

Microsoft[®] SQL Server[™] 2005 Analysis Services Step by Step

*Reed Jacobson; Stacia
Misner; Hitachi Consulting*

To learn more about this book, visit Microsoft Learning at
<http://www.microsoft.com/MSPress/books/8592.aspx>

9780735621992
Publication Date: March 2006

Microsoft
Press

Table of Contents

Introduction	ix
Finding Your Best Starting Point	ix
About the Companion CD-ROM	x
System Requirements	xi
Installing and Using the Sample Files	xi
Conventions and Features in This Book	xii
Part I Getting Started with Analysis Services	
1 Understanding Business Intelligence and Data Warehousing	3
Introducing Business Intelligence	3
Reviewing Data Warehousing Concepts	5
The Purpose of a Data Warehouse	5
The Structure of a Dimensional Database	6
A Fact Table	10
Dimension Tables	11
Chapter 1 Quick Reference	16
2 Understanding OLAP and Analysis Services	17
Understanding OLAP	17
Consistently Fast Response	18
Metadata-Based Queries	20
Spreadsheet-Style Formulas	22
Understanding Analysis Services	23
Analysis Services and Speed	24
Analysis Services and Metadata	24
Analysis Services Formulas	26
Analysis Services Tools	28
Chapter 2 Quick Reference	29
3 Building Your First Cube	31
Exploring Business Intelligence Development Studio	31

What do you think of this book? We want to hear from you!	Microsoft is interested in hearing your feedback about this publication so we can continually improve our books and learning resources for you. To participate in a brief online survey, please visit: www.microsoft.com/learning/booksurvey/
---	---

- Examining the Contents of an Analysis Services Project 32
- Exploring Menu Commands 35
- Preparing to Create a Cube 36
 - Reviewing the Analysis Requirements 37
 - Creating a New Analysis Services Project 37
- Creating a Cube 38
 - Using the Cube Wizard Without a Data Source 38
 - Reviewing the Cube Structure in the Cube Designer. 45
- Generating a Schema 47
 - Using the Schema Generation Wizard 47
 - Loading Data into the Relational Schema 52
- Processing and Browsing a Cube 55
 - Deploying and Processing a Cube 55
 - Browsing a Cube 56
- Chapter 3 Quick Reference. 58

Part II Design Fundamentals

- 4 Designing Dimensions 63**
 - Reviewing the Data Warehouse Structure 63
 - Building a Standard Dimension 64
 - Adding a Data Source. 65
 - Creating a Data Source View 67
 - Using the Dimension Wizard 69
 - Deploying a Dimension 74
 - Changing Attribute Properties 76
 - Working with a Time Dimension. 77
 - Modifying a Data Source View 78
 - Creating a Time Dimension 79
 - Working with Role-Playing Dimensions. 84
 - Creating a Parent-Child Dimension. 85
 - Adding an Employee Dimension. 86
 - Totaling Data for Non-Leaf-Level Data Members 88
 - Managing Levels within a Parent-Child Dimension 92
 - Chapter 4 Quick Reference. 96
- 5 Designing Measure Groups and Measures. 99**
 - Adding Measure Groups to a Cube 99

	Building a Cube	100
	Changing Properties for Measure Groups and Measures	103
	Specifying Dimension Usage	104
	Browsing Multiple Measure Groups	107
	Aggregating Semiadditive Measures.	113
	Adding a Measure Group to an Existing Cube	113
	Using a Semiadditive Aggregate Function.	115
	Calculating Distinct Counts.	117
	Creating Simple Calculations	119
	Adding a Calculation to a Cube.	120
	Applying Conditional Formatting	126
	Chapter 5 Quick Reference	127
6	Working with a Finance Measure Group	129
	Designing an Account Dimension	129
	Working with Account Intelligence.	130
	Using Unary Operators	135
	Aggregating by Account.	139
	Designing Nonadditive Financial Measures	144
	Creating a Nonadditive Measure.	145
	Chapter 6 Quick Reference	148
7	Designing Aggregations and Hierarchies.	149
	Understanding Aggregation Design	149
	Using the Aggregation Design Wizard	151
	Inspecting Aggregations.	155
	Changing Partition Counts	158
	Adding Attributes to the Aggregation Design	160
	Designing User Hierarchies.	161
	Adding a User Hierarchy.	162
	Aggregating User Hierarchies	165
	Optimizing Aggregations	167
	Using the Query Log	168
	Viewing Usage Data.	170
	Using the Usage-Based Optimization Wizard	171
	Maintaining the Query Log	172
	Chapter 7 Quick Reference	173

Part III Advanced Design

8	Using MDX	177
	Creating Tuple-Based Calculated Members	177
	Creating an MDX Calculation for Percent of Total	182
	Creating an MDX Calculation for Percent of Parent	186
	Querying with MDX	188
	Executing MDX Queries	188
	Working with Basic MDX Queries	193
	Designing Custom Members	197
	Creating a Calculated Member Using a Set-Based Function	197
	Creating Cumulative Calculations	200
	Working with MDX Scripts	202
	Managing the Sequence of Calculations	202
	Adding a Script Assignment	205
	Developing Key Performance Indicators	209
	Comparing Cube Values to Goals	209
	Using MDX Expressions with Key Performance Indicators	212
	Chapter 8 Quick Reference	215
9	Exploring Special Features	217
	Defining Dimension Relationships	217
	Using a Referenced Relationship Type	217
	Using a Many-to-Many Relationship Type	221
	Supporting Currency Conversions	229
	Localizing Cubes	231
	Adding Translations	231
	Browsing Translations	235
	Organizing Information with Folders and Perspectives	236
	Organizing Measures	236
	Using Perspectives	238
	Chapter 9 Quick Reference	242
10	Interacting with Cubes	245
	Implementing Actions	245
	Using Standard Actions	246
	Linking to Reports	249
	Adding Drillthrough	251

Using Writeback	253
Write-Enabling a Dimension	254
Dynamically Adding Members to a Dimension.....	255
Modifying the Cube Structure for Writeback	257
Writing Values Back to a Cube	261
Chapter 10 Quick Reference.....	267

Part IV Production Management

11	Implementing Security	271
	Using Role-Based Security.....	271
	Creating Security Roles	272
	Managing Roles	277
	Applying Security to a Dimension	278
	Restricting Access to a Dimension.....	278
	Restricting Access to Specific Members of a Dimension	281
	Controlling Visual Totals for a Dimension	283
	Defining a Default Member for a Dimension	284
	Securing Data at the Cell Level.....	287
	Preventing Values in Cells from Being Read.....	287
	Allowing Users to Write to Cells.....	290
	Setting Administration Security	291
	Creating Security Roles for Processing	291
	Chapter 11 Quick Reference.....	293
12	Managing Partitions and Database Processing.....	295
	Managing Very Large Databases	295
	Understanding Partition Strategies.....	295
	Creating Partitions	296
	Merging Partitions	301
	Working with Storage	304
	Understanding Analysis Services Storage Modes	305
	Setting Storage Options	306
	Changing Data in a Warehouse.....	308
	Managing OLAP Processing	312
	Processing a Dimension	313
	Processing a Cube	318
	Configuring Proactive Caching	320

- Monitoring Cube Activity 326
 - Profiling Analysis Services Queries 326
 - Using the Performance Monitor 330
- Chapter 12 Quick Reference 333
- 13 Managing Deployment 335**
 - Reviewing Deployment Options 335
 - Building a Database 336
 - Deploying a Database 341
 - Processing a Database 348
 - Managing Database Objects Programmatically 351
 - Working with XMLA Scripts 352
 - Automating Database Processing 356
 - Creating a SQL Server Integration Services Package 357
 - Using the Analysis Services Processing Task 358
 - Handling Task Failures 359
 - Scheduling a SQL Server Integration Services Package 361
 - Planning for Disaster and Recovery 364
 - Backing Up an Analysis Services Database 365
 - Restoring an Analysis Services Database 366
 - Chapter 13 Quick Reference 368
- Glossary 369**
- Index 373**

What do you think of this book?
We want to hear from you!

Microsoft is interested in hearing your feedback about this publication so we can continually improve our books and learning resources for you. To participate in a brief online survey, please visit: www.microsoft.com/learning/booksurvey/

Chapter 4

Designing Dimensions

After completing this chapter, you will be able to:

- Define a data source for an Analysis Services project.
- Create a data source view (DSV).
- Use the Dimension Wizard to build standard, time, and parent-child dimensions.
- Deploy Analysis Services database objects.
- Change dimension properties.
- Work with special properties of parent-child dimensions.

In Chapter 3, “Building Your First Cube,” you created dimensions from a template and then used the Schema Generation Wizard to build tables in a Structured Query Language (SQL) Server database to correspond to the dimension design. In this chapter, you’ll use the Dimension Wizard to create dimensions from existing tables and work with the Dimension Designer to review a dimension’s structure, modify its properties, and browse its data. You’ll work with three different kinds of dimensions—standard, time, and parent-child—to become familiar with the range of options available to you when designing and building dimensions.

Reviewing the Data Warehouse Structure

The dimensions and the cube you’ll create in this chapter and in Chapter 5, “Designing Measure Groups and Measures,” are based on a simple Microsoft SQL Server database supplied on this book’s companion CD. The database file (SSAS Step by Step DW.mdf) is an extremely simplified version of the Adventure Works DW database that ships with Microsoft SQL Server 2005, with a few modifications to aid your exploration of Analysis Services. The SSAS Step by Step DW database is small enough that you should be able to clearly understand its structure, but it illustrates the common dimension types that are the focus of this chapter.

The SSAS Step by Step DW database contains data for a fictitious manufacturing company called Adventure Works. It is designed as a small data warehouse to support analysis of products, customers, employees, financial accounting, sales, and much more. The following table

provides a brief description of the tables that you'll be using to create dimensions in this chapter and to create measures in Chapter 5:

Table Name	Description
DimEmployee	Personal information for all employees: Employee Names, Addresses, Titles, Managers, Hire Date, etc.
DimProduct	Product information: Name, Cost, Color, Size, etc.
DimProductSubcategory	Classifications of products within categories: Bikes can be Mountain Bikes, Road Bikes, or Touring Bikes. Accessories can be Pedals, Chains, Wheels, etc. Clothing can be Gloves, Jerseys, Shorts, etc.
DimProductCategory	Major classifications of products: Bikes, Accessories, Clothing, and Components.
DimTime	Contains date-related data for the business: Year, Quarter, Month, Date, etc.
FactInternetSales	Transaction details for sales made through an Internet e-commerce site.
FactResellerSales	Transaction details for sales made through the reseller's channel.

The two fact tables, FactResellerSales and FactInternetSales, both have some dimension keys in common—ProductKey, OrderDateKey, DueDateKey, and ShipDateKey. They also have two measures in common—SalesAmount and OrderQuantity. ProductKey is joined to a chain of dimension tables (snowflake schema): DimProduct, DimProductSubcategory, and DimProductCategory. OrderDateKey, DueDateKey, and ShipDateKey all join to a single DimTime table. FactResellerSales also has a dimension key, EmployeeKey, that is joined to the DimEmployee dimension table (parent-child). You'll work with other tables in the database in later chapters as you focus on special design features of Analysis Services.

Building a Standard Dimension

A cube must contain at least one dimension. Unless you are building a cube from a template, as you did in Chapter 3, you will probably create most of your dimensions before creating the cube. Standard dimensions are the most common kind of dimension in a cube. Time and parent-child dimensions have additional properties to address specific analytical requirements, which you'll learn about later in this chapter.

Before you get started with your first dimension, you'll need to create a data source so that Analysis Services can connect to the data warehouse. You'll also need to create a data source view (DSV) so that Analysis Services knows which tables from the data warehouse contain the data to load into the dimension. Once you have a data source and a DSV in place, you're ready to build dimensions. When you build a standard dimension, you can use a single table (star schema) or you can use multiple related tables (snowflake schema) from the data warehouse.

Adding a Data Source

One of the first tasks in building a cube from the bottom up is to add a data source. The data source is the connection information that Analysis Services uses to connect to the database that hosts the data. The data source contains the connection string which specifies the server and the database hosting the data as well as any necessary authentication credentials.

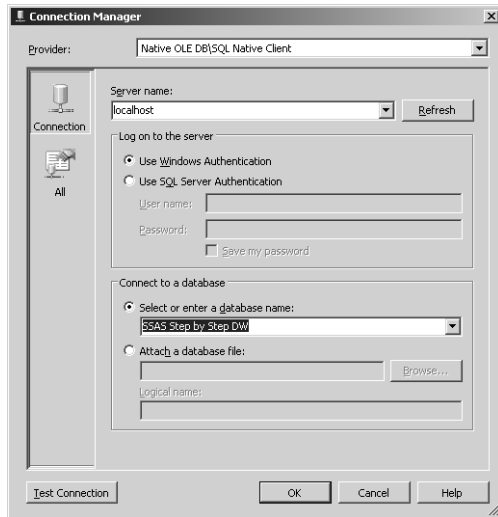
The data source can be a .NET provider or a native OLE DB provider. OLE DB is an industry-standard technology that is a generalized replacement for the Open Database Connectivity (ODBC) standard used for many years. When accessing data from Microsoft SQL Server 2005 relational databases, you should use the SQL Native Client, which is the SQL OLE DB and SQL ODBC providers rolled into one with some added functionality to support new SQL Server 2005 features. This provider type is selected by default when you create a new data source for your Analysis Services projects.

In this procedure, you'll create a data source that defines a connection to the SSAS Step by Step DW database in your local SQL Server using Microsoft Windows authentication.

Add a data source to an Analysis Services project

1. If necessary, start SQL Server Business Intelligence Development Studio (BIDS).
2. On the File menu, point to New, and then click Project.
3. In the New Project dialog box, click the Analysis Services Project template in the Templates pane.
4. Type a name for the project: **SSAS Step by Step**.
5. If necessary, change the location for the project to **C:\Documents and Settings \<username>\My Documents\Microsoft Press\as2005sbs\Workspace**, and click OK.
6. In Solution Explorer, right-click the Data Sources folder, click New Data Source, and then click Next.
7. Click the New button to create a new connection.
8. In the Connection Manager dialog box, type a server name: **localhost**.
9. In the Select Or Enter A Database Name list box, select SSAS Step by Step DW.

The Connection Manager dialog box looks like this:

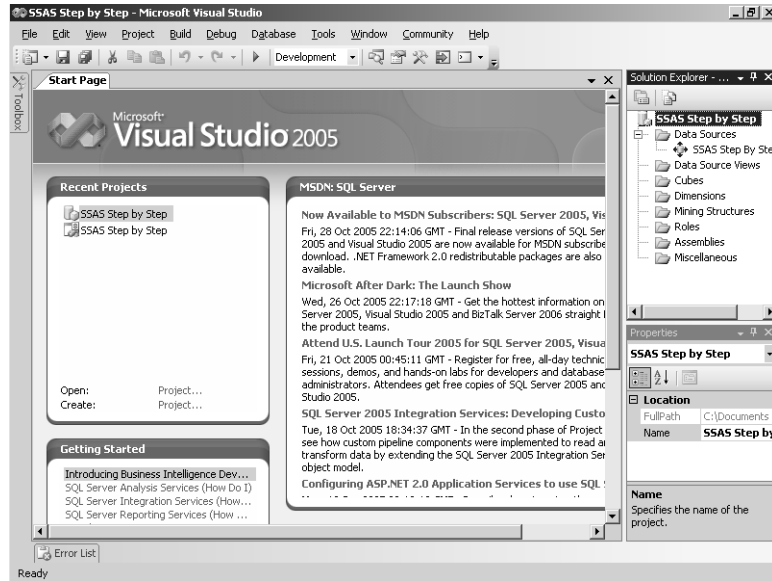


10. Click Test Connection and then click OK to close the message box.
11. Click OK to close the Connection Manager dialog box.
12. Click Next.
13. On the Impersonation Information page of the Data Source Wizard, click Use The Service Account and click Next.

Here you can see that the default data source name matches the name of the database that you selected in the Connection Manager dialog box. It's generally recommended to keep the data source name in your project consistent with the name of the database so that anyone reviewing the project files can tell at a glance where to find the source data for cube.

14. To complete the wizard, click Finish.

The Solution Explorer window looks like this:



Notice that the Data Sources folder now contains a data source Step item, SSAS Step by Step DW.ds. You can change the name of the data source at any time, but you must retain the .ds file extension in the name of this item or the data source will not be recognizable by the Dimension Designer and Cube Designer in Visual Studio.

15. Save the solution.

Creating a Data Source View

In Chapter 2, “Understanding OLAP and Analysis Services,” you were introduced to the concept of the Unified Dimensional Model (UDM) that is the centerpiece of an online analytical processing (OLAP) solution. The UDM provides an intermediate logical layer between the physical relational database that is used as a data source and the proprietary cube and dimension structures that are used to resolve user queries. There are at least four good reasons for creating a UDM for analysis instead of allowing users to directly access data in the data sources:

1. Manipulating data sources is easier for system administrators and developers, and more difficult for business users. With a UDM, the initial setup of data sources is left to system administrators and developers so that business users can focus on analyzing data.
2. Companies often use multiple databases, systems, files, and services to store data. You spare each business user the effort of locating all the pertinent data and systems, and figuring out how to work with it, by combining this disparate data into a single UDM.

3. Data must be consolidated into a unified format and summarized across systems. If you don't already have a data warehouse, a UDM can help you present the data in a unified and summarized manner.
4. The business user must research the business rules associated with each system and apply them consistently. Using a UDM, you can define those business rules in one place for consistent application.

An important component of the UDM in Analysis Services is the DSV. Because the scope of analysis within a single cube can be more limited than the scope of a data warehouse, you can pick and choose the data that you need for the dimensions and for the cube by creating a subset view of the data warehouse. This subset view is called a DSV. Selecting objects to study from the data warehouse is like aiming a telescope at the sky and selecting only the celestial objects within its scope to analyze. Further, just as you can aim your telescope at a different section of the sky, you can aim the scope of your DSV at a different section of the data. You can also add and remove tables as you change the features built into your cube, which you will learn how to do throughout the chapters of this book.

In this procedure, you'll create a DSV that includes the DimProduct, DimProductSubcategory, and DimProductCategory tables.

Use the Data Source View Wizard

1. In Solution Explorer, right-click the Data Source Views folder, click New Data Source View, and then click Next.
2. Click SSAS Step by Step DW to select the relational data source, and then click Next.
3. Double-click dbo.DimProduct to add it to the Included Objects list.

Alternatively, you can click on a table and then click the arrow pointing to the right (which looks like a greater-than sign) to move it to the Included Objects list. You can select multiple tables by pressing the Ctrl key while clicking on each table.

4. Repeat the previous step to add dbo.DimProductCategory and dbo.DimProductSubcategory to the Included Objects list.

There is an existing relationship defined in the data warehouse among these three tables. Products are organized into Product Subcategories and Product Subcategories are organized into Product Categories. For example, note the relationships in the following table:

Product Category	Product Subcategory	Product
Bikes	Mountain Bikes	Mountain-100 Silver, 38
Bikes	Mountain Bikes	Mountain-100 Black, 44
Bikes	Road Bikes	Road-150 Red, 62
Bikes	Road Bikes	Road-650 Black, 52
Bikes	Touring Bikes	Touring-2000 Blue, 60
Bikes	Touring Bikes	Touring-1000 Yellow, 46

You will build a dimension in the next procedure that uses data from these three tables, so all three tables must be included in the DSV.

In addition, you could add a Named Query to the table to reference data from these three tables without adding them explicitly to the DSV. A Named Query is analogous to a view in a relational database. You create a Transact-SQL statement to define the tables and columns that you want to combine into a single object. You need to create the data source view with at least one object before you can add a Named Query to it.

5. Click Next, and then click Finish.

You have the option to change the default DSV name before completing the wizard.

6. Save the solution.

Using the Dimension Wizard

With the data source and the DSV added to the project, Analysis Services now has access to the underlying data that will be loaded into your dimension. The next step in building a standard dimension is to use the Dimension Wizard to create the initial structure of the dimension. When using the Visual Studio development environment, the only way to build a new dimension is to use this wizard.

The Dimension Wizard is a flexible tool that will guide you through the steps required to create the dimension. First, you choose whether or not to use an underlying data source. The next step is to define the dimension type. If the dimension will be based on a database table, then the next steps are to specify the main table of the dimension, identify the key column and name column for the leaf-level attribute, select attributes for inclusion in the dimension, and associate related tables with the dimension.

In this procedure, you'll create a Product dimension that is based upon the DimProduct, DimProductSubcategory, and DimProductCategory tables in the DSV.

Create a standard dimension

1. In Solution Explorer, right-click the Dimensions folder, click New Dimension, and then click Next.

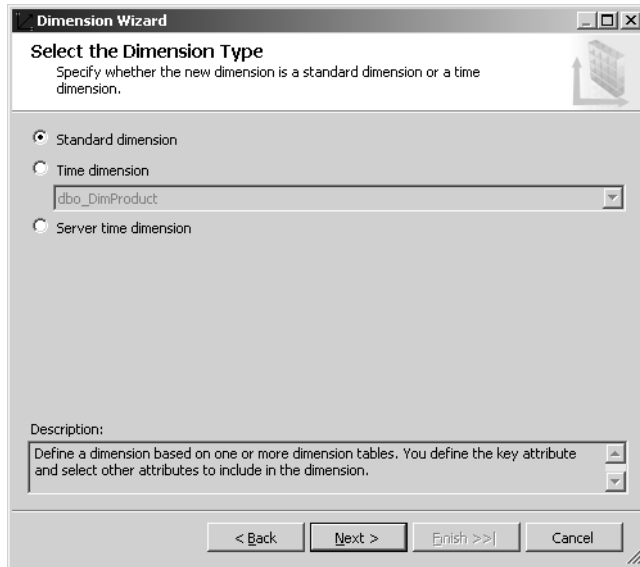
If you choose to build the dimension with an underlying database and to use Auto Build, the Dimension Wizard will analyze the data in the source tables and create attributes and hierarchies for you. If you prefer more control over the structure of your dimension, you should disable Auto Build. On the other hand, if your goal is to build a prototype cube, building the dimension without an underlying data source will allow you to work out a structure and get a model quickly built. Later, you can load data into it. As you saw in Chapter 3, you can even use templates to help you build the model. For now, you will build a dimension manually to learn how the process works and to give you greater control over the structure of the new dimension as you work through this chapter.

2. Clear the Auto Build check box, and then click Next.
3. Click the SSAS Step by Step DW data source view.

You can review the diagram for the DSV if you click Browse on this page of the wizard. This feature is helpful when you have multiple DSV in a project and want to be sure that you select the correct one.

4. Click Next.

The Select The Dimension Type page of the Dimension Wizard looks like this:



You'll use the default option, Standard Dimension, in this procedure. You'll learn more about the options to create a Time Dimension and a Server Time Dimension later in this chapter.

5. Click Next.

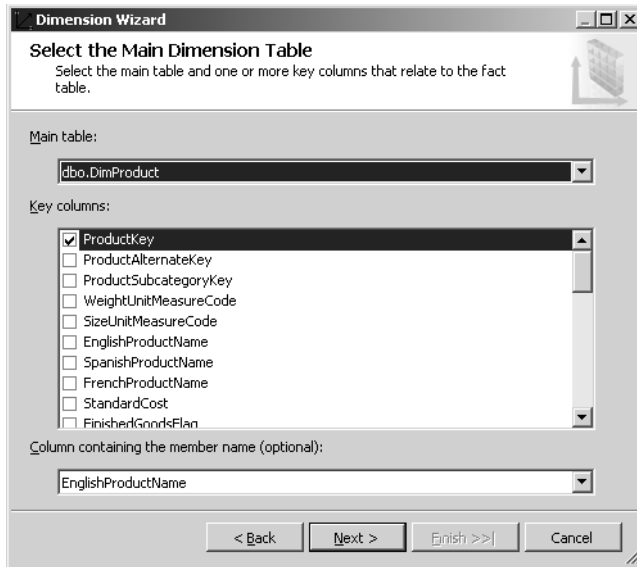
The Main Table defaults to `dbo.DimProduct` since it is first alphabetically in the list of tables contained in the DSV. When necessary, you can choose a different table from the list to represent the leaf level of the dimension. In this case, the `DimProduct` table contains the most granular level of detail required for this dimension, so you will keep this default selection for Main Table.

On this page of the wizard, you'll also need to select the key column(s) that uniquely identify each dimension member. For this dimension, `ProductKey` is unique for each row in the `DimProduct` table. As a default, a dimension uses the same column for the member keys and for the member names, but you can also specify a name for each member that's separate from the member key. A member name is the text that appears in a heading row or column on a report, so you should make these names meaningful and

descriptive. Because ProductKey is not useful for reporting, you should designate EnglishProductName as the member name column.

6. In the Key Columns list, select ProductKey, and then select EnglishProductName in the Column Containing The Member Name drop-down list.

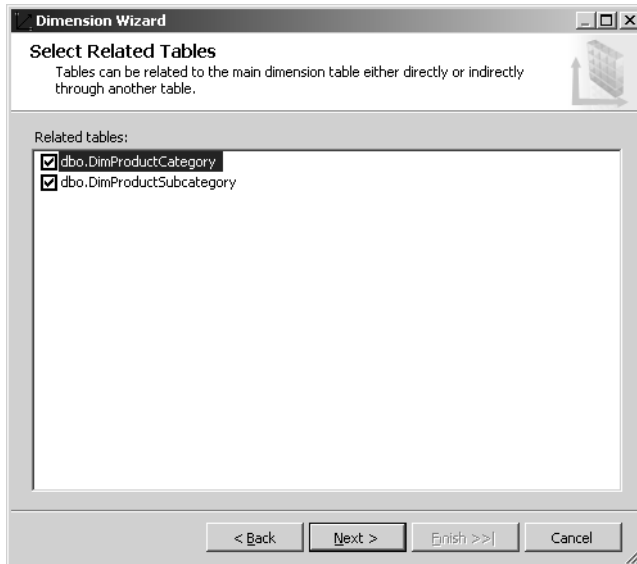
The current page of the wizard looks like this:



The screenshot shows the 'Dimension Wizard' window with the title 'Select the Main Dimension Table'. Below the title is the instruction: 'Select the main table and one or more key columns that relate to the fact table.' The 'Main table:' dropdown menu is set to 'dbo.DimProduct'. The 'Key columns:' list contains several items, with 'ProductKey' selected (checked). Other items include ProductAlternateKey, ProductSubcategoryKey, WeightUnitMeasureCode, SizeUnitMeasureCode, EnglishProductName, SpanishProductName, FrenchProductName, StandardCost, and FinishedGoodsFlag. The 'Column containing the member name (optional):' dropdown menu is set to 'EnglishProductName'. At the bottom are buttons for '< Back', 'Next >', 'Finish >>', and 'Cancel'.

7. Click Next.

The Select Related Tables page of the wizard looks like this:



The screenshot shows the 'Dimension Wizard' window with the title 'Select Related Tables'. Below the title is the instruction: 'Tables can be related to the main dimension table either directly or indirectly through another table.' The 'Related tables:' list contains two items, both selected (checked): 'dbo.DimProductCategory' and 'dbo.DimProductSubcategory'. At the bottom are buttons for '< Back', 'Next >', 'Finish >>', and 'Cancel'.

The `dbo.DimProductCategory` and `dbo.DimProductSubcategory` tables are already selected as related tables because these two tables are joined to the main table, `dbo.DimProduct`, in the selected DSV. Because you want to include attributes from these two tables in the dimension, you should accept the default selection of these tables on this page.

8. Click Next, and then, on the Select Dimension Attributes page of the wizard, select the check box next to the following attributes: Color, List Price, and Size.

All columns from the main table and related tables are available for selection as dimension attributes. You can choose all columns from all tables, or you can select a few of the available columns.

9. Select the check box to the left of Dim Product Subcategory, and then, in the same row, click `dbo.DimProductCategory.EnglishProductSubcategoryName` in the Attribute Name Column drop-down list.



Note You may need to resize the wizard dialog box or the width of the columns (or both) to be able to view the names in the list box.

Color, List Price, and Size do not have separate key and name columns like Product, Product Category, and Product Subcategory. When there are separate columns in the dimension table to represent the unique key of an attribute and its user-friendly name, you should adjust the Attribute Key Column or Attribute Name Column as necessary on this page of the wizard. Be careful to select the correct column!

You can also change the name of the attribute on this page of the wizard. For example, you probably don't want users to see attributes named Dim Product Category or Dim Product Subcategory. You can click in the Attribute Name column and type in a more appropriate name here. However, since you will learn an alternative method for fixing the attribute name later in this chapter, you can keep the default attribute names here.

10. Select the check box to the left of Dim Product Category, and then, in the same row, click `dbo.DimProductCategory.EnglishProductCategoryName` in the Attribute Name Column drop-down list.
11. Click Next.

On the Specify Dimension Type page of the wizard, the default dimension type is Regular. You'll learn more about specifying dimension types in Chapter 6, "Working with a Finance Measure Group." Some dimension types are used by client applications that query Analysis Services, while other dimension types direct the behavior of a dimension, such as applying special aggregation rules for an accounts dimension. Keep the default setting for this dimension.

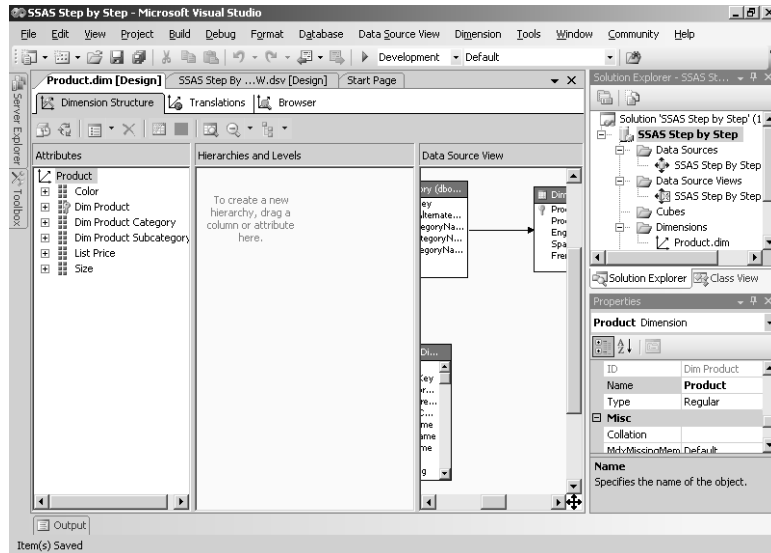
12. On the Specify Dimension Type page, click Next.

13. On the Define Parent-Child Relationship page, click Next.

Since you're not currently building a parent-child dimension, you don't need to change the Define Parent-Child Relationship page of the wizard. Later in this chapter, you will learn more about working with this page of the wizard.

14. Change the name of the dimension from Dim Product to **Product**, and then click Finish.

Your screen looks like this:



After completing the dimension wizard, you'll see Product.dim as an object in the Dimensions folder of the Solution Explorer window. In the main window of Visual Studio, you see the Dimension Designer. The Dimension Designer has three tabs—Dimension Structure, Translations, and Browser.

On the Dimension Structure tab, you can review the list of attributes for the dimension as well as view a diagram of the tables from the DSV that were used to create the dimension. Notice also the Properties window in the lower right corner. The Properties window displays all properties for the object currently selected in the main window, which is the dimension object in the Attributes pane when you first finish the wizard.

15. In the Attributes pane of the Dimension Structure tab, right-click Dim Product, click Rename, and then type **Product**.
16. Repeat the previous step to rename Dim Product Category as **Category**, and Dim Product Subcategory as **Subcategory**.

Alternatively, you can change the *Name* property of an attribute which you will do later in this chapter.

17. Save the solution.

Deploying a Dimension

Although the Dimension Structure tab of the Cube Designer is useful for reviewing the attributes of a dimension and their properties, you can't browse the members of a dimension here. Instead, you use the Browser of the Dimension Designer to review the members of a dimension within their respective attribute hierarchies. You should browse a dimension early in the development cycle to make sure you get the right columns selected for the member name column and that the sort order of members is correct.

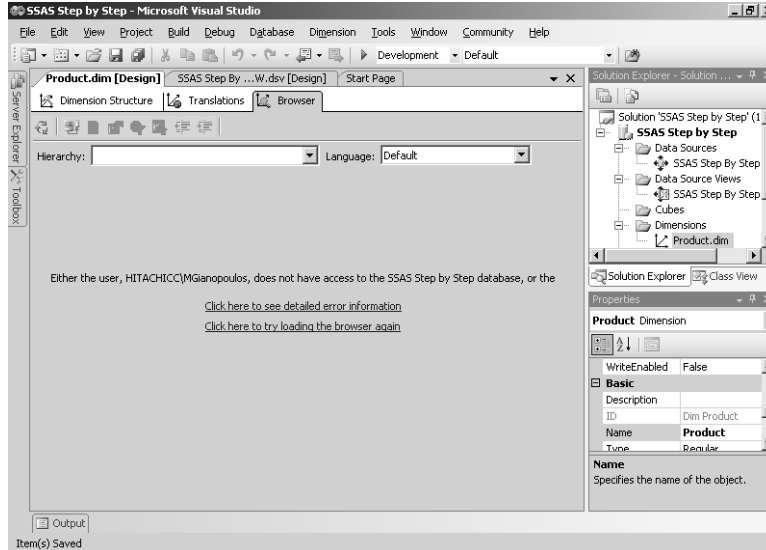
However, right now your dimension is defined only in an XML file, Product.dim, that is contained in an Analysis Services project on your workstation. If you try to browse the dimension now, you will get an error message because the dimension must first be deployed to the Analysis Server. Deploying the dimension to the server copies your local XML file to the server's data folder, creates a dimension object on the server, and then processes the dimension to load data into the new object.

In this procedure, you'll deploy the SSAS Step by Step solution to the local Analysis Server and browse the processed Product dimension.

Deploy a dimension to the Analysis Server

1. Click the Browser tab.

Your screen looks like this:



Because the dimension has not yet been deployed to the Analysis Server, it is unavailable for browsing.

2. Right-click the SSAS Step by Step project in the Solution Explorer window, and then click Deploy.

The default target server for deployment is the local server.

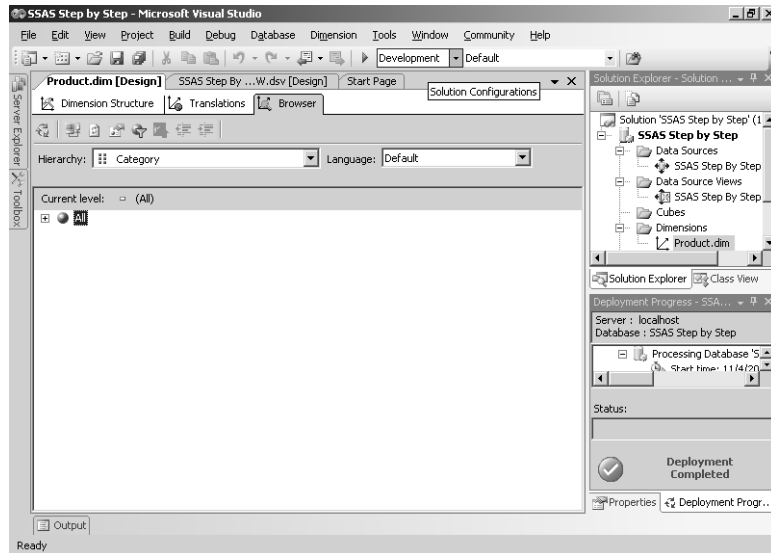


Tip If you want to deploy to a different server, you should first right-click on the SSAS Step by Step project in the Solution Explorer window, click Properties, click Deployment in the Configuration Properties tree, and then change the *Server* property to the correct target server.

The Deployment Progress window displays the current status of the operation.

- When deployment is complete, click the Reconnect button in the Browser toolbar.

Your screen now looks like this:



Each time that you make changes to objects in an Analysis Services project and deploy those changes to the server, you must use the Reconnect button in the Browser tab of a designer if you have previously attempted to browse a database object.

- In the Hierarchy drop-down list, select Category.

Each attribute is available for selection in the Hierarchy drop-down list. By default, an attribute defined for a dimension results in the creation of an attribute hierarchy. Attribute hierarchies are useful for grouping information for analysis. For example, in this dimension, you can group information about products by category, by subcategory, or by individual product. You can also group product information by color or by size. When you select an attribute hierarchy in the browser, you can view the members of that attribute in a two-level hierarchy. The top level of each hierarchy is the All member, which is used to display the aggregated value of the attribute members for each measure when viewed in a cube.

5. Click the plus sign to the left of the All member in the Category hierarchy tree.
6. Click Subcategory in the Hierarchy drop-down list, and then expand All to view the members of this attribute hierarchy.
7. Repeat the previous step to view the members of each remaining hierarchy.

Changing Attribute Properties

You can modify attribute properties when you want to change the appearance of members within an attribute hierarchy. For example, you can use the *Order* property to control the sort sequence of members, or you can specify a different member name column for an attribute by changing its *NameColumn* property. Some properties control how an attribute functions, such as the *Type* and *Usage* properties, which you'll learn more about in Chapter 6. Other properties that you'll learn about later in this chapter and in Chapter 7, "Designing Aggregations and Hierarchies," such as the *IsAggregatable* and *AggregationUsage* properties, affect how an attribute hierarchy is aggregated (or not).

After you create a dimension, one of the first properties that you should consider changing is *AttributeHierarchyEnabled*. Every attribute that you add to a dimension becomes an attribute hierarchy, as you saw in the previous procedure. When users browse a cube, they select attribute hierarchies for grouping measure values, so you should be careful about which attribute hierarchies are available for grouping data. In the current dimension, for example, grouping by Category, Subcategory, Product, Color, or Size seems logical and desirable. However, grouping by List Price is probably not useful. Typically, an attribute like this, which can have a high number of distinct members relative to the leaf-level attribute, is used for filtering purposes or for detailed reporting.

In this procedure, you'll disable the *AttributeHierarchyEnabled* property of the List Price attribute.

Disable the *AttributeHierarchyEnabled* property

1. Click the Dimension Structure tab, right-click the List Price attribute, and then click Properties.
2. In the Properties window, scroll to the *AttributeHierarchyEnabled* property, and then select False in the property's drop-down list.

Notice that the icon in the Attributes pane next to List Price is now gray, while all the other icons remain blue. The gray icon in the Attributes pane indicates that the attribute hierarchy is disabled for this attribute.

3. Right-click the SSAS Step by Step project in the Solution Explorer window, and then click Deploy.

Each time you want to view the results of a change that you have made to a dimension, you must deploy the project to the server before you can browse the dimension.

4. When deployment is complete, click the Reconnect button in the Browser toolbar.
Notice that List Price is no longer available as a hierarchy in the Hierarchy list box.

The Unknown Member

Oftentimes when you are dealing with relational databases, data integrity issues will be a problem, especially when you are in a prototyping stage of development or when you are building a cube from a source that is not a standard data warehouse. In these situations, it's possible that a record referencing a new dimension record is loaded into a fact table prior to the addition of the new dimension record to the dimension table. Or perhaps the cube is processed before the dimension is processed. You'll learn more about the challenges of managing new data in fact and dimension tables in Chapter 12, "Managing Partitions and Database Processing."

Normally, if you try to process a cube from a fact table that contains a dimension key that is not yet found in the dimension, the processing operation will fail. However, when you are building a prototype or when your solution requires near real-time information, you may want to ignore this type of error and allow processing to continue. By enabling the *UnknownMember* property of a dimension and by setting the value of the *KeyErrorAction* property of a cube to *ConvertToUnknown*, you can avoid processing errors when a fact table contains a missing or invalid key for that dimension. The aggregated values of each measure in the cube will consequently always equal the corresponding aggregated values in the fact table as a result.

The *UnknownMember* property has three possible values: Visible, Hidden, and None. The default setting for the *UnknownMember* property is Visible, so you'll notice a member named Unknown in each attribute hierarchy of a new dimension, even though you don't have a record with that name in your dimension table. If you prefer a different name for this member, set the value of the *UnknownMemberName* property.

Working with a Time Dimension

There's practically no such thing as an OLAP database without a Time dimension. Often, a Time dimension contains months as the lowest level of detail—aggregated into quarters and years. Sometimes, a Time dimension will contain days as the lowest level of detail. On occasion, particularly if you're monitoring a manufacturing operation or Internet activity, you might create a dimension with minutes or even seconds as the lowest level of detail. Whatever the level of detail, a Time dimension has certain unique qualities.

For example, time typically occurs in regular intervals. Each hour contains 60 minutes, each day contains 24 hours, each quarter contains 3 months, and each year contains 4 quarters. This repetitive nature of time encourages certain questions, such as, "How does this month

compare to the same month of last year?” The multidimensional expressions (MDX) language, which you’ll learn about in Chapter 8, “Using MDX,” has functions that make it easy to answer this type of question. By flagging certain dimensions as Time dimensions, and certain levels within a dimension as specific units of time, you can make those functions easy to use.

Of course, time isn’t completely uniform because the 365 days in a year aren’t evenly divisible by the 7 days in a week or the 12 months in a year. Some months have 30 days; some have 31, or 28, or occasionally 29. Months begin on different days of the week. Irregularities are a fact of life in Time dimensions, and when working with time, you need to be prepared for both the regularities and the irregularities.

One irregularity that frequently arises when dealing with time is that many organizations use a fiscal year—where the starting day of the year isn’t January 1. As you saw in Chapter 3, Analysis Services can build a Time dimension based on a specified date range and add a special calendar to this dimension for Fiscal Year. For greater flexibility, you should use a dimension table from your data warehouse because you include special properties for a date, such as the season for a month if this information is important to analysis over time in your organization.

Modifying a Data Source View

As you learned previously in this chapter, the DSV is a very flexible structure that can be modified as you add dimensions and cubes to your project. You can add tables to the DSV at any time. In addition, you can add a *Named Query* to limit the columns available from a single table or to construct a logical view by combining columns from multiple tables. As another alternative, you can add a derived column, known as a *Named Calculation*, to a table. Using a Named Query or a Named Calculation gives you the ability to manipulate the data structures for use by Analysis Services even if you don’t have permissions to make similar changes at the database level.

If you use Analysis Services to generate a Server Time dimension for you based on a range of dates, you need to use a Date/Time column in the fact table to use this dimension. (Creating a Server Time dimension is discussed later in this chapter.) However, it’s generally recommended that you use a separate dimension table for dates rather than use a Date/Time column in the fact table. As with other dimension tables, you would then use an integer key to join the date dimension table with the fact table. This difference in data types in the fact table—date/time which requires 8 bytes versus integer which requires 4 bytes—can have a big impact on the amount of space required to store the fact table in your data warehouse when the fact table contains millions of records. Further, as already noted, you can create a Time dimension table that includes special time-based attributes that your business uses when analyzing data that won’t be available if you rely solely on the Server Time dimension.

In this procedure, you’ll add tables to the DSV.

Add tables to a data source view

1. In Solution Explorer, double-click SSAS Step by Step DW.dsv to open the Data Source View Designer.

Alternatively, if the designer is still open in the main window of Visual Studio, you can click the SSAS Step by Step DW.dsv [Design] tab to continue working with the DSV.
2. Right-click the background of the Data Source View Diagram pane, and then click Add/Remove Tables.
3. In the Add/Remove Tables dialog box, double-click both `dbo.DimTime` and `dbo.Fact-ResellerSales` in the Available Objects list to move these two tables to the Included Objects list, and then click OK.

You'll use the `DimTime` table as the main table for the Time dimension that you create in the next procedure. In a later procedure, you'll use the `FactResellerSales` table to learn how the same time dimension can be used in different contexts, known as role-playing. Notice in the data source view diagram that there are three relationships between `Fact-ResellerSales` and `DimTime`.

Creating a Time Dimension

When you create a Standard dimension using the Dimension Wizard, you first specify the table(s) used to create the dimension and then you select the desired attributes. You'll notice that the process is slightly different when you create a Time Dimension because you can specify only one dimension table and you then map its columns to special time-related attributes, such as year, quarter, or month, to name a few. After the dimension is created, you might need to rename objects and adjust attribute properties to display dimension members correctly in each attribute hierarchy.

In this procedure, you'll add a Time dimension to your Analysis Services project.

Build a time dimension

1. In Solution Explorer, right-click the Dimensions folder, click New Dimension, and then click Next.
2. Clear the Auto Build check box, and then click Next.
3. Click the SSAS Step by Step DW DSV, and then click Next.
4. On the Select the Dimension Type page of the wizard, click Time Dimension, and then click `dbo_DimTime` in the corresponding drop-down list.

All tables in the DSV are available in the drop-down list. Since the wizard is unable to identify which table is the time dimension, you must select the appropriate table here.

5. Click Next.

6. On the Define Time Periods page, click the drop-down list for Year in the Time Table Columns, scroll through the list of columns, and then click CalendarYear to assign this column to the Year time property.
7. Repeat the previous step to assign table columns to specific time properties, as shown in the following table:

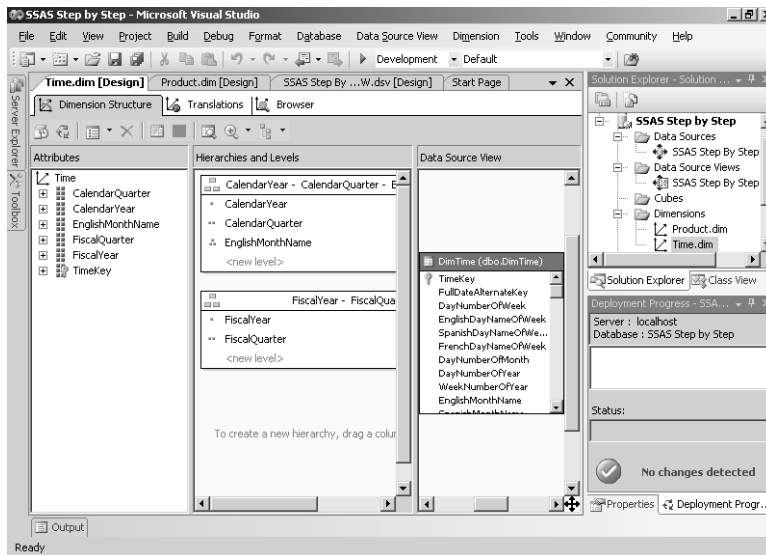
<i>Time Property</i>	<i>Time Table Columns</i>
Quarter	CalendarQuarter
Month	EnglishMonthName
Fiscal Quarter	FiscalQuarter
Fiscal Year	FiscalYear

8. Click Next.

The Review New Hierarchies page of the wizard shows you the user hierarchies that will be created based on your selection of time properties on the previous page of the wizard. You'll learn more about user hierarchies in Chapter 7. You have the option here to remove the autogenerated hierarchies or to remove levels from a hierarchy. For now, you'll leave the hierarchies as they are.

9. Click Next, change the dimension name from Dim Time to **Time**, and then click Finish to complete the wizard.

Your screen looks like this:



Now Time.dim is available in the Dimensions folder of the Solution Explorer window and the Dimension Designer for the new dimension is open in Visual Studio.

10. Rename attributes as shown in the following table:

Rename this	As this
CalendarQuarter	Calendar Quarter
CalendarYear	Calendar Year
EnglishMonthName	Month
FiscalQuarter	Fiscal Quarter
FiscalYear	Fiscal Year
TimeKey	Date

Attribute names are often used as row or column headings in reports, so you might need to alter the default attribute names to a more suitable name for reporting purposes.

11. Rename user hierarchies as shown in the following table:

Rename this	As this
CalendarYear - CalendarQuarter - EnglishMonthName	Calendar
FiscalYear - FiscalQuarter	Fiscal

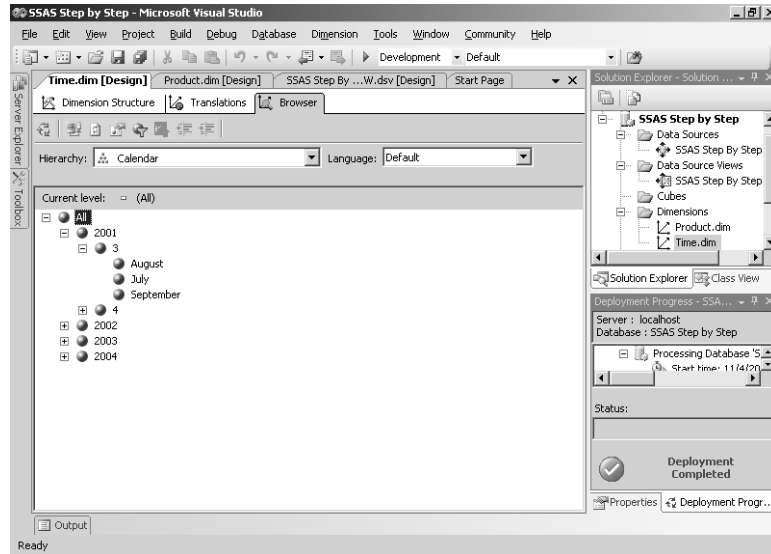
Notice that the level names displayed in the user hierarchy were not changed when you renamed attributes. For example, EnglishMonthName is still visible and was not renamed as Month. For consistency in a production database, you should rename levels in the user hierarchy to match the current attribute name, but since this dimension is for learning purposes only, you can leave the level names as they are.

12. In the Attributes list, right-click Date, click Properties, and then, in the Properties window, scroll to the *NameColumn* property, click (new) in the property's drop-down list, click FullDateAlternateKey, and then click OK.

The key column for this attribute is still TimeKey, which is used to join the dimension to related fact tables, but the key has no meaning to users. Instead, you can use the *NameColumn* property to display the name of an attribute member. In this case, users will see the value from the FullDateAlternateKey column of the DimTime table, which is a date value in a mm/dd/yy format.

13. Right-click the SSAS Step by Step project in the Solution Explorer window, and then click Deploy.
14. When deployment is complete, click the Browser tab, and then, in the Hierarchy drop-down list, click Calendar.
15. Expand the All member in the Calendar hierarchy tree, expand 2001, and then expand 3.

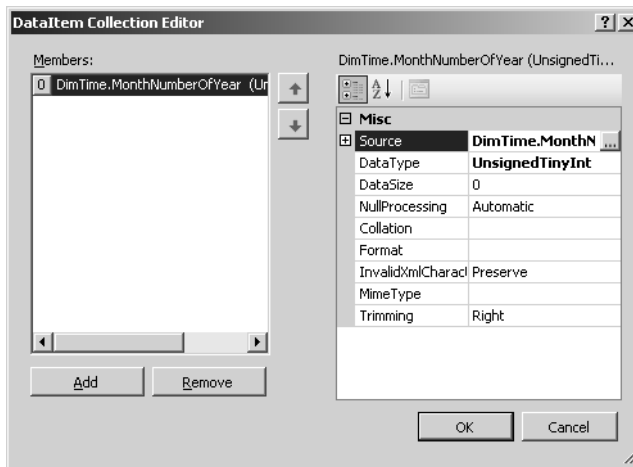
Your screen looks like this:



Notice that the months are sorted alphabetically: August, July, September. Sort order of attributes within the attribute hierarchy is determined by the *OrderBy* property which defaults to *Key*. Because the key value for Month is *DimTime.EnglishMonthName*, the result is an alphabetical sort based on the name of the month. However, you can produce the correct sort order by using another column in the *DimTime* table—*MonthNumberOfYear*.

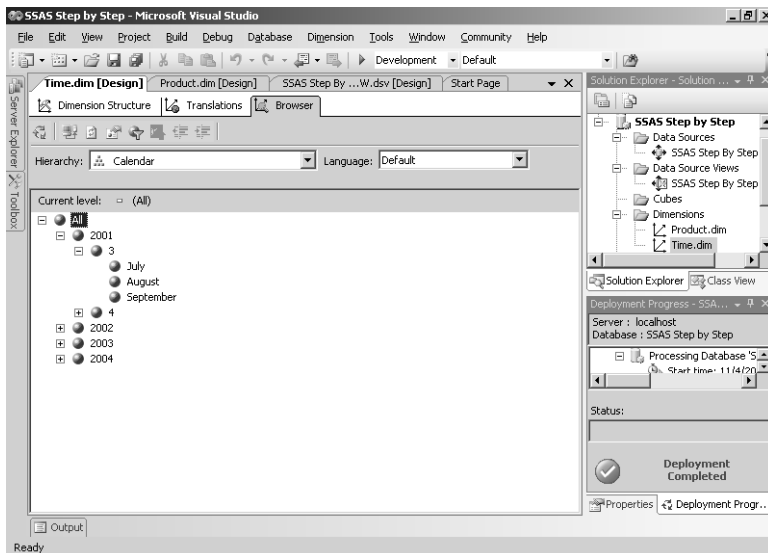
16. Click the Dimension Structure tab, right-click Month, click Properties, and then, in the Properties window, scroll to the *KeyColumns* property.
17. Click the ellipsis button (...) in the property's list box, and then click the ellipsis button in the *Source* property in the DataItem Collection Editor dialog box.

18. Click MonthNumberOfYear in the Source Columns list, and then click OK. Your screen looks like this:



19. Click OK to close the DataItem Collection Editor dialog box.
20. Right-click the SSAS Step by Step project in the Solution Explorer window, and then click Deploy.
21. When deployment is complete, click the Browser tab, and then click the Reconnect button in the Browser toolbar. If necessary, expand the All member in the Calendar hierarchy tree, expand 2001, and then expand 3.

Your screen now looks like this:



22. The months are now correctly sorted.

The Server Time Dimension

Using the Dimension Wizard, you can create a Standard dimension, a Time dimension, or a Server Time dimension. The Server Time dimension is unique because the data for this dimension does not come from a dimension table in your data warehouse, but is generated by Analysis Services and stored in a proprietary file structure on the server. You simply specify the beginning date and end date of the dimension, select the time periods to include such as year, quarter, month, or date, and choose the special calendars, if any, to add to the dimension. In fact, the process to define a Server Time dimension is identical to the process you used to define the Time dimension in Chapter 3. In Chapter 3, however, you generated a schema to hold data for the dimension which Analysis Services populated for you. When you create a Server Time dimension, no such table is created. The data available to this dimension type is maintained solely by Analysis Services. To use this dimension with a fact table, you will need to have a date/time column instead of a dimension key in the fact table. Analysis Services will use the date in this column to join the fact table with the Server Time dimension.

Working with Role-Playing Dimensions

As you learned previously in this chapter, a dimension can play different roles in a fact table. You can recognize a *role-playing dimension* when there are multiple columns in a fact table that each have foreign keys to the same dimension table. For example, in the SSAS Step by Step DW database, there are three dimension keys in the FactInternetSales and FactResellerSales tables which all refer to the DimTime table. The same time dimension is used to track sales by order date, by shipment date, and by delivery date. If you add the Time dimension to a cube that contains either of these fact tables, the corresponding role-playing dimensions are automatically added to the cube.

In this procedure, you'll add a predefined cube object to your Analysis Services project.

Add an object to a project

1. In Solution Explorer, right-click the SSAS Step by Step project, point to Add, and then click Existing Item.
2. Browse to C:\Documents and Settings\\My Documents\Microsoft Press\as2005sbs\chap04, and then double-click Simple Cube.cube to add it to your project.

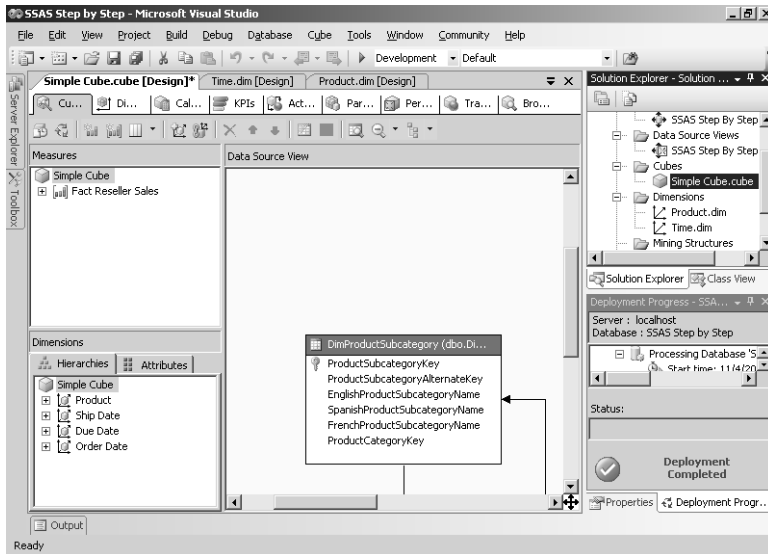
When you use the same DSV across multiple projects, you can also reuse object files in different projects. By adding an existing item, you created a copy of the item's file that is now associated with your current project. Changes to this file in the SSAS Step by Step project do not affect the original version of Simple Cube.cube in the C:\Documents and Settings\\My Documents\Microsoft Press\as2005sbs\chap04 folder.

In this procedure, you add role-playing time dimensions to a cube.

Add role-playing time dimensions to a cube

1. In Solution Explorer, double-click Simple Cube.cube to open the Cube Designer.
2. In the Dimensions pane of the Cube Designer, right-click the background, and then click Add Cube Dimension.
3. Click Time in the Select Dimension list, and then click OK.

Your screen looks like this:



Notice that Time is not added to the Dimensions pane, but there are three new dimensions: Ship Date, Due Date, and Order Date. Notice also there are no corresponding .dim files for these dimensions in the Dimensions folder of Solution Explorer. All that you see is the Time Dimension that you added previously. That is because the three new dimensions are role-playing dimensions for time.

Creating a Parent-Child Dimension

A Parent-Child dimension has a different table structure than most dimension tables. So far in this chapter, you have created a Product dimension which is a snowflake schema because it is based on multiple tables. You have also created a Time dimension which is a star schema because it is based on a single table. A Parent-Child dimension is also built from a single table, but it contains a built-in recursive hierarchical structure that you use to aggregate values across levels within each branch of the hierarchy.

For example, organizational data is often structured as a recursive hierarchy. An employee table includes records for each employee which, at their simplest, include a column to hold the unique employee key, a column for the employee name, and a column to hold the key to the record for the employee's manager. Because managers are also employees, there is another record in the same table that forms a branch of the employee hierarchy. You can recognize these parent-child relationships in your DSV by the self-referencing relationship that links the foreign key column of a table to the primary key column of the same table.

Adding an Employee Dimension

Just as with any other standard dimension, you use the Dimension Wizard to create a Parent-Child dimension. After specifying the dimension table and selecting attributes, you confirm the parent-child relationship in the table that the wizard detected by the self-referencing join on the table.

In this procedure, you'll add the DimEmployee table to the DSV and then use the table to create a parent-child dimension.

Create a parent-child dimension

1. Open SSAS Step by Step DW.dsv.
2. Right-click the background of the Data Source View Diagram pane, and then click Add/Remove Tables.
3. In the Add/Remove Tables dialog box, double-click `dbo.DimEmployee` in the Available Objects list to move this table to the Included Objects list, and then click OK.

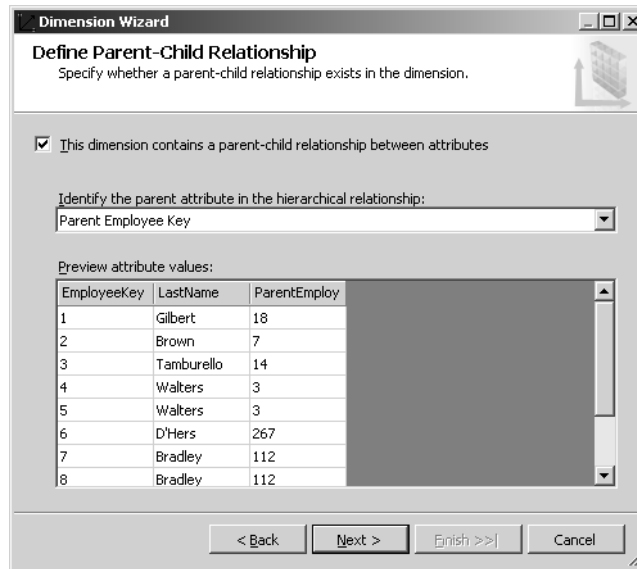
Notice the self-referencing relationship for the DimEmployee table in the DSV. This relationship is indicated by the arrow leading from the DimEmployee table back to itself.

4. In Solution Explorer, right-click the Dimensions folder, click New Dimension, click Next, clear the Auto Build check box, click Next, click the SSAS Step by Step DW DSV, and then click Next twice.
5. Click `dbo.DimEmployee` in the Main Table drop-down list, click `EmployeeKey` in the Key Columns list, click `LastName` in the Column Containing The Member Name drop-down list, and then click Next.
6. On the Select Dimension Attributes page of the wizard, select the check box next to `ParentEmployeeKey`.

You must include the attribute that defines the foreign key column of a record in a parent-child dimension.

7. Click Next twice.

The Define Parent-Child Relationship page of the wizard looks like this:



Notice that the This Dimension Contains A Parent-Child Relationship Between Attributes check box is selected. This relationship was automatically detected by the wizard based on the self-referencing join on this table in the DSV. ParentEmployeeKey is also correctly detected as the parent attribute as a result of the self-referencing join. This page of the wizard also allows you to preview attribute keys, attribute values, and the corresponding parent attribute keys for the top records in the table so that you can verify that the parent-child structure has been set up correctly.

- Click Next, change the name of the dimension from Dim Employee to **Employee**, and then click Finish.

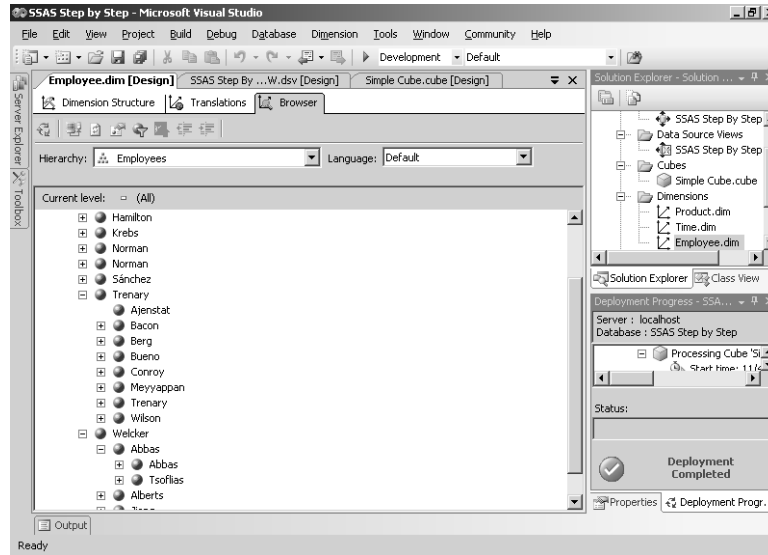
The Employee parent-child dimension is now added to your project.

- In the Attributes pane of the Dimension Structure tab, right-click Dim Employee, click Rename, and then type **Employee**.
- Repeat the previous step to rename Parent Employee Key as **Employees**.
- Keeping Employees selected in the Attributes pane, open the Properties window and locate the *Usage* property.

Valid values for the *Usage* property are Key, Parent, and Regular. When you create a Parent-Child dimension, the Dimension Wizard automatically sets the value of this property to Parent for the parent attribute. There can be only one attribute with *Usage* set to Parent within a dimension. Similarly, there can be only one attribute with *Usage* set to Key. All other attributes must have a *Usage* value equal to Regular or dimension processing will fail.

12. Right-click the SSAS Step by Step project in the Solution Explorer window, and then click Deploy.
13. When deployment is complete, click the Browser tab, and then, in the Hierarchy drop-down list, click Employees.
14. Expand the All member in the Employees hierarchy tree, and then expand Sánchez, Trenary, Ajenstat, Welcker, and Abbas.

Your screen looks like this:



You can see that there are a different number of levels within the hierarchy. Notice that Trenary is a parent member with several child members. Only one of the child members—Conroy—is also a parent member. Trenary contains a data member also named Trenary. This type of structure is called a ragged hierarchy because the leaf-level members (those without child members) do not all share the same number of ancestors. For example, Ajenstat has three ancestor members—Trenary, Sánchez, and the All member, whereas Tsofilias has four ancestor members—Abbas, Welcker, Sánchez, and the All member.

Totaling Data for Non-Leaf-Level Data Members

In a standard dimension, only the leaf-level members can correspond to values in the fact table. For example, in a Time dimension, you can't have some rows in the fact table with monthly values and other rows with quarterly values. In a parent-child dimension, on the other hand, it's common to have values in the fact table at both the leaf level and at a parent level. For example, in the SSAS Step by Step DW database, Abbas is a manager, but is also directly responsible for some of the sales in the SalesFact table, as well as being indirectly

responsible for the sales of direct reports. You need to decide whether or not cube browsers can see sales that Abbas made or only the sales for the direct reports.

In this procedure, you'll change the *MembersWithData* property value to compare the difference between the available options.

Set the *MembersWithData* property

1. Open the Simple Cube cube designer, and then, in the Dimensions pane, right-click the background, and then click Add Cube Dimension.
2. Click Employee in the Select Dimension list, and then click OK.
3. Deploy the project, and then click the Browser tab of the Cube Designer when deployment is complete.
4. In the cube metadata tree, expand Employee, drag the Employees hierarchy (not the Employee hierarchy!) to the area of the grid labeled Drop Row Fields Here, and then drag the Measures object to the area of the grid labeled Drop Totals or Detail Fields Here.
5. Click the plus sign next to Sánchez to drill down one level, click the plus sign next to Welcker to drill down another level, and then click the plus sign next to Abbas to drill down one more level.

Your screen looks like this:

The screenshot shows the SSAS Step by Step cube designer interface. The main window displays a cube design with dimensions Employee, Order Date, Product, and Ship Date. The Employee dimension is expanded to show a hierarchy of members: Sánchez, Welcker, Abbas, and Tsollias. The data table shows sales amounts for each member and their parents. The deployment progress window at the bottom right shows 'Deployment Completed'.

Level 02	Level 03	Level 04	Level 05	Order Quantity	Sales Amount
			Abbas	825	172524.4515
			Tsollias	4123	1421810.9252
			Total	4948	1594335.3767
		Alberts		49223	15535946.255
		Jiang		160207	63320315.349
		Total		214378	80450596.982
	Sánchez			214378	80450596.982
	Welcker			214378	80450596.982
	Total			214378	80450596.982
	Grand Total			214378	80450596.982

Notice that when you expand Abbas, you can see on Level 05 the total sales of 172,524.4515 for Abbas and total sales of 1,421,810.9252 for Tsollias. The combined sales for these two individuals is 1,594,335.3767, which appears on the total row beneath these two members.

6. Click the minus sign next to Abbas in Level 04 to collapse this branch of the hierarchy.

The total sales amount for Abbas is 1,594,335.3767, which, as you saw in the previous step, is actually total sales for both Abbas and Tsoflias.

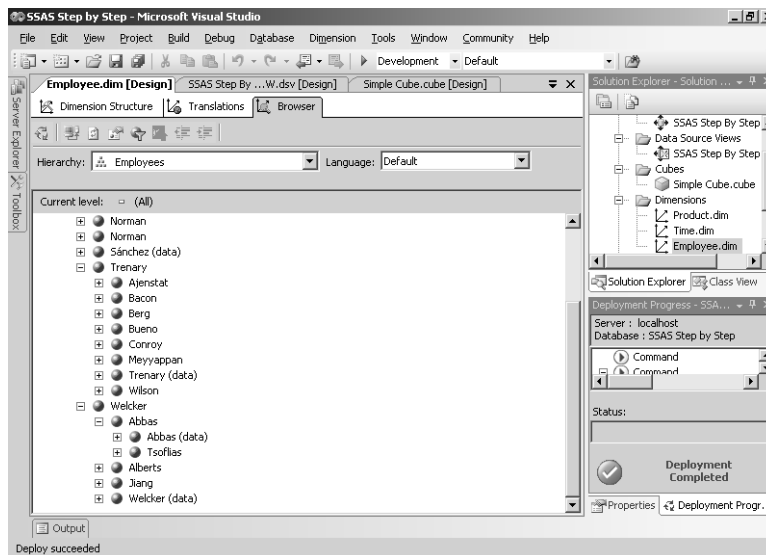
Other than the fact that you saw one Abbas member in Level 04 and another Abbas member in Level 05, it was not obvious from the member names which member represented the aggregated totals from child members and which were the aggregated values for the parent member from the fact table. You can change the template that creates the new member name to make a distinction.

7. Open the Employee Dimension Designer, click the Dimension Structure tab if necessary, right-click the Employees attribute in the Attributes pane, and then click Properties.
8. In the Properties window, change the *MembersWithDataCaption* to * (**data**).

The asterisk is a placeholder for the member name.

9. Deploy the project, and then click the Browser tab of the Cube Designer when deployment is complete. Click the Reconnect button, and then drill down to Level 05 of the Employee hierarchy.

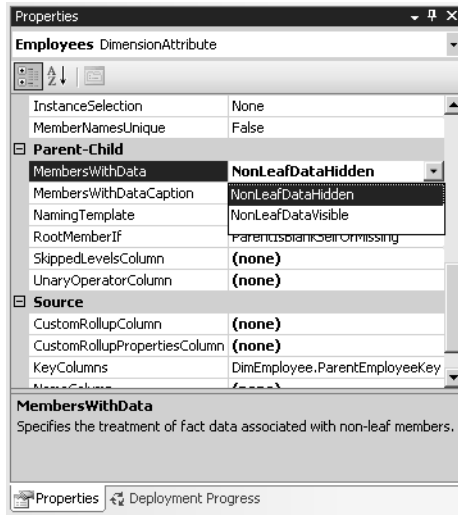
Your screen looks like this:



The managers' data members now are more easily distinguished from the other child members as well as their parent member.

10. Switch back to the Dimension Structure tab in the Employee Dimension Designer, right-click the Employees attribute in the Attributes pane, and then click Properties.
11. In the Properties window, click NonLeafDataHidden in the *MembersWithData* property drop-down list to review the available values.

The Properties window looks like this:

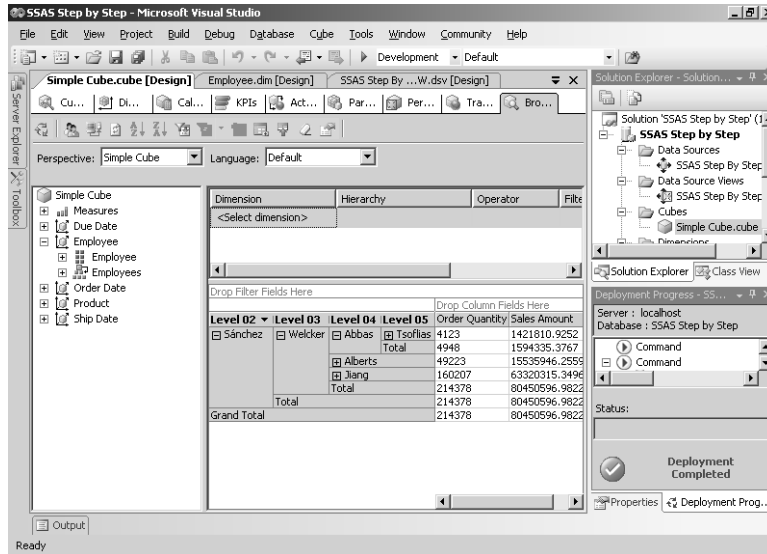


The *MembersWithData* property is available only for a parent-child dimension, and more specifically is available only for an attribute with its *Usage* property set to Parent. The *MembersWithData* property has two possible values: NonLeafDataVisible and NonLeafDataHidden.

NonLeafDataVisible (the default) creates a new member for each parent. That new member can display the values linked to that member in the fact table. This option is appropriate when you want to directly compare a manager's sales with those of the direct reports. NonLeafDataHidden does not create a new member for each parent. However, the total for a parent member may be greater than the sum of its visible children.

12. Click NonLeafDataHidden in the *MembersWithData* property drop-down list.
13. Deploy the project, and then click the Browser tab of the Cube Designer when deployment is complete. Click the Reconnect button, and then drill down to Level 05 of the Employee hierarchy.

Your screen looks like this:



Notice that you can no longer see the individual sales contribution of Abbas, only that of Tsoflias. However, the total sales for Abbas are still 1,594,335.3767. Non-leaf-level data is always included in the aggregated values at each level of the hierarchy. The *MembersWithData* property simply controls whether or not you see this data allocated to a parent member.

Managing Levels within a Parent-Child Dimension

In a parent-child dimension, sometimes there is only one member that does not truly have a parent. For example, the values for the manager at the top level of the Employee dimension are the same values for the All member. In your Employee dimension, there's no need to have an All level above Sánchez. The All level can be removed for other dimension types as well, such as a time dimension when you have a business rule that time should not be aggregated for all years in the cube.

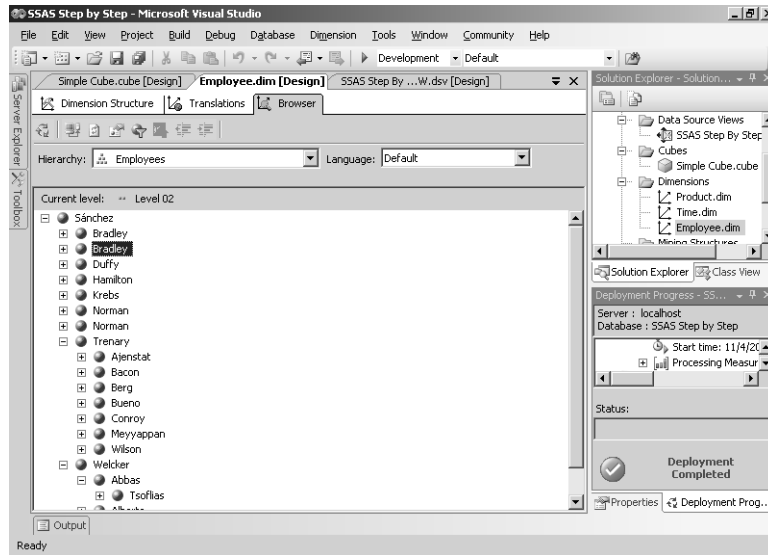
Besides removing the All level, you can also manage the names of levels for a parent-child hierarchy. Recall that when you browsed the cube in the previous procedure, each level was numbered sequentially, beginning with Level 02 and continuing through Level 05. A parent-child dimension has an exclusive property that you can use to provide a more meaningful name to each level.

In this procedure, you'll remove the All level from the Employees hierarchy.

Remove the All level from a hierarchy

1. Open the Employee Dim Designer, right-click Employees in the Attributes pane of the Dimension Structure tab, and then click Properties.
2. Scroll to the *IsAggregatable* property, and then click False in the property's drop-down list.
3. Deploy the project, and then click the Browser tab of the Dimension Designer when deployment is complete. Click the Reconnect button.

Your screen looks like this:



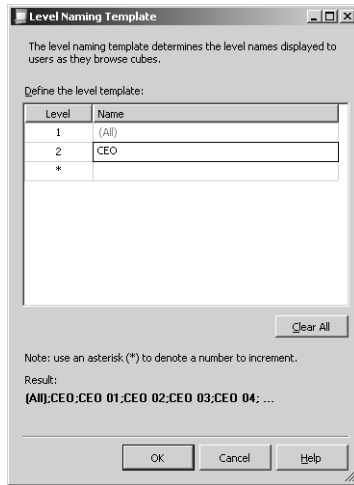
The All level is now removed. Since the All level for this parent-child hierarchy is not visible in the cube browser, you will not notice any change there. This property only affects queries that return the list of members contained in the hierarchy for which this property has been set to False.

In this procedure, you'll define a level naming template for the Employees hierarchy.

Create a level naming template for a parent-child hierarchy

1. Click the Dimension Structure tab, right-click Employees in the Attributes pane, and then click Properties.
2. Scroll to the *NamingTemplate* property, and then click the ellipsis button in the property's list box.
3. Click in the Name box to the right of the asterisk (*), and type CEO.

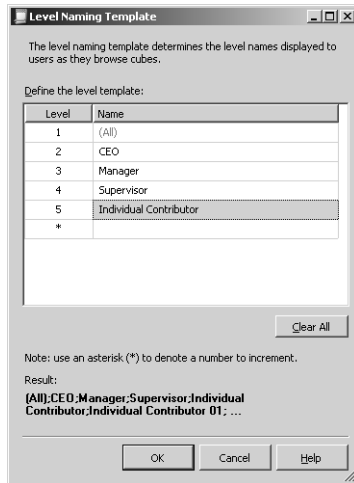
Your screen looks like this:



As soon as you begin typing, the asterisk changes to a 2, and a new row, with an asterisk, appears.

4. Click in the Name box in the new row, and type **Manager**. In the box of the next row, type **Supervisor**, and in the box of the final row, type **Individual Contributor**.

Your screen looks like this:

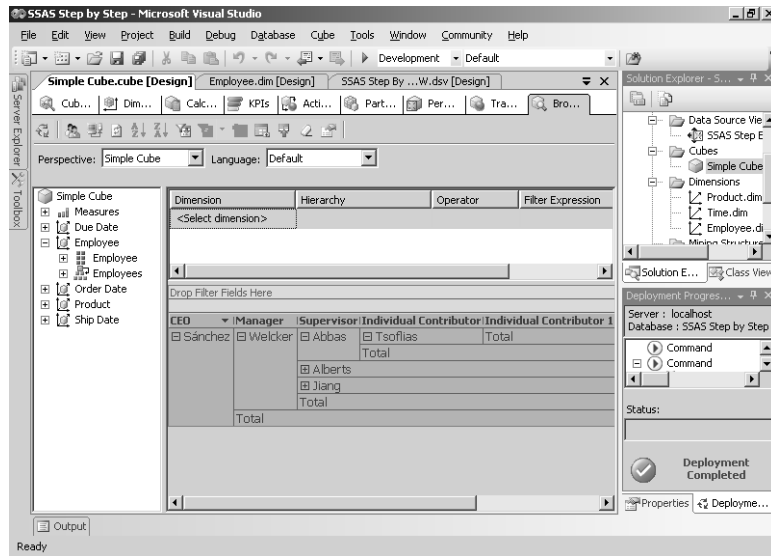


As you enter values into the level naming template grid, the result is displayed at the bottom of the grid. Any levels that are automatically created below the lowest level specified in the template are given sequentially incremented numbers.

5. Click OK to close the Level Naming Template dialog box.

6. Deploy the project, click the Browser tab of the Cube Designer when deployment is complete, and then click the Reconnect button.
7. Drag the Employees hierarchy to the grid area labeled Drop Row Fields Here, and then drill down until you can see five levels displayed.

Your screen looks like this:



Notice that instead of Level 01, Level 02, etc. as the column labels for the Employees hierarchy, you now see the names of each level that you specified: CEO, Manager, Supervisor, and Individual Contributor. Any levels below the lowest level specified in the template are given sequentially incremented numbers.

8. Save the solution.

Linked Dimensions

A *linked dimension* is based on a dimension that is stored in a separate Analysis Services database which may or may not be on the same server. You can create and maintain a dimension in just one database and then reuse that dimension by creating linked dimensions for use in multiple databases. Users will not notice any difference between a regular dimension and a linked dimension. As a developer, you will be able to browse the members of a linked dimension in the Dimension Designer, but you can only modify the dimension within its original Analysis Services database.

Chapter 4 Quick Reference

To	Do this
Add a data source to an Analysis Services project	In Solution Explorer, right-click the Data Sources folder, click New Data Source, click New, select a provider, enter a server name, specify the authentication method and credentials for authentication, and, if needed, select a database name, and then specify the credentials for Analysis Services to impersonate when accessing the data source during processing.
Create a data source view (DSV)	In Solution Explorer, right-click the Data Source Views folder, click New Data Source View, select an existing data source (or create a new one), and then add tables or views to the Included Objects list.
Create a standard dimension from an existing data source without using Auto Build	In Solution Explorer, right-click the Dimensions folder, click New Dimension, clear the Auto Build check box, select a data source view, select Standard Dimension, specify the main table, select the key column, optionally select the member name column, include any related tables, select attributes, and name the dimension.
Deploy an Analysis Services project	In Solution Explorer, right-click the Analysis Services project, and then click Deploy.
Browse a dimension	In the Dimension Designer, click the Browser tab, and then, if the project has been recently deployed, click the Reconnect button on the Browser toolbar. Select the appropriate hierarchy in the drop-down list and expand the attribute hierarchy tree to view the hierarchy members.
Disable an attribute hierarchy	In the Attributes pane of the Dimension Structure tab in the Dimension Designer, click an attribute, and then, in the Properties window, change the <i>AttributeHierarchyEnabled</i> property to False.
Add a table to a DSV	In the Data Source View Designer, right-click the Data Source View Diagram pane, click Add/Remove Tables, and add tables or views to the Included Objects list.
Create a time dimension from an existing data source	In Solution Explorer, right-click the Dimensions folder, click New Dimension, clear the Auto Build check box, select a DSV, select Time Dimension, specify the time dimension table, assign columns from the dimension table to built-in time periods, and then name the dimension.
Add an object file to a project	Right-click the project in the Solution Explorer, point to Add, click Existing Item, and then navigate to the folder containing the object to open the file.
Add role-playing dimensions to a cube	Ensure multiple relations are defined between a fact table and a dimension, create the dimension by using the Dimension Wizard, open the Cube Designer, right-click the background of the Dimension pane of the Cube Structure tab, click Add Cube Dimension, click the dimension in the list, and then click OK.
Create a parent-child dimension manually	Follow the procedure for creating a standard dimension and, on the Define Parent-Child Relationship page of the Dimension Wizard, select the check box This Dimension Contains a Parent-Child Relationship Between Attributes and identify the parent attribute key.

To	Do this
Hide data for non-leaf-level data members	In the Attributes pane of the Dimension Structure of the Dimension Designer, click the parent attribute of a parent-child dimension, and then, in the Properties window, change the <i>MembersWithData</i> property to <i>NonLeafDataHidden</i> .
Remove the All level of a hierarchy	In the Attributes pane of the Dimension Structure of the Dimension Designer, click an attribute, and then, in the Properties window, change the <i>IsAggregatable</i> property to <i>False</i> .
Apply a level naming template to a parent-child hierarchy	In the Attributes pane of the Dimension Structure of the Dimension Designer, click the parent attribute of a parent-child dimension, and then, in the Properties window, click the ellipsis button for the <i>NamingTemplate</i> property, and then type in a name for each level.

