A Brief History of the Microsoft BI Platform

Ted Pattison  
Critical Path Training  
February 2015

## Abstract

Over the past few years Microsoft has shifted its business intelligence (BI) strategy to focus providing applications, tools and service that embrace the tenants of self-service BI (SSBI). To this end, Microsoft has introduced a new tabular model for data analysis which has many benefits over the older multidimensional model. Microsoft has complimented the new tabular model with a new set of SSBI tools in Microsoft Excel 2013 which include Power Query, PowerPivot, Power View and Power Map. These new tools combined with Excel Services provides a fast and effective way to complete a BI project with shared dashboards and reports in an on-premises environment running SharePoint Server 2013.

In the year 2015, Microsoft is now investing in a cloud-based BI strategy in the Office 365 environment which is known as Power BI. Power BI provides a robust infrastructure which can extract data from just about anywhere on the Internet and load it into a data cache in the Office 365 cloud where it can be used for data analysis and reporting. Microsoft has positioned Excel 2013 as the high-end analysis tool for Power BI because it provides powerful SSBI tools such as Power Query, PowerPivot, Power View and Power Map. However, Power BI also provides a wide reach strategy based on pure HTML5 which makes it possible to create BI dashboards and reports that target any type of browser as well as mobile devices including iPhones, iPads, Android phones and Windows phones.

Contents

[Abstract 1](#_Toc411336359)

[Introduction 2](#_Toc411336360)

[A Tale of Two Models 2](#_Toc411336361)

[Understanding the New Tabular Model 3](#_Toc411336362)

[Choosing Between the Tabular Model and the Multidimensional Model 4](#_Toc411336363)

[A Data Model for Business Users 4](#_Toc411336364)

[Understanding Microsoft's SSBI Tools 5](#_Toc411336365)

[Extracting and Shaping Data with Power Query 5](#_Toc411336366)

[Modeling Data with PowerPivot 6](#_Toc411336367)

[Creating Reports with Power View and Power Map 8](#_Toc411336368)

[Server-side support for Tabular Databases 9](#_Toc411336369)

[PowerPivot for SharePoint 10](#_Toc411336370)

[Power View for SharePoint 11](#_Toc411336371)

[BI Developer Tools versus SSBI Tools 12](#_Toc411336372)

[Extracting and Transforming Data with Visual Studio 12](#_Toc411336373)

[Modeling a Tabular Database with Visual Studio 13](#_Toc411336374)

[Reporting on the Data Model with Visual Studio 14](#_Toc411336375)

[The Road Ahead into Power BI 15](#_Toc411336376)

[Connecting to Data from Power BI 15](#_Toc411336377)

[Publishing Shared Queries in the Data Catalog 16](#_Toc411336378)

[Analyzing and Reporting with Power BI 16](#_Toc411336379)

[Conclusion 18](#_Toc411336380)

[BI Training for Business Users and Developers 18](#_Toc411336381)

# Introduction

Over the last decade, we have witnessed a pretty amazing growth period for SharePoint technologies in which Microsoft made billions of dollars in sales revenue as SharePoint Server found its way into tens of thousands of on-premises environments. So what was the secret of SharePoint's overall success? To a large degree, SharePoint succeeded because it allows simple, everyday business people to create and customize things like sites, lists and libraries and to accomplish a host of other tasks that used to require a professional software developer or someone from the IT department.

There's a new wave in the IT industry that seems to have all the strength of the SharePoint wave back in 2007. This new wave is already shifting the focus of business intelligence (BI) technologies in favor of applications, tools and services that support **self-service BI (SSBI)**. The roots of SSBI can be traced back over the last few years to BI products from smaller software vendors such as Tableau and TIBCO Spotfire. However, the central concepts of SSBI have caught on and are now being fully embraced by larger software and service vendors such as Microsoft, IBM, SAP and Oracle.

SSBI is founded on the notion that an organization should provide an environment that enables business users to become more self-reliant and less dependent on the IT department and developers. For starters, business users require easier access to data for analysis and reporting. Going one step further, business users also require new capabilities to extract data from a variety of data sources and to model this data to fulfill the business needs for data analysis and reporting.

If you have a vested interest in creating successful BI projects on the Microsoft platform, it's essential that you understand what's changed with the recent shift in Microsoft's BI strategy. There are a few key architectural changes and a whole new set of analysis and reporting tools. If you get a jump on learning what's new, you'll have the potential to light up the new SSBI features and to create BI projects that anticipate the next 5 to 10 years of Microsoft's emerging BI technologies.

## A Tale of Two Models

Around the year 2000 at the turn of the millennium, Microsoft and several other vendors began to support the **multidimensional model**. In the multidimensional model, the data to be analyzed is processed and built into multidimensional cubes which is written to persistent storage on a hard drive. That means a business user can execute a BI query against a cube that will retrieve aggregated values that were evaluated ahead of time when the cube was built. This is one of the primary reasons that multidimensional cubes can offer great performance when analyzing a dataset that is very large and complex.

Now imagine you are going back in time to the year 2010 and that your company has just rolled out SharePoint 2010 in an on-premises environment. If your company was following Microsoft best practices at the time, you would be designing and deploying multidimensional cubes with SQL Server Analysis Services (SSAS) to provide a foundation for BI projects as shown in the following diagram.



Once a multidimensional cube has been developed and deployed, it can then be consumed by business users who are able to analyze and report on the data inside it. For example, a business user with Microsoft Excel 2010 can create a connection to a cube and use it to create PivotTables and PivotCharts. Business users can also share their efforts with other team members by publishing their Excel workbooks to Excel Services in SharePoint 2010. SharePoint Server 2010 provides additional BI support for creating dashboards and reports through both PerformancePoint Services and SQL Server Reporting Services (SSRS).

**PerformancePoint Services** is a service application that is part of SharePoint Server 2010 Enterprise Edition and was designed to create dashboards on top of multidimensional cubes. PerformancePoint Services is complemented with the **PerformancePoint Dashboard Designer** which is a click-once desktop application that allows you to create dashboard pages within a SharePoint site with BI objects such as analytic reports, charts, filters, KPIs and scorecards. When you create and deploy a dashboard to a SharePoint site, behind the scenes the PerformancePoint Dashboard Designer is actually creating web part pages on the fly and adding preconfigured web parts to display BI data and visualizations.

Another important BI feature of SharePoint Server 2010 is its integration with **SQL Server Reporting Service (SSRS)**. The SSRS integration in SharePoint Server 2010 is complemented with the **Report Builder** utility, which is another click-once desktop application that allows you to create interactive browser-based reports as well as professional-looking reports that will be sent to a physical printer.

One common question that arises is "who is the intended audience for the PerformancePoint Dashboard Designer and the SSRS Report Builder utility?" The answer is not overly clear. While some technically-proficient business users have found success with these tools, many other business users have found these tools overwhelming and too difficult to use. This means that much of the work done with PerformancePoint Services and SSRS has required involvement from a company's IT staff or a BI developer, because business users were not able to complete the work on their own.

# Understanding the New Tabular Model

So, what's the big problem with the multidimensional model? In a nutshell, the problem lies in the work required to design and implement a multidimensional cube. For starters, the creation of a multidimensional cube is beyond the reach of business users and usually beyond the reach of a company's IT staff. It requires a technical specialist known as a BI developer. Secondly, projects which require the design and deployment of multidimensional cubes are rarely completed quickly.

Historical evidence tells us that multidimensional modeling projects take months and sometimes years to complete. This causes frustration for BI project owners and other BI stakeholders who want to see the results in days or weeks but are told it will be 3 to 6 months before they will even be able to see the first prototype. And then to add insult to injury, a significant percentage of multidimensional BI projects fail and are decommissioned midstream providing absolutely nothing in return for all the time and money invested.

At the center of Microsoft's SSBI strategy is a new alternative to the multidimensional model. This alternative is known as the **tabular model** which has been designed to be simpler and much easier to use. The tabular model isn't just simpler and easier. It's much faster to use because it holds out the promise of getting a BI project up and running in days or weeks instead of months and years.

Let's cover a few fundamentals about the tabular model. A **tabular database** is the top-level object in the tabular model which will be familiar to people who have worked with relational databases such as SQL Server. That's because a tabular database contains tables and table relationships. Tables in a tabular database consist of rows and columns of read-only data that have been imported from external sources. You can extend the design of a table by adding calculated columns, calculated fields (aka measures), dimensional hierarchies and key performance indicators (KPIs). The following diagram shows an example of how sales data can be modeled using a tabular database.



Microsoft’s implementation of the tabular model is based on a query execution engine known as the **xVelocity engine**. There is both a desktop version and a server-side version of the xVelocity engine. The desktop version runs inside Microsoft Excel 2013 and the server-side version runs in SQL Server Analysis Services (SSAS). This means that a tabular database can scale from a single-user desktop scenario to an enterprise scenario where hundreds or thousands of users are concurrently accessing a tabular database running on a high-performance database server.

From an architectural perspective, the tabular model is quite different from the multidimensional model. In the multidimensional model, all the data is written to persistent storage as the cube is built. It is noteworthy that the multidimensional model was created over 15 years ago when disk space was cheap and RAM was far more expensive than it is today.

The tabular model assumes that desktop computers and servers have enough RAM to load large datasets into memory. In the tabular model, all the data is loaded into RAM and calculated columns are evaluated prior to the execution of any query. That means query execution reads data from memory without any need to retrieve anything from storage. Moreover, table data is stored in a highly-compressed, column-based format which saves space and allows for very large datasets.

## Choosing Between the Tabular Model and the Multidimensional Model

While the tabular model is at the center of Microsoft's ongoing BI investment, Microsoft has stated that it plans to continue supporting the older Multidimensional model for the foreseeable future. That means you must choose one of these two models when starting a new BI project.

So, which model should you choose? Of course, the answer is "it depends" but our recommendation is to start any new BI project with the assumption that you will use the tabular model unless you can justify that you really need the multidimensional model. Always keep in mind that multidimensional modeling projects will take significantly longer and cost more money so make sure you think through all the relevant issues when making this decision because changing models in the middle of a project in many ways means you are starting over.

Neither the tabular model nor the multidimensional model have an all-around performance advantage over the other. In some scenarios the tabular model will outperform the multidimensional model because the query processor doesn’t need to read values from persistent storage. In other scenarios such as calculating aggregated values from millions or billions of rows, the multidimensional model will often outperform the tabular model because the aggregated values are calculated ahead of time when the cube is built.

So what are the most common scenarios in which you can justify starting a new BI project using the older multidimensional model? First, you might have a complex design scenario involving tables with many-to-many relationships which is supported by the multidimensional model but not the tabular model. Secondly, multidimensional cubes are written to disk whereas tabular databases are loaded into memory. Therefore, the multidimensional model can deal with much larger datasets that contain billions or trillions of rows. While the compression of the tabular model makes it possible to load millions of rows into memory, at some point you will start hitting the limitations of what can be loaded into RAM.

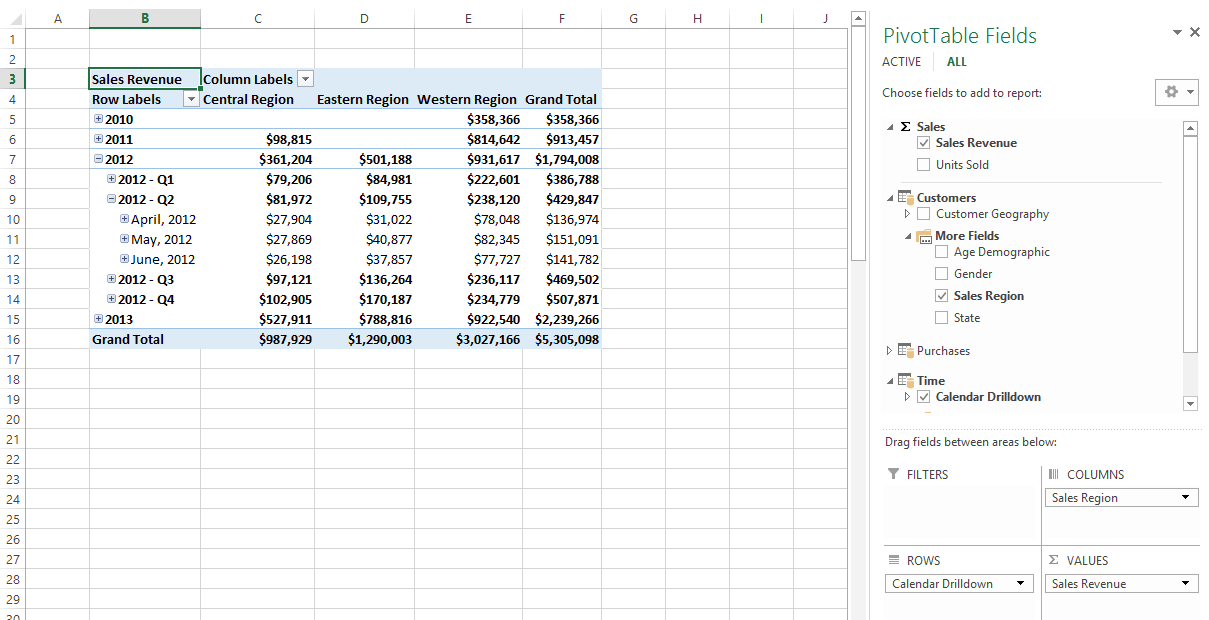
But even in a new BI project where you've decided to use the multidimensional model, it still could make sense to complete your initial prototyping with Excel 2013 using the tabular model. Prototyping with the tabular model will allow you to quickly create a first draft of your data model and to mock up a set of PivotTables and reports that can be shown to project stakeholders to solicit feedback. It will significantly improve your chances for success in a BI project if you're able to have business users validate that you are on the right track in the early stages.

## A Data Model for Business Users

You can get started building your first tabular database using Microsoft Excel 2013. The Excel 2013 workbook file format has been extended to support what's called the **data model**. Behind the scenes, the new Excel data model is really just a tabular database that is persisted inside the Excel workbook file format and then loaded into memory when the workbook is opened. A key aspect of the data model is that its data is separate from any worksheet data in the current workbook as shown in the following diagram.



Once you have created and designed a data model, you can use it to create PivotTables and PivotCharts in worksheets inside the same workbook. The user experience for analyzing the data in a tabular database is similar to analyzing data from a multidimensional cube. The following screenshot shows an example of an Excel PivotTable based on a data model which displays a calculated field (i.e. measure) with total sales revenue broken out by the dimensions of time and customer geography. In this example, the Time table contains a dimensional hierarchy named Calendar Drilldown which allows the user to drill down into more granular units of time.



Keep in mind that a tabular database is not a multidimensional cube. From an architectural perspective, they are quite different. However, there is a potential point of confusion because a tabular database acts as a cube. This support was added to Microsoft's tabular database implementation so that analysis and reporting tools originally created for the multidimensional model are able to connect to a tabular database and query it using MDX.

It is possible to use older tools such as the Report Builder utility provided by SQL Server Reporting Services (SSRS) to create reports based on data in a tabular database. However, you will find that the user experience designing reports is significantly better with a tool like Power View that was designed especially for the tabular model.

# Understanding Microsoft's SSBI Tools

While the Excel data model is natively supported in all new Excel 2013 workbooks, you'll need to activate a few Excel add-ins to take advantage of Microsoft's new SSBI feature set. These new Excel add-ins include Power Query, PowerPivot, Power View and Power Map as shown in the following diagram.



When you install Excel 2013, the add-ins for PowerPivot, Power View and Power Map are installed along with it. While these add-in are installed as part of Excel 2013, they are not activated by default. Therefore, you must activate them before they appear in the Excel ribbon.

The Power Query add-in, on the other hand, is not installed along with Excel 2013. Instead, you must download and install the Power Query add-in separately after you have installed Excel 2013. Despite the fact that Power Query is not part of the Excel 2013 release, you should still consider that Power Query is a core part of Microsoft's SSBI strategy moving forward.

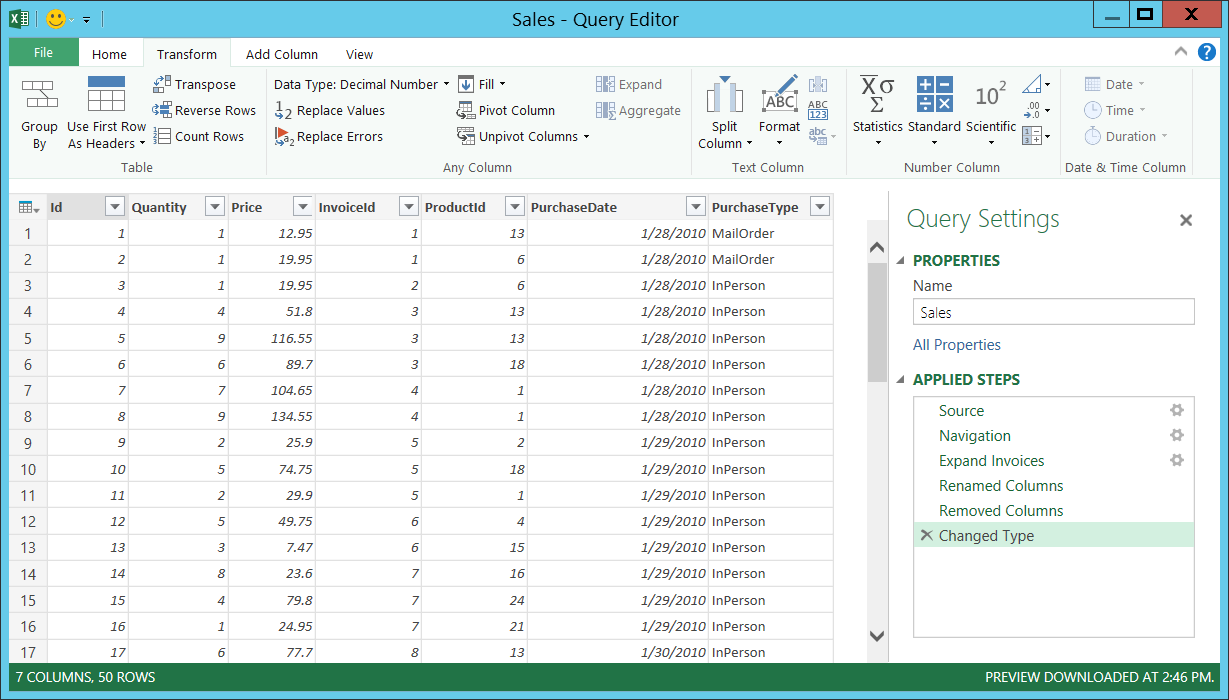
## Extracting and Shaping Data with Power Query

**Power Query** is a SSBI tool for discovering data and extracting it from a wide variety of data sources on a local network or from across the Internet. A key point is that you use Power Query to import data from external source into a worksheet or into the data model of an Excel workbook. Importing data is typically the first step in a BI project because importing data is how you create tables inside the data model.

Power Query supports extracting data directly from common file types such as Excel workbooks, Access databases and CSV files as well as database systems such as SQL Server, Oracle and DB2. It also offers capabilities to extract data from HTML tables in web pages and from public datasets that have been published on the Internet. Power Query provides a valuable productivity layer when it comes to importing data from SharePoint lists. Behind the scenes, Power Query figures out how to extract SharePoint lists by creating OData queries that are executed against the SharePoint REST API.

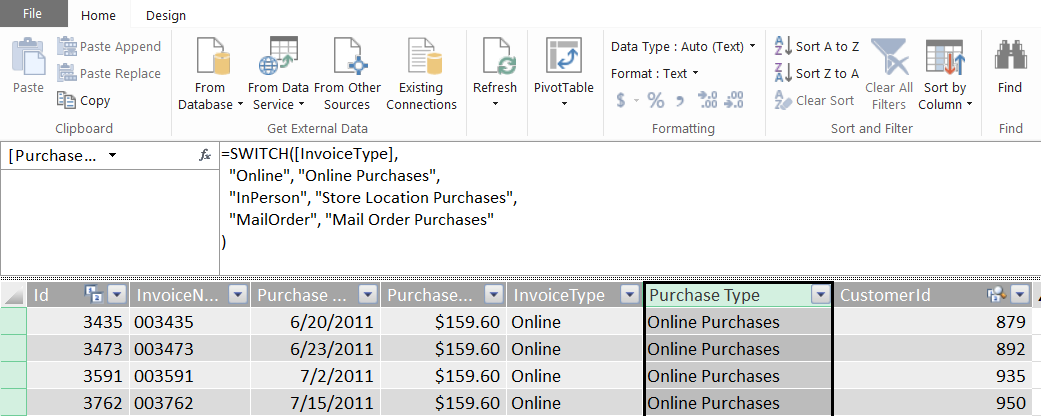
When extracting data from external data sources, Power Query provides the ability to perform transformation operations to shape the data as it is being loaded into Excel. For example, you can split a column with delimited values into multiple columns. You can combine multiple columns into a single column using string parsing or mathematical operations. You can perform table operations such as merging columns and appending rows from multiple data sources. You can also perform powerful transform operations such as transposing, pivoting and un-pivoting the rows and columns in a source table.

When you work with Power Query, you work with a Query Editor to extract the data and to shape it as you import into Excel. A query in Power Query represents a set of steps (i.e. extract and transform operations) that are applied on data that has been extracted from an external data source. A query is saved inside the Excel workbook and can be executed to import data and then to refresh that data on demand. The following screenshot shows an example of working with the Query Editor and demonstrates how this tools allows you to see the data are you are shaping before you import it into the data model.

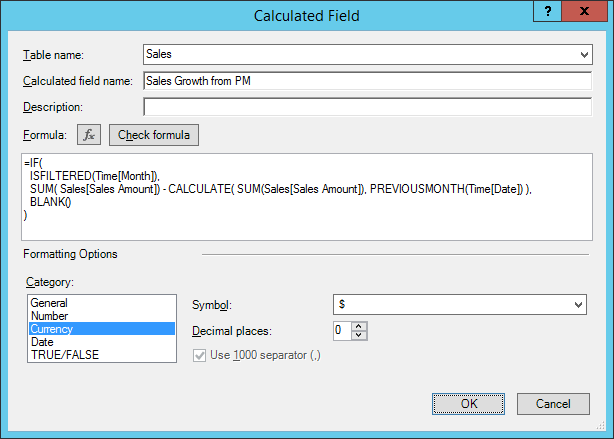


## Modeling Data with PowerPivot

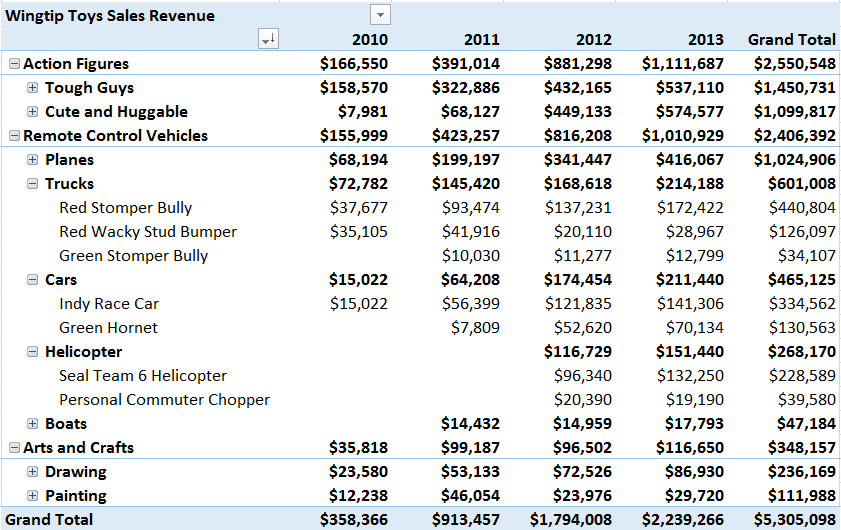
Once you have imported data and created tables inside the data model, the next step is to begin the data modeling phase. This is where the PowerPivot add-in comes in. **PowerPivot** is a design tool that allows you to see and work with the tables in the data model inside a specific Excel workbook. The PowerPivot window provides you with a design view to view and work with the tables and table relationships that exist in the current data model. PowerPivot also provides a tabular view for each table which allows you to add new calculated columns using Dimensional Expression language (DAX) as shown in the following diagram.



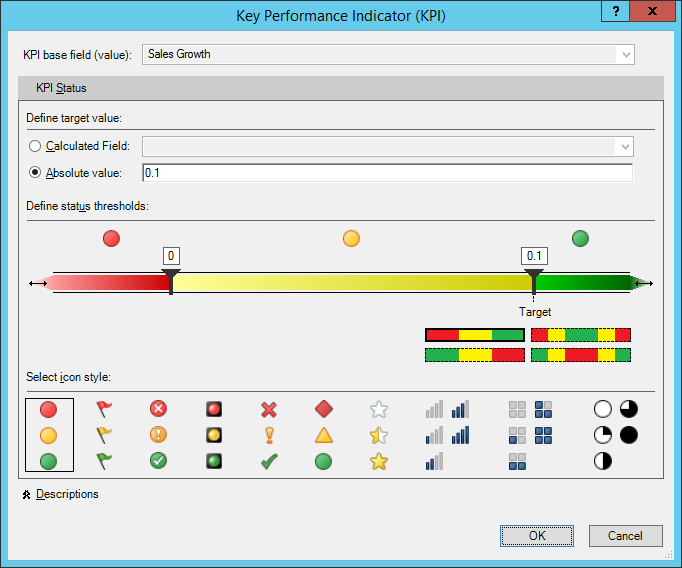
When it comes to creating calculated fields, the Excel application window provides a convenient option in the PowerPivot tab of the ribbon. If you select the **New Calculated Field**… button, you will be presented with a dialog box that allows you to create and validate a calculated field using DAX. When creating a calculated field in this manner you also have control over the format of the output as shown in the following screenshot.



When you are designing a data model, you can extend tables by adding dimensional hierarchies. The purpose of a **dimensional hierarchy** is to provide the user with drill-down capabilities within a specific dimension. For example, the Customers table can be designed with a Customer Geography hierarchy that enables users to drill down into smaller geographical regions when analyzing and reporting on sales data. The Products table can be designed with a Product Category hierarchy enabling the user to drill down on sales revenue figures from product category to subcategory and all the way down to the individual product totals as shown in the PivotTable in the following screenshot.



The tabular model also supports the familiar BI concept of key performance indicators (KPIs). A **key performance indicator (KPI)** in the tabular model is really just a visualization layered on top of a calculated field. This visualization often comes in the form of a stop light analogy displaying a green light, a yellow light or a red light to indicate a measure of health or performance. The PowerPivot add-in provides a special dialog for creating a KPI from an existing calculated field as shown in the following screenshot.



Once you have used PowerPivot to design a data model, you can simplify the model for other business users by renaming or hiding tables and columns. If a data model has significant complexity or it has a large number of tables and columns, you can also create a simplified view of the data model known as a **perspective**. The purpose of a perspective is to allow the designer of a data model to provide a simplified view with a subset of tables, columns and fields for less technical business users who are creating reports and dashboards.

Microsoft provides a version of the PowerPivot add-in for Excel 2013 and a second version for Excel 2010. However, it's recommended that a company migrates to Excel 2013 before beginning BI project work involving data models and the PowerPivot add-in. While it is possible to complete a BI project using the PowerPivot add-in in Excel 2010, there are some issues you should be aware of.

First, Excel 2010 does not include the PowerPivot add-in so it must be separately downloaded and installed. Second, Excel 2010 does not natively support Excel workbooks which contain a data model. This means users will experience errors when attempting to open a workbook with a data model unless they have already installed and activated the PowerPivot add-in. Finally, there are low-level changes between versions which can make a data model created with Excel 2010 inaccessible to users running Excel 2013. If this is the case, you have to go through a conversion process to upgrade the data model before it can be consumed by business users running Excel 2013 and SharePoint Server 2013.

## Creating Reports with Power View and Power Map

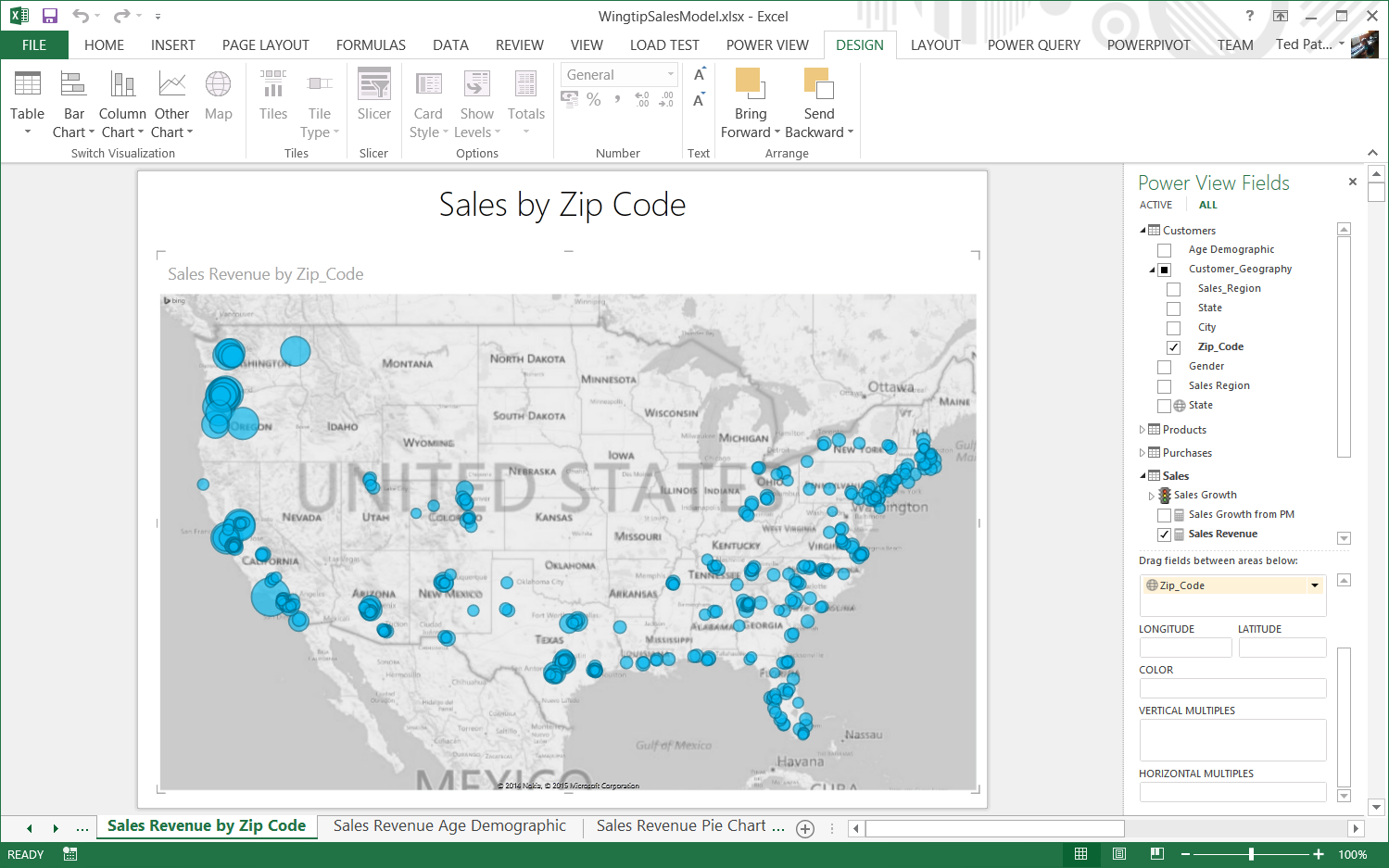
Once the data modeling process is complete, it's time to put the data model to use. You've already seen that Excel PivotTables and PivotCharts provide an excellent way to analyze the data inside the data model. Excel 2013 provides two additional SSBI add-ins for reporting which are Power View and Power Map.

The **Power View** add-in for Excel 2013 is a tool that allows business users to create reports and BI visualizations on top of a tabular data model. When you create a new Power View report in Excel 2013, the reportq is added to the Excel workbook and appears on the bottom tab along with the worksheets. The Power View report designer provides a Fields list in the task pane on the right which is very similar to working with the fields list when working with an Excel PivotTable. You design a Power View report by adding tables and other BI visualizations to a canvas as shown in the following screenshot.



Another potential point of confusion exists because Microsoft has released two different versions of Power View. The one you've just seen is the Power View add-in that loads into Excel 2013 on the desktop which saves its reports inside an Excel workbook file. The other version of Power View is supported on the server side by SharePoint Server 2010 and SharePoint Server 2013. When using the Power View with SharePoint Server, the business user designing a report uses a browser-based version of Power View to connect to a tabular database. Also, reports created using Power View for SharePoint are saved as standalone files in a SharePoint document library with a file extension of .RDLX. While the two different versions of Power View provide a similar user experience for designing reports, there are some obvious differences because one is a browser-based tool and the other is not.

Power View is complimented by another Excel add-in which known as Power Map. **Power Map** provides the ability to visualize data spread out across geographic locations when creating a Power View report. For example, imagine you have dimensional customer data such as country, state or zip code that can be translated to a specific point in longitude and latitude. Power Map interacts with the Bing location service behind the scenes to translate dimensional data to geographical coordinates. This allows Power Map to produce reports which visualize how BI data is spread out across a geographic distribution.



It's important to remember that the current version of Power View and Power Map have a dependency on Microsoft Silverlight. That means business users working on a BI project must install Silverlight on their computers before they can begin to design BI reports in Excel 2013 using Power View or Power Map. Moreover, the requirement to install Silverlight also applies to any business user who is just viewing Power View reports using Excel 2013. This also applies to a business user that is viewing a Power View report rendered through the browser by Excel Services in SharePoint Server 2013 in an on-premises environment. Later in this whitepaper, you will hear more about Microsoft's ongoing effort to remove these dependencies from their BI reporting layer.

Now that you know about the tabular model and the new SSBI tools from Microsoft, one thing should be clear: You can build an entire BI solution using nothing more than Excel 2013. For small and medium-sized BI projects, Excel 2013 and the SSBI add-ins might be all you need. Even in larger BI projects, the use of Excel 2013 is still the recommended way to get started by creating prototypes.

In an afternoon, an experienced BI specialist can create an Excel 2013 workbook with a prototype for a data model and an associated set of PivotTables and reports. Other project stakeholders can test out the prototyping simply by opening this workbook in Excel 2013. There is no need for the prototype workbook to connect to any backend system because the data model is self-contained in the Excel workbook file format. This is an example of where the tabular model really shines over the multidimensional model in terms of how fast project stakeholders can be shown a prototype and provide feedback to validate what they see in the prototype meets their requirements for data analysis and reporting.

# Server-side support for Tabular Databases

As you have seen, an Excel workbook file can contain a data model and a set of associated PivotTables, charts and reports designed to analyze and report on the data inside. In a simple desktop scenario where a user opens a workbook in Excel 2013 containing a data model, the data inside the data model and its calculation logic loads into memory on the user's computer. This allows Excel 2013 to execute queries against the data model locally without any external dependencies.

Things become more complicated when you upload an Excel workbook with a data model to a SharePoint document library and then you view that workbook using the browser. In this scenario, Excel Services must be able to execute queries against the data model over on the server side. To understand how this works from an architectural perspective, you must learn about the integration between Excel Services and SQL Server Analysis Services (SSAS). But first, let's quickly review the features offered by Excel Services prior to the release of SharePoint Server 2013.

Excel Services is a SharePoint service application that was first introduced with SharePoint Server 2007. The initial release of Excel Services in 2007 added BI support for publishing and browser-enabled Excel workbooks. Moreover, browser-enabled workbooks have the ability to connect to backend data sources such as relational databases and multidimensional cubes. Therefore, Excel Services since its initial release in 2007 has had the ability to connect to pre-existing databases over on the server side and to execute queries in response to user actions such as drilling down into greater detail within a PivotTable built using a multidimensional cube.

With SharePoint Server 2013, Excel Services has been extended to support Excel workbooks containing a data model. The new data model support in Excel Services is made possible by integrating an on-premises farm running SharePoint Server 2013 with an instance of SQL Server Analysis Services (SSAS) running in SharePoint mode. A SSAS instance running in SharePoint mode is what provides Excel Services with the server-side support for creating a tabular database and executing queries against it.

Once a SharePoint farm administrator has configured Excel Services to use a specific SSAS instance running in SharePoint mode, Excel Services then has the ability to communicate with the SSAS instance in order to create a new tabular database on demand. In the scenario where a business user uploads an Excel workbook with a data model, Excel Services creates a server-side copy of this data model. It does this by creating a new tabular database in the SSAS instance and then copying all the data and logic from the data model in the workbook over to the copy of the data model running in SSAS. The workbook is then updated to contain an external connection to the new server-side data model as shown in the following diagram.



Once the data model has been published as a new tabular database in the SSAS instance, Excel Services can query it on demand in response to user actions in the browser. When a user interacts with a PivotTable or a Power View report, Excel Service is able to execute queries against the server-side version of the data model to provide drilldown capabilities.

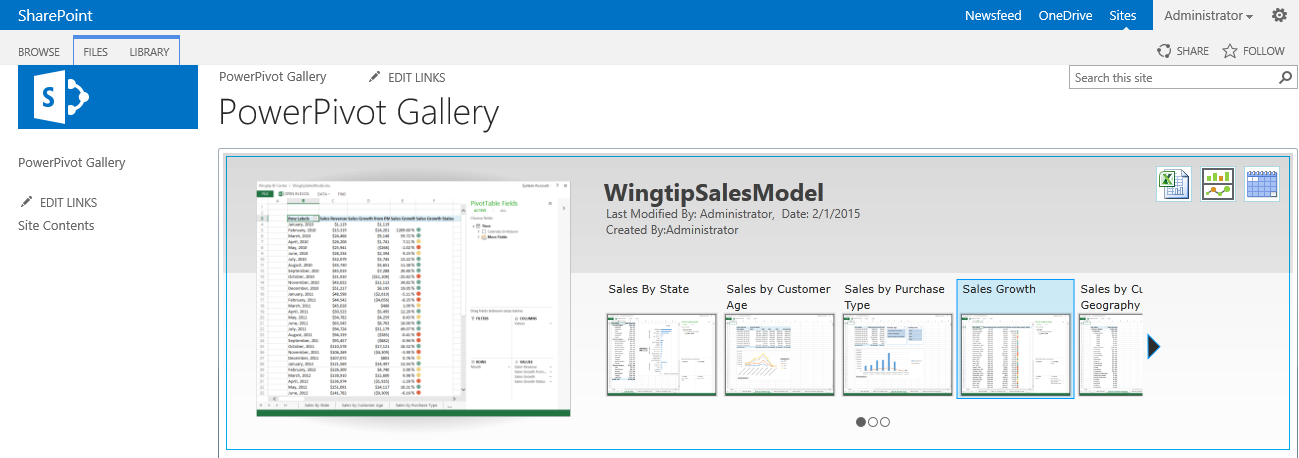
Once you have published a workbook with a data model using Excel Services, you can then create secondary workbooks and Power View reports that connect to it using a special type of SharePoint connection is known as a **BISM connection**. When you configure a BISM connection, you can use the URL of the workbook itself as it exists in the SharePoint document library. SharePoint 2013 and Excel Services can figure out how to translate that URL into the information it needs to connect to the tabular database in the SSAS instance.

## PowerPivot for SharePoint

As you have seen, Excel Services in SharePoint 2013 provides the underlying support to publish and query a server-side version of the data model. However, you can complement this support in Excel Services with additional BI support by installing an optional SharePoint component known as Power Pivot for SharePoint 2013.

**Power Pivot for SharePoint 2013** is a SharePoint add-in that can be installed in a SharePoint Server 2013 on-premises farm. When you install this add-in, it creates a new SharePoint service application which provides a monitoring capability in Central Administration through the **PowerPivot Management Dashboard**. The PowerPivot Management Dashboard makes it possible for the IT staff to monitor and report on query execution performance between Excel Services and the SSAS instance. The installation of the PowerPivot for SharePoint 2013 add-in also install a SharePoint feature with the title **PowerPivot Feature Integration for Site Collections**. You can activate this feature within a site collection to add a document library template which makes it possible to create a special type of document library known as the PowerPivot Gallery.

The **Power Pivot Gallery** is a SharePoint document libraries that was designed to upload Excel workbooks that contain BI elements such as a data model, PivotTables and PivotCharts. When you upload an Excel workbook to the PowerPivot Gallery, SharePoint inspects the content of the workbook in order to create a thumbnail image for each of the worksheets inside. Once the thumbnail images have been created, the PowerPivot Gallery is able to show the user a view of the worksheets inside the workbook using a user interface experience that is powered by Microsoft Silverlight as shown in the following diagram.

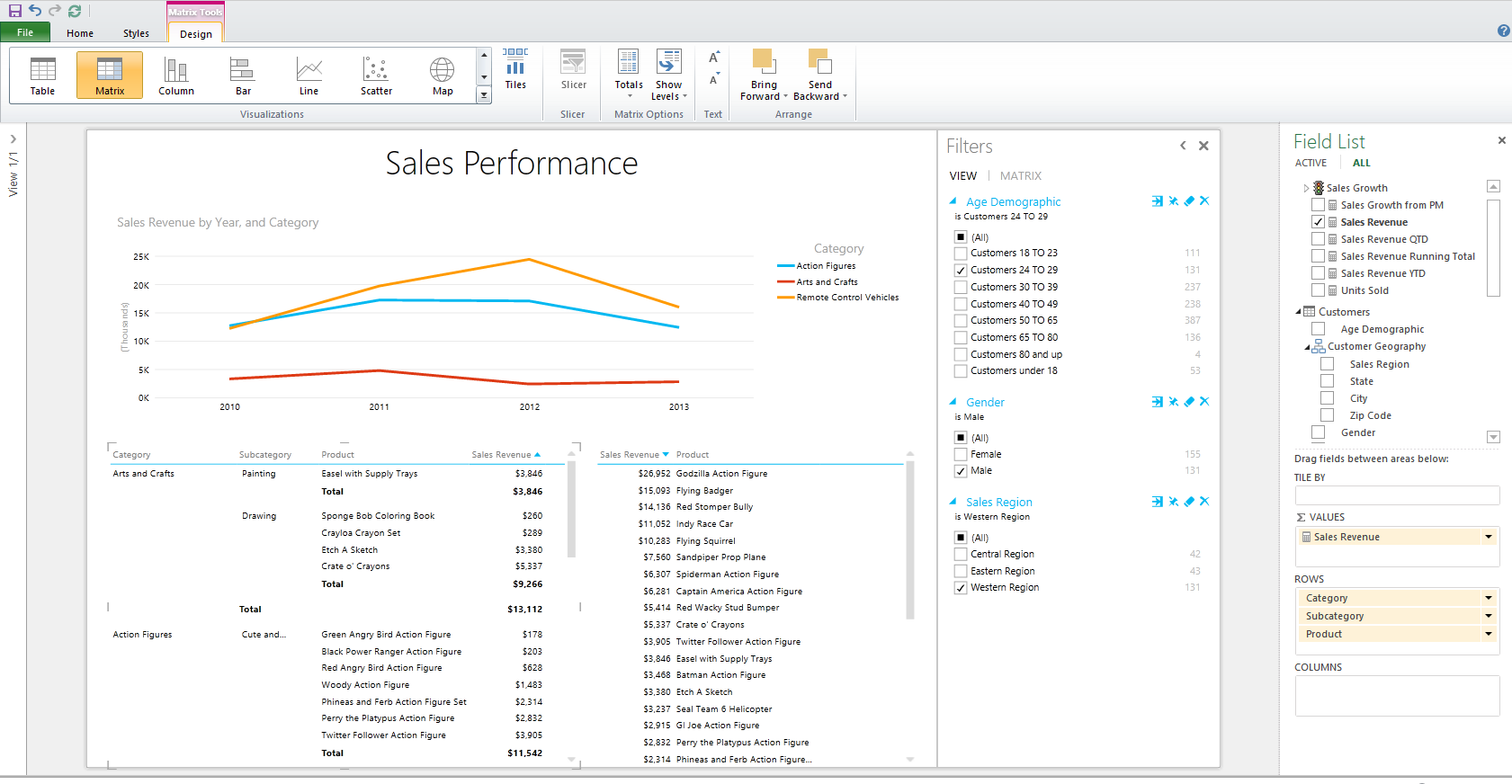


When you examine the PowerPivot Gallery view for an Excel workbook containing a data model, you will see three buttons in the upper right hand corner. The first button with the Excel file icon allows you to quickly create a new Excel workbook. The second button with the chart icon allows you to quickly create a new Power View report. The convenience of these two buttons is that the new Excel workbook or the new Power View report is created with an existing BISM connection to the data model associated with the current worksheet.

The third button on the far right with the calendar image provides support for refreshing the data inside the data model. When you click this button, you are redirected to a special page that allows you to configure data refresh operations. For example, you can schedule an automated job that runs daily or weekly and refreshes the data in the data model. If the data model has multiple data sources, you can schedule refresh settings for each data source individually, including configuring which credentials that will be used against the external data sources when running the refresh operations.

## Power View for SharePoint

You have already seen how you can create Power View reports using the Power View add-in in Excel 2013. Now it's time to look into the other version of Power View which runs in SharePoint. **Power View for SharePoint** provides a browser-based environment for both report authoring and report viewing. The following screenshot shows an example of using Power View for SharePoint to design a report from a data model published using an Excel workbook.



To get Power View for SharePoint up and running in an on-premises farm, you must begin by installing the PowerPivot for SharePoint add-in. Next, you must install and configure SQL Server Reporting Services (SSRS) in SharePoint mode on one of the web servers within the SharePoint farm. Finally, you must install the SQL Server Reporting Services add-in for SharePoint 2013. Once you have gone through these configuration steps, you should then be able to create a Power View report in the browser using the button with the chart icon in the PowerPivot Gallery.

# BI Developer Tools versus SSBI Tools

For many BI projects, all the work required to create a data model and to design an associated set of PivotTables, charts and reports can be completed using the SSBI tools available inside Excel 2013. This is true even when you need to publish a data model with Excel Services and to provide a shared dashboard of worksheets and reports using the PowerPivot Gallery. However, there are certain scenarios where the SSBI tools alone will not cut it. That's when you need help from a set of BI developer tools available in Visual Studio known as the **SQL Server Developer Tools (SSDT)**.

There are three different phases in a BI project where you might be required to use the SSDT. First, you might need them in the data extraction phase. Second, you might need them in the data modeling phase. Third, you might need them in the reporting phase. The following diagrams shows how each of these three phases in a BI project maps to both the SSBI tools and to the BI developer tools available when using SSDT.



## Extracting and Transforming Data with Visual Studio

If you are using the SSBI tools, then you use Power Query to extract and transform the data which is loaded into the data model in an Excel workbook. However, you might encounter a scenario in which you must extract a really large dataset such as a table with a billion rows. A set of data can grow so large that it is both unreliable and impractical to import it with a tool like Power Query that runs on the desktop.

In BI projects in which you must analyze large datasets, it is common to create a database that plays the role of a data warehouse. The key concept is that you extract the data from wherever it lives and then stage it in SQL Server by loading it into tables in a SQL Server database that plays the role of a data warehouse. When extracting data and loading it into a data warehouse, it's a common practice to perform various types of transforms to reshape the data so that it is better suited for analysis and reporting. Among BI developers, this phase of a BI project is often referred to as the Extract-Transform-Load (ETL) phase.

You can create a data warehouse using a standard SQL Server database. The first important task is to design a set of tables for the data warehouse where the data is to be loaded. After that you can begin to leverage the SQL Server features to extract, transform and load data into a data warehouse. These ETL features are provided through **SQL Server Integration Services (SSIS)**. The way that you leverage these ETL features in a BI project is to create a new **Integration Services project** in Visual Studio using the SQL Server Data Tools (SSDT).

When you work with an Integration Services project in Visual Studio, you complete your work in terms of **SSIS packages**. Visual Studio provides a visual designer for developing SSIS packages which you can use to define ETL elements such as data flows, transformations and control of flow logic. When you are developing an SSIS package, Visual Studio makes it relatively simple to test and debug it using a developer instance of SSIS. When you have completed your testing and debugging, Visual Studio is able to generate an SSIS package as output that can be deployed to a production server running SSIS.

Designing and deploying a data warehouse in SQL Server is not a trivial undertaking. It's going to add a good deal of extra work to any BI project so you have to justify when it's worth the extra cost and time. If you are developing a multidimensional database, you usually don't have a choice. You need to create a data warehouse using a SQL Server database because SQL Server Analysis Services (SSAS) can only load data for cubes from tables in SQL Server. In other words, you must stage the data in SQL Server before you can load it into a cube. However, the tabular model provides more flexibility because you can pull the data directly into a tabular database from anywhere it lives.

When you are working on a BI project with the tabular model, you have the option to create a data warehouse in scenarios where it makes sense. In many BI projects which involve SSBI tools, creating a data warehouse will be overkill. However, the creation of a data warehouse might be justified in larger BI projects even when you are modeling your data with a tabular database.

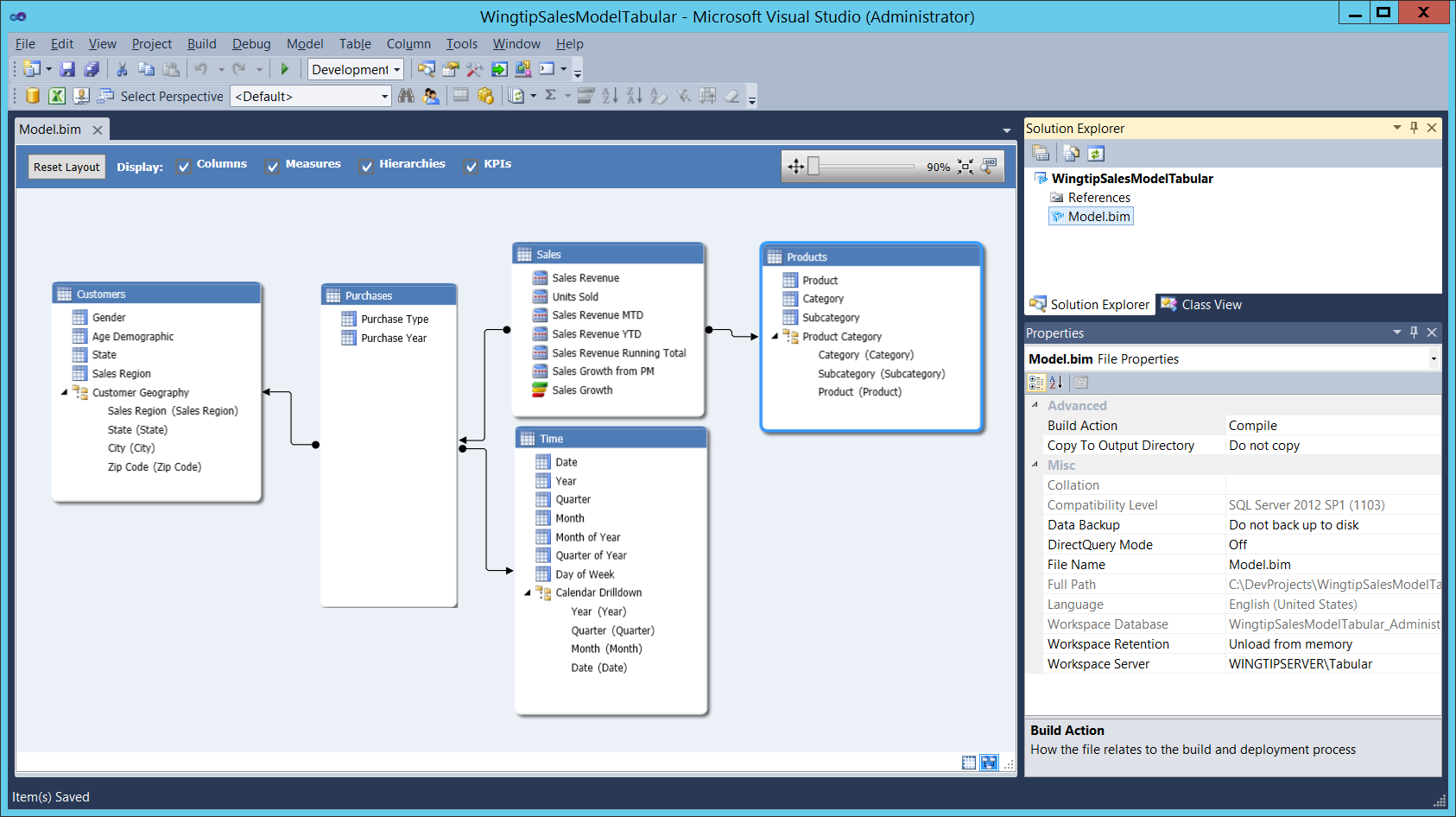
It might be worthwhile to create a data warehouse in a BI project which deals with large datasets or external data sources that take several hours to return their data. There are also security benefits because consumers of the data warehouse don't need to deal with the security credentials for external data sources. Additionally, you can configure permissions to the tables in the data warehouse to enforce a custom security policy across a set of Active Directory users and groups.

A final motivation for creating a data warehouse has to do with producing BI datasets that are trustworthy and reliable. In an environment where business users are extracting data from multiple places around a corporate LAN in an ad-hoc fashion, it can be hard to have confidence that all the data used in reports is accurate and complete. While the creation of a data warehouse can be expensive, it also builds a degree of governance into a BI project to ensure that the underlying data is trustworthy and can be used with confidence to create accurate dashboards and reports.

## Modeling a Tabular Database with Visual Studio

When you are working with Excel 2013, you use the PowerPivot add-in to create a data model inside an Excel workbook. As a BI developer, you can use a different approach to create a tabular database with a data model directly in SQL Server Analysis Services (SSAS). You accomplish this by creating a new **Analysis Services Tabular project** in Visual Studio using the SQL Server Developer Tools (SSDT).

When you are working on an Analysis Services Tabular project, Visual Studio provides a visual design environment for designing the data model that is similar to the PowerPivot add-in window in Excel 2013. For example, Visual Studio provides a tabular view for each table in which you can add calculated columns and calculated fields using Data Access Expression language (DAX). Visual Studio also provides a diagram view of the data model showing tables and table relationships as shown in the following screenshot.



In general, you will find that most of the techniques for designing the data model are the same whether you are working with the PowerPivot add-in in Excel 2013 or in an Analysis Services Tabular project in Visual Studio. You can use either tool to successfully create a complex data model which includes calculated columns, calculated fields, dimensional hierarchies and KPIs. So that brings up an important question. If both tools support a similar data modeling experience, when should you design a tabular database (i.e. a data model) using Visual Studio instead of using the PowerPivot add-in in Excel 2013?

First of all, creating an Analysis Services Tabular project in Visual Studio makes it possible to deploy a tabular database directly to SSAS without having to go through SharePoint Server 2013 or Excel Services. This opens up opportunities and features that are not available when publishing a data model using Microsoft's SSBI tools. For example, you can load a really large data model (e.g. billions of rows) with an Analysis Services Tabular project that would be too large to publish through Excel Services using Excel 2013. There are also several tabular database features in SSAS that are not available when publishing a data model using Excel Services.

The first tabular database feature that is only accessible when using Visual Studio is **DirectQuery mode**. DirectQuery mode is very different than the standard query mode of a tabular database where the data has been cached in memory after it's been extracted from an external source. While standard query mode uses cached data which results in faster query execution, it has a notable drawback. Standard query mode uses a copy of the data which might get out of sync with the data source from where the data was extracted. With DirectQuery mode, SSAS continues to query and fetch data from external data sources instead of using cached data in RAM.

The benefit of DirectQuery mode is that queries are run against live data as opposed to a cached copy of the data. This means that queries return real-time results containing the latest updates. The drawback to DirectQuery mode is that it has degraded performance because SSAS must retrieve and load external data each time a query is executed. Because of this performance degradation, the scenarios in which you can effectively use DirectQuery mode are fairly limited. It only makes sense to use DirectQuery mode in a project which requires real time results against a live database which is small and fast enough to return its data back to SSAS in an acceptable window of time.

The second feature that is only accessible through Visual Studio development is the ability to define **Security Roles** within the context of a tabular database. By creating a Security Role in an Analysis Services project, you can effectively define a set of users that have access to only certain tables within the data model. The way you accomplish this is by configuring a Security Role with a **row filter**. A row filter is created by writing a DAX expression to define a subset of tables, columns and rows that members of the Security Role are able to view and analyze.

The third feature that is only accessible through Visual Studio development is the ability to use **Table** **Partitions**. Table partitions are used to optimize the loading and refreshing of data within the tables inside the data model. For example, imagine your data model has a large table with sales data collected over a 10 year period. This table could be so big that it takes several hours to load or to refresh. What if you need to refresh the data so that you can get the latest sales data from the last week? Without partitions, you must refresh all the data in the entire table which is going to take a long time. When you use partitions, you can break the table up into logic sections that can be refreshed independently of one another. For example, you could create a partition for each year. Then it would only be necessary to refresh the partition for the current year because the partitions from earlier years do not contain any data that is changing.

You have just seen several scenarios in which you might prefer creating and designing a tabular database with an Analysis Services Tabular project instead of using the PowerPivot add-in in Excel 2013. For example, Visual Studio development will give you a greater ability to scale to larger datasets and allow you to utilize SSAS features such as DirectQuery mode, Security Roles and Table Partitions.

Now consider a scenario in which you have already created a complex data model using the PowerPivot add-in with Excel 2013 and then you decide that you would like to take advantage of a SSAS features such as security Roles or Table Partitions. There is good news because you do not have to start the new Analysis Service Tabular project from scratch. That's because Visual Studio makes it possible to create a new Analysis Service Tabular project by importing an Excel workbook which contains an existing data model. This technique works well in a scenario where a business user creates the first draft of a data model in Excel 2013 and then gets feedback from other project stakeholder to ensure that the BI project is on track. After that point, a developer can take over and leverage this work by importing the existing data model into a new Analysis Service Tabular project where it's possible to leverage the SSAS features that are only available through Visual Studio.

If the need arises, you can also use the SQL Server Data Tools (SSDT) to develop a multidimensional database containing cubes by creating an **Analysis Service Multidimensional and Data Mining project**. While modeling OLAP data using the new tabular model is much easier and faster, you might still encounter a scenario involving datasets so large that the multidimensional model is a better choice. The bad news here is that it is going to take a while to get up to speed with multidimensional database development. The good news is that BI developers have been developing multidimensional databases for quite a few years so it's not hard to find good books and educational resources to climb up the learning curve.

## Reporting on the Data Model with Visual Studio

If you want to use Visual Studio to create reports for a BI project, you can use the SQL Server Data Tools (SSDT) to create a **Report Server project**. A Report Server project makes it possible to create and deploy reports to an instance of SQL Server Reporting Services (SSRS). If you are working with a SharePoint Server 2013 on-premises farm which has been configured to use a SSRS instance running in SharePoint mode, you can use a Report Server project to deploy reports directly to a SharePoint document library. This approach allows business users to discover and view SSRS reports within the context of a SharePoint site.

You develop a Report Server project by adding data sources, datasets and reports. You can create data sources based on SQL Server databases, SharePoint lists, and cubes inside a multidimensional database. Many companies have found that using Visual Studio or the SSRS Report Builder utility to create and deploy SSRS reports to a SharePoint document library provides a great way to provide a reporting layer on top of SharePoint lists, SQL Server databases and multidimensional cubes. The SSRS integration with SharePoint Server 2013 provides additional features to automate scheduling the generation and publishing of SSRS reports as well as sending notifications to users when new reports are ready for viewing.

As of February 2015, the support in Visual Studio for developing SSRS reports provided by the SQL Server Developer Tools (SSDT) has not yet been updated to embrace the new tabular model. While you can create a data source in a Project Server project based on a tabular database, Visual Studio recognizes it as a multidimensional cube instead of as a tabular database. For example, Visual Studio will create queries for the tabular database using MDX instead of DAX. While you can create SSRS reports with data from a tabular database using a Report Server project, you also might elect to use a more modern reporting tool that has been designed to embrace the tabular model.

You hopefully remember from earlier that the Power View reporting tools in Excel 2013 and SharePoint Server 2013 were designed to embrace the new tabular model. The key observation here is that Microsoft's current set of SSBI reporting tools that are more up to date than the reporting tools in Visual Studio for developers. In a BI project in which you are required to use the SQL Server Developer Tools (SSDT) for the ETL phase and/or the data modeling phase, you should still consider using Power View for the reporting phase.

# The Road Ahead into Power BI

It's no secret that Microsoft is trying to move all its customers into the cloud. You've probably noticed Microsoft investing in strategies that help them to move customers currently using SharePoint Server on-premises over to Office 365 and SharePoint Online. When it comes to Microsoft's evolving BI strategy, their interest in pushing customers into the cloud and Office 365 is equally as strong.

Microsoft's ongoing investment in their BI platform is currently focused on making their cloud-based BI offering in Office 365 as strong as possible. In February 2014, Microsoft released the first generation of their cloud-based BI platform under the name of **Power BI for Office 365**. In January of 2015, Microsoft released the second generation of this new BI platform in a beta preview program under the shortened name of **Power BI**.

At its core, **Power BI** is a business analytics service which fits into the cloud-based computing model of Software-as-a-Service (SaaS). Power BI has been designed to support clients using Microsoft Excel as well as various types of thin clients. The driving vision of Power BI is to foster an environment where business users can quickly create reports and dashboards and share them across a wide range of browsers and devices such as tablets and mobile phones.

## Connecting to Data from Power BI

A valuable feature of the Power BI service is its ability to extract data from just about anywhere on the Internet and load it into a server-side cache. This means you can avoid pulling large amounts of data from external sources into a desktop tool such as Excel 2013. Instead, you have robust and scalable server-side support to pull data from wherever it lives into a central location inside the Power BI service.

Power BI provides the **Power BI Admin Center** site for IT administrators which provides pages for monitoring system health and for configuring gateways and data sources. The main idea is that a technical person such as an IT professional can configure gateways and data sources which can then be used by less technical business users. Power BI uses gateways to create trusted connections to on-premises data inside a corporate LAN such as standard SQL Server databases as well as tabular databases and multidimensional databases running in SQL Server Analysis Services (SSAS).

The gateway configuration in the Power BI service must be complemented by client software agents known as the **Data Management Gateway** and the **Analysis Services Connector**. One or both of these client software agents must be installed on computers running inside the on-premises environment. These agents facilitate connectivity to the Power BI service allowing an organization to maintain all its live business data and OLAP data models inside their corporate firewall and then to just rely on Power BI for dashboards and reporting.

The vision of Power BI goes far beyond just connecting to corporate databases in an on-premises environment. The architecture of the Power BI service provides the potential to extract data from many new cloud-based data sources. The following diagram shows the wide variety of data sources that can be used to load data into the Power BI service.



Power BI provides the ability to connect to data sources in Office 365 such as Active Directory and Exchange. This allows you to analyze datasets specific to a group of Active Directory users or mail message in an Exchange inbox. For example, you can analyze your inbox to see who has sent you the greatest number of messages over the last year. You could also analyze what percentage of messages that you sent in the morning hours compared to afternoon hours and evening hours.

Power BI also supports connecting to a growing number of the software industry's most popular SaaS applications such as SalesForce, Dynamics CRM, Zendesk, Marketo, SendGrid, GitHub and FaceBook. There is powerful synergy between Power BI and popular SaaS applications in which the data schema has been standardized across all customers. When you connect Power BI to a SaaS application such as Salesforce, Power BI already knows the underlying table schema and is able to generate a set of reports and dashboards showing the most common types of data analysis that business users want to see. This is an area where Microsoft can continue to add value over time by adding support for new SaaS applications and enriching the support to generate reports and dashboards.

Another data source that has interesting potential with Power BI is **Microsoft Azure Stream Analytics**. Stream Analytics is a cloud-based event processing engine that broadcasts streams of real-time data from a wide variety of devices, instruments and sensors. For example, Stream Analytics can provide Power BI with a stream which constantly updates the geographical location for each truck in a large fleet of trucks. Power BI can provide a dashboard with a map showing where each truck is located within a geographical region. In another example, Stream Analytics could provide Power BI with a stream of temperature readings for all the refrigerators for a large supermarket chain. By analyzing this stream of data using Power BI, you could create dashboards and reports to discover problems such as which refrigerators were not able to produce consistent temperature readings across a period of time.

## Publishing Shared Queries in the Data Catalog

As you learned earlier, Power Query is Microsoft's new SSBI tool for extracting and transforming data. Power Query is especially powerful in the cloud because it allows you to shape the data as you are pulling it into the Power BI service. For example, you can write powerful expressions in Power Query to generate calculated columns from existing columns. You can merge columns and append rows from multiple data sources into a single table. If you are working with a table in which you do not like the layout of rows and columns, you can perform a table operation such as transpose, pivot or unpivot.

Once you have created a query with Power Query, you can then published it into a shared repository in Power BI known as the **Data Catalog**. The Data Catalog fosters collaboration because it allows business users with Power Query skills to share their efforts with less technical users. Microsoft has coined the term **Data Steward** to describe an individual who uses Power Query to create shared queries for other users. Once a Data Steward has created a set of queries, less-technical business users can discover them in the Data Catalog through search and use them to create dashboards and reports.

One critical aspect of a BI project is validating that the underlying data is reliable and trustworthy. After all, a BI report is only as accurate as the data used to create it. Therefore, a BI project should include some form of governance that is used to validate the data that will be used to create dashboards and reports. To this end, Power BI supports the idea of certifying a shared query that's been added to the Data Catalog. By publishing a certified query, a Data Steward is vouching that it will return data that is trustworthy and reliable. By using certified queries, a company can enforce a governance policy on the underlying data while still embracing the main tenants of SSBI.

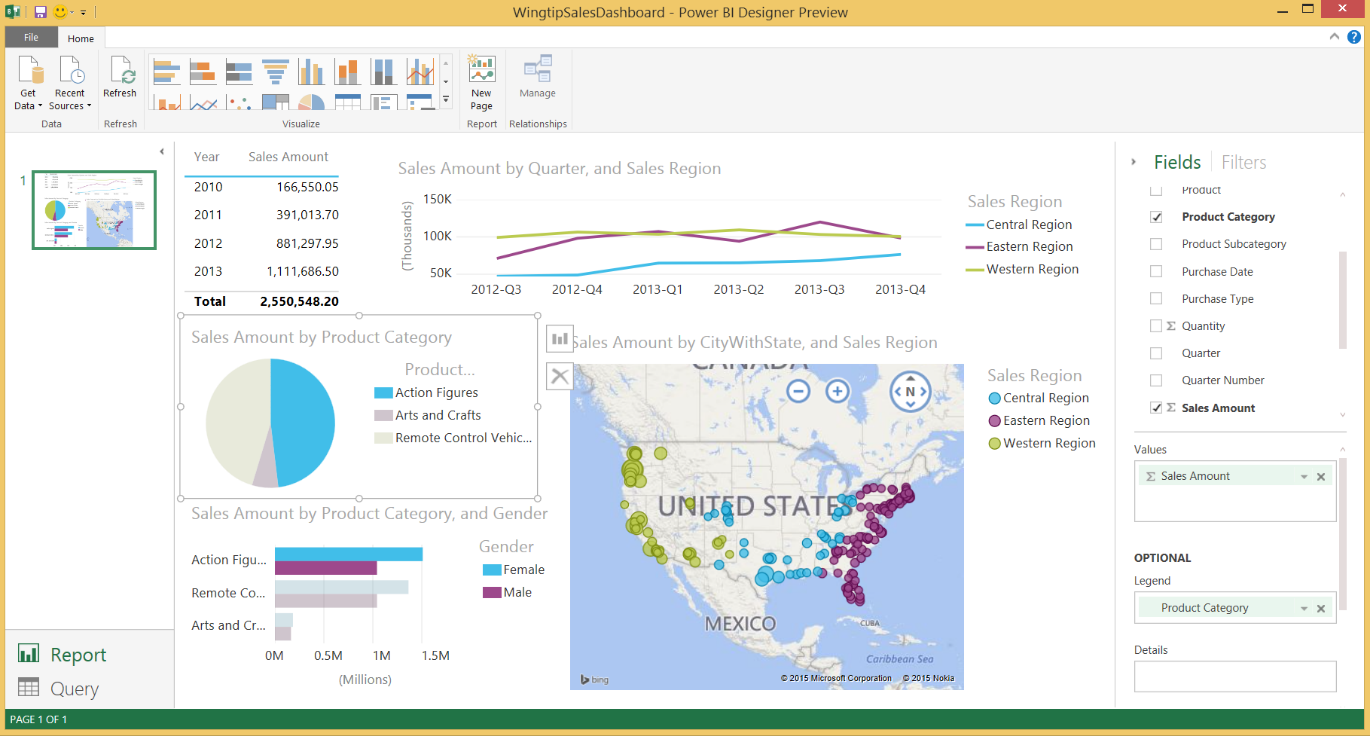
## Analyzing and Reporting with Power BI

Microsoft Excel 2013 is currently positioned as the high-end tool for data analysts using Power BI. A business user working with Excel 2013 can discover and execute shared queries from the Data Catalog to import data into an Excel workbook for analysis using PivotTables and PivotCharts. An Excel user with PowerPivot skills can go a step further by importing data into the data model of an Excel workbook where it can be extended with table relationships, calculated columns, calculated fields, dimensional hierarchies and KPIs. Power View and Power Map complete the SSBI story allowing an Excel user to add reports into the same Excel workbook that contains the data model.

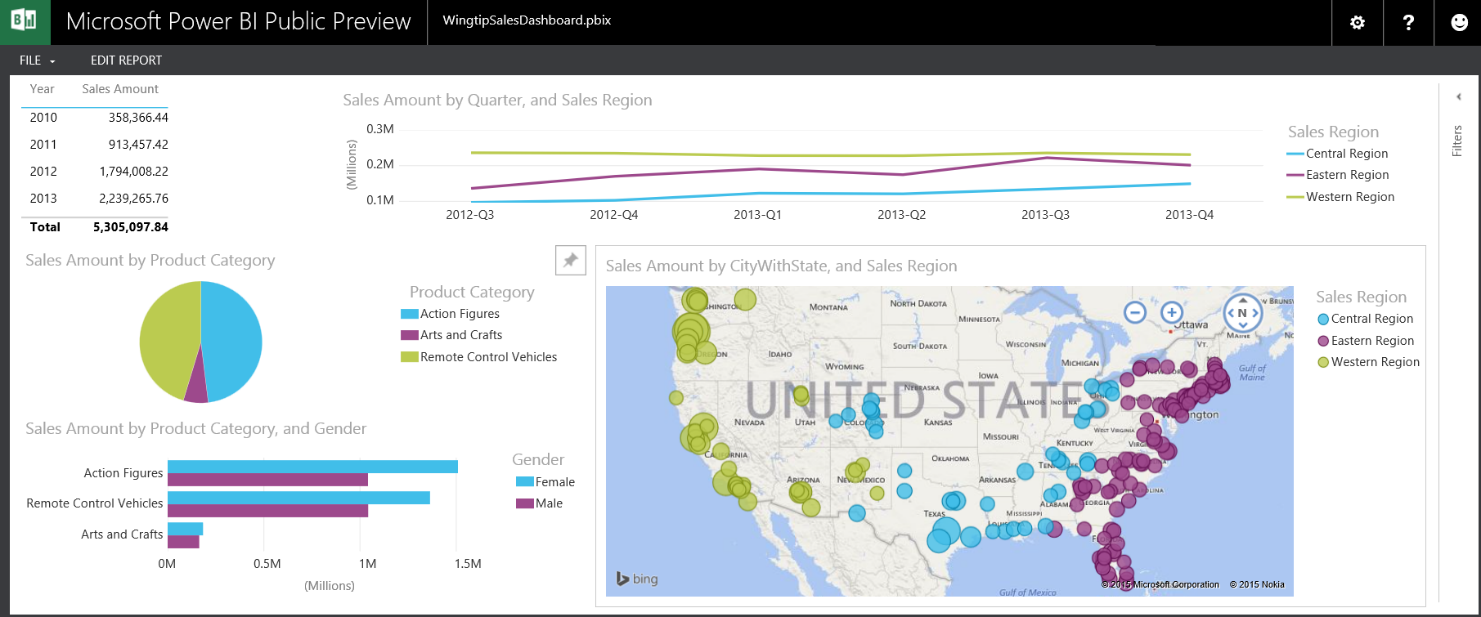
While Excel 2013 provides the richest BI tools for data analysis in Power BI, it's not the only game in town. Power BI also supports creating reports and dashboards which target browser-based clients and mobile devices. To achieve this goal, Microsoft has designed a new type of web site for Power BI in which they have removed all dependencies on Microsoft Silverlight from their BI reporting layer. This means that the reports and dashboards created in a Power BI site are created using pure HTML5, which extend their reach across many different types of devices.

When working with a Power BI site, you can create reports and dashboards using any type of browser. Alternatively, you can create reports and dashboards using Microsoft's new SSBI tool named the **Power BI Designer**. The Power BI Designer is a simplified tool which allows business users to query data and to create reports and dashboards in a Power BI site.

If you have prior experience designing queries with Power Query, the Power BI Designer will be instantly familiar because it provides the same ribbon and the same set of powerful query tools as the Power Query add-in in Excel 2013. When working with the Power BI Designer, you first create a query and then you create a report on top of the dataset returned by the query. When it comes to designing reports, the Power BI Designer is somewhat similar but less complicated than working with the Power View and Power Map add-ins in Excel 2013. The following screenshot shows an example of the report design environment supplied by the preview version of the Power BI Designer.

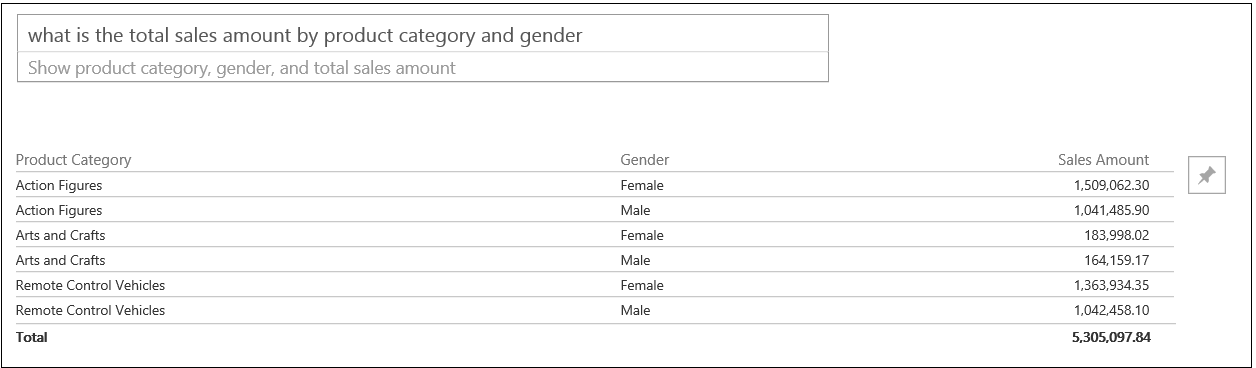


Once you have created a set of queries and reports with the Power BI Designer, you can publish your work to a Power BI site. This is a two-step process with the preview version of the Power BI Designer. First, you must save your work as a Power BI Designer file with a .pbix extension. Next, you upload the Power BI Designer file to your Power BI site to generate datasets, reports and a dashboard. Next, you share the dashboard and reports to make it available to other users. The following screenshot shows an example of what a Power BI dashboard looks like from the perspective of a browser-based client.



You can access dashboards and reports in a Power BI site using just about any type of modern browser. Microsoft is also committed to making Power BI sites accessible to a growing number of tablets and mobile phones. To this end Microsoft has already published Power BI apps for iPhones and for iPads in the Apple store. Microsoft is currently at work developing additional Power BI apps to add support for other popular mobile devices such as Android phones and tablets and the Windows phone.

Power BI provides an interesting new feature known as **Natural Language Q&A**. The idea is that a business user can create a query by asking a simple questions such as "what is the total sales amount by product category and gender?" The Natural Language Q&A search engine is able to interpret the question and transform it into a query behind the scenes as shown in the following screenshot. Once you have returned data using the Natural Language Q&A features, it's easy to pin this data to a dashboard.



# Conclusion

Over the past few years Microsoft has shifted its BI strategy to focus on the tenants of self-service BI (SSBI). Microsoft's introduction of the tabular model and the new SSBI tools in Excel 2013 have changed the way that companies are approaching BI projects. When you combine the SSBI tools in Excel 2013 with the BI support in Excel Services for PowerPivot and Power View, you have the potential to complete BI projects in a SharePoint Server 2013 environment in a much shorter time period when compared to complete a BI project using an older approach.

While Microsoft's strategy for creating BI projects in a SharePoint Server 2013 on-premises environment is fairly complete, their strategy for providing BI features in the cloud with Office 365 is just beginning to evolve. You can expect to see heavy investments from Microsoft and many enhancements to the Power BI platform over the next few years. However, you can be confident that learning about the new tabular model and the new SSBI tools in Excel 2013 today will position you to be ready for what's coming next.

## BI Training for Business Users and Developers

At Critical Path Training, we see an amazing amount of potential in Microsoft's emerging BI platform. To that end, we have created a BI curriculum to accommodate both business users and BI developers. If you are familiar with our SharePoint training curriculum, you can expect the same level of depth and real-world experience in our BI training.

Critical Path Training offers a BI course designed for business users titled: [**SBI2013: Building BI Solutions Using Excel and SharePoint 2013**](https://www.criticalpathtraining.com/sharepoint-courses/building-bi-solutions-using-excel-sharepoint-2013/). This 3-day course introduces key concepts of business intelligence, tabular databases and the new SSBI tools available in Excel 2013. When attending this course you will gain plenty of hands-on experience by completing lab exercises designed to get you up to speed with the new SBBI tools in Excel 2013 including Power Query, PowerPivot, Power View and Power Map. This course also teaches business users how to collaborate in a SharePoint 2013 on-premises environment by publishing BI dashboards and reports using Excel Services and the PowerPivot Gallery. The final lecture of this course examines the emerging Power BI platform and discusses strategies for migrating a BI project into the cloud.

Critical Path Training offers a BI training for professional developers through our new Infusion seminar titled [**Infusion-BI: Developing BI Solutions Using SQL Server, SharePoint and Excel**](https://www.criticalpathtraining.com/sharepoint-courses/developing-bi-solutions-using-sql-server-sharepoint-excel/). This intensive 2-day seminar begins with a quick primer on the SSBI tools in Excel 2013 and then dives into developer-oriented details of when and how to create BI projects using the SQL Server Developer Tools (SSDT) in Visual Studio. You will learn the fundamentals of designing a data warehouse in SQL Server and performing ETL operations to load data using Visual Studio and SQL Server Integration Services (SSIS). You will learn how to develop and deploy tabular databases using Visual Studio which take advantage of SSAS features such as DirectQuery mode, Security Roles and Table Partitions. We'll even get you started with developing cubes in a multidimensional database for those BI projects that require higher levels of complexity or scalability.