# Using the CONIFERS Dynamic Link Library (DLL) from Visual Basic for Applications in Microsoft Access

Jeff D. Hamann<sup>1</sup> Forest Informatics, Inc. Corvallis, Oregon 97331 jeff.hamann@forestinformatics.com

August 25, 2008

 $^{1}541$ -754-1428

#### 1 Introduction

This tutorial is designed to familiarize you with using the CONIFERS Dynamic Link Library (DLL). Using a Microsoft Access database to store a small forest inventory, the goal is to project the inventory into the future, using simple programming constructs in the Visual Basic for Applications environment. We expend little or no effort on the minutiae of Microsoft Visual Basic or fundamental programming constructs such as for-loops, or if-else-endif code blocks assuming the reader moderately versed in procedural programming (i.e. FOR-TRAN, SAS, VB, R/S+, C, etc.). Our goal is to present the basic procedures for interfacing with the DLL itself, and in order to do so, we use a Microsoft Access database and Visual Basic for Applications, with some basic utilizes specific for this inventory, to read and write data, translate species codes between our internal representation (e.g. "BM") and the functional species in the CONIFERS (i.e. integers), and to call functions within the library itself.

The first tutorial is to familiarize you with installing the Visual Basic for Applications Module (i.e. conifers.bas) that allows you to interface the CONIFERS growth model using Visual Basic. Next, we map our species codes to those available in CONIFERS by using some simple Visual Basic code to map our species table to the CONIFERS function species and use the interface function, set\_species\_map. In the third tutorial, we load data into the CONIFERS data types, by calling a utility function, and impute missing values using the CONIFERS equations using the fill\_in\_missing\_values function. In tutorial four, we call the ex\_update\_total\_summaries function in CONIFERS to compute summary statistics, and in tutorial five, we finally project data using the CONIFERS growth model using the project\_sample function. In tutorial six, we create a table to store our current and projected summaries, add the code to open and delete any current contents of our projected stands. Last, in tutorial seven, we demonstrate projecting multiple stands, updating the database annually, and perform thinning for each projected sample.

Throughout this tutorial, there will be large blocks of "source code". If you are familiar with Visual Basic for Applications (i.e. VBA), you may want to skip over them. If not, try reading over the code to gain an understanding of the syntax. Specifically, a "\_" represents a line continuation. Source code (i.e. variables, VBA functions, CONIFERS functions), and is presented in **courier**, and when new code is to be added to previous code, we include a " $\gg$ " in the left margin.

### 2 First Tutorial: Installing the VBA Interface Module

Open the Visual Basic Editor in Microsoft Access. To get to the Visual Basic Editor, go to the Module View in the Database View and click the "New" button. To install the interface module, import the file named "conifers.bas", using the "File, Import..." menu option from within the Visual Basic Editor. When you have successfully imported the interface module, you should have a screen similar to the one shown in Figure 1. If not, either double click on the module, or use the "View module code" menu item.

The interface module contains user defined constants (i.e. Public Const), user defined data types (e.g. Type PLANT\_RECORD ... End Type), and the interface functions (e.g. subroutines) themselves, used to communicate with the the CONIFERS library. These are the building blocks for all functions associated with CONIFERS.

# 3 Second Tutorial: Setting the Species Map

In this tutorial, we map our species codes to those available in CONIFERS. We create an array of SPECIES\_MAP\_RECORDs which will serve as our functional species map into CONIFERS, and assign values of the SPECIES\_MAP\_RECORD's elements from the species definitions in our inventory database. We use a few basic Visual Basic routines to obtain the sizes we need for our arrays of user defined data types and finally assign the species map using the conifers interface function, set\_species\_map.

The method which CONIFERS handles multiple species is by mapping real, or nominal, species to "functional species", since most growth models only handle a few species. The seven variables for a nominal species are defined in the user defined data type SPECIES\_MAP\_RECORD:

```
Type SPECIES_MAP_RECORD
```

```
idx As Long
fsp_idx As Long
endemic_mortality As Double
max_sdi As Double
browse_damage As Double
mechanical_damage As Double
genetic_worth_h As Double
genetic_worth_d As Double
End Type
```

so that an array of which are used in the function:

```
Public Declare Sub set_species_map Lib _
    "conifers.dll" _
    (ByRef return_code As Long, _
    ByVal model_variant As Long, _
    ByVal n_species As Long, _
    ByRef species_ptr As SPECIES_MAP_RECORD)
```

set\_species\_map to set the CONIFERS functional species to your species map. Once you have set the species map, you need only define the species idx in your array of PLANT\_RECORDs.

A simple technique to initialize the species map is to perform a query from your own species table. We begin with a practice script to select a set of species from the species table, looping over a Microsoft ADO Recordset, determining how large to make our array of SPECIES\_MAP\_RECORD, and finally closing the open Recordset:

着 Microsoft Visual Basic - conifer	k tutorial 1 - [conifers (Code)]		🖉 Microshena 🔛 Distation 🖬 Vitics Commend 🛛 Distation Russed 🔐 Tarla 🥒 Handwijina 🖄 Oranijas Bad 🕤 🗧 🖪 🗙
💰 Eile Edit View Insert D	ebug Run Iook Add-Ins Window Help		X 2 B.X. A voice cuming a supervision of the second state of t
¥ 1 1 × 1 × 1	🕫 🍋 🕨 冒 🔟 🎉 💣 👑 🌾 🎯   In 1, Col 1	n 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	
Project - wrkshop 🗙	(General)	[(Declarations)	
🕒 🐯 wrkshop (conifers_tutor	Option Compare Database Option Explicit Option Base 1		
Cuming for	Public Conte CONIFER As Public Conte AtD0000 As Public Conte SHUB Public Conte SHUB Public Conte NOM STOCKED As Public Conte NOM STOCKED As Public Conte PLANT_TYPES As	Long = 0 Long = 1 Long = 2 Long = 3 Long = 4 Long = 5	
	Public Const CON BAT SIZE	Long = 750 $1/4$ single array for bal/cat $4/$	
	<pre>' variants not currently used Public Const CONIFERS_SUO As Long = 0 Public Const CONIFERS_SUC As Long = 1</pre>		
	<pre>'/* thinning constants */ Public Const Do_J91_MORT Public Const Do_IST_MORT Public Const Do_ISTP_F9 PHING Public Const Do_ISTP_FHINFPONELOU Public Const Do_ISTP_SP_THINFPONELOU Public Const Do_ISTP_SP_THINFROM_BELOU </pre>	<pre>As long = 0 '/* thinning type 0 is add mort */ As long = 1 '/* thin single sp on expf */ As long = 2 '/* thin single sp on expf */ As long = 3 '/* thin to melow based on dbh */ As long = 3 '/* thin from below based on dbh-sp */</pre>	
Properties - conifers X	Type PLOT_RECORD		
Alphabetic Categorized	' FILL IN ONLY THESE VARIABLES plot às Long '	* plot id */	
(Name) conifers	latitude Ås Double '; longitude Ås Double ';	/* plot latitude, in dd MODD10 */ /* plot longitude, in dd MODD10 */	
	elevation As Double '; slope As Double '; aspect As Double '; weren remarter as Double ';	<pre>/ piot elevation above mai, ft */ / siope in trecent? * / / aspect (0 - 50 degrees) */ // meare history conserver * /</pre>	
	mean_annual_precip As Double ';	/* amual precip	
	error As Long '; /* erro	r flag	
	<pre>//* these are temp variables //* used for projections and stat shrub_pet_cover &amp; shouble ', shrub_mean_height &amp;s Double ';</pre>	*/ :* */ /* temp variables */	
	shrub_expf &s Double '; besal ares as Double '; d6 ares As Double '; expf &s Double ';	<pre>/* shrub expansion factor */ * total stand an et. feet */ /* hesal area fointh. */ /* total trees per acre */</pre>	
	hann_wang_XO As Double '; bh_expf As Double '; qmd As Double '; sdi As Double ';	<pre>/* hann &amp; wang XO value mort init */ /* expf in trees he+con over 4.5 ft*/ /* quadratic mean diameter */ /* areand femily index */</pre>	
	<pre>sdimax As Double ';</pre>	/* meximum stand dens index */	
	immediate		

Figure 1: CONIFERS Visual Basic for Applications interface module.

Sub main()

```
' conifers data definition (type)
Dim sp_recs() As conifers.SPECIES_MAP_RECORD
Dim species_rs As New ADODB.Recordset
                                                ' vba object
Dim n_species As Long
' issue an sql statement to open the table (or some subset)
' the variant is SMC_CONIFERS (variant = 1)
species_rs.Open "select * from species where variant = 1", _
   CurrentProject.Connection, adOpenKeyset, adLockOptimistic
' get a count of the species recordsets
species_rs.MoveLast
n_species = species_rs.RecordCount
ReDim sp_recs(n_species)
' move the first record in the record set
' and loop over the access records and initialize the conifers.SPECIES_RECORD
' close the recordset (ado)
species_rs.Close
Set species_rs = Nothing
```

#### End Sub

Just under the notes to yourself about looping over the records, insert the code to extract the values from the recordset (from the database) to the array we ReDim'd. The code requires we go back to the first record in the recordset, and then loop over each record, and assign the values to the elements in our array:

```
sp_recs(i).genetic_worth_h = species_rs("genetic_worth_h").Value
sp_recs(i).genetic_worth_d = species_rs("genetic_worth_d").Value
i = i + 1
species_rs.MoveNext
Loop
```

and after closing the Recordset, calling the set\_species\_map function in the CONIFERS library:

```
' finally, call the conifers shared library function (set_species_map)
Call conifers.set_species_map(return_code, conifers.CONIFERS_SMC, n_species, sp_recs(1))
If (return_code) Then
        Debug.Print "species map not initialized -- stopping"
        Exit Sub
End If
```

so that the final, and complete subroutine looks like,

```
Sub main()
```

```
Dim i As Long
Dim return_code As Long
Dim sp_recs() As conifers.SPECIES_MAP_RECORD ' conifers data definition (type)
Dim species_rs As New ADODB.Recordset 'vba object
Dim n_species As Long
' issue an sql statement to open the table (or some subset)
species_rs.Open "select * from species where variant = 1", _
 CurrentProject.Connection, adOpenKeyset, adLockOptimistic
' get a count of the species recordsets
species_rs.MoveLast
n_species = species_rs.RecordCount
ReDim sp_recs(n_species)
' move the first record in the record set
' and loop over the access records and initialize the conifers.SPECIES_RECORD
species_rs.MoveFirst
i = 1
Do While (Not species_rs.EOF)
    ' print some basic data to make sure you can see it.
    Debug.Print "Variant = "; species_rs("variant").Value; _
      " Species = "; species_rs("idx").Value; " "; species_rs("fsp_idx").Value
```

```
' initialize the elements of the conifers.SPECIES_MAP_RECORD
        sp_recs(i).idx = species_rs("idx").Value
        sp_recs(i).fsp_idx = species_rs("fsp_idx").Value
        sp_recs(i).endemic_mortality = species_rs("endemic_mortality").Value
        sp_recs(i).max_sdi = species_rs("max_sdi").Value
        sp_recs(i).browse_damage = species_rs("browse_damage").Value
        sp_recs(i).mechanical_damage = species_rs("mechanical_damage").Value
        sp_recs(i).genetic_worth_h = species_rs("genetic_worth_h").Value
        sp_recs(i).genetic_worth_d = species_rs("genetic_worth_d").Value
        i = i + 1
        species_rs.MoveNext
    Loop
    ' close the species recordset (ado)
    species_rs.Close
    Set species_rs = Nothing
    ' finally, call the conifers shared library function (set_species_map)
    Call conifers.set_species_map(return_code, conifers.CONIFERS_SMC, n_species, sp_recs(1))
    If (return_code) Then
        Debug.Print "species map not initialized -- stopping"
        Exit Sub
    End If
    Debug.Print n_species; " species initialized in species map"
End Sub
```

which when we run the script, yields:

```
Variant = 1 Species = 0 0
Variant = 1 Species = 1 1
Variant = 1 Species = 2 2
3 species initialized in species map
```

in the Immediate window as shown in Figure 2.

Depending on how your data is structured, you may use this simple technique to load data into CONIFERS user defined data types or you can write any number of several subroutines to meet your needs for the remainder of the tutorial. For convenience we have included a utility module with several helper subroutines for this specific database. Of course, you are free to modify them or create your own from scratch.

🎢 Microsoft Visual Basic - conifers	Lutorial 1 - [tutorial 1. (dob)]	
: 🤻 Elle Edit Yiew Insert Det	bug Bun Tods Addrifs Mindow Heb	× ₽ I ►
	2 ℃ \> = = 볼 (영 홈 량 ※) @   m+cai	
Project - wrkshop	(General) 🚽 Inain	Þ
Wirkshop (conifers_tutorial_ Oddies Conifers Maint_species_lb	<pre>this subroutime loops over the units and performs the following tasks:     l loads the reserves are records from the table     l into the CONIFERS shared liberary </pre>	<b>a</b>
	Output ons	
<pre> tutorial_1 tutorial_2 tutorial_2 </pre>	Dim i As Long Dim return_code As long	
	Dim spress() As confers.SFRCHES_MAP_RECORD ' confers data definition (rype) Dim species rata nous Recorder ' vha object ' vha object Dim rate and nous Recorder ' vha object	
	<pre>' issue an sql statement to open the table (or some subset) species_ts.Open "select " from species where variant = 1", CurrentProject.Connection, adOpenKeyset, adLockOptimistic</pre>	
	<pre>' get a count of the species recordsets species random and and and and and and and and and and</pre>	
Properties - tutorial_1     X	<pre>' move the first record in the record set species_rs.MoveFirst 1 = 1</pre>	
tutorial_1 Module	Do While (Not species_rs_DOF)	
Alphabetic Categorized (Name) tutorial_1	' print some basic data to make sure you can see it. Debug.Print "Variant "", species_rs("Yariant").Value; " Species_rs("idx").Value; " "; species_rs("fsp_idx").Value	
	<pre>' initialize the elements of the conffers.SPCIFS_MAP_RECORD spreec(1).idf sepecies_res("rank").Value spreec(1).endenge: procreatizety and "May").Value spreec(1).endengers: procreatizety appecies res("rank").Value spreece(1).hrowe damage = species res("rank").Value spreece(1).hrowe damage = species res("rank").Value spreece(1).endender = species res("rank").Value spreece(1).endender = species res("rank").Value spreeces(1).endender = species res("rank").Value spreeces(1).endender = species res("rank").Value spreeces(1).endender = species res("rank").Value spreeces res(1).endenter = 0. loop</pre>	
	' close the species recordset (ado) species rs.close Set species_rs = Noching	
	<pre>finally.call the confers shared library function (set_species_map) call confersers provide some providers.confers.confers.confFERS_SMC, n_species, sp_recs(1)) if (return code) Then</pre>	
	Debuy. Frint n_species: " species initialized in species map"	
	End Stab	•
	Immediate Variant = 1 Species = 0 D	X
	Vurtant = 1 Species = 1 1 Vurtant = 1 Species = 2 2 3 species initialized in species map	
		•

Figure 2: Immediate window output

#### 4 Third Tutorial: Loading Data

Let's begin by importing the workshop\_utils.bas module we included to support the tutorial database. In this module, we include functions to load our species table into the CONIFERS library, which we used in the last tutorial, a function to translate our species into CONIFERS functional species, and a function to allocate and assign arrays of CONIFERS data types from the plot and plant records in our database. In short, these functions are the data interface functions between the tutorial database and the modules we present here.

First, create a new module and construct a basic skeleton (simple subroutine) for looping over the **stands**. Your subroutine should have a couple of variables for the number of plots and plant records, as well as variable sized arrays to hold the **n\_plots** plots and **n\_plants** plants. You'll need variables for your database interface (e.g. ADODB), code to open the species recordset, and code to open and loop over the units.

Sub main()

```
Dim return_code As Long
Dim n_plots As Long
Dim n_plants As Long
Dim plots() As conifers.PLOT_RECORD
                                          ' conifers data definition (type)
Dim plants() As conifers.PLANT_RECORD
                                          ' conifers data definition (type)
' load the plot and plants for stand 1
Dim sql_str As String
                                               ' vba object
Dim stands_rs As New ADODB.Recordset
Dim species_rs As New ADODB.Recordset
                                                ' vba object
Dim conn As ADODB.Connection
' open the current connection
Set conn = CurrentProject.Connection
'initialize the species in the shared library
Set species_rs = New ADODB.Recordset
sql_str = "species"
species_rs.Open sql_str, conn, adOpenKeyset, adLockOptimistic, adCmdTableDirect
species_rs.Index = "sp_code"
' issue an sql statement to open the table (or some subset)
sql_str = "select * from unit"
stands_rs.Open sql_str, CurrentProject.Connection, adOpenKeyset, adLockOptimistic
Do While (Not stands_rs.EOF)
    ' initialize the species map in the library from the species table
    Call init_species_lib.load_species(stands_rs("variant").Value)
```

```
' grow the sample
' print summaries
' write out the results to the database
stands_rs.MoveNext
Loop
species_rs.Close
Set species_rs = Nothing
```

#### End Sub

Make sure you're code is working correctly by examining the output in the Immediate window. Once you have verified your code is working correctly, call the utility function load\_stand, to load data from the current stand\_rs record, resize the plant and plot arrays, and finally, assign values from the database to the resized arrays. To make sure that we have loaded the data correctly, add a Debug.Print statement to report the number of plots, and plant loaded from the units, as well as the name of the tree and plots able where the name in the units table represents the file name for the original conifers file.

Sub main()

```
Dim i As Long
Dim return_code As Long
Dim age As Long
Dim n_plots As Long
Dim n_plants As Long
Dim plots() As conifers.PLOT_RECORD
                                          ' conifers data definition (type)
Dim plants() As conifers.PLANT_RECORD
                                          ' conifers data definition (type)
Dim summary_rec As conifers.SUMMARY_RECORD
                                               ' conifers data definition (type)
' load the plot and plants for stand 1
Dim sql_str As String
Dim stands_rs As New ADODB.Recordset
                                               ' vba object
Dim species_rs As New ADODB.Recordset
                                                ' vba object
Dim conn As ADODB.Connection
' open the current connection
Set conn = CurrentProject.Connection
'initialize the species in the shared library
Set species_rs = New ADODB.Recordset
sql_str = "species"
```

```
species_rs.Open sql_str, conn, adOpenKeyset, adLockOptimistic, adCmdTableDirect
species_rs.Index = "sp_code"
' issue an sql statement to open the table (or some subset)
sql_str = "select * from unit"
stands_rs.Open sql_str, CurrentProject.Connection, adOpenKeyset, adLockOptimistic
Do While (Not stands_rs.EOF)
    ' initialize the species map in the library from the species table
    Call workshop_utils.load_species(stands_rs("variant").Value)
    ' load the data for this database...
    Call workshop_utils.load_stand(species_rs, stands_rs, _
                                    n_plots, plots, n_plants, plants)
    ' print out the results to the intermediate window
    Debug.Print "loaded "; _
                n_plots; " plots and "; n_plants; " plants from # "; _
                stands_rs("Stand").Value; _
                "named "; stands_rs("name").Value; _
                " return_code = "; return_code
    ' grow the sample
    ' use the conifers summary function...
    ' you may not want to use this one for your particular project
    ' write out the results to the database
    stands_rs.MoveNext
Loop
species_rs.Close
Set species_rs = Nothing
```

```
End Sub
```

Now that we have successfully loaded data, let's call the CONIFERS library function to impute missing values called fill\_in\_missing\_values, by placing the function call between the load\_data function call and the Debug.Print statement.

Sub main()

Dim i As Long Dim return\_code As Long Dim age As Long

×	×		F	1		×	•
7							
	tion for hely						
	vpe a quest						
	Ê						
			main		lite		
			F		e") .Va.		
		۵ چ			s.s.		
		3			, tenda t		
					i i i i i i i i i i i i i i i i i i i		
					n (typo (typo dloocký Liues fr		
		? •			finitio fisitio 		
		-			ista de ista de bject (Openke values. Jants piants		
		<b>)</b>			ificers d ficers d some s some s ton, ad at ton, ad at ton, ad at ton, at ton to ton to to to to to to to to to to to to to t		
					- con. - con. le (or onnect onnect - value - value - ve ant to fi		A A A A A A A A A A A A A A A A A A A
					0 0RD mmd 1 set 1		6 8175 6 8150 6 8150 6 8150 6 8175 9 8 90 10 ut
		20, Col 7			T_RECOF ANT_RECOF for set Records to pen t tent ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) )		from di from di from di from di from di from di from di
		Ś			Dng Errs.PLO Frank Starts FL Jants FL Jacob Starts FL Jacob Starts FL L.PLOSE L.PLOSE L.PLOSE FL L.PLOSE L.PLOSE L.PLOSE L.PLOSE L.PLOSE L.FLO		plants plants plants plants fants iants fr
	ow <u>H</u> elp	*		abase	<pre>As Lc long s long s long s conite s conite s conite s conite s conite as New as New New As New As New As New As New As New As New As New As New</pre>		ed 34 1 35 1 34 1 55 1 55 1 5 9 1 1 5 9 1 1 5 9 1 1 5 9 1 1 5 9 1 1 5 9 1 1 5 1 1 5 1 5 1 5 1 5 5 1 5 5 1 5 5 1 5
(Lode)	ns <u>Wi</u> nd.	<b>1</b> 33		re Date 1 cit	introduction intro		tializ. traliz. trand trand trand trand sand sand sand
ICONAL_Z	il-bb <u>A</u> si	2		n Compa 1 Base 1 Expli			tes ini 10 plo 12 plo 13 plo 13 plo 15 plot 2 plot
191 - 1 - 1 CC	Run Ioo	-	eneral)	Option Option Option	Sector Control	■ diate	adeed adeed adeed adeed adeed adeed adeed a
ers_tutor	Debug	5	X I	l di la l			~ 4 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
sic - conire	Insert			irs_tutori			
Visual Ba	dit <u>Vi</u> ew	~	shop	op (conifi dules	conference of the second secon		
Microsoft	e Ele	*	ject - wrk	© wrksh ∭ 0	A state at the state of the sta		
۵		<u>۳</u>	E.	🐸 🗋 📩	▼ 읽 현 경 🔜		

Figure 3: Loaded unit data from the database in the Immediate window.

Dim n\_plots As Long Dim n\_plants As Long Dim plots() As conifers.PLOT\_RECORD ' conifers data definition (type) Dim plants() As conifers.PLANT\_RECORD ' conifers data definition (type) Dim summary\_rec As conifers.SUMMARY\_RECORD ' conifers data definition (type) ' load the plot and plants for stand 1 Dim sql\_str As String Dim stands\_rs As New ADODB.Recordset ' vba object ' vba object Dim species\_rs As New ADODB.Recordset Dim conn As ADODB.Connection ' open the current connection Set conn = CurrentProject.Connection 'initialize the species in the shared library Set species\_rs = New ADODB.Recordset sql\_str = "species" species\_rs.Open sql\_str, conn, adOpenKeyset, adLockOptimistic, adCmdTableDirect species\_rs.Index = "sp\_code" ' issue an sql statement to open the table (or some subset) sql\_str = "select \* from unit" stands\_rs.Open sql\_str, CurrentProject.Connection, adOpenKeyset, adLockOptimistic Do While (Not stands\_rs.EOF) ' initialize the species map in the library from the species table Call workshop\_utils.load\_species(stands\_rs("variant").Value) ' load the data for this database... Call workshop\_utils.load\_stand(species\_rs, stands\_rs, \_ n\_plots, plots, n\_plants, plants) ' call the CONIFERS library function to fill in any missing values 'Call conifers.ex\_fill\_in\_missing\_values(return\_code, \_ stands\_rs("variant").Value, n\_plants, plants(1), \_ n\_plots, plots(1), \_ 11.78, 5.6, 40#) ' print out the results to the intermediate window Debug.Print "loaded "; \_ n\_plots; " plots and "; n\_plants; " plants from # "; \_ stands\_rs("Stand").Value; \_

```
"named "; stands_rs("name").Value; _
            " return_code = "; return_code
' grow the sample
' write out the results to the database
    stands_rs.MoveNext
Loop
species_rs.Close
Set species_rs = Nothing
```

End Sub

Finally, let's put a breakpoint at the Debug.Print statement so we can examine individual attributes of the plant records to verify that the values were filled in as we intended. Run the macro, and when the breakpoint to reached, Visual Basic for Applications will stop, allowing you to add variables to the watch window (Figure 5). To add variables to the watch window, right-click on the variable and using the pull-down-menu, select the "Add to Watch..." menu item.

## 5 Fourth Tutorial: Computing Summaries

We could construct our own code to loop over plant and plot records and create any statistics we need by using some simple programming constructs. While this may be what you ultimately do for your inventory, in this tutorial, we call the ex\_update\_total\_summaries function in CONIFERS to compute summary statistics for an array of plot and plant records. The ex\_update\_total\_summaries function fills in values in a a SUMMARY\_RECORD.

Now that you have the code skeleton for your simple application, you can either add directly to, or add to a copy of, the application we developed in the previous tutorial, by simply adding a CONIFERS SUMMARY\_RECORD data type and the call to ex\_update\_total\_summaries.

```
Sub main()
```

```
Dim i As Long

Dim return_code As Long

Dim age As Long

Dim n_plots As Long

Dim n_plants As Long

Dim plots() As conifers.PLOT_RECORD ' conifers data definition (type)

Dim plants() As conifers.PLANT_RECORD ' conifers data definition (type)

Dim summary_rec As conifers.SUMMARY_RECORD ' conifers data definition (type)
```

🚰 Microsoft Visual Basic - conifers	[aboy] 3 (cates) 2 (cates)	×
🚜 Elle Edit Yiew Insert Det	ug Bun Tock dd4ins Window Łeb 7. pre aquestion for help # >	×
M 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	2 ( + = = 보( 정 영 영 ( Presone) ] [보   + = = =   @ 역 년 역 [ 1] 김 김 상 등	
Project - wrkshop	(General) + main	-
		,П
- 행 wrkshop (conifers_tutorial_ 는 습 Modules - 생 confers	Initialise the species in the shared library Set species is = New ADOB.Recorded and in "mean-stark and	•
tutorial_2	species;s:open ad. str, conn, adOpenKeyset, adLockOptimistic, adCmdTableDirect species:s:lndex = "sp_code"	
workshop_utils	' issue an sql statement to open the table (or some subset) sql_str *select * trom unit* stands_rs*.open sql_str, Curtentit*	
	Do While (Not stands rs.EOF)	
	' initialize the species map in the library from the species table Call workshop_ut1s.load_species(stand_rs("variant").Value)	
	' load the data for this database Call workshop_utils.load_stand(species_rs, stands_rs, n_plots, plots, n_plants, plants)	
Properties - tutorial_3	<pre>call the CONFERS library function to fill in any missing values Call conffers.ex_fill_in_missing_value(return_code,</pre>	
Lutonial_3 Module Abhabetic Categorized (Utonial_3) Utonial_3	<pre>' print out the results to the intermediate window Debug.Print "loade "_ plotes " plotes and ", n plants; " plants from # "; _ stand re!("stand").Value; _ "mamed "; return_value; _ " return_code = "; return_code</pre>	
	<pre>' grow the sample ' call grow_sample()</pre>	
	' use the conffers summary function ' you may not want to use this one for your particular project	
	' write out the results to the database ' write_back_to_database()	
	stands_rs.MoveNext Loop	
	speciesrs.Close Set speciesrs Mothing	
	End Sub	
	Immediate E	×
	3 species initialized 3 species initialized 3 species initia and 51 plants from # 4 named df_siSO_Ocv return_code = D	•
	loaded 9 plots and b/ plants Ifom # 5 named of sloven return code = U bodged s plots initialized here # 6 named After our return code = D	
	3 species initialised 10 aded 2 plots and 5 plants from # 7 named difour return_code = 0	
		1



🧖 Microsoft Visual Basic - conifers	ilies tutorial 1 [heeks]- [tutorial 3 (Code)]	
i 🦑 Eile Edit Yiew Insert Del	Debug Bun Tode Addris Mindow Heb	Type a question for help
- N - N - N - N - N - N - N - N - N - N	34   2 ℃   > = = 14   36 🖷 15 ☆ / @   n +e orse ] ] [★] > = = = 1 ◆ 12 [ 1 ] □ 12 (2 ↔ 12 ]	
Project - wrkshop 🗙	X (General) I main	
Section (conifers_tutorial)      Coniens     Section (conifers_tutorial)      Section (coniens)     Section (conie	<pre>unal 'initialize the species in the shared library Set species rs "species" aq_st = "species" (ont, addpenKeyset, adLockOptimistic, adGmdTableDirect species_rs:her = "sp_code"</pre>	q
- 44% tutorial_3 - 44% tutorial_4 - 44% workshop_utik	<ul> <li>issue an sqi statement to open the table (or some subset)</li> <li>aq_atr = "setor = iton unit"</li> <li>action = solven sqi str, CurrentForject.Connection, adDocKoptimistic</li> </ul>	
	Do While (Not stands_rs.EOF)	
	' initialize the species map in the library from the species table Call workshop_utils.load_species(stands_fr("variant").Value)	
	' load the daca for this datahare Call workshop_urls.load_sraudpercie_rs, stenda_rs, n_plote, plante, plant	XI
	<ul> <li>call the CONTERS library function to fill in any missing values</li> <li>call conferes.ex fill_in_missing_values(retur code.</li> <li>call conferes.ex fill_in_missing_values(retur code.</li> <li>call conteres.ex fill_in_missing_values(retur code.</li> <li>call conteres.ex fill_in_missing_values(retur code.</li> </ul>	
Properties - tutorial_3 X	Image: Decision for the state of t	
tutorial_3 Module  Alphabetic   Categorized	trint out the results to the intermediate window     term of a c4520813463055     Dude tudnel_men     term of a c4520813463055     Dude tudnel_men     term of a c4520813463055     Dude tudnel_men     tudnel     tudnel_men     tudnel_men     tudnel_men     tudnel     tudn	
c_menous	Totole Maria and a function frame. And the function of th	
	Teturn code = "; return code     Tode = "; return code     Tode = "; return code     Tode 1 and     Tode1 and     Tode1 and     Tode1 and     Tode1 and	
	'gruthe semple under the semple call gruthe semple to budde throwig main 'call grut semple') call grut semple () Dudde throwig main 'call grut semple () Dudde throwig main throwing	
	' use the conifers summary function ' you may not want to use this one for your particular project	
	' write buck to database to the database to the database to database to but 23 tong tudnid Jamin ' write buck to database() to the database to but 3 tong tudnid Jamin ' write buck to database()	
	-         -         -         -         -         1         Long         tutoriel_3 mein           stands_rs.RoveNext         -         +         +         +         2.3         Double         tutoriel_3 mein	
	Loop <sup>–</sup> transmin 0 Double tutorial 3main – user_code 0 Long tutorial 3main	
	Bpart(S()         E_DMart(S()         FLANT_ECORD         tutatial_3mmin           Employed	
	End Sub	Þ
		1
	Innucliste Inaded 13 niore and 51 niants from <u>#</u> 4 named Af ei50 Dovrzetnow cude = 0	X
	3 species initialized 10 species initialized	



```
' load the plot and plants for stand 1
Dim sql_str As String
Dim stands_rs As New ADODB.Recordset
                                            ' vba object
Dim species_rs As New ADODB.Recordset
                                             'vba object
Dim conn As ADODB.Connection
' open the current connection
Set conn = CurrentProject.Connection
'initialize the species in the shared library
Set species_rs = New ADODB.Recordset
sql_str = "species"
species_rs.Open sql_str, conn, adOpenKeyset, adLockOptimistic, adCmdTableDirect
species_rs.Index = "sp_code"
' issue an sql statement to open the table (or some subset)
sql_str = "select * from unit"
stands_rs.Open sql_str, CurrentProject.Connection, adOpenKeyset, adLockOptimistic
Do While (Not stands_rs.EOF)
    ' initialize the species map in the library from the species table
    Call workshop_utils.load_species(stands_rs("variant").Value)
    ' load the data for this database...
    Call workshop_utils.load_stand(species_rs, stands_rs, _
                                    n_plots, plots, n_plants, plants)
    ' call the CONIFERS library function to fill in any missing values
    'Call conifers.ex_fill_in_missing_values(return_code, _
                                            stands_rs("variant").Value,
                                            n_plants, plants(1), _
                                            n_plots, plots(1), _
                                            11.78, 5.6, 40#)
    ' print out the results to the intermediate window
    Debug.Print "loaded "; _
                n_plots; " plots and "; n_plants; " plants from # "; _
                stands_rs("Stand").Value; _
                "named "; stands_rs("name").Value; _
                " return_code = "; return_code
    ' compute the total stats for the all the species together...
    'Call conifers.ex_update_total_summaries(return_code, _
        n_plots, _
```

```
16
```

```
n_plants, _
        plants(1), _
        summary_rec)
    ' print out the results to the intermediate window
    'age = stands_rs("age").Value
    'Debug.Print "age = "; age; _
                "rc = "; return_code; _
                "tree_expf = "; summary_rec.tree_expf; _
                " bh_expf = "; summary_rec.bh_expf; _
                "basal_area ="; summary_rec.basal_area; _
                "qmd = "; summary_rec.qmd
    ' grow the sample
    ' use the conifers summary function...
    ' you may not want to use this one for your particular project
    ' write out the results to the database
    stands_rs.MoveNext
Loop
species_rs.Close
Set species_rs = Nothing
```

End Sub

Now we've successfully obtained summary statistics for the current set of data, we can then encapsulate this code block within a for-loop to call the project\_sample function and ultimately transfer back into the database, or to some other application.

For brevity, we no longer put the entire subroutine in the tutorial. These modules are available in the installation and can be imported into the Visual Basic editor, using the "File, Import..." menu item as we did in the first few tutorials.

#### 6 Fifth Tutorial: Growing

In this tutorial, we are going to project data using the CONIFERS growth model. To accomplish this, we need to add some variables that are used to control the growth model:

Dim xO As Double Dim endemic\_mortality As Long Dim sdi\_mortality As Long

🕇 Microsoft Visual Basic - conifers	(totalal 1 - [tutalal 4 (Code)]	×
🚜 Elle Edit View Insert De	a Ban Took Addins Window Heb	l x
🛃 💸 + 🖬 🐰 🖿 🐴 🕅	2 연   ▷ Ⅱ 표 보 ( 영 명 영 양 영 영 명 8 년 10 년	
Project - wrkshop	((ceneral) J (ceneral)	
💷 📑 🎦 🔁 🔤	Set species rs New ADODB.Recordset	
	species_rs.open sql_gtr, conn, adOpenKeyset, adLocKOptimistic, adCmdTableDirect species_rs.index = "sp_code"	
tutorial_2 tutorial_3 tutorial_4	' issue an ega statement to open the table (or some subset) sel_st="selst" from unit" from unit," stadds_tsobent supret/ CurrentProject.Connection, adOpenKeyset, adLockOptimistic	
workshop_utils	Do While (Not stand_rs.EOr)	
	' initialize the species map in the library from the species table Call workshop_utils.load_species(stand_rs("variant").Value)	
	' load the data for this database Call workshop_utils.load_stand(species_rs, stands_rs, _ n_plots, plots, n_plants)	
•	<pre>' call the CONTERS library function to fill in any missing values Call confers.ex_fill_in_missing_values(reture_code,</pre>	
Properties - tutorial 4 X tutorial 4 Module  Aphabetic [categorized] (Usama) tutorial 4	' print out the results to the intermediate window Debug.Print "plotes" " plots and "; n_plants; " plants from # "; _ stands ref"?eard", Values: "stands ref"reme", values: _ " return_code = "; return_code	
	<pre>' compute the total stars for the all the species together call constarts.x_update_cotal_summaries(return_code, _ n_plots, _ n_plants)_ n_plants n_aummary_res;</pre>	
	<pre>' print out the results to the intermediate window age = stends refear') view pebugyPrint "age = ", age; _ rece syst = "; return codds' " bh expt = "; summary_ret.the expt; _ " bh expt = "; summary_ret.hexpt: _ "qnd = "; summary_ret.hexpls."</pre>	
	· grow the sample • call grow_sample()	
	· use the confiders summary function	
		×
	loaded 9 plots and 67 plants from # 5 mamed df_2175.cvh return_code = 0 dg = 3 rc = 0 tree_expf = 744.44444444 bh_expf = 0 basal_area = 0 qmd = 0 66 species initialized	1
	loaded 9 plots and 39 plants from # 6 named dfpp_suo return_code = 0 age = 3 ret o tree expf = 433.33333333 bh_expf = 0 basal_area = 0 qmd = 0 3 smeris funtionitys	
	loaded 2 picts and 5 plants from # 7 named df four return.code = 0 age = 3 rc = 0 tree_expf = 250 bh.expf = 250 bhsal_area = 0.629612482864 gmd = 0.6795211151866	
		F



```
Dim use_random_error As Long
Dim random_seed As Long
Dim use_precip_in_hg As Long
Dim use_genetic_gains As Long
Dim age As Long
Dim current_year As Long
Dim n_years As Long
' set the control variables
x0 = 0\# , for mortality
n_years = 20
             ' project for 20 years
'endemic_mortality = 1 can be 0/1 - 0 means no background
' mortality, if 1 then mortality will then depend on morts_ptr.backgound values
'sdi_mortality = 1 can be 0/1 - 0 means no sdi competition
' induced mortality, mortality will then depend on morts_ptr.max_sdi values
endemic_mortality = 1
sdi_mortality = 1
use_random_error = 0
random_seed = 0
use_precip_in_hg = 0
use_genetic_gains = 0
```

where variable  $n_years$  is the number of years that projected and can range from 0 to  $2^{32}$  to facilitate any number of projection lengths. For this workshop will simply project stands forward for twenty (20) years and export the two summaries: one summary at time "age" and twenty years into the future.

```
' grow the sample
Call conifers.project_sample( _
        return_code, _
        n_years, _
        n_plants, _
        plants(1), _
        n_plots, _
        plots(1), _
        x0, _
        endemic_mortality, _
        sdi_mortality, _
        use_random_error, _
        random_seed, _
        use_precip_in_hg, _
           stands_rs("variant").Value, _
            use_genetic_gains, _
        age, _
```

The Debug.Print statement prints out the summaries at the end of the projection and your screen should look something like Figure 7.

Instead of projecting a sample for one 20-year-period, we could have projected for twenty one-year periods (annual) and checked the age iteration or any number of stand conditions that are tracked in the SUMMARY\_RECORD to control the execution path of the subroutine. For example, if we wanted to thin at age 15, we could use an if-then-endif statement to check for age (e.g. age==15) and performed some thinning function using your own code, or one of the thinning functions in the CONIFERS module.

#### 7 Sixth Tutorial: Writing Data back to the Database

This tutorial involves creating a table to store our current and projected summaries, adding the code to open and delete any current contents of our projected summaries table, and a few lines of code to assign field values from our SUMMARY\_RECORD.

Using the design view, create a table definition that includes stand, period, quadratic mean diameter (qmd) and trees per acre (tree\_expf), and the total number of trees per acre over 4.5' tall, the breast height trees per acre (bh\_expf) and the stand basal area (basal\_area) (Figure 8):

Don't forget to change the data types to the appropriate field size (e.g. longer for integers and doubles for floating point numbers). Save the table by naming it projected\_summaries. Don't worry about having a primary key, this is a tutorial. If you want to add a primary key, add it so that it includes the stand and period.

We now need to add the recordset variable for the projected summaries like we did with the species table the other tables:

Dim prj\_sum\_rs As New ADODB.Recordset 'vba object

🚰 Microsoft Visual Basic - conifers	- [4] X - [4] X
: 🏀 File Edit View Insert Di	to be adding which who he adding which he to be adding to
Project - wrkshop X	(General)
🕒 😻 wrkshop (conifers_tutorial.	I rupuic_s: "Poord' and "", n_plants" plants from # ",
conifers	"named", stands rs("name").Value? " return_code = "; return_code
tutorial_2	compute the total stats for the all the species together
tutorial_3	age = stand_rs("age").Value Call conifers.ex update_total_summaries(return_code, _
• 42 tutorial_5 • 42 workshop_utils	n ploces _ n plances _ n
	plants(1),
	Debug.Print Mage = ** age:
	rtee_oyf * " summit_ee type
	n birexpr - ; soumary rec.bir.expr; _ Toal erea": sumary rec.basil area; _ Mowel = 1, sumary rec.basil area; _
	to the second
	(WIL CONTREST DOCJECT SARPLE)
Properties - tutorial_5 ×	protects,
tutorial_5 Module	
Alphabetic Categorized	
(Name) tutorial_5	WJ _ concertity _
	ad_mortalityue
	trandom geneduee precision in
	<pre>Etands Ts("variant").Value,use_genetic gains,</pre>
	age, _ current_year)
	Debug Frint return_code
	use the confirs summary function
	Call ev update_total_summaries(retur_code, _
	Debug Print mage = * age: Tree = * : return code:
	"tree expf = ", summary rec.tree expf: " hh expf = ", summary rec.hh expf:
	"bead_area ="; summer_rec.baad_area; _
	Immediate
	ar = 3 rc = 0 tre_extf = 43.333333333 bh_extf = 0 basal_area = 0 qmd = 0
	age = 23 tc = 0 tree_expf = 419.814215331296 bh_expf = 330.925329642407 basal_area = 58.8819650582354 qmd = 5.71166229519659
	3 species initialized  aded 2 pics and 5 pints from # 7 named df four return_code = 0 age = 3 re tree expf = 250 bh expf = 250 basal area = 0.629612440284266 qmd = 0.6795211151666
	D = 23 rc = 0 tree exit = 240.187739256566 bh exit = 240.1877392565666 bh exit = 240.187739256566 bh exit = 240.1877392565666 bh exit = 240.187739256566 bh exit = 240.187739565666 bh exit = 240.18773956666 bh exit = 240.187739565666 bh exit = 240.1877395656666 bh exit = 240.1877395656666666 bh exit = 240.187739565666666666666666666666666666666666

Figure 7: Projecting the units for a single 20 year projection.

I polyceted_summaries1ahe E polyceted_summaries1ahe Ambler E and Number Des of Number N		
I projected summaries 1 allo and Field Name 1 allo and Provided Number 1 and the scott Number 1 and the scott Number 1 and besiden Number 1 and besiden Number 1 and besiden Number 1 and and 1 and 1		
Administration and the second	Providela.	
period Number be- tree_expf Number hot: basel_area Number bot: qmd Number qui	Indusen	
bh_expf Number nurr basal_area Number tot: qmd Number qur	al mitter of trees per acce	
pasal_area Number too	ther of treases reace wath tht > = 4.5	
	indots and endots for the first section of the sect	
		F
	Field Properties	
Lookup   Lookup	The field description is optional. It helps you describe the field and is also displayed in the schaus bar when you select this field on a form. Press P1 for help or descriptions.	
view. F6 = Switch panes. F1 = Help.		

Figure 8: New table definition for storing the projected summaries.

and add a statement to execute an SQL statement handled by the connection (i.e. conn) to delete all records from projected summaries, and then open the empty recordset:

```
'initialize the species in the shared library
   Set species_rs = New ADODB.Recordset
   sql_str = "species"
    species_rs.Open sql_str, conn, adOpenKeyset, adLockOptimistic, adCmdTableDirect
    species_rs.Index = "sp_code"
    ' delete any records from the projected_summaries table
      conn.Execute "delete * from projected_summaries"
>>
      ' open the project summaries recordset
>>
      Set prj_sum_rs = New ADODB.Recordset
>>
      sql_str = "projected_summaries"
>>
>>
     prj_sum_rs.Open sql_str, conn, adOpenKeyset, adLockOptimistic, adCmdTable
    ' issue an sql statement to open the table (or some subset)
    sql_str = "select * from unit"
    stands_rs.Open sql_str, CurrentProject.Connection, adOpenKeyset, adLockOptimistic
```

Since we are only interested in two time periods for this tutorial (t=0,1), place a update\_total\_summarie and the code to add a new record to the table, function call after the Debug.Print statement immediately after the fill\_in\_missing\_values,

```
Debug.Print "age = "; age; _
                    "rc = "; return_code; _
                    "tree_expf = "; summary_rec.tree_expf; _
                    " bh_expf = "; summary_rec.bh_expf; _
                    "basal_area ="; summary_rec.basal_area; _
                    "qmd = "; summary_rec.qmd
          ' write the data to the database
>>
>>
          prj_sum_rs.AddNew
          prj_sum_rs("stand").Value = stands_rs("stand").Value
>>
          prj_sum_rs("period").Value = 0
>>
          prj_sum_rs("tree_expf").Value = summary_rec.tree_expf
>>
          prj_sum_rs("qmd").Value = summary_rec.qmd
>>
>>
          prj_sum_rs.Update
        ' grow the sample
        ' call grow_sample()
        Call conifers.project_sample( _
                return_code, _
                n_years, _
```

```
n_plants, _
plants(1), _
n_plots, _
plots(1), _
x0, _
endemic_mortality, _
sdi_mortality, _
use_random_error, _
random_seed, _
use_precip_in_hg, _
stands_rs("variant").Value, _
use_genetic_gains, _
age, _
current_year)
Debug.Print return_code
```

and a set of calls to update\_total\_summaries and AddNew/Update code blocks after the post-project\_sample function call:

```
' write out the results to the database
prj_sum_rs.AddNew
prj_sum_rs("stand").Value = stands_rs("stand").Value
prj_sum_rs("period").Value = 1
prj_sum_rs("tree_expf").Value = summary_rec.tree_expf
prj_sum_rs("qmd").Value = summary_rec.qmd
prj_sum_rs.Update
```

The output should be similar to Figure 9.

## 8 Seventh Tutorial: Simulation

Replace the single projection with a for-loop that projects the sample for twenty one-year periods, beginning by changing the value of  $n_years$  from 20 to 1 at the top of the subroutine where we initialize variables,

```
x0 = 0# ' for mortality
>> n_years = 1 ' project for 1 years
'endemic_mortality = 1 can be 0/1 - 0 means no background
' mortality, if 1 then mortality will then depend on morts_ptr.backgound values
'sdi_mortality = 1 can be 0/1 - 0 means no sdi competition
' induced mortality, mortality will then depend on morts_ptr.max_sdi values
endemic_mortality = 1
sdi_mortality = 1
use_random_error = 0
```

I I I I I I I I I I I I I I I I I I I
1     5     1

Figure 9: Projected summaries before and at the end of the projection.

```
random_seed = 0
    use_precip_in_hg = 0
    use_genetic_gains = 0
, and adding the for-loop,
' write the data to the database
        prj_sum_rs.AddNew
        prj_sum_rs("stand").Value = stands_rs("stand").Value
        prj_sum_rs("period").Value = 0
        prj_sum_rs("tree_expf").Value = summary_rec.tree_expf
        prj_sum_rs("qmd").Value = summary_rec.qmd
        prj_sum_rs.Update
        ' grow the sample
>>
          For i = 1 To 20
            Call conifers.project_sample( _
                    return_code, _
                    n_years, _
                    n_plants, _
                    plants(1), _
                    n_plots, _
                    plots(1), _
                    x0.
                    endemic_mortality, _
                    sdi_mortality, _
                    use_random_error, _
                    random_seed, _
                    use_precip_in_hg, _
                       stands_rs("variant").Value, _
                        use_genetic_gains, _
                    age, _
                    current_year)
            ' use the conifers summary function...
            ' you may not want to use this one for your particular project
            Call ex_update_total_summaries(return_code, _
                n_plots, _
                n_plants, _
                plants(1), _
                summary_rec)
            ' write out the results to the database
            prj_sum_rs.AddNew
            prj_sum_rs("stand").Value = stands_rs("stand").Value
              prj_sum_rs("period").Value = i
>>
```

```
prj_sum_rs("tree_expf").Value = summary_rec.tree_expf
prj_sum_rs("qmd").Value = summary_rec.qmd
prj_sum_rs.Update
>> Next i
```

stands\_rs.MoveNext Loop

which yields the following projected\_summaries contents:

Finally, ensure that include a pre-commercial thinning to reduce the density at age 13, we first add an *if-then-elseif* block to the loop where we check the age of the stand:

```
For i = 1 To 20
```

```
' grow the sample for a single year
            Call conifers.project_sample( _
                    return_code, _
                    n_years, _
                    n_plants, _
                    plants(1), _
                    n_plots, _
                    plots(1), _
                    x0,
                    endemic_mortality, _
                    sdi_mortality, _
                    use_random_error, _
                    random_seed, _
                    use_precip_in_hg, _
                       stands_rs("variant").Value, _
                        use_genetic_gains, _
                    age, _
                    current_year)
            ' when the stand is 13 years old, perform a pre-commercial thinning
>>
              If (age = 13) Then
                  Call conifers.thin(return_code, _
>>
>>
                                   n_plants, _
                                   plants(1), _
>>
>>
                                   n_plots, _
                                   plots(1), _
>>
>>
                                   0, _
                                   conifers.DO_EXPF_THIN, _
>>
>>
                                   280#, _
                                   expf_removed, _
>>
                                   ba_removed)
>>
```

🌽 Microsoft Access				× B
Ele Edit View Insert Format	Records Iools Window Help	1	Type a question	for help +
i 🔟 🕂 🛃 🐿 i 🦪 🕰 🍄 i 🐰 🎙	ù 🖭   ማ   😓   24   75 🖄 🔽   🗥   🛌 🤞	🔏 •   📀 💂		
🔳 projected_summaries : Table				
stand period	tree_expf bh_expf basal_area	• dund		
4	21 323.15609/302 266.462/8961 61.61/8230299 22 322 524.5725757 264 931864030 68 0373529643	6.52360495176071 6.86487080033273		
<b>م ا</b>	3 744.444444444			
9	4 742.955555566 33.2666666667 0	0		
5	5 741.469644444 398.4016 0.04693022776	0.146961000307582		
	6 / 39/386/05156 / 39/386/05156 0.97/241153764	0.490850681548826		
5 4 1	600000240000 (04/10/000000/ 04/10/0000/ / / 0000000/ / 0000000/ / / /	01/0002705745745745745745745745745745745745745745		
) LO	9 736.555658845 735.555658845 14.4749228877	1.89948848118702		
0	10 734.084547527 734.084547527 23.2775617095	2.41119318539195		
Q	11 732.616378432 732.616378432 33.7969638524	2.90828415518408		
ۍ ا	12 731.151145676 731.151145676 45.4426036411	3.37570374972836		
ۍ ا	13 280 22.1847873201	3.81140346529789		
50 1	14 279.44 279.44 28.3456822345	4.31256323480891		
0	15 2/8.88112 2/8.88112 34.9/94/U	4./92U5315B24U53		
n u	12 2/0.32333/10 2/0.32333/0 2/0.32333/0 4/1/04030431	0.24001.00040020 6. 67.006801.366607		
	18 277 211177622 277 211177622 56 7803796682	5.07396638433197		
, w	19 276.656755267 276.656755267 62.7041966027	6.44635130940846		
G	20 276.103441757 276.103441757 69.4697641485	6.79200985633601		
0	21 275:551234873 275.551234873 76.0199474664	7.11211723919412		
Q	22 275.000132403 275.000132403 82.3143703955	7.40811446698705		
Q	23 274.450132139 274.450132139 88.3265835376	7.68157542818225		
6	3 433.333333333			
	r 404 010000000 00 00 00 00 00			
0 0	U 441.900044444 394.0UU4 0 444.000 0 444.000 0 0 0 0 0 0 0 0 0			
	7 430 786073428 764 730724804 0 33878576060	0.27 007 0430147 132		
	8 429 907639139 341 01375075 D 86876871375	0.683242697511291		
) (0	9 429.220611638 340.331722749 1.74003861191	0.968198782207578		
9	10 428.539948193 339.651059304 3.00885492721	1.27444307844637		
9	11 427.860646074 338.971757185 4.70417320666	1.59513042980983		
9	12 427.18270256 338.293813671 6.85442448333	1.92741296240107		
9	13 350.230179705 261.341290816 7.67047448822	2.31976186313027		
9	14 349.707497123 260.818608234 10.1741639715	2.6743385530108		
9	15 349.185859906 260.296971018 13.0856194363	3.03597989360611		
٥	16 348.666266964 269.776377076 16.4078874124	3.40300675504263		
	1/ 348.145/13/10 259.256824321 20.135342/304	3.//356531U58838		
	51/2045/52.42 C1001/2021/2022 200551/2014/245 01	4. 1400000/10012 A. 6474760464A6A		
0	20 346 593281272 257,704392383 33,5515557488	4.88675563584362		
	21 346.077872487 257.188983598 38.6619168896	5.24990948570917		
Record: 14 4 84 >	●● ● ●● ●● ●● ●● ●● ●● ●● ●● ●● ●● ●● ●			
Datasheet View				

Figure 10: Projected annual summaries before the thinning.

```
End If
' use the conifers summary function...
' you may not want to use this one for your particular project
Call ex_update_total_summaries(return_code, _
    n_plots, _
    n_plants, _
    plants(1), _
    summary_rec)
' write out the results to the database
prj_sum_rs.AddNew
prj_sum_rs("stand").Value = stands_rs("stand").Value
prj_sum_rs("period").Value = age
prj_sum_rs("tree_expf").Value = summary_rec.tree_expf
prj_sum_rs("bh_expf").Value = summary_rec.bh_expf
prj_sum_rs("qmd").Value = summary_rec.qmd
prj_sum_rs.Update
```

Next i

and add the two variables we need to return the number of plants and the respective basal area removed, we need to add two variables that are updated when we call the thinning function:

Dim expf\_removed As Double Dim ba\_removed As Double

To perform the thinning, call the thin function with the DO\_EXPF\_THIN argument for the type of thinning. To perform for other types of thinning, see the on-line documentation or look in the interface module for the constants defined associated with thinning.

When we run this "scenario", we get the following results in the projected\_summaries table:

#### 9 Final Thoughts

In these seven tutorials, we have focused on moving data into and out of the CONIFERS library. While these are the basic building blocks, there are no restrictions on how you put these commands together for inventory updates, simulation, and optimization.

While the GUI for the CONIFERS growth model may seem have more functionality, it is limited in it's flexibility and utility for simulation or optimization. The advantage of the DLL are that the user is in complete control. The downside, of course, is also, that the user is in complete control. In fact, all of the functions that CONIFERS itself uses are encapsulated in the functions available using this interface. Of course, you are not limited

>>

Microsoft Access			×
Ele Edit Yiew   K +   N 12   4	(nsert Figmank Records Iooks Window Help [고 *** k & 다 빠 하 하 \$\$, 2,4 2,4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	Type a question for h	ion for help
🖬 projected_sun	imaries : Table		
stand	period tree_expf bh_expf basal_area		
	4 21 568.5512/2043 510.958964350 98.9104581029 72 567 529354114 509 937046422 106 839777711		
•	5 3 744.44444444 0 0 0 0 0		
	5 4 742.95555556 33.2666666667 0		
	5 5 741.469644444 398.4016 0.04693022776	0.146661000307562	
	5 6 739.986705156 739.986705156 0.97241153764 5 738.606731746 738.606731746 3.466743030660	0.4006.0014.0014.0014.0014.0014.0014.001	
	5 8 737.029718282 737.029718282 7.6222807484		
	5 735.555658845 735.555658845 14.4749228877	1. 189484814116772	
	5 10 734.084547527 734.084547527 23.2775617095	2.4111931653165	
	5 11 732.616378432 732.616378432 33.7969638524 2 734 4144757 734 4144757 47 444555	2.900341616161616	
	5 13 729 688643364 729 688843364 57 6260659261		
	5 14 728.229465697 728.229465697 69.8512828437	4.13952289761079	
	5 15 726.773006766 726.773006766 81.7603823923	4.54159463308797	
	5 16 725.319460753 725.319460753 93.1186234815	- 486/162/32/386 - 486/162/32/386	
	5 17 723.868821831 723.868821831 103.794040362 5 10 777 47169467 777 47164467 113 776696054	5.12/348/0008 5.27/248/0008	
	5 19 720.976242019 720.976242019 122.913869036		
	5 20 697.488011865 697.488011865 127.344774723	5.787344430268	
	5 21 676.804325009 676.804325009 131.376681122	6.96673401029007	
	5 22 668.547957954 658.547957954 135.070989088 70 544 007470453 544 007470453 130 200705545		
	6 3 433 333333333 041,30/4/0403 041,30/4/0403 130,300/30043	2k1 7000001 /007 0	
	6 4 432.64444444 0 0 0		
	6 5 431.956933333 99.6004 0		
	6 431.270797244 176.713243022 0.07496617662	0 279870408147192	
	6 7 430.586033428 Z64,539/24804 0.338/8520506	<ul> <li>Address 2.338195</li> <li>Address 2.338195</li> </ul>	
	a 4/29/30/2023/39/34/2013/24/2013/24/2012/22/22/22/22/22/22/22/22/22/22/22/22/2		
	6 10 428.539948193 339.651059304 3.00885492721		
	6 11 427.860646074 338.971757185 4.70417320666	(1.556130120803	
	6 12 427.18270256 338.293813671 6.85442448333	1.92741295240107	
	6 13 426.506114932 337,617226043 9.48467688998	2 2.00000001/422	
	6 15 425.156996497 336.268107608 16.2525422697	2.9768096.42	
	6 16 424.484460282 335.595571393 20.3995345995	3.33840761976547	
	6 17 423.813269139 334.924380260 26.0418721475	3702/10600433	
	6 16 4.23.14.34.2U3/9 534.254.55453149 3U.15853050545 6 10 477.47461335 555 5557377 55 553545454		
	6 20 421.807739271 332.918850382 41.60930625885	4.786381.202951	
	6 21 421.14190157 332.253012681 47.8486412539	€ 1385033789652	
Record: IN A	84 > 11 >* 0f 147		
Datasheet View			

Figure 11: Projected annual summaries.

by the functions made available in the DLL. You can create complex scenarios with more complex programming structures and should you wish to obtain additional metrics (e.g. volumes), you can construct subroutines to perform these procedures as well. You are free to use this code and modify it to meet your needs. Enjoy.

# 10 FAQ

#### 10.1 Errors

#### 10.1.1 CONIFERS crashes Access

A defect was found in conifers.bas (i.e. DLL interface module) that was not updated in the repository before release. The DLL interface subroutine for fill\_in\_missing\_values did not include the correct number of arguments (i.e. added the variant argument). The modules, tables, and tutorial documents were updated to correspond with the DLL.