Community Experience Distilled

PowerPivot for Advanced Reporting and Dashboards

Leverage the capabilities of the Microsoft Excel PowerPivot add-on to create advanced BI solutions

Robert Bosco J

[PACKT] PUBLISHING

PowerPivot for Advanced Reporting and Dashboards

Leverage the capabilities of the Microsoft Excel PowerPivot add-on to create advanced BI solutions

Robert Bosco J

[PACKT] PUBLISHING

BIRMINGHAM - MUMBAI

PowerPivot for Advanced Reporting and Dashboards

First published: November 2013

Production Reference: 1191113

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-84969-868-9

`www.packtpub.com`

Cover Image by Aniket Sawant (`aniket_sawant_photography@hotmail.com`)

Credits

Author
Robert Bosco J

Reviewers
Luis Guilamo
Ceriel van Halle

Acquisition Editor
Antony Lowe
Mary Nadar

Commissioning Editor
Sharvari Tawde

Technical Editors
Krishnaveni Haridas
Shruti Rawool

Copy Editors
Sayanee Mukherjee
Alfida Paiva
Tanvi Gaitonde
Shambhavi Pai

Project Coordinator
Sageer Parkar

Proofreaders
Maria Gould
Ameesha Green

Indexer
Mariammal Chettiyar

Graphics
Ronak Dhruv
Abhinash Sahu
Yuvraj Mannari

Production Coordinator
Pooja Chiplunkar

Cover Work
Pooja Chiplunkar

About the Author

Robert Bosco J is a BI Reports Developer and holds an international Master's degree in Business Intelligence from Université Tours François Rabelais, Blois, France, along with an internship as a BI Analyst at Yomii (a French e-commerce company for design products) in Paris, France. He has a Bachelor of Engineering degree in Computer Science from the Oxford Engineering college, Tiruchirappalli, India. Robert also worked as a Data Processing Executive at Meritgroup Ltd., India, before pursuing his Master's.

I would like to especially thank Packt Publishing for giving me this great opportunity to write the book, and the reviewers, Mr. Ceriel van Halle and Mr. Luis Guilamo, for being a great help in making this book accurate. I dedicate this book to my parents, John Peter and Sepasthi, for educating me and supporting me in pursuing my Master's. I would like to thank my elder and younger sisters, Anitha and Femina, who have always encouraged me in order to grow in my career. I pass on my gratitude to all my professors from Université Tours François Rabelais for introducing me to the world of Business Intelligence technologies. Thanks to them, I was able to write this book, and my heartfelt gratitude to my colleagues Chandramouly P, Nooman Hussain, and Parag Mohite for their help and suggestions in the editing process of this book.

About the Reviewers

Ceriel van Halle is a freelance Microsoft Business Intelligence Consultant and owner of footprintBI. For the past nine years, he has been working on several small and large data warehouse and reporting projects. In several roles, from a developer and trainer to the BI team lead, he has worked for companies in the healthcare, telecommunications, banking and insurance, utilities, and transportation sectors. He is a TDWI Certified Business Intelligence Professional (CBIP).

Luis Guilamo is a global business professional, a proven leader, and a Business Intelligence expert. He is an educated engineer from the University of Illinois at Urbana-Champaign, turned Business Intelligence Consultant. He focuses on end-to-end data warehouse implementations and has an extensive experience in telecommunications. He is an articulate communicator with experience in bridging business and IT across international markets in Asia, Europe, and America.

Table of Contents

Preface

Welcome to *Powerpivot for Advanced Reporting and Dashboards*. This book aims to teach readers on how to use PowerPivot to create advanced reports and prepare dashboards, all from the familiar Excel interface, and how to publish reports using the SharePoint Server. This book will cover all the fundamentals of Business Intelligence and reporting concepts with PowerPivot in the real world. We have prepared five chapters in order to be precise with the PowerPivot technologies; each and every chapter will give you a prominent idea about the PowerPivot tool. At the end of this book, users will be well-versed with PowerPivot tools in order to create advanced reports.

What this book covers

Chapter 1, *Principles and Installing PowerPivot*, illustrates quick installation of PowerPivot, some of its features, the differences in Excel's 32-bit and 64-bit version of PowerPivot, and vital information about the principles of PowerPivot, which will leverage the user's ability to work on self-service Business Intelligence.

Chapter 2, *Preparation Analysis of Data Source*, helps users explore ways to prepare data for analysis. Different types of data sources that can be imported into the PowerPivot interface have been described. A general overview of the ETL process has also been given. Users will get to now know how the ETL process works in the PowerPivot interface. An introduction to DAX and its advantages are also explained.

Chapter 3, *Data Model*, provides the user with clear information about data models' features and fundamental concepts of relationship and hierarchy. The user by now will know how the ETL process is performed in PowerPivot. One of the most important things is the DAX expression and how to use it to make structured data from unstructured data. Finally, the user will learn how to turn data from the heterogeneous type to homogeneous.

Chapter 4, Business *Intelligence Solutions Analysis*, gives users a great idea about Business Intelligence Solutions Analysis. The user must know the general concepts of BI report creation and how to make an analysis based on the data, and importance of the reports. By now, the user would know how a variety of data sources will appear on one single screen as a dashboard report. By the end of this chapter, the user will have knowledge of business scorecard's, benefits, and a general idea of Perspectives, types of Perspectives, and so on.

Chapter 5, *Publishing Reports Using the SharePoint Server*, explores the various ways a user can publish and schedule the created PowerPivot report in a SharePoint Server. The user will get more information on deployment of PowerPivot reports.

What you need for this book

Basic data analysis and an intermediate level of Excel skills are expected. Familiarity with Pivot Tables is assumed. Basic knowledge of VBA scripting and SharePoint would be useful but not necessary.

The software required are Microsoft Excel 2010 and PowerPivot for Excel add-in, which are mandatory. PowerPivot for SharePoint and SharePoint Server 2010 are not mandatory, as a user can access them remotely on a SharePoint Server that has Excel services and PowerPivot for SharePoint.

Who this book is for

This book is ideal for data analysts, reporting and MIS professionals, business analysts, managers, dashboard makers, Business Intelligence professionals, self-service Business Intelligence personnel, and students.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "The folder contains five files namely, `ContosoSales`, `Geography`, `ProductCategories`, `SQLQuery`, and `Stores`."

A block of code is set as follows:

```
CONCATENATE (Customer [First_Name], CONCATENATE ("", "", Customer
[Last_Name]))
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example appear in the text like this: "Select the language and click on the **Download** button."

[Warnings or important notes appear in a box like this.]

[Tips and tricks appear like this.]

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com`, and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at `http://www.packtpub.com`. If you purchased this book elsewhere, you can visit `http://www.packtpub.com/support` and register to have the files e-mailed directly to you.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/submit-errata`, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from `http://www.packtpub.com/support`.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with any aspect of the book, and we will do our best to address it.

1
Principles and Installing PowerPivot

The PowerPivot add-in was developed for Microsoft Excel 2010 and supports its later versions. It does not support Microsoft Excel 2007, but in Microsoft Excel 2007, a user can only open the workbook since they are created in the .xlsx format. In Microsoft Excel 2007 users can only do Excel-level changes to the worksheet, such as change the color, font, and styles, but cannot interact with PivotTables and PivotCharts. When users upgrade PowerPivot, they can use the workbooks created in the previous versions but the opposite is not possible.

This chapter will cover the following topics:

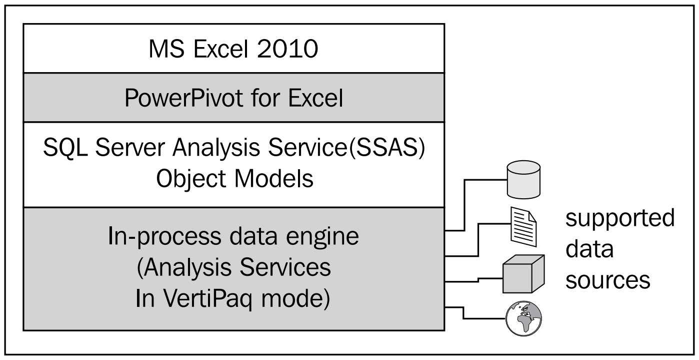
- The principle of PowerPivot for Excel
- The benefits of using PowerPivot
- Installing PowerPivot
- Verification of the installation

The principle of PowerPivot for Excel

PowerPivot is a powerful data analysis tool with advanced reporting; it leverages self-service business intelligence for everyone. PowerPivot is a free add-in for Microsoft Excel 2010 that enables the user to import massive amounts of data from a variety of sources and manipulates it into one workbook, which means PowerPivot helps in solving heterogeneous problems. Millions of rows of data can be accessed in seconds.

PowerPivot is a business intelligence tool of Microsoft, known as **Microsoft Business Intelligence** (**MSBI**). In PowerPivot, the features, functionalities, calculations, and report visualization are like other traditional BI tools. PowerPivot is based on xVelocity in-memory technologies, providing unmatched analytical performance and calculation capabilities to process billions of rows at the speed of thought using **Data Analysis Expressions** (**DAX**) functions. Basically, DAX is a business logic of PowerPivot and is used for data manipulation. DAX functions are very basic and easy to learn, even for a person who doesn't have a technical background.

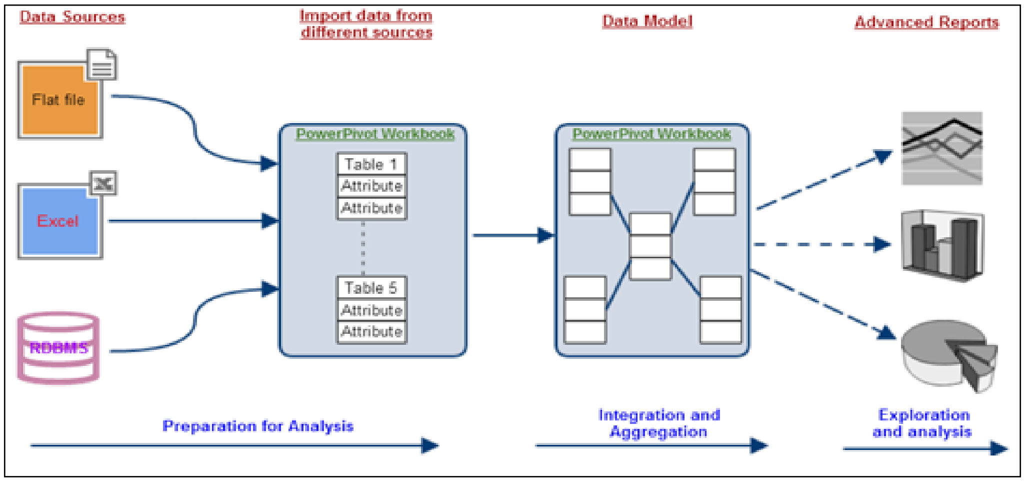
The following figure explains the architecture of PowerPivot and how it works in Excel:


MS Excel 2010
PowerPivot for Excel
SQL Server Analysis Service(SSAS) Object Models
In-process data engine (Analysis Services In VertiPaq mode)
supported data sources

Architecture of PowerPivot in Excel

PowerPivot for Excel uses the **Analysis Services xVelocity in-memory analytics** engine (**VertiPaq**) that runs in-process in Excel and helps in importing data from a variety of data sources. The data that the user adds to the workbook is stored internally just like an embedded PowerPivot database inside the .xlsx file and data processing is done through this engine; it compresses and loads the data and makes it available as visualization objects, such as PivotTables and PivotCharts, in a worksheet. There are no administration or configuration tasks that need to be performed in PowerPivot. The engine is an internal component of PowerPivot, which is an add-in to Excel.

PowerPivot gives drag-and-drop methods to users. Users can perform aggregation, filtering, create metrics by using DAX, and create diverse reports like PivotTables, PivotCharts, and KPIs. There are three principles, namely, preparation, integration and aggregation, and data analysis. There will be brief explanations about PowerPivot in this book, based on the following figure:



Principles of PowerPivot

Benefits of using PowerPivot

Today it's quite difficult to find successful industries that have not leveraged BI technologies. Hence, everyone seeks BI tools for their data analysis and reporting purposes in order to make business decisions. In that case, PowerPivot meets their expectations and produces advanced reporting.

Most organizations spend a lot of time using Excel for data analysis, but don't allocate adequate time to fully analyze the data. By using PowerPivot, you can analyze the data in less time. It also has more advanced features than Excel, along with a seamless integration with the powerful visualization features of Excel. Some organizations have stored their data in different kinds of sources such as RDBMS, flat files, and CSVs. For these organizations, usage of ETL tools to create integration and data models and usage of some reporting tools for data analysis and reporting will be very helpful. However, since many forms of BI and ETL products are quite expensive for small organizations, PowerPivot may be used instead, since it is very feasible for these kinds of processes.

The vital advantage of PowerPivot is that you can do the analysis without the knowledge of **SQL** and **Multidimensional Expressions** (**MDX**). It's very user-friendly and has an easily understandable **Graphical User Interface** (**GUI**).

Installing PowerPivot

Here you can find the installation requirements of the PowerPivot add-in for Microsoft Excel 2010 only. There is no need to follow this installation requirement for the Excel 2013 PowerPivot version, since the PowerPivot add-in would have already installed when the installation of your Microsoft Office 2013 package was completed. So, if you have successfully installed the Excel 2013 version on your computer, you only have to enable the PowerPivot add-in. You will find related information in the following pages on how to enable your PowerPivot add-in for Excel 2013.

Requirements (32-bit and 64-bit versions of Windows)

Machines that will be used to create PowerPivot workbooks must meet the following minimum hardware and software requirements for Office 2010.

Hardware requirements

Before installation of the PowerPivot add-in for Excel 2010, make sure that the following hardware requirements are met for the machine on which the PowerPivot is being loaded:

Component	**Minimum requirements**
Processor	500 MHz, 32-bit or 64-bit.
RAM	2 to 4 GB of RAM is feasible. The add-in will take 25 MB of RAM and additionally 33 MB will be filled when the first PivotTable is added to the worksheet. The 32-bit version of PowerPivot enables you to work with up to 2 GB of data in memory and the 64-bit version of PowerPivot enables you to work with up to 4 GB of data in memory.
Storage space	100 MB is required for program files and additional disk space is required for storing the workbook. The storage will vary based on the amount of data that you are going to analyze using PowerPivot. There is no way to know the disk requirement in advance, so it is advisable to check the disk space before saving the workbook.

Software requirements

Make sure the following software requirements are met for the machine on which PowerPivot is being loaded:

Component	**Minimum requirements**
Operating system	Windows 7 and 8, Windows Vista SP2, and Windows XP SP3 (32-bit only).
Windows features	Microsoft .NET Framework 4.0 or Windows 8 Microsoft .NET Framework 4.5 (to be installed before installing Microsoft Office).
Excel version	Microsoft Excel 2010 (either 32-bit or 64-bit).
Office features	Office shared features must be installed with Excel 2010.
	.NET programmability support in Microsoft Excel must also be installed.
	Visual Studio 2010 tools for Office runtime (to be installed before or after Office, but before using PowerPivot for Excel).

Recommendations

The 64-bit version and the 32-bit version of Microsoft Excel PowerPivot have some differences. The 32-bit version, as compared to the 64-bit version, lacks some important features for large data users. The 32-bit version of Windows allows only up to 1 million rows, which sums up to 2 GB of data to be managed and 500–700 MB of file size. If this limit is exceeded, data inconsistencies can occur. When using such large volumes of data, it is recommended you switch to Windows 64-bit, since the 64-bit version of PowerPivot can manage 4 GB of data and 2 GB of file size. But in Excel 2013, for the 64-bit version of PowerPivot, all these limitations have been removed. It means that, the only restriction that is placed on the physical data is that of one's own machine.

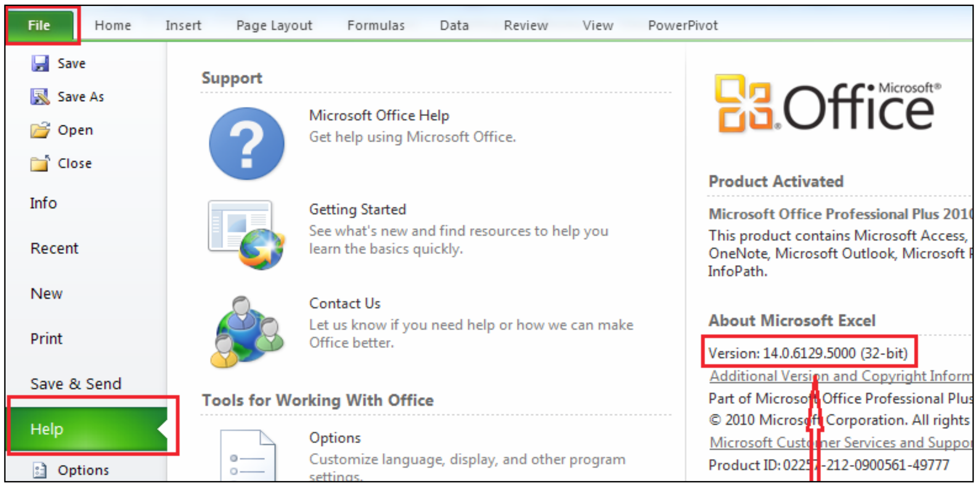
Users of the 32-bit Windows version of Microsoft PowerPivot will be able to manage data satisfactorily as long as the 1 million row limit is observed. Fast and efficient performance analysis with sufficient amounts of memory for larger volumes of data is attainable for those using the 64-bit version of Windows.

Getting the PowerPivot add-in for Excel 2010

Before downloading the free PowerPivot add-in, you have to determine if you are using the 32-bit or 64-bit version of Excel 2010. If you have installed the 32-bit version of Excel, you must install the 32-bit version of PowerPivot for Excel. Likewise, if you have installed the 64-bit version of Excel, you must install the 64-bit version of PowerPivot for Excel.

In order to know which version of Excel 2010 you have installed on your computer, perform the following steps:

1. Open Microsoft Excel 2010.
2. Click on **File** from the menu bar.
3. Select **Help**.
4. Note the version of Excel you have installed on your computer.



Microsoft Office Help
Get help using Microsoft Office.

Getting Started
See what's new and find resources to help you learn the basics quickly.

Contact Us
Let us know if you need help or how we can make Office better.

Options
Customize language, display, and other program settings.

Product Activated
Microsoft Office Professional Plus 201
This product contains Microsoft Access, OneNote, Microsoft Outlook, Microsoft InfoPath.

About Microsoft Excel
Version: 14.0.6129.5000 (32-bit)
Additional Version and Copyright Inform
Part of Microsoft Office Professional Plus
© 2010 Microsoft Corporation. All rights
Microsoft Customer Services and Suppo
Product ID: 02257-212-0900561-49777

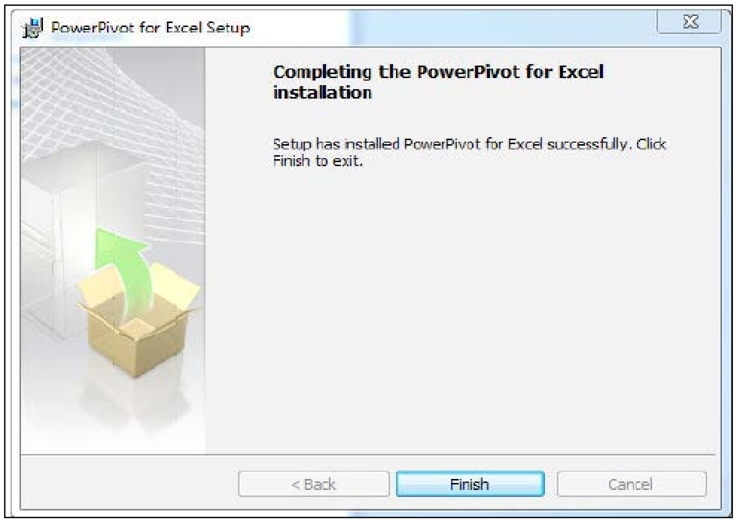
Identifying the version of Excel 2010

Downloading and installing the PowerPivot add-in for Excel 2010

The download and installation of the PowerPivot add-in for Excel 2010 is easily done with the help of the following instructions:

1. Go to `http://www.microsoft.com/en-us/download/details.aspx?id=29074`.
2. Select the language and click on the **Download** button.

3. Select the version you want to download and click on the **Next** button.
4. Once you have clicked on the **Next** button, your setup file will be downloaded.
5. Double-click on the setup file and then click on **Run**.
6. Click on **Next** to get started.
7. Accept the license agreement and then click on **Next**.
8. Click on **Install**.
9. Click on **Finish**.



Completing the PowerPivot for Excel installation

Setup has installed PowerPivot for Excel successfully. Click Finish to exit.

Completing the installation

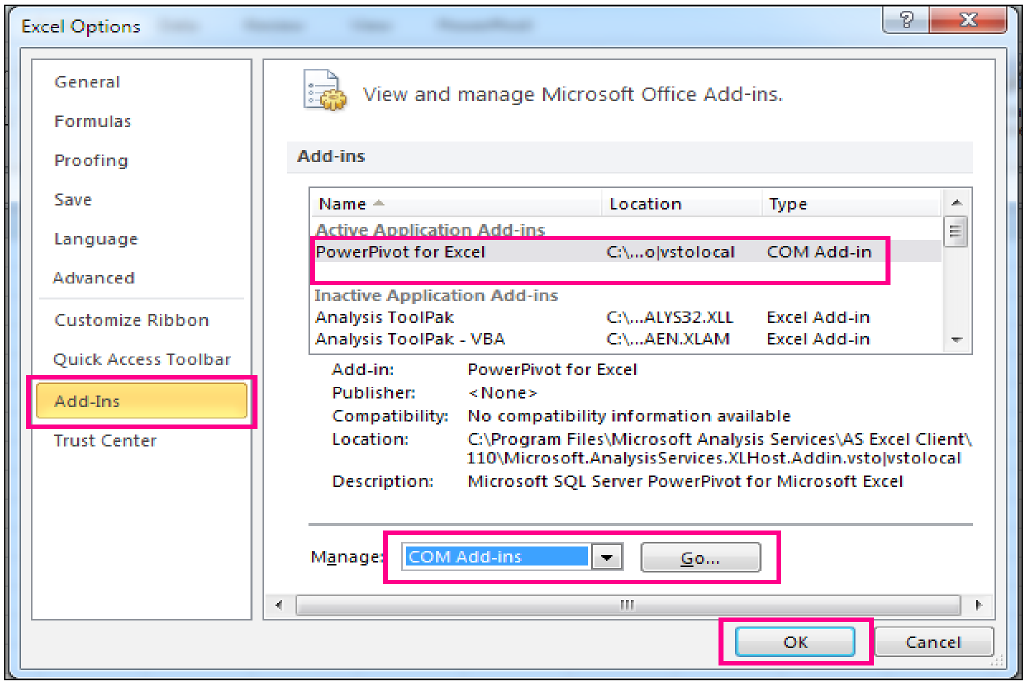
Enabling the PowerPivot add-in for Excel 2013

If PowerPivot in Excel 2013 has already been installed on your Microsoft Office 2013 Professional Plus package, but wasn't enabled, to enable the add-in, perform the following instructions:

1. Open Excel 2013.
2. Navigate to **File** | **Options** | **Add-Ins**.
3. In the **Manage** box select **COM Add-ins** and click on the **Go...** button.
4. Check the **PowerPivot for Excel** box, and then click on **OK**.

If you have other versions of the PowerPivot add-in installed, those versions will also be listed in the COM Add-ins list. Be sure to select the PowerPivot add-in for Excel 2013.

5. Now, enable the PowerPivot add-in in your Excel 2013.



View and manage Microsoft Office Add-ins.

Add-ins

Name	Location	Type
Active Application Add-ins		
PowerPivot for Excel	C:\...o\vstolocal	COM Add-in
Inactive Application Add-ins		
Analysis ToolPak	C:\...ALYS32.XLL	Excel Add-in
Analysis ToolPak - VBA	C:\...AEN.XLAM	Excel Add-in

Add-in:	PowerPivot for Excel
Publisher:	<None>
Compatibility:	No compatibility information available
Location:	C:\Program Files\Microsoft Analysis Services\AS Excel Client\110\Microsoft.AnalysisServices.XLHost.Addin.vsto\vstolocal
Description:	Microsoft SQL Server PowerPivot for Microsoft Excel

Manage: COM Add-ins

Enabling the PowerPivot add-in

If the PowerPivot add-in does not appear in the Excel ribbon, you have to troubleshoot using the following instructions:

1. Open Excel 2013.
2. Navigate to **File** | **Options** | **Add-Ins**.
3. In the **Manage** box, select **Disabled Items** and then click on **Go**.
4. Select **PowerPivot for Excel** and then click on **Enable**.
5. Now PowerPivot will be enabled.

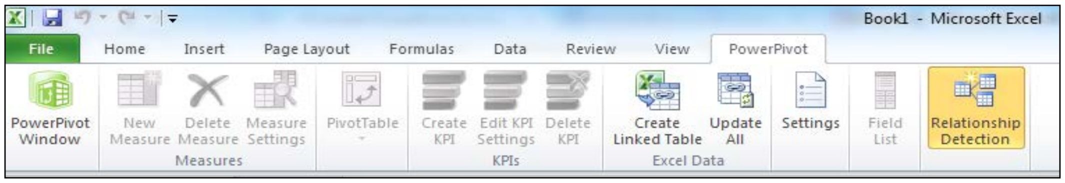
Verifying installation

After installing the PowerPivot add-in for Excel 2010 or enabling the PowerPivot add-in for Excel 2013, you must verify and test the add-in to check if it has been installed and enabled properly.

Testing the PowerPivot add-in

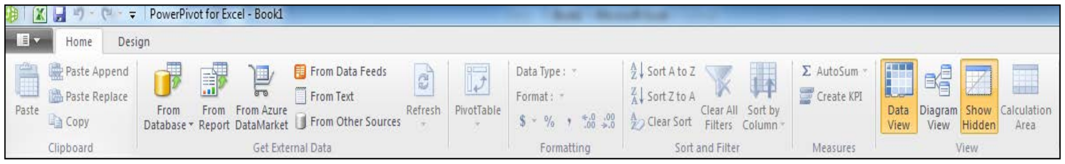
In order to test the PowerPivot add-in, perform the following steps:

1. Open Excel. Your **PowerPivot** tab will be displayed on the Excel ribbon, as shown in the following figure:



PowerPivot tab on the Excel ribbon

2. Click on **PowerPivot Window**. A new PowerPivot workbook will open, as shown in the following figure. From here, data will be imported from a variety of sources. Creation of a data model and making advanced reports will also be done here. Upon completion of this book, users will have a clear understanding of these wizards.



PowerPivot workbook

Summary

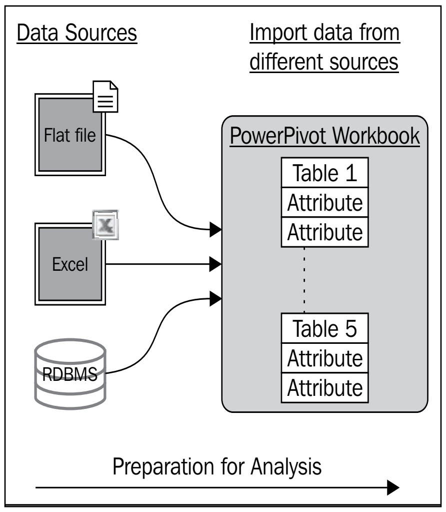
This chapter illustrates quick installation, some features and differences of Excel 32-bit and 64-bit versions of PowerPivot, and the vital information about the principles of PowerPivot, which will leverage the user's ability to work on self-service business intelligence.

In the next chapter, a brief description about the first principle (preparations for data analysis) will be given, which will explain the different types of data sources a user can import and how different types of data sources will help the users in their analysis. These functionalities will correlate with some real-time examples, helping the user gain sufficient understanding to make Microsoft PowerPivot easy.

2
Preparation Analysis of Data Source

After installing PowerPivot and getting to know the basic concepts, we will now take a deep dive into data sources and the different types of data sources that can be imported to the PowerPivot interface.

As discussed in *Chapter 1*, *Principles and Installing PowerPivot*, there are three principles in PowerPivot; the first principle is **Preparation**, which is displayed in a red colored box in the following figure:



Preparation for data analysis

The most important thing is that a lot of users may have both structured and unstructured data coming from a variety of sources (for their urgent data analysis). The usage of the DAX formula in PowerPivot will solve this heterogeneous problem. Examples explaining the structured and unstructured data are given in the *General Overview of ETL* section of this chapter.

The Microsoft Excel 2010 PowerPivot add-in is being used for our practical purposes and the screenshots are explained too. There is no comparable difference between Excel 2013 and 2010. However, the Excel 2013 version has some advantages.

The following topics will be covered in this chapter:

- List of data sources
- Purpose of import data from a variety of sources
- Adding a data source in a single worksheet from a variety of data sources
- Data refresh

List of data sources

Here, the wide varieties of data sources that are supported in the PowerPivot interface are given in brief. The vital part is to install providers such as **OLE DB** and **ODBC** that support the existing data source, because when installing the PowerPivot add-in, it will not install the provider too and some providers might already be installed with other applications.

For instance, if there is a SQL Server installed in the system, the OLE DB provider will be installed with the SQL Server, so that later on it wouldn't be necessary to install OLE DB while adding data sources by using the SQL Server as a data source. Hence, make sure to verify the provider before it is added as a data source.

Perhaps the provider is required only for relational database data sources.

Relational database

By using RDBMS, you can import tables and views into the PowerPivot workbook. The following is a list of various data sources:

- Microsoft SQL Server
- Microsoft SQL Azure
- Microsoft SQL Server Parallel Data Warehouse
- Microsoft Access
- Oracle
- Teradata
- Sybase
- Informix
- IBM DB2
- Other providers (OLE DB / ODBC)

Multidimensional sources

Multidimensional data sources can only be added from Microsoft Analysis Services (SSAS).

Data feeds

The three types of data feeds are as follows:

- SQL Server Reporting Service (SSRS)
- Azure DataMarket dataset
- Other feeds such as atom service documents and single data feed

Text files

The two types of text files are given as follows:

- Excel file
- Text file

[You could also use OLE DB instead of the ODBC provider in order to import your data from RDBMS. While choosing from different providers for importing the same data source, it is recommended to use OLE DB as it speeds up the importing process.]

Purpose of importing data from a variety of sources

In order to make a decision about a particular subject area, you should analyze all the required data that is relevant to the subject area. If the data is stored in a variety of data sources, importing the data from different data sources has to be done. If all the data is only in one data source, only the data needs to be imported from the required tables for the subject and then various types of analysis can be done.

The reason why users need to import data from different data sources is that they would then have an ample amount of data when they need to make any analysis. Another generic reason would be to cross-analyze data from different business systems such as **Customer Relationship Management** (**CRM**) and **Campaign Management System** (**CMS**). Data sourcing from only one source wouldn't be as sophisticated as an analysis done from different data sources as the amount of data from which the analysis was done for multisourced data is more detailed than the data from only a single source. It also might reveal conflicts between multiple data sources.

In real time, usually in the e-commerce industry, blogs, and forum websites wouldn't ask for more details about customers at the time of registration, because the time consumed for long registrations would discourage the user, leading to cancellation the of the order.

For instance, the customer table that would be stored in the database of an e-commerce industry would contain the following attributes:

Customer
FirstName
LastName
E-mail
BirthDate
Zip Code
Gender

However, this kind of industry needs to know their customers more in order to increase their sales. Since the industry only saves a few attributes about the customer during registration, it is difficult to track down the customers and it is even more difficult to advertise according to individual customers. Therefore, in order to find some relevant data about the customers, the e-commerce industries try to make another data source using the Internet or other types of sources.

For instance, by using the Postcode given by the customer during registration, the user can get Country | State | City from various websites and then use the information obtained to make a table format either in Excel or CSV, as follows:

Location
Postcode
City
State
Country

So, finally the user would have two sources—one source is from the user's RDBMS database and the other source is from the Excel file created later—both of these can be used in order to make a customer analysis based on their location.

General overview of ETL

The main goal is to facilitate the development of data migration applications by applying data transformations.

Extraction Transform Load (**ETL**) comprises of the first step in a data warehouse process. It is the most important and the hardest part, because it determines the quality of the data warehouse and the scope of the analyses the users will be able to build upon it.

Let us discuss in detail what ETL is.

The first substep is extraction. As the users would want to regroup all the information a company has, they will have to collect the data from various heterogeneous sources (operational data such as databases and/or external data such as CSV files) and various models (relational, geographical, web pages, and so on).

The difficulty starts here. As data sources are heterogeneous, formats are usually different, and the users would have different types of data models for similar information (different names and/or formats) and different keys for the same objects. These are some of the main problems.

The aim of the transformation task is to correct such problems, as much as possible.

Let us now see what a transformation task is.

The users would need to transform heterogeneous data of different sources into homogeneous data. Here are some examples of what they can do:

- Extraction of the data sources
- Identification of relevant data sources
- Filtering of non-admissible data
- Modification of formats or values

Examples

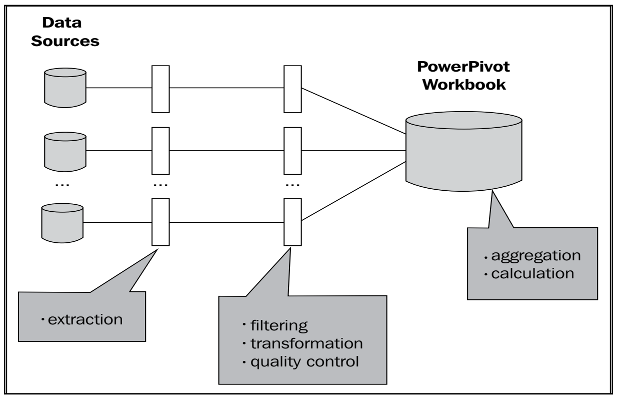
The following is a list of examples for various types of functions:

- Conversion and normalization functions:
    - Uppercase→lowercase, date format (dd/mm/yy→dd/mm/yyyy)
    - Mapping tables (0/1→F/M)
- Cleaning the domain-specific data (for example, names and addresses):
    - Valid postal codes, synonyms, and abbreviation dictionaries (for example, Mr., Ms., Monsieur, Sr.)
- Correspondence-search algorithms:
    - Same key, same prefix, and some equal values (fuzzy matching)

PowerPivot performance in ETL

Generally, for the ETL process we have to use some specific tools to perform this kind of a process, but in PowerPivot it is not necessary to use any of those specific tools in order to create the ETL tier process. DAX and PowerPivot features are adequate to complete the ETL process. PowerPivot accomplishes the ETL processes for a medium amount of data, but not large chunks of data.

The following figure illustrates how PowerPivot performs the ETL process. The users will be able to do data transformation, filtering, cleaning, data format changing, and aggregations with the help of PowerPivot features and DAX expressions.



The ETL process in PowerPivot

Introducing Data Analysis Expressions (DAX)

This is the most vital functionality in PowerPivot interfaces. Data Analysis Expressions, or simply DAX, is used to achieve our second principle, **Integration** and **Aggregation**. The second principle relies on DAX since the PowerPivot interface has more tables and connects with a relational table in order to analyze. In the preceding figure, the functionalities in green-colored boxes can be done mostly by DAX.

DAX is more or less similar to the Excel formulae and it is not a programming language. Explicitly, its understandable functionalities are easy for the users to learn. You can use the DAX formula either in PowerPivot tables or PivotTables.

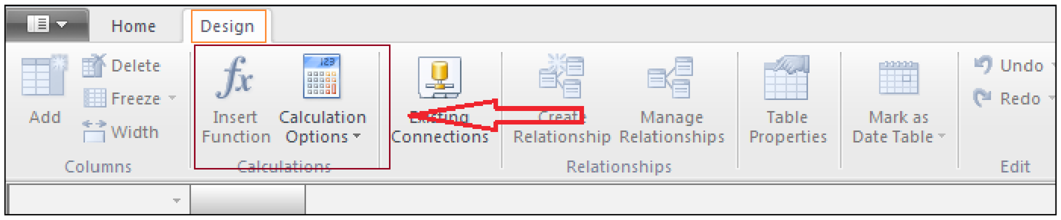
Unlike Excel, the DAX formula in PowerPivot cannot be applied for rows or for particular cells. Hence the users must use DAX for either the entire table or entire column. The users can create calculated columns, metrics, and can make relationships with other tables using DAX.

The functionalities are as follows:

- Date and time functions
- Filter functions
- Information functions
- Logical functions
- Mathematical functions
- Statistical functions
- Text functions
- Time intelligence functions

These eight functions will be helpful to change data from unstructured to structured from a variety of sources. The usage of these functionalities in the PowerPivot interface will solve the problem with heterogeneous data.

Open the **PowerPivot** window and select the **Design** tab and the users will be able to view the **fx** button, as shown in the following figure:



DAX function screen

Once the users click on the **fx** button, it will list out all the functions that are available in PowerPivot and the users can select whichever function they want and write their custom formula. Another important thing is that DAX is useful for creating custom calculations for PowerPivot tables.

DAX expressions start with an equals (=) sign and the users can select a function from the **fx** button or write an expression. The most important thing is that the users can use the nested function up to 64 levels of the function in calculated columns; but nesting will be difficult, so it is recommended to be careful when writing nested functions.

An example of a DAX formula is given as follows:

To concatenate two columns such as First_Name and Last_Name in a customer table, just add a new column in the PowerPivot table. The following syntax has to be used:

```
CONCATENATE (<text1>, <text2>)
```

An example for this type can be written as follows:

```
CONCATENATE (Customer [First_Name], CONCATENATE ("", "", Customer
[Last_Name]))
```

To find the month from the `date` attribute, the following syntax has to be used:

```
MONTH (<date>)
```

It will return a number between 1 and 12 based on the corresponding date.

To find the age from the `BirthDate` attribute, use the following syntax:

```
Age =Year(NOW()) - Year([BirthDate])
```

More precisely, DAX functions will be used in the next chapter with some practical examples to see how it works. Here, only an introduction to DAX is given, since it's important in the importing of the data source and since it is one of the most prominent features in solving heterogeneous problems for unstructured data.

Adding a data source in a single worksheet from a variety of data sources

In this chapter, we've been looking at the preparation of data for analyses and creation of advanced powerful reports in more detail. This will give the readers a general idea about the preparation of data analysis.

This topic will show the readers how to import that data from a variety of data sources in the PowerPivot interface; analysis of two types of industry data will be done—one is of e-commerce customer activities and another one is of sales analysis data—which was provided by Microsoft for training.

For a better understanding of the users, a brief explanation of some of the terms that will be used from here on is given as follows:

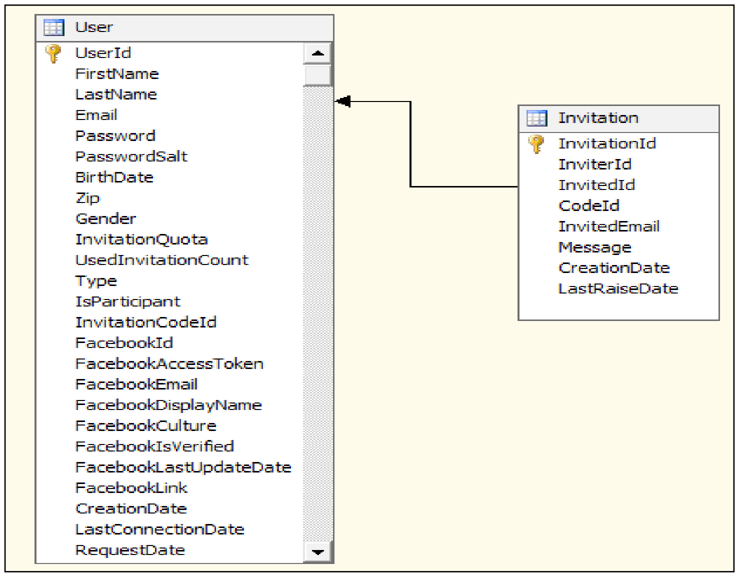
- **PowerPivot interface/workspace**: This is a user interface/ workspace of the PowerPivot application that will be visible to the user.
- **PowerPivot workbook**: This is a space where all the tasks are done in PowerPivot.
- **PowerPivot table**: Everything that is imported from different data sources into the PowerPivot workbook is called a PowerPivot table.
- **PowerPivot window**: This is a window in the Excel workbook, which launches the PowerPivot application.

E-commerce customer activity data importing

Data importation of the e-commerce industry data from the SQL Server database will be discussed in this section. There are two tables, **User** and **Invitation**, stored in a SQL Server; it's attribute details are shown in the next figure.

The **User** table consists of customer detail attributes such as **FirstName**, **LastName**, **BirthDate**, and **Zip** code, and if any user has registered using Facebook, the user's table will also have their Facebook details. The **Invitation** table consists of **InviterId**, **InvitedId**, **Email id**, **Message**, **CreationDate**, and **LastRaiseDate** attributes.

Apparently, the e-commerce industry mainly focuses on inviter in order to improve the number of customers. So, they put up some special offer for the inviters.
Take a look at the relationship between user and invitation tables in the following screenshot:

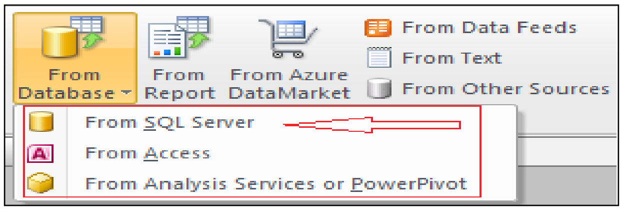


Relationship between user and invitation table

Importing data from the SQL Server

In order to import data from the SQL Server, perform the following steps:

1. Open a new Excel file and select the **PowerPivot** tab. By clicking on the **PowerPivot** window, a new PowerPivot window will open. Then select the **Home** tab.
2. By clicking on the **From Database** icon, a list of sources as shown in the following figure will be displayed. Click on the **From SQL Server** icon.



Choosing the SQL server

3. After clicking on the icon, the **Table Import Wizard** window will be displayed, which is used for the database connection. Here, the **Friendly connection name** is **SqlServer**, a **Server name** is given, and a **Database name** is selected from the list of databases created by the users. To test the connection, click on the **Test Connection** button in order to confirm that the database connection is successful. After receiving the success message, click on the **Next** button, otherwise, troubleshoot the database connection.

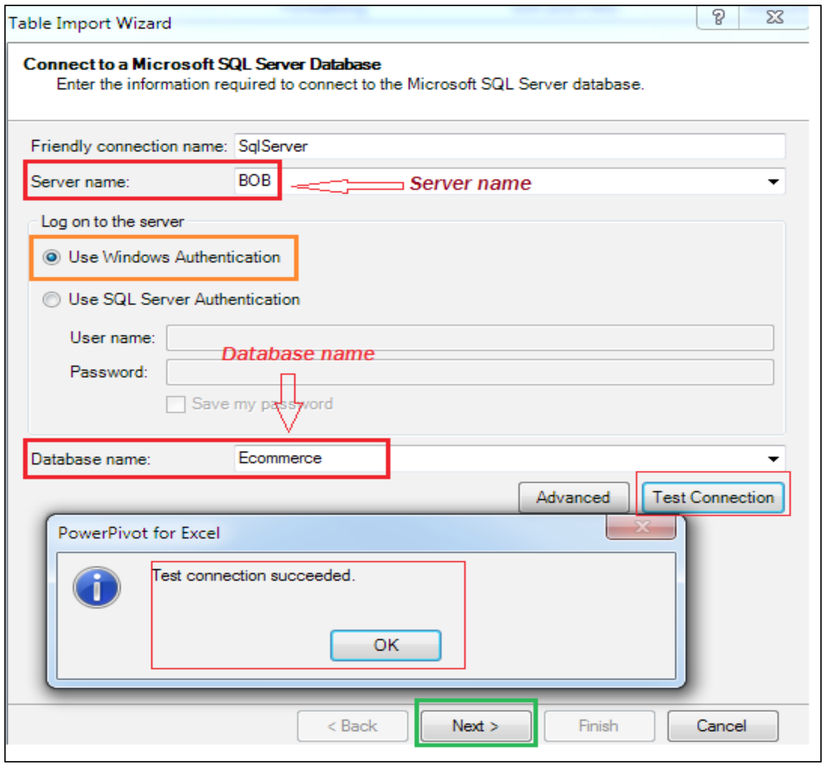


Table Import Wizard

Connect to a Microsoft SQL Server Database
Enter the information required to connect to the Microsoft SQL Server database.

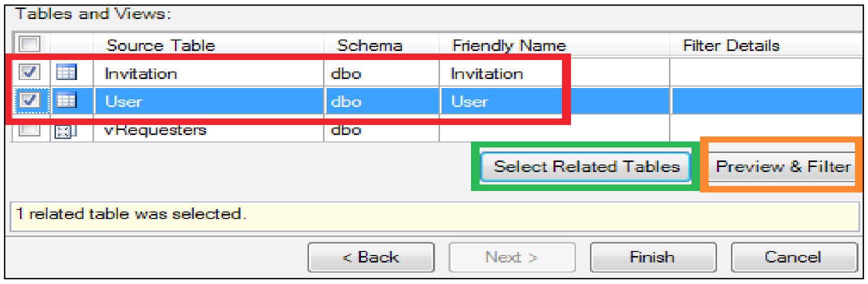
Friendly connection name: SqlServer
Server name: BOB
Log on to the server
Use Windows Authentication (selected)
Use SQL Server Authentication
User name:
Password:
Save my password
Database name: Ecommerce

PowerPivot for Excel
Test connection succeeded.

SQL Server Connection Wizard

4. After clicking on the **Next** button, the users have to choose how to import data. It will show two options, which are **Select from a list of tables and views to choose the data to import** and **Write a query that will specify the data to import**. In the current example, the first option has been selected since the data importation is being done in the form of tables from the database and these options can only be used for relational databases.

5. By clicking on the **Next** button, it will list all the **Tables** and **Views** of the database. The salient thing here is the **Select Related Tables** button. Select one table and click on this button and it will automatically select all the related tables (the relationship here is based on foreign keys) and show the number of related tables selected. This helps in easy identification of the related tables for analysis. In the example being given here, there are two tables. User tables are related to the invitation table; select the **User** table and click on the **Select Related Tables** button. It will automatically check the **Invitation** table because of the relationship.


Tables and Views:

☐		Source Table	Schema	Friendly Name	Filter Details
✓		Invitation	dbo	Invitation	
✓		User	dbo	User	
☐		vRequesters	dbo		

1 related table was selected.

Selecting related tables

6. The **Preview & Filter** button will help you to preview the data we've created and managed so far. Uncheck the column name if it is an attribute that's not required for the analysis and filter the data based on requirements before importing the data with this button. In this example, the password attributes from the **User** table are being ignored since it's not necessary for analysis and none of the data is being filtered because all the data is necessary. After clicking on the **Preview & Filter** button, the result is shown in an Excel table as shown in the following figure. If there are no columns left unchecked, click on the **Finish** button, otherwise, click on **OK** and then click on **Finish**.

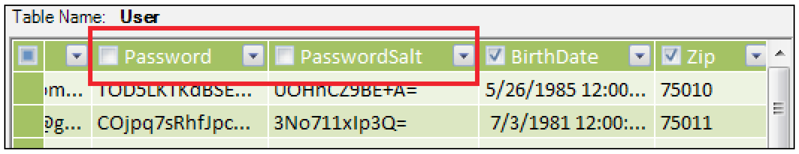
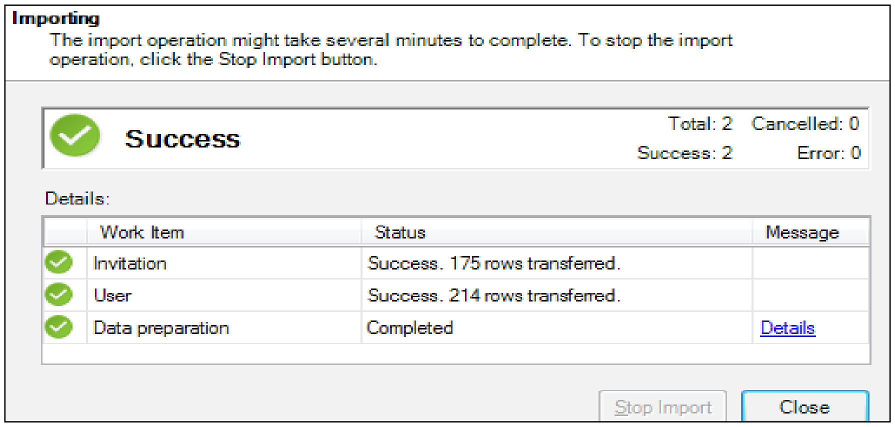


Table Name: **User**

	☐ Password	☐ PasswordSalt	✓ BirthDate	✓ Zip
m...	TOD5LKTKdBSE...	UOHnCZ9BE+A=	5/26/1985 12:00...	75010
g...	COjpq7sRhfJpc...	3No711xIp3Q=	7/3/1981 12:00:...	75011

Preview & Filter

7. After clicking on the **Finish** button, it will show the progress of the data that is being imported. If there is a significant amount of data that needs to be processed, it will take several minutes to import all of it. After this is finished, it will show the status of importing statistics by which the users will be able to make sure whether or not all the data has been imported.



Importing
The import operation might take several minutes to complete. To stop the import operation, click the Stop Import button.

Success

Total: 2 Cancelled: 0
Success: 2 Error: 0

Details:

	Work Item	Status	Message
✓	Invitation	Success. 175 rows transferred.	
✓	User	Success. 214 rows transferred.	
✓	Data preparation	Completed	Details

Importing status

8. After successful import, click on the **Close** button and we will see two tables in the PowerPivot interface; these are the tables that are being imported. Here the user will be able to check all the attributes. It would look like an Excel sheet, but with more graphical content. At the bottom the names of the tables will be given. When selected, all the data in the table will be shown. For instance, if there is an Excel file with multiple sheets, they will be shown at the bottom of the file with the sheet name. PowerPivot also has the same kind of interface. Users must stay on the same PowerPivot workbook since the importing of another data source will be done for the same analysis.

Importing data from a flat file

As discussed previously, by navigating to Postcode | Country | State | City, the data was found and retrieved from a different source. Here the data was found from the Internet and was later stored as a text file, since the User table only had a zip code attribute. It is this kind of data that is needed if the users need to analyze customer activities with geographical locations. The following steps explain how to import data from a flat file:

1. Find the **From Text** icon in the external data section in order to import a flat file; it will show the Table Import Wizard. Here, enter the information required to get data from the flat files.
2. Here the data is stored in a local drive. Select the data using the **Browse** button; the most important thing to do is to select what column separator was used to separate the column in the flat file. Generally, there are six column separators in the flat file, such as **Tab**, **Comma**, **Semicolon**, **Space**, **Colon**, and **Vertical bar**. Here the **Tab** separator was used, since the data was stored as a tab. Then, if the data that was imported has the column names in the first row, check **Use first row as column headers**, otherwise, it will automatically create fields such as **F1**, **F2**,...**Fn**, and then the user can rename the column names as required. In the flat file shown in the following figure, there is no column in the first row, hence the **Use first row as column headers** was not selected.
3. Click on the **Finish** button.

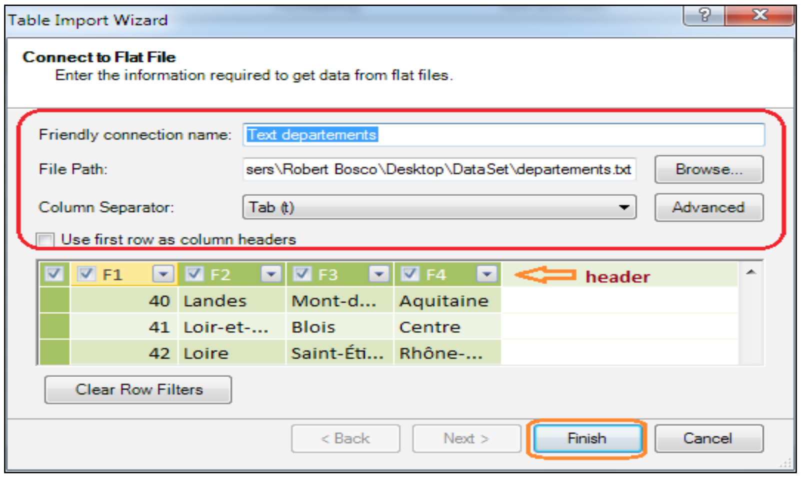


Table Import Wizard

Connect to Flat File
Enter the information required to get data from flat files.

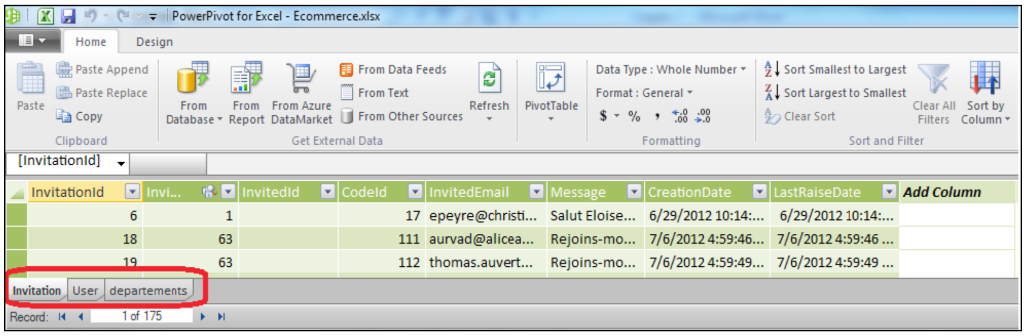
Friendly connection name: Text departements
File Path: sers\Robert Bosco\Desktop\DataSet\departements.txt
Column Separator: Tab (t)
Use first row as column headers

F1	F2	F3	F4
40	Landes	Mont-d...	Aquitaine
41	Loir-et-...	Blois	Centre
42	Loire	Saint-Éti...	Rhône-...

Flat file Table Import Wizard

4. After clicking on the **Finish** button, the flat file data will be transferred to PowerPivot while showing importing statistics as shown in the *Importing status* figure.

5. Finally, after successfully importing three tables into one **PowerPivot** window for the e-commerce industry's customer activities analysis, the following figure shows the three tables that were imported:



InvitationId	Invi...	InvitedId	CodeId	InvitedEmail	Message	CreationDate	LastRaiseDate	Add Column
6	1		17	epeyre@christi...	Salut Eloise...	6/29/2012 10:14:...	6/29/2012 10:14:...	
18	63		111	aurvad@alicea...	Rejoins-mo...	7/6/2012 4:59:46...	7/6/2012 4:59:46 ...	
19	63		112	thomas.auvert...	Rejoins-mo...	7/6/2012 4:59:49...	7/6/2012 4:59:49 ...	

E-commerce PowerPivot window

In order to save the PowerPivot workbook, use the Excel window that is already open. The Excel workbook that was started and PowerPivot are stored in the `*.xlsx` format.

[When a project is started, it is important to save the workbook, since multiple sources will be imported, and doing this would avoid data loss due to system crash or power failure.]

Sales analysis data importing

Microsoft has provided some sample sales data for PowerPivot testing; the sales store name is **Contoso**. Here, this data will also be imported. In order to make the analysis, users can download the sample data from `http://powerpivotsdr.codeplex.com/`. The folder contains five files, namely, `ContosoSales`, `Geography`, `ProductCategories`, `SQLQuery`, and `Stores`.

- `ContosoSales`: This is stored as an Access database file, and it contains the sales transaction details of Contoso
- `Geography`: This is stored as an Excel file and contains only one sheet containing the geographical details of Contoso selling products
- `ProductCategories`: This is also stored as an Access database file, but it has only one table containing product categories of Contoso
- `SQLQuery`: This is a text file, which contains a SQL query in order to import data by using SQL
- `Stores`: This is stored in the Excel file format, which contains one sheet containing the store details

Importing data from the MS Access database

Here, data is going to be imported from `ContosoSales`. However, since the data is stored in an MS Access database file, the user should perform the following steps to import data by using MS Access:

1. Open a new Excel file for sales analysis data import, and open the **PowerPivot** window.
2. As mentioned previously, save the workbook at the beginning of the project. Here, we have saved it as `Sales Analysis`.
3. By clicking on **From Database**, a list of three sources will appear as shown in the *Choose the SQL Server* figure, select **From Access**.
4. A Table Import Wizard for the MS Access database connection will be shown, but it's quite different from the SQL Server.
5. Enter `Contoso` in the **Friendly connection name** field.
6. Browse the database file where the downloaded sample data is located by using the **Browse** button, and select the `ContosoSales.accdb` file.
7. In the Table Import Wizard, there is a **Log on to the database** field. If your Access database contains a password, the user must provide the username and password, but the sample data will not be encrypted in the password, therefore, it is just avoided.
8. Click on the **Test Connection** button in order to find out the status of the connection.

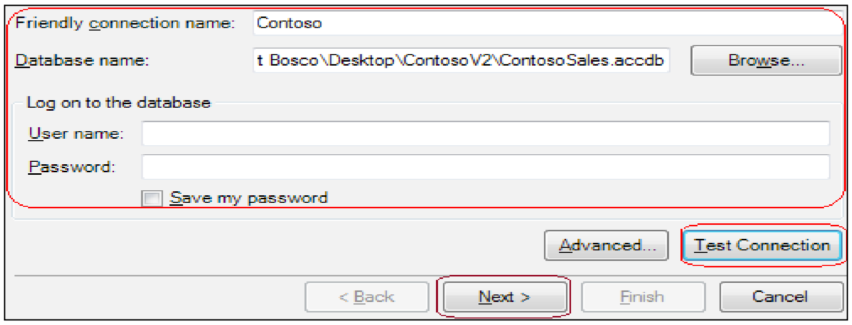
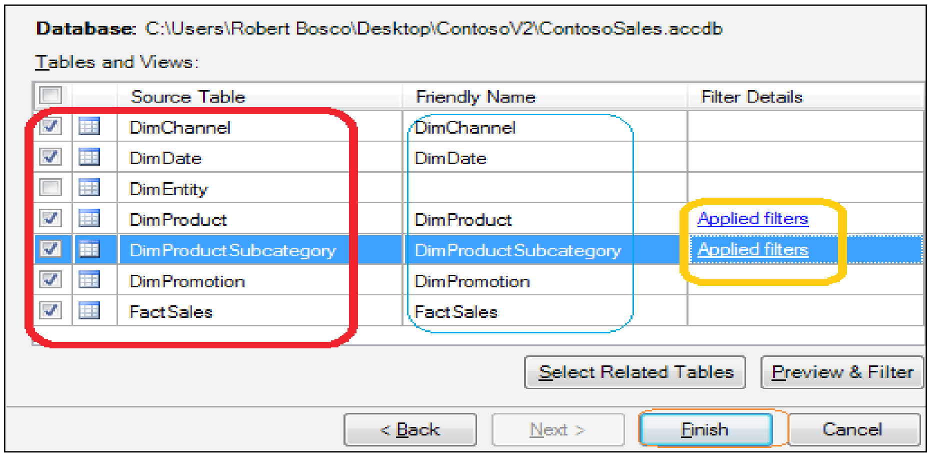

Friendly connection name: Contoso
Database name: t Bosco\Desktop\ContosoV2\ContosoSales.accdb
Log on to the database
User name:
Password:

Table Import Wizard for MS Access

9. Click on **Next** and check **Select from a list of tables and views** to choose the data to import.

10. Click on **Next** and all the tables stored in Contoso DB will be displayed. Select the tables that should be imported for analysis. Select the following tables: **DimChannel**, **DimDate**, **DimProduct**, **DimProductSubcategory**, and **FactSales**.
11. It has already been explained that the **Preview & Filter** button is used while importing that data by using the SQL Server. This is in order to fill the data and ignore unwanted columns for analysis. Select the **DimProduct** table that will turn the background row into blue, as shown in the following figure (in which the **DimProductSubcategory** table has been selected), and then click on **Preview & Filter**. The preview for the selected table opens a dialog box with all the columns in the **DimProduct** table displayed as a list. Uncheck the **ClassID**, **StockTypeID**, **ETLLoadID**, **ProductURL**, and **ImageURL** attributes, since all these are unnecessary for the analysis. In the same way, select the **DimProductSubcategory** table, and then click on **Preview & Filter**. The preview for the selected table is shown in a dialog box. At the top of the Product Category key column, click on the arrow on the right-side of the cell, scroll down, unselect **7** and **8**, and then click on **OK**. The unselected options are games and home appliances, which can be ignored since there are no games and home appliance products in the Product table.
12. In the **Friendly connection name** field, change the table name to whatever is the client's/user's wish.


Database: C:\Users\Robert Bosco\Desktop\ContosoV2\ContosoSales.accdb

Tables and Views:

☐	Source Table	Friendly Name	Filter Details
✓	DimChannel	DimChannel	
✓	DimDate	DimDate	
☐	DimEntity		
✓	DimProduct	DimProduct	Applied filters
✓	DimProductSubcategory	DimProductSubcategory	Applied filters
✓	DimPromotion	DimPromotion	
✓	FactSales	FactSales	

Applying filtering

13. Click on the **Finish** button and as usual the wizard displays how many rows were transferred. After importing all the data without any error, the user will receive a success message being displayed. The **FactSales** table only has two million rows. Click on **Close**, and the users will be able to see the six tables that were imported from the `ContosoSales` database file.

The user needs to stay on the same PowerPivot workbook in order to import another data source for the same analysis.

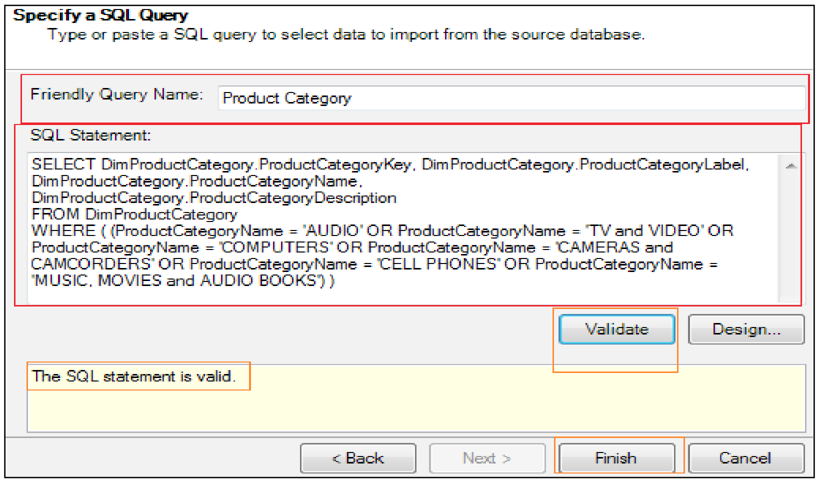
Importing data using a SQL query

Usually, users can use a SQL query to import data into the **PowerPivot** window; although this is applicable for relational databases only. A SQL query would be useful for users as they would be able to create custom columns, some typical mathematical calculations, and some functions in their data. Here in the sample data folder, there is a `SQLQuerytext` file, and in this file there is a SQL query that extracts some data from the `ProductCategories` MS Access file in the sample data folder.

The following steps have to be performed to import data using a SQL query:

1. In the same **PowerPivot** window, click on **From Database** and select **From Access**.
2. In the Table Import Wizard, enter `Product Category` in the **Friendly Query Name** box and click on the **Browse** button. Navigate to the sample data folder, select `ProductCategories`, click on **Open**, and test the connection. If it succeeds, click on the **Next** button.
3. Select **Write a query** that will specify the data to import, which will specify the data to import option, and then click on **Next**.
4. As shown in the following figure, enter `Product Category` in the **Friendly Query Name** box of the Table Import Wizard. It will be displayed as the table name in the **PowerPivot** window.
5. Here, users can write or paste a SQL query to select the data to be imported from the source database. Otherwise, those who are not an expert in SQL can use the **Design** button in order to create a query by GUI.
6. Here, there already is a SQL query, so we will just copy and paste it into the **SQL Statement** box and click on **Validate**. If it's a valid query, the message that will be displayed is **The SQL statement is valid**.

7. Click on the **Finish** button; as usual the wizard displays the number of rows that were transferred. Then click on **Close**.



Specify a SQL Query
Type or paste a SQL query to select data to import from the source database.

Friendly Query Name: Product Category

SQL Statement:

```
SELECT DimProductCategory.ProductCategoryKey, DimProductCategory.ProductCategoryLabel,
DimProductCategory.ProductCategoryName,
DimProductCategory.ProductCategoryDescription
FROM DimProductCategory
WHERE ( (ProductCategoryName = 'AUDIO' OR ProductCategoryName = 'TV and VIDEO' OR
ProductCategoryName = 'COMPUTERS' OR ProductCategoryName = 'CAMERAS and
CAMCORDERS' OR ProductCategoryName = 'CELL PHONES' OR ProductCategoryName =
'MUSIC, MOVIES and AUDIO BOOKS') )
```

The SQL statement is valid.

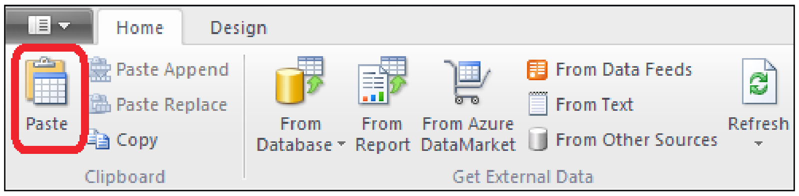
SQL query Table Import Wizard

8. In the **PowerPivot** window, the users will be able to see the new table Product Category, which was imported by using contain SQL query.

Importing data based on copy and paste

Contoso sells its products globally in the given sample data. In the Excel file, there is Geography, which has the location details where Contoso is selling products. By importing this data, the analysis of sales performance based on an organization hierarchy will be possible. Let's try to import this data using the copy-and-paste method.

1. Go to the sample data folder. Double-click on the `Geography` Excel file and locate the **DimGeography** sheet that the location details of Contoso.
2. Select the columns and rows, which has only contain data and copy them. Go to the **PowerPivot** window on the **Home** tab, and click on **Paste**. The **Paste Preview** dialog box displays the new table that will be created.



Copy and paste

3. Enter `Geography` in the **Table Name** text box and verify that **Use first row as column headers** is selected. Click on **OK**. The new table with the name **Geography** is created in the **PowerPivot** window.

There is a need to import one more piece of data in the same PowerPivot window for sales analysis, so follow the next importing process to import another data source.

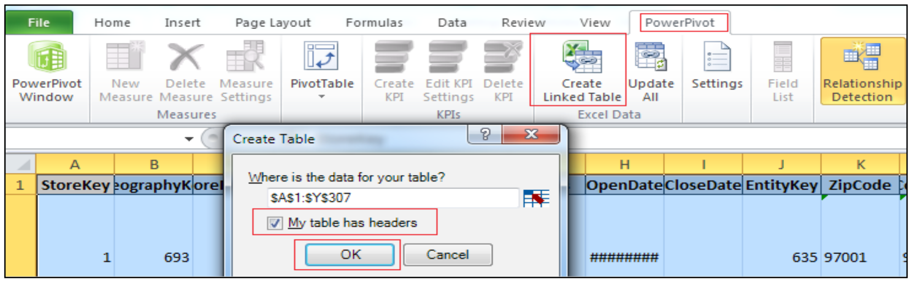
Importing data using an Excel-linked table

Now almost all the data has been imported to do the sales analysis, but an important thing here is that there is a need to import the stored details. In the sample data folder, there is an Excel sheet named `Stores` that has the Contoso store information.

More precisely, a link between the Excel table and the PowerPivot window can be created instead of importing all the data into the PowerPivot Workbook. The advantage of this methodology is that instead of creating and maintaining the data in Excel and importing it from various data sources, users will be able to modify the values in Excel while analyzing the data in PowerPivot.

The following steps have to be performed to import data using an Excel linked table:

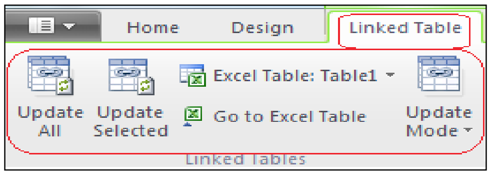
1. Open the `Stores.xlsx` file from the sample data folder and copy the worksheet named `Stores`. Paste it into an already opened Excel workbook that was created to start PowerPivot.
2. After copying the data into the Excel book using paste, change the sheet name, which is named `Sheet1`, `Sheet2`, and `Sheet3` in Excel by default. Right-click on the sheet name where the data was copied and select **Rename** and name the sheet `Stores`.
3. Select the **PowerPivot** tab and click on **Create Linked Table**. It will show a **Create Table** dialog box; select **My table has headers** and click on the **OK** button.



A	B		H	I	J	K
StoreKey	eographyKo	ore	OpenDate	CloseDate	EntityKey	ZipCode
1	693		#########		635	97001

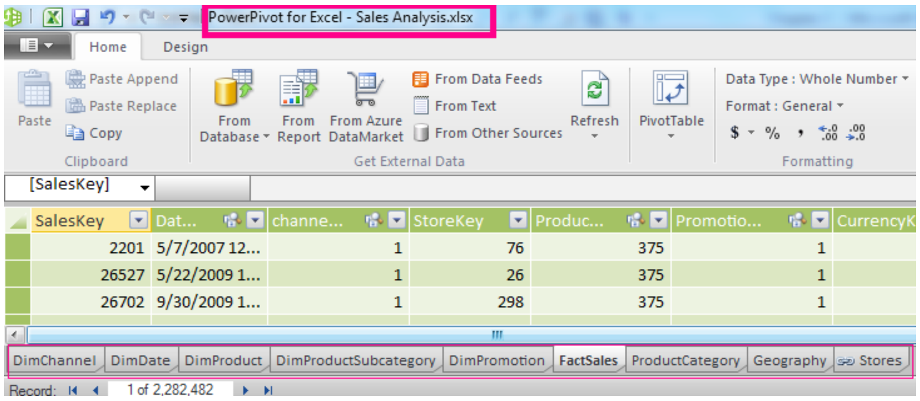
Excel linked table

4. After clicking on the **OK** button, the store data will be imported as a linked table, namely, Table1 that has been renamed as `Stores`, in PowerPivot.
5. Notice that there will be an extra tab in the **PowerPivot** window named **Linked Table**, in which there will be some features. These features will be applicable for the linked table only.



Linked table features

Finally, after successfully importing all the related tables into one PowerPivot window for the sales analysis, the following screenshot displays all the tables of the workbook:



SalesKey	Dat...	channe...	StoreKey	Produc...	Promotio...	CurrencyK
2201	5/7/2007 12...	1	76	375	1	
26527	5/22/2009 1...	1	26	375	1	
26702	9/30/2009 1...	1	298	375	1	

Sales analysis of the PowerPivot Workbook

Data refresh

After importing data into the PowerPivot Workbook, keeping the data up-to-date for analysis is important, as data sources that were added would have some rows or columns that might have been added. Hence, using the features such as data refresh in the PowerPivot Workbook would come in handy in times like these. For instance, if the user is importing sales transaction data, refreshing the data must be done several times per day, in order to know the time interval for this kind of process. More importance has to be given to data refresh. Here, two types of industry data have been imported; they are e-commerce and retail. Both have relation database connection source and they would be all the more important if the users are using relational data source, which most probably intakes the transaction data.

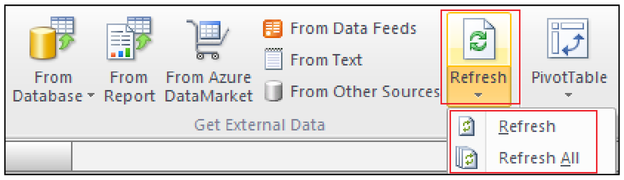
There are two kinds of data refresh processes in PowerPivot:

- Doing data refresh manually in the **PowerPivot** window itself.
- If the reports were published in the SharePoint Server, an automatic schedule for data refresh in SharePoint can be done. More on data refresh in SharePoint is discussed in *Chapter 5*, *Publishing Reports Using the SharePoint Server*.

In this chapter, only the manual refresh process will be shown.

To refresh one or more tables in the PowerPivot Workbook, perform the following steps.

In the **PowerPivot** window, on the **Home** tab, in the **Get External Data** section, there will be a **Refresh** option. To refresh a particular table only, select the table and click on **Refresh**. It will show the data refresh progress status, and if there is a need to refresh all the tables in the PowerPivot Workbook, just click on the **Refresh All** button.



Data refresh

To refresh all the tables that use the same connection, perform the following steps.

In the **PowerPivot** window, on the **Design** tab, click on **Existing Connections** to receive the dialog box. Select the connection that needs to be refreshed and click on **Refresh**; this will show the data refresh progress status.

Refreshing for the copy and paste import method cannot be done since the data through this procedure becomes static. The preceding steps will not be applicable for linked table data refresh, but there is another way to do data refresh. Let us assume the store table of sales analysis has been imported into PowerPivot by using the linked table data import method. If users follow the preceding general steps, they won't find any options to do so. In the *Linked table features* figure, the user would find options such as **Update all** (used in order to update entire table), **Update selected** (used in order to update selected rows and columns of a linked table), and **Update mode** (used in order to set the update mode to either automatic or manual; by default, it is in the automatic mode, and a user can also update manually from the **PowerPivot** window). By using these options, a user can refresh their linked table data in the PowerPivot interface.

[

If the user deletes a table, if a column is deleted or renamed in the source, or if the database is offline and if the user has no linger permissions, an error will occur during data refresh. Therefore, the user is supposed to verify these things if they encounter any problems during the time refresh. And when the user shares the PowerPivot Workbook with other people, the user is supposed to help them in avoiding data refresh errors by reminding them to request for permissions on the data sources that are providing the data.

]

Summary

This chapter shows the users how to prepare data for analysis. We also covered different types of data sources that can be imported into the PowerPivot interface. Some information about the provider and a general overview of the ETL process has also been given. Users now know how the ETL process works in the PowerPivot interface; also, users were shown an introduction and the advantages of DAX.

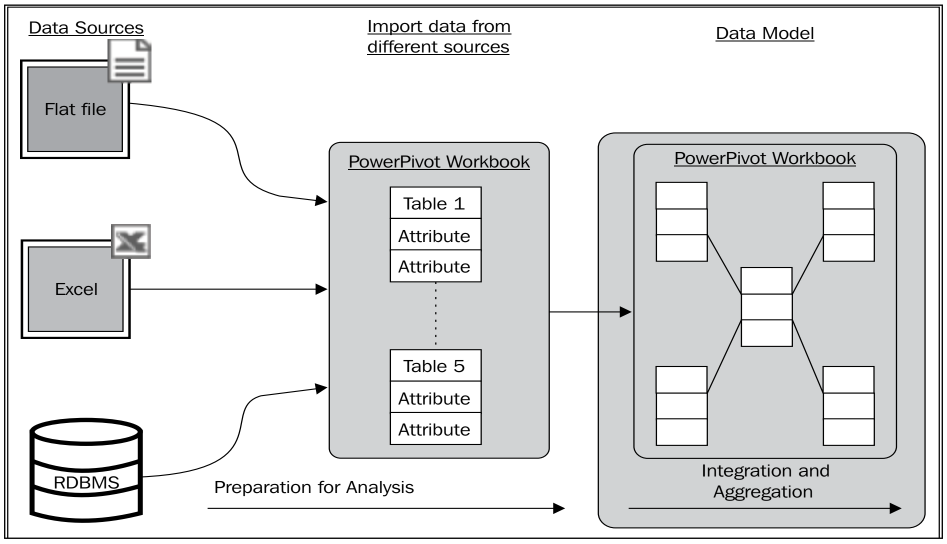
Users have learned how to import different data sources into one PowerPivot workspace. In this chapter, a successful import of two kinds of industry data from different data sources has been shown and explained in detail with each of the functionalities and features of the data source import wizard, connection details, filtering, ignoring unwanted columns, and so on. The user now knows details on data refresh with provided detailed diagrams and screenshots for easy understanding.

In the next chapter, we will cover a brief introduction of the data model and later, a full description of data model creation based on the two types of industry data that were created in this chapter will also be given. A brief description of all kinds of the DAX formula with a practical functionality will also be discussed.

3
Data Model

In business intelligence technology, the prominent logic of the data model design is that if the user designs a good logically designed schema, they can easily analyze a large volume of data. This chapter gives the fundamental concepts of a data model and how to design the logical schema based on the data sources.

A successful importation of two industry-type data for the analysis has been done in the previous chapter. In this chapter, a fundamental idea about the data model design will be given and we will continually learn how to practically design a data model for the imported data with the use of PowerPivot functionalities.



Data model

The preceding diagram illustrates how a data model inside the PowerPivot Workbook will be; the prominent factor is that this chapter is fully based on integration and aggregation from imported data sources.

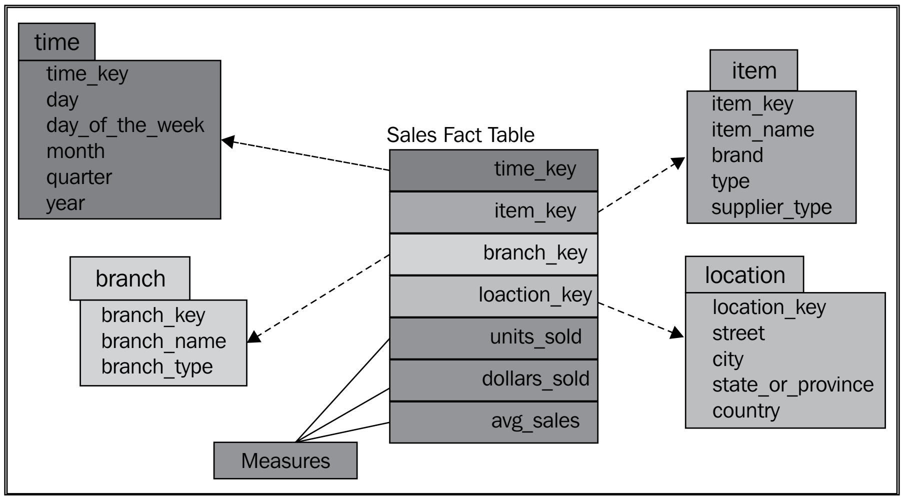
General overview of a data model

It provides decisional information at an organizational level from integrated and aggregated data, and gives users an intuitive and manageable view of information. A Data Model creates a new repository that integrates data from various sources and then makes the data available for analysis and evaluation aimed at decision-making processes.

A data model should generally be used for analysis. For general analysis, a data model should be or is generally expected to be:

- Homogenized
- Integrated
- Prepared
- Detailed and aggregated

The logical design of a data warehouse is based on principles that are different from those used in operational databases as there are issues such as data redundancy and denormalization with relations. Denormalization is the process of integrating multiple small tables into one big table in which there are data redundancy issues, but it is deliberately designed for the data model since it speeds up the query performance. Normalization is the reverse process of denormalization; the user finds the definition and importance of normalization in the data relationship section.



Star schema

The **star schema** is perhaps the simplest data warehouse schema. It is called a star schema because the entity-relationship diagram of this schema resembles a star, with points radiating outwards from a central table. The center of the star consists of a large fact table and the points of the star are the dimension tables. A star schema is characterized by one or more very large fact tables that contain primary information about the data warehouse and a number of much smaller dimension tables. A star query is a join between a fact table and a number of dimension tables. Each dimension table is joined to the fact table using a primary key to a foreign key join, but the dimension tables are not joined to each other.

A typical fact table contains keys and measures. For example, in the sample schema (in the preceding diagram), the `SalesFact` table, contains measures such as `units_sold`, `dollars_sold`, and `avg_sales`, and keys such as `time_key`, `item_key`, `branch_key`, and `location_key`. The dimension tables are `time`, `branch`, `item`, and `location`.

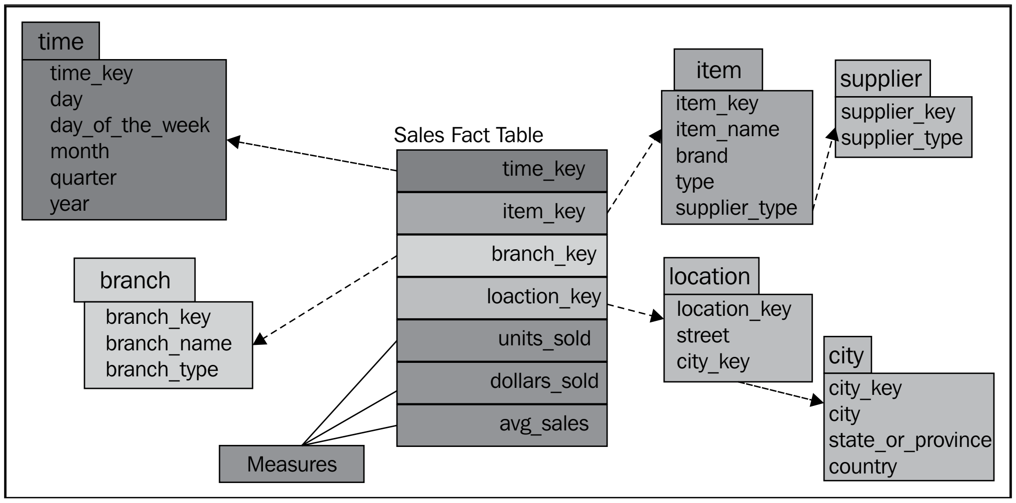
The main advantages of a star schema are as follows::

- They provide a direct and intuitive mapping between the business entities being analyzed by the end users and the schema design
- They provide highly optimized performance for typical star queries
- They are widely supported by a large number of business intelligence tools, which may anticipate or even require that the data warehouse schema contains dimension tables

Some of the characteristics of the star schema is given in the following section, specifying the features of different tables.

- Features of the dimension table are as follows:
    - It contains attributes describing all the dimension levels
    - It is in denormalized form
    - It contains embedded hierarchies
    - It contains a key which acts as an identifier of the lowest hierarchy level
- Features of the fact table are as follows:
    - It contains measures and dimension keys, where keys indicate a concatenation of all the dimension keys

The **Snowflake schema** is a more complex data warehouse model than a star schema, and yet it is a type of star schema since the dimensional tables are normalized. It is called a snowflake schema because it resembles a snowflake as shown in the following diagram:



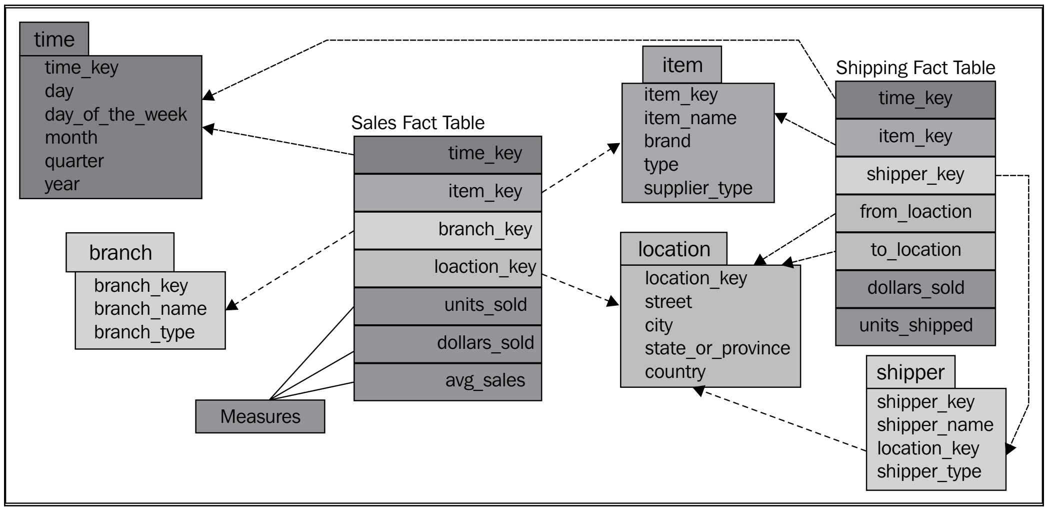
The Snowflake schema

Characteristics of a snowflake schema are as follows:

- The dimension hierarchies appear implicitly
- It can be constructed from a star schema by applying normalization algorithms to dimension tables

A snowflake schema normalizes dimensions to eliminate redundancy, that is, the dimension data is grouped into multiple tables instead of one large table. For example, a location dimension table in a star schema may be normalized into a location table and a city table in a snowflake schema. While this saves space, it increases the number of dimension tables and requires more foreign key joins. This results in more complex queries and reduced query performance. The preceding diagram presents a graphical representation of a **Snowflake schema**.

The last type of schema is the **Constellation schema**, which is a generalization of the star and snowflake schemas. The structure of this schema would be in multiple fact tables and shared dimension/level tables. This schema is used mainly for aggregate fact tables or when the user wants to split a fact table for better comprehension. A fact table is split only when the user wants to focus on the aggregation of a few facts and dimensions.



The Constellation schema

A data model in PowerPivot

Preparation of data for analysis has been done, but for full and meaningful analysis relationships have to be created between the tables that were imported from different data sources. By creating a relationship, integration of columns, creation of attribute formats, metrics, and calculation of columns using DAX for our analysis and advanced reporting can be done as per the requirements. One of the prominent things here is that a hierarchy can be created based on these attributes. So finally the logical design of the PowerPivot interface will be based on the star schema, the snowflake schema, or the constellation schema. After all these processes in the PowerPivot interface are completed, it will fulfill the third principle of integration and aggregation. The **extract, transform, and load (ETL)** process and DAX functional feature have already been discussed and this will help a lot in designing the data model.

The importation of the data of two industries for the analysis has been done. Now, we will proceed with the data model design for sales analysis first and then design a data model for e-commerce customer activities.

Sales analysis data model design process

In the Sales Analysis PowerPivot Workbook, importation of nine tables from different data sources has been done. Here, some tables may have no relationship and some may have different data formats, some tables may need a few calculated columns, and so on. Solving these issues will make the data homogenized, integrated, prepared, and detailed and aggregated; this is known as a logical design.

The following are the tables that were imported into the Sales Analysis PowerPivot Workbook:

- `DimChannel`
- `DimDate`
- `DimProduct`
- `DimProductSubcategory`
- `DimPromotion`
- `FactSales`
- `ProductCategory`
- `Stores`
- `Geography`

Creating a data relationship for the sales analysis data

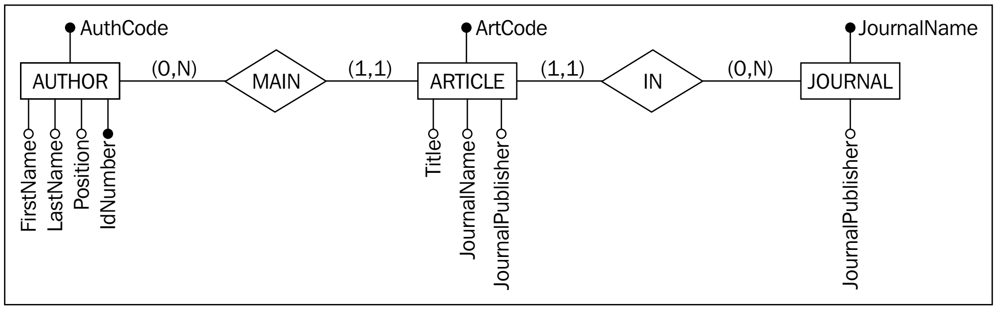
Before creating a relationship the users must know about data relationships, what is the use of it, and the classification of relationships. Usually the relationship is derived from a normalized database. Normalization is the process of breaking a big table into two or more smaller tables, so to get the information from the tables, a connection has to be made between the tables, which is correlated to the requirement because with normalization, users can avoid the insert, update, delete, and redundancy anomalies that may occur. But if there is a big table, it will have a redundant value when the users do the insert, delete, and update operations. Also, it may have some problems; in order to avoid these kinds of problems, split the table.

A relational database has a concept called integrity constraint, and using this we can assign primary or foreign keys to a table. A relationship is based on the foreign key, which refers to either a primary or unique key column of another table.

The following are the three types of relationships:

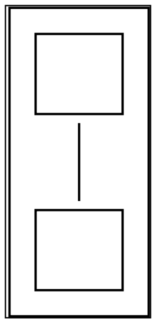
- One-to-one relationship
- One-to-many or many-to-one relationship
- Many-to-many relationship

Let's consider the `RESEARH_DB` database where there are three tables, namely, `Author`, `Article`, and `Journal` and the bullet attributes are keys of tables.



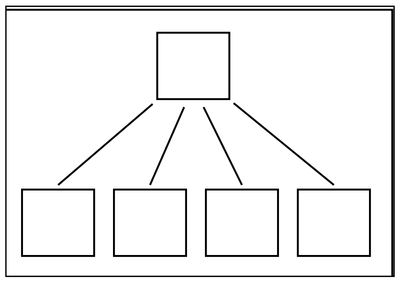
Entity relationship

A record in the first table relates to a record in the second table; this is known as a one-to-one relationship. Let us consider the preceding diagram where the **ARTICLE** (**Artcode**) has one **AUTHOR** (**Authcode**) and one **JOURNAL** (**JournalName**). These are one-to-one relationships. Inside the brackets is a key attribute of the relation tables based on the attribute that can help in creating a relationship between the tables. One-to-one relationships can be illustrated as follows:



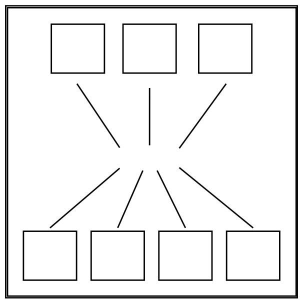
One-to-one

A record in one table relates to multiple records in the second table, but the records in the second table relate to only one record in the first table; this is called a one-to-many relationship. One **AUTHOR** (**Authcode**) has many **ARTICLES** (**Artcode**) but the **ARTICLE** (**Artcode**) relates only to one record in the **AUTHOR** (**Authcode**) table; one **JOURNAL** (**JournalName**) has many **ARTICLES** (**Artcode**) but the **ARTICLE** (**Artcode**) relates only to one record in the **JOURNAL** (**JournalName**) table.



One-to-many

Multiple records in one table relate to multiple records in a second table and the records in the second table relate to many records in the first table; this is called a many-to-many relationship. In the present RESEARCH_DB example, there is no many-to-many relationship. However, let's consider the `Product`,and `Customer` tables, a table with customers who can purchase many different products and a table with products that can be purchased by many different customers.



Many-to-many

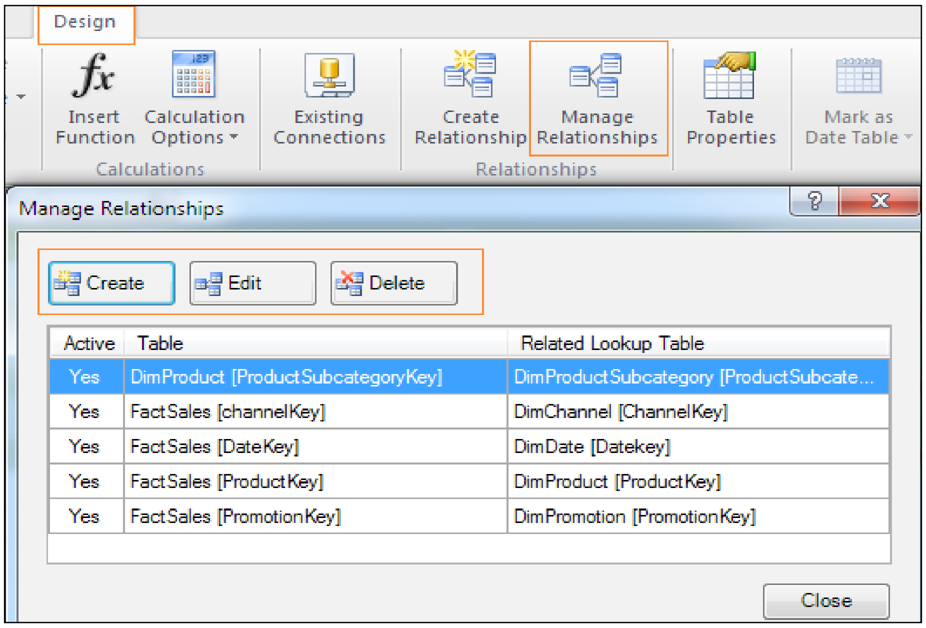
The vital information is that the PowerPivot interface will support only a one-to-one and one-to-many relationship; it won't support a many-to-many relationship. But there is one option in order to create a many-to-many relationship, that is, using the DAX function. Another prominent note is that it is not possible to create more than one relationship between two tables.

Let's create a relationship in the Sales Analysis PowerPivot Workbook.

Importation of data has been done using three different sources for the sales analysis. If the users are importing data from the relational database, they can see the existing relationship of those tables in the PowerPivot interface. In our case, importing data from the six tables was done from among the nine tables in the MS Access database.

In order to see the existing relationship, perform the following steps:

1. Open the `Sales Analysis` Microsoft Excel file from where it was stored in the local drive. Once opened, the Excel file will show the **Stores** datasheet that was imported by using an Excel-linked table.
2. Select the **PowerPivot** tab from the ribbon and click on the PowerPivot window. Now, the users will be able see the tables that were imported for the sales analysis process.
3. On the **Design** tab in the **Relationship** section, click on **Manage Relationships**.
4. The users will be able see the **Manage Relationships** dialog box. Here, the relationship between the tables that were imported from the MS Access database will be shown.

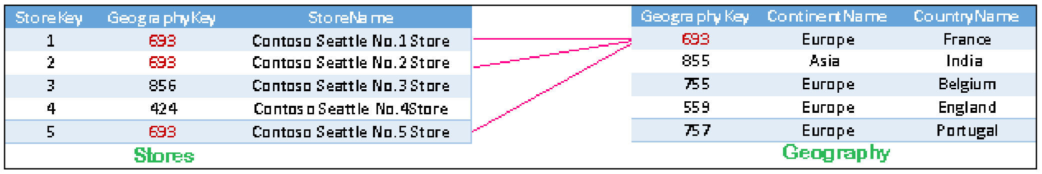


Active	Table	Related Lookup Table
Yes	DimProduct [ProductSubcategoryKey]	DimProductSubcategory [ProductSubcate...
Yes	FactSales [channelKey]	DimChannel [ChannelKey]
Yes	FactSales [DateKey]	DimDate [Datekey]
Yes	FactSales [ProductKey]	DimProduct [ProductKey]
Yes	FactSales [PromotionKey]	DimPromotion [PromotionKey]

The Manage Relationships dialog box

5. In the dialog box there are three columns, namely, **Active**, **Table**, and **Related Lookup Table**. From the **Active** column, users will be able to know the status of the relationship, from the **Table** column they will be able to see the foreign key relationship table and the foreign key attribute name inside the bracket, and in the **Related Lookup table** column they will be able to see the related table and either a primary key or a unique attribute name will be displayed inside the bracket.
6. Another important thing is that there are three buttons, namely, **Create**, **Edit**, and **Delete**. By clicking on the **Create** button, users will be able to create a new relationship with the existing table in the PowerPivot Workbook, by clicking on the **Edit** button they will be able to change the relationship from the existing relationship based on their requirement and also there is a checkbox named **Active**, which unchecked can deactivate the existing relationship. By unchecking it, the users will be able to see a **No** in the **Active** column. And by clicking on the **Delete** button; the users will be able to delete the existing relationship.

Currently there are three tables, namely, `ProductCategory`, `Stores`, and `Geography`, but without relationships. Hence, in order to do a legitimate analysis, all the tables must be related with other tables. A relationship between the `Stores` and `Geography` tables will be made since in the `Stores` table we only have `Geography Key` to analyze the stores' transaction location wise. A relationship has to be made with the `Geography` table because this table consists of geographical information and also there is a column named `Geography Key` which is supposed to be the primary key of this table and in the `Stores` table the `Geography Key` is supposed to be the foreign key. This will either be called one-to-many or many-to-one.



StoreKey	GeographyKey	StoreName
1	693	Contoso Seattle No.1 Store
2	693	Contoso Seattle No.2 Store
3	856	Contoso Seattle No.3 Store
4	424	Contoso Seattle No.4Store
5	693	Contoso Seattle No.5 Store

Stores

GeographyKey	ContinentName	CountryName
693	Europe	France
855	Asia	India
755	Europe	Belgium
559	Europe	England
757	Europe	Portugal

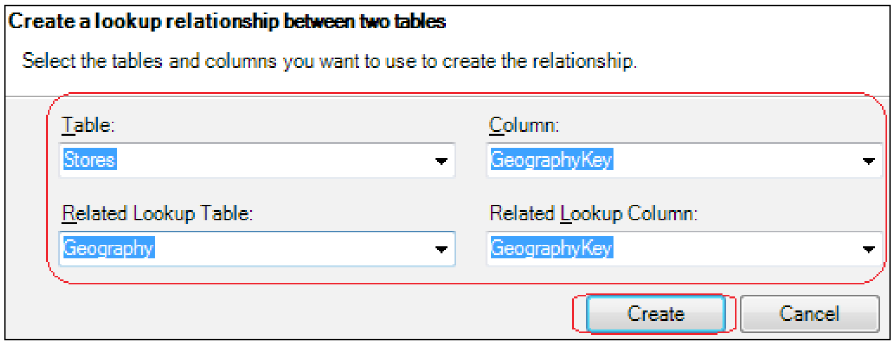
Geography

Relationship between the Stores and Geography tables

In order to create a relationship between the `Stores` and `Geography` tables perform the following steps (there are three ways for creating relationships in the PowerPivot interface, here the first way is being shown):

1. On the **Design** tab, in the **Relationship** section, click on **Manage Relationships**.

2. Click on the **Create** button, the users will see a Create relationship dialog box. In this dialog box, users have to select the tables and the columns with which they want to create a relationship. In the **Table** combobox select the table with a foreign key relationship, and here select the **Stores** table. In the **Column** box select **GeographyKey**. In the related table area, **Geography** is selected for the **Related Lookup Table** field and select **GeographyKey**, which is the primary key to the table selected in the **Related Lookup Column** box.



Create a lookup relationship between two tables

Select the tables and columns you want to use to create the relationship.

Table: Stores

Column: GeographyKey

Related Lookup Table: Geography

Related Lookup Column: GeographyKey

Create relationship between Stores and Geography

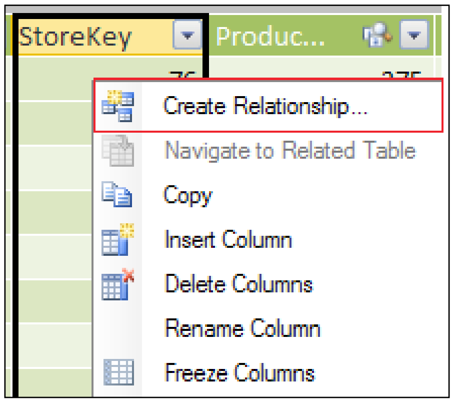
3. Click on the **Create** button. This will create a new relationship between the two tables that were given and the users will see the **Manage Relationship** dialog box where they will be able to see the new relationship.
4. Click on **Close**.

[Please keep in mind that upon creating the relationship, users must use either the primary or unique key attributes for **Related Lookup Column,** otherwise the users won't be able to create the relationship and it will show an error message upon trial.]

Now, let's create a relationship between the `FactSales` and `Stores` tables in order to analyze the sales transaction based on the store and location. Users will have to create a relationship between both the tables. The `Geography` table is already related to the `Stores` table. So, if the user can create a relationship between them it will be adequate for this kind of analysis.

Using the second way, create a relationship between the `FactSales` and `Stores` tables as given in the following steps:

1. In the PowerPivot Workbook, select the `FactSales` table.
2. Select the **SalesKey** column header; right-click on it and select the **Create Relationship...** option.



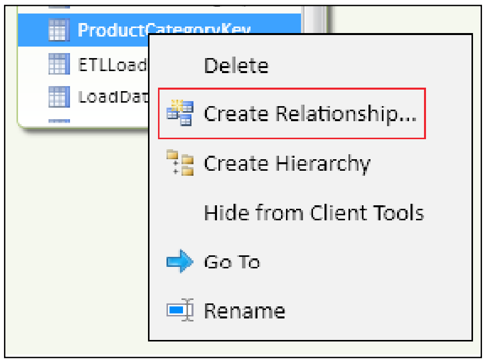
Create Relationship between FactSales and Stores

3. The user will see the create relationship dialog box in which there are tables and columns. The user will be able to see **FactSales** and **StoreKey**, which are foreign key attributes. But since **Stores** was selected in the **Related Lookup Table** combobox it will automatically detect **StoreKey**, which is the primary key attribute in the **Related Lookup Column** box.
4. Click on the **Create** button. This will create a new relationship between the two tables that were given.

In the Sales Analysis workbook there is still one more table, namely, `ProductCategory` without a relationship. Here, a relationship with the `DimProductSubcategory` table is going to be made, which already has a relationship with the `DimProduct` table and is also related to the `FactSales` table. By using these kind of relationships, users will be able to analyze the sales transaction based on the product and product category.

In order to create a relationship between the `DimProductSubcategory` and `ProductCategory` tables, perform the following steps. For this relationship the third way, which is GUI-based, will be used:

1. On the **Home** tab, there is a **View** section in the right-hand corner where users can see **Data View** and **Diagram View**. Right now, users are in the data view. So in order to create a relationship by using the third way, click on the **Diagram View** option. After clicking on it users will be able to see all the tables, which have a relationship with other tables, in a diagram format, excluding the `ProductCategory` table since we haven't created a relationship for it yet.
2. Select the **ProductCategoryKey** attribute from the `DimProductSubcategory` table by right-clicking on it and select the **Create Relationship…** option.



Create a relationship between DimProductSubcategory and ProductCategory

3. As usual, the user will see the **Create Relationship** dialog box, which displays the table and column boxes where the users can see **DimProductSubcategory** in which **ProductCategoryKey** is the foreign key attribute. But here, **ProductCategory** was selected in **Related Lookup Table** and **ProductCategory** automatically detects **ProductCategoryKey**, which is the primary key attribute in the **Related Lookup Column** box.
4. Click on the **Create** button. This will create a new relationship between the two given tables.

Finally, relationships with tables have been successfully made for the Sales Analysis workbook. By using the **Diagram View** option users will be able to see the relationship table.

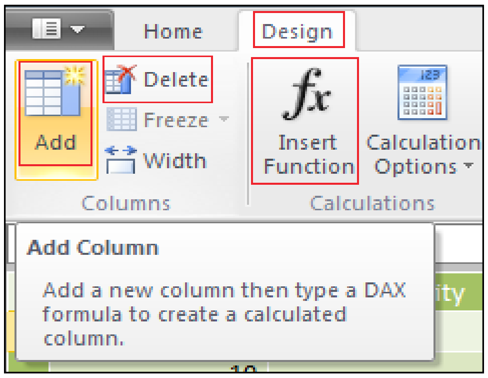
Adding a new column using DAX for sales analysis data

Here we will add a new column, that is, a calculated column that is going to be added to help us analyze meaningfully. This column will be based on the data that already exists in the table. The DAX expression is going to be used for this data.
As we discussed in the previous chapter, DAX will be helpful for ETL processes such as changing the data format and calculation. DAX also plays a vital role in the second principal data model.

Let's consider the `FactSales` table, which consists of sales transaction details and probably has many tables related to it. However, in this table there is no `Profit` attribute, so if the users want to do the analysis and create reports based on profits based on the product , store, location, and so on, they won't be able to do so as for that, they have to create a new column, namely, `ProfitAmount`, in which they can get data about the profit amount.

In order to create `ProfitAmount`, perform the following steps:

1. If from **Diagram View**, you can switch back to **Data View**, select the **FactSales** table and click on the **Design** tab. In the **Columns** section there is an **Add** option; clicking on it will move the cursor to the formula bar.
2. By using the **Insert Function** option, select the category of logic for the DAX function as listed in *Chapter 2*, *Preparation Analysis of Data Source*, and then select the function. This is the basic level of writing for the DAX function; it will be time consuming for people who are not familiar with Excel formulas as DAX is more or less similar to Excel.

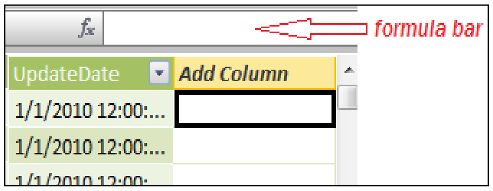


Add Column

Add a new column then type a DAX formula to create a calculated column.

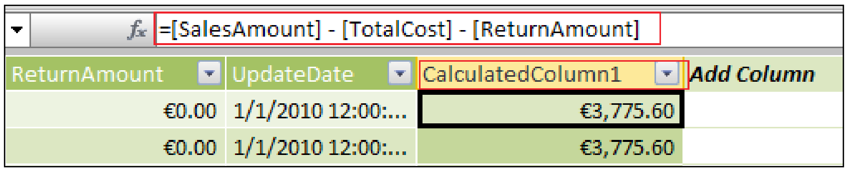
DAX functions in PowerPivot

3. When the importation of the table into PowerPivot Workbook, there will be an **Add Column** in the last column of the table that is used to create the calculated column. So, just click on the cursor and write the formula in the formula bar as given in the following screenshot:



The DAX formula bar

4. Now, write the formula in order to find out the profit amount. Our `FactSales` table has `SalesAmount`, `TotalCost`, and `ReturnAmount`. The amount that we will get after subtracting all the attributes would be the profit amount since `SalesAmount` is the sold amount, `TotalCost` is the original price, and `ReturnAmount` is for any returns.
5. As the logic for finding the profit amount is clear, this has to be applied in the formula bar. In order to apply it, the second step is being used; the formula is: `=[SalesAmount] - [TotalCost] - [ReturnAmount]`. Keep in mind that `'['`,`']'` has to be used to mention the attribute name. When `'['` is written, it automatically shows all the attributes in that table with `']'`; the users just have to select the attribute instead of writing it on their own.
6. In the formula bar, paste the formula and click on **Enter**; the new column, namely, `CalculatedColumn1` will be formed. Here, the users will be able to view the profit amount.



=[SalesAmount] - [TotalCost] - [ReturnAmount]

ReturnAmount	UpdateDate	CalculatedColumn1	Add Column
€0.00	1/1/2010 12:00:...	€3,775.60	
€0.00	1/1/2010 12:00:...	€3,775.60	

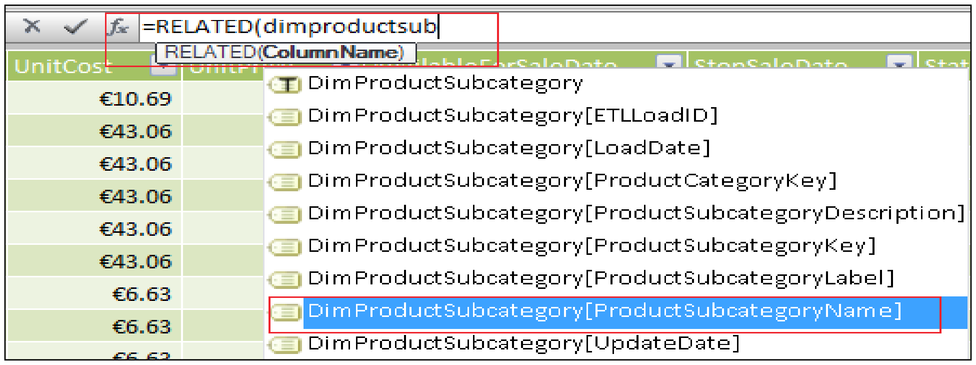
The ProfitAmount formula in the formula bar

7. Double-click on the **CalculatedColumn1** column and rename it as `ProfitAmount`. It is important to select **Add Column** in order to create the calculated column. After applying the formula users will see `CalculatedColumn1` with the values calculated from the formula. The users will also see **Add Column** which is next in the sequence.

Next, a calculated column is being created into `DimProduct` from related tables such as `DimProductSubcategory` and `ProductCategory`. Let us consider `DimProduct` that has product information, cost, and so on. But if we need to create a hierarchy based on products, all the attributes in the same tables are needed. But in this case, all the required attributes are present but not in the same table; it has related tables, so by using the RELATED DAX function (to get more information about RELATED go to: `http://technet.microsoft.com/en-us/library/ee634202.aspx)`; users will be able to get product category and product subcategory data into the `DimProduct` table. The reason why we are using it will be explained in the following topic on hierarchy.

In order to get the product subcategory and product category data into the `DimProduct` table, perform the following steps:

1. In the **PowerPivot** Workbook, select the `DimProduct` table and in the **Design** tab select the **Add from Columns** section.
2. The cursor will be in the formula bar; the right formula to get product subcategory into this table is: `=RELATED(DimProductSubcategory[ProductSubcategoryName])`
3. The way to type the formula in the formula bar is type `"=RELATED("` and then selecting the desired table, press the *Tab* key and select the desired lookup column and type `')'` to conclude the statement.



```
=RELATED(dimproductsub
RELATED(ColumnName)
```

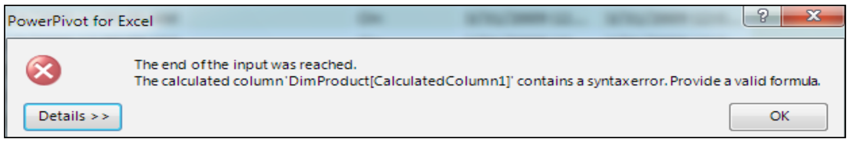
UnitCost
€10.69
€43.06
€43.06
€43.06
€43.06
€43.06
€6.63
€6.63

```
DimProductSubcategory
DimProductSubcategory[ETLLoadID]
DimProductSubcategory[LoadDate]
DimProductSubcategory[ProductCategoryKey]
DimProductSubcategory[ProductSubcategoryDescription]
DimProductSubcategory[ProductSubcategoryKey]
DimProductSubcategory[ProductSubcategoryLabel]
DimProductSubcategory[ProductSubcategoryName]
DimProductSubcategory[UpdateDate]
```

DAX formula for the RELATED function

4. Once the formula is written, press the *Enter* key or tick the option beside **fx** to execute the formula and it will populate `DimProductSubcategory`, along with the `CalculatedColumn1`. Later, rename it as `ProductSubcategory`. Follow the same steps as earlier in order to get product category data. The formula is `=RELATED (ProductCategory [ProductCategoryName])`. Once populated, rename the column as `ProductCategory`.

5. If any of the parentheses in the formula bar are missed, users will receive an error message.


PowerPivot for Excel

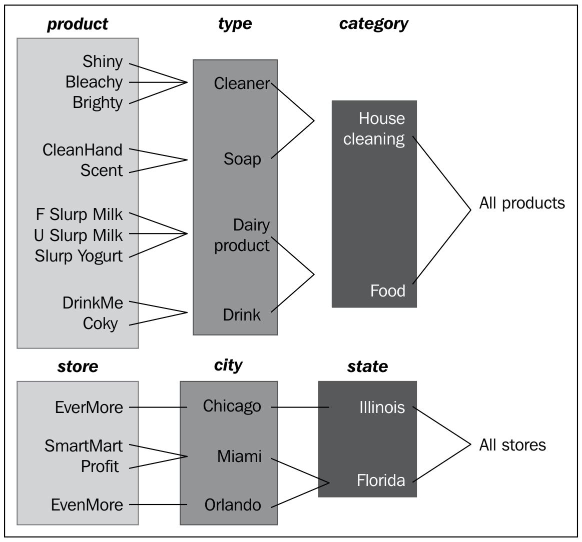
The end of the input was reached.
The calculated column 'DimProduct[CalculatedColumn1]' contains a syntax error. Provide a valid formula.

Error message

Finally, two extra columns in the `DimProduct` table have been added. The reason for adding these two extra columns will be shown in the next topic. Here, a few DAX functions have been used for the sales analysis to achieve the results.

Creating hierarchies for sales analysis data

Ideally, hierarchies are supposed to have a one-to-many relationship and it's vital for creating a data model in the PowerPivot interface. Hierarchies will be useful for end users since the table has more columns, so if an end user wants to do an analysis and make a report they will face a few difficulties. If users have created hierarchies (it consists of multiple columns), they can drag-and-drop the hierarchies into the reports. More precisely, hierarchies will be useful for Pivot tables while making the Pivot tables report, so that users will be able to fully understand the benefits of hierarchies.



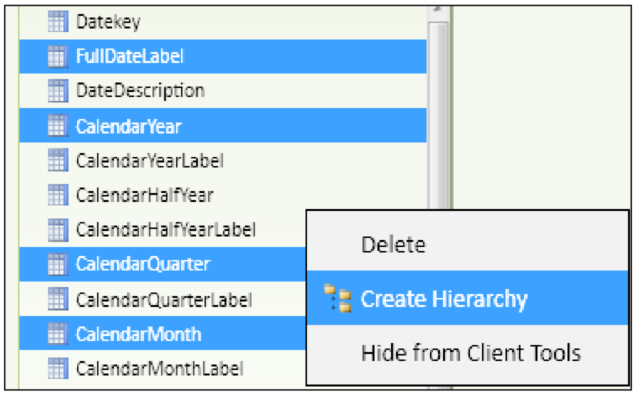
An example of hierarchies

For instance, in the `DimProduct` table hierarchies such as ProductCategory→ ProductSubcategory→ProductName can be created. The users have to know the one-to-many relationship of hierarchies. Each product category has many product subcategories and each product subcategory has many products.

Here, we will create two kinds of hierarchies for our sales analysis data, namely, date- and product-based hierarchies.

In order to create date-based hierarchies, perform the following steps:

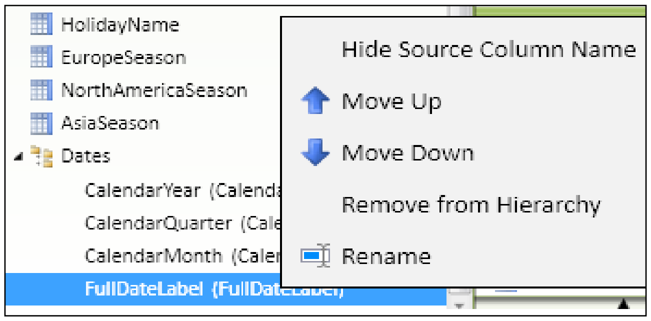
1. In the **PowerPivot** window, switch to the **Diagram** view and expand the `DimDate` table in order to see all the attributes of the table; it should have more than 30 attributes. If users analyze the sales transaction date-wise it will create confusion.
2. Press and hold *Ctrl* and select **CalendarYear**, **CalendarQuarter**, **CalendarMonth**, and **FullDateLabel** in the same order and right-click on the selected columns; then click on the **Create Hierarchy** button. The important thing here is to select from the parent node, **CalendarYear**. The child node for a parent node can be a parent node for a different child node and vice versa.



Create hierarchy in the DimDate table

3. Now the users will be able to see the created hierarchy at the bottom of the `DimDate` table namely **Hierarchy 1**; rename it as `Dates`.

4. Once created, users can do the customizations such as edit, delete, and so on. Here `FullDateLabel` is a child node of `CalendarMonth` but the name of the attribute does not seem to be related to the **Dates** hierarchy so it is going to be renamed as `CalendarDate`. By right-clicking on **FullDateLabel** users will see the right-click options. From here, they can rename and carry out another such customizations. Here **Rename** was selected and its name was changed to `CalendarDate`. Here, inside the bracket, users will be able to see the original attribute name.



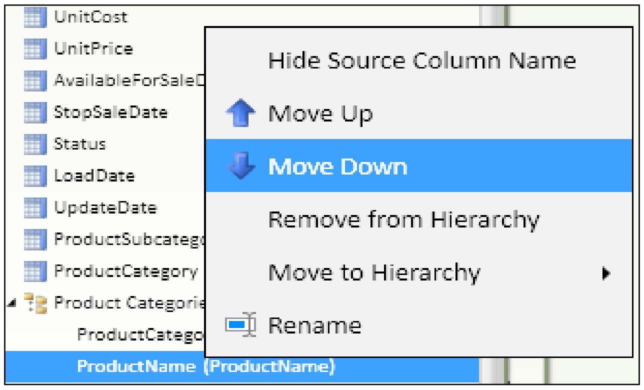
The edit hierarchy in the DimDate table

In order to create product-based hierarchies, perform the following steps. Here the creation of a hierarchy in the PowerPivot interface will be done using another way:

1. In the **Diagram** view, expand the `DimProduct` table, right-click on the header of the table, and click on the **Create Hierarchy** button. An empty hierarchy parent node, namely, **Hierarchy 1** appears at the bottom of the table; rename it to `Product Categories`.

2. Drag the attribute from a parent to child as ProductCategory→→ ProductSubcategory→→ProductName, under the Product Categories hierarchy model.

3. We dragged the **ProductName** attribute under **ProductCategory**, but this is the child node of **ProductSubcategory**. So users will have to select the **ProductName** attribute and right-click on it. In the properties window click on the **Move Down** option; now it will be under **ProductSubcategory**.

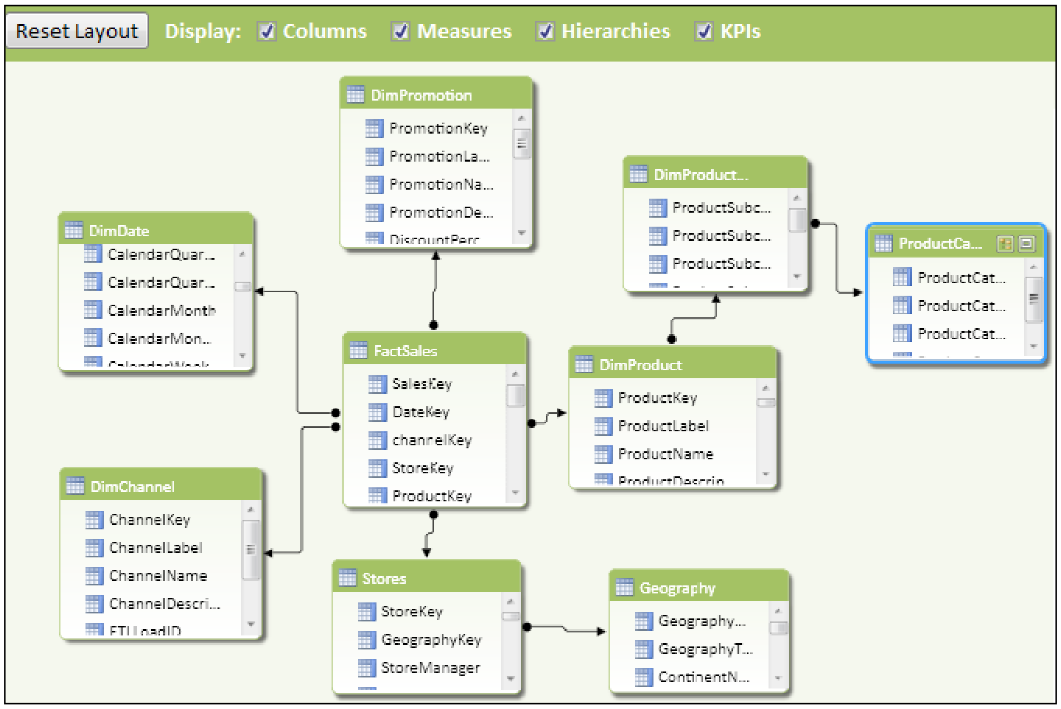


Move down either the parent node or the child node hierarchy

Here two hierarchy models have been created for our sales analysis and generally, users can make, hierarchy for a location as Continent→→ Country→→ State→ → City in the `Geography` table. But in that table there is one attribute, namely, `GeographyType` in which the categories are `City`, `Continent`, `Country/Region`, and `State/Province`. Hence if the users filter only `City`, they can get the location hierarchy data; otherwise they will get data based on the `GeographyType` attribute. For instance, if users want to filter data country-wise or region-wise, they will only get continent and country data, they won't get state and city data. In this case, the hierarchy won't be useful and it will provide a wrong analysis and reports. Hence, the creation of a location-wise hierarchy has been ignored here.

Viewing the logical design data model of sales analysis

Finally, designing the data model for sales analysis in the PowerPivot Workbook has been completed successfully. In order to view the logical design, select the **Diagram** view.



Data model of sales analysis

As discussed with regard to a general overview of the data model, in the beginning of the chapter, there are three kinds of logical schema. In the sales analysis data model, the snowflake schema has been used because there is only one Fact table, namely, `FactSales` that is related to other tables that are dimension tables, and the most important thing is that dimension tables are normalized, for instance `Stores` is normalized as a `Geography` table. Hence, our data model is a snowflake schema.

Attributes of dimension tables are also called descriptive attributes. Descriptive attributes store information from the Fact table and have measures that cannot be used for aggregation. Fact tables have foreign keys for dimension tables and have measures of dimension attributes. If the table has only foreign keys without measures it is known as an empty Fact table.

E-commerce data model design process

Importation of three tables, namely, `User`, `Invitation`, and `Departments` has been done. The `Departments` table is not related to any tables since they were imported as a flat file, but there is a need to create a relationship and to make a good logical design in order to make reports and analysis.

The most vital thing is that the `User` and `Invitation` tables were imported from a relational database and precisely it contains transaction data. This means the tables are created based on the index and optimization and stored as physical models as the `Gender` attribute has values `1` and `2`, where `1` denotes male and `2` denotes female. But the users must make everything into a logical model, since the end user will be able to understand easily, otherwise if we provide inputs such as 1 and 2, they won't understand; also, it's not the way reporting should be done. When we provide the report to the end user, they must easily understand all the data in it just by glancing at the report; for that, users need to design a logical model from the physical model and ensure that the logical model is only readable and not writable. Hence, users can change whatever they want to change only in a structural way in the PowerPivot interface.

The `User` table here is full of physical model data and it has many unwanted attributes; this means that the data is unstructured for reporting and analysis. Hence, there is a need to make a structured data model by using PowerPivot features and DAX expressions.

Earlier when the `User` table was imported, the `Password` attribute was ignored but here some more attributes are going to be taken out and the `User` table will also be split into three tables for the logical model. The calculated column will also be created using the DAX expression. In this scenario, two tables, namely, `Invitation` and `Departments`, which have only necessary attributes for the relationship, will also be renamed. So a split will be created and a calculation will be made but the original data will be kept untouched; we will only create the relationship. Users will be familiarized to splitting and relationships by the end of this chapter.

If the exact tables and attributes were imported, creating a relationship and calculated columns for our data model, just as in the sales analysis data, are the only things left to be done. But here, the tables were deliberately imported differently for the e-commerce customer activities analysis. In the upcoming topics, rearrangement of attributes and split tables will be done since all these things are allowed in the PowerPivot interface even if the user has imported unwanted attributes or tables.

When a user imports data, it's better to design the data model in white paper and import the table and attributes into the PowerPivot Workbook. Another vital thing is that all the attributes must be utilized by the user to their own advantage.

Splitting the User table for e-commerce data analysis

The `User` table has a key attributes, descriptive attributes, and measures in only one table. Hence, splitting of the tables will be done into three, namely, `FactUser`, which has only the key attribute and measures, the `DimUser` table, which has only customer-related attributes such as `name`, `birth date`, `zip code`, and so on, and `DimDate`, which has the user's `CreationDate` and `LastConnectionDate` of the user login. By using these attributes it will be easy to find `Year`, `Quarter`, and `Month` using DAX for the monthly analysis and to check how many customers are logged in on a quarterly basis and vice versa.

In order to create `DimUser`, perform the following steps:

1. There is already the `User` table that has a relationship with the `Invitation` table in the e-commerce PowerPivot Workbook. By using the `User` table we are going to make a `DimUser` tableand look at relationship between user and invitation table (in *Chapter 2*, *Preparation Analysis of Data Source*).
2. Now only these attributes: `UserId`, `FirstName`, `LastName`, `Email`, `BirthDate`, `Gender`, `Type`, `FacebookId`, `FacebookEmail`, and `FacebookDisplayName` are to be kept in the table.
3. Delete the remaining attributes. In order to delete the attributes in the **PowerPivot** Workbook, select the `User` table, and select the column headers that are not needed for the analysis by right-clicking on them. Here, click on the **Delete Columns** button and it will show **Are you sure want to Permanently delete the selected columns?**; click on the **OK** button to delete the attributes that are not required.
4. Now the user will have necessary attributes for the `DimUser` table and then rename it as `DimUser` from `User`.

Now create the `FactUser` table, which will have only the key attribute for the dimension table and measure, by performing the following steps:

1. In order to create `FactUser`, import the necessary attributes from the `User` table in the SQL Server database.
2. Do the importing process for our e-commerce analysis.

3. In order to create the `FactUser` table, import only the `User` table from the database and select only the key attribute and measures from our `User` table. The key values are `UserId` and `Zip`; the `UserId` value is used to create a relationship with `DimUser` and `DimDate` with the latter having only the date when the user created it and the last connection date, so this is also based on the `UserId` relationship. When a relationship with this table is made it will be fully understood that `Zip` is used to create a relationship with the `Department` table. When the data is imported based on this relationship, the user can analyze it based on `Geography` and the measure is `UsedInvitationCount` which is nothing but the data about the customer, along with the number of people invited by the customer.
4. Importation of three attributes, namely, `UserId`, `Zip`, and `UsedInvitationCount` has been done by using the SQLServer, which is explained previously and the table was named as `FactUser`.

So finally we have to create `DimDate` in order to make a good data model for our e-commerce customer data activities, by performing the following steps:

1. Follow the same steps that were taken for creating the `FactUser` table to create `DimDate` since here importation of data from the `User` table will only be done from the SQL Server.
2. In this table, use only three attributes from the `User` table, namely, `UserId`, `CreationDate`, and `LastConnectionDate`. The `UserId` attribute is used to create a relationship with the `FactUser` table.
3. Name the table as `DimDate`.
4. In the **PowerPivot** window, on the **Design** tab, click on **Existing Connections** to see a dialog box. It will display three SQL Server data connections, since now two more tables have been added for this analysis. If the **Friendly** connection name was given in **Table Import Wizard for SQL Server**, a database connection will be shown. Otherwise, it will automatically create SQL Server, SQL Server 1, SQL Server 2, and so on, just as in our e-commerce customer data. But by clicking on **Existing Connections**, it was edited to the **Friendly** connection name and later the table names are given according to the understanding of the user. Here, the tables can also be refreshed.

Creating relationships

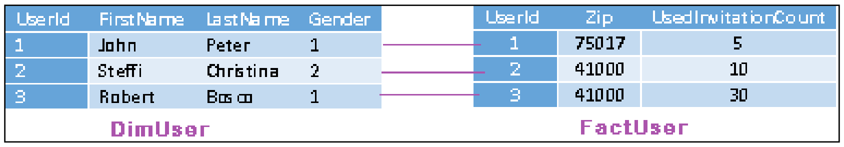
In the Ecommerce PowerPivot Workbook, there are five tables and only two tables have relationships, namely, `Invitation` and `DimUser`; the remaining do not have relationships with other tables. Hence, relationships are going to be created for these tables.

The following are the tables that were imported in the Ecommerce PowerPivot Workbook:

- `Invitation`
- `DimUser`
- `Departments`
- `FactUser`
- `DimDate`

First, create a relationship between `FactUser` and `DimUser` by performing the following steps:

1. In the **PowerPivot** window of the Ecommerce workbook, switch to the **Diagram** view. Now, the user will be able to see all the tables.
2. Select the `UserId` attribute from the `FactUser` table, right-click on it, and select **Create Relationship**.
3. The user will see the **Create Relationship** dialog box in which the table and column boxes will be filled by `FactUser` where `UserId` is the key attribute, but users must select **DimUser** in the **Related Lookup Table** field as it automatically detects the **UserId** attribute in the **Related Lookup Column** box.
4. Click on the **Create** button. This will create a new relationship between the two tables. Now, the user will be able to see the relationship between the two tables in **Diagram View**.
5. This relationship is called a one-to-one relationship since the `User` table was split to make the relationship, as given in the following screenshot:



UserId	FirstName	LastName	Gender
1	John	Peter	1
2	Steffi	Christina	2
3	Robert	Bosco	1

DimUser

UserId	Zip	UsedInvitationCount
1	75017	5
2	41000	10
3	41000	30

FactUser

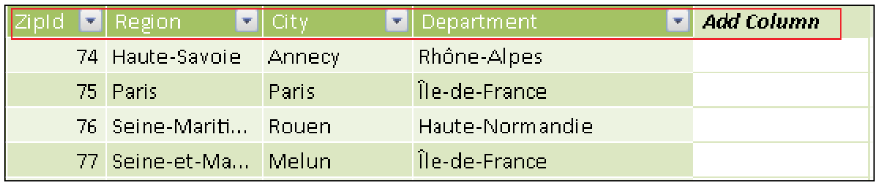
Relationship between DimUser and FactUser

Second, a relationship with `FactUser` and `DimDate` will be created by performing the following steps:

1. In the **Diagram** view, select the `UserId` attribute from the `FactUser` table; right-click on it and select the **Create Relationship** option.
2. The user will see the **Create Relationship** dialog box in which the table and column boxes will be filled by `FactUser`, where `UserId` is the key attribute, but the users must select the `DimDate` table in **Related Lookup Table** and it will automatically detect `UserId` in the **Related Lookup Column** box.
3. Click on the **Create** button. This will create a new relationship between two tables that were given and now the user will be able to see the relationship between the two tables in **Diagram View**.
4. This relationship is a one-to-one relationship.

Third, a relationship with `FactUser` and `Departments` will be created. This is quite different from the preceding two relationships. This e-commerce customer activities' data is from a French e-commerce industry's sample data, and hence it has only French customers from France.

While importing the `Departments` table there are no column headers in the flat file, so it automatically takes and creates fields such as **F1**, **F2**, **F3**, and **F4** and later renames them as `ZipId`, `Region`, `City`, and `Department` respectively.



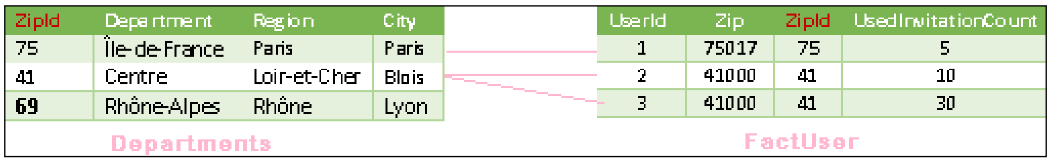
ZipId	Region	City	Department	Add Column
74	Haute-Savoie	Annecy	Rhône-Alpes	
75	Paris	Paris	Île-de-France	
76	Seine-Mariti...	Rouen	Haute-Normandie	
77	Seine-et-Ma...	Melun	Île-de-France	

The Department table

Generally the **Geography** hierarchy is Country→ State → City but the **France** hierarchy is Department→ Region → City. In our `FactUser` table there is another key attribute, that is, `Zip`; by using this as a relationship in the `Department` table's key as `ZipId`, users will be able to analyze the customer activities with the geographical locations. This can be made possible but there is one complexity in order to create a relationship, as the France zip code is a full numeric value and its length is 5. It is stored in the `FactUser` table but in the `Department` table the `ZipId` attribute has first the two texts of the zip code. By using this the user can find the **Location** hierarchy of France. So we have to get the first two texts of zip and store as `ZipId` in the `FactUser` table, but for this the DAX text functions will be used. To learn more about this go to `http://social.technet.microsoft.com/wiki/contents/articles/685.powerpivot-dax-text-functions.aspx`.

Let's see how to create a relationship between the `FactUser` and `Departments` tables:

1. First, create `ZipId` in the `FactUser` table by using the DAX expression.
2. In order to get the first two texts from the data in the SQL, use `Substring`, for instance `Substring( zip, 1, 2)` but in DAX use this formula: `=LEFT([Zip],2)`. This is very similar to Excel and we need to mention the attribute name for the DAX formula by using `([])`.
3. In the **PowerPivot** Workbook, switch to **Data View** and select the `FactUser` table and move the cursor into the **Add columnheader** row and in the formula bar use this formula: `=LEFT([Zip],2)` to get the first two texts of zip from the `FactUser` table. Once the user clicks on **Enter** it will create a new **CalculatedColumn1** header, in that it will populate the first text of zip and later the user can rename it as `ZipId`.
4. Now, there is a structured attribute to make the relationship. In the **Diagram** view, select the `ZipId` attribute that was created by using the DAX expression from the `FactUser` table. Right-click on it and click on the **Create Relationship** option.
5. The **Create Relationship** dialog box will open in which the table and column box will be filled by `FactUser` and `ZipId` which is the key attribute. Select the `Departments` table in **Related Lookup Table** and it will automatically detect the `ZipId` attribute in the **Related Lookup Column** box.
6. Click on the **Create** button; it will create a new relationship with two tables that were given, and now the user will be able to see the relationship between the two tables in the **Diagram** view.
7. This relationship is called a one-to-many relationship between the `Departments` and `FactUser` tables.

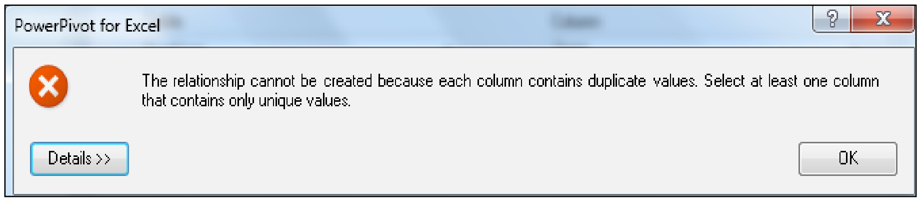


ZipId	Department	Region	City
75	Île-de-France	Paris	Paris
41	Centre	Loir-et-Cher	Blois
69	Rhône-Alpes	Rhône	Lyon

UserId	Zip	ZipId	UsedInvitationCount
1	75017	75	5
2	41000	41	10
3	41000	41	30

Relationship between Departments and FactUser

8. Rename the `Departments` table as `DimDepartment`. The important thing here is that if a correct attribute name is not provided while creating the relationship, it is not going make a correct relationship. The user will receive the error shown in the following screenshot, so please make sure that the attributes are given correctly while creating a relationship with the table:



The relationship cannot be created because each column contains duplicate values. Select at least one column that contains only unique values.

Wrong relationship error message

A successful split in the `User` table has been made and the creation of a relationship for a meaningful analysis by us has also been done. The reason why the table was split is that there are going to be many calculated columns as logical data from the physical model. If it is done only in the `User` table, it will increase the attribute value and there will be a big confusion. For this very reason, splitting of the table was done. Now, the calculated column, by using the DAX expression, will be created.

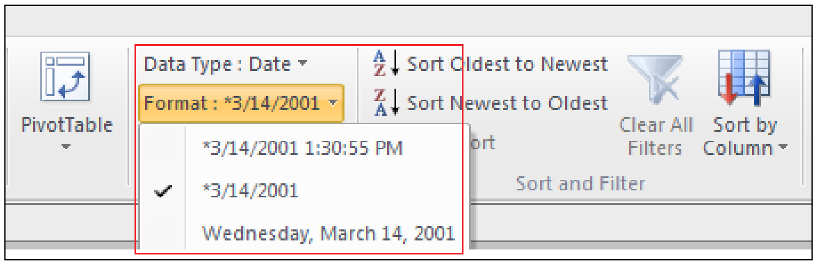
Adding a new calculated column using DAX and creating hierarchies for e-commerce data

Previously in the Sales Analysis PowerPivot Workbook, some DAX expressions were made and here in this workbook also some DAX expressions will be created. In the second chapter, eight functions were mentioned in the introduction of the DAX part which has many functions. So in order to learn everything, refer to this link: `http://technet.microsoft.com/en-us/library/ee634396.aspx`.

First create the calculated column in the `DimDate` table, which has only three attributes named `UserId`, `CreationDate`, and `LastConnectionDate`.
The `CreationDate` attribute consists of the users registered on the e-commerce website. By using this, users can analyze how many customers have registered by using data hierarchy. But there is only the `Date` attribute, so by using the date and time functions in DAX, attributes can be created. `LastConnectionDate` consists of the last login date of the user. Using this and based on the frequency of days, the customers can be divided into active and passive statuses.

In order to make calculated columns in the `DimDate` table, perform the following steps:

1. In the **PowerPivot** window, select the `DimDate` table where the `CreationDate` and `LastConnectionDate` data are stored as the `DateTime` data type. But for this analysis, time is not necessary so just change the data into the **Date** format. In order to do this, on the **Home** tab there is one section named **Format** section; select the `CreationDate` header and in the formatting section the user will be able to see **Data Type** and **Format** of the column selected. The data type is data but **Format** shows date with time. So click on the format and select the **Date** type only and move the cursor into the **Add Columnheader** row.

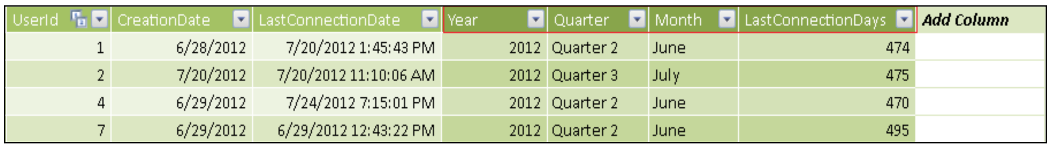


Data type format change

2. Now to create the `year` attribute from the `CreationDate` attribute, enter this formula: `=YEAR([CreationDate])`. It will populate only the year; a new calculated column will be created and we need to rename it `Year`.

3. We need to create Quarter and Month name from the `CreationDate` attribute in order to analyze the number of customers by quarter and month. In order to create a `Quarter` column use this formula: `=CONCATENATE("Quarter " ,FORMAT([CreationDate],"Q"))`. Once the user presses *Enter*, the quarter name will be given to the `CreationDate` attribute and we need to rename the column. Similarly, to create the `Month` column use this formula: `=FORMAT([CreationDate],"MMMM")`. Once the user presses *Enter*, the month name will be given to the `CreationDate` attribute and we need to rename the column.

4. To get how many days it has been since the last connection, use the `LastConnectionDate` attribute with this formula: `=1. * (TODAY()-[LastConnectionDate])`. This will return days with decimal numbers, since `Today()` returns with the present date and time and also `LastConnectionDate` is in the `Date` time date type. So, it calculates days with hours and when we make an analysis the result would be in decimal numbers. Since it does not make sense, change the data type to a whole number in the formatting section and rename the calculated column as `LastConnectionDays`. For instance, to find the duration on an hourly basis, use this formula: `=24. * (TODAY()-[LastConnectionDate])` and for a duration in minutes, use `=24.* 60*(TODAY()-[LastConnectionDate]`. In Excel, there is a `DATEDIF` function and in SQL too that does the same calculations.



UserId	CreationDate	LastConnectionDate	Year	Quarter	Month	LastConnectionDays	Add Column
1	6/28/2012	7/20/2012 1:45:43 PM	2012	Quarter 2	June	474	
2	7/20/2012	7/20/2012 11:10:06 AM	2012	Quarter 3	July	475	
4	6/29/2012	7/24/2012 7:15:01 PM	2012	Quarter 2	June	470	
7	6/29/2012	6/29/2012 12:43:22 PM	2012	Quarter 2	June	495	

Calculated column in the DimDate table

5. In the table, it is easy to identify which are the attributes for the calculated columns since they will be in dark green color.
6. Create date hierarchies for the `DimDate` table by following the steps that explained how to create hierarchies. Likewise, a hierarchy has been created `CreationDateHierarchy` like Year→Quarter→Month→CreationDate.

Next, to create a calculated column in the `DimUser` table, there are three attributes `BirthDate`, `Gender`, and `Type`. By using these attributes some calculated columns can be created.

Let's see how to create a calculated column in the `DimUser` table:

1. In **Data View**, select the `DimUser` table.
2. First, create `Age` from the `BirthDate` attribute by using this formula: `=Year (NOW ()) - Year ([BirthDate])`. After the column gets populated with age, save `Age` as the column name.

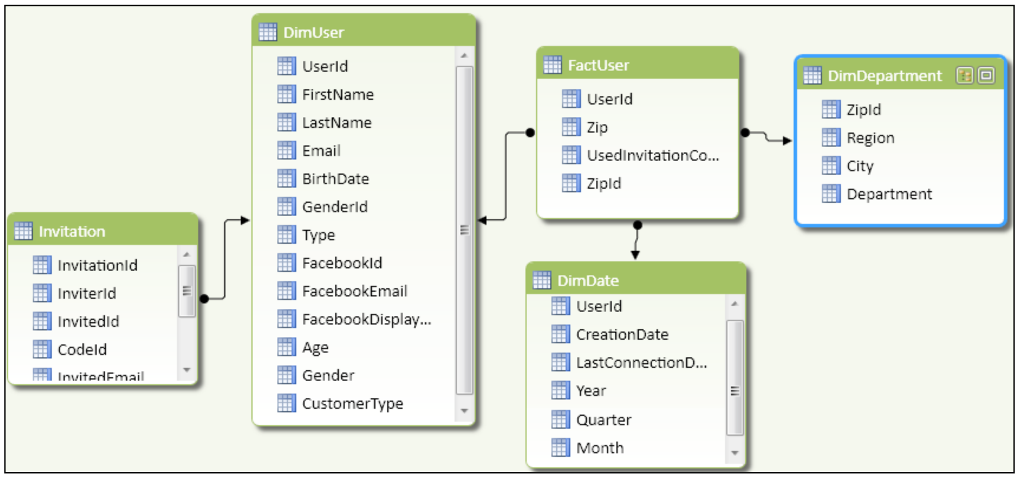
3. Second, there is a `Gender` attribute that can have the values `1` or `2`, `1` for male and `2` for female. If a numeric value is stored in a relational database it will reduce the size of the storage and the users might use the bitmap index for optimization in, order to speed up the query. But for reporting, only the logical name is going to be used. Create an attribute by using this formula (we rename it as `GenerId` since it is going to act only as an ID): `=SWITCH([GenderId],1,"Male",2,"Female","Unknown")`. Now populate the data and rename the attribute as `Gender`.
4. Third, if there is a `Type` attribute that consists of data as given in the following table, for example, **Physical Data**, but there is a need to change it to logical data for analysis, use the formula `=SWITCH([Type], -1, "Anonymous", 0,"Requester",1,"Member",2,"Member  Unlimited",3 ,"Curator",4,"Curator Friend",5,"Founder", "Unknown")`. This will populate the logical data based on the `Type` attribute and later rename the calculated column as `CustomerType`.

Physical Data	**Logical Data**
-1	Anonymous
0	Requester
1	Member
2	Member Unlimited
3	Curator
4	Curator Friend
5	Founder

5. Finally, the calculated column and hierarchy for e-commerce customer data analysis has been successfully created. The hierarchy for the `Departments` table can also be created but it has only three or four attributes, hence it's not suitable for this data analysis.

Viewing the logically designed data model for e-commerce data analysis

Finally, the data model for customer activities in the e-commerce PowerPivot Workbook has been designed. In order to view the logical design, use **Diagram View** as given in the following screenshot:



Logical data model of e-commerce data

Summary

This chapter allows the user to explore the features of data model and the fundamental concepts of relationship and hierarchy. The user, by now, should know how the ETL process and DAX expression are performed in PowerPivot, and how to use them to create structured data from unstructured data. Finally, the user will know how data can be homogenized from heterogeneous data.

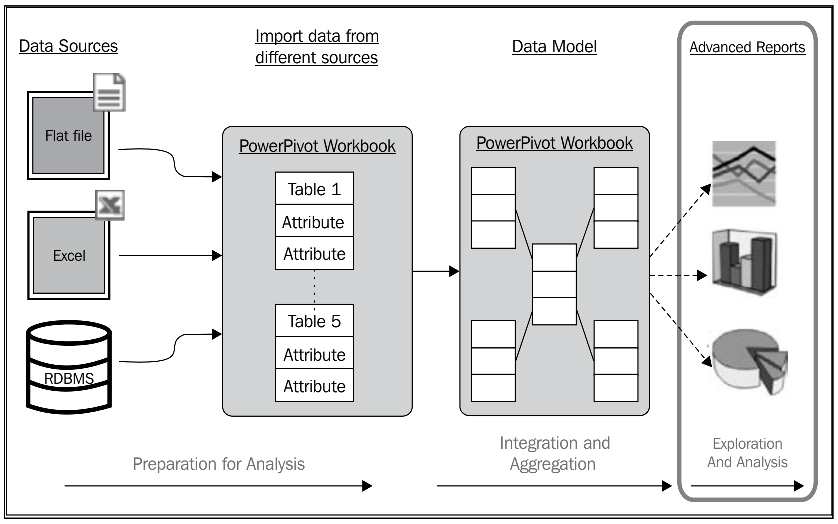
More precisely, practical information about two industrial data has been clearly explained in order to make a good logical data model design, in which the user will know how to use PowerPivot to create a relationship, hierarchy, and make calculated columns, and do some troubleshooting if they encounter any errors while practising any of the functions. So, finally two different types of logical data model designs have been shown practically.

The next chapter fully covers BI analytical reporting, metrics, KPI, and so on, based on the data model that we have created in this chapter and explains how to make a report layout and plot the data based on the layouts. The most important thing is that these reports help the end users in making a decision.

4 Business Intelligence Solutions Analysis

Business Intelligence (BI) deals with decisional processes. It enables a company to transform its business data into timely and accurate information. Business intelligence systems are used by decision makers to get comprehensive knowledge of the business and the factors that affect it, as well as to define and support their business strategies. The goal is to enable data-driven decisions aimed at gaining a competitive advantage, improving operative performance, responding more quickly to changes, increasing the profitability of the company.

This chapter gives a typical explanation of BI reports. The users will be given instructions on how to use the reports and make advanced reports using the PowerPivot interface and, also, the vital concepts of each report will be explained. The users will learn how to create decision-making reports from a large volume of data.



Advanced reports

The preceding diagram illustrates the last part of the principle of advanced reports, that is, creating the PowerPivot Workbook. In order to make a decision, a good, logical data model has been successfully designed using PowerPivot features and DAX expressions for the two different types of industrial data, that is, sales and e-commerce. This data will be used to create advanced reports.

A general overview of BI reports creation

Traditionally, as discussed earlier, the three principles of the PowerPivot interface will be implemented using separate tools and different people can be involved in doing so. One person will not do everything, since they might have to deal with lots and lots of tables and a variety of data sources. For big data warehouses, it will be kind of a big process and it's possible to complete it with some manpower.

Once they have designed the data warehouse schema, they have to create a data mart (provide decisional information for a specific business area, for example, sales, marketing, and human resources) for the end user to be able do the analysis using different kinds of tools for different analyses, such as multidimensional analysis with the OLAP tool, complex reports using the reporting tool, and predictive analytics with the data mining tool, so they can then take a decision.

More precisely, the data mart will be done for big data warehouses, for instance, an organization that has several departments, such as sales, marketing, management, and customer service. While doing the analysis, the users will be given the necessary data for specific departments, for instance, if they want to give the data for the sales department in their first analysis, the users must import only the necessary fact table and dimension tables for the sales department.

OLTP versus OLAP

On-Line Transactional Processing (**OLTP**) systems execute transactions that generally read/write the number of rows from/to many tables connected by simple relations. The essential workload core is *frozen* in application programs and ad hoc data queries are occasionally run for data maintenance. OLTP queries are executed on the operational source databases and they should be based on normalized data. Examples of OLTP systems include *ERP, CRM, SCM,* **Point-of-Sale** *(***POS***) applications, and call centers.*

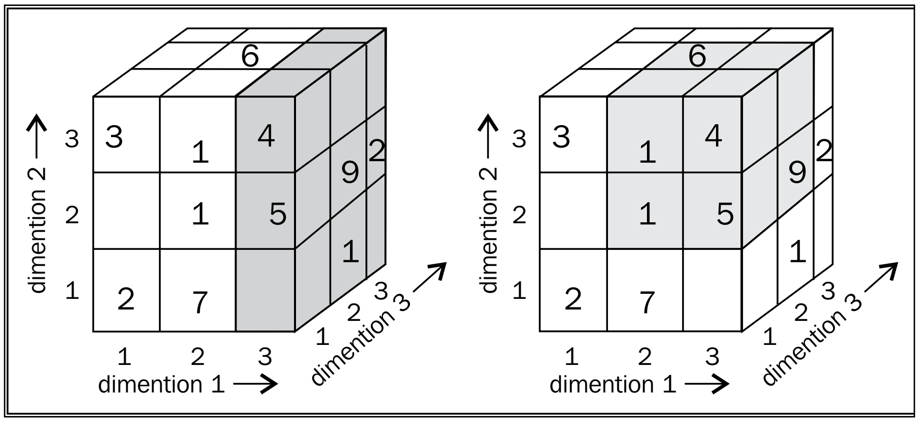
On-Line Analytical Processing (**OLAP**) systems deal with dynamic, multidimensional analyses that need to scan a huge amount of records to process a set of numeric data summing up the performance of an enterprise and also carry out the interactive and exploratory execution of complex queries over large volumes of data, and deliberately over the denormalized data model, in order to speed up the query.

As the goals of these two systems are totally different, they cannot coexist in the same process—a mix of analytical queries with transactional routine queries inevitably slows down the system.

The logical conclusion is to separate OLAP from OLTP by creating a new repository that integrates data from various sources and then makes it available for analysis and evaluation aimed at taking decisions for processes.

OLAP analyses

The main goal of OLAP is to facilitate, summarize, synthesize, consolidate, and browse to apply formulas to the data depending on several dimensions. When dealing with a data warehouse, a multidimensional model has to be managed in which data is represented in an intuitive way, closer to the user's perception. In a data warehouse, the facts of interest are represented in cubes; this is illustrated in the following diagram:



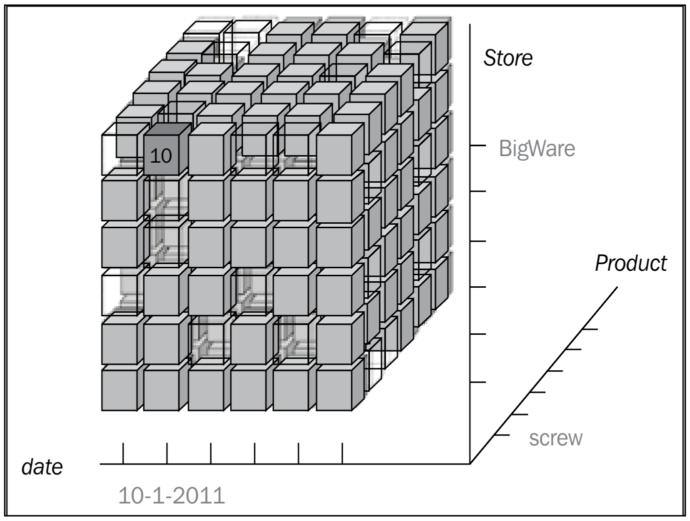
Example of an OLAP cube

The OLAP cube allows business users to slice and dice, drill down, roll up, and pivot the data. Normally, the data in an organization is distributed in multiple data sources which are incompatible with each other. Part of the OLAP implementation process involves extracting data from the various data repositories and making them compatible. The OLAP cube is typically stored as either a star or snowflake schema.

The goals of OLAP are as follows:

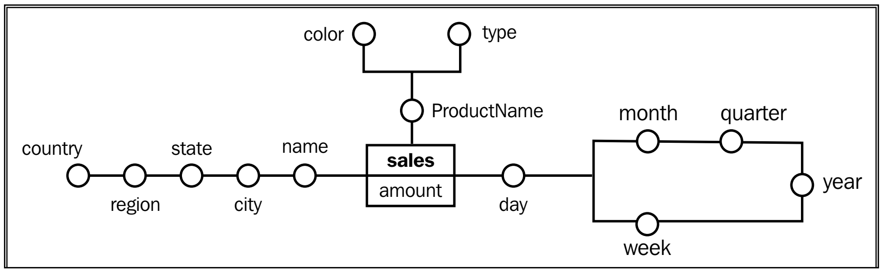
- To provide effective query mechanisms using interactive and successful queries and those which are understood by end users
- To free analysts from building complex models and writing complex queries

The following diagram illustrates a sample OLAP cube model. There is a `SalesFact` table of sales and a query is expressed over several dimensions (**store**, **product**, and **date**). **10** is the numerical value of one of the measures of the `Sales` table. It means that, while selecting the values **BigWare**, **screw**, and **10-1-2011**, the results of the measure will be **10**.



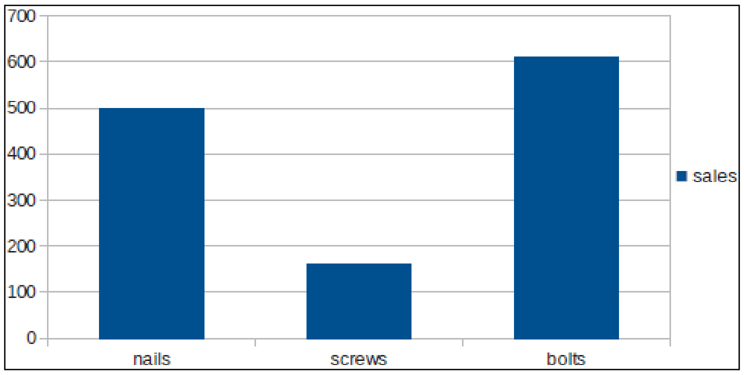
Sample OLAP cube model

The following diagram is an example of an OLAP analysis over a `SalesFact` table of sales; the user would know how to make a decision. As mentioned earlier, once the data warehouse and the data mart analyses are done, queries can be processed over them to get results that decision makers can use. The following diagram illustrates one fact table, namely `Sales`, and three dimensional tables: `Product`, `Time` and `Location` (however, cubes are not limited to three dimensions, they can have more). As mentioned earlier, the OLAP model is typically stored as either a star or snowflake schema. This has already been discussed, since the following example has one fact table and the peripheral tables are dimensional tables:



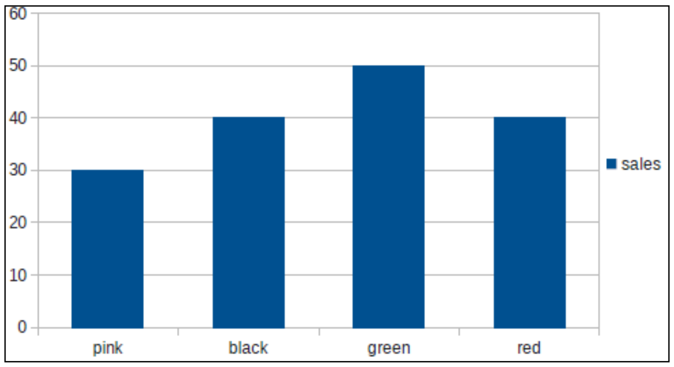
The logical data model for sales

The following model expresses sales in accordance with the location, product, and time dimensions:



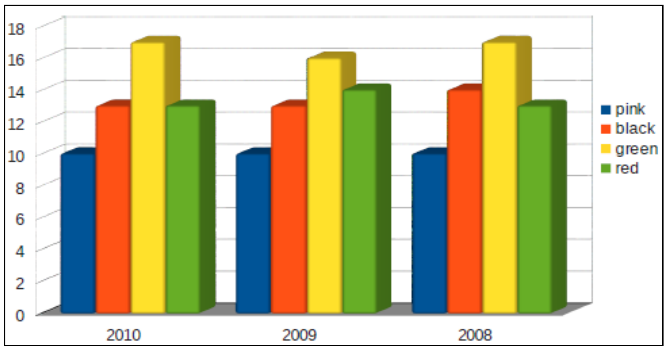
Sales based on the type of product

The preceding diagram shows a simple analysis of sales over various products. It can be seen that the number of screws that have been sold are less compared to nails and bolts. This leads to the main questions of this analysis: is there any reason for this unexpected, lower number of sales? Why did the screw sales go down? Which colors were responsible? Was it caused due to a specific color of screws? So, let's analyze screw sales details based on color, as shown in the following diagram:



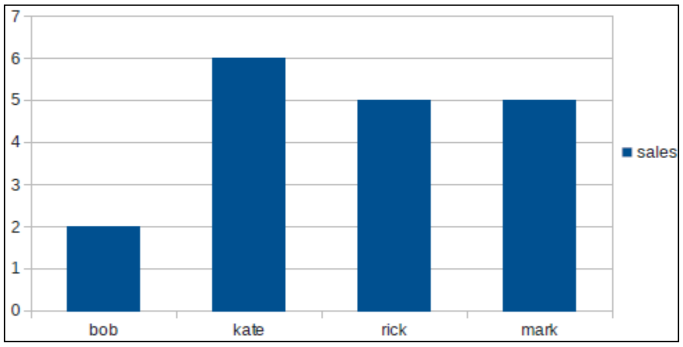
Sales of screws by color

The preceding diagram indicates that fewer pink screws have been sold than those in other colors, but the difference is not very huge. Let us do another analysis: let us cross the colors with the years, since the decision makers think that the sale of screws by color is specific to a particular year, as shown in the following diagram:



Sales for screws per year and by color

The same conclusions are reached as in the preceding diagram—we cannot determine the reason for the lower number of sales for the screws. So, finally: What is the exact cause? Are the salespersons the reason? Salespersons' analysis can be tried; but, in our example, there is no salesperson dimension table, so just assume that there is a salesperson table and we are performing a sales analysis based on this table, as shown in the following diagram:



Sales for screws by salesperson

This last analysis reveals the cause of the problem: the salesperson **bob** sells much less than the other vendors!

The conclusions of this mini analysis are as follows:

- Pink screws are sold less than other colors
- Bob sold fewer screws than the other salespersons

What can the analyst (and then the managers) do with such conclusions? They have to find answers to the following questions:

- What do my customers think about pink screws?
- Do my competitors face the same problem? What do they do about it?
- Let's have a look at some social networks, news bulletins, and so on.
- Is Bob my boss's relative?

Once all these questions have been answered, a decision can be taken by the managers in order to increase the productivity of the company.

All the previous results can be derived by querying the OLAP cube and its traditional BI analysis. In the reporting tool, it is not necessary to write an OLAP query; there are drag-and-drop technologies so that the end user can easily make the reports.

The importance of reports

Before creating advanced reports using the PowerPivot interface, the user must know the importance of reports and how, based on them, the organization will benefit. Some typical reports will be explained in this section.

The goals of reports are as follows:

- To allow a wide audience to benefit from the BI data
- To facilitate easy distribution of data (mail, the Web, Intranet, and so on) inside the organization and to external customers
- To consolidate data from different sources/tools (for example, OLAP and data mining) by including all kinds of functions and presentations
- To offer powerful visualizations which allow for the easier understanding of tendencies and relationships
- To allow analysts to focus on data and relationships among data

The benefits for the organization are as follows:

- Data will be available for the entire organization in a variety of formats that ease the distribution and may be sent by mail, accessed via the Web, and so on
- Static reports do not need training
- Reports may provide certain interactivity options (for example, templates or parameters) and limited analytical capabilities

Dashboards

Decision makers will be able to see the data from a variety of sources on a single screen. The goal of dashboards is to allow decision makers to see a variety of data that affects their departments and divisions. A dashboard has more objectives and is more detailed than a business scorecard and is generally personalized for each user.

Decision makers view a variety of information targeted to their department. This allows them to focus on only the items over which they have control. Information in a dashboard is more detailed than that of a scorecard and the tools in the dashboard often have better analytic capabilities than a scorecard.

A dashboard generally contains a variety of views to represent data, such as charts and tables focused on a functional area; it may also include scorecard elements. Data generally consists of KPIs, but also shows trends, breakdowns, and comparisons against a forecast or historical data. It can contain other data as well; for example, e-mail, news, stock quotes, and so on.

Business scorecards

In business intelligence technology, the scorecard report is one of the prominent concepts for decision makers. It was developed by *Dr. Robert Kaplan* (from Harvard Business School) and *Dr. David Norton*. They detected that companies only used financial indicators because financial metrics are easy to obtain and follow; but, unfortunately, financial indicators hide some aspects of company health. Hence, there is a need for developing further metrics for "balancing".

The main goal of a scorecard is to provide a quick visual representation of the organization's health and provide a high-level representation of the main business operations and their objectives. The scorecard data must be as recent as possible to make it more effective.

By having a quick look at the scorecard, you can see the performance and health of the business and the organization, obtain an instant value without making an analysis, and visualize the current values, but also compare the values with those during planning and, also, the tendencies. This kind of information will be useful in order for executives to make decisions.

Dr. Robert Kaplan and *Dr.David Norton* felt that a balanced scorecard categorizes metrics into the following four groups called Perspectives:

- Financial
- Business process
- Customer
- Learning and growth

Perspectives are defined as follows and information for each, which will help the decision makers of a particular organization, is given:

The financial Perspective contains:

- **Timely financial metrics**: This uses automated solutions to capture and process current/recent financial data

The business process Perspective contains:

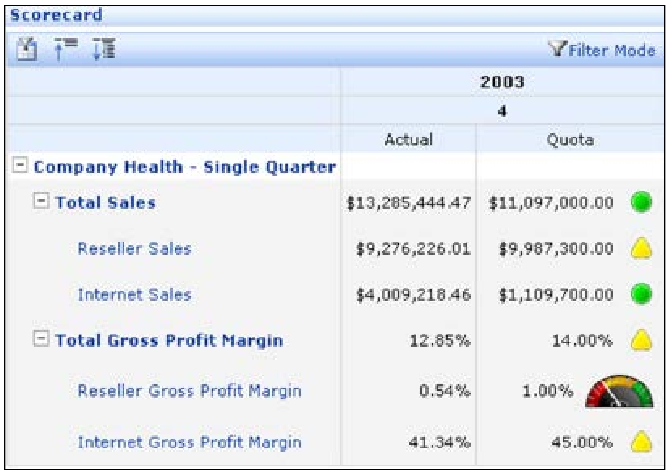
- Metrics related to internal business processes
- Alignment with the stated business goals: This is measured against goals and not history and may be new from year to year

The learning and growth Perspective contains:

- Metrics related to employee training, individual improvement, and corporate improvement
- Access to metrics is critical (for example, an effective Intranet site)

The customer Perspective contains:

- Metrics related to customer satisfaction
- Leading indicators of future results which are often useful to segment customers to improve the level of details available



Scorecard

Filter Mode

	2003		
	4		
	Actual	Quota	
Company Health - Single Quarter			
Total Sales	$13,285,444.47	$11,097,000.00	●
Reseller Sales	$9,276,226.01	$9,987,300.00	▲
Internet Sales	$4,009,218.46	$1,109,700.00	●
Total Gross Profit Margin	12.85%	14.00%	▲
Reseller Gross Profit Margin	0.54%	1.00%	◔
Internet Gross Profit Margin	41.34%	45.00%	▲

An example of a business Scorecard

A scorecard usually contains (at least) one of the following elements:

- **Key Performance Indicators** (**KPIs**)
- KPIs and the current and historic values, illustrating the tendencies and evolution for trend analysis
- KPIs and actual values compared with the forecast or targets
- Rankings for different departments, products, and so on

Key Performance Indicators (KPIs)

Another important concept is KPIs. They will clearly convey the business goals of the industry to the decision makers. KPIs are the first thing that the CEO of an organization requests before making any decisions and then, later, considers other reports. It is a kind of metric; for example, there are two important KPIs for the retail industry, which are as follows:

- **Conversion rate**: `Buyers/members * 100`
- **Acquisition rate**: `Members/ Invitations * 100`

KPIs are a measure of success of an organization or a particular activity. Success may be defined in terms of making progress towards strategic goals; for example, increases in availability, reduced mean delivery time, and the repeated achievement of some level of operational goals (for example, zero defects, 10/10 customer satisfaction). KPIs have either an expected or target value associated with them, which indicates health/success.

Here, the user can find some good examples of KPIs for some other departments, which will help the user's business to grow.

In order to make good decisions for the marketing department, the users can use the following points to create a KPI:

- Acquiring new customers
- Potential customers
- Status of the existing customers
- Customer attrition

In the following list, the users can find some typical KPIs for the retail industry, but lots of KPIs can be created for the retail industry based on the following requirements of the client:

- Conversion rate
- Acquisition rate
- Inventory to sales ratio
- Sales incremental

In the following list, the user can find some typical KPIs for the IT department:

- Availability
- Mean time between failures
- Mean time to repair
- Unplanned availability

Depending on their business and client requirements, the user would be able to create some typical **KPIs** for their business.

Creating a report using the PowerPivot interface

Now, the users will have a clear idea about general data analysis and report creation, the traditional BI concepts, and especially multidimensional (OLAP) analysis, which have been clearly explained; some typical report explanations have also been given. PowerPivot supports the multidimensional data source which can only be added from **Microsoft SQL Server Analysis Services** (**SSAS**).

An OLTP system is nothing but an operational data source from which **SQL** queries can be used to get millions of rows from/to many tables connected by simple relations. Let's take **Point-of-Sale** *applications (***POS***)*, wherein, in the operational database, there are around *214* tables and each table has lots of attributes. In order to perform analyses and make reports for these applications, importing all the tables isn't necessary since it's very difficult as all the tables follow a physical design. For these kinds of applications, SQL, the PowerPivot interface's features, and **DAX** are used. Using these features, users can extract the necessary attributes and import them into the PowerPivot Workbook. In the PowerPivot interface, there is an option for the *SQL* query to import the table. Once imported, the table can create the relationship.

In our case, two kinds of industrial data have been imported. This data has been imported only from the *OLTP* system because the *OLAP* data source hasn't been used and this data source has also been imported from operational data sources. The best example is our e-commerce data. But, PowerPivot works better with structured data such as the dimensional model. Once the data is imported into PowerPivot and the required data is selected, it is structured like an *OLAP* model. Hence, the e-commerce table has been split and made into a dimensional model. Since the sales analysis data has a structured format already, when the users have to make exact relationships, it becomes like the **Snowflake schema**. The reason PowerPivot becomes like an *OLAP* model is that it solves the *ETL* problem. Based only on the *ETL* process, the traditional dimension model can be made. Hence, in PowerPivot, *ETLs* are available.

The user is also obliged to know now about that **tabular model** in Microsoft Analysis Services. Now there are two models in SSAS: the **multidimensional model and the tabular model** which is called the BI semantic model. The tabular model is very similar to PowerPivot because every feature is similar to the data source support. **DAX** is the business logic of the tabular model and the design type of the tabular model is called Tabular; the PowerPivot design type is also tabular. But, the Tabular model's storage is within in-memory databases; in analysis services, security and speedup query are different, since their data access is an in-memory direct query. The Tabular model is not a traditional BI tool, but, today, a lot of organizations plan on using this model. However, PowerPivot features are more similar to the Tabular model. The BI semantic model can also be called the PowerPivot interface. The BI semantic model is the only model for all end user experiences, whether it is Personal BI (PowerPivot for Excel), TeamBI (PowerPivot for SharePoint), or Corporate BI (Analysis Service).

Data mart is a kind of subject-oriented data and it has already been explained. PowerPivot leverages self-service business intelligence for small data warehouses. So, more precisely, there will be one type of subject-oriented data. And, if the users have more subject-oriented data for their analysis, they have to create separate PowerPivot workbooks for each department analysis. For instance, if the users have sales and marketing data in their operational data source, they would have given the sales data for only the sales department team and the marketing data for only the marketing department team. In this case, the users can't give the team the same PowerPivot Workbook, shown earlier in the chapter, about the sales and marketing data warehouses. In such cases, the users will have to create separate PowerPivot workbooks for sales and marketing and import the relevant tables and make relationships. In our case, there are two kinds of data: one for the **sales analysis** workbook, which is for the sales department team to analyze the data, and the other one is the **e-commerce** industry data. But, our PowerPivot Workbook has customer activities data, so we need to give a report for **Customer relationship management** (**CRM**) in order to analyze the customer's activities.

Finally, everyone would be able to understand how PowerPivot works on BI concepts. Now, the third principle will be performed; this means that advanced reporting will be done from the two PowerPivot workbooks that have been created using the PowerPivot interface.

More precisely, PowerPivot gives **ad hoc reporting** facilities to the end user (non-technical end users), which means that the end user can prepare questions based on their company's data. They will create reports based on that, because PowerPivot gives *drag*-a*nd-drop* features in order to create reports from the PowerPivot data. For this, it is not necessary to contact the IT people in order to make reports. The best example is that, here, the data has been imported and a data model has been developed for sales analysis. This data will be given to the sales team in order to make a decision based on the data. Hence, the sales team members will have to prepare a business question and then, based on that, they would create interactive reports to make a decision and, for this, they do not require any IT knowledge, which is why it is called **ad hoc reporting**.

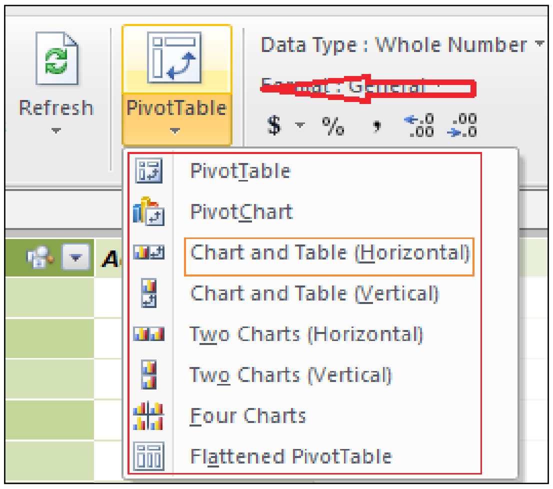
Creating simple reports in PowerPivot

How to create simple reports will be shown using the e-commerce PowerPivot Workbook data and the user will learn some of the vital features of report functionalities in PowerPivot. Analysis of a customer in the e-commerce website by a classified customer type will be shown in this section. In the previous chapter, the type of the customer is mentioned and that is what will be shown in this report.

In order to create our first report, we need to perform the following steps:

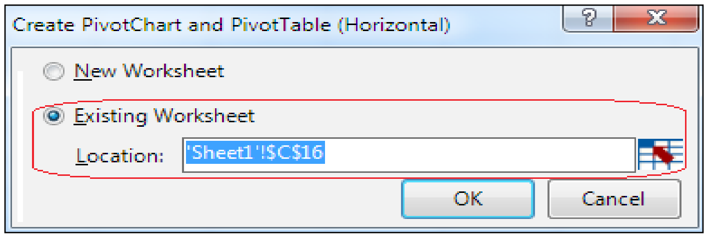
1. The Ecommerce PowerPivot Workbook has been opened from the local drive. Once opened, the user can click on the **PowerPivot** tab from the ribbon and then click on the PowerPivot window.
2. It launches the e-commerce data with all tables and relationships and our newly calculated columns. In other words, using this report, we will create exactly the same structured data model which was created in the previous chapter.

3. The user can find the **PivotTable** function between the **Get External Data** and **Formatting** sections. By clicking on **PivotTable**, the user will receive eight kinds of report layout options, as shown in the following screenshot:



PowerPivot report layout options

4. Click on the **Chart and Table (Horizontal)** report layout option to make our first report; after clicking on that option, the user will receive the **Create PivotChart and PivotTable (Horizontal)** dialog box in which there are two options, namely **New Worksheet** and **Existing Worksheet**. We have to decide which worksheet has to be used to create the report. If the user selects the **New Worksheet** option, a new Excel worksheet opens and the user can create the report. In this example, **Existing Worksheet** has been selected and the `Ecommerce.xlsx` file will be opened. An Excel file will be opened, after we click on **PowerPivot**, in order to go to the PowerPivot Workbook. In the Excel file, the users would already have sheets and new ones can be added. We have selected the **Existing Worksheet** option and, in the **Location** box, we need to enter `'Sheet1'!$C$16`. This means, that reports will be made in `Sheet1`. Finally, we click on the **OK** button, as shown in the following screenshot:


Create PivotChart and PivotTable (Horizontal)

New Worksheet

[✓ Existing Worksheet]

Location: 'Sheet1'!$C$16

Worksheet Selection

5. After clicking on the **OK** button, the user report layout will be launched, in which there is one chart and table layout, and, to the right-hand side, there is a PowerPivot field list in which the user can find all the tables that are present in the Ecommerce PowerPivot Workbook and the user can find the report creation functionalities in the following screenshot (the user will learn all the features by the end of this chapter):

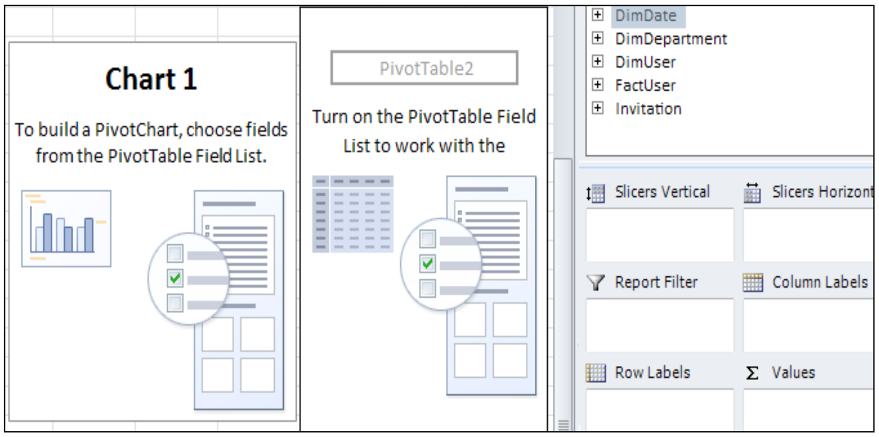


Chart 1

To build a PivotChart, choose fields from the PivotTable Field List.

PivotTable2

Turn on the PivotTable Field List to work with the

Report layout on the PowerPivot interface

6. If the user expands the table, he will see all the attributes of the tables. The first chart is going to be created, so we need to select the chart and expand the `DimUser` table, and drag the `CustomerType` attributes into **Axis Fields(Categories)**, which is the horizontal axis of the chart and, here, the user will not be able to see the following screenshot since this screenshot was taken while selecting the **Table** layout features. But, if the user selects the **Chart** layout, the user will find **Axis Fields(Categories)** instead of the **Row Labels** option and, since the report is supposed to show how many customers there are based on the customer type, a count of all the customers has to be shown. To show a count of all the customers, drag **UserId** into the ∑ **Values** box (it's for aggregation function features and mainly uses the fact table attribute since measures are available only in the fact table) from the `FactUser` table in order to count the customers based on the type of customers. As soon as the user drags `UserId`, the sum of the IDs will be displayed. But, the count of the customers has to be calculated, so click on the sum of UserId from the ∑ **Values** box. This will display the properties of this option and the user will find some useful features. Here, the user will find the **Edit Measures** option; this will display the **Measure settings** dialog box and the user will see the aggregation function option that they can use in the PowerPivot option. In the **Table Name** option, the user will see **FactUser** and, in **Source Name**, they will see **UserId**, which shows the table and attributes that the user used for the ∑ **Values** box and the count of the **UserId** values in the **Custom Name** box will be displayed by selecting the **Count** function; but, it has been renamed to **Total Customer** and, at the bottom of the screen, in the **Measure will use this formula** box, the user will find the automatically converted DAX function of this measure. This is one of the best features in PowerPivot—the drag-and-drop reporting technology:

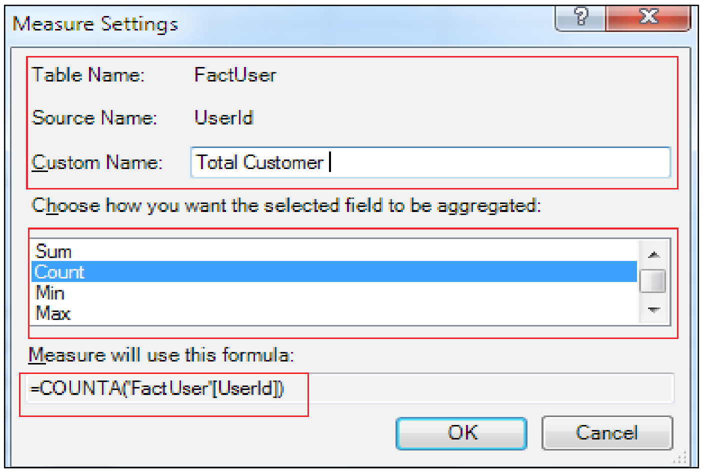


Table Name: FactUser

Source Name: UserId

Custom Name: Total Customer

Choose how you want the selected field to be aggregated:

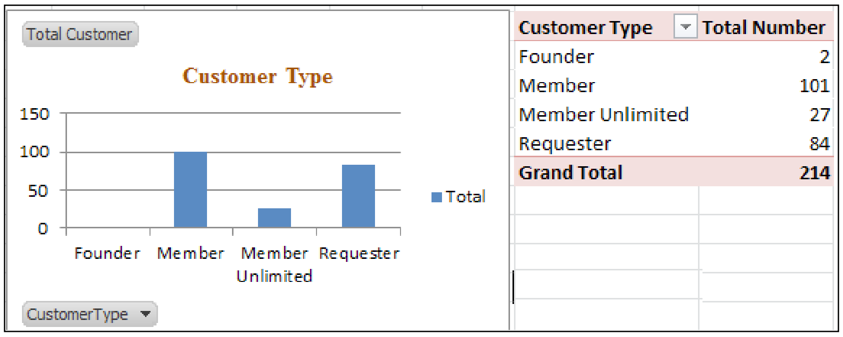
Sum
Count
Min
Max

Measure will use this formula:

```
=COUNTA('FactUser'[UserId])
```

Measure Settings dialog box

7. Click on the **OK** button; now, the user will see the chart and rename it as `Customer Type`. Users might wonder why `UserId` has been used from the `FactUser` table in order to count the number of customers instead of the `UserId` attribute from the `DimUser` table. Of course, that can also be done. But, here, the formal approach (`UserId` from FactUser) was used since tables (`FactUser` & `DimUser`) are related in a one-to-one relationship and, in the `DimUser` table, `UserId` is available and each user has one unique ID. If it is a one-to-many relationship, we can use many relationship tables' attributes, generally the `fact` table for the aggregation function. The user will learn more about this in the rest of the chapter.
8. Now, we will create a table and, using the data in the table, the chart will be created. Select the table layout and the user can see, in the right-hand side table, the layout features. As shown in the preceding screenshot, drag the `CustomerType` column from the `DimUser` table into **Row Labels** and drag the `UserId` column from the `DimUser` table into the ∑ **Values** box. Change **Count** to make it the aggregation function in order to count the customers and the user will see the table and change the column names as the table requires. Now, notice that two attributes have been used from the same table, but, for our chart, we have used the two tables; it depends on the attributes of reports and relationships among the tables.



Customer Type	Total Number
Founder	2
Member	101
Member Unlimited	27
Requester	84
Grand Total	**214**

Customer type report

9. One small report has been created and we have renamed the sheet as **Customer type**. While creating the next report, we have to use the next sheet and so on. If all the reports have been created in the same Excel file, it will be useful, since everything will be in one file. However, this depends on the end user.

Creating a PivotTable

PivotTable is a prominent report of data analysis. Comparison, finding relationships, and discovering trends and differences can be done here; it also gives drill-down capabilities to reports, which means that the rows and columns are supposed to have a one-to-many relationship. Hierarchy is created, as it is useful for the creation of PivotTable reports.

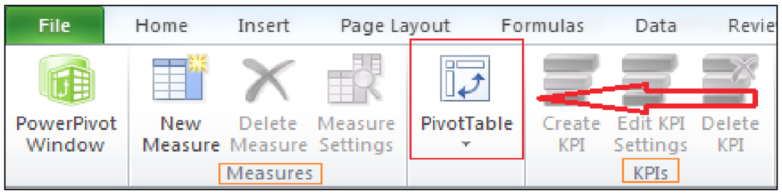
The Sales Analysis PowerPivot Workbook is going to be used in order to create Pivot tables and the rest of the reports too; it will be helpful since the user will be able to do work. As it has many tables, attributes, and data, it will be helpful to show the users some more advanced and interactive reports compared to our Ecommerce PowerPivot Workbook.

Now `Sales amount` and `Sales quantity` will be analyzed by the `DimDate` and `DimChannel` tables and the `Sales amount` and `Sales quantity` attributes are from the `FactSales` table; this is because these attributes are measures and `DimChannel` has details of the store channel information, such as `Catalog`, `Online`, `Reseller`, and `Store`, and apparently knows that the `DimDate` table has the date information for the sales transactions.

In order to create the Pivot table, perform the following steps:

1. Open the Sales analysis PowerPivot Workbook from the local drive. After opening the workbook, the user needs to click on the **PowerPivot** tab from the ribbon and then click on the PowerPivot window.
2. The sales analysis data will be launched with all the tables and relationships and the new calculated column, which is exactly the structured data model which was created in the previous chapter. Using this, reports can be created.
3. Click on the **Home** tab in the PowerPivot window and then click on **PivotTable** and select the **PivotTable** layout, which is the first among the eight layouts. Select the **Existing Worksheet** option and, in the **Location** field, enter `'Sheet2'!$B$2` and click on the **OK** button. Then, the PivotTable layout will be displayed in Sheet 2. Start from the **B** column in the second row; now, the user will know the reason why `'Sheet2'!$B$2` was entered in the **Location** field instead of the sheet's location. Move the cursor into the Excel sheet where the user wants to create the report in the sales analysis Excel workbook. The stored tables are only there in Sheet 1, as the user can't create the report in the existing table.

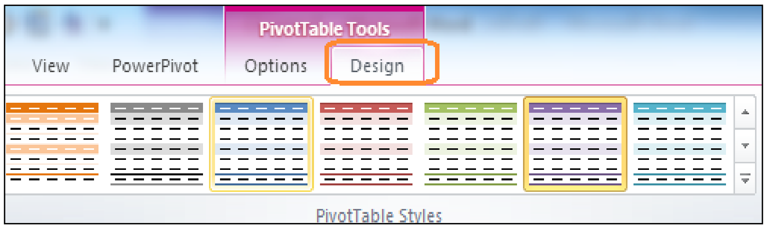
4. Another way to select the report layout in the sales analysis Excel workbook, and not in the PowerPivot Workbook, is by selecting the **PowerPivot** tab from the ribbon. The user will be able to find the **PivotTable** option between the **Measures** and **KPIs** sections (this is not available in Microsoft 2013). By clicking on **PivotTable**, the same eight report layouts will be displayed; reports can also be created by clicking on this and it will be used in our next report since the **PivotTable** layout has already been selected in Sheet 2 of the sales analysis Excel workbook, as shown in the following screenshot:



PowerPivot report layout selection option

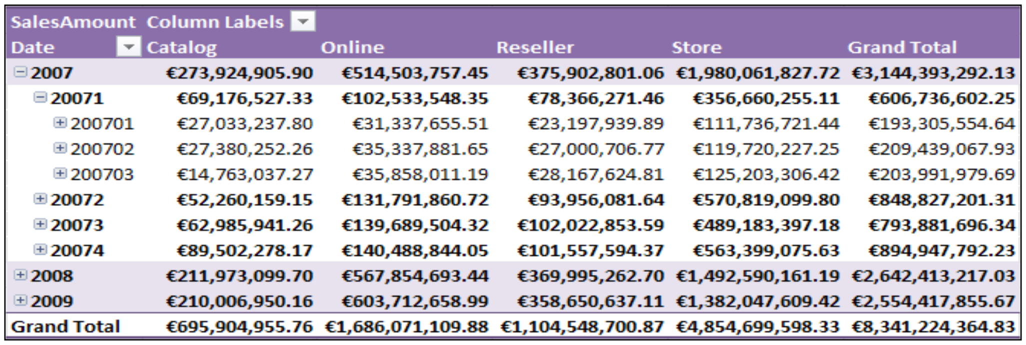
5. The user will be able to see the **PivotTable** layout in Sheet 2. Now expand the `DimDate` table from the **PowerPivot Field** list and drag the `Dates` attribute (it's the hierarchy of dates that we have created in the previous chapter) into **Row Labels**. The date hierarchy is the row data of our PivotTable. Once done with the `Dates` attributes, the user will be able to see the `Date` hierarchy in the PivotTable's row, which will display only years; if the user expands it, all the dates will be displayed in a `CalendarYear` | `CalendarQuarter` | `CalendarMonth` | `CalendarDate` hierarchy. Now expand the `DimChannel` table and drag the `ChannelName` attribute into **Column Labels**; the channel name will be displayed in the column of our Pivot table. Select the measures in order to fulfill the PivotTable and drag the `Sales amount` and `Sales quantity` attributes into the ∑ **Values** box from the `FactSales` table the measures attributes automatically calculate the sum aggregation function (the function is kept the same, since the total sales amount and sales quantity by date and channel is being calculated). We need to rename the `Sales Amount` attribute to `Sum of Sales Amount` using the **Measure Settings** dialog box and change the row's name to `Date`.

6. Now, the user will be able to see the PivotTable, but with the default design. So, in order to change the design, select the created PivotTable and, in the **PivotTable Tools** option's ribbon, the user will be able to see the **Options** and **Design** tabs. Click on the **Design** tab and the user will be able to see several designs of PivotTable, as shown in the following screenshot. By selecting a design, the user can change the design of the PivotTable:



PivotTable designs

7. Select one design among the several and the user will be able to see the following PivotTable:



SalesAmount	Column Labels				
Date	Catalog	Online	Reseller	Store	Grand Total
⊟ 2007	€273,924,905.90	€514,503,757.45	€375,902,801.06	€1,980,061,827.72	€3,144,393,292.13
⊟ 20071	€69,176,527.33	€102,533,548.35	€78,366,271.46	€356,660,255.11	€606,736,602.25
⊞ 200701	€27,033,237.80	€31,337,655.51	€23,197,939.89	€111,736,721.44	€193,305,554.64
⊞ 200702	€27,380,252.26	€35,337,881.65	€27,000,706.77	€119,720,227.25	€209,439,067.93
⊞ 200703	€14,763,037.27	€35,858,011.19	€28,167,624.81	€125,203,306.42	€203,991,979.69
⊞ 20072	€52,260,159.15	€131,791,860.72	€93,956,081.64	€570,819,099.80	€848,827,201.31
⊞ 20073	€62,985,941.26	€139,689,504.32	€102,022,853.59	€489,183,397.18	€793,881,696.34
⊞ 20074	€89,502,278.17	€140,488,844.05	€101,557,594.37	€563,399,075.63	€894,947,792.23
⊞ 2008	€211,973,099.70	€567,854,693.44	€369,995,262.70	€1,492,590,161.19	€2,642,413,217.03
⊞ 2009	€210,006,950.16	€603,712,658.99	€358,650,637.11	€1,382,047,609.42	€2,554,417,855.67
Grand Total	€695,904,955.76	€1,686,071,109.88	€1,104,548,700.87	€4,854,699,598.33	€8,341,224,364.83

Sales by Channel and Date PivotTable

8. In the PivotTable, the user can know the sales based on the current and past years' records and make year-wise and channel-wise comparisons. They can do a drill-down by expanding the `Date` row and see the subtotal and grand total of the sales amount. If the **Dates** hierarchy hasn't been created, the user has to drag all the attributes into **Row Label** because the end user will get confused, especially non-IT people; in order to avoid these difficulties, the hierarchy is supposed to be created.

9. If the user wants to delete the PivotTable and use another layout in the Excel sheet, apparently, PowerPivot also provides these options. In order to delete the PivotTable, select the created PivotTable. In the **PivotTable Tools** ribbon, the user will be able to see **Options** and **Design** and the user needs to click on **Options** to be able to see the **Select** option in the **Action** section. By clicking on the **Entire PivotTable** option, the user will be able to see the selected PivotTable. Now, click on the **Home** tab and the **Delete** option in the cells between the **Styles** and **Editing** sections. Now, our PivotTable will be deleted if the **Delete** option is clicked on. ThePivotTable isn't deleted here in the example, but has been renamed as `Sales` by channel and date and saved in the Excel workbook.

Creating a PivotChart

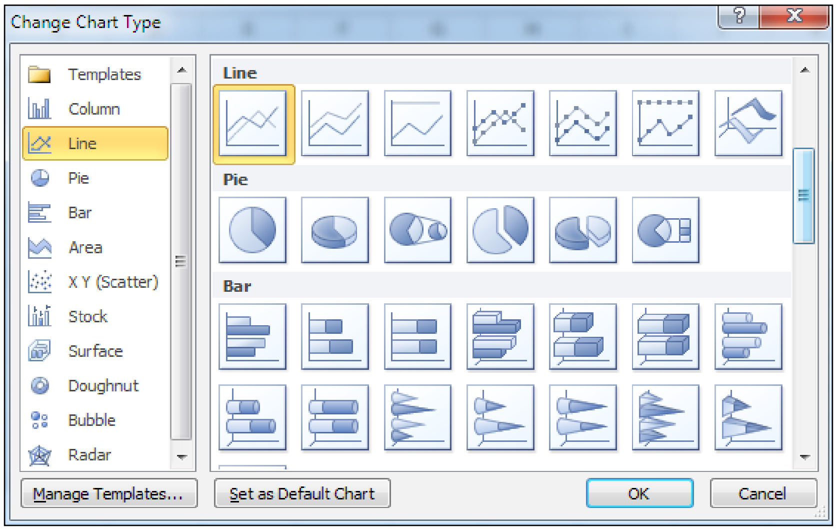
Another vital report is PivotChart, which is a kind of interactive graphical representation that is easier to understand for the end user in order to make a decision. The end user is usually prepared for the PivotChart because, after seeing the PivotCharts, they will be able to understand how their business is doing without any typical explanation.

The PowerPivot interface provides many of the chart templates, such as column, line, pie, bar, area, X-Y (scatter), stock, surface, doughnut, bubble, and radar and each chart has a variety of types. Based on our analysis, an appropriate chart template has been used. For instance, a pie chart gives the percentage type and, using an X-Y (scatter) chart, the user will be able to see the customer cluster. For comparing classes or groups, the bar chart can be used, since it gives the graphical visualization and other features.

We can create two types of charts using the sales analysis PowerPivot Workbook by performing the following steps:

1. Select Sheet 3 in the Sales Analysis PowerPivot Excel workbook and move the cursor to the position where the PivotChart needs to be created. Click on the **PowerPivot** tab from the ribbon and click on the **PivotTable** option in order to select the PivotChart into Sheet 3. Select **PivotChart**, which, in the displayed list, is the second option among the eight report layouts, and use an existing worksheet Sheet 3; click on the **OK** button.
2. Now, the user will be able to see the **PivotChart** template in Sheet 3. A line type of chart is going to be created here. Hence, the first user will need to change the type of chart.

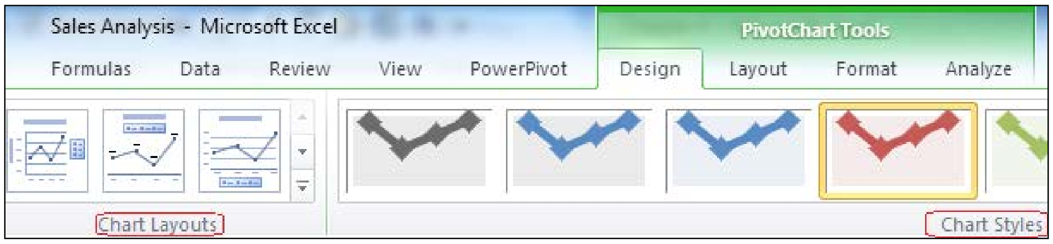
3. In order to change the type of chart, by selecting the empty chart, the user will be able to see **PivotChart Tools** in the corner of the ribbon. Under this tab, there are four options: **Design**, **Layout**, **Format**, and **Analyze**, which provide different kinds of features for the PivotCharts in the PowerPivot interface. By selecting the **Design** option, the user will be able to find the **Type** section, which is the very first. Here, there are two options: **Change Chart Type** and **Save as Template**. By clicking on the **Change Chart Type** option, the user will receive the **Change Chart Type** dialog box, in which they can find all the chart templates and chart types. In the following screenshot, the **Line** template and the first chart type **Line** have been selected:



Change Chart Type dialog box

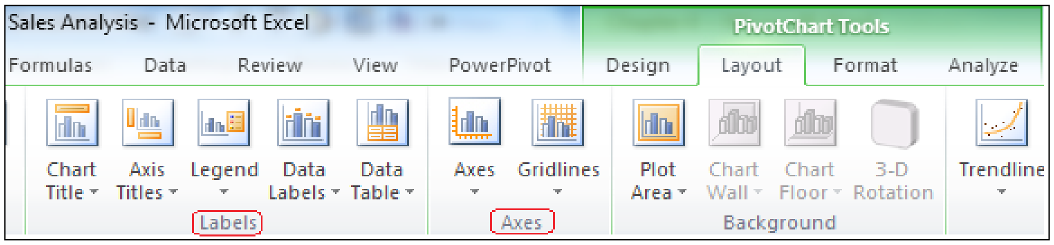
4. Two kinds of charts—sales by channel and sales by channel and year—are going to be created here. Previously, a PivotTable had been created and, based on that analysis, the same PivotTable is going to be shown to the user in the form of a graphical visualization.
5. Drag the `SalesAmount` attribute into the ∑ **Values** box from the `FactSales` table; place the sum aggregation function and rename the attribute as `Sales Amount`, which is the y axis in the Line chart, and drag the `ChannelName` attribute into **Axis Fields (Categories)**, which is the x axis in the Line chart. Now, the user will be able to see the Line chart in Sheet 3.
6. Now, the chart style and layout is going to be changed here; add the label names of two axes and the chart name so that the user can utilize the PivotChart tools.

7. In order to change the chart styles and layout, click on the **Design** option in the **PivotChart Tools** tab, so the user can find the chart layouts and chart styles section.



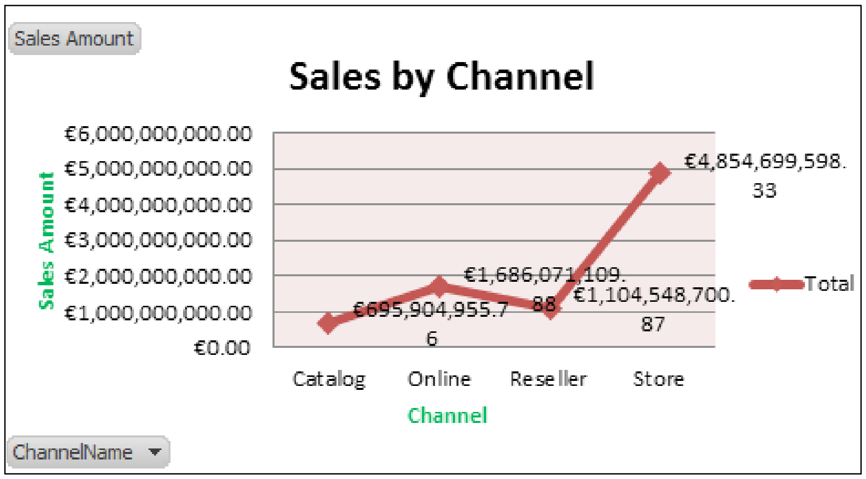
The chart layouts and chart styles section

8. The user can change the chart layouts and styles from this section in order to create a good graphical presentation for their clients. They can also use some different types of styles and layouts.
9. In order to add the label names of two axes and the chart name, the user needs to click on the **Layout** tab and then they will find the layout features; then, go to the **Labels** section. Using the tools in this section, users can change their requirements.



PivotChart Layout tab

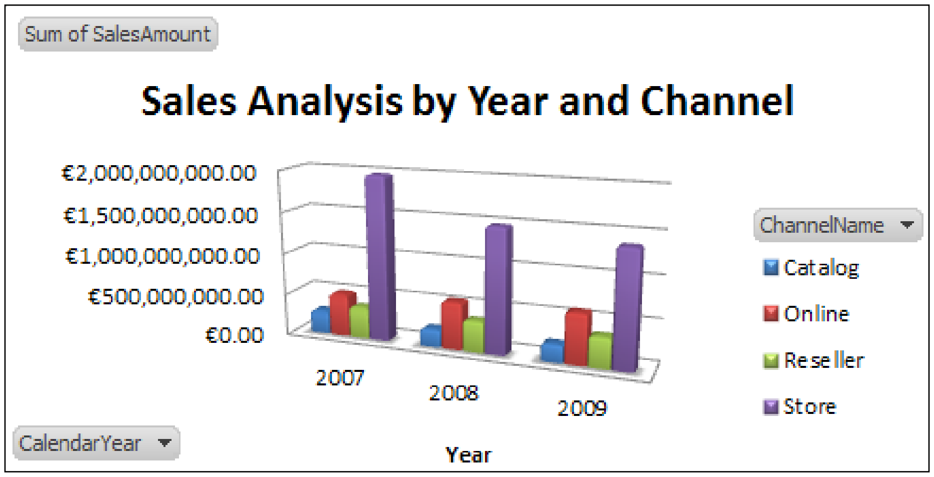
10. In the **Label** section, the user will find the **Chart Title** option. By clicking on this, the user will receive three options regarding where the title should be displayed in the chart. Here, the chart in the following screenshot was selected and renamed as `Sales by Channel`. When the user clicks on the **Axis Titles** option, they will find a lot of options for creating the axis title; in our example, one vertical and one horizontal title have been created and renamed and we have also changed their color. Now, the user can find the fully filled line chart, as shown in the following screenshot:



Sales by Channel PivotChart

11. By clicking on the **ChannelName** option in the report, the user can also do filtering.
12. Now, another PivotChart for sales analysis by year and channel is going to be created in the same worksheet. The user needs to move the cursor to the position where they want to create the PivotChart, select the **PowerPivot** tab from the ribbon, and add the PivotChart into the same worksheet.
13. Once added, change the **Chart Template** field to **Column** and select the **3-D Clustered Column** type, which means this chart will be displayed in 3-D.
14. Drag the `SalesAmount` attribute into the ∑ **Values** box from the `FactSales` table, keep the sum aggregation function, and rename the attribute as `Sales Amount`, which is the y axis in the chart. Drag the `CalendarYear` attribute into **Axis Fields(Categories)**, which is the x axis of the chart, and drag the `ChannelName` attribute into **Legend Fields (Series)**. Based on this value and the year, the PivotChart will be displayed.

15. As usual, change the chart title to `Sales Analysis by Year and Channel` and create the axis title; change the chart layouts and styles precisely, utilizing the **PivotChart Tools** option. The user will be able to see the chart 'as shown' in the following screenshot:



PivotChart of Sales by year and channel

16. By clicking on **ChannelName** and **CalendarYear** in the report, the user can also do filtering. Finally, we have created two PivotCharts successfully in Sheet 3 and renamed them as `Sales by year and channel`.

17. If the user wants to delete a PivotChart, they need to just right-click on the chart and select the **Cut** option.

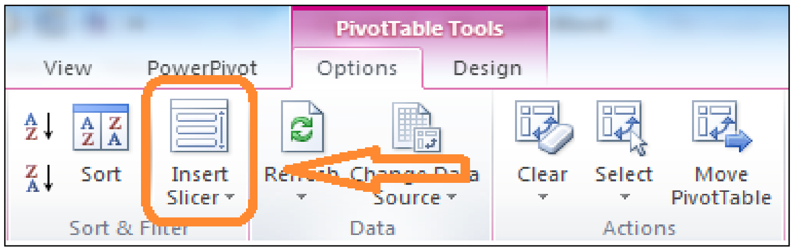
The user can easily create the PivotChart since it's not very complicated if a structured data model is available. The PowerPivot interface provides drag-and-drop technologies. There are a lot of PivotChart templates, so the user can try to create the PivotChart using all the template and chart types. If they try all the templates, they will know PivotChart fully and be able to deliver a PivotChart that exactly meets the needs of their client.

Creating a PivotTable with Slicers

Now the user knows how to create a PivotTable. Now the PivotTable will be created with Slicers. The Excel user would probably know about Slicers because they can be used in the Excel workbook and the PowerPivot Workbook. A Slicer is a kind of filtering feature that provides interactive reporting features. Slicers can be applied based on the report's requirement.

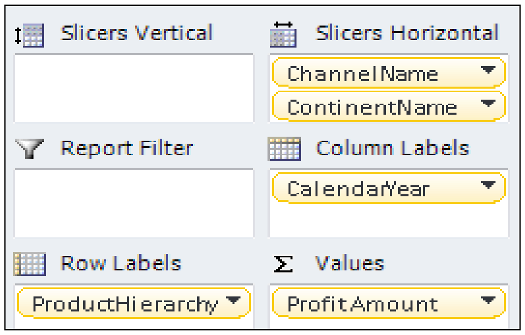
In order to apply Slicers in the PivotTable, a PivotTable is needed. Hence, the user can use the PivotTable that was created earlier, but a new PivotTable needs to be created to report another kind of analysis to which Slicers will then be added. To do so, perform the following steps:

1. Add a new PivotTable layout into a new sheet and drag the **Product Categories** hierarchy into **Row Labels** from the `DimProduct` table; drag the `CalendarYear` attribute into **Column Labels** from the `DimDate` table, drag the `ProfitAmount` attribute into the ∑ **Values** box from the `FactSales` table, and keep the sum aggregation function and rename it as `Profit Amount`.
2. Now the user will be able to see the PivotTable for `ProfitAmount` by products' categories and years. In this PivotTable, the end user will be able to find out about each and every product's price, its profit based on year, and which product sold most.
3. There are two ways to insert the Slicer. The first way is to select the PivotTable and click on the **Options** tab in **PivotTable Tools**. By doing this, the user will be able to find the **Sort & Filter** section between **Group** and **Data** as shown in the following screenshot:



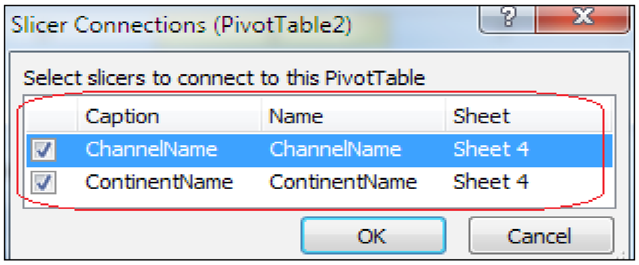
The Insert Slicer Option

4. By clicking on the **Insert Slicer** option, the user will be able to see two suboptions **Insert Slicer** and **Slicer connection** . Click on the **Insert Slicer** option; the users will be able to see all the tables and attributes of the sales analysis PowerPivot Workbook. By selecting the attributes, it will create the Slicer. The user can arrange the Slicer by dragging it to wherever they want to place it. The second way is via two boxes, namely **Slicers Vertical** and **Slicers Horizontal**, under **PowerPivot Field List**. The user can drag the attributes into the **Slicers** box. This will create the slicer; in the following screenshot, `ContinentName` has been dragged into **Slicers Horizontal** from the `DimGeography` table and the `ChannelName` attribute has also been dragged into **Slicers Horizontal** from the `DimChannel` table:



The added slicer

5. Now the user will be able to see the Slicer with the PivotTable in the report sheet, which is been renamed as `PivotTable with Slicers`. If the user clicks on **Slicer Connection** from the **Insert Slicer** option, the **Slicer Connections** dialog box will be shown in which they can see the details of the Slicers added in the worksheets as shown in the following screenshot:

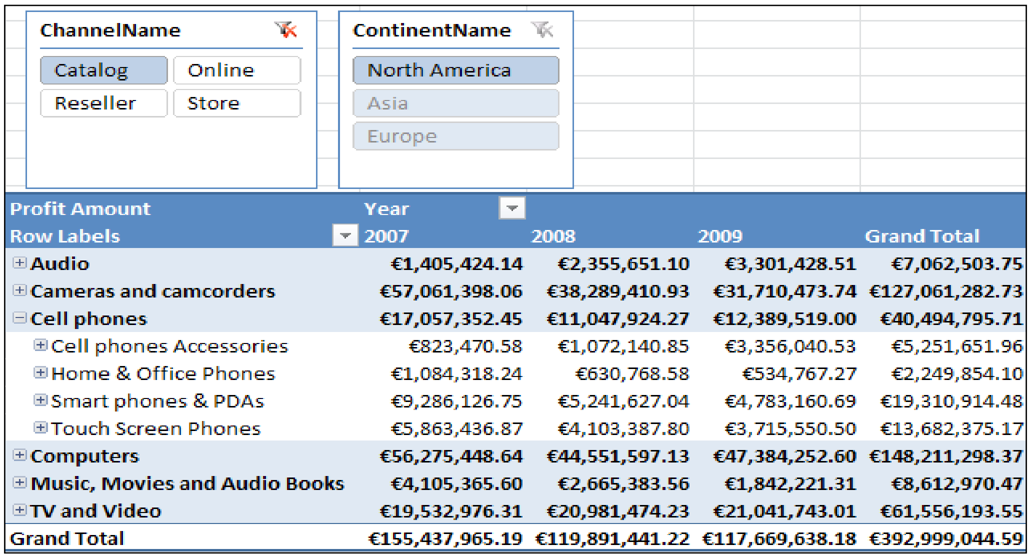

Slicer Connections (PivotTable2)

Select slicers to connect to this PivotTable

	Caption	Name	Sheet
✓	ChannelName	ChannelName	Sheet 4
✓	ContinentName	ContinentName	Sheet 4

Slicer Connections dialog box

6. The PivotTable with Slicers, namely `ChannelName` and `ContinentName`, have been created as shown in the following screenshot:



Profit Amount	Year			
Row Labels	2007	2008	2009	Grand Total
⊞Audio	€1,405,424.14	€2,355,651.10	€3,301,428.51	€7,062,503.75
⊞Cameras and camcorders	€57,061,398.06	€38,289,410.93	€31,710,473.74	€127,061,282.73
⊟Cell phones	€17,057,352.45	€11,047,924.27	€12,389,519.00	€40,494,795.71
⊞Cell phones Accessories	€823,470.58	€1,072,140.85	€3,356,040.53	€5,251,651.96
⊞Home & Office Phones	€1,084,318.24	€630,768.58	€534,767.27	€2,249,854.10
⊞Smart phones & PDAs	€9,286,126.75	€5,241,627.04	€4,783,160.69	€19,310,914.48
⊞Touch Screen Phones	€5,863,436.87	€4,103,387.80	€3,715,550.50	€13,682,375.17
⊞Computers	€56,275,448.64	€44,551,597.13	€47,384,252.60	€148,211,298.37
⊞Music, Movies and Audio Books	€4,105,365.60	€2,665,383.56	€1,842,221.31	€8,612,970.47
⊞TV and Video	€19,532,976.31	€20,981,474.23	€21,041,743.01	€61,556,193.55
Grand Total	€155,437,965.19	€119,891,441.22	€117,669,638.18	€392,999,044.59

PivotTable with Slicers

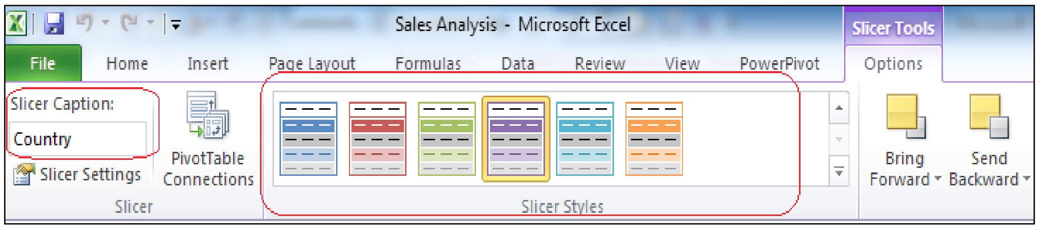
7. The preceding table explains that we have clicked the **Catalog** channel of sales from the **ChannelName** Slicer. Note that there is a filter applied in the **ContinentName** Slicer. You can see only the **North America** field and the remaining two continents are hidden, which means that in the **Catalog** channel, the sales of only North America are shown and the PivotTable shows the profit of sales by year and product category; when you click on another Slicer, it will automatically change the data in the PivotTable based on the filter done by it. If the user wants to remove the filter, they can do it by just clicking on the Filter icon in the Slicer.

Creating PivotChart with Slicers

The same features of the Slicers in the PivotTable will be applicable to the PivotChart table. Two PivotCharts and two Slicers can be created by performing the following steps:

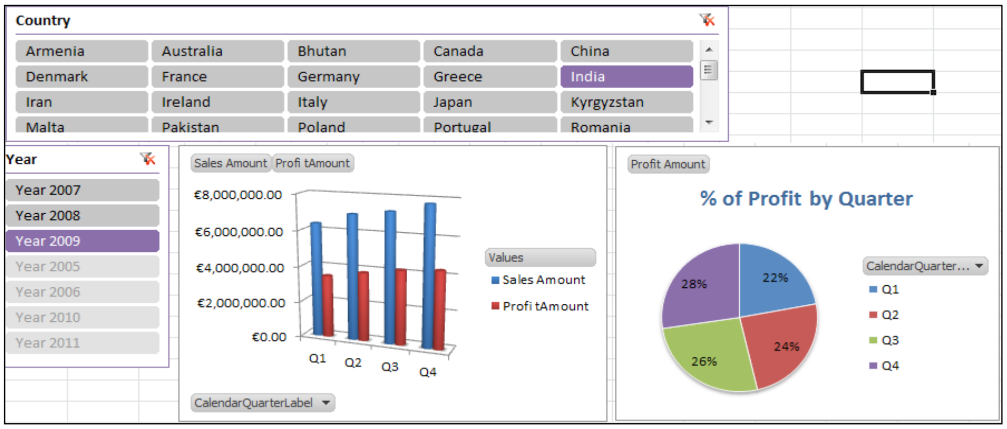
1. Add some extra empty sheets in the sales analysis PowerPivot Excel workbook and move the cursor where the PivotChart needs to be created. Select the **PowerPivot** tab from the ribbon and click on the **PivotTable** option in order to select the PivotChart into the sheet. Select **Two Charts(Horizontal)**, which is fifth among the eight report layouts, and use the existing worksheet; click on the **OK** button.
2. Now the user will be able to see the two PivotChart templates horizontally in the Excel sheet; analysis of the sales by quarter will be done in these reports.
3. Click on the first PivotChart and the user will be able to see the **PowerPivot Field List** pane on the right-hand side; drag the `CalendarQuarter` label into the **Axis Fields (Categories)** box from the `DimDate` table and rename it as `Quarter`; drag `SalesAmount` and `ProfitAmount` into the ∑ **Values** box from the `FactSales` table; keep the sum aggregation function and rename it as `Sales Amount` and `Profit Amount`. The user must note that the chart type hasn't been changed before dragging the attributes into the **Fields** box, which means that it will automatically use the **Column** template. We have kept the same template and changed the chart to the **Clustered Cylinder** type. Now, the user will be able to see the sales analysis by quarter in the first PivotChart.
4. Now select the second PivotChart, change the pie chart to a chart template and first keep the chart type as **Pie**. Drag the `CalendarQuarter` label into the **Axis Fields (Categories)** box from the `DimDate` table, rename it as `Quarter`, drag the `ProfitAmount` attribute into the ∑ **Values** box from the `FactSales` table, keep the sum aggregation function and rename it as `Profit Amount`. Change the chart layout. Now, the users will be able to see the percentage of profit amount by quarter and they need to rename the chart title `% of Profit by Quarter`.
5. Two Slicers are going to be used here, so drag `RegionCountryName` into **Slicers Horizontal**, which is located horizontally above the two PivotCharts, and drag the `CalendarYear` label into **Slicers Vertical**, which is located vertically beside the two PivotCharts.

6. The user can also change the Slicers' properties using Slicer tools. If the user wants to change the Slicer name, styles, and so on, they can do it by selecting the Slicer and the user will be able to see the **Slicer Tools** tab; they can change the styles using the **Slicer Styles** option and rename the Slicer. There is an option in the Slicer section in which there is a **Slicer Caption** box. The user can change the Slicer name there. The Slicer styles have been changed here and they has been renamed as `Country` from `RegionCountryName` and `Year` from the `CalendarYearLabel` slicer.



The slicer tools tab

7. The PivotCharts with Slicers have been created and shown in the following screenshot. The sheet has been renamed as `PivotCharts with Slicers`. In the *following screenshot*, sales analysis by quarter is shown for 2009 for India:



PivotCharts with Slicers

When users create a PivotChart, the data table will be created in another sheet in the same workbook. For instance let's assume % of Profit by Quarter PivotChart(Preceding screen shot) which was created in the sheet 6, and in chart 2 in the worksheet when the user creates the chart, they can see one extra sheet, namely **Data** for Sheet6 Chart 2, which means that the data for PivotCharts comes via PivotTable. Once the PivotChart is created, if the user deletes the `Dat` table of PivotCharts, that won't change the PivotCharts and, even when the user clicks on the PivotChart, the *PowerPivot Field List* won't be displayed any more. Hence, they must be careful and should not get confused with this at the time of creating a report.

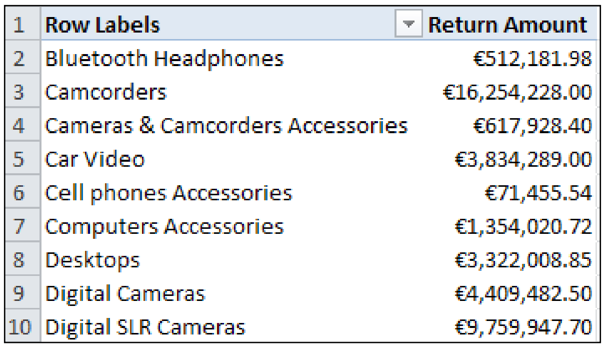
PowerPivot Dashboards

The Dashboard report has been clearly explained in the *Importance of reports* section. Hence, the user might already know about Dashboard reports. It can be simply said that decision makers would be able to see data from a variety of sources on a single screen and a report contains a variety of different views of data, such as charts and tables, focused on a functional area.

First, create a Dashboard report using the Sales Analysis workbook; this report will be focused on the sales return amount and sales return quantity using product category, year, and geography dimension tables.

1. Select the sheet where the user wants to create their report, select the **PowerPivot** tab from the ribbon, and click on the **PivotTable** option in order to select the report layout into the sheet. Select the **Four Charts** option, which is seventh among the eight reports' layouts, and use an existing worksheet; click on the **OK** button.
2. Now the user will be able to see the **Four PivotChart** template in the Excel sheet while the four sheets will be added as data for Sheet 7 Chart 1, data for Sheet 7 Chart 2, data for Sheet 7 Chart 3, and data for Sheet 7 Chart 1, since four chart layouts have been added in Sheet 7. Once the chart is created, the user will be able to see the data table of the chart.

3. Click on the first PivotChart now and the user will be able to see the **PowerPivot Field List** pane to the right-hand side. Keep the chart template as **Column**, drag the `ProductSubcategoryName` attribute into the **Axis Fields (Categories)** box from the `DimProductSubcategory` table and rename it `Product Subcategory`. Drag the `ReturnAmount` attribute into the ∑ **Values** box from the `FactSales` table, keep the sum aggregation function, and rename it `Return Amount`. The chart title has been renamed `Return amount` by `Product Subcategory`. Now, the user must note that in the data for Sheet 7 Chart 1, they are able to see the table shown in the following screenshot:

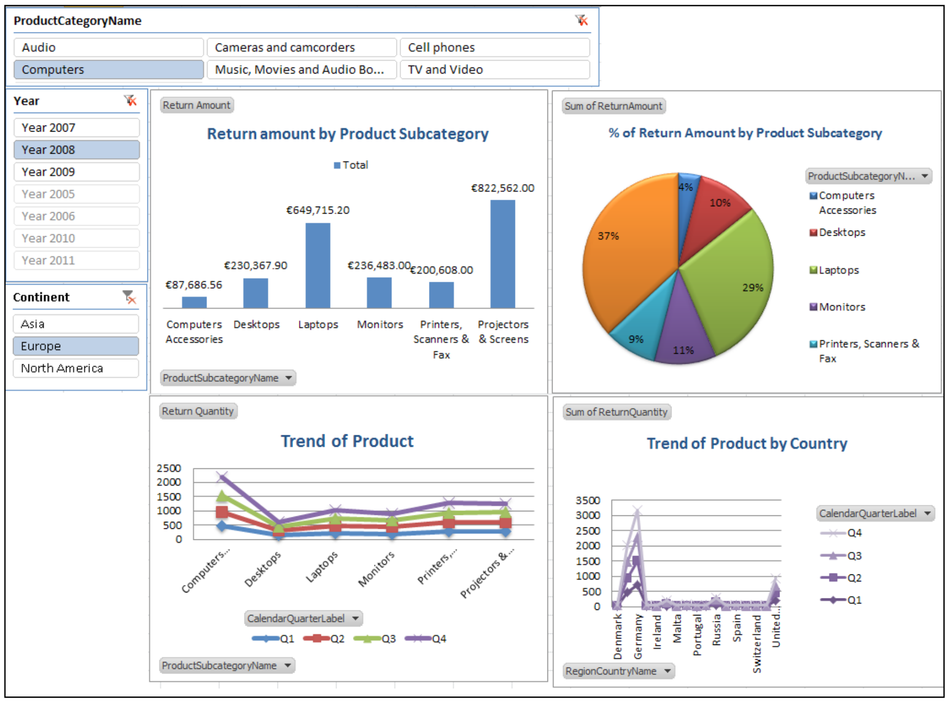


1	Row Labels	Return Amount
2	Bluetooth Headphones	€512,181.98
3	Camcorders	€16,254,228.00
4	Cameras & Camcorders Accessories	€617,928.40
5	Car Video	€3,834,289.00
6	Cell phones Accessories	€71,455.54
7	Computers Accessories	€1,354,020.72
8	Desktops	€3,322,008.85
9	Digital Cameras	€4,409,482.50
10	Digital SLR Cameras	€9,759,947.70

Data for Return amount by the Product Subcategory PivotChart

4. Now, select the second PivotChart, change the line chart to a chart template, and keep the chart type as **Stacked line with Markers**. Now drag the `ProductSubcategoryName` attribute into the **Axis Fields (Categories)** box from the `DimProductSubcategory` table and rename it as `Product Subcategory`. Drag the `ReturnQuantity` attribute into the ∑ **Values** box from the `FactSales` table, keep the sum aggregation function, and rename it as `Return Quantity`; drag the `CalendarQuarterLabel` attribute into the **Legend Fields (Series)** box from the `DimDate` table, rename it as `Quarter`, and rename the chart title as `Trend of Product`; also, change the styles.
5. Now, select the third PivotChart, change the pie chart to a chart template, and keep the chart type as **Pie**. Now drag the `ProductSubcategoryName` attribute into the **Axis Fields (Categories)** box from the `DimProductSubcategory` table and rename it as `Product Subcategory`. Drag the `ReturnAmount` attribute into the ∑ **Values** box from the `FactSales` table, keep the sum aggregation function, and rename it as `Return Amount`. The chart title has been renamed as `% of Return Amount` by `Product Subcategory` and the styles of the pie chart have also been changed.

6. Now select the fourth PivotChart, change the Line chart to a chart template, and keep the chart type as **Stacked Line**. Now, drag the `RegionCountryName` attribute into the **Axis Fields (Categories)** box from the `DimGeography` table and rename it as `Country`. Drag the `ReturnQuantity` attribute into the ∑ **Values** box from the `FactSales` table, keep the sum aggregation function, and rename it as `Return Quantity`. Drag the `CalendarQuarterLabel` attribute into the **Legend Fields (Series)** box from the `DimDate` table and rename it as `Quarter`. The chart title has been renamed as `Trend of Product` by country and the styles have also been changed.
7. Three Slicers are going to be added to this report in order to make a good, interactive dashboard report and notice that if Slicers are available in the report, different kinds of filtering can be done in order to analyze the data in a different manner.
8. Drag the `CalendarYearLabel` attribute from the `DimDate` table and `ContinentName` from the `DimGeography` table into the **Slicers Vertical** box. These Slicers that are located vertically change their names to `Year` and `Continent` and drag the `ProductCategoryName` attribute from `DimProductCategory` into **Slicers Horizontal**. This attribute will be located horizontally, change its name to `ProductCategory` and also change the styles of Slicers.



Dashboard report of the Sales Analysis PowerPivot Workbook

9. The dashboard report has been successfully created and the sheet is named `Dashboard report of Return`. The preceding screenshot illustrates the function of the return amount and return quantity of sales done in Europe in 2008 and only in the computer product category. The user can find the returns of the product and quantity details and also make different kinds of analyses using the Slicer.

> When the user adds the Slicer, it will be applicable to what the user has imported into the report layout. Suppose the user adds the new layout in the same sheet; the Slicer won't work for this newly added layout. The user is supposed to select the layout and click on the **Slicer connection** option from **Insert Slicer** in which the user can select the slicer they want to add to the selected layout.

Creating key performance indicators

The user must know the importance of KPIs since a typical explanation and some examples of KPIs for different industries have already been explained and explored. The KPIs will be created based on measures, measures will be created based on the aggregation function, and the user can define their own formula using the DAX expression. The measure will be created in order to be used in either a PivotTable or PivotChart.

Before creating the KPIs, the user has to create measures using DAX. Using these measures, KPI reports can be created. The Sales Analysis PowerPivot Workbook is being used here in order to create the KPIs. Two different kinds of KPI creation methods will be shown to the user.

First, the Average Sales KPIs are going to be created; let's see how to make measures and KPIs:

1. A new PivotTable layout has been added into the existing workbook, a new sheet has been added, and `ContinentName` and `RegionCountryName` have been dragged into **Row Labels** from the `DimGeography` table. Now, drag the `SalesAmount` and `SalesQuantity` attributes into the ∑ **Values** box from the `FactSales` table and make the sum aggregation function as the sales quantity. Now, the user will be able to see the PivotTable for the sales amount and the quantity by continent and country.

2. Now, create a newly calculated column in order to find average sales using DAX. Just move the cursor into the **Sum of SalesQuantity** row and, in the **PowerPivot** tab, the user will be able to see something called the **Measure** section, in which the user will find the **New Measure** option. By clicking on the **New Measure** option, the user will receive the **Measure Settings** dialog box, in which there is a **Table name** box; `FactSales` has been entered in that since the `SalesAmount` and `SalesQuantity` attributes are being used and they are from the `FactSales` table. Also, the measure will be stored in this table. The **Measure name (all PivotTables)** box has been named as `Average Sales` and **Custom name (this PivotTable)** has also been named as the attribute name of the calculated column. In the **Formula** box, provide the formula `= [Sum of SalesAmount] / [Sum of SalesQuantity]` to find the average sales. The user can understand this formula easily. There is a button named **Check formula**; by clicking on this button, the user can validate their formula. Finally, there are formatting options; using these, the user can change the data type of the newly calculated column. Here, the category has been changed to **CurrencySymbol** and the Euro symbol has been selected, and select **2** in the **Decimal places** field. Now, the user will be shown the extra calculated column named `Average Sales` next to the `Sum of SalesQuantity` column.

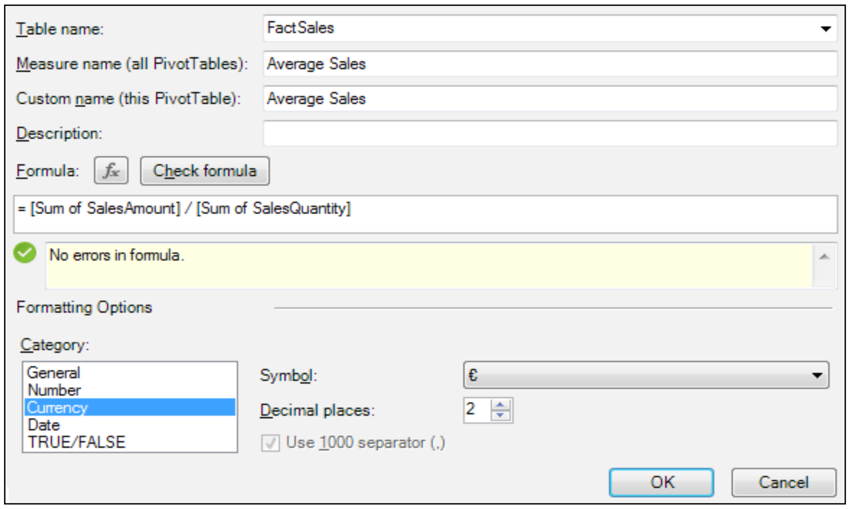
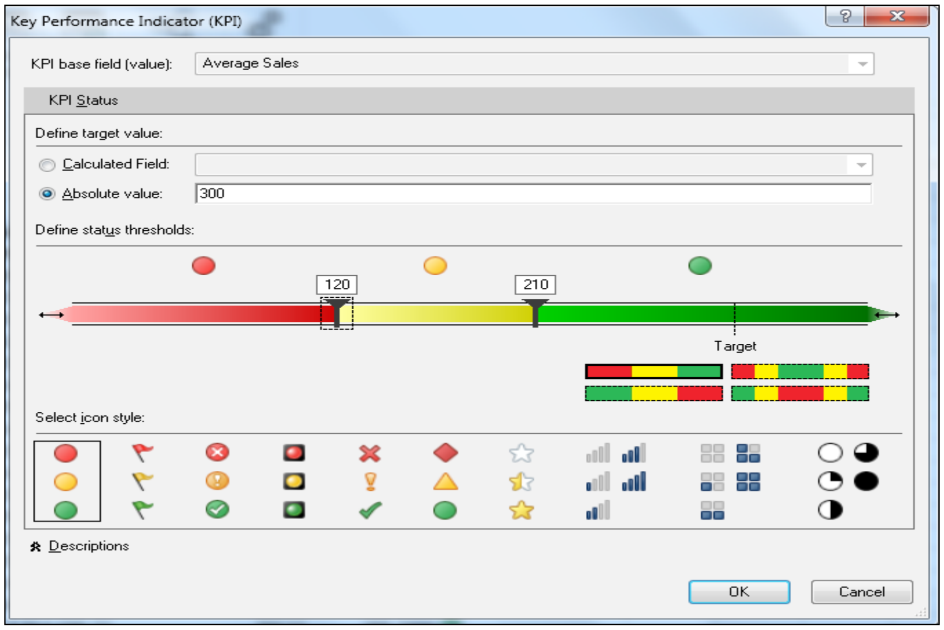


Table name: FactSales
Measure name (all PivotTables): Average Sales
Custom name (this PivotTable): Average Sales
Description:
Formula: = [Sum of SalesAmount] / [Sum of SalesQuantity]
No errors in formula.
Formatting Options
Category: General, Number, Currency, Date, TRUE/FALSE
Symbol: €
Decimal places: 2
Use 1000 separator (,)

The calculated field dialog box

3. In the Sales Analysis PowerPivot Workbook, switch to **Data View** and select the `FactSales` table. At the bottom of the window, the user will find the practice area (if they don't find it, they need to click on **[Calculation Area]** in the top-right part of the ribbon). There, the user will be able to find the **Average Sales** measure in the FactSales practice area below the `SalesKey` attribute column.
4. Right-click on the **Average Sales** measure and click on **Create KPI**. The user will receive the **Key Performance Indicator (KPI)** dialog box. There is an option box called **KPI base field (value)**, which is automatically set to **Average Sales**. The **Define the target value** section is a vital section in KPI creation and, based on the average sales value, it will be indicated to the end user. In this section, there is a **Calculated field** combo box and an **Absolute value** field in which the **Absolute value** option has been checked and `300` has been entered into it. Depending on the user, the thresholds have been split into three parts using the **Define status thresholds** section. For example, 0-120 average sales will be indicated in red, 121-210 average sales will be indicated in yellow, and 211-300 average sales will be indicated in green. Based on these indicators, the end user can easily identify the average sales status. Yet another option is **Select icon style**. Using this, the user can change the style of the indicator and, here, the default one is chosen. The following screenshot depicts all these sections:


Key Performance Indicator (KPI)

KPI base field (value): Average Sales

KPI Status

Define target value:

Calculated Field:

Absolute value: 300

Define status thresholds:

120

210

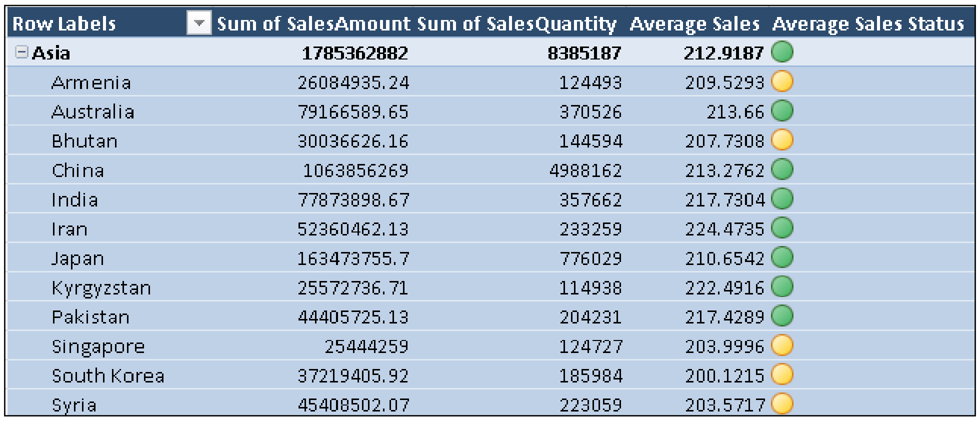
Target

Select icon style:

Descriptions

The Key Performance Indicator (KPI) dialog box

5. Click on the **OK** button. Our KPI has been created successfully. Select the PivotTable and the user will be able to expand the `FactSales` table from **PivotTable Fields List**. In the table, the user will find the `Average Sales KPI` attribute. By expanding this, the user will be able to see three checkbox options, namely **Value (Average Sales)**, which is our calculated column, **Goal** is what value has been entered in the **Absolute Value** field, and **Status** which is an indicator. Once, if the user checks the status, the indicators will be displayed as shown in the following screenshot:



Row Labels	Sum of SalesAmount	Sum of SalesQuantity	Average Sales	Average Sales Status
⊟Asia	**1785362882**	**8385187**	**212.9187**	🟢
Armenia	26084935.24	124493	209.5293	🟡
Australia	79166589.65	370526	213.66	🟢
Bhutan	30036626.16	144594	207.7308	🟡
China	1063856269	4988162	213.2762	🟢
India	77873898.67	357662	217.7304	🟢
Iran	52360462.13	233259	224.4735	🟢
Japan	163473755.7	776029	210.6542	🟢
Kyrgyzstan	25572736.71	114938	222.4916	🟢
Pakistan	44405725.13	204231	217.4289	🟢
Singapore	25444259	124727	203.9996	🟡
South Korea	37219405.92	185984	200.1215	🟡
Syria	45408502.07	223059	203.5717	🟡

Average Sales KPI

6. The sheet has been saved and named `Average Sales KPI`. Using this, the end user will find out the status of the average sales and can use the different dimension tables in order to create the KPI.

> The user must know that, without measures, they can't create the KPI. Suppose the user clicks on the KPIs option, a message will be shown saying **Users do not have calculated column fields. User cannot create the KPIs**. As has been discussed previously, the KPI is based on metrics. So, please understand that, without the measures, the user can't create the KPIs.

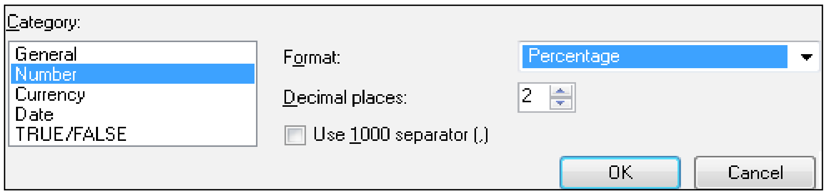
Creating a KPI using a Perspective

An idea about Perspectives, their history, and types of Perspectives have already been provided in the business scorecard section. The PowerPivot interface also provides the same features that were discussed previously. To put it simply, Perspective is kind of like the data mart concept. Data mart will have separate data models for each department from the primary data warehouse. The same can be done by creating the measures and KPIs using PowerPivot features. Specify the table attributes in order to use the created measures and KPIs for each analysis. Let's see how they are being created and, after that, the user must learn about the concepts of Perspectives. Here the KPIs will be created using another method.

In order to create a KPI using Perspectives, perform the following steps:

1. In this analysis, a comparison of the sales of the current and previous years will be done. Hence, the necessary measures have to be taken in order to find the previous year's sales.
2. In the Sales Analysis PowerPivot Workbook, switch to **Data View** and select the `FactSales` table at the bottom of the window; the user will find the practice area (if it doesn't show, click on **Calculation Area** at the top-right part of the ribbon.) Here, the user can place the measure using a DAX expression. Previously, the **Average Sales** measures have been created and stored in the `FactSales` table. The user will find **Average Sales** in the **FactSales** practice area beneath the `SalesKey` attribute column. Now, the other three measures will be created directly for `FactSales`.
3. The first measure is to find out the sales values of **Online store**; for that, the user can use the DAX expression `OnlineSales:=CALCULATE(SUM(FactSales[SalesAmount]), DimChannel[ChannelName]="Online").` Using this expression, the user can easily filter only the `Online` data in the `ChannelName` attribute from the `DimChannel` table and calculate the sum of the sales amount from the `FactSales` table. The cell under **Average Sales** in the **FactSales** practice area beneath the `SalesKey` attribute column will be selected; paste the DAX expression into the formula bar. The user must notice here that the DAX expression starts from the `=` sign, but, in this code, `OnlineSales:` has been used before the `=` sign. In the practice area, when the measure is created, the measure caption has to be specified; for this `OnlineSales:` has been used and it will work only in the practice area and not in the PowerPivot workbook tables. Now, the user will be able to see the `OnlineSales` measure in the **FactSales** practice area column.

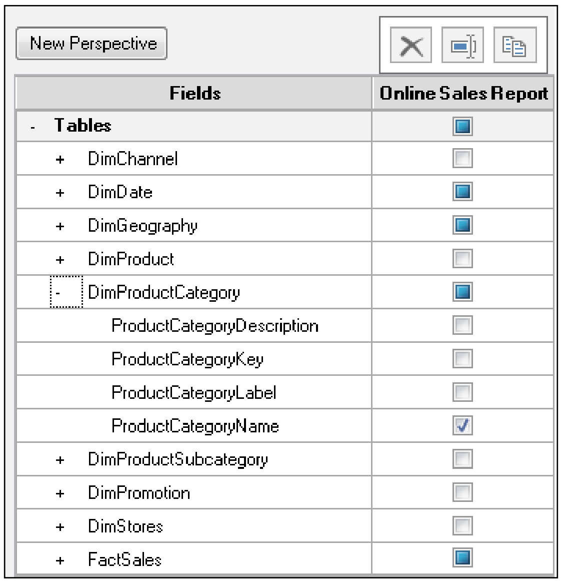
4. The second measure is to view the sales of the online store for the previous year and, for that, the user can use the DAX expression `OnlineSalesPreviousYear:=CALCULATE ([OnlineSales], DATEADD (DimDate[Datekey], -1, YEAR))`. The cell under **OnlineSales** in the **FactSales** practice area beneath the `SalesKey` attribute column will be selected; paste the DAX expression into the formula bar. This formula calculates the previous year's sales. Now, the user will be able to see the `OnlineSalesPreviousYear` measure in the **FactSales** practice area column.
5. The third measure is to calculate the year-over-year growth and, for that, the user can use the DAX expression `YOYGrowth:=([OnlineSales] - [OnlineSalesPreviousYear]) / [OnlineSalesPreviousYear]`. The cell under `OnlineSalesPreviousYear` in the FactSales practice area beneath the `SalesKey` attribute column will be selected; paste the DAX expression into the formula bar. This formula calculates the year-over-year growth. Now, the user will be able to view the year-over-year growth measure in the **FactSales** practice area column.
6. The three measures have been successfully created and the format of the created measures has been changed. Right-click on **OnlineSales** and then select **Format**. In the formatting dialog box, select **Currency** as the category, the symbol as **Euro**, and **2** as the number of decimals; then click on **OK**. Format the `OnlineSalesPreviousYear` measure and, for the `YOYGrowth` measure, choose **Percentage**, as shown in the following screenshot:


Category: Number
Format: Percentage
Decimal places: 2
Use 1000 separator (,): not checked

Formatting measure

7. Create the KPI using the `YOYGrowth` measure. Right-click on **YOYGrowth** and then click on Create KPI, the **Key Performance Indicator** (**KPI**) dialog box will be launched and the user can define the status threshold. Assign the absolute value as 0, threshold value as -0.05, and the high value as 0.05. The threshold value -0.05 indicates 5% negative growth and a high value 0.05 indicates 5% positive growth. Select the traffic light icon style from the list of available icon styles. Users can describe the threshold value by clicking on the description option underneath the style icons.

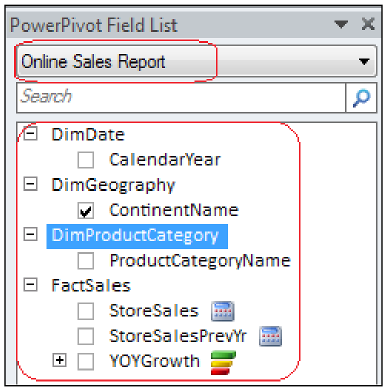
8. In order to apply our measure and KPI, the Perspective has to be created. In the PowerPivot window, navigate to the **Advanced** tab and, if it is not shown in the user's PowerPivot Workbook, by clicking on the **File** button and the **Switch to Advance Mode** option at the bottom of list, the user will be able to get the **Advanced** tab.
9. Now, by clicking on the **Advanced** tab, the user will find the **Perspective** option, and, by clicking on it, the **Perspectives** dialog box will appear. Click on the **New Perspective** button and, in the **Field** option, name it `Online Sales Report`. The user will be able to find all the tables of the `Sales Analysis` PowerPivot Workbook and, by expanding the table, the user will be able to find the attributes of the tables.
10. Now, select `CalendarYear` from the `DimDate` table, select `ProductCategoryName` from the `DimProductCategory` table, select `ContinentName` from the `DimGeography` table, expand the `FactSales` table, and select the measures and KPIs that have been created, namely `OnlineSales`, `OnlineSalesPreviousYear`, and `YOYGrowth`, and click on the **OK** button; the **Online Sales Report** will be created and, if the user wants to edit the created Perspective, it can be done by clicking on the **Perspective** option.



Fields	Online Sales Report
- Tables	■
+ DimChannel	☐
+ DimDate	■
+ DimGeography	■
+ DimProduct	☐
- DimProductCategory	■
ProductCategoryDescription	☐
ProductCategoryKey	☐
ProductCategoryLabel	☐
ProductCategoryName	✓
+ DimProductSubcategory	☐
+ DimPromotion	☐
+ DimStores	☐
+ FactSales	■

Perspectives dialog box

11. Here, the KPI is going to be created using the newly created Perspective; a new PivotTable layout will be added into the existing workbook.
12. At the top of **PowerPivot Field List**, the user will be able to find the default Perspective, select **Online Sales Report**, which is the one that was created. Once selected, the user will be able to find only the selected table and the attribute that was selected in the **Online Sales Report** Perspective; they won't find all the tables of the **Sales Analysis** PowerPivot Workbook. So, if the KPI was created based on the Perspective, the end user won't get confused and the developer can create the KPI based on different departments and use necessary attributes; they can also intimate to the end user as to which Perspective needs to be used for a certain KPI and so on.

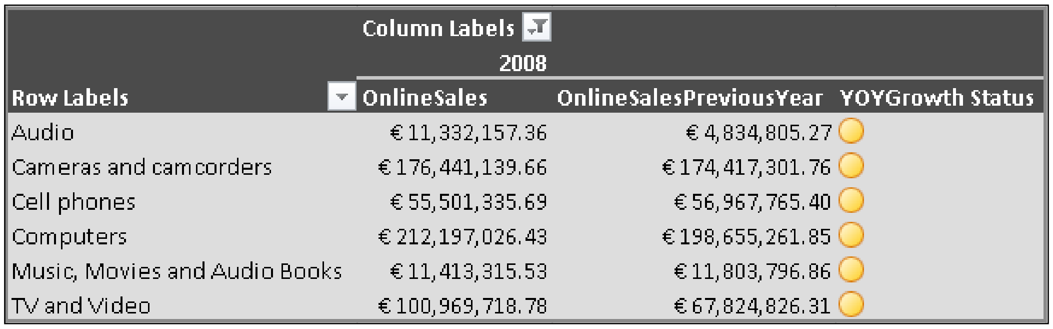


Online Sales Report

- DimDate
  - [ ] CalendarYear
- DimGeography
  - [✓] ContinentName
- DimProductCategory
  - [ ] ProductCategoryName
- FactSales
  - [ ] StoreSales
  - [ ] StoreSalesPrevYr
  - [ ] YOYGrowth

Perspectives option in PowerPivot Field List

13. Now, the KPI will be created using the `Online Sales Report` Perspective. Drag `ProductCategoryName` from the `DimProductCategory` table into the **Row Labels** section, drag `CalendarYear` from the `DimDate` table into the **Column Labels** section, and drag the `OnlineSalesandOnlineSalesPreviousYear` measures from the `FactSales` table into the ∑ **Values** box. Now, the user will be able to see the PivotTable sales of the current and previous years; filter only **2008** and expand the `FactSales` table from **PowerPivot Field List**. There, the user will find `YOYGrowth`, which checks the **Status option**. In the PivotTable, the status will be as shown in the following screenshot:



	Column Labels		
	2008		
Row Labels	OnlineSales	OnlineSalesPreviousYear	YOYGrowth Status
Audio	€ 11,332,157.36	€ 4,834,805.27	●
Cameras and camcorders	€ 176,441,139.66	€ 174,417,301.76	●
Cell phones	€ 55,501,335.69	€ 56,967,765.40	●
Computers	€ 212,197,026.43	€ 198,655,261.85	●
Music, Movies and Audio Books	€ 11,413,315.53	€ 11,803,796.86	●
TV and Video	€ 100,969,718.78	€ 67,824,826.31	●

Online Sales report KPI

14. The KPI has been successfully created using the concepts of Perspective and the sheet has been renamed as `Online Sales Report KPI`. Now the user will be familiar with the features of Perspective.

Summary

This chapter gives a great idea about business intelligence solution's analysis; the user must know the general concepts of creating a BI report, how to perform analysis based on the data and importance of the reports, know how a variety of data sources will be on one single screen as the Dashboard report. Now, the user will have knowledge of the business scorecard, its benefits, and a general idea about Perspective, the types of Perspectives, and so on.

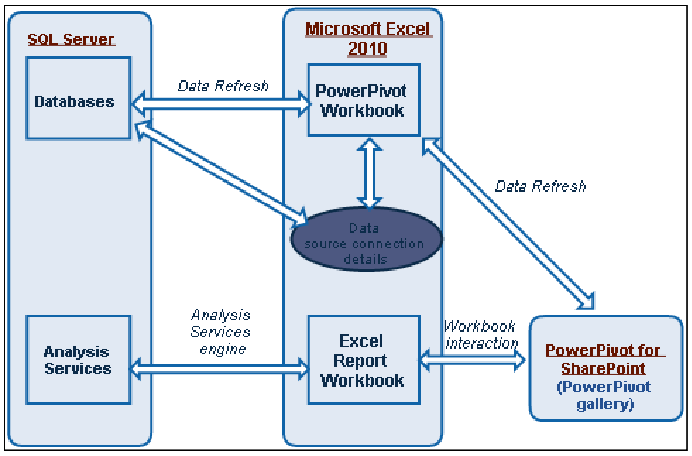
After explaining the general overview of BI reports, several reports have been created in the PowerPivot interface using our two types of industrial data. The user must have a good idea about how to create a variety of reports and the properties of all the report functionalities and should be able to give a brief explanation, tips, and so on. The vital aspect is that the user would understand features of a Perspective after creating the KPI using a Perspective, understanding the measures, and so on. Finally, the user will have a great idea about creating an advanced report using the PowerPivot interface.

In the next chapter, various options on how to publish and schedule our created PowerPivot report in the Share point server will be discussed and the user will have a better idea about the deployment of the PowerPivot reports.

5
Publishing Reports Using the SharePoint Server

After users create the PowerPivot Workbook, they can share it with their colleagues or some other important people. For this kind of process, Microsoft has a feature called PowerPivot for SharePoint, which shows the PowerPivot gallery in the SharePoint site if the user enables this feature. The people who are going to see the report need the permissions and URL in order to enter into the PowerPivot gallery. Herein, they can modify the file, interact with reports, and slice and filter the data in the browser mode. Everything that is done in the Excel workbook to create a report can also be done in the browser mode of PowerPivot through the SharePoint Server. The user can use PowerPivot in the SharePoint Server as a data source for another PowerPivot Workbook, and it is not necessary to install PowerPivot or even Excel on your local machine if you are using PowerPivot through SharePoint.

In this chapter, some typical explanations about PowerPivot for SharePoint, how to configure PowerPivot for SharePoint, and how to publish, schedule, give notifications, and refresh the data created in a PowerPivot report in a SharePoint Server will be explained. By the end of this chapter, the user will have a good idea about the deployment of the PowerPivot reports.



PowerPivot for SharePoint architecture

The preceding diagram illustrates how the PowerPivot report will interact with the SharePoint Server and all other features. By the end of this chapter, the user will understand the entire life cycle of PowerPivot for SharePoint. However, after the user understands this architecture, it will be very easy for him/her to understand other things.

PowerPivot for SharePoint

PowerPivot for SharePoint is a collection of server components through which query processing and management control over PowerPivot workbooks can be done. Later, these workbooks are published through the SharePoint server. This package comes under the Enterprise version of the SQL Server setup file. This supports the Excel workbook that contains the PowerPivot data model and enables the user to share the data model and reports and provides the PowerPivot gallery library; this is one of the special types of the SharePoint document library. This library will be created at a point where PowerPivot for SharePoint is configured in the SharePoint Server website.

This library provides all the deployment features of reports and a preview of reports, PowerPivot workbooks, and enables the users to create ad hoc reports based on PowerPivot workbooks. Users will enjoy the interactive reports by using the SharePoint website. Users must know that this SharePoint Server is a web browser based application, and once the users configure it, they are supposed to open the application only by the web browser. Hence, a user can access the PowerPivot gallery through the URL that the user receives after configuring PowerPivot in the SharePoint server.

PowerPivot for the SharePoint site relies on security, which is very important for the user. The SharePoint administrators have to give permissions to people who can publish reports based on the workbook, and there are multiple permissions in order to secure the PowerPivot gallery, such as read-only permissions, read/write permissions, and export and import permissions. These kinds of permissions meet the security and compliance policies.

More precisely, the users who are working in a big enterprise with serial clients are supposed to have the PowerPivot for SharePoint site in order to publish and share reports for their clients who are at different global locations. In order to apply this feature, the user must contact the SharePoint administrator. The administrator will know how to configure it, as it's quite difficult, and once it is configured, the user can easily deploy a report. There is no need to know any programming languages as it mostly involves a GUI, which the user will understand very easily.

The important thing here is that all of this is not necessary for small enterprises, self-service business intelligence (Personal BI), and for those who are not going to share their reports and data model. This is because in order to configure these things, the user must have power computers such as Windows Sever 2008, SharePoint Server 2010 Enterprise Edition, and SQL Server 2008 R2 Enterprise Edition, and the user must also know that the BI feature is enabled only in the Enterprise version. This means that the user might have to spend more money on these things. It will also make the systems very slow with the necessary heavy hardware and software requirements. Hence, it's not required at a low level, but the user must know the features since it will help them in the future.

Let us consider one small e-commerce company, which is using PowerPivot for creating its analysis and reports. The developer wants to share his reports with his colleagues, which means that for different departments to know the status of their websites, the developer will need all the requirements, software, and permissions from the head of IT. If the head of IT doesn't provide permissions as his negotiation is supposed to happen if we configure PowerPivot for SharePoint, a power computer, all the software requirements, and so on, which are needed, but aren't available, this task cannot be carried out since it involves a SharePoint server. The developer will have to find some other way. The developer can create reports in an Excel workbook, and instead of creating every report in one single workbook, they need to create reports in different workbooks based on the departments, and those reports have to be shared locally and via e-mails.

Configuring PowerPivot for SharePoint

PowerPivot for SharePoint is configured either in a new SharePoint server, an existing SharePoint server, or a multiserver SharePoint farm that depends on which types of SharePoint servers have been installed on a system.

For instance, if users are using a SharePoint Server for another reporting purpose, they can configure PowerPivot for SharePoint on an existing SharePoint server. And a vital thing here is the prerequisites. So, the users should be very careful with the hardware and software requirements.

Some of the typical and minimum hardware and software requirements are as follows:

- 64-bit dual-core processor, 3 GHz
- 8 gigabytes of RAM
- 80 gigabyte storage
- Windows Server 2008 SP2 or R2 (64-bit) operating system
- SharePoint 2010 Enterprise Edition
- SQL Server 2008 R2 Enterprise Edition

If users are using Microsoft Excel 2013 PowerPivot, they are supposed to have SharePoint 2013 Enterprise Edition and SQL Server 2012 Enterprise Edition, as PowerPivot is based on SQL Server 2012.

The procedure to configure PowerPivot for SharePoint in a new SharePoint Server 2010 is given here along with some brief explanations, as this portion will be handled by the SharePoint administrator. However, the user must have basic knowledge about this. If users want to know more or they want to install it on their own, they can use the following link:

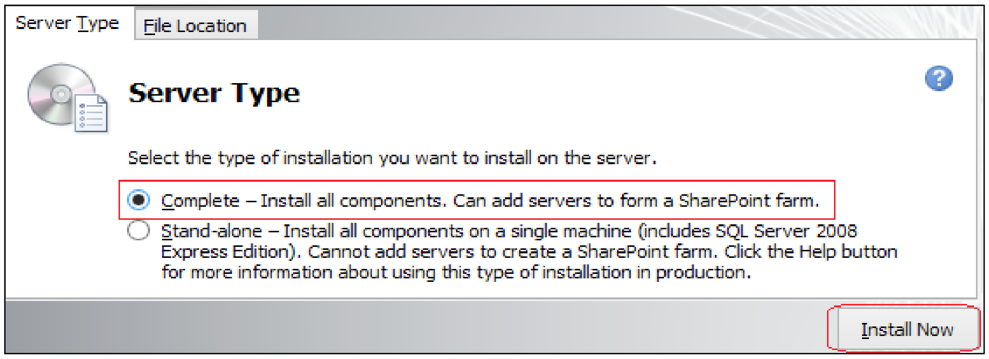
`http://technet.microsoft.com/en-us/library/ee210650(v=sql.105).aspx`

Here, the user will get adequate information about configuring PowerPivot for SharePoint. The left-hand side will have the contents under **Deployment** (PowerPivot for SharePoint). By expanding this, the user will have all the information.

The first installation of the SharePoint Server 2010 will be shown here with Unconfigured, by performing the following steps:

1. Open the folder that contains the setup files for SharePoint 2010, and find the **Prerequisite Installer** application type file; just click on it and it will install all the prerequisites; only the prerequisite software that is not present in the system will be installed. If it's already installed, it won't install any unnecessary software. Hence, the user can install this in order to avoid any problems.
2. Now run the setup file in order to install the SharePoint Server software; it will ask for the **Product Key** details; enter the key and click on **Accept the Microsoft Software License Terms of agreement**, and then click on **Continue**.
3. Next, the user will have an option to select an installation type (such as **Server Farm** or **Standalone**). The **Server Farm** option must be selected as **Standalone** deployments won't let the user share the reports with others, and hence this option doesn't meet the goal of sharing the PowerPivot workbooks with others. After selecting the **Server Farm** option, the **Claims** infrastructure should be added so that the administrative and data connections are supported.

4. After selecting the **Server Type** option, the user will receive the **Server Type** and **File Location** screen, as shown in the following screenshot, to include all of the SharePoint Server features required for both application server and web frontend roles on the same server. Select the **Complete** option in the **Server Type** tab. In the **File Location** page, the user can change the location where they want to install the SharePoint server.



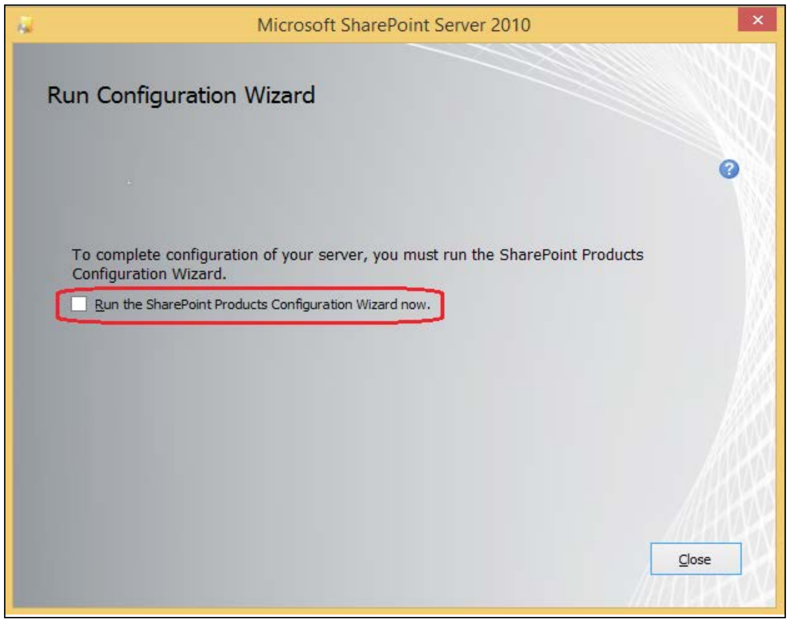
Select the type of installation you want to install on the server.

[✓ Complete – Install all components. Can add servers to form a SharePoint farm.]

Stand-alone – Install all components on a single machine (includes SQL Server 2008 Express Edition). Cannot add servers to create a SharePoint farm. Click the Help button for more information about using this type of installation in production.

The Server Type and File Location screen

5. Now click on the **Install Now** button; the SharePoint installation will begin and it will take some time. Once the installation is completed, **Run Configuration Wizard** will be shown, in which we need to uncheck the **Run the SharePoint Products and Technologies Configuration Wizard now** option and click on **Close**. If the user runs the configuration wizard now, he/she will get an error because the database server is not yet installed. When the database server is installed by using the SQL Server 2008 R2 Enterprise Edition setup, the server configuration can be done, and this is performed in a subsequent step.



To complete configuration of your server, you must run the SharePoint Products Configuration Wizard.

Run the SharePoint Products Configuration Wizard now.

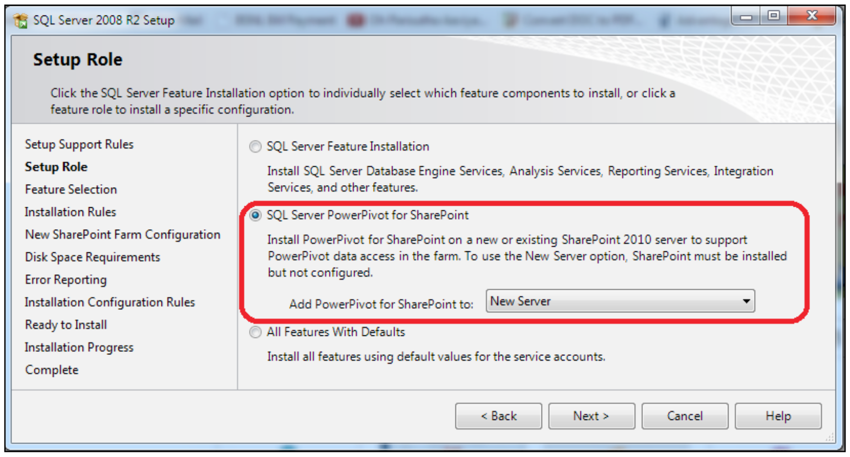
Run Configuration Wizard

6. Now that the SharePoint Server is installed, by clicking on the **Start** menu the user will be able to see **SharePoint 2010 Products Configuration Wizard** and the **SharePoint 2010 Central Administration** options.

By using the SQL Server 2008 R2 setup, the configuration of SharePoint for PowerPivot will be done. PowerPivot for SharePoint and a database engine instance can be configured and installed here by a user. The SharePoint Server will use this instance as a database server. This can be done by performing the following steps:

1. Open the SQL Server 2008 R2 Enterprise Edition setup file, right-click on `Setup.exe`, and select **Run** as administrator. From the left side of the navigation pane, select **Installation** and then click on **New installation** or add features to an existing installation.
2. In **Setup Support Rules**, click on **OK**. Enter a product key, click on **Accept the Microsoft Software License Terms of agreement**, and then click on **Next**. In the **Setup Support Files** page, click on **Install**.

3. Next, the user will be taken to the **Setup Role** screen. Here, the user will be able to see three kinds of installations in which we need to check **SQL Server PowerPivot for SharePoint**, and in the **Add PowerPivot for SharePoint to** combo box, choose **New Server** as we are going to configure PowerPivot for SharePoint in a new SharePoint server. Click on **Next**.


SQL Server 2008 R2 Setup

Setup Role

Click the SQL Server Feature Installation option to individually select which feature components to install, or click a feature role to install a specific configuration.

SQL Server Feature Installation

Install SQL Server Database Engine Services, Analysis Services, Reporting Services, Integration Services, and other features.

[✓ SQL Server PowerPivot for SharePoint]

Install PowerPivot for SharePoint on a new or existing SharePoint 2010 server to support PowerPivot data access in the farm. To use the New Server option, SharePoint must be installed but not configured.

Add PowerPivot for SharePoint to: New Server

All Features With Defaults

Install all features using default values for the service accounts.

SQL Server PowerPivot for SharePoint

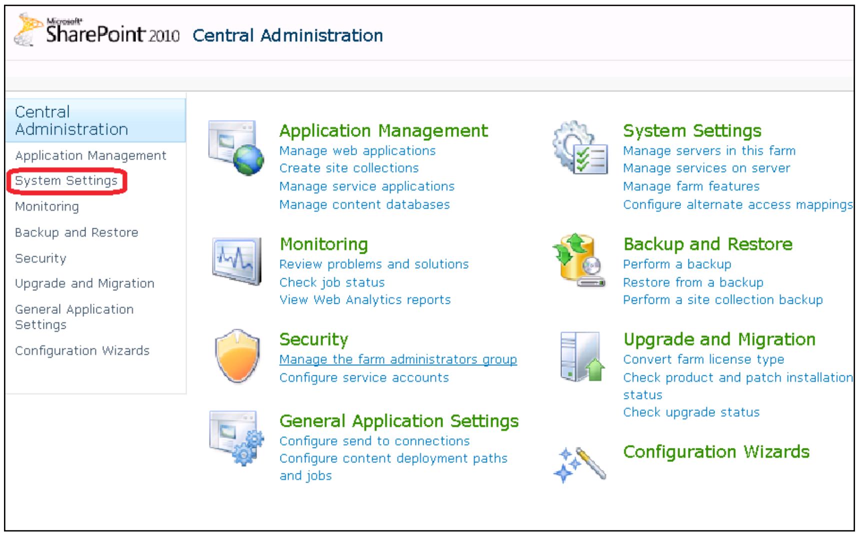
4. In the **Feature Selection** window, review and then click on **Next**; do the same for the **Installation Rules and Instance Configuration** window.
5. In the **New SharePoint Farm Configuration** window, enter a domain user account for the server farm account. For simplicity, the user can use the username and password of the account that is currently logged on (for example, `Packt\administrator Password: <user password> Pass phrase: <user pass phrase> Confirm: <user pass phrase>`) and **Port number**; if the user wants to change the port number, they can change it from `1` to `65535`. Otherwise, keep the port number generated by the system, then click on the **Next** button.
6. To install the features, make sure that the system has sufficient disk capacity. In the **Disk Space Requirements** window, click on **Next**.
7. In the **Server Configuration** window, specify the account credentials in order to run SQL Server Agent, SQL Server Database Engine, and SQL Server Analysis Services. The user can use the same account that the system is currently logged in with, for example `Packt\administrator`. Click on **Next**. This is a very important part, so for further information, the user can use this link: `http://technet.microsoft.com/en-us/library/ee210603(v=sql.105).aspx`.

8. In the **Database Engine Configuration** window, add the current user (click on the **Add Current User** button) and other users (as required, click on **Add...**) to the list of SQL Server's administrators. Review the **Data directories** tab. Click on the **Next** button. And do the same for the **Analysis Services Configuration** window, and then click on the **Add Current User** and **Next** buttons.
9. Until you get to the **Ready to install** page, keep clicking on **Next** for each of the remaining pages. Now click on **Install** and the user setup will progress. After the installation of everything is done, it will take 45-60 minutes for the user to get a message.

Verifying the installation

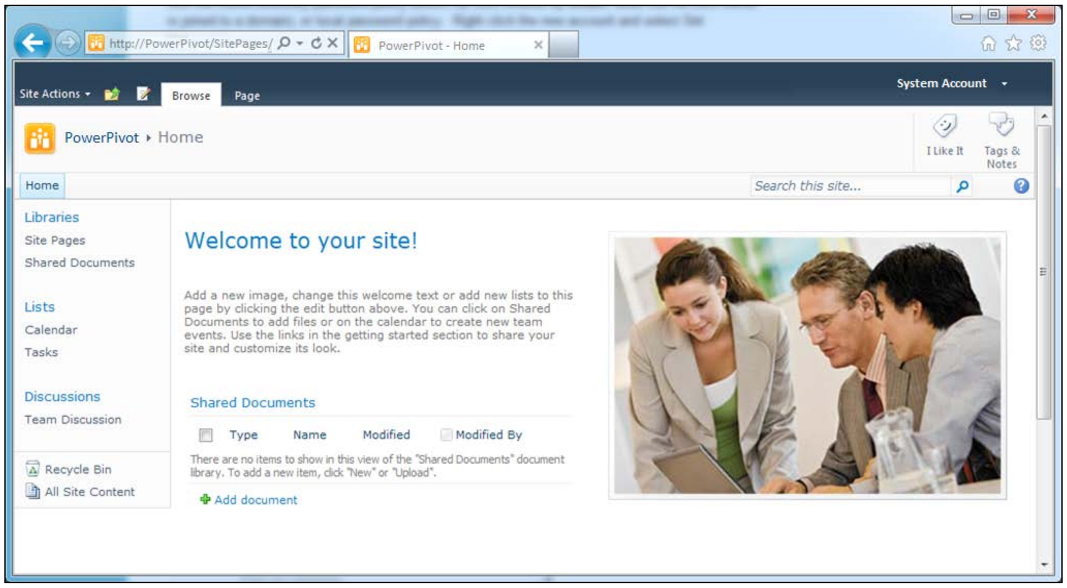
After finishing the installations, the features of PowerPivot for SharePoint that the user installed in a SharePoint Server farm are controlled via the SharePoint server's central administration. To test the PowerPivot integration with the central administration site, perform the following steps:

1. Navigate to **Start** | **All Programs** | **Microsoft SharePoint 2010 Products** | **SharePoint 2010 Central Administration**. Then input the username and password that were given during the installation, and then click on **OK**. The user will see the web page as shown in the following screenshot, and the user can change the browser settings to avoid inputting the username and password while opening the **Central Administration** page. It is advisable that the user makes these **Trusted Sites**.


Microsoft SharePoint 2010
Central Administration
Central Administration
Application Management
System Settings
Monitoring
Backup and Restore
Security
Upgrade and Migration
General Application Settings
Configuration Wizards
Application Management
Manage web applications
Create site collections
Manage service applications
Manage content databases
System Settings
Manage servers in this farm
Manage services on server
Manage farm features
Configure alternate access mappings
Monitoring
Review problems and solutions
Check job status
View Web Analytics reports
Backup and Restore
Perform a backup
Restore from a backup
Perform a site collection backup
Security
Manage the farm administrators group
Configure service accounts
Upgrade and Migration
Convert farm license type
Check product and patch installation status
Check upgrade status
General Application Settings
Configure send to connections
Configure content deployment paths and jobs
Configuration Wizards

SharePoint 2010 Central Administration

2. In the **Central Administration** page, click on the **System Settings** option, and click on **Manage farm features** from the options and confirm that the PowerPivot integration feature is active. In the same **System Settings** option, click on **Manage services on server** from the options and confirm that **SQL Server Analysis Services** and **SQL Server PowerPivot System Service** are started. Switch to the **Central Administration** option and click on **Manage service applications** in **Application Management**. In order to open the **PowerPivot Management** Dashboard for this application, click on **Default PowerPivot Service Application**. For more information, use the given link: `http://technet.microsoft.com/en-us/library/ee210685(v=sql.105).aspx`.
3. In order to verify the PowerPivot site integration in the SharePoint server, open Internet Explorer or your default browser and enter `http://<Your Computer Name>` (for example, `http://Packt`) in the URL field. It might take a few minutes to open the PowerPivot site and to respond for the first time; for systems with a low speed, it might take around 5 to 10 minutes. Once opened, the user will be able to see the following screenshot:



Add a new image, change this welcome text or add new lists to this page by clicking the edit button above. You can click on Shared Documents to add files or on the calendar to create new team events. Use the links in the getting started section to share your site and customize its look.

There are no items to show in this view of the "Shared Documents" document library. To add a new item, click "New" or "Upload".

The PowerPivot site

The user will be able to see **PowerPivot Gallery** under the **Libraries** section, which provides the PowerPivot features. By using these libraries, the user can publish their reports.

Types of users

Here, the user can find the users who will benefit from the business intelligence technology, and the important thing here is that in an organization, not everyone can see all the reports and analyses of the data, as organizational data is confidential (PowerPivot for SharePoint gives all the security features). So for this, only certain people are allowed to access reports. The list is as follows:

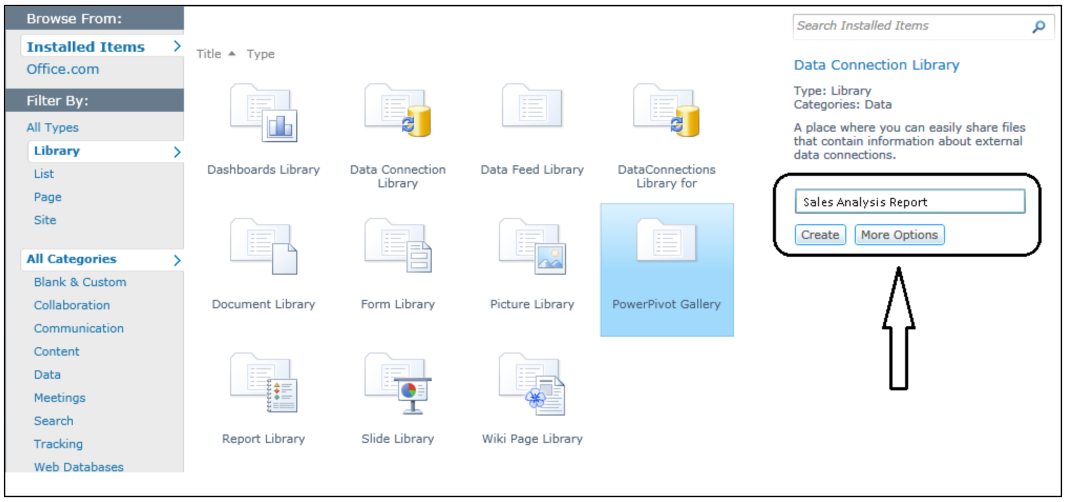
- Executives (generally the President/CEO and the Vice Presidents) can focus on the following analyses:
    - The performance of the entire business
    - They need the results of complex analyses quickly and in such a way that will help them make informed decisions
- Business decision makers (typically directors) can focus on the following analysis:
    - Focus on vertical slices of the business (finance, HR, manufacture, and so on)
- Information workers (typically middle managers) can focus on the following analyses:
    - Managers and non-managers who need to perform some level of analysis
    - They need to produce reports by pulling data together from multiple sources and providing results and recommendations
    - They need powerful tools to explore data and construct complex models, often using advanced mathematics and statistics
- Analysts (special kinds of information workers) can focus on the following analyses:
    - Focus on performing in-depth, detailed analysis of data
    - They need powerful tools to explore data and construct complex models, often using advanced mathematics and statistics
- Line workers can focus on the following analysis:
    - Do not analyse data but BI may be a part of the tools they already use
    - They may use BI as a part of their tasks (for example, if a sales clerk checks out a customer, the system might make suggestions on what to recommend as additional purchases)

Deploying the reports in a PowerPivot site

PowerPivot for SharePoint supports the Microsoft Excel 2010 workbook, which contains the PowerPivot data model. The users can import this Excel workbook into a PowerPivot gallery in order to publish their reports. As a result, PowerPivot for SharePoint is necessary for the end users as they can view and interact with the reports; also, it's not necessary to install it into every end user's system where the PowerPivot add-in is installed. For these things, the Analysis service farm process in the SharePoint Server takes the workbook, and the memory is used by the analysis service; also, it helps in data aggregation, data processing, and some other features. If a user refers to the previous screenshot, which is the PowerPivot for SharePoint architecture, he/she will find a connection between the Excel workbook and analysis service by the analysis service PowerPivot engine. PowerPivot for SharePoint works based on these concepts.

Importation of the `Sales Analysis.xlsx` workbook into the PowerPivot gallery will be done in the following steps:

1. In the PowerPivot web page, click on the **Click Site Actions** option and then select **More Options**. Click on **Library** and then select **PowerPivot Gallery**. There will be a box to the right of this option, which is used for creating any gallery under the PowerPivot gallery. The `Sales Analysis Report` gallery has been created in order to import all the sales analysis reports that were created previously.



Data Connection Library

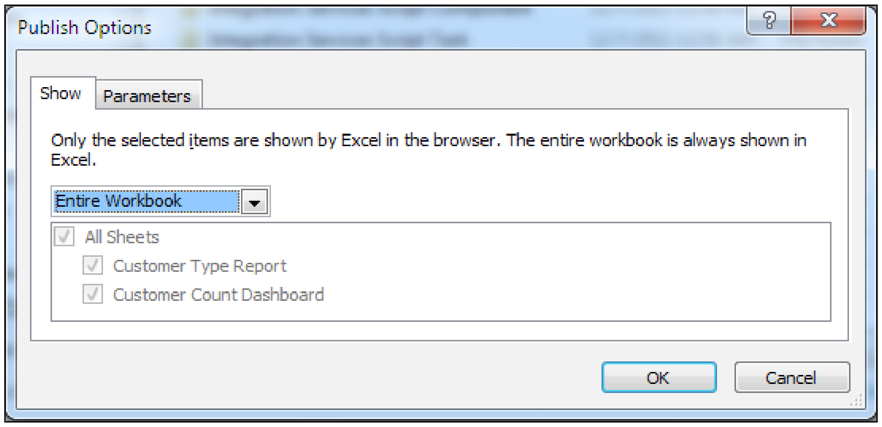
Type: Library
Categories: Data

A place where you can easily share files that contain information about external data connections.

Sales Analysis Report

Create the Sales Analysis Report gallery

2. Add the **Sales Analysis.xlsx** sheet into `Sales Analysis Report`, and if the user is creating more workbook reports for sales analysis, the user can add multiple workbooks under one gallery. Now the user will be able to see the workbook by clicking on the file. It will enter into the workbook that was added, and the user will be able to see the reports in the PowerPivot site and also interact with the report by using a slicer or filtering.
3. There is another way to import the workbook into a PowerPivot gallery. Open the **Ecommerce** Excel workbook, navigate to **File** | **Save & Send** | **Save to SharePoint** and select the PowerPivot gallery location, and click on **Save As**. Now, the user will be able to see the **Save As** dialog box; click on the **Publish option** button. Here, the users can select the workbook according to their wishes, such as an entire book, sheets option, or an item in the workbook. Check the **Open with Excel in the browser** option in order to open the report in the PowerPivot gallery after saving. Now click on the **Save** button; the selected workbook will be opened in the PowerPivot site.



Only the selected items are shown by Excel in the browser. The entire workbook is always shown in Excel.

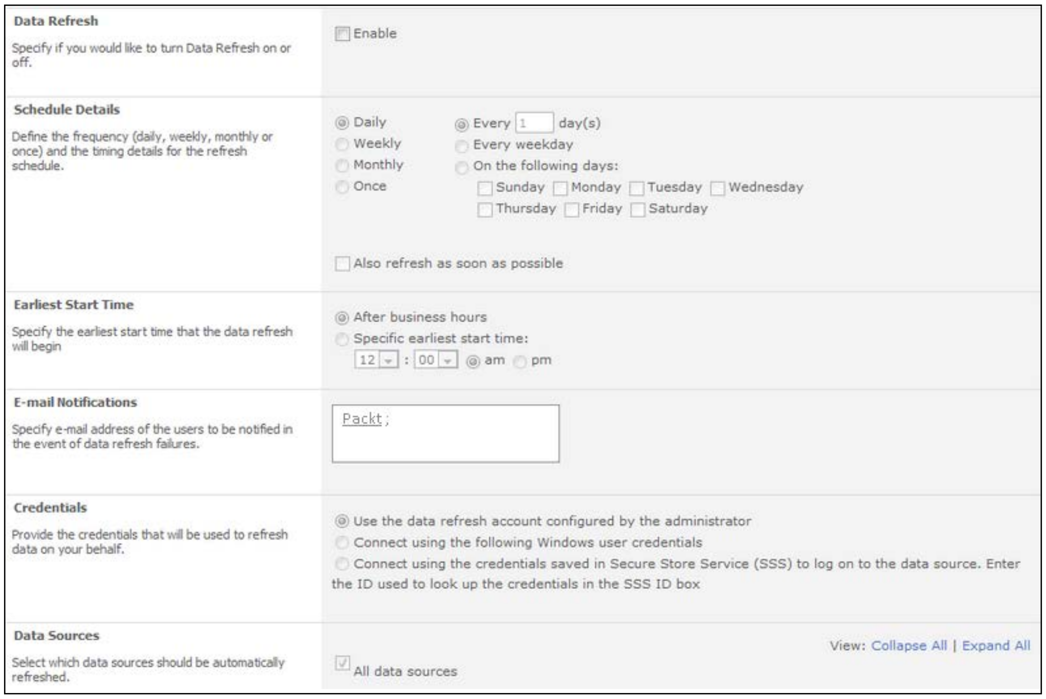
Entire Workbook

✓ All Sheets
✓ Customer Type Report
✓ Customer Count Dashboard

The Publish Options dialog box

4. And if the users want to change the gallery view, select the **Sales Analysis Report** gallery, click on the drop-down list under **Current View** from the **Manage Views** section, and choose one gallery view. In this instance, we have chosen the **Theater** view.

5. In order to schedule and refresh the data, select the workbook where refreshing needs to be done and select **Manage Data Refresh** from the drop-down list that is next to the workbook name. The user will see the data refresh document properties dialog box as shown in the following screenshot. By using this, the user can schedule and refresh the data.


Data Refresh
Specify if you would like to turn Data Refresh on or off.
☐ Enable

Schedule Details
Define the frequency (daily, weekly, monthly or once) and the timing details for the refresh schedule.
[✓ Daily] Weekly Monthly Once
[✓ Every 1 day(s)] Every weekday On the following days:
☐ Sunday ☐ Monday ☐ Tuesday ☐ Wednesday ☐ Thursday ☐ Friday ☐ Saturday
☐ Also refresh as soon as possible

Earliest Start Time
Specify the earliest start time that the data refresh will begin
[✓ After business hours] Specific earliest start time:
12 : 00 [✓ am] pm

E-mail Notifications
Specify e-mail address of the users to be notified in the event of data refresh failures.
Packt ;

Credentials
Provide the credentials that will be used to refresh data on your behalf.
[✓ Use the data refresh account configured by the administrator]
Connect using the following Windows user credentials
Connect using the credentials saved in Secure Store Service (SSS) to log on to the data source. Enter the ID used to look up the credentials in the SSS ID box

Data Sources
Select which data sources should be automatically refreshed.
View: Collapse All | Expand All
✓ All data sources

Data refresh document properties dialog box

6. Another great feature in PowerPivot for SharePoint is that the user can create a new report from an existing PowerPivot Workbook data (as re-using) that was uploaded into the PowerPivot gallery. In the PowerPivot gallery, select one of the workbooks that was uploaded into the PowerPivot gallery. Click on the New Report icon besides **Manage Data Refresh**, and click on the **Open New Excel** workbook and open the Excel workbook. Here, the PowerPivot add-in brings the data that was stored in the workbook. By using this, the user can create reports.

Summary

This chapter shows the user how to publish and schedule the created PowerPivot reports in the SharePoint server, and the user will also have some idea about the deployment of the PowerPivot reports.

By the end of this chapter, the user will know the PowerPivot for SharePoint features and how they work. Some typical information about the hardware and software requirements and installation has also been provided to increase the user's understanding of PowerPivot.

Index