Function ActSaldosXFiltro(AñoAct As String)

Dim DB As Database

Dim RCAux As Recordset, RcAños As Recordset

Dim RC As Recordset, RcMovDia As Recordset

Dim qry As String, Año As String, Mes As String

Dim CtrlSaldos As String, CtrlCarAbo As String

Dim PolDia As String, MovDia As String

Dim polmes As String, movmes As String

Dim catMes As String, CatRep As String

Dim CatAux As String, CatAuxRep As String

Dim MesAnt As Integer, MesAct As Integer

Dim intMes As Integer, MesFin As String

Dim HoraInicio As Date, Fec\_Proc As Date

Dim Actualizar As Boolean

On Error GoTo ErrorActSaldos

**'############################################################**

HoraInicio = Now()

Set DB = CurrentDb()

PolDia = "PolDia" & AñoAct

MovDia = "MovDia" & AñoAct

CatRep = "CatRep" & AñoAct

CatAuxRep = "CatAuxRep" & AñoAct

CtrlCarAbo = "ControlCarAbo" & AñoAct

**'############################################################**

**'Ya fué actualizado?**

Set RC = DB.OpenRecordset(CtrlCarAbo, dbOpenDynaset)

If RC.EOF Then

**'No existen registros**

Beep

MsgBox "¡Aún no ha iniciado una actualización de Cargos y Abonos!, o en otro caso se hicieron modificaciones a las pólizas después de haber generado el proceso", vbCritical

RC.Close

Exit Function

End If

MesAnt = RC("Mes")

**'Aqui se agrego un ciclo por cada año generado para mover**

**'los saldos de un dia a otro**

qry = "SELECT right(Ejercicio,4) as Anos FROM Ejercicios WHERE CINT(right(Ejercicio,4))>=" & AñoAct & \_

" Order by Ejercicio"

Set RcAños = DB.OpenRecordset(qry, dbOpenSnapshot)

Do While Not RcAños.EOF

Año = RcAños!Anos

If Año <> AñoAct Then MesAnt = 1

**'Busco el último mes cerrado por ejercicio generado**

Set RCAux = DB.OpenRecordset("Control" & Año, dbOpenDynaset)

RCAux.MoveLast

MesAct = MesNom2Num(Mid(RCAux("NueCat"), 4, 3))

MesFin = Mid(RCAux("NueCat"), 4, 3)

RCAux.Close

**'############################################################**

For intMes = MesAnt To MesAct

Mes = Left(MesNum2Nom(intMes), 3)

catMes = "Cat" & Mes & Año

CatAux = "CatAux" & Mes & Año

polmes = "Pol" & Mes & Año

movmes = "Mov" & Mes & Año

**'Fué un mes actualizado?**

RC.FindFirst ("Mes = '" & intMes & "'")

If Not RC.NoMatch And Año = AñoAct Then

**'Actualizar los Saldos**

DoCmd.RunSQL ("UPDATE " & catMes & " SET [SANTPROCESO]=[SACT],[SACT]=[SACT]+[CARPROCESO]-[ABOPROCESO]," & \_

"[SPROMANT]=[SPROM],[SPROM2ANT]=[SPROM2],[CAR]=[CAR]+[CARPROCESO]," & \_

"[ABO]=[ABO]+[ABOPROCESO] WHERE [CARPROCESO]<>0 OR [ABOPROCESO]<>0")

**'Actualizar los Saldos del Catalogo Auxiliar**

DoCmd.RunSQL ("UPDATE " & CatAux & " SET [SANTPROCESO]=[SACT],[SACT]=[SACT]+[CARPROCESO]-[ABOPROCESO]," & \_

"[SPROMANT]=[SPROM],[SPROM2ANT]=[SPROM2],[CAR]=[CAR]+[CARPROCESO]," & \_

"[ABO]=[ABO]+[ABOPROCESO] WHERE [CARPROCESO]<>0 OR [ABOPROCESO]<>0")

**'Agregar los movimientos a las tablas de los meses correspondientes**

qry = "INSERT INTO " & polmes & " SELECT " & PolDia & ".\* " & \_

"FROM " & MovDia & " INNER JOIN " & PolDia & " " & \_

"ON (" & MovDia & ".Fecha\_mov = " & PolDia & ".Fecha\_mov) " & \_

"AND (" & MovDia & ".Referencia = " & PolDia & ".Referencia) " & \_

"WHERE (" & MovDia & ".Status = 2) " & \_

"AND (" & MovDia & ".MES\_MOV = '" & Mes & "')"

DoCmd.RunSQL (qry)

qry = "INSERT INTO " & movmes & " SELECT \* " & \_

"FROM " & MovDia & " " & \_

"WHERE (Status = 2) " & \_

"AND (MES\_MOV = '" & Mes & "')"

DoCmd.RunSQL (qry)

**'Borrar las cuentas actualizadas de las tablas temporales**

qry = "DELETE P.\* FROM " & MovDia & " AS M INNER JOIN " & PolDia & " AS P " & \_

"ON (M.Fecha\_Mov = P.Fecha\_mov) AND (M.Referencia = P.Referencia) " & \_

"WHERE M.Status=2 AND M.Mes\_Mov='" & Mes & "'"

DoCmd.RunSQL (qry)

qry = "DELETE \* " & \_

"FROM " & MovDia & " " & \_

"WHERE Status = 2 AND MES\_MOV = '" & Mes & "'"

DoCmd.RunSQL (qry)

End If

If intMes <> MesAnt Or Año <> AñoAct Then

**'Actualizar los saldos del mes anterior al de ahora**

qry = "UPDATE " & CatRep & " INNER JOIN " & catMes & " ON (" & \_

CatRep & ".Llave = " & catMes & ".Llave) AND (" & \_

CatRep & ".Moneda = " & catMes & ".Moneda) " & \_

"SET " & \_

catMes & ".SAnt = " & catMes & ".SAnt + " & CatRep & ".SAct, " & \_

catMes & ".SAct = " & catMes & ".SAct + " & CatRep & ".SAct, " & \_

catMes & ".SProm = " & catMes & ".SProm + " & CatRep & ".SAct, " & \_

catMes & ".SProm2 = " & catMes & ".SProm2 + " & CatRep & ".SAct, " & \_

catMes & ".SAntProceso = " & catMes & ".SAntProceso + " & CatRep & ".SAct"

If Año = AñoAct Then

qry = qry & " WHERE (" & CatRep & ".FECCAP < " & intMes & ")"

End If

**'Actualizar los cambios**

DoCmd.RunSQL (qry)

**'Actualizar los saldos del mes anterior al de ahora del Catalogo Auxiliar**

qry = "UPDATE " & CatAuxRep & " INNER JOIN " & CatAux & " ON (" & \_

CatAuxRep & ".Llave = " & CatAux & ".Llave) AND (" & \_

CatAuxRep & ".Moneda = " & CatAux & ".Moneda) AND (" & \_

CatAuxRep & ".Region\_Id = " & CatAux & ".Region\_Id) AND (" & \_

CatAuxRep & ".Sucursal\_Id = " & CatAux & ".Sucursal\_Id) AND (" & \_

CatAuxRep & ".Sector\_Id = " & CatAux & ".Sector\_Id) " & \_

"SET " & \_

CatAux & ".SAnt = " & CatAux & ".SAnt + " & CatAuxRep & ".SAct, " & \_

CatAux & ".SAct = " & CatAux & ".SAct + " & CatAuxRep & ".SAct, " & \_

CatAux & ".SProm = " & CatAux & ".SProm + " & CatAuxRep & ".SAct, " & \_

CatAux & ".SProm2 = " & CatAux & ".SProm2 + " & CatAuxRep & ".SAct, " & \_

CatAux & ".SAntProceso = " & CatAux & ".SAntProceso + " & CatAuxRep & ".SAct"

If Año = AñoAct Then

qry = qry & " WHERE (" & CatAuxRep & ".FECCAP < " & intMes & ")"

End If

**'Actualizar los cambios**

DoCmd.RunSQL (qry)

If MesFin = "Fin" And intMes = 13 And Año = AñoAct Then

**'Último mes del ejercicio, CASO ESPECIAL**

**'Genera Póliza del resultado del ejercicio**

**'siempre las cuentas de IN o EG se hayan movido**

**'Verificamos que las cuentas de IN O EG se hayan alterado**

qry = "SELECT Llave, Naturaleza\_id, SAct FROM Rangos, CatAuxCie" & \_

Año & " WHERE Sucursal\_id=0 AND Region\_id=0 AND " & \_

"(Naturaleza\_id='IN' Or Naturaleza\_id='EG') AND SAct<>0 " & \_

"AND Left([Llave],4) Between [llave\_inicial] And [llave\_final]"

Set RCAux = CurrentDb().OpenRecordset(qry, dbOpenDynaset)

If Not RCAux.EOF Then

MsgBox "Es Necesario Capturar la cuenta de Resultado del Ejercicio " & Año & " para la " & Chr(10) & \_

"Póliza de Cierre, ¡Verifique Captura de Cuentas Predetermidas!", vbCritical

DoCmd.OpenForm "f\_cuentas\_predeterminadas\_aux", acNormal, , , , acDialog, "CIE" & Año

**'Aquí se genera una poliza de Cierre para poner las cuentas**

**'de ingresos y egresos en Ceros y poner en la cueta predeterminada**

**'El resultado del ejercicio**

Call GeneraPolizaCierre("Cie", Año, True)

Pedro: aquí se manda llamar ese método, a continuación, dejo lo que contine el método:

Public Function GeneraPolizaCierre(Mes As String, Año As String, Agregar As Boolean) As Boolean

Dim RC As Recordset, RCAux As Recordset

Dim ConsCab As Integer

Dim Cons As Integer, NoDer As Integer

Dim TCons As Integer, NoIzq As Integer

Dim TCargos As Double

Dim Tabonos As Double

Dim CuentaResultado As String, Moneda As String

Dim TipoPol As String, Llave As String

Dim qry As String, Ceros As String

Dim Utilidad As Currency, Poliza As String

Dim i As Integer, Actual As String

Dim RCAreas As Recordset

Dim SucursalGlobal\_ID As Long

Dim intMes As Integer

Dim DB As Database

Dim q As QueryDef

GeneraPolizaCierre = False

Set DB = CurrentDb()

**'Catalogo Cierre**

qry = "SELECT \* FROM Cat" & Mes & Año

Set q = DB.QueryDefs("CatMesCie")

q.SQL = qry

q.Close

qry = "SELECT \* FROM CatAux" & Mes & Año

Set q = DB.QueryDefs("CatAuxMesCie")

q.SQL = qry

q.Close

**'Mes a procesar**

intMes = MesNom2Num(Mes)

**'Obtiene Tipo de Polia y Cuenta de Resultado del Ejercicio**

Set RC = CurrentDb().OpenRecordset("Select CuentaInterna, TipoPoliza, Moneda From Cuentas\_predeterminadas WHERE ID = 1", dbOpenSnapshot)

If RC.EOF Then

MsgBox "Es Necesario Capturar la cuenta de Resultado del Ejercicio para la " & Chr(10) & \_

"Póliza de Cierre, ¡Verifique Captura de Cuentas Predetermidas!", vbCritical

RC.Close

Exit Function

Else

If IsNull(RC!CuentaInterna) Or IsNull(RC!TipoPoliza) Or IsNull(RC!Moneda) Or RC!CuentaInterna = "" Or RC!TipoPoliza = "" Then

MsgBox "Es Necesario Capturar la cuenta de Resultado del Ejercicio para la " & Chr(10) & \_

"Póliza de Cierre, ¡Verifique Captura de Cuentas Predetermidas!", vbCritical

RC.Close

Exit Function

End If

CuentaResultado = RC!CuentaInterna

TipoPol = RC!TipoPoliza

Moneda = RC!Moneda

End If

RC.Close

**'Se detecto que había un error, que la póliza de fin el ejercicio se contabilizaba CORPORATIVA (0,0)**

**'Tremendo error, porque, aunque la cuenta de Resultado del Ejercicio no sea Sucursalizable,**

**'las cuentas de resultado se deberan contabilizar a nivel de Sucursal\_ID, de lo contrario las saldos**

**'solo se arrastran a nivel de corporativo y no por CC**

'<HDH><20110117>

**'Parametro "Esquema Contable", 2;Significa que todo el catálogo de cuentas es sucursalizable,**

**'entonces se va a generar una póliza por cada "Area o Región", de lo contrario se hace lo mismo una sola póliza Corporativa (0,0)**

**'<HDH>Add<20110118>**

If parms(12) = 2 Then

Actual = ""

qry = "SELECT CatAuxMesCie.Region\_id FROM CatAuxMesCie, Rangos INNER JOIN SubRangos ON Rangos.Rango\_id = SubRangos.Rango\_id " & \_

"WHERE (((Rangos.Naturaleza\_id)='IN' Or (Rangos.Naturaleza\_id)='EG') AND ((CatAuxMesCie.SAct)<>0) AND ((CatAuxMesCie.Region\_id)<>0) " & \_

"AND ((Left([Llave],4)) Between [subrangos].[llave\_inicial] And [subrangos].[llave\_final])) GROUP BY CatAuxMesCie.Region\_id"

Set RCAreas = CurrentDb().OpenRecordset(qry)

If RCAreas.EOF Then

MsgBox "No existen movimentos para la póliza de resultado del ejercicio", vbInformation, "Póliza de Cierre sin Movimientos"

RCAreas.Close

Exit Function

End If

Do While Not RCAreas.EOF

**'Se verifica para las Areas o Regiones afectadas tenga la cuenta de Resultado del Ejercicio este dada de alta la cuenta con el CC Global**

qry = "SELECT CatAuxMesCie.Sucursal\_id " & \_

"FROM CatAuxMesCie INNER JOIN [Departamento o Sucursal] ON (CatAuxMesCie.Region\_id = [Departamento o Sucursal].Region\_id) " & \_

"AND (CatAuxMesCie.Sucursal\_id = [Departamento o Sucursal].Sucursal\_id) " & \_

"WHERE ((([Departamento o Sucursal].Global)=True) AND ((CatAuxMesCie.Llave)='" & CuentaResultado & "') AND ((CatAuxMesCie.Region\_id)=" & RCAreas!Region\_id & "))"

Set RCAux = CurrentDb().OpenRecordset(qry)

If RCAux.EOF Then

Actual = Actual & RCAreas!Region\_id & Chr(13) & Chr(10)

End If

RCAux.Close

RCAreas.MoveNext

Loop

RCAreas.MoveFirst

If Actual <> "" Then

Actual = "La cuenta [" & CuentaResultado & "] de Resultado del Ejercicio para la Póliza de Cierre " & Chr(10) & \_

"en las siguientes Áreas o Regiones no tiene relación en el catálogo contable con un Depto Global " & Chr(10) & \_

Actual

MsgBox Actual, vbInformation, "Cuenta de resultado Inexistente"

Exit Function

End If

Else

Set RCAreas = CurrentDb().OpenRecordset("SELECT Region\_Id FROM [Departamento o Sucursal] WHERE Region\_id=0 and Sucursal\_ID=0")

If RCAreas.EOF Then

MsgBox "No existen registro del Departamento CORPORATIVO en el catálogo de Departamento o Sucursal ", vbInformation, "No existe Departamento Corporativo"

RCAreas.Close

Exit Function

End If

End If

'Limpiar Cargos y Abonos del Catalogo actual

DoCmd.RunSQL ("UPDATE CatMesCie SET [CarExt]=0, [AboExt]=0, [CARPROCESO]=0, [ABOPROCESO]=0 WHERE [CARPROCESO]<>0 OR [ABOPROCESO]<>0")

DoCmd.RunSQL ("UPDATE CatAuxMesCie SET [CarExt]=0, [AboExt]=0, [CARPROCESO]=0, [ABOPROCESO]=0 WHERE [CARPROCESO]<>0 OR [ABOPROCESO]<>0")

Ceros = "0000000000000000"

**'Se agrego Ciclo de Regiones, por cada una se generará una Póliza**

**'<HDH><20110118>**

Do While Not RCAreas.EOF

**'Si el esquema contable es 2; Significa que todo el catalogo de cuentas es sucursalizable**

**'entonces solo se traen las cuentas de resultado de la region en curso**

If parms(12) = 2 Then

Set RC = CurrentDb().OpenRecordset("SELECT \* FROM c\_cuentas\_del\_resultado WHERE Region\_ID=" & RCAreas!Region\_id, dbOpenDynaset)

**'Obtenemos Sucursal\_ID Global por Region\_ID**

SucursalGlobal\_ID = DLookup("Sucursal\_id", "[Departamento o Sucursal]", "Region\_Id=" & RCAreas!Region\_id & " AND Global=true")

Else

qry = "SELECT \* FROM c\_cuentas\_del\_resultado WHERE (((Sucursalizable) = True) And ((Sucursal\_id) <> 0) And ((Region\_id) <> 0)) UNION " & \_

"SELECT \* FROM c\_cuentas\_del\_resultado WHERE (((Sucursalizable) = False) And ((Sucursal\_id) = 0) And ((Region\_id) = 0))"

Set RC = CurrentDb().OpenRecordset(qry, dbOpenDynaset)

**'Obtenemos Sucursal\_ID Global por Region\_ID**

SucursalGlobal\_ID = 0

End If

If Not RC.EOF Then

**'Cargar número consecutivo de poliza**

qry = "SELECT \* FROM ContPol" & Año & " WHERE " & \_

"[TIPO] = '" & TipoPol & "' AND " & \_

"Left([MES],3) = '" & Left(Mes, 3) & "'"

Set RCAux = CurrentDb().OpenRecordset(qry, dbOpenDynaset)

If Not RCAux.EOF Then

Poliza = TipoPol & Format(IIf(IsNull(RCAux("Contador")), 1, RCAux("Contador")), "000")

RCAux.Edit

RCAux("Contador") = RCAux("Contador") + 1

RCAux.Update

Else

Poliza = TipoPol & "001"

End If

RCAux.Close

Else

RC.Close

Exit Function

End If

Do While Not RC.EOF

**'Obtiene el consecutivo**

Cons = IIf(IsNull(DLookup("max(cons)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'")), 1, DLookup("max(cons)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'") + 1)

**'If Cons = 35 Then**

**' MsgBox "DETENTE"**

**'End If**

qry = "INSERT INTO Pol" & Mes & Año & " ( Referencia, Cons, Moneda, Fecha\_mov, Llave, " & \_

"Concepto, Importe, Documento, Clave, FecCap, Sector\_id, Sucursal\_id, Region\_id, " & \_

"Importe\_Cargo, Importe\_Abono ) VALUES('" & Poliza & "'," & \_

Cons & ",'" & RC("Moneda") & "',date(),'" & RC("Llave") & \_

"','Póliza de cierre de ejercicio'," & RC("Cargo") + RC("Abono") & \_

",'CIE','" & RC("Clave") & "',date()," & \_

"'" & RC("Sector\_id") & "'," & RC("Sucursal\_id") & "," & RC("Region\_id") & \_

"," & RC("Cargo") & "," & RC("Abono") & ")"

DoCmd.RunSQL (qry)

**'Actualiza Cargos y Abonos en el catalogo de cuentas**

Llave = RC("Llave")

NoDer = 6

NoIzq = 10

**'Actualizar la cuenta corporativa**

Call GraCarAboXFiltro(1, Llave, RC("Moneda"), RC("Clave"), RC("Cargo") + RC("Abono"), \_

RC("Region\_id"), RC("Sucursal\_id"), "00", intMes, "CatRep" & Año, "Cat" & Mes & Año, False)

**'Actualizar la cuenta Auxiliar en el catalogo auxiliar**

Call GraCarAboXFiltro(2, Llave, RC("Moneda"), RC("Clave"), RC("Cargo") + RC("Abono"), \_

RC("Region\_id"), RC("Sucursal\_id"), "00", intMes, "CatAuxRep" & Año, "CatAux" & Mes & Año, False)

If RC("Sucursal\_Id") <> 0 Then

**'Actualizar la cuenta Auxiliar en el catalogo auxiliar para toda la empresa**

Call GraCarAboXFiltro(2, Llave, RC("Moneda"), RC("Clave"), RC("Cargo") + RC("Abono"), \_

0, 0, "00", intMes, "CatAuxRep" & Año, "CatAux" & Mes & Año, False)

End If

**'Actualizar a nivel de SCta, SSCta, SSSCta**

For i = 1 To 3

Actual = Right(Left(Llave, NoIzq), 2)

If Actual <> "00" Then

Llave = Left(Llave, NoIzq) & Right(Ceros, NoDer)

**'Actualiza a los padres de la pirámide**

Call GraCarAboXFiltro(1, Llave, RC("Moneda"), RC("Clave"), RC("Cargo") + RC("Abono"), \_

RC("Region\_id"), RC("Sucursal\_id"), "00", intMes, "CatRep" & Año, "Cat" & Mes & Año, False)

End If

NoDer = NoDer + 2

NoIzq = NoIzq - 2

Next i

**'Actualizar a nivel de mayor**

Llave = Left(Llave, NoIzq) & Right(Ceros, NoDer)

**'Actualiza a los padres de la pirámide**

Call GraCarAboXFiltro(1, Llave, RC("Moneda"), RC("Clave"), RC("Cargo") + RC("Abono"), \_

RC("Region\_id"), RC("Sucursal\_id"), "00", intMes, "CatRep" & Año, "Cat" & Mes & Año, False)

RC.MoveNext

Loop

RC.Close

**' Obtenemos la Utilidad**

Utilidad = IIf(IsNull(DLookup("Ingreso", "c\_Ingreso\_PolizaCierre", "Region\_Id=" & RCAreas!Region\_id)), 0, DLookup("Ingreso", "c\_Ingreso\_PolizaCierre", "Region\_Id=" & RCAreas!Region\_id))

Utilidad = Utilidad + IIf(IsNull(DLookup("Egreso", "c\_Egreso\_PolizaCierre", "Region\_Id=" & RCAreas!Region\_id)), 0, DLookup("Egreso", "c\_Egreso\_PolizaCierre", "Region\_Id=" & RCAreas!Region\_id))

Cons = IIf(IsNull(DLookup("max(cons)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'")), 1, DLookup("max(cons)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'") + 1)

qry = "INSERT INTO Pol" & Mes & Año & " (Referencia, Cons, Moneda, Fecha\_mov, Llave, " & \_

"Concepto, Importe, Documento, Clave, FecCap, Sector\_id, Sucursal\_id, Region\_id, " & \_

"Importe\_Cargo, Importe\_Abono ) VALUES('" & Poliza & "'," & \_

Cons & ",'" & Moneda & "',date(),'" & CuentaResultado & \_

"','Póliza de cierre de ejercicio'," & Utilidad \* IIf(Utilidad < 0, -1, 1) & \_

",'CIE','" & IIf(Utilidad < 0, "A", "C") & "',date()," & \_

"'00'," & SucursalGlobal\_ID & "," & RCAreas!Region\_id & \_

"," & IIf(Utilidad < 0, 0, Utilidad) & "," & IIf(Utilidad < 0, Utilidad \* -1, 0) & ")"

DoCmd.RunSQL (qry)

**'Actualiza Cargos y Abonos en el catálogo de cuentas**

Llave = CuentaResultado

NoDer = 6

NoIzq = 10

**'Actualizar la cuenta corporativa**

Call GraCarAboXFiltro(1, Llave, Moneda, IIf(Utilidad < 0, "A", "C"), Utilidad \* IIf(Utilidad < 0, -1, 1), \_

RCAreas!Region\_id, SucursalGlobal\_ID, "00", intMes, "CatRep" & Año, "Cat" & Mes & Año, False)

**Pedro aquí se manda llamar un método, y se repite varias veces abajo, pero en todos es el mismo**

Sub GraCarAboXFiltro(Tipo As Byte, Llave As String, Moneda As String, Clave As String, Importe As Currency, Region\_id As Long, \_

Sucursal\_id As Long, Sector\_id As String, Mes As Integer, CatRep As String, catMes As String, Agregar As Boolean)

**'\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**'Programador: Hilda Díaz Hernandez**

**'Fecha: 10/06/2004**

**'Descripcion: Proceso que se encarga de aplicar los cargos y los abonos de las cuentas auxliares**

**' y corporativas a los catalogos contables del mes de proceso**

**'Tipo: Define si la cuenta es (1) - auxiliar(CatAuxMES) o (2) - corporativa(CATMES)**

**'\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Dim Criterio As String

Dim RcCatMes As Recordset

Dim RcCatRep As Recordset

On Error GoTo GraCarAboxfiltroError

Criterio = "[LLAVE] = '" & Llave & "' AND [MONEDA] = '" & Moneda & "'"

If Tipo = 2 Then Criterio = Criterio & " AND [SUCURSAL\_ID] = " & Sucursal\_id & " AND [SECTOR\_ID] = '" & Sector\_id & "'"

**'Abrir la tabla del catálogo del mes que corresponda**

Set RcCatMes = CurrentDb().OpenRecordset("SELECT \* FROM " & catMes & " WHERE " & Criterio, dbOpenDynaset)

If Not RcCatMes.EOF Then

**'Grabar el correspondiente Cargo o Abono**

RcCatMes.Edit

**'Grabar registro para el reproceso?**

If Agregar Then

Set RcCatRep = CurrentDb().OpenRecordset("SELECT \* FROM " & CatRep & " WHERE " & \_

Criterio & " AND [FecCap]=" & Mes, dbOpenDynaset)

If RcCatRep.EOF Then

'No existe agregar

RcCatRep.AddNew

RcCatRep("Llave") = Llave

RcCatRep("Moneda") = Moneda

RcCatRep("SAct") = 0

RcCatRep("FecCap") = Mes

If Tipo = 1 Then

RcCatRep("Niv") = 0

Else

RcCatRep("Region\_Id") = Region\_id

RcCatRep("Sucursal\_Id") = Sucursal\_id

RcCatRep("Sector\_Id") = Sector\_id

End If

Else

**'Existe?, editar**

RcCatRep.Edit

End If

**'Si es Cargo lo sumo, Abono lo resto**

If Clave = "C" Then

RcCatRep("SAct") = RcCatRep("SAct") + Importe

Else

RcCatRep("SAct") = RcCatRep("SAct") - Importe

End If

RcCatRep.Update

RcCatRep.Close

End If

If Clave = "C" Then

**'Cargos del proceso por dia**

RcCatMes("CarExt") = RcCatMes("CarExt") + Importe

RcCatMes("CarProceso") = RcCatMes("CarProceso") + Importe

Else

**'Abonos extemporáneos del catalogo**

RcCatMes("AboExt") = RcCatMes("AboExt") + Importe

RcCatMes("AboProceso") = RcCatMes("AboProceso") + Importe

End If

RcCatMes.Update

End If

RcCatMes.Close

Salir:

Exit Sub

GraCarAboxfiltroError:

MsgBox Err.Description, vbInformation, "GraCarAbo1Error: " & Err.Number

Err.Clear

Resume Salir

End Sub

**'Actualizar la cuenta Auxiliar en el catálogo auxiliar**

Call GraCarAboXFiltro(2, Llave, Moneda, IIf(Utilidad < 0, "A", "C"), Utilidad \* IIf(Utilidad < 0, -1, 1), \_

RCAreas!Region\_id, SucursalGlobal\_ID, "00", intMes, "CatAuxRep" & Año, "CatAux" & Mes & Año, False)

If SucursalGlobal\_ID <> 0 Then

**'Actualizar la cuenta Auxiliar en el catálogo auxiliar para toda la empresa**

Call GraCarAboXFiltro(2, Llave, Moneda, IIf(Utilidad < 0, "A", "C"), Utilidad \* IIf(Utilidad < 0, -1, 1), \_

0, 0, "00", intMes, "CatAuxRep" & Año, "CatAux" & Mes & Año, False)

End If

**'Actualizar a nivel de SCta, SSCta, SSSCta**

For i = 1 To 3

Actual = Right(Left(Llave, NoIzq), 2)

If Actual <> "00" Then

Llave = Left(Llave, NoIzq) & Right(Ceros, NoDer)

**'Actualiza a los padres de la pirámide**

Call GraCarAboXFiltro(1, Llave, Moneda, IIf(Utilidad < 0, "A", "C"), Utilidad \* IIf(Utilidad < 0, -1, 1), \_

RCAreas!Region\_id, SucursalGlobal\_ID, "00", intMes, "CatRep" & Año, "Cat" & Mes & Año, False)

End If

NoDer = NoDer + 2

NoIzq = NoIzq - 2

Next i

**'Actualizar a nivel de mayor**

Llave = Left(Llave, NoIzq) & Right(Ceros, NoDer)

**'Actualiza a los padres de la pirámide**

Call GraCarAboXFiltro(1, Llave, Moneda, IIf(Utilidad < 0, "A", "C"), Utilidad \* IIf(Utilidad < 0, -1, 1), \_

RCAreas!Region\_id, SucursalGlobal\_ID, "00", intMes, "CatRep" & Año, "Cat" & Mes & Año, False)

**'Obtiene total de partidas**

TCons = IIf(IsNull(DLookup("count(\*)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'")), 0, DLookup("count(\*)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'"))

'Obtiene total los cargos

TCargos = IIf(IsNull(DLookup("sum(Importe\_Cargo)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'")), 0, DLookup("sum(Importe\_Cargo)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'"))

**'Obtiene total los Abonos**

Tabonos = IIf(IsNull(DLookup("sum(Importe\_Abono)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'")), 0, DLookup("sum(Importe\_Abono)", "Pol" & Mes & Año, "Referencia='" & Poliza & "'"))

qry = "INSERT INTO Mov" & Mes & Año & "(Referencia, TCons, Fecha\_mov, Fecha\_cap, " & \_

"Concepto, Cargos, Abonos, Usuario, TipoPoliza, Status, Mes\_Mov) VALUES('" & \_

Poliza & "'," & TCons & ", date(), date(), " & \_

"'Póliza de cierre de ejercicio'," & TCargos & "," & Tabonos & ",'" & \_

parametro(1) & "','" & TipoPol & "',2,'" & Mes & "')"

DoCmd.RunSQL (qry)

**'Ceros = "0000000000000000"**

**'NoDer = 6**

**'NoIzq = 10**

**'Llave = CuentaResultado**

**'Utilidad = Utilidad \* -1**

**'**

RCAreas.MoveNext

Loop

RCAreas.Close

**'Actualizar los Saldos definitivos en el Catálogo de cuentas**

DoCmd.RunSQL ("UPDATE Cat" & Mes & Año & " SET [SANTPROCESO]=[SACT],[SACT]=[SACT]+[CARPROCESO]-[ABOPROCESO]," & \_

"[SPROMANT]=[SPROM],[SPROM2ANT]=[SPROM2],[CAR]=[CAR]+[CARPROCESO]," & \_

"[ABO]=[ABO]+[ABOPROCESO] WHERE [CARPROCESO]<>0 OR [ABOPROCESO]<>0")

'Actualizar los Saldos definitivos del Catalogo Auxiliar

DoCmd.RunSQL ("UPDATE CatAux" & Mes & Año & " SET [SANTPROCESO]=[SACT],[SACT]=[SACT]+[CARPROCESO]-[ABOPROCESO]," & \_

"[SPROMANT]=[SPROM],[SPROM2ANT]=[SPROM2],[CAR]=[CAR]+[CARPROCESO]," & \_

"[ABO]=[ABO]+[ABOPROCESO] WHERE [CARPROCESO]<>0 OR [ABOPROCESO]<>0")

'Estos Querys se sacaron del ciclo de Regiones ya que Pone en CEROS a todas las cuentas de resultados de todo el catalogo

'Ejecuta Querys para poner en Ceros el catálogo contable

'DoCmd.OpenQuery ("c\_modifica\_CatAuxCie")

'DoCmd.OpenQuery ("c\_modifica\_CatCie")

'<HDH>FindeModificiación<20110118>

GeneraPolizaCierre = True

End Function

**'Borrar de la tabla de reproceso a la tabla de reproceso,**

**'las cuentas de IN o EG**

qry = "UPDATE Rangos, CatAuxRep" & Año & " SET SAct=0 " & \_

"WHERE (Naturaleza\_id='IN' Or Naturaleza\_id='EG') " & \_

"AND Left([Llave],4) " & \_

"Between [llave\_inicial] And [llave\_final];"

DoCmd.RunSQL qry

qry = "UPDATE Rangos, CatRep" & Año & " SET SAct=0 " & \_

"WHERE (Naturaleza\_id='IN' Or Naturaleza\_id='EG') AND Left([Llave],4) " & \_

"Between [llave\_inicial] And [llave\_final];"

DoCmd.RunSQL qry

End If

RCAux.Close

End If

End If

Next intMes

RcAños.MoveNext

Loop

RC.Close

RcAños.Close

**'Borrar todas las actualizaciones del proceso anterior**

Call DelBitacora(CtrlCarAbo)

Call DelBitacora(CatRep)

Call DelBitacora(CatAuxRep)

**'############################################################**

**'Proceso finalizado**

Beep

MsgBox "Operaciones realizadas con éxito!", vbInformation

ExitActSaldos:

Exit Function

ErrorActSaldos:

MsgBox Err.Description, vbCritical, "ActSaldos: " & Err.Number

Err.Clear

Resume ExitActSaldos

End Function