

## VBA Final Project: Automating Technology Transfer Processes

### **Executive Summary**

I recently quit working in BYU's Intellectual Property Services, specifically the Technology Transfer Office. This office focuses on taking professors' disclosed inventions and patenting them in the U.S. and other countries. Furthermore, once the process has begun to patent these technologies, the Technology Transfer Office seeks out companies who would be interested in licensing the technology, and enters into license agreements with these companies.

Because of the nature of the work (patenting), the Technology Transfer does a lot of work with law firms outside of BYU. Currently, the process to approve and pay the law firms' invoices, as well as subsequently billing the licensees to reimburse for the invoices paid is an arduous, manual task. My VBA project eliminates most of the manual effort (excluding data entry) involved in this accounts payable/receivable process by downloading the necessary tables of information from an access database, creating approval summaries for the director and his associates, creating summaries pages for each law firm that is getting paid, and finally creating an invoice for each licensee that has licensed a technology that was paid on.

### **Implementation**

I approached my project in the following stages: getting data from an access database, creating separate summary sheets the director and his two associates, emailing those sheets to the respective individuals, creating summary sheets for each law firm that is going to be paid, and

finally creating a separate invoice for each licensee based on the costs they are responsible, and saving those invoices as PDFs. It's important to note that the excel file that is used for this project comes with two sheets prepopulated: a sheet that lists the current director and associate directors, as well as their IDs and email addresses; the other sheet is an invoice template that is used to populate invoices to the licensees.

Additionally, I included a ribbon feature that allows the user to click an icon on the ribbon which will run the macros in the background (see image below).



Furthermore, I also wrote a sub procedure that clears all of the sheets except for the two prepopulated sheets.

### *Getting Data from Access*

To be honest, this initial step wasn't too terribly difficult. I recorded myself pulling a query from Access, which includes all of the legal bills that will be paid in the next batch. Then I went into the actual code and made a few modifications. Specifically, I made the sub procedure add a new sheet after all the existing sheets in the workbook that contained this data, and then name that sheet "Legal Bills."

The module then begins another sub procedure that pulls a table from Access that contains all of the law firm's information – of specific importance is the law firm code that is unique to each law firm and is used inside the database to help relate each invoice to a law firm. Pulling this table was similar to pulling the query titled "Legal Bills." I recorded myself pulling query and then modified to the sub procedure to add a new sheet after all the existing sheets, which contained the imported table, and then named the sheet "Law Firms." The following is an example of what the code looks like:

```
Sub getInvoiceInfo()

    Application.ScreenUpdating = False

    Sheets().Add After:=ThisWorkbook.Sheets(ThisWorkbook.Sheets.Count)

    With ActiveSheet.ListObjects.Add(SourceType:=0, Source:=Array( _
        "OLEDB;Provider=Microsoft.ACE.OLEDB.12.0;Password=""";User ID=Admin;Data Source=" & ThisWorkbook.Path & "\TT Objects.mdb;" _
        "Mode=Share Deny Write;Extended Properties=""";Jet OLEDB:System database=""";Jet OLEDB:Registry Path=""";Jet OLEDB:Database F
        "="""";Jet OLEDB:Engine Type=5;Jet OLEDB:Database Locking Mode=0;Jet OLEDB:Global Partial Bulk Ops=2;Jet OLEDB:Global Bulk Trans
        "ions=1;Jet OLEDB:New Database Password=""";Jet OLEDB:Create System Database=False;Jet OLEDB:Encrypt Database=False;Jet OLEDB:D
        "t Copy Locale on Compact=False;Jet OLEDB:Compact Without Replica Repair=False;Jet OLEDB:SFP=False;Jet OLEDB:Support Complex Dat
        "=False;Jet OLEDB:Bypass User Info Validation=False;Jet OLEDB:Limited DB Caching=False;Jet OLEDB:Bypass ChoiceField Validation=Fa
        , "se"), Destination:=Range("$A$1")).QueryTable
        .CommandType = xlCmdTable
        .CommandText = Array("Legal Bills")
        .PreserveFormatting = True
        .BackgroundQuery = True
        .SaveData = True
        .AdjustColumnWidth = True
        .PreserveColumnInfo = True
        .SourceDataFile = ThisWorkbook.Path & "\TT Objects.mdb"
        .Refresh BackgroundQuery:=False
    End With
```

### *Summary Sheets for the Director and Associate Directors*

In order for the invoices to get paid they need to be approved by the director and associate directors. I began this step by writing a sub procedure that creates a two-dimensional array based on the information in the prepopulated tab "TTO Employees." This builds an array that contains a name, ID, and email address for the director and two associate directors. Next, the sub procedure creates a worksheet for employee listed in the array and names that worksheet the employees' ID, and puts the employee's first name in cell A1 of the respective sheet. Then

the sub procedure copies the headings from the “Legal Bills” page and pastes them in the second row.

The next step populates each employee’s worksheet with invoices from the “Legal Bills” worksheet if those legal bills fall into the employee’s area of responsibility. It’s important to note here that each invoice is tied to one of the employees via the employee’s ID. The sub procedure evaluates all the cells that have the employee’s ID in a specific column on the “Legal Bills” sheet. This is accomplished with a For loop – if the cell matches the employees ID in the array based on the x value in the For loop, then that row of information is copied and pasted into the respective employee’s sheet. An “If” statement is included in this procedure to define where the copied legal bill information should go. The following is what the code looks like:

```
y = Range("A2", Range("A2").End(xlDown)).Rows.Count

'Establish TTOEmployee array and create new worksheet for employee
For x = 1 To y
    ThisWorkbook.Worksheets("TTO Employee").Activate
    TTOEmployee(x - 1, 0) = Range("A2").Offset(x - 1, 0)
    TTOEmployee(x - 1, 1) = Range("B2").Offset(x - 1, 0)
    Set ws = ThisWorkbook.Sheets.Add(After:=ThisWorkbook.Sheets(ThisWorkbook.Sheets.Count))
    ws.name = TTOEmployee(x - 1, 0)
    ws.Range("A1").Value = UCase(TTOEmployee(x - 1, 1))
    ThisWorkbook.Sheets("Legal Bills").Activate
    Range("A1", Range("A1").End(xlToRight)).Select
    With Selection
        .Copy
    End With
    Sheets(TTOEmployee(x - 1, 0)).Range("A2").PasteSpecial xlPasteValues
Next

y = UBound(TTOEmployee) + 1

'Populate the TTO Employee worksheets with respective line items
For x = 1 To y
    Worksheets("Legal Bills").Activate
    For Each Cell In Range("A2", Range("A2").End(xlDown))
        If Cell.Value = TTOEmployee(x - 1, 0) Then 'Need to consider the RIP TTO IDs
            Range(Cell.Address, Range(Cell.Address).End(xlToRight)).Select
            With Selection
                .Copy
            End With
            If Sheets(TTOEmployee(x - 1, 0)).Range("A3").Value = "" Then
                Sheets(TTOEmployee(x - 1, 0)).Range("A3").PasteSpecial xlPasteValues
            Else
                Sheets(TTOEmployee(x - 1, 0)).Range("A1").End(xlDown).Offset(1, 0).PasteSpecial xlPasteValues
            End If
        End If
    Next
```

These sheets are then formatted based on another sub procedure I wrote. This sub procedure performs the following formatting:

- Autofit the columns
- Define the width of the “Description” column (‘H’). Subsequently left and top vertically aligns the cells, and wraps the text.
- Vertically top aligns the rest of the data
- Centers and underlines the data headings
- Formats the first two rows as title rows to be printed on each page (if multiple pages)
- Formats the amount column as “comma”
- Creates a sum cell for all the amounts listed, bolds this sum, and puts a top border on the cell
- Orients the paper to landscape instead of portrait
- Fits sheet to one page wide
- Automatically prints the sheet in black and white

The code for the formatting looks like the following:

```
Dim lastRow As Long

'Columns("A:G").Select CHECK
Columns("A:G").EntireColumn.AutoFit
Columns("H:H").Select
With Selection
    .ColumnWidth = 54.5
    .HorizontalAlignment = xlLeft
    .WrapText = True
End With
lastRow = Cells(Cells.Rows.Count, "H").End(xlUp).Row
Range("A2:H" & lastRow).Select
'Range(Selection, Selection.End(xlDown)).Select
With Selection
    .VerticalAlignment = xlTop
    .Resize(Selection.Rows.Count, Selection.Columns.Count - 1).Select
    With Selection
        .HorizontalAlignment = xlCenter
    End With
End With
Range("A2:H2").Select
Selection.Font.Underline = xlUnderlineStyleSingle 'Don't know if I need
'Maybe stop here
Application.PrintCommunication = False
With ActiveSheet.PageSetup
    .PrintTitleRows = "$1:$2"
    .Orientation = xlLandscape
    .BlackAndWhite = True
    .FitToPagesWide = 1
    .FitToPagesTall = 0
    .ScaleWithDocHeaderFooter = True
    .AlignMarginsHeaderFooter = True
End With
Application.PrintCommunication = True
```

And the actual worksheet looks like this:

	A	B	C	D	E	F	G	H
1	DAVE							
2	<u>TEmployeeID</u>	<u>Disc#</u>	<u>LawFirm</u>	<u>Invoice#</u>	<u>Docket#</u>	<u>Amount</u>	<u>Speedtype</u>	<u>Description of Transaction</u>
3	DMB	2008-22	ARH	117902	1737-2-019-1	2,500.00	19202572	Prepare and file response to Office Action
4	DMB	2014-022	BHB	19464	0076-014P01	26.50	TT000017	Prepare and file assignment
5	DMB	2014-022	BHB	19465	0076-014P02	26.50	TT000017	Prepare and file assignment
6	DMB	2014-089	BHB	19589	0076-032S01	648.90	19202101	Review and search for ref related to invention disclosure materials
7	DMB	2008-12	KM	1148335	10557.22	-	19202571	
8	DMB	2013-056	KA	30574	2200.2.20	70.00	TT000202	USPTO- late filing fee
9	DMB	2014-003	KA	30575	3074-01	5,300.00	TT000026	prep utility patent application; filing fees and surcharge for late declaration paid to patent office
10	DMB	2014-007	KA	30576	3076-01	5,510.00	TT000028	Prep utility patent application; filing fees;
11	DMB	2011-077	KA	30801	2200.2.12	2,700.00	19202151	prep office action response; discussion w/inventor; file office action response
12	DMB	2012-015	KA	30802	2200.2.14	1,200.00	19202101	Prep office action; examiner interview; file action response; extension fee
13	DMB	2011-049	WNS	895105	17709.39	1,605.00	19202165	review communication, prep letter; review drawings, determine and review and priority claims;
14						19,586.90		

The final step in the process emails the summary sheet to each respective employee. This is done by creating and saving a temporary copy of the worksheet. I should note that the code is written such that Microsoft Outlook has to be open in order to send the email. When Outlook is open, the procedure creates generates the email address to be used (based on the previous two-dimensional array), and a name that consists of the employee's ID, the name of the workbook, and the date – this name is used as the subject line for the email. The sub procedure then sends the email and deletes the temporary copy of the worksheet.

### *Summary Sheets for the Law Firms*


The module then creates a summary sheet for each law firm that will be paid. This follows the same process as the one used to create the employee's summary sheets and the same formatting. One difference for this process is establishing a query based on each unique law firm code that is on the "Legal Bills" worksheet. One other difference with this process is that the summary is only printed out and not emailed to anyone.

### *Create Invoice for Licensee and save as PDF*

The sub procedure I wrote begins this process by pulling a different query of the legal bill information that contains a column of which licensee are responsible for the legal bills. A table of licensee information is also pulled from the database (“Licensee Info”). Similar to the Law Firm summary sheets, this process creates an array of unique licensee values from the newly populated query. This array is used, in conjunction with the “Invoice Template” sheet to create new invoice sheets for each unique licensee value. This process subsequently populates the new invoice sheet with the licenses name and address (by looking up the information on the “Licensee Info” sheet), and copying over line items that each licensee is responsible for. Each sheet is formatted such that the columns will all fit on one page wide, and then each sheet is saved as a PDF. The following code makes up the majority of the sub procedure:

```
For x = 1 To y
    'Make sheets for each licensee & populate address
    wsName = Trim(Left(licensees(x - 1), 10))
    Sheets("Invoice Template").Copy After:=Sheets(Sheets.Count)
    ActiveSheet.name = wsName
    On Error Resume Next
    Worksheets("Licensee Info").Activate
    For Each Cell In Worksheets("Licensee Info").Range("B2", Range("B" & Cells(Rows.Count, "B").End(xlUp).Row))
        If Cell.Value = licensees(x - 1) Then
            Worksheets(wsName).Range("C12").Value = Worksheets("Licensee Info").Range(Cell.Address)
            Worksheets(wsName).Range("C12").Offset(1, 0).Value = Worksheets("Licensee Info").Range(Cell.Offset(0, 2).Address)
            If Cell.Offset(0, 3).Value = "" Then
                Worksheets(wsName).Range("C12").Offset(2, 0).Value = Worksheets("Licensee Info").Range(Cell.Address).Offset(0, 5).Value
            Exit For
        Else
            Worksheets(wsName).Range("C12").Offset(2, 0).Value = Worksheets("Licensee Info").Range(Cell.Offset(0, 3).Address)
            If Cell.Offset(0, 4).Value = "" Then
                Worksheets(wsName).Range("C12").Offset(3, 0).Value = Worksheets("Licensee Info").Range(Cell.Address).Offset(0, 5).Value
            Exit For
        Else
            Worksheets(wsName).Range("C12").Offset(3, 0).Value = Worksheets("Licensee Info").Range(Cell.Offset(0, 4).Address)
            Worksheets(wsName).Range("C12").Offset(3, 0).Value = Worksheets("Licensee Info").Range(Cell.Address).Offset(0, 5).Value
            Exit For
        End If
    End If
End If
Next
z = 1
Worksheets("DB Info").Activate
For Each Cell1 In Worksheets("DB Info").Range("B2", Range("B" & Cells(Rows.Count, "B").End(xlUp).Row))
    If Cell1.Value = licensees(x - 1) Then
        Worksheets(wsName).Activate
        If Worksheets(wsName).Range("A21").Value = "" Then
            Worksheets(wsName).Range("A21").Activate
            ActiveCell.Value = z
        End If
    End If
End For
```

This an example of what the legal invoices looks like when completed:

		<b>BRIGHAM YOUNG UNIVERSITY</b> <b>Technology Transfer Office</b> 3760 HBLL, Provo, Utah 84602-6844 Tax ID Number - 87-0217280 (801) 422-6266		
		April 15, 2015		
		Invoice Number:		
To:	Merck Serono S.A. Merck Patent GmbH Frankfurter Str. 250 Darmstadt, 64293			
<b>Item No.</b>	<b>Description</b>	<b>Original Amount</b>	<b>% Reimbursement</b>	<b>Amount Due</b>
1	BHGL 611598	37.50		37.50
2	BHGL 611598	150.00		150.00
3	BHGL 611598	37.50		37.50
			<b>Total Amount Due:</b>	<b>\$ 225.00</b>

## Lessons Learned

First and foremost, this project taught me the importance of saving your work consistently and especially before running your code. There were several times I ran my code without saving it previously, and excel would unexpectedly quit and I lost all my work I had just added. When I hit a few hard spots that I got stuck on, I learned is how helpful it is to walk away from the project for a few minutes and come back with a fresh set of eyes. One of the more difficult things times I had involved writing code to populate the invoices for the licensees. This was cumbersome in part because the information pulled from access either had wrong information or was formatted poorly. But the more difficult part was trying to write the code the way I had things put together in my head. The lesson I learned here is that it's better to write down on paper what you're trying to accomplish before you start writing the VBA code – this helps clear up the logical flow of information and keeps things straight in your head.



**Assistance**

When I got stuck I looked up some code (the email process establishing an array for unique values) on the internet and used it in my project, but modified it accordingly to fit my needs. I also received some help from a classmate while setting up the ribbon and button to run the code.

**Conclusion**

Overall, this project was a great learning experience in helping me understand how effective VBA can be at solving real business problems, and how beneficial it can be to an organization. This completed project will help save an average of 10 man hours/week at the Technology Transfer Office.