

Financial Statement Analysis of S&P 500 stocks

Group 10
Yang Liu, Jiawei Xue

Agenda

- 1.Introduction
- 2.Data sources
- 3.Data model
- 4.Data summary
- 5.Data preparation
- 6.Propose questions and answers
- 7.Conclusion
- 8.Future work
- 9.References

Introduction

Financial statement analysis is the process of analyzing a company's financial statements for decision-making purposes and to seek the undervalued stocks for investments. Financial statements record financial data, which must be evaluated through financial statement analysis to become more useful to investors, shareholders, managers, and other interested parties.

In this project, we attempted to use Mysql and financial statement analysis to achieve two goals.

First, we aimed to implement the value investing method, known as Warren Buffett's and Benjamin Graham's investment methodology, to find out which stocks to invest in. Value investing is an investment strategy where stocks are selected and traded for less than their intrinsic values. Using this investment strategy, we could find the opportunity to profit because most of the speculators on the market overreact to good and bad news, resulting in stock price movements that do not correspond with a company's long-term fundamentals.

Second, we utilized the DuPont analysis to analyze a company's ability to increase its return on equity. In other words, this model breaks down the return on equity ratio to explain how companies can increase their return for investors. The Dupont analysis looks at three main components of the ROE ratio: profit margin, total asset turnover, and financial leverage. Based on these three

performance measures the model concludes that a company can raise its ROE by maintaining a high profit margin, increasing asset turnover, or leveraging assets more effectively.

Data Sources

We retrieved our datasets from Kaggle-New York Stock Exchange(<https://www.kaggle.com/dgawlik/nyse>) (<https://www.kaggle.com/dgawlik/nyse>), one competition on the most famous data science community.

The datasets in this Kaggle competition consisted three following files:

1.Securities.csv: general information of each company information from EDGAR SEC databases (<https://www.sec.gov/edgar/searchedgar/companysearch.html>) (<https://www.sec.gov/edgar/searchedgar/companysearch.html>), such as stock symbol, company name, industry, address, the first date that a company became a listed company and the CIK number of each company (A Central Index Key or CIK number is a number given to an individual or company by the United States Securities and Exchange Commission. The number is used to identify the filings of a company, person, or entity in several online databases, including EDGAR).

2.Financial statements.csv: Public companies' financial statements' data from Nasdaq Financials(<https://www.nasdaq.com/quotes/company-financials.aspx>) (<https://www.nasdaq.com/quotes/company-financials.aspx>).

Financial statements is a formal record of the financial activities and position of a business, person, or other entity, it reports on a company's assets, liabilities, and owners equity at a given point in time, a company's income, expenses, and profits over a period of time, the changes in equity of the company during the stated period, and a company's cash flow activities, particularly its operating, investing and financing activities.

3.Prices.csv: Historical stock data from Yahoo Finance (<https://finance.yahoo.com/>) (<https://finance.yahoo.com/>), contains the information of stock symbol, transaction date, open price, close price, lowest price, highest price and the transaction volume in one day.

Data Model

We built the entity-relationship diagram (ER model for short) as our data model. An ER diagram describes how entities relate to each other.

Design reasoning:

(a)By drawing ER diagrams to visualize database design ideas, we had a chance to identify the mistakes and design flaws, and to make corrections before executing the changes in the database.

(b)Considering that there may be some ambiguities or unnecessary processes if the users or other developers do not understand the database very well, we decided to present our data schema in a graphical form, which offers the users and developers a easier and more efficient way to manipulate and communicate.

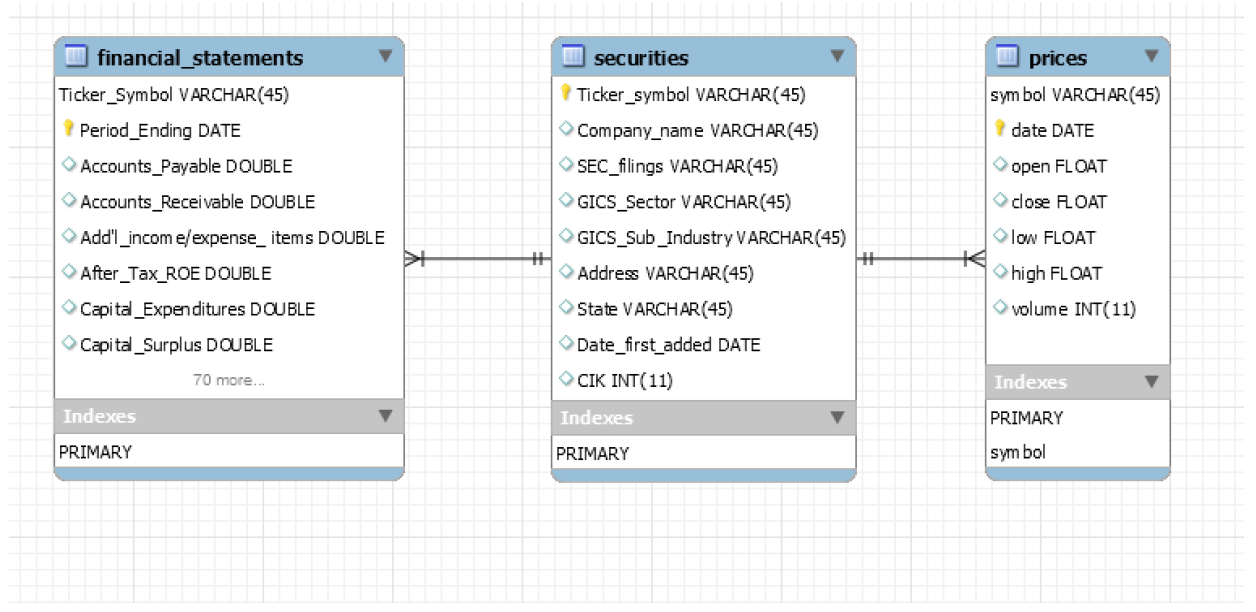
(c)By visualizing a database schema with an ERD, we have a full picture of the entire database schema. We can easily locate entities, view their attributes and to identify the relationships they have with others. These allows us to analyze an existing database and to reveal database problem easier.

Design method:

Our methodology when designing the ER diagram is to connect all the data we needed together and to implement this in the simplest way. The main table in the ERD is the securities table because it has unique values in the column of ticker_symbol and each row could be joined to other tables which also contain the column of ticker_symbol. The primary key in the securities table is ticker_symbol.

For the tables of financial_statements and prices, as they do not have unique values in any column, we chose to use composite keys as their primary keys. The composite primary key of financial_statements is ticker_symbol and period_ending. The values of this combination in each row is unique. Similarly, the composite primary key of prices are symbol and date. The financial_statements uses ticker_symbol as its foreign key while prices uses symbol as its foreign key so that they can both refer to ticker_symbol, the primary key of securities.

In summary, both the relationships between securities and financial_statements and between securities and prices are one to many relationships.

ER Diagram**Data Summary**

Tables	Number of records	Period
securitis	505	NA
prices	851264	2010-01-04 to 2016-12-30
financial statements	1776	2012-03-03 to 2016-12-31

Samples of data records

Securites

Ticker_symbol	Company_name	SEC_filings	GICS_Sector	GICS_Sub_Industry	Address	State	Date_first_added	CIK
A	Agilent Technologies Inc	reports	Health Care	Health Care Equipment	Santa Clara	California	0000-00-00	1090872
AAL	American Airlines Group	reports	Industrials	Airlines	Fort Worth	Texas	2015-03-23	6201
AAP	Advance Auto Parts	reports	Consumer Discretionary	Automotive Retail	Roanoke	Virginia	2015-07-09	1158449
AAPL	Apple Inc.	reports	Information Technology	Computer Hardware	Cupertino	California	1982-11-30	320193
ABBV	AbbVie	reports	Health Care	Pharmaceuticals	North Chicago	Illinois	2012-12-31	1551152

Prices

symbol	date	open	close	low	high	volume
A	2010-01-04	22.4535	22.3891	22.2675	22.6252	3815500
AAL	2010-01-04	4.84	4.77	4.66	4.94	9837300
AAP	2010-01-04	40.7	40.38	40.36	41.04	1701700
AAPL	2010-01-04	30.49	30.5729	30.34	30.6429	123432400
ABC	2010-01-04	26.29	26.63	26.14	26.69	2455900

Financial Statements

Ticker_Symbol	Period_Ending	Accounts_Payable	Accounts_Receivable	Add'l_Income/expense_items	After_Tax_ROE	Capital_Expenditures	Capital_Surplus
AAL	2012-12-31	3068000000.0	-222000000.0	-1961000000.0	23.0	-1888000000.0	4695000000.0
AAL	2013-12-31	4975000000.0	-93000000.0	-2723000000.0	67.0	-3114000000.0	10592000000.0
AAL	2014-12-31	4668000000.0	-160000000.0	-150000000.0	143.0	-5311000000.0	15135000000.0
AAL	2015-12-31	5102000000.0	352000000.0	-708000000.0	135.0	-6151000000.0	11591000000.0
AAP	2012-12-29	2409453000.0	-89482000.0	600000.0	32.0	-271182000.0	520215000.0

Cash_Ratio	Cash_and_Cash_Equivalents	Changes_in_Inventories	Common_Stocks	Cost_of_Revenue	Current_Ratio	Deferred_Asset_Charges	Deferred_Liability
53.0	1330000000.0	0.0	127000000.0	10499000000.0	78.0	0.0	223
75.0	2175000000.0	0.0	5000000.0	11019000000.0	104.0	0.0	93€
60.0	1768000000.0	0.0	7000000.0	15620000000.0	88.0	0.0	82€
51.0	1085000000.0	0.0	6000000.0	11096000000.0	73.0	2477000000.0	667
23.0	598111000.0	-260298000.0	7000.0	3106967000.0	124.0	0.0	

Depreciation	Earnings_Before_Interest_and_Tax	Earnings_Before_Tax	Effect_of_Exchange_Rate	Equity_Earnings/Loss_Unconsolidated_Subsiary	Fixed_Assets
1001000000.0	-1813000000.0	-2445000000.0	0.0	0.0	13402000000.0
1020000000.0	-1324000000.0	-2180000000.0	0.0	0.0	19259000000.0
1342000000.0	4099000000.0	3212000000.0	0.0	0.0	23084000000.0
1487000000.0	5496000000.0	4616000000.0	0.0	0.0	27510000000.0
189544000.0	657915000.0	624074000.0	0.0	0.0	1292547000.0

Goodwill	Gross_Margin	Gross_Profit	Income_Tax	Intangible_Assets	Interest_Expense	Inventory	Investments	Liabilities	Long-Term_Debt
0.0	58.0	14356000000.0	-569000000.0	869000000.0	632000000.0	580000000.0	306000000.0	473000000.0	7116000000.0
4086000000.0	59.0	15724000000.0	-346000000.0	2311000000.0	856000000.0	1012000000.0	-1181000000.0	-235000000.0	15353000000.0
4091000000.0	63.0	27030000000.0	330000000.0	2240000000.0	887000000.0	1004000000.0	1799000000.0	-1026000000.0	16043000000.0
4091000000.0	73.0	29894000000.0	-2994000000.0	2249000000.0	880000000.0	863000000.0	443000000.0	-633000000.0	18330000000.0
76389000.0	50.0	3098036000.0	236404000.0	28845000.0	33841000.0	2308609000.0	0.0	426323000.0	604461000.0

Long-Term_Investments	Minority_Interest	Misc_Stocks	Net_Borrowings	Net_Cash_Flow	Net_Cash_Flow-Operating	Net_Cash_Flows-Financing	Net_Cash_Flows-Investing	Net_Income
0.0	0.0	0.0	-1020000000.0	197000000.0	1285000000.0	483000000.0	-1571000000.0	-1876000000.0
0.0	0.0	0.0	2208000000.0	660000000.0	675000000.0	3799000000.0	-3814000000.0	-1834000000.0
0.0	0.0	0.0	170000000.0	-146000000.0	3080000000.0	-315000000.0	-2911000000.0	2882000000.0
0.0	0.0	0.0	2856000000.0	-604000000.0	6249000000.0	-1259000000.0	-5594000000.0	7610000000.0
0.0	0.0	0.0	177445000.0	540210000.0	685281000.0	127907000.0	-272978000.0	387670000.0
Net_Income_Adjustments	Net_Income_Applicable_to_Common_Shareholders			Net_Income-Cont_Operations	Net_Receivables	Non-Recurring_Items	Operating_Income	
2050000000.0				-1876000000.0	-4084000000.0	1124000000.0	386000000.0	148000000.0
1873000000.0				-1834000000.0	-4489000000.0	1560000000.0	559000000.0	1399000000.0
542000000.0				2882000000.0	2882000000.0	1771000000.0	800000000.0	4249000000.0
-2662000000.0				7610000000.0	7610000000.0	1425000000.0	1051000000.0	6204000000.0
23311000.0				387670000.0	387670000.0	229866000.0	0.0	657315000.0
Operating_Margin	Other_Assets	Other_Current_Assets	Other_Current_Liabilities	Other_Equity	Other_Financing_Activities	Other_Investing_Activities		
1.0	2167000000.0	626000000.0	4524000000.0	-2980000000.0	1509000000.0	11000000.0		
5.0	2299000000.0	1465000000.0	7385000000.0	-2032000000.0	1711000000.0	481000000.0		
10.0	2060000000.0	898000000.0	7059000000.0	-4559000000.0	817000000.0	601000000.0		
15.0	2103000000.0	748000000.0	6272000000.0	-4732000000.0	96000000.0	114000000.0		
11.0	31833000.0	47614000.0	149558000.0	2667000.0	-33499000.0	-1796000.0		
Other_Liabilities	Other_Operating_Activities	Other_Operating_Items	Pre-Tax_Margin	Pre-Tax_ROE	Profit_Margin	Quick_Ratio	Research_and_Development	
15147000000.0	-141000000.0	845000000.0	10.0	31.0	8.0	72.0	0.0	
14915000000.0	-56000000.0	853000000.0	8.0	80.0	7.0	96.0	0.0	
10928000000.0	-500000000.0	1295000000.0	8.0	159.0	7.0	80.0	0.0	
10178000000.0	95000000.0	1364000000.0	11.0	82.0	19.0	67.0	0.0	
239021000.0	8213000.0	0.0	10.0	52.0	6.0	34.0	0.0	
Retained_Earnings	Sale_and_Purchase_of_Stock	Sales_General_and_Admin	Term_Debt/_Current_Portion_of_Long-Term_Debt	Short-Term_Investments	Short-Term_Investments		Total_Assets	
-9462000000.0	0.0	12977000000.0		1419000000.0	3412000000.0	23510000000.0		
-11296000000.0	0.0	12913000000.0		1446000000.0	8111000000.0	42278000000.0		
-8562000000.0	-1052000000.0	20686000000.0		1677000000.0	6309000000.0	43225000000.0		
-1230000000.0	-3846000000.0	21275000000.0		2231000000.0	5864000000.0	48415000000.0		
714900000.0	-18600000.0	2440721000.0		627000.0	0.0	4613814000.0		

Data Preparation

```
In [1]: import warnings
warnings.filterwarnings('ignore')
```

```
In [2]: %load_ext sql
        %sql mysql+pymysql://root:@fe512_mysql/fe512db
        %sql USE fe512db;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[2]: []

See the list of databases

```
In [3]: %sql SHOW DATABASES;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
6 rows affected.
```

Out[3]:

Database
FE512
fe512db
information_schema
mysql
performance_schema
sys

Choose the database we are going to use

```
In [4]: %sql SELECT DATABASE();
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[4]: DATABASE()

fe512db

Clear up the database by dropping unnecessary tables and views

```
In [5]: %%sql
DROP VIEW IF EXISTS MSQR;
DROP VIEW IF EXISTS HPE_earlier;
DROP VIEW IF EXISTS HPE_later;
DROP VIEW IF EXISTS DHI_earlier;
DROP VIEW IF EXISTS DHI_later;
DROP TABLE IF EXISTS ROE_SUM1;
DROP TABLE IF EXISTS prices;
DROP TABLE IF EXISTS financial_statements;
DROP TABLE IF EXISTS securities;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
```

Out[5]: []

See the list of existed tables

```
In [6]: %sql SHOW FULL TABLES;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
4 rows affected.
```

```
Out[6]: Tables_in_fe512db  Table_type
          13mar            VIEW
          16mar            VIEW
        2013report  BASE TABLE
        2016report  BASE TABLE
```

Create the table of securities

```
In [7]: %%sql
CREATE TABLE `fe512db`.`securities` (
  `Ticker_symbol` VARCHAR(45) NOT NULL,
  `Company_name` VARCHAR(45) NULL,
  `SEC_filings` VARCHAR(45) NULL,
  `GICS_Sector` VARCHAR(45) NULL,
  `GICS_Sub_Industry` VARCHAR(45) NULL,
  `Address` VARCHAR(45) NULL,
  `State` VARCHAR(45) NULL,
  `Date_first_added` Date NULL,
  `CIK` INT,
  PRIMARY KEY (`Ticker_symbol`));
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[7]: []

Check the creation of table securities

```
In [8]: %sql SHOW tables;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

Out[8]: **Tables_in_fe512db**

```
13mar
16mar
2013report
2016report
securities
```

Set up the configuration of mysql

```
In [9]: %sql SET sql_mode = '';
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[9]: []

Load the data into the table of securities


```
In [10]: %%sql
LOAD DATA INFILE '/home/data/securities.csv'
INTO TABLE securities
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
IGNORE 1 LINES;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
505 rows affected.
```

Out[10]: []

Check the creation of securities

```
In [11]: %sql DESCRIBE securities;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
9 rows affected.
```

Out[11]:

	Field	Type	Null	Key	Default	Extra
	Ticker_symbol	varchar(45)	NO	PRI	None	
	Company_name	varchar(45)	YES		None	
	SEC_filings	varchar(45)	YES		None	
	GICS_Sector	varchar(45)	YES		None	
	GICS_Sub_Industry	varchar(45)	YES		None	
	Address	varchar(45)	YES		None	
	State	varchar(45)	YES		None	
	Date_first_added	date	YES		None	
	CIK	int(11)	YES		None	

Show five rows of securities for example

In [12]: %sql SELECT * FROM securities LIMIT 5;

* mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.

Out[12]:

Ticker_symbol	Company_name	SEC_filings	GICS_Sector	GICS_Sub_Industry	Address	State
A	Agilent Technologies Inc	reports	Health Care	Health Care Equipment	Santa Clara	California
AAL	American Airlines Group	reports	Industrials	Airlines	Fort Worth	Texas
AAP	Advance Auto Parts	reports	Consumer Discretionary	Automotive Retail	Roanoke	Virginia
AAPL	Apple Inc.	reports	Information Technology	Computer Hardware	Cupertino	California
ABBV	AbbVie	reports	Health Care	Pharmaceuticals	North Chicago	Illinois

See how many rows in the table of securities

In [13]: %sql SELECT COUNT(*) FROM securities;

* mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.

Out[13]:

COUNT(*)
505

Create the table prices

```
In [14]: %%sql
CREATE TABLE `fe512db`.`prices` (
  `symbol` VARCHAR(45) NOT NULL,
  `date` DATE NOT NULL,
  `open` FLOAT NULL,
  `close` FLOAT NULL,
  `low` FLOAT NULL,
  `high` FLOAT NULL,
  `volume` INT NULL,
  PRIMARY KEY (`date`, `symbol`),
  CONSTRAINT `symbol`
    FOREIGN KEY (`symbol`)
    REFERENCES `fe512db`.`securities` (`Ticker_symbol`));
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[14]: []

Check the creation of prices

```
In [15]: %%sql DESCRIBE prices;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
7 rows affected.
```

Out[15]:

	Field	Type	Null	Key	Default	Extra
	symbol	varchar(45)	NO	PRI	None	
	date	date	NO	PRI	None	
	open	float	YES		None	
	close	float	YES		None	
	low	float	YES		None	
	high	float	YES		None	
	volume	int(11)	YES		None	

Load the data into table prices

```
In [16]: %%sql
LOAD DATA INFILE '/home/data/prices.csv'
INTO TABLE prices fields
TERMINATED BY ','
IGNORE 1 LINES;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
851264 rows affected.
```

Out[16]: []

See a sample of data records in prices

In [17]: %sql SELECT * FROM prices LIMIT 5;

* mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.

Out[17]:

	symbol	date	open	close	low	high	volume
	A	2010-01-04	22.4535	22.3891	22.2675	22.6252	3815500
	AAL	2010-01-04	4.84	4.77	4.66	4.94	9837300
	AAP	2010-01-04	40.7	40.38	40.36	41.04	1701700
	AAPL	2010-01-04	30.49	30.5729	30.34	30.6429	123432400
	ABC	2010-01-04	26.29	26.63	26.14	26.69	2455900

Check the total number of records and period in prices

In [18]: %sql SELECT COUNT(symbol),MIN(date), MAX(date) FROM prices;

* mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.

Out[18]:

	COUNT(symbol)	MIN(date)	MAX(date)
	851264	2010-01-04	2016-12-30

Create the table of financial_statements

```

In [19]: %%sql
CREATE TABLE `fe512db`.`financial_statements` (
  `Ticker_Symbol` VARCHAR(45) NOT NULL,
  `Period_Ending` DATE NOT NULL,
  `Accounts_Payable` DOUBLE NULL,
  `Accounts_Receivable` DOUBLE NULL,
  `Add'l_income/expense_items` DOUBLE NULL,
  `After_Tax_ROE` DOUBLE NULL,
  `Capital_Expenditures` DOUBLE NULL,
  `Capital_Surplus` DOUBLE NULL,
  `Cash_Ratio` DOUBLE NULL,
  `Cash_and_Cash_Equivalents` DOUBLE NULL,
  `Changes_in_Inventories` DOUBLE NULL,
  `Common_Stocks` DOUBLE NULL,
  `Cost_of_Revenue` DOUBLE NULL,
  `Current_Ratio` DOUBLE NULL,
  `Deferred_Asset_Charges` DOUBLE NULL,
  `Deferred_Liability_Charges` DOUBLE NULL,
  `Depreciation` DOUBLE NULL,
  `Earnings_Before_Interest_and_Tax` DOUBLE NULL,
  `Earnings_Before_Tax` DOUBLE NULL,
  `Effect_of_Exchange_Rate` DOUBLE NULL,
  `Equity_Earnings/Loss_Unconsolidated_Subsiary` DOUBLE NULL,
  `Fixed_Assets` DOUBLE NULL,
  `Goodwill` DOUBLE NULL,
  `Gross_Margin` DOUBLE NULL,
  `Gross_Profit` DOUBLE NULL,
  `Income_Tax` DOUBLE NULL,
  `Intangible_Assets` DOUBLE NULL,
  `Interest_Expense` DOUBLE NULL,
  `Inventory` DOUBLE NULL,
  `Investments` DOUBLE NULL,
  `Liabilities` DOUBLE NULL,
  `Long-Term_Debt` DOUBLE NULL,
  `Long-Term_Investments` DOUBLE NULL,
  `Minority_Interest` DOUBLE NULL,
  `Misc._Stocks` DOUBLE NULL,
  `Net_Borrowings` DOUBLE NULL,
  `Net_Cash_Flow` DOUBLE NULL,
  `Net_Cash_Flow-Operating` DOUBLE NULL,
  `Net_Cash_Flows-Financing` DOUBLE NULL,
  `Net_Cash_Flows-Investing` DOUBLE NULL,
  `Net_Income` DOUBLE NULL,
  `Net_Income_Adjustments` DOUBLE NULL,
  `Net_Income_Applicable_to_Common_Shareholders` DOUBLE NULL,
  `Net_Income-Cont._Operations` DOUBLE NULL,
  `Net_Receivables` DOUBLE NULL,
  `Non-Recurring_Items1` DOUBLE NULL,
  `Operating_Income` DOUBLE NULL,
  `Operating_Margin` DOUBLE NULL,
  `Other_Assets` DOUBLE NULL,
  `Other_Current_Assets` DOUBLE NULL,
  `Other_Current_Liabilities` DOUBLE NULL,
  `Other_Equity` DOUBLE NULL,
  `Other_Financing_Activities` DOUBLE NULL,
  `Other_Investing_Activities` DOUBLE NULL,

```

```

`Other_Liabilities` DOUBLE NULL,
`Other_Operating_Activities` DOUBLE NULL,
`Other_Operating_Items` DOUBLE NULL,
`Pre-Tax_Margin` DOUBLE NULL,
`Pre-Tax_ROE` DOUBLE NULL,
`Profit_Margin` DOUBLE NULL,
`Quick_Ratio` DOUBLE NULL,
`Research_and_Development` DOUBLE NULL,
`Retained_Earnings` DOUBLE NULL,
`Sale_and_Purchase_of_Stock` DOUBLE NULL,
`Sales_General_and_Admin` DOUBLE NULL,
`Short-Term_Debt`/_Current_Portion_of_Long-Term_Debt` DOUBLE NULL,
`Short-Term_Investments` DOUBLE NULL,
`Total_Assets` DOUBLE NULL,
`Total_Current_Assets` DOUBLE NULL,
`Total_Current_Liabilities` DOUBLE NULL,
`Total_Equity` DOUBLE NULL,
`Total_Liabilities` DOUBLE NULL,
`Total_Liabilities_&_Equity` DOUBLE NULL,
`Total_Revenue` DOUBLE NULL,
`Treasury_Stock` DOUBLE NULL,
`For_Year` DOUBLE NULL,
`Earnings_Per_Share` DOUBLE NULL,
`Estimated_Shares_Outstanding` DOUBLE NULL,
PRIMARY KEY (Ticker_Symbol, Period_Ending),
CONSTRAINT `Ticker_Symbol`
FOREIGN KEY (`Ticker_Symbol`)
REFERENCES `fe512db`.`securities` (`Ticker_symbol`)
ON DELETE NO ACTION
ON UPDATE NO ACTION);

```

```

* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.

```

Out[19]: []

Load data into the table of financial_statements

```

In [20]: %%sql
LOAD DATA INFILE '/home/data/financial_statements.csv'
INTO TABLE financial_statements fields
TERMINATED BY ','
IGNORE 1 LINES;

```

```

* mysql+pymysql://root:***@fe512_mysql/fe512db
1776 rows affected.

```

Out[20]: []

In [21]: %sql SELECT * FROM financial_statements LIMIT 5;

* mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.

Out[21]:

Ticker_Symbol	Period_Ending	Accounts_Payable	Accounts_Receivable	Add'l_income/expense_items	Af
AAL	2012-12-31	3068000000.0	-222000000.0	-1961000000.0	
AAL	2013-12-31	4975000000.0	-93000000.0	-2723000000.0	
AAL	2014-12-31	4668000000.0	-160000000.0	-150000000.0	
AAL	2015-12-31	5102000000.0	352000000.0	-708000000.0	
AAP	2012-12-29	2409453000.0	-89482000.0	600000.0	

Check the number of records and period in financial statements

In [22]: %sql SELECT COUNT(Ticker_Symbol),MIN(period_ending), MAX(period_ending) FROM finar

* mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.

Out[22]:

COUNT(Ticker_Symbol)	MIN(period_ending)	MAX(period_ending)
1776	2012-03-03	2016-12-31

Propose Questions and Answers

This section is mainly about two topics. The first part is value investing, and the second part is Dupont Analysis.

Part I Value Investing

We start our research with checking the database to see whether it meet our anticipation.

Q1.1 What are the top 10 companies which had the largest net assets at the end of 2016?

```
In [23]: %%sql
SELECT fs.Ticker_Symbol,s.Company_name,fs.Period_Ending,(fs.Total_Assets-fs.Total_
FROM financial_statements AS fs
INNER JOIN securities AS s
ON fs.Ticker_Symbol=s.Ticker_Symbol
WHERE Period_Ending LIKE "2016-12%"
ORDER BY Net_assests DESC
LIMIT 10;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
10 rows affected.
```

```
Out[23]:
```

Ticker_Symbol	Company_name	Period_Ending	Net_assests
INTC	Intel Corp.	2016-12-31	65899000000.0
FB	Facebook	2016-12-31	59194000000.0
GM	General Motors	2016-12-31	44000000000.0
CHTR	Charter Communications	2016-12-31	40000000000.0
UNH	United Health Group Inc.	2016-12-31	38464000000.0
CVS	CVS Health	2016-12-31	36830000000.0
KMI	Kinder Morgan	2016-12-31	34431000000.0
AMGN	Amgen Inc	2016-12-31	29875000000.0
F	Ford Motor	2016-12-31	29000000000.0
UTX	United Technologies	2016-12-31	27579000000.0

1.1 Interpretation

The net assets represent how much a company is worth on its book value. The top 10 companies with the largest net assets should be the big names and also have good operating performance. The output meets our anticipation. At the end of 2016, the top 10 largest companies include Intel, Facebook, and CVS, which are all very familiar to us. The company which had the largest net assets is Intel. Its net assets at the end of 2016 is \$65,899,000,000.00 .

Q1.2 What are the top 10 companies that had the highest market values at the end of 2016?


```
In [24]: %%sql
SELECT p.symbol,s.company_name,p.date,ROUND(p.close*fs.Estimated_Shares_Outstandir
FROM securities AS s
RIGHT JOIN financial_statements as fs ON s.ticker_symbol=fs.ticker_Symbol
RIGHT JOIN prices as p ON s.Ticker_symbol=p.symbol
WHERE p.date = "2016-12-30" AND fs.Period_Ending LIKE "2016%"
ORDER BY market_value DESC
LIMIT 10;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
10 rows affected.
```

```
Out[24]:
```

symbol	company_name	date	market_value
AAPL	Apple Inc.	2016-12-30	633708781565.15
MSFT	Microsoft Corp.	2016-12-30	492371561222.67
WMT	Wal-Mart Stores	2016-12-30	221757493512.01
DE	Deere & Co.	2016-12-30	172552370803.56
HD	Home Depot	2016-12-30	171177911316.24
DIS	The Walt Disney Company	2016-12-30	169918408193.89
CSCO	Cisco Systems	2016-12-30	152362710166.44
QCOM	QUALCOMM Inc.	2016-12-30	96866141277.67
NKE	Nike	2016-12-30	86480003112.06
SBUX	Starbucks Corp.	2016-12-30	81905081298.35

1.2 Interpretation

Market value is used to refer to the market capitalization of a publicly-traded company. Market value can be obtained by multiplying the number of its outstanding shares by the current share price. The output of Q1.2 denotes that the top 10 companies which had the highest market value at the end of 2016 are Apple Inc., Microsoft Corp., Wal-Mart Stores, Deere & Co., Home Depot, The Walt Disney Company, Cisco Systems, QUALCOMM Inc., Nike, and Starbucks Corp. This means they are valued the highest on the open market.

Here a problem comes out. The outputs of Q1.1 and Q1.2 have different companies. This means that there is a difference between how much the investors value a company and how much a company's book value is. Our objective is to find out the companies that are undervalued. Therefore, we keep exploring.

Q1.3 Which companies had the largest price-to-book ratio at the end of their fiscal year in 2016?

```
In [25]: %%sql DROP VIEW IF EXISTS MSQR;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

```
Out[25]: []
```

```
In [26]: %%sql
CREATE VIEW MSQR AS(
SELECT
s.ticker_symbol,s.company_name,s.GICS_sector,s.State,fs.Period_Ending,
ROUND(p.close*fs.Estimated_Shares_Outstanding,2) AS market_value,
(fs.Total_Assets-fs.Total_Liabilities) AS Net_assests,
ROUND((p.close*fs.Estimated_Shares_Outstanding)/(fs.Total_Assets-fs.Total_Liabili
FROM securities AS s
RIGHT JOIN financial_statements as fs ON s.ticker_symbol=fs.ticker_Symbol
RIGHT JOIN prices as p ON s.Ticker_symbol=p.symbol
WHERE fs.Period_Ending LIKE "2016%"
AND (p.date=fs.Period_Ending)
AND fs.Estimated_Shares_Outstanding !=0
AND (fs.Total_assets-fs.total_liabilities)>0
ORDER BY PB_ratio ASC
LIMIT 50);
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

```
Out[26]: []
```

In [27]: %sql SELECT * FROM MSQR;

* mysql+pymysql://root:***@fe512_mysql/fe512db
46 rows affected.

Out[27]:

ticker_symbol	company_name	GICS_sector	State	Period_Ending	market_value	1
NWSA	News Corp. Class A	Consumer Discretionary	New York	2016-06-30	6772166894.66	115
WDC	Western Digital	Information Technology	California	2016-07-01	11134396332.27	111
HPE	Hewlett Packard Enterprise	Information Technology	California	2016-10-31	38601993394.4	314
WRK	WestRock Company	Materials	Virginia	2016-09-30	12116892191.54	97
MU	Micron Technology	Information Technology	Idaho	2016-09-01	17009777150.17	120
JEC	Jacobs Engineering Group	Industrials	California	2016-09-30	6220083779.62	42
HP	Helmerich & Payne	Energy	Oklahoma[5]	2016-09-30	7082452911.26	45
DHI	D. R. Horton	Consumer Discretionary	Texas	2016-09-30	11199272249.95	67
NTAP	NetApp	Information Technology	California	2016-04-29	6940461359.51	28
TEL	TE Connectivity Ltd.	Information Technology	Switzerland	2016-09-30	23559091890.02	84
COO	The Cooper Companies	Health Care	California	2016-10-31	8534574662.86	26
PH	Parker-Hannifin	Industrials	Ohio	2016-06-30	14627359806.28	45
KMX	Carmax Inc	Consumer Discretionary	Virginia	2016-02-29	9394064572.5	29
FDX	FedEx Corporation	Industrials	Tennessee	2016-05-31	45560759065.9	137
HRS	Harris Corporation	Information Technology	Florida	2016-07-01	10252551266.68	30
SYMC	Symantec Corp.	Information Technology	California	2016-04-01	12346113105.37	36
VIAB	Viacom Inc.	Consumer Discretionary	New York	2016-09-30	15093057247.2	42
GPN	Global Payments Inc	Information Technology	Georgia	2016-05-31	10295479123.54	27
CAH	Cardinal Health Inc.	Health Care	Ohio	2016-06-30	25532172718.95	65
MCK	McKesson Corp.	Health Care	California	2016-03-31	36157892056.45	89
SWKS	Skyworks Solutions	Information Technology	Massachusetts	2016-09-30	14378468192.6	35
DG	Dollar General	Consumer Discretionary	Tennessee	2016-01-29	22083561098.98	53

APD	Air Products & Chemicals Inc	Materials	Pennsylvania	2016-09-30	30058284952.16	7C
STX	Seagate Technology	Information Technology	Ireland	2016-07-01	7180048397.39	15
MCHP	Microchip Technology	Information Technology	Arizona	2016-03-31	9825888458.69	21
EMR	Emerson Electric Company	Industrials	Missouri	2016-09-30	35851635260.71	75
BDX	Becton Dickinson	Health Care	New Jersey	2016-09-30	38217097134.81	76
COL	Rockwell Collins	Industrials	Iowa	2016-09-30	11003497728.35	2C
VAR	Varian Medical Systems	Health Care	California	2016-09-30	9488369314.46	17
MSFT	Microsoft Corp.	Information Technology	Washington	2016-06-30	405449825131.5	72C
CTAS	Cintas Corporation	Industrials	Ohio	2016-05-31	10435825099.51	18
EA	Electronic Arts	Information Technology	California	2016-03-31	20488783027.97	33
FFIV	F5 Networks	Information Technology	Washington	2016-09-30	8397820806.27	11
AYI	Acuity Brands Inc	Industrials	Georgia	2016-08-31	11994736818.73	16
NKE	Nike	Consumer Discretionary	Oregon	2016-05-31	93948961349.37	122
ROK	Rockwell Automation Inc.	Industrials	Wisconsin	2016-09-30	15941338447.89	19
ABC	AmerisourceBergen Corp	Health Care	Pennsylvania	2016-09-30	17139391218.92	21
LOW	Lowe's Cos.	Consumer Discretionary	North Carolina	2016-01-29	66830171913.4	76
RHT	Red Hat Inc.	Information Technology	North Carolina	2016-02-29	11952754535.81	13
EL	Estee Lauder Cos.	Consumer Staples	New York	2016-06-30	33704614037.3	35
PAYX	Paychex Inc.	Information Technology	New York	2016-05-31	19539855679.05	19
MON	Monsanto Co.	Materials	Missouri	2016-08-31	47113907289.0	45
KLAC	KLA-Tencor Corp.	Information Technology	California	2016-06-30	11415688383.4	6
DE	Deere & Co.	Industrials	Illinois	2016-10-31	147868543606.02	65
CLX	The Clorox Company	Consumer Staples	California	2016-06-30	17899544836.04	2
CSRA	CSRA Inc.	Information Technology	Virginia	2016-04-01	5197561087.35	

1.3 Interpretation

We used the price-to-book ratio to compare a firm's market to book value by dividing price per

share by book value per share. If a market value is equal to its book value, then the P/B ratio should be one. A lower P/B ratio could mean the stock is undervalued. The output shows the companies that are most undervalued. The period ending means the date that a company issued its latest annual financial statements. In the where clause, we set the constraint that the estimated outstanding shares are not zero because we found that the companies which showed on the outputs of top companies when we first wrote the query without that constraint were all zero. This may be caused by missing values of the raw data or because the company just came to the market so that it's hard to estimate the outstanding shares. Another constraint we should pay attention to is that total assets minus total liabilities greater than zero. This constraint is necessary because the lowest PB_ratio would be a negative value if we don't set this constraint, which is not what we expected.

Q1.4 Which companies had the lowest price-earnings ratio while also had the good price-book ratio at the end of 2016?

```
In [28]: %%sql
SELECT
MSQR.ticker_symbol,MSQR.company_name,MSQR.GICS_sector,MSQR.State,
ROUND(MSQR.market_value/fs.Estimated_Shares_Outstanding,2) AS Market_value_per_share,
ROUND((MSQR.market_value/fs.Estimated_Shares_Outstanding)/fs.Earnings_Per_Share,2) AS PB_ratio
FROM MSQR
LEFT JOIN financial_statements AS fs ON MSQR.ticker_symbol=fs.ticker_symbol
WHERE fs.Period_Ending LIKE "2016%"
AND fs.Estimated_Shares_Outstanding !=0
AND fs.Earnings_Per_Share >0
AND MSQR.PB_ratio<=2
ORDER BY PE_ratio;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

```
Out[28]:
```

ticker_symbol	company_name	GICS_sector	State	Market_value_per_share	Earnings_Per_Share
HPE	Hewlett Packard Enterprise	Information Technology	California	22.47	1.84
DHI	D. R. Horton	Consumer Discretionary	Texas	30.2	2.39
JEC	Jacobs Engineering Group	Industrials	California	51.72	1.75
NWSA	News Corp. Class A	Consumer Discretionary	New York	11.35	0.3
WDC	Western Digital	Information Technology	California	46.47	1.01

1.4 Interpretation

The price-earnings ratio (P/E ratio) is the ratio for valuing a company that measures its current share price relative to its per-share earnings. In essence, the price-earnings ratio indicates the

dollar amount an investor can expect to invest in a company in order to receive one dollar of that company's earnings. Typically, if the companies pay dividends at the same level, then for the investors, a lower P/E ratio is better. Therefore, we calculated the P/E ratio to find the potentially undervalued stocks.

The outputs just have five companies because we set the constraint "PB ratio no more than 2" in the where clause. This constraint came from the methodology of Benjamin Graham, the father of value investing, which is called the margin of safety. The margin of safety is the principle of buying a security at a significant discount to its intrinsic value, which is thought to not only provide high-return opportunities, but also to minimize the downside risk of an investment. Usually, the P/B ratio varies in different sectors, but a P/B ratio no more than two is a safe number that allows us to buy an equity at a big enough discount to leave some room for error in the estimation of value.

Similar to Q1.3, we set another two constraints "outstanding estimated shares!=0" and "earnings per share>0" because with these two constraints the lowest P/E ratio will be negative, which will cause us to make the opposite conclusion.

There is no exact rule for what is a good P/E ratio, but in practice, the average P/E ratio in the whole market is usually between 20 to 25, for a healthy market. Therefore, we also applied the methodology of the margin of safety to our selection standard. From the output, we can see only two companies, Hewlett Packard Enterprise and D. R. Horton have P/E ratio no more than 20. Therefore, the final decision about which security we should buy is to buy HPE and DHI.

Q1.5 Assume we had invested the two securities we screened, HPE and DHI, at the earliest. What is the ratio of return on our capital if we hold these two stocks at the end of 2016?

```
In [29]: %%sql
DROP VIEW IF EXISTS HPE_earlier;
DROP VIEW IF EXISTS HPE_later;
DROP VIEW IF EXISTS DHI_earlier;
DROP VIEW IF EXISTS DHI_later;

* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
```

Out[29]: []

```
In [30]: %%sql
CREATE VIEW HPE_earlier AS
(SELECT symbol,date AS earlier_date,close AS earlier_close FROM prices
WHERE (symbol,date) IN
(SELECT symbol,MIN(date) FROM prices WHERE symbol= 'HPE'));

CREATE VIEW HPE_later AS
(SELECT symbol,date AS later_date,close AS later_close FROM prices
WHERE (symbol,date) IN
(SELECT symbol,MAX(date) FROM prices WHERE symbol= 'HPE'));

CREATE VIEW DHI_earlier AS
(SELECT symbol,date AS earlier_date,close AS earlier_close FROM prices
WHERE (symbol,date) IN
(SELECT symbol,MIN(date) FROM prices WHERE symbol= 'DHI'));

CREATE VIEW DHI_later AS
(SELECT symbol,date AS later_date,close AS later_close FROM prices
WHERE (symbol,date) IN
(SELECT symbol,MAX(date) FROM prices WHERE symbol= 'DHI'));

* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
```

```
Out[30]: []
```

```
In [31]: %%sql
SELECT symbol,earlier_date,earlier_close,later_date,later_close,
ROUND((later_close-earlier_close),2) AS dif_close, DATEDIFF(later_date,earlier_date) AS dif_date,
ROUND(((later_close-earlier_close)/earlier_close)/DATEDIFF(later_date,earlier_date)) AS HPE_JEC
FROM (
    (SELECT HPE_earlier.symbol,HPE_earlier.earlier_date,HPE_earlier.earlier_close,HPE_later.later_date,HPE_later.later_close FROM HPE_earlier
    INNER JOIN HPE_later
    ON HPE_earlier.symbol=HPE_later.symbol)
    UNION
    (SELECT DHI_earlier.symbol,DHI_earlier.earlier_date,DHI_earlier.earlier_close,DHI_later.later_date,DHI_later.later_close FROM DHI_earlier
    INNER JOIN DHI_later
    ON DHI_earlier.symbol=DHI_later.symbol)
) AS HPE_JEC
ORDER BY annual_rate_of_increase DESC;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
2 rows affected.
```

```
Out[31]:
```

symbol	earlier_date	earlier_close	later_date	later_close	dif_close	dif_date	annual_rate_of_increase
HPE	2015-10-19	16.55	2016-12-30	23.14	6.59	438	0.33
DHI	2010-01-04	11.16	2016-12-30	27.33	16.17	2552	0.21

1.5 Interpretation

The return on capital is equal to the profit we gained divided by our invested capital. In order to simplify the calculation, we assume the stocks have no dividends. Then, the profit we gained during our holding period is mainly from the fluctuation of stock price, the latest close price minus the price when we invested it.

In the output, the earlier date is the earliest date that we could invest in a stock, and the later date is the latest date we hold that stock, the end of 2016. The earlier date of HPE is not same as DHI, but just Oct 19th, 2015 because the HPE company had become a listed company until 2015. The earlier_close means the original price when we bought a stock, and the later_close is the close price at the end of 2016. The dif_close means the difference between the latest close price and the original price. Since we hold different stocks for different periods, is not comparable for the return of each stock, so we use the ratio of return instead of the absolute return value. The dif_date means how long we hold a stock. Then, the ratio of return of each stock could be calculated by dividing the price appreciation by the holding period. The result is the daily ratio of return. After that, we multiply this rate by 365 and get the annual rate of return.

From the output, we can see that the annual rate of return for HPE is 33%, and the rate of 21% for DHI. These two numbers are both very high comparing to the rate of return on the whole market, which proved that we made a good choice.

Part II Analyzing S&P 500 in the view of sectors

Q2.1 How many sectors covered in S&P 500?

Check if missing records existed in sector column?

```
In [32]: %sql SELECT COUNT(Ticker_symbol) FROM securities WHERE GICS_Sector = "";
%sql SELECT COUNT(Ticker_symbol) FROM securities WHERE GICS_Sector IS NULL;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
* mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

```
Out[32]: COUNT(Ticker_symbol)
0
```

distinctly count sectors

```
In [33]: %sql SELECT DISTINCT GICS_Sector FROM securities;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
11 rows affected.
```

```
Out[33]: GICS_Sector
Health Care
Industrials
Consumer Discretionary
Information Technology
Consumer Staples
Utilities
Financials
Real Estate
Materials
Energy
Telecommunications Services
```

2.1 Interpretation:

From the output above, 11 sectors covered in S&P 500.

Q2.2 Background

DuPont Analysis

The Dupont analysis also called the Dupont model is a financial ratio based on the return on equity ratio that is used to analyze a company's ability to increase its return on equity. The Dupont analysis looks at three main components of the ROE ratio: Profit Margin Total Asset Turnover Financial Leverage

$$\text{ROE} = \text{Profit Margin} * \text{Total Assets Turnover} * \text{Financial Leverage}$$
$$\text{ROE} = \frac{\text{Net Income}}{\text{Total Equity}} = \left(\frac{\text{Net Income}}{\text{Net Sales}} \right) * \left(\frac{\text{Net Sales}}{\text{Average total Assets}} \right) * \left(\frac{\text{Total Assets}}{\text{Total Equity}} \right)$$

Profit Margin measures the amount of net income earned with each dollar of sales generated by comparing the net income and net sales of a company. That is, this ration can tell us how effectively a company can convert sales into net income.

Total Assets Turnover shows how efficiently a company can use its assets to generate sales.

Financial Leverage shows the level of debt financing is used to acquire assets and maintain operations.

Q2.2 How are we going to use the DuPont Analysis?

Many companies in different industries are always trying to exploring the right strategy they can exploit. What if they were not sure their strategies are appropriate. Questions come like " Do we run our company efficiently?" and "Do we need to lower our company's financial leverage to reduce financial risk?" For these business owners, it is a good way to examine successful corporate strategy examples, which can help them identify methods to target within their own organizations. S&P 500 covers 11 sectors and each company listed in S&P 500 can be considered as the successful benchmark in a certain sector.

In this part, we are going to derive some information from all the financial statements filed by most of S&P 500 companies and their general description to offer some guidelines to other companies from the view of DuPont Analysis.

Choose the year when most companies' reports had been collected

```
In [34]: %%sql (SELECT YEAR(Period_Ending) AS Year, COUNT(Ticker_Symbol) FROM financial_statements
          UNION
          (SELECT YEAR(Period_Ending) AS Year, COUNT(Ticker_Symbol) FROM financial_statements)
          UNION
          (SELECT YEAR(Period_Ending) AS Year, COUNT(Ticker_Symbol) FROM financial_statements)
          UNION
          (SELECT YEAR(Period_Ending) AS Year, COUNT(Ticker_Symbol) FROM financial_statements)
          UNION
          (SELECT YEAR(Period_Ending) AS Year, COUNT(Ticker_Symbol) FROM financial_statements)
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

```
Out[34]:
```

Year	COUNT(Ticker_Symbol)
2016	220
2015	445
2014	436
2013	439
2012	236

The query above shows how many reports had been collected in different years in our database. From this output, we decided to perform the DuPont analysis based on the year 2015 since we could analyze the most companies-445.

Use table"financial_statements" to calculate financial ratios which will be used in DuPont Analysis

```
In [35]: %%sql DROP TEMPORARY TABLE IF EXISTS ROE_SUM;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

```
Out[35]: []
```

```
In [36]: %%sql
CREATE TEMPORARY TABLE ROE_SUM AS
(SELECT Ticker_Symbol,Period_Ending,(Profit_Margin/100) AS Profit_Margin,
ROUND(ABS((Net_Income/(Profit_Margin/100))/Total_Assets),2) AS Total_Assets_Turnov
ROUND(ABS(Total_Assets/Total_Equity ),2) AS Financial_Leverage,
ABS(round((Profit_Margin*((Net_Income/(Profit_Margin/100))/Total_Assets)* (Total_#
(After_Tax_ROE/100) AS After_Tax_ROE
FROM financial_statements
ORDER BY After_Tax_ROE DESC);
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
1776 rows affected.
```

```
Out[36]: []
```

```
In [37]: %%sql SELECT * FROM ROE_SUM LIMIT 1;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

```
Out[37]:
```

Ticker_Symbol	Period_Ending	Profit_Margin	Total_Assets_Turnover	Financial_Leverage	roe	Aft
LB	2015-01-31	0.09	1.55	415.33	57.89	

Through queries above, we calculated each companies ROE and the three components including Profit Margin, Total Assets Turnover, and Financial Leverage. We created a temporary table "ROE_SUM" to summarize our calculated results.

Combine temporary table "ROE_SUM" with base table "securities"

```
In [38]: %%sql DROP TABLE IF EXISTS ROE_SUM1;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

```
Out[38]: []
```

```
In [39]: %%sql
CREATE TABLE ROE_SUM1 AS
(SELECT ROE_SUM.Ticker_Symbol, securities.Company_name AS name, securities.GICS_Sc
ROE_SUM.Period_Ending, ROE_SUM.Profit_Margin, ROE_SUM.Total_Assets_Turnover,
ROE_SUM.Financial_Leverage,ROE_SUM.After_Tax_ROE
FROM ROE_SUM
INNER JOIN securities
ON securities.Ticker_symbol= ROE_SUM.Ticker_Symbol
ORDER BY After_Tax_ROE DESC);
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
1776 rows affected.
```

```
Out[39]: []
```

```
In [40]: %sql SELECT * FROM ROE_SUM1 LIMIT 1;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

```
Out[40]:
```

Ticker_Symbol	name	sector	Period_Ending	Profit_Margin	Total_Assets_Turnover	Financial
LB	Brands Inc.	Consumer Discretionary	2015-01-31	0.09	1.55	

Financial ratios vary a lot depending on sectors. Different companies in different sectors couldn't use the same standard. Because the operational risk companies in different sectors face and financial risk they undertake are significantly different. Hence, we used queries above to combine their financial ratios with their sector information and created the table "ROE_SUM1" based on this combination.

Calculate the financial ratios by sectors

```
In [41]: %%sql SELECT ROE_SUM1.sector, YEAR(ROE_SUM1.Period_Ending) AS Year,
          ROUND(AVG(ROE_SUM1.Profit_Margin),2) AS AVG_PM,
          ROUND(AVG(ROE_SUM1.Total_Assets_Turnover),2) AS AVG_TAT,
          ROUND(AVG(ROE_SUM1.Financial_Leverage),2) AS AVG_FL,
          ROUND(AVG(ROE_SUM1.After_Tax_ROE),2) AS AVG_ROE
FROM      ROE_SUM1 WHERE YEAR(ROE_SUM1.Period_Ending)=2015
AND      ROE_SUM1.After_Tax_ROE < 5
GROUP BY ROE_SUM1.sector
ORDER BY AVG_ROE DESC;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
11 rows affected.
```

```
Out[41]:
```

	sector	Year	AVG_PM	AVG_TAT	AVG_FL	AVG_ROE
	Consumer Staples	2015	0.09	1.37	5.72	0.6
	Energy	2015	0.42	0.54	4.17	0.41
	Industrials	2015	0.1	0.99	4.82	0.37
	Telecommunications Services	2015	0.15	0.36	5.76	0.33
	Materials	2015	0.12	0.74	4.06	0.31
	Consumer Discretionary	2015	0.09	1.17	4.31	0.28
	Health Care	2015	0.12	0.85	4.27	0.26
	Information Technology	2015	0.17	0.6	2.73	0.23
	Financials	2015	0.16	0.21	8.18	0.16
	Real Estate	2015	0.28	0.21	3.21	0.12
	Utilities	2015	0.11	0.26	3.78	0.1

```
In [ ]:
```

Q2.2 Interpretation:

In queries above, we calculated the average profit margin, average total assets turnover, average financial leverage and average return on equity by sectors in 2015. We should note one thing that we excluded companies whose ROE exceeded 500. The reason for excluding these companies is that their fairly high ROE came for a special reason and other companies are unlikely to refer to their financial ratios. We will continue to discuss these outliers in later queries.

So far, the query results can be presented to companies in different sectors as one of their references to set their goals or strategies. For example, a company in Health Care expects to achieve ROE about 0.26 which means that every dollar of common stockholders' equity generates 0.22 dollar of net income, then it's advisable for this company to adopt the financial leverage no more than 4.3 and be more effective at using company's asset to create added values, say profit margin should be 12% and total assets turnover be 0.85.

Q2.2 Extension: Further check the companies with exceptional performances

```
In [42]: %%sql SELECT ROE_SUM1.sector, YEAR(ROE_SUM1.Period_Ending) AS Year,
          ROUND(AVG(ROE_SUM1.Profit_Margin),2) AS AVG_PM,
          ROUND(AVG(ROE_SUM1.Total_Assets_Turnover),2) AS AVG_TAT,
          ROUND(AVG(ROE_SUM1.Financial_Leverage),2) AS AVG_FL,
          ROUND(AVG(ROE_SUM1.After_Tax_ROE),2) AS AVG_ROE
FROM      ROE_SUM1 WHERE YEAR(ROE_SUM1.Period_Ending)=2015
AND      ROE_SUM1.After_Tax_ROE >= 5
GROUP BY ROE_SUM1.sector
ORDER BY AVG_ROE DESC;
```

* mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.

```
Out[42]:
```

	sector	Year	AVG_PM	AVG_TAT	AVG_FL	AVG_ROE
	Consumer Discretionary	2015	0.06	0.89	635.02	31.89
	Energy	2015	2.42	0.54	7.73	8.02
	Industrials	2015	0.07	0.97	88.4	6.01
	Financials	2015	0.22	0.64	42.18	5.96
	Consumer Staples	2015	0.05	1.37	85.3	5.82

```
In [43]: %%sql
SELECT * FROM ROE_SUM1
WHERE YEAR(ROE_SUM1.Period_Ending)=2015
AND ROE_SUM1.After_Tax_ROE >= 5
ORDER BY After_Tax_ROE DESC;
```

* mysql+pymysql://root:***@fe512_mysql/fe512db
7 rows affected.

```
Out[43]:
```

	Ticker_Symbol	name	sector	Period_Ending	Profit_Margin	Total_Assets_Turnover
	LB	L Brands Inc.	Consumer Discretionary	2015-01-31	0.09	1.55
	APA	Apache Corporation	Energy	2015-12-31	3.69	0.34
	CHK	Chesapeake Energy	Energy	2015-12-31	1.15	0.74
	ALLE	Allegion	Industrials	2015-12-31	0.07	0.97
	SPGI	S&P Global, Inc.	Financials	2015-12-31	0.22	0.64
	CHTR	Charter Communications	Consumer Discretionary	2015-12-31	0.03	0.23
	KMB	Kimberly-Clark	Consumer Staples	2015-12-31	0.05	1.37

Q2.2 Extension interpretation

We further checked the outliers via queries screening out all the companies with ROE more than 5. As the output shown, there were 7 seven companies with fairly high ROE because of high financial leverage used or abnormal profit margin created in 2015. If we involved these seven companies to calculate the average standard and offer advice, our advice would be misleading. These companies' exceptional ratios of profit margin and financial leverage represented high operational risk and financial risk. On the other hand, these exceptional ratios also mean exceptional returns. So, these companies might be the first choice for speculators.

Q2.3 Check if there is a coincidence between the financial performance and market trends

First, we found the company with the best financial performance from 2013 to 2016

```
In [44]: %%sql DROP TABLE IF EXISTS 2016report;
          DROP TABLE IF EXISTS 2013report;
          CREATE TABLE 2016report AS(SELECT * FROM financial_statements WHERE YEAR(Per
          CREATE TABLE 2013report AS(SELECT * FROM financial_statements WHERE YEAR(Per
          SELECT * FROM 2016report LIMIT 1;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
0 rows affected.
220 rows affected.
439 rows affected.
1 rows affected.
```

Out[44]:

Ticker_Symbol	Period_Ending	Accounts_Payable	Accounts_Receivable	Add'l_income/expense_items	Af
AAP	2016-01-02	3757085000.0	-21476000.0	-7484000.0	


```
In [45]: %%sql SELECT 2016report.Ticker_Symbol, ((2016report.Net_Cash_Flow-2013report.Net_Cash_Flow)/2013report.Net_Cash_Flow)*100 AS NCFchange_billion,
ROUND(((2016report.Net_Income_Applicable_to_Common_Shareholders-2013report.Net_Income_Applicable_to_Common_Shareholders)/2013report.Net_Income_Applicable_to_Common_Shareholders),2) AS percentII
FROM 2016report INNER JOIN 2013report ON 2016report.Ticker_Symbol=2013report.Ticker_Symbol
ORDER BY NCFchange_billion DESC LIMIT 5;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

```
Out[45]: Ticker_Symbol  NCFchange_billion  percentII  NIncome_billion  percentII
          FB              3.057          3.26          8.697          5.83
          AAPL            -4.149         -1.18          8.65           0.23
          GM              -3.771         -2.36          5.657           1.5
          EBAY             2.307         -0.99          4.41           1.54
          CHTR             1.516        108.29          3.691        -21.84
```

The query above indicated that FB(Facebook) had the best financial performance. In the next step, we looked at the stock market and spotted out which company's close price experienced the fastest growing.

```
In [46]: %%sql DROP VIEW IF EXISTS 16mar;
          DROP VIEW IF EXISTS 13mar;

          CREATE VIEW 13mar AS (SELECT symbol, SUM(volume) AS SUM_volume13, ROUND(AVG(
          GROUP BY symbol));

          CREATE VIEW 16mar AS (SELECT symbol, SUM(volume) AS SUM_volume16, ROUND(AVG(
          GROUP BY symbol));

          SELECT * FROM 16mar LIMIT 5;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
5 rows affected.
```

```
Out[46]:
```

symbol	SUM_volume16	AVG_close16
A	524927900	43.37
AAL	2309784700	38.18
AAP	260668100	155.49
AAPL	9680671300	104.6
ABBV	2019860600	60.67

```
In [47]: %%sql SELECT 16mar.symbol, 16mar.SUM_volume16, 13mar.SUM_volume13,
          (16mar.SUM_volume16-13mar.SUM_volume13)/13mar.SUM_volume13 AS volume_dif,
          16mar.AVG_close16, 13mar.AVG_close13,
          (16mar.AVG_close16-13mar.AVG_close13)/13mar.AVG_close13 AS AvgClose_dif
          FROM 16mar, 13mar
          WHERE 16mar.symbol = 13mar.symbol
          ORDER BY AvgClose_dif DESC LIMIT 5;
```

```
* mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

```
Out[47]:
```

symbol	SUM_volume16	SUM_volume13	volume_dif	AVG_close16	AVG_close13	AvgClose_dif
AVGO	807593700	650915300	0.2407	157.19	38.54	3.07862
NVDA	2788055600	2228684400	0.2510	53.76	14.19	2.788584
EA	906326300	1201599300	-0.2457	73.75	21.82	2.379927
FB	6414357000	15143182600	-0.5764	117.04	35.48	2.29876
SWKS	795435200	839161400	-0.0521	70.63	23.68	1.982686

Q2.3 interpretation

This results showed us that Facebook's average close price increased by 229% and its volume decreased by 57%. Even though in the growth of close price term, Facebook didn't reach the first

place in the stock market from 2013 to 2016, it still remains at the top five with its outstanding financial performance. There is no coincidence. Markets can greatly reflect the companies performance.

Conclusion & Future work

Conclusion:

1. We should invest the two most undervalued companies, Hewlett Packard Enterprise and D. R. Horton.
2. We would get at most 33% and 21% if we invested HPE and DHI as earlier as possible since 2010.
3. Companies can improve their strategies by referring to the industrial standard generated by DuPont analysis and finally raise their returns on equity.
4. Facebook had the best financial performance from 2013 to 2016.

Future Work

In this project, we have a lot of limitations when we tried to use some models and formulas to analyze the stocks. This is mainly caused by the lack of data in our table.

On one hand, some models and formulas need some features that do not exist in our database. For example, when we try to build the capital asset pricing model (CAPM) to evaluate the securities' returns and risks more technically and accurately, we could not achieve this because our database did not have the risk-free rate and other key factors.

On the other hand, the existed features in our database do not have complete data. For instance, when I using the DuPont Analysis on Q2, we chose to perform it on 2015 but not on 2016 just because the companies in 2016 are far less than the companies in 2015.

In the future, we will directly get the raw data from original website, not from Kaggle. This method can allow us get more abundant data that we want, and then we will be able to perform further analysis.

References

Entity–relationship model, (https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model)
Benjamin Graham's Timeless Investment Principles,
(<https://www.investopedia.com/articles/basics/07/grahamprinciples.asp>)