Monitoring

In this second example, we have designed a hidden form with a *Timer* event that polls a table and advises users when the system will shut down for maintenance. If the time has elapsed, then the routine will close all the user's forms and reports and quit the application. An administrator can use the table depicted in Figure 6-30 to set a shutdown time and post a user message.

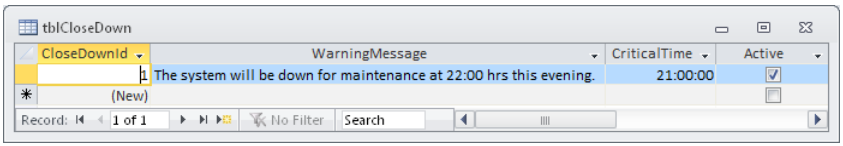

Figure 6-30 The Active flag indicates the event is scheduled for activation.

This application will have a hidden form that is launched upon startup and monitors the table for warning messages. Users are then be given a number of warnings, after which the application will close.

To implement this, you need the following code at startup:

```
Sub modTimer_LaunchMonitor()
    DoCmd.Echo False
    DoCmd.OpenForm "frmSilent"
    Forms("frmSilent").Visible = False
    DoCmd.Echo True
End Sub
```

The form shown in Figure 6-31 and the code that follows it incorporates a Dismiss button that allows the user to ignore the warning until they have completed their work (as long as they don't ignore it for too long).

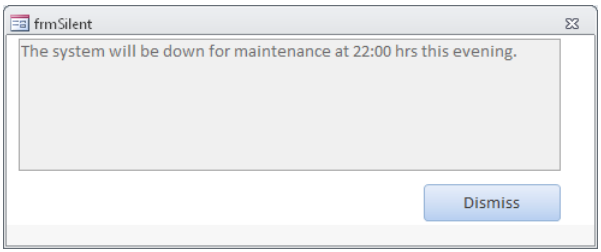

Figure 6-31 The user can press the Dismiss button to hide the window.

```
Option Compare Database
Option Explicit
Dim lngCriticalWarnings As Long ' flags critical warnings to a user
Const lngMaxWarnings = 2
Dim blUnloading As Boolean

Private Sub cmdDismiss_Click()
    ' hide the form
    Me.Visible = False
End Sub
Private Sub Form_Load()
    lngCriticalWarnings = 0
    blUnloading = False
End Sub

Private Sub Form_Timer()
    Dim db As Database
    Dim rst As Recordset
    Dim rstLog As Recordset
    Set db = CurrentDb
    Dim strSQL As String
    ' find any critical shutdown messages that apply
    strSQL = "SELECT *  FROM tblCloseDown " & _
            " WHERE CriticalTime > dateadd('n',-30,time())" & _
            " AND Active = True"

    Set rst = db.OpenRecordset(strSQL, dbOpenDynaset)
    If Not rst.EOF Then
        lngCriticalWarnings = lngCriticalWarnings + 1
        If lngCriticalWarnings > lngMaxWarnings Then
            ShutDown
        Else
            ' warn the user
            Me.Visible = True
            Me.txtWarning = rst!WarningMessage
            ' this stays on the screen until the user clicks dismiss
        End If
    End If
    rst.Close
End Sub

Sub ShutDown()
    ' shut down the application
    ' close any forms
    On Error Resume Next
    Dim lngfrmCount As Long
    Dim lngrptCount As Long
    Dim lngCount As Long
    lngfrmCount = Forms.Count
    lngrptCount = Reports.Count
    ' close all forms
```

```
    For lngCount = 0 To lngfrmCount - 1
        DoCmd.SelectObject acForm, Forms(lngCount).Name
        DoCmd.Close
    Next
    ' close all reports
    For lngCount = 0 To lngrptCount - 1
        DoCmd.SelectObject acReport, Reports(lngCount).Name
        DoCmd.Close
    Next
    blUnloading = True
    DoCmd.Quit
End Sub

Private Sub Form_Unload(Cancel As Integer)
    If Not blUnloading Then
        Me.Visible = False
        Cancel = True
    End If
End Sub
```

If you are using hidden forms, it's worthwhile considering what happens to these forms if a user attempts to leave the application. In the sample menu form *frmFormsMenu*, the following code is added to the *Close* event to tidy up the hidden form:

```
Private Sub Form_Close()
    ' close frmSilent if it is loaded
    If CurrentProject.AllForms("frmSilent").IsLoaded Then
        Forms("frmSilent").ShutDown
    End If
End Sub
```

The *Mouse* Events

Mouse events occur for the form, the different form sections, and the controls on the form that can be used to control how a user interacts with the data in the form. With these events, you can determine the mouse position, whether the Shift key is being held down, and which mouse button is pressed. There is also a *MouseWheel* event on the form.

The *frmMouseEvents* form, which can be found in the companion content that accompanies this chapter, can be used to investigate these events and display the corresponding values in screen controls for the mouse position and state of the buttons, as shown in Figure 6-32.