CHAPTER 18

Editing the Business Area
In this chapter we will expand on what you learned in Chapter 17 and show you how to continue with the remaining business area checks. We will show you how to manage lists of values and joins. Later we will show you how to add new folders, followed by managing hierarchies. Finally we will show you some of the more advanced mechanisms for item management, culminating in an explanation of how to manage descriptive lists of values.

**Remaining Business Area Checks**
The last two checks that we need to make on a new business are

- Managing lists of values
- Managing joins

**Managing Lists of Values**
The next stage to successful business area management is to create lists of values. Depending upon whether you allowed Discoverer to create its own lists of values during the import stage determines whether you will have to create your own list of values or check existing lists of values. As we advised you not to allow Discoverer to automatically create lists of values, we will start by showing you how to create these.

**Defining Lists of Values**
If you followed our advice and did not allow Discoverer to create its own lists of values during the import stage, you get to have some fun and create them now. Just as when riding the Tour de France, you have come over that monstrous climb in the middle of the stage to find that you have a long downhill section before the next climb, so creating lists of values following all that checking comes like a breath of fresh air. This is a chance to catch your breath before the next climb.

In Discoverer’s language a list of values is an item class. According to the 10.1.2 Discoverer Administration Help guide, *item classes* are “groups of items that share some similar properties. An item class enables you to define item properties once, and then assign the item class to other items that share similar properties.”

Now that you have set up the business area with the items you want, it is time to create the lists of values that your users will need when creating their worksheets. Before we show you the steps required to create a list of values, we will discuss how they will be used.

**Lists of Values Usage**
You do not need to place a list of values on every item in a folder. We have rarely seen an occasion when users need to query lists of identifiers. They usually don’t need to see lists of values on dates either. If you recall from our discussions with the Plus tool, Discoverer Plus will display a calendar pop-up whenever a user is offered a parameter that is a date. You, as the administrator, do not need to do anything to enable this behavior, so why go to the trouble of creating a list of values on the date?

Another reason for not allowing Discoverer to automatically create its own lists of values is that during the automatic creation stage Discoverer does not check to see if an existing item class already exists. For example, let’s say you create a new business area and import two objects, one for the Purchase Order header and one for the Purchase Order detail. Both objects may well contain identical items. At the very least you will probably have the PO Number in both. If you...
allowed Discoverer to automatically create its own lists of values, it would have created two, one for the PO Number in the Header folder and one for the PO Number in the Detail folder. This may not sound like too big of a deal, but imagine that you have created a large business area that has ten folders, each of which contains a Product Number. Discoverer will create ten item classes, not a very efficient way of working.

NOTE
We have deliberately used the word “object” in the preceding paragraph because we can import data definitions based on tables, views, or materialized views.

Continuing with the usage of lists of values, we will give you another very good reason not to automatically allow Discoverer to create its own lists of values. Let’s say you have a table that contains ten million rows, not an uncommon scenario. Let’s say the table contains a column called Gender that has two values, Male and Female. If you allow Discoverer to create a list of values based on the Gender column in this table, every time a user clicks the list of values, SQL will execute a select distinct gender clause from that table. In essence you are forcing the database to look for two values out of a ten million row table. Later in this chapter we will show you how to overcome this by using a custom folder to generate this list of values.

Creating a List of Values
Now that we have explained why you should not allow Discoverer to automatically create its own list of values, we will walk you through creating manual lists of values.

To create a manual list of values, use the following workflow:

1. Determine the item for which you want a list of values.
2. Navigate to and right-click the item, and from the pop-up menu, select New Item Class.
3. Discoverer launches the Item Class Wizard and starts at Step 2.
NOTE
If you want to create a list of values using an Alternative Sort or for use in Drill to Detail, these switches can be found on Step 1 of the wizard. To get to Step 1 of the wizard, click the Back button. For this exercise, we will create a standard list of values.

4. The item that you right-clicked in Step 1 of this workflow will be selected automatically. Although you can change it if you wish, we don’t recommend you alter the settings once you have started. Having confirmed that the item checked is the one that you want to use for generating this new list of values, click Next.

NOTE
If you have opted to create a list of values using an alternative sort, Discoverer inserts an additional step into the wizard at this point. Because we will discuss alternative sorts later in the chapter, you should not see this screen at this time.

5. You are in Step 3 of the wizard, or Step 4 if you are creating an alternative sort. In this step, as you can see next, you can optionally associate the new list of values with other items in the business area. Unless you are an experienced Discoverer administrator, we don’t recommend you do this, because you can easily make a mistake and assign the list of values to the wrong item.

6. Click Next.

7. Step 4 of the wizard (Step 5 if you created an alternative sort) is where you assign advanced options. As you can see in the following illustration, you can assign five options:
Chapter 18: Editing the Business Area

8. After you have assigned your advanced options, click Next.

9. Step 5 of the wizard (Step 6 if you created an alternative sort) is where you assign a meaningful name and optional description. We recommend you name these ChannelLOV, ProductLOV, and so on. While not mandatory, we recommend you make use of the description box if you have changed any of the default options. It is acceptable to have spaces and mixed case in the names; however, since the end user will not see
these names, we don’t add spaces. It is really a holdover from the days when few applications allowed spaces, and we have become accustomed to omitting spaces when they are not needed. This step is shown here.

10. When done, click Finish. Discoverer creates the list of values, which you should test by clicking the plus icon alongside the item.

NOTE
If Discoverer gives you a warning saying that this could take a long time, ignore it. As the administrator you need to test that your list of values a) works and b) is efficient! If the retrieval times out, the limit can be increased by going to the Tools | Options Query Governor tab and increasing the default value.

Removing a List of Values
From time to time it will become necessary for you to remove a list of values from the system. This is very easy.

To remove a list of values, use the following workflow:

1. Click the Item Classes tab in the main Administrator window.

2. Locate and right-click the list of values that you want to remove, and from the pop-up menu, select Delete Item Class.
Chapter 18: Editing the Business Area

3. The Confirm Delete dialog box will open.

![Confirm Delete dialog box]

4. When prompted “Are you sure you want to delete this,” you have four options:
   - **Yes**  Click this to delete the item class.
   - **No**  Click this to cancel.
   - **Impact**  Click this to see what impact deleting this item class will have. If Discoverer displays only one item in the Impact dialog box, you are probably safe in deleting the item class. If Discoverer displays more than one impacted item, you might want to reconsider.

![Impact dialog box]

- **Help**  Click this to open context-sensitive help in a browser window.

5. If you click Yes, the item class will be deleted.

**Checking Lists of Values**

If we had allowed Discoverer to automatically create its own lists of values, or if you need to undertake a cursory examination of an existing list, you will need to examine the item class settings.
To check a list of values, follow this workflow:

1. Click the Item Classes tab in the main Administrator window.
2. Locate and right-click the list of values that you want to check, and from the pop-up menu, select either Edit Item Class or Properties.
3. If you selected Edit Item Class, as you can see next, Discoverer opens the Edit Item Class dialog box.

**NOTE**

The Edit Item Class dialog box is a multtabbed box that corresponds to the steps in the Item Class Wizard. After making any changes that are required, click OK.

4. If you selected Properties, as you can see in Figure 18-1, Discoverer opens the Item Class Properties dialog box.

The Item Class Properties dialog box is a form that allows you to view or change the properties of a list of values. If you take a close look at this box, you will notice that there are a couple of additional properties that cannot be set anywhere else. These are explained in the following section.

**Additional Item Class Properties**

In the preceding section we showed you how to examine the properties of an item class by launching the Item Class Properties dialog box, seen in Figure 18-1. In this box there are two additional properties that cannot be set anywhere else:

- **Auto generate name**  
  This is a most interesting property. By default it is set to No, which means that Discoverer allows you to assign your own name for the item class. If you change this to Yes, Discoverer replaces your name with an auto-generated name consisting of the folder name and the item name upon which the item class is based. Unless you switch this property back to No, you will not be allowed to edit the name again. Furthermore, although we cannot show this to you in a black-and-white illustration, Discoverer colors the name of the item class on the Item Classes tab. If you...
redisplay the Item Class Properties, you will notice that the name is now grayed out. This is to indicate that the item class name is non-editable.

- **Identifier** Discoverer assigns unique identifiers to item classes. By default, this is the uppercased value of the item class name. There is no need to change this name unless your corporate standards dictate.

**NOTE**

Even though setting the Auto generate name property to Yes causes Discoverer to rename the item class, this has no impact on the Identifier. The Identifier can be changed only manually.

**Required Lists of Values**

In our tutorial business area, we need to create 14 lists of values. The following table shows the lists of values we will create:

<table>
<thead>
<tr>
<th>Folder Name</th>
<th>Item Name</th>
<th>List of Value Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales</td>
<td>Status</td>
<td>StatusLOV</td>
</tr>
<tr>
<td>Channel</td>
<td>Channel Name</td>
<td>ChannelLOV</td>
</tr>
<tr>
<td>City</td>
<td>CityName</td>
<td>CityNameLOV</td>
</tr>
<tr>
<td></td>
<td>Country Code</td>
<td>CntryCodeLOV</td>
</tr>
<tr>
<td>Customer</td>
<td>Cust Name</td>
<td>CustNameLOV</td>
</tr>
<tr>
<td></td>
<td>State</td>
<td>StateLOV</td>
</tr>
</tbody>
</table>

Copyright © McGraw-Hill Companies 2006
This is a sample chapter for review purposes only.
Having created the lists of values, click the Item Classes tab. You should see the lists displayed as seen here.

**Issues with Lists of Values**

These are the most common issues that occur with lists of values:

- Lists of values take too long to generate and time out.
- Lists of values show the data in the wrong sort sequence.

If users report that the list of values is timing out or it is not producing the full list, you have the following three possible solutions.

**Increase the Time Out Limit**  If the user is using Desktop or Plus, tell him/her do the following:

1. Launch Discoverer Desktop.
2. From the Tools menu, select Options | Query Governor.
3. Change the option called “Cancel value retrieval after” to a higher value.
4. Click OK.

If the issue is occurring in Plus, do the following:

1. Launch Discoverer Plus.
2. From the Tools menu, select Options | Query Governor.
3. Change the option called “Cancel list-of-values retrieval after” to a higher value.
4. Click Apply.

If the issue is occurring in Viewer, do the following:

1. Launch Discoverer Viewer.
2. The user should click the link called Preferences.
3. Change the option called “Cancel list-of-values retrieval after” to a higher value.
4. Click Apply.
NOTE

Between Plus and Viewer there is only one set of preferences, so if a user changes the retrieval value in one, it is automatically reflected in the other the next time the user connects.

If the issue is occurring in the Administrator edition, do the following:

1. Launch Discoverer Administrator.
2. From the Tools menu, select Options | Query Governor.
3. Change the option called “Cancel value retrieval after” to a higher value.
4. Click OK.

Force the Use of the Search Dialog Box  
Launch the Item Class Properties dialog box and set the property called Display Search Dialog to Yes. This will force the Discoverer end user to be prompted for the item that he or she wants rather than displaying a large list of values.

Use Another Data Item to Create the List of Values  
If neither of the preceding two options rectify the situation, you should try creating the list of values on an alternate item, perhaps from another table. You might also want to consider using a custom folder for this purpose.

NOTE

Another option for improving retrieval times is to optimize the database table by adding an index.

When checking lists of values, you need to be aware that Discoverer creates these based on the following underlying data types:

- Date
- Number
- Varchar or Varchar2

Sorts based on dates and numbers will always be in the correct order. Sorts based on Varchar or Varchar2 data types, strings to you and me, can create problems because they will sort alphabetically. Sometimes this is not the order that we wish them to sort in. The following section on alternative sorts addresses this issue.

Using Alternative Sorts

Sometimes, when you look at a list of values, you will see that it is not in the order that you would like. To overcome this, you can use what is called an alternative sort. In order to use an alternative sort, you must have another unique item available in the folder that contains the item on which you created the list of values.

This alternative item must be one of the following:

- Another unique and available item, such as month end date
- An artificially created sequence number

Among the most common data elements that sort in the wrong order are days of the week and months of the year. For both of these you will need to create an alternative sort if you want your users to see the list displayed in the right order.
For example, look at the following table:

<table>
<thead>
<tr>
<th>Natural Order</th>
<th>Logical Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>April</td>
<td>January</td>
</tr>
<tr>
<td>August</td>
<td>February</td>
</tr>
<tr>
<td>December</td>
<td>March</td>
</tr>
<tr>
<td>February</td>
<td>April</td>
</tr>
<tr>
<td>January</td>
<td>May</td>
</tr>
<tr>
<td>July</td>
<td>June</td>
</tr>
<tr>
<td>June</td>
<td>July</td>
</tr>
<tr>
<td>March</td>
<td>August</td>
</tr>
<tr>
<td>May</td>
<td>September</td>
</tr>
<tr>
<td>November</td>
<td>October</td>
</tr>
<tr>
<td>October</td>
<td>November</td>
</tr>
<tr>
<td>September</td>
<td>December</td>
</tr>
</tbody>
</table>

**Creating an Alternative Sort**

Let’s look at the Month data in the sample database. Open the Month folder and click the plus character alongside the Fiscal Month data item. This opens the current list of values which, as you can see next, is displayed in alphabetical order.

To add an alternative sort, use the following workflow:

1. Click the Item Classes tab.
2. Expand the list of item classes.
3. Locate the list of values that needs the alternative sort. In our case we are looking for the MonthLOV.
4. Right-click the item class, and from the pop-up menu, select Edit Item Class.
5. In the Edit Item Class dialog box, click the Alternative Sort tab.
6. Select the item that will be used to generate the new alternative sort. In our case, we can use the Month End Date because for any one month there is only one month ending date. As mentioned previously, because dates always sort correctly, this is an excellent item on which to sort.
7. Click OK.
8. Recheck the list of values. As you can see in this illustration, the months are sorted correctly.

**NOTE**

Days of the week are trickier because there is no obvious alternative item that you can use to sort by. However, if we think outside the box, we can create an artificial sort using a sequence number as the alternate item. We will show you how to do this when we discuss custom folders later in this chapter.

**Removing an Alternative Sort**

If you make a mistake by setting an alternative sort for an item that does not need this type of sort, you can remove the alternative sort. This is a new feature. If you have used previous versions of Discoverer, you will know how hard it was to remove an alternative sort until now.

To remove an alternative sort, use the following workflow:

1. Right-click the item class that has the alternative sort you want to remove, and from the pop-up menu, select Edit Item Class.
2. Click the Alternative Sort tab.
3. Click the Remove Alternative Sort button.
4. Click OK.
Managing Joins

The next thing we need to do in working with a newly created business area is to create the joins that we need between our folders or check the joins that already exist. As we hope you recall from the creation of our business area, we asked Discoverer to automatically create joins for us based on the primary and foreign key constraints in the database. Looking at Figure 18-2, of the data model that we used for the tutorial database, you see that there are twelve tables, interlinked by a series of one-to-many (1:N) relationships.

**Defining Joins**

In Discoverer, a join relates two folders using one or more matching items, whereas in the database a join relates two tables using one or more matching columns. Just like in the database, where the DBA has to know which table is the parent and which is the child, Discoverer needs to know which folder is the master and which is the detail. What we are looking for are one-to-many (1:N) or one-to-one (1:1) relationships. A many-to-many (N:N) relationship cannot be handled by Discoverer, so these need to be resolved in some other way.
When you create a join between two folders, Discoverer gives end users the rights to pull items out of both folders into the same worksheet. The user does not need to know how the join was created or which folder is the master. This complexity is hidden from the end user by the End User Layer’s metadata. Our job, as Discoverer administrators, is to make life as easy as possible for our end users. We should be looking for ways to make it easier for the end users to build the queries that they need. Users do not have this power, so if we do not create joins, our end users will have a difficult time.

If you look back at Figure 18-2, which outlines the GSW data model, you will see that there are twelve tables, interlinked by a series of one-to-many (1:N) relationships.

Data Types
Discoverer has a rule that the data items on both ends of a join must be of the same data type and size; otherwise, it cannot create the join. The prime responsibility therefore rests with the data designer and the DDL creator to ensure that the items are the same. However, as the Discoverer administrator, if you find that the data items do not match, what can you do?

The real question is—what should you do? Our first piece of advice is to take a deep breath and not panic. These things happen from time to time, and hopefully this data mismatch is just an oversight. Go back to the data designer and the DDL creator to see if they are aware of this situation and ask whether this was intentional. Tact will dictate that you should not use the words bug or bad design when discussing a data model with your data designer. You do need to find out if a data model revision is planned. If the answer is No, and the data model cannot be changed, the question is—what can you do?

Correcting this situation in Discoverer is not the most effective way and will almost certainly cause performance issues. However, by using a function to convert the data type and size of one of the items to match the data type and size of the other item, you can at least create the join. Should you do this? Ideally, no—but sometimes you will have no choice. Workarounds are one of our greatest weapons!

But which item should you change? Speaking logically, you need to change the one that has the least impact. In most master/detail relationships you will typically be joining from the primary key of a master folder to an item in the detail folder that may or may not be associated with an index. You do not need Primary/Foreign key constraints enabled in the database to make this work.

Applying a function to change the data type of an item in the primary key of the master folder could have serious consequences for performance. Doing this will probably negate searches on the primary key and cause the Oracle database to do a full table scan, something you want to avoid if you can. We therefore need to look at the other end of the relationship, at the properties of the detail item. We will cover this later in this section when we discuss creating a complex join.

Creating a Simple Join
Let’s focus on one pair of tables, the GS_Prodline and GS_Products folders. Looking at the data model, we can see that the primary key of the GS_Prodline table is the column called Lineid. When we look inside the GS_Products table, we can see that there is also a column called Lineid. There is a 1:N join between these tables, and the parent table is the GS_Prodline.

Looking inside Discoverer, we have folders that correspond to these two tables. The Discoverer folder corresponding to the GS_Prodline table is called Product Line, while the folder corresponding to the GS_Products table is called Products. We therefore need to create a join from the Product Line folder to the Products folder.
To create a join, use the following workflow:

1. Determine the two folders that need to be joined.
2. Determine the names of the items that need to be joined.
3. Determine which folder is the master and which folder is the detail.
4. Navigate to and open the folder that contains the master item.
5. Navigate to and right-click the master item, and from the pop-up menu, select New Join.
6. Discoverer opens the Join Wizard. Looking at the first screen of the Join Wizard, you can see that it has the following seven areas:

- **Name**  Join names are critical to the successful operation of Discoverer workbooks. Every join must have a unique name, not only in the business area but across the entire EUL. Discoverer will auto-generate a name if you leave this blank.

- **Description**  This is a placeholder for you to type a description about the join.

- **Choose the items to join**  This area comprises three components: a Master Item, an Operator, and a Detail Item. These components are the heart of the join and tell Discoverer exactly how the join is put together.

- **Add**  Use this button to add additional join items, in effect creating a composite join between multiple items. For example, in Apps you might have both a Category Code and a Category Item Code in use joining the two tables together.

Copyright © McGraw-Hill Companies 2006
This is a sample chapter for review purposes only.
Chapter 18: Editing the Business Area

- **Delete** Use this button to delete the secondary components that are in use in a composite join.

- **Next** Use this button to navigate to the second screen of the Join Wizard. It will be active only when the three join items, the Master item, the Operator, and the Detail item, have been entered.

- **Finish** Use this button to complete the join. It is active only in Step 2.

7. Discoverer starts to create the join by inserting, under the heading Master Items, the folder and item names of the item that we selected. Click the drop-down icon under the heading Detail Items. This opens the New Join dialog box.

8. In the New Join dialog box you need to select the business area, folder, and detail item required to complete the join, and then click OK.

9. When control is passed back to the wizard, you should notice that the folder and item name of the detail item have been filled in.

10. While a join name is mandatory, Discoverer will auto-generate the name for you based on the names of the folders. Therefore, you should notice that the Next key is now active. You should also note that the Finish button is still grayed out.

**NOTE**

In version 9.0.4, and continued with 10.1.2, Oracle added a number of SQL optimizations to the SQL generation capabilities of Discoverer. One of these is join trimming, which means that in a query that joins two or more tables, if the query can be answered and still satisfy all constraints by not including a table at all in the SQL, then Discoverer will drop that table from the SQL that it sends to the database. The Next screen when defining a join is for that purpose: to force the Discoverer administrator to make a choice and not simply rely on the defaults. It is therefore good practice to pay more attention to this second screen than ever before.

11. Click Next to proceed to the second screen of the wizard.
12. In Step 2 of the Join Wizard you need to provide information about the join details and set any optional configurations you need. These are the two main areas of this screen:

- **Choose the join details** In this area you must tell Discoverer whether the detail items always exist in the master folder or whether these are optional. You must check one of the two radio buttons. Typically, you will nearly always encounter the situation where the detail items do exist in the master folder. This is the default.

- **Choose any optional configurations** In this area you must tell Discoverer about the join itself, whether there is an outer join, whether the outer join is on the detail or the master, or if the join is a one-to-one (1:1). Normally the configuration is such that the join is not an outer join. This is the default.

**NOTE**

The option for “Outer join on master” is grayed out. This option is available only when you check the button called “Detail item values might not exist in master folder” in the join detail section. Unless you really need to and you know what you are doing, take care adjusting these settings. Making incorrect join settings will result in worksheets that at best run slow or at worst give incorrect results.

13. Click Finish.

14. Discoverer completes the join and closes the wizard.

**Creating Complex Joins**

A complex join occurs in two situations:

- When one of the data types is wrong.
- When multiple items are used to join the folders together.
What follows is a fictitious example of a multi-item join. This is a join between a Product folder and a materialized view that has been created against Sales. After speaking with the data modeler, it was determined that the items shown in the following table make up this join:

<table>
<thead>
<tr>
<th>Master Folder</th>
<th>Master Item</th>
<th>Detail Folder</th>
<th>Detail Item</th>
<th>Join Type</th>
<th>Join Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>Org ID</td>
<td>Sales MV</td>
<td>Org ID</td>
<td>Complex</td>
<td>First</td>
</tr>
<tr>
<td>Product</td>
<td>Prod ID</td>
<td>Sales MV</td>
<td>Prod ID</td>
<td>Complex</td>
<td>Second</td>
</tr>
</tbody>
</table>

As mentioned earlier, the only stipulation that Discoverer has is that the data types of all the items in the join must match. However, in our situation we have a further complication. As defined in the database, the data type of Prod ID in the Product table is NUMBER(4), whereas in the Sales materialized views it is defined as VARCHAR2(4). The lengths are the same, but the data types are not. When we tried building this join, Discoverer responded with this error message:

Clicking OK returned us to the Join Wizard, and we cannot continue with the join. One of the data types must be changed.

**Changing Data Types for a Join** Whenever you have a situation where the data types are not the same and you need to create a join, you need to change the data type for one of the join items.

To change the data type, use the following workflow:

1. Click the heading for the folder that contains the data type you want to change, in this case the Sales MV.
2. From the pop-up menu, select New Item.
3. Discoverer displays the following New Item dialog box:
4. Enter a meaningful, unique name for the item. In our case we will name it New Prod ID.
5. Expand the list of items in the Show box and locate the item you want to change.
6. Paste or push the item into the Calculation box and use a converter function to make the change. Your calculation should look like this:

![Image of Oracle Discoverer 10g interface]

7. Click OK to close the Edit Calculation dialog box.
8. Because the new item was created using the TO_NUMBER function, it will have been created with a default position of Data Point. Using the Properties dialog box, change the default position to Side.

Now you can use this new item in the join.

Creating a Multi-Item Join

Now that we have added a new item with the data type we need for the complex join, we can create the join. We are referring to the new item in the Sales MV folder.

<table>
<thead>
<tr>
<th>Master Folder</th>
<th>Master Item</th>
<th>Detail Folder</th>
<th>Detail Item</th>
<th>Join Type</th>
<th>Join Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>Org ID</td>
<td>Sales MV</td>
<td>Org ID</td>
<td>Complex</td>
<td>First</td>
</tr>
<tr>
<td>Product</td>
<td>Prod ID</td>
<td>Sales MV</td>
<td>New Prod ID</td>
<td>Complex</td>
<td>Second</td>
</tr>
</tbody>
</table>

To create the complex multi-item join, use the following workflow:

1. Use the workflow that was outlined earlier in this section and create a simple join on the Org ID between the Product and Sales MV folders. Do not click Next after you have added this first element, but click Add.
2. Add a second join item between the Prod ID in the Product folder and our new item. Call this second item New Prod ID in the Sales MV folder.
3. The screen should look like this:

4. Click Next, and then click Finish.

NOTE
Do not forget that when you bring tables into Discoverer, each table equates to a folder, and the columns in the table equate to items within the folder.

Checking Joins
From time to time it may become necessary for you to check a join. Let’s expand the Channel folder so that we can see all of the items.

As you can see in this illustration, this folder has only two items, a channel and a channel ID. The channel ID is in bold, while below it is another item with a “crows foot” icon. These are the join items. The channel ID is in bold because it is the primary key in the primary/foreign key relationship.

NOTE
The corresponding foreign key in the Sales order will be italicized. Discoverer bolds all primary keys and italicizes all foreign keys.

Let’s take a closer look at the join. To examine a join, use the following workflow:

1. Right-click the join you want to examine and from the pop-up menu, select Edit Join.

2. Discoverer displays the following Edit Join dialog box:
Oracle Discoverer 10g Handbook

As you can see in the illustration, the Edit Join dialog box has two tabs: one called Items and one called Options.

**The Items Tab**  The Items tab has the following five areas:

- **Name**  Join names are critical to the successful operation of Discoverer workbooks. Every join must have a unique identifier, not only in the business area but across the entire EUL. Even though join names can be duplicated within the business area, we recommend giving joins unique names to match the identifiers. You will find this to be extremely helpful if you need to debug the system at a later date.

- **Description**  This is a placeholder for you to type a description about the join.

- **Choose the items to join**  This area comprises three components: a Master Item, an Operator, and a Detail Item. These components are the heart of the join and tell Discoverer exactly how the join is put together.

- **Add**  Use this button to add additional join items, in effect creating a composite join between multiple items. For example, in Apps you might have both a Category Code and a Category Item Code in use joining two tables together.

- **Delete**  Use this button to delete the secondary components that are in use in a composite join.

**The Options Tab**  The Options tab, illustrated here, has two main areas:

- **Choose the join details**  In this area you must tell Discoverer whether the detail items always exist in the master folder or whether these are optional. You must check one of
the two radio buttons. Typically, you will nearly always encounter the situation where the detail items do exist in the master folder.

■ **Choose any optional configurations** In this area you must tell Discoverer about the join itself, whether there is an outer join, whether the outer join is on the detail or the master, or if the join is a one-to-one (1:1). Normally the configuration is such that the join is not an outer join.

**NOTE**

You will notice that the option for “Outer join on master” is grayed out. This option is available only when you check the button called “Detail item values might not exist in master folder” in the join detail section. Do not adjust any of these settings unless you really need to and you know what you are doing. Making incorrect join settings will result in worksheets that at best run slow or at worst give incorrect results.

Now that you know how to look at the join settings, you need to check that Discoverer has correctly created all of the joins that you were expecting. If all of your joins are in place, well done, you have successfully completed checking your new business area. We will discuss joins in more detail in the next section when we discuss maintaining business areas.

**Enhancing Business Areas**

After you have created a new business area and made all of the necessary checks to ensure that the items you imported and their associated properties are correct, you can begin your enhancements. Unfortunately, many inexperienced Discoverer administrators think that they have finished at this point. You would be amazed at some of the horror stories that we could tell about poorly designed business areas.

We will outline the enhancements that need to be made to complete your business area. We have identified five major areas of work still needed:

■ Folder enhancements
■ Hierarchy enhancements
■ Item enhancements
■ User access
■ Testing

For the remainder of this chapter we will discuss the first three of these major work areas in some detail.

**Folder Enhancements**

The folder enhancements that we will discuss primarily concern adding new folders. There are three types of new folders:

■ **New** This type of folder is the sort that we will use when we wish to create a new, complex folder. A complex folder is one that is made up of items from existing folders. You will use these folders often.
Custom This type of folder allows you to key the SQL yourself directly into Discoverer. We use these folders for manually adding lists of values or alternative sorts.

From Database This type of folder is associated with an object from the database. These objects could be a table, a view, or a materialized view.

Creating New Folders
To create a new folder, follow this workflow:

1. Single-click the business area heading such that the business area name is highlighted.
2. Right-click the business area name; Discoverer will pop up a context menu.
3. From the context pop-up menu, click the type of folder that you want to create.

NOTE
An alternate method for Step 2 is to select Insert | Folder from the menu bar.

Adding Complex Folders
Complex folders are folders that combine data from one or more other folders in the End User Layer. They enable your users to see data from multiple folders at the same time, somewhat similar to the way a view works in the database. You use complex folders to simplify what the user sees when logging in. Rather than seeing numerous interlinking folders, the user will see one folder. If your company has strict rules for creating database views, you may want to consider complex folders because you do not need the CREATE VIEW privilege to create a complex folder. You can also use complex folders to create subsets of the data for different parts of your organization, leaving the original folders intact.

NOTE
To combine data from multiple folders, just as with a database view, there must be joins in place between the folders.

Choosing New Complex Folders How do you choose when to use a complex folder? Take a look back at Figure 18-2, which gives you an entity-relationship diagram for the database that we are using in this book. While this may look like a star schema, it is actually a snowflake schema.

NOTE
You will find information concerning star and snowflake schemas in Appendix C.

Look at the joins that radiate out from the GS Sales folder. You can see that there are four, one each to GS Channel, GS Customer, GS Day, and GS Product, respectively. Beyond the GS Customer, GS Day, and GS Product folders, there are more folders. Beyond Products, there are...
Chapter 18: Editing the Business Area

Product Lines, while beyond Customer there is a complete geographic hierarchy linking to Cities, Districts, and Regions. Looking beyond the Day folder, you can see a time hierarchy linking to Months, Quarters, and Years. We will use complex folders based around each of these hierarchies so that our end users will see one Geographic folder containing all of the items from Customer, City, District, and Region; and one Products folder containing all of the items from Products and Product Lines.

You might ask, “What about the time hierarchy, shouldn’t we create a complex folder on this too?” The answer is yes, we will do this, but we will not build one complex folder but four, one folder for each of the dates that we have in our Sales folder: Cancel Date, Order Date, Requested Date, and Ship Date. Why four complex folders, wouldn’t one suffice? The answer is no. If we want our users to build queries such as “show me how many of the sales orders that were ordered in Q3 did not ship until Q4,” they cannot do this if we have one date folder. However, if we have a separate folder for Order Date, containing our month, quarter, and year data along with the calendar dates, and we have another, similarly defined folder for the Ship Date, our users can build such a query.

NOTE
GSW, our fictitious retailing company, uses a fiscal time calendar running from October to September. Our Q1 is therefore October to December, while Q2 is January to March of the following fiscal year. We thus cannot use the standard Discoverer date hierarchy.

We need to create six complex folders. The data model shown in Figure 18-3 shows the business area model after we have added the new folders. If you think that this looks like a star schema, you would be right. Discoverer works well with star schemas, and this is what you should be trying to do with your models. If you cannot create a star schema in the database, you should try to use complex folders to do this for you in the tool. Users understand star schema data models better than any other kind of data model.

Fiscal Date Complex Folders As mentioned previously, and as seen in Figure 18-3, in our Sales folder there are four dates, one each for

- Cancel Date
- Order Date
- Request Date
- Ship Date

We will create one new folder for each date and populate it from the existing Day, Month, Quarter, and Year simple folders. We will describe only the process for the new Cancel Date folder in detail. Having shown how to do this for Cancel Date, we will then show you how to create the other three folders by copying and pasting what you have just done.
As shown in the following illustration, it is much easier to do the next tasks if you split the screen into two windows using Window | Split Window from the menu bar.
To create the Cancel Date complex folder, use the following workflow:

1. In the right-hand window, right-click the business area name, and from the pop-up menu select New Folder.
2. As you can see next, Discoverer creates a new empty folder named NewFolder1, expands the list of existing folders, and places the new folder at the bottom of the list.
3. Right-click the new folder, and from the pop-up menu select Properties. The Folder Properties dialog box will open. Using the General tab, rename the folder to Cancel Date and change the Identifier to CANCELDATE.

4. Click OK when warned that changing identifiers can be risky.

5. If there is already a folder in the EUL by the name that you are trying to use, Discoverer will give this warning telling you that the name must be unique:

   NOTE
   It is perfectly safe to change the identifier when creating a new folder. Waiting until later, however, when users may well have workbooks based on the folder, increases the risk that workbooks will no longer open and is not recommended.
6. In the left-hand window, expand the Day folder.
7. Click the item called Calendar Date and drag it across the window separator into the new folder.
8. In the left-hand window, minimize the Day folder and expand the Month folder.
9. Click the items called Fiscal Month and Mth End Date and drag these across the window separator into the new folder.

**NOTE**

To drag multiple items at the same time, left-click one of the items to highlight that item, and hold down the **Ctrl** key while you left-click the other items. With all items highlighted, you can drag the set across the window separator into the new folder.

10. In the left-hand window, minimize the Month folder and expand the Quarter folder.
11. Click the items called Fiscal Quarter and Qtr End Date and drag these across the window separator into the new folder.
12. In the left-hand window, minimize the Quarter folder and expand the Year folder.
13. Click the items called Fiscal Year and Year End Date and drag these across the window separator into the new folder.

**NOTE**

Existing lists of values are maintained and remain attached to the items as they are dragged into the new folder.

14. Your new folder should look like this:

**NOTE**

As you become more comfortable with highlighting and dragging items between folders, you may well find it easier to highlight the items in all of the simple folders first and then drag them all over at the same time.

After creating our base complex folder, we need to make it ready for use. If the items in the new folder do not have meaningful names, these should be renamed. To help our users, we will rename the item called Calendar Item to Cancel Date.

Having created a new folder, we need to join it to the Sales folder. The new folder that we created is for the Cancel Date, so it makes sense to create a join from here to the Cancel Date item in the Sales folder. This is shown here.
To finish the remaining date folders, we need to create three more complex folders, one each for the Order Date, Requested Date, and Ship Date. This process is much easier than creating the Cancel Date folder because we can take advantage of what we have just done by copying the new Cancel Date folder and pasting a replica of its definition into the business area three times.

**Replicating a Folder**  To replicate a complex folder, use the following workflow:

1. Click the name for the folder you want to copy so that the name is highlighted.
2. Right-click the folder, and from the pop-up menu, select Copy.
3. Right-click the heading for the business area, and from the pop-up menu, select Paste.
4. Give the new folder a meaningful name and identifier.

**Completing the Date Folders**  To complete our complex data folders, use the following workflow:

1. Use the workflow in the previous section to copy the Cancel Date folder three times.
2. After you have made these three copies, rename the new folders to **Order Date**, **Requested Date**, and **Ship Date**, respectively.
3. Make sure you assign a unique identifier to each new folder.
4. Rename the Cancel Date items in these folders to be **Order Date**, **Requested Date**, and **Ship Date**, respectively.
5. Add one join from each new folder to the Sales folder. As you do this, make sure you join on the correct items, with the Order Date folder joining to the Orderdate item, the Request Date joining to the Reqdate item, and the Ship Date joining to the Shipdate item.
6. Expanding the Sales folder, it now looks like this:

As you can see in the illustration, there are now seven joins into the Sales folder.

**Creating a New Complex Geographic folder**  We will create one new folder for the geographic rollup, containing all of the customer data, along with the city, district, and region data.

In our Sales fact folder there is a customer ID, which has a N:1 (many-to-one) join to the customer folder. The customer folder, in turn, has a N:1 join to the city, which in turn joins to district, which joins to region.

To create the Geographic rollup, use the following workflow:

1. Create a new complex folder.
2. Name both the new folder and its identifier **Geography**.
3. Drag the Cust ID from our Customer folder into the new folder.
4. Drag the remaining customer items.
5. Drag the City Name from the City folder.
6. Drag the District Name from the District folder.
Chapter 18: Editing the Business Area  681

7. Drag the Region Name from the Region folder.
8. Create a new join from the new folder to the Sales folder, joining on the Customer ID.
9. Hide the Customer ID in the Sales folder.
10. Hide the four original folders.

Creating a New Complex Products Folder  We will create one new folder for the product rollup, containing all of the product and product line data. In our Sales folder there is a product ID, which has a N:1 (many-to-one) join to the product folder. The product folder, in turn, has a N:1 join to the product line.

To create the Geographic rollup, use the following workflow:

1. Create a new complex folder.
2. Name both the new folder and its identifier Product.
3. Drag the Prod ID from our Products folder into the new folder.
4. Drag the remaining product items.
5. Drag the Line Name from the Prod Line folder.
6. Create a new join from the new folder to the Sales folder, joining on the Product ID.
7. Hide the Product ID in the Sales folder.
8. Hide the two original folders.

Notes about Complex Folders  Whenever you use complex folders, note these points:

- Lists of values are copied, and the item classes are shared with the original items.
- Joins are not copied, they must be created manually.
- Complex folders make great candidates for creating hierarchies.
- Hierarchies make great candidates for cascading parameters. A prerequisite of being able to create cascading parameters is that the items must belong to a hierarchy.

Adding Custom Folders  A custom folder is based on SQL that you key directly into Discoverer. While there are many reasons for using custom folders, there are three that we would like to discuss here:

- To create a new list of values
- To create an alternative sort sequence for a data set that does not have a natural sort
- To pull data from a database link

We will discuss each of these in turn, but first let us show you how easy it is to create a custom folder.

To create a custom folder, use the following workflow:

1. Single-click the business area heading such that the business area name is highlighted.
2. Right-click the business area name; Discoverer will pop up a context menu.
3. From the context pop-up menu, select New Custom Folder. Discoverer opens the following Custom Folder dialog box:

![Custom Folder dialog box](image)

4. Key or paste the SQL code into the box.
5. Give the folder a unique Name.
6. Click the Validate SQL button to check if the SQL is okay for Discoverer’s use.
7. If the SQL is invalid, you will see the following message:

![Oracle Discoverer Administrator dialog box](image)

**NOTE**
The actual ORA error message will vary depending upon what is wrong with your code. In this situation, the ORA 00936 means that we have forgotten to include a required element. We actually forgot to SELECT anything from our SQL!

8. Please note that not all SQL that is valid in SQL Plus is valid in Discoverer. We will explain this in the next section. You may have to keep plugging away until Discoverer says it is valid.
9. If the SQL is valid, you will see the following message:
10. Whether you have entered valid or invalid SQL, so long as you enter a unique name for the folder, Discoverer will allow you to click OK and save the folder.

Copyright © McGraw-Hill Companies 2006
This is a sample chapter for review purposes only.
Chapter 18: Editing the Business Area 683

NOTE
Discoverer will allow you to create a custom folder with invalid SQL. This is done for development purposes, allowing you to save what you have done so far and come back to it later when you have figured out what the problem is. This also allows you to create some SQL and then ask a colleague to take a look at it to see if he or she can figure out why it is not working.

If you have a custom folder that has invalid code, perhaps you should save what you have done so far and come back to it later. When the code is invalid and you click OK, Discoverer displays the following message:

If you click Yes, Discoverer will save the folder. Don’t worry, your users will not be able to select from this folder while it is in an invalid state.

If you have a custom folder that is currently valid and you add some additional code that makes it invalid, you will see this message:

Getting the SQL Valid in a Custom Folder  One of the most common complaints that we see from inexperienced administrators is that Discoverer says that their SQL is invalid. As mentioned in the preceding section, not all SQL that is valid in SQL Plus will compile inside Discoverer. You have to remember that Discoverer needs to have a unique name for every item in the SELECT statement.

Look at the following code, which is valid in SQL Plus but not valid in Discoverer. Because Discoverer strips off table prefixes from custom code, it will not be able to differentiate between the two HEADER_ID columns.

```
SELECT
SOH.HEADER_ID,
SOL.HEADER_ID
FROM
SO_HEADERS_ALL SOH, SO_LINES_ALL SOL
WHERE
SOL.HEADER_ID = SOH.HEADER_ID
```
Changing the code as follows is valid:

```
SELECT
SOH.HEADER_ID SOH_HEADER_ID,
SOL.HEADER_ID SOL_LINE_ID
FROM
SO_HEADERS_ALL SOH, SO_LINES_ALL SOL
WHERE
SOL.HEADER_ID = SOH.HEADER_ID
```

Using a Custom Folder to Create a New List of Values  We mentioned earlier in this chapter that you should not be creating lists of values from tables that have millions of rows of data. This is because of the performance hit on the database by Discoverer executing a SELECT DISTINCT from a million-row table. To overcome this, the solution is to use a custom folder.

To use a custom folder to create a list of values, use the following workflow:

1. Determine what the list of values should be. You can do this either by selecting from a separate dimension folder or by using DUAL.
2. Create a custom folder using the SQL that you have developed.
3. Create a new list of values based on the item in the custom folder.
4. Attach the list of values to the original item by right-clicking the item and, from the pop-up menu, selecting Properties.
5. In the Item Properties box look for the property called Item Class. When you click in here, you will be presented with a list of the valid item classes that Discoverer thinks could be suitable.
6. Select the new item class from the list presented.

**NOTE**

*Discoverer does some filtering on the item classes to not show you the lists of values that are not suitable. For example, if the column you are working on is defined as being a Number, Discoverer will offer you only numeric lists of values. It won’t offer you anything based on a VARCHAR or DATE, because it knows these would not work!*

7. Click OK.

From here on, every time a user clicks the plus icon alongside the item in the original folder, Discoverer will execute the code contained in the custom folder to generate the list of values. The following two examples show this to good effect.

**Custom Folder Based on a Dimension Table**

Let’s say you have an Oracle Applications system and you need to get a list of values based on the currency contained in your PO Lines, but you have millions of purchase orders and it takes too long to get the list by querying the PO Lines. The following code will give you a list of just the active currency codes contained in the system:

```
SELECT CURR_CODE
FROM APPS.FND_CURRENCIES_ACTIVE_V
```
If you place this code inside a custom folder, you will see a single item called CURR_CODE.
Creating a list of values on this item will produce the list in seconds.

**Custom Folder Using DUAL**

Let's say you have a system that has ten million rows, and you have a column called Active that has only two values, a Y to indicate Yes and N to indicate No. The following code will give you a list containing just two values, Y and N:

```sql
SELECT 'Y' ACTIVE_STATUS FROM DUAL
UNION
SELECT 'N' ACTIVE_STATUS FROM DUAL
```

Placing this code within a custom folder will cause Discoverer to generate a new folder containing a single item called ACTIVE_STATUS. Creating a list of values on this item will produce the list in a fraction of a second.

**Using a Custom Folder to Create an Alternative Sort Sequence**

Earlier in this chapter we mentioned that we would show you how to use a custom folder to generate the days of the week list of values. The following code will give you a new folder containing two items, one called DOY and one called SEQUENCE:

```sql
SELECT 'SUNDAY' DOY, 1 SEQUENCE FROM DUAL
UNION
SELECT 'MONDAY' DOY, 2 SEQUENCE FROM DUAL
UNION
SELECT 'TUESDAY' DOY, 3 SEQUENCE FROM DUAL
UNION
SELECT 'WEDNESDAY' DOY, 4 SEQUENCE FROM DUAL
UNION
SELECT 'THURSDAY' DOY, 5 SEQUENCE FROM DUAL
UNION
SELECT 'FRIDAY' DOY, 6 SEQUENCE FROM DUAL
UNION
SELECT 'SATURDAY' DOY, 7 SEQUENCE FROM DUAL
```

If you create a new list of values based on the DOY, using the SEQUENCE as the alternative sort item, the list will be generated in the order you want.

**Using a Custom Folder to Pull Data from a Database Link**

Sometimes when working with data, you may find that the piece of data you need is located in another database. Once again, let's say we are working in an Oracle Applications environment and that you have a data warehouse that has pulled in the SALESREP_ID but somehow not the SALESREP_NAME, and your users want to see the name. You ask your Oracle Warehouse Builder team if they can correct the code, and they say they can but not for another six months. Rather than telling your users that the request will have to wait, you use the following code to pull both the SALESREP_ID and SALESREP_NAME:

```sql
SELECT DISTINCT
RSA.RA_SALESREP_ID SALESREP_ID,
RSA.NAME SALESREP_NAME
FROM
APPS.RA_SALESREPS@DW_TO_PROD RSA
```
Adding New Folders from the Database
You can add new folders from the database very easily. In fact, this is what we did to create the business area in the first place.

To create a new folder from the database, use the following workflow:

1. Single-click the business area heading so the business area name is highlighted.
2. Right-click the business area name; Discoverer will open a pop-up menu.
3. From the context pop-up menu, select New Folder from Database.

Discoverer reopens the Load Wizard that we used at the beginning of Chapter 17.

Managing Folder Properties
Like all other objects in Discoverer, folders have properties. These properties are managed from the Folder Properties dialog box.

To display the Folder Properties dialog box, use the following workflow:

1. Right-click the folder whose properties you want to maintain, and from the pop-up menu select Properties.
2. Discoverer displays the Folder Properties dialog box, as shown in Figures 18-4 and 18-5.
3. Having changed any property, click OK to commit the change.

As you can see in Figures 18-4 and 18-5, there are four tabs to the Folder Properties dialog box:

- **General** On this tab, shown in Figure 18-4 for a complex folder and in Figure 18-5 for a folder based on a database object, you will find the following general properties:
  - **Name** This is the name of the folder and must be unique in the EUL.
  - **Description** This is an optional property and one that is greatly underused. When an end user clicks a folder that has a Description, Desktop or Plus displays the text that is contained in it. You should therefore make good use of this property.
  - **Visible to User** This is a Yes/No property. When set to No, the folder will not be visible to the end user. Be careful making folders invisible after the fact because this can upset users and even prevent existing workbooks from opening.
  - **Database** If the folder is based on a database object, Discoverer displays the name of the database where the object is located.
  - **Owner** If the folder is based on a database object, Discoverer displays the name of the owner of the object here.
  - **Object** If the folder is based on a database object, Discoverer displays the name of the object here.
  - **Optimizer Hints** If you are used to working with optimizer hints and you find that end-user performance is bad, you might want to try adding optimizer hints here. In the authors’ experience, if you find that hints are necessary, you may well need to create a database view instead and base the folder on that view.
  - **Identifier** This is the unique identifier for the folder in the EUL.

Copyright © McGraw-Hill Companies 2006
This is a sample chapter for review purposes only.
Chapter 18: Editing the Business Area

**FIGURE 18-4** Folder Properties dialog box for a complex folder

- **Dependents** On this tab Discoverer will display a list of all the secondary objects that are dependent upon this folder. Examples of secondary objects are:
  - Item classes (lists of values)
  - Joins
  - Workbooks

**FIGURE 18-5** Folder Properties dialog box for a folder based on a database object

Copyright © McGraw-Hill Companies 2006

This is a sample chapter for review purposes only.
The following illustration shows the Dependents tab for our Sales folder. We recommend that you always check for workbook dependencies whenever you are about to make a major change to a folder, such as to remove a join or disable reach-through to a base folder.

NOTE
If you do not check for dependent workbooks, and you proceed with your change, the workbooks will no longer open and it will be your fault. We can almost guarantee that when the user complains that a workbook no longer opens, you will not connect the two events and will be left scratching your head.

- **Components**  On this tab, which is displayed only for a complex folder, Discoverer displays a list of the objects that are being used to create the complex folder. These objects include:
  - Folders
  - Items
  - Joins

Here, you can see the Components tab for our Cancel Date folder:

- **Reach-Through**  On this tab, which is also displayed only for a complex folder, Discoverer allows you to define the objects that can be reached through this folder. We will describe reach-through in detail in the next section.

**Using Reach-Through from a Complex Folder**  One of the really clever features of Discoverer is reach-through. This feature allows...
Chapter 18: Editing the Business Area

Desktop and Plus users to add items to their worksheets from the base folders that were used to create the complex folders. A base folder can be any of the following:

- A simple folder based on a database object
- A custom folder based on SQL
- Another complex folder

If reach-through is enabled, when a user selects from one of the complex folders, all of the enabled reach-through folders from which the complex folder was created become available for selection.

You may say that you could already achieve this by adding joins from the complex folder back to its base folders. This is true, except it is not very elegant and not very efficient. For a start, you have all those extra joins to maintain, and it can also affect performance. The best way is to enable reach-through.

To enable reach-through from a complex folder back to one or more of its constituent base folders, use the following workflow:

1. Right-click the complex folder that you want reach-through to enable, and from the pop-up menu, select Properties. Discoverer displays the Folder Properties dialog box.
2. Click the Reach-Through tab.
3. Click the check box against the base folder(s) that you want to allow the users to reach.
4. Click OK to complete the reach-through.

The illustration shows the Reach-Through tab for our Cancel Date folder with reach-through enabled back to the original Month and Quarter folders.

Canceling Reach-Through While canceling reach-through is a very simple task, you need to be very careful. If you cancel reach-through to a folder that has been used in a user’s workbook, the user will no longer be able to open that workbook.

To cancel reach-through, use the following workflow:

1. Right-click the complex folder that is currently reach-through enabled, and from the pop-up menu, select Properties. Discoverer displays the Folder Properties dialog box.
2. Click the Reach-Through tab.
3. Uncheck the box against the base folder that you no longer want to allow users to reach.
4. Discoverer understands that disabling reach-through can have disastrous consequences for your users and displays the Remove Reach-Through dialog box.

Copyright © McGraw-Hill Companies 2006
This is a sample chapter for review purposes only.
5. When prompted “Are you sure you wish to remove reach-through,” clicking Yes confirms your understanding of the consequences, while clicking No abandons the reach-through removal. If you continue with the removal and there was a workbook using the reach-through-enabled folder, the user will get this message in Desktop:

![](image1)

The user will get this message in Plus:

![](image2)

To rectify this situation, all you have to do is re-enable the reach-through, if you can remember what you did. Perhaps it is better that you check dependencies first!

Even if you clicked Yes in the previous step, it is still not too late to abandon the removal. Clicking Cancel will abort what you have started.

If you are sure that you want to remove the reach-through, click OK.

Reach-Through Restrictions  As an administrator, there are several restrictions that you need to be aware of when working with the reach-through capability of Discoverer:

- The Discoverer Administrator tool will not allow you to enable reach-through if there is a join in existence between the base folder and the complex folder. Discoverer will generate the following error message:

![](image3)

- The Discoverer Administrator tool will not allow you to create a join between a base folder and its associated complex folder if reach-through has been enabled. Discoverer will generate the following error message:

![](image4)

- When you have a reach-through-enabled folder, and you are using Discoverer’s automatic summary management (ASM) feature to create summary folders, Discoverer will not generate summary folder information on the folders that are reach-through enabled.
Chapter 18: Editing the Business Area 691

- Items that are in folders that are joined to the reach-through-enabled folder are not available for selection by the end user. The reach-through works only to the folder that is reach-through enabled using the implied join.

- There must be at least one item from the base folder in the complex folder for reach-through to work. In other words, if you edit a complex folder and remove all the items from a reach-through-enabled base folder, the implied join between the two folders will be removed. Any user workbook that was using the implied join to get data from the base folder will fail.

- If you remove a join between a complex folder and a base table in order to enable reach-through and a user’s workbook was using that join, the workbook will no longer open and the user will see the dreaded “Cannot join tables used in the workbook. Item dependency” error message. Therefore, before removing any join, please check the impact.

Hiding a Folder
After you have created complex folders, it is common practice to hide the non-reach-through-enabled base folders upon which the complex folder has been created.

To hide a folder, use this workflow:

1. Right-click the folder you want to hide.
2. From the pop-up menu select Properties.
3. Change Visible to user from Yes to No.
4. Click OK.

Hierarchy Enhancements
A hierarchy allows a user to drill through the data. Typically we use 1:N relationships to define hierarchies in a business area so that users can roll up data or drill down. In a database, there are two types of drill:

- **Roll Up**  This reduces a query by allowing a user to include data from a higher level of the hierarchy, essentially summarizing the data. For example, you could roll up from a Month to a Quarter to see the whole quarter’s revenue.

- **Drill Down**  This expands a query by allowing a user to include data from a lower level, essentially bringing in more detail items to the query. For example, you could drill down from a Quarter to a Month to see the revenue for all of the months in the quarter.

In both types, Discoverer may requery the database. As well as the two database hierarchy types, there are two kinds of Discoverer hierarchy:

- **Item hierarchies**  According to Oracle, these are hierarchies on anything that is not a date; however, we will show you how to use these for dates too.

- **Date hierarchies**  These work only on dates. When you opt to allow Discoverer to create a date hierarchy, you need to be aware of what Discoverer will do and how this can impact performance. We will cover this later in the chapter.
Date Hierarchy or Item Hierarchy?
We are frequently asked where we stand with regard to using date hierarchies. To be truthfully honest, we rarely use date hierarchies and never use them for anything other than the simplest of folders. Let us explain.

As a concept, Oracle's implementation of date hierarchies in Discoverer is clever, really clever. What happens is that Discoverer creates a hierarchy of date format masks, inserting new items into the folder, one for each time element. At run time, Discoverer applies those format masks, using the TO_CHAR function, to evaluate each element of the hierarchy. The problem here is that these functions have to be evaluated against every row of the data in order to determine whether a filter that is being applied against one of those elements is true or false.

During the early summer of 2005, we had cause to visit a client of ours who was having trouble with performance. They complained that a worksheet running against a single table was taking over ten minutes to run. The first thing we noticed is that they had defined a date hierarchy on every date in the transaction folder. There were ten dates in that folder, and every one was using five different time elements. This made us step back and think. Our curiosity aroused, we wondered how many rows of data there were in that table. When we looked at the database, we were amazed to find that the table in question had more than two million rows. To make things worse, the users had constructed workbooks that used all of the time elements for all of the dates. In effect, 50 TO_CHAR functions were being performed on every row of the data, and of course the users were filtering on non-indexed items.

The first thing we did was to drop all of the date hierarchies. We then created a new materialized view that had all of the time elements prebuilt and pre-indexed. Next we created individual date folders, as you have seen us do in our tutorial, for each of the important dates. Finally, we created new item hierarchies, one per date folder, to replace the old date hierarchies. The time filter was applied against an index in this folder, joining back to an index in the main folder. When we ran a workbook against this setup, it ran in less than one minute.

You may think this is an extreme case. Believe us, it is not. There are organizations out there with billions of rows of data in single tables. If you attempt to create a date hierarchy on these, you will be asking for trouble. Our recommendation is to use a materialized view for your date elements using an item hierarchy that simply links the elements together. No functions, no TO_CHAR, no hassle.

Date Hierarchies and Indexes   We recommend that you not create date hierarchies on dates that are indexed. This is an area of great concern to us and one that we have seen affect performance so many times.

According to Oracle,"If you apply a date hierarchy to a date item from an indexed table, a query that includes one of these date items will not use the indexes (which can reduce performance)." We will qualify this statement by saying that this will affect performance, big time. The index will not be used, and Oracle will undertake full table scans every time.

We will show you how to create both types of Discoverer hierarchy. In both cases we will use the Cancel Date for the hierarchy.

Creating an Item Hierarchy
To create an item hierarchy, use the following workflow:

1. Click the Hierarchies tab.
2. Right-click the business area in which you wish to create the hierarchy, and from the pop-up list, select New Hierarchy. This opens the Hierarchy Wizard as seen in Figure 18-6.
3. Check Item Hierarchy.
4. Click Next. This opens the second screen of the Hierarchy Wizard.

5. Locate and open the folder containing the items that will be in the hierarchy. As you can see in the preceding image, we have selected the Cancel Date, Fiscal Month, Fiscal Quarter, and Fiscal Year.
6. Select the lowest-level drill item and move or paste it in the right-hand side.
7. Select the next items, in order, until you have selected all of the items you need.

Copyright © McGraw-Hill Companies 2006
This is a sample chapter for review purposes only.
8. When all items are selected, click Next. This opens the third and last screen of the Hierarchy Wizard.

9. Give a meaningful name and optional description to the hierarchy.

10. Click Finish.

Creating a Date Hierarchy

To create a date hierarchy, use the following workflow:

1. Click the Hierarchies tab.

2. Right-click the business area in which you wish to create the hierarchy, and from the pop-up list, select New Hierarchy. This opens the Hierarchy Wizard as seen in Figure 18-6.

3. Check Date Hierarchy.

4. Click Next. This opens the second screen of the Hierarchy Wizard.
Chapter 18: Editing the Business Area

5. Select the hierarchy items that you want to include by expanding the group elements and pasting the required time element into the right-hand side.

6. If the time element that you need is not among the available elements in a group, click the New Format button. This opens the New Date Format dialog box. Select the Group that you want to define a new element for, and in the New Format box type the format mask for the new time element. When done, click OK. The new element will be available in the chosen group.

NOTE

In the preceding box, we have defined a new element called YYYY in the Year group. For a list of all the valid date format masks, please refer to Appendix B.

7. Select the time elements, in descending order of Group, until you have selected all of the elements that you need.

8. In the following example, you can see that we have chosen ‘YYYY’ from the Year group, “Q”Q from the Quarter group, “Mon-RR” from the Month group, and “DD-Mon-YYYY” from the Date group.

NOTE

The Promote, Demote, and Description boxes in the Hierarchy Wizard will be grayed out until you click one of the elements that you have chosen. Then you can provide a description for the element and manipulate the hierarchy by demoting and promoting elements.
9. When all elements have been selected, click Next. This opens the third screen of the Hierarchy Wizard.

10. Optionally select the date item that you want to associate with this hierarchy. As you can see in the preceding example, we have used the Cancel Date.

**NOTE**

_You do not need to select any date item in this step. If you leave the Selected box empty, you will be creating a new template. For more information on date templates, see the next section._

11. When you have finished with Step 3, click Next. This opens the fourth and last screen of the Hierarchy Wizard.
12. Give a meaningful name and optional description to the hierarchy.
13. Click Finish.
14. Discoverer creates and adds one new item per time element to the folder, as you can see in this example:

Date Hierarchy Templates
In the preceding section we commented that if you left the third screen of the Date Hierarchy Wizard empty, you would in effect be creating a new date hierarchy template. But what is a date hierarchy template? Put in its simplest form, according to Oracle, “A date hierarchy template enables you to define a date hierarchy that you can apply to date items. A date item uses information that specifies the date, month, year, and time. Discoverer uses this information to calculate, for example, quarter, week, and days of the week. A date hierarchy template automatically creates items based on a date item, for example to represent the year or month.”

It is far more efficient to reuse a date hierarchy than to have to redefine the hierarchy every time you want to create a new one. This is where the template comes in. By saving a set of date hierarchy elements as a date hierarchy template, you can reuse this any number of times.

Creating and Selecting Your Hierarchies
How many hierarchies do you need in your business area? The number will vary depending upon how many rollups or drills you need. However, in our case, we need to create one item hierarchy for each of the six new complex folders that we have created:

- Cancel Date
- Order Date
- Request Date
- Ship Date
- Geography
- Product

We have already shown you how to create the Cancel Date hierarchy using both Item and Date hierarchies. Because we are using a custom fiscal calendar that has been prebuilt and pre-indexed, we will use item hierarchies for all of our dates. Simply following the instructions given earlier for creating an item hierarchy, we created the three other time hierarchies on the Order Date, Request Date, and Ship Date folders.

Turning to the Geographic and Product hierarchies, these are simple to build. They are both item hierarchies.

Geographic Hierarchy
Our geographic hierarchy is based on the Geography folder. At the lowest level this folder joins on the Customer ID to our Sales folder. The customer is the lowest-level item in the hierarchy. Looking at the available items, we can see that the Customer Name is available to use in the
Geography folder. In turn, we need to pick something from the other base folders that make up our complex Geography folder. If you recall, we have four 1:N folders making up the complex folder:

- Customer
- City
- District
- Region

It makes sense, therefore, that because the tables that these base folders are based on have a primary/foreign key relationship, we select one City item, one District item, and one Region item. In our final hierarchy we chose the following:

- Customer Name
- City Name
- District Name
- Region Name

Any sort of drilling on these items will perform very fast.

Product Hierarchy
We will turn our attention to the product hierarchy that will be based on the Product complex folder. At the lowest level this folder joins on the Product ID to our Sales folder. The product is therefore the lowest-level item in this hierarchy. Looking at the available items, we can see that both the Product Name and the Product Size are available to us in the Product folder. However, the join between the base Products folder and the Sales folder is on the Product ID, which in turn is linked to the Product Name. Therefore, the Product Name becomes the lowest element in our hierarchy. We need to pick something from the other base folder that forms part of our complex Product folder. If you recall, we have two 1:N folders making up this complex folder:

- Products
- Product Line

It makes sense that, because the tables that these base folders are based on have a primary/foreign key relationship, we should select the Product Line. In our final hierarchy we have the following:

- Product Name
- Product Line

Item Management
Why do we create new items? We create new items in the End User Layer to enable our users to

- Make better sense of the data
Avoid mistakes
Have guaranteed working algorithms

For these reasons, we will always look to create new algorithms in the business area whenever possible. If you look back to the early part of this chapter where we created a new item to convert a string into a number for use in a complex join, you already know how to add a new item to a folder. We therefore will not dwell on this point, other than to say that you will be using the New Item dialog box.

In this section we will not belabor how to create new items but concentrate instead on the following:

- Types of items
- Useful algorithms
- Sorting items
- Assigning indexes to an item

**Types of Items**
Discoverer understands three types of items:

- **Columns** These items are based on columns from objects in the underlying database.
- **Components** These items are based on the components of a custom folder.
- **Calculations** These items are based on calculated items that you, the administrator, create on behalf of your users.

**Useful Algorithms**
A number of useful, standard algorithms can be created in Discoverer. We will describe three of these:

- **Profit** Calculated as the Selling Price minus the Cost Price
- **Standard margin** Calculated as profit divided by SUM of the selling price
- **Standard markup** Calculated as profit divided by SUM of the cost price

When creating calculated items using division, we must make sure that we allow for and ensure that we do not introduce divide by zero errors. This subject was covered in depth in Chapter 13, where we described the techniques that you must use to avoid these errors.

**Adding Items to a Complex Folder**
If you are creating new items in a complex folder, Discoverer lists all of the folders that were used to create the complex folder. Discoverer also displays all of the items in the original base folders, irrespective of whether they are being used in the complex folder. This is very useful to know, especially if you are adding calculated items to the complex folder.

This allows you to build new items on any item from the base folders, not just on the items in the complex folder itself.

Copyright © McGraw-Hill Companies 2006
This is a sample chapter for review purposes only.
Sorting Items
Should you wish to sort the items in a folder, Discoverer provides a means to do so. To sort the items in a folder, use the following workflow:

1. Right-click the folder that you wish to sort.
2. From the pop-up menu, select Sort Items.
3. Discoverer displays the following Alphabetical Sort dialog box:

![Alphabetical Sort Dialog Box]

4. As you can see in the preceding illustration, Discoverer tells you how many items are to be sorted and prompts “Do you want to continue?” Click Yes to sort, or No to cancel.

Descriptive Lists of Values: Assigning Indexes to an Item
Another new feature of Discoverer is the ability to add an indexed item to another item to improve parameter selection performance in the end-user tools. So long as both items exist in the same folder, and neither is hidden, you can tell Discoverer to search an index rather than the original item.

Let’s explore this some more by using the following example. Suppose you have a base table that has a Product Name and a Product ID, with the Product ID being indexed and the Product Name not indexed. Most users will know a product by its name, not by its ID value, and so will want to create parameters based on the name. However, because the name is not indexed, the query could take a long time. To overcome this poor performance, Oracle has introduced the concept of descriptive lists of values. The list of values, in the parameter selection screen, is shown along with its associated ID. However, when the query is executed, Discoverer uses the ID.

To set up a descriptive list of values, use the following workflow:

1. Open and expand a folder that contains both the item and the indexed item.
2. Right-click the item, and from the pop-up menu, select Properties.
3. Navigate to the property called Indexed Item. It is located toward the bottom of the list.

NOTE
In order for this to work, the item must have a list of values associated with it before the Indexed Item property is available. If you do not have a list of values associated with the item, the property will be grayed out.

4. Click the Indexed Item property and from the pop-up list, select the indexed item that you wish to use.
5. As you can see in the preceding illustration, Discoverer inserts a marker alongside any indexed item that it finds in the folder.

6. Select the indexed item that matches one-to-one with the original item and click OK.

7. As you can see next, Discoverer prompts “Setting an indexed item requires a new LOV to be created” and offers you OK or Cancel.

8. Click OK to complete the attachment of the index.

In Discoverer Plus, whenever the user creates a parameter on the customer name, he or she will be given the opportunity to search by name or by the index. Searching by the index will greatly improve performance.

Using Descriptive Lists of Values in Plus
Having set up the descriptive lists of values, perhaps you might be interested in seeing how an end user can take advantage of this in Discoverer Plus. For this example, we created a descriptive list of values on the Channel using the Channel ID as the indexed item.

We created a new workbook using the Channel and then started to create a new condition using a parameter. As you can see in the illustration that follows, Discoverer now allows the user to check the box named “Enable users to select either indexes or values.”
With the indexes or values box checked, the boxes immediately below are enabled. The first, as you can see in the preceding image, is a drop-down that allows the user to specify that an Index or a Value should be used as the default. As you can see, we have opted to use the index as the default and have selected “(2) INTERNET” from the drop-down.

When this parameter is presented to a user in a working query, Discoverer displays the revised Edit Parameter Values dialog box.

In the preceding dialog box, you can see that Discoverer is allowing the user to choose whether he or she wants to input the index value or the actual value. Many users actually know the index values or codes that are being used on the system and may prefer to key in the code value when selecting parameters. The preceding explanation shows you how to enable this.

Summary

In this chapter you learned how to continue with the remaining business area checks. You learned the value of a well-designed business area as it affects the end user. We demonstrated how to manage lists of values and joins. We showed you how to add new folders, followed by managing hierarchies. Finally we showed you some of the more advanced mechanisms for item management, culminating in an explanation of how to manage descriptive lists of values.