

THE EXPERT'S VOICE® IN BUSINESS INTELLIGENCE

Power Query for Power BI and Excel

*BOOST YOUR BUSINESS
INTELLIGENCE WITH DATA
FROM ANY SOURCE, ANYTIME*



Chris Webb

Apress®

For your convenience Apress has placed some of the front matter material after the index. Please use the Bookmarks and Contents at a Glance links to access them.



Apress®

Contents at a Glance

About the Author	xv
About the Technical Reviewer	xvii
Acknowledgments	xix
Preface	xxi
■ Chapter 1: Introducing Power Query	1
■ Chapter 2: Power Query Data Sources.....	17
■ Chapter 3: Transforming Data with Power Query	63
■ Chapter 4: Data Destinations.....	99
■ Chapter 5: Introduction to M	115
■ Chapter 6: Working with Multiple Queries.....	149
■ Chapter 7: Power Query and Power BI for Office 365.....	167
■ Chapter 8: Power Query Recipes	189
Index.....	243

CHAPTER 1



Introducing Power Query

This book is for people who spend a lot of time working with Excel building reports and dashboards. More specifically, this book is for people who work with Excel building reports and dashboards and who are bored with copying and pasting data into worksheets, bored with clicking the same sequence of buttons every month to clean and shape that data, and bored with fixing the problems associated with complex formulas, dirty data, and the errors that are inevitable when you have to follow the same procedures over and over and over again. The good news is that Power Query is here to free you from these dull, repetitive tasks and give you time to concentrate on what's important: analyzing your data and gaining insights from it. Even better, Power Query is easy to use and lots of fun to learn (so long as you're the kind of person who thinks that playing with data can be fun—no need to feel ashamed if you do) and, as a result, it will make you more efficient, more productive, and, hopefully, less bored.

Power Query and Power BI

Power Query is an Excel add-in developed by Microsoft, and its purpose is to make it easier to load data into Excel from external data sources. It's part of the Power BI suite of tools and, as such, it's just one tool out of many that you can use when developing what are called “self-service Business Intelligence solutions” in Excel. What is “self-service Business Intelligence?” I hear you ask. Well, it's just a fancy term for those reports and dashboards that you build as part of your job. The aim of a Business Intelligence (usually shortened to “BI”) solution is to make business information accessible to people so that they can use it to make informed decisions about how to do their job; other terms you may have heard include “management information systems” and “decision support systems”. The whole process is referred to as “self-service” because the people who want to use this data, the analysts, the accountants, the managers and so on, are also the people who are building the reports—they do not have to rely on help from the IT department to do so. Of course, this has always been one of the most popular uses for Excel but, with Power BI, Microsoft has focused on making it even easier to do this. By doing so, Microsoft has reaffirmed Excel's position as the tool of choice for anyone who works with data as part of their job.

Given that Power Query is just one part of the Power BI suite, it's important to look at all of the components of Power BI so you can put Power Query in context, understand what it does, understand when you should use it and when another tool is more appropriate.

Power BI Components

The components of Power BI divide up into two groups: Excel add-ins and the cloud services that are available via Power BI for Office 365. Following are the Excel add-ins:

- Power Query
- Power Pivot
- Power View
- Power Map

And following are the cloud services available via Power BI for Office 365:

- Power BI Sites
- Q&A
- Power BI Data Catalog
- Mobile BI app

Now, let's look at each one of them in a bit more detail, in the order that you are likely to use them, so that you can see what their roles are.

Power Query

Conveniently enough for this book, Power Query is going to be the first component in the Power BI suite that you use when building a new BI solution. That's because the first step in creating any kind of report or dashboard is to get hold of the source data for it. Power Query allows you to connect to a wide variety of different data sources, extract data from them quickly and easily, and define a series of repeatable steps to clean, filter, and otherwise transform your data before it gets loaded into Excel. Power Query gives you the option of loading data directly into the worksheet or, if you are using Excel 2013, into the new Excel Data Model. Obviously, this is a greatly simplified account of what Power Query is capable of, and the rest of the book will go into a lot more detail!

Power Pivot and the Excel 2013 Data Model

The Excel Data Model is an in-memory database engine that runs inside Excel and that allows you to load very large amounts of data (much larger than you could ever fit on a single worksheet—it can handle millions, even tens of millions of rows of data) for analysis purposes. In a traditional Excel BI solution, you might be used to loading your raw data onto a hidden worksheet and then using VLOOKUPs to move values from this hidden worksheet into a report, or using PivotTables to slice and dice your data. The Excel Data Model gives you a better way of doing both of these things. If you load your data into the Excel Data Model, you no longer have to use a hidden worksheet to store your data; once the data is in the Excel Data Model, you can create PivotTables directly from it and you can use Excel's Cube Formulas to import individual values into cells in the worksheet.

The Excel Data Model is, in fact, the database engine behind the Power Pivot add-in, first released for Excel 2010. In Excel 2010, Power Pivot was a self-contained unit: the add-in consisted of a user interface as well as a database engine. In Excel 2013, it was split in two and the database engine was built directly into Excel. The Power Pivot add-in still remains but only as a user interface. You have to use the Power Pivot add-in to be able to use certain, more advanced functionality, but you do not need it if you only want to perform basic tasks. The split between Power Pivot and the Excel Data Model in Excel 2013 has confused many people and, as a result, you will find plenty of books and articles that still refer to both as Power Pivot.

Apart from the ability to work with very large data volumes, loading data into the Excel Data Model has many other advantages over the "hidden sheet" method including the following:

- Multiple tables of data can be loaded into it. After you've loaded the tables of data, you can create relationships between those tables similar to those that you can create in Access or other relational databases. Once relationships have been created, you can create PivotTables that include data from all of these tables—traditional PivotTables are limited to just one source table.
- Complex calculations can be written in DAX, the language of Power Pivot. These include time series calculations, percentage shares, and many other types of calculations that are difficult to implement in regular Excel formulas.

- The Excel Data Model compresses data very efficiently so loading your data into it, instead of into the worksheet, can drastically reduce the size of your workbooks. This is particularly important if you intend to upload your workbook to a Power BI site because there are limits on the size of workbooks that can be uploaded, as you'll soon find out.
- You can add features such as hierarchies, which allow your users to drill down through your data following a predefined path, and KPIs, which are special calculations that help users track business performance.

Power Pivot and the Excel Data Model are together a gigantic topic and deserve a book to themselves; indeed many excellent books, white papers, and blog posts are already out there. If you are serious about building BI solutions in Excel, then you owe it to yourself to learn more about this topic.

Power View

Power View is a new feature of Excel 2013 that allows you to create attractive, interactive dashboards as new worksheets inside your workbook. It is very closely related to, but not to be confused with, a stand-alone application that is also called Power View but that is only available in SharePoint. Power View in Excel takes data stored in a worksheet or from the Excel Data Model as its starting point, and then, using drag and drop, it allows you to create tables, charts, and maps that can be easily filtered and sliced. Even if you can already build dashboards and reports in a regular worksheet with PivotTables or PivotCharts, Power View gives you more options for visualizing your data, it is easy to use, and the end product is more visually appealing.

Power Map

Power Map is an Excel add-in that allows you to overlay geographic data onto 3-D maps. For example, if you work with retail data, you can use Power Map to plot sales of all of your stores on a map. At the location of each store, you can use it to show a bar graph of sales within that store. Once you've done this, you can explore these maps, panning and zooming in, and even record "tours" that show data for many different locations. Like Power View, its main strength is its "wow" factor—swooping low over a city and seeing hundreds of tiny graphs appear over a neighborhood you know makes for a very impressive spectacle.

SharePoint and Power BI for Office 365

Having loaded your data into Excel using Power Query, created PivotTables and reports using Power Pivot and the Excel Data Model, and created dashboards and visualizations using Power View and Power Map, the next thing you are likely to want to do is share the fruits of your labor with your colleagues. There's nothing stopping you from e-mailing your workbook to them, but there is a better way: SharePoint.

The on-premises version of SharePoint is used by many organizations to manage documents. SharePoint Enterprise Edition includes Excel Services, which allows Excel workbooks to be viewed and edited in a browser. Excel Services does not support all Excel desktop functionality, but it does have very good support for Excel's BI features: PivotTables, Excel Cube Formulas, and Power View sheets can all be viewed in a browser. This makes SharePoint and Excel Services a great way to make your reports available to your coworkers.

An increasing number of organizations are now moving to SharePoint Online, Microsoft's cloud-hosted version of SharePoint, which is available as part of Office 365. If your organization has moved to Office 365 and uses SharePoint Online, you can publish your workbooks to a SharePoint Online document library, share them with other people, and interact with them in a browser via Excel Services. For simple scenarios, this may be sufficient for your reports and dashboards.

If you purchase an additional Power BI for Office 365 subscription, you get several very useful features in addition to what is available with a regular Office 365 subscription:

- While SharePoint Online allows you to upload documents of up to 2GB in size, if you want to be able to view your BI workbooks using Excel Services in a browser, you are limited to workbooks of only 10MB. Power BI for Office 365 allows you to increase this limit by distinguishing between the size of the data in the Excel Data Model (which can be anything up to 250MB at the time of writing) and the rest of the workbook (which is still limited to 10MB).
- Data held in the Excel Data Model of workbooks can be refreshed on a schedule, even if the data sources are not themselves in the cloud. Installing the Data Management Gateway component on-premises allows Power BI for Office 365 to connect your workbook in the cloud with your organization's SQL Server and Oracle databases.
- The Power BI Data Catalog allows administrators to provide access to a variety of public and corporate data sources through a Power BI site by exposing them as OData data feeds.
- Power Query users can also share their work with their colleagues via the Power BI Data Catalog. (This will be explained in more detail in Chapter 7.)
- Power BI sites provide a smarter, more businesslike way to organize your reports and dashboards than a regular SharePoint Online document library.
- Power BI sites also provide management dashboards that allow you to track which workbooks your colleagues have been using and when.
- Q&A is a feature of Power BI for Office 365 that allows users to query the data held in the Excel Data Model of workbooks using English language queries and then see the results of these queries displayed in a Power View report. While Q&A only speaks English at the moment, other languages may be supported in the future.

Mobile BI

The final component of the Power BI suite is the Mobile BI app. If you have uploaded a workbook to a Power BI site, you have the ability to see parts of it—tables, charts, named ranges, PivotTables, PivotCharts, and Power View dashboards—via the Mobile BI app. This is a Windows 8.1 app designed to allow users of Windows tablets to access BI data while they are away from their desktops. Similar apps for other mobile platforms such as iOS will be released in the future.

Workflow Summary

In summary, then, the Power BI workflow is:

1. load data from external data sources into Excel using Power Query
2. model data in the Excel Data Model and add calculations using Power Pivot
3. build reports and dashboards on this data using PivotTables, Excel Cube Formulas, Power View, and Power Map
4. publish your workbook to a Power BI site in SharePoint Online so other users can access the reports in it

Figure 1-1 shows the workflow in more detail.

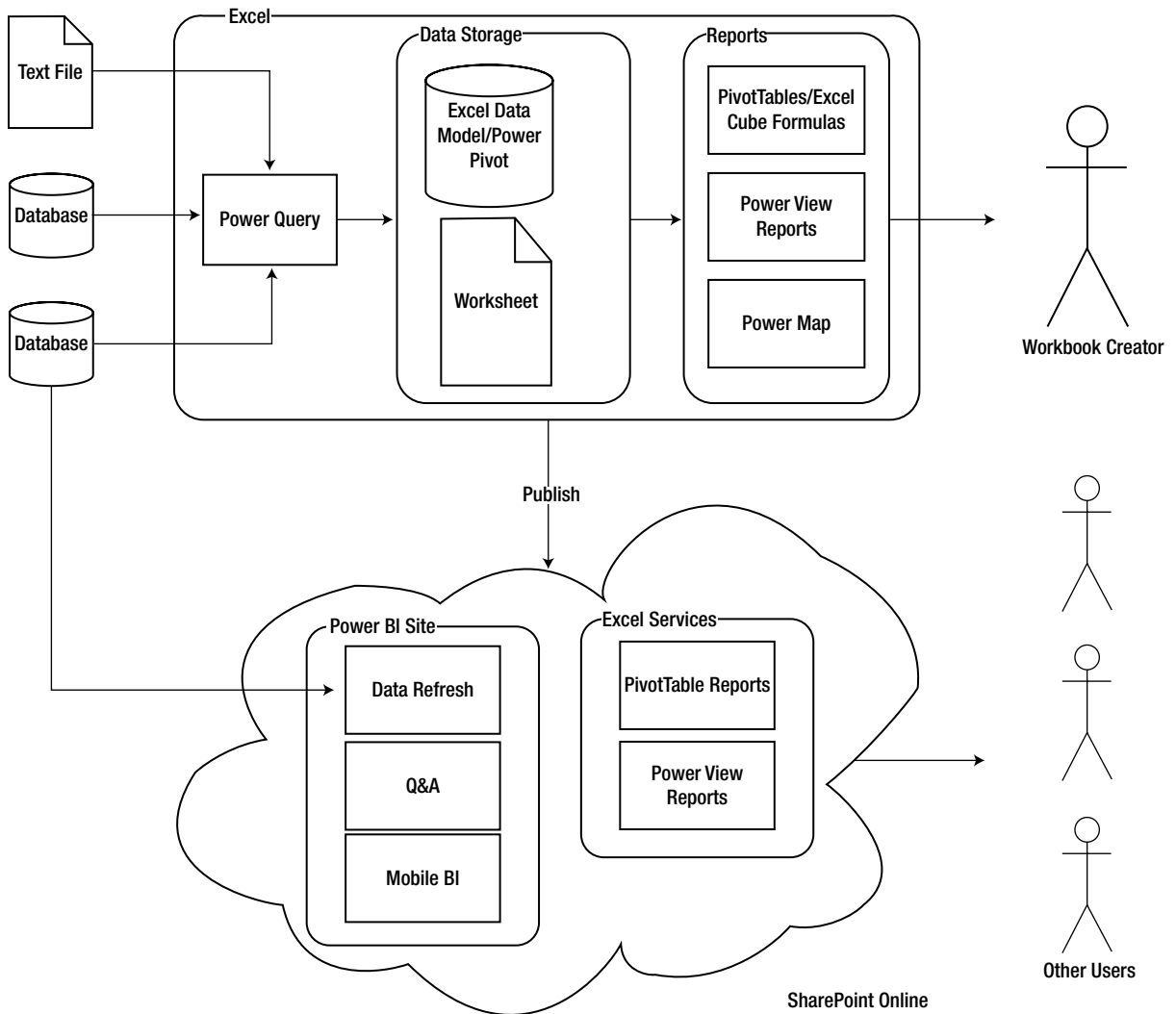


Figure 1-1. The Power BI workflow

Power Query and Power BI Licensing

The licensing model for Power Query and Power BI for Office 365 is not straightforward and requires some explanation. For Excel 2013 users, Power Query is licensed in exactly the same way as Power Pivot and Power View (though not Power Map, which is only available if you have an Office 365 subscription), and it is available as for free if you have bought one of the following:

- the stand-alone version of Excel 2013
- an Office 2013 Professional Plus license for your desktop
- an Office 365 subscription that gives you access to the Office 2013 Professional Plus desktop tools. There are a large number of subscription SKUs, but the Office 365 ProPlus, E3, and E4 SKUs all include the right version of Excel.

If you have Excel 2010, the only edition that supports Power Query is Office Professional Plus Service Pack 1. Power View and Power Map are not available for Excel 2010, but Power Pivot is available for all SKUs of Excel 2010.

The ability to view Excel workbooks stored in the on-premises version of SharePoint in a web browser is only available in SharePoint Enterprise Edition, and the functionality works best if you have SharePoint 2013. In addition, you will need to have Power Pivot for SharePoint installed.

The cloud-based services included in Power BI for Office 365 (Power BI Sites, Q&A, the Power BI Data Catalog, and the Mobile BI app), are covered by a separate subscription model. A prerequisite of using these services is an Office 365 subscription that includes SharePoint Online; you then need to purchase an extra subscription to use the services themselves.

Full licensing details, as well as links to install the various Power BI components, can be found at <http://www.microsoft.com/powerbi>. None of the Power BI Excel add-ins are available in the Apple Mac or Windows RT versions of Excel.

Installing Power Query

Power Query must be downloaded and installed separately after Excel has been installed; if you do not see the Power Query tab in the ribbon after you have installed it, you may need to enable the add-in manually. It requires Windows Vista or greater, if you are installing it on a desktop PC, or Windows 2008 or greater if you are installing it on a server. It also requires .NET 3.5 Service Pack 1 and Internet Explorer 9 or greater.

There are 32-bit and 64-bit versions of Power Query available. If you have the 32-bit version of Excel installed, you must install the 32-bit version of Power Query; if you have the 64-bit version of Excel, you must install the 64-bit version of Power Query. The 64-bit version is recommended if you are going to be working with large amounts of data.

New versions of Power Query are released on a regular basis and these can include significant new functionality. The Update version on the Power Query tab in Excel will light up when a new version is available to download.

Power Query Concepts

Now that you have seen where Power Query fits in the Power BI suite, it's time to start looking at it in more detail. A good place to start is by defining some of the concepts and terminology you'll encounter when working with Power Query. Luckily, there isn't much terminology to define. (If you find it easier to learn by seeing rather than reading, you may prefer to skim over this section and then move onto the next one, where you'll find the same concepts described by means of a worked example.)

The most important concept in Power Query is that of a *query*. (Unfortunately, this means you have to talk about "Power Query queries.") A query is a job that imports data from one or more data sources, optionally does something to the data such as filter it or aggregate it, and then loads it into Excel. Queries are stored inside an Excel workbook, and one workbook can contain multiple queries. Queries themselves can be used as data sources for other queries; two queries may also be merged together rather like one table can be joined to another in SQL, and a query can be appended onto the end of another.

A query is composed of one or more *steps* that are arranged in a specific order. A step may connect to a data source to retrieve data (usually the first step in a query does this, but subsequent steps may do this, too), or it may take the data returned by a preceding step and apply some kind of transformation to it. The last step in a query returns the output of the query as a whole. Steps are defined in Power Query's own expression language, "M," which sadly bears no resemblance to Excel formula language or VBA and is, in fact, a functional language like F#. When you create a new step in the Power Query user interface, Power Query will generate the M code necessary for that step automatically. If you are a confident programmer, you can edit the code behind each step yourself or create new steps by writing your own M code. However, in the majority of cases, this is not necessary.

Although Power Query steps may connect to external data sources, there is no concept of a connection to a data source as a separate object, and Power Query cannot use connections defined elsewhere in Excel (that's to say, the connections that can be found on the Data tab in the ribbon). However, the output of a Power Query query does become a traditional Excel connection and can be used just like any other connection. This allows Power Query to be able to output results to a table in a worksheet, for example.

Power Query Walkthrough

This section illustrates the concepts described above by walking through the creation and editing of a simple Power Query query. It also acts as a brief guided tour of the Power Query user interface. As you start to use Power Query, you'll find that all kinds of extra menus, panes, and tabs appear depending on what you're doing. In the following chapters, as you learn more about Power Query functionality, you'll become more familiar with the details of the user interface.

Creating a Simple Query

The starting point for all of your Power Query work will be the Power Query tab on Excel's ribbon menu, shown in Figure 1-2.

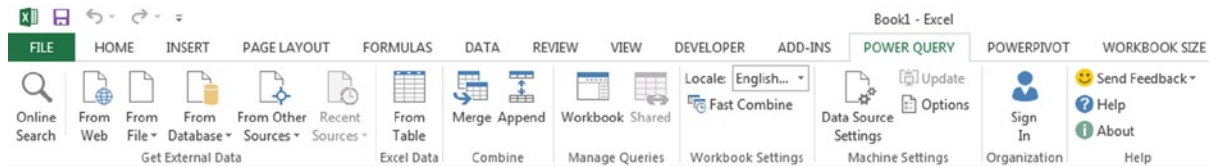


Figure 1-2. The Power Query tab

On the left-hand side of the Power Query tab, in the Get External Data section, you can see the options to import data from different types of data sources, while in the Excel Data section you can see a button to import data from an Excel table in the current workbook. Elsewhere in the Power Query tab, you can see options for combining data from different queries as well as various other administrative options that will be covered later in this book.

Click the From File button. On the drop-down menu click on the From CSV option as shown in Figure 1-3.

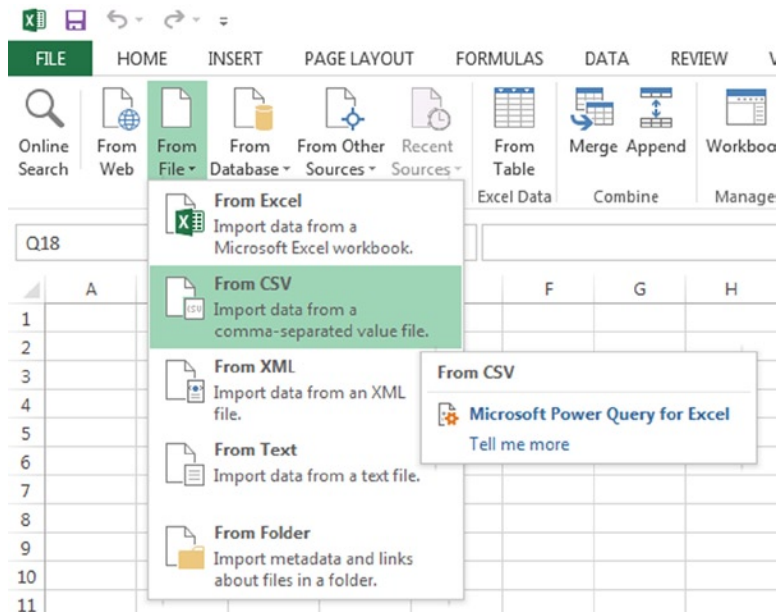


Figure 1-3. The From CSV button

A dialog box will appear. You should select the file 01_01_SimpleSales.csv from the sample files for this chapter. When you do this, the Power Query Query Editor window will appear as shown in Figure 1-4. The five main areas of the Query Editor window, marked in Figure 1-4, are the following:

1. **The Query Editor toolbar.** This is a ribbon menu containing all of the options for creating new steps in your query.
2. **The Formula Bar.** This is where you can edit the M expression for each step in the query.
3. **The Navigator pane.** This collapsible pane allows you to navigate through certain types of data sources such as XML files so you can find the particular data you are looking for.
4. **The Results pane.** This is where you can see the data returned by the currently selected step.
5. **The Query Settings pane.** This pane displays the name of the query you are editing, a description, a list of all of the steps in the query in the Applied Steps section, and two check boxes that control where the output of your query will be sent to in the Load Settings section.

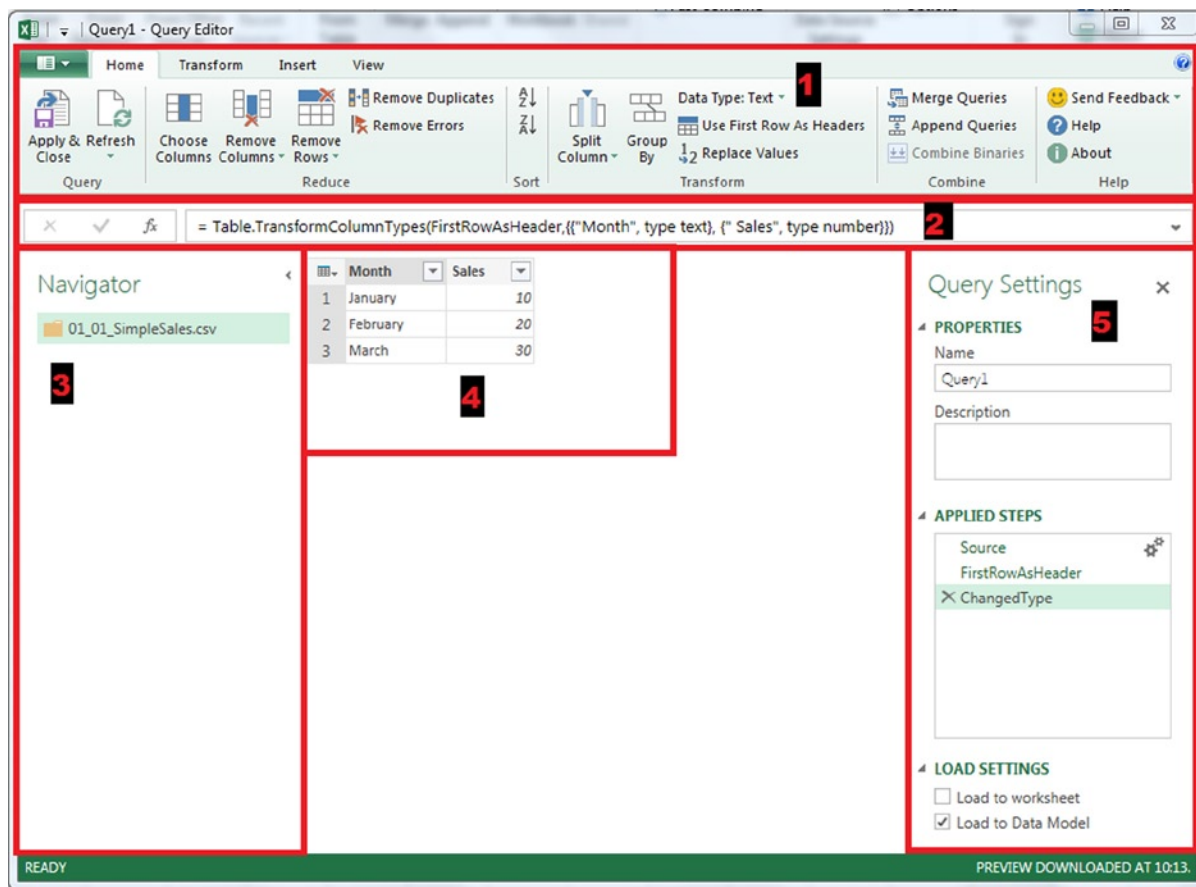


Figure 1-4. The Query Editor window

At this point, a new query will have been created, called Query1, which has three steps already created for you that are listed in the Applied Steps section on the right-hand side of the screen in the Query Settings pane. The output of the third step, called ChangedType, will be visible in the Results Pane. If you click any of the other earlier steps, you will see the output of the query after that step has been executed there instead. (This makes it easy to debug a Power Query query because it allows you to see how each step changes the data.)

Now click the Apply & Close button in the top left-hand corner of the editor and the window will close. Since the Load to Worksheet check box (found under Load Settings in the bottom right-hand corner of the screen) was checked, a new worksheet will be created and the output of this query will be shown in a new Excel table located at cell A1 of the new worksheet, as shown in Figure 1-5.

	A	B	C	D	E	F
1	Month	Sales				
2	January	10				
3	February	20				
4	March	30				
5						
6						

Figure 1-5. Query output

Congratulations! You have created your first Power Query query.

Editing an Existing Query

Queries can be edited after they have been created. Before you can do this, though, you need to go back to the Power Query tab on the ribbon and click the Workbook button in the Manage Queries section, so you can see a list of all the queries present in this workbook. When you do this, the Workbook Queries pane will appear on the right-hand side of the screen, as shown in Figure 1-6. The query you have just created, Query1, will be visible in this pane.

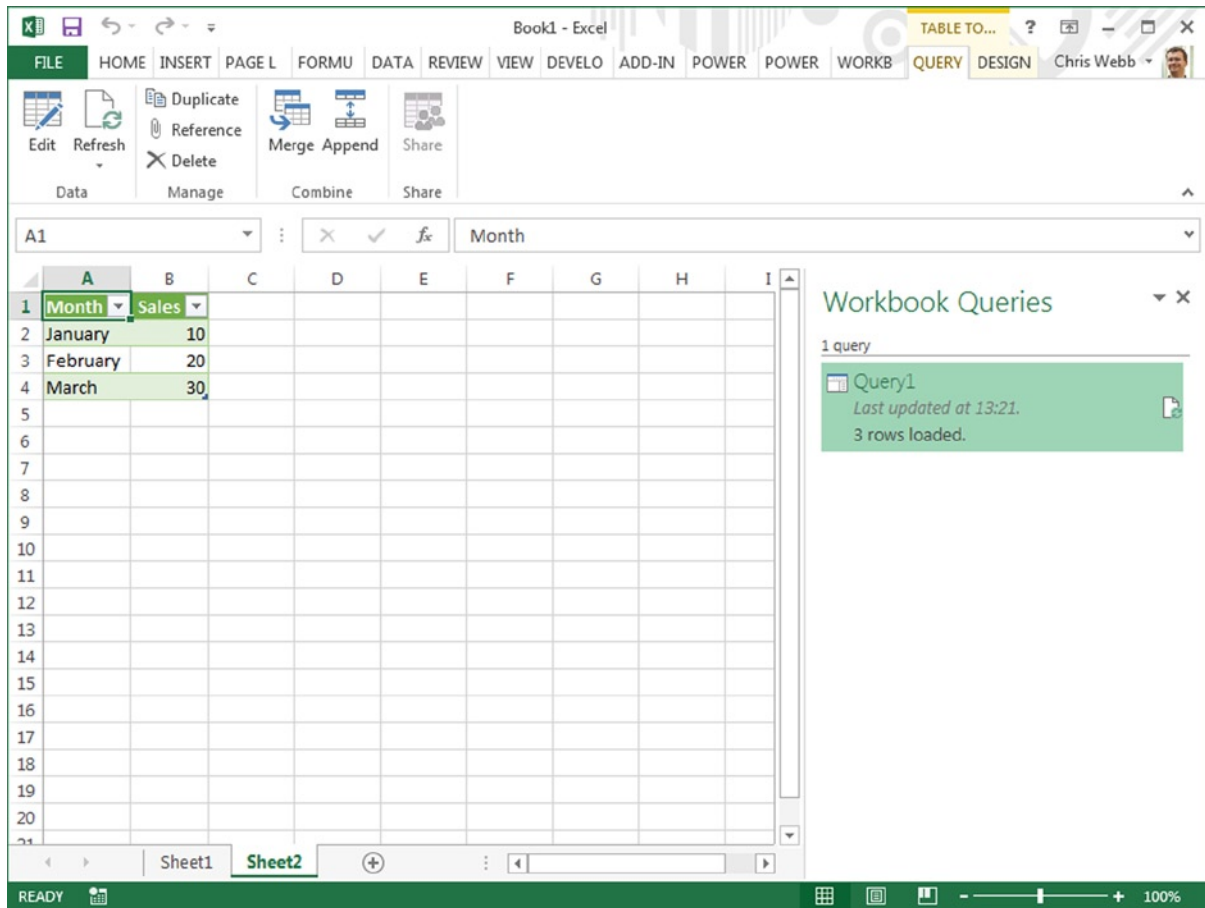


Figure 1-6. The Workbook Queries pane

Now, move your mouse cursor over the Workbook Queries pane and Query1. When you do this, a fly-out box will appear showing a preview of the data, when the query was last refreshed, and where the query is to be loaded. Also on the fly-out box is a series of clickable menu options plus ellipses that, when clicked, reveal even more menu options. (See Figure 1-7.) The menu options in this fly-out box are also available in a drop-down menu if you right-click the query in the Workbook Queries pane.

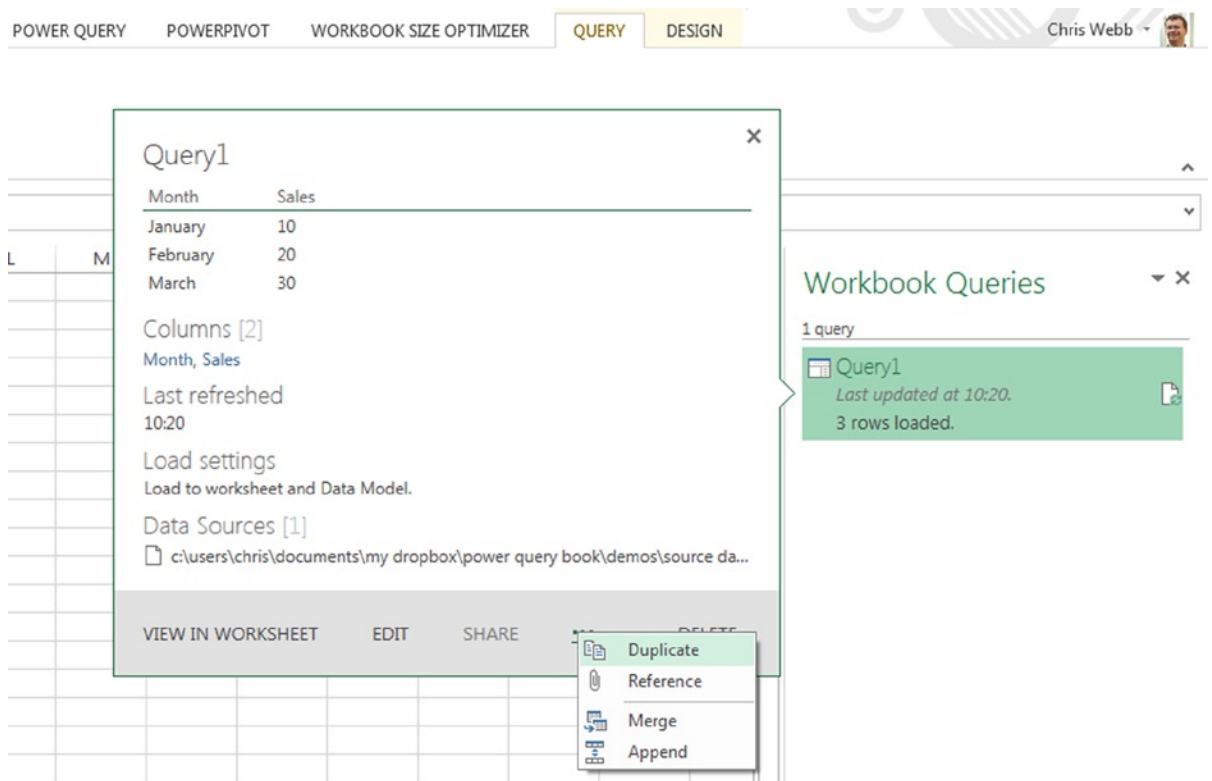


Figure 1-7. The Workbook Queries fly-out menu

Click the Edit option on the fly-out box and the Query Editor window will reopen, looking just as it does in Figure 1-4. Next, click the Remove Rows drop-down menu, which can be found in the center left of the Home tab of the Query Editor toolbar in the Reduce section, and click Keep Top Rows. This will add a new step onto the end of your query that filters out all but a given number of rows at the top of your query. Before it can do this, though, you will need to tell Power Query how many rows you want to keep. When you click the button, Power Query will open a dialog box, as shown in Figure 1-8, asking you for the number of rows to keep. Enter the value 2 to keep only the top two rows and click OK.

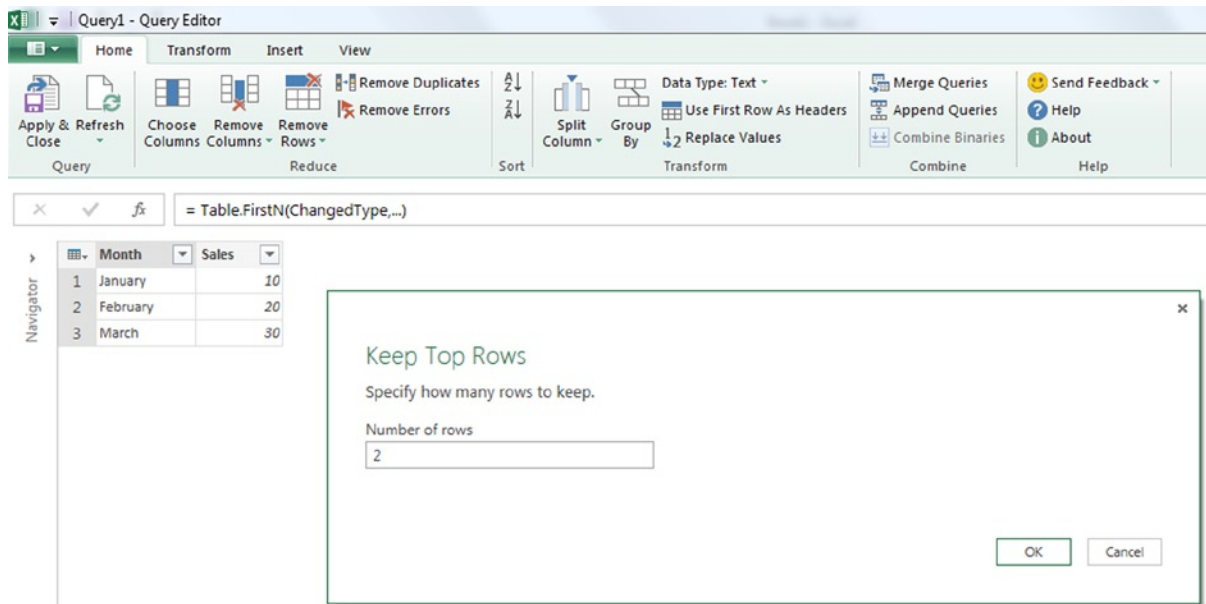


Figure 1-8. The Keep Top Rows dialog box

Once you have clicked OK, you will see the data has changed. Before, there were three months listed—January, February and March. Now, there are only two—January and February, as shown in Figure 1-9. A new step has also been added to the query called KeptFirstRows and the M code for this step is visible in the formula bar.

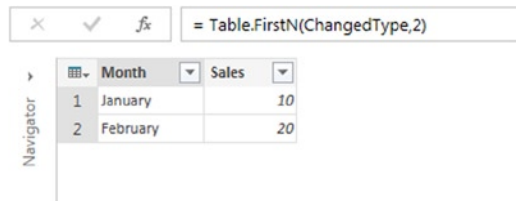


Figure 1-9. The output of the KeptFirstRows step

Finally, click the Apply & Close button, and you will see the new output of the query appear in the Excel table in the worksheet, as shown in Figure 1-10.

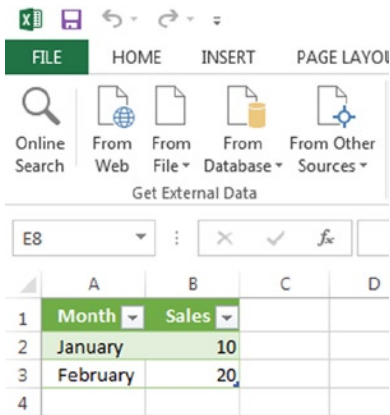


Figure 1-10. The output of the edited query

Why Use Power Query?

If you are an experienced Excel user, you may be thinking that you can already do everything that Power Query does by using the functionality present on the Data tab in combination with Excel formulas and maybe some VBA, without having to learn a completely new tool. That may be true up to a point, but there are some very strong arguments in favor of using Power Query over these well-known and trusted techniques. This section will make these arguments in some detail.

Faster Release Cycles

Microsoft Excel is one of the most widely used applications. As a result, every change that Microsoft makes to Excel has to be tested thoroughly; a bug could affect millions of users worldwide and cause untold disruption in their organizations. This means that Microsoft only releases a new version of Excel, on average, every three years—a very slow release cycle by modern standards and one that makes it difficult to get new functionality to users quickly. The new, streamed installation option for all Office products, which is available to Office 365 subscribers, will solve this problem eventually and allow Microsoft to push changes in Excel out to users as soon as they are available; however, it will be several years before most organizations change over to use this type of installation.

In the meantime, making the Power BI Excel functionality available through Excel add-ins rather than integrating it directly in Excel gives Microsoft a lot of flexibility it would not otherwise have. New releases of Power Query appeared on an almost monthly basis in the second half of 2013. Although the tempo has slowed since then, releases are still appearing regularly. This allows the Power Query team to release new features, fix bugs, and respond to user feedback much faster than if they were tied to the Office release cycle.

Based on the way Power Query started life as a completely separate add-in for Excel 2010 and then had its engine integrated natively into Excel 2013, it's a reasonable assumption to make that in the future Power Query functionality will be integrated into Excel in the same way. At the moment, there is a lot of overlap between what you can achieve on the Data tab in Excel and with Power Query; in the future, Power Query might be the replacement for the Data tab.

Therefore, if you are the kind of person who likes to get your hands on new functionality as soon as possible or if you have already run into the limitations of existing Excel functionality, Power Query is for you. Microsoft is making a huge investment in Power BI as a whole and Power Query in particular. Even if you find that Power Query can't do something that you need today, it could well be the case that it will be able to do it in a very short while.

Connectivity to New Data Sources

The most obvious example of where Power Query scores over native Excel functionality, and where the gap widens with each release of Power Query, is with the number of data sources that it supports. Power Query supports almost all of the data sources that can be accessed via the Data tab and it adds many new ones including Facebook, Active Directory, Exchange, SharePoint, OData feeds, and HDInsight. It also adds support for connections to web services. All of these data sources will be examined at length in Chapter 2.

Admittedly, at the time of writing, there are some data sources that Power Query does not support yet: SQL Server Analysis Services, ODBC, and some OLEDB data sources beyond the major relational databases. Hopefully, these gaps will be addressed in a future release.

Improved Connectivity to Existing Data Sources

Where Power Query supports a data source that is already supported natively by the Excel Data tab, it is often the case that it improves on the native functionality. One example is the ability to scrape data from a web page; Power Query is able to extract data from far more web pages than the native functionality is able to. Another example is the way that Power Query can merge data from a folder containing multiple csv files into a single result set with minimal effort.

Automate Data Loads

Power Query allows you to replace manual processes for loading data into Excel with a series of automated, repeatable steps that can be run with the click of a button. This not only saves time, but it is also much less prone to errors. Compared with VBA, Power Query is likely to offer better performance and also much easier maintenance for automating data loads, given that so many tasks require no code whatsoever.

Create Transformations and Calculations Easily

Building on from the previous point, as you saw in the walkthrough, Power Query makes it very easy to filter, aggregate, sort, pivot and unpivot, as well as to add calculations, just by clicking a button. Of course, not everything can be done through the user interface, but the Power Query development team has done an excellent job of identifying and implementing all of the common scenarios that need to be dealt with when loading data into Excel. Some more complex scenarios will require you to write M code; some scenarios will still be better handled using Excel formulas or by using DAX once the data has been loaded into the Excel Data Model. However, you will be pleasantly surprised at how easy Power Query is to use and how much it increases your productivity even if you are already a whiz at Excel formulas or VBA.

Do the Work in the Data Source, Not on the Desktop

One aspect of the way Power Query works that is not obvious but extremely important is the way that it will always try to push processing back to the data source. For example, if you are using a SQL Server data source and your query contains several steps, Power Query will translate all of the steps in your query into a single SQL SELECT statement, where possible. If the source table is very large, it will be much faster for any aggregation or filtering to take place in the SQL Server than for the whole of the table to be downloaded to Excel and the work to be done there. Note that this is not going to be possible with some types of data source (such as text files) or all of the transformations that Power Query is capable of. A full discussion of this behavior, called “query folding,” is given in Chapter 3.

Share Queries between Workbooks and Users

Finally, Power BI's integration with Power BI Sites means that you can reuse queries in multiple workbooks and even share them with other users. Excel workbooks have a bad reputation for being silos of forgotten code and stale data. Power Query goes some way to try to address this problem, at least as far as data loading goes. If you think of all of the Excel reports and dashboards that exist in an organization, it's likely that a large number of them will use data from the same data sources and need to do the same things to that data before it can be of any use. If a single, shared query can be used to do this rather than multiple duplicated queries (with all the risk of errors or different implementations of business logic that go with that), then Excel reports will be faster to develop and more consistent between themselves.

Summary

By now, you should understand what Power Query does and what role it plays in the Power BI suite. You should also be familiar with the Power Query user interface and some basic concepts such as queries and steps. These topics have only been dealt with at a high level so, from now on, this book will concentrate on the details. In the next chapter, you will learn about all of the data sources supported by Power Query and how they can be accessed as well as related topics such as authentication and user locale.

CHAPTER 2



Power Query Data Sources

One of the best things about Power Query is the wide range of data sources that it supports. In addition to the obvious types of data source, such as relational databases and text files, it supports more exotic sources including Facebook, Active Directory, and OData, which are becoming more and more important in self-service BI scenarios. This chapter will deal with that all-important first step when creating a Power Query query—making a connection to a data source and extracting the raw data you want to work with. It will provide a guided tour of all of the data sources supported by Power Query, and it will also discuss related topics such as authentication.

Querying Relational Databases

Relational databases are the most commonly used type of data source for Power Query. The following relational database types are supported by Power Query: SQL Server, Windows Azure SQL Database, Access, Oracle, DB2, MySQL, PostgreSQL, Sybase, and Teradata. Unfortunately, there is no way of connecting directly to a relational database that is not on this list and, at the time of writing, there is no support for connections to other OLEDB or ODBC data sources. Luckily (at least for the purposes of this book), the user interface for connecting to each of these data sources is almost identical, so SQL Server will be used as the main example and then any differences for other databases will be listed afterwards.

Connecting to SQL Server

In order to create a new Power Query query that connects to and extracts data from SQL Server, you must first go to the Power Query tab in the Excel ribbon, click the From Database button, and then select From SQL Server Database (as shown in [Figure 2-1](#)).

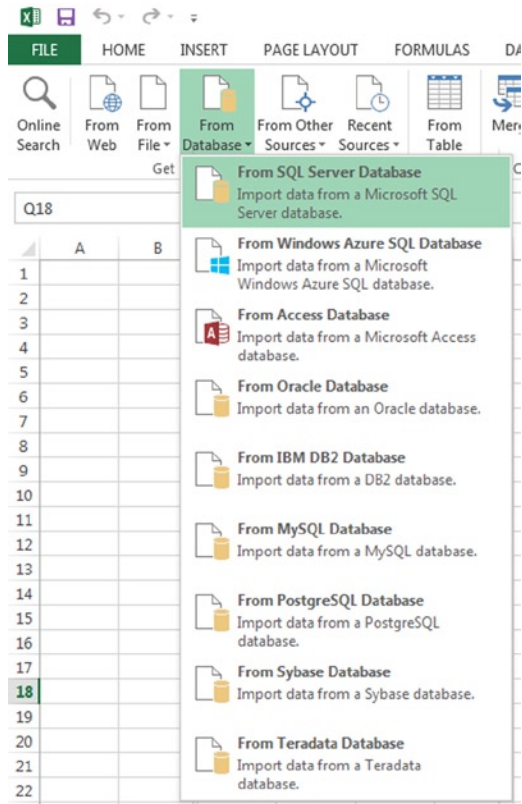


Figure 2-1. The From Database button

Once you have done this task, a dialog will appear that allows you to enter the name of the server and instance that you want to connect to and (optionally) the name of the database. You can also enter your own SQL query at this point if you have already written one. The dialog is shown in Figure 2-2.

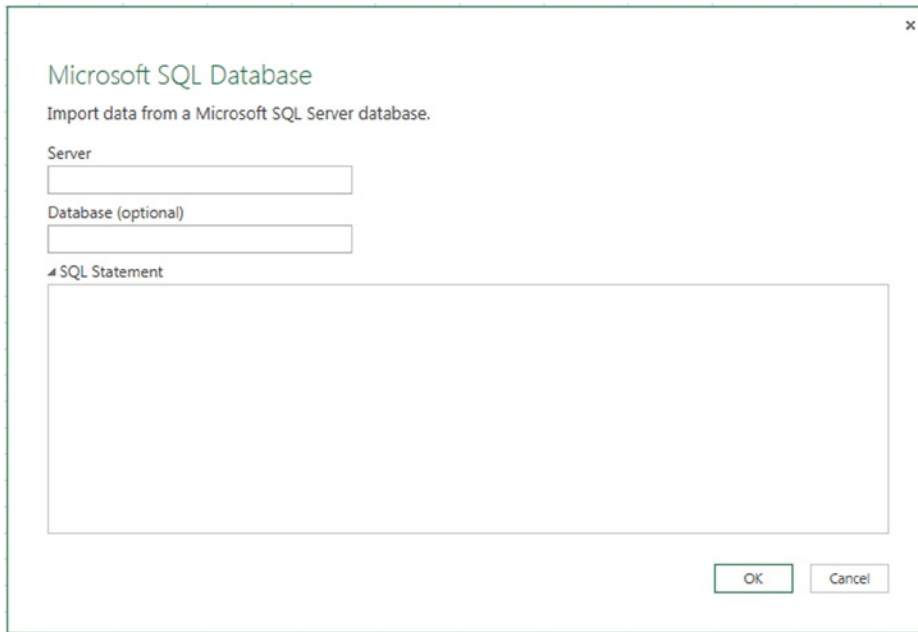


Figure 2-2. The Microsoft SQL Database connection dialog

The SQL Statement text box can take either a SQL Select Statement or a call to a stored procedure. In the latter case, the syntax you use is exactly the same as you'd use in SQL Server Management Studio, for example:

```
EXEC [sys].[sp_who];
```

If this is the first time you are connecting to the specified instance of SQL Server, Power Query will open a new dialog asking how the connection should be authenticated. Connections to SQL Server can use either Windows authentication, which means that your own Windows credentials will be used to authenticate your connection and nothing else needs to be entered or, if your database is in Mixed Mode, you can use SQL Server authentication and enter a username and password. When the dialog opens, you will see the Windows tab, shown in Figure 2-3; you will need to click the Database tab if you are using SQL Server authentication, as shown in Figure 2-4. In both cases, there is a check box to use an encrypted connection that is checked by default. If you are connecting to an instance of SQL Server that has not been configured to accept encrypted connections, you will need to uncheck this box. Your credentials will be saved after you have entered them and used automatically for all future connections to the same database.

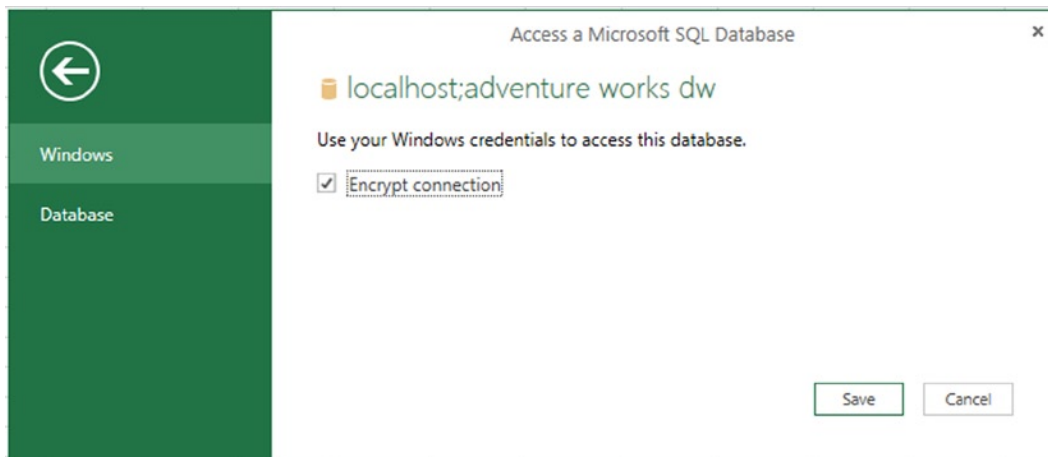


Figure 2-3. The SQL Server authentication dialog for Windows authentication

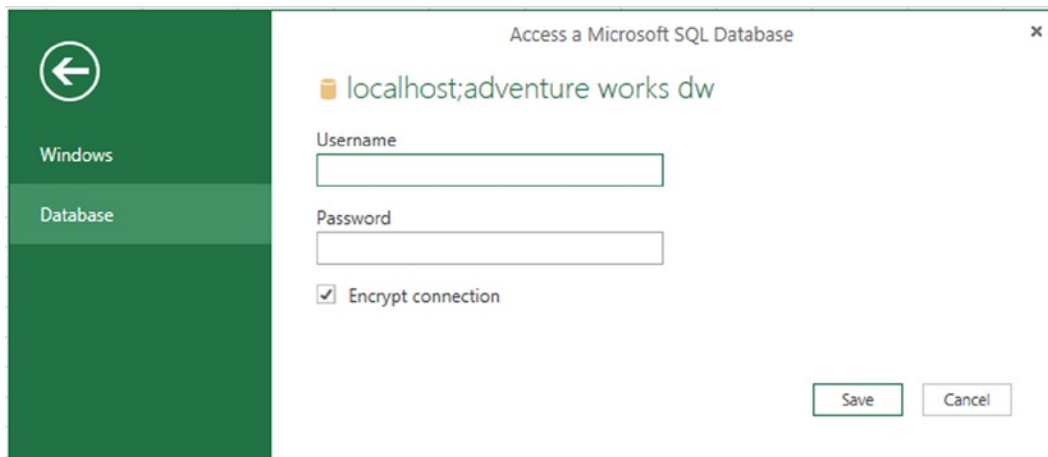


Figure 2-4. The SQL Server authentication dialog for SQL Server authentication

If you do enter your own query or call a stored procedure, once you click OK in the original Microsoft SQL Database connection dialog, the query or stored procedure will be run and the Query Editor window will open to show the results.

■ **Note** If your query returns a large number of rows and you intend to apply further transformations (such as filters or group bys) in subsequent steps inside your query, you may be better off not using your own query at all. You may get better performance connecting directly to the underlying tables in your database and using Power Query alone to implement the logic in your query. The reason for this advice will be explained in detail in Chapter 3 in the section on Query Folding.

If, on the other hand, you do not enter a database name or leave the SQL Statement text box blank, some extra functionality for navigating through a relational database is enabled. The first thing that will happen is a new pane called the Navigator pane appears on the right-hand side of the screen allowing you to select one or more tables, views, scalar functions, or table-valued functions in your database, as shown in Figure 2-5.

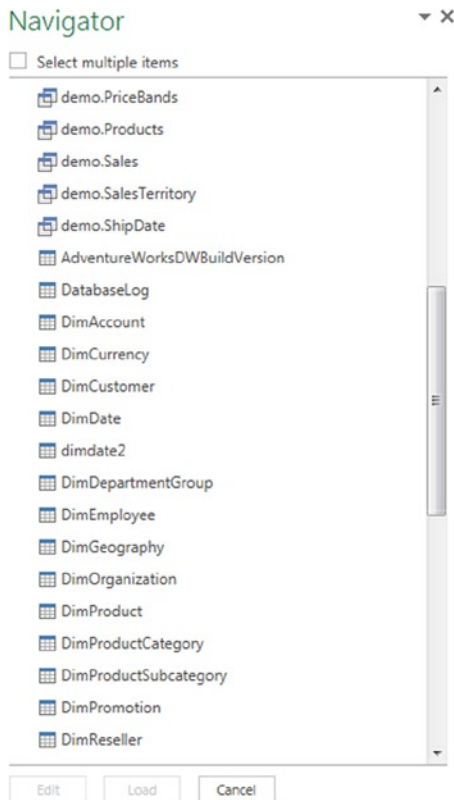


Figure 2-5. The Navigator pane

Initially, the Navigator pane will only allow you to select one object, but, if you check the Select Multiple Items box, you will be allowed to select multiple objects. Selecting multiple objects will result in multiple Power Query queries being created, one for each object that has been selected; the Edit button will disappear and new check boxes will appear asking whether you want to load the data to the Excel Data Model and/or direct to a worksheet when you click Load. If you do not check the Select Multiple Items box, you can open the Query Editor by clicking the Edit button or load data directly to the worksheet by clicking the Load button after you have made your selection.

■ **Note** If you select a function in the Navigator pane, it will be imported as a Power Query function (a topic that will be dealt with in Chapter 5).

Navigating through a Database in the Query Editor

If you have imported a table into Power Query using the Navigator pane, then some extra functionality is enabled in the Query Editor for navigating along the relationships of your database. To illustrate this, I'm going to use the following tables from Microsoft's Adventure Works DW sample database: DimDate and FactInternetSales. These tables are shown in Figure 2-6. There are three foreign key relationships defined between the two tables: the OrderDateKey, DueDateKey, and ShipDateKey columns on FactInternetSales join to the DateKey column on DimDate.

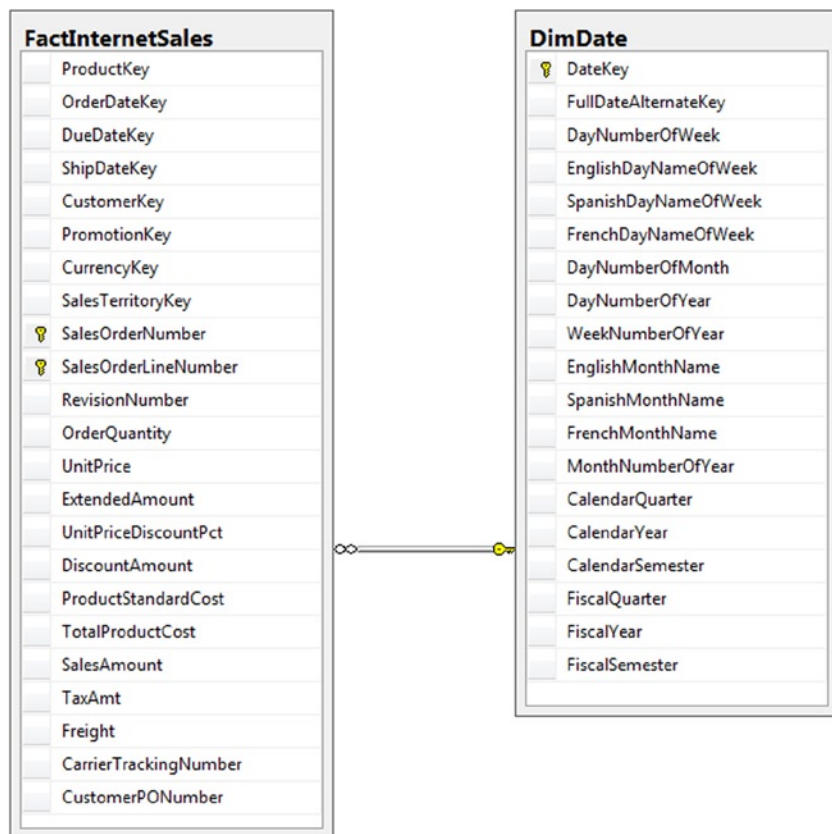


Figure 2-6. Adventure Works DW table relationships

Having imported just the DimDate table into Power Query using the Navigator pane, you will see a number of columns that contain the word “Table” if you scroll to the right-hand side of the table once you are in the Query Editor pane, as shown in Figure 2-7. These columns represent data from all tables that have a foreign key relationship with the DimDate table in the database.

When you click this icon, you will see two options, shown in Figure 2-10: Expand, which will perform an inner join between the two tables, and Aggregate, which will add new columns to your table containing values aggregated from the destination table.

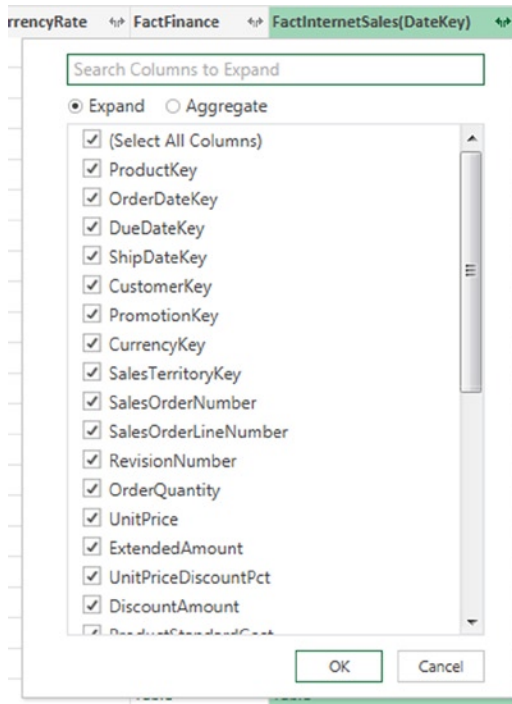


Figure 2-10. The Expand/Aggregate dialog with the Expand option selected

By default, the Expand option is selected. If all of the columns in the destination table (FactInternetSales, in this case) are checked, you will get a table that contains all of the columns from both DimDate and FactInternetSales when you click OK. Because Power Query is performing an inner join, the resulting table will have as many rows as there are in the destination table that have an equivalent row in the source table. The same thing can also be achieved by selecting the column in the Results Pane and clicking the Expand button on the Transform tab of the Query Editor toolbar.

If you select the Aggregate option (as shown in Figure 2-11), you will only see the numeric columns from the destination table, but you will see each numeric column listed multiple times, once for each possible aggregate function that can be applied to it. Each box that you check will result in a new column added to the source table containing a value aggregated from the destination table.

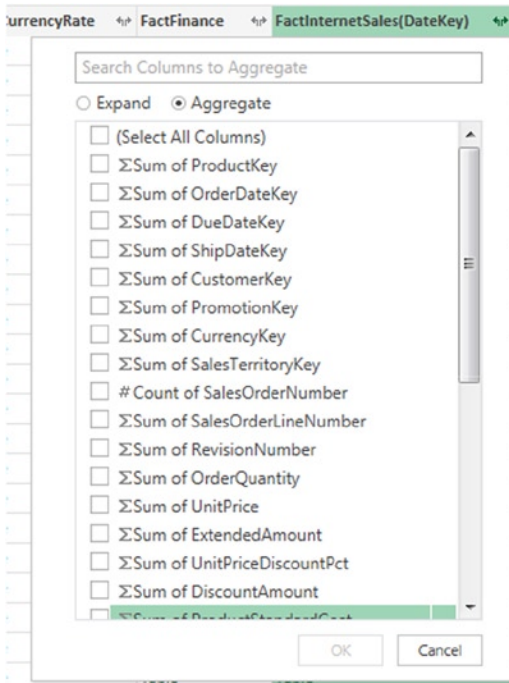


Figure 2-11. The Expand/Aggregate dialog with the Aggregate option selected

It is also possible to navigate “up” a relationship as well as “down” a relationship. When an appropriate relationship exists—for example, if you started in the FactInternetSales table and wanted to look up a value from the DimDate table—you will see a column in your table containing the value “Value” as shown in Figure 2-12.

SalesAmount	TaxAmt	Freight	CarrierTrackingNumber	CustomerPONum...	DimCurrency	DimCustomer
3578.27	286.2616	89.4568	null	null	Value	Value
3578.27	286.2616	89.4568	null	null	Value	Value
3578.27	286.2616	89.4568	null	null	Value	Value
3578.27	286.2616	89.4568	null	null	Value	Value
699.0982	55.9279	17.4775	null	null	Value	Value

Figure 2-12. Relationship columns on the FactInternetSales table

Clicking “Value” in a cell will return a record object—basically the single row from the lookup table that joins to the row in the table you just clicked, but unpivoted. An example of a record is shown in Figure 2-13; it can easily be converted to a table by clicking the Into Table button on the toolbar above.