = walmart_df.describe()
```

```
from pyspark.sql.functions import *
describeDF = describeDF.select(
  col("summary"),
  format_number(col("Open").cast("double"), 2).alias("Open"),
  format_number(col("High").cast("double"), 2).alias("High"),
  format_number(col("Low").cast("double"), 2).alias("Low"),
  format_number(col("Close").cast("double"), 2).alias("Close"),
  format_number(col("Volume").cast("double"), 2).alias("Volume"),
  format_number(col("Adj Close").cast("double"), 2).alias("Adj Close")
)
describeDF.show()
```

```
describeDF.show()
```

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	8,222,093.48	67.24
stddev	6.77	6.77	6.74	6.76	4,519,780.84	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	80,898,100.00	84.91

Scenario 03: Create a new dataframe with a column called HighToLow Ratio that is the ratio of the High Price versus Low Price of stock traded for a day.

```
# Now we have the desired Output we can store it in a new dataframe

walmart_stock_updated = spark.sql('''SELECT Date, round(Open, 2) as Open, round(High, 2) as High,
                                     round(Low, 2) as Low, round(Close,2) as Close, Volume,
                                     round(`Adj Close`,2) as `Adj Close`,
                                     round(Open/High, 3) as `OpenToHighRatio` FROM walmart_data ''')
```

```
walmart_stock_updated.show()
```

```
+-------------------+-----+-----+-----+-----+--------+---------+---------------+
|               Date| Open| High|  Low|Close|  Volume|Adj Close|OpenToHighRatio|
+-------------------+-----+-----+-----+-----+--------+---------+---------------+
|2012-01-03 00:00:00|59.97|61.06|59.87|60.33|12668800|    52.62|          0.982|
|2012-01-04 00:00:00|60.21|60.35|59.47|59.71| 9593300|    52.08|          0.998|
|2012-01-05 00:00:00|59.35|59.62|58.37|59.42|12768200|    51.83|          0.995|
|2012-01-06 00:00:00|59.42|59.45|58.87| 59.0| 8069400|    51.46|          0.999|
|2012-01-09 00:00:00|59.03|59.55|58.92|59.18| 6679300|    51.62|          0.991|
|2012-01-10 00:00:00|59.43|59.71|58.98|59.04| 6907300|    51.49|          0.995|
|2012-01-11 00:00:00|59.06|59.53|59.04| 59.4| 6365600|    51.81|          0.992|
|2012-01-12 00:00:00|59.79| 60.0| 59.4| 59.5| 7236400|     51.9|          0.997|
|2012-01-13 00:00:00|59.18|59.61|59.01|59.54| 7729300|    51.93|          0.993|
|2012-01-17 00:00:00|59.87|60.11|59.52|59.85| 8500000|     52.2|          0.996|
|2012-01-18 00:00:00|59.79|60.03|59.65|60.01| 5911400|    52.34|          0.996|
|2012-01-19 00:00:00|59.93|60.73|59.75|60.61| 9234600|    52.86|          0.987|
|2012-01-20 00:00:00|60.75|61.25|60.67|61.01|10378800|    53.21|          0.992|
|2012-01-23 00:00:00|60.81|60.98|60.51|60.91| 7134100|    53.13|          0.997|
|2012-01-24 00:00:00|60.75| 62.0|60.75|61.39| 7362800|    53.54|           0.98|
|2012-01-25 00:00:00|61.18|61.61|61.04|61.47| 5915800|    53.61|          0.993|
|2012-01-26 00:00:00| 61.8|61.84|60.77|60.97| 7436200|    53.18|          0.999|
|2012-01-27 00:00:00|60.86|61.12|60.54|60.71| 6287300|    52.95|          0.996|
|2012-01-30 00:00:00|60.47|61.32|60.35| 61.3| 7636900|    53.47|          0.986|
|2012-01-31 00:00:00|61.53|61.57|60.58|61.36| 9761500|    53.52|          0.999|
+-------------------+-----+-----+-----+-----+--------+---------+---------------+
only showing top 20 rows
```

Scenario 04: What day had the Peak High in Price?

```
spark.sql("SELECT Date, High FROM walmart_data WHERE HIGH = (SELECT MAX(High) FROM walmart_data) ").show()
```

```
+-------------------+---------+
|               Date|     High|
+-------------------+---------+
|2015-01-13 00:00:00|90.970001|
+-------------------+---------+
```

Scenario 05: What is the mean of the Close column?

```
In [10]: closing_price_mean = spark.sql('SELECT ROUND(AVG(Close),3) AS closing_mean FROM walmart_stock')
```

```
In [11]: closing_price_mean.show()
```

```
+------------+
|closing_mean|
+------------+
|      72.388|
+------------+
```

Scenario 06: What is the max and min of the Volume column?

```
data_rdd=sc.textFile("walmart_stock.csv")

headers = data_rdd.first()

rdd = data_rdd.filter(lambda line:line!=headers)

max_volume = rdd.map(lambda line:float(line.split(",")[5])).max()

min_volume = rdd.map(lambda line:float(line.split(",")[5])).min()

df_max_min=sc.parallelize([Row(max_volume,min_volume)]).toDF(["max_volume","min_volume"])

df_max_min.createOrReplaceTempView("min_max_volume")

df_max_min.show()
```

```
+----------+----------+
|max_volume|min_volume|
+----------+----------+
| 8.08981E7| 2094900.0|
+----------+----------+
```

Scenario 07: How many days was the Close lower than 60 dollars?

```
headers = data_rdd.first()

rdd = data_rdd.filter(lambda line:line!=headers)

cnt=rdd.filter(lambda line: float(line.split(",")[4])<60.00).count()

df_cnt=sc.parallelize([Row(cnt)]).toDF(["close_lower_than_60"])

df_cnt.createOrReplaceTempView("close_lower_than_60")

df_cnt.show()
```

```
+-------------------+
|close_lower_than_60|
+-------------------+
|                 81|
+-------------------+
```

Scenario 08: What percentage of the time was the High greater than 80 dollars?

```
hive> create table walmart_stock(Date string,Open double,high double,Low double,close double,
volume int,Adj_close double)row format delimited fields terminated by ',' tblproperties("skip
.header.line.count"="1");
```

```
hive> select round((count(high_date)/count(date))*100,2) as ratio from(select *,if(high>80,1,
null) as high_date from walmart_stock)a;
```

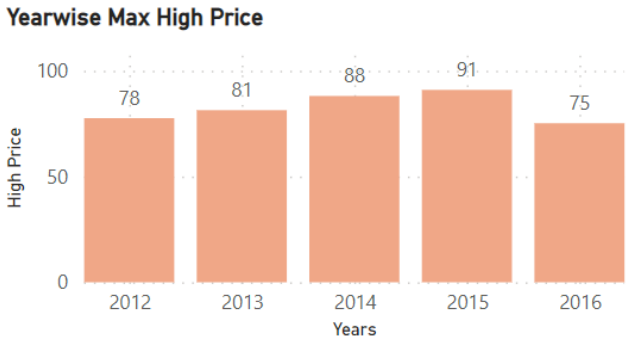
```
mysql> select * from high_80perc;
+-------------+
| perecentage |
+-------------+
|        9.14 |
+-------------+
1 row in set (0.00 sec)
```

Scenario 09: What is the max High per year?

```
hive> create table walmart_stock(Date string,Open double,high double,Low double,close double,
volume int,Adj_close double)row format delimited fields terminated by ',' tblproperties("skip
.header.line.count"="1");
```

```
hive> select year(date),max(high) from walmart_stock group by year(date);
```

```
mysql> select * from max_high_year;
+------+----------+
| date | max_high |
+------+----------+
| 2012 |     77.6 |
| 2016 |    75.19 |
| 2013 |    81.37 |
| 2014 |    88.09 |
| 2015 |    90.97 |
+------+----------+
5 rows in set (0.01 sec)
```

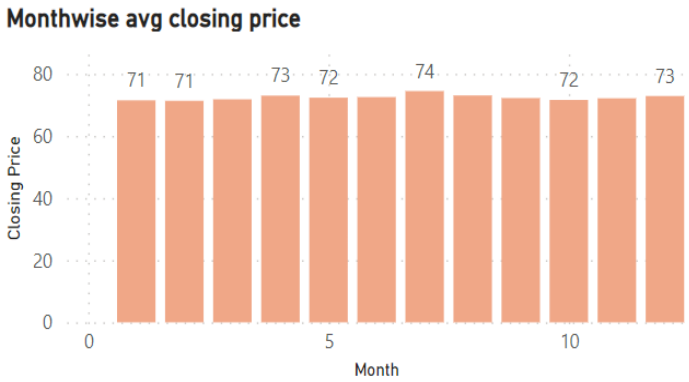


Scenario 10: What is the average Close for each Calendar Month? In other words, across all the years, what is the average Close price for Jan,Feb, Mar, etc... Your result will have a value for each of these months

In [14]:
```
monthly_avg_closing = spark.sql('''SELECT MONTH(Date) AS month, ROUND(AVG(Close),3)
                                   AS closing_mean FROM walmart_stock GROUP BY MONTH(Date) ORDER BY month''')
```

In [15]:
```
monthly_avg_closing.show()
```

```
+-----+------------+
|month|closing_mean|
+-----+------------+
|    1|      71.448|
|    2|      71.307|
|    3|      71.778|
|    4|      72.974|
|    5|       72.31|
|    6|      72.495|
|    7|       74.44|
|    8|       73.03|
|    9|      72.184|
|   10|      71.579|
|   11|      72.111|
|   12|      72.848|
+-----+------------+
```



Scenario 11: What is the Pearson correlation between High and Volume

In [15]:
```
pearson_corr = df.select(corr("High", "Volume").alias('Pearson Correlation'))
pearson_corr.show()
```

```
+-------------------+
|Pearson Correlation|
+-------------------+
|-0.3384326061737161|
+-------------------+
```