

Identifying Star Investors

Executive Summary

My project is designed to assist me with data collection for a research article that I am currently working on. The intent of the article is to determine how much, if any, benefit is gained by acquisitions, through interaction with a star investor. The theory in principle is that a company is more likely to be successful if it is acquired by a star investor than if some unknown company had acquired it. This Information Systems project is designed to help me quickly gather data on multiple companies and assess whether or not they qualify as star investors.

I will identify thirty companies to be designated as star investors and used in my study. I begin with Forbes list of the top 100 businesses in the United States as these are all companies that are well known and thus carry a strong brand name. Using those company names I proceed to use web queries to retrieve the company's stock ticker symbols, as well as measures of the financial performance of each company. I use the following three measures to assess financial performance and the ability to be a star investor:

1. Total Cash - represents the company's ability to invest
2. Operating Margin - represents the company's profitability
3. Return on Assets (ROA) - represents managerial effectiveness

These are arbitrary measures that may change in the future if I deem other measures to be more indicative of star investors, but regardless, for the purposes of this project a framework will be in place that allows for such changes in the future. I next manipulate the data and rank the companies in regards to their performance in each area. An aggregate of these rankings is used to determine the thirty star investors. An interface is then developed that allows the user to interact with the data and compare the results from each company. Initially, I thought it would be necessary to calculate measures such as ROA but as they were available, I deemed it redundant to repeat the calculation.

This project will draw deeply on principles learned this semester such as the use of arrays, loops, range manipulation, and web queries among others.

Implementation

I will discuss the implementation of this project through a discussion of the VBA code that powers the spreadsheet and the interface with the user through the workbook. First, the user interface.

Description of Spreadsheets:

The first sheet of the workbook is titled “Forbes Top” and contains most of the tools that a user will need to access the data. There is a dropdown box at the top of the sheet from which the user can choose any of 100 companies in the project. Once a company is selected, the relevant information about the company is displayed alongside a column of descriptions. Additional companies may be added and viewed side by side. Another button next to the dropdown box allows the user to clear the companies being viewed and start over. Underneath the dropdown box is another button labeled “show top 30” that when pressed displays the top thirty star investors that the program determined.

The second sheet is titled “Table” and contains the complete dataset that is accessed through the dropdown box on the “Forbes Top” sheet. Next to the table containing the dataset is a button that is titled “Update Table” which allows the user to update the information in the table. This will probably not be necessary often as the data retrieved is all updated on the website at most monthly. In addition, this command has a lag of about five minutes until completion due to the multiple web queries being run.

Finally, the third and fourth sheets are used in running the program and are not meant to interact with the user. The third sheet is titled “Web Query” and is where the data downloaded through the web query is placed so that it can be searched through. The fourth sheet is titled “Data Manipulation” and is where the data is stored while it is being changed to the format that is seen on the “Table” sheet.

Description of VBA Code:

As I wrote the code, I divided it conceptually into different subs to allow me to better visualize the process. I will review the code sub by sub as I feel this represents the most logical progression.

1. **Sub fillTable** - This is the main sub and I use it to call all of the other subs that relate to the process of gathering the data and formatting it to fit on the sheet “Table”. This sub calls subs 2-9 which will be described below. This sub is connected to the Update Table button.
2. **Sub fillCompName** - This sub uses a For Next loop to populate an array named compName() from the list of 100 company names which I pulled from Forbes.com. The original list of companies is found in a range on the “Data Manipulation” sheet.

```
For n = 1 To 100
    compName(n) = .Offset(n, 0).Value
Next
```
3. **Sub getTckrSymbol** – This sub uses a web query to pull ticker symbols from a website. Due to complications I was unable to use this sub to accurately retrieve ticker symbols. Further description of the work done on this sub will appear in the section on conceptual difficulties as this sub does not relate to the implementation of the project. I manually pulled the ticker symbols offline and placed them in a range on the “Data Manipulation” sheet.

- 4. Sub fillTicker** – This sub is almost identical to the fillCompName sub above. It uses a For Next loop to populate an array named ticker() from the range that contains the ticker symbols on the “Data Manipulation” sheet.
- 5. Sub getData** – This sub uses a web query to retrieve data from finance.yahoo.com and pastes the data to the “Web Query” sheet. The web query is within another For Next loop which allow the process to repeat for each ticker symbol in the array ticker(). The code used to connect to the website is shown below:

```
.Connection = "URL;http://finance.yahoo.com/q/ks?s=" & ticker(n) & "+Key+Statistics"
```

The data required was located on a link named “key statistics” found on the main page returned by the initial search. This required me to concatenate the link and the array ticker() onto the end of the URL. Once the web query brought the data to the worksheet, I used the find method to search for the financial measures I needed as seen in the code below:

```
Cells.Find(what:="Operating Margin (ttm)", After:=ActiveCell, LookIn:=xlFormulas, _  
LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:=xlNext, _  
MatchCase:=False, SearchFormat:=False).Activate  
ActiveCell.Offset(0, 1).Select  
opMargin(n) = ActiveCell.Value  
Sheets("Data Manipulation").Cells(n + 1, 4).Value = opMargin(n)  
Range("OperatingMarginTable").Cells(n).Value = opMargin(n)
```

The value for Operating Margin was then entered into the array opMargin() and from there sent to the finished table on the “Table” sheet and to a range on the “Data Manipulation” page. The reason for sending the data to multiple locations is that the data had to be collected as strings because it contained symbols as well as numbers. For example, Total Cash was returned in billions and millions with the amount designated by an “M” or “B”. I wanted the data presented formally like this on the table, but also needed to change the data into integers in order to perform calculations. The find method shown above was repeated for Total Cash and Return on Assets.

- 6. Sub cleanData** – This sub was used to prepare data to be used in mathematical calculations. As stated above, the data was collected as string and needed to be transformed into integers. This was complicated by the fact that Total Cash was not expressed consistently but shown as either millions or billions as designated by a “M” or “B”. To change the data, I used an if statement that determined whether the string contained an “M” or “B” using the Instr string function. If an “M” was found, the period in the string was removed using the replace method and a new

period was concatenated onto the beginning of the string, effectively changing the number to the billions place.

```
If InStr(Range("TotalCash").Cells(n).Value, "M") <> 0 Then  
  
Range("TotalCash").Cells(n).Select  
  
Selection.Replace what:=".", Replacement:="", LookAt:=xlPart, _  
SearchOrder:=xlByRows, MatchCase:=False, SearchFormat:=False, _  
ReplaceFormat:=False  
  
Range("TotalCash").Cells(n).Value = "." & Range("TotalCash").Cells(n).Value
```

This code was again placed in a For Next loop so that it could be repeated throughout the data set. Finally all of the “M’s” and “B’s” in the data set were removed using the replace method similar to what is shown above.

7. **Sub sortData** – This sub sorts the data according to the values of the financial measures for each company. This is done to find out which company is performing the best in each area. The range containing the values was placed in an array in with the original list of companies. The array was the sorted in descending value according to the values of the financial measures. This effectively rearranged the list of company names into the order dictated by their performance.

```
Range("CashOrder").Sort key1:=Range("K2"), Order1:=xlDescending, Header:=xlYes
```

This process was repeated for each financial measure. The lists of companies are replicated anew each time sub runs in order to avoid mismatching the values and the company names. This was all placed within a For Next loop so that it could be replicated across the values of the dataset.

8. **Sub getRankOrder** – This sub retrieves the rank order value corresponding to each company’s position within the range. This was done by using a find method and the array compName() to find the location of each company within the newly ordered range. The offset property then allowed me to find the ranking corresponding with each company. The order rank values were then returned the table on the “Table” sheet. This process was placed within a For Next loop and repeated for each financial measure. The code used is shown below.

```
Range("CashOrder").Select  
  
Selection.Find(what:=compName(n), After:=ActiveCell, LookIn:=xlFormulas, _  
LookAt:=xlPart, SearchOrder:=xlByRows, SearchDirection:=xlNext, _  
MatchCase:=False, SearchFormat:=False).Activate
```

```
ActiveCell.Offset(0, -1).Select
```

```
Range("TCRank").Cells(n).Value = ActiveCell.Value
```

- 9. Sub starOrder** – This sub calculates an average ranking for each company from the rankings on financial measures already attained and once again sorts the companies according to these values. Finally, the find method and the offset property are again used to discover the rank order value for each company and these values are returned to the finished table on the “Table” sheet. Once again a For Next loop is used to move through the ranges involved. The code used to find the average rank is shown below.

```
Range("SortedStar").Cells(n).Value = (Range("TCRank").Cells(n).Value + _  
Range("OMRank").Cells(n).Value + Range("ROARank").Cells(n).Value) / 3
```

- 10. Sub showData** – This sub allows the user to interface with the data through the use of a dropdown box. First, a new column is inserted next to the column containing the headings. Next, a value is obtained from the dropdown box which is then used to identify the company information from the finished table on the “Table” sheet. Finally, these values are returned to the inserted column and the column is autofitted. A For Next loop that is used to transfer the data from the table to the inserted column. This sub is connected to the Show Data button and the dropdown box.
- 11. Sub removeData** – This sub removes all of the columns that have been inserted using the dropdown box. It does this using the enumerations xlDown and xlToRight and the delete method. This sub is connected to the Remove Data button.
- 12. Sub ShowData** – This sub simply takes the top 30 company names and final rankings from their range on the “Data Manipulation” sheet and places it in a range on the “Forbes Top” sheet. This sub is connected to the Show Top 30 button.

Conceptual Difficulties and What I Learned

Importance of Clearly Declaring Variables and Ranges:

I started this project by working on individual subs and defining all of my variables and arrays within each individual sub. I quickly learned that this was an error as many of my variables and especially my arrays needed to carry their values throughout the project. I declared all of my arrays at the top of the module on the project giving them module-level scope. I also declared Option Base 1 at the top of the module as this helped me conceptually when working with both arrays and ranges. I also discovered that it begins to matter very much what you name things when you start to accrue large amounts of variables and arrays. Specifically with the arrays, I found myself mistaking which ones were which because the names I gave them were too similar.

Avoid Copy & Paste:

When I began this project I used the copy method to retrieve data from the web query as well as to transfer data from one array to another. This quickly became messy and I was frequently having correct errors. Once I learned to effectively use ranges and arrays within For Next loops my code became a lot cleaner and more standard.

Test Sections of Code at a Time:

I started this project by trying to write as much of the code that I felt like I knew all at once. My reasoning was that it would be easier to get it all down and then go back through it more slowly. I learned however, that this only made it more difficult to pinpoint exactly what the problems in the code were. Testing the code one small piece at a time gave me a lot more confidence in the work that I had done and focused my problem solving efforts. Specifically in regards to the web queries, I learned that it was better to adjust my For Next loops to only look up a small fraction of the array values at a time to avoid the lag time of waiting for the entire array to pass through the web query.

Retrieving Ticker Symbols Online:

The portion of the code that I struggled the most with was attempting to use web queries to retrieve ticker symbol data. The problem that I encountered is that websites are simply not standard enough in their approach to company names and ticker symbols to use web queries. I attempted to use about ten different websites, but was unable to find any that I felt like would consistently return the correct values. A list of some of the difficulties I encountered and what I did to solve those difficulties is listed below:

- I discovered that the symbols in some of the company names were represented differently in the URL code. Two such examples are that the & in AT&T is represented by %26 and spaces are represented by +. I used the find method to replace all such symbols in my array of company names before using them in my web queries.
- Some of the websites would return multiple ticker symbols and the correct one was often not first. I tried inserting the array compName () into the find method to identify the correct ticker symbol but the company name was too often found in other locations on the page as well to consistently return the correct results.
- Many of the websites would offer a dropdown list to choose from rather than link to another page with a list to choose from.
- Some of the company's names double as their stock ticker and in this instance instead of going to a list of ticker symbols, the website would link you straight to the company page. I tried using an if statement to use different find methods depending on the length (len) of each company name thinking that name length would probably correspond with whether or not the name doubled as a ticker. This assumption proved not to hold true with examples being 3M and AT&T.

I was able to return the values of quite a few of the ticker symbols, but in the end I determined that my methods were not consistent enough for me to feel confident in the results. I manually entered the ticker symbols into excel and continued my project from that point. I do not think that this greatly affects the value of my code going forward a ticker symbol is just as easy and valid a starting place as a company name for this project.

Conclusion

This project allowed me to successfully identify companies that can be used in my research article and provided me with a framework that can be easily modified to accommodate future needs. In addition, I greatly expanded my knowledge and familiarity with VBA code. Finally, and perhaps most importantly it gave me the confidence that I can successfully write a useful program in VBA on my own.