Chapter 22

SQL Server Analysis Services

In This Chapter

- SSAS Terminology
- Developing a Multidimensional Cube Using BIDS
- Retrieving and Delivering Data
- Security of SQL Server Analysis Services
SQL Server Analysis Services (SSAS) is a group of services that is used to manage data that is stored in a data warehouse or data mart. SSAS organizes data from a data warehouse into multidimensional cubes (see Chapter 21) with aggregates to allow the execution of sophisticated reports and complex queries. The key features of SSAS are:

- Ease of use
- Support of different architectures
- Support of several APIs

SSAS offer wizards for almost every task that is executed during the design and implementation of a data warehouse. For example, the Data Source Wizard allows you to specify one or more data sources, while the Cube Wizard is used to create a multidimensional cube where aggregate data is stored. Additionally, ease of use is guaranteed by Business Intelligence Development Studio (BIDS). You can use this tool to develop databases and other data warehouse objects. This means that BIDS offers one interface for developing SSAS projects as well as SQL Server Integration Services (SSIS) and Reporting Services (SSRS) projects.

In contrast to most other data warehouse systems, SSAS allow you to use the architecture that is most appropriate for your needs. You can choose between the three architectures (MOLAP, ROLAP, and HOLAP) discussed in detail in Chapter 21.

SSAS provides many different APIs that can be used to retrieve and deliver data. One of these is OLE DB for OLAP interface that allows you to access SSAS cubes. Several APIs are described later in this chapter, in the section “Retrieving and Delivering Data.”

Security aspects of SSAS are discussed at the end of the chapter.

**SSAS Terminology**

The following are the most important terms related to SSAS:

- Cube
- Dimension
- Member
- Hierarchy
- Cell
A cube is a multidimensional structure that contains all or part of the data from a data warehouse. Although the term “cube” implies three dimensions, a multidimensional cube generally can have many more dimensions. Each cube contains all other components in the preceding list.

A dimension is a set of logically related attributes (stored together in a dimension table) that closely describes measures (stored in the fact table). For instance, Time, Product, and Customer are the typical dimensions that are part of many BI applications. (These three dimensions from the AdventureWorksDW database are used in the example in the following section that demonstrates how to create and process a multidimensional cube using BIDS.)

**NOTE**

One important dimension of a cube is the Measures dimension, which includes all measures defined in the fact table.

Each discrete value in a dimension is called a member. For instance, the members of a Product dimension could be Computers, Disks, and CPUs. Each member can be calculated, meaning that its value is calculated at run time using an expression that is specified during the definition of the member. (Because calculated members are not stored on the disk, they allow you to add new members without increasing the size of a corresponding cube.)

Hierarchies specify groupings of multiple members within each dimension. They are used to refine queries concerning data analysis.

Cells are parts of a multidimensional cube that are identified by coordinates (x-, y-, and z-coordinates, if the cube is three-dimensional). This means that a cell is a set containing members from each dimension. For instance, consider the three-dimensional cube in Chapter 21 (see Figure 21-3) that represents car sales for a single region within a quarter. The cells with the following coordinates belong, among others, to the cube:

- First quarter, South America, Falcon
- Third quarter, Asia, Eagle
When you define hierarchies, you define them in terms of their levels. In other words, *levels* describe the hierarchy from the highest (most summarized) level to the lowest (most detailed) level of data. The following list displays the possible hierarchy levels for the Time dimension:

- Quarter (Q1, Q2, Q3, Q4)
- Month (January, February, …)
- Day (Day1, Day2, …)

As you already know from Chapter 21, measures are numerical values, such as price or quantity, that appear in a fact table but do not build its primary key. A *measure group* is a set of measures that together build a logical unit for business purposes. Each measure group is built on the fly, using corresponding metadata information.

A cube can be divided into one or more partitions. *Partitions* are used by SSAS to manage and store data and aggregations for a measure group in a cube. Every measure group has at least one partition, which is created when the measure group is defined. Partitions are a powerful and flexible means of managing large cubes.

### Developing a Multidimensional Cube Using BIDS

The main component of SSAS is Business Intelligence Development Studio (BIDS), a management tool that provides one development platform for different BI applications. Built on Visual Studio, BIDS supports an integrated platform for system developers in the business intelligence area.

You can use BIDS not only to create and manage cubes, but also to design capabilities for SQL Server Reporting Services and SQL Server Integration Services. (SSRS is discussed in Chapter 24, while the description of SSIS is beyond the scope of this book.)

**NOTE**

The user interface of BIDS is similar to the interface of SQL Server Management Studio. However, these two tools differ in their deployment: you should use BIDS to develop BI projects, while the goal of SQL Server Management Studio is mainly to operate and maintain database objects in relation to business intelligence.

The following steps are necessary to create and process a multidimensional cube using Business Intelligence Development Studio:

1. Create a BI project.
2. Identify data sources.
3. Specify data source views.
4. Create a cube.
5. Design storage aggregation.
6. Process the cube.
7. Browse the cube.

The following sections describe in detail these steps.

**Create a BI Project**

The first step in building an analytic application is to create a new project in BIDS. To start BIDS, choose Start | All Programs | Microsoft SQL Server 2012 | SQL Server Business Intelligence Development Studio. Next, choose File | New | Project. In the New Project dialog box, in the Installed Templates pane (see Figure 22-1), select Analysis Services (under the Business Intelligence folder) and select Analysis Services Multidimensional and Data Mining Project. Type the name of the project...
and its location in the Name and Location text boxes, respectively. For purposes of this example, name the project **BI_Project**, as shown in Figure 22-1. Click OK to create the new project.

The new project is always created in a new solution. A **solution** is the largest management unit in BIDS and always comprises one or more projects. (In this example, the solution has the same name as the project.)

**NOTE**

*If the Solution Explorer pane, which allows you to view and manage objects in a solution or a project, is not visible, choose View | Solution Explorer to view it.*

**Identify Data Sources**

To identify data sources, right-click the Data Sources folder in the Solution Explorer pane and choose New Data Source. The Data Source Wizard appears, which guides you through the process of creating a data source. (This example uses the **AdventureWorksDW** sample database as the data source.)

First, on the Select How to Define the Connection page, make sure that the Create a Data Source Based on an Existing or New Connection radio button is activated and click New. In the Connection Manager dialog box (see Figure 22-2), select Native OLE DB/SQL Server Native Client 11.0 in the Provider drop-down list and select the name of your database server in the Server Name drop-down list. (The choice of Native OLE DB/SQL Server Native Client 11.0 allows you to make a connection to an existing database of an instance of the Database Engine.) In the same dialog box, choose Use Windows Authentication and, from the Select or Enter a Database Name drop-down list, choose the **AdventureWorksDW** database. Before you click OK, click the Test Connection button to test the connection to the database. (The Select How to Define the Connection page appears only when you connect to the data source for the first time.)

The next step of the wizard is the Impersonation Information page. These settings determine which user account SSAS uses when connecting to the underlying source of data using Windows authentication. Which setting is appropriate depends on how this data source is being used. Click the Use a Specific Windows User Name and Password radio button and type your username and password in the corresponding fields. Click Next.

Finally, on the Completing the Wizard page, give the new data source a name (for this example, call it **BI_Source**) and click Finish. The new data source appears in the Solution Explorer pane in the Data Sources folder.
Chapter 22: SQL Server Analysis Services

After identifying data sources in general, you have to determine exactly which data you want to select from the data source. In our example, it means that you have to select tables from the *AdventureWorksDW* database, which will be used to build a cube. This step involves specifying data source views, discussed next.

Specify Data Source Views

To create data source views, right-click the Data Source Views folder in the Solution Explorer pane and choose New Data Source View. The Data Source View Wizard guides...
you through the steps that are necessary to create a data source view. (This example creates a view called **BI_View**, which is based on the **Customer** and **Project** entities.)

First, on the Select a Data Source page, select an existing relational data source (for this example, select **BI_Source**) and click Next. On the next wizard page, Select Tables and Views, choose tables that belong to your cube either as dimension tables or fact tables. To choose a table, select its name in the Available Objects pane and click the > button to move it to the Included Objects pane. For this example, choose the tables for customers and products (**Dim Customer** and **Dim Product**, respectively) in the **AdventureWorksDW** database. These tables will be used to build cube dimensions. They build the set of dimension tables used for your star schema.

Next, on the same wizard page, you need to specify one or more fact tables that correspond to the preceding dimension tables. (One fact table, together with the corresponding dimension tables, creates a star schema.) To do so, click the Add Related Tables button below the Included Objects pane. This instructs the system to find tables that are related to the **Dim Customer** and **Dim Product** tables. (To find related tables, the system searches all primary key/foreign key relationships that exist in the database.)

The system finds several fact tables and adds them to the Included Objects pane. Of these tables, you need only one, **Fact Internet Sales**, to build the star schema. Besides the corresponding fact tables, the system also searches for and adds other tables that are created separately for a hierarchy level of the corresponding dimension. One such table to keep is **Dim Product Subcategory**, which incarnates a hierarchy level called **Subcategory** of the **Product** dimension. Also keep the **Dim Date** table, because the Time dimension is almost always a part of a cube.

Thus, for your star schema, you need the following five tables (as shown in Figure 22-3):

- **Fact Internet Sales**
- **Dim Customer**
- **Dim Date**
- **Dim Product**
- **Dim Product Subcategory**

Exclude all other system-chosen tables that appear in the right pane by selecting them and clicking the < button.

After restructuring the tables, click Next. On the Completing the Wizard page, the wizard will be completed by choosing a name for a new source view (**BI_View**).
After you click Finish, the Data Source View Designer displays a graphical representation of the tables in the data schema you have defined, as shown in Figure 22-4. (Data Source View Designer is a tool that is used to show a graphical representation of the data schema you have defined.)

**NOTE**

Using drag and drop, I changed the design of the source view in Figure 22-4 so that the tables have the convenient form of the star schema. When you take a look at the figure, you will see that the fact table is in the middle and the corresponding dimension tables build the circle around it. (Figure 22-4 actually has the form of a snowflake schema, because the Dim Product Subcategory table presents the hierarchy level of the Product dimension.)

Data Source View Designer offers several useful functions. To inspect the objects you have in your source view, move your mouse pointer to the cross-arrow icon in the
bottom-right corner. When the pointer changes to a cross-arrow icon, click the icon and hold it. The Navigation window appears. Now you can navigate from one part of the diagram to another part. (This is especially useful when you have a diagram with dozens of entities.) To view the data in a table, right-click the table and choose Explore Data. The content of the table appears in a separate window.

You can also create named queries, which are queries that are persistently stored and therefore can be accessed like any table. To create such a query, click Data Source View
in the menu bar and then select the New Named Query icon. The Create Named Query dialog box allows you to create any query in relation to selected tables.

Create a Cube

Before you create a cube, you must specify one or more data sources and create a data source view, as previously described. After that, you can use the Cube Wizard to create a cube.

To create a cube, right-click the Cubes folder of the BI_Project project in the Solution Explorer pane and choose New Cube. The welcome page of the Cube Wizard appears. Click Next. On the Select Creation Method page, choose Use Existing Tables, because the data source view exists and can be used to build a cube. Click Next.

On the Select Measure Group Tables page, you select measures from the fact table(s). Therefore, select the only fact table, Fact Internet Sales, and click Next. The wizard chooses all possible measures from the selected fact table and presents them on the Select Measures page. Check only the Total Product Costs column of the Fact Internet Sales table as the single measure (see Figure 22-5). Click Next.

![Figure 22-5 The Select Measures wizard page](image_url)
On the Select New Dimensions page, select all three dimensions (Dim Date, Dim Product, and Dim Customer) to be created, based on the available tables. The final page, Completing the Wizard, shows the summary of all selected measures and dimensions. Click Finish to finish creating the cube called BI_Cube.

Design Storage Aggregation

As you already know from Chapter 21, basic data from the fact table can be summarized in advance and stored in persistent tables. This process is called aggregation, and it can significantly enhance the response time of queries, because scanning millions of rows to calculate the aggregation on the fly can take a very long time.

There is a tradeoff between storage requirements and the percentage of possible aggregations that are calculated and stored. Creating all possible aggregations in a cube and storing all of them on the disk results in the fastest possible response time for all queries, because the response to each query is almost immediate. The disadvantage of this approach is that the storage and processing time required for the aggregations can be substantial.

On the other hand, if no aggregations are calculated and stored, you do not need any additional disk storage, but response time for queries concerning aggregate functions will be slow because each aggregate has to be calculated on the fly.

SSAS provides the Aggregation Design Wizard to help you design aggregations optimally. To start the wizard, you have first to start the Cube Designer. (The Cube Designer is used to edit various properties of an existing cube, including the measure groups and measures, cube dimensions and dimension relationships.) To start it, right-click the cube in Solution Explorer and select Open or View Designer from the context menu. Now, click the Aggregations tab in the main menu of the Cube Designer. In the table that appears in the Cube Designer (Fact Internet Sales), right-click the cell under the Aggregations column and choose Design Aggregations. That starts the Aggregation Design Wizard.

In the first step of the wizard, Review Aggregation Usage, you review aggregation usage settings. In this step, you can include or exclude the attributes that appear on the page. Leave the settings as they are and click Next.

The next step is to specify the number of members in each attribute. You do this on the Specify Object Counts page. For each selected cube object, you have to enter the estimated count value or partition count value, before the wizard starts to create and store the selected aggregations. If you click the Count button, the wizard automatically performs the object counts and displays the obtained counts. (Figure 22-6 shows the Specify Objects Counts page after clicking the Count button.) Click Next.
In the second-to-last step, the Set Aggregation Options page, choose one of the four options to specify up to what point (or not at all) aggregations should be designed:

- **Estimated storage reaches __ MB**  Specifies the maximum amount of disk storage that should be used for precomputed aggregations. The larger the amount, the more precomputed aggregations that will be created.

- **Performance gain reaches __ %**  Specifies the performance gain that you want to achieve. The higher the percentage of precomputed aggregations, the better the performance.

- **I click Stop**  Enables you to decide when to stop the design process.

- **Do not design aggregation (0%)**  Specifies that no precomputed aggregations should be created.

**NOTE**

Generally, you should choose one of the first two alternatives. I prefer the second one, because it is very difficult to estimate the amount of storage for different star schemas and different sets of queries. A value between 80 percent and 90 percent is optimal in most cases.
Figure 22-7 shows the result of choosing the second option with the value set to 80 percent and clicking the Start button. The system created six aggregations and uses 243.4KB for them.

Click the Next button to go to the Completing the Wizard page. On this page, you can choose whether to process aggregations immediately (Deploy and process now) or later (Save the aggregations but do not process them). Choose the second option and click Finish.

Process the Cube

If you chose the Save the Aggregations But Do Not Process Them option as the final step in the preceding section, as recommended, you now have to process the cube. (A cube must be processed when you first create it and each time you modify it. If a cube has a lot of data and precomputed aggregations, processing the cube can be very time consuming.) To process the cube, right-click the name of your cube in the Cubes folder of Solution Explorer and select Process. The system starts processing the cube and displays the progress of this activity (see Figure 22-8).
Chapter 22: SQL Server Analysis Services

Browse the Cube

To browse a cube, right-click the cube name (in the Cubes folder of the Solution-Explorer) and choose Browse. The Browse view appears. You can add any dimension to the query by right-clicking the dimension name in the left pane and choosing Add to Query. You can also add a measure from that pane in the same way. (Adding measures first is recommended.) Figure 22-9 shows the tabular representation of the total product costs for Internet sales for different customers and products.

The approach is different if you want to calculate values of measures for particular dimensions and their hierarchies. For example, suppose that you want to deliver for customers with customer IDs 11008 and 11741 total product costs for all products they have ordered in the time period 2006/03/01 through 2006/08/31. In this case, you first drop the measure (Total Product Cost) from the left pane into the editing pane, and then you choose values in the pane above it to restrict the conditions for...
each dimension (see Figure 22-10). First, in the Dimension column, choose the **Dim Customer** table, and in the Hierarchy column, choose the primary key of this table (**Customer Key**). In the Operator column, choose Equal, and in the Filter Expression column, choose both values 11008 and 11741 one after the other.

In the same way, choose the conditions for the **Dim Product** dimension table. The only difference is that all product values should be included. For this reason, in the Filter Expression column you should choose the root of the dimension. (The root of each dimension is specified by All.) Finally, the table’s column **OrderDate** is chosen for
the Dim Date dimension table, and with the corresponding key. The Operator column in this case is Range (Inclusive) and the Filter Expression column allows you to type the beginning and end of the time period. As you can see in Figure 22-10, the total product costs for the both customers are 2171.2942.

Retrieving and Delivering Data

Now that you have seen how to build and browse the cube using BIDS, you are ready to learn how to retrieve data from a cube and deliver it to users. The primary goal of Development Studio is to develop BI projects, not to retrieve and deliver data to users. For this task, there are many other interfaces, such as:

- PowerPivot for Excel
- Multidimensional Expressions (MDX)
PowerPivot for Excel and MDX are discussed separately in the following two subsections, while the other interfaces in the list are briefly described here. (The reason I give more attention to these two interfaces than the others is that PowerPoint for Excel is the most important interface for end-users, while MDX is a tool that is used by many third-party SSAS solutions.)

To browse a cube in SQL Server Management Studio, start it and connect to the SSAS server on which you deployed your cube. After that, expand the Database folder and expand Cube. You’ll see all cubes that you have created for this database. Right-click the cube you want to use and choose Browse. As shown in Figures 22-9 and 22-10, the interface is the same as you saw earlier for SSAS. Thus, you can browse data in the same way as described in the section “Browse the Cube.”

**NOTE**

The most important thing to note when using Management Studio for multidimensional analysis is that you do not connect to the Database Engine. The Database Engine manages relational data, while SSAS stores and manages multidimensional cubes. For this reason, connect to your SSAS server.

OLE DB for OLAP is an industry standard for multidimensional data processing, published by Microsoft. It is a set of entities and interfaces that extends the ability of OLE DB to provide access to multidimensional data stores. OLE DB for OLAP enables users to perform data analysis through interactive access to a variety of possible views of the underlying data. Many independent software vendors use the specification of OLE DB for OLAP to implement different interfaces that allow users to access cubes created by SSAS. Additionally, using OLE DB for OLAP, the vendors can implement OLAP applications that can uniformly access both relational and nonrelational data stored in diverse information sources, regardless of location or type.

ADOMD (ActiveX Data Objects Multidimensional) is a Microsoft .NET Framework data provider that is designed to communicate with SSAS. With this interface, you can access and manipulate objects in a multidimensional cube, enabling web-based OLAP application development. This interface uses the XML for Analysis protocol to communicate with analytical data sources. Commands are usually sent in MDX. By using ADOMD.NET, you can also view and work with metadata.
Querying Data Using PowerPivot for Excel

PowerPivot for Excel is a tool that allows you to analyze data using what is probably the most popular Microsoft tool for such purpose: Microsoft Excel. It is a user-friendly way to perform data analysis using features such as PivotTable, PivotChart views, and slices.

**NOTE**

To work with PowerPivot, you need Microsoft Office 2010. You can also use SharePoint 2010, but this chapter discusses only PowerPivot for Excel.

Before you learn how to use this tool, take a look at the advantages of PowerPivot:

- Familiar Excel tools and features for delivering data are available.
- Very large data sets can be loaded from virtually any source.
- New analytical capabilities, such as Data Analysis Expressions (DAX), are available.

As you will see in a moment, you can use the same sources that you use for SSAS in almost the same way for PowerPivot. (You will use a cube similar to the one you created in the previous section to learn how to deliver data from a cube. You will use this cube afterward for different exercises.)

Data Analysis Expressions (DAX) is a new PowerPivot language that allows you to define custom calculations in PowerPivot tables and in Excel PivotTables. DAX comprises some of the functions that are used in Excel formulas, and additional operations that are designed to work with relational data.

**Working with PowerPivot for Excel**

Your first step is to import data from one or more data sources in Excel. Open Excel 2010 and click the PowerPivot tab. In the PowerPivot ribbon, click the PowerPivot Window tab. This opens the PowerPivot for Excel window. Your task is to create a cube similar to the one you already created using SSAS. So, create and process a cube according to the steps described in the previous section and with the table design shown in Figure 22-11.

The following list shows the tables you should choose (see also Figure 22-11):

- Fact Internet Sales
- Fact Reseller Sales
Figure 22-11  Data Source View Designer with the tables that you have to select

- Dim Customer
- Dim Date
- Dim Product
- Dim Product Subcategory
To choose these tables, use the same data source (BI_Source) as for the BI_Cube cube and create a new source view. After that, use the Cube Wizard to create the new cube (called BI_Cube2). In the wizard step called Select Measure Group Tables, choose the following measures from both fact tables (Fact Internet Sales and Fact Reseller Sales): Sales Amount, Total Product Costs, and Freight.

The next task after creating the BI_Cube2 cube is to connect to the cube. To do this, click From Other Sources on the Get External Data tab of the PowerPivot for Excel ribbon and choose SQL Server Analysis Services. This opens the Table Import Wizard with the Connect to a Data Source page displayed, as shown in Figure 22-12. Click Next.

Figure 22-12 The Connect to a Data Source wizard page
NOTE

You can choose as a data source Microsoft databases (Access or SQL Server), third-party databases (Oracle, Teradata, etc.), as well as nondatabase data sources.

On the next wizard page, Connect to Microsoft SQL Server Analysis Services, you need to enter the cube information. In the Friendly Connection Name box (see Figure 22-13), type a name for your connection (PowerPivot_Project, for instance), enter the server name in the Server or File Name field, and choose how you want to log on to the server. Finally, type the name of an existing cube (BL_Cube2) in the Database Name field. (You can also choose the name of the cube from the list of existing cube names that appears for the particular server.) Click Next.

Figure 22-13  The Connect to Microsoft SQL Server Analysis Services wizard page
In the next step, you specify an MDX query to select data to import from the data source. You can type (or paste) such a query and click the Validate button, or leave the creation of a query to the Query Designer. (MDX will be described in a moment.)

Click the Design button to graphically create your query. The window of the Table Import Wizard appears (see Figure 22-14). Now you can choose measures from existing fact tables and fields from dimension tables, which you need for your calculations. As shown in Figure 22-14, drag and drop the Sales Amount measure from the Fact Reseller Sales fact table and the Date Key and Product Key measures from the Due Date and Dim Product dimension tables, respectively. Click OK.

The system shows you the corresponding MDX query, which you can now save if you want. Click Finish to end your task. The summary field shows you the success (or failure) of the process.

The PowerPivot for Excel window shows you the selected data (see Figure 22-15). Now you can use one of Excel's presentation forms to present your data. This example explains how you can create cross tabs (pivot tables) from your data to present the data.
in Excel. In the PivotTable for Excel ribbon, click PivotTable on the Reports tab. The Create PivotTable dialog box appears. Choose New Worksheet and click OK. The new sheet appears in Excel.

In the right pane, called PowerPivot Field List, you can choose the columns that will be presented in the sheet. (If the Field List doesn't appear, click Field List on the ribbon.) In this example, the measure and the two columns from the dimension tables are checked. Now, you can present data in the sheet by dragging and dropping it into the Row Labels, Column Labels, or Values box. In Figure 22-16, the key column of the Dim Product dimension table is dropped in the Row Labels area, while the key column of the Dim Date dimension table is dropped in the Column Labels area. (The measure is dropped in the Values area.)
NOTE
The data presented in Figure 22-16 shows the calculation using the COUNT aggregate function. If you want to modify the form of calculation (to add the values, for instance), right-click a blank area in the Values box and choose the appropriate function (in this case, SUM).

Querying Data Using Multidimensional Expressions
Multidimensional Expressions (MDX) is a language that you can use to query multidimensional data stored in OLAP cubes. (MDX can also be used to create cubes.) In MDX, the SELECT statement specifies a result set that contains a subset...
of multidimensional data that has been returned from a cube. To specify a result set, an MDX query must contain the following information:

- One or more axes that you use to specify the result set. You can specify up to 128 axes in an MDX query. You use the ON COLUMNS clause to specify the first axis and the ON ROWS clause for the second. If you have more than two axes, the alternative syntax is to use the numbers: ON AXIS(0) for the first axis, ON AXIS(1) for the second one, and so on.
- The set of members or tuples to include on each axis of the MDX query. This is written in the SELECT list.
- The name of the cube that sets the context of the MDX query, specified in the FROM clause of the query.
- The set of members or tuples to include on the “slicer axis,” specified in the WHERE clause (see Examples 22.1 and 22.2).

**NOTE**

The semantic meaning of the WHERE clause in SQL is different from its semantic meaning in MDX. In SQL it means filtering of rows by specified criteria. The WHERE clause in MDX means slicing the multidimensional query. While these concepts are somewhat similar, they are not equivalent.

Example 22.1 will be used to explain the syntax of the language. You can execute your MDX queries directly in SQL Server Management Studio. Use the MDX Query Editor to design and execute statements and scripts written in the MDX language. First, type scripts in the query editor pane. Then, to execute the scripts, press F5 or click Execute on the toolbar.

**EXAMPLE 22.1**

Display for each customer total product costs that are due on March 1, 2007:

```sql
SELECT [Measures].MEMBERS ON COLUMNS,
       [Dim Customer].[Customer Key].MEMBERS ON ROWS
FROM BI_Cube
WHERE ([Due Date].[Date Key].[20070301])
```

Example 22.1 queries data from the BI_Cube cube. The SELECT list of the first query axis displays all members of the Measures dimension. In other words, it displays the values of the Total Product Costs column, because the only existing measure in
this cube is **Total Product Costs**. The second query axis displays all members of the
**Customer Key** column of the **Dim Customer** dimension.

The FROM clause indicates that the data source in this case is the **BI_Cube** cube.
The WHERE clause “slices” through the **Due Date** dimension according to the key
values using the single date value 2007/03/01.

Example 22.2 shows another MDX query.

### EXAMPLE 22.2

Calculate for the customer with the customer key 11741 and for the product with the
product key 7 the total product costs that are due in March 2007:

```sql
SELECT [Measures].MEMBERS ON COLUMNS
FROM BI_Cube
WHERE ({[Due Date].[Date Key].[20070301]:[Due Date].[Date Key].[20070331]},
       [Dim Customer].[Customer Key].[11741],
       [Dim Product].[Product Key].[7])
```

The SELECT list in the query in Example 22.2 contains only the members of the
**Measures** dimension. For this reason, the query displays the value of the **Total Product
Costs** column. The WHERE clause in Example 22.2 is more complex than in Example
22.1. First, there are three slices, which are separated using commas. Only one member
of the **Customer** dimension and one member of the **Product** dimension are used
for slicing, while from the **Due Date** dimension, the dates from 2007/03/01 through
2007/03/31 are sliced. (As you can see from the query, the : sign is used to specify a
range of dates.)

### NOTE

MDX is a very complex language. This section provided only a concise description of the language. Use Books
Online to learn more about this language. Also, I highly recommend “MDX for Everyone” by Mosha Pasumansky,
one of the founders of the MDX language. You can find this article at www.mosha.com/msolap/articles/
MDXForEveryone.htm.

### Security of SQL Server Analysis Services

SQL Server Analysis Services security issues correspond to the security issues of
the Database Engine. This means that SSAS supports the same general features—
authorization and authentication—that the Database Engine does, but in a restricted form.
Authorization defines which user has legitimate access to SSAS. This issue is tightly connected to the operating system authorization. In other words, SSAS imposes user authorization based on the access rights granted to the user by the Windows operating system.

You can limit the number of users that can perform administrative functions for SSAS. You can also specify which end users can access data and delineate the types of operations they can perform. Additionally, you can control their access at different levels of data, such as the cube, dimension, and cube cell level. This is done using roles.

As you already know, a database role specifies a group of database users that can access the same objects of the database. Each role is defined at the SSAS database level and then assigned to cubes. After that, individual users or other roles are assigned to that role.

To create a role, right-click the Roles folder in the Solution Explorer pane and choose New Role. Change the default name of the role in the Properties window. After that, in the main window, shown in Figure 22-17, you can use the various tabs to specify different controls for the role. On the Membership tab, you can specify all users who should be members of the role. (Figure 22-17 shows several users added to the BI_Role role.) The Data Sources and Cubes tabs specify the data sources—that is, cubes that can be used by the role’s members. Authorization for the specific cell can be assigned using the Cell Data tab. The Dimensions and Dimensions Data tabs specify which dimensions (dimensional data) can be accessed.

![Figure 22-17 Creation of a new role for SSAS](image-url)
Summary

With its SQL Server Analysis Services, Microsoft offers a set of data warehousing components that can be used for entry- and intermediate-level data analysis. The main component of SSAS is Business Intelligence Development Studio (BIDS), which is based on Visual Studio and gives users an easy way to design and develop data warehouses and data marts.

SSAS is wizard oriented, with wizards for almost every task that is executed during the design and implementation of a data warehouse. The Data Source Wizard allows you to specify one or more data sources, while the Cube Wizard is used to create a multidimensional cube where aggregate data is stored. To import tables you can use the Table Import Wizard, and the Aggregation Design Wizard is used to help you design and create aggregations optimally.

To deliver analytic data to users, you can use any of several different interfaces. The most important are SQL Server Management Studio, MDX, and OLE DB for OLAP.

The next chapter describes SQL/OLAP extensions in Transact-SQL.

Exercises

E.22.1

Using SQL Server Management Studio and the BI_Cube2 cube, find the amount of sales for all products for a customer with the customer number 11111.

E.22.2

Using SQL Server Management Studio and the BI_Cube2 cube, find the total product costs for all customers and for the product with the product number 14.