

## Developing an Introductory Class in Business Intelligence (BI) Using MS Excel Powerpivot

Dr. Sam Hijazi  
Trevor Curtis  
Texas Lutheran University  
1000 West Court Street  
Seguin, Texas 78130  
[shijazi@tlu.edu](mailto:shijazi@tlu.edu)

### Abstract

Asking questions about your data is a constant application of all business organizations. To facilitate decision making and improve business performance, a business intelligence application must be an integral part of everyday management practices. Microsoft Excel added PowerPivot and PowerPivot officially to facilitate this process with minimum cost, knowing that many business people are already familiar with MS Excel.

This paper will design an introductory class to business intelligence (BI) using Excel PowerPivot. If an educator decides to adopt this paper for teaching an introductory BI class, students should have previous familiarity with Excel's functions and formulas. This paper will focus on four significant phases all students need to complete in a three-credit class. First, students must understand the process of achieving small database normalization and how to bring these tables to Excel or develop them directly within Excel PowerPivot. This paper will walk the reader through these steps to complete the task of creating the normalization, along with the linking and bringing the tables and their relationships to excel. Second, an introduction to Data Analysis Expression (DAX) will be discussed.

### Introduction

It is not that difficult to realize the increase in the amount of data we have generated in the recent memory of our existence as a human race. To realize that more than 90% of the world's data has been amassed in the past two years alone (Vidas M.) is to realize the need to manage such volume. Small and large businesses alike will be required to create and store large amounts of data not only for now but for years to come. The exponential increase in data accumulation would require some cleansing, storing, manipulating, and deep mining to get the benefits we expect from a costly investment. The effectiveness of decision making depends on analyzing the data to produce some Business Intelligence (BI) values to all aspects of any organization. Business Intelligence is just that, where management at all levels uses technology to acquire insight into a broad set of data. Often factors such as products, customers, employees, partners, and others play a big role in BI. This is where the execution of Power Pivot and data analysis expressions (DAX) comes in handy. It is available as Microsoft Excel Add-Ins and with the latest version of MS Excel 2016, is free. PowerPivot allows the customization of the very complicated business model that requires the use of normalized tables, based on a predetermined relationship with millions of rows.

This paper attempts to encourage educators in the Information Systems field to explore the option of using PowerPivot and DAX as an introductory class in BI. Students who will graduate soon are more likely to have to use MS Excel as a number crunching, charting, and what-if analysis software. Adding Power Pivot and DAX to their list of classes will guarantee not only deep theoretical knowledge but also will add much needed applicable skills to their resumes.

### **Problem Background**

Many agree that MS Excel is a powerful spreadsheet. Having said that, it should be noted that Excel, until recently, could not organize data based on their relationships as entities. The other major limitation is found in the small number of rows and columns, considering the massive increase in the size of data generated by businesses.

### **Absences of Data Model in MS Excel**

As powerful as MS Excel was, it never allowed the formation of a genuine relation among different tables as you would construct in a relational database such MS Access. In the past, to get the advantage of the best world, we need to use both database and spreadsheet applications. We had to integrate the two software applications, but they were still two separate applications. This has caused some problems since MS Access has good query manipulation but limited calculation and analytical capabilities. Excel is a great tool for performing tedious calculation but lacked the structure and the data integrity found in a relational database such as MS Access.

### **Limited Number of Rows and Columns**

As was stated in the introduction of the paper, you could easily discover that a sheet in MS Excel is limited to 16,386 columns by 1,048,576 rows. What about if we need to bring multimillion records from a database for additional analysis in MS Excel? It is nearly impossible to move that many records into a sheet resulting in frustration and additional dependency on large but expensive software.

### **Literature Review**

The focus of this paper is to convince educators who are interested in introducing their students to BI to use MS Excel's new Add-Ins found in PowerPivot and DAX. The availability of Excel makes it a good choice if and only if it comes with the required concepts and tools to create a three-credit class.

The use of a relational database was restricted to pure database management systems (DBMS). Now, these capabilities of linking multiple entities based on the type of their relationships are available in PowerPivot. This has been a dream that has come true. Not only we can use the traditional functions and formulas found in a powerful spreadsheet such as Excel, but also, we can reduce data anomalies especially, data redundancy in our data set. Also, this paper will introduce through screenshots some of the features found in PowerPivot and DAX.

To assist an educator who wishes to teach Business Intelligent class successfully, the literature review will be divided into three sections. These are the data model in PowerPivot, DAX relationship to MS

## *2018 ASCUE Proceedings*

Excel, and reporting capabilities linked to PowerPivot as these are related to PivotTables and Power View.

### **PowerPivot Data Model**

The paper will use MS Excel 2013, where, now, the last version is MS Excel 2016. When it comes to crunching and building complicated formulas, Excel is an absolute champion. Having said that, we need to know that Excel sheet has an obvious limitation, where it has only  $2^{20}$  or 1,048,576 rows by 16,386 columns. PowerPivot has dealt with this issue by expanding the number of rows and columns to much larger values, to be exact 1,999,999,997 rows and 2,147,483,647 columns according to Microsoft Developer Network website.

Alexander (n.d.) stated that “At its core, PowerPivot is essentially a SQL Server Analysis Services engine made available by way of an in-memory process that runs directly within Excel and is “referred to as the internal Data Model. By using the PowerPivot Ribbon interface, you can shrink imported data to tenth its typical size.” Alexander adds, a 100MB text file only takes up to 10MB in the Internet Data Model. This speed will reduce the frustration usually associated with a large spreadsheet.

With a large data set, a user has the capability of creating relationships between tables, construct table hierarchy, and, more than anything, established the relationships between these data like any relational database. It is important to note that the tables can come from many different sources, which can all be normalized into a uniform data set on which the program will operate. According to Jackson (2010), Data can be reorganized around one column to be compared to another set of data from different sources. Data can be divided by time, geography, location or many other parameters. Finally, the author stated, “since it runs Microsoft's business intelligence software on the back end, it can do much of what a full-fledged BI application can do.”

### **Data Analysis Expressions, DAX.**

DAX in Power Pivot aids in the ease of use and quick execution of calculations. DAX formulas are in some ways like Excel formulas. Users may come across DAX functions that have the same name as an Excel function but were adapted to suit more customized means of input and output (Microsoft, “Filter Functions (DAX)”).

### **PivotTables and PowerViews**

**PivotTables.** According to Michaloudis (n.d.) PivotTables are “the most powerful features within Microsoft Excel.” It allows users to have access to millions of datasets with just a click of a mouse, resulting in neatly formatted tables. From an article by Association of Business Training (2013), the author states that PivotTables have many benefits including ease of use, easy data analysis, easy summary of data, ease identification of data patterns, quick report creation, and great decision-making tool. In addition to all these capabilities, let’s not forget about the powerful feature that is tightly linked to a PivotTable, that is, PivotChart.

**PowerViews.** According to the article “Charts and other visualizations in PowerView” (n.d.), PowerView allows a user to quickly create a variety of data visualizations based on tabulated data or matrices by converting them into a multitude of charts including the known ones, such as bar, column, and bubble chart.

## Going Through a Complete Example

### Setting Up Power Pivot and DAX

**Getting Started with The Add-Ins.** After enabling the PowerPivot Add-ins, users should take their time to familiarize themselves with the overall layout and Guided User Interface (GUI) within the new “Power Pivot” tab. Figure 1 illustrates the main options a user has when in the menu, which contains Manage, Measures, KPIs, Add to Data Model, Update All, Detect, and Settings within the tab. A tab in Excel is a part of the overall top ribbon and is how users have access to display options.

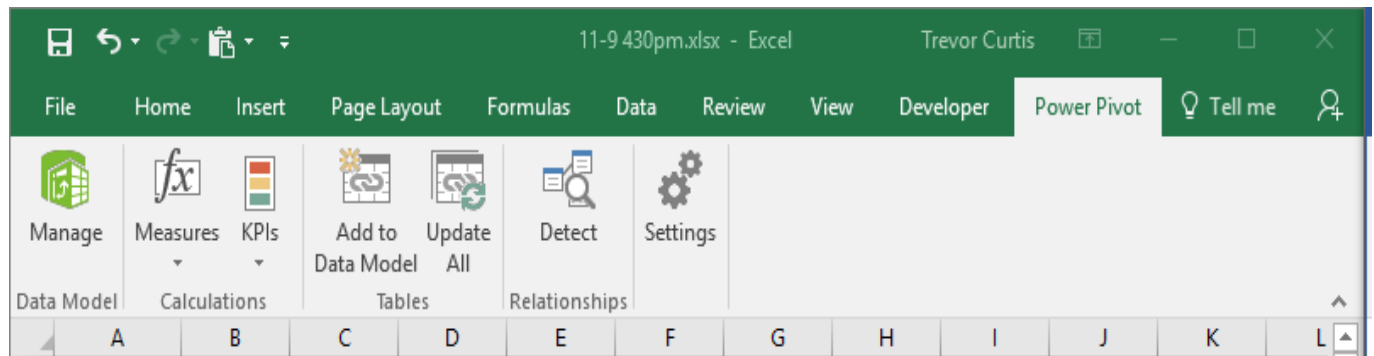


Figure 1: Power Pivot Ribbon Menu

When the Manage button is clicked, a new window opens with a new ribbon at the top. As shown in Figure 2, the Home, Design, and Advanced ribbons each contain usable tools. Data will not be automatically added to the Power Pivot data model and must be done manually.

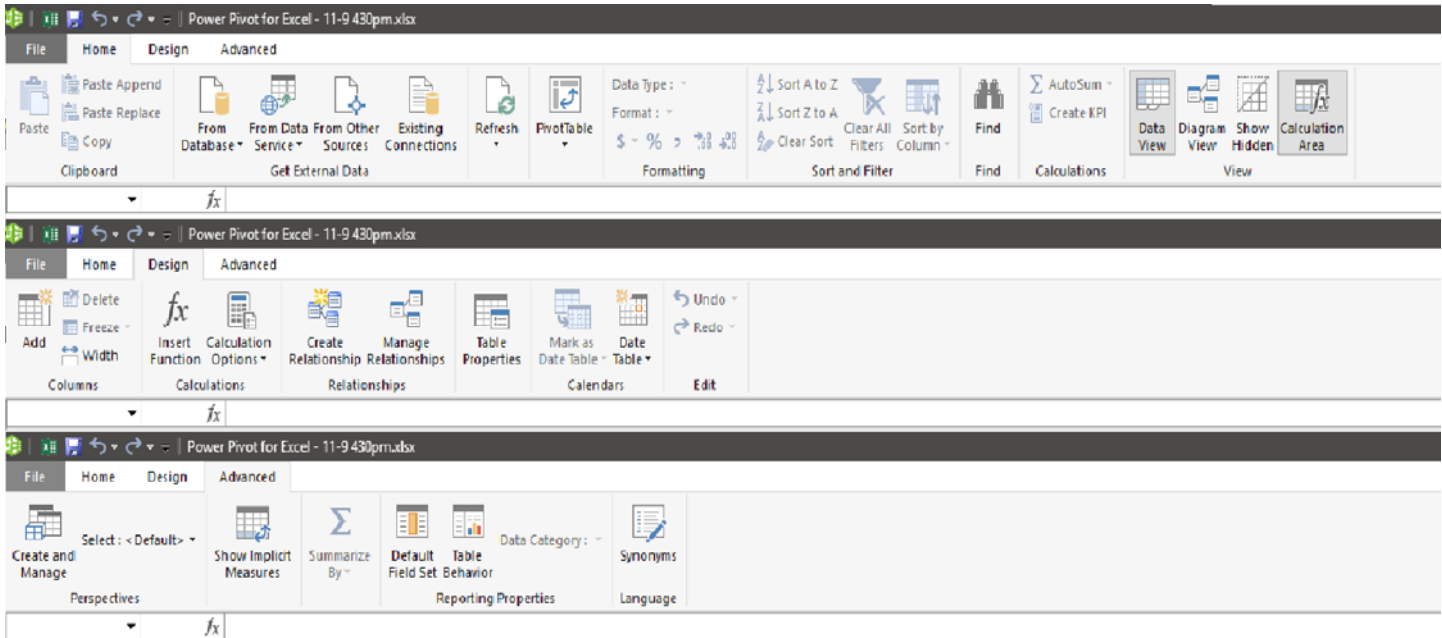


Figure 2: “Manage” Tool

Next on the Power Pivot ribbon is the Measures Tool. Measures are formulas that can be applied to the whole dataset using PivotTables/PivotCharts (Microsoft, “Data Analysis Expressions (DAX) in Power Pivot”). When considering Measures, a user must realize that a customized formula could be entered in a single cell and will affect the whole data set. Measures set up a whole new aspect of drill down techniques in PivotTables because they can pull data from other tables, and not just one like normal Excel PivotTables. Measures can be added two different ways, through the Power Pivot ribbon using the Measures button or via the Manage tab by using the Calculation Area toggled on. Figure 3 shows the simplicity of adding a measure to a PivotTable through the Measures button within the PowerPivot ribbon.

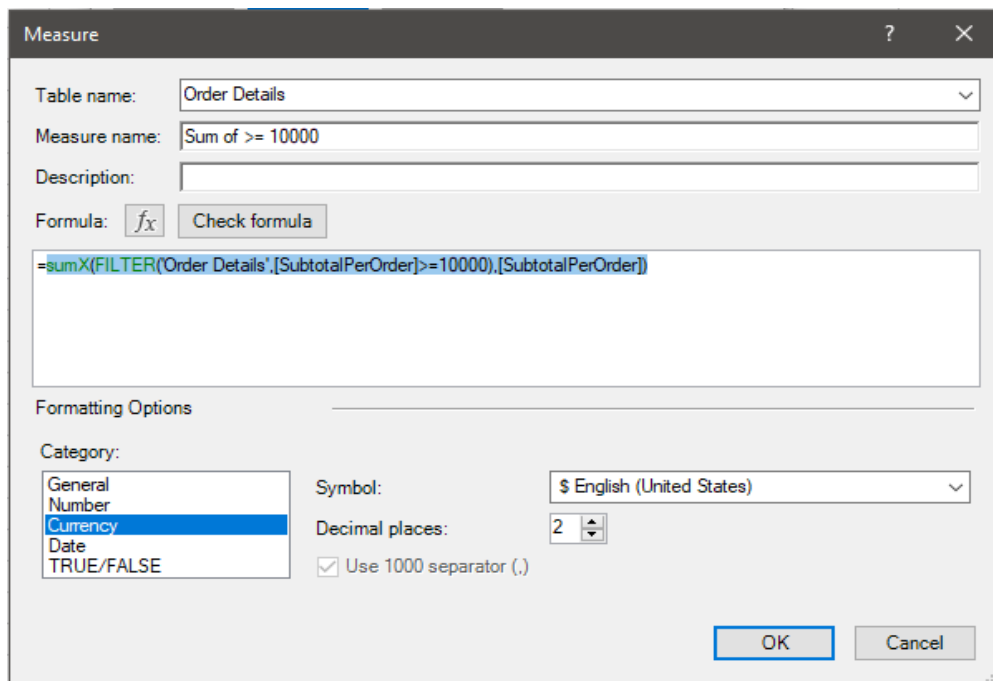


Figure 3: Adding a Measure

The formula feature can be used to illustrate the syntax and easy to use GUI. Once entered, a formula can be checked for syntax with the corresponding button, resulting in instant alerts if any errors are detected. Since the formatting option Currency is selected, it will never need to be specified in later implementations of the Measure. This measure can be found later in the PivotTable selections under the Order Details table.

Key Performance Indicators (KPIs) are the third tool available through the Power Pivot ribbon and provide for visual effects and indicators for a selected section of the data set. KPIs are most used for calendars as well as comparing historical with projected data (Parke Godfrey, Jarek G. & Piotr L.)

Add to Data Model, Update All, and Detect are options in the Power Pivot ribbon that add convenience to the user, so they are not constantly switching through the Manage and normal Excel window. While in the normal Excel spreadsheet mode, a table is either added to the data model locally through Excel or from most common sources such as Microsoft SQL Server, Microsoft SQL Azure, Oracle, and many others.

### Normalizing and Importing Data

To remove most, if not all of the data anomalies within a database, the dataset must be normalized into separate tables. Relationships must be carefully examined and applied. Normalization was achieved by removing redundancies resulting in increased data integrity and by establishing relationships between columns of data amongst tables (Surajit C., Umeshwar D. & Vivek N.) To further assure the accuracy of the normalization process, the sheets were exported to Microsoft Access, where the relationships could be established correctly and with complete assurance that all relationships are created accordingly. This was performed using the Get External Data tool in Manage, pictured previously in Figure 2.

The ID for both the Shippers and Suppliers table had to be created by taking the first three letters of the Company Name and appending three random numbers to the end. This allowed for a unique ID that would follow the rules of a normalized table while providing a unique set of characters specific to the company. The formula used to complete this task was:

`=UPPER((LEFT([@CompanyName],3))) & RANDBETWEEN(100,999)`

where the UPPER function stops the count at the specified three characters, and the LEFT function tells the system where to start counting from one to three.

Once the tables were imported, a user should use the Diagram View within Manage, pictured below in Figure 4, to check the relationships and the flow of data. There were no more problems with filter direction, and data manipulation could then be performed.

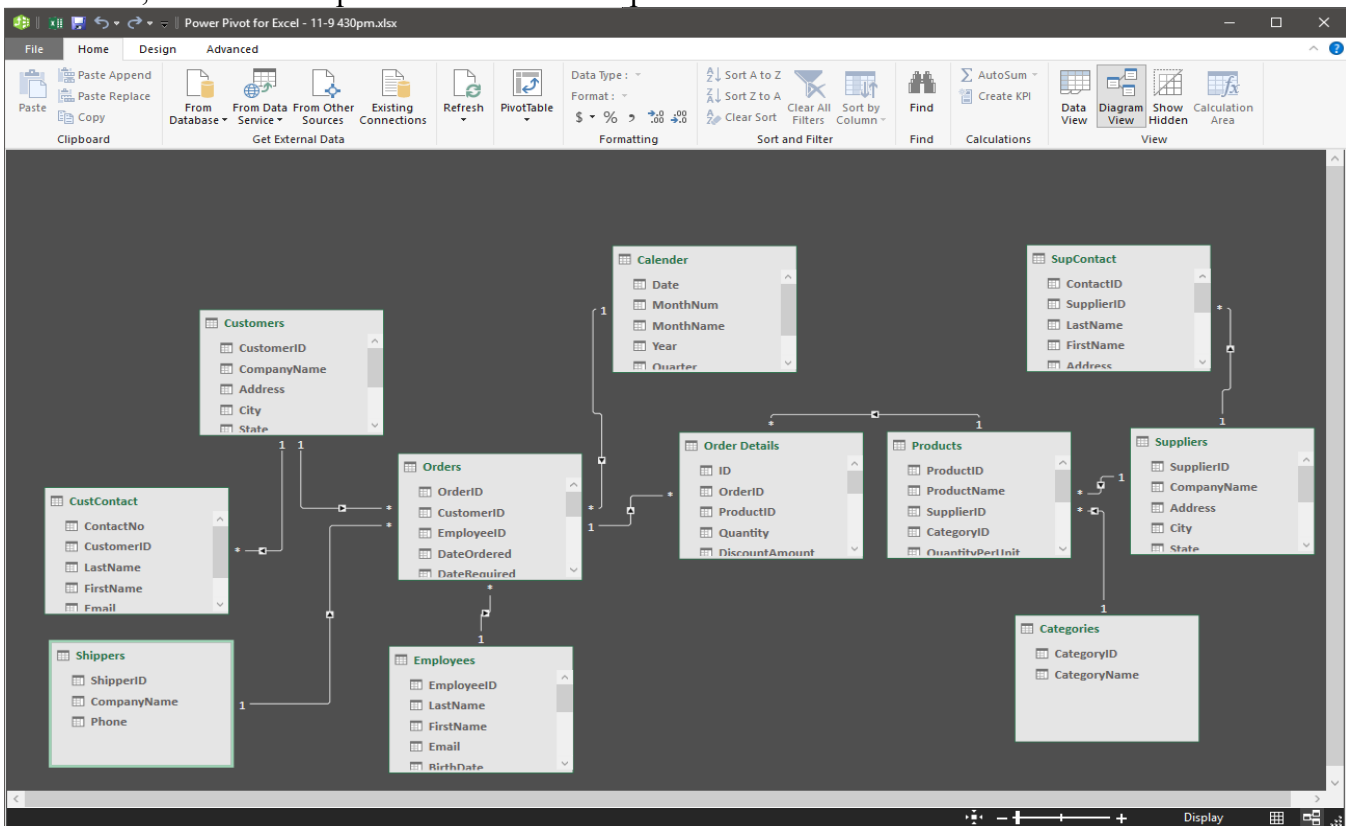


Figure 4: Diagram View of Normalized Data

## Implementation

**Addition of the Calendar Table.** After minor experimentation with the massive data set that created, normalized, and imported, it was realized that most of the questions involved assessing the date of sales. The calendar year is important for businesses and is usually broken down into quarters (Stavros Valsamidis et al.). A user can use the local Excel workbook to import a table into the data set consist-

ing of every single day from the time the first sale was made to current time. Once imported, a user can use PowerPivot's foreign-key detection tool Detect to automatically link the Date from the Calendar table with DateOrdered in the Orders table (Zhimin C., Vivek N., and Surajit C.). Upon switching to Data View in the Manage window, seven calculated columns with the following formulas were created:

Single out the month for "MonthNum"

`=MONTH([Date])`

Name the month using three characters for "MonthName"

`=FORMAT([Date],"mmm")`

Single out the year for "Year"

`=([Date])`

Determine the Quarter for "Quarter"

`=CEILING(Calendar[MonthNum],3)/3`

Year concatenated with Quarter for "Year-Quarter"

`=Calendar[Year]&"-Q"&Calendar[Quarter]`

Fiscal Quarter number

`=IF('Calendar'[MonthNum]<4,4,'Calendar'[Quarter]-1)`

Fiscal Year

`=IF('Calendar'[MonthNum]<4,'Calendar'[Year]-1,'Calendar'[Year])`

Fiscal Year concatenated with Fiscal Quarter number

`= 'Calendar'[FiscalYear]&"-Q"&'Calendar'[FiscalQuarterNumber]`

The above formulas are DAX functions that also happen to be Excel formulas. It was easy to expand these formulas and apply the DAX syntax which uses Table[Column] format. The MONTH and FORMAT formulas were used to format the existing date into manageable columns. The CEILING formula refers to the MonthNum column in the Calendar table and changes the month number to a multiple of three. The formula was then divided by three to get the quarters I needed. Finally, the year was concatenated with the quarter in a visually appealing way for future manipulation and granular analysis of earlier company performance (Sumit G., William H., and Rishabh S.). Concatenation is the linking of two calculated fields together to form one uniform calculated field. The same steps were repeated for the fiscal year and quarter but months with ceiling > 4 had -1, so the fiscal calendar started in March. Figure 5 illustrates the result for the Calendar table in the Manage window.



	[Quarter]	fx						
		=CEILING(Calendar[MonthNum],3)/3						
Date	Mon...	MonthName	Year	Quar...	Year-Quarter	FiscalQ...	FiscalYear	FY-Quarter
1/2/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/3/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/4/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/5/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/6/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/7/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/8/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/9/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/10/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/11/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/12/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/13/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/14/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/15/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/16/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/17/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/18/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/19/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/20/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/21/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/22/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/23/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/24/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4
1/25/2008 12:00:00 ...	1	Jan	2008	1	2008-Q1	4	2007	2007-Q4

Figure 5: Calendar Table

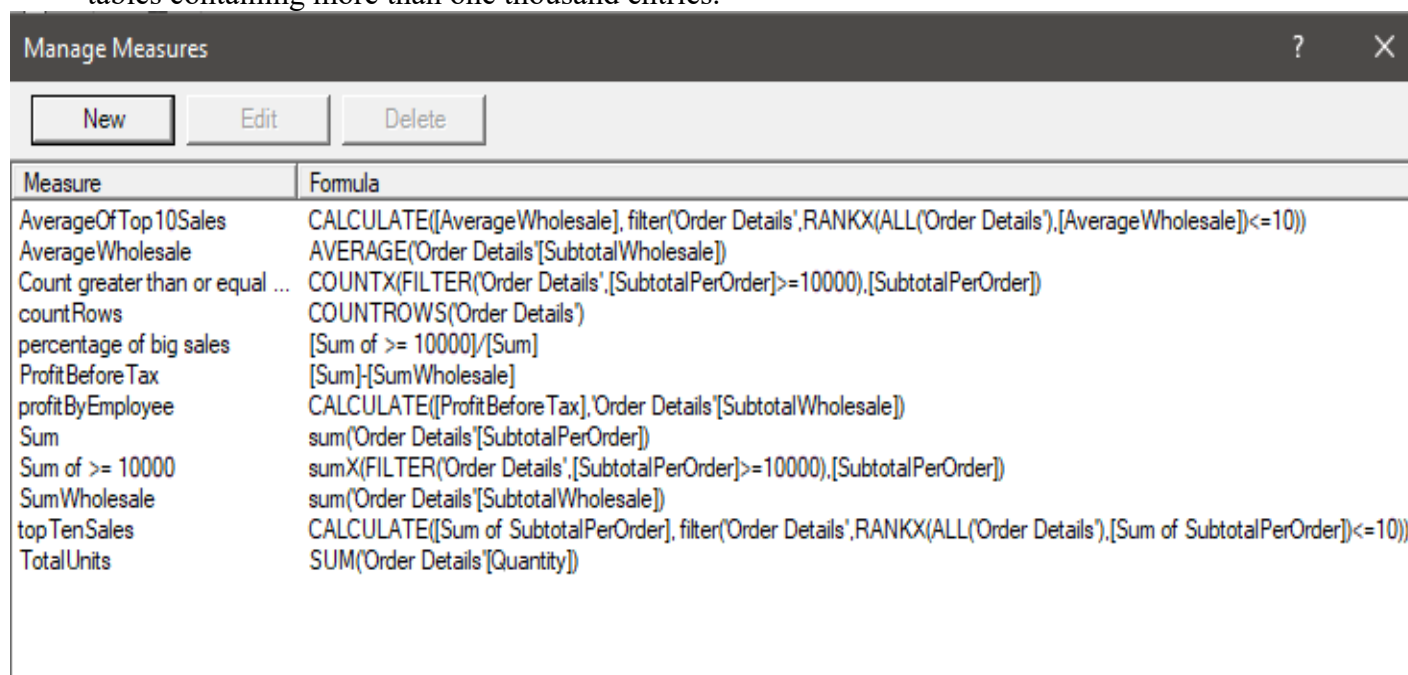
**Applying Measures.** The first application of measures in this walkthrough example was creating a Sum formula. To do this, there was a need to create calculated columns in the Order Details table that would allow the access to the numbers necessary for the Sum calculation. Calculated columns differ from measures because measures are only one cell and do not operate on scalar values (Danyel F., Steven D., and Mary C.) like normal Excel spreadsheets, and pierce through every single datasheet. Calculated Columns resemble traditional Excel formulas and do operate on scalar values.

Once the SubtotalPerOrder column was created and calculated using:

$$=RELATED(Products[SalesPrice]) *(1-'Order Details'[DiscountAmount])*'Order Details'[Quantity]$$

the DAX function RELATED was implemented and introduces the calculation of calculating values that can only be achieved through relational database sets. Products[SalesPrice] is related to the Order Details table so values can be used wherever necessary to obtain the calculations. Figure 6 below shows the Manage Measures tool in the PowerPivot ribbon where six more measures are entered and calculated to be used in Pivot Tables. Each measure has its cell in the Order Details table and can be edited accordingly. Any time a table name is changed in the data set, or data that formulas are related to

is changed, then PowerPivot will update the entire set. This is convenient when working with over ten tables containing more than one thousand entries.



Measure	Formula
AverageOfTop10Sales	CALCULATE([AverageWholesale], filter('Order Details',RANKX(ALL('Order Details'),[AverageWholesale])<=10))
AverageWholesale	AVERAGE('Order Details'[SubtotalWholesale])
Count greater than or equal ...	COUNTX(FILTER('Order Details',[SubtotalPerOrder]>=10000),[SubtotalPerOrder])
countRows	COUNTROWS('Order Details')
percentage of big sales	[Sum of >= 10000]/[Sum]
Profit Before Tax	[Sum]-[SumWholesale]
profitByEmployee	CALCULATE([Profit Before Tax], 'Order Details'[SubtotalWholesale])
Sum	sum('Order Details'[SubtotalPerOrder])
Sum of >= 10000	sumX(FILTER('Order Details',[SubtotalPerOrder]>=10000),[SubtotalPerOrder])
SumWholesale	sum('Order Details'[SubtotalWholesale])
topTenSales	CALCULATE([Sum of SubtotalPerOrder], filter('Order Details',RANKX(ALL('Order Details'),[Sum of SubtotalPerOrder])<=10))
TotalUnits	SUM('Order Details'[Quantity])

Figure 6: Managing Measures

The measure “Sum of >= 10000” was one of the first questions answered, where the SUMX formula was combined with FILTER to single out special cases where a total order was over ten thousand dollars. The syntax was easy to catch on to, but it was not for some time that I found that SUMX was to be used instead of SUM since the formulas affected more than one column. Figure 7 illustrates a PivotTable that uses both the calculated column and SUMX formula to group OrderIDs in their corresponding Customer.

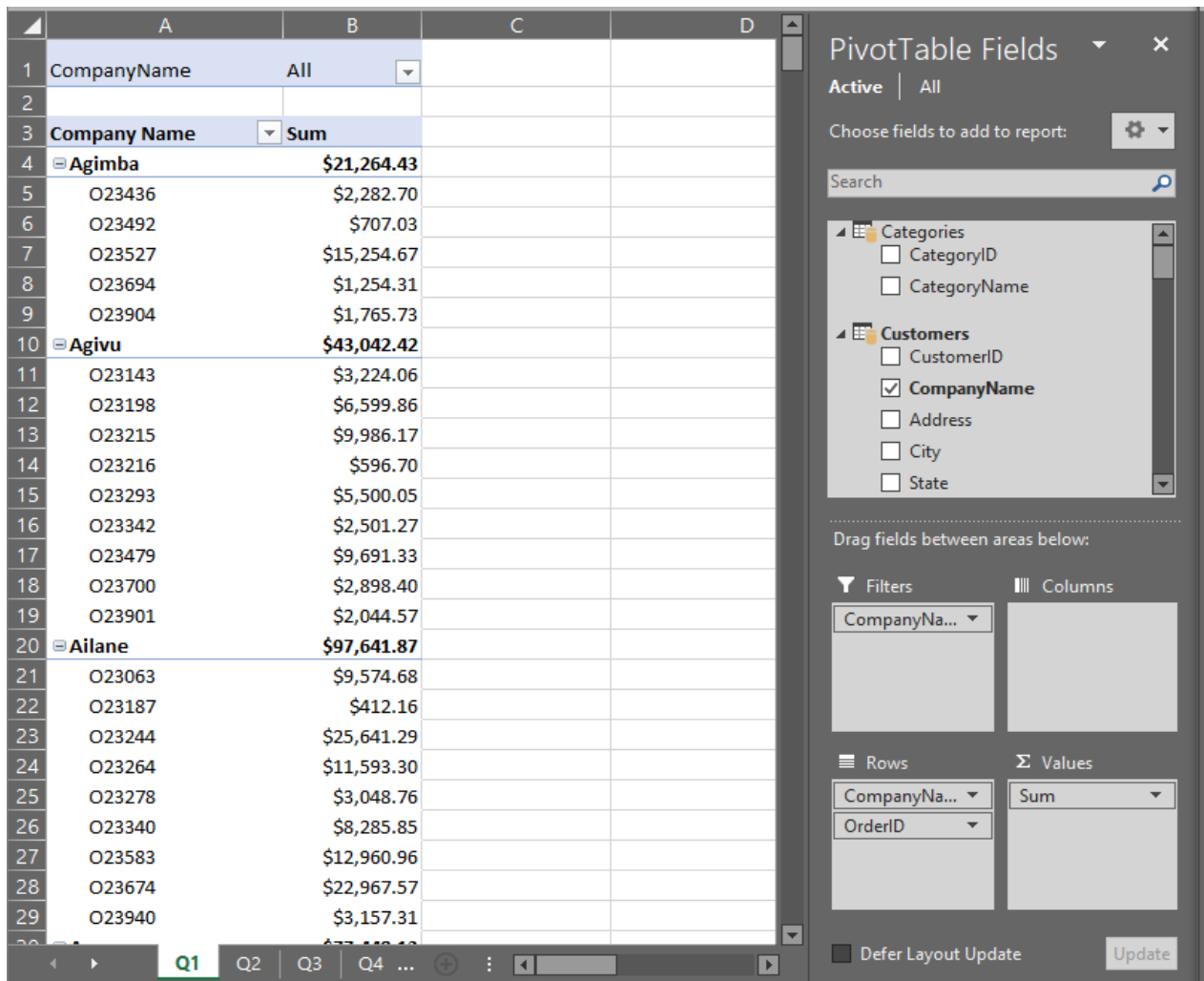


Figure 7: OrderID Grouped by Customer (Pivot Table)

For anyone who has not used a PivotTable before, the above table will give them a new understanding of Excel tables by demonstrating that values can be pulled from tables without having to do any guessing and checking. With the relational functionality, data is again updated throughout so PivotTables are constantly updated when the set is changed by the user or by calculations. The table can be filtered by additional fields and is completely customizable to show whichever range of output a user chooses.

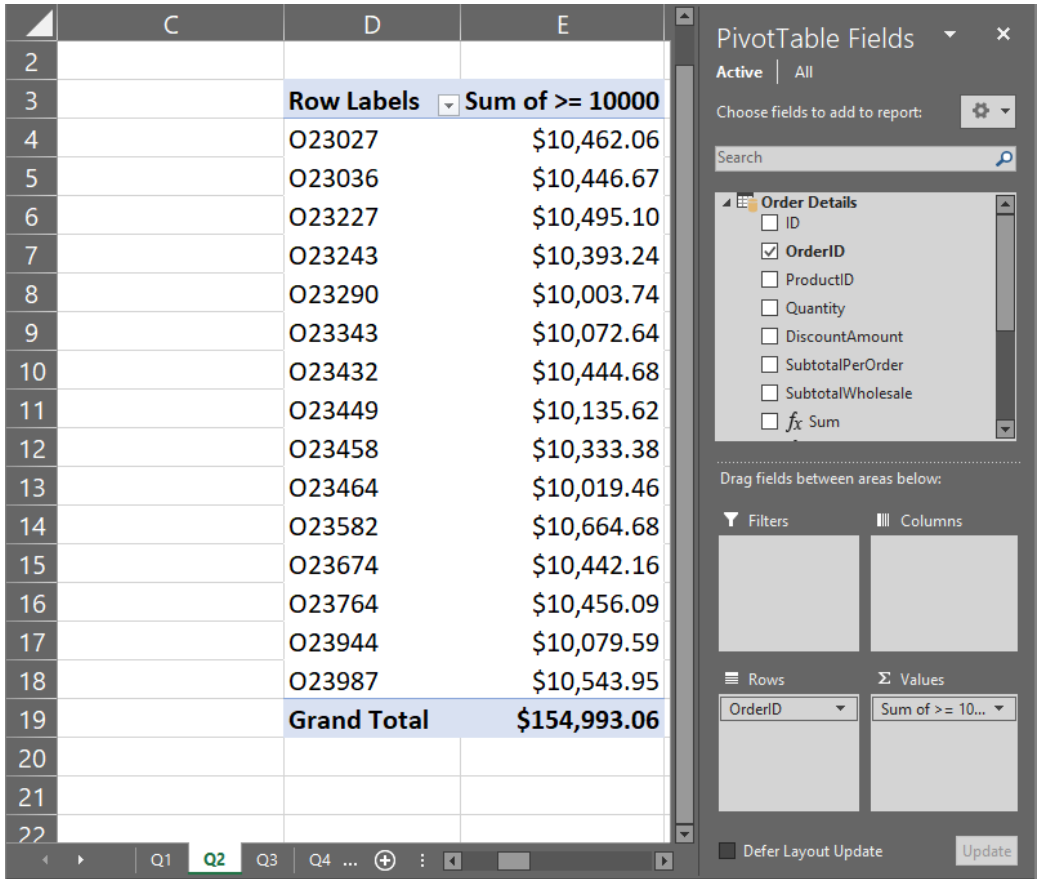


Figure 8: Orders Greater Than or Equal to \$10,000 (PivotTable)

Figure 8 shows another PivotTable where the SUMX and FILTER DAX formulas were used to single out big orders, under the FILTER expression  $\geq 10000$ . Companies can use easy calculations like this to drill down to important sales and make compensations accordingly. Since the OrderID is linked to the Order Details table, an employee name or ID can be accessed and go towards determining bonuses or even employee of the month type situations.

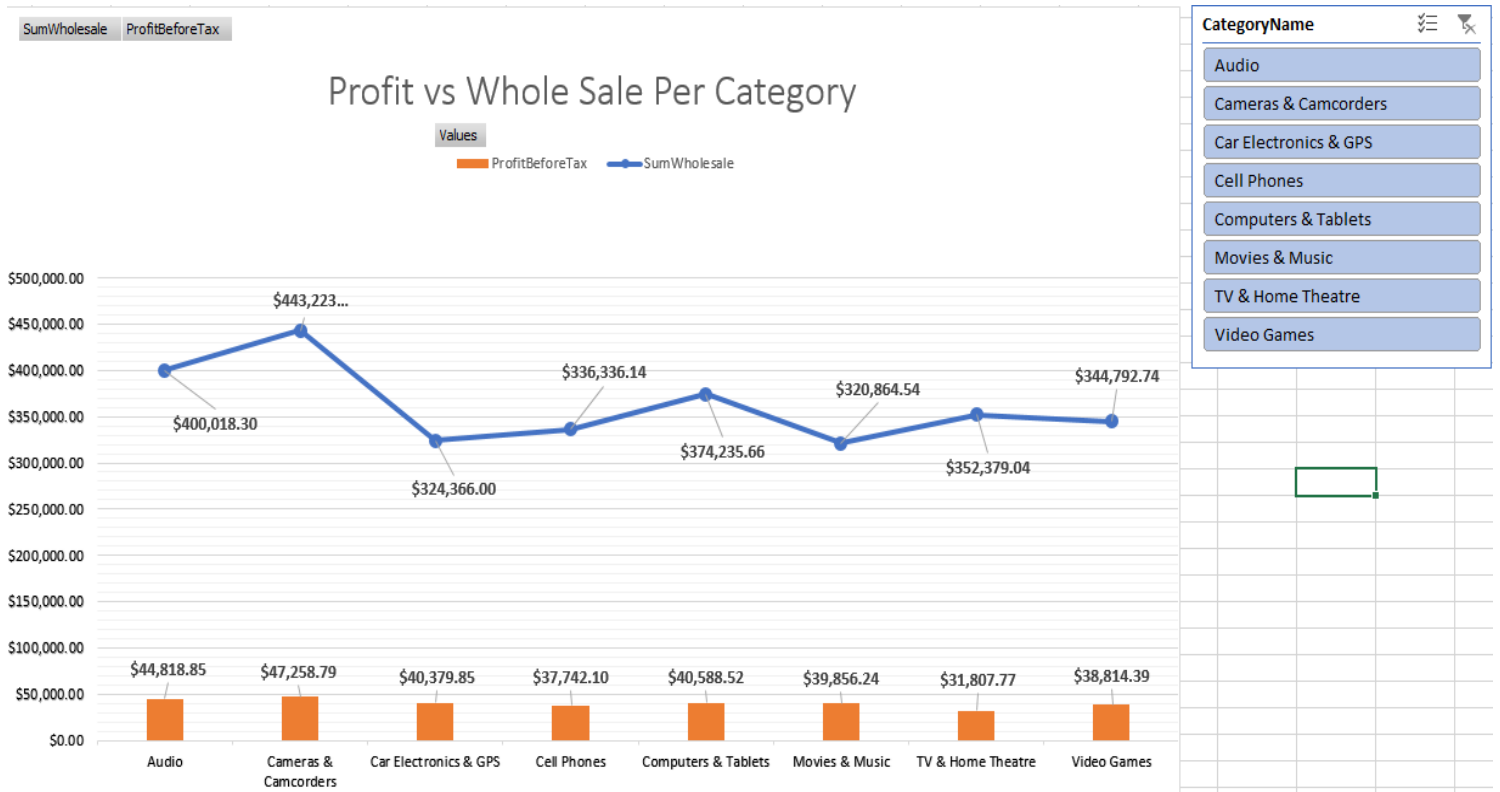


Figure 9: Profit vs. the Whole Sale Per Category

**Data Slicer**

The Data Slicer tool in PowerPivot allowed answering questions such as what sales looked like in each state while also breaking them down by CategoryName. A normal PivotTable is used in conjunction with the Slicer tool and a bar graph that changes every time a user selects a category from the tool. The categories: Audio, Cameras & Camcorders, Car Electronics & GPS, Cell Phones, Computers & Tablets, Movies & Music, TV & Home Theatre, and Video Games appear as buttons in the slicer tool, and the Pivot Table and bar chart adjust to the sales SubtotalPerOrder in each state. Figure 10 below shows all three aspects of this visual aid while Figure 9 above shows the slicer and a graph.

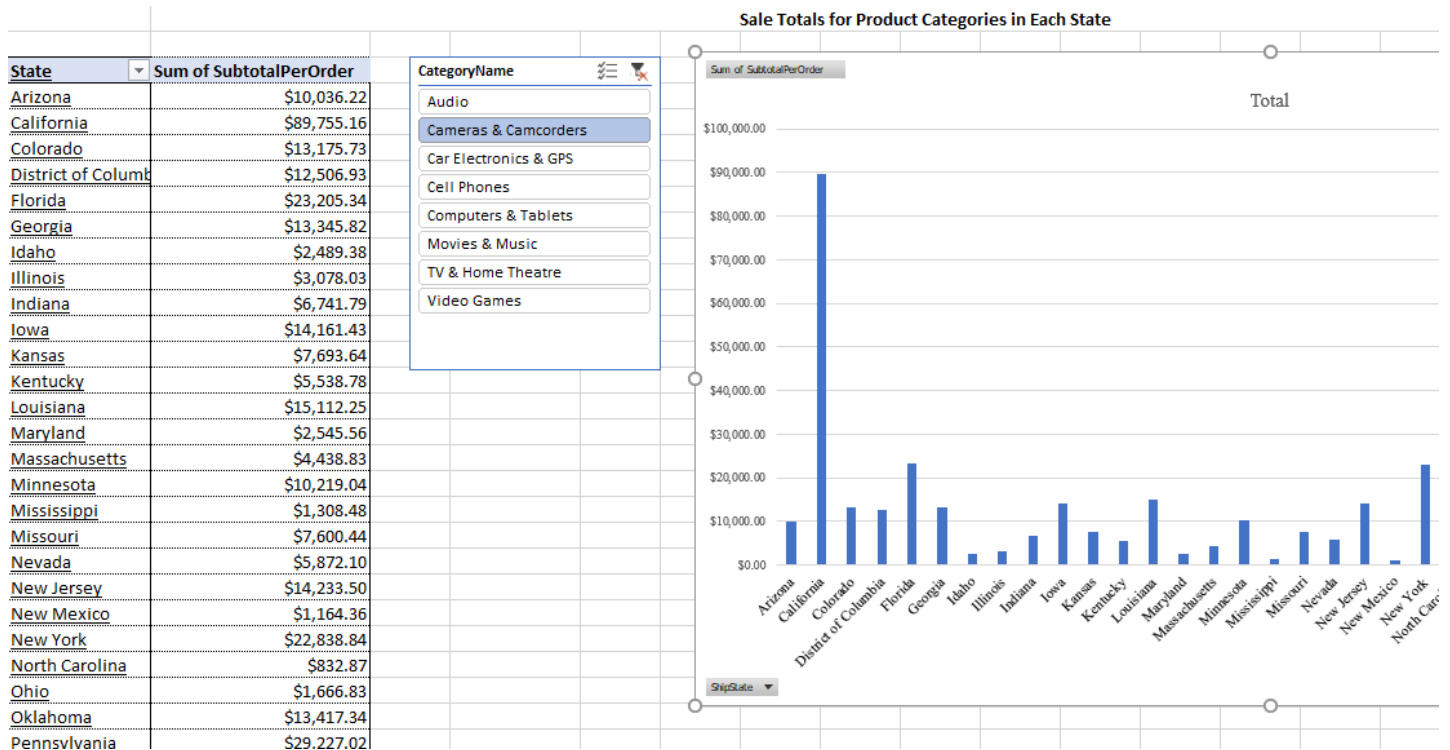


Figure 10: Sale Totals for Product Categories in Each State

### PowerView

PowerView is another Add-In mentioned previously and is a visual aid tool where the measures and Pivot Tables created through the entirety of the project can be implemented to show visual characteristics of the data. Like PivotTables, the relational database can be sifted through to provide important information in sequential graphs spread throughout a map or chart. Figure 11 illustrates the use of PowerView to create a visual for the total sales originating in each state of the United States. Each circle represents at least one sale and the larger the circle, the more sales there are in that state. It is apparent that the most sales were in California, Texas, and Florida based on the graphic. Visuals like this can aid in determining where the most business takes place, but also to evaluate which employee covers the most territory in the United States.

## Sales Totals for Each State Shipped

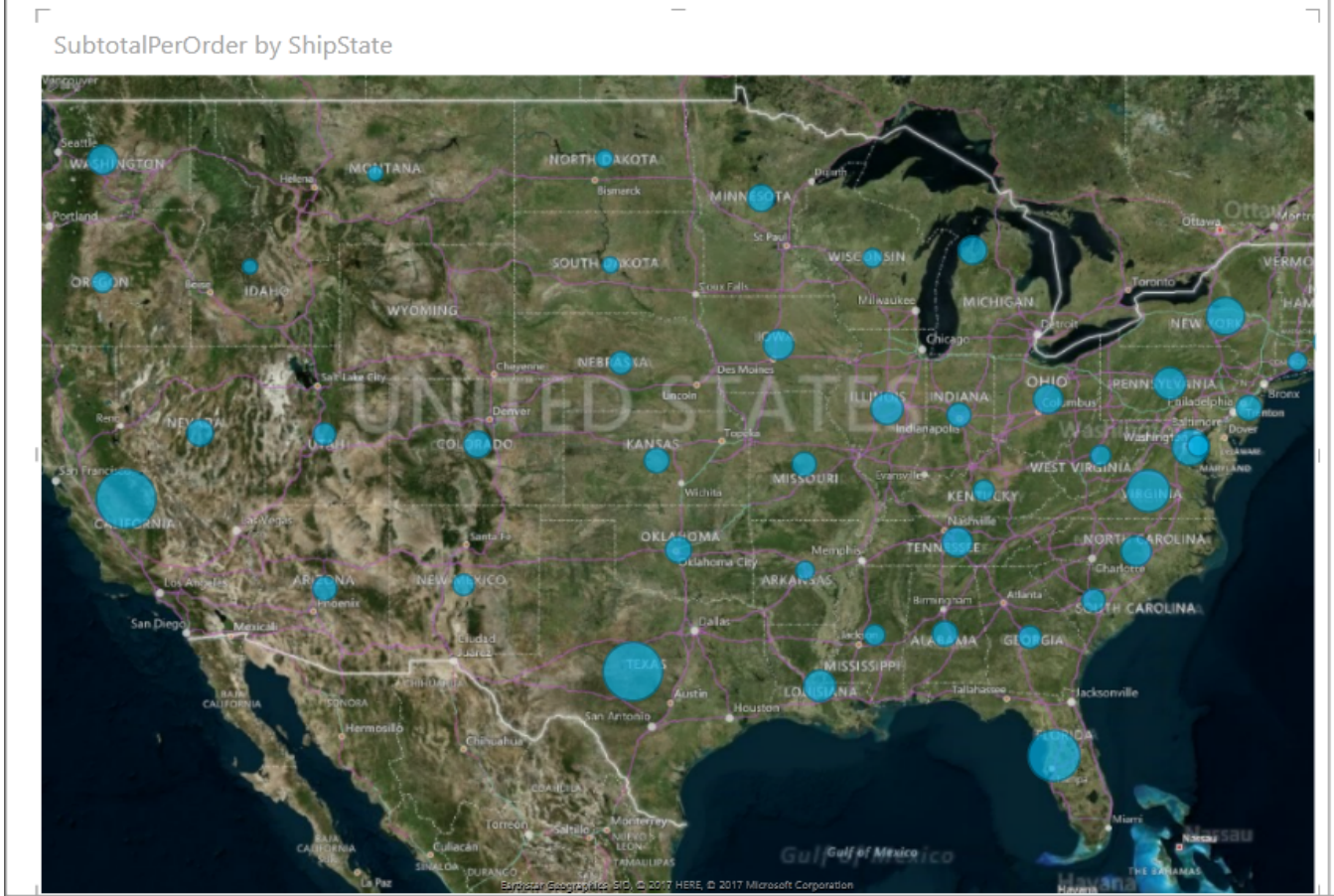


Figure 11: Sales Totals for Each State Shipped

The purpose of the next PowerView visual was to show the country for each supplier along with its contribution to the sales of each category. It is more in-depth than the sales totals for each state shipped since it adds another layer of filtering by adding the CategoryName, linked to the Order Details table by the CategoryID, which adds a pie chart effect to the Power View. Figure 12 shows sales for each supplier divided up by category, where the view can be edited within the software to provide the most customizability possible. The key to the right of the map is automatically constructed when the map option is chosen.

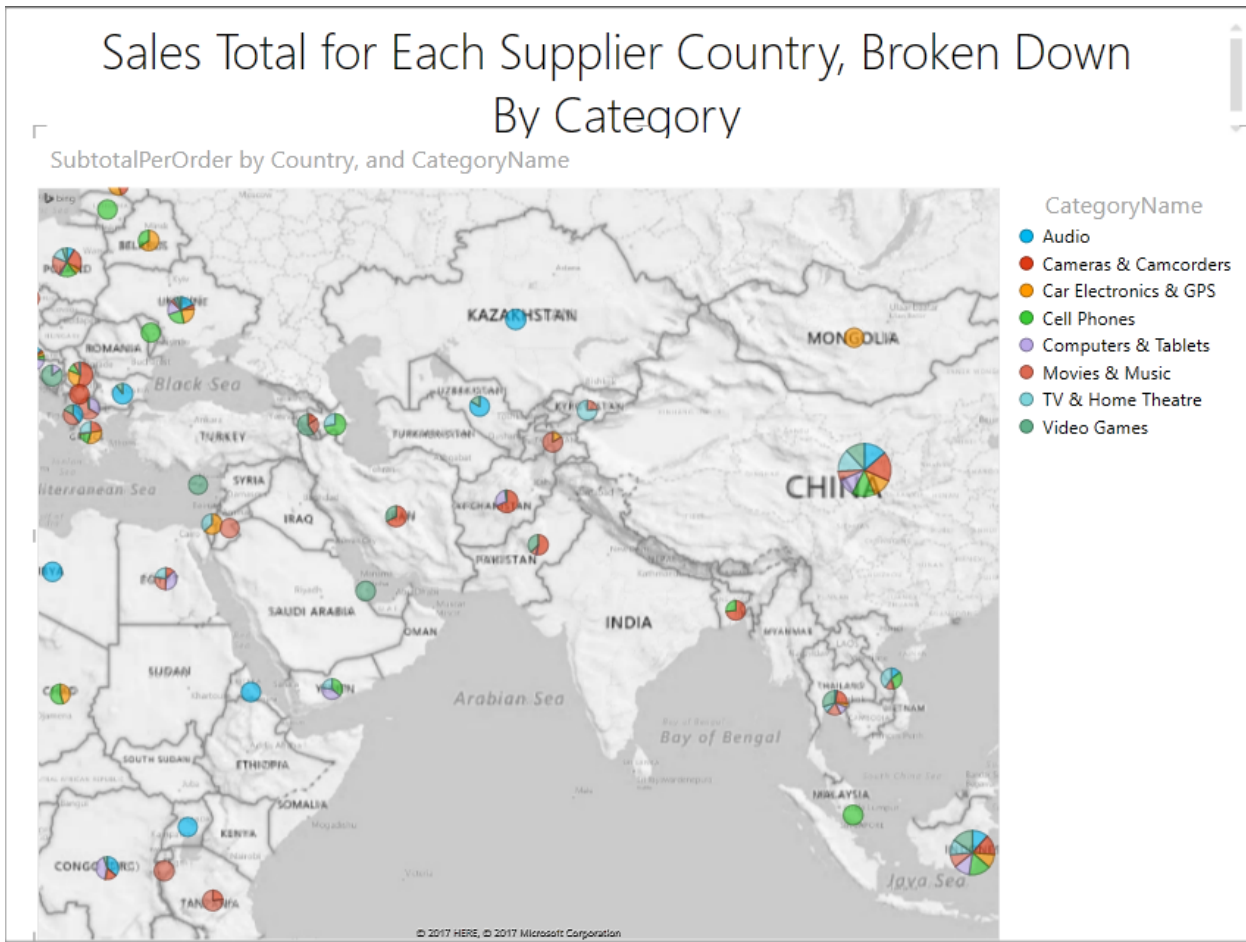


Figure 12: Sales Total for Each Supplier Country by Category

Profit before tax was calculated and displayed visually by implementing both Measures and Power View. A new Measure named ProfitBeforeTax was created using the Calculation Area in the Order Details table that expanded on the original Sum Measure. The Sum Measure calculated the sum of each sale for the sales price, but the ProfitBeforeTax measure used the wholesale price to the company to find the difference. A new calculated field was necessary, “SubtotalWholesale”:

$$=RELATED(Products[UnitPrice]) *(1-'Order Details'[DiscountAmount])*'Order$$

and is only different than SubtotalPerOrder where UnitPrice is used instead of SalePrice.

### Conclusions

PowerPivot and DAX have recently started to get the spotlight from Excel users, and many that were found from reading through the literature review have not had the full awareness of its potential. BI is a hot topic that will stay around for years to come. Businesses are always looking for ways to save both time and money, and the techniques implemented in this suggested class have potential to generate a decent level of understanding toward the application of BI. This difference can be to boost performance or even to help a company that is going under and has no other options.



An educator can use the tools mentioned in this paper to format an Introductory class in BI. A student should be able to learn much in a semester in a course that incorporates the ideas and applications discussed in this paper. There is no question that PowerPivot and DAX are here to stay. Having this type of knowledge and skills in an introductory data analytic or decision-making class will set many students in the right direction.

## References

- Alexander M. (n.d.). *The PowerPivot Internal Data Model*. Retrieved from <http://www.dummies.com/software/microsoft-office/excel/power-pivot-internal-data-model/>
- Association of Business Training (2013). What is a Pivot Table and what are the Benefits of Utilizing Them in Excel? Retrieved from [http://www.associationofbusinessstraining.org/articles/view.php?article\\_id=11022](http://www.associationofbusinessstraining.org/articles/view.php?article_id=11022)
- Charts and other visualizations in Power View(n.d.). Retrieved from <https://support.office.com/en-us/article/charts-and-other-visualizations-in-power-view-141bd462-9853-4973-ac37-842e8345f51e>
- Danyel Fisher, Steven Drucker, and Mary Czerwinski, “Business Intelligence Analytics Guest Editors’ Introduction,” IEEE Computer Graphics and Applications, vol. 34, pp. 22-24, September 2014.
- Examples,” Communications of the ACM, vol. 55, no. 8, pp 97-105, August 2012.
- Jackson J. (2010). *Excel PowerPivot Disrupts Business Intelligence*. Retrieved from [https://www.pcworld.com/article/205260/how\\_microsoft\\_powerpivot\\_will\\_disrupt\\_bi.html](https://www.pcworld.com/article/205260/how_microsoft_powerpivot_will_disrupt_bi.html)
- Michaloudis J. (n.d.). 50 Things You Can Do With Excel Pivot Tables. Available online <https://www.myexcelonline.com/blog/50-things-you-can-do-with-excel-pivot-tables/>
- Microsoft Developer Network (n.d.). *PowerPivot Capacity Specification*. Retrieved from [https://msdn.microsoft.com/en-us/library/gg413465\(v=sql.110\).aspx](https://msdn.microsoft.com/en-us/library/gg413465(v=sql.110).aspx)
- Microsoft, “Data Analysis Expressions (DAX) in Power Pivot.” Retrieved from [https://support.office.com/en-us/article/Data-Analysis-Expressions-DAX-in-Power-Pivot-bab3fbe3-2385-485a-980b-5f64d3b0f730\\_](https://support.office.com/en-us/article/Data-Analysis-Expressions-DAX-in-Power-Pivot-bab3fbe3-2385-485a-980b-5f64d3b0f730_)
- Microsoft, “Filter Functions (DAX)” <https://msdn.microsoft.com/en-us/library/ee634807.aspx>. Accessed November 1, 2017.
- Parke Godfrey, Jarek Gryz, Piotr Lasek (n.d.). "Interactive Visualization of Large Data Sets," IEEE Transactions on Knowledge & Data Engineering, vol. 28, pp. 2142-2157, August 2016
- Stavros Valsamidis et al., “A Proposed Methodology for E-Business Intelligence Measurement Using Data Mining Techniques,” Communications of the ACM, pp. 1-6, October 2014.
- Sumit Gulwani, William R. Harris, and Rishabh Singh, “Spreadsheet Data Manipulation Using
- Surajit Chaudhuri, Umeshwar Dayal, and Vivek Narasayya, “An Overview of Business Intelligence Technology,” Communications of the ACM, Vol. 54, No. 8, pp 88-98, January 2011.
- Vidas Matelis, “List of Power Pivot DAX Functions with Description,” Power Pivot-Info.” Retrieved from <http://www.Power-Pivot-info.com/post/52-list-of-Power-Pivot-dax-functions-with-description>. Accessed October 20, 2017
- [Zhimin Chen, Vivek Narasayya, and Surajit Chaudhuri, “Fast foreign-key detection in Microsoft SQL Server Power Pivot for Excel,” Proceedings of the VLDB Endowment, vol. 7, no. 13, pp. 1417-1428, August 2014.](#)