Sub ExeCmdBtn\_Click()

On Error GoTo ErrorBlock

Dim sSQLQry As String, sDML As String

Dim filepath As String

Dim rs As Variant, iCols As Integer, rowsAffected As Long, noOfRows As Long

If AuthenticateUsers() = False Then

MsgBox "ERROR: User not authorized to query this DB."

GoTo Done

End If

Call ExeCmdBtnQC

Call ExeCmdBtnCamp

ThisWorkbook.RefreshAll

ThisWorkbook.Sheets("Dashboard").Select

Done:

MsgBox ("Data Extraction from Admin DB is Complete")

Exit Sub

ErrorBlock:

DisplayError Err.Source, Err.Description, "Sheet7.ExeCmdBtn\_Click", Erl

Resume Done

End Sub

Sub ExeCmdBtnCamp()

On Error GoTo ErrorBlock

Dim sSQLQry As String, sDML As String

Dim filepath As String

Dim rs As Variant, iCols As Integer, rowsAffected As Long, noOfRows As Long

connectToDB conn, Base64DecodeString(configSheet.Cells(1, 2)), "", "", configSheet.Cells(1, 14)

If conn.State = 0 Then

MsgBox "Not connected to DB!!!"

GoTo Done

End If

'Sheet3.Cells(QRYROW, QRYCOL).Select

'Sheet3.Range(Cells(OUTPUTROW, 1), Cells(MAXROW, MAXCOL)).Clear

'Sheet3.Cells.ColumnWidth = 8.38

sSQLQry = "Select AU\_CC\_CONVERSATION.\*,X.Report\_Title from AU\_CC\_CONVERSATION left join (select S\_id\_from\_AU\_post,Report\_Title from HIST\_AU\_ENTITY\_ASW union select S\_id\_from\_AU\_post,Report\_Title from HIST\_AU\_IND\_ASW)X on AU\_CC\_CONVERSATION.CC\_Conv\_Case\_Id=X.S\_id\_from\_AU\_post;"

If InStr(sSQLQry, Chr(10)) > 0 Then

sSQLQry = Replace(sSQLQry, Chr(10), " ")

End If

Debug.Print sSQLQry

ThisWorkbook.Sheets("Campaign Data").Visible = True

ThisWorkbook.Sheets("Campaign Data").Select

Selection.CurrentRegion.Select

Selection.ClearContents

'sDML = UCase(Left(sSQLQry, 6))

'If sDML = "SELECT" Then

Set rs = CreateObject("ADODB.Recordset")

'rs.CursorLocation = adUseClient

rs.Open sSQLQry, conn

If rs.EOF = False Then

For iCols = 0 To rs.Fields.Count - 1

Sheets("Campaign Data").Cells(OUTPUTROW, iCols + 1).Value = rs.Fields(iCols).Name

Next

Sheets("Campaign Data").Range(Sheets("Campaign Data").Cells(OUTPUTROW, 1), Sheets("Campaign Data").Cells(OUTPUTROW, rs.Fields.Count)).Font.Bold = True

'Sheets("Campaign Data").Range("A2").CopyFromRecordset rs

Sheets("Campaign Data").Range(Sheets("Campaign Data").Cells(OUTPUTROW + 1, 1), Sheets("Campaign Data").Cells(OUTPUTROW + 1, 1)).CopyFromRecordset rs

'rowsAffected = rs.RecordCount

rowsAffected = (Sheets("Campaign Data").Cells(Sheets("Campaign Data").Rows.Count, 1).End(xlUp).row) - OUTPUTROW

noOfRows = Sheets("Campaign Data").Cells(Sheets("Campaign Data").Rows.Count, 1).End(xlUp).row

'Sheets("Campaign Data").Range(Sheets("Campaign Data").Cells(OUTPUTROW, 1), Sheets("Campaign Data").Cells(noOfRows, rs.Fields.Count)).Columns.AutoFit

Else

rowsAffected = rs.RecordCount

End If

' ElseIf sDML = "UPDATE" Or sDML = "INSERT" Or sDML = "DELETE" Then

' conn.Execute sSQLQry, rowsAffected

' End If

ThisWorkbook.Sheets("Campaign Data").Select

Selection.CurrentRegion.Select

Dim objTable As ListObject

Set objTable = ActiveSheet.ListObjects.Add(xlSrcRange, Selection, , xlYes)

objTable.Name = "Table1"

If Not rs Is Nothing Then

If rs.State = 1 Then

rs.Close

End If

End If

Set rs = Nothing

disconnectFromDB conn

ThisWorkbook.Sheets("Campaign Data").Visible = False

' If sDML = "SELECT" Then

' MsgBox rowsAffected & " records selected."

' If rowsAffected > 0 Then

' 'Sheet3.Activate

' Sheet3.Cells(QRYROW, 1).Select

' ActiveWindow.ScrollRow = 1

' 'ActiveWindow.Zoom = 60

' End If

'' ElseIf sDML = "UPDATE" Then

' MsgBox rowsAffected & " records updated."

' ElseIf sDML = "INSERT" Then

' MsgBox rowsAffected & " records inserted."

' ElseIf sDML = "DELETE" Then

' MsgBox rowsAffected & " records deleted."

' End If

Done:

If Not rs Is Nothing Then

If rs.State = 1 Then

rs.Close

End If

End If

Set rs = Nothing

disconnectFromDB conn

Exit Sub

ErrorBlock:

DisplayError Err.Source, Err.Description, "Sheet7.ExeCmdBtn\_Click", Erl

Resume Done

End Sub

Sub ExeCmdBtnQC()

On Error GoTo ErrorBlock

Dim sSQLQry As String, sDML As String

Dim filepath As String

Dim rs As Variant, iCols As Integer, rowsAffected As Long, noOfRows As Long

connectToDB conn, Base64DecodeString(configSheet.Cells(1, 2)), "", "", configSheet.Cells(1, 14)

If conn.State = 0 Then

MsgBox "Not connected to DB!!!"

GoTo Done

End If

'Sheet3.Cells(QRYROW, QRYCOL).Select

'Sheet3.Range(Cells(OUTPUTROW, 1), Cells(MAXROW, MAXCOL)).Clear

'Sheet3.Cells.ColumnWidth = 8.38

sSQLQry = "Select \* from AU\_CC\_QC\_Main;"

If InStr(sSQLQry, Chr(10)) > 0 Then

sSQLQry = Replace(sSQLQry, Chr(10), " ")

End If

Debug.Print sSQLQry

ThisWorkbook.Sheets("Quality Data").Visible = True

ThisWorkbook.Sheets("Quality Data").Select

Selection.CurrentRegion.Select

Selection.ClearContents

'sDML = UCase(Left(sSQLQry, 6))

'If sDML = "SELECT" Then

Set rs = CreateObject("ADODB.Recordset")

'rs.CursorLocation = adUseClient

rs.Open sSQLQry, conn

If rs.EOF = False Then

For iCols = 0 To rs.Fields.Count - 1

Sheets("Quality Data").Cells(OUTPUTROW, iCols + 1).Value = rs.Fields(iCols).Name

Next

Sheets("Quality Data").Range(Sheets("Quality Data").Cells(OUTPUTROW, 1), Sheets("Quality Data").Cells(OUTPUTROW, rs.Fields.Count)).Font.Bold = True

'Sheets("Quality Data").Range("A2").CopyFromRecordset rs

Sheets("Quality Data").Range(Sheets("Quality Data").Cells(OUTPUTROW + 1, 1), Sheets("Quality Data").Cells(OUTPUTROW + 1, 1)).CopyFromRecordset rs

'rowsAffected = rs.RecordCount

rowsAffected = (Sheets("Quality Data").Cells(Sheets("Quality Data").Rows.Count, 1).End(xlUp).row) - OUTPUTROW

noOfRows = Sheets("Quality Data").Cells(Sheets("Quality Data").Rows.Count, 1).End(xlUp).row

'Sheets("Quality Data").Range(Sheets("Quality Data").Cells(OUTPUTROW, 1), Sheets("Quality Data").Cells(noOfRows, rs.Fields.Count)).Columns.AutoFit

Else

rowsAffected = rs.RecordCount

End If

' ElseIf sDML = "UPDATE" Or sDML = "INSERT" Or sDML = "DELETE" Then

' conn.Execute sSQLQry, rowsAffected

' End If

ThisWorkbook.Sheets("Quality Data").Select

Selection.CurrentRegion.Select

Dim objTable As ListObject

Set objTable = ActiveSheet.ListObjects.Add(xlSrcRange, Selection, , xlYes)

objTable.Name = "Table3"

If Not rs Is Nothing Then

If rs.State = 1 Then

rs.Close

End If

End If

Set rs = Nothing

disconnectFromDB conn

ThisWorkbook.Sheets("Quality Data").Visible = False

' If sDML = "SELECT" Then

' MsgBox rowsAffected & " records selected."

' If rowsAffected > 0 Then

' 'Sheet3.Activate

' Sheet3.Cells(QRYROW, 1).Select

' ActiveWindow.ScrollRow = 1

' 'ActiveWindow.Zoom = 60

' End If

'' ElseIf sDML = "UPDATE" Then

' MsgBox rowsAffected & " records updated."

' ElseIf sDML = "INSERT" Then

' MsgBox rowsAffected & " records inserted."

' ElseIf sDML = "DELETE" Then

' MsgBox rowsAffected & " records deleted."

' End If

Done:

If Not rs Is Nothing Then

If rs.State = 1 Then

rs.Close

End If

End If

Set rs = Nothing

disconnectFromDB conn

Exit Sub

ErrorBlock:

DisplayError Err.Source, Err.Description, "Sheet7.ExeCmdBtn\_Click", Erl

Resume Done

End Sub

Private Declare Function CoRegisterMessageFilter Lib "OLE32.DLL" (ByVal lFilterIn As Long, ByRef lPreviousFilter) As Long

Sub AgentStatus(Username As String, Pword As String)

Dim cvsApp As Object

Dim cvsConn As Object

Dim cvsSrv As Object

Dim Rep As Object

Dim Info As Object, Log As Object, b As Object

Dim CMSRunning As String

Dim objWMIcimv2 As Object

Dim objProcess As Object

Dim objList As Object

Dim Flag As Integer

On Error GoTo errorhandler

CoRegisterMessageFilter 0&, m1PreviousFilter

CMSRunning = "acsSRV.exe"

Set objWMIcimv2 = GetObject("winmgmts:" \_

& "{impersonationLevel=impersonate}!\\.\root\cimv2") 'Connect to CIMV2 Namespace

Set objList = objWMIcimv2.ExecQuery \_

("select \* from win32\_process where name='" & CMSRunning & "'") 'determine if CMS is running

USERLANID = Environ$("Username")

Set cvsApp = CreateObject("ACSUP.cvsApplication")

Set cvsConn = CreateObject("ACSCN.cvsConnection")

Set cvsSrv = CreateObject("ACSUPSRV.cvsServer")

Set Rep = CreateObject("ACSREP.cvsReport")

If cvsApp.CreateServer(Username, "", "", "10.99.132.21", False, "ENU", cvsSrv, cvsConn) Then

If cvsConn.Login(Username, Pword, "10.99.132.21", "ENU") Then

Application.ScreenUpdating = 0

Set cvsSrv = cvsApp.Servers(1)

Application.ScreenUpdating = 1

On Error Resume Next

cvsSrv.Reports.ACD = 1

rpdate = "01/01/2018-" & Date

Repdetails = "Report for the date " & Replace(rpdate, "-", " to ")

Set Info = cvsSrv.Reports.Reports("Historical\Designer\MIS Multi Sk Sum Dly BENNY")

b = cvsSrv.Reports.CreateReport(Info, Rep)

If b Then

Rep.Window.Top = 1830

Rep.Window.Left = 975

Rep.Window.Wime.DTh = 17610

Rep.Window.Height = 11910

Rep.SetProperty "Skill/s", 945 'change as needed for report variables

Rep.SetProperty "Date/s", rpdate

'Rep.SetProperty "Time", "07:45-20:30"

'Rep.SetProperty "ACD", "ANZ"

b = Rep.ExportData(Application.ThisWorkbook.path & "\Export.txt", 44, 1, True, True, True)

Rep.Quit

If Not cvsSrv.Interactive Then cvsSrv.ActiveTasks.Remove Rep.TaskID

Set Rep = Nothing

End If

filepath = Application.ThisWorkbook.path & "\Export.txt"

Set Info = Nothing

cvsConn.Logout

cvsConn.Disconnect

cvsSrv.Connected = False

Set Log = Nothing

Set Rep = Nothing

Set cvsSrv = Nothing

Set cvsConn = Nothing

Set cvsApp = Nothing

ThisWorkbook.Sheets("RD").Visible = True

ThisWorkbook.Sheets("RD").Select

ThisWorkbook.Sheets("RD").Range("A1:BZ100000").ClearContents

With ActiveSheet.QueryTables.Add(Connection:= \_

"TEXT;" & filepath, \_

Destination:=ThisWorkbook.Sheets("RD").Range("$A$12"))

.Name = "Export"

.FieldNames = True

.RowNumbers = False

.FillAdjacentFormulas = False

.PreserveFormatting = True

.RefreshOnFileOpen = False

.RefreshStyle = xlInsertDeleteCells

.SavePassword = False

.SaveData = True

.AdjustColumnWidth = True

.RefreshPeriod = 0

.TextFilePromptOnRefresh = False

.TextFilePlatform = 437

.TextFileStartRow = 1

.TextFileParseType = xlDelimited

.TextFileTextQualifier = xlTextQualifierNone

.TextFileConsecutiveDelimiter = False

.TextFileTabDelimiter = False

.TextFileSemicolonDelimiter = False

.TextFileCommaDelimiter = True

.TextFileSpaceDelimiter = False

.TextFileColumnDataTypes = Array(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)

.TextFileTrailingMinusNumbers = True

.Refresh BackgroundQuery:=False

End With

ThisWorkbook.Connections("Export").Delete

'Dim Pivot As PivotTable

'For Each Pivot In Sheet5.PivotTables

' Pivot.RefreshTable

' Pivot.Update

'Next

ThisWorkbook.Sheets("RD").Visible = False

ThisWorkbook.RefreshAll

ThisWorkbook.Sheets("Dashboard").Select

'ThisWorkbook.Sheets("Report").Visible = True

'ThisWorkbook.Sheets("Report").Range("A8").Value = Repdetails

'ThisWorkbook.Sheets("Contents").Visible = False

MsgBox ("Key Metrics Downloaded")

CoRegisterMessageFilter m1PreviousFilter, 0&

Exit Sub

Else

MsgBox ("Invalid Credentials!. Try to Relogin.")

CoRegisterMessageFilter m1PreviousFilter, 0&

CMSLoginUsrfrm.Show

Exit Sub

End If

Else

MsgBox ("Invalid UserName!. Try to Relogin.")

CoRegisterMessageFilter m1PreviousFilter, 0&

CMSLoginUsrfrm.Show

Exit Sub

End If

errorhandler:

MsgBox ("Unable to Login to CMS Avaya, Please Try to Relogin!")

CoRegisterMessageFilter m1PreviousFilter, 0&

End Sub

Private Sub Workbook\_SheetPivotTableUpdate(ByVal Pivots As Object, ByVal Target As PivotTable)

' Description:Update Timeline Slicers from Master Slicer

' Inputs: Sh PivotTable's worksheet

' Target PivotTable being changed/updated

' Outputs: \*None

' Requisites: \*None

' Example: \*None - This is an event handler found in ThisWorkbook module

' Date Ini Modification

' 10/19/16 CWH Initial Development

' Declarations

Const cRoutine As String = "Workbook\_SheetPivotTableUpdate"

Dim oSlicer As SlicerCache 'Current Slicer

Dim pSlicer As SlicerCache

Dim qSlicer As SlicerCache

'Const CC\_Conv\_Conversation\_Date As Long = 1 'Master Slicer

Const CC\_Date As Long = 1 'Master Slicer

'Const CC\_QC\_Dt\_Assmnt As Long = 1 'Master Slicer

Dim dStartDate As Date 'Start Date

Dim dEndDate As Date 'End Date

Dim bCleared As Boolean 'Filter Cleared Flag

Dim bEvents As Boolean 'Events Enabled Flag

' Error Handling Initialization

On Error GoTo ErrHandler

' Prevent cascading events

bEvents = Application.EnableEvents

Application.EnableEvents = False

' Get Master Slicer's dates

Set oSlicer = ThisWorkbook.SlicerCaches(CC\_Date)

'Set pSlicer = ThisWorkbook.SlicerCaches(CC\_Conv\_Conversation\_Date)

'Set qSlicer = ThisWorkbook.SlicerCaches(CC\_QC\_Dt\_Assmnt)

bCleared = oSlicer.FilterCleared

If Not bCleared Then

With oSlicer.TimelineState

dStartDate = .FilterValue1

dEndDate = .FilterValue2

End With

End If

' Set All other Timeline Slicer Dates

For Each oSlicer In ThisWorkbook.SlicerCaches

If oSlicer.SlicerCacheType = xlTimeline And \_

oSlicer.Index <> CC\_Date Then

If bCleared Then

oSlicer.ClearAllFilters

Else

oSlicer.TimelineState.SetFilterDateRange dStartDate, dEndDate

End If

End If

Next

ErrHandler:

Select Case Err.Number

Case Is = 0: 'Do nothing

Case Is = 9: 'Do Nothing Master Slicer Missing

Case Else:

Select Case MsgBox(Prompt:=Err.Description, \_

Buttons:=vbAbortRetryIgnore, \_

Title:=cRoutine, \_

HelpFile:=Err.HelpFile, \_

Context:=Err.HelpContext)

Case Is = vbAbort: Stop: Resume 'Debug mode - Trace

Case Is = vbRetry: Resume 'Try again

Case Is = vbIgnore: 'End routine

End Select

End Select

' Clean up: Resume responding to events

Application.EnableEvents = bEvents

End Sub

Option Explicit

Private Const LINE\_NO\_TEXT As String = "Line no: "

Dim AlreadyUsed As Boolean

' Reraises an error and adds line number and current procedure name

Sub RaiseError(ByVal errorNo As Long \_

, ByVal src As String \_

, ByVal proc As String \_

, ByVal desc As String \_

, ByVal lineNo As Long)

Dim sSource As String

' If called for the first time then add line number

If AlreadyUsed = False Then

' Add error line number if present

If lineNo <> 0 Then

sSource = vbNewLine & LINE\_NO\_TEXT & lineNo & " "

End If

' Add procedure to source

sSource = sSource & vbNewLine & proc

AlreadyUsed = True

Else

' If error has already been raised simply add on procedure name

sSource = src & vbNewLine & proc

End If

' Pause the code here when debugging

'(To Debug: "Tools->VBA Properties" from the menu.

' Add "Debugging=1" to the ' "Conditional Compilation Arguments.)

#If Debugging = 1 Then

Debug.Assert False

#End If

' Reraise the error so it will be caught in the caller procedure

' (Note: If the code stops here, make sure DisplayError has been

' placed in the topmost procedure)

Err.Raise errorNo, sSource, desc

End Sub

' Displays the error when it reaches the topmost sub

' Note: You can add a call to logging from this sub

Sub DisplayError(ByVal src As String, ByVal desc As String \_

, ByVal sProcname As String, lineNo As Long)

' Check If the error happens in topmost sub

If AlreadyUsed = False Then

' Reset string to remove "VBAProject" and add line number if it exists

src = IIf(lineNo = 0, "", vbNewLine & LINE\_NO\_TEXT & lineNo)

End If

' Build the final message

Dim sMsg As String

sMsg = "The following error occurred: " & vbNewLine & Err.Description \_

& vbNewLine & vbNewLine & "Error Location is: "

sMsg = sMsg & src & vbNewLine & sProcname

' Display the message

MsgBox sMsg, Title:="Error"

' reset the boolean value

AlreadyUsed = False

'disconnectFromDB conn

End Sub

Option Explicit

Public conn As Object

Public USERFULLNAME As String

Public USERLANID As String

Public Const QRYROW = 1

Public Const QRYCOL = 1

Public Const OUTPUTROW = 1

Public Const MAXROW = 1048576

Public Const MAXCOL = 16384

Sub testDBConnection()

On Error GoTo ErrorBlock

Dim wsCF As Worksheet

Dim userId As String, password As String

Dim conn As Object

Dim rs As Object

Set wsCF = Sheets("Config")

wsCF.Activate

'userId = ActiveSheet.userId.Text

'password = ActiveSheet.password.Text

'connectToDB conn, "", userId, password, "Oracle"

If conn.State = 1 Then

MsgBox "Connection successful"

End If

Set rs = Nothing

disconnectFromDB conn

Done:

Exit Sub

ErrorBlock:

RaiseError Err.Number, Err.Source, "Module1.testDBConnection", Err.Description, Erl

End Sub

Sub connectToDB(pConn As Object, pDBPath As String, pUserId As String, pPassword As String, pDBType As String)

On Error GoTo ErrorBlock

Dim sconnect As String

Set pConn = CreateObject("ADODB.Connection")

If pDBType = "Text" Then

'connect string for database as .csv file (text file)

sconnect = "Provider=Microsoft.Jet.OLEDB.4.0;" & \_

"Data Source=" & pDBPath & ";" & \_

"Extended Properties=""text;HDR=Yes"""

End If

If pDBType = "Excel" Then

'connect string for database as excel file

sconnect = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & pDBPath & \_

";Extended Properties=""Excel 12.0 Xml;HDR=Yes;IMEX=1"";"

End If

If pDBType = "MSAccess" Then

'connect string for database as ms access

sconnect = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" & pDBPath & ";"

End If

If pDBType = "Oracle" Then

sconnect = "Provider=MSDAORA;Data Source=FTCST01P\_SSL;User ID=" & pUserId & ";Password=" & pPassword

End If

pConn.Open sconnect

Done:

Exit Sub

ErrorBlock:

RaiseError Err.Number, Err.Source, "Module1.connectToDB", Err.Description, Erl

End Sub

Sub disconnectFromDB(pConn As Object)

On Error GoTo ErrorBlock

If Not pConn Is Nothing Then

If pConn.State = 1 Then

pConn.Close

End If

End If

Set pConn = Nothing

Done:

Exit Sub

ErrorBlock:

RaiseError Err.Number, Err.Source, "Module1.disconnectFromDB", Err.Description, Erl

End Sub

Sub fetchInputFileData(pPathname As String)

On Error GoTo ErrorBlock

Dim fullpath As String

fullpath = Application.GetOpenFilename \_

(Title:="Please choose a file", \_

Filefilter:="Access DB Files,\*.accdb")

If fullpath = "False" Then

MsgBox "No File Specified.", vbExclamation, "ERROR"

Exit Sub

End If

pPathname = fullpath

Done:

Exit Sub

ErrorBlock:

RaiseError Err.Number, Err.Source, "Module1.fetchInputFileData", Err.Description, Erl

End Sub

Function fetchRowNo(pParaName As String) As Integer

On Error GoTo ErrorBlock

Dim noOfRows As Long, row As Long

configSheet.Activate

noOfRows = Cells(ActiveSheet.Rows.Count, 2).End(xlUp).row

For row = 1 To noOfRows

If Trim(Cells(row, 2)) = pParaName Then

Exit For

End If

Next row

fetchRowNo = row

Done:

Exit Function

ErrorBlock:

RaiseError Err.Number, Err.Source, "Module1.fetchRowNo", Err.Description, Erl

End Function

Function FileExists(filepath As String) As Boolean

On Error GoTo ErrorBlock

Dim TestStr As String

TestStr = ""

On Error Resume Next

TestStr = Dir(filepath)

On Error GoTo 0

If TestStr = "" Then

FileExists = False

Else

FileExists = True

End If

Done:

Exit Function

ErrorBlock:

RaiseError Err.Number, Err.Source, "Module1.FileExists", Err.Description, Erl

End Function

Function AuthenticateUsers() As Boolean

On Error GoTo ErrorBlock

Dim sSQLQry As String, counter As Integer, found As Boolean

Dim rs As Object

Set rs = CreateObject("ADODB.Recordset")

USERLANID = Environ$("Username")

connectToDB conn, Base64DecodeString(configSheet.Cells(1, 2)), "", "", "MSAccess"

If conn.State = 0 Then

MsgBox "Not connected to DB!!!"

End If

sSQLQry = "Select fullname " & \_

"from AuthenticatedUsers au, access\_level al " & \_

"where au.lanid = al.lanid " & \_

"and au.lanid = '" & USERLANID & "' " & \_

"and al.AccessLevel in ('Manager','Agent')"

rs.Open sSQLQry, conn

found = False

If rs.EOF = False Then

USERFULLNAME = rs(0)

found = True

End If

If Not rs Is Nothing Then

If rs.State = 1 Then

rs.Close

End If

End If

Set rs = Nothing

disconnectFromDB conn

AuthenticateUsers = found

Done:

Exit Function

ErrorBlock:

RaiseError Err.Number, Err.Source, "Module1.AuthenticateUsers", Err.Description, Erl

End Function

Sub MTD()

' Description:Update Timeline Slicers from Master Slicer

' Inputs: Sh PivotTable's worksheet

' Target PivotTable being changed/updated

' Outputs: \*None

' Requisites: \*None

' Example: \*None - This is an event handler found in ThisWorkbook module

' Date Ini Modification

' 10/19/16 CWH Initial Development

' Declarations

'Const cRoutine As String = "Workbook\_SheetPivotTableUpdate"

Dim oSlicer As SlicerCache 'Current Slicer

Dim pSlicer As SlicerCache

Dim qSlicer As SlicerCache

'Const CC\_Conv\_Conversation\_Date As Long = 1 'Master Slicer

Const CC\_Date As Long = 1 'Master Slicer

'Const CC\_QC\_Dt\_Assmnt As Long = 1 'Master Slicer

Dim dStartDate As Date 'Start Date

Dim dEndDate As Date 'End Date

Dim bCleared As Boolean 'Filter Cleared Flag

Dim bEvents As Boolean 'Events Enabled Flag

' Error Handling Initialization

On Error GoTo ErrHandler

' Prevent cascading events

bEvents = Application.EnableEvents

Application.EnableEvents = False

' Get Master Slicer's dates

Set oSlicer = ThisWorkbook.SlicerCaches(CC\_Date)

'Set pSlicer = ThisWorkbook.SlicerCaches(CC\_Conv\_Conversation\_Date)

'Set qSlicer = ThisWorkbook.SlicerCaches(CC\_QC\_Dt\_Assmnt)

bCleared = oSlicer.FilterCleared

'If Not bCleared Then

' With oSlicer.TimelineState

dStartDate = DateSerial(Year(Date), Month(Date), 1)

dEndDate = Date

' End With

'End If

' Set All other Timeline Slicer Dates

For Each oSlicer In ThisWorkbook.SlicerCaches

If oSlicer.SlicerCacheType = xlTimeline Then

oSlicer.TimelineState.SetFilterDateRange dStartDate, dEndDate

End If

Next

ErrHandler:

Select Case Err.Number

Case Is = 0: 'Do nothing

Case Is = 9: 'Do Nothing Master Slicer Missing

Case Else:

Select Case MsgBox(Prompt:=Err.Description, \_

Buttons:=vbAbortRetryIgnore, \_

Title:="MTD Filter", \_

HelpFile:=Err.HelpFile, \_

Context:=Err.HelpContext)

Case Is = vbAbort: Stop: Resume 'Debug mode - Trace

Case Is = vbRetry: Resume 'Try again

Case Is = vbIgnore: 'End routine

End Select

End Select

' Clean up: Resume responding to events

Application.EnableEvents = bEvents

End Sub

Sub YTD()

' Description:Update Timeline Slicers from Master Slicer

' Inputs: Sh PivotTable's worksheet

' Target PivotTable being changed/updated

' Outputs: \*None

' Requisites: \*None

' Example: \*None - This is an event handler found in ThisWorkbook module

' Date Ini Modification

' 10/19/16 CWH Initial Development

' Declarations

'Const cRoutine As String = "Workbook\_SheetPivotTableUpdate"

Dim oSlicer As SlicerCache 'Current Slicer

Dim pSlicer As SlicerCache

Dim qSlicer As SlicerCache

'Const CC\_Conv\_Conversation\_Date As Long = 1 'Master Slicer

Const CC\_Date As Long = 1 'Master Slicer

'Const CC\_QC\_Dt\_Assmnt As Long = 1 'Master Slicer

Dim dStartDate As Date 'Start Date

Dim dEndDate As Date 'End Date

Dim bCleared As Boolean 'Filter Cleared Flag

Dim bEvents As Boolean 'Events Enabled Flag

' Error Handling Initialization

On Error GoTo ErrHandler

' Prevent cascading events

bEvents = Application.EnableEvents

Application.EnableEvents = False

' Get Master Slicer's dates

Set oSlicer = ThisWorkbook.SlicerCaches(CC\_Date)

'Set pSlicer = ThisWorkbook.SlicerCaches(CC\_Conv\_Conversation\_Date)

'Set qSlicer = ThisWorkbook.SlicerCaches(CC\_QC\_Dt\_Assmnt)

bCleared = oSlicer.FilterCleared

'If Not bCleared Then

' With oSlicer.TimelineState

dStartDate = DateSerial(IIf(Month(Date) > 9, Year(Date), Year(Date) - 1), 10, 1)

dEndDate = Date

' End With

'End If

' Set All other Timeline Slicer Dates

For Each oSlicer In ThisWorkbook.SlicerCaches

If oSlicer.SlicerCacheType = xlTimeline Then

oSlicer.TimelineState.SetFilterDateRange dStartDate, dEndDate

End If

Next

ErrHandler:

Select Case Err.Number

Case Is = 0: 'Do nothing

Case Is = 9: 'Do Nothing Master Slicer Missing

Case Else:

Select Case MsgBox(Prompt:=Err.Description, \_

Buttons:=vbAbortRetryIgnore, \_

Title:="YTD Filter", \_

HelpFile:=Err.HelpFile, \_

Context:=Err.HelpContext)

Case Is = vbAbort: Stop: Resume 'Debug mode - Trace

Case Is = vbRetry: Resume 'Try again

Case Is = vbIgnore: 'End routine

End Select

End Select

' Clean up: Resume responding to events

Application.EnableEvents = bEvents

End Sub

Sub RefreshBtn\_Click()

ThisWorkbook.RefreshAll

MsgBox ("All Pivots and Tables are Refreshed")

End Sub

Sub FinYear()

' Description:Update Timeline Slicers from Master Slicer

' Inputs: Sh PivotTable's worksheet

' Target PivotTable being changed/updated

' Outputs: \*None

' Requisites: \*None

' Example: \*None - This is an event handler found in ThisWorkbook module

' Date Ini Modification

' 10/19/16 CWH Initial Development

' Declarations

'Const cRoutine As String = "Workbook\_SheetPivotTableUpdate"

Dim oSlicer As SlicerCache 'Current Slicer

Dim pSlicer As SlicerCache

Dim qSlicer As SlicerCache

'Const CC\_Conv\_Conversation\_Date As Long = 1 'Master Slicer

Const CC\_Date As Long = 1 'Master Slicer

'Const CC\_QC\_Dt\_Assmnt As Long = 1 'Master Slicer

Dim dStartDate As Date 'Start Date

Dim dEndDate As Date 'End Date

Dim bCleared As Boolean 'Filter Cleared Flag

Dim bEvents As Boolean 'Events Enabled Flag

' Error Handling Initialization

On Error GoTo ErrHandler

' Prevent cascading events

bEvents = Application.EnableEvents

Application.EnableEvents = False

FinYearUsrFrm.Show

' Get Master Slicer's dates

Set oSlicer = ThisWorkbook.SlicerCaches(CC\_Date)

'Set pSlicer = ThisWorkbook.SlicerCaches(CC\_Conv\_Conversation\_Date)

'Set qSlicer = ThisWorkbook.SlicerCaches(CC\_QC\_Dt\_Assmnt)

bCleared = oSlicer.FilterCleared

'If Not bCleared Then

' With oSlicer.TimelineState

dStartDate = DateSerial(IIf(Month(Date) > 9, Year(Date) - 1, Year(Date) - 2), 10, 1)

dEndDate = DateSerial(Year(dStartDate) + 1, 9, 30)

' End With

'End If

' Set All other Timeline Slicer Dates

For Each oSlicer In ThisWorkbook.SlicerCaches

If oSlicer.SlicerCacheType = xlTimeline Then

oSlicer.TimelineState.SetFilterDateRange dStartDate, dEndDate

End If

Next

ErrHandler:

Select Case Err.Number

Case Is = 0: 'Do nothing

Case Is = 9: 'Do Nothing Master Slicer Missing

Case Else:

Select Case MsgBox(Prompt:=Err.Description, \_

Buttons:=vbAbortRetryIgnore, \_

Title:="YTD Filter", \_

HelpFile:=Err.HelpFile, \_

Context:=Err.HelpContext)

Case Is = vbAbort: Stop: Resume 'Debug mode - Trace

Case Is = vbRetry: Resume 'Try again

Case Is = vbIgnore: 'End routine

End Select

End Select

' Clean up: Resume responding to events

Application.EnableEvents = bEvents

End Sub

Sub WTD()

' Description:Update Timeline Slicers from Master Slicer

' Inputs: Sh PivotTable's worksheet

' Target PivotTable being changed/updated

' Outputs: \*None

' Requisites: \*None

' Example: \*None - This is an event handler found in ThisWorkbook module

' Date Ini Modification

' 10/19/16 CWH Initial Development

' Declarations

'Const cRoutine As String = "Workbook\_SheetPivotTableUpdate"

Dim oSlicer As SlicerCache 'Current Slicer

Dim pSlicer As SlicerCache

Dim qSlicer As SlicerCache

'Const CC\_Conv\_Conversation\_Date As Long = 1 'Master Slicer

Const CC\_Date As Long = 1 'Master Slicer

'Const CC\_QC\_Dt\_Assmnt As Long = 1 'Master Slicer

Dim dStartDate As Date 'Start Date

Dim dEndDate As Date 'End Date

Dim bCleared As Boolean 'Filter Cleared Flag

Dim bEvents As Boolean 'Events Enabled Flag

' Error Handling Initialization

On Error GoTo ErrHandler

' Prevent cascading events

bEvents = Application.EnableEvents

Application.EnableEvents = False

' Get Master Slicer's dates

Set oSlicer = ThisWorkbook.SlicerCaches(CC\_Date)

'Set pSlicer = ThisWorkbook.SlicerCaches(CC\_Conv\_Conversation\_Date)

'Set qSlicer = ThisWorkbook.SlicerCaches(CC\_QC\_Dt\_Assmnt)

bCleared = oSlicer.FilterCleared

'If Not bCleared Then

' With oSlicer.TimelineState

dStartDate = IIf(Weekday(Date) = 2, Date, Date - Weekday(Date, 3))

dEndDate = IIf(Weekday(Date) = 7 Or Weekday(Date) = 1, Date - Weekday(Date, 7), Date)

' End With

'End If

' Set All other Timeline Slicer Dates

For Each oSlicer In ThisWorkbook.SlicerCaches

If oSlicer.SlicerCacheType = xlTimeline Then

oSlicer.TimelineState.SetFilterDateRange dStartDate, dEndDate

End If

Next

ErrHandler:

Select Case Err.Number

Case Is = 0: 'Do nothing

Case Is = 9: 'Do Nothing Master Slicer Missing

Case Else:

Select Case MsgBox(Prompt:=Err.Description, \_

Buttons:=vbAbortRetryIgnore, \_

Title:="MTD Filter", \_

HelpFile:=Err.HelpFile, \_

Context:=Err.HelpContext)

Case Is = vbAbort: Stop: Resume 'Debug mode - Trace

Case Is = vbRetry: Resume 'Try again

Case Is = vbIgnore: 'End routine

End Select

End Select

' Clean up: Resume responding to events

Application.EnableEvents = bEvents

End Sub

Sub ToggleBtn\_Click()

ThisWorkbook.Sheets("Dashboard").Select

If ThisWorkbook.Sheets("Dashboard").Range("AF1").Value = "Chart" Then

ThisWorkbook.Sheets("Dashboard").Shapes("Chart 1").Visible = False

ThisWorkbook.Sheets("Dashboard").Range("AF1").Value = "Table"

ThisWorkbook.Sheets("Dashboard").Range("K4:R15").Select

With Selection.Interior

.Pattern = xlSolid

.PatternColorIndex = xlAutomatic

.ThemeColor = xlThemeColorAccent5

.TintAndShade = 0.799981688894314

.PatternTintAndShade = 0

End With

Selection.Borders(xlDiagonalDown).LineStyle = xlNone

Selection.Borders(xlDiagonalUp).LineStyle = xlNone

With Selection.Borders(xlEdgeLeft)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlEdgeTop)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlEdgeBottom)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlEdgeRight)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlInsideVertical)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlInsideHorizontal)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

ThisWorkbook.Sheets("Dashboard").Buttons("Button 10").Text = "Overall Stats"

ElseIf ThisWorkbook.Sheets("Dashboard").Range("AF1").Value = "Table" Then

ThisWorkbook.Sheets("Dashboard").Shapes("Chart 1").Visible = True

ThisWorkbook.Sheets("Dashboard").Range("AF1").Value = "Chart"

ThisWorkbook.Sheets("Dashboard").Range("K4:R15").Select

With Selection.Interior

.Pattern = xlSolid

.PatternColorIndex = xlAutomatic

.ThemeColor = xlThemeColorAccent1

.TintAndShade = 0.799981688894314

.PatternTintAndShade = 0

End With

Selection.Borders(xlDiagonalDown).LineStyle = xlNone

Selection.Borders(xlDiagonalUp).LineStyle = xlNone

Selection.Borders(xlEdgeLeft).LineStyle = xlNone

Selection.Borders(xlEdgeTop).LineStyle = xlNone

Selection.Borders(xlEdgeBottom).LineStyle = xlNone

Selection.Borders(xlEdgeRight).LineStyle = xlNone

Selection.Borders(xlInsideVertical).LineStyle = xlNone

Selection.Borders(xlInsideHorizontal).LineStyle = xlNone

ThisWorkbook.Sheets("Dashboard").Range("K6:L12").Select

With Selection.Interior

.Pattern = xlSolid

.PatternColorIndex = xlAutomatic

.ThemeColor = xlThemeColorAccent5

.TintAndShade = 0.799981688894314

.PatternTintAndShade = 0

End With

Selection.Borders(xlDiagonalDown).LineStyle = xlNone

Selection.Borders(xlDiagonalUp).LineStyle = xlNone

With Selection.Borders(xlEdgeLeft)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlEdgeTop)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlEdgeBottom)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlEdgeRight)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlInsideVertical)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

With Selection.Borders(xlInsideHorizontal)

.LineStyle = xlContinuous

.ThemeColor = 1

.TintAndShade = 0

.Weight = xlMedium

End With

ThisWorkbook.Sheets("Dashboard").Buttons("Button 10").Text = "Campaign Details"

End If

ThisWorkbook.Sheets("Dashboard").Range("E1").Select

End Sub

Private InitDone As Boolean

Private Map1(0 To 63) As Byte

Private Map2(0 To 127) As Byte

' Decodes a string from Base64 format.

' Parameters:

' s a Base64 String to be decoded.

' Returns a String containing the decoded data.

Public Function Base64DecodeString(ByVal s As String) As String

If s = "" Then Base64DecodeString = "": Exit Function

Base64DecodeString = ConvertBytesToString(Base64Decode(s))

End Function

' Decodes a byte array from Base64 format.

' Parameters

' s a Base64 String to be decoded.

' Returns: an array containing the decoded data bytes.

Public Function Base64Decode(ByVal s As String) As Byte()

If Not InitDone Then Init

Dim IBuf() As Byte: IBuf = ConvertStringToBytes(s)

Dim ILen As Long: ILen = UBound(IBuf) + 1

If ILen Mod 4 <> 0 Then Err.Raise vbObjectError, , "Invalid Config.prop file"

Do While ILen > 0

If IBuf(ILen - 1) <> Asc("=") Then Exit Do

ILen = ILen - 1

Loop

Dim OLen As Long: OLen = (ILen \* 3) \ 4

Dim Out() As Byte

ReDim Out(0 To OLen - 1) As Byte

Dim ip As Long

Dim op As Long

Do While ip < ILen

Dim i0 As Byte: i0 = IBuf(ip): ip = ip + 1

Dim i1 As Byte: i1 = IBuf(ip): ip = ip + 1

Dim i2 As Byte: If ip < ILen Then i2 = IBuf(ip): ip = ip + 1 Else i2 = Asc("A")

Dim i3 As Byte: If ip < ILen Then i3 = IBuf(ip): ip = ip + 1 Else i3 = Asc("A")

If i0 > 127 Or i1 > 127 Or i2 > 127 Or i3 > 127 Then \_

Err.Raise vbObjectError, , "Invalid Config.prop file"

Dim b0 As Byte: b0 = Map2(i0)

Dim b1 As Byte: b1 = Map2(i1)

Dim b2 As Byte: b2 = Map2(i2)

Dim b3 As Byte: b3 = Map2(i3)

If b0 > 63 Or b1 > 63 Or b2 > 63 Or b3 > 63 Then \_

Err.Raise vbObjectError, , "Invalid Config.prop file."

Dim o0 As Byte: o0 = (b0 \* 4) Or (b1 \ &H10)

Dim o1 As Byte: o1 = ((b1 And &HF) \* &H10) Or (b2 \ 4)

Dim o2 As Byte: o2 = ((b2 And 3) \* &H40) Or b3

Out(op) = o0: op = op + 1

If op < OLen Then Out(op) = o1: op = op + 1

If op < OLen Then Out(op) = o2: op = op + 1

Loop

Base64Decode = Out

End Function

Private Sub Init()

Dim c As Integer, i As Integer

' set Map1

i = 0

For c = Asc("A") To Asc("Z"): Map1(i) = c: i = i + 1: Next

For c = Asc("a") To Asc("z"): Map1(i) = c: i = i + 1: Next

For c = Asc("0") To Asc("9"): Map1(i) = c: i = i + 1: Next

Map1(i) = Asc("+"): i = i + 1

Map1(i) = Asc("/"): i = i + 1

' set Map2

For i = 0 To 127: Map2(i) = 255: Next

For i = 0 To 63: Map2(Map1(i)) = i: Next

InitDone = True

End Sub

Private Function ConvertStringToBytes(ByVal s As String) As Byte()

Dim b1() As Byte: b1 = s

Dim l As Long: l = (UBound(b1) + 1) \ 2

If l = 0 Then ConvertStringToBytes = b1: Exit Function

Dim b2() As Byte

ReDim b2(0 To l - 1) As Byte

Dim p As Long

For p = 0 To l - 1

Dim c As Long: c = b1(2 \* p) + 256 \* CLng(b1(2 \* p + 1))

If c >= 256 Then c = Asc("?")

b2(p) = c

Next

ConvertStringToBytes = b2

End Function

Private Function ConvertBytesToString(b() As Byte) As String

Dim l As Long: l = UBound(b) - LBound(b) + 1

Dim b2() As Byte

ReDim b2(0 To (2 \* l) - 1) As Byte

Dim p0 As Long: p0 = LBound(b)

Dim p As Long

For p = 0 To l - 1: b2(2 \* p) = b(p0 + p): Next

Dim s As String: s = b2

ConvertBytesToString = s

End Function

Sub Button3\_Click()

CMSLoginUsrfrm.Show

End Sub

Sub Pivot\_CallOutcome()

Dim PvtTbl As PivotTable

Dim cnt As Integer

Dim val As String

Set PvtTbl = Worksheets("Pivots").PivotTables("PivotTable16")

'delete all filters currently applied to the PivotTable, using the PivotTable.ClearAllFilters Method

PvtTbl.ClearAllFilters

PvtTbl.PivotFields("CC\_Conv\_Nbr").PivotItems("0").Visible = False

PvtTbl.PivotFields("CC\_Conv\_Type\_Of\_Call").CurrentPage = "Out-Bound"

cnt = ThisWorkbook.Sheets("Pivots").Range("AP1004").Value

Dim pivot\_item As PivotItem

Dim match As Boolean

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name <> "(blank)" Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(pivot\_item.Name).Visible = False

End If

Next pivot\_item

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AP" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

Set PvtTbl = Worksheets("Pivots").PivotTables("PivotTable17")

'delete all filters currently applied to the PivotTable, using the PivotTable.ClearAllFilters Method

PvtTbl.ClearAllFilters

PvtTbl.PivotFields("CC\_Conv\_Nbr").PivotItems("0").Visible = False

PvtTbl.PivotFields("CC\_Conv\_Type\_Of\_Call").CurrentPage = "Out-Bound"

cnt = ThisWorkbook.Sheets("Pivots").Range("AQ1004").Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name <> "(blank)" Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(pivot\_item.Name).Visible = False

End If

Next pivot\_item

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AQ" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

Set PvtTbl = Worksheets("Pivots").PivotTables("PivotTable18")

'delete all filters currently applied to the PivotTable, using the PivotTable.ClearAllFilters Method

PvtTbl.ClearAllFilters

PvtTbl.PivotFields("CC\_Conv\_Nbr").PivotItems("0").Visible = False

PvtTbl.PivotFields("CC\_Conv\_Type\_Of\_Call").CurrentPage = "Out-Bound"

cnt = ThisWorkbook.Sheets("Pivots").Range("AR1004").Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name <> "(blank)" Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(pivot\_item.Name).Visible = False

End If

Next pivot\_item

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AR" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

Set PvtTbl = Worksheets("Pivots").PivotTables("PivotTable19")

'delete all filters currently applied to the PivotTable, using the PivotTable.ClearAllFilters Method

PvtTbl.ClearAllFilters

PvtTbl.PivotFields("CC\_Conv\_Nbr").PivotItems("0").Visible = False

PvtTbl.PivotFields("CC\_Conv\_Type\_Of\_Call").CurrentPage = "Out-Bound"

cnt = ThisWorkbook.Sheets("Pivots").Range("AS1004").Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name <> "(blank)" Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(pivot\_item.Name).Visible = False

End If

Next pivot\_item

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AS" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

Set PvtTbl = Worksheets("Pivots").PivotTables("PivotTable21")

'delete all filters currently applied to the PivotTable, using the PivotTable.ClearAllFilters Method

PvtTbl.ClearAllFilters

PvtTbl.PivotFields("CC\_Conv\_Nbr").PivotItems("0").Visible = False

PvtTbl.PivotFields("CC\_Conv\_Type\_Of\_Call").CurrentPage = "Out-Bound"

cnt = ThisWorkbook.Sheets("Pivots").Range("AT1004").Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name <> "(blank)" Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(pivot\_item.Name).Visible = False

End If

Next pivot\_item

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AT" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

Set PvtTbl = Worksheets("Pivots").PivotTables("PivotTable24")

'delete all filters currently applied to the PivotTable, using the PivotTable.ClearAllFilters Method

PvtTbl.ClearAllFilters

PvtTbl.PivotFields("CC\_Conv\_Nbr").PivotItems("0").Visible = False

PvtTbl.PivotFields("CC\_Conv\_Type\_Of\_Call").CurrentPage = "Out-Bound"

cnt = ThisWorkbook.Sheets("Pivots").Range("AU1004").Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name <> "(blank)" Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(pivot\_item.Name).Visible = False

End If

Next pivot\_item

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AU" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

Set PvtTbl = Worksheets("Pivots").PivotTables("PivotTable25")

'delete all filters currently applied to the PivotTable, using the PivotTable.ClearAllFilters Method

PvtTbl.ClearAllFilters

PvtTbl.PivotFields("CC\_Conv\_Nbr").PivotItems("0").Visible = False

PvtTbl.PivotFields("CC\_Conv\_Type\_Of\_Call").CurrentPage = "Out-Bound"

cnt = ThisWorkbook.Sheets("Pivots").Range("AV1004").Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name <> "(blank)" Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(pivot\_item.Name).Visible = False

End If

Next pivot\_item

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AV" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

MsgBox ("Done")

End Sub

Sub Pivot\_CallOutcome\_All()

Dim PvtTbl As PivotTable

Dim cnt As Integer

Dim val As String

Set PvtTbl = Worksheets("Pivots").PivotTables("PivotTable23")

'delete all filters currently applied to the PivotTable, using the PivotTable.ClearAllFilters Method

PvtTbl.ClearAllFilters

PvtTbl.PivotFields("CC\_Conv\_Nbr").PivotItems("0").Visible = False

PvtTbl.PivotFields("CC\_Conv\_Type\_Of\_Call").CurrentPage = "Out-Bound"

cnt = ThisWorkbook.Sheets("Pivots").Range("AP1004").Value

Dim pivot\_item As PivotItem

Dim match As Boolean

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name <> "(blank)" Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(pivot\_item.Name).Visible = False

End If

Next pivot\_item

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AP" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

cnt = ThisWorkbook.Sheets("Pivots").Range("AQ1004").Value

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AQ" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

cnt = ThisWorkbook.Sheets("Pivots").Range("AR1004").Value

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AR" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

cnt = ThisWorkbook.Sheets("Pivots").Range("AS1004").Value

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AS" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

cnt = ThisWorkbook.Sheets("Pivots").Range("AT1004").Value

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AT" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

cnt = ThisWorkbook.Sheets("Pivots").Range("AU1004").Value

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AU" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

cnt = ThisWorkbook.Sheets("Pivots").Range("AV1004").Value

For i = 1 To cnt

val = ThisWorkbook.Sheets("Pivots").Range("AV" & 1005 + i).Value

For Each pivot\_item In PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems

If pivot\_item.Name = val Then

match = True

Exit For

Else

match = False

End If

Next pivot\_item

If match Then

PvtTbl.PivotFields("CC\_Conv\_Call\_Outcome").PivotItems(val).Visible = True

End If

Next i

MsgBox ("Done")

End Sub