Box Office Crawler

A VBA Project by
Matt Winn

IS 520
Section 1
Professor Gove Allen

Executive Summary

For my project, I decided to create something extremely practical and immediately useful in my own work and school. In Business Intelligence, my team and I wanted to research the answers to many important questions about the movie industry, such as what makes a movie successful. So, I created a VBA script to meet our huge need of grabbing thousands of pieces of data as well as converting them to usable forms and cleaning them in preparation for analysis. However, it has since evolved into a powerful data scraping and analyzing tool for anyone who wishes to use it. My VBA project works in conjunction with Box Office Mojo, the largest and most complete movie database in the world (owned by IMDB). Anybody who wishes to grab useful information on a movie or any number of movies, and then analyze the success of those movies based on many different factors, are welcome to use my workbook, and even embellish upon what I have created. The script I wrote to scrape data, easily executed through a custom button in the ribbon ("Get Movie Info"), uses the hyperlinks connected to any movie title copied from BoxOfficeMojo.com in the Title column of the "Movies A-Z" worksheet to grab the producing studio, total box office gross, opening date, rating, genre, director (if listed/known), total known days in theaters, day of the week opened, budget (if listed/known), and runtime of the movie, all in two seconds or less (per movie). It also converts all gross and budget figures to 2014 dollars, so that movies from all years may be compared side-by-side, and it extracts the specific year and month of the movie release, for purposes of analyzing. Once all records and related data have been retrieved, the user may click "Create Result Charts" and the workbook will generate multiple charts to compare movies based on earnings.

Implementation Documentation

The following is a hierarchy of the code in BOMOJO_Graphs&Ribbon.xlsm:

- xRibbon.btngetInfo_click (ribbon subroutine)
    - Runs WQGetValues.FillVariables
        - Runs WQGetValues.Format
        - Runs WQGetValues.Scrape
            - Uses WQGetValues.FindMonth (function)
            - Uses WQGetValues.GetDays (function)
        - Runs WQGetValues.Calculate
- xRibbon.btnmakeCharts_click (ribbon subroutine)
    - Runs WQGetValues.ChartMaker
        - Uses WQGetValues.ClearCharts
        - Uses WQGetValues.ResizeCharts

When using the modules in this workbook, the user must first copy and paste the titles of the desired movies from BoxOfficeMojo.com into the Title column of the “Movies A-Z” worksheet (see Figure 1). All titles copied from the BoxOfficeMojo movie results section will have a hyperlink connected to them that can be used to navigate the web query wizard (see Figure 2). After all desired movie titles are copied to the Title column, the user need only click the “Get Movie Info” button under the “Movie Functions” tab. When this button is clicked, btngetInfo_click in the xRibbon module is called, which in turns calls the FillVariables subroutine in the WQGetValues module.

FillVariables runs three separate subroutines: Format, Scrape, and Calculate. The Format subroutine changes all variable columns to the correct formats, automatically deletes rows that have “n/a” in them (since this has proven to be indicative of how little information is available on that movie), and extracts both the day of the week and the year on which each movie was released.

The Scrape subroutine uses the web query wizard in conjunction with the movie title hyperlinks to navigate to each movie web page and scrape all text from the page (See Figure 3). This text is dumped into the “WebQuery” worksheet, and Scrape uses the find method to extract the appropriate variables from the text. Most of these variables do not stand alone in the text and therefore Scrape also splices and parses many clumps of text to produce the desired variables. In some cases, a figure does not exist for the much desired Days In Theater column; in these cases, Scrape extracts the Close Date from the page text, utilizes the FindMonth function to parse the date and the GetDays function to manually calculate the number of days between the release date and the close date. Finally, it converts all numeric strings to their appropriate data types and uses an average ticket prices by year chart on the “Prices” worksheet to calculate how many box office tickets were sold for each movie and thereby finds the adjusted 2014 box office gross amount (see Figure 4).

The Calculate subroutine works similarly to the Scrape subroutine, but with one purpose: to adjust all available Budget figures for inflation. It uses the Consumer Price Index Inflation Calculator from data.bls.gov to input budget figures and subsequently scrape and convert output figures back into numeric variables (see Figure 5). As one might expect, this makes use not only of the Budget column but also of the Year column created in Format.

Once all movie data has been collected and formatted, the user may wish to see how the movies compare to each other in their success based on their genre, rating, or release month. In this case, the user can click on the “Create Result Charts” button and get their wish. The button calls the btnmakeCharts_click subroutine, which in turn calls the ChartMaker subroutine.

The first thing that ChartMaker does is it calls the ClearCharts subroutine to empty the “Charts” worksheet and pave the way for a new set of charts. After that, it selects all of the data from the “Movies A-Z” worksheet and creates a pivot table with it in the “PTables” worksheet. It does this three times, each time creating a different pair of variables in the pivot table. The first pivot table finds movie success (adjusted gross) by rating, the second pivot table finds movie success by genre, and the third finds movie success by release month (e.g. do movies make more when released as a summer blockbuster or around Christmastime?). Once the pivot tables are built, ChartMaker takes them and creates appropriate visual graphs from them. The ratings table is shown in a column chart, the genre table is shown in a bar chart, and the release month table is shown in a pie chart (I felt like those chart types were most appropriate for their respective data sets). If the user wishes to add a different chart in the future, all he need do is simply copy and paste the code from a previous pivot table and chart and switch the variable names. After each chart has been created, ChartMaker calls on the ResizeCharts subroutine, which conveniently resizes each chart depending on its type, thus rendering it much more readable for the user (see Figure 6).

Learning and Conceptual Difficulties

There were many difficulties that I encountered during this project. During my first run of writing the VBA code for this, I used Agent (Gove Allen's creation) to navigate through the web pages and scrape the data. I had very little experience with Agent, so the learning curve was steep. I found it especially difficult to use the tag-identifying methods to find the data pieces I wanted; it was even more difficult when I discovered that the structure of each page was different. After finishing the majority of my VBA script, I came to a crossroads. At this point, Agent was taking approximately 30 seconds on average to grab the data for each movie. Considering that I was trying to scrape the data for over 2,200 records, this was a major time investment. However, when a previous student mentioned that Web Query might be able scrape the data more quickly than Agent, I decided to look into it. Working with Web Query is extremely different from working with Agent, so it, too, held a steep learning curve for me. Yet, the results were worth it. Whereas Agent took 30 seconds to grab the data for a movie, Web Query takes only 2 seconds on average. This was a great triumph because it suddenly took the required running time for our records from almost a whole day to about an hour.

That being said, a few other specific walls I ran into and had to climb over include when I found out that each BoxOfficeMojo page was structurally different—this forced me to rethink my searching methods and write a lot of contingency code; when I was writing the code for resizing the charts—there was no publicly available method that would work in my scenario, so I had to come up with a bit of original code; and when I had to learn how to create pivot tables. Learning how to create pivot tables from the movie data proved to be something of a

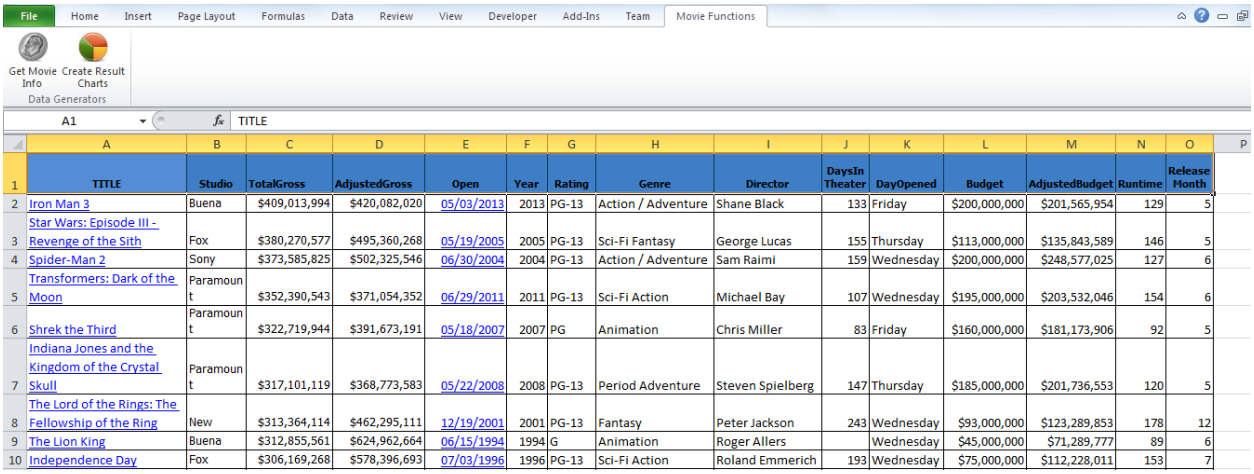
nightmare. It took me hours to figure out how to simply get Excel to stop yelling at me for illegal commands, and all this even after using Excel's own macro code to learn how it all works.

Assistance

I received absolutely no help from any other individual when working on this project.

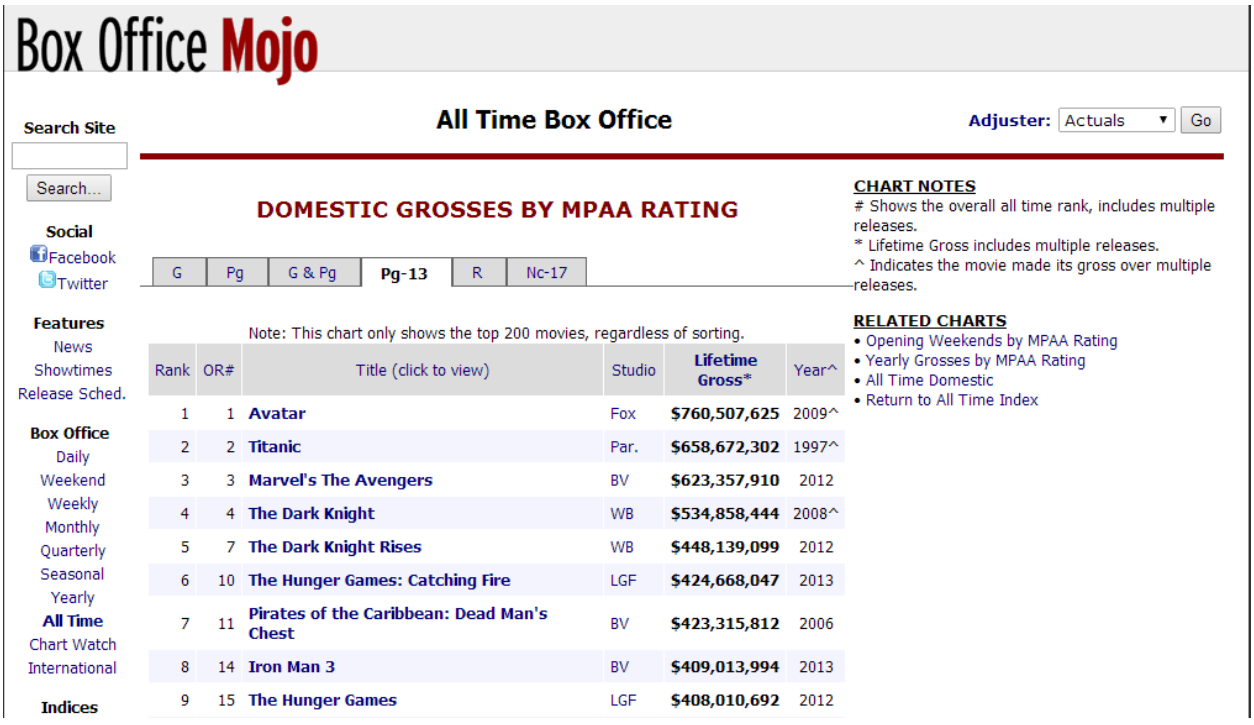
Appendix

Figure 1 – "Movies A-Z" worksheet and "Movie Functions" ribbon buttons



	TITLE	Studio	TotalGross	AdjustedGross	Open	Year	Rating	Genre	Director	DaysIn Theater	DayOpened	Budget	AdjustedBudget	Runtime	Release Month
2	Iron Man 3	Buena	$409,013,994	$420,082,020	05/03/2013	2013	PG-13	Action / Adventure	Shane Black	133	Friday	$200,000,000	$201,565,954	129	5
3	Star Wars: Episode III - Revenge of the Sith	Fox	$380,270,577	$495,360,268	05/19/2005	2005	PG-13	Sci-Fi Fantasy	George Lucas	155	Thursday	$113,000,000	$135,843,589	146	5
4	Spider-Man 2	Sony	$373,585,825	$502,325,546	06/30/2004	2004	PG-13	Action / Adventure	Sam Raimi	159	Wednesday	$200,000,000	$248,577,025	127	6
5	Transformers: Dark of the Moon	Paramount	$352,390,543	$371,054,352	06/29/2011	2011	PG-13	Sci-Fi Action	Michael Bay	107	Wednesday	$195,000,000	$203,532,046	154	6
6	Shrek the Third	Paramount	$322,719,944	$391,673,191	05/18/2007	2007	PG	Animation	Chris Miller	83	Friday	$160,000,000	$181,173,906	92	5
7	Indiana Jones and the Kingdom of the Crystal Skull	Paramount	$317,101,119	$368,773,583	05/22/2008	2008	PG-13	Period Adventure	Steven Spielberg	147	Thursday	$185,000,000	$201,736,553	120	5
8	The Lord of the Rings: The Fellowship of the Ring	New	$313,364,114	$462,295,111	12/19/2001	2001	PG-13	Fantasy	Peter Jackson	243	Wednesday	$93,000,000	$123,289,853	178	12
9	The Lion King	Buena	$312,855,561	$624,962,664	06/15/1994	1994	G	Animation	Roger Allers		Wednesday	$45,000,000	$71,289,777	89	6
10	Independence Day	Fox	$306,169,268	$578,396,693	07/03/1996	1996	PG-13	Sci-Fi Action	Roland Emmerich	193	Wednesday	$75,000,000	$112,228,011	153	7

Figure 2 – Table of movies from BoxOfficeMojo.com



Rank	OR#	Title (click to view)	Studio	Lifetime Gross*	Year^
1	1	Avatar	Fox	$760,507,625	2009^
2	2	Titanic	Par.	$658,672,302	1997^
3	3	Marvel's The Avengers	BV	$623,357,910	2012
4	4	The Dark Knight	WB	$534,858,444	2008^
5	7	The Dark Knight Rises	WB	$448,139,099	2012
6	10	The Hunger Games: Catching Fire	LGF	$424,668,047	2013
7	11	Pirates of the Caribbean: Dead Man's Chest	BV	$423,315,812	2006
8	14	Iron Man 3	BV	$409,013,994	2013
9	15	The Hunger Games	LGF	$408,010,692	2012

Figure 3 – Sample movie page from BoxOfficeMojo.com

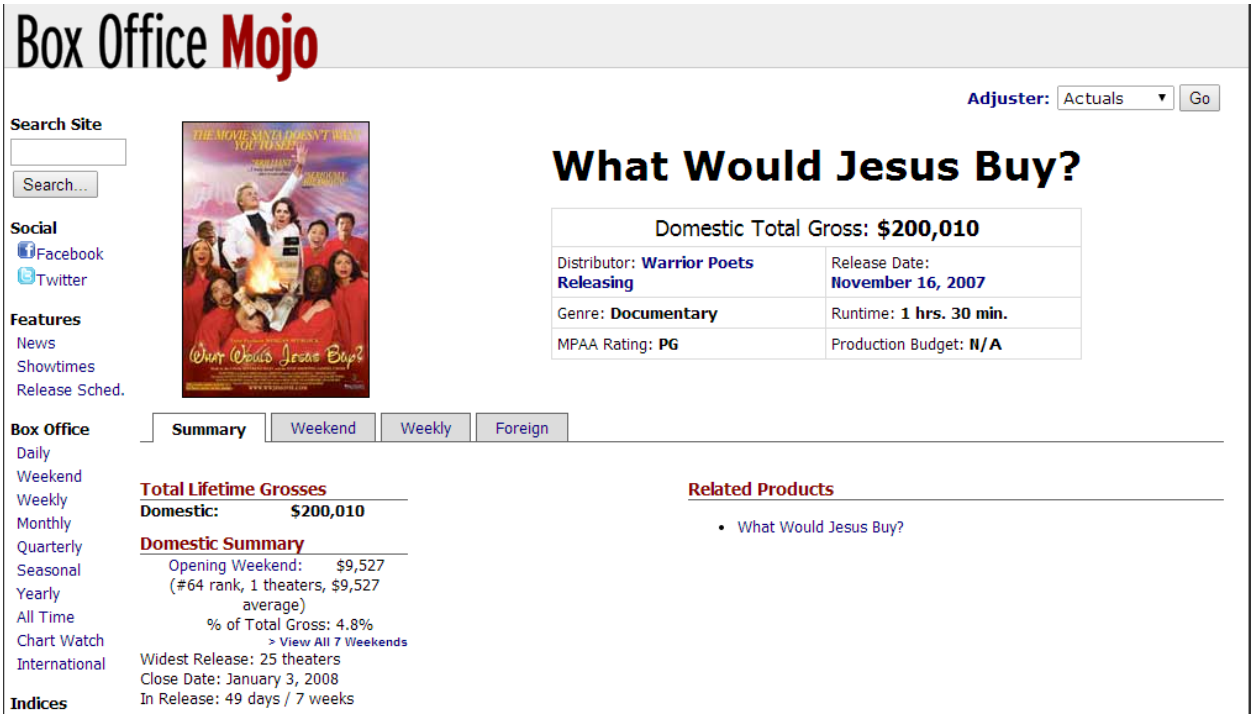


Figure 4 – Average ticket price by year table on "Prices" worksheet

Year	Avg. Price
2014	$8.35
2013	$8.13
2012	$7.96
2011	$7.93
2010	$7.89
2009	$7.50
2008	$7.18
2007	$6.88
2006	$6.55
2005	$6.41
2004	$6.21
2003	$6.03
2002	$5.81
2001	$5.66
2000	$5.39
1999	$5.08
1998	$4.69
1997	$4.59

Figure 5 – Screenshot of Consumer Price Index Inflation Calculator at data.bls.gov