**Introduction**

This is the first article of my new series **Introduction to MSSQL Server 2000 Analysis Services**, which I hope will help new users get up to speed quickly on this exciting functionality. The series is designed to provide hands-on application of the fundamentals of MS SQL Server 2000 Analysis Services (to which I will refer in many cases as simply **"Analysis Services"** to save time and space); our primary focus will be the creation of simple multi-dimensional OLAP cubes, with each installment progressively adding more features designed to meet specific real-world needs.

Having examined the use of the wizards that come packaged with Analysis Services in our first session, we will move forward to more advanced concepts in later articles; through the creation of samples of "real-world" OLAP cubes, each as a discrete project, we will gain a foundation from which to expand the rudiments learned up to that point to our individual reporting and analysis needs. Articles within this series will cover working with dimensions as a part of cube builds, with a special emphasis on time dimensions and parent-child dimensions. Once we've covered the basic components of Analysis Services and multi-dimensional cubes introduced in each session, we will take a look at examples we might find in our own reporting environments.

In this article, we will:

* Prepare Analysis Services, as well as our environment, for the cube model we intend to design;
* Create the basic cube model;
* Perform dimension design and other steps as part of the cube creation process;
* Save the model;
* Design storage for the cube we have planned;
* Process the cube; and
* Overview basic cube browse functionality.

**Introducing Analysis Manager**

When we installed MSSQL 2000 Analysis Services, **Analysis Manager** was also installed as a tool for Analysis Server administration. Like Microsoft SQL Server Enterprise Manager, the interface that we see when using Analysis Manager is a snap-in administrative utility. A console tree appears in the left pane of the management console, where we view the familiar hierarchical structure that resembles Windows Explorer.

To start Analysis Manager, we go to the Start button on the desktop, then to the Programs group, from which we then need to point to the Microsoft SQL Server Group. Within MSSQL Server 2000, we see the Analysis Services Group: here we click Analysis Manager. (Paths may differ, of course, based upon choices made during the installation of the product on our individual machines).

Once inside Analysis Manager's console, we can see all the analysis servers established for our environment on the left-hand side. The example screen shot shown below (in **Illustration 1**) shows only one server (MOTHER), because I am providing a view of a simple laptop implementation. The analysis server name is automatically derived from the installation of MSSQL Server 2000, and is likely to be the name of the physical machine upon which each of us is working, provided defaults were accepted throughout the MSSQL Server 2000 installation. I will use this environment for most of the pictures in this series.

  
**Illustration 1: First Look at the Analysis Manager Console**

[Page 2: Setting up the Database and Data Source](http://www.databasejournal.com/features/mssql/article.php/10894_1429671_2)

**Setting up the Database and Data Source**

Before we can design a new cube, we need to set up a ***database*** -- more specifically, in Analysis Manager, we need to set up an **OLAP database**. The OLAP database we create will organize cubes, roles, data sources, shared dimensions, and other objects that we will cover in later segments. We will call our OLAP database MyFirstCube (be careful here -- you cannot rename a database in the Analysis Services console once it is created), setting it up initially by right-clicking on the server we see at the left of the console. Keep in mind that, among other objects, the database can contain multiple cubes, each with a single data source. In addition to setting up our database, we will link a **data source** into our database before we start to construct our cube.

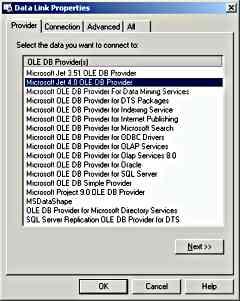
We right-click on the Analysis Server name (shown as MOTHER above). As shown in **Illustration 2** below, the dialog box appears, and we fill in the Database Name, MyFirstCube for this session, and a description that might be of value to a user down the road. Let's simply add "Initial exploration of cube creation in MSSQL Server 2000 Analysis Services," here -- it is certainly optional.

  
**Illustration 2: The Dialog Box for Naming Our Database**

Click OK, and note that MyFirstCube appears in the left-side tree, complete with predefined, empty folders for object storage later. We will need to connect to the data source from which we wish to draw values -- a simple act if you've ever dealt with data sources in Windows before -- if not, we'll get there together in short order. The data source will need to be available via the OLE DB providers that are offered, as we shall see -- with the **Microsoft OLE DB Provider for ODBC Drivers** acting as a possible choice for older systems that might not have had the benefit of OLE DB's debut.

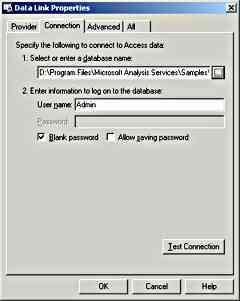
We will at this point expand MyFirstCubes database folder, (clicking on the plus (+) sign to the left of the folder will do the trick), right click the folder within MyFirstCube called Data Sources, and on the popup shortcut menu, we will select New Data Source.

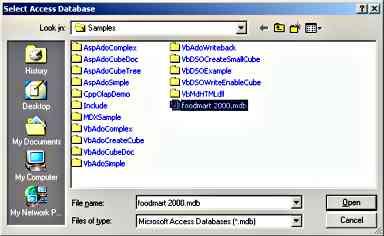
Next, we see the Data Link Properties dialog box appear, with its Provider, Connection, Advanced and All tabs, as shown in **Illustration 3** below. We will select the **Microsoft Jet 4.0 OLE DB Provider** (the native **MS Access OLE DB provider**). Keep in mind that, while OLE DB Drivers for ODBC will often provide a secondary means of obtaining a connection, the OLE DB Driver (assuming one exists) is certainly typically the more optimal choice from a performance standpoint. (For more on this, review the on-line documentation for OLE DB Providers, and data sources in general).

  
**Illustration 3: The Data Link Properties Dialog - Provider Tab**

If we consider what a cube is doing, organizing and storing data for analysis, it becomes obvious that the cube needs to get its data (typically values from a fact table in a star schema data warehouse / data mart arrangement) from somewhere. Our data source will be the FoodMart 2000 sample .mdb provided with Analysis Services.

Click Next, and the focus moves to the Connection tab. We will select the FoodMart 2000.mdb database, located by default in the [Install Directory]:\Program Files\Microsoft Analysis Services\Samples directory, by clicking on the ellipses (...) button, as shown in **Illustration 5** below, then navigating to the FoodMart 2000.mdb database. We then highlight FoodMart 2000.mdb, and click Open, returning to the Connection tab.

  
**Illustration 4: The Data Link Properties Dialog - Connection Tab**

  
**Illustration 5: Selecting the FoodMart 2000.mdb Database**

Next, leaving the other setpoints at their defaults for this session, we can test the connection by clicking the Test Connection button in the bottom right-hand corner of the Connection tab; we should get a verification dialog confirming that the source has been established in our definition (as shown in **Illustration 6**). Click OK, and the verification (Microsoft Data Link) dialog box closes.

  
**Illustration 6: Data Source Verification Dialog**

Now, when we click OK on the Data Link Properties tab, the Properties dialog box closes, and we can see that the source appears under the Data Sources folder in the tree area on the left side of the management console, probably displaying the actual file name.

We might want to make this a more intuitive -- or at least shorter -- name in order to keep a tidy appearance. A simple "rename" capability is not in the cards -- unlike circumstances in a similar looking, but functionally different, Windows Explorer scenario. A right-mouse click on our new data source will, however, allow a Copy action. We will right-click our new data source, click Copy, highlight the Data Sources folder, and then right-click the MyFirstCube database folder we created earlier. Next, well select Paste from the popup menu, causing Analysis Services to indicate that a duplicate has been detected, and to prompt us for a unique name to rectify the confusion. We will respond to the new name request with MyFoodMart, using the dialog box that appears (as shown below in **Illustration 7**).

  
**Illustration 7: Changing the Name of the Newly Copied Cube as a Means of Renaming**

The Duplicate Name dialog thus acts as our agent of change, and, once we click OK, does our bidding and adds the newly named data source under the data sources folder. We have only to delete the original data source (right-click it, and select Delete on the popup menu, then click the Yes button) to organize our new data source folder.

We now have an **OLAP database** in place, linked to a valid **data source**. The next two steps in our exploration will be to introduce the **Cube Wizard** and to design a simple cube.

[Page 3: Designing Our Cube](http://www.databasejournal.com/features/mssql/article.php/10894_1429671_3)

**Designing Our Cube**

Cube design first entails a determination of what we want to capture as **measures**, or the quantitative values from our database that we want to analyze / monitor as indicators of business activity. "Actual" measures, such as revenues / sales, as well as expense / costs, are typically desirable, along with "Budget", "Plan", or "Forecast" values, to achieve a good analysis of organizational performance.

Moreover, measures are analyzed against the different dimension categories of a cube. **Dimensions** represent perspectives, or "views" within the context of which measures have relevance and (hopefully) actionable meaning. The lowest levels of detail for the values that we choose as measures typically reside in a more-or-less relational **fact table**. While operational data often comes from a variety of original data repositories, the most common way of managing relational data for multidimensional reporting in the business environments of today is with a star schema-based warehouse / mart, or similar storage concept. A **star schema** in its simplest form consists of a single **fact table**, linked to multiple **dimension tables** through a common key or keys shared between each member of a linked-table pair. While the many possible variations of the basic star schema are mercifully beyond the scope of our excursion into Analysis Services in this series, we can, for the purposes of our examination, rely upon the circumstance that the Analysis Services cube we design will have only one fact table, with a simple arrangement of dimensions that will be selected, across which to analyze its measures.

Our next objective is to select a few measures from our data source's specified fact table, and to also stipulate the dimensions we wish to capture. One dimension that is virtually always useful is a **time dimension**. We will specify a time dimension, together with a few others, to illustrate the design and construction of a simple cube. We are fortunate in this pursuit to be assisted by the Analysis Services Cube Wizard, which prompts us to make selections of these values in a straightforward manner.

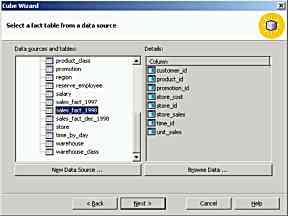
**Selecting the Fact Table and Measures for Our Cube**

The Cube Wizard first guides us through the selection of a single fact table; once we tell it where to find this table, we are prompted to select the measures, or numerical values, that our cube will summarize for analytical purposes. We begin our design process with the following steps.

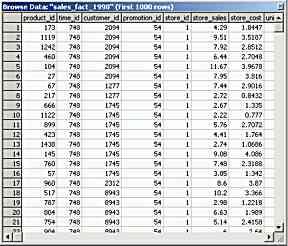
We initialize the Cube Wizard with a right-click to the Cubes folder that appears within the MyFirstCube database we created in the foregoing steps. We will select New Cube, then Wizard, from the popup menu, to raise the Cube Wizard Welcome dialog box shown below in **Illustration 8** (which we can skip by checking the box at the lower bottom half of the Welcome dialog, once we become old hands at the process).

  
**Illustration 8: The Cube Wizard Welcome Dialog**

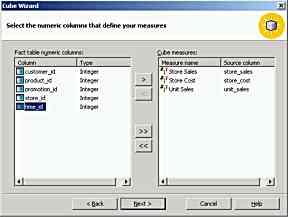
We click Next, and are presented with a list of tables from our newly established data source, MyFoodMart, from which we will select the sales\_fact\_1998 table. As shown below (**Illustration 9**), we are then presented with the various columns in the table.

  
**Illustration 9: The Cube Wizard Welcome Dialog**

We can also verify our selection by taking a peek at the actual data via the Browse Data... button, as pictured in **Illustration 10** below. We will select the table by clicking Next, at which point we are greeted with a dialog box that prompts us to select the contained measures that we wish to incorporate into our cube design.

  
**Illustration 10: A Small Subset of Cube Wizard Browse Data Results**

Let's select the store\_sales, store\_cost, and unit\_sales measures, as shown in **Illustration 11**. We click Next, and the Wizard moves into the Dimension selection phase of the design process. Having selected the values that we want the cube to summarize for us, we now need to tell it what dimensions we want to use as perspectives, or views, of those values / measures.

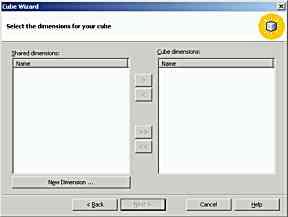
  
**Illustration 11: Our Selected Measures in the Cube Wizard Column Selection Dialog**

[Page 4: Making the Cube Multi-Dimensional](http://www.databasejournal.com/features/mssql/article.php/10894_1429671_4)

**Making the Cube Multi-Dimensional: Creating Meaningful Perspectives in our Cube**

Each of the \_id columns we saw in the last dialog box represented dimensions -- in the form of key columns that might be joined to dimension tables from the fact table in the MyFoodMart star schema. These joins provide a means of allowing us to pull in more details about the dimensions, and, in effect, act as the "link" between fact and dimensions. Much power lies in this concept, and a more in-depth look at the relational concepts surrounding, and the physical attributes and capabilities of, joins might be of interest at another time. Suffice it to say that we now have the opportunity to select existing dimensions -- or to even create new ones -- thus allowing us significant flexibility within the standard operations of the Cube Wizard -- or more accurately, within a second, "child" wizard, the **Dimension Wizard**.

From the "Select Dimensions" dialog that appeared when we clicked Next at the measures selection step shown above (**Illustration 11**), we will now click New Dimension at the dialog box we encounter next, shown below (**Illustration 12**). This results in a launch of the Dimension Wizard, as shown immediately next (**Illustration 13**), whose Welcome dialog can also be skipped going forward (with the check box), should we choose to short circuit the Welcome dialog later.

  
**Illustration 12: Click New Dimension to Start the Dimension Wizard**

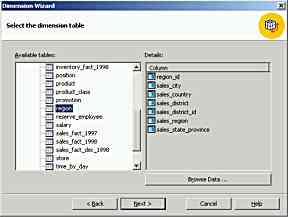
  
**Illustration 13: The Welcome Dialog for the Dimension Wizard**

**Creating a Dimension from a Star Schema: A Single Dimension Table**

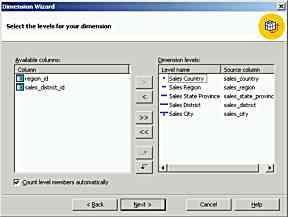
We click Next for now, and arrive at a dialog that asks how we want to create the current dimension. Important, contextually sensitive information is displayed for each of the five possible options. For purposes of our examination of the basics, we'll accept the default, **Star Schema**, as shown below in **Illustration 14**.

  
**Illustration 14: The Dimension Wizard offers Five Options for Dimension Creation**

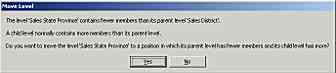
Clicking Next at this juncture brings us to the initial prompt to Select the Dimension Table, from which we will select region. Many attributes of the region dimension appear as details in the right pane, as shown below in **Illustration 15**, and, as with the Fact Table Selection dialog in earlier steps, we can verify our choices by reviewing a snapshot of actual data using the Browse Data... button.

  
**Illustration 15: We Select the Region Table as a Source for our First Dimension**

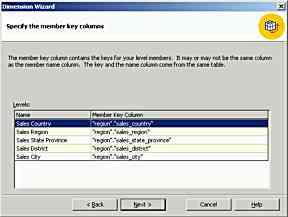
We click Next, and the Wizard prompts us to Select the Levels for ... the Dimension we have chosen, when such levels exist. We can see that various hierarchical levels might be of interest within the region dimension. We select five for this dimension, sales\_country, sales\_region, sales\_state\_province, sales\_district, and sales\_city, then we attempt to rank them in what we think is the logical hierarchy, for optimal reporting purposes, as shown below in **Illustration 16**.

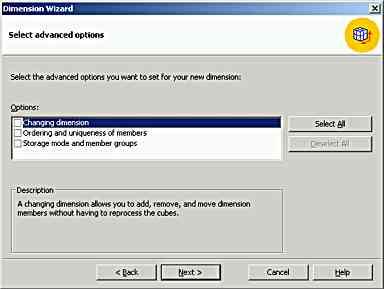
  
**Illustration 16: At the Prompt, we select sales\_country, sales\_region, sales\_state\_province, sales\_district, and sales\_city**

Attempting to rank these in logical order, from summarized to detailed, might be a bit confusing. Knowing the data well is the best defense, but say we weren't sure and made an attempt to move district above province (there are fewer districts than provinces, as the former are, in reality, a subset of the latter). The Wizard warns us that the arrangement might not be the most logical, and provides a valuable tip regarding the best order for reporting purposes, as depicted in **Illustration 17**. We decide to follow the implied advice, and abandon the contemplated re-sort with a click of the Yes button. The Wizard places levels in the order that appears most logical (small to larger, from the top down, in this case) as a result.

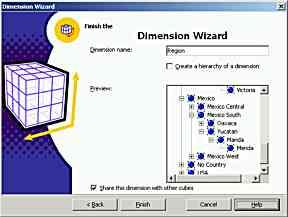
  
**Illustration 17: The Dimension Wizard Offers Advice regarding potential errors in logic**

Once we are returned to the Dimension Selection dialog, clicking Next takes us through the Specification of Member Keys dialog (see **Illustration 18**) -- which broaches topics beyond the scope of our simple overview -- then through the Advanced Options dialog (**Illustration 19**), a topic for discussion later in our series, which we will leave at default setpoints.

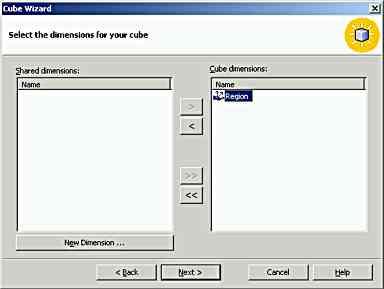
  
**Illustration 18: The Specification of Member Key Columns dialog**

  
**Illustration 19: The Select Advanced Options dialog of the Dimension Wizard**

We next arrive at the Finish the Dimension Wizard dialog, a preview point, where we name the Dimension "Region", simply enough. Using the "+" sign to the left of the topmost level of the dimension tree to expand the levels listed, we can preview the hierarchies that will be created (and check our design, to some extent), as shown in the following picture (**Illustration 20**). We will leave the rest of the setpoints at default for now, and click Finish.

  
**Illustration 20: The Finish the Dimension Wizard Dialog**

We return to the Cube Wizard, where we see immediately that Region appears in the list of Cube Dimensions, as shown below (**Illustration 21**).

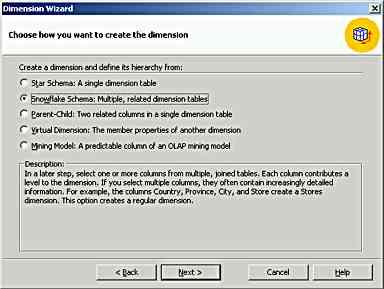
  
**Illustration 21: Back at the Cube Wizard, we see Region as a Listed Dimension**

[Page 5: Creating a Dimension from a Snowflake Schema](http://www.databasejournal.com/features/mssql/article.php/10894_1429671_5)

**Creating a Dimension from a Snowflake Schema: Multiple, Related Dimension Tables**

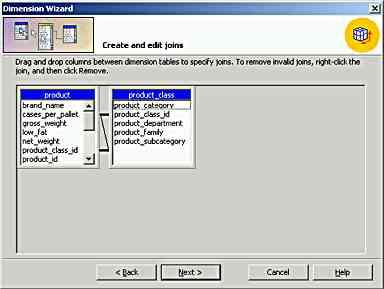
Since the product\_id key in our fact table is joined to a series of Product Dimension tables, and because each of these tables contains columns that are needed to define the respective levels of a multiple-tier hierarchy, we will need to approach this slightly more complex scenario from the perspective of a **snowflake schema** dimension. With the exception of defining joins, and a few additional steps, the Dimension Wizard certainly comes to our assistance in making the process straightforward and efficient.

We begin, once more, by clicking the New Dimension button in the Cube Wizard, where we left it in the previous example. We click Next at the Dimension Wizard's Welcome screen, and then select the Snowflake Schema radio button at the Choose How You Want to Create the Dimension dialog, as shown below in **Illustration 22**. We are given additional information about the choice we have made (in the Description area at the bottom of the dialog), and then we click Next.

  
**Illustration 22: Choosing Snowflake Schema at the Choose How You Want to Create the Dimension Dialog**

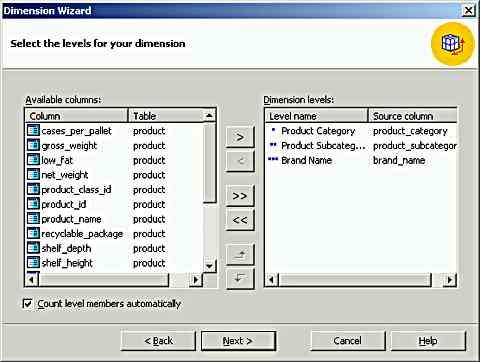
We will select the product and product\_class tables, as shown below (**Illustration 23**), then click Next to continue to the Create and Edit Joins dialog, where we can see the joins (relationships) that exist between the tables we have selected.

  
**Illustration 23: Selecting the product and product\_class Tables to Define Dimensions**

  
**Illustration 24: The Create and Edit Joins Dialog, where we can Review / Edit Table Relationships**

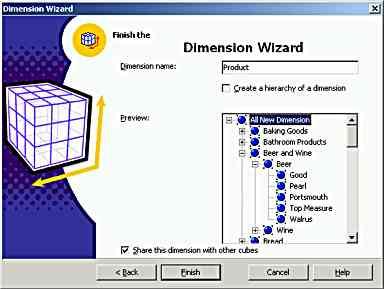
The join between the two tables we have chosen appears appropriate at the product\_class\_id field, as shown in **Illustration 24** above. Data types and other considerations need to be reviewed, or problems will almost certainly result later, but in our simple scenario, the join depicted is as it should be. (For more information on deleting or creating new joins, or on the criteria that dictate correct definition and placement of these relationships, see the online documentation, as well as the Analysis Services volume of the MSSQL Server Reference Library, for starters). Click Next to move forward.

We again select levels for the dimension at the next dialog, with warnings again given if we seem to be placing the levels out of order. We appropriately order them as shown below (**Illustration 25**) from most summarized to most detailed, from top to bottom product\_category, product\_subcategory, and brand\_name.

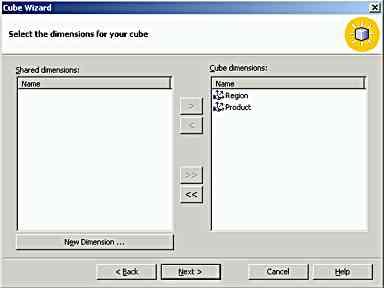
  
**Illustration 25: Selecting Levels for the Product Dimension**

We click Next, again skipping the Specify Member Key Columns for our present purposes, as well as passing over Advanced Options as beyond our immediate scope in this session.

Finally, we arrive at the Finish ... dialog, and enter Product in the Dimension Name box, and leave the other options with defaults selected, as shown in **Illustration 26**.

  
**Illustration 26: Finishing the Dimension Design for the Product Dimension**

After clicking Finish, we again arrive at the Cube Wizard dialog, where, as illustrated below (**Illustration 27**), we see the Product Dimension is now a member of the Cube Dimensions list.

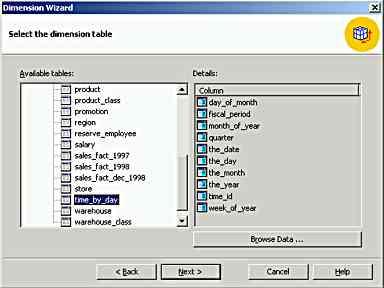
  
**Illustration 27: The Product Dimension now appears in the Cube Wizard Dimension List**

[Page 6: Adding a Time Dimension](http://www.databasejournal.com/features/mssql/article.php/10894_1429671_6)

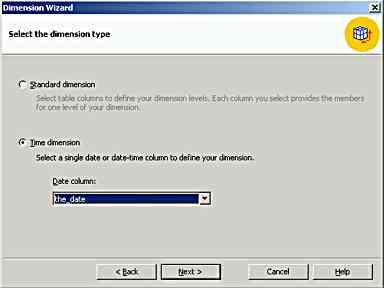
**Just in Time -- Adding a Time Dimension**

We will now focus on what I consider the most common dimension of all -- the **time dimension**. While I practically always add this dimension first, as it appears in virtually every cube I build (time has a way of being important in almost any financial -- and other human -- objective), I saved it until last for our first cube together, to illustrate the considerations that arise with a time dimension.

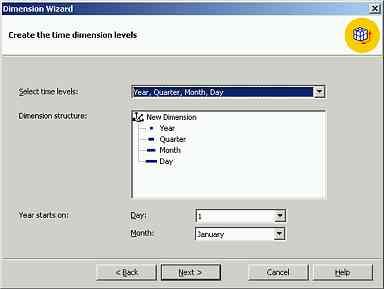
Back at the Select the Dimensions dialog of the Cube Wizard, we once again click New Dimension to summon the Dimension Wizard. We click through the Welcome screen by clicking Next again, and arrive at the Choose how you want to create the dimension dialog box. We select the Star schema: a single dimension table radio button, and click Next. We arrive at the Select the dimension table dialog, where we will select the provided time dimension table, as depicted below in **Illustration 28**.

  
**Illustration 28: The Product Dimension now appears in the Cube Wizard Dimension List**

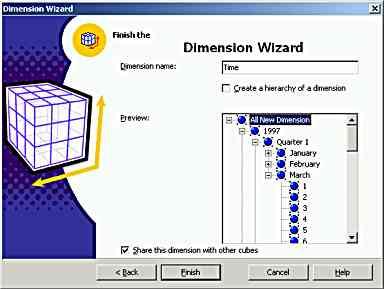
In our example, the FoodMart 2000 database, we have a time dimension table called time\_by\_day. While in the real world, it is sometimes the case that the time dimension is derived from a column in the fact table, more often than not, in a Data warehouse of any sophistication, a time dimension table, such as the one we see here, is in place, and is joined to the fact table by a time\_id (or similar) key. Having selected the time\_by\_day table as shown, we click Next, and are prompted to select a dimension type. The Select the dimension type dialog is presented any time the Dimension Wizard senses a date/time column within the table selection we provide. If there are multiple date columns available, we're given the opportunity to select which column we wish to use to drive our date dimension; if not, the drop down box defaults to the single available date column. In our present exercise, we will select the Time dimension radio button and leave the date column drop down box at the\_date, as shown below (**Illustration 29**).

  
**Illustration 29: The Select the Dimension Type Dialog for a Time Dimension**

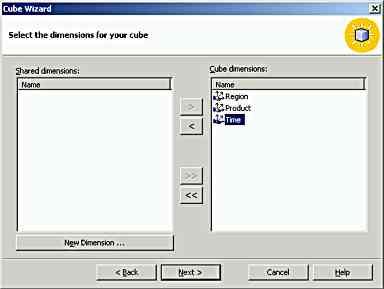
Clicking Next, we arrive at the Create the time dimension levels dialog box. The Dimension Wizard offers numerous common date hierarchy options from which we can choose to save time in constructing our cube. In addition, we are provided an opportunity to select the beginning day and month of the year; this is particularly valuable in building cubes for organizations with fiscal years. We will leave the selections in this dialog box at default, for the sake of simplicity, and assume a fiscal year that coincides with the calendar year (with January the first as its start date). This is illustrated in **Illustration 30**, shown below.

  
**Illustration 30: The Create the Time Dimension Levels Dialog for a Time Dimension**

Because the wizard does not provide for user-selected date dimension levels, per se, when we click Next we skip to the Advanced Options dialog -- a step we will also, again, simply bypass by clicking Next. We then arrive at the Finish the Dimension Wizard dialog, where we can type the word Time, to rename the dimension from default of All New Dimension. Note that the Wizard has proactively created typical time dimension hierarchies for us (year, quarter, month and day) by expanding the hierarchy in the Preview list box. You should see something similar to the illustration excerpted in **Illustration 31**, below.

  
**Illustration 31: The Create the Time Dimension Levels Dialog for a Time Dimension**

Clicking Finish returns us to the Cube Wizard, where we see the addition of our newly crafted dimension appearing alongside the others in the Cube Dimensions list, as pictured below (**Illustration 32**). We have now reached a stage of completeness in the simple model we are using in our examination of the Cube Wizard. We have defined a sample set of dimensions, and we've directed the Wizard where to find the values, or measures, that we want to incorporate into the cube (after specifying the fact table in which those measures reside).

  
**Illustration 32: The Create the Time Dimension Levels Dialog for a Time Dimension**

**Saving the Cube Model as Designed**

Clicking Next in the Cube Wizard at this point results in the appearance of the following message box (**Illustration 33**):

  
**Illustration 33: Message: Time May Be Involved ...**

As we will decide to count the rows in our simple model, we'll click Yes to proceed. The next message we encounter is the following error (**Illustration 34**) -- that is, if we've worked completely in parallel through all the steps I have outlined in this session. ***My intention is to illustrate a point in a meaningful way -- as forgetting to ensure the presence of joins is a very common mistake for early cube architects to make.***

  
**Illustration 34: An Error Message to Understand ...**

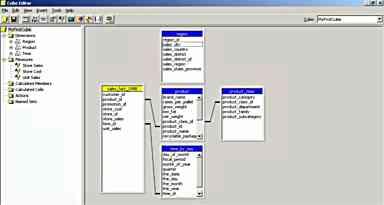
[Page 7: Saving the Cube Model (Continued)](http://www.databasejournal.com/features/mssql/article.php/10894_1429671_7)

We saw, in the earlier section, Creating a Dimension from a Snowflake Schema: Multiple, Related Dimension Tables above, that the Wizard provided a step in our creation of the Product Dimension (see **Illustration 24** in the section concerned) that it did not provide us in the design steps of our first dimension (Region, in the Creating a Dimension from a Star Schema: A Single Dimension Table section). Since the Star Schema option typically applies to a single table, relationships / joins don't factor in as a consideration -- at least not at the time of specification. The issues arise when the Wizard attempts to process the cube and determines that no "obvious" (typically consisting of like--named keys in table pairs) joins exist between the specified dimension and fact tables. We are being told that we will need to manually designate the join(s) so that processing may continue. We will do so, but let's save our work first.

We handle this by first acknowledging the message and clicking OK. Doing so takes us to the Finish the Cube Wizard Dialog Box, as shown below (**Illustration 35**). We input the name of the cube (for purposes of the session, and to save the model we have created). The save takes place when we click Finish, and then we are met with the Cube Editor dialog.

  
**Illustration 35: The Finish the Cube Wizard Dialog Box, where we Save Our Work**

The **Cube Editor** dialog provides us the opportunity to wrap up a couple of loose ends -- specifically to take care of the missing relationship of which we were warned in the error message just before we saved the cube (see **Illustration 34**). We'll add the missing information by performing a simple join between the only isolated table (region) and the fact table (sales\_fact\_1998), as they appear in **Illustration 36** below-- Or will we?

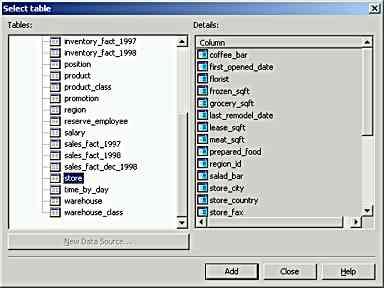
  
**Illustration 36: The Cube Editor -- Where we will add the Necessary Join(s)**

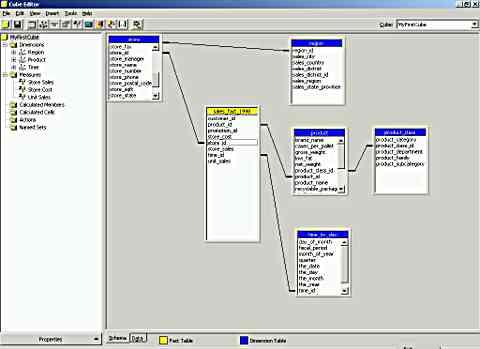
Adding a join is very simple under normal circumstances. But, as anyone familiar with the nature of proper joins knows, we must have a matching key in both tables (to put it simply) to perform the appropriate association between the tables. A closer examination reveals that no common key appears to exist between the sales\_fact\_1998 and the region table. We will need to define and use a "bridge" or intermediate table to accomplish a logical association.

To do this, we will need to add a new table to the group already selected; we will accomplish this with the Insert Table icon in this instance, as shown below in **Illustration 37**. This resurrects the Select Tables dialog that we visited earlier, where we will select the store table (by either highlighting it and clicking Add, or by double-clicking it).

  
**Illustration 37: The Insert Table is a Means of Adding Tables to a Cube**

Next, we will close the Select Tables dialog box (see **Illustration 38**) (it remains open in case multiple selections need to be made -- actually quite convenient for those that use it a great deal). At this point, we see the store table has appeared alongside the other selected tables (see **Illustration 39**), complete with a wizard-induced join between the region and store tables (the wizard makes "obvious" choices by default. We always need to examine its accuracy, but in this case the association of region\_id is certainly correct.

  
**Illustration 38: The Select Tables Dialog, where we Add the store Table**

  
**Illustration 39: Viewing the Newly Added store Table in the Cube Editor**

Now we need to handle the other side of the "bridge" to sales\_fact\_1998. We do so by locating the appropriate key pair (again, an easy choice, as both tables contain store\_id), highlight store\_id in the store table, and drag over with the mouse to the sales\_fact\_1998 table, where we drop onto the corresponding store\_id field. (Should you accidentally connect the wrong two fields, deleting the bad join is as simple as highlighting it by clicking directly on it with the mouse, then right-clicking for a Remove option in the pop-out shortcut menu. Clicking this puts us back to the original state, giving us an opportunity to make the corrections we need.)

We then close the Cube Editor by selecting Exit from the File menu item. We are prompted to save the cube after our changes, to which we respond by clicking Yes. We are then prompted to Design Storage Options in the Design Storage dialog. At this point, we can either save and come back later, or click Yes to design storage now. We will continue to design storage by clicking Yes.

[Page 8: Designing Storage and Processing the Cube](http://www.databasejournal.com/features/mssql/article.php/10894_1429671_8)

**Designing Storage and Processing the Cube**

Analysis Services offers multiple storage options for the data and the aggregations of data in our cubes. These types, or modes, of storage include:

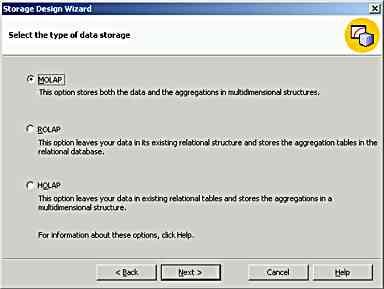
* Multidimensional OLAP (MOLAP)
* Relational OLAP (ROLAP)
* Hybrid OLAP (HOLAP)

Having completed the structural design of our new cube, we will need to designate one of the above **storage modes**, as well as to tell Analysis Services how to generate **aggregations**, or to precalculate summaries, to improve query performance and to otherwise enhance cube efficiency. Next, we'll process the cube, which will load data from the data source we have designated, and which will generate the summary calculations that we have defined in our aggregation instructions.

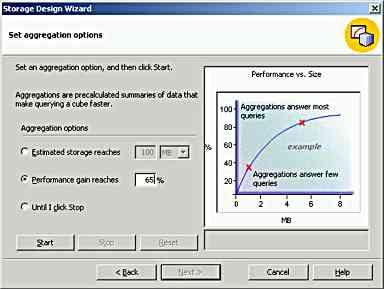
To design storage, we will access and use the Storage Design Wizard. If we clicked Yes at the last dialog, we arrive at the Welcome to the Storage Design Wizard dialog shown in **Illustration 40** below. (If, alternatively, we had saved the cube and exited the process at the dialog, we will need to access the Storage Design Wizard from the Analysis Manager console. This is done by right-clicking on MyFirstCube, the selecting Design Storage from the popup menu, at which point we are greeted with the Welcome to the Storage Design Wizard.)

  
**Illustration 40: The Design Storage Wizard Welcome Dialog**

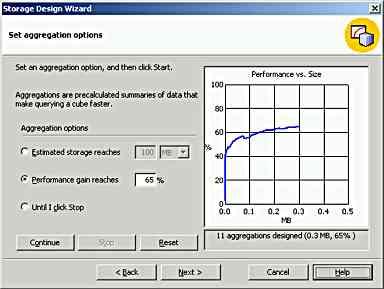
Clicking Next takes us to the Select the Type of Data Storage dialog, where radio buttons with each of the modes we discussed above are presented, along with a brief discussion of the meaning of each (more information is available in Books Online and the Reference Library). We'll select the default, MOLAP, as our data storage mode (as depicted below in **Illustration 41**), and then click Next.

  
**Illustration 41: The Select the Type of Data Storage Dialog**

We arrive at the Set Aggregation Options dialog, where we tell Analysis Services to provide a performance boost of up to 65 percent, without consideration for the disk space required to do so, by selecting the Performance Gain Reaches radio button and typing in "65" to indicate the desired percent, as shown below in **Illustration 42**. As is obvious, the tradeoff between disk space consumed and overall cube query performance can be managed with this option.

  
**Illustration 42: The Set Aggregation Options Dialog**

We can now click Start and get an idea of the Performance vs. Size considerations of our choice, the tradeoff being, of course, that performance increases require the use of greater disk space. An illustration of possible results (which may vary slightly between environments / hardware) is shown below, in **Illustration 43**.

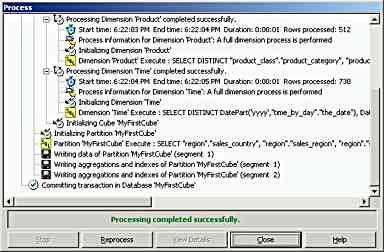
  
**Illustration 43: The Set Aggregation Options Dialog, with Performance vs. Size Chart Display**

We click Next and reach the Finish the Storage Design Wizard dialog box. Under the "What do you want to do?" heading, we select the Process Now radio button, as shown in **Illustration 44**. The next step is to Process the Cube or to Save it for later processing, perhaps after further alterations. We will select Process Now, and click Finish.

  
**Illustration 44: The Finish the Storage Design Wizard Dialog**

The Process window that appears next allows us to monitor the processing of the cube, both in phases, activities, and statistics generated at the bottom of the window. When processing is finished, we should see the "Processing completed successfully" message at the bottom of the dialog, as shown in **Illustration 45**.

The cube having processed with no apparent problems, we return to the Analysis Manager console by clicking Close.

  
**Illustration 45: The Process Window in Action**

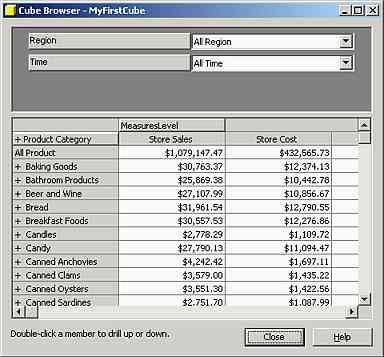
[Page 9: Browsing the Cube](http://www.databasejournal.com/features/mssql/article.php/10894_1429671_9)

**Browsing the Cube**

Once the cube is processed within Analysis Services, the data that we have accumulated in the cube is ready for analysis. We can approach this in many ways and, for the time being, will focus on the "slice and dice" and drill-down capabilities available to us within the Cube Browser. While many tools can be used to access the data for analytical operations, the Cube Browser is provided within Analysis Services to afford us a means of browsing our data without the necessity of adding additional software to our machines.

What's more, the Cube Browser is, in itself, a relatively useful data analysis tool: It makes it possible not only for us to browse the data in our cubes, but to tailor the amount of dimension data that is visible (via **filtering**), to **drill up** (to see less detail / more summary) or to **drill down** (to see more detail, particularly that underlying a chosen summary number). It is particularly useful in checking the accuracy, completeness and effectiveness of our model design at the Analysis Services level, regardless of our ultimate plans for our cubes.

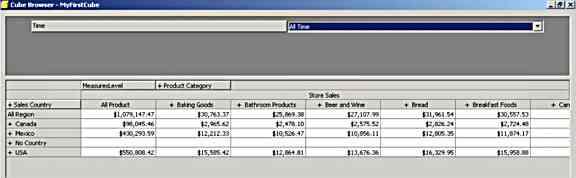
We will begin by going once more to the Analysis Manager console, where we will expand the MyFirstCube data source, then expand the Cubes folder, to see MyFirstCube appear. We right-click the cube, then select Browse Data from the popup menu. The Cube Browser appears at this point, as shown below in **Illustration 46**. We see a single dimension, Product Category, in the initial display, and two measures, Store Sales and Store Cost. Scrolling to the right reveals a third measure, Unit Sales; at the top of Cube Browser, we can see two additional dimensions, Region and Time.

  
**Illustration 46: The Cube Browser, Initial Display**

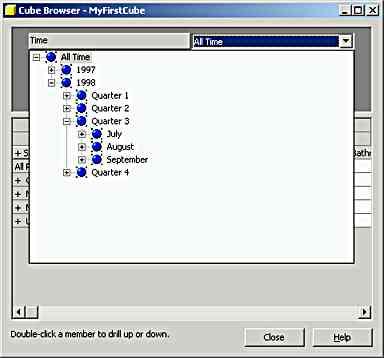
We choose to change our dimensions' positions by dragging the Region dimension at the top of the Browser to the top of the Product column, watching for the appearance of a double-ended arrow pointer, then by dropping the Region dimension onto the top of the column. The resulting grid appears below in **Illustration 47**. Notice, too, that the Product dimension has assumed the original position of the Region dimension, atop the Browser. We can now see the aggregate Store Sales of each country, side by side with the associated country's Store Costs and Unit Sales. We note that the Time dimension at the top indicates "All Time," meaning that the measures we see apply to all time measured in our cube. The dimensions default to highest summary level from the outset of the browse, and we can always tell their relative setpoints, and thus the perspective of the cube, simply by glancing at the dimensions' setpoints atop the Browser. Keep this in mind anytime you attempt a Browse, to prevent becoming "lost" in the cube at any point.

  
**Illustration 47: The Cube Browser, After Changing Positions of Dimensions -- also Scrolled to Right to Display Other Measures**

Next, say we wish to see the makeup of the Sales and Unit measures from a Product perspective, and thus want to add, rather than replace, a dimension to the grid. We accomplish this by simply dragging the Product dimension to the center of the grid. When we drop the Product dimension here, we see our Store measures break out the individual product components automatically in the grid, as illustrated below (**Illustration 48**). We also note that the All Products column remains, acting as a "totals" column, as well as a basis for numerous calculations and analyses.

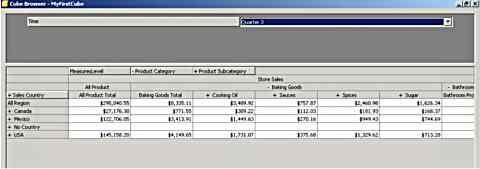
  
**Illustration 48: The Cube Browser, after Adding the Product Dimension**

We perform a filter process by simply clicking the arrow to the right of the Time dimension, and expanding the resulting hierarchical icons down to the third quarter of 1998, as shown in **Illustration 49**. We click the Quarter 3 icon, and see it appear in the Time dropdown box at the top of the browser. Additionally, we see the measures in the grid adjust to reflect the values that belong to the filtered time frame. (**Illustration 50**).

  
**Illustration 49: Selection of the Third Quarter of 1998 in the Time Dimension Hierarchy**

  
**Illustration 50: Results of Filtering through the Time Hierarchy based upon the Above Selections**

Next we'll perform a **drill down**, seeking lower levels of summary than we currently observe. While we'll explore many more features of the drill down process in future articles, for now we'll keep things simple. We double-click on the Product Category cell (the fact that it displays a "+" sign indicates its capacity as a drill down point), and note that drill down occurs to the Product Subcategory level (itself a drill down point, as it displays a "+" at the left of the cell), breaking down the measures in the grid to their respective subcomponents, as one might anticipate (see **Illustration 51** below).

  
**Illustration 51: Results of Drilling Down from the Product Category to the Product Subcategory**

We can always **drill up** simply by double-clicking the newly expanded Product Category cell. Note, too, that the presence of the Category and Subcategory cells, much like the **dimension setpoints** we noted earlier, serves to alert us at any given time "where we are" in the drill down hierarchy.

We could certainly go much further, drilling down, for instance, on the various countries through Regions, Provinces, Districts and Cities, to reach the level of analysis that might be most meaningful. The combination of various drill downs with combinations of dimensions ("**slice and dice**") can provide sophisticated information about the relationships we arrange for analysis, and is the essence of OLAP reporting. And we can always drill up again, as we noted earlier, to return to higher levels of summary as our needs dictate.