Perspective

In this chapter, you learned the skills for working with parameters and stored procedures. These are essential skills for working with commands as you develop database applications. In the next chapter, you'll learn an essential skill for working with a series of related commands.

Terms

parameter
parameterized query
input parameter
output parameter
stored procedure

Exercise 7-1 Enhance the Vendor Maintenance application

In this exercise, you'll enhance the Vendor Maintenance application that's presented in this chapter. That will give you a chance to work with parameters and stored procedures. It will also demonstrate that it's relatively easy to make changes to a three-layer application.

Review the application

1. Open the Vendor Maintenance application that's in your Chapter 7 directory. Then, use the Solution Explorer to review the files for this application.
2. In the PayablesData project, note that each business class is matched with a DB class. In the VendorMaintenace project, note that the Validator class provides the methods for validating data.
3. Test the application to see how it works. First, modify a row. Then, add a new row.

Add a delete function to the application

4. Add a Delete button to the Vendor Maintenance form to the right of the Modify button.

5. Add a DeleteVendor method to the VendorDB class that deletes the current vendor row. This method should have one parameter that accepts a Vendor object, and it should return a Boolean value of True if the deletion works and False if it doesn't.

   Like the UpdateVendor method in the VendorDB class, the deletion shouldn't be done if the vendor data has changed since it was retrieved from the database. Also, if a SQL exception occurs during the operation, the exception should be thrown to the calling procedure.

   One way to code this method is to copy and modify the code for the UpdateVendor method. Recall from figure 1-11 that the Delete statement doesn't require any values to be Set and note that only one SQL parameter is needed. So your completed DeleteVendor method will be much shorter than the UpdateVendor method.

6. Add a Click event handler for the Delete button. This event handler should start by using a message box to confirm that the user wants to delete the current vendor. If the user confirms, the event handler should execute the DeleteVendor method in the VendorDB class.

   Then, if the method returns True, the controls on the form should be cleared. If the method returns False, an error message should be displayed that indicates that another user updated or deleted the current vendor before the delete operation. And if the method throws a SQL exception, the exception message should be displayed.

7. Test and debug this enhancement until it works right.

Use a stored procedure for the GetVendor method in the VendorDB class

8. Use the View menu to display the Server Explorer (Database Explorer if you are using Visual Basic 2010 Express edition). Then, expand the nodes for the Payables database until you reach Stored Procedures. There, you can see that the database includes a stored procedure named spGetVendor, and if you double-click on it you can see the details of the stored procedure.

9. Modify the GetVendor method in the VendorDB class so it uses the spGetVendor stored procedure instead of the coded Select statement. To do that, comment out the statements that you won't need. Then, use figure 7-10 as a guide for writing the statements that you do need for using the stored procedure.

10. Test and debug this enhancement until it works right. Note that this enhancement doesn't require changes to any of the presentation or business classes.