Chapter 1

Introducing Business Intelligence Semantic Model

1.1 What is Business Intelligence Semantic Model? 1
1.2 Understanding Tabular 11
1.3 When to Use Tabular 15
1.4 Comparing Tabular with Other Models 22
1.5 Summary 28

In the digital and fast-paced world of today, technologies undergo monumental shifts to help organizations streamline their processes and transact business in more efficient ways. Information technology, for example, has witnessed these changes with the Internet boom, cloud computing, and mobile devices. In the business intelligence (BI) space, the arrival of Microsoft Business Intelligence Semantic Model (BISM) in Microsoft SQL Server 2012 is a paradigm shift because it allows you to implement a new class of applications for personal, team, and organizational business intelligence (BI) solutions. BISM makes BI pervasive and accessible to both business users and BI professionals. If you're a business user, you can use BISM to import massive amounts of data and to build personal BI solutions without relying too much on your IT department. And, if you're a BI professional, you'll find that BISM empowers you to implement both relational and multidimensional solutions on a single platform.

This guide discusses the tabular capabilities of BISM, and this chapter is the most important chapter of the book. I'll start by introducing you to BISM, explaining how it fits into the Microsoft BI stack and when to use it. If you're familiar with Analysis Services cubes, you'll learn how the Multidimensional and Tabular paths compare with each other. Then, I'll take you on a tour of the BISM features and tools. I'll help you understand the product architecture and programming interfaces so that you have the necessary technical background to tackle more advanced topics later on in this book.

1.1 What is Business Intelligence Semantic Model?

Before I explain what Business Intelligence Semantic Model (BISM) is, I'll clarify what business intelligence (BI) is. You'll probably be surprised to learn that even BI professionals disagree about its definition. In fact, Forester Research offers two definitions (see http://bit.ly/foresterbi).

DEFINITION Broadly defined, BI is a set of methodologies, processes, architectures, and technologies that transform raw data into meaningful and useful information that's used to enable more effective strategic, tactical, and operational insights and decision-making. A narrower definition of BI might refer to just the top layers of the BI architectural stack, such as reporting, analytics, and dashboards.

In the world of Microsoft BI, BISM represents a semantic layer at the top of the BI architectural stack. In general, semantics relates to discovering the meaning of the message behind the words. In the context of data and BI, semantics represents the user's perspective of data: how the end

user views the data to derive knowledge from it. As a modeler, your job is to translate the machine-friendly database structures and terminology into a user-friendly semantic layer that describes the business problems to be solved. To address this need, you create a Business Intelligence Semantic Model (BISM).

Microsoft BISM is a unifying name for both multidimensional (OLAP) and tabular (relational) features in the Microsoft SQL Server 2012 release of Analysis Services. Since its first release in 1998, Analysis Services has provided Online Analytical Processing (OLAP) capabilities so that IT professionals can implement multidimensional OLAP cubes for slicing and dicing data. The OLAP side of Analysis Services is now referred to as *Multidimensional*. SQL Server 2012 expands the Analysis Services capabilities by adding a new path for implementing analytical models where entities are represented as relational constructs, such as two-dimensional tables, columns, and relationships. I'll refer to this new path as *Tabular*.

DEFINITION Analysis Services is a unified platform for implementing data analytical models for personal, team, and organizational BI solutions. BISM is a semantic layer that supports two paths for implementing analytical models: Multidimensional for OLAP cubes and Tabular for relational-like models.

Several terms in this definition might be unfamiliar to you, so let's take a closer look. To start with, Analysis Services is a platform that uses BISM as the metadata definition and semantic layer. Analysis Services provides tools to help you develop custom BI solutions for the full spectrum of BI needs:

- Personal BI (or self-service BI) – Personal BI enables business users to offload effort from IT pros and to build BI models for self-service data exploration and reporting. Suppose that Maya from the sales department wants to analyze some sales data that's stored in a Microsoft Access database or in an Excel workbook. With a few clicks Maya can import the data into a PowerPivot model, build pivot reports, and gain valuable insights.
- Team BI – Business users can share the reports and dashboards they've implemented with other team members without requiring them to install modeling or reporting tools. Suppose that Maya would like to share her sales analysis application with her coworker, Martin. Once Maya has uploaded the PowerPivot model to SharePoint, Martin can view online the pivot reports Maya has created, or he can author his own reports that connect to Maya's model.
- Organizational BI (or corporate BI) – BI professionals can enhance the tabular models developed by business users and deploy them to a dedicated server for added performance and security. Or, BI professionals can implement multidimensional cubes for data warehousing and OLAP. Continuing the previous example, suppose that over time the sales database has grown to millions of rows. Moreover, Maya's sales model has grown in popularity, and more coworkers are requesting access to it. Maya can now hand over the model to Bob, who is a BI pro. Bob can upgrade the model to a scalable and secure model that coexists with the multidimensional cubes that Bob has created for historical and trend analysis.

As you could imagine, Analysis Services is a versatile BI platform that enables different groups of users to implement a wide range of BI solutions. Writing a single book that covers in detail both the multidimensional and tabular aspects of BISM would be an impossible task. I discussed the OLAP capabilities of Analysis Services in my book *Applied Analysis Services 2005* (Prologika Press, 2005). This book covers Tabular only. However, if you're familiar with OLAP cubes, then you'll find plenty of information in this chapter and later chapters in order to understand Tabular from an OLAP point of view. Then you can decide which path is a better choice for the task at hand.

1.1.1 Understanding Analysis Services History

BISM is a component of Microsoft SQL Server Analysis Services. Before we get into the BISM technical details and capabilities, let's step back for a moment and review the Analysis Services history to understand its genesis and evolution. **Figure 1.1** shows the major product releases.

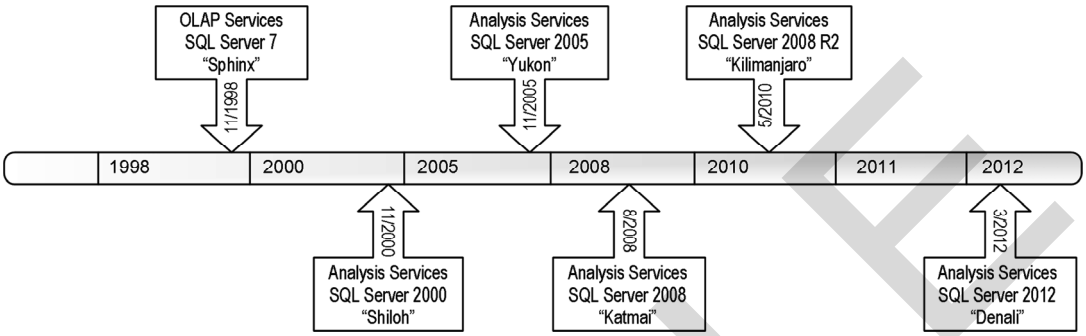

Figure 1.1 The SQL Server Analysis Services history at a glance.

After acquiring the OLAP software technology from an Israeli-based company, Panorama Software, Microsoft bundled the first release with SQL Server 7 in 1998 under the name "OLAP Services". The product provided tools to building OLAP cubes and supported Multidimensional Expressions (MDX) as a query language. In 2000, the name changed to "Analysis Services" due to inclusion of data mining capabilities for trend discovery and data mining.

About UDM

In SQL Server 2005, Microsoft completely redesigned Analysis Services and introduced the next generation of OLAP and data mining technologies under the name Unified Dimensional Model (UDM). UDM departed from the classic OLAP space and provided more flexibility for modeling multidimensional cubes, as explained in the "Introduction to the Unified Dimensional Model (UDM)" article by Paul Sanders at http://bit.ly/udmintro. For example, instead of limiting data exploration across predefined hierarchies, such as Category ⇨ Subcategory ⇨ Product, UDM allows you to slice and dice data by each column (attribute), such as Color, Size, Price, and so on. UDM became very successful over time, and it made Analysis Services an industry-leading BI platform and the most popular OLAP engine according to the Gartner's Magic Quadrant for Business Intelligence Platforms (http://bit.ly/gartnerquadrant).

NOTE The terms "UDM" and "cube" are often used interchangeably. Although initially positioned as a unification of dimensional and relational reporting, UDM is an OLAP-based model with additional features that aren't found in classic OLAP, such as attribute hierarchies, relationships, key performance indicators (KPIs), perspectives, and so on. The physical manifestation of UDM is the Analysis Services cube. As a term, UDM is no longer used in SQL Server 2012, and it's succeeded by BISM.

SQL Server 2008 brought incremental changes to Analysis Services in the areas of performance and manageability. For example, a block computation mode was introduced to eliminate unnecessary aggregation calculations and to improve the performance of queries.

About BISM

SQL Server 2008 R2 added a new technology for personal and team BI called PowerPivot, whose data engine later became the foundation of Tabular. If you were going to compare PowerPivot to the world of database development, then PowerPivot is to Analysis Services as Microsoft Access is

to SQL Server. Extending the Excel capabilities, PowerPivot enabled business users to create their own BI applications. At PASS Summit 2010, Microsoft announced its BI roadmap for SQL Server 2012 and introduced BISM as a continuum of personal, team, and organizational BI on a single platform, as explained in the "Analysis Services – Roadmap for SQL Server "Denali" and Beyond" blog post by T.K. Anand (http://bit.ly/bismatpass). The plans for BISM were to extend the PowerPivot capabilities so BI professionals can build relational-like tabular models for organizational BI solutions.

Later at TechEd North America 2011, Microsoft rebranded BISM as an umbrella name for both multidimensional and tabular models. The TechEd presentation, "What's New in Microsoft SQL Server Code-Named "Denali" for SQL Server Analysis Services and PowerPivot" by T.K. Anand and Ashvini Sharma, unveiled the new positioning, followed by the blog post "Analysis Services – Vision & Roadmap Update" (http://bit.ly/bismatteched). The article states "By bringing the two data models together, we will provide a powerful yet flexible platform that can tackle the diverse needs of BI applications – needs such as advanced analytics, sophisticated business logic, professional developer tools, choice of end user tools, performance, scalability, ease of use, and time to solution."

One of the prevailing themes of SQL Server 2012 is breakthrough insights. SQL Server 2012 adds new BI capabilities for pervasive data discovery and analysis across the organization. And, BISM is the analytical layer that powers BI.

1.1.2 Understanding the Microsoft BI Platform

Analysis Services is not the only BI product that Microsoft provides. It is an integral part of the Microsoft BI platform that was initiated in early 2004 with the powerful promise to bring "BI to the masses." Microsoft subsequently extended the message to "BI to the masses, by the masses" to emphasize its commitment to make BI broadly accessible. Indeed, a few years after Microsoft got into the BI space, the BI landscape changed dramatically. Once a domain of cost-prohibitive and highly specialized tools, BI is now within the reach of every user and organization.

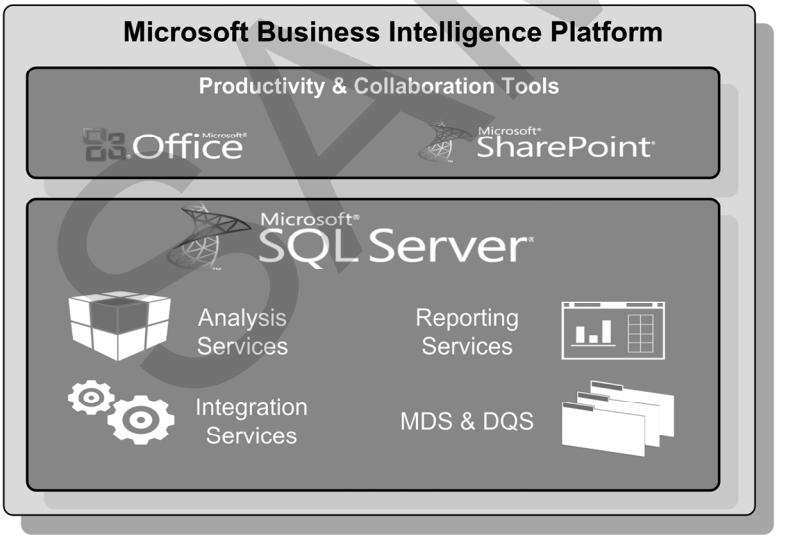

Figure 1.2 The Microsoft BI Platform provides services and tools that address various data analytics and management needs.

DEFINITION The Microsoft Business Intelligence Platform is a multi-product offering that addresses the most pressing data analytics and management needs that many organizations encounter every day.

Given Forester Research's broad definition of BI that I mentioned above, the Microsoft BI Platform includes tools that address the entire spectrum of BI needs. **Figure 1.2** illustrates the building blocks of the Microsoft BI Platform. Microsoft SQL Server forms the foundation of the Microsoft BI Platform and bundles services for structural data storage, data integration, reporting, and analysis.

Analysis Services

As I mentioned, Analysis Services provides OLAP and data mining services. Traditionally, organizations have used the OLAP capabilities of Analysis Services to implement multidimensional cubes on top of a data warehouse or data mart for trend and historical reporting. An Analysis Services cube can help business users analyze numeric data (measures) from different subject areas (dimensions). Coupled with a "smart" OLAP browser, such as Microsoft Excel, an Analysis Services cube could be a dream come true for every manager or decision maker. That's because analytical needs have been traditionally addressed by authoring operational reports, such as Sales by Product, Sales by Country, Sales by Year, and so on. Please don't misunderstand me. Operational reports do have their purpose. However, a multidimensional cube allows power users to author their own interactive reports without relying on IT pros.

REAL WORLD How much effort does your company spend on canned reports? One of our clients had an IT department with some 30 BI pros whose main responsibility was authoring operational reports from stored procedures. There was a significant schema overlap between the datasets returned by the stored procedures. Report results were inconsistent because business metrics were not shared between reports. Performance was an issue due to the extensive data crunching that the reports had to perform to aggregate large datasets from the company's data warehouse database. We addressed these challenges by implementing a multidimensional cube. While operational reports were still required to address external reporting needs, their number was significantly reduced, as well as the effort to prepare and verify the data.

Data mining is one of my favorite Analysis Services features. It also happens to be one of the least understood because it is usually confused with slicing and dicing data. To the contrary, data mining is about discovering patterns that are not easily discernible. These hidden patterns can't be discovered with traditional data exploration since data relationships might be too complex or because there is too much data for a human to analyze. Typical data mining tasks include forecasting, customer profiling, and basket analysis. Data mining can answer questions, such as "What are the forecasted sales numbers for the next few months?", "What other products is a customer likely to buy along with the product he or she already chose?", and "What type of customer (described in terms of gender, age group, income, and so on) is likely to buy a given product?"

REAL WORLD I conducted a BI training class once for a well-known online recruitment company. At the end of the training, the BI manager mentioned that he had a project to better understand their customers and the customers' buying behavior. I showed him how to build an Analysis Services data mining model for customer profiling. This company was using multidimensional cubes, but the BI manager didn't know about the Analysis Services data mining capabilities. He said that my mining model demo was "the icing on the cake."

In SQL Server 2012, Analysis Services adds the Tabular model, which is the subject of this book. If you're a BI pro and you implement OLAP cubes, SQL Server 2012 adds incremental features to Multidimensional, including dimension stores that are larger than 4GB to accommodate string-based attributes with millions of members, support for SQL Server Extended Events (XEvents), Analysis Management Objects (AMO) for PowerShell, and improved support for Non-Uniform Memory Access (NUMA) memory architectures.

Integration Services

Today's enterprise IT shop is often required to maintain an assortment of data sources and technologies. These include desktop databases, legacy mainframe systems (that no one dares to touch), relational database management systems (RDBMS), and so on. For example, order tracking data could reside in a SQL Server database, and HR data could be stored in an Oracle database, while the manufacturing data could be located in a mainframe database. Integrating disparate and heterogeneous data sources presents a major challenge for many organizations. Integration Services helps you address this challenge. It's typically used for extracting, transforming, and loading (ETL) processes.

Integration Services complements BISM very well because BISM doesn't have capabilities to transform and clean data. You would typically use Integration Services to extract data from the source systems, cleanse it, and then load it to a database, such as your company data warehouse.

New features in the 2012 release include usability, developer productivity, and manageability improvements, as discussed in the "What's New in Microsoft SQL Server Code-Named "Denali" for SQL Server Integration Services" video presentation by Matt Masson (http://bit.ly/ssis2012).

Reporting Services

Reporting is an essential feature of every BI application. With Reporting Services, you can create standard and ad hoc reports from various data sources, including BISM models. Since the 2005 release, Reporting Services includes a graphical query designer that auto-generates cube and data mining queries. To create basic queries, you can simply drag the required measures and dimensions to the query results area and then filter the results in the filter area. As a result, BI professionals can create standard reports and power users can author ad hoc reports. You can also implement reports that leverage Analysis Services data mining capabilities to display prediction results, such as forecasted sales, as I discussed in my article, "Implementing Smart Reports with the Microsoft Business Intelligence Platform" at http://bit.ly/smartreports. Once you've authored the report, you can publish it to the report server or to a SharePoint site and then make it available to other users.

The major focus for the SQL Server 2012 release was a new SharePoint 2010-integrated tool, named Power View, for authoring ad-hoc reports for data visualization and exploration. Similar to Report Builder, Power View targets business users, but it doesn't require report authoring experience. Suppose that Maya has uploaded her PowerPivot model to SharePoint. Now Maya (or anyone else who has access to the model) can quickly build a great-looking tabular or chart report in 20 minutes that visualizes data from the PowerPivot model.

Unlike Report Builder, Power View is highly interactive and presentation-ready, meaning that it doesn't require switching between report layout and preview modes. **Figure 1.3** illustrates a Power View report that shows the correlation between a sales persons' actual sales and their sales quota. You can click the play button on the CalendarYear play axis to see an animated video that shows how the performance changes over time.

Another welcome end-user oriented feature in this release is data alerts. Data alerts allow report users to subscribe to alerts when the report data changes. For example, suppose that your IT department has implemented and deployed an operational report that shows sales by countries. You want to be notified when the United States sales exceed one million. Once you set up a data alert, the report server will send you an e-mail when the condition is met. For more information about data alerts, read my blog post, "Reporting Services Data Alerts", at http://bit.ly/rsdataalerts. For more information about the Reporting Services enhancements in SQL Server 2012, see the video, "What's New in Microsoft SQL Server Code-Named "Denali" for Reporting Services" by Carolyn Chao (http://bit.ly/ssrs2012). You might find my book, *Applied SQL Server 2008 Reporting Services* (http://amzn.to/appliedssrs), useful for an in-depth knowledge of Reporting Services.

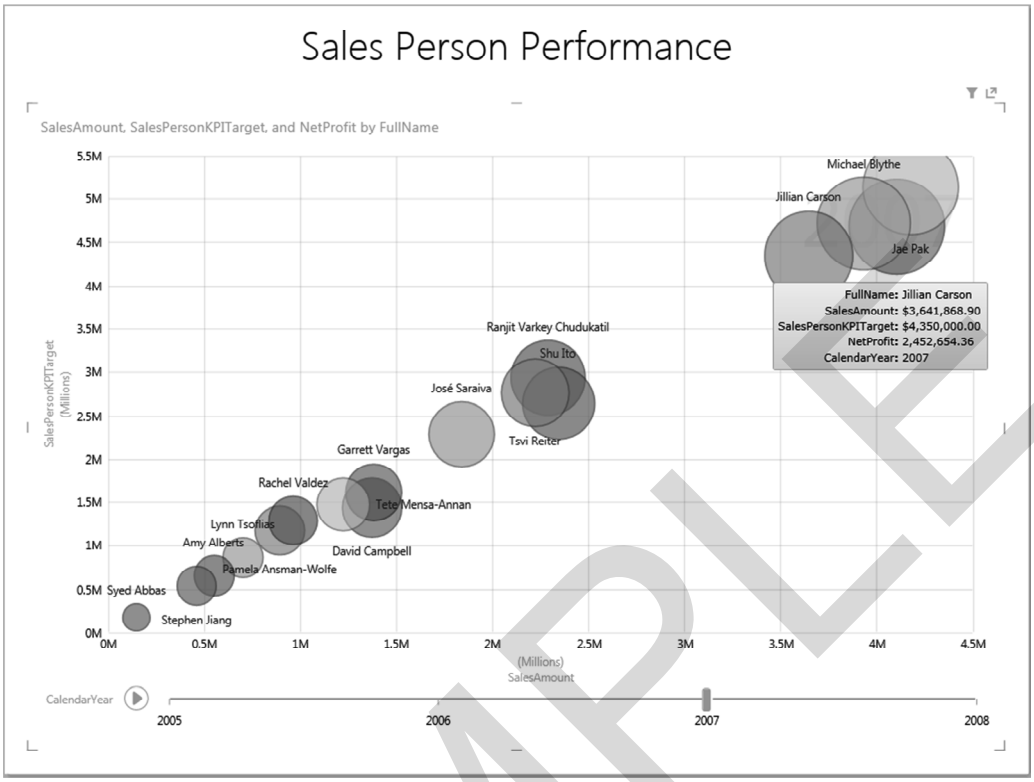

Figure 1.3 Business users can use Power View to create interactive reports.

Master Data Services and Data Quality Services

There are other SQL Server components that you might find more or less relevant to your BI projects. As you would probably agree, data quality and stewardship is an issue with many mid-size and large organizations. Chances are that your organization would like to maintain a centralized list of important entities, such as products, customers, vendors, and so on. In SQL Server 2008 R2, Microsoft introduced Master Data Services (MDS) to help enterprises centrally manage and standardize master data. The 2012 release adds the ability to use Microsoft Excel as a front-end tool to manage master data, as demonstrated in the video, "Managing Master Data with MDS and Microsoft Excel" by John McAllister (http://bit.ly/mds2012).

SQL Server 2012 introduces Data Quality Services (DQS) to enable organizations to create a knowledge base and to specify domain rules in order to perform data cleansing. For example, you can use DQS to standardize customer data, such as to change "St." to "Street" and to enrich data by filling in missing elements, such as to change "1 Microsoft way Redmond 98006" to "1 Microsoft Way, Redmond, WA 98006". Integration Services includes a new DQS Cleansing task that BI professionals can use to automate the DQS processes. For more information, see the video, "Using Knowledge to Cleanse Data with Data Quality Services" by Elad Ziklik (http://bit.ly/dqs2012).

Productivity and collaboration tools

Data by itself is useless if there is no way to make it available to the people who need it. Besides disseminating data via Reporting Services reports, the Microsoft BI platform supports other data presentation channels, such as Microsoft Office and Microsoft SharePoint.

Microsoft significantly broadened the BI features in the Microsoft Office suite of products. First, Microsoft positioned the ubiquitous Microsoft Excel as a premium client for Analysis Services. For example, business users can use Excel to connect to a tabular or multidimensional model and build interactive PivotTable reports to slice the cube data. To help end users perform data mining tasks in Excel, Microsoft released a Data Mining add-in for Microsoft Office. Unfortunately, the BI trend somewhat ebbed away and Microsoft Excel 2010 added only incremental BI features, such as slicers and the ability to search dimension members. Let's hope that a future version will add modern Power View-like data visualization features to supplement the venerable PivotTable and PivotChart reports.

Whether you like SharePoint or not, Microsoft continues adding BI features to SharePoint. Organizations can use SharePoint to build BI portals and dashboards that contain Reporting Services reports and Excel reports that are connected to BISM models. To simplify setup, the SQL Server 2012 version of Reporting Services now integrates natively with SharePoint 2010 via the SharePoint service application architecture. As a result, you don't need to install a standalone report server for SharePoint integration mode. With the Excel Services components of SharePoint 2010 Enterprise, you can deploy and process Excel spreadsheets on the server and then view them in HTML via a web browser. With PerformancePoint Services, you can implement scorecards and management dashboards, such as a scorecard that displays key performance indicators (KPIs) defined in a multidimensional or tabular model. Finally, as I pointed out, Power View augments the SharePoint 2010 BI ecosystem with ad hoc reporting from predefined BISM models.

You might have heard about the Azure Services Platform, which is a Microsoft cloud offering for hosting and scaling applications and databases through Microsoft datacenters. Microsoft Azure gives you the ability to focus on your application and to outsource infrastructure maintenance to Microsoft. IT trends indicate that cloud computing will be the next frontier for software development. As of the time of this writing, however, Analysis Services isn't available on Azure. As a result, you can't deploy multidimensional and tabular models to the cloud. However, a tabular model can import data from the cloud, and more cloud-based BI features are anticipated in future releases.

Table 1.1 summarizes the products in the Microsoft BI Platform listed in the order discussed.

Table 1.1 The Microsoft BI Platform addresses the most pressing data analysis and management needs.

Product	Acronym	Purpose
SQL Server Analysis Services	SSAS	Provides analytical and data mining services. You can use the analytical services to implement OLAP multidimensional cubes and Tabular relational-like models.
SQL Server Integration Services	SSIS	Allows BI professionals to implement extraction, transformation, and loading (ETL) processes, such as to load data from text files into a data warehouse database.
SQL Server Reporting Services	SSRS	Empowers business users and BI professionals to create standard and ad hoc reports from a variety of data sources.
SQL Server Master Data Services	MDS	Allows organizations to manage non-transactional entities, such as customer, product, and vendor, with the goal of compiling master lists.
SQL Server Data Quality Services	DQS	Provides tools to set up a knowledge base to perform a variety of data quality tasks, including correction, enrichment, standardization, and de-duplication of your data.
Microsoft Excel	Excel	Includes BI client features, such as browsing multidimensional cubes, implementing personal PowerPivot models, and performing data mining tasks.
SharePoint Products and Technologies	SharePoint	Hosts reports, PowerPivot models, and dashboards.

1.1.3 Understanding the BISM Components

Now that I've introduced you to BISM and the Microsoft BI Platform, let's take a look at its logical architecture. This will help you understand at a high-level its building blocks and how multidimensional and tabular paths fit in the BISM ecosystem. Don't worry if you don't immediately understand some of the acronyms and technologies. I'll clarify them throughout the rest of the chapter.

Vertically, BISM can be visualized as a three-tier model consisting of data access, business logic, and data model layers (see **Figure 1.4**). Horizontally, it has two implementation paths: Multidimensional and Tabular. I'll be quick to point out that although there is a significant feature overlap between these two paths, they're very different from an implementation standpoint. If you take the multidimensional path, you'll end up with an OLAP cube(s), as with prior versions of Analysis Services.

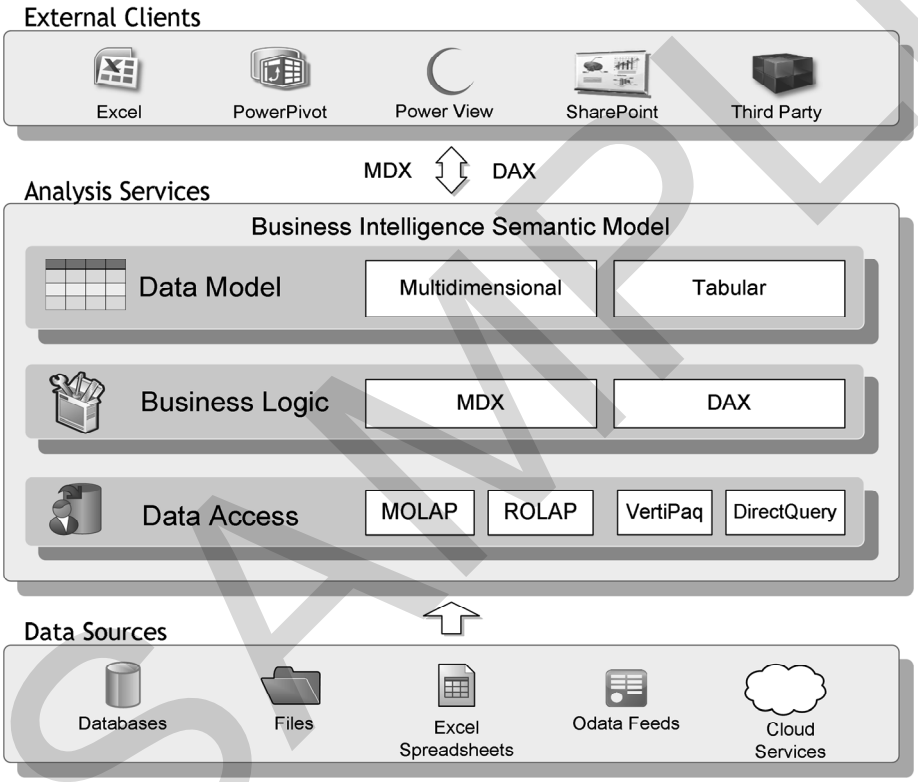

Figure 1.4 BISM has two implementation paths: Multidimensional and Tabular.

By contrast, the Tabular path leads to, you've guessed it, a tabular model. In SQL Server 2012, Microsoft doesn't provide a migration option between Multidimensional and Tabular. Therefore, it's important to carefully consider both options and then make an appropriate choice based on your requirements. Later in this chapter, Section 1.4 provides a detailed feature comparison between Multidimensional and Tabular.

Logical components
Next, I'll explain the BISM logical components.

- Data Model layer – This is the conceptual model layer that's exposed to external clients. Again, Multidimensional equates OLAP cubes. Tabular is a new model that's more relational in nature and represents data as tables and columns.
- Business Logic layer – This layer allows the modeler to define business logic, such as variances, time calculations, and key performance indicators (KPIs). Multidimensional empowers the modeler to implement this layer using Multidimensional Expressions (MDX) artifacts, such as calculated members, named sets, and scope assignments. Tabular embraces a new Excel-like expression language called Data Analysis Expressions (DAX). DAX was first introduced in PowerPivot 1.0, which was released with SQL Server 2008 R2. SQL Server 2012 adds additional formulas and extends DAX to organizational models. This release also adds a DAX-based query language so that Power View reports can query tabular models natively.
- Data Access layer – This layer interfaces with external data sources. By default, both Multidimensional and Tabular import data from the data sources and cache the dataset on the server for best performance. The default multidimensional storage option is Multidimensional OLAP (MOLAP), where data is stored in a compressed disk-based format. The default Tabular storage option is VertiPaq, which was first introduced in PowerPivot. VertiPaq is an in-memory columnar database that compresses and stores data in the computer main memory. Both Multidimensional and Tabular support real-time data access by providing a Relational OLAP (ROLAP) storage mode for Multidimensional and a DirectQuery storage mode for Tabular (organization BI models only). When a multidimensional cube is configured for ROLAP, Analysis Services doesn't process and cache data on the server. Instead, it auto-generates and sends queries to the database. Similarly, when a tabular model is configured for DirectQuery, Analysis Services doesn't keep data in VertiPaq but sends queries directly to the data source.

Data sources
Although Multidimensional and Tabular can import data from any data source that provides standard connectivity options, there are some differences. Multidimensional works best with a single database, such as a data warehouse database. By contrast, Tabular can connect and import data equally well from heterogeneous data sources, such as databases, text files, and even data feeds, such as a Reporting Services report or a SharePoint list.

External clients
BISM provides two interfaces to the outside world for querying Multidimensional and Tabular models: MDX and DAX. OLAP browsers, such as Microsoft Excel, can see both models as OLAP cubes and can send MDX queries. And, Tabular-aware clients, such as Power View, can send DAX queries to tabular models.

NOTE As of now, the only Microsoft client that generates DAX queries is Power View. As it stands, Power View auto-generates DAX queries to tabular models only. Microsoft is working on extending BISM with DAX support for multidimensional cubes so that Power View can use them as data sources. The rest of the Microsoft-provided reporting tools, such as Reporting Services designers and Excel, can connect to a tabular model as a cube and send MDX queries to it.

Table 1.2 shows the most commonly used Microsoft-provided clients and the query options they support for integrating with BISM.

Understanding BISM unification
Outside query and management interfaces, there aren't many commonalities between Multidimensional and Tabular in SQL Server 2012. As you'll see in Section 1.4.1, which compares Multi-

dimensional and Tabular side by side, there is a significant functionality overlap, but each model implements features on its own, and the models are not compatible with each other. For example, you can't get a hierarchy definition from Multidimensional and add it to Tabular or use MDX to define calculations in Tabular. So, why are we implying unification here? Let's think of BISM as a prediction of the future. In time, Tabular might "borrow" OLAP features, such as MDX calculations, and Multidimensional might get Tabular artifacts, such as in-memory data storage. Thus, the difference between the two paths is likely to blur to a point where Tabular is as feature-rich as Multidimensional.

Table 1.2 Microsoft-provided clients and query options.

Tool	Multidimensional	Tabular	Business Intelligence Need
Excel	MDX	MDX	Personal and organizational BI
Excel PowerPivot	MDX	MDX	Personal and team BI
Power View	N/A	DAX	Personal, team, and organizational BI
PerformancePoint	MDX	MDX	Organizational BI
SSRS report designers	MDX	MDX	Personal and Organizational BI

Taking this further, I'd like to indulge myself and think that one day, I hope in the not-so-distant future, BISM will evolve to become a true single model that delivers on the unification promise and that combines the best of Multidimensional and Tabular. Then, we would be able to pick and mix features from both paths, such as in-memory storage with MDX calculations, and we could use the best technology for the task at hand. Until then, we will have to choose between the multidimensional and tabular paths. The next section provides the necessary technical information to help you make this choice.

1.2 Understanding Tabular

Let's now take a more detailed look at Tabular. I'll start by explaining why we need it. Next, I'll discuss when to use it and scenarios where it might not be a good fit. Finally, I'll provide a feature comparison between Tabular and Multidimensional, and between Tabular and Report Builder models.

1.2.1 Why We Need Tabular

The short answer is to have a simple and efficient BI model that promotes rapid BI development and is well suited for both personal and organizational BI projects. Next, I'll review the main events that led to the development of Tabular.

Advances in computer hardware

When OLAP systems came to existence and the first version of Analysis Services was released, the computer landscape was quite different. Back then, one gigabyte was a lot of data. Nowadays, terabyte-sized databases are not uncommon. You can buy an inexpensive laptop for home use that's equipped with a 64-bit multi-core processor, 4GB of RAM, and one terabyte of disk space

for less than $1,000. A typical enterprise database server has four physical processors and at least 64GB of RAM. That's a lot of raw power!

REAL WORLD I recall working as a consultant in 2000 on a data warehouse project and I wondered where to store a few gigabytes of call center data. To get us out of this predicament, the customer eventually shipped a very expensive server that could handle the "massive" data volumes.

These advances in desktop and server hardware led to the rise of the in-memory databases (IMDBs). Unlike the traditional database management systems (DBMS), which employ a disk-storage mechanism, an in-memory database stores data in the computer main memory (RAM) because RAM is still the fastest storage medium. IMDBs have actually been around since the 1970s, but high-capacity RAM has only recently become an affordable option to store large volumes of data, such as millions or even billions of rows.

Demand for self-service business intelligence

On the software side of things, in recent years there has been a need for personal (self-service) BI tools, such as to allow a business user to import and analyze some data without reliance on IT pros. However, OLAP and multidimensional cubes were never intended for personal BI. In a typical OLAP project, BI professionals would develop and deploy a cube to a dedicated server, and end users would run standard or interactive reports that source data from the cube. Although the data is clean and trusted, it's restricted to what's in the cube and how often the cube is refreshed.

The BI competitive landscape has also changed. New vendors and products are emerging every day, including QlikView, Tableau, SiSense Prism, Spotfire, and BusinessObjects to name a few. The main selling point of these BI vendors is that organizational BI is too complex and cost-prohibitive. They promise a faster BI track. In reality, however, you should view organizational BI and personal BI as completing and not competing technologies because both might be required for companies to meet their analytical needs. For example, many companies require extract, transform, and load (ETL) processes, a data warehouse, and an OLAP layer. At the same time, there are cases where power users can do data analysis and reporting without reliance on IT pros. And, in many cases there is a demand for both.

REAL WORLD You should doubt any statements where someone says that organizational BI or OLAP are not needed or are obsolete. I've seen cases where BI vendors swear by self-service BI, but they build demos directly on top of a data warehouse where data is neatly organized and stored in a set of dimensions and fact tables. However, you should know that the effort required for extracting, cleaning, and loading data usually takes about 60-80% of the development effort for implementing a BI solution. Once this is done, building analytical layers is a straightforward process. A major hospital once decided they didn't need organizational BI and purchased a popular self-service BI tool. In less than a year, they went back to data warehousing and OLAP. As with most things in life, if something sounds too good to be true, it probably is.

1.2.2 Understanding Tabular Design Goals

The Analysis Services architects were looking for ways to capitalize on the modern hardware advances and to provide tools for personal BI. They envisioned a model that's simple to use and yet performs well. They had a dilemma — should they continue building on the OLAP foundation or go back to the drawing board. Microsoft realized that, although it's very successful and widely adopted, the Multidimensional internal architecture and storage model have become increasingly complex over the years, thus barring major changes and growth. Instead of continuing to build on the past, they've decided to architect a new model – Tabular.

Striving for simplicity

Simplicity was the main design goal for Tabular. A self-service BI tool that targets business users shouldn't assume expert technical knowledge or skills. A lot of effort went into simplifying Tabular in all directions. As its name suggests, Tabular embraces a more intuitive entity representation, using relational constructs such as tables and relationships, as opposed to multidimensional artifacts, such as cubes and dimensions. Moving to the business logic layer, Data Analysis Expressions (DAX) uses the standard Excel formula syntax. There are millions of Microsoft Excel users who are familiar with using Excel formulas to perform calculations. DAX continues this trend by extending Excel formulas so that users can define business logic in Tabular that ranges from simple calculated columns, such as a column that concatenates the customer's first and last name, to more advanced aggregations, such as YTD and QTD calculations.

NOTE It's easy to get started with DAX, to create basic expression-based columns, and to implement simple calculated measures, such as distinct count, time calculations, and so on. However, I'll be quick to point out that DAX can get complicated. If your goal in moving to Tabular is to avoid MDX, you might be disappointed. While a future release might be easier to learn, for now, after you moved beyond the basics, be prepared to struggle and invest time to learn DAX.

Furthermore, the Tabular data access layer makes it really easy to import data from virtually any data source without requiring scripting or query knowledge. Finally, data analysis and reporting can be performed with the familiar Excel PivotTable and PivotChart or with Power View, all without requiring prior report authoring experience. These are just a few examples of how Microsoft designed Tabular with simplicity in mind.

Of course, a simple model probably won't go very far if it lacks essential features. Although in its first release (or second if you count PowerPivot), you might find Tabular surprisingly feature-rich, depending on your reference point. For example, you'll probably find Tabular more advanced and powerful than other in-memory offerings on the market. However, if you compare it with multidimensional cubes, you'll undoubtedly find missing features. This isn't surprising because Multidimensional sets a very high bar.

Striving for performance

Next to simplicity and ease of use, the second tenet of the Tabular design is performance. Tabular is designed to give you high performance by default without requiring special tuning. To provide the best storage performance, Microsoft implemented a proprietary in-memory store called VertiPaq. VertiPaq is an in-memory database, which means it loads data in the computer main memory. As its name suggests, it stores and compresses data vertically by columns to pack as much data as possible in order to minimize the storage footprint. Column-based storage fits BI like a glove because data is typically analyzed by columns. And, the lower the data cardinality (that's the more repeating values a column has), the higher its compression rate is. **Figure 1.5** shows a hypothetical table containing some sales data.

Two distinct values	Seven distinct values	Five distinct values	Two distinct values	Five distinct values
Date	ProductName	Product Subcategory	ProductCategory	SalesAmount
1/1/2011	Hitch Rack - 4-Bike	Bike Racks	Accessories	44.88
1/1/2011	Bike Wash - Dissolver	Cleaners	Accessories	2.9733
1/1/2011	Mountain-400-W Silver, 38	Mountain Bikes	Bikes	419.7784
1/2/2011	Mountain-400-W Silver, 40	Mountain Bikes	Bikes	419.7784
1/2/2011	Road-250 Red, 44	Road Bikes	Bikes	1518.7864
1/2/2011	Road-250 Red, 48	Road Bikes	Bikes	1518.7864
1/2/2011	Sport-100 Helmet, Red	Helmets	Accessories	12.0278

Figure 1.5 VertiPaq compresses well columns with many repeating values.

After reviewing the column data, we realize that across all rows the Date column has only two unique values, the ProductName column has seven, the ProductSubcategory has five, ProductCategory two, and Sales Amount has five. Consequently, the Product Category and Date columns will compress the most since they have the most repeating values, while the ProductName column won't compress so well. Since it's common to have columns with many repeating values in large datasets, expect a high data compression ratio (five times or higher).

Another performance-related effort went into DAX. Specifically, Microsoft optimized DAX for modern multi-core processors. For readers experienced in Analysis Services and MDX, all DAX formulas operate in a block computation mode and should give you performance at par with optimized MDX queries.

1.2.3 Understanding Tabular and VertiPaq Implementations

Having reviewed the Tabular logical architecture and genesis, let's discuss its physical realization. As it turns out, SQL Server 2012 includes more than one implementation of both Tabular and VertiPaq, as shown in **Figure 1.6**.

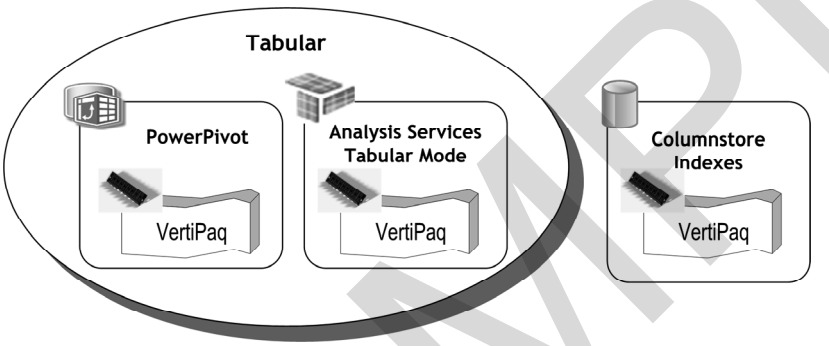

Figure 1.6 In SQL Server 2012, Tabular is delivered as PowerPivot and Analysis Services in Tabular mode.

Tabular implementations

In SQL Server 2012, Tabular is delivered in two ways:

- PowerPivot (for Excel and SharePoint) – PowerPivot is an implementation of Tabular for personal and team BI. When the model is loaded, data resides in the in-memory VertiPaq store.
- Analysis Services in Tabular mode – Analysis Services in Tabular mode is an implementation of Tabular for organizational BI. By default, Analysis Services in Tabular mode caches data in the VertiPaq store, but it can be configured to generate and pass queries to a SQL Server database.

Both PowerPivot and Analysis Services in Tabular mode include modeling environments for implementing tabular solutions, and therefore, they are Tabular.

VertiPaq implementations

Microsoft provides three VertiPaq hosting implementations:

- PowerPivot (Excel and SharePoint) – When an end-user creates a PowerPivot model in Excel, behind the scenes PowerPivot stores the imported data in an in-process VertiPaq store. Similarly, if the user deploys the model to SharePoint, an Analysis Services server running in a SharePoint integration mode extracts the model data and caches it in a VertiPaq engine that's hosted by an Analysis Services instance configured in SharePoint integration mode.

- Analysis Services in Tabular mode – SQL Server 2012 allows you to install a stand-alone instance of Analysis Services in Tabular mode that uses the VertiPaq engine by default for storage without requiring SharePoint. Because of this, you can deploy a tabular model to a dedicated server just like you could deploy a cube. One caveat is that you can't deploy both Tabular and Multidimensional models to the same Analysis Services instance. However, you can install multiple Analysis Services instances, such as one instance for Multidimensional and another Tabular, on the same server.
- Columnstore indexes – Another interesting VertiPaq implementation that debuts in SQL Server 2012 is creating a new type of index on a database table, such as a fact table in a data warehouse database. A columnstore index might dramatically improve performance of queries that aggregate data from that table, such as an operational report that sources and aggregates data directly from a data warehouse database. Behind the scenes, a columnstore index is powered by VertiPaq running inside the SQL Server Database Engine. For more information about columnstore indexes, see the video, "Columnstore Indexes Unveiled" by Eric Hanson (http://bit.ly/colulmnstoreindex). Columnstore indexes are not an implementation of Tabular because they don't provide a modeling environment.

At this point, you should have a good high-level understanding of Tabular and what factors led to its invention. Let's now turn our focus to its usage scenarios.

1.3 When to Use Tabular

You can use Tabular to implement BI solutions that can address various personal, team, and organizational BI needs on a single platform, as shown in **Figure 1.7**. Before explaining each scenario, let me emphasize the flexibility that the platform offers. As the diagram suggests, a business user can start the BI journey with a small, personal model that she implements in Excel. Then, she might decide to share it with co-workers by deploying the Excel workbook to SharePoint. At the other end of the spectrum, BI pros can upgrade the model for organizational use when it exceeds Excel capabilities. But you don't necessarily have to follow this continuum story. For example, a BI professional can develop a PowerPivot model and then deploy it to SharePoint if business requirements call for online and offline Excel-based reporting. Or, the BI pro can jump straight into organizational BI and implement a tabular model from scratch on top of an existing database. There are many possibilities that depend on your requirements.

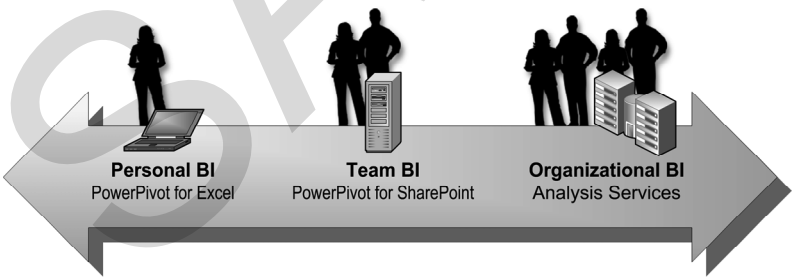

Figure 1.7 You can use Tabular to implement personal, team, and organizational BI solutions.

In general, Tabular is specifically suited for personal BI (PowerPivot for Excel) and team BI (PowerPivot for SharePoint) projects. Organizational BI projects that seek to implement a centralized analytical layer, such as a layer on top of a data warehouse database for trend or historical analysis, will require careful evaluation of Multidimensional and Tabular in order to choose the best technology for the task at hand. The following sections clarify how you might use Tabular usage.

1.3.1 Personal Business Intelligence

Remember that personal BI is about empowering business users to build ad hoc BI models and to analyze data on their own. The natural choice for a personal BI tool would be the most popular application, which is used by millions of users – Microsoft Excel. A special Excel add-in, called PowerPivot for Excel, allows you to import data from virtually any data source, and you use the familiar Excel PivotTable and PivotChart to analyze the data.

NOTE Personal BI requires only Microsoft Excel 2010 and the free PowerPivot for Excel add-in. You don't need SharePoint, SQL Server, or any other software if you don't plan to share your PowerPivot models with other users.

Understanding personal BI components

Personal BI deployment is simple (see **Figure 1.8**). The PowerPivot for Excel add-in extends Microsoft Excel's capabilities and enables you to import tables of data, set up relationships, and define calculations. The imported data is saved inside the Excel 2010 workbook file, but it's loaded into the in-memory VertiPaq engine when you work with and query the model.

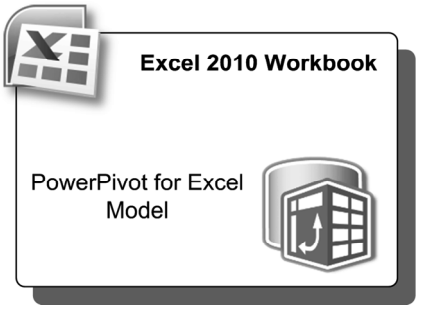

Figure 1.8 PowerPivot for Excel saves the model data inside the Excel 2010 workbook file.

For reporting, PowerPivot for Excel falls back on the Excel PivotTable and PivotChart. **Figure 1.9** shows a sample PowerPivot for Excel dashboard, which you'll design in Chapter 5.

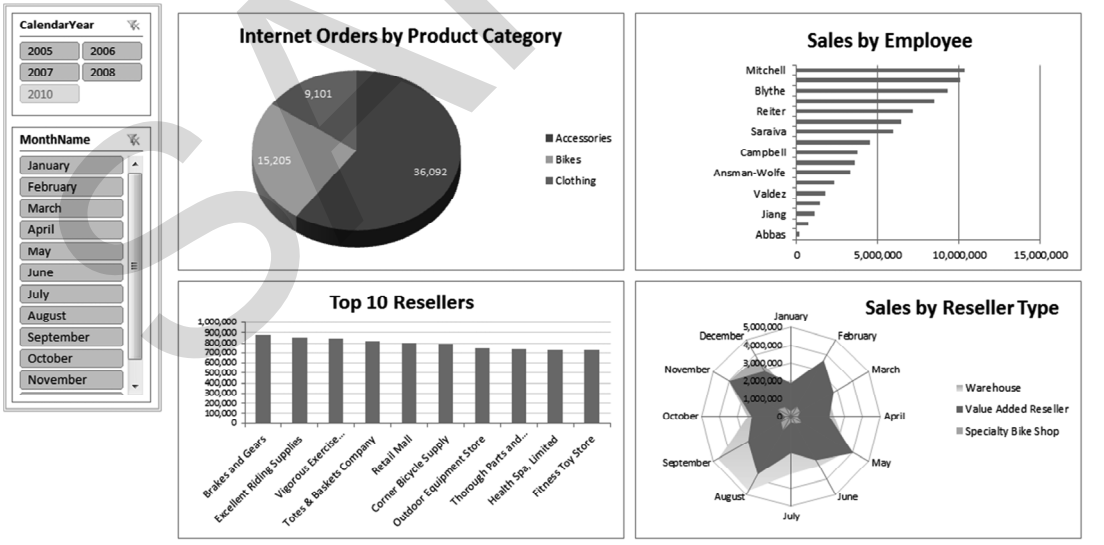

Figure 1.9 Use PowerPivot for Excel to create dashboards consisting of PivotTable and PivotChart reports.

PowerPivot for Excel lifts the Excel data restrictions and lets business users import more than a million rows from external data sources, if needed. However, because SharePoint is limited to two gigabytes of maximum file upload size, PowerPivot for Excel limits the Excel workbook file to that size. Although this may sound limiting, recall that the data is saved in a highly compressed state. Therefore, a gigabyte of compressed data might contain millions of rows. If the file size becomes a limiting factor, consider upgrading the PowerPivot model to an organizational model that's deployed to a dedicated Analysis Services server, as I'll discuss later in Section 1.3.3.

New features in PowerPivot for Excel

Microsoft introduced PowerPivot for Excel in May 2010. PowerPivot version 2 coincides with the SQL Server 2012 release and adds features that extend PowerPivot capabilities and that make business users more productive. For example, one of the most requested features was the ability to visualize the model schema and relationships graphically. PowerPivot 2.0 responds to this need by introducing Diagram View, as shown in **Figure 1.10**.

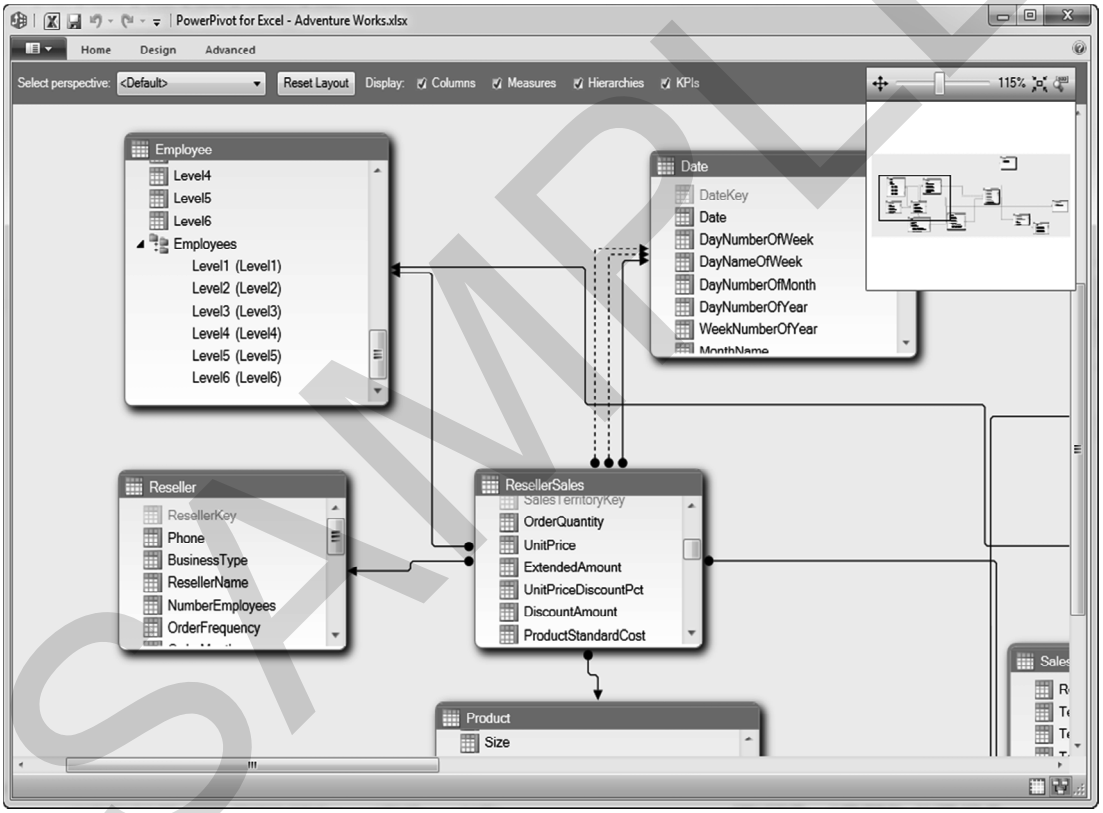

Figure 1.10 PowerPivot for Excel adds Diagram View to visualize tables and relationships.

PowerPivot 2.0 introduces new modeling features to make the model more intuitive for analysis and reporting. You can now create hierarchies to provide useful navigation, such as a product hierarchy where you drill into data by product category, subcategory, and product levels. With key performance indicators (KPIs), you can track the company's performance against a predefined goal. And, perspectives define logical views that reduce the perceived complexity of the model.

PowerPivot now automatically supports drillthrough actions in Excel so you could see the level of detail behind a cell.

Microsoft improved the metadata presentation, including sorting column values by another column, applying model data types and formats to pivot reports, sorting object names alphabetically, formatting calculated measures, and providing descriptions for tables and measures. PowerPivot now supports dates better and allows you to mark a table as a Date table to enable date filters in Excel reports. A set of reporting properties was introduced to support Power View reports. For example, PowerPivot now supports importing images so you can display them on a Power View report, such as to allow the user to filter the report data by clicking a product image.

Finally, this release adds new DAX functions for implementing advanced business calculations. DAX has been extended with new ranking, statistical functions, information functions, and functions to handle parent-child hierarchies, such as an organizational chart. DAX now supports a query language to allow report clients, such as Power View, to query models deployed to SharePoint and Analysis Services servers.

1.3.2 Team Business Intelligence

Team BI enables business users to share the BI artifacts they create. Microsoft SharePoint 2010 has been extended to support hosting PowerPivot models and to allow business users to create reports from them. If you've used Multidimensional, you've probably deployed Excel and Reporting Services reports to SharePoint so team BI is nothing new to you. If you've started down the personal BI road with Tabular, you'll probably find e-mailing large Excel files to your coworkers impractical. Instead, consider deploying PowerPivot models to SharePoint to take advantage of features, such as online viewing of Excel pivot reports, authoring interactive Power View reports, and setting PowerPivot models to refresh on schedule.

NOTE Team BI requires SharePoint Server 2010 Enterprise and SQL Server 2012 Business Intelligence or Enterprise Edition. To learn about SQL Server editions and licensing, refer to the SQL Server 2012 website at http://bit.ly/sql2012editions. An Analysis Services instance configured in SharePoint integrated mode needs to be installed on the SharePoint application server.

Understanding team BI components

Due to the nature of the SharePoint architecture, team BI has a more complicated deployment model that requires installing and configuring PowerPivot for SharePoint and SQL Server 2012 on the SharePoint server, as shown in **Figure 1.11**. This diagram illustrates only the high-level logical components and hides their interactions.

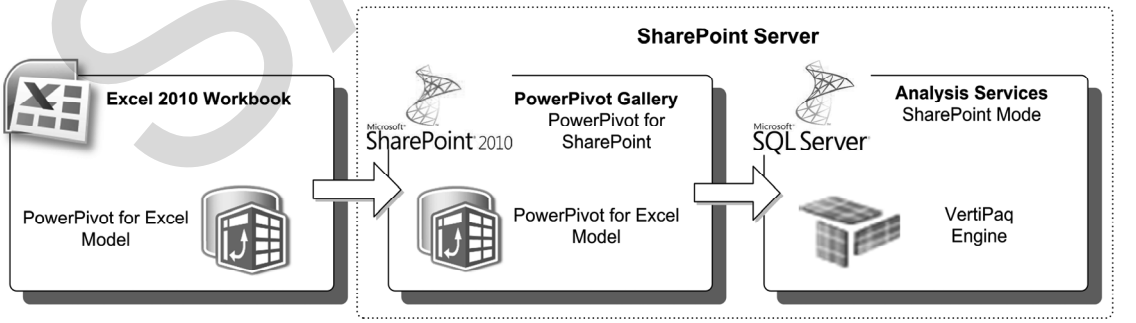

Figure 1.11 Team BI requires SharePoint Server, SQL Server, and PowerPivot for SharePoint.

The PowerPivot for SharePoint add-in extends SharePoint's capabilities to support deploying and querying PowerPivot models. End users can upload the Excel workbooks to a special SharePoint PowerPivot Gallery document library, which is specifically designed to support PowerPivot models and Power View reports. When a user clicks on the workbook link, SharePoint renders the workbook in a thin-client HTML format inside the web browser. Consequently, Excel 2010 doesn't need to be installed on the desktop for the sole purpose of viewing reports only. This is no different than rendering Excel pivot reports that are connected to multidimensional cubes and deployed to SharePoint.

Excel web reports support limited interactivity when filtering and sorting data on the report. However, the end user can't change the report layout. For example, the user can't add or remove fields and can't switch between a pivot report to a chart report. When the user attempts an interactive feature, Excel forwards the request to an Analysis Services instance configured in SharePoint integrated mode. Analysis Services extracts the model data, restores it on the Analysis Services instance, and services report queries from that instance. In this way, the PowerPivot model can scale up to many simultaneous users. PowerPivot for SharePoint also adds the ability to refresh the model data on a set schedule, such as once every month.

As you would probably agree, personal BI is great, but "managed" personal BI is even better. PowerPivot for SharePoint provides a management dashboard for IT pros to gain insights about the server and model utilization and to take actions. For example, after analyzing the server workload and activity, the IT department might want to upgrade a frequently used PowerPivot model to an organizational tabular model to make it more scalable or to apply row-level security.

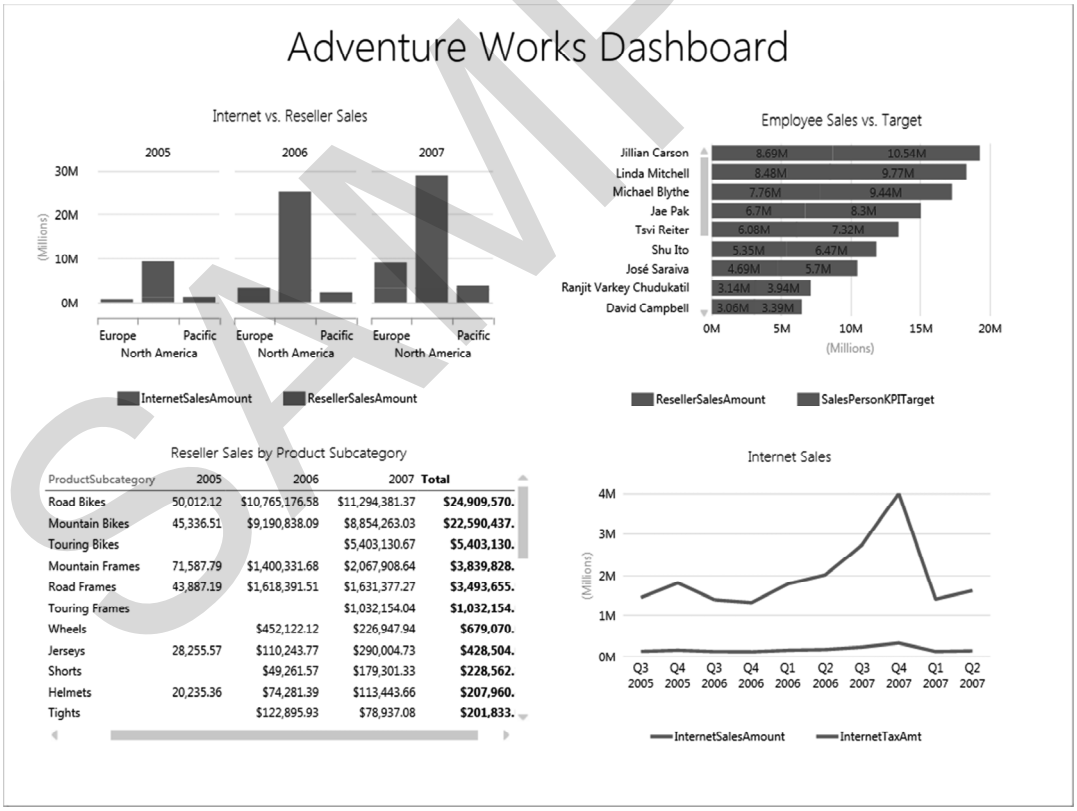

Figure 1.12 Business users can author interactive reports with Power View.

New features for team BI

SQL Server 2012 adds new features to team BI, including:

- Power View reporting – With a click of a button, you can launch Power View and create an interactive report that's using a PowerPivot model or an organizational model as a data source. And, you can quickly assemble interactive reports that consolidate and aggregate information to analyze important metrics at a glance, such as the report shown in **Figure 1.12**.
- Easier setup – SQL Server setup has been refactored to exclude the PowerPivot for SharePoint installation tasks in order to simplify configuration and troubleshooting. Microsoft has provided the new PowerPivot Configuration Tool for configuring, repairing, and uninstalling PowerPivot for SharePoint.

1.3.3 Organizational Business Intelligence

Personal and team BI scenarios with PowerPivot have been possible since its first release in May 2010. However, using the VertiPaq technology to implement organizational analytical models is new with SQL Server 2012. Organizational BI is powered by an Analysis Services instance configured in Tabular mode. For a lack of a better term, I'll refer to organizational BI with Tabular as *Analysis Services Tabular*. Consider Analysis Services Tabular when:

- You've outgrown the PowerPivot capabilities. For example, a business user might have started with a PowerPivot model but have exceeded the Excel file size limit. Or, you might require row-level security to the model – a feature that PowerPivot doesn't support.
- You start from scratch, and all you need is a simple model for slicing and dicing data that wraps an existing database. Be careful here because many projects start simple but grow in complexity over time. For this reason, I personally favor Multidimensional as an analytical layer for data warehousing projects because it's more mature and feature-rich. I'll provide a detailed feature comparison between Tabular and Multidimensional in the next section to help you make an educated choice.

NOTE Organizational BI requires SQL Server 2012 Business Intelligence or Enterprise Edition. A dedicated Analysis Services instance configured in Tabular mode is required to deploy tabular models. You can't deploy tabular models to an Analysis Services instance that's configured in SharePoint mode or to an Analysis Services instance that's configured in Multidimensional mode.

Understanding Analysis Services Tabular components

As a BI pro, you would use a professional toolset to model organizational models, as shown in **Figure 1.13**.

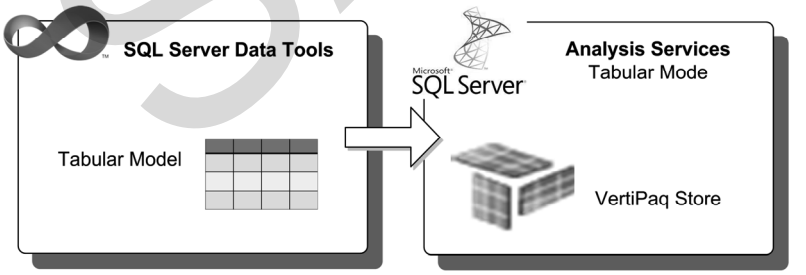

Figure 1.13 Professionals can use SQL Server Data Tools to design models for organizational BI and deploy them to Analysis Services configured in Tabular mode.

This release introduces SQL Server Data Tools, which provides an integrated environment for database and BI developers to design for SQL Server (both on and off premise) within the Visual Studio 2010 IDE. SQL Server Data Tools (SSDT) succeeds Business Intelligence Development Studio (BIDS) and extends its capabilities to support SQL Server development (hence the name change). As far as tabular modeling goes, the design experience is very similar to PowerPivot. However, SSDT adds new features that aren't available in PowerPivot, such as source code control and project deployment. Similar to implementing multidimensional cubes, once the model is ready it can be deployed to a standalone Analysis Services instance. The difference is that this instance must run in Tabular mode, which is new to this release of Analysis Services.

Understanding organizational BI features

Analysis Services Tabular adds two main features for organizational use that are not found in PowerPivot. First, the modeler can define row-level security by using row filters and DAX formulas to restrict access to data. Also, the row filters can use the Windows identity of the user to implement dynamic data security, such as to restrict a manager to see data for his subordinates only. As with Multidimensional, Analysis Services Tabular supports only Windows-based security where the user is authenticated and authorized, based on her Windows logon account.

Analysis Services Tabular also adds the ability to partition large tables in order to manage them more efficiently, such as to refresh only new data instead of reimporting all the data. VertiPaq is the default storage of Analysis Services Tabular. Consequently, the compressed data resides in memory. This works great when all the data fits in a computer's main memory. When it doesn't, Analysis Services Tabular starts paging data to disk in order to reduce the memory consumption at the expense of query performance. Interestingly, Analysis Services Tabular supports a second storage mode where the server auto-generates and passes through queries to the SQL Server database that's used as a data source. This storage option, called DirectQuery, could be useful when data doesn't fit in memory or when you need real-time data access. The name "DirectQuery" reflects the main characteristics of this storage mode, which is to directly query the database.

1.3.4 Comparing Tabular Features

To summarize, Tabular supports personal BI (PowerPivot), team BI (PowerPivot for SharePoint), and organizational BI (Analysis Services Tabular) deployment scenarios. Let's take a moment to recap features and to compare these three deployment scenarios. This will help you understand their commonalities and differences. **Table 1.3** compares PowerPivot for Excel, PowerPivot for SharePoint, and Analysis Services Tabular features.

Data model differences

To start with, only PowerPivot for Excel and Analysis Services Tabular (with SQL Server Data Tools) provide modeling capabilities. You can use PowerPivot for SharePoint to publish PowerPivot models and to report from published models or external Analysis Services Tabular models. Only SQL Server Data Tools (SSDT), which is used to develop organizational models deployed to Analysis Services Tabular, provides source control. Although PowerPivot for SharePoint doesn't support source control, SharePoint can be configured to maintain document versions and checking out/in documents. Each time you check in a PowerPivot workbook, a new version is created.

Only Analysis Services Tabular models support row-level security. PowerPivot for Excel models are saved as Excel files and can't be accessed and queried by external clients without SharePoint. By contrast, team BI models deployed to SharePoint and organizational BI models deployed to Analysis Services Tabular can be used as data sources and queried by report clients.

Table 1.3 Understanding Tabular features for personal, team, and organizational BI

Layer	Feature	PowerPivot for Excel Personal BI	PowerPivot for SharePoint Team BI	Analysis Services Tabular Organizational BI
Data Model	Modeling environment	Yes	No	Yes
	Source control	No	No	Yes
	Row-level security	No	No	Yes
	Externally accessible	No	Yes	Yes
Data Access	Dataset size	2GB	2GB	Unlimited
	Partitions	No	No	Yes
	DirectQuery	No	No	Yes
	Auto-refresh	No	Yes	Yes

Data access differences
PowerPivot models are limited to 2GB of compressed data. By contrast, organizational models don't have this restriction. Only Analysis Services Tabular models can have partitions and paging. This makes sense, considering they can store large volumes of data. Partitions can help the administrator refresh data selectively, such as to import only the most recent data.

DirectQuery facilitates real-time data access by passing queries directly to the database instead of caching data in the model. DirectQuery is supported for organizational BI models only. Finally, PowerPivot for Excel doesn't support refreshing data automatically. Both PowerPivot for SharePoint and Analysis Services provide options to schedule automatic data refreshes.

1.4 Comparing Tabular with Other Models

As you've seen, Tabular is an excellent choice for personal and team BI. That's because Multidimensional was never meant to fulfill these needs. But which path should BI professionals take for organizational BI solutions? This is where things get trickier because you need to weigh the different model capabilities and make a choice. This section is meant for readers who have experience with multidimensional cubes or Report Builder models. First, I'll compare Tabular and Multidimensional followed by an evaluation of Tabular and Report Builder models.

1.4.1 Comparing Tabular and Multidimensional

Table 1.4 provides a side-by-side feature comparison of Tabular and Multidimensional. You'll find that Tabular is somewhat lacking in features in comparison with Multidimensional. That shouldn't surprise you, considering that Microsoft OLAP has been around for 15 years and has had many releases of features and improvements.

Table 1.4 Comparing Tabular and Multidimensional

Layer	Feature	Tabular	Multidimensional
Data Model	Schema	Relational (tables, columns)	Multidimensional (cubes, dimensions, attributes, measures)
	Declarative relationships	Regular, role-playing (with DAX only)	Regular, parent-child, many-to-many, referenced, role-playing, data mining
	End-user model	KPIs, perspectives, default drillthrough action	KPIs, perspectives, translations, actions
	Aggregation functions	Sum, Count, Min, Max, Average, DistinctCount	Sum, Count, Min, Max, DistinctCount, semi-additive functions
Business Logic	Expression language	DAX	MDX
	Constructs	Calculated columns, calculated measures	Calculated members, named sets, scope assignments
	Extensibility	No	.NET stored procedures
Data Access	Primary storage	Memory	Disk
	Partitions	Processed serially	Processed in parallel
	Aggregations	N/A	Yes
	Data sources	Relational, multidimensional, flat files, OData, SSRS reports	Relational (single database best)
	Storage modes	VertiPaq, DirectQuery	MOLAP, ROLAP, HOLAP, proactive caching

Let's discuss these results in more detail, starting with the data model layer.

Data model

The Tabular schema is expressed in relational constructs, such as tables and columns. Therefore, novice users might find Tabular easier to start with and more suitable for simple models. By contrast, Multidimensional has deep roots in OLAP and its artifacts: cubes, dimensions, and measures. Multidimensional is a more sophisticated and mature model, and it presents a higher learning curve. There are more knobs to turn, many of which don't have Tabular equivalents, such as default members, custom rollup, discretized attributes, unary operators, attribute types, and controlling the visibility of hierarchy levels.

REAL WORLD I worked once for a provider of solutions for the finance industry, and I implemented OLAP cubes for accounting and profitability management. Finance applications tend to be rather complex. Some of the Multidimensional features mentioned above proved very useful to meet our requirements, such as custom rollup and unary operators to aggregate a chart of accounts, as well as scope assignments to implement custom aggregation. The point I am trying to make is that although Tabular is simple, it's also lacking features. The more complex the requirements are, the more you should gravitate toward Multidimensional.

Declarative relationships help you define entity relationships when you're designing the model (design time), without using programming constructs. Tabular supports only regular and role-playing declarative relationships with the limitation that only one role-playing relationship can be configured as active. By contrast, Multidimensional supports more flexible relationship types. The most useful types are parent-child relationships that represent recursive hierarchies and many-to-

many relationships. An example of a parent-child relationship is an organizational chart where a manager can have subordinates.

A classic example for a many-to-many relationship is a joint bank account that's shared by two or more individuals. With Tabular, you can use DAX formulas to handle parent-child relationships and many-to-many relationships but there are some pitfalls. For example, you need to "naturalize" a parent-child hierarchy by using DAX functions that expand levels into columns. To handle many-to-many relationships, you need to define calculated measures for each numeric column that needs to be aggregated. This increases the model complexity and might present usability and maintenance challenges.

By end-user features, I mean additional features that can be added to the model to further enrich it and make it more intuitive and useful for end users. With Tabular, you can define key performance indicators (KPIs), which are essentially "super" measures with goal and target expressions. You can use perspectives to define logical metadata views to reduce the perceived complexity of models with many tables. In addition, Tabular supports default drillthrough actions that show the detail rows and all the table columns behind an aggregated cell value. Tabular doesn't support other action types, such as report actions that launch a web page or a Reporting Services report.

NOTE The BIDS Helper project (http://bidshelper.codeplex.com) allows you to customize columns returned by drillthrough actions in tabular models. It also supports report, URL, and rowset actions.

Tabular doesn't support translations to localize the metadata and data for international users. Display folders and member properties are not supported in Tabular. Multidimensional supports semi-additive functions, such as to support account and inventory balances. Tabular requires DAX calculations to support semi-additive measures.

Business logic

Now we are moving to the business logic layer. Data Analysis Expressions (DAX) is the expression language for Tabular. You can use DAX to define calculated columns that store (materialize) the expression results. The closest Multidimensional equivalent of DAX calculated columns are named calculations in the Data Source View (DSV). You can also define calculated measures whose aggregation behavior is evaluated dynamically, depending on the execution context, that is, how the user slices and dices data. Calculated measures are to the MDX regular calculated members.

Tabular doesn't support named sets (pre-defined sets of members, such as top ten products), dimension calculated members, and scope assignments. The most important of these are scope assignments that let you write to the multidimensional cube space. Since scope assignments are not supported in Tabular, you have no other choice but to create a calculated measure for each column that needs a custom aggregation, such as Sales Amount YTD, Order Quantity YTD, and so on. Again, this might present maintenance and usability challenges. Finally, Multidimensional can be extended with external .NET code in the form of Analysis Services stored procedures. This option isn't available with Tabular.

Tabular doesn't include a business intelligence wizard and it doesn't support pre-defined analytical features, such as currency conversion, time intelligence, unary operators, custom member formulas, and account intelligence. Since Tabular doesn't support writeback, you can't use Excel what-if analysis.

Data access

The default storage mode for Tabular is memory while Multidimensional keeps data on the disk. With the exception of distinct count calculations, you shouldn't expect a substantial performance improvement by just migrating from Multidimensional to Tabular. That's because Multidimensional caches query results in memory too. Moreover, the Windows operating system maintains

its own cache and keeps raw file data in memory. Consequently, chances are that the more memory you have on the server and the more end users query the cube, the more of its data will end up in memory. Aggregations, which are pre-calculated summaries of data in multidimensional cubes, can improve dramatically the performance of multidimensional cubes. Aggregations are not supported in Tabular. However, you might find better out-of-the-box performance with Tabular compared to a cube that isn't optimized. That's because VertiPaq has been specifically designed for retrieving data fast from in-memory data structures, which is not the case for cubes.

Both Multidimensional and Tabular support partitioning large tables to improve manageability. However, Tabular processes partitions within a table sequentially. As a result, processing large tables will take much longer with Tabular. Multidimensional has a strict division between dimensions and measures groups. When configuring incremental processing, the administrator must account for object dependencies. For example, fully processing a dimension invalidates the cube referencing the dimension. Processing dimensions with Process Update, drops indexes and aggregations for that dimension on related partitions so you have to restore them with Process Index. Processing is simplified in Tabular where all tables are treated the same. And, processing a table doesn't impact other tables.

VertiPaq compresses data by columns while Multidimensional OLAP (MOLAP) compresses data by rows. The typical VertiPaq compression ratio is five to ten times while the typical MOLAP compression ratio is about three times. However, the actual VertiPaq compression ratio depends on the data cardinality. For example, typically 99% of data in a data warehouse database is stored in the fact tables. Suppose you need to import most of the fact table columns, such as for end users to drill down to details. Chances are that will import high cardinality columns with many unique values, such as order number, transaction identifier, and so on. As I explained, high-cardinality columns will reduce the VertiPaq compression ratio, and you might not get a substantially better data footprint than Multidimensional.

The data acquisition capabilities of Tabular are impressive. It can import data from a variety of data sources, including relational databases, cubes, flat files, Excel files, Reporting Services R2 or later reports, and any application that exposes data as an Open Data Protocol (OData) feed, such as a SharePoint list. By contrast, Multidimensional is designed and works best when it sources data from a single database, such as a data warehouse database. Each model imports and caches unless the modeler configures the model in a pass-through (DirectQuery) mode for Tabular or Relational OLAP (ROLAP) for Multidimensional.

In cached mode (default), both models have to be explicitly processed to refresh the contained data. Tabular works best when all the data fits into memory. When estimating the Tabular memory requirements, you should account for at least twice the size of the disk footprint because additional memory is needed for refreshing data. Tabular models deployed to a dedicated Analysis Services server support basic paging, where the server pages memory to disk under memory pressure. Multidimensional has extensive paging support, and it's designed to scale to terabytes of data.

Choosing between Tabular and Multidimensional

To summarize, consider Tabular for simple analytical models where all or most of data fits into the server memory. Consider Multidimensional for more complex and enterprise-wide projects that can scale up to large data volumes. Here are a few examples to help you decide between Tabular and Multidimensional:

- Upgrading personal BI models – Suppose a business user has implemented a PowerPivot model and the model has been gaining popularity. You need to accommodate larger data volumes and secure its data. Tabular is a natural choice for this scenario. By deploying the model to Analysis Services in Tabular mode, you gain scalability and data security.

- Rapid development -- The marketing department has a new advertising effort that is scheduled for next week. They require some custom product categorizations, new dimensions that don't exist in the data warehouse and new attributes to existing dimensions, such as a custom hierarchy. The marketing department is testing three different kinds of advertising options. They'd like to compare data and analyze which of the three campaigns sold the best by grouping results by product. There is no time to develop ETL processes and enhance multidimensional cubes, especially given that this will be one-time analysis. This scenario is also a good candidate for Tabular thanks to its rapid development capabilities.
- Financial models – Suppose that you're tasked to implement an analytical layer for financial reporting. The requirements call for fairly complex business rules and aggregations, such as weighted averages and currency conversion. I recommend you use Multidimensional because it supports features specifically suited for such requirements, such as custom operators, scope assignments, and semi-additive functions.
- Data warehousing – A data warehouse database requires an analytical layer for historical and trend reporting. My preference is Multidimensional due to its maturity, richness, and ability to scale to terabytes of data.

1.4.2 Comparing Tabular and Report Builder Models

If you've used Reporting Services Report Builder models, then you might wonder how they compare with Tabular. Microsoft introduced the Report Builder technology in SQL Server 2005 as a feature of Reporting Services for business users to author ad hoc reports from pre-defined models. Report Builder consists of two components: a Report Builder model that is layered on top of a database and a Report Builder client that information workers can use for report authoring.

In SQL Server 2012, Report Builder models are deprecated and superseded by Tabular models. Report Model projects are no longer supported. Therefore, you can't create new Report Model projects or open existing Report Model projects in SQL Server Data Tools. Microsoft doesn't provide a migration path from Report Builder to Tabular. If you don't plan to upgrade Report Builder models, you need to keep Business Intelligence Development Studio (BIDS) from prior releases of SQL Server. **Table 1.5** provides a side-by-side feature comparison of Tabular and Report Builder models.

NOTE Microsoft deprecated the Report Builder models only. The Report Builder client is not deprecated and supports both tabular and multidimensional models. End users can use the Report Builder client to author ad hoc reports from Tabular just like they can do so from multidimensional cubes.

Data model

Unlike Tabular, Report Builder embraces the Object Role Modeling (ORM) methodology and represents database artifacts as entities, fields, and roles. One interesting feature that the Report Builder client supports is role navigation, where you can select only related entities in the report. Power View has a similar feature that inspects the table relationships defined in the model and that prevents the user from selecting columns from unrelated tables.

In general, the Report Builder supports only regular relationships where tables are directly related to each other. It also supports more advanced relationship features that don't have Tabular equivalents, including entity inheritance (for example, a sales person is an employee), role expansion (adding fields from the Contact entity to the Employee entity to expand employees with contact details), and lookup entities (bringing entities from another lookup table, such as adding (denormalizing) a product category from the Product Category entity to the Product entity).

Another Report Builder-specific feature is infinite clickthrough, which gives the modeler the ability to define drillthrough paths in the model. Then, the end user can click any clickthrough-enabled fields to see the details of the related entity. The closest Tabular equivalent of this feature is the default drillthrough action, although it's isolated to a single table only.

Table 1.5 Comparing Tabular and Report Builder models

Layer	Feature	Tabular Model	Report Builder Model
Data Model	Schema	Relational (tables and columns)	Semantic (entities, fields, roles)
	Declarative relationships	Regular, role-playing	Regular, entity inhertiance, role expansion, lookup entities
	End-user model	KPIs, perspectives, actions (drillthrough)	Clickthrough (requires Enterprise edition)
	Aggregation functions	Sum, Count, Min, Max, Average, DistinctCount	Sum, Count, Min, Max, Average, DistinctCount
Business Logic	Expression language	DAX	Visual Basic functions
	Constructs	Calculated columns, calculated measures	Named calculations
Data Access	Storage type	In-memory cache and basic passthrough	Passthrough
	Data sources	Relational, multidimensional, flat files, OData, SSRS reports	SQL Server, Oracle, Analysis Services, Teradata
	Storage modes	VertiPaq, DirectQuery	Pass-through queries

Business logic

The business logic capabilities of Report Builder are limited to a subset of Visual Basic functions that can be used to define custom expressions, such as aggregate functions. DAX is a more powerful query language. The Report Builder business logic constructs are limited to defining calculations in the data source view. This works in the same way as it does with Multidimensional.

Data access

The most important difference between the two models in terms of data access is that Report Builder doesn't cache data. Instead, it auto-generates queries to the source database in a pass-through mode. I mentioned that, by default, Tabular caches data for the best possible performance. Since data is cached, you need to explicitly process the model to synchronize the data changes. On the upside, Tabular performs much better when aggregating larger data volumes.

Report Builder supports SQL Server, Analysis Services, Oracle, and Teradata databases only. By contrast, as long as data is accessible, Tabular can retrieve data from virtually anywhere. Finally, Report Builder doesn't store data. It simply receives the report from the client and auto-generates queries to the supported data sources. By contrast, Tabular caches data by default.

In summary, if you decide to replace your Report Builder models with tabular models, you'll undoubtedly find that Tabular is a richer and more efficient model. Unfortunately, migrating to Tabular will require a complete rewrite. If this isn't an option, you can still continue using your Report Builder models with SQL Server 2012. Although you must use Business Intelligence Development Studio (BIDS) to make changes to the model, you can deploy the model to SQL Server 2012 Reporting Services.

1.5 Summary

This chapter has been a whirlwind tour of the innovative Business Intelligence Semantic Model (BISM) and its features. By now, you should view BISM as a flexible platform that meets a variety of BI requirements. A component of SQL Server 2012 Analysis Services, BISM is a collective name of Multidimensional and Tabular models. You've learned about the history of Analysis Services and how it fits into the Microsoft BI stack. We took a close look at the BISM logical architecture and its data model, business logic, and data access layers.

Next, this chapter focused on Tabular. I explained the events that led to the arrival of Tabular and its design objectives to help you create BI solutions that are simpler to implement and that perform well. These solutions can span the entire spectrum of personal, team, and organizational BI.

Readers familiar with multidimensional cubes saw how Tabular and Multidimensional compare and how to choose between them. This chapter also provided a feature comparison between Tabular and Report Builder models.

Having laid the foundation, we are ready to put our knowledge to use. Let's continue our journey by exploring personal BI with PowerPivot.