Making the Most of the PDMWorks API

By

Jerry Winters

SolidWorks World 2007

Jerry Winters
jerryw@solidworksvba.com
(801) 508-0106

Introduction

We have 90 minutes to make the most of the SolidWorks API. Obviously, to make the most of anything, we want to exploit as much of it's capabilities as possible. Here's a bulleted list of the items we are going to work with in this class:

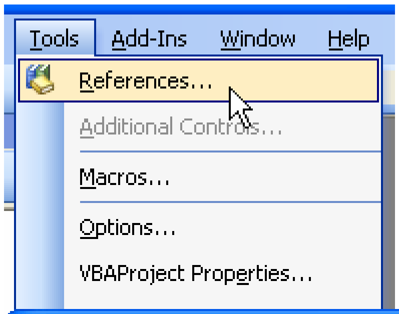
- Attach to PDMWorks
- Check In a Document
- SaveAs a Document from the Vault
- Query the PDMWorks Vault
- Export queried data to Excel
- Export queried data to Access
- Export queried data to SQL Server
- Export queried data to Oracle
- Set up Triggers
- Capture Trigger Information
- Store Trigger Information in a Database

If we have any time after doing the above, we will pull a few tricks out just for those in attendance.

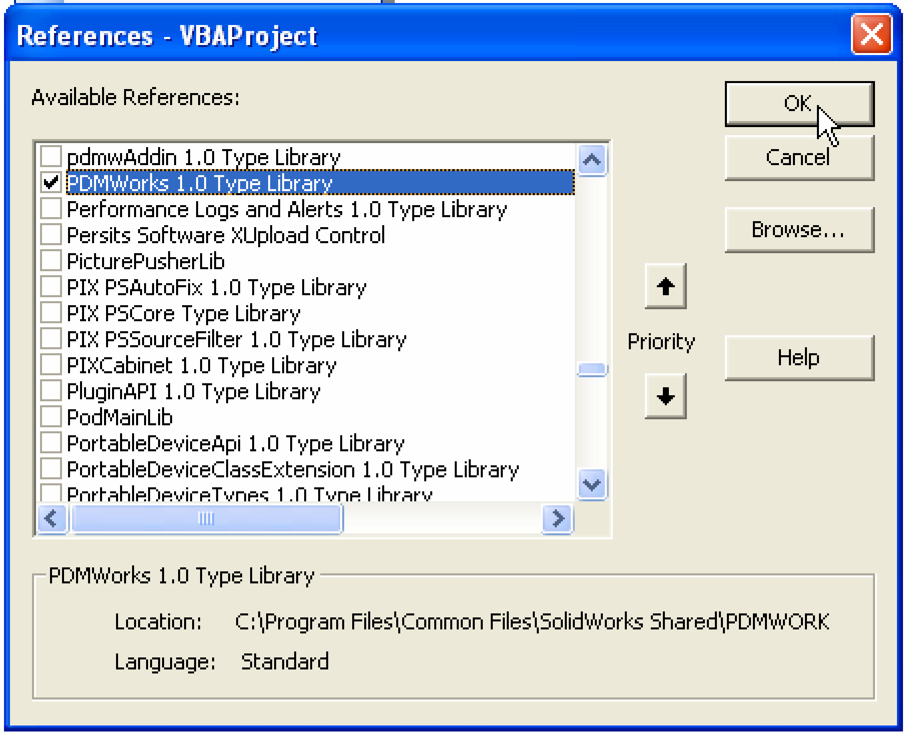
Attach to PDMWorks

How difficult is it to Attach to PDMWorks? We are going to begin by 'attaching' to PDMWorks in Microsoft Excel.

Attaching in VBA



The first thing we need to do to attach to PDMWorks is add a Reference to the "PDMWorks 1.0 Type Library".



Available References:

- [ ] pdmwAddin 1.0 Type Library
- [✓] PDMWorks 1.0 Type Library
- [ ] Performance Logs and Alerts 1.0 Type Library
- [ ] Persits Software XUpload Control
- [ ] PicturePusherLib
- [ ] PIX PSAutoFix 1.0 Type Library
- [ ] PIX PSCore Type Library
- [ ] PIX PSSourceFilter 1.0 Type Library
- [ ] PIXCabinet 1.0 Type Library
- [ ] PluginAPI 1.0 Type Library
- [ ] PodMainLib
- [ ] PortableDeviceApi 1.0 Type Library
- [ ] PortableDeviceClassExtension 1.0 Type Library
- [ ] PortableDeviceTypes 1.0 Type Library

PDMWorks 1.0 Type Library

Location: C:\Program Files\Common Files\SolidWorks Shared\PDMWORK

Language: Standard

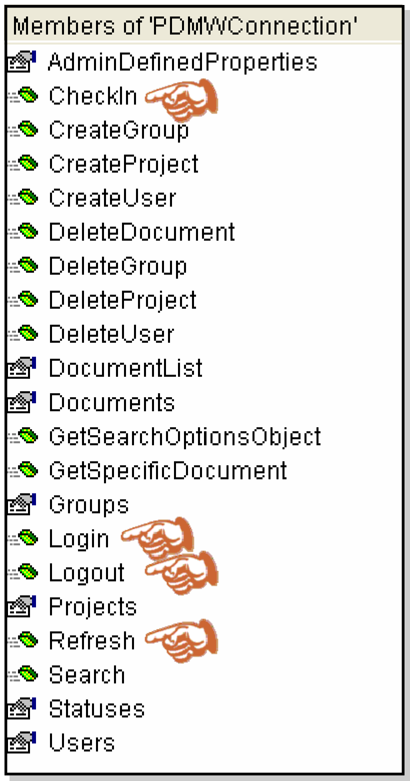
Once we select the correct Reference, we can click the OK button.

```
1.  Sub AttachToPDMWorks()
2.     Dim myPDMConn As New PDMWConnection
3.     myPDMConn.Login "pdmwadmin", "pdmwadmin", "localhost"
4.     myPDMConn.Refresh
5.     myPDMConn.Logout
6.  End Sub
```

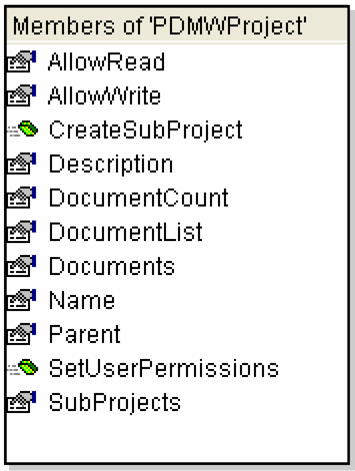
The code is fairly straight forward.

1. The Procedure is named "AttachToPDMWorks".
2. Declare a variable as a New PDMWConnection.
3. Login to the Connection with a Username, Password, and Server Name.
4. Refresh the PDMWorks Vault.
5. Logout of the PDMWorks Vault.
6. End of Procedure.

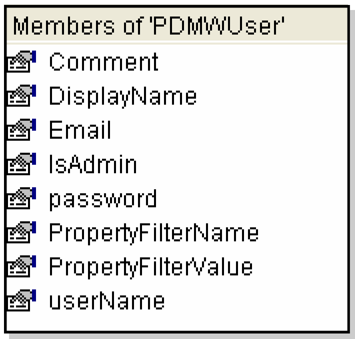
Once we Login to a PDMWorks Vault, we can work with the Vault.



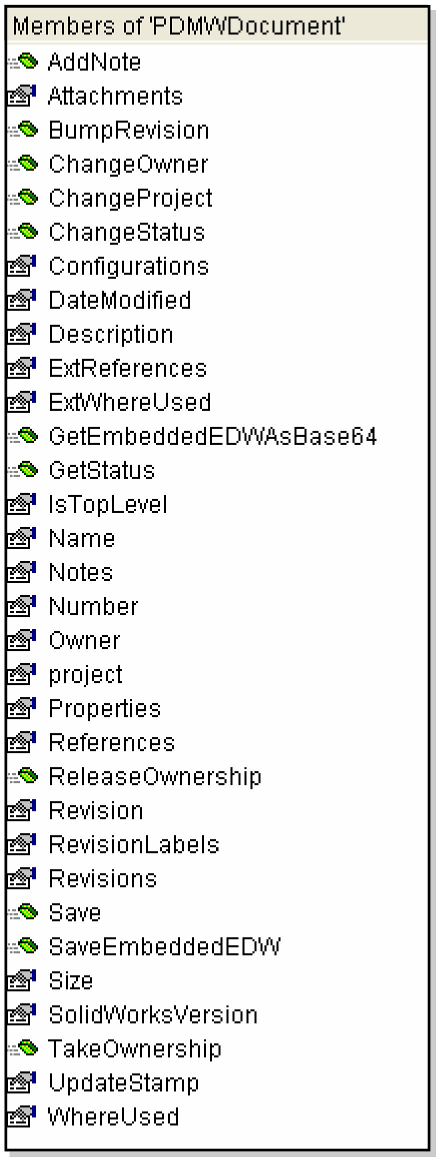
Members of 'PDMWConnection'
- AdminDefinedProperties
- CheckIn
- CreateGroup
- CreateProject
- CreateUser
- DeleteDocument
- DeleteGroup
- DeleteProject
- DeleteUser
- DocumentList
- Documents
- GetSearchOptionsObject
- GetSpecificDocument
- Groups
- Login
- Logout
- Projects
- Refresh
- Search
- Statuses
- Users



Members of 'PDMWProject'
- AllowRead
- AllowWrite
- CreateSubProject
- Description
- DocumentCount
- DocumentList
- Documents
- Name
- Parent
- SetUserPermissions
- SubProjects



Members of 'PDMWUser'
- Comment
- DisplayName
- Email
- IsAdmin
- password
- PropertyFilterName
- PropertyFilterValue
- userName



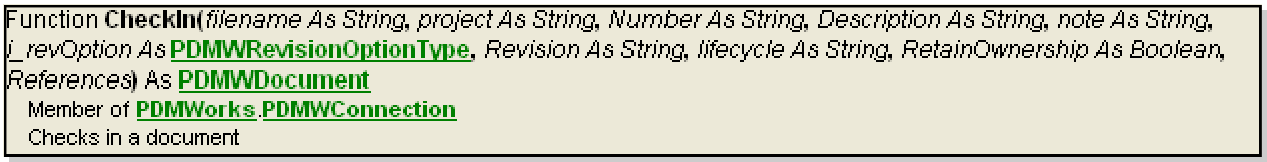
Members of 'PDMWDocument'
- AddNote
- Attachments
- BumpRevision
- ChangeOwner
- ChangeProject
- ChangeStatus
- Configurations
- DateModified
- Description
- ExtReferences
- ExtWhereUsed
- GetEmbeddedEDWAsBase64
- GetStatus
- IsTopLevel
- Name
- Notes
- Number
- Owner
- project
- Properties
- References
- ReleaseOwnership
- Revision
- RevisionLabels
- Revisions
- Save
- SaveEmbeddedEDW
- Size
- SolidWorksVersion
- TakeOwnership
- UpdateStamp
- WhereUsed

Here we see the Members of several important Objects in the PDMWorks Object Library. We have already used Login, Refresh, and Logout. Let's see if we can perform a CheckIn.

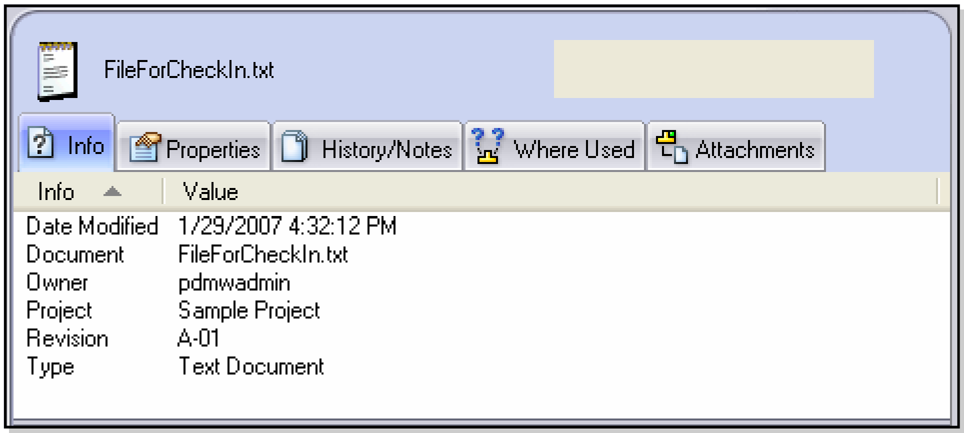
CheckIn a Document

How difficult is it to Check a Document into PDMWorks with the API?

```
Sub CheckInPDMWorks()
   Dim myPDMConn As New PDMWConnection
   myPDMConn.Login "pdmwadmin", "pdmwadmin", "localhost"
   myPDMConn.CheckIn "C:\data\SWWorld 2007\PDMWorks\FileForCheckIn.txt", _
             "sample", _
             "10001", _
             "Simple TXT File", _
             "Checking in via the API", _
             PDMWRevisionOptionType.Default, _
             "A-01", _
             "WIP", _
             True, _
             Nothing
   myPDMConn.Refresh
   myPDMConn.Logout
End Sub
```



Function **CheckIn**(*filename As String*, *project As String*, *Number As String*, *Description As String*, *note As String*, *i_revOption As* PDMWRevisionOptionType, *Revision As String*, *lifecycle As String*, *RetainOwnership As Boolean*, *References*) As PDMWDocument
Member of PDMWorks.PDMWConnection
Checks in a document

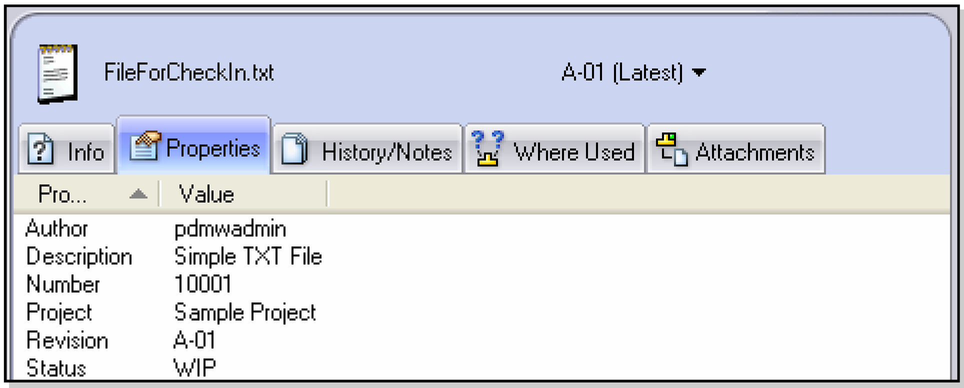


FileForCheckIn.txt

Info	Value
Date Modified	1/29/2007 4:32:12 PM
Document	FileForCheckIn.txt
Owner	pdmwadmin
Project	Sample Project
Revision	A-01
Type	Text Document

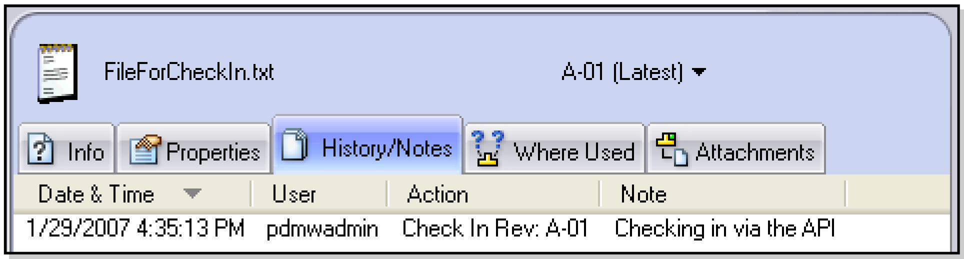
When we look in SolidWorks Explorer at the document we just checked in, we can see how the parameters we supplied are used and where they go.

The only thing that may cause a few questions here is the Project. We asked PDMWorks to check the document into the "sample" project but it looks like it was checked into the "Sample Project" project. They are actually the same thing. The actual project is named "sample" and it's Description is "Sample Project".



FileForCheckIn.txt A-01 (Latest)

Pro...	Value
Author	pdmwadmin
Description	Simple TXT File
Number	10001
Project	Sample Project
Revision	A-01
Status	WIP



FileForCheckIn.txt A-01 (Latest)

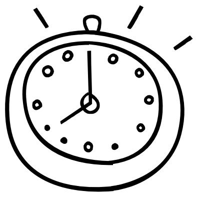
Date & Time	User	Action	Note
1/29/2007 4:35:13 PM	pdmwadmin	Check In Rev: A-01	Checking in via the API

SaveAs a Document from the Vault

Attaching to PDMWorks is easy. Checking a document into the Vault is easy. Saving a file in the vault out to the system should be easy too. Right?

```
1.  Sub SaveAsADocument()
2.      Dim myPDMConn As New PDMWConnection
3.      myPDMConn.Login "pdmwadmin", "pdmwadmin", "localhost"
4.      myPDMConn.Refresh
5.      Dim myDoc As PDMWDocument
6.      Set myDoc = myPDMConn.GetSpecificDocument("FileForCheckIn.txt")
7.      myDoc.Save "C:\data\SWWorld 2007"
8.      myPDMConn.Logout
9.  End Sub
```

Lines 5, 6, and 7 are the lines of code that save the file to the local system. We are using the Save method of the PDMWDocument object (See page 3 for the members of the PDMWDocument object).



Review Time.

We have accomplished quite a bit so far. We know how to attach to PDMWorks, CheckIn a Document, and Save a Document in the Vault to our local system. And where have we done this? In Microsoft Excel's VBA environment. The same code will work from with Access, SolidWorks, or any other program that uses VBA. That's pretty cool. But what does it take to create a stand-alone program (.exe file) that does these things? Let's take a look.

Using VB.NET Express Edition

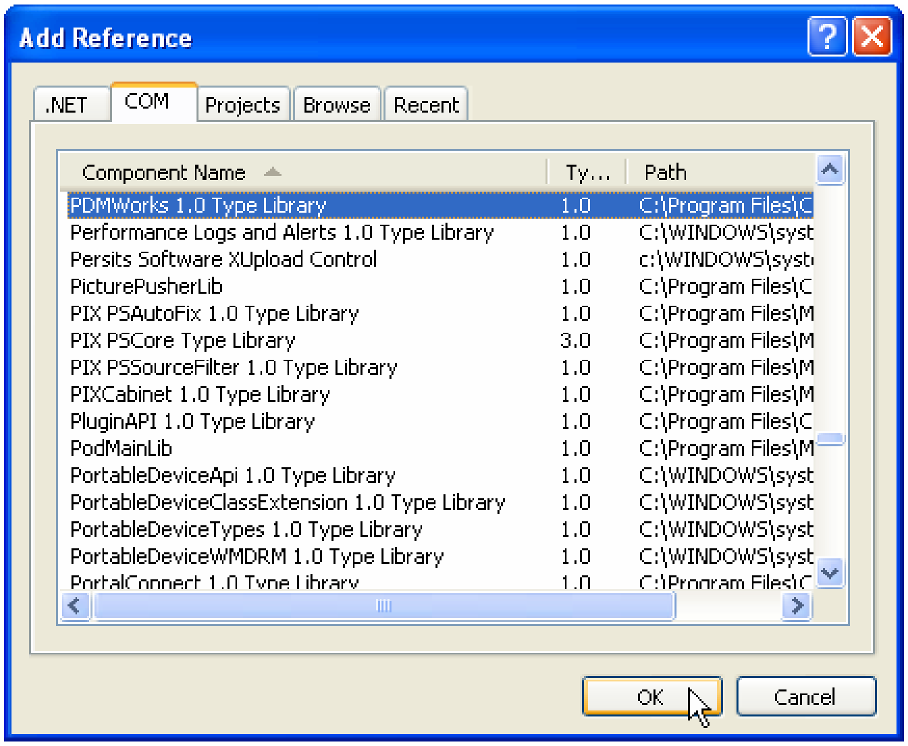
Microsoft has provided a VB.NET development environment for us FREE OF CHARGE. Yes, you read that correctly. It said "FREE OF CHARGE". It can be downloaded at:

http://msdn.microsoft.com/vstudio/express/vb/

Let's create a new VB.NET Express Project right now and see if we can get our code to work in this new environment.

After we begin a new Project, what's the first thing we need to do? Add a Reference.

Do a right-click on the Project, select Add Reference, and select "PDMWorks 1.0 Type Library" from the list.



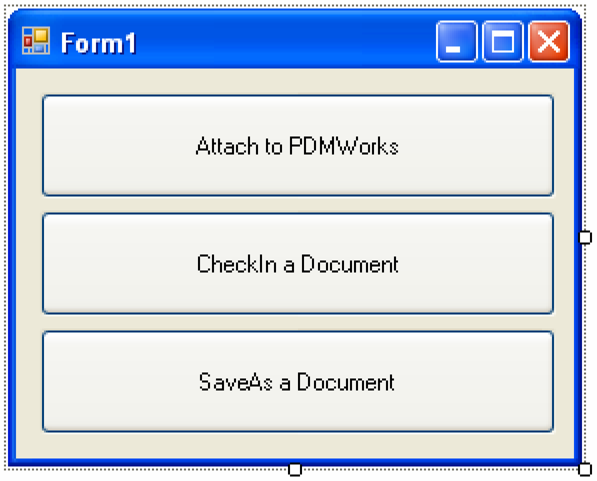
Component Name	Ty...	Path
PDMWorks 1.0 Type Library	1.0	C:\Program Files\C
Performance Logs and Alerts 1.0 Type Library	1.0	C:\WINDOWS\syst
Persits Software XUpload Control	1.0	c:\WINDOWS\syst
PicturePusherLib	1.0	C:\Program Files\C
PIX PSAutoFix 1.0 Type Library	1.0	C:\Program Files\M
PIX PSCore Type Library	3.0	C:\Program Files\M
PIX PSSourceFilter 1.0 Type Library	1.0	C:\Program Files\M
PIXCabinet 1.0 Type Library	1.0	C:\Program Files\M
PluginAPI 1.0 Type Library	1.0	C:\Program Files\C
PodMainLib	1.0	C:\Program Files\M
PortableDeviceApi 1.0 Type Library	1.0	C:\WINDOWS\syst
PortableDeviceClassExtension 1.0 Type Library	1.0	C:\WINDOWS\syst
PortableDeviceTypes 1.0 Type Library	1.0	C:\WINDOWS\syst
PortableDeviceWMDRM 1.0 Type Library	1.0	C:\WINDOWS\syst
PortalConnect 1.0 Type Library	1.0	C:\Program Files\C

Here's the code in it's entirety. Notice the "Imports" statement. This keeps us from typing "PDMWorks" in front of all PDMWorks calls.

```
1.  Imports PDMWorks
2.  Public Class Form1
3.     Private Sub Button1_Click(ByVal sender As System.Object, _
          ByVal e As System.EventArgs) Handles Button1.Click
4.        Dim myPDMConn As New PDMWConnection
5.        myPDMConn.Login("pdmwadmin", "pdmwadmin", "localhost")
6.        myPDMConn.Refresh()
7.        myPDMConn.Logout()
8.     End Sub


9.     Private Sub Button2_Click(ByVal sender As System.Object, _
          ByVal e As System.EventArgs) Handles Button2.Click
10.       Dim myPDMConn As New PDMWConnection
11.       myPDMConn.Login("pdmwadmin", "pdmwadmin", "localhost")
12.       myPDMConn.CheckIn("C:\data\SWWorld 2007\PDMWorks\FileForCheckIn.txt", _
             "sample", _
             "10001", _
             "Simple TXT File", _
             "Checking in via the API", _
             PDMWRevisionOptionType.Default, _
             "A-01", _
             "WIP", _
             True, _
             Nothing)
13.       myPDMConn.Refresh()
14.       myPDMConn.Logout()
15.    End Sub


16.    Private Sub Button3_Click(ByVal sender As System.Object, _
          ByVal e As System.EventArgs) Handles Button3.Click
17.       Dim myPDMConn As New PDMWConnection
18.       myPDMConn.Login("pdmwadmin", "pdmwadmin", "localhost")
19.       myPDMConn.Refresh()
20.       Dim myDoc As PDMWDocument
21.       myDoc = myPDMConn.GetSpecificDocument("FileForCheckIn.txt")
22.       myDoc.Save("C:\data\SWWorld 2007")
23.       myPDMConn.Logout()
24.    End Sub
25. End Class
```

A close look at the code will reveal very slight differences between VBA and VB.NET. The great thing is, though, that if the Imports statement (Line 1) is in place, we can copy and paste the code from VBA directly into VB.NET and VB.NET will take care of the formatting for us.

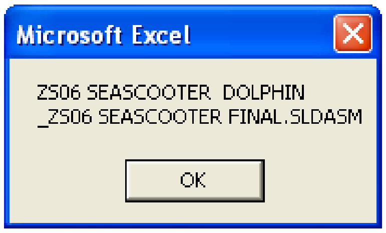
Now we have a program that can be compiled into an .exe file and distributed. We also have access to the .NET Framework.



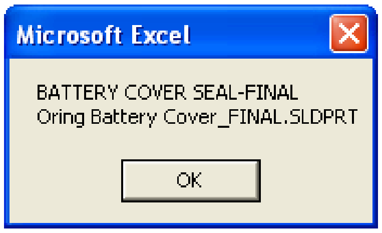
Query the PDMWorks Vault

We have seen an example of using the Save method of the PDMWDocument object. We retrieved this document by using the “GetSpecificDocument” method. In the previous example, we knew the filename. That was an advantage for us. But we won’t always know the filename. There are times when we need to ask the Vault to return a collection of documents so we can sort through them one by one.

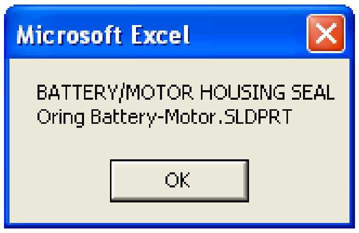
```
1.  Sub QueryTheVaultA()
2.     Dim myPDMConn As New PDMWConnection
3.     Dim myPDMWSO As PDMWSearchOptions
4.     Dim myPDMWSRs As PDMWSearchResults
5.     Dim myPDMWSR As PDMWSearchResult
6.     myPDMConn.Login "pdmwadmin", "pdmwadmin", "localhost"
7.     myPDMConn.Refresh
8.     Set myPDMWSO = myPDMConn.GetSearchOptionsObject
9.     myPDMWSO.SearchCriteria.AddCriteria pdmwAnd, _
           pdmwDescription, _
           "", _
           pdmwContains, _
           "sea"
10.    Set myPDMWSRs = myPDMConn.Search(myPDMWSO)
11.    For Each myPDMWSR In myPDMWSRs
12.        If Not myPDMWSR.Document Is Nothing Then
13.            MsgBox myPDMWSR.Description & vbCr & myPDMWSR.Name
14.        End If
15.    Next
16.    myPDMConn.Logout
17. End Sub
```



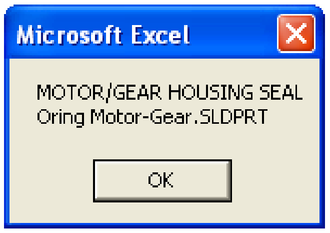
```
ZS06 SEASCOOTER  DOLPHIN
_ZS06 SEASCOOTER FINAL.SLDASM
```



```
BATTERY COVER SEAL-FINAL
Oring Battery Cover_FINAL.SLDPRT
```



```
BATTERY/MOTOR HOUSING SEAL
Oring Battery-Motor.SLDPRT
```

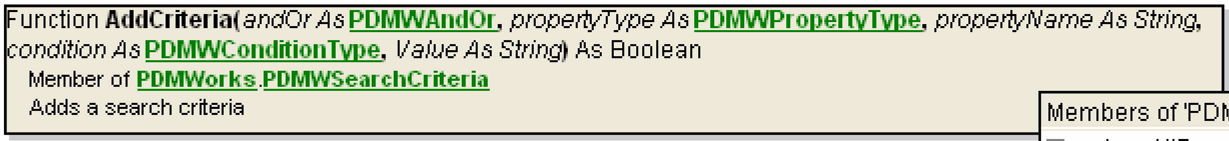


```
MOTOR/GEAR HOUSING SEAL
Oring Motor-Gear.SLDPRT
```

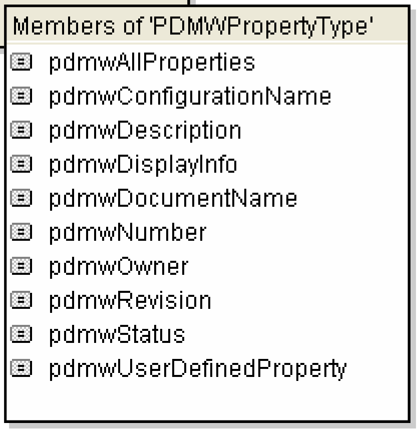
We can see the results of the code shown above. Let’s talk a little about the process of querying the PDMWorks Vault.

A. Obtain the SearchOptionsObject from the PDMWorks Connection. (See Line 8.)
B. Add criteria. Multiple Criteria can be added. This example shows only one. (See Line 9.)
C. Search the Vault using the SearchOptionsObject. (See Line 10.)
D. Traverse the PDMWSearchResults using a For Each statement. (See Lines 11 thru 15.)

In the above example, we are looking for the letters “sea” in the Description property.



Function **AddCriteria**(*andOr As* **PDMWAndOr**, *propertyType As* **PDMWPropertyType**, *propertyName As String*, *condition As* **PDMWConditionType**, *Value As String*) As Boolean
Member of **PDMWorks**.**PDMWSearchCriteria**
Adds a search criteria



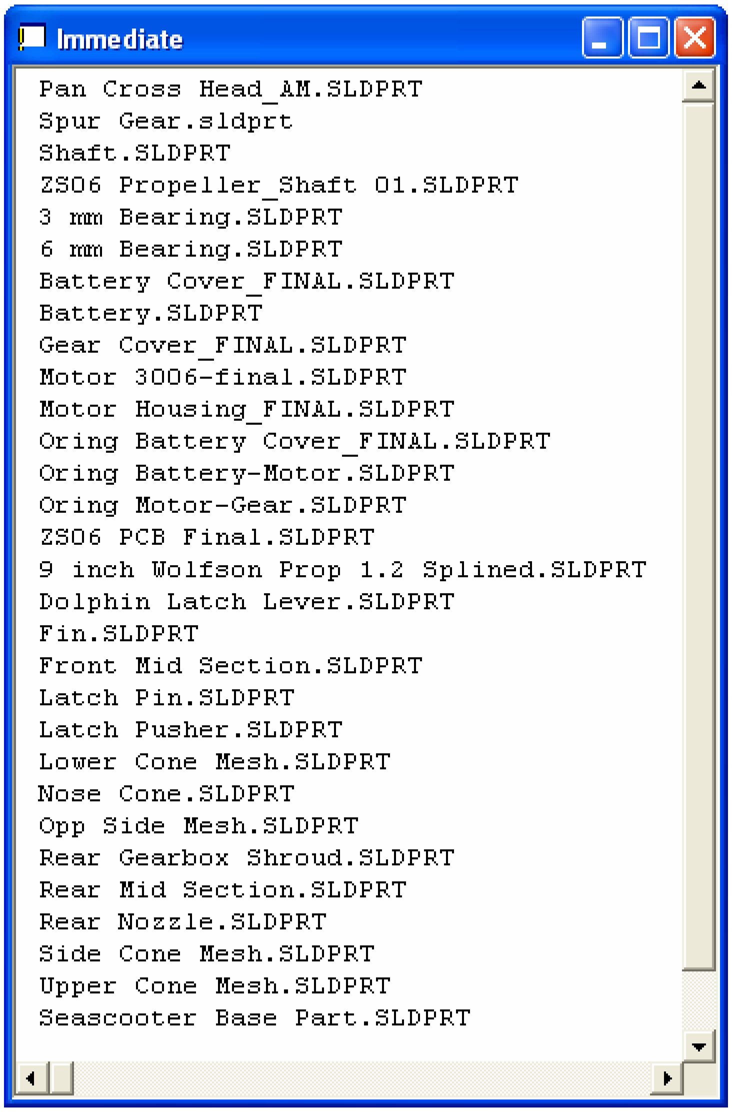
Members of 'PDMWPropertyType'
- pdmwAllProperties
- pdmwConfigurationName
- pdmwDescription
- pdmwDisplayInfo
- pdmwDocumentName
- pdmwNumber
- pdmwOwner
- pdmwRevision
- pdmwStatus
- pdmwUserDefinedProperty

Notice the PDMWPropertyType parameter above. The PDMWPropertyType enumeration is shown here. If we use “pdmwUserDefinedProperty” then we must supply the propertyName parameter. Otherwise we can enter an empty string (“”) for that parameter as we have done in the example above.

Let’s try a few more queries.

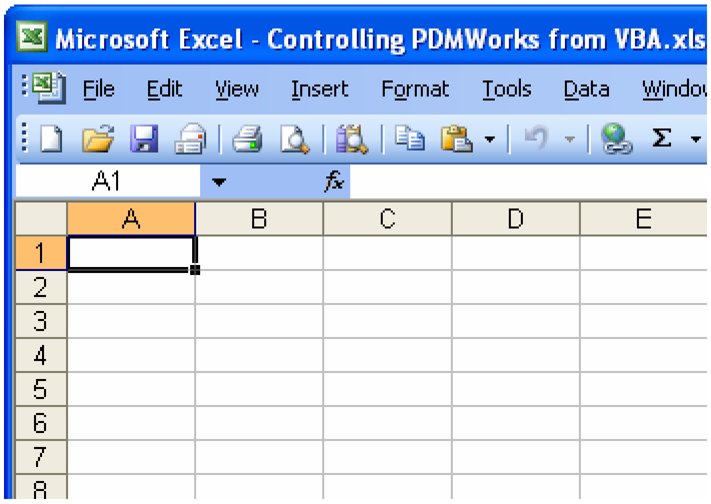
```
Sub QueryTheVaultB()
    Dim myPDMConn As New PDMWConnection
    Dim myPDMWSRs As PDMWSearchResults
    Dim myPDMWSO As PDMWSearchOptions
    Dim myPDMWSR As PDMWSearchResult
    myPDMConn.Login "pdmwadmin", "pdmwadmin", "localhost"
    myPDMConn.Refresh
    Set myPDMWSO = myPDMConn.GetSearchOptionsObject
    myPDMWSO.SearchCriteria.AddCriteria pdmwAnd, _
              pdmwDocumentName, _
              "", _
              pdmwContains, _
              ".sldprt"

    myPDMWSO.SearchCriteria.AddCriteria pdmwOr, _
              pdmwDocumentName, _
              "", _
              pdmwContains, _
              ".prt"
    Set myPDMWSRs = myPDMConn.Search(myPDMWSO)
    For Each myPDMWSR In myPDMWSRs
        If Not myPDMWSR.Document Is Nothing Then
            Debug.Print myPDMWSR.Name
        End If
    Next
    myPDMConn.Logout
End Sub
```

Now we are looking for Documents with ".sldprt" OR ".prt" in file name. Printing the results to the Immediate window (Debug.Print prints to the Immediate window) is a start. But if we are going to take the time to query PDMWorks, let's put the data in a place where we can make some use of it.



```
Pan Cross Head_AM.SLDPRT
Spur Gear.sldprt
Shaft.SLDPRT
ZS06 Propeller_Shaft 01.SLDPRT
3 mm Bearing.SLDPRT
6 mm Bearing.SLDPRT
Battery Cover_FINAL.SLDPRT
Battery.SLDPRT
Gear Cover_FINAL.SLDPRT
Motor 3006-final.SLDPRT
Motor Housing_FINAL.SLDPRT
Oring Battery Cover_FINAL.SLDPRT
Oring Battery-Motor.SLDPRT
Oring Motor-Gear.SLDPRT
ZS06 PCB Final.SLDPRT
9 inch Wolfson Prop 1.2 Splined.SLDPRT
Dolphin Latch Lever.SLDPRT
Fin.SLDPRT
Front Mid Section.SLDPRT
Latch Pin.SLDPRT
Latch Pusher.SLDPRT
Lower Cone Mesh.SLDPRT
Nose Cone.SLDPRT
Opp Side Mesh.SLDPRT
Rear Gearbox Shroud.SLDPRT
Rear Mid Section.SLDPRT
Rear Nozzle.SLDPRT
Side Cone Mesh.SLDPRT
Upper Cone Mesh.SLDPRT
Seascooter Base Part.SLDPRT
```

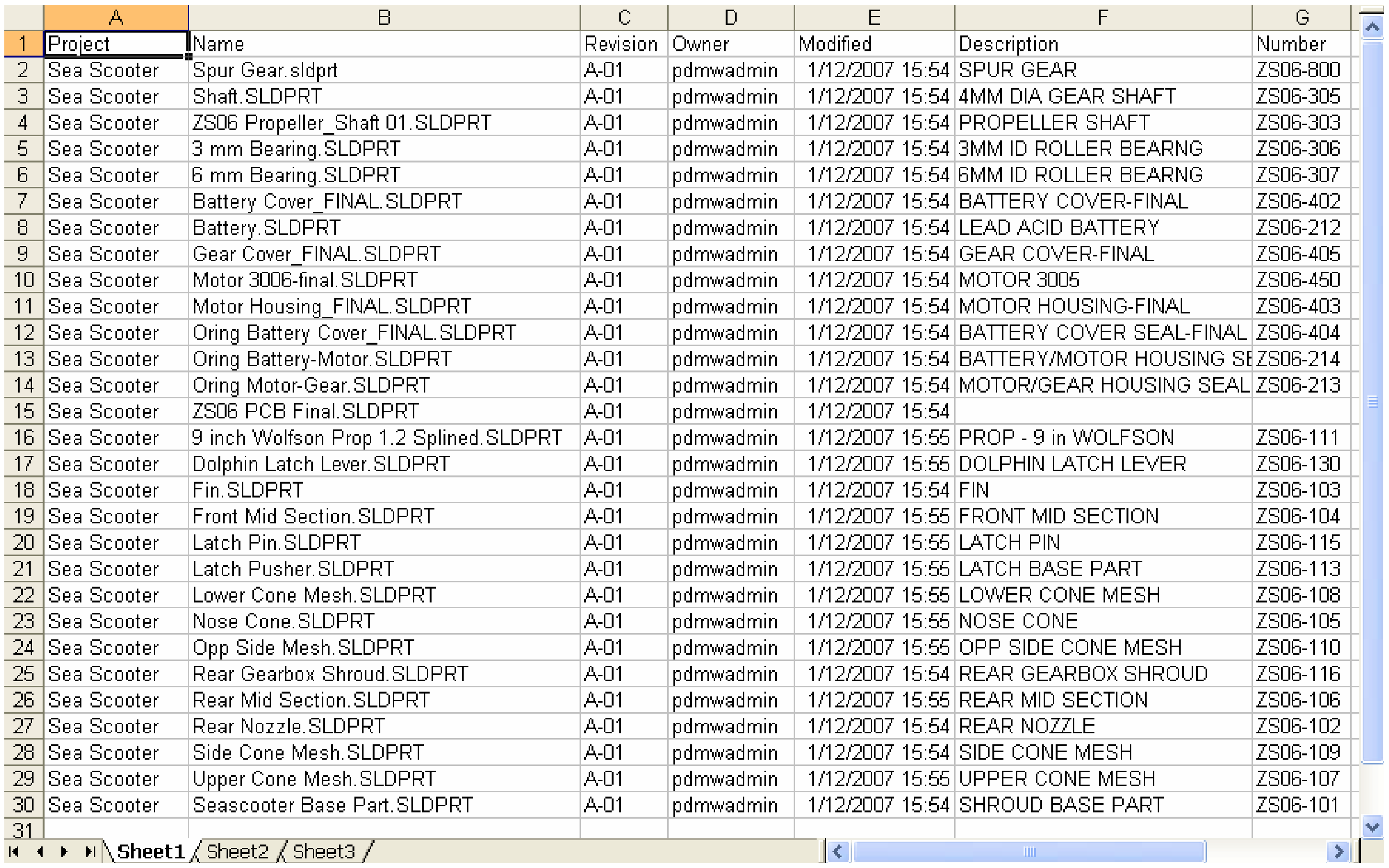
Export Queried Data to Excel


Microsoft Excel - Controlling PDMWorks from VBA.xls
File Edit View Insert Format Tools Data Window
A1
A B C D E
1 2 3 4 5 6 7 8

Microsoft Excel. Workbooks, Worksheets, Rows, Columns, Cells. Everything we need to store information. Right? Let's see if we can store PDMWorks Vault information in an Excel Worksheet.

The results of "QueryTheVaultC" are shown on the next page. But don't turn the page. Take a look at the code and see if you can tell what the results will look like.

```
Sub QueryTheVaultC()
    Dim myPDMConn As New PDMWConnection
    Dim myPDMWSO As PDMWSearchOptions
    Dim myPDMWSRs As PDMWSearchResults
    Dim myPDMWSR As PDMWSearchResult
    Dim CurRow As Long
    myPDMConn.Login "pdmwadmin", "pdmwadmin", "localhost"
    myPDMConn.Refresh
    Set myPDMWSO = myPDMConn.GetSearchOptionsObject
    myPDMWSO.SearchCriteria.AddCriteria pdmwAnd, _
             pdmwDocumentName, _
            "", _
            pdmwContains, _
            ".sldprt"

    myPDMWSO.SearchCriteria.AddCriteria pdmwOr, _
             pdmwDocumentName, _
            "", _
            pdmwContains, _
            ".prt"
    Set myPDMWSRs = myPDMConn.Search(myPDMWSO)
    CurRow = 2
    Sheet1.Cells(1, "A") = "Project"
    Sheet1.Cells(1, "B") = "Name"
    Sheet1.Cells(1, "C") = "Revision"
    Sheet1.Cells(1, "D") = "Owner"
    Sheet1.Cells(1, "E") = "Modified"
    Sheet1.Cells(1, "F") = "Description"
    Sheet1.Cells(1, "G") = "Number"
    For Each myPDMWSR In myPDMWSRs
        If Not myPDMWSR.Document Is Nothing Then
            If myPDMWSR.project = "Sea Scooter" Then
                Sheet1.Cells(CurRow, "A") = myPDMWSR.Document.project
                Sheet1.Cells(CurRow, "B") = myPDMWSR.Document.Name
                Sheet1.Cells(CurRow, "C") = myPDMWSR.Document.Revision
                Sheet1.Cells(CurRow, "D") = myPDMWSR.Document.Owner
                Sheet1.Cells(CurRow, "E") = myPDMWSR.Document.DateModified
                Sheet1.Cells(CurRow, "F") = myPDMWSR.Document.Description
                Sheet1.Cells(CurRow, "G") = myPDMWSR.Document.Number
                CurRow = CurRow + 1
            End If
        End If
    Next
    myPDMConn.Logout
End Sub
```

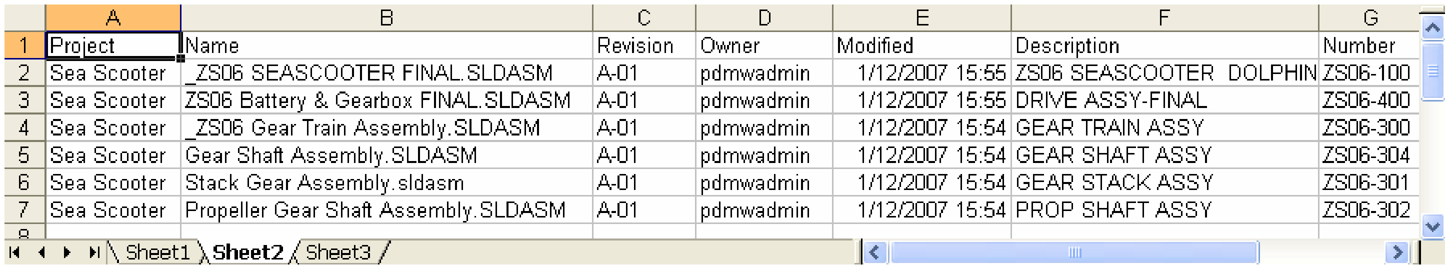


	A	B	C	D	E	F	G
1	Project	Name	Revision	Owner	Modified	Description	Number
2	Sea Scooter	Spur Gear.sldprt	A-01	pdmwadmin	1/12/2007 15:54	SPUR GEAR	ZS06-800
3	Sea Scooter	Shaft.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	4MM DIA GEAR SHAFT	ZS06-305
4	Sea Scooter	ZS06 Propeller_Shaft 01.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	PROPELLER SHAFT	ZS06-303
5	Sea Scooter	3 mm Bearing.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	3MM ID ROLLER BEARNG	ZS06-306
6	Sea Scooter	6 mm Bearing.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	6MM ID ROLLER BEARNG	ZS06-307
7	Sea Scooter	Battery Cover_FINAL.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	BATTERY COVER-FINAL	ZS06-402
8	Sea Scooter	Battery.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	LEAD ACID BATTERY	ZS06-212
9	Sea Scooter	Gear Cover_FINAL.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	GEAR COVER-FINAL	ZS06-405
10	Sea Scooter	Motor 3006-final.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	MOTOR 3005	ZS06-450
11	Sea Scooter	Motor Housing_FINAL.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	MOTOR HOUSING-FINAL	ZS06-403
12	Sea Scooter	Oring Battery Cover_FINAL.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	BATTERY COVER SEAL-FINAL	ZS06-404
13	Sea Scooter	Oring Battery-Motor.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	BATTERY/MOTOR HOUSING SE	ZS06-214
14	Sea Scooter	Oring Motor-Gear.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	MOTOR/GEAR HOUSING SEAL	ZS06-213
15	Sea Scooter	ZS06 PCB Final.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54		
16	Sea Scooter	9 inch Wolfson Prop 1.2 Splined.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	PROP - 9 in WOLFSON	ZS06-111
17	Sea Scooter	Dolphin Latch Lever.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	DOLPHIN LATCH LEVER	ZS06-130
18	Sea Scooter	Fin.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	FIN	ZS06-103
19	Sea Scooter	Front Mid Section.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	FRONT MID SECTION	ZS06-104
20	Sea Scooter	Latch Pin.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	LATCH PIN	ZS06-115
21	Sea Scooter	Latch Pusher.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	LATCH BASE PART	ZS06-113
22	Sea Scooter	Lower Cone Mesh.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	LOWER CONE MESH	ZS06-108
23	Sea Scooter	Nose Cone.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	NOSE CONE	ZS06-105
24	Sea Scooter	Opp Side Mesh.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	OPP SIDE CONE MESH	ZS06-110
25	Sea Scooter	Rear Gearbox Shroud.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	REAR GEARBOX SHROUD	ZS06-116
26	Sea Scooter	Rear Mid Section.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	REAR MID SECTION	ZS06-106
27	Sea Scooter	Rear Nozzle.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	REAR NOZZLE	ZS06-102
28	Sea Scooter	Side Cone Mesh.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	SIDE CONE MESH	ZS06-109
29	Sea Scooter	Upper Cone Mesh.SLDPRT	A-01	pdmwadmin	1/12/2007 15:55	UPPER CONE MESH	ZS06-107
30	Sea Scooter	Seascooter Base Part.SLDPRT	A-01	pdmwadmin	1/12/2007 15:54	SHROUD BASE PART	ZS06-101
31							

Now, that's a thing of beauty. Look at all of that great data. It looks like "pdmwadmin" has been busy. Let's try another query. This time we will be looking for Assembly files.

```
Sub QueryTheVaultD()
    Dim myPDMConn As New PDMWConnection
    Dim myPDMWSO As PDMWSearchOptions
    Dim myPDMWSRs As PDMWSearchResults
    Dim myPDMWSR As PDMWSearchResult
    Dim CurRow As Long
    myPDMConn.Login "pdmwadmin", "pdmwadmin", "localhost"
    myPDMConn.Refresh
    Set myPDMWSO = myPDMConn.GetSearchOptionsObject
    myPDMWSO.SearchCriteria.AddCriteria pdmwAnd, _
             pdmwDocumentName, _
            "", _
            pdmwContains, _
            ".sldasm"

    myPDMWSO.SearchCriteria.AddCriteria pdmwOr, _
             pdmwDocumentName, _
            "", _
            pdmwContains, _
            ".asm"
    Set myPDMWSRs = myPDMConn.Search(myPDMWSO)
    CurRow = 2
    Sheet2.Cells(1, "A") = "Project"
    Sheet2.Cells(1, "B") = "Name"
    Sheet2.Cells(1, "C") = "Revision"
    Sheet2.Cells(1, "D") = "Owner"
    Sheet2.Cells(1, "E") = "Modified"
    Sheet2.Cells(1, "F") = "Description"
    Sheet2.Cells(1, "G") = "Number"
    For Each myPDMWSR In myPDMWSRs
        If Not myPDMWSR.Document Is Nothing Then
            If myPDMWSR.project = "Sea Scooter" Then
                Sheet2.Cells(CurRow, "A") = myPDMWSR.Document.project
                Sheet2.Cells(CurRow, "B") = myPDMWSR.Document.Name
                Sheet2.Cells(CurRow, "C") = myPDMWSR.Document.Revision
                Sheet2.Cells(CurRow, "D") = myPDMWSR.Document.Owner
                Sheet2.Cells(CurRow, "E") = myPDMWSR.Document.DateModified
                Sheet2.Cells(CurRow, "F") = myPDMWSR.Document.Description
                Sheet2.Cells(CurRow, "G") = myPDMWSR.Document.Number
                CurRow = CurRow + 1
            End If
        End If
    Next
    myPDMConn.Logout
End Sub
```

OK. That was really hard. Everything that had said "Sheet1" now says "Sheet2". And we are looking for ".sldasm" and ".asm". It's just so difficult. Or not.

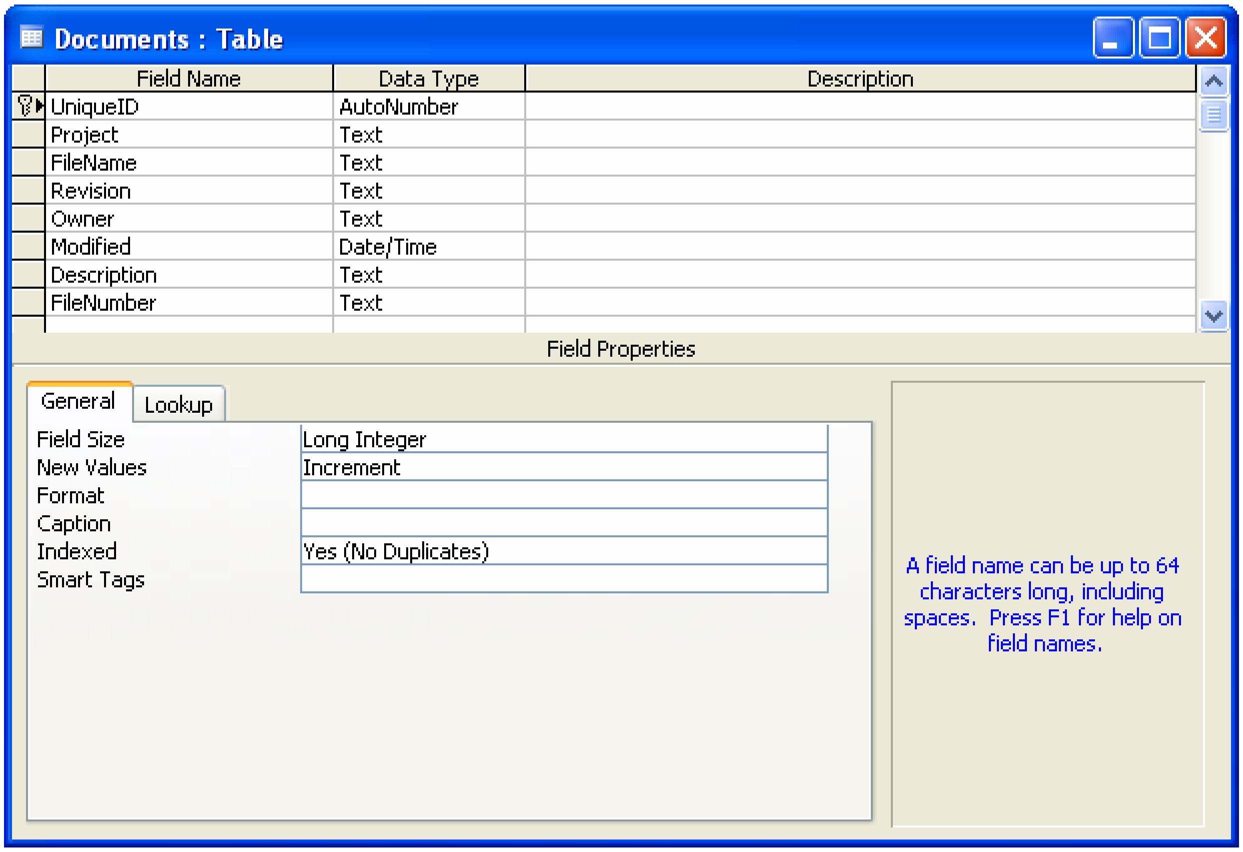


	A	B	C	D	E	F	G
1	Project	Name	Revision	Owner	Modified	Description	Number
2	Sea Scooter	_ZS06 SEASCOOTER FINAL.SLDASM	A-01	pdmwadmin	1/12/2007 15:55	ZS06 SEASCOOTER DOLPHIN	ZS06-100
3	Sea Scooter	ZS06 Battery & Gearbox FINAL.SLDASM	A-01	pdmwadmin	1/12/2007 15:55	DRIVE ASSY-FINAL	ZS06-400
4	Sea Scooter	_ZS06 Gear Train Assembly.SLDASM	A-01	pdmwadmin	1/12/2007 15:54	GEAR TRAIN ASSY	ZS06-300
5	Sea Scooter	Gear Shaft Assembly.SLDASM	A-01	pdmwadmin	1/12/2007 15:54	GEAR SHAFT ASSY	ZS06-304
6	Sea Scooter	Stack Gear Assembly.sldasm	A-01	pdmwadmin	1/12/2007 15:54	GEAR STACK ASSY	ZS06-301
7	Sea Scooter	Propeller Gear Shaft Assembly.SLDASM	A-01	pdmwadmin	1/12/2007 15:54	PROP SHAFT ASSY	ZS06-302
8							

What would we do if we wanted to query for Drawing files? Instead of ".sldasm", we would use ______________. Rather than ".asm" we would use ________. And if we wanted to place the Drawing data on Sheet 3, we would change "Sheet2" to __________.

Export Queried Data to Access

We can see how easy it is to 'export' data from PDMWorks into Excel. Is this because the code is running in Excel? Surely, saving data to an Access database must be difficult.



Field Name	Data Type	Description
UniqueID	AutoNumber	
Project	Text	
FileName	Text	
Revision	Text	
Owner	Text	
Modified	Date/Time	
Description	Text	
FileNumber	Text	

Field Size	Long Integer
New Values	Increment
Format	
Caption	
Indexed	Yes (No Duplicates)
Smart Tags	

A field name can be up to 64 characters long, including spaces. Press F1 for help on field names.

Above, we see the structure for an Access Database Table named "Documents". Take a look at the File Names. They look a lot like the row headings in the Excel extraction we just worked with. Don't they?

The database we are going to use has a path of:

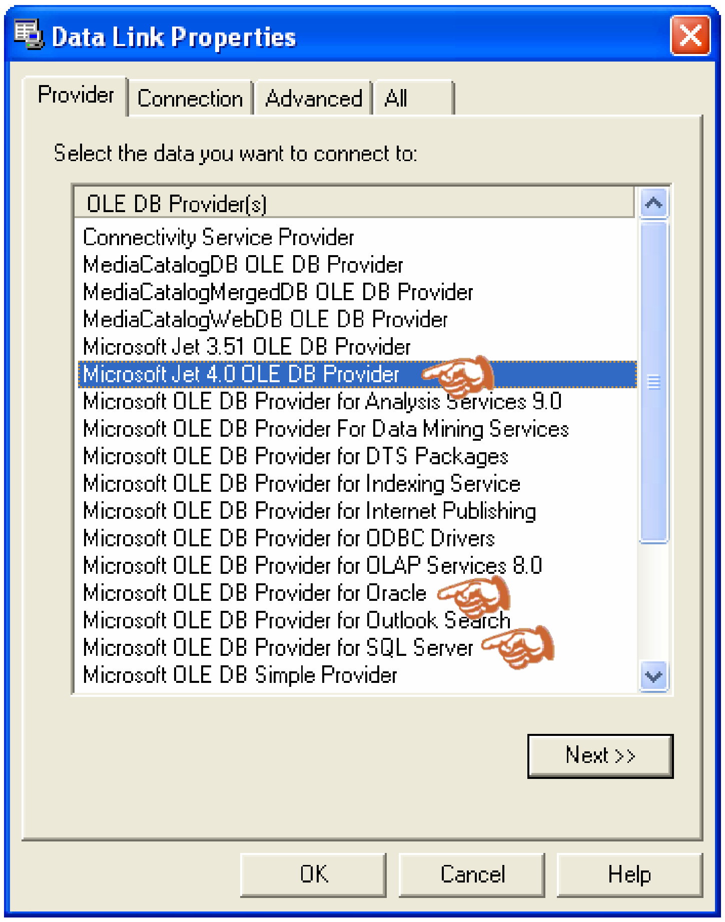
C:\data\SWWorld 2007\PDMWorks\PDMWorks Data.mdb

We are going to use a UDL File to help us work with this database. The path of the UDL file is:

C:\data\SWWorld 2007\PDMWorks\PDMWorksDatabase.udl

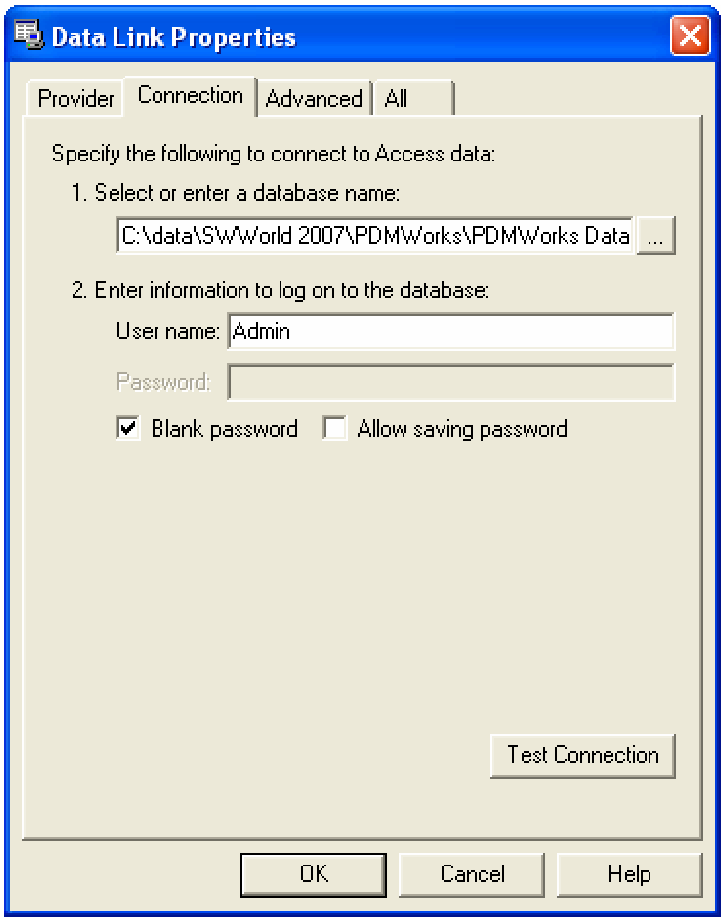
To create a UDL file, follow these steps:

A. Right-click in a Windows Explorer Folder and select **New > Text Document**.
B. Windows will create a new .txt file and will wait for you to rename it. Change the name to "PDMWorksDatabase.udl".
C. Windows will ask if you are sure the file extension should be changed. Click "Yes".
D. Double-click on the UDL file to change it's settings.



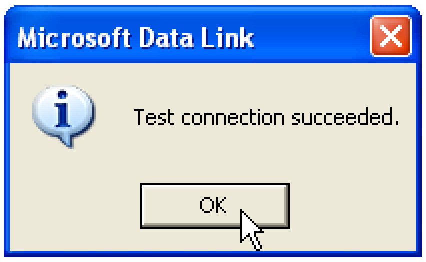
Select the data you want to connect to:

OLE DB Provider(s)
Connectivity Service Provider
MediaCatalogDB OLE DB Provider
MediaCatalogMergedDB OLE DB Provider
MediaCatalogWebDB OLE DB Provider
Microsoft Jet 3.51 OLE DB Provider
Microsoft Jet 4.0 OLE DB Provider
Microsoft OLE DB Provider for Analysis Services 9.0
Microsoft OLE DB Provider For Data Mining Services
Microsoft OLE DB Provider for DTS Packages
Microsoft OLE DB Provider for Indexing Service
Microsoft OLE DB Provider for Internet Publishing
Microsoft OLE DB Provider for ODBC Drivers
Microsoft OLE DB Provider for OLAP Services 8.0
Microsoft OLE DB Provider for Oracle
Microsoft OLE DB Provider for Outlook Search
Microsoft OLE DB Provider for SQL Server
Microsoft OLE DB Simple Provider



Specify the following to connect to Access data:
1. Select or enter a database name:
C:\data\SWWorld 2007\PDMWorks\PDMWorks Data
2. Enter information to log on to the database:
User name: Admin
Password:
[✓] Blank password [ ] Allow saving password

Here we can see that we are using the “Microsoft Jet 4.0 OLE EB Provider”. And the path to the database is selected in the Connection Tab.



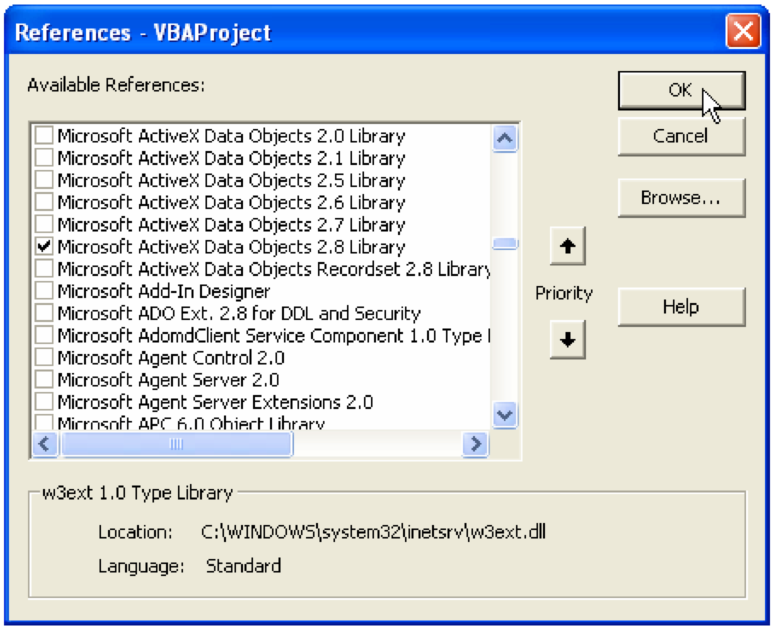
Test connection succeeded.

If we click the “Test Connection” button, we should get a favorable result.

We will experience why this UDL file is so important in a little while. But for now, just take a look at the OLE DB Providers list. Notice the Oracle and SQL Server providers? I have a feeling we are going to make use of these Providers.

OK. We have a Database. We have a UDL file. It’s time to get back into our programming environment.

The next thing we need to do is add a Reference to the “Microsoft ActiveX Data Objects #.# Library”. As we see here, 2.8 is the highest version on this computer so we will use it. It is alright if the highest version on your computer is different.



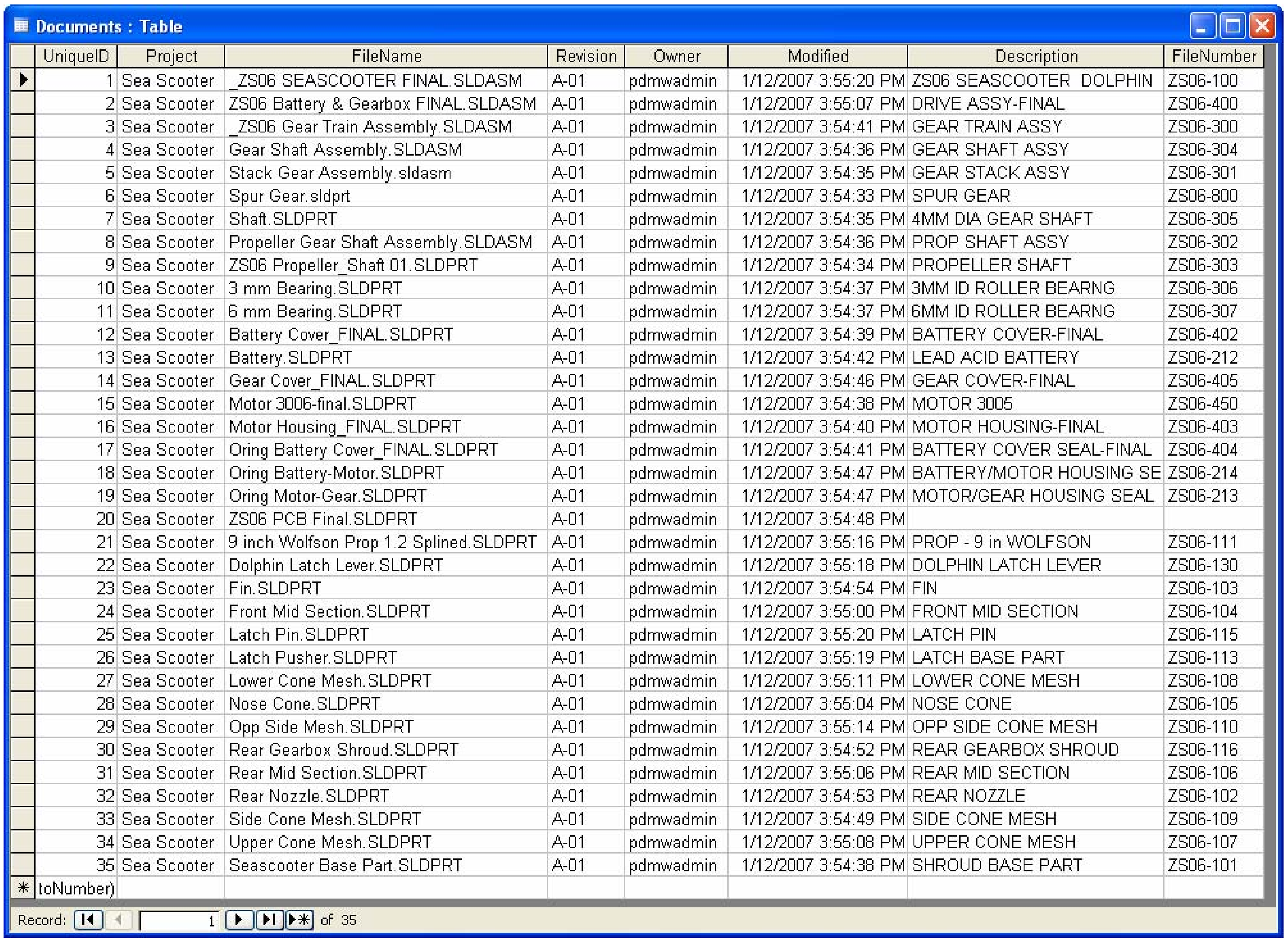
Available References:

[ ] Microsoft ActiveX Data Objects 2.0 Library
[ ] Microsoft ActiveX Data Objects 2.1 Library
[ ] Microsoft ActiveX Data Objects 2.5 Library
[ ] Microsoft ActiveX Data Objects 2.6 Library
[ ] Microsoft ActiveX Data Objects 2.7 Library
[✓] Microsoft ActiveX Data Objects 2.8 Library
[ ] Microsoft ActiveX Data Objects Recordset 2.8 Library
[ ] Microsoft Add-In Designer
[ ] Microsoft ADO Ext. 2.8 for DDL and Security
[ ] Microsoft AdomdClient Service Component 1.0 Type l
[ ] Microsoft Agent Control 2.0
[ ] Microsoft Agent Server 2.0
[ ] Microsoft Agent Server Extensions 2.0
[ ] Microsoft APC 6.0 Object Library

w3ext 1.0 Type Library
Location: C:\WINDOWS\system32\inetsrv\w3ext.dll
Language: Standard

```
1.  Sub QueryTheVaultE()
2.      Dim myPDMConn As New PDMWConnection
3.      Dim myPDMWSO As PDMWSearchOptions
4.      Dim myPDMWSRs As PDMWSearchResults
5.      Dim myPDMWSR As PDMWSearchResult
6.      Dim myDB As New ADODB.Connection
7.      Dim myRS As New ADODB.Recordset
8.      myPDMConn.Login "pdmwadmin", "pdmwadmin", "localhost"
9.      myPDMConn.Refresh
10.     Set myPDMWSO = myPDMConn.GetSearchOptionsObject
11.     myPDMWSO.SearchCriteria.AddCriteria pdmwAnd, _
                 pdmwDocumentName, _
                "", _
                pdmwContains, _
                ".sldprt"

12.     myPDMWSO.SearchCriteria.AddCriteria pdmwOr, _
                 pdmwDocumentName, _
                "", _
                pdmwContains, _
                ".sldasm"
13.     Set myPDMWSRs = myPDMConn.Search(myPDMWSO)
14.     myDB.Open "File name=C:\data\SWWorld 2007\PDMWorks\PDMWorksDatabase.udl"
15.     myRS.Open "Select * from Documents Where UniqueID = 0", myDB, _
             adOpenDynamic, adLockOptimistic
16.     For Each myPDMWSR In myPDMWSRs
17.         If Not myPDMWSR.Document Is Nothing Then
18.             If myPDMWSR.project = "Sea Scooter" Then
19.                 myRS.AddNew
20.                 myRS("Project").Value = myPDMWSR.Document.project
21.                 myRS("FileName").Value = myPDMWSR.Document.Name
22.                 myRS("Revision").Value = myPDMWSR.Document.Revision
23.                 myRS("Owner").Value = myPDMWSR.Document.Owner
24.                 myRS("Modified").Value = myPDMWSR.Document.DateModified
25.                 myRS("Description").Value = myPDMWSR.Document.Description
26.                 myRS("FileNumber").Value = myPDMWSR.Document.Number
27.                 myRS.Update
28.             End If
29.         End If
30.     Next
31.     myRS.Close
32.     myDB.Close
33.     myPDMConn.Logout
34. End Sub
```

Much of the code is the same as when we were writing to Microsoft Excel. Notice Line 14. This is where we open the database using the UDL file. We open the “Documents” table on Line 15. Lines 19 thru 27 are used to add new records to the database.



Documents : Table

	UniqueID	Project	FileName	Revision	Owner	Modified	Description	FileNumber
▶	1	Sea Scooter	_ZS06 SEASCOOTER FINAL.SLDASM	A-01	pdmwadmin	1/12/2007 3:55:20 PM	ZS06 SEASCOOTER DOLPHIN	ZS06-100
	2	Sea Scooter	ZS06 Battery & Gearbox FINAL.SLDASM	A-01	pdmwadmin	1/12/2007 3:55:07 PM	DRIVE ASSY-FINAL	ZS06-400
	3	Sea Scooter	_ZS06 Gear Train Assembly.SLDASM	A-01	pdmwadmin	1/12/2007 3:54:41 PM	GEAR TRAIN ASSY	ZS06-300
	4	Sea Scooter	Gear Shaft Assembly.SLDASM	A-01	pdmwadmin	1/12/2007 3:54:36 PM	GEAR SHAFT ASSY	ZS06-304
	5	Sea Scooter	Stack Gear Assembly.sldasm	A-01	pdmwadmin	1/12/2007 3:54:35 PM	GEAR STACK ASSY	ZS06-301
	6	Sea Scooter	Spur Gear.sldprt	A-01	pdmwadmin	1/12/2007 3:54:33 PM	SPUR GEAR	ZS06-800
	7	Sea Scooter	Shaft.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:35 PM	4MM DIA GEAR SHAFT	ZS06-305
	8	Sea Scooter	Propeller Gear Shaft Assembly.SLDASM	A-01	pdmwadmin	1/12/2007 3:54:36 PM	PROP SHAFT ASSY	ZS06-302
	9	Sea Scooter	ZS06 Propeller_Shaft 01.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:34 PM	PROPELLER SHAFT	ZS06-303
	10	Sea Scooter	3 mm Bearing.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:37 PM	3MM ID ROLLER BEARNG	ZS06-306
	11	Sea Scooter	6 mm Bearing.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:37 PM	6MM ID ROLLER BEARNG	ZS06-307
	12	Sea Scooter	Battery Cover_FINAL.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:39 PM	BATTERY COVER-FINAL	ZS06-402
	13	Sea Scooter	Battery.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:42 PM	LEAD ACID BATTERY	ZS06-212
	14	Sea Scooter	Gear Cover_FINAL.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:46 PM	GEAR COVER-FINAL	ZS06-405
	15	Sea Scooter	Motor 3006-final.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:38 PM	MOTOR 3005	ZS06-450
	16	Sea Scooter	Motor Housing_FINAL.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:40 PM	MOTOR HOUSING-FINAL	ZS06-403
	17	Sea Scooter	Oring Battery Cover_FINAL.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:41 PM	BATTERY COVER SEAL-FINAL	ZS06-404
	18	Sea Scooter	Oring Battery-Motor.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:47 PM	BATTERY/MOTOR HOUSING SE	ZS06-214
	19	Sea Scooter	Oring Motor-Gear.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:47 PM	MOTOR/GEAR HOUSING SEAL	ZS06-213
	20	Sea Scooter	ZS06 PCB Final.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:48 PM		
	21	Sea Scooter	9 inch Wolfson Prop 1.2 Splined.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:16 PM	PROP - 9 in WOLFSON	ZS06-111
	22	Sea Scooter	Dolphin Latch Lever.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:18 PM	DOLPHIN LATCH LEVER	ZS06-130
	23	Sea Scooter	Fin.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:54 PM	FIN	ZS06-103
	24	Sea Scooter	Front Mid Section.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:00 PM	FRONT MID SECTION	ZS06-104
	25	Sea Scooter	Latch Pin.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:20 PM	LATCH PIN	ZS06-115
	26	Sea Scooter	Latch Pusher.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:19 PM	LATCH BASE PART	ZS06-113
	27	Sea Scooter	Lower Cone Mesh.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:11 PM	LOWER CONE MESH	ZS06-108
	28	Sea Scooter	Nose Cone.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:04 PM	NOSE CONE	ZS06-105
	29	Sea Scooter	Opp Side Mesh.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:14 PM	OPP SIDE CONE MESH	ZS06-110
	30	Sea Scooter	Rear Gearbox Shroud.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:52 PM	REAR GEARBOX SHROUD	ZS06-116
	31	Sea Scooter	Rear Mid Section.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:06 PM	REAR MID SECTION	ZS06-106
	32	Sea Scooter	Rear Nozzle.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:53 PM	REAR NOZZLE	ZS06-102
	33	Sea Scooter	Side Cone Mesh.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:49 PM	SIDE CONE MESH	ZS06-109
	34	Sea Scooter	Upper Cone Mesh.SLDPRT	A-01	pdmwadmin	1/12/2007 3:55:08 PM	UPPER CONE MESH	ZS06-107
	35	Sea Scooter	Seascooter Base Part.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:38 PM	SHROUD BASE PART	ZS06-101
*	toNumber)							

Record: 1 of 35

How’s this data looking? Not bad. Not bad at all.

Now, Microsoft Access is a great database. It only has problems when people attempt to use it as a major software development environment. You will notice that we are using it to store data, not to run code. The code we are running is in Excel. The code would run equally well in SolidWorks or VB.NET. But let’s imagine that people begin to get worried that having data in Access is problematic. It is time to scale up to SQL Server.

So, the big boss says, “Jones, I want you to scale up your data extraction program to SQL Server. What do you need?”

STOP!!!! Think before you answer. The correct answer is:

“Scaling up to SQL Server is going to take a substantial effort. But I really think if I could get away to the corporate condo in Hawaii for about 3 weeks, I could pull it off.”

The big boss says, “3 weeks? That’s all? I thought it would take longer.”

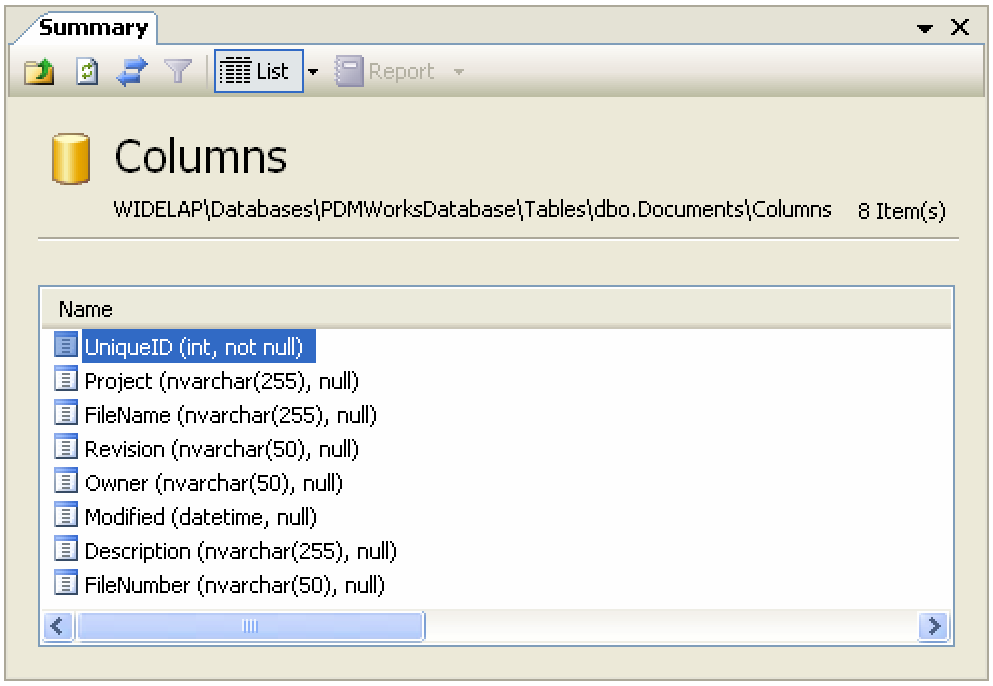
You say, “I will work day and night until it is finished.”

Export Queried Data to SQL Server

SQL Server is an expensive, enterprise-wide, massive, huge, database. Remember, we have 3 weeks to complete the migration from Access to SQL Server.

You arrive at the corporate condo in Hawaii on Monday morning. It is 8:00 AM.

Task 1: Create a SQL Server Database that mimics the Access Database. Time to complete: 6 minutes.

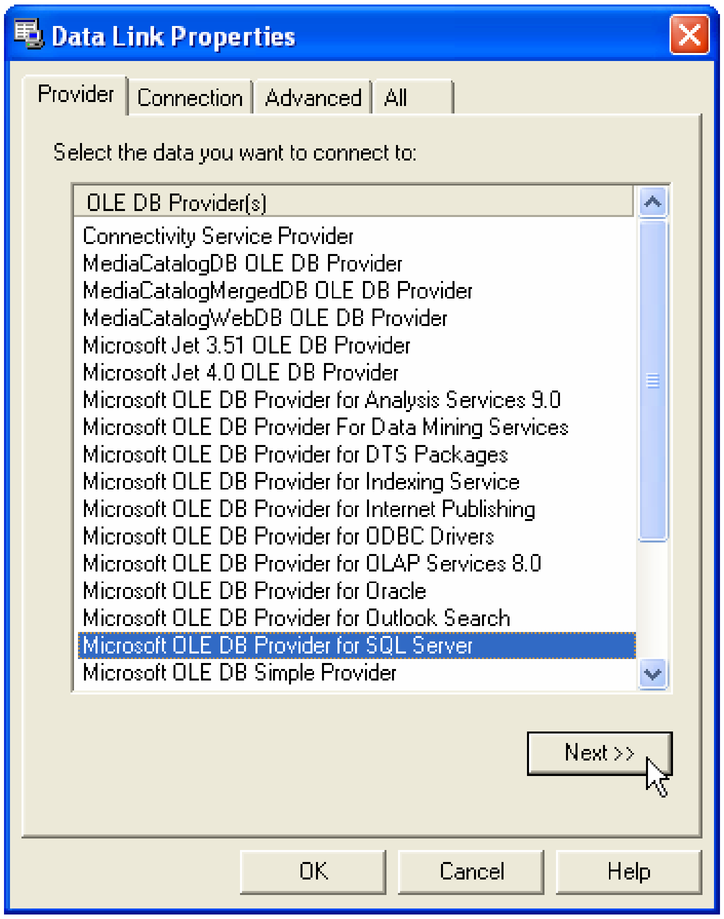


Columns

WIDELAP\Databases\PDMWorksDatabase\Tables\dbo.Documents\Columns 8 Item(s)

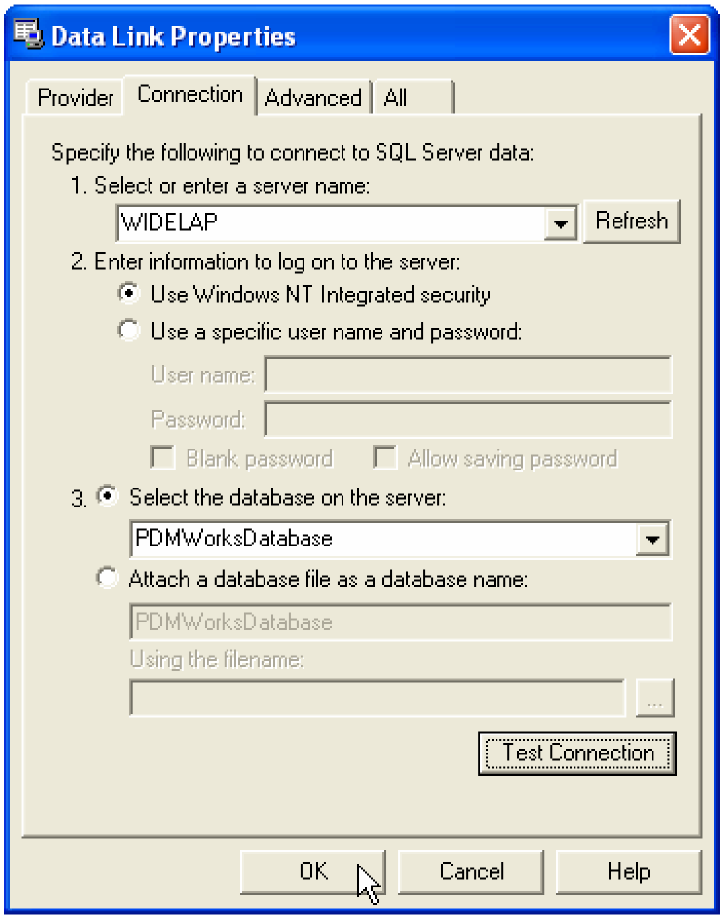
Name
UniqueID (int, not null)
Project (nvarchar(255), null)
FileName (nvarchar(255), null)
Revision (nvarchar(50), null)
Owner (nvarchar(50), null)
Modified (datetime, null)
Description (nvarchar(255), null)
FileNumber (nvarchar(50), null)

Task 2: Double-click on the UDL file. Time to complete: 1 minute.
Task 3: Change the Provider and Connection Tabs as shown below. Time to complete: 2 minutes.



Select the data you want to connect to:

OLE DB Provider(s)
Connectivity Service Provider
MediaCatalogDB OLE DB Provider
MediaCatalogMergedDB OLE DB Provider
MediaCatalogWebDB OLE DB Provider
Microsoft Jet 3.51 OLE DB Provider
Microsoft Jet 4.0 OLE DB Provider
Microsoft OLE DB Provider for Analysis Services 9.0
Microsoft OLE DB Provider For Data Mining Services
Microsoft OLE DB Provider for DTS Packages
Microsoft OLE DB Provider for Indexing Service
Microsoft OLE DB Provider for Internet Publishing
Microsoft OLE DB Provider for ODBC Drivers
Microsoft OLE DB Provider for OLAP Services 8.0
Microsoft OLE DB Provider for Oracle
Microsoft OLE DB Provider for Outlook Search
Microsoft OLE DB Provider for SQL Server
Microsoft OLE DB Simple Provider



Specify the following to connect to SQL Server data:
1. Select or enter a server name: WIDELAP
2. Enter information to log on to the server:
[✓ Use Windows NT Integrated security]
Use a specific user name and password:
User name:
Password:
Blank password Allow saving password
3. [✓ Select the database on the server:] PDMWorksDatabase
Attach a database file as a database name: PDMWorksDatabase
Using the filename:

Task 4: Run the Program. Time to complete: 1 minute.

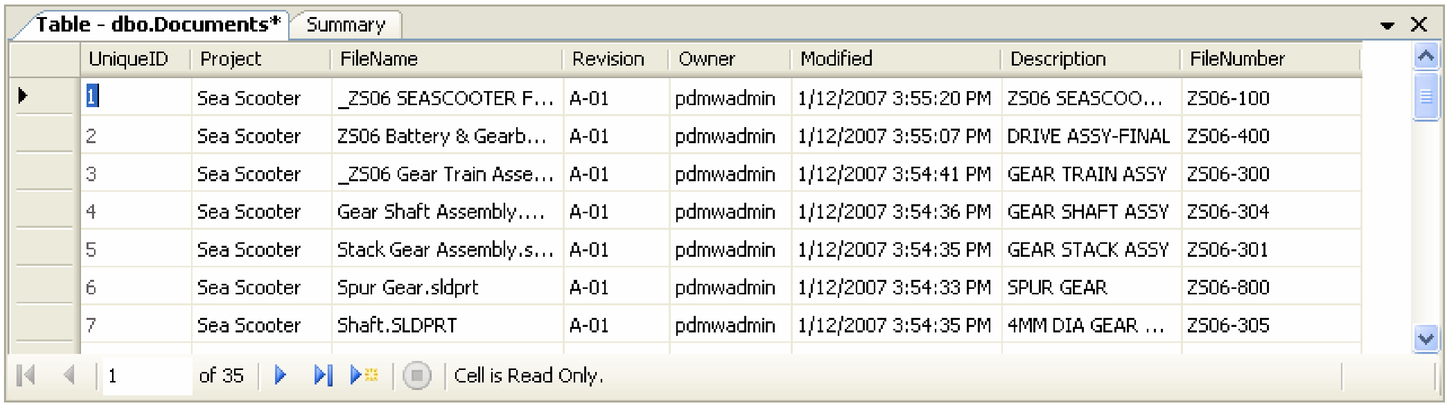


Table - dbo.Documents* | Summary

UniqueID	Project	FileName	Revision	Owner	Modified	Description	FileNumber
1	Sea Scooter	_ZS06 SEASCOOTER F...	A-01	pdmwadmin	1/12/2007 3:55:20 PM	ZS06 SEASCOO...	ZS06-100
2	Sea Scooter	ZS06 Battery & Gearb...	A-01	pdmwadmin	1/12/2007 3:55:07 PM	DRIVE ASSY-FINAL	ZS06-400
3	Sea Scooter	_ZS06 Gear Train Asse...	A-01	pdmwadmin	1/12/2007 3:54:41 PM	GEAR TRAIN ASSY	ZS06-300
4	Sea Scooter	Gear Shaft Assembly....	A-01	pdmwadmin	1/12/2007 3:54:36 PM	GEAR SHAFT ASSY	ZS06-304
5	Sea Scooter	Stack Gear Assembly.s...	A-01	pdmwadmin	1/12/2007 3:54:35 PM	GEAR STACK ASSY	ZS06-301
6	Sea Scooter	Spur Gear.sldprt	A-01	pdmwadmin	1/12/2007 3:54:33 PM	SPUR GEAR	ZS06-800
7	Sea Scooter	Shaft.SLDPRT	A-01	pdmwadmin	1/12/2007 3:54:35 PM	4MM DIA GEAR ...	ZS06-305

1 of 35 | Cell is Read Only.

The time is now 8:10 AM on Monday morning. What are you going to do until 5:00 PM today? What about tomorrow? What about for the next 2 weeks and 6 days? I have great confidence that you will think of something to do.

Tuesday.
Wednesday.
Thursday.
Friday.
Saturday.
Sunday.
Monday.
Tuesday.
Wednesday.
Thursday.
Friday.
Saturday.
Sunday.
Monday.
Tuesday.
Wednesday.
Thursday.
Friday.
Saturday.
Sunday.

It's time to go home. You're exhausted. Not because of the sleepless nights due to this massive programming effort but because of the fun you have had in Hawaii.

ANSWER THIS QUESTION:

How much code did we re-write? ______________________________

Export Queried Data to Oracle

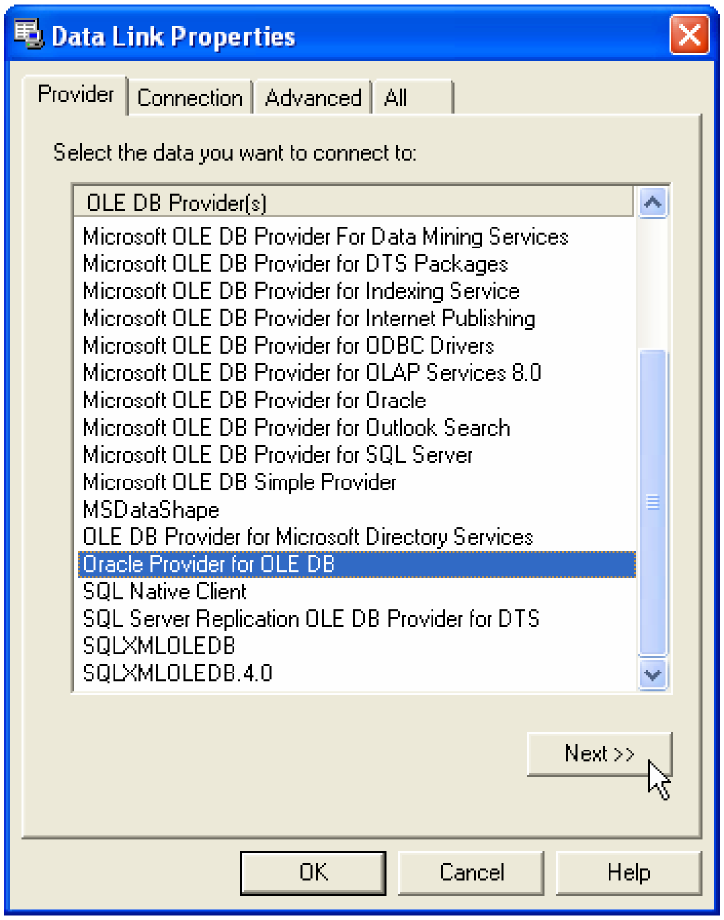
Your boss is so impressed with the hard work you have done that you are given 2 weeks of paid vacation at . . . . the corporate condo in Hawaii. You graciously turn down the offer and are given a substantial raise instead. And it's a good thing that you are in the office because a new CIO is hired and the new CIO loves Oracle and despises everything else.

You receive a phone call asking you to attend the emergency Board of Directors meeting. The new CIO demands that our SQL Server tracking system be moved into Oracle. Fortunately, he has just left a company where a similar migration took 6 months to complete. We want to impress him, though. Right? So, in front of the entire Board of Directors, we suggest that we may be able to complete the task in 4 months of uninterrupted time. But where could we go? Aaah. French. You don't speak French. If you could only be persuaded to spend 4 months in the Paris, France Condo. You would be less likely to be interrupted because you don't even speak the language. And if your company could leave you alone, that would be good too. You will send bi-monthly updates to the CIO directly.

Off you go to France. You're in the condo at 8:00 AM.

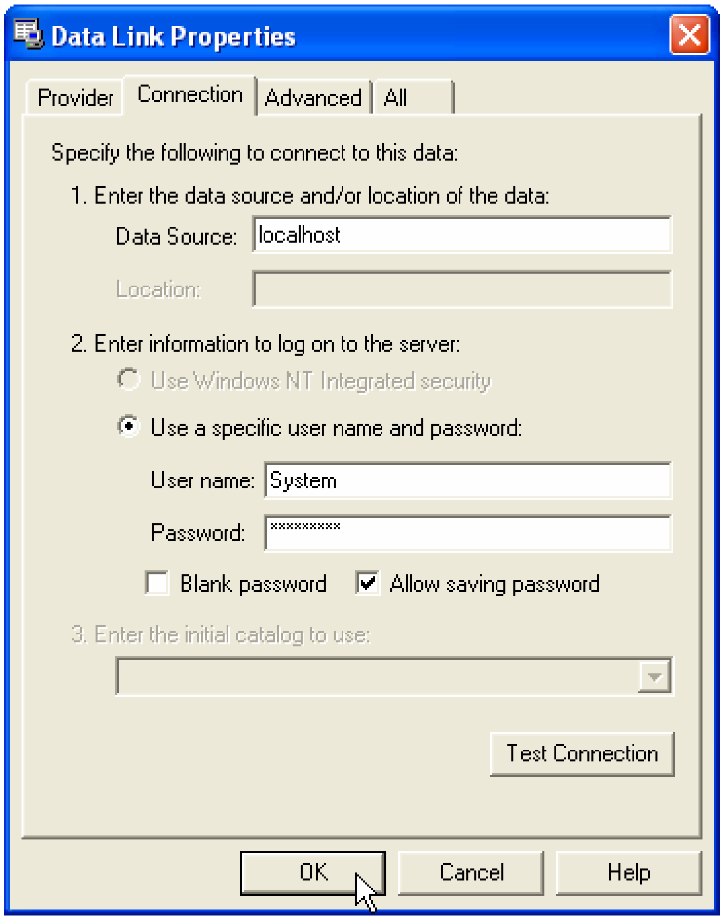
Task 1: Create an Oracle Database that mimics the SQL Server Database. Time to complete: 6 minutes.
Task 2: Double-click on the UDL file. Time to complete: 1 minute.
Task 3: Change the Provider and Connection Tabs as shown below. Time to complete: 2 minutes.

Username: System
Password: pdmwadmin



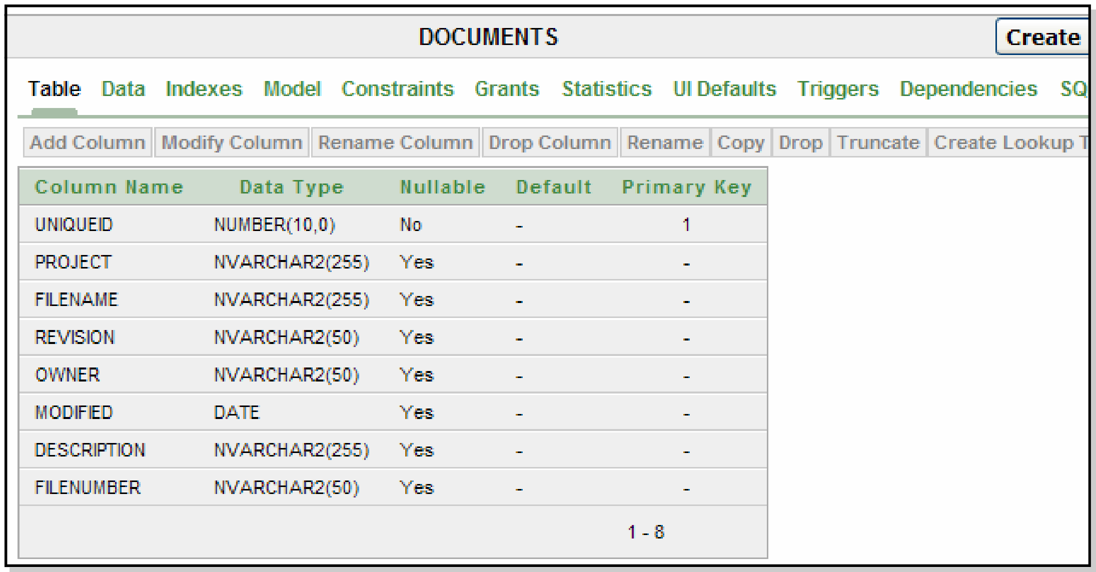
Select the data you want to connect to:

OLE DB Provider(s)
Microsoft OLE DB Provider For Data Mining Services
Microsoft OLE DB Provider for DTS Packages
Microsoft OLE DB Provider for Indexing Service
Microsoft OLE DB Provider for Internet Publishing
Microsoft OLE DB Provider for ODBC Drivers
Microsoft OLE DB Provider for OLAP Services 8.0
Microsoft OLE DB Provider for Oracle
Microsoft OLE DB Provider for Outlook Search
Microsoft OLE DB Provider for SQL Server
Microsoft OLE DB Simple Provider
MSDataShape
OLE DB Provider for Microsoft Directory Services
Oracle Provider for OLE DB
SQL Native Client
SQL Server Replication OLE DB Provider for DTS
SQLXMLOLEDB
SQLXMLOLEDB.4.0



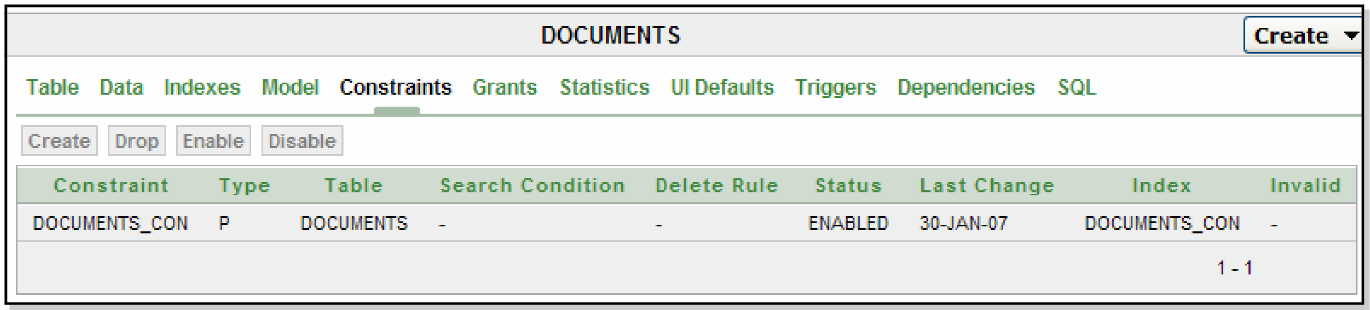
Specify the following to connect to this data:
1. Enter the data source and/or location of the data:
Data Source: localhost
Location:
2. Enter information to log on to the server:
○ Use Windows NT Integrated security
● Use a specific user name and password:
User name: System
Password: *********
☐ Blank password ☑ Allow saving password
3. Enter the initial catalog to use:

Task 4: Run the Program. Time to complete: 1 minute.


DOCUMENTS
Create
Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQ
Add Column Modify Column Rename Column Drop Column Rename Copy Drop Truncate Create Lookup T

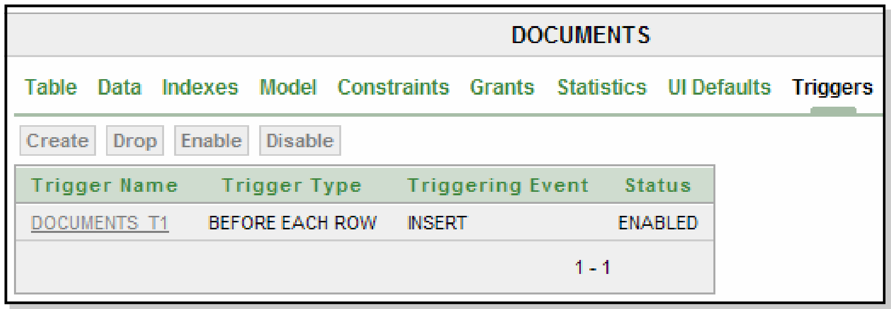
Column Name	Data Type	Nullable	Default	Primary Key
UNIQUEID	NUMBER(10,0)	No	-	1
PROJECT	NVARCHAR2(255)	Yes	-	-
FILENAME	NVARCHAR2(255)	Yes	-	-
REVISION	NVARCHAR2(50)	Yes	-	-
OWNER	NVARCHAR2(50)	Yes	-	-
MODIFIED	DATE	Yes	-	-
DESCRIPTION	NVARCHAR2(255)	Yes	-	-
FILENUMBER	NVARCHAR2(50)	Yes	-	-

1 - 8


DOCUMENTS
Create
Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL
Create Drop Enable Disable

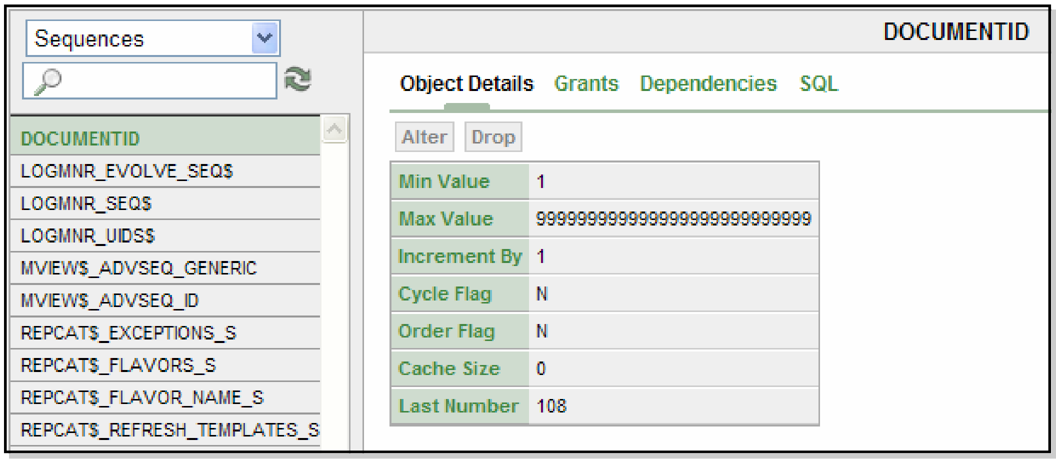
Constraint	Type	Table	Search Condition	Delete Rule	Status	Last Change	Index	Invalid
DOCUMENTS_CON	P	DOCUMENTS	-	-	ENABLED	30-JAN-07	DOCUMENTS_CON	-

1 - 1


DOCUMENTS
Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers
Create Drop Enable Disable

Trigger Name	Trigger Type	Triggering Event	Status
DOCUMENTS_T1	BEFORE EACH ROW	INSERT	ENABLED

1 - 1



DOCUMENTID
LOGMNR_EVOLVE_SEQ$
LOGMNR_SEQ$
LOGMNR_UIDS$
MVIEWS$_ADVSEQ_GENERIC
MVIEWS$_ADVSEQ_ID
REPCAT$_EXCEPTIONS_S
REPCAT$_FLAVORS_S
REPCAT$_FLAVOR_NAME_S
REPCAT$_REFRESH_TEMPLATES_S

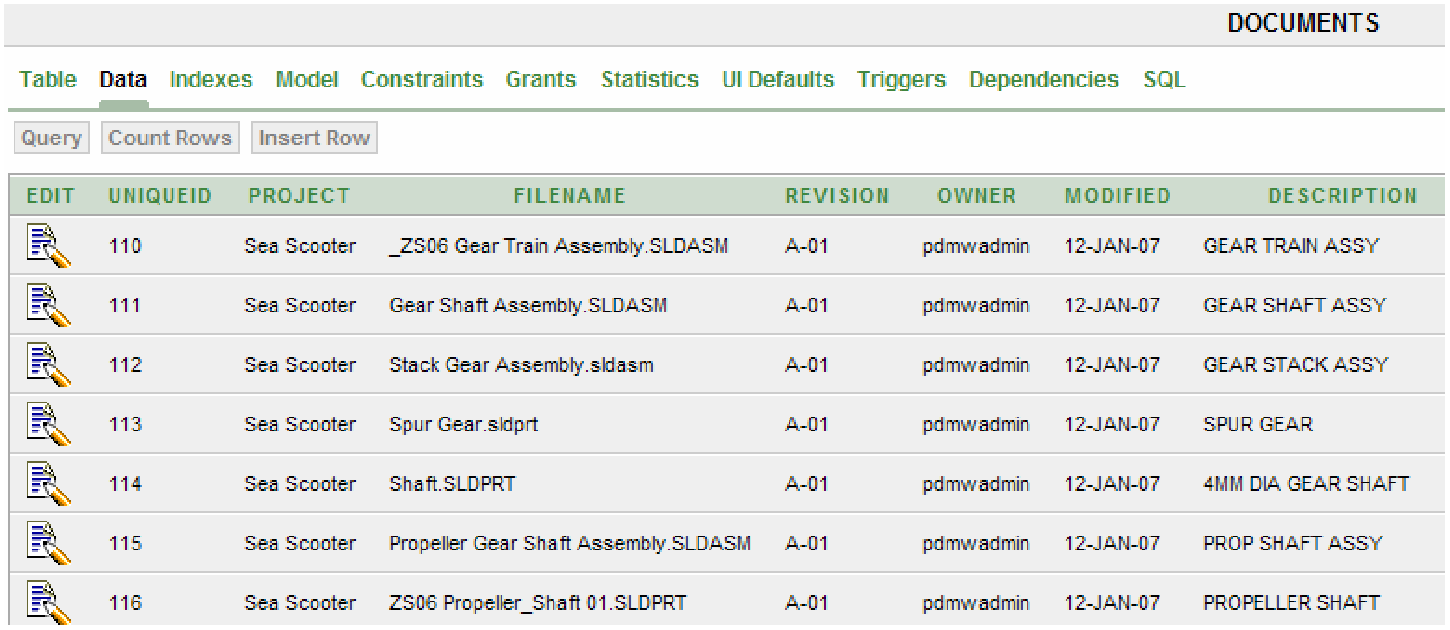
Min Value	1
Max Value	999999999999999999999999999
Increment By	1
Cycle Flag	N
Order Flag	N
Cache Size	0
Last Number	108

A Table named Documents has a Primary Key Constraint on the UniqueID Field. A Trigger inserts the next DocumentID Sequence value in the UniqueID field in the Documents table. This is how a ‘Autonumber’ field is replicated in Oracle.

Time to review. It's now Monday at 8:10 AM. The migration to Oracle has been completed. You are in Paris, France for 4 months. What are you going to do for the next 3 months 29 days?

I like to joke about taking advantage of employers who are unaware about how easy many things are to accomplish. But the reason we can joke about them is because many people in the software development world would take 4 months to perform an Oracle migration as we have just described. If we are smart, if we plan in advance, if we think before we begin writing programs, we will write programs that are database platform dependent.

By the way, here's the data in Oracle.


DOCUMENTS
Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL
Query Count Rows Insert Row

EDIT	UNIQUEID	PROJECT	FILENAME	REVISION	OWNER	MODIFIED	DESCRIPTION
	110	Sea Scooter	_ZS06 Gear Train Assembly.SLDASM	A-01	pdmwadmin	12-JAN-07	GEAR TRAIN ASSY
	111	Sea Scooter	Gear Shaft Assembly.SLDASM	A-01	pdmwadmin	12-JAN-07	GEAR SHAFT ASSY
	112	Sea Scooter	Stack Gear Assembly.sldasm	A-01	pdmwadmin	12-JAN-07	GEAR STACK ASSY
	113	Sea Scooter	Spur Gear.sldprt	A-01	pdmwadmin	12-JAN-07	SPUR GEAR
	114	Sea Scooter	Shaft.SLDPRT	A-01	pdmwadmin	12-JAN-07	4MM DIA GEAR SHAFT
	115	Sea Scooter	Propeller Gear Shaft Assembly.SLDASM	A-01	pdmwadmin	12-JAN-07	PROP SHAFT ASSY
	116	Sea Scooter	ZS06 Propeller_Shaft 01.SLDPRT	A-01	pdmwadmin	12-JAN-07	PROPELLER SHAFT

FREE, FREE, FREE

Now, we have discussed the availability of VB.NET Express Edition. It's free. Microsoft has an Express Edition for SQL Server. It can be downloaded at:

http://msdn.microsoft.com/vstudio/express/sql/register/default.aspx

And what about Oracle? You can download the free edition of Oracle at:

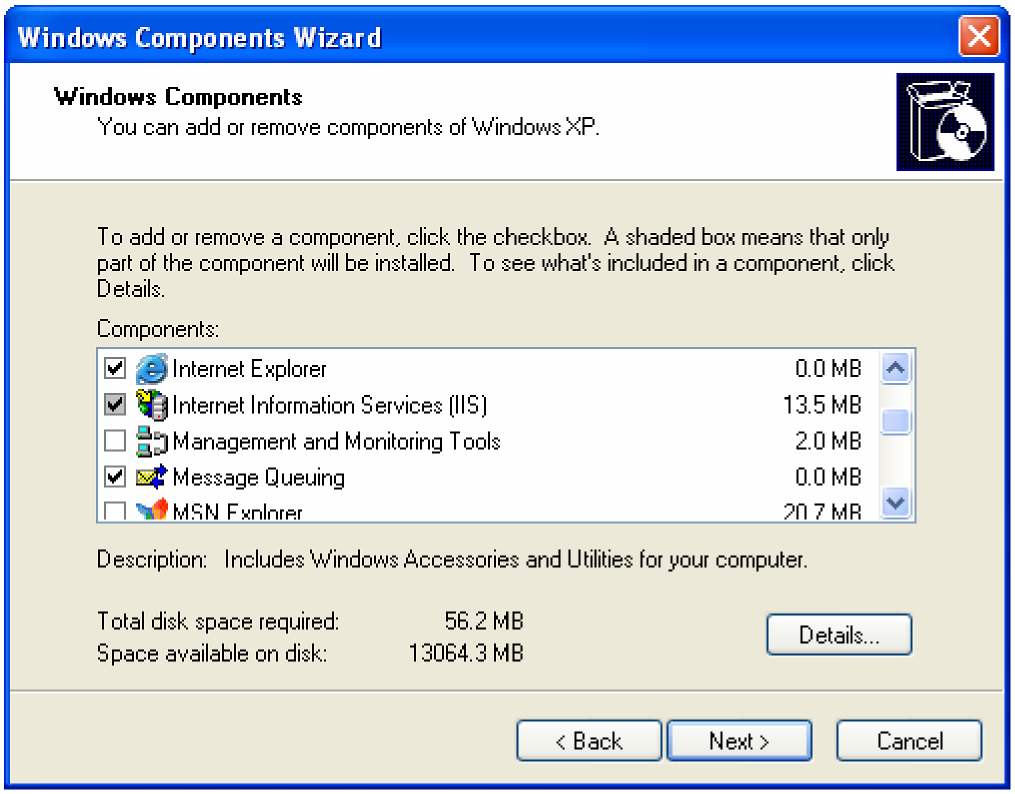
http://www.oracle.com/technology/software/products/database/xe/index.html

And the source code for this class? It's yours for only $995. And we will throw in a free set of steak knives. What a bargain.

Set up Triggers

Triggers. I love triggers. We have learned how to query the Vault. Right? We can query the vault every hour on the hour in order to see what has changed. Or we can use Triggers and have the Vault tell us what has changed.

Step 1: The first step to making use of PDMWorks Triggers is to install MSMQ on the Vault Server.



Windows Components Wizard

Windows Components
You can add or remove components of Windows XP.

To add or remove a component, click the checkbox. A shaded box means that only part of the component will be installed. To see what's included in a component, click Details.

Components:

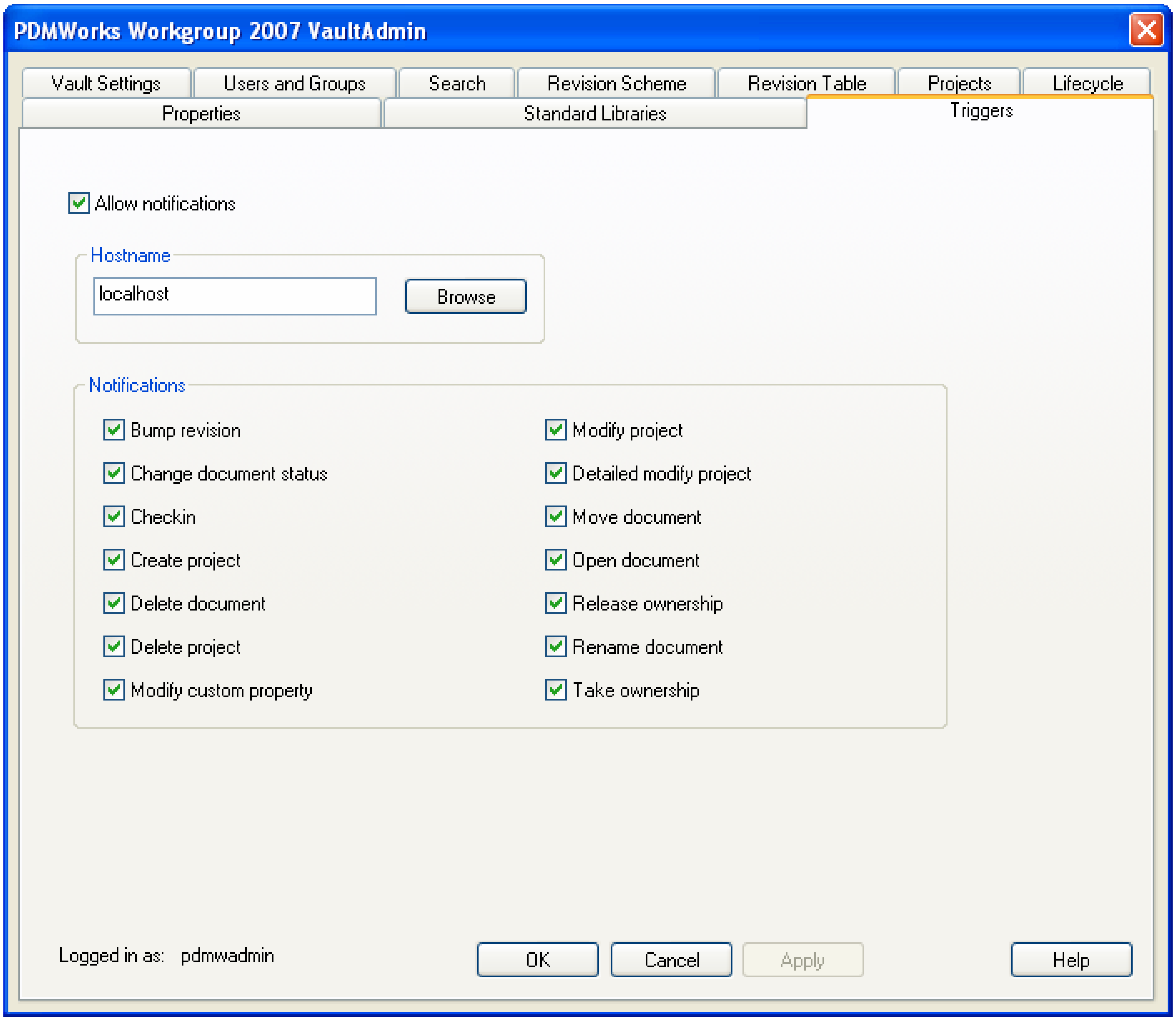
✓	Internet Explorer	0.0 MB
✓	Internet Information Services (IIS)	13.5 MB
☐	Management and Monitoring Tools	2.0 MB
✓	Message Queuing	0.0 MB
☐	MSN Explorer	20.7 MB

Description: Includes Windows Accessories and Utilities for your computer.

Total disk space required: 56.2 MB
Space available on disk: 13064.3 MB

Notice the Message Queuing item in the Add/Remove Windows Components dialog.

Step 2: The next step is to make sure that Triggers are set up in the PDMWorks Vault Admin.



- [✓] Allow notifications

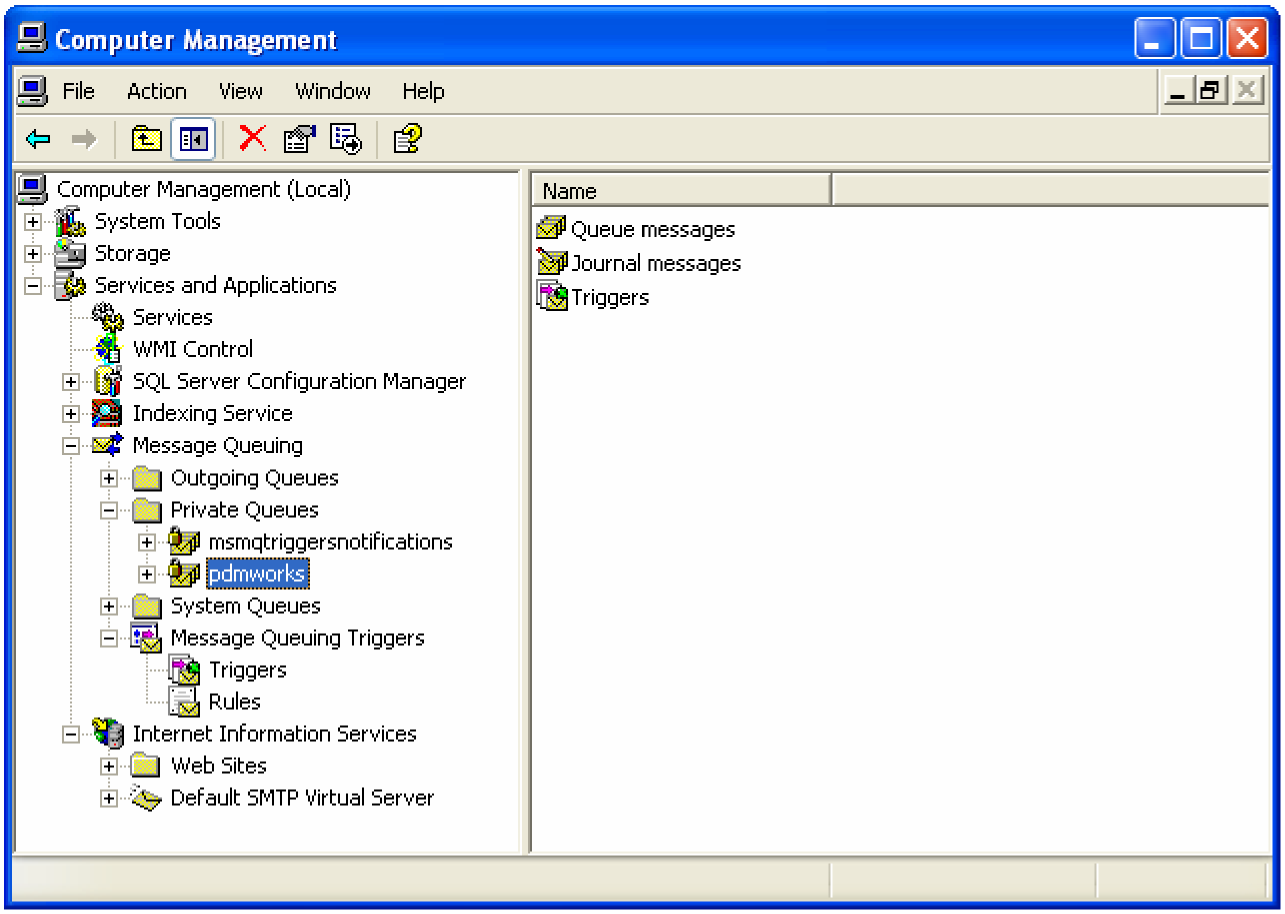
Hostname: localhost

Notifications

- [✓] Bump revision
- [✓] Change document status
- [✓] Checkin
- [✓] Create project
- [✓] Delete document
- [✓] Delete project
- [✓] Modify custom property
- [✓] Modify project
- [✓] Detailed modify project
- [✓] Move document
- [✓] Open document
- [✓] Release ownership
- [✓] Rename document
- [✓] Take ownership

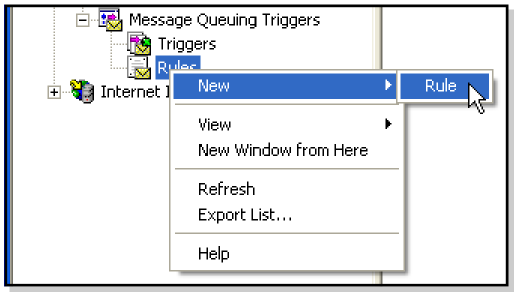
Logged in as: pdmwadmin

Step 3: Create a new MSMQ Queue named “pdmworks”. This is found in the Computer Management which is under the Administrative Tools window.

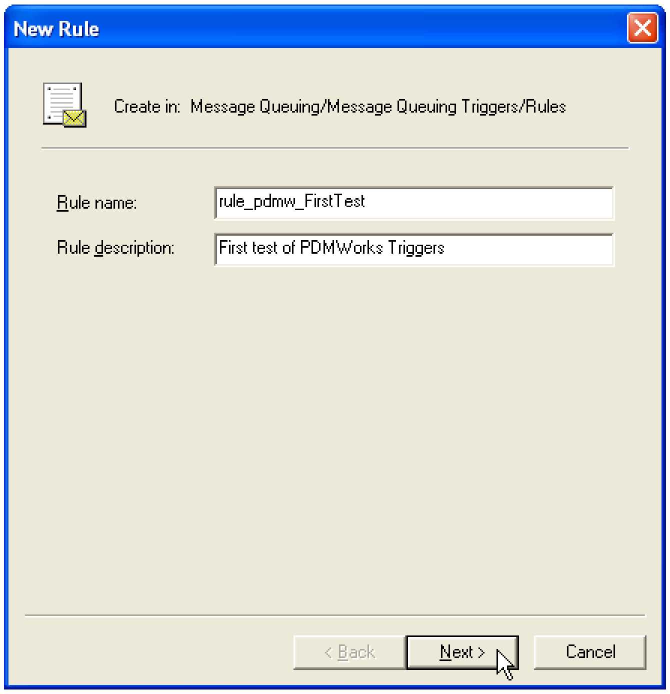


Step 4: Create a new Rule

Now we are going to create a New Rule.



We give the rule a Name and Description.

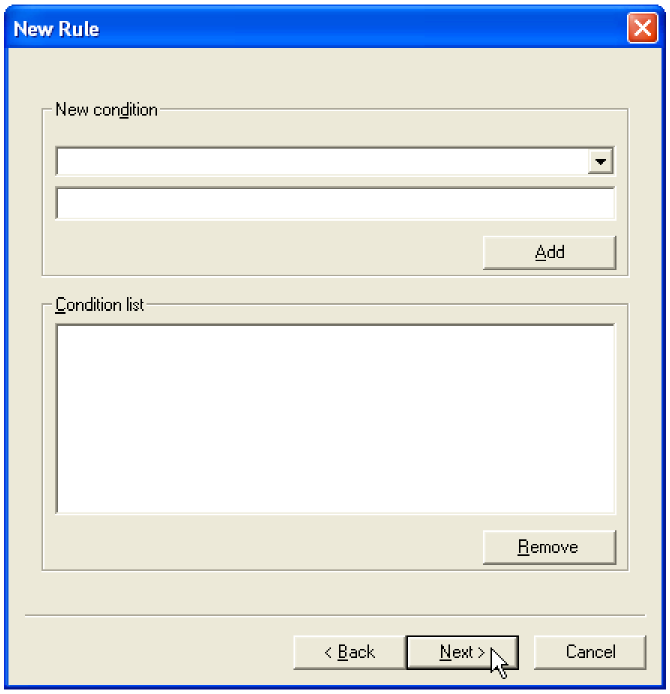


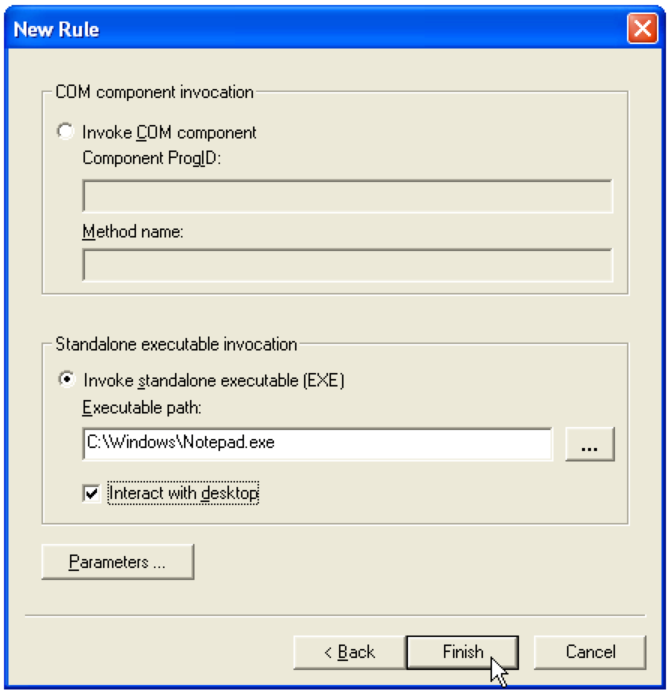
Create in: Message Queuing/Message Queuing Triggers/Rules

Rule name: rule_pdmw_FirstTest

Rule description: First test of PDMWorks Triggers

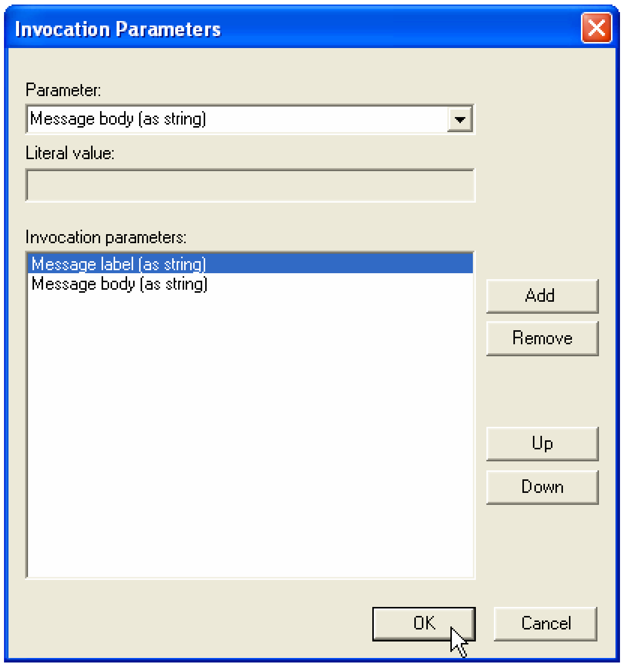
We are not going to apply any Rules, so just click the Next button.





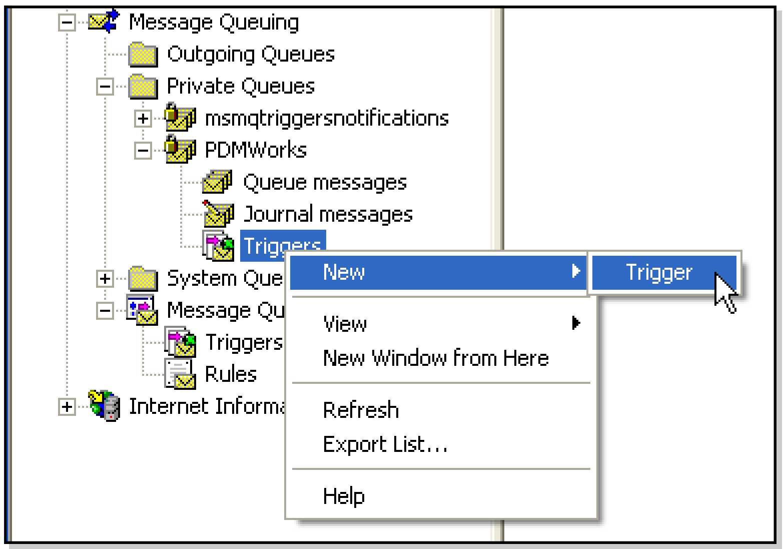
The last thing we do is specify what to do when the Rule is invoked. In this example, we are starting Notepad.

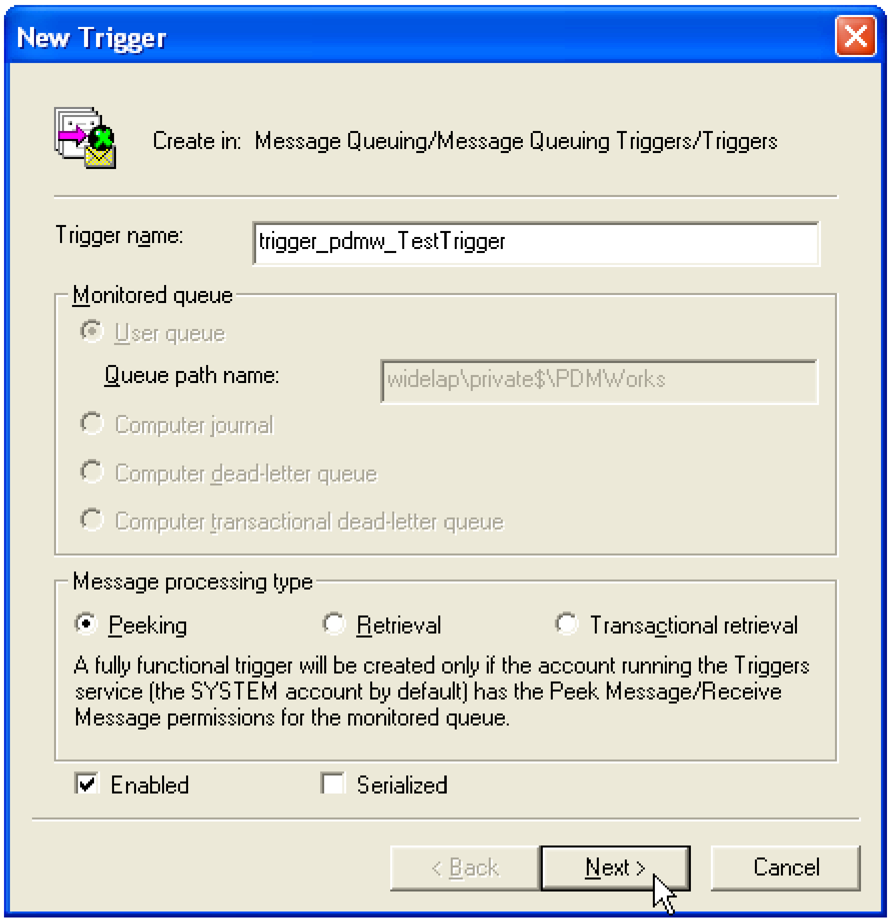
Before clicking the Finish button, click the Parameters button. This is where we specify what information we want sent to the application entered (Notepad in this example).



We are adding the Message Label and the Message Body. Both are requested in String format.

Step 5: Add a new Trigger to the “pdmworks” Queue.




New Trigger

Create in: Message Queuing/Message Queuing Triggers/Triggers

Trigger name: trigger_pdmw_TestTrigger

Monitored queue

[✓ User queue]

Queue path name: widelap\private$\PDMWorks

Computer journal

Computer dead-letter queue

Computer transactional dead-letter queue

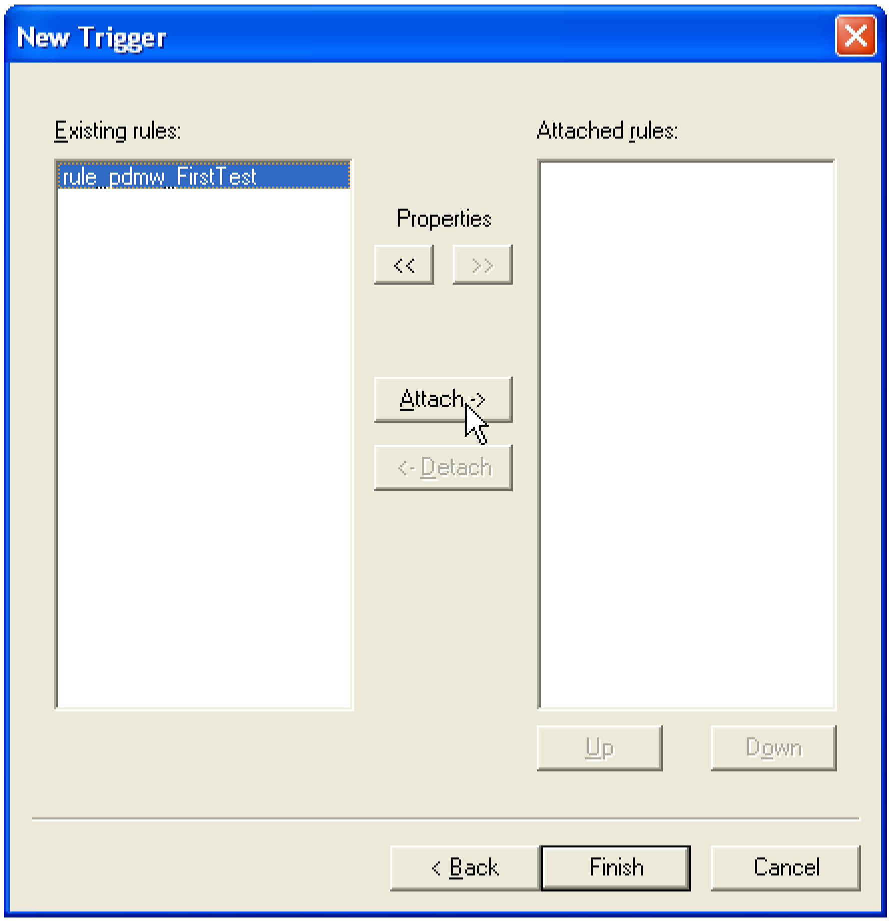
Message processing type

[✓ Peeking] Retrieval Transactional retrieval

A fully functional trigger will be created only if the account running the Triggers service (the SYSTEM account by default) has the Peek Message/Receive Message permissions for the monitored queue.

[✓ Enabled] Serialized

Give the Trigger a Name.


New Trigger

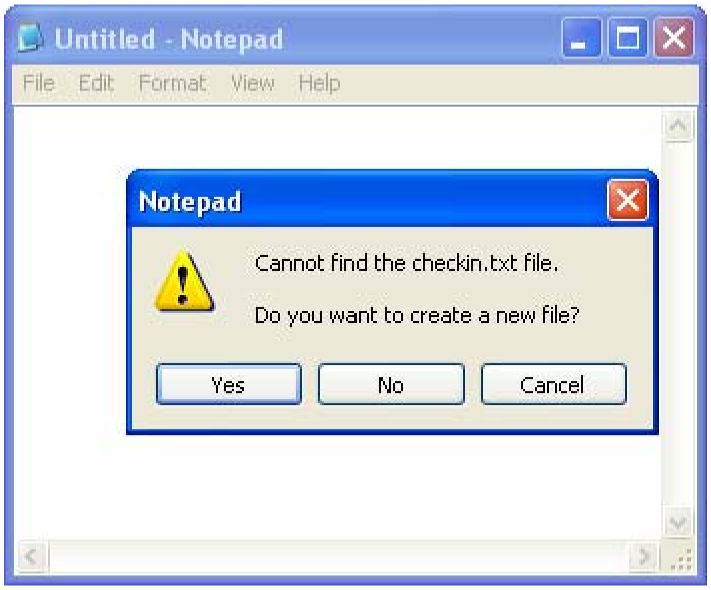
Existing rules: rule_pdmw_FirstTest

Attached rules:

Select a Rule to execute when the Trigger is activated.

Click the Finish Button.

Step 6: Test the Trigger by checking a document into the Vault.

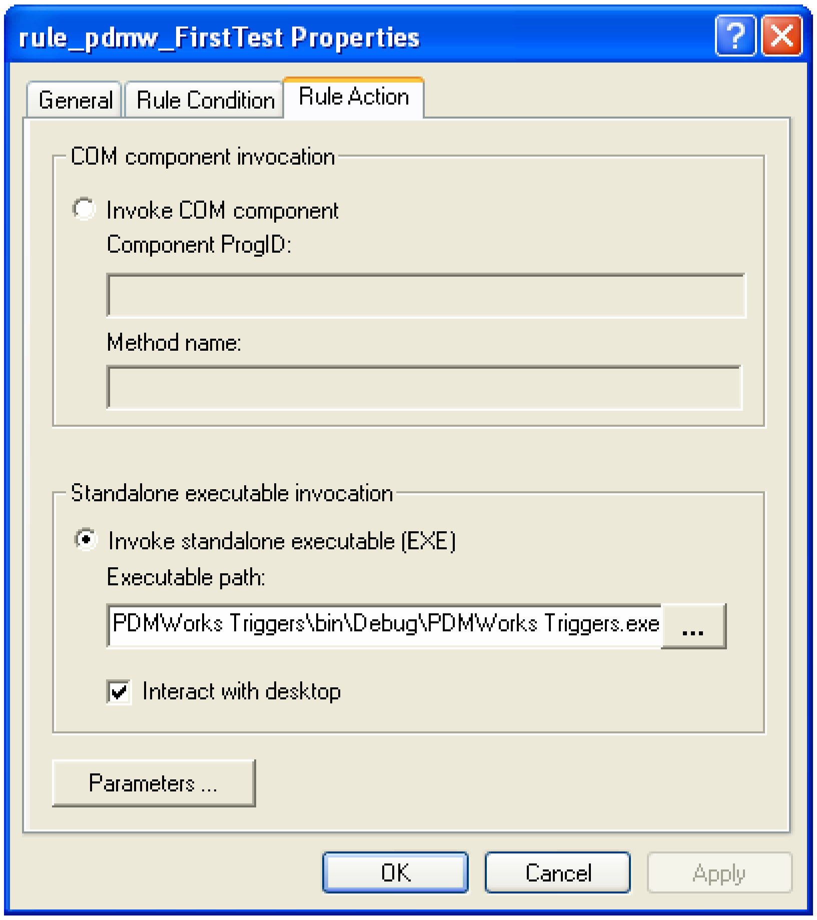


Cannot find the checkin.txt file.

Do you want to create a new file?

Remember, we are using Notepad to test our Trigger right now. When a document is checked into the Vault, we see Notepad and this message. Why would this happen? Because Triggers send command-line information along with the path to the .exe file.

This does confirm that the trigger is working correctly.



Invoke COM component [✓ Invoke standalone executable (EXE)]

Component ProgID:

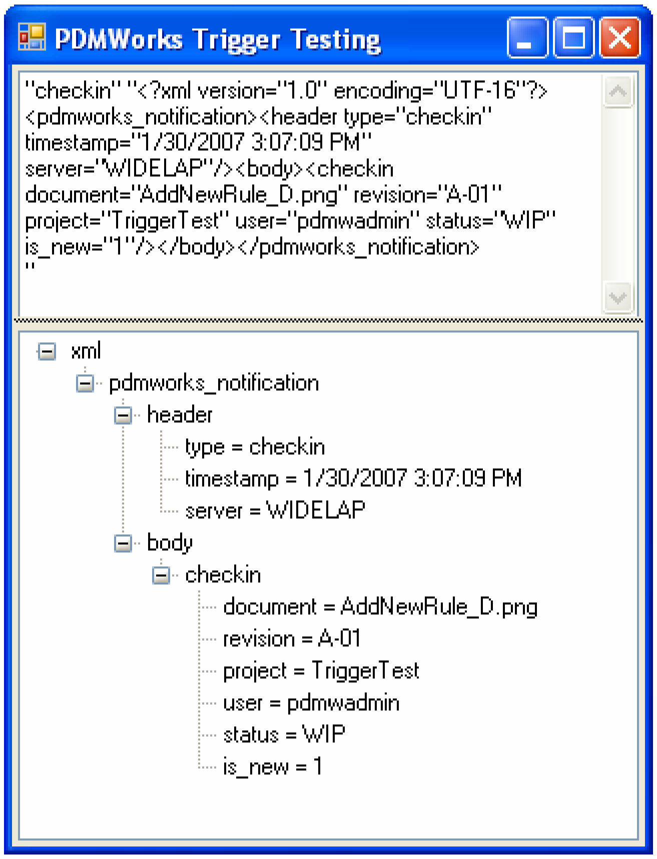
Method name:

Executable path: PDMWorks Triggers\bin\Debug\PDMWorks Triggers.exe

[✓ Interact with desktop]

We are going to modify the Rule that had been pointing to Notepad. Notepad, after all, doesn't help us much.

This is a program written in VB.NET. After setting this up, we will try checking in another document.

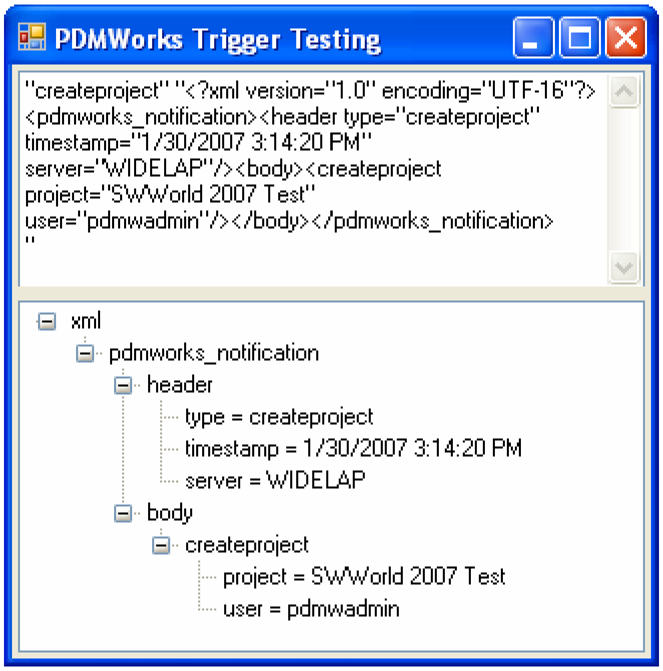


```
"checkin" "<?xml version="1.0" encoding="UTF-16"?>
<pdmworks_notification><header type="checkin"
timestamp="1/30/2007 3:07:09 PM"
server="WIDELAP"/><body><checkin
document="AddNewRule_D.png" revision="A-01"
project="TriggerTest" user="pdmwadmin" status="WIP"
is_new="1"/></body></pdmworks_notification>
"
```

```
xml
  pdmworks_notification
    header
      type = checkin
      timestamp = 1/30/2007 3:07:09 PM
      server = WIDELAP
    body
      checkin
        document = AddNewRule_D.png
        revision = A-01
        project = TriggerTest
        user = pdmwadmin
        status = WIP
        is_new = 1
```

Here we have the PDMWorks Trigger Testing Form showing us the information that is being 'transmitted'. We are shown two views of the data. The first is the raw data as it comes across in the command line. The second is a tree view of the data.

Notice that the first thing shown is "checkin". This is why Notepad was looking for a file named "checkin" when it was being launched by the trigger.



```
"createproject" "<?xml version="1.0" encoding="UTF-16"?>
<pdmworks_notification><header type="createproject"
timestamp="1/30/2007 3:14:20 PM"
server="WIDELAP"/><body><createproject
project="SWWorld 2007 Test"
user="pdmwadmin"/></body></pdmworks_notification>
"
```

```
xml
  pdmworks_notification
    header
      type = createproject
      timestamp = 1/30/2007 3:14:20 PM
      server = WIDELAP
    body
      createproject
        project = SWWorld 2007 Test
        user = pdmwadmin
```

Can we see what has happened here? We can see that a new Project has been created. It's name is "SWWorld 2007 Test". It was created by the user "pdmwadmin".

The PDMWorks Trigger Testing program is a VB.NET Express Edition Program. The code is shown on the following page.

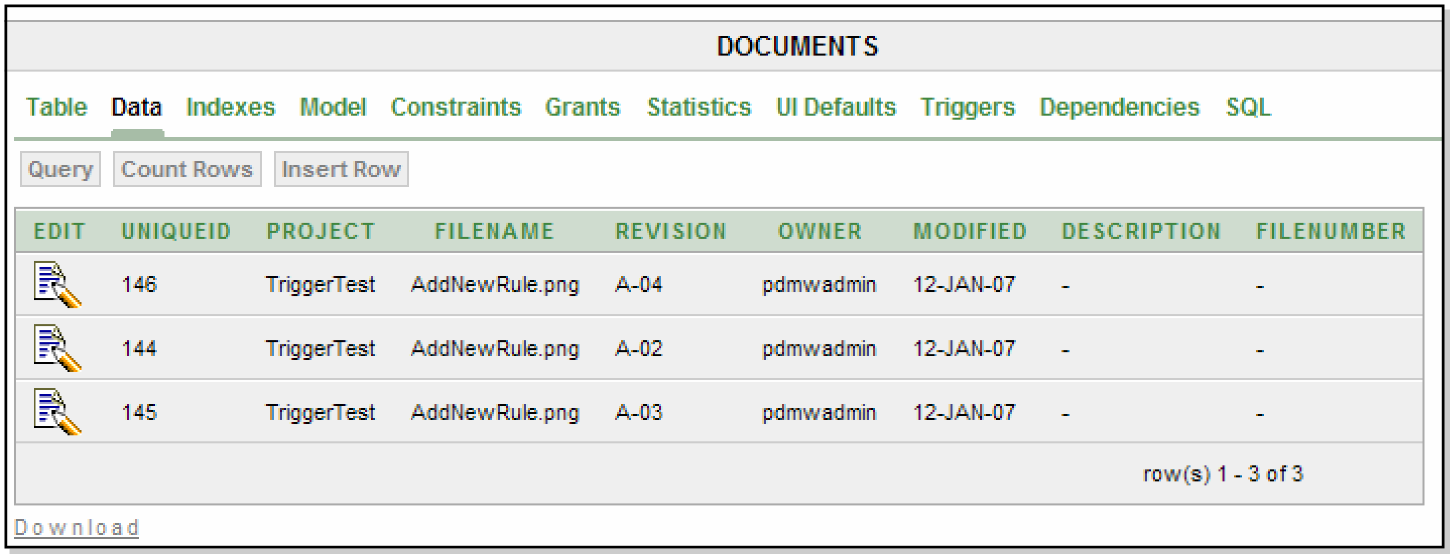
```
Private Sub Form1_Load(ByVal sender As System.Object, _
      ByVal e As System.EventArgs) Handles MyBase.Load
   'Get the Body of the Message
   Dim XSplit() As String
   Dim ReJoin As String
   Dim tvParent1 As TreeNode
   Dim tvParent2 As TreeNode
   Dim tvParent3 As TreeNode
   Dim tvParent4 As TreeNode
   Dim tvChild1 As TreeNode
   Dim myXMLItemAtts As Xml.XmlAttributeCollection
   Dim myXMLItemAtt As Xml.XmlAttribute
   TextBox1.Text = Command()
   XSplit = Split(Command, " ")
   XSplit(0) = ""
   ReJoin = Join(XSplit, " ")
   ReJoin = Mid(ReJoin, 3, Len(ReJoin) - 3)
   Dim myDom As New Xml.XmlDocument
   myDom.InnerXml = ReJoin

   Dim myXML As Xml.XmlNode
   Dim myPDMWxml As Xml.XmlNode
   Dim myHeader As Xml.XmlNode
   Dim myBody As Xml.XmlNode
   Dim myBodyItem As Xml.XmlNode
   myXML = myDom.FirstChild
   tvParent1 = tv1.Nodes.Add(myXML.Name)

   myPDMWxml = myXML.NextSibling
   tvParent2 = tvParent1.Nodes.Add(myPDMWxml.Name)

   myHeader = myPDMWxml.ChildNodes(0)
   tvParent3 = tvParent2.Nodes.Add(myHeader.Name)
   myXMLItemAtts = myHeader.Attributes
   For Each myXMLItemAtt In myXMLItemAtts
      tvParent3.Nodes.Add(myHeader.Name & "." & myXMLItemAtt.Name, _
         myXMLItemAtt.Name & " = " & myXMLItemAtt.Value)
   Next

   myBody = myPDMWxml.ChildNodes(1)
   tvParent4 = tvParent2.Nodes.Add(myBody.Name)
   For Each myBodyItem In myBody.ChildNodes
      tvChild1 = tvParent4.Nodes.Add(myBodyItem.Name)
      myXMLItemAtts = myBodyItem.Attributes
      For Each myXMLItemAtt In myXMLItemAtts
         tvChild1.Nodes.Add(myBodyItem.Name & "." & myXMLItemAtt.Name, _
            myXMLItemAtt.Name & " = " & myXMLItemAtt.Value)
      Next
   Next
End Sub
```

Store Trigger Information in a Database

So, the PDMWorks Trigger Testing program shows us the information we are given any time a trigger is triggered. Let's see if we can do something useful with the data.

```
Sub QueryTheVaultUpdateTheDatabase(ByVal Filename As String)
   Dim myPDMConn As New PDMWConnection
   Dim myPDMWDoc As PDMWDocument
   Dim myDB As New ADODB.Connection
   Dim myRS As New ADODB.Recordset

   myPDMConn.Login("pdmwadmin", "pdmwadmin", "localhost")
   myPDMWDoc = myPDMConn.GetSpecificDocument(Filename)

   myDB.Open("File name=C:\data\SWWorld 2007\PDMWorks\PDMWorksDatabase.udl")
   myRS.Open("Select * from Documents ", myDB, _
      ADODB.CursorTypeEnum.adOpenDynamic, ADODB.LockTypeEnum.adLockOptimistic)

   If Not myPDMWDoc Is Nothing Then
      myRS.AddNew()
      myRS("Project").Value = myPDMWDoc.project
      myRS("FileName").Value = myPDMWDoc.Name
      myRS("Revision").Value = myPDMWDoc.Revision
      myRS("Owner").Value = myPDMWDoc.Owner
      myRS("Modified").Value = myPDMWDoc.DateModified
      myRS("Description").Value = myPDMWDoc.Description
      myRS("FileNumber").Value = myPDMWDoc.Number
      myRS.Update()
   End If
   myRS.Close()
   myDB.Close()
   myPDMConn.Logout()
End Sub
```

Now, before anyone yells at me, I will say it. This is not necessarily the most efficient way to run this code but it does demonstrate the goal. This procedure accepts a Filename parameter. We query the Vault for the provided Filename and write it's information to the database.


DOCUMENTS
Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL
Query Count Rows Insert Row

EDIT	UNIQUEID	PROJECT	FILENAME	REVISION	OWNER	MODIFIED	DESCRIPTION	FILENUMBER
✎	146	TriggerTest	AddNewRule.png	A-04	pdmwadmin	12-JAN-07	-	-
✎	144	TriggerTest	AddNewRule.png	A-02	pdmwadmin	12-JAN-07	-	-
✎	145	TriggerTest	AddNewRule.png	A-03	pdmwadmin	12-JAN-07	-	-

row(s) 1 - 3 of 3

Right now it's writing to Oracle. What do we do if we want the data written to SQL Server? Do we change the code and re-compile? No. We change the UDL File.

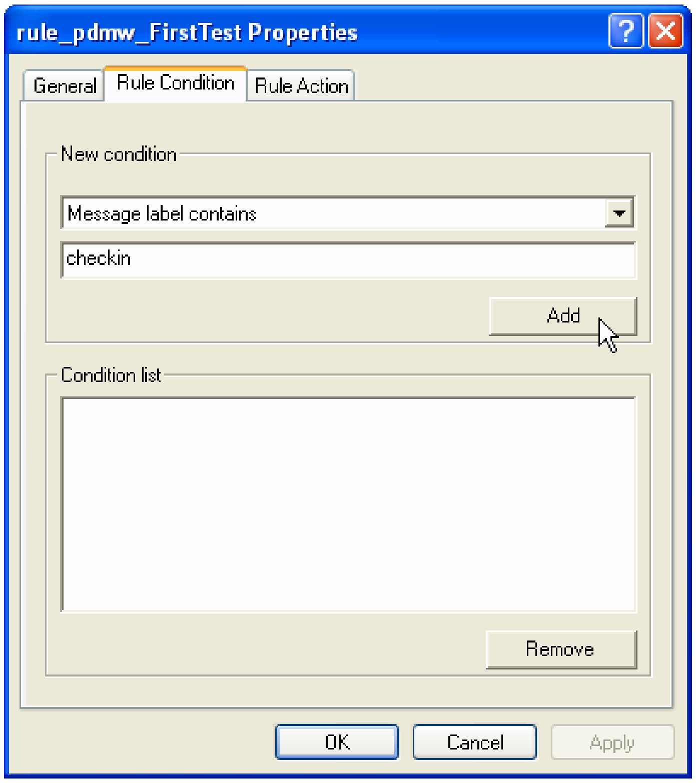
Let's take a look at the code that calls "QueryTheVaultUpdateTheDatabase". It should look a lot like the code on the previous page.

```
1.  Private Sub Form1_Load(ByVal sender As System.Object, _
          ByVal e As System.EventArgs) Handles MyBase.Load
2.     'Get the Body of the Message
3.     Dim XSplit() As String
4.     Dim ReJoin As String
5.     Dim tvParent1 As TreeNode
6.     Dim tvParent2 As TreeNode
7.     Dim tvParent3 As TreeNode
8.     Dim tvParent4 As TreeNode
9.     Dim tvChild1 As TreeNode
10.    Dim myXMLItemAtts As Xml.XmlAttributeCollection
11.    Dim myXMLItemAtt As Xml.XmlAttribute
12.    TextBox1.Text = Command()
13.    XSplit = Split(Command, " ")
14.    XSplit(0) = ""
15.    ReJoin = Join(XSplit, " ")
16.    ReJoin = Mid(ReJoin, 3, Len(ReJoin) - 3)
17.    Dim myDom As New Xml.XmlDocument
18.    myDom.InnerXml = ReJoin
19.    Dim TriggerType As String = ""
20.    Dim TriggerFile As String = ""

21.    Dim myXML As Xml.XmlNode
22.    Dim myPDMWxml As Xml.XmlNode
23.    Dim myHeader As Xml.XmlNode
24.    Dim myBody As Xml.XmlNode
25.    Dim myBodyItem As Xml.XmlNode
26.    myXML = myDom.FirstChild
27.    tvParent1 = tv1.Nodes.Add(myXML.Name)

28.    myPDMWxml = myXML.NextSibling
29.    tvParent2 = tvParent1.Nodes.Add(myPDMWxml.Name)

30.    myHeader = myPDMWxml.ChildNodes(0)
31.    tvParent3 = tvParent2.Nodes.Add(myHeader.Name)
32.    myXMLItemAtts = myHeader.Attributes
33.    For Each myXMLItemAtt In myXMLItemAtts
34.        tvParent3.Nodes.Add(myHeader.Name & "." & myXMLItemAtt.Name, _
               myXMLItemAtt.Name & " = " & myXMLItemAtt.Value)
35.        If myXMLItemAtt.Name = "type" Then
36.            TriggerType = myXMLItemAtt.Value
37.        End If
38.    Next

39.    myBody = myPDMWxml.ChildNodes(1)
40.    tvParent4 = tvParent2.Nodes.Add(myBody.Name)
41.    For Each myBodyItem In myBody.ChildNodes
42.        tvChild1 = tvParent4.Nodes.Add(myBodyItem.Name)
43.        myXMLItemAtts = myBodyItem.Attributes
44.        For Each myXMLItemAtt In myXMLItemAtts
45.            tvChild1.Nodes.Add(myBodyItem.Name & "." & myXMLItemAtt.Name, _
                           myXMLItemAtt.Name & " = " & myXMLItemAtt.Value)
46.            If myXMLItemAtt.Name = "document" Then
47.                TriggerFile = myXMLItemAtt.Value
48.            End If
49.        Next
50.    Next
51.    Select Case TriggerType
52.        Case ""
53.        Case "checkin"
54.            QueryTheVaultUpdateTheDatabase(TriggerFile)
55.    End Select
56. End Sub
```

The new lines of code are shown in bold. The goal is to capture what type of operation has triggered the trigger and to act accordingly in lines 51 through 55.

Our PDMWorks Trigger Testing application is working well. It captures and returns information for each of the PDMWorks Triggers available (See Page 22).

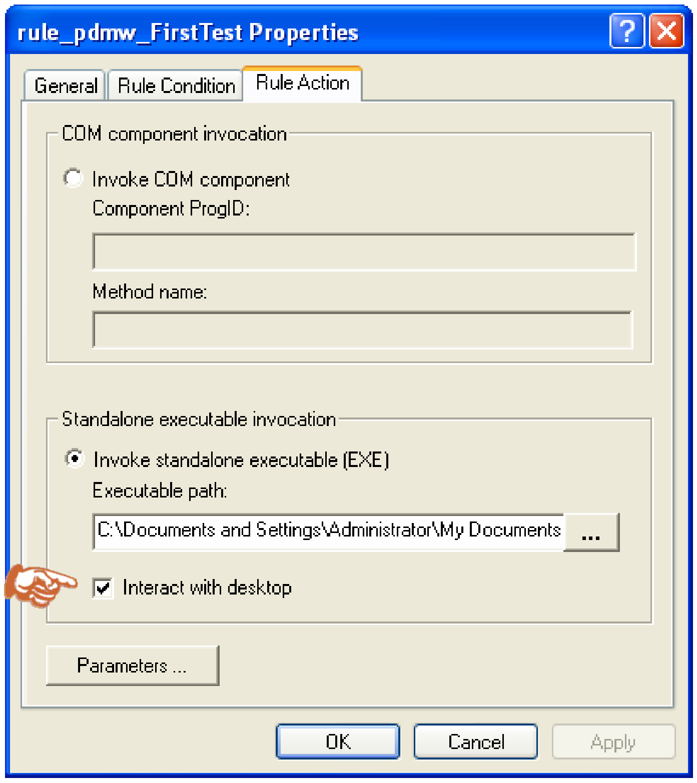
Let's look at two additional things before we wrap things up.



Message label contains

checkin

When we initially set up our Rule, we did not add any conditions. If we select "Message label contains" and enter "checkin" and add this condition, the Rule Action will only fire when the condition is met.

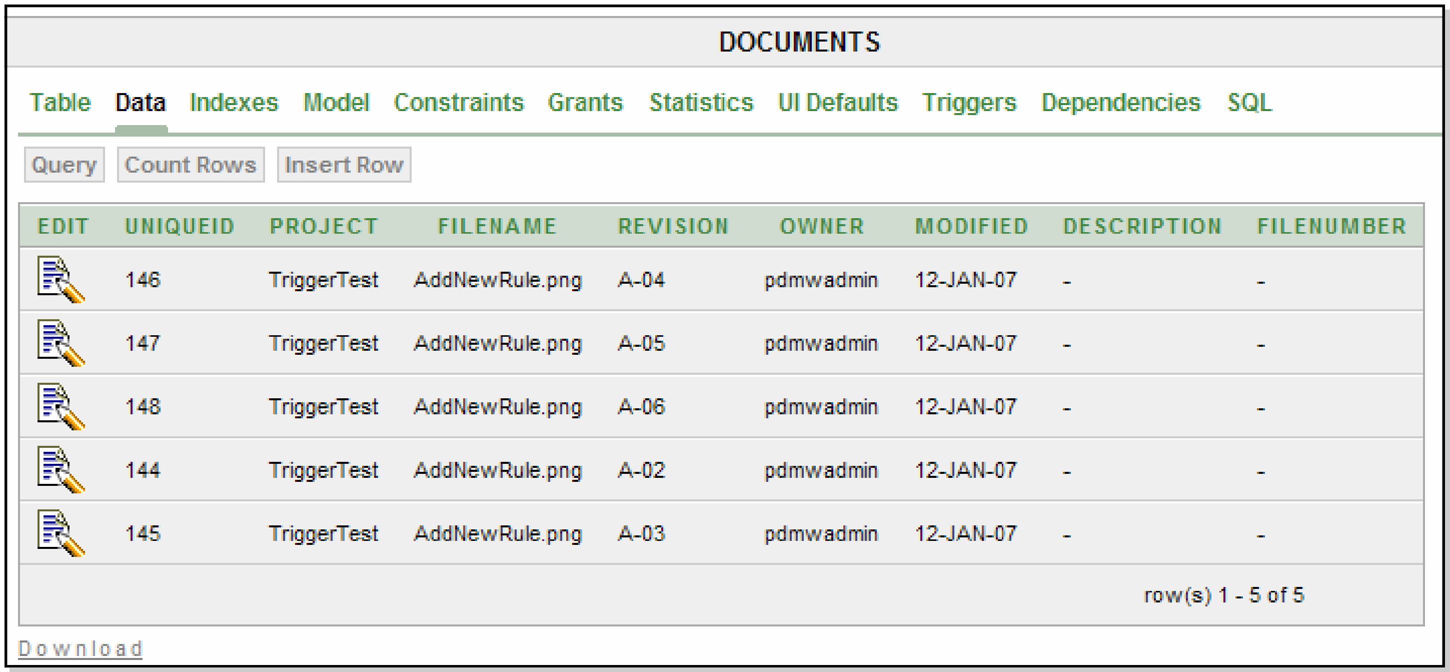


● Invoke standalone executable (EXE)

C:\Documents and Settings\Administrator\My Documents

✓ Interact with desktop

The other thing we want to look at is the "Interact with desktop" setting in the Rule Action tab. This has been selected so far. Let's see what happens if we uncheck it.



DOCUMENTS

Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL
Query Count Rows Insert Row

EDIT	UNIQUEID	PROJECT	FILENAME	REVISION	OWNER	MODIFIED	DESCRIPTION	FILENUMBER
	146	TriggerTest	AddNewRule.png	A-04	pdmwadmin	12-JAN-07	-	-
	147	TriggerTest	AddNewRule.png	A-05	pdmwadmin	12-JAN-07	-	-
	148	TriggerTest	AddNewRule.png	A-06	pdmwadmin	12-JAN-07	-	-
	144	TriggerTest	AddNewRule.png	A-02	pdmwadmin	12-JAN-07	-	-
	145	TriggerTest	AddNewRule.png	A-03	pdmwadmin	12-JAN-07	-	-

row(s) 1 - 5 of 5

As we can see here, document information continues to be added to the database but the User Interface (the Form) is not shown.

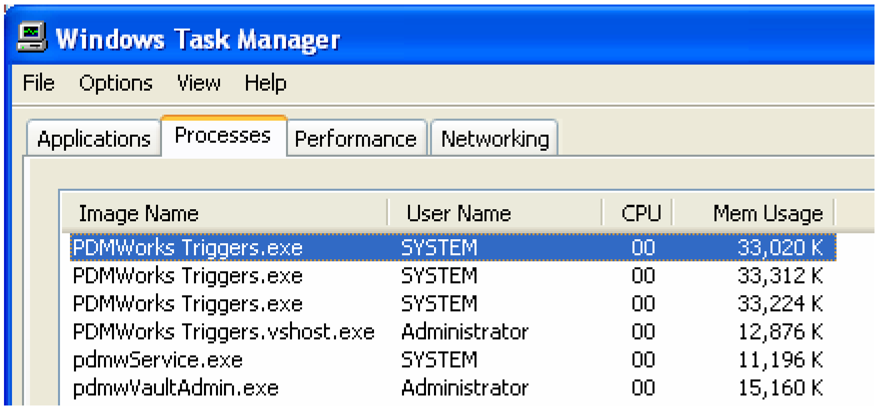


Image Name	User Name	CPU	Mem Usage
PDMWorks Triggers.exe	SYSTEM	00	33,020 K
PDMWorks Triggers.exe	SYSTEM	00	33,312 K
PDMWorks Triggers.exe	SYSTEM	00	33,224 K
PDMWorks Triggers.vshost.exe	Administrator	00	12,876 K
pdmwService.exe	SYSTEM	00	11,196 K
pdmwVaultAdmin.exe	Administrator	00	15,160 K

Even though the User Interface is not being shown, the Application is still running. And not just one Application but an instance for each document that is checked into the Vault.

A Final Review

When we started about 90 minutes ago, we had a list of goals.

- Attach to PDMWorks
- Check In a Document
- SaveAs a Document from the Vault
- Query the PDMWorks Vault
- Export queried data to Excel
- Export queried data to Access
- Export queried data to SQL Server
- Export queried data to Oracle
- Set up Triggers
- Capture Trigger Information
- Store Trigger Information in a Database

You should have everything you need to build a framework for your own specific needs. PDMWorks is great and SolidWorks has given us the ability to capitalize on the data that is in the Vault as well as the interaction each user is having with PDMWorks.

I hope your SolidWorks World experience is a good one and I look forward to associating with you in the future.

Jerry Winters

jerryw@solidworksvba.com
(801) 508-0106