

Option Compare Database

' g stands for global

Public gRstPersonID As DAO.Recordset, gCurPersonBookmark As Variant

Public gRstPeople As DAO.Recordset, gRstAddresses As DAO.Recordset

Public gRstPeopleLookUp As DAO.Recordset, gRstBiogAddrType As DAO.Recordset

Public gRstEdge As DAO.Recordset, gRst As DAO.Recordset, gRstBiogADDR As DAO.Recordset

Public gRstAssocFilter As DAO.Recordset, gRstImportPeople As DAO.Recordset

Public gMaxNodeDist As Integer

Public gMaxFilterMilitary As Integer, gFilterMilitaryCount As Integer

Public gMaxFilterScholar As Integer, gFilterScholarCount As Integer

Public gMaxFilterTotal As Integer, gFilterTotalCount As Integer

Public gMaxFilterWritings As Integer, gFilterWritingsCount As Integer

Public gMaxFilterPolitics As Integer, gFilterPoliticsCount As Integer

Public gDisplayLanguage As String, gLabelsOK As Boolean, gImportPeople As Boolean, gImportPlaces As Boolean

Public gUsePersonID As Boolean, gUseADDRID As Boolean

Public gRstKin As DAO.Recordset, gRstNonKin As DAO.Recordset

Public gQuerySelectKin As String, gQuerySelectNonkin As String, gQueryIndexYear As String

Public gQuerySelectKinADDR As String, gQuerySelectNonkinADDR As String

Public gQuerySelectNonkinADDRFiltered As String, gQuerySelectNonkinFiltered As String

Public gQueryKindist As String, gQueryAssocFilter As String

Public gQueryBaseKin As String, gQueryBase As String, gQueryBaseNonkin As String

Public gQueryStr As String, gLoopMax As Integer, gUseFilter As Integer

Public gQueryKinAddrFilter As String, gQueryNonKinAddrFilter As String

Public gFromDynasty As Integer, gToDynasty As Integer, gUseIndexYears As Boolean, gUseDynasties As Boolean, gFromDynastyBegin As Integer, gFromDynastyEnd As Integer, gToDynastyBegin As Integer, gToDynastyEnd As Integer

Private Sub ChkFamily_Click()

' -1 (true) and 0 (false)

If ChkFamily.Value = -1 Then

gFilterTotalCount = gFilterTotalCount + 1

If gFilterTotalCount = gMaxFilterTotal Then

ChkSelectAll.Value = -1

End If

Else

gFilterTotalCount = gFilterTotalCount - 1

ChkSelectAll.Value = 0

End If

Call CheckRunCriteria

End Sub

Private Sub ChkFinance_Click()

' -1 (true) and 0 (false)

If ChkFinance.Value = -1 Then

gFilterTotalCount = gFilterTotalCount + 1

If gFilterTotalCount = gMaxFilterTotal Then

ChkSelectAll.Value = -1

End If

Else

gFilterTotalCount = gFilterTotalCount - 1

ChkSelectAll.Value = 0

End If

Call CheckRunCriteria

End Sub

Private Sub ChkFriendship_Click()

' -1 (true) and 0 (false)

If ChkFriendship.Value = -1 Then

gFilterTotalCount = gFilterTotalCount + 1

If gFilterTotalCount = gMaxFilterTotal Then

ChkSelectAll.Value = -1

End If

Else

gFilterTotalCount = gFilterTotalCount - 1

ChkSelectAll.Value = 0

End If

Call CheckRunCriteria

End Sub

Private Sub ChkIndexYear_Click()

TxtFrom.Enabled = ChkIndexYear.Value

TxtTo.Enabled = ChkIndexYear.Value

End Sub

End Sub

Private Sub ChkKin_Click()

```
If ChkKin.Value Then
    Me.ChkKinshipParam.Enabled = True
    TxtMaxUp.Enabled = True
    TxtMaxDwn.Enabled = True
    TxtMaxCol.Enabled = True
    TxtMaxMar.Enabled = True
Else
    Me.ChkKinshipParam.Enabled = False
    TxtMaxUp.Enabled = False
    TxtMaxDwn.Enabled = False
    TxtMaxCol.Enabled = False
    TxtMaxMar.Enabled = False
End If
Call CheckRunCriteria
End Sub
```

Private Sub ChkMedicine_Click()

```
' -1 (true) and 0 (false)

If ChkMedicine.Value = -1 Then
    gFilterTotalCount = gFilterTotalCount + 1
    If gFilterTotalCount = gMaxFilterTotal Then
        ChkSelectAll.Value = -1
    End If
Else
    gFilterTotalCount = gFilterTotalCount - 1
    ChkSelectAll.Value = 0
End If
Call CheckRunCriteria
```

End Sub

Private Sub ChkMilitaryAll_Click()

```
Dim tDelta As Integer
' -1 (true) and 0 (false)

tDelta = gMaxFilterMilitary - gFilterMilitaryCount
If ChkMilitaryAll.Value = -1 Then
    gFilterMilitaryCount = gMaxFilterMilitary
    gFilterTotalCount = gFilterTotalCount + tDelta

    ChkMilitaryOppose.Value = -1
    ChkMilitarySupport.Value = -1

    If gFilterTotalCount = gMaxFilterTotal Then
        ChkSelectAll.Value = -1
    End If
Else
    gFilterTotalCount = gFilterTotalCount - gFilterMilitaryCount
    gFilterMilitaryCount = 0

    ChkMilitaryOppose.Value = 0
    ChkMilitarySupport.Value = 0

    ChkSelectAll.Value = 0
End If
Call CheckRunCriteria
```

End Sub

Private Sub ChkMilitaryOppose_Click()

```
' -1 (true) and 0 (false)

If ChkMilitaryOppose.Value = -1 Then
    gFilterMilitaryCount = gFilterMilitaryCount + 1
    gFilterTotalCount = gFilterTotalCount + 1
    If gFilterMilitaryCount = gMaxFilterMilitary Then
        ChkMilitaryAll.Value = -1
    End If
    If gFilterTotalCount = gMaxFilterTotal Then
        ChkSelectAll.Value = -1
    End If
Else
    gFilterMilitaryCount = gFilterMilitaryCount - 1
    gFilterTotalCount = gFilterTotalCount - 1
```

End Sub

```
        ChkMilitaryAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
```

End Sub

```
Private Sub ChkMilitarySupport_Click()
    ' -1 (true) and 0 (false)
```

```
    If ChkMilitarySupport.Value = -1 Then
        gFilterMilitaryCount = gFilterMilitaryCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterMilitaryCount = gMaxFilterMilitary Then
            ChkMilitaryAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterMilitaryCount = gFilterMilitaryCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkMilitaryAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
```

End Sub

```
Private Sub ChkNonKin_Click()
```

```
    If ChkNonKin.Value = -1 Then

        ChkSelectAll.Enabled = True

        ChkMilitaryAll.Enabled = True
        ChkMilitaryOppose.Enabled = True
        ChkMilitarySupport.Enabled = True
```

```
        ChkPoliticsAll.Enabled = True
        ChkPolEqual.Enabled = True
        ChkPolOppose.Enabled = True
        ChkPolSponsor.Enabled = True
        ChkPolSub.Enabled = True
        ChkPolSup.Enabled = True
        ChkPolSupport.Enabled = True
```

```
        ChkScholarshipAll.Enabled = True
        ChkSchTeacher.Enabled = True
        ChkSchAffiliation.Enabled = True
        ChkSchAttack.Enabled = True
        ChkSchLitArt.Enabled = True
        ChkSchMember.Enabled = True
        ChkSchPatron.Enabled = True
        ChkSchTopic.Enabled = True
```

```
        ChkWritingsAll.Enabled = True
        ChkWriBiog.Enabled = True
        ChkWriCommem.Enabled = True
        ChkWriEpitaph.Enabled = True
        ChkWriExplain.Enabled = True
        ChkWriLetters.Enabled = True
        ChkWriMottos.Enabled = True
        ChkWriOccasion.Enabled = True
        ChkWriPreface.Enabled = True
        ChkWriRitual.Enabled = True
```

```
        ChkFamily.Enabled = True
        ChkFinance.Enabled = True
        ChkFriendship.Enabled = True
        ChkMedicine.Enabled = True
        ChkReligion.Enabled = True
```

```
    Else
```

```
        ChkSelectAll.Enabled = False

        ChkMilitaryAll.Enabled = False
        ChkMilitaryOppose.Enabled = False
        ChkMilitarySupport.Enabled = False

        ChkPoliticsAll.Enabled = False
```

```

    ChkPolEqual.Enabled = False
    ChkPolOppose.Enabled = False
    ChkPolSponsor.Enabled = False
    ChkPolSub.Enabled = False
    ChkPolSup.Enabled = False
    ChkPolSupport.Enabled = False

    ChkScholarshipAll.Enabled = False
    ChkSchTeacher.Enabled = False
    ChkSchAffiliation.Enabled = False
    ChkSchAttack.Enabled = False
    ChkSchLitArt.Enabled = False
    ChkSchMember.Enabled = False
    ChkSchPatron.Enabled = False
    ChkSchTopic.Enabled = False

```

```

    ChkWritingsAll.Enabled = False
    ChkWriBiog.Enabled = False
    ChkWriCommem.Enabled = False
    ChkWriEpitaph.Enabled = False
    ChkWriExplain.Enabled = False
    ChkWriLetters.Enabled = False
    ChkWriMottos.Enabled = False
    ChkWriOccasion.Enabled = False
    ChkWriPreface.Enabled = False
    ChkWriRitual.Enabled = False

```

```

    ChkFamily.Enabled = False
    ChkFinance.Enabled = False
    ChkFriendship.Enabled = False
    ChkMedicine.Enabled = False
    ChkReligion.Enabled = False

```

```

End If
Call CheckRunCriteria

```

```

End Sub

```

```

Private Sub chkPolEqual_Click()
    ' -1 (true) and 0 (false)

```

```

    If ChkPolEqual.Value = -1 Then
        gFilterPoliticsCount = gFilterPoliticsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterPoliticsCount = gMaxFilterPolitics Then
            ChkPoliticsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterPoliticsCount = gFilterPoliticsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkPoliticsAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria

```

```

End Sub

```

```

Private Sub ChkPoliticsAll_Click()
    Dim tDelta As Integer
    ' -1 (true) and 0 (false)

```

```

    tDelta = gMaxFilterPolitics - gFilterPoliticsCount
    If ChkPoliticsAll.Value = -1 Then
        gFilterPoliticsCount = gMaxFilterPolitics
        gFilterTotalCount = gFilterTotalCount + tDelta

        ChkPolEqual.Value = -1
        ChkPolOppose.Value = -1
        ChkPolSponsor.Value = -1
        ChkPolSub.Value = -1
        ChkPolSup.Value = -1
        ChkPolSupport.Value = -1

```

```

        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else

```

```


```

```

5
gFilterTotalCount = gFilterTotalCount - gFilterPoliticsCount
gFilterPoliticsCount = 0

ChkPolEqual.Value = 0
ChkPolOppose.Value = 0
ChkPolSponsor.Value = 0
ChkPolSub.Value = 0
ChkPolSup.Value = 0
ChkPolSupport.Value = 0

ChkSelectAll.Value = 0
End If
Call CheckRunCriteria

End Sub

Private Sub ChkPolOppose_Click()
' -1 (true) and 0 (false)

If ChkPolOppose.Value = -1 Then
gFilterPoliticsCount = gFilterPoliticsCount + 1
gFilterTotalCount = gFilterTotalCount + 1
If gFilterPoliticsCount = gMaxFilterPolitics Then
ChkPoliticsAll.Value = -1
End If
If gFilterTotalCount = gMaxFilterTotal Then
ChkSelectAll.Value = -1
End If
Else
gFilterPoliticsCount = gFilterPoliticsCount - 1
gFilterTotalCount = gFilterTotalCount - 1
ChkPoliticsAll.Value = 0
ChkSelectAll.Value = 0
End If
Call CheckRunCriteria

End Sub

Private Sub ChkPolSponsor_Click()
' -1 (true) and 0 (false)

If ChkPolSponsor.Value = -1 Then
gFilterPoliticsCount = gFilterPoliticsCount + 1
gFilterTotalCount = gFilterTotalCount + 1
If gFilterPoliticsCount = gMaxFilterPolitics Then
ChkPoliticsAll.Value = -1
End If
If gFilterTotalCount = gMaxFilterTotal Then
ChkSelectAll.Value = -1
End If
Else
gFilterPoliticsCount = gFilterPoliticsCount - 1
gFilterTotalCount = gFilterTotalCount - 1
ChkPoliticsAll.Value = 0
ChkSelectAll.Value = 0
End If
Call CheckRunCriteria

End Sub

Private Sub ChkPolSub_Click()
' -1 (true) and 0 (false)

If ChkPolSub.Value = -1 Then
gFilterPoliticsCount = gFilterPoliticsCount + 1
gFilterTotalCount = gFilterTotalCount + 1
If gFilterPoliticsCount = gMaxFilterPolitics Then
ChkPoliticsAll.Value = -1
End If
If gFilterTotalCount = gMaxFilterTotal Then
ChkSelectAll.Value = -1
End If
Else
gFilterPoliticsCount = gFilterPoliticsCount - 1
gFilterTotalCount = gFilterTotalCount - 1
ChkPoliticsAll.Value = 0
ChkSelectAll.Value = 0
End If
Call CheckRunCriteria

```

End Sub

End Sub

```
Private Sub ChkPolSup_Click()  
    ' -1 (true) and 0 (false)  
  
    If ChkPolSup.Value = -1 Then  
        gFilterPoliticsCount = gFilterPoliticsCount + 1  
        gFilterTotalCount = gFilterTotalCount + 1  
        If gFilterPoliticsCount = gMaxFilterPolitics Then  
            ChkPoliticsAll.Value = -1  
        End If  
        If gFilterTotalCount = gMaxFilterTotal Then  
            ChkSelectAll.Value = -1  
        End If  
    Else  
        gFilterPoliticsCount = gFilterPoliticsCount - 1  
        gFilterTotalCount = gFilterTotalCount - 1  
        ChkPoliticsAll.Value = 0  
        ChkSelectAll.Value = 0  
    End If  
    Call CheckRunCriteria  
End Sub
```

End Sub

```
Private Sub ChkPolSupport_Click()  
    ' -1 (true) and 0 (false)  
  
    If ChkPolSupport.Value = -1 Then  
        gFilterPoliticsCount = gFilterPoliticsCount + 1  
        gFilterTotalCount = gFilterTotalCount + 1  
        If gFilterPoliticsCount = gMaxFilterPolitics Then  
            ChkPoliticsAll.Value = -1  
        End If  
        If gFilterTotalCount = gMaxFilterTotal Then  
            ChkSelectAll.Value = -1  
        End If  
    Else  
        gFilterPoliticsCount = gFilterPoliticsCount - 1  
        gFilterTotalCount = gFilterTotalCount - 1  
        ChkPoliticsAll.Value = 0  
        ChkSelectAll.Value = 0  
    End If  
    Call CheckRunCriteria  
End Sub
```

End Sub

```
Private Sub ChkReligion_Click()  
    ' -1 (true) and 0 (false)  
  
    If ChkReligion.Value = -1 Then  
        gFilterTotalCount = gFilterTotalCount + 1  
        If gFilterTotalCount = gMaxFilterTotal Then  
            ChkSelectAll.Value = -1  
        End If  
    Else  
        gFilterTotalCount = gFilterTotalCount - 1  
        ChkSelectAll.Value = 0  
    End If  
    Call CheckRunCriteria  
End Sub
```

End Sub

```
Private Sub ChkSchAffiliation_Click()  
    ' -1 (true) and 0 (false)  
  
    If ChkSchAffiliation.Value = -1 Then  
        gFilterScholarCount = gFilterScholarCount + 1  
        gFilterTotalCount = gFilterTotalCount + 1  
        If gFilterScholarCount = gMaxFilterScholar Then  
            ChkScholarshipAll.Value = -1  
        End If  
        If gFilterTotalCount = gMaxFilterTotal Then  
            ChkSelectAll.Value = -1  
        End If  
    Else  
        gFilterScholarCount = gFilterScholarCount - 1  
        gFilterTotalCount = gFilterTotalCount - 1  
        ChkScholarshipAll.Value = 0  
        ChkSelectAll.Value = 0  
    End If  
End Sub
```

```
End If
Call CheckRunCriteria
```

```
End Sub
```

```
Private Sub ChkSchAttack_Click()
```

```
    ' -1 (true) and 0 (false)

    If ChkSchAttack.Value = -1 Then
        gFilterScholarCount = gFilterScholarCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterScholarCount = gMaxFilterScholar Then
            ChkScholarshipAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterScholarCount = gFilterScholarCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkScholarshipAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
```

```
End Sub
```

```
Private Sub ChkSchLitArt_Click()
```

```
    ' -1 (true) and 0 (false)

    If ChkSchLitArt.Value = -1 Then
        gFilterScholarCount = gFilterScholarCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterScholarCount = gMaxFilterScholar Then
            ChkScholarshipAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterScholarCount = gFilterScholarCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkScholarshipAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
```

```
End Sub
```

```
Private Sub ChkSchMember_Click()
```

```
    ' -1 (true) and 0 (false)

    If ChkSchMember.Value = -1 Then
        gFilterScholarCount = gFilterScholarCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterScholarCount = gMaxFilterScholar Then
            ChkScholarshipAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterScholarCount = gFilterScholarCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkScholarshipAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
```

```
End Sub
```

```
Private Sub ChkScholarshipAll_Click()
```

```
    Dim tDelta As Integer
    ' -1 (true) and 0 (false)

    If ChkScholarshipAll.Value = -1 Then
        tDelta = gMaxFilterScholar - gFilterScholarCount
```

```

gFilterScholarCount = gMaxFilterScholar
gFilterTotalCount = gFilterTotalCount + tDelta

```

```

ChkSchTeacher.Value = -1
ChkSchAffiliation.Value = -1
ChkSchAttack.Value = -1
ChkSchLitArt.Value = -1
ChkSchMember.Value = -1
ChkSchPatron.Value = -1
ChkSchTopic.Value = -1

```

```

If gFilterTotalCount = gMaxFilterTotal Then
    ChkSelectAll.Value = -1
End If

```

```

Else
    gFilterTotalCount = gFilterTotalCount - gFilterScholarCount
    gFilterScholarCount = 0

```

```

    ChkSchTeacher.Value = 0
    ChkSchAffiliation.Value = 0
    ChkSchAttack.Value = 0
    ChkSchLitArt.Value = 0
    ChkSchMember.Value = 0
    ChkSchPatron.Value = 0
    ChkSchTopic.Value = 0

```

```

    ChkSelectAll.Value = 0

```

```

End If
Call CheckRunCriteria

```

```

End Sub

```

```

Private Sub ChkSchPatron_Click()

```

```

    ' -1 (true) and 0 (false)

```

```

If ChkSchPatron.Value = -1 Then
    gFilterScholarCount = gFilterScholarCount + 1
    gFilterTotalCount = gFilterTotalCount + 1
    If gFilterScholarCount = gMaxFilterScholar Then
        ChkScholarshipAll.Value = -1
    End If
    If gFilterTotalCount = gMaxFilterTotal Then
        ChkSelectAll.Value = -1
    End If

```

```

Else
    gFilterScholarCount = gFilterScholarCount - 1
    gFilterTotalCount = gFilterTotalCount - 1
    ChkScholarshipAll.Value = 0
    ChkSelectAll.Value = 0
End If
Call CheckRunCriteria

```

```

End Sub

```

```

Private Sub ChkSchTeacher_Click()

```

```

    ' -1 (true) and 0 (false)

```

```

If ChkSchTeacher.Value = -1 Then
    gFilterScholarCount = gFilterScholarCount + 1
    gFilterTotalCount = gFilterTotalCount + 1
    If gFilterScholarCount = gMaxFilterScholar Then
        ChkScholarshipAll.Value = -1
    End If
    If gFilterTotalCount = gMaxFilterTotal Then
        ChkSelectAll.Value = -1
    End If

```

```

Else
    gFilterScholarCount = gFilterScholarCount - 1
    gFilterTotalCount = gFilterTotalCount - 1
    ChkScholarshipAll.Value = 0
    ChkSelectAll.Value = 0
End If
Call CheckRunCriteria

```

```

End Sub

```

```

Private Sub ChkSchTopic_Click()

```

```

    ' -1 (true) and 0 (false)

```



```

If ChkSchTopic.Value = -1 Then
    gFilterScholarCount = gFilterScholarCount + 1
    gFilterTotalCount = gFilterTotalCount + 1
    If gFilterScholarCount = gMaxFilterScholar Then
        ChkScholarshipAll.Value = -1
    End If
    If gFilterTotalCount = gMaxFilterTotal Then
        ChkSelectAll.Value = -1
    End If
Else
    gFilterScholarCount = gFilterScholarCount - 1
    gFilterTotalCount = gFilterTotalCount - 1
    ChkScholarshipAll.Value = 0
    ChkSelectAll.Value = 0
End If
Call CheckRunCriteria
End Sub

```

```

Private Sub ChkSelectAll_Click()
    ' -1 (true) and 0 (false)

```

```

If ChkSelectAll.Value = -1 Then
    gFilterTotalCount = gMaxFilterTotal
    gFilterPoliticsCount = gMaxFilterPolitics
    gFilterScholarCount = gMaxFilterScholar
    gFilterWritingsCount = gMaxFilterWritings
    gFilterMilitaryCount = gMaxFilterMilitary

    ChkMilitaryAll.Value = -1
    ChkMilitaryOppose.Value = -1
    ChkMilitarySupport.Value = -1

    ChkPoliticsAll.Value = -1
    ChkPolEqual.Value = -1
    ChkPolOppose.Value = -1
    ChkPolSponsor.Value = -1
    ChkPolSub.Value = -1
    ChkPolSup.Value = -1
    ChkPolSupport.Value = -1

    ChkScholarshipAll.Value = -1
    ChkSchTeacher.Value = -1
    ChkSchAffiliation.Value = -1
    ChkSchAttack.Value = -1
    ChkSchLitArt.Value = -1
    ChkSchMember.Value = -1
    ChkSchPatron.Value = -1
    ChkSchTopic.Value = -1

    ChkWritingsAll.Value = -1
    ChkWriBiog.Value = -1
    ChkWriCommem.Value = -1
    ChkWriEpitaph.Value = -1
    ChkWriExplain.Value = -1
    ChkWriLetters.Value = -1
    ChkWriMottos.Value = -1
    ChkWriOccasion.Value = -1
    ChkWriPreface.Value = -1
    ChkWriRitual.Value = -1

    ChkFamily.Value = -1
    ChkFinance.Value = -1
    ChkFriendship.Value = -1
    ChkMedicine.Value = -1
    ChkReligion.Value = -1

```

```

Else
    gFilterTotalCount = 0
    gFilterWritingsCount = 0
    gFilterScholarCount = 0
    gFilterPoliticsCount = 0
    gFilterMilitaryCount = 0

    'ChkNonKin.Value = 0

    ChkMilitaryAll.Value = 0
    ChkMilitaryOppose.Value = 0
    ChkMilitarySupport.Value = 0

```

```

    ChkPoliticsAll.Value = 0
    ChkPolEqual.Value = 0
    ChkPolOppose.Value = 0
    ChkPolSponsor.Value = 0
    ChkPolSub.Value = 0
    ChkPolSup.Value = 0
    ChkPolSupport.Value = 0

    ChkScholarshipAll.Value = 0
    ChkSchTeacher.Value = 0
    ChkSchAffiliation.Value = 0
    ChkSchAttack.Value = 0
    ChkSchLitArt.Value = 0
    ChkSchMember.Value = 0
    ChkSchPatron.Value = 0
    ChkSchTopic.Value = 0

    ChkWritingsAll.Value = 0
    ChkWriBiog.Value = 0
    ChkWriCommem.Value = 0
    ChkWriEpitaph.Value = 0
    ChkWriExplain.Value = 0
    ChkWriLetters.Value = 0
    ChkWriMottos.Value = 0
    ChkWriOccasion.Value = 0
    ChkWriPreface.Value = 0
    ChkWriRitual.Value = 0

    ChkFamily.Value = 0
    ChkFinance.Value = 0
    ChkFriendship.Value = 0
    ChkMedicine.Value = 0
    ChkReligion.Value = 0

```

```

End If
Call CheckRunCriteria

```

End Sub

```

Private Sub ChkWriBiog_Click()
    ' -1 (true) and 0 (false)

    If ChkWriBiog.Value = -1 Then
        gFilterWritingsCount = gFilterWritingsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterWritingsCount = gMaxFilterWritings Then
            ChkWritingsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterWritingsCount = gFilterWritingsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkWritingsAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria

```

End Sub

```

Private Sub ChkWriCommem_Click()
    ' -1 (true) and 0 (false)

    If ChkWriCommem.Value = -1 Then
        gFilterWritingsCount = gFilterWritingsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterWritingsCount = gMaxFilterWritings Then
            ChkWritingsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterWritingsCount = gFilterWritingsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkWritingsAll.Value = 0
        ChkSelectAll.Value = 0
    End If

```

```

Call CheckRunCriteria

End Sub

Private Sub ChkWriEpitaph_Click()
    ' -1 (true) and 0 (false)

    If ChkWriEpitaph.Value = -1 Then
        gFilterWritingsCount = gFilterWritingsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterWritingsCount = gMaxFilterWritings Then
            ChkWritingsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterWritingsCount = gFilterWritingsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkWritingsAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
End Sub

Private Sub ChkWriExplain_Click()
    ' -1 (true) and 0 (false)

    If ChkWriExplain.Value = -1 Then
        gFilterWritingsCount = gFilterWritingsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterWritingsCount = gMaxFilterWritings Then
            ChkWritingsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterWritingsCount = gFilterWritingsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkWritingsAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
End Sub

Private Sub ChkWriLetters_Click()
    ' -1 (true) and 0 (false)

    If ChkWriLetters.Value = -1 Then
        gFilterWritingsCount = gFilterWritingsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterWritingsCount = gMaxFilterWritings Then
            ChkWritingsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterWritingsCount = gFilterWritingsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkWritingsAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
End Sub

Private Sub ChkWriMottos_Click()
    ' -1 (true) and 0 (false)

    If ChkWriMottos.Value = -1 Then
        gFilterWritingsCount = gFilterWritingsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterWritingsCount = gMaxFilterWritings Then
            ChkWritingsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterWritingsCount = gFilterWritingsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkWritingsAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
End Sub

```

```

        ChkSelectAll.Value = -1
    End If
Else
    gFilterWritingsCount = gFilterWritingsCount - 1
    gFilterTotalCount = gFilterTotalCount - 1
    ChkWritingsAll.Value = 0
    ChkSelectAll.Value = 0
End If
Call CheckRunCriteria
End Sub

```

```

Private Sub ChkWriOccasion_Click()
    ' -1 (true) and 0 (false)

    If ChkWriOccasion.Value = -1 Then
        gFilterWritingsCount = gFilterWritingsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterWritingsCount = gMaxFilterWritings Then
            ChkWritingsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterWritingsCount = gFilterWritingsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkWritingsAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
End Sub

```

```

Private Sub ChkWriPreface_Click()
    ' -1 (true) and 0 (false)

    If ChkWriPreface.Value = -1 Then
        gFilterWritingsCount = gFilterWritingsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterWritingsCount = gMaxFilterWritings Then
            ChkWritingsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterWritingsCount = gFilterWritingsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkWritingsAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
End Sub

```

```

Private Sub ChkWriRitual_Click()
    ' -1 (true) and 0 (false)

    If ChkWriRitual.Value = -1 Then
        gFilterWritingsCount = gFilterWritingsCount + 1
        gFilterTotalCount = gFilterTotalCount + 1
        If gFilterWritingsCount = gMaxFilterWritings Then
            ChkWritingsAll.Value = -1
        End If
        If gFilterTotalCount = gMaxFilterTotal Then
            ChkSelectAll.Value = -1
        End If
    Else
        gFilterWritingsCount = gFilterWritingsCount - 1
        gFilterTotalCount = gFilterTotalCount - 1
        ChkWritingsAll.Value = 0
        ChkSelectAll.Value = 0
    End If
    Call CheckRunCriteria
End Sub

```

```

Private Sub ChkWritingsAll_Click()
    Dim tDelta As Integer

```

' -1 (true) and 0 (false)

If ChkWritingsAll.Value = -1 Then

tDelta = gMaxFilterWritings - gFilterWritingsCount
gFilterWritingsCount = gMaxFilterWritings
gFilterTotalCount = gFilterTotalCount + tDelta

ChkWriBiog.Value = -1
ChkWriCommem.Value = -1
ChkWriEpitaph.Value = -1
ChkWriExplain.Value = -1
ChkWriLetters.Value = -1
ChkWriMottos.Value = -1
ChkWriOccasion.Value = -1
ChkWriPreface.Value = -1
ChkWriRitual.Value = -1

If gFilterTotalCount = gMaxFilterTotal Then

ChkSelectAll.Value = -1

End If

Else

gFilterTotalCount = gFilterTotalCount - gFilterWritingsCount
gFilterWritingsCount = 0

ChkWriBiog.Value = 0
ChkWriCommem.Value = 0
ChkWriEpitaph.Value = 0
ChkWriExplain.Value = 0
ChkWriLetters.Value = 0
ChkWriMottos.Value = 0
ChkWriOccasion.Value = 0
ChkWriPreface.Value = 0
ChkWriRitual.Value = 0

ChkSelectAll.Value = 0

End If

Call CheckRunCriteria

End Sub

Private Sub CmdAllDynasties_Click()

gFromDynasty = -2
gToDynasty = -2
TxtFromDynasty.Value = ""
TxtFromDynastyPY.Value = "All"
TxtToDynasty.Value = ""
TxtToDynastyPY.Value = "All"

End Sub

Private Sub CmdAllPeople_Click()

On Error GoTo Err_CmdAllPeople_Click

TxtPersonID.Value = -1
TxtNameChn.Value = ""
TxtName.Value = ""
gUsePersonID = False
CmdAllPeople.Enabled = False
'CmdRerun.Enabled = False
Call CheckRunCriteria

Exit_CmdAllPeople_Click:

Exit Sub

Err_CmdAllPeople_Click:

MsgBox Err.Description
Resume Exit_CmdAllPeople_Click

End Sub

Private Sub CmdAllPlaces_Click()

On Error GoTo Err_CmdAllPlaces_Click

TxtAddrID.Value = -1

TxtPlaceChn.Value = ""
TxtPlace.Value = ""
gUseADDRID = False
ChkXYRef.Enabled = False
ChkSubUnits.Enabled = False

```

Me.ChkPlaceLimit.Enabled = False
Call CheckRunCriteria

Exit_CmdAllPlaces_Click:
Exit Sub

Err_CmdAllPlaces_Click:
MsgBox Err.Description
Resume Exit_CmdAllPlaces_Click

End Sub

Private Sub CmdClose_Click()
On Error GoTo Err_CmdClose_Click

DoCmd.Close

Exit_CmdClose_Click:
Exit Sub

Err_CmdClose_Click:
MsgBox Err.Description
Resume Exit_CmdClose_Click

End Sub

Private Sub CmdFantiDisplay_Click()
On Error GoTo Err_CmdFantiDisplay_Click
If gDisplayLanguage = "T" Then
    gDisplayLanguage = "E"
Else
    gDisplayLanguage = "T"
End If

Call changeDisplayLanguage

Exit_CmdFantiDisplay_Click:
Exit Sub

Err_CmdFantiDisplay_Click:
MsgBox Err.Description
Resume Exit_CmdFantiDisplay_Click

End Sub

Private Sub CmdFromDynasty_Click()
Dim stDocName As String
Dim stLinkCriteria As String
Dim strFromDynasty As String

If gFromDynasty < 0 Then
    strFromDynasty = ""
Else
    strFromDynasty = Str(gFromDynasty)
End If

stDocName = "frmPickDynasty"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

    Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
    gFromDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
    gFromDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
    gFromDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
    '
    ' check to see if we have a problem and reject selection
    '
    If gToDynasty > -1 Then
        If gFromDynastyBegin > gToDynastyEnd Then
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' D
ynasty ends!", vbExclamation
            gFromDynasty = -1
            TxtFromDynasty.Value = ""

```

```

15
        TxtFromDynastyPY.Value = ""
    End If
End If
'
' value is OK
'
If gFromDynasty > -1 Then
    Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
    TxtFromDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
    TxtFromDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
End If

DoCmd.Close acForm, stDocName
'
' reset ToDynasty if necessary (-2 = all dynasties)
'
If gToDynasty = -2 Then
    gToDynasty = -1
    TxtToDynastyPY.Value = ""
    TxtToDynastyPY.Value = ""
End If
'
End If

```

End Sub

```

Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call writeKML
    Exit Sub
End If
'
' This program will dump the results to a .gis file
'
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGIS_Click
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
Else
    tStream.Charset = "gb2312"
    tCodeStr = "GB2312"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
'
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
    tFileName = tFN
    If Not tFileName = "" Then
        Exit For
    End If

```

```

Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdGIS_Click
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".tab"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
        tFileName = tFileName + ".tab"
    End If
End If
'
' write the file
'
'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
'
' process the table
'
Set tRstNode = ZZ_SCRATCH_PEOPLE.Form.Recordset
tC = Chr(9) ' the tab
'
With tRstNode
    ' write the header
    '
    tStr = "Name" + tC + "NameChn" + tC + "Female" + tC + "IndexYear" + tC
    tStr = tStr + "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC
    tStr = tStr + "xy_count" + tC + "NodeDist"
    tStream.WriteText tStr, adWriteLine
    .MoveFirst
    Do While Not .EOF
        If !c_female = -1 Then
            tFemale = "F"
        Else
            tFemale = "M"
        End If
        ' must guard against NULLs, even where there should not be any
        '
        If IsNull(!c_name) Then
            tStr = "[Bad Data]" + tC
        Else
            tStr = !c_name + tC
        End If

        If IsNull(!c_name_chn) Then
            tStr = tStr + "[Bad Data]" + tC
        Else
            If Trim(!c_name_chn) = "" Then
                tStr = tStr + "[?]" + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If
        End If

        tStr = tStr + tFemale + tC

        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Str(!c_index_year) + tC
        End If

        ' here guard against blanks as well
        If IsNull(!c_addr_name) Then
            tStr = tStr + "[?]" + tC
        ElseIf Trim(!c_addr_name) = "" Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_addr_name + tC
        End If

        If IsNull(!c_addr_chn) Then
            tStr = tStr + "[?]" + tC
        ElseIf Trim(!c_addr_chn) = "" Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_addr_chn + tC
        End If
    End While
End With

```



```

        If IsNull(!x_coord) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Str(!x_coord) + tC
        End If

        If IsNull(!y_coord) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Str(!y_coord) + tC
        End If

        If IsNull(!xy_count) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Str(!xy_count) + tC
        End If

        tStr = tStr + Str(!c_node_dist)

        tStream.WriteText tStr, adWriteLine
        .MoveNext
    Loop
End With

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGIS_Click:
Exit Sub

Err_CmdGIS_Click:
MsgBox Err.Description
Resume Exit_CmdGIS_Click

End Sub

Private Sub CmdGUESS_Click()
On Error GoTo Err_CmdGUESS_Click
'
' This program will dump the results of the search to a .gdf file
'
' for the moment I'll just describe the format of the .gdf file
'
' nodedef> name, label, labelvisible, style, pinyin VARCHAR(50), nodedist INT
'     name = str(c_person_id)
'     label = c_name_chn
'     style = 4 (text inside a rectangle)
'     pinyin = c_name
'     nodedist = c_node_dist INT
'     indexyear = c_index_year INT
'     sex = c_female > (F,M)
'
' edgedef> node1, node2, label, labelvisible, edge_desc VARCHAR(50)
'     node1 = str(c_person_id) for node1
'     node2 = str(c_node_id) for node2
'     label = c_link_chn
'     edge_desc = c_link_desc
'     edgetype= c_link_type (K,N)
'
' the central question is whether to do distance optimizations
'
' first see if there are any records to process
'
If ZZ_SOCIAL_NETWORK.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGUESS_Click

```

```

End If
'
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGUESS_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer
'Dim tFileSystem, tGDF

Dim tStream As ADODB.Stream, tCodeStr As String
Set tStream = New ADODB.Stream

If CodeFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf CodeFrame.Value = 2 Then
    tStream.Charset = "big5"
    tCodeStr = "BIG5"
ElseIf CodeFrame.Value = 3 Then
    tStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    tStream.Charset = "iso-8859-1"
    tCodeStr = "ASCII"
    tPinyin = True
End If

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network_" + tCodeStr + ".gdf"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdGUESS_Click
        Else
            ' make sure the file name has a gdf extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".gdf"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".gdf") Then
                tFileName = tFileName + ".gdf"
            End If
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
    'Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)
    tStream.Mode = adModeReadWrite
    tStream.Type = adTypeText
    tStream.Open

    ' process the two tables
    '
    Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
    Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
    tC = Chr(44) ' the comma
    tQuote = Chr(34) 'the Quote delimiter
    '
    ' first the nodes: define the record structure
    ' if ASCII, no characters, and no pinyin field
    If tCodeStr = "ASCII" Then
        tStr = "nodedef> name VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible BOOLEAN" + _

```

```

tC + "style INT" + tC + "nodedist INT" + _
tC + "indexyear INT" + tC + "dynasty_code INT" + tC + "dynasty VARCHAR" + tC + "sex VARCHAR(1
)" + _
tC + "addr_name VARCHAR" + tC + "latitude DOUBLE" + tC + "longitude DOUBLE"
Else
tStr = "nodedef> name VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible BOOLEAN" + _
tC + "style INT" + tC + "pinyin VARCHAR(50)" + tC + "nodedist INT" + _
tC + "indexyear INT" + tC + "dynasty_code INT" + tC + "dynasty VARCHAR" + tC + "dynasty_chn V
ARCHAR" + tC + "sex VARCHAR(1)" + _
tC + "addr_chn VARCHAR" + tC + "addr_name VARCHAR" + tC + "latitude DOUBLE" + tC + "longitude
DOUBLE"
End If
'MsgBox "Writing " + tStr
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
'
With tRstNode
.MoveFirst
Do While Not .EOF
' name = the ID of the person
tStr = Trim(Str(!c_person_id)) + tC

' label
If tCodeStr = "ASCII" Then
If IsNull(!c_name) Then
tStr = tStr + "[Missing]" + tC
Else
tStr = tStr + !c_name + tC
End If
Else
If IsNull(!c_name_chn) Then
tStr = tStr + tC
Else
tStr = tStr + !c_name_chn + tC
End If
End If

' labelvisible = true, style = 4 (text inside a rectangle)
tStr = tStr + "true" + tC + "4" + tC

If Not (tCodeStr = "ASCII") Then
' pinyin = c_name
tStr = tStr + !c_name + tC
End If

' nodedist = c_node_dist INT
tStr = tStr + Trim(Str(!c_node_dist)) + tC

' indexyear = c_index_year INT
If IsNull(!c_index_year) Then
tStr = tStr + "-2000" + tC
Else
tStr = tStr + Trim(Str(!c_index_year)) + tC
End If

' dynasty information
If tCodeStr = "ASCII" Then
If IsNull(!c_dynasty) Then
tStr = tStr + "0" + tC + "Unknown" + tC
Else
tStr = tStr + Str(!c_dy) + tC + !c_dynasty + tC
End If
Else
If IsNull(!c_dynasty) Then
tStr = tStr + "0" + tC + "Unknown" + tC + "Unknown" + tC
Else
tStr = tStr + Str(!c_dy) + tC + !c_dynasty + tC + !c_dynasty_chn + tC
End If
End If

' sex = c_female > (F,M)
If !c_female = -1 Then
tStr = tStr + "F" + tC
Else
tStr = tStr + "M" + tC
End If

' address names
If tCodeStr = "ASCII" Then
If IsNull(!c_addr_name) Then

```

```

        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If
Else
    If IsNull(!c_addr_chn) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_chn + tC
    End If
    If IsNull(!c_addr_name) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If
End If

'    latitude = !y_coord
If IsNull(!y_coord) Then
    tStr = tStr + "0.0" + tC
Else
    tStr = tStr + Str(!y_coord) + tC
End If

'    longitude = !x_coord
If IsNull(!x_coord) Then
    tStr = tStr + "0.0"
Else
    tStr = tStr + Str(!x_coord)
End If

'MsgBox "Writing " + tStr
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
.MoveNext
Loop
End With
'
' now the edges:  define the record structure
' if ASCII, then the description is the English and there is no edge_desc
If tCodeStr = "ASCII" Then
    tStr = "edgedef> node1 VARCHAR" + tC + "node2 VARCHAR" + tC + "label VARCHAR" + tC + "edgetype VA
RCHAR(1) "
Else
    tStr = "edgedef> node1 VARCHAR" + tC + "node2 VARCHAR" + tC + "label VARCHAR" + _
        tC + "edge_desc VARCHAR(50)" + tC + "edgetype VARCHAR(1) "
End If
'MsgBox "Writing " + tStr
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)

With tRstEdge
.MoveNext
Do While Not .EOF
    '    node1 = str(c_person_id) for node1
    tStr = Trim(Str(!c_person_id)) + tC

    '    node2 = str(c_node_id) for node2
    tStr = tStr + Trim(Str(!c_node_id)) + tC

    '    label
    If tCodeStr = "ASCII" Then
        If IsNull(!c_link_desc) Then
            tStr = tStr + tC
        Else
            tStr = tStr + tQuote + Trim(Left(!c_link_desc + Space(50), 50)) + tQuote + tC
        End If
    Else
        If IsNull(!c_link_chn) Then
            tStr = tStr + tC
        Else
            tStr = tStr + tQuote + !c_link_chn + tQuote + tC
        End If
    End If

    '    edge_desc = c_link_desc, if used
    If Not (tCodeStr = "ASCII") Then
        If IsNull(!c_link_desc) Then
            tStr = tStr + tC
        Else

```

```

        tStr = tStr + tQuote + Trim(Left(!c_link_desc + Space(50), 50)) + tQuote + tC
    End If
End If

'    edgetype= c_link_type (K,N)
tStr = tStr + !c_link_type

'MsgBox "Writing " + tStr
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
.MoveNext

Loop
End With
'
' now make sure all the data is copied to tStream
'MsgBox "Flushing..."
tStream.Flush
' and write the stream to the file
'MsgBox "Writing..."
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
'MsgBox "Closing..."
tStream.Close
Set tStream = Nothing
'tGDF.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
'Set tGDF = Nothing
'Set tFileSystem = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGUESS_Click:
Exit Sub

Err_CmdGUESS_Click:
' finish writing and close the file
MsgBox Err.Description
'
If Not IsNull(tStr) Then
    tStream.WriteText tStr, adWriteLine
End If
tStream.Flush
' and write the stream to the file
'MsgBox "Writing..."
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
'MsgBox "Closing..."
tStream.Close
Set tStream = Nothing

Resume Exit_CmdGUESS_Click

End Sub

Private Sub CmdImportPeople_Click()
On Error GoTo Err_CmdImportPeople_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String, tLen As Integer, tQuit As Boolean

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tFileSystem, tList

'
tQuit = False
If Not tQuit Then
    ' open the list

    Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

```

```

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = ""
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdImportPeople_Click
        End If
    End If
End With
'
' Clear the various tables now that we are ready to go
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "ImportPeopleList_Space", "TempImportList", tFileName, 0
'    TransferType=acImportDelim
'    SpecificationName = "TempImportList" (apparently it is saved in the database itself)
'    TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
'    HasFieldNames = False (0)
'
' copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
    "FROM BIOG_MAIN RIGHT JOIN TempImportList ON BIOG_MAIN.c_personid = TempImportList.ImportID " + _
    "WHERE ((BIOG_MAIN.c_personid) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If
'
' copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id ) SELECT DISTINCT TempImportList.ImportID " + _
    "FROM BIOG_MAIN INNER JOIN TempImportList ON BIOG_MAIN.c_personid = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    TxtName.Value = "[Imported List]"
    TxtNameChn.Value = "[" + ChrW(&H8F38) + ChrW(&H5165) + ChrW(&H7684) + ChrW(&H4EBA) + ChrW(&H540D) + "
] "
    ' shu = 8F38, ru = 56DE, de = 7684, ren = 4EBA, ming = 540D
    gUsePersonID = True
    Me.CmdAllPeople.Enabled = True
    CmdRun.Enabled = True
    'CmdRerun.Enabled = False
End If

Set cmdSQL = Nothing
Set tFileSystem = Nothing
End If
Call CheckRunCriteria

```

```

Exit_CmdImportPeople_Click:
    Exit Sub

Err_CmdImportPeople_Click:
    MsgBox Err.Description
    Resume Exit_CmdImportPeople_Click

End Sub

Private Sub CmdImportPlaces_Click()
    On Error GoTo Err_CmdImportPlaces_Click

    Dim stDocName As String, tRstAddresses As DAO.Recordset
    Dim stLinkCriteria As String
    Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String, tLen As Integer, tQuit As Boolean

    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tFileSystem, tList

    '
    tQuit = False
    If Not tQuit Then
        ' open the list

        Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

        'Use a With...End With block to reference the FileDialog object.
        With dlgSaveAs
            .InitialFileName = ""
            If .Show = -1 Then
                '
                tFileName = ""
                For Each tFN In .SelectedItems
                    tFileName = tFN
                    If Not tFileName = "" Then
                        Exit For
                    End If
                Next
                If tFileName = "" Then
                    MsgBox "Bad file Name."
                    GoTo Exit_CmdImportPlaces_Click
                End If
            End If
        End With

        ' Clear the address table now that we are ready to go
        '

        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText

        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_LIST"
        cmdSQL.Execute tRecDeleted

        cmdSQL.CommandText = "Delete * from InputErrorList"
        cmdSQL.Execute tRecDeleted

        cmdSQL.CommandText = "Delete * from TempImportList"
        cmdSQL.Execute tRecDeleted

        DoCmd.TransferText acImportDelim, "ImportPlaceList_Space", "TempImportList", tFileName, 0
        ' TransferType=acImportDelim
        ' SpecificationName = "TempImportList" (apparently it is saved in the database itself)
        ' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
        ' HasFieldNames = False (0)
        '
        ' copy the bad IDs
        '
        tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
            "FROM ADDR_CODES RIGHT JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID " + _
            "WHERE ((ADDR_CODES.c_addr_id) Is Null))"

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted

        If tRecDeleted > 0 Then

```

```

24
    MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If
'
' copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT TempImportList.ImportID " + _
    "FROM ADDR_CODES INNER JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    TxtPlace.Value = "[Imported List]"
    Me.TxtPlaceChn.Value = "[Imported List]"
    TxtPlaceChn.Value = "[" + ChrW(&H8F38) + ChrW(&H5165) + ChrW(&H7684) + ChrW(&H5730) + ChrW(&H540D) + _
"]"

    ' shu = 8F38, ru = 56DE, de = 7684, di = 5730, ming = 540D
    gUseADDRID = True
    ChkXYRef.Enabled = True
    ChkSubUnits.Enabled = True
    ChkPlaceLimit.Enabled = True
    CmdRun.Enabled = True
End If

Set cmdSQL = Nothing
Set tFileSystem = Nothing
End If

Call CheckRunCriteria

Exit_CmdImportPlaces_Click:
Exit Sub

Err_CmdImportPlaces_Click:
MsgBox Err.Description
Resume Exit_CmdImportPlaces_Click

End Sub

Private Sub CmdJiantiDisplay_Click()
On Error GoTo Err_CmdJiantiDisplay_Click
If gDisplayLanguage = "S" Then
    gDisplayLanguage = "E"
Else
    gDisplayLanguage = "S"
End If

Call changeDisplayLanguage

Exit_CmdJiantiDisplay_Click:
Exit Sub

Err_CmdJiantiDisplay_Click:
MsgBox Err.Description
Resume Exit_CmdJiantiDisplay_Click

End Sub

Private Sub CmdPajek_Old_Click()
On Error GoTo Err_CmdPajek_Click
'
' This program will dump the results of the search to a .net file
'
' for the moment I'll just describe the format of the .gdf file
'
' *Vertices NUM
' ID label "box" ic [color] bc [color]
' ID = str(c_person_id)
' label = c_name_chn
' color = red (1), orange (2), yellow (3), green (4), blue (5)
'
' *Edges
' node1 node2 1 1 "label"
' node1 = str(c_person_id) for node1
' node2 = str(c_node_id) for node2
' color = red (1), orange (2), yellow (3), green (4), blue (5)
' label = c_link_desc
'
'
'

```



```

' first see if there are any records to process
'
If ZZ_SOCIAL_NETWORK.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
'
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstNodeList As DAO.Recordset
Dim tRstEdge As DAO.Recordset, tRstAssocType As DAO.Recordset
Dim tRstAssocCodeType As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tQuote As String, tFindStr As String
Dim tColor(20) As String
Dim tFileSystem

Dim tStream As ADODB.Stream

Set tStream = New ADODB.Stream
tStream.Charset = "ascii"
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network.net"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdPajek_Click
        Else
            ' make sure the file name has a net extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".net"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".net") Then
                tFileName = tFileName + ".net"
            End If
        End If
        '
        ' zap and open the scratch file
        '
        Dim cmdSQL As ADODB.Command
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
        cmdSQL.Execute tRecDeleted
        '
        ' fill the node list
        '
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id, ZZ_SCRATCH_PEOPLE.c_name, " + _
            "ZZ_SCRATCH_PEOPLE.c_node_dist, val(c_person_id) AS c_v_num FROM ZZ_SCRATCH_PEOPLE"

        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted

        Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
    End With

```

```

tRstNodeList.Index = "c_ID"
'
'   there probably is an SQL way to do this, but...
'
ti = 1
With tRstNodeList
    .MoveFirst
    Do While Not .EOF
        .Edit
        !c_v_num = Trim(Str(ti))
        .Update
        ti = ti + 1
        .MoveNext
    Loop
End With
tRstNodeList.Close
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK_EDGE"
cmdSQL.Execute tRecDeleted
'
'   fill the edge list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK_EDGE ( c_node_1, c_node_2, c_edge_count, c_edge_dist, c_edg
e_desc )" + _
"SELECT Val([ZZ_SCRATCH_PAJEK].[c_v_num]) AS c_node_1, Val([ZZ_SCRATCH_PAJEK_1].[c_v_num]) " + _
"AS c_node_2, ZZ_SOCIAL_NETWORK_AGGREGATE.c_link_count, ZZ_SOCIAL_NETWORK_AGGREGATE.c_edge_dist,
" + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_link_desc " + _
"FROM ZZ_SCRATCH_PAJEK INNER JOIN (ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 INNER JOIN " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE ON ZZ_SCRATCH_PAJEK_1.c_ID = ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_id)
" + _
"ON ZZ_SCRATCH_PAJEK.c_ID = ZZ_SOCIAL_NETWORK_AGGREGATE.c_person_id"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'
'   set the Quote delimiter
'
tQuote = Chr(34)
'
'   define the colors for the nodes
'
tColor(1) = "White"
tColor(2) = "Blue"
tColor(3) = "Green"
tColor(4) = "Yellow"
tColor(5) = "Orange"
For ti = 6 To 20
    tColor(ti) = "Red"
Next
'
'   process the two tables
'
Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenDynaset)
Set tRstEdgeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK_EDGE", dbOpenDynaset)
tC = Chr(44) ' the comma
'
'   first the nodes:  define the record structure
tRstNodeList.MoveLast
tStr = "*Vertices " + Trim(Str(tRstNodeList.RecordCount))
tStream.WriteText tStr, adWriteLine
'
ti = 1
With tRstNodeList
    .MoveFirst
    Do While Not .EOF
        tStream.WriteText !c_v_num + " "
        '
        If IsNull(!c_lbl) Then
            tStream.WriteText Chr(34)
            tStream.WriteText "Error-" + Trim(Str(!c_ID))
            tStream.WriteText Chr(34)
            tStream.WriteText " box "
        Else
            If !c_lbl = "" Then
                tStream.WriteText Chr(34)
                tStream.WriteText "Error-" + Trim(Str(!c_ID))
                tStream.WriteText Chr(34)
                tStream.WriteText " box "
            End If
        End If
    Loop
End With

```

```

Else
    tStream.WriteText Chr(34)
    tStream.WriteText !c_lbl
    If ChkIncludeID.Value Then
        tStream.WriteText ":" + Trim(Str(!c_ID))
    End If
    tStream.WriteText Chr(34)
    tStream.WriteText " box "
End If
End If
' label
tStr = " ic " + tColor(!c_distance + 1)
tStr = tStr + " bc " + tColor(!c_distance + 1)
' color = white (1), blue (2), green (3), yellow (4), orange (5)
tStream.WriteText tStr, adWriteLine
'
.MoveNext
Loop
End With

'
' now the edges: define the record structure
'
tStream.WriteText "*Edges", adWriteLine

If tRstEdgeList.RecordCount > 0 Then
    With tRstEdgeList
        .MoveFirst
        Do While Not .EOF
            tStr = Trim(Str(!c_node_1)) + " " + Trim(Str(!c_node_2))
            '
            ' now get the weight
            '
            If !c_edge_count < 6 Then
                tStr = tStr + " " + Trim(Str(!c_edge_count)) + " "
            Else
                tStr = tStr + " 5 "
            End If
            '
            ' now get the label
            '
            tStr = tStr + "l " + tQuote
            If !c_edge_count = 1 Then
                tStr = tStr + !c_edge_desc + tQuote + " "
            Else
                tStr = tStr + Trim(Str(!c_edge_count)) + " links" + tQuote + " "
            End If

            tStr = tStr + "c " + tColor(!c_edge_dist + 1)
            ' color = white (1), blue (2), green (3), yellow (4), orange (5)
            '
            tStream.WriteText tStr, adWriteLine
            '
            .MoveNext
        Loop
    End With
End If
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tRstNodeList.Close

tStream.Close
'
Set tStream = Nothing
Set tFileSystem = Nothing
Set tRstNodeList = Nothing
Set tRstEdgeList = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdPajek_Click:

```

```

28
Exit Sub

Err_CmdPajek_Click:
MsgBox Err.Description
Resume Exit_CmdPajek_Click

End Sub

Private Sub CmdRerun_Click()
On Error GoTo Err_CmdRerun_Click

Dim tQueryStr As String, tRecCount As Long
Dim cmdSQL As ADODB.Command

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted

tQueryStr = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id ) " +
"SELECT ZZ_SCRATCH_PEOPLE.c_person_id FROM ZZ_SCRATCH_PEOPLE"

MsgBox "Populating people list table: " + tQueryStr

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount

If tRecCount > 0 Then
gUsePersonID = True
Call CmdRun_Click
Else
MsgBox "There were no results to reuse."
End If

Exit_CmdRerun_Click:
Exit Sub

Err_CmdRerun_Click:
MsgBox Err.Description
Resume Exit_CmdRerun_Click

End Sub

Private Sub CmdNeo4j_Click()
On Error GoTo Err_CmdNeo4j_Click

' This routine will be close to that for LookAtAssociations and, if used, that for LookAtKinship
' The additional wrinkle is that, while the first step is to split associatin from kinship relations, we sti
ll need to gather all
' the people from both.
'
' first see if there are any records to process
'
If Me.ZZ_SOCIAL_NETWORK.Form.Recordset.RecordCount = 0 Then
MsgBox "There are no records to save."
GoTo Exit_CmdNeo4j_Click
End If

' allocate the file variables
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant

' next get the People file
'
Dim tRstPeople As DAO.Recordset, tRstAssoc As DAO.Recordset, tRstPlace As DAO.Recordset, tRstPeoplePlace As DAO.Recordset
Dim tStr As String, tC As String, tQueryStr As String, tRstAssocCode As DAO.Recordset, tRstKin As DAO.Recordset

Dim gStream As ADODB.Stream, tCodeStr As String, tTempLong As Long

' set up the stream to write to

Set gStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
gStream.Charset = "utf-8"
tCodeStr = "UTF8"

```

```

ElseIf CodeFrame.Value = 2 Then
    gStream.Charset = "big5"
    tCodeStr = "BIG5"
ElseIf CodeFrame.Value = 3 Then
    gStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    gStream.Charset = "ascii"
    tCodeStr = "ascii"
End If

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
    'Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)
    '
    ' we have a file name: now open the stream for writing

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    '
    ' prepare the temp tables for the people, place, peoplePlace and assoc codes

    Dim cmdSQL As ADODB.Command
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' Get the people from 5 sources: c_person_id, c_node_id, c_kin_id, c_assoc_kin_id, and c_assoc_claim
    '

    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_TEXT"
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (1)
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id " + _
        "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_person_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (2)
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_node_id " + _
        "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_node_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (3)
    '

```

```

tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_kin_id
" + _
        "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_kin_id)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' copy the data for people (4)
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_assoc_
kin_id " + _
        "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_assoc_kin_id)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' copy the data for people (5)
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_assoc_
claimer_id " + _
        "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_assoc_claimer_id)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_nam
e_chn, ZZZ_BIOG_MAIN.c_index_year, " + _
        "ZZZ_BIOG_MAIN.c_dynasty, ZZZ_BIOG_MAIN.c_dynasty_chn, ZZZ_BIOG_MAIN.c_index_addr_id,
ZZZ_BIOG_MAIN.c_index_addr_name, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_chn, ZZZ_BIOG_MAIN.c_index_addr_type_code, ZZZ_BIOG_MAIN.
c_index_addr_type_desc, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_chn, ZZZ_BIOG_MAIN.c_female, ZZZ_BIOG_MAIN.x_coord,
ZZZ_BIOG_MAIN.y_coord " + _
        "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_B
IOG_MAIN.c_personid"
,
Set tRstPeople = CurrentDb.OpenRecordset(tQueryStr, dbOpenDynaset)
,
' process the four tables
,
tC = Chr(44) ' the comma
,
' first the nodes: define the record structure
,
' if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a
pinyin field
If tCodeStr = "ascii" Then
    tStr = "nameID" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
Else
    tStr = "nameID" + tC + "nameHZ" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
End If
gStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
,
With tRstPeople
    .MoveFirst
    Do While Not .EOF
        ' the ID of the person
        tStr = Trim(Str(!c_person_id)) + tC
        ,
        ' name
        ,
        If tCodeStr = "ascii" Then
            If IsNull(!c_name) Then
                tStr = tStr + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        Else
            If IsNull(!c_name_chn) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If

            If IsNull(!c_name) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        End If
    End While
End With

```

```

End If
'
'   indexyear = c_index_year INT
'
If IsNull(!c_index_year) Then
    tStr = tStr + "-2000" + tC
Else
    tStr = tStr + Trim(Str(!c_index_year)) + tC
End If

'   dynasty information
'
If IsNull(!c_dynasty) Then
    tStr = tStr + "unknown" + tC
Else
    If tCodeStr = "ascii" Then
        tStr = tStr + !c_dynasty + tC
    Else
        tStr = tStr + !c_dynasty_chn + tC
    End If
End If

'
'   sex = c_female > (F,M)
tStr = tStr + IIf(!c_female, "F", "M")
'
gStream.WriteText tStr, adWriteLine
'
.MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
'   now places:  since the association "event" is not linked to a place, the only addresses are the index
addresses
'
'               of the people involved, recorded in ZZ_SCRATCH_PEOPLE
'
'   get a file name
'
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    '   now process the file
    '
    tQueryStr = "SELECT DISTINCT ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_name, " + _
                "ZZZ_BIOG_MAIN.c_index_addr_chn, ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord " + _
                "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_B
IOG_MAIN.c_personid " + _
                "WHERE (ZZZ_BIOG_MAIN.c_index_addr_id > 0)"

    Set tRstPlace = CurrentDb.OpenRecordset(tQueryStr)

```

```

'
If tCodeStr = "ascii" Then
    tStr = "placeID" + tC + "placePY" + tC + "placeX" + tC + "placeY"
Else
    tStr = "placeID" + tC + "placePY" + tC + "placeHZ" + tC + "placeX" + tC + "placeY"
End If
gStream.WriteText tStr, adWriteLine
With tRstPlace
    .MoveFirst
    Do While Not .EOF
        ' the ID of the place
        If Not IsNull(!c_index_addr_id) Then
            tStr = Trim(Str(!c_index_addr_id)) + tC
            '
            ' address name

            If IsNull(!c_index_addr_name) Then
                tStr = tStr + "unknown" + tC
            Else
                tStr = tStr + !c_index_addr_name + tC
            End If
            '
            If Not (tCodeStr = "ascii") Then
                If IsNull(!c_index_addr_chn) Then
                    tStr = tStr + "unknown" + tC
                Else
                    tStr = tStr + !c_index_addr_chn + tC
                End If
            End If

            ' longitude = !x_coord
            If IsNull(!x_coord) Then
                tStr = tStr + "0.0" + tC
            Else
                tStr = tStr + Str(!x_coord) + tC
            End If

            ' latitude = !y_coord
            If IsNull(!y_coord) Then
                tStr = tStr + "0.0"
            Else
                tStr = tStr + Str(!y_coord)
            End If
            '
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces
'
dlgSaveAs.InitialFileName = "PeoplePlaces_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        End If
    End If
End If

```



```

        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_M
AIN.c_index_addr_type_code, " +
                "ZZZ_BIOG_MAIN.c_index_addr_type_desc, ZZZ_BIOG_MAIN.c_index_addr_type_chn " +
                "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_B
IOG_MAIN.c_personid " +
                "WHERE (ZZZ_BIOG_MAIN.c_index_addr_id > 0)"

    Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

    If (tCodeStr = "ascii") Then
        tStr = "nameID" + tC + "placeID" + tC + "personPlaceCode" + tC + "personPlaceTrans"
    Else
        tStr = "nameID" + tC + "placeID" + tC + "personPlaceCode" + tC + "personPlaceTrans" + tC + "perso
nPlaceHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstPeoplePlace
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_index_addr_id) Then
                ,
                tStr = Trim(Str(!c_person_id)) + tC
                ,
                tStr = tStr + Trim(Str(!c_index_addr_id)) + tC
                ,
                tStr = tStr + Trim(Str(!c_index_addr_type_code)) + tC
                ,
                tStr = tStr + Trim(Str(!c_index_addr_type_desc))
                ,
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(Str(!c_index_addr_type_chn))
                End If
                ,
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    ,
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
,
' now the association records
,
dlgSaveAs.InitialFileName = "PeopleAssociations_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
End If

```

```

        End If
    End If
    '
    gStream.Open
    '
    ' now the associations:  define the record structure
    ' Because of the complexity of the primary key, this gets a bit complicated
    '
    Set tRstAssoc = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
    '
    tStr = "Person1_ID" + tC + "Person2_ID" + tC + "Association_Code" + tC + "Association_FirstYear" + tC
+ "Kin_ID" + tC + "Kin_Code" + tC + _
        "AssocKin_ID" + tC + "AssocKin_Code" + tC + "LiteraryGenreCode" + tC + "OccasionCode" + tC +
        "TopicCode" + tC + "InstitutionCode" + tC + "TextTitle" + tC + "AssociationClaimer_ID"
    gStream.WriteText tStr, adWriteLine
    'tGDF.WriteLine (tStr)

With tRstAssoc
    .MoveFirst
    Do While Not .EOF
        If !c_link_type = "N" And (Not IsNull(!c_link_code)) Then
            tStr = Trim(Str(!c_person_id)) + tC
            '    node1 = str(c_person_id) for node1
            tStr = tStr + Trim(Str(!c_node_id)) + tC
            '    node2 = str(c_node_id) for node2
            tStr = tStr + Trim(Str(!c_link_code)) + tC
            '
            '    FirstYear (cannot be NULL)
            '
            tStr = tStr + Str(!c_link_first_year) + tC

            '    kin ID
            If IsNull(!c_kin_id) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Str(!c_kin_id) + tC
            End If

            '    kin code
            If IsNull(!c_kin_code) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Str(!c_kin_code) + tC
            End If

            '    assoc kin ID
            If IsNull(!c_assoc_kin_id) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Str(!c_assoc_kin_id) + tC
            End If

            '    assoc kin code
            If IsNull(!c_assoc_kin_code) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Str(!c_assoc_kin_code) + tC
            End If

            '    literary genre code
            If IsNull(!c_litgenre_code) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Str(!c_litgenre_code) + tC
            End If

            '    occasion code
            If IsNull(!c_occasion_code) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Str(!c_occasion_code) + tC
            End If

            '    topic code
            If IsNull(!c_topic_code) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Str(!c_topic_code) + tC
            End If
        End If
    End While
End With

```

```

'    institution code
If IsNull(!c_inst_code) Then
    tStr = tStr + "0" + tC
Else
    tStr = tStr + Str(!c_inst_code) + tC
End If

'    text title
If IsNull(!c_text_title) Then
    tStr = tStr + "N/A" + tC
Else
    tStr = tStr + !c_text_title + tC
End If

'    association claimer ID
If IsNull(!c_assoc_claimer_id) Then
    tStr = tStr + "0" + tC
Else
    tStr = tStr + Str(!c_assoc_claimer_id)
End If

gStream.WriteText tStr, adWriteLine
End If
.MoveNext
Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
End If
'
' now the kinship relations: first, will there be any? -1 = "True"
'
If ChkKin.Value = -1 Then
    cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP"
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_kin_code, c_kinrel, c_kinrel_total ) " +
        "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_link_code, ZZ_SOCIAL_NETWORK.c_link_desc, ZZ_SOCIAL_
NETWORK.c_link_chn " +
        "FROM ZZ_SOCIAL_NETWORK " +
        "WHERE ((ZZ_SOCIAL_NETWORK.c_link_type)='K') AND ((ZZ_SOCIAL_NETWORK.c_link_code)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted

    If tRecDeleted > 0 Then
        dlgSaveAs.InitialFileName = "KinshipRelations_" + tCodeStr + ".csv"
        If dlgSaveAs.Show = -1 Then
            '
            tFileName = ""
            For Each tFN In dlgSaveAs.SelectedItems
                tFileName = tFN
                If Not tFileName = "" Then
                    Exit For
                End If
            Next
            If tFileName = "" Then
                MsgBox "Bad file Name."
                GoTo Exit_CmdNeo4j_Click
            Else
                ' make sure the file name has a txt extension
                If Len(tFileName) < 5 Then
                    tFileName = tFileName + ".csv"
                ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                    tFileName = tFileName + ".csv"
                End If
            End If
        End If
        '
        gStream.Open
        '
        tStr = "PersonID" + tC + "KinID" + tC + "KinCode"

```

```

gStream.WriteText tStr, adWriteLine
'
' still using ZZ_SOCIAL_NETWORK
'
With tRstAssoc
.MoveFirst
Do While Not .EOF
    If !c_link_type = "K" And (Not IsNull(!c_link_code)) Then
        tStr = Trim(Str(!c_person_id)) + tC
        tStr = Trim(Str(!c_node_id)) + tC
        tStr = Trim(Str(!c_link_code))
        gStream.WriteText tStr, adWriteLine
    End If
    .MoveNext
Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
End If
'
' now the association codes
'
dlgSaveAs.InitialFileName = "AssociationCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    gStream.Open
    tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_link_code, ZZ_SOCIAL_NETWORK.c_link_desc, ZZ_SOCIAL_NETWORK.c_link_chn " + _
        "FROM ZZ_SOCIAL_NETWORK " + _
        "WHERE ((ZZ_SOCIAL_NETWORK.c_link_type = 'N') and (ZZ_SOCIAL_NETWORK.c_link_code > 0))"
    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr, dbOpenDynaset)
    If tCodeStr = "ascii" Then
        tStr = "AssociationCode" + tC + "AssociationTrans"
    Else
        tStr = "AssociationCode" + tC + "AssociationTrans" + tC + "AssociationHZ"
    End If
    gStream.WriteText tStr, adWriteLine
    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_link_code) Then

```

```

        tStr = Trim(Str(!c_link_code)) + tC
        '
        tStr = tStr + Trim(!c_link_desc)
        '
        If Not (tCodeStr = "ascii") Then
            tStr = tStr + tC + Trim(!c_link_chn)
        End If
        gStream.WriteText tStr, adWriteLine
    End If
    .MoveNext
Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' there are codes that MAY require additional tables: c_kin_code, c_litgenrte_code, c_occasion_code, c_t
opic_code, c_inst_code
'
' test for kin codes
'
' tRecDeleted is the number of kinship codes inserted into ZZ_KIN_LIST_TEMP for the earlier test
'
tTempLong = tRecDeleted
tQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_kin_code, c_kinrel, c_kinrel_total ) " + _
"SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_kin_code, ZZ_SOCIAL_NETWORK.c_kin_desc, ZZ_SOCIAL_NETWORK
K.c_kin_desc_chn " + _
"FROM ZZ_SOCIAL_NETWORK " + _
"WHERE (((ZZ_SOCIAL_NETWORK.c_kin_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

tTempLong = tTempLong + tRecDeleted
tQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_kin_code, c_kinrel, c_kinrel_total ) " + _
"SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_assoc_kin_code, ZZ_SOCIAL_NETWORK.c_assoc_kin_desc, ZZ_S
OCIAL_NETWORK.c_assoc_kin_desc_chn " + _
"FROM ZZ_SOCIAL_NETWORK " + _
"WHERE (((ZZ_SOCIAL_NETWORK.c_assoc_kin_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

tTempLong = tTempLong + tRecDeleted
'
' debug
'
MsgBox "Kinship code records = " + Trim(Str(tTempLong))
'
If tTempLong > 0 Then
    dlgSaveAs.InitialFileName = "KinshipCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
    End If
End If
'

```

```

gStream.Open
,
tQueryStr = "SELECT DISTINCT ZZ_KIN_LIST_TMP.c_kin_code, ZZ_KIN_LIST_TMP.c_kinrel, ZZ_KIN_LIST_TM
P.c_kinrel_total " + _
                "FROM ZZ_KIN_LIST_TMP"

Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "KinshipCode" + tC + "KinshipTrans"
Else
    tStr = "KinshipCode" + tC + "KinshipTrans" + tC + "KinshipHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstAssocCode
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_kin_code) Then
            ,
            tStr = Trim(Str(!c_kin_code)) + tC
            ,
            tStr = tStr + Trim(!c_kinrel)
            ,
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + Trim(!c_kinrel_total)
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
,
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
,
' test for literary genre codes
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
            "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id " + _
            "FROM ZZ_SOCIAL_NETWORK " + _
            "WHERE (((ZZ_SOCIAL_NETWORK.c_litgenre_code)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' debug
,
MsgBox "Literary genre code records = " + Trim(Str(tRecDeleted))
,
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "LiteraryGenreCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
    End If
End If

```

```

        End If
    End If
    gStream.Open
    tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_litgenre_code, ZZ_SOCIAL_NETWORK.c_litgenre_desc
, ZZ_SOCIAL_NETWORK.c_litgenre_desc_chn " + _
        "FROM ZZ_SOCIAL_NETWORK " + _
        "WHERE (((ZZ_SOCIAL_NETWORK.c_litgenre_code)>0))"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "LitGenreCode" + tC + "LitGenreTrans"
    Else
        tStr = "LitGenreCode" + tC + "LitGenreTrans" + tC + "LitGenreHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_litgenre_code) Then
                tStr = Trim(Str(!c_litgenre_code)) + tC
                tStr = tStr + Trim(!c_litgenre_desc)
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_litgenre_desc_chn)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With

    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
' test for institution codes
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
    "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id " + _
    "FROM ZZ_SOCIAL_NETWORK " + _
    "WHERE (((ZZ_SOCIAL_NETWORK.c_inst_code)>0))"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
' debug
MsgBox "Institution code records = " + Trim(Str(tRecDeleted))
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "InstitutionCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension

```

```

        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_inst_code, ZZ_SOCIAL_NETWORK.c_inst_name_py, ZZ_
SOCIAL_NETWORK.c_inst_name_hz " + _
                "FROM ZZ_SOCIAL_NETWORK " + _
                "WHERE ((ZZ_SOCIAL_NETWORK.c_inst_code)>0))"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "InstitutionCode" + tC + "InstitutionNamePY"
    Else
        tStr = "InstitutionCode" + tC + "InstitutionNamePY" + tC + "InstitutionNameHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_inst_code) Then
                ,
                tStr = Trim(Str(!c_inst_code)) + tC
                ,
                tStr = tStr + Trim(!c_inst_name_py)
                ,
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_inst_name_hz)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    ,
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
,
' test for occasion codes
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
            "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id " + _
            "FROM ZZ_SOCIAL_NETWORK " + _
            "WHERE (((ZZ_SOCIAL_NETWORK.c_occasion_code)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' debug
,
MsgBox "Occasion code records = " + Trim(Str(tRecDeleted))
,
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "OccasionCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then

```



```

        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_occasion_code, ZZ_SOCIAL_NETWORK.c_occasion_desc
, ZZ_SOCIAL_NETWORK.c_occasion_desc_chn " + _
        "FROM ZZ_SOCIAL_NETWORK " + _
        "WHERE ((ZZ_SOCIAL_NETWORK.c_occasion_code)>0))"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "OccasionCode" + tC + "OccasionTrans"
    Else
        tStr = "OccasionCode" + tC + "OccasionTrans" + tC + "OccasionHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_occasion_code) Then
                '
                tStr = Trim(Str(!c_occasion_code)) + tC
                '
                tStr = tStr + Trim(!c_occasion_desc)
                '
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_occasion_desc_chn)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    '
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
'
' test for topic codes
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
    "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id " + _
    "FROM ZZ_SOCIAL_NETWORK " + _
    "WHERE (((ZZ_SOCIAL_NETWORK.c_topic_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' debug
'
MsgBox "Topic code records = " + Trim(Str(tRecDeleted))
'
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "TopicCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then

```

```

        Exit For
    End If
Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdNeo4j_Click
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".csv"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
        tFileName = tFileName + ".csv"
    End If
End If
'
gStream.Open
' because ZZZ_NONKIN_BIOG_ADDR does not have the topic descriptions (must fix), I need to use a join to TOPIC_CODES
'
tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_topic_code, SCHOLARLYTOPIC_CODES.c_topic_desc, SCHOLARLYTOPIC_CODES.c_topic_desc_chn " +
            "FROM ZZ_SOCIAL_NETWORK INNER JOIN SCHOLARLYTOPIC_CODES ON ZZ_SOCIAL_NETWORK.c_topic_code = SCHOLARLYTOPIC_CODES.c_topic_code " +
            "WHERE (((ZZ_SOCIAL_NETWORK.c_topic_code)>0))"

Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "TopicCode" + tC + "TopicTrans"
Else
    tStr = "TopicCode" + tC + "TopicTrans" + tC + "TopicHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstAssocCode
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_topic_code) Then
            '
            tStr = Trim(Str(!c_topic_code)) + tC
            '
            tStr = tStr + Trim(!c_topic_desc)
            '
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + Trim(!c_topic_desc_chn)
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
End If
MsgBox "Finished saving to Neo4j"
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click

End Sub

Private Sub CmdRun_Click()
On Error GoTo Err_CmdRun_Click

```

```

Dim tTrue As Integer, tFalse As Integer, tLoop As Integer
Dim tMale As Integer, tFemale As Integer, tBothSexes As Integer, tContinue As Integer
Dim tRstDummy As DAO.Recordset
Dim tLoopInfoStr As String, tNonkinQueryStr As String, tKinQueryStr As String
Dim tRstAddrList As DAO.Recordset

Dim tQueryAssocAddrAssocFromStr As String, tQueryAssocAddrFromStr As String, tQueryAssocAssocFromStr As String
Dim tQueryAssocFromStr As String, tQueryKinAddrFromStr As String, tQueryKinFromStr As String
Dim tQueryAssocLastAddrAssocFromStr As String, tQueryAssocLastAddrFromStr As String, tQueryAssocLastAssocFromStr As String
Dim tQueryAssocLastFromStr As String, tQueryKinLastAddrFromStr As String, tQueryKinLastFromStr As String

Dim tNonkinWhereQueryStr As String, tKinWhereQueryStr As String, tKinWhereFirstQueryStr As String
Dim tQuerySelectFirstNonkin As String, tQuerySelectFirstKin As String, tNonkinWhereFirstQueryStr As String
Dim tQueryAssocFirstAddrAssocFromStr As String, tQueryAssocFirstAddrFromStr As String
Dim tQueryAssocFirstAssocFromStr As String, tQueryKinFirstAddrFromStr As String, tQueryAppendStr As String
Dim tQueryAssocFirstFromStr As String, tPruneTmpQueryDuplesStr As String

Dim tQueryKinStr As String, tQueryKinFirstStr As String, tNodeDistQueryStr As String
Dim tQueryNonkinStr As String, tQueryNonkinFirstStr As String, tPruneTmpQuery As String
Dim tQueryNonkinLastStr As String, tQueryKinLastStr As String, tQueryCopyNonkinStr As String, tQueryCopyKinStr As String
Dim tQueryPruneTmpAssocInverse1Str As String, tQueryPruneTmpAssocInverse2Str As String
Dim tQueryPruneTmpKinInverse1Str As String, tQueryPruneTmpKinInverse2Str As String
Dim tQueryPruneTmpKinInverse3Str As String, tQueryPruneTmpKinInverse4Str As String
Dim tRecCountKin As Long, tRecCountNonkin As Long, tStrQuerySet As String

tTrue = -1
tFalse = 0
tFemale = -1
tMale = 0
tBothSexes = 1
gLoopMax = TxtMaxLoop.Value
gMaxNodeDist = TxtNodeDist.Value

'
' The design of this routine is to build the SQL queries that will sweep together the needed people
' The logic has three different query types
' 1. Using a name or list of names to generate the first set of associations
' 2. Using those results to look for the next cycles of associations
' 3. When the final node distance has been reached, look among the new people for relations with one another
'
' We begin with error checking for the interface:
'
' First thing is to see if the person has specified "everyone"
'
' Do we need the association filter? We need to do this first because of the condition that follows
'
gUseFilter = tFalse
If ChkNonKin.Value = tTrue Then
    If gFilterTotalCount < gMaxFilterTotal Then
        If gFilterTotalCount = 0 Then
            ChkNonKin.Value = tFalse
        Else
            gUseFilter = tTrue
        End If
    End If
End If

If Not gUsePersonID And Not gUseADDRID And gUseFilter = tFalse Then
    MsgBox "Please select a person and/or a place."
    GoTo Exit_CmdRun_Click
End If

' Next see if at least one network checkbox has been clicked
'
If ChkKin.Value = tFalse And ChkNonKin.Value = tFalse Then
    MsgBox "Please select the kinship and/or non-kinship option."
    GoTo Exit_CmdRun_Click
Else
    If ChkKin.Value = tFalse And ChkNonKin.Value = tTrue Then
        If gFilterTotalCount = 0 Then
            MsgBox "Please select at least one non-kinship relation."
            GoTo Exit_CmdRun_Click
        End If
    End If
End If

```

```

' Next see if at least one sex checkbox has been clicked

If ChkMale.Value = tFalse And ChkFemale.Value = tFalse Then
    MsgBox "Please select the male and/or female option."
    GoTo Exit_CmdRun_Click
End If
'
' We've survived and now need to do the actual processing
'
Dim tQuery As DAO.QueryDef
'
Dim cmdSQL As ADODB.Command, tRecDeleted As Long, strSQL As String
'
' Next we need to do the preliminaries of preparing scratch tables
'
' Clear the output table and the scratch personID table, if needed
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
'   clear the two scratch files
'
MsgBox "Clearing tables..."
cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST_TMP"
cmdSQL.Execute tRecDeleted

Set gRstEdge = ZZ_SOCIAL_NETWORK.Form.Recordset
Set gRstPersonID = ZZ_SCRATCH_PEOPLE.Form.Recordset
'
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SN", dbOpenDynaset)
Set ZZ_SOCIAL_NETWORK.Form.Recordset = tRstDummy
Set ZZ_SOCIAL_NETWORK_AGGREGATED.Form.Recordset = tRstDummy
'
gRstEdge.Close
cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
' now zap the scratch person file
'
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SP", dbOpenDynaset)
Set ZZ_SCRATCH_PEOPLE.Form.Recordset = tRstDummy
'
'   to get rid of superfluous deleted records, briefly close ZZ_SCRATCH_PEOPLE
'
gRstPersonID.Close
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted
'
' now zap the association filter table and refill, if needed
'
If gUseFilter = tTrue Then
    MsgBox "Building ASSOC filter table..."

    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ASSOC_FILTER"
    cmdSQL.Execute tRecDeleted
    '
    Call makeAssocFilter
End If
'
' now see if address IDs will be used.  If so, zap the scratch file and repopulate
'
If gUseADDRID Then
    MsgBox "Starting to get Place data:  clearing scratch file"
    '
    MsgBox "Building address table"
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR"
    cmdSQL.Execute tRecDeleted
    '
    '   get all the lower level address IDs, if needed
    '
    If ChkSubUnits.Value Then
        MsgBox "adding addresses"
        tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT ZZZ_BELONGS_TO.c_addr_id " + _
            "FROM ZZ_SCRATCH_ADDR_LIST INNER JOIN ZZZ_BELONGS_TO ON ZZ_SCRATCH_ADDR_LIST.c_addr_id = " + _

```

```

"ZZZ_BELONGS_TO.c_belongs_to"
Else
    tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT c_addr_id FROM ZZ_SCRATCH_ADDR_LIST"
End If

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'
' see if we need to use the historical XY search
'
If ChkXYRef.Value Then
'
' the strategy here is to dump the IDs to ZZ_ADDRESSES then copy to ZZ_SCRATCH_ADDR_LIST
' (I borrow ZZ_ADDRESSES from the Pick Addresses form in order to keep the initial selection
' of addresses for the query intact.)
'
' zap the list
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' run the query
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT DISTINCT ADDR_CODES.c_addr_id " + _
"FROM ADDR_CODES, ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES AS ADDR_CODES_1 ON " + _
"ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES_1.c_addr_id " + _
"WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.03) And " + _
"(ADDR_CODES.x_coord)<=([ADDR_CODES_1].[x_coord]+0.03)) AND " + _
"((ADDR_CODES.y_coord)>=([ADDR_CODES_1].[y_coord]-0.03) And " + _
"(ADDR_CODES.y_coord)<=([ADDR_CODES_1].[y_coord]+0.03)))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now get the address IDs from the initial list that have no xy coordinates
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT ZZ_SCRATCH_ADDR.c_addr_id " + _
"FROM ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES ON " + _
"ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES.c_addr_id " + _
"WHERE (((ADDR_CODES.x_coord) Is Null)) OR (((ADDR_CODES.y_coord) Is Null))"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' zap ZZ_SCRATCH_ADDR
'
tQueryStr = "DELETE * FROM ZZ_SCRATCH_ADDR"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' copy the list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT ZZ_ADDRESSES.c_addr_id " + _
"FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' zap the temporary list
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
End If
'
' Importing place names does not automatically reset to gUseAddrId = TRUE
'
gUseADDRID = True
End If
'
' now, if available, populate the initial ZZ_SCRATCH_PEOPLE with either a single record or with the list
'
If gUsePersonID Then
tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_node_dist ) " + _
"SELECT ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id, 0 as c_node_dist FROM ZZ_SCRATCH_IMPORT_PEOPLE"

'MsgBox "Populating people list table: " + tQueryStr

```

```

        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
    End If
    '
    ' define the copy query strings for the step of copying over the data
    ' Note that the JOIN constraints for copying from ZZZ_NONKIN_BIOG_ADDR are just:
    ' 1. Person ID
    ' 2. Link code (i.e association code)
    ' 3. Node ID (i.e. associate ID)
    ' This copies over ALL records that apply, so that the specificity of texts, years and place of associa
tion, etc. are captured here.

    tQueryCopyNonkinStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_node_id, c_person_id, c_link_code, c_link_chn, c_lin
k_desc, c_kin_code, c_kin_id, c_kin_name, " + _
        "c_kin_chn, c_assoc_kin_code, c_assoc_kin_desc, c_assoc_kin_desc_chn, c_assoc_kin_id, c_assoc_kin_nam
e, c_assoc_kin_chn, c_litgenre_code, " + _
        "c_litgenre_desc, c_litgenre_desc_chn, c_occasion_code, c_occasion_desc, c_occasion_desc_chn, c_topic
_code, c_topic_desc, c_topic_desc_chn, " + _
        "c_inst_code, c_inst_name_hz, c_text_title, c_assoc_claimer_id, c_assoc_claimer_name, c_assoc_claimer
_name_chn, c_kin_desc, c_kin_desc_chn, " + _
        "c_inst_name_py, c_link_count, type_id, c_edge_dist, c_link_type, c_distance, c_source, c_link_first
year, c_link_last_year, c_link_addr_id ) " + _
        "SELECT DISTINCT NA.c_node_id, NA.c_personid, NA.c_link_code, NA.c_link_chn, NA.c_link_desc, NA.c_kin_cod
e, NA.c_kin_id, NA.c_kin_name, NA.c_kin_chn, " + _
        "NA.c_assoc_kin_code, NA.c_assoc_kinrel, NA.c_assoc_kinrel_chn, NA.c_assoc_kin_id, NA.c_assoc_kin_nam
e, NA.c_assoc_kin_chn, NA.c_litgenre_code, " + _
        "NA.c_litgenre_desc, NA.c_litgenre_desc_chn, NA.c_occasion_code, NA.c_occasion_desc, NA.c_occasion
_desc_chn, NA.c_topic_code, NA.c_topic_desc, " + _
        "NA.c_topic_desc_chn, NA.c_inst_code, NA.c_inst_name_hz, NA.c_text_title, NA.c_assoc_claimer_id, NA.c
_assoc_claimer_name, NA.c_assoc_claimer_chn, " + _
        "NA.c_kinrel, NA.c_kinrel_chn, NA.c_inst_name_py, NA.c_link_count, 'N' AS type_id, ZZ_NETWORK_LIST.c
distance, 'N' AS c_link_type, NA.c_distance, " + _
        "NA.c_source, NA.c_assoc_first_year, NA.c_assoc_last_year, NA.c_assoc_addr_id " + _
        "FROM ZZZ_NONKIN_BIOG_ADDR AS NA INNER JOIN ZZ_NETWORK_LIST ON (NA.c_node_id = ZZ_NETWORK_LIST.c_node_id)
AND (NA.c_link_code = ZZ_NETWORK_LIST.c_edge_id) " + _
        "AND (NA.c_personid = ZZ_NETWORK_LIST.c_personid) " + _
        "WHERE (((ZZ_NETWORK_LIST.c_edge_type)='N'))"

    '
    ' Similarly, note that the JOIN constraints for copying from ZZZ_KIN_BIOG_ADDR are just:
    ' 1. Person ID
    ' 2. Link code (i.e kinship code)
    ' 3. Node ID (i.e. kin ID)
    ' This copies over all records that apply

    tQueryCopyKinStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_node_id, c_link_code, c_source, c_text_tit
le, c_link_count, c_distance, " + _
        "c_link_type, type_id, c_link_chn, c_link_desc ) " + _
        "SELECT ZZZ_KIN_BIOG_ADDR.c_personid, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_code, ZZZ_KIN
_BIOG_ADDR.c_source, " + _
        "ZZZ_KIN_BIOG_ADDR.c_title_chn, ZZZ_KIN_BIOG_ADDR.c_link_count, ZZZ_KIN_BIOG_ADDR.c_distance, ZZ_NETW
ORK_LIST.c_edge_type, 'K' AS type_id, " + _
        "ZZZ_KIN_BIOG_ADDR.c_link_chn, ZZZ_KIN_BIOG_ADDR.c_link_desc " + _
        "FROM ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_NETWORK_LIST ON (ZZZ_KIN_BIOG_ADDR.c_personid = ZZ_NETWORK_LIST.c_p
ersonid) AND " + _
        "(ZZZ_KIN_BIOG_ADDR.c_node_id = ZZ_NETWORK_LIST.c_node_id) AND (ZZZ_KIN_BIOG_ADDR.c_link_code = ZZ_NE
TWORK_LIST.c_edge_id) " + _
        "WHERE (((ZZ_NETWORK_LIST.c_edge_type)='K'));"

    '
    ' initialize basic components of Query strings
    '
    ' INSERT INTO ZZ_NETWORK_LIST_TMP ( c_personid, c_node_id, c_edge_id, c_edge_type, c_up_total, c_down_total,
    ' c_mar_total, c_col_total, c_distance )
    ' SELECT ZZ_NETWORK_LIST.c_node_id, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_code, "K" AS Expr1
    '
    ' ZZ_NETWORK_LIST.c_up_total+ZZZ_KIN_BIOG_ADDR.c_upstep AS c_up_total,
    ' ZZ_NETWORK_LIST.c_down_total+ZZZ_KIN_BIOG_ADDR.c_dwnstep AS c_down_total,
    ' ZZ_NETWORK_LIST.c_mar_total+ZZZ_KIN_BIOG_ADDR.c_marstep AS c_mar_total,
    ' ZZ_NETWORK_LIST.c_col_total+ZZZ_KIN_BIOG_ADDR.c_colstep AS c_col_total,
    ' [ZZ_NETWORK_LIST].[c_distance]+1 AS c_distance
    '
    ' FROM ZZ_SCRATCH_ADDR INNER JOIN (ZZ_NETWORK_LIST INNER JOIN ZZZ_KIN_BIOG_ADDR ON
    ' ZZ_NETWORK_LIST.c_node_id = ZZZ_KIN_BIOG_ADDR.c_personid) ON ZZ_SCRATCH_ADDR.c_addr_id =
    ' ZZZ_KIN_BIOG_ADDR.c_node_addr_id
    '
    ' WHERE ((([ZZ_NETWORK_LIST].[c_up_total]+[ZZZ_KIN_BIOG_ADDR].[c_upstep])<4)
    ' AND ([ZZ_NETWORK_LIST].[c_down_total]+[ZZZ_KIN_BIOG_ADDR].[c_dwnstep])<4)
    ' AND ([ZZ_NETWORK_LIST].[c_mar_total]+[ZZZ_KIN_BIOG_ADDR].[c_marstep])<2)

```

```

' AND (([ZZ_NETWORK_LIST].[c_col_total]+[ZZZ_KIN_BIOG_ADDR].[c_colstep])<2))
' AND ([ZZ_NETWORK_LIST].[c_distance]=0)
'
'
' INSERT INTO ZZ_NETWORK_LIST_TMP ( c_personid, c_node_id, c_edge_id, c_edge_type, c_up_total,
' c_down_total, c_mar_total, c_col_total, c_distance )
' SELECT DISTINCT ZZ_NETWORK_LIST.c_node_id, ZZZ_NONKIN_BIOG_ADDR.c_node_id,
' ZZZ_NONKIN_BIOG_ADDR.c_link_code, "N" AS c_edge_type, ZZ_NETWORK_LIST.c_up_total,
' ZZ_NETWORK_LIST.c_down_total, ZZ_NETWORK_LIST.c_mar_total, ZZ_NETWORK_LIST.c_col_total,
' [ZZ_NETWORK_LIST].[c_distance]+1 AS c_distance
'
' FROM ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_ASSOC_FILTER INNER JOIN (ZZ_NETWORK_LIST INNER
' JOIN ZZZ_NONKIN_BIOG_ADDR ON ZZ_NETWORK_LIST.c_node_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) ON
' ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) ON
' ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id
'
' WHERE ((([ZZ_NETWORK_LIST].[c_distance]+1)=1));

tQuerySelectFirstNonkin = "INSERT INTO ZZ_NETWORK_LIST_TMP ( c_personid, c_node_id, c_edge_id, " + _
" c_edge_type, c_up_total, c_down_total, c_mar_total, c_col_total, c_distance ) " + _
"SELECT DISTINCT ZZZ_NONKIN_BIOG_ADDR.c_personid, ZZZ_NONKIN_BIOG_ADDR.c_node_id, " + _
" ZZZ_NONKIN_BIOG_ADDR.c_link_code, 'N' AS c_edge_type, 0 AS c_up_total, 0 AS c_down_total, " + _
" 0 AS c_mar_total, 0 AS c_col_total, 0 AS c_distance "

gQuerySelectNonkin = "INSERT INTO ZZ_NETWORK_LIST_TMP ( c_personid, c_node_id, c_edge_id, c_edge_type, " + _
" c_up_total, c_down_total, c_mar_total, c_col_total, c_distance ) " + _
"SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id, ZZZ_NONKIN_BIOG_ADDR.c_node_id, " + _
" ZZZ_NONKIN_BIOG_ADDR.c_link_code, 'N' AS c_edge_type, 0 AS c_up_total, " + _
" 0 AS c_down_total, 0 AS c_mar_total, 0 AS c_col_total, " + _
" [ZZ_SCRATCH_PEOPLE].[c_node_dist] AS c_distance "

tQuerySelectFirstKin = "INSERT INTO ZZ_NETWORK_LIST_TMP ( c_personid, c_node_id, c_edge_id, c_edge_type, " + _
" c_up_total, c_down_total, c_mar_total, c_col_total, c_distance ) " + _
"SELECT DISTINCT ZZZ_KIN_BIOG_ADDR.c_personid, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_code
, " + _
" 'K' AS Expr1, ZZZ_KIN_BIOG_ADDR.c_upstep, ZZZ_KIN_BIOG_ADDR.c_dwnstep, ZZZ_KIN_BIOG_ADDR.c_marstep,
" + _
" ZZZ_KIN_BIOG_ADDR.c_colstep, 0 AS c_distance "

gQuerySelectKin = "INSERT INTO ZZ_NETWORK_LIST_TMP ( c_personid, c_node_id, c_edge_id, c_edge_type, " + _
" c_up_total, c_down_total, c_mar_total, c_col_total, c_distance ) " + _
"SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_cod
e, " + _
" 'K' AS Expr1, ZZZ_KIN_BIOG_ADDR.c_upstep AS c_up_total, " + _
" ZZZ_KIN_BIOG_ADDR.c_dwnstep AS c_down_total, " + _
" ZZZ_KIN_BIOG_ADDR.c_marstep AS c_mar_total, " + _
" ZZZ_KIN_BIOG_ADDR.c_colstep AS c_col_total, " + _
" [ZZ_SCRATCH_PEOPLE].[c_node_dist] AS c_distance "

'gQueryIndexYear = "((c_index_year)>= " + Str(TxtFrom.Value) + _
" And (c_index_year)<=" + Str(TxtTo.Value) + _
" AND (c_node_index_year)>=" + Str(TxtFrom.Value) + _
" And (c_node_index_year)<=" + Str(TxtTo.Value) + ") "

'
gQueryKindist = "((c_upstep)<=" + Str(TxtMaxUp.Value) + ") " + _
" AND ((c_dwnstep)<=" + Str(TxtMaxDwn.Value) + ") " + _
" AND ((c_marstep)<=" + Str(TxtMaxMar.Value) + ") " + _
" AND ((c_colstep)<=" + Str(TxtMaxCol.Value) + ") "

' The various options for the tables used in FROM statements for restricted search

' (1) Non-restricted ASSOC starting with a people list (with and without dynasty restrictions)

tQueryAssocFromStr = "FROM ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid "

tQueryAssocDynastyFromStr = "FROM DYNASTIES AS DYNASTIES_1 INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIE
S INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy "

' (2) ASSOC starting with a people list restricted by a list of association codes (with and without dynasty
restrictions)

tQueryAssocAssocFromStr = "FROM ZZ_SCRATCH_ASSOC_FILTER INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_NONKIN_B
IOG_ADDR " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code "

```

```

tQueryAssocAssocDynastyFromStr = "FROM ZZ_SCRATCH_ASSOC_FILTER INNER JOIN (DYNASTIES AS DYNASTIES_1 " + _
    "INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
    "ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
    "ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
    "ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code "

' (3) ASSOC starting with a people list restricted by a list of address codes (with and without dynasty restrictions)

tQueryAssocAddrFromStr = "FROM ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_NONKIN_BIOG_ADDR
" + _
    "ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
    "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id "

tQueryAssocAddrDynastyFromStr = "FROM (DYNASTIES AS DYNASTIES_1 INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
    "ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
    "ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
    "INNER JOIN ZZ_SCRATCH_ADDR " + _
    "ON ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id = ZZ_SCRATCH_ADDR.c_addr_id "

' (4) ASSOC starting with a people list restricted by lists of association codes AND address codes (with and without dynasty restrictions)

tQueryAssocAddrAssocFromStr = "FROM ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_ASSOC_FILTER INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
    "ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) " + _
    "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id "

tQueryAssocAddrAssocDynastyFromStr = "FROM ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_ASSOC_FILTER " + _
    "INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
    "ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
    "ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
    "ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) " + _
    "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id "

' (5) ASSOC restricted by NOTHING AT ALL (with and without dynasty restrictions)

tQueryAssocFirstFromStr = "FROM ZZZ_NONKIN_BIOG_ADDR "

tQueryAssocDynastyFirstFromStr = "FROM DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
    "ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy "

' (6) ASSOC restricted by a list of association codes (with and without dynasty restrictions)

tQueryAssocFirstAssocFromStr = "FROM ZZ_SCRATCH_ASSOC_FILTER INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code "

tQueryAssocFirstAssocDynastyFromStr = "FROM ZZ_SCRATCH_ASSOC_FILTER INNER JOIN (DYNASTIES AS DYNASTIES_1 " + _
    "INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
    "ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
    "ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code "

' (7) ASSOC restricted by a list of address codes (with and without dynasty restrictions)

tQueryAssocFirstAddrFromStr = "FROM ZZ_SCRATCH_ADDR AS ZZ_SCRATCH_ADDR_1 INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id) " + _
    "ON ZZ_SCRATCH_ADDR_1.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_person_addr_id "

tQueryAssocFirstAddrDynastyFromStr = "FROM ZZ_SCRATCH_ADDR AS ZZ_SCRATCH_ADDR_1 INNER JOIN (ZZ_SCRATCH_ADDR " + _
    "INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
    "ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
    "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id) " + _
    "ON ZZ_SCRATCH_ADDR_1.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_person_addr_id "

' (8) ASSOC restricted by a list of association codes AND address codes (with and without dynasty restrictions)

```



```

tQueryAssocFirstAddrAssocFromStr = "FROM ZZ_SCRATCH_ADDR AS ZZ_SCRATCH_ADDR_1 INNER JOIN " + _
"(ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_ASSOC_FILTER INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id) " + _
"ON ZZ_SCRATCH_ADDR_1.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_person_addr_id "

tQueryAssocFirstAddrAssocDynastyFromStr = "FROM ZZ_SCRATCH_ASSOC_FILTER INNER JOIN (ZZ_SCRATCH_ADDR AS ZZ_SCRATCH_ADDR_1 " + _
"INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
"ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id) " + _
"ON ZZ_SCRATCH_ADDR_1.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_person_addr_id) " + _
"ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code "

' (9) Non-restricted KINSHIP starting with a people list (with and without dynasty restrictions)

tQueryKinFromStr = "FROM ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_KIN_BIOG_ADDR.c_personid "

tQueryKinDynastyFromStr = "FROM ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_KIN_BIOG_ADDR.c_dy) " + _
"ON DYNASTIES_1.c_dy = ZZZ_KIN_BIOG_ADDR.c_node_dy) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_KIN_BIOG_ADDR.c_personid "

' (10) KINSHIP starting with a people list restricted by a list of address codes (with and without dynasty restrictions)

tQueryKinAddrFromStr = "FROM ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_KIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_KIN_BIOG_ADDR.c_node_addr_id "

tQueryKinAddrDynastyFromStr = "FROM ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES AS DYNASTIES_1 " + _
"INNER JOIN (DYNASTIES INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_KIN_BIOG_ADDR.c_dy) " + _
"ON DYNASTIES_1.c_dy = ZZZ_KIN_BIOG_ADDR.c_node_dy) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_KIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_KIN_BIOG_ADDR.c_node_addr_id "

' (11) KINSHIP restricted by a list of address codes (with and without dynasty restrictions)

tQueryKinFirstAddrFromStr = "FROM ZZ_SCRATCH_ADDR AS ZZ_SCRATCH_ADDR_1 INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_KIN_BIOG_ADDR.c_node_addr_id) " + _
"ON ZZ_SCRATCH_ADDR_1.c_addr_id = ZZZ_KIN_BIOG_ADDR.c_person_addr_id "

tQueryKinFirstAddrDynastyFromStr = "FROM ZZ_SCRATCH_ADDR AS ZZ_SCRATCH_ADDR_1 INNER JOIN (ZZ_SCRATCH_ADDR " + _
"INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_KIN_BIOG_ADDR.c_dy) " + _
"ON DYNASTIES_1.c_dy = ZZZ_KIN_BIOG_ADDR.c_node_dy) " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_KIN_BIOG_ADDR.c_node_addr_id) " + _
"ON ZZ_SCRATCH_ADDR_1.c_addr_id = ZZZ_KIN_BIOG_ADDR.c_person_addr_id "

' (12) Final restricted search: Non-restricted ASSOC starting with a people list (with and without dynasty restrictions)

tQueryAssocLastFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN " + _
"(ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_node_id "

tQueryAssocDynastyLastFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN " + _
"(ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
"ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_node_id "

' (13) Final restricted search: ASSOC starting with a people list restricted by a list of association codes (with and without dynasty restrictions)

tQueryAssocLastAssocFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN " + _
"(ZZ_SCRATCH_PEOPLE INNER JOIN (ZZ_SCRATCH_ASSOC_FILTER INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) " + _

```

```
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_node_id " _
```

```
tQueryAssocLastAssocDynastyFromStr = "FROM ZZ_SCRATCH_ASSOC_FILTER INNER JOIN (ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 " + _
"INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
"ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_node_id) " + _
"ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code " _
```

' (14) Final restricted search: ASSOC starting with a people list restricted by a list of address code (with and without dynasty restrictions)

```
tQueryAssocLastAddrFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN " + _
"(ZZ_SCRATCH_PEOPLE INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_node_id " _
```

```
tQueryAssocLastAddrDynastyFromStr = "FROM ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 " + _
"INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
"ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_node_id) " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id " _
```

' (15) Final restricted search: ASSOC starting with a people list restricted by lists of association codes AND address codes (with and without dynasty restrictions)

```
tQueryAssocLastAddrAssocFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN " + _
"(ZZ_SCRATCH_PEOPLE INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_ASSOC_FILTER INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_node_id " _
```

```
tQueryAssocLastAddrAssocDynastyFromStr = "FROM ZZ_SCRATCH_ASSOC_FILTER INNER JOIN (ZZ_SCRATCH_ADDR " + _
"INNER JOIN (ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES AS DYNASTIES_1 " + _
"INNER JOIN (DYNASTIES INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) " + _
"ON DYNASTIES_1.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_node_dy) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_node_id) " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id) " + _
"ON ZZ_SCRATCH_ASSOC_FILTER.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code " _
```

' (16) Final restricted search: Non-restricted KINSHIP starting with a people list (with and without dynasty restrictions)

```
tQueryKinLastFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN " + _
"(ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_KIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_KIN_BIOG_ADDR.c_node_id " _
```

```
tQueryKinDynastyLastFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN (ZZ_SCRATCH_PEOPLE " + _
"INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
"ON DYNASTIES.c_dy = ZZZ_KIN_BIOG_ADDR.c_dy) " + _
"ON DYNASTIES_1.c_dy = ZZZ_KIN_BIOG_ADDR.c_node_dy) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_KIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_KIN_BIOG_ADDR.c_node_id " _
```

' (17) Final restricted search: KINSHIP starting with a people list restricted by a list of address codes (with and without dynasty restrictions)

```
tQueryKinLastAddrFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN " + _
"(ZZ_SCRATCH_PEOPLE INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_KIN_BIOG_ADDR.c_node_addr_id) " + _
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_KIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_KIN_BIOG_ADDR.c_node_id " _
```

```
tQueryKinLastAddrDynastyFromStr = "FROM ZZ_SCRATCH_ADDR INNER JOIN (ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1
```

```

51
" + -
"INNER JOIN (ZZ_SCRATCH_PEOPLE INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ZZZ_KIN_BIOG_ADDR " + -
"ON DYNASTIES.c_dy = ZZZ_KIN_BIOG_ADDR.c_dy) " + -
"ON DYNASTIES_1.c_dy = ZZZ_KIN_BIOG_ADDR.c_node_dy) " + -
"ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_KIN_BIOG_ADDR.c_personid) " + -
"ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZZ_KIN_BIOG_ADDR.c_node_id) " + -
"ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_KIN_BIOG_ADDR.c_node_addr_id "

' now determine the tables and constraints that need to be included in the query when building the WHERE part of the query
tNonkinWhereQueryStr = "Where ("
tKinWhereQueryStr = "Where ("

If gUseIndexYears Then
    tNonkinWhereQueryStr = tNonkinWhereQueryStr +
        "((ZZZ_NONKIN_BIOG_ADDR.c_index_year)>= " + Str(TxtFrom.Value) + -
        " And (ZZZ_NONKIN_BIOG_ADDR.c_index_year)<=" + Str(TxtTo.Value) + -
        " AND (ZZZ_NONKIN_BIOG_ADDR.c_node_index_year)>=" + Str(TxtFrom.Value) + -
        " And (ZZZ_NONKIN_BIOG_ADDR.c_node_index_year)<=" + Str(TxtTo.Value) + ") "
    tKinWhereQueryStr = tKinWhereQueryStr +
        "((ZZZ_KIN_BIOG_ADDR.c_index_year)>=" + Str(TxtFrom.Value) + -
        " And (ZZZ_KIN_BIOG_ADDR.c_index_year)<=" + Str(TxtTo.Value) + -
        " AND (ZZZ_KIN_BIOG_ADDR.c_node_index_year)>=" + Str(TxtFrom.Value) + -
        " And (ZZZ_KIN_BIOG_ADDR.c_node_index_year)<=" + Str(TxtTo.Value) + ") "
ElseIf gUseDynasties Then
    ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
    '
    ' WHERE (((DYNASTIES.c_start)<5) AND ((DYNASTIES.c_end)>=6) AND ((DYNASTIES_1.c_start)<7) AND ((DYNASTIES_1.c_end)>=8))
    '
    tStrToDynastyEnd = Str(gToDynastyEnd)
    tStrFromDynastyBegin = Str(gFromDynastyBegin)

    If gFromDynasty = -2 Then
        tNonkinWhereQueryStr = tNonkinWhereQueryStr +
            "((ZZZ_NONKIN_BIOG_ADDR.c_dy) > 0 AND (ZZZ_NONKIN_BIOG_ADDR.c_node_dy) > 0 ) "
        tKinWhereQueryStr = tKinWhereQueryStr +
            "((ZZZ_KIN_BIOG_ADDR.c_dy) > 0 AND (ZZZ_KIN_BIOG_ADDR.c_node_dy) > 0 ) "
    ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
        tNonkinWhereQueryStr = tNonkinWhereQueryStr +
            "((DYNASTIES.c_start)<" + tStrToDynastyEnd + " AND (DYNASTIES_1.c_start)<" + tStrToDynastyEnd +
") "
        tKinWhereQueryStr = tKinWhereQueryStr +
            "((DYNASTIES.c_start)<" + tStrToDynastyEnd + " AND (DYNASTIES_1.c_start)<" + tStrToDynastyEnd +
") "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tNonkinWhereQueryStr = tNonkinWhereQueryStr +
            "((DYNASTIES.c_end)>" + tStrFromDynastyBegin + " AND (DYNASTIES_1.c_end)>" + tStrFromDynastyBegin
n + ") "
        tKinWhereQueryStr = tKinWhereQueryStr +
            "((DYNASTIES.c_end)>" + tStrFromDynastyBegin + " AND (DYNASTIES_1.c_end)>" + tStrFromDynastyBegin
n + ") "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tNonkinWhereQueryStr = tNonkinWhereQueryStr +
            "((DYNASTIES.c_dy)=" + Str(gToDynasty) + ") "
        tKinWhereQueryStr = tKinWhereQueryStr +
            "((DYNASTIES.c_dy)=" + Str(gToDynasty) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tNonkinWhereQueryStr = tNonkinWhereQueryStr +
            "((DYNASTIES.c_end)>" + tStrFromDynastyBegin + " AND (DYNASTIES_1.c_end)>" + tStrFromDynastyBegin
+ ") AND " + -
            "((DYNASTIES.c_start)<" + tStrToDynastyEnd + " AND (DYNASTIES_1.c_start)<" + tStrToDynastyEnd +
") "
        tKinWhereQueryStr = tKinWhereQueryStr +
            "((DYNASTIES.c_end)>" + tStrFromDynastyBegin + " AND (DYNASTIES_1.c_end)>" + tStrFromDynastyBegin
+ ") AND " + -
            "((DYNASTIES.c_start)<" + tStrToDynastyEnd + " AND (DYNASTIES_1.c_start)<" + tStrToDynastyEnd +
") "
    End If
End If

If ChkMale.Value = tTrue And ChkFemale.Value = tFalse Then

    If gUseIndexYears Or (gUseDynasties And Not (gFromDynasty = -1 And gToDynasty = -1)) Then
        tNonkinWhereQueryStr = tNonkinWhereQueryStr + " AND "
        tKinWhereQueryStr = tKinWhereQueryStr + " AND "
    End If
End If

```

```

End If
tNonkinWhereQueryStr = tNonkinWhereQueryStr + "((ZZZ_NONKIN_BIOG_ADDR.c_node_female)=False) "
tKinWhereQueryStr = tKinWhereQueryStr + "((ZZZ_KIN_BIOG_ADDR.c_node_female)=False) "

ElseIf ChkMale.Value = tFalse And ChkFemale.Value = tTrue Then

    If gUseIndexYears Or (gUseDynasties And Not (gFromDynasty = -1 And gToDynasty = -1)) Then
        tNonkinWhereQueryStr = tNonkinWhereQueryStr + " AND "
        tKinWhereQueryStr = tKinWhereQueryStr + " AND "
    End If
    tNonkinWhereQueryStr = tNonkinWhereQueryStr + "((ZZZ_NONKIN_BIOG_ADDR.c_node_female)=True) "
    tKinWhereQueryStr = tKinWhereQueryStr + "((ZZZ_KIN_BIOG_ADDR.c_node_female)=True) "

End If

If Not (tKinWhereQueryStr = "WHERE (") Then
    tKinWhereQueryStr = tKinWhereQueryStr + "AND "
End If

If ChkKinshipParam.Value = tTrue Then
    MsgBox "Up = " + Str(TxtMaxUp.Value) + " Down = " + Str(TxtMaxDwn.Value) + _
        " Mar = " + Str(Me.TxtMaxMar.Value) + " Col = " + Str(Me.TxtMaxCol.Value)
    tKinWhereFirstQueryStr = tKinWhereQueryStr + _
        "([ZZZ_KIN_BIOG_ADDR].[c_upstep]<= " + Str(TxtMaxUp.Value) + ") " + _
        "AND ([ZZZ_KIN_BIOG_ADDR].[c_dwnstep]<= " + Str(TxtMaxDwn.Value) + ") " + _
        "AND ([ZZZ_KIN_BIOG_ADDR].[c_marstep]<= " + Str(Me.TxtMaxMar.Value) + ") " + _
        "AND ([ZZZ_KIN_BIOG_ADDR].[c_colstep]<= " + Str(Me.TxtMaxCol.Value) + ") "
    tKinWhereQueryStr = tKinWhereFirstQueryStr + "AND ([ZZ_SCRATCH_PEOPLE].[c_node_dist]="

Else
    tKinWhereQueryStr = tKinWhereQueryStr + "([ZZ_SCRATCH_PEOPLE].[c_node_dist]="
    If tKinWhereQueryStr = " WHERE (" Then
        tKinWhereFirstQueryStr = ""
    Else
        tKinWhereFirstQueryStr = tKinWhereQueryStr
    End If
End If
'
' In the new version, I filter out all node IDs with 0 for the non-kin search
'

If tNonkinWhereQueryStr = "WHERE (" Then
    tNonkinWhereQueryStr = "WHERE ((ZZZ_NONKIN_BIOG_ADDR.c_node_id > 0) AND ([ZZ_SCRATCH_PEOPLE].[c_node_dist]
]="
    tNonkinWhereFirstQueryStr = "WHERE (ZZZ_NONKIN_BIOG_ADDR.c_node_id > 0)"
Else
    tNonkinWhereFirstQueryStr = tNonkinWhereQueryStr + " AND (ZZZ_NONKIN_BIOG_ADDR.c_node_id > 0))"
    tNonkinWhereQueryStr = tNonkinWhereQueryStr + "AND (ZZZ_NONKIN_BIOG_ADDR.c_node_id > 0) AND ([ZZ_SCRATCH_
PEOPLE].[c_node_dist]="
End If

' Now handle the 8 combinations of possibilities in defining the query using the three pieces

If gUseFilter = tTrue Then

    If gUsePersonID Then
        If gUseADDRID Then
            ' uses ZZ_NETWORK_LIST from the start, uses the address and the assoc filters
            '
            MsgBox "This query uses people, address, and assoc filter..."
            If Me.ChkNonKin.Value = tTrue Then
                If Me.ChkPlaceLimit.Value Then
                    If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                        tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAddrAssocDynastyFromStr + tNonkinWh
ereQueryStr
                        tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrAssocDynastyFromStr + t
NonkinWhereQueryStr
                    Else
                        tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAddrAssocFromStr + tNonkinWhereQuer
yStr
                        tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrAssocFromStr + tNonkinW
hereQueryStr
                    End If
                Else
                    If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                        tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAssocDynastyFromStr + tNonkinWhereQ
ueryStr

```

```

53
    If gMaxNodeDist = 0 Then
        tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrAssocDynastyFromStr
+ tNonkinWhereQueryStr
    Else
        tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAssocDynastyFromStr + t
NonkinWhereQueryStr
    End If
Else
    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAssocFromStr + tNonkinWhereQueryStr
    If gMaxNodeDist = 0 Then
        tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrAssocFromStr + tNon
kinWhereQueryStr
    Else
        tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAssocFromStr + tNonkinW
hereQueryStr
    End If
End If
If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
    tQueryNonkinFirstStr = gQuerySelectNonkin + tQueryAssocAddrAssocDynastyFromStr + _
    tNonkinWhereQueryStr + "0))"
Else
    tQueryNonkinFirstStr = gQuerySelectNonkin + tQueryAssocAddrAssocFromStr + _
    tNonkinWhereQueryStr + "0))"
End If
End If

If ChkKin.Value = tTrue Then
    If Me.ChkPlaceLimit.Value Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryKinStr = gQuerySelectKin + tQueryKinAddrDynastyFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrDynastyFromStr + tKinWhereQuery
Str
        Else
            tQueryKinStr = gQuerySelectKin + tQueryKinAddrFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrFromStr + tKinWhereQueryStr
        End If
    Else
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryKinStr = gQuerySelectKin + tQueryKinDynastyFromStr + tKinWhereQueryStr
            If gMaxNodeDist = 0 Then
                tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrDynastyFromStr + tKinWhereQ
ueryStr
            Else
                tQueryKinLastStr = gQuerySelectKin + tQueryKinDynastyLastFromStr + tKinWhereQuery
Str
            End If
        Else
            tQueryKinStr = gQuerySelectKin + tQueryKinFromStr + tKinWhereQueryStr
            If gMaxNodeDist = 0 Then
                tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrFromStr + tKinWhereQueryStr
            Else
                tQueryKinLastStr = gQuerySelectKin + tQueryKinLastFromStr + tKinWhereQueryStr
            End If
        End If
    End If
    If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
        tQueryKinFirstStr = gQuerySelectKin + tQueryKinAddrDynastyFromStr + tKinWhereQueryStr + "
0))"
    Else
        tQueryKinFirstStr = gQuerySelectKin + tQueryKinAddrFromStr + tKinWhereQueryStr + "0))"
    End If
End If
Else
    '
    ' uses ZZ_NETWORK_LIST from the start, uses the assoc filters but NOT the address
    '
    'MsgBox "This query uses people, (NO address), and assoc filter..."
    If Me.ChkNonKin.Value = tTrue Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAssocDynastyFromStr + tNonkinWhereQuery
Str
            tQueryNonkinFirstStr = tQueryNonkinStr + "0))"
            tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAssocDynastyFromStr + tNonkinWh
ereQueryStr
        Else
            tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAssocFromStr + tNonkinWhereQueryStr
            tQueryNonkinFirstStr = tQueryNonkinStr + "0))"
            tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAssocFromStr + tNonkinWhereQuer
yStr

```

```

End If
End If

If ChkKin.Value = tTrue Then
    If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
        tQueryKinStr = gQuerySelectKin + tQueryKinDynastyFromStr + tKinWhereQueryStr
        tQueryKinFirstStr = tQueryKinStr + "0))"
        tQueryKinLastStr = gQuerySelectKin + tQueryKinDynastyLastFromStr + tKinWhereQueryStr
    Else
        tQueryKinStr = gQuerySelectKin + tQueryKinFromStr + tKinWhereQueryStr
        tQueryKinFirstStr = tQueryKinStr + "0))"
        tQueryKinLastStr = gQuerySelectKin + tQueryKinLastFromStr + tKinWhereQueryStr
    End If
End If
End If
Else
    If gUseADDRID Then
        '
        ' does NOT use ZZ_NETWORK_LIST at first, uses the address and the assoc filters
        '
        'MsgBox "This query uses (NO people), address, and assoc filter..."
        If Me.ChkNonKin.Value = tTrue Then
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tQueryNonkinFirstStr = tQuerySelectFirstNonkin + tQueryAssocFirstAddrAssocDynastyFromStr + _
                    tNonkinWhereFirstQueryStr
            Else
                tQueryNonkinFirstStr = tQuerySelectFirstNonkin + tQueryAssocFirstAddrAssocFromStr + _
                    tNonkinWhereFirstQueryStr
            End If

            If ChkPlaceLimit.Value Then
                If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAddrAssocDynastyFromStr + tNonkinWhereQueryStr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrAssocDynastyFromStr + tNonkinWhereQueryStr
                Else
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAddrAssocFromStr + tNonkinWhereQueryStr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrAssocFromStr + tNonkinWhereQueryStr
                End If
            Else
                If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAssocDynastyFromStr + tNonkinWhereQueryStr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAssocDynastyFromStr + tNonkinWhereQueryStr
                Else
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAssocFromStr + tNonkinWhereQueryStr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAssocFromStr + tNonkinWhereQueryStr
                End If
            End If
        End If
    End If

    If ChkKin.Value = tTrue Then
        If ChkPlaceLimit.Value Then
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tQueryKinStr = gQuerySelectKin + tQueryKinAddrDynastyFromStr + tKinWhereQueryStr
                tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrDynastyFromStr + tKinWhereQueryStr
            Else
                tQueryKinStr = gQuerySelectKin + tQueryKinAddrFromStr + tKinWhereQueryStr
                tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrFromStr + tKinWhereQueryStr
            End If
        Else
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tQueryKinStr = gQuerySelectKin + tQueryKinDynastyFromStr + tKinWhereQueryStr
                tQueryKinLastStr = gQuerySelectKin + tQueryKinDynastyLastFromStr + tKinWhereQueryStr
            Else
                tQueryKinStr = gQuerySelectKin + tQueryKinFromStr + tKinWhereQueryStr
                tQueryKinLastStr = gQuerySelectKin + tQueryKinLastFromStr + tKinWhereQueryStr
            End If
        End If

        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryKinFirstStr = gQuerySelectKin + tQueryKinAddrDynastyFromStr + tKinWhereQueryStr + "0))"

```

```

Else
    tQueryKinFirstStr = gQuerySelectKin + tQueryKinAddrFromStr + tKinWhereQueryStr + "0))"
End If
End If
Else
    '
    ' does NOT use ZZ_NETWORK_LIST from the start, uses the assoc filter but NOT address
    '
    'MsgBox "This query uses (NO people), (NO address), and assoc filter..."
    If Me.ChkNonKin.Value = tTrue Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryNonkinFirstStr = tQuerySelectFirstNonkin + tQueryAssocFirstAssocDynastyFromStr + _
                tNonkinWhereFirstQueryStr
            tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAssocDynastyFromStr + tNonkinWhereQuery
Str
            tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAssocDynastyFromStr + tNonkinWh
ereQueryStr
        Else
            tQueryNonkinFirstStr = tQuerySelectFirstNonkin + tQueryAssocFirstAssocFromStr + _
                tNonkinWhereFirstQueryStr
            tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAssocFromStr + tNonkinWhereQueryStr
            tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAssocFromStr + tNonkinWhereQuer
yStr
        End If
    End If

    If ChkKin.Value = tTrue Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryKinStr = gQuerySelectKin + tQueryKinDynastyFromStr + tKinWhereQueryStr
            tQueryKinFirstStr = tQueryKinStr + "0))"
            tQueryKinLastStr = gQuerySelectKin + tQueryKinDynastyLastFromStr + tKinWhereQueryStr
        Else
            tQueryKinStr = gQuerySelectKin + tQueryKinFromStr + tKinWhereQueryStr
            tQueryKinFirstStr = tQueryKinStr + "0))"
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastFromStr + tKinWhereQueryStr
        End If
    End If
End If
Else
    If gUsePersonID Then
        If gUseADDRID Then
            '
            ' uses ZZ_NETWORK_LIST from the start, uses the address but NOT the assoc filters
            '
            'MsgBox "This query uses people, address, and (NO assoc filter)..."
            If Me.ChkNonKin.Value = tTrue Then
                If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                    tQueryNonkinFirstStr = gQuerySelectNonkin + tQueryAssocAddrDynastyFromStr + tNonkinWhereQ
queryStr + _
                        "0))"
                Else
                    tQueryNonkinFirstStr = gQuerySelectNonkin + tQueryAssocAddrFromStr + tNonkinWhereQueryStr
+ _
                        "0))"
                End If
            End If
            If ChkPlaceLimit.Value Then
                If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAddrDynastyFromStr + tNonkinWhereQu
eryStr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrDynastyFromStr + tNonki
nWhereQueryStr
                Else
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAddrFromStr + tNonkinWhereQueryStr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrFromStr + tNonkinWhereQueryStr
                End If
            Else
                If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocDynastyFromStr + tNonkinWhereQueryS
tr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocDynastyLastFromStr + tNonkinWhe
reQueryStr
                Else
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocFromStr + tNonkinWhereQueryStr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastFromStr + tNonkinWhereQuery
Str
                End If
            End If
        End If
    End If
End If

```

```

If ChkKin.Value = tTrue Then
    If ChkPlaceLimit.Value Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryKinStr = gQuerySelectKin + tQueryKinAddrDynastyFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrDynastyFromStr + tKinWhereQueryStr
        Else
            tQueryKinStr = gQuerySelectKin + tQueryKinAddrFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrFromStr + tKinWhereQueryStr
        End If
    Else
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryKinStr = gQuerySelectKin + tQueryKinDynastyFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinDynastyLastFromStr + tKinWhereQueryStr
        Else
            tQueryKinStr = gQuerySelectKin + tQueryKinFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastFromStr + tKinWhereQueryStr
        End If
    End If

    If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
        tQueryKinFirstStr = gQuerySelectKin + tQueryKinAddrDynastyFromStr + tKinWhereQueryStr + "
0))"
    Else
        tQueryKinFirstStr = gQuerySelectKin + tQueryKinAddrFromStr + tKinWhereQueryStr + "0))"
    End If
End If
Else
    '
    ' uses ZZ_NETWORK_LIST from the start but does NOT use EITHER the address or the assoc filters
    '
    MsgBox "This query uses people, (NO address), and (NO assoc filter)..."
    If Me.ChkNonKin.Value = tTrue Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocDynastyFromStr + tNonkinWhereQueryStr
            tQueryNonkinFirstStr = gQuerySelectNonkin + tQueryAssocDynastyFromStr + tNonkinWhereQueryStr
            tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocDynastyLastFromStr + tNonkinWhereQueryStr
        Else
            tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocFromStr + tNonkinWhereQueryStr
            tQueryNonkinFirstStr = gQuerySelectNonkin + tQueryAssocFromStr + tNonkinWhereQueryStr + "
0))"
            tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastFromStr + tNonkinWhereQueryStr
        End If
    End If

    If ChkKin.Value = tTrue Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryKinStr = gQuerySelectKin + tQueryKinDynastyFromStr + tKinWhereQueryStr
            tQueryKinFirstStr = tQueryKinStr + "0))"
            tQueryKinLastStr = gQuerySelectKin + tQueryKinDynastyLastFromStr + tKinWhereQueryStr
        Else
            tQueryKinStr = gQuerySelectKin + tQueryKinFromStr + tKinWhereQueryStr
            tQueryKinFirstStr = tQueryKinStr + "0))"
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastFromStr + tKinWhereQueryStr
        End If
    End If
End If
Else
    If gUseADDRID Then
        '
        ' does NOT use ZZ_NETWORK_LIST at first, uses the address but NOT the assoc filters
        '
        MsgBox "This query uses (NO people), address, and (NO assoc filter)..."
        If Me.ChkNonKin.Value = tTrue Then
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tQueryNonkinFirstStr = tQuerySelectFirstNonkin + tQueryAssocFirstAddrDynastyFromStr + _
                tNonkinWhereFirstQueryStr
            Else
                tQueryNonkinFirstStr = tQuerySelectFirstNonkin + tQueryAssocFirstAddrFromStr + _
                tNonkinWhereFirstQueryStr
            End If

            If ChkPlaceLimit.Value Then
                If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAddrDynastyFromStr + tNonkinWhereQueryStr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrDynastyFromStr + tNonkinWhereQueryStr
                Else
                    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocFromStr + tNonkinWhereQueryStr
                    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastFromStr + tNonkinWhereQueryStr
                End If
            End If
        End If
    End If
End If

```



```

57
nWhereQueryStr
Else
    tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocAddrFromStr + tNonkinWhereQueryStr
    tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastAddrFromStr + tNonkinWhereQ
ueryStr
End If
Else
    If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
        tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocDynastyFromStr + tNonkinWhereQueryS
tr
        tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocDynastyLastFromStr + tNonkinWhe
reQueryStr
    Else
        tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocFromStr + tNonkinWhereQueryStr
        tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastFromStr + tNonkinWhereQuery
Str
    End If
End If
End If

If ChkKin.Value = tTrue Then
    If ChkPlaceLimit.Value Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryKinStr = gQuerySelectKin + tQueryKinAddrDynastyFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrDynastyFromStr + tKinWhereQuery
Str
        Else
            tQueryKinStr = gQuerySelectKin + tQueryKinAddrFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastAddrFromStr + tKinWhereQueryStr
        End If
    Else
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryKinStr = gQuerySelectKin + tQueryKinDynastyFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinDynastyLastFromStr + tKinWhereQueryStr
        Else
            tQueryKinStr = gQuerySelectKin + tQueryKinFromStr + tKinWhereQueryStr
            tQueryKinLastStr = gQuerySelectKin + tQueryKinLastFromStr + tKinWhereQueryStr
        End If
    End If

    If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
        tQueryKinFirstStr = gQuerySelectKin + tQueryKinAddrDynastyFromStr + tKinWhereQueryStr + "
0))"
    Else
        tQueryKinFirstStr = gQuerySelectKin + tQueryKinAddrFromStr + tKinWhereQueryStr + "0))"
    End If
End If
Else
    '
    ' does NOT uses ZZ_NETWORK_LIST from the start, and does NOT use the address or the assoc filter
s
    ' This, by the way, should never happen unless someone wants all the records in the system
    '
    'MsgBox "This query uses (NO people), (NO address), and (NO assoc filter)..."
    If Me.ChkNonKin.Value = tTrue Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            Else
                tQueryNonkinFirstStr = tQuerySelectFirstNonkin + tQueryAssocDynastyFirstFromStr + _
                    tNonkinWhereFirstQueryStr
                tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocDynastyFromStr + tNonkinWhereQueryStr
                tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocDynastyLastFromStr + tNonkinWhereQu
eryStr
            End If
            tQueryNonkinFirstStr = tQuerySelectFirstNonkin + tQueryAssocFirstFromStr + _
                tNonkinWhereFirstQueryStr
            tQueryNonkinStr = gQuerySelectNonkin + tQueryAssocFromStr + tNonkinWhereQueryStr
            tQueryNonkinLastStr = gQuerySelectNonkin + tQueryAssocLastFromStr + tNonkinWhereQueryStr
        End If

        If ChkKin.Value = tTrue Then
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tQueryKinStr = gQuerySelectKin + tQueryKinDynastyFromStr + tKinWhereQueryStr
                tQueryKinFirstStr = tQueryKinStr + "0))"
                tQueryKinLastStr = gQuerySelectKin + tQueryKinDynastyLastFromStr + tKinWhereQueryStr
            Else
                tQueryKinStr = gQuerySelectKin + tQueryKinFromStr + tKinWhereQueryStr
                tQueryKinFirstStr = tQueryKinStr + "0))"
                tQueryKinLastStr = gQuerySelectKin + tQueryKinLastFromStr + tKinWhereQueryStr
            End If
        End If
    End If

```

```

        End If
    End If
End If

'
' the query for moving the records
'
tQueryAppendStr = "INSERT INTO ZZ_NETWORK_LIST ( c_personid, c_node_id, c_edge_id, c_edge_type, c_up_total, " +
+ "c_down_total, c_col_total, c_mar_total, c_distance, c_delete ) " +
+ "SELECT ZZ_NETWORK_LIST_TMP.c_personid, ZZ_NETWORK_LIST_TMP.c_node_id, ZZ_NETWORK_LIST_TMP.c_edge_id, " +
+ "ZZ_NETWORK_LIST_TMP.c_edge_type, ZZ_NETWORK_LIST_TMP.c_up_total, ZZ_NETWORK_LIST_TMP.c_down_total, " +
+ "ZZ_NETWORK_LIST_TMP.c_col_total, ZZ_NETWORK_LIST_TMP.c_mar_total, ZZ_NETWORK_LIST_TMP.c_distance, " +
+ "0 as c_delete FROM ZZ_NETWORK_LIST_TMP"

' since the person ID in the query results already is in ZZ_NETWORK_LIST, the question is
' whether the c_node_id is in the list: if YES, then set c_node_dist at the current level

tNodeDistQueryStr = "UPDATE ZZ_NETWORK_LIST INNER JOIN ZZ_SCRATCH_PEOPLE ON " +
+ "ZZ_NETWORK_LIST.c_node_id = ZZ_SCRATCH_PEOPLE.c_person_id " +
+ "SET ZZ_NETWORK_LIST.c_node_dist = [ZZ_SCRATCH_PEOPLE].[c_node_dist] " +
+ "WHERE ((ZZ_NETWORK_LIST.c_node_dist) Is Null)"

'
' for insurance, explicitly delete duplicate results
'
tPruneTmpQuery = "UPDATE ZZ_NETWORK_LIST INNER JOIN ZZ_NETWORK_LIST_TMP ON " +
+ "(ZZ_NETWORK_LIST.c_edge_type = ZZ_NETWORK_LIST_TMP.c_edge_type) AND " +
+ "(ZZ_NETWORK_LIST.c_edge_id = ZZ_NETWORK_LIST_TMP.c_edge_id) AND " +
+ "(ZZ_NETWORK_LIST.c_node_id = ZZ_NETWORK_LIST_TMP.c_node_id) AND " +
+ "(ZZ_NETWORK_LIST.c_personid = ZZ_NETWORK_LIST_TMP.c_personid) " +
+ "SET ZZ_NETWORK_LIST_TMP.c_delete = 1;"

tPruneTmpQueryDuplesStr = "UPDATE ZZ_NETWORK_LIST_TMP AS ZZ_NETWORK_LIST_TMP_1 INNER JOIN " +
+ "ZZ_NETWORK_LIST_TMP ON (ZZ_NETWORK_LIST_TMP_1.c_personid = ZZ_NETWORK_LIST_TMP.c_personid) " +
+ "AND (ZZ_NETWORK_LIST_TMP_1.c_node_id = ZZ_NETWORK_LIST_TMP.c_node_id) " +
+ "AND (ZZ_NETWORK_LIST_TMP_1.c_edge_id = ZZ_NETWORK_LIST_TMP.c_edge_id) " +
+ "AND (ZZ_NETWORK_LIST_TMP_1.c_edge_type = ZZ_NETWORK_LIST_TMP.c_edge_type) " +
+ "SET ZZ_NETWORK_LIST_TMP.c_delete = 1 " +
+ "WHERE ((([ZZ_NETWORK_LIST_TMP].[c_up_total]*1000+[ZZ_NETWORK_LIST_TMP].[c_down_total]*100+[ZZ_NETWORK_LIST_TMP].[c_col_total]*10+[ZZ_NETWORK_LIST_TMP].[c_mar_total])>" +
+ "[ZZ_NETWORK_LIST_TMP_1].[c_up_total]*1000+[ZZ_NETWORK_LIST_TMP_1].[c_down_total]*100+[ZZ_NETWORK_LIST_TMP_1].[c_col_total]*10+[ZZ_NETWORK_LIST_TMP_1].[c_mar_total])))"

tQueryPruneTmpAssocInverse1Str = "UPDATE (ZZ_NETWORK_LIST_TMP INNER JOIN ZZ_NETWORK_LIST ON " +
+ "(ZZ_NETWORK_LIST_TMP.c_edge_type = ZZ_NETWORK_LIST.c_edge_type) AND " +
+ "(ZZ_NETWORK_LIST_TMP.c_node_id = ZZ_NETWORK_LIST.c_personid) AND " +
+ "(ZZ_NETWORK_LIST_TMP.c_personid = ZZ_NETWORK_LIST.c_node_id)) " +
+ "INNER JOIN ASSOC_CODES ON (ZZ_NETWORK_LIST_TMP.c_edge_id = ASSOC_CODES.c_assoc_pair) AND " +
+ "(ZZ_NETWORK_LIST.c_edge_id = ASSOC_CODES.c_assoc_code) " +
+ "SET ZZ_NETWORK_LIST_TMP.c_delete = 1"

tQueryPruneTmpAssocInverse2Str = "UPDATE (ZZ_NETWORK_LIST_TMP INNER JOIN ZZ_NETWORK_LIST_TMP AS " +
+ "ZZ_NETWORK_LIST_TMP_1 ON (ZZ_NETWORK_LIST_TMP.c_personid = ZZ_NETWORK_LIST_TMP_1.c_node_id) AND " +
+ "(ZZ_NETWORK_LIST_TMP.c_node_id = ZZ_NETWORK_LIST_TMP_1.c_personid) AND " +
+ "(ZZ_NETWORK_LIST_TMP.c_edge_type = ZZ_NETWORK_LIST_TMP_1.c_edge_type)) " +
+ "INNER JOIN ASSOC_CODES ON (ZZ_NETWORK_LIST_TMP.c_edge_id = ASSOC_CODES.c_assoc_code) AND " +
+ "(ZZ_NETWORK_LIST_TMP_1.c_edge_id = ASSOC_CODES.c_assoc_pair) " +
+ "SET ZZ_NETWORK_LIST_TMP.c_delete = 1 " +
+ "WHERE (((ZZ_NETWORK_LIST_TMP.c_edge_type)='N') AND (ASSOC_CODES.c_assoc_role_type = 'M')) AND " +
+ "((ZZ_NETWORK_LIST_TMP.c_personid)>[ZZ_NETWORK_LIST_TMP_1].[c_personid]))"

tQueryPruneTmpAssocInverse3Str = "UPDATE (ZZ_NETWORK_LIST_TMP INNER JOIN ZZ_NETWORK_LIST_TMP AS " +
+ "ZZ_NETWORK_LIST_TMP_1 ON (ZZ_NETWORK_LIST_TMP.c_personid = ZZ_NETWORK_LIST_TMP_1.c_node_id) AND " +
+ "(ZZ_NETWORK_LIST_TMP.c_node_id = ZZ_NETWORK_LIST_TMP_1.c_personid) AND " +
+ "(ZZ_NETWORK_LIST_TMP.c_edge_type = ZZ_NETWORK_LIST_TMP_1.c_edge_type)) " +
+ "INNER JOIN ASSOC_CODES ON (ZZ_NETWORK_LIST_TMP.c_edge_id = ASSOC_CODES.c_assoc_code) AND " +
+ "(ZZ_NETWORK_LIST_TMP_1.c_edge_id = ASSOC_CODES.c_assoc_pair) " +
+ "SET ZZ_NETWORK_LIST_TMP.c_delete = 1 " +
+ "WHERE (((ZZ_NETWORK_LIST_TMP.c_edge_type)='N') AND " +
+ "(ASSOC_CODES.c_assoc_role_type = 'P')))"

tQueryPruneTmpKinInverse1Str = "UPDATE (KINSHIP_CODES INNER JOIN ZZ_NETWORK_LIST ON " +
+ "KINSHIP_CODES.c_kincode = ZZ_NETWORK_LIST.c_edge_id) INNER JOIN ZZ_NETWORK_LIST_TMP ON " +
+ "(ZZ_NETWORK_LIST_TMP.c_edge_id = KINSHIP_CODES.c_kin_pair1) AND " +
+ "(ZZ_NETWORK_LIST.c_edge_type = ZZ_NETWORK_LIST_TMP.c_edge_type) AND " +
+ "(ZZ_NETWORK_LIST.c_personid = ZZ_NETWORK_LIST_TMP.c_node_id) AND " +
+ "(ZZ_NETWORK_LIST.c_node_id = ZZ_NETWORK_LIST_TMP.c_personid) SET ZZ_NETWORK_LIST_TMP.c_delete = 1"

tQueryPruneTmpKinInverse2Str = "UPDATE (KINSHIP_CODES INNER JOIN ZZ_NETWORK_LIST ON " +

```

```
"KINSHIP_CODES.c_kincode = ZZ_NETWORK_LIST.c_edge_id) INNER JOIN ZZ_NETWORK_LIST_TMP ON " + _
"(ZZ_NETWORK_LIST_TMP.c_edge_id = KINSHIP_CODES.c_kin_pair2) AND " + _
"(ZZ_NETWORK_LIST.c_edge_type = ZZ_NETWORK_LIST_TMP.c_edge_type) AND " + _
"(ZZ_NETWORK_LIST.c_personid = ZZ_NETWORK_LIST_TMP.c_node_id) AND " + _
"(ZZ_NETWORK_LIST.c_node_id = ZZ_NETWORK_LIST_TMP.c_personid) SET ZZ_NETWORK_LIST_TMP.c_delete = 1"
```

```
tQueryPruneTmpKinInverse3Str = "UPDATE KINSHIP_CODES INNER JOIN (ZZ_NETWORK_LIST_TMP AS ZZ_NETWORK_LIST_TMP_1
INNER JOIN " + _
"ZZ_NETWORK_LIST_TMP ON (ZZ_NETWORK_LIST_TMP_1.c_node_id = ZZ_NETWORK_LIST_TMP.c_personid) AND " + _
"(ZZ_NETWORK_LIST_TMP_1.c_personid = ZZ_NETWORK_LIST_TMP.c_node_id)) ON KINSHIP_CODES.c_kincode = " + _
"ZZ_NETWORK_LIST_TMP.c_edge_id SET ZZ_NETWORK_LIST_TMP.c_delete = 1 " + _
"WHERE (((ZZ_NETWORK_LIST_TMP.c_edge_type)='K') AND " + _
"((ZZ_NETWORK_LIST_TMP.c_personid)>[ZZ_NETWORK_LIST_TMP_1].[c_personid]) AND " + _
"((ZZ_NETWORK_LIST_TMP_1.c_edge_id)=[KINSHIP_CODES].[c_kin_pair1] " + _
"OR (ZZ_NETWORK_LIST_TMP_1.c_edge_id)=[KINSHIP_CODES].[c_kin_pair2])) "
```

```
' the basic loop is to start with the first person in gRstPersonID
' and march through to the end. It will grow as the search continues
```

```
tDebug = 1
```

```
If tDebug = 1 Then
```

```
cmdSQL.CommandText = "DELETE * FROM ZZ_DEBUG"
```

```
cmdSQL.Execute tRecCountKin
```

```
End If
```

```
tLoop = 1
```

```
tRecCountKin = 1
```

```
tRecCountNonkin = 1
```

```
Do While tLoop <= gMaxNodeDist + 1 And (tRecCountKin + tRecCountNonkin) > 0
```

```
If Me.ChkNonKin.Value = tTrue Then
```

```
'MsgBox "About to run Non-kin query"
```

```
If tLoop = 1 Then
```

```
If gMaxNodeDist = 0 Then
```

```
'MsgBox tQueryNonkinFirstStr
```

```
cmdSQL.CommandText = tQueryNonkinLastStr + "0))"
```

```
Else
```

```
'MsgBox tQueryNonkinFirstStr
```

```
cmdSQL.CommandText = tQueryNonkinFirstStr
```

```
End If
```

```
ElseIf tLoop = gMaxNodeDist + 1 Then
```

```
'MsgBox tQueryNonkinLastStr
```

```
cmdSQL.CommandText = tQueryNonkinLastStr + Str(gMaxNodeDist) + ")))"
```

```
Else
```

```
'MsgBox tQueryNonkinStr
```

```
cmdSQL.CommandText = tQueryNonkinStr + Str(tLoop - 1) + ")))"
```

```
End If
```

```
' run the query
```

```
'MsgBox "gUseIndexYears = " + IIf(gUseIndexYears, "True", "False")
```

```
'MsgBox Right(cmdSQL.CommandText, 300)
```

```
cmdSQL.Execute tRecCount
```

```
' mark the internal dupes created by path dependencies
```

```
'MsgBox "Prune step 1"
```

```
cmdSQL.CommandText = tPruneTmpQueryDuplesStr
```

```
cmdSQL.Execute tRecCount
```

```
' remove duplicates
```

```
'MsgBox "Prune step 2"
```

```
cmdSQL.CommandText = tPruneTmpQuery
```

```
cmdSQL.Execute tRecCount
```

```
' remove the inverse dupes between ZZ_NETWORK_LIST and ZZ_NETWORK_LIST_TMP
```

```
'MsgBox "Prune step 3"
```

```
cmdSQL.CommandText = tQueryPruneTmpAssocInverse1Str
```

```
cmdSQL.Execute tRecCount
```

```
cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST_TMP where c_delete = 1"
```

```
cmdSQL.Execute tRecCount
```

```
' remove the inverse dupes internal to ZZ_NETWORK_LIST_TMP
```

```
'MsgBox "Prune step 4"
```

```
cmdSQL.CommandText = tQueryPruneTmpAssocInverse2Str
```

```

cmdSQL.Execute tRecCount

cmdSQL.CommandText = tQueryPruneTmpAssocInverse3Str
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST_TMP where c_delete = 1"
cmdSQL.Execute tRecCount
'
'   transfer the results
'
'MsgBox "Transferring results"
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecCountNonkin
'
'   clear the scratch table

cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST_TMP"
cmdSQL.Execute tRecDeleted
'
Else
    tRecCountNonkin = 0
End If

If ChkKin.Value = tTrue Then
'
'MsgBox "About to run Kin query"
If tLoop = 1 Then
    If gMaxNodeDist = 0 Then
        'MsgBox tQueryKinFirstStr
        cmdSQL.CommandText = tQueryKinLastStr + "0))"
    Else
        'MsgBox tQueryKinFirstStr
        cmdSQL.CommandText = tQueryKinFirstStr
    End If
ElseIf tLoop = gMaxNodeDist + 1 Then
    'MsgBox tQueryKinLastStr
    cmdSQL.CommandText = tQueryKinLastStr + Str(gMaxNodeDist) + ")))"
Else
    'MsgBox tQueryKinStr
    cmdSQL.CommandText = tQueryKinStr + Str(tLoop - 1) + ")))"
End If
'
'   run the query
'
cmdSQL.Execute tRecCount
'
'   mark the internal dupes created by path dependencies
'
cmdSQL.CommandText = tPruneTmpQueryDuplesStr
cmdSQL.Execute tRecCount
'
'   remove duplicates with ZZ_NETWORK_LIST
'
cmdSQL.CommandText = tPruneTmpQuery
cmdSQL.Execute tRecCount
'
'   remove inverses (two possible) between ZZ_NETWORK_LIST and ZZ_NETWORK_LIST_TMP
'
cmdSQL.CommandText = tQueryPruneTmpKinInverse1Str
cmdSQL.Execute tRecCount
'
cmdSQL.CommandText = tQueryPruneTmpKinInverse2Str
cmdSQL.Execute tRecCount
'
cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST_TMP where c_delete = 1"
cmdSQL.Execute tRecCount
'
'   remove inverses internal to ZZ_NETWORK_LIST_TMP
'
cmdSQL.CommandText = tQueryPruneTmpKinInverse3Str
cmdSQL.Execute tRecCount
'
cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST_TMP where c_delete = 1"
cmdSQL.Execute tRecCount
'
'   fix the node distance
'
cmdSQL.CommandText = tNodeDistQueryStr
cmdSQL.Execute tRecCount

```

```

'
'   transfer the results
'
'MsgBox "Transferring results"
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecCountKin
'
'   clear the scratch table

cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST_TMP"
cmdSQL.Execute tRecDeleted
'
Else
    tRecCountKin = 0
End If
'
'   if the first round of searching did not include people, add names using the IDs in the person ID field
'   Set the node distance to 1 so the next round of search (which may be unrestricted) will use them as we
11.
'
If tLoop = 1 And Not gUsePersonID Then
    cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_node_dist ) " + _
        "SELECT DISTINCT ZZ_NETWORK_LIST.c_personid, 1 AS c_node_dist " + _
        "FROM ZZ_NETWORK_LIST"
    cmdSQL.Execute tRecDeleted
    '
    '   to make the node distances match, update all c_distance values to 1 in ZZ_NETWORK_LIST
    '
    cmdSQL.CommandText = "UPDATE ZZ_NETWORK_LIST SET ZZ_NETWORK_LIST.c_distance = 1"
    cmdSQL.Execute tRecDeleted
End If
'
'   now identify the new people, assign them an incremented distance, and add them to ZZ_SCRATCH_PEOPLE
'
'MsgBox "Adding People step 1"
cmdSQL.CommandText = "INSERT INTO ZZ_NETWORK_LIST_TMP (c_personid) " + _
    "SELECT DISTINCT ZZ_NETWORK_LIST.c_node_id FROM ZZ_NETWORK_LIST " + _
    "WHERE (((ZZ_NETWORK_LIST.c_node_dist) Is Null))"
cmdSQL.Execute tRecDeleted
'
'   now mark for deletion those which already exist in ZZ_SCRATCH_PEOPLE
'
'MsgBox "Adding People step 2"
cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PEOPLE INNER JOIN ZZ_NETWORK_LIST_TMP ON " + _
    "ZZ_SCRATCH_PEOPLE.c_person_id = ZZ_NETWORK_LIST_TMP.c_personid " + _
    "SET ZZ_NETWORK_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDeleted
'
'   now copy the new records and zap ZZ_NETWORK_LIST_TMP
'
'MsgBox "Adding People step 3"
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_node_dist ) " + _
    "SELECT ZZ_NETWORK_LIST_TMP.c_personid, " + Trim(Str(tLoop)) + _
    " AS c_node_dist FROM ZZ_NETWORK_LIST_TMP"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from ZZ_NETWORK_LIST_TMP"
cmdSQL.Execute tRecDeleted
'
'   finally, update the node distances in ZZ_NETWORK_LIST
'
'MsgBox "Updating ZZ_NETWORK_LIST c_node_dist"
cmdSQL.CommandText = tNodeDistQueryStr
cmdSQL.Execute tRecDeleted

tLoop = tLoop + 1
Loop
'
'   finally, use the results to build the full records
'
'MsgBox "About to fill the network table with ASSOC..."
'Set tQuery = CurrentDb.QueryDefs("ZZ_NETWORK_LIST copy ASSOC Query")
'tQuery.Execute
cmdSQL.CommandText = tQueryCopyNonkinStr
cmdSQL.Execute tRecDeleted
'

```

```

'MsgBox "About to fill the network table with KIN..."
'Set tQuery = CurrentDb.QueryDefs("ZZ_NETWORK_LIST copy KIN Query")
'tQuery.Execute
cmdSQL.CommandText = tQueryCopyKinStr
cmdSQL.Execute tRecDeleted
'
' to keep life simple, I add the people, source, and address information last
'
tQueryStr = "UPDATE ZZ_SOCIAL_NETWORK INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SOCIAL_NETWORK.c_person_id = ZZZ_BIOG_MA
IN.c_personid " + _
"SET ZZ_SOCIAL_NETWORK.c_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SOCIAL_NETWORK.c_name_chn = [ZZZ_BIOG_MAIN].
[c_name_chn], " + _
"ZZ_SOCIAL_NETWORK.c_index_year = [ZZZ_BIOG_MAIN].[c_index_year], ZZ_SOCIAL_NETWORK.c_index_year_type
_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
"ZZ_SOCIAL_NETWORK.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
"ZZ_SOCIAL_NETWORK.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " + _
"ZZ_SOCIAL_NETWORK.c_dy = [ZZZ_BIOG_MAIN].[c_dy], ZZ_SOCIAL_NETWORK.c_dynasty = [ZZZ_BIOG_MAIN].[c_dy
nasty], " + _
"ZZ_SOCIAL_NETWORK.c_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn], ZZ_SOCIAL_NETWORK.c_female = [ZZZ
_BIOG_MAIN].[c_female], " + _
"ZZ_SOCIAL_NETWORK.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], ZZ_SOCIAL_NETWORK.c_addr_name = [ZZ
Z_BIOG_MAIN].[c_index_addr_name], " + _
"ZZ_SOCIAL_NETWORK.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], ZZ_SOCIAL_NETWORK.c_addr_type = [
ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
"ZZ_SOCIAL_NETWORK.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], ZZ_SOCIAL_NETWORK.c_addr_d
esc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
"ZZ_SOCIAL_NETWORK.x_coord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SOCIAL_NETWORK.y_coord = [ZZZ_BIOG_MAIN].[
y_coord]"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
tQueryStr = "UPDATE ZZ_SOCIAL_NETWORK INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SOCIAL_NETWORK.c_node_id = ZZZ_BIOG_MAIN
.c_personid " + _
"SET ZZ_SOCIAL_NETWORK.c_node_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SOCIAL_NETWORK.c_node_chn = [ZZZ_BIOG_M
AIN].[c_name_chn], " + _
"ZZ_SOCIAL_NETWORK.c_node_index_year = [ZZZ_BIOG_MAIN].[c_index_year], " + _
"ZZ_SOCIAL_NETWORK.c_node_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
"ZZ_SOCIAL_NETWORK.c_node_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
"ZZ_SOCIAL_NETWORK.c_node_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " + _
"ZZ_SOCIAL_NETWORK.c_node_dy = [ZZZ_BIOG_MAIN].[c_dy], ZZ_SOCIAL_NETWORK.c_node_dynasty = [ZZZ_BIOG_M
AIN].[c_dynasty], " + _
"ZZ_SOCIAL_NETWORK.c_node_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn], ZZ_SOCIAL_NETWORK.c_node_fem
ale = [ZZZ_BIOG_MAIN].[c_female], " + _
"ZZ_SOCIAL_NETWORK.c_node_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], ZZ_SOCIAL_NETWORK.c_node_addr_
name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
"ZZ_SOCIAL_NETWORK.c_node_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], ZZ_SOCIAL_NETWORK.c_node_addr
r_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
"ZZ_SOCIAL_NETWORK.c_node_addr_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
"ZZ_SOCIAL_NETWORK.c_node_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
"ZZ_SOCIAL_NETWORK.node_xcoord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SOCIAL_NETWORK.node_ycoord = [ZZZ_BIOG
_MAIN].[y_coord]"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
tQueryStr = "UPDATE ZZ_SOCIAL_NETWORK INNER JOIN ADDR_CODES ON ZZ_SOCIAL_NETWORK.c_link_addr_id = ADDR_CODES.
c_addr_id " + _
"SET ZZ_SOCIAL_NETWORK.c_link_addr_name = [ADDR_CODES].[c_name], " + _
"ZZ_SOCIAL_NETWORK.c_link_addr_chn = [ADDR_CODES].[c_name_chn], " + _
"ZZ_SOCIAL_NETWORK.c_link_xcoord = [ADDR_CODES].[x_coord], " + _
"ZZ_SOCIAL_NETWORK.c_link_ycoord = [ADDR_CODES].[y_coord]"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
tQueryStr = "UPDATE ZZ_SOCIAL_NETWORK INNER JOIN TEXT_CODES ON ZZ_SOCIAL_NETWORK.c_source = TEXT_CODES.c_text
id " + _
"SET ZZ_SOCIAL_NETWORK.c_source_text = [TEXT_CODES].[c_title], " + _
"ZZ_SOCIAL_NETWORK.c_source_txt_chn = [TEXT_CODES].[c_title_chn]"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
'MsgBox "About to fill the People table..."

tQueryStr = "UPDATE ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_BIOG_MAIN ON " + _
"ZZ_SCRATCH_PEOPLE.c_person_id = ZZZ_BIOG_MAIN.c_personid " + _
"SET ZZ_SCRATCH_PEOPLE.c_name = [ZZZ_BIOG_MAIN].[c_name], " + _
"ZZ_SCRATCH_PEOPLE.c_name_chn = [ZZZ_BIOG_MAIN].[c_name_chn], " + _
"ZZ_SCRATCH_PEOPLE.c_index_year = [ZZZ_BIOG_MAIN].[c_index_year], " + _

```

```

"ZZ_SCRATCH_PEOPLE.c_dy = [ZZZ_BIOG_MAIN].[c_dy], " + _
"ZZ_SCRATCH_PEOPLE.c_dynasty = [ZZZ_BIOG_MAIN].[c_dynasty], " + _
"ZZ_SCRATCH_PEOPLE.c_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn], " + _
"ZZ_SCRATCH_PEOPLE.c_female = [ZZZ_BIOG_MAIN].[c_female], " + _
"ZZ_SCRATCH_PEOPLE.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
"ZZ_SCRATCH_PEOPLE.c_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
"ZZ_SCRATCH_PEOPLE.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
"ZZ_SCRATCH_PEOPLE.c_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
"ZZ_SCRATCH_PEOPLE.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
"ZZ_SCRATCH_PEOPLE.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
"ZZ_SCRATCH_PEOPLE.x_coord = [ZZZ_BIOG_MAIN].[x_coord], " + _
"ZZ_SCRATCH_PEOPLE.y_coord = [ZZZ_BIOG_MAIN].[y_coord]"

```

```
cmdSQL.CommandText = tQueryStr
```

```
cmdSQL.Execute tRecCount
```

```
,
```

```
' get the XY count
```

```
,
```

```
'MsgBox "About to count XYs..."
```

```
,
```

```
cmdSQL.CommandText = "Delete * from tmpXY"
```

```
cmdSQL.Execute tRecDeleted
```

```
,
```

```

tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
"SELECT ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord, Count(ZZ_SCRATCH_PEOPLE.x_coord) " + _
"AS CountOfx_coord, Count(ZZ_SCRATCH_PEOPLE.y_coord) AS CountOfy_coord " + _
"FROM ZZ_SCRATCH_PEOPLE " + _
"GROUP BY ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord;"

```

```
,
```

```
cmdSQL.CommandText = tQueryStr
```

```
cmdSQL.Execute tRecDeleted
```

```
,
```

```

tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_PEOPLE ON (tmpXY.y_coord = " + _
"ZZ_SCRATCH_PEOPLE.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_PEOPLE.x_coord) SET " + _
"ZZ_SCRATCH_PEOPLE.xy_count = [tmpXY].[CountOfx_coord];"

```

```
cmdSQL.CommandText = tQueryStr
```

```
cmdSQL.Execute tRecDeleted
```

```
,
```

```
' the final step is to fill in the aggregated version of the social network
```

```
,
```

```
cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK_AGGREGATE"
```

```
cmdSQL.Execute tRecDeleted
```

```
,
```

```
' make the aggregated list
```

```
,
```

```

tQueryStr = "INSERT INTO ZZ_SOCIAL_NETWORK_AGGREGATE ( c_person_id, c_node_id, c_link_count, c_edge_dist, c_rec_count ) " + _
"SELECT ZZ_SOCIAL_NETWORK.c_person_id, ZZ_SOCIAL_NETWORK.c_node_id, " + _
"Sum(ZZ_SOCIAL_NETWORK.c_link_count) AS SumOfc_link_count, Min(ZZ_SOCIAL_NETWORK.c_edge_dist) AS MinOfc_edge_dist, " + _
"Count(ZZ_SOCIAL_NETWORK.c_edge_dist) as c_rec_count " + _
"FROM ZZ_SOCIAL_NETWORK " + _
"GROUP BY ZZ_SOCIAL_NETWORK.c_person_id, ZZ_SOCIAL_NETWORK.c_node_id"

```

```
cmdSQL.CommandText = tQueryStr
```

```
cmdSQL.Execute tRecDeleted
```

```
,
```

```
' now fill in the basic information
```

```
,
```

```

tQueryStr = "UPDATE ZZ_SOCIAL_NETWORK INNER JOIN ZZ_SOCIAL_NETWORK_AGGREGATE ON " + _
"(ZZ_SOCIAL_NETWORK.c_person_id = ZZ_SOCIAL_NETWORK_AGGREGATE.c_person_id) AND " + _
"(ZZ_SOCIAL_NETWORK.c_node_id = ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_id) SET "

```

```
tStrQuerySetPerson = "ZZ_SOCIAL_NETWORK_AGGREGATE.c_name = [ZZ_SOCIAL_NETWORK].[c_name], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_name_chn = [ZZ_SOCIAL_NETWORK].[c_name_chn], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_index_year = [ZZ_SOCIAL_NETWORK].[c_index_year], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_dy = [ZZ_SOCIAL_NETWORK].[c_dy], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_dynasty = [ZZ_SOCIAL_NETWORK].[c_dynasty], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_dynasty_chn = [ZZ_SOCIAL_NETWORK].[c_dynasty_chn], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_female = [ZZ_SOCIAL_NETWORK].[c_female], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_addr_id = [ZZ_SOCIAL_NETWORK].[c_addr_id], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_addr_name = [ZZ_SOCIAL_NETWORK].[c_addr_name], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_addr_chn = [ZZ_SOCIAL_NETWORK].[c_addr_chn], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_addr_type = [ZZ_SOCIAL_NETWORK].[c_addr_type], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_addr_desc = [ZZ_SOCIAL_NETWORK].[c_addr_desc], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_addr_desc_chn = [ZZ_SOCIAL_NETWORK].[c_addr_desc_chn], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.x_coord = [ZZ_SOCIAL_NETWORK].[x_coord], " + _
```

```
"ZZ_SOCIAL_NETWORK_AGGREGATE.y_coord = [ZZ_SOCIAL_NETWORK].[y_coord], "
```

```
tStrQuerySetNode = "ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_name = [ZZ_SOCIAL_NETWORK].[c_node_name], " + _
```

```

"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_chn = [ZZ_SOCIAL_NETWORK].[c_node_chn], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_index_year = [ZZ_SOCIAL_NETWORK].[c_node_index_year], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_dy = [ZZ_SOCIAL_NETWORK].[c_node_dy], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_dynasty = [ZZ_SOCIAL_NETWORK].[c_node_dynasty], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_dynasty_chn = [ZZ_SOCIAL_NETWORK].[c_node_dynasty_chn], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_addr_id = [ZZ_SOCIAL_NETWORK].[c_node_addr_id], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_addr_name = [ZZ_SOCIAL_NETWORK].[c_node_addr_name], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_addr_chn = [ZZ_SOCIAL_NETWORK].[c_node_addr_chn], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_addr_type = [ZZ_SOCIAL_NETWORK].[c_node_addr_type], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_addr_desc = [ZZ_SOCIAL_NETWORK].[c_node_addr_desc], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_addr_desc_chn = [ZZ_SOCIAL_NETWORK].[c_node_addr_desc_chn], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.node_xcoord = [ZZ_SOCIAL_NETWORK].[node_xcoord], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.node_ycoord = [ZZ_SOCIAL_NETWORK].[node_ycoord], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_distance = ZZ_SOCIAL_NETWORK.c_distance"

```

```

cmdSQL.CommandText = tQueryStr + tStrQuerySetPerson + tStrQuerySetNode
cmdSQL.Execute tRecDeleted

```

```

' Set tQuery = CurrentDb.QueryDefs("ZZ_SOCIAL_NETWORK_AGGREGATE Query")
tQuery.Execute

```

```

' get the descriptions

```

```

tQueryStr = "UPDATE (ZZ_SOCIAL_NETWORK INNER JOIN ZZ_SOCIAL_NETWORK_AGGREGATE ON " + _
"ZZ_SOCIAL_NETWORK.c_person_id = ZZ_SOCIAL_NETWORK_AGGREGATE.c_person_id AND " + _
"ZZ_SOCIAL_NETWORK.c_node_id = ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_id) " + _
"SET ZZ_SOCIAL_NETWORK_AGGREGATE.c_link_desc = " + _
"[ZZ_SOCIAL_NETWORK].[type_id]+'_'+[ZZ_SOCIAL_NETWORK].[c_link_desc], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_link_chn = " + _
"[ZZ_SOCIAL_NETWORK].[type_id]+'_'+[ZZ_SOCIAL_NETWORK].[c_link_chn] " + _
"WHERE (((ZZ_SOCIAL_NETWORK_AGGREGATE.c_rec_count)=1))"

```

```

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

```

```

tQueryStr = "UPDATE ZZ_SOCIAL_NETWORK_AGGREGATE SET ZZ_SOCIAL_NETWORK_AGGREGATE.c_link_desc = " + _
"'Multiple associations merged', ZZ_SOCIAL_NETWORK_AGGREGATE.c_link_chn = '" + _
ChrW(32317) + ChrW(21512) + ChrW(22810) + ChrW(31278) + ChrW(38364) + ChrW(20418) + "'" + _
"WHERE (((ZZ_SOCIAL_NETWORK_AGGREGATE.c_rec_count)>1))"

```

```

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

```

' the code is messy enough that I put the task of updating the new index year information into three queries at the very end

```

cmdSQL.CommandText = "UPDATE ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 INNER JOIN (ZZZ_BIOG_MAIN INNER JOIN ZZ_SOCIAL_NETWORK " + _
"ON ZZZ_BIOG_MAIN.c_personid = ZZ_SOCIAL_NETWORK.c_person_id) ON ZZZ_BIOG_MAIN_1.c_personid = ZZ_SOCIAL_NETWORK.c_node_id " + _
"SET ZZ_SOCIAL_NETWORK.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
"ZZ_SOCIAL_NETWORK.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
"ZZ_SOCIAL_NETWORK.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " + _
"ZZ_SOCIAL_NETWORK.c_node_index_year_type_code = [ZZZ_BIOG_MAIN_1].[c_index_year_type_code], " + _
"ZZ_SOCIAL_NETWORK.c_node_index_year_type_desc = [ZZZ_BIOG_MAIN_1].[c_index_year_type_desc], " + _
"ZZ_SOCIAL_NETWORK.c_node_index_year_type_hz = [ZZZ_BIOG_MAIN_1].[c_index_year_type_hz]"

```

```

cmdSQL.Execute tRecDeleted

```

```

cmdSQL.CommandText = "UPDATE ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 INNER JOIN (ZZZ_BIOG_MAIN INNER JOIN ZZ_SOCIAL_NETWORK_AGGREGATE " + _
"ON ZZZ_BIOG_MAIN.c_personid = ZZ_SOCIAL_NETWORK_AGGREGATE.c_person_id) ON ZZZ_BIOG_MAIN_1.c_personid = ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_id " + _
"SET ZZ_SOCIAL_NETWORK_AGGREGATE.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_index_year_type_desc = [ZZZ_BIOG_MAIN_1].[c_index_year_type_desc], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_index_year_type_hz = [ZZZ_BIOG_MAIN_1].[c_index_year_type_hz], " + _
"ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_index_year_type_code = [ZZZ_BIOG_MAIN_1].[c_index_year_type_code]"

```

```

cmdSQL.Execute tRecDeleted

```

```

cmdSQL.CommandText = "UPDATE ZZZ_BIOG_MAIN INNER JOIN ZZ_SCRATCH_PEOPLE ON ZZZ_BIOG_MAIN.c_personid = ZZ_SCRATCH_PEOPLE.c_person_id " + _
"SET ZZ_SCRATCH_PEOPLE.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
"ZZ_SCRATCH_PEOPLE.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
"ZZ_SCRATCH_PEOPLE.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz]"
cmdSQL.Execute tRecDeleted

```


Prepare_to_Exit_CmdRun_Click:

```

    Set gRstEdge = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
    Set ZZ_SOCIAL_NETWORK.Form.Recordset = gRstEdge
    Set gRstPersonID = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
    Set ZZ_SCRATCH_PEOPLE.Form.Recordset = gRstPersonID

    Set ZZ_SOCIAL_NETWORK_AGGREGATED.Form.Recordset = _
        CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK_AGGREGATE", dbOpenDynaset)

    If gRstPersonID.RecordCount > 0 Then
        Me.CmdGIS.Enabled = True
        Me.CmdGUESS.Enabled = True
        Me.CmdUCINet.Enabled = True
        Me.CmdPajek.Enabled = True
        Me.CmdNeo4j.Enabled = True
        Me.ChkIncludeID.Enabled = True
        'Me.CmdRerun.Enabled = True
        CmdStoreID.Enabled = True
    Else
        Me.CmdGIS.Enabled = False
        Me.CmdGUESS.Enabled = False
        Me.CmdUCINet.Enabled = False
        Me.CmdPajek.Enabled = False
        Me.CmdNeo4j.Enabled = False
        Me.ChkIncludeID.Enabled = False
        'Me.CmdRerun.Enabled = False
        CmdStoreID.Enabled = False
    End If

```

Exit_CmdRun_Click:

```

    ' close the tables
    Set gRstAssocFilter = Nothing
    Set tRstDummy = Nothing
    'Forms("ZZZ_DONT_PANIC").Visible = False

Exit Sub

```

Err_CmdRun_Click:

```

    MsgBox Err.Description
    Resume Exit_CmdRun_Click

```

End Sub

Private Sub update_info(tLoops As Integer)

```

    'Forms("ZZZ_DONT_PANIC").Modal = True
    'tLoopInfoStr = "Loops = " & Str(tLoops) & " Nodes = " & Str(gRstPersonID.RecordCount) _
        & " Edges = " & Str(gRstEdge.RecordCount)
    'Forms("ZZZ_DONT_PANIC")!LblCounter.Caption = tLoopInfoStr
    'Forms("ZZZ_DONT_PANIC").Repaint
    'Forms("ZZZ_DONT_PANIC").Modal = False

```

End Sub

Private Sub calculate_xy_count()

```

    Dim tX As Double, tY As Double, tXY As Integer, tBM As Variant, tWrite As Integer
    ' the strategy is to first throw a bookmark at the first new value
    ' then count the number, then go back to the bookmark and update each record

    With gRstPersonID
        .Index = "xy"
        .MoveFirst

        tX = -1#
        tY = -1#
        tXY = 0
        tWrite = 0
        tBM = .Bookmark

        Do While Not .EOF
            If tX <> !x_coord Or tY <> !y_coord Then
                If tWrite = 1 Then
                    ' go back to the first record with the value
                    .Bookmark = tBM
                    Do While tX = !x_coord And tY = !y_coord
                        .Edit
                        !xy_count = tXY
                        .Update
                    
```

```

        .MoveNext
    Loop
Else
    tWrite = 1
End If
' reset
tXY = 0
tBM = .Bookmark
tX = !x_coord
tY = !y_coord
End If
' increment the count and move to the next
tXY = tXY + 1
.MoveNext
Loop
'
' the last xy value still needs to be written
'
.Bookmark = tBM
Do While Not .EOF
    .Edit
    !xy_count = tXY
    .Update
    .MoveNext
Loop
.Index = "index_year"
End With
End Sub
Private Sub makeAssocFilter()
'
' this is a brute-force routine that looks at the filer request and builds the filter
' table (remember true = -1, false = 0)
'
Dim cmdSQL As ADODB.Command, tRecCount As Long, strBaseSQL As String

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' start with the single-unit categories
'
strBaseSQL = "INSERT INTO ZZ_SCRATCH_ASSOC_FILTER ( c_assoc_code ) SELECT ASSOC_CODE_TYPE_REL.c_assoc_code FR
OM ASSOC_CODE_TYPE_REL WHERE "
'
If ChkFamily.Value = -1 Then
    cmdSQL.CommandText = strBaseSQL + "((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code],2)='09'))"
    cmdSQL.Execute tRecCount
End If
'
If ChkFinance.Value = -1 Then
    cmdSQL.CommandText = strBaseSQL + "((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code],2)='10'))"
    cmdSQL.Execute tRecCount
End If
'
If ChkFriendship.Value = -1 Then
    cmdSQL.CommandText = strBaseSQL + "((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code],2)='03'))"
    cmdSQL.Execute tRecCount
End If
'
If ChkMedicine.Value = -1 Then
    cmdSQL.CommandText = strBaseSQL + "((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code],2)='07'))"
    cmdSQL.Execute tRecCount
End If
'
If ChkReligion.Value = -1 Then
    cmdSQL.CommandText = strBaseSQL + "((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code],2)='08'))"
    cmdSQL.Execute tRecCount
End If
'
' now for the larger categories
If gFilterMilitaryCount = gMaxFilterMilitary Then
    cmdSQL.CommandText = strBaseSQL + "((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code],2)='06'))"
    cmdSQL.Execute tRecCount
ElseIf gFilterMilitaryCount > 0 Then
    ' look for each component
    If ChkMilitarySupport.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0602')"
        cmdSQL.Execute tRecCount
    End If
    '

```

```

        If ChkMilitaryOppose.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0603')"
            cmdSQL.Execute tRecCount
        End If
    End If
    '
    If gFilterScholarCount = gMaxFilterScholar Then
        cmdSQL.CommandText = strBaseSQL + "((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code],2)='02'))"
        cmdSQL.Execute tRecCount
    ElseIf gFilterScholarCount > 0 Then
        ' look for each component
        If ChkSchTeacher.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0202')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkSchAffiliation.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0203')"
            cmdSQL.Execute tRecCount
        End If
        '
        If Me.ChkSchTopic.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0204')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkSchMember.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0205')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkSchPatron.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0206')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkSchLitArt.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0207')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkSchAttack.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0208')"
            cmdSQL.Execute tRecCount
        End If
    End If
    '
    If gFilterPoliticsCount = gMaxFilterPolitics Then
        cmdSQL.CommandText = strBaseSQL + "((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code],2)='04'))"
        cmdSQL.Execute tRecCount
    ElseIf gFilterPoliticsCount > 0 Then
        ' look for each component
        If ChkPolEqual.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0402')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkPolSub.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0403')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkPolSup.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0404')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkPolSupport.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0405')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkPolSponsor.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0406')"
            cmdSQL.Execute tRecCount
        End If
        '
        If ChkPolOppose.Value = -1 Then
            cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0407')"
            cmdSQL.Execute tRecCount
        End If
    End If
    '

```

```

68
End If
'
End If
'
If gFilterWritingsCount = gMaxFilterWritings Then
    cmdSQL.CommandText = strBaseSQL + "((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code],2)='05'))"
    cmdSQL.Execute tRecCount
ElseIf gFilterWritingsCount > 0 Then
    ' look for each component
    If ChkWriCommem.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0502')"
        cmdSQL.Execute tRecCount
    End If
    '
    If ChkWriEpitaph.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0503')"
        cmdSQL.Execute tRecCount
    End If
    '
    If ChkWriPreface.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0504')"
        cmdSQL.Execute tRecCount
    End If
    '
    If ChkWriRitual.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0505')"
        cmdSQL.Execute tRecCount
    End If
    '
    If ChkWriBiog.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0506')"
        cmdSQL.Execute tRecCount
    End If
    '
    If ChkWriExplain.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0507')"
        cmdSQL.Execute tRecCount
    End If
    '
    If ChkWriMottos.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0508')"
        cmdSQL.Execute tRecCount
    End If
    '
    If ChkWriLetters.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0509')"
        cmdSQL.Execute tRecCount
    End If
    '
    If ChkWriOccasion.Value = -1 Then
        cmdSQL.CommandText = strBaseSQL + "([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]='0510')"
        cmdSQL.Execute tRecCount
    End If
End If
'
End Sub
Private Sub fill_first_record()
    Dim tAddrID As Long, strSeek As String
    '
    ' fill in the data in the node file for the first record
    '
    gRst.MoveFirst
    gRstPersonID.MoveFirst
    gRstPersonID.Edit
    gRstPersonID!c_name = gRst!c_person_name
    gRstPersonID!c_name_chn = gRst!c_person_name_chn
    gRstPersonID!c_index_year = gRst!c_index_year
    gRstPersonID!c_female = gRst!c_female
    gRstPersonID!c_node_dist = 0
    '
    If IsNull(gRst!c_addr_id) Then
        gRstPersonID!x_coord = 0#
        gRstPersonID!y_coord = 0#
        gRstPersonID!c_addr_id = 0
        gRstPersonID!c_addr_name = ""
        gRstPersonID!c_addr_chn = ""
        gRstPersonID!c_addr_type = 0
        gRstPersonID!c_addr_desc = ""
        gRstPersonID!c_addr_desc_chn = ""
    Else

```

```

gRstPersonID!x_coord = gRst!x_coord
gRstPersonID!y_coord = gRst!y_coord
gRstPersonID!c_addr_id = gRst!c_addr_id
gRstPersonID!c_addr_name = gRst!c_addr_name
gRstPersonID!c_addr_chn = gRst!c_addr_chn
gRstPersonID!c_addr_type = 1
gRstPersonID!c_addr_desc = "Basic"
gRstPersonID!c_addr_desc_chn = ChrW(&H7C4D) & ChrW(&H8CAB)
End If
'
gRstPersonID.Update

End Sub
Private Sub CmdSelectPerson_Click()
On Error GoTo Err_CmdSelectPerson_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strPERSON_ID As String, cmdSQL As ADODB.Command

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

TxtPersonID.Visible = True
TxtPersonID.SetFocus
strPERSON_ID = TxtPersonID.Text

stDocName = "frmSelectPerson"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strPERSON_ID

If CurrentProject.AllForms("frmSelectPerson").IsLoaded Then
Dim lngPERSON_ID As Long
Dim strPERSON_NM As String
Dim strPERSON_NM_CHN As String

Forms!frmSelectPerson!frmPersonSearch.Form!c_personid.SetFocus
lngPERSON_ID = Forms!frmSelectPerson!frmPersonSearch.Form!c_personid.Value
TxtPersonID.Value = lngPERSON_ID

Forms!frmSelectPerson!frmPersonSearch.Form!c_name_chn.SetFocus
strPERSON_NM_CHN = Forms!frmSelectPerson!frmPersonSearch.Form!c_name_chn.Value
TxtNameChn.Value = strPERSON_NM_CHN

Forms!frmSelectPerson!frmPersonSearch.Form!c_name.SetFocus
strPERSON_NM = Forms!frmSelectPerson!frmPersonSearch.Form!c_name.Value
TxtName.Value = strPERSON_NM

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted

'
' add the name
'
tStrQuery = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id) SELECT " + Str(lngPERSON_ID) + " as
c_person_id"

cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecDeleted

gUsePersonID = True
Me.CmdAllPeople.Enabled = True
CmdRun.Enabled = True
'CmdRerun.Enabled = False

DoCmd.Close acForm, stDocName
End If

CmdSelectPerson.SetFocus
TxtPersonID.Visible = False
Call CheckRunCriteria

Exit_CmdSelectPerson_Click:
Exit Sub

Err_CmdSelectPerson_Click:
MsgBox Err.Description
Resume Exit_CmdSelectPerson_Click

```

End Sub

```
Private Sub CmdSelectPlace_Click()
On Error GoTo Err_CmdSelectPlace_Click
```

```
Dim stDocName As String
Dim stLinkCriteria As String
Dim strADDR As String
Dim cmdSQL As ADODB.Command
```

```
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
```

```
TxtAddrID.Visible = True
TxtAddrID.SetFocus
strADDR = TxtAddrID.Text
```

```
stDocName = "frmPickAddresses_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR
```

```
If CurrentProject.AllForms("frmPickAddresses_multi").IsLoaded Then
Dim tAddrID As Long, tRstAddr As DAO.Recordset
Dim strADDR_CHN As String, strADDR_PY As String
```

```
CmdAllPlaces.Enabled = True
ChkPlaceLimit.Enabled = True
ChkXYRef.Enabled = True
ChkSubUnits.Enabled = True
```

```
gUseADDRID = True
```

```
'Set tRstAddresses = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
'tRstAddresses.MoveFirst
'If tRstAddresses.RecordCount = 0 Then
Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Visible = True
Forms!frmPickAddresses_multi.Form!TxtAddrFilter.SetFocus
If Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Value Then
```

```
    '
    TxtAddrID.Value = 0
    strADDR_PY = Forms!frmPickAddresses_multi.Form!TxtFilterPY
    strADDR_CHN = Forms!frmPickAddresses_multi.Form!TxtFilterChn
```

```
    If strADDR_CHN = "" Then
        TxtPlaceChn.Value = "[[Filter]]"
        TxtPlace.Value = "[" + strADDR_PY + "]"
    Else
        TxtPlaceChn.Value = "[" + strADDR_CHN + "]"
        TxtPlace.Value = "[[Filter]]"
    End If
```

```
Else
Forms!frmPickAddresses_multi.Form!TxtSelectCount.Visible = True
Forms!frmPickAddresses_multi.Form!TxtSelectCount.SetFocus
If Forms!frmPickAddresses_multi.Form!TxtSelectCount.Value > 1 Then
    TxtPlaceChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
    TxtPlace.Value = "[[Multi-Select]]"
    TxtAddrID.Value = 0
```

```
Else
    ' only one record in ZZ_ADDRESSES: get its field values
    '
```

```
    Set tRstAddr = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
    tRstAddr.MoveFirst
    'MsgBox "Checking zz_addresses: no records"
    TxtAddrID.Value = tRstAddr!c_addr_id
    TxtPlaceChn.Value = tRstAddr!c_name_chn
    TxtPlace.Value = tRstAddr!c_name
    tRstAddr.Close
    Set tRstAddr = Nothing
```

```
End If
```

```
End If
```

```
' now copy the records
'
```

```
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_LIST"
cmdSQL.Execute tRecDeleted
```

```
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT " + _
    "ZZ_ADDRESSES.c_addr_id FROM ZZ_ADDRESSES"
cmdSQL.Execute tRecDeleted
```

```

        CmdRun.Enabled = True
        DoCmd.Close acForm, stDocName
    End If

    CmdSelectPlace.SetFocus
    TxtAddrID.Visible = False
    Call CheckRunCriteria

Exit_CmdSelectPlace_Click:
    Exit Sub

Err_CmdSelectPlace_Click:
    MsgBox Err.Description
    Resume Exit_CmdSelectPlace_Click

End Sub

Private Sub CmdToDynasty_Click()
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strToDynasty As String

    If gToDynasty = -1 Then
        strToDynasty = ""
    Else
        strToDynasty = Str(gToDynasty)
    End If

    stDocName = "frmPickDynasty"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

    If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

        Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
        gToDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
        gToDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
        gToDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
        '
        ' check to see if we have a problem and reject selection if needed
        '
        If gFromDynasty > -1 Then
            If gFromDynastyBegin > gToDynastyEnd Then
                MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' D
ynasty ends!", vbExclamation
                gToDynasty = -1
                TxtToDynasty.Value = ""
                TxtToDynastyPY.Value = ""
            End If
        End If
        '
        ' value is OK
        '
        If gToDynasty > -1 Then
            Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
            TxtToDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

            Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
            TxtToDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
        End If

        DoCmd.Close acForm, stDocName
        '
        ' reset FromDynasty if necessary (-2 = all dynasties)
        '
        If gFromDynasty = -2 Then
            gFromDynasty = -1
            TxtFromDynasty.Value = ""
            TxtFromDynastyPY.Value = ""
        End If
    End If

End Sub

Private Sub CmdUCINet_Click()

```

```

72
On Error GoTo Err_CmdUCINet_Click
'
' This program will dump the results of the search to a .vna file
'
' for the moment I'll just describe the format of the .vna file
'
' *node data
' ID index_year sex x_coord y_coord nodedist
' ID = str(c_person_id)
' indexyear = c_index_year INT
' nodedist = c_node_dist INT
' sex = c_female > (F,M)
' *node properties
' ID color shape size shortlabel active
' color = red (1), orange (2), yellow (3), green (4), blue (5)
' shortlabel = c_name
' shape = 2
' active = TRUE
'
' *tie data
' from to edgetype nodedist
' from = str(c_person_id)
' to = str(c_node_id)
' edgetype= c_link_type (K,N)
'
' *tie properties
' from to color size active
' from = str(c_person_id)
' to = str(c_node_id)
' color = red (255), orange (26367), yellow (65535), green (32768), blue (16711680)
' size = 1-5 (the weight)
'
' the central question is whether to do distance optimizations
'
' first see if there are any records to process
'
If ZZ_SOCIAL_NETWORK.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUCINet_Click
End If
'
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUCINet_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstAssocType As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tSearchStr As String
Dim tColor(20) As String, tQuote As String
Dim tFileSystem, tVNA

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

' open the assoc type look-up table
Set tRstAssocType = CurrentDb.OpenRecordset("ASSOC_CODE_TYPE_REL", dbOpenDynaset)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network.vna"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdUCINet_Click
        Else
            ' make sure the file name has a vna extension
            If Len(tFileName) < 5 Then

```



```

        tFileName = tFileName + ".vna"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".vna") Then
        tFileName = tFileName + ".vna"
    End If
End If
'
' now process the file (second true removed to make ASCII)
'
Set tFileSystem = CreateObject("Scripting.FileSystemObject")
Set tVNA = tFileSystem.CreateTextFile(tFileName, True)

' define the colors for the nodes
'
tColor(1) = "0 "      ' black
tColor(2) = "16711680 " ' blue
tColor(3) = "32768 "  ' green
tColor(4) = "65535 "  ' yellow
tColor(5) = "26367 "  ' orange
For ti = 6 To 20
    tColor(ti) = "255 " ' red
Next
'
' process the two tables
'
Set tRstEdge = ZZ_SOCIAL_NETWORK.Form.Recordset
Set tRstNode = ZZ_SCRATCH_PEOPLE.Form.Recordset
tQuote = Chr(34) ' the quotation mark
'
' first the nodes: define the node data structure
tVNA.WriteLine ("*node data")
tVNA.WriteLine ("ID index_year dynasty_code dynasty sex x_coord y_coord nodedist")
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' name = the ID of the person
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' indexyear = c_index_year INT
        If IsNull(!c_index_year) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + " "
        End If
        '
        ' dynasty_code = c_dy INT
        If IsNull(!c_dynasty) Then
            tStr = tStr + "0 " + tQuote + "Unknown" + tQuote + " "
        Else
            tStr = tStr + Trim(Str(!c_dy)) + " " + tQuote + !c_dynasty + tQuote + " "
        End If
        '
        ' sex = c_female > (F,M)
        If !c_female = -1 Then
            tStr = tStr + tQuote + "F" + tQuote + " "
        Else
            tStr = tStr + tQuote + "M" + tQuote + " "
        End If
        '
        ' x_coord
        If IsNull(!x_coord) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!x_coord)) + " "
        End If
        '
        ' y_coord
        If IsNull(!y_coord) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!y_coord)) + " "
        End If
        '
        ' node distance
        tStr = tStr + Trim(Str(!c_node_dist))
        '
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With

```

```

'
' now the node properties
'
' Note: ACTIVE was removed as a property (MAF 201807/22)
'
tVNA.WriteLine ("*node properties")
tVNA.WriteLine ("ID color shape size shortlabel")
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' ID = the ID of the person
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' color = black (1), blue (2), green (3), yellow (4), orange (5)
        tStr = tStr + tColor(!c_node_dist + 1)
        '
        ' shape = 2? / size = 1?
        tStr = tStr + "2 1 "
        '
        ' shortlabel (+ Active = TRUE removed)
        If IsNull(!c_name) Then
            tStr = tStr + "[Missing]"
        Else
            tStr = tStr + tQuote + !c_name + tQuote
        End If
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the edges: define the record structure
'
tStr = "from to " + tQuote + "EdgeWeight" + tQuote + " " + tQuote + "edgetype"
tStr = tStr + tQuote + " " + tQuote + "edgedist" + tQuote
tVNA.WriteLine ("*tie data")
tVNA.WriteLine (tStr)
'
' For the moment, I am not combining parallel edges
'
With tRstEdge
    .MoveFirst
    Do While Not .EOF
        '
        ' From = str(c_person_id) for node1
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' to = str(c_node_id) for node2
        tStr = tStr + Trim(Str(!c_node_id)) + " 1 "
        '
        ' edgetype
        If !c_link_type = "K" Then
            If IsNull(!c_link_desc) Then
                tStr = tStr + "K "
            Else
                tStr = tStr + tQuote + "K_" + !c_link_desc + tQuote + " "
            End If
        Else
            tSearchStr = "c_assoc_code = " + Trim(Str(!c_link_code))
            tRstAssocType.FindFirst tSearchStr
            If tRstAssocType.NoMatch Then
                tStr = tStr + "N_00 "
            Else
                tStr = tStr + "N_" + Trim(tRstAssocType!c_assoc_type_code) + " "
            End If
        End If
        '
        ' edgedist
        tStr = tStr + Trim(Str(!c_edge_dist))
        '
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the edges properties
'
'tVNA.WriteLine ("*tie properties")
'tVNA.WriteLine ("from to color size active")

```

```

        'With tRstEdge
        '    .MoveFirst
        '    Do While Not .EOF
        '        '
        '        '        from = str(c_person_id) for node1
        '        'tStr = Trim(Str(!c_person_id)) + " "
        '        '
        '        '        to = str(c_node_id) for node2
        '        'tStr = tStr + Trim(Str(!c_node_id)) + " 1 "
        '        '
        '        '        color = black (1), blue (2), green (3), yellow (4), orange (5)
        '        'tStr = tStr + tColor(!c_edge_dist)
        '        '
        '        '        size = 1? active = TRUE
        '        'tStr = tStr + "1 TRUE"
        '        '
        '        'tVNA.WriteLine (tStr)
        '        '    .MoveNext
        '    Loop
        'End With
        '
        tVNA.Close
        '
        Set tRstNode = Nothing
        Set tRstEdge = Nothing
        Set tVNA = Nothing
        Set tFileSystem = Nothing
        Set tRstAssocType = Nothing
    Else
        'The user pressed Cancel.
    End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

```

```

Exit_CmdUCINet_Click:
Exit Sub

```

```

Err_CmdUCINet_Click:
MsgBox Err.Description
Resume Exit_CmdUCINet_Click

```

```

End Sub

```

```

Private Sub CmdPajek_Click()
On Error GoTo Err_CmdUTF8Pajek_Click
'
'    This program will dump the results of the search to a .net file
'
'    for the moment I'll just describe the format of the .gdf file
'
'    *Vertices NUM
'    ID label "box" ic [color] bc [color]
'        ID = str(c_person_id)
'        label = c_name_chn
'        color = red (1), orange (2), yellow (3), green (4), blue (5)
'
'    *Edges
'    node1 node2 1 1 "label"
'        node1 = str(c_person_id) for node1
'        node2 = str(c_node_id) for node2
'        color = red (1), orange (2), yellow (3), green (4), blue (5)
'        label = c_link_desc
'
'    first see if there are any records to process
'
If ZZ_SOCIAL_NETWORK.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUTF8Pajek_Click
End If
'
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUTF8Pajek_Click
End If
'
'    next get a file

```

```

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
Dim tPinyin As Boolean
Dim tRstNode As DAO.Recordset, tRstNodeList As DAO.Recordset
Dim tRstEdge As DAO.Recordset, tRstAssocType As DAO.Recordset
Dim tRstAssocCodeType As DAO.Recordset, tRstEdgeList As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tQuote As String, tFindStr As String
Dim tColor(20) As String, tStrNode1 As String, tStrNode2 As String, tCodeStr As String, tQueryStr As String

tPinyin = False
' to write to a UTF-8 file, use the ADO stream object
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

If CodeFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8.net"
ElseIf CodeFrame.Value = 2 Then
    tStream.Charset = "big5"
    tCodeStr = "BIG5.net"
ElseIf CodeFrame.Value = 3 Then
    tStream.Charset = "gb2312"
    tCodeStr = "GB2312.net"
Else
    tStream.Charset = "iso-8859-1"
    tCodeStr = ".net"
    tPinyin = True
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network_" + tCodeStr
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdUTF8Pajek_Click
        Else
            ' make sure the file name has a net extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".net"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".net") Then
                tFileName = tFileName + ".net"
            End If
        End If
        '
        ' zap and open the scratch file
        '
        Dim cmdSQL As ADODB.Command
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
        cmdSQL.Execute tRecDeleted
        '
        ' fill the node list
        If tPinyin Then
            tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
                "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id, ZZ_SCRATCH_PEOPLE.c_name, " + _
                "ZZ_SCRATCH_PEOPLE.c_node_dist, val(c_person_id) AS c_v_num, TRUE as c_delete FROM ZZ_SCRATCH"
            _PEOPLE"
        Else
            tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
                "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id, ZZ_SCRATCH_PEOPLE.c_name_chn, " + _

```

```

        "ZZ_SCRATCH_PEOPLE.c_node_dist, val(c_person_id) AS c_v_num, TRUE as c_delete FROM ZZ_SCRATCH
PEOPLE"
End If

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' fill in any missing names
'
If Not tPinyin Then
    tQueryStr = "UPDATE ZZ_SCRATCH_PEOPLE INNER JOIN ZZ_SCRATCH_PAJEK ON " + _
        "ZZ_SCRATCH_PEOPLE.c_person_id = ZZ_SCRATCH_PAJEK.c_ID SET ZZ_SCRATCH_PAJEK.c_lbl = " + _
        "[ZZ_SCRATCH_PEOPLE].[c_name] WHERE (((ZZ_SCRATCH_PAJEK.c_lbl) Is Null))"

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
End If
'
' if needed, find the 0-degree nodes, using the edge list to mark the node list
'
If ChkDegree.Value Then
    cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PAJEK INNER JOIN ZZ_SOCIAL_NETWORK_AGGREGATE " + _
        "ON ZZ_SCRATCH_PAJEK.c_id = ZZ_SOCIAL_NETWORK_AGGREGATE.c_person_id " + _
        "SET ZZ_SCRATCH_PAJEK.c_delete = False"
    cmdSQL.Execute tRecDeleted
    cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PAJEK INNER JOIN ZZ_SOCIAL_NETWORK_AGGREGATE " + _
        "ON ZZ_SCRATCH_PAJEK.c_id = ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_id " + _
        "SET ZZ_SCRATCH_PAJEK.c_delete = False"
    cmdSQL.Execute tRecDeleted
    '
    ' remove records where c_delete = TRUE
    '
    'MsgBox "Got through update"
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK WHERE ((ZZ_SCRATCH_PAJEK.c_delete) = TRUE )"
    cmdSQL.Execute tRecDeleted
End If

Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
tRstNodeList.Index = "c_ID"
'
' there probably is an SQL way to do this, but...
'
ti = 1
With tRstNodeList
    .MoveFirst
    Do While Not .EOF
        .Edit
        !c_v_num = Trim(Str(ti))
        .Update
        ti = ti + 1
        .MoveNext
    Loop
End With
tRstNodeList.Close
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK_EDGE"
cmdSQL.Execute tRecDeleted
'
' fill the edge list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK_EDGE ( c_node_1, c_node_2, c_edge_count, c_edge_dist, c_edg
e_desc )" + _
    "SELECT Val([ZZ_SCRATCH_PAJEK].[c_v_num]) AS c_node_1, Val([ZZ_SCRATCH_PAJEK_1].[c_v_num]) " + _
    "AS c_node_2, ZZ_SOCIAL_NETWORK_AGGREGATE.c_link_count, ZZ_SOCIAL_NETWORK_AGGREGATE.c_edge_dist,
" + _
    "ZZ_SOCIAL_NETWORK_AGGREGATE.c_link_desc " + _
    "FROM ZZ_SCRATCH_PAJEK INNER JOIN (ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 INNER JOIN " + _
    "ZZ_SOCIAL_NETWORK_AGGREGATE ON ZZ_SCRATCH_PAJEK_1.c_ID = ZZ_SOCIAL_NETWORK_AGGREGATE.c_node_id)
" + _
    "ON ZZ_SCRATCH_PAJEK.c_ID = ZZ_SOCIAL_NETWORK_AGGREGATE.c_person_id"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'
Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenDynaset)
Set tRstEdgeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK_EDGE", dbOpenDynaset)

```

```

'
' set the Quote delimiter
'
tQuote = Chr(34)
'
' define the colors for the nodes
'
tColor(1) = "Black"
tColor(2) = "Blue"
tColor(3) = "Green"
tColor(4) = "Yellow"
tColor(5) = "Orange"
For ti = 6 To 20
    tColor(ti) = "Red"
Next
'
tC = Chr(44) ' the comma
'
' first the nodes:  define the record structure
'
tRstNodeList.MoveLast
tStr = "*Vertices " + Trim(Str(tRstNodeList.RecordCount))
tStream.WriteText tStr, adWriteLine
'
ti = 1

With tRstNodeList
    .MoveFirst
    Do While Not .EOF
        tStream.WriteText !c_v_num + " "
        '
        If IsNull(!c_lbl) Then
            tStream.WriteText Chr(34)
            tStream.WriteText "Error-" + Trim(Str(!c_ID))
            tStream.WriteText Chr(34)
            tStream.WriteText " box "
        Else
            If !c_lbl = "" Then
                tStream.WriteText Chr(34)
                tStream.WriteText "Error-" + Trim(Str(!c_ID))
                tStream.WriteText Chr(34)
                tStream.WriteText " box "
            Else
                tStream.WriteText Chr(34)
                tStream.WriteText !c_lbl
                If ChkIncludeID.Value Then
                    tStream.WriteText ":" + Trim(Str(!c_ID))
                End If
                tStream.WriteText Chr(34)
                tStream.WriteText " box "
            End If
        End If
        ' label
        tStr = " ic " + tColor(!c_distance + 1)
        tStr = tStr + " bc " + tColor(!c_distance + 1)
        ' color = white (1), blue (2), green (3), yellow (4), orange (5)
        tStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With

'
' now the edges:  define the record structure
'
tStream.WriteText "*Edges", adWriteLine

If tRstEdgeList.RecordCount > 0 Then
    With tRstEdgeList
        .MoveFirst
        Do While Not .EOF
            tStr = Trim(Str(!c_node_1)) + " " + Trim(Str(!c_node_2))
            '
            ' now get the weight
            '
            If !c_edge_count < 6 Then
                tStr = tStr + " " + Trim(Str(!c_edge_count)) + " "
            Else
                tStr = tStr + " 5 "
            End If
        End If
    End With
End If

```

```

        '
        ' now get the label
        '
        tStr = tStr + "l " + tQuote
        If !c_edge_count = 1 Then
            tStr = tStr + !c_edge_desc + tQuote + " "
        Else
            tStr = tStr + Trim(Str(!c_edge_count)) + " links" + tQuote + " "
        '
    End If

    tStr = tStr + "c " + tColor(!c_edge_dist + 1)
    '    color = white (1), blue (2), green (3), yellow (4), orange (5)
    '
    tStream.WriteText tStr, adWriteLine
    '
    .MoveNext
Loop
End With
End If
'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tRstNodeList.Close

tStream.Close
Set tStream = Nothing
'
'Set tGDF = Nothing
'Set tFileSystem = Nothing
Set tRstNodeList = Nothing
Set tRstEdgeList = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdUTF8Pajek_Click:
Exit Sub

Err_CmdUTF8Pajek_Click:
MsgBox Err.Description
Resume Exit_CmdUTF8Pajek_Click

End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim tRstDummy As DAO.Recordset
    Dim cmdDel As ADODB.Command, tRecDeleted As Long
    '
    ' set the language
    Dim tmli As MsoLanguageID
    ' get the labels
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    gLabelsOK = True
    If tmli = msoLanguageIDSimplifiedChinese Then
        gDisplayLanguage = "S"
        Call changeDisplayLanguage
    ElseIf tmli = msoLanguageIDTraditionalChinese Then
        gDisplayLanguage = "T"
        Call changeDisplayLanguage
    ElseIf tmli = msoLanguageIDEnglishUS Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "E"
    End If
    '
    gUsePersonID = False
    gUseADDRID = False
    'gRerunQuery = False

    gFromDynasty = -1
    gToDynasty = -1

```

```

Me.CmdAllPlaces.Enabled = False
Me.ChkPlaceLimit.Enabled = False
'Me.ChkIndexYear.Value = True
If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
    CmdRecallID.Enabled = True
End If
'
' Clear the Edge output table
'
Set cmdDel = New ADODB.Command
cmdDel.ActiveConnection = CurrentProject.Connection
cmdDel.CommandType = adCmdText
'
Set gRstEdge = Forms!LookAtNetworks!ZZ_SOCIAL_NETWORK.Form.Recordset
'
If gRstEdge.RecordCount > 0 Then
    '
    Set Forms!LookAtNetworks!ZZ_SOCIAL_NETWORK.Form.Recordset = _
        CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SN", dbOpenDynaset)
    gRstEdge.Close
    '
    cmdDel.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
    cmdDel.Execute tRecDeleted
    '
    Set gRstEdge = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
    Set Forms!LookAtNetworks!ZZ_SOCIAL_NETWORK.Form.Recordset = gRstEdge
    '
End If
'
Set Forms!LookAtNetworks!ZZ_SOCIAL_NETWORK_AGGREGATED.Form.Recordset = _
    CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SN", dbOpenDynaset)
'
cmdDel.CommandText = "Delete * from ZZ_SOCIAL_NETWORK_AGGREGATE"
cmdDel.Execute tRecDeleted
'
Set ZZ_SOCIAL_NETWORK_AGGREGATED.Form.Recordset =
    CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK_AGGREGATE", dbOpenDynaset)
'
' Clear the Node output table
'
Set tRstDummy = ZZ_SCRATCH_PEOPLE.Form.Recordset
'
If tRstDummy.RecordCount > 0 Then
    '
    Set ZZ_SCRATCH_PEOPLE.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SP", dbOpenDynaset)
    tRstDummy.Close
    '
    cmdDel.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
    cmdDel.Execute tRecDeleted
    '
    Set ZZ_SCRATCH_PEOPLE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
End If
'
' we no longer need to open the message window
'
'DoCmd.OpenForm "ZZZ_DONT_PANIC", acNormal
'Forms("ZZZ_DONT_PANIC").Visible = False

'initialize some global variables
gMaxFilterTotal = 29
gMaxFilterScholar = 7
gMaxFilterWritings = 9
gMaxFilterPolitics = 6
gMaxFilterMilitary = 2
gFilterTotalCount = 29
gFilterScholarCount = 7
gFilterWritingsCount = 9
gFilterPoliticsCount = 6
gFilterMilitaryCount = 2

' zap the scratch files

cmdDel.CommandText = "Delete * from ZZ_SCRATCH_ADDR_LIST"
cmdDel.Execute tRecDeleted

cmdDel.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
cmdDel.Execute tRecDeleted

```



```

Private Sub CmdDeselect_Click()
On Error GoTo Err_CmdDeselect_Click

    Dim tTrue As Integer, tFalse As Integer
    tTrue = -1
    tFalse = 0

    ChkFriendship.Value = tFalse
    ChkMedicine.Value = tFalse
    ChkReligion.Value = tFalse
    ChkFamily.Value = tFalse
    ChkFinance.Value = tFalse

    ChkMilitarySupport.Value = tFalse
    ChkMilitaryOppose.Value = tFalse
    ChkMilitaryAll.Value = tFalse

    ChkScholarshipAll.Value = tFalse
    ChkSchTeacher.Value = tFalse
    ChkSchAffiliation.Value = tFalse
    ChkSchTopic.Value = tFalse
    ChkSchMember.Value = tFalse
    ChkSchPatron.Value = tFalse
    ChkSchLitArt.Value = tFalse
    ChkSchAttack.Value = tFalse

    ChkPoliticsAll.Value = tFalse
    ChkPolEqual.Value = tFalse
    ChkPolSub.Value = tFalse
    ChkPolSup.Value = tFalse
    ChkPolSupport.Value = tFalse
    ChkPolSponsor.Value = tFalse
    ChkPolOppose.Value = tFalse

    ChkWritingsAll.Value = tFalse
    ChkWriCommem.Value = tFalse
    ChkWriEpitaph.Value = tFalse
    ChkWriPreface.Value = tFalse
    ChkWriRitual.Value = tFalse
    ChkWriBiog.Value = tFalse
    ChkWriExplain.Value = tFalse
    ChkWriMottos.Value = tFalse
    ChkWriLetters.Value = tFalse
    ChkWriOccasion.Value = tFalse

Exit_CmdDeselect_Click:
    Exit Sub

Err_CmdDeselect_Click:
    MsgBox Err.Description
    Resume Exit_CmdDeselect_Click

End Sub

Private Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 89) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 89 And Not .EOF
            If !c_form = "LAN" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            
```

```

        End If
        .MoveNext
    Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If gLabelsOK Then
    If gDisplayLanguage = "E" Then
        tLang = 1
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '
    Me.CmdSelectPerson.Caption = tLabelLanguage(tLang, 1)
    Me.CmdAllPeople.Caption = tLabelLanguage(tLang, 2)
    Me.CmdSelectPlace.Caption = tLabelLanguage(tLang, 3)
    Me.CmdAllPlaces.Caption = tLabelLanguage(tLang, 4)
    Me.LblFrom.Caption = tLabelLanguage(tLang, 5)
    Me.LblTo.Caption = tLabelLanguage(tLang, 6)
    Me.LblMaxNode.Caption = tLabelLanguage(tLang, 7)
    Me.LblMaxLoop.Caption = tLabelLanguage(tLang, 8)
    Me.LblKin.Caption = tLabelLanguage(tLang, 9)
    Me.LblNonKin.Caption = tLabelLanguage(tLang, 10)
    Me.LblMale.Caption = tLabelLanguage(tLang, 11)
    Me.LblFemale.Caption = tLabelLanguage(tLang, 12)
    Me.CmdRun.Caption = tLabelLanguage(tLang, 13)

    Me.CmdFantiDisplay.Caption = tLabelLanguage(tLang, 14)
    Me.CmdJiantiDisplay.Caption = tLabelLanguage(tLang, 15)

    Me.PageRelFilter.Caption = tLabelLanguage(tLang, 16)
    Me.PageEdgeData.Caption = tLabelLanguage(tLang, 17)
    Me.PageNodeData.Caption = tLabelLanguage(tLang, 18)

    Me.LblChkSelectAll.Caption = tLabelLanguage(tLang, 19)
    Me.LblChkFriendship.Caption = tLabelLanguage(tLang, 20)
    Me.LblChkFamily.Caption = tLabelLanguage(tLang, 21)
    Me.LblChkReligion.Caption = tLabelLanguage(tLang, 22)
    Me.LblChkFinance.Caption = tLabelLanguage(tLang, 23)
    Me.LblChkMedicine.Caption = tLabelLanguage(tLang, 24)
    Me.LblChkMilitaryAll.Caption = tLabelLanguage(tLang, 25)
    Me.LblChkMilitarySupport.Caption = tLabelLanguage(tLang, 26)
    Me.LblChkMilitaryOppose.Caption = tLabelLanguage(tLang, 27)

    Me.LblChkScholarshipAll.Caption = tLabelLanguage(tLang, 28)
    Me.LblChkSchTeacher.Caption = tLabelLanguage(tLang, 29)
    Me.LblChkSchAffiliation.Caption = tLabelLanguage(tLang, 30)
    Me.LblChkSchTopic.Caption = tLabelLanguage(tLang, 31)
    Me.LblChkSchMember.Caption = tLabelLanguage(tLang, 32)
    Me.LblChkSchPatron.Caption = tLabelLanguage(tLang, 33)
    Me.LblChkSchLitArt.Caption = tLabelLanguage(tLang, 34)
    Me.LblChkSchAttack.Caption = tLabelLanguage(tLang, 35)

    Me.LblChkPoliticsAll.Caption = tLabelLanguage(tLang, 36)
    Me.LblChkPolEqual.Caption = tLabelLanguage(tLang, 37)
    Me.LblChkPolSub.Caption = tLabelLanguage(tLang, 38)
    Me.LblChkPolSup.Caption = tLabelLanguage(tLang, 39)
    Me.LblChkPolSupport.Caption = tLabelLanguage(tLang, 40)
    Me.LblChkPolSponsor.Caption = tLabelLanguage(tLang, 41)
    Me.LblChkPolOppose.Caption = tLabelLanguage(tLang, 42)

    Me.LblChkWritingsAll.Caption = tLabelLanguage(tLang, 43)
    Me.LblChkWriCommem.Caption = tLabelLanguage(tLang, 44)
    Me.LblChkWriEpitaph.Caption = tLabelLanguage(tLang, 45)
    Me.LblChkWriPreface.Caption = tLabelLanguage(tLang, 46)
    Me.LblChkWriRitual.Caption = tLabelLanguage(tLang, 47)
    Me.LblChkWriBiog.Caption = tLabelLanguage(tLang, 48)
    Me.LblChkWriExplain.Caption = tLabelLanguage(tLang, 49)
    Me.LblChkWriMotto.Caption = tLabelLanguage(tLang, 50)
    Me.LblChkWriLetters.Caption = tLabelLanguage(tLang, 51)
    Me.LblChkWriOccasion.Caption = tLabelLanguage(tLang, 52)

    Me.CmdUCINet.Caption = tLabelLanguage(tLang, 53)
    Me.CmdPajekCaption = tLabelLanguage(tLang, 54)
    ' Me.CmdUTF8Pajek.Caption = tLabelLanguage(tLang, 55)

```

```

83
Me.CmdGIS.Caption = tLabelLanguage(tLang, 56)
Me.CmdGUESS.Caption = tLabelLanguage(tLang, 57)
Me.CmdClose.Caption = tLabelLanguage(tLang, 58)

Me.LblSaveClipboard.Caption = tLabelLanguage(tLang, 59)

Me.Caption = tLabelLanguage(tLang, 60)

Me.CmdImportPeople.Caption = tLabelLanguage(tLang, 61)
Me.CmdImportPlaces.Caption = tLabelLanguage(tLang, 62)
'Me.LblChkIndexYear.Caption = tLabelLanguage(tLang, 63)

Me.PageAggregate.Caption = tLabelLanguage(tLang, 64)

Me.LblIncludeID.Caption = tLabelLanguage(tLang, 65)
Me.LblMaxUp.Caption = tLabelLanguage(tLang, 66)
Me.LblMaxDwn.Caption = tLabelLanguage(tLang, 67)
Me.LblMaxCol.Caption = tLabelLanguage(tLang, 68)
Me.LblMaxMar.Caption = tLabelLanguage(tLang, 69)
Me.LblKinshipParam.Caption = tLabelLanguage(tLang, 70)
Me.LblDisplay.Caption = tLabelLanguage(tLang, 71)
Me.CmdHelp.Caption = tLabelLanguage(tLang, 72)
'Me.CmdRerun.Caption = tLabelLanguage(tLang, 73)
Me.CmdStoreID.Caption = tLabelLanguage(tLang, 74)
Me.CmdRecallID.Caption = tLabelLanguage(tLang, 75)
Me.LblChkDegree.Caption = tLabelLanguage(tLang, 76)
Me.LblXYRef.Caption = tLabelLanguage(tLang, 77)
Me.Label152.Caption = tLabelLanguage(tLang, 78)
Me.LblChkSubUnits.Caption = tLabelLanguage(tLang, 79)

Me.LblDynasties.Caption = tLabelLanguage(tLang, 80)
Me.CmdFromDynasty.Caption = tLabelLanguage(tLang, 81)
Me.CmdToDynasty.Caption = tLabelLanguage(tLang, 82)
Me.CmdAllDynasties.Caption = tLabelLanguage(tLang, 83)

Me.LblIndexYears.Caption = tLabelLanguage(tLang, 84)
Me.LblOptNoDates.Caption = tLabelLanguage(tLang, 85)
Me.LblOptIndexYears.Caption = tLabelLanguage(tLang, 86)
Me.LblOptDynasties.Caption = tLabelLanguage(tLang, 87)

Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 88)

If gDisplayLanguage = "S" Or gDisplayLanguage = "T" Then
    Me.CmdClose.Caption = ChrW(&H9000) + ChrW(&H51FA)
Else
    Me.CmdClose.Caption = "Exit"
End If
End If

End Sub

Private Sub CmdHelp_Click()
On Error GoTo Err_CmdHelp_Click

Dim tStrPDF As String

tStrPDF = Application.CurrentProject.Path + "\HelpFiles\HelpFile_LookAtNetworks.pdf"

'MsgBox tStrPDF

Application.FollowHyperlink tStrPDF, , True

Exit_CmdHelp_Click:
Exit Sub

Err_CmdHelp_Click:
MsgBox Err.Description
Resume Exit_CmdHelp_Click

End Sub

Private Sub writeKML()
'<kml xmlns="http://www.opengis.net/kml/2.2">
'<Document>
'    <name>ExtendedData+SchemaData</name>
'    <open>1</open>
'    <!-- Create a balloon template referring to the user-defined type -->
'    <Style id="assoc-balloon-template">
'        <BalloonStyle>

```

```

    <text>
      <![CDATA[
        ${AssocPerson/PersonNameHZ} <br/>
        ID: ${AssocPerson/PersonID} <br/>
        Index Year: ${AssocPerson/IndexYear} <br/>
        Address: ${AssocPerson/AddrName} ${AssocPerson/AddrNameHZ} <br/>
        XY Count: ${AssocPerson/XYCount} <br/><br/>
      ]]>
    </text>
  </BalloonStyle>
</Style>
<!-- Declare the type "AssocPerson" with 6 fields -->
<Schema name="AssocPerson" id="AssocPersonId">
  <SimpleField type="string" name="PersonNameHZ">
    <displayName><![CDATA[<b>Person</b>]]></displayName>
  </SimpleField>
  <SimpleField type="string" name="AddrName">
    <displayName><![CDATA[<b>Person</b>]]></displayName>
  </SimpleField>
  <SimpleField type="string" name="AddrNameHZ">
    <displayName><![CDATA[<b>Person</b>]]></displayName>
  </SimpleField>
  <SimpleField type="uint" name="PersonID">
    <displayName><![CDATA[ID]]></displayName>
  </SimpleField>
  <SimpleField type="int" name="IndexYear">
    <displayName><![CDATA[Index Year]]></displayName>
  </SimpleField>
  <SimpleField type="int" name="XYCount">
    <displayName><![CDATA[XY Count]]></displayName>
  </SimpleField>
</Schema>
<!-- Instantiate some Placemarks extended with AssocPerson fields -->
<Placemark>
  <name>Easy trail</name>
  <styleUrl>#assoc-balloon-template</styleUrl>
  <ExtendedData>
    <SchemaData schemaUrl="#AssocPersonId">
      <SimpleData name="PersonID">3.14159</SimpleData>
      <SimpleData name="PersonNameHZ">Pi in the sky</SimpleData>
      <SimpleData name="IndexYear">10</SimpleData>
      <SimpleData name="AddrName">Pi in the sky</SimpleData>
      <SimpleData name="AddrNameHZ">Pi in the sky</SimpleData>
      <SimpleData name="XYCount">10</SimpleData>
    </SchemaData>
  </ExtendedData>
  <Point>
    <coordinates>-122.000,37.002</coordinates>
  </Point>
</Placemark>
<Placemark>
  <name>Difficult trail</name>
  <styleUrl>#assoc-balloon-template</styleUrl>
  <ExtendedData>
    <SchemaData schemaUrl="#AssocPersonId">
      <SimpleData name="TrailHeadName">Mount Everest</SimpleData>
      <SimpleData name="TrailLength">347.45</SimpleData>
      <SimpleData name="ElevationGain">10000</SimpleData>
    </SchemaData>
  </ExtendedData>
  <Point>
    <coordinates>-121.998,37.0078</coordinates>
  </Point>
</Placemark>
</Document>
</kml>

```

```
Dim tStrKML As String
```

```
' This program will dump the results to a .gis file
```

```
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
```

```
MsgBox "There are no records to save."
```

```
GoTo Exit_writeKML
```

```
End If
```

```
Dim tStream As ADODB.Stream
```

```
Set tStream = New ADODB.Stream
```

```
If GISFrame.Value = 1 Then
```

```

5
tStream.Charset = "utf-8"
tCodeStr = "UTF8"
Else
tStream.Charset = "gb2312"
tCodeStr = "GB2312"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, tDQ As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
'
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
tFileName = tFN
If Not tFileName = "" Then
Exit For
End If
Next
If tFileName = "" Then
MsgBox "Bad file Name."
GoTo Exit_writeKML
Else
' make sure the file name has a txt extension
If Len(tFileName) < 5 Then
tFileName = tFileName + ".kml"
ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
tFileName = tFileName + ".kml"
End If
End If
'
' write the file
'
'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
'
' process the table
'
Set tRstNode = ZZ_SCRATCH_PEOPLE.Form.Recordset
tC = Chr(9) ' the tab
tDQ = Chr(34) ' the double quotation mark
'
' write the header
'
tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
tStream.WriteText "<Document>", adWriteLine
tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
tStream.WriteText tC + "<open>1</open>", adWriteLine
tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

tStream.WriteText tC + "<Style id=" + tDQ + "assoc-balloon-template" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
tStream.WriteText tC + tC + tC + "<text>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "<![CDATA[" + tC + "[", adWriteLine
tStream.WriteText tC + tC + tC + tC + "$[AssocPerson/PersonNameHZ] <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "ID: $[AssocPerson/PersonID] <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Index Year: $[AssocPerson/IndexYear] <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Address: $[AssocPerson/AddrName] $[AssocPerson/AddrNameHZ] <br/>",
adWriteLine
tStream.WriteText tC + tC + tC + tC + "XY Count: $[AssocPerson/XYCount] <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine
tStream.WriteText tC + "<!-- Declare the type " + tDQ + "AssocPerson" + tDQ + " with 6 fields -->", adWriteLine

tStream.WriteText tC + "<Schema name=" + tDQ + "AssocPerson" + tDQ + " id=" + tDQ + "AssocPersonId" + tDQ
+ ">", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "PersonNameHZ"

```

```

+ tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[<b>Person</b>]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + t
DQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[<b>Person</b>]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrNameHZ" +
tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[<b>Person</b>]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ
+ ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[ID]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" +
tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs, even where there should not be any
        '
        ' write the point header
        '
        tStream.WriteText tC + "<Placemark>", adWriteLine

        If IsNull(!c_name) Then
            tStr = "[Bad Data]"
        Else
            tStr = !c_name
        End If
        tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

        tStream.WriteText tC + tC + "<styleUrl>#assoc-balloon-template</styleUrl>", adWriteLine
        '
        ' Index Year as time stamp
        '
        If IsNull(!c_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_index_year)
        End If
        tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
        '
        tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#AssocPersonId" + tDQ + ">", a
dWriteLine
        '
        ' person ID
        '
        tStr = Str(!c_person_id)
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
        '
        ' Chinese Name
        '
        If IsNull(!c_name_chn) Then
            tStr = "[Bad Data]"
        Else
            If Trim(!c_name_chn) = "" Then
                tStr = "[?]"
            Else
                tStr = !c_name_chn
            End If
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonNameHZ" + tDQ + ">" + tS
tr + "</SimpleData>", adWriteLine
        '
        ' Index Year
        '
        If IsNull(!c_index_year) Then

```

```

        tStr = "N/A"
    Else
        tStr = Str(!c_index_year)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr +
+ "</SimpleData>", adWriteLine
    '
    ' Address Name
    '
    If IsNull(!c_addr_name) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_name) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_name
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
+ "</SimpleData>", adWriteLine
    '
    ' Address Name Chinese
    '
    If IsNull(!c_addr_chn) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_chn
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrNameHZ" + tDQ + ">" + tStr +
+ "</SimpleData>", adWriteLine
    '
    ' XY Count
    '
    If IsNull(!xy_count) Then
        tStr = "0"
    Else
        tStr = Str(!xy_count)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
+ "</SimpleData>", adWriteLine
    '
    tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
    tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
    tStream.WriteText tC + tC + "<Point>", adWriteLine
    '
    ' coordinates
    If IsNull(!x_coord) Then
        tStr = "0"
    Else
        tStr = Str(!x_coord)
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + ",0"
    Else
        tStr = tStr + "," + Str(!y_coord)
    End If
    tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
    '
    ' footer
    '

    tStream.WriteText tC + tC + "</Point>", adWriteLine
    tStream.WriteText tC + "</Placemark>", adWriteLine
    .MoveNext
Loop
End With
'
' footer
'

tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite

```

```

Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_writeKML:
Exit Sub

Err_writeKML:
MsgBox Err.Description
Resume Exit_writeKML

End Sub

Private Sub CheckRunCriteria()
' This routine checks whether it is OK to run the query and either enables or disables the CmdRun button

Dim tTrue As Integer, tFalse As Integer
tTrue = -1
tFalse = 0

' using nonkin

If ChkNonKin.Value = tTrue Then
If gFilterTotalCount = gMaxFilterTotal Then
If gUsePersonID Or gUseADDRID Then
CmdRun.Enabled = True
Else
CmdRun.Enabled = False
End If
ElseIf gFilterTotalCount = 0 Then
If (gUsePersonID Or gUseADDRID) And ChkKin.Value = tTrue Then
CmdRun.Enabled = True
Else
CmdRun.Enabled = False
End If
Else
CmdRun.Enabled = True
End If
ElseIf ChkKin.Value = tTrue Then
If gUsePersonID Or gUseADDRID Then
CmdRun.Enabled = True
Else
CmdRun.Enabled = False
End If
Else
CmdRun.Enabled = False
End If
'

End Sub

Private Sub CmdStoreID_Click()
Dim cmdSQL As ADODB.Command, tRecCount As Variant

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
' Display message.
If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) = vbNo Then
Exit Sub
Else
cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
cmdSQL.Execute tRecCount
End If
End If

tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id FROM ZZ_SCRATCH_PEOPLE"

cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecCount
MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."
'
' update storage source
cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='Networks' WHERE PersonIDSource.LineNum =1"
cmdSQL.Execute tRecCount

```



```

End Sub
Private Sub CmdRecallID_Click()
On Error GoTo Err_CmdRecallID_Click
Dim tStrSQL As String, cmdSQL As ADODB.Command, tRecCount As Variant, tRst As DAO.Recordset

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

If DCount("*", "ZZ_SCRATCH_IMPORT_PEOPLE") > 0 Then
' Display message.
If MsgBox("Do you wish to replace the current import list?", vbYesNo + vbQuestion + vbDefaultButton2) = vbYes Then
bNo Then
Exit Sub
Else
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecCount
End If
End If
'
' Clear the error table now that we are ready to go
'
cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecCount
'
' copy the IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id ) SELECT DISTINCT c_personid FROM ZZ_STORE_PERSON_ID"
ON_ID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecCount

If tRecCount = 0 Then
TxtName.Value = "[Error]"
TxtNameChn.Value = "[Error]"

gUsePersonID = False
CmdAllPeople.Enabled = False
CmdRun.Enabled = False
'CmdRerun.Enabled = True
Else
If tRecCount = 1 Then
Set tRst = CurrentDb.OpenRecordset("SELECT ZZ_STORE_PERSON_ID.c_personid FROM ZZ_STORE_PERSON_ID")
tRst.MoveFirst
tID = tRst!c_personid

Set tRst = CurrentDb.OpenRecordset("SELECT BIOG_MAIN.c_name, BIOG_MAIN.c_name_chn FROM BIOG_MAIN WHERE ((BIOG_MAIN.c_personid)=" + Str(tID) + ")")
tRst.MoveFirst

TxtName.Value = tRst!c_name
TxtNameChn.Value = tRst!c_name_chn
tRst.Close
Set tRst = Nothing
Else
TxtName.Value = "[Recalled List]"
TxtNameChn.Value = "[" + ChrW(&H53EC) + ChrW(&H56DE) + ChrW(&H7684) + ChrW(&H4EBA) + ChrW(&H540D) + "]"
]

End If
' zhao = 53EC, hui = 56DE, de = 7684, ren = 4EBA, ming = 540D

gUsePersonID = True
CmdAllPeople.Enabled = True
CmdRun.Enabled = True
'CmdRerun.Enabled = Talse
End If

Set cmdSQL = Nothing

Exit_CmdRecallID_Click:
Exit Sub

Err_CmdRecallID_Click:
MsgBox Err.Description
Resume Exit_CmdRecallID_Click

End Sub

```

```
Private Sub FrameFilterYears_Click()
```

```

'
' the simplest approach is to turn it all off and then turn on the appropriate objects

' disable all
Me.CmdFromDynasty.Enabled = False
Me.CmdToDynasty.Enabled = False
Me.CmdAllDynasties.Enabled = False
Me.TxtFromDynasty.Enabled = False
Me.TxtFromDynastyPY.Enabled = False
Me.TxtToDynasty.Enabled = False
Me.TxtToDynastyPY.Enabled = False
Me.TxtFromDynasty.Locked = False
Me.TxtFromDynastyPY.Locked = False
Me.TxtToDynasty.Locked = False
Me.TxtToDynastyPY.Locked = False

Me.TxtFrom.Enabled = False
Me.TxtTo.Enabled = False

gUseIndexYears = False
gUseDynasties = False

If FrameFilterYears.Value = 2 Then

    ' enable index years
    Me.TxtFrom.Enabled = True
    Me.TxtTo.Enabled = True
    gUseIndexYears = True

ElseIf FrameFilterYears.Value = 3 Then

    ' enable dynasties
    Me.CmdFromDynasty.Enabled = True
    Me.CmdToDynasty.Enabled = True
    Me.CmdAllDynasties.Enabled = True
    Me.TxtFromDynasty.Enabled = True
    Me.TxtFromDynastyPY.Enabled = True
    Me.TxtToDynasty.Enabled = True
    Me.TxtToDynastyPY.Enabled = True
    Me.TxtFromDynasty.Locked = True
    Me.TxtFromDynastyPY.Locked = True
    Me.TxtToDynasty.Locked = True
    Me.TxtToDynastyPY.Locked = True
    gUseDynasties = True

End If
'MsgBox "FrameFilterYears = " + Str(FrameFilterYears.Value)
'MsgBox "gUseIndexYears = " + IIf(gUseIndexYears, "True", "False")

```

```
End Sub
```

```
1
Option Compare Database
Public gDisplayLanguage As String, gLabelsOK As Boolean, gPersonID As Long
```

```
Private Sub CmdSaveToFile_Click()
On Error GoTo Err_CmdSaveToFile_Click
    Dim tPersonName As String
    '
    ' first, just test if it knows the perdon ID
    '
    gPersonID = Me.BIOG_MAIN_2_Subform.Form.c_personid.Value
    MsgBox Str(gPersonID)
    tPersonName = Me.frmPeopleLookup2.Form.c_name_chn.Value
    If IsNull(tPersonName) Then
        MsgBox "Name is NULL"
        Exit Sub
    End If

    ' the routine tests whether thre isinformation to be written for each category
    '
    ' set the language
    Dim tmli As MsoLanguageID, tLang As String
    ' get the labels
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)

    If tmli = msoLanguageIDSimplifiedChinese Then
        tLang = "S"
    ElseIf tmli = msoLanguageIDTraditionalChinese Then
        tLang = "T"
    ElseIf tmli = msoLanguageIDEnglishUS Then
        tLang = "E"
    Else
        tLang = "E"
    End If

    '
    ' The challenge here is that we have 7 tables of results:
    '     ZZZ_BIOG_MAIN
    '     ZZ_SCRATCH_KIN
    '     ZZ_SCRATCH_STATUS
    '     ZZ_SCRATCH_OFFICE
    '     ZZ_SCRATCH_ENTRY
    '     ZZ_SCRATCH_BIOG_TEXT_DATA
    '     ZZ_SCRATCH_BIOG_ADDR_DATA
    '
    ' We need to check all of these for people (entry has 3 IDs) and address IDs, along with their specific data
    '
    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer, tFileName As String, tFN As Variant
    '
    Dim tRstPeople As DAO.Recordset, tRst As DAO.Recordset
    Dim tStr As String, tC As String
    Dim tQueryStr As String, tPersonID As Long, tCount As Long, tStrPersonID As String
    '

    tStrPersonID = Str(gPersonID)

    Dim gStream As ADODB.Stream, tCodeStr As String
    '
    ' set up the stream to write to

    Set gStream = New ADODB.Stream
    '
    gStream.Charset = "utf-8"
    tCodeStr = "UTF8"
    '
    ' Other options
    'gStream.Charset = "big5"
    'tCodeStr = "BIG5"
    'gStream.Charset = "gb2312"
    'tCodeStr = "GB2312"
    'gStream.Charset = "ascii"
    'tCodeStr = "ascii"
    '
    tC = Chr(44) ' the comma
    '

    Dim cmdSQL As ADODB.Command, tRecCount As Long
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
```

```
' get the basic information
```

```
' Create the file
```

```
Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)
```

```
dlgSaveAs.InitialFileName = tPersonName + tCodeStr + ".htm"
```

```
If dlgSaveAs.Show = -1 Then
```

```
    '
    tFileName = ""
```

```
    For Each tFN In dlgSaveAs.SelectedItems
```

```
        tFileName = tFN
```

```
        If Not tFileName = "" Then
```

```
            Exit For
```

```
        End If
```

```
    Next
```

```
    If tFileName = "" Then
```

```
        MsgBox "Bad file Name."
```

```
        GoTo Exit_CmdSaveToFile_Click
```

```
    Else
```

```
        ' make sure the file name has a htm extension
```

```
        If Len(tFileName) < 5 Then
```

```
            tFileName = tFileName + ".htm"
```

```
        ElseIf Not (LCase(Right(tFileName, 4)) = ".htm") Then
```

```
            tFileName = tFileName + ".htm"
```

```
        End If
```

```
    End If
```

```
    '
```

```
    ' we have a file name: now open the stream for writing
```

```
    gStream.Mode = adModeReadWrite
```

```
    gStream.Type = adTypeText
```

```
    gStream.Open
```

```
    tQueryStr = "SELECT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MA  
IN.c_index_year, " +
```

```
        "ZZZ_BIOG_MAIN.c_index_year_type_desc, ZZZ_BIOG_MAIN.c_index_year_type_hz, ZZZ_BIOG_MAIN.c_dynast
```

```
y, " +
```

```
        "ZZZ_BIOG_MAIN.c_dynasty_chn, ZZZ_BIOG_MAIN.c_female, ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_M
```

```
AIN.c_index_addr_chn, " +
```

```
        "ZZZ_BIOG_MAIN.c_index_addr_type_desc, ZZZ_BIOG_MAIN.c_index_addr_type_chn, ZZZ_BIOG_MAIN.x_coord
```

```
, " +
```

```
        "ZZZ_BIOG_MAIN.y_coord, ZZZ_BIOG_MAIN.c_birthyear, ZZZ_BIOG_MAIN.c_deathyear, ZZZ_BIOG_MAIN.c_dea
```

```
th_age, " +
```

```
        "ZZZ_BIOG_MAIN.c_fl_earliest_year, ZZZ_BIOG_MAIN.c_fl_latest_year, ZZZ_BIOG_MAIN.c_fl_ly_nh_year,
```

```
" +
```

```
        "ZZZ_BIOG_MAIN.c_name_proper, ZZZ_BIOG_MAIN.c_name_rm, ZZZ_BIOG_MAIN.c_ethnicity_chn, ZZZ_BIOG_MA
```

```
IN.c_ethnicity_rmn, " +
```

```
        "ZZZ_BIOG_MAIN.c_choronym_desc, ZZZ_BIOG_MAIN.c_choronym_chn, " +
```

```
        "ZZZ_BIOG_MAIN.c_household_status_desc, ZZZ_BIOG_MAIN.c_household_status_desc_chn " +
```

```
        "FROM ZZZ_BIOG_MAIN " +
```

```
        "WHERE ((ZZZ_BIOG_MAIN.c_personid)=" + tStrPersonID + ")))"
```

```
    MsgBox "Writing header"
```

```
    gStream.WriteText "<HTML>", adWriteLine
```

```
    gStream.WriteText "<BODY>", adWriteLine
```

```
    tStr = "<P><B>Basic Information</B></P>"
```

```
    gStream.WriteText tStr, adWriteLine
```

```
    MsgBox "Getting Person Data"
```

```
    Set tRstPeople = CurrentDb.OpenRecordset(tQueryStr)
```

```
    MsgBox "Writing Person Basic Data"
```

```
    With tRstPeople
```

```
        .MoveFirst
```

```
        Do While Not .EOF
```

```
            ' the ID of the person
```

```
            gStream.WriteText "<P><I>CBDB ID</I>", adWriteLine
```

```
            gStream.WriteText Trim(Str(!c_personid)) + "</P>", adWriteLine
```

```
            '
```

```
            ' name
```

```
            '
```

```
            gStream.WriteText "<P><I>Name</I>", adWriteLine
```

```
            If IsNull(!c_name) Then
```

```
                tStr = "[Missing]"
```

```
            Else
```

```
                tStr = !c_name
```

```
            End If
```

```
            '
```

```
            If IsNull(!c_name_chn) Then
```

```

        tStr = tStr + " [Missing]"
Else
    tStr = tStr + " " + !c_name_chn
End If
gStream.WriteText tStr, adWriteLine

' c_name_proper, c_name_rm
If Not IsNull(!c_name_proper) Then
    gStream.WriteText "<BR>" + !c_name_proper, adWriteLine
End If
If Not IsNull(!c_name_rm) Then
    gStream.WriteText "<BR>" + !c_name_rm, adWriteLine
End If
gStream.WriteText "</P>", adWriteLine

gStream.WriteText "<P><I>Sex</I>", adWriteLine
If IsNull(!c_female) Then
    tStr = "[Unknown]"
Else
    tStr = IIf(!c_female, "F", "M")
End If
gStream.WriteText tStr + "</P>", adWriteLine
'
' indexyear = c_index_year INT
'
gStream.WriteText "<P><I>Index Year</I>", adWriteLine
If IsNull(!c_index_year) Then
    tStr = "[Unknown]"
Else
    tStr = Trim(Str(!c_index_year))
End If
gStream.WriteText tStr + "<BR>", adWriteLine
'
' indexyear = c_index_year_type_desc STR
'
gStream.WriteText "<I>Index Year Type</I>", adWriteLine
If IsNull(!c_index_year_type_desc) Then
    tStr = "Unknown "
Else
    tStr = Trim(!c_index_year_type_desc) + " "
End If
'
' indexyear = c_index_year_type_hz STR
'
If Not IsNull(!c_index_year_type_hz) Then
    tStr = tStr + Trim(!c_index_year_type_hz)
End If
gStream.WriteText tStr, adWriteLine

' Additional Year Information

"c_birthday, c_deathyear, c_death_age, c_fl_earliest_year, c_fl_latest_year "
If Not IsNull(!c_birthday) Then
    gStream.WriteText "<BR>Birth Year: " + Str(!c_birthday), adWriteLine
End If
If Not IsNull(!c_deathyear) Then
    gStream.WriteText "<BR>Death Year: " + Str(!c_deathyear), adWriteLine
End If
If Not IsNull(!c_death_age) Then
    gStream.WriteText "<BR>Death Age: " + Str(!c_death_age), adWriteLine
End If
If Not IsNull(!c_fl_earliest_year) Then
    gStream.WriteText "<BR>Earliest Floruit Year: " + Str(!c_fl_earliest_year), adWriteLine
End If
If Not IsNull(!c_fl_latest_year) Then
    gStream.WriteText "<BR>Latest Floruit Year: " + Str(!c_fl_latest_year), adWriteLine
End If
gStream.WriteText "</P>", adWriteLine

' dynasty information
'
tStr = "<P><I>Dynasty</I>: "
If IsNull(!c_dynasty) Then
    tStr = tStr + "[Unknown]"
Else
    tStr = tStr + !c_dynasty + " " + !c_dynasty_chn
End If
gStream.WriteText tStr + "</P>", adWriteLine
'
' Index Address information

```

```

'c_index_addr_name, c_index_addr_chn, c_index_addr_type_desc, c_index_addr_type_chn, x_coord, y_c
coord, "

gStream.WriteText "<P><I>Index Address</I>", adWriteLine
If IsNull(!c_index_addr_name) Then
    gStream.WriteText "[Unknown]</P>", adWriteLine
Else
    tStr = Trim(!c_index_addr_name) + " " + Trim(!c_index_addr_chn)
    gStream.WriteText tStr + "<BR>", adWriteLine
    If Not IsNull(!x_coord) Then
        gStream.WriteText "Coordinates: " + Str(!x_coord) + tC + Str(!y_coord) + "<BR>", adWriteL
    End If
    '
    gStream.WriteText "<I>Index Address Type</I>", adWriteLine
    If IsNull(!c_index_addr_type_desc) Then
        tStr = "Unknown "
    Else
        tStr = Trim(!c_index_addr_type_desc) + " "
    End If
    '
    If Not IsNull(!c_index_addr_type_chn) Then
        tStr = tStr + Trim(!c_index_addr_type_chn)
    End If
    gStream.WriteText tStr + "</P>", adWriteLine
End If

'", c_ethnicity_chn, c_ethnicity_rmn, "
If Not IsNull(!c_ethnicity_chn) Then
    gStream.WriteText "<P>Ethnicity: " + !c_ethnicity_rmn + " " + !c_ethnicity_chn + "</P>", adWr
iteLine

End If
'"c_choronym_desc, c_choronym_chn, "
If Not IsNull(!c_choronym_desc) Then
    gStream.WriteText "<P>Choronym: " + !c_choronym_desc + " " + !c_choronym_chn + "</P>", adWrit
eLine

End If
'"c_household_status_desc, c_household_status_desc_chn "
If Not IsNull(!c_household_status_desc) Then
    gStream.WriteText "<P>Household Status: " + !c_household_status_desc + " " + !c_household_sta
tus_desc_chn + "</P>", adWriteLine
End If
.MoveNext

Loop
End With
Else
    'The user pressed Cancel.
    GoTo Exit_CmdSaveToFile_Click
End If
'
' Alt Names
'
MsgBox "Writing alternate names"
tQueryStr = "SELECT ZZZ_ALTNAME_DATA.c_personid, ZZZ_ALTNAME_DATA.c_alt_name, ZZZ_ALTNAME_DATA.c_alt_name_chn
, " + _
    "ZZZ_ALTNAME_DATA.c_name_type_desc, ZZZ_ALTNAME_DATA.c_name_type_desc_chn, ZZZ_ALTNAME_DATA.c_sequenc
e, ZZZ_ALTNAME_DATA.c_source, " + _
    "ZZZ_ALTNAME_DATA.c_title_chn, ZZZ_ALTNAME_DATA.c_title, ZZZ_ALTNAME_DATA.c_pages, ZZZ_ALTNAME_DATA.c
_notes " + _
    "FROM ZZZ_ALTNAME_DATA " + _
    "WHERE ((ZZZ_ALTNAME_DATA.c_personid)=" + tStrPersonID + ")")
Set tRst = CurrentDb.OpenRecordset(tQueryStr)

If Not (tRst.EOF) Then
    '
    tRst.MoveLast
    tStr = "<P><B><I>Alternate Names</I></B> "
    If tRst.RecordCount = 1 Then
        tStr = tStr + "(1 record)"
    Else
        tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
    End If
    gStream.WriteText tStr + "</P>", adWriteLine
    '
    ' !c_alt_name, !c_alt_name_chn, " + _
    ' !c_name_type_desc, !c_name_type_desc_chn, !c_sequence, !c_source, " + _
    ' !c_title_chn, !c_title, !c_pages, !c_notes
    '
    With tRst

```

```

.MoveFirst
Do While Not .EOF
    ,
    tStr = "<P><I>Name</I>: " + Trim(!c_alt_name) + " " + Trim(!c_alt_name_chn) + "<BR>"
    gStream.WriteText tStr, adWriteLine
    ,
    tStr = "Type: " + Trim(!c_name_type_desc) + " " + Trim(!c_name_type_desc_chn) + "<BR>"
    gStream.WriteText tStr, adWriteLine
    ,
    If Not IsNull(!c_sequence) Then
        tStr = "Sequence: " + Trim(Str(!c_sequence)) + "<BR>"
        gStream.WriteText tStr, adWriteLine
    End If
    ,
    If Not IsNull(!c_title) Then
        tStr = "Source: " + Trim(!c_title) + " " + Trim(!c_title_chn)
        If Not IsNull(!c_pages) Then
            tStr = tStr + " (" + Trim(!c_pages) + ")"
        End If
        gStream.WriteText tStr + "<BR>", adWriteLine
    End If
    ,
    If Not IsNull(!c_notes) Then
        tStr = "Notes: " + Trim(!c_notes) + "<BR>"
        gStream.WriteText tStr, adWriteLine
    End If
    ,
    gStream.WriteText "</P>", adWriteLine
    ,
    .MoveNext
Loop
End With
End If
,
' now Places
,
' replace the tLang condition with the count of addresses
,
tQueryStr = "SELECT ZZZ_BIOG_ADDR_DATA.c_personid, ZZZ_BIOG_ADDR_DATA.c_addr_name, ZZZ_BIOG_ADDR_DATA.c_addr_
chn, " + _
"ZZZ_BIOG_ADDR_DATA.x_coord, ZZZ_BIOG_ADDR_DATA.y_coord, ZZZ_BIOG_ADDR_DATA.c_addr_desc, ZZZ_BIOG_ADDR
R_DATA.c_addr_desc_chn, " + _
"ZZZ_BIOG_ADDR_DATA.c_firstyear, ZZZ_BIOG_ADDR_DATA.c_lastyear, ZZZ_BIOG_ADDR_DATA.c_source_title, ZZ
Z_BIOG_ADDR_DATA.c_source_chn, " + _
"ZZZ_BIOG_ADDR_DATA.c_pages, ZZZ_BIOG_ADDR_DATA.c_notes " + _
"FROM ZZZ_BIOG_ADDR_DATA " + _
"WHERE (((ZZZ_BIOG_ADDR_DATA.c_personid)=" + tStrPersonID + "))"

Set tRst = CurrentDb.OpenRecordset(tQueryStr)

MsgBox "Writing address information"
If Not (tRst.EOF) Then
    ,
    ' now the BIOG_ADDR data
    ,
    tRst.MoveLast
    tStr = "<P><B><I>Place Information</I></B> "
    If tRst.RecordCount = 1 Then
        tStr = tStr + "(1 record)"
    Else
        tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
    End If
    gStream.WriteText tStr + "</P>", adWriteLine
    ,
    ' c_addr_name, c_addr_chn, x_coord, y_coord, c_addr_desc, c_addr_desc_chn, "
    ' c_firstyear, c_lastyear, "
    ' c_source_title, c_source_chn, c_pages, c_notes "

    With tRst
        .MoveFirst
        Do While Not .EOF
            tStr = "<P><I>" + Trim(!c_addr_desc) + " " + Trim(c_addr_desc_chn) + "</I>: " + Trim(!c_addr_name
) + " " + Trim(!c_addr_chn)
            gStream.WriteText tStr, adWriteLine

            If Not IsNull(!x_coord) Then
                ,
                tStr = " (" + Trim(Str(!x_coord)) + " " + Trim(Str(!y_coord)) + ")<BR>"
                gStream.WriteText tStr, adWriteLine
            End If
        End While
    End With

```

```

If Not IsNull(!c_firstyear) Then
    tStr = "First year: " + Trim(Str(!c_firstyear)) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If

If Not IsNull(!c_lastyear) Then
    tStr = "Last year: " + Trim(Str(!c_lastyear)) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If

If Not IsNull(!c_source_title) Then
    tStr = "Source: " + Trim(!c_source_title) + " " + Trim(!c_source_chn)
    If Not IsNull(!c_pages) Then
        tStr = tStr + " " + Trim(!c_pages)
    End If
    tStr = tStr + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If

If Not IsNull(!c_notes) Then
    tStr = "Notes: " + Trim(!c_notes) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
gStream.WriteText "</P>", adWriteLine
.MoveNext
Loop
End With
'
End If

' now kinship
'
tQueryStr = "SELECT ZZ_SCRATCH_KIN.c_kin_id, ZZ_SCRATCH_KIN.c_kin_name, ZZ_SCRATCH_KIN.c_kin_chn, ZZ_SCRATCH_KIN.c_kin_index_year, " + _
    "ZZ_SCRATCH_KIN.c_kin_dynasty, ZZ_SCRATCH_KIN.c_kin_dynasty_chn, ZZ_SCRATCH_KIN.c_kin_sex, ZZ_SCRATCH_KIN.c_kin_rel_total, " + _
    "ZZ_SCRATCH_KIN.c_up, ZZ_SCRATCH_KIN.c_down, ZZ_SCRATCH_KIN.c_marriage, ZZ_SCRATCH_KIN.c_collateral, ZZ_SCRATCH_KIN.c_kin_addr_name, " + _
    "ZZ_SCRATCH_KIN.c_kin_addr_chn, ZZ_SCRATCH_KIN.kin_x_coord, ZZ_SCRATCH_KIN.kin_y_coord, ZZ_SCRATCH_KIN.c_source_text, " + _
    "ZZ_SCRATCH_KIN.c_source_text_chn, ZZ_SCRATCH_KIN.c_pages, ZZ_SCRATCH_KIN.c_notes " + _
    "FROM ZZ_SCRATCH_KIN"
Set tRst = CurrentDb.OpenRecordset(tQueryStr)

MsgBox "Writing kinship data"
If Not (tRst.EOF) Then
    '
    tRst.MoveLast
    tStr = "<P><B><I>Kinship</I></B> "
    If tRst.RecordCount = 1 Then
        tStr = tStr + "(1 record)"
    Else
        tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
    End If
    gStream.WriteText tStr + "</P>", adWriteLine

    ' c_kin_id, c_kin_name, c_kin_chn, c_kin_sex,
    ' c_kin_rel_total, c_up, c_down, c_marriage, c_collateral,
    ' c_kin_index_year, c_kin_dynasty, c_kin_dynasty_chn,
    ' c_kin_addr_name, c_kin_addr_chn, kin_x_coord, kin_y_coord,
    ' c_source_text, c_source_text_chn, c_pages, c_notes"
    '
    With tRst
        .MoveFirst
        Do While Not .EOF
            tStr = "<P>" + Trim(!c_kin_name) + " " + Trim(!c_kin_chn) + "(" + !c_kin_sex + ") [CBDB ID " + Trim(Str(!c_kin_id)) + "><BR>"
            gStream.WriteText tStr, adWriteLine

            tStr = "Relationship: " + Trim(!c_kin_rel_total) + " (" + Str(!c_up) + "-" + Str(!c_down) + "-" + Str(!c_marriage) + "-" + Str(!c_collateral) + "><BR>"
            gStream.WriteText tStr, adWriteLine

            If Not IsNull(!c_kin_index_year) Then
                tStr = "Index Year: " + Trim(Str(!c_kin_index_year)) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If
        End While
    End With

```



```

If Not IsNull(!c_kin_dynasty) Then
    tStr = "Dynasty: " + Trim(!c_kin_dynasty) + " " + Trim(!c_kin_dynasty_chn) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If

If Not IsNull(!c_kin_addr_name) Then
    tStr = "Index Address: " + Trim(!c_kin_addr_name) + " " + Trim(!c_kin_addr_chn)
    If Not IsNull(!kin_x_coord) Then
        tStr = tStr + " (" + Trim(Str(!kin_x_coord)) + tC + Trim(Str(!kin_y_coord)) + ")"
    End If
    tStr = tStr + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If

If Not IsNull(!c_source_text) Then
    tStr = "Source: " + Trim(!c_source_text) + " " + Trim(!c_source_text_chn)
    If Not IsNull(!c_pages) Then
        tStr = tStr + " (" + Trim(!c_pages) + ")"
    End If
    tStr = tStr + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If

If Not IsNull(!c_notes) Then
    tStr = "Notes: " + Trim(!c_notes) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
gStream.WriteText "</P>", adWriteLine
.MoveNext
Loop
End With
'
End If
'
' now Associations
'
tQueryStr = "SELECT ZZZ_NONKIN_BIOG_ADDR.c_personid, ZZZ_NONKIN_BIOG_ADDR.c_node_id, ZZZ_NONKIN_BIOG_ADDR.c_node_name, " + _
    "ZZZ_NONKIN_BIOG_ADDR.c_node_chn , ZZZ_NONKIN_BIOG_ADDR.c_node_index_year, ZZZ_NONKIN_BIOG_ADDR.c_node_dynasty, " + _
    "ZZZ_NONKIN_BIOG_ADDR.c_node_dynasty_chn, ZZZ_NONKIN_BIOG_ADDR.c_node_female, ZZZ_NONKIN_BIOG_ADDR.c_node_link_chn, " + _
    "ZZZ_NONKIN_BIOG_ADDR.c_link_desc, ZZZ_NONKIN_BIOG_ADDR.c_lit_genre_desc, ZZZ_NONKIN_BIOG_ADDR.c_lit_genre_desc_chn, " + _
    "ZZZ_NONKIN_BIOG_ADDR.c_occasion_desc, ZZZ_NONKIN_BIOG_ADDR.c_occasion_desc_chn, ZZZ_NONKIN_BIOG_ADDR.c_topic_desc, " + _
    "ZZZ_NONKIN_BIOG_ADDR.c_topic_desc_chn, ZZZ_NONKIN_BIOG_ADDR.c_text_title, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_name, " + _
    "ZZZ_NONKIN_BIOG_ADDR.c_node_addr_chn, ZZZ_NONKIN_BIOG_ADDR.node_xcoord, ZZZ_NONKIN_BIOG_ADDR.node_ycoord, " + _
    "ZZZ_NONKIN_BIOG_ADDR.c_inst_name_hz, ZZZ_NONKIN_BIOG_ADDR.c_inst_name_py, ZZZ_NONKIN_BIOG_ADDR.c_link_count, " + _
    "ZZZ_NONKIN_BIOG_ADDR.c_assoc_year, ZZZ_NONKIN_BIOG_ADDR.c_source_chn, ZZZ_NONKIN_BIOG_ADDR.c_source_title, " + _
    "ZZZ_NONKIN_BIOG_ADDR.c_pages, ZZZ_NONKIN_BIOG_ADDR.c_notes " + _
    "FROM ZZZ_NONKIN_BIOG_ADDR " + _
    "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_personid)=" + tStrPersonID + "))"
Set tRst = CurrentDb.OpenRecordset(tQueryStr)

MsgBox "Writing association data"
If Not (tRst.EOF) Then
    tRst.MoveLast
    tStr = "<P><B><I>Associations</I></B> "
    If tRst.RecordCount = 1 Then
        tStr = tStr + "(1 record)"
    Else
        tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
    End If
    gStream.WriteText tStr + "</P>", adWriteLine
    '
    ' c_personid, c_node_id, c_node_name, c_node_chn , c_node_female,
    ' c_link_chn, c_link_desc, c_assoc_year, c_link_count,
    ' c_text_title,
    ' c_node_index_year, c_node_dynasty, c_node_dynasty_chn,
    ' c_node_addr_name, c_node_addr_chn, node_xcoord, node_ycoord,
    ' c_lit_genre_desc, c_lit_genre_desc_chn, c_occasion_desc, c_occasion_desc_chn, c_topic_desc, c_topic_desc_chn,
    ' c_inst_name_hz, c_inst_name_py,
    ' c_source_chn, c_source_title,

```

```
' c_pages, c_notes
```

```
With tRst
```

```
    .MoveFirst
```

```
    Do While Not .EOF
```

```
        tStr = "<P>" + !c_node_name + " " + !c_node_chn + IIf(!c_node_female, " (F)", " (M)") + "<BR>"
        gStream.WriteText tStr, adWriteLine
```

```
        tStr = !c_link_desc + " " + !c_link_chn
```

```
        If Not IsNull(!c_assoc_year) Then
```

```
            tStr = tStr + "(Year: " + Trim(Str(!c_assoc_year)) + ")"
```

```
        End If
```

```
        If Not IsNull(!c_link_count) Then
```

```
            tStr = tStr + "[Count: " + Trim(Str(!c_link_count)) + "]"
```

```
        End If
```

```
        gStream.WriteText tStr + "<BR>", adWriteLine
```

```
        If Not IsNull(!c_text_title) Then
```

```
            tStr = "Title: " + Trim(!c_text_title) + "<BR>"
```

```
            gStream.WriteText tStr, adWriteLine
```

```
        End If
```

```
        tStr = "<I>Personal Information</I><BR>"
```

```
        gStream.WriteText tStr, adWriteLine
```

```
        If Not IsNull(!c_node_index_year) Then
```

```
            tStr = "Index Year: " + Trim(Str(!c_node_index_year)) + "<BR>"
```

```
            gStream.WriteText tStr, adWriteLine
```

```
        End If
```

```
        If Not IsNull(!c_node_dynasty) Then
```

```
            tStr = "Dynasty: " + Trim(!c_node_dynasty) + Trim(!c_node_dynasty_chn) + "<BR>"
```

```
            gStream.WriteText tStr, adWriteLine
```

```
        End If
```

```
        If Not IsNull(!c_node_addr_name) Then
```

```
            tStr = "Index Address: " + Trim(!c_node_addr_name) + Trim(!c_node_addr_chn)
```

```
            If Not IsNull(!node_xcoord) Then
```

```
                tStr = tStr + "(" + Trim(Str(!node_xcoord)) + tC + Trim(Str(!node_ycoord)) + ")"
```

```
            End If
```

```
            gStream.WriteText tStr + "<BR>", adWriteLine
```

```
        End If
```

```
        If Not IsNull(!c_lit_genre_desc) Then
```

```
            tStr = "Literary Genre: " + Trim(!c_lit_genre_desc) + Trim(!c_lit_genre_desc_chn) + "<BR>"
```

```
            gStream.WriteText tStr, adWriteLine
```

```
        End If
```

```
        If Not IsNull(!c_occasion_desc) Then
```

```
            tStr = "Occasion: " + Trim(!c_occasion_desc) + Trim(!c_occasion_desc_chn) + "<BR>"
```

```
            gStream.WriteText tStr, adWriteLine
```

```
        End If
```

```
        If Not IsNull(!c_topic_desc) Then
```

```
            tStr = "Topic: " + Trim(!c_topic_desc) + Trim(!c_topic_desc_chn) + "<BR>"
```

```
            gStream.WriteText tStr, adWriteLine
```

```
        End If
```

```
        If Not IsNull(!c_inst_name_py) Then
```

```
            tStr = "Institution: " + Trim(!c_inst_name_py) + Trim(!c_inst_name_hz) + "<BR>"
```

```
            gStream.WriteText tStr, adWriteLine
```

```
        End If
```

```
        If Not IsNull(!c_source_title) Then
```

```
            tStr = "Source: " + Trim(!c_source_title) + " " + Trim(!c_source_chn)
```

```
            If Not IsNull(!c_pages) Then
```

```
                tStr = tStr + " (" + Trim(!c_pages) + ")"
```

```
            End If
```

```
            tStr = tStr + "<BR>"
```

```
            gStream.WriteText tStr, adWriteLine
```

```
        End If
```

```
        If Not IsNull(!c_notes) Then
```

```
            tStr = "Notes: " + Trim(!c_notes) + "<BR>"
```

```
            gStream.WriteText tStr, adWriteLine
```

```
        End If
```

```
        .MoveNext
```

```
        gStream.WriteText "</P>", adWriteLine
```

```
    Loop
```

```
End With
```

```

9
'
End If
' now Entry
'
tQueryStr = "SELECT ZZZ_ENTRY_DATA.c_personid, ZZZ_ENTRY_DATA.c_entry_desc, ZZZ_ENTRY_DATA.c_entry_desc_chn,
" +
"ZZZ_ENTRY_DATA.c_sequence, ZZZ_ENTRY_DATA.c_exam_rank, ZZZ_ENTRY_DATA.c_kinrel_chn, ZZZ_ENTRY_DATA.c_kinrel, " +
"ZZZ_ENTRY_DATA.c_kin_name, ZZZ_ENTRY_DATA.c_kin_name_chn, ZZZ_ENTRY_DATA.c_assoc_desc, ZZZ_ENTRY_DATA.c_assoc_desc_chn, " +
"ZZZ_ENTRY_DATA.c_assoc_name, ZZZ_ENTRY_DATA.c_assoc_name_chn, ZZZ_ENTRY_DATA.c_year, ZZZ_ENTRY_DATA.c_inst_name_hz, " +
"ZZZ_ENTRY_DATA.c_inst_name_py, ZZZ_ENTRY_DATA.c_title_chn, ZZZ_ENTRY_DATA.c_title, ZZZ_ENTRY_DATA.c_pages, " +
"ZZZ_ENTRY_DATA.c_notes, ZZZ_ENTRY_DATA.c_age, ZZZ_ENTRY_DATA.c_parental_status_desc, ZZZ_ENTRY_DATA.c_parental_status_desc_chn, " +
"ZZZ_ENTRY_DATA.c_entry_addr_name, ZZZ_ENTRY_DATA.c_entry_addr_chn, ZZZ_ENTRY_DATA.c_entry_xcoord, ZZZ_ENTRY_DATA.c_entry_ycoord " +
"FROM ZZZ_ENTRY_DATA " +
"WHERE ((ZZZ_ENTRY_DATA.c_personid)=" + tStrPersonID + ")")
Set tRst = CurrentDb.OpenRecordset(tQueryStr)

MsgBox "Writing entry data"
If Not (tRst.EOF) Then
'
' !c_entry_desc, !c_entry_desc_chn, !c_year, !c_sequence, !c_age, !c_exam_rank,
' !c_parental_status_desc, !c_parental_status_desc_chn,
' !c_entry_addr_name, !c_entry_addr_chn, !c_entry_xcoord, !c_entry_ycoord
' !c_kinrel_chn, !c_kinrel, !c_kin_name, !c_kin_name_chn,
' !c_assoc_desc, !c_assoc_desc_chn, !c_assoc_name, !c_assoc_name_chn,
' !c_inst_name_hz, !c_inst_name_py,
' !c_title_chn, !c_title, !c_pages, !c_notes,
'
tRst.MoveLast
tStr = "<P><B><I>Entry into Government</I></B> "
If tRst.RecordCount = 1 Then
tStr = tStr + "(1 record)"
Else
tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
End If
gStream.WriteText tStr + "</P>", adWriteLine
'
With tRst
.MoveFirst
Do While Not .EOF
'
tStr = "<P><I>Entry</I>: " + Trim(!c_entry_desc) + " " + Trim(!c_entry_desc_chn) + "<BR>"
gStream.WriteText tStr, adWriteLine
'
If Not IsNull(!c_year) Then
tStr = "Year: " + Trim(Str(!c_year)) + "<BR>"
gStream.WriteText tStr, adWriteLine
End If
'
If Not IsNull(!c_sequence) Then
tStr = "Sequence: " + Trim(Str(!c_sequence)) + "<BR>"
gStream.WriteText tStr, adWriteLine
End If
'
If Not IsNull(!c_age) Then
tStr = "Age: " + Trim(Str(!c_age)) + "<BR>"
gStream.WriteText tStr, adWriteLine
End If
'
If Not IsNull(!c_exam_rank) Then
tStr = "Exam Rank: " + Trim(Str(!c_exam_rank)) + "<BR>"
gStream.WriteText tStr, adWriteLine
End If
'
If Not IsNull(!c_parental_status_desc) Then
tStr = "Parental Status: " + Trim(Str(!c_parental_status_desc)) + " " + Trim(!c_parental_status_desc_chn) + "<BR>"
gStream.WriteText tStr, adWriteLine
End If
'
If Not IsNull(!c_entry_addr_name) Then
tStr = "Location: " + Trim(Str(!c_entry_addr_name)) + " " + Trim(!c_entry_addr_chn)
If Not IsNull(!c_entry_xcoord) Then
tStr = tStr + " (" + Trim(Str(!c_entry_xcoord)) + " " + Trim(Str(!c_entry_ycoord)) + ")"

```

```

        End If
        gStream.WriteText tStr + "<BR>", adWriteLine
    End If
    '
    If Not IsNull(!c_kinrel) Then
        tStr = "Kinship Relation: " + Trim(!c_kinrel) + " " + Trim(!c_kinrel_chn)
        gStream.WriteText tStr + "<BR>", adWriteLine
        If Not IsNull(!c_kin_name) Then
            tStr = "Kin Name: " + Trim(!c_kin_name) + " " + Trim(!c_kin_name_chn)
            gStream.WriteText tStr + "<BR>", adWriteLine
        End If
    End If
    '
    If Not IsNull(!c_assoc_desc) Then
        tStr = "Association: " + Trim(!c_assoc_desc) + " " + Trim(!c_assoc_desc_chn)
        gStream.WriteText tStr + "<BR>", adWriteLine
        If Not IsNull(!c_kin_name) Then
            tStr = "Associate Name: " + Trim(!c_assoc_name) + " " + Trim(!c_assoc_name_chn)
            gStream.WriteText tStr + "<BR>", adWriteLine
        End If
    End If
    '
    If Not IsNull(!c_inst_name_py) Then
        tStr = "Institution: " + Trim(!c_inst_name_py) + " " + Trim(!c_inst_name_hz) + "<BR>"
        gStream.WriteText tStr, adWriteLine
    End If
    '
    If Not IsNull(!c_title) Then
        tStr = "Source: " + Trim(!c_title) + " " + Trim(!c_title_chn)
        If Not IsNull(!c_pages) Then
            tStr = tStr + " (" + Trim(!c_pages) + ")"
        End If
        gStream.WriteText tStr + "<BR>", adWriteLine
    End If
    '
    If Not IsNull(!c_notes) Then
        tStr = "Notes: " + Trim(!c_notes) + "<BR>"
        gStream.WriteText tStr, adWriteLine
    End If
    '
    gStream.WriteText "</P>", adWriteLine
    '
    .MoveNext
Loop
End With
End If
'
' now Office
'
tQueryStr = "SELECT ZZZ_POSTED_TO_ADDR_DATA.c_personid, ZZZ_POSTED_TO_ADDR_DATA.c_office_pinyin, ZZZ_POSTED_TO_ADDR_DATA.c_office_chn, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_office_trans, ZZZ_POSTED_TO_ADDR_DATA.c_sequence, ZZZ_POSTED_TO_ADDR_DATA.c_firstyear, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_lastyear, ZZZ_POSTED_TO_ADDR_DATA.c_appt_desc_chn, ZZZ_POSTED_TO_ADDR_DATA.c_appt_desc, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_assume_office_desc_chn, ZZZ_POSTED_TO_ADDR_DATA.c_assume_office_desc, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_inst_name_py, ZZZ_POSTED_TO_ADDR_DATA.c_inst_name_hz, ZZZ_POSTED_TO_ADDR_DATA.c_title, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_title_chn, ZZZ_POSTED_TO_ADDR_DATA.c_pages, ZZZ_POSTED_TO_ADDR_DATA.c_notes, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_dynasty, ZZZ_POSTED_TO_ADDR_DATA.c_dynasty_chn, ZZZ_POSTED_TO_ADDR_DATA.c_category_desc, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_category_desc_chn, ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_name, ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_chn, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.office_x_coord, ZZZ_POSTED_TO_ADDR_DATA.office_y_coord " + _
    "FROM ZZZ_POSTED_TO_ADDR_DATA " + _
    "WHERE ((ZZZ_POSTED_TO_ADDR_DATA.c_personid)=" + tStrPersonID + ")"
Set tRst = CurrentDb.OpenRecordset(tQueryStr)

MsgBox "Writing postings data"
If Not (tRst.EOF) Then
    tRst.MoveLast
    tStr = "<P><B><I>Office Appointments</I></B> "
    If tRst.RecordCount = 1 Then
        tStr = tStr + "(1 record)"
    Else
        tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
    End If
    gStream.WriteText tStr + "</P>", adWriteLine

```

```

'
With tRst
.MoveFirst
'
' !c_personid, !c_office_pinyin, !c_office_chn, !c_office_trans
' !c_office_addr_name, !c_office_addr_chn
' !office_x_coord, !office_y_coord
' !c_sequence, !c_dynasty, !c_dynasty_chn, !c_firstyear, !c_lastyear
' !c_category_desc, !c_category_desc_chn, !c_appt_desc_chn, !c_appt_desc
' !c_assume_office_desc_chn, !c_assume_office_desc
' !c_inst_name_py, !c_inst_name_hz
' !c_title, !c_title_chn, !c_pages, !c_notes

Do While Not .EOF
'   the ID of the person
tStr = "<P><I>Appointment</I>: " + Trim(!c_office_pinyin) + " " + Trim(!c_office_chn)
If Not IsNull(!c_office_trans) Then
    tStr = tStr + " " + Trim(!c_office_trans)
End If
gStream.WriteText tStr + "<BR>", adWriteLine
'

If Not IsNull(!c_office_addr_name) Then
    tStr = "Location: " + Trim(!c_office_addr_name) + " " + Trim(!c_office_addr_chn)
    If Not IsNull(!office_x_coord) Then
        tStr = tStr + " (" + Trim(Str(!office_x_coord)) + tC + Trim(Str(!office_y_coord)) + ")"
    End If
    gStream.WriteText tStr + "<BR>", adWriteLine
End If
'

If Not IsNull(!c_sequence) Then
    tStr = "Sequence: " + Trim(Str(!c_sequence)) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
'

If Not IsNull(!c_dynasty) Then
    tStr = "Dynasty: " + Trim(!c_dynasty) + " " + Trim(!c_dynasty_chn) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
'

If Not IsNull(!c_firstyear) Then
    tStr = "First Year: " + Trim(Str(!c_firstyear)) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
'

If Not IsNull(!c_lastyear) Then
    tStr = "Last Year: " + Trim(Str(!c_lastyear)) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
'

If Not IsNull(!c_category_desc) Then
    tStr = "Category: " + Trim(!c_category_desc) + " " + Trim(!c_category_desc_chn) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
'

If Not IsNull(!c_appt_desc) Then
    tStr = "Appointment Type: " + Trim(!c_appt_desc) + " " + Trim(!c_appt_desc_chn) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
'

If Not IsNull(!c_assume_office_desc) Then
    tStr = "Assuming Office: " + Trim(!c_assume_office_desc) + " " + Trim(!c_assume_office_desc_chn) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
'

If Not IsNull(!c_inst_name_py) Then
    tStr = "Institution: " + Trim(!c_inst_name_py) + " " + Trim(!c_inst_name_hz) + "<BR>"
    gStream.WriteText tStr, adWriteLine
End If
'

If Not IsNull(!c_title) Then
    tStr = "Source: " + Trim(!c_title) + " " + Trim(!c_title_chn)
    If Not IsNull(!c_pages) Then
        tStr = tStr + " (" + Trim(!c_pages) + ")"
    End If
    gStream.WriteText tStr + "<BR>", adWriteLine
End If
'

If Not IsNull(!c_notes) Then
    tStr = "Notes: " + Trim(!c_notes) + "<BR>"

```

```

        gStream.WriteText tStr, adWriteLine
    End If
    '
    gStream.WriteText "</P>", adWriteLine
    '
    .MoveNext
Loop
End With
End If
'
' now Status
'
tQueryStr = "SELECT ZZZ_STATUS_DATA.c_personid, ZZZ_STATUS_DATA.c_status_desc, ZZZ_STATUS_DATA.c_status_desc_chn, " + _
    "ZZZ_STATUS_DATA.c_firstyear, ZZZ_STATUS_DATA.c_lastyear, ZZZ_STATUS_DATA.c_source, ZZZ_STATUS_DATA.c_title_chn, " + _
    "ZZZ_STATUS_DATA.c_title, ZZZ_STATUS_DATA.c_pages, ZZZ_STATUS_DATA.c_notes " + _
    "FROM ZZZ_STATUS_DATA " + _
    "WHERE (((ZZZ_STATUS_DATA.c_personid)=" + tStrPersonID + "))"
Set tRst = CurrentDb.OpenRecordset(tQueryStr)

MsgBox "Writing status data"
If Not (tRst.EOF) Then
    '
    tRst.MoveLast
    tStr = "<P><B><I>Social Distinction</I></B> "
    If tRst.RecordCount = 1 Then
        tStr = tStr + "(1 record)"
    Else
        tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
    End If
    gStream.WriteText tStr + "</P>", adWriteLine
    '
    ' !c_personid, !c_status_desc, !c_status_desc_chn,
    ' !c_firstyear, !c_lastyear, !c_source, !c_title_chn, !c_title, !c_pages, !c_notes
    '
    With tRst
        .MoveFirst
        Do While Not .EOF
            '
            tStr = "<P><I>Status</I>: " + Trim(!c_status_desc) + " " + Trim(!c_status_desc_chn) + "<BR>"
            gStream.WriteText tStr, adWriteLine
            '
            If Not IsNull(!c_firstyear) Then
                tStr = "First Year: " + Trim(Str(!c_firstyear)) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If
            '
            If Not IsNull(!c_lastyear) Then
                tStr = "Last Year: " + Trim(Str(!c_lastyear)) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If
            '
            If Not IsNull(!c_title) Then
                tStr = "Source: " + Trim(!c_title) + " " + Trim(!c_title_chn)
                If Not IsNull(!c_pages) Then
                    tStr = tStr + " (" + Trim(!c_pages) + ") "
                End If
                gStream.WriteText tStr + "<BR>", adWriteLine
            End If
            '
            If Not IsNull(!c_notes) Then
                tStr = "Notes: " + Trim(!c_notes) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If
            '
            gStream.WriteText "</P>", adWriteLine
            '
            .MoveNext
        Loop
    End With
End If
'
' now Organizations
'
tQueryStr = "SELECT ZZZ_BIOG_INST_DATA.c_personid, ZZZ_BIOG_INST_DATA.c_inst_name_hz, ZZZ_BIOG_INST_DATA.c_inst_name_py, " + _
    "ZZZ_BIOG_INST_DATA.c_inst_type_hz, ZZZ_BIOG_INST_DATA.c_inst_addr_pinyin, ZZZ_BIOG_INST_DATA.c_inst_addr_chn, " + _
    "ZZZ_BIOG_INST_DATA.c_inst_addr_type_desc, ZZZ_BIOG_INST_DATA.c_inst_addr_type_chn, ZZZ_BIOG_INST_DATA"

```

```

A.c_bi_role_desc, " +
    "ZZZ_BIOG_INST_DATA.c_bi_role_chn, ZZZ_BIOG_INST_DATA.c_bi_begin_year, ZZZ_BIOG_INST_DATA.c_bi_end_year, " +
    "ZZZ_BIOG_INST_DATA.c_source_chn, ZZZ_BIOG_INST_DATA.c_source_py, ZZZ_BIOG_INST_DATA.c_pages, ZZZ_BIOG_INST_DATA.c_notes, " +
    "ZZZ_BIOG_INST_DATA.inst_xcoord, ZZZ_BIOG_INST_DATA.inst_ycoord " +
    "FROM ZZZ_BIOG_INST_DATA " +
    "WHERE ((ZZZ_BIOG_INST_DATA.c_personid)=" + tStrPersonID + ")")
Set tRst = CurrentDb.OpenRecordset(tQueryStr)

MsgBox "Writing institution data"
If Not (tRst.EOF) Then
    tRst.MoveLast
    tStr = "<P><B><I>Institutions</I></B> "
    If tRst.RecordCount = 1 Then
        tStr = tStr + "(1 record)"
    Else
        tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
    End If
    gStream.WriteText tStr + "</P>", adWriteLine

    ' !c_personid, !c_inst_name_hz, !c_inst_name_py,
    ' !c_inst_type_hz, !c_inst_addr_pinyin, !c_inst_addr_chn,
    ' !c_inst_addr_type_desc, !c_inst_addr_type_chn, !c_bi_role_desc,
    ' !c_bi_role_chn, !c_bi_begin_year, !c_bi_end_year,
    ' !c_source_chn, !c_source_py, !c_pages, !c_notes, !inst_xcoord, !inst_ycoord

    With tRst
        .MoveFirst
        Do While Not .EOF
            tStr = "<P><I>Institution</I>: " + Trim(!c_inst_name_py) + " " + Trim(!c_inst_name_hz) + "<BR>"
            gStream.WriteText tStr, adWriteLine

            If Not IsNull(!c_inst_type_hz) Then
                tStr = "Type: " + Trim(!c_inst_type_hz) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If

            If Not IsNull(!c_inst_addr_pinyin) Then
                tStr = "Location: " + Trim(!c_inst_addr_pinyin) + " " + Trim(!c_inst_addr_chn)
                If Not IsNull(!inst_xcoord) Then
                    tStr = tStr + "(" + Trim(Str(!inst_xcoord)) + tC + Trim(Str(!inst_ycoord)) + ")"
                End If
                If Not IsNull(!c_inst_addr_type_desc) Then
                    gStream.WriteText tStr + "<BR>", adWriteLine
                    tStr = "Type: " + !c_inst_addr_type_desc + " " + !c_inst_addr_type_chn
                End If
                gStream.WriteText tStr + "<BR>", adWriteLine
            End If

            tStr = "Role: " + Trim(!c_bi_role_desc) + " " + Trim(!c_bi_role_chn) + "<BR>"
            gStream.WriteText tStr, adWriteLine

            If Not IsNull(!c_bi_begin_year) Then
                tStr = "First Year: " + Trim(Str(!c_bi_begin_year)) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If

            If Not IsNull(!c_bi_end_year) Then
                tStr = "Last Year: " + Trim(Str(!c_bi_end_year)) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If

            If Not IsNull(!c_source_py) Then
                tStr = "Source: " + Trim(!c_source_py) + " " + Trim(!c_source_chn)
                If Not IsNull(!c_pages) Then
                    tStr = tStr + "(" + Trim(!c_pages) + ")"
                End If
                gStream.WriteText tStr + "<BR>", adWriteLine
            End If

            If Not IsNull(!c_notes) Then
                tStr = "Notes: " + Trim(!c_notes) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If

            gStream.WriteText "</P>", adWriteLine
        End While
    End With
End If

```

```

14
        .MoveNext
    Loop
End With
End If
'
' now Texts
'
tQueryStr = "SELECT ZZZ_BIOG_TEXT_DATA.c_title, ZZZ_BIOG_TEXT_DATA.c_title_chn, ZZZ_BIOG_TEXT_DATA.c_bibl_cat_desc, " + _
"ZZZ_BIOG_TEXT_DATA.c_bibl_cat_desc_chn, ZZZ_BIOG_TEXT_DATA.c_role_desc, ZZZ_BIOG_TEXT_DATA.c_role_desc_chn, " + _
"ZZZ_BIOG_TEXT_DATA.c_source_title, ZZZ_BIOG_TEXT_DATA.c_source_chn, ZZZ_BIOG_TEXT_DATA.c_pages, ZZZ_BIOG_TEXT_DATA.c_notes, " + _
"ZZZ_BIOG_TEXT_DATA.c_personid " + _
"FROM ZZZ_BIOG_TEXT_DATA " + _
"WHERE (((ZZZ_BIOG_TEXT_DATA.c_personid)=" + tStrPersonID + "))"
Set tRst = CurrentDb.OpenRecordset(tQueryStr)

MsgBox "Writing text data"
If Not (tRst.EOF) Then
    '
    tRst.MoveLast
    tStr = "<P><B><I>Texts</I></B> "
    If tRst.RecordCount = 1 Then
        tStr = tStr + "(1 record)"
    Else
        tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
    End If
    gStream.WriteText tStr + "</P>", adWriteLine
    '
    ' !c_title, !c_title_chn, !c_bibl_cat_desc, !c_bibl_cat_desc_chn,
    ' !c_role_desc, !c_role_desc_chn,
    ' !c_source_title, !c_source_chn, !c_pages, !c_notes,

    With tRst
        .MoveFirst
        Do While Not .EOF
            '
            tStr = "<P><I>Text</I>: "
            If Not IsNull(!c_title) Then
                tStr = tStr + Trim(!c_title) + " "
            End If
            If IsNull(!c_title_chn) Then
                tStr = tStr + "Title Missing"
            Else
                tStr = tStr + Trim(!c_title_chn)
            End If
            tStr = tStr + "<BR>"
            gStream.WriteText tStr, adWriteLine
            '
            If Not IsNull(!c_bibl_cat_desc) Then
                tStr = "Category: " + Trim(!c_bibl_cat_desc) + " " + Trim(!c_bibl_cat_desc_chn) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If
            '
            If Not IsNull(!c_role_desc) Then
                tStr = "Role: " + Trim(!c_role_desc) + " " + Trim(!c_role_desc_chn) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If
            '
            If Not IsNull(!c_source_title) Then
                tStr = "Source: " + Trim(!c_source_title) + " " + Trim(!c_source_chn)
                If Not IsNull(!c_pages) Then
                    tStr = tStr + " (" + Trim(!c_pages) + ")"
                End If
                gStream.WriteText tStr + "<BR>", adWriteLine
            End If
            '
            If Not IsNull(!c_notes) Then
                tStr = "Notes: " + Trim(!c_notes) + "<BR>"
                gStream.WriteText tStr, adWriteLine
            End If
            '
            gStream.WriteText "</P>", adWriteLine
            '
            .MoveNext
        Loop
    End With
End If
'

```



```

' now Sources
'
tQueryStr = "SELECT ZZZ_BIOG_SOURCE_DATA.c_personid, ZZZ_BIOG_SOURCE_DATA.c_title_chn, ZZZ_BIOG_SOURCE_DATA.c_title, " +
"ZZZ_BIOG_SOURCE_DATA.c_pages, ZZZ_BIOG_SOURCE_DATA.c_notes, ZZZ_BIOG_SOURCE_DATA.c_hyperlink " +
"FROM ZZZ_BIOG_SOURCE_DATA " +
"WHERE (((ZZZ_BIOG_SOURCE_DATA.c_personid)=" + tStrPersonID + "))"
Set tRst = CurrentDb.OpenRecordset(tQueryStr)

MsgBox "Writing source data"
If Not (tRst.EOF) Then
'
tRst.MoveLast
tStr = "<P><B><I>Sources</I></B> "
If tRst.RecordCount = 1 Then
tStr = tStr + "(1 record)"
Else
tStr = tStr + "(" + Trim(Str(tRst.RecordCount)) + " records)"
End If
gStream.WriteText tStr + "</P>", adWriteLine
'
' !c_title_chn, !c_title,
' !c_pages, !c_notes, !c_hyperlink
'
With tRst
.MoveFirst
Do While Not .EOF
'
tStr = "<P><I>Source</I>: " + Trim(!c_title) + " " + Trim(!c_title_chn)
If Not IsNull(!c_pages) Then
tStr = tStr + " (" + Trim(!c_pages) + ")"
End If
gStream.WriteText tStr + "<BR>", adWriteLine
'
If Not IsNull(!c_hyperlink) Then
tStr = "Link: " + Trim(!c_hyperlink) + "<BR>"
gStream.WriteText tStr, adWriteLine
End If
'
If Not IsNull(!c_notes) Then
tStr = "Notes: " + Trim(!c_notes) + "<BR>"
gStream.WriteText tStr, adWriteLine
End If
'
gStream.WriteText "</P>", adWriteLine
'
.MoveNext
Loop
End With
End If
'
' Set the object variable to Nothing.
Set dlgSaveAs = Nothing
gStream.WriteText "</BODY>", adWriteLine
gStream.WriteText "</HTML>", adWriteLine
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
' MsgBox "Finished"
MsgBox "Finished saving to File"

Exit_CmdSaveToFile_Click:
Exit Sub

Err_CmdSaveToFile_Click:
MsgBox Err.Description
Resume Exit_CmdSaveToFile_Click

End Sub

Private Sub CmdSearchByOffice_Click()
Dim stDocName As String, stLinkCriteria As String
Dim tRstSearch As DAO.Recordset

stDocName = "frmSearchPeopleOffice"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog

```

```

If CurrentProject.AllForms("frmSearchPeopleOffice").IsLoaded Then
    Set tRstSearch = CurrentDb.OpenRecordset("Z_NAME_SEARCH", dbOpenDynaset)
    If tRstSearch.RecordCount > 0 Then
        Me.CmdClearSearch.Enabled = True
        Set Me.frmPeopleLookup2.Form.Recordset = tRstSearch
    End If

    DoCmd.Close acForm, stDocName
End If

End Sub

Private Sub CmdStoreID_Click()
    Dim cmdSQL As ADODB.Command, tRecCount As Variant, tRst As DAO.Recordset, tID As Long

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
        ' Display message.
        If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) = vbNo Then
            Exit Sub
        Else
            cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
            cmdSQL.Execute tRecCount
        End If
    End If

    ' get the value
    Set tRst = Me.frmPeopleLookup2.Form.Recordset
    tID = tRst!c_personid

    cmdSQL.CommandText = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT " + Str(tID) + " AS c_personid"
    cmdSQL.Execute tRecCount
    MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."
End Sub

Private Sub Form_Open(Cancel As Integer)
    BIOG_MAIN_2_Subform.Form.OrderBy = "c_personid"
    BIOG_MAIN_2_Subform.Form.OrderByOn = True
    frmPeopleLookup2.Form.OrderBy = "c_name"
    frmPeopleLookup2.Form.OrderByOn = True

    ' set the language
    Dim tmli As MsoLanguageID
    ' get the labels
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    gLabelsOK = True
    If tmli = msoLanguageIDSimplifiedChinese Then
        gDisplayLanguage = "S"
    ElseIf tmli = msoLanguageIDTraditionalChinese Then
        gDisplayLanguage = "T"
    ElseIf tmli = msoLanguageIDEnglishUS Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "E"
    End If
    Call changeDisplayLanguage

    gPersonID = 0
End Sub

Private Sub CmdSearch_Click()
On Error GoTo Err_CmdSearch_Click

    Dim tRstSearch As DAO.Recordset, tStr As String, tQt As String, tQuery As QueryDef, tStrName As String
    Dim cmdSQL As ADODB.Command, tRecNum As Long

    ' the logic of search is to use the characters first, then the pinyin
    ' the search first looks at ZZZ_BIOG_MAIN's c_name and c_name_chn
    ' it then looks at c_name_proper and c_name_rm in ZZZ_BIOG_MAIN
    ' then it looks at ZZZ_ALTNAMES

    tQt = Chr(34)

    ' first make sure that the browser recordset is a dummy

```

```

Set tRstSearch = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_NAME_SEARCH", dbOpenDynaset)
Set Me.frmPeopleLookup2.Form.Recordset = tRstSearch

' Now zap Z_NAME_SEARCH

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from Z_NAME_SEARCH"
cmdSQL.Execute tRecNum

' now populate from ZZZ_NAMES

If IsNull(TxtNameChn.Value) Then
    If IsNull(TxtName.Value) Then
        tStr = "Quit"
    Else
        If Me.TxtName.Value = "" Then
            tStr = "Quit"
        Else
            tStrName = TxtName.Value
            If Left(tStrName, 1) = "!" Then
                tStrName = Mid(TxtName.Value, 2)
                tStr = " Left(c_name," + Str(Len(tStrName)) + ") = " + tQt + Trim(tStrName) + tQt
            ElseIf UCase(Left(tStrName, 1)) = Left(tStrName, 1) Then
                tStr = " Left(c_name," + Str(Len(tStrName)) + ") = " + tQt + Trim(tStrName) + tQt + _
                    " OR c_name LIKE " + tQt + "%" + " " + Trim(tStrName) + "%" + tQt
            Else
                tStr = " c_name LIKE " + tQt + "%" + Trim(tStrName) + "%" + tQt
            End If
        End If
    End If
Else
    If Me.TxtNameChn.Value = "" Then
        If IsNull(TxtName.Value) Then
            tStr = "Quit"
        Else
            If Me.TxtName.Value = "" Then
                tStr = "Quit"
            Else
                tStrName = TxtName.Value
                If Left(tStrName, 1) = "!" Then
                    tStrName = Mid(TxtName.Value, 2)
                    tStr = " Left(c_name," + Str(Len(tStrName)) + ") = " + tQt + Trim(tStrName) + tQt
                ElseIf UCase(Left(tStrName, 1)) = Left(tStrName, 1) Then
                    tStr = " Left(c_name," + Str(Len(tStrName)) + ") = " + tQt + Trim(tStrName) + tQt + _
                        " OR c_name LIKE " + tQt + "%" + " " + Trim(tStrName) + "%" + tQt
                Else
                    tStr = " c_name LIKE " + tQt + "%" + Trim(tStrName) + "%" + tQt
                End If
            End If
        End If
    Else
        tStr = " c_name_chn LIKE " + tQt + "%" + Trim(TxtNameChn.Value) + "%" + tQt
    End If
End If

If Not (tStr = "Quit") Then
    tStr = "INSERT INTO Z_NAME_SEARCH SELECT c_personid, c_name, c_name_chn " + _
        "FROM ZZZ_NAMES WHERE" + tStr

    cmdSQL.CommandText = tStr
    cmdSQL.Execute tRecNum
End If

Set tRstSearch = CurrentDb.OpenRecordset("Z_NAME_SEARCH", dbOpenDynaset)
If tRstSearch.RecordCount = 0 Then
    Set tRstSearch = CurrentDb.OpenRecordset("ZZZ_BIOG_MAIN", dbOpenDynaset)
    Me.CmdClearSearch.Enabled = False
Else
    Me.CmdClearSearch.Enabled = True
End If
' tRstSearch.Index = "c_name"
Set Me.frmPeopleLookup2.Form.Recordset = tRstSearch

```

```

Err_CmdSearch_Click:
    MsgBox Err.Description
    Resume Exit_CmdSearch_Click

End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

    If gDisplayLanguage = "T" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "T"
    End If

    Call changeDisplayLanguage

Exit_CmdFanti_Click:
    Exit Sub

Err_CmdFanti_Click:
    MsgBox Err.Description
    Resume Exit_CmdFanti_Click

End Sub

Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

    If gDisplayLanguage = "S" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "S"
    End If

    Call changeDisplayLanguage

Exit_CmdJianti_Click:
    Exit Sub

Err_CmdJianti_Click:
    MsgBox Err.Description
    Resume Exit_CmdJianti_Click

End Sub

Private Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 8) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 8 And Not .EOF
            If !c_form = "BROWSE" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then

```

```

19
        tLang = 1
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '
    Me.CmdSearch.Caption = tLabelLanguage(tLang, 1)
    Me.CmdFanti.Caption = tLabelLanguage(tLang, 2)
    Me.CmdJianti.Caption = tLabelLanguage(tLang, 3)
    Me.CmdClearSearch.Caption = tLabelLanguage(tLang, 5)
    Me.CmdSearchByOffice.Caption = tLabelLanguage(tLang, 6)
    Me.CmdSaveToFile.Caption = tLabelLanguage(tLang, 7)

    ' now do the subform
    BIOG_MAIN_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
    BIOG_MAIN_2_Subform.Form.changeDisplayLanguage
End If

End Sub

Private Sub TxtNameChn_LostFocus()
    '
    ' clear the pinyin name
    '
    TxtName.Value = ""
End Sub

Private Sub TxtName_LostFocus()
    '
    ' clear the Chinese name
    '
    TxtNameChn.Value = ""
End Sub

Private Sub CmdClearSearch_Click()
On Error GoTo Err_CmdClearSearch_Click

    Dim tRst As DAO.Recordset, tQuery As QueryDef, tID As Long, tQueryStr As String

    Set tRst = Me.frmPeopleLookup2.Form.Recordset

    tID = tRst!c_personid

    tQueryStr = "SELECT BIOG_MAIN.c_personid, BIOG_MAIN.c_name, BIOG_MAIN.c_name_chn" + _
        " FROM BIOG_MAIN ORDER BY BIOG_MAIN.c_name"

    'Set tQuery = CurrentDb.QueryDefs("Selected PERSON_NAME_DATA Query")
    'Set tRst = tQuery.OpenRecordset(dbOpenDynaset)
    'Set tRst = CurrentDb.OpenRecordset("BIOG_MAIN", dbOpenDynaset)
    Set tRst = CurrentDb.OpenRecordset(tQueryStr)
    'tRst.Index = "c_name"
    tRst.FindFirst "c_personid = " + Trim(Str(tID))

    Set Me.frmPeopleLookup2.Form.Recordset = tRst

    Me.CmdClearSearch.Enabled = False
    Me.TxtName.Value = ""
    Me.TxtNameChn.Value = ""

Exit_CmdClearSearch_Click:
    Exit Sub

Err_CmdClearSearch_Click:
    MsgBox Err.Description
    Resume Exit_CmdClearSearch_Click

End Sub

```

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Private Sub c_by_nh_code_AfterUpdate()

Dim rst As ADODB.Recordset

Set rst = New ADODB.Recordset

If IsNull(c_by_nh_code.Value) Then

TxtBYNH.Value = ""

Else

rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _

adLockOptimistic

rst.Find "c_nianhao_id = " & c_by_nh_code.Value

TxtBYNH.Value = rst.Fields("c_nianhao_chn")

rst.Close

End If

End Sub

Private Sub c_dy_nh_code_AfterUpdate()

Dim rst As ADODB.Recordset

Set rst = New ADODB.Recordset

If IsNull(c_dy_nh_code.Value) Then

TxtDYNH.Value = ""

Else

rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _

adLockOptimistic

rst.Find "c_nianhao_id = " & c_dy_nh_code.Value

TxtDYNH.Value = rst.Fields("c_nianhao_chn")

rst.Close

End If

End Sub

Private Sub c_fl_ey_nh_code_AfterUpdate()

Dim rst As ADODB.Recordset

Set rst = New ADODB.Recordset

If IsNull(c_fl_ey_nh_code.Value) Then

TxtFlEyNH.Value = ""

Else

rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _

adLockOptimistic

rst.Find "c_nianhao_id = " & c_fl_ey_nh_code.Value

TxtFlEyNH.Value = rst.Fields("c_nianhao_chn")

rst.Close

End If

End Sub

Private Sub c_fl_ey_notes_Click()

Dim stDocName As String

Dim stLinkCriteria As String

Dim strNH As String

c_fy_nh_code.Visible = True

c_fy_nh_code.SetFocus

strNH = c_fy_nh_code.Text

stDocName = "frmPickNIAN_HAO"

DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then

Dim intNH As Integer

Dim strNH_CHN As String

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus

intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value

c_fy_nh_code.Value = intNH

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus

strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value

TxtFYNH.Value = strNH_CHN

DoCmd.Close acForm, stDocName

End If

```
If Counter > 0 Then  
TxtNameChn.SetFocus  
MsgBox ""μ"bžì°l-F" & Counter & "-lĐǾŭîÊÈ«İàí-µǺ¼oä»j;ŁÖ´ ŒJ éwİĀōýÔÚÝ"ÈèµĀĐǺİçĈc"μ"bžì-FóĐ"μ"b»]ÓĐöŦñ};Ł
```

```
éwîÄ;ÉÔÔÊ¹ÓÄÔÔÉİ ·¼µÄ²éÔf°´âo;çÔÔ¬FÓÐÓ>ä>ÐÖÄÛ `ÊÑÊ÷™Ú£¬²éÔf¬FÓÐ"µ"p."
```

```
End If
End Sub
```

```
Private Sub c_mingzi_chn_GotFocus()
If IsNull(c_surname.Value) Or c_surname = "" Then
MsgBox "Õ^İĖŸ"Ėë Xing !"
c_surname.SetFocus
Else
If IsNull(c_mingzi.Value) Or c_mingzi = "" Then
MsgBox "Õ^İĖŸ"Ėë Ming!"
c_mingzi.SetFocus
Else
If IsNull(c_surname_chn.Value) Or c_surname_chn = "" Then
MsgBox "Õ^İĖŸ"Ėë ÐÖ!"
c_surname_chn.SetFocus
End If
End If
End If
End Sub
```

```
Private Sub c_mingzi_GotFocus()
If IsNull(c_surname.Value) Or c_surname = "" Then
MsgBox "Õ^İĖŸ"Ėë Xing !"
c_surname.SetFocus
End If
End Sub
```

```
Private Sub c_surname_AfterUpdate()

Dim intLastID As Long
Dim intID As Long

If IsNull(c_surname.Value) Or c_surname = "" Then
Else
If IsNull(c_personid.Value) Then
intLastID = DMax("c_personid", "BIOG_MAIN")
TxtLastID.Value = intLastID

TxtLastID.Visible = True
TxtLastID.SetFocus
intID = TxtLastID.Value
c_personid.Value = intID + 1

c_mingzi.SetFocus
TxtLastID.Visible = False
End If
End If

End Sub
```

```
Private Sub c_surname_chn_BeforeUpdate(Cancel As Integer)
Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset
Dim strSUR As String
Dim strNM As String
Dim strSUR_Find As String
Dim strNM_Find As String
Dim Counter As Long
Dim intPerson As Long

If Not IsNull(c_mingzi_chn.Value) And Not IsNull(c_surname_chn.Value) Then
Counter = 0

strSUR = c_surname_chn.Value
strNM = c_mingzi_chn.Value
intPerson = c_personid.Value

rst.Open "BIOG_MAIN", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
Do
If rst.EOF = True Then
Exit Do
End If

If IsNull(rst!c_surname_chn.Value) Then
Else
```



```

strSUR_Find = rst!c_surname_chn.Value
End If

If IsNull(rst!c_mingzi_chn.Value) Then
Else
strNM_Find = rst!c_mingzi_chn.Value
End If

If StrComp(strSUR_Find, strSUR) = 0 And StrComp(strNM_Find, strNM) = 0 Then
If rst!c_personid = intPerson Then
'This is to exclude the current record from being counted.
rst.MoveNext
Else
Counter = Counter + 1
rst.MoveNext
End If
Else
rst.MoveNext
End If
Loop

If Counter > 0 Then
TxtNameChn.SetFocus
MsgBox ""p`pžī°l~F" & Counter & "-lĐŌĀŭíēĒ«īāí~pĀ%oā>;łŌ^' ŌJ éwīĀŌýŌÚÝ"ÈèpĀĐĀīčĀc"p`pžī~FŌĐ"p`p>]ŌĐŌŌŊ} ;ł
éwīĀ;ĒŌŌĒ°ŌĀŌŌĒī~pĀ²éŌf°'áo;čŌŌ~FŌĐŌ>ā>ĐŌĀŭ 'ĒĒĒ÷™Ūł~²éŌf~FŌĐ"p`p."
End If

End If

End Sub

Private Sub c_surname_chn_GotFocus()
If IsNull(c_surname.Value) Or c_surname = "" Then
MsgBox "Ō^īĒÝ"Èè Xing !"
c_surname.SetFocus
Else
If IsNull(c_mingzi.Value) Or c_mingzi = "" Then
MsgBox "Ō^īĒÝ"Èè Ming!"
c_mingzi.SetFocus
End If
End If

End Sub

Private Sub CmdExplainInput_Click()
On Error GoTo Err_CmdExplainInput_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "FrmExplainInput"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdExplainInput_Click:
Exit Sub

Err_CmdExplainInput_Click:
MsgBox Err.Description
Resume Exit_CmdExplainInput_Click

End Sub

Private Sub CmdPickBYNH_Click()
On Error GoTo Err_CmdPickBYNH_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strNH As String

c_by_nh_code.Visible = True
c_by_nh_code.SetFocus
strNH = c_by_nh_code.Text

stDocName = "frmPickNIAN_HAO"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
Dim intNH As Integer
Dim strNH_CHN As String

```

```

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
c_by_nh_code.Value = intNH

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
TxtBYNH.Value = strNH_CHN

DoCmd.Close acForm, stDocName
End If

CmdPickBYNH.SetFocus
c_by_nh_code.Visible = False

Exit_CmdPickBYNH_Click:
Exit Sub

Err_CmdPickBYNH_Click:
MsgBox Err.Description
Resume Exit_CmdPickBYNH_Click

End Sub

Private Sub CmdPickChoronym_Click()
On Error GoTo Err_CmdPickChoronym_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim stChoro As String
Dim intChoro As Integer
Dim strChoro As String
Dim strChoro_CHN As String
Dim rsChoro As DAO.Recordset

Forms!BIOG_MAIN!c_choronym_code.Visible = True
Forms!BIOG_MAIN!c_choronym_code.SetFocus
stChoro = Forms!BIOG_MAIN!c_choronym_code.Text

stDocName = "frmPickChoronym"
DoCmd.OpenForm stDocName, acNormal, , stLinkCriteria, , acDialog, stChoro
If CurrentProject.AllForms(stDocName).IsLoaded Then
Forms!frmPickChoronym!frmChoronyms.Form!ChoroCode.SetFocus
intChoro = Forms!frmPickChoronym!frmChoronyms.Form!ChoroCode.Value
Forms!BIOG_MAIN!c_choronym_code.Value = intChoro

Forms!frmPickChoronym!frmChoronyms.Form!c_choronym_desc.SetFocus
strChoro = Forms!frmPickChoronym!frmChoronyms.Form!c_choronym_desc.Value
Forms!BIOG_MAIN!TxtChoroDesc.Value = strChoro

Forms!frmPickChoronym!frmChoronyms.Form!c_choronym_chn.SetFocus
strChoro_CHN = Forms!frmPickChoronym!frmChoronyms.Form!c_choronym_chn.Value
Forms!BIOG_MAIN!TxtChoroDescCHN.Value = strChoro_CHN

DoCmd.Close acForm, stDocName
End If
CmdPickChoronym.SetFocus
Forms!BIOG_MAIN!c_choronym_code.Visible = False

Exit_CmdPickChoronym_Click:
Exit Sub

Err_CmdPickChoronym_Click:
MsgBox Err.Description
Resume Exit_CmdPickChoronym_Click

End Sub

Private Sub CmdPickDynasty_Click()
On Error GoTo Err_CmdPickDynasty_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strDy As String

c_dy.Visible = True
c_dy.SetFocus
strDy = c_dy.Text

```

```

stDocName = "frmPickDynasty"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strDy

If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then
    Dim intDy As Integer
    Dim strDy_Desc As String
    Dim strDy_CHN As String

    Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
    intDy = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value
    c_dy.Value = intDy

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
    strDy_Desc = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value
    TxtDynasty.Value = strDy_Desc

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
    strDy_CHN = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
    TxtDynastyCHN.Value = strDy_CHN

    DoCmd.Close acForm, stDocName
End If

CmdPickDynasty.SetFocus
c_dy.Visible = False

Exit_CmdPickDynasty_Click:
Exit Sub

Err_CmdPickDynasty_Click:
MsgBox Err.Description
Resume Exit_CmdPickDynasty_Click

End Sub

Private Sub CmdPickNIAN_HAO_Click()
On Error GoTo Err_CmdPickNIAN_HAO_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strNH As String

    c_by_nh_code.Visible = True
    c_by_nh_code.SetFocus
    strNH = c_by_nh_code.Text

    stDocName = "frmPickNIAN_HAO"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

    If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
        Dim intNH As Integer
        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
        intDy = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
        c_dy.Value = intNH
        DoCmd.Close acForm, stDocName
    End If

    CmdPickBYNH.SetFocus
    c_by_nh_code.Visible = False

Exit_CmdPickNIAN_HAO_Click:
Exit Sub

Err_CmdPickNIAN_HAO_Click:
MsgBox Err.Description
Resume Exit_CmdPickNIAN_HAO_Click

End Sub

Private Sub CmdPickDYNH_Click()
On Error GoTo Err_CmdPickDYNH_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strNH As String

    c_dy_nh_code.Visible = True
    c_dy_nh_code.SetFocus
    strNH = c_dy_nh_code.Text

```

```

stDocName = "frmPickNIAN_HAO"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
    Dim intNH As Integer
    Dim strNH_CHN As String

    Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
    intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
    c_dy_nh_code.Value = intNH

    Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
    strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
    TxtDYNH.Value = strNH_CHN

    DoCmd.Close acForm, stDocName
End If

CmdPickDYNH.SetFocus
c_dy_nh_code.Visible = False

Exit_CmdPickDYNH_Click:
Exit Sub

Err_CmdPickDYNH_Click:
MsgBox Err.Description
Resume Exit_CmdPickDYNH_Click

End Sub

Private Sub CmdPickFlEyNH_Click()
On Error GoTo Err_CmdPickFlEyNH_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    Dim strNH As String

    c_fl_ey_nh_code.Visible = True
    c_fl_ey_nh_code.SetFocus
    strNH = c_fl_ey_nh_code.Text

    stDocName = "frmPickNIAN_HAO"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

    If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
        Dim intNH As Integer
        Dim strNH_CHN As String

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
        intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
        c_fl_ey_nh_code.Value = intNH

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
        strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
        TxtFlEyNH.Value = strNH_CHN

        DoCmd.Close acForm, stDocName
    End If

CmdPickFlEyNH.SetFocus
c_fl_ey_nh_code.Visible = False

Exit_CmdPickFlEyNH_Click:
Exit Sub

Err_CmdPickFlEyNH_Click:
MsgBox Err.Description
Resume Exit_CmdPickFlEyNH_Click

End Sub

Private Sub CmdPickFlLyNH_Click()
On Error GoTo Err_CmdPickFlLyNH_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    Dim strNH As String

```

```

c_fl_ly_nh_code.Visible = True
c_fl_ly_nh_code.SetFocus
strNH = c_fl_ly_nh_code.Text

```

```

stDocName = "frmPickNIAN_HAO"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

```

```

If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
    Dim intNH As Integer
    Dim strNH_CHN As String

    Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
    intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
    c_fl_ly_nh_code.Value = intNH

    Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
    strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
    TxtFlLyNH.Value = strNH_CHN

    DoCmd.Close acForm, stDocName
End If

```

```

CmdPickFlLyNH.SetFocus
c_fl_ly_nh_code.Visible = False

```

```

Exit_CmdPickFlLyNH_Click:
Exit Sub

```

```

Err_CmdPickFlLyNH_Click:
MsgBox Err.Description
Resume Exit_CmdPickFlLyNH_Click
End Sub

```

```

Private Sub CmdPickSource_Click()
On Error GoTo Err_CmdPickSource_Click

```

```

Dim stDocName As String
Dim stLinkCriteria As String
Dim strSC As String

```

```

c_source.Visible = True
c_source.SetFocus
strSC = c_source.Text

```

```

stDocName = "frmPickTEXTS"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strSC

```

```

If CurrentProject.AllForms("frmPickTEXTS").IsLoaded Then
    Dim intSC As Long
    Dim strSC_CHN As String

    Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.SetFocus
    intSC = Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.Value
    c_source.Value = intSC

    Forms!frmPickTEXTS!frmTEXTS.Form!c_title.SetFocus
    strSC_CHN = Forms!frmPickTEXTS!frmTEXTS.Form!c_title_chn.Value
    TxtTitle_CHN.Value = strSC_CHN

    DoCmd.Close acForm, stDocName
End If

```

```

CmdPickSource.SetFocus
c_source.Visible = False

```

```

Exit_CmdPickSource_Click:
Exit Sub

```

```

Err_CmdPickSource_Click:
MsgBox Err.Description
Resume Exit_CmdPickSource_Click

End Sub

```

```

Private Sub Command190_Click()

```

End Sub

```
Private Sub Form_AfterDelConfirm(STATUS As Integer)
```

```
Dim rst As ADODB.Recordset
```

```
Set rst = New ADODB.Recordset
```

```
If STATUS = acDeleteOK Then
```

```
    rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
```

```
    rst.Find "c_flag = " & 1
```

```
    rst!c_flag = 0
```

```
    rst.Update
```

```
    rst.Close
```

```
Else
```

```
    rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
```

```
    rst.Find "c_flag = " & 1
```

```
    rst.Delete
```

```
    rst.Update
```

```
    rst.Close
```

```
End If
```

End Sub

```
Private Sub Form_BeforeUpdate(Cancel As Integer)
```

```
Dim tTest As Integer
```

```
Dim tStr1 As String, tStr2 As String, tMing As String, tMingChn As String
```

```
tTest = 0
```

```
tMing = ""
```

```
tMingChn = ""
```

```
,
```

```
' test to see that surnames are non-NULL
```

```
,
```

```
If Not IsNull(c_surname.Value) And Not IsNull(c_surname_chn.Value) Then
```

```
,
```

```
    ' then look for any changes
```

```
,
```

```
    If StrComp(c_surname.OldValue, c_surname.Value, 0) = 0 Then
```

```
    Else
```

```
        tTest = 1
```

```
    End If
```

```
,
```

```
    If StrComp(c_surname_chn.OldValue, c_surname_chn.Value, 0) = 0 Then
```

```
    Else
```

```
        tTest = 1
```

```
    End If
```

```
,
```

```
    ' personal names are allowed to be NULL, so we need to check if a name has
```

```
    ' been deleted as well as if one has been added
```

```
,
```

```
If Not IsNull(c_mingzi.Value) And Not IsNull(c_mingzi_chn.Value) Then
```

```
    tMingChn = Trim(c_mingzi_chn.Value)
```

```
    tMing = Trim(c_mingzi.Value)
```

```
,
```

```
    If IsNull(c_mingzi.OldValue) Then
```

```
        tTest = 1
```

```
    Else
```

```
        If StrComp(c_mingzi.OldValue, c_mingzi.Value, 0) = 0 Then
```

```
        Else
```

```
            tTest = 1
```

```
        End If
```

```
    End If
```

```
,
```

```
    If IsNull(c_mingzi_chn.OldValue) Then
```

```
        tTest = 1
```

```
    Else
```

```
        If StrComp(c_mingzi_chn.OldValue, c_mingzi_chn.Value, 0) = 0 Then
```

```
        Else
```

```
            tTest = 1
```

```
        End If
```

```
    End If
```

```
Else
```

```
    If IsNull(c_mingzi.Value) And IsNull(c_mingzi_chn.Value) Then
```

```
        If Not (IsNull(c_mingzi.OldValue) And IsNull(c_mingzi_chn.OldValue)) Then
```

```
            tTest = 1
```

```
        End If
```

```
    End If
```

```
End If
```

```
If tTest = 1 Then
```

```

        tStr1 = Trim(c_surname_chn.Value) + tMingChn
        tStr2 = Trim(c_surname.Value) + " " + tMing
        c_name_chn.Value = tStr1
        c_name.Value = tStr2
    End If
End If
End Sub

Private Sub Form_Current()

    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    If IsNull(c_choronym_code.Value) Then
        TxtChoroDesc.Value = ""
        TxtChoroDescCHN.Value = ""
    Else
        rst.Open "CHORONYM_CODES", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_choronym_code = " & c_choronym_code.Value

        TxtChoroDesc.Value = rst.Fields("c_choronym_desc")
        TxtChoroDescCHN.Value = rst.Fields("c_choronym_chn")
        rst.Close
    End If

    If IsNull(c_dy.Value) Then
        TxtDynasty.Value = ""
        TxtDynastyCHN.Value = ""
    Else
        rst.Open "DYNASTIES", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_dy = " & c_dy.Value

        TxtDynasty.Value = rst.Fields("c_dynasty")
        TxtDynastyCHN.Value = rst.Fields("c_dynasty_chn")
        rst.Close
    End If

    If IsNull(c_by_nh_code.Value) Then
        TxtBYNH.Value = ""
    Else
        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_nianhao_id = " & c_by_nh_code.Value

        TxtBYNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If

    If IsNull(c_dy_nh_code.Value) Then
        TxtDYNH.Value = ""
    Else
        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_nianhao_id = " & c_dy_nh_code.Value

        TxtDYNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If

    If IsNull(c_fl_ey_nh_code.Value) Then
        TxtFlEyNH.Value = ""
    Else
        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_nianhao_id = " & c_fl_ey_nh_code.Value

        TxtFlEyNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If

    If IsNull(c_fl_ly_nh_code.Value) Then
        TxtFlLyNH.Value = ""
    End If

```

```

Else
    rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_nianhao_id = " & c_fl_ly_nh_code.Value

    TxtFlLyNH.Value = rst.Fields("c_nianhao_chn")
    rst.Close
End If

If IsNull(c_source.Value) Then
    TxtTitle_CHN.Value = ""
Else
    rst.Open "TEXT_CODES", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_textid = " & c_source.Value

    TxtTitle_CHN.Value = rst.Fields("c_title_chn")
    rst.Close
End If

If IsNull(c_ethnicity_code.Value) Then
    TxtEthnicity.Value = ""
Else
    rst.Open "Ethnicity_codes", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_ethnicity_code = " & c_ethnicity_code.Value

    TxtEthnicity.Value = rst.Fields("c_ethnicity_desc_chn")
    rst.Close
End If

' look up the ID to see if it has a link to a database

rst.Open "Database_link_data", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.Find "c_person_id = " + Trim(Str(Me.c_personid.Value))

' an ADO recordset goes to the end of the file if nothing is found

If rst.EOF Then
    Me.CmdDBLink.HyperlinkAddress = ""
    Me.CmdDBLink.Enabled = False
    Me.CmdDBLink.Visible = False
Else
    Me.CmdDBLink.Enabled = True
    Me.CmdDBLink.Visible = True
    If rst!c_db_id = 1 Then
        Me.CmdDBLink.Caption = ChrW(26126) + ChrW(28165) + ChrW(27284) + ChrW(26696)
        Me.CmdDBLink.HyperlinkAddress = "http://archive.ihp.sinica.edu.tw/ttsweb/html_name/"
    ElseIf rst!c_db_id = 2 Then
        Me.CmdDBLink.Caption = ChrW(21776) + ChrW(20195) + ChrW(20154) + ChrW(29289)
        Me.CmdDBLink.HyperlinkAddress = "http://tkb.zinbun.kyoto-u.ac.jp/pers-db/" + Trim(Str(rst!c_db_sys_id
))
    ElseIf rst!c_db_id = 3 Then
        Me.CmdDBLink.Caption = ChrW(26126) + ChrW(28165) + ChrW(23142) + ChrW(22899) + ChrW(33879) + ChrW(203
16)
        Me.CmdDBLink.HyperlinkAddress = "http://digital.library.mcgill.ca/mingqing/search/details-poet.php?po
etID=" + Trim(Str(rst!c_db_sys_id)) _
        + "&showbio=1&showanth=1&showshihuaon=1&language=eng"
    End If
End If
rst.Close

End Sub

Private Sub CmdPickEthnicity_Click()
On Error GoTo Err_CmdPickEthnicity_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strETHN As String

    c_ethnicity_code.Visible = True
    c_ethnicity_code.SetFocus
    strETHN = c_ethnicity_code.Text

stDocName = "frmPickETHNICITY"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strETHN

    If CurrentProject.AllForms("frmPickETHNICITY").IsLoaded Then

```



```
Dim intETHN As Integer
Dim strETHN_CHN As String
```

```
Forms!frmPickETHNICITY!frmETHNICITY.Form!c_ethnicity_code.SetFocus
intETHN = Forms!frmPickETHNICITY!frmETHNICITY.Form!c_ethnicity_code.Value
c_ethnicity_code.Value = intETHN
```

```
Forms!frmPickETHNICITY!frmETHNICITY.Form!c_ethnicity_desc_chn.SetFocus
strETHN_CHN = Forms!frmPickETHNICITY!frmETHNICITY.Form!c_ethnicity_desc_chn.Value
TxtEthnicity.Value = strETHN_CHN
```

```
DoCmd.Close acForm, stDocName
End If
```

```
CmdPickEthnicity.SetFocus
c_ethnicity_code.Visible = False
```

```
Exit_CmdPickEthnicity_Click:
Exit Sub
```

```
Err_CmdPickEthnicity_Click:
MsgBox Err.Description
Resume Exit_CmdPickEthnicity_Click
```

```
End Sub
```

```
Private Sub CmdDelete_Click()
On Error GoTo Err_CmdDelete_Click
```

```
Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset
```

```
Dim blnRecordAdded As Boolean
```

```
If Not IsNull(c_surname_chn) Then
rst.Open "Del_Log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.AddNew
rst!c_personid = c_personid
rst!c_subform = "BIOG_MAIN"
rst!c_flag = 1
```

```
rst!c_surname_chn = c_surname_chn
rst!c_surname = c_surname
rst!c_mingzi_chn = c_mingzi_chn
rst!c_mingzi = c_mingzi_chn
rst!c_female = c_female
rst!c_birthyear = c_birthyear
rst!c_deathyear = c_deathyear
rst!c_by_nh_code = c_by_nh_code
rst!c_by_nh_year = c_by_nh_year
rst!c_by_range = c_by_range
rst!c_dy_nh_code = c_dy_nh_code
rst!c_dy_nh_year = c_dy_nh_year
rst!c_dy_range = c_dy_range
rst!c_fl_earliest_year = c_fl_earliest_year
rst!c_fl_latest_year = c_fl_latest_year
rst!c_fl_ey_nh_code = c_fl_ey_nh_code
rst!c_fl_ly_nh_code = c_fl_ly_nh_code
rst!c_fl_ey_nh_year = c_fl_ey_nh_year
rst!c_fl_ly_nh_year = c_fl_ly_nh_year
rst!c_fl_ey_notes = c_fl_ey_notes
rst!c_fl_ly_notes = c_fl_ly_notes
rst!c_choronym_code = c_choronym_code
rst!c_dy = c_dy
rst!c_index_year = c_index_year
rst!c_death_age = c_death_age
rst!c_ethnicity_code = c_ethnicity_code
rst!c_tribe = c_tribe
rst!c_surname_code = c_surname_code
rst!c_jia = c_jia
rst!c_zu = c_zu
```

```
rst!c_source = c_source
rst!c_pages = c_pages
rst!c_notes = c_notes
rst.Update
blnRecordAdded = True
rst.Close
```

```

End If

DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70

Exit_CmdDelete_Click:
Exit Sub

Err_CmdDelete_Click:
MsgBox Err.Description
Resume Exit_CmdDelete_Click

End Sub

Private Sub CmdOpenTEXTS_EnterForm_Click()
On Error GoTo Err_CmdOpenTEXTS_EnterForm_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "TEXTS EnterForm"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdOpenTEXTS_EnterForm_Click:
Exit Sub

Err_CmdOpenTEXTS_EnterForm_Click:
MsgBox Err.Description
Resume Exit_CmdOpenTEXTS_EnterForm_Click

End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

Screen.PreviousControl.SetFocus
DoCmd.DoMenuItem acFormBar, acEditMenu, 10, , acMenuVer70

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click

End Sub

Private Sub CmdAddNew_Click()
On Error GoTo Err_CmdAddNew_Click

DoCmd.GoToRecord , , acNewRec

c_surname.SetFocus

Exit_CmdAddNew_Click:
Exit Sub

Err_CmdAddNew_Click:
MsgBox Err.Description
Resume Exit_CmdAddNew_Click

End Sub

Private Sub CmdOpenEvents_Click()
On Error GoTo Err_CmdOpenEvents_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "EVENT_CODES EnterForm"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdOpenEvents_Click:
Exit Sub

Err_CmdOpenEvents_Click:
MsgBox Err.Description
Resume Exit_CmdOpenEvents_Click

End Sub

Public Sub changeDisplayLanguage()
Dim tLabellLanguage(3, 54) As String, tLang As Integer

```

```

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False
With tRstLabelList
    .MoveFirst
    ti = 1

    Do While ti < 54 And Not .EOF
        If !c_form = "BIO" Then
            gLabelsOK = True
            If ti <> !c_label_id Then
                MsgBox "Uh oh: mismatched label table"
                gLabelsOK = False
                Exit Do
            End If
            tLabelLanguage(1, ti) = !c_english
            tLabelLanguage(2, ti) = !c_fanti
            tLabelLanguage(3, ti) = !c_jianti
            ti = ti + 1
        End If
        .MoveNext
    Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If gLabelsOK Then
    If gDisplayLanguage = "E" Then
        tLang = 1
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '
    Me.LblFemale.Caption = tLabelLanguage(tLang, 1)
    Me.LblIndexYear.Caption = tLabelLanguage(tLang, 2)
    Me.LblTribe.Caption = tLabelLanguage(tLang, 3)
    ' Me.LblFullNameChn.Caption = tLabelLanguage(tLang, 4)
    ' Me.LblFullname.Caption = tLabelLanguage(tLang, 5)
    Me.LblPages.Caption = tLabelLanguage(tLang, 6)
    Me.LblNotes.Caption = tLabelLanguage(tLang, 7)
    Me.LblBirthyear.Caption = tLabelLanguage(tLang, 8)
    Me.LblBYNianhaoDate.Caption = tLabelLanguage(tLang, 9)
    Me.LblBYIntercalary.Caption = tLabelLanguage(tLang, 10)
    Me.LblBYGZ.Caption = tLabelLanguage(tLang, 11)
    Me.LblBYRange.Caption = tLabelLanguage(tLang, 12)
    Me.LblDeathyear.Caption = tLabelLanguage(tLang, 13)
    Me.LblDYNianhaoDate.Caption = tLabelLanguage(tLang, 14)
    Me.LblDYIntercalary.Caption = tLabelLanguage(tLang, 15)
    Me.LblDYGZ.Caption = tLabelLanguage(tLang, 16)
    Me.LblDYRange.Caption = tLabelLanguage(tLang, 17)
    Me.LblDeathAge.Caption = tLabelLanguage(tLang, 18)
    Me.LblDeathAgeRange.Caption = tLabelLanguage(tLang, 19)
    Me.LblFloruitFirstYear.Caption = tLabelLanguage(tLang, 20)
    Me.LblFloruitFirstNHYear.Caption = tLabelLanguage(tLang, 21)
    Me.LblFloruitFirstYearNotes.Caption = tLabelLanguage(tLang, 22)
    Me.LblFloruitLastYear.Caption = tLabelLanguage(tLang, 23)
    Me.LblFloruitLastNHYear.Caption = tLabelLanguage(tLang, 24)
    Me.LblFloruitLastYearNotes.Caption = tLabelLanguage(tLang, 25)
    ' Me.LblOpenEvents.Caption = tLabelLanguage(tLang, 26)

    Me.CmdPickDynasty.Caption = tLabelLanguage(tLang, 27)
    ' Me.CmdFind.Caption = tLabelLanguage(tLang, 28)
    Me.CmdPickChoronym.Caption = tLabelLanguage(tLang, 29)
    Me.CmdPickEthnicity.Caption = tLabelLanguage(tLang, 30)
    Me.CmdPickSource.Caption = tLabelLanguage(tLang, 31)
    Me.CmdPickBYNH.Caption = tLabelLanguage(tLang, 32)
    Me.CmdPickDYNH.Caption = tLabelLanguage(tLang, 33)
    Me.CmdPickFlEyNH.Caption = tLabelLanguage(tLang, 34)
    Me.CmdPickFlLyNH.Caption = tLabelLanguage(tLang, 35)
    ' Me.CmdDelete.Caption = tLabelLanguage(tLang, 36)
    ' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 37)

```

```

' Me.CmdJianti.Caption = tLabelLanguage(tLang, 38)
' Me.CmdFanti.Caption = tLabelLanguage(tLang, 39)
' Me.CmdExplain.Caption = tLabelLanguage(tLang, 40)
' Me.CmdOpenTEXTS_EnterForm.Caption = tLabelLanguage(tLang, 41)
' Me.CmdOpenEvents.Caption = tLabelLanguage(tLang, 42)

```

```

Me.PageBirthDeathYears.Caption = tLabelLanguage(tLang, 43)
Me.PageAddresses.Caption = tLabelLanguage(tLang, 44)
Me.PageAltNames.Caption = tLabelLanguage(tLang, 45)
Me.PageWritings.Caption = tLabelLanguage(tLang, 46)
Me.PageEntry.Caption = tLabelLanguage(tLang, 47)
Me.PageEvents.Caption = tLabelLanguage(tLang, 48)
Me.PageKinship.Caption = tLabelLanguage(tLang, 49)
Me.PageAssociations.Caption = tLabelLanguage(tLang, 50)
Me.PagePossessions.Caption = tLabelLanguage(tLang, 51)
Me.PageStatus.Caption = tLabelLanguage(tLang, 52)
Me.PagePosting.Caption = tLabelLanguage(tLang, 53)

```

```

' now for the subforms

```

```

Me.ASSOC_DATA_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.ASSOC_DATA_Subform.Form.changeDisplayLanguage

```

```

Me.BARE_AUTHORS_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.BARE_AUTHORS_Subform.Form.changeDisplayLanguage

```

```

Me.BIOG_ADDR_DATA_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.BIOG_ADDR_DATA_Subform.Form.changeDisplayLanguage

```

```

Me.ENTRY_DATA_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.ENTRY_DATA_Subform.Form.changeDisplayLanguage

```

```

Me.EVENTS_DATA_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.EVENTS_DATA_Subform.Form.changeDisplayLanguage

```

```

Me.FIX_ALTNAMES_Query_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.FIX_ALTNAMES_Query_Subform.Form.changeDisplayLanguage

```

```

Me.KIN_DATA_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.KIN_DATA_Subform.Form.changeDisplayLanguage

```

```

Me.POSSESSION_DATA_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.POSSESSION_DATA_Subform.Form.changeDisplayLanguage

```

```

Me.POST_DATA_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.POST_DATA_Subform.Form.changeDisplayLanguage

```

```

Me.STATUS_DATA_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.STATUS_DATA_Subform.Form.changeDisplayLanguage

```

```

End If

```

```

End Sub

```

Option Compare Database

```
Private Sub c_by_nh_code_AfterUpdate()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    If IsNull(c_by_nh_code.Value) Then
        TxtBYNH.Value = ""
    Else
        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
        adLockOptimistic
        rst.Find "c_nianhao_id = " & c_by_nh_code.Value

        TxtBYNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If
End Sub
```

End Sub

```
Private Sub c_dy_nh_code_AfterUpdate()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    If IsNull(c_dy_nh_code.Value) Then
        TxtDYNH.Value = ""
    Else
        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
        adLockOptimistic
        rst.Find "c_nianhao_id = " & c_dy_nh_code.Value

        TxtDYNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If
End Sub
```

End Sub

```
Private Sub c_fl_ey_nh_code_AfterUpdate()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    If IsNull(c_fl_ey_nh_code.Value) Then
        TxtFlEyNH.Value = ""
    Else
        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
        adLockOptimistic
        rst.Find "c_nianhao_id = " & c_fl_ey_nh_code.Value

        TxtFlEyNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If
End Sub
```

End Sub

```
Private Sub c_fl_ly_nh_code_AfterUpdate()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    If IsNull(c_fl_ly_nh_code.Value) Then
        TxtFlLyNH.Value = ""
    Else
        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
        adLockOptimistic
        rst.Find "c_nianhao_id = " & c_fl_ly_nh_code.Value

        TxtFlLyNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If
End Sub
```

End Sub

```
Private Sub c_mingzi_chn_AfterUpdate()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset
    Dim strSUR As String
    Dim strNM As String
    Dim strSUR_Find As String
    Dim strNM_Find As String
    Dim Counter As Integer
    Dim intPerson As Long
```



```

        TxtLastID.Visible = True
        TxtLastID.SetFocus
        intID = TxtLastID.Value
        c_personid.Value = intID + 1

        c_mingzi.SetFocus
        TxtLastID.Visible = False
    End If
End If

End Sub

Private Sub c_surname_chn_BeforeUpdate(Cancel As Integer)
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset
    Dim strSUR As String
    Dim strNM As String
    Dim strSUR_Find As String
    Dim strNM_Find As String
    Dim Counter As Integer
    Dim intPerson As Long

    If Not IsNull(c_mingzi_chn.Value) And Not IsNull(c_surname_chn.Value) Then
        Counter = 0

        strSUR = c_surname_chn.Value
        strNM = c_mingzi_chn.Value
        intPerson = c_personid.Value

        rst.Open "BIOG_MAIN", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
        Do
            If rst.EOF = True Then
                Exit Do
            End If

            If IsNull(rst!c_surname_chn.Value) Then
            Else
                strSUR_Find = rst!c_surname_chn.Value
            End If

            If IsNull(rst!c_mingzi_chn.Value) Then
            Else
                strNM_Find = rst!c_mingzi_chn.Value
            End If

            If StrComp(strSUR_Find, strSUR) = 0 And StrComp(strNM_Find, strNM) = 0 Then
                If rst!c_personid = intPerson Then
                    'This is to exclude the current record from being counted.
                    rst.MoveNext
                Else
                    Counter = Counter + 1
                    rst.MoveNext
                End If
            Else
                rst.MoveNext
            End If
        Loop

        If Counter > 0 Then
            TxtNameChn.SetFocus
            MsgBox "μ`pžī°l~F" & Counter & "-lĐŌÄüíêÈ«İàÍ~μÄ¼oä>ı;£Ō^´ ŌJ éwİÄŌŸŌÚŸ"ÈèμÄĐÄİçĂc"μ`pžī~FŌĐ"μ`p>]ŌĐŌŰŰ};£  

éwİÄ;ÉŌŌÊ°ŌÄŌÊİ°¼μÄ²éŌf°´âo;çŌŌ~FŌĐŌ>ä>ĐŌÄŰ 'ÈÑÈ÷™Ű£~²éŌf~FŌĐ"μ`p."
            End If
        End If
    End If
End Sub

Private Sub c_surname_chn_GotFocus()
    If IsNull(c_surname.Value) Or c_surname = "" Then
        MsgBox "Ō^İŸŸ"Èè Xing !"
        c_surname.SetFocus
    Else
        If IsNull(c_mingzi.Value) Or c_mingzi = "" Then
            MsgBox "Ō^İŸŸ"Èè Ming!"
            c_mingzi.SetFocus
        End If
    End If
End Sub

```

```
End If
End Sub
```

```
Private Sub CmbBYRG_AfterUpdate()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    If IsNull(CmbBYRG.Value) Then
        TxtBYRG.Value = ""
    Else
        rst.Open "year_range_codes", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_range_code = " & CmbBYRG.Value

        TxtBYRG.Value = rst.Fields("c_range_chn")
        rst.Close
    End If
End Sub
```

```
Private Sub CmbDYRG_AfterUpdate()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    If IsNull(CmbDYRG.Value) Then
        TxtDYRG.Value = ""
    Else
        rst.Open "year_range_codes", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_range_code = " & CmbDYRG.Value

        TxtDYRG.Value = rst.Fields("c_range_chn")
        rst.Close
    End If
End Sub
```

```
Private Sub CmdPickBYNH_Click()
On Error GoTo Err_CmdPickBYNH_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strNH As String

    c_by_nh_code.Visible = True
    c_by_nh_code.SetFocus
    strNH = c_by_nh_code.Text

    stDocName = "frmPickNIAN_HAO"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

    If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
        Dim intNH As Integer
        Dim strNH_CHN As String

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
        intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
        c_by_nh_code.Value = intNH

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
        strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
        TxtBYNH.Value = strNH_CHN

        DoCmd.Close acForm, stDocName
    End If

    CmdPickBYNH.SetFocus
    c_by_nh_code.Visible = False

Exit_CmdPickBYNH_Click:
Exit Sub

Err_CmdPickBYNH_Click:
MsgBox Err.Description
Resume Exit_CmdPickBYNH_Click

End Sub

Private Sub CmdPickChoronym_Click()
On Error GoTo Err_CmdPickChoronym_Click
```



```

Dim stDocName As String
Dim stLinkCriteria As String
Dim stChoro As String
Dim intChoro As Integer
Dim strChoro As String
Dim strChoro_CHN As String
Dim rsChoro As DAO.Recordset

```

```

Forms!BIOG_MAIN!c_choronym_code.Visible = True
Forms!BIOG_MAIN!c_choronym_code.SetFocus
stChoro = Forms!BIOG_MAIN!c_choronym_code.Text

```

```

stDocName = "frmPickChoronym"
DoCmd.OpenForm stDocName, acNormal, , stLinkCriteria, , acDialog, stChoro
If CurrentProject.AllForms(stDocName).IsLoaded Then
    Forms!frmPickChoronym!frmChoronys.Form!ChoroCode.SetFocus
    intChoro = Forms!frmPickChoronym!frmChoronys.Form!ChoroCode.Value
    Forms!BIOG_MAIN!c_choronym_code.Value = intChoro

    Forms!frmPickChoronym!frmChoronys.Form!c_choronym_desc.SetFocus
    strChoro = Forms!frmPickChoronym!frmChoronys.Form!c_choronym_desc.Value
    Forms!BIOG_MAIN!TxtChoroDesc.Value = strChoro

    Forms!frmPickChoronym!frmChoronys.Form!c_choronym_chn.SetFocus
    strChoro_CHN = Forms!frmPickChoronym!frmChoronys.Form!c_choronym_chn.Value
    Forms!BIOG_MAIN!TxtChoroDescCHN.Value = strChoro_CHN

    DoCmd.Close acForm, stDocName
End If
CmdPickChoronym.SetFocus
Forms!BIOG_MAIN!c_choronym_code.Visible = False

```

```
Exit_CmdPickChoronym_Click:
```

```
Exit Sub
```

```
Err_CmdPickChoronym_Click:
```

```

MsgBox Err.Description
Resume Exit_CmdPickChoronym_Click

```

```
End Sub
```

```
Private Sub CmdPickDynasty_Click()
```

```
On Error GoTo Err_CmdPickDynasty_Click
```

```

Dim stDocName As String
Dim stLinkCriteria As String
Dim strDy As String

```

```

c_dy.Visible = True
c_dy.SetFocus
strDy = c_dy.Text

```

```

stDocName = "frmPickDynasty"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strDy

```

```
If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then
```

```

Dim intDy As Integer
Dim strDy_Desc As String
Dim strDy_CHN As String

```

```

Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
intDy = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value
c_dy.Value = intDy

```

```

Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
strDy_Desc = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value
TxtDynasty.Value = strDy_Desc

```

```

Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
strDy_CHN = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
TxtDynastyCHN.Value = strDy_CHN

```

```

DoCmd.Close acForm, stDocName
End If

```

```

CmdPickDynasty.SetFocus
c_dy.Visible = False

```

```
Exit_CmdPickDynasty_Click:
```

```

Exit Sub

Err_CmdPickDynasty_Click:
MsgBox Err.Description
Resume Exit_CmdPickDynasty_Click

End Sub

Private Sub CmdPickNIAN_HAO_Click()
On Error GoTo Err_CmdPickNIAN_HAO_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strNH As String

c_by_nh_code.Visible = True
c_by_nh_code.SetFocus
strNH = c_by_nh_code.Text

stDocName = "frmPickNIAN_HAO"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
Dim intNH As Integer
Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
intDy = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
c_dy.Value = intNH
DoCmd.Close acForm, stDocName
End If

CmdPickBYNH.SetFocus
c_by_nh_code.Visible = False

Exit_CmdPickNIAN_HAO_Click:
Exit Sub

Err_CmdPickNIAN_HAO_Click:
MsgBox Err.Description
Resume Exit_CmdPickNIAN_HAO_Click

End Sub

Private Sub CmdPickDYNH_Click()
On Error GoTo Err_CmdPickDYNH_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strNH As String

c_dy_nh_code.Visible = True
c_dy_nh_code.SetFocus
strNH = c_dy_nh_code.Text

stDocName = "frmPickNIAN_HAO"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
Dim intNH As Integer
Dim strNH_CHN As String

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
c_dy_nh_code.Value = intNH

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
TxtDYNH.Value = strNH_CHN

DoCmd.Close acForm, stDocName
End If

CmdPickDYNH.SetFocus
c_dy_nh_code.Visible = False

Exit_CmdPickDYNH_Click:
Exit Sub

Err_CmdPickDYNH_Click:
MsgBox Err.Description
Resume Exit_CmdPickDYNH_Click

```

End Sub

```
Private Sub CmdPickFlEyNH_Click()
On Error GoTo Err_CmdPickFlEyNH_Click
```

```
Dim stDocName As String
Dim stLinkCriteria As String
```

```
Dim strNH As String
```

```
c_fl_ey_nh_code.Visible = True
c_fl_ey_nh_code.SetFocus
strNH = c_fl_ey_nh_code.Text
```

```
stDocName = "frmPickNIAN_HAO"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH
```

```
If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
```

```
Dim intNH As Integer
Dim strNH_CHN As String
```

```
Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
c_fl_ey_nh_code.Value = intNH
```

```
Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
TxtFlEyNH.Value = strNH_CHN
```

```
DoCmd.Close acForm, stDocName
```

```
End If
```

```
CmdPickFlEyNH.SetFocus
c_fl_ey_nh_code.Visible = False
```

```
Exit_CmdPickFlEyNH_Click:
Exit Sub
```

```
Err_CmdPickFlEyNH_Click:
MsgBox Err.Description
Resume Exit_CmdPickFlEyNH_Click
```

End Sub

```
Private Sub CmdPickFlLyNH_Click()
On Error GoTo Err_CmdPickFlLyNH_Click
```

```
Dim stDocName As String
Dim stLinkCriteria As String
```

```
Dim strNH As String
```

```
c_fl_ly_nh_code.Visible = True
c_fl_ly_nh_code.SetFocus
strNH = c_fl_ly_nh_code.Text
```

```
stDocName = "frmPickNIAN_HAO"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH
```

```
If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
```

```
Dim intNH As Integer
Dim strNH_CHN As String
```

```
Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
c_fl_ly_nh_code.Value = intNH
```

```
Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
TxtFlLyNH.Value = strNH_CHN
```

```
DoCmd.Close acForm, stDocName
```

```
End If
```

```
CmdPickFlLyNH.SetFocus
c_fl_ly_nh_code.Visible = False
```

```
Exit_CmdPickFlLyNH_Click:
Exit Sub
```

```
Err_CmdPickFllyNH_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickFllyNH_Click
End Sub
```

```
Private Sub CmdPickSource_Click()
On Error GoTo Err_CmdPickSource_Click
```

```
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strSC As String
```

```
    c_source.Visible = True
    c_source.SetFocus
    strSC = c_source.Text
```

```
    stDocName = "frmPickTEXTS"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strSC
```

```
    If CurrentProject.AllForms("frmPickTEXTS").IsLoaded Then
        Dim intSC As Integer
        Dim strSC_CHN As String

        Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.SetFocus
        intSC = Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.Value
        c_source.Value = intSC

        Forms!frmPickTEXTS!frmTEXTS.Form!c_title.SetFocus
        strSC_CHN = Forms!frmPickTEXTS!frmTEXTS.Form!c_title_chn.Value
        TxtTitle_CHN.Value = strSC_CHN
```

```
    DoCmd.Close acForm, stDocName
End If
```

```
CmdPickSource.SetFocus
c_source.Visible = False
```

```
Exit_CmdPickSource_Click:
Exit Sub
```

```
Err_CmdPickSource_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickSource_Click
```

```
End Sub
```

```
Private Sub DeathAgeApprox_AfterUpdate()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset
```

```
    If IsNull(DeathAgeApprox.Value) Then
        TxtApprox.Value = ""
    Else
        rst.Open "year_range_codes", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_range_code = " & DeathAgeApprox.Value

        TxtApprox.Value = rst.Fields("c_approx_chn")
        rst.Close
    End If
```

```
End Sub
```

```
Private Sub Form_AfterDelConfirm(STATUS As Integer)
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset
```

```
    If STATUS = acDeleteOK Then
        rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
        rst.Find "c_flag = " & 1
        rst!c_flag = 0
        rst.Update
        rst.Close
    Else
```

```
        rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
        rst.Find "c_flag = " & 1
        rst.Delete
        rst.Update
    End If
```

```

        rst.Close
    End If

End Sub

Private Sub Form_BeforeUpdate(Cancel As Integer)
    Dim tTest As Integer
    Dim tStr1 As String, tStr2 As String, tMing As String, tMingChn As String

    tTest = 0
    tMing = ""
    tMingChn = ""
    '
    ' test to see that surnames are non-NULL
    '
    If Not IsNull(c_surname.Value) And Not IsNull(c_surname_chn.Value) Then
        '
        ' then look for any changes
        '
        If StrComp(c_surname.OldValue, c_surname.Value, 0) = 0 Then
            Else
                tTest = 1
            End If
        '
        If StrComp(c_surname_chn.OldValue, c_surname_chn.Value, 0) = 0 Then
            Else
                tTest = 1
            End If
        '
        ' personal names are allowed to be NULL, so we need to check if a name has
        ' been deleted as well as if one has been added
        '
        If Not IsNull(c_mingzi.Value) And Not IsNull(c_mingzi_chn.Value) Then
            tMingChn = Trim(c_mingzi_chn.Value)
            tMing = Trim(c_mingzi.Value)
            '
            If IsNull(c_mingzi.OldValue) Then
                tTest = 1
            Else
                If StrComp(c_mingzi.OldValue, c_mingzi.Value, 0) = 0 Then
                    Else
                        tTest = 1
                    End If
                End If
            '
            If IsNull(c_mingzi_chn.OldValue) Then
                tTest = 1
            Else
                If StrComp(c_mingzi_chn.OldValue, c_mingzi_chn.Value, 0) = 0 Then
                    Else
                        tTest = 1
                    End If
                End If
            End If
        Else
            If IsNull(c_mingzi.Value) And IsNull(c_mingzi_chn.Value) Then
                If Not (IsNull(c_mingzi.OldValue) And IsNull(c_mingzi_chn.OldValue)) Then
                    tTest = 1
                End If
            End If
        End If

        If tTest = 1 Then
            tStr1 = Trim(c_surname_chn.Value) + tMingChn
            tStr2 = Trim(c_surname.Value) + " " + tMing
            '
            c_name_chn.Value = tStr1
            c_name.Value = tStr2
        End If
    End If
End Sub

Private Sub Form_Current()

```

```

    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

```

```

    If IsNull(c_choronym_code.Value) Then
        TxtChoroDesc.Value = ""
        TxtChoroDescCHN.Value = ""
    End If

```

```

Else
    rst.Open "CHORONYM_CODES", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_choronym_code = " & c_choronym_code.Value

    TxtChoroDesc.Value = rst.Fields("c_choronym_desc")
    TxtChoroDescCHN.Value = rst.Fields("c_choronym_chn")
    rst.Close
End If

If IsNull(c_dy.Value) Then
    TxtDynasty.Value = ""
    TxtDynastyCHN.Value = ""
Else
    rst.Open "DYNASTIES", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_dy = " & c_dy.Value

    TxtDynasty.Value = rst.Fields("c_dynasty")
    TxtDynastyCHN.Value = rst.Fields("c_dynasty_chn")
    rst.Close
End If

If IsNull(c_by_nh_code.Value) Then
    TxtBYNH.Value = ""
Else
    rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_nianhao_id = " & c_by_nh_code.Value

    TxtBYNH.Value = rst.Fields("c_nianhao_chn")
    rst.Close
End If

If IsNull(c_dy_nh_code.Value) Then
    TxtDYNH.Value = ""
Else
    rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_nianhao_id = " & c_dy_nh_code.Value

    TxtDYNH.Value = rst.Fields("c_nianhao_chn")
    rst.Close
End If

If IsNull(c_fl_ey_nh_code.Value) Then
    TxtFlEyNH.Value = ""
Else
    rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_nianhao_id = " & c_fl_ey_nh_code.Value

    TxtFlEyNH.Value = rst.Fields("c_nianhao_chn")
    rst.Close
End If

If IsNull(c_fl_ly_nh_code.Value) Then
    TxtFlLyNH.Value = ""
Else
    rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_nianhao_id = " & c_fl_ly_nh_code.Value

    TxtFlLyNH.Value = rst.Fields("c_nianhao_chn")
    rst.Close
End If

If IsNull(CmbBYRG.Value) Then
    TxtBYRG.Value = ""
Else
    rst.Open "year_range_codes", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_range_code = " & CmbBYRG.Value

    TxtBYRG.Value = rst.Fields("c_range_chn")

```

```

        rst.Close
    End If

    If IsNull(CmbDYRG.Value) Then
        TxtDYRG.Value = ""
    Else
        rst.Open "year_range_codes", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_range_code = " & CmbDYRG.Value

        TxtDYRG.Value = rst.Fields("c_range_chn")
        rst.Close
    End If

    If IsNull(DeathAgeApprox.Value) Then
        TxtApprox.Value = ""
    Else
        rst.Open "year_range_codes", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_range_code = " & DeathAgeApprox.Value

        TxtApprox.Value = rst.Fields("c_approx_chn")
        rst.Close
    End If

    If IsNull(c_source.Value) Then
        TxtTitle_CHN.Value = ""
    Else
        rst.Open "TEXT_CODES", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_textid = " & c_source.Value

        TxtTitle_CHN.Value = rst.Fields("c_title_chn")
        rst.Close
    End If

    If IsNull(c_ethnicity_code.Value) Then
        TxtEthnicity.Value = ""
    Else
        rst.Open "Ethnicity_codes", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_ethnicity_code = " & c_ethnicity_code.Value

        TxtEthnicity.Value = rst.Fields("c_ethnicity_desc_chn")
        rst.Close
    End If

End Sub

Private Sub CmdPickEthnicity_Click()
On Error GoTo Err_CmdPickEthnicity_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strETHN As String

    c_ethnicity_code.Visible = True
    c_ethnicity_code.SetFocus
    strETHN = c_ethnicity_code.Text

    stDocName = "frmPickETHNICITY"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strETHN

    If CurrentProject.AllForms("frmPickETHNICITY").IsLoaded Then
        Dim intETHN As Integer
        Dim strETHN_CHN As String

        Forms!frmPickETHNICITY!frmETHNICITY.Form!c_ethnicity_code.SetFocus
        intETHN = Forms!frmPickETHNICITY!frmETHNICITY.Form!c_ethnicity_code.Value
        c_ethnicity_code.Value = intETHN

        Forms!frmPickETHNICITY!frmETHNICITY.Form!c_ethnicity_desc_chn.SetFocus
        strETHN_CHN = Forms!frmPickETHNICITY!frmETHNICITY.Form!c_ethnicity_desc_chn.Value
        TxtEthnicity.Value = strETHN_CHN

        DoCmd.Close acForm, stDocName
    End If

    CmdPickEthnicity.SetFocus

```

```

        c_ethnicity_code.Visible = False
Exit_CmdPickEthnicity_Click:
    Exit Sub

Err_CmdPickEthnicity_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickEthnicity_Click

End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    If IsNull(c_ethnicity_code.Value) Then
        TxtEthnicity.Value = ""
    Else
        rst.Open "Ethnicity_codes", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_ethnicity_code = " & c_ethnicity_code.Value

        TxtEthnicity.Value = rst.Fields("c_ethnicity_desc_chn")
        rst.Close
    End If
End Sub

Private Sub CmdDelete_Click()
On Error GoTo Err_CmdDelete_Click

Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

Dim blnRecordAdded As Boolean

If Not IsNull(c_surname_chn) Then
    rst.Open "Del_Log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.AddNew
    rst!c_personid = c_personid
    rst!c_subform = "BIOG_MAIN"
    rst!c_flag = 1

    rst!c_surname_chn = c_surname_chn
    rst!c_surname = c_surname
    rst!c_mingzi_chn = c_mingzi_chn
    rst!c_mingzi = c_mingzi_chn
    rst!c_female = c_female
    rst!c_birthyear = c_birthyear
    rst!c_deathyear = c_deathyear
    rst!c_by_nh_code = c_by_nh_code
    rst!c_by_nh_year = c_by_nh_year
    rst!c_by_range = c_by_range
    rst!c_dy_nh_code = c_dy_nh_code
    rst!c_dy_nh_year = c_dy_nh_year
    rst!c_dy_range = c_dy_range
    rst!c_fl_earliest_year = c_fl_earliest_year
    rst!c_fl_latest_year = c_fl_latest_year
    rst!c_fl_ey_nh_code = c_fl_ey_nh_code
    rst!c_fl_ly_nh_code = c_fl_ly_nh_code
    rst!c_fl_ey_nh_year = c_fl_ey_nh_year
    rst!c_fl_ly_nh_year = c_fl_ly_nh_year
    rst!c_fl_ey_notes = c_fl_ey_notes
    rst!c_fl_ly_notes = c_fl_ly_notes
    rst!c_choronym_code = c_choronym_code
    rst!c_dy = c_dy
    rst!c_index_year = c_index_year
    rst!c_death_age = c_death_age
    rst!c_ethnicity_code = c_ethnicity_code
    rst!c_tribe = c_tribe
    rst!c_surname_code = c_surname_code
    rst!c_jia = c_jia
    rst!c_zu = c_zu

    rst!c_source = c_source
    rst!c_pages = c_pages
    rst!c_notes = c_notes
    rst.Update
    blnRecordAdded = True
    rst.Close
End If

```



```
DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70

Exit_CmdDelete_Click:
Exit Sub

Err_CmdDelete_Click:
MsgBox Err.Description
Resume Exit_CmdDelete_Click

End Sub

Private Sub CmdOpenTEXTS_EnterForm_Click()
On Error GoTo Err_CmdOpenTEXTS_EnterForm_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "TEXTS EnterForm"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdOpenTEXTS_EnterForm_Click:
Exit Sub

Err_CmdOpenTEXTS_EnterForm_Click:
MsgBox Err.Description
Resume Exit_CmdOpenTEXTS_EnterForm_Click

End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

Screen.PreviousControl.SetFocus
DoCmd.DoMenuItem acFormBar, acEditMenu, 10, , acMenuVer70

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click

End Sub

Private Sub CmdAddNew_Click()
On Error GoTo Err_CmdAddNew_Click

DoCmd.GoToRecord , , acNewRec

c_surname.SetFocus

Exit_CmdAddNew_Click:
Exit Sub

Err_CmdAddNew_Click:
MsgBox Err.Description
Resume Exit_CmdAddNew_Click

End Sub
```

```

Option Compare Database
Public gLabelsOK As Boolean, gDisplayLanguage As String
Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 10) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 10 And Not .EOF
            If !c_form = "SAN" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then
            tLang = 2
        Else
            tLang = 3
        End If
        '
        ' now comes the basic routine
        '
        ' Me.CmdDelete.Caption = tLabelLanguage(tLang, 1)
        ' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 2)
        Me.LblSource.Caption = tLabelLanguage(tLang, 3)
        Me.LBL_alt_name.Caption = tLabelLanguage(tLang, 4)
        Me.LBL_alt_name_chn.Caption = tLabelLanguage(tLang, 5)
        Me.LblNameType.Caption = tLabelLanguage(tLang, 6)
        Me.LBL_c_pages.Caption = tLabelLanguage(tLang, 7)
        Me.LBL_c_notes.Caption = tLabelLanguage(tLang, 8)
        Me.LBL_c_sequence.Caption = tLabelLanguage(tLang, 9)

    End If
End Sub

```

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 28) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 28 And Not .EOF

If !c_form = "SF_ASD" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

Me.LblSequence.Caption = tLabelLanguage(tLang, 1)

Me.LblYear.Caption = tLabelLanguage(tLang, 2)

Me.LblRange.Caption = tLabelLanguage(tLang, 3)

Me.LblNHYear.Caption = tLabelLanguage(tLang, 4)

Me.LblIntercalary.Caption = tLabelLanguage(tLang, 5)

Me.LblGZ.Caption = tLabelLanguage(tLang, 6)

Me.LblSupplement.Caption = tLabelLanguage(tLang, 7)

Me.LblOccasion.Caption = tLabelLanguage(tLang, 8)

Me.LblTitle.Caption = tLabelLanguage(tLang, 9)

Me.LblGenre.Caption = tLabelLanguage(tLang, 10)

Me.LblPages.Caption = tLabelLanguage(tLang, 11)

Me.LblNotes.Caption = tLabelLanguage(tLang, 12)

Me.LblAssocName.Caption = tLabelLanguage(tLang, 13)

Me.LblAssocDesc.Caption = tLabelLanguage(tLang, 14)

Me.LblKinRel.Caption = tLabelLanguage(tLang, 15)

Me.LblKinName.Caption = tLabelLanguage(tLang, 16)

Me.LblAssocKinRel.Caption = tLabelLanguage(tLang, 17)

Me.LblAssocKinChn.Caption = tLabelLanguage(tLang, 18)

Me.LblAssocAddr.Caption = tLabelLanguage(tLang, 19)

Me.LblAssocNH.Caption = tLabelLanguage(tLang, 20)

Me.LblTopicChn.Caption = tLabelLanguage(tLang, 21)

Me.LblSocInst.Caption = tLabelLanguage(tLang, 22)

Me.LblSource.Caption = tLabelLanguage(tLang, 23)

'Me.CmdDelete.Caption = tLabelLanguage(tLang, 24)

'Me.CmdAddNew.Caption = tLabelLanguage(tLang, 25)

Me.LblClaimerChn.Caption = tLabelLanguage(tLang, 26)

Me.LblCount.Caption = tLabelLanguage(tLang, 27)

End If

End Sub

Public Sub noEdits()

Me.AllowAdditions = False

Me.AllowDeletions = False

Me.AllowEdits = False

End Sub

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 22) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 22 And Not .EOF

If !c_form = "SF_ADDR" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

Me.LblSequence.Caption = tLabelLanguage(tLang, 1)

Me.LblMaternal.Caption = tLabelLanguage(tLang, 2)

Me.LblFirstyear.Caption = tLabelLanguage(tLang, 3)

Me.LblFYRange.Caption = tLabelLanguage(tLang, 4)

Me.LblFYNHYear.Caption = tLabelLanguage(tLang, 5)

Me.LblFYIntercalary.Caption = tLabelLanguage(tLang, 6)

Me.LblFYGZ.Caption = tLabelLanguage(tLang, 7)

Me.LblLastyear.Caption = tLabelLanguage(tLang, 8)

Me.LblLYRange.Caption = tLabelLanguage(tLang, 9)

Me.LblLYNHyear.Caption = tLabelLanguage(tLang, 10)

Me.LblLYIntercalary.Caption = tLabelLanguage(tLang, 11)

Me.LblLYGZ.Caption = tLabelLanguage(tLang, 12)

Me.LblPages.Caption = tLabelLanguage(tLang, 13)

Me.LblNotes.Caption = tLabelLanguage(tLang, 14)

Me.LblAddrType.Caption = tLabelLanguage(tLang, 15)

Me.LblAddrName.Caption = tLabelLanguage(tLang, 16)

Me.LblFYNH.Caption = tLabelLanguage(tLang, 17)

Me.LblLYNH.Caption = tLabelLanguage(tLang, 18)

Me.LblSource.Caption = tLabelLanguage(tLang, 19)

' Me.CmdDelete.Caption = tLabelLanguage(tLang, 20)

' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 21)

End If

End Sub

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean, gFirstTime As Integer

Private Sub c_by_nh_code_AfterUpdate()

Dim rst As ADODB.Recordset

Set rst = New ADODB.Recordset

If IsNull(c_by_nh_code.Value) Then

TxtBYNH.Value = ""

Else

rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _

adLockOptimistic

rst.Find "c_nianhao_id = " & c_by_nh_code.Value

TxtBYNH.Value = rst.Fields("c_nianhao_chn")

rst.Close

End If

End Sub

Private Sub c_dy_nh_code_AfterUpdate()

Dim rst As ADODB.Recordset

Set rst = New ADODB.Recordset

If IsNull(c_dy_nh_code.Value) Then

TxtDYNH.Value = ""

Else

rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _

adLockOptimistic

rst.Find "c_nianhao_id = " & c_dy_nh_code.Value

TxtDYNH.Value = rst.Fields("c_nianhao_chn")

rst.Close

End If

End Sub

Private Sub c_fl_ey_nh_code_AfterUpdate()

Dim rst As ADODB.Recordset

Set rst = New ADODB.Recordset

If IsNull(c_fl_ey_nh_code.Value) Then

TxtFlEyNH.Value = ""

Else

rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _

adLockOptimistic

rst.Find "c_nianhao_id = " & c_fl_ey_nh_code.Value

TxtFlEyNH.Value = rst.Fields("c_nianhao_chn")

rst.Close

End If

End Sub

Private Sub c_fl_ey_notes_Click()

Dim stDocName As String

Dim stLinkCriteria As String

Dim strNH As String

c_fy_nh_code.Visible = True

c_fy_nh_code.SetFocus

strNH = c_fy_nh_code.Text

stDocName = "frmPickNIAN_HAO"

DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then

Dim intNH As Integer

Dim strNH_CHN As String

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus

intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value

c_fy_nh_code.Value = intNH

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus

strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value

TxtFYNH.Value = strNH_CHN

DoCmd.Close acForm, stDocName

End If

```

CmdPickFYNH.SetFocus
c_fy_nh_code.Visible = False

```

```

Exit_CmdPickFYNH_Click:
Exit Sub

```

```

Err_CmdPickFYNH_Click:
MsgBox Err.Description
Resume Exit_CmdPickFYNH_Click

```

```

End Sub

```

```

Private Sub c_fl_ly_nh_code_AfterUpdate()
Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

If IsNull(c_fl_ly_nh_code.Value) Then
    TxtFlLyNH.Value = ""
Else
    rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_nianhao_id = " & c_fl_ly_nh_code.Value

    TxtFlLyNH.Value = rst.Fields("c_nianhao_chn")
    rst.Close
End If
End Sub

```

```

Private Sub c_mingzi_chn_AfterUpdate()
Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset
Dim strSUR As String
Dim strNM As String
Dim strSUR_Find As String
Dim strNM_Find As String
Dim Counter As Long
Dim intPerson As Long

If Not IsNull(c_mingzi_chn.Value) And Not IsNull(c_surname_chn.Value) Then
    Counter = 0

    strSUR = c_surname_chn.Value
    strNM = c_mingzi_chn.Value
    intPerson = c_personid.Value

    rst.Open "BIOG_MAIN", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    Do
        If rst.EOF = True Then
            Exit Do
        End If

        If IsNull(rst!c_surname_chn.Value) Then
        Else
            strSUR_Find = rst!c_surname_chn.Value
        End If

        If IsNull(rst!c_mingzi_chn.Value) Then
        Else
            strNM_Find = rst!c_mingzi_chn.Value
        End If

        If StrComp(strSUR_Find, strSUR) = 0 And StrComp(strNM_Find, strNM) = 0 Then
            If rst!c_personid = intPerson Then
                'This is to exclude the current record from being counted.
                rst.MoveNext
            Else
                Counter = Counter + 1
                rst.MoveNext
            End If
        Else
            rst.MoveNext
        End If
    Loop

    If Counter > 0 Then
        TxtNameChn.SetFocus
        MsgBox "" & Chr(10) & Counter & " - 1ĐŌÄüíēÈ«İàİ-µÄ«œ>;ŁŒ^'_ŌJ éwİÂŌýŌÚŸ"ÈèµÄĐĀİčĂc"µ"pŽì-FŌĐ"µ"p>]ŌĐŌŦŦ};Ł
    End If
End Sub

```

```

Form_BIOG_MAIN_2_Subform - 3
éwĩÄ;ÉÒÔÊ¹ÓÄÖÊİ ·¼µÄ²éÔf°´âo;çÒÔ-FÓÐÓ>ä>ÐÖÄÛ `ÊÑÊ÷™Ú£¬²éÔf-FÓÐ"µ"p."
End If

End If
End Sub

Private Sub c_mingzi_chn_GotFocus()
If IsNull(c_surname.Value) Or c_surname = "" Then
MsgBox "Õ^İĚŸ"Èè Xing !"
c_surname.SetFocus
Else
If IsNull(c_mingzi.Value) Or c_mingzi = "" Then
MsgBox "Õ^İĚŸ"Èè Ming!"
c_mingzi.SetFocus
Else
If IsNull(c_surname_chn.Value) Or c_surname_chn = "" Then
MsgBox "Õ^İĚŸ"Èè ÐÖ!"
c_surname_chn.SetFocus
End If
End If
End If
End Sub

Private Sub c_mingzi_GotFocus()
If IsNull(c_surname.Value) Or c_surname = "" Then
MsgBox "Õ^İĚŸ"Èè Xing !"
c_surname.SetFocus
End If
End Sub

Private Sub c_personid_current()
'
' if there is a person ID, enable the store-ID command
'
MsgBox c_personid.Text
If IsNull(Me.c_personid.Value) Then
Me.CmdStoreID.Enabled = False
Else
Me.CmdStoreID.Enabled = True
End If
End Sub

End Sub

Private Sub c_surname_AfterUpdate()

Dim intLastID As Long
Dim intID As Long

If IsNull(c_surname.Value) Or c_surname = "" Then
Else
If IsNull(c_personid.Value) Then
intLastID = DMax("c_personid", "BIOG_MAIN")
TxtLastID.Value = intLastID

TxtLastID.Visible = True
TxtLastID.SetFocus
intID = TxtLastID.Value
c_personid.Value = intID + 1

c_mingzi.SetFocus
TxtLastID.Visible = False
End If
End If
End Sub

Private Sub c_surname_chn_BeforeUpdate(Cancel As Integer)
Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset
Dim strSUR As String
Dim strNM As String
Dim strSUR_Find As String
Dim strNM_Find As String
Dim Counter As Long
Dim intPerson As Long

If Not IsNull(c_mingzi_chn.Value) And Not IsNull(c_surname_chn.Value) Then
Counter = 0

```



```
strSUR = c_surname_chn.Value
strNM = c_mingzi_chn.Value
intPerson = c_personid.Value
```

```
rst.Open "BIOG_MAIN", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
Do
```

```
    If rst.EOF = True Then
        Exit Do
    End If
```

```
    If IsNull(rst!c_surname_chn.Value) Then
    Else
        strSUR_Find = rst!c_surname_chn.Value
    End If
```

```
    If IsNull(rst!c_mingzi_chn.Value) Then
    Else
        strNM_Find = rst!c_mingzi_chn.Value
    End If
```

```
    If StrComp(strSUR_Find, strSUR) = 0 And StrComp(strNM_Find, strNM) = 0 Then
        If rst!c_personid = intPerson Then
```

```
            'This is to exclude the current record from being counted.
```

```
            rst.MoveNext
```

```
        Else
```

```
            Counter = Counter + 1
```

```
            rst.MoveNext
```

```
        End If
```

```
    Else
```

```
        rst.MoveNext
```

```
    End If
```

```
Loop
```

```
If Counter > 0 Then
```

```
    TxtNameChn.SetFocus
```

```
    MsgBox ""p"pŽi°l~F" & Counter & "-lĐŌĂŭíÊÈ«İàİ~µÄ¼oä>ı£Ō^´ ŌJ éwİÄŌŸŌÚŸ"ÈèµÄĐÄİçÅc"µ"pŽi~FŌĐ"µ"p>]ŌĐŌŌŊ};£  
éwİÄŸÉŌŌÊ¹ŌÄŌŌÊİ·¼µÄ²éŌf°´âoıçŌŌ~FŌĐŌ>ä>ĐŌĂŭ `ÈÑÈ÷™Ū£~²éŌf~FŌĐ"µ"p."
```

```
End If
```

```
End If
```

```
End Sub
```

```
Private Sub CmdExplainInput_Click()
On Error GoTo Err_CmdExplainInput_Click
```

```
    Dim stDocName As String
    Dim stLinkCriteria As String
```

```
    stDocName = "FrmExplainInput"
```

```
    DoCmd.OpenForm stDocName, , , stLinkCriteria
```

```
Exit_CmdExplainInput_Click:
```

```
    Exit Sub
```

```
Err_CmdExplainInput_Click:
```

```
    MsgBox Err.Description
```

```
    Resume Exit_CmdExplainInput_Click
```

```
End Sub
```

```
Private Sub CmdStoreID_Click()
```

```
    Dim cmdSQL As ADODB.Command, tRecCount As Variant
```

```
    Set cmdSQL = New ADODB.Command
```

```
    cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
    cmdSQL.CommandType = adCmdText
```

```
    If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
```

```
        ' Display message.
```

```
        If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) =  
vbNo Then
```

```
            Exit Sub
```

```
        Else
```

```
            cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
```

```
            cmdSQL.Execute tRecCount
```

```

        End If
    End If

    cmdSQL.CommandText = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT " + Str(Me.c_personid.Value) + " A
S c_personid"
    cmdSQL.Execute tRecCount
    MsgBox "Person ID successfully stored. Click on 'Recall Person IDs' to reuse this ID in other forms."
    '
    ' update storage source
    cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='Browser' WHERE PersonIDSource.LineNum =1"
    cmdSQL.Execute tRecCount
End Sub

```

```

Private Sub Form_AfterDelConfirm(STATUS As Integer)
Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

If STATUS = acDeleteOK Then
    rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.Find "c_flag = " & 1
    rst!c_flag = 0
    rst.Update
    rst.Close
Else
    rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.Find "c_flag = " & 1
    rst.Delete
    rst.Update
    rst.Close
End If
End Sub

```

```

Private Sub Form_BeforeUpdate(Cancel As Integer)
Dim tTest As Integer
Dim tStr1 As String, tStr2 As String, tMing As String, tMingChn As String

tTest = 0
tMing = ""
tMingChn = ""
'
' test to see that surnames are non-NULL
'
If Not IsNull(c_surname.Value) And Not IsNull(c_surname_chn.Value) Then
    '
    ' then look for any changes
    '
    If StrComp(c_surname.OldValue, c_surname.Value, 0) = 0 Then
    Else
        tTest = 1
    End If
    '
    If StrComp(c_surname_chn.OldValue, c_surname_chn.Value, 0) = 0 Then
    Else
        tTest = 1
    End If
    '
    ' personal names are allowed to be NULL, so we need to check if a name has
    ' been deleted as well as if one has been added
    '
    If Not IsNull(c_mingzi.Value) And Not IsNull(c_mingzi_chn.Value) Then
        tMingChn = Trim(c_mingzi_chn.Value)
        tMing = Trim(c_mingzi.Value)
        '
        If IsNull(c_mingzi.OldValue) Then
            tTest = 1
        Else
            If StrComp(c_mingzi.OldValue, c_mingzi.Value, 0) = 0 Then
            Else
                tTest = 1
            End If
        End If
        '
        If IsNull(c_mingzi_chn.OldValue) Then
            tTest = 1
        Else
            If StrComp(c_mingzi_chn.OldValue, c_mingzi_chn.Value, 0) = 0 Then
            Else
                tTest = 1
            End If
        End If
    End If
End Sub

```

```

        End If
    End If
Else
    If IsNull(c_mingzi.Value) And IsNull(c_mingzi_chn.Value) Then
        If Not (IsNull(c_mingzi.OldValue) And IsNull(c_mingzi_chn.OldValue)) Then
            tTest = 1
        End If
    End If
End If

If tTest = 1 Then
    tStr1 = Trim(c_surname_chn.Value) + tMingChn
    tStr2 = Trim(c_surname.Value) + " " + tMing
    '
    c_name_chn.Value = tStr1
    c_name.Value = tStr2
End If
End If
End Sub

```

```
Private Sub Form_Current()
```

```
    ' make sure the other subform is pointing to the right record
```

```

    'If c_personid.Value <> Forms!CBDB_Browser!frmPeopleLookup.Form!c_personid.Value Then
    '    Set tRstLookup = Forms!CBDB_Browser!frmPeopleLookup.Form.Recordset
    '    tRstLookup.FindFirst "c_personid = " + Trim(Str(c_personid.Value))
    '    Set tRstLookup = Nothing
'End If

```

```

If IsNull(Me.c_personid.Value) Then
    Me.CmdStoreID.Enabled = False
Else
    Me.CmdStoreID.Enabled = True
End If

```

```

End Sub
Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 59) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 59 And Not .EOF
            If !c_form = "BIO" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

```

```

If gLabelsOK Then
    If gDisplayLanguage = "E" Then
        tLang = 1
        Me.TabCtl14.FontSize = 8
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
        Me.TabCtl14.FontSize = 10
    Else
        tLang = 3
        Me.TabCtl14.FontSize = 10
    End If
End If

```

```

End If
'
'   now comes the basic routine
'
Me.LblFemale.Caption = tLabelLanguage(tLang, 1)
Me.LblIndexYear.Caption = tLabelLanguage(tLang, 2)
' Me.LblTribe.Caption = tLabelLanguage(tLang, 3)
' Me.LblFullNameChn.Caption = tLabelLanguage(tLang, 4)
' Me.LblFullname.Caption = tLabelLanguage(tLang, 5)
' Me.LblPages.Caption = tLabelLanguage(tLang, 6)
Me.LblNotes.Caption = tLabelLanguage(tLang, 7)
Me.LblBirthyear.Caption = tLabelLanguage(tLang, 8)
Me.LblBYNianhaoDate.Caption = tLabelLanguage(tLang, 9)
Me.LblBYIntercalary.Caption = tLabelLanguage(tLang, 10)
Me.LblBYGZ.Caption = tLabelLanguage(tLang, 11)
Me.LblBYRange.Caption = tLabelLanguage(tLang, 12)
Me.LblDeathyear.Caption = tLabelLanguage(tLang, 13)
Me.LblDYNianhaoDate.Caption = tLabelLanguage(tLang, 14)
Me.LblDYIntercalary.Caption = tLabelLanguage(tLang, 15)
Me.LblDYGZ.Caption = tLabelLanguage(tLang, 16)
Me.LblDYRange.Caption = tLabelLanguage(tLang, 17)
Me.LblDeathAge.Caption = tLabelLanguage(tLang, 18)
Me.LblDeathAgeRange.Caption = tLabelLanguage(tLang, 19)
Me.LblFloruitFirstYear.Caption = tLabelLanguage(tLang, 20)
Me.LblFloruitFirstNHYear.Caption = tLabelLanguage(tLang, 21)
Me.LblFloruitFirstYearNotes.Caption = tLabelLanguage(tLang, 22)
Me.LblFloruitLastYear.Caption = tLabelLanguage(tLang, 23)
Me.LblFloruitLastNHYear.Caption = tLabelLanguage(tLang, 24)
Me.LblFloruitLastYearNotes.Caption = tLabelLanguage(tLang, 25)
' Me.LblOpenEvents.Caption = tLabelLanguage(tLang, 26)

Me.LblDynasty.Caption = tLabelLanguage(tLang, 27)
' Me.CmdFind.Caption = tLabelLanguage(tLang, 28)
Me.LblChoronym.Caption = tLabelLanguage(tLang, 29)
Me.LblEthnicity.Caption = tLabelLanguage(tLang, 30)
' Me.LblSource.Caption = tLabelLanguage(tLang, 31)
Me.LblBYNH.Caption = tLabelLanguage(tLang, 32)
Me.LblDYNH.Caption = tLabelLanguage(tLang, 33)
Me.LblFLEY.Caption = tLabelLanguage(tLang, 34)
Me.LblFLLY.Caption = tLabelLanguage(tLang, 35)
' Me.CmdDelete.Caption = tLabelLanguage(tLang, 36)
' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 37)
' Me.CmdJianti.Caption = tLabelLanguage(tLang, 38)
' Me.CmdFanti.Caption = tLabelLanguage(tLang, 39)
' Me.CmdExplain.Caption = tLabelLanguage(tLang, 40)
' Me.CmdOpenTEXTS_EnterForm.Caption = tLabelLanguage(tLang, 41)
' Me.CmdOpenEvents.Caption = tLabelLanguage(tLang, 42)

Me.PageBirthDeathYears.Caption = tLabelLanguage(tLang, 43)
Me.PageAddresses.Caption = tLabelLanguage(tLang, 44)
Me.PageAltNames.Caption = tLabelLanguage(tLang, 45)
Me.PageWritings.Caption = tLabelLanguage(tLang, 46)
Me.PageEntry.Caption = tLabelLanguage(tLang, 47)
Me.PageEvents.Caption = tLabelLanguage(tLang, 48)
Me.PageKinship.Caption = tLabelLanguage(tLang, 49)
Me.PageAssociations.Caption = tLabelLanguage(tLang, 50)
Me.PagePossessions.Caption = tLabelLanguage(tLang, 51)
Me.PageStatus.Caption = tLabelLanguage(tLang, 52)
Me.PagePosting.Caption = tLabelLanguage(tLang, 53)
Me.PageBiogInst.Caption = tLabelLanguage(tLang, 54)
Me.PageSource.Caption = tLabelLanguage(tLang, 55)
Me.Label284.Caption = tLabelLanguage(tLang, 56)
Me.CmdStoreID.Caption = tLabelLanguage(tLang, 57)
Me.LblPersonID.Caption = tLabelLanguage(tLang, 58)

' MsgBox "Finished BIOG_MAIN_2 labels"
'   now for the subforms

' MsgBox "Beginning ASSOC_DATA_2 labels"

Me.ASSOC_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.ASSOC_DATA_2_Subform.Form.changeDisplayLanguage

' MsgBox "Beginning TEXT_DATA_2 labels"

Me.TEXT_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.TEXT_DATA_2_Subform.Form.changeDisplayLanguage

' MsgBox "Beginning ADDR_DATA_2 labels"

```

```
Me.BIOG_ADDR_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.BIOG_ADDR_DATA_2_Subform.Form.changeDisplayLanguage
```

```
'MsgBox "Beginning ENTRY_DATA_2 labels"
```

```
Me.ENTRY_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.ENTRY_DATA_2_Subform.Form.changeDisplayLanguage
```

```
'MsgBox "Beginning EVENTS_DATA_2 labels"
```

```
Me.EVENTS_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.EVENTS_DATA_2_Subform.Form.changeDisplayLanguage
```

```
'MsgBox "Beginning ALT_NAME_2 labels"
```

```
Me.ALTNAME_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.ALTNAME_DATA_2_Subform.Form.changeDisplayLanguage
```

```
'MsgBox "Beginning KIN_DATA_2 labels"
```

```
Me.KIN_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.KIN_DATA_2_Subform.Form.changeDisplayLanguage
```

```
'MsgBox "Beginning POSSESSION_DATA_2 labels"
```

```
Me.POSSESSION_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.POSSESSION_DATA_2_Subform.Form.changeDisplayLanguage
```

```
'MsgBox "Beginning STATUS_DATA_2 labels"
```

```
Me.STATUS_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.STATUS_DATA_2_Subform.Form.changeDisplayLanguage
```

```
'MsgBox "Beginning POSTING_DATA_2 labels"
```

```
Me.POSTING_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
Me.POSTING_DATA_2_Subform.Form.changeDisplayLanguage
```

```
'MsgBox "Beginning frmBIOG_SOURCE_DATA labels"
```

```
Me.frmBIOG_SOURCE_DATA.Form.gDisplayLanguage = gDisplayLanguage
Me.frmBIOG_SOURCE_DATA.Form.changeDisplayLanguage
```

```
'MsgBox "Beginning frmBIOG_INST_DATA labels"
```

```
Me.frmBIOG_INST_CODES.Form.gDisplayLanguage = gDisplayLanguage
Me.frmBIOG_INST_CODES.Form.changeDisplayLanguage
```

```
'MsgBox "Finished all subforms"
```

```
End If
```

```
End Sub
```

Option Compare Database

Public gLabelsOK As Boolean, gDisplayLanguage As String

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 21) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 21 And Not .EOF

If !c_form = "SEN" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

Me.LblEntry.Caption = tLabelLanguage(tLang, 1)

Me.LblNianHao.Caption = tLabelLanguage(tLang, 2)

Me.LblPosting.Caption = tLabelLanguage(tLang, 3)

Me.LblKinRel.Caption = tLabelLanguage(tLang, 4)

Me.LblKinName.Caption = tLabelLanguage(tLang, 5)

Me.LblAssocDesc.Caption = tLabelLanguage(tLang, 6)

Me.LblAssocName.Caption = tLabelLanguage(tLang, 7)

Me.Lbl_source.Caption = tLabelLanguage(tLang, 8)

Me.LBL_c_sequence.Caption = tLabelLanguage(tLang, 11)

Me.LBL_c_year.Caption = tLabelLanguage(tLang, 12)

Me.LBL_nianhao_year.Caption = tLabelLanguage(tLang, 13)

Me.LBL_time_range.Caption = tLabelLanguage(tLang, 14)

Me.LBL_explain.Caption = tLabelLanguage(tLang, 15)

Me.LBL_c_pages.Caption = tLabelLanguage(tLang, 16)

Me.LBL_c_notes.Caption = tLabelLanguage(tLang, 17)

Me.LBL_c_exam_rank.Caption = tLabelLanguage(tLang, 18)

Me.LBL_entry_age.Caption = tLabelLanguage(tLang, 19)

Me.LblEntryPlace.Caption = tLabelLanguage(tLang, 20)

End If

End Sub

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 4) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 4 And Not .EOF

If !c_form = "SF_EVA" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

Me.LblName.Caption = tLabelLanguage(tLang, 1)

' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 2)

' Me.CmdDelete.Caption = tLabelLanguage(tLang, 3)

End If

End Sub

Public Sub noEdits()

Me.AllowAdditions = False

Me.AllowDeletions = False

Me.AllowEdits = False

End Sub

Option Compare Database

```
Private Sub c_event_name_chn_AfterUpdate()
```

```
Dim intLastID As Long
```

```
Dim intID As Long
```

```
If IsNull(c_event_name_chn.Value) Or c_title_chn = "" Then
Else
```

```
    If IsNull(c_event_code.Value) Then
```

```
        intLastID = DMax("c_event_code", "TEXT_CODES")
```

```
        TxtLastID.Value = intLastID
```

```
        TxtLastID.Visible = True
```

```
        TxtLastID.SetFocus
```

```
        intID = TxtLastID.Value
```

```
        c_event_code.Value = intID + 1
```

```
        c_event_code.SetFocus
```

```
        TxtLastID.Visible = False
```

```
    End If
```

```
End If
```

```
Dim rst As ADODB.Recordset
```

```
Set rst = New ADODB.Recordset
```

```
Dim strEVENT As String
```

```
Dim strEVENT_Find As String
```

```
Dim Counter As Long
```

```
Dim intEVENT As Long
```

```
Counter = 0
```

```
strEVENT = c_event_name_chn.Value
```

```
intEVENT = c_event_code.Value
```

```
rst.Open "EVENT_CODES", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
Do
```

```
    If rst.EOF = True Then
```

```
        Exit Do
```

```
    End If
```

```
    If IsNull(rst!c_event_name_chn.Value) Then
```

```
    Else
```

```
        strEVENT_Find = rst!c_event_name_chn.Value
```

```
    End If
```

```
    If StrComp(strEVENT_Find, strEVENT) = 0 Then
```

```
        If rst!c_event_code = intEVENT Then
```

```
            'This is to exclude the current record from being counted.
```

```
            rst.MoveNext
```

```
        Else
```

```
            Counter = Counter + 1
```

```
            rst.MoveNext
```

```
        End If
```

```
    Else
```

```
        rst.MoveNext
```

```
    End If
```

```
Loop
```

```
If Counter > 0 Then
```

```
MsgBox "μ`pžī°l~F" & Counter & "-lĒÂ¼pĀû·QíêE«İâí~μĀ¼oä>ıŁŌ~` ŌJ éwİĀŌýŌÚÝ"EèμĀĐĀİċĀc"μ`pžī~FŌĐ"μ`p>]ŌĐŌŌN}
;éwİĀ;ÉŌŌÉ'ŌĀŌŌÉİ·¼μĀ²éŌf°'âo;ċŌŌŌĐİĀĒĒ¼pĀû·QžéĒĒĒ÷"ŪŁ~²éŌf~FŌĐ"μ`p."
```

```
End If
```

```
End Sub
```

```
Private Sub CmdDelete_Click()
```

```
On Error GoTo Err_CmdDelete_Click
```

```
Dim rst As ADODB.Recordset
```

```
Set rst = New ADODB.Recordset
```

```
Dim blnRecordAdded As Boolean
```

```
If Not IsNull(c_event_code) Then
```

```
'This If condition is to make sure that the record one is attempting to delete is not an empty record. Thus, the
field used in the condition must be a required field for the record.
```

```
rst.Open "Del_Log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
```


Form_EVENT_CODES EnterForm -2

```
rst.AddNew
rst!c_event_code = c_event_code
rst!c_subform = "EVENT_CODES EnterForm"
rst!c_flag = 1
rst!c_event_name_chn = c_event_name_chn
rst!c_event_name = c_event_name
rst!c_firstyear = c_fy_yr
rst!c_fy_nh_code = c_fy_nh_code
rst!c_fy_nh_yr = c_fy_nh_yr
rst!c_fy_range = c_fy_range
rst!c_lastyear = c_ly_yr
rst!c_ly_nh_code = c_ly_nh_code
rst!c_ly_range = c_ly_range
rst!c_dy = c_dy
rst!c_addr_id = c_addr_id
rst!c_source = c_source
rst!c_pages = c_pages
rst!c_notes = c_event_notes
```

```
rst.Update
blnRecordAdded = True
rst.Close
End If
```

```
DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
```

```
Exit_CmdDelete_Click:
Exit Sub
```

```
Err_CmdDelete_Click:
MsgBox Err.Description
Resume Exit_CmdDelete_Click
End Sub
```

```
Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click
```

```
Screen.PreviousControl.SetFocus
DoCmd.DoMenuItem acFormBar, acEditMenu, 10, , acMenuVer70
```

```
Exit_CmdFind_Click:
Exit Sub
```

```
Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub CmdPickAddr_Click()
On Error GoTo Err_CmdPickAddr_Click
```

```
Dim stDocName As String
Dim stLinkCriteria As String
Dim strNH As String
```

```
c_addr_id.Visible = True
c_addr_id.SetFocus
strNH = c_addr_id.Text
```

```
stDocName = "frmPickADDRESSES"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH
```

```
If CurrentProject.AllForms("frmPickADDRESSES").IsLoaded Then
Dim intNH As Integer
Dim strNH_CHN As String
```

```
Forms!frmPickADDRESSES!frmADDRESSES.Form!c_addr_id.SetFocus
intNH = Forms!frmPickADDRESSES!frmADDRESSES.Form!c_addr_id.Value
c_addr_id.Value = intNH
```

```
Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name_chn.SetFocus
strNH_CHN = Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name_chn.Value
TxtADDR.Value = strNH_CHN
```

```
DoCmd.Close acForm, stDocName
End If
```

```

    CmdPickAddr.SetFocus
    c_addr_id.Visible = False

```

```

Exit_CmdPickAddr_Click:
    Exit Sub

```

```

Err_CmdPickAddr_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickAddr_Click
End Sub

```

```

Private Sub CmdPickDynasty_Click()
On Error GoTo Err_CmdPickDynasty_Click

```

```

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strDy As String

```

```

    c_dy.Visible = True
    c_dy.SetFocus
    strDy = c_dy.Text

```

```

    stDocName = "frmPickDynasty"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strDy

```

```

    If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

```

```

        Dim intDy As Integer
        Dim strDy_Desc As String
        Dim strDy_CHN As String

```

```

        Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
        intDy = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value
        c_dy.Value = intDy

```

```

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
        strDy_Desc = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value
        TxtDynasty.Value = strDy_Desc

```

```

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
        strDy_CHN = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
        TxtDynastyCHN.Value = strDy_CHN

```

```

        DoCmd.Close acForm, stDocName
    End If

```

```

    CmdPickDynasty.SetFocus
    c_dy.Visible = False

```

```

Exit_CmdPickDynasty_Click:
    Exit Sub

```

```

Err_CmdPickDynasty_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickDynasty_Click

```

```

End Sub

```

```

Private Sub CmdPickFYNH_Click()
On Error GoTo Err_CmdPickFYNH_Click

```

```

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strNH As String

```

```

    c_fy_nh_code.Visible = True
    c_fy_nh_code.SetFocus
    strNH = c_fy_nh_code.Text

```

```

    stDocName = "frmPickNIAN_HAO"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

```

```

    If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
        Dim intNH As Integer
        Dim strNH_CHN As String

```

```

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
        intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value

```

```

        c_fy_nh_code.Value = intNH

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
        strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
        TxtFYNH.Value = strNH_CHN

        DoCmd.Close acForm, stDocName
    End If

    CmdPickFYNH.SetFocus
    c_fy_nh_code.Visible = False

Exit_CmdPickFYNH_Click:
    Exit Sub

Err_CmdPickFYNH_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickFYNH_Click
End Sub

Private Sub CmdPickLYNH_Click()
On Error GoTo Err_CmdPickLYNH_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strNH As String

    c_ly_nh_code.Visible = True
    c_ly_nh_code.SetFocus
    strNH = c_ly_nh_code.Text

    stDocName = "frmPickNIAN_HAO"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

    If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
        Dim intNH As Integer
        Dim strNH_CHN As String

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
        intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
        c_ly_nh_code.Value = intNH

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
        strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
        TxtLYNH.Value = strNH_CHN

        DoCmd.Close acForm, stDocName
    End If

    CmdPickLYNH.SetFocus
    c_ly_nh_code.Visible = False

Exit_CmdPickLYNH_Click:
    Exit Sub

Err_CmdPickLYNH_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickLYNH_Click
End Sub

Private Sub CmdPickSource_Click()
On Error GoTo Err_CmdPickSource_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strSC As String

    c_source.Visible = True
    c_source.SetFocus
    strSC = c_source.Text

    stDocName = "frmPickTEXTS"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strSC

    If CurrentProject.AllForms("frmPickTEXTS").IsLoaded Then
        Dim intSC As Long
        Dim strSC_CHN As String

        Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.SetFocus

```

```

    intSC = Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.Value
    c_source.Value = intSC

    Forms!frmPickTEXTS!frmTEXTS.Form!c_title.SetFocus
    strSC_CHN = Forms!frmPickTEXTS!frmTEXTS.Form!c_title_chn.Value
    TxtTitle_CHN.Value = strSC_CHN

```

```

    DoCmd.Close acForm, stDocName
End If

```

```

CmdPickSource.SetFocus
c_source.Visible = False

```

```

Exit_CmdPickSource_Click:
Exit Sub

```

```

Err_CmdPickSource_Click:
MsgBox Err.Description
Resume Exit_CmdPickSource_Click
End Sub

```

```

Private Sub CmdAddNew_Click()
On Error GoTo Err_CmdAddNew_Click

```

```

    DoCmd.GoToRecord , , acNewRec

```

```

Exit_CmdAddNew_Click:
Exit Sub

```

```

Err_CmdAddNew_Click:
MsgBox Err.Description
Resume Exit_CmdAddNew_Click

```

```

End Sub
Private Sub CmdOpenBIOG_MAIN_Click()
On Error GoTo Err_CmdOpenBIOG_MAIN_Click

```

```

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "BIOG_MAIN"
    DoCmd.OpenForm stDocName, , , stLinkCriteria

```

```

Exit_CmdOpenBIOG_MAIN_Click:
Exit Sub

```

```

Err_CmdOpenBIOG_MAIN_Click:
MsgBox Err.Description
Resume Exit_CmdOpenBIOG_MAIN_Click

```

```

End Sub

```

```

Private Sub Form_AfterDelConfirm(STATUS As Integer)
Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

```

```

If STATUS = acDeleteOK Then
    rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.Find "c_flag = " & 1
    rst!c_flag = 0
    rst.Update
    rst.Close

```

```

Else
    rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.Find "c_flag = " & 1
    rst.Delete
    rst.Update
    rst.Close

```

```

End If

```

```

End Sub

```

```

Private Sub Form_Current()

```

```

Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

```

```

If IsNull(c_addr_id.Value) Then
    TxtADDR.Value = ""

```

```

Else
    rst.Open "ADDRESSES", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_addr_id = " & c_addr_id.Value

    TxtADDR.Value = rst.Fields("c_name_chn")
    rst.Close
End If

If IsNull(c_dy.Value) Then
    TxtDynastyCHN.Value = ""
Else
    rst.Open "Dynasties", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_dy = " & c_dy.Value

    TxtDynastyCHN.Value = rst.Fields("c_dynasty_chn")
    rst.Close
End If

If IsNull(c_fy_nh_code.Value) Then
    TxtFYNH.Value = ""
Else
    rst.Open "nian hao", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_nianhao_id = " & c_fy_nh_code.Value

    TxtFYNH.Value = rst.Fields("c_nianhao_chn")
    rst.Close
End If

If IsNull(c_ly_nh_code.Value) Then
    TxtLYNH.Value = ""
Else
    rst.Open "nian hao", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_nianhao_id = " & c_ly_nh_code.Value

    TxtLYNH.Value = rst.Fields("c_nianhao_chn")
    rst.Close
End If

If IsNull(c_source.Value) Then
    TxtTitle_CHN.Value = ""
Else
    rst.Open "TEXT_CODES", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_textid = " & c_source.Value

    TxtTitle_CHN.Value = rst.Fields("c_title_chn")
    rst.Close
End If

End Sub

Private Sub CmdOpenTEXTS_Click()
On Error GoTo Err_CmdOpenTEXTS_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "TEXTS EnterForm"
    DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdOpenTEXTS_Click:
Exit Sub

Err_CmdOpenTEXTS_Click:
MsgBox Err.Description
Resume Exit_CmdOpenTEXTS_Click

End Sub

```

```

Option Compare Database
Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 16) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 16 And Not .EOF
            If !c_form = "SF_EVD" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then
            tLang = 2
        Else
            tLang = 3
        End If
        '
        ' now comes the basic routine
        '
        Me.LblRole.Caption = tLabelLanguage(tLang, 1)
        Me.LblSequence.Caption = tLabelLanguage(tLang, 2)
        Me.LblYear.Caption = tLabelLanguage(tLang, 3)
        Me.LblRange.Caption = tLabelLanguage(tLang, 4)
        Me.LblNHYear.Caption = tLabelLanguage(tLang, 5)
        Me.LblIntercalary.Caption = tLabelLanguage(tLang, 6)
        Me.LblGZ.Caption = tLabelLanguage(tLang, 7)
        Me.LblPages.Caption = tLabelLanguage(tLang, 8)
        Me.LblNotes.Caption = tLabelLanguage(tLang, 9)
        Me.LblEventName.Caption = tLabelLanguage(tLang, 10)
        Me.LblNH.Caption = tLabelLanguage(tLang, 11)
        Me.LblAddr.Caption = tLabelLanguage(tLang, 12)
        Me.LblSource.Caption = tLabelLanguage(tLang, 13)
        ' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 14)
        ' Me.CmdDelete.Caption = tLabelLanguage(tLang, 15)

        ' now for the subform

        'Me.EVENT_ADDR_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
        'Me.EVENT_ADDR_2_Subform.Form.changeDisplayLanguage

    End If
End Sub

```

```

Option Compare Database
Option Explicit
Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 8) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 8 And Not .EOF
            If !c_form = "SF_BID" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then
            tLang = 2
        Else
            tLang = 3
        End If
        '
        ' now comes the basic routine
        '
        Me.c_place_name_Label.Caption = tLabelLanguage(tLang, 1)
        Me.c_place_type_Label.Caption = tLabelLanguage(tLang, 2)
        Me.c_pages_Label.Caption = tLabelLanguage(tLang, 3)
        Me.c_bi_begin_year_Label.Caption = tLabelLanguage(tLang, 4)
        Me.c_bi_end_year_Label.Caption = tLabelLanguage(tLang, 5)
        Me.c_source_Label.Caption = tLabelLanguage(tLang, 6)
        Me.c_notes_Label.Caption = tLabelLanguage(tLang, 7)
    End If
End Sub

```

```

Option Compare Database
Option Explicit
Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 6) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 6 And Not .EOF
            If !c_form = "SF_BSD" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then
            tLang = 2
        Else
            tLang = 3
        End If
        '
        ' now comes the basic routine
        '
        Me.LblNotes.Caption = tLabelLanguage(tLang, 1)
        Me.LblPages.Caption = tLabelLanguage(tLang, 2)
        Me.LblHyperlink.Caption = tLabelLanguage(tLang, 3)
        Me.LblSelfBiography.Caption = tLabelLanguage(tLang, 4)
        Me.LblMainSource.Caption = tLabelLanguage(tLang, 5)
    End If
End Sub

```


Form_frmChoronyms - 1

Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
    MsgBox Err.Description  
    Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
On Error GoTo Err_CmdSelect_Click
```

```
    Forms("frmChoronyms").Visible = False
```

```
Exit_CmdSelect_Click:  
Exit Sub
```

```
Err_CmdSelect_Click:  
    MsgBox Err.Description  
    Resume Exit_CmdSelect_Click
```

```
End Sub
```

Option Compare Database

```
Private Sub CmdOK_Click()  
On Error GoTo Err_CmdOK_Click
```

```
    If Me.Dirty Then Me.Dirty = False  
    DoCmd.Close
```

```
Exit_CmdOK_Click:  
Exit Sub
```

```
Err_CmdOK_Click:  
MsgBox Err.Description  
Resume Exit_CmdOK_Click
```

End Sub

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    If Me.Dirty Then Me.Dirty = False  
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

End Sub

Form_FrmExplainAddRecord - 1

Option Compare Database

```
Private Sub CmdClose_Click()  
On Error GoTo Err_CmdClose_Click
```

```
    If Me.Dirty Then Me.Dirty = False  
    DoCmd.Close
```

```
Exit_CmdClose_Click:  
Exit Sub
```

```
Err_CmdClose_Click:  
MsgBox Err.Description  
Resume Exit_CmdClose_Click
```

```
End Sub
```

Form_frmExplainInput - 1

Option Compare Database

```
Private Sub Command0_Click()  
On Error GoTo Err_Command0_Click
```

```
    DoCmd.Close
```

```
Exit_Command0_Click:  
Exit Sub
```

```
Err_Command0_Click:  
    MsgBox Err.Description  
    Resume Exit_Command0_Click
```

```
End Sub
```

frm_frmGetDataVersion - 1

Option Compare Database

```
Private Sub CmdOK_Click()  
On Error GoTo Err_CmdOK_Click
```

```
    'Dim stDocName As String  
    'Dim stLinkCriteria As String  
  
    'stDocName = "POST_ADDR Subform"  
    'DoCmd.OpenForm stDocName, , , stLinkCriteria  
  
    Me.c_data_version.SetFocus  
    If Len(Me.c_data_version.Text) = 0 Then  
        MsgBox "Please provide a date, eg. 20180412"  
    Else  
        Me.Form.Visible = False  
    End If
```

```
Exit_CmdOK_Click:  
Exit Sub
```

```
Err_CmdOK_Click:  
MsgBox Err.Description  
Resume Exit_CmdOK_Click
```

End Sub

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    If Me.Dirty Then Me.Dirty = False  
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

End Sub

```
Private Sub CmdHelp_Click()  
On Error GoTo Err_CmdHelp_Click
```

```
    MsgBox "The Data Version is the date that is part of the names of the three data files."
```

```
Exit_CmdHelp_Click:  
Exit Sub
```

```
Err_CmdHelp_Click:  
MsgBox Err.Description  
Resume Exit_CmdHelp_Click
```

End Sub

```
Private Sub Form_Open(Cancel As Integer)
```

```
    If Not IsNull(Me.OpenArgs) Then  
        Dim strDataset As String  
        strDataset = Me.OpenArgs  
        Me.c_data_version.SetFocus  
        Me.c_data_version.Value = strDataset  
    End If
```

End Sub

Option Compare Database

```
Private Sub CmdYes_Click()  
On Error GoTo Err_CmdYes_Click
```

```
    c_list.Value = False  
    Me.Visible = False
```

```
Exit_CmdYes_Click:  
Exit Sub
```

```
Err_CmdYes_Click:  
MsgBox Err.Description  
Resume Exit_CmdYes_Click
```

End Sub

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    c_list.Value = True  
    Me.Visible = False
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

End Sub

Option Compare Database

```
Private Sub CmdYes_Click()  
On Error GoTo Err_CmdYes_Click
```

```
    c_list.Value = False  
    Me.Visible = False
```

```
Exit_CmdYes_Click:  
Exit Sub
```

```
Err_CmdYes_Click:  
MsgBox Err.Description  
Resume Exit_CmdYes_Click
```

End Sub

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    c_list.Value = True  
    Me.Visible = False
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

End Sub

frm_frmIndexAddr - 1

Option Compare Database
Public gValue As Long

Private Sub CmdDisableIndexAddress2_Click()
On Error GoTo Err_CmdDisableIndexAddress2_Click

```
If Cmb_Index_addr_2.Enabled Then
    Cmb_Index_addr_2.Enabled = False
    CmdDisableIndexAddress2.Caption = "Enable"
    '
    ' now disable everything below it
    '
    Cmb_Index_addr_3.Enabled = False
    Cmb_Index_addr_4.Enabled = False
    Cmb_Index_addr_5.Enabled = False
    Cmb_Index_addr_6.Enabled = False
    Cmb_Index_addr_7.Enabled = False
    Cmb_Index_addr_8.Enabled = False
    Cmb_Index_addr_9.Enabled = False
    '
    CmdDisableIndexAddress3.Enabled = False
    CmdDisableIndexAddress4.Enabled = False
    CmdDisableIndexAddress5.Enabled = False
    CmdDisableIndexAddress6.Enabled = False
    CmdDisableIndexAddress7.Enabled = False
    CmdDisableIndexAddress8.Enabled = False
    CmdDisableIndexAddress9.Enabled = False
Else
    If Cmb_Index_addr_1.Value = 1000 Then
        MsgBox "Please select an address type code for rank = 1 before enabling rank = 2"
    Else
        Cmb_Index_addr_2.Enabled = True
        CmdDisableIndexAddress2.Caption = "Disable"
        '
        ' re-enable the one below it
        '
        CmdDisableIndexAddress3.Enabled = True
        CmdDisableIndexAddress3.Caption = "Enable"
    End If
End If
```

Exit_CmdDisableIndexAddress2_Click:
Exit Sub

Err_CmdDisableIndexAddress2_Click:
MsgBox Err.Description
Resume Exit_CmdDisableIndexAddress2_Click

End Sub

Private Sub CmdDisableIndexAddress3_Click()

```
If Cmb_Index_addr_3.Enabled Then
    Cmb_Index_addr_3.Enabled = False
    CmdDisableIndexAddress3.Caption = "Enable"
    '
    ' now disable everything below it
    '
    Cmb_Index_addr_4.Enabled = False
    Cmb_Index_addr_5.Enabled = False
    Cmb_Index_addr_6.Enabled = False
    Cmb_Index_addr_7.Enabled = False
    Cmb_Index_addr_8.Enabled = False
    Cmb_Index_addr_9.Enabled = False
    '
    CmdDisableIndexAddress4.Enabled = False
    CmdDisableIndexAddress5.Enabled = False
    CmdDisableIndexAddress6.Enabled = False
    CmdDisableIndexAddress7.Enabled = False
    CmdDisableIndexAddress8.Enabled = False
    CmdDisableIndexAddress9.Enabled = False
Else
    If Cmb_Index_addr_2.Value = 1000 Then
        MsgBox "Please select an address type code for rank = 2 before enabling rank = 3"
    Else
        Cmb_Index_addr_3.Enabled = True
        CmdDisableIndexAddress3.Caption = "Disable"
        '
        ' re-enable the one below it
        '

```


frmIndexAddr - 2

```

    '
    CmdDisableIndexAddress4.Enabled = True
    CmdDisableIndexAddress4.Caption = "Enable"
End If
End If

End Sub

Private Sub CmdDisableIndexAddress4_Click()
    If Cmb_Index_addr_4.Enabled Then
        Cmb_Index_addr_4.Enabled = False
        CmdDisableIndexAddress4.Caption = "Enable"
        '
        ' now disable everything below it
        '
        Cmb_Index_addr_5.Enabled = False
        Cmb_Index_addr_6.Enabled = False
        Cmb_Index_addr_7.Enabled = False
        Cmb_Index_addr_8.Enabled = False
        Cmb_Index_addr_9.Enabled = False
        '
        CmdDisableIndexAddress5.Enabled = False
        CmdDisableIndexAddress6.Enabled = False
        CmdDisableIndexAddress7.Enabled = False
        CmdDisableIndexAddress8.Enabled = False
        CmdDisableIndexAddress9.Enabled = False
    Else
        If Cmb_Index_addr_3.Value = 1000 Then
            MsgBox "Please select an address type code for rank = 3 before enabling rank = 4"
        Else
            Cmb_Index_addr_4.Enabled = True
            CmdDisableIndexAddress4.Caption = "Disable"
            '
            ' re-enable the one below it
            '
            '
            CmdDisableIndexAddress5.Enabled = True
            CmdDisableIndexAddress5.Caption = "Enable"
        End If
    End If
End Sub

Private Sub CmdDisableIndexAddress5_Click()
    If Cmb_Index_addr_5.Enabled Then
        Cmb_Index_addr_5.Enabled = False
        CmdDisableIndexAddress5.Caption = "Enable"
        '
        ' now disable everything below it
        '
        Cmb_Index_addr_6.Enabled = False
        Cmb_Index_addr_7.Enabled = False
        Cmb_Index_addr_8.Enabled = False
        Cmb_Index_addr_9.Enabled = False
        '
        CmdDisableIndexAddress6.Enabled = False
        CmdDisableIndexAddress7.Enabled = False
        CmdDisableIndexAddress8.Enabled = False
        CmdDisableIndexAddress9.Enabled = False
    Else
        If Cmb_Index_addr_4.Value = 1000 Then
            MsgBox "Please select an address type code for rank = 4 before enabling rank = 5"
        Else
            Cmb_Index_addr_5.Enabled = True
            CmdDisableIndexAddress5.Caption = "Disable"
            '
            ' re-enable the one below it
            '
            CmdDisableIndexAddress6.Enabled = True
            CmdDisableIndexAddress6.Caption = "Enable"
        End If
    End If
End Sub

Private Sub CmdDisableIndexAddress6_Click()
    If Cmb_Index_addr_6.Enabled Then
        Cmb_Index_addr_6.Enabled = False
        CmdDisableIndexAddress6.Caption = "Enable"
    '
End If
End Sub
```

frmIndexAddr - 3

```
' now disable everything below it
'
Cmb_Index_addr_7.Enabled = False
Cmb_Index_addr_8.Enabled = False
Cmb_Index_addr_9.Enabled = False
'
CmdDisableIndexAddress7.Enabled = False
CmdDisableIndexAddress8.Enabled = False
CmdDisableIndexAddress9.Enabled = False
Else
If Cmb_Index_addr_5.Value = 1000 Then
    MsgBox "Please select an address type code for rank = 5 before enabling rank = 6"
Else
    Cmb_Index_addr_6.Enabled = True
    CmdDisableIndexAddress6.Caption = "Disable"
    '
    ' re-enable the one below it
    '
    CmdDisableIndexAddress7.Enabled = True
    CmdDisableIndexAddress7.Caption = "Enable"
End If
End If
End Sub

Private Sub CmdDisableIndexAddress7_Click()
If Cmb_Index_addr_7.Enabled Then
    Cmb_Index_addr_7.Enabled = False
    CmdDisableIndexAddress7.Caption = "Enable"
    '
    ' now disable everything below it
    '
    Cmb_Index_addr_8.Enabled = False
    Cmb_Index_addr_9.Enabled = False
    '
    CmdDisableIndexAddress8.Enabled = False
    CmdDisableIndexAddress9.Enabled = False
Else
If Cmb_Index_addr_6.Value = 1000 Then
    MsgBox "Please select an address type code for rank = 6 before enabling rank = 7"
Else
    Cmb_Index_addr_7.Enabled = True
    CmdDisableIndexAddress7.Caption = "Disable"
    '
    ' re-enable the one below it
    '
    CmdDisableIndexAddress8.Enabled = True
    CmdDisableIndexAddress8.Caption = "Enable"
End If
End If
End Sub

Private Sub CmdDisableIndexAddress8_Click()
If Cmb_Index_addr_8.Enabled Then
    Cmb_Index_addr_8.Enabled = False
    CmdDisableIndexAddress8.Caption = "Enable"
    '
    ' now disable everything below it
    '
    Cmb_Index_addr_9.Enabled = False
    '
    CmdDisableIndexAddress9.Enabled = False
Else
If Cmb_Index_addr_7.Value = 1000 Then
    MsgBox "Please select an address type code for rank = 7 before enabling rank = 8"
Else
    Cmb_Index_addr_8.Enabled = True
    CmdDisableIndexAddress8.Caption = "Disable"
    '
    ' re-enable the one below it
    '
    CmdDisableIndexAddress9.Enabled = True
    CmdDisableIndexAddress9.Caption = "Enable"
End If
End If
End Sub

Private Sub CmdDisableIndexAddress9_Click()
```

```

If Cmb_Index_addr_9.Enabled Then
    Cmb_Index_addr_9.Enabled = False
    CmdDisableIndexAddress9.Caption = "Enable"
Else
    If Cmb_Index_addr_8.Value = 1000 Then
        MsgBox "Please select an address type code for rank = 8 before enabling rank = 9"
    Else
        Cmb_Index_addr_9.Enabled = True
        CmdDisableIndexAddress9.Caption = "Disable"
    End If
End If

End Sub

Private Sub CmdReset_Click()
    Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tRecCount As Long
    Dim ti As Integer, tQueryStr As String

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' all I need to do is to use tRecCount to see if there are any changes
    '
    cmdSQL.CommandText = "UPDATE BIOG_ADDR_CODES SET BIOG_ADDR_CODES.c_index_addr_rank = [BIOG_ADDR_CODES].[c_index_addr_default_rank] " + _
        "WHERE (BIOG_ADDR_CODES.c_index_addr_rank <> [BIOG_ADDR_CODES].[c_index_addr_default_rank])"
    cmdSQL.Execute tRecCount
    '
    If tRecCount > 0 Then
        MsgBox "Updating BIOG_MAIN: This will take a while."
        Call UpdateBiogMain
        '
        MsgBox "Updating ZZZ_BIOG_MAIN: This will take a while."
        Call updateZZZ_BIOG_MAIN
        '
        MsgBox "Updating ZZZ_ENTRY_DATA."
        Call updateZZZ_ENTRY_DATA
        '
        MsgBox "Updating ZZZ_STATUS_DATA."
        Call updateZZZ_STATUS_DATA
        '
        MsgBox "Updating ZZZ_POSTED_TO_ADDR_DATA."
        Call updateZZZ_POSTED_TO_ADDR_DATA
        '
        MsgBox "Updating ZZZ_KIN_BIOG_ADDR: This will take a while."
        Call updateZZZ_KIN_BIOG_ADDR
        '
        MsgBox "Updating ZZZ_NONKIN_BIOG_ADDR: This will take a while."
        Call updateZZZ_NONKIN_BIOG_ADDR
        '
        MsgBox "Finished! Please Compact the Database."
    Else
        MsgBox "The current ranking already is the default ranking."
        ' Exit Sub
    End If
    '
    ' finally, reser the form
    '
    ' set the values of the combo boxes
    '
    tQueryStr = "SELECT BIOG_ADDR_CODES.c_addr_type, BIOG_ADDR_CODES.c_index_addr_rank " + _
        "FROM BIOG_ADDR_CODES " + _
        "ORDER BY BIOG_ADDR_CODES.c_index_addr_rank"

    Set tRst = CurrentDb.OpenRecordset(tQueryStr)
    tRst.MoveFirst
    '
    ' since, in theory, the the records are sorted by rank, as soon as it hits 100, the initialization is complete
    '
    ti = 1
    Do While ti < 10
        Select Case ti
            Case 1
                Cmb_Index_addr_1.Value = tRst!c_addr_type
            Case 2
                If tRst!c_index_addr_rank < 10 Then
                    Cmb_Index_addr_2.Value = tRst!c_addr_type
                    ' setting Enabled = False makes the click routine reset it to True
                End If
            End Select
            ti = ti + 1
        Loop
    End Sub

```

```

        Cmb_Index_addr_2.Enabled = False
    Else
        ti = 100
    End If
    Call CmdDisableIndexAddress2_Click
Case 3
    If tRst!c_index_addr_rank < 10 Then
        Cmb_Index_addr_3.Value = tRst!c_addr_type
        Cmb_Index_addr_3.Enabled = False
    Else
        ti = 100
    End If
    Call CmdDisableIndexAddress3_Click
Case 4
    If tRst!c_index_addr_rank < 10 Then
        Cmb_Index_addr_4.Value = tRst!c_addr_type
        Cmb_Index_addr_4.Enabled = False
    Else
        ti = 100
    End If
    Call CmdDisableIndexAddress4_Click
Case 5
    If tRst!c_index_addr_rank < 10 Then
        Cmb_Index_addr_5.Value = tRst!c_addr_type
        Cmb_Index_addr_5.Enabled = False
    Else
        ti = 100
    End If
    Call CmdDisableIndexAddress5_Click
Case 6
    If tRst!c_index_addr_rank < 10 Then
        Cmb_Index_addr_6.Value = tRst!c_addr_type
        Cmb_Index_addr_6.Enabled = False
    Else
        ti = 100
    End If
    Call CmdDisableIndexAddress6_Click
Case 7
    If tRst!c_index_addr_rank < 10 Then
        Cmb_Index_addr_7.Value = tRst!c_addr_type
        Cmb_Index_addr_7.Enabled = False
    Else
        ti = 100
    End If
    Call CmdDisableIndexAddress7_Click
Case 8
    If tRst!c_index_addr_rank < 10 Then
        Cmb_Index_addr_8.Value = tRst!c_addr_type
        Cmb_Index_addr_8.Enabled = False
    Else
        ti = 100
    End If
    Call CmdDisableIndexAddress8_Click
Case 9
    If tRst!c_index_addr_rank < 10 Then
        Cmb_Index_addr_9.Value = tRst!c_addr_type
        Cmb_Index_addr_9.Enabled = False
    Else
        ti = 100
    End If
    Call CmdDisableIndexAddress9_Click
Case Else
    ti = 100
End Select
ti = ti + 1
tRst.MoveNext
Loop
'
' clean up
'
tRst.Close
Set tRst = Nothing

```

End Sub

```

Private Sub CmdUpdate_Click()
    Call SetIndexAddrRanks
End Sub

```

```

Private Sub Form_Open(Cancel As Integer)

```

```

Dim tRst As DAO.Recordset, ti As Integer, tQueryStr As String
'
' set initial values for the combo boxes
'
Cmb_Index_addr_2.Value = 1000
Cmb_Index_addr_3.Value = 1000
Cmb_Index_addr_4.Value = 1000
Cmb_Index_addr_5.Value = 1000
Cmb_Index_addr_6.Value = 1000
Cmb_Index_addr_7.Value = 1000
Cmb_Index_addr_8.Value = 1000
Cmb_Index_addr_9.Value = 1000
'
' next, set the values of the combo boxes from BIOG_ADDR_CODES
'
tQueryStr = "SELECT BIOG_ADDR_CODES.c_addr_type, BIOG_ADDR_CODES.c_index_addr_rank " + _
"FROM BIOG_ADDR_CODES " + _
"ORDER BY BIOG_ADDR_CODES.c_index_addr_rank"

Set tRst = CurrentDb.OpenRecordset(tQueryStr)
tRst.MoveFirst
'
' since, in theory, the the records are sorted by rank, as soon as it hits 100, the initialization is compl
ete
'
ti = 1
Do While ti < 10
    Select Case ti
        Case 1
            Cmb_Index_addr_1.Value = tRst!c_addr_type
        Case 2
            If tRst!c_index_addr_rank < 10 Then
                Cmb_Index_addr_2.Value = tRst!c_addr_type
            Else
                ti = 100
                Call CmdDisableIndexAddress2_Click
            End If
        Case 3
            If tRst!c_index_addr_rank < 10 Then
                Cmb_Index_addr_3.Value = tRst!c_addr_type
            Else
                ti = 100
                Call CmdDisableIndexAddress3_Click
            End If
        Case 4
            If tRst!c_index_addr_rank < 10 Then
                Cmb_Index_addr_4.Value = tRst!c_addr_type
            Else
                ti = 100
                Call CmdDisableIndexAddress4_Click
            End If
        Case 5
            If tRst!c_index_addr_rank < 10 Then
                Cmb_Index_addr_5.Value = tRst!c_addr_type
            Else
                ti = 100
                Call CmdDisableIndexAddress5_Click
            End If
        Case 6
            If tRst!c_index_addr_rank < 10 Then
                Cmb_Index_addr_6.Value = tRst!c_addr_type
            Else
                ti = 100
                Call CmdDisableIndexAddress6_Click
            End If
        Case 7
            If tRst!c_index_addr_rank < 10 Then
                Cmb_Index_addr_7.Value = tRst!c_addr_type
            Else
                ti = 100
                Call CmdDisableIndexAddress7_Click
            End If
        Case 8
            If tRst!c_index_addr_rank < 10 Then
                Cmb_Index_addr_8.Value = tRst!c_addr_type
            Else
                ti = 100
                Call CmdDisableIndexAddress8_Click
            End If
        Case 9
    End Select
    ti = ti + 1

```

frmIndexAddr - 7

```
        If tRst!c_index_addr_rank < 10 Then
            Cmb_Index_addr_9.Value = tRst!c_addr_type
        Else
            Call CmdDisableIndexAddress9_Click
            ti = 100
        End If
    Case Else
        ti = 100
    End Select
    ti = ti + 1
    tRst.MoveNext
Loop
'
' clean up
'
tRst.Close
Set tRst = Nothing
End Sub

Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click

    If Me.Dirty Then Me.Dirty = False
    DoCmd.Close

Exit_CmdCancel_Click:
    Exit Sub

Err_CmdCancel_Click:
    MsgBox Err.Description
    Resume Exit_CmdCancel_Click

End Sub

Private Function Compact_DB(tStrDatabase As String)
    Dim tStrPath As String, tRstLinkInit As DAO.Recordset, tNameLen As Integer

    ' get the current dataset

    Set tRstLinkInit = CurrentDb.OpenRecordset("LinkListInit", dbOpenDynaset)

    tRstLinkInit.MoveFirst
    tStrDataBaseVersion = tRstLinkInit!c_dataset
    tRstLinkInit.Close

    'MsgBox "Beginning"
    tStrPath = CurrentProject.FullName

    'MsgBox tStrPath

    If InStr(UCase(tStrPath), "ADMIN") > 0 Then
        tStrUserType = "ADMIN"
        tNameLen = 13
    Else
        tStrUserType = "User"
        tNameLen = 12
    End If

    tStrPathBase = Left(tStrPath, Len(tStrPath) - tNameLen) + "_" + Trim(tStrDataBaseVersion) + "_DATA"

    'SET PATH
    tStrPath = "C:\MyFiles\dev\"

    'COMPACT CHOSEN DATABASE, TO TEMPORARY DATABASE NAME
    DBEngine.CompactDatabase tStrPathBase + tStrDatabase + ".mdb", tStrPathBase + "TEMP" + tStrDatabase + ".mdb"

    'DELETE OLD DATABASE
    Kill tStrPathBase + tStrDatabase + ".mdb"

    'RENAME TEMPORARY DATABASE TO ORIGINAL NAME
    Name tStrPathBase + "TEMP" + tStrDatabase + ".mdb" As tStrPathBase + tStrDatabase + ".mdb"

End Function

Private Sub SetIndexAddrRanks()
    Dim tRankedCode(9) As Integer, ti As Integer, tj As Integer, tRst As DAO.Recordset, tQueryStr As String, tProceed As Boolean
    Dim tContinue As Integer, cmdSQL As ADODB.Command
    '
    ' first, get the new ranking and check if it is OK
    '
    ' The first combo box is always enabled
```

```

frmIndexAddr - 8
'
tRankedCode(1) = Me.Cmb_Index_addr_1.Value
'
If Cmb_Index_addr_2.Enabled Then
    tRankedCode(2) = Cmb_Index_addr_2.Value
Else
    tRankedCode(2) = 1000
End If
'
If Cmb_Index_addr_3.Enabled Then
    tRankedCode(3) = Cmb_Index_addr_3.Value
Else
    tRankedCode(3) = 1000
End If
'
If Cmb_Index_addr_4.Enabled Then
    tRankedCode(4) = Cmb_Index_addr_4.Value
Else
    tRankedCode(4) = 1000
End If
'
If Cmb_Index_addr_5.Enabled Then
    tRankedCode(5) = Cmb_Index_addr_5.Value
Else
    tRankedCode(5) = 1000
End If
'
If Cmb_Index_addr_6.Enabled Then
    tRankedCode(6) = Cmb_Index_addr_6.Value
Else
    tRankedCode(6) = 1000
End If
'
If Cmb_Index_addr_7.Enabled Then
    tRankedCode(7) = Cmb_Index_addr_7.Value
Else
    tRankedCode(7) = 1000
End If
'
If Cmb_Index_addr_8.Enabled Then
    tRankedCode(8) = Cmb_Index_addr_8.Value
Else
    tRankedCode(8) = 1000
End If
'
If Cmb_Index_addr_9.Enabled Then
    tRankedCode(9) = Cmb_Index_addr_9.Value
Else
    tRankedCode(9) = 1000
End If
'
' next, check duplicated value. The algorithm is brute-force but should be fast anyhow
'
ti = 2
Do While ti < 10
    If tRankedCode(ti) = 1000 Then
        ti = 100
    Else
        tj = 1
        Do While tj < ti
            If tRankedCode(ti) = tRankedCode(tj) Then
                ' warn about the duplication and trigger the end of processing
                MsgBox "The same address type is used in ranking " + Str(tj) + " and " + Str(ti) + ". Please
fix."
                tj = 200
                ti = 100
            End If
            tj = tj + 1
        Loop
    End If
    ti = ti + 1
Loop
'
' check for error
'
If tj = 201 Then
    Exit Sub
End If

```

frmIndexAddr - 9

```
'
' now see if the new ranking matches the current: if so, inform the user and exit
'
' first, get the ranked BIOG_ADDR_CODES records
tQueryStr = "SELECT BIOG_ADDR_CODES.c_addr_type, BIOG_ADDR_CODES.c_index_addr_rank " + _
"FROM BIOG_ADDR_CODES " + _
"ORDER BY BIOG_ADDR_CODES.c_index_addr_rank"

Set tRst = CurrentDb.OpenRecordset(tQueryStr)
tRst.MoveFirst
'
' now compare tRankedCode() with the current setting
'
tProceed = False
ti = 1
Do While ti < 10
    If tRst!c_index_addr_rank < 100 And tRankedCode(ti) = 1000 Then
        tProceed = True
        Exit Do
    ElseIf tRst!c_index_addr_rank = 100 And tRankedCode(ti) < 1000 Then
        tProceed = True
        Exit Do
    ElseIf tRst!c_index_addr_rank = 100 And tRankedCode(ti) = 1000 Then
        '
        ' we've come to the end of both the current and the new list with no changes
        '
        Exit Do
    ElseIf tRst!c_addr_type <> tRankedCode(ti) Then
        tProceed = True
        Exit Do
    End If

    ti = ti + 1
    tRst.MoveNext
Loop
tRst.Close
'
If tProceed Then
    tQueryStr = "This procedure updates many files and will take a long time. Do you wish to continue?"
    tContinue = MsgBox(tQueryStr, vbQuestion + vbYesNo + vbDefaultButton2, "Change Index Address Ranking?")
    If tContinue = vbYes Then
        '
        ' the next step is to update the rankings in BIOG_ADDR_CODES
        '
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        ' first, reset all rankings to 100, and then update the relevent ones
        '
        cmdSQL.CommandText = "UPDATE BIOG_ADDR_CODES SET BIOG_ADDR_CODES.c_index_addr_rank = 100"
        cmdSQL.Execute tRecCount
        '
        ti = 1
        Do While ti < 10
            If tRankedCode(ti) < 1000 Then
                cmdSQL.CommandText = "UPDATE BIOG_ADDR_CODES SET BIOG_ADDR_CODES.c_index_addr_rank = " + Str(
ti) + " " + _
                "WHERE (BIOG_ADDR_CODES.c_addr_type = " + Str(tRankedCode(ti)) + " )"
                cmdSQL.Execute tRecCount
                ti = ti + 1
            Else
                ti = 11
            End If
        Loop
        '
        ' for debugging
        '
        'Exit Sub
        '
        ' Finally, call the procedures to update the clusters of tables for the three files (with the proced
ure to compact the files at the end)
        '
        MsgBox "Updating BIOG_MAIN: This will take a while."
        Call UpdateBiogMain
        'Exit Sub
        '
        MsgBox "Updating ZZZ_BIOG_MAIN: This will take a while."
        Call updateZZZ_BIOG_MAIN
    End If
End If
```



```

        'Exit Sub
    ,
    MsgBox "Updating ZZZ_ENTRY_DATA."
    Call updateZZZ_ENTRY_DATA
    ,
    MsgBox "Updating ZZZ_STATUS_DATA."
    Call updateZZZ_STATUS_DATA
    ,
    MsgBox "Updating ZZZ_POSTED_TO_ADDR_DATA."
    Call updateZZZ_POSTED_TO_ADDR_DATA
    ,
    MsgBox "Updating ZZZ_KIN_BIOG_ADDR: This will take a while."
    Call updateZZZ_KIN_BIOG_ADDR
    ,
    MsgBox "Updating ZZZ_NONKIN_BIOG_ADDR: This will take a while."
    Call updateZZZ_NONKIN_BIOG_ADDR

    MsgBox "Finished! Please Compact the Database."
Else
    MsgBox "Never mind."
End If
Else
    MsgBox "The new ranking matches the current ranking. There is nothing to change."
End If
,
End Sub

Private Sub UpdateBiogMain()
    Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tQueryStr As String, tRecCount As Long

    tQueryStr = "SELECT BIOG_ADDR_CODES.c_addr_type, BIOG_ADDR_CODES.c_index_addr_rank " + _
        "FROM BIOG_ADDR_CODES " + _
        "ORDER BY BIOG_ADDR_CODES.c_index_addr_rank"

    Set tRst = CurrentDb.OpenRecordset(tQueryStr)
    tRst.MoveFirst

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    ' first, delete all the index address values in BIOG_MAIN

    cmdSQL.CommandText = "UPDATE BIOG_MAIN SET BIOG_MAIN.c_index_addr_id = Null, BIOG_MAIN.c_index_addr_type_code"
    = Null"
    cmdSQL.Execute tRecCount

    ' now fill in the new data
    ' note that because c_sequence in BIOG_ADDR_DATA allows one to have multiple values for any address type,
    ' this routine picks the maximum sequence number for any category.

    cmdSQL.CommandText = "DELETE * FROM TMP_BIOG_ADDR_DATA"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "INSERT INTO TMP_BIOG_ADDR_DATA ( c_personid, c_addr_type, c_sequence ) " + _
        "SELECT BIOG_ADDR_DATA.c_personid, BIOG_ADDR_DATA.c_addr_type, Max(BIOG_ADDR_DATA.c_sequence) AS MaxOfc_s"
    equence " + _
        "FROM BIOG_ADDR_DATA " + _
        "GROUP BY BIOG_ADDR_DATA.c_personid, BIOG_ADDR_DATA.c_addr_type"
    cmdSQL.Execute tRecCount

    tQueryStr = "UPDATE (BIOG_MAIN INNER JOIN BIOG_ADDR_DATA " + _
        "ON BIOG_MAIN.c_personid = BIOG_ADDR_DATA.c_personid) INNER JOIN TMP_BIOG_ADDR_DATA " + _
        "ON (BIOG_ADDR_DATA.c_sequence = TMP_BIOG_ADDR_DATA.c_sequence) AND (BIOG_ADDR_DATA.c_addr_type = TMP"
    _BIOG_ADDR_DATA.c_addr_type) " + _
        "AND (BIOG_ADDR_DATA.c_personid = TMP_BIOG_ADDR_DATA.c_personid) " + _
        "SET BIOG_MAIN.c_index_addr_id = [BIOG_ADDR_DATA].[c_addr_id], " + _
        "BIOG_MAIN.c_index_addr_type_code = [BIOG_ADDR_DATA].[c_addr_type] " + _
        "WHERE (BIOG_MAIN.c_index_addr_id Is Null) AND (BIOG_ADDR_DATA.c_addr_type = "

    tQueryStr = "UPDATE BIOG_MAIN INNER JOIN BIOG_ADDR_DATA " + _
        "ON BIOG_MAIN.c_personid = BIOG_ADDR_DATA.c_personid " + _
        "SET BIOG_MAIN.c_index_addr_id = [BIOG_ADDR_DATA].[c_addr_id], " + _
        "BIOG_MAIN.c_index_addr_type_code = [BIOG_ADDR_DATA].[c_addr_type] " + _
        "WHERE (BIOG_MAIN.c_index_addr_id is NULL) AND (BIOG_ADDR_DATA.c_addr_type = "

    ' now the loop

    Do While tRst!c_index_addr_rank < 100 And Not tRst.EOF
        cmdSQL.CommandText = tQueryStr + Str(tRst!c_addr_type) + ")"

```

```

        cmdSQL.Execute tRecCount

        tRst.MoveNext
    Loop

    cmdSQL.CommandText = "DELETE * FROM TMP_BIOG_ADDR_DATA"
    cmdSQL.Execute tRecCount
    '
    ' because the file is linked to tables, it is in use and cannot be compacted (rats)
    ' well, let's see what the cost is in terms of space for running this routine

    'Call Compact_DB("1")

End Sub
Private Sub updateZZZ_BIOG_MAIN()
    Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tQueryStr As String, tRecCount As Long

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    ' strip values from the table
    cmdSQL.CommandText = "UPDATE ZZZ_BIOG_MAIN " + _
        "SET ZZZ_BIOG_MAIN.c_index_addr_id = Null, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_code = Null, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_name = Null, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_chn = Null, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_desc = Null, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_chn = Null, " + _
        "ZZZ_BIOG_MAIN.x_coord = Null, " + _
        "ZZZ_BIOG_MAIN.y_coord = Null"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "UPDATE BIOG_ADDR_CODES INNER JOIN ((BIOG_MAIN INNER JOIN ZZZ_BIOG_MAIN " + _
        "ON BIOG_MAIN.c_personid = ZZZ_BIOG_MAIN.c_personid) INNER JOIN ADDR_CODES ON BIOG_MAIN.c_index_addr_
id = ADDR_CODES.c_addr_id) " + _
        "ON BIOG_ADDR_CODES.c_addr_type = BIOG_MAIN.c_index_addr_type_code " + _
        "SET ZZZ_BIOG_MAIN.c_index_addr_id = [BIOG_MAIN].[c_index_addr_id], " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_code = [BIOG_MAIN].[c_index_addr_type_code], " + _
        "ZZZ_BIOG_MAIN.c_index_addr_name = [ADDR_CODES].[c_name], " + _
        "ZZZ_BIOG_MAIN.c_index_addr_chn = [ADDR_CODES].[c_name_chn], " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_desc = [BIOG_ADDR_CODES].[c_addr_desc], " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_chn = [BIOG_ADDR_CODES].[c_addr_desc_chn], " + _
        "ZZZ_BIOG_MAIN.x_coord = [ADDR_CODES].[x_coord], " + _
        "ZZZ_BIOG_MAIN.y_coord = [ADDR_CODES].[y_coord]"
    cmdSQL.Execute tRecCount

End Sub
Private Sub updateZZZ_ENTRY_DATA()
    Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tRecCount As Long

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    ' strip values from the table
    cmdSQL.CommandText = "UPDATE ZZZ_ENTRY_DATA " + _
        "SET ZZZ_ENTRY_DATA.c_addr_id = Null, " + _
        "ZZZ_ENTRY_DATA.c_addr_type = Null, " + _
        "ZZZ_ENTRY_DATA.c_addr_name = Null, " + _
        "ZZZ_ENTRY_DATA.c_addr_chn = Null, " + _
        "ZZZ_ENTRY_DATA.c_addr_desc = Null, " + _
        "ZZZ_ENTRY_DATA.c_addr_desc_chn = Null, " + _
        "ZZZ_ENTRY_DATA.x_coord = Null, " + _
        "ZZZ_ENTRY_DATA.y_coord = Null"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "UPDATE ZZZ_ENTRY_DATA INNER JOIN ZZZ_BIOG_MAIN ON ZZZ_ENTRY_DATA.c_personid = ZZZ_BIOG_
MAIN.c_personid " + _
        "SET ZZZ_ENTRY_DATA.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
        "ZZZ_ENTRY_DATA.c_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
        "ZZZ_ENTRY_DATA.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
        "ZZZ_ENTRY_DATA.c_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
        "ZZZ_ENTRY_DATA.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
        "ZZZ_ENTRY_DATA.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
        "ZZZ_ENTRY_DATA.x_coord = [ZZZ_BIOG_MAIN].[x_coord], " + _
        "ZZZ_ENTRY_DATA.y_coord = [ZZZ_BIOG_MAIN].[y_coord]"
    cmdSQL.Execute tRecCount

End Sub

```

```
Private Sub updateZZZ_STATUS_DATA()
```

```
Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tRecCount As Long
```

```
Set cmdSQL = New ADODB.Command
```

```
cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
cmdSQL.CommandType = adCmdText
```

```
' strip values from the table
```

```
cmdSQL.CommandText = "UPDATE ZZZ_STATUS_DATA " + _
    "SET ZZZ_STATUS_DATA.c_addr_id = Null, " + _
    "ZZZ_STATUS_DATA.c_addr_type = Null, " + _
    "ZZZ_STATUS_DATA.c_addr_name = Null, " + _
    "ZZZ_STATUS_DATA.c_addr_chn = Null, " + _
    "ZZZ_STATUS_DATA.c_addr_desc = Null, " + _
    "ZZZ_STATUS_DATA.c_addr_desc_chn = Null, " + _
    "ZZZ_STATUS_DATA.x_coord = Null, " + _
    "ZZZ_STATUS_DATA.y_coord = Null"
```

```
cmdSQL.Execute tRecCount
```

```
cmdSQL.CommandText = "UPDATE ZZZ_STATUS_DATA INNER JOIN ZZZ_BIOG_MAIN ON ZZZ_STATUS_DATA.c_personid = ZZZ_BIOG_MAIN.c_personid " + _
```

```
"SET ZZZ_STATUS_DATA.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
    "ZZZ_STATUS_DATA.c_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
    "ZZZ_STATUS_DATA.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
    "ZZZ_STATUS_DATA.c_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
    "ZZZ_STATUS_DATA.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
    "ZZZ_STATUS_DATA.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
    "ZZZ_STATUS_DATA.x_coord = [ZZZ_BIOG_MAIN].[x_coord], " + _
    "ZZZ_STATUS_DATA.y_coord = [ZZZ_BIOG_MAIN].[y_coord]"
```

```
cmdSQL.Execute tRecCount
```

```
End Sub
```

```
Private Sub updateZZZ_POSTED_TO_ADDR_DATA()
```

```
Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tRecCount As Long
```

```
Set cmdSQL = New ADODB.Command
```

```
cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
cmdSQL.CommandType = adCmdText
```

```
' strip values from the table
```

```
cmdSQL.CommandText = "UPDATE ZZZ_POSTED_TO_ADDR_DATA " + _
    "SET ZZZ_POSTED_TO_ADDR_DATA.c_addr_id = Null, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_type = Null, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_name = Null, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_chn = Null, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_desc = Null, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_desc_chn = Null, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.x_coord = Null, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.y_coord = Null"
```

```
cmdSQL.Execute tRecCount
```

```
cmdSQL.CommandText = "UPDATE ZZZ_POSTED_TO_ADDR_DATA INNER JOIN ZZZ_BIOG_MAIN ON ZZZ_POSTED_TO_ADDR_DATA.c_personid = ZZZ_BIOG_MAIN.c_personid " + _
```

```
"SET ZZZ_POSTED_TO_ADDR_DATA.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
    "ZZZ_POSTED_TO_ADDR_DATA.x_coord = [ZZZ_BIOG_MAIN].[x_coord], " + _
    "ZZZ_POSTED_TO_ADDR_DATA.y_coord = [ZZZ_BIOG_MAIN].[y_coord]"
```

```
cmdSQL.Execute tRecCount
```

```
End Sub
```

```
Private Sub updateZZZ_KIN_BIOG_ADDR()
```

```
Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tRecCount As Long
```

```
Set cmdSQL = New ADODB.Command
```

```
cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
cmdSQL.CommandType = adCmdText
```

```
' strip values from the table
```

```
cmdSQL.CommandText = "UPDATE ZZZ_KIN_BIOG_ADDR " + _
    "SET ZZZ_KIN_BIOG_ADDR.c_addr_id = Null, " + _
    "ZZZ_KIN_BIOG_ADDR.c_addr_type = Null, " + _
    "ZZZ_KIN_BIOG_ADDR.c_addr_name = Null, " + _
    "ZZZ_KIN_BIOG_ADDR.c_addr_chn = Null, " + _
    "ZZZ_KIN_BIOG_ADDR.c_addr_desc = Null, " + _
    "ZZZ_KIN_BIOG_ADDR.c_addr_desc_chn = Null, " + _
    "ZZZ_KIN_BIOG_ADDR.x_coord = Null, " + _
```

```

"ZZZ_KIN_BIOG_ADDR.y_coord = Null, " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_id = Null, " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_type = Null, " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_desc = Null, " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_desc_chn = Null, " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_name = Null, " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_chn = Null, " +
"ZZZ_KIN_BIOG_ADDR.node_xcoord = Null, " +
"ZZZ_KIN_BIOG_ADDR.node_ycoord = Null"
cmdSQL.Execute tRecCount
,
cmdSQL.CommandText = "UPDATE ZZZ_KIN_BIOG_ADDR INNER JOIN ZZZ_BIOG_MAIN ON ZZZ_KIN_BIOG_ADDR.c_personid = ZZZ_
_BIOG_MAIN.c_personid " +
"SET ZZZ_KIN_BIOG_ADDR.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " +
"ZZZ_KIN_BIOG_ADDR.c_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " +
"ZZZ_KIN_BIOG_ADDR.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " +
"ZZZ_KIN_BIOG_ADDR.c_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " +
"ZZZ_KIN_BIOG_ADDR.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " +
"ZZZ_KIN_BIOG_ADDR.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " +
"ZZZ_KIN_BIOG_ADDR.x_coord = [ZZZ_BIOG_MAIN].[x_coord], " +
"ZZZ_KIN_BIOG_ADDR.y_coord = [ZZZ_BIOG_MAIN].[y_coord]"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "UPDATE ZZZ_KIN_BIOG_ADDR INNER JOIN ZZZ_BIOG_MAIN ON ZZZ_KIN_BIOG_ADDR.c_node_id = ZZZ_
_BIOG_MAIN.c_personid " +
"SET ZZZ_KIN_BIOG_ADDR.c_node_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " +
"ZZZ_KIN_BIOG_ADDR.c_node_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " +
"ZZZ_KIN_BIOG_ADDR.node_xcoord = [ZZZ_BIOG_MAIN].[x_coord], " +
"ZZZ_KIN_BIOG_ADDR.node_ycoord = [ZZZ_BIOG_MAIN].[y_coord]"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "UPDATE ZZZ_KIN_BIOG_ADDR SET ZZZ_KIN_BIOG_ADDR.c_t_dist = " +
"Sqr((Sin(3.1415926536*(y_coord-node_ycoord)/360))^2+Cos(3.1415926536*y_coord/180)*" +
"Cos(3.1415926536*node_ycoord/180)*(Sin(3.1415926536*(x_coord-node_xcoord)/360))^2) " +
"WHERE (((ZZZ_KIN_BIOG_ADDR.x_coord)>0) AND ((ZZZ_KIN_BIOG_ADDR.y_coord)>0) AND " +
"((ZZZ_KIN_BIOG_ADDR.node_xcoord)>0) AND ((ZZZ_KIN_BIOG_ADDR.node_ycoord)>0));"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "UPDATE ZZZ_KIN_BIOG_ADDR SET ZZZ_KIN_BIOG_ADDR.c_distance = " +
"25484*Atn(c_t_dist/(1+Sqr(1-c_t_dist*c_t_dist))) " +
"WHERE (((ZZZ_KIN_BIOG_ADDR.x_coord)>0) AND ((ZZZ_KIN_BIOG_ADDR.y_coord)>0) AND " +
"((ZZZ_KIN_BIOG_ADDR.node_xcoord)>0) AND ((ZZZ_KIN_BIOG_ADDR.node_ycoord)>0))"
cmdSQL.Execute tRecCount

End Sub
Private Sub updateZZZ_NONKIN_BIOG_ADDR()
Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tRecCount As Long

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

' strip values from the table
cmdSQL.CommandText = "UPDATE ZZZ_NONKIN_BIOG_ADDR " +
"SET ZZZ_NONKIN_BIOG_ADDR.c_addr_id = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_addr_type = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_addr_name = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_addr_chn = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_addr_desc = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_addr_desc_chn = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.x_coord = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.y_coord = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_type = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc_chn = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_name = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_chn = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.node_xcoord = Null, " +
"ZZZ_NONKIN_BIOG_ADDR.node_ycoord = Null"
cmdSQL.Execute tRecCount
,
cmdSQL.CommandText = "UPDATE ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZZ_BIOG_MAIN ON ZZZ_NONKIN_BIOG_ADDR.c_personid
= ZZZ_BIOG_MAIN.c_personid " +

```

```

"SET ZZZ_NONKIN_BIOG_ADDR.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
"ZZZ_NONKIN_BIOG_ADDR.x_coord = [ZZZ_BIOG_MAIN].[x_coord], " + _
"ZZZ_NONKIN_BIOG_ADDR.y_coord = [ZZZ_BIOG_MAIN].[y_coord]"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "UPDATE ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZZ_BIOG_MAIN ON ZZZ_NONKIN_BIOG_ADDR.c_node_id
= ZZZ_BIOG_MAIN.c_personid " + _
"SET ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
"ZZZ_NONKIN_BIOG_ADDR.node_xcoord = [ZZZ_BIOG_MAIN].[x_coord], " + _
"ZZZ_NONKIN_BIOG_ADDR.node_ycoord = [ZZZ_BIOG_MAIN].[y_coord]"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "UPDATE ZZZ_NONKIN_BIOG_ADDR SET ZZZ_NONKIN_BIOG_ADDR.c_t_dist = " + _
"Sqr((Sin(3.1415926536*(y_coord-node_ycoord)/360))^2+Cos(3.1415926536*y_coord/180)*" + _
"Cos(3.1415926536*node_ycoord/180)*(Sin(3.1415926536*(x_coord-node_xcoord)/360))^2) " + _
"WHERE ((ZZZ_NONKIN_BIOG_ADDR.x_coord)>0) AND ((ZZZ_NONKIN_BIOG_ADDR.y_coord)>0) AND " + _
"((ZZZ_NONKIN_BIOG_ADDR.node_xcoord)>0) AND ((ZZZ_NONKIN_BIOG_ADDR.node_ycoord)>0));"

cmdSQL.Execute tRecCount

cmdSQL.CommandText = "UPDATE ZZZ_NONKIN_BIOG_ADDR SET ZZZ_NONKIN_BIOG_ADDR.c_distance = " + _
"25484*Atn(c_t_dist/(1+Sqr(1-c_t_dist*c_t_dist))) " + _
"WHERE ((ZZZ_NONKIN_BIOG_ADDR.x_coord)>0) AND ((ZZZ_NONKIN_BIOG_ADDR.y_coord)>0) AND " + _
"((ZZZ_NONKIN_BIOG_ADDR.node_xcoord)>0) AND ((ZZZ_NONKIN_BIOG_ADDR.node_ycoord)>0)"

cmdSQL.Execute tRecCount
End Sub

```

frm_frmKinReductionWarning -1

```
Option Compare Database
Private Sub Command7_Click()
On Error GoTo Err_Command7_Click
```

```
    If Me.Dirty Then Me.Dirty = False
    DoCmd.Close
```

```
Exit_Command7_Click:
    Exit Sub
```

```
Err_Command7_Click:
    MsgBox Err.Description
    Resume Exit_Command7_Click
```

```
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
    Dim tLabelLanguage(3, 3) As String, tLang As Integer
    Dim tRstLabelList As DAO.Recordset, ti As Integer

    gLCID = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    If gLCID = 2052 Or gLCID = 3076 Then      ' 2052 = PRC, 3076 = Hong Kong
        gDisplayLanguage = "S"
    ElseIf gLCID = 4100 Or gLCID = 1028 Then  ' 4100 = Singapore, 1028 = Taiwan
        gDisplayLanguage = "T"
    Else
        gDisplayLanguage = "E"
    End If

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 3 And Not .EOF
            If !c_form = "KRW" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then
            tLang = 2
        Else
            tLang = 3
        End If
        '
        ' now comes the basic routine
        '
        Me.Label1.Caption = tLabelLanguage(tLang, 1)
        Me.Label2.Caption = tLabelLanguage(tLang, 2)
    End If
End Sub
```

Option Compare Database

```
Private Sub CmdRun_Click()
On Error GoTo Err_CmdRun_Click
```

```
' This program will dump a file to a .net file
'
' The form allows the user to define as many as 20 attributes
'
' for the moment I'll just describe the format of the .gdf file
'
' *Vertices NUM
' ID label "box" ic [color] bc [color]
'   ID = str(c_person_id)
'   label = c_name_chn
'   color = red (1), orange (2), yellow (3), green (4), blue (5)
'
' *Edges
' node1 node2 1 1 "label"
'   node1 = str(c_person_id) for node1
'   node2 = str(c_node_id) for node2
'   color = red (1), orange (2), yellow (3), green (4), blue (5)
'   label = c_link_desc
```

```
If IsNull(Me.TxtTableName.Value) Then
    MsgBox "Please provide a table to save."
    GoTo Exit_CmdRun_Click
End If
```

```
If IsNull(Me.TxtN1_ID.Value) Then
    MsgBox "Please provide a field for the first node ID."
    GoTo Exit_CmdRun_Click
End If
```

```
If IsNull(Me.TxtN1_Lbl.Value) Then
    MsgBox "Please provide a field for the first node label."
    GoTo Exit_CmdRun_Click
End If
```

```
If IsNull(Me.TxtN2_ID.Value) Then
    MsgBox "Please provide a field for the second node ID."
    GoTo Exit_CmdRun_Click
End If
```

```
If IsNull(Me.TxtN2_Lbl.Value) Then
    MsgBox "Please provide a field for the second node label."
    GoTo Exit_CmdRun_Click
End If
```

```
Dim dlgSaveAs As FileDialog, tFileNum As Integer, tFileName As String, tFN As Variant
Dim tRstNodeList As DAO.Recordset, tRstEdgeList As DAO.Recordset, tRstSource As DAO.Recordset
```

```
Dim tStr As String, tC As String, ti As Integer, tQuote As String, tFindStr As String
Dim tColor(20) As String, tStrNode1 As String, tStrNode2 As String, tCodeStr As String
Dim tN1_ID As String, tN1_Label As String, tN1_LblColor As String, tN1_Shape As String, tN1_BorderColor As String
Dim tN1_InnerColor As String, tN2_ID As String, tN2_Label As String, tN2_LblColor As String, tN2_Shape As String
Dim tN2_BorderColor As String, tN2_InnerColor As String, tEdge_Label As String, tEdge_LblColor As String
Dim tEdge_Weight As String, tEdge_Color As String, tAdd As Boolean
```

```
' first see if there are any records to process
```

```
Set tRstSource = CurrentDb.OpenRecordset(TxtTableName.Value, dbOpenDynaset)
If tRstSource.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdRun_Click
End If
```

```
' next get a file
```

```
' to write to a UTF-8 file, use the ADO stream object
```

```
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8.net"
ElseIf CodeFrame.Value = 2 Then
    tStream.Charset = "big5"
    tCodeStr = "BIG5.net"
```

```

Else
    tStream.Charset = "gb2312"
    tCodeStr = "GB2312.net"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network_" + tCodeStr
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdRun_Click
        End If
        '
        ' zap and open the scratch file
        '
        Dim cmdSQL As ADODB.Command
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
        cmdSQL.Execute tRecDeleted
        '
        Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
        tRstNodeList.Index = "c_ID"
        '
        '
        ' set the Quote delimiter
        '
        tQuote = Chr(34)
        tC = Chr(44) ' the comma
        '
        ' now get all the field names
        '
        tN1_ID = Me.TxtN1_ID.Value
        tN1_Label = Me.TxtN1_Lbl.Value

        If IsNull(Me.TxtN1_LblColor.Value) Then
            tN1_LblColor = ""
        Else
            tN1_LblColor = Me.TxtN1_LblColor.Value
        End If

        If IsNull(Me.TxtN1_InnerColor.Value) Then
            tN1_InnerColor = ""
        Else
            tN1_InnerColor = Me.TxtN1_InnerColor.Value
        End If

        If IsNull(Me.TxtN1_BorderColor.Value) Then
            tN1_BorderColor = ""
        Else
            tN1_BorderColor = Me.TxtN1_BorderColor.Value
        End If

        If IsNull(Me.TxtN1_Shape.Value) Then
            tN1_Shape = ""
        Else
            tN1_Shape = Me.TxtN1_Shape.Value
        End If

        tN2_ID = Me.TxtN2_ID.Value
        tN2_Label = Me.TxtN2_Lbl.Value
    End With

```



```

If IsNull(Me.TxtN2_LblColor.Value) Then
    tN2_LblColor = ""
Else
    tN2_LblColor = Me.TxtN2_LblColor.Value
End If

If IsNull(Me.TxtN2_InnerColor.Value) Then
    tN2_InnerColor = ""
Else
    tN2_InnerColor = Me.TxtN2_InnerColor.Value
End If

If IsNull(Me.TxtN2_BorderColor.Value) Then
    tN2_BorderColor = ""
Else
    tN2_BorderColor = Me.TxtN2_BorderColor.Value
End If

If IsNull(Me.TxtN2_Shape.Value) Then
    tN2_Shape = ""
Else
    tN2_Shape = Me.TxtN2_Shape.Value
End If

If IsNull(Me.TxtEdge_Color.Value) Then
    tEdge_Color = ""
Else
    tEdge_Color = Me.TxtEdge_Color.Value
End If

If IsNull(Me.TxtEdge_Lbl.Value) Then
    tEdge_Label = ""
Else
    tEdge_Label = Me.TxtEdge_Lbl.Value
End If

If IsNull(Me.TxtEdge_LblColor.Value) Then
    tEdge_LblColor = ""
Else
    tEdge_LblColor = Me.TxtEdge_LblColor.Value
End If

If IsNull(Me.TxtEdge_Weight) Then
    tEdge_Weight = ""
Else
    tEdge_Weight = Me.TxtEdge_Weight
End If
'
' process the table for nodes
'
ti = 1
With tRstSource
    .MoveFirst
    Do While Not .EOF
        '
        ' check to see if we have the first node
        '
        If tRstNodeList.RecordCount = 0 Then
            tAdd = True
        Else
            tRstNodeList.Seek "=", Str(.Fields(tN1_ID))
            tAdd = tRstNodeList.NoMatch
        End If

        If tAdd Then
            '
            ' add the node to the list
            '
            tRstNodeList.AddNew
            tRstNodeList!c_v_sort = ti
            tRstNodeList!c_v_num = Str(ti)
            tRstNodeList!c_ID = .Fields(tN1_ID)
            tRstNodeList!c_lbl = .Fields(tN1_Label)
            If Not (tN1_LblColor = "") Then
                tRstNodeList!c_lc = .Fields(tN1_LblColor)
            End If
            If Not (tN1_InnerColor = "") Then
                tRstNodeList!c_ic = .Fields(tN1_InnerColor)
            End If
            If Not (tN1_BorderColor = "") Then

```

```

        tRstNodeList!c_bc = .Fields(tN1_BorderColor)
    End If
    If Not (tN1_Shape = "") Then
        tRstNodeList!c_shape = .Fields(tN1_Shape)
    End If
    tRstNodeList.Update
    ti = ti + 1
End If
'
'
'   check to see if we have the second node
'
If tRstNodeList.RecordCount = 0 Then
    tAdd = True
Else
    tRstNodeList.Seek "=", Str(.Fields(tN2_ID))
    tAdd = tRstNodeList.NoMatch
End If

If tAdd Then
    '
    '   add the node to the list
    '
    tRstNodeList.AddNew
    tRstNodeList!c_v_sort = ti
    tRstNodeList!c_v_num = Str(ti)
    tRstNodeList!c_ID = .Fields(tN2_ID)
    tRstNodeList!c_lbl = .Fields(tN2_Label)
    If Not (tN2_LblColor = "") Then
        tRstNodeList!c_lc = .Fields(tN2_LblColor)
    End If
    If Not (tN2_InnerColor = "") Then
        tRstNodeList!c_ic = .Fields(tN2_InnerColor)
    End If
    If Not (tN2_BorderColor = "") Then
        tRstNodeList!c_bc = .Fields(tN2_BorderColor)
    End If
    If Not (tN2_Shape = "") Then
        tRstNodeList!c_shape = .Fields(tN2_Shape)
    End If
    tRstNodeList.Update
    ti = ti + 1
End If
'
.MoveNext
Loop
End With
'
'   now write the nodes to file
'
tStr = "*Vertices " + Trim(Str(tRstNodeList.RecordCount))
tStream.WriteText tStr, adWriteLine
'
With tRstNodeList
    .Index = "c_v_sort"
    .MoveFirst
    Do While Not .EOF
        tStr = !c_v_num + " "
        '
        '   label (required)
        '
        tStr = tStr + Chr(34) + Trim(!c_lbl) + Chr(34)
        '
        '   shape (required)
        '
        If IsNull(!c_shape) Then
            tStr = tStr + " box "
        Else
            tStr = tStr + " " + !c_shape
        End If
        '
        '   other parameters
        '
        If Not IsNull(!c_lc) Then
            tStr = " lc " + !c_lc
        End If
        '
        If Not IsNull(!c_ic) Then
            tStr = tStr + " ic " + !c_ic
        End If
    End If

```

```

    '
    If Not IsNull(!c_bc) Then
        tStr = tStr + " bc " + !c_bc
    End If
    '
    tStream.WriteText tStr, adWriteLine
    .MoveNext
Loop
.Index = "c_ID"
End With

'
' now the edges: define the record structure
'
tStream.WriteText "*Edges", adWriteLine

With tRstSource
.MoveFirst
Do While Not .EOF
    '
    ' find the vertex number of the first node
    '
    tRstNodeList.Seek "=", Str(.Fields(tN1_ID))
    If Not tRstNodeList.NoMatch Then
        '
        tStrNode1 = tRstNodeList!c_v_num
        '
        ' find the vertex number of the second node
        '
        tRstNodeList.Seek "=", Str(.Fields(tN2_ID))
        If Not tRstNodeList.NoMatch Then
            '
            ' see if an edge already exists
            '
            tStrNode2 = tRstNodeList!c_v_num

            tStr = tStrNode1 + " " + tStrNode2 + " 1"
            '
            ' now get the weight
            '
            If Not (tEdge_Weight = "") Then
                tStr = tStr + " w " + Trim(Str(.Fields(tEdge_Weight)))
            End If
            '
            If Not (tEdge_Color = "") Then
                tStr = tStr + " c " + .Fields(tEdge_Color)
            End If
            '
            ' now get the label
            '
            If Not (tEdge_Label = "") Then
                tStr = tStr + " l " + tQuote + .Fields(tEdge_Label) + tQuote
            End If

            If Not (tEdge_LblColor = "") Then
                tStr = tStr + " lc " + tQuote + .Fields(tEdge_LblColor) + tQuote
            End If
            '
            tStream.WriteText tStr, adWriteLine
        End If
    End If
    .MoveNext
Loop
End With

'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tRstNodeList.Close

tStream.Close
Set tStream = Nothing
'
Set tRstSource = Nothing
Set tGDF = Nothing
Set tFileSystem = Nothing
Set tRstNodeList = Nothing
Else

```

```
        'The user pressed Cancel.
    End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing
```

```
Exit_CmdRun_Click:
Exit Sub
```

```
Err_CmdRun_Click:
MsgBox Err.Description
Resume Exit_CmdRun_Click
```

```
End Sub
```

```
Private Sub CmdHelp_Click()
On Error GoTo Err_CmdHelp_Click
```

```
    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "frmPajekHelp"
    DoCmd.OpenForm stDocName, , , stLinkCriteria
```

```
Exit_CmdHelp_Click:
Exit Sub
```

```
Err_CmdHelp_Click:
MsgBox Err.Description
Resume Exit_CmdHelp_Click
```

```
End Sub
```

Option Compare Database
Public gFirstTime As Integer

```
Private Sub Form_Current()
    Dim tRst As DAO.Recordset

    If gFirstTime = 0 Then
        gFirstTime = 1
    Else
        Set tRst = Forms!CBDB_Browser_2!BIOG_MAIN_2_Subform.Form.Recordset
        tRst.FindFirst "c_personid = " & Str(c_personid.Value)
        If tRst.NoMatch Then
            gPersonID = 0
            Forms!CBDB_Browser_2.Form.CmdSaveToFile.Enabled = False
        Else
            gPersonID = c_personid.Value
            Forms!CBDB_Browser_2!BIOG_MAIN_2_Subform.Form.Refresh
            Call getKinship(c_personid.Value)
            Forms!CBDB_Browser_2.Form.CmdSaveToFile.Enabled = True
        End If
    End If
End Sub
```

Private Sub getKinship(t_personid As Long)

```
' this routine searches for the immediate kin of the current person
'
Dim tMaxUp As Integer, tMaxDown As Integer, tMaxCol As Integer, tMaxMarr As Integer
Dim tTrue As Integer, tFalse As Integer, tLoopCount As Long, tErrorStr As String
Dim tContinue As Integer, tAddrID As Long, tExitDo As Boolean, tRecCount As Long, tRecDelete As Long
Dim tRstDummy As DAO.Recordset, tAppendQuery As QueryDef
Dim tSeekStr As String, tLoopMax As Long, tLoopInfoStr As String, tKinQueryStr As String, tQueryStr As String
Dim tNodeDistQueryStr As String, tPruneTmpQueryDuplesStr As String, tPruneTmpQuery As String
Dim tPruneInversesQueryStr1 As String, tPruneTmpInversesQueryStr1 As String, tPruneInversesQueryStr2 As String
Dim tAppendQueryStr As String, tPruneTmpInversesQueryStr2 As String
Dim tKinFirstQueryStr As String, tPruneTmpQueryDuplesStr2 As String, tPruneTmpQuery2 As String

tTrue = -1
tFalse = 0

tLoopMax = 10
tMaxUp = 2
tMaxDown = 2
tMaxCol = 1
tMaxMarr = 1

Dim KinQuery As DAO.QueryDef
Dim prm As DAO.Parameter
Dim cmdSQL As ADODB.Command, tRecDeleted As Long, strSQL As String

' Clear the tables
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

' clear the working files
'
cmdSQL.CommandText = "Delete * from ZZ_KIN_LIST"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "Delete * from ZZ_KIN_LIST_TMP"
cmdSQL.Execute tRecCount

' now zap the ego-relative form person file
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_KIN"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_KINNET"
cmdSQL.Execute tRecDeleted

'
' this copies the people on the import list (which is just the selected person if one does not use a list)
'
tQueryStr = "INSERT INTO ZZ_KIN_LIST ( c_personid, c_kin_id, c_kinrel, c_kinrel_total, c_kinrel_total_raw, c_kinrel_total_simplified, " + _
    "c_kin_code, c_up_total, c_down_total, c_mar_total, c_col_total, c_distance, c_up, c_down, c_mar, c_col, " + _
    "c_prior_female, c_kin_female, c_kin_sex, c_female, c_sex, c_personid_root ) " + _
```

```
"SELECT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_personid AS c_kin_id, " + _
    "'ego' AS c_kin_rel, 'ego' AS c_kin_rel_total, 'ego' AS c_kin_rel_total_raw, 'ego' AS c_kin_rel_total_simplified, -3 AS c_kin_code, " + _
    "0 AS c_up_total, 0 AS c_down_total, 0 AS c_mar_total, 0 AS c_col_total, 0 AS c_distance, " + _
    "0 AS c_up, 0 AS c_down, 0 AS c_mar, 0 AS c_col, ZZZ_BIOG_MAIN.c_female AS c_prior_female, ZZZ_BIOG_MAIN.c_female AS c_kin_female, " + _
    "iif(ZZZ_BIOG_MAIN.c_female, 'F', 'M'), ZZZ_BIOG_MAIN.c_female, iif(ZZZ_BIOG_MAIN.c_female, 'F', 'M'), ZZZ_BIOG_MAIN.c_personid AS c_personid_root " + _
    "FROM ZZZ_BIOG_MAIN WHERE ZZZ_BIOG_MAIN.c_personid = " + Str(t_personid)
```

' the initial list of "ego" roots is now initialized in ZZ_KIN_LIST, and the personID is stored as c_personid_root: use this to create ZZ_KIN_LIST_TMP
' in the first query, one begins to build out with a first layer of kinship relations
' as the first layer, we put the kin_rel as both the kin_rel and the kin_rel_total

```
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
```

' the new logic is to not test the metrics until after the reduction routine is run
' the reduction routine will remove the first layer of kin who do not meet the 2-2-1-1 test criterion; these will need to be added back in at the end

```
tKinFirstQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_personid, c_kin_id, c_kinrel, c_kinrel_total, " + _
    "c_kinrel_total_raw, c_kinrel_total_simplified, c_kin_code, c_up_total, c_down_total, c_mar_total, c_col_total, c_distance, " + _
    "c_up, c_down, c_mar, c_col, c_personid_root, c_prior_female, c_kin_female, c_kin_sex, c_female, c_sex, c_notes, " + _
    "c_source, c_source_text_chn, c_source_text ) " + _
    "SELECT DISTINCT ZZZ_KIN_BIOG_ADDR.c_personid, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_desc, " + _
    "KINSHIP_CODES.c_kinrel_simplified AS c_kinrel_total, ZZZ_KIN_BIOG_ADDR.c_link_desc AS c_kinrel_total_raw, " + _
    "KINSHIP_CODES.c_kinrel_simplified AS c_kinrel_total_simplified, " + _
    "ZZZ_KIN_BIOG_ADDR.c_link_code, ZZZ_KIN_BIOG_ADDR.c_upstep AS c_up_total, ZZZ_KIN_BIOG_ADDR.c_dwnstep AS c_down_total, " + _
    "ZZZ_KIN_BIOG_ADDR.c_marstep AS c_mar_total, ZZZ_KIN_BIOG_ADDR.c_colstep AS c_col_total, 0 AS c_distance, " + _
    "ZZZ_KIN_BIOG_ADDR.c_upstep, ZZZ_KIN_BIOG_ADDR.c_dwnstep, ZZZ_KIN_BIOG_ADDR.c_marstep, ZZZ_KIN_BIOG_ADDR.c_colstep, " + _
    "ZZ_KIN_LIST.c_personid_root, ZZ_KIN_LIST.c_female AS c_prior_female, ZZZ_KIN_BIOG_ADDR.c_node_female, iif(ZZZ_KIN_BIOG_ADDR.c_node_female, 'F', 'M'), " + _
    "ZZZ_KIN_BIOG_ADDR.c_female, iif(ZZZ_KIN_BIOG_ADDR.c_female, 'F', 'M'), " + _
    "'Notes: ' + ZZZ_KIN_BIOG_ADDR.c_person_name_chn + ' > ' + ZZZ_KIN_BIOG_ADDR.c_node_chn + ' (' + ZZZ_KIN_BIOG_ADDR.c_link_desc + ') ' AS c_notes, " + _
    "ZZZ_KIN_BIOG_ADDR.c_source, ZZZ_KIN_BIOG_ADDR.c_title_chn, ZZZ_KIN_BIOG_ADDR.c_title " + _
    "FROM (ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_KIN_LIST ON ZZZ_KIN_BIOG_ADDR.c_personid = ZZ_KIN_LIST.c_kin_id) " + _
    "INNER JOIN KINSHIP_CODES ON ZZZ_KIN_BIOG_ADDR.c_link_code = KINSHIP_CODES.c_kincode"
```

' each subsequent layer adds the new kin_rel to the kin_rel_total and the total cumulative steps are summed

```
tKinQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_personid, c_kin_id, c_kinrel, c_kinrel_total_simplified, c_kinrel_total, " + _
    "c_kinrel_total_raw, c_kin_code, c_up_total, c_down_total, c_mar_total, c_col_total, c_distance, " + _
    "c_up, c_down, c_mar, c_col, c_personid_root, c_prior_female, c_kin_female, c_kin_sex, c_female, c_sex, c_notes, " + _
    "c_source, c_source_text_chn, c_source_text) " + _
    "SELECT DISTINCT ZZZ_KIN_BIOG_ADDR.c_personid, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_desc, " + _
    "[ZZ_KIN_LIST].[c_kinrel_total_simplified]+[KINSHIP_CODES].[c_kinrel_simplified] AS c_kinrel_total_simplified, " + _
    "[ZZ_KIN_LIST].[c_kinrel_total]+[KINSHIP_CODES].[c_kinrel_simplified] AS c_kinrel_total, " + _
    "ZZ_KIN_LIST.c_kinrel_total_raw+ZZZ_KIN_BIOG_ADDR.c_link_desc AS c_kinrel_total_raw, " + _
    "ZZZ_KIN_BIOG_ADDR.c_link_code, ZZZ_KIN_BIOG_ADDR.c_upstep+ZZ_KIN_LIST.c_up_total AS c_up_total, " + _
    "ZZZ_KIN_BIOG_ADDR.c_dwnstep+ZZ_KIN_LIST.c_down_total AS c_down_total, " + _
    "ZZZ_KIN_BIOG_ADDR.c_marstep+ZZ_KIN_LIST.c_mar_total AS c_mar_total, " + _
    "ZZZ_KIN_BIOG_ADDR.c_colstep+ZZ_KIN_LIST.c_col_total AS c_col_total, " + _
    "ZZ_KIN_LIST.c_distance, ZZZ_KIN_BIOG_ADDR.c_upstep, ZZZ_KIN_BIOG_ADDR.c_dwnstep, ZZZ_KIN_BIOG_ADDR.c_marstep, ZZZ_KIN_BIOG_ADDR.c_colstep, " + _
    "ZZ_KIN_LIST.c_personid_root, ZZ_KIN_LIST.c_female AS c_prior_female, ZZZ_KIN_BIOG_ADDR.c_node_female, iif(ZZZ_KIN_BIOG_ADDR.c_node_female, 'F', 'M'), " + _
    "ZZZ_KIN_BIOG_ADDR.c_female, iif(ZZZ_KIN_BIOG_ADDR.c_female, 'F', 'M'), " + _
    "ZZ_KIN_LIST.c_notes + ' > ' + ZZZ_KIN_BIOG_ADDR.c_node_chn + ' (' + ZZZ_KIN_BIOG_ADDR.c_link_desc + ') ' AS c_notes, " + _
    "ZZZ_KIN_BIOG_ADDR.c_source, ZZZ_KIN_BIOG_ADDR.c_title_chn, ZZZ_KIN_BIOG_ADDR.c_title " + _
    "FROM (ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_KIN_LIST ON ZZZ_KIN_BIOG_ADDR.c_personid = ZZ_KIN_LIST.c_kin_id) " + _
    "INNER JOIN KINSHIP_CODES ON ZZZ_KIN_BIOG_ADDR.c_link_code = KINSHIP_CODES.c_kincode " + _
    "WHERE ((ZZ_KIN_LIST.c_distance)=0"
```



```

"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair1])) OR " + _
"((ZZ_KIN_LIST_TMP.c_distance)=[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " + _
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair1]) AND " + _
"((ZZ_KIN_LIST_TMP.c_personid)<[ZZ_KIN_LIST_TMP_1].[c_personid])) OR " + _
"((ZZ_KIN_LIST_TMP.c_distance)<[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " + _
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair2])) OR " + _
"((ZZ_KIN_LIST_TMP.c_distance)=[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " + _
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair2]) AND " + _
"((ZZ_KIN_LIST_TMP.c_personid)<[ZZ_KIN_LIST_TMP_1].[c_personid]))"

```

```

tAppendQueryStr = "INSERT INTO ZZ_KIN_LIST ( c_personid, c_kin_id, c_kin_code, c_personid_root, c_kinrel, " + _
_
_
" c_kinrel_total, c_kinrel_total_raw, c_kinrel_total_simplified, c_up, c_down, c_col, c_mar, c_up_total, c_
_
_
down_total, c_col_total, " + _
" c_mar_total, c_distance, c_female, c_sex, c_kin_female, c_kin_sex, c_prior_female, c_notes, c_source, c_
_
source_text_chn, c_source_text ) " + _
"SELECT DISTINCT ZZ_KIN_LIST_TMP.c_personid, ZZ_KIN_LIST_TMP.c_kin_id, ZZ_KIN_LIST_TMP.c_kin_code, " + _
_
"ZZ_KIN_LIST_TMP.c_personid_root, ZZ_KIN_LIST_TMP.c_kinrel, ZZ_KIN_LIST_TMP.c_kinrel_total, " + _
_
"ZZ_KIN_LIST_TMP.c_kinrel_total_raw, ZZ_KIN_LIST_TMP.c_kinrel_total_simplified, " + _
_
"ZZ_KIN_LIST_TMP.c_up, ZZ_KIN_LIST_TMP.c_down, ZZ_KIN_LIST_TMP.c_col, ZZ_KIN_LIST_TMP.c_mar, " + _
_
"ZZ_KIN_LIST_TMP.c_up_total, ZZ_KIN_LIST_TMP.c_down_total, ZZ_KIN_LIST_TMP.c_col_total, " + _
_
"ZZ_KIN_LIST_TMP.c_mar_total, ZZ_KIN_LIST_TMP.c_distance, ZZ_KIN_LIST_TMP.c_female, ZZ_KIN_LIST_TMP.c_
_
sex, " + _
_
"ZZ_KIN_LIST_TMP.c_kin_female, ZZ_KIN_LIST_TMP.c_kin_sex, ZZ_KIN_LIST_TMP.c_prior_female, ZZ_KIN_LIST
_
TMP.c_notes, " + _
_
"ZZ_KIN_LIST_TMP.c_source, ZZ_KIN_LIST_TMP.c_source_text_chn, ZZ_KIN_LIST_TMP.c_source_text " + _
_
"FROM ZZ_KIN_LIST_TMP"

```

```

tLoopCount = 1
tExitDo = False

```

```

Do While tLoopCount <= tLoopMax And tRecCount > 0

```

```

    If tLoopCount = 1 Then

```

```

        ' MsgBox "Running first query"

```

```

        cmdSQL.CommandText = tKinFirstQueryStr

```

```

    Else

```

```

        ' MsgBox "Running query"

```

```

        cmdSQL.CommandText = tKinQueryStr + Str(tLoopCount - 1) + ")"

```

```

    End If

```

```

    cmdSQL.Execute tRecCount

```

```

    If tRecCount > 0 Then

```

```

        '
        ' process the results for addition
        '

```

```

        ' update the distance
        '

```

```

        'MsgBox "Fixing node distance"

```

```

        cmdSQL.CommandText = tNodeDistQueryStr

```

```

        cmdSQL.Execute tRecDelete

```

```

        '
        ' then mark the duplicates and delete them
        '

```

```

        'MsgBox "Fixing dupes 1"

```

```

        cmdSQL.CommandText = tPruneTmpQuery

```

```

        cmdSQL.Execute tRecDelete

```

```

        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"

```

```

        cmdSQL.Execute tRecDelete

```

```

        'MsgBox "Fixing dupes 2"

```

```

        cmdSQL.CommandText = tPruneTmpQuery2

```

```

        cmdSQL.Execute tRecDelete

```

```

        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"

```

```

        cmdSQL.Execute tRecDelete

```

```

        'MsgBox "Fixing dupes 3"

```

```

        cmdSQL.CommandText = tPruneTmpQueryDuplesStr

```

```

        cmdSQL.Execute tRecDelete

```

```

        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"

```

```

        cmdSQL.Execute tRecDelete

```

```

        'MsgBox "Fixing dupes 4"

```

```

        cmdSQL.CommandText = tPruneTmpQueryDuplesStr2

```

```

        cmdSQL.Execute tRecDelete

```

```

        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"

```

```

        cmdSQL.Execute tRecDelete

```

```

        'MsgBox "Fixing inverses"

```

```

        cmdSQL.CommandText = tPruneTmpInversesQueryStr1

```

```

        cmdSQL.Execute tRecDelete

```



```

cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = tPruneInversesQueryStr1
cmdSQL.Execute tRecDelete
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = tPruneInversesQueryStr2
cmdSQL.Execute tRecDelete
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDelete
'
' now simplify the kinship string, add to the total and reduce if possible
'
'
' Reduce kinship strings
'
' first just get the string length

cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP SET ZZ_KIN_LIST_TMP.c_kinrel_len = Len([ZZ_KIN_LIST_TMP].[c_kinrel_total])"
cmdSQL.Execute tRecDelete
'
' then deal with len = 2
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP " +
"SET ZZ_KIN_LIST_TMP.c_kinrel_root_text = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-2), " +
"ZZ_KIN_LIST_TMP.c_kinrel_test_text = Right([ZZ_KIN_LIST_TMP].[c_kinrel_total],2), " +
"ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total_simplified],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-2)" +
"WHERE (([ZZ_KIN_LIST_TMP.c_kinrel_len]=2))"
cmdSQL.Execute tRecDelete
'
' replace where relevant

cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " +
"ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " +
"SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZ_KINREL_REDUCTION].[c_kinrel_replacement], " +
"ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified +
" + " + "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ')', " +
"ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " +
" + "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ') ', " +
"ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total]+[ZZ_KINREL_REDUCTION].[c_up_change], " +
"ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total]+[ZZ_KINREL_REDUCTION].[c_down_change], " +
"ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total]+[ZZ_KINREL_REDUCTION].[c_col_change], " +
"ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total]+[ZZ_KINREL_REDUCTION].[c_mar_change] " +
"WHERE ((([ZZ_KINREL_REDUCTION.c_kinrel_target] Is Not Null AND [ZZ_KINREL_REDUCTION.c_required] = 1))"
cmdSQL.Execute tRecDelete
'
' then deal with len > 2
'
' copy the target string and string root

cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP " +
"SET ZZ_KIN_LIST_TMP.c_kinrel_root_text = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-2), " +
"ZZ_KIN_LIST_TMP.c_kinrel_test_text = Right([ZZ_KIN_LIST_TMP].[c_kinrel_total],2), " +
"ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total_simplified],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-2)" +
"WHERE ((([ZZ_KIN_LIST_TMP.c_kinrel_len]>2))"
cmdSQL.Execute tRecDelete
'
' replace where relevant

cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " +
"ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " +
"SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZ_KINREL_REDUCTION].[c_kinrel_replacement], " +
"ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified +

```

```

" + _
                "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_
replacement] + ')', " + _
                "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " + _
                "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_r
replacement] + ') ', " + _
                "ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total]+[ZZZ_KINREL_REDUCTION].[c_up_cha
nge], " + _
                "ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total]+[ZZZ_KINREL_REDUCTION].[c_do
wn_change], " + _
                "ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total]+[ZZZ_KINREL_REDUCTION].[c_col
change], " + _
                "ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total]+[ZZZ_KINREL_REDUCTION].[c_mar
change] " + _
                "WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null AND ZZZ_KINREL_REDUCTION.c_required )
)"
cmdSQL.Execute tRecDelete
'
' now mark the records with bad metrics
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP SET ZZ_KIN_LIST_TMP.c_delete = 1 " + _
    "WHERE (((ZZ_KIN_LIST_TMP.c_up_total)>" + Str(tMaxUp) + ")) OR " + _
    "(((ZZ_KIN_LIST_TMP.c_down_total)>" + Str(tMaxDown) + ")) OR " + _
    "(((ZZ_KIN_LIST_TMP.c_col_total)>" + Str(tMaxCol) + ")) OR " + _
    "(((ZZ_KIN_LIST_TMP.c_mar_total)>" + Str(tMaxMarr) + "))"
cmdSQL.Execute tRecDelete
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDelete
'
' one final test: the only difference between ZZ_KIN_LIST_TEMP records is in PRIOR_FEMALE
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 INNER JOIN ZZ_KIN_LIST_TMP " + _
    "ON (ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_kin_id) AND " + _
    "(ZZ_KIN_LIST_TMP_1.c_personid = ZZ_KIN_LIST_TMP.c_personid) AND " + _
    "(ZZ_KIN_LIST_TMP_1.c_kin_code = ZZ_KIN_LIST_TMP.c_kin_code) " + _
    "SET ZZ_KIN_LIST_TMP.c_delete = 1 " + _
    "WHERE (((ZZ_KIN_LIST_TMP.c_prior_female)=True) AND ((ZZ_KIN_LIST_TMP_1.c_prior_female)=False))"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' it turns out that getting rid of inverse records is tougher than I would like, so we try one last
time
'
cmdSQL.CommandText = tPruneTmpInversesQueryStr2
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' one last pair of clean-up routines is necessary. One can arrive at the same results through differ
ent paths
'
' first, take the shorter path
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 ON " + _
    "(ZZ_KIN_LIST_TMP.c_personid_root = ZZ_KIN_LIST_TMP_1.c_personid_root) " + _
    "AND (ZZ_KIN_LIST_TMP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id) AND (ZZ_KIN_LIST_TMP.c_personid = ZZ
_KIN_LIST_TMP_1.c_personid) " + _
    "AND (ZZ_KIN_LIST_TMP.c_kinrel = ZZ_KIN_LIST_TMP_1.c_kinrel) " + _
    "SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " + _
    "WHERE (((Len([ZZ_KIN_LIST_TMP].[c_notes]))<Len([ZZ_KIN_LIST_TMP_1].[c_notes])))"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
'MsgBox "Last step"
'
' then take the string with the smaller value
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 ON " + _
    "(ZZ_KIN_LIST_TMP.c_personid_root = ZZ_KIN_LIST_TMP_1.c_personid_root) " + _
    "AND (ZZ_KIN_LIST_TMP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id) AND (ZZ_KIN_LIST_TMP.c_personid = ZZ
_KIN_LIST_TMP_1.c_personid) " + _
    "AND (ZZ_KIN_LIST_TMP.c_kinrel = ZZ_KIN_LIST_TMP_1.c_kinrel) " + _
    "SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " + _
    "WHERE ('X'+[ZZ_KIN_LIST_TMP].[c_notes] > 'X'+[ZZ_KIN_LIST_TMP_1].[c_notes]))"
cmdSQL.Execute tRecDelete

```

```

'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' now append the results:  if no records are added, this should stop the looping
'
' MsgBox "Copying to ZZ_KIN_LIST"
cmdSQL.CommandText = tAppendQueryStr
cmdSQL.Execute tRecCount
'
' and clear ZZ_KIN_LIST_TMP
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP"
cmdSQL.Execute tRecDelete
'
End If

tLoopCount = tLoopCount + 1

If tLoopCount > tLoopMax Then
    MsgBox "Loop limit hit."
    tExitDo = True
    Exit Do
End If

Loop
'
' clean up the results
'
' MsgBox "Fixing ZZ_KIN_LIST inverses"
cmdSQL.CommandText = tAppendQueryStr
cmdSQL.Execute tRecDelete
'
' there is no need to copy to the kinship table
'
' insert

'tQueryStr = "INSERT INTO ZZ_SCRATCH_KINNET ( c_person_id, c_kin_id, c_kin_code, c_kin_rel, c_source, c_source_text_chn, c_source_text) " + _
"SELECT DISTINCT ZZ_KIN_LIST.c_personid, ZZ_KIN_LIST.c_kin_id, ZZ_KIN_LIST.c_kin_code, ZZ_KIN_LIST.c_kin_rel, ZZ_KIN_LIST.c_source, " + _
"ZZ_KIN_LIST.c_source_text_chn, ZZ_KIN_LIST.c_source_text " + _
"FROM ZZ_KIN_LIST WHERE (((ZZ_KIN_LIST.c_personid)<>[ZZ_KIN_LIST].[c_kin_id]))"

'cmdSQL.CommandText = tQueryStr
'cmdSQL.Execute tRecDelete
'
' update the person
'
'tQueryStr = "UPDATE ZZ_SCRATCH_KINNET INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KINNET.c_person_id = ZZZ_BIOG_MAIN.c_personid " + _
"SET ZZ_SCRATCH_KINNET.c_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SCRATCH_KINNET.c_name_chn = [ZZZ_BIOG_MAIN].[c_name_chn], " + _
"ZZ_SCRATCH_KINNET.c_female = [ZZZ_BIOG_MAIN].[c_female], ZZ_SCRATCH_KINNET.c_sex = IIf([ZZZ_BIOG_MAIN].[c_female], 'F', 'M'), " + _
"ZZ_SCRATCH_KINNET.c_index_year = [ZZZ_BIOG_MAIN].[c_index_year], " + _
"ZZ_SCRATCH_KINNET.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
"ZZ_SCRATCH_KINNET.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
"ZZ_SCRATCH_KINNET.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
"ZZ_SCRATCH_KINNET.x_coord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SCRATCH_KINNET.y_coord = [ZZZ_BIOG_MAIN].[y_coord], " + _
"ZZ_SCRATCH_KINNET.c_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
"ZZ_SCRATCH_KINNET.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
"ZZ_SCRATCH_KINNET.c_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn]"

'cmdSQL.CommandText = tQueryStr
'cmdSQL.Execute tRecDelete
'
' update the kin
'
'tQueryStr = "UPDATE ZZ_SCRATCH_KINNET INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KINNET.c_kin_id = ZZZ_BIOG_MAIN.c_personid " + _
"SET ZZ_SCRATCH_KINNET.c_kin_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SCRATCH_KINNET.c_kin_chn = [ZZZ_BIOG_MAIN].[c_name_chn], " + _
"ZZ_SCRATCH_KINNET.c_kin_female = [ZZZ_BIOG_MAIN].[c_female], ZZ_SCRATCH_KINNET.c_kin_sex = IIf([ZZZ_BIOG_MAIN].[c_female], 'F', 'M'), " + _
"ZZ_SCRATCH_KINNET.c_kin_index_year = [ZZZ_BIOG_MAIN].[c_index_year], " + _
"ZZ_SCRATCH_KINNET.c_kin_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
"ZZ_SCRATCH_KINNET.c_kin_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
"ZZ_SCRATCH_KINNET.c_kin_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
"ZZ_SCRATCH_KINNET.kin_x_coord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SCRATCH_KINNET.kin_y_coord = [ZZZ_BIOG_MAIN].[y_coord]"

```

```

ZZZ_BIOG_MAIN].[y_coord], " +
    "ZZ_SCRATCH_KINNET.c_kin_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " +
    "ZZ_SCRATCH_KINNET.c_kin_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " +
    "ZZ_SCRATCH_KINNET.c_kin_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn]"
'cmdSQL.CommandText = tQueryStr
'cmdSQL.Execute tRecDelete
'
' update the relation
'
'tQueryStr = "UPDATE ZZ_SCRATCH_KINNET INNER JOIN ZZZ_KIN_BIOG_ADDR ON (ZZ_SCRATCH_KINNET.c_kin_code = ZZZ_KI
N_BIOG_ADDR.c_link_code) AND " +
    "(ZZ_SCRATCH_KINNET.c_kin_id = ZZZ_KIN_BIOG_ADDR.c_node_id) AND (ZZ_SCRATCH_KINNET.c_person_i
d = ZZZ_KIN_BIOG_ADDR.c_personid) " +
    "SET ZZ_SCRATCH_KINNET.c_notes = ZZZ_KIN_BIOG_ADDR.c_notes, " +
    "ZZ_SCRATCH_KINNET.c_upstep = ZZZ_KIN_BIOG_ADDR.c_upstep, ZZ_SCRATCH_KINNET.c_dwnstep = ZZZ_K
IN_BIOG_ADDR.c_dwnstep, " +
    "ZZ_SCRATCH_KINNET.c_marstep = ZZZ_KIN_BIOG_ADDR.c_marstep, ZZ_SCRATCH_KINNET.c_colstep = ZZZ
_KIN_BIOG_ADDR.c_colstep, " +
    "ZZ_SCRATCH_KINNET.c_distance = ZZZ_KIN_BIOG_ADDR.c_distance "
'cmdSQL.CommandText = tQueryStr
'cmdSQL.Execute tRecDelete
'
'
' the final step is to add the index year descriptive information
'
'cmdSQL.CommandText = "UPDATE (ZZ_SCRATCH_KINNET INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KINNET.c_person_id =
ZZZ_BIOG_MAIN.c_personid) " +
    "INNER JOIN ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 ON ZZ_SCRATCH_KINNET.c_kin_id = ZZZ_BIOG_MAIN_1.c_personid "
+
    "SET ZZ_SCRATCH_KINNET.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " +
    "ZZ_SCRATCH_KINNET.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " +
    "ZZ_SCRATCH_KINNET.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " +
    "ZZ_SCRATCH_KINNET.c_kin_index_year_type_code = [ZZZ_BIOG_MAIN_1].[c_index_year_type_code], " +
    "ZZ_SCRATCH_KINNET.c_kin_index_year_type_desc = [ZZZ_BIOG_MAIN_1].[c_index_year_type_desc], " +
    "ZZ_SCRATCH_KINNET.c_kin_index_year_type_hz = [ZZZ_BIOG_MAIN_1].[c_index_year_type_hz]"
'cmdSQL.Execute tRecDelete
'
' copy to the ego-relative kinship table
'
' Before copying we need to clean up the data
'
' There is a bug in the algorithm that creates the occasional null value in c_kinrel_total. To debug, for t
he moment plug the hole
'
'MsgBox "Patching NULL bug"
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST SET ZZ_KIN_LIST.c_kinrel_total_simplified = '[Program Error]' WHERE
((ZZ_KIN_LIST.c_kinrel_total_simplified) Is Null)"
'cmdSQL.Execute tRecDelete
'
'MsgBox "Inserting ego-relative"
tQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_personid, c_kin_id, c_kinrel, c_kinrel_total, c_kinrel_total_sim
plified, c_up, c_down, c_col, c_mar, " +
    "c_notes, c_kin_code, c_source, c_source_text, c_source_text_chn ) " +
    "SELECT DISTINCT ZZ_KIN_LIST.c_personid_root, ZZ_KIN_LIST.c_kin_id, ZZ_KIN_LIST.c_kinrel_total_raw, ZZ_KI
N_LIST.c_kinrel_total, " +
    "ZZ_KIN_LIST.c_kinrel_total_simplified, ZZ_KIN_LIST.c_up_total, ZZ_KIN_LIST.c_down_total, ZZ_KIN_LIST
.c_col_total, ZZ_KIN_LIST.c_mar_total, " +
    "ZZ_KIN_LIST.c_notes, 0 AS c_kin_code, ZZ_KIN_LIST.c_source, ZZ_KIN_LIST.c_source_text, ZZ_KIN_LIST.c
_source_text_chn " +
    "FROM ZZ_KIN_LIST"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete

' first just get the string length

cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP SET ZZ_KIN_LIST_TMP.c_kinrel_len = Len([ZZ_KIN_LIST_TMP].[c_kinr
el_total_simplified])"
cmdSQL.Execute tRecDelete

' delete the longer strings (this may solve most of the problems)

cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 ON ZZ_KIN_LIST_T
MP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id " +
    "SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " +
    "WHERE ([ZZ_KIN_LIST_TMP_1].[c_kinrel_len]>[ZZ_KIN_LIST_TMP].[c_kinrel_len]))"
cmdSQL.Execute tRecDelete

cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDelete

```

```

'
' the next version uses the string-compare function because sometimes the strings are of the same length
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 INNER JOIN " + _
    "ZZ_KIN_LIST_TMP ON (ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_kin_id) " + _
    "SET ZZ_KIN_LIST_TMP.c_delete = 1 " + _
    "WHERE (((StrComp([ZZ_KIN_LIST_TMP].[c_kinrel_total_simplified], [ZZ_KIN_LIST_TMP_1].[c_kinrel_total_simp" + _
    "lified])) > 0))"
cmdSQL.Execute tRecDelete

cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' the last version uses the total kinship path: take the shortest value
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 INNER JOIN " + _
    "ZZ_KIN_LIST_TMP ON (ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_kin_id) " + _
    "SET ZZ_KIN_LIST_TMP.c_delete = 1 " + _
    "WHERE ((([ZZ_KIN_LIST_TMP].[c_down] + [ZZ_KIN_LIST_TMP].[c_col] + [ZZ_KIN_LIST_TMP].[c_mar] + [ZZ_KIN_LIS" + _
    "T_TMP].[c_up]>" + [ZZ_KIN_LIST_TMP_1].[c_down]+[ZZ_KIN_LIST_TMP_1].[c_col]+[ZZ_KIN_LIST_TMP_1].[c_mar]+[ZZ_KIN_LIS" + _
    "T_TMP_1].[c_up])))"
cmdSQL.Execute tRecDelete

cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' one last clean-up: remove results that are the same but takes a longer path to get there
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 " + _
    "ON (ZZ_KIN_LIST_TMP.c_kinrel = ZZ_KIN_LIST_TMP_1.c_kinrel) AND (ZZ_KIN_LIST_TMP.c_personid = ZZ_KIN_LIST" + _
    "TMP_1.c_personid) AND " + _
    "(ZZ_KIN_LIST_TMP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id) " + _
    "SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " + _
    "WHERE ((Len([ZZ_KIN_LIST_TMP].[c_notes])<Len([ZZ_KIN_LIST_TMP_1].[c_notes])))"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
'MsgBox "Last step"
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 " + _
    "ON (ZZ_KIN_LIST_TMP.c_kinrel = ZZ_KIN_LIST_TMP_1.c_kinrel) AND (ZZ_KIN_LIST_TMP.c_personid = ZZ_KIN_LIST" + _
    "TMP_1.c_personid) AND " + _
    "(ZZ_KIN_LIST_TMP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id) " + _
    "SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " + _
    "WHERE ('X'+[ZZ_KIN_LIST_TMP].[c_notes] > 'X'+[ZZ_KIN_LIST_TMP_1].[c_notes]))"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_KIN ( c_person_id, c_kin_id, c_kin_rel, c_kin_rel_total, c_kin_rel_0, c_u" + _
    "p, c_down, c_collateral, c_marriage, " + _
    "c_notes, c_source, c_source_text, c_source_text_chn ) " + _
    "SELECT DISTINCT ZZ_KIN_LIST_TMP.c_personid, ZZ_KIN_LIST_TMP.c_kin_id, ZZ_KIN_LIST_TMP.c_kinrel, ZZ_KIN_L" + _
    "IST_TMP.c_kinrel_total, " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified, ZZ_KIN_LIST_TMP.c_up, ZZ_KIN_LIST_TMP.c_down, ZZ_KIN_LIST" + _
    "TMP.c_col, ZZ_KIN_LIST_TMP.c_mar, " + _
    "ZZ_KIN_LIST_TMP.c_notes, ZZ_KIN_LIST_TMP.c_source, ZZ_KIN_LIST_TMP.c_source_text, ZZ_KIN_LIST_TMP.c" + _
    "source_text_chn " + _
    "FROM ZZ_KIN_LIST_TMP"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete
'
' add back in the kin not captures by the 2-2-1-1 parameters before updating the information
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_KIN ( c_person_id, c_kin_id, c_kin_code, c_kin_rel, c_kin_rel_0," + _
    "c_kin_rel_total, c_up, c_down, " + _
    "c_marriage, c_collateral ) " + _
    "SELECT ZZZ_KIN_BIOG_ADDR.c_personid, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_code, ZZZ_KIN" + _
    "BIOG_ADDR.c_link_chn, " + _
    "ZZZ_KIN_BIOG_ADDR.c_link_desc AS c_kel_rel_0, ZZZ_KIN_BIOG_ADDR.c_link_desc, ZZZ_KIN_BIOG_ADDR.c_ups" + _
    "tstep, ZZZ_KIN_BIOG_ADDR.c_dwnstep, " + _
    "ZZZ_KIN_BIOG_ADDR.c_marstep, ZZZ_KIN_BIOG_ADDR.c_colstep " + _
    "FROM ZZZ_KIN_BIOG_ADDR " + _
    "WHERE (ZZZ_KIN_BIOG_ADDR.c_personid = " + Str(t_personid) + " AND (ZZZ_KIN_BIOG_ADDR.c_upstep > 2 " + _
    "OR ZZZ_KIN_BIOG_ADDR.c_dwnstep > 2 OR ZZZ_KIN_BIOG_ADDR.c_marstep > 1 OR ZZZ_KIN_BIOG_ADDR.c_colstep"

```

```

> 1 ) )"
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_KIN ( c_person_id, c_kin_id, c_kin_code, c_kin_rel, c_kin_rel_0,
c_kin_rel_total, c_up, " + _
    "c_down, c_marriage, c_collateral, c_source, c_pages, c_notes, c_source_text_chn, c_source_text ) " + _
"SELECT ZZZ_KIN_BIOG_ADDR.c_personid, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_code, ZZZ_KIN_BIOG_ADDR.c_link_chn, " + _
    "ZZZ_KIN_BIOG_ADDR.c_link_desc AS c_kel_rel_0, ZZZ_KIN_BIOG_ADDR.c_link_desc, ZZZ_KIN_BIOG_ADDR.c_upstep, ZZZ_KIN_BIOG_ADDR.c_dwnstep, " + _
    "ZZZ_KIN_BIOG_ADDR.c_marstep, ZZZ_KIN_BIOG_ADDR.c_colstep, ZZZ_KIN_BIOG_ADDR.c_source, ZZZ_KIN_BIOG_ADDR.c_pages, ZZZ_KIN_BIOG_ADDR.c_notes, " + _
    "ZZZ_KIN_BIOG_ADDR.c_title_chn, ZZZ_KIN_BIOG_ADDR.c_title " + _
"FROM ZZZ_KIN_BIOG_ADDR " + _
"WHERE (((ZZZ_KIN_BIOG_ADDR.c_personid)= " + Str(t_personid) + " ) AND ((ZZZ_KIN_BIOG_ADDR.c_upstep)>2))
OR " + _
    "(((ZZZ_KIN_BIOG_ADDR.c_personid)= " + Str(t_personid) + " ) AND ((ZZZ_KIN_BIOG_ADDR.c_dwnstep)>2)) O
R " + _
    "(((ZZZ_KIN_BIOG_ADDR.c_personid)= " + Str(t_personid) + " ) AND ((ZZZ_KIN_BIOG_ADDR.c_marstep)>1)) O
R " + _
    "(((ZZZ_KIN_BIOG_ADDR.c_personid)= " + Str(t_personid) + " ) AND ((ZZZ_KIN_BIOG_ADDR.c_colstep)>1));"

cmdSQL.Execute tRecDelete

tQueryStr = "UPDATE (ZZZ_BIOG_MAIN INNER JOIN ZZ_SCRATCH_KIN ON ZZZ_BIOG_MAIN.c_personid = ZZ_SCRATCH_KIN.c_p
erson_id) " + _
    "INNER JOIN ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 ON ZZ_SCRATCH_KIN.c_kin_id = ZZZ_BIOG_MAIN_1.c_personid " + _
"SET ZZ_SCRATCH_KIN.c_kin_name = [ZZZ_BIOG_MAIN_1].[c_name], ZZ_SCRATCH_KIN.c_kin_chn = [ZZZ_BIOG_MAIN_1]
.[c_name_chn], " + _
    "ZZ_SCRATCH_KIN.c_kin_index_year = [ZZZ_BIOG_MAIN_1].[c_index_year], ZZ_SCRATCH_KIN.c_kin_female = [Z
ZZ_BIOG_MAIN_1].[c_female], " + _
    "ZZ_SCRATCH_KIN.c_kin_sex = iif([ZZZ_BIOG_MAIN_1].[c_female],'F','M'), ZZ_SCRATCH_KIN.c_kin_code = 0,
" + _
    "ZZ_SCRATCH_KIN.c_kin_addr_id = [ZZZ_BIOG_MAIN_1].[c_index_addr_id], ZZ_SCRATCH_KIN.c_kin_addr_name =
[ZZZ_BIOG_MAIN_1].[c_index_addr_name], " + _
    "ZZ_SCRATCH_KIN.c_kin_addr_chn = [ZZZ_BIOG_MAIN_1].[c_index_addr_chn], ZZ_SCRATCH_KIN.c_kin_addr_type
= [ZZZ_BIOG_MAIN_1].[c_index_addr_type_code], " + _
    "ZZ_SCRATCH_KIN.c_kin_addr_desc = [ZZZ_BIOG_MAIN_1].[c_index_addr_type_desc], " + _
    "ZZ_SCRATCH_KIN.c_kin_addr_desc_chn = [ZZZ_BIOG_MAIN_1].[c_index_addr_type_chn], " + _
    "ZZ_SCRATCH_KIN.kin_x_coord = [ZZZ_BIOG_MAIN_1].[x_coord], " + _
    "ZZ_SCRATCH_KIN.kin_y_coord = [ZZZ_BIOG_MAIN_1].[y_coord]"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete
'
' get the index year descriptive data
'
cmdSQL.CommandText = "UPDATE (ZZ_SCRATCH_KIN INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KIN.c_person_id = ZZZ_BIO
G_MAIN.c_personid) " + _
    "INNER JOIN ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 ON ZZ_SCRATCH_KIN.c_kin_id = ZZZ_BIOG_MAIN_1.c_personid " + _
"SET ZZ_SCRATCH_KIN.c_kin_index_year_type_code = [ZZZ_BIOG_MAIN_1].[c_index_year_type_code], " + _
    "ZZ_SCRATCH_KIN.c_kin_index_year_type_desc = [ZZZ_BIOG_MAIN_1].[c_index_year_type_desc], " + _
    "ZZ_SCRATCH_KIN.c_kin_index_year_type_hz = [ZZZ_BIOG_MAIN_1].[c_index_year_type_hz]"
cmdSQL.Execute tRecDelete
'
' update the dynasty information
'
cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_KIN INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KIN.c_kin_id = ZZZ_BIOG_MA
IN.c_personid " + _
    "SET ZZ_SCRATCH_KIN.c_kin_dy = [ZZZ_BIOG_MAIN].[c_dy], " + _
    "ZZ_SCRATCH_KIN.c_kin_dynasty = [ZZZ_BIOG_MAIN].[c_dynasty], " + _
    "ZZ_SCRATCH_KIN.c_kin_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn]"
cmdSQL.Execute tRecDelete

Exit_getKinship:
Exit Sub

Err_getKinship:
MsgBox Err.Description + tErrorStr
Resume Exit_getKinship

Return
End Sub

```

Form_frmPersonSearch - 1

Option Compare Database

Option Explicit

Option Compare Database

```
Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
Exit Sub
```

```
Err_CmdCancel_Click:
MsgBox Err.Description
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdFilter_Click()
Dim tStrFilterPY As String, tStrFilterChn As String, tStrFilter As String, tStrLen As String
```

```
tStrFilter = ""
```

```
'Me.TxtFilterChn.SetFocus
If TxtFilterChn.Value <> "" Then
tStrFilterChn = Trim(TxtFilterChn.Value)
tStrLen = Str(LenB(tStrFilterChn))
tStrFilter = "LeftB(c_name_chn," + tStrLen + ") = '" + tStrFilterChn + "'"
```

```
Else
'TxtFilterPY.SetFocus
If TxtFilterPY.Value <> "" Then
tStrFilterPY = Trim(TxtFilterPY.Value)
tStrLen = Str(Len(tStrFilterPY))
tStrFilter = "Left(c_name," + tStrLen + ") = '" + tStrFilterPY + "'"
End If
End If
```

```
If tStrFilter <> "" Then
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
```

```
cmdSQL.CommandText = "Delete * from ZZ_ADDRESSES"
cmdSQL.Execute tRecDeleted
```

```
'MsgBox tStrSQL
```

```
cmdSQL.CommandText = "INSERT INTO ZZ_ADDRESSES SELECT ZZZ_ADDR_BELONGS.* FROM ZZZ_ADDR_BELONGS " + _
"WHERE ((" + tStrFilter + "))"
cmdSQL.Execute tRecDeleted
```

```
Dim rsAddr As DAO.Recordset
Set rsAddr = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
Set frmADDRESSES.Form.Recordset = rsAddr
```

```
End If
```

```
CmdFilterClear.Enabled = True
CmdSelectAllFiltered.Enabled = True
```

```
End Sub
Private Sub CmdFilterClear_Click()
Dim rsAddr As DAO.Recordset
Set rsAddr = CurrentDb.OpenRecordset("ZZZ_ADDR_BELONGS", dbOpenDynaset)
Set frmADDRESSES.Form.Recordset = rsAddr
```

```
CmdFilterClear.Enabled = False
CmdSelectAllFiltered.Enabled = False
```

```
End Sub
```

```
Private Sub CmdSelect_Click()
TxtAddrFilter.Value = False
frmADDRESSES.SetFocus
Forms!frmPickADDRESSES.Visible = False
End Sub
```

```
Private Sub CmdSelectAllFiltered_Click()
'
' simply close the form: the receiving form will do the look-up
'
```


frm_frmPickADDRESSES - 2

```
TxtAddrFilter.Value = True
Forms!frmPickADDRESSES.Visible = False
```

End Sub

```
Private Sub Form_Open(Cancel As Integer)
    Dim cmdSQL As ADODB.Command
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    cmdSQL.CommandText = "Delete * from ZZ_ADDRESSES"
    cmdSQL.Execute tRecDeleted

    frmADDRESSES.Form.OrderBy = "c_name,c_name_chn,c_firstyear"
    frmADDRESSES.Form.OrderByOn = True
```

```
If Not IsNull(Me.OpenArgs) Then
    Dim strADDR As String, rsAddr As DAO.Recordset
    strADDR = Me.OpenArgs
```

```
    Set rsAddr = frmADDRESSES.Form.Recordset
    rsAddr.FindFirst "c_addr_id = " & strADDR
```

```
End If
CmdSelectAllFiltered.Enabled = False
```

End Sub

```
Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click
```

```
    Dim tStrSearch As String, tStrSearchStr As String
```

```
    tStrSearchStr = ""
```

```
Me.TxtSearchChn.SetFocus
```

```
If TxtSearchChn.Text <> "" Then
    tStrSearch = Trim(TxtSearchChn.Text)
    tStrSearchStr = "c_name_chn = '" & tStrSearch & "'"
```

```
Else
    TxtSearchPY.SetFocus
    If TxtSearchPY.Text <> "" Then
        tStrSearch = Trim(TxtSearchPY.Text)
        tStrSearchStr = "c_name = '" & tStrSearch & "'"
    End If
End If
```

```
If tStrSearchStr <> "" Then
    Dim rsAddr As DAO.Recordset
    Set rsAddr = frmADDRESSES.Form.Recordset
```

```
    rsAddr.FindFirst tStrSearchStr
End If
```

Exit_CmdFind_Click:

```
Exit Sub
```

Err_CmdFind_Click:

```
    MsgBox Err.Description
    Resume Exit_CmdFind_Click
```

End Sub

```
Private Sub TxtFilterPY_Change()
    If Me.TxtFilterPY.Text = "" Or IsNull(Me.TxtFilterPY.Text) Then
        If Me.TxtFilterChn.Value = "" Or IsNull(Me.TxtFilterChn.Value) Then
            Me.CmdFilter.Enabled = False
        End If
    Else
        Me.TxtFilterChn.Value = ""
        Me.CmdFilter.Enabled = True
    End If
End Sub
```

End Sub

```
Private Sub TxtfilterChn_Change()
```

```
    If Me.TxtFilterChn.Text = "" Or IsNull(Me.TxtFilterChn.Text) Then
        If Me.TxtFilterPY.Value = "" Or IsNull(Me.TxtFilterPY.Value) Then
            Me.CmdFilter.Enabled = False
        End If
    Else
```

```
        Me.TxtFilterPY.Value = ""
    End If
End Sub
```

```
        Me.CmdFilter.Enabled = True
    End If
End Sub

Private Sub TxtSearchPY_Change()

    If Me.TxtSearchPY.Text = "" Or IsNull(Me.TxtSearchPY.Text) Then
        If Me.TxtSearchChn.Value = "" Or IsNull(Me.TxtSearchChn.Value) Then
            Me.CmdFind.Enabled = False
        End If
    Else
        Me.TxtSearchChn.Value = ""
        Me.CmdFind.Enabled = True
    End If
End Sub

Private Sub TxtSearchChn_Change()

    If Me.TxtSearchChn.Text = "" Or IsNull(Me.TxtSearchChn.Text) Then
        If Me.TxtSearchPY.Value = "" Or IsNull(Me.TxtSearchPY.Value) Then
            Me.CmdFind.Enabled = False
        End If
    Else
        Me.TxtSearchPY.Value = ""
        Me.CmdFind.Enabled = True
    End If
End Sub
```

frmPickAddresses_multi -1

Option Compare Database
Public gSelectCount As Integer

Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click

DoCmd.Close

Exit_CmdCancel_Click:
Exit Sub

Err_CmdCancel_Click:
MsgBox Err.Description
Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdFilter_Click()
Dim tStrFilterPY As String, tStrFilterChn As String, tStrFilter As String, tStrLen As String

tStrFilter = ""

'Me.TxtFilterChn.SetFocus
If TxtFilterChn.Value <> "" Then
tStrFilterChn = Trim(TxtFilterChn.Value)
tStrLen = Str(LenB(tStrFilterChn))
tStrFilter = "LeftB(c_name_chn," + tStrLen + ") = '" + tStrFilterChn + "'"

Else
'TxtFilterPY.SetFocus
If TxtFilterPY.Value <> "" Then
tStrFilterPY = Trim(TxtFilterPY.Value)
tStrLen = Str(Len(tStrFilterPY))
tStrFilter = "Left(c_name," + tStrLen + ") = '" + tStrFilterPY + "'"

End If
End If

If tStrFilter <> "" Then
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

cmdSQL.CommandText = "Delete * from ZZ_ADDRESSES_TMP"
cmdSQL.Execute tRecDeleted

'MsgBox tStrSQL

cmdSQL.CommandText = "INSERT INTO ZZ_ADDRESSES_TMP (c_addr_id, c_name, c_name_chn, c_admin_type, c_first
year, c_lastyear, x_coord, y_coord, " + _
"belongs_to_ID, belongs_to_py, belongs_to_chn) " + _
"SELECT ZZZ_ADDR_BELONGS.c_addr_id, ZZZ_ADDR_BELONGS.c_name, ZZZ_ADDR_BELONGS.c_name_chn, ZZZ_ADDR_BE
LONGS.c_admin_type, " + _
"ZZZ_ADDR_BELONGS.c_firstyear, ZZZ_ADDR_BELONGS.c_lastyear, ZZZ_ADDR_BELONGS.x_coord, ZZZ_ADD
R_BELONGS.y_coord, " + _
" ZZZ_ADDR_BELONGS.belongs_to_ID, ZZZ_ADDR_BELONGS.belongs_to_py, ZZZ_ADDR_BELONGS.belongs_to_c
hn " + _
"FROM ZZZ_ADDR_BELONGS " + _
"WHERE ((" + tStrFilter + "))"
cmdSQL.Execute tRecDeleted

ListAddr.Requery
'For ti = 0 To ListAddr.ListCount
' ListAddr.Selected(ti) = False
'Next ti
gSelectCount = 0

End If

CmdFilterClear.Enabled = True
CmdSelectAllFiltered.Enabled = True

End Sub

Private Sub CmdFilterClear_Click()
Dim cmdSQL As ADODB.Command, tRecDeleted As Long, ti As Long

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

cmdSQL.CommandText = "Delete * from ZZ_ADDRESSES_TMP"

frmPickAddresses_multi -2

```
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "INSERT INTO ZZ_ADDRESSES_TMP ( c_addr_id, c_name, c_name_chn, c_admin_type, c_firstyear, c_lastyear, x_coord, y_coord, " + _
    "belongs_to_ID, belongs_to_py, belongs_to_chn ) " + _
    "SELECT ZZ_ADDR_BELONGS.c_addr_id, ZZ_ADDR_BELONGS.c_name, ZZ_ADDR_BELONGS.c_name_chn, ZZ_ADDR_BELONGS.c_admin_type, " + _
    "ZZ_ADDR_BELONGS.c_firstyear, ZZ_ADDR_BELONGS.c_lastyear, ZZ_ADDR_BELONGS.x_coord, ZZ_ADDR_BELONGS.y_coord, " + _
    "ZZ_ADDR_BELONGS.belongs_to_ID, ZZ_ADDR_BELONGS.belongs_to_py, ZZ_ADDR_BELONGS.belongs_to_chn " + _
    "FROM ZZ_ADDR_BELONGS"
cmdSQL.Execute tRecDeleted

ListAddr.Requery
gSelectCount = 0

CmdFilterClear.Enabled = False
CmdSelectAllFiltered.Enabled = False
```

End Sub

```
Private Sub CmdSelect_Click()
    Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, varItm As Variant, ti As Long
```

```
    CmdSelect.Enabled = False
```

```
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
```

```
    cmdSQL.CommandText = "DELETE * FROM ZZ_ADDRESSES"
    cmdSQL.Execute tRecCount
```

```
    TxtSelectCount.Value = gSelectCount
```

```
    ' first copy the records over to a scratch table
```

```
    Set tRst = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
```

```
    For Each varItm In ListAddr.ItemsSelected
        tRst.AddNew
        tRst!c_addr_id = ListAddr.Column(9, varItm)
        tRst!c_name = ListAddr.Column(0, varItm)
        tRst!c_name_chn = ListAddr.Column(1, varItm)
        tRst!c_admin_type = ListAddr.Column(2, varItm)
        tRst!c_firstyear = ListAddr.Column(3, varItm)
        tRst!c_lastyear = ListAddr.Column(4, varItm)
        If Not (ListAddr.Column(5, varItm) = "") Then
            tRst!x_coord = ListAddr.Column(5, varItm)
            tRst!y_coord = ListAddr.Column(6, varItm)
        End If
        tRst!belongs_to_py = ListAddr.Column(7, varItm)
        tRst!belongs_to_chn = ListAddr.Column(8, varItm)
        tRst.Update
    Next varItm
    tRst.Close
```

```
    ListAddr.Requery
    gSelectCount = 0
```

```
    TxtAddrFilter.Value = False
    Forms!frmPickAddresses_multi.Visible = False
```

End Sub

```
Private Sub CmdSelectAllFiltered_Click()
    Dim cmdSQL As ADODB.Command, tRecDeleted As Long, ti As Long
```

```
    ' copy ZZ_ADDRESSES_TMP into ZZ_ADDRESSES
```

```
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
```

```
    cmdSQL.CommandText = "Delete * from ZZ_ADDRESSES"
    cmdSQL.Execute tRecDeleted
```

```
    cmdSQL.CommandText = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, c_admin_type, c_firstyear, c_lastyear, " + _
        "x_coord, y_coord, belongs_to_ID, belongs_to_py, belongs_to_chn ) " + _
        "SELECT ZZ_ADDRESSES_TMP.c_addr_id, ZZ_ADDRESSES_TMP.c_name, ZZ_ADDRESSES_TMP.c_name_chn, ZZ_ADDRESSES_TMP
```

```

Pnm_frmPickAddresses_multi -3

P.c_admin_type, " + _
    "ZZ_ADDRESSES_TMP.c_firstyear, ZZ_ADDRESSES_TMP.c_lastyear, ZZ_ADDRESSES_TMP.x_coord, ZZ_ADDRESSES_TM
P.y_coord, " + _
    "ZZ_ADDRESSES_TMP.belongs_to_ID, ZZ_ADDRESSES_TMP.belongs_to_py, ZZ_ADDRESSES_TMP.belongs_to_chn " +
-
    "FROM ZZ_ADDRESSES_TMP"
cmdSQL.Execute tRecDeleted

TxtAddrFilter.Value = True
Forms!frmPickAddresses_multi.Visible = False

End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim cmdSQL As ADODB.Command, tRecDeleted As Long, ti As Long

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    cmdSQL.CommandText = "Delete * from ZZ_ADDRESSES_TMP"
    cmdSQL.Execute tRecDeleted

    cmdSQL.CommandText = "INSERT INTO ZZ_ADDRESSES_TMP ( c_addr_id, c_name, c_name_chn, c_admin_type, c_firstyear
, c_lastyear, x_coord, y_coord, " + _
        "belongs_to_ID, belongs_to_py, belongs_to_chn ) " + _
        "SELECT ZZZ_ADDR_BELONGS.c_addr_id, ZZZ_ADDR_BELONGS.c_name, ZZZ_ADDR_BELONGS.c_name_chn, ZZZ_ADDR_BELONG
S.c_admin_type, " + _
        "ZZZ_ADDR_BELONGS.c_firstyear, ZZZ_ADDR_BELONGS.c_lastyear, ZZZ_ADDR_BELONGS.x_coord, ZZZ_ADDR_BE
LONGS.y_coord, " + _
        "ZZZ_ADDR_BELONGS.belongs_to_ID, ZZZ_ADDR_BELONGS.belongs_to_py, ZZZ_ADDR_BELONGS.belongs_to_chn
" + _
        "FROM ZZZ_ADDR_BELONGS"
    cmdSQL.Execute tRecDeleted

    ListAddr.Requery
    'For ti = 0 To ListAddr.ListCount
    '    ListAddr.Selected(ti) = False
    'Next ti

    gSelectCount = 0

    If Not IsNull(Me.OpenArgs) Then
        Dim strADDR As String, rsAddr As DAO.Recordset
        strADDR = Me.OpenArgs

    End If
    CmdSelectAllFiltered.Enabled = False
End Sub

Private Sub ListAddr_Click()
    Dim ti As Long, tUnclicked As Boolean
    Dim varItm As Variant

    gSelectCount = 0
    For Each varItm In ListAddr.ItemsSelected
        gSelectCount = gSelectCount + 1
    Next varItm
    'MsgBox ListAddr.Column(1, ti + 1) + ": Select Count = " + Str(gSelectCount)

    If gSelectCount = 0 Then
        Me.CmdSelect.Enabled = False
    Else
        Me.CmdSelect.Enabled = True
    End If
End Sub

End Sub

Private Sub TxtFilterPY_Change()
    If Me.TxtFilterPY.Text = "" Or IsNull(Me.TxtFilterPY.Text) Then
        If Me.TxtFilterChn.Value = "" Or IsNull(Me.TxtFilterChn.Value) Then
            Me.CmdFilter.Enabled = False
        End If
    Else
        Me.TxtFilterChn.Value = ""
        Me.CmdFilter.Enabled = True
    End If
End Sub

End Sub

```

```
Private Sub TxtfilterChn_Change()  
  
    If Me.TxtFilterChn.Text = "" Or IsNull(Me.TxtFilterChn.Text) Then  
        If Me.TxtFilterPY.Value = "" Or IsNull(Me.TxtFilterPY.Value) Then  
            Me.CmdFilter.Enabled = False  
        End If  
    Else  
        Me.TxtFilterPY.Value = ""  
        Me.CmdFilter.Enabled = True  
    End If  
End Sub
```

Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
    MsgBox Err.Description  
    Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
    Forms!frmPickASSOC_CODES.Visible = False  
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
    frmASSOC_CODES.Form.OrderBy = "c_sortorder"  
    frmASSOC_CODES.Form.OrderByOn = True
```

```
        If Not IsNull(Me.OpenArgs) Then  
            Dim strAssoc As String  
            strAssoc = Me.OpenArgs  
            Dim rsAssoc As DAO.Recordset  
            Set rsAssoc = frmASSOC_CODES.Form.Recordset  
            rsAssoc.FindFirst "c_assoc_code = " & strAssoc  
        End If
```

```
End Sub
```

```
Private Sub CmdFind_Click()  
On Error GoTo Err_CmdFind_Click
```

```
    Dim StrSearch As String  
    Me.TxtSearch.SetFocus  
    StrSearch = Me.TxtSearch.Value  
    If StrSearch <> "" Then  
        Dim rsAssocCodes As DAO.Recordset  
        Set rsAssocCodes = frmASSOC_CODES.Form.Recordset  
        Dim StrSearchStr As String  
        StrSearchStr = "c_assoc_desc_chn = " + Chr(34) + StrSearch + Chr(34)  
        rsAssocCodes.FindFirst StrSearchStr  
    End If
```

```
Exit_CmdFind_Click:  
Exit Sub
```

```
Err_CmdFind_Click:  
    MsgBox Err.Description  
    Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()  
    If Me.TxtSearch.Text = "" Then  
        Me.CmdFind.Enabled = False  
    Else  
        Me.CmdFind.Enabled = True  
    End If  
End Sub
```

Option Compare Database

```
Public gRstAssocCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String, gSelectCount As Integer
```

```
'#####Treeview Code#####
```

```
'Add this to your form's declaration section
```

```
Public WithEvents mcTree As clsTreeview
```

```
Private mbExit As Boolean ' to exit a SpinButton event
```

```
'#####Treeview Code#####
```

```
Private Sub CmdCancel_Click()
```

```
On Error GoTo Err_CmdCancel_Click
```

```
Clear_SelectAll
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
```

```
Exit Sub
```

```
Err_CmdCancel_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()
```

```
Dim cmdSQL As ADODB.Command, tRecCount As Long, tRst As DAO.Recordset, varItm As Variant, ti As Integer
```

```
,
```

```
CmdSelectAll.SetFocus
```

```
CmdSelect.Enabled = False
```

```
,
```

```
' if SelectAll is the selection status, then all records in ZZ_ASSOC_CODE_TMP are used
```

```
' otherwise, we clear the table and copy the selected rows
```

```
,
```

```
' get the count
```

```
,
```

```
gSelectCount = 0
```

```
For Each varItm In ListAssoc.ItemsSelected
```

```
gSelectCount = gSelectCount + 1
```

```
Next varItm
```

```
,
```

```
'MsgBox "gSelectCount = " + Str(gSelectCount) + " ListCount = " + Str(ListAssoc.ListCount)
```

```
,
```

```
' put the description in the boxes
```

```
,
```

```
If gSelectCount = 0 Then
```

```
Me.TxtAssocDesc.Value = ""
```

```
Me.TxtAssocDescChn.Value = ""
```

```
Me.TxtAssocID.Value = 0
```

```
Me.CmdSelect.Enabled = False
```

```
Else
```

```
Set cmdSQL = New ADODB.Command
```

```
cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
cmdSQL.CommandType = adCmdText
```

```
If gSelectCount = 1 Then
```

```
,
```

```
' there is just one item in the collection
```

```
,
```

```
For Each varItm In ListAssoc.ItemsSelected
```

```
Me.TxtAssocDesc.Value = ListAssoc.Column(1, varItm)
```

```
Me.TxtAssocDescChn.Value = ListAssoc.Column(2, varItm)
```

```
Me.TxtAssocID.Value = ListAssoc.Column(0, varItm)
```

```
Next varItm
```

```
ElseIf gSelectCount = ListAssoc.ListCount - 1 Then
```

```
,
```

```
' when a category is selected in the tree, all codes are put into ZZ_ASSOC_CODE_TMP
```

```
' when "Select All", all the records are selected and therefore are copied over
```

```
,
```

```
Me.TxtAssocDesc.Value = "All"
```

```
Me.TxtAssocDescChn.Value = "All"
```

```
Me.TxtAssocID.Value = -1
```

```
Else
```

```
Me.TxtAssocDesc.Value = "Multi-select"
```

```
Me.TxtAssocDescChn.Value = "Multi-select"
```

```
Me.TxtAssocID.Value = -2
```

```
End If
```

```
,
```

```
' now process the records
```

```
,
```



```
cmdSQL.CommandText = "DELETE * FROM ZZ_ASSOC_CODE"
cmdSQL.Execute tRecCount
```

```
' copy the records over to the table
```

```
Set tRst = CurrentDb.OpenRecordset("ZZ_ASSOC_CODE", dbOpenDynaset)
For Each varItm In ListAssoc.ItemsSelected
    tRst.AddNew
    tRst!c_assoc_code = ListAssoc.Column(0, varItm)
    tRst!c_assoc_desc = ListAssoc.Column(1, varItm)
    tRst!c_assoc_desc_chn = ListAssoc.Column(2, varItm)
    tRst.Update
Next varItm
tRst.Close
```

```
ListAssoc.Requery
'For ti = 0 To ListAssoc.ListCount
'    ListAssoc.Selected(ti) = False
'Next ti
gSelectCount = 0
```

```
End If
Forms!frmPickAssoc_multi.Visible = False
End Sub
```

```
Private Sub CmdSelectAll_Click()
    If CmdSelectAll.Caption = "Select All" Then
        CmdSelectAll.Caption = "De-select All"

        For ti = 0 To ListAssoc.ListCount
            ListAssoc.Selected(ti) = True
        Next ti
```

```
        CmdSelect.Enabled = True
        TxtAssocID.Value = -1
```

```
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub
```

```
Private Sub Clear_SelectAll()
    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    For ti = 0 To ListAssoc.ListCount
        ListAssoc.Selected(ti) = False
    Next ti
```

```
    gSelectCount = 0
    CmdSelect.Enabled = False
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
    Dim cRoot As clsNode, strKey As String, strCaption As String
    Dim cNode1 As clsNode, cNode2 As clsNode
    Dim tRst As DAO.Recordset, cmdSQL As ADODB.Command, tRecCount As Long
```

```
    ' initialize the text fields
    '
```

```
Me.TxtTypeDesc.Value = ""
Me.TxtTypeDescChn.Value = ""
Me.TxtTypeID.Value = "000"
gSelectCount = 0
```

```
    ' initialize the listbox
    '
```

```
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
```

```
cmdSQL.CommandText = "DELETE * FROM ZZ_ASSOC_CODE_TMP"
cmdSQL.Execute tRecCount
```

```
cmdSQL.CommandText = "INSERT INTO ZZ_ASSOC_CODE_TMP ( c_assoc_code, c_assoc_desc, c_assoc_desc_chn ) " + _
    "SELECT ASSOC_CODES.c_assoc_code, ASSOC_CODES.c_assoc_desc, ASSOC_CODES.c_assoc_desc_chn" + _
    "FROM ASSOC_CODES"
cmdSQL.Execute tRecCount
```

```

ListAssoc.Requery

If Not IsNull(Me.OpenArgs) Then
    Dim strAssoc As String
    strAssoc = Me.OpenArgs
End If
'
' build treeview
'
Set tRst = CurrentDb.OpenRecordset("ASSOC_TYPES", dbOpenDynaset)

' set the language
Dim tmli As MsoLanguageID
'
tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
'
If tmli = msoLanguageIDSimplifiedChinese Then
    gDisplayLanguage = "S"
ElseIf tmli = msoLanguageIDTraditionalChinese Then
    gDisplayLanguage = "T"
Else
    gDisplayLanguage = "E"
End If
'
'
' MsgBox "About to build tree"
tRst.MoveFirst
Set mcTree = Me.subTreeView.Form.pTreeview
With mcTree
    .NodesClear
    .AppName = AppName ' Title for message boxes:
    ' use the appropriate caption
    If gDisplayLanguage = "T" Then
        strCaption = ChrW(31038) + ChrW(26371) + ChrW(38364) + ChrW(20418) + ChrW(20998) + ChrW(39006)
    ElseIf gdisplaylangauge = "S" Then
        strCaption = ChrW(31038) + ChrW(20250) + ChrW(20851) + ChrW(31995) + ChrW(20998) + ChrW(31612)
    Else
        strCaption = "Categories of Social Relations"
    End If

    Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
    ' Add a Root node with main and expanded icons and make it bold
    cRoot.Bold = True
    ' Loop through the records
    Do While Not tRst.EOF
        ' Add node
        strKey = tRst!c_assoc_type_code
        If gDisplayLanguage = "E" Then
            strCaption = tRst!c_assoc_type_desc
        Else
            strCaption = tRst!c_assoc_type_desc_chn
        End If
        If Len(tRst!c_assoc_type_code) = 2 Then
            Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode1.Expanded = False
        Else
            Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode2.Expanded = False
        End If
        tRst.MoveNext
    Loop
    ' Create the node controls and display the tree
    .Refresh
End With
'
Set tRst = Nothing
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

'Dim StrSearch As String
'Me.TxtSearch.SetFocus
'StrSearch = Me.TxtSearch.Value
'If StrSearch <> "" Then
'    Dim rsAssocCodes As DAO.Recordset
'    Set rsAssocCodes = frmASSOC_CODES.Form.Recordset
'    Dim StrSearchStr As String
'    StrSearchStr = "c_assoc_desc_chn = " + Chr(34) + StrSearch + Chr(34)
'    rsAssocCodes.FindFirst StrSearchStr
'End If

```

Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click

End Sub

```
Private Sub ListAssoc_Click()
    Dim ti As Long, tUnclicked As Boolean
    Dim varItm As Variant

    gSelectCount = 0

    For Each varItm In ListAssoc.ItemsSelected
        gSelectCount = gSelectCount + 1
    Next varItm

    If gSelectCount = 0 Then
        'Me.TxtAssocDesc.Value = ""
        'Me.TxtAssocDescChn.Value = ""
        'Me.TxtAssocID.Value = 0
        Me.CmdSelect.Enabled = False
    Else
        'If gSelectCount = 1 Then
        '    If tUnclicked Then
        '        ' this means that there is only on selected item, but it is NOT this item
        '        ' we therefore need to locate the selected item and put its values into the text boxes
        '        ' I may not even use these boxes anymore, but just in case...
        '        For Each varItm In ListAssoc.ItemsSelected
        '            Me.TxtAssocDesc.Value = ListAssoc.Column(1, varItm)
        '            Me.TxtAssocDescChn.Value = ListAssoc.Column(2, varItm)
        '            Me.TxtAssocID.Value = ListAssoc.Column(0, varItm)
        '            'MsgBox ListAssoc.Column(1, varItm)
        '        Next varItm
        '    Else
        '        Me.TxtAssocDesc.Value = ListAssoc.Column(1, ti + 1)
        '        Me.TxtAssocDescChn.Value = ListAssoc.Column(2, ti + 1)
        '        Me.TxtAssocID.Value = ListAssoc.Column(0, ti + 1)
        '        'MsgBox ListAssoc.Column(1, ti + 1)
        '    End If
        'Else
        '    Me.TxtAssocDesc.Value = "Multi-select"
        '    Me.TxtAssocDescChn.Value = "Multi-select"
        '    Me.TxtAssocID.Value = -2
        '    'MsgBox "Multi-select"
        'End If
        Me.CmdSelect.Enabled = True
    End If
End Sub
```

```
Private Sub mcTree_Click(cNode As clsNode)
    Dim tRst As DAO.Recordset, tRstAssoc As DAO.Recordset, tStrSQL As String, ti As Integer
    Dim tAssocCodeQuery As DAO.QueryDef, prm As DAO.Parameter
    Dim tRstAssocCode As DAO.Recordset, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command
```

```
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
```

```
TxtAssocID.Value = -1
TxtAssocDesc.Value = ""
TxtAssocDescChn.Value = ""
CmdSelect.Enabled = False
```

```
' reset the form colors
```

```
,
```

```
Clear_SelectAll
```

```
,
```

```
If cNode.Key = "Root" Then
    TxtTypeID.Value = ""
    TxtTypeDesc.Value = ""
    TxtTypeDescChn.Value = ""
    CmdSelectAll.Enabled = False
    '
    ' reset the entry code choices
```

frm_frmPickASSOC_multi - 5

```
'
cmdSQL.CommandText = "DELETE * FROM ZZ_ASSOC_CODE_TMP"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "INSERT INTO ZZ_ASSOC_CODE_TMP ( c_assoc_code, c_assoc_desc, c_assoc_desc_chn ) " +
_chn " + _
"SELECT ASSOC_CODES.c_assoc_code, ASSOC_CODES.c_assoc_desc, ASSOC_CODES.c_assoc_desc
"FROM ASSOC_CODES"
cmdSQL.Execute tRecCount

ListAssoc.Requery
For ti = 0 To ListAssoc.ListCount
    ListAssoc.Selected(ti) = False
Next ti
gSelectCount = 0
Else
    Set tRst = CurrentDb.OpenRecordset("ASSOC_TYPES", dbOpenDynaset)
    '
    tRst.MoveFirst
    tRst.FindFirst "c_assoc_type_code = " + Chr(34) + cNode.Key + Chr(34)
    TxtTypeID.Value = cNode.Key
    TxtTypeDesc.Value = tRst!c_assoc_type_desc
    TxtTypeDescChn.Value = tRst!c_assoc_type_desc_chn
    tRst.Close
    Set tRst = Nothing
    '
    CmdSelectAll.Enabled = True
    '
    ' we need to distinguish between type / subtype
    '
    tStrSQL = "INSERT INTO ZZ_ASSOC_CODE_TMP ( c_assoc_code, c_assoc_desc, c_assoc_desc_chn, c_sortorder ) "
+ _
    "SELECT ASSOC_CODE_TYPE_REL.c_assoc_code AS c_assoc_code, ASSOC_CODES.c_assoc_desc, " + _
    "ASSOC_CODES.c_assoc_desc_chn, ASSOC_CODES.c_sortorder " + _
    "FROM ASSOC_CODES INNER JOIN ASSOC_CODE_TYPE_REL ON " + _
    "ASSOC_CODES.c_assoc_code = ASSOC_CODE_TYPE_REL.c_assoc_code "

    If Len(cNode.Key) = 2 Then
        tStrSQL = tStrSQL + "WHERE (((Left([ASSOC_CODE_TYPE_REL].[c_assoc_type_code]),2))='" + _
        TxtTypeID.Value + "'))"
    Else
        tStrSQL = tStrSQL + "WHERE (((ASSOC_CODE_TYPE_REL.c_assoc_type_code)='" + _
        TxtTypeID.Value + "'))"
    End If

    cmdSQL.CommandText = "Delete * from ZZ_ASSOC_CODE_TMP"
    cmdSQL.Execute tRecDeleted
    '
    MsgBox "tStrSQL = " + tStrSQL
    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    '
    ListAssoc.Requery
    For ti = 0 To ListAssoc.ListCount
        ListAssoc.Selected(ti) = False
    Next ti
    gSelectCount = 0
    '
End If

End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Or IsNull(TxtSearch.Text) Then
        If TxtSearchChn.Value = "" Or IsNull(TxtSearchChn.Value) Then
            CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Or IsNull(TxtSearchChn.Text) Then
        If TxtSearch.Value = "" Or IsNull(TxtSearch.Value) Then
            Me.CmdFind.Enabled = False
        End If
    End If
```

```

Else
    TxtSearch.Value = ""
    CmdFind.Enabled = True
End If
CmdFindNext.Enabled = False
End Sub

Private Sub CmdFindNext_Click()
    Dim tRstAssocCodes As DAO.Recordset, tRstAssocTypes As DAO.Recordset, cNode As clsNode
    Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command, ti As Integer

    TxtAssocID.Value = -1
    TxtAssocDesc.Value = ""
    TxtAssocDescChn.Value = ""

    If IsNull(gRstAssocCode) Then
        MsgBox "Error in search: table closed."
    Else
        If gStrSearch = "" Then
            MsgBox "Error in search: string empty."
        Else
            'MsgBox "Looking for entry"

            If gRstAssocCode.EOF Then
                CmdFindNext.Enabled = False
            Else
                ' gRstAssocCode.MoveNext

                If gUseAlt Then
                    gRstAssocCode.FindNext gStrSearchAlt
                Else
                    gRstAssocCode.FindNext gStrSearch

                    If gRstAssocCode.NoMatch Then
                        gRstAssocCode.FindFirst gStrSearchAlt
                        gUseAlt = True
                    End If
                End If
            End If

            If gRstAssocCode.NoMatch Then
                If gUseAlt Then
                    gUseAlt = False
                End If
                CmdFindNext.Enabled = False
            Else
                '
                ' next find the entry_type
                '
                'MsgBox "Looking for entry type"

                Set tRstAssocTypes = CurrentDb.OpenRecordset("ASSOC_CODE_TYPE_REL", dbOpenDynaset)

                tRstAssocTypes.FindNext "c_assoc_code = " + Str(gRstAssocCode!c_assoc_code)

                If Not tRstAssocTypes.NoMatch Then
                    '
                    ' set the values
                    '
                    TxtAssocID.Value = gRstAssocCode!c_assoc_code
                    If Not IsNull(gRstAssocCode!c_assoc_desc) Then
                        TxtAssocDesc.Value = gRstAssocCode!c_assoc_desc
                    End If
                    If Not IsNull(gRstAssocCode!c_assoc_desc_chn) Then
                        TxtAssocDescChn.Value = gRstAssocCode!c_assoc_desc_chn
                    End If
                    '
                    ' define the string
                    '
                    tStr = tRstAssocTypes!c_assoc_type_code
                    '
                    ' search the tree

                    Set cNode = mcTree.Nodes(tStr)
                    '
                    'MsgBox "found node"
                    '
                    If Not IsNull(cNode) Then
                        If cNode.Key = gNode.Key Then
                            'Set tRstAssocCodes = frmZZZ_ASSOC_CODE.Form.Recordset
                            'tRstAssocCodes.FindNext "c_assoc_code = " + Str(gRstAssocCode!c_assoc_code)

```

```

        'frmZZZ_ASSOC_CODE.Form.Refresh

gSelectCount = 0
For i = 0 To ListAssoc.ListCount - 1
    If gRstAssocCode!c_assoc_code = ListAssoc.Column(0, i) Then
        ListAssoc.ListIndex = i
        ListAssoc.Selected(i) = True
        gSelectCount = gSelectCount + 1
    End If
Next i
Else
    Set mcTree.ActiveNode = cNode
    'cNode.Selected = True
    '
    ' then one makes it visible
    '
    'tNode.EnsureVisible
    '
    Set gNode = cNode
    '
    ' Finally populate the options and select the record.
    '
    CmdSelectAll.Enabled = True
    '
    tStrSQL = "INSERT INTO ZZ_ASSOC_CODE_TMP ( c_assoc_code, c_assoc_desc, c_assoc_de
        "SELECT ASSOC_CODE_TYPE_REL.c_assoc_code AS c_assoc_code, ASSOC_CODES.c_assoc
        "ASSOC_CODES.c_assoc_desc_chn, ASSOC_CODES.c_sortorder " + _
        "FROM ASSOC_CODES INNER JOIN ASSOC_CODE_TYPE_REL ON " + _
        "ASSOC_CODES.c_assoc_code = ASSOC_CODE_TYPE_REL.c_assoc_code " + _
        "WHERE (((ASSOC_CODE_TYPE_REL.c_assoc_type_code)=" + tStr + "')))"

    '
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    cmdSQL.CommandText = "Delete * from ZZ_ASSOC_CODE_TMP"
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    '
    ListAssoc.Requery
    For ti = 0 To ListAssoc.ListCount
        ListAssoc.Selected(ti) = False
    Next ti
    gSelectCount = 0

    'Set tRstAssocCodes = CurrentDb.OpenRecordset("ZZ_ASSOC_CODE_TMP", dbOpenDynaset)
    '
    For i = 0 To ListAssoc.ListCount - 1
        If gRstAssocCode!c_assoc_code = ListAssoc.Column(0, i) Then
            ListAssoc.ListIndex = i
            ListAssoc.Selected(i) = True
            gSelectCount = gSelectCount + 1
        End If
    Next i
    'Set frmZZZ_ASSOC_CODE.Form.Recordset = tRstAssocCodes
    'tRstAssocCodes.FindNext "c_assoc_code = " + Str(gRstAssocCode!c_assoc_code)
    'frmZZZ_ASSOC_CODE.Form.Refresh
    '
    '
    ' set the type values
    '
    Set tRstAssocTypes = CurrentDb.OpenRecordset("ASSOC_TYPES", dbOpenDynaset)
    '
    tRstAssocTypes.MoveFirst
    tRstAssocTypes.FindFirst "c_assoc_type_code = " + Chr(34) + cNode.Key + Chr(34)
    TxtTypeID.Value = cNode.Key
    TxtTypeDesc.Value = tRstAssocTypes!c_assoc_type_desc
    TxtTypeDescChn.Value = tRstAssocTypes!c_assoc_type_desc_chn
    tRstAssocTypes.Close
    Set tRstAssocTypes = Nothing
End If
End If
End If
End If

```

```

        End If
    End If
End If

If gSelectCount = 0 Then
    CmdSelect.Enabled = False
Else
    CmdSelect.Enabled = True
End If
End Sub

Private Sub NodeSearch()

    Dim cNode As clsNode, tStr As String, tRstAssocCodes As DAO.Recordset, tRstAssocTypes As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrLen As String, cmdSQL As ADODB.Command

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    TxtAssocID.Value = -1
    TxtAssocDesc.Value = ""
    TxtAssocDescChn.Value = ""

    ' all specific association codes will have a type of the form 0101
    ' hence the ValuePath for the relevant node will be "K000/K01/K0101"
    ' The command to locate the relevant node is:
    ' tNode = TreeViewType.FindNode(tStrValuePath)
    ' TreeViewType.SelectedNode = tNode

    ' search for the search string in ASSOC_CODES
    TxtSearchChn.SetFocus
    tStrSearchChn = Me.TxtSearchChn.Text
    TxtSearch.SetFocus
    tStrSearchEng = Me.TxtSearch.Text
    CmdFind.SetFocus

    gStrSearch = ""
    gUseAlt = False

    If tStrSearchChn <> "" Then
        tStrLen = Str(LenB(tStrSearchChn))
        gStrSearch = "LeftB(c_assoc_desc_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
        gStrSearchAlt = "InStrB(1,c_assoc_desc_chn,'" + tStrSearchChn + "') > 0"
    ElseIf tStrSearchEng <> "" Then
        tStrLen = Str(Len(tStrSearchEng))
        gStrSearch = "Left(c_assoc_desc," + tStrLen + ") = '" + tStrSearchEng + "'"
        gStrSearchAlt = "InStr(1,c_assoc_desc,'" + tStrSearchEng + "') > 0"
    End If

    If Not (gStrSearch = "") Then

        'MsgBox "Looking for assoc"

        Set gRstAssocCode = CurrentDb.OpenRecordset("ASSOC_CODES", dbOpenDynaset)

        gRstAssocCode.FindFirst gStrSearch

        If gRstAssocCode.NoMatch Then
            gRstAssocCode.FindFirst gStrSearchAlt
            gUseAlt = True
        End If

        If gRstAssocCode.NoMatch Then
            gUseAlt = False
            CmdFindNext.Enabled = False
        Else
            '
            CmdFindNext.Enabled = True
            '
            ' next find the assoc_type

            'MsgBox "Looking for assoc type"

            Set tRstAssocTypes = CurrentDb.OpenRecordset("ASSOC_CODE_TYPE_REL", dbOpenDynaset)

            tRstAssocTypes.FindNext "c_assoc_code = " + Str(gRstAssocCode!c_assoc_code)

```

```

If Not tRstAssocTypes.NoMatch Then
    '
    ' set the values
    '
    TxtAssocID.Value = gRstAssocCode!c_assoc_code
    If Not IsNull(gRstAssocCode!c_assoc_desc) Then
        TxtAssocDesc.Value = gRstAssocCode!c_assoc_desc
    End If
    If Not IsNull(gRstAssocCode!c_assoc_desc_chn) Then
        TxtAssocDescChn.Value = gRstAssocCode!c_assoc_desc_chn
    End If
    '
    ' define the string
    '
    tStr = tRstAssocTypes!c_assoc_type_code
    '
    ' search the tree

Set cNode = mcTree.Nodes(tStr)
'
'MsgBox "found node"
'
If Not IsNull(cNode) Then
    '
    Set mcTree.ActiveNode = cNode
    'cNode.Selected = True
    '
    ' then one makes it visible
    '
    'tNode.EnsureVisible
    '
    Set gNode = cNode
    '
    ' Finally populate the options and select the record.
    '
    CmdSelectAll.Enabled = True
    '
    tStrSQL = "INSERT INTO ZZ_ASSOC_CODE_TMP ( c_assoc_code, c_assoc_desc, c_assoc_desc_chn, c_so
rtorder ) " + _
        "SELECT ASSOC_CODE_TYPE_REL.c_assoc_code AS c_assoc_code, ASSOC_CODES.c_assoc_desc, " + _
        "ASSOC_CODES.c_assoc_desc_chn, ASSOC_CODES.c_sortorder " + _
        "FROM ASSOC_CODES INNER JOIN ASSOC_CODE_TYPE_REL ON " + _
        "ASSOC_CODES.c_assoc_code = ASSOC_CODE_TYPE_REL.c_assoc_code " + _
        "WHERE ((ASSOC_CODE_TYPE_REL.c_assoc_type_code)='" + tStr + "') "

    '
    cmdSQL.CommandText = "Delete * from ZZ_ASSOC_CODE_TMP"
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    '
    'Set tRstAssocCodes = CurrentDb.OpenRecordset("ZZ_ASSOC_CODE_TMP", dbOpenDynaset)
    '
    'Set frmZZZ_ASSOC_CODE.Form.Recordset = tRstAssocCodes
    'tRstAssocCodes.FindNext "c_assoc_code = " + Str(gRstAssocCode!c_assoc_code)
    'frmZZZ_ASSOC_CODE.Form.Refresh
    '
    'Set tRstDummy = Nothing
    '
    ListAssoc.Requery
    For i = 0 To ListAssoc.ListCount
        ListAssoc.Selected(i) = False
    Next i
    gSelectCount = 0
    '
    For i = 0 To ListAssoc.ListCount - 1
        If gRstAssocCode!c_assoc_code = ListAssoc.Column(0, i) Then
            ListAssoc.ListIndex = i
            ListAssoc.Selected(i) = True
            gSelectCount = gSelectCount + 1
            CmdSelect.Enabled = True
        End If
    Next i
    '
    ' set the type values
    '
    Set tRstAssocTypes = CurrentDb.OpenRecordset("ASSOC_TYPES", dbOpenDynaset)
    '
    tRstAssocTypes.MoveFirst

```



```
        tRstAssocTypes.FindFirst "c_assoc_type_code = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key
        TxtTypeDesc.Value = tRstAssocTypes!c_assoc_type_desc
        TxtTypeDescChn.Value = tRstAssocTypes!c_assoc_type_desc_chn
        tRstAssocTypes.Close
        Set tRstAssocTypes = Nothing
    End If
End If
End If
End If

If gSelectCount = 0 Then
    CmdSelect.Enabled = False
End If

End Sub
```

frm_frmPickBAC_multi - 1

Option Compare Database
Public gSelectCount As Integer

Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click

DoCmd.Close

Exit_CmdCancel_Click:
Exit Sub

Err_CmdCancel_Click:
MsgBox Err.Description
Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdSelect_Click()
Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, varItm As Variant, ti As Integer

CmdSelect.Enabled = False

'MsgBox "gSelectCount = " + Str(gSelectCount) + " and ListCount = " + Str(ListBAC.ListCount)

gSelectCount = 0

For Each varItm In ListBAC.ItemsSelected
gSelectCount = gSelectCount + 1
Next varItm

If gSelectCount = ListBAC.ListCount Then
TxtSelectAll.Value = True
Else
TxtSelectAll.Value = False
End If

If gSelectCount > 0 And gSelectCount < ListBAC.ListCount Then

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

cmdSQL.CommandText = "DELETE * FROM ZZ_BIOG_ADDR_CODES"
cmdSQL.Execute tRecCount

TxtSelectCount.Value = gSelectCount

' first copy the records over to a scratch table

'MsgBox "copying codes"

Set tRst = CurrentDb.OpenRecordset("ZZ_BIOG_ADDR_CODES", dbOpenDynaset)
For Each varItm In ListBAC.ItemsSelected
tRst.AddNew
tRst!c_addr_type = ListBAC.Column(2, varItm)
tRst!c_addr_desc = ListBAC.Column(0, varItm)
tRst!c_addr_desc_chn = ListBAC.Column(1, varItm)
tRst.Update
Next varItm
tRst.Close

End If

ListBAC.Requery
'For ti = 0 To ListBAC.ListCount
' ListBAC.Selected(ti) = False
'Next ti
gSelectCount = 0

Forms!frmPickBAC_multi.Visible = False

End Sub

Private Sub CmdSelectAll_Click()

If CmdSelectAll.Caption = "Select All" Then
CmdSelectAll.Caption = "De-select All"

```

        For ti = 0 To ListBAC.ListCount
            ListBAC.Selected(ti) = True
        Next ti

        gSelectCount = ListBAC.ListCount
        CmdSelect.Enabled = True
        TxtSelectAll.Value = True
    Else
        TxtSelectAll.Value = False
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub

Private Sub Clear_SelectAll()
    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    For ti = 0 To ListBAC.ListCount
        ListBAC.Selected(ti) = False
    Next ti

    gSelectCount = 0
    CmdSelect.Enabled = False
End Sub

Private Sub Form_Open(Cancel As Integer)
    'Dim cmdSQL As ADODB.Command, tRecDeleted As Long, ti As Long

    'Set cmdSQL = New ADODB.Command
    'cmdSQL.ActiveConnection = CurrentProject.Connection
    'cmdSQL.CommandType = adCmdText

    'cmdSQL.CommandText = "Delete * from ZZ_BIOG_ADDR_CODES_TMP"
    'cmdSQL.Execute tRecDeleted

    'cmdSQL.CommandText = "INSERT INTO ZZ_BIOG_ADDR_CODES_TMP ( c_addr_type, c_addr_desc, c_addr_desc_chn ) " + _
        "SELECT BIOG_ADDR_CODES.c_addr_type, BIOG_ADDR_CODES.c_addr_desc, BIOG_ADDR_CODES.c_addr_desc_chn " + _
        "FROM BIOG_ADDR_CODES"
    'cmdSQL.Execute tRecDeleted

    ListBAC.Requery
    'For ti = 0 To ListBAC.ListCount
    '    ListBAC.Selected(ti) = False
    'Next ti

    TxtSelectAll.Value = False
    gSelectCount = 0
End Sub

Private Sub ListBAC_Click()
    Dim ti As Long, tUnclicked As Boolean
    Dim varItm As Variant

    'MsgBox "gSelectCount = " + Str(gSelectCount)

    ' this routine will just brute-force the count

    gSelectCount = 0
    For Each varItm In ListBAC.ItemsSelected
        gSelectCount = gSelectCount + 1
    Next varItm

    'MsgBox "gSelectCount = " + Str(gSelectCount)

    If gSelectCount = 0 Then
        Me.CmdSelect.Enabled = False
    Else
        Me.CmdSelect.Enabled = True
    End If

    If gSelectCount = ListBAC.ListCount Then
        TxtSelectAll.Value = True
    Else
        TxtSelectAll.Value = False
    End If
End Sub

```

Option Compare Database

```
Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
Exit Sub
```

```
Err_CmdCancel_Click:
MsgBox Err.Description
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()
On Error GoTo Err_CmdSelect_Click
```

```
Dim tval As Integer
```

```
Forms("frmPickChoronym").Visible = False
```

```
Exit_CmdSelect_Click:
Exit Sub
```

```
Err_CmdSelect_Click:
MsgBox Err.Description
Resume Exit_CmdSelect_Click
```

```
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
frmChoronyms.Form.OrderBy = "c_choronym_desc"
frmChoronyms.Form.OrderByOn = True
```

```
If Not IsNull(Me.OpenArgs) Then
Dim strChoro As String
strChoro = Me.OpenArgs
Dim rsChoro As DAO.Recordset
Set rsChoro = frmChoronyms.Form.Recordset
rsChoro.FindFirst "c_choronym_code = " & strChoro
End If
```

```
End Sub
```

```
Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click
```

```
Dim StrSearch As String
Me.TxtSearch.SetFocus
StrSearch = Me.TxtSearch.Value
If StrSearch <> "" Then
Dim rsChoro As DAO.Recordset
Set rsChoro = frmChoronyms.Form.Recordset
Dim StrSearchStr As String
StrSearchStr = "c_choronym_chn = " + Chr(34) + StrSearch + Chr(34)
rsChoro.FindFirst StrSearchStr
End If
```

```
Exit_CmdFind_Click:
Exit Sub
```

```
Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()
```

```
If Me.TxtSearch.Text = "" Then
Me.CmdFind.Enabled = False
Else
Me.CmdFind.Enabled = True
End If
```

```
End Sub
```

Option Compare Database

```
Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
Exit Sub
```

```
Err_CmdCancel_Click:
MsgBox Err.Description
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()
On Error GoTo Err_CmdSelect_Click
```

```
Forms!frmpickdynasty.Visible = False
```

```
Exit_CmdSelect_Click:
Exit Sub
```

```
Err_CmdSelect_Click:
MsgBox Err.Description
Resume Exit_CmdSelect_Click
```

```
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
frmDYNASTIES.Form.OrderBy = "c_start"
frmDYNASTIES.Form.OrderByOn = True
```

```
If Not IsNull(Me.OpenArgs) Then
Dim strDy As String
strDy = Me.OpenArgs
Dim rsDy As DAO.Recordset
Set rsDy = frmDYNASTIES.Form.Recordset
rsDy.FindFirst "c_dy = " & strDy
End If
```

```
End Sub
```

```
Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click
```

```
Dim StrSearch As String
Me.TxtSearch.SetFocus
StrSearch = Me.TxtSearch.Value
If StrSearch <> "" Then
Dim rsDy As DAO.Recordset
Set rsDy = frmDYNASTIES.Form.Recordset
Dim StrSearchStr As String
StrSearchStr = "c_dynasty_chn = " & Chr(34) & StrSearch & Chr(34)
rsDy.FindFirst StrSearchStr
If rsDy.NoMatch Then
StrSearchStr = "c_dynasty = " & Chr(34) & StrSearch & Chr(34)
rsDy.FindFirst StrSearchStr
End If
End If
```

```
Exit_CmdFind_Click:
Exit Sub
```

```
Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()
If Me.TxtSearch.Text = "" Then
Me.CmdFind.Enabled = False
Else
Me.CmdFind.Enabled = True
End If
End Sub
```

```

Option Compare Database
Public gRstEntryCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String
'#####Treeview Code#####
'Add this to your form's declaration section
Public WithEvents mcTree As clsTreeview
Private mbExit As Boolean ' to exit a SpinButton event
'#####Treeview Code#####

Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click

    Clear_SelectAll
    DoCmd.Close

Exit_CmdCancel_Click:
    Exit Sub

Err_CmdCancel_Click:
    MsgBox Err.Description
    Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdFindNext_Click()
    Dim tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset, cNode As clsNode
    Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

    TxtEntryDesc.Value = ""
    TxtEntryChn.Value = ""

    If Not IsNull(gRstEntryCodes) Then
        If Not (gStrSearch = "") Then

            'MsgBox "Looking for entry"

            If Not gUseAlt Then
                gRstEntryCode.FindNext gStrSearch

                If gRstEntryCode.NoMatch Then
                    gRstEntryCode.FindFirst gStrSearchAlt
                    gUseAlt = True
                End If
            Else
                gRstEntryCode.FindNext gStrSearchAlt
            End If

            If gRstEntryCode.NoMatch Then
                If gUseAlt Then
                    gUseAlt = False
                End If
                CmdFindNext.Enabled = False
            Else
                '
                ' next find the entry_type
                '
                'MsgBox "Looking for entry type"

                Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)

                tRstEntryTypes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)

                If Not tRstEntryTypes.NoMatch Then
                    '
                    ' update the code value
                    '
                    TxtEntryCode.Value = gRstEntryCode!c_entry_code
                    If Not IsNull(gRstEntryCode!c_entry_desc) Then
                        TxtEntryDesc.Value = gRstEntryCode!c_entry_desc
                    End If
                    If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then
                        TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn
                    End If
                    '
                    ' define the string
                    '
                    'MsgBox "Looking for node"
                    '
                    tStr = Trim(tRstEntryTypes!c_entry_type)
                    '

```

```

' search the tree
Set cNode = mcTree.Nodes(tStr)
'
If Not IsNull(cNode) Then
    If cNode.Key = gNode.Key Then
        Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
        tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
        frmZZZ_ENTRY_CODE.Form.Refresh
    Else
        '
        'MsgBox "found node"
        '
        Set mcTree.ActiveNode = cNode
        'tNode.Selected = True
        '
        ' then one makes it visible
        '
        'tNode.EnsureVisible
        '
        Set gNode = cNode
        '
        ' Finally populate the options and select the record.
        '
        CmdSelectAll.Enabled = True
        '
        tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) "
+ _
        "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
+ _
        "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
+ _
        "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = "
+ _
        "ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
+ _
        "WHERE (((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

        Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
        Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
        Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstDummy
        tRstEntryCodes.Close
        '
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText

        cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
        cmdSQL.Execute tRecDeleted

        'MsgBox tStrSQL

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
        Set tRstEntryCodes = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
        '
        tRstEntryCodes.MoveFirst
        Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCodes
        tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
        frmZZZ_ENTRY_CODE.Form.Refresh
        '
        Set tRstDummy = Nothing
        '
        ' set the type values
        '
        Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
        '
        tRstEntryTypes.MoveFirst
        tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key
        TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
        TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
        tRstEntryTypes.Close
        Set tRstEntryTypes = Nothing
    End If
End If
End If
End If
End If
End Sub

Private Sub CmdSelect_Click()

```

frmPickEntry - 3

```
Dim tRst As DAO.Recordset
Clear_SelectAll
CmdSelectAll.SetFocus
CmdSelect.Enabled = False

' Set tRst = frmZZZ_ENTRY_CODE.Form.Recordset
'
' when someone clicks on an entry record, the following data is put into the form:
'TxtEntryCode.Value = tRst!c_entry_code
'TxtEntryDesc.Value = tRst!c_entry_desc
'TxtEntryChn.Value = tRst!c_entry_desc_chn
'TxtTypeID.Value = ""
'TxtTypeDesc.Value = ""
'TxtTypeChn.Value = ""

Forms!frmPickEntry.Visible = False
End Sub

Private Sub CmdSelectAll_Click()
Dim tRst As DAO.Recordset

If CmdSelectAll.Caption = "Select All" Then
    CmdSelectAll.Caption = "De-select All"
    frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(255, 255, 255)
    frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(0, 0, 0)
    CmdSelect.Enabled = True
    Me.TxtEntryCode.Value = -1
    Me.TxtEntryDesc.Value = ""
    Me.TxtEntryChn.Value = ""
Else
    Set tRst = frmZZZ_ENTRY_CODE.Form.Recordset
    '
    TxtEntryCode.Value = tRst!c_entry_code
    TxtEntryDesc.Value = tRst!c_entry_desc
    TxtEntryChn.Value = tRst!c_entry_desc_chn
    CmdSelectAll.SetFocus
    Clear_SelectAll
End If
End Sub

Private Sub Clear_SelectAll()
CmdSelectAll.Caption = "Select All"
'
' reset the form colors
'
frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
End Sub

Private Sub Form_Open(Cancel As Integer)
Dim tStrEntry As String
Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset
' Courtesy Hans Vogelaar
' Populate the treeview with data from the tables
Dim cRoot As clsNode
' Four levels of nodes
Dim cNode1 As clsNode
Dim cNode2 As clsNode
Dim cNode3 As clsNode
Dim cNode4 As clsNode
' Key and caption for the nodes
Dim strKey As String
Dim strCaption As String
'
' initialize the type values
'
TxtTypeID.Value = ""
TxtTypeDesc.Value = "All"
TxtTypeChn.Value = "All"
'
' initialize the Entry Codes dataset
'
Set gRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)
Set frmZZZ_ENTRY_CODE.Form.Recordset = gRstEntryCode
'
Set tRstEntry = frmZZZ_ENTRY_CODE.Form.Recordset
'
If Not IsNull(Me.OpenArgs) Then
    tStrEntry = Me.OpenArgs
    tRstEntry.FindFirst "c_entry_code = " & tStrEntry
End If
'
```



```

' set the language
Dim tmli As MsoLanguageID
'
tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
'
If tmli = msoLanguageIDSimplifiedChinese Then
    gDisplayLanguage = "S"
ElseIf tmli = msoLanguageIDTraditionalChinese Then
    gDisplayLanguage = "T"
Else
    gDisplayLanguage = "E"
End If
'
' build treeview
'
Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
tRst.MoveFirst
Set mcTree = Me.subTreeView.Form.pTreeView
With mcTree
    .NodesClear
    .AppName = AppName ' Title for message boxes:
    ' Add a Root node with main and expanded icons and make it bold
    ' use the appropriate caption
    If gDisplayLanguage = "T" Then
        strCaption = ChrW(20837) + ChrW(20181) + ChrW(36884) + ChrW(24465) + ChrW(20998) + ChrW(39006)
    ElseIf gdisplaylangauge = "S" Then
        strCaption = ChrW(20837) + ChrW(20181) + ChrW(36884) + ChrW(24452) + ChrW(20998) + ChrW(31612)
    Else
        strCaption = "Categories of Modes of Entry"
    End If

    Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
    cRoot.Bold = True
    ' Loop through the records
    Do While Not tRst.EOF
        ' Add node
        strKey = tRst!c_entry_type
        If gDisplayLanguage = "E" Then
            strCaption = tRst!c_entry_type_desc
        Else
            strCaption = tRst!c_entry_type_desc_chn
        End If
        'If Len(tRst!c_entry_type) = 2 Then
        '    Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
        '    cNode1.Expanded = False
        'Else
        '    Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
        '    cNode2.Expanded = False
        ' Move to next date
        'End If
        Select Case Len(strKey)
            Case 2
                Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode1.Expanded = False
            Case 4
                Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode2.Expanded = False
            Case 6
                Set cNode3 = cNode2.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode3.Expanded = False
            Case 8
                Set cNode4 = cNode3.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode4.Expanded = False
        End Select
        tRst.MoveNext
    Loop
    ' Create the node controls and display the tree
    .Refresh
End With
Set tRst = Nothing
'
CmdSelect.Enabled = False
frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
tRstEntry.MoveFirst
End Sub
Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

    Call NodeSearch

```

```
Exit_CmdFind_Click:
```

```
Exit Sub
```

```
Err_CmdFind_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub mcTree_Click(cNode As clsNode)
```

```
Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset
```

```
Dim tRstEntryCode As DAO.Recordset, tRstDummy As DAO.Recordset
```

```
Dim tStrSQL As String
```

```
Set cmdSQL = New ADODB.Command
```

```
cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
cmdSQL.CommandType = adCmdText
```

```
Me.TxtEntryCode.Value = -1
```

```
Me.TxtEntryDesc.Value = ""
```

```
Me.TxtEntryChn.Value = ""
```

```
CmdSelect.Enabled = False
```

```
' reset the form colors
```

```
,
```

```
Clear_SelectAll
```

```
,
```

```
If cNode.Key = "Root" Then
```

```
    TxtTypeID.Value = ""
```

```
    TxtTypeDesc.Value = ""
```

```
    TxtTypeChn.Value = ""
```

```
    CmdSelectAll.Enabled = False
```

```
,
```

```
    ' reset the entry code choices
```

```
,
```

```
    Set tRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)
```

```
,
```

```
    Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCode
```

```
    frmZZZ_ENTRY_CODE.Form.Refresh
```

```
Else
```

```
    Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
```

```
,
```

```
    tRst.MoveFirst
```

```
    tRst.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
```

```
    TxtTypeID.Value = cNode.Key
```

```
    TxtTypeDesc.Value = tRst!c_entry_type_desc
```

```
    TxtTypeChn.Value = tRst!c_entry_type_desc_chn
```

```
    tRst.Close
```

```
    Set tRst = Nothing
```

```
,
```

```
    CmdSelectAll.Enabled = True
```

```
,
```

```
    ' clear the form
```

```
,
```

```
    Set tRstEntryCode = frmZZZ_ENTRY_CODE.Form.Recordset
```

```
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
```

```
    Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstDummy
```

```
,
```

```
    tRstEntryCode.Close
```

```
,
```

```
    cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
```

```
    cmdSQL.Execute tRecDeleted
```

```
,
```

```
    ' Refresh ZZ_ENTRY_CODE: we need to distinguish between type / subtype
```

```
,
```

```
tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
    "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
    "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
    "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
    "ENTRY_CODE_TYPE_REL.c_ENTRY_code "
```

```
If Len(cNode.Key) = 2 Then
```

```
    tStrSQL = tStrSQL + "WHERE (((Left((ENTRY_CODE_TYPE_REL.c_ENTRY_type),2))= '" + _
    TxtTypeID.Value + "'))"
```

```
ElseIf Len(cNode.Key) = 4 Then
```

```
    tStrSQL = tStrSQL + "WHERE (((Left((ENTRY_CODE_TYPE_REL.c_ENTRY_type),4))= '" + _
    TxtTypeID.Value + "'))"
```

```
Else
```

```
    tStrSQL = tStrSQL + "WHERE (ENTRY_CODE_TYPE_REL.c_ENTRY_type = '" + _
    TxtTypeID.Value + "'))"
```

```

        End If
    '
    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    '
    Set tRstEntryCode = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
    '
    Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCode
    '
    Set tRstDummy = Nothing
    '
End If
End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Then
        If TxtSearchChn.Value = "" Then
            CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Then
        If TxtSearch.Value = "" Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearch.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub NodeSearch()

    Dim cNode As clsNode, tStr As String, tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrLen As String, cmdSQL As ADODB.Command

    ' all specific association codes will have a type of the form 0101
    ' hence the ValuePath for the relevant node will be "K000/K01/K0101"
    ' The command to locate the relevant node is:
    ' cNode = mcTree.FindNode(tStrValuePath)
    ' mcTree.activeNode = cNode
    '
    TxtEntryDesc.Value = ""
    TxtEntryChn.Value = ""
    '
    ' search for the search string in ASSOC_CODES
    TxtSearchChn.SetFocus
    tStrSearchChn = Trim(Me.TxtSearchChn.Text)
    TxtSearch.SetFocus
    tStrSearchEng = Trim(Me.TxtSearch.Text)
    CmdFind.SetFocus
    '
    ' because the user may have a hard time picking the exact term, I'll treat it as
    ' (1) the beginning of the actual term
    ' (2) part of the term
    '
    gUseAlt = False
    gStrSearch = ""
    If tStrSearchChn <> "" Then
        tStrLen = Str(LenB(tStrSearchChn))
        gStrSearch = "LeftB(c_entry_desc_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
        gStrSearchAlt = "InStrB(1,c_entry_desc_chn,'" + tStrSearchChn + "') > 0"
        'tStrSearch = "c_entry_desc_chn = '" + tStrSearchChn + "'"
    ElseIf tStrSearchEng <> "" Then
        tStrLen = Str(Len(tStrSearchEng))
        gStrSearch = "Left(c_entry_desc," + tStrLen + ") = '" + tStrSearchEng + "'"
        gStrSearchAlt = "InStr(1,c_entry_desc,'" + tStrSearchEng + "') > 0"
        'tStrSearch = "c_entry_desc = '" + tStrSearchEng + "'"
    End If

    If Not (gStrSearch = "") Then

        'MsgBox "Looking for entry"
    End If

```

```

Set gRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)

gRstEntryCode.FindFirst gStrSearch

If gRstEntryCode.NoMatch Then
    gRstEntryCode.FindFirst gStrSearchAlt
    gUseAlt = True
End If

If gRstEntryCode.NoMatch Then
    If gUseAlt Then
        gUseAlt = False
    End If
    CmdFindNext.Enabled = False
Else
    CmdFindNext.Enabled = True
    '
    ' next find the entry_type
    '
    'MsgBox "Looking for entry type"

Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)

tRstEntryTypes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)

If Not tRstEntryTypes.NoMatch Then
    '
    ' set the code values
    '
    TxtEntryCode.Value = gRstEntryCode!c_entry_code
    If Not IsNull(gRstEntryCode!c_entry_desc) Then
        TxtEntryDesc.Value = gRstEntryCode!c_entry_desc
    End If
    If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then
        TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn
    End If
    '
    ' define the string
    '
    'MsgBox "Looking for node"
    '
    tStr = Trim(tRstEntryTypes!c_entry_type)
    '
    ' search the tree

Set cNode = mcTree.Nodes(tStr)
'
If Not IsNull(cNode) Then
    '
    'MsgBox "found node"
    '
    Set mcTree.ActiveNode = cNode
    'cNode.Selected = True
    '
    ' then one makes it visible
    '
    'cNode.EnsureVisible = True
    '
    Set gNode = cNode
    '
    ' Finally populate the options and select the record.
    '
    CmdSelectAll.Enabled = True
    '
    tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
        "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
        "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
        "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
        "ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
        "WHERE (((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstDummy
tRstEntryCodes.Close
'

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

```

```
cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
cmdSQL.Execute tRecDeleted

'MsgBox tStrSQL

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted
'
Set tRstEntryCodes = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
'
tRstEntryCodes.MoveFirst
Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCodes
tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
frmZZZ_ENTRY_CODE.Form.Requery
'
Set tRstDummy = Nothing
'
' set the type values
'
Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
'
tRstEntryTypes.MoveFirst
tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
TxtTypeID.Value = cNode.Key
TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
tRstEntryTypes.Close
Set tRstEntryTypes = Nothing
End If
End If
End If
End If
```

End Sub

Option Compare Database

```
Public gRstEntryCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String, gSelectCount As Integer
```

```
'#####Treeview Code#####
```

```
'Add this to your form's declaration section
```

```
Public WithEvents mcTree As clsTreeview
```

```
Private mbExit As Boolean ' to exit a SpinButton event
```

```
'#####Treeview Code#####
```

```
Private Sub CmdCancel_Click()
```

```
On Error GoTo Err_CmdCancel_Click
```

```
Clear_SelectAll
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
```

```
Exit Sub
```

```
Err_CmdCancel_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdFindNext_Click()
```

```
Dim tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset, cNode As clsNode, ti As Integer
Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command
```

```
TxtEntryDesc.Value = ""
```

```
TxtEntryChn.Value = ""
```

```
If Not IsNull(gRstEntryCodes) Then
```

```
    If Not (gStrSearch = "") Then
```

```
        'MsgBox "Looking for entry"
```

```
        If Not gUseAlt Then
```

```
            gRstEntryCode.FindNext gStrSearch
```

```
            If gRstEntryCode.NoMatch Then
```

```
                gRstEntryCode.FindFirst gStrSearchAlt
```

```
                gUseAlt = True
```

```
            End If
```

```
        Else
```

```
            gRstEntryCode.FindNext gStrSearchAlt
```

```
        End If
```

```
        If gRstEntryCode.NoMatch Then
```

```
            If gUseAlt Then
```

```
                gUseAlt = False
```

```
            End If
```

```
            CmdFindNext.Enabled = False
```

```
        Else
```

```
            ,
```

```
            ' next find the entry_type
```

```
            ,
```

```
            'MsgBox "Looking for entry type"
```

```
            Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)
```

```
            tRstEntryTypes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
```

```
            If Not tRstEntryTypes.NoMatch Then
```

```
                ,
```

```
                ' update the code value
```

```
                ,
```

```
                TxtEntryCode.Value = gRstEntryCode!c_entry_code
```

```
                If Not IsNull(gRstEntryCode!c_entry_desc) Then
```

```
                    TxtEntryDesc.Value = gRstEntryCode!c_entry_desc
```

```
                End If
```

```
                If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then
```

```
                    TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn
```

```
                End If
```

```
                ,
```

```
                ' define the string
```

```
                ,
```

```
                'MsgBox "Looking for node"
```

```
                ,
```

```
                tStr = Trim(tRstEntryTypes!c_entry_type)
```

```
                ,
```

```

' search the tree
Set cNode = mcTree.Nodes(tStr)
'
If Not IsNull(cNode) Then
    If cNode.Key = gNode.Key Then
        For i = 0 To ListEntry.ListCount - 1
            If gRstEntryCode!c_entry_code = ListEntry.Column(0, i) Then
                ListEntry.ListIndex = i
                ListEntry.Selected(i) = True
            End If
        Next i
    Else
        'MsgBox "found node"
        '
        Set mcTree.ActiveNode = cNode
        'tNode.Selected = True
        '
        ' then one makes it visible
        '
        'tNode.EnsureVisible
        '
        Set gNode = cNode
        '
        ' Finally populate the options and select the record.
        '
        CmdSelectAll.Enabled = True
        '
        tStrSQL = "INSERT INTO ZZ_ENTRY_CODE_TMP (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_ch
n) " + _
        "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
        "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
        "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = "
        "ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
        "WHERE (((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText

        cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE_TMP"
        cmdSQL.Execute tRecDeleted

        'MsgBox tStrSQL

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
        ListEntry.Requery
        For ti = 0 To ListEntry.ListCount
            ListEntry.Selected(ti) = False
        Next ti
        gSelectCount = 0
        '
        ' set the type values
        '
        Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
        '
        tRstEntryTypes.MoveFirst
        tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key
        TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
        TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
        tRstEntryTypes.Close
        Set tRstEntryTypes = Nothing
    End If
End If
End If
End If
End If
End Sub

Private Sub CmdSelect_Click()
    Dim tRst As DAO.Recordset
    Dim cmdSQL As ADODB.Command, tRecCount As Long, varItm As Variant, ti As Integer

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection

```

Form_frmPickEntry_multi - 3

```
cmdSQL.CommandType = adCmdText

gSelectCount = 0
For Each varItm In ListEntry.ItemsSelected
    gSelectCount = gSelectCount + 1
Next varItm
'
' MsgBox "gSelectCount = " + Str(gSelectCount) + " ListCount = " + Str(ListEntry.ListCount)
'
' put the description in the boxes
'
If gSelectCount = 0 Then
    Me.TxtEntryDesc.Value = ""
    Me.TxtEntryChn.Value = ""
    Me.TxtEntryCode.Value = 0
    Me.CmdSelect.Enabled = False
ElseIf gSelectCount = ListEntry.ListCount - 1 Then
    Me.TxtEntryDesc.Value = "All"
    Me.TxtEntryChn.Value = "All"
    Me.TxtEntryCode.Value = -1

    cmdSQL.CommandText = "DELETE * FROM ZZ_ENTRY_CODE"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "INSERT INTO ZZ_ENTRY_CODE ( c_entry_code, c_entry_desc, c_entry_desc_chn ) " +
        "SELECT ZZ_ENTRY_CODE_TMP.c_entry_code, ZZ_ENTRY_CODE_TMP.c_entry_desc, ZZ_ENTRY_COD
E_TMP.c_entry_desc_chn " + " - "
        "FROM ZZ_ENTRY_CODE_TMP"
    cmdSQL.Execute tRecCount
Else
    If gSelectCount = 1 Then
        '
        ' there is just one item in the collection
        '
        For Each varItm In ListEntry.ItemsSelected
            Me.TxtEntryDesc.Value = ListEntry.Column(1, varItm)
            Me.TxtEntryChn.Value = ListEntry.Column(2, varItm)
            Me.TxtEntryCode.Value = ListEntry.Column(0, varItm)
        Next varItm
    Else
        Me.TxtEntryDesc.Value = "Multi-Select"
        Me.TxtEntryChn.Value = "Multi-Select"
        Me.TxtEntryCode.Value = -2
    End If
    '
    ' now process the records
    '
    cmdSQL.CommandText = "DELETE * FROM ZZ_ENTRY_CODE"
    cmdSQL.Execute tRecCount

    ' first copy the records over to a scratch table

    Set tRst = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
    For Each varItm In ListEntry.ItemsSelected
        tRst.AddNew
        tRst!c_entry_code = ListEntry.Column(0, varItm)
        tRst!c_entry_desc = ListEntry.Column(1, varItm)
        tRst!c_entry_desc_chn = ListEntry.Column(2, varItm)
        tRst.Update
    Next varItm
    tRst.Close

    ListEntry.Requery
    'For ti = 0 To ListEntry.ListCount
    '    ListEntry.Selected(ti) = False
    'Next ti

End If

Clear_SelectAll
CmdSelectAll.SetFocus
CmdSelect.Enabled = False

Forms!frmPickEntry_multi.Visible = False
End Sub

Private Sub CmdSelectAll_Click()
    Dim tRst As DAO.Recordset

    If CmdSelectAll.Caption = "Select All" Then
```



```

        CmdSelectAll.Caption = "De-select All"

        For ti = 0 To ListEntry.ListCount
            ListEntry.Selected(ti) = True
        Next ti

        CmdSelect.Enabled = True
        TxtEntryCode.Value = -1
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub

Private Sub Clear_SelectAll()
    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    For ti = 0 To ListEntry.ListCount
        ListEntry.Selected(ti) = False
    Next ti

    gSelectCount = 0
    CmdSelect.Enabled = False
End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim tStrEntry As String
    Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset, cmdSQL As ADODB.Command, tRecCount As Long, ti As Integer
    ' Courtesy Hans Vogelaar
    ' Populate the treeview with data from the tables
    Dim cRoot As clsNode
    ' Four levels of nodes
    Dim cNode1 As clsNode
    Dim cNode2 As clsNode
    Dim cNode3 As clsNode
    Dim cNode4 As clsNode
    ' Key and caption for the nodes
    Dim strKey As String
    Dim strCaption As String
    '
    ' initialize the type values
    '
    TxtTypeID.Value = ""
    TxtTypeDesc.Value = "All"
    TxtTypeChn.Value = "All"
    gSelectCount = 0
    '
    ' initialize the listbox
    '
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    cmdSQL.CommandText = "DELETE * FROM ZZ_ENTRY_CODE_TMP"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "INSERT INTO ZZ_ENTRY_CODE_TMP ( c_entry_code, c_entry_desc, c_entry_desc_chn ) " + _
        "SELECT ENTRY_CODES.c_entry_code, ENTRY_CODES.c_entry_desc, ENTRY_CODES.c_entry_desc_chn " + _
        "FROM ENTRY_CODES"
    cmdSQL.Execute tRecCount

    ListEntry.Requery
    For ti = 0 To ListEntry.ListCount
        ListEntry.Selected(ti) = False
    Next ti
    ListEntry.Requery
    '
    ' set the language
    Dim tmli As MsoLanguageID
    '
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    '
    If tmli = msoLanguageIDSimplifiedChinese Then
        gDisplayLanguage = "S"
    ElseIf tmli = msoLanguageIDTraditionalChinese Then
        gDisplayLanguage = "T"
    Else

```

```

        gDisplayLanguage = "E"
    End If
    '
    ' build treeview
    '
    Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
    tRst.MoveFirst
    Set mcTree = Me.subTreeView.Form.pTreeView
    With mcTree
        .NodesClear
        .AppName = AppName ' Title for message boxes:
        ' Add a Root node with main and expanded icons and make it bold
        ' use the appropriate caption
        If gDisplayLanguage = "T" Then
            strCaption = ChrW(20837) + ChrW(20181) + ChrW(36884) + ChrW(24465) + ChrW(20998) + ChrW(39006)
        ElseIf gdisplaylangauge = "S" Then
            strCaption = ChrW(20837) + ChrW(20181) + ChrW(36884) + ChrW(24452) + ChrW(20998) + ChrW(31612)
        Else
            strCaption = "Categories of Modes of Entry"
        End If

        Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
        cRoot.Bold = True
        ' Loop through the records
        Do While Not tRst.EOF
            ' Add node
            strKey = tRst!c_entry_type
            If gDisplayLanguage = "E" Then
                strCaption = tRst!c_entry_type_desc
            Else
                strCaption = tRst!c_entry_type_desc_chn
            End If
            'If Len(tRst!c_entry_type) = 2 Then
            '    Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
            '    cNode1.Expanded = False
            'Else
            '    Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
            '    cNode2.Expanded = False
            ' Move to next date
            'End If
            Select Case Len(strKey)
                Case 2
                    Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode1.Expanded = False
                Case 4
                    Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode2.Expanded = False
                Case 6
                    Set cNode3 = cNode2.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode3.Expanded = False
                Case 8
                    Set cNode4 = cNode3.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode4.Expanded = False
            End Select
            tRst.MoveNext
        Loop
        ' Create the node controls and display the tree
        .Refresh
    End With
    Set tRst = Nothing
    '
    CmdSelect.Enabled = False
End Sub

Private Sub CmdFind_Click()
    On Error GoTo Err_CmdFind_Click

    Call NodeSearch

Exit_CmdFind_Click:
    Exit Sub

Err_CmdFind_Click:
    MsgBox Err.Description
    Resume Exit_CmdFind_Click
End Sub

Private Sub ListEntry_Click()
    Dim varItm As Variant

```

frmPickEntry_multi - 6

```
gSelectCount = 0

For Each varItm In ListEntry.ItemsSelected
    gSelectCount = gSelectCount + 1
Next varItm

If gSelectCount = 0 Then
    Me.CmdSelect.Enabled = False
Else
    Me.CmdSelect.Enabled = True
End If
End Sub

Private Sub mcTree_Click(cNode As clsNode)
    Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset
    Dim tRstEntryCode As DAO.Recordset, tRstDummy As DAO.Recordset
    Dim tStrSQL As String, ti As Integer

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    Me.TxtEntryCode.Value = -1
    Me.TxtEntryDesc.Value = ""
    Me.TxtEntryChn.Value = ""
    CmdSelect.Enabled = False

    ' reset the form colors
    '
    Clear_SelectAll
    '
    If cNode.Key = "Root" Then
        TxtTypeID.Value = ""
        TxtTypeDesc.Value = ""
        TxtTypeChn.Value = ""
        CmdSelectAll.Enabled = False
        '
        ' reset the entry code choices
        '
        cmdSQL.CommandText = "DELETE * FROM ZZ_ENTRY_CODE_TMP"
        cmdSQL.Execute tRecCount

        cmdSQL.CommandText = "INSERT INTO ZZ_ENTRY_CODE_TMP ( c_entry_code, c_entry_desc, c_entry_desc_chn ) " + _
            "SELECT ENTRY_CODES.c_entry_code, ENTRY_CODES.c_entry_desc, ENTRY_CODES.c_entry_desc_chn " + _
            "FROM ENTRY_CODES"
        cmdSQL.Execute tRecCount

        ListEntry.Requery
        For ti = 0 To ListEntry.ListCount
            ListEntry.Selected(ti) = False
        Next ti
        gSelectCount = 0
    Else
        Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
        '
        tRst.MoveFirst
        tRst.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key
        TxtTypeDesc.Value = tRst!c_entry_type_desc
        TxtTypeChn.Value = tRst!c_entry_type_desc_chn
        tRst.Close
        Set tRst = Nothing
        CmdSelectAll.Enabled = True
        '
        cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE_TMP"
        cmdSQL.Execute tRecDeleted
        '
        ' Refresh ZZ_ENTRY_CODE_TMP: we need to distinguish between type / subtype
        '
        tStrSQL = "INSERT INTO ZZ_ENTRY_CODE_TMP ( c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
            "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
            "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
            "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
            "ENTRY_CODE_TYPE_REL.c_ENTRY_code "

        If Len(cNode.Key) = 2 Then
            tStrSQL = tStrSQL + "WHERE ((Left((ENTRY_CODE_TYPE_REL.c_ENTRY_type),2))= '" + _
```

frmPickEntry_multi - 7

```
        TxtTypeID.Value + "')))"
    ElseIf Len(cNode.Key) = 4 Then
        tStrSQL = tStrSQL + "WHERE (((Left((ENTRY_CODE_TYPE_REL.c_ENTRY_type),4))= '" + _
            TxtTypeID.Value + "')))"
    Else
        tStrSQL = tStrSQL + "WHERE (ENTRY_CODE_TYPE_REL.c_ENTRY_type = '" + _
            TxtTypeID.Value + "')"
    End If
    '
    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    '
    ListEntry.Requery
    For ti = 0 To ListEntry.ListCount
        ListEntry.Selected(ti) = False
    Next ti
    ListEntry.Requery
    gSelectCount = 0
    '
End If
End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Then
        If TxtSearchChn.Value = "" Then
            CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Then
        If TxtSearch.Value = "" Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearch.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub NodeSearch()
    Dim cNode As clsNode, tStr As String, tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrLen As String, cmdSQL As ADODB.Command, ti As Integer

    ' all specific entry codes will have a type of the form 0101
    ' hence the ValuePath for the relevant node will be "K000/K01/K0101"
    ' The command to locate the relevant node is:
    ' cNode = mcTree.FindNode(tStrValuePath)
    ' mcTree.activeNode = cNode
    '
    TxtEntryDesc.Value = ""
    TxtEntryChn.Value = ""
    '
    ' search for the search string in ENTRY_CODES
    TxtSearchChn.SetFocus
    tStrSearchChn = Trim(Me.TxtSearchChn.Text)
    TxtSearch.SetFocus
    tStrSearchEng = Trim(Me.TxtSearch.Text)
    CmdFind.SetFocus
    '
    ' because the user may have a hard time picking the exact term, I'll treat it as
    ' (1) the beginning of the actual term
    ' (2) part of the term
    '
    gUseAlt = False
    gStrSearch = ""
    If tStrSearchChn <> "" Then
        tStrLen = Str(LenB(tStrSearchChn))
        gStrSearch = "LeftB(c_entry_desc_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
        gStrSearchAlt = "InStrB(1,c_entry_desc_chn,'" + tStrSearchChn + "') > 0"
        'tStrSearch = "c_entry_desc_chn = '" + tStrSearchChn + "'"
    ElseIf tStrSearchEng <> "" Then
        tStrLen = Str(Len(tStrSearchEng))
    End If
```

```

    gStrSearch = "Left(c_entry_desc," + tStrLen + ") ='" + tStrSearchEng + "'"
    gStrSearchAlt = "InStr(1,c_entry_desc,'" + tStrSearchEng + "') > 0"
    'tStrSearch = "c_entry_desc = '" + tStrSearchEng + "'"
End If

'MsgBox gStrSearch
If Not (gStrSearch = "") Then

    'MsgBox "Looking for entry"

    Set gRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)

    gRstEntryCode.FindFirst gStrSearch

    If gRstEntryCode.NoMatch Then
        'MsgBox "No Match"
        gRstEntryCode.FindFirst gStrSearchAlt
        gUseAlt = True
    End If

    If gRstEntryCode.NoMatch Then
        'MsgBox "Still no match"
        If gUseAlt Then
            gUseAlt = False
        End If
        CmdFindNext.Enabled = False
    Else
        CmdFindNext.Enabled = True
        '
        ' next find the entry_type
        '
        'MsgBox "Looking for entry type"

        Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)

        tRstEntryTypes.FindFirst "c_entry_code = " + Str(gRstEntryCode!c_entry_code)

        If tRstEntryTypes.NoMatch Then
            'MsgBox "No entry type found for " + Str(gRstEntryCode!c_entry_code)
        Else
            '
            ' set the code values
            '
            'MsgBox "Match found"
            TxtEntryCode.Value = gRstEntryCode!c_entry_code
            If Not IsNull(gRstEntryCode!c_entry_desc) Then
                TxtEntryDesc.Value = gRstEntryCode!c_entry_desc
            End If
            If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then
                TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn
            End If
            '
            ' define the string
            '
            'MsgBox "Looking for node"
            '
            tStr = Trim(tRstEntryTypes!c_entry_type)
            'MsgBox "Entry type = " + tstr
            '
            ' search the tree

            Set cNode = mcTree.Nodes(tStr)
            '
            If Not IsNull(cNode) Then
                '
                'MsgBox "found node"
                '
                Set mcTree.ActiveNode = cNode
                'cNode.Selected = True
                '
                ' then one makes it visible
                '
                'cNode.EnsureVisible = True
                '
                Set gNode = cNode
                '
                ' Finally populate the options and select the record.
                '
                CmdSelectAll.Enabled = True
            End If
        End If
    End If

```

```

tStrSQL = "INSERT INTO ZZ_ENTRY_CODE_TMP (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
"SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
"ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
"FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
"ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
"WHERE (((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

```

```

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

```

```

cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE_TMP"
cmdSQL.Execute tRecDeleted

```

```

'MsgBox tStrSQL

```

```

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

```

```

ListEntry.Requery
For ti = 0 To ListEntry.ListCount
    ListEntry.Selected(ti) = False
Next ti
gSelectCount = 0

```

```

For ti = 0 To ListEntry.ListCount - 1
    If gRstEntryCode!c_entry_code = ListEntry.Column(0, ti + 1) Then
        ListEntry.ListIndex = ti + 1
        ListEntry.Selected(ti + 1) = True
    End If
Next ti

```

```

' set the type values

```

```

Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)

```

```

tRstEntryTypes.MoveFirst
tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
TxtTypeID.Value = cNode.Key
TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
tRstEntryTypes.Close
Set tRstEntryTypes = Nothing

```

```

End If

```

```

End If

```

```

End If

```

```

End If

```

```

End Sub

```

Option Compare Database

```
Public gRstEntryCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String
'#####Treeview Code#####
'Add this to your form's declaration section
Public WithEvents mcTree As clsTreeview
Private mbExit As Boolean ' to exit a SpinButton event
'#####Treeview Code#####
```

```
Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click
```

```
Clear_SelectAll
DoCmd.Close
```

```
Exit_CmdCancel_Click:
Exit Sub
```

```
Err_CmdCancel_Click:
MsgBox Err.Description
Resume Exit_CmdCancel_Click
```

End Sub

```
Private Sub CmdFindNext_Click()
Dim tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset, cNode As clsNode
Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command
```

```
TxtEntryDesc.Value = ""
TxtEntryChn.Value = ""
```

```
If Not IsNull(gRstEntryCodes) Then
If Not (gStrSearch = "") Then
```

```
MsgBox "Looking for entry"
```

```
If Not gUseAlt Then
gRstEntryCode.FindNext gStrSearch
```

```
If gRstEntryCode.NoMatch Then
gRstEntryCode.FindFirst gStrSearchAlt
gUseAlt = True
End If
```

```
Else
gRstEntryCode.FindNext gStrSearchAlt
End If
```

```
If gRstEntryCode.NoMatch Then
If gUseAlt Then
gUseAlt = False
End If
CmdFindNext.Enabled = False
```

```
Else
,
' next find the entry_type
,
MsgBox "Looking for entry type"
```

```
Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)
```

```
tRstEntryTypes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
```

```
If Not tRstEntryTypes.NoMatch Then
```

```
,
' update the code value
,
```

```
TxtEntryCode.Value = gRstEntryCode!c_entry_code
If Not IsNull(gRstEntryCode!c_entry_desc) Then
TxtEntryDesc.Value = gRstEntryCode!c_entry_desc
End If
```

```
If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then
TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn
End If
```

```
,
' define the string
,
```

```
MsgBox "Looking for node"
```

```
tStr = Trim(tRstEntryTypes!c_entry_type)
```

```

' search the tree
Set cNode = mcTree.Nodes(tStr)
'
If Not IsNull(cNode) Then
    If cNode.Key = gNode.Key Then
        Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
        tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
        frmZZZ_ENTRY_CODE.Form.LB_Entry.Refresh
    Else
        '
        'MsgBox "found node"
        '
        Set mcTree.ActiveNode = cNode
        'tNode.Selected = True
        '
        ' then one makes it visible
        '
        'tNode.EnsureVisible
        '
        Set gNode = cNode
        '
        ' Finally populate the options and select the record.
        '
        CmdSelectAll.Enabled = True
        '
        tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) "
+ _
        "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
+ _
        "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
        "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = "
+ _
        "ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
        "WHERE (((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

        Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset
        Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
        Set frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset = tRstDummy
        tRstEntryCodes.Close
        '
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText

        cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
        cmdSQL.Execute tRecDeleted

        'MsgBox tStrSQL

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
        Set tRstEntryCodes = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
        '
        tRstEntryCodes.MoveFirst
        Set frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset = tRstEntryCodes
        tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
        frmZZZ_ENTRY_CODE.Form.LB_Entry.Refresh
        '
        Set tRstDummy = Nothing
        '
        ' set the type values
        '
        Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
        '
        tRstEntryTypes.MoveFirst
        tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key
        TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
        TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
        tRstEntryTypes.Close
        Set tRstEntryTypes = Nothing
    End If
End If
End If
End If
End If
End Sub

Private Sub CmdSelect_Click()

```


frmPickEntry2 - 3

```
Clear_SelectAll
CmdSelectAll.SetFocus
CmdSelect.Enabled = False
'
'   when someone clicks on an entry record, the following data is put into the form:
'   TxtEntryCode.Value = tRst!c_entry_code
'   TxtEntryDesc.Value = tRst!c_entry_desc
'   TxtEntryChn.Value = tRst!c_entry_desc_chn
'   TxtTypeID.Value = ""
'   TxtTypeDesc.Value = ""
'   TxtTypeChn.Value = ""
```

```
Forms!frmPickEntry2.Visible = False
End Sub
```

```
Private Sub CmdSelectAll_Click()
    If CmdSelectAll.Caption = "Select All" Then
        CmdSelectAll.Caption = "De-select All"
        frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(255, 255, 255)
        frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(0, 0, 0)
        CmdSelect.Enabled = True
        Me.TxtEntryCode.Value = -1
        Me.TxtEntryDesc.Value = ""
        Me.TxtEntryChn.Value = ""
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub
```

```
Private Sub Clear_SelectAll()
    CmdSelectAll.Caption = "Select All"
    '
    '   reset the form colors
    '
    frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
    frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
    Dim tStrEntry As String
    Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset
    ' Courtesy Hans Vogelaar
    ' Populate the treeview with data from the tables
    Dim cRoot As clsNode
    ' Four levels of nodes
    Dim cNode1 As clsNode
    Dim cNode2 As clsNode
    Dim cNode3 As clsNode
    Dim cNode4 As clsNode
    ' Key and caption for the nodes
    Dim strKey As String
    Dim strCaption As String
    '
    '   initialize the type values
    '
    TxtTypeID.Value = ""
    TxtTypeDesc.Value = "All"
    TxtTypeChn.Value = "All"
    '
    '   initialize the Entry Codes dataset
    '
    Set gRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)
    Set frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset = gRstEntryCode
    '
    Set tRstEntry = frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset
    '
    If Not IsNull(Me.OpenArgs) Then
        tStrEntry = Me.OpenArgs
        tRstEntry.FindFirst "c_entry_code = " & tStrEntry
    End If
    '
    '   set the language
    Dim tmli As MsoLanguageID
    '
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    '
    If tmli = msoLanguageIDSimplifiedChinese Then
        gDisplayLanguage = "S"
    ElseIf tmli = msoLanguageIDTraditionalChinese Then
        gDisplayLanguage = "T"
    Else
```

```

        gDisplayLanguage = "E"
    End If
    '
    ' build treeview
    '
    Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
    tRst.MoveFirst
    Set mcTree = Me.subTreeView.Form.pTreeView
    With mcTree
        .NodesClear
        .AppName = AppName ' Title for message boxes:
        ' Add a Root node with main and expanded icons and make it bold
        ' use the appropriate caption
        If gDisplayLanguage = "T" Then
            strCaption = ChrW(20837) + ChrW(20181) + ChrW(36884) + ChrW(24465) + ChrW(20998) + ChrW(39006)
        ElseIf gdisplaylangauge = "S" Then
            strCaption = ChrW(20837) + ChrW(20181) + ChrW(36884) + ChrW(24452) + ChrW(20998) + ChrW(31612)
        Else
            strCaption = "Categories of Modes of Entry"
        End If

        Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
        cRoot.Bold = True
        ' Loop through the records
        Do While Not tRst.EOF
            ' Add node
            strKey = tRst!c_entry_type
            If gDisplayLanguage = "E" Then
                strCaption = tRst!c_entry_type_desc
            Else
                strCaption = tRst!c_entry_type_desc_chn
            End If
            'If Len(tRst!c_entry_type) = 2 Then
            '    Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
            '    cNode1.Expanded = False
            'Else
            '    Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
            '    cNode2.Expanded = False
            ' Move to next date
            'End If
            Select Case Len(strKey)
                Case 2
                    Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode1.Expanded = False
                Case 4
                    Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode2.Expanded = False
                Case 6
                    Set cNode3 = cNode2.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode3.Expanded = False
                Case 8
                    Set cNode4 = cNode3.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode4.Expanded = False
            End Select
            tRst.MoveNext
        Loop
        ' Create the node controls and display the tree
        .Refresh
    End With
    Set tRst = Nothing
    '
    CmdSelect.Enabled = False
    frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
    frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
    tRstEntry.MoveFirst
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

    Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
    MsgBox Err.Description
    Resume Exit_CmdFind_Click

End Sub

Private Sub mcTree_Click(cNode As clsNode)

```

```

Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset
Dim tRstEntryCode As DAO.Recordset, tRstDummy As DAO.Recordset
Dim tStrSQL As String

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

Me.TxtEntryCode.Value = -1
Me.TxtEntryDesc.Value = ""
Me.TxtEntryChn.Value = ""
CmdSelect.Enabled = False

' reset the form colors
,
Clear_SelectAll
,
If cNode.Key = "Root" Then
    TxtTypeID.Value = ""
    TxtTypeDesc.Value = ""
    TxtTypeChn.Value = ""
    CmdSelectAll.Enabled = False
    ,
    ' reset the entry code choices
    ,
    Set tRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)
    ,
    Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCode
    frmZZZ_ENTRY_CODE.Form.Refresh
Else
    Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
    ,
    tRst.MoveFirst
    tRst.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
    TxtTypeID.Value = cNode.Key
    TxtTypeDesc.Value = tRst!c_entry_type_desc
    TxtTypeChn.Value = tRst!c_entry_type_desc_chn
    tRst.Close
    Set tRst = Nothing
    ,
    CmdSelectAll.Enabled = True
    ,
    ' we need to distinguish between type / subtype
    ,
    tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
        "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
        "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
        "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
        "ENTRY_CODE_TYPE_REL.c_ENTRY_code "

    If Len(cNode.Key) = 2 Then
        tStrSQL = tStrSQL + "WHERE (((Left((ENTRY_CODE_TYPE_REL.c_ENTRY_type),2))= '" + _
            TxtTypeID.Value + "')))"
    ElseIf Len(cNode.Key) = 4 Then
        tStrSQL = tStrSQL + "WHERE (((Left((ENTRY_CODE_TYPE_REL.c_ENTRY_type),4))= '" + _
            TxtTypeID.Value + "')))"
    Else
        tStrSQL = tStrSQL + "WHERE (ENTRY_CODE_TYPE_REL.c_ENTRY_type = '" + _
            TxtTypeID.Value + "')"
    End If

    ,
    Set tRstEntryCode = frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
    Set frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset = tRstDummy
    ' frmZZZ_ENTRY_CODE.Form.LB_Entry.Refresh
    tRstEntryCode.Close
    ,
    cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
    cmdSQL.Execute tRecDeleted

    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    ,
    Set tRstEntryCode = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
    ,
    Set frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset = tRstEntryCode
    ' frmZZZ_ENTRY_CODE.Form.LB_Entry.Refresh
    Set tRstDummy = Nothing
    ,

```

```

frmPickEntry2 - 6

End If
End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Then
        If TxtSearchChn.Value = "" Then
            CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Then
        If TxtSearch.Value = "" Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearch.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub NodeSearch()

    Dim cNode As clsNode, tStr As String, tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrLen As String, cmdSQL As ADODB.Command

    ' all specific association codes will have a type of the form 0101
    ' hence the ValuePath for the relevant node will be "K000/K01/K0101"
    ' The command to locate the relevant node is:
    ' cNode = mcTree.FindNode(tStrValuePath)
    ' mcTree.activeNode = cNode
    '
    TxtEntryDesc.Value = ""
    TxtEntryChn.Value = ""
    '
    ' search for the search string in ASSOC_CODES
    TxtSearchChn.SetFocus
    tStrSearchChn = Trim(Me.TxtSearchChn.Text)
    TxtSearch.SetFocus
    tStrSearchEng = Trim(Me.TxtSearch.Text)
    CmdFind.SetFocus
    '
    ' because the user may have a hard time picking the exact term, I'll treat it as
    ' (1) the beginning of the actual term
    ' (2) part of the term
    '
    gUseAlt = False
    gStrSearch = ""
    If tStrSearchChn <> "" Then
        tStrLen = Str(LenB(tStrSearchChn))
        gStrSearch = "LeftB(c_entry_desc_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
        gStrSearchAlt = "InStrB(1,c_entry_desc_chn,'" + tStrSearchChn + "') > 0"
        'tStrSearch = "c_entry_desc_chn = '" + tStrSearchChn + "'"
    ElseIf tStrSearchEng <> "" Then
        tStrLen = Str(Len(tStrSearchEng))
        gStrSearch = "Left(c_entry_desc," + tStrLen + ") = '" + tStrSearchEng + "'"
        gStrSearchAlt = "InStr(1,c_entry_desc,'" + tStrSearchEng + "') > 0"
        'tStrSearch = "c_entry_desc = '" + tStrSearchEng + "'"
    End If

    If Not (gStrSearch = "") Then

        'MsgBox "Looking for entry"

        Set gRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)

        gRstEntryCode.FindFirst gStrSearch

        If gRstEntryCode.NoMatch Then
            gRstEntryCode.FindFirst gStrSearchAlt
            gUseAlt = True
        End If

        If gRstEntryCode.NoMatch Then

```

```

    If gUseAlt Then
        gUseAlt = False
    End If
    CmdFindNext.Enabled = False
Else
    CmdFindNext.Enabled = True
    '
    ' next find the entry_type
    '
    'MsgBox "Looking for entry type"

    Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)

    tRstEntryTypes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)

    If Not tRstEntryTypes.NoMatch Then
        '
        ' set the code values
        '
        TxtEntryCode.Value = gRstEntryCode!c_entry_code
        If Not IsNull(gRstEntryCode!c_entry_desc) Then
            TxtEntryDesc.Value = gRstEntryCode!c_entry_desc
        End If
        If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then
            TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn
        End If
        '
        ' define the string
        '
        'MsgBox "Looking for node"
        '
        tStr = Trim(tRstEntryTypes!c_entry_type)
        '
        ' search the tree

        Set cNode = mcTree.Nodes(tStr)
        '
        If Not IsNull(cNode) Then
            '
            'MsgBox "found node"
            '
            Set mcTree.ActiveNode = cNode
            'cNode.Selected = True
            '
            ' then one makes it visible
            '
            'cNode.EnsureVisible = True
            '
            Set gNode = cNode
            '
            ' Finally populate the options and select the record.
            '
            CmdSelectAll.Enabled = True
            '
            tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
                "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
                "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
                "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
                "ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
                "WHERE ((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

            Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset
            Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
            Set frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset = tRstDummy
            tRstEntryCodes.Close
            '
            Set cmdSQL = New ADODB.Command
            cmdSQL.ActiveConnection = CurrentProject.Connection
            cmdSQL.CommandType = adCmdText

            cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
            cmdSQL.Execute tRecDeleted

            'MsgBox tStrSQL

            cmdSQL.CommandText = tStrSQL
            cmdSQL.Execute tRecDeleted
            '
            Set tRstEntryCodes = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
            '

```

```
tRstEntryCodes.MoveFirst
Set frmZZZ_ENTRY_CODE.Form.LB_Entry.Recordset = tRstEntryCodes
tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
frmZZZ_ENTRY_CODE.Form.LB_Entry.Requery
'
Set tRstDummy = Nothing
'
' set the type values
'
Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
'
tRstEntryTypes.MoveFirst
tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
TxtTypeID.Value = cNode.Key
TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
tRstEntryTypes.Close
Set tRstEntryTypes = Nothing
End If
End If
End If
End If
End Sub
```

```

Option Compare Database
Public gRstEntryCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String
'#####Treeview Code#####
'Add this to your form's declaration section
Public WithEvents mcTree As clsTreeview
Private mbExit As Boolean ' to exit a SpinButton event
'#####Treeview Code#####

Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click

    Clear_SelectAll
    DoCmd.Close

Exit_CmdCancel_Click:
    Exit Sub

Err_CmdCancel_Click:
    MsgBox Err.Description
    Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdFindNext_Click()
    Dim tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset, cNode As clsNode
    Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

    TxtEntryDesc.Value = ""
    TxtEntryChn.Value = ""

    If Not IsNull(gRstEntryCodes) Then
        If Not (gStrSearch = "") Then

            'MsgBox "Looking for entry"

            If Not gUseAlt Then
                gRstEntryCode.FindNext gStrSearch

                If gRstEntryCode.NoMatch Then
                    gRstEntryCode.FindFirst gStrSearchAlt
                    gUseAlt = True
                End If
            Else
                gRstEntryCode.FindNext gStrSearchAlt
            End If

            If gRstEntryCode.NoMatch Then
                If gUseAlt Then
                    gUseAlt = False
                End If
                CmdFindNext.Enabled = False
            Else
                '
                ' next find the entry_type
                '
                'MsgBox "Looking for entry type"

                Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)

                tRstEntryTypes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)

                If Not tRstEntryTypes.NoMatch Then
                    '
                    ' update the code value
                    '
                    TxtEntryCode.Value = gRstEntryCode!c_entry_code
                    If Not IsNull(gRstEntryCode!c_entry_desc) Then
                        TxtEntryDesc.Value = gRstEntryCode!c_entry_desc
                    End If
                    If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then
                        TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn
                    End If
                    '
                    ' define the string
                    '
                    'MsgBox "Looking for node"
                    '
                    tStr = Trim(tRstEntryTypes!c_entry_type)
                    '

```

```

' search the tree
Set cNode = mcTree.Nodes(tStr)
'
If Not IsNull(cNode) Then
    If cNode.Key = gNode.Key Then
        Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
        tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
        frmZZZ_ENTRY_CODE.Form.Refresh
    Else
        '
        'MsgBox "found node"
        '
        Set mcTree.ActiveNode = cNode
        'tNode.Selected = True
        '
        ' then one makes it visible
        '
        'tNode.EnsureVisible
        '
        Set gNode = cNode
        '
        ' Finally populate the options and select the record.
        '
        CmdSelectAll.Enabled = True
        '
        tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) "
+ _
        "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
        "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
+ _
        "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = "
+ _
        "ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
        "WHERE (((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

        Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
        Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
        Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstDummy
        tRstEntryCodes.Close
        '
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText

        cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
        cmdSQL.Execute tRecDeleted

        'MsgBox tStrSQL

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
        Set tRstEntryCodes = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
        '
        tRstEntryCodes.MoveFirst
        Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCodes
        tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
        frmZZZ_ENTRY_CODE.Form.Refresh
        '
        Set tRstDummy = Nothing
        '
        ' set the type values
        '
        Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
        '
        tRstEntryTypes.MoveFirst
        tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key
        TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
        TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
        tRstEntryTypes.Close
        Set tRstEntryTypes = Nothing
    End If
End If
End If
End If
End If
End Sub

Private Sub CmdSelect_Click()

```


frmPickEntry2.archive -3

```
Clear_SelectAll
CmdSelectAll.SetFocus
CmdSelect.Enabled = False
'
' when someone clicks on an entry record, the following data is put into the form:
' TxtEntryCode.Value = tRst!c_entry_code
' TxtEntryDesc.Value = tRst!c_entry_desc
' TxtEntryChn.Value = tRst!c_entry_desc_chn
' TxtTypeID.Value = ""
' TxtTypeDesc.Value = ""
' TxtTypeChn.Value = ""

Forms!frmPickEntry2.Visible = False
End Sub

Private Sub CmdSelectAll_Click()
    If CmdSelectAll.Caption = "Select All" Then
        CmdSelectAll.Caption = "De-select All"
        frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(255, 255, 255)
        frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(0, 0, 0)
        CmdSelect.Enabled = True
        Me.TxtEntryCode.Value = -1
        Me.TxtEntryDesc.Value = ""
        Me.TxtEntryChn.Value = ""
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub

Private Sub Clear_SelectAll()
    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
    frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim tStrEntry As String
    Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset
    ' Courtesy Hans Vogelaar
    ' Populate the treeview with data from the tables
    Dim cRoot As clsNode
    ' Three levels of nodes
    Dim cNode1 As clsNode
    Dim cNode2 As clsNode
    Dim cNode3 As clsNode
    ' Key and caption for the nodes
    Dim strKey As String
    Dim strCaption As String
    '
    ' initialize the type values
    '
    TxtTypeID.Value = ""
    TxtTypeDesc.Value = "All"
    TxtTypeChn.Value = "All"
    '
    ' initialize the Entry Codes dataset
    '
    Set gRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)
    Set frmZZZ_ENTRY_CODE.Form.Recordset = gRstEntryCode
    '
    Set tRstEntry = frmZZZ_ENTRY_CODE.Form.Recordset
    '
    If Not IsNull(Me.OpenArgs) Then
        tStrEntry = Me.OpenArgs
        tRstEntry.FindFirst "c_entry_code = " & tStrEntry
    End If
    '
    ' set the language
    Dim tmli As MsoLanguageID
    '
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    '
    If tmli = msoLanguageIDSimplifiedChinese Then
        gDisplayLanguage = "S"
    ElseIf tmli = msoLanguageIDTraditionalChinese Then
        gDisplayLanguage = "T"
    Else
        gDisplayLanguage = "E"
    End If
End Sub
```

```

End If
'
' build treeview
'
Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
tRst.MoveFirst
Set mcTree = Me.subTreeView.Form.pTreeview
With mcTree
    .NodesClear
    .AppName = AppName ' Title for message boxes:
    ' Add a Root node with main and expanded icons and make it bold
    ' use the appropriate caption
    If gDisplayLanguage = "T" Then
        strCaption = ChrW(20837) + ChrW(20181) + ChrW(36884) + ChrW(24465) + ChrW(20998) + ChrW(39006)
    ElseIf gdisplaylanguage = "S" Then
        strCaption = ChrW(20837) + ChrW(20181) + ChrW(36884) + ChrW(24452) + ChrW(20998) + ChrW(31612)
    Else
        strCaption = "Categories of Modes of Entry"
    End If

    Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
    cRoot.Bold = True
    ' Loop through the records
    Do While Not tRst.EOF
        ' Add node
        strKey = tRst!c_entry_type
        If gDisplayLanguage = "E" Then
            strCaption = tRst!c_entry_type_desc
        Else
            strCaption = tRst!c_entry_type_desc_chn
        End If
        If Len(tRst!c_entry_type) = 2 Then
            Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode1.Expanded = False
        Else
            Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode2.Expanded = False
        End If
        ' Move to next date
        tRst.MoveNext
    Loop
    ' Create the node controls and display the tree
    .Refresh
End With
Set tRst = Nothing
'
frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
tRstEntry.MoveFirst
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

    Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click

End Sub

Private Sub mcTree_Click(cNode As clsNode)
Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset
Dim tRstEntryCode As DAO.Recordset, tRstDummy As DAO.Recordset
Dim tStrSQL As String

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

Me.TxtEntryCode.Value = -1
Me.TxtEntryDesc.Value = ""
Me.TxtEntryChn.Value = ""
CmdSelect.Enabled = False

' reset the form colors
'
Clear_SelectAll

```

```

'
If cNode.Key = "Root" Then
    TxtTypeID.Value = ""
    TxtTypeDesc.Value = ""
    TxtTypeChn.Value = ""
    CmdSelectAll.Enabled = False
'
' reset the entry code choices
'
Set tRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)
'

Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCode
frmZZZ_ENTRY_CODE.Form.Refresh
Else
    Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
'

tRst.MoveFirst
tRst.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
TxtTypeID.Value = cNode.Key
TxtTypeDesc.Value = tRst!c_entry_type_desc
TxtTypeChn.Value = tRst!c_entry_type_desc_chn
tRst.Close
Set tRst = Nothing
'

CmdSelectAll.Enabled = True
'

' we need to distinguish between type / subtype
'

tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
"SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
"ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
"FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
"ENTRY_CODE_TYPE_REL.c_ENTRY_code "

If Len(cNode.Key) = 2 Then
    tStrSQL = tStrSQL + "WHERE (((Left((ENTRY_CODE_TYPE_REL.c_ENTRY_type),2))= '" + _
    TxtTypeID.Value + "'))"
Else
    tStrSQL = tStrSQL + "WHERE (ENTRY_CODE_TYPE_REL.c_ENTRY_type = '" + _
    TxtTypeID.Value + "')"
End If

'

Set tRstEntryCode = frmZZZ_ENTRY_CODE.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstDummy
tRstEntryCode.Close
'

cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted
'

Set tRstEntryCode = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
'

Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCode
Set tRstDummy = Nothing
'

End If
End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Then
        If TxtSearchChn.Value = "" Then
            CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Then
        If TxtSearch.Value = "" Then
            Me.CmdFind.Enabled = False
        End If
    Else

```

```

        TxtSearch.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub
Private Sub NodeSearch()

    Dim cNode As clsNode, tStr As String, tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrLen As String, cmdSQL As ADODB.Command

    ' all specific association codes will have a type of the form 0101
    ' hence the ValuePath for the relevant node will be "K000/K01/K0101"
    ' The command to locate the relevant node is:
    ' cNode = mcTree.FindNode(tStrValuePath)
    ' mcTree.activeNode = cNode
    '
    TxtEntryDesc.Value = ""
    TxtEntryChn.Value = ""
    '
    ' search for the search string in ASSOC_CODES
    TxtSearchChn.SetFocus
    tStrSearchChn = Trim(Me.TxtSearchChn.Text)
    TxtSearch.SetFocus
    tStrSearchEng = Trim(Me.TxtSearch.Text)
    CmdFind.SetFocus
    '
    ' because the user may have a hard time picking the exact term, I'll treat it as
    ' (1) the beginning of the actual term
    ' (2) part of the term
    '
    gUseAlt = False
    gStrSearch = ""
    If tStrSearchChn <> "" Then
        tStrLen = Str(LenB(tStrSearchChn))
        gStrSearch = "LeftB(c_entry_desc_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
        gStrSearchAlt = "InStrB(1,c_entry_desc_chn,'" + tStrSearchChn + "') > 0"
        'tStrSearch = "c_entry_desc_chn = '" + tStrSearchChn + "'"
    ElseIf tStrSearchEng <> "" Then
        tStrLen = Str(Len(tStrSearchEng))
        gStrSearch = "Left(c_entry_desc," + tStrLen + ") = '" + tStrSearchEng + "'"
        gStrSearchAlt = "InStr(1,c_entry_desc,'" + tStrSearchEng + "') > 0"
        'tStrSearch = "c_entry_desc = '" + tStrSearchEng + "'"
    End If

    If Not (gStrSearch = "") Then

        'MsgBox "Looking for entry"

        Set gRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)

        gRstEntryCode.FindFirst gStrSearch

        If gRstEntryCode.NoMatch Then
            gRstEntryCode.FindFirst gStrSearchAlt
            gUseAlt = True
        End If

        If gRstEntryCode.NoMatch Then
            If gUseAlt Then
                gUseAlt = False
            End If
            CmdFindNext.Enabled = False
        Else
            CmdFindNext.Enabled = True
            '
            ' next find the entry_type
            '
            'MsgBox "Looking for entry type"

            Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)

            tRstEntryTypes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)

            If Not tRstEntryTypes.NoMatch Then
                '
                ' set the code values
                '
                TxtEntryCode.Value = gRstEntryCode!c_entry_code
                If Not IsNull(gRstEntryCode!c_entry_desc) Then

```

```

        TxtEntryDesc.Value = gRstEntryCode!c_entry_desc
    End If
    If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then
        TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn
    End If
    '
    ' define the string
    '
    'MsgBox "Looking for node"
    '
    tStr = Trim(tRstEntryTypes!c_entry_type)
    '
    ' search the tree

Set cNode = mcTree.Nodes(tStr)
'
If Not IsNull(cNode) Then
    '
    'MsgBox "found node"
    '
    Set mcTree.ActiveNode = cNode
    'cNode.Selected = True
    '
    ' then one makes it visible
    '
    'cNode.EnsureVisible = True
    '
    Set gNode = cNode
    '
    ' Finally populate the options and select the record.
    '
    CmdSelectAll.Enabled = True
    '
    tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
        "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
        "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
        "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
        "ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
        "WHERE ((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

    Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
    Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstDummy
    tRstEntryCodes.Close
    '
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
    cmdSQL.Execute tRecDeleted

    'MsgBox tStrSQL

    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    '
    Set tRstEntryCodes = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
    '
    tRstEntryCodes.MoveFirst
    Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCodes
    tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
    frmZZZ_ENTRY_CODE.Form.Refresh
    '
    Set tRstDummy = Nothing
    '
    ' set the type values
    '
    Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_Types", dbOpenDynaset)
    '
    tRstEntryTypes.MoveFirst
    tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + cNode.Key + Chr(34)
    TxtTypeID.Value = cNode.Key
    TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
    TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
    tRstEntryTypes.Close
    Set tRstEntryTypes = Nothing
End If
End If
End If

```

Form_frmPickEntry2_archive -8

End If

End Sub

frmPickEntry2_old - 1

Option Compare Database

Public gRstEntryCode As DAO.Recordset, gNode As Node, gStrSearch As String, gStrSearchAlt As String

Public gUseAlt As Boolean

Private Sub CmdCancel_Click()

On Error GoTo Err_CmdCancel_Click

Clear_SelectAll

DoCmd.Close

Exit_CmdCancel_Click:

Exit Sub

Err_CmdCancel_Click:

MsgBox Err.Description

Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdFindNext_Click()

Dim tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset, tNode As Node

Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

TxtEntryDesc.Value = ""

TxtEntryChn.Value = ""

If Not IsNull(gRstEntryCodes) Then

If Not (gStrSearch = "") Then

'MsgBox "Looking for entry"

If Not gUseAlt Then

gRstEntryCode.FindNext gStrSearch

If gRstEntryCode.NoMatch Then

gRstEntryCode.FindFirst gStrSearchAlt

gUseAlt = True

End If

Else

gRstEntryCode.FindNext gStrSearchAlt

End If

If gRstEntryCode.NoMatch Then

If gUseAlt Then

gUseAlt = False

End If

CmdFindNext.Enabled = False

Else

,

' next find the entry_type

,

'MsgBox "Looking for entry type"

Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)

tRstEntryTypes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)

If Not tRstEntryTypes.NoMatch Then

,

' update the code value

,

TxtEntryCode.Value = gRstEntryCode!c_entry_code

If Not IsNull(gRstEntryCode!c_entry_desc) Then

TxtEntryDesc.Value = gRstEntryCode!c_entry_desc

End If

If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then

TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn

End If

,

' define the string

,

'MsgBox "Looking for node"

,

tStr = Trim(tRstEntryTypes!c_entry_type)

,

' search the tree

Set tNode = TreeViewTypes.Nodes("K" + tStr)

,

If Not IsNull(tNode) Then

```

If tNode = gNode Then
    Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
    tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
    frmZZZ_ENTRY_CODE.Form.Refresh
Else
    ,
    'MsgBox "found node"
    ,
    tNode.Selected = True
    ,
    ' then one makes it visible
    ,
    tNode.EnsureVisible
    ,
    Set gNode = tNode
    ,
    ' Finally populate the options and select the record.
    ,
    CmdSelectAll.Enabled = True
    ,
    tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) "
+ _
        "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
        "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
        "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = "
+ _
        "ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
        "WHERE (((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

    Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
    Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstDummy
    tRstEntryCodes.Close
    ,
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
    cmdSQL.Execute tRecDeleted

    'MsgBox tStrSQL

    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    ,
    Set tRstEntryCodes = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
    ,
    tRstEntryCodes.MoveFirst
    Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCodes
    tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
    frmZZZ_ENTRY_CODE.Form.Refresh
    ,
    Set tRstDummy = Nothing
    ,
    ' set the type values
    ,
    Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
    ,
    tRstEntryTypes.MoveFirst
    tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + tNode.Tag + Chr(34)
    TxtTypeID.Value = tNode.Tag
    TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
    TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
    tRstEntryTypes.Close
    Set tRstEntryTypes = Nothing
End If
End If
End If
End If
End If
End Sub

```

```

Private Sub CmdSelect_Click()
    Clear_SelectAll
    CmdSelectAll.SetFocus
    CmdSelect.Enabled = False
    Forms!frmPickEntry2.Visible = False
End Sub

```



```

Private Sub CmdSelectAll_Click()
    If CmdSelectAll.Caption = "Select All" Then
        CmdSelectAll.Caption = "De-select All"
        frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(255, 255, 255)
        frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(0, 0, 255)
        CmdSelect.Enabled = True
        Me.TxtEntryCode.Value = -1
        Me.TxtEntryDesc.Value = ""
        Me.TxtEntryChn.Value = ""
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub

Private Sub Clear_SelectAll()
    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
    frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim tNode As Node, tStrNode As String, tStrParent As String, tStrEntry As String
    Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset
    '
    ' initialize the Entry Codes dataset
    '
    Set gRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)
    Set frmZZZ_ENTRY_CODE.Form.Recordset = gRstEntryCode
    '
    Set tRstEntry = frmZZZ_ENTRY_CODE.Form.Recordset
    '
    If Not IsNull(Me.OpenArgs) Then
        tStrEntry = Me.OpenArgs
        tRstEntry.FindFirst "c_entry_code = " & tStrEntry
    End If
    '
    ' build treeview
    '
    Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
    TreeViewTypes.Nodes.Clear
    Set tNode = TreeViewTypes.Nodes.Add(, , "K000", "All Entry Types")
    tNode.Expanded = True
    tNode.Tag = "000"
    '
    ' the general syntax for adding nodes is:
    '     TreeViewTypes.Nodes.Add("K...", tvwChild, parent node, "Text")
    '
    tRst.MoveFirst
    Do While Not tRst.EOF
        If tRst!c_entry_type_parent_id = "0" Then
            tStrParent = "K000"
        Else
            tStrParent = "K" + Trim(tRst!c_entry_type_parent_id)
        End If
        tStrNode = "K" + Trim(tRst!c_entry_type)
        Set tNode = TreeViewTypes.Nodes.Add(tStrParent, tvwChild, tStrNode, tRst!c_entry_type_desc)
        tNode.Tag = tRst!c_entry_type
        tRst.MoveNext
    Loop
    '
    Set tRst = Nothing

    frmZZZ_ENTRY_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
    frmZZZ_ENTRY_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
    tRstEntry.MoveFirst
End Sub

Private Sub CmdFind_Click()
    On Error GoTo Err_CmdFind_Click

    Call NodeSearch

Exit_CmdFind_Click:
    Exit Sub

Err_CmdFind_Click:
    MsgBox Err.Description
    Resume Exit_CmdFind_Click

```

End Sub

```

Private Sub TreeViewTypes_NodeClick(ByVal Node As Object)
    Dim tRst As DAO.Recordset, tRstEntry As DAO.Recordset
    Dim tRstEntryCode As DAO.Recordset, tRstDummy As DAO.Recordset
    Dim tStrSQL As String

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    Me.TxtEntryCode.Value = -1
    Me.TxtEntryDesc.Value = ""
    Me.TxtEntryChn.Value = ""
    CmdSelect.Enabled = False

    ' reset the form colors
    '
    Clear_SelectAll
    '
    If Node.Tag = "000" Then
        TxtTypeID.Value = ""
        TxtTypeDesc.Value = ""
        TxtTypeChn.Value = ""
        CmdSelectAll.Enabled = False
        '
        ' reset the entry code choices
        '
        Set tRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)
        '
        Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCode
        frmZZZ_ENTRY_CODE.Form.Refresh
    Else
        Set tRst = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
        '
        tRst.MoveFirst
        tRst.FindFirst "c_entry_type = " + Chr(34) + Node.Tag + Chr(34)
        TxtTypeID.Value = Node.Tag
        TxtTypeDesc.Value = tRst!c_entry_type_desc
        TxtTypeChn.Value = tRst!c_entry_type_desc_chn
        tRst.Close
        Set tRst = Nothing
        '
        CmdSelectAll.Enabled = True
        '
        ' we need to distinguish between type / subtype
        '
        tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
            "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
            "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
            "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
            "ENTRY_CODE_TYPE_REL.c_ENTRY_code "

        If Len(Node.Tag) = 2 Then
            tStrSQL = tStrSQL + "WHERE (((Left((ENTRY_CODE_TYPE_REL.c_ENTRY_type),2))= '" + _
                TxtTypeID.Value + "')))"
        Else
            tStrSQL = tStrSQL + "WHERE (ENTRY_CODE_TYPE_REL.c_ENTRY_type = '" + _
                TxtTypeID.Value + "')"
        End If

        '
        Set tRstEntryCode = frmZZZ_ENTRY_CODE.Form.Recordset
        Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
        Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstDummy
        tRstEntryCode.Close
        '
        cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
        cmdSQL.Execute tRecDeleted

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
        Set tRstEntryCode = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)
        '
        Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCode
        Set tRstDummy = Nothing
        '
    
```

frmPickEntry2_old - 5

```
End If
End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Then
        If TxtSearchChn.Value = "" Then
            CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Then
        If TxtSearch.Value = "" Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearch.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub NodeSearch()

    Dim tNode As Node, tStr As String, tRstEntryCodes As DAO.Recordset, tRstEntryTypes As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrLen As String, cmdSQL As ADODB.Command

    ' all specific association codes will have a type of the form 0101
    ' hence the ValuePath for the relevant node will be "K000/K01/K0101"
    ' The command to locate the relevant node is:
    ' tNode = TreeViewType.FindNode(tStrValuePath)
    ' TreeViewType.SelectedNode = tNode
    '
    TxtEntryDesc.Value = ""
    TxtEntryChn.Value = ""
    '
    ' search for the search string in ASSOC_CODES
    TxtSearchChn.SetFocus
    tStrSearchChn = Trim(Me.TxtSearchChn.Text)
    TxtSearch.SetFocus
    tStrSearchEng = Trim(Me.TxtSearch.Text)
    CmdFind.SetFocus
    '
    ' because the user may have a hard time picking the exact term, I'll treat it as
    ' (1) the beginning of the actual term
    ' (2) part of the term
    '
    gUseAlt = False
    gStrSearch = ""
    If tStrSearchChn <> "" Then
        tStrLen = Str(LenB(tStrSearchChn))
        gStrSearch = "LeftB(c_entry_desc_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
        gStrSearchAlt = "InStrB(1,c_entry_desc_chn,'" + tStrSearchChn + "') > 0"
        'tStrSearch = "c_entry_desc_chn = '" + tStrSearchChn + "'"
    ElseIf tStrSearchEng <> "" Then
        tStrLen = Str(Len(tStrSearchEng))
        gStrSearch = "Left(c_entry_desc," + tStrLen + ") = '" + tStrSearchEng + "'"
        gStrSearchAlt = "InStr(1,c_entry_desc,'" + tStrSearchEng + "') > 0"
        'tStrSearch = "c_entry_desc = '" + tStrSearchEng + "'"
    End If

    If Not (gStrSearch = "") Then

        'MsgBox "Looking for entry"

        Set gRstEntryCode = CurrentDb.OpenRecordset("ENTRY_CODES", dbOpenDynaset)

        gRstEntryCode.FindFirst gStrSearch

        If gRstEntryCode.NoMatch Then
            gRstEntryCode.FindFirst gStrSearchAlt
            gUseAlt = True
        End If
    End If
End Sub
```

```

If gRstEntryCode.NoMatch Then
    If gUseAlt Then
        gUseAlt = False
    End If
    CmdFindNext.Enabled = False
Else
    CmdFindNext.Enabled = True
    '
    ' next find the entry_type
    '
    'MsgBox "Looking for entry type"

    Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_CODE_TYPE_REL", dbOpenDynaset)

    tRstEntryTypes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)

    If Not tRstEntryTypes.NoMatch Then
        '
        ' set the code values
        '
        TxtEntryCode.Value = gRstEntryCode!c_entry_code
        If Not IsNull(gRstEntryCode!c_entry_desc) Then
            TxtEntryDesc.Value = gRstEntryCode!c_entry_desc
        End If
        If Not IsNull(gRstEntryCode!c_entry_desc_chn) Then
            TxtEntryChn.Value = gRstEntryCode!c_entry_desc_chn
        End If
        '
        ' define the string
        '
        'MsgBox "Looking for node"
        '
        tStr = Trim(tRstEntryTypes!c_entry_type)
        '
        ' search the tree

        Set tNode = TreeViewTypes.Nodes("K" + tStr)
        '
        If Not IsNull(tNode) Then
            'MsgBox "found node"
            '
            tNode.Selected = True
            '
            ' then one makes it visible
            '
            tNode.EnsureVisible
            '
            Set gNode = tNode
            '
            ' Finally populate the options and select the record.
            '
            CmdSelectAll.Enabled = True
            '
            tStrSQL = "INSERT INTO ZZ_ENTRY_CODE (c_ENTRY_code, c_ENTRY_desc, c_ENTRY_desc_chn) " + _
                "SELECT ENTRY_CODE_TYPE_REL.c_ENTRY_code AS c_ENTRY_code, " + _
                "ENTRY_CODES.c_ENTRY_desc, ENTRY_CODES.c_ENTRY_desc_chn " + _
                "FROM ENTRY_CODES INNER JOIN ENTRY_CODE_TYPE_REL ON ENTRY_CODES.c_ENTRY_code = " + _
                "ENTRY_CODE_TYPE_REL.c_ENTRY_code " + _
                "WHERE ((ENTRY_CODE_TYPE_REL.c_entry_type)='" + tStr + "'))"

            Set tRstEntryCodes = frmZZZ_ENTRY_CODE.Form.Recordset
            Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
            Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstDummy
            tRstEntryCodes.Close
            '
            Set cmdSQL = New ADODB.Command
            cmdSQL.ActiveConnection = CurrentProject.Connection
            cmdSQL.CommandType = adCmdText

            cmdSQL.CommandText = "Delete * from ZZ_ENTRY_CODE"
            cmdSQL.Execute tRecDeleted

            'MsgBox tStrSQL

            cmdSQL.CommandText = tStrSQL
            cmdSQL.Execute tRecDeleted
            '
            Set tRstEntryCodes = CurrentDb.OpenRecordset("ZZ_ENTRY_CODE", dbOpenDynaset)

```

```
    '
    tRstEntryCodes.MoveFirst
    Set frmZZZ_ENTRY_CODE.Form.Recordset = tRstEntryCodes
    tRstEntryCodes.FindNext "c_entry_code = " + Str(gRstEntryCode!c_entry_code)
    frmZZZ_ENTRY_CODE.Form.Refresh
    '
    Set tRstDummy = Nothing
    '
    ' set the type values
    '
    Set tRstEntryTypes = CurrentDb.OpenRecordset("ENTRY_TYPES", dbOpenDynaset)
    '
    tRstEntryTypes.MoveFirst
    tRstEntryTypes.FindFirst "c_entry_type = " + Chr(34) + tNode.Tag + Chr(34)
    TxtTypeID.Value = tNode.Tag
    TxtTypeDesc.Value = tRstEntryTypes!c_entry_type_desc
    TxtTypeChn.Value = tRstEntryTypes!c_entry_type_desc_chn
    tRstEntryTypes.Close
    Set tRstEntryTypes = Nothing
End If
End If
End If
End Sub
```

frm_frmPickETHNICITY - 1

Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
Forms!frmPickETHNICITY.Visible = False  
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
    If Not IsNull(Me.OpenArgs) Then  
        Dim strETHN As String  
        strETHN = Me.OpenArgs  
        Dim rsEthn As DAO.Recordset  
        Set rsEthn = frmETHNICITY.Form.Recordset  
        rsEthn.FindFirst "c_ethnicity_code = " & strETHN  
    End If  
End Sub
```

```
Private Sub CmdFind_Click()  
On Error GoTo Err_CmdFind_Click
```

```
    Dim StrSearch As String  
    Me.TxtSearch.SetFocus  
    StrSearch = Me.TxtSearch.Value  
    If StrSearch <> "" Then  
        Dim rsEthn As DAO.Recordset  
        Set rsEthn = frmETHNICITY.Form.Recordset  
        Dim StrSearchStr As String  
        StrSearchStr = "c_name_chn = " + Chr(34) + StrSearch + Chr(34)  
        rsEthn.FindFirst StrSearchStr  
    End If
```

```
Exit_CmdFind_Click:  
Exit Sub
```

```
Err_CmdFind_Click:  
MsgBox Err.Description  
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()  
    If Me.TxtSearch.Text = "" Then  
        Me.CmdFind.Enabled = False  
    Else  
        Me.CmdFind.Enabled = True  
    End If  
End Sub
```

Form_frmPickEVENT - 1

Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
    MsgBox Err.Description  
    Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
    Forms!frmPickeVENT.Visible = False  
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)  
    If Not IsNull(Me.OpenArgs) Then  
        Dim strEVENT As String  
        strEVENT = Me.OpenArgs  
        Dim rsEvent As DAO.Recordset  
        Set rsEvent = frmNIAN_HAO.Form.Recordset  
        rsEvent.FindFirst "c_event_code = " & strEVENT  
    End If
```

```
End Sub
```

frm_frmPickInst - 1

Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
    MsgBox Err.Description  
    Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
    Forms!frmPickInst.Visible = False  
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
    frmINST_CODES.Form.OrderBy = "c_inst_name_py"  
    frmINST_CODES.Form.OrderByOn = True
```

```
        If Not IsNull(Me.OpenArgs) Then  
            Dim strINST As String  
            strINST = Me.OpenArgs  
            Dim rsInst As DAO.Recordset  
            Set rsInst = frmINST_CODES.Form.Recordset  
            rsInst.FindFirst "c_inst_code = " & strINST  
        End If
```

```
End Sub
```

```
Private Sub TxtSearch_Change()  
    If Me.TxtSearch.Text = "" Then  
        Me.CmdFind.Enabled = False  
    Else  
        Me.CmdFind.Enabled = True  
    End If  
End Sub
```

```
Private Sub CmdFind_Click()  
On Error GoTo Err_CmdFind_Click
```

```
    Dim StrSearch As String  
    Me.TxtSearch.SetFocus  
    StrSearch = Me.TxtSearch.Value  
    If StrSearch <> "" Then  
        Dim rsInst As DAO.Recordset  
        Set rsInst = frmINST_CODES.Form.Recordset  
        Dim StrSearchStr As String  
        StrSearchStr = "c_inst_name_hz = " + Chr(34) + StrSearch + Chr(34)  
        rsInst.FindFirst StrSearchStr  
    End If
```

```
Exit_CmdFind_Click:  
Exit Sub
```

```
Err_CmdFind_Click:  
    MsgBox Err.Description  
    Resume Exit_CmdFind_Click
```

```
End Sub
```


Option Compare Database

```
Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
Exit Sub
```

```
Err_CmdCancel_Click:
MsgBox Err.Description
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()
Forms!frmPickKINSHIP_CODES.Visible = False
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
    If Not IsNull(Me.OpenArgs) Then
        Dim strKIN As String
        strKIN = Me.OpenArgs
        Dim rsKin As DAO.Recordset
        Set rsKin = frmKINSHIP_CODES.Form.Recordset
        rsKin.FindFirst "c_kincode = " & strKIN
    End If
```

```
End Sub
```

```
Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click
```

```
    Dim StrSearch As String
    Me.TxtSearch.SetFocus
    StrSearch = Me.TxtSearch.Value
    If StrSearch <> "" Then
        Dim rsKinCodes As DAO.Recordset
        Set rsKinCodes = frmKINSHIP_CODES.Form.Recordset
        Dim StrSearchStr As String
        StrSearchStr = "c_kinrel_chn = " + Chr(34) + StrSearch + Chr(34)
        rsKinCodes.FindFirst StrSearchStr
    End If
```

```
Exit_CmdFind_Click:
Exit Sub
```

```
Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()
    If Me.TxtSearch.Text = "" Then
        Me.CmdFind.Enabled = False
    Else
        Me.CmdFind.Enabled = True
    End If
End Sub
```

Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
Forms!frmPickNIAN_HAO.Visible = False  
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
    If Not IsNull(Me.OpenArgs) Then  
        Dim strNian As String  
        strNian = Me.OpenArgs  
        Dim rsNian As DAO.Recordset  
        Set rsNian = frmNIAN_HAO.Form.Recordset  
        rsNian.FindFirst "c_nianhao_id = " & strNian  
    End If
```

```
End Sub
```

```
Private Sub CmdFind_Click()  
On Error GoTo Err_CmdFind_Click
```

```
    Dim StrSearch As String  
    Me.TxtSearch.SetFocus  
    StrSearch = Me.TxtSearch.Value  
    If StrSearch <> "" Then  
        Dim rsNH As DAO.Recordset  
        Set rsNH = frmNIAN_HAO.Form.Recordset  
        Dim StrSearchStr As String  
        StrSearchStr = "c_nianhao_chn = " + Chr(34) + StrSearch + Chr(34)  
        rsNH.FindFirst StrSearchStr  
    End If
```

```
Exit_CmdFind_Click:  
Exit Sub
```

```
Err_CmdFind_Click:  
MsgBox Err.Description  
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()  
    If Me.TxtSearch.Text = "" Then  
        Me.CmdFind.Enabled = False  
    Else  
        Me.CmdFind.Enabled = True  
    End If  
End Sub
```

Form_frmPickOfficeTree - 1

Option Compare Database

Public gRstOfficeCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String

Public gUseAlt As Boolean, gDisplayLanguage As String

'#####Treeview Code#####

'Add this to your form's declaration section

Public WithEvents mcTree As clsTreeview

Private mbExit As Boolean ' to exit a SpinButton event

'#####Treeview Code#####

Private Sub CmdCancel_Click()

On Error GoTo Err_CmdCancel_Click

Clear_SelectAll

DoCmd.Close

Exit_CmdCancel_Click:

Exit Sub

Err_CmdCancel_Click:

MsgBox Err.Description

Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdFindNext_Click()

Dim tRstAssocCodes As DAO.Recordset, tRstAssocTypes As DAO.Recordset, cNode As clsNode

Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

TxtOfficeCode.Value = -1

TxtOfficeDesc.Value = ""

TxtOfficeDescChn.Value = ""

,

' clear the scratch table

,

Set cmdSQL = New ADODB.Command

cmdSQL.ActiveConnection = CurrentProject.Connection

cmdSQL.CommandType = adCmdText

,

If IsNull(gRstOfficeCode) Then

MsgBox "Error in search: table closed."

Else

If gStrSearch = "" Then

MsgBox "Error in search: string empty."

Else

MsgBox "Looking for entry"

If gRstOfficeCode.EOF Then

CmdFindNext.Enabled = False

Else

If gUseAlt Then

gRstOfficeCode.FindNext gStrSearchAlt

Else

gRstOfficeCode.FindNext gStrSearch

If gRstOfficeCode.NoMatch Then

gRstOfficeCode.FindFirst gStrSearchAlt

gUseAlt = True

End If

End If

If gRstOfficeCode.NoMatch Then

If gUseAlt Then

gUseAlt = False

End If

CmdFindNext.Enabled = False

Else

,

' next find the entry_type

,

MsgBox "Looking for entry type"

Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_CODE_TYPE_REL", dbOpenDynaset)

tRstOfficeTreeIDs.FindNext "c_office_id = " + Str(gRstOfficeCode!c_office_id)

If Not tRstOfficeTreeIDs.NoMatch Then

,

```

' set the code values
TxtOfficeCode.Value = gRstOfficeCode!c_office_id
If Not IsNull(gRstOfficeCode!c_office_trans) Then
    TxtOfficeDesc.Value = gRstOfficeCode!c_office_trans
End If
If Not IsNull(gRstOfficeCode!c_office_chn) Then
    TxtOfficeDescChn.Value = gRstOfficeCode!c_office_chn
End If
'
' define the string
'
'MsgBox "Looking for node"
'
tStr = Trim(trstOfficeTreeIDs!c_office_tree_id)
'
' search the tree

Set cNode = mcTree.Nodes(tStr)
'
If Not IsNull(cNode) Then
    If cNode.Key = gNode.Key Then
        Set trstOfficeCode = frmZZZ_OFFICE_CODE.Form.Recordset
        trstOfficeCode.FindFirst "c_office_id = " + Str(gRstOfficeCode!c_office_id)
        frmZZZ_OFFICE_CODE.Form.Refresh
    Else
        Set mcTree.ActiveNode = cNode
        'cNode.Selected = True
        '
        ' then one makes it visible
        '
        'tNode.EnsureVisible
        '
        Set gNode = cNode
        '
        ' Finally populate the options and select the record.
        '
        CmdSelectAll.Enabled = True
        '
        tStrSQL = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn
) " + _
            "SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c
_office_chn " + _
            "FROM OFFICE_CODES INNER JOIN OFFICE_CODE_TYPE_REL ON " + _
            "OFFICE_CODES.c_office_id = OFFICE_CODE_TYPE_REL.c_office_id " + _
            "WHERE (c_office_tree_id = '" + tStr + "')"

        Set trstOfficeCode = frmZZZ_OFFICE_CODE.Form.Recordset
        Set trstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_OC", dbOpenDynaset)
        Set frmZZZ_OFFICE_CODE.Form.Recordset = trstDummy
        trstOfficeCode.Close
        '
        cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
        cmdSQL.Execute tRecDeleted

        'MsgBox tStrSQL

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
        Set trstOfficeCode = CurrentDb.OpenRecordset("ZZ_OFFICE_CODE", dbOpenDynaset)
        '
        Set frmZZZ_OFFICE_CODE.Form.Recordset = trstOfficeCode
        trstOfficeCode.FindFirst "c_office_id = " + Str(gRstOfficeCode!c_office_id)
        frmZZZ_OFFICE_CODE.Form.Refresh
        '
        Set trstDummy = Nothing
        '
        ' briefly set the focus on the tree to get the highlighted node
        '
        subTreeView.SetFocus
        frmZZZ_OFFICE_CODE.SetFocus
        '
        ' set the tree values
        '
        Set trstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
        '
        trstOfficeTreeIDs.MoveFirst
        trstOfficeTreeIDs.FindFirst "c_office_type_node_id = " + Chr(34) + cNode.Key + Ch
r(34)

```

frmPickOfficeTree - 3

```
                'TxtTypeID.Value = Node.Tag
                TxtTypeDesc.Value = tRstOfficeTreeIDs!c_office_type_desc
                TxtTypeDescChn.Value = tRstOfficeTreeIDs!c_office_type_desc_chn
                tRstOfficeTreeIDs.Close
                Set tRstOfficeTreeIDs = Nothing
            End If
        End If
    End If
End If
End If
End If
End Sub
```

```
Private Sub CmdSelect_Click()
    Clear_SelectAll
    CmdSelectAll.SetFocus
    CmdSelect.Enabled = False
    If TxtOfficeCode.Value > -1 Then
        TxtOfficeCode.Value = frmZZZ_OFFICE_CODE.Form.Recordset!c_office_id
    End If
    ' MsgBox "Office Code: " + Str(TxtOfficeCode.Value)
    Forms!frmPickOfficeTree.Visible = False
End Sub
```

```
Private Sub CmdSelectAll_Click()
    If CmdSelectAll.Caption = "Select All" Then
        CmdSelectAll.Caption = "De-select All"
        frmZZZ_OFFICE_CODE.Form.DatasheetForeColor = RGB(255, 255, 255)
        frmZZZ_OFFICE_CODE.Form.DatasheetBackColor = RGB(0, 0, 255)
        CmdSelect.Enabled = True
        TxtOfficeCode.Value = -1
        TxtOfficeDesc.Value = ""
        TxtOfficeDescChn.Value = ""
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub
```

```
Private Sub Clear_SelectAll()
    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    frmZZZ_OFFICE_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
    frmZZZ_OFFICE_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
    CmdSelect.Enabled = False
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
    Dim cRoot As clsNode, cNode1 As clsNode, cNode2 As clsNode, cNode3 As clsNode
    Dim cNode4 As clsNode, cNode5 As clsNode, cNode6 As clsNode, cNode7 As clsNode
    Dim strKey As String, strCaption As String
    Dim tRst As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    '
    ' initialize the Assoc Codes dataset
    '
    Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)
    Set frmZZZ_OFFICE_CODE.Form.Recordset = gRstOfficeCode
    '
    'frmZZZ_OFFICE_CODE.Form.OrderBy = "c_sortorder"
    'frmZZZ_OFFICE_CODE.Form.OrderByOn = True

    If Not IsNull(Me.OpenArgs) Then
        Dim strOffice As String
        strOffice = Me.OpenArgs
        Dim rsOffice As DAO.Recordset
        Set rsOffice = frmZZZ_OFFICE_CODE.Form.Recordset
        rsOffice.FindFirst "c_office_id = " & strOffice
    End If

    ' set the language
    Dim tmli As MsoLanguageID
    '
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    ' gLabelsOK = True
```

```

If tmli = msoLanguageIDSimplifiedChinese Then
    gDisplayLanguage = "S"
ElseIf tmli = msoLanguageIDTraditionalChinese Then
    gDisplayLanguage = "T"
Else
    gDisplayLanguage = "E"
End If
'
' build treeview
'
Set tRst = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)

tRst.MoveFirst
Set mcTree = Me.subTreeView.Form.pTreeView
With mcTree
    .NodesClear
    .AppName = AppName ' Title for message boxes:
    ' Add a Root node with main and expanded icons and make it bold

    ' use the appropriate caption
    If gDisplayLanguage = "T" Then
        strCaption = ChrW(23448) + ChrW(32887) + ChrW(20998) + ChrW(39006)
    ElseIf gdisplaylangauge = "S" Then
        strCaption = ChrW(23448) + ChrW(32844) + ChrW(20998) + ChrW(31612)
    Else
        strCaption = "Administrative Category"
    End If

    Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
    cRoot.Bold = True
    ' Loop through the records
    Do While Not tRst.EOF
        ' Add node
        strKey = tRst!c_office_type_node_id
        If gDisplayLanguage = "E" Then
            strCaption = tRst!c_office_type_desc + " " + tRst!c_office_type_desc_chn
        Else
            strCaption = tRst!c_office_type_desc_chn
        End If
        '
        ' Note that since the root record has an ID of "0", its length is 1 and is automatically skipped
        '
        Select Case Len(strKey)
            Case 2
                Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode1.Expanded = False
            Case 4
                Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode2.Expanded = False
            Case 6
                Set cNode3 = cNode2.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode3.Expanded = False
            Case 8
                Set cNode4 = cNode3.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode4.Expanded = False
            Case 10
                Set cNode5 = cNode4.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode5.Expanded = False
            Case 12
                Set cNode6 = cNode5.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode6.Expanded = False
            Case 14
                Set cNode7 = cNode6.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode7.Expanded = False
        End Select
        tRst.MoveNext
    Loop
    ' Create the node controls and display the tree
    .Refresh
End With
'
TxtOfficeCode.Value = -1
TxtOfficeTypeType.Value = -1
TxtOfficeL1.Value = -1
TxtOfficeL2.Value = -1
TxtOfficeL3.Value = -1
TxtOfficeL4.Value = -1
TxtOfficeL5.Value = -1
TxtTypeDesc.Value = ""
TxtTypeDescChn.Value = ""

```

frm_frmPickOfficeTree - 5

```
TxtOfficeDesc.Value = ""
TxtOfficeDescChn.Value = ""

' clear the first pass of the file

cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
cmdSQL.Execute tRecDeleted

Set cmdSQL = Nothing
Set tRst = Nothing
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

'Dim StrSearch As String
'Me.TxtSearch.SetFocus
'StrSearch = Me.TxtSearch.Value
'If StrSearch <> "" Then
'    Dim rsOfficeCodes As DAO.Recordset
'    Set rsOfficeCodes = frmZZZ_OFFICE_CODE.Form.Recordset
'    Dim StrSearchStr As String
'    StrSearchStr = "c_office_chn = " + Chr(34) + StrSearch + Chr(34)
'    rsOfficeCodes.FindFirst StrSearchStr
'End If
Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click

End Sub

Private Sub mcTree_Click(cNode As clsNode)
Dim tRst As DAO.Recordset, tRstOffice As DAO.Recordset
Dim tOfficeCodeQuery As DAO.QueryDef, prm As DAO.Parameter
Dim tRstOfficeCode As DAO.Recordset, tRstDummy As DAO.Recordset, tLen As Integer
Dim tStrQuery As String

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'

Me.TxtOfficeCode.Value = -1
Me.TxtOfficeDesc.Value = ""
Me.TxtOfficeDescChn.Value = ""

CmdSelect.Enabled = False

' reset the form colors
'
Clear_SelectAll
'
If cNode.Key = "Root" Then
    CmdSelectAll.Enabled = False
    '
    ' reset the entry code choices
    '
    Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)
    '
    Set frmZZZ_OFFICE_CODE.Form.Recordset = gRstOfficeCode
    frmZZZ_OFFICE_CODE.Form.Refresh
Else
    Set tRst = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
    '
    tRst.MoveFirst
    tRst.FindFirst "c_office_type_node_id = " + Chr(34) + cNode.Key + Chr(34)

    'TxtTypeID.Value = Node.Tag
    TxtTypeDesc.Value = tRst!c_office_type_desc
    TxtTypeDescChn.Value = tRst!c_office_type_desc_chn
    tRst.Close
    Set tRst = Nothing
    '

```

```

        CmdSelectAll.Enabled = True
    '
    ' Clear the table
    '
    Set tRstOfficeCode = frmZZZ_OFFICE_CODE.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_OC", dbOpenDynaset)
    Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstDummy
    tRstOfficeCode.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
    cmdSQL.Execute tRecDeleted
    '
    tLen = Len(cNode.Key)
    tStrQuery = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn ) " +
        "SELECT DISTINCT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn " +
        "FROM OFFICE_CODES INNER JOIN OFFICE_CODE_TYPE_REL ON " +
        "OFFICE_CODES.c_office_id = OFFICE_CODE_TYPE_REL.c_office_id " +
        "WHERE (((Left([c_office_tree_id], " & Str(tLen) & ")") = " & cNode.Key & "'))"
    '
    ' now repopulate
    '
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    '
    Set tRstOfficeCode = CurrentDb.OpenRecordset("ZZ_OFFICE_CODE", dbOpenDynaset)
    '
    Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstOfficeCode
    Set tRstDummy = Nothing
    '
End If
End Sub

```

```

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Or IsNull(TxtSearchChn.Text) Then
        If TxtSearch.Value = "" Or IsNull(TxtSearch.Value) Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearch.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Or IsNull(TxtSearch.Text) Then
        If TxtSearchChn.Value = "" Or IsNull(TxtSearchChn.Value) Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

```

```

Private Sub NodeSearch()

    Dim cNode As clsNode, tStr As String, tRstOfficeCode As DAO.Recordset, tRstOfficeTreeIDs As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrSearch As String, tStrLen As String, cmdSQL As ADODB.Command, tStrSearchAlt As String

    ' all specific association codes will have a type of the form 0101
    ' hence the ValuePath for the relevant node will be "K000/K01/K0101"
    ' The command to locate the relevant node is:
    ' cNode = mcTreee.Node(tStrValuePath)
    ' mcTree.activeNode = cNode
    '
    TxtOfficeCode.Value = -1
    TxtOfficeDesc.Value = ""
    TxtOfficeDescChn.Value = ""
    '
    ' search for the search string in ASSOC_CODES
    TxtSearchChn.SetFocus
    tStrSearchChn = Trim(Me.TxtSearchChn.Text)
    TxtSearch.SetFocus
    tStrSearchEng = Trim(Me.TxtSearch.Text)
    tStrSearch = ""

```



```

CmdFind.SetFocus
'
' clear the scratch table
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
cmdSQL.Execute tRecDeleted
'
' because the user may have a hard time picking the exact term, I'll treat it as
' (1) the beginning of the actual term
' (2) part of the term
'
If tStrSearchChn <> "" Then
    tStrLen = Str(LenB(tStrSearchChn))
    gStrSearch = "LeftB(c_office_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
    gStrSearchAlt = "InStrB(1,c_office_chn,'" + tStrSearchChn + "') > 0"
    'tStrSearch = "c_entry_desc_chn = '" + tStrSearchChn + "'"
ElseIf tStrSearchEng <> "" Then
    tStrLen = Str(Len(tStrSearchEng))
    gStrSearch = "Left(c_office_trans," + tStrLen + ") = '" + tStrSearchEng + "'"
    gStrSearchAlt = "InStr(1,c_office_trans,'" + tStrSearchEng + "') > 0"
    'tStrSearch = "c_entry_desc = '" + tStrSearchEng + "'"
End If

gUseAlt = False
If Not (gStrSearch = "") Then

    'MsgBox "Looking for entry"

    Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)

    gRstOfficeCode.FindFirst gStrSearch

    If gRstOfficeCode.NoMatch Then
        gRstOfficeCode.FindFirst gStrSearchAlt
        gUseAlt = True
    End If

    If gRstOfficeCode.NoMatch Then
        gUseAlt = False
        CmdFindNext.Enabled = False
    Else
        CmdFindNext.Enabled = True
        '
        ' next find the entry_type
        '
        'MsgBox "Looking for entry type"

        Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_CODE_TYPE_REL", dbOpenDynaset)

        tRstOfficeTreeIDs.FindNext "c_office_id = " + Str(gRstOfficeCode!c_office_id)

        If Not tRstOfficeTreeIDs.NoMatch Then
            '
            ' set the code values
            TxtOfficeCode.Value = gRstOfficeCode!c_office_id
            If Not IsNull(gRstOfficeCode!c_office_trans) Then
                TxtOfficeDesc.Value = gRstOfficeCode!c_office_trans
            End If
            If Not IsNull(gRstOfficeCode!c_office_chn) Then
                TxtOfficeDescChn.Value = gRstOfficeCode!c_office_chn
            End If
            '
            ' define the string
            '
            'MsgBox "Looking for node"
            '
            tStr = Trim(tRstOfficeTreeIDs!c_office_tree_id)
            '
            ' search the tree
            Set cNode = mcTree.Nodes(tStr)
            '
            If Not IsNull(cNode) Then
                '
                'MsgBox "found node"
                '
                Set mcTree.ActiveNode = cNode
            End If
        End If
    End If

```

```

'
'   then one makes it visible
'
tNode.EnsureVisible
'
Set gNode = cNode
'
'   Finally populate the options and select the record.
'
CmdSelectAll.Enabled = True
'
tStrSQL = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn ) " + _
"SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn
" + _
"FROM OFFICE_CODES INNER JOIN OFFICE_CODE_TYPE_REL ON " + _
"OFFICE_CODES.c_office_id = OFFICE_CODE_TYPE_REL.c_office_id " + _
"WHERE (c_office_tree_id = '" + tStr + "')"

Set tRstOfficeCode = frmZZZ_OFFICE_CODE.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_OC", dbOpenDynaset)
Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstDummy
tRstOfficeCode.Close
'

cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
cmdSQL.Execute tRecDeleted

'MsgBox tStrSQL

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted
'

Set tRstOfficeCode = CurrentDb.OpenRecordset("ZZ_OFFICE_CODE", dbOpenDynaset)
'

Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstOfficeCode
tRstOfficeCode.FindFirst "c_office_id = " + Str(gRstOfficeCode!c_office_id)
frmZZZ_OFFICE_CODE.Form.Refresh
'

Set tRstDummy = Nothing
'
'   briefly set the focus on the tree to get the highlighted node
'

subTreeView.SetFocus
frmZZZ_OFFICE_CODE.SetFocus
'
'   set the tree values
'

Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
'

tRstOfficeTreeIDs.MoveFirst
tRstOfficeTreeIDs.FindFirst "c_office_type_node_id = " + Chr(34) + cNode.Key + Chr(34)

'TxtTypeID.Value = Node.Tag
TxtTypeDesc.Value = tRstOfficeTreeIDs!c_office_type_desc
TxtTypeDescChn.Value = tRstOfficeTreeIDs!c_office_type_desc_chn
tRstOfficeTreeIDs.Close
Set tRstOfficeTreeIDs = Nothing
End If
End If
End If
End If

End Sub

```

frm_frmPickOfficeTree_multi 1

Option Compare Database

Public gRstOfficeCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String, gSelectCount As Integer

'#####Treeview Code#####

'Add this to your form's declaration section

Public WithEvents mcTree As clsTreeview

Private mbExit As Boolean ' to exit a SpinButton event

'#####Treeview Code#####

Private Sub CmdCancel_Click()

On Error GoTo Err_CmdCancel_Click

Clear_SelectAll

DoCmd.Close

Exit_CmdCancel_Click:

Exit Sub

Err_CmdCancel_Click:

MsgBox Err.Description

Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdFindNext_Click()

Dim tRstOfficeCodes As DAO.Recordset, tRstOfficeTypes As DAO.Recordset, cNode As clsNode
Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

TxtOfficeCode.Value = -1

TxtOfficeDesc.Value = ""

TxtOfficeDescChn.Value = ""

,

' clear the scratch table

,

Set cmdSQL = New ADODB.Command

cmdSQL.ActiveConnection = CurrentProject.Connection

cmdSQL.CommandType = adCmdText

,

If IsNull(gRstOfficeCode) Then

MsgBox "Error in search: table closed."

Else

If gStrSearch = "" Then

MsgBox "Error in search: string empty."

Else

MsgBox "Looking for entry"

If gRstOfficeCode.EOF Then

CmdFindNext.Enabled = False

Else

If gUseAlt Then

gRstOfficeCode.FindNext gStrSearchAlt

Else

gRstOfficeCode.FindNext gStrSearch

If gRstOfficeCode.NoMatch Then

gRstOfficeCode.FindFirst gStrSearchAlt

gUseAlt = True

End If

End If

If gRstOfficeCode.NoMatch Then

If gUseAlt Then

gUseAlt = False

End If

CmdFindNext.Enabled = False

Else

,

' next find the entry_type

,

MsgBox "Looking for entry type"

Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_CODE_TYPE_REL", dbOpenDynaset)

tRstOfficeTreeIDs.FindNext "c_office_id = " + Str(gRstOfficeCode!c_office_id)

If Not tRstOfficeTreeIDs.NoMatch Then

,

```

' set the code values
TxtOfficeCode.Value = gRstOfficeCode!c_office_id
If Not IsNull(gRstOfficeCode!c_office_trans) Then
    TxtOfficeDesc.Value = gRstOfficeCode!c_office_trans
End If
If Not IsNull(gRstOfficeCode!c_office_chn) Then
    TxtOfficeDescChn.Value = gRstOfficeCode!c_office_chn
End If
'
' define the string
'
'MsgBox "Looking for node"
'
tStr = Trim(tRstOfficeTreeIDs!c_office_tree_id)
'
' search the tree

Set cNode = mcTree.Nodes(tStr)
'
If Not IsNull(cNode) Then
    If cNode.Key = gNode.Key Then
        gSelectCount = 0
        For i = 0 To ListOffice.ListCount - 1
            If gRstOfficeCode!c_office_id = ListOffice.Column(0, i) Then
                ListOffice.ListIndex = i
                ListOffice.Selected(i) = True
                gSelectCount = gSelectCount + 1
            End If
        Next i
    Else
        Set mcTree.ActiveNode = cNode
        'cNode.Selected = True
        '
        ' then one makes it visible
        '
        'tNode.EnsureVisible
        '
        Set gNode = cNode
        '
        ' Finally populate the options and select the record.
        '
        CmdSelectAll.Enabled = True
        tStrSQL = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn" & _
            "SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c" & _
            "FROM OFFICE_CODES INNER JOIN OFFICE_CODE_TYPE_REL ON " & _
            "OFFICE_CODES.c_office_id = OFFICE_CODE_TYPE_REL.c_office_id " & _
            "WHERE (c_office_tree_id = '" & tStr & "')"

        cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
        cmdSQL.Execute tRecDeleted

        'MsgBox tStrSQL

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted

        ListOffice.Requery
        For ti = 0 To ListOffice.ListCount
            ListOffice.Selected(ti) = False
        Next ti
        gSelectCount = 0
        '
        Set tRstDummy = Nothing
        '
        ' briefly set the focus on the tree to get the highlighted node
        '
        subTreeView.SetFocus
        'frmZZZ_OFFICE_CODE.SetFocus
        '
        ' set the tree values
        '
        Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
        '
        tRstOfficeTreeIDs.MoveFirst
        tRstOfficeTreeIDs.FindFirst "c_office_type_node_id = " & Chr(34) & cNode.Key & Chr(34)
    End If
End If

```

```

        'TxtTypeID.Value = Node.Tag
        TxtTypeDesc.Value = tRstOfficeTreeIDs!c_office_type_desc
        TxtTypeDescChn.Value = tRstOfficeTreeIDs!c_office_type_desc_chn
        tRstOfficeTreeIDs.Close
        Set tRstOfficeTreeIDs = Nothing
    End If
End If
End If
End If
End If
End If
End Sub

Private Sub CmdSelect_Click()
    Clear_SelectAll
    CmdSelectAll.SetFocus
    CmdSelect.Enabled = False
    If TxtOfficeCode.Value <> -1 Then

        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText

        cmdSQL.CommandText = "DELETE * FROM Z_SCRATCH_DUMMY_OC"
        cmdSQL.Execute tRecCount

        ' first copy the records over to a scratch table

        Set tRst = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_OC", dbOpenDynaset)
        For Each varItm In ListOffice.ItemsSelected
            tRst.AddNew
            tRst!c_office_id = ListOffice.Column(0, varItm)
            tRst!c_office_trans = ListOffice.Column(1, varItm)
            tRst!c_office_chn = ListOffice.Column(2, varItm)
            tRst.Update
        Next varItm
        tRst.Close

        ' then clear out
        cmdSQL.CommandText = "DELETE * FROM ZZ_OFFICE_CODE"
        cmdSQL.Execute tRecCount

        cmdSQL.CommandText = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn ) " +
            "SELECT Z_SCRATCH_DUMMY_OC.c_office_id, Z_SCRATCH_DUMMY_OC.c_office_trans, Z_SCRATCH_DUMMY_OC.c_office_chn " +
            "FROM Z_SCRATCH_DUMMY_OC"
        cmdSQL.Execute tRecCount

        ListOffice.Requery
        For ti = 0 To ListOffice.ListCount
            ListOffice.Selected(ti) = False
        Next ti
        gSelectCount = 0

        cmdSQL.CommandText = "DELETE * FROM Z_SCRATCH_DUMMY_OC"
        cmdSQL.Execute tRecCount

    End If
    ' MsgBox "Office Code: " + Str(TxtOfficeCode.Value)
    Forms!frmPickOfficeTree_multi.Visible = False
End Sub

Private Sub CmdSelectAll_Click()
    If CmdSelectAll.Caption = "Select All" Then
        CmdSelectAll.Caption = "De-select All"
        ListOffice.ForeColor = RGB(255, 255, 255)
        ListOffice.BackColor = RGB(0, 0, 255)
        CmdSelect.Enabled = True

        TxtOfficeCode.Value = -1
        TxtOfficeDesc.Value = ""
        TxtOfficeDescChn.Value = ""
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub

Private Sub Clear_SelectAll()

```

```

    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    ListOffice.ForeColor = RGB(0, 0, 0)
    ListOffice.BackColor = RGB(255, 255, 255)
    CmdSelect.Enabled = False
End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim cRoot As clsNode, cNode1 As clsNode, cNode2 As clsNode, cNode3 As clsNode
    Dim cNode4 As clsNode, cNode5 As clsNode, cNode6 As clsNode, cNode7 As clsNode
    Dim strKey As String, strCaption As String
    Dim tRst As DAO.Recordset, cmdSQL As ADODB.Command

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' initialize the Office Codes dataset
    '
    Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)
    '
    cmdSQL.CommandText = "DELETE * FROM ZZ_OFFICE_CODE"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn ) " + _
        "SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn " + _
        "FROM OFFICE_CODES"
    cmdSQL.Execute tRecCount

    ListOffice.Requery
    For ti = 0 To ListOffice.ListCount
        ListOffice.Selected(ti) = False
    Next ti
    'frmZZZ_OFFICE_CODE.Form.OrderBy = "c_sortorder"
    'frmZZZ_OFFICE_CODE.Form.OrderByOn = True

    If Not IsNull(Me.OpenArgs) Then
        Dim strOffice As String
        strOffice = Me.OpenArgs
        Dim rsOffice As DAO.Recordset
    End If

    ' set the language
    Dim tmli As MsoLanguageID
    '
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    ' gLabelsOK = True
    If tmli = msoLanguageIDSimplifiedChinese Then
        gDisplayLanguage = "S"
    ElseIf tmli = msoLanguageIDTraditionalChinese Then
        gDisplayLanguage = "T"
    Else
        gDisplayLanguage = "E"
    End If
    '
    ' build treeview
    '

    Set tRst = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)

    tRst.MoveFirst
    Set mcTree = Me.subTreeView.Form.pTreeview
    With mcTree
        .NodesClear
        .AppName = AppName ' Title for message boxes:
        ' Add a Root node with main and expanded icons and make it bold

        ' use the appropriate caption
        If gDisplayLanguage = "T" Then
            strCaption = ChrW(23448) + ChrW(32887) + ChrW(20998) + ChrW(39006)
        ElseIf gdisplaylangauge = "S" Then
            strCaption = ChrW(23448) + ChrW(32844) + ChrW(20998) + ChrW(31612)
        Else
            strCaption = "Administrative Category"
        End If

        Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
    End With

```

```

cRoot.Bold = True
' Loop through the records
Do While Not tRst.EOF
    ' Add node
    strKey = tRst!c_office_type_node_id
    If gDisplayLanguage = "E" Then
        strCaption = tRst!c_office_type_desc + " " + tRst!c_office_type_desc_chn
    Else
        strCaption = tRst!c_office_type_desc_chn
    End If
    '
    ' Note that since the root record has an ID of "0", its length is 1 and is automatically skipped
    '
    Select Case Len(strKey)
        Case 2
            Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode1.Expanded = False
        Case 4
            Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode2.Expanded = False
        Case 6
            Set cNode3 = cNode2.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode3.Expanded = False
        Case 8
            Set cNode4 = cNode3.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode4.Expanded = False
        Case 10
            Set cNode5 = cNode4.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode5.Expanded = False
        Case 12
            Set cNode6 = cNode5.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode6.Expanded = False
        Case 14
            Set cNode7 = cNode6.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode7.Expanded = False
    End Select
    tRst.MoveNext
Loop
' Create the node controls and display the tree
.Refresh
End With
'
TxtOfficeCode.Value = -1
TxtOfficeTypeType.Value = -1
TxtOfficeL1.Value = -1
TxtOfficeL2.Value = -1
TxtOfficeL3.Value = -1
TxtOfficeL4.Value = -1
TxtOfficeL5.Value = -1
TxtTypeDesc.Value = ""
TxtTypeDescChn.Value = ""
TxtOfficeDesc.Value = ""
TxtOfficeDescChn.Value = ""
gSelectCount = 0

CmdSelect.Enabled = False
Set cmdSQL = Nothing
Set tRst = Nothing
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

'Dim StrSearch As String
'Me.TxtSearch.SetFocus
'StrSearch = Me.TxtSearch.Value
'If StrSearch <> "" Then
'    Dim rsOfficeCodes As DAO.Recordset
'    Set rsOfficeCodes = frmZZZ_OFFICE_CODE.Form.Recordset
'    Dim StrSearchStr As String
'    StrSearchStr = "c_office_chn = " + Chr(34) + StrSearch + Chr(34)
'    rsOfficeCodes.FindFirst StrSearchStr
'End If
Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click

```

End Sub

```
Private Sub ListOffice_Click()
    Dim ti As Long, tUnclicked As Boolean
    Dim varItm As Variant

    ti = ListOffice.ListIndex
    If ListOffice.Selected(ti + 1) Then
        gSelectCount = gSelectCount + 1
        tUnclicked = False
    Else
        gSelectCount = gSelectCount - 1
        tUnclicked = True
    End If
    'MsgBox ListOffice.Column(1, ti + 1) + ": Select Count = " + Str(gSelectCount)

    If gSelectCount = 0 Then
        Me.TxtOfficeDesc.Value = ""
        Me.TxtOfficeDescChn.Value = ""
        Me.TxtOfficeCode.Value = 0
        Me.CmdSelect.Enabled = False
    Else
        If gSelectCount = 1 Then
            If tUnclicked Then
                ' this means that there is only on selected item, but it is NOT this item
                ' we therefore need to locate the selected item and put its values into the text boxes
                ' I may not even use these boxes anymore, but just in case...
                For Each varItm In ListOffice.ItemsSelected
                    Me.TxtOfficeDesc.Value = ListOffice.Column(1, varItm)
                    Me.TxtOfficeDescChn.Value = ListOffice.Column(2, varItm)
                    Me.TxtOfficeCode.Value = ListOffice.Column(0, varItm)
                    'MsgBox ListOffice.Column(1, varItm)
                Next varItm
            Else
                Me.TxtOfficeDesc.Value = ListOffice.Column(1, ti + 1)
                Me.TxtOfficeDescChn.Value = ListOffice.Column(2, ti + 1)
                Me.TxtOfficeCode.Value = ListOffice.Column(0, ti + 1)
                'MsgBox ListOffice.Column(1, ti + 1)
            End If
        Else
            Me.TxtOfficeDesc.Value = "Multi-select"
            Me.TxtOfficeDescChn.Value = "Multi-select"
            Me.TxtOfficeCode.Value = -2
            'MsgBox "Multi-select"
        End If
        Me.CmdSelect.Enabled = True
    End If
End Sub
```

End Sub

```
Private Sub mcTree_Click(cNode As clsNode)
    Dim tRst As DAO.Recordset, tRstOffice As DAO.Recordset
    Dim tOfficeCodeQuery As DAO.QueryDef, prm As DAO.Parameter
    Dim tRstOfficeCode As DAO.Recordset, tRstDummy As DAO.Recordset, tLen As Integer
    Dim tStrQuery As String

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    Me.TxtOfficeCode.Value = -1
    Me.TxtOfficeDesc.Value = ""
    Me.TxtOfficeDescChn.Value = ""

    CmdSelect.Enabled = False

    ' reset the form colors
    '
    Clear_SelectAll
    '
    If cNode.Key = "Root" Then
        CmdSelectAll.Enabled = False
        '
        ' reset the entry code choices
        '
        Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)
        '
        cmdSQL.CommandText = "DELETE * FROM ZZ_OFFICE_CODE"
```



```

cmdSQL.Execute tRecCount

cmdSQL.CommandText = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn ) " + _
                        "SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn " + _
                        "FROM OFFICE_CODES"
cmdSQL.Execute tRecCount

Else
Set tRst = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
'
tRst.MoveFirst
tRst.FindFirst "c_office_type_node_id = " + Chr(34) + cNode.Key + Chr(34)

'TxtTypeID.Value = Node.Tag
TxtTypeDesc.Value = tRst!c_office_type_desc
TxtTypeDescChn.Value = tRst!c_office_type_desc_chn
tRst.Close
Set tRst = Nothing
'
CmdSelectAll.Enabled = True
'
cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
cmdSQL.Execute tRecDeleted
'
tLen = Len(cNode.Key)
tStrQuery = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn ) " +
            "SELECT DISTINCT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn " +
            "FROM OFFICE_CODES INNER JOIN OFFICE_CODE_TYPE_REL ON " +
            "OFFICE_CODES.c_office_id = OFFICE_CODE_TYPE_REL.c_office_id " +
            "WHERE (((Left([c_office_tree_id], " & Str(tLen) & ")='" & cNode.Key & "')))"
'
' now repopulate
'
cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecDeleted
'
'
End If

ListOffice.Requery
For ti = 0 To ListOffice.ListCount
ListOffice.Selected(ti) = False
Next ti
gSelectCount = 0
End Sub

Private Sub TxtSearchChn_Change()
If TxtSearchChn.Text = "" Or IsNull(TxtSearchChn.Text) Then
If TxtSearch.Value = "" Or IsNull(TxtSearch.Value) Then
Me.CmdFind.Enabled = False
End If
Else
TxtSearch.Value = ""
CmdFind.Enabled = True
End If
CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearch_Change()
If TxtSearch.Text = "" Or IsNull(TxtSearch.Text) Then
If TxtSearchChn.Value = "" Or IsNull(TxtSearchChn.Value) Then
Me.CmdFind.Enabled = False
End If
Else
TxtSearchChn.Value = ""
CmdFind.Enabled = True
End If
CmdFindNext.Enabled = False
End Sub

Private Sub NodeSearch()

Dim cNode As clsNode, tStr As String, tRstOfficeCode As DAO.Recordset, tRstOfficeTreeIDs As DAO.Recordset
Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
Dim tStrSearch As String, tStrLen As String, cmdSQL As ADODB.Command, tStrSearchAlt As String

' all specific association codes will have a type of the form 0101
' hence the ValuePath for the relevant node will be "K000/K01/K0101"

```

```

' The command to locate the relevant node is:
' cNode = mcTreeee.Node(tStrValuePath)
' mcTree.activeNode = cNode
'
TxtOfficeCode.Value = -1
TxtOfficeDesc.Value = ""
TxtOfficeDescChn.Value = ""
'
' search for the search string in ASSOC_CODES
TxtSearchChn.SetFocus
tStrSearchChn = Trim(Me.TxtSearchChn.Text)
TxtSearch.SetFocus
tStrSearchEng = Trim(Me.TxtSearch.Text)
tStrSearch = ""
CmdFind.SetFocus
'
' clear the scratch table
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
cmdSQL.Execute tRecDeleted
'
' because the user may have a hard time picking the exact term, I'll treat it as
' (1) the beginning of the actual term
' (2) part of the term
'
If tStrSearchChn <> "" Then
    tStrLen = Str(LenB(tStrSearchChn))
    gStrSearch = "LeftB(c_office_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
    gStrSearchAlt = "InStrB(1,c_office_chn,'" + tStrSearchChn + "') > 0"
    'tStrSearch = "c_entry_desc_chn = '" + tStrSearchChn + "'"
ElseIf tStrSearchEng <> "" Then
    tStrLen = Str(Len(tStrSearchEng))
    gStrSearch = "Left(c_office_trans," + tStrLen + ") = '" + tStrSearchEng + "'"
    gStrSearchAlt = "InStr(1,c_office_trans,'" + tStrSearchEng + "') > 0"
    'tStrSearch = "c_entry_desc = '" + tStrSearchEng + "'"
End If

gUseAlt = False
If Not (gStrSearch = "") Then

    'MsgBox "Looking for entry"

    Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)

    gRstOfficeCode.FindFirst gStrSearch

    If gRstOfficeCode.NoMatch Then
        gRstOfficeCode.FindFirst gStrSearchAlt
        gUseAlt = True
    End If

    If gRstOfficeCode.NoMatch Then
        gUseAlt = False
        CmdFindNext.Enabled = False
    Else
        CmdFindNext.Enabled = True
        '
        ' next find the entry_type
        '
        'MsgBox "Looking for entry type"

        Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_CODE_TYPE_REL", dbOpenDynaset)

        tRstOfficeTreeIDs.FindNext "c_office_id = " + Str(gRstOfficeCode!c_office_id)

        If Not tRstOfficeTreeIDs.NoMatch Then
            '
            ' set the code values
            TxtOfficeCode.Value = gRstOfficeCode!c_office_id
            If Not IsNull(gRstOfficeCode!c_office_trans) Then
                TxtOfficeDesc.Value = gRstOfficeCode!c_office_trans
            End If
            If Not IsNull(gRstOfficeCode!c_office_chn) Then
                TxtOfficeDescChn.Value = gRstOfficeCode!c_office_chn
            End If
            '
        End If
    End If

```

```

'   define the string
'
'MsgBox "Looking for node"
'
tStr = Trim(tRstOfficeTreeIDs!c_office_tree_id)
'
'   search the tree
Set cNode = mcTree.Nodes(tStr)
'
If Not IsNull(cNode) Then
'
'MsgBox "found node"
'
Set mcTree.ActiveNode = cNode
'
'   then one makes it visible
'
'tNode.EnsureVisible
'
Set gNode = cNode
'
'   Finally populate the options and select the record.
'
CmdSelectAll.Enabled = True
'
cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
cmdSQL.Execute tRecDeleted

tStrSQL = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn ) " + _
"SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn

"FROM OFFICE_CODES INNER JOIN OFFICE_CODE_TYPE_REL ON " + _
"OFFICE_CODES.c_office_id = OFFICE_CODE_TYPE_REL.c_office_id " + _
"WHERE (c_office_tree_id = '" + tStr + "')"

'MsgBox tStrSQL

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted
'
ListOffice.Requery
For i = 0 To ListOffice.ListCount
    ListOffice.Selected(i) = False
Next i
gSelectCount = 0
'
For i = 0 To ListOffice.ListCount - 1
    If gRstOfficeCode!c_office_id = ListOffice.Column(0, i) Then
        ListOffice.ListIndex = i
        ListOffice.Selected(i) = True
        gSelectCount = gSelectCount + 1
    End If
Next i
'
'   briefly set the focus on the tree to get the highlighted node
'
subTreeView.SetFocus
'
'   set the tree values
'
Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
'
tRstOfficeTreeIDs.MoveFirst
tRstOfficeTreeIDs.FindFirst "c_office_type_node_id = " + Chr(34) + cNode.Key + Chr(34)

'TxtTypeID.Value = Node.Tag
TxtTypeDesc.Value = tRstOfficeTreeIDs!c_office_type_desc
TxtTypeDescChn.Value = tRstOfficeTreeIDs!c_office_type_desc_chn
tRstOfficeTreeIDs.Close
Set tRstOfficeTreeIDs = Nothing
End If
End If
End If
If gSelectCount = 0 Then
    CmdSelect.Enabled = False
Else
    CmdSelect.Enabled = True
End If
End Sub

```



```
Option Compare Database
```

```
Public gRstOfficeCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String, gSelectCount As Integer, gStrDynasty As String, gStrDynast
yChn As String
```

```
'#####Treeview Code#####
```

```
'Add this to your form's declaration section
```

```
Public WithEvents mcTree As clsTreeview
```

```
Private mbExit As Boolean ' to exit a SpinButton event
```

```
'/#####Treeview Code#####
```

```
Private Sub CmdCancel_Click()
```

```
On Error GoTo Err_CmdCancel_Click
```

```
Clear_SelectAll
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
```

```
Exit Sub
```

```
Err_CmdCancel_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()
```

```
Dim cmdSQL As ADODB.Command, connSQL As ADODB.Connection, ti As Integer, tMultiDynasty As Boolean, tStrDynast
y As String, tStrDynastyChn As String
```

```
CmdSelectAll.SetFocus
```

```
CmdSelect.Enabled = False
```

```
gSelectCount = 0
```

```
For Each varItm In ListOffice.ItemsSelected
```

```
gSelectCount = gSelectCount + 1
```

```
Next varItm
```

```
If gSelectCount = 1 Then
```

```
' this means that there is only on selected item
```

```
For Each varItm In ListOffice.ItemsSelected
```

```
Me.TxtTypeDesc.Value = ListOffice.Column(1, varItm)
```

```
Me.TxtTypeDescChn.Value = ListOffice.Column(2, varItm)
```

```
Me.TxtOfficeDesc.Value = ListOffice.Column(3, varItm)
```

```
Me.TxtOfficeDescChn.Value = ListOffice.Column(4, varItm)
```

```
Me.TxtOfficeCode.Value = ListOffice.Column(0, varItm)
```

```
Next varItm
```

```
ElseIf gSelectCount = ListOffice.ListCount - 1 Then
```

```
Me.TxtOfficeDesc.Value = "All"
```

```
Me.TxtOfficeDescChn.Value = "All"
```

```
Me.TxtOfficeCode.Value = -1
```

```
Else
```

```
Me.TxtOfficeDesc.Value = "Multi-select"
```

```
Me.TxtOfficeDescChn.Value = "Multi-select"
```

```
Me.TxtOfficeCode.Value = -2
```

```
MsgBox "Multi-select"
```

```
End If
```

```
Set cmdSQL = New ADODB.Command
```

```
cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
cmdSQL.CommandType = adCmdText
```

```
cmdSQL.CommandText = "DELETE * FROM ZZ_OFFICE_CODE"
```

```
cmdSQL.Execute tRecCount
```

```
tMultiDynasty = False
```

```
Set tRst = CurrentDb.OpenRecordset("ZZ_OFFICE_CODE", dbOpenDynaset)
```

```
For Each varItm In ListOffice.ItemsSelected
```

```
tRst.AddNew
```

```
tRst!c_office_id = ListOffice.Column(0, varItm)
```

```
tRst!c_office_trans = ListOffice.Column(3, varItm)
```

```
tRst!c_office_chn = ListOffice.Column(4, varItm)
```

```
tRst!c_dynasty = ListOffice.Column(1, varItm)
```

```
tRst!c_dynasty_chn = ListOffice.Column(2, varItm)
```

```
tRst.Update
```

```
Next varItm
```

```
tRst.Close
```

```
,
```

```

frmPickOfficeTree_multi2 - 2
' get the dynasty code
'
cmdSQL.CommandText = "UPDATE OFFICE_CODES INNER JOIN ZZ_OFFICE_CODE ON OFFICE_CODES.c_office_id = ZZ_OFFICE_C
ODE.c_office_id " + _
"SET ZZ_OFFICE_CODE.c_dy = [OFFICE_CODES].[c_dy];"

cmdSQL.Execute tRecCount

ListOffice.Requery
'For ti = 0 To ListOffice.ListCount
'    ListOffice.Selected(ti) = False
'Next ti
gSelectCount = 0

' MsgBox "Office Code: " + Str(TxtOfficeCode.Value)
Forms!frmPickOfficeTree_multi2.Visible = False
End Sub

Private Sub CmdSelectAll_Click()
Dim ti As Long
If CmdSelectAll.Caption = "Select All" Then
    CmdSelectAll.Caption = "De-select All"
    For ti = 0 To ListOffice.ListCount
        ListOffice.Selected(ti) = True
    Next ti
    CmdSelect.Enabled = True

    TxtOfficeCode.Value = -1
    TxtOfficeDesc.Value = gStrDynasty
    TxtOfficeDescChn.Value = gStrDynastyChn
Else
    TxtOfficeDesc.Value = ""
    TxtOfficeDescChn.Value = ""
    CmdSelectAll.SetFocus
    Clear_SelectAll
End If
End Sub

Private Sub Clear_SelectAll()
CmdSelectAll.Caption = "Select All"
'
'    reset the form colors
'
For ti = 0 To ListOffice.ListCount
    ListOffice.Selected(ti) = False
Next ti
CmdSelect.Enabled = False
End Sub

Private Sub Form_Open(Cancel As Integer)
Dim cRoot As clsNode, cNode1 As clsNode, cNode2 As clsNode, cNode3 As clsNode
Dim cNode4 As clsNode, cNode5 As clsNode, cNode6 As clsNode, cNode7 As clsNode
Dim strKey As String, strCaption As String
Dim tRst As DAO.Recordset, cmdSQL As ADODB.Command, connSQL As ADODB.Connection

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
'    initialize the Office Codes dataset
'
Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)
'
cmdSQL.CommandText = "DELETE * FROM ZZ_OFFICE_CODE_TMP"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "INSERT INTO ZZ_OFFICE_CODE_TMP ( c_office_id, c_office_trans, c_office_chn, c_dy, c_dyn
asty, c_dynasty_chn ) " + _
"SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn
, DYNASTIES.c_dy, " + _
"DYNASTIES.c_dynasty, DYNASTIES.c_dynasty_chn " + _
"FROM OFFICE_CODES INNER JOIN DYNASTIES ON OFFICE_CODES.c_dy = DYNASTIES.c_dy"
cmdSQL.Execute tRecCount

ListOffice.Requery
'For ti = 0 To ListOffice.ListCount
'    ListOffice.Selected(ti) = False
'Next ti
'frmZZZ_OFFICE_CODE.Form.OrderBy = "c_sortorder"
'frmZZZ_OFFICE_CODE.Form.OrderByOn = True

```

```

If Not IsNull(Me.OpenArgs) Then
    Dim strOffice As String
    strOffice = Me.OpenArgs
    Dim rsOffice As DAO.Recordset
End If

' set the language
Dim tmli As MsoLanguageID
'
tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
' gLabelsOK = True
If tmli = msoLanguageIDSimplifiedChinese Then
    gDisplayLanguage = "S"
ElseIf tmli = msoLanguageIDTraditionalChinese Then
    gDisplayLanguage = "T"
Else
    gDisplayLanguage = "E"
End If
'
' build treeview
'
Set tRst = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)

tRst.MoveFirst
Set mcTree = Me.subTreeView.Form.pTreeview
With mcTree
    .NodesClear
    .AppName = AppName ' Title for message boxes:
    ' Add a Root node with main and expanded icons and make it bold

    ' use the appropriate caption
    If gDisplayLanguage = "T" Then
        strCaption = ChrW(23448) + ChrW(32887) + ChrW(20998) + ChrW(39006)
    ElseIf gdisplaylangauge = "S" Then
        strCaption = ChrW(23448) + ChrW(32844) + ChrW(20998) + ChrW(31612)
    Else
        strCaption = "Administrative Category"
    End If

    Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
    cRoot.Bold = True
    ' Loop through the records
    Do While Not tRst.EOF
        ' Add node
        strKey = tRst!c_office_type_node_id
        If gDisplayLanguage = "E" Then
            strCaption = tRst!c_office_type_desc + " " + tRst!c_office_type_desc_chn
        Else
            strCaption = tRst!c_office_type_desc_chn
        End If
        '
        ' Note that since the root record has an ID of "0", its length is 1 and is automatically skipped
        '
        Select Case Len(strKey)
            Case 2
                Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode1.Expanded = False
            Case 4
                Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode2.Expanded = False
            Case 6
                Set cNode3 = cNode2.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode3.Expanded = False
            Case 8
                Set cNode4 = cNode3.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode4.Expanded = False
            Case 10
                Set cNode5 = cNode4.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode5.Expanded = False
            Case 12
                Set cNode6 = cNode5.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode6.Expanded = False
            Case 14
                Set cNode7 = cNode6.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode7.Expanded = False
        End Select
        tRst.MoveNext
    Loop
    ' Create the node controls and display the tree

```

```

.Refresh
End With
'
TxtOfficeCode.Value = -1
TxtOfficeTypeType.Value = -1
TxtOfficeL1.Value = -1
TxtOfficeL2.Value = -1
TxtOfficeL3.Value = -1
TxtOfficeL4.Value = -1
TxtOfficeL5.Value = -1
TxtTypeDesc.Value = ""
TxtTypeDescChn.Value = ""
TxtOfficeDesc.Value = ""
TxtOfficeDescChn.Value = ""
gSelectCount = 0
gStrDynasty = ""
gStrDynastyChn = ""
'
' adjust language of labels
'
Dim tLabelLanguage(3, 5) As String, tLang As Integer, tRstLabelList As DAO.Recordset, tLabelsOK As Boolean

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

tLabelsOK = False
With tRstLabelList
.MoveFirst
ti = 1

Do While ti < 5 And Not .EOF
If !c_form = "POT" Then
tLabelsOK = True
If ti <> !c_label_id Then
MsgBox "Uh oh: mismatched label table"
tLabelsOK = False
Exit Do
End If
tLabelLanguage(1, ti) = !c_english
tLabelLanguage(2, ti) = !c_fanti
tLabelLanguage(3, ti) = !c_jianti
ti = ti + 1
End If
.MoveNext
Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If tLabelsOK Then
If gDisplayLanguage = "E" Then
tLang = 1
ElseIf gDisplayLanguage = "T" Then
tLang = 2
Else
tLang = 3
End If
'
' now comes the basic routine
'
Me.CmdCancel.Caption = tLabelLanguage(tLang, 1)
Me.CmdSelect.Caption = tLabelLanguage(tLang, 2)
Me.CmdSelectAll.Caption = tLabelLanguage(tLang, 3)
Me.LblChkAltNames.Caption = tLabelLanguage(tLang, 4)
End If

CmdSelect.Enabled = False
Set cmdSQL = Nothing
Set tRst = Nothing
Set connSQL = Nothing
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

```



```
Err_CmdFind_Click:
```

```
    MsgBox Err.Description
    Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub ListOffice_Click()
```

```
    Dim ti As Long, tUnclicked As Boolean
    Dim varItm As Variant
```

```
    gSelectCount = 0
```

```
    For Each varItm In ListOffice.ItemsSelected
```

```
        gSelectCount = gSelectCount + 1
```

```
    Next varItm
```

```
    'MsgBox ListOffice.Column(1, ti + 1) + ": Select Count = " + Str(gSelectCount)
```

```
    If gSelectCount = 0 Then
```

```
        Me.TxtOfficeDesc.Value = ""
```

```
        Me.TxtOfficeDescChn.Value = ""
```

```
        Me.TxtOfficeCode.Value = 0
```

```
        Me.CmdSelect.Enabled = False
```

```
    Else
```

```
        Me.CmdSelect.Enabled = True
```

```
    End If
```

```
End Sub
```

```
Private Sub mcTree_Click(cNode As clsNode)
```

```
    Dim tRst As DAO.Recordset, tRstOffice As DAO.Recordset
```

```
    Dim tOfficeCodeQuery As DAO.QueryDef, prm As DAO.Parameter
```

```
    Dim tRstOfficeCode As DAO.Recordset, tRstDummy As DAO.Recordset, tLen As Integer
```

```
    Dim tStrQuery As String, cmdSQL As ADODB.Command, connSQL As ADODB.Connection
```

```
    Set cmdSQL = New ADODB.Command
```

```
    cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
    cmdSQL.CommandType = adCmdText
```

```
    Me.TxtOfficeCode.Value = -1
```

```
    Me.TxtOfficeDesc.Value = ""
```

```
    Me.TxtOfficeDescChn.Value = ""
```

```
    CmdSelect.Enabled = False
```

```
    ' reset the form colors
```

```
    Clear_SelectAll
```

```
    If cNode.Key = "Root" Then
```

```
        CmdSelectAll.Enabled = False
```

```
        ' reset the entry code choices
```

```
        Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)
```

```
        cmdSQL.CommandText = "DELETE * FROM ZZ_OFFICE_CODE_TMP"
```

```
        cmdSQL.Execute tRecCount
```

```
        cmdSQL.CommandText = "INSERT INTO ZZ_OFFICE_CODE_TMP ( c_office_id, c_office_trans, c_office_chn, c_dy, c_dynasty, c_dynasty_chn ) " +
```

```
        "SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn, DYNASTIES.c_dy, " +
```

```
        "DYNASTIES.c_dynasty, DYNASTIES.c_dynasty_chn " +
```

```
        "FROM OFFICE_CODES INNER JOIN DYNASTIES ON OFFICE_CODES.c_dy = DYNASTIES.c_dy"
```

```
        cmdSQL.Execute tRecCount
```

```
        gStrDynasty = ""
```

```
        gStrDynastyChn = ""
```

```
    Else
```

```
        Set tRst = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
```

```
        tRst.MoveFirst
```

```
        tRst.FindFirst "c_office_type_node_id = " + Chr(34) + cNode.Key + Chr(34)
```

```
        'TxtTypeID.Value = Node.Tag
```

```
        TxtTypeDesc.Value = tRst!c_office_type_desc
```

```
        TxtTypeDescChn.Value = tRst!c_office_type_desc_chn
```

```
        tRst.Close
```

```

' since the keys all refer to a dynasty in the left(cNode.Key,2), we can find the dynasty
'
Set tRst = CurrentDb.OpenRecordset("SELECT c_dynasty, c_dynasty_chn FROM DYNASTIES WHERE DYNASTIES.c_dy = " + Left(cNode.Key, 2))
tRst.MoveFirst
gStrDynasty = tRst!c_dynasty
gStrDynastyChn = tRst!c_dynasty_chn
tRst.Close
Set tRst = Nothing
'
CmdSelectAll.Enabled = True
'
cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE_TMP"
cmdSQL.Execute tRecDeleted
'
tLen = Len(cNode.Key)
tStrQuery = "INSERT INTO ZZ_OFFICE_CODE_TMP ( c_office_id, c_office_trans, c_office_chn, c_dy, c_dynasty, c_dynasty_chn ) " + _
"SELECT DISTINCT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn, DYNASTIES.c_dy, " + _
"DYNASTIES.c_dynasty, DYNASTIES.c_dynasty_chn " + _
"FROM (OFFICE_CODES INNER JOIN DYNASTIES ON OFFICE_CODES.c_dy = DYNASTIES.c_dy) INNER JOIN OF" & _
"FFICE_CODE_TYPE_REL ON " + _
"OFFICE_CODES.c_office_id = OFFICE_CODE_TYPE_REL.c_office_id " + _
"WHERE (((Left([c_office_tree_id]," + Str(tLen) + ")=' " + cNode.Key + "')))"
'
' now repopulate
'
cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecDeleted
'
End If

ListOffice.Requery
'For ti = 0 To ListOffice.ListCount
' ListOffice.Selected(ti) = False
'Next ti
gSelectCount = 0
End Sub

Private Sub TxtSearchChn_Change()
If TxtSearchChn.Text = "" Or IsNull(TxtSearchChn.Text) Then
If TxtSearch.Value = "" Or IsNull(TxtSearch.Value) Then
Me.CmdFind.Enabled = False
End If
Else
TxtSearch.Value = ""
CmdFind.Enabled = True
End If
End Sub

Private Sub TxtSearch_Change()
If TxtSearch.Text = "" Or IsNull(TxtSearch.Text) Then
If TxtSearchChn.Value = "" Or IsNull(TxtSearchChn.Value) Then
Me.CmdFind.Enabled = False
End If
Else
TxtSearchChn.Value = ""
CmdFind.Enabled = True
End If
End Sub

Private Sub NodeSearch()

'Dim cNode As clsNode, tStr As String, tRstOfficeCode As DAO.Recordset, tRstOfficeTreeIDs As DAO.Recordset
'Dim tStrSQL As String, tRstDummy As DAO.Recordset, tStrSearch As String, tStrLen As String
Dim tStrSearchChn As String, tStrSearchEng As String, tRecNum As Long, tRecNumAlt As Long, tStr As String
Dim cmdSQL As ADODB.Command, tStrSearchAlt As String, connSQL As ADODB.Connection

' all specific association codes will have a type of the form 0101
' hence the ValuePath for the relevant node will be "K000/K01/K0101"
' The command to locate the relevant node is:
' cNode = mcTreee.Node(tStrValuePath)
' mcTree.activeNode = cNode
'
TxtOfficeCode.Value = -1
TxtOfficeDesc.Value = ""
TxtOfficeDescChn.Value = ""
'
' search for the search string in ASSOC_CODES

```

```

TxtSearchChn.SetFocus
tStrSearchChn = Trim(Me.TxtSearchChn.Text)
TxtSearch.SetFocus
tStrSearchEng = Trim(Me.TxtSearch.Text)
CmdFind.SetFocus
'
'tStrSearch = ""
'
' clear the scratch table
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' Instead of a search, this is now a filter
'
tStr = "Quit"
If IsNull(tStrSearchChn) Then
    If Not IsNull(tStrSearchEng) Then
        If Not (tStrSearchEng = "") Then
            tStr = " OFFICE_CODES.c_office_trans LIKE '%" + Trim(tStrSearchEng) + "%'"
        End If
    End If
Else
    If tStrSearchChn = "" Then
        If Not IsNull(tStrSearchEng) Then
            If Not (tStrSearchEng = "") Then
                tStr = " OFFICE_CODES.c_office_trans LIKE '%" + Trim(tStrSearchEng) + "%'"
            End If
        End If
    Else
        tStr = " OFFICE_CODES.c_office_chn LIKE '%" + Trim(tStrSearchChn) + "%'"
    End If
End If

tRecNum = 0
If (tStr = "Quit") Then
    connSQL.Close
    Exit Sub
Else
    cmdSQL.CommandText = "Delete * from Z_SCRATCH_DUMMY_OC"
    cmdSQL.Execute tRecNum
    '
    cmdSQL.CommandText = "INSERT INTO Z_SCRATCH_DUMMY_OC ( c_office_id, c_office_trans, c_office_chn, c_dy, c_dynasty, c_dynasty_chn ) " + _
        "SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn, DYNASTIES.c_dy, " + _
        "DYNASTIES.c_dynasty, DYNASTIES.c_dynasty_chn " + _
        "FROM OFFICE_CODES INNER JOIN DYNASTIES ON OFFICE_CODES.c_dy = DYNASTIES.c_dy " + _
        "WHERE " + tStr
    cmdSQL.Execute tRecNum
    '
    If Me.ChkAltNames.Value Then
        If tStrSearchChn = "" Then
            If Not IsNull(tStrSearchEng) Then
                If Not (tStrSearchEng = "") Then
                    tStr = " OFFICE_CODES.c_office_trans_alt LIKE '%" + Trim(tStrSearchEng) + "%'"
                End If
            End If
        Else
            tStr = " OFFICE_CODES.c_office_chn_alt LIKE '%" + Trim(tStrSearchChn) + "%'"
        End If
        cmdSQL.CommandText = "INSERT INTO Z_SCRATCH_DUMMY_OC ( c_office_id, c_office_trans, c_office_chn, c_dy, c_dynasty, c_dynasty_chn ) " + _
            "SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn, DYNASTIES.c_dy, " + _
            "DYNASTIES.c_dynasty, DYNASTIES.c_dynasty_chn " + _
            "FROM OFFICE_CODES INNER JOIN DYNASTIES ON OFFICE_CODES.c_dy = DYNASTIES.c_dy " + _
            "WHERE " + tStr
        cmdSQL.Execute tRecNumAlt

        tRecNum = tRecNum + tRecNumAlt
    End If

    If tRecNum = 0 Then
        connSQL.Close
        Exit Sub
    Else
        cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE_TMP"
    End If
End If

```

```

        cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = "INSERT INTO ZZ_OFFICE_CODE_TMP ( c_office_id, c_office_trans, c_office_chn, c_dynasty, c_dynasty_chn ) " + _
        "SELECT DISTINCT Z_SCRATCH_DUMMY_OC.c_office_id, Z_SCRATCH_DUMMY_OC.c_office_trans, Z_SCRATCH_DUMMY_OC.c_office_chn, " + _
        "Z_SCRATCH_DUMMY_OC.c_dy, Z_SCRATCH_DUMMY_OC.c_dynasty, Z_SCRATCH_DUMMY_OC.c_dynasty_chn " + _
        "FROM Z_SCRATCH_DUMMY_OC " + _
        "ORDER BY Z_SCRATCH_DUMMY_OC.c_dynasty"
    cmdSQL.Execute tRecNum

    ListOffice.Requery
    'For i = 0 To ListOffice.ListCount
    '    ListOffice.Selected(i) = False
    'Next i
    gSelectCount = 0

    'Set cNode = mcTree.Nodes(tstr)
    '
    '    the idea here is that this will be a mix and match, so set the tree to the top
    '
    'cNode.Key = "Root"
End If
End If
'
' Put the filter string into the type field
If IsNull(tStrSearchChn) Then
    If Not IsNull(tStrSearchEng) Then
        If Not (tStrSearchEng = "") Then
            TxtTypeDescChn.Value = ""
            TxtTypeDesc.Value = "[Filter] " + tStrSearchEng
        End If
    End If
Else
    If tStrSearchChn = "" Then
        If Not IsNull(tStrSearchEng) Then
            If Not (tStrSearchEng = "") Then
                TxtTypeDescChn.Value = ""
                TxtTypeDesc.Value = "[Filter] " + tStrSearchEng
            End If
        End If
    Else
        TxtTypeDesc.Value = ""
        TxtTypeDescChn.Value = "[Filter] " + tStrSearchChn
    End If
End If

If gSelectCount = 0 Then
    CmdSelect.Enabled = False
Else
    CmdSelect.Enabled = True
End If
End Sub

```

frmPickOfficeTree_old -1

Option Compare Database

Public gRstOfficeCode As DAO.Recordset, gNode As Node, gStrSearch As String, gStrSearchAlt As String

Public gUseAlt As Boolean

Private Sub CmdCancel_Click()

On Error GoTo Err_CmdCancel_Click

Clear_SelectAll

DoCmd.Close

Exit_CmdCancel_Click:

Exit Sub

Err_CmdCancel_Click:

MsgBox Err.Description

Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdFindNext_Click()

Dim tRstAssocCodes As DAO.Recordset, tRstAssocTypes As DAO.Recordset, tNode As Node

Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

TxtOfficeCode.Value = -1

TxtOfficeDesc.Value = ""

TxtOfficeDescChn.Value = ""

,

' clear the scratch table

,

Set cmdSQL = New ADODB.Command

cmdSQL.ActiveConnection = CurrentProject.Connection

cmdSQL.CommandType = adCmdText

,

If IsNull(gRstOfficeCode) Then

MsgBox "Error in search: table closed."

Else

If gStrSearch = "" Then

MsgBox "Error in search: string empty."

Else

MsgBox "Looking for entry"

If gRstOfficeCode.EOF Then

CmdFindNext.Enabled = False

Else

If gUseAlt Then

gRstOfficeCode.FindNext gStrSearchAlt

Else

gRstOfficeCode.FindNext gStrSearch

If gRstOfficeCode.NoMatch Then

gRstOfficeCode.FindFirst gStrSearchAlt

gUseAlt = True

End If

End If

If gRstOfficeCode.NoMatch Then

If gUseAlt Then

gUseAlt = False

End If

CmdFindNext.Enabled = False

Else

,

' next find the entry_type

,

MsgBox "Looking for entry type"

Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_CODE_TYPE_REL", dbOpenDynaset)

tRstOfficeTreeIDs.FindNext "c_office_id = " + Str(gRstOfficeCode!c_office_id)

If Not tRstOfficeTreeIDs.NoMatch Then

,

' set the code values

TxtOfficeCode.Value = gRstOfficeCode!c_office_id

If Not IsNull(gRstOfficeCode!c_office_trans) Then

TxtOfficeDesc.Value = gRstOfficeCode!c_office_trans

End If

```

    If Not IsNull(gRstOfficeCode!c_office_chn) Then
        TxtOfficeDescChn.Value = gRstOfficeCode!c_office_chn
    End If
    '
    '   define the string
    '
    'MsgBox "Looking for node"
    '
    tStr = Trim(tRstOfficeTreeIDs!c_office_tree_id)
    '
    '   search the tree

Set tNode = TreeViewTypes.Nodes("K" + tStr)
'
If Not IsNull(tNode) Then
    If tNode = gNode Then
        Set tRstOfficeCode = frmZZZ_OFFICE_CODE.Form.Recordset
        tRstOfficeCode.FindFirst "c_office_id = " + Str(gRstOfficeCode!c_office_id)
        frmZZZ_OFFICE_CODE.Form.Refresh
    Else
        tNode.Selected = True
        '
        '   then one makes it visible
        '
        tNode.EnsureVisible
        '
        Set gNode = tNode
        '
        '   Finally populate the options and select the record.
        '
        CmdSelectAll.Enabled = True
        '
        tStrSQL = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn
) " + _
                "SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c
                "FROM OFFICE_CODES INNER JOIN OFFICE_CODE TYPE_REL ON " +
                "OFFICE_CODES.c_office_id = OFFICE_CODE TYPE_REL.c_office_id " + _
                "WHERE (c_office_tree_id = '" + tStr + "')"

        Set tRstOfficeCode = frmZZZ_OFFICE_CODE.Form.Recordset
        Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_OC", dbOpenDynaset)
        Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstDummy
        tRstOfficeCode.Close
        '
        cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
        cmdSQL.Execute tRecDeleted

        'MsgBox tStrSQL

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
        Set tRstOfficeCode = CurrentDb.OpenRecordset("ZZ_OFFICE_CODE", dbOpenDynaset)
        '
        Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstOfficeCode
        tRstOfficeCode.FindFirst "c_office_id = " + Str(gRstOfficeCode!c_office_id)
        frmZZZ_OFFICE_CODE.Form.Refresh
        '
        Set tRstDummy = Nothing
        '
        '   briefly set the focus on the tree to get the highlighted node
        '
        TreeViewTypes.SetFocus
        frmZZZ_OFFICE_CODE.SetFocus
        '
        '   set the tree values
        '
        Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
        '
        tRstOfficeTreeIDs.MoveFirst
        tRstOfficeTreeIDs.FindFirst "c_office_type_node_id = " + Chr(34) + tNode.Tag + Ch

        'TxtTypeID.Value = Node.Tag
        TxtTypeDesc.Value = tRstOfficeTreeIDs!c_office_type_desc
        TxtTypeDescChn.Value = tRstOfficeTreeIDs!c_office_type_desc_chn
        tRstOfficeTreeIDs.Close
        Set tRstOfficeTreeIDs = Nothing

```

frmPickOfficeTree_old -3

```
End If
End If
End If
End If
End If
End If
End Sub
```

```
Private Sub CmdSelect_Click()
    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    frmZZZ_OFFICE_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
    frmZZZ_OFFICE_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
    CmdSelectAll.SetFocus
    CmdSelect.Enabled = False
    If TxtOfficeCode.Value > -1 Then
        TxtOfficeCode.Value = frmZZZ_OFFICE_CODE.Form.Recordset!c_office_id
    End If
    ' MsgBox "Office Code: " + Str(TxtOfficeCode.Value)
    Forms!frmPickOfficeTree.Visible = False
End Sub
```

```
Private Sub CmdSelectAll_Click()
    If CmdSelectAll.Caption = "Select All" Then
        CmdSelectAll.Caption = "De-select All"
        frmZZZ_OFFICE_CODE.Form.DatasheetForeColor = RGB(255, 255, 255)
        frmZZZ_OFFICE_CODE.Form.DatasheetBackColor = RGB(0, 0, 255)
        CmdSelect.Enabled = True
        TxtOfficeCode.Value = -1
        TxtOfficeDesc.Value = ""
        TxtOfficeDescChn.Value = ""
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub
```

```
Private Sub Clear_SelectAll()
    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    frmZZZ_OFFICE_CODE.Form.DatasheetForeColor = RGB(0, 0, 0)
    frmZZZ_OFFICE_CODE.Form.DatasheetBackColor = RGB(255, 255, 255)
    CmdSelect.Enabled = False
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
    Dim tNode As Node, tStrNode As String, tStrParent As String
    Dim tRst As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    '
    ' initialize the Assoc Codes dataset
    '
    Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)
    Set frmZZZ_OFFICE_CODE.Form.Recordset = gRstOfficeCode
    '
    'frmZZZ_OFFICE_CODE.Form.OrderBy = "c_sortorder"
    'frmZZZ_OFFICE_CODE.Form.OrderByOn = True

    If Not IsNull(Me.OpenArgs) Then
        Dim strOffice As String
        strOffice = Me.OpenArgs
        Dim rsOffice As DAO.Recordset
        Set rsOffice = frmZZZ_OFFICE_CODE.Form.Recordset
        rsOffice.FindFirst "c_office_id = " & strOffice
    End If
    '
    ' build treeview
    '
    Set tRst = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
    TreeViewTypes.Nodes.Clear
    Set tNode = TreeViewTypes.Nodes.Add(, , "K000", "Dynastic Office Hierarchies")
    tNode.Expanded = True
    tNode.Tag = "000"
    '
    ' the general syntax for adding nodes is:
```

```

'      TreeViewTypes.Nodes.Add("K...", tvwChild, parent node, "Text")
'
tRst.MoveFirst
Do While Not tRst.EOF
    If Not IsNull(tRst!c_parent_id) Then
        If tRst!c_parent_id = "0" Then
            tStrParent = "K000"
        Else
            tStrParent = "K" + Trim(tRst!c_parent_id)
        End If
        tStrNode = "K" + Trim(tRst!c_office_type_node_id)
        Set tNode = TreeViewTypes.Nodes.Add(tStrParent, tvwChild, tStrNode, tRst!c_office_type_desc)
        ' Since I have eliminated the secondary tree elements, the following code is no longer valid
        ' I simply assign the node ID to the tag
        '
        ' snip the headers for the secondary tree elements for the tags
        '
        'If Val(Mid(tRst!c_office_type_node_id, 1, 2)) > 4 Then
        '    tNode.Tag = Mid(tRst!c_office_type_node_id, 3)
        'Else
        '    tNode.Tag = tRst!c_office_type_node_id
        'End If
        tNode.Tag = tRst!c_office_type_node_id
    End If
    tRst.MoveNext
Loop
'
TxtOfficeCode.Value = -1
TxtOfficeTypeType.Value = -1
TxtOfficeL1.Value = -1
TxtOfficeL2.Value = -1
TxtOfficeL3.Value = -1
TxtOfficeL4.Value = -1
TxtOfficeL5.Value = -1
TxtTypeDesc.Value = ""
TxtTypeDescChn.Value = ""
TxtOfficeDesc.Value = ""
TxtOfficeDescChn.Value = ""

' clear the first pass of the file

cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
cmdSQL.Execute tRecDeleted

Set cmdSQL = Nothing
Set tRst = Nothing
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

'Dim StrSearch As String
'Me.TxtSearch.SetFocus
'StrSearch = Me.TxtSearch.Value
'If StrSearch <> "" Then
'    Dim rsOfficeCodes As DAO.Recordset
'    Set rsOfficeCodes = frmZZZ_OFFICE_CODE.Form.Recordset
'    Dim StrSearchStr As String
'    StrSearchStr = "c_office_chn = " + Chr(34) + StrSearch + Chr(34)
'    rsOfficeCodes.FindFirst StrSearchStr
'End If
Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click

End Sub

Private Sub TreeViewTypes_NodeClick(ByVal Node As Object)
Dim tRst As DAO.Recordset, tRstOffice As DAO.Recordset
Dim tOfficeCodeQuery As DAO.QueryDef, prm As DAO.Parameter
Dim tRstOfficeCode As DAO.Recordset, tRstDummy As DAO.Recordset, tLen As Integer
Dim tStrQuery As String

```



```

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

```

```

Me.TxtOfficeCode.Value = -1
Me.TxtOfficeDesc.Value = ""
Me.TxtOfficeDescChn.Value = ""

```

```

CmdSelect.Enabled = False

```

```

' reset the form colors

```

```

Clear_SelectAll

```

```

If Node.Tag = "000" Then

```

```

    CmdSelectAll.Enabled = False

```

```

    ' reset the entry code choices

```

```

    Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)

```

```

    Set frmZZZ_OFFICE_CODE.Form.Recordset = gRstOfficeCode

```

```

    frmZZZ_OFFICE_CODE.Form.Refresh

```

```

Else

```

```

    Set tRst = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)

```

```

    tRst.MoveFirst

```

```

    tRst.FindFirst "c_office_type_node_id = " + Chr(34) + Node.Tag + Chr(34)

```

```

    'TxtTypeID.Value = Node.Tag

```

```

    TxtTypeDesc.Value = tRst!c_office_type_desc

```

```

    TxtTypeDescChn.Value = tRst!c_office_type_desc_chn

```

```

    tRst.Close

```

```

    Set tRst = Nothing

```

```

    CmdSelectAll.Enabled = True

```

```

    ' Clear the table

```

```

    Set tRstOfficeCode = frmZZZ_OFFICE_CODE.Form.Recordset

```

```

    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_OC", dbOpenDynaset)

```

```

    Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstDummy

```

```

    tRstOfficeCode.Close

```

```

    cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"

```

```

    cmdSQL.Execute tRecDeleted

```

```

    tLen = Len(Node.Tag)

```

```

    tStrQuery = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn ) " + _
        "SELECT DISTINCT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn " + _

```

```

        "FROM OFFICE_CODES INNER JOIN OFFICE_CODE_TYPE_REL ON " + _

```

```

        "OFFICE_CODES.c_office_id = OFFICE_CODE_TYPE_REL.c_office_id " + _

```

```

        "WHERE (((Left([c_office_tree_id], " & Str(tLen) & ")='" & Node.Tag & "')))"

```

```

    ' now repopulate

```

```

    cmdSQL.CommandText = tStrQuery

```

```

    cmdSQL.Execute tRecDeleted

```

```

    Set tRstOfficeCode = CurrentDb.OpenRecordset("ZZ_OFFICE_CODE", dbOpenDynaset)

```

```

    Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstOfficeCode

```

```

    Set tRstDummy = Nothing

```

```

End If

```

```

End Sub

```

```

Private Sub TxtSearchChn_Change()

```

```

    If TxtSearchChn.Text = "" Or IsNull(TxtSearchChn.Text) Then

```

```

        If TxtSearch.Value = "" Or IsNull(TxtSearch.Value) Then

```

```

            Me.CmdFind.Enabled = False

```

```

        End If

```

```

    Else

```

```

        TxtSearch.Value = ""

```

```

        CmdFind.Enabled = True

```

```

    End If

```

```

    CmdFindNext.Enabled = False
End Sub
Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Or IsNull(TxtSearch.Text) Then
        If TxtSearchChn.Value = "" Or IsNull(TxtSearchChn.Value) Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub NodeSearch()

    Dim tNode As Node, tStr As String, tRstOfficeCode As DAO.Recordset, tRstOfficeTreeIDs As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrSearch As String, tStrLen As String, cmdSQL As ADODB.Command, tStrSearchAlt As String

    ' all specific association codes will have a type of the form 0101
    ' hence the ValuePath for the relevant node will be "K000/K01/K0101"
    ' The command to locate the relevant node is:
    ' tNode = TreeViewType.FindNode(tStrValuePath)
    ' TreeViewType.SelectedNode = tNode
    '
    TxtOfficeCode.Value = -1
    TxtOfficeDesc.Value = ""
    TxtOfficeDescChn.Value = ""
    '
    ' search for the search string in ASSOC_CODES
    TxtSearchChn.SetFocus
    tStrSearchChn = Trim(Me.TxtSearchChn.Text)
    TxtSearch.SetFocus
    tStrSearchEng = Trim(Me.TxtSearch.Text)
    tStrSearch = ""
    CmdFind.SetFocus
    '
    ' clear the scratch table
    '
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
    cmdSQL.Execute tRecDeleted
    '
    ' because the user may have a hard time picking the exact term, I'll treat it as
    ' (1) the beginning of the actual term
    ' (2) part of the term
    '
    If tStrSearchChn <> "" Then
        tStrLen = Str(LenB(tStrSearchChn))
        gStrSearch = "LeftB(c_office_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
        gStrSearchAlt = "InStrB(1,c_office_chn,'" + tStrSearchChn + "') > 0"
        'tStrSearch = "c_entry_desc_chn = '" + tStrSearchChn + "'"
    ElseIf tStrSearchEng <> "" Then
        tStrLen = Str(Len(tStrSearchEng))
        gStrSearch = "Left(c_office_trans," + tStrLen + ") = '" + tStrSearchEng + "'"
        gStrSearchAlt = "InStr(1,c_office_trans,'" + tStrSearchEng + "') > 0"
        'tStrSearch = "c_entry_desc = '" + tStrSearchEng + "'"
    End If

    gUseAlt = False
    If Not (gStrSearch = "") Then

        'MsgBox "Looking for entry"

        Set gRstOfficeCode = CurrentDb.OpenRecordset("OFFICE_CODES", dbOpenDynaset)

        gRstOfficeCode.FindFirst gStrSearch

        If gRstOfficeCode.NoMatch Then
            gRstOfficeCode.FindFirst gStrSearchAlt
            gUseAlt = True
        End If

        If gRstOfficeCode.NoMatch Then
            gUseAlt = False
        End If
    End If

```

```

    CmdFindNext.Enabled = False
Else
    CmdFindNext.Enabled = True
    '
    ' next find the entry_type
    '
    'MsgBox "Looking for entry type"

    Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_CODE_TYPE_REL", dbOpenDynaset)

    tRstOfficeTreeIDs.FindNext "c_office_id = " + Str(gRstOfficeCode!c_office_id)

    If Not tRstOfficeTreeIDs.NoMatch Then
        '
        ' set the code values
        TxtOfficeCode.Value = gRstOfficeCode!c_office_id
        If Not IsNull(gRstOfficeCode!c_office_trans) Then
            TxtOfficeDesc.Value = gRstOfficeCode!c_office_trans
        End If
        If Not IsNull(gRstOfficeCode!c_office_chn) Then
            TxtOfficeDescChn.Value = gRstOfficeCode!c_office_chn
        End If
        '
        ' define the string
        '
        'MsgBox "Looking for node"
        '
        tStr = Trim(tRstOfficeTreeIDs!c_office_tree_id)
        '
        ' search the tree

        Set tNode = TreeViewTypes.Nodes("K" + tStr)
        '
        If Not IsNull(tNode) Then
            '
            'MsgBox "found node"
            '
            tNode.Selected = True
            '
            ' then one makes it visible
            '
            tNode.EnsureVisible
            '
            Set gNode = tNode
            '
            ' Finally populate the options and select the record.
            '
            CmdSelectAll.Enabled = True
            '
            tStrSQL = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id, c_office_trans, c_office_chn ) " + _
                "SELECT OFFICE_CODES.c_office_id, OFFICE_CODES.c_office_trans, OFFICE_CODES.c_office_chn " + _
                "FROM OFFICE_CODES INNER JOIN OFFICE_CODE_TYPE_REL ON " + _
                "OFFICE_CODES.c_office_id = OFFICE_CODE_TYPE_REL.c_office_id " + _
                "WHERE (c_office_tree_id = '" + tStr + "')"

            Set tRstOfficeCode = frmZZZ_OFFICE_CODE.Form.Recordset
            Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_OC", dbOpenDynaset)
            Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstDummy
            tRstOfficeCode.Close
            '
            cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
            cmdSQL.Execute tRecDeleted

            'MsgBox tStrSQL

            cmdSQL.CommandText = tStrSQL
            cmdSQL.Execute tRecDeleted
            '
            Set tRstOfficeCode = CurrentDb.OpenRecordset("ZZ_OFFICE_CODE", dbOpenDynaset)
            '
            Set frmZZZ_OFFICE_CODE.Form.Recordset = tRstOfficeCode
            tRstOfficeCode.FindFirst "c_office_id = " + Str(gRstOfficeCode!c_office_id)
            frmZZZ_OFFICE_CODE.Form.Refresh
            '
            Set tRstDummy = Nothing
            '
            ' briefly set the focus on the tree to get the highlighted node
            '
            TreeViewTypes.SetFocus

```

```
        frmZZZ_OFFICE_CODE.SetFocus
    '
    '   set the tree values
    '
    Set tRstOfficeTreeIDs = CurrentDb.OpenRecordset("OFFICE_TYPE_TREE", dbOpenDynaset)
    '
    tRstOfficeTreeIDs.MoveFirst
    tRstOfficeTreeIDs.FindFirst "c_office_type_node_id = " + Chr(34) + tNode.Tag + Chr(34)

    'TxtTypeID.Value = Node.Tag
    TxtTypeDesc.Value = tRstOfficeTreeIDs!c_office_type_desc
    TxtTypeDescChn.Value = tRstOfficeTreeIDs!c_office_type_desc_chn
    tRstOfficeTreeIDs.Close
    Set tRstOfficeTreeIDs = Nothing
End If
End If
End If
End Sub
```

Option Compare Database

```
Private Sub CmdClose_Click()
On Error GoTo Err_CmdClose_Click
```

```
    DoCmd.Close
```

```
Exit_CmdClose_Click:
Exit Sub
```

```
Err_CmdClose_Click:
MsgBox Err.Description
Resume Exit_CmdClose_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()
On Error GoTo Err_CmdSelect_Click
```

```
Forms("frmPickPeople").Visible = False
```

```
Exit_CmdSelect_Click:
Exit Sub
```

```
Err_CmdSelect_Click:
MsgBox Err.Description
Resume Exit_CmdSelect_Click
```

```
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
frmPerson.Form.OrderBy = "c_name"
frmPerson.Form.OrderByOn = True
```

```
    If Not IsNull(Me.OpenArgs) Then
        Dim strID As String
        strID = Me.OpenArgs
        Dim rst As DAO.Recordset
        Set rst = frmPerson.Form.Recordset
        rst.FindFirst "c_personid = " & strID
    End If
```

```
End Sub
```

```
Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click
```

```
Dim StrSearchChn As String, StrSearchPY As String
Dim StrSearchStr As String
Dim rsPpl As DAO.Recordset
```

```
Set rsPpl = frmPerson.Form.Recordset
' Me.TxtSearch.SetFocus
StrSearchChn = Me.TxtSearch.Value
StrSearchPY = Me.TxtSearchName.Value
If StrSearchChn <> "" Then
    StrSearchStr = "c_name_chn = " & Chr(34) & StrSearchChn & Chr(34)
    rsPpl.FindFirst StrSearchStr
ElseIf StrSearchPY <> "" Then
    StrSearchStr = "c_name = " & Chr(34) & StrSearchPY & Chr(34)
    rsPpl.FindFirst StrSearchStr
End If
frmPerson.Form.Refresh
```

```
Exit_CmdFind_Click:
Exit Sub
```

```
Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()
If TxtSearch.Text = "" Then
    If TxtSearchName.Value = "" Then
        CmdFind.Enabled = False
    End If
Else
    TxtSearchName.Value = ""
    CmdFind.Enabled = True
End If
```

Form_frmPickPeople - 2

```
End If
End Sub

Private Sub TxtSearchName_Change()
    If Me.TxtSearchName.Text = "" Then
        If TxtSearch.Value = "" Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearch.Value = ""
        CmdFind.Enabled = True
    End If
End Sub
```

frm_frmPickPosting - 1

Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
Forms!frmPickPosting.Visible = False  
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)  
frmPosting.Form.OrderBy = "c_posting_id DESC"  
frmPosting.Form.OrderByOn = True
```

```
    If Not IsNull(Me.OpenArgs) Then  
        Dim strPosting As String  
        strPosting = Me.OpenArgs  
        Dim rst As DAO.Recordset  
        Set rst = frmPosting.Form.Recordset  
        rst.FindFirst "c_posting_id = " & strPosting  
    End If
```

```
End Sub
```

```
Private Sub CmdFind_Click()  
On Error GoTo Err_CmdFind_Click
```

```
    Dim StrSearch As String  
    Me.TxtSearch.SetFocus  
    StrSearch = Me.TxtSearch.Value  
    If StrSearch <> "" Then  
        Dim rsPerson As DAO.Recordset  
        Set rsPerson = frmPosting.Form.Recordset  
        Dim StrSearchStr As String  
        StrSearchStr = "c_name_chn = " & Chr(34) & StrSearch & Chr(34)  
        rsPerson.FindFirst StrSearchStr  
    End If
```

```
Exit_CmdFind_Click:  
Exit Sub
```

```
Err_CmdFind_Click:  
MsgBox Err.Description  
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()  
    If Me.TxtSearch.Text = "" Then  
        Me.CmdFind.Enabled = False  
    Else  
        Me.CmdFind.Enabled = True  
    End If  
End Sub
```

frm_frmPickScholarlyTopic -1

Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
Forms!frmPickScholarlyTopic.Visible = False  
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
    frmScholarlyTopic.Form.OrderBy = "c_sortorder"  
    frmScholarlyTopic.Form.OrderByOn = True
```

```
    If Not IsNull(Me.OpenArgs) Then  
        Dim strTOPIC As String  
        strTOPIC = Me.OpenArgs  
        Dim rsTopic As DAO.Recordset  
        Set rsTopic = frmScholarlyTopic.Form.Recordset  
        rsTopic.FindFirst "c_topic_code = " & strTOPIC  
    End If
```

```
End Sub
```

```
Private Sub TxtSearch_Change()  
If Me.TxtSearch.Text = "" Then  
    Me.CmdFind.Enabled = False  
Else  
    Me.CmdFind.Enabled = True  
End If  
End Sub
```

```
Private Sub CmdFind_Click()  
On Error GoTo Err_CmdFind_Click
```

```
    Dim StrSearch As String  
    Me.TxtSearch.SetFocus  
    StrSearch = Me.TxtSearch.Value  
    If StrSearch <> "" Then  
        Dim rsTopic As DAO.Recordset  
        Set rsTopic = frmScholarlyTopic.Form.Recordset  
        Dim StrSearchStr As String  
        StrSearchStr = "c_topic_desc_chn = " + Chr(34) + StrSearch + Chr(34)  
        rsTopic.FindFirst StrSearchStr  
    End If
```

```
Exit_CmdFind_Click:  
Exit Sub
```

```
Err_CmdFind_Click:  
MsgBox Err.Description  
Resume Exit_CmdFind_Click
```

```
End Sub
```


Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
Forms!frmPickSTATUS_CODES.Visible = False  
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)  
If Not IsNull(Me.OpenArgs) Then  
    Dim strStatus As String  
    strStatus = Me.OpenArgs  
    Dim rsStatus As DAO.Recordset  
    Set rsStatus = frmSTATUS_CODES.Form.Recordset  
    rsStatus.FindFirst "c_status_code = " & strStatus  
End If
```

```
End Sub
```

```
Private Sub CmdFind_Click()  
On Error GoTo Err_CmdFind_Click
```

```
    Dim StrSearch As String  
    Me.TxtSearch.SetFocus  
    StrSearch = Me.TxtSearch.Value  
    If StrSearch <> "" Then  
        Dim rsStatus As DAO.Recordset  
        Set rsStatus = frmSTATUS_CODES.Form.Recordset  
        Dim StrSearchStr As String  
        StrSearchStr = "c_status_desc_chn = " + Chr(34) + StrSearch + Chr(34)  
        rsStatus.FindFirst StrSearchStr  
    End If
```

```
Exit_CmdFind_Click:  
Exit Sub
```

```
Err_CmdFind_Click:  
MsgBox Err.Description  
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()  
If Me.TxtSearch.Text = "" Then  
    Me.CmdFind.Enabled = False  
Else  
    Me.CmdFind.Enabled = True  
End If  
End Sub
```

frmPickStatus_multi - 1

Option Compare Database

Public gRstStatusCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String, gSelectCount As Integer

'#####Treeview Code#####

'Add this to your form's declaration section

Public WithEvents mcTree As clsTreeview

Private mbExit As Boolean ' to exit a SpinButton event

'#####Treeview Code#####

Private Sub CmdCancel_Click()

On Error GoTo Err_CmdCancel_Click

Clear_SelectAll

DoCmd.Close

Exit_CmdCancel_Click:

Exit Sub

Err_CmdCancel_Click:

MsgBox Err.Description

Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdSelect_Click()

Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, varItm As Variant, ti As Integer

gSelectCount = 0

For Each varItm In ListStatus.ItemsSelected

gSelectCount = gSelectCount + 1

Next varItm

If gSelectCount = 0 Then

Me.TxtStatusDesc.Value = ""

Me.TxtStatusDescChn.Value = ""

Me.TxtStatusID.Value = 0

Me.CmdSelect.Enabled = False

Else

If gSelectCount = 1 Then

' this means that there is only on selected item

For Each varItm In ListStatus.ItemsSelected

Me.TxtStatusDesc.Value = ListStatus.Column(1, varItm)

Me.TxtStatusDescChn.Value = ListStatus.Column(2, varItm)

Me.TxtStatusID.Value = ListStatus.Column(0, varItm)

'MsgBox ListStatus.Column(1, varItm)

Next varItm

ElseIf gSelectCount = ListStatus.ListCount - 1 Then

Me.TxtStatusDesc.Value = "All"

Me.TxtStatusDescChn.Value = "All"

Me.TxtStatusID.Value = -1

Else

Me.TxtStatusDesc.Value = "Multi-select"

Me.TxtStatusDescChn.Value = "Multi-select"

Me.TxtStatusID.Value = -2

'MsgBox "Multi-select"

End If

Set cmdSQL = New ADODB.Command

cmdSQL.ActiveConnection = CurrentProject.Connection

cmdSQL.CommandType = adCmdText

' copy the records over to ZZ_STATUS_CODE

cmdSQL.CommandText = "DELETE * FROM ZZ_STATUS_CODE"

cmdSQL.Execute tRecCount

Set tRst = CurrentDb.OpenRecordset("ZZ_STATUS_CODE", dbOpenDynaset)

For Each varItm In ListStatus.ItemsSelected

tRst.AddNew

tRst!c_status_code = ListStatus.Column(0, varItm)

tRst!c_status_desc = ListStatus.Column(1, varItm)

tRst!c_status_desc_chn = ListStatus.Column(2, varItm)

tRst.Update

Next varItm

tRst.Close

ListStatus.Requery

'For ti = 0 To ListStatus.ListCount

' ListStatus.Selected(ti) = False

```

frmPickStatus_multi - 2
        'Next ti
        gSelectCount = 0

    End If
    Forms!frmPickStatus_multi.Visible = False
End Sub

Private Sub CmdSelectAll_Click()
    Dim ti As Long

    If CmdSelectAll.Caption = "Select All" Then
        CmdSelectAll.Caption = "De-select All"

        For ti = 0 To ListStatus.ListCount
            ListStatus.Selected(ti) = True
        Next ti

        CmdSelect.Enabled = True
        TxtStatusID.Value = -1
        TxtStatusDesc.Value = ""
        TxtStatusDescChn.Value = ""
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub

Private Sub Clear_SelectAll()
    Dim ti As Long

    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    For ti = 0 To ListStatus.ListCount
        ListStatus.Selected(ti) = False
    Next ti

    gSelectCount = 0
    CmdSelect.Enabled = False
End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim cRoot As clsNode, strKey As String, strCaption As String
    Dim cNode1 As clsNode, cNode2 As clsNode
    Dim tRst As DAO.Recordset, cmdSQL As ADODB.Command
    '
    ' initialize the text fields
    '
    Me.TxtTypeDesc.Value = ""
    Me.TxtTypeDescChn.Value = ""
    Me.TxtTypeID.Value = "000"
    gSelectCount = 0
    '
    ' initialize the Status Codes dataset
    '
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    Set gRstStatusCode = CurrentDb.OpenRecordset("STATUS_CODES", dbOpenDynaset)

    cmdSQL.CommandText = "DELETE * FROM ZZ_STATUS_CODE_TMP"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "INSERT INTO ZZ_STATUS_CODE_TMP ( c_status_code, c_status_desc, c_status_desc_chn ) " + _
        "SELECT STATUS_CODES.c_status_code, STATUS_CODES.c_status_desc, STATUS_CODES.c_status_de" & _
        "sc_chn " + _
        "FROM STATUS_CODES"
    cmdSQL.Execute tRecCount

    ListStatus.Requery
    For ti = 0 To ListStatus.ListCount
        ListStatus.Selected(ti) = False
    Next ti
    '
    'frmSTATUS_CODES.Form.OrderBy = "c_sortorder"
    'frmSTATUS_CODES.Form.OrderByOn = True

    If Not IsNull(Me.OpenArgs) Then
        Dim strStatus As String

```

```

        strStatus = Me.OpenArgs
    End If
    '
    ' build treeview
    '
    Set tRst = CurrentDb.OpenRecordset("STATUS_TYPES", dbOpenDynaset)

    ' set the language
    Dim tmli As MsoLanguageID
    '
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    '
    If tmli = msoLanguageIDSimplifiedChinese Then
        gDisplayLanguage = "S"
    ElseIf tmli = msoLanguageIDTraditionalChinese Then
        gDisplayLanguage = "T"
    Else
        gDisplayLanguage = "E"
    End If
    '
    '
    'MsgBox "About to build tree"
    tRst.MoveFirst
    Set mcTree = Me.subTreeView.Form.pTreeview
    With mcTree
        .NodesClear
        .AppName = AppName ' Title for message boxes:
        ' use the appropriate caption
        If gDisplayLanguage = "T" Then
            strCaption = ChrW(31038) + ChrW(26371) + ChrW(38364) + ChrW(20418) + ChrW(20998) + ChrW(39006)
        ElseIf gdisplaylangauge = "S" Then
            strCaption = ChrW(31038) + ChrW(20250) + ChrW(20851) + ChrW(31995) + ChrW(20998) + ChrW(31612)
        Else
            strCaption = "Categories of Social Status"
        End If

        Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
        ' Add a Root node with main and expanded icons and make it bold
        cRoot.Bold = True
        ' Loop through the records
        Do While Not tRst.EOF
            ' Add node
            strKey = tRst!c_status_type_code
            If gDisplayLanguage = "E" Then
                strCaption = tRst!c_status_type_desc
            Else
                strCaption = tRst!c_status_type_chn
            End If
            If Len(tRst!c_status_type_code) = 2 Then
                Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode1.Expanded = False
            Else
                Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
                cNode2.Expanded = False
            End If
            tRst.MoveNext
        Loop
        ' Create the node controls and display the tree
        .Refresh
    End With
    '
    Set tRst = Nothing
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

    Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click

End Sub

Private Sub ListStatus_Click()
    Dim ti As Long, tUnclicked As Boolean
    Dim varItm As Variant

```

```

gSelectCount = 0

For Each varItm In ListStatus.ItemsSelected
    gSelectCount = gSelectCount + 1
Next varItm

If gSelectCount = 0 Then
    Me.CmdSelect.Enabled = False
Else
    Me.CmdSelect.Enabled = True
End If
'MsgBox ListStatus.Column(1, ti + 1) + ": Select Count = " + Str(gSelectCount)

End Sub

Private Sub mcTree_Click(cNode As clsNode)
    Dim tRst As DAO.Recordset, tRstStatus As DAO.Recordset, tStrSQL As String
    Dim tStatusCodeQuery As DAO.QueryDef, prm As DAO.Parameter
    Dim tRstStatusCode As DAO.Recordset, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    TxtStatusID.Value = -1
    TxtStatusDesc.Value = ""
    TxtStatusDescChn.Value = ""
    CmdSelect.Enabled = False

    ' reset the form colors
    ,
Clear_SelectAll
    ,
    If cNode.Key = "Root" Then
        TxtTypeID.Value = ""
        TxtTypeDesc.Value = ""
        TxtTypeDescChn.Value = ""
        CmdSelectAll.Enabled = False
        ,
        ' reset the entry code choices
        ,
        cmdSQL.CommandText = "Delete * from ZZ_STATUS_CODE_TMP"
        cmdSQL.Execute tRecDeleted
        ,
        Set gRstStatusCode = CurrentDb.OpenRecordset("STATUS_CODES", dbOpenDynaset)
        ,
        cmdSQL.CommandText = "INSERT INTO ZZ_STATUS_CODE_TMP ( c_status_code, c_status_desc, c_status_desc_chn ) " + _
            "SELECT STATUS_CODES.c_status_code, STATUS_CODES.c_status_desc, STATUS_CODES.c_status_desc_chn " + _
            "FROM STATUS_CODES"
        cmdSQL.Execute tRecCount

    Else
        Set tRst = CurrentDb.OpenRecordset("STATUS_TYPES", dbOpenDynaset)
        ,
        tRst.MoveFirst
        tRst.FindFirst "c_status_type_code = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key
        TxtTypeDesc.Value = tRst!c_status_type_desc
        TxtTypeDescChn.Value = tRst!c_status_type_chn
        tRst.Close
        Set tRst = Nothing
        ,
        CmdSelectAll.Enabled = True
        ,
        ' we need to distinguish between type / subtype
        ,
        tStrSQL = "INSERT INTO ZZ_STATUS_CODE_TMP ( c_status_code, c_status_desc, c_status_desc_chn ) " + _
            "SELECT STATUS_CODE_TYPE_REL.c_status_code AS c_status_code, STATUS_CODES.c_status_desc, " + _
            "STATUS_CODES.c_status_desc_chn " + _
            "FROM STATUS_CODES INNER JOIN STATUS_CODE_TYPE_REL ON " + _
            "STATUS_CODES.c_status_code = STATUS_CODE_TYPE_REL.c_status_code "

        If Len(cNode.Key) = 2 Then
            tStrSQL = tStrSQL + "WHERE (((Left([STATUS_CODE_TYPE_REL].[c_status_type_code],2))=' " + _
                TxtTypeID.Value + "'))"
        Else

```

frmPickStatus_multi - 5

```
tStrSQL = tStrSQL + "WHERE (((STATUS_CODE_TYPE_REL.c_status_type_code)='" + _
    TxtTypeID.Value + "'))"
End If

cmdSQL.CommandText = "Delete * from ZZ_STATUS_CODE_TMP"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted
'
End If

ListStatus.Requery
For ti = 0 To ListStatus.ListCount
    ListStatus.Selected(ti) = False
Next ti
gSelectCount = 0

End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Or IsNull(TxtSearch.Text) Then
        If TxtSearchChn.Value = "" Or IsNull(TxtSearchChn.Value) Then
            CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Or IsNull(TxtSearchChn.Text) Then
        If TxtSearch.Value = "" Or IsNull(TxtSearch.Value) Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearch.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub CmdFindNext_Click()
    Dim tRstStatusCodes As DAO.Recordset, tRstStatusTypes As DAO.Recordset, cNode As clsNode
    Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

    TxtStatusID.Value = -1
    TxtStatusDesc.Value = ""
    TxtStatusDescChn.Value = ""

    If IsNull(gRstStatusCode) Then
        MsgBox "Error in search: table closed."
    Else
        If gStrSearch = "" Then
            MsgBox "Error in search: string empty."
        Else
            'MsgBox "Looking for entry"

            If gRstStatusCode.EOF Then
                CmdFindNext.Enabled = False
            Else
                ' gRstStatusCode.MoveNext

                If gUseAlt Then
                    gRstStatusCode.FindNext gStrSearchAlt
                Else
                    gRstStatusCode.FindNext gStrSearch

                    If gRstStatusCode.NoMatch Then
                        gRstStatusCode.FindFirst gStrSearchAlt
                        gUseAlt = True
                    End If
                End If
            End If

            If gRstStatusCode.NoMatch Then
                If gUseAlt Then
                    gUseAlt = False
                End If
            End If
        End If
    End Sub
```

```

        CmdFindNext.Enabled = False
Else
    '
    ' next find the entry_type
    '
    'MsgBox "Looking for entry type"

    Set tRstStatusTypes = CurrentDb.OpenRecordset("STATUS_CODE_TYPE_REL", dbOpenDynaset)

    tRstStatusTypes.FindNext "c_status_code = " + Str(gRstStatusCode!c_status_code)

    If Not tRstStatusTypes.NoMatch Then
        '
        ' set the values
        '
        TxtStatusID.Value = gRstStatusCode!c_status_code
        If Not IsNull(gRstStatusCode!c_status_desc) Then
            TxtStatusDesc.Value = gRstStatusCode!c_status_desc
        End If
        If Not IsNull(gRstStatusCode!c_status_desc_chn) Then
            TxtStatusDescChn.Value = gRstStatusCode!c_status_desc_chn
        End If
        '
        ' define the string
        '
        tStr = tRstStatusTypes!c_status_type_code
        '
        ' search the tree

        Set cNode = mcTree.Nodes(tStr)
        '
        'MsgBox "found node"
        '
        If Not IsNull(cNode) Then
            If cNode.Key <> gNode.Key Then
                Set mcTree.ActiveNode = cNode
                'cNode.Selected = True
                '
                ' then one makes it visible
                '
                'tNode.EnsureVisible
                '
                Set gNode = cNode
                '
                ' Finally populate the options and select the record.
                '
                CmdSelectAll.Enabled = True
                '
                tStrSQL = "INSERT INTO ZZ_STATUS_CODE_TMP ( c_status_code, c_status_desc, c_status_desc_chn ) " + _
                    "SELECT STATUS_CODE_TYPE_REL.c_status_code AS c_status_code, STATUS_CODES.c_s" & _
                    "STATUS_CODES.c_status_desc_chn " + _
                    "FROM STATUS_CODES INNER JOIN STATUS_CODE_TYPE_REL ON " + _
                    "STATUS_CODES.c_status_code = STATUS_CODE_TYPE_REL.c_status_code " + _
                    "WHERE (((STATUS_CODE_TYPE_REL.c_status_type_code)='" + tStr + "'))" & _
                    " "
                Set cmdSQL = New ADODB.Command
                cmdSQL.ActiveConnection = CurrentProject.Connection
                cmdSQL.CommandType = adCmdText
                '
                cmdSQL.CommandText = "Delete * from ZZ_STATUS_CODE_TMP"
                cmdSQL.Execute tRecDeleted
                '
                cmdSQL.CommandText = tStrSQL
                cmdSQL.Execute tRecDeleted
                '
                '
                ' set the type values
                '
                Set tRstStatusTypes = CurrentDb.OpenRecordset("STATUS_TYPES", dbOpenDynaset)
                '
                tRstStatusTypes.MoveFirst
                tRstStatusTypes.FindFirst "c_status_type_code = " + Chr(34) + cNode.Key + Chr(34)
                TxtTypeID.Value = cNode.Key
                TxtTypeDesc.Value = tRstStatusTypes!c_status_type_desc
                TxtTypeDescChn.Value = tRstStatusTypes!c_status_type_chn
                tRstStatusTypes.Close
                Set tRstStatusTypes = Nothing
            End If
        End If
    End If

```

frmPickStatus_multi - 7

```
                ListStatus.Requery
                For ti = 0 To ListStatus.ListCount
                    ListStatus.Selected(ti) = False
                Next ti
                gSelectCount = 0
            End If
        End If
    End If
End If
End If
End If
End Sub
```

Private Sub NodeSearch()

```
Dim cNode As clsNode, tStr As String, tRstStatusCodes As DAO.Recordset, tRstStatusTypes As DAO.Recordset
Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
Dim tStrLen As String, cmdSQL As ADODB.Command
```

```
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
```

```
txtStatusID.Value = -1
txtStatusDesc.Value = ""
txtStatusDescChn.Value = ""
```

```
' all specific statusation codes will have a type of the form 0101
' hence the ValuePath for the relevant node will be "K000/K01/K0101"
' The command to locate the relevant node is:
' tNode = TreeViewType.FindNode(tStrValuePath)
' TreeViewType.SelectedNode = tNode
```

```
' search for the search string in STATUS_CODES
txtSearchChn.SetFocus
tStrSearchChn = Me.txtSearchChn.Text
txtSearch.SetFocus
tStrSearchEng = Me.txtSearch.Text
cmdFind.SetFocus
```

```
gStrSearch = ""
gUseAlt = False
```

```
If tStrSearchChn <> "" Then
    tStrLen = Str(LenB(tStrSearchChn))
    gStrSearch = "LeftB(c_status_desc_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
    gStrSearchAlt = "InStrB(1,c_status_desc_chn,'" + tStrSearchChn + "') > 0"
ElseIf tStrSearchEng <> "" Then
    tStrLen = Str(Len(tStrSearchEng))
    gStrSearch = "Left(c_status_desc," + tStrLen + ") = '" + tStrSearchEng + "'"
    gStrSearchAlt = "InStr(1,c_status_desc,'" + tStrSearchEng + "') > 0"
End If
```

```
If Not (gStrSearch = "") Then
```

```
    'MsgBox "Looking for status"
```

```
    Set gRstStatusCode = CurrentDb.OpenRecordset("STATUS_CODES", dbOpenDynaset)
```

```
    gRstStatusCode.FindFirst gStrSearch
```

```
If gRstStatusCode.NoMatch Then
    gRstStatusCode.FindFirst gStrSearchAlt
    gUseAlt = True
End If
```

```
If gRstStatusCode.NoMatch Then
    gUseAlt = False
    cmdFindNext.Enabled = False
Else
```

```
    cmdFindNext.Enabled = True
    ' next find the status_type
```

```
    'MsgBox "Looking for status type"
```



```
Set tRstStatusTypes = CurrentDb.OpenRecordset("STATUS_CODE_TYPE_REL", dbOpenDynaset)
```

```
tRstStatusTypes.FindNext "c_status_code = " + Str(gRstStatusCode!c_status_code)
```

```
If Not tRstStatusTypes.NoMatch Then
```

```
    ' set the values
```

```
    TxtStatusID.Value = gRstStatusCode!c_status_code
```

```
    If Not IsNull(gRstStatusCode!c_status_desc) Then
```

```
        TxtStatusDesc.Value = gRstStatusCode!c_status_desc
```

```
    End If
```

```
    If Not IsNull(gRstStatusCode!c_status_desc_chn) Then
```

```
        TxtStatusDescChn.Value = gRstStatusCode!c_status_desc_chn
```

```
    End If
```

```
    '
```

```
    ' define the string
```

```
    '
```

```
    tStr = tRstStatusTypes!c_status_type_code
```

```
    '
```

```
    ' search the tree
```

```
Set cNode = mcTree.Nodes(tStr)
```

```
    '
```

```
    'MsgBox "found node"
```

```
    '
```

```
If Not IsNull(cNode) Then
```

```
    '
```

```
    Set mcTree.ActiveNode = cNode
```

```
    'cNode.Selected = True
```

```
    '
```

```
    ' then one makes it visible
```

```
    '
```

```
    'tNode.EnsureVisible
```

```
    '
```

```
    Set gNode = cNode
```

```
    '
```

```
    ' Finally populate the options and select the record.
```

```
    '
```

```
    CmdSelectAll.Enabled = True
```

```
    '
```

```
    cmdSQL.CommandText = "Delete * from ZZ_STATUS_CODE_TMP"
```

```
    cmdSQL.Execute tRecDeleted
```

```
    '
```

```
    tStrSQL = "INSERT INTO ZZ_STATUS_CODE_TMP ( c_status_code, c_status_desc, c_status_desc_chn )
```

```
        "SELECT STATUS_CODE_TYPE_REL.c_status_code AS c_status_code, STATUS_CODES.c_status_desc,
```

```
        "STATUS_CODES.c_status_desc_chn " +
```

```
        "FROM STATUS_CODES INNER JOIN STATUS_CODE_TYPE_REL ON " +
```

```
        "STATUS_CODES.c_status_code = STATUS_CODE_TYPE_REL.c_status_code " +
```

```
        "WHERE (((STATUS_CODE_TYPE_REL.c_status_type_code)='" + tStr + "'))"
```

```
    cmdSQL.CommandText = tStrSQL
```

```
    cmdSQL.Execute tRecDeleted
```

```
    '
```

```
    ListStatus.Requery
```

```
    For i = 0 To ListStatus.ListCount
```

```
        ListStatus.Selected(i) = False
```

```
    Next i
```

```
    gSelectCount = 0
```

```
    '
```

```
    For i = 0 To ListStatus.ListCount - 1
```

```
        If gRstStatusCode!c_status_code = ListStatus.Column(0, i) Then
```

```
            ListStatus.ListIndex = i
```

```
            ListStatus.Selected(i) = True
```

```
            gSelectCount = gSelectCount + 1
```

```
        End If
```

```
    Next i
```

```
    '
```

```
    ' set the type values
```

```
    '
```

```
Set tRstStatusTypes = CurrentDb.OpenRecordset("STATUS_TYPES", dbOpenDynaset)
```

```
    '
```

```
    tRstStatusTypes.MoveFirst
```

```
    tRstStatusTypes.FindFirst "c_status_type_code = " + Chr(34) + cNode.Key + Chr(34)
```

```
    TxtTypeID.Value = cNode.Key
```

```
    TxtTypeDesc.Value = tRstStatusTypes!c_status_type_desc
```

```
    TxtTypeDescChn.Value = tRstStatusTypes!c_status_type_chn
```

```
    tRstStatusTypes.Close
```

```
Form_frmPickStatus_multi - 9
        Set tRstStatusTypes = Nothing
    End If
End If
End If
End If
End Sub
```

frmPickStatus2 - 1

Option Compare Database

Public gRstStatusCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String

'#####Treeview Code#####

'Add this to your form's declaration section

Public WithEvents mcTree As clsTreeview

Private mbExit As Boolean ' to exit a SpinButton event

'#####Treeview Code#####

Private Sub CmdCancel_Click()

On Error GoTo Err_CmdCancel_Click

Clear_SelectAll

DoCmd.Close

Exit_CmdCancel_Click:

Exit Sub

Err_CmdCancel_Click:

MsgBox Err.Description

Resume Exit_CmdCancel_Click

End Sub

Private Sub CmdSelect_Click()

Clear_SelectAll

CmdSelectAll.SetFocus

CmdSelect.Enabled = False

Forms!frmPickStatus2.Visible = False

End Sub

Private Sub CmdSelectAll_Click()

If CmdSelectAll.Caption = "Select All" Then

CmdSelectAll.Caption = "De-select All"

frmSTATUS_CODES.Form.DatasheetForeColor = RGB(255, 255, 255)

frmSTATUS_CODES.Form.DatasheetBackColor = RGB(0, 0, 255)

CmdSelect.Enabled = True

TxtStatusID.Value = -1

TxtStatusDesc.Value = ""

TxtStatusDescChn.Value = ""

Else

CmdSelectAll.SetFocus

Clear_SelectAll

End If

End Sub

Private Sub Clear_SelectAll()

CmdSelectAll.Caption = "Select All"

,

' reset the form colors

,

frmSTATUS_CODES.Form.DatasheetForeColor = RGB(0, 0, 0)

frmSTATUS_CODES.Form.DatasheetBackColor = RGB(255, 255, 255)

End Sub

Private Sub Form_Open(Cancel As Integer)

Dim cRoot As clsNode, strKey As String, strCaption As String

Dim cNode1 As clsNode, cNode2 As clsNode

Dim tRst As DAO.Recordset

,

' initialize the text fields

,

Me.TxtTypeDesc.Value = ""

Me.TxtTypeDescChn.Value = ""

Me.TxtTypeID.Value = "000"

,

' initialize the Status Codes dataset

,

Set gRstStatusCode = CurrentDb.OpenRecordset("STATUS_CODES", dbOpenDynaset)

Set frmSTATUS_CODES.Form.Recordset = gRstStatusCode

,

'frmSTATUS_CODES.Form.OrderBy = "c_sortorder"

'frmSTATUS_CODES.Form.OrderByOn = True

If Not IsNull(Me.OpenArgs) Then

Dim strStatus As String

strStatus = Me.OpenArgs

Dim rsStatus As DAO.Recordset

Set rsStatus = frmSTATUS_CODES.Form.Recordset

rsStatus.FindFirst "c_status_code = " & strStatus

End If

,

```

' build treeview
'
Set tRst = CurrentDb.OpenRecordset("STATUS_TYPES", dbOpenDynaset)

' set the language
Dim tmli As MsoLanguageID
'
tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
'
If tmli = msoLanguageIDSimplifiedChinese Then
    gDisplayLanguage = "S"
ElseIf tmli = msoLanguageIDTraditionalChinese Then
    gDisplayLanguage = "T"
Else
    gDisplayLanguage = "E"
End If
'
'
'MsgBox "About to build tree"
tRst.MoveFirst
Set mcTree = Me.subTreeView.Form.pTreeview
With mcTree
    .NodesClear
    .AppName = AppName ' Title for message boxes:
    ' use the appropriate caption
    If gDisplayLanguage = "T" Then
        strCaption = ChrW(31038) + ChrW(26371) + ChrW(38364) + ChrW(20418) + ChrW(20998) + ChrW(39006)
    ElseIf gdisplaylanguage = "S" Then
        strCaption = ChrW(31038) + ChrW(20250) + ChrW(20851) + ChrW(31995) + ChrW(20998) + ChrW(31612)
    Else
        strCaption = "Categories of Social Status"
    End If

    Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
    ' Add a Root node with main and expanded icons and make it bold
    cRoot.Bold = True
    ' Loop through the records
    Do While Not tRst.EOF
        ' Add node
        strKey = tRst!c_status_type_code
        If gDisplayLanguage = "E" Then
            strCaption = tRst!c_status_type_desc
        Else
            strCaption = tRst!c_status_type_chn
        End If
        If Len(tRst!c_status_type_code) = 2 Then
            Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode1.Expanded = False
        Else
            Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
            cNode2.Expanded = False
        End If
        tRst.MoveNext
    Loop
    ' Create the node controls and display the tree
    .Refresh
End With
'
Set tRst = Nothing
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

'Dim StrSearch As String
'Me.TxtSearch.SetFocus
'StrSearch = Me.TxtSearch.Value
'If StrSearch <> "" Then
'Dim rsStatusCodes As DAO.Recordset
'Set rsStatusCodes = frmSTATUS_CODES.Form.Recordset
'Dim StrSearchStr As String
'StrSearchStr = "c_status_desc_chn = " + Chr(34) + StrSearch + Chr(34)
'rsStatusCodes.FindFirst StrSearchStr
'End If

Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:

```

```
MsgBox Err.Description
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub mcTree_Click(cNode As clsNode)
    Dim tRst As DAO.Recordset, tRstStatus As DAO.Recordset, tStrSQL As String
    Dim tStatusCodeQuery As DAO.QueryDef, prm As DAO.Parameter
    Dim tRstStatusCode As DAO.Recordset, tRstDummy As DAO.Recordset

    Set cmdSQL = New ADODB.Command

    TxtStatusID.Value = -1
    TxtStatusDesc.Value = ""
    TxtStatusDescChn.Value = ""
    CmdSelect.Enabled = False

    ' reset the form colors
    '
    Clear_SelectAll
    '
    If cNode.Key = "Root" Then
        TxtTypeID.Value = ""
        TxtTypeDesc.Value = ""
        TxtTypeDescChn.Value = ""
        CmdSelectAll.Enabled = False
        '
        ' reset the entry code choices
        '
        Set gRstStatusCode = CurrentDb.OpenRecordset("STATUS_CODES", dbOpenDynaset)
        '
        Set frmSTATUS_CODES.Form.Recordset = gRstStatusCode
        frmSTATUS_CODES.Form.Refresh
    Else
        Set tRst = CurrentDb.OpenRecordset("STATUS_TYPES", dbOpenDynaset)
        '
        tRst.MoveFirst
        tRst.FindFirst "c_status_type_code = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key
        TxtTypeDesc.Value = tRst!c_status_type_desc
        TxtTypeDescChn.Value = tRst!c_status_type_chn
        tRst.Close
        Set tRst = Nothing
        '
        CmdSelectAll.Enabled = True
        '
        ' we need to distinguish between type / subtype
        '
        tStrSQL = "INSERT INTO ZZ_STATUS_CODE ( c_status_code, c_status_desc, c_status_desc_chn ) " + _
            "SELECT STATUS_CODE_TYPE_REL.c_status_code AS c_status_code, STATUS_CODES.c_status_desc, " + _
            "STATUS_CODES.c_status_desc_chn " + _
            "FROM STATUS_CODES INNER JOIN STATUS_CODE_TYPE_REL ON " + _
            "STATUS_CODES.c_status_code = STATUS_CODE_TYPE_REL.c_status_code "

        If Len(cNode.Key) = 2 Then
            tStrSQL = tStrSQL + "WHERE (((Left([STATUS_CODE_TYPE_REL].[c_status_type_code]),2))='" + _
                TxtTypeID.Value + "'))"
        Else
            tStrSQL = tStrSQL + "WHERE (((STATUS_CODE_TYPE_REL.c_status_type_code)='" + _
                TxtTypeID.Value + "'))"
        End If

        Set tRstStatusCode = frmSTATUS_CODES.Form.Recordset
        Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SC", dbOpenDynaset)
        Set frmSTATUS_CODES.Form.Recordset = tRstDummy

        tRstStatusCode.Close
        '
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        cmdSQL.CommandText = "Delete * from ZZ_STATUS_CODE"
        cmdSQL.Execute tRecDeleted
        '
        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
        ' to get rid of superfluous deleted records, briefly close the table
        '
        Set tRstStatusCode = CurrentDb.OpenRecordset("ZZ_STATUS_CODE", dbOpenDynaset)
```

```

        ,
        Set frmSTATUS_CODES.Form.Recordset = tRstStatusCode
        Set tRstDummy = Nothing
        ,
    End If

End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Or IsNull(TxtSearch.Text) Then
        If TxtSearchChn.Value = "" Or IsNull(TxtSearchChn.Value) Then
            CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Or IsNull(TxtSearchChn.Text) Then
        If TxtSearch.Value = "" Or IsNull(TxtSearch.Value) Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtSearch.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub CmdFindNext_Click()
    Dim tRstStatusCodes As DAO.Recordset, tRstStatusTypes As DAO.Recordset, cNode As clsNode
    Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

    TxtStatusID.Value = -1
    TxtStatusDesc.Value = ""
    TxtStatusDescChn.Value = ""

    If IsNull(gRstStatusCode) Then
        MsgBox "Error in search: table closed."
    Else
        If gStrSearch = "" Then
            MsgBox "Error in search: string empty."
        Else
            'MsgBox "Looking for entry"

            If gRstStatusCode.EOF Then
                CmdFindNext.Enabled = False
            Else
                ' gRstStatusCode.MoveNext

                If gUseAlt Then
                    gRstStatusCode.FindNext gStrSearchAlt
                Else
                    gRstStatusCode.FindNext gStrSearch

                    If gRstStatusCode.NoMatch Then
                        gRstStatusCode.FindFirst gStrSearchAlt
                        gUseAlt = True
                    End If
                End If
            End If

            If gRstStatusCode.NoMatch Then
                If gUseAlt Then
                    gUseAlt = False
                End If
                CmdFindNext.Enabled = False
            Else
                ,
                ' next find the entry_type
                ,
                'MsgBox "Looking for entry type"

                Set tRstStatusTypes = CurrentDb.OpenRecordset("STATUS_CODE_TYPE_REL", dbOpenDynaset)

                tRstStatusTypes.FindNext "c_status_code = " + Str(gRstStatusCode!c_status_code)

                If Not tRstStatusTypes.NoMatch Then

```

```

'
'   set the values
'
TxtStatusID.Value = gRstStatusCode!c_status_code
If Not IsNull(gRstStatusCode!c_status_desc) Then
    TxtStatusDesc.Value = gRstStatusCode!c_status_desc
End If
If Not IsNull(gRstStatusCode!c_status_desc_chn) Then
    TxtStatusDescChn.Value = gRstStatusCode!c_status_desc_chn
End If
'
'   define the string
'
tStr = tRstStatusTypes!c_status_type_code
'
'   search the tree

Set cNode = mcTree.Nodes(tStr)
'
'   MsgBox "found node"
'
If Not IsNull(cNode) Then
    If cNode.Key = gNode.Key Then
        Set tRstStatusCodes = frmSTATUS_CODES.Form.Recordset
        tRstStatusCodes.FindNext "c_status_code = " + Str(gRstStatusCode!c_status_code)
        frmSTATUS_CODES.Form.Refresh
    Else
        Set mcTree.ActiveNode = cNode
        'cNode.Selected = True
        '
        '   then one makes it visible
        '
        'tNode.EnsureVisible
        '
        Set gNode = cNode
        '   Finally populate the options and select the record.
        '
        CmdSelectAll.Enabled = True
        '
        tStrSQL = "INSERT INTO ZZ_STATUS_CODE ( c_status_code, c_status_desc, c_status_de
sc_chn ) " + _
        "SELECT STATUS_CODE_TYPE_REL.c_status_code AS c_status_code, STATUS_CODES.c_s
tatus_desc, " + _
        "STATUS_CODES.c_status_desc_chn " + _
        "FROM STATUS_CODES INNER JOIN STATUS_CODE_TYPE_REL ON " + _
        "STATUS_CODES.c_status_code = STATUS_CODE_TYPE_REL.c_status_code " + _
        "WHERE (((STATUS_CODE_TYPE_REL.c_status_type_code)='" + tStr + "'))" -

        Set tRstStatusCodes = frmSTATUS_CODES.Form.Recordset
        Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_AC", dbOpenDynaset)
        Set frmSTATUS_CODES.Form.Recordset = tRstDummy

        tRstStatusCodes.Close
        '
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        cmdSQL.CommandText = "Delete * from ZZ_STATUS_CODE"
        cmdSQL.Execute tRecDeleted
        '
        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
        Set tRstStatusCodes = CurrentDb.OpenRecordset("ZZ_STATUS_CODE", dbOpenDynaset)
        '
        Set frmSTATUS_CODES.Form.Recordset = tRstStatusCodes
        tRstStatusCodes.FindNext "c_status_code = " + Str(gRstStatusCode!c_status_code)
        frmSTATUS_CODES.Form.Refresh
        '
        Set tRstDummy = Nothing
        '
        '   set the type values
        '
        Set tRstStatusTypes = CurrentDb.OpenRecordset("STATUS_TYPES", dbOpenDynaset)
        '
        tRstStatusTypes.MoveFirst
        tRstStatusTypes.FindFirst "c_status_type_code = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key

```

```

        TxtTypeDesc.Value = tRstStatusTypes!c_status_type_desc
        TxtTypeDescChn.Value = tRstStatusTypes!c_status_type_chn
        tRstStatusTypes.Close
        Set tRstStatusTypes = Nothing
    End If
End If
End If
End If
End If
End If
End Sub

```

```
Private Sub NodeSearch()
```

```

    Dim cNode As clsNode, tStr As String, tRstStatusCodes As DAO.Recordset, tRstStatusTypes As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrLen As String, cmdSQL As ADODB.Command

```

```
    Set cmdSQL = New ADODB.Command
```

```

    TxtStatusID.Value = -1
    TxtStatusDesc.Value = ""
    TxtStatusDescChn.Value = ""

```

```

' all specific statusiation codes will have a type of the form 0101
' hence the ValuePath for the relevant node will be "K000/K01/K0101"
' The command to locate the relevant node is:
' tNode = TreeViewType.FindNode(tStrValuePath)
' TreeViewType.SelectedNode = tNode

```

```

' search for the search string in STATUS_CODES
TxtSearchChn.SetFocus
tStrSearchChn = Me.TxtSearchChn.Text
TxtSearch.SetFocus
tStrSearchEng = Me.TxtSearch.Text
CmdFind.SetFocus

```

```

gStrSearch = ""
gUseAlt = False

```

```

If tStrSearchChn <> "" Then
    tStrLen = Str(LenB(tStrSearchChn))
    gStrSearch = "LeftB(c_status_desc_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
    gStrSearchAlt = "InStrB(1,c_status_desc_chn,'" + tStrSearchChn + "') > 0"
ElseIf tStrSearchEng <> "" Then
    tStrLen = Str(Len(tStrSearchEng))
    gStrSearch = "Left(c_status_desc," + tStrLen + ") = '" + tStrSearchEng + "'"
    gStrSearchAlt = "InStr(1,c_status_desc,'" + tStrSearchEng + "') > 0"
End If

```

```
If Not (gStrSearch = "") Then
```

```
    'MsgBox "Looking for status"
```

```
    Set gRstStatusCode = CurrentDb.OpenRecordset("STATUS_CODES", dbOpenDynaset)
```

```
    gRstStatusCode.FindFirst gStrSearch
```

```

    If gRstStatusCode.NoMatch Then
        gRstStatusCode.FindFirst gStrSearchAlt
        gUseAlt = True
    End If

```

```

    If gRstStatusCode.NoMatch Then
        gUseAlt = False
        CmdFindNext.Enabled = False
    Else

```

```

        ,
        CmdFindNext.Enabled = True
        ,
        ' next find the status_type
        ,

```

```
    'MsgBox "Looking for status type"
```

```
    Set tRstStatusTypes = CurrentDb.OpenRecordset("STATUS_CODE_TYPE_REL", dbOpenDynaset)
```

```
    tRstStatusTypes.FindNext "c_status_code = " + Str(gRstStatusCode!c_status_code)
```



```

If Not tRstStatusTypes.NoMatch Then
    '
    ' set the values
    '
    TxtStatusID.Value = gRstStatusCode!c_status_code
    If Not IsNull(gRstStatusCode!c_status_desc) Then
        TxtStatusDesc.Value = gRstStatusCode!c_status_desc
    End If
    If Not IsNull(gRstStatusCode!c_status_desc_chn) Then
        TxtStatusDescChn.Value = gRstStatusCode!c_status_desc_chn
    End If
    '
    ' define the string
    '
    tStr = tRstStatusTypes!c_status_type_code
    '
    ' search the tree

Set cNode = mcTree.Nodes(tStr)
'
'MsgBox "found node"
'
If Not IsNull(cNode) Then
    '
    Set mcTree.ActiveNode = cNode
    'cNode.Selected = True
    '
    ' then one makes it visible
    '
    'tNode.EnsureVisible
    '
    Set gNode = cNode
    '
    ' Finally populate the options and select the record.
    '
    CmdSelectAll.Enabled = True
    '
    tStrSQL = "INSERT INTO ZZ_STATUS_CODE ( c_status_code, c_status_desc, c_status_desc_chn ) " + _
        "SELECT STATUS_CODE_TYPE_REL.c_status_code AS c_status_code, STATUS_CODES.c_status_desc, " + _
        "STATUS_CODES.c_status_desc_chn " + _
        "FROM STATUS_CODES INNER JOIN STATUS_CODE_TYPE_REL ON " + _
        "STATUS_CODES.c_status_code = STATUS_CODE_TYPE_REL.c_status_code " + _
        "WHERE (((STATUS_CODE_TYPE_REL.c_status_type_code)='" + tStr + "'))" -

    Set tRstStatusCodes = frmSTATUS_CODES.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SC", dbOpenDynaset)
    Set frmSTATUS_CODES.Form.Recordset = tRstDummy

    tRstStatusCodes.Close
    '
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    cmdSQL.CommandText = "Delete * from ZZ_STATUS_CODE"
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    '
    Set tRstStatusCodes = CurrentDb.OpenRecordset("ZZ_STATUS_CODE", dbOpenDynaset)
    '
    Set frmSTATUS_CODES.Form.Recordset = tRstStatusCodes
    tRstStatusCodes.FindNext "c_status_code = " + Str(gRstStatusCode!c_status_code)
    frmSTATUS_CODES.Form.Refresh
    '
    Set tRstDummy = Nothing
    '
    ' set the type values
    '
    Set tRstStatusTypes = CurrentDb.OpenRecordset("STATUS_TYPES", dbOpenDynaset)
    '
    tRstStatusTypes.MoveFirst
    tRstStatusTypes.FindFirst "c_status_type_code = " + Chr(34) + cNode.Key + Chr(34)
    TxtTypeID.Value = cNode.Key
    TxtTypeDesc.Value = tRstStatusTypes!c_status_type_desc
    TxtTypeDescChn.Value = tRstStatusTypes!c_status_type_chn
    tRstStatusTypes.Close

```

Form_frmPickStatus2 - 8

```
                Set tRstStatusTypes = Nothing
            End If
        End If
    End If
End If

End Sub
```

Option Compare Database

```
Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
Exit Sub
```

```
Err_CmdCancel_Click:
MsgBox Err.Description
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()
Forms!frmPickTEXT_CAT.Visible = False
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
If Not IsNull(Me.OpenArgs) Then
Dim strTextCat As String
strTextCat = Me.OpenArgs
Dim rsTextCat As DAO.Recordset
Set rsTextCat = frmTEXT_BIBLCAT.Form.Recordset
rsTextCat.FindFirst "c_text_cat_code = " & strTextCat
End If
End Sub
```

```
Private Sub TxtSearch_Change()
If Me.TxtSearch.Text = "" Then
Me.CmdFind.Enabled = False
Else
Me.CmdFind.Enabled = True
End If
End Sub
```

```
Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click
```

```
Dim StrSearch As String
Me.TxtSearch.SetFocus
StrSearch = Me.TxtSearch.Value
If StrSearch <> "" Then
Dim rsTextCat As DAO.Recordset
Set rsTextCat = frmTEXT_CAT.Form.Recordset
Dim StrSearchStr As String
StrSearchStr = "c_text_cat_desc_chn = " & Chr(34) & StrSearch & Chr(34)
rsTextCat.FindFirst StrSearchStr
End If
```

```
Exit_CmdFind_Click:
Exit Sub
```

```
Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click
```

```
End Sub
```

Option Compare Database

```
Public gRstTextCatCode As DAO.Recordset, gNode As clsNode, gStrSearch As String, gStrSearchAlt As String
Public gUseAlt As Boolean, gDisplayLanguage As String, gSelectCount As Integer
```

```
'#####Treeview Code#####
```

```
'Add this to your form's declaration section
```

```
Public WithEvents mcTree As clsTreeview
```

```
Private mbExit As Boolean ' to exit a SpinButton event
```

```
'#####Treeview Code#####
```

```
Private Sub CmdCancel_Click()
```

```
On Error GoTo Err_CmdCancel_Click
```

```
Clear_SelectAll
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
```

```
Exit Sub
```

```
Err_CmdCancel_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()
```

```
Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, varItm As Variant, ti As Integer
```

```
CmdSelectAll.SetFocus
```

```
CmdSelect.Enabled = False
```

```
gSelectCount = 0
```

```
For Each varItm In ListTextCat.ItemsSelected
```

```
gSelectCount = gSelectCount + 1
```

```
Next varItm
```

```
If gSelectCount = 0 Then
```

```
Me.TxtTextCatDesc.Value = ""
```

```
Me.TxtTextCatDescChn.Value = ""
```

```
Me.TxtTextCatID.Value = 0
```

```
Me.CmdSelect.Enabled = False
```

```
Else
```

```
If gSelectCount = 1 Then
```

```
' this means that there is only on selected item
```

```
For Each varItm In ListTextCat.ItemsSelected
```

```
Me.TxtTextCatDesc.Value = ListTextCat.Column(1, varItm)
```

```
Me.TxtTextCatDescChn.Value = ListTextCat.Column(2, varItm)
```

```
Me.TxtTextCatID.Value = ListTextCat.Column(0, varItm)
```

```
'MsgBox ListTextCat.Column(1, varItm)
```

```
Next varItm
```

```
ElseIf gSelectCount = ListTextCat.ListCount - 1 Then
```

```
Me.TxtTextCatDesc.Value = "All"
```

```
Me.TxtTextCatDescChn.Value = "All"
```

```
Me.TxtTextCatID.Value = -1
```

```
Else
```

```
Me.TxtTextCatDesc.Value = "Multi-select"
```

```
Me.TxtTextCatDescChn.Value = "Multi-select"
```

```
Me.TxtTextCatID.Value = -2
```

```
'MsgBox "Multi-select"
```

```
End If
```

```
Me.CmdSelect.Enabled = True
```

```
Set cmdSQL = New ADODB.Command
```

```
cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
cmdSQL.CommandType = adCmdText
```

```
cmdSQL.CommandText = "DELETE * FROM ZZ_TEXT_BIBLCAT_CODES"
```

```
cmdSQL.Execute tRecCount
```

```
' first copy the records over to the table
```

```
Set tRst = CurrentDb.OpenRecordset("ZZ_TEXT_BIBLCAT_CODES", dbOpenDynaset)
```

```
For Each varItm In ListTextCat.ItemsSelected
```

```
tRst.AddNew
```

```
tRst!c_text_cat_code = ListTextCat.Column(0, varItm)
```

```
tRst!c_text_cat_desc = ListTextCat.Column(1, varItm)
```

```
tRst!c_text_cat_desc_chn = ListTextCat.Column(2, varItm)
```

```
tRst.Update
```

```
Next varItm
```

```
tRst.Close
```

```

        ListTextCat.Requery
        'For ti = 0 To ListTextCat.ListCount
        '    ListTextCat.Selected(ti) = False
        'Next ti
        gSelectCount = 0

    End If
    Forms!frmPickTextCat_multi.Visible = False
End Sub

Private Sub CmdSelectAll_Click()
    If CmdSelectAll.Caption = "Select All" Then
        CmdSelectAll.Caption = "De-select All"

        For ti = 0 To ListTextCat.ListCount
            ListTextCat.Selected(ti) = True
        Next ti

        CmdSelect.Enabled = True
        TxtTextCatID.Value = -1
        TxtTextCatDesc.Value = ""
        TxtTextCatDescChn.Value = ""
    Else
        CmdSelectAll.SetFocus
        Clear_SelectAll
    End If
End Sub

Private Sub Clear_SelectAll()
    CmdSelectAll.Caption = "Select All"
    '
    ' reset the form colors
    '
    For ti = 0 To ListTextCat.ListCount
        ListTextCat.Selected(ti) = False
    Next ti

    gSelectCount = 0
    CmdSelect.Enabled = False
End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim cRoot As clsNode, strKey As String, strCaption As String
    Dim cNode1 As clsNode, cNode2 As clsNode, cNode3 As clsNode
    Dim tRst As DAO.Recordset, cmdSQL As ADODB.Command
    '
    ' initialize the text fields
    '
    Me.TxtTypeDesc.Value = ""
    Me.TxtTypeDescChn.Value = ""
    Me.TxtTypeID.Value = "000"
    gSelectCount = 0
    '
    ' initialize the TextCat Codes dataset
    '
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    Set gRstTextCatCode = CurrentDb.OpenRecordset("TEXT_BIBLCAT_CODES", dbOpenDynaset)

    cmdSQL.CommandText = "DELETE * FROM ZZ_TEXT_BIBLCAT_CODES_TMP"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "INSERT INTO ZZ_TEXT_BIBLCAT_CODES_TMP ( c_text_cat_code, c_text_cat_desc, c_text_cat_desc_chn ) " + _
        "SELECT TEXT_BIBLCAT_CODES.c_text_cat_code, TEXT_BIBLCAT_CODES.c_text_cat_desc, TEXT_BIBLCAT_CODES.c_text_cat_desc_chn " + _
        "FROM TEXT_BIBLCAT_CODES"
    cmdSQL.Execute tRecCount

    ListTextCat.Requery
    'For ti = 0 To ListTextCat.ListCount
    '    ListTextCat.Selected(ti) = False
    'Next ti
    '
    'frmTEXT_BIBLCAT_CODES.Form.OrderBy = "c_sortorder"
    'frmTEXT_BIBLCAT_CODES.Form.OrderByOn = True

    If Not IsNull(Me.OpenArgs) Then
        Dim strTextCat As String

```

```

        strTextCat = Me.OpenArgs
    End If
    '
    ' build treeview
    '
    Set tRst = CurrentDb.OpenRecordset("TEXT_BIBLCAT_TYPES", dbOpenDynaset)

    ' set the language
    Dim tmli As MsoLanguageID
    '
    tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    '
    If tmli = msoLanguageIDSimplifiedChinese Then
        gDisplayLanguage = "S"
    ElseIf tmli = msoLanguageIDTraditionalChinese Then
        gDisplayLanguage = "T"
    Else
        gDisplayLanguage = "E"
    End If
    '
    '
    'MsgBox "About to build tree"
    tRst.MoveFirst
    Set mcTree = Me.subTreeView.Form.pTreeview
    With mcTree
        .NodesClear
        .AppName = AppName ' Title for message boxes:
        ' use the appropriate caption
        If gDisplayLanguage = "T" Then
            strCaption = ChrW(31038) + ChrW(26371) + ChrW(38364) + ChrW(20418) + ChrW(20998) + ChrW(39006)
        ElseIf gdisplaylangauge = "S" Then
            strCaption = ChrW(31038) + ChrW(20250) + ChrW(20851) + ChrW(31995) + ChrW(20998) + ChrW(31612)
        Else
            strCaption = "Categories of Text"
        End If

        Set cRoot = .AddRoot("Root", strCaption, "FolderClosed", "FolderOpen")
        ' Add a Root node with main and expanded icons and make it bold
        cRoot.Bold = True
        ' Loop through the records
        Do While Not tRst.EOF
            ' Add node
            strKey = tRst!c_text_cat_type_id
            If gDisplayLanguage = "E" Then
                strCaption = tRst!c_text_cat_type_desc
            Else
                strCaption = tRst!c_text_cat_type_desc_chn
            End If
            Select Case Len(strKey)
                Case 2
                    Set cNode1 = cRoot.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode1.Expanded = False
                Case 5
                    Set cNode2 = cNode1.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode2.Expanded = False
                Case 7
                    Set cNode3 = cNode2.AddChild(sKey:=strKey, vCaption:=strCaption)
                    cNode3.Expanded = False
            End Select
            tRst.MoveNext
        Loop
        ' Create the node controls and display the tree
        .Refresh
    End With
    '
    Set tRst = Nothing
End Sub

Private Sub CmdFind_Click()
On Error GoTo Err_CmdFind_Click

    Call NodeSearch

Exit_CmdFind_Click:
Exit Sub

Err_CmdFind_Click:
MsgBox Err.Description
Resume Exit_CmdFind_Click

End Sub

```

```

Private Sub ListTextCat_Click()
    Dim ti As Long, tUnClicked As Boolean
    Dim varItm As Variant

    gSelectCount = 0

    For Each varItm In ListTextCat.ItemsSelected
        gSelectCount = gSelectCount + 1
    Next varItm

    If gSelectCount = 0 Then
        Me.CmdSelect.Enabled = False
    Else
        Me.CmdSelect.Enabled = True
    End If
    'MsgBox ListTextCat.Column(1, ti + 1) + ": Select Count = " + Str(gSelectCount)

    If gSelectCount = 0 Then
        Me.TxtTextCatDesc.Value = ""
        Me.TxtTextCatDescChn.Value = ""
        Me.TxtTextCatID.Value = 0
        Me.CmdSelect.Enabled = False
    Else
        If gSelectCount = 1 Then
            If tUnClicked Then
                ' this means that there is only on selected item, but it is NOT this item
                ' we therefore need to locate the selected item and put its values into the text boxes
                ' I may not even use these boxes anymore, but just in case...
                For Each varItm In ListTextCat.ItemsSelected
                    Me.TxtTextCatDesc.Value = ListTextCat.Column(1, varItm)
                    Me.TxtTextCatDescChn.Value = ListTextCat.Column(2, varItm)
                    Me.TxtTextCatID.Value = ListTextCat.Column(0, varItm)
                    'MsgBox ListTextCat.Column(1, varItm)
                Next varItm
            Else
                Me.TxtTextCatDesc.Value = ListTextCat.Column(1, ti + 1)
                Me.TxtTextCatDescChn.Value = ListTextCat.Column(2, ti + 1)
                Me.TxtTextCatID.Value = ListTextCat.Column(0, ti + 1)
                'MsgBox ListTextCat.Column(1, ti + 1)
            End If
        Else
            Me.TxtTextCatDesc.Value = "Multi-select"
            Me.TxtTextCatDescChn.Value = "Multi-select"
            Me.TxtTextCatID.Value = -2
            'MsgBox "Multi-select"
        End If
        Me.CmdSelect.Enabled = True
    End If
End Sub

```

```

Private Sub mcTree_Click(cNode As clsNode)
    Dim tRst As DAO.Recordset, tRstTextCat As DAO.Recordset, tStrSQL As String
    Dim tTextCatCodeQuery As DAO.QueryDef, prm As DAO.Parameter
    Dim tRstTextCatCode As DAO.Recordset, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    '
    TxtTextCatID.Value = -1
    TxtTextCatDesc.Value = ""
    TxtTextCatDescChn.Value = ""
    CmdSelect.Enabled = False

    ' reset the form colors
    ,
    Clear_SelectAll
    ,
    If cNode.Key = "Root" Then
        TxtTypeID.Value = ""
        TxtTypeDesc.Value = ""
        TxtTypeDescChn.Value = ""
        CmdSelectAll.Enabled = False
        ,
        ' reset the text category code choices
        ,
    End If

```

```

        Set gRstTextCatCode = CurrentDb.OpenRecordset("TEXT_BIBLCAT_CODES", dbOpenDynaset)
        '
        cmdSQL.CommandText = "Delete * from ZZ_TEXT_BIBLCAT_CODES_TMP"
        cmdSQL.Execute tRecDeleted
        '
        cmdSQL.CommandText = "INSERT INTO ZZ_TEXT_BIBLCAT_CODES_TMP ( c_text_cat_code, c_text_cat_desc, c_text_cat_desc_chn ) " + _
            "SELECT TEXT_BIBLCAT_CODES.c_text_cat_code, TEXT_BIBLCAT_CODES.c_text_cat_desc, TEXT_BIBLCAT_CODES.c_text_cat_desc_chn " + _
            "FROM TEXT_BIBLCAT_CODES"
        cmdSQL.Execute tRecCount

    Else
        Set tRst = CurrentDb.OpenRecordset("TEXT_BIBLCAT_TYPES", dbOpenDynaset)
        '
        tRst.MoveFirst
        tRst.FindFirst "c_text_cat_type_id = " + Chr(34) + cNode.Key + Chr(34)
        TxtTypeID.Value = cNode.Key
        TxtTypeDesc.Value = tRst!c_text_cat_type_desc
        TxtTypeDescChn.Value = tRst!c_text_cat_type_desc_chn
        tRst.Close
        Set tRst = Nothing
        '
        CmdSelectAll.Enabled = True
        '
        ' we need to distinguish between type / subtype
        '
        tStrSQL = "INSERT INTO ZZ_TEXT_BIBLCAT_CODES_TMP ( c_text_cat_code, c_text_cat_desc, c_text_cat_desc_chn ) " + _
            "SELECT TEXT_BIBLCAT_CODES.c_text_cat_code, TEXT_BIBLCAT_CODES.c_text_cat_desc, TEXT_BIBLCAT_CODES.c_text_cat_desc_chn " + _
            "FROM TEXT_BIBLCAT_CODES INNER JOIN TEXT_BIBLCAT_CODE_TYPE_REL ON " + _
            "(TEXT_BIBLCAT_CODES.c_text_cat_code = TEXT_BIBLCAT_CODE_TYPE_REL.c_text_cat_code) "

        Select Case Len(cNode.Key)
            Case 2
                tStrSQL = tStrSQL + "WHERE (((Left([TEXT_BIBLCAT_CODE_TYPE_REL].[c_text_cat_type_id]),2))='" + _
                    TxtTypeID.Value + "'))"
            Case 5
                tStrSQL = tStrSQL + "WHERE (((Left([TEXT_BIBLCAT_CODE_TYPE_REL].[c_text_cat_type_id]),5))='" + _
                    TxtTypeID.Value + "'))"
            Case 7
                tStrSQL = tStrSQL + "WHERE (((Left([TEXT_BIBLCAT_CODE_TYPE_REL].[c_text_cat_type_id]),7))='" + _
                    TxtTypeID.Value + "'))"
        End Select

        cmdSQL.CommandText = "Delete * from ZZ_TEXT_BIBLCAT_CODES_TMP"
        cmdSQL.Execute tRecDeleted
        '
        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted
        '
    End If

    ListTextCat.Requery
    For ti = 0 To ListTextCat.ListCount
        ListTextCat.Selected(ti) = False
    Next ti
    gSelectCount = 0
End Sub

Private Sub TxtSearch_Change()
    If TxtSearch.Text = "" Or IsNull(TxtSearch.Text) Then
        If TxtSearchChn.Value = "" Or IsNull(TxtSearchChn.Value) Then
            CmdFind.Enabled = False
        End If
    Else
        TxtSearchChn.Value = ""
        CmdFind.Enabled = True
    End If
    CmdFindNext.Enabled = False
End Sub

Private Sub TxtSearchChn_Change()
    If TxtSearchChn.Text = "" Or IsNull(TxtSearchChn.Text) Then
        If TxtSearch.Value = "" Or IsNull(TxtSearch.Value) Then
            Me.CmdFind.Enabled = False
        End If
    End If
End Sub

```



```

Else
    TxtSearch.Value = ""
    CmdFind.Enabled = True
End If
CmdFindNext.Enabled = False
End Sub

Private Sub CmdFindNext_Click()
    Dim tRstTextCatCodes As DAO.Recordset, tRstTextCatTypes As DAO.Recordset, cNode As clsNode
    Dim tStrSQL As String, tRstDummy As DAO.Recordset, cmdSQL As ADODB.Command

    TxtTextCatID.Value = -1
    TxtTextCatDesc.Value = ""
    TxtTextCatDescChn.Value = ""

    If IsNull(gRstTextCatCode) Then
        MsgBox "Error in search: table closed."
    Else
        If gStrSearch = "" Then
            MsgBox "Error in search: string empty."
        Else
            'MsgBox "Looking for entry"

            If gRstTextCatCode.EOF Then
                CmdFindNext.Enabled = False
            Else
                ' gRstTextCatCode.MoveNext

                If gUseAlt Then
                    gRstTextCatCode.FindNext gStrSearchAlt
                Else
                    gRstTextCatCode.FindNext gStrSearch

                    If gRstTextCatCode.NoMatch Then
                        gRstTextCatCode.FindFirst gStrSearchAlt
                        gUseAlt = True
                    End If
                End If
            End If

            If gRstTextCatCode.NoMatch Then
                If gUseAlt Then
                    gUseAlt = False
                End If
                CmdFindNext.Enabled = False
            Else
                '
                ' next find the text type
                '
                'MsgBox "Looking for text type"

                Set tRstTextCatTypes = CurrentDb.OpenRecordset("TEXT_BIBLCAT_CODE_TYPE_REL", dbOpenDynaset)
                tRstTextCatTypes.FindNext "c_text_cat_code = " + Str(gRstTextCatCode!c_text_cat_code)

                If Not tRstTextCatTypes.NoMatch Then
                    '
                    ' set the values
                    '
                    TxtTextCatID.Value = gRstTextCatCode!c_text_cat_code
                    If Not IsNull(gRstTextCatCode!c_text_cat_desc) Then
                        TxtTextCatDesc.Value = gRstTextCatCode!c_text_cat_desc
                    End If
                    If Not IsNull(gRstTextCatCode!c_text_cat_desc_chn) Then
                        TxtTextCatDescChn.Value = gRstTextCatCode!c_text_cat_desc_chn
                    End If
                    '
                    ' define the string
                    '
                    tStr = tRstTextCatTypes!c_text_cat_type_id
                    '
                    ' search the tree

                    Set cNode = mcTree.Nodes(tStr)
                    '
                    'MsgBox "found node"
                    '
                    If Not IsNull(cNode) Then
                        If cNode.Key = gNode.Key Then
                            gSelectCount = 0
                            For i = 0 To ListTextCat.ListCount - 1

```

```

        If gRstTextCatCode!c_text_cat_code = ListTextCat.Column(0, i) Then
            ListTextCat.ListIndex = i
            ListTextCat.Selected(i) = True
            gSelectCount = gSelectCount + 1
        End If
    Next i
Else
    Set mcTree.ActiveNode = cNode
    'cNode.Selected = True
    '
    '    then one makes it visible
    '
    'tNode.EnsureVisible
    '
    Set gNode = cNode
    '
    '    Finally populate the options and select the record.
    '
    CmdSelectAll.Enabled = True
    '
    tStrSQL = "INSERT INTO ZZ_TEXT_BIBLCAT_CODES_TMP ( c_text_cat_code, c_text_cat_desc, c_text_cat_desc_chn ) " + _
        "SELECT TEXT_BIBLCAT_CODES.c_text_cat_code, TEXT_BIBLCAT_CODES.c_text_cat_desc, TEXT_BIBLCAT_CODES.c_text_cat_desc_chn " + _
        "FROM TEXT_BIBLCAT_CODES INNER JOIN TEXT_BIBLCAT_CODE_TYPE_REL ON " + _
        "(TEXT_BIBLCAT_CODES.c_text_cat_code = TEXT_BIBLCAT_CODE_TYPE_REL.c_text_cat_code) " + _
        "WHERE (((TEXT_BIBLCAT_CODE_TYPE_REL.c_text_cat_type_id)='" + tStr + "'))"
    '
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    cmdSQL.CommandText = "Delete * from ZZ_TEXT_BIBLCAT_CODES_TMP"
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    '
    ListTextCat.Requery
    For ti = 0 To ListTextCat.ListCount
        ListTextCat.Selected(ti) = False
    Next ti
    gSelectCount = 0
    '
    '    set the type values
    '
    Set tRstTextCatTypes = CurrentDb.OpenRecordset("TEXT_BIBLCAT_TYPES", dbOpenDynase
    '
    tRstTextCatTypes.MoveFirst
    tRstTextCatTypes.FindFirst "c_text_cat_type_id = " + Chr(34) + cNode.Key + Chr(34)

    TxtTypeID.Value = cNode.Key
    TxtTypeDesc.Value = tRstTextCatTypes!c_text_cat_type_desc
    TxtTypeDescChn.Value = tRstTextCatTypes!c_text_cat_type_desc_chn
    tRstTextCatTypes.Close
    Set tRstTextCatTypes = Nothing
End If
End If
End If
End If
End If
End If
End Sub

```

```
Private Sub NodeSearch()
```

```

    Dim cNode As clsNode, tStr As String, tRstTextCatCodes As DAO.Recordset, tRstTextCatTypes As DAO.Recordset
    Dim tRstDummy As DAO.Recordset, tStrSQL As String, tStrSearchChn As String, tStrSearchEng As String
    Dim tStrLen As String, cmdSQL As ADODB.Command

```

```

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '

```

```

    TxtTextCatID.Value = -1
    TxtTextCatDesc.Value = ""

```

```
TxtTextCatDescChn.Value = ""
```

```
' all specific TextCatiation codes will have a type of the form 0101
' hence the ValuePath for the relevant node will be "K000/K01/K0101"
' The command to locate the relevant node is:
' tNode = TreeViewType.FindNode(tStrValuePath)
' TreeViewType.SelectedNode = tNode
```

```
' search for the search string in TEXT_BIBLCAT_CODES
TxtSearchChn.SetFocus
tStrSearchChn = Me.TxtSearchChn.Text
TxtSearch.SetFocus
tStrSearchEng = Me.TxtSearch.Text
CmdFind.SetFocus
```

```
gStrSearch = ""
gUseAlt = False
```

```
If tStrSearchChn <> "" Then
    tStrLen = Str(LenB(tStrSearchChn))
    gStrSearch = "LeftB(c_text_cat_desc_chn," + tStrLen + ") = '" + tStrSearchChn + "'"
    gStrSearchAlt = "InStrB(1,c_text_cat_desc_chn,'" + tStrSearchChn + "') > 0"
ElseIf tStrSearchEng <> "" Then
    tStrLen = Str(Len(tStrSearchEng))
    gStrSearch = "Left(c_text_cat_desc," + tStrLen + ") = '" + tStrSearchEng + "'"
    gStrSearchAlt = "InStr(1,c_text_cat_desc,'" + tStrSearchEng + "') > 0"
End If
```

```
If Not (gStrSearch = "") Then
```

```
    'MsgBox "Looking for TextCat"
```

```
    Set gRstTextCatCode = CurrentDb.OpenRecordset("TEXT_BIBLCAT_CODES", dbOpenDynaset)
```

```
    gRstTextCatCode.FindFirst gStrSearch
```

```
    If gRstTextCatCode.NoMatch Then
        gRstTextCatCode.FindFirst gStrSearchAlt
        gUseAlt = True
    End If
```

```
    If gRstTextCatCode.NoMatch Then
        gUseAlt = False
        CmdFindNext.Enabled = False
    Else
```

```
        ,
        CmdFindNext.Enabled = True
        ,
        ' next find the TextCat_type
        ,
```

```
        'MsgBox "Looking for TextCat type"
```

```
        Set tRstTextCatTypes = CurrentDb.OpenRecordset("TEXT_BIBLCAT_CODE_TYPE_REL", dbOpenDynaset)
```

```
        tRstTextCatTypes.FindNext "c_text_cat_code = " + Str(gRstTextCatCode!c_text_cat_code)
```

```
        If Not tRstTextCatTypes.NoMatch Then
```

```
            ,
            ' set the values
            ,
```

```
            TxtTextCatID.Value = gRstTextCatCode!c_text_cat_code
            If Not IsNull(gRstTextCatCode!c_text_cat_desc) Then
                TxtTextCatDesc.Value = gRstTextCatCode!c_text_cat_desc
            End If
            If Not IsNull(gRstTextCatCode!c_text_cat_desc_chn) Then
                TxtTextCatDescChn.Value = gRstTextCatCode!c_text_cat_desc_chn
            End If
            ,
```

```
            ' define the string
            ,
```

```
            tStr = tRstTextCatTypes!c_text_cat_type_id
            ,
```

```
            ' search the tree
```

```
            Set cNode = mcTree.Nodes(tStr)
```

```
            ,
            'MsgBox "found node"
            ,
```

```
            If Not IsNull(cNode) Then
```

```

    '
    Set mcTree.ActiveNode = cNode
    'cNode.Selected = True
    '
    ' then one makes it visible
    '
    'tNode.EnsureVisible
    '
    Set gNode = cNode
    '
    ' Finally populate the options and select the record.
    '
    CmdSelectAll.Enabled = True
    '
    cmdSQL.CommandText = "Delete * from ZZ_TEXT_BIBLCAT_CODES_TMP"
    cmdSQL.Execute tRecDeleted
    '
    tStrSQL = "INSERT INTO ZZ_TEXT_BIBLCAT_CODES_TMP ( c_text_cat_code, c_text_cat_desc, c_text_cat_desc_chn ) " + _
        "SELECT TEXT_BIBLCAT_CODES.c_text_cat_code, TEXT_BIBLCAT_CODES.c_text_cat_desc, TEXT_BIBLCAT_CODES.c_text_cat_desc_chn " + _
        "FROM TEXT_BIBLCAT_CODES INNER JOIN TEXT_BIBLCAT_CODE_TYPE_REL ON " + _
        "(TEXT_BIBLCAT_CODES.c_text_cat_code = TEXT_BIBLCAT_CODE_TYPE_REL.c_text_cat_code) " + _
        "WHERE ((TEXT_BIBLCAT_CODE_TYPE_REL.c_text_cat_type_id)='" + tStr + "'))"

    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted
    '
    ListTextCat.Requery
    For i = 0 To ListTextCat.ListCount
        ListTextCat.Selected(i) = False
    Next i
    gSelectCount = 0
    '
    For i = 0 To ListTextCat.ListCount - 1
        If gRstTextCatCode!c_text_cat_code = ListTextCat.Column(0, i) Then
            ListTextCat.ListIndex = i
            ListTextCat.Selected(i) = True
            gSelectCount = gSelectCount + 1
        End If
    Next i
    '
    ' set the type values
    '
    Set tRstTextCatTypes = CurrentDb.OpenRecordset("TEXT_BIBLCAT_TYPES", dbOpenDynaset)
    '
    tRstTextCatTypes.MoveFirst
    tRstTextCatTypes.FindFirst "c_text_cat_type_id = " + Chr(34) + cNode.Key + Chr(34)
    TxtTypeID.Value = cNode.Key
    TxtTypeDesc.Value = tRstTextCatTypes!c_text_cat_type_desc
    TxtTypeDescChn.Value = tRstTextCatTypes!c_text_cat_type_desc_chn
    tRstTextCatTypes.Close
    Set tRstTextCatTypes = Nothing
End If
End If
End If
End If
End Sub

```

frm_frmPickTEXTS - 1

Option Compare Database

```
Private Sub CmdCancel_Click()  
On Error GoTo Err_CmdCancel_Click
```

```
    DoCmd.Close
```

```
Exit_CmdCancel_Click:  
Exit Sub
```

```
Err_CmdCancel_Click:  
MsgBox Err.Description  
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdSelect_Click()  
Forms!frmPickTEXTS.Visible = False  
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
    frmTEXTS.Form.OrderBy = "c_title"  
    frmTEXTS.Form.OrderByOn = True
```

```
    If Not IsNull(Me.OpenArgs) Then  
        Dim strTexts As String  
        strTexts = Me.OpenArgs  
        Dim rsTexts As DAO.Recordset  
        Set rsTexts = frmTEXTS.Form.Recordset  
        rsTexts.FindFirst "c_textid = " & strTexts  
    End If
```

```
End Sub
```

```
Private Sub CmdFind_Click()  
On Error GoTo Err_CmdFind_Click
```

```
    Dim StrSearch As String  
    Me.TxtSearch.SetFocus  
    StrSearch = Me.TxtSearch.Value  
    If StrSearch <> "" Then  
        Dim rsTitle As DAO.Recordset  
        Set rsTitle = frmTEXTS.Form.Recordset  
        Dim StrSearchStr As String  
        StrSearchStr = "c_title_chn = " + Chr(34) + StrSearch + Chr(34)  
        rsTitle.FindFirst StrSearchStr  
    End If
```

```
Exit_CmdFind_Click:  
Exit Sub
```

```
Err_CmdFind_Click:  
MsgBox Err.Description  
Resume Exit_CmdFind_Click
```

```
End Sub
```

```
Private Sub TxtSearch_Change()  
    If Me.TxtSearch.Text = "" Then  
        Me.CmdFind.Enabled = False  
    Else  
        Me.CmdFind.Enabled = True  
    End If  
End Sub
```

frm_frmSearchPeopleOffice -1

```
Public gUseIndexYears As Boolean, gUseDynasties As Boolean, gFromDynasty As Integer, gToDynasty As Integer, gFromDynastyBegin As Integer, gFromDynastyEnd As Integer, gToDynastyBegin As Integer, gToDynastyEnd As Integer
Option Compare Database
```

```
Private Sub CmdCancel_Click()
On Error GoTo Err_CmdCancel_Click
```

```
DoCmd.Close
```

```
Exit_CmdCancel_Click:
Exit Sub
```

```
Err_CmdCancel_Click:
MsgBox Err.Description
Resume Exit_CmdCancel_Click
```

```
End Sub
```

```
Private Sub CmdDynastyFrom_Click()
Dim stDocName As String
Dim stLinkCriteria As String
Dim strFromDynasty As String, tStrFromDynastyText As String
```

```
If gFromDynasty < 0 Then
    strFromDynasty = ""
Else
    strFromDynasty = Str(gFromDynasty)
End If
```

```
stDocName = "frmPickDynasty"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty
```

```
If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then
```

```
Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
gFromDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value
```

```
Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
gFromDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value
```

```
Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
gFromDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
```

```
' check to see if we have a problem and reject selection
```

```
If gToDynasty > -1 Then
    If gFromDynastyBegin > gToDynastyEnd Then
        MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
        gFromDynasty = -1
        TxtDynastyFrom.Value = ""
    End If
End If
```

```
' value is OK
```

```
If gFromDynasty > -1 Then
    Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
    tStrFromDynastyText = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value
```

```
Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
TxtDynastyFrom.Value = tStrFromDynastyText + " " + Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
End If
```

```
DoCmd.Close acForm, stDocName
' reset ToDynasty if necessary (-2 = all dynasties)
```

```
If gToDynasty = -2 Then
    gToDynasty = -1
    TxtDynastyTo.Value = ""
End If
```

```
End If
```

```
End Sub
```

```

Private Sub CmdDynastyTo_Click()
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strToDynasty As String, tStrToDynastyText As String

    If gToDynasty = -1 Then
        strToDynasty = ""
    Else
        strToDynasty = Str(gToDynasty)
    End If

    stDocName = "frmPickDynasty"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

    If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

        Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
        gToDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
        gToDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
        gToDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
        '
        ' check to see if we have a problem and reject selection if needed
        '
        If gFromDynasty > -1 Then
            If gFromDynastyBegin > gToDynastyEnd Then
                MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
                gToDynasty = -1
                TxtDynastyTo.Value = ""
            End If
        End If
        '
        ' value is OK
        '
        If gToDynasty > -1 Then
            Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
            tStrToDynastyText = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

            Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
            TxtDynastyTo.Value = tStrToDynastyText + " " + Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
        End If

        DoCmd.Close acForm, stDocName
        '
        ' reset FromDynasty if necessary (-2 = all dynasties)
        '
        If gFromDynasty = -2 Then
            gFromDynasty = -1
            TxtDynastyFrom.Value = ""
        End If
    End If
End Sub

Private Sub CmdSearch_Click()
On Error GoTo Err_CmdSearch_Click

    Dim tStrSurname As String, tStrOffice As String, tStrQueryInsert As String, tStrQueryFrom As String, tStrQueryWhere As String, _
        tStrAndYears As String
    Dim cmdSQL As ADODB.Command, tRecCount

    tStrSurname = ""
    tStrOffice = ""
    tStrQuery = ""
    tRecCount = 0

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' define the FROM joins
    '
    tStrQueryFrom = "FROM ZZZ_BIOG_MAIN INNER JOIN ZZZ_BIOG_NAME_OFFICE ON ZZZ_BIOG_MAIN.c_personid = ZZZ_BIOG_NAME"

```

ME_OFFICE.c_personid "

' define the additional WHERE logic as blank

tStrAndYears = ""

If gUseIndexYears Then

If IsNull(Me.TxtIndexYearFrom.Value) Then

Me.TxtIndexYearFrom.Value = ""

End If

If IsNull(Me.TxtIndexYearTo.Value) Then

Me.TxtIndexYearTo.Value = ""

End If

If Me.TxtIndexYearFrom.Value = "" And TxtIndexYearTo.Value = "" Then

gUseIndexYears = False

ElseIf Me.TxtIndexYearFrom.Value = "" Then

tStrAndYears = "(ZZZ_BIOG_MAIN.c_index_year<= " + Str(TxtIndexYearTo.Value) + ") AND "

ElseIf Me.TxtIndexYearTo.Value = "" Then

tStrAndYears = "(ZZZ_BIOG_MAIN.c_index_year>= " + Str(TxtIndexYearFrom.Value) + ") AND "

Else

tStrAndYears = "((ZZZ_BIOG_MAIN.c_index_year<= " + Str(TxtIndexYearTo.Value) + ") AND (ZZZ_BIOG_MAIN.c_index_year>= " + Str(TxtIndexYearFrom.Value) + ")) AND "

End If

ElseIf gUseDynasties Then

If gFromDynasty = -2 Then

tStrAndYears = "((ZZZ_BIOG_MAIN.c_dy) > 0) AND "

ElseIf gFromDynasty = -1 And gToDynasty > 0 Then

tStrAndYears = "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") AND "

'

redefine the FROM joins to include DYNASTIES

'

tStrQueryFrom = "FROM (ZZZ_BIOG_MAIN INNER JOIN ZZZ_BIOG_NAME_OFFICE ON ZZZ_BIOG_MAIN.c_personid = ZZZ_BIOG_NAME_OFFICE.c_personid) " +

"INNER JOIN DYNASTIES ON ZZZ_BIOG_MAIN.c_dy = DYNASTIES.c_dy "

ElseIf gFromDynasty > 0 And gToDynasty = -1 Then

tStrAndYears = "((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND "

'

redefine the FROM joins to include DYNASTIES

'

tStrQueryFrom = "FROM (ZZZ_BIOG_MAIN INNER JOIN ZZZ_BIOG_NAME_OFFICE ON ZZZ_BIOG_MAIN.c_personid = ZZZ_BIOG_NAME_OFFICE.c_personid) " +

"INNER JOIN DYNASTIES ON ZZZ_BIOG_MAIN.c_dy = DYNASTIES.c_dy "

ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then

tStrAndYears = "((ZZZ_BIOG_MAIN.c_dy) = " + Str(gToDynasty) + ") AND "

ElseIf gFromDynasty > 0 And gToDynasty > 0 Then

tStrAndYears = "((DYNASTIES.c_end>" + Str(gFromDynastyBegin) + ") AND (DYNASTIES.c_start<=" + Str(gToDynastyEnd) + ")) AND "

'

redefine the FROM joins to include DYNASTIES

'

tStrQueryFrom = "FROM (ZZZ_BIOG_MAIN INNER JOIN ZZZ_BIOG_NAME_OFFICE ON ZZZ_BIOG_MAIN.c_personid = ZZZ_BIOG_NAME_OFFICE.c_personid) " +

"INNER JOIN DYNASTIES ON ZZZ_BIOG_MAIN.c_dy = DYNASTIES.c_dy "

Else

tStrAndYears = "((ZZZ_BIOG_MAIN.c_dy) > 0) AND "

End If

End If

'Me.TxtSurnameChn.SetFocus

If TxtSurnameChn.Value <> "" And TxtOfficeChn.Value <> "" Then

tStrSurname = Trim(TxtSurnameChn.Value)

tStrOffice = Trim(TxtOfficeChn.Value)

tStrQueryInsert = "INSERT INTO Z_NAME_SEARCH (c_personid, c_name, c_name_chn) " +

"SELECT DISTINCT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_chn "

tStrQueryWhere = "WHERE (" +

"(" + tStrAndYears +

"(ZZZ_BIOG_NAME_OFFICE.c_surname_chn='" + tStrSurname + "') AND (ZZZ_BIOG_NAME_OFFICE.c_office_ch

n like '%" + tStrOffice + "%')) OR " +

"(" + tStrAndYears +

"(ZZZ_BIOG_NAME_OFFICE.c_surname_chn='" + tStrSurname + "') AND (ZZZ_BIOG_NAME_OFFICE.c_office_ch

n_alt like '%" + tStrOffice + "%'))"

Else

'TxtSearchPY.SetFocus

If TxtSurnamePY.Value <> "" And TxtOfficePY.Value <> "" Then

tStrSurname = Trim(TxtSurnamePY.Value)

tStrOffice = Trim(TxtOfficePY.Value)

tStrQueryInsert = "INSERT INTO Z_NAME_SEARCH (c_personid, c_name, c_name_chn) " +

"SELECT DISTINCT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_chn "


```

        tStrQueryWhere = "WHERE (" + _
            "(" + tStrAndYears + _
            "(ZZZ_BIOG_NAME_OFFICE.c_surname='" + tStrSurname + "') AND (ZZZ_BIOG_NAME_OFFICE.c_office_pinyin
like '%" + tStrOffice + "%')) OR " + _
            "(" + tStrAndYears + _
            "(ZZZ_BIOG_NAME_OFFICE.c_surname='" + tStrSurname + "') AND (ZZZ_BIOG_NAME_OFFICE.c_office_pinyin
alt like '%" + tStrOffice + "%')))"
    End If
End If
'MsgBox tStrAndYears

If tStrQueryInsert <> "" Then
    cmdSQL.CommandText = "Delete * from Z_NAME_SEARCH"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = tStrQueryInsert + tStrQueryFrom + tStrQueryWhere
    cmdSQL.Execute tRecCount
End If
If tRecCount > 0 Then
    Me.Form.Visible = False
    'Forms!frmSearchPeopleOffice.visibe = False
Else
    DoCmd.Close
End If

Exit_CmdSearch_Click:
Exit Sub

Err_CmdSearch_Click:
MsgBox Err.Description
Resume Exit_CmdSearch_Click

End Sub

Private Sub Form_Open(Cancel As Integer)
    Me.TxtSurnameChn.Value = ""
    Me.TxtSurnamePY.Value = ""
    Me.TxtOfficeChn.Value = ""
    Me.TxtOfficePY.Value = ""
    Me.TxtIndexYearFrom.Value = ""
    Me.TxtIndexYearTo.Value = ""
    Me.CmdSearch.Enabled = False

    gFromDynasty = -1
    gToDynasty = -1
    gUseIndexYears = False
    gUseDynasties = False
End Sub

Private Sub FrameFilterYears_Click()
' disable all
    Me.CmdDynastyFrom.Enabled = False
    Me.CmdDynastyTo.Enabled = False
    Me.TxtDynastyFrom.Enabled = False
    Me.TxtDynastyTo.Enabled = False
    Me.TxtDynastyFrom.Locked = False
    Me.TxtDynastyTo.Locked = False

    Me.TxtIndexYearFrom.Enabled = False
    Me.TxtIndexYearTo.Enabled = False

    gUseIndexYears = False
    gUseDynasties = False

    If FrameFilterYears.Value = 2 Then

        ' enable index years
        Me.TxtIndexYearFrom.Enabled = True
        Me.TxtIndexYearTo.Enabled = True
        gUseIndexYears = True

    ElseIf FrameFilterYears.Value = 3 Then

        ' enable dynasties
        Me.CmdDynastyFrom.Enabled = True
        Me.CmdDynastyTo.Enabled = True
        Me.TxtDynastyFrom.Enabled = True
        Me.TxtDynastyTo.Enabled = True
        Me.TxtDynastyFrom.Locked = True

```

```
Me.TxtDynastyTo.Locked = True  
gUseDynasties = True
```

```
End If
```

```
End Sub
```

```
Private Sub TxtOfficeChn_Change()
```

```
    If Me.TxtSurnameChn.Value = "" Or Me.TxtOfficeChn.Text = "" Then  
        Me.CmdSearch.Enabled = False
```

```
    Else
```

```
        Me.TxtSurnamePY.Value = ""
```

```
        Me.TxtOfficePY.Value = ""
```

```
        Me.CmdSearch.Enabled = True
```

```
    End If
```

```
End Sub
```

```
Private Sub TxtOfficePY_Change()
```

```
    If Me.TxtSurnamePY.Value = "" Or Me.TxtOfficePY.Text = "" Then  
        Me.CmdSearch.Enabled = False
```

```
    Else
```

```
        Me.TxtSurnameChn.Value = ""
```

```
        Me.TxtOfficeChn.Value = ""
```

```
        Me.CmdSearch.Enabled = True
```

```
    End If
```

```
End Sub
```

```
Private Sub TxtSurnameChn_Change()
```

```
    If Me.TxtSurnameChn.Text = "" Or Me.TxtOfficeChn.Value = "" Then  
        Me.CmdSearch.Enabled = False
```

```
    Else
```

```
        Me.TxtSurnamePY.Value = ""
```

```
        Me.TxtOfficePY.Value = ""
```

```
        Me.CmdSearch.Enabled = True
```

```
    End If
```

```
End Sub
```

```
Private Sub TxtSurnamePY_Change()
```

```
    If Me.TxtSurnamePY.Text = "" Or Me.TxtOfficePY.Value = "" Then  
        Me.CmdSearch.Enabled = False
```

```
    Else
```

```
        Me.TxtSurnameChn.Value = ""
```

```
        Me.TxtOfficeChn.Value = ""
```

```
        Me.CmdSearch.Enabled = True
```

```
    End If
```

```
End Sub
```

frm_frmSelectPerson - 1

Option Compare Database

```
Private Sub CmdClose_Click()  
On Error GoTo Err_CmdClose_Click
```

```
DoCmd.Close
```

```
Exit_CmdClose_Click:  
Exit Sub
```

```
Err_CmdClose_Click:  
MsgBox Err.Description  
Resume Exit_CmdClose_Click
```

```
End Sub  
Private Sub CmdSelect_Click()  
On Error GoTo Err_CmdSelect_Click
```

```
Forms("frmSelectPerson").Visible = False
```

```
Exit_CmdSelect_Click:  
Exit Sub
```

```
Err_CmdSelect_Click:  
MsgBox Err.Description  
Resume Exit_CmdSelect_Click
```

```
End Sub
```

```
Private Sub Form_Open(Cancel As Integer)
```

```
If Not IsNull(Me.OpenArgs) Then  
Dim strID As String  
strID = Me.OpenArgs  
Dim cmdSQL As ADODB.Command, tRecNum As Long
```

```
Set cmdSQL = New ADODB.Command  
cmdSQL.ActiveConnection = CurrentProject.Connection  
cmdSQL.CommandType = adCmdText  
,
```

```
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"  
cmdSQL.Execute tRecNum
```

```
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_in  
dex_year_type_desc, " + _  
"c_index_year_type_hz, c_dynasty, c_dynasty_chn, c_female ) " + _  
"SELECT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MAIN.c_ind  
ex_year, " + _  
"ZZZ_BIOG_MAIN.c_index_year_type_desc, ZZZ_BIOG_MAIN.c_index_year_type_hz, ZZZ_BIOG_MAIN.c_dynast  
y, " + _  
"ZZZ_BIOG_MAIN.c_dynasty_chn, ZZZ_BIOG_MAIN.c_female " + _  
"FROM ZZZ_BIOG_MAIN " + _  
"WHERE (((ZZZ_BIOG_MAIN.c_personid)=" + Trim(strID) + "))"  
cmdSQL.Execute tRecNum
```

```
If tRecNum > 0 Then  
Set frmPersonSearch.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)  
End If  
End If
```

```
End Sub  
Private Sub CmdFind_Click()  
On Error GoTo Err_CmdFind_Click
```

```
Dim tRstSearch As DAO.Recordset, tStr As String, tQt As String, tQuery As QueryDef, tStrName As String  
Dim cmdSQL As ADODB.Command, tRecNum As Long
```

```
' the logic of search is to use the characters first, then the pinyin  
' the search first looks at ZZZ_BIOG_MAIN's c_name and c_name_chn  
' it then looks at c_name_proper and c_name_rm in ZZZ_BIOG_MAIN  
' then it looks at ZZZ_ALTNAMES
```

```
tQt = Chr(34)
```

```
' first make sure that the browser recordset is a dummy
```

```
Set tRstSearch = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_PL", dbOpenDynaset)  
Set Me.frmPersonSearch.Form.Recordset = tRstSearch
```

```

' Now zap Z_NAME_SEARCH

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecNum

cmdSQL.CommandText = "Delete * from Z_NAME_SEARCH"
cmdSQL.Execute tRecNum

' now populate Z_NAME_SEARCH SELECT from ZZZ_NAMES

If IsNull(TxtNameChn.Value) Then
    If IsNull(TxtName.Value) Then
        tStr = "Quit"
    Else
        If Me.TxtName.Value = "" Then
            tStr = "Quit"
        Else
            tStrName = TxtName.Value
            If Left(tStrName, 1) = "!" Then
                tStrName = Mid(TxtName.Value, 2)
                tStr = " Left(c_name," + Str(Len(tStrName)) + ") = " + tQt + Trim(tStrName) + tQt
            ElseIf UCase(Left(tStrName, 1)) = Left(tStrName, 1) Then
                tStr = " Left(c_name," + Str(Len(tStrName)) + ") = " + tQt + Trim(tStrName) + tQt + _
                    " OR c_name LIKE " + tQt + "%" + " " + Trim(tStrName) + "%" + tQt
            Else
                tStr = " c_name LIKE " + tQt + "%" + Trim(tStrName) + "%" + tQt
            End If
        End If
    End If
Else
    If Me.TxtNameChn.Value = "" Then
        If IsNull(TxtName.Value) Then
            tStr = "Quit"
        Else
            If Me.TxtName.Value = "" Then
                tStr = "Quit"
            Else
                tStrName = TxtName.Value
                If Left(tStrName, 1) = "!" Then
                    tStrName = Mid(TxtName.Value, 2)
                    tStr = " Left(c_name," + Str(Len(tStrName)) + ") = " + tQt + Trim(tStrName) + tQt
                ElseIf UCase(Left(tStrName, 1)) = Left(tStrName, 1) Then
                    tStr = " Left(c_name," + Str(Len(tStrName)) + ") = " + tQt + Trim(tStrName) + tQt + _
                        " OR c_name LIKE " + tQt + "%" + " " + Trim(tStrName) + "%" + tQt
                Else
                    tStr = " c_name LIKE " + tQt + "%" + Trim(tStrName) + "%" + tQt
                End If
            End If
        End If
    Else
        tStr = " c_name_chn LIKE " + tQt + "%" + Trim(TxtNameChn.Value) + "%" + tQt
    End If
End If

tRecNum = 0
If Not (tStr = "Quit") Then
    tStr = "INSERT INTO Z_NAME_SEARCH SELECT c_personid, c_name, c_name_chn " + _
        "FROM ZZZ_NAMES WHERE" + tStr

    cmdSQL.CommandText = tStr
    cmdSQL.Execute tRecNum
End If

If tRecNum > 0 Then
    tStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_index_year_type_
desc, " + _
        "c_index_year_type_hz, c_dynasty, c_dynasty_chn, c_female ) " + _
        "SELECT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MAIN.c_ind
ex_year, " + _
        "ZZZ_BIOG_MAIN.c_index_year_type_desc, ZZZ_BIOG_MAIN.c_index_year_type_hz, ZZZ_BIOG_MAIN.c_dynast
y, " + _
        "ZZZ_BIOG_MAIN.c_dynasty_chn, ZZZ_BIOG_MAIN.c_female " + _
        "FROM Z_NAME_SEARCH INNER JOIN ZZZ_BIOG_MAIN ON Z_NAME_SEARCH.c_personid = ZZZ_BIOG_MAIN.c_personid"

    cmdSQL.CommandText = tStr
    cmdSQL.Execute tRecNum

```

```
        Set tRstSearch = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
    End If
    ' tRstSearch.Index = "c_name"
    Set Me.frmPersonSearch.Form.Recordset = tRstSearch

Exit_CmdFind_Click:
    Exit Sub

Err_CmdFind_Click:
    MsgBox Err.Description
    Resume Exit_CmdFind_Click

End Sub

Private Sub TxtNameChn_Change()
    If TxtNameChn.Text = "" Then
        If TxtName.Value = "" Then
            CmdFind.Enabled = False
        End If
    Else
        TxtName.Value = ""
        CmdFind.Enabled = True
    End If
End Sub

Private Sub TxtName_Change()
    If Me.TxtName.Text = "" Then
        If TxtNameChn.Value = "" Then
            Me.CmdFind.Enabled = False
        End If
    Else
        TxtNameChn.Value = ""
        CmdFind.Enabled = True
    End If
End Sub
```

frmSTATUS_CODES - 1

Option Compare Database

```
Private Sub c_status_code_Click()  
    Dim tRst As DAO.Recordset  
  
    Set tRst = Me.Recordset  
    Forms!frmPickStatus2.Form!TxtStatusID.Value = tRst!c_status_code  
    Forms!frmPickStatus2.Form!TxtStatusDesc.Value = tRst!c_status_desc  
    Forms!frmPickStatus2.Form!TxtStatusDescChn.Value = tRst!c_status_desc_chn  
    Forms!frmPickStatus2.Form!TxtTypeID.Value = "000"  
    Forms!frmPickStatus2.Form!TxtTypeDescChn.Value = ""  
    Forms!frmPickStatus2.Form!TxtTypeDesc.Value = ""  
    Me.DatasheetBackColor = RGB(255, 255, 255)  
    Me.DatasheetForeColor = RGB(0, 0, 0)  
    Forms!frmPickStatus2.Form!CmdSelectAll.Caption = "Select All"  
    Forms!frmPickStatus2.Form!CmdSelectAll.Enabled = True  
    Forms!frmPickStatus2.Form!CmdSelect.Enabled = True  
    Set tRst = Nothing
```

End Sub

```
Private Sub c_status_desc_chn_Click()  
    Dim tRst As DAO.Recordset  
  
    Set tRst = Me.Recordset  
    Forms!frmPickStatus2.Form!TxtStatusID.Value = tRst!c_status_code  
    Forms!frmPickStatus2.Form!TxtStatusDesc.Value = tRst!c_status_desc  
    Forms!frmPickStatus2.Form!TxtStatusDescChn.Value = tRst!c_status_desc_chn  
    Forms!frmPickStatus2.Form!TxtTypeID.Value = "000"  
    Forms!frmPickStatus2.Form!TxtTypeDescChn.Value = ""  
    Forms!frmPickStatus2.Form!TxtTypeDesc.Value = ""  
    Me.DatasheetBackColor = RGB(255, 255, 255)  
    Me.DatasheetForeColor = RGB(0, 0, 0)  
    Forms!frmPickStatus2.Form!CmdSelectAll.Caption = "Select All"  
    Forms!frmPickStatus2.Form!CmdSelectAll.Enabled = True  
    Forms!frmPickStatus2.Form!CmdSelect.Enabled = True  
    Set tRst = Nothing
```

End Sub

```
Private Sub c_status_desc_Click()  
    Dim tRst As DAO.Recordset  
  
    Set tRst = Me.Recordset  
    Forms!frmPickStatus2.Form!TxtStatusID.Value = tRst!c_status_code  
    Forms!frmPickStatus2.Form!TxtStatusDesc.Value = tRst!c_status_desc  
    Forms!frmPickStatus2.Form!TxtStatusDescChn.Value = tRst!c_status_desc_chn  
    Forms!frmPickStatus2.Form!TxtTypeID.Value = "000"  
    Forms!frmPickStatus2.Form!TxtTypeDesc.Value = ""  
    Forms!frmPickStatus2.Form!TxtTypeDescChn.Value = ""  
    Me.DatasheetBackColor = RGB(255, 255, 255)  
    Me.DatasheetForeColor = RGB(0, 0, 0)  
    Forms!frmPickStatus2.Form!CmdSelectAll.Caption = "Select All"  
    Forms!frmPickStatus2.Form!CmdSelectAll.Enabled = True  
    Forms!frmPickStatus2.Form!CmdSelect.Enabled = True  
    Set tRst = Nothing
```

End Sub

Option Compare Database

```
Private Sub c_title_LostFocus()  
    Dim intLastTextID As Long  
    Dim intTextID As Long  
  
    If IsNull(c_textid.Value) And Not IsNull(c_title) Then  
        intLastTextID = DMax("c_textid", "TEXT_CODES")  
        TxtLastTextID.Value = intLastTextID  
  
        TxtLastTextID.Visible = True  
        TxtLastTextID.SetFocus  
        intTextID = TxtLastTextID.Value  
        c_textid.Value = intTextID + 1  
  
        c_textid.SetFocus  
        TxtLastTextID.Visible = False  
    End If  
End Sub
```


Option Compare Database

```
Private Sub c_assoc_desc_chn_Click()  
    Dim tRst As DAO.Recordset  
  
    Set tRst = Me.Recordset  
    Forms!frmPickAssoc2.Form!TxtAssocID.Value = tRst!c_assoc_code  
    Forms!frmPickAssoc2.Form!TxtAssocDesc.Value = tRst!c_assoc_desc  
    Forms!frmPickAssoc2.Form!TxtAssocDescChn.Value = tRst!c_assoc_desc_chn  
    Forms!frmPickAssoc2.Form!TxtTypeID.Value = "000"  
    Forms!frmPickAssoc2.Form!TxtTypeDescChn.Value = ""  
    Forms!frmPickAssoc2.Form!TxtTypeDesc.Value = ""  
    Me.DatasheetBackColor = RGB(255, 255, 255)  
    Me.DatasheetForeColor = RGB(0, 0, 0)  
    Forms!frmPickAssoc2.Form!CmdSelectAll.Caption = "Select All"  
    Forms!frmPickAssoc2.Form!CmdSelectAll.Enabled = True  
    Forms!frmPickAssoc2.Form!CmdSelect.Enabled = True  
    Set tRst = Nothing
```

End Sub

```
Private Sub c_assoc_desc_Click()  
    Dim tRst As DAO.Recordset  
  
    Set tRst = Me.Recordset  
    Forms!frmPickAssoc2.Form!TxtAssocID.Value = tRst!c_assoc_code  
    Forms!frmPickAssoc2.Form!TxtAssocDesc.Value = tRst!c_assoc_desc  
    Forms!frmPickAssoc2.Form!TxtAssocDescChn.Value = tRst!c_assoc_desc_chn  
    Forms!frmPickAssoc2.Form!TxtTypeID.Value = "000"  
    Forms!frmPickAssoc2.Form!TxtTypeDesc.Value = ""  
    Forms!frmPickAssoc2.Form!TxtTypeDescChn.Value = ""  
    Me.DatasheetBackColor = RGB(255, 255, 255)  
    Me.DatasheetForeColor = RGB(0, 0, 0)  
    Forms!frmPickAssoc2.Form!CmdSelectAll.Caption = "Select All"  
    Forms!frmPickAssoc2.Form!CmdSelectAll.Enabled = True  
    Forms!frmPickAssoc2.Form!CmdSelect.Enabled = True  
    Set tRst = Nothing
```

End Sub

Option Compare Database

```
Private Sub c_desc_chn_Click()  
    Dim tRst As DAO.Recordset  
  
    Set tRst = Me.Recordset  
    Forms!frmPickEntry.Form!TxtEntryCode.Value = tRst!c_entry_code  
    Forms!frmPickEntry.Form!TxtEntryDesc.Value = tRst!c_entry_desc  
    Forms!frmPickEntry.Form!TxtEntryChn.Value = tRst!c_entry_desc_chn  
    Forms!frmPickEntry.Form!TxtTypeID.Value = ""  
    Forms!frmPickEntry.Form!TxtTypeDesc.Value = ""  
    Forms!frmPickEntry.Form!TxtTypeChn.Value = ""  
    Me.DatasheetBackColor = RGB(255, 255, 255)  
    Me.DatasheetForeColor = RGB(0, 0, 0)  
    Forms!frmPickEntry.Form!CmdSelectAll.Caption = "Select All"  
    Forms!frmPickEntry.Form!CmdSelectAll.Enabled = True  
    Forms!frmPickEntry.Form!CmdSelect.Enabled = True  
    Set tRst = Nothing  
End Sub
```

```
Private Sub c_desc_Click()  
    Dim tRst As DAO.Recordset  
  
    Set tRst = Me.Recordset  
    Forms!frmPickEntry.Form!TxtEntryCode.Value = tRst!c_entry_code  
    Forms!frmPickEntry.Form!TxtEntryDesc.Value = tRst!c_entry_desc  
    Forms!frmPickEntry.Form!TxtEntryChn.Value = tRst!c_entry_desc_chn  
    Forms!frmPickEntry.Form!TxtTypeID.Value = ""  
    Forms!frmPickEntry.Form!TxtTypeDesc.Value = ""  
    Forms!frmPickEntry.Form!TxtTypeChn.Value = ""  
    Me.DatasheetBackColor = RGB(255, 255, 255)  
    Me.DatasheetForeColor = RGB(0, 0, 0)  
    Forms!frmPickEntry.Form!CmdSelectAll.Caption = "Select All"  
    Forms!frmPickEntry.Form!CmdSelectAll.Enabled = True  
    Forms!frmPickEntry.Form!CmdSelect.Enabled = True  
    Set tRst = Nothing  
End Sub
```

Option Compare Database

```
Private Sub c_office_desc_chn_Click()  
    Dim tRst As DAO.Recordset  
  
    Set tRst = Me.Recordset  
    Forms!frmPickOfficeTree.Form!TxtOfficeCode.Value = tRst!c_office_id  
    Forms!frmPickOfficeTree.Form!TxtOfficeDesc.Value = tRst!c_office_trans  
    Forms!frmPickOfficeTree.Form!TxtOfficeDescChn.Value = tRst!c_office_chn  
    Forms!frmPickOfficeTree.Form!TxtTypeDescChn.Value = ""  
    Forms!frmPickOfficeTree.Form!TxtTypeDesc.Value = ""  
    Me.DatasheetBackColor = RGB(255, 255, 255)  
    Me.DatasheetForeColor = RGB(0, 0, 0)  
    Forms!frmPickOfficeTree.Form!CmdSelectAll.Caption = "Select All"  
    Forms!frmPickOfficeTree.Form!CmdSelectAll.Enabled = True  
    Forms!frmPickOfficeTree.Form!CmdSelect.Enabled = True  
    Set tRst = Nothing  
  
End Sub
```

```
Private Sub c_office_desc_Click()  
    Dim tRst As DAO.Recordset  
  
    Set tRst = Me.Recordset  
    Forms!frmPickOfficeTree.Form!TxtOfficeCode.Value = tRst!c_office_id  
    Forms!frmPickOfficeTree.Form!TxtOfficeDesc.Value = tRst!c_office_trans  
    Forms!frmPickOfficeTree.Form!TxtOfficeDescChn.Value = tRst!c_office_chn  
    Forms!frmPickOfficeTree.Form!TxtTypeDescChn.Value = ""  
    Forms!frmPickOfficeTree.Form!TxtTypeDesc.Value = ""  
    Me.DatasheetBackColor = RGB(255, 255, 255)  
    Me.DatasheetForeColor = RGB(0, 0, 0)  
    Forms!frmPickOfficeTree.Form!CmdSelectAll.Caption = "Select All"  
    Forms!frmPickOfficeTree.Form!CmdSelectAll.Enabled = True  
    Forms!frmPickOfficeTree.Form!CmdSelect.Enabled = True  
    Set tRst = Nothing  
  
End Sub
```

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Private Sub CmdPickKinID_Click()

On Error GoTo Err_CmdPickKinID_Click

```
Dim stDocName As String
Dim stLinkCriteria As String
Dim strNH As String
```

```
c_kin_id.Visible = True
c_kin_id.SetFocus
strNH = c_kin_id.Text
```

```
stDocName = "frmPickPeople"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH
```

If CurrentProject.AllForms("frmPickPeople").IsLoaded Then

```
Dim intKinID As Long
Dim strKinName As String
Dim strKinNameChn As String
```

```
Forms!frmPickPeople!frmPerson.Form!c_personid.Visible = True
intKinID = Forms!frmPickPeople!frmPerson.Form!c_personid.Value
```

```
Forms!frmPickPeople!frmPerson.Form!c_name.SetFocus
strKinName = Forms!frmPickPeople!frmPerson.Form!c_name.Value
```

```
Forms!frmPickPeople!frmPerson.Form!c_name_chn.SetFocus
strKinNameChn = Forms!frmPickPeople!frmPerson.Form!c_name_chn.Value
```

```
c_kin_id.Value = intKinID
TxtKinNM.Value = strKinName
TxtKinNM_CHN.Value = strKinNameChn
```

DoCmd.Close acForm, stDocName

End If

```
CmdPickKinID.SetFocus
c_kin_id.Visible = False
```

Exit CmdPickKinID_Click:

```
CmdPickKinID.SetFocus
c_kin_id.Visible = False
Exit Sub
```

Err_CmdPickKinID_Click:

```
MsgBox Err.Description
Resume Exit_CmdPickKinID_Click
```

End Sub

Private Sub CmdPickKinRel_Click()

On Error GoTo Err_CmdPickKinRel_Click

```
Dim stDocName As String
Dim stLinkCriteria As String
Dim strNH As String
```

```
c_kin_code.Visible = True
c_kin_code.SetFocus
strKR = c_kin_code.Text
```

stDocName = "frmPickKINSHIP_CODES"

DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strKR

If CurrentProject.AllForms("frmPickKINSHIP_CODES").IsLoaded Then

```
Dim intKR As Integer
Dim strKR_EN As String
Dim strKR_CHN As String
```

```
Forms!frmPickKINSHIP_CODES!frmKINSHIP_CODES.Form!c_kincode.SetFocus
intKR = Forms!frmPickKINSHIP_CODES!frmKINSHIP_CODES.Form!c_kincode.Value
c_kin_code.Value = intKR
```

```
Forms!frmPickKINSHIP_CODES!frmKINSHIP_CODES.Form!c_kinrel.SetFocus
strKR_EN = Forms!frmPickKINSHIP_CODES!frmKINSHIP_CODES.Form!c_kinrel.Value
TxtKinRel.Value = strKR_EN
```

Forms!frmPickKINSHIP_CODES!frmKINSHIP_CODES.Form!c_kinrel_chn.SetFocus

```

    strKR_CHN = Forms!frmPickKINSHIP_CODES!frmKINSHIP_CODES.Form!c_kinrel_chn.Value
    TxtKinRel_CHN.Value = strKR_CHN

```

```

    DoCmd.Close acForm, stDocName
End If

```

```

    CmdPickKinRel.SetFocus
    c_kin_code.Visible = False

```

```

Exit_CmdPickKinRel_Click:
    Exit Sub

```

```

Err_CmdPickKinRel_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickKinRel_Click

```

```

End Sub

```

```

Private Sub CmdPickSource_Click()
On Error GoTo Err_CmdPickSource_Click

```

```

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strSC As String

```

```

    c_source.Visible = True
    c_source.SetFocus
    strSC = c_source.Text

```

```

    stDocName = "frmPickTEXTS"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strSC

```

```

    If CurrentProject.AllForms("frmPickTEXTS").IsLoaded Then
        Dim intSC As Long
        Dim strSC_CHN As String

```

```

        Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.SetFocus
        intSC = Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.Value
        c_source.Value = intSC

```

```

        Forms!frmPickTEXTS!frmTEXTS.Form!c_title.SetFocus
        strSC_CHN = Forms!frmPickTEXTS!frmTEXTS.Form!c_title_chn.Value
        TxtTitle_CHN.Value = strSC_CHN

```

```

        DoCmd.Close acForm, stDocName
    End If

```

```

CmdPickSource.SetFocus
c_source.Visible = False

```

```

Exit_CmdPickSource_Click:
    Exit Sub

```

```

Err_CmdPickSource_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickSource_Click

```

```

End Sub

```

```

Private Sub Form_AfterDelConfirm(STATUS As Integer)
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

```

```

    Dim intKinOld As Long
    Dim intKinCodeOld As Integer
    Dim intKinPairOld As Integer
    Dim tGenderOld As Integer
    Dim intKinCodeFind As Integer
    Dim intPersonFind As Long
    Dim intPerson As Long
    Dim blnRecordAdded As Boolean

```

```

    If STATUS = acDeleteOK Then

```

```

        'Find the old record (if there is one) and delete it
        intPerson = TxtDelPersonID.Value
        intKinOld = TxtDelKinID.Value
        intKinCodeOld = TxtDelKinCode.Value

```

```

        'Add a new entry to Del_Log
        rst.Open "DEL_LOG", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
        rst.AddNew
        rst!c_personid = intPerson
        rst!c_subform = "KINSHIP"
        rst!c_kin_id = intKinOld
        rst!c_kin_code = intKinCodeOld
        rst!c_source = TxtDelSource.Value
        rst!c_pages = TxtDelPages.Value
        rst!c_notes = TxtDelNotes.Value

        rst.Update
        blnRecordAdded = True
        rst.Close

        'Determine the gender of the kin
        rst.Open "BIOG_MAIN", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
        rst.Find "c_personid = " & intPerson
        tGenderOld = rst.Fields("c_female")
        rst.Close

        'Find the pair-code
        rst.Open "KINSHIP_CODES", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
        rst.Find "c_kincode = " & intKinCodeOld

        If tGenderOld = 0 Then
            intKinPairOld = rst.Fields("c_kin_pair1")
        Else
            intKinPairOld = rst.Fields("c_kin_pair2")
        End If

        rst.Close

        rst.Open "KIN_DATA", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
        Do
            rst.Find "c_personid = " & intKinOld
            intKinCodeFind = rst.Fields("c_kin_code")
            intPersonFind = rst.Fields("c_kin_id")

            If intKinCodeFind = intKinPairOld And intPersonFind = intPerson Then
                rst.Delete
                rst.Update
                rst.Close
                Exit Do
            Else
                rst.MoveNext
            End If
        Loop

    End If
End Sub

```

```
Private Sub Form_BeforeUpdate(Cancel As Integer)
```

```
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset
```

```
    Dim intKinCode As Integer
    Dim intKinPair As Integer
    Dim intPerson As Long
    Dim tGender As Integer
    Dim intKin As Long
    Dim blnRecordAdded As Boolean
```

```
    Dim intKinOld As Long
    Dim intKinPairOld As Integer
    Dim tGenderOld As Integer
    Dim intPersonFind As Long
    Dim intKinCodeFind As Integer
```

```
    intPerson = c_personid.Value
    intKin = c_kin_id.Value
    intKinCode = c_kin_code.Value
```

```
    'Determine the gender of the kin
    rst.Open "BIOG_MAIN", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.Find "c_personid = " & intPerson
    tGender = rst.Fields("c_female")
    rst.Close

```

```

'Find the pair-code
rst.Open "KINSHIP_CODES", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.Find "c_kincode = " & c_kin_code.Value

If tGender = 0 Then
    intKinPair = rst.Fields("c_kin_pair1")
Else
    intKinPair = rst.Fields("c_kin_pair2")
End If

rst.Close

'Find the old record (if there is one) and delete it
If Not IsNull(c_kin_code.OldValue) Or Not IsNull(c_kin_id.OldValue) Then
    If Not IsNull(c_kin_id.OldValue) Then
        intKinOld = c_kin_id.OldValue
    End If

    'Determine the gender of the kin
    rst.Open "BIOG_MAIN", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.Find "c_personid = " & intPerson
    tGenderOld = rst.Fields("c_female")
    rst.Close

    'Find the pair-code
    rst.Open "KINSHIP_CODES", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.Find "c_kincode = " & c_kin_code.OldValue

    If tGenderOld = 0 Then
        intKinPairOld = rst.Fields("c_kin_pair1")
    Else
        intKinPairOld = rst.Fields("c_kin_pair2")
    End If

    rst.Close

    rst.Open "KIN_DATA", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    Do
        rst.Find "c_personid = " & intKinOld
        intKinCodeFind = rst.Fields("c_kin_code")
        intPersonFind = rst.Fields("c_kin_id")

        If intKinCodeFind = intKinPairOld And intPersonFind = intPerson Then
            rst.Delete
            rst.Update
            rst.Close
            Exit Do
        Else
            rst.MoveNext
        End If
    Loop
End If

'Add the new entry
rst.Open "KIN_DATA", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.AddNew
rst!c_personid = intKin
rst!c_kin_id = intPerson
rst!c_kin_code = intKinPair

rst!c_notes = c_notes
rst!c_source = c_source
rst!c_pages = c_pages
rst!c_autogen_fr_id = intPerson
rst!c_autogen_fr_code = intKinCode
rst!c_autogen_notes = "Auto-generated from PersonID = " & Trim(intPerson) & ", KinCode = " & Trim(intKinC
ode) & "."

rst.Update
blnRecordAdded = True

rst.Close

End Sub

Private Sub Form_Current()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

```

```

If IsNull(c_source.Value) Then
    TxtTitle_CHN.Value = ""
Else
    rst.Open "TEXT_CODES", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_textid = " & c_source.Value

    TxtTitle_CHN.Value = rst.Fields("c_title_chn")
    rst.Close
End If

If IsNull(c_kin_code.Value) Then
    TxtKinRel.Value = ""
    TxtKinRel_CHN.Value = ""
Else
    rst.Open "KINSHIP_CODES", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_kincode = " & c_kin_code.Value

    TxtKinRel.Value = rst.Fields("c_kinrel")
    TxtKinRel_CHN.Value = rst.Fields("c_kinrel_chn")

    rst.Close
End If

If IsNull(c_kin_id.Value) Then
    TxtKinNM.Value = ""
    TxtKinNM_CHN.Value = ""
Else
    rst.Open "BIOG_MAIN", CurrentProject.Connection, adOpenDynamic, _
    adLockOptimistic
    rst.Find "c_personid = " & c_kin_id.Value

    TxtKinNM.Value = rst.Fields("c_name")
    TxtKinNM_CHN.Value = rst.Fields("c_name_chn")

    rst.Close
End If

```

End Sub

```

Private Sub CmdDelete_Click()
On Error GoTo Err_CmdDelete_Click

```

```

    TxtDelPersonID.Value = c_personid
    TxtDelKinID.Value = c_kin_id
    TxtDelKinCode.Value = c_kin_code
    TxtDelSource.Value = c_source
    TxtDelPages.Value = c_pages
    TxtDelNotes.Value = c_notes

```

```

    DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
    DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70

```

Exit_CmdDelete_Click:

```
Exit Sub
```

Err_CmdDelete_Click:

```

    MsgBox Err.Description
    Resume Exit_CmdDelete_Click

```

End Sub

```

Private Sub CmdAddNew_Click()
On Error GoTo Err_CmdAddNew_Click

```

```
DoCmd.GoToRecord , , acNewRec
```

Exit_CmdAddNew_Click:

```
Exit Sub
```

Err_CmdAddNew_Click:

```

    MsgBox Err.Description
    Resume Exit_CmdAddNew_Click

```

End Sub

```

Private Sub CmdAutoGen_Click()
On Error GoTo Err_CmdAutoGen_Click

```

```
Dim rst As ADODB.Recordset
```



```

Set rst = New ADODB.Recordset

Dim intKinCode As Integer
Dim intKinPair As Integer
Dim intPerson As Long
Dim tGender As Integer
Dim intKin As Long
Dim blnRecordAdded As Boolean

Dim intKinOld As Long
Dim intKinPairOld As Integer
Dim tGenderOld As Integer
Dim intPersonFind As Long
Dim intKinCodeFind As Integer

intPerson = c_personid.Value
intKin = c_kin_id.Value
intKinCode = c_kin_code.Value

'Determine the gender of the kin
rst.Open "BIOG_MAIN", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.Find "c_personid = " & intPerson
tGender = rst.Fields("c_female")
rst.Close

'Find the pair-code
rst.Open "KINSHIP_CODES", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.Find "c_kincode = " & c_kin_code.Value

If tGender = 0 Then
    intKinPair = rst.Fields("c_kin_pair1")
Else
    intKinPair = rst.Fields("c_kin_pair2")
End If

rst.Close

'Add the new entry
rst.Open "KIN_DATA", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.AddNew
rst!c_personid = intKin
rst!c_kin_id = intPerson
rst!c_kin_code = intKinPair

rst!c_notes = c_notes
rst!c_source = c_source
rst!c_pages = c_pages
rst!c_autogen_fr_id = intPerson
rst!c_autogen_fr_code = intKinCode
rst!c_autogen_notes = "Auto-generated from PersonID = " & Trim(intPerson) & ", KinCode = " & Trim(intKinC
ode) & "."

rst.Update
blnRecordAdded = True

rst.Close

Exit_CmdAutoGen_Click:
Exit Sub

Err_CmdAutoGen_Click:
MsgBox Err.Description
Resume Exit_CmdAutoGen_Click

End Sub

Public Sub changeDisplayLanguage()
Dim tLabelLanguage(3, 8) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False
With tRstLabelList

```

```

.MoveFirst
ti = 1

Do While ti < 8 And Not .EOF
    If !c_form = "SF_KIN" Then
        gLabelsOK = True
        If ti <> !c_label_id Then
            MsgBox "Uh oh: mismatched label table"
            gLabelsOK = False
            Exit Do
        End If
        tLabelLanguage(1, ti) = !c_english
        tLabelLanguage(2, ti) = !c_fanti
        tLabelLanguage(3, ti) = !c_jianti
        ti = ti + 1
    End If
    .MoveNext
Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If gLabelsOK Then
    If gDisplayLanguage = "E" Then
        tLang = 1
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '
    Me.LblPages.Caption = tLabelLanguage(tLang, 1)
    Me.LblNotes.Caption = tLabelLanguage(tLang, 2)
    Me.CmdPickKinRel.Caption = tLabelLanguage(tLang, 3)
    Me.CmdPickKinID.Caption = tLabelLanguage(tLang, 4)
    Me.CmdPickSource.Caption = tLabelLanguage(tLang, 5)
    Me.CmdAddNew.Caption = tLabelLanguage(tLang, 6)
    Me.CmdDelete.Caption = tLabelLanguage(tLang, 7)
End If

End Sub

Public Sub noEdits()

    Me.AllowAdditions = False
    Me.AllowDeletions = False
    Me.AllowEdits = False

End Sub

```

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 12) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 12 And Not .EOF

If !c_form = "SF_KIN" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

Me.LblPages.Caption = tLabelLanguage(tLang, 1)

Me.LblNotes.Caption = tLabelLanguage(tLang, 2)

Me.LblKinRel.Caption = tLabelLanguage(tLang, 3)

Me.LblKinName.Caption = tLabelLanguage(tLang, 4)

Me.LblSource.Caption = tLabelLanguage(tLang, 5)

' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 6)

' Me.CmdDelete.Caption = tLabelLanguage(tLang, 7)

Me.Lbl_c_up.Caption = tLabelLanguage(tLang, 8)

Me.Lbl_c_down.Caption = tLabelLanguage(tLang, 9)

Me.Lbl_c_collateral.Caption = tLabelLanguage(tLang, 10)

Me.Lbl_c_marriage.Caption = tLabelLanguage(tLang, 11)

End If

End Sub

Public Sub noEdits()

Me.AllowAdditions = False

Me.AllowDeletions = False

Me.AllowEdits = False

End Sub

frm_LookAtAssociationPairs -1

Option Compare Database

Public gRstPeople As DAO.Recordset, gDisplayLanguage As String, gLabelsOK As Boolean

Public gImportPlaces As Boolean, gUseADDRID As Boolean, gRstEdge As DAO.Recordset

Public gRst As DAO.Recordset

Public gFromDynasty As Integer, gToDynasty As Integer, gUseIndexYears As Boolean, gToYear As String, gFromYear As String, gUseDynasties As Boolean, _

gFromDynastyBegin As Integer, gFromDynastyEnd As Integer, gToDynastyBegin As Integer, gToDynastyEnd As Integer, gFromStr As String, gToStr As String

Private Sub ChkIndexYears_Click()

Me.TxtFromYear.Enabled = ChkIndexYears.Value

Me.TxtToYear.Enabled = ChkIndexYears.Value

End Sub

Private Sub CmdAllDynasties_Click()

gFromDynasty = -2

gToDynasty = -2

TxtFromDynasty.Value = ""

TxtFromDynastyPY.Value = "All"

TxtToDynasty.Value = ""

TxtToDynastyPY.Value = "All"

End Sub

Private Sub CmdFromDynasty_Click()

Dim stDocName As String

Dim stLinkCriteria As String

Dim strFromDynasty As String

If gFromDynasty < 0 Then

strFromDynasty = ""

Else

strFromDynasty = Str(gFromDynasty)

End If

stDocName = "frmPickDynasty"

DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus

gFromDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus

gFromDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus

gFromDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value

'

' check to see if we have a problem and reject selection

'

If gToDynasty > -1 Then

If gFromDynastyBegin > gToDynastyEnd Then

MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation

gFromDynasty = -1

TxtFromDynasty.Value = ""

TxtFromDynastyPY.Value = ""

End If

End If

'

' value is OK

'

If gFromDynasty > -1 Then

Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus

TxtFromDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus

TxtFromDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value

End If

DoCmd.Close acForm, stDocName

'

' reset ToDynasty if necessary (-2 = all dynasties)

'

If gToDynasty = -2 Then

gToDynasty = -1

```

        TxtToDynasty.Value = ""
        TxtToDynastyPY.Value = ""
    End If
    '
End If

End Sub

Private Sub CmdGephi_Click()
    On Error GoTo Err_CmdGephi_Click
    '
    ' This program will dump the results of the search to a .gdf file
    '
    ' for the moment I'll just describe the format of the .gdf file
    '
    ' nodedef> name, label, labelvisible, style, pinyin VARCHAR(50), nodedist INT
    '     name = str(c_person_id)
    '     label = c_name_chn
    '     style = 4 (text inside a rectangle)
    '     pinyin = c_name
    '     nodedist = c_node_dist INT
    '     indexyear = c_index_year INT
    '     sex = c_female > (F,M)
    '
    ' edgedef> node1, node2, label, labelvisible, edge_desc VARCHAR(50)
    '     node1 = str(c_person_id) for node1
    '     node2 = str(c_node_id) for node2
    '     label = c_link_chn
    '     edge_desc = c_link_desc
    '     edgetype= c_link_type (K,N)
    '
    ' the central question is whether to do distance optimizations
    '
    ' first see if there are any records to process
    '
    If ZZ_SOCIAL_NETWORK.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_CmdGephi_Click
    End If
    '
    If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_CmdGephi_Click
    End If
    '
    ' next get a file
    '
    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tRstNode As DAO.Recordset
    Dim tRstEdge As DAO.Recordset
    Dim tStr As String, tC As String, ti As Integer, tCodeStr As String
    'Dim tFileSystem, tGDF

    ' set up the stream for writing
    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream
    If CodeFrame.Value = 1 Then
        tStream.Charset = "utf-8"
        tCodeStr = "UTF8"
    ElseIf CodeFrame.Value = 2 Then
        tStream.Charset = "big5"
        tCodeStr = "BIG5"
    ElseIf CodeFrame.Value = 3 Then
        tStream.Charset = "gb18030"
        tCodeStr = "GB18030"
    Else
        tStream.Charset = "ascii"
        tCodeStr = "ASCII"
    End If

    Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    'Use a With...End With block to reference the FileDialog object.
    With dlgSaveAs
        .InitialFileName = "assoc_links_" + tCodeStr + ".gdf"
        If .Show = -1 Then
            '
            tFileName = ""

```

```

For Each tFN In .SelectedItems
    tFileName = tFN
    If Not tFileName = "" Then
        Exit For
    End If
Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdGephi_Click
Else
    ' make sure the file name has a gdf extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".gdf"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".gdf") Then
        tFileName = tFileName + ".gdf"
    End If
End If
'
' now process the file (second true removed to make ASCII)
'
'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
'Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)

' process the two tables
'
Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
tC = Chr(44) ' the comma
tQuote = Chr(34) 'the Quote delimiter
'
' ready to go: now open the stream
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' first the nodes: define the record structure
' if ASCII, no characters, no pinyin
'
If tCodeStr = "ASCII" Then
    tStr = "nodedef> name VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible BOOLEAN" + _
        tC + "style INT" + tC + "indexyear INT" + tC + "sex VARCHAR(1)" + _
        tC + "addr_name VARCHAR" + tC + "latitude DOUBLE" + tC + "longitude DOUBLE"
Else
    tStr = "nodedef> name VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible BOOLEAN" + _
        tC + "style INT" + tC + "pinyin VARCHAR(50)" + tC + "indexyear INT" + tC + "sex VARCHAR(1)" + _
        tC + "addr_chn VARCHAR" + tC + "addr_name VARCHAR" + tC + "latitude DOUBLE" + tC + "longitude
DOUBLE"
End If
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' name = the ID of the person
        tStr = Trim(Str(!c_person_id)) + tC

        ' label
        If tCodeStr = "ASCII" Then
            If IsNull(!c_name) Then
                tStr = tStr + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        Else
            If IsNull(!c_name_chn) Then
                tStr = tStr + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If
        End If

        ' labelvisible = true, style = 4 (text inside a rectangle)
        tStr = tStr + "true" + tC + "4" + tC

        If Not (tCodeStr = "ASCII") Then
            ' pinyin = c_name
            tStr = tStr + !c_name + tC
        End If
    End While
End With

```

```

'   indexyear = c_index_year INT
If IsNull(!c_index_year) Then
    tStr = tStr + "-2000" + tC
Else
    tStr = tStr + Trim(Str(!c_index_year)) + tC
End If

'   sex = F,M
tStr = tStr + IIf(!c_female, "F", "M") + tC

If tCodeStr = "ASCII" Then
    If IsNull(!c_addr_name) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If
Else
    If IsNull(!c_addr_chn) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_chn + tC
    End If
    '
    If IsNull(!c_addr_name) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If
End If
'   latitude = !y_coord
If IsNull(!y_coord) Then
    tStr = tStr + "0.0" + tC
Else
    tStr = tStr + Str(!y_coord) + tC
End If

'   longitude = !x_coord
If IsNull(!x_coord) Then
    tStr = tStr + "0.0"
Else
    tStr = tStr + Str(!x_coord)
End If
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
.MoveNext
Loop
End With
'
' now the edges:  define the record structure
tStr = "edgedef> node1 VARCHAR" + tC + "node2 VARCHAR" + tC + "label VARCHAR" + tC + "edge_desc VARCH
AR(50)" + _
        tC + "link_count INT"
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)

With tRstEdge
.MoveNext
Do While Not .EOF
    '   node1 = str(c_person_id) for node1
    tStr = Trim(Str(!c_person_id)) + tC

    '   node2 = str(c_node_id) for node2
    tStr = tStr + Trim(Str(!c_node_id)) + tC

    '   label = c_link_chn
    If IsNull(!c_link_chn) Then
        tStr = tStr + tC
    Else
        tStr = tStr + tQuote + !c_link_chn + tQuote + tC
    End If

    '   edge_desc = c_link_desc
    If IsNull(!c_link_desc) Then
        tStr = tStr + tC
    Else
        tStr = tStr + tQuote + Trim(Left(!c_link_desc + Space(50), 50)) + tQuote + tC
    End If

    '   link_count = c_link_count

```

```

        tStr = tStr + Str(!c_link_count)

        tStream.WriteText tStr, adWriteLine
        'tGDF.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tStream.Close
Set tStream = Nothing
'tGDF.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
'Set tGDF = Nothing
'Set tFileSystem = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGephi_Click:
Exit Sub

Err_CmdGephi_Click:
MsgBox Err.Description
Resume Exit_CmdGephi_Click

End Sub

Private Sub CmdNeo4j_Click()
On Error GoTo Err_CmdNeo4j_Click
'
' This routine will be close to that for LookAtAssociations and, if used, that for LookAtKinship
' The additional wrinkle is that, while the first step is to split associatin from kinship relations, we sti
ll need to gather all
' the people from both.
'
' first see if there are any records to process
'
If Me.ZZ_SOCIAL_NETWORK.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdNeo4j_Click
End If
'
' allocate the file variables
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
'
' next get the People file
'
Dim tRstPeople As DAO.Recordset, tRstAssoc As DAO.Recordset, tRstPlace As DAO.Recordset, tRstPeoplePlace As DAO.Recordset
Dim tStr As String, tC As String, tQueryStr As String, tRstAssocCode As DAO.Recordset, tRstKin As DAO.Recordset
Dim gStream As ADODB.Stream, tCodeStr As String, tTempLong As Long
'
' set up the stream to write to

Set gStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
    gStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf CodeFrame.Value = 2 Then
    gStream.Charset = "big5"
    tCodeStr = "BIG5"
ElseIf CodeFrame.Value = 3 Then
    gStream.Charset = "gb2312"
    tCodeStr = "GB2312"

```



```

Else
    gStream.Charset = "ascii"
    tCodeStr = "ascii"
End If

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
    'Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)
    '
    ' we have a file name: now open the stream for writing

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    '
    ' prepare the temp tables for the people, place, peoplePlace and assoc codes

    Dim cmdSQL As ADODB.Command
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' Get the people from 5 sources: c_person_id, c_node_id, c_kin_id, c_assoc_kin_id, and c_assoc_claim
er_id

    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_TEXT"
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (1)
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person
_id " + _
        "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_person_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (2)
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_node_i
d " + _
        "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_node_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (3)
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_kin_id
" + _
        "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_kin_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted

```

```

'
'   copy the data for people (4)
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_assoc_
kin_id " + _
            "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_assoc_kin_id)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'   copy the data for people (5)
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_assoc_
claimer_id " + _
            "FROM ZZ_SOCIAL_NETWORK WHERE (((ZZ_SOCIAL_NETWORK.c_assoc_claimer_id)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_nam
e_chn, ZZZ_BIOG_MAIN.c_index_year, " + _
            "ZZZ_BIOG_MAIN.c_dynasty, ZZZ_BIOG_MAIN.c_dynasty_chn, ZZZ_BIOG_MAIN.c_index_addr_id,
ZZZ_BIOG_MAIN.c_index_addr_name, " + _
            "ZZZ_BIOG_MAIN.c_index_addr_chn, ZZZ_BIOG_MAIN.c_index_addr_type_code, ZZZ_BIOG_MAIN.
c_index_addr_type_desc, " + _
            "ZZZ_BIOG_MAIN.c_index_addr_type_chn, ZZZ_BIOG_MAIN.c_female, ZZZ_BIOG_MAIN.x_coord,
ZZZ_BIOG_MAIN.y_coord " + _
            "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_B
IOG_MAIN.c_personid"
'
Set tRstPeople = CurrentDb.OpenRecordset(tQueryStr, dbOpenDynaset)
'
' process the four tables
'
tC = Chr(44) ' the comma
'
' first the nodes:  define the record structure
'
'   if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a
pinyin field
If tCodeStr = "ascii" Then
    tStr = "nameID" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
Else
    tStr = "nameID" + tC + "nameHZ" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
End If
gStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
'
With tRstPeople
    .MoveFirst
    Do While Not .EOF
        '   the ID of the person
        tStr = Trim(Str(!c_person_id)) + tC
        '
        '   name
        '
        If tCodeStr = "ascii" Then
            If IsNull(!c_name) Then
                tStr = tStr + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        Else
            If IsNull(!c_name_chn) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If

            If IsNull(!c_name) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        End If

        '   indexyear = c_index_year INT
        '
        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        End If
    End While
End With

```

```

Else
    tStr = tStr + Trim(Str(!c_index_year)) + tC
End If

' dynasty information
'
If IsNull(!c_dynasty) Then
    tStr = tStr + "unknown" + tC
Else
    If tCodeStr = "ascii" Then
        tStr = tStr + !c_dynasty + tC
    Else
        tStr = tStr + !c_dynasty_chn + tC
    End If
End If

'
' sex = c_female > (F,M)
tStr = tStr + IIf(!c_female, "F", "M")
'
gStream.WriteText tStr, adWriteLine
'
.MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If

' now places: since the association "event" is not linked to a place, the only addresses are the index
addresses
'
' of the people involved, recorded in ZZ_SCRATCH_PEOPLE
'
' get a file name
'
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    ' now process the file
    '
    tQueryStr = "SELECT DISTINCT ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_name, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_chn, ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord " + _
        "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_B
IOG_MAIN.c_personid " + _
        "WHERE (ZZZ_BIOG_MAIN.c_index_addr_id > 0)"

    Set tRstPlace = CurrentDb.OpenRecordset(tQueryStr)
    '
    If tCodeStr = "ascii" Then
        tStr = "placeID" + tC + "placePY" + tC + "placeX" + tC + "placeY"
    Else
        tStr = "placeID" + tC + "placePY" + tC + "placeHZ" + tC + "placeX" + tC + "placeY"
    End If

```

```

gStream.WriteText tStr, adWriteLine
With tRstPlace
    .MoveFirst
    Do While Not .EOF
        ' the ID of the place
        If Not IsNull(!c_index_addr_id) Then
            tStr = Trim(Str(!c_index_addr_id)) + tC
            ' address name

            If IsNull(!c_index_addr_name) Then
                tStr = tStr + "unknown" + tC
            Else
                tStr = tStr + !c_index_addr_name + tC
            End If
            '
            If Not (tCodeStr = "ascii") Then
                If IsNull(!c_index_addr_chn) Then
                    tStr = tStr + "unknown" + tC
                Else
                    tStr = tStr + !c_index_addr_chn + tC
                End If
            End If

            ' longitude = !x_coord
            If IsNull(!x_coord) Then
                tStr = tStr + "0.0" + tC
            Else
                tStr = tStr + Str(!x_coord) + tC
            End If

            ' latitude = !y_coord
            If IsNull(!y_coord) Then
                tStr = tStr + "0.0"
            Else
                tStr = tStr + Str(!y_coord)
            End If
            '
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces
'
dlgSaveAs.InitialFileName = "PeoplePlaces_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
End If
'
gStream.Open

```

```

    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_M
AIN.c_index_addr_type_code, " + _
                "ZZZ_BIOG_MAIN.c_index_addr_type_desc, ZZZ_BIOG_MAIN.c_index_addr_type_chn " + _
                "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_B
IOG_MAIN.c_personid " + _
                "WHERE (ZZZ_BIOG_MAIN.c_index_addr_id > 0)"

    Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

    If (tCodeStr = "ascii") Then
        tStr = "nameID" + tC + "placeID" + tC + "personPlaceCode" + tC + "personPlaceTrans"
    Else
        tStr = "nameID" + tC + "placeID" + tC + "personPlaceCode" + tC + "personPlaceTrans" + tC + "perso
nPlaceHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstPeoplePlace
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_index_addr_id) Then
                '
                tStr = Trim(Str(!c_person_id)) + tC
                '
                tStr = tStr + Trim(Str(!c_index_addr_id)) + tC
                '
                tStr = tStr + Trim(Str(!c_index_addr_type_code)) + tC
                '
                tStr = tStr + Trim(Str(!c_index_addr_type_desc))
                '
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(Str(!c_index_addr_type_chn))
                End If
                '
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With

    '
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now the association records
'
dlgSaveAs.InitialFileName = "PeopleAssociations_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    ' now the associations: define the record structure

```

```
' Because of the complexity of the primary key, this gets a bit complicated
```

```
Set tRstAssoc = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
```

```
tStr = "Person1_ID" + tC + "Person2_ID" + tC + "Association_Code" + tC + "Kin_ID" + tC + "Kin_Code" +  
tC + _  
"AssocKin_ID" + tC + "AssocKin_Code" + tC + "LiteraryGenreCode" + tC + "OccasionCode" + tC +  
"TopicCode" + tC + "InstitutionCode" + tC + "TextTitle" + tC + "AssociationClaimer_ID"
```

```
gStream.WriteText tStr, adWriteLine
```

```
'tGDF.WriteLine (tStr)
```

```
With tRstAssoc
```

```
.MoveFirst
```

```
Do While Not .EOF
```

```
    If !c_link_type = "N" And (Not IsNull(!c_link_code)) Then
```

```
        tStr = Trim(Str(!c_person_id)) + tC
```

```
        ' node1 = str(c_person_id) for node1
```

```
        tStr = tStr + Trim(Str(!c_node_id)) + tC
```

```
        ' node2 = str(c_node_id) for node2
```

```
        tStr = tStr + Trim(Str(!c_link_code)) + tC
```

```
        ' kin ID
```

```
        If IsNull(!c_kin_id) Then
```

```
            tStr = tStr + "0" + tC
```

```
        Else
```

```
            tStr = tStr + Str(!c_kin_id) + tC
```

```
        End If
```

```
        ' kin code
```

```
        If IsNull(!c_kin_code) Then
```

```
            tStr = tStr + "0" + tC
```

```
        Else
```

```
            tStr = tStr + Str(!c_kin_code) + tC
```

```
        End If
```

```
        ' assoc kin ID
```

```
        If IsNull(!c_assoc_kin_id) Then
```

```
            tStr = tStr + "0" + tC
```

```
        Else
```

```
            tStr = tStr + Str(!c_assoc_kin_id) + tC
```

```
        End If
```

```
        ' assoc kin code
```

```
        If IsNull(!c_assoc_kin_code) Then
```

```
            tStr = tStr + "0" + tC
```

```
        Else
```

```
            tStr = tStr + Str(!c_assoc_kin_code) + tC
```

```
        End If
```

```
        ' literary genre code
```

```
        If IsNull(!c_litgenre_code) Then
```

```
            tStr = tStr + "0" + tC
```

```
        Else
```

```
            tStr = tStr + Str(!c_litgenre_code) + tC
```

```
        End If
```

```
        ' occasion code
```

```
        If IsNull(!c_occasion_code) Then
```

```
            tStr = tStr + "0" + tC
```

```
        Else
```

```
            tStr = tStr + Str(!c_occasion_code) + tC
```

```
        End If
```

```
        ' topic code
```

```
        If IsNull(!c_topic_code) Then
```

```
            tStr = tStr + "0" + tC
```

```
        Else
```

```
            tStr = tStr + Str(!c_topic_code) + tC
```

```
        End If
```

```
        ' institution code
```

```
        If IsNull(!c_inst_code) Then
```

```
            tStr = tStr + "0" + tC
```

```
        Else
```

```
            tStr = tStr + Str(!c_inst_code) + tC
```

```
        End If
```

```
        ' text title
```

```
        If IsNull(!c_text_title) Then
```

```

        tStr = tStr + "N/A" + tC
    Else
        tStr = tStr + !c_text_title + tC
    End If

    ' association claimer ID
    If IsNull(!c_assoc_claimer_id) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Str(!c_assoc_claimer_id)
    End If

    gStream.WriteText tStr, adWriteLine
End If
.MoveNext

Loop
End With

' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
gStream.Close
Else
    'The user pressed Cancel.
End If

' now the kinship relations: first, will there be any?
If ChkKinship.Value Then
    cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP"
    cmdSQL.Execute tRecDeleted

    tQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_kin_code, c_kinrel, c_kinrel_total ) " +
        "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_link_code, ZZ_SOCIAL_NETWORK.c_link_desc, ZZ_SOCIAL_NETWORK.c_link_chn " +
        "FROM ZZ_SOCIAL_NETWORK " +
        "WHERE (((ZZ_SOCIAL_NETWORK.c_link_type)='K') AND ((ZZ_SOCIAL_NETWORK.c_link_code)>0))"

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted

    If tRecDeleted > 0 Then
        dlgSaveAs.InitialFileName = "KinshipRelations_" + tCodeStr + ".csv"
        If dlgSaveAs.Show = -1 Then
            tFileName = ""
            For Each tFN In dlgSaveAs.SelectedItems
                tFileName = tFN
                If Not tFileName = "" Then
                    Exit For
                End If
            Next
            If tFileName = "" Then
                MsgBox "Bad file Name."
                GoTo Exit_CmdNeo4j_Click
            Else
                ' make sure the file name has a txt extension
                If Len(tFileName) < 5 Then
                    tFileName = tFileName + ".csv"
                ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                    tFileName = tFileName + ".csv"
                End If
            End If

            gStream.Open

            tStr = "PersonID" + tC + "KinID" + tC + "KinCode"

            gStream.WriteText tStr, adWriteLine

            ' still using ZZ_SOCIAL_NETWORK

            With tRstAssoc
                .MoveFirst
                Do While Not .EOF
                    If !c_link_type = "K" And (Not IsNull(!c_link_code)) Then
                        tStr = Trim(Str(!c person id)) + tC
                    End If
                Loop
            End With
        End If
    End If
End Sub

```

```

        ,
        tStr = Trim(Str(!c_node_id)) + tC
        ,
        tStr = Trim(Str(!c_link_code))
        ,
        gStream.WriteText tStr, adWriteLine
    End If
    .MoveNext
    Loop
End With
,
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
,
' now the association codes
,
dlgSaveAs.InitialFileName = "AssociationCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_link_code, ZZ_SOCIAL_NETWORK.c_link_desc, ZZ_SOCIAL_
NETWORK.c_link_chn " + _
        "FROM ZZ_SOCIAL_NETWORK " + _
        "WHERE ((ZZ_SOCIAL_NETWORK.c_link_type = 'N') and (ZZ_SOCIAL_NETWORK.c_link_code > 0))"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr, dbOpenDynaset)
    ,
    If tCodeStr = "ascii" Then
        tStr = "AssociationCode" + tC + "AssociationTrans"
    Else
        tStr = "AssociationCode" + tC + "AssociationTrans" + tC + "AssociationHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_link_code) Then
                ,
                tStr = Trim(Str(!c_link_code)) + tC
                ,
                tStr = tStr + Trim(!c_link_desc)
                ,
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_link_chn)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        End While
    End With
End With

```



```

        Loop
    End With
    '
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' there are codes that MAY require additional tables: c_kin_code, c_litgenrte_code, c_occasion_code, c_t
opic_code, c_inst_code
'
' test for kin codes
'
' tRecDeleted is the number of kinship codes inserted into ZZ_KIN_LIST_TEMP for the earlier test
'
tTempLong = tRecDeleted
tQueryStr = "INSERT INTO ZZ_KIN_LIST_TEMP ( c_kin_code, c_kinrel, c_kinrel_total ) " + _
"SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_kin_code, ZZ_SOCIAL_NETWORK.c_kin_desc, ZZ_SOCIAL_NETWORK
K.c_kin_desc_chn " + _
"FROM ZZ_SOCIAL_NETWORK " + _
"WHERE (((ZZ_SOCIAL_NETWORK.c_kin_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

tTempLong = tTempLong + tRecDeleted
tQueryStr = "INSERT INTO ZZ_KIN_LIST_TEMP ( c_kin_code, c_kinrel, c_kinrel_total ) " + _
"SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_assoc_kin_code, ZZ_SOCIAL_NETWORK.c_assoc_kin_desc, ZZ_S
SOCIAL_NETWORK.c_assoc_kin_desc_chn " + _
"FROM ZZ_SOCIAL_NETWORK " + _
"WHERE (((ZZ_SOCIAL_NETWORK.c_assoc_kin_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

tTempLong = tTempLong + tRecDeleted
'
' debug
'
MsgBox "Kinship code records = " + Trim(Str(tTempLong))
'
If tTempLong > 0 Then
    dlgSaveAs.InitialFileName = "KinshipCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_KIN_LIST_TEMP.c_kin_code, ZZ_KIN_LIST_TEMP.c_kinrel, ZZ_KIN_LIST_TM
P.c_kinrel_total " + _
"FROM ZZ_KIN_LIST_TEMP"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "KinshipCode" + tC + "KinshipTrans"
    End If
End If

```

```

Else
    tStr = "KinshipCode" + tC + "KinshipTrans" + tC + "KinshipHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstAssocCode
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_kin_code) Then
            ,
            tStr = Trim(Str(!c_kin_code)) + tC
            ,
            tStr = tStr + Trim(!c_kinrel)
            ,
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + Trim(!c_kinrel_total)
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
,
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
,
' test for literary genre codes
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
"SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id " + _
"FROM ZZ_SOCIAL_NETWORK " + _
"WHERE (((ZZ_SOCIAL_NETWORK.c_litgenre_code)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' debug
,
MsgBox "Literary genre code records = " + Trim(Str(tRecDeleted))
,
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "LiteraryGenreCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
        ,
        gStream.Open
        ,
        tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_litgenre_code, ZZ_SOCIAL_NETWORK.c_litgenre_desc
, ZZ_SOCIAL_NETWORK.c_litgenre_desc chn " + _
"FROM ZZ_SOCIAL_NETWORK " + _
"WHERE (((ZZ_SOCIAL_NETWORK.c_litgenre_code)>0))"

```

```

Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "LitGenreCode" + tC + "LitGenreTrans"
Else
    tStr = "LitGenreCode" + tC + "LitGenreTrans" + tC + "LitGenreHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstAssocCode
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_litgenre_code) Then
            tStr = Trim(Str(!c_litgenre_code)) + tC
            tStr = tStr + Trim(!c_litgenre_desc)
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + Trim(!c_litgenre_desc_chn)
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With

' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If

' test for institution codes
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
"SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id " + _
"FROM ZZ_SOCIAL_NETWORK " + _
"WHERE ((ZZ_SOCIAL_NETWORK.c_inst_code)>0))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

' debug
MsgBox "Institution code records = " + Trim(Str(tRecDeleted))

If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "InstitutionCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
        gStream.Open
        tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_inst_code, ZZ_SOCIAL_NETWORK.c_inst_name_py, ZZ

```

```

SOCIAL_NETWORK.c_inst_name_hz " +
    "FROM ZZ_SOCIAL_NETWORK " +
    "WHERE ((ZZ_SOCIAL_NETWORK.c_inst_code)>0))"

Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "InstitutionCode" + tC + "InstitutionNamePY"
Else
    tStr = "InstitutionCode" + tC + "InstitutionNamePY" + tC + "InstitutionNameHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstAssocCode
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_inst_code) Then
            ,
            tStr = Trim(Str(!c_inst_code)) + tC
            ,
            tStr = tStr + Trim(!c_inst_name_py)
            ,
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + Trim(!c_inst_name_hz)
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
,
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
,
' test for occasion codes
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
    "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id " + _
    "FROM ZZ_SOCIAL_NETWORK " + _
    "WHERE ((ZZ_SOCIAL_NETWORK.c_occasion_code)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' debug
,
MsgBox "Occasion code records = " + Trim(Str(tRecDeleted))
,
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "OccasionCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
    End If
End If

```

```

    ,
    gStream.Open
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_occasion_code, ZZ_SOCIAL_NETWORK.c_occasion_desc
, ZZ_SOCIAL_NETWORK.c_occasion_desc_chn " + _
                "FROM ZZ_SOCIAL_NETWORK " + _
                "WHERE (((ZZ_SOCIAL_NETWORK.c_occasion_code)>0))"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "OccasionCode" + tC + "OccasionTrans"
    Else
        tStr = "OccasionCode" + tC + "OccasionTrans" + tC + "OccasionHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_occasion_code) Then
                ,
                tStr = Trim(Str(!c_occasion_code)) + tC
                ,
                tStr = tStr + Trim(!c_occasion_desc)
                ,
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_occasion_desc_chn)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    ,
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
,
' test for topic codes
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
            "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id " + _
            "FROM ZZ_SOCIAL_NETWORK " + _
            "WHERE (((ZZ_SOCIAL_NETWORK.c_topic_code)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' debug
,
MsgBox "Topic code records = " + Trim(Str(tRecDeleted))
,
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "TopicCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            End If
        End If
    End If
End If

```

```

        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    ' because ZZZ_NONKIN_BIOG_ADDR does not have the topic descriptions (must fix), I need to use a j
    oin to TOPIC_CODES
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_topic_code, SCHOLARLYTOPIC_CODES.c_topic_desc, S
    CHOLARLYTOPIC_CODES.c_topic_desc_chn " + _
        "FROM ZZ_SOCIAL_NETWORK INNER JOIN SCHOLARLYTOPIC_CODES ON ZZ_SOCIAL_NETWORK.c_topic
    code = SCHOLARLYTOPIC_CODES.c_topic_code " + _
        "WHERE (((ZZ_SOCIAL_NETWORK.c_topic_code)>0))"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "TopicCode" + tC + "TopicTrans"
    Else
        tStr = "TopicCode" + tC + "TopicTrans" + tC + "TopicHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_topic_code) Then
                ,
                tStr = Trim(Str(!c_topic_code)) + tC
                ,
                tStr = tStr + Trim(!c_topic_desc)
                ,
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_topic_desc_chn)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    ,
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
End If
MsgBox "Finished saving to Neo4j"

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click
End Sub

Private Sub CmdPickPerson1_Click()
On Error GoTo Err_CmdPickPerson1_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strPERSON_ID As String

TxtID1.Visible = True
TxtID1.SetFocus
strPERSON_ID = TxtID1.Text

```

```

stDocName = "frmSelectPerson"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strPERSON_ID

If CurrentProject.AllForms("frmSelectPerson").IsLoaded Then
    Dim lngPERSON_ID As Long
    Dim strPERSON_NM As String
    Dim strPERSON_NM_CHN As String

    Forms!frmSelectPerson!frmPersonSearch.Form!c_personid.SetFocus
    lngPERSON_ID = Forms!frmSelectPerson!frmPersonSearch.Form!c_personid.Value
    TxtID1.Value = lngPERSON_ID

    Forms!frmSelectPerson!frmPersonSearch.Form!c_name_chn.SetFocus
    strPERSON_NM_CHN = Forms!frmSelectPerson!frmPersonSearch.Form!c_name_chn.Value
    TxtPerson1Chn.Value = strPERSON_NM_CHN

    Forms!frmSelectPerson!frmPersonSearch.Form!c_name.SetFocus
    strPERSON_NM = Forms!frmSelectPerson!frmPersonSearch.Form!c_name.Value
    TxtPerson1.Value = strPERSON_NM

    DoCmd.Close acForm, stDocName
    '
    ' now enable the Run button
    '
    If TxtID2.Value > 0 Then
        CmdQuery.Enabled = True
    End If
End If

CmdPickPerson1.SetFocus
TxtID1.Visible = False

Exit_CmdPickPerson1_Click:
Exit Sub

Err_CmdPickPerson1_Click:
MsgBox Err.Description
Resume Exit_CmdPickPerson1_Click

End Sub

Private Sub CmdPickPerson2_Click()
On Error GoTo Err_CmdPickPerson2_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strPERSON_ID As String

TxtID2.Visible = True
TxtID2.SetFocus
strPERSON_ID = TxtID2.Text

stDocName = "frmSelectPerson"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strPERSON_ID

If CurrentProject.AllForms("frmSelectPerson").IsLoaded Then
    Dim lngPERSON_ID As Long
    Dim strPERSON_NM As String
    Dim strPERSON_NM_CHN As String

    Forms!frmSelectPerson!frmPersonSearch.Form!c_personid.SetFocus
    lngPERSON_ID = Forms!frmSelectPerson!frmPersonSearch.Form!c_personid.Value
    TxtID2.Value = lngPERSON_ID

    Forms!frmSelectPerson!frmPersonSearch.Form!c_name_chn.SetFocus
    strPERSON_NM_CHN = Forms!frmSelectPerson!frmPersonSearch.Form!c_name_chn.Value
    TxtPerson2Chn.Value = strPERSON_NM_CHN

    Forms!frmSelectPerson!frmPersonSearch.Form!c_name.SetFocus
    strPERSON_NM = Forms!frmSelectPerson!frmPersonSearch.Form!c_name.Value
    TxtPerson2.Value = strPERSON_NM

    DoCmd.Close acForm, stDocName
    '
    ' now enable the Run button
    '
    If TxtID1.Value > 0 Then
        CmdQuery.Enabled = True
    End If
End If

```

```

CmdPickPerson2.SetFocus
TxtID2.Visible = False

Exit_CmdPickPerson2_Click:
Exit Sub

Err_CmdPickPerson2_Click:
MsgBox Err.Description
Resume Exit_CmdPickPerson2_Click

End Sub

Private Sub CmdQuery_Click()
On Error GoTo Err_CmdQuery_Click

Dim tRstDummy As DAO.Recordset, tContinue As Integer, tQueryStr As String, tQueryFromStr As String
Dim tQueryWhereStr As String, tQueryAppendStr As String, tQueryIntoStr As String
Dim tQuerySelectStr As String, tQuery1stStr As String, tQuery2ndStr As String
Dim tQuery1stWhereStr As String, tQuery2ndWhereStr As String
Dim tID1Str As String, tID2Str As String, tFromStr As String, tToStr As String
Dim cmdSQL As ADODB.Command, tRecDeleted As Long, tCurADDRBookmark As Variant

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' to clear the table, close and then delete records
'
Set ZZ_SOCIAL_NETWORK.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SN", dbOpenDynaset)
'
cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
' now the people table
'
Set ZZ_SCRATCH_PEOPLE.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SP", dbOpenDynaset)
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted
'
' get the index year constraints
'
If gUseIndexYears Then
'
TxtFromYear.SetFocus
If TxtFromYear.Value = "" Then
gFromStr = ""
Else
gFromStr = Str(TxtFromYear.Value)
End If

TxtToYear.SetFocus
If TxtToYear.Value = "" Then
gToStr = ""
Else
gToStr = Str(TxtToYear.Value)
End If

CmdQuery.SetFocus
'
' four possibilities
'
If gFromStr = "" And gToStr = "" Then
tQueryWhereStr = ""
ElseIf gFromStr = "" Then
tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)<=" + gToStr + ") "
ElseIf gToStr = "" Then
tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)>=" + gFromStr + ") "
Else
tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)<=" + gToStr + ") AND " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_index_year)>=" + gFromStr + ") "
End If
ElseIf tUseDynasties Then
'
' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
'
If gFromDynasty = -2 Then
tQueryWhereStr = "Where (((ZZZ_NONKIN_BIOG_ADDR.c_dy) > 0 ) "
ElseIf gFromDynasty = -1 And gToDynasty > 0 Then

```



```

        tQueryWhereStr = "WHERE (((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_dy) = " + Str(gFromDynasty) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
            "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    Else
        tQueryWhereStr = ""
    End If
End If

' define the query for appending to ZZ_SCRATCH_PEOPLE without duplication
tQueryAppendStr = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_PAIR_PEOPLE.* " + _
    "FROM ZZ_SCRATCH_PAIR_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
    "ZZ_SCRATCH_PAIR_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
    "WHERE (((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null))"

' first add the target people (if CmdClearList is enabled, ImportList was successful)
If CmdClearList.Enabled Then
    ' MsgBox "Using list"
    tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
        "c_female, c_addr_id, c_addr_type, c_addr_desc, c_addr_desc_chn, c_addr_name, c_addr_chn, " + _
        "x_coord, y_coord, c_node_dist ) " + _
        "SELECT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, " + _
        "ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MAIN.c_index_year, " + _
        "ZZZ_BIOG_MAIN.c_female, ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_type_code, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_desc, ZZZ_BIOG_MAIN.c_index_addr_type_chn, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_MAIN.c_index_addr_chn, ZZZ_BIOG_MAIN.x_coord, " + _
        "ZZZ_BIOG_MAIN.y_coord, 0 AS c_node_dist " + _
        "FROM ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZZ_BIOG_MAIN ON " + _
        "ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZZ_BIOG_MAIN.c_personid"
Else
    ' get the ID strings for the two people
    Me.TxtID1.Visible = True
    Me.TxtID1.SetFocus
    tID1Str = Trim(Str(TxtID1.Value))
    Me.TxtID2.Visible = True
    Me.TxtID2.SetFocus
    tID2Str = Trim(Str(TxtID2.Value))
    Me.CmdQuery.SetFocus
    Me.TxtID1.Visible = False
    Me.TxtID2.Visible = False

    ' MsgBox "Using people pairs"
    tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
        "c_female, c_addr_id, c_addr_type, c_addr_desc, c_addr_desc_chn, c_addr_name, c_addr_chn, x_coord, " + _
        "y_coord, c_node_dist ) " + _
        "SELECT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, " + _
        "ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MAIN.c_index_year, ZZZ_BIOG_MAIN.c_female, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_type_code, ZZZ_BIOG_MAIN.c_index_addr_type_desc, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_chn, ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_MAIN.c_index_addr_chn, " + _
        "ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord, 0 AS c_node_dist " + _
        "FROM ZZZ_BIOG_MAIN " + _
        "WHERE (((ZZZ_BIOG_MAIN.c_personid)=" + tID1Str + _
        " Or (ZZZ_BIOG_MAIN.c_personid)=" + tID2Str + "))"
End If

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

' get the first-order linking people
Call Link1stOrder("NONKIN", "NONKIN")

' if kinship ties are to be included, get first-order kinship connections
' The situation is a bit more complicated than it might appear, since a kin of X may be an associate of Y
' or an associate of X may be a kin of Y or a kin of X may be a kin of Y:
'   x a=kin(x)   y=kin(a)   00
'   x a=kin(x)   y=assoc(a) 01
'   x a=assoc(x) y=kin(a)   10

```

```

If ChkKinship.Value Then
    '
    '   kin-kin
    Call Link1stOrder("KIN", "KIN")
    '
    '   kin-assoc
    Call Link1stOrder("KIN", "NONKIN")
    '
    '   assoc-kin
    Call Link1stOrder("NONKIN", "KIN")
End If
'
' now get the second order linking people (first the first person in the pair, then the second)
' Add these to the temp file and copy only the non-duplicates
'
If Me.Chk2Nodes.Value Then
    MsgBox "Warning: the two-node routine takes a while. Don't Panic. Click to start."

    Call Link2ndOrder("NONKIN", "NONKIN", "NONKIN")
    '
    ' if kinship ties are to be included, get second-order kinship connections.
    ' here, there are seven(!) possibilities that must be considered:
    '   x a=kin(x) b=kin(a) y=kin(b)         000
    '   x a=kin(x) b=kin(a) y=assoc(b)      001
    '   x a=kin(x) b=assoc(a) y=kin(b)      010
    '   x a=kin(x) b=assoc(a) y=assoc(b)    011
    '   x a=assoc(x) b=kin(a) y=kin(b)      100
    '   x a=assoc(x) b=kin(a) y=assoc(b)    101
    '   x a=assoc(x) b=assoc(a) y=kin(b)    110
    ' Therefore we need to run seven queries by swapping out the tables we use
    '
    If ChkKinship.Value Then
        '
        '   kin-kin-kin
        Call Link2ndOrder("KIN", "KIN", "KIN")
        '
        '   kin-kin-assoc
        Call Link2ndOrder("KIN", "KIN", "NONKIN")
        '
        '   kin-assoc-kin
        Call Link2ndOrder("KIN", "NONKIN", "KIN")
        '
        '   kin-assoc-assoc
        Call Link2ndOrder("KIN", "NONKIN", "NONKIN")
        '
        '   assoc-kin-kin
        Call Link2ndOrder("NONKIN", "KIN", "KIN")
        '
        '   assoc-kin-assoc
        Call Link2ndOrder("NONKIN", "KIN", "NONKIN")
        '
        '   assoc-assoc-kin
        Call Link2ndOrder("NONKIN", "NONKIN", "KIN")
    End If
End If
'
' remove duplicates
'
tQueryStr = "UPDATE ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN ZZ_SCRATCH_PEOPLE ON " + _
    "ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
    "SET ZZ_SCRATCH_PEOPLE.c_delete = True " + _
    "WHERE ((ZZ_SCRATCH_PEOPLE.c_node_dist)>[ZZ_SCRATCH_PEOPLE_1].[c_node_dist]))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PEOPLE WHERE c_delete = True"
cmdSQL.Execute tRecDeleted
'
' now use the people table to get the edges:
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
' first non-kin
'
tQueryIntoStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year, " + _
    "c_female, c_node_id, c_node_name, c_node_chn, c_node_index_year, c_node_female, c_link_type, " + _
    "c_link_code, c_link_desc, c_link_chn, c_link_count, c_addr_id, c_addr_name, c_addr_chn, " + _

```

```

        "c_addr_type, c_addr_desc, c_addr_desc_chn, x_coord, y_coord, c_node_addr_id, c_node_addr_name, " + _
        "c_node_addr_chn, c_node_addr_type, c_node_addr_desc, c_node_addr_desc_chn, node_xcoord, node_ycoord, " + _
        "c_edge_dist, type_id, c_kin_desc, c_kin_desc_chn, c_kin_name, c_kin_chn, c_kin_id, c_kin_code, " + _
        "c_assoc_kin_desc, c_assoc_kin_desc_chn, c_assoc_kin_name, c_assoc_kin_chn, c_assoc_kin_id, " + _
        "c_assoc_kin_code, c_litgenre_code, c_litgenre_desc, c_litgenre_desc_chn, c_topic_code, " + _
        "c_topic_desc, c_topic_desc_chn, c_occasion_code, c_occasion_desc, c_occasion_desc_chn, " + _
        "c_inst_code, c_inst_name_hz, c_text_title, c_assoc_claimer_id, c_assoc_claimer_name, " + _
        "c_assoc_claimer_name_chn, c_distance, c_source, c_link_first_year, c_link_last_year, " + _
        "c_link_addr_id, c_link_addr_name, c_link_addr_chn ) "
tQuerySelectStr = "SELECT ASSOC_REL.c_personid, ASSOC_REL.c_person_name, ASSOC_REL.c_person_name_chn, " + _
"ASSOC_REL.c_index_year, ASSOC_REL.c_female, ASSOC_REL.c_node_id, ASSOC_REL.c_node_name, " + _
"ASSOC_REL.c_node_chn, ASSOC_REL.c_node_index_year, ASSOC_REL.c_node_female, 'N' AS c_link_type, " + _
"ASSOC_REL.c_link_code, ASSOC_REL.c_link_desc, ASSOC_REL.c_link_chn, ASSOC_REL.c_link_count, " + _
"ASSOC_REL.c_person_addr_id, ASSOC_REL.c_person_addr_name, ASSOC_REL.c_person_addr_chn, ASSOC_REL.c_perso
n_addr_type, " + _
"ASSOC_REL.c_person_addr_desc, ASSOC_REL.c_person_addr_desc_chn, ASSOC_REL.person_x_coord, ASSOC_REL.pers
on_y_coord, " + _
"ASSOC_REL.c_node_addr_id, ASSOC_REL.c_node_addr_name, ASSOC_REL.c_node_addr_chn, " + _
"ASSOC_REL.c_node_addr_type, ASSOC_REL.c_node_addr_desc, ASSOC_REL.c_node_addr_desc_chn, " + _
"ASSOC_REL.node_xcoord, ASSOC_REL.node_ycoord, ZZ_SCRATCH_PEOPLE.c_node_dist, 'N' AS type_id, ASSOC_REL.c
_kinrel, " + _
"ASSOC_REL.c_kinrel_chn, ASSOC_REL.c_kin_name, ASSOC_REL.c_kin_chn, ASSOC_REL.c_kin_id, " + _
"ASSOC_REL.c_kin_code, ASSOC_REL.c_assoc_kinrel, ASSOC_REL.c_assoc_kinrel_chn, " + _
"ASSOC_REL.c_assoc_kin_name, ASSOC_REL.c_assoc_kin_chn, ASSOC_REL.c_assoc_kin_id, " + _
"ASSOC_REL.c_assoc_kin_code, ASSOC_REL.c_litgenre_code, ASSOC_REL.c_litgenre_desc, " + _
"ASSOC_REL.c_litgenre_desc_chn, ASSOC_REL.c_topic_code, ASSOC_REL.c_topic_desc, " + _
"ASSOC_REL.c_topic_desc_chn, ASSOC_REL.c_occasion_code, ASSOC_REL.c_occasion_desc, " + _
"ASSOC_REL.c_occasion_desc_chn, ASSOC_REL.c_inst_code, ASSOC_REL.c_inst_name_hz, " + _
"ASSOC_REL.c_text_title, ASSOC_REL.c_assoc_claimer_id, " + _
"ASSOC_REL.c_assoc_claimer_name, ASSOC_REL.c_assoc_claimer_chn, ASSOC_REL.c_distance, ASSOC_REL.c_source
, " + _
"ASSOC_REL.c_assoc_first_year, ASSOC_REL.c_assoc_last_year, ASSOC_REL.c_assoc_addr_id, ASSOC_REL.c_assoc_
addr_name, ASSOC_REL.c_assoc_addr_chn "
tQueryFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE 1 INNER JOIN (ZZ_SCRATCH_PEOPLE INNER " + _
"JOIN ZZZ_NONKIN_BIOG_ADDR AS ASSOC_REL ON ZZ_SCRATCH_PEOPLE.c_person_id = " + _
"ASSOC_REL.c_personid) ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ASSOC_REL.c_node_id "

'If CmdClearList.Enabled Then
    '
    ' allow the people from the list to serve as nodes, then delete them
    '
    'cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr
    'cmdSQL.Execute tRecDeleted
    '
    ' now mark the nodes
    '
    'cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZ_SOCIAL_NETWORK " + _
    "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SOCIAL_NETWORK.c_node_id " + _
    "SET ZZ_SOCIAL_NETWORK.c_delete = 1"
    'cmdSQL.Execute tRecDeleted
    '
    ' and delete
    '
    'cmdSQL.CommandText = "Delete from ZZ_SOCIAL_NETWORK where c_delete = 1"
    'cmdSQL.Execute tRecCount
'Else
    'tQueryWhereStr = "WHERE (((ASSOC_REL.c_node_id)<> " + tID1Str + _
    " And (ASSOC_REL.c_node_id)<>" + tID2Str + "))"

    'cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr + tQueryWhereStr
    'cmdSQL.Execute tRecDeleted
'End If
cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr
cmdSQL.Execute tRecDeleted

```

```

If ChkKinship.Value Then

```

```

    ' get from the kinship table

```

```

tQueryIntoStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year, " + _
"c_female, c_node_id, c_node_name, c_node_chn, c_node_index_year, c_node_female, c_link_type, " + _
"c_link_code, c_link_desc, c_link_chn, c_link_count, c_addr_id, c_addr_name, c_addr_chn, " + _
"c_addr_type, c_addr_desc, c_addr_desc_chn, x_coord, y_coord, c_node_addr_id, c_node_addr_name, " + _
"c_node_addr_chn, c_node_addr_type, c_node_addr_desc, c_node_addr_desc_chn, node_xcoord, " + _
"node_ycoord, c_edge_dist, type_id, c_distance, c_source ) "
tQuerySelectStr = "SELECT ZZ_SCRATCH_PEOPLE.c_person_id, KIN_REL.c_person_name, " + _
"KIN_REL.c_person_name_chn, KIN_REL.c_index_year, KIN_REL.c_female, KIN_REL.c_node_id, " + _
"KIN_REL.c_node_name, KIN_REL.c_node_chn, KIN_REL.c_node_index_year, KIN_REL.c_node_female, " + _
"'K' AS c_link_type, KIN_REL.c_link_code, KIN_REL.c_link_desc, KIN_REL.c_link_chn, " + _

```

```

"1 AS c_link_count, KIN_REL.c_person_addr_id, KIN_REL.c_person_addr_name, KIN_REL.c_person_addr_chn,
" + _
"KIN_REL.c_person_addr_type, KIN_REL.c_person_addr_desc, KIN_REL.c_person_addr_desc_chn, KIN_REL.pers
on_x_coord, " + _
"KIN_REL.person_y_coord, KIN_REL.c_node_addr_id, KIN_REL.c_node_addr_name, KIN_REL.c_node_addr_chn, "
+ _
"KIN_REL.c_node_addr_type, KIN_REL.c_node_addr_desc, KIN_REL.c_node_addr_desc_chn, " + _
"KIN_REL.node_xcoord, KIN_REL.node_ycoord, ZZ_SCRATCH_PEOPLE.c_node_dist, 'K' AS type_id, " + _
"KIN_REL.c_distance, KIN_REL.c_source "
tQueryFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN (ZZ_SCRATCH_PEOPLE " + _
"INNER JOIN ZZZ_KIN_BIOG_ADDR AS KIN_REL ON ZZ_SCRATCH_PEOPLE.c_person_id = " + _
"KIN_REL.c_personid) ON ZZ_SCRATCH_PEOPLE_1.c_person_id = KIN_REL.c_node_id "

'If CmdClearList.Enabled Then
'
'    allow the people from the list to serve as nodes, then delete them
'
'cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr
'cmdSQL.Execute tRecDeleted
'
'    now mark the nodes
'
'cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZ_SOCIAL_NETWORK " + _
"ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SOCIAL_NETWORK.c_node_id " + _
"SET ZZ_SOCIAL_NETWORK.c_delete = 1"
'cmdSQL.Execute tRecDeleted
'
'    and delete
'
'cmdSQL.CommandText = "Delete from ZZ_SOCIAL_NETWORK where c_delete = 1"
'cmdSQL.Execute tRecCount
'Else
'tQueryWhereStr = "WHERE (((KIN_REL.c_node_id)<>" + tID1Str + _
" And (KIN_REL.c_node_id)<>" + tID2Str + "))"

'cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr + tQueryWhereStr
'cmdSQL.Execute tRecDeleted
'End If
cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr
cmdSQL.Execute tRecDeleted
End If
'
'    now mark all the duplicates that are inverses of other records for deletion
'
tQueryStr = "UPDATE ASSOC_CODES INNER JOIN (ZZ_SOCIAL_NETWORK AS ZZ_SOCIAL_NETWORK_1 INNER JOIN " + _
"ZZ_SOCIAL_NETWORK ON (ZZ_SOCIAL_NETWORK_1.c_node_id = ZZ_SOCIAL_NETWORK.c_person_id) AND " + _
"(ZZ_SOCIAL_NETWORK_1.c_person_id = ZZ_SOCIAL_NETWORK.c_node_id)) ON (ASSOC_CODES.c_assoc_pair = " + _
"ZZ_SOCIAL_NETWORK_1.c_link_code) AND (ASSOC_CODES.c_assoc_code = ZZ_SOCIAL_NETWORK.c_link_code) " + _
"SET ZZ_SOCIAL_NETWORK.c_delete = 1 " + _
"WHERE (((ZZ_SOCIAL_NETWORK.c_link_type)='N') AND " + _
"((ZZ_SOCIAL_NETWORK.c_edge_dist)>[ZZ_SOCIAL_NETWORK_1].[c_edge_dist])) OR " + _
"(((ZZ_SOCIAL_NETWORK.c_link_type)='N') AND " + _
"((ZZ_SOCIAL_NETWORK.c_edge_dist)=[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) AND " + _
"((ZZ_SOCIAL_NETWORK.c_person_id)>[ZZ_SOCIAL_NETWORK_1].[c_person_id]))"

'MsgBox "About to mark inverses..."
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
If Me.ChkKinship.Value Then
tQueryStr = "UPDATE KINSHIP_CODES INNER JOIN (ZZ_SOCIAL_NETWORK AS ZZ_SOCIAL_NETWORK_1 INNER JOIN " + _
"ZZ_SOCIAL_NETWORK ON (ZZ_SOCIAL_NETWORK_1.c_node_id = ZZ_SOCIAL_NETWORK.c_person_id) AND " + _
"(ZZ_SOCIAL_NETWORK_1.c_person_id = ZZ_SOCIAL_NETWORK.c_node_id)) ON KINSHIP_CODES.c_kincode = " + _
"ZZ_SOCIAL_NETWORK.c_link_code SET ZZ_SOCIAL_NETWORK.c_delete = 1 " + _
"WHERE (((ZZ_SOCIAL_NETWORK.c_link_type)='K') AND " + _
"((ZZ_SOCIAL_NETWORK.c_edge_dist)>[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) AND " + _
"((ZZ_SOCIAL_NETWORK_1.c_link_code)=[KINSHIP_CODES].[c_kin_pair1])) OR (((ZZ_SOCIAL_NETWORK.c_link_ty
pe)='K') " + _
"AND ((ZZ_SOCIAL_NETWORK.c_edge_dist)=[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) AND " + _
"((ZZ_SOCIAL_NETWORK.c_person_id)>[ZZ_SOCIAL_NETWORK_1].[c_person_id]) AND " + _
"((ZZ_SOCIAL_NETWORK_1.c_link_code)=[KINSHIP_CODES].[c_kin_pair1])) OR (((ZZ_SOCIAL_NETWORK.c_link_ty
pe)='K') " + _
"AND ((ZZ_SOCIAL_NETWORK.c_edge_dist)>[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) " + _
"AND ((ZZ_SOCIAL_NETWORK_1.c_link_code)=[KINSHIP_CODES].[c_kin_pair2])) OR (((ZZ_SOCIAL_NETWORK.c_lin
k_type)='K') " + _
"AND ((ZZ_SOCIAL_NETWORK.c_edge_dist)=[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) AND " + _
"((ZZ_SOCIAL_NETWORK.c_person_id)>[ZZ_SOCIAL_NETWORK_1].[c_person_id]) AND " + _
"((ZZ_SOCIAL_NETWORK_1.c_link_code)=[KINSHIP_CODES].[c_kin_pair2]))"

cmdSQL.CommandText = tQueryStr

```

```

        cmdSQL.Execute tRecCount
    End If
    '
    ' now delete
    '
    'MsgBox "About to delete inverses..."
    cmdSQL.CommandText = "Delete from ZZ_SOCIAL_NETWORK where c_delete = 1"
    cmdSQL.Execute tRecCount
    '
    ' this approach still produces situations where people are associated with themselves. Those records must be removed
    '
    tQueryStr = "UPDATE ZZ_SOCIAL_NETWORK SET ZZ_SOCIAL_NETWORK.c_delete = 1 " + _
        "WHERE (((ZZ_SOCIAL_NETWORK.c_person_id)=[ZZ_SOCIAL_NETWORK].[c_node_id]))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = "Delete from ZZ_SOCIAL_NETWORK where c_delete = 1"
    cmdSQL.Execute tRecCount
    '
    ' now that everything else is settled, I add the dynasty and new index year descriptive information
    '
    cmdSQL.CommandText = "UPDATE ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 INNER JOIN (ZZZ_BIOG_MAIN INNER JOIN ZZ_SOCIAL_NETWORK " + _
        "ON ZZZ_BIOG_MAIN.c_personid = ZZ_SOCIAL_NETWORK.c_person_id) ON ZZZ_BIOG_MAIN_1.c_personid = ZZ_SOCIAL_NETWORK.c_node_id " + _
        "SET ZZ_SOCIAL_NETWORK.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
        "ZZ_SOCIAL_NETWORK.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
        "ZZ_SOCIAL_NETWORK.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " + _
        "ZZ_SOCIAL_NETWORK.c_node_index_year_type_code = [ZZZ_BIOG_MAIN_1].[c_index_year_type_code], " + _
        "ZZ_SOCIAL_NETWORK.c_node_index_year_type_desc = [ZZZ_BIOG_MAIN_1].[c_index_year_type_desc], " + _
        "ZZ_SOCIAL_NETWORK.c_node_index_year_type_hz = [ZZZ_BIOG_MAIN_1].[c_index_year_type_hz], " + _
        "ZZ_SOCIAL_NETWORK.c_dy = [ZZZ_BIOG_MAIN].[c_dy], " + _
        "ZZ_SOCIAL_NETWORK.c_dynasty = [ZZZ_BIOG_MAIN].[c_dynasty], " + _
        "ZZ_SOCIAL_NETWORK.c_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn], " + _
        "ZZ_SOCIAL_NETWORK.c_node_dy = [ZZZ_BIOG_MAIN_1].[c_dy], " + _
        "ZZ_SOCIAL_NETWORK.c_node_dynasty = [ZZZ_BIOG_MAIN_1].[c_dynasty], " + _
        "ZZ_SOCIAL_NETWORK.c_node_dynasty_chn = [ZZZ_BIOG_MAIN_1].[c_dynasty_chn]"
    cmdSQL.Execute tRecDeleted
    '
    ' finally, add the source text information to ZZ_SOCIAL_NETWORK
    '
    cmdSQL.CommandText = "UPDATE ZZ_SOCIAL_NETWORK INNER JOIN TEXT_CODES ON ZZ_SOCIAL_NETWORK.c_source = TEXT_CODES.c_textid " + _
        "SET ZZ_SOCIAL_NETWORK.c_source_text = [TEXT_CODES].[c_title], " + _
        "ZZ_SOCIAL_NETWORK.c_source_txt_chn = [TEXT_CODES].[c_title_chn]"
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = "UPDATE ZZZ_BIOG_MAIN INNER JOIN ZZ_SCRATCH_PEOPLE ON ZZZ_BIOG_MAIN.c_personid = ZZ_SCRATCH_PEOPLE.c_person_id " + _
        "SET ZZ_SCRATCH_PEOPLE.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
        "ZZ_SCRATCH_PEOPLE.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
        "ZZ_SCRATCH_PEOPLE.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " + _
        "ZZ_SCRATCH_PEOPLE.c_dy = [ZZZ_BIOG_MAIN].[c_dy], " + _
        "ZZ_SCRATCH_PEOPLE.c_dynasty = [ZZZ_BIOG_MAIN].[c_dynasty], " + _
        "ZZ_SCRATCH_PEOPLE.c_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn]"
    cmdSQL.Execute tRecDeleted
    '
    '
    ' the final step is to calculate the xy_count
    '
    ' get the XY count
    '
    'MsgBox "About to count XYs..."
    cmdSQL.CommandText = "Delete * from tmpXY"
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
        "SELECT ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord, Count(ZZ_SCRATCH_PEOPLE.x_coord) " + _
        "AS CountOfx_coord, Count(ZZ_SCRATCH_PEOPLE.y_coord) AS CountOfy_coord " + _
        "FROM ZZ_SCRATCH_PEOPLE " + _
        "GROUP BY ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord;"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_PEOPLE ON (tmpXY.y_coord = " + _
        "ZZ_SCRATCH_PEOPLE.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_PEOPLE.x_coord) SET " + _

```

```
"ZZ_SCRATCH_PEOPLE.xy_count = [tmpXY].[CountOfx_coord];"
```

```
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
```

```
Set gRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
gRstPeople.MoveLast
```

```
If gRstPeople.RecordCount > 0 Then
```

```
    CmdGIS.Enabled = True
    CmdPajek.Enabled = True
    CmdGephi.Enabled = True
    CmdUCINet.Enabled = True
    CmdNeo4j.Enabled = True
    ChkIncludeID.Enabled = True
    CmdStoreID.Enabled = True
```

```
Else
```

```
    CmdGIS.Enabled = False
    CmdPajek.Enabled = False
    CmdGephi.Enabled = False
    CmdUCINet.Enabled = False
    CmdNeo4j.Enabled = False
    ChkIncludeID.Enabled = False
    CmdStoreID.Enabled = False
```

```
End If
```

```
Exit_CmdQuery_Click:
```

```
    ' restore the form tables
```

```
Set ZZ_SOCIAL_NETWORK.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
```

```
Set ZZ_SCRATCH_PEOPLE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
```

```
    '
```

```
Set cmdSQL = Nothing
```

```
Exit Sub
```

```
Err_CmdQuery_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdQuery_Click
```

```
End Sub
```

```
Private Sub calculate_xy_count()
```

```
Dim tX As Double, tY As Double, tXY As Integer
```

```
Dim tBM As Variant, tWrite As Integer
```

```
    '
```

```
    ' the strategy is to first throw a bookmark at the first new value
```

```
    ' then count the number, then go back to the bookmark and update each record
```

```
    '
```

```
With gRstPeople
```

```
    .Index = "xy"
```

```
    .MoveFirst
```

```
tX = -1#
```

```
tY = -1#
```

```
tXY = 0
```

```
tWrite = 0
```

```
tBM = .Bookmark
```

```
Do While Not .EOF
```

```
    If tX <> !x_coord Or tY <> !y_coord Then
```

```
        If tWrite = 1 Then
```

```
            ' go back to the first record with the value
```

```
            .Bookmark = tBM
```

```
            Do While tX = !x_coord And tY = !y_coord
```

```
                .Edit
```

```
                !xy_count = tXY
```

```
                .Update
```

```
                .MoveNext
```

```
            Loop
```

```
        Else
```

```
            tWrite = 1
```

```
        End If
```

```
        ' reset
```

```
tXY = 0
```

```
tBM = .Bookmark
```

```
tX = !x_coord
```

```
tY = !y_coord
```

```
End If
```

```
    ' increment the count and move to the next
```

```
tXY = tXY + 1
```

```
.MoveNext
```

```

    Loop
    '
    ' the last xy value still needs to be written
    '
    .Bookmark = tBM
    Do While Not .EOF
        .Edit
        !xy_count = tXY
        .Update
        .MoveNext
    Loop
    .Index = "index_year"
End With
End Sub
Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call writeKML
    Exit Sub
End If
'
' This program will dump the results to a .gis file
'
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGIS_Click
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".txt"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdGIS_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".txt"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".txt") Then
            tFileName = tFileName + ".txt"
        End If
    End If
    '
    ' write the file
    '

```

```

'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
'
' process the table
'
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
tC = Chr(44) ' the comma
'
With tRstNode
    ' write the header
    '
    tStr = "Name" + tC + "NameChn" + tC + "Female" + tC + "IndexYear" + tC
    tStr = tStr + "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC
    tStr = tStr + "xy_count"
    tStream.WriteText tStr, adWriteLine
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs
        '
        If IsNull(!c_name) Then
            tStr = "[?]" + tC
        Else
            If Trim(!c_name) = "" Then
                tStr = "[?]" + tC
            Else
                tStr = !c_name + tC
            End If
        End If

        If IsNull(!c_name_chn) Then
            tStr = tStr + "[?]" + tC
        Else
            If Trim(!c_name_chn) = "" Then
                tStr = tStr + "[?]" + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If
        End If

        If !c_female Then
            tStr = tStr + "F" + tC
        Else
            tStr = tStr + "M" + tC
        End If

        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Str(!c_index_year) + tC
        End If

        ' here guard against blanks as well

        If IsNull(!c_addr_name) Then
            tStr = tStr + "[?]" + tC
        ElseIf Trim(!c_addr_name) = "" Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_addr_name + tC
        End If

        If IsNull(!c_addr_chn) Then
            tStr = tStr + "[?]" + tC
        ElseIf Trim(!c_addr_chn) = "" Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_addr_chn + tC
        End If

        If IsNull(!x_coord) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Str(!x_coord) + tC
        End If

        If IsNull(!y_coord) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Str(!y_coord) + tC
        End If
    End If
End With

```



```

    If IsNull(!xy_count) Then
        tStr = tStr + "0"
    Else
        tStr = tStr + Str(!xy_count)
    End If

```

```

    tStream.WriteText tStr, adWriteLine
    .MoveNext

```

```

    Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite

```

```

Else
    'The user pressed Cancel.
End If

```

```

Set tRstNode = Nothing

```

```

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

```

```

Exit_CmdGIS_Click:
Exit Sub

```

```

Err_CmdGIS_Click:
MsgBox Err.Description
Resume Exit_CmdGIS_Click

```

```

End Sub

```

```

Private Sub CmdToDynasty_Click()
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strToDynasty As String

```

```

    If gToDynasty = -1 Then
        strToDynasty = ""
    Else
        strToDynasty = Str(gToDynasty)
    End If

```

```

    stDocName = "frmPickDynasty"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

```

```

    If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

```

```

        Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
        gToDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

```

```

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
        gToDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

```

```

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
        gToDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value

```

```

        ' check to see if we have a problem and reject selection if needed

```

```

        If gFromDynasty > -1 Then
            If gFromDynastyBegin > gToDynastyEnd Then
                MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
                gToDynasty = -1
                TxtToDynasty.Value = ""
                TxtToDynastyPY.Value = ""
            End If
        End If

```

```

        ' value is OK

```

```

        If gToDynasty > -1 Then
            Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
            TxtToDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

```

```

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
        TxtToDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
    End If

    DoCmd.Close acForm, stDocName
    '
    ' reset FromDynasty if necessary (-2 = all dynasties)
    '
    If gFromDynasty = -2 Then
        gFromDynasty = -1
        TxtFromDynasty.Value = ""
        TxtFromDynastyPY.Value = ""
    End If
    '
End If

End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim cmdSQL As ADODB.Command, tRecDeleted As Variant
    Dim tRstAssocCode As DAO.Recordset, tRstDummy As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    '
    ' to clear the tables, briefly close and then delete records
    '
    '
    ' Clear the Edge output table
    '
    Set gRstEdge = ZZ_SOCIAL_NETWORK.Form.Recordset
    '
    If gRstEdge.RecordCount > 0 Then
        Set ZZ_SOCIAL_NETWORK.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SN", dbOpenDynaset)
        gRstEdge.Close

        Set cmdDel = New ADODB.Command
        cmdDel.ActiveConnection = CurrentProject.Connection
        cmdDel.CommandType = adCmdText
        '
        cmdDel.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
        cmdDel.Execute tRecDeleted
        Set cmdDel = Nothing
        '
        Set gRstEdge = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
        Set ZZ_SOCIAL_NETWORK.Form.Recordset = gRstEdge
    End If
    '
    ' Clear the Node output table
    '
    Set tRstDummy = ZZ_SCRATCH_PEOPLE.Form.Recordset
    '
    If tRstDummy.RecordCount > 0 Then
        Set ZZ_SCRATCH_PEOPLE.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SP", dbOpenDynaset)
        tRstDummy.Close

        Set cmdDel = New ADODB.Command
        cmdDel.ActiveConnection = CurrentProject.Connection
        cmdDel.CommandType = adCmdText
        '
        cmdDel.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
        cmdDel.Execute tRecDeleted
        Set cmdDel = Nothing
        '
        Set ZZ_SCRATCH_PEOPLE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
    End If
    Set tRstDummy = Nothing

    ' first determine the language
    gLCID = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    If gLCID = 2052 Or gLCID = 3076 Then ' 2052 = PRC, 3076 = Hong Kong
        gDisplayLanguage = "S"
    ElseIf gLCID = 4100 Or gLCID = 1028 Then ' 4100 = Singapore, 1028 = Taiwan
        gDisplayLanguage = "T"
        Call changeDisplayLanguage
    Else
        gDisplayLanguage = "E"
        Call changeDisplayLanguage
    End If

```

```
gFromDynasty = -1
gToDynasty = -1
```

```
Me.TxtFromYear.Enabled = False
Me.TxtToYear.Enabled = False
'Me.ChkIndexYears.Value = False
Me.ChkIncludeID.Value = False
```

```
If DCount("?", "ZZ_STORE_PERSON_ID") > 0 Then
    Me.CmdRecallID.Enabled = True
Else
    Me.CmdRecallID.Enabled = False
End If
```

```
End Sub
```

```
Private Sub CmdPajek_Click()
On Error GoTo Err_CmdPajek_Click
```

```
' This program will dump the results of the search to a .net file
' for the moment I'll just describe the format of the .gdf file
'
' *Vertices NUM
' ID label "box" ic [color] bc [color]
'   ID = str(c_person_id)
'   label = c_name_chn
'   color = red (1), orange (2), yellow (3), green (4), blue (5)
'
' *Edges
' node1 node2 1 1 "label"
'   node1 = str(c_person_id) for node1
'   node2 = str(c_node_id) for node2
'   color = red (1), orange (2), yellow (3), green (4), blue (5)
'   label = c_link_desc
```

```
' first see if there are any records to process
'
If ZZ_SOCIAL_NETWORK.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
```

```
'
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
```

```
' next get a file
```

```
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstNodeList As DAO.Recordset
Dim tRstEdge As DAO.Recordset, tRstAssocType As DAO.Recordset
Dim tRstAssocCodeType As DAO.Recordset, tRstEdgeList As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tQuote As String, tFindStr As String
Dim tColor(20) As String, tStrNode1 As String, tStrNode2 As String, tCodeStr As String
```

```
' to write to a UTF-8 file, use the ADO stream object
```

```
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8.net"
ElseIf CodeFrame.Value = 2 Then
    tStream.Charset = "big5"
    tCodeStr = "BIG5.net"
ElseIf CodeFrame.Value = 3 Then
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030.net"
Else
    tStream.Charset = "ascii"
    tCodeStr = "ASCII.net"
```

```
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
```

```
Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)
```

```
'Use a With...End With block to reference the FileDialog object.
```

```
With dlgSaveAs
```

```
    .InitialFileName = "network_" + tCodeStr
```

```
    If .Show = -1 Then
```

```
    '
```

```
        tFileName = ""
```

```
        For Each tFN In .SelectedItems
```

```
            tFileName = tFN
```

```
            If Not tFileName = "" Then
```

```
                Exit For
```

```
            End If
```

```
        Next
```

```
        If tFileName = "" Then
```

```
            MsgBox "Bad file Name."
```

```
            GoTo Exit_CmdPajek_Click
```

```
        Else
```

```
            ' make sure the file name has a net extension
```

```
            If Len(tFileName) < 5 Then
```

```
                tFileName = tFileName + ".net"
```

```
            ElseIf Not (LCase(Right(tFileName, 4)) = ".net") Then
```

```
                tFileName = tFileName + ".net"
```

```
            End If
```

```
        End If
```

```
    '
```

```
    ' zap and open the scratch file
```

```
    '
```

```
    Dim cmdSQL As ADODB.Command
```

```
    Set cmdSQL = New ADODB.Command
```

```
    cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
    cmdSQL.CommandType = adCmdText
```

```
    '
```

```
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
```

```
    cmdSQL.Execute tRecDeleted
```

```
    '
```

```
    ' fill the node list
```

```
    If CodeFrame.Value = 4 Then
```

```
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
```

```
        "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id, ZZ_SCRATCH_PEOPLE.c_name, " + _
```

```
        "ZZ_SCRATCH_PEOPLE.c_node_dist, val(c_person_id) AS c_v_num, TRUE as c_delete FROM ZZ_SCRATCH
```

```
_PEOPLE"
```

```
    Else
```

```
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
```

```
        "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id, ZZ_SCRATCH_PEOPLE.c_name_chn, " + _
```

```
        "ZZ_SCRATCH_PEOPLE.c_node_dist, val(c_person_id) AS c_v_num, TRUE as c_delete FROM ZZ_SCRATCH
```

```
_PEOPLE"
```

```
    End If
```

```
    cmdSQL.CommandText = tQueryStr
```

```
    cmdSQL.Execute tRecDeleted
```

```
    '
```

```
    ' fill in any missing names
```

```
    '
```

```
    tQueryStr = "UPDATE ZZ_SCRATCH_PEOPLE INNER JOIN ZZ_SCRATCH_PAJEK ON " + _
```

```
    "ZZ_SCRATCH_PEOPLE.c_person_id = ZZ_SCRATCH_PAJEK.c_ID SET ZZ_SCRATCH_PAJEK.c_lbl = " + _
```

```
    "[ZZ_SCRATCH_PEOPLE].[c_name] WHERE (((ZZ_SCRATCH_PAJEK.c_lbl) Is Null))"
```

```
    '
```

```
    ' if needed, find the 0-degree nodes, using the edge list to mark the node list
```

```
    '
```

```
    If ChkDegree.Value Then
```

```
    '
```

```
        cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PAJEK INNER JOIN ZZ_SOCIAL_NETWORK " + _
```

```
        "ON ZZ_SCRATCH_PAJEK.c_id = ZZ_SOCIAL_NETWORK.c_person_id " + _
```

```
        "SET ZZ_SCRATCH_PAJEK.c_delete = False"
```

```
        cmdSQL.Execute tRecDeleted
```

```
    '
```

```
        cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PAJEK INNER JOIN ZZ_SOCIAL_NETWORK " + _
```

```
        "ON ZZ_SCRATCH_PAJEK.c_id = ZZ_SOCIAL_NETWORK.c_node_id " + _
```

```
        "SET ZZ_SCRATCH_PAJEK.c_delete = False"
```

```
        cmdSQL.Execute tRecDeleted
```

```
    '
```

```
        ' remove records where c_delete = TRUE
```

```
    '
```

```
        'MsgBox "Got through update"
```

```
    '
```

```
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK WHERE ((ZZ_SCRATCH_PAJEK.c_delete) = TRUE )"
```

```
        cmdSQL.Execute tRecDeleted
```

```

End If

Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
tRstNodeList.Index = "c_ID"
'
'   there probably is an SQL way to do this, but...
'
ti = 1
With tRstNodeList
    .MoveFirst
    Do While Not .EOF
        .Edit
        !c_v_num = Trim(Str(ti))
        .Update
        ti = ti + 1
        .MoveNext
    Loop
End With
tRstNodeList.Close
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK_EDGE"
cmdSQL.Execute tRecDeleted
'
'   fill the edge list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK_EDGE ( c_node_1, c_node_2, c_edge_count, c_edge_dist ) " + _
    "SELECT Val([ZZ_SCRATCH_PAJEK].[c_v_num]) AS c_node_1, " + _
    "Val([ZZ_SCRATCH_PAJEK_1].[c_v_num]) AS c_node_2, " + _
    "Sum(ZZ_SOCIAL_NETWORK.c_link_count) AS SumOfc_link_count, " + _
    "Min(ZZ_SOCIAL_NETWORK.c_edge_dist) AS MinOfc_edge_dist " + _
    "FROM ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 INNER JOIN " + _
    "(ZZ_SOCIAL_NETWORK INNER JOIN ZZ_SCRATCH_PAJEK ON ZZ_SOCIAL_NETWORK.c_person_id = " + _
    "ZZ_SCRATCH_PAJEK.c_ID) ON ZZ_SCRATCH_PAJEK_1.c_ID = ZZ_SOCIAL_NETWORK.c_node_id " + _
    "GROUP BY Val([ZZ_SCRATCH_PAJEK].[c_v_num]), Val([ZZ_SCRATCH_PAJEK_1].[c_v_num])"

'   NOTE:  the commented-out code is for when we do not allow parallel edges and have to aggregate the
results
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK_EDGE ( c_node_1, c_node_2, c_edge_desc, c_edge_count, c_ed
ge_dist ) " + _
    "SELECT Val([ZZ_SCRATCH_PAJEK].[c_v_num]) AS c_node_1, " + _
    "Val([ZZ_SCRATCH_PAJEK_1].[c_v_num]) AS c_node_2, " + _
    "ZZ_SOCIAL_NETWORK.c_link_desc, " + _
    "ZZ_SOCIAL_NETWORK.c_link_count, " + _
    "ZZ_SOCIAL_NETWORK.c_edge_dist " + _
    "FROM ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 INNER JOIN " + _
    "(ZZ_SOCIAL_NETWORK INNER JOIN ZZ_SCRATCH_PAJEK ON ZZ_SOCIAL_NETWORK.c_person_id = " + _
    "ZZ_SCRATCH_PAJEK.c_ID) ON ZZ_SCRATCH_PAJEK_1.c_ID = ZZ_SOCIAL_NETWORK.c_node_id "

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'   *****If we are allowing parallel edges, this section is no longer releva
nt
'
'   now fill in the edge description.  This requires three steps
'
cmdSQL.CommandText = "DROP TABLE tmp_scratch_pajek"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "SELECT ZZ_SCRATCH_PAJEK.c_ID, Val(ZZ_SCRATCH_PAJEK.c_v_num) AS c_v_num INTO " + _
    "TMP_SCRATCH_PAJEK FROM ZZ_SCRATCH_PAJEK"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "UPDATE ((ZZ_SOCIAL_NETWORK INNER JOIN TMP_SCRATCH_PAJEK ON " + _
    "ZZ_SOCIAL_NETWORK.c_person_id = TMP_SCRATCH_PAJEK.c_ID) " + _
    "INNER JOIN ZZ_SCRATCH_PAJEK_EDGE ON " + _
    "TMP_SCRATCH_PAJEK.c_v_num = ZZ_SCRATCH_PAJEK_EDGE.c_node_1) " + _
    "INNER JOIN TMP_SCRATCH_PAJEK AS TMP_SCRATCH_PAJEK_1 ON (TMP_SCRATCH_PAJEK_1.c_v_num = " + _
    "ZZ_SCRATCH_PAJEK_EDGE.c_node_2) AND (ZZ_SOCIAL_NETWORK.c_node_id = TMP_SCRATCH_PAJEK_1.c_ID)
" + _
    "SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = [ZZ_SOCIAL_NETWORK].[type_id]+'&'+[ZZ_SOCIAL_NETWORK
].[c_link_desc] " + _
    "WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_count)=1))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

```

```

cmdSQL.CommandText = "DROP TABLE tmp_scratch_pajek"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "UPDATE ZZ_SCRATCH_PAJEK_EDGE SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = " + _
    "'Parallel Edges merged' WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_count)>1))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenDynaset)
Set tRstEdgeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK_EDGE", dbOpenDynaset)
'
' set the Quote delimiter
'
tQuote = Chr(34)
'
' define the colors for the nodes
'
tColor(1) = "Black"
tColor(2) = "Blue"
tColor(3) = "Green"
tColor(4) = "Yellow"
tColor(5) = "Orange"
For ti = 6 To 20
    tColor(ti) = "Red"
Next
'
tC = Chr(44) ' the comma
'
' first the nodes: define the record structure
'
tRstNodeList.MoveLast
tStr = "*Vertices " + Trim(Str(tRstNodeList.RecordCount))
tStream.WriteText tStr, adWriteLine
'
ti = 1

With tRstNodeList
    .MoveFirst
    Do While Not .EOF
        tStream.WriteText !c_v_num + " "
        '
        If IsNull(!c_lbl) Then
            tStream.WriteText Chr(34)
            tStream.WriteText "Error-" + Trim(Str(!c_ID))
            tStream.WriteText Chr(34)
            tStream.WriteText " box "
        Else
            If !c_lbl = "" Then
                tStream.WriteText Chr(34)
                tStream.WriteText "Error-" + Trim(Str(!c_ID))
                tStream.WriteText Chr(34)
                tStream.WriteText " box "
            Else
                tStream.WriteText Chr(34)
                tStream.WriteText !c_lbl
                If ChkIncludeID.Value Then
                    tStream.WriteText ":" + Trim(Str(!c_ID))
                End If
                tStream.WriteText Chr(34)
                tStream.WriteText " box "
            End If
        End If
        ' label
        tStr = " ic " + tColor(!c_distance + 1)
        tStr = tStr + " bc " + tColor(!c_distance + 1)
        ' color = white (1), blue (2), green (3), yellow (4), orange (5)
        tStream.WriteText tStr, adWriteLine
        '
    .MoveNext
Loop
End With
'
' now the edges: define the record structure
'
tStream.WriteText "*Edges", adWriteLine

If tRstEdgeList.RecordCount > 0 Then

```

```

        With tRstEdgeList
            .MoveFirst
            Do While Not .EOF
                tStr = Trim(Str(!c_node_1)) + " " + Trim(Str(!c_node_2))
                '
                ' now get the weight
                '
                If !c_edge_count < 6 Then
                    tStr = tStr + " " + Trim(Str(!c_edge_count)) + " "
                Else
                    tStr = tStr + " 5 "
                End If
                '
                ' now get the label
                '
                tStr = tStr + "l " + tQuote
                If !c_edge_count = 1 Then
                    tStr = tStr + !c_edge_desc + tQuote + " "
                Else
                    tStr = tStr + Trim(Str(!c_edge_count)) + " links" + tQuote + " "
                End If

                tStr = tStr + "c " + tColor(!c_edge_dist + 1)
                '
                ' color = white (1), blue (2), green (3), yellow (4), orange (5)
                '
                tStream.WriteText tStr, adWriteLine
                '
                .MoveNext
            Loop
        End With
    End If
    '
    ' now make sure all the data is copied to tStream
    tStream.Flush
    ' and write the stream to the file
    tStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    tRstNodeList.Close

    tStream.Close
    Set tStream = Nothing
    '
    'Set tGDF = Nothing
    'Set tFileSystem = Nothing
    Set tRstNodeList = Nothing
    Set tRstEdgeList = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdPajek_Click:
Exit Sub

Err_CmdPajek_Click:
MsgBox Err.Description
Resume Exit_CmdPajek_Click

End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

If gDisplayLanguage = "T" Then
    gDisplayLanguage = "E"
Else
    gDisplayLanguage = "T"
End If

Call changeDisplayLanguage

Exit_CmdFanti_Click:
Exit Sub

Err_CmdFanti_Click:
MsgBox Err.Description

```

```

Resume Exit_CmdFanti_Click

End Sub

Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

If gDisplayLanguage = "S" Then
    gDisplayLanguage = "E"
Else
    gDisplayLanguage = "S"
End If

Call changeDisplayLanguage

Exit_CmdJianti_Click:
Exit Sub

Err_CmdJianti_Click:
MsgBox Err.Description
Resume Exit_CmdJianti_Click

End Sub

Public Sub changeDisplayLanguage()
Dim tLabelLanguage(3, 36) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False
With tRstLabelList
.MoveFirst
ti = 1

Do While ti < 36 And Not .EOF
    If !c_form = "LAAP" Then
        gLabelsOK = True
        If ti <> !c_label_id Then
            MsgBox "Uh oh: mismatched label table"
            gLabelsOK = False
            Exit Do
        End If
        tLabelLanguage(1, ti) = !c_english
        tLabelLanguage(2, ti) = !c_fanti
        tLabelLanguage(3, ti) = !c_jianti
        ti = ti + 1
    End If
.MoveNext
Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If gLabelsOK Then
    If gDisplayLanguage = "E" Then
        tLang = 1
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '
    Me.LblFrom.Caption = tLabelLanguage(tLang, 1)
    Me.LblTo.Caption = tLabelLanguage(tLang, 2)
    ' Me.Lbl1Node.Caption = tLabelLanguage(tLang, 3)
    Me.Lbl2Node.Caption = tLabelLanguage(tLang, 4)
    ' Me.Lbl1NodeNode.Caption = tLabelLanguage(tLang, 5)
    Me.LblKin.Caption = tLabelLanguage(tLang, 6)
    Me.CmdPickPerson1.Caption = tLabelLanguage(tLang, 7)
    Me.CmdPickPerson2.Caption = tLabelLanguage(tLang, 8)
    Me.CmdQuery.Caption = tLabelLanguage(tLang, 9)
    Me.CmdGIS.Caption = tLabelLanguage(tLang, 10)
    Me.CmdPajek.Caption = tLabelLanguage(tLang, 11)
    Me.CmdFanti.Caption = tLabelLanguage(tLang, 12)
    Me.CmdJianti.Caption = tLabelLanguage(tLang, 13)

```



```

Me.PageAssoc.Caption = tLabelLanguage(tLang, 14)
Me.PagePeople.Caption = tLabelLanguage(tLang, 15)
Me.LblIncludeID.Caption = tLabelLanguage(tLang, 16)
'Me.LblIndexYears.Caption = tLabelLanguage(tLang, 17)
Me.LblDisplayLanguage.Caption = tLabelLanguage(tLang, 18)
Me.CmdImportList.Caption = tLabelLanguage(tLang, 19)
Me.CmdClearList.Caption = tLabelLanguage(tLang, 20)
Me.CmdStoreID.Caption = tLabelLanguage(tLang, 21)
Me.CmdRecallID.Caption = tLabelLanguage(tLang, 22)
Me.LblChkDegree.Caption = tLabelLanguage(tLang, 23)
Me.CmdUCINet.Caption = tLabelLanguage(tLang, 24)
Me.CmdHelp.Caption = tLabelLanguage(tLang, 25)
Me.CmdGephi.Caption = tLabelLanguage(tLang, 26)
Me.LblDynasties.Caption = tLabelLanguage(tLang, 27)
Me.CmdFromDynasty.Caption = tLabelLanguage(tLang, 28)
Me.CmdToDynasty.Caption = tLabelLanguage(tLang, 29)
Me.CmdAllDynasties.Caption = tLabelLanguage(tLang, 30)
Me.LblIndexYears.Caption = tLabelLanguage(tLang, 31)
Me.LblOptNoDates.Caption = tLabelLanguage(tLang, 32)
Me.LblOptIndexYears.Caption = tLabelLanguage(tLang, 33)
Me.LblOptDynasties.Caption = tLabelLanguage(tLang, 34)
Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 35)
End If

```

End Sub

Private Sub process_records(tType As String)

```

Dim tTestStr As String, tContinue As Integer

```

```

With gRst

```

```

    'MsgBox "processing edges"

```

```

    .MoveFirst

```

```

    Do While Not .EOF

```

```

        ' see if we have the record

```

```

        tContinue = 1

```

```

        If gRstEdge.RecordCount > 0 Then

```

```

            tTestStr = "c_person_id = " + Trim(Str(!c_personid))

```

```

            tTestStr = tTestStr + " AND c_node_id = " + Trim(Str(!c_node_id))

```

```

            tTestStr = tTestStr + " AND c_link_code = " + Trim(Str(!c_link_code))

```

```

            If tType = "N" Then

```

```

                If Not IsNull(!c_kin_code) Then

```

```

                    If !c_kin_code > 0 Then

```

```

                        tTestStr = tTestStr + " AND c_kin_code = " + Trim(Str(!c_kin_code))

```

```

                        tTestStr = tTestStr + " AND c_kin_id = " + Trim(Str(!c_kin_id))

```

```

                    End If

```

```

                End If

```

```

                If Not IsNull(!c_assoc_kin_code) Then

```

```

                    If !c_assoc_kin_code > 0 Then

```

```

                        tTestStr = tTestStr + " AND c_assoc_kin_code = " + Trim(Str(!c_assoc_kin_code))

```

```

                        tTestStr = tTestStr + " AND c_assoc_kin_id = " + Trim(Str(!c_assoc_kin_id))

```

```

                    End If

```

```

                End If

```

```

            End If

```

```

            'MsgBox "about to look for string1: " + tTestStr

```

```

            gRstEdge.FindFirst tTestStr

```

```

            If Not gRstEdge.NoMatch Then

```

```

                tContinue = 0

```

```

            End If

```

```

            tTestStr = "c_person_id = " + Trim(Str(!c_node_id))

```

```

            tTestStr = tTestStr + " AND c_node_id = " + Trim(Str(!c_personid))

```

```

            tTestStr = tTestStr + " AND c_link_code = " + Trim(Str(!c_link_pair))

```

```

            If tType = "N" Then

```

```

                If Not IsNull(!c_assoc_kin_code) Then

```

```

                    If !c_assoc_kin_code > 0 Then

```

```

                        tTestStr = tTestStr + " AND c_kin_code = " + Trim(Str(!c_assoc_kin_code))

```

```

                        tTestStr = tTestStr + " AND c_kin_id = " + Trim(Str(!c_assoc_kin_id))

```

```

                    End If

```

```

                End If

```

```

                If Not IsNull(!c_kin_code) Then

```

```

                    If !c_kin_code > 0 Then

```

```

                        tTestStr = tTestStr + " AND c_assoc_kin_code = " + Trim(Str(!c_kin_code))

```

```

                tTestStr = tTestStr + " AND c_assoc_kin_id = " + Trim(Str(!c_kin_id))
            End If
        End If
    End If
    'MsgBox "about to look for string2: " + tTestStr

    gRstEdge.FindFirst tTestStr
    If Not gRstEdge.NoMatch Then
        tContinue = 0
    End If
    'MsgBox "Strings OK"
End If

If tContinue = 1 Then
    '
    ' add the record and fill in basic info
    '
    'MsgBox "Adding edge 1"
    gRstEdge.AddNew
    gRstEdge!c_person_id = !c_personid
    gRstEdge!c_name = !c_person_name
    gRstEdge!c_name_chn = !c_person_name_chn
    gRstEdge!c_index_year = !c_index_year
    gRstEdge!c_female = !c_female
    gRstEdge!c_link_type = tType
    If !c_personid = TxtID1.Value Or !c_personid = TxtID2.Value Then
        gRstEdge!c_edge_dist = 0
    Else
        gRstEdge!c_edge_dist = 1
    End If
    '
    'MsgBox "Adding edge 2"
    gRstEdge!c_link_code = !c_link_code
    gRstEdge!c_link_desc = !c_link_desc
    gRstEdge!c_link_chn = !c_link_chn
    '
    ' now check for kinship data
    '
    'MsgBox "Adding edge 3"
    gRstEdge!c_kin_desc = ""
    gRstEdge!c_kin_desc_chn = ""
    gRstEdge!c_kin_name = ""
    gRstEdge!c_kin_chn = ""
    gRstEdge!c_assoc_kin_desc = ""
    gRstEdge!c_assoc_kin_desc_chn = ""
    gRstEdge!c_assoc_kin_name = ""
    gRstEdge!c_assoc_kin_chn = ""

    If tType = "N" Then
        'MsgBox "Adding edge 4"
        If IsNull(!c_kin_id) Then
            gRstEdge!c_kin_id = 0
        Else
            gRstEdge!c_kin_id = !c_kin_id
        End If

        If IsNull(!c_kin_code) Then
            gRstEdge!c_kin_code = 0
        Else
            gRstEdge!c_kin_code = !c_kin_code
            If !c_kin_code > 0 Then
                gRstEdge!c_kin_desc = !c_kinrel
                gRstEdge!c_kin_desc_chn = !c_kinrel
                If !c_kin_id > 0 Then
                    gRstEdge!c_kin_name = !c_kin_name
                    gRstEdge!c_kin_chn = !c_kin_chn
                End If
            End If
        End If
    End If

    'MsgBox "Adding edge 5"
    If IsNull(!c_assoc_kin_code) Then
        gRstEdge!c_assoc_kin_code = 0
    Else
        If !c_assoc_kin_code > 0 Then
            gRstEdge!c_assoc_kin_id = !c_assoc_kin_id
            gRstEdge!c_assoc_kin_code = !c_assoc_kin_code
            gRstEdge!c_assoc_kin_desc = !c_assoc_kinrel
            gRstEdge!c_assoc_kin_desc_chn = !c_assoc_kinrel

```

```

        If !c_assoc_kin_id > 0 Then
            gRstEdge!c_assoc_kin_name = !c_assoc_kin_name
            gRstEdge!c_assoc_kin_chn = !c_assoc_kin_chn
        End If
    End If
End If
'
' now the address data
'
'MsgBox "Adding edge 6"
If IsNull(!c_addr_id) Then
    gRstEdge!c_addr_id = 0
    gRstEdge!c_addr_name = "[Unknown]"
    gRstEdge!c_addr_chn = "[Unknown]"
    gRstEdge!c_addr_type = 0
    gRstEdge!c_addr_desc = "[Unknown]"
    gRstEdge!c_addr_desc_chn = "[Unknown]"
    gRstEdge!x_coord = 0#
    gRstEdge!y_coord = 0#
Else
    gRstEdge!c_addr_id = !c_addr_id
    gRstEdge!c_addr_name = !c_addr_name
    gRstEdge!c_addr_chn = !c_addr_chn
    gRstEdge!c_addr_type = !c_addr_type
    gRstEdge!c_addr_desc = !c_addr_desc
    gRstEdge!c_addr_desc_chn = !c_addr_desc_chn
    gRstEdge!x_coord = !x_coord
    gRstEdge!y_coord = !y_coord
End If
'
' now repeat the process for the associated person
'
' fill in basic info for the associate
'
'MsgBox "Adding edge 7"
gRstEdge!c_node_id = !c_node_id
'
gRstEdge!c_node_name = !c_node_name
gRstEdge!c_node_chn = !c_node_chn
gRstEdge!c_node_index_year = !c_node_index_year
gRstEdge!c_node_female = !c_node_female
'
' now the address data
'
'MsgBox "Adding edge 8"
If IsNull(!c_node_addr_id) Then
    gRstEdge!c_node_addr_id = 0
    gRstEdge!c_node_addr_name = "[Unknown]"
    gRstEdge!c_node_addr_chn = "[Unknown]"
    gRstEdge!c_node_addr_type = 0
    gRstEdge!c_node_addr_desc = "[Unknown]"
    gRstEdge!c_node_addr_desc_chn = "[Unknown]"
    gRstEdge!node_xcoord = 0#
    gRstEdge!node_ycoord = 0#
Else
    gRstEdge!c_node_addr_id = !c_node_addr_id
    gRstEdge!c_node_addr_name = !c_node_addr_name
    gRstEdge!c_node_addr_chn = !c_node_addr_chn
    gRstEdge!c_node_addr_type = !c_node_addr_type
    gRstEdge!c_node_addr_desc = !c_node_addr_desc
    gRstEdge!c_node_addr_desc_chn = !c_node_addr_desc_chn
    gRstEdge!node_xcoord = !node_xcoord
    gRstEdge!node_ycoord = !node_ycoord
End If
'
' finally, get the supplemental data if there is any
'
'MsgBox "About to add extra data 9"
If tType = "N" Then
    If Not IsNull(!c_litgenre_code) Then
        gRstEdge!c_litgenre_desc = !c_lit_genre_desc
        gRstEdge!c_litgenre_desc_chn = !c_lit_genre_desc_chn
    End If
    '
    If Not IsNull(!c_occasion_code) Then
        gRstEdge!c_occasion_desc = !c_occasion_desc
        gRstEdge!c_occasion_desc_chn = !c_occasion_desc_chn
    End If
    '

```

```

        If Not IsNull(!c_topic_code) Then
            gRstEdge!c_topic_desc = !c_topic_desc
            gRstEdge!c_topic_desc_chn = !c_topic_desc_chn
        End If
        ,
        If Not IsNull(!c_inst_code) Then
            gRstEdge!c_inst_name_py = !c_inst_name_py
            gRstEdge!c_inst_name_hz = !c_inst_name_hz
        End If
        ,
        If Not IsNull(!c_text_title) Then
            gRstEdge!c_text_title = !c_text_title
        End If
        ,
        If Not IsNull(!c_assoc_claimer_id) Then
            gRstEdge!c_assoc_claimer_name = !c_assoc_claimer_name
            gRstEdge!c_assoc_claimer_name_chn = !c_assoc_claimer_chn
        End If
    End If
    ,
    gRstEdge.Update
End If
.MoveNext
Loop
End With

End Sub

Private Sub process_people()
    Dim tRstNonkinBiogAddr As DAO.Recordset

    With gRstEdge
        .MoveFirst
        Do While Not .EOF
            ' do we have c_personid?
            gRstPeople.FindFirst "c_person_id = " + Trim(Str(!c_person_id))
            If gRstPeople.NoMatch Then
                gRstPeople.AddNew
                gRstPeople!c_person_id = !c_person_id
                gRstPeople!c_name = !c_name
                gRstPeople!c_name_chn = !c_name_chn
                gRstPeople!c_index_year = !c_index_year
                gRstPeople!c_female = !c_female
                gRstPeople!c_addr_id = !c_addr_id
                gRstPeople!c_addr_name = !c_addr_name
                gRstPeople!c_addr_chn = !c_addr_chn
                gRstPeople!c_addr_type = !c_addr_type
                gRstPeople!c_addr_desc = !c_addr_desc
                gRstPeople!c_addr_desc_chn = !c_addr_desc_chn
                gRstPeople!x_coord = !x_coord
                gRstPeople!y_coord = !y_coord
                If !c_person_id = TxtID1.Value Or !c_person_id = TxtID2.Value Then
                    gRstPeople!c_node_dist = 0
                Else
                    gRstPeople!c_node_dist = 1
                End If
                gRstPeople.Update
            End If
            ,
            ' now look at the associate
            gRstPeople.FindFirst "c_person_id = " + Trim(Str(!c_node_id))
            If gRstPeople.NoMatch Then
                gRstPeople.AddNew
                gRstPeople!c_person_id = !c_node_id
                gRstPeople!c_name = !c_node_name
                gRstPeople!c_name_chn = !c_node_chn
                gRstPeople!c_index_year = !c_node_index_year
                gRstPeople!c_female = !c_node_female
                gRstPeople!c_addr_id = !c_node_addr_id
                gRstPeople!c_addr_name = !c_node_addr_name
                gRstPeople!c_addr_chn = !c_node_addr_chn
                gRstPeople!c_addr_type = !c_node_addr_type
                gRstPeople!c_addr_desc = !c_node_addr_desc
                gRstPeople!c_addr_desc_chn = !c_node_addr_desc_chn
                gRstPeople!x_coord = !node_xcoord
                gRstPeople!y_coord = !node_ycoord
                If !c_node_id = TxtID1.Value Or !c_node_id = TxtID2.Value Then
                    gRstPeople!c_node_dist = 0
                Else
                    gRstPeople!c_node_dist = 1
                End If
            End If
        End While
    End With

```

```

        End If
        gRstPeople.Update
    End If

    .MoveNext
Loop
End With
'
' it turns out that the base person can be left out, so double-check, and if needed, add.
'
Set tRstNonkinBiogAddr = CurrentDb.OpenRecordset("ZZZ_NONKIN_BIOG_ADDR", dbOpenTable)
tRstNonkinBiogAddr.Index = "person_id"

gRstPeople.FindFirst "c_person_id = " + Trim(Str(TxtID1.Value))
If gRstPeople.NoMatch Then
    '
    ' get the data from the base table
    '
    tRstNonkinBiogAddr.Seek "=", TxtID1.Value
    If Not tRstNonkinBiogAddr.NoMatch Then
        gRstPeople.AddNew
        gRstPeople!c_person_id = tRstNonkinBiogAddr!c_personid
        gRstPeople!c_name = tRstNonkinBiogAddr!c_person_name
        gRstPeople!c_name_chn = tRstNonkinBiogAddr!c_person_name_chn
        gRstPeople!c_index_year = tRstNonkinBiogAddr!c_index_year
        gRstPeople!c_female = tRstNonkinBiogAddr!c_female
        gRstPeople!c_addr_id = tRstNonkinBiogAddr!c_addr_id
        gRstPeople!c_addr_name = tRstNonkinBiogAddr!c_addr_name
        gRstPeople!c_addr_chn = tRstNonkinBiogAddr!c_addr_chn
        gRstPeople!c_addr_type = tRstNonkinBiogAddr!c_addr_type
        gRstPeople!c_addr_desc = tRstNonkinBiogAddr!c_addr_desc
        gRstPeople!c_addr_desc_chn = tRstNonkinBiogAddr!c_addr_desc_chn
        gRstPeople!x_coord = tRstNonkinBiogAddr!x_coord
        gRstPeople!y_coord = tRstNonkinBiogAddr!y_coord
        gRstPeople!c_node_dist = 0
        gRstPeople.Update
    End If
End If

gRstPeople.FindFirst "c_person_id = " + Trim(Str(TxtID2.Value))
If gRstPeople.NoMatch Then
    '
    ' get the data from the base table
    '
    tRstNonkinBiogAddr.Seek "=", TxtID2.Value
    If Not tRstNonkinBiogAddr.NoMatch Then
        gRstPeople.AddNew
        gRstPeople!c_person_id = tRstNonkinBiogAddr!c_personid
        gRstPeople!c_name = tRstNonkinBiogAddr!c_person_name
        gRstPeople!c_name_chn = tRstNonkinBiogAddr!c_person_name_chn
        gRstPeople!c_index_year = tRstNonkinBiogAddr!c_index_year
        gRstPeople!c_female = tRstNonkinBiogAddr!c_female
        gRstPeople!c_addr_id = tRstNonkinBiogAddr!c_addr_id
        gRstPeople!c_addr_name = tRstNonkinBiogAddr!c_addr_name
        gRstPeople!c_addr_chn = tRstNonkinBiogAddr!c_addr_chn
        gRstPeople!c_addr_type = tRstNonkinBiogAddr!c_addr_type
        gRstPeople!c_addr_desc = tRstNonkinBiogAddr!c_addr_desc
        gRstPeople!c_addr_desc_chn = tRstNonkinBiogAddr!c_addr_desc_chn
        gRstPeople!x_coord = tRstNonkinBiogAddr!x_coord
        gRstPeople!y_coord = tRstNonkinBiogAddr!y_coord
        gRstPeople!c_node_dist = 0
        gRstPeople.Update
    End If
End If

tRstNonkinBiogAddr.Close
Set tRstNonkinBiogAddr = Nothing

End Sub

Private Sub CmdUCINet_Click()
On Error GoTo Err_CmdUCINet_Click
'
' This program will dump the results of the search to a .vna file
'
' for the moment I'll just describe the format of the .vna file
'
' *node data
' ID index_year sex x_coord y_coord nodedist
' ID = str(c_person_id)
' indexyear = c_index_year INT

```

```

'         nodedist = c_node_dist INT
'         sex = c_female > (F,M)
' *node properties
' ID color shape size shortlabel active
'     color = red (1), orange (2), yellow (3), green (4), blue (5)
'     shortlabel = c_name
'     shape = 2
'     active = TRUE
'
' *tie data
' from to edgetype nodedist
'     from = str(c_person_id)
'     to = str(c_node_id)
'     edgetype= c_link_type (K,N)
'
' *tie properties
' from to color size active
'     from = str(c_person_id)
'     to = str(c_node_id)
'     color = red (255), orange (26367), yellow (65535), green (32768), blue (16711680)
'     size = 1-5 (the weight)
'
' the central question is whether to do distance optimizations
'
' first see if there are any records to process
'
If ZZ_SOCIAL_NETWORK.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUCINet_Click
End If
'
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUCINet_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstAssocType As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tSearchStr As String
Dim tColor(20) As String, tQuote As String
Dim tFileSystem, tVNA

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

' open the assoc type look-up table
Set tRstAssocType = CurrentDb.OpenRecordset("ASSOC_CODE_TYPE_REL", dbOpenDynaset)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network.vna"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdUCINet_Click
        Else
            ' make sure the file name has a vna extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".vna"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".vna") Then
                tFileName = tFileName + ".vna"
            End If
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    Set tFileSystem = CreateObject("Scripting.FileSystemObject")
    Set tVNA = tFileSystem.CreateTextFile(tFileName, True)

```

```

' define the colors for the nodes
'
tColor(1) = "0 "      ' black
tColor(2) = "16711680 " ' blue
tColor(3) = "32768 "  ' green
tColor(4) = "65535 "  ' yellow
tColor(5) = "26367 "  ' orange
For ti = 6 To 20
    tColor(ti) = "255 " ' red
Next
'
' process the two tables
'
Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
tQuote = Chr(34) ' the quotation mark
'
' first the nodes: define the node data structure
tVNA.WriteLine ("*node data")
tVNA.WriteLine ("ID index_year sex x_coord y_coord nodedist")
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' name = the ID of the person
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' indexyear = c_index_year INT
        If IsNull(!c_index_year) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + " "
        End If
        '
        ' sex = c_female > (F,M)
        If !c_female = -1 Then
            tStr = tStr + tQuote + "F" + tQuote + " "
        Else
            tStr = tStr + tQuote + "M" + tQuote + " "
        End If
        '
        ' x_coord
        If IsNull(!x_coord) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!x_coord)) + " "
        End If
        '
        ' y_coord
        If IsNull(!y_coord) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!y_coord)) + " "
        End If
        '
        ' node distance
        tStr = tStr + Trim(Str(!c_node_dist))
        '
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the node properties
'
' Note: ACTIVE removed as a property (MAF 2018/07/22)
'
tVNA.WriteLine ("*node properties")
tVNA.WriteLine ("ID color shape size shortlabel")
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' ID = the ID of the person
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' color = black (1), blue (2), green (3), yellow (4), orange (5)
        tStr = tStr + tColor(!c_node_dist + 1)
        '

```

```

' shape = 2? / size = 1?
tStr = tStr + "2 1 "
'
' shortlabel (+ Active = TRUE removed)
If IsNull(!c_name) Then
    tStr = tStr + "[Missing]"
Else
    tStr = tStr + tQuote + !c_name + tQuote
End If
tVNA.WriteLine (tStr)
.MoveNext
Loop
End With
'
' now the edges: define the record structure
'
tStr = "from to " + tQuote + "EdgeWeight" + tQuote + " " + tQuote + "edgetype"
tStr = tStr + tQuote + " " + tQuote + "edgelist" + tQuote
tVNA.WriteLine ("*tie data")
tVNA.WriteLine (tStr)
'
' For the moment, I am not combining parallel edges
'
With tRstEdge
.MoveNext
Do While Not .EOF
    '
    ' From = str(c_person_id) for node1
    tStr = Trim(Str(!c_person_id)) + " "
    '
    ' to = str(c_node_id) for node2
    tStr = tStr + Trim(Str(!c_node_id)) + " 1 "
    '
    ' edgetype
    If !c_link_type = "K" Then
        If IsNull(!c_link_desc) Then
            tStr = tStr + "K "
        Else
            tStr = tStr + tQuote + "K_" + !c_link_desc + tQuote + " "
        End If
    Else
        tSearchStr = "c_assoc_code = " + Trim(Str(!c_link_code))
        tRstAssocType.FindFirst tSearchStr
        If tRstAssocType.NoMatch Then
            tStr = tStr + "N_00 "
        Else
            tStr = tStr + "N_" + Trim(tRstAssocType!c_assoc_type_code) + " "
        End If
    End If
    '
    ' edgelist
    tStr = tStr + Trim(Str(!c_edge_dist))
    '
    tVNA.WriteLine (tStr)
    .MoveNext
Loop
End With
'
' now the edges properties
'
'tVNA.WriteLine ("*tie properties")
'tVNA.WriteLine ("from to color size active")

'With tRstEdge
.MoveNext
'Do While Not .EOF
    '
    ' from = str(c_person_id) for node1
    'tStr = Trim(Str(!c_person_id)) + " "
    '
    ' to = str(c_node_id) for node2
    'tStr = tStr + Trim(Str(!c_node_id)) + " 1 "
    '
    ' color = black (1), blue (2), green (3), yellow (4), orange (5)
    'tStr = tStr + tColor(!c_edge_dist)
    '
    ' size = 1? active = TRUE
    'tStr = tStr + "1 TRUE"
    '
    'tVNA.WriteLine (tStr)

```



```

        '.MoveNext
    'Loop
'End With
'
tVNA.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tVNA = Nothing
Set tFileSystem = Nothing
Set tRstAssocType = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdUCINet_Click:
Exit Sub

Err_CmdUCINet_Click:
MsgBox Err.Description
Resume Exit_CmdUCINet_Click

End Sub

Private Sub GetEdges()
    Dim tQueryStr As String, tQueryFromStr As String
    Dim tQueryWhereStr As String, tQueryAppendStr As String
    Dim cmdSQL As ADODB.Command, tRecDeleted As Long

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' define the query for appending to ZZ_SCRATCH_PEOPLE without duplication
    '
    tQueryAppendStr = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_IMPORT_PEOPLE.* " + _
        "FROM ZZ_SCRATCH_IMPORT_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
        "ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
        "WHERE (((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null))"
    '
    ' get the first-order linking people
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_female, " + _
        "c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, x_coord, y_coord, " + _
        "c_node_dist ) " + _
        "SELECT DISTINCT ZZZ_NONKIN_BIOG_ADDR.c_node_id, ZZZ_NONKIN_BIOG_ADDR.c_node_name, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_chn, ZZZ_NONKIN_BIOG_ADDR.c_node_index_year, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_female, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_addr_name, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_chn, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_addr_type, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc_chn, ZZZ_NONKIN_BIOG_ADDR.c_node_xcoord, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_ycoord, 1 AS c_node_dist " + _
        "FROM ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_1 ON " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_id = ZZZ_NONKIN_BIOG_ADDR_1.c_node_id " + _
        "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_personid)=" + tID1Str + ") AND " + _
        "((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)=" + TxtID2.Text + "))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' if kinship ties are to be included, get first-order kinship connections
    ' The situation is a bit more complicated than it might appear, since a kin of X may be an associate of Y
    ' or an associate of X may be a kin of Y or a kin of X may be a kin of Y:
    '
    '     x a=kin(x)   y=kin(a)      00
    '     x a=kin(x)   y=assoc(a)   01
    '     x a=assoc(x) y=kin(a)     10
    '
    '
    If ChkKinship.Value Then
        ' two strings remain fixed
        '
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
            "c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, " + _
            "x_coord, y_coord, c_node_dist ) " + _
            "SELECT DISTINCT ZABA_1.c_node_id, ZABA_1.c_node_name, " + _
            "ZABA_1.c_node_chn, ZABA_1.c_node_index_year, " + _

```

```

        "ZABA_1.c_node_female, ZABA_1.c_node_addr_id, " + _
        "ZABA_1.c_node_addr_name, ZABA_1.c_node_addr_chn, " + _
        "ZABA_1.c_node_addr_type, ZABA_1.c_node_addr_desc, " + _
        "ZABA_1.c_node_addr_desc_chn, ZABA_1.node_xcoord, " + _
        "ZABA_1.node_ycoord, 1 AS c_node_dist "
tQueryWhereStr = "WHERE (((ZABA_1.c_personid)=" + TxtID2.Text + ") AND " + _
        "((ZABA.c_personid)=" + tID1Str + "))"

' one changes

' kin-kin
tQueryFromStr = "FROM ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = ZABA_1.c_node_id "

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
' kin-assoc
tQueryFromStr = "FROM ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = ZABA_1.c_node_id "

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
' assoc-kin
tQueryFromStr = "FROM ZZZ_NONKIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = ZABA_1.c_node_id "

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
End If
'
' now get the second order linking people (first the first person in the pair, then the second)
' Add these to the temp file and copy only the non-duplicates
'
If Me.Chk2Nodes.Value Then
    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id, c_name, c_name_chn, " + _
        "c_index_year, c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, " + _
        "c_addr_desc_chn, x_coord, y_coord, c_node_dist ) " + _
        "SELECT DISTINCT ZZZ_NONKIN_BIOG_ADDR.c_node_id, ZZZ_NONKIN_BIOG_ADDR.c_node_name, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_chn, ZZZ_NONKIN_BIOG_ADDR.c_node_index_year, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_female, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_addr_name, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_chn, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_addr_type, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc_chn, ZZZ_NONKIN_BIOG_ADDR.node_xcoord, " + _
        "ZZZ_NONKIN_BIOG_ADDR.node_ycoord, 2 AS c_node_dist " + _
        "FROM (ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_1 " + _
        "ON ZZZ_NONKIN_BIOG_ADDR.c_node_id = ZZZ_NONKIN_BIOG_ADDR_1.c_node_id) " + _
        "INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_2 ON " + _
        "ZZZ_NONKIN_BIOG_ADDR_1.c_personid = ZZZ_NONKIN_BIOG_ADDR_2.c_node_id " + _
        "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_personid)=" + tID1Str + ") AND " + _
        "((ZZZ_NONKIN_BIOG_ADDR_2.c_personid)=" + TxtID2.Text + "))"

```

```

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
' now get the second person in the pair
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
" c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, " + _
" x_coord, y_coord, c_node_dist ) " + _
" SELECT DISTINCT ZZZ_NONKIN_BIOG_ADDR_1.c_personid, ZZZ_NONKIN_BIOG_ADDR_1.c_person_name, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_person_name_chn, ZZZ_NONKIN_BIOG_ADDR_1.c_index_year, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_female, ZZZ_NONKIN_BIOG_ADDR_1.c_addr_id, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_addr_name, ZZZ_NONKIN_BIOG_ADDR_1.c_addr_chn, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_addr_type, ZZZ_NONKIN_BIOG_ADDR_1.c_addr_desc, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_addr_desc_chn, ZZZ_NONKIN_BIOG_ADDR_1.x_coord, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.y_coord, 2 AS c_node_dist " + _
" FROM (ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_1 " + _
" ON ZZZ_NONKIN_BIOG_ADDR.c_node_id = ZZZ_NONKIN_BIOG_ADDR_1.c_node_id) " + _
" INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_2 ON " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_personid = ZZZ_NONKIN_BIOG_ADDR_2.c_node_id " + _
" WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_personid)=" + tID1Str + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_2.c_personid)=" + TxtID2.Text + ")))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_IMPORT_PEOPLE.* " + _
" FROM ZZ_SCRATCH_IMPORT_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
" ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
" WHERE (((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' if kinship ties are to be included, get second-order kinship connections.
' here, there are seven(!) possibilities that must considered:
'
' x a=kin(x) b=kin(a) y=kin(b) 000
' x a=kin(x) b=kin(a) y=assoc(b) 001
' x a=kin(x) b=assoc(a) y=kin(b) 010
' x a=kin(x) b=assoc(a) y=assoc(b) 011
' x a=assoc(x) b=kin(a) y=kin(b) 100
' x a=assoc(x) b=kin(a) y=assoc(b) 101
' x a=assoc(x) b=assoc(a) y=kin(b) 110
' Therefore we need to run seven queries by swapping out the tables we use
'
If ChkKinship.Value Then
'
' these two parts of the query stay the same
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
" c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, " + _
" x_coord, y_coord, c_node_dist ) " + _
" SELECT DISTINCT ZABA_1.c_personid, ZABA_1.c_person_name, " + _
" ZABA_1.c_person_name_chn, ZABA_1.c_index_year, " + _
" ZABA_1.c_female, ZABA_1.c_addr_id, " + _
" ZABA_1.c_addr_name, ZABA_1.c_addr_chn, " + _
" ZABA_1.c_addr_type, ZABA_1.c_addr_desc, " + _
" ZABA_1.c_addr_desc_chn, ZABA_1.x_coord, " + _
" ZABA_1.y_coord, 1 AS c_node_dist " + _
tQueryWhereStr = "WHERE (((ZABA.c_personid)=" + tID1Str + ") AND " + _
" ((ZABA_2.c_personid)=" + TxtID2.Text + ")))"

' what changes are the tables to be considered
'
' kin-kin-kin
tQueryFromStr = "FROM (ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
" AS ZABA_1 ON ZABA.c_node_id = " + _
" ZABA_1.c_node_id) INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_2 " + _
" ON ZABA_1.c_personid = ZABA_2.c_node_id "

```

```

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'   kin-kin-assoc
tQueryFromStr = "FROM (ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
    "AS ZABA_1 ON ZABA.c_node_id = " + _
    "ZABA_1.c_node_id) INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_2 " + _
    "ON ZABA_1.c_personid = ZABA_2.c_node_id "

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'   kin-assoc-kin
tQueryFromStr = "FROM (ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "AS ZABA_1 ON ZABA.c_node_id = " + _
    "ZABA_1.c_node_id) INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_2 " + _
    "ON ZABA_1.c_personid = ZABA_2.c_node_id "

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'   kin-assoc-assoc
tQueryFromStr = "FROM (ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
    "AS ZABA_1 ON ZABA.c_node_id = " + _
    "ZABA_1.c_node_id) INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_2 " + _
    "ON ZABA_1.c_personid = ZABA_2.c_node_id "

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'   assoc-kin-kin
tQueryFromStr = "FROM (ZZZ_NONKIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
    "AS ZABA_1 ON ZABA.c_node_id = " + _
    "ZABA_1.c_node_id) INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_2 " + _
    "ON ZABA_1.c_personid = ZABA_2.c_node_id "

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'   assoc-kin-assoc

```

```

tQueryFromStr = "FROM (ZZZ_NONKIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
    "AS ZABA_1 ON ZABA.c_node_id = " + _
    "ZABA_1.c_node_id) INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_2 " + _
    "ON ZABA_1.c_personid = ZABA_2.c_node_id "

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'

cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'

cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'

End If

End If

End Sub

Private Sub CmdImportList_Click()
On Error GoTo Err_CmdImport_Click
Dim stDocName As String, stLinkCriteria As String
Dim tRstPeople As DAO.Recordset, tRstImportPeople As DAO.Recordset
Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String, tLen As Integer

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tFileSystem, tList

'
' open the list

Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

'Use a With...End With block to reference the FileDialog object.

tFileName = ""

With dlgSaveAs
.InitialFileName = ""
If .Show = -1 Then
'
For Each tFN In .SelectedItems
tFileName = tFN
If Not tFileName = "" Then
Exit For
End If
Next
If tFileName = "" Then
MsgBox "Bad file Name."
GoTo Exit_CmdImport_Click
End If
End If
End With

' Clear the people table now that we are ready to go
'

If Not (tFileName = "") Then
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
'

cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "ImportPeopleList_Space", "TempImportList", tFileName, 0
' TransferType=acImportDelim
' SpecificationName = "TempImportList" (apparently it is saved in the database itself)
' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
' HasFieldNames = False (0)
'

```

```

' copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
"FROM BIOG_MAIN RIGHT JOIN TempImportList ON BIOG_MAIN.c_personid = TempImportList.ImportID " + _
"WHERE ((BIOG_MAIN.c_personid) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If
'
' copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id ) SELECT DISTINCT TempImportList.ImportID "
+ _
"FROM BIOG_MAIN INNER JOIN TempImportList ON BIOG_MAIN.c_personid = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted = 0 Then
    TxtPerson1.Value = "[Error]"
    TxtPerson1Chn.Value = "[Error]"
    TxtPerson2.Value = "[Error]"
    TxtPerson2Chn.Value = "[Error]"

    CmdQuery.Enabled = False
Else
    TxtPerson1.Value = "[Imported List]"
    TxtPerson1Chn.Value = "[Imported List]"
    TxtPerson2.Value = "[Imported List]"
    TxtPerson2Chn.Value = "[Imported List]"

    CmdQuery.Enabled = True
    CmdImportList.Enabled = False
    CmdClearList.Enabled = True
    CmdPickPerson1.Enabled = False
    CmdPickPerson2.Enabled = False
End If

Set cmdSQL = Nothing
Set tFileSystem = Nothing
End If

Exit_CmdImport_Click:
Exit Sub

Err_CmdImport_Click:
MsgBox Err.Description
Resume Exit_CmdImport_Click

End Sub
Sub CmdClearList_Click()
    TxtPerson1.Value = ""
    TxtPerson1Chn.Value = ""
    TxtPerson2.Value = ""
    TxtPerson2Chn.Value = ""

    CmdClearList.Enabled = False
    CmdImportList.Enabled = True
    CmdPickPerson1.Enabled = True
    CmdPickPerson2.Enabled = True
    CmdQuery.Enabled = False
End Sub

Sub CmdQueryOld()
    Dim tRstDummy As DAO.Recordset, tContinue As Integer, tQueryStr As String, tQueryFromStr As String
    Dim tQueryWhereStr As String, tQueryAppendStr As String, tQueryIntoStr As String
    Dim tQuerySelectStr As String, tQuery1stStr As String, tQuery2ndStr As String
    Dim tQuery1stWhereStr As String, tQuery2ndWhereStr As String
    Dim tID1Str As String, tID2Str As String, tFromStr As String, tToStr As String
    Dim cmdSQL As ADODB.Command, tRecDeleted As Long, tCurADDRBookmark As Variant

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' to clear the table, close and then delete records

```

```

'
Set ZZ_SOCIAL_NETWORK.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SN", dbOpenDynaset)
'
cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
' now the people table
'
Set ZZ_SCRATCH_PEOPLE.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SP", dbOpenDynaset)
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted
'
' get the index year constraints
'
If ChkIndexYears.Value Then
    TxtFromYear.SetFocus
    tFromStr = Trim(Str(TxtFromYear.Value))
    TxtToYear.SetFocus
    tToStr = Trim(Str(TxtToYear.Value))
End If

' define the query for appending to ZZ_SCRATCH_PEOPLE without duplication
'
tQueryAppendStr = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_PAIR_PEOPLE.* " + _
    "FROM ZZ_SCRATCH_PAIR_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
    "ZZ_SCRATCH_PAIR_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
    "WHERE ((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null)"
'
' first add the target people (if CmdClearList is enabled, ImportList was successful)
'
If CmdClearList.Enabled Then
    ' MsgBox "Using list"
    tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
        "c_female, c_addr_id, c_addr_type, c_addr_desc, c_addr_desc_chn, c_addr_name, c_addr_chn, " + _
        "x_coord, y_coord, c_node_dist ) " + _
        "SELECT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, " + _
        "ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MAIN.c_index_year, " + _
        "ZZZ_BIOG_MAIN.c_female, ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_type_code, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_desc, ZZZ_BIOG_MAIN.c_index_addr_type_chn, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_MAIN.c_index_addr_chn, ZZZ_BIOG_MAIN.x_coord, " + _
        "ZZZ_BIOG_MAIN.y_coord, 0 AS c_node_dist " + _
        "FROM ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZZ_BIOG_MAIN ON " + _
        "ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZZ_BIOG_MAIN.c_personid"
Else
    ' get the ID strings for the two people
    '
    Me.TxtID1.Visible = True
    Me.TxtID1.SetFocus
    tID1Str = Trim(Str(TxtID1.Value))
    Me.TxtID2.Visible = True
    Me.TxtID2.SetFocus
    tID2Str = Trim(Str(TxtID2.Value))
    Me.CmdQuery.SetFocus
    Me.TxtID1.Visible = False
    Me.TxtID2.Visible = False

    ' MsgBox "Using people pairs"
    tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
        "c_female, c_addr_id, c_addr_type, c_addr_desc, c_addr_desc_chn, c_addr_name, c_addr_chn, x_coord, " + _
        "y_coord, c_node_dist ) " + _
        "SELECT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, " + _
        "ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MAIN.c_index_year, ZZZ_BIOG_MAIN.c_female, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_type_code, ZZZ_BIOG_MAIN.c_index_addr_type_desc, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_chn, ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_MAIN.c_index_addr_chn, " + _
        "ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord, 0 AS c_node_dist " + _
        "FROM ZZZ_BIOG_MAIN " + _
        "WHERE (((ZZZ_BIOG_MAIN.c_personid)=" + tID1Str + _
        " Or (ZZZ_BIOG_MAIN.c_personid)=" + tID2Str + "))"
End If
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' get the first-order linking people
'

```

```

tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_female, " + _
"c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, x_coord, y_coord, " + _
"c_node_dist ) " + _
"SELECT DISTINCT ZZZ_NONKIN_BIOG_ADDR.c_node_id, ZZZ_NONKIN_BIOG_ADDR.c_node_name, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_chn, ZZZ_NONKIN_BIOG_ADDR.c_node_index_year, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_female, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_name, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_chn, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_type, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc_chn, ZZZ_NONKIN_BIOG_ADDR.c_node_xcoord, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_ycoord, 1 AS c_node_dist "

If CmdClearList.Enabled Then
    If ChkIndexYears.Value Then
        tQueryWhereStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 " + _
"INNER JOIN (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN (ZZZ_NONKIN_BIOG_ADDR " + _
"INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_1 ON " + _
"ZZZ_NONKIN_BIOG_ADDR_1.c_node_id = ZZZ_NONKIN_BIOG_ADDR_1.c_node_id) ON " + _
"ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR_1.c_personid) ON " + _
"ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR_1.c_personid " + _
"WHERE (((ZZZ_NONKIN_BIOG_ADDR_1.c_node_index_year) >=" + tFromStr + _
" And (ZZZ_NONKIN_BIOG_ADDR_1.c_node_index_year) <=" + tToStr + ") " + _
"And ((ZZZ_NONKIN_BIOG_ADDR_1.c_personid) > [ZZZ_NONKIN_BIOG_ADDR].[c_personid]))"

    Else
        tQueryWhereStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 " + _
"INNER JOIN (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN (ZZZ_NONKIN_BIOG_ADDR " + _
"INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_1 ON " + _
"ZZZ_NONKIN_BIOG_ADDR_1.c_node_id = ZZZ_NONKIN_BIOG_ADDR_1.c_node_id) ON " + _
"ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR_1.c_personid) ON " + _
"ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR_1.c_personid " + _
"WHERE (((ZZZ_NONKIN_BIOG_ADDR_1.c_personid) > [ZZZ_NONKIN_BIOG_ADDR].[c_personid]))"

    End If
Else
    If ChkIndexYears.Value Then
        tQueryWhereStr = "FROM ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS " + _
"ZZZ_NONKIN_BIOG_ADDR_1 ON ZZZ_NONKIN_BIOG_ADDR.c_node_id = ZZZ_NONKIN_BIOG_ADDR_1.c_node_id " + _
"WHERE (((ZZZ_NONKIN_BIOG_ADDR_1.c_node_index_year) >=" + tFromStr + _
" And (ZZZ_NONKIN_BIOG_ADDR_1.c_node_index_year) <=" + tToStr + _
") AND ((ZZZ_NONKIN_BIOG_ADDR_1.c_personid) = " + tID2Str + _
") AND ((ZZZ_NONKIN_BIOG_ADDR.c_personid) = " + tID1Str + "))"

    Else
        tQueryWhereStr = "FROM ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS " + _
"ZZZ_NONKIN_BIOG_ADDR_1 ON ZZZ_NONKIN_BIOG_ADDR.c_node_id = ZZZ_NONKIN_BIOG_ADDR_1.c_node_id " + _
"WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_personid) = " + tID1Str + ") AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_1.c_personid) = " + tID2Str + "))"

    End If
End If
'
cmdSQL.CommandText = tQueryStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' if kinship ties are to be included, get first-order kinship connections
' The situation is a bit more complicated than it might appear, since a kin of X may be an associate of Y
' or an associate of X may be a kin of Y or a kin of X may be a kin of Y:
'     x a=kin(x)   y=kin(a)    00
'     x a=kin(x)   y=assoc(a)  01
'     x a=assoc(x) y=kin(a)    10
'
If ChkKinship.Value Then
    '
    ' two strings remain fixed
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
"c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, " + _
"x_coord, y_coord, c_node_dist ) " + _
"SELECT DISTINCT ZABA_1.c_node_id, ZABA_1.c_node_name, " + _
"ZABA_1.c_node_chn, ZABA_1.c_node_index_year, " + _
"ZABA_1.c_node_female, ZABA_1.c_node_addr_id, " + _
"ZABA_1.c_node_addr_name, ZABA_1.c_node_addr_chn, " + _
"ZABA_1.c_node_addr_type, ZABA_1.c_node_addr_desc, " + _
"ZABA_1.c_node_addr_desc_chn, ZABA_1.c_node_xcoord, " + _
"ZABA_1.c_node_ycoord, 1 AS c_node_dist "

    If CmdClearList.Enabled Then
        If ChkIndexYears.Value Then
            tQueryWhereStr = "WHERE (((ZABA_1.c_node_index_year) >=" + tFromStr + _
" And (ZABA_1.c_node_index_year) <=" + tToStr + ") " + _
"AND ((ZABA_1.c_personid) > [ZABA].[c_personid]))"

        Else
            '
        End If
    End If
End If

```



```

        tQueryWhereStr = "WHERE (((ZABA_1.c_personid)>" + _
            "[ZABA].[c_personid]))"
    End If
Else
    If ChkIndexYears.Value Then
        tQueryWhereStr = "WHERE (((ZABA_1.c_node_index_year)>=" + tFromStr + _
            " And (ZABA_1.c_node_index_year)<=" + tToStr + _
            ") AND ((ZABA_1.c_personid)=" + tID2Str + _
            ") AND ((ZABA.c_personid)=" + tID1Str + "))"
    Else
        tQueryWhereStr = "WHERE (((ZABA_1.c_personid)=" + tID2Str + ") AND " + _
            "((ZABA.c_personid)=" + tID1Str + "))"
    End If
End If

' one changes

' kin-kin
If CmdClearList.Enabled Then
    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 " + _
        "INNER JOIN (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN (ZZZ_KIN_BIOG_ADDR AS " + _
        "ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_1 " + _
        "ON ZABA.c_node_id = ZABA_1.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_1.c_personid "
Else
    tQueryFromStr = "FROM ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = ZABA_1.c_node_id "
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
' kin-assoc
If CmdClearList.Enabled Then
    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 " + _
        "INNER JOIN (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN (ZZZ_KIN_BIOG_ADDR AS " + _
        "ZABA INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_1 " + _
        "ON ZABA.c_node_id = ZABA_1.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_1.c_personid "
Else
    tQueryFromStr = "FROM ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = ZABA_1.c_node_id "
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted

' assoc-kin
If CmdClearList.Enabled Then
    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 " + _
        "INNER JOIN (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN (ZZZ_NONKIN_BIOG_ADDR AS " + _
        "ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_1 " + _
        "ON ZABA.c_node_id = ZABA_1.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_1.c_personid "
Else
    tQueryFromStr = "FROM ZZZ_NONKIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = ZABA_1.c_node_id "
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted

```

```

'
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
End If
'
'   now get the second order linking people (first the first person in the pair, then the second)
'   Add these to the temp file and copy only the non-duplicates
'
If Me.Chk2Nodes.Value Then
MsgBox "Warning: the two-node routine takes a while. Don't Panic. Click to start."
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name, c_name_chn, " + _
"c_index_year, c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, " + _
"c_addr_desc_chn, x_coord, y_coord, c_node_dist ) " + _
"SELECT DISTINCT ZZZ_NONKIN_BIOG_ADDR.c_node_id, ZZZ_NONKIN_BIOG_ADDR.c_node_name, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_chn, ZZZ_NONKIN_BIOG_ADDR.c_node_index_year, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_female, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_name, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_chn, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_type, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc, " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_addr_desc_chn, ZZZ_NONKIN_BIOG_ADDR.node_xcoord, " + _
"ZZZ_NONKIN_BIOG_ADDR.node_ycoord, 2 AS c_node_dist "

If CmdClearList.Enabled Then
tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 INNER JOIN " + _
"(ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ((ZZZ_NONKIN_BIOG_ADDR INNER JOIN " + _
"ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_1 ON " + _
"ZZZ_NONKIN_BIOG_ADDR.c_node_id = ZZZ_NONKIN_BIOG_ADDR_1.c_node_id) INNER JOIN " + _
"ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_2 ON " + _
"ZZZ_NONKIN_BIOG_ADDR_1.c_personid = ZZZ_NONKIN_BIOG_ADDR_2.c_node_id) " + _
"ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZZ_NONKIN_BIOG_ADDR.c_personid) " + _
"ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZZZ_NONKIN_BIOG_ADDR_2.c_personid "

If ChkIndexYears.Value Then
tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_node_id)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE_1].[c_person_id]) AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id] AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_node_index_year)>=" + tFromStr + " And " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_node_index_year)<=" + tToStr + ")))"
Else
tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_node_id)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE_1].[c_person_id]) AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id] AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]))"
End If
Else
tQueryFromStr = "FROM (ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS " + _
"ZZZ_NONKIN_BIOG_ADDR_1 ON ZZZ_NONKIN_BIOG_ADDR.c_node_id = ZZZ_NONKIN_BIOG_ADDR_1.c_node_id) " + _
"INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZZZ_NONKIN_BIOG_ADDR_2 ON " + _
"ZZZ_NONKIN_BIOG_ADDR_1.c_personid = ZZZ_NONKIN_BIOG_ADDR_2.c_node_id "

If ChkIndexYears.Value Then
tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)<>" + tID1Str + ") AND " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_node_index_year)>=" + tFromStr + " And " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_node_index_year)<=" + tToStr + ") AND " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_personid)=" + tID1Str + ") AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_personid)=" + tID2Str + ") AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_node_id)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_node_id)<>" + tID2Str + ")))"
Else
tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)<>" + tID1Str + ") AND " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_personid)=" + tID1Str + ") AND " + _

```

```

"((ZZZ_NONKIN_BIOG_ADDR_2.c_personid)=" + tID2Str + ") AND " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)>0 And " + _
"((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
"((ZZZ_NONKIN_BIOG_ADDR.c_node_id)>0 And (ZZZ_NONKIN_BIOG_ADDR.c_node_id)<>" + tID2Str + "))"

End If
End If

cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
' now get the second person in the pair
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
" c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, " + _
" x_coord, y_coord, c_node_dist ) " + _
"SELECT DISTINCT ZZZ_NONKIN_BIOG_ADDR_1.c_personid, ZZZ_NONKIN_BIOG_ADDR_1.c_person_name, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_person_name_chn, ZZZ_NONKIN_BIOG_ADDR_1.c_index_year, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_female, ZZZ_NONKIN_BIOG_ADDR_1.c_addr_id, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_addr_name, ZZZ_NONKIN_BIOG_ADDR_1.c_addr_chn, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_addr_type, ZZZ_NONKIN_BIOG_ADDR_1.c_addr_desc, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.c_addr_desc_chn, ZZZ_NONKIN_BIOG_ADDR_1.x_coord, " + _
" ZZZ_NONKIN_BIOG_ADDR_1.y_coord, 2 AS c_node_dist "

If CmdClearList.Enabled Then
    If ChkIndexYears.Value Then
        tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_node_id)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE_1].[c_person_id]) AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR_1.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
" (ZZZ_NONKIN_BIOG_ADDR_2.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_1.c_index_year)>=" + tFromStr + " And " + _
" (ZZZ_NONKIN_BIOG_ADDR_1.c_index_year)<=" + tToStr + "))"
    Else
        tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_node_id)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE_1].[c_person_id]) AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
" (ZZZ_NONKIN_BIOG_ADDR_1.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
" (ZZZ_NONKIN_BIOG_ADDR_2.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]))"
    End If
Else
    If ChkIndexYears.Value Then
        tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR_1.c_personid)<>" + tID1Str + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_1.c_index_year)>=" + tFromStr + " And " + _
" (ZZZ_NONKIN_BIOG_ADDR_1.c_index_year)<=" + tToStr + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR.c_personid)=" + tID1Str + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_2.c_personid)=" + tID2Str + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR.c_node_id)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR.c_node_id)<>" + tID2Str + "))"
    Else
        tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR_1.c_personid)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR_1.c_personid)<>" + tID1Str + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR.c_personid)=" + tID1Str + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_2.c_personid)=" + tID2Str + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR.c_node_id)>0 And (ZZZ_NONKIN_BIOG_ADDR.c_node_id)<>" + tID2Str + "))"
    End If
End If
cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_PAIR_PEOPLE.* " + _
"FROM ZZ_SCRATCH_PAIR_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
"ZZ_SCRATCH_PAIR_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
"WHERE (((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null))"
'
cmdSQL.CommandText = tQueryStr

```

```

cmdSQL.Execute tRecDeleted
'
' if kinship ties are to be included, get second-order kinship connections.
' here, there are seven(!) possibilities that must be considered:
'   x a=kin(x) b=kin(a) y=kin(b)      000
'   x a=kin(x) b=kin(a) y=assoc(b)    001
'   x a=kin(x) b=assoc(a) y=kin(b)    010
'   x a=kin(x) b=assoc(a) y=assoc(b)  011
'   x a=assoc(x) b=kin(a) y=kin(b)    100
'   x a=assoc(x) b=kin(a) y=assoc(b)  101
'   x a=assoc(x) b=assoc(a) y=kin(b)  110
' Therefore we need to run seven queries by swapping out the tables we use
'
If ChkKinship.Value Then
'
'   these two parts of the query stay the same
'
tQuery1stStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
+ -
" c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, " + _
" x_coord, y_coord, c_node_dist ) " + _
" SELECT DISTINCT ZABA_1.c_personid, ZABA_1.c_person_name, " + _
" ZABA_1.c_person_name_chn, ZABA_1.c_index_year, " + _
" ZABA_1.c_female, ZABA_1.c_addr_id, " + _
" ZABA_1.c_addr_name, ZABA_1.c_addr_chn, " + _
" ZABA_1.c_addr_type, ZABA_1.c_addr_desc, " + _
" ZABA_1.c_addr_desc_chn, ZABA_1.x_coord, " + _
" ZABA_1.y_coord, 1 AS c_node_dist "

tQuery2ndStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name, c_name_chn, " + _
" c_index_year, c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, " + _
" c_addr_desc_chn, x_coord, y_coord, c_node_dist ) " + _
" SELECT DISTINCT ZABA.c_node_id, ZABA.c_node_name, " + _
" ZABA.c_node_chn, ZABA.c_node_index_year, " + _
" ZABA.c_node_female, ZABA.c_node_addr_id, " + _
" ZABA.c_node_addr_name, ZABA.c_node_addr_chn, " + _
" ZABA.c_node_addr_type, ZABA.c_node_addr_desc, " + _
" ZABA.c_node_addr_desc_chn, ZABA.node_xcoord, " + _
" ZABA.node_ycoord, 2 AS c_node_dist "

If CmdClearList.Enabled Then
If ChkIndexYears.Value Then
tQuery1stWhereStr = "WHERE (((ZABA.c_node_id)>0 And " + _
" (ZABA.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE_1].[c_person_id]) AND " + _
" ((ZABA_1.c_personid)>0 And " + _
" (ZABA_1.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
" (ZABA_1.c_index_year)>=" + tFromStr + " And " + _
" (ZABA_1.c_index_year)<=" + tToStr + "))"
tQuery2ndWhereStr = "WHERE (((ZABA.c_node_id)>0 And " + _
" (ZABA.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE_1].[c_person_id]) AND " + _
" ((ZABA_1.c_personid)>0 And " + _
" (ZABA_1.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
" (ZABA.c_node_index_year)>=" + tFromStr + " And " + _
" (ZABA.c_node_index_year)<=" + tToStr + "))"
Else
tQuery1stWhereStr = "WHERE (((ZABA.c_node_id)>0 And " + _
" (ZABA.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE_1].[c_person_id]) AND " + _
" ((ZABA_1.c_personid)>0 And " + _
" (ZABA_1.c_personid)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]))"
tQuery2ndWhereStr = tQuery1stWhereStr
End If
Else
If ChkIndexYears.Value Then
tQuery1stWhereStr = "WHERE (((ZABA_1.c_personid)>0 And " + _
" (ZABA_1.c_personid)<>" + tID1Str + ") AND " + _
" ((ZABA_1.c_index_year)>=" + tFromStr + " And " + _
" (ZABA_1.c_index_year)<=" + tToStr + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
" ((ZABA.c_personid)=" + tID1Str + ") AND " + _
" ((ZABA_2.c_personid)=" + tID2Str + ") AND " + _
" ((ZABA.c_node_id)>0 And " + _
" (ZABA.c_node_id)<>" + tID2Str + "))"
tQuery2ndWhereStr = "WHERE (((ZABA_1.c_personid)>0 And " + _
" (ZABA_1.c_personid)<>" + tID1Str + ") AND " + _
" ((ZABA.c_node_index_year)>=" + tFromStr + " And " + _
" (ZABA.c_node_index_year)<=" + tToStr + ") AND " + _
" ((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)>0 And " + _
" (ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
" ((ZABA.c_personid)=" + tID1Str + ") AND " + _

```

```

        "((ZABA_2.c_personid)=" + tID2Str + ") AND " + _
        "((ZABA.c_node_id)>0 And " + _
        "(ZABA.c_node_id)<>" + tID2Str + ")))"
    Else
        tQuery1stWhereStr = "WHERE (((ZABA_1.c_personid)>0 And " + _
            "(ZABA_1.c_personid)<>" + tID1Str + ") AND " + _
            "((ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)>0 And " + _
            "(ZZZ_NONKIN_BIOG_ADDR_2.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id]) AND " + _
            "((ZABA.c_personid)=" + tID1Str + ") AND " + _
            "((ZABA_2.c_personid)=" + tID2Str + ") AND " + _
            "((ZABA.c_node_id)>0 And " + _
            "(ZABA.c_node_id)<>" + tID2Str + ")))"
        tQuery2ndWhereStr = tQuery1stWhereStr
    End If
End If

'tQueryWhereStr = "WHERE (((ZABA.c_personid)=" + tID1Str + ") AND " + _
'    "((ZABA_2.c_personid)=" + tID2Str + ")))"

'    what changes are the tables to be considered

'    kin-kin-kin
If CmdClearList.Enabled Then
    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 INNER JOIN " + _
        "(ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ((ZZZ_KIN_BIOG_ADDR AS ZABA " + _
        "INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_1 ON " + _
        "ZABA.c_node_id = ZABA_1.c_node_id) INNER JOIN " + _
        "ZZZ_KIN_BIOG_ADDR AS ZABA_2 ON " + _
        "ZABA_1.c_personid = ZABA_2.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_2.c_personid "
Else
    tQueryFromStr = "FROM (ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = " + _
        "ZABA_1.c_node_id) INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_2 " + _
        "ON ZABA_1.c_personid = ZABA_2.c_node_id "
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery1stStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'    copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'    get second person
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery2ndStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'    copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'    kin-kin-assoc
If CmdClearList.Enabled Then
    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 INNER JOIN " + _
        "(ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ((ZZZ_KIN_BIOG_ADDR AS ZABA " + _
        "INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_1 ON " + _
        "ZABA.c_node_id = ZABA_1.c_node_id) INNER JOIN " + _
        "ZZZ_NONKIN_BIOG_ADDR AS ZABA_2 ON " + _
        "ZABA_1.c_personid = ZABA_2.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_2.c_personid "
Else
    tQueryFromStr = "FROM (ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = " + _
        "ZABA_1.c_node_id) INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_2 " + _
        "ON ZABA_1.c_personid = ZABA_2.c_node_id "
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"

```

```

cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery1stStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'   get second person
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery2ndStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'
'   kin-assoc-kin
If CmdClearList.Enabled Then

```

```

    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 INNER JOIN " + _
        "(ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ((ZZZ_KIN_BIOG_ADDR AS ZABA " + _
        "INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_1 ON " + _
        "ZABA.c_node_id = ZABA_1.c_node_id) INNER JOIN " + _
        "ZZZ_KIN_BIOG_ADDR AS ZABA_2 ON " + _
        "ZABA_1.c_personid = ZABA_2.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_2.c_personid "

```

```
Else

```

```

    tQueryFromStr = "FROM (ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = " + _
        "ZABA_1.c_node_id) INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_2 " + _
        "ON ZABA_1.c_personid = ZABA_2.c_node_id "

```

```
End If

```

```

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery1stStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'   get second person
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery2ndStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'
'   kin-assoc-assoc
If CmdClearList.Enabled Then

```

```

    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 INNER JOIN " + _
        "(ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ((ZZZ_KIN_BIOG_ADDR AS ZABA " + _
        "INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_1 ON " + _
        "ZABA.c_node_id = ZABA_1.c_node_id) INNER JOIN " + _
        "ZZZ_NONKIN_BIOG_ADDR AS ZABA_2 ON " + _
        "ZABA_1.c_personid = ZABA_2.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_2.c_personid "

```

```
Else

```

```

    tQueryFromStr = "FROM (ZZZ_KIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = " + _
        "ZABA_1.c_node_id) INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_2 " + _

```

```

        "ON ZABA_1.c_personid = ZABA_2.c_node_id "
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery1stStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'   get second person
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery2ndStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'
'   assoc-kin-kin
If CmdClearList.Enabled Then
    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 INNER JOIN " + _
        "(ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ((ZZZ_NONKIN_BIOG_ADDR AS ZABA " + _
        "INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_1 ON " + _
        "ZABA.c_node_id = ZABA_1.c_node_id) INNER JOIN " + _
        "ZZZ_KIN_BIOG_ADDR AS ZABA_2 ON " + _
        "ZABA_1.c_personid = ZABA_2.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_2.c_personid "
Else
    tQueryFromStr = "FROM (ZZZ_NONKIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = " + _
        "ZABA_1.c_node_id) INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_2 " + _
        "ON ZABA_1.c_personid = ZABA_2.c_node_id "
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery1stStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'   get second person
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery2ndStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
'   copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'
'   assoc-kin-assoc
If CmdClearList.Enabled Then
    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 INNER JOIN " + _
        "(ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ((ZZZ_NONKIN_BIOG_ADDR AS ZABA " + _
        "INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_1 ON " + _
        "ZABA.c_node_id = ZABA_1.c_node_id) INNER JOIN " + _
        "ZZZ_NONKIN_BIOG_ADDR AS ZABA_2 ON " + _
        "ZABA_1.c_personid = ZABA_2.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_2.c_personid "

```

```

Else
    tQueryFromStr = "FROM (ZZZ_NONKIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_KIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = " + _
        "ZABA_1.c_node_id) INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_2 " + _
        "ON ZABA_1.c_personid = ZABA_2.c_node_id "
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery1stStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
' get second person
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery2ndStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
'
'
' assoc-assoc-kin
If CmdClearList.Enabled Then
    tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 INNER JOIN " + _
        "(ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ((ZZZ_NONKIN_BIOG_ADDR AS ZABA " + _
        "INNER JOIN ZZZ_NONKIN_BIOG_ADDR AS ZABA_1 ON " + _
        "ZABA.c_node_id = ZABA_1.c_node_id) INNER JOIN " + _
        "ZZZ_KIN_BIOG_ADDR AS ZABA_2 ON " + _
        "ZABA_1.c_personid = ZABA_2.c_node_id) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_2.c_personid "
Else
    tQueryFromStr = "FROM (ZZZ_NONKIN_BIOG_ADDR AS ZABA INNER JOIN ZZZ_NONKIN_BIOG_ADDR " + _
        "AS ZABA_1 ON ZABA.c_node_id = " + _
        "ZABA_1.c_node_id) INNER JOIN ZZZ_KIN_BIOG_ADDR AS ZABA_2 " + _
        "ON ZABA_1.c_personid = ZABA_2.c_node_id "
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery1stStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'
' get second person
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQuery2ndStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
'
' copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = tQueryAppendStr
cmdSQL.Execute tRecDeleted
'

End If

End If
'
' remove duplicates
'

```



```

tQueryStr = "UPDATE ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN ZZ_SCRATCH_PEOPLE ON " + _
    "ZZ_SCRATCH_PEOPLE_1.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
    "SET ZZ_SCRATCH_PEOPLE.c_delete = True " + _
    "WHERE (((ZZ_SCRATCH_PEOPLE.c_node_dist)>[ZZ_SCRATCH_PEOPLE_1].[c_node_dist]))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PEOPLE WHERE c_delete = True"
cmdSQL.Execute tRecDeleted
,
' now use the people table to get the edges:
,
cmdSQL.CommandText = "DELETE * FROM ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
,
' first non-kin
,
tQueryIntoStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year, " + _
    "c_female, c_node_id, c_node_name, c_node_chn, c_node_index_year, c_node_female, c_link_type, " + _
    "c_link_code, c_link_desc, c_link_chn, c_link_count, c_addr_id, c_addr_name, c_addr_chn, " + _
    "c_addr_type, c_addr_desc, c_addr_desc_chn, x_coord, y_coord, c_node_addr_id, c_node_addr_name, " + _
    "c_node_addr_chn, c_node_addr_type, c_node_addr_desc, c_node_addr_desc_chn, node_xcoord, " + _
    "c_edge_dist, type_id, c_kin_desc, c_kin_desc_chn, c_kin_name, c_kin_chn, c_kin_id, c_kin_code, " + _
    "c_assoc_kin_desc, c_assoc_kin_desc_chn, c_assoc_kin_name, c_assoc_kin_chn, c_assoc_kin_id, " + _
    "c_assoc_kin_code, c_litgenre_code, c_litgenre_desc, c_litgenre_desc_chn, c_topic_code, " + _
    "c_topic_desc, c_topic_desc_chn, c_occasion_code, c_occasion_desc, c_occasion_desc_chn, " + _
    "c_inst_code, c_inst_name_hz, c_text_title, c_assoc_claimer_id, c_assoc_claimer_name, " + _
    "c_assoc_claimer_name_chn, c_distance )" + _
tQuerySelectStr = "SELECT ASSOC_REL.c_personid, ASSOC_REL.c_person_name, ASSOC_REL.c_person_name_chn, " + _
    "ASSOC_REL.c_index_year, ASSOC_REL.c_female, ASSOC_REL.c_node_id, ASSOC_REL.c_node_name, " + _
    "ASSOC_REL.c_node_chn, ASSOC_REL.c_node_index_year, ASSOC_REL.c_node_female, 'N' AS c_link_type, " + _
    "ASSOC_REL.c_link_code, ASSOC_REL.c_link_desc, ASSOC_REL.c_link_chn, ASSOC_REL.c_link_count, " + _
    "ASSOC_REL.c_addr_id, ASSOC_REL.c_addr_name, ASSOC_REL.c_addr_chn, ASSOC_REL.c_addr_type, " + _
    "ASSOC_REL.c_addr_desc, ASSOC_REL.c_addr_desc_chn, ASSOC_REL.x_coord, ASSOC_REL.y_coord, " + _
    "ASSOC_REL.c_node_addr_id, ASSOC_REL.c_node_addr_name, ASSOC_REL.c_node_addr_chn, " + _
    "ASSOC_REL.c_node_addr_type, ASSOC_REL.c_node_addr_desc, ASSOC_REL.c_node_addr_desc_chn, " + _
    "ASSOC_REL.node_xcoord, ZZ_SCRATCH_PEOPLE.c_node_dist, 'N' AS type_id, ASSOC_REL.c_kinrel, " + _
    "ASSOC_REL.c_kinrel_chn, ASSOC_REL.c_kin_name, ASSOC_REL.c_kin_chn, ASSOC_REL.c_kin_id, " + _
    "ASSOC_REL.c_kin_code, ASSOC_REL.c_assoc_kinrel, ASSOC_REL.c_assoc_kinrel_chn, " + _
    "ASSOC_REL.c_assoc_kin_name, ASSOC_REL.c_assoc_kin_chn, ASSOC_REL.c_assoc_kin_id, " + _
    "ASSOC_REL.c_assoc_kin_code, ASSOC_REL.c_litgenre_code, ASSOC_REL.c_litgenre_desc, " + _
    "ASSOC_REL.c_litgenre_desc_chn, ASSOC_REL.c_topic_code, ASSOC_REL.c_topic_desc, " + _
    "ASSOC_REL.c_topic_desc_chn, ASSOC_REL.c_occasion_code, ASSOC_REL.c_occasion_desc, " + _
    "ASSOC_REL.c_occasion_desc_chn, ASSOC_REL.c_inst_code, ASSOC_REL.c_inst_name_hz, " + _
    "ASSOC_REL.c_text_title, ASSOC_REL.c_assoc_claimer_id, " + _
    "ASSOC_REL.c_assoc_claimer_name, ASSOC_REL.c_assoc_claimer_chn, ASSOC_REL.c_distance "
tQueryFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN (ZZ_SCRATCH_PEOPLE INNER " + _
    "JOIN ZZZ_NONKIN_BIOG_ADDR AS ASSOC_REL ON ZZ_SCRATCH_PEOPLE.c_person_id = " + _
    "ASSOC_REL.c_personid) ON ZZ_SCRATCH_PEOPLE_1.c_person_id = ASSOC_REL.c_node_id"
,
'If CmdClearList.Enabled Then
,
'    allow the people from the list to serve as nodes, then delete them
,
'cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr
'cmdSQL.Execute tRecDeleted
,
' now mark the nodes
,
'cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZ_SOCIAL_NETWORK " + _
    "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SOCIAL_NETWORK.c_node_id " + _
    "SET ZZ_SOCIAL_NETWORK.c_delete = 1"
'cmdSQL.Execute tRecDeleted
,
' and delete
,
'cmdSQL.CommandText = "Delete from ZZ_SOCIAL_NETWORK where c_delete = 1"
'cmdSQL.Execute tRecCount
,
'Else
'    tQueryWhereStr = "WHERE (((ASSOC_REL.c_node_id)<> " + tID1Str + _
        " And (ASSOC_REL.c_node_id)<>" + tID2Str + "))"
,
'cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr + tQueryWhereStr
'cmdSQL.Execute tRecDeleted
,
'End If
cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr
cmdSQL.Execute tRecDeleted
,
If ChkKinship.Value Then

```

```

'
' get from the kinship table
'
tQueryIntoStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year, " + _
    "c_female, c_node_id, c_node_name, c_node_chn, c_node_index_year, c_node_female, c_link_type, " + _
    "c_link_code, c_link_desc, c_link_chn, c_link_count, c_addr_id, c_addr_name, c_addr_chn, " + _
    "c_addr_type, c_addr_desc, c_addr_desc_chn, x_coord, y_coord, c_node_addr_id, c_node_addr_name, " + _
    "c_node_addr_chn, c_node_addr_type, c_node_addr_desc, c_node_addr_desc_chn, node_xcoord, " + _
    "node_ycoord, c_edge_dist, type_id, c_distance ) "
tQuerySelectStr = "SELECT ZZ_SCRATCH_PEOPLE.c_person_id, KIN_REL.c_person_name, " + _
    "KIN_REL.c_person_name_chn, KIN_REL.c_index_year, KIN_REL.c_female, KIN_REL.c_node_id, " + _
    "KIN_REL.c_node_name, KIN_REL.c_node_chn, KIN_REL.c_node_index_year, KIN_REL.c_node_female, " + _
    "'K' AS c_link_type, KIN_REL.c_link_code, KIN_REL.c_link_desc, KIN_REL.c_link_chn, " + _
    "1 AS c_link_count, KIN_REL.c_addr_id, KIN_REL.c_addr_name, KIN_REL.c_addr_chn, " + _
    "KIN_REL.c_addr_type, KIN_REL.c_addr_desc, KIN_REL.c_addr_desc_chn, KIN_REL.x_coord, " + _
    "KIN_REL.y_coord, KIN_REL.c_node_addr_id, KIN_REL.c_node_addr_name, KIN_REL.c_node_addr_chn, " + _
    "KIN_REL.c_node_addr_type, KIN_REL.c_node_addr_desc, KIN_REL.c_node_addr_desc_chn, " + _
    "KIN_REL.node_xcoord, KIN_REL.node_ycoord, ZZ_SCRATCH_PEOPLE.c_node_dist, 'K' AS type_id, " + _
    "KIN_REL.c_distance "
tQueryFromStr = "FROM ZZ_SCRATCH_PEOPLE AS ZZ_SCRATCH_PEOPLE_1 INNER JOIN (ZZ_SCRATCH_PEOPLE " + _
    "INNER JOIN ZZZ_KIN_BIOG_ADDR AS KIN_REL ON ZZ_SCRATCH_PEOPLE.c_person_id = " + _
    "KIN_REL.c_personid) ON ZZ_SCRATCH_PEOPLE_1.c_person_id = KIN_REL.c_node_id "

'If CmdClearList.Enabled Then
'
'    allow the people from the list to serve as nodes, then delete them
'
'cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr
'cmdSQL.Execute tRecDeleted
'
'    now mark the nodes
'
'cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZ_SOCIAL_NETWORK " + _
    "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SOCIAL_NETWORK.c_node_id " + _
    "SET ZZ_SOCIAL_NETWORK.c_delete = 1"
'cmdSQL.Execute tRecDeleted
'
'    and delete
'
'cmdSQL.CommandText = "Delete from ZZ_SOCIAL_NETWORK where c_delete = 1"
'cmdSQL.Execute tRecCount
'Else
'
'tQueryWhereStr = "WHERE (((KIN_REL.c_node_id)<>" + tID1Str + _
    " And (KIN_REL.c_node_id)<>" + tID2Str + "))"

'cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr + tQueryWhereStr
'cmdSQL.Execute tRecDeleted
'End If
cmdSQL.CommandText = tQueryIntoStr + tQuerySelectStr + tQueryFromStr
cmdSQL.Execute tRecDeleted
End If
'
' now mark all the duplicates that are inverses of other records for deletion
'
tQueryStr = "UPDATE ASSOC_CODES INNER JOIN (ZZ_SOCIAL_NETWORK AS ZZ_SOCIAL_NETWORK_1 INNER JOIN " + _
    "ZZ_SOCIAL_NETWORK ON (ZZ_SOCIAL_NETWORK_1.c_node_id = ZZ_SOCIAL_NETWORK.c_person_id) AND " + _
    "(ZZ_SOCIAL_NETWORK_1.c_person_id = ZZ_SOCIAL_NETWORK.c_node_id)) ON (ASSOC_CODES.c_assoc_pair = " + _
    "ZZ_SOCIAL_NETWORK_1.c_link_code) AND (ASSOC_CODES.c_assoc_code = ZZ_SOCIAL_NETWORK.c_link_code) " + _
    "SET ZZ_SOCIAL_NETWORK.c_delete = 1 " + _
    "WHERE (((ZZ_SOCIAL_NETWORK.c_link_type)='N') AND " + _
    "((ZZ_SOCIAL_NETWORK.c_edge_dist)>[ZZ_SOCIAL_NETWORK_1].[c_edge_dist])) OR " + _
    "(((ZZ_SOCIAL_NETWORK.c_link_type)='N') AND " + _
    "((ZZ_SOCIAL_NETWORK.c_edge_dist)=[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) AND " + _
    "((ZZ_SOCIAL_NETWORK.c_person_id)>[ZZ_SOCIAL_NETWORK_1].[c_person_id]))"

'MsgBox "About to mark inverses..."
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
If Me.ChkKinship.Value Then
    tQueryStr = "UPDATE KINSHIP_CODES INNER JOIN (ZZ_SOCIAL_NETWORK AS ZZ_SOCIAL_NETWORK_1 INNER JOIN " + _
        "ZZ_SOCIAL_NETWORK ON (ZZ_SOCIAL_NETWORK_1.c_node_id = ZZ_SOCIAL_NETWORK.c_person_id) AND " + _
        "(ZZ_SOCIAL_NETWORK_1.c_person_id = ZZ_SOCIAL_NETWORK.c_node_id)) ON KINSHIP_CODES.c_kincode = " + _
        "ZZ_SOCIAL_NETWORK.c_link_code SET ZZ_SOCIAL_NETWORK.c_delete = 1 " + _
        "WHERE (((ZZ_SOCIAL_NETWORK.c_link_type)='K') AND " + _
        "((ZZ_SOCIAL_NETWORK.c_edge_dist)>[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) AND " + _
        "((ZZ_SOCIAL_NETWORK_1.c_link_code)=[KINSHIP_CODES].[c_kin_pair1])) OR (((ZZ_SOCIAL_NETWORK.c_link_ty" + _
        "pe)='K') " + _
        "AND ((ZZ_SOCIAL_NETWORK.c_edge_dist)=[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) AND " + _
        "((ZZ_SOCIAL_NETWORK.c_person_id)>[ZZ_SOCIAL_NETWORK_1].[c_person_id]) AND " + _

```

```

        "((ZZ_SOCIAL_NETWORK_1.c_link_code)=[KINSHIP_CODES].[c_kin_pair1])) OR (((ZZ_SOCIAL_NETWORK.c_link_ty
pe)='K') " + _
        "AND ((ZZ_SOCIAL_NETWORK.c_edge_dist)>[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) " + _
        "AND ((ZZ_SOCIAL_NETWORK_1.c_link_code)=[KINSHIP_CODES].[c_kin_pair2])) OR (((ZZ_SOCIAL_NETWORK.c_lin
k_type)='K') " + _
        "AND ((ZZ_SOCIAL_NETWORK.c_edge_dist)=[ZZ_SOCIAL_NETWORK_1].[c_edge_dist]) AND " + _
        "((ZZ_SOCIAL_NETWORK.c_person_id)>[ZZ_SOCIAL_NETWORK_1].[c_person_id]) AND " + _
        "((ZZ_SOCIAL_NETWORK_1.c_link_code)=[KINSHIP_CODES].[c_kin_pair2]))"
    ,
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecCount
End If
'
' now delete
'
'MsgBox "About to delete inverses..."
cmdSQL.CommandText = "Delete from ZZ_SOCIAL_NETWORK where c_delete = 1"
cmdSQL.Execute tRecCount
'
'
' the final step is to calculate the xy_count
'
' get the XY count
'
'MsgBox "About to count XYs..."
'
cmdSQL.CommandText = "Delete * from tmpXY"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
    "SELECT ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord, Count(ZZ_SCRATCH_PEOPLE.x_coord) " + _
    "AS CountOfx_coord, Count(ZZ_SCRATCH_PEOPLE.y_coord) AS CountOfy_coord " + _
    "FROM ZZ_SCRATCH_PEOPLE " + _
    "GROUP BY ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord;"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_PEOPLE ON (tmpXY.y_coord = " + _
    "ZZ_SCRATCH_PEOPLE.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_PEOPLE.x_coord) SET " + _
    "ZZ_SCRATCH_PEOPLE.xy_count = [tmpXY].[CountOfx_coord];"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

Set gRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
gRstPeople.MoveLast

If gRstPeople.RecordCount > 0 Then
    CmdGIS.Enabled = True
    CmdPajek.Enabled = True
    CmdUCINet.Enabled = True
    ChkIncludeID.Enabled = True
Else
    CmdGIS.Enabled = False
    CmdPajek.Enabled = False
    CmdUCINet.Enabled = False
    ChkIncludeID.Enabled = False
End If

Exit_CmdQuery_Click:
' restore the form tables
Set ZZ_SOCIAL_NETWORK.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
Set ZZ_SCRATCH_PEOPLE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
'
Set cmdSQL = Nothing
Exit Sub

Err_CmdQuery_Click:
MsgBox Err.Description
Resume Exit_CmdQuery_Click

End Sub

Private Sub Link1stOrder(tTable1Str As String, tTable2Str As String)
Dim tQueryStr As String, tQueryWhereStr As String, tQueryFromStr As String
Dim tFromStr As String, tToStr As String, tID1Str As String, tID2Str As String
Dim cmdSQL As ADODB.Command, tRecDeleted As Long

Set cmdSQL = New ADODB.Command

```

```

cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' get the index year constraints: these should be on the NODE rather than the first person
'
If gUseIndexYears Then
    '
    ' four possibilities
    '
    If gFromStr = "" And gToStr = "" Then
        tQueryWhereStr = "WHERE "
    ElseIf gFromStr = "" Then
        tQueryWhereStr = "WHERE ((ZABA.c_node_index_year)<=" + gToStr + ") AND "
    ElseIf gToStr = "" Then
        tQueryWhereStr = "WHERE ((ZABA.c_node_index_year)>=" + gFromStr + ") AND "
    Else
        tQueryWhereStr = "WHERE ((ZABA.c_node_index_year)<=" + gToStr + ") AND ((ZABA.c_node_index_year)>=" +
gFromStr + ") AND "
    End If
ElseIf gUseDynasties Then
    '
    ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
    ' The constraint looks at
    '
    If gFromDynasty = -2 Then
        tQueryWhereStr = "Where ((ZZZ_NONKIN_BIOG_ADDR.c_dy) > 0 ) AND "
    ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") AND "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_dy) = " + Str(gFromDynasty) + ") AND "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
        "((DYNASTIES.c_start)<=" + Str(gToDynastyEnd) + ") AND "
    Else
        tQueryWhereStr = ""
    End If
Else
    tQueryWhereStr = "WHERE "
End If

' first add the target people (if CmdClearList is enabled, ImportList was successful)
'
If Not CmdClearList.Enabled Then
    '
    ' get the ID strings for the two people
    '
    Me.TxtID1.Visible = True
    Me.TxtID1.SetFocus
    tID1Str = Trim(Str(TxtID1.Value))
    Me.TxtID2.Visible = True
    Me.TxtID2.SetFocus
    tID2Str = Trim(Str(TxtID2.Value))
    Me.CmdQuery.SetFocus
    Me.TxtID1.Visible = False
    Me.TxtID2.Visible = False
End If

' ZABA points to either ZZZ_KIN_BIOG_ADDR or ZZZ_NONKIN_BIOG_ADDR depending on the search type (kin or non-kin)

tQueryStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
"c_female, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, " + _
"x_coord, y_coord, c_node_dist ) " + _
"SELECT DISTINCT ZABA_1.c_node_id, ZABA_1.c_node_name, " + _
"ZABA_1.c_node_chn, ZABA_1.c_node_index_year, " + _
"ZABA_1.c_node_female, ZABA_1.c_node_addr_id, " + _
"ZABA_1.c_node_addr_name, ZABA_1.c_node_addr_chn, " + _
"ZABA_1.c_node_addr_type, ZABA_1.c_node_addr_desc, " + _
"ZABA_1.c_node_addr_desc_chn, ZABA_1.c_node_xcoord, " + _
"ZABA_1.c_node_ycoord, 1 AS c_node_dist "

If CmdClearList.Enabled Then
    tQueryWhereStr = tQueryWhereStr + "((ZABA_1.c_personid)>[ZABA].[c_personid]) AND (ZABA_1.c_node_id <> 999) "
Else
    tQueryWhereStr = tQueryWhereStr + "((ZABA_1.c_personid)=" + tID2Str + " AND (ZABA.c_personid)=" + tID1Str
+ ") AND " + _

```

```

        "(ZABA_1.c_node_id <> 9999)"
    End If

    ' one changes: get the tables from the passed strings (either "KIN" or "NONKIN")

    If CmdClearList.Enabled Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 "
+ -
            "INNER JOIN (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN (ZZZ_" + tTable1Str + "_BIOG_ADDR AS ZABA " + _
            "INNER JOIN ZZZ_" + tTable2Str + "_BIOG_ADDR AS ZABA_1 " + _
            "ON ZABA.c_node_id = ZABA_1.c_node_id) " + _
            "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
            "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_1.c_personid) " + _
            "ON DYNASTIES.c_dy = ZABA.c_node_dy "
            'tQueryFromStr = "FROM DYNASTIES INNER JOIN (DYNASTIES AS DYNASTIES_1 INNER JOIN (ZZ_SCRATCH_IMPORT_P
PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 " + _
            "INNER JOIN (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN (ZZZ_" + tTable1Str + "_BIOG_ADDR AS ZABA " + _
            "INNER JOIN ZZZ_" + tTable2Str + "_BIOG_ADDR AS ZABA_1 " + _
            "ON ZABA.c_node_id = ZABA_1.c_node_id) " + _
            "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
            "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_1.c_personid) " + _
            "ON DYNASTIES_1.c_dy = ZABA_1.c_dy) ON DYNASTIES.c_dy = ZABA.c_dy " -
        Else
            tQueryFromStr = "FROM ZZ_SCRATCH_IMPORT_PEOPLE AS ZZ_SCRATCH_IMPORT_PEOPLE_1 " + _
            "INNER JOIN (ZZ_SCRATCH_IMPORT_PEOPLE " + _
            "INNER JOIN (ZZZ_" + tTable1Str + "_BIOG_ADDR AS ZABA " + _
            "INNER JOIN ZZZ_" + tTable2Str + "_BIOG_ADDR AS ZABA_1 " + _
            "ON ZABA.c_node_id = ZABA_1.c_node_id) " + _
            "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
            "ON ZZ_SCRATCH_IMPORT_PEOPLE_1.c_person_id = ZABA_1.c_personid " -
        End If
    Else
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_" + tTable1Str + "_BIOG_ADDR AS ZABA INNER JOIN ZZZ_"
+ tTable2Str + "_BIOG_ADDR AS ZABA_1 " + _
            "ON ZABA.c_node_id = ZABA_1.c_node_id) " + _
            "ON DYNASTIES.c_dy = ZABA.c_node_dy "
        Else
            tQueryFromStr = "FROM ZZZ_" + tTable1Str + "_BIOG_ADDR AS ZABA " + _
            "INNER JOIN ZZZ_" + tTable2Str + "_BIOG_ADDR AS ZABA_1 " + _
            "ON ZABA.c_node_id = ZABA_1.c_node_id "
        End If
    End If

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
    cmdSQL.Execute tRecDeleted
    '
    ' define the query for appending to ZZ_SCRATCH_PEOPLE without duplication
    '
    cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_PAIR_PEOPLE.* " + _
    "FROM ZZ_SCRATCH_PAIR_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
    "ZZ_SCRATCH_PAIR_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
    "WHERE (((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null))"
    '
    cmdSQL.Execute tRecDeleted

End Sub

Private Sub Link2ndOrder(tTable1Str As String, tTable2Str As String, tTable3Str As String)
    Dim tQueryStr As String, tQueryWhereStr As String, tQueryNodeWhereStr As String, tQueryFromStr As String, tQueryNodeFromStr As String
    Dim tFromStr As String, tToStr As String, tID1Str As String, tID2Str As String
    Dim cmdSQL As ADODB.Command, tRecDeleted As Long

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    'MsgBox "In Link2ndOrder"
    '
    ' get the index year constraints
    '
    If gUseIndexYears Then
        '
        ' four possibilities
        '
        If gFromStr = "" And gToStr = "" Then

```

```

        tQueryWhereStr = "WHERE "
        tQueryNodeWhereStr = "WHERE "
    ElseIf gFromStr = "" Then
        tQueryWhereStr = "WHERE ((ZABA.c_node_index_year)<=" + gToStr + ") " + _
            "And (ZABA_1.c_node_index_year)<=" + gToStr + ") AND "
        tQueryNodeWhereStr = "WHERE ((ASSOC_2.c_node_index_year)<=" + gToStr + ") AND " + _
            "((ASSOC_2.c_index_year)<=" + gToStr + ") AND "
    ElseIf gToStr = "" Then
        tQueryWhereStr = "WHERE ((ZABA.c_node_index_year)>=" + gFromStr + " " + _
            "AND (ZABA_1.c_node_index_year)>=" + gFromStr + ") AND "
        tQueryNodeWhereStr = "WHERE ((ASSOC_2.c_node_index_year)>=" + gFromStr + ") And " + _
            "((ASSOC_2.c_index_year)>=" + gFromStr + ") AND "
    Else
        tQueryWhereStr = "WHERE ((ZABA.c_node_index_year)>=" + gFromStr + " " + _
            "And (ZABA.c_node_index_year)<=" + gToStr + ") " + _
            "AND ((ZABA_1.c_node_index_year)>=" + gFromStr + " " + _
            "And (ZABA_1.c_node_index_year)<=" + gToStr + ") AND "
        tQueryNodeWhereStr = "WHERE ((ASSOC_2.c_node_index_year)>=" + gFromStr + " And " + _
            "(ASSOC_2.c_node_index_year)<=" + gToStr + ") AND " + _
            "((ASSOC_2.c_index_year)>=" + gFromStr + " And " + _
            "(ASSOC_2.c_index_year)<=" + gToStr + ") AND "
    End If

ElseIf gUseDynasties Then
    '
    ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
    '
    ' for tQueryNodeWhere, DYNASTIES is linked to ASSOC_2.c_dy while DYNASTIES_1 is linked to ASSOC_2.c_node
_dy
    '
    If gFromDynasty = -2 Then
        tQueryWhereStr = "Where ((ZABA.c_node_dy) > 0 AND (ZABA_1.c_node_dy) > 0) AND "
        tQueryNodeWhereStr = "WHERE ((ASSOC_2.c_dy) > 0 AND (ASSOC_2.c_node_dy) > 0) AND "
    ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + " AND (DYNASTIES_1.c_start)<" +
Str(gToDynastyEnd) + ") AND "
        tQueryNodeWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + " AND (DYNASTIES_1.c_start)
<" + Str(gToDynastyEnd) + ") AND "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + " AND (DYNASTIES_1.c_end)>" +
Str(gFromDynastyBegin) + ") AND "
        tQueryNodeWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + " AND (DYNASTIES_1.c_end)
>" + Str(gFromDynastyBegin) + ") AND "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_dy) = " + Str(gFromDynasty) + " AND (DYNASTIES_1.c_dy) = " + St
r(gFromDynasty) + ") AND "
        tQueryNodeWhereStr = "WHERE ((DYNASTIES.c_dy) = " + Str(gFromDynasty) + " AND (DYNASTIES_1.c_dy) = "
+ Str(gFromDynasty) + ") AND "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + " AND (DYNASTIES_1.c_end)>" +
Str(gFromDynastyBegin) + ") AND " + _
            "((DYNASTIES.c_start)<=" + Str(gToDynastyEnd) + " AND (DYNASTIES_1.c_start)<=" + Str(gToDynastyEn
d) + ") AND "
        tQueryNodeWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + " AND (DYNASTIES_1.c_end)
>" + Str(gFromDynastyBegin) + " AND " + _
            "(DYNASTIES.c_start)<=" + Str(gToDynastyEnd) + " AND (DYNASTIES_1.c_start)<=" + Str(gToDynastyEnd
) + ") AND "
    Else
        tQueryWhereStr = "WHERE "
        tQueryNodeWhereStr = "WHERE "
    End If
Else
    tQueryWhereStr = "WHERE "
    tQueryNodeWhereStr = "WHERE "
End If

If Not CmdClearList.Enabled Then
    '
    ' get the ID strings for the two people
    '
    Me.TxtID1.Visible = True
    Me.TxtID1.SetFocus
    tID1Str = Trim(Str(TxtID1.Value))
    Me.TxtID2.Visible = True
    Me.TxtID2.SetFocus
    tID2Str = Trim(Str(TxtID2.Value))
    Me.CmdQuery.SetFocus
    Me.TxtID1.Visible = False
    Me.TxtID2.Visible = False
End If

```

```

'MsgBox "Set parameters"

' the new logic is to get a list of all the connections of connections and then see which of
' those are linked to others in the IMPORT table

' The FROM statements change if DYNASTIES has to be linked in, so we
,
If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
    tQueryFromStr = "FROM DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES INNER JOIN ((ZZ_SCRATCH_IMPORT_PEOP
E " + _
        "INNER JOIN ZZZ_" + tTable1Str + "_BIOG_ADDR AS ZABA ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c
_personid) " + _
        "INNER JOIN ZZZ_" + tTable2Str + "_BIOG_ADDR AS ZABA_1 " + _
        "ON ZABA.c_node_id = ZABA_1.c_personid) " + _
        "ON DYNASTIES.c_dy = ZABA.c_dy) " + _
        "ON DYNASTIES_1.c_dy = ZABA_1.c_dy "

    tQueryNodeFromStr = "FROM DYNASTIES AS DYNASTIES_1 INNER JOIN (DYNASTIES " + _
        "INNER JOIN ((ZZ_" + tTable1Str + "_BIOG_ADDR AS ASSOC_1 INNER JOIN ZZZ_" + tTable2Str + "_BIOG_ADDR
AS ASSOC_2 " + _
        "ON ASSOC_1.c_node_id = ASSOC_2.c_personid) " + _
        "INNER JOIN ZZZ_" + tTable3Str + "_BIOG_ADDR AS ASSOC_3 " + _
        "ON ASSOC_2.c_node_id = ASSOC_3.c_node_id) " + _
        "ON DYNASTIES.c_dy = ASSOC_2.c_dy) " + _
        "ON DYNASTIES_1.c_dy = ASSOC_2.c_node_dy "

Else
    tQueryFromStr = "FROM (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZZ_" + tTable1Str + "_BIOG_ADDR AS ZABA " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "INNER JOIN ZZZ_" + tTable2Str + "_BIOG_ADDR AS ZABA_1 ON " + _
        "ZABA.c_node_id = ZABA_1.c_personid "

    tQueryNodeFromStr = "FROM (ZZ_" + tTable1Str + "_BIOG_ADDR AS ASSOC_1 INNER JOIN " + _
        "ZZ_" + tTable2Str + "_BIOG_ADDR AS ASSOC_2 ON ASSOC_1.c_node_id = ASSOC_2.c_personid) " + _
        "INNER JOIN ZZZ_" + tTable3Str + "_BIOG_ADDR AS ASSOC_3 " + _
        "ON ASSOC_2.c_node_id = ASSOC_3.c_node_id "

End If

If CmdClearList.Enabled Then
    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_NETWORK"
    cmdSQL.Execute tRecDeleted

    tQueryStr = "INSERT INTO zz_scratch_pair_network ( c_pid_1, c_pid_2, c_link_1, c_pid_3, c_link_2 ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id AS c_pid_1, " + _
        "ZABA.c_node_id AS c_pid_2, ZABA.c_link_code AS c_link_1, " + _
        "ZABA_1.c_node_id AS c_pid_3, ZABA_1.c_link_code AS c_link_2 "

    tQueryFromStr = "FROM (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZZ_" + tTable1Str + "_BIOG_ADDR AS ZABA " + _
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZABA.c_personid) " + _
        "INNER JOIN ZZZ_" + tTable2Str + "_BIOG_ADDR AS ZABA_1 ON " + _
        "ZABA.c_node_id = ZABA_1.c_personid "

    tQueryWhereStr = tQueryWhereStr + "(((ZABA.c_node_id)<>9999) AND " + _
        "((ZABA_1.c_node_id)<>9999 And " + _
        "(ZABA_1.c_node_id)<>[ZZ_SCRATCH_IMPORT_PEOPLE].[c_person_id])) "

'MsgBox "tQueryFromStr = " + tQueryFromStr
'MsgBox "tQueryWhereStr = " + tQueryWhereStr

cmdSQL.CommandText = tQueryStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted

' Delete all those where c_pid_2 appears in ZZ_SCRATCH_IMPORT_PEOPLE

cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN zz_scratch_pair_network ON " + _
    "ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = zz_scratch_pair_network.c_pid_2 " + _
    "SET zz_scratch_pair_network.c_delete = 1"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_NETWORK WHERE c_delete = 1"
cmdSQL.Execute tRecDeleted

' then see how many of those work

tQueryStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name_chn, c_name, c_index_year, c_dy, c
dynasty, c_dynasty_chn, " + _
    "c_female, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_addr_type, c_addr_desc, " + _
    "c_addr_desc_chn, c_node_dist ) " + _
    "SELECT DISTINCT ZABA.c_personid, ZABA.c_person_name_chn, " + _
    "ZABA.c_person_name, ZABA.c_index_year, " + _

```

```

        "ZABA.c_dy, ZABA.c_dynasty, ZABA.c_dynasty_chn, " + _
        "ZABA.c_female, ZABA.c_addr_id, " + _
        "ZABA.c_addr_name, ZABA.c_addr_chn, " + _
        "ZABA.x_coord, ZABA.y_coord, " + _
        "ZABA.c_addr_type, ZABA.c_addr_desc, " + _
        "ZABA.c_addr_desc_chn, 2 AS c_node_dist " + _
        "FROM (zz_scratch_pair_network INNER JOIN ZZ_Z " + tTable3Str + " _BIOG_ADDR AS ZABA " + _
        "ON zz_scratch_pair_network.c_pid_3 = ZABA.c_personid) " + _
        "INNER JOIN ZZ_SCRATCH_IMPORT_PEOPLE ON " + _
        "ZABA.c_node_id = ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id " + _
        "WHERE (((ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id)<>[zz_scratch_pair_network].[c_pid_1]))"

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' define the query for appending to ZZ_SCRATCH_PEOPLE without duplication
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_PAIR_PEOPLE.* " + _
        "FROM ZZ_SCRATCH_PAIR_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
        "ZZ_SCRATCH_PAIR_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
        "WHERE (((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null))"
'
cmdSQL.Execute tRecDeleted
'
' now get the second person
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name_chn, c_name, c_index_year, c_dy, c_
dynasty, c_dynasty_chn, " + _
        "c_female, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_addr_type, c_addr_desc, " + _
        "c_addr_desc_chn, c_node_dist ) " + _
        "SELECT DISTINCT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name_chn, " + _
        "ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_index_year, ZZZ_BIOG_MAIN.c_dy, ZZZ_BIOG_MAIN.c_dynasty, ZZZ_B
IOG_MAIN.c_dynasty_chn, " + _
        "ZZZ_BIOG_MAIN.c_female, ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_MAI
N.c_index_addr_chn, " + _
        "ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord, ZZZ_BIOG_MAIN.c_index_addr_type_code, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_desc, ZZZ_BIOG_MAIN.c_index_addr_type_chn, 2 AS c_node_dist " + _
        "FROM ZZZ_BIOG_MAIN INNER JOIN ((zz_scratch_pair_network INNER JOIN " + _
        "ZZZ " + tTable3Str + " _BIOG_ADDR AS ZABA " + _
        "ON zz_scratch_pair_network.c_pid_3 = ZABA.c_personid) " + _
        "INNER JOIN ZZ_SCRATCH_IMPORT_PEOPLE ON " + _
        "ZABA.c_node_id = ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id) ON " + _
        "ZZZ_BIOG_MAIN.c_personid = zz_scratch_pair_network.c_pid_2 " + _
        "WHERE (((ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id)<>[zz_scratch_pair_network].[c_pid_1]))"

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' define the query for appending to ZZ_SCRATCH_PEOPLE without duplication
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_PAIR_PEOPLE.* " + _
        "FROM ZZ_SCRATCH_PAIR_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
        "ZZ_SCRATCH_PAIR_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
        "WHERE (((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null))"
'
cmdSQL.Execute tRecDeleted
Else
tQuery1stStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
        "c_dy, c_dynasty, c_dynasty_chn, " + _
        "c_female, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_addr_type, " + _
        "c_addr_desc, c_addr_desc_chn, c_node_dist ) " + _
        "SELECT DISTINCT ASSOC_2.c_personid, ASSOC_2.c_person_name, ASSOC_2.c_person_name_chn, ASSOC_2.c_inde
x_year, " + _
        "ASSOC_2.c_dy, ASSOC_2.c_dynasty, ASSOC_2.c_dynasty_chn, " + _
        "ASSOC_2.c_female, ASSOC_2.c_addr_id, ASSOC_2.c_addr_name, " + _
        "ASSOC_2.c_addr_chn, ASSOC_2.x_coord, ASSOC_2.y_coord, ASSOC_2.c_addr_type, " + _
        "ASSOC_2.c_addr_desc, ASSOC_2.c_addr_desc_chn, 2 AS c_node_dist "
'
tQuery2ndStr = "INSERT INTO ZZ_SCRATCH_PAIR_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, " + _
        "c_dy, c_dynasty, c_dynasty_chn, " + _
        "c_female, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_addr_type, " + _
        "c_addr_desc, c_addr_desc_chn, c_node_dist) " + _
        "SELECT DISTINCT ASSOC_2.c_node_id, ASSOC_2.c_node_name, ASSOC_2.c_node_chn, ASSOC_2.c_node_index_yea
r, " + _

```



```

"ASSOC_2.c_node_dy, ASSOC_2.c_node_dynasty, ASSOC_2.c_node_dynasty_chn, " + _
"ASSOC_2.c_node_female, ASSOC_2.c_node_addr_id, " + _
"ASSOC_2.c_node_addr_name, ASSOC_2.c_node_addr_chn, ASSOC_2.c_node_xcoord, ASSOC_2.c_node_ycoord, " + _
"ASSOC_2.c_node_addr_type, ASSOC_2.c_node_addr_desc, ASSOC_2.c_node_addr_desc_chn, " + _
"2 AS c_node_dist "

'tQueryNodeFromStr = "FROM (ZZZ " + tTable1Str + " BIOG_ADDR AS ASSOC_1 INNER JOIN " +
"ZZZ " + tTable2Str + " BIOG_ADDR AS ASSOC_2 ON ASSOC_1.c_node_id = ASSOC_2.c_personid) " + _
"INNER JOIN ZZZ " + tTable3Str + " BIOG_ADDR AS ASSOC_3 " + _
"ON ASSOC_2.c_node_id = ASSOC_3.c_node_id "

tQueryNodeWhereStr = tQueryNodeWhereStr + "((ASSOC_1.c_personid)=" + tID1Str + ") AND " + _
"((ASSOC_2.c_personid)<>" + tID2Str + " And ASSOC_2.c_personid <> 9999 AND " + _
"((ASSOC_2.c_personid)<>" + tID1Str + ") AND " + _
"((ASSOC_2.c_node_id)<>" + tID2Str + " And ASSOC_2.c_node_id <> 9999 AND " + _
"((ASSOC_2.c_node_id)<>" + tID1Str + ") AND " + _
"((ASSOC_3.c_personid)=" + tID2Str + ") )"

'MsgBox "tQuery1stStr = " + tQuery1stStr
'MsgBox "tQuery2ndStr = " + tQuery2ndStr
'MsgBox "tQueryNodeFromStr = " + tQueryNodeFromStr
'MsgBox "tQueryNodeWhereStr = " + tQueryNodeWhereStr

' get the first person

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted

'MsgBox "Getting first person"
,
cmdSQL.CommandText = tQuery1stStr + tQueryNodeFromStr + tQueryNodeWhereStr
cmdSQL.Execute tRecDeleted
,
' define the query for appending to ZZ_SCRATCH_PEOPLE without duplication

'MsgBox "Adding first person"
,
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_PAIR_PEOPLE.* " + _
"FROM ZZ_SCRATCH_PAIR_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
"ZZ_SCRATCH_PAIR_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
"WHERE ((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null)"
,
cmdSQL.Execute tRecDeleted

' get the second person

'MsgBox "Getting second person"

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PAIR_PEOPLE"
cmdSQL.Execute tRecDeleted
,
cmdSQL.CommandText = tQuery2ndStr + tQueryNodeFromStr + tQueryNodeWhereStr
cmdSQL.Execute tRecDeleted
,
' define the query for appending to ZZ_SCRATCH_PEOPLE without duplication

'MsgBox "Adding second person"
,
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE SELECT ZZ_SCRATCH_PAIR_PEOPLE.* " + _
"FROM ZZ_SCRATCH_PAIR_PEOPLE LEFT JOIN ZZ_SCRATCH_PEOPLE ON " + _
"ZZ_SCRATCH_PAIR_PEOPLE.c_person_id = ZZ_SCRATCH_PEOPLE.c_person_id " + _
"WHERE ((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null)"
,
cmdSQL.Execute tRecDeleted

End If
End Sub
Private Sub CmdHelp_Click()
Dim tStrPDF As String

tStrPDF = Application.CurrentProject.Path + "\HelpFiles\HelpFile_LookAtAssociationPairs.pdf"

'MsgBox tStrPDF

Application.FollowHyperlink tStrPDF, , True

End Sub

Private Sub writeKML()
,
' This program will dump the results to a .gis file

```

```

'
If ZZ_SCRATCH_PEOPLE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_writeKML
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If

tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

' next get a file
'

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_writeKML
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".kml"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
            tFileName = tFileName + ".txt"
        End If
    End If
    '
    ' write the file
    '
    'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
    '
    ' process the table
    '
    Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
    tC = Chr(9) ' the tab
    tDQ = Chr(34) ' the double quotation mark
    '
    ' write the header
    '
    tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
    tStream.WriteText "<Document>", adWriteLine
    tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
    tStream.WriteText tC + "<open>1</open>", adWriteLine
    tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

    tStream.WriteText tC + "<Style id=" + tDQ + "assoc-balloon-template" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
    tStream.WriteText tC + tC + tC + "<text>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "<![CDATA[" + tC + "ID: ${AssocGIS/PersonID} <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Name Chn: ${AssocGIS/NameHZ] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Index Year: ${AssocGIS/IndexYear] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Sex: ${AssocGIS/Sex] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Address: ${AssocGIS/AddrName] ${AssocGIS/AddrHZ] <br/>", adWriteLine

```

```

ne
    tStream.WriteText tC + tC + tC + tC + "XY Count: $[AssocGIS/XYCount] <br/><br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
    tStream.WriteText tC + tC + tC + "</text>", adWriteLine
    tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
    tStream.WriteText tC + "</Style>", adWriteLine
    tStream.WriteText tC + "<!-- Declare the type " + tDQ + "AssocGIS" + tDQ + " with 8 fields -->", adWriteLine
ine
    tStream.WriteText tC + "<Schema name=" + tDQ + "AssocGIS" + tDQ + " id=" + tDQ + "AssocGISId" + tDQ + ">"
, adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ
+ ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person ID]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "NameHZ" + tDQ
+ ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Name Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "Sex" + tDQ +
">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Sex]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + t
DQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrHZ" + tDQ
+ ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" +
tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "NodeDist" + tDQ
+ ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Node Distance]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
,
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs, even where there should not be any
        '
        ' write the point header
        '
        tStream.WriteText tC + "<Placemark>", adWriteLine

        If IsNull(!c_name) Then
            tStr = "[Bad Data] "
        Else
            tStr = !c_name
        End If
        tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

        tStream.WriteText tC + tC + "<styleUrl>#assoc-balloon-template</styleUrl>", adWriteLine
        '
        ' First Year as time stamp
        '
        If IsNull(!c_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_index_year)
        End If
        tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
        '
        tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#AssocGISId" + tDQ + ">", adWr
iteLine
        '
        ' person ID
        '
        tStr = Str(!c_person_id)
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +

```

```

"</SimpleData>", adWriteLine
'
'   Person Name Chn
'
If IsNull(!c_name_chn) Then
    tStr = tStr + "[Bad Data]"
Else
    If Trim(!c_name_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_name_chn
    End If
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NameHZ" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
'
'   Index Year
'
If IsNull(!c_index_year) Then
    tStr = "N/A"
Else
    tStr = Str(!c_index_year)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr + "
+ "</SimpleData>", adWriteLine
'
'   Node Distance
'
If IsNull(!c_node_dist) Then
    tStr = "0"
Else
    tStr = Str(!c_node_dist)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NodeDist" + tDQ + ">" + tStr + "
"</SimpleData>", adWriteLine
'
'   Sex
'
If !c_female Then
    tStr = "F"
Else
    tStr = "M"
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "Sex" + tDQ + ">" + tStr + "</S
impleData>", adWriteLine
'
'   Address Name
'
If IsNull(!c_addr_name) Then
    tStr = "[?]"
ElseIf Trim(!c_addr_name) = "" Then
    tStr = "[?]"
Else
    tStr = !c_addr_name
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr + "
"</SimpleData>", adWriteLine
'
'   Address Name Chinese
'
If IsNull(!c_addr_chn) Then
    tStr = "[?]"
ElseIf Trim(!c_addr_chn) = "" Then
    tStr = "[?]"
Else
    tStr = !c_addr_chn
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrHZ" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
'
'   XY Count
'
If IsNull(!xy_count) Then
    tStr = "0"
Else
    tStr = Str(!xy_count)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr + "
"</SimpleData>", adWriteLine
'

```

```

        tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
        tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + "<Point>", adWriteLine
        '
        ' coordinates
        If IsNull(!x_coord) Then
            tStr = "0"
        Else
            tStr = Str(!x_coord)
        End If

        If IsNull(!y_coord) Then
            tStr = tStr + ",0"
        Else
            tStr = tStr + "," + Str(!y_coord)
        End If
        tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
        '
        ' footer
        '
        tStream.WriteText tC + tC + "</Point>", adWriteLine
        tStream.WriteText tC + "</Placemark>", adWriteLine
        .MoveNext
    Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_writeKML:
Exit Sub

Err_writeKML:
MsgBox Err.Description
Resume Exit_writeKML

End Sub

Private Sub CmdStoreID_Click()
    Dim cmdSQL As ADODB.Command, tRecCount As Variant

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    If DCount("?", "ZZ_STORE_PERSON_ID") > 0 Then
        ' Display message.
        If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) = vbNo Then
            Exit Sub
        Else
            cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
            cmdSQL.Execute tRecCount
        End If
    End If

    tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id " +
        "FROM ZZ_SCRATCH_PEOPLE WHERE ZZ_SCRATCH_PEOPLE.c_person_id > 0"

    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecCount
    MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."

```

```

' update storage source
cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='AssocPairs' WHERE PersonIDSource.LineNum =1"
cmdSQL.Execute tRecCount

End Sub
Private Sub CmdRecallID_Click()
On Error GoTo Err_CmdRecallID_Click
Dim tStrSQL As String, tRst As DAO.Recordset

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

' first things first, we need two people for the routine to work, so test if there are at least two stored ID
s
If DCount("*", "ZZ_STORE_PERSON_ID") = 1 Then
Set tRst = CurrentDb.OpenRecordset("SELECT ZZ_STORE_PERSON_ID.c_personid FROM ZZ_STORE_PERSON_ID")
tRst.MoveFirst
tID = tRst!c_personid
Me.TxtID1 = tID

Set tRst = CurrentDb.OpenRecordset("SELECT BIOG_MAIN.c_name, BIOG_MAIN.c_name_chn FROM BIOG_MAIN WHERE ((
(BIOG_MAIN.c_personid)=" + Str(tID) + ")))")
tRst.MoveFirst

Me.TxtPerson1.Value = tRst!c_name
Me.TxtPerson1Chn.Value = tRst!c_name_chn
tRst.Close
Set tRst = Nothing
Exit Sub
End If

If DCount("*", "ZZ_SCRATCH_IMPORT_PEOPLE") > 0 Then
' Display message.
If MsgBox("Do you wish to replace the current import list?", vbYesNo + vbQuestion + vbDefaultButton2) = v
bNo Then
Exit Sub
Else
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecCount
End If
End If
'
' Clear the error table now that we are ready to go
'
cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted
'
' copy the IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id ) SELECT DISTINCT c_personid FROM ZZ_STORE_PERS
ON_ID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted = 0 Then
TxtPerson1.Value = "[Error]"
TxtPerson1Chn.Value = "[Error]"
TxtPerson2.Value = "[Error]"
TxtPerson2Chn.Value = "[Error]"

CmdQuery.Enabled = False
Else
TxtPerson1.Value = "[Recalled List]"
TxtPerson1Chn.Value = "[Recalled List]"
TxtPerson2.Value = "[Recalled List]"
TxtPerson2Chn.Value = "[Recalled List]"

CmdQuery.Enabled = True
CmdImportList.Enabled = False
CmdClearList.Enabled = True
CmdPickPerson1.Enabled = False
CmdPickPerson2.Enabled = False
End If

Set cmdSQL = Nothing
Set tFileSystem = Nothing

```

```
Exit_CmdRecallID_Click:
```

```
Exit Sub
```

```
Err_CmdRecallID_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdRecallID_Click
```

```
End Sub
```

```
Private Sub FrameFilterYears_Click()
```

```
'
```

```
' the simplest approach is to turn it all off and then turn on the appropriate objects
```

```
' disable all
```

```
Me.CmdFromDynasty.Enabled = False
```

```
Me.CmdToDynasty.Enabled = False
```

```
Me.CmdAllDynasties.Enabled = False
```

```
Me.TxtFromDynasty.Enabled = False
```

```
Me.TxtFromDynastyPY.Enabled = False
```

```
Me.TxtToDynasty.Enabled = False
```

```
Me.TxtToDynastyPY.Enabled = False
```

```
Me.TxtFromDynasty.Locked = False
```

```
Me.TxtFromDynastyPY.Locked = False
```

```
Me.TxtToDynasty.Locked = False
```

```
Me.TxtToDynastyPY.Locked = False
```

```
Me.TxtFromYear.Enabled = False
```

```
Me.TxtToYear.Enabled = False
```

```
gUseIndexYears = False
```

```
gUseDynasties = False
```

```
If FrameFilterYears.Value = 2 Then
```

```
' enable index years
```

```
Me.TxtFromYear.Enabled = True
```

```
Me.TxtToYear.Enabled = True
```

```
gUseIndexYears = True
```

```
ElseIf FrameFilterYears.Value = 3 Then
```

```
' enable dynasties
```

```
Me.CmdFromDynasty.Enabled = True
```

```
Me.CmdToDynasty.Enabled = True
```

```
Me.CmdAllDynasties.Enabled = True
```

```
Me.TxtFromDynasty.Enabled = True
```

```
Me.TxtFromDynastyPY.Enabled = True
```

```
Me.TxtToDynasty.Enabled = True
```

```
Me.TxtToDynastyPY.Enabled = True
```

```
Me.TxtFromDynasty.Locked = True
```

```
Me.TxtFromDynastyPY.Locked = True
```

```
Me.TxtToDynasty.Locked = True
```

```
Me.TxtToDynastyPY.Locked = True
```

```
gUseDynasties = True
```

```
End If
```

```
'MsgBox "gUseIndexYears = " + IIf(gUseIndexYears, "True", "False")
```

```
End Sub
```

```
Option Compare Database
```

```
Public gRstPeople As DAO.Recordset, gDisplayLanguage As String, gLabelsOK As Boolean
```

```
Public gImportPlaces As Boolean, gUseADDRID As Boolean, gFromStr As String, gToStr As String
```

```
Public gAssocCodeStr As String, gAssocTypeStr As String
```

```
Public gFromDynasty As Integer, gToDynasty As Integer, gUseIndexYears As Boolean, gUseDynasties As Boolean, _  
gFromDynastyBegin As Integer, gFromDynastyEnd As Integer, gToDynastyBegin As Integer, gToDynastyEnd As Integer
```

```
Private Sub ChkIndexYears_Click()
```

```
    If TxtFromYear.Enabled Then
```

```
        TxtFromYear.Enabled = False
```

```
        TxtToYear.Enabled = False
```

```
    Else
```

```
        TxtFromYear.Enabled = True
```

```
        TxtToYear.Enabled = True
```

```
    End If
```

```
End Sub
```

```
Private Sub CmdAllDynasties_Click()
```

```
    gFromDynasty = -2
```

```
    gToDynasty = -2
```

```
    TxtFromDynasty.Value = ""
```

```
    TxtFromDynastyPY.Value = "All"
```

```
    TxtToDynasty.Value = ""
```

```
    TxtToDynastyPY.Value = "All"
```

```
End Sub
```

```
Private Sub CmdFromDynasty_Click()
```

```
    Dim stDocName As String
```

```
    Dim stLinkCriteria As String
```

```
    Dim strFromDynasty As String
```

```
    If gFromDynasty < 0 Then
```

```
        strFromDynasty = ""
```

```
    Else
```

```
        strFromDynasty = Str(gFromDynasty)
```

```
    End If
```

```
    stDocName = "frmPickDynasty"
```

```
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty
```

```
    If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
```

```
        gFromDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
```

```
        gFromDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
```

```
        gFromDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
```

```
    '
```

```
    ' check to see if we have a problem and reject selection
```

```
    '
```

```
    If gToDynasty > -1 Then
```

```
        If gFromDynastyBegin > gToDynastyEnd Then
```

```
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
```

```
            gFromDynasty = -1
```

```
            TxtFromDynasty.Value = ""
```

```
            TxtFromDynastyPY.Value = ""
```

```
        End If
```

```
    End If
```

```
    '
```

```
    ' value is OK
```

```
    '
```

```
    If gFromDynasty > -1 Then
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
```

```
        TxtFromDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
```

```
        TxtFromDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
```

```
    End If
```

```
    DoCmd.Close acForm, stDocName
```

```
    '
```

```
    ' reset ToDynasty if necessary (-2 = all dynasties)
```

```
    '
```

```
    If gToDynasty = -2 Then
```



```

        gToDynasty = -1
        TxtToDynasty.Value = ""
        TxtToDynastyPY.Value = ""
    End If
End If

End Sub

Private Sub CmdGephi_Click()
    On Error GoTo Err_CmdGephi_Click
    '
    ' This program will dump the results of the search to a .gdf file
    '
    ' for the moment I'll just describe the format of the .gdf file
    '
    ' nodedef> name, label, labelvisible, style, pinyin VARCHAR(50), nodedist INT
    '     name = str(c_person_id)
    '     label = c_name_chn
    '     style = 4 (text inside a rectangle)
    '     pinyin = c_name
    '     nodedist = c_node_dist INT
    '     indexyear = c_index_year INT
    '     sex = c_female > (F,M)
    '
    ' edgedef> node1, node2, label, labelvisible, edge_desc VARCHAR(50)
    '     node1 = str(c_person_id) for node1
    '     node2 = str(c_node_id) for node2
    '     label = c_link_chn
    '     edge_desc = c_link_desc
    '     edgetype= c_link_type (K,N)
    '
    ' the central question is whether to do distance optimizations
    '
    ' first see if there are any records to process
    '
    If ZZ_SCRATCH_ASSOC.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_CmdGephi_Click
    End If
    '
    If ZZ_SCRATCH_P_ASSOC.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_CmdGephi_Click
    End If
    '
    ' next get a file
    '
    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tRstNode As DAO.Recordset
    Dim tRstEdge As DAO.Recordset
    Dim tStr As String, tC As String, ti As Integer, tCodeStr As String
    'Dim tFileSystem, tGDF

    ' to write to a UTF-8 file, use the ADO stream object
    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream

    tPinyin = False
    If CodeFrame.Value = 1 Then
        tStream.Charset = "utf-8"
        tCodeStr = "UTF8"
    ElseIf CodeFrame.Value = 2 Then
        tStream.Charset = "big5"
        tCodeStr = "BIG5"
    ElseIf CodeFrame.Value = 3 Then
        tStream.Charset = "gb18030"
        tCodeStr = "GB18030"
    Else
        tStream.Charset = "ascii"
        tCodeStr = "ASCII"
        tPinyin = True
    End If

    Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    'Use a With...End With block to reference the FileDialog object.

```

```

With dlgSaveAs
    .InitialFileName = "assoc_" + tCodeStr + ".gdf"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdGephi_Click
        Else
            ' make sure the file name has a gdf extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".gdf"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".gdf") Then
                tFileName = tFileName + ".gdf"
            End If
        End If
        '
        ' now process the file (second true removed to make ASCII)
        '
        'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
        'Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)
        '
        tStream.Mode = adModeReadWrite
        tStream.Type = adTypeText
        tStream.Open

        ' process the two tables
        '
        Set tRstEdge = ZZ_SCRATCH_ASSOC.Form.Recordset
        Set tRstNode = ZZ_SCRATCH_P_ASSOC.Form.Recordset
        tC = Chr(44) ' the comma
        tQuote = Chr(34) 'the Quote delimiter
        '
        ' first the nodes: define the record structure
        ' if ASCII, no pinyin field, no characters
        '
        If tCodeStr = "ASCII" Then
            tStr = "nodedef> name VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible BOOLEAN" + _
                tC + "style INT" + tC + "indexyear INT" + tC + "sex VARCHAR(1)" + _
                tC + "addr_name VARCHAR" + tC + "latitude DOUBLE" + tC + "longitude DOUBLE"
        Else
            tStr = "nodedef> name VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible BOOLEAN" + _
                tC + "style INT" + tC + "pinyin VARCHAR(50)" + tC + "indexyear INT" + tC + "sex VARCHAR(1)" + _
                tC + "addr_chn VARCHAR" + tC + "addr_name VARCHAR" + tC + "latitude DOUBLE" + tC + "longitude
DOUBLE"
        End If
        tStream.WriteText tStr, adWriteLine
        'tGDF.WriteLine (tStr)
        '
        With tRstNode
            .MoveFirst
            Do While Not .EOF
                ' name = the ID of the person
                tStr = Trim(Str(!c_person_id)) + tC

                ' label
                If tCodeStr = "ASCII" Then
                    If IsNull(!c_name) Then
                        tStr = tStr + tC
                    Else
                        tStr = tStr + !c_name + tC
                    End If
                Else
                    If IsNull(!c_name_chn) Then
                        tStr = tStr + tC
                    Else
                        tStr = tStr + !c_name_chn + tC
                    End If
                End If
                ' labelvisible = true, style = 4 (text inside a rectangle)
                tStr = tStr + "true" + tC + "4" + tC

                If Not (tCodeStr = "ASCII") Then

```

```

        ' pinyin = c_name
        tStr = tStr + !c_name + tC
    End If

    ' indexyear = c_index_year INT
    If IsNull(!c_index_year) Then
        tStr = tStr + "-2000" + tC
    Else
        tStr = tStr + Trim(Str(!c_index_year)) + tC
    End If

    ' sex = F,M
    tStr = tStr + !c_sex + tC

    ' address name(s)
    If tCodeStr = "ASCII" Then
        If IsNull(!c_addr_name) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_addr_name + tC
        End If
    Else
        If IsNull(!c_addr_chn) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_addr_chn + tC
        End If
        If IsNull(!c_addr_name) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_addr_name + tC
        End If
    End If

    ' latitude = !y_coord
    If IsNull(!y_coord) Then
        tStr = tStr + "0.0" + tC
    Else
        tStr = tStr + Str(!y_coord) + tC
    End If

    ' longitude = !x_coord
    If IsNull(!x_coord) Then
        tStr = tStr + "0.0"
    Else
        tStr = tStr + Str(!x_coord)
    End If
    tStream.WriteText tStr, adWriteLine
    'tGDF.WriteLine (tStr)
    .MoveNext
Loop
End With
'
' now the edges: define the record structure
' if ASCII, the label is the assoc_desc and there is not edge_desc
'
If tCodeStr = "ASCII" Then
    tStr = "edgedef> node1 VARCHAR" + tC + "node2 VARCHAR" + tC + "label VARCHAR(50)"
Else
    tStr = "edgedef> node1 VARCHAR" + tC + "node2 VARCHAR" + tC + "label VARCHAR" + tC + "edge_desc V
ARCHAR(50)"
End If
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)

With tRstEdge
    .MoveFirst
    Do While Not .EOF
        ' node1 = str(c_person_id) for node1
        tStr = Trim(Str(!c_person_id)) + tC

        ' node2 = str(c_assoc_id) for node2
        tStr = tStr + Trim(Str(!c_assoc_id)) + tC

        ' label
        If tCodeStr = "ASCII" Then
            If IsNull(!c_assoc_desc) Then
                tStr = tStr + tQuote + "[none]" + tQuote
            Else
                tStr = tStr + tQuote + Trim(Left(!c_assoc_desc + Space(50), 50)) + tQuote
            End If
        End If
    End With

```

```

        End If
    Else
        If IsNull(!c_assoc_desc_chn) Then
            tStr = tStr + tC
        Else
            tStr = tStr + tQuote + !c_assoc_desc_chn + tQuote + tC
        End If
    End If

    If Not (tCodeStr = "ASCII") Then
        ' edge_desc = c_link_desc
        If IsNull(!c_assoc_desc) Then
            tStr = tStr + tQuote + "[none]" + tQuote
        Else
            tStr = tStr + tQuote + Trim(Left(!c_assoc_desc + Space(50), 50)) + tQuote
        End If
    End If

    tStream.WriteText tStr, adWriteLine
    'tGDF.WriteLine (tStr)
    .MoveNext
Loop
End With
'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tStream.Close
Set tStream = Nothing
'
'tGDF.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
'Set tGDF = Nothing
'Set tFileSystem = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGephi_Click:
Exit Sub

Err_CmdGephi_Click:
MsgBox Err.Description
Resume Exit_CmdGephi_Click

End Sub

Private Sub CmdImportAssociations_Click()
On Error GoTo Err_CmdImportAssociations_Click

Dim stDocName As String, tRstAssociations As DAO.Recordset
Dim stLinkCriteria As String, tRstImportAssociations As DAO.Recordset
Dim tString As String, tOfficeID As Long, ti As Integer, tStrID As String, tQuit As Boolean
Dim tLen As Integer, cmdSQL As ADODB.Command

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tFileSystem, tList

' first see if we already have a list

tQuit = False
'
If Not tQuit Then
    ' open the list

    Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

    'Use a With...End With block to reference the FileDialog object.
    With dlgSaveAs
        .InitialFileName = ""
    End With

```

```

    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdImportAssociations_Click
        End If
    End If
End With
'
' Clear the address table now that we are ready to go
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_ASSOC_CODE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "AssocCodeListImport Specification", "TempImportList", tFileName, 0
'
' TransferType=acImportDelim
' SpecificationName = "TempImportList" (apparently it is saved in the database itself)
' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
' HasFieldNames = False (0)
'
' copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
    "FROM ASSOC_CODES RIGHT JOIN TempImportList ON ASSOC_CODES.c_assoc_code = TempImportList.ImportID " + _
    "WHERE (((ASSOC_CODES.c_assoc_code) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If
'
' copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_ASSOC_CODE ( c_assoc_code ) SELECT DISTINCT TempImportList.ImportID " + _
    "FROM ASSOC_CODES INNER JOIN TempImportList ON ASSOC_CODES.c_assoc_code = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

Me.TxtTypeDesc.Value = ""
Me.TxtTypeChn.Value = ""
If tRecDeleted > 0 Then
    Me.TxtAssocDesc.Value = "[Imported List]"
    Me.TxtAssocChn.Value = "[Imported List]"
    Me.CmdQuery.Enabled = True
    Me.CmdSaveAssociations.Enabled = True
Else
    Me.TxtAssocDesc.Value = ""
    Me.TxtAssocChn.Value = ""
    Me.CmdQuery.Enabled = False
    Me.CmdSaveAssociations.Enabled = False
End If

Set cmdSQL = Nothing
End If

```

```

Exit_CmdImportAssociations_Click:
Exit Sub

```

```

Err_CmdImportAssociations_Click:

```

```
MsgBox Err.Description
Resume Exit_CmdImportAssociations_Click
```

```
End Sub
```

```
Private Sub CmdNeo4j_Click()
On Error GoTo Err_CmdNeo4j_Click
'
' This program will dump the results of the search to four CSV files
'
' for the moment I'll just describe the format of the CSV file
' Note: Neo4j seems to treat all fields as strings, so there is no need to explicitly mark strings
'
' People.CSV
' nameID, NameHZ, NamePY, indexyear, sex
'     nameID = c_person_id
'     nameHZ = c_name_chn
'     namePY = c_name
'     indexyear = c_index_year
'     personDynasty = c_dynasty
'     sex = c_female > (F,M)
'
' Places.CSV
'     placeID = c_addr_id
'     placeHZ = c_addr_chn
'     placePY = c_addr_name
'     placeX = x_coord
'     placeY = y_coord
'
' PeoplePlaces.CSV
'     nameID
'     placeID
'     personPlaceRelation
'
' PeopleKinship.CSV
' node1_ID, node2_ID, kinshipRelation
'     node1 = str(c_person_id) for node1
'     node2 = str(c_node_id) for node2
'     kinshipRelation = c_link_desc
'
' first see if there are any records to process
'
If Me.ZZ_SCRATCH_P_ASSOC.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdNeo4j_Click
End If
'
' allocate the file variables
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
'
' next get the People file
'
Dim tRstPeople As DAO.Recordset, tRstAssoc As DAO.Recordset, tRstPlace As DAO.Recordset, tRstPeoplePlace As DAO.Recordset
Dim tStr As String, tC As String, tQueryStr As String, tRstAssocCode As DAO.Recordset
Dim gStream As ADODB.Stream, tCodeStr As String
'
' set up the stream to write to
'
Set gStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
    gStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf CodeFrame.Value = 2 Then
    gStream.Charset = "big5"
    tCodeStr = "BIG5"
ElseIf CodeFrame.Value = 3 Then
    gStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    gStream.Charset = "ascii"
    tCodeStr = "ascii"
End If

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
```

```

If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If

    '
    ' now process the file (second true removed to make ASCII)
    '
    'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
    'Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)
    '
    ' we have a file name: now open the stream for writing

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    '
    ' prepare the temp tables for the people, place, peoplePlace and assoc codes

    Dim cmdSQL As ADODB.Command
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    ' Get the people from 5 sources: c_person_id, c_assoc_id, c_kin_id, c_assoc_kin_id, and c_assoc_clai
mer_id

    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_TEXT"
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (1)
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_person_
id " + _
        "FROM ZZ_SCRATCH_ASSOC WHERE (((ZZ_SCRATCH_ASSOC.c_person_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (2)
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_assoc_i
d " + _
        "FROM ZZ_SCRATCH_ASSOC WHERE (((ZZ_SCRATCH_ASSOC.c_assoc_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (3)
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_kin_id
" + _
        "FROM ZZ_SCRATCH_ASSOC WHERE (((ZZ_SCRATCH_ASSOC.c_kin_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' copy the data for people (4)
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_assoc_k
in_id " + _
        "FROM ZZ_SCRATCH_ASSOC WHERE (((ZZ_SCRATCH_ASSOC.c_assoc_kin_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr

```



```

        ' indexyear = c_index_year INT
        ,
        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + tC
        End If

        ' dynasty information
        ,
        If IsNull(!c_dynasty) Then
            tStr = tStr + "unknown" + tC
        Else
            If tCodeStr = "ascii" Then
                tStr = tStr + !c_dynasty + tC
            Else
                tStr = tStr + !c_dynasty_chn + tC
            End If
        End If

        '
        sex = c_female > (F,M)
        tStr = tStr + IIf(!c_female, "F", "M")
        ,
        gStream.WriteText tStr, adWriteLine
        ,
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now places: since the association "event" is not linked to a place, the only addresses are the index
addresses
'
' of the people involved, recorded in ZZ_SCRATCH_PEOPLE
'
' get a file name
'
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    ' now process the file
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_addr_id, ZZ_SCRATCH_PEOPLE.c_addr_name, ZZ_SCRATCH_P
PEOPLE.c_addr_chn, " + _
                "ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord " + _
                "FROM ZZ_SCRATCH_PEOPLE"

    Set tRstPlace = CurrentDb.OpenRecordset(tQueryStr)
    ,
    If tCodeStr = "ascii" Then
        tStr = "placeID" + tC + "placePY" + tC + "placeX" + tC + "placeY"
    End If

```

```

Else
    tStr = "placeID" + tC + "placePY" + tC + "placeHZ" + tC + "placeX" + tC + "placeY"
End If
gStream.WriteText tStr, adWriteLine
With tRstPlace
    .MoveFirst
    Do While Not .EOF
        ' the ID of the place
        If Not IsNull(!c_addr_id) Then
            tStr = Trim(Str(!c_addr_id)) + tC
            '
            ' address name

            If IsNull(!c_addr_name) Then
                tStr = tStr + "unknown" + tC
            Else
                tStr = tStr + !c_addr_name + tC
            End If
            '
            If Not (tCodeStr = "ascii") Then
                If IsNull(!c_addr_chn) Then
                    tStr = tStr + "unknown" + tC
                Else
                    tStr = tStr + !c_addr_chn + tC
                End If
            End If
            '
            ' latitude = !y_coord
            If IsNull(!y_coord) Then
                tStr = tStr + "0.0" + tC
            Else
                tStr = tStr + Str(!y_coord) + tC
            End If
            '
            ' longitude = !x_coord
            If IsNull(!x_coord) Then
                tStr = tStr + "0.0" + tC
            Else
                tStr = tStr + Str(!x_coord) + tC
            End If
            '
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces
'
dlgSaveAs.InitialFileName = "PeoplePlaces_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
End If

```

```

End If
,
gStream.Open
,
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id, ZZ_SCRATCH_PEOPLE.c_addr_id, ZZ_SCRATCH_P
EOPLE.c_addr_type, " + _
                "ZZ_SCRATCH_PEOPLE.c_addr_desc, ZZ_SCRATCH_PEOPLE.c_addr_desc_chn " + _
                "FROM ZZ_SCRATCH_PEOPLE"

Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

tStr = "nameID" + tC + "placeID" + tC + "personPlaceTrans" + tC + "personPlaceHZ"

gStream.WriteText tStr, adWriteLine

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_addr_id) Then
            ,
            tStr = Trim(Str(!c_person_id)) + tC
            ,
            tStr = tStr + Trim(Str(!c_addr_id)) + tC
            ,
            tStr = tStr + Trim(Str(!c_addr_type))
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
,
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
,
' now the association records
,
dlgSaveAs.InitialFileName = "PeopleAssociations_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    ' now the associations: define the record structure
    ' Because of the complexity of the primary key, this gets a bit complicated
    ,
    Set tRstAssoc = CurrentDb.OpenRecordset("ZZ_SCRATCH_ASSOC", dbOpenDynaset)
    ,
    tStr = "Person1_ID" + tC + "Person2_ID" + tC + "Association_Code" + tC + "Kin_ID" + tC + "Kin_Code" +
tC + _
        "AssocKin_ID" + tC + "AssocKin_Code" + tC + "LiteraryGenreCode" + tC + "OccasionCode" + tC +
_
        "TopicCode" + tC + "InstitutionCode" + tC + "TextTitle" + tC + "AssociationClaimer_ID"
    gStream.WriteText tStr, adWriteLine
    'tGDF.WriteLine (tStr)

```

```

With tRstAssoc
.MoveFirst
Do While Not .EOF
  If Not IsNull(!c_assoc_code) Then
    tStr = Trim(Str(!c_person_id)) + tC
    '   node1 = str(c_person_id) for node1
    tStr = tStr + Trim(Str(!c_assoc_id)) + tC
    '   node2 = str(c_node_id) for node2
    tStr = tStr + Trim(Str(!c_assoc_code)) + tC

    '   kin ID
    If IsNull(!c_kin_id) Then
      tStr = tStr + "0" + tC
    Else
      tStr = tStr + Str(!c_kin_id) + tC
    End If

    '   kin code
    If IsNull(!c_kin_code) Then
      tStr = tStr + "0" + tC
    Else
      tStr = tStr + Str(!c_kin_code) + tC
    End If

    '   assoc kin ID
    If IsNull(!c_assoc_kin_id) Then
      tStr = tStr + "0" + tC
    Else
      tStr = tStr + Str(!c_assoc_kin_id) + tC
    End If

    '   assoc kin code
    If IsNull(!c_assoc_kin_code) Then
      tStr = tStr + "0" + tC
    Else
      tStr = tStr + Str(!c_assoc_kin_code) + tC
    End If

    '   literary genre code
    If IsNull(!c_litgenre_code) Then
      tStr = tStr + "0" + tC
    Else
      tStr = tStr + Str(!c_litgenre_code) + tC
    End If

    '   occasion code
    If IsNull(!c_occasion_code) Then
      tStr = tStr + "0" + tC
    Else
      tStr = tStr + Str(!c_occasion_code) + tC
    End If

    '   topic code
    If IsNull(!c_topic_code) Then
      tStr = tStr + "0" + tC
    Else
      tStr = tStr + Str(!c_topic_code) + tC
    End If

    '   institution code
    If IsNull(!c_inst_code) Then
      tStr = tStr + "0" + tC
    Else
      tStr = tStr + Str(!c_inst_code) + tC
    End If

    '   text title
    If IsNull(!c_text_title) Then
      tStr = tStr + "N/A" + tC
    Else
      tStr = tStr + !c_text_title + tC
    End If

    '   association claimer ID
    If IsNull(!c_assoc_claimer_id) Then
      tStr = tStr + "0" + tC
    Else
      tStr = tStr + Str(!c_assoc_claimer_id)
    End If
  End If
End While

```

```

        gStream.WriteText tStr, adWriteLine
    End If
    .MoveNext
Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
End If
'
' now the association codes
'
dlgSaveAs.InitialFileName = "AssociationCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_assoc_code, ZZ_SCRATCH_ASSOC.c_assoc_type, ZZ_SCRATCH_ASSOC.c_assoc_desc, " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_desc_chn " + _
        "FROM ZZ_SCRATCH_ASSOC"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr, dbOpenDynaset)
    '
    If tCodeStr = "ascii" Then
        tStr = "AssociationCode" + tC + "AssociationTypeID" + tC + "AssociationTrans"
    Else
        tStr = "AssociationCode" + tC + "AssociationTypeID" + tC + "AssociationTrans" + tC + "Association
HZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_assoc_code) Then
                '
                tStr = Trim(Str(!c_assoc_code)) + tC
                '
                tStr = tStr + Trim(!c_assoc_type) + tC
                '
                tStr = tStr + Trim(!c_assoc_desc)
                '
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_assoc_desc_chn)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    '
    ' now make sure all the data is copied to tStream

```

```

        gStream.Flush
        ' and write the stream to the file
        gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' there are codes that MAY require additional tables: c_kin_code, c_litgenrte_code, c_occasion_code, c_t
opic_code, c_inst_code
'
' test for kin codes
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_person_id " + _
            "FROM ZZ_SCRATCH_ASSOC " + _
            "WHERE (((ZZ_SCRATCH_ASSOC.c_kin_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' debug
MsgBox "Kinship code records = " + Trim(Str(tRecDeleted))
'
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "KinshipCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
        '
        gStream.Open
        '
        ' there is an additional complication for kinship codes because there are two sources: c_kin_code
and c_assoc_kin_code
        '
        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP"
        cmdSQL.Execute tRecDeleted
        '
        tQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_kin_code, c_kinrel, c_kinrel_total ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_kin_code, ZZ_SCRATCH_ASSOC.c_kin_desc, ZZ_SCRATCH
_ASSOC.c_kin_desc_chn " + _
            "FROM ZZ_SCRATCH_ASSOC " + _
            "WHERE (ZZ_SCRATCH_ASSOC.c_kin_code > 0)"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
        tQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_kin_code, c_kinrel, c_kinrel_total ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_assoc_kin_code, ZZ_SCRATCH_ASSOC.c_assoc_kin_desc
, ZZ_SCRATCH_ASSOC.c_assoc_kin_desc_chn " + _
            "FROM ZZ_SCRATCH_ASSOC " + _
            "WHERE (ZZ_SCRATCH_ASSOC.c_assoc_kin_code > 0)"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
        tQueryStr = "SELECT DISTINCT ZZ_KIN_LIST_TMP.c_kin_code, ZZ_KIN_LIST_TMP.c_kinrel, ZZ_KIN_LIST_TM
P.c_kinrel_total " + _
            "FROM ZZ_KIN_LIST_TMP"
        '
        Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)
    End If
End If

```

```

    If tCodeStr = "ascii" Then
        tStr = "KinshipCode" + tC + "KinshipTrans"
    Else
        tStr = "KinshipCode" + tC + "KinshipTrans" + tC + "KinshipHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_kin_code) Then
                ,
                tStr = Trim(Str(!c_kin_code)) + tC
                ,
                tStr = tStr + Trim(!c_kinrel)
                ,
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_kinrel_total)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    ,
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
,
' test for literary genre codes
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
"SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_person_id " + _
"FROM ZZ_SCRATCH_ASSOC " + _
"WHERE (((ZZ_SCRATCH_ASSOC.c_litgenre_code)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' debug
,
MsgBox "Literary genre code records = " + Trim(Str(tRecDeleted))
,
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "LiteraryGenreCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
    End If
    ,
    gStream.Open
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_litgenre_code, ZZ_SCRATCH_ASSOC.c_litgenre_desc,
ZZ_SCRATCH_ASSOC.c_litgenre_desc_chn " + _

```

```

        "FROM ZZ_SCRATCH_ASSOC " + _
        "WHERE ((ZZ_SCRATCH_ASSOC.c_litgenre_code)>0))"

Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "LitGenreCode" + tC + "LitGenreTrans"
Else
    tStr = "LitGenreCode" + tC + "LitGenreTrans" + tC + "LitGenreHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstAssocCode
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_litgenre_code) Then
            ,
            tStr = Trim(Str(!c_litgenre_code)) + tC
            ,
            tStr = tStr + Trim(!c_litgenre_desc)
            ,
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + Trim(!c_litgenre_desc_chn)
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
,
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
,
' test for institution codes
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_person_id " + _
            "FROM ZZ_SCRATCH_ASSOC " + _
            "WHERE (((ZZ_SCRATCH_ASSOC.c_inst_code)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' debug
,
MsgBox "Institution code records = " + Trim(Str(tRecDeleted))
,
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "InstitutionCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
    End If
End If
,

```



```

        gStream.Open
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_inst_code, ZZ_SCRATCH_ASSOC.c_inst_name_py, ZZ_SCRATCH_ASSOC.c_inst_name_hz " + _
        "FROM ZZ_SCRATCH_ASSOC " + _
        "WHERE ((ZZ_SCRATCH_ASSOC.c_inst_code)>0))"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "InstitutionCode" + tC + "InstitutionNamePY"
    Else
        tStr = "InstitutionCode" + tC + "InstitutionNamePY" + tC + "InstitutionNameHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_inst_code) Then
                ,
                tStr = Trim(Str(!c_inst_code)) + tC
                ,
                tStr = tStr + Trim(!c_inst_name_py)
                ,
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_inst_name_hz)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With

    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close

Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If

' test for occasion codes
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
    "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_person_id " + _
    "FROM ZZ_SCRATCH_ASSOC " + _
    "WHERE ((ZZ_SCRATCH_ASSOC.c_occasion_code)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' debug
,
MsgBox "Occasion code records = " + Trim(Str(tRecDeleted))
,
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "OccasionCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then

```

```

        tFileName = tFileName + ".csv"
    End If
End If
'
gStream.Open
'
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_occasion_code, ZZ_SCRATCH_ASSOC.c_occasion_desc,
ZZ_SCRATCH_ASSOC.c_occasion_desc_chn " + _
"FROM ZZ_SCRATCH_ASSOC " + _
"WHERE (((ZZ_SCRATCH_ASSOC.c_occasion_code)>0))"

Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "OccasionCode" + tC + "OccasionTrans"
Else
    tStr = "OccasionCode" + tC + "OccasionTrans" + tC + "OccasionHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstAssocCode
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_occasion_code) Then
            '
            tStr = Trim(Str(!c_occasion_code)) + tC
            '
            tStr = tStr + Trim(!c_occasion_desc)
            '
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + Trim(!c_occasion_desc_chn)
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
'
' test for topic codes
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
"SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_person_id " + _
"FROM ZZ_SCRATCH_ASSOC " + _
"WHERE (((ZZ_SCRATCH_ASSOC.c_topic_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' debug
'
MsgBox "Topic code records = " + Trim(Str(tRecDeleted))
'
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "TopicCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else

```

```

        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ' because ZZZ_NONKIN_BIOG_ADDR does not have the topic descriptions (must fix), I need to use a j
    oin to TOPIC_CODES
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_topic_code, SCHOLARLYTOPIC_CODES.c_topic_desc, SC
    HOLARLYTOPIC_CODES.c_topic_desc_chn " + _
    "FROM ZZ_SCRATCH_ASSOC INNER JOIN SCHOLARLYTOPIC_CODES ON ZZ_SCRATCH_ASSOC.c_topic_co
    de = SCHOLARLYTOPIC_CODES.c_topic_code " + _
    "WHERE (((ZZ_SCRATCH_ASSOC.c_topic_code)>0))"

    Set tRstAssocCode = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "TopicCode" + tC + "TopicTrans"
    Else
        tStr = "TopicCode" + tC + "TopicTrans" + tC + "TopicHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstAssocCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_topic_code) Then
                ,
                tStr = Trim(Str(!c_topic_code)) + tC
                ,
                tStr = tStr + Trim(!c_topic_desc)
                ,
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_topic_desc_chn)
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    ,
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
MsgBox "Finished saving to Neo4j"

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click

End Sub

Private Sub CmdPickAssoc_Click()
    On Error GoTo Err_CmdPickAssoc_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strAssoc As String

    TxtAssocCode.Visible = True

```

```

TxtAssocCode.SetFocus
strAssoc = TxtAssocCode.Text

stDocName = "frmPickAssoc_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strAssoc

If CurrentProject.AllForms("frmPickAssoc_multi").IsLoaded Then
    Dim intAssoc As Integer
    Dim strAssoc_DESC As String

    Forms!frmPickAssoc_multi.Form!TxtAssocID.Visible = True
    Forms!frmPickAssoc_multi.Form!TxtAssocID.SetFocus
    intAssoc = Forms!frmPickAssoc_multi.Form!TxtAssocID.Value
    Forms!frmPickAssoc_multi.Form!subTreeView.SetFocus
    Forms!frmPickAssoc_multi.Form!TxtAssocID.Visible = False
    TxtAssocCode.Value = intAssoc
    gAssocCodeStr = Trim(Str(intAssoc))

    If TxtAssocCode.Value = -1 Or TxtAssocCode.Value = -2 Then
        If TxtAssocCode.Value = -1 Then
            TxtAssocDesc.Value = "[[All]]"
            TxtAssocChn.Value = "[[All]]"
        Else
            TxtAssocDesc.Value = "[[Multi]]"
            TxtAssocChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
        End If

        Forms!frmPickAssoc_multi.Form!TxtTypeID.Visible = True
        Forms!frmPickAssoc_multi.Form!TxtTypeID.SetFocus
        strAssoc_DESC = Forms!frmPickAssoc_multi.Form!TxtTypeID.Value
        Forms!frmPickAssoc_multi.Form!subTreeView.SetFocus
        Forms!frmPickAssoc_multi.Form!TxtTypeID.Visible = False
        TxtTypeCode.Value = strAssoc_DESC
        gAssocTypeStr = Trim(strAssoc_DESC)

        If TxtTypeCode.Value = "000" Then
            TxtTypeDesc.Value = "[ALL]"
            TxtTypeChn.Value = "[ALL]"
        Else
            Forms!frmPickAssoc_multi.Form!TxtTypeDesc.Visible = True
            Forms!frmPickAssoc_multi.Form!TxtTypeDesc.SetFocus
            strAssoc_DESC = Forms!frmPickAssoc_multi.Form!TxtTypeDesc.Value
            Forms!frmPickAssoc_multi.Form!subTreeView.SetFocus
            Forms!frmPickAssoc_multi.Form!TxtTypeDesc.Visible = False
            TxtTypeDesc.Value = strAssoc_DESC

            Forms!frmPickAssoc_multi.Form!TxtTypeDescChn.Visible = True
            Forms!frmPickAssoc_multi.Form!TxtTypeDescChn.SetFocus
            strAssoc_DESC = Forms!frmPickAssoc_multi.Form!TxtTypeDescChn.Value
            Forms!frmPickAssoc_multi.Form!subTreeView.SetFocus
            Forms!frmPickAssoc_multi.Form!TxtTypeDescChn.Visible = False
            TxtTypeChn.Value = strAssoc_DESC
        End If
    Else
        Forms!frmPickAssoc_multi.Form!TxtAssocDesc.Visible = True
        Forms!frmPickAssoc_multi.Form!TxtAssocDesc.SetFocus
        strAssoc_DESC = Forms!frmPickAssoc_multi.Form!TxtAssocDesc.Value
        Forms!frmPickAssoc_multi.Form!subTreeView.SetFocus
        Forms!frmPickAssoc_multi.Form!TxtAssocDesc.Visible = False
        TxtAssocDesc.Value = strAssoc_DESC

        Forms!frmPickAssoc_multi.Form!TxtAssocDescChn.Visible = True
        Forms!frmPickAssoc_multi.Form!TxtAssocDescChn.SetFocus
        strAssoc_DESC = Forms!frmPickAssoc_multi.Form!TxtAssocDescChn.Value
        Forms!frmPickAssoc_multi.Form!subTreeView.SetFocus
        Forms!frmPickAssoc_multi.Form!TxtAssocDescChn.Visible = False
        TxtAssocChn.Value = strAssoc_DESC

        TxtTypeCode.Value = ""
        TxtTypeDesc.Value = "N/A"
        TxtTypeChn.Value = "N/A"
    End If

    DoCmd.Close acForm, stDocName
    '
    CmdQuery.Enabled = True
    CmdSaveAssociations.Enabled = True
Else
    CmdQuery.Enabled = False

```

```

    CmdSaveAssociations.Enabled = False
End If

CmdPickAssoc.SetFocus
TxtAssocCode.Visible = False

Exit_CmdPickAssoc_Click:
Exit Sub

Err_CmdPickAssoc_Click:
MsgBox Err.Description
Resume Exit_CmdPickAssoc_Click

End Sub

Private Sub CmdQuery_Click()
    On Error GoTo Err_Run_Query

    Dim rst As DAO.Recordset, tContinue As Integer
    Dim tRstAssoc As DAO.Recordset, tRstAddrList As DAO.Recordset, tRstDummy As DAO.Recordset
    Dim tQueryInsertStr As String, tQuerySelectStr As String, tQueryFromStr As String, tQueryWhereStr As String
    Dim tQueryStr As String, tRecDrop As Long, tStrWhereSQL As String
    Dim tUseAddr As Boolean

    Dim cmdSQL As ADODB.Command, tRecCount As Long

    Set cmdSQL = New ADODB.Command
    '
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' to clear the table, close and then delete records
    '
    Set tRstAssoc = ZZ_SCRATCH_ASSOC.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_AC", dbOpenDynaset)
    Set ZZ_SCRATCH_ASSOC.Form.Recordset = tRstDummy
    tRstAssoc.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ASSOC"
    cmdSQL.Execute tRecCount
    '
    ' now the people table
    '
    Set gRstPeople = ZZ_SCRATCH_P_ASSOC.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_AP", dbOpenDynaset)
    Set ZZ_SCRATCH_P_ASSOC.Form.Recordset = tRstDummy
    gRstPeople.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_ASSOC"
    cmdSQL.Execute tRecCount
    '
    ' now see if address IDs will be used. If so, zap the scratch file and repopulate
    '
    ' MsgBox "About to process address"
    If gUseADDRID Then
        '
        ' the strategy here is to fill the scratch file with all the relevant addresses from ZZZ_BELONGS_TO
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR"
        cmdSQL.Execute tRecCount
        '
        If ChkSubUnits.Value Then
            tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) " + _
                "SELECT DISTINCT ZZZ_BELONGS_TO.c_addr_id " + _
                "FROM ZZ_SCRATCH_ADDR_LIST INNER JOIN ZZZ_BELONGS_TO ON " + _
                "ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZZZ_BELONGS_TO.c_belongs_to"
        Else
            tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) " + _
                "SELECT DISTINCT c_addr_id " + _
                "FROM ZZ_SCRATCH_ADDR_LIST"
        End If
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecCount
        '
        ' see if we need to use the historical XY search
        '
        If ChkXYRef.Value Then
            '
            ' the strategy here is to dump the IDs to ZZ_ADDRESSES then copy to ZZ_SCRATCH_ADDR_LIST

```

```

' (I borrow ZZ_ADDRESSES from the Pick Addresses form in order to keep the initial selection
' of addresses for the query intact.)
'
' zap the list
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' run the query
'
' FrameXY.Value = 2 :: Narrow, FrameXY.Value = 1 :: Broad
'
If FrameXY.Value = 2 Then
    tStrWhereSQL = "WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.03) And " + _
        " (ADDR_CODES.x_coord)<=([ADDR_CODES_1].[x_coord]+0.03)) AND " + _
        " ((ADDR_CODES.y_coord)>=([ADDR_CODES_1].[y_coord]-0.03) And " + _
        " (ADDR_CODES.y_coord)<=([ADDR_CODES_1].[y_coord]+0.03)))"
Else
    tStrWhereSQL = "WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.06) And " + _
        " (ADDR_CODES.x_coord)<=([ADDR_CODES_1].[x_coord]+0.06)) AND " + _
        " ((ADDR_CODES.y_coord)>=([ADDR_CODES_1].[y_coord]-0.06) And " + _
        " (ADDR_CODES.y_coord)<=([ADDR_CODES_1].[y_coord]+0.06)))"
End If
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT DISTINCT ADDR_CODES.c_addr_id " + _
    "FROM ADDR_CODES, ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES AS ADDR_CODES_1 ON " + _
    "ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES_1.c_addr_id " + tStrWhereSQL

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now get the address IDs from the initial list that have no xy coordinates
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT ZZ_SCRATCH_ADDR.c_addr_id " + _
    "FROM ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES ON " + _
    "ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES.c_addr_id " + _
    "WHERE (((ADDR_CODES.x_coord) Is Null)) OR (((ADDR_CODES.y_coord) Is Null))"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' zap ZZ_SCRATCH_ADDR
'
tQueryStr = "DELETE * FROM ZZ_SCRATCH_ADDR"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' copy the list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT ZZ_ADDRESSES.c_addr_id " + _
    "FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' zap the temporary list
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
End If

tUseAddr = True
Else
    tUseAddr = False
End If
' next build the appropriate query string

tQueryInsertStr = "INSERT INTO ZZ_SCRATCH_ASSOC ( c_person_id, c_assoc_id, c_assoc_code, c_assoc_desc_chn, c_
assoc_desc, " + _
    "c_kin_code, c_kin_id, c_kin_name, c_kin_chn, c_assoc_kin_code, " + _
    "c_assoc_kin_id, c_assoc_kin_name, c_assoc_kin_chn, c_litgenre_code, c_litgenre_desc, c_litgenre_desc_chn
, " + _
    "c_occasion_code, c_occasion_desc, c_occasion_desc_chn, c_topic_code, c_topic_desc, c_topic_desc_chn, c_i
nst_code, " + _
    "c_inst_name_hz, c_text_title, c_assoc_claimer_id, c_assoc_claimer_name, c_assoc_claimer_name_chn, c_kin
_desc, " + _
    "c_kin_desc_chn, c_inst_name_py, c_assoc_count, c_assoc_first_year, c_assoc_last_year, c_assoc_place_addr
_id, c_assoc_place_addr_name, " + _
    "c_assoc_place_addr_chn, c_source_text_chn, c_source, c_source_text, c_distance ) "

```

```

tQuerySelectStr = "SELECT ZZZ_NONKIN_BIOG_ADDR.c_personid, ZZZ_NONKIN_BIOG_ADDR.c_node_id, ZZZ_NONKIN_BIOG_ADDR.c_link_code, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_link_chn, ZZZ_NONKIN_BIOG_ADDR.c_link_desc, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_kin_code, ZZZ_NONKIN_BIOG_ADDR.c_kin_id, ZZZ_NONKIN_BIOG_ADDR.c_kin_name, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_kin_chn, ZZZ_NONKIN_BIOG_ADDR.c_assoc_kin_code, ZZZ_NONKIN_BIOG_ADDR.c_assoc_kin_id, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_assoc_kin_name, ZZZ_NONKIN_BIOG_ADDR.c_assoc_kin_chn, ZZZ_NONKIN_BIOG_ADDR.c_litg_enre_code, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_lit_genre_desc, ZZZ_NONKIN_BIOG_ADDR.c_lit_genre_desc_chn, ZZZ_NONKIN_BIOG_ADDR.c_occasion_code, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_occasion_desc, ZZZ_NONKIN_BIOG_ADDR.c_occasion_desc_chn, ZZZ_NONKIN_BIOG_ADDR.c_topic_code, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_topic_desc, ZZZ_NONKIN_BIOG_ADDR.c_topic_desc_chn, ZZZ_NONKIN_BIOG_ADDR.c_inst_code, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_inst_name_hz, ZZZ_NONKIN_BIOG_ADDR.c_text_title, ZZZ_NONKIN_BIOG_ADDR.c_assoc_claimer_id, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_assoc_claimer_name, ZZZ_NONKIN_BIOG_ADDR.c_assoc_claimer_chn, ZZZ_NONKIN_BIOG_ADDR.c_kinrel, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_kinrel_chn, ZZZ_NONKIN_BIOG_ADDR.c_inst_name_py, ZZZ_NONKIN_BIOG_ADDR.c_link_count, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_assoc_first_year, ZZZ_NONKIN_BIOG_ADDR.c_assoc_last_year, ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_id, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_name, ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_chn, ZZZ_NONKIN_BIOG_ADDR.c_source_chn, " +
    "ZZZ_NONKIN_BIOG_ADDR.c_source, ZZZ_NONKIN_BIOG_ADDR.c_source_title, ZZZ_NONKIN_BIOG_ADDR.c_distance "

' set the from tables and the dynasties table, if needed

' With the introduction of multi-select, I now join the ZZ_ASSOC_CODE to the NONKIN to get the associations
If tUseAddr Then
    If gUseDynasties And gToDynasty > -2 Then
        tQueryFromStr = "FROM (DYNASTIES INNER JOIN (ZZ_ASSOC_CODE INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR ON " +
            "ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_person_addr_id) ON ZZ_ASSOC_CODE.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) ON " +
            "DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy) INNER JOIN ASSOC_CODE_TYPE_REL ON " +
            "ZZZ_NONKIN_BIOG_ADDR.c_link_code = ASSOC_CODE_TYPE_REL.c_assoc_code "

    Else
        tQueryFromStr = "FROM (ZZ_ASSOC_CODE INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR ON " +
            "ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_person_addr_id) ON ZZ_ASSOC_CODE.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) " +
            "INNER JOIN ASSOC_CODE_TYPE_REL ON ZZZ_NONKIN_BIOG_ADDR.c_link_code = ASSOC_CODE_TYPE_REL.c_assoc_code "

    End If
Else
    If gUseDynasties And gToDynasty > -2 Then
        tQueryFromStr = "FROM DYNASTIES INNER JOIN (ASSOC_CODE_TYPE_REL INNER JOIN ZZZ_NONKIN_BIOG_ADDR " +
            "ON ASSOC_CODE_TYPE_REL.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy "
        tQueryFromStr = "FROM DYNASTIES INNER JOIN ((ZZ_ASSOC_CODE INNER JOIN ZZZ_NONKIN_BIOG_ADDR ON " +
            "ZZ_ASSOC_CODE.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) INNER JOIN ASSOC_CODE_TYPE_REL ON " +
            "ZZZ_NONKIN_BIOG_ADDR.c_link_code = ASSOC_CODE_TYPE_REL.c_assoc_code) ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy "
    Else
        tQueryFromStr = "FROM (ZZ_ASSOC_CODE INNER JOIN ZZZ_NONKIN_BIOG_ADDR ON ZZ_ASSOC_CODE.c_assoc_code = ZZZ_NONKIN_BIOG_ADDR.c_link_code) " +
            "INNER JOIN ASSOC_CODE_TYPE_REL ON ZZZ_NONKIN_BIOG_ADDR.c_link_code = ASSOC_CODE_TYPE_REL.c_assoc_code "

    End If
End If

' set the where conditions
'
' Start with the index years
'
tQueryWhereStr = ""

If gUseIndexYears Then
    '
    ' four possibilities
    '
    If gFromStr = "" And gToStr = "" Then
        tQueryWhereStr = ""
    ElseIf gFromStr = "" Then
        tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)<=" + gToStr + ") "

```

```

ElseIf gToStr = "" Then
    tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)>= " + gFromStr + ") "
Else
    tQueryWhereStr = "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)<= " + gToStr + ") AND " + _
        "((ZZZ_NONKIN_BIOG_ADDR.c_index_year)>= " + gFromStr + ") "
End If
ElseIf gUseDynasties Then
    ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
    If gFromDynasty = -2 Then
        tQueryWhereStr = "Where (((ZZZ_NONKIN_BIOG_ADDR.c_dy) > 0 ) "
    ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_dy)=" + Str(gFromDynasty) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
            "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    Else
        tQueryWhereStr = ""
    End If
End If

If Not (tQueryWhereStr = "") Then
    tQueryWhereStr = tQueryWhereStr + ")"
End If

cmdSQL.CommandText = tQueryInsertStr + tQuerySelectStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecCount

' Because the quesry is complex enough as is, I add some xy information and get information from BIOG_MAIN i
n three separate steps
If tRecCount > 0 Then
    cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_ASSOC INNER JOIN ADDR_CODES ON ZZ_SCRATCH_ASSOC.c_assoc_place_addr
r_id = ADDR_CODES.c_addr id " + _
        "SET ZZ_SCRATCH_ASSOC.c_assoc_place_addr_xcoord = [ADDR_CODES].[x_coord], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_place_addr_ycoord = [ADDR_CODES].[y_coord]"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_ASSOC INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_ASSOC.c_person_id =
ZZZ_BIOG_MAIN.c_personid " + _
        "SET ZZ_SCRATCH_ASSOC.c_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SCRATCH_ASSOC.c_name_chn = [ZZZ_BIOG_MAIN
].[c_name_chn], " + _
        "ZZ_SCRATCH_ASSOC.c_index_year = [ZZZ_BIOG_MAIN].[c_index_year], " + _
        "ZZ_SCRATCH_ASSOC.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
        "ZZ_SCRATCH_ASSOC.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
        "ZZ_SCRATCH_ASSOC.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
        "ZZ_SCRATCH_ASSOC.c_dy = [ZZZ_BIOG_MAIN].[c_dy], ZZ_SCRATCH_ASSOC.c_dynasty = [ZZZ_BIOG_MAIN].[c
dynasty], " + _
        "ZZ_SCRATCH_ASSOC.c_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn], " + _
        "ZZ_SCRATCH_ASSOC.c_sex = IIf([ZZZ_BIOG_MAIN].[c_female], 'F', 'M'), " + _
        "ZZ_SCRATCH_ASSOC.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
        "ZZ_SCRATCH_ASSOC.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
        "ZZ_SCRATCH_ASSOC.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
        "ZZ_SCRATCH_ASSOC.x_coord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SCRATCH_ASSOC.y_coord = [ZZZ_BIOG_MAIN
].[y_coord]"
    cmdSQL.Execute tRecCount

    cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_ASSOC INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_ASSOC.c_assoc_id = Z
ZZ_BIOG_MAIN.c_personid " + _
        "SET ZZ_SCRATCH_ASSOC.c_assoc_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SCRATCH_ASSOC.c_assoc_chn = [ZZZ_BI
OG_MAIN].[c_name_chn], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_index_year = [ZZZ_BIOG_MAIN].[c_index_year], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_dy = [ZZZ_BIOG_MAIN].[c_dy], ZZ_SCRATCH_ASSOC.c_assoc_dynasty = [ZZZ_BI
OG_MAIN].[c_dynasty], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_sex = IIf([ZZZ_BIOG_MAIN].[c_female], 'F', 'M'), " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
        "ZZ_SCRATCH_ASSOC.c_assoc_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
        "ZZ_SCRATCH_ASSOC.assoc_xcoord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SCRATCH_ASSOC.assoc_ycoord = [ZZZ_B
IOG_MAIN].[y_coord]"
    cmdSQL.Execute tRecCount

```



```

End If
'
' the next step is to clean up the data (remove duplicates) and make the table of people from the associatio
ns
'
If tRecCount = 0 Then
  CmdPajek.Enabled = False
  CmdUCINet.Enabled = False
  CmdGephi.Enabled = False
Else
  CmdPajek.Enabled = True
  CmdUCINet.Enabled = True
  CmdGephi.Enabled = True
  '
  ' remove duplicated associations
  '
  ' (1) mark the passive members of the pair
  '
  tQueryStr = "UPDATE (ZZ_SCRATCH_ASSOC AS ZZ_SCRATCH_ASSOC_1 INNER JOIN ZZ_SCRATCH_ASSOC ON " + _
    "(ZZ_SCRATCH_ASSOC.c_person_id = ZZ_SCRATCH_ASSOC_1.c_assoc_id) AND " + _
    "(ZZ_SCRATCH_ASSOC_1.c_person_id = ZZ_SCRATCH_ASSOC.c_assoc_id)) " + _
    "INNER JOIN ASSOC_CODES ON (ZZ_SCRATCH_ASSOC_1.c_assoc_code = ASSOC_CODES.c_assoc_pair) AND " + _
    "(ZZ_SCRATCH_ASSOC.c_assoc_code = ASSOC_CODES.c_assoc_code) " + _
    "SET ZZ_SCRATCH_ASSOC.c_delete = 1 " + _
    "WHERE (((ASSOC_CODES.c_assoc_role_type)='P'))"
  '
  cmdSQL.CommandText = tQueryStr
  cmdSQL.Execute tRecCount
  '
  ' (2) mark the higher person ID from mutual relations
  '
  tQueryStr = "UPDATE (ZZ_SCRATCH_ASSOC AS ZZ_SCRATCH_ASSOC_1 INNER JOIN ZZ_SCRATCH_ASSOC ON " + _
    "(ZZ_SCRATCH_ASSOC.c_person_id = ZZ_SCRATCH_ASSOC_1.c_assoc_id) AND " + _
    "(ZZ_SCRATCH_ASSOC_1.c_person_id = ZZ_SCRATCH_ASSOC.c_assoc_id)) INNER JOIN ASSOC_CODES ON " + _
    "(ZZ_SCRATCH_ASSOC_1.c_assoc_code = ASSOC_CODES.c_assoc_pair) AND " + _
    "(ZZ_SCRATCH_ASSOC.c_assoc_code = ASSOC_CODES.c_assoc_code) " + _
    "SET ZZ_SCRATCH_ASSOC.c_delete = 1 " + _
    "WHERE (((ASSOC_CODES.c_assoc_role_type)='M') AND ((ZZ_SCRATCH_ASSOC.c_person_id)>[ZZ_SCRATCH_ASSOC_1
].c_person_id)))"
  '
  cmdSQL.CommandText = tQueryStr
  cmdSQL.Execute tRecCount
  '
  ' now delete
  '
  tQueryStr = "DELETE * FROM ZZ_SCRATCH_ASSOC WHERE c_delete = 1"
  '
  cmdSQL.CommandText = tQueryStr
  cmdSQL.Execute tRecCount
  '
  ' now get the people: dump the person first, and then the associate, into a temporary table, then copy
  '
  tQueryStr = "DELETE * FROM ZZ_SCRATCH_IMPORT_PEOPLE"
  '
  cmdSQL.CommandText = tQueryStr
  cmdSQL.Execute tRecCount
  '
  tQueryStr = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_dy,
c_dynasty, c_dynasty_chn, " + _
    "c_sex, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
    "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_person_id, ZZ_SCRATCH_ASSOC.c_name, ZZ_SCRATCH_ASSOC.c_name_chn,
" + _
    "ZZ_SCRATCH_ASSOC.c_index_year, ZZ_SCRATCH_ASSOC.c_dy, ZZ_SCRATCH_ASSOC.c_dynasty, ZZ_SCRATCH_ASSOC.c
_dynasty_chn, " + _
    "ZZ_SCRATCH_ASSOC.c_sex, ZZ_SCRATCH_ASSOC.c_addr_id, ZZ_SCRATCH_ASSOC.c_addr_name, " + _
    "ZZ_SCRATCH_ASSOC.c_addr_chn, ZZ_SCRATCH_ASSOC.x_coord, ZZ_SCRATCH_ASSOC.y_coord " + _
    "FROM ZZ_SCRATCH_ASSOC"
  cmdSQL.CommandText = tQueryStr
  cmdSQL.Execute tRecCount
  '
  ' now copy all the assoc IDs that do not also appear as person IDs
  '
  tQueryStr = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_dy,
c_dynasty, c_dynasty_chn, " + _
    "c_sex, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
    "SELECT DISTINCT ZZ_SCRATCH_ASSOC.c_assoc_id, ZZ_SCRATCH_ASSOC.c_assoc_name, ZZ_SCRATCH_ASSOC.c_assoc
_chn, " + _
    "ZZ_SCRATCH_ASSOC.c_assoc_index_year, ZZ_SCRATCH_ASSOC.c_assoc_dy, ZZ_SCRATCH_ASSOC.c_assoc_dynasty,
ZZ_SCRATCH_ASSOC.c_assoc_dynasty_chn, " + _
    "ZZ_SCRATCH_ASSOC.c_assoc_sex, ZZ_SCRATCH_ASSOC.c_assoc_addr_id, " + _

```

```

        "ZZ_SCRATCH_ASSOC.c_assoc_addr_name, ZZ_SCRATCH_ASSOC.c_assoc_addr_chn, ZZ_SCRATCH_ASSOC.assoc_xcoord
, ZZ_SCRATCH_ASSOC.assoc_ycoord " + _
        "FROM ZZ_SCRATCH_ASSOC LEFT JOIN ZZ_SCRATCH_ASSOC AS ZZ_SCRATCH_ASSOC_1 ON ZZ_SCRATCH_ASSOC.c_assoc_i
d = ZZ_SCRATCH_ASSOC_1.c_person_id " + _
        "WHERE (((ZZ_SCRATCH_ASSOC_1.c_assoc_code) Is Null)))"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecCount

        tQueryStr = "INSERT INTO ZZ_SCRATCH_P_ASSOC ( c_person_id, c_name, c_name_chn, c_index_year, c_dy, c_dyna
sty, c_dynasty_chn, c_sex, c_addr_id, " + _
        "c_addr_name, c_addr_chn, x_coord, y_coord, c_addr_type, c_addr_desc, c_addr_desc_chn ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id, ZZ_SCRATCH_IMPORT_PEOPLE.c_name, ZZ_SCRATCH_I
MPORT_PEOPLE.c_name_chn, " + _
        "ZZ_SCRATCH_IMPORT_PEOPLE.c_index_year, ZZ_SCRATCH_IMPORT_PEOPLE.c_dy, ZZ_SCRATCH_IMPORT_PEOPLE.c_dy
nasty, " + _
        "ZZ_SCRATCH_IMPORT_PEOPLE.c_dynasty_chn, ZZ_SCRATCH_IMPORT_PEOPLE.c_sex, ZZ_SCRATCH_IMPORT_PEOPLE.c_
addr_id, " + _
        "ZZ_SCRATCH_IMPORT_PEOPLE.c_addr_name, ZZ_SCRATCH_IMPORT_PEOPLE.c_addr_chn, ZZ_SCRATCH_IMPORT_PEOPLE
.x_coord, " + _
        "ZZ_SCRATCH_IMPORT_PEOPLE.y_coord, ZZZ_BIOG_MAIN.c_index_addr_type_code, ZZZ_BIOG_MAIN.c_index_addr_
type_desc, ZZZ_BIOG_MAIN.c_index_addr_type_chn " + _
        "FROM ZZZ_BIOG_MAIN INNER JOIN ZZ_SCRATCH_IMPORT_PEOPLE ON ZZZ_BIOG_MAIN.c_personid = ZZ_SCRATCH_IMP
ORT_PEOPLE.c_person_id"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecCount
    '
    ' now get the new index year information from ZZZ_BIOG_MAIN
    '
    cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_P_ASSOC INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_ASSOC.c_person_i
d = ZZZ_BIOG_MAIN.c_personid " + _
        "SET ZZ_SCRATCH_P_ASSOC.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
        "ZZ_SCRATCH_P_ASSOC.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
        "ZZ_SCRATCH_P_ASSOC.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz]"
    cmdSQL.Execute tRecCount
    '
    ' the final step is to calculate the xy_count
    '
    If tRecCount > 0 Then
        '
        cmdSQL.CommandText = "Delete * from tmpXY"
        cmdSQL.Execute tRecDeleted
        '
        tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
            "SELECT ZZ_SCRATCH_P_ASSOC.x_coord, ZZ_SCRATCH_P_ASSOC.y_coord, Count(ZZ_SCRATCH_P_ASSOC.x_coord)
" + _
            "AS CountOfx_coord, Count(ZZ_SCRATCH_P_ASSOC.y_coord) AS CountOfy_coord " + _
            "FROM ZZ_SCRATCH_P_ASSOC " + _
            "GROUP BY ZZ_SCRATCH_P_ASSOC.x_coord, ZZ_SCRATCH_P_ASSOC.y_coord"
        '
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecCount
        '
        tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_P_ASSOC ON (tmpXY.y_coord = " + _
            "ZZ_SCRATCH_P_ASSOC.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_P_ASSOC.x_coord) " + _
            "SET ZZ_SCRATCH_P_ASSOC.xy_count = [tmpXY].[CountOfx_coord]"
        '
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecCount
        CmdGIS.Enabled = True
        CmdStoreID.Enabled = True
        CmdNeo4j.Enabled = True
    Else
        CmdNeo4j.Enabled = False
        CmdGIS.Enabled = False
        CmdStoreID.Enabled = False
    End If
End If

Exit_Run_Query:
    '
    ' now reopen the tables
    '
    Set tRstAssoc = CurrentDb.OpenRecordset("ZZ_SCRATCH_ASSOC", dbOpenDynaset)
    Set ZZ_SCRATCH_ASSOC.Form.Recordset = tRstAssoc
    '
    Set gRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_ASSOC", dbOpenDynaset)
    Set ZZ_SCRATCH_P_ASSOC.Form.Recordset = gRstPeople
    '
    ' close everything

```

```

'
Set rst = Nothing
Set AssocQuery = Nothing
Set AddressQuery = Nothing
Set tRstDummy = Nothing
Set cmdSQL = Nothing
Exit Sub

Err_Run_Query:
MsgBox Err.Description
Resume Exit_Run_Query

End Sub

Private Sub calculate_xy_count()
Dim tX As Double, tY As Double, tXY As Integer
Dim tBM As Variant, tWrite As Integer
'
' the strategy is to first throw a bookmark at the first new value
' then count the number, then go back to the bookmark and update each record
'
With gRstPeople
.Index = "xy"
.MoveFirst

tX = -1#
tY = -1#
tXY = 0
tWrite = 0
tBM = .Bookmark

Do While Not .EOF
If tX <> !x_coord Or tY <> !y_coord Then
If tWrite = 1 Then
' go back to the first record with the value
.Bookmark = tBM
Do While tX = !x_coord And tY = !y_coord
.Edit
!xy_count = tXY
.Update
.MoveNext
Loop
Else
tWrite = 1
End If
' reset
tXY = 0
tBM = .Bookmark
tX = !x_coord
tY = !y_coord
End If
' increment the count and move to the next
tXY = tXY + 1
.MoveNext
Loop
'
' the last xy value still needs to be written
'
.Bookmark = tBM
Do While Not .EOF
.Edit
!xy_count = tXY
.Update
.MoveNext
Loop
.Index = "index_year"
End With
End Sub

Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
Call writeKML
Exit Sub
End If

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer

```

```

Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tPinyin As Boolean
Dim tFileSystem, tGDF
'
'   This program will dump the results to a .gis file
'
If ZZ_SCRATCH_P_ASSOC.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGIS_Click
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False
If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
'   next get a file
'

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdGIS_Click
    Else
        '   make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If
    '
    '   write the file
    '
    'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
    '
    '   process the table
    '
    Set tRstNode = ZZ_SCRATCH_P_ASSOC.Form.Recordset
    tC = Chr(9) ' the tab
    '
    With tRstNode
        '
        '   write the header
        '
        If tPinyin Then
            tStr = "Name" + tC + "Female" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "X" + tC + "Y" + tC + "xy_count"
        Else
            tStr = "Name" + tC + "NameChn" + tC + "Female" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC + "xy_count"
        End If
        tStream.WriteText tStr, adWriteLine
        .MoveFirst
    End With

```

```

Do While Not .EOF
    ' must guard against NULLs
    ,
    If Trim(!c_name) = "" Then
        tStr = "[?]" + tC
    Else
        tStr = !c_name + tC
    End If

    If Not tPinyin Then
        If Trim(!c_name_chn) = "" Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_name_chn + tC
        End If
    End If

    If IsNull(!c_sex) Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_sex + tC
    End If

    If IsNull(!c_index_year) Then
        tStr = tStr + "-2000" + tC
    Else
        tStr = tStr + Str(!c_index_year) + tC
    End If

    ' here guard against blanks as well

    If IsNull(!c_addr_name) Then
        tStr = tStr + "[?]" + tC
    ElseIf Trim(!c_addr_name) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If

    If Not tPinyin Then
        If IsNull(!c_addr_chn) Then
            tStr = tStr + "[?]" + tC
        ElseIf Trim(!c_addr_chn) = "" Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_addr_chn + tC
        End If
    End If

    If IsNull(!x_coord) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Str(!x_coord) + tC
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Str(!y_coord) + tC
    End If

    If IsNull(!xy_count) Then
        tStr = tStr + "0"
    Else
        tStr = tStr + Str(!xy_count)
    End If

    tStream.WriteText tStr, adWriteLine
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
,
Else
    'The user pressed Cancel.
End If

```

```

Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGIS_Click:
Exit Sub

Err_CmdGIS_Click:
MsgBox Err.Description
Resume Exit_CmdGIS_Click

End Sub

Private Sub CmdSaveAssociations_Click()
On Error GoTo Err_CmdSaveAssociations_Click
'
' This program will store the current list of office IDs to a .txt file
'
Dim tStream As ADODB.Stream, tStreamNoBOM As ADODB.Stream
Set tStream = New ADODB.Stream

tStream.Charset = "utf-8"
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set tStreamNoBOM = New ADODB.Stream
tStreamNoBOM.Type = adTypeBinary
tStreamNoBOM.Open
'
' next get a file
'

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstIDs As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "assoc_code_list.txt"
If dlgSaveAs.Show = -1 Then
'
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
tFileName = tFN
If Not tFileName = "" Then
Exit For
End If
Next
If tFileName = "" Then
MsgBox "Bad file Name."
GoTo Exit_CmdSaveAssociations_Click
Else
' make sure the file name has a txt extension
If Len(tFileName) < 5 Then
tFileName = tFileName + ".txt"
ElseIf Not (LCase(Right(tFileName, 4)) = ".txt") Then
tFileName = tFileName + ".txt"
End If
End If
'
' write the file
'
' process the table
'

tStr = "SELECT ZZ_ASSOC_CODE.c_assoc_code, ASSOC_CODES.c_assoc_desc, ASSOC_CODES.c_assoc_desc_chn " + _
"FROM ZZ_ASSOC_CODE INNER JOIN ASSOC_CODES ON ZZ_ASSOC_CODE.c_assoc_code = ASSOC_CODES.c_assoc_code"

Set tRstIDs = CurrentDb.OpenRecordset(tStr, dbOpenDynaset)
'
tTab = Chr(9)

With tRstIDs

```

```

        .MoveFirst
        ' MsgBox "writing file"
        Do While Not .EOF
            '
            tStr = Str(!c_assoc_code) + tTab + !c_assoc_desc + tTab + !c_assoc_desc_chn
            tStream.WriteText tStr, adWriteLine
            .MoveNext
        Loop
    End With
    ' now make sure all the data is copied to tStream
    tStream.Flush
    tStream.Position = 3
    ' and write the stream to the file
    tStream.CopyTo tStreamNoBOM
    tStreamNoBOM.SaveToFile tFileName, adSaveCreateOverWrite
    '
Else
    'The user pressed Cancel.
End If

Set tRstIDs = Nothing

tStream.Close
Set tStream = Nothing
tStreamNoBOM.Close
Set tStreamNoBOM = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdSaveAssociations_Click:
Exit Sub

Err_CmdSaveAssociations_Click:
MsgBox Err.Description
Resume Exit_CmdSaveAssociations_Click

End Sub

Private Sub CmdToDynasty_Click()
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strToDynasty As String

    If gToDynasty = -1 Then
        strToDynasty = ""
    Else
        strToDynasty = Str(gToDynasty)
    End If

    stDocName = "frmPickDynasty"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

    If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

        Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
        gToDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
        gToDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
        gToDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
        '
        ' check to see if we have a problem and reject selection if needed
        '
        If gFromDynasty > -1 Then
            If gFromDynastyBegin > gToDynastyEnd Then
                MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
                gToDynasty = -1
                TxtToDynasty.Value = ""
                TxtToDynastyPY.Value = ""
            End If
        End If
        '
        ' value is OK
        '
        If gToDynasty > -1 Then
            Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
            TxtToDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value
        End If
    End If

```

```

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
        TxtToDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
    End If

    DoCmd.Close acForm, stDocName
    '
    ' reset FromDynasty if necessary (-2 = all dynasties)
    '
    If gFromDynasty = -2 Then
        gFromDynasty = -1
        TxtFromDynasty.Value = ""
        TxtFromDynastyPY.Value = ""
    End If
    '
End If

End Sub

Private Sub CmdUCINet_Click()
On Error GoTo Err_CmdUCINet_Click
    '
    ' This program will dump the results of the search to a .vna file
    '
    ' for the moment I'll just describe the format of the .vna file
    '
    ' *node data
    ' ID index_year sex x_coord y_coord nodedist
    '     ID = str(c_person_id)
    '     indexyear = c_index_year INT
    '     nodedist = c_node_dist INT
    '     sex = c_female > (F,M)
    ' *node properties
    ' ID color shape size shortlabel active
    '     color = red (1), orange (2), yellow (3), green (4), blue (5)
    '     shortlabel = c_name
    '     shape = 2
    '     active = TRUE
    '
    ' *tie data
    ' from to edgetype nodedist
    '     from = str(c_person_id)
    '     to = str(c_node_id)
    '     edgetype= c_link_type (K,N)
    '
    ' *tie properties
    ' from to color size active
    '     from = str(c_person_id)
    '     to = str(c_node_id)
    '     color = red (255), orange (26367), yellow (65535), green (32768), blue (16711680)
    '     size = 1-5 (the weight)
    '
    ' the central question is whether to do distance optimizations
    '
    ' first see if there are any records to process
    '
    If ZZ_SCRATCH_ASSOC.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_CmdUCINet_Click
    End If
    '
    If ZZ_SCRATCH_P_ASSOC.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_CmdUCINet_Click
    End If
    '
    ' next get a file
    '
    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tRstNode As DAO.Recordset, tRstAssocType As DAO.Recordset
    Dim tRstEdge As DAO.Recordset
    Dim tStr As String, tC As String, ti As Integer, tSearchStr As String
    Dim tColor(20) As String, tQuote As String
    Dim tFileSystem, tVNA

    Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    ' open the assoc type look-up table

```



```

Set tRstAssocType = CurrentDb.OpenRecordset("ASSOC_CODE_TYPE_REL", dbOpenDynaset)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network.vna"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdUCINet_Click
        Else
            ' make sure the file name has a vna extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".vna"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".vna") Then
                tFileName = tFileName + ".vna"
            End If
        End If
        '
        ' now process the file (second true removed to make ASCII)
        '
        Set tFileSystem = CreateObject("Scripting.FileSystemObject")
        Set tVNA = tFileSystem.CreateTextFile(tFileName, True)

        ' process the two tables
        '
        Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SCRATCH_ASSOC", dbOpenDynaset)
        Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_ASSOC", dbOpenDynaset)
        tQuote = Chr(34) ' the quotation mark
        '
        ' first the nodes: define the node data structure
        tVNA.WriteLine ("*node data")
        tVNA.WriteLine ("ID index_year sex x_coord y_coord")
        '
        With tRstNode
            .MoveFirst
            Do While Not .EOF
                ' name = the ID of the person
                tStr = Trim(Str(!c_person_id)) + " "
                '
                ' indexyear = c_index_year INT
                If IsNull(!c_index_year) Then
                    tStr = tStr + "0 "
                Else
                    tStr = tStr + Trim(Str(!c_index_year)) + " "
                End If
                '
                ' sex = c_female > (F,M)
                If !c_sex = "F" Then
                    tStr = tStr + tQuote + "F" + tQuote + " "
                Else
                    tStr = tStr + tQuote + "M" + tQuote + " "
                End If
                '
                ' x_coord
                If IsNull(!x_coord) Then
                    tStr = tStr + "0 "
                Else
                    tStr = tStr + Trim(Str(!x_coord)) + " "
                End If
                '
                ' y_coord
                If IsNull(!y_coord) Then
                    tStr = tStr + "0 "
                Else
                    tStr = tStr + Trim(Str(!y_coord)) + " "
                End If
                '
                tVNA.WriteLine (tStr)
                .MoveNext
            Loop
        End With
        '

```

```

' now the node properties
'
' Note: ACTIVE removed as a property (MAF 2018/07/22)
'
tVNA.WriteLine ("*node properties")
tVNA.WriteLine ("ID shape size shortlabel")
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' ID = the ID of the person
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' shape = 2? / size = 1?
        tStr = tStr + "2 1 "
        '
        ' shortlabel (+ Active = TRUE removed)
        If IsNull(!c_name) Then
            tStr = tStr + "[Missing]"
        Else
            tStr = tStr + tQuote + !c_name + tQuote
        End If
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the edges: define the record structure
'
tStr = "from to " + tQuote + "EdgeWeight" + tQuote + " " + tQuote + "edgedesc" + tQuote
tVNA.WriteLine ("*tie data")
tVNA.WriteLine (tStr)
'
' For the moment, I am not combining parallel edges
'
With tRstEdge
    .MoveFirst
    Do While Not .EOF
        '
        ' From = str(c_person_id) for node1
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' to = str(c_assoc_id) for node2
        tStr = tStr + Trim(Str(!c_assoc_id)) + " 1 "
        '
        ' edgedesc
        '
        tStr = tStr + tQuote + Trim(!c_assoc_desc) + tQuote
        '
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the edges properties
'
tVNA.WriteLine ("*tie properties")
tVNA.WriteLine ("from to color size active")
'
With tRstEdge
    .MoveFirst
    Do While Not .EOF
        '
        ' from = str(c_person_id) for node1
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' to = str(c_node_id) for node2
        tStr = tStr + Trim(Str(!c_node_id)) + " 1 "
        '
        ' color = black (1), blue (2), green (3), yellow (4), orange (5)
        tStr = tStr + tColor(!c_edge_dist)
        '
        ' size = 1? active = TRUE
        tStr = tStr + "1 TRUE"
        '
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With
'

```

```

        tVNA.Close
    ,
    Set tRstNode = Nothing
    Set tRstEdge = Nothing
    Set tVNA = Nothing
    Set tFileSystem = Nothing
    Set tRstAssocType = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdUCINet_Click:
Exit Sub

Err_CmdUCINet_Click:
MsgBox Err.Description
Resume Exit_CmdUCINet_Click

End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim cmdSQL As ADODB.Command, tRecDeleted As Variant
    Dim tRstAssocCode As DAO.Recordset, tRstDummy As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    ,
    ' to clear the tables, briefly close and then delete records
    ,
    Set tRstAssocCode = ZZ_SCRATCH_ASSOC.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_AC", dbOpenDynaset)
    Set ZZ_SCRATCH_ASSOC.Form.Recordset = tRstDummy
    tRstAssocCode.Close
    ,
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ASSOC"
    cmdSQL.Execute tRecDeleted
    ,
    ' now reopen
    ,
    Set tRstAssocCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_ASSOC", dbOpenDynaset)
    Set ZZ_SCRATCH_ASSOC.Form.Recordset = tRstAssocCode
    ,
    Set tRstAssocCode = ZZ_SCRATCH_P_ASSOC.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_AP", dbOpenDynaset)
    Set ZZ_SCRATCH_P_ASSOC.Form.Recordset = tRstDummy
    tRstAssocCode.Close
    ,
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_ASSOC"
    cmdSQL.Execute tRecDeleted
    ,
    ' now reopen
    ,
    Set tRstAssocCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_ASSOC", dbOpenDynaset)
    Set ZZ_SCRATCH_P_ASSOC.Form.Recordset = tRstAssocCode

    ' first determine the language
    gLCID = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    If gLCID = 2052 Or gLCID = 3076 Then ' 2052 = PRC, 3076 = Hong Kong
        gDisplayLanguage = "S"
    ElseIf gLCID = 4100 Or gLCID = 1028 Then ' 4100 = Singapore, 1028 = Taiwan
        gDisplayLanguage = "T"
        Call changeDisplayLanguage
    Else
        gDisplayLanguage = "E"
        Call changeDisplayLanguage
    End If

    ' set the index year and dynasty default values

    gFromStr = "-200"
    gToStr = "1911"
    TxtFromYear.Value = -200
    TxtToYear.Value = 1911
    gFromDynasty = -1
    gToDynasty = -1

```

```

    gUseIndexYears = False
    gUseDynasties = False
End Sub

Private Sub CmdPajek_Click()
On Error GoTo Err_CmdPajek_Click
'
' This program will dump the results of the search to a .net file
'
' for the moment I'll just describe the format of the .gdf file
'
' *Vertices NUM
' ID label "box" ic [color] bc [color]
'     ID = str(c_person_id)
'     label = c_name_chn
'     color = red (1), orange (2), yellow (3), green (4), blue (5)
'
' *Edges
' node1 node2 1 1 "label"
'     node1 = str(c_person_id) for node1
'     node2 = str(c_node_id) for node2
'     color = red (1), orange (2), yellow (3), green (4), blue (5)
'     label = c_link_desc
'
' first see if there are any records to process
'
If ZZ_SCRATCH_ASSOC.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
'
If ZZ_SCRATCH_P_ASSOC.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstNodeList As DAO.Recordset
Dim tRstEdge As DAO.Recordset, tRstEdgeList As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tQuote As String, tFindStr As String, tPinyin As Boolean
Dim tColor(20) As String, tStrNode1 As String, tStrNode2 As String, tCodeStr As String, tRecDeleted As Long

' to write to a UTF-8 file, use the ADO stream object
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False
If CodeFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8.net"
ElseIf CodeFrame.Value = 2 Then
    tStream.Charset = "big5"
    tCodeStr = "BIG5.net"
ElseIf CodeFrame.Value = 3 Then
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030.net"
Else
    tStream.Charset = "ascii"
    tCodeStr = "ascii.net"
    tPinyin = True
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network_" + tCodeStr
    If .Show = -1 Then
        tFileName = ""
    
```

```

For Each tFN In .SelectedItems
    tFileName = tFN
    If Not tFileName = "" Then
        Exit For
    End If
Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdPajek_Click
Else
    ' make sure the file name has a net extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".net"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".net") Then
        tFileName = tFileName + ".net"
    End If
End If

'
' zap and open the scratch file
'

Dim cmdSQL As ADODB.Command
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
cmdSQL.Execute tRecDeleted

Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
tRstNodeList.Index = "c_ID"

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK_EDGE"
cmdSQL.Execute tRecDeleted

'
' set the Quote delimiter
'

tQuote = Chr(34)

' define the colors for the nodes
'

tColor(1) = "Black"
tColor(2) = "Blue"
tColor(3) = "Green"
tColor(4) = "Yellow"
tColor(5) = "Orange"
For ti = 6 To 20
    tColor(ti) = "Red"
Next

'
' process the two tables
'

Set tRstEdge = ZZ_SCRATCH_ASSOC.Form.Recordset
Set tRstNode = ZZ_SCRATCH_P_ASSOC.Form.Recordset
tC = Chr(44) ' the comma

' first the nodes: define the record structure
'

tRstNode.MoveFirst
tStr = "*Vertices " + Trim(Str(tRstNode.RecordCount))
tStream.WriteText tStr, adWriteLine

ti = 1
With tRstNode
    .MoveFirst
    Do While Not .EOF
        tStream.WriteText Trim(Str(ti)) + " "
        If IsNull(!c_name_chn) Then
            If !c_name = "" Or Left(!c_name, 12) = "***BAD DATA**" Then
                tStream.WriteText Chr(34)
                tStream.WriteText "Error-" + Trim(Str(!c_person_id))
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            Else
                tStream.WriteText Chr(34)
                tStream.WriteText !c_name
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            End If
        Else
            tStream.WriteText Chr(34)
            tStream.WriteText !c_name
            tStream.WriteText Chr(34)
            tStream.WriteText " box ", adWriteLine
        End If
        ti = ti + 1
    Loop
End With
tStream.WriteText "

```

```

        If !c_name_chn = "" Then
            If !c_name = "" Or Left(!c_name, 12) = "***BAD DATA**" Then
                tStream.WriteText Chr(34)
                tStream.WriteText "Error-" + Trim(Str(!c_person_id))
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            Else
                tStream.WriteText Chr(34)
                tStream.WriteText !c_name
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            End If
        Else
            tStream.WriteText Chr(34)
            If tPinyin Then
                tStream.WriteText !c_name
            Else
                tStream.WriteText !c_name_chn
            End If
            If ChkIDs.Value Then
                tStream.WriteText " (" + Trim(Str(!c_person_id)) + ")"
            End If
            tStream.WriteText Chr(34)
            tStream.WriteText " box ", adWriteLine
        End If
    End If
    '
    ' add the node to the list
    '
    tRstNodeList.AddNew
    tRstNodeList!c_v_num = Str(ti)
    tRstNodeList!c_ID = !c_person_id
    tRstNodeList.Update
    '
    .MoveNext
    ti = ti + 1
Loop
End With
tRstNodeList.Close
'
' now the edges: define the record structure
'
tStream.WriteText "*Edges", adWriteLine
'
' first aggregate the data to a temporary table (use the edge weight to count the number of records)
'
cmdSQL.CommandText = "SELECT ZZ_SCRATCH_ASSOC.c_person_id, ZZ_SCRATCH_ASSOC.c_assoc_id, " + _
    "Count(ZZ_SCRATCH_ASSOC.c_assoc_code) AS CountOfc_assoc_code, " + _
    "Sum(ZZ_SCRATCH_ASSOC.c_assoc_count) AS SumOfc_assoc_count INTO tmp_pajek_edge " + _
    "FROM ZZ_SCRATCH_ASSOC GROUP BY ZZ_SCRATCH_ASSOC.c_person_id, ZZ_SCRATCH_ASSOC.c_assoc_id"
cmdSQL.Execute tRecDeleted
'
' now join to the node IDs and copy to the edge table
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PAJEK_EDGE ( c_node_1, c_node_2, c_edge_weight, c_edge_c
ount, " + _
    "c_node_1_str, c_node_2_str ) " + _
    "SELECT Val([ZZ_SCRATCH_PAJEK].[c_v_num]) AS c_node_1, Val(ZZ_SCRATCH_PAJEK_1.c_v_num) AS c_node_
2, " + _
    "tmp_pajek_edge.CountOfc_assoc_code, tmp_pajek_edge.SumOfc_assoc_count, [ZZ_SCRATCH_PAJEK].[c_v_n
um], " + _
    "[ZZ_SCRATCH_PAJEK_1].[c_v_num] " + _
    "FROM ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 INNER JOIN (ZZ_SCRATCH_PAJEK INNER JOIN " + _
    "tmp_pajek_edge ON ZZ_SCRATCH_PAJEK.c_ID = tmp_pajek_edge.c_person_id) " + _
    "ON ZZ_SCRATCH_PAJEK_1.c_ID = tmp_pajek_edge.c_assoc_id"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "DROP TABLE tmp_pajek_edge"
cmdSQL.Execute tRecDeleted
'
' now fill in the edge description.
'
If tPinyin Then
    tQueryStr = "UPDATE ((ZZ_SCRATCH_PAJEK_EDGE INNER JOIN ZZ_SCRATCH_PAJEK " + _
        "ON ZZ_SCRATCH_PAJEK_EDGE.c_node_1_str = ZZ_SCRATCH_PAJEK.c_v_num) INNER JOIN " + _
        "ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 " + _
        "ON ZZ_SCRATCH_PAJEK_EDGE.c_node_2_str = ZZ_SCRATCH_PAJEK_1.c_v_num) INNER JOIN " + _
        "ZZ_SCRATCH_ASSOC ON (ZZ_SCRATCH_ASSOC.c_assoc_id = ZZ_SCRATCH_PAJEK_1.c_ID) " + _
        "AND (ZZ_SCRATCH_PAJEK.c_ID = ZZ_SCRATCH_ASSOC.c_person_id) " + _
        "SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = [ZZ_SCRATCH_ASSOC].[c_assoc_desc] " + _

```

```

        "WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_weight)=1))"
Else
    tQueryStr = "UPDATE ((ZZ_SCRATCH_PAJEK_EDGE INNER JOIN ZZ_SCRATCH_PAJEK " + _
        "ON ZZ_SCRATCH_PAJEK_EDGE.c_node_1_str = ZZ_SCRATCH_PAJEK.c_v_num) INNER JOIN " + _
        "ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 " + _
        "ON ZZ_SCRATCH_PAJEK_EDGE.c_node_2_str = ZZ_SCRATCH_PAJEK_1.c_v_num) INNER JOIN " + _
        "ZZ_SCRATCH_ASSOC ON (ZZ_SCRATCH_ASSOC.c_assoc_id = ZZ_SCRATCH_PAJEK_1.c_ID) " + _
        "AND (ZZ_SCRATCH_PAJEK.c_ID = ZZ_SCRATCH_ASSOC.c_person_id) " + _
        "SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = [ZZ_SCRATCH_ASSOC].[c_assoc_desc] + ' ' + " + _
        "[ZZ_SCRATCH_ASSOC].[c_assoc_desc_chn] WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_weight)=1))"
End If

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

tQueryStr = "UPDATE ZZ_SCRATCH_PAJEK_EDGE SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = " + _
    "'Parallel Edges merged' WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_weight)>1))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' open the table
'
Set tRstEdgeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK_EDGE", dbOpenDynaset)

With tRstEdgeList
    .MoveFirst
    Do While Not .EOF
        tStr = !c_node_1_str + " " + !c_node_2_str
        '
        ' now get the weight
        '
        If !c_edge_count < 6 Then
            tStr = tStr + " " + Trim(Str(!c_edge_count)) + " "
        Else
            tStr = tStr + " 5 "
        End If
        '
        ' now get the label
        '
        tStr = tStr + "1 " + tQuote
        If !c_edge_weight = 1 Then
            tStr = tStr + !c_edge_desc + tQuote + " "
        Else
            tStr = tStr + Trim(Str(!c_edge_count)) + " relations merged" + tQuote + " "
        End If

        tStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tStream.Close
Set tStream = Nothing
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tRstNodeList = Nothing
Set tRstEdgeList = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdPajek_Click:
Exit Sub

Err_CmdPajek_Click:
MsgBox Err.Description
Resume Exit_CmdPajek_Click

```

```

End Sub
Private Sub guess_write()
'
'   This program will dump the results of the search to a .gdf file
'
'   for the moment I'll just describe the format of the .gdf file
'
'   nodedef> name, color, label, labelvisible, style, pinyin VARCHAR(50), nodedist INT
'       name = str(c_person_id)
'       color = red (1), orange (2), yellow (3), green (4), blue (5)
'       label = c_name_chn
'       style = 4 (text inside a rectangle)
'       pinyin = c_name
'       nodedist = c_node_dist INT
'       indexyear = c_index_year INT
'       sex = c_female > (F,M)
'
'   edgedef> node1, node2, color, label, labelvisible, edge_desc VARCHAR(50)
'       node1 = str(c_person_id) for node1
'       node2 = str(c_node_id) for node2
'       color = red (1), orange (2), yellow (3), green (4), blue (5)
'       label = c_link_chn
'       edge_desc = c_link_desc
'
'   the central question is whether to do distance optimizations
'
'   first see if there are any records to process
'
If ZZ_SCRATCH_ASSOC.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGUESS_Click
End If
'
'   next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer
Dim tColor(50) As String, tMetricSum As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "default.gdf"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdGUESS_Click
        End If
        '
        '   now process the file (second true removed to make ASCII)
        '
        Set tFileSystem = CreateObject("Scripting.FileSystemObject")
        Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)

        ' define the colors for the nodes
        '
        tColor(1) = "white"
        tColor(2) = "blue"
        tColor(3) = "green"
        tColor(4) = "yellow"
        tColor(5) = "orange"
        For ti = 6 To 50
            tColor(ti) = "red"
        Next
    End With

```



```

'
' process the two tables
'
Set tRstEdge = ZZ_SCRATCH_ASSOC.Form.Recordset
Set tRstNode = ZZ_SCRATCH_P_ASSOC.Form.Recordset
tC = Chr(44) ' the comma
'
' first the nodes: define the record structure
tStr = "nodedef> name" + tC + "color" + tC + "label" + tC + "labelvisible"
tStr = tStr + tC + "style" + tC + "pinyin VARCHAR(50)"
tStr = tStr + tC + "indexyear INT" + tC + "sex VARCHAR(1)"
tGDF.WriteLine (tStr)
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        tStr = Trim(Str(!c_person_id)) + tC
        ' name = the ID of the person
        '
        tStr = tStr + tColor(1) + tC
        '
        If IsNull(!c_name_chn) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_name_chn + tC
        End If
        ' label
        tStr = tStr + "true" + tC + "4" + tC
        ' labelvisible = true, style = 4 (text inside a rectangle)
        If IsNull(!c_name) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_name + tC
        End If
        ' pinyin = c_name
        '
        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + tC
        End If
        ' indexyear = c_index_year INT
        If Not IsNull(!c_sex) Then
            tStr = tStr + !c_sex
        End If
        tGDF.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the edges: define the record structure
tStr = "edgedef> node1" + tC + "node2" + tC + "color" + tC + "label"
tGDF.WriteLine (tStr)

With tRstEdge
    .MoveFirst
    Do While Not .EOF
        tStr = Trim(Str(!c_person_id)) + tC
        ' node1 = str(c_person_id) for node1
        tStr = tStr + Trim(Str(!c_assoc_id)) + tC
        ' node2 = str(c_node_id) for node2
        '
        tStr = tStr + tColor(1) + tC
        ' color = white (1), blue (2), green (3), yellow (4), orange (5)
        '
        If IsNull(!c_assoc_desc) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_assoc_desc
        End If
        ' label = the association
        '
        tGDF.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
tGDF.Close
'
Set tRstNode = Nothing

```

```

        Set tRstEdge = Nothing
        Set tGDF = Nothing
        Set tFileSystem = Nothing
    Else
        'The user pressed Cancel.
    End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGUESS_Click:
Exit Sub

Err_CmdGUESS_Click:
MsgBox Err.Description
Resume Exit_CmdGUESS_Click

End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

    If gDisplayLanguage = "T" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "T"
    End If

    Call changeDisplayLanguage

Exit_CmdFanti_Click:
Exit Sub

Err_CmdFanti_Click:
MsgBox Err.Description
Resume Exit_CmdFanti_Click

End Sub

Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

    If gDisplayLanguage = "S" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "S"
    End If

    Call changeDisplayLanguage

Exit_CmdJianti_Click:
Exit Sub

Err_CmdJianti_Click:
MsgBox Err.Description
Resume Exit_CmdJianti_Click

End Sub

Public Sub changeDisplayLanguage()
    Dim tLabellLanguage(3, 37) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 37 And Not .EOF
            If !c_form = "LAA" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                End If
            End If
            ti = ti + 1
        Loop
    End With
End Sub

```

```

        Exit Do
    End If
    tLabelLanguage(1, ti) = !c_english
    tLabelLanguage(2, ti) = !c_fanti
    tLabelLanguage(3, ti) = !c_jianti
    ti = ti + 1
End If
.MoveNext
Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If gLabelsOK Then
    If gDisplayLanguage = "E" Then
        tLang = 1
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '
    Me.LblFrom.Caption = tLabelLanguage(tLang, 1)
    Me.LblTo.Caption = tLabelLanguage(tLang, 2)
    Me.LblType.Caption = tLabelLanguage(tLang, 3)
    Me.CmdPickAssoc.Caption = tLabelLanguage(tLang, 4)
    Me.CmdQuery.Caption = tLabelLanguage(tLang, 5)
    Me.CmdGIS.Caption = tLabelLanguage(tLang, 6)
    Me.CmdPajek.Caption = tLabelLanguage(tLang, 7)
    Me.CmdFanti.Caption = tLabelLanguage(tLang, 8)
    Me.CmdJianti.Caption = tLabelLanguage(tLang, 9)
    Me.PageAssoc.Caption = tLabelLanguage(tLang, 10)
    Me.PagePeople.Caption = tLabelLanguage(tLang, 11)
    'Me.LblChkIndexYears.Caption = tLabelLanguage(tLang, 12)
    Me.CmdSelectPlace.Caption = tLabelLanguage(tLang, 13)
    Me.CmdImportPlaces.Caption = tLabelLanguage(tLang, 14)
    Me.CmdAllPlaces.Caption = tLabelLanguage(tLang, 15)
    Me.LblIDs.Caption = tLabelLanguage(tLang, 16)
    Me.LblDisplay.Caption = tLabelLanguage(tLang, 17)
    Me.CmdHelp.Caption = tLabelLanguage(tLang, 18)
    Me.LblXYRef.Caption = tLabelLanguage(tLang, 19)
    Me.LblNarrow.Caption = tLabelLanguage(tLang, 20)
    Me.LblBroad.Caption = tLabelLanguage(tLang, 21)
    Me.CmdStoreID.Caption = tLabelLanguage(tLang, 22)
    Me.CmdUCINet.Caption = tLabelLanguage(tLang, 23)
    Me.LblChkSubUnits.Caption = tLabelLanguage(tLang, 24)
    Me.CmdGephi.Caption = tLabelLanguage(tLang, 25)
    Me.LblDynasties.Caption = tLabelLanguage(tLang, 26)
    Me.CmdFromDynasty.Caption = tLabelLanguage(tLang, 27)
    Me.CmdToDynasty.Caption = tLabelLanguage(tLang, 28)
    Me.CmdAllDynasties.Caption = tLabelLanguage(tLang, 29)
    Me.LblIndexYears.Caption = tLabelLanguage(tLang, 30)
    Me.LblOptNoDates.Caption = tLabelLanguage(tLang, 31)
    Me.LblOptIndexYears.Caption = tLabelLanguage(tLang, 32)
    Me.LblOptDynasties.Caption = tLabelLanguage(tLang, 33)
    Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 34)
    Me.CmdImportAssociations.Caption = tLabelLanguage(tLang, 35)
    Me.CmdSaveAssociations.Caption = tLabelLanguage(tLang, 36)
End If

End Sub

Private Sub CmdSelectPlace_Click()
On Error GoTo Err_CmdSelectPlace_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strADDR As String
Dim cmdSQL As ADODB.Command

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

TxtAddrID.Visible = True
TxtAddrID.SetFocus
strADDR = TxtAddrID.Text

stDocName = "frmPickAddresses_multi"

```

```
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR
```

```
If CurrentProject.AllForms("frmPickAddresses_multi").IsLoaded Then
    Dim tAddrID As Long, tRstAddr As DAO.Recordset
    Dim strADDR_CHN As String, strADDR_PY As String
```

```
    gUseADDRID = True
    CmdAllPlaces.Enabled = True
    ChkXYRef.Enabled = True
    ChkSubUnits.Enabled = True
    FrameXY.Enabled = True
```

```
    Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Visible = True
    Forms!frmPickAddresses_multi.Form!TxtAddrFilter.SetFocus
    If Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Value Then
        ,
        TxtAddrID.Value = 0
        strADDR_PY = Forms!frmPickAddresses_multi.Form!TxtFilterPY
        strADDR_CHN = Forms!frmPickAddresses_multi.Form!TxtFilterChn
```

```
        If strADDR_CHN = "" Then
            TxtPlaceChn.Value = "[[Filter]]"
            TxtPlace.Value = "[" + strADDR_PY + "]"
        Else
            TxtPlaceChn.Value = "[" + strADDR_CHN + "]"
            TxtPlace.Value = "[[Filter]]"
        End If
```

```
    Else
        Forms!frmPickAddresses_multi.Form!TxtSelectCount.Visible = True
        Forms!frmPickAddresses_multi.Form!TxtSelectCount.SetFocus
        If Forms!frmPickAddresses_multi.Form!TxtSelectCount.Value > 1 Then
            TxtPlaceChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
            TxtPlace.Value = "[[Multi-Select]]"
            TxtAddrID.Value = 0
```

```
        Else
            ' only one record in ZZ_ADDRESSES: get its field values
            ,
```

```
            Set tRstAddr = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
            tRstAddr.MoveFirst
            'MsgBox "Checking zz_addresses: no records"
            TxtAddrID.Value = tRstAddr!c_addr_id
            TxtPlaceChn.Value = tRstAddr!c_name_chn
            TxtPlace.Value = tRstAddr!c_name
            tRstAddr.Close
            Set tRstAddr = Nothing
```

```
        End If
```

```
    End If
```

```
    ,
    ' now copy the records
    ,
```

```
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_LIST"
    cmdSQL.Execute tRecDeleted
```

```
    cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT " + _
        "ZZ_ADDRESSES.c_addr_id FROM ZZ_ADDRESSES"
    cmdSQL.Execute tRecDeleted
```

```
End If
```

```
DoCmd.Close acForm, stDocName
CmdSelectPlace.SetFocus
TxtAddrID.Visible = False
```

```
Exit_CmdSelectPlace_Click:
```

```
Exit Sub
```

```
Err_CmdSelectPlace_Click:
```

```
MsgBox Err.Description
Resume Exit_CmdSelectPlace_Click
```

```
End Sub
```

```
Private Sub CmdAllPlaces_Click()
On Error GoTo Err_CmdAllPlaces_Click
```

```
    TxtAddrID.Value = -1
```

```
    TxtPlaceChn.Value = ""
    TxtPlace.Value = ""
    gUseADDRID = False
    ChkXYRef.Enabled = False
    ChkSubUnits.Enabled = False
```

```

        FrameXY.Enabled = False
Exit_CmdAllPlaces_Click:
    Exit Sub

Err_CmdAllPlaces_Click:
    MsgBox Err.Description
    Resume Exit_CmdAllPlaces_Click

End Sub

Private Sub CmdImportPlaces_Click()
    On Error GoTo Err_CmdImportPlaces_Click

    Dim stDocName As String, tRstAddresses As DAO.Recordset, tRstImportPlaces As DAO.Recordset
    Dim stLinkCriteria As String
    Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String, tLen As Integer, tQuit As Boolean

    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tFileSystem, tList

    tQuit = False
    '
    If Not tQuit Then
        ' open the list

        Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

        'Use a With...End With block to reference the FileDialog object.
        With dlgSaveAs
            .InitialFileName = ""
            If .Show = -1 Then
                '
                tFileName = ""
                For Each tFN In .SelectedItems
                    tFileName = tFN
                    If Not tFileName = "" Then
                        Exit For
                    End If
                Next
                If tFileName = "" Then
                    MsgBox "Bad file Name."
                    GoTo Exit_CmdImportPlaces_Click
                End If
            End If
        End With

        ' Clear the address table now that we are ready to go
        '
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_LIST"
        cmdSQL.Execute tRecDeleted

        cmdSQL.CommandText = "Delete * from InputErrorList"
        cmdSQL.Execute tRecDeleted

        cmdSQL.CommandText = "Delete * from TempImportList"
        cmdSQL.Execute tRecDeleted

        DoCmd.TransferText acImportDelim, "ImportPlaceList_Space", "TempImportList", tFileName, 0
        ' TransferType=acImportDelim
        ' SpecificationName = "TempImportList" (apparently it is saved in the database itself)
        ' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
        ' HasFieldNames = False (0)
        '
        ' copy the bad IDs
        '
        tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
            "FROM ADDR_CODES RIGHT JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID " + _
            "WHERE ((ADDR_CODES.c_addr_id) Is Null))"

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted

        If tRecDeleted > 0 Then
            MsgBox "Some ID were not successfully imported: please look at InputErrorList."
        End If
    End If
End Sub

```

```

End If
'
'   copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT TempImportList.ImportID " + _
          "FROM ADDR_CODES INNER JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    Me.TxtPlace.Value = "[Imported List]"
    Me.TxtPlaceChn.Value = "[Imported List]"
    gUseADDRID = False
    ChkXYRef.Enabled = True
    ChkSubUnits.Enabled = True
    FrameXY.Enabled = True
End If

Set cmdSQL = Nothing
Set tFileSystem = Nothing
End If

Exit_CmdImportPlaces_Click:
Exit Sub

Err_CmdImportPlaces_Click:
MsgBox Err.Description
Resume Exit_CmdImportPlaces_Click

End Sub

Private Sub FrameFilterYears_Click()
'
'   the simplest approach is to turn it all off and then turn on the appropriate objects
'
'   disable all
Me.CmdFromDynasty.Enabled = False
Me.CmdToDynasty.Enabled = False
Me.CmdAllDynasties.Enabled = False
Me.TxtFromDynasty.Enabled = False
Me.TxtFromDynastyPY.Enabled = False
Me.TxtToDynasty.Enabled = False
Me.TxtToDynastyPY.Enabled = False
Me.TxtFromDynasty.Locked = False
Me.TxtFromDynastyPY.Locked = False
Me.TxtToDynasty.Locked = False
Me.TxtToDynastyPY.Locked = False

Me.TxtFromYear.Enabled = False
Me.TxtToYear.Enabled = False

gUseIndexYears = False
gUseDynasties = False

If FrameFilterYears.Value = 2 Then
    ' enable index years
    Me.TxtFromYear.Enabled = True
    Me.TxtToYear.Enabled = True
    gUseIndexYears = True

ElseIf FrameFilterYears.Value = 3 Then
    ' enable dynasties
    Me.CmdFromDynasty.Enabled = True
    Me.CmdToDynasty.Enabled = True
    Me.CmdAllDynasties.Enabled = True
    Me.TxtFromDynasty.Enabled = True
    Me.TxtFromDynastyPY.Enabled = True
    Me.TxtToDynasty.Enabled = True
    Me.TxtToDynastyPY.Enabled = True
    Me.TxtFromDynasty.Locked = True
    Me.TxtFromDynastyPY.Locked = True
    Me.TxtToDynasty.Locked = True
    Me.TxtToDynastyPY.Locked = True
    gUseDynasties = True

```

```

End If

End Sub

Private Sub TxtFromYear_LostFocus()
    gFromStr = Trim(TxtFromYear.Text)
End Sub

Private Sub TxtToYear_LostFocus()
    gToStr = Trim(TxtToYear.Text)
End Sub

Private Sub CmdHelp_Click()
    Dim tStrPDF As String

    tStrPDF = Application.CurrentProject.Path + "\HelpFiles\HelpFile_LookAtAssociations.pdf"

    'MsgBox tStrPDF

    Application.FollowHyperlink tStrPDF, , True

End Sub

Private Sub writeKML()
On Error GoTo Err_writeKML
    '
    ' This program will dump the results to a .gis file
    '
    If ZZ_SCRATCH_P_ASSOC.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_writeKML
    End If
    '
    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream

    tPinyin = False
    If GISFrame.Value = 1 Then
        tStream.Charset = "utf-8"
        tCodeStr = "UTF8"
    ElseIf GISFrame.Value = 2 Then
        tStream.Charset = "ascii"
        tCodeStr = "ASCII"
        tPinyin = True
    Else
        tStream.Charset = "gb18030"
        tCodeStr = "GB18030"
    End If
    tStream.Mode = adModeReadWrite
    tStream.Type = adTypeText
    tStream.Open
    '
    ' next get a file
    '

    Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".kml"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_writeKML
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".kml"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
                tFileName = tFileName + ".kml"
            End If
        End If
        '
        ' write the file
    End Sub

```

```

Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count
'
' process the table
'
Set tRstNode = ZZ_SCRATCH_P_ASSOC.Form.Recordset
tC = Chr(9) ' the tab
tDQ = Chr(34) ' the double quotation mark
'
' write the header
'
tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
tStream.WriteText "<Document>", adWriteLine
tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
tStream.WriteText tC + "<open>1</open>", adWriteLine
tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine
tStream.WriteText tC + "<Style id=" + tDQ + "assoc-balloon-template" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
tStream.WriteText tC + tC + tC + "<text>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "<![CDATA[" + tC + tC + tC + tC + "ID: ${AssocGIS/PersonID} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Name Chn: ${AssocGIS/NameHZ} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Index Year: ${AssocGIS/IndexYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Sex: ${AssocGIS/Sex} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Address: ${AssocGIS/AddrName} ${AssocGIS/AddrHZ} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "XY Count: ${AssocGIS/XYCount} <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine
tStream.WriteText tC + "<!-- Declare the type " + tDQ + "AssocGIS" + tDQ + " with 6 fields -->", adWriteLine
tStream.WriteText tC + "<Schema name=" + tDQ + "AssocGIS" + tDQ + " id=" + tDQ + "AssocGISId" + tDQ + ">"
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ + ">"
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person ID]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "NameHZ" + tDQ + ">"
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Name Chn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "Sex" + tDQ + ">"
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Sex]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + tDQ + ">"
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrHZ" + tDQ + ">"
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" + tDQ + ">"
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ + ">"
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs, even where there should not be any
        ' write the point header
        tStream.WriteText tC + "<Placemark>", adWriteLine

        If IsNull(!c_name) Then
            tStr = "[Bad Data]"
        Else
            tStr = !c_name
        End If
    End Do

```



```

End If
tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

tStream.WriteText tC + tC + "<styleUrl>#assoc-balloon-template</styleUrl>", adWriteLine
'
'   First Year as time stamp
'
If IsNull(!c_index_year) Then
    tStr = "N/A"
Else
    tStr = Str(!c_index_year)
End If
tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
'
tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#AssocGISId" + tDQ + ">", adWr
iteLine
'
'   person ID
'
tStr = Str(!c_person_id)
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
'   Person Name Chn
'
If IsNull(!c_name_chn) Then
    tStr = tStr + "[Bad Data]"
Else
    If Trim(!c_name_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_name_chn
    End If
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NameHZ" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
'
'   Index Year
'
If IsNull(!c_index_year) Then
    tStr = "N/A"
Else
    tStr = Str(!c_index_year)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
'   sex
'
If IsNull(!c_sex) Then
    tStr = "[?]"
Else
    tStr = !c_sex
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "Sex" + tDQ + ">" + tStr + "</S
impleData>", adWriteLine
'
'   Address Name
'
If IsNull(!c_addr_name) Then
    tStr = "[?]"
ElseIf Trim(!c_addr_name) = "" Then
    tStr = "[?]"
Else
    tStr = !c_addr_name
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
'   Address Name Chinese
'
If IsNull(!c_addr_chn) Then
    tStr = "[?]"
ElseIf Trim(!c_addr_chn) = "" Then
    tStr = "[?]"
Else
    tStr = !c_addr_chn
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrHZ" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
'

```

```

        ' XY Count
        '
        If IsNull(!xy_count) Then
            tStr = "0"
        Else
            tStr = Str(!xy_count)
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
        '
        tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
        tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + "<Point>", adWriteLine
        '
        ' coordinates
        If IsNull(!x_coord) Then
            tStr = "0"
        Else
            tStr = Str(!x_coord)
        End If

        If IsNull(!y_coord) Then
            tStr = tStr + ",0"
        Else
            tStr = tStr + "," + Str(!y_coord)
        End If
        tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
        '
        ' footer
        '
        tStream.WriteText tC + tC + "</Point>", adWriteLine
        tStream.WriteText tC + "</Placemark>", adWriteLine
        .MoveNext
    Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_writeKML:
Exit Sub

Err_writeKML:
MsgBox Err.Description
Resume Exit_writeKML

End Sub

Private Sub CmdStoreID_Click()
    Dim cmdSQL As ADODB.Command, tRecCount As Variant

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    If DCount("?", "ZZ_STORE_PERSON_ID") > 0 Then
        ' Display message.
        If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) =
vbNo Then
            Exit Sub
        Else
            cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
            cmdSQL.Execute tRecCount
        End If
    End If

```

End If

tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID (c_personid) SELECT DISTINCT ZZ_SCRATCH_P_ASSOC.c_person_id FROM ZZ_SCRATCH_P_ASSOC"

cmdSQL.CommandText = tStrQuery

cmdSQL.Execute tRecCount

MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."

'

' update storage source

cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='Associations' WHERE PersonIDSource.LineNum =1"

cmdSQL.Execute tRecCount

End Sub

```
Option Compare Database
```

```
Dim gRstPeople As DAO.Recordset, gDisplayLanguage As String, gLabelsOK As Boolean
```

```
Public gUseADDRID As Boolean, gUseIndexYears As Boolean, gUseEntryYears As Boolean, gUseDynasties As Boolean
```

```
Public gFromDynasty As Integer, gToDynasty As Integer
```

```
Public gFromDynastyBegin As Integer, gFromDynastyEnd As Integer, gToDynastyBegin As Integer, gToDynastyEnd As Integer
```

```
Private Sub ChkUseYears_Click()
```

```
    If ChkUseYears.Value = True Then
```

```
        ' ChkUseYears.Value = False
```

```
        TxtFromYear.Enabled = True
```

```
        TxtToYear.Enabled = True
```

```
        FrameYears.Enabled = True
```

```
    Else
```

```
        ' ChkUseYears.Value = False
```

```
        TxtFromYear.Enabled = False
```

```
        TxtToYear.Enabled = False
```

```
        FrameYears.Enabled = False
```

```
    End If
```

```
End Sub
```

```
Private Sub CmdAllDynasties_Click()
```

```
    gFromDynasty = -2
```

```
    gToDynasty = -2
```

```
    TxtFromDynasty.Value = ""
```

```
    TxtFromDynastyPY.Value = "All"
```

```
    TxtToDynasty.Value = ""
```

```
    TxtToDynastyPY.Value = "All"
```

```
End Sub
```

```
Private Sub CmdFromDynasty_Click()
```

```
    Dim stDocName As String
```

```
    Dim stLinkCriteria As String
```

```
    Dim strFromDynasty As String
```

```
    If gFromDynasty < 0 Then
```

```
        strFromDynasty = ""
```

```
    Else
```

```
        strFromDynasty = Str(gFromDynasty)
```

```
    End If
```

```
    stDocName = "frmPickDynasty"
```

```
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty
```

```
    If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
```

```
        gFromDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
```

```
        gFromDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
```

```
        gFromDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
```

```
        '
```

```
        ' check to see if we have a problem and reject selection
```

```
        '
```

```
    If gToDynasty > -1 Then
```

```
        If gFromDynastyBegin > gToDynastyEnd Then
```

```
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
```

```
            gFromDynasty = -1
```

```
            TxtFromDynasty.Value = ""
```

```
            TxtFromDynastyPY.Value = ""
```

```
        End If
```

```
    End If
```

```
    '
```

```
    ' value is OK
```

```
    '
```

```
    If gFromDynasty > -1 Then
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
```

```
        TxtFromDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
```

```
        TxtFromDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
```

```
    End If
```

```
    DoCmd.Close acForm, stDocName
```

Form_LookAtEntry - 2

```
'
' reset ToDynasty if necessary (-2 = all dynasties)
'
If gToDynasty = -2 Then
    gToDynasty = -1
    TxtToDynasty.Value = ""
    TxtToDynastyPY.Value = ""
End If
'
End If

End Sub

Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call writeKML
    Exit Sub
End If
'
' This program will dump the results to a .gis file
'
If Entry_Address_Query.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGIS_Click
End If
'
' I leave this code here as the frame if I need to find a way to define the coding of the export file

Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If
'
' next get a file
'

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tC As String
Dim tRstGIS As DAO.Recordset
Dim tStr As String, tStrFileType As String

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)
'
' .tab for default
'

tStrFileType = ".tab"
dlgSaveAs.InitialFileName = "entry_gis_" + tCodeStr + tStrFileType
If dlgSaveAs.Show = -1 Then
'
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdGIS_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + tStrFileType
        ElseIf Not (LCase(Right(tFileName, 4)) = tStrFileType) Then
            tFileName = tFileName + tStrFileType
        End If
    End If
'
    tStrQuery = "SELECT ZZ_SCRATCH_ENTRY.c_personid, ZZ_SCRATCH_ENTRY.c_name, ZZ_SCRATCH_ENTRY.c_name_chn, "
```

```

+ -
    "ZZ_SCRATCH_ENTRY.c_index_year, ZZ_SCRATCH_ENTRY.c_entry_desc, ZZ_SCRATCH_ENTRY.c_entry_chn, " +
    "ZZ_SCRATCH_ENTRY.c_exam_rank, ZZ_SCRATCH_ENTRY.c_kin_name, ZZ_SCRATCH_ENTRY.c_kin_chn, ZZ_SCRATCH_ENTRY.c_kin_desc, " +
    "ZZ_SCRATCH_ENTRY.c_addr_name, ZZ_SCRATCH_ENTRY.c_addr_chn, " +
    str(ZZ_SCRATCH_ENTRY.x_coord) as X, str(ZZ_SCRATCH_ENTRY.y_coord) as Y, "
    str(ZZ_SCRATCH_ENTRY.x_coord) + ',' + str(ZZ_SCRATCH_ENTRY.y_coord) AS XY, " +
    "ZZ_SCRATCH_ENTRY.xy_count, ZZ_SCRATCH_ENTRY.c_entry_addr_name, ZZ_SCRATCH_ENTRY.c_entry_addr_chn, "
+ -
    "ZZ_SCRATCH_ENTRY.c_entry_xcoord, ZZ_SCRATCH_ENTRY.c_entry_ycoord, ZZ_SCRATCH_ENTRY.c_entry_xy_count
" + -
    "FROM ZZ_SCRATCH_ENTRY "
    "WHERE ZZ_SCRATCH_ENTRY.c_addr_id > 0"
'DoCmd.TransferText acExportDelim, "ENTRY_GIS_QUERY", tFileName, True

tC = Chr(9) ' tab
'
' we have a file name: now open the stream for writing
'
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText

tStream.Open
'
' process the table
'
Set tRstGIS = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
'
' write the header
'
tStr = "PersonID" + tC + "Name" + tC + "NameChn" + tC + "IndexYear" + tC + "EntryDesc" + tC +
    "EntryDescChn" + tC + "ExamRank" + tC + "KinName" + tC + "KinNameChn" + tC + "KinshipRel" + tC +
    "AddrName" + tC + "AddrNameChn" + tC + "x_coord" + tC + "y_coord" + tC + "XY" + tC + "xy_count" + tC
+ -
    "EntryAddrName" + tC + "EntryAddrChn" + tC + "Entry_xcoord" + tC + "Entry_ycoord" + tC + "Entry_xy_co
unt"

tStream.WriteText tStr, adWriteLine
'
With tRstGIS
.MoveFirst
Do While Not .EOF
    tStr = ""
    '
    If IsNull(!c_personid) Then
        tStr = "[Person ID Missing]"
    Else
        tStr = Str(!c_personid)
    End If
    '
    If IsNull(!c_name) Then
        tStr = tStr + tC + "[Name Missing]"
    Else
        tStr = tStr + tC + !c_name
    End If
    '
    If IsNull(!c_name_chn) Then
        tStr = tStr + tC + "[Name Missing]"
    Else
        tStr = tStr + tC + !c_name_chn
    End If
    '
    If IsNull(!c_index_year) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + Str(!c_index_year)
    End If
    '
    If IsNull(!c_entry_desc) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + !c_entry_desc
    End If
    '
    If IsNull(!c_entry_chn) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + !c_entry_chn
    EndIf
    '

```

```

If IsNull(!c_exam_rank) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + !c_exam_rank
End If
,
If IsNull(!c_kin_name) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + !c_kin_name
End If
,
If IsNull(!c_kin_chn) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + !c_kin_chn
End If
,
If IsNull(!c_kin_desc) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + !c_kin_desc
End If
,
If IsNull(!c_addr_name) Then
    tStr = tStr + tC + "[Addr Name Missing]"
Else
    tStr = tStr + tC + !c_addr_name
End If
,
If IsNull(!c_addr_chn) Then
    tStr = tStr + tC + "[Addr Chn Missing]"
Else
    tStr = tStr + tC + !c_addr_chn
End If
,
If IsNull(!x_coord) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + CStr(!x_coord)
End If
,
If IsNull(!y_coord) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + CStr(!y_coord)
End If
,
If IsNull(!x_coord) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + CStr(!x_coord) + "," + CStr(!y_coord)
End If
,
If IsNull(!xy_count) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + Str(!xy_count)
End If
,
If IsNull(!c_entry_addr_name) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + !c_entry_addr_name
End If
,
If IsNull(!c_entry_addr_chn) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + !c_entry_addr_chn
End If
,
If IsNull(!c_entry_xcoord) Then
    tStr = tStr + tC + "[ ]"
Else
    tStr = tStr + tC + CStr(!c_entry_xcoord)
End If
,
If IsNull(!c_entry_ycoord) Then
    tStr = tStr + tC + "[ ]"

```

```

        Else
            tStr = tStr + tC + CStr(!c_entry_ycoord)
        End If
    ,
    If IsNull(!c_entry_xy_count) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + Str(!c_entry_xy_count)
    End If
    ,
    If Not (tStr = "") Then
        tStream.WriteText tStr, adWriteLine
    End If
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
,
tStream.Close
Else
    'The user pressed Cancel.

End If
,
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGIS_Click:
Exit Sub

Err_CmdGIS_Click:
MsgBox Err.Description
Resume Exit_CmdGIS_Click
End Sub

Private Sub CmdHelp_Click()
    Dim tStrPDF As String

    tStrPDF = Application.CurrentProject.Path + "\HelpFiles\HelpFile_LookAtEntry.pdf"

    'MsgBox tStrPDF

    Application.FollowHyperlink tStrPDF, , True

End Sub

Private Sub CmdImportEntryCodes_Click()
On Error GoTo Err_CmdImportEntryCodess_Click

    Dim stDocName As String, tRstEntryCodes As DAO.Recordset
    Dim stLinkCriteria As String, tRstImportEntryCodess As DAO.Recordset
    Dim tString As String, tEntryCode As Long, ti As Integer, tStrID As String, tQuit As Boolean
    Dim tLen As Integer, cmdSQL As ADODB.Command

    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tFileSystem, tList

    ' first see if we already have a list

tQuit = False
,
If Not tQuit Then
    ' open the list

    Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

    'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
        .InitialFileName = ""
        If .Show = -1 Then
            ,
            tFileName = ""
            ForEach tFN In .SelectedItems
                tFileName = tFN
            End ForEach
        End If
    End With
End With

```



```

        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdImportEntryCodess_Click
    End If
End If
End With
'
' Clear the address table now that we are ready to go
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ENTRY_CODE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "EntryListImport Specification", "TempImportList", tFileName, 0
'    TransferType=acImportDelim
'    SpecificationName = "TempImportList" (apparently it is saved in the database itself)
'    TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
'    HasFieldNames = False (0)
'
' copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " +
    "FROM ENTRY_CODES RIGHT JOIN TempImportList ON ENTRY_CODES.c_entry_code = TempImportList.ImportID " +
    "WHERE ((ENTRY_CODES.c_entry_code) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If
'
' copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_ENTRY_CODE ( c_entry_code ) SELECT DISTINCT TempImportList.ImportID " +
    "FROM ENTRY_CODES INNER JOIN TempImportList ON ENTRY_CODES.c_entry_code = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

Me.TxtTypeDesc.Value = ""
Me.TxtTypeChn.Value = ""
If tRecDeleted > 0 Then
    Me.TxtEntryDesc.Value = "[Imported List]"
    Me.TxtEntryChn.Value = "[Imported List]"
    Me.CmdQuery.Enabled = True
    Me.CmdSaveEntryCodes.Enabled = True
Else
    Me.TxtEntryDesc.Value = ""
    Me.TxtEntryChn.Value = ""
    Me.CmdQuery.Enabled = False
    Me.CmdSaveEntryCodes.Enabled = False
End If

Set cmdSQL = Nothing
End If

Exit_CmdImportEntryCodess_Click:
Exit Sub

Err_CmdImportEntryCodess_Click:
MsgBox Err.Description
Resume Exit_CmdImportEntryCodess_Click

End Sub

```

```

Private Sub CmdNeo4j_Click()
On Error GoTo Err_CmdNeo4j_Click
'
'   This program will dump the results of the search to five CSV files
'
'   for the moment I'll just describe the format of the CSV file
'   Note: Neo4j seems to treat all fields as strings, so there is no need to explicitly mark strings
'
'   1. People.CSV
'       nameID = c_person_id
'       nameHZ = c_name_chn
'       namePY = c_name
'       indexyear = c_index_year
'       personDynasty = c_dynasty
'       sex = c_female > (F,M)
'
'   2. Places.CSV
'       placeID = c_addr_id
'       placeHZ = c_addr_chn
'       placePY = c_addr_name
'       placeX = x_coord
'       placeY = y_coord
'
'   3. PeoplePlaces.CSV
'       nameID
'       placeID
'       personPlaceRelation
'
'   4. PeopleEntry.CSV
'       nameID = str(c_person_id)
'       entryID = str(c_node_id)
'       entryPlaceID
'       kinID
'       kinRelID
'       AssocPersonID
'       AssocRelID
'       SocialInstID
'       SocialInstNameID
'       EntryYear
'       EntryDynasty
'
'   5. PeoplePlaceCodes
'
'   6. EntryCodes.CSV
'       entryID = str(c_entry_id)
'       entryDesc = c_entry_desc
'
'   7. KinCodes.CSV
'       kinCode
'       kinDesc
'
'   8. AssocCodes.CSV
'
'   9. Institution codes
'
'   first see if there are any records to process
'
If Entry_Address_Query.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdNeo4j_Click
End If
'
'   warn the user that a lot of files will be created
'
MsgBox "Neo4j requires that from 6 to 9 files be created."
'
'   allocate the file variables
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
'
'   next get the People file
'
Dim tRstPeople As DAO.Recordset, tRstEntry As DAO.Recordset, tRstEntryCodes As DAO.Recordset, tRstPlace As DAO.Recordset
Dim tRstPeopleEntry As DAO.Recordset, tRstPeoplePlace As DAO.Recordset, tStr As String, tC As String, ti As Integer, tUseList As Boolean
Dim tQueryStr As String, tPersonID As Long
Dim gStream As ADODB.Stream, tCodeStr As String

```

```

'
' the optional recordsets
'
Dim tRstKinCodes As DAO.Recordset, tRstAssocCodes As DAO.Recordset, tRstInstitutions As DAO.Recordset
'
'Dim tFileSystem, tGDF
'
' set up the stream to write to
Set gStream = New ADODB.Stream
'
' for the moment, set the character set to UTF-8
gStream.Charset = "utf-8"
tCodeStr = "UTF8"
'If CodeFrame.Value = 1 Then
'    gStream.Charset = "utf-8"
'    tCodeStr = "UTF8"
'ElseIf CodeFrame.Value = 2 Then
'    gStream.Charset = "big5"
'    tCodeStr = "BIG5"
'ElseIf CodeFrame.Value = 3 Then
'    gStream.Charset = "gb2312"
'    tCodeStr = "GB2312"
'Else
'    gStream.Charset = "ascii"
'    tCodeStr = "ascii"
'End If
'
tC = Chr(44) ' the comma
'
' prepare the temp tables for the people, place, peoplePlace and entry data

Dim cmdSQL As ADODB.Command
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' start with people: there will be three sources for people.
' the people who entered
' the kin who might have had a role in entry
' the associates who might have played a role
' the strategy is to just dump all such IDs to a scratch table and append (distinct) to a table for export
' ZZ_SCRATCH_P_TEXT is a convenient table for collecting IDs (no primary key)

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted
'
' get the people IDs
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_personid FROM ZZ_SCRATCH_ENTRY"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_kin_id FROM ZZ_SCRATCH_ENTRY " + _
"WHERE (((ZZ_SCRATCH_ENTRY.c_kin_id)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_assoc_id FROM ZZ_SCRATCH_ENTRY " + _
"WHERE (((ZZ_SCRATCH_ENTRY.c_assoc_id)>0));"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now clear ZZ_SCRATCH_PEOPLE and copy the records
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_dynasty, c_dynasty_chn, c_female, c_addr_id, " + _
"c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MAIN.c_index_year, " + _

```

```

        "ZZZ_BIOG_MAIN.c_dynasty, ZZZ_BIOG_MAIN.c_dynasty_chn, ZZZ_BIOG_MAIN.c_female, ZZZ_BIOG_MAIN.
c_index_addr_id, " +
        "ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_MAIN.c_index_addr_chn, ZZZ_BIOG_MAIN.c_index_addr_
type_code, ZZZ_BIOG_MAIN.c_index_addr_type_desc, " +
        "ZZZ_BIOG_MAIN.c_index_addr_type_chn, ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord " +
        "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_BIOG_MAIN
.c_personid"
    ,
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    ,
    Set tRstPeopleEntry = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
    Set tRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)

' Open the People file

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    ' now process the file (second true removed to make ASCII)
    ,
    ' we have a file name: now open the stream for writing

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    tRstPeople.MoveLast
    ,
    ' process the four tables
    ,
    ' first the nodes: define the record structure
    ,
    ' if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a pin
y in field
    If tCodeStr = "ascii" Then
        tStr = "nameID" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
    Else
        tStr = "nameID" + tC + "nameHZ" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
    End If
    gStream.WriteText tStr, adWriteLine
    ,
    With tRstPeople
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_person_id)) + tC
            ,
            ' name
            ,
            If tCodeStr = "ascii" Then
                If IsNull(!c_name) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
            Else
                If IsNull(!c_name_chn) Then
                    tStr = tStr + "Missing" + tC
                End If
            End If
        Loop
    End With

```

```

        Else
            tStr = tStr + !c_name_chn + tC
        End If

        If IsNull(!c_name) Then
            tStr = tStr + "Missing" + tC
        Else
            tStr = tStr + !c_name + tC
        End If
    End If
    '
    ' indexyear = c_index_year INT
    '
    If IsNull(!c_index_year) Then
        tStr = tStr + "-2000" + tC
    Else
        tStr = tStr + Trim(Str(!c_index_year)) + tC
    End If

    ' dynasty information
    '
    If IsNull(!c_dynasty) Then
        tStr = tStr + "unknown" + tC
    Else
        If tCodeStr = "ascii" Then
            tStr = tStr + !c_dynasty + tC
        Else
            tStr = tStr + !c_dynasty_chn + tC
        End If
    End If
    '
    ' sex = c_female > (F,M)
    tStr = tStr + IIf(!c_female, "F", "M")
    '
    gStream.WriteText tStr, adWriteLine
    '
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now the PeopleEntry file
'
dlgSaveAs.InitialFileName = "PeopleEntry_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
End If
'
gStream.Mode = adModeReadWrite
gStream.Type = adTypeText
gStream.Open

tStr = "NameID" + tC + "EntryCode" + tC + "EntryPlaceID" + tC + "KinID" + tC + "KinRelCode" + tC + _
    "AssocPersonID" + tC + "AssocRelCode" + tC + "SocialInstID" + tC + "EntryYear" + tC + "EntryDynas

```

ty"

```
gStream.WriteText tStr, adWriteLine
```

```
With tRstPeopleEntry
```

```
  .MoveFirst
```

```
  Do While Not .EOF
```

```
    ' the ID of the person
```

```
    tStr = Trim(Str(!c_personid)) + tC
```

```
    '
```

```
    ' entry code
```

```
    '
```

```
    If IsNull(!c_entry_code) Then
```

```
      tStr = tStr + "0" + tC
```

```
    Else
```

```
      tStr = tStr + Trim(Str(!c_entry_code)) + tC
```

```
    End If
```

```
    '
```

```
    ' entry addr id
```

```
    '
```

```
    If IsNull(!c_entry_addr_id) Then
```

```
      tStr = tStr + "0" + tC
```

```
    Else
```

```
      tStr = tStr + Trim(Str(!c_entry_addr_id)) + tC
```

```
    End If
```

```
    '
```

```
    ' kin ID
```

```
    '
```

```
    If IsNull(!c_kin_id) Then
```

```
      tStr = tStr + "0" + tC
```

```
    Else
```

```
      tStr = tStr + Trim(Str(!c_kin_id)) + tC
```

```
    End If
```

```
    '
```

```
    ' kin rel ID
```

```
    '
```

```
    If IsNull(!c_kin_code) Then
```

```
      tStr = tStr + "0" + tC
```

```
    Else
```

```
      tStr = tStr + Trim(Str(!c_kin_code)) + tC
```

```
    End If
```

```
    '
```

```
    ' assoc ID
```

```
    '
```

```
    If IsNull(!c_assoc_id) Then
```

```
      tStr = tStr + "0" + tC
```

```
    Else
```

```
      tStr = tStr + Trim(Str(!c_assoc_id)) + tC
```

```
    End If
```

```
    '
```

```
    ' assoc desc
```

```
    '
```

```
    If IsNull(!c_assoc_code) Then
```

```
      tStr = tStr + "N/A" + tC
```

```
    Else
```

```
      tStr = tStr + Trim(Str(!c_assoc_code)) + tC
```

```
    End If
```

```
    '
```

```
    ' social inst ID
```

```
    '
```

```
    If IsNull(!c_inst_code) Then
```

```
      tStr = tStr + "0" + tC
```

```
    Else
```

```
      tStr = tStr + Right("000000" + Trim(Str(!c_inst_code)), 6) + Right("000000" + Trim(Str(!c_inst_name_code)), 6) + tC
```

```
    End If
```

```
    '
```

```
    ' entry year
```

```
    '
```

```
    If IsNull(!c_year) Then
```

```
      tStr = tStr + "0" + tC
```

```
    Else
```

```
      tStr = tStr + Trim(Str(!c_year)) + tC
```

```
    End If
```

```
    '
```

```
    ' dynasty
```

```
    '
```

```
    If IsNull(!c_dy) Then
```

```
      tStr = tStr + "0"
```

```
    Else
```

```
      tStr = tStr + Trim(Str(!c_dy))
```

```

        End If
    ,
    gStream.WriteText tStr, adWriteLine
    ,
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
' The user pressed Cancel.
GoTo Exit_CmdNeo4j_Click
End If
,
' now places
,
' get a file name
,
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
,
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
tFileName = tFN
If Not tFileName = "" Then
Exit For
End If
Next
If tFileName = "" Then
MsgBox "Bad file Name."
GoTo Exit_CmdNeo4j_Click
Else
' make sure the file name has a txt extension
If Len(tFileName) < 5 Then
tFileName = tFileName + ".csv"
ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
tFileName = tFileName + ".csv"
End If
End If
,
gStream.Open
,
' now process the file
,
' there are three sources of places: the list of people, the entry locations, and the list of institutio
ns
' since ZZ_SCRATCH_P_TEXT has the required fields, just reuse it before copying to ZZ_ADDRESSES
,

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted
,
' get the people IDs
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_addr_id, ZZ_SCRATCH_PEOPLE.c_addr_name, ZZ_SCRATCH_PEOPL
E.c_addr_chn, " + _
"ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord " + _
"FROM ZZ_SCRATCH_PEOPLE"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_entry_addr_id, ZZ_SCRATCH_ENTRY.c_entry_addr_name, ZZ_SCR
ATCH_ENTRY.c_entry_addr_chn, " + _
"ZZ_SCRATCH_ENTRY.c_entry_xcoord, ZZ_SCRATCH_ENTRY.c_entry_ycoord " + _
"FROM ZZ_SCRATCH_ENTRY " + _
"WHERE (((ZZ_SCRATCH_ENTRY.c_entry_addr_id)>0))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT SOCIAL_INSTITUTION_ADDR.c_inst_addr_id, ADDR_CODES.c_name, ADDR_CODES.c_name
_chn, ADDR_CODES.x_coord, ADDR_CODES.y_coord " + _

```

```

"FROM ADDR_CODES INNER JOIN (ZZ_SCRATCH_ENTRY INNER JOIN SOCIAL_INSTITUTION_ADDR " + _
"ON (ZZ_SCRATCH_ENTRY.c_inst_name_code = SOCIAL_INSTITUTION_ADDR.c_inst_name_code) " + _
"AND (ZZ_SCRATCH_ENTRY.c_inst_code = SOCIAL_INSTITUTION_ADDR.c_inst_code)) " + _
"ON (ADDR_CODES.c_addr_id = SOCIAL_INSTITUTION_ADDR.c_inst_addr_id) AND (ADDR_CODES.c_addr_id = SOCIAL_INSTITUTION_ADDR.c_inst_addr_id) " + _
"WHERE (((ZZ_SCRATCH_ENTRY.c_inst_code)>0)))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' now copy the results
cmdSQL.CommandText = "Delete * from ZZ_ADDRESSES"
cmdSQL.Execute tRecDeleted
,
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_addr_id, ZZ_SCRATCH_P_TEXT.c_addr_name, ZZ_SCRATCH_P_TEX
T.c_addr_chn, " + _
"ZZ_SCRATCH_P_TEXT.x_coord, ZZ_SCRATCH_P_TEXT.y_coord " + _
"FROM ZZ_SCRATCH_P_TEXT WHERE (((ZZ_SCRATCH_P_TEXT.c_addr_id)>0)))"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

Set tRstPlace = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
,
If tCodeStr = "ascii" Then
    tStr = "placeID" + tC + "placePY" + tC + "placeX" + tC + "placeY"
Else
    tStr = "placeID" + tC + "placePY" + tC + "placeHZ" + tC + "placeX" + tC + "placeY"
End If
gStream.WriteText tStr, adWriteLine
With tRstPlace
    .MoveFirst
    Do While Not .EOF
        ' the ID of the place
        If Not IsNull(!c_addr_id) Then
            tStr = Trim(Str(!c_addr_id)) + tC
            ,
            ' address name

            If IsNull(!c_name) Then
                tStr = tStr + "unknown" + tC
            Else
                tStr = tStr + !c_name + tC
            End If
            ,
            If Not (tCodeStr = "ascii") Then
                If IsNull(!c_name_chn) Then
                    tStr = tStr + "unknown" + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If
            End If

            ' latitude = !y_coord
            If IsNull(!y_coord) Then
                tStr = tStr + "0.0" + tC
            Else
                tStr = tStr + Str(!y_coord) + tC
            End If

            ' longitude = !x_coord
            If IsNull(!x_coord) Then
                tStr = tStr + "0.0"
            Else
                tStr = tStr + Str(!x_coord)
            End If
            ,
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
,
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close

```



```

Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdNeo4j_Click
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".csv"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
        tFileName = tFileName + ".csv"
    End If
End If
',
gStream.Open
',
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_addr_type, ZZ_SCRATCH_PEOPLE.c_addr_desc, ZZ_SCRATCH_PEO
PLE.c_addr_desc_chn " + _
"FROM ZZ_SCRATCH_PEOPLE WHERE (((ZZ_SCRATCH_PEOPLE.c_addr_type) > 0))"

Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "personPlaceCode" + tC + "personPlaceTrans"
Else
    tStr = "personPlaceCode" + tC + "personPlaceTrans" + tC + "personPlaceHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_addr_type) Then
            '
            tStr = Trim(Str(!c_addr_type)) + tC
            '
            tStr = tStr + !c_addr_desc
            '
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + !c_addr_desc_chn
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
',
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
',
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
',
' finally, get entry codes, kinship codes, association codes, and institution codes, if there are any
',
' now the EntryCode file
',
dlgSaveAs.InitialFileName = "EntryCode_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then

```

```

        tFileName = tFileName + ".csv"
    End If
End If
'
gStream.Mode = adModeReadWrite
gStream.Type = adTypeText
gStream.Open

If tCodeStr = "ascii" Then
    tStr = "EntryCode" + tC + "EntryDesc"
Else
    tStr = "EntryCode" + tC + "EntryDesc" + tC + "EntryDescHZ"
End If
gStream.WriteText tStr, adWriteLine
'
' get the codes
'
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_entry_code, ZZ_SCRATCH_ENTRY.c_entry_desc, ZZ_SCRATCH_ENT
RY.c_entry_chn FROM ZZ_SCRATCH_ENTRY"
Set tRstEntryCode = CurrentDb.OpenRecordset(tQueryStr)
With tRstEntryCode
    .MoveFirst
    Do While Not .EOF
        '
        tStr = Trim(Str(!c_entry_code)) + tC
        '
        ' entry desc
        '
        If IsNull(!c_entry_desc) Then
            tStr = tStr + "Missing"
        Else
            tStr = tStr + Trim(!c_entry_desc)
        End If
        '
        ' kin ID
        '
        If Not (tCodeStr = "ascii") Then
            tStr = tStr + tC + Trim(!c_entry_chn)
        End If
        '
        gStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
"SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_personid " + _
"FROM ZZ_SCRATCH_ENTRY " + _
"WHERE ((ZZ_SCRATCH_ENTRY.c_kin_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_kin_code, ZZ_SCRATCH_ENTRY.c_kin_desc FROM ZZ_SCRATCH_ENTRY W
HERE (((ZZ_SCRATCH_ENTRY.c_kin_code)>0))"
'
If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "KinshipCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        End For
    End If
End If

```

```

Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdNeo4j_Click
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".csv"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
        tFileName = tFileName + ".csv"
    End If
End If
,
gStream.Open
,
Set tRstKinCodes = CurrentDb.OpenRecordset(tQueryStr)
,
tStr = "KinCode" + tC + "KinDesc"
gStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)

With tRstKinCodes
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_kin_code) Then
            tStr = Trim(Str(!c_kin_code)) + tC
            ,
            tStr = tStr + Trim(!c_kin_desc)
            ,
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
,
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
'tGDF.Close
,

Else
    'The user pressed Cancel.
End If

End If
,

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted
,

tQueryStr = "INSERT INTO ZZ_SCRATCH_P TEXT ( c_person_id ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_personid " + _
            "FROM ZZ_SCRATCH_ENTRY " + _
            "WHERE (((ZZ_SCRATCH_ENTRY.c_assoc_code)>0))"
,

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,

tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_assoc_code, ZZ_SCRATCH_ENTRY.c_assoc_desc, ZZ_SCRATCH_ENTRY.c_
_assoc_desc_chn " + _
            "FROM ZZ_SCRATCH_ENTRY WHERE (((ZZ_SCRATCH_ENTRY.c_assoc_code)>0))"

If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "AssocCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then

```

```

        tFileName = tFileName + ".csv"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
        tFileName = tFileName + ".csv"
    End If
End If
'
gStream.Open
'
Set tRstAssocCodes = CurrentDb.OpenRecordset(tQueryStr)
'
If tCodeStr = "ascii" Then
    tStr = "AssocCode" + tC + "AssocDesc"
Else
    tStr = "AssocCode" + tC + "AssocDesc" + tC + "AssocDescHZ"
End If
gStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)

With tRstAssocCodes
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_assoc_code) Then
            tStr = Trim(Str(!c_assoc_code)) + tC
            '
            tStr = tStr + Trim(!c_assoc_desc)
            '
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + !c_assoc_desc_chn
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
End If
End If
'
' the final selection is for social institutions
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_personid " + _
            "FROM ZZ_SCRATCH_ENTRY " + _
            "WHERE (((ZZ_SCRATCH_ENTRY.c_inst_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_inst_code, ZZ_SCRATCH_ENTRY.c_inst_name_code, ZZ_SCRATCH_ENTR
Y.c_inst_name_hz, " + _
            "ZZ_SCRATCH_ENTRY.c_inst_name_py " + _
            "FROM ZZ_SCRATCH_ENTRY WHERE (((ZZ_SCRATCH_ENTRY.c_inst_code)>0))"

If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "InstitutionCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension

```

```

        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    Set tRstInstitutions = CurrentDb.OpenRecordset(tQueryStr)
    '
    If tCodeStr = "ascii" Then
        tStr = "InstitutionCode" + tC + "InstitutionNamePY"
    Else
        tStr = "InstitutionCode" + tC + "InstitutionNamePY" + tC + "InstitutionNameHZ"
    End If
    gStream.WriteText tStr, adWriteLine
    'tGDF.WriteLine (tStr)

    With tRstAssocCodes
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_inst_code) Then
                tStr = Right("000000" + Trim(Str(!c_inst_code)), 6) + Right("000000" + Trim(Str(!c_inst_n
ame_code)), 6) + tC
                '
                If IsNull(!c_inst_name_py) Then
                    tStr = tStr + "NameMissing"
                Else
                    tStr = tStr + Trim(!c_inst_name_py)
                End If
                '
                If Not (tCodeStr = "ascii") Then
                    If IsNull(!c_inst_name_hz) Then
                        tStr = tStr + tC + "NameMissing"
                    Else
                        tStr = tStr + tC + !c_inst_name_hz
                    End If
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    '
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
End If
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted

MsgBox "Finished saving to Neo4j"

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click

End Sub

Private Sub CmdQuery_Click()
    On Error GoTo Err_CmdQuery_Click

    Dim rst As DAO.Recordset
    Dim EntryQuery As DAO.QueryDef, AddressQuery As DAO.QueryDef
    Dim tRstDummy As DAO.Recordset, tRstAddress As DAO.Recordset
    Dim prm As DAO.Parameter, tRstBiogMain As DAO.Recordset

```

```

Dim tRstKinCodes As DAO.Recordset, tRstADDRID As DAO.Recordset
Dim tRstAddrList As DAO.Recordset, tQstr As String, tQt As String
Dim cmdSQL As ADODB.Command, tStrYears As String, tStrFromYear As String, tStrToYear As String, tStrFromAddr
As String, tStrFrom As String

tQt = Chr(34)

Set cmdSQL = New ADODB.Command
'
' to clear the table, briefly close and then delete records
'
Set gRstPeople = Entry_Address_Query.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
Set Entry_Address_Query.Form.Recordset = tRstDummy
gRstPeople.Close

cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ENTRY"
cmdSQL.Execute tRecDeleted
'
' now see if address IDs will be used. If so, zap the scratch file and repopulate
' by looking for address that belong to the address

'MsgBox "About to process address"
If gUseADDRID Then
'
' ZZ_SCRATCH_ADDR has at least one record
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_ADDR_LIST"
cmdSQL.Execute tRecDeleted

'
If ChkSubUnits.Value Then
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) " + _
"SELECT DISTINCT ZZZ_BELONGS_TO.c_addr_id " + _
"FROM ZZ_SCRATCH_ADDR INNER JOIN ZZZ_BELONGS_TO ON " + _
"ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_BELONGS_TO.c_belongs_to"
Else
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT c_addr_id FROM ZZ_SCRATCH
_ADDR"
End If

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

'
' see if we need to use the historical XY search
'
If ChkXYRef.Value Then
'
' the strategy here is to dump the IDs to ZZ_ADDRESSES then copy to ZZ_SCRATCH_ADDR_LIST
' (I borrow ZZ_ADDRESSES from the Pick Addresses form in order to keep the initial selection
' of addresses for the query intact.)
'
' zap the list
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' run the query
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT DISTINCT ADDR_CODES.c_addr_id " + _
"FROM ADDR_CODES, ZZ_SCRATCH_ADDR_LIST INNER JOIN ADDR_CODES AS ADDR_CODES_1 ON " + _
"ZZ_SCRATCH_ADDR_LIST.c_addr_id = ADDR_CODES_1.c_addr_id " + _
"WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.03) And " + _
"(ADDR_CODES.x_coord)<=([ADDR_CODES_1].[x_coord]+0.03)) AND " + _
"((ADDR_CODES.y_coord)>=([ADDR_CODES_1].[y_coord]-0.03) And " + _
"(ADDR_CODES.y_coord)<=([ADDR_CODES_1].[y_coord]+0.03)))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now get the address IDs from the initial list that have no xy coordinates
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT ZZ_SCRATCH_ADDR_LIST.c_addr_id " + _
"FROM ZZ_SCRATCH_ADDR_LIST INNER JOIN ADDR_CODES ON " + _
"ZZ_SCRATCH_ADDR_LIST.c_addr_id = ADDR_CODES.c_addr_id " + _

```

```

        "WHERE (((ADDR_CODES.x_coord) Is Null)) OR (((ADDR_CODES.y_coord) Is Null))"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'   zap ZZ_SCRATCH_ADDR
'
tQueryStr = "DELETE * FROM ZZ_SCRATCH_ADDR_LIST"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'   copy the list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id )SELECT DISTINCT ZZ_ADDRESSES.c_addr_id " +
-
        "FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'   zap the temporary list
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
End If
End If
'MsgBox "Finished processing address"

tStrFromYear = Str(TxtFromYear.Value)
tStrToYear = Str(TxtToYear.Value)

gUseEntryYears = False
gUseIndexYears = False
gUseDynasties = False
If FrameYears.Value = 1 Then      ' Entry Years
    If Not (tStrFromYear = "") And tStrToYear = "" Then
        tStrYears = "(ZZZ_ENTRY_DATA.c_year >=" + Str(TxtFromYear.Value) + ")"
    ElseIf tStrFromYear = "" And Not (tStrToYear = "") Then
        tStrYears = "(ZZZ_ENTRY_DATA.c_year <=" + Str(TxtToYear.Value) + ")"
    ElseIf Not (tStrFromYear = "") And Not (tStrToYear = "") Then
        tStrYears = "(ZZZ_ENTRY_DATA.c_year >=" + Str(TxtFromYear.Value) + _
            " And ZZZ_ENTRY_DATA.c_year <=" + Str(TxtToYear.Value) + ")"
    Else
        tStrYears = ""
    End If
    If Not (tStrYears = "") Then
        gUseEntryYears = True
    End If
ElseIf FrameYears.Value = 2 Then      ' Index Years
    If Not (tStrFromYear = "") And tStrToYear = "" Then
        tStrYears = "(ZZZ_ENTRY_DATA.c_index_year >=" + Str(TxtFromYear.Value) + ")"
    ElseIf tStrFromYear = "" And Not (tStrToYear = "") Then
        tStrYears = "(ZZZ_ENTRY_DATA.c_index_year <=" + Str(TxtToYear.Value) + ")"
    ElseIf Not (tStrFromYear = "") And Not (tStrToYear = "") Then
        tStrYears = "(ZZZ_ENTRY_DATA.c_index_year >=" + Str(TxtFromYear.Value) + _
            " And ZZZ_ENTRY_DATA.c_index_year <=" + Str(TxtToYear.Value) + )"
    Else
        tStrYears = ""
    End If
    If Not (tStrYears = "") Then
        gUseIndexYears = True
    End If
ElseIf FrameYears.Value = 3 Then      ' Dynasties
    If gFromDynasty = -2 Then
        tStrYears = "((ZZZ_ENTRY_DATA.c_dy) > 0 ) "
    ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
        tStrYears = "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tStrYears = "((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tStrYears = "((DYNASTIES.c_dy)=" + Str(gFromDynasty) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tStrYears = "((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
            "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    Else
        ' no constraint have been set, so just ignore

        tStrYears = ""
    End If
    If Not (tStrYears = "") Then

```



```

        gUseDynasties = True
    End If
Else
    tStrYears = ""
End If

tQstr = "INSERT INTO ZZ_SCRATCH_entry ( c_personid, c_name, c_name_chn, c_index_year, c_index_year_type_code,
c_index_year_type_desc, " + _
        "c_index_year_type_hz, c_dy, c_dynasty_chn, c_dynasty, c_entry_code, c_entry_desc, c_entry_chn, c_year, c_
_exam_rank, c_addr_id, c_addr_desc_chn, " + _
        "c_addr_name, c_addr_chn, c_kin_id, c_kin_code, c_kin_desc, c_kin_name, c_kin_chn, c_assoc_code, c_assoc_
_desc, c_assoc_desc_chn, c_assoc_name, " + _
        "c_assoc_name_chn, c_parental_status_desc, c_parental_status_desc_chn, x_coord, y_coord, " + _
        "c_entry_addr_id, c_entry_addr_name, c_entry_addr_chn, c_entry_xcoord, c_entry_ycoord, c_source, c_source
_text_chn, c_source_text ) " + _
        "SELECT ZZZ_ENTRY_DATA.c_personid, ZZZ_ENTRY_DATA.c_name, ZZZ_ENTRY_DATA.c_name_chn, ZZZ_ENTRY_DATA.c_ind
ex_year, " + _
        "ZZZ_ENTRY_DATA.c_index_year_type_code, ZZZ_ENTRY_DATA.c_index_year_type_desc, ZZZ_ENTRY_DATA.c_index_yea
r_type_hz, " + _
        "ZZZ_ENTRY_DATA.c_dy, ZZZ_ENTRY_DATA.c_dynasty_chn, ZZZ_ENTRY_DATA.c_dynasty, " + _
        "ZZZ_ENTRY_DATA.c_entry_code, ZZZ_ENTRY_DATA.c_entry_desc, ZZZ_ENTRY_DATA.c_entry_desc_chn, ZZZ_ENTRY_DAT
A.c_year, ZZZ_ENTRY_DATA.c_exam_rank, " + _
        "ZZZ_ENTRY_DATA.c_addr_id, ZZZ_ENTRY_DATA.c_addr_desc_chn, ZZZ_ENTRY_DATA.c_addr_name, ZZZ_ENTRY_DATA.c_a
ddr_chn, " + _
        "ZZZ_ENTRY_DATA.c_kin_id, ZZZ_ENTRY_DATA.c_kin_code, " + _
        "ZZZ_ENTRY_DATA.c_kinrel AS c_kin_desc, ZZZ_ENTRY_DATA.c_kin_name, ZZZ_ENTRY_DATA.c_kin_name_chn, ZZZ_ENT
RY_DATA.c_assoc_code, ZZZ_ENTRY_DATA.c_assoc_desc, " + _
        "ZZZ_ENTRY_DATA.c_assoc_desc_chn, ZZZ_ENTRY_DATA.c_assoc_name, ZZZ_ENTRY_DATA.c_assoc_name_chn, ZZZ_ENTRY
_DATA.c_parental_status_desc, " + _
        "ZZZ_ENTRY_DATA.c_parental_status_desc_chn, ZZZ_ENTRY_DATA.x_coord, ZZZ_ENTRY_DATA.y_coord, ZZZ_ENTRY_DAT
A.c_entry_addr_id, " + _
        "ZZZ_ENTRY_DATA.c_entry_addr_name, ZZZ_ENTRY_DATA.c_entry_addr_chn, ZZZ_ENTRY_DATA.c_entry_xcoord, ZZZ_EN
TRY_DATA.c_entry_ycoord, " + _
        "ZZZ_ENTRY_DATA.c_source, ZZZ_ENTRY_DATA.c_title_chn, ZZZ_ENTRY_DATA.c_title "

' the FROM statement gets complicated because of the nesting of the inner joins for address, entry code, and
dynasty
' This code, for the sake of clarity, simply sets out all the options

If gUseADDRID Then
    '
    ' use person address = 1, use entry address = 2
    '
    If FrameAddress.Value = 1 Then
        tStrFromAddr = " ZZZ_ENTRY_DATA.c_addr_id "
    Else
        tStrFromAddr = " ZZZ_ENTRY_DATA.c_entry_addr_id "
    End If

    If TxtEntryDesc.Value = "[All]" And TxtTypeCode.Value = "" Then ' No entry codes

        If gUseDynasties Then

            ' join both address and dynasty but no entry code

            tStrFrom = " FROM DYNASTIES INNER JOIN (ZZZ_ENTRY_DATA INNER JOIN ZZ_SCRATCH_ADDR_LIST " + _
                "ON " + tStrFromAddr + " = ZZ_SCRATCH_ADDR_LIST.c_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "
        Else

            ' join just address

            tStrFrom = " FROM ZZZ_ENTRY_DATA INNER JOIN ZZ_SCRATCH_ADDR_LIST ON " + tStrFromAddr + " = ZZ_SCR
ATCH_ADDR_LIST.c_addr_id "
            End If

        ,

    Else ' entry code(s) are specified as well as address

        ' the table ZZ_SCRATCH_ENTRY_CODE contains either one selected code or all the codes for a partic
ular selected TYPE

        If gUseDynasties Then ' this joins all three: address, entry, dynasty

            tStrFrom = " FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ENTRY_CODE INNER JOIN (ZZZ_ENTRY_DATA INNER JO
IN ZZ_SCRATCH_ADDR_LIST " + _
                "ON " + tStrFromAddr + " = ZZ_SCRATCH_ADDR_LIST.c_addr_id) ON ZZ_SCRATCH_ENTRY_CODE.c_entry_c
ode = ZZZ_ENTRY_DATA.c_entry_code) " + _
                "ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "

```

```

Else      ' this joins just address and entry

    tStrFrom = "FROM ZZ_SCRATCH_ENTRY_CODE INNER JOIN " + _
        "(ZZZ_ENTRY_DATA INNER JOIN ZZ_SCRATCH_ADDR_LIST ON " + tStrFromAddr + " = ZZ_SCRATCH_ADDR_LIST.c_addr_id) " + _
        "ON ZZ_SCRATCH_ENTRY_CODE.c_entry_code = ZZZ_ENTRY_DATA.c_entry_code"
    End If

End If

Else      ' No addresses

    If TxtEntryDesc.Value = "[All]" And TxtTypeCode.Value = "" Then ' This is unconstrained and a bad idea unless there is a dynasty constrain

        ' all entry codes are OK: selection is just by place
        If gUseDynasties Then
            tStrFrom = " FROM DYNASTIES INNER JOIN ZZZ_ENTRY_DATA ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "
        Else
            tStrFrom = " FROM ZZZ_ENTRY_DATA "
        End If
    Else
        '
        If gUseDynasties Then ' join dynasty and entry codes

            tStrFrom = " FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ENTRY_CODE INNER JOIN ZZZ_ENTRY_DATA " + _
                "ON ZZ_SCRATCH_ENTRY_CODE.c_entry_code = ZZZ_ENTRY_DATA.c_entry_code) ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "

        Else ' join just entry codes

            tStrFrom = " FROM ZZ_SCRATCH_ENTRY_CODE INNER JOIN ZZZ_ENTRY_DATA ON ZZ_SCRATCH_ENTRY_CODE.c_entry_code = ZZZ_ENTRY_DATA.c_entry_code "
        End If
    End If

    ' FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ENTRY_CODE INNER JOIN ZZZ_ENTRY_DATA ON ZZ_SCRATCH_ENTRY_CODE.c_entry_code = ZZZ_ENTRY_DATA.c_entry_code) ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy;
End If

tQstr = tQstr + tStrFrom

' add the years constraint, if needed

If gUseEntryYears Or gUseIndexYears Or gUseDynasties Then
    ' one last paranoid check
    If Not (tStrYears = "") Then
        tQstr = tQstr + " WHERE (" + tStrYears + ")"
    End If
End If

' MsgBox tQstr

' run the query

cmdSQL.CommandText = tQstr
cmdSQL.Execute tRecDeleted

' now reopen

Set gRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
Set Entry_Address_Query.Recordset = gRstPeople

' the final step is to calculate the xy_count

If gRstPeople.RecordCount > 0 Then
    ' use three SQL calls

    cmdSQL.CommandText = "Delete * from tmpXY"
    cmdSQL.Execute tRecDeleted

    tQstr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
        "SELECT ZZ_SCRATCH_ENTRY.x_coord, ZZ_SCRATCH_ENTRY.y_coord, Count(ZZ_SCRATCH_ENTRY.x_coord) " + _
        "AS CountOfx_coord, Count(ZZ_SCRATCH_ENTRY.y_coord) AS CountOfy_coord " + _
        "FROM ZZ_SCRATCH_ENTRY " + _
        "GROUP BY ZZ_SCRATCH_ENTRY.x_coord, ZZ_SCRATCH_ENTRY.y_coord;"

```

```

cmdSQL.CommandText = tQstr
cmdSQL.Execute tRecDeleted
'
tQstr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_ENTRY ON (tmpXY.y_coord = " + _
"ZZ_SCRATCH_ENTRY.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_ENTRY.x_coord) SET " + _
"ZZ_SCRATCH_ENTRY.xy_count = [tmpXY].[CountOfx_coord];"

cmdSQL.CommandText = tQstr
cmdSQL.Execute tRecDeleted
'
'calculate_xy_count
'
Set gRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
CmdGIS.Enabled = True
CmdStoreID.Enabled = True
CmdNeo4j.Enabled = True
Else
CmdGIS.Enabled = False
CmdStoreID.Enabled = False
CmdNeo4j.Enabled = False
End If

Exit_CmdQuery_Click:
'
' close everything
'
Set rst = Nothing
Set tRstKinCodes = Nothing
Set tRstAddr = Nothing
Set tRstBiogMain = Nothing
Set EntryQuery = Nothing
Set AddressQuery = Nothing
Set tRstDummy = Nothing
Set cmdSQL = Nothing
Exit Sub

Err_CmdQuery_Click:
MsgBox Err.Description
Resume Exit_CmdQuery_Click

End Sub

Private Sub calculate_xy_count()
Dim tX As Double, tY As Double, tXY As Integer, tBM As Variant, tWrite As Integer
'
' the strategy is to first throw a bookmark at the first new value
' then count the number, then go back to the bookmark and update each record
'
With gRstPeople
.Index = "xy"
.MoveFirst

tX = -1#
tY = -1#
tXY = 0
tWrite = 0
tBM = .Bookmark

Do While Not .EOF
If tX <> !x_coord Or tY <> !y_coord Then
If tWrite = 1 Then
' go back to the first record with the value
.Bookmark = tBM
Do While tX = !x_coord And tY = !y_coord
.Edit
!xy_count = tXY
.Update
.MoveNext
Loop
Else
tWrite = 1
End If
' reset
tXY = 0
tBM = .Bookmark
tX = !x_coord
tY = !y_coord
End If
' increment the count and move to the next
tXY = tXY + 1
.MoveNext

```

```

    Loop
    '
    ' the last xy value still needs to be written
    '
    .Bookmark = tBM
    Do While Not .EOF
        .Edit
        !xy_count = tXY
        .Update
        .MoveNext
    Loop
    '
    ' now repeat the process with the entry xy
    '
    .Index = "entry_xy"
    .MoveFirst

    tX = -1#
    tY = -1#
    tXY = 0
    tWrite = 0
    tBM = .Bookmark

    Do While Not .EOF
        If tX <> !c_entry_xcoord Or tY <> !c_entry_ycoord Then
            If tWrite = 1 Then
                ' go back to the first record with the value
                .Bookmark = tBM
                Do While tX = !c_entry_xcoord And tY = !c_entry_ycoord
                    .Edit
                    !c_entry_xy_count = tXY
                    .Update
                    .MoveNext
                Loop
            Else
                tWrite = 1
            End If
            ' reset
            tXY = 0
            tBM = .Bookmark
            tX = !c_entry_xcoord
            tY = !c_entry_ycoord
        End If
        ' increment the count and move to the next
        tXY = tXY + 1
        .MoveNext
    Loop
    '
    ' the last xy value still needs to be written
    '
    .Bookmark = tBM
    Do While Not .EOF
        .Edit
        !c_entry_xy_count = tXY
        .Update
        .MoveNext
    Loop
    .Index = "IndexYear"
End With
End Sub

Private Sub CmdPickEntry_Click()
On Error GoTo Err_CmdPickEntry_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strENTRY As String

Dim cmdSQL As ADODB.Command, tStrID As String
Dim varItm As Variant, tCount As Integer

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

TxtEntryCode.Visible = True
TxtEntryCode.SetFocus
strENTRY = TxtEntryCode.Text

stDocName = "frmPickEntry_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strENTRY

```

```

If CurrentProject.AllForms("frmPickEntry_multi").IsLoaded Then
    Dim intENTRY As Integer
    Dim strENTRY_DESC As String

    'MsgBox "Getting Entry Code value"
    Forms!frmPickEntry_multi.Form!TxtEntryCode.Visible = True
    Forms!frmPickEntry_multi.Form!TxtEntryCode.SetFocus
    intENTRY = Forms!frmPickEntry_multi.Form!TxtEntryCode.Value

    Forms!frmPickEntry_multi.Form!subTreeView.SetFocus
    Forms!frmPickEntry_multi.Form!TxtEntryCode.Visible = False
    TxtEntryCode.Value = intENTRY

    ' zap the temporary table

    cmdSQL.CommandText = "Delete * from zz_scratch_entry_code"
    cmdSQL.Execute tdeleted

    'MsgBox "Processing values"
    If TxtEntryCode.Value < 0 Then
        If TxtEntryCode.Value = -1 Then
            TxtEntryDesc.Value = "[[All]]"
            TxtEntryChn.Value = "[[All]]"
        Else
            TxtEntryDesc.Value = "[[Multi-Select]]"
            TxtEntryChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
        End If

        'MsgBox "Getting TxtTypeID"
        Forms!frmPickEntry_multi.Form!TxtTypeID.Visible = True
        Forms!frmPickEntry_multi.Form!TxtTypeID.SetFocus
        strENTRY_DESC = Forms!frmPickEntry_multi.Form!TxtTypeID.Value
        Forms!frmPickEntry_multi.Form!subTreeView.SetFocus
        Forms!frmPickEntry_multi.Form!TxtTypeID.Visible = False
        TxtTypeCode.Value = strENTRY_DESC

        If TxtTypeCode.Value = "" Then
            TxtTypeDesc.Value = "[ALL]"
            '
            ' multi-select from the root
            '
            If TxtEntryCode.Value = -2 Then
                cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ENTRY_CODE ( c_entry_code ) SELECT DISTINCT c_en
try_code FROM ZZ_ENTRY_CODE"
                cmdSQL.Execute tdeleted
            End If
        Else
            'MsgBox "Getting TxtTypeDesc"
            Forms!frmPickEntry_multi.Form!TxtTypeDesc.Visible = True
            Forms!frmPickEntry_multi.Form!TxtTypeDesc.SetFocus
            strENTRY_DESC = Forms!frmPickEntry_multi.Form!TxtTypeDesc.Value
            Forms!frmPickEntry_multi.Form!subTreeView.SetFocus
            Forms!frmPickEntry_multi.Form!TxtTypeDesc.Visible = False
            TxtTypeDesc.Value = strENTRY_DESC

            'MsgBox "Getting TxtTypeChn"
            Forms!frmPickEntry_multi.Form!TxtTypeChn.Visible = True
            Forms!frmPickEntry_multi.Form!TxtTypeChn.SetFocus
            strENTRY_DESC = Forms!frmPickEntry_multi.Form!TxtTypeChn.Value
            Forms!frmPickEntry_multi.Form!subTreeView.SetFocus
            Forms!frmPickEntry_multi.Form!TxtTypeChn.Visible = False
            TxtTypeChn.Value = strENTRY_DESC

            cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ENTRY_CODE ( c_entry_code ) SELECT DISTINCT c_entry_
code FROM ZZ_ENTRY_CODE"
            cmdSQL.Execute tdeleted

        End If
    Else
        ' extract the entry code
        '
        cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ENTRY_CODE ( c_entry_code ) SELECT " + Str(intENTRY) + "
as c_entry_code"
        cmdSQL.Execute tdeleted

        Forms!frmPickEntry_multi.Form!TxtEntryDesc.Visible = True
        Forms!frmPickEntry_multi.Form!TxtEntryDesc.SetFocus
        strENTRY_DESC = Forms!frmPickEntry_multi.Form!TxtEntryDesc.Value
        Forms!frmPickEntry_multi.Form!subTreeView.SetFocus

```

```

Forms!frmPickEntry_multi.Form!TxtEntryDesc.Visible = False
TxtEntryDesc.Value = strENTRY_DESC

'MsgBox "Getting TxtEntryDescChn"
Forms!frmPickEntry_multi.Form!TxtEntryChn.Visible = True
Forms!frmPickEntry_multi.Form!TxtEntryChn.SetFocus
strENTRY_DESC = Forms!frmPickEntry_multi.Form!TxtEntryChn.Value
Forms!frmPickEntry_multi.Form!subTreeView.SetFocus
Forms!frmPickEntry_multi.Form!TxtEntryChn.Visible = False
TxtEntryChn.Value = strENTRY_DESC

'MsgBox "Getting TxtTypeDesc"
Forms!frmPickEntry_multi.Form!TxtTypeDesc.Visible = True
Forms!frmPickEntry_multi.Form!TxtTypeDesc.SetFocus
strENTRY_DESC = Forms!frmPickEntry_multi.Form!TxtTypeDesc.Value
Forms!frmPickEntry_multi.Form!subTreeView.SetFocus
Forms!frmPickEntry_multi.Form!TxtTypeDesc.Visible = False
TxtTypeDesc.Value = strENTRY_DESC

'MsgBox "Getting TxtTypeChn"
Forms!frmPickEntry_multi.Form!TxtTypeChn.Visible = True
Forms!frmPickEntry_multi.Form!TxtTypeChn.SetFocus
strENTRY_DESC = Forms!frmPickEntry_multi.Form!TxtTypeChn.Value
Forms!frmPickEntry_multi.Form!subTreeView.SetFocus
Forms!frmPickEntry_multi.Form!TxtTypeChn.Visible = False
TxtTypeChn.Value = strENTRY_DESC
TxtTypeCode.Value = ""
End If

' now enable the search

CmdQuery.Enabled = True
CmdSaveEntryCodes.Enabled = True

DoCmd.Close acForm, stDocName
End If

CmdPickEntry.SetFocus
TxtEntryCode.Visible = False

Exit_CmdPickEntry_Click:
Exit Sub

Err_CmdPickEntry_Click:
MsgBox Err.Description
Resume Exit_CmdPickEntry_Click

End Sub

Private Sub CmdSaveEntryCodes_Click()
On Error GoTo Err_CmdSaveEntryCodes_Click
'
' This program will store the current list of office IDs to a .txt file
'
Dim tStream As ADODB.Stream, tStreamNoBOM As ADODB.Stream
Set tStream = New ADODB.Stream

tStream.Charset = "utf-8"
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set tStreamNoBOM = New ADODB.Stream
tStreamNoBOM.Type = adTypeBinary
tStreamNoBOM.Open

' next get a file

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstIDs As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "entry_id_list.txt"
If dlgSaveAs.Show = -1 Then

```

```

'
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
    tFileName = tFN
    If Not tFileName = "" Then
        Exit For
    End If
Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdSaveEntryCodes_Click
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".txt"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".txt") Then
        tFileName = tFileName + ".txt"
    End If
End If

' write the file
' process the table

tStr = "SELECT ZZ_SCRATCH_ENTRY_CODE.c_entry_code, ENTRY_CODES.c_entry_desc, ENTRY_CODES.c_entry_desc_chn
" + _
    "FROM ZZ_SCRATCH_ENTRY_CODE INNER JOIN ENTRY_CODES ON ZZ_SCRATCH_ENTRY_CODE.c_entry_code = ENTRY_CODE
S.c_entry_code"

Set tRstIDs = CurrentDb.OpenRecordset(tStr, dbOpenDynaset)

tTab = Chr(9)

With tRstIDs
    .MoveFirst
    ' MsgBox "writing file"
    Do While Not .EOF
        '
        tStr = Str(!c_entry_code) + tTab + !c_entry_desc + tTab + !c_entry_desc_chn
        tStream.WriteText tStr, adWriteLine
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.Position = 3
MsgBox "Copying to stream"
tStream.CopyTo tStreamNoBOM
MsgBox "Writing to file"
tStreamNoBOM.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstIDs = Nothing

tStream.Close
Set tStream = Nothing
tStreamNoBOM.Close
Set tStreamNoBOM = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdSaveEntryCodes_Click:
Exit Sub

Err_CmdSaveEntryCodes_Click:
MsgBox Err.Description
Resume Exit_CmdSaveEntryCodes_Click

End Sub

Private Sub CmdStoreID_Click()
    Dim cmdSQL As ADODB.Command, tStrQuery As String

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection

```

```

cmdSQL.CommandType = adCmdText
'
If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
    ' Display message.
    If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) =
vbNo Then
        Exit Sub
    Else
        cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
        cmdSQL.Execute tRecCount
    End If
End If

tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_personid FROM Z
Z_SCRATCH_ENTRY"

cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecCount
MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."
'
' update storage source
cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='Entry' WHERE PersonIDSource.LineNum =1"
cmdSQL.Execute tRecCount

End Sub

Private Sub CmdToDynasty_Click()
Dim stDocName As String
Dim stLinkCriteria As String
Dim strToDynasty As String

If gToDynasty = -1 Then
    strToDynasty = ""
Else
    strToDynasty = Str(gToDynasty)
End If

stDocName = "frmPickDynasty"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

    Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
    gToDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
    gToDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
    gToDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
    '
    ' check to see if we have a problem and reject selection if needed
    '
    If gFromDynasty > -1 Then
        If gFromDynastyBegin > gToDynastyEnd Then
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' D
ynasty ends!", vbExclamation
            gToDynasty = -1
            TxtToDynasty.Value = ""
            TxtToDynastyPY.Value = ""
        End If
    End If
    '
    ' value is OK
    '
    If gToDynasty > -1 Then
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
        TxtToDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
        TxtToDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
    End If

    DoCmd.Close acForm, stDocName
    '
    ' reset FromDynasty if necessary (-2 = all dynasties)
    '
    If gFromDynasty = -2 Then
        gFromDynasty = -1
        TxtFromDynasty.Value = ""
    End If

```



```

        TxtFromDynastyPY.Value = ""
    End If
    '
End If

End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim cmdSQL As ADODB.Command
    Dim tRstEntryCode As DAO.Recordset, tRstDummy As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    '
    ' to clear the table, briefly close and then delete records
    '
    Set tRstEntryCode = Entry_Address_Query.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_EC", dbOpenDynaset)
    Set Entry_Address_Query.Form.Recordset = tRstDummy
    tRstEntryCode.Close
    '
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ENTRY"
    cmdSQL.Execute tRecDeleted
    '
    ' now reopen
    '
    Set tRstEntryCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
    Set Entry_Address_Query.Form.Recordset = tRstEntryCode

    ' first determine the language
    gLCID = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    If gLCID = 2052 Or gLCID = 3076 Then      ' 2052 = PRC, 3076 = Hong Kong
        gDisplayLanguage = "S"
    ElseIf gLCID = 4100 Or gLCID = 1028 Then ' 4100 = Singapore, 1028 = Taiwan
        gDisplayLanguage = "T"
        Call changeDisplayLanguage
    Else
        gDisplayLanguage = "E"
        Call changeDisplayLanguage
    End If

    gFromDynasty = -1
    gToDynasty = -1
    gUseIndexYears = False
    gUseDynasties = False
End Sub

Private Sub CmdExit_Click()
On Error GoTo Err_CmdExit_Click

    DoCmd.Close

Exit_CmdExit_Click:
    Exit Sub

Err_CmdExit_Click:
    MsgBox Err.Description
    Resume Exit_CmdExit_Click

End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

    If gDisplayLanguage = "T" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "T"
    End If

    Call changeDisplayLanguage

Exit_CmdFanti_Click:
    Exit Sub

Err_CmdFanti_Click:
    MsgBox Err.Description
    Resume Exit_CmdFanti_Click

```

```

End Sub
Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

    If gDisplayLanguage = "S" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "S"
    End If

    Call changeDisplayLanguage

Exit_CmdJianti_Click:
Exit Sub

Err_CmdJianti_Click:
MsgBox Err.Description
Resume Exit_CmdJianti_Click

End Sub

Public Sub changeDisplayLanguage()
Dim tLabelLanguage(3, 34) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False
With tRstLabelList
    .MoveFirst
    ti = 1

    Do While ti < 34 And Not .EOF
        If !c_form = "LAE" Then
            gLabelsOK = True
            If ti <> !c_label_id Then
                MsgBox "Uh oh: mismatched label table"
                gLabelsOK = False
                Exit Do
            End If
            tLabelLanguage(1, ti) = !c_english
            tLabelLanguage(2, ti) = !c_fanti
            tLabelLanguage(3, ti) = !c_jianti
            ti = ti + 1
        End If
        .MoveNext
    Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If gLabelsOK Then
    If gDisplayLanguage = "E" Then
        tLang = 1
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '

Me.LblFrom.Caption = tLabelLanguage(tLang, 1)
Me.LblTo.Caption = tLabelLanguage(tLang, 2)
Me.LblType.Caption = tLabelLanguage(tLang, 3)
Me.CmdPickEntry.Caption = tLabelLanguage(tLang, 4)
Me.CmdQuery.Caption = tLabelLanguage(tLang, 5)
Me.CmdGIS.Caption = tLabelLanguage(tLang, 6)
Me.CmdFanti.Caption = tLabelLanguage(tLang, 8)
Me.CmdJianti.Caption = tLabelLanguage(tLang, 9)
Me.CmdExit.Caption = tLabelLanguage(tLang, 10)
Me.CmdSelectPlace.Caption = tLabelLanguage(tLang, 11)
Me.CmdImportPlaces.Caption = tLabelLanguage(tLang, 12)
Me.CmdAllPlaces.Caption = tLabelLanguage(tLang, 13)
Me.LblIndexYears.Caption = tLabelLanguage(tLang, 14)
' Me.LblUseYears.Caption = tLabelLanguage(tLang, 15)
Me.LblDisplay.Caption = tLabelLanguage(tLang, 16)

```

```

    Me.LblExamYears.Caption = tLabelLanguage(tLang, 17)
    Me.LblUsePersonAddr.Caption = tLabelLanguage(tLang, 18)
    Me.LblUseEntryAddr.Caption = tLabelLanguage(tLang, 19)
    Me.CmdStoreID.Caption = tLabelLanguage(tLang, 20)
    Me.CmdHelp.Caption = tLabelLanguage(tLang, 21)
    Me.Label37.Caption = tLabelLanguage(tLang, 22)
    Me.LblChkSubUnits.Caption = tLabelLanguage(tLang, 23)
    Me.LblDynasties.Caption = tLabelLanguage(tLang, 24)
    Me.CmdFromDynasty.Caption = tLabelLanguage(tLang, 25)
    Me.CmdToDynasty.Caption = tLabelLanguage(tLang, 26)
    Me.CmdAllDynasties.Caption = tLabelLanguage(tLang, 27)
    Me.LblYears.Caption = tLabelLanguage(tLang, 28)
    Me.LblOptNoDates.Caption = tLabelLanguage(tLang, 29)
    Me.LblOptDynasties.Caption = tLabelLanguage(tLang, 30)
    Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 31)
    Me.CmdImportEntryCodes.Caption = tLabelLanguage(tLang, 32)
    Me.CmdSaveEntryCodes.Caption = tLabelLanguage(tLang, 33)
End If

End Sub

Private Sub CmdAllPlaces_Click()
On Error GoTo Err_CmdAllPlaces_Click

    TxtAddrID.Value = -1

    TxtPlaceChn.Value = ""
    TxtPlace.Value = ""
    gUseADDRID = False
    ChkXYRef.Enabled = False
    ChkSubUnits.Enabled = False

Exit_CmdAllPlaces_Click:
Exit Sub

Err_CmdAllPlaces_Click:
MsgBox Err.Description
Resume Exit_CmdAllPlaces_Click

End Sub

Private Sub CmdImportPlaces_Click()
On Error GoTo Err_CmdImportPlaces_Click

Dim stDocName As String, tRstAddresses As DAO.Recordset
Dim stLinkCriteria As String, tRstImportPlaces As DAO.Recordset
Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String
Dim tLen As Integer, cmdSQL As ADODB.Command

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tFileSystem, tList

' open the list

Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = ""
    If .Show = -1 Then
        ,
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdImportPlaces_Click
        End If
    End If
End With

' Clear the address table now that we are ready to go

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection

```

```

cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR"
cmdSQL.Execute tRecDeleted
'

cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "ImportPlaceList_Space", "TempImportList", tFileName, 0
'   TransferType=acImportDelim
'   SpecificationName = "ImportPlaceList_Space" (apparently it is saved in the database itself)
'   TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
'   HasFieldNames = False (0)
'
'   copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
"FROM ADDR_CODES RIGHT JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID " + _
"WHERE (((ADDR_CODES.c_addr_id) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported:  please look at InputErrorList."
End If
'
'   copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT TempImportList.ImportID " + _
"FROM ADDR_CODES INNER JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    Me.TxtPlace.Value = "[Imported List]"
    Me.TxtPlaceChn.Value = "[Imported List]"
    gUseADDRID = True
    ChkXYRef.Enabled = True
    ChkSubUnits.Enabled = True
End If

Set cmdSQL = Nothing
Set tFileSystem = Nothing

Exit_CmdImportPlaces_Click:
Exit Sub

Err_CmdImportPlaces_Click:
MsgBox Err.Description
Resume Exit_CmdImportPlaces_Click

End Sub

Private Sub CmdSelectPlace_Click()
On Error GoTo Err_CmdSelectPlace_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strADDR As String

TxtAddrID.Visible = True
TxtAddrID.SetFocus
strADDR = TxtAddrID.Text

stDocName = "frmPickAddresses_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR

If CurrentProject.AllForms("frmPickAddresses_multi").IsLoaded Then

    '   if the user selected a group of addresses, ZZ_ADDRESSES will have records

    Dim tAddrID As Long, tRstAddr As DAO.Recordset
    Dim cmdSQL As ADODB.Command

    Set cmdSQL = New ADODB.Command

```

```

cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

gUseADDRID = True
CmdAllPlaces.Enabled = True
ChkXYRef.Enabled = True
ChkSubUnits.Enabled = True

'MsgBox "Checking zz_addresses"

Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Visible = True
Forms!frmPickAddresses_multi.Form!TxtAddrFilter.SetFocus
If Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Value Then
    '
    TxtAddrID.Value = 0
    strADDR_PY = Forms!frmPickAddresses_multi.Form!TxtFilterPY
    strADDR_CHN = Forms!frmPickAddresses_multi.Form!TxtFilterChn

    If strADDR_CHN = "" Then
        TxtPlaceChn.Value = "[[Filter]]"
        TxtPlace.Value = "[" + strADDR_PY + "]"
    Else
        TxtPlaceChn.Value = "[" + strADDR_CHN + "]"
        TxtPlace.Value = "[[Filter]]"
    End If
Else
    Forms!frmPickAddresses_multi.Form!TxtSelectCount.Visible = True
    Forms!frmPickAddresses_multi.Form!TxtSelectCount.SetFocus
    If Forms!frmPickAddresses_multi.Form!TxtSelectCount.Value > 1 Then
        TxtPlaceChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
        TxtPlace.Value = "[[Multi-Select]]"
        TxtAddrID.Value = 0
    Else
        ' only one record in ZZ_ADDRESSES: get its field values
        '
        Set tRstAddr = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
        tRstAddr.MoveFirst
        'MsgBox "Checking zz_addresses: no records"
        TxtAddrID.Value = tRstAddr!c_addr_id
        TxtPlaceChn.Value = tRstAddr!c_name_chn
        TxtPlace.Value = tRstAddr!c_name
        tRstAddr.Close
        Set tRstAddr = Nothing
    End If
End If
'
' now copy the records
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT " + _
    "ZZ_ADDRESSES.c_addr_id FROM ZZ_ADDRESSES"
cmdSQL.Execute tRecDeleted

DoCmd.Close acForm, "frmPickAddresses_multi"

End If
CmdSelectPlace.SetFocus
TxtAddrID.Visible = False

```

```

Exit_CmdSelectPlace_Click:
Exit Sub

```

```

Err_CmdSelectPlace_Click:
MsgBox Err.Description
Resume Exit_CmdSelectPlace_Click

```

```
End Sub
```

```

Private Sub writeKML()
'<kml xmlns="http://www.opengis.net/kml/2.2">
'<Document>
'  <name>ExtendedData+SchemaData</name>
'  <open>1</open>
'  <!-- Create a balloon template referring to the user-defined type -->
'  <Style id="assoc-balloon-template">
'    <BalloonStyle>
'      <text>
'        <![CDATA[

```

```

    $[AssocPerson/PersonNameHZ] <br/>
    ID: $[AssocPerson/PersonID] <br/>
    Index Year: $[AssocPerson/IndexYear] <br/>
    Address: $[AssocPerson/AddrName] $[AssocPerson/AddrNameHZ] <br/>
    XY Count: $[AssocPerson/XYCount] <br/><br/>
  ]]>
</text>
</BalloonStyle>
</Style>
<!-- Declare the type "AssocPerson" with 6 fields -->
<Schema name="AssocPerson" id="AssocPersonId">
  <SimpleField type="string" name="PersonNameHZ">
    <displayName><![CDATA[<b>Person</b>]]></displayName>
  </SimpleField>
  <SimpleField type="string" name="AddrName">
    <displayName><![CDATA[<b>Person</b>]]></displayName>
  </SimpleField>
  <SimpleField type="string" name="AddrNameHZ">
    <displayName><![CDATA[<b>Person</b>]]></displayName>
  </SimpleField>
  <SimpleField type="uint" name="PersonID">
    <displayName><![CDATA[ID]]></displayName>
  </SimpleField>
  <SimpleField type="int" name="IndexYear">
    <displayName><![CDATA[Index Year]]></displayName>
  </SimpleField>
  <SimpleField type="int" name="XYCount">
    <displayName><![CDATA[XY Count]]></displayName>
  </SimpleField>
</Schema>
<!-- Instantiate some Placemarks extended with AssocPerson fields -->
<Placemark>
  <name>Easy trail</name>
  <styleUrl>#assoc-balloon-template</styleUrl>
  <ExtendedData>
    <SchemaData schemaUrl="#AssocPersonId">
      <SimpleData name="PersonID">3.14159</SimpleData>
      <SimpleData name="PersonNameHZ">Pi in the sky</SimpleData>
      <SimpleData name="IndexYear">10</SimpleData>
      <SimpleData name="AddrName">Pi in the sky</SimpleData>
      <SimpleData name="AddrNameHZ">Pi in the sky</SimpleData>
      <SimpleData name="XYCount">10</SimpleData>
    </SchemaData>
  </ExtendedData>
  <Point>
    <coordinates>-122.000,37.002</coordinates>
  </Point>
</Placemark>
<Placemark>
  <name>Difficult trail</name>
  <styleUrl>#assoc-balloon-template</styleUrl>
  <ExtendedData>
    <SchemaData schemaUrl="#AssocPersonId">
      <SimpleData name="TrailHeadName">Mount Everest</SimpleData>
      <SimpleData name="TrailLength">347.45</SimpleData>
      <SimpleData name="ElevationGain">10000</SimpleData>
    </SchemaData>
  </ExtendedData>
  <Point>
    <coordinates>-121.998,37.0078</coordinates>
  </Point>
</Placemark>
</Document>
</kml>

```

```
Dim tStrKML As String
```

```
' This program will dump the results to a .gis file
```

```
If Entry_Address_Query.Form.Recordset.RecordCount = 0 Then
```

```
MsgBox "There are no records to save."
```

```
GoTo Exit_writeKML
```

```
End If
```

```
Dim tStream As ADODB.Stream
```

```
Set tStream = New ADODB.Stream
```

```
If GISFrame.Value = 1 Then
```

```
tStream.Charset = "utf-8"
```

```
tCodeStr = "UTF8"
```

```

Else
    tStream.Charset = "gb2312"
    tCodeStr = "GB2312"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, tDQ As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "entry_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_writeKML
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".kml"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
            tFileName = tFileName + ".kml"
        End If
    End If
    '
    ' write the file
    '
    'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
    'Name,NameChn,IndexYear,EntryDesc,EntryChn,EntryYear,
    'AddrName,AddrChn,X,Y,xy_count
    '
    ' process the table
    '
    Set tRstNode = Entry_Address_Query.Form.Recordset
    tC = Chr(9) ' the tab
    tDQ = Chr(34) ' the double quotation mark
    '
    ' write the header
    '
    tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
    tStream.WriteText "<Document>", adWriteLine
    tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
    tStream.WriteText tC + "<open>1</open>", adWriteLine
    tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

    tStream.WriteText tC + "<Style id=" + tDQ + "entry-balloon-template" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
    tStream.WriteText tC + tC + tC + "<text>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "<![CDATA["", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "$[EntryPerson/PersonNameHZ] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "ID: $[EntryPerson/PersonID] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Index Year: $[EntryPerson/IndexYear] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Entry Year: $[EntryPerson/EntryYear] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Entry Desc: $[EntryPerson/EntryDesc] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Entry Chn: $[EntryPerson/EntryDescHZ] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Entry Rank: $[EntryPerson/EntryRank] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Address: $[EntryPerson/AddrName] $[EntryPerson/AddrNameHZ] <br/>",
adWriteLine
    tStream.WriteText tC + tC + tC + tC + "XY Count: $[EntryPerson/XYCount] <br/><br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
    tStream.WriteText tC + tC + tC + "</text>", adWriteLine
    tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
    tStream.WriteText tC + "</Style>", adWriteLine
    tStream.WriteText tC + "<!-- Declare the type " + tDQ + "EntryPerson" + tDQ + " with 10 fields -->", adWr

```

```

iteLine
    tStream.WriteText tC + "<Schema name=" + tDQ + "EntryPerson" + tDQ + " id=" + tDQ + "EntryPersonId" + tDQ
+ ">", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "PersonNameHZ"
+ tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[<b>Person</b>]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + t
DQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[<b>Person</b>]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrNameHZ" +
tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[<b>Person</b>]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ
+ ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[ID]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" +
tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "EntryYear" + tDQ
+ ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Entry Year]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "EntryDesc" +
tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Entry Desc]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "EntryDescHZ"
+ tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Entry Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "EntryRank" + tDQ
+ ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Entry Rank]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs, even where there should not be any
        '
        ' write the point header
        '
        tStream.WriteText tC + "<Placemark>", adWriteLine

        If IsNull(!c_name) Then
            tStr = "[Bad Data]"
        Else
            tStr = !c_name
        End If
        tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

        tStream.WriteText tC + tC + "<styleUrl>#entry-balloon-template</styleUrl>", adWriteLine
        '
        ' Index Year as time stamp
        '
        If IsNull(!c_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_index_year)
        End If
        tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
        '
        tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#EntryPersonId" + tDQ + ">", a
dWriteLine
        '
        ' person ID
        '
        tStr = Str(!c_personid)

```



```

tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
' Chinese Name
'
If IsNull(!c_name_chn) Then
    tStr = tStr + "[Bad Data]"
Else
    If Trim(!c_name_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_name_chn
    End If
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonNameHZ" + tDQ + ">" + tStr
tr + "</SimpleData>", adWriteLine
'
' Index Year
'
If IsNull(!c_index_year) Then
    tStr = "N/A"
Else
    tStr = Str(!c_index_year)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
' Entry Year
'
If IsNull(!c_year) Then
    tStr = "-2000"
Else
    tStr = Str(!c_year)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "EntryYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
' Entry Desc
'
If IsNull(!c_entry_desc) Then
    tStr = "[Missing Data]"
Else
    tStr = !c_entry_desc
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "EntryDesc" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
' Entry Chn
'
If IsNull(!c_entry_chn) Then
    tStr = "[Missing Data]"
Else
    tStr = !c_entry_chn
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "EntryDescHZ" + tDQ + ">" + tStr
r + "</SimpleData>", adWriteLine
'
' Entry Rank
'
If IsNull(!c_exam_rank) Then
    tStr = "0"
Else
    tStr = !c_exam_rank
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "EntryRank" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
' Address Name
'
If IsNull(!c_addr_name) Then
    tStr = "[?]"
ElseIf Trim(!c_addr_name) = "" Then
    tStr = "[?]"
Else
    tStr = !c_addr_name
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
' Address Name Chinese

```

```

    '
    If IsNull(!c_addr_chn) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_chn
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrNameHZ" + tDQ + ">" + tStr + "</SimpleData>", adWriteLine
    '
    ' XY Count
    '
    If IsNull(!xy_count) Then
        tStr = "0"
    Else
        tStr = Str(!xy_count)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr + "</SimpleData>", adWriteLine
    '
    tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
    tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
    tStream.WriteText tC + tC + "<Point>", adWriteLine
    '
    ' coordinates
    If IsNull(!x_coord) Then
        tStr = "0"
    Else
        tStr = Str(!x_coord)
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + ",0"
    Else
        tStr = tStr + "," + Str(!y_coord)
    End If
    tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
    '
    ' footer
    '
    tStream.WriteText tC + tC + "</Point>", adWriteLine
    tStream.WriteText tC + "</Placemark>", adWriteLine
    .MoveNext
    Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_writeKML:
Exit Sub

Err_writeKML:
MsgBox Err.Description
Resume Exit_writeKML

End Sub

Private Sub FrameYears_Click()

    ' Turn off usage

```

```

gUseEntryYears = False
gUseIndexYears = False
gUseDynasties = False

```

```

' Turn off Dynasty text boxes

```

```

Me.TxtFromDynasty.Enabled = False
Me.TxtFromDynastyPY.Enabled = False
Me.TxtToDynasty.Enabled = False
Me.TxtToDynastyPY.Enabled = False
Me.TxtFromDynasty.Locked = False
Me.TxtFromDynastyPY.Locked = False
Me.TxtToDynasty.Locked = False
Me.TxtToDynastyPY.Locked = False

```

```

If FrameYears.Value = 1 Or FrameYears.Value = 2 Then

```

```

    ' entry years or index years
    Me.CmdFromDynasty.Enabled = False
    Me.CmdToDynasty.Enabled = False
    Me.CmdAllDynasties.Enabled = False

```

```

    Me.TxtFromYear.Enabled = True
    Me.TxtToYear.Enabled = True
    If FrameYears.Value = 1 Then
        gUseEntryYears = True
    Else
        gUseIndexYears = True
    End If

```

```

ElseIf FrameYears.Value = 3 Then

```

```

    ' enable dynasties
    Me.CmdFromDynasty.Enabled = True
    Me.CmdToDynasty.Enabled = True
    Me.CmdAllDynasties.Enabled = True
    Me.TxtFromDynasty.Locked = True
    Me.TxtFromDynastyPY.Locked = True
    Me.TxtToDynasty.Locked = True
    Me.TxtToDynastyPY.Locked = True
    ' disable index years
    Me.TxtFromYear.Enabled = False
    Me.TxtToYear.Enabled = False

```

```

    gUseDynasties = True

```

```

Else
    ' disable all
    Me.CmdFromDynasty.Enabled = False
    Me.CmdToDynasty.Enabled = False
    Me.CmdAllDynasties.Enabled = False
    '
    Me.TxtFromYear.Enabled = False
    Me.TxtToYear.Enabled = False
End If

```

```

End Sub

```

Option Compare Database

' g stands for global

Public gDisplayLanguage As String, gLCID As Integer, gStream As ADODB.Stream, gRstImportPeople As DAO.Recordset
Public gStatusRecCount As Long, gOfficeRecCount As Long, gEntryRecCount As Long, gTextRecCount As Long, gPlaceRec
Count As Long
Public gChkCount As Integer, gChkGisCount As Integer, gPeopleCount As Long

Private Sub ChkAddr_Click()

' the value is AFTER the click

If ChkAddr.Value Then

gChkCount = gChkCount + 1

If gPeopleCount > 0 Then

CmdRun.Enabled = True

End If

FrameQueryAddress.Enabled = True

Else

gChkCount = gChkCount - 1

If gChkCount = 0 Then

CmdRun.Enabled = False

End If

FrameQueryAddress.Enabled = False

End If

'MsgBox Str(gChkCount)

End Sub

Private Sub ChkEntry_Click()

' the value is AFTER the click

If ChkEntry.Value Then

gChkCount = gChkCount + 1

' just as an initial check that I'm doing this right

If gPeopleCount > 0 Then

CmdRun.Enabled = True

End If

Else

gChkCount = gChkCount - 1

If gChkCount = 0 Then

CmdRun.Enabled = False

End If

End If

'MsgBox Str(gChkCount)

End Sub

Private Sub ChkOffice_Click()

' the value is AFTER the click

If ChkOffice.Value Then

gChkCount = gChkCount + 1

' just as an initial check that I'm doing this right

If gPeopleCount > 0 Then

CmdRun.Enabled = True

End If

Else

gChkCount = gChkCount - 1

If gChkCount = 0 Then

CmdRun.Enabled = False

End If

End If

'MsgBox Str(gChkCount)

End Sub

Private Sub ChkStatus_Click()

' the value is AFTER the click

If ChkStatus.Value Then

gChkCount = gChkCount + 1

' just as an initial check that I'm doing this right

If gPeopleCount > 0 Then

CmdRun.Enabled = True

End If

Else

gChkCount = gChkCount - 1

If gChkCount = 0 Then

CmdRun.Enabled = False

End If

End If

'MsgBox Str(gChkCount)

End Sub

```

Private Sub ChkText_Click()
    ' the value is AFTER the click
    If ChkText.Value Then
        gChkCount = gChkCount + 1
        If gPeopleCount > 0 Then
            CmdRun.Enabled = True
        End If
    Else
        gChkCount = gChkCount - 1
        If gChkCount = 0 Then
            CmdRun.Enabled = False
        End If
    End If
    'MsgBox Str(gChkCount)
End Sub

Private Sub CmdClose_Click()
On Error GoTo Err_CmdClose_Click

    DoCmd.Close

Exit_CmdClose_Click:
    Exit Sub

Err_CmdClose_Click:
    MsgBox Err.Description
    Resume Exit_CmdClose_Click

End Sub

Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
    '
    ' If it is a KML file, call the routine and exit
    '
    If ChkGisStatus.Value Then
        Call WriteGIS_Status
    End If
    '
    If Me.ChkGisOffice Then
        Call WriteGIS_OfficeOffice
    End If
    If Me.ChkGisOfficePeople Then
        Call WriteGIS_OfficePeople
    End If
    '
    If ChkGisEntry.Value Then
        Call WriteGIS_Entry
    End If
    '
    If ChkGisText.Value Then
        Call WriteGIS_Text
    End If
    '
    If ChkGisAddr.Value Then
        Call WriteGIS_Addr
    End If
    '

Exit_CmdGIS_Click:
    Exit Sub

Err_CmdGIS_Click:
    MsgBox Err.Description
    Resume Exit_CmdGIS_Click

End Sub

Private Sub CmdImport_Click()
On Error GoTo Err_CmdImport_Click
    Dim cmdSQL As ADODB.Command
    Dim tStrQuestion As String, tQuit As Boolean

    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tFileSystem, tList

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

```

```

tQuit = False
'
If Not tQuit Then
    ' open the list

    Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

    'Use a With...End With block to reference the FileDialog object.

    tFileName = ""

    With dlgSaveAs
        .InitialFileName = ""
        If .Show = -1 Then
            '
            For Each tFN In .SelectedItems
                tFileName = tFN
                If Not tFileName = "" Then
                    Exit For
                End If
            Next
            If tFileName = "" Then
                MsgBox "Bad file Name."
                GoTo Exit_CmdImport_Click
            End If
        End If
    End With
    ' Clear the people table now that we are ready to go
    '
    If Not (tFileName = "") Then
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
        cmdSQL.Execute tRecDeleted
        '
        cmdSQL.CommandText = "Delete * from InputErrorList"
        cmdSQL.Execute tRecDeleted

        cmdSQL.CommandText = "Delete * from TempImportList"
        cmdSQL.Execute tRecDeleted

        DoCmd.TransferText acImportDelim, "ImportPeopleList_Space", "TempImportList", tFileName, 0
        ' TransferType=acImportDelim
        ' SpecificationName = "TempImportList" (apparently it is saved in the database itself)
        ' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
        ' HasFieldNames = False (0)
        '
        ' copy the bad IDs
        '
        tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
            "FROM BIOG_MAIN RIGHT JOIN TempImportList ON BIOG_MAIN.c_personid = TempImportList.ImportID " + _
            "WHERE (((BIOG_MAIN.c_personid) Is Null) AND (TempImportList.ImportID is Not Null))"

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute tRecDeleted

        If tRecDeleted > 0 Then
            MsgBox "Some ID were not successfully imported: please look at InputErrorList."
        End If
        '
        ' copy the good IDs
        '
        tStrSQL = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id ) SELECT DISTINCT TempImportList.Import
ID " + _
            "FROM BIOG_MAIN INNER JOIN TempImportList ON BIOG_MAIN.c_personid = TempImportList.ImportID"

        cmdSQL.CommandText = tStrSQL
        cmdSQL.Execute gPeopleCount

        If gPeopleCount = 0 Then
            CmdRun.Enabled = False
            CmdStoreID.Enabled = False
        Else
            If gChkCount > 0 Then
                CmdRun.Enabled = True
                CmdStoreID.Enabled = True
            End If
        End If
    End If

```

```

        Set cmdSQL = Nothing
        Set tFileSystem = Nothing
    End If

End If

Exit_CmdImport_Click:
Exit Sub

Err_CmdImport_Click:
MsgBox Err.Description
Resume Exit_CmdImport_Click

End Sub

Private Sub CmdNeo4j_Click()
On Error GoTo Err_CmdNeo4j_Click
'
'   The challenge here is that we have 5 tables of results:
'       ZZ_SCRATCH_STATUS
'       ZZ_SCRATCH_OFFICE
'       ZZ_SCRATCH_ENTRY
'       ZZ_SCRATCH_BIOG_TEXT_DATA
'       ZZ_SCRATCH_BIOG_ADDR_DATA
'
'   We need to check all of these for people (entry has 3 IDs) and address IDs, along with their specific data
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
'
Dim tRstPeople As DAO.Recordset, tRstPlace As DAO.Recordset, tRstPeoplePlace As DAO.Recordset
Dim tRstEntryCodes As DAO.Recordset, tRstPeopleEntry As DAO.Recordset, tRstInstitutionCodes As DAO.Recordset
Dim tRstOfficeCodes As DAO.Recordset, tRstPeopleOffice As DAO.Recordset
Dim tRstTextCodes As DAO.Recordset, tRstPeopleText As DAO.Recordset
Dim tRstPeopleStatus As DAO.Recordset, tRstStatusCodes As DAO.Recordset, tStr As String, tC As String
Dim tQueryStr As String, tPersonID As Long, tCount As Long
'
Dim gStream As ADODB.Stream, tCodeStr As String
'
'   set up the stream to write to

Set gStream = New ADODB.Stream
'
If GISFrame.Value = 1 Then
    gStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    gStream.Charset = "big5"
    tCodeStr = "BIG5"
ElseIf GISFrame.Value = 3 Then
    gStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    gStream.Charset = "ascii"
    tCodeStr = "ascii"
End If
'
tC = Chr(44) ' the comma
'
'   get the People file
'
'   prepare the temp tables for the people, place, peoplePlace and entry data

Dim cmdSQL As ADODB.Command, tRecDeleted As Long
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

'   collect the person IDs

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted
'
'   collect from entry
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_personid FROM ZZ_SCRATCH_ENTRY"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

```

```

'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_kin_id " + _
            "FROM ZZ_SCRATCH_ENTRY WHERE (ZZ_SCRATCH_ENTRY.c_kin_id > 0)"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_assoc_id " + _
            "FROM ZZ_SCRATCH_ENTRY WHERE (ZZ_SCRATCH_ENTRY.c_assoc_id > 0)"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' collect from office
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_personid FROM
ZZ_SCRATCH_OFFICE"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' collect from status
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_STATUS.c_personid FROM
ZZ_SCRATCH_STATUS"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' collect from texts
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_BIOG_TEXT_DATA.c_person
id " + _
            "FROM ZZ_SCRATCH_BIOG_TEXT_DATA"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' collect from addresses
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) SELECT DISTINCT ZZ_SCRATCH_BIOG_ADDR_DATA.c_person
id " + _
            "FROM ZZ_SCRATCH_BIOG_ADDR_DATA"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' Open the People file

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
'
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
'
' now process the file (second true removed to make ASCII)
'
' we have a file name: now open the stream for writing

gStream.Mode = adModeReadWrite
gStream.Type = adTypeText
gStream.Open

tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_ch
n, " + _
            "ZZZ_BIOG_MAIN.c_index_year, ZZZ_BIOG_MAIN.c_index_year_type_code, ZZZ_BIOG_MAIN.c_index_
year_type_desc, " + _
            "ZZZ_BIOG_MAIN.c_index_year_type_hz, ZZZ_BIOG_MAIN.c_dy, ZZZ_BIOG_MAIN.c_dynasty, ZZZ_BIO

```



```

G_MAIN.c_dynasty_chn, " + _
    "ZZZ_BIOG_MAIN.c_female " + _
    "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_BIOG_
MAIN.c_personid"

Set tRstPeople = CurrentDb.OpenRecordset(tQueryStr)
'
' process the four tables
'
' first the nodes: define the record structure
'
' if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a pin
y in field
If tCodeStr = "ascii" Then
    tStr = "NameID" + tC + "NamePY" + tC + "IndexYear" + tC + "IndexYearTypeCode" + tC + "IndexYearTypeDe
sc" + tC + _
        "Dynasty" + tC + "Sex"
Else
    tStr = "NameID" + tC + "NameHZ" + tC + "NamePY" + tC + "IndexYear" + tC + "IndexYearTypeCode" + tC +
"IndexYearTypeDesc" + tC + _
        "IndexYearTypeDescHZ" + tC + "Dynasty" + tC + "Sex"
End If
gStream.WriteText tStr, adWriteLine
'
With tRstPeople
    .MoveFirst
    Do While Not .EOF
        ' the ID of the person
        tStr = Trim(Str(!c_person_id)) + tC
        '
        ' name
        '
        If tCodeStr = "ascii" Then
            If IsNull(!c_name) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        Else
            If IsNull(!c_name_chn) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If

            If IsNull(!c_name) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        End If
        '
        ' indexyear = c_index_year INT
        '
        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + tC
        End If
        '
        ' indexyear = c_index_year_type_code STR
        '
        If IsNull(!c_index_year_type_code) Then
            tStr = tStr + "Unknown" + tC
        Else
            tStr = tStr + Trim(!c_index_year_type_code) + tC
        End If
        '
        ' indexyear = c_index_year_type_desc STR
        '
        If IsNull(!c_index_year_type_desc) Then
            tStr = tStr + "Unknown" + tC
        Else
            tStr = tStr + Trim(!c_index_year_type_desc) + tC
        End If
        '
        ' indexyear = c_index_year_type_hz STR
        '
        If Not (tCodeStr = "ascii") Then
            If IsNull(!c_index_year_type_hz) Then

```

```

                tStr = tStr + "Unknown" + tC
            Else
                tStr = tStr + Trim(!c_index_year_type_hz) + tC
            End If
        End If
        ' dynasty information
        ,
        If IsNull(!c_dynasty) Then
            tStr = tStr + "unknown" + tC
        Else
            If tCodeStr = "ascii" Then
                tStr = tStr + !c_dynasty + tC
            Else
                tStr = tStr + !c_dynasty_chn + tC
            End If
        End If
        ,
        If IsNull(!c_female) Then
            tStr = tStr + "Missing"
        Else
            tStr = tStr + IIf(!c_female, "F", "M")
        End If
        ,
        gStream.WriteText tStr, adWriteLine
        ,
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
,
' now places
,
' There are various files from which to address ID: c_index_addr_id, c_entry_addr_id, c_office_addr_id, and
c_inst_addr_id
,
' get a file name
,
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    ' first add index addresses for ZZ_SCRATCH_P_TEXT to ZZ_ADDRESSES
    ,
    cmdSQL.CommandText = "DELETE * FROM ZZ_ADDRESSES"
    cmdSQL.Execute tRecDeleted
    ,
    tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_MAI
N.c_index_addr_chn, " + _
        "ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord " + _
        "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_BIOG

```

MAIN.c_personid"

```

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now from ZZ_SCRATCH_BIOG_ADDR_DATA: the index address for the kin and associates are not in this table
, and all place
' associations for the listed people (more than index addresses) will be in this table
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_id, ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_name, " + _
"ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_chn, ZZ_SCRATCH_BIOG_ADDR_DATA.x_coord, ZZ_SCRATCH_BIOG_ADDR_DATA.y_coord " + _
"FROM ZZ_SCRATCH_BIOG_ADDR_DATA"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now c_entry_addr_id
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_entry_addr_id, ZZ_SCRATCH_ENTRY.c_entry_addr_name, " + _
"ZZ_SCRATCH_ENTRY.c_entry_addr_chn, ZZ_SCRATCH_ENTRY.c_entry_xcoord, ZZ_SCRATCH_ENTRY.c_e
ntry_ycoord " + _
"FROM ZZ_SCRATCH_ENTRY " + _
"WHERE (((ZZ_SCRATCH_ENTRY.c_entry_addr_id)>0))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now c_inst_addr_id
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT SOCIAL_INSTITUTION_ADDR.c_inst_addr_id, ZZ_SCRATCH_ENTRY.c_inst_name_py, " + _
"ZZ_SCRATCH_ENTRY.c_inst_name_hz, SOCIAL_INSTITUTION_ADDR.inst_xcoord, SOCIAL_INSTITUTION
_ADDR.inst_ycoord " + _
"FROM ZZ_SCRATCH_ENTRY INNER JOIN SOCIAL_INSTITUTION_ADDR ON " + _
"(ZZ_SCRATCH_ENTRY.c_inst_name_code = SOCIAL_INSTITUTION_ADDR.c_inst_name_code) AND " + _
"(ZZ_SCRATCH_ENTRY.c_inst_code = SOCIAL_INSTITUTION_ADDR.c_inst_code) " + _
"WHERE (((SOCIAL_INSTITUTION_ADDR.c_inst_addr_id)>0))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now c_office_addr_id
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_office_addr_id, ZZ_SCRATCH_OFFICE.c_office_addr_name, " + _
"ZZ_SCRATCH_OFFICE.c_office_addr_chn, ZZ_SCRATCH_OFFICE.office_x_coord, ZZ_SCRATCH_OFFICE
.office_y_coord " + _
"FROM ZZ_SCRATCH_OFFICE " + _
"WHERE (((ZZ_SCRATCH_OFFICE.c_office_addr_id)>0))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' finally, institution addresses from the office table
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT SOCIAL_INSTITUTION_ADDR.c_inst_addr_id, ZZ_SCRATCH_OFFICE.c_inst_name_py, " + _
"ZZ_SCRATCH_OFFICE.c_inst_name_hz, SOCIAL_INSTITUTION_ADDR.inst_xcoord, SOCIAL_INSTITUTIO
N_ADDR.inst_ycoord " + _
"FROM ZZ_SCRATCH_OFFICE INNER JOIN SOCIAL_INSTITUTION_ADDR ON " + _
"(ZZ_SCRATCH_OFFICE.c_inst_name_code = SOCIAL_INSTITUTION_ADDR.c_inst_name_code) AND " + _
"(ZZ_SCRATCH_OFFICE.c_inst_code = SOCIAL_INSTITUTION_ADDR.c_inst_code) " + _
"WHERE (((SOCIAL_INSTITUTION_ADDR.c_inst_addr_id)>0))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now get the distinct records from ZZ_ADDRESSES
'
tQueryStr = "SELECT DISTINCT ZZ_ADDRESSES.c_addr_id, ZZ_ADDRESSES.c_name, ZZ_ADDRESSES.c_name_chn, " + _
"ZZ_ADDRESSES.x_coord, ZZ_ADDRESSES.y_coord " + _
"FROM ZZ_ADDRESSES"

```

```

'
Set tRstPlace = CurrentDb.OpenRecordset(tQueryStr, dbOpenDynaset)
'
If tCodeStr = "ascii" Then
    tStr = "PlaceID" + tC + "PlacePY" + tC + "PlaceX" + tC + "PlaceY"
Else
    tStr = "PlaceID" + tC + "PlacePY" + tC + "PlaceHZ" + tC + "PlaceX" + tC + "PlaceY"
End If
gStream.WriteText tStr, adWriteLine
With tRstPlace
    .MoveFirst
    Do While Not .EOF
        ' the ID of the place
        If Not IsNull(!c_addr_id) Then
            tStr = Trim(Str(!c_addr_id)) + tC
            ' address name

            If IsNull(!c_name) Then
                tStr = tStr + "unknown" + tC
            Else
                tStr = tStr + !c_name + tC
            End If
            '
            If Not (tCodeStr = "ascii") Then
                If IsNull(!c_name_chn) Then
                    tStr = tStr + "unknown" + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If
            End If

            If IsNull(!x_coord) Then
                tStr = tStr + "0.0" + tC
            Else
                tStr = tStr + Str(!x_coord) + tC
            End If

            If IsNull(!y_coord) Then
                tStr = tStr + "0.0"
            Else
                tStr = tStr + Str(!y_coord)
            End If
            '
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces
'
' the difficulty here is that we want information on both the people (in ZZ_SCRATCH_BIOG_ADDR_DATA) and thei
r associates who appear in
' other tables (listed in ZZ_SCRATCH_P_TEXT)
'
dlgSaveAs.InitialFileName = "PeoplePlaces_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    End If
End If

```

```

Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".csv"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
        tFileName = tFileName + ".csv"
    End If
End If
'
gStream.Open
'
' get all the index address relations
'
cmdSQL.CommandText = "DELETE * FROM ZZ_PLACE"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_PLACE ( c_personid, c_addr_id, c_rel_code, c_rel_desc, c_rel_chn ) " + _
"SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN." & _
c_index_addr_type_code, " + _
"ZZZ_BIOG_MAIN.c_index_addr_type_desc, ZZZ_BIOG_MAIN.c_index_addr_type_chn " + _
"FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_BIOG_" & _
MAIN.c_personid " + _
"WHERE (((ZZZ_BIOG_MAIN.c_index_addr_id)>0))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now the BIOG_ADDR data
'
tQueryStr = "INSERT INTO ZZ_PLACE ( c_personid, c_addr_id, c_rel_code, c_rel_desc, c_rel_chn ) " + _
"SELECT DISTINCT ZZ_SCRATCH_BIOG_ADDR_DATA.c_personid, ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_id, " & _
+ _
"ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_type, ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_desc, " + _
"ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_desc_chn " + _
"FROM ZZ_SCRATCH_BIOG_ADDR_DATA " + _
"WHERE (((ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_id)>0))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_addr_id, ZZ_PLACE.c_rel_code FROM ZZ_PLACE"
Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

tStr = "NameID" + tC + "PlaceID" + tC + "PersonPlaceCode"

gStream.WriteText tStr, adWriteLine

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_personid) Then
            '
            tStr = Trim(Str(!c_personid)) + tC
            '
            tStr = tStr + Trim(Str(!c_addr_id)) + tC
            '
            tStr = tStr + Trim(Str(!c_rel_code))
            '
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaceCode: use ZZ_PLACE
'
dlgSaveAs.InitialFileName = "PeoplePlacesCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then

```

```

'
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
    tFileName = tFN
    If Not tFileName = "" Then
        Exit For
    End If
Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdNeo4j_Click
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".csv"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
        tFileName = tFileName + ".csv"
    End If
End If
'
gStream.Open
'
tQueryStr = "SELECT DISTINCT ZZ_PLACE.c_rel_code, ZZ_PLACE.c_rel_desc, ZZ_PLACE.c_rel_chn FROM ZZ_PLACE"

Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "personPlaceCode" + tC + "personPlaceTrans"
Else
    tStr = "personPlaceCode" + tC + "personPlaceTrans" + tC + "personPlaceHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_rel_code) Then
            '
            tStr = Trim(Str(!c_rel_code)) + tC
            '
            tStr = tStr + !c_rel_desc
            '
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + !c_rel_chn
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' the hard work is over: now we just need to write the status data, entry data, office data, and institution
data
'
' now the PeopleStatus file
'
dlgSaveAs.InitialFileName = "PeopleStatus_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
    End If

```

```

        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    tStr = "NameID" + tC + "StatusCode" + tC + "FirstYear" + tC + "LastYear"
    gStream.WriteText tStr, adWriteLine

    Set tRstPeopleStatus = CurrentDb.OpenRecordset("ZZ_SCRATCH_STATUS", dbOpenDynaset)

    With tRstPeopleStatus
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_personid)) + tC
            ' entry code
            If IsNull(!c_status_code) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_status_code)) + tC
            End If
            ' first year
            If IsNull(!c_firstyear) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_firstyear)) + tC
            End If
            ' last year
            If IsNull(!c_lastyear) Then
                tStr = tStr + "0"
            Else
                tStr = tStr + Trim(Str(!c_lastyear))
            End If
            gStream.WriteText tStr, adWriteLine
            .MoveNext
        Loop
    End With
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If

' finally, get status codes

dlgSaveAs.InitialFileName = "StatusCode_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    End If
End If

```

```

Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".csv"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
        tFileName = tFileName + ".csv"
    End If
End If
'
gStream.Mode = adModeReadWrite
gStream.Type = adTypeText
gStream.Open

If tCodeStr = "ascii" Then
    tStr = "StatusCode" + tC + "StatusDesc"
Else
    tStr = "StatusCode" + tC + "StatusDesc" + tC + "StatusDescHZ"
End If
gStream.WriteText tStr, adWriteLine
'
' get the codes
'
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_STATUS.c_status_code, ZZ_SCRATCH_STATUS.c_status_desc, ZZ_SCRATCH_
_STATUS.c_status_desc_chn " + _
"FROM ZZ_SCRATCH STATUS " + _
"WHERE (ZZ_SCRATCH_STATUS.c_status_code > 0)"

Set tRstStatusCodes = CurrentDb.OpenRecordset(tQueryStr)
With tRstStatusCodes
    .MoveFirst
    Do While Not .EOF
        '
        tStr = Trim(Str(!c_status_code)) + tC
        '
        ' entry desc
        '
        If IsNull(!c_status_desc) Then
            tStr = tStr + "Missing"
        Else
            tStr = tStr + Trim(!c_status_desc)
        End If
        '
        ' kin ID
        '
        If Not (tCodeStr = "ascii") Then
            tStr = tStr + tC + Trim(!c_status_desc_chn)
        End If
        '
        gStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now the PeopleOffice file
'
dlgSaveAs.InitialFileName = "PeopleOffice_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else

```



```

' make sure the file name has a txt extension
If Len(tFileName) < 5 Then
    tFileName = tFileName + ".csv"
ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
    tFileName = tFileName + ".csv"
End If
End If
'
gStream.Mode = adModeReadWrite
gStream.Type = adTypeText
gStream.Open

tStr = "NameID" + tC + "OfficeCode" + tC + "OfficeAddrID" + tC + "SocialInstID" + tC + "SocialInstID" + tC + _
    "PostingFirstYear" + tC + "PostingLastYear" + tC + "PostingDynasty"

gStream.WriteText tStr, adWriteLine
'
Set tRstPeopleOffice = CurrentDb.OpenRecordset("ZZ_SCRATCH_OFFICE", dbOpenDynaset)
With tRstPeopleOffice
    .MoveFirst
    Do While Not .EOF
        ' the ID of the person
        tStr = Trim(Str(!c_personid)) + tC
        '
        ' office code
        '
        If IsNull(!c_office_id) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_office_id)) + tC
        End If
        '
        ' office addr id
        '
        If IsNull(!c_office_addr_id) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_office_addr_id)) + tC
        End If
        '
        ' social inst IDs
        '
        If IsNull(!c_inst_code) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_inst_code)) + tC
        End If
        '
        If IsNull(!c_inst_name_code) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_inst_name_code)) + tC
        End If
        '
        ' posting first year
        '
        If IsNull(!c_firstyear) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_firstyear)) + tC
        End If
        '
        ' posting last year
        '
        If IsNull(!c_lastyear) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_lastyear)) + tC
        End If
        '
        ' posting dynasty
        '
        If IsNull(!c_dy) Then
            tStr = tStr + "0"
        Else
            tStr = tStr + Trim(Str(!c_dy))
        End If
        '
        gStream.WriteText tStr, adWriteLine
    
```

```

        ,
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' get office codes
'
dlgSaveAs.InitialFileName = "OfficeCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    If tCodeStr = "ascii" Then
        tStr = "OfficeCode" + tC + "OfficeTrans" + tC + "OfficePinyin"
    Else
        tStr = "OfficeCode" + tC + "OfficeTrans" + tC + "OfficePinyin" + tC + "OfficeHZ"
    End If
    gStream.WriteText tStr, adWriteLine
    '
    ' get the codes
    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_office_id, ZZ_SCRATCH_OFFICE.c_office_trans, " + _
        "ZZ_SCRATCH_OFFICE.c_office_pinyin, ZZ_SCRATCH_OFFICE.c_office_chn " + _
        "FROM ZZ_SCRATCH_OFFICE"

    Set tRstOfficeCodes = CurrentDb.OpenRecordset(tQueryStr)
    With tRstOfficeCodes
        .MoveFirst
        Do While Not .EOF
            '
            tStr = Trim(Str(!c_office_id)) + tC
            '
            ' office trans
            '
            If IsNull(!c_office_trans) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + Trim(!c_office_trans) + tC
            End If
            '
            ' office pinyin
            '
            If IsNull(!c_office_pinyin) Then
                tStr = tStr + "Missing"
            Else
                tStr = tStr + Trim(!c_office_pinyin)
            End If
            '
            ' office HZ

```

```

    '
    If Not (tCodeStr = "ascii") Then
        tStr = tStr + tC + Trim(!c_office_chn)
    End If
    '
    gStream.WriteText tStr, adWriteLine
    '
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now the PeopleEntry file
'
dlgSaveAs.InitialFileName = "PeopleEntry_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    tStr = "NameID" + tC + "EntryCode" + tC + "EntryPlaceID" + tC + "KinID" + tC + "KinRelCode" + tC + _
        "AssocPersonID" + tC + "AssocRelCode" + tC + "SocialInstID" + tC + "SocialInstNameID" + tC + "Ent
ryYear" + tC + "EntryDynasty"
    gStream.WriteText tStr, adWriteLine
    '
    Set tRstPeopleEntry = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
    With tRstPeopleEntry
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_personid)) + tC
            '
            ' entry code
            '
            If IsNull(!c_entry_code) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_entry_code)) + tC
            End If
            '
            ' entry addr id
            '
            If IsNull(!c_entry_addr_id) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_entry_addr_id)) + tC
            End If
            '
            ' kin ID
            '
            If IsNull(!c_kin_id) Then

```

```

        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Trim(Str(!c_kin_id)) + tC
    End If
    '
    '   kin rel ID
    '
    If IsNull(!c_kin_code) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Trim(Str(!c_kin_code)) + tC
    End If
    '
    '   assoc ID
    '
    If IsNull(!c_assoc_id) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Trim(Str(!c_assoc_id)) + tC
    End If
    '
    '   assoc desc
    '
    If IsNull(!c_assoc_code) Then
        tStr = tStr + "N/A" + tC
    Else
        tStr = tStr + Trim(Str(!c_assoc_code)) + tC
    End If
    '
    '   social inst IDs
    '
    If IsNull(!c_inst_code) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Trim(Str(!c_inst_code)) + tC
    End If
    '
    If IsNull(!c_inst_name_code) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Trim(Str(!c_inst_name_code)) + tC
    End If
    '
    '   entry year
    '
    If IsNull(!c_year) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Trim(Str(!c_year)) + tC
    End If
    '
    '   dynasty
    '
    If IsNull(!c_dy) Then
        tStr = tStr + "0"
    Else
        tStr = tStr + Trim(Str(!c_dy))
    End If
    '
    gStream.WriteText tStr, adWriteLine
    '
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now the EntryCode file
'
dlgSaveAs.InitialFileName = "EntryCode_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '

```

```

tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
    tFileName = tFN
    If Not tFileName = "" Then
        Exit For
    End If
Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdNeo4j_Click
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".csv"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
        tFileName = tFileName + ".csv"
    End If
End If
'
gStream.Mode = adModeReadWrite
gStream.Type = adTypeText
gStream.Open

If tCodeStr = "ascii" Then
    tStr = "EntryCode" + tC + "EntryDesc"
Else
    tStr = "EntryCode" + tC + "EntryDesc" + tC + "EntryDescHZ"
End If
gStream.WriteText tStr, adWriteLine
'
' get the codes
'
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_entry_code, ZZ_SCRATCH_ENTRY.c_entry_desc, ZZ_SCRATCH_ENT
RY.c_entry_chn " + _
"FROM ZZ_SCRATCH_ENTRY WHERE (ZZ_SCRATCH_ENTRY.c_entry_code > -1)"
Set tRstEntryCodes = CurrentDb.OpenRecordset(tQueryStr)
With tRstEntryCodes
    .MoveFirst
    Do While Not .EOF
        '
        tStr = Trim(Str(!c_entry_code)) + tC
        '
        ' entry desc
        '
        If IsNull(!c_entry_desc) Then
            tStr = tStr + "Missing"
        Else
            tStr = tStr + Trim(!c_entry_desc)
        End If
        '
        ' kin ID
        '
        If Not (tCodeStr = "ascii") Then
            tStr = tStr + tC + Trim(!c_entry_chn)
        End If
        '
        gStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' the last step is to collect institution codes from Entry and Office
' first, see if there are any
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_BIOG_INST_DATA"
cmdSQL.Execute tRecDeleted
'
tCount = 0
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_BIOG_INST_DATA ( c_inst_code, c_inst_name_code, c_inst_name_hz,

```

```

c_inst_name_py ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_ENTRY.c_inst_code, ZZ_SCRATCH_ENTRY.c_inst_name_code, " + _
        "ZZ_SCRATCH_ENTRY.c_inst_name_hz, ZZ_SCRATCH_ENTRY.c_inst_name_py " + _
        "FROM ZZ_SCRATCH_ENTRY " + _
        "WHERE (((ZZ_SCRATCH_ENTRY.c_inst_code)>0))"
cmdSQL.Execute tRecDeleted
tCount = tRecDeleted

cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_BIOG_INST_DATA ( c_inst_code, c_inst_name_code, c_inst_name_hz,
c_inst_name_py ) " + _
        "SELECT ZZ_SCRATCH_OFFICE.c_inst_code, ZZ_SCRATCH_OFFICE.c_inst_name_code, " + _
        "ZZ_SCRATCH_OFFICE.c_inst_name_hz, ZZ_SCRATCH_OFFICE.c_inst_name_py " + _
        "FROM ZZ_SCRATCH_OFFICE " + _
        "WHERE (((ZZ_SCRATCH_OFFICE.c_inst_code)>0))"
cmdSQL.Execute tRecDeleted
tCount = tCount + tRecDeleted
'
If tCount > 0 Then
    dlgSaveAs.InitialFileName = "InstitutionCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
        '
        gStream.Mode = adModeReadWrite
        gStream.Type = adTypeText
        gStream.Open

        If tCodeStr = "ascii" Then
            tStr = "InstitutionCode" + tC + "InstitutionNameCode" + tC + "InstitutionName" + tC + "InstitutionAddrID"
        Else
            tStr = "InstitutionCode" + tC + "InstitutionNameCode" + tC + "InstitutionName" + tC + "InstitutionNameHZ" + tC + "InstitutionAddrID"
        End If
        gStream.WriteText tStr, adWriteLine

        tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_BIOG_INST_DATA.c_inst_code, ZZ_SCRATCH_BIOG_INST_DATA.c_inst_name_code, " + _
            "ZZ_SCRATCH_BIOG_INST_DATA.c_inst_name_hz, ZZ_SCRATCH_BIOG_INST_DATA.c_inst_name_py, " + _
            "SOCIAL_INSTITUTION_ADDR.c_inst_addr_id " + _
            "FROM ZZ_SCRATCH_BIOG_INST_DATA INNER JOIN SOCIAL_INSTITUTION_ADDR ON " + _
            "(ZZ_SCRATCH_BIOG_INST_DATA.c_inst_name_code = SOCIAL_INSTITUTION_ADDR.c_inst_name_code) AND " + _
            "(ZZ_SCRATCH_BIOG_INST_DATA.c_inst_code = SOCIAL_INSTITUTION_ADDR.c_inst_code)"

        Set tRstInstitutionCodes = CurrentDb.OpenRecordset(tQueryStr)

        With tRstInstitutionCodes
            .MoveFirst
            Do While Not .EOF
                '
                tStr = Trim(Str(!c_inst_code)) + tC
                '
                If IsNull(!c_inst_name_code) Then
                    tStr = tStr + "0" + tC
                Else
                    tStr = tStr + Trim(Str(!c_inst_name_code)) + tC
                End If
                '
                ' Name (pinyin)
                '
                If IsNull(!c_inst_name_py) Then

```

```

        tStr = tStr + "Missing" + tC
    Else
        tStr = tStr + Trim(!c_inst_name_py) + tC
    End If
    '
    '   Name   (HZ)
    '
    If Not (tCodeStr = "ascii") Then
        If IsNull(!c_inst_name_hz) Then
            tStr = tStr + "Missing" + tC
        Else
            tStr = tStr + Trim(!c_inst_name_hz) + tC
        End If
    End If
    '
    If IsNull(!c_inst_addr_id) Then
        tStr = tStr + "0"
    Else
        tStr = tStr + Trim(Str(!c_inst_addr_id))
    End If
    '
    gStream.WriteText tStr, adWriteLine
    '
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
End If
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing
MsgBox "Finished saving to Neo4j"

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click

End Sub

Private Sub CmdRun_Click()
On Error GoTo Err_CmdRun_Click

If ChkStatus.Value Then
    Call queryStatus
End If

If ChkOffice.Value Then
    Call queryOffice
End If

If ChkEntry.Value Then
    Call queryEntry
End If

If ChkText.Value Then
    Call queryText
End If

If ChkAddr.Value Then
    Call queryAddr
End If

If CmdGIS.Enabled Then
    CmdNeo4j.Enabled = True
Else
    CmdNeo4j.Enabled = False
End If

Exit_CmdRun_Click:

```

```

'
' close the tables
'
Exit Sub

Err_CmdRun_Click:
MsgBox Err.Description + tErrorStr
Resume Exit_CmdRun_Click

End Sub

Private Sub calculate_xy_count()
Dim tX As Double, tY As Double, tXY As Integer, tBM As Variant, tWrite As Integer
Dim tID As Long
'
' the strategy is to first throw a bookmark at the first new value
' then count the number, then go back to the bookmark and update each record
'
' in order to allow the use of the index, gRstPersonID must be reopened as a table
'
Set gRstPersonID = CurrentDb.OpenRecordset("ZZ_SCRATCH_KIN", dbOpenTable)
With gRstPersonID
.Index = "xy"
.MoveFirst

tX = -1#
tY = -1#
tXY = 0
tWrite = 0
tBM = .Bookmark

Do While Not .EOF
If tX <> !kin_x_coord Or tY <> !kin_y_coord Then
If tWrite = 1 Then
' go back to the first record with the value
.Bookmark = tBM
Do While tX = !kin_x_coord And tY = !kin_y_coord
.Edit
!xy_count = tXY
.Update
.MoveNext
Loop
Else
tWrite = 1
End If
' reset
tXY = 0
tBM = .Bookmark
tX = !kin_x_coord
tY = !kin_y_coord
tID = -2
End If
' increment the count and move to the next
If tID <> !c_kin_id Then
tXY = tXY + 1
tID = !c_kin_id
End If
.MoveNext
Loop
' the last xy value still needs to be written
'
.Bookmark = tBM
Do While Not .EOF
.Edit
!xy_count = tXY
.Update
.MoveNext
Loop
.Index = "IndexYear"
End With
End Sub

Private Sub Form_Open(Cancel As Integer)
Dim tRstDummy As DAO.Recordset
Dim cmdDel As ADODB.Command, tRecDeleted As Long
'
' Clear the input and output tables
'

```



```

Set cmdDel = New ADODB.Command
cmdDel.ActiveConnection = CurrentProject.Connection
cmdDel.CommandType = adCmdText
cmdDel.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
cmdDel.Execute tRecDeleted
'
' clear status
'
Set Me.STATUS.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SC", dbOpenDynaset)
cmdDel.CommandText = "Delete * from ZZ_SCRATCH_STATUS"
cmdDel.Execute tRecDeleted
Set Me.STATUS.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_STATUS", dbOpenDynaset)

cmdDel.CommandText = "DELETE * FROM ZZ_SCRATCH_P_STATUS"
cmdDel.Execute gStatusRecCount
'
' clear office
Set Me.OFFICE.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_OF", dbOpenDynaset)
cmdDel.CommandText = "Delete * from ZZ_SCRATCH_OFFICE"
cmdDel.Execute tRecDeleted
Set Me.OFFICE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_OFFICE", dbOpenDynaset)
'
' clear entry
'
Set Me.ENTRY.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_ENTRY", dbOpenDynaset)
cmdDel.CommandText = "Delete * from ZZ_SCRATCH_ENTRY"
cmdDel.Execute tRecDeleted
Set Me.ENTRY.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
'
' clear text data
'
Set Me.Text.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_TR", dbOpenDynaset)
cmdDel.CommandText = "Delete * from ZZ_SCRATCH_BIOG_TEXT_DATA"
cmdDel.Execute tRecDeleted
Set Me.Text.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_BIOG_TEXT_DATA", dbOpenDynaset)

cmdDel.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
cmdDel.Execute gStatusRecCount
'
' clear place association
'
Set Me.PLACE.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_BA", dbOpenDynaset)
cmdDel.CommandText = "Delete * from ZZ_SCRATCH_BIOG_ADDR_DATA"
cmdDel.Execute tRecDeleted
Set Me.PLACE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_BIOG_ADDR_DATA", dbOpenDynaset)

Set cmdDel = Nothing
'
' Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_KIN", dbOpenDynaset)
' Set Forms!LookAtKinship!frmZZ_SCRATCH_KIN.Form.Recordset = tRstDummy
' gRstPersonID.Close
' Set gRstPersonID = CurrentDb.OpenRecordset("ZZ_SCRATCH_KIN", dbOpenDynaset)
' Set Forms!LookAtKinship!frmZZ_SCRATCH_KIN.Form.Recordset = gRstPersonID
'
' Set Forms!LookAtKinship!frmZZ_SCRATCH_KINNET.Form.Recordset = tRstDummy
' gRstKinList.Close
' Set gRstKinList = CurrentDb.OpenRecordset("ZZ_SCRATCH_KINNET", dbOpenDynaset)
' Set Forms!LookAtKinship!frmZZ_SCRATCH_KINNET.Form.Recordset = gRstKinList
'
' Set tRstDummy = Nothing
'
' first determine the language
gLCID = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
If gLCID = 2052 Or gLCID = 4100 Then      ' 2052 = PRC, 4100 = Singapore
    gDisplayLanguage = "S"
ElseIf gLCID = 3076 Or gLCID = 1028 Then ' 3076 = Hong Kong, 1028 = Taiwan
    gDisplayLanguage = "T"
    'Call changeDisplayLanguage
Else
    gDisplayLanguage = "E"
    'Call changeDisplayLanguage
End If

If DCount("?", "ZZ_STORE_PERSON_ID") > 0 Then
    CmdRecallID.Enabled = True
End If
'
' initially all 5 checkboxes are checked
'
gChkCount = 5

```

```

    gChkGisCount = 0
    gPeopleCount = 0
End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

    If gDisplayLanguage = "T" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "T"
    End If

    Call changeDisplayLanguage

Exit_CmdFanti_Click:
    Exit Sub

Err_CmdFanti_Click:
    MsgBox Err.Description
    Resume Exit_CmdFanti_Click

End Sub

Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

    If gDisplayLanguage = "S" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "S"
    End If

    Call changeDisplayLanguage

Exit_CmdJianti_Click:
    Exit Sub

Err_CmdJianti_Click:
    MsgBox Err.Description
    Resume Exit_CmdJianti_Click

End Sub

Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 31) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 31 And Not .EOF
            If !c_form = "LAG" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then

```

```

        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '
    Me.CmdImport.Caption = tLabelLanguage(tLang, 1)
    Me.CmdRecallID.Caption = tLabelLanguage(tLang, 2)
    Me.CmdRun.Caption = tLabelLanguage(tLang, 3)
    Me.CmdStoreID.Caption = tLabelLanguage(tLang, 4)
    Me.CmdFanti.Caption = tLabelLanguage(tLang, 5)
    Me.CmdJianti.Caption = tLabelLanguage(tLang, 6)
    Me.CmdHelp.Caption = tLabelLanguage(tLang, 7)
    Me.CmdClose.Caption = tLabelLanguage(tLang, 8)
    Me.CmdGIS.Caption = tLabelLanguage(tLang, 9)
    '
    Me.LblChkStatus.Caption = tLabelLanguage(tLang, 10)
    Me.LblChkOffice.Caption = tLabelLanguage(tLang, 11)
    Me.LblChkEntry.Caption = tLabelLanguage(tLang, 12)
    Me.LblChkText.Caption = tLabelLanguage(tLang, 13)
    Me.LblChkAddr.Caption = tLabelLanguage(tLang, 14)
    Me.LblChkGisStatus.Caption = tLabelLanguage(tLang, 15)
    Me.LblChkGisOfficeOffice.Caption = tLabelLanguage(tLang, 16)
    Me.LblChkGisOfficePeople.Caption = tLabelLanguage(tLang, 17)
    Me.LblChkGisEntry.Caption = tLabelLanguage(tLang, 18)
    Me.LblChkGisText.Caption = tLabelLanguage(tLang, 19)
    Me.LblChkGisAddr.Caption = tLabelLanguage(tLang, 20)
    '
    Me.PageStatus.Caption = tLabelLanguage(tLang, 21)
    Me.PageOffice.Caption = tLabelLanguage(tLang, 22)
    Me.PageEntry.Caption = tLabelLanguage(tLang, 23)
    Me.PageTexts.Caption = tLabelLanguage(tLang, 24)
    Me.PagePlace.Caption = tLabelLanguage(tLang, 25)
    '
    Me.LblOptIndexAddr.Caption = tLabelLanguage(tLang, 26)
    Me.LblOptAllAddr.Caption = tLabelLanguage(tLang, 27)
    Me.LblGIS_PY.Caption = tLabelLanguage(tLang, 28)
    '
    Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 29)
    Me.LblDisplay.Caption = tLabelLanguage(tLang, 30)
End If

End Sub

Private Sub CmdHelp_Click()
On Error GoTo Err_CmdHelp_Click
    Dim tStrPDF As String

    tStrPDF = Application.CurrentProject.Path + "\HelpFiles\HelpFile_LookAtKinship.pdf"

    'MsgBox tStrPDF

    Application.FollowHyperlink tStrPDF, , True

Exit_CmdHelp_Click:
Exit Sub

Err_CmdHelp_Click:
    MsgBox Err.Description
    Resume Exit_CmdHelp_Click

End Sub

Private Sub WriteKML_Office()

    Dim tStrKML As String, tPinyin As Boolean
    '
    ' This program will dump the results to a .kml file
    '
    If gOfficeRecCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_WriteKML_Office
    End If
    '
    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream

    tPinyin = False

```

```

If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    tStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "office_office_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_WriteKML_Office
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".kml"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
            tFileName = tFileName + ".kml"
        End If
    End If
    '
    ' write the file
    '
    'SELECT ZZ_SCRATCH_OFFICE.c_name AS Name, ZZ_SCRATCH_OFFICE.c_name_chn AS NameChn,
    'ZZ_SCRATCH_OFFICE.c_index_year AS IndexYear, ZZ_SCRATCH_OFFICE.c_sex AS Sex,
    'ZZ_SCRATCH_OFFICE.c_addr_name AS AddrName, ZZ_SCRATCH_OFFICE.c_addr_chn AS AddrChn,
    'Str(ZZ_SCRATCH_OFFICE.x_coord) AS PersonX, Str(ZZ_SCRATCH_OFFICE.y_coord) AS PersonY,
    'ZZ_SCRATCH_OFFICE.c_office_trans AS Office, ZZ_SCRATCH_OFFICE.c_office_chn AS OfficeChn,
    'ZZ_SCRATCH_OFFICE.c_firstyear AS FirstYear, ZZ_SCRATCH_OFFICE.c_lastyear AS LastYear,
    'ZZ_SCRATCH_OFFICE.c_dy_desc AS Dynasty,
    'ZZ_SCRATCH_OFFICE.c_office_addr_name AS OfficeAddr,
    'ZZ_SCRATCH_OFFICE.c_office_addr_chn AS OfficeAddrChn,
    'Str(ZZ_SCRATCH_OFFICE.office_x_coord) AS X, Str(ZZ_SCRATCH_OFFICE.office_y_coord) AS Y,
    'ZZ_SCRATCH_OFFICE.office_xy_count AS XY_count

    '
    tStr = "PostingID" (c_posting_id) + "Office" (c_name) + "OfficeChn" (c_name_chn) + _
        "FirstYear" (c_firstyear) + "LastYear" + (c_lastyear) _
        "Dynasty" (c_dy) + "OfficeAddr" (c_office_addr_name) + "OfficeAddrHZ" (c_office_addr_chn) + _
        "X" (office_x_coord) + "Y" (office_y_coord) + "xy_count" (office_xy_count)
    '
    ' process the table
    '
    Set tRstNode = OFFICE.Form.Recordset
    tC = Chr(9) ' the tab
    tDQ = Chr(34) ' the double quotation mark
    '
    ' write the header
    '
    tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
    tStream.WriteText "<Document>", adWriteLine
    tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
    tStream.WriteText tC + "<open>1</open>", adWriteLine
    tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLi

```

ne

```

tStream.WriteText tC + "<Style id=" + tDQ + "office-balloon-template" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
tStream.WriteText tC + tC + tC + "<text>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "<![CDATA["", adWriteLine
tStream.WriteText tC + tC + tC + tC + "ID: ${OfficePosting/PostingID} <br/>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Person Chn: ${OfficePosting/PersonNameHZ} <br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "Begin Year: ${OfficePosting/BeginYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "End Year: ${OfficePosting/EndYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Office Desc: ${OfficePosting/OfficeName} <br/>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Office Chn: ${OfficePosting/OfficeNameHZ} <br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "Office Dynasty: ${OfficePosting/OfficeDyn} <br/>", adWriteLine
If tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Address: ${OfficePosting/AddrName} <br/>", adWriteLine
Else
    tStream.WriteText tC + tC + tC + tC + "Address: ${OfficePosting/AddrName} ${OfficePosting/AddrNameHZ}
<br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "XY Count: ${OfficePosting/XYCount} <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine
tStream.WriteText tC + "<!-- Declare the type " + tDQ + "OfficePosting" + tDQ + " with 6 fields -->", adW
riteLine
tStream.WriteText tC + "<Schema name=" + tDQ + "OfficePosting" + tDQ + " id=" + tDQ + "OfficePostingId" +
tDQ + ">", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "PersonNam
eHZ" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + t
DQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrNameH
Z" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PostingID" + tD
Q + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[ID]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "FirstYear" + tDQ
+ ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Begin Year]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "LastYear" + tDQ
+ ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[End Year]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "OfficeName" +
tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Office Name]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "OfficeNam
eHZ" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Office Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "OfficeDyn" + tDQ
+ ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Office Dyn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + "</Schema>", adWriteLine

```

With tRstNode

```

'
.MoveFirst
Do While Not .EOF
' must guard against NULLs, even where there should not be any
'
' write the point header
'
tStream.WriteText tC + "<Placemark>", adWriteLine

If IsNull(!c_person_name) Then
tStr = "[Bad Data] " + Str(!c_posting_id)
Else
tStr = !c_person_name + " " + Str(!c_posting_id)
End If
tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

tStream.WriteText tC + tC + "<styleUrl>#office-balloon-template</styleUrl>", adWriteLine
'
' First Year as time stamp
'
If IsNull(!c_firstyear) Then
tStr = "0"
Else
tStr = Str(!c_firstyear)
End If
tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
'
tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#OfficePostingID" + tDQ + ">",
adWriteLine
'
' posting ID
'
tStr = Str(!c_posting_id)
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PostingID" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
' Person Name Chn
'
If Not tPinyin Then
If IsNull(!c_person_name_chn) Then
tStr = tStr + "[Bad Data]"
Else
If Trim(!c_person_name_chn) = "" Then
tStr = "[?]"
Else
tStr = !c_person_name_chn
End If
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonNameHZ" + tDQ + ">"
+ tStr + "</SimpleData>", adWriteLine
End If
'
' Office Name
'
If IsNull(!c_office_trans) Then
tStr = tStr + "[Bad Data]"
Else
If Trim(!c_office_trans) = "" Then
tStr = "[?]"
Else
tStr = !c_office_trans
End If
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "OfficeName" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
' Office Chinese Name
'
If Not tPinyin Then
If IsNull(!c_office_chn) Then
tStr = tStr + "[Bad Data]"
Else
If Trim(!c_office_chn) = "" Then
tStr = "[?]"
Else
tStr = !c_office_chn
End If
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "OfficeNameHZ" + tDQ + ">"

```

```

+ tStr + "</SimpleData>", adWriteLine
    End If
    '
    ' First Year
    '
    If IsNull(!c_firstyear) Then
        tStr = "0"
    Else
        tStr = Str(!c_firstyear)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "FirstYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    ' Last Year
    '
    If IsNull(!c_lastyear) Then
        tStr = "0"
    Else
        tStr = Str(!c_lastyear)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "LastYear" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    ' Office Dynasty
    '
    If IsNull(!c_dy) Then
        tStr = "0"
    Else
        tStr = Str(!c_dy)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "OfficeDyn" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    ' Address Name
    '
    If IsNull(!c_office_addr_name) Then
        tStr = "[?]"
    ElseIf Trim(!c_office_addr_name) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_office_addr_name
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    ' Address Name Chinese
    '
    If Not tPinyin Then
        If IsNull(!c_office_addr_chn) Then
            tStr = "[?]"
        ElseIf Trim(!c_office_addr_chn) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_office_addr_chn
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrNameHZ" + tDQ + ">" +
tStr + "</SimpleData>", adWriteLine
    End If
    '
    ' XY Count
    '
    If IsNull(!office_xy_count) Then
        tStr = "0"
    Else
        tStr = Str(!office_xy_count)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
    tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
    tStream.WriteText tC + tC + "<Point>", adWriteLine
    '
    ' coordinates
    If IsNull(!office_x_coord) Then
        tStr = "0"
    Else
        tStr = Str(!office_x_coord)
    End If

```

```

        If IsNull(!office_y_coord) Then
            tStr = tStr + ",0"
        Else
            tStr = tStr + "," + Str(!office_y_coord)
        End If
        tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
    '
    ' footer
    '
    tStream.WriteText tC + tC + "</Point>", adWriteLine
    tStream.WriteText tC + "</Placemark>", adWriteLine
    .MoveNext
Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteKML_Office:
Exit Sub

Err_WriteKML_Office:
MsgBox Err.Description
Resume Exit_WriteKML_Office

End Sub

Private Sub CmdStoreID_Click()
    Dim cmdSQL As ADODB.Command, tRecCount As Variant

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
        ' Display message.
        If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) = vbNo Then
            Exit Sub
        Else
            cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
            cmdSQL.Execute tRecCount
        End If
    End If

    tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT DISTINCT ZZ_SCRATCH_IMPORT_PEOPLE.c_personid FROM ZZ_SCRATCH_IMPORT_PEOPLE"

    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecCount
    MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."

End Sub

Private Sub CmdRecallID_Click()
On Error GoTo Err_CmdRecallID_Click
    Dim tStrSQL As String, tRecCount As Long, tStrQuestion As String, tRst As DAO.Recordset, tID As Long

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    tRecCount = DCount("*", "ZZ_SCRATCH_IMPORT_PEOPLE")

```



```

If tRecCount > 0 Then
    If tRecCount = 1 Then
        tStrQuestion = "Do you wish to replace the current person?"
    Else
        tStrQuestion = "Do you wish to replace the current list of IDs?"
    End If
    ' Display message.
    If MsgBox(tStrQuestion, vbYesNo + vbQuestion + vbDefaultButton2) = vbNo Then
        Exit Sub
    Else
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
        cmdSQL.Execute tRecCount
    End If
End If
'
' Clear the error table now that we are ready to go
'
cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted
'
' copy the IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id ) SELECT DISTINCT c_personid FROM ZZ_STORE_PERS
ON_ID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute gPeopleCount

If gPeopleCount = 0 Then
    CmdRun.Enabled = False
    CmdStoreID.Enabled = False
Else
    If gChkCount > 0 Then
        CmdRun.Enabled = True
    End If
    CmdStoreID.Enabled = True
End If

Set cmdSQL = Nothing

Exit_CmdRecallID_Click:
Exit Sub

Err_CmdRecallID_Click:
MsgBox Err.Description
Resume Exit_CmdRecallID_Click

End Sub

Private Sub queryStatus()
    Dim cmdSQL As ADODB.Command, tStrInsert As String, tStrSelect As String, tStrFrom As String, tRst As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    ' first clear ZZ_SCRATCH_STATUS

    Set Me.STATUS.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SC", dbOpenDynaset)

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_STATUS"
    cmdSQL.Execute gStatusRecCount

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_STATUS"
    cmdSQL.Execute gStatusRecCount

    tStrInsert = "INSERT INTO ZZ_SCRATCH_STATUS ( c_personid, c_sequence, c_status_code, c_status_desc, c_status_
desc_chn, " + _
        "c_firstyear, c_fy_nh_code, c_fy_nh_chn, c_fy_nh_py, c_fy_nh_year, c_fy_range, c_fy_range_desc, c_fy_rang
e_chn, c_lastyear, " + _
        "c_ly_nh_code, c_ly_nh_chn, c_ly_nh_py, c_ly_nh_year, c_ly_range, c_ly_range_desc, c_ly_range_chn, c_sour
ce, c_title_chn, c_title, " + _
        "c_pages, c_notes, c_name, c_name_chn, c_index_year, c_index_year_type_code, c_index_year_type_desc, c_in
dex_year_type_hz, c_sex, " + _
        "c_ethnicity_code, c_ethnicity_chn, c_ethnicity_rmn, c_dy, c_dynasty, c_dynasty_chn, c_addr_id, c_addr_na
me, c_addr_chn, x coord, y coord ) "
    tStrSelect = "SELECT ZZZ_STATUS_DATA.c_personid, ZZZ_STATUS_DATA.c_sequence, ZZZ_STATUS_DATA.c_status_code, Z
ZZ_STATUS_DATA.c_status_desc, " + _
        "ZZZ_STATUS_DATA.c_status_desc_chn, ZZZ_STATUS_DATA.c_firstyear, ZZZ_STATUS_DATA.c_fy_nh_code, ZZZ_STATUS

```

```

_DATA.c_fy_nh_chn, " + _
    "ZZZ_STATUS_DATA.c_fy_nh_py, ZZZ_STATUS_DATA.c_fy_nh_year, ZZZ_STATUS_DATA.c_fy_range, ZZZ_STATUS_DATA.c_fy_range_desc, " + _
    "ZZZ_STATUS_DATA.c_fy_range_chn, ZZZ_STATUS_DATA.c_lastyear, ZZZ_STATUS_DATA.c_ly_nh_code, ZZZ_STATUS_DATA.c_ly_nh_chn, " + _
    "ZZZ_STATUS_DATA.c_ly_nh_py, ZZZ_STATUS_DATA.c_ly_nh_year, ZZZ_STATUS_DATA.c_ly_range, ZZZ_STATUS_DATA.c_ly_range_desc, " + _
    "ZZZ_STATUS_DATA.c_ly_range_chn, ZZZ_STATUS_DATA.c_source, ZZZ_STATUS_DATA.c_title_chn, ZZZ_STATUS_DATA.c_title, ZZZ_STATUS_DATA.c_pages, " + _
    "ZZZ_STATUS_DATA.c_notes, ZZZ_STATUS_DATA.c_name, ZZZ_STATUS_DATA.c_name_chn, ZZZ_STATUS_DATA.c_index_year, " + _
    "ZZZ_STATUS_DATA.c_index_year_type_code, ZZZ_STATUS_DATA.c_index_year_type_desc, ZZZ_STATUS_DATA.c_index_year_type_hz, " + _
    "ZZZ_STATUS_DATA.c_sex, ZZZ_STATUS_DATA.c_ethnicity_code, ZZZ_STATUS_DATA.c_ethnicity_chn, ZZZ_STATUS_DATA.c_ethnicity_rmn, " + _
    "ZZZ_STATUS_DATA.c_dy, ZZZ_STATUS_DATA.c_dynasty, ZZZ_STATUS_DATA.c_dynasty_chn, ZZZ_STATUS_DATA.c_addr_id, ZZZ_STATUS_DATA.c_addr_name, " + _
    "ZZZ_STATUS_DATA.c_addr_chn, ZZZ_STATUS_DATA.x_coord, ZZZ_STATUS_DATA.y_coord "
tStrFrom = "FROM ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZZ_STATUS_DATA ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZZ_STATUS_DATA.c_personid"

cmdSQL.CommandText = tStrInsert + tStrSelect + tStrFrom
cmdSQL.Execute gStatusRecCount
'
' the final step is to insert the people into P_STATUS and calculate the xy_count
'
If gStatusRecCount > 0 Then
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_STATUS ( c_person_id, c_name, c_name_chn, c_index_year, c_index_year_type_code, c_index_year_type_desc, " + _
        "c_index_year_type_hz, c_dy, c_dynasty, c_dynasty_chn, c_sex, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_STATUS.c_personid, ZZ_SCRATCH_STATUS.c_name, ZZ_SCRATCH_STATUS.c_name_chn, ZZ_SCRATCH_STATUS.c_index_year, " + _
        "ZZ_SCRATCH_STATUS.c_index_year_type_code, ZZ_SCRATCH_STATUS.c_index_year_type_desc, ZZ_SCRATCH_STATUS.c_index_year_type_hz, " + _
        "ZZ_SCRATCH_STATUS.c_dy, ZZ_SCRATCH_STATUS.c_dynasty, ZZ_SCRATCH_STATUS.c_dynasty_chn, " + _
        "ZZ_SCRATCH_STATUS.c_sex, ZZ_SCRATCH_STATUS.c_addr_id, ZZ_SCRATCH_STATUS.c_addr_name, " + _
        "ZZ_SCRATCH_STATUS.c_addr_chn, ZZ_SCRATCH_STATUS.x_coord, ZZ_SCRATCH_STATUS.y_coord " + _
        "FROM ZZ_SCRATCH_STATUS"
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecCount
    ' calculate the xy_count
    cmdSQL.CommandText = "Delete * from tmpXY"
    cmdSQL.Execute tRecDeleted
    tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
        "SELECT ZZ_SCRATCH_P_STATUS.x_coord, ZZ_SCRATCH_P_STATUS.y_coord, Count(ZZ_SCRATCH_P_STATUS.x_coord) " + _
        "AS CountOfx_coord, Count(ZZ_SCRATCH_P_STATUS.y_coord) AS CountOfy_coord " + _
        "FROM ZZ_SCRATCH_P_STATUS " + _
        "GROUP BY ZZ_SCRATCH_P_STATUS.x_coord, ZZ_SCRATCH_P_STATUS.y_coord"
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecCount
    tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_P_STATUS ON (tmpXY.y_coord = " + _
        "ZZ_SCRATCH_P_STATUS.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_P_STATUS.x_coord) " + _
        "SET ZZ_SCRATCH_P_STATUS.xy_count = [tmpXY].[CountOfx_coord]"
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecCount

Me.ChkGisStatus.Enabled = True
If Me.ChkGisStatus.Value = False Then
    gChkGisCount = gChkGisCount + 1
    Me.ChkGisStatus.Value = True
End If
CmdGIS.Enabled = True
Else
    If ChkGisStatus.Value Then
        Me.ChkGisStatus.Value = False
        gChkGisCount = gChkGisCount - 1
    End If
    Me.ChkGisStatus.Enabled = False

If gChkGisCount = 0 Then
    CmdGIS.Enabled = False

```

```

        End If
    End If

    Set Me.STATUS.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_STATUS", dbOpenDynaset)
End Sub
Private Sub queryOffice()
    Dim cmdSQL As ADODB.Command, tStrInsert As String, tStrSelect As String, tStrFrom As String, tRst As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    ' first clear ZZ_SCRATCH_OFFICE
    Set Me.OFFICE.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_OF", dbOpenDynaset)

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_OFFICE"
    cmdSQL.Execute gOfficeRecCount

    tStrInsert = "INSERT INTO ZZ_SCRATCH_OFFICE ( c_person_name, c_person_name_chn, c_index_year, c_index_year_type_code, c_index_year_type_desc, " + _
        "c_index_year_type_hz, c_female, c_sex, c_person_dy, c_person_dynasty, c_person_dy_chn, c_personid, c_posting_id, c_office_id, " + _
        "c_office_pinyin, c_office_chn, c_office_trans, c_sequence, c_firstyear, c_fy_nh_code, c_fy_nh_chn, c_fy_nh_py, c_fy_nh_year, " + _
        "c_fy_range, c_fy_range_desc, c_fy_range_chn, c_lastyear, c_ly_nh_code, c_ly_nh_chn, c_ly_nh_py, c_ly_nh_year, c_ly_nh_range, " + _
        "c_ly_range_desc, c_ly_range_chn, c_appt_code, c_appt_desc_chn, c_appt_desc, c_assume_office_code, " + _
        "c_assume_office_desc_chn, c_assume_office_desc, c_inst_code, c_inst_name_code, c_inst_name_hz, c_inst_name_py, c_source, c_title_chn, " + _
        "c_title, c_pages, c_notes, c_fy_intercalary, c_fy_month, c_ly_intercalary, c_ly_month, c_fy_day, c_ly_day, c_fy_day_gz, c_fy_day_gz_chn, " + _
        "c_fy_day_gz_py, c_ly_day_gz, c_ly_day_gz_chn, c_ly_day_gz_py, c_dy, c_dynasty, c_dynasty_chn, c_office_category_id, c_category_desc, " + _
        "c_category_desc_chn, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_admin_type, c_office_addr_id, c_office_addr_name, " + _
        "c_office_addr_chn, office_x_coord, office_y_coord, c_addr_type, c_addr_desc, c_addr_desc_chn ) "

    tStrSelect = "SELECT ZPAD.c_person_name, ZPAD.c_person_name_chn, ZPAD.c_index_year, ZPAD.c_index_year_type_code, ZPAD.c_index_year_type_desc, " + _
        "ZPAD.c_index_year_type_hz, ZPAD.c_female, IIf([ZPAD.c_female],'F','M') AS c_sex, ZPAD.c_person_dy, ZPAD.c_person_dynasty, " + _
        "ZPAD.c_person_dy_chn, ZPAD.c_personid, ZPAD.c_posting_id, ZPAD.c_office_id, ZPAD.c_office_pinyin, ZPAD.c_office_chn, " + _
        "ZPAD.c_office_trans, ZPAD.c_sequence, ZPAD.c_firstyear, ZPAD.c_fy_nh_code, ZPAD.c_fy_nh_chn, ZPAD.c_fy_nh_py, ZPAD.c_fy_nh_year, " + _
        "ZPAD.c_fy_range, ZPAD.c_fy_range_desc, ZPAD.c_fy_range_chn, ZPAD.c_lastyear, ZPAD.c_ly_nh_code, ZPAD.c_ly_nh_chn, ZPAD.c_ly_nh_py, " + _
        "ZPAD.c_ly_nh_year, ZPAD.c_ly_range, ZPAD.c_ly_range_desc, ZPAD.c_ly_range_chn, ZPAD.c_appt_code, ZPAD.c_appt_desc_chn, " + _
        "ZPAD.c_appt_desc, ZPAD.c_assume_office_code, ZPAD.c_assume_office_desc_chn, ZPAD.c_assume_office_desc, ZPAD.c_inst_code, " + _
        "ZPAD.c_inst_name_code, ZPAD.c_inst_name_hz, ZPAD.c_inst_name_py, ZPAD.c_source, ZPAD.c_title_chn, ZPAD.c_title, ZPAD.c_pages, " + _
        "ZPAD.c_notes, ZPAD.c_fy_intercalary, ZPAD.c_fy_month, ZPAD.c_ly_intercalary, ZPAD.c_ly_month, ZPAD.c_fy_day, ZPAD.c_ly_day, " + _
        "ZPAD.c_fy_day_gz, ZPAD.c_fy_day_gz_chn, ZPAD.c_fy_day_gz_py, ZPAD.c_ly_day_gz, ZPAD.c_ly_day_gz_chn, ZPAD.c_ly_day_gz_py, " + _
        "ZPAD.c_dy, ZPAD.c_dynasty, ZPAD.c_dynasty_chn, ZPAD.c_office_category_id, ZPAD.c_category_desc, ZPAD.c_category_desc_chn, " + _
        "ZPAD.c_addr_id, ZPAD.c_addr_name, ZPAD.c_addr_chn, ZPAD.x_coord, ZPAD.y_coord, ZPAD.c_admin_type, ZPAD.c_office_addr_id, " + _
        "ZPAD.c_office_addr_name, ZPAD.c_office_addr_chn, ZPAD.office_x_coord, ZPAD.office_y_coord, ZPAD.c_addr_type, ZPAD.c_addr_desc, " + _
        "ZPAD.c_addr_desc_chn "

    tStrFrom = "FROM ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS ZPAD ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZPAD.c_personid"
    cmdSQL.CommandText = tStrInsert + tStrSelect + tStrFrom
    cmdSQL.Execute gOfficeRecCount

    '
    ' the next step is to calculate the xy_count for office addresses
    '
    If gOfficeRecCount > 0 Then
        ' Now the People
        '
        tStrQuery = "INSERT INTO ZZ_SCRATCH_P_OFFICE ( c_personid, c_name, c_name_chn, c_index_year, " + _
            "c_index_year_type_code, c_index_year_type_desc, c_index_year_type_hz, " + _

```

```

        "c_female, c_sex, c_dy, c_dynasty, c_dynasty_chn, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, " +
        "c_addr_desc, c_addr_desc_chn, x_coord, y_coord ) " +
        "SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_personid, ZZ_SCRATCH_OFFICE.c_person_name AS c_name, " +
        "ZZ_SCRATCH_OFFICE.c_person_name_chn AS c_name_chn, ZZ_SCRATCH_OFFICE.c_index_year, " +
        "ZZ_SCRATCH_OFFICE.c_index_year_type_code, ZZ_SCRATCH_OFFICE.c_index_year_type_desc, ZZ_SCRATCH_OFFICE.c_index_year_type_hz, " +
        "ZZ_SCRATCH_OFFICE.c_female, ZZ_SCRATCH_OFFICE.c_sex, ZZ_SCRATCH_OFFICE.c_person_dy AS c_dy, ZZ_SCRATCH_OFFICE.c_person_dynasty AS c_dynasty, " +
        "ZZ_SCRATCH_OFFICE.c_person_dy_chn AS c_dynasty_chn, ZZ_SCRATCH_OFFICE.c_addr_id, " +
        "ZZ_SCRATCH_OFFICE.c_addr_name, ZZ_SCRATCH_OFFICE.c_addr_chn, ZZ_SCRATCH_OFFICE.c_addr_type, " +
        "ZZ_SCRATCH_OFFICE.c_addr_desc, ZZ_SCRATCH_OFFICE.c_addr_desc_chn, ZZ_SCRATCH_OFFICE.x_coord, " +
        "ZZ_SCRATCH_OFFICE.y_coord " +
        "FROM ZZ_SCRATCH_OFFICE"
    ,
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    ,
    ' first the people xy
    ,
    ' use three SQL calls
    ,
    cmdSQL.CommandText = "DELETE * FROM tmpXY"
    cmdSQL.Execute tRecDeleted
    ,
    tStrQuery = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " +
        "SELECT ZZ_SCRATCH_P_OFFICE.x_coord, ZZ_SCRATCH_P_OFFICE.y_coord, Count(ZZ_SCRATCH_P_OFFICE.x_coord) " +
        "AS CountOfx_coord, Count(ZZ_SCRATCH_P_OFFICE.y_coord) AS CountOfy_coord " +
        "FROM ZZ_SCRATCH_P_OFFICE " +
        "GROUP BY ZZ_SCRATCH_P_OFFICE.x_coord, ZZ_SCRATCH_P_OFFICE.y_coord;"
    ,
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    ,
    tStrQuery = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_P_OFFICE ON (tmpXY.y_coord = " +
        "ZZ_SCRATCH_P_OFFICE.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_P_OFFICE.x_coord) " +
        "SET ZZ_SCRATCH_P_OFFICE.xy_count = [tmpXY].[CountOfx_coord];"
    ,
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    ,
    ' then the offices
    ,
    cmdSQL.CommandText = "DELETE * FROM tmpXY"
    cmdSQL.Execute tRecDeleted
    ,
    tStrQuery = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " +
        "SELECT ZZ_SCRATCH_OFFICE.office_x_coord, ZZ_SCRATCH_OFFICE.office_y_coord, " +
        "Count(ZZ_SCRATCH_OFFICE.office_x_coord) AS CountOfx_coord, " +
        "Count(ZZ_SCRATCH_OFFICE.office_y_coord) AS CountOfy_coord " +
        "FROM ZZ_SCRATCH_OFFICE " +
        "GROUP BY ZZ_SCRATCH_OFFICE.office_x_coord, ZZ_SCRATCH_OFFICE.office_y_coord"
    ,
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    ,
    tStrQuery = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_OFFICE ON (tmpXY.y_coord = " +
        "ZZ_SCRATCH_OFFICE.office_y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_OFFICE.office_x_coord) " +
        "SET ZZ_SCRATCH_OFFICE.office_xy_count = [tmpXY].[CountOfx_coord];"
    ,
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    ,
    Me.ChkGisOffice.Enabled = True
    Me.ChkGisOfficePeople.Enabled = True
    If Me.ChkGisOffice.Value = False Then
        gChkGisCount = gChkGisCount + 1
        Me.ChkGisOffice.Value = True
    End If
    If Me.ChkGisOfficePeople.Value = False Then
        gChkGisCount = gChkGisCount + 1
        Me.ChkGisOfficePeople.Value = True
    End If
    CmdGIS.Enabled = True
Else
    If ChkGisOffice.Value Then
        Me.ChkGisOffice.Value = False
        gChkGisCount = gChkGisCount - 1
    End If

```

```

End If
If ChkGisOfficePeople.Value Then
    Me.ChkGisOfficePeople.Value = False
    gChkGisCount = gChkGisCount - 1
End If
Me.ChkGisOfficePeople.Enabled = False
Me.ChkGisOffice.Enabled = False

If gChkGisCount = 0 Then
    CmdGIS.Enabled = False
End If
End If
,
Set Me.OFFICE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_OFFICE", dbOpenDynaset)
End Sub
Private Sub queryEntry()
    Dim cmdSQL As ADODB.Command, tStrInsert As String, tStrSelect As String, tStrFrom As String, tRst As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    ' first clear ZZ_SCRATCH_STATUS

    Set Me.ENTRY.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_ENTRY", dbOpenDynaset)

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_ENTRY"
    cmdSQL.Execute gEntryRecCount

    tStrInsert = "INSERT INTO ZZ_SCRATCH_ENTRY ( c_personid, c_name, c_name_chn, c_index_year, c_index_year_type_code, c_index_year_type_desc, " + _
        "c_index_year_type_hz, c_dy, c_dynasty, c_dynasty_chn, c_entry_code, c_entry_desc, c_entry_chn, c_sequence, c_exam_rank, c_kin_id, " + _
        "c_kin_code, c_kin_desc, c_kin_name, c_kin_chn, c_assoc_desc, c_assoc_desc_chn, c_assoc_id, c_assoc_name, c_assoc_name_chn, c_year, " + _
        "c_inst_code, c_inst_name_code, c_inst_name_hz, c_inst_name_py, c_source, c_source_text_chn, c_source_text, c_age, c_parental_status_desc, " + _
        "c_parental_status_desc_chn, c_addr_id, c_addr_desc, c_addr_desc_chn, c_addr_name, c_addr_chn, x_coord, y_coord, c_entry_addr_id, " + _
        "c_entry_addr_name, c_entry_addr_chn, c_entry_xcoord, c_entry_ycoord ) "
    tStrSelect = "SELECT ZED.c_personid, ZED.c_name, ZED.c_name_chn, ZED.c_index_year, ZED.c_index_year_type_code, ZED.c_index_year_type_desc, " + _
        "ZED.c_index_year_type_hz, ZED.c_dy, ZED.c_dynasty, ZED.c_dynasty_chn, ZED.c_entry_code, ZED.c_entry_desc, ZED.c_entry_desc_chn, " + _
        "ZED.c_sequence, ZED.c_exam_rank, ZED.c_kin_id, ZED.c_kin_code, ZED.c_kinrel, ZED.c_kin_name, ZED.c_kin_name_chn, " + _
        "ZED.c_assoc_desc, ZED.c_assoc_desc_chn, ZED.c_assoc_id, ZED.c_assoc_name, ZED.c_assoc_name_chn, ZED.c_year, ZED.c_inst_code, " + _
        "ZED.c_inst_name_code, ZED.c_inst_name_hz, ZED.c_inst_name_py, ZED.c_source, ZED.c_title_chn, ZED.c_title, ZED.c_age, " + _
        "ZED.c_parental_status_desc, ZED.c_parental_status_desc_chn, ZED.c_addr_id, ZED.c_addr_desc, ZED.c_addr_desc_chn, ZED.c_addr_name, " + _
        "ZED.c_addr_chn, ZED.x_coord, ZED.y_coord, ZED.c_entry_addr_id, ZED.c_entry_addr_name, ZED.c_entry_addr_chn, ZED.c_entry_xcoord, " + _
        "ZED.c_entry_ycoord "
    tStrFrom = "FROM ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZZ_ENTRY_DATA AS ZED ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZED.c_personid"

    cmdSQL.CommandText = tStrInsert + tStrSelect + tStrFrom
    cmdSQL.Execute gEntryRecCount
    ,
    ' the final step is to calculate the xy_count
    ,
    If gEntryRecCount > 0 Then
        ,
        cmdSQL.CommandText = "Delete * from tmpXY"
        cmdSQL.Execute tRecDeleted
        ,
        tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
            "SELECT ZZ_SCRATCH_ENTRY.x_coord, ZZ_SCRATCH_ENTRY.y_coord, Count(ZZ_SCRATCH_ENTRY.x_coord) " + _
            "AS CountOfx_coord, Count(ZZ_SCRATCH_ENTRY.y_coord) AS CountOfy_coord " + _
            "FROM ZZ_SCRATCH_ENTRY " + _
            "GROUP BY ZZ_SCRATCH_ENTRY.x_coord, ZZ_SCRATCH_ENTRY.y_coord"
        ,
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecCount
        ,
        tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_ENTRY ON (tmpXY.y_coord = " + _
            "ZZ_SCRATCH_ENTRY.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_ENTRY.x_coord) " + _

```

```

        "SET ZZ_SCRATCH_ENTRY.xy_count = [tmpXY].[CountOfx_coord]"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount

Me.ChkGisEntry.Enabled = True
If Me.ChkGisEntry.Value = False Then
    gChkGisCount = gChkGisCount + 1
    Me.ChkGisEntry.Value = True
End If
CmdGIS.Enabled = True
Else
    If ChkGisEntry.Value Then
        Me.ChkGisEntry.Value = False
        gChkGisCount = gChkGisCount - 1
    End If
    Me.ChkGisEntry.Enabled = False

    If gChkGisCount = 0 Then
        CmdGIS.Enabled = False
    End If
End If

Set Me.ENTRY.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
End Sub

Private Sub queryText()
    Dim cmdSQL As ADODB.Command, tStrInsert As String, tStrSelect As String, tStrFrom As String, tRst As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    ' first clear ZZ_SCRATCH_STATUS

    Set Me.Text.Form.Recordset = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_TR", dbOpenDynaset)

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_BIOG_TEXT_DATA"
    cmdSQL.Execute gTextRecCount

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
    cmdSQL.Execute gTextRecCount

    tStrInsert = "INSERT INTO ZZ_SCRATCH_BIOG_TEXT_DATA ( c_personid, c_name_chn, c_name, c_sex, c_dy, c_dynasty, c_dynasty_chn, " + _
        "c_index_year, c_index_year_type_code, c_index_year_type_desc, c_index_year_type_hz, c_addr_id, c_addr_name, x_coord, y_coord, c_textid, " + _
        "c_title, c_title_chn, c_text_cat_code, c_text_cat_desc, c_text_cat_desc_chn, c_role_id, c_role_desc, c_role_desc_chn, " + _
        "c_source, c_source_chn, c_source_title, c_pages, c_notes ) "
    tStrSelect = "SELECT ZZ_BIOG_TEXT_DATA.c_personid, ZZ_BIOG_TEXT_DATA.c_name_chn, ZZ_BIOG_TEXT_DATA.c_name, ZZ_BIOG_TEXT_DATA.c_sex, " + _
        "ZZ_BIOG_TEXT_DATA.c_dy, ZZ_BIOG_TEXT_DATA.c_dynasty, ZZ_BIOG_TEXT_DATA.c_dynasty_chn, ZZ_BIOG_TEXT_DATA.c_index_year, " + _
        "ZZ_BIOG_TEXT_DATA.c_index_year_type_code, ZZ_BIOG_TEXT_DATA.c_index_year_type_desc, ZZ_BIOG_TEXT_DATA.c_index_year_type_hz, " + _
        "ZZ_BIOG_TEXT_DATA.c_addr_id, ZZ_BIOG_TEXT_DATA.c_addr_name, ZZ_BIOG_TEXT_DATA.x_coord, ZZ_BIOG_TEXT_DATA.y_coord, " + _
        "ZZ_BIOG_TEXT_DATA.c_textid, ZZ_BIOG_TEXT_DATA.c_title, ZZ_BIOG_TEXT_DATA.c_title_chn, ZZ_BIOG_TEXT_DATA.c_bibl_cat_code, " + _
        "ZZ_BIOG_TEXT_DATA.c_bibl_cat_desc, ZZ_BIOG_TEXT_DATA.c_bibl_cat_desc_chn, ZZ_BIOG_TEXT_DATA.c_role_id, " + _
        "ZZ_BIOG_TEXT_DATA.c_role_desc, ZZ_BIOG_TEXT_DATA.c_role_desc_chn, ZZ_BIOG_TEXT_DATA.c_source, ZZ_BIOG_TEXT_DATA.c_source_chn, " + _
        "ZZ_BIOG_TEXT_DATA.c_source_title, ZZ_BIOG_TEXT_DATA.c_pages, ZZ_BIOG_TEXT_DATA.c_notes "
    tStrFrom = "FROM ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZ_BIOG_TEXT_DATA ON ZZ_SCRATCH_IMPORT_PEOPLE.c_personid = ZZ_BIOG_TEXT_DATA.c_personid"

    cmdSQL.CommandText = tStrInsert + tStrSelect + tStrFrom
    cmdSQL.Execute gTextRecCount

    ' the final step is to calculate the xy_count

    If gTextRecCount > 0 Then
        tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id, c_name_chn, c_name, c_sex, c_dy, c_dynasty, c_dynasty_chn, c_index_year, " + _
            "c_index_year_type_code, c_index_year_type_desc, c_index_year_type_hz, c_addr_id, c_addr_name, c_addr"

```

```

_chn, x_coord, y_coord ) " +
    "SELECT DISTINCT ZZ_SCRATCH_BIOG_TEXT_DATA.c_personid, ZZ_SCRATCH_BIOG_TEXT_DATA.c_name_chn, ZZ_SCRAT
CH_BIOG_TEXT_DATA.c_name, " +
    "ZZ_SCRATCH_BIOG_TEXT_DATA.c_sex, ZZ_SCRATCH_BIOG_TEXT_DATA.c_dy, ZZ_SCRATCH_BIOG_TEXT_DATA.c_dyn
asty, " +
    "ZZ_SCRATCH_BIOG_TEXT_DATA.c_dynasty_chn, ZZ_SCRATCH_BIOG_TEXT_DATA.c_index_year, ZZ_SCRATCH_BIOG
_TEXT_DATA.c_index_year_type_code, " +
    "ZZ_SCRATCH_BIOG_TEXT_DATA.c_index_year_type_desc, ZZ_SCRATCH_BIOG_TEXT_DATA.c_index_year_type_hz
, " +
    "ZZ_SCRATCH_BIOG_TEXT_DATA.c_addr_id, ZZ_SCRATCH_BIOG_TEXT_DATA.c_addr_name, ZZ_SCRATCH_BIOG_TEXT
_DATA.c_addr_chn, " +
    "ZZ_SCRATCH_BIOG_TEXT_DATA.x_coord, ZZ_SCRATCH_BIOG_TEXT_DATA.y_coord " +
    "FROM ZZ_SCRATCH_BIOG_TEXT_DATA"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount

'
' the final step is to calculate the xy_count
'
cmdSQL.CommandText = "Delete * from tmpXY"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " +
    "SELECT ZZ_SCRATCH_P_TEXT.x_coord, ZZ_SCRATCH_P_TEXT.y_coord, Count(ZZ_SCRATCH_P_TEXT.x_coord) " +
    "AS CountOfx_coord, Count(ZZ_SCRATCH_P_TEXT.y_coord) AS CountOfy_coord " +
    "FROM ZZ_SCRATCH_P_TEXT " +
    "GROUP BY ZZ_SCRATCH_P_TEXT.x_coord, ZZ_SCRATCH_P_TEXT.y_coord"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_P_TEXT ON (tmpXY.y_coord = " +
    "ZZ_SCRATCH_P_TEXT.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_P_TEXT.x_coord) " +
    "SET ZZ_SCRATCH_P_TEXT.xy_count = [tmpXY].[CountOfx_coord]"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount

Me.ChkGisText.Enabled = True
If Me.ChkGisText.Value = False Then
    gChkGisCount = gChkGisCount + 1
    Me.ChkGisText.Value = True
End If
CmdGIS.Enabled = True
Else
    If ChkGisText.Value Then
        Me.ChkGisText.Value = False
        gChkGisCount = gChkGisCount - 1
    End If
    Me.ChkGisText.Enabled = False

    If gChkGisCount = 0 Then
        CmdGIS.Enabled = False
    End If
End If

Set Me.Text.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_BIOG_TEXT_DATA", dbOpenDynaset)
End Sub

Private Sub queryAddr()
    Dim cmdSQL As ADODB.Command, tStrInsert As String, tStrSelect As String, tStrFrom As String, tRst As DAO.Reco
rdset

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    ' first clear ZZ_SCRATCH_STATUS

    Set Me.PLACE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_DUMMY_BA", dbOpenDynaset)

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_BIOG_ADDR_DATA"
    cmdSQL.Execute gPlaceRecCount

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_PEOPLE"
    cmdSQL.Execute gPlaceRecCount

    tStrInsert = "INSERT INTO ZZ_SCRATCH_BIOG_ADDR_DATA ( c_personid, c_name, c_name_chn, c_index_year, c_index_y
ear_type_code, " +
    "c_index_year_type_desc, c_index_year_type_hz, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_ad

```

```

dr_type, c_addr_desc, " +
    "c_addr_desc_chn, c_dy, c_dynasty, c_dynasty_chn, c_firstyear, c_lastyear, c_source, c_source_title, c_source_chn, c_pages, " +
    "c_notes, c_fy_nh_code, c_fy_nh_chn, c_fy_nh_py, c_ly_nh_code, c_ly_nh_chn, c_ly_nh_py, c_fy_nh_year, c_ly_nh_year, c_fy_range, " +
    "c_ly_range_desc, c_ly_range_chn, c_natal, c_fy_intercalary, c_ly_intercalary, c_fy_month, c_ly_month, c_fy_day, c_ly_day, " +
    "c_fy_day_gz, c_ly_day_gz_chn, c_fy_day_gz_py, c_ly_day_gz, c_ly_day_gz_chn, c_ly_day_gz_py, c_sequence )
"
tStrSelect = "SELECT ZBAD.c_personid, ZBAD.c_name, ZBAD.c_name_chn, ZBAD.c_index_year, ZBAD.c_index_year_type_code, " +
    "ZBAD.c_index_year_type_desc, ZBAD.c_index_year_type_hz, ZBAD.c_addr_id, ZBAD.c_addr_name, ZBAD.c_addr_chn, ZBAD.x_coord, " +
    "ZBAD.y_coord, ZBAD.c_addr_type, ZBAD.c_addr_desc, ZBAD.c_addr_desc_chn, ZBAD.c_dy, ZBAD.c_dynasty, ZBAD.c_dynasty_chn, " +
    "ZBAD.c_firstyear, ZBAD.c_lastyear, ZBAD.c_source, ZBAD.c_source_title, ZBAD.c_source_chn, ZBAD.c_pages, ZBAD.c_notes, " +
    "ZBAD.c_fy_nh_code, ZBAD.c_fy_nh_chn, ZBAD.c_fy_nh_py, ZBAD.c_ly_nh_code, ZBAD.c_ly_nh_chn, ZBAD.c_ly_nh_py, " +
    "ZBAD.c_fy_nh_year, ZBAD.c_ly_nh_year, ZBAD.c_fy_range, ZBAD.c_ly_range_desc, ZBAD.c_ly_range_chn, ZBAD.c_natal, " +
    "ZBAD.c_fy_intercalary, ZBAD.c_ly_intercalary, ZBAD.c_fy_month, ZBAD.c_ly_month, ZBAD.c_fy_day, ZBAD.c_ly_day, ZBAD.c_fy_day_gz, " +
    "ZBAD.c_ly_day_gz_chn, ZBAD.c_fy_day_gz_py, ZBAD.c_ly_day_gz, ZBAD.c_ly_day_gz_chn, ZBAD.c_ly_day_gz_py, ZBAD.c_sequence "
If Me.FrameQueryAddress.Value = 1 Then
    tStrFrom = "FROM ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZZ_BIOG_ADDR_DATA AS ZBAD ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZBAD.c_personid"
Else
    tStrFrom = "FROM (ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZZ_BIOG_ADDR_DATA AS ZBAD " +
        "ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZBAD.c_personid) INNER JOIN BIOG_MAIN " +
        "ON (ZBAD.c_addr_id = BIOG_MAIN.c_index_addr_id) AND (ZBAD.c_personid = BIOG_MAIN.c_personid)"
End If

cmdSQL.CommandText = tStrInsert + tStrSelect + tStrFrom
cmdSQL.Execute gPlaceRecCount
'
' the final step is to calculate the xy_count
'
If gPlaceRecCount > 0 Then
    tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_index_year_type_code, c_index_year_type_desc, " +
        "c_index_year_type_hz, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_addr_type, c_addr_desc, c_addr_desc_chn, c_dy, c_dynasty, c_dynasty_chn ) " +
        "SELECT DISTINCT ZZ_SCRATCH_BIOG_ADDR_DATA.c_personid, ZZ_SCRATCH_BIOG_ADDR_DATA.c_name, ZZ_SCRATCH_BIOG_ADDR_DATA.c_name_chn, " +
        "ZZ_SCRATCH_BIOG_ADDR_DATA.c_index_year, ZZ_SCRATCH_BIOG_ADDR_DATA.c_index_year_type_code, ZZ_SCRATCH_BIOG_ADDR_DATA.c_index_year_type_desc, " +
        "ZZ_SCRATCH_BIOG_ADDR_DATA.c_index_year_type_hz, ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_id, ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_name, " +
        "ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_chn, ZZ_SCRATCH_BIOG_ADDR_DATA.x_coord, ZZ_SCRATCH_BIOG_ADDR_DATA.y_coord, ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_type, " +
        "ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_desc, ZZ_SCRATCH_BIOG_ADDR_DATA.c_addr_desc_chn, ZZ_SCRATCH_BIOG_ADDR_DATA.c_dy, " +
        "ZZ_SCRATCH_BIOG_ADDR_DATA.c_dynasty, ZZ_SCRATCH_BIOG_ADDR_DATA.c_dynasty_chn " +
        "FROM ZZ_SCRATCH_BIOG_ADDR_DATA"
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = "Delete * from tmpXY"
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " +
        "SELECT ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord, Count(ZZ_SCRATCH_PEOPLE.x_coord) " +
        "AS CountOfx_coord, Count(ZZ_SCRATCH_PEOPLE.y_coord) AS CountOfy_coord " +
        "FROM ZZ_SCRATCH_PEOPLE " +
        "GROUP BY ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecCount
    '
    tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_PEOPLE ON (tmpXY.y_coord = " +
        "ZZ_SCRATCH_PEOPLE.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_PEOPLE.x_coord) " +
        "SET ZZ_SCRATCH_PEOPLE.xy_count = [tmpXY].[CountOfx_coord]"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecCount
    '
Me.ChkGisAddr.Enabled = True

```



```

        If Me.ChkGisAddr.Value = False Then
            gChkGisCount = gChkGisCount + 1
            Me.ChkGisAddr.Value = True
        End If
        CmdGIS.Enabled = True
    Else
        If ChkGisAddr.Value Then
            Me.ChkGisAddr.Value = False
            gChkGisCount = gChkGisCount - 1
        End If
        Me.ChkGisAddr.Enabled = False

        If gChkGisCount = 0 Then
            CmdGIS.Enabled = False
        End If
    End If

    Set Me.PLACE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_BIOG_ADDR_DATA", dbOpenDynaset)
End Sub

Private Sub WriteGIS_OfficeOffice()
On Error GoTo Err_WriteGIS_OfficeOffice
'
'   If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call WriteKML_Office
    Exit Sub
End If
'
'   This program will dump the results to a .gis file
'
If gOfficeRecCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_WriteGIS_OfficeOffice
End If
'
Dim tStream As ADODB.Stream, tPinyin As Boolean

Set tStream = New ADODB.Stream

tPinyin = False
If Me.GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
Else
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
'   next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "office_office_gis_" + tCodeStr + ".txt"
If dlgSaveAs.Show = -1 Then
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then

```

```

        MsgBox "Bad file Name."
        GoTo Exit_WriteGIS_OfficeOffice
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".txt"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".txt") Then
        tFileName = tFileName + ".tab"
    End If
End If
'
' write the file
'
'SELECT ZZ_SCRATCH_OFFICE.c_name AS Name, ZZ_SCRATCH_OFFICE.c_name_chn AS NameChn,
'ZZ_SCRATCH_OFFICE.c_index_year AS IndexYear, ZZ_SCRATCH_OFFICE.c_sex AS Sex,
'ZZ_SCRATCH_OFFICE.c_addr_name AS AddrName, ZZ_SCRATCH_OFFICE.c_addr_chn AS AddrChn,
'Str(ZZ_SCRATCH_OFFICE.x_coord) AS PersonX, Str(ZZ_SCRATCH_OFFICE.y_coord) AS PersonY,
'ZZ_SCRATCH_OFFICE.c_office_trans AS Office, ZZ_SCRATCH_OFFICE.c_office_chn AS OfficeChn,
'ZZ_SCRATCH_OFFICE.c_firstyear AS FirstYear, ZZ_SCRATCH_OFFICE.c_lastyear AS LastYear,
'ZZ_SCRATCH_OFFICE.c_dy_desc AS Dynasty,
'ZZ_SCRATCH_OFFICE.c_office_addr_name AS OfficeAddr,
'ZZ_SCRATCH_OFFICE.c_office_addr_chn AS OfficeAddrChn,
'Str(ZZ_SCRATCH_OFFICE.office_x_coord) AS X, Str(ZZ_SCRATCH_OFFICE.office_y_coord) AS Y,
'ZZ_SCRATCH_OFFICE.office_xy_count AS XY_count
'
' process the table
'
Set tRstNode = Me.OFFICE.Form.Recordset
tTab = Chr(9) ' the tab character
'
With tRstNode
    ' write the header
    '
    If tPinyin Then
        tStr = "Office" + tTab + "FirstYear" + tTab + "LastYear" + tTab + _
            "Dynasty" + tTab + "OfficeAddr" + tTab + _
            "X" + tTab + "Y" + tTab + "xy_count"
    Else
        tStr = "Office" + tTab + "OfficeChn" + tTab + "FirstYear" + tTab + "LastYear" + tTab + _
            "Dynasty" + tTab + "OfficeAddr" + tTab + "OfficeAddrChn" + tTab + _
            "X" + tTab + "Y" + tTab + "xy_count"
    End If

    tStream.WriteText tStr, adWriteLine

    .MoveFirst
    ' MsgBox "writing file"
    Do While Not .EOF
        ' must guard against NULLs
        '
        If IsNull(!c_office_trans) Then
            tStr = "No Translation" + tTab
        Else
            tStr = !c_office_trans + tTab
        End If

        If Not tPinyin Then
            If IsNull(!c_office_chn) Then
                tStr = tStr + "[?]" + tTab
            Else
                tStr = tStr + !c_office_chn + tTab
            End If
        End If

        If IsNull(!c_firstyear) Then
            tStr = tStr + "[?]" + tTab
        Else
            tStr = tStr + Str(!c_firstyear) + tTab
        End If

        If IsNull(!c_lastyear) Then
            tStr = tStr + "[?]" + tTab
        Else
            tStr = tStr + Str(!c_lastyear) + tTab
        End If

        If IsNull(!c_dynasty) Then
            tStr = tStr + "[?]" + tTab
        Else

```

```

        tStr = tStr + !c_dynasty + tTab
    End If

    If IsNull(!c_office_addr_name) Then
        tStr = tStr + "[?]" + tTab
    ElseIf Trim(!c_office_addr_name) = "" Then
        tStr = tStr + "[?]" + tTab
    Else
        tStr = tStr + !c_office_addr_name + tTab
    End If

    If Not tPinyin Then
        If IsNull(!c_office_addr_chn) Then
            tStr = tStr + "[?]" + tTab
        ElseIf Trim(!c_office_addr_chn) = "" Then
            tStr = tStr + "[?]" + tTab
        Else
            tStr = tStr + !c_office_addr_chn + tTab
        End If
    End If

    If IsNull(!office_x_coord) Then
        tStr = tStr + "0" + tTab
    Else
        tStr = tStr + Str(!office_x_coord) + tTab
    End If

    If IsNull(!office_y_coord) Then
        tStr = tStr + "0" + tTab
    Else
        tStr = tStr + Str(!office_y_coord) + tTab
    End If

    If IsNull(!office_xy_count) Then
        tStr = tStr + "0"
    Else
        tStr = tStr + Str(!office_xy_count)
    End If

    tStream.WriteText tStr, adWriteLine
    .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteGIS_OfficeOffice:
Exit Sub

Err_WriteGIS_OfficeOffice:
MsgBox Err.Description
Resume Exit_WriteGIS_OfficeOffice

End Sub

Private Sub WriteGIS_Status()
On Error GoTo Err_WriteGIS_Status
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call WriteKML_Status
    Exit Sub
End If

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String

```

```

Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tPinyin As Boolean
Dim tFileSystem, tGDF
'
' This program will dump the results to a .gis file
'
If gStatusRecCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_WriteGIS_Status
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False
If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 3 Then
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "status_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_WriteGIS_Status
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If
    '
    ' write the file
    '
    'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
    '
    ' process the table
    '
    Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_STATUS", dbOpenDynaset)
    tC = Chr(9) ' the tab
    '
    With tRstNode
        '
        ' write the header
        '
        If tPinyin Then
            tStr = "Name" + tC + "Sex" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "X" + tC + "Y" + tC + "xy_count"
        Else
            tStr = "Name" + tC + "NameChn" + tC + "Sex" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC + "xy_count"
        End If
        tStream.WriteText tStr, adWriteLine
        .MoveFirst
        Do While Not .EOF

```

```

' must guard against NULLs
,
If Trim(!c_name) = "" Then
    tStr = "[?]" + tC
Else
    tStr = !c_name + tC
End If

If Not tPinyin Then
    If Trim(!c_name_chn) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_name_chn + tC
    End If
End If

If IsNull(!c_sex) Then
    tStr = tStr + "[?]" + tC
Else
    tStr = tStr + !c_sex + tC
End If

If IsNull(!c_index_year) Then
    tStr = tStr + "-2000" + tC
Else
    tStr = tStr + Str(!c_index_year) + tC
End If

```

```

' here guard against blanks as well

```

```

If IsNull(!c_addr_name) Then
    tStr = tStr + "[?]" + tC
ElseIf Trim(!c_addr_name) = "" Then
    tStr = tStr + "[?]" + tC
Else
    tStr = tStr + !c_addr_name + tC
End If

If Not tPinyin Then
    If IsNull(!c_addr_chn) Then
        tStr = tStr + "[?]" + tC
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_addr_chn + tC
    End If
End If

```

```

If IsNull(!x_coord) Then
    tStr = tStr + "0" + tC
Else
    tStr = tStr + Str(!x_coord) + tC
End If

```

```

If IsNull(!y_coord) Then
    tStr = tStr + "0" + tC
Else
    tStr = tStr + Str(!y_coord) + tC
End If

```

```

If IsNull(!xy_count) Then
    tStr = tStr + "0"
Else
    tStr = tStr + Str(!xy_count)
End If

```

```

tStream.WriteText tStr, adWriteLine
.MoveNext

```

```

Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
,

```

```

Else
    'The user pressed Cancel.
End If

```

```

Set tRstNode = Nothing

```

```

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

```

```

Exit_WriteGIS_Status:
Exit Sub

```

```

Err_WriteGIS_Status:
MsgBox Err.Description
Resume Exit_WriteGIS_Status

```

```
End Sub
```

```
Private Sub WriteKML_Status()
```

```
On Error GoTo Err_WriteKML_Status
```

```

'
' This program will dump the results to a .gis file
'

```

```

If gStatusRecCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_WriteKML_Status
End If

```

```

'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

```

```

tPinyin = False
If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 3 Then
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True

```

```

Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If

```

```

tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

```

```

'
' next get a file
'

```

```
Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)
```

```
dlgSaveAs.InitialFileName = "status_gis_" + tCodeStr + ".kml"
```

```
If dlgSaveAs.Show = -1 Then
```

```

'
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
    tFileName = tFN
    If Not tFileName = "" Then
        Exit For
    End If
Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_WriteKML_Status
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".kml"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
        tFileName = tFileName + ".kml"
    End If
End If

```

```
End If
```

```

' write the file
'

```

```
'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count
```

```

' process the table
'

```

```
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_STATUS", dbOpenDynaset)
```

```
tC = Chr(9) ' the tab
```

```
tDQ = Chr(34) ' the double quotation mark
```

```

'

```

```

' write the header
,
tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
tStream.WriteText "<Document>", adWriteLine
tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
tStream.WriteText tC + "<open>1</open>", adWriteLine
tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

tStream.WriteText tC + "<Style id=" + tDQ + "status-balloon-template" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
tStream.WriteText tC + tC + tC + "<text>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "<![CDATA[", adWriteLine
tStream.WriteText tC + tC + tC + tC + "ID: ${StatusGIS/PersonID} <br/>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Name Chn: ${StatusGIS/NameHZ} <br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "Index Year: ${StatusGIS/IndexYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Sex: ${StatusGIS/Sex} <br/>", adWriteLine
If tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Address: ${StatusGIS/AddrName} <br/>", adWriteLine
Else
    tStream.WriteText tC + tC + tC + tC + "Address: ${StatusGIS/AddrName} ${StatusGIS/AddrHZ} <br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "XY Count: ${StatusGIS/XYCount} <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine
tStream.WriteText tC + "<!-- Declare the type " + tDQ + "StatusGIS" + tDQ + " with 6 fields -->", adWriteLine

tStream.WriteText tC + "<Schema name=" + tDQ + "StatusGIS" + tDQ + " id=" + tDQ + "StatusGISId" + tDQ + "
>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ
+ ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person ID]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "NameHZ" +
tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Name Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "Sex" + tDQ +
">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Sex]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + t
DQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrHZ" +
tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" +
tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
,
.MoveFirst
Do While Not .EOF
    ' must guard against NULLs, even where there should not be any
    ,
    ' write the point header
    ,
    tStream.WriteText tC + "<Placemark>", adWriteLine

    If IsNull(!c_name) Then
        tStr = "[Bad Data] "
    Else

```

```

        tStr = !c_name
    End If
    tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

    tStream.WriteText tC + tC + "<styleUrl>#status-balloon-template</styleUrl>", adWriteLine
    '
    ' First Year as time stamp
    '
    If IsNull(!c_index_year) Then
        tStr = "N/A"
    Else
        tStr = Str(!c_index_year)
    End If
    tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
    '
    tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
    tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#StatusGISId" + tDQ + ">", adW
riteLine
    '
    ' person ID
    '
    tStr = Str(!c_person_id)
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
    '
    ' Person Name Chn
    '
    If Not tPinyin Then
        If IsNull(!c_name_chn) Then
            tStr = tStr + "[Bad Data]"
        Else
            If Trim(!c_name_chn) = "" Then
                tStr = "[?]"
            Else
                tStr = !c_name_chn
            End If
        End If
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NameHZ" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
    '
    ' Index Year
    '
    If IsNull(!c_index_year) Then
        tStr = "N/A"
    Else
        tStr = Str(!c_index_year)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    If IsNull(!c_sex) Then
        tStr = "[?]"
    Else
        tStr = !c_sex
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "Sex" + tDQ + ">" + tStr + "</S
impleData>", adWriteLine
    '
    ' Address Name
    '
    If IsNull(!c_addr_name) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_name) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_name
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
    '
    ' Address Name Chinese
    '
    If Not tPinyin Then
        If IsNull(!c_addr_chn) Then
            tStr = "[?]"
        ElseIf Trim(!c_addr_chn) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_addr_chn
        End If
    End If

```



```

        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrHZ" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    End If
    '
    ' XY Count
    '
    If IsNull(!xy_count) Then
        tStr = "0"
    Else
        tStr = Str(!xy_count)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
    tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
    tStream.WriteText tC + tC + "<Point>", adWriteLine
    '
    ' coordinates
    If IsNull(!x_coord) Then
        tStr = "0"
    Else
        tStr = Str(!x_coord)
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + ",0"
    Else
        tStr = tStr + "," + Str(!y_coord)
    End If
    tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
    '
    ' footer
    '
    tStream.WriteText tC + tC + "</Point>", adWriteLine
    tStream.WriteText tC + tC + "</Placemark>", adWriteLine
    .MoveNext
    Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteKML_Status:
Exit Sub

Err_WriteKML_Status:
MsgBox Err.Description
Resume Exit_WriteKML_Status

End Sub
Private Sub WriteGIS_OfficePeople()
On Error GoTo Err_WriteGIS_OfficePeople
Dim tPinyin As Boolean
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call WriteKML_OfficePeople
    Exit Sub
End If

```

```

' This program will dump the results to a .gis file
'
If gOfficeRecCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_WriteGIS_OfficePeople
End If
'
tPinyin = False
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
Else
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "office_people_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_WriteGIS_OfficePeople
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If
    '
    ' write the file
    '
    Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_OFFICE", dbOpenDynaset)
    tTab = Chr(9) ' the tab character
    '
    With tRstNode
        ' write the header
        '
        If tPinyin Then
            tStr = "Name" + tTab + "Sex" + tTab + "IndexYear" + tTab + _
                "AddrID" + tTab + "AddrName" + tTab + _
                "X" + tTab + "Y" + tTab + "xy_count"
        Else
            tStr = "Name" + tTab + "NameChn" + tTab + "Sex" + tTab + "IndexYear" + tTab + _
                "AddrID" + tTab + "AddrName" + tTab + "AddrChn" + tTab + _
                "X" + tTab + "Y" + tTab + "xy_count"
        End If
        tStream.WriteText tStr, adWriteLine
    End With

```

```

.MoveFirst
Do While Not .EOF
    ' must guard against NULLs
    '
    If Trim(!c_name) = "" Then
        tStr = "[?]" + tTab
    Else
        tStr = !c_name + tTab
    End If

    If Not tPinyin Then
        If Trim(!c_name_chn) = "" Then
            tStr = tStr + "[?]" + tTab
        Else
            tStr = tStr + !c_name_chn + tTab
        End If
    End If

    'If IsNull(!c_sex) Then
    '    tStr = tStr + "?" + tTab
    'Else
    '    tStr = tStr + !c_sex + tTab
    'End If
    tStr = tStr + IIf(!c_female, "F", "M") + tTab

    If IsNull(!c_index_year) Then
        tStr = tStr + "-2000" + tTab
    Else
        tStr = tStr + Str(!c_index_year) + tTab
    End If

    ' here guard against blanks as well

    If IsNull(!c_addr_id) Then
        tStr = tStr + "0" + tTab
    Else
        tStr = tStr + Str(!c_addr_id) + tTab
    End If

    If IsNull(!c_addr_name) Then
        tStr = tStr + "[?]" + tTab
    ElseIf Trim(!c_addr_name) = "" Then
        tStr = tStr + "[?]" + tTab
    Else
        tStr = tStr + !c_addr_name + tTab
    End If

    If Not tPinyin Then
        If IsNull(!c_addr_chn) Then
            tStr = tStr + "[?]" + tTab
        ElseIf Trim(!c_addr_chn) = "" Then
            tStr = tStr + "[?]" + tTab
        Else
            tStr = tStr + !c_addr_chn + tTab
        End If
    End If

    If IsNull(!x_coord) Then
        tStr = tStr + "0" + tTab
    Else
        tStr = tStr + Str(!x_coord) + tTab
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + "0" + tTab
    Else
        tStr = tStr + Str(!y_coord) + tTab
    End If

    If IsNull(!xy_count) Then
        tStr = tStr + "0"
    Else
        tStr = tStr + Str(!xy_count)
    End If

    tStream.WriteText tStr, adWriteLine
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream

```

```

        tStream.Flush
        ' and write the stream to the file
        tStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
Else
    'The user pressed Cancel.
End If

Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteGIS_OfficePeople:
Exit Sub

Err_WriteGIS_OfficePeople:
MsgBox Err.Description
Resume Exit_WriteGIS_OfficePeople

End Sub

Private Sub WriteKML_OfficePeople()

    Dim tStrKML As String, tPinyin As Boolean
    '
    ' This program will dump the results to a .kml file
    '
    If gOfficeRecCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_WriteKML_OfficePeople
    End If
    '
    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream

    tPinyin = False
    If GISFrame.Value = 1 Then
        tStream.Charset = "utf-8"
        tCodeStr = "UTF8"
    ElseIf GISFrame.Value = 2 Then
        tStream.Charset = "gb18030"
        tCodeStr = "GB18030"
    Else
        tStream.Charset = "ascii"
        tCodeStr = "ASCII"
        tPinyin = True
    End If
    tStream.Mode = adModeReadWrite
    tStream.Type = adTypeText
    tStream.Open
    '
    ' next get a file
    '
    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant, tFemale As String
    Dim tRstNode As DAO.Recordset
    Dim tStr As String, tTab As String, ti As Integer
    Dim tFileSystem, tGDF

    Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    dlgSaveAs.InitialFileName = "office_people_gis_" + tCodeStr + ".kml"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_WriteKML_OfficePeople
        Else
            ' make sure the file name has a txt extension

```

```

    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".kml"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
        tFileName = tFileName + ".kml"
    End If
End If
'
' write the file
'
'SELECT ZZ_SCRATCH_OFFICE.c_name AS Name, ZZ_SCRATCH_OFFICE.c_name_chn AS NameChn,
'ZZ_SCRATCH_OFFICE.c_index_year AS IndexYear, ZZ_SCRATCH_OFFICE.c_sex AS Sex,
'ZZ_SCRATCH_OFFICE.c_addr_name AS AddrName, ZZ_SCRATCH_OFFICE.c_addr_chn AS AddrChn,
'Str(ZZ_SCRATCH_OFFICE.x_coord) AS PersonX, Str(ZZ_SCRATCH_OFFICE.y_coord) AS PersonY,
'ZZ_SCRATCH_OFFICE.c_office_trans AS Office, ZZ_SCRATCH_OFFICE.c_office_chn AS OfficeChn,
'ZZ_SCRATCH_OFFICE.c_firstyear AS FirstYear, ZZ_SCRATCH_OFFICE.c_lastyear AS LastYear,
'ZZ_SCRATCH_OFFICE.c_dy_desc AS Dynasty,
'ZZ_SCRATCH_OFFICE.c_office_addr_name AS OfficeAddr,
'ZZ_SCRATCH_OFFICE.c_office_addr_chn AS OfficeAddrChn,
'Str(ZZ_SCRATCH_OFFICE.office_x_coord) AS X, Str(ZZ_SCRATCH_OFFICE.office_y_coord) AS Y,
'ZZ_SCRATCH_OFFICE.office_xy_count AS XY_count

'
' tStr = "PostingID" (c_posting_id) + "Office" (c_name) + "OfficeChn" (c_name_chn) + _
' "FirstYear" (c_firstyear) + "LastYear" + (c_lastyear) + _
' "Dynasty" (c_dy) + "OfficeAddr" (c_office_addr_name) + "OfficeAddrHZ" (c_office_addr_chn) + _
' "X" (office_x_coord) + "Y" (office_y_coord) + "xy_count" (office_xy_count)
'
' process the table
'
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_OFFICE", dbOpenDynaset)
tC = Chr(9) ' the tab
tDQ = Chr(34) ' the double quotation mark
'
' write the header
'
tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
tStream.WriteText "<Document>", adWriteLine
tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
tStream.WriteText tC + "<open>1</open>", adWriteLine
tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

tStream.WriteText tC + "<Style id=" + tDQ + "office-balloon-template" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
tStream.WriteText tC + tC + tC + "<text>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "<![CDATA[" + tC + "ID: ${OfficePosting/PersonID} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Name Chn: ${OfficePosting/NameHZ} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Index Year: ${OfficePosting/IndexYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Dynasty: ${OfficePosting/Dyn} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Address: ${OfficePosting/AddrName} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Address: ${OfficePosting/AddrName} ${OfficePosting/AddrNameHZ} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "XY Count: ${OfficePosting/XYCount} <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine
tStream.WriteText tC + "<!-- Declare the type " + tDQ + "OfficePosting" + tDQ + " with 6 fields -->", adWriteLine

tStream.WriteText tC + "<Schema name=" + tDQ + "OfficePosting" + tDQ + " id=" + tDQ + "OfficePersonID" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[ID]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "NameHZ" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person Chn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "</Schema>", adWriteLine

```

```

        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrNameH
Z" + tDQ + ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    End If
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" +
tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "Dyn" + tDQ + ">"
, adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Dyn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs, even where there should not be any
        '
        ' write the point header
        '
        tStream.WriteText tC + "<Placemark>", adWriteLine

        If IsNull(!c_name) Then
            tStr = "[Bad Data] " + Str(!c_personid)
        Else
            tStr = !c_name + " " + Str(!c_personid)
        End If
        tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

        tStream.WriteText tC + tC + "<styleUrl>#office-balloon-template</styleUrl>", adWriteLine
        '
        ' First Year as time stamp
        '
        If IsNull(!c_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_index_year)
        End If
        tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
        '
        tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#OfficePersonID" + tDQ + ">",
adWriteLine
        '
        ' person ID
        '
        tStr = Str(!c_personid)
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
        '
        ' Person Name Chn
        '
        If Not tPinyin Then
            If IsNull(!c_name_chn) Then
                tStr = tStr + "[Bad Data]"
            Else
                If Trim(!c_name_chn) = "" Then
                    tStr = "[?]"
                Else
                    tStr = !c_name_chn
                End If
            End If
            tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NameHZ" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
        End If
        '
        ' Index Year
        '
        If IsNull(!c_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_index_year)
        End If
    End While
End With

```

```

        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    ' Dynasty
    '
    If IsNull(!c_dy) Then
        tStr = "0"
    Else
        tStr = Str(!c_dy)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "Dyn" + tDQ + ">" + tStr + "</S
impleData>", adWriteLine
    '
    ' Address Name
    '
    If IsNull(!c_addr_name) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_name) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_name
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    ' Address Name Chinese
    '
    If Not tPinyin Then
        If IsNull(!c_addr_chn) Then
            tStr = "[?]"
        ElseIf Trim(!c_addr_chn) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_addr_chn
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrNameHZ" + tDQ + ">" +
tStr + "</SimpleData>", adWriteLine
    End If
    '
    ' XY Count
    '
    If IsNull(!xy_count) Then
        tStr = "0"
    Else
        tStr = Str(!xy_count)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
    tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
    tStream.WriteText tC + tC + "<Point>", adWriteLine
    '
    ' coordinates
    If IsNull(!x_coord) Then
        tStr = "0"
    Else
        tStr = Str(!x_coord)
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + ",0"
    Else
        tStr = tStr + "," + Str(!y_coord)
    End If
    tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
    '
    ' footer
    '
    tStream.WriteText tC + tC + "</Point>", adWriteLine
    tStream.WriteText tC + tC + "</Placemark>", adWriteLine
    .MoveNext
Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else

```

```

        'The user pressed Cancel.
    End If

    ' now make sure all the data is copied to tStream
    tStream.Flush
    ' and write the stream to the file
    tStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    Set tRstNode = Nothing

    tStream.Close
    Set tStream = Nothing
    'Set the object variable to Nothing.
    Set dlgSaveAs = Nothing

Exit_WriteKML_OfficePeople:
    Exit Sub

Err_WriteKML_OfficePeople:
    MsgBox Err.Description
    Resume Exit_WriteKML_OfficePeople

End Sub

Private Sub WriteKML_Entry()
    '<kml xmlns="http://www.opengis.net/kml/2.2">
    '<Document>
    '  <name>ExtendedData+SchemaData</name>
    '  <open>1</open>
    '  <!-- Create a balloon template referring to the user-defined type -->
    '  <Style id="assoc-balloon-template">
    '    <BalloonStyle>
    '      <text>
    '        <![CDATA[
    '          $[AssocPerson/PersonNameHZ] <br/>
    '          ID: $[AssocPerson/PersonID] <br/>
    '          Index Year: $[AssocPerson/IndexYear] <br/>
    '          Address: $[AssocPerson/AddrName] $[AssocPerson/AddrNameHZ] <br/>
    '          XY Count: $[AssocPerson/XYCount] <br/><br/>
    '        ]]>
    '      </text>
    '    </BalloonStyle>
    '  </Style>
    '  <!-- Declare the type "AssocPerson" with 6 fields -->
    '  <Schema name="AssocPerson" id="AssocPersonId">
    '    <SimpleField type="string" name="PersonNameHZ">
    '      <displayName><![CDATA[<b>Person</b>]]></displayName>
    '    </SimpleField>
    '    <SimpleField type="string" name="AddrName">
    '      <displayName><![CDATA[<b>Person</b>]]></displayName>
    '    </SimpleField>
    '    <SimpleField type="string" name="AddrNameHZ">
    '      <displayName><![CDATA[<b>Person</b>]]></displayName>
    '    </SimpleField>
    '    <SimpleField type="uint" name="PersonID">
    '      <displayName><![CDATA[ID]]></displayName>
    '    </SimpleField>
    '    <SimpleField type="int" name="IndexYear">
    '      <displayName><![CDATA[Index Year]]></displayName>
    '    </SimpleField>
    '    <SimpleField type="int" name="XYCount">
    '      <displayName><![CDATA[XY Count]]></displayName>
    '    </SimpleField>
    '  </Schema>
    '  <!-- Instantiate some Placemarks extended with AssocPerson fields -->
    '  <Placemark>
    '    <name>Easy trail</name>
    '    <styleUrl>#assoc-balloon-template</styleUrl>
    '    <ExtendedData>
    '      <SchemaData schemaUrl="#AssocPersonId">
    '        <SimpleData name="PersonID">3.14159</SimpleData>
    '        <SimpleData name="PersonNameHZ">Pi in the sky</SimpleData>
    '        <SimpleData name="IndexYear">10</SimpleData>
    '        <SimpleData name="AddrName">Pi in the sky</SimpleData>
    '        <SimpleData name="AddrNameHZ">Pi in the sky</SimpleData>
    '        <SimpleData name="XYCount">10</SimpleData>
    '      </SchemaData>
    '    </ExtendedData>
    '    <Point>
    '      <coordinates>-122.000,37.002</coordinates>

```



```

'      </Point>
'    </Placemark>
'  <Placemark>
'    <name>Difficult trail</name>
'    <styleUrl>#assoc-balloon-template</styleUrl>
'    <ExtendedData>
'      <SchemaData schemaUrl="#AssocPersonId">
'        <SimpleData name="TrailHeadName">Mount Everest</SimpleData>
'        <SimpleData name="TrailLength">347.45</SimpleData>
'        <SimpleData name="ElevationGain">10000</SimpleData>
'      </SchemaData>
'    </ExtendedData>
'    <Point>
'      <coordinates>-121.998,37.0078</coordinates>
'    </Point>
'  </Placemark>
</Document>
</kml>

```

```

Dim tStrKML As String, tPinyin As Boolean
'
' This program will dump the results to a .gis file
'
If gEntryRecCount = 0 Then
  MsgBox "There are no records to save."
  GoTo Exit_WriteKML_Entry
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False
If GISFrame.Value = 1 Then
  tStream.Charset = "utf-8"
  tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
  tStream.Charset = "gb2312"
  tCodeStr = "GB2312"
Else
  tStream.Charset = "ascii"
  tCodeStr = "ASCII"
  tPinyin = True
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, tDQ As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "entry_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
  '
  tFileName = ""
  For Each tFN In dlgSaveAs.SelectedItems
    tFileName = tFN
    If Not tFileName = "" Then
      Exit For
    End If
  Next
  If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_WriteKML_Entry
  Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
      tFileName = tFileName + ".kml"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
      tFileName = tFileName + ".kml"
    End If
  End If
End If
'

```

```

' write the file
'
'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
'Name,NameChn,IndexYear,EntryDesc,EntryChn,EntryYear,
'AddrName,AddrChn,X,Y,xy_count
'
' process the table
'
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
tC = Chr(9) ' the tab
tDQ = Chr(34) ' the double quotation mark
'
' write the header
'
tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
tStream.WriteText "<Document>", adWriteLine
tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
tStream.WriteText tC + "<open>1</open>", adWriteLine
tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

tStream.WriteText tC + "<Style id=" + tDQ + "entry-balloon-template" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
tStream.WriteText tC + tC + tC + "<text>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "<![CDATA["", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "$[EntryPerson/PersonNameHZ] <br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "ID: $[EntryPerson/PersonID] <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Index Year: $[EntryPerson/IndexYear] <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Entry Year: $[EntryPerson/EntryYear] <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Entry Desc: $[EntryPerson/EntryDesc] <br/>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Entry Chn: $[EntryPerson/EntryDescHZ] <br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "Entry Rank: $[EntryPerson/EntryRank] <br/>", adWriteLine
If tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Address: $[EntryPerson/AddrName] <br/>", adWriteLine
Else
    tStream.WriteText tC + tC + tC + tC + "Address: $[EntryPerson/AddrName] $[EntryPerson/AddrNameHZ] <br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "XY Count: $[EntryPerson/XYCount] <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine
tStream.WriteText tC + "<!-- Declare the type " + tDQ + "EntryPerson" + tDQ + " with 10 fields -->", adWriteLine

tStream.WriteText tC + "<Schema name=" + tDQ + "EntryPerson" + tDQ + " id=" + tDQ + "EntryPersonId" + tDQ + ">", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "PersonNameHZ" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[<b>Person</b>]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[<b>Person</b>]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrNameHZ" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[<b>Person</b>]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[ID]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "EntryYear" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Entry Year]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "EntryDesc" + tDQ + ">", adWriteLine

```

```

tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Entry Desc]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "EntryDesc" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Entry Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "EntryRank" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Entry Rank]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs, even where there should not be any
        '
        ' write the point header
        '
        tStream.WriteText tC + "<Placemark>", adWriteLine

        If IsNull(!c_name) Then
            tStr = "[Bad Data]"
        Else
            tStr = !c_name
        End If
        tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

        tStream.WriteText tC + tC + "<styleUrl>#entry-balloon-template</styleUrl>", adWriteLine
        '
        ' Index Year as time stamp
        '
        If IsNull(!c_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_index_year)
        End If
        tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
        '
        tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#EntryPersonId" + tDQ + ">", adWriteLine
        '
        ' person ID
        '
        tStr = Str(!c_personid)
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr + "</SimpleData>", adWriteLine
        '
        ' Chinese Name
        '
        If Not tPinyin Then
            If IsNull(!c_name_chn) Then
                tStr = tStr + "[Bad Data]"
            Else
                If Trim(!c_name_chn) = "" Then
                    tStr = "[?]"
                Else
                    tStr = !c_name_chn
                End If
            End If
            tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonNameHZ" + tDQ + ">" + tStr + "</SimpleData>", adWriteLine
        End If
        '
        ' Index Year
        '
        If IsNull(!c_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_index_year)
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr

```

```

+ "</SimpleData>", adWriteLine
    '
    ' Entry Year
    '
    If IsNull(!c_year) Then
        tStr = "-2000"
    Else
        tStr = Str(!c_year)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "EntryYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    ' Entry Desc
    '
    If IsNull(!c_entry_desc) Then
        tStr = "[Missing Data]"
    Else
        tStr = !c_entry_desc
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "EntryDesc" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    ' Entry Chn
    '
    If Not tPinyin Then
        If IsNull(!c_entry_chn) Then
            tStr = "[Missing Data]"
        Else
            tStr = !c_entry_chn
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "EntryDescHZ" + tDQ + ">" +
tStr + "</SimpleData>", adWriteLine
    End If
    '
    ' Entry Rank
    '
    If IsNull(!c_exam_rank) Then
        tStr = "0"
    Else
        tStr = !c_exam_rank
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "EntryRank" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    ' Address Name
    '
    If IsNull(!c_addr_name) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_name) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_name
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    ' Address Name Chinese
    '
    If Not tPinyin Then
        If IsNull(!c_addr_chn) Then
            tStr = "[?]"
        ElseIf Trim(!c_addr_chn) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_addr_chn
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrNameHZ" + tDQ + ">" +
tStr + "</SimpleData>", adWriteLine
    End If
    '
    ' XY Count
    '
    If IsNull(!xy_count) Then
        tStr = "0"
    Else
        tStr = Str(!xy_count)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '

```

```

        tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
        tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + "<Point>", adWriteLine
        '
        ' coordinates
        If IsNull(!x_coord) Then
            tStr = "0"
        Else
            tStr = Str(!x_coord)
        End If

        If IsNull(!y_coord) Then
            tStr = tStr + ",0"
        Else
            tStr = tStr + "," + Str(!y_coord)
        End If
        tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
        '
        ' footer
        '
        tStream.WriteText tC + tC + "</Point>", adWriteLine
        tStream.WriteText tC + tC + "</Placemark>", adWriteLine
        .MoveNext
    Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteKML_Entry:
Exit Sub

Err_WriteKML_Entry:
MsgBox Err.Description
Resume Exit_WriteKML_Entry

End Sub

Private Sub WriteGIS_Entry()
On Error GoTo Err_WriteGIS_Entry
Dim tPinyin As Boolean
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call WriteKML_Entry
    Exit Sub
End If
'
' This program will dump the results to a .gis file
'
If gEntryRecCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_WriteGIS_Entry
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False
If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"

```

```

ElseIf GISFrame.Value = 2 Then
    tStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "entry_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_WriteGIS_Entry
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If
    '
    ' write the file
    '
    'Name,NameChn,PersonID,IndexYear,EntryDesc,EntryChn,EntryYear,EntryRank,KinType,KinName,KinChn,
    'AddrName,AddrChn,X,Y,xy_count
    '
    ' process the table
    '
    Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_ENTRY", dbOpenDynaset)
    tTab = Chr(9) ' the tab character
    '
    With tRstNode
        ' write the header
        '
        If tPinyin Then
            tStr = "Name" + tTab + "PersonID" + tTab + "IndexYear" + tTab + _
                "EntryDesc" + tTab + "EntryYear" + tTab + "EntryRank" + tTab + _
                "KinType" + tTab + "KinName" + tTab + _
                "AddrName" + tTab + "X" + tTab + "Y" + tTab + "xy_count"
        Else
            tStr = "Name" + tTab + "NameChn" + tTab + "PersonID" + tTab + "IndexYear" + tTab + _
                "EntryDesc" + tTab + "EntryChn" + tTab + "EntryYear" + tTab + "EntryRank" + tTab + _
                "KinType" + tTab + "KinName" + tTab + "KinChn" + tTab + _
                "AddrName" + tTab + "AddrChn" + tTab + "X" + tTab + "Y" + tTab + "xy_count"
        End If
        tStream.WriteText tStr, adWriteLine
        .MoveFirst
        Do While Not .EOF
            ' must guard against NULLs
            '
            If Trim(!c_name) = "" Then
                tStr = "[?]" + tTab
            Else
                tStr = !c_name + tTab
            End If

```

```

If Not tPinyin Then
    If Trim(!c_name_chn) = "" Then
        tStr = tStr + "[?]" + tTab
    Else
        tStr = tStr + !c_name_chn + tTab
    End If
End If

tStr = tStr + Trim(Str(!c_personid)) + tTab

If IsNull(!c_index_year) Then
    tStr = tStr + "-2000" + tTab
Else
    tStr = tStr + Str(!c_index_year) + tTab
End If

If IsNull(!c_entry_desc) Then
    tStr = tStr + "[?]" + tTab
ElseIf Trim(!c_entry_desc) = "" Then
    tStr = tStr + "[?]" + tTab
Else
    tStr = tStr + !c_entry_desc + tTab
End If

If Not tPinyin Then
    If IsNull(!c_entry_chn) Then
        tStr = tStr + "[?]" + tTab
    ElseIf Trim(!c_entry_chn) = "" Then
        tStr = tStr + "[?]" + tTab
    Else
        tStr = tStr + !c_entry_chn + tTab
    End If
End If

If IsNull(!c_year) Then
    tStr = tStr + "[?]" + tTab
Else
    tStr = tStr + Trim(Str(!c_year)) + tTab
End If

If IsNull(!c_exam_rank) Then
    tStr = tStr + "[?]" + tTab
ElseIf Trim(!c_exam_rank) = "" Then
    tStr = tStr + "[?]" + tTab
Else
    tStr = tStr + !c_exam_rank + tTab
End If

If IsNull(!c_kin_desc) Then
    tStr = tStr + "[?]" + tTab
ElseIf Trim(!c_kin_desc) = "" Then
    tStr = tStr + "[?]" + tTab
Else
    tStr = tStr + !c_kin_desc + tTab
End If

If IsNull(!c_kin_name) Then
    tStr = tStr + "[?]" + tTab
ElseIf Trim(!c_kin_name) = "" Then
    tStr = tStr + "[?]" + tTab
Else
    tStr = tStr + !c_kin_name + tTab
End If

If Not tPinyin Then
    If IsNull(!c_kin_chn) Then
        tStr = tStr + "[?]" + tTab
    ElseIf Trim(!c_kin_chn) = "" Then
        tStr = tStr + "[?]" + tTab
    Else
        tStr = tStr + !c_kin_chn + tTab
    End If
End If

' here guard against blanks as well

If IsNull(!c_addr_name) Then
    tStr = tStr + "[?]" + tTab
ElseIf Trim(!c_addr_name) = "" Then

```

```

        tStr = tStr + "[?]" + tTab
    Else
        tStr = tStr + !c_addr_name + tTab
    End If

    If Not tPinyin Then
        If IsNull(!c_addr_chn) Then
            tStr = tStr + "[?]" + tTab
        ElseIf Trim(!c_addr_chn) = "" Then
            tStr = tStr + "[?]" + tTab
        Else
            tStr = tStr + !c_addr_chn + tTab
        End If
    End If

    If IsNull(!x_coord) Then
        tStr = tStr + "0" + tTab
    Else
        tStr = tStr + Str(!x_coord) + tTab
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + "0" + tTab
    Else
        tStr = tStr + Str(!y_coord) + tTab
    End If

    If IsNull(!xy_count) Then
        tStr = tStr + "0"
    Else
        tStr = tStr + Str(!xy_count)
    End If

    tStream.WriteText tStr, adWriteLine
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteGIS_Entry:
Exit Sub

Err_WriteGIS_Entry:
MsgBox Err.Description
Resume Exit_WriteGIS_Entry
End Sub

Private Sub WriteKML_Text()
On Error GoTo Err_WriteKML_Text
'
' This program will dump the results to a .gis file
'
If gTextRecCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_WriteKML_Text
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False
If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    tStream.Charset = "gb18030"

```



```

        tCodeStr = "GB18030"
    Else
        tStream.Charset = "ascii"
        tCodeStr = "ASCII"
        tPinyin = True
    End If
    tStream.Mode = adModeReadWrite
    tStream.Type = adTypeText
    tStream.Open
'
' next get a file
'

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "text_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
'
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_WriteKML_Text
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".kml"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
            tFileName = tFileName + ".kml"
        End If
    End If
'
' write the file
'
'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count
'
' process the table
'

Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_TEXT", dbOpenDynaset)
tC = Chr(9) ' the tab
tDQ = Chr(34) ' the double quotation mark
'
' write the header
'
tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
tStream.WriteText "<Document>", adWriteLine
tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
tStream.WriteText tC + "<open>1</open>", adWriteLine
tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

tStream.WriteText tC + "<Style id=" + tDQ + "TextCat-balloon-template" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
tStream.WriteText tC + tC + tC + "<text>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "<![CDATA[" , adWriteLine
tStream.WriteText tC + tC + tC + tC + "ID: ${TextCatGIS/PersonID} <br/>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Name Chn: ${TextCatGIS/NameHZ} <br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "Index Year: ${TextCatGIS/IndexYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Sex: ${TextCatGIS/Sex} <br/>", adWriteLine
If tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Address: ${TextCatGIS/AddrName} <br/>", adWriteLine
Else
    tStream.WriteText tC + tC + tC + tC + "Address: ${TextCatGIS/AddrName} ${TextCatGIS/AddrHZ} <br/>", a
adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "XY Count: ${TextCatGIS/XYCount} <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine
tStream.WriteText tC + "<!-- Declare the type " + tDQ + "TextCatGIS" + tDQ + " with 6 fields -->", adWriteLine
'
tStream.WriteText tC + "<Schema name=" + tDQ + "TextCatGIS" + tDQ + " id=" + tDQ + "TextCatGISId" + tDQ +
">", adWriteLine

```

```

        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ
+ ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person ID]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        If Not tPinyin Then
            tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "NameHZ" +
tDQ + ">", adWriteLine
            tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Name Chn]]></displayName>", adWriteLine
            tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        End If
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "Sex" + tDQ +
">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Sex]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + t
DQ + ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        If Not tPinyin Then
            tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrHZ" +
tDQ + ">", adWriteLine
            tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
            tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        End If
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" +
tDQ + ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + "</Schema>", adWriteLine

        With tRstNode
            .MoveFirst
            Do While Not .EOF
                ' must guard against NULLs, even where there should not be any
                '
                ' write the point header
                '
                tStream.WriteText tC + "<Placemark>", adWriteLine

                If IsNull(!c_name) Then
                    tStr = "[Bad Data] "
                Else
                    tStr = !c_name
                End If
                tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

                tStream.WriteText tC + tC + "<styleUrl>#TextCat-balloon-template</styleUrl>", adWriteLine
                '
                ' First Year as time stamp
                '
                If IsNull(!c_index_year) Then
                    tStr = "N/A"
                Else
                    tStr = Str(!c_index_year)
                End If
                tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
                '
                tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
                tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#TextCatGISId" + tDQ + ">", ad
WriteLine
                '
                ' person ID
                '
                tStr = Str(!c_person_id)
                tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
                '
                ' Person Name Chn
                '
                If Not tPinyin Then
                    If IsNull(!c_name_chn) Then
                        tStr = tStr + "[Bad Data]"
                    Else
                        If Trim(!c_name_chn) = "" Then
                            tStr = "[?]"

```

```

        Else
            tStr = !c_name_chn
        End If
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NameHZ" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
End If
'
' Index Year
'
If IsNull(!c_index_year) Then
    tStr = "N/A"
Else
    tStr = Str(!c_index_year)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
If IsNull(!c_sex) Then
    tStr = "[?]"
Else
    tStr = !c_sex
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "Sex" + tDQ + ">" + tStr + "</S
impleData>", adWriteLine
'
' Address Name
'
If IsNull(!c_addr_name) Then
    tStr = "[?]"
ElseIf Trim(!c_addr_name) = "" Then
    tStr = "[?]"
Else
    tStr = !c_addr_name
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
' Address Name Chinese
'
If Not tPinyin Then
    If IsNull(!c_addr_chn) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_chn
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrHZ" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
End If
'
' XY Count
'
If IsNull(!xy_count) Then
    tStr = "0"
Else
    tStr = Str(!xy_count)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
tStream.WriteText tC + tC + "<Point>", adWriteLine
'
' coordinates
If IsNull(!x_coord) Then
    tStr = "0"
Else
    tStr = Str(!x_coord)
End If

If IsNull(!y_coord) Then
    tStr = tStr + ",0"
Else
    tStr = tStr + "," + Str(!y_coord)
End If
tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
'

```

```

        ' footer
        '
        tStream.WriteText tC + tC + "</Point>", adWriteLine
        tStream.WriteText tC + "</Placemark>", adWriteLine
        .MoveNext
    Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteKML_Text:
Exit Sub

Err_WriteKML_Text:
MsgBox Err.Description
Resume Exit_WriteKML_Text

End Sub

Private Sub WriteGIS_Text()
On Error GoTo Err_WriteGIS_Text
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call WriteKML_Text
    Exit Sub
End If

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tPinyin As Boolean
Dim tFileSystem, tGDF
'
' This program will dump the results to a .gis file
'
If gTextRecCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_WriteGIS_Text
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False
If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
Else
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'

```

```

' next get a file
'
Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "text_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_WriteGIS_Text
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If
    ' write the file
    '
    'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
    '
    ' process the table
    '
    Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_TEXT", dbOpenDynaset)
    tC = Chr(9) ' the tab
    '
    With tRstNode
        ' write the header
        '
        If tPinyin Then
            tStr = "Name" + tC + "Sex" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "X" + tC + "Y" + tC + "xy_count"
        Else
            tStr = "Name" + tC + "NameChn" + tC + "Sex" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC + "xy_count"
        End If
        tStream.WriteText tStr, adWriteLine
        .MoveFirst
        Do While Not .EOF
            ' must guard against NULLs
            '
            If Trim(!c_name) = "" Then
                tStr = "[?]" + tC
            Else
                tStr = !c_name + tC
            End If

            If Not tPinyin Then
                If Trim(!c_name_chn) = "" Then
                    tStr = tStr + "[?]" + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If
            End If

            If IsNull(!c_sex) Then
                tStr = tStr + "[?]" + tC
            Else
                tStr = tStr + !c_sex + tC
            End If

            If IsNull(!c_index_year) Then
                tStr = tStr + "-2000" + tC
            Else
                tStr = tStr + Str(!c_index_year) + tC
            End If

            ' here guard against blanks as well

```

```

        If IsNull(!c_addr_name) Then
            tStr = tStr + "[?]" + tC
        ElseIf Trim(!c_addr_name) = "" Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_addr_name + tC
        End If

        If Not tPinyin Then
            If IsNull(!c_addr_chn) Then
                tStr = tStr + "[?]" + tC
            ElseIf Trim(!c_addr_chn) = "" Then
                tStr = tStr + "[?]" + tC
            Else
                tStr = tStr + !c_addr_chn + tC
            End If
        End If

        If IsNull(!x_coord) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Str(!x_coord) + tC
        End If

        If IsNull(!y_coord) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Str(!y_coord) + tC
        End If

        If IsNull(!xy_count) Then
            tStr = tStr + "0"
        Else
            tStr = tStr + Str(!xy_count)
        End If

        tStream.WriteText tStr, adWriteLine
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteGIS_Text:
Exit Sub

Err_WriteGIS_Text:
MsgBox Err.Description
Resume Exit_WriteGIS_Text

End Sub

Private Sub WriteKML_Addr()
On Error GoTo Err_WriteKML_Addr
'
' This program will dump the results to a .gis file
'
If gPlaceRecCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_WriteKML_Addr
End If
'
Dim tStream As ADODB.Stream, tPinyin As Boolean
Set tStream = New ADODB.Stream

tPinyin = False
If GISFrame.Value = 1 Then

```

```

        tStream.Charset = "utf-8"
        tCodeStr = "UTF8"
    ElseIf GISFrame.Value = 3 Then
        tStream.Charset = "ascii"
        tCodeStr = "ASCII"
        tPinyin = True
    Else
        tStream.Charset = "gb18030"
        tCodeStr = "GB18030"
    End If
    tStream.Mode = adModeReadWrite
    tStream.Type = adTypeText
    tStream.Open
'
' next get a file
'

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "addr_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
'
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_WriteKML_Addr
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".kml"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
            tFileName = tFileName + ".kml"
        End If
    End If
'
' write the file
'
'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count
'
' process the table
'

Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
tC = Chr(9) ' the tab
tDQ = Chr(34) ' the double quotation mark
'
' write the header
'

tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
tStream.WriteText "<Document>", adWriteLine
tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
tStream.WriteText tC + "<open>1</open>", adWriteLine
tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

tStream.WriteText tC + "<Style id=" + tDQ + "addr-balloon-template" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
tStream.WriteText tC + tC + tC + "<text>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "<![CDATA[", adWriteLine
tStream.WriteText tC + tC + tC + tC + "ID: ${AddrGIS/PersonID} <br/>", adWriteLine
If Not tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Name Chn: ${AddrGIS/NameHZ} <br/>", adWriteLine
End If
tStream.WriteText tC + tC + tC + tC + "Index Year: ${AddrGIS/IndexYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Sex: ${AddrGIS/Sex} <br/>", adWriteLine
If tPinyin Then
    tStream.WriteText tC + tC + tC + tC + "Address: ${AddrGIS/AddrName} <br/>", adWriteLine
Else
    tStream.WriteText tC + tC + tC + tC + "Address: ${AddrGIS/AddrName} ${AddrGIS/AddrHZ} <br/>", adWriteLine
End If

tStream.WriteText tC + tC + tC + tC + "XY Count: ${AddrGIS/XYCount} <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine

```

```

tStream.WriteText tC + "<!-- Declare the type " + tDQ + "AddrGIS" + tDQ + " with 6 fields -->", adWriteLi
ne
tStream.WriteText tC + "<Schema name=" + tDQ + "AddrGIS" + tDQ + " id=" + tDQ + "AddrGISId" + tDQ + ">",
adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ
+ ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person ID]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
If Not tPinyin Then
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "NameHZ" +
tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Name Chn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "Sex" + tDQ +
">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Sex]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + t
DQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
If Not tPinyin Then
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrHZ" +
tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
End If
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" +
tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
,
.MoveFirst
Do While Not .EOF
' must guard against NULLs, even where there should not be any
,
' write the point header
,
tStream.WriteText tC + "<Placemark>", adWriteLine

If IsNull(!c_name) Then
tStr = "[Bad Data] "
Else
tStr = !c_name
End If
tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

tStream.WriteText tC + tC + "<styleUrl>#status-balloon-template</styleUrl>", adWriteLine
,
' First Year as time stamp
,
If IsNull(!c_index_year) Then
tStr = "N/A"
Else
tStr = Str(!c_index_year)
End If
tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
,
tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#AddrGISId" + tDQ + ">", adWri
teLine
,
' person ID
,
tStr = Str(!c_person_id)
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
,
' Person Name Chn
,
If Not tPinyin Then
If IsNull(!c_name_chn) Then

```



```

        tStr = tStr + "[Bad Data]"
    Else
        If Trim(!c_name_chn) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_name_chn
        End If
    End If
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NameHZ" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
'
'   Index Year
'
If IsNull(!c_index_year) Then
    tStr = "N/A"
Else
    tStr = Str(!c_index_year)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
'   Sex
'
tStr = IIf(!c_female, "F", "M")
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "Sex" + tDQ + ">" + tStr + "</S
impleData>", adWriteLine
'
'   Address Name
'
If IsNull(!c_addr_name) Then
    tStr = "[?]"
ElseIf Trim(!c_addr_name) = "" Then
    tStr = "[?]"
Else
    tStr = !c_addr_name
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
'   Address Name Chinese
'
If Not tPinyin Then
    If IsNull(!c_addr_chn) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_chn
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrHZ" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
End If
'
'   XY Count
'
If IsNull(!xy_count) Then
    tStr = "0"
Else
    tStr = Str(!xy_count)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
tStream.WriteText tC + tC + "<Point>", adWriteLine
'
'   coordinates
'
If IsNull(!x_coord) Then
    tStr = "0"
Else
    tStr = Str(!x_coord)
End If

If IsNull(!y_coord) Then
    tStr = tStr + ",0"
Else
    tStr = tStr + "," + Str(!y_coord)
End If
tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
'

```

```

        ' footer
        '
        tStream.WriteText tC + tC + "</Point>", adWriteLine
        tStream.WriteText tC + "</Placemark>", adWriteLine
        .MoveNext
    Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteKML_Addr:
Exit Sub

Err_WriteKML_Addr:
MsgBox Err.Description
Resume Exit_WriteKML_Addr

End Sub

Private Sub WriteGIS_Addr()
On Error GoTo Err_WriteGIS_Addr
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call WriteKML_Addr
    Exit Sub
End If

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tPinyin As Boolean
Dim tFileSystem, tGDF
'
' This program will dump the results to a .gis file
'
If gPlaceRecCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_WriteGIS_Addr
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False
If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 3 Then
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'

```

```

' next get a file
'
Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "place_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_WriteGIS_Addr
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If
    '
    ' write the file
    '
    'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
    '
    ' process the table
    '
    Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
    tC = Chr(9) ' the tab
    '
    With tRstNode
        '
        ' write the header
        '
        If tPinyin Then
            tStr = "Name" + tC + "Sex" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "X" + tC + "Y" + tC + "xy_count"
        Else
            tStr = "Name" + tC + "NameChn" + tC + "Sex" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC + "xy_count"
        End If
        tStream.WriteText tStr, adWriteLine
        .MoveFirst
        Do While Not .EOF
            ' must guard against NULLs
            '
            If Trim(!c_name) = "" Then
                tStr = "[?]" + tC
            Else
                tStr = !c_name + tC
            End If

            If Not tPinyin Then
                If Trim(!c_name_chn) = "" Then
                    tStr = tStr + "[?]" + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If
            End If

            tStr = tStr + IIf(!c_female, "F", "M") + tC

            If IsNull(!c_index_year) Then
                tStr = tStr + "-2000" + tC
            Else
                tStr = tStr + Str(!c_index_year) + tC
            End If

            ' here guard against blanks as well

            If IsNull(!c_addr_name) Then
                tStr = tStr + "[?]" + tC
            ElseIf Trim(!c_addr_name) = "" Then
                tStr = tStr + "[?]" + tC
            End If
        Loop
    End With

```

```

Else
    tStr = tStr + !c_addr_name + tC
End If

If Not tPinyin Then
    If IsNull(!c_addr_chn) Then
        tStr = tStr + "[?]" + tC
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_addr_chn + tC
    End If
End If

If IsNull(!x_coord) Then
    tStr = tStr + "0" + tC
Else
    tStr = tStr + Str(!x_coord) + tC
End If

If IsNull(!y_coord) Then
    tStr = tStr + "0" + tC
Else
    tStr = tStr + Str(!y_coord) + tC
End If

If IsNull(!xy_count) Then
    tStr = tStr + "0"
Else
    tStr = tStr + Str(!xy_count)
End If

tStream.WriteText tStr, adWriteLine
.MoveNext
Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_WriteGIS_Addr:
Exit Sub

Err_WriteGIS_Addr:
MsgBox Err.Description
Resume Exit_WriteGIS_Addr

End Sub

```

Option Compare Database

' g stands for global

' half of these probably are now irrelevant since I reworked the code

```
Public gRstPersonID As DAO.Recordset, gCurPersonBookmark As Variant
Public gCurPersonID As Long, gCurPersonName As String, gCurPersonNameChn As String
Public gRstPeopleLookUp As DAO.Recordset, gRst As DAO.Recordset, gRstKinList As DAO.Recordset
Public gRstMourning As DAO.Recordset, gRstSimplify As DAO.Recordset
Public gRstRelConv As DAO.Recordset, gRstAddresses As DAO.Recordset, gRstBiogADDR As DAO.Recordset
Public gRstBiogAddrType As DAO.Recordset
Public gMaxUp As Integer, gMaxDown As Integer, gMaxCol As Integer
Public gMaxMarr As Integer, gJustAffine As Boolean, gMourningCircle As Boolean
Public gSaveName As String, gSaveNameChn As String, gSavePersonID As Long, gDisplayLanguage As String
Public gLCID As Integer
Public gRstNodeList As DAO.Recordset, gRstEdge As DAO.Recordset, gRstEdgeLookUp As DAO.Recordset
Public gGdf, gPriorSex As String, gPriorID As Long
Public gStream As ADODB.Stream, gRstImportPeople As DAO.Recordset
Public gCurRecallSource As String
```

```
Private Sub ChkMourning_Click()
    TxtUp.Enabled = Not TxtUp.Enabled
    TxtDown.Enabled = Not TxtDown.Enabled
    TxtCol.Enabled = Not TxtCol.Enabled
    TxtMarr.Enabled = Not TxtMarr.Enabled
End Sub
```

```
Private Sub ChkSimplify_Click()
    If Me.ChkSimplify.Value Then
        DoCmd.OpenForm "frmKinReductionWarning", , , , , acDialog
    End If
End Sub
```

```
Private Sub CmdClose_Click()
On Error GoTo Err_CmdClose_Click
```

```
    DoCmd.Close
```

```
Exit_CmdClose_Click:
    Exit Sub
```

```
Err_CmdClose_Click:
    MsgBox Err.Description
    Resume Exit_CmdClose_Click
```

```
End Sub

Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
    Dim cmdSQL As ADODB.Command, tRecDeleted As Long
    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' If it is a KML file, call the routine and exit
    '
    If ChkKML.Value Then
        Call writeKML
        Exit Sub
    End If
    '
    ' This program will dump the results to a .gis file
    '
    If frmZZ_SCRATCH_KIN.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_CmdGIS_Click
    End If
    '
    ' next get a file
    '
    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tRstPeople As DAO.Recordset, tUseList As Boolean
    Dim tStr As String, tQueryStr As String, tC As String, ti As Integer
    Dim tFileSystem, tGDF

    Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    dlgSaveAs.InitialFileName = "kin_gis.tab"
```

```

If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdGIS_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If

    ' now process the file

    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream

    If GISFrame.Value = 1 Then
        tStream.Charset = "utf-8"
        tCodeStr = "UTF8"
    Else
        tStream.Charset = "gb18030"
        tCodeStr = "GB18030"
    End If
    tStream.Mode = adModeReadWrite
    tStream.Type = adTypeText
    tStream.Open

    tC = Chr(9) ' the tab

    '
    ' put the list of people into ZZ_SCRATCH_PLACE_PEOPLE
    '
    ' and calculate the xy_count
    '
    ' use four SQL calls
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PLACE_PEOPLE"
    cmdSQL.Execute tRecDeleted
    '
    ' get PersonID
    '
    ' see if we include the ego-list
    '
    ' double-check to eliminate listed people, if necessary
    If ChkEgo.Value Then
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PLACE_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_index_addr_id, c_index_addr_py, c_index_addr_hz, x_coord, y_coord ) " +
            "SELECT DISTINCT ZZ_SCRATCH_KIN.c_person_id, ZZ_SCRATCH_KIN.c_name, ZZ_SCRATCH_KIN.c_name_chn, ZZ_SCRATCH_KIN.c_index_year, " +
            "ZZ_SCRATCH_KIN.c_index_addr_id, ZZ_SCRATCH_KIN.c_index_addr_py, ZZ_SCRATCH_KIN.c_index_addr_hz, ZZ_SCRATCH_KIN.x_coord, ZZ_SCRATCH_KIN.y_coord " +
            "FROM ZZ_SCRATCH_IMPORT_PEOPLE RIGHT JOIN ZZ_SCRATCH_KIN ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SCRATCH_KIN.c_person_id " +
            "WHERE ((ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id) Is Null))"
    Else
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PLACE_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_index_addr_id, c_index_addr_py, c_index_addr_hz, x_coord, y_coord ) " +
            "SELECT DISTINCT ZZ_SCRATCH_KIN.c_person_id, ZZ_SCRATCH_KIN.c_name, ZZ_SCRATCH_KIN.c_name_chn, ZZ_SCRATCH_KIN.c_index_year, " +
            "ZZ_SCRATCH_KIN.c_index_addr_id, ZZ_SCRATCH_KIN.c_index_addr_py, ZZ_SCRATCH_KIN.c_index_addr_hz, ZZ_SCRATCH_KIN.x_coord, ZZ_SCRATCH_KIN.y_coord " +
            "FROM ZZ_SCRATCH_KIN"
    End If

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' add KinID
    '
    ' double-check to eliminate listed people, if necessary

```

```

    If ChkEgo.Value Then
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PLACE_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_index_addr_id, c_index_addr_py, c_index_addr_hz, x_coord, y_coord ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_KIN.c_kin_id, ZZ_SCRATCH_KIN.c_kin_name, ZZ_SCRATCH_KIN.c_kin_chn, ZZ_SCRATCH_KIN.c_kin_index_year, " + _
            "ZZ_SCRATCH_KIN.c_kin_addr_id, ZZ_SCRATCH_KIN.c_kin_addr_name, ZZ_SCRATCH_KIN.c_kin_addr_chn, ZZ_SCRATCH_KIN.kin_x_coord, " + _
            "ZZ_SCRATCH_KIN.kin_y_coord " + _
            "FROM ZZ_SCRATCH_IMPORT_PEOPLE RIGHT JOIN (ZZ_SCRATCH_KIN LEFT JOIN ZZ_SCRATCH_PLACE_PEOPLE ON " + _
            "ZZ_SCRATCH_KIN.c_kin_id = ZZ_SCRATCH_PLACE_PEOPLE.c_person_id) ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SCRATCH_KIN.c_kin_id " + _
            "WHERE (((ZZ_SCRATCH_PLACE_PEOPLE.c_person_id) Is Null) AND ((ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id) Is Null)))"
    Else
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PLACE_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_index_addr_id, c_index_addr_py, c_index_addr_hz, x_coord, y_coord ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_KIN.c_kin_id, ZZ_SCRATCH_KIN.c_kin_name, ZZ_SCRATCH_KIN.c_kin_chn, ZZ_SCRATCH_KIN.c_kin_index_year, " + _
            "ZZ_SCRATCH_KIN.c_kin_addr_id, ZZ_SCRATCH_KIN.c_kin_addr_name, ZZ_SCRATCH_KIN.c_kin_addr_chn, " + _
            "ZZ_SCRATCH_KIN.kin_x_coord, ZZ_SCRATCH_KIN.kin_y_coord " + _
            "FROM ZZ_SCRATCH_KIN LEFT JOIN ZZ_SCRATCH_PLACE_PEOPLE ON ZZ_SCRATCH_KIN.c_kin_id = ZZ_SCRATCH_PLACE_PEOPLE.c_person_id " + _
            "WHERE (((ZZ_SCRATCH_PLACE_PEOPLE.c_person_id) Is Null)))"
    End If

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted

    cmdSQL.CommandText = "Delete * from tmpXY"
    cmdSQL.Execute tRecDeleted

    tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
        "SELECT ZZ_SCRATCH_PLACE_PEOPLE.x_coord, ZZ_SCRATCH_PLACE_PEOPLE.y_coord, Count(ZZ_SCRATCH_PLACE_PEOPLE.x_coord) AS CountOfx_coord, " + _
        "Count(ZZ_SCRATCH_PLACE_PEOPLE.y_coord) AS CountOfy_coord " + _
        "FROM ZZ_SCRATCH_PLACE_PEOPLE " + _
        "GROUP BY ZZ_SCRATCH_PLACE_PEOPLE.x_coord, ZZ_SCRATCH_PLACE_PEOPLE.y_coord"

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted

    tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_PLACE_PEOPLE ON (tmpXY.y_coord = " + _
        "ZZ_SCRATCH_PLACE_PEOPLE.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_PLACE_PEOPLE.x_coord) " + _
        "SET ZZ_SCRATCH_PLACE_PEOPLE.xy_count = [tmpXY].[CountOfx_coord];"

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted

    Set tRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_PLACE_PEOPLE", dbOpenDynaset)

    'If Left(TxtName.Value, 1) = "[" Then
    '    tStr = "SELECT KIN_QUERY.*, ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id " + _
    '        "FROM (SELECT DISTINCT ZZ_SCRATCH_KIN.c_kin_id, ZZ_SCRATCH_KIN.c_kin_name, ZZ_SCRATCH_KIN.c_kin_chn, " + _
    '        "ZZ_SCRATCH_KIN.kin_x_coord, ZZ_SCRATCH_KIN.kin_y_coord, ZZ_SCRATCH_KIN.c_kin_addr_name, " + _
    '        "ZZ_SCRATCH_KIN.c_kin_addr_chn, ZZ_SCRATCH_KIN.c_kin_addr_desc_chn, ZZ_SCRATCH_KIN.c_kin_index_year, " + _
    '        "ZZ_SCRATCH_KIN.c_kin_female, ZZ_SCRATCH_KIN.xy_count " + _
    '        "FROM ZZ_SCRATCH_KIN) AS KIN_QUERY " + _
    '        "LEFT JOIN ZZ_SCRATCH_IMPORT_PEOPLE " + _
    '        "ON KIN_QUERY.c_kin_id = ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id"
    '    Set tRstNode = CurrentDb.OpenRecordset(tStr, dbOpenDynaset)
    '    tUseList = True
    'Else
    '    Set tRstNode = frmZZ_SCRATCH_KIN.Form.Recordset
    '    tUseList = False
    'End If

    With tRstPeople
        ' write the header
        tStr = "Name" + tC + "NameChn" + tC + "IndexYear" + tC + "Sex" + tC + _
            "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC + "XY_count"
        tStream.WriteText tStr, adWriteLine
        .MoveFirst
    End With

```

```

Do While Not .EOF
    ' must guard against NULLs
    '
    If IsNull(!c_name) Then
        tStr = "[?]" + tC
    Else
        If Trim(!c_name) = "" Then
            tStr = "[?]" + tC
        Else
            tStr = !c_name + tC
        End If
    End If

    If IsNull(!c_name_chn) Then
        tStr = tStr + "[?]" + tC
    Else
        If Trim(!c_name_chn) = "" Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_name_chn + tC
        End If
    End If

    If IsNull(!c_index_year) Then
        tStr = tStr + "-2000" + tC
    Else
        tStr = tStr + Str(!c_index_year) + tC
    End If

    If !c_female Then
        tStr = tStr + "F" + tC
    Else
        tStr = tStr + "M" + tC
    End If

    If IsNull(!c_index_addr_py) Then
        tStr = tStr + "[?]" + tC
    ElseIf Trim(!c_index_addr_py) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_index_addr_py + tC
    End If

    If IsNull(!c_index_addr_hz) Then
        tStr = tStr + "[?]" + tC
    ElseIf Trim(!c_index_addr_hz) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_index_addr_hz + tC
    End If

    If IsNull(!x_coord) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Str(!x_coord) + tC
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Str(!y_coord) + tC
    End If

    If IsNull(!xy_count) Then
        tStr = tStr + "0"
    Else
        tStr = tStr + Str(!xy_count)
    End If

    tStream.WriteText tStr, adWriteLine
    .MoveNext
Loop
End With

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstPeople = Nothing

```



```

tStream.Close
Set tStream = Nothing
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PLACE_PEOPLE"
cmdSQL.Execute tRecDeleted
'
Else
'The user pressed Cancel.
End If

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGIS_Click:
Exit Sub

Err_CmdGIS_Click:
MsgBox Err.Description
Resume Exit_CmdGIS_Click

End Sub

Private Sub CmdGUESS_Click()
On Error GoTo Err_CmdGUESS_Click
'
' This program will dump the results of the search to a .gdf file
'
' for the moment I'll just describe the format of the .gdf file
'
' nodedef> name, color, label, labelvisible, style, pinyin VARCHAR(50), nodedist INT
' name = str(c_person_id)
' color = red (1), orange (2), yellow (3), green (4), blue (5)
' label = c_name_chn
' style = 4 (text inside a rectangle)
' pinyin = c_name
' nodedist = c_node_dist INT
' indexyear = c_index_year INT
' sex = c_female > (F,M)
'
' edgedef> node1, node2, color, label, labelvisible, edge_desc VARCHAR(50)
' node1 = str(c_person_id) for node1
' node2 = str(c_node_id) for node2
' color = red (1), orange (2), yellow (3), green (4), blue (5)
' label = c_link_chn
' edge_desc = c_link_desc
'
' the central question is whether to do distance optimizations
'
' first see if there are any records to process
'
If frmZZ_SCRATCH_KIN.Form.Recordset.RecordCount = 0 Then
MsgBox "There are no records to save."
GoTo Exit_CmdGUESS_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tUseList As Boolean
Dim tColor(50) As String, tMetricSum As Integer, tQueryStr As String, tPersonID As Long
Dim gStream As ADODB.Stream, tCodeStr As String
'Dim tFileSystem, tGDF

' set up the stream to write to

Set gStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
gStream.Charset = "utf-8"
tCodeStr = "UTF8"
ElseIf CodeFrame.Value = 2 Then
gStream.Charset = "big5"
tCodeStr = "BIG5"
ElseIf CodeFrame.Value = 3 Then
gStream.Charset = "gb2312"
tCodeStr = "GB2312"
Else

```

```

        gStream.Charset = "ascii"
        tCodeStr = "ascii"
    End If

    Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    'Use a With...End With block to reference the FileDialog object.
    With dlgSaveAs
        .InitialFileName = "kin_" + tCodeStr + ".gdf"
        If .Show = -1 Then
            '
            tFileName = ""
            For Each tFN In .SelectedItems
                tFileName = tFN
                If Not tFileName = "" Then
                    Exit For
                End If
            Next
            If tFileName = "" Then
                MsgBox "Bad file Name."
                GoTo Exit_CmdGUESS_Click
            Else
                ' make sure the file name has a txt extension
                If Len(tFileName) < 5 Then
                    tFileName = tFileName + ".gdf"
                ElseIf Not (LCase(Right(tFileName, 4)) = ".gdf") Then
                    tFileName = tFileName + ".gdf"
                End If
            End If
            '
            ' now process the file (second true removed to make ASCII)
            '
            'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
            'Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)
            '
            ' we have a file name: now open the stream for writing

            gStream.Mode = adModeReadWrite
            gStream.Type = adTypeText
            gStream.Open

            ' define the colors for the edges
            '
            tColor(1) = "black"
            tColor(2) = "blue"
            tColor(3) = "green"
            tColor(4) = "yellow"
            tColor(5) = "orange"
            For ti = 6 To 20
                tColor(ti) = "red"
            Next

            'MsgBox "Preparing temp tables 1"
            '
            ' prepare the temp table for the edge data

            Dim cmdSQL As ADODB.Command
            Set cmdSQL = New ADODB.Command
            cmdSQL.ActiveConnection = CurrentProject.Connection
            cmdSQL.CommandType = adCmdText

            cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_KINNET_EDGE"
            cmdSQL.Execute tRecDeleted
            '
            ' copy the data for determining edges
            '
            tQueryStr = "INSERT INTO ZZ_SCRATCH_KINNET_EDGE SELECT ZZ_SCRATCH_KINNET.* FROM ZZ_SCRATCH_KINNET " + _
                "WHERE (((ZZ_SCRATCH_KINNET.c_kin_rel)<>'ego'))"
            '
            cmdSQL.CommandText = tQueryStr
            cmdSQL.Execute tRecDeleted
            '
            ' now delete the unneeded edges
            '
            tQueryStr = "UPDATE (ZZ_SCRATCH_KINNET_EDGE INNER JOIN ZZ_SCRATCH_KINNET_EDGE AS " + _
                "ZZ_SCRATCH_KINNET_EDGE_1 ON ZZ_SCRATCH_KINNET_EDGE.c_person_id = ZZ_SCRATCH_KINNET_EDGE_1.c_pers" + _
                "ON_id) " + _
                "INNER JOIN ZZ_SCRATCH_KINNET_EDGE AS ZZ_SCRATCH_KINNET_EDGE_2 ON " + _
                "(ZZ_SCRATCH_KINNET_EDGE_1.c_kin_id = ZZ_SCRATCH_KINNET_EDGE_2.c_person_id) AND " + _

```

```

" (ZZ_SCRATCH_KINNET_EDGE.c_kin_id = ZZ_SCRATCH_KINNET_EDGE_2.c_kin_id) " + _
"SET ZZ_SCRATCH_KINNET_EDGE.c_delete = 1 " + _
"WHERE (((ZZ_SCRATCH_KINNET_EDGE.c_upstep)=[ZZ_SCRATCH_KINNET_EDGE_1].[c_upstep]+ " + _
"[ZZ_SCRATCH_KINNET_EDGE_2].[c_upstep]) AND ((ZZ_SCRATCH_KINNET_EDGE.c_dwnstep)=" + _
"[ZZ_SCRATCH_KINNET_EDGE_1].[c_dwnstep]+[ZZ_SCRATCH_KINNET_EDGE_2].[c_dwnstep]) AND " + _
"((ZZ_SCRATCH_KINNET_EDGE.c_marstep)=[ZZ_SCRATCH_KINNET_EDGE_1].[c_marstep]+ " + _
"[ZZ_SCRATCH_KINNET_EDGE_2].[c_marstep]) AND ((ZZ_SCRATCH_KINNET_EDGE.c_colstep)=" + _
"[ZZ_SCRATCH_KINNET_EDGE_1].[c_colstep]+[ZZ_SCRATCH_KINNET_EDGE_2].[c_colstep])AND " + _
"((ZZ_SCRATCH_KINNET_EDGE.c_person_id)<>[ZZ_SCRATCH_KINNET_EDGE_1].[c_kin_id]))"
,
'MsgBox "About to prune"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_KINNET_EDGE WHERE c_delete = 1"
cmdSQL.Execute tRecDeleted
'MsgBox "Pruning complete"
,
' now collect the node information
,
'MsgBox "Preparing temp tables 2"
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_GEPHI_NODE"
cmdSQL.Execute tRecDeleted
,
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_GEPHI_NODE_DISTINCT"
cmdSQL.Execute tRecDeleted
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_GEPHI_NODE ( c_person_id, c_name, c_name_chn, c_index_year, c_fem
ale, c_addr_id, c_addr_name, c_addr_chn, " + _
"x_coord, y_coord, c_dy, c_dynasty, c_dynasty_chn) " + _
"SELECT ZZ_SCRATCH_KINNET_EDGE.c_person_id, ZZ_SCRATCH_KINNET_EDGE.c_name, ZZ_SCRATCH_KINNET_EDGE
.c_name_chn, " + _
"ZZ_SCRATCH_KINNET_EDGE.c_index_year, ZZ_SCRATCH_KINNET_EDGE.c_female, ZZ_SCRATCH_KINNET_EDGE.c_a
ddr_id, ZZ_SCRATCH_KINNET_EDGE.c_addr_name, " + _
"ZZ_SCRATCH_KINNET_EDGE.c_addr_chn, ZZ_SCRATCH_KINNET_EDGE.x_coord, ZZ_SCRATCH_KINNET_EDGE.y_coord,
" + _
"ZZ_SCRATCH_KINNET_EDGE.c_dy, ZZ_SCRATCH_KINNET_EDGE.c_dynasty, ZZ_SCRATCH_KINNET_EDGE.c_dynasty_
chn " + _
"FROM ZZ_SCRATCH_KINNET_EDGE"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_GEPHI_NODE ( c_person_id, c_name, c_name_chn, c_index_year, c_fem
ale, c_addr_id, c_addr_name, c_addr_chn, " + _
"x_coord, y_coord, c_dy, c_dynasty, c_dynasty_chn) " + _
"SELECT ZZ_SCRATCH_KINNET_EDGE.c_kin_id, ZZ_SCRATCH_KINNET_EDGE.c_kin_name, ZZ_SCRATCH_KINNET_EDG
E.c_kin_chn, " + _
"ZZ_SCRATCH_KINNET_EDGE.c_kin_index_year, ZZ_SCRATCH_KINNET_EDGE.c_kin_female, ZZ_SCRATCH_KINNET_
EDGE.c_kin_addr_id, ZZ_SCRATCH_KINNET_EDGE.c_kin_addr_name, " + _
"ZZ_SCRATCH_KINNET_EDGE.c_kin_addr_chn, ZZ_SCRATCH_KINNET_EDGE.kin_x_coord, ZZ_SCRATCH_KINNET_EDG
E.kin_y_coord, " + _
"ZZ_SCRATCH_KINNET_EDGE.c_kin_dy, ZZ_SCRATCH_KINNET_EDGE.c_kin_dynasty, ZZ_SCRATCH_KINNET_EDGE.c_
kin_dynasty_chn " + _
"FROM ZZ_SCRATCH_KINNET_EDGE"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' append the results
,
tQueryStr = "INSERT INTO ZZ_SCRATCH_GEPHI_NODE_DISTINCT ( c_person_id, c_name, c_name_chn, c_index_ye
ar, c_female, c_imported, c_addr_id, c_addr_name, c_addr_chn, " + _
"x_coord, y_coord, c_dy, c_dynasty, c_dynasty_chn) " + _
"SELECT DISTINCT ZZ_SCRATCH_GEPHI_NODE.c_person_id, ZZ_SCRATCH_GEPHI_NODE.c_name, ZZ_SCRATCH_GEPH
I_NODE.c_name_chn, " + _
"ZZ_SCRATCH_GEPHI_NODE.c_index_year, ZZ_SCRATCH_GEPHI_NODE.c_female, FALSE AS c_imported, " + _
"ZZ_SCRATCH_GEPHI_NODE.c_addr_id, ZZ_SCRATCH_GEPHI_NODE.c_addr_name, ZZ_SCRATCH_GEPHI_NODE.c_addr
_chn, ZZ_SCRATCH_GEPHI_NODE.x_coord, " + _
"ZZ_SCRATCH_GEPHI_NODE.y_coord, ZZ_SCRATCH_GEPHI_NODE.c_dy, ZZ_SCRATCH_GEPHI_NODE.c_dynasty, ZZ_S
CRATCH_GEPHI_NODE.c_dynasty_chn " + _
"FROM ZZ_SCRATCH_GEPHI_NODE"
,
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
,
' if the original IDs come from a list, mark the list people
,
tQueryStr = "UPDATE ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZ_SCRATCH_GEPHI_NODE_DISTINCT " + _
"ON ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c_person_id " + _

```

```

"SET ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c_imported = True"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SCRATCH_KINNET_EDGE", dbOpenDynaset)
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_GEPHI_NODE_DISTINCT", dbOpenDynaset)
tRstNode.MoveLast
'
' process the two tables
'
tC = Chr(44) ' the comma
'
' first the nodes: define the record structure
'
' if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a
pinyin field
If tCodeStr = "ascii" Then
    tStr = "nodedef> name VARCHAR" + tC + "color VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible
BOOLEAN" + _
        tC + "style INT" + tC + "indexyear INT" + tC + "sex VARCHAR(1)" + tC + "addr_name VARCHAR" +
tC + _
        "latitude DOUBLE" + tC + "longitude DOUBLE" + tC + "DynastyCode INT" + tC + "dynasty VARCHAR"
Else
    tStr = "nodedef> name VARCHAR" + tC + "color VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible
BOOLEAN" + _
        tC + "style INT" + tC + "pinyin VARCHAR(50)" + tC + "indexyear INT" + tC + "sex VARCHAR(1)" +
tC + "addr_name VARCHAR" + tC + "addr_chn VARCHAR" + tC + _
        "latitude DOUBLE" + tC + "longitude DOUBLE" + tC + "DynastyCode INT" + tC + "dynasty VARCHAR"
+ tC + "dynasty_chn VARCHAR"
End If
gStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
'
'MsgBox "processing tables"
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' name = the ID of the person
        tStr = Trim(Str(!c_person_id)) + tC
        '
        ' color
        If !c_imported Then
            tStr = tStr + "Red" + tC
        Else
            tStr = tStr + "Blue" + tC
        End If
        '
        ' label
        If tCodeStr = "ascii" Then
            If IsNull(!c_name) Then
                tStr = tStr + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        Else
            If IsNull(!c_name_chn) Then
                tStr = tStr + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If
        End If

        ' labelvisible = true, style = 4 (text inside a rectangle)
        tStr = tStr + "true" + tC + "4" + tC

        ' pinyin = c_name (if using characters)
        If Not (tCodeStr = "ascii") Then
            tStr = tStr + !c_name + tC
        End If
        '
        ' indexyear = c_index_year INT
        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + tC
        End If

        ' sex = c_female > (F,M)
        tStr = tStr + IIf(!c_female, "F", "M") + tC
    
```

```

'    address name
If tCodeStr = "ascii" Then
    If IsNull(!c_addr_name) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If
Else
    If IsNull(!c_addr_chn) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_chn + tC
    End If
    If IsNull(!c_addr_name) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If
End If

'    latitude = !y_coord
If IsNull(!y_coord) Then
    tStr = tStr + "0.0" + tC
Else
    tStr = tStr + Str(!y_coord) + tC
End If

'    longitude = !x_coord
If IsNull(!x_coord) Then
    tStr = tStr + "0.0" + tC
Else
    tStr = tStr + Str(!x_coord) + tC
End If

'    dynasty information
'
If tCodeStr = "ascii" Then
    If IsNull(!c_dynasty) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Str(!c_dy) + tC + !c_dynasty
    End If
Else
    If Not IsNull(!c_dynasty) Then
        tStr = tStr + Str(!c_dy) + tC + !c_dynasty + tC + !c_dynasty_chn
    End If
End If
gStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
.MoveNext
Loop
End With

'
' now the edges:  define the record structure
tStr = "edgedef> node1" + tC + "node2" + tC + "color" + tC + "label"
gStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)

With tRstEdge
.MoveNext
Do While Not .EOF
    tStr = Trim(Str(!c_person_id)) + tC
    '    node1 = str(c_person_id) for node1
    tStr = tStr + Trim(Str(!c_kin_id)) + tC
    '    node2 = str(c_node_id) for node2
    '
    '    add all the metrics together
    tMetricSum = !c_marstep + !c_upstep + !c_dwnstep + !c_colstep
    tStr = tStr + tColor(tMetricSum + 1) + tC
    '    color = white (1), blue (2), green (3), yellow (4), orange (5)

    If IsNull(!c_kin_rel) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_kin_rel + tC
    End If
    '    label = c_kin_rel
    '
    gStream.WriteText tStr, adWriteLine

```

```

        'tGDF.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Set gStream = Nothing
'tGDF.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
'Set tGDF = Nothing
'Set tFileSystem = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGUESS_Click:
Exit Sub

Err_CmdGUESS_Click:
MsgBox Err.Description
Resume Exit_CmdGUESS_Click

End Sub

Private Sub CmdImport_Click()
On Error GoTo Err_CmdImport_Click
Dim cmdSQL As ADODB.Command
Dim tStrQuestion As String, tQuit As Boolean

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tFileSystem, tList

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

tQuit = False
'
If Not tQuit Then
    ' open the list

    Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

    'Use a With...End With block to reference the FileDialog object.

    tFileName = ""

    With dlgSaveAs
        .InitialFileName = ""
        If .Show = -1 Then
            '
            For Each tFN In .SelectedItems
                tFileName = tFN
                If Not tFileName = "" Then
                    Exit For
                End If
            Next
            If tFileName = "" Then
                MsgBox "Bad file Name."
                GoTo Exit_CmdImport_Click
            End If
        End If
    End With

    ' Clear the people table now that we are ready to go

```

```

If Not (tFileName = "") Then
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = "Delete * from InputErrorList"
    cmdSQL.Execute tRecDeleted

    cmdSQL.CommandText = "Delete * from TempImportList"
    cmdSQL.Execute tRecDeleted

    DoCmd.TransferText acImportDelim, "ImportPeopleList_Space", "TempImportList", tFileName, 0
    '
    ' TransferType=acImportDelim
    ' SpecificationName = "TempImportList" (apparently it is saved in the database itself)
    ' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
    ' HasFieldNames = False (0)
    '
    ' copy the bad IDs
    '
    tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
        "FROM BIOG_MAIN RIGHT JOIN TempImportList ON BIOG_MAIN.c_personid = TempImportList.ImportID " + _
        "WHERE (((BIOG_MAIN.c_personid) Is Null))"

    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted

    If tRecDeleted > 0 Then
        MsgBox "Some ID were not successfully imported: please look at InputErrorList."
    End If
    '
    ' copy the good IDs
    '
    tStrSQL = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id ) SELECT DISTINCT TempImportList.Import
ID " + _
        "FROM BIOG_MAIN INNER JOIN TempImportList ON BIOG_MAIN.c_personid = TempImportList.ImportID"

    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted

    If tRecDeleted = 0 Then
        TxtName.Value = "[Error]"
        TxtNameChn.Value = "[Error]"
        CmdRun.Enabled = False
    Else
        TxtName.Value = "[Imported List]"
        TxtNameChn.Value = "[Imported List]"
        CmdRun.Enabled = True
    End If
    gCurRecallSource = ""
    Set cmdSQL = Nothing
    Set tFileSystem = Nothing
End If

End If

```

```

Exit_CmdImport_Click:
Exit Sub

```

```

Err_CmdImport_Click:
MsgBox Err.Description
Resume Exit_CmdImport_Click

```

```
End Sub
```

```

Private Sub CmdNeo4j_Click()
    Call saveNeo4jFiles
End Sub

```

```

Private Sub CmdRun_Click()
On Error GoTo Err_CmdRun_Click

```

```

Dim tTrue As Integer, tFalse As Integer, tLoopCount As Long, tErrorStr As String
Dim tContinue As Integer, tAddrID As Long, tExitDo As Boolean, tRecCount As Long, tRecDelete As Long
Dim tRstDummy As DAO.Recordset, tAppendQuery As QueryDef
Dim tSeekStr As String, tLoopMax As Long, tLoopInfoStr As String, tKinQueryStr As String, tQueryStr As String
Dim tNodeDistQueryStr As String, tPruneTmpQueryDuplesStr As String, tPruneTmpQuery As String
Dim tPruneInversesQueryStr1 As String, tPruneTmpInversesQueryStr1 As String, tPruneInversesQueryStr2 As String
Dim tAppendQueryStr As String, tPruneTmpInversesQueryStr2 As String
Dim tKinFirstQueryStr As String, tPruneTmpQueryDuplesStr2 As String, tPruneTmpQuery2 As String

```

frm_LookAtKinship - 12

```
tTrue = -1
tFalse = 0

tLoopMax = TxtMaxLoop.Value
gMaxUp = TxtUp.Value
gMaxDown = TxtDown.Value
gMaxCol = TxtCol.Value
gMaxMarr = TxtMarr.Value
gMourningCircle = ChkMourning.Value

' If this is a mourning circle search, set the max constraints

If gMourningCircle Then
    gMaxUp = 4
    gMaxDown = 4
    gMaxCol = 1
    gMaxMarr = 2
End If

Dim KinQuery As DAO.QueryDef
Dim prm As DAO.Parameter
Dim cmdSQL As ADODB.Command, tRecDeleted As Long, strSQL As String

' Clear the tables

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

' clear the working files

cmdSQL.CommandText = "Delete * from ZZ_KIN_LIST"
cmdSQL.Execute tRecCount

cmdSQL.CommandText = "Delete * from ZZ_KIN_LIST_TMP"
cmdSQL.Execute tRecCount

' close ZZ_SCRATCH_KIN

Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_KIN", dbOpenDynaset)

Set gRstPersonID = frmZZ_SCRATCH_KIN.Form.Recordset
Set frmZZ_SCRATCH_KIN.Form.Recordset = tRstDummy
gRstPersonID.Close

' now zap the ego-relative form person file

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_KIN"
cmdSQL.Execute tRecCount

' repeat for ZZ_SCRATCH_KINNET, the kinship form file

Set gRstKinList = frmZZ_SCRATCH_KINNET.Form.Recordset
Set frmZZ_SCRATCH_KINNET.Form.Recordset = tRstDummy
gRstKinList.Close

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_KINNET"
cmdSQL.Execute tRecDeleted

Set tRstDummy = Nothing

' initialize the public recordsets to be used by the process_record routine

' ZZ_KIN_CONV needs to be opened as a table in order to use the indices

Set gRstRelConv = CurrentDb.OpenRecordset("ZZ_KINREL_CONV", dbOpenTable)

' Set the index
'gRstRelConv.Index = "REL_CONV"

' this copies the people on the import list (which is just the selected person if one does not use a list)

tQueryStr = "INSERT INTO ZZ_KIN_LIST ( c_personid, c_kin_id, c_kinrel, c_kinrel_total, c_kinrel_total_raw, c_kinrel_total_simplified, " + _
    "c_kin_code, c_up_total, c_down_total, c_mar_total, c_col_total, c_distance, c_up, c_down, c_mar, c_col, " + _
    "c_prior_female, c_kin_female, c_kin_sex, c_female, c_sex, c_personid_root ) " + _
    "SELECT ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id, ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id AS c_kin_id, " + _
    "'ego' AS c_kin_rel, 'ego' AS c_kin_rel_total, 'ego' AS c_kin_rel_total_raw, 'ego' AS c_kin_rel_total
```



```

_simplified, -3 AS c_kin_code, " +
    "0 AS c_up_total, 0 AS c_down_total, 0 AS c_mar_total, 0 AS c_col_total, 0 AS c_distance, " +
    "0 AS c_up, 0 AS c_down, 0 AS c_mar, 0 AS c_col, ZZZ_BIOG_MAIN.c_female AS c_prior_female, ZZZ_BIOG_M
AIN.c_female AS c_kin_female, " +
    "iif(ZZZ_BIOG_MAIN.c_female,'F','M'), ZZZ_BIOG_MAIN.c_female, iif(ZZZ_BIOG_MAIN.c_female,'F','M'), ZZ
_SCRATCH_IMPORT_PEOPLE.c_person_id AS c_personid_root " +
    "FROM ZZZ_BIOG_MAIN INNER JOIN ZZ_SCRATCH_IMPORT_PEOPLE ON ZZZ_BIOG_MAIN.c_personid = ZZ_SCRATCH_IMPORT_P
EOPLE.c_person_id"

' the initial list of "ego" roots is now intialized in ZZ_KIN_LIST, and the personID is stored as c_personid
root: use this to create ZZ_KIN_LIST_TMP
' in the first query, one begins to build out with a first layer of kinship relations
' as the first layer, we put the kin_rel as both the kin_rel and the kin_rel_total

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount

' the new logic is to not test the metrics until after the reduction routine is run

tKinFirstQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_personid, c_kin_id, c_kinrel, c_kinrel_total, " +
    "c_kinrel_total_raw, c_kinrel_total_simplified, c_kin_code, c_up_total, c_down_total, c_mar_total, c_col_
total, c_distance, " +
    "c_up, c_down, c_mar, c_col, c_personid_root, c_prior_female, c_kin_female, c_kin_sex, c_female, c_sex, c
_notes, " +
    "c_source, c_source_text_chn, c_source_text ) " +
    "SELECT DISTINCT ZZZ_KIN_BIOG_ADDR.c_personid, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_desc
, " +
    "KINSHIP_CODES.c_kinrel_simplified AS c_kinrel_total, ZZZ_KIN_BIOG_ADDR.c_link_desc AS c_kinrel_total
_raw, " +
    "KINSHIP_CODES.c_kinrel_simplified AS c_kinrel_total_simplified, " +
    "ZZZ_KIN_BIOG_ADDR.c_link_code, ZZZ_KIN_BIOG_ADDR.c_upstep AS c_up_total, ZZZ_KIN_BIOG_ADDR.c_dwnstep
AS c_down_total, " +
    "ZZZ_KIN_BIOG_ADDR.c_marstep AS c_mar_total, ZZZ_KIN_BIOG_ADDR.c_colstep AS c_col_total, 0 AS c_dista
nce, " +
    "ZZZ_KIN_BIOG_ADDR.c_upstep, ZZZ_KIN_BIOG_ADDR.c_dwnstep, ZZZ_KIN_BIOG_ADDR.c_marstep, ZZZ_KIN_BIOG_A
DDR.c_colstep, " +
    "ZZ_KIN_LIST.c_personid_root, ZZ_KIN_LIST.c_female AS c_prior_female, ZZZ_KIN_BIOG_ADDR.c_node_female
, iif(ZZZ_KIN_BIOG_ADDR.c_node_female,'F','M'), " +
    "ZZZ_KIN_BIOG_ADDR.c_female, iif(ZZZ_KIN_BIOG_ADDR.c_female,'F','M'), " +
    "'Notes: ' + ZZZ_KIN_BIOG_ADDR.c_person_name_chn + ' > ' + ZZZ_KIN_BIOG_ADDR.c_node_chn + ' (' + ZZZ_
KIN_BIOG_ADDR.c_link_desc + ') ' AS c_notes, " +
    "ZZZ_KIN_BIOG_ADDR.c_source, ZZZ_KIN_BIOG_ADDR.c_title_chn, ZZZ_KIN_BIOG_ADDR.c_title " +
    "FROM (ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_KIN_LIST ON ZZZ_KIN_BIOG_ADDR.c_personid = ZZ_KIN_LIST.c_kin_id) "
+
    "INNER JOIN KINSHIP_CODES ON ZZZ_KIN_BIOG_ADDR.c_link_code = KINSHIP_CODES.c_kincode"

' each subsequent layer adds the new kin_rel to the kin_rel_total and the total cumulative steps are summed

tKinQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_personid, c_kin_id, c_kinrel, c_kinrel_total_simplified, c_ki
nrel_total, " +
    "c_kinrel_total_raw, c_kin_code, c_up_total, c_down_total, c_mar_total, c_col_total, c_distance, " +
    "c_up, c_down, c_mar, c_col, c_personid_root, c_prior_female, c_kin_female, c_kin_sex, c_female, c_sex, c
_notes, " +
    "c_source, c_source_text_chn, c_source_text) " +
    "SELECT DISTINCT ZZZ_KIN_BIOG_ADDR.c_personid, ZZZ_KIN_BIOG_ADDR.c_node_id, ZZZ_KIN_BIOG_ADDR.c_link_desc
, " +
    "[ZZ_KIN_LIST].[c_kinrel_total_simplified]+[KINSHIP_CODES].[c_kinrel_simplified] AS c_kinrel_total_si
mplified, " +
    "[ZZ_KIN_LIST].[c_kinrel_total]+[KINSHIP_CODES].[c_kinrel_simplified] AS c_kinrel_total, " +
    "ZZ_KIN_LIST.c_kinrel_total_raw+ZZZ_KIN_BIOG_ADDR.c_link_desc AS c_kinrel_total_raw, " +
    "ZZZ_KIN_BIOG_ADDR.c_link_code, ZZZ_KIN_BIOG_ADDR.c_upstep+ZZ_KIN_LIST.c_up_total AS c_up_total, " +
    "ZZZ_KIN_BIOG_ADDR.c_dwnstep+ZZ_KIN_LIST.c_down_total AS c_down_total, " +
    "ZZZ_KIN_BIOG_ADDR.c_marstep+ZZZ_KIN_LIST.c_mar_total AS c_mar_total, " +
    "ZZZ_KIN_BIOG_ADDR.c_colstep+ZZZ_KIN_LIST.c_col_total AS c_col_total, " +
    "ZZ_KIN_LIST.c_distance, ZZZ_KIN_BIOG_ADDR.c_upstep, ZZZ_KIN_BIOG_ADDR.c_dwnstep, ZZZ_KIN_BIOG_ADDR.c
_marstep, ZZZ_KIN_BIOG_ADDR.c_colstep, " +
    "ZZ_KIN_LIST.c_personid_root, ZZ_KIN_LIST.c_female AS c_prior_female, ZZZ_KIN_BIOG_ADDR.c_node_female
, iif(ZZZ_KIN_BIOG_ADDR.c_node_female,'F','M'), " +
    "ZZZ_KIN_BIOG_ADDR.c_female, iif(ZZZ_KIN_BIOG_ADDR.c_female,'F','M'), " +
    "ZZ_KIN_LIST.c_notes + ' > ' + ZZZ_KIN_BIOG_ADDR.c_node_chn + ' (' + ZZZ_KIN_BIOG_ADDR.c_link_desc +
') ', " +
    "ZZZ_KIN_BIOG_ADDR.c_source, ZZZ_KIN_BIOG_ADDR.c_title_chn, ZZZ_KIN_BIOG_ADDR.c_title " +
    "FROM (ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_KIN_LIST ON ZZZ_KIN_BIOG_ADDR.c_personid = ZZ_KIN_LIST.c_kin_id) "
+
    "INNER JOIN KINSHIP_CODES ON ZZZ_KIN_BIOG_ADDR.c_link_code = KINSHIP_CODES.c_kincode " +
    "WHERE ((ZZ_KIN_LIST.c_distance)=)"

' the various queries for cleaning up the results (need editing)
' ZZ_KIN_LIST is our collection of current results

```

```

' ZZ_KIN_LIST_TMP is the new material coming from the most recent query loop which looks for kin of the c_ki
n_id

' if the new kin (in c_kin_id) does not already show up as a relative of someone else already in the databas
e, this is one more step distant
tNodeDistQueryStr = "UPDATE ZZ_KIN_LIST_TMP LEFT JOIN ZZ_KIN_LIST ON ZZ_KIN_LIST_TMP.c_kin_id = ZZ_KIN_LIST.c
_kin_id " +
"SET ZZ_KIN_LIST_TMP.c_distance = [ZZ_KIN_LIST_TMP].[c_distance]+1 " +
"WHERE (((ZZ_KIN_LIST.c_personid) Is Null))"

' for insurance, explicitly delete duplicate results

tPruneTmpQuery = "UPDATE ZZ_KIN_LIST INNER JOIN ZZ_KIN_LIST_TMP ON " +
"(ZZ_KIN_LIST.c_kin_id = ZZ_KIN_LIST_TMP.c_kin_id) AND " +
"(ZZ_KIN_LIST.c_personid = ZZ_KIN_LIST_TMP.c_personid) " +
"SET ZZ_KIN_LIST_TMP.c_delete = 1;"

' delete inverse results
tPruneTmpQuery2 = "UPDATE ZZ_KIN_LIST INNER JOIN ZZ_KIN_LIST_TMP ON " +
"(ZZ_KIN_LIST.c_personid = ZZ_KIN_LIST_TMP.c_kin_id) AND " +
"(ZZ_KIN_LIST.c_kin_id = ZZ_KIN_LIST_TMP.c_personid) " +
"SET ZZ_KIN_LIST_TMP.c_delete = 1;"

tPruneTmpQueryDuplesStr = "UPDATE ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 INNER JOIN " +
"ZZ_KIN_LIST_TMP ON (ZZ_KIN_LIST_TMP_1.c_personid = ZZ_KIN_LIST_TMP.c_personid) " +
"AND (ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_kin_id) " +
"AND (ZZ_KIN_LIST_TMP_1.c_kin_code = ZZ_KIN_LIST_TMP.c_kin_code) " +
"SET ZZ_KIN_LIST_TMP.c_delete = 1 " +
"WHERE ((([ZZ_KIN_LIST_TMP].[c_up_total]*1000+[ZZ_KIN_LIST_TMP].[c_down_total]*100+[ZZ_KIN_LIST_TMP].[c_co
l_total]*10+[ZZ_KIN_LIST_TMP].[c_mar_total]>") +
"[ZZ_KIN_LIST_TMP_1].[c_up_total]*1000+[ZZ_KIN_LIST_TMP_1].[c_down_total]*100+[ZZ_KIN_LIST_TMP_1].[c_col
total]*10+[ZZ_KIN_LIST_TMP_1].[c_mar_total])))"

' if the data is good I should not need to do this
tPruneTmpQueryDuplesStr2 = "UPDATE ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 INNER JOIN " +
"ZZ_KIN_LIST_TMP ON (ZZ_KIN_LIST_TMP_1.c_personid = ZZ_KIN_LIST_TMP.c_personid) " +
"AND (ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_kin_id) " +
"AND (ZZ_KIN_LIST_TMP_1.c_kin_code = ZZ_KIN_LIST_TMP.c_kin_code) " +
"SET ZZ_KIN_LIST_TMP.c_delete = 1 " +
"WHERE (((StrComp([ZZ_KIN_LIST_TMP].[c_kinrel_total_raw], [ZZ_KIN_LIST_TMP_1].[c_kinrel_total_raw])) > 0)
)"

tPruneInversesQueryStr1 = "UPDATE ZZ_KIN_LIST INNER JOIN (KINSHIP_CODES INNER JOIN ZZ_KIN_LIST_TMP ON KINSHIP
_CODES.c_kincode = ZZ_KIN_LIST_TMP.c_kin_code) ON " +
"(ZZ_KIN_LIST.c_kin_id = ZZ_KIN_LIST_TMP.c_personid) AND (ZZ_KIN_LIST.c_personid = ZZ_KIN_LIST_TMP.c_kin
id) SET ZZ_KIN_LIST_TMP.c_delete = 1 " +
"WHERE (((ZZ_KIN_LIST.c_kin_code)=[KINSHIP_CODES].[c_kin_pair1])) OR (((ZZ_KIN_LIST.c_kin_code)=[KINSHIP
_CODES].[c_kin_pair2]))"

tPruneInversesQueryStr2 = "UPDATE (ZZ_KIN_LIST INNER JOIN ZZ_KIN_LIST_TMP ON (ZZ_KIN_LIST.c_personid = ZZ_KIN
_LIST_TMP.c_kin_id) AND " +
"(ZZ_KIN_LIST.c_kin_id = ZZ_KIN_LIST_TMP.c_personid)) INNER JOIN KINSHIP_CODES ON ZZ_KIN_LIST.c_kin_code
= KINSHIP_CODES.c_kincode SET ZZ_KIN_LIST_TMP.c_delete = 1 " +
"WHERE (((ZZ_KIN_LIST_TMP.c_kin_code)=[KINSHIP_CODES].[c_kin_pair1] Or (ZZ_KIN_LIST_TMP.c_kin_code)=[KINS
HIP_CODES].[c_kin_pair2]))"

tPruneTmpInversesQueryStr1 = "UPDATE KINSHIP_CODES INNER JOIN (ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 " +
"INNER JOIN ZZ_KIN_LIST_TMP ON (ZZ_KIN_LIST_TMP_1.c_personid = ZZ_KIN_LIST_TMP.c_kin_id) AND " +
"(ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_personid)) ON " +
"KINSHIP_CODES.c_kincode = ZZ_KIN_LIST_TMP.c_kin_code SET ZZ_KIN_LIST_TMP.c_delete = 1 " +
"WHERE (((ZZ_KIN_LIST_TMP.c_distance)>[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " +
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair1])) OR " +
"((ZZ_KIN_LIST_TMP.c_distance)=[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " +
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair1]) AND " +
"((ZZ_KIN_LIST_TMP.c_personid)>[ZZ_KIN_LIST_TMP_1].[c_personid])) OR " +
"((ZZ_KIN_LIST_TMP.c_distance)>[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " +
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair2])) OR " +
"((ZZ_KIN_LIST_TMP.c_distance)=[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " +
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair2]) AND " +
"((ZZ_KIN_LIST_TMP.c_personid)>[ZZ_KIN_LIST_TMP_1].[c_personid]))"

tPruneTmpInversesQueryStr2 = "UPDATE KINSHIP_CODES INNER JOIN (ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 " +
"INNER JOIN ZZ_KIN_LIST_TMP ON (ZZ_KIN_LIST_TMP_1.c_personid = ZZ_KIN_LIST_TMP.c_kin_id) AND " +
"(ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_personid)) ON " +
"KINSHIP_CODES.c_kincode = ZZ_KIN_LIST_TMP.c_kin_code SET ZZ_KIN_LIST_TMP.c_delete = 1 " +
"WHERE (((ZZ_KIN_LIST_TMP.c_distance)<[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " +
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair1])) OR " +
"((ZZ_KIN_LIST_TMP.c_distance)=[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " +
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair1]) AND " +

```

```

"((ZZ_KIN_LIST_TMP.c_personid)<[ZZ_KIN_LIST_TMP_1].[c_personid])) OR " +
"((ZZ_KIN_LIST_TMP.c_distance)<[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " +
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair2])) OR " +
"((ZZ_KIN_LIST_TMP.c_distance)=[ZZ_KIN_LIST_TMP_1].[c_distance]) AND " +
"((ZZ_KIN_LIST_TMP_1.c_kin_code)=[KINSHIP_CODES].[c_kin_pair2]) AND " +
"((ZZ_KIN_LIST_TMP.c_personid)<[ZZ_KIN_LIST_TMP_1].[c_personid]))"

```

```

tAppendQueryStr = "INSERT INTO ZZ_KIN_LIST ( c_personid, c_kin_id, c_kin_code, c_personid_root, c_kinrel, " +
" c_kinrel_total, c_kinrel_total_raw, c_kinrel_total_simplified, c_up, c_down, c_col, c_mar, c_up_total, c_down_total, c_col_total, " +
" c_mar_total, c_distance, c_female, c_sex, c_kin_female, c_kin_sex, c_prior_female, c_notes, c_source, c_source_text_chn, c_source_text ) " +
"SELECT DISTINCT ZZ_KIN_LIST_TMP.c_personid, ZZ_KIN_LIST_TMP.c_kin_id, ZZ_KIN_LIST_TMP.c_kin_code, " +
"ZZ_KIN_LIST_TMP.c_personid_root, ZZ_KIN_LIST_TMP.c_kinrel, ZZ_KIN_LIST_TMP.c_kinrel_total, " +
"ZZ_KIN_LIST_TMP.c_kinrel_total_raw, ZZ_KIN_LIST_TMP.c_kinrel_total_simplified, " +
"ZZ_KIN_LIST_TMP.c_up, ZZ_KIN_LIST_TMP.c_down, ZZ_KIN_LIST_TMP.c_col, ZZ_KIN_LIST_TMP.c_mar, " +
"ZZ_KIN_LIST_TMP.c_up_total, ZZ_KIN_LIST_TMP.c_down_total, ZZ_KIN_LIST_TMP.c_col_total, " +
"ZZ_KIN_LIST_TMP.c_mar_total, ZZ_KIN_LIST_TMP.c_distance, ZZ_KIN_LIST_TMP.c_female, ZZ_KIN_LIST_TMP.c_sex, " +
"ZZ_KIN_LIST_TMP.c_kin_female, ZZ_KIN_LIST_TMP.c_kin_sex, ZZ_KIN_LIST_TMP.c_prior_female, ZZ_KIN_LIST_TMP.c_notes, " +
"ZZ_KIN_LIST_TMP.c_source, ZZ_KIN_LIST_TMP.c_source_text_chn, ZZ_KIN_LIST_TMP.c_source_text " +
"FROM ZZ_KIN_LIST_TMP"

```

```

tLoopCount = 1
tExitDo = False
tLoopMax = TxtMaxLoop.Value

```

```

Do While tLoopCount <= tLoopMax And tRecCount > 0

```

```

    If tLoopCount = 1 Then

```

```

        ' MsgBox "Running first query"

```

```

        cmdSQL.CommandText = tKinFirstQueryStr

```

```

    Else

```

```

        ' MsgBox "Running query"

```

```

        cmdSQL.CommandText = tKinQueryStr + Str(tLoopCount - 1) + ")"

```

```

    End If

```

```

    cmdSQL.Execute tRecCount

```

```

    If tRecCount > 0 Then

```

```

        '

```

```

        ' process the results for addition

```

```

        '

```

```

        ' update the distance

```

```

        '

```

```

        'MsgBox "Fixing node distance"

```

```

        cmdSQL.CommandText = tNodeDistQueryStr

```

```

        cmdSQL.Execute tRecDelete

```

```

        '

```

```

        ' then mark the duplicates and delete them

```

```

        '

```

```

        'MsgBox "Fixing dupes 1"

```

```

        cmdSQL.CommandText = tPruneTmpQuery

```

```

        cmdSQL.Execute tRecDelete

```

```

        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"

```

```

        cmdSQL.Execute tRecDelete

```

```

        '

```

```

        'MsgBox "Fixing dupes 2"

```

```

        cmdSQL.CommandText = tPruneTmpQuery2

```

```

        cmdSQL.Execute tRecDelete

```

```

        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"

```

```

        cmdSQL.Execute tRecDelete

```

```

        '

```

```

        'MsgBox "Fixing dupes 3"

```

```

        cmdSQL.CommandText = tPruneTmpQueryDuplesStr

```

```

        cmdSQL.Execute tRecDelete

```

```

        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"

```

```

        cmdSQL.Execute tRecDelete

```

```

        '

```

```

        'MsgBox "Fixing dupes 4"

```

```

        cmdSQL.CommandText = tPruneTmpQueryDuplesStr2

```

```

        cmdSQL.Execute tRecDelete

```

```

        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"

```

```

        cmdSQL.Execute tRecDelete

```

```

        '

```

```

        'MsgBox "Fixing inverses"

```

```

        cmdSQL.CommandText = tPruneTmpInversesQueryStr1

```

```

        cmdSQL.Execute tRecDelete

```

```

        cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"

```

```

        cmdSQL.Execute tRecDelete

```

```

'
cmdSQL.CommandText = tPruneInversesQueryStr1
cmdSQL.Execute tRecDelete
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = tPruneInversesQueryStr2
cmdSQL.Execute tRecDelete
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDelete
'
' now simplify the kinship string, add to the total and reduce if possible
'
'cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP " + _
'    "SET ZZ_KIN_LIST_TMP.c_kinrel_total = 'ego', " + _
'        "ZZ_KIN_LIST_TMP.c_up_total = 0, " + _
'        "ZZ_KIN_LIST_TMP.c_down_total = 0, " + _
'        "ZZ_KIN_LIST_TMP.c_col_total = 0, " + _
'        "ZZ_KIN_LIST_TMP.c_mar_total = 0, " + _
'        "ZZ_KIN_LIST_TMP.c_kinrel_total_raw = 'ego' " + _
'        "WHERE (([ZZ_KIN_LIST_TMP].[c_kin_id]=[ZZ_KIN_LIST_TMP].[c_personid_root]))"
'cmdSQL.Execute tRecDelete
'
' Reduce kinship strings
'
' first just get the string length

cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP SET ZZ_KIN_LIST_TMP.c_kinrel_len = Len([ZZ_KIN_LIST_TMP].[c_kinrel_total])"
cmdSQL.Execute tRecDelete
'
' then deal with len = 2
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP " + _
    "SET ZZ_KIN_LIST_TMP.c_kinrel_root_text = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-2), " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_test_text = Right([ZZ_KIN_LIST_TMP].[c_kinrel_total],2), " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total_simplified],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-2) " + _
    "WHERE (((ZZ_KIN_LIST_TMP.c_kinrel_len)=2))"
cmdSQL.Execute tRecDelete
'
' replace where relevant

cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " + _
    "ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " + _
    "SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZ_KINREL_REDUCTION].[c_kinrel_replacement], " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified + " + _
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' +[ZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ')', " + _
    "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " + _
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' +[ZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ') ', " + _
    "ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total]+[ZZ_KINREL_REDUCTION].[c_up_change], " + _
    "ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total]+[ZZ_KINREL_REDUCTION].[c_down_change], " + _
    "ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total]+[ZZ_KINREL_REDUCTION].[c_col_change], " + _
    "ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total]+[ZZ_KINREL_REDUCTION].[c_mar_change] " + _
    "WHERE (((ZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null AND ZZZ_KINREL_REDUCTION.c_required = 1))"
cmdSQL.Execute tRecDelete
'
' then deal with len > 2
'
' copy the target string and string root

cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP " + _
    "SET ZZ_KIN_LIST_TMP.c_kinrel_root_text = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-2), " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_test_text = Right([ZZ_KIN_LIST_TMP].[c_kinrel_total],2), " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total_simplified],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-2) " + _
    "WHERE (((ZZ_KIN_LIST_TMP.c_kinrel_len)>2))"
cmdSQL.Execute tRecDelete

```

```

'
'   replace where relevant

cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " +
    "ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " +
    "SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZZ_KINREL_REDUCTION].[c_kinrel_replacement], " +
    "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified +
    " +
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ')', " +
    "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " +
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ') ', " +
    "ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total]+[ZZZ_KINREL_REDUCTION].[c_up_change], " +
    "ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total]+[ZZZ_KINREL_REDUCTION].[c_down_change], " +
    "ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total]+[ZZZ_KINREL_REDUCTION].[c_col_change], " +
    "ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total]+[ZZZ_KINREL_REDUCTION].[c_mar_change] " +
    "WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null AND ZZZ_KINREL_REDUCTION.c_required = FALSE))"
cmdSQL.Execute tRecDelete
'
'   the next set of procedures to simplify the code seems to need to be carried out in 6 parts (B, M, and F; 3 and 2)
'
If ChkSimplify.Value Then
    '   first, prep the ZZ_KIN_LIST_TMP file for codes of length three, with ZZZ_KINREL_REDUCTION.c_required = FALSE

    '   then get the B as simpler
    cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP " +
        "SET ZZ_KIN_LIST_TMP.c_kinrel_root_text = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-3), " +
        "ZZ_KIN_LIST_TMP.c_kinrel_test_text = Right([ZZ_KIN_LIST_TMP].[c_kinrel_total],3), " +
        "ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total_simplified],[ZZ_KIN_LIST_TMP].[c_kinrel_len]-3) " +
        "WHERE (((ZZ_KIN_LIST_TMP.c_kinrel_len)>2))"
    cmdSQL.Execute tRecDelete
    '
    '   replace where relevant

    cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " +
        "ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " +
        "SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZZ_KINREL_REDUCTION].[c_kinrel_replacement], " +
        "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified + " +
        "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ')', " +
        "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " +
        "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ') ', " +
        "ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total]+[ZZZ_KINREL_REDUCTION].[c_up_change], " +
        "ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total]+[ZZZ_KINREL_REDUCTION].[c_down_change], " +
        "ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total]+[ZZZ_KINREL_REDUCTION].[c_col_change], " +
        "ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total]+[ZZZ_KINREL_REDUCTION].[c_mar_change] " +
        "WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null AND Not ZZZ_KINREL_REDUCTION.c_required AND (ZZZ_KINREL_REDUCTION.c_sex = 'B')))"
    cmdSQL.Execute tRecDelete

    '   then M

    cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " +
        "ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " +
        "SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZZ_KINREL_REDUCTION].[c_kinrel_replacement], " +
        "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified + " +
        "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ')', " +
        "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " +

```

```

        "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kin
rel_replacement] + ')', " + _
        "ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total] + [ZZZ_KINREL_REDUCTION].[c_up
_change], " + _
        "ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total] + [ZZZ_KINREL_REDUCTION].[c
_down_change], " + _
        "ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total] + [ZZZ_KINREL_REDUCTION].[c
_col_change], " + _
        "ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total] + [ZZZ_KINREL_REDUCTION].[c
_mar_change] " + _
        "WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null AND Not ZZZ_KINREL_REDUCTION.c_re
quired AND " + _
        "(ZZZ_KINREL_REDUCTION.c_sex = 'M' AND NOT ZZZ_KINREL_REDUCTION.c_check_ego AND NOT ZZ
_KIN_LIST_TMP.c_prior_female)))"
        cmdSQL.Execute tRecDelete

    ' then F

    'cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " + _
    "ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " + _
    "SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text] + [ZZZ_KINREL_REDU
CTION].[c_kinrel_replacement], " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplifie
d + " + _
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kin
rel_replacement] + ')', " + _
    "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " + _
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kin
rel_replacement] + ')', " + _
    "ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total] + [ZZZ_KINREL_REDUCTION].[c_up
_change], " + _
    "ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total] + [ZZZ_KINREL_REDUCTION].[c
_down_change], " + _
    "ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total] + [ZZZ_KINREL_REDUCTION].[c
_col_change], " + _
    "ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total] + [ZZZ_KINREL_REDUCTION].[c
_mar_change] " + _
    "WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null AND Not ZZZ_KINREL_REDUCTION.c_re
quired AND " + _
    "(ZZZ_KINREL_REDUCTION.c_sex = 'F' AND NOT ZZZ_KINREL_REDUCTION.c_check_ego AND ZZ_KIN
_LIST_TMP.c_prior_female)))"
    'cmdSQL.Execute tRecDelete

    ' next, prep the ZZ_KIN_LIST_TMP file for codes of lenght 2: at least the code here is the same
except for string lengths

    ' then get the B as simpler

    cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP " + _
    "SET ZZ_KIN_LIST_TMP.c_kinrel_root_text = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total], [ZZ_KIN_LIS
T_TMP].[c_kinrel_len]-2), " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_test_text = Right([ZZ_KIN_LIST_TMP].[c_kinrel_total], 2), " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified = Left([ZZ_KIN_LIST_TMP].[c_kinrel_total_s
implified], [ZZ_KIN_LIST_TMP].[c_kinrel_len]-2) " + _
    "WHERE (((ZZ_KIN_LIST_TMP.c_kinrel_len) > 1))"
    cmdSQL.Execute tRecDelete
    '
    ' replace where relevant

    cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " + _
    "ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " + _
    "SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text] + [ZZZ_KINREL_REDU
CTION].[c_kinrel_replacement], " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplifie
d + " + _
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kin
rel_replacement] + ')', " + _
    "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " + _
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kin
rel_replacement] + ')', " + _
    "ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total] + [ZZZ_KINREL_REDUCTION].[c_up
_change], " + _
    "ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total] + [ZZZ_KINREL_REDUCTION].[c
_down_change], " + _
    "ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total] + [ZZZ_KINREL_REDUCTION].[c
_col_change], " + _
    "ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total] + [ZZZ_KINREL_REDUCTION].[c
_mar_change] " + _
    "WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null AND Not ZZZ_KINREL_REDUCTION.c_re
quired AND (ZZZ_KINREL_REDUCTION.c_sex = 'B')))"

```

```

cmdSQL.Execute tRecDelete

' then M

cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " +
    "ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " +
    "SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZZ_KINREL_REDUCTION].[c_kinrel_replacement], " +
    "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified + " +
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ')', " +
    "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " +
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ')', " +
    "ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total]+[ZZZ_KINREL_REDUCTION].[c_up_change], " +
    "ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total]+[ZZZ_KINREL_REDUCTION].[c_down_change], " +
    "ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total]+[ZZZ_KINREL_REDUCTION].[c_col_change], " +
    "ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total]+[ZZZ_KINREL_REDUCTION].[c_mar_change] " +
    "WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null AND Not ZZZ_KINREL_REDUCTION.c_required AND " +
    "(ZZZ_KINREL_REDUCTION.c_sex = 'M' AND NOT ZZZ_KINREL_REDUCTION.c_check_ego AND NOT ZZ_KIN_LIST_TMP.c_prior_female)))"
cmdSQL.Execute tRecDelete

' then F

cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP " +
    "ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " +
    "SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZZ_KINREL_REDUCTION].[c_kinrel_replacement], " +
    "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified + " +
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ')', " +
    "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes] + " +
    "'(' + ZZZ_KINREL_REDUCTION.c_kinrel_target + '>' + [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement] + ')', " +
    "ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total]+[ZZZ_KINREL_REDUCTION].[c_up_change], " +
    "ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total]+[ZZZ_KINREL_REDUCTION].[c_down_change], " +
    "ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total]+[ZZZ_KINREL_REDUCTION].[c_col_change], " +
    "ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total]+[ZZZ_KINREL_REDUCTION].[c_mar_change] " +
    "WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null AND Not ZZZ_KINREL_REDUCTION.c_required AND " +
    "(ZZZ_KINREL_REDUCTION.c_sex = 'F' AND NOT ZZZ_KINREL_REDUCTION.c_check_ego AND ZZ_KIN_LIST_TMP.c_prior_female)))"
cmdSQL.Execute tRecDelete

' final, prep the ZZ_KIN_LIST_TMP file for codes of length two, with ZZZ_KINREL_REDUCTION.c_ego_check = TRUE

cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP " +
    "SET ZZ_KIN_LIST_TMP.c_kinrel_root_text = '', " +
    "ZZ_KIN_LIST_TMP.c_kinrel_test_text = [ZZ_KIN_LIST_TMP].[c_kinrel_total_simplified], " +
    "ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified = '' " +
    "WHERE ((len([ZZ_KIN_LIST_TMP].[c_kinrel_total_simplified])=2))"
cmdSQL.Execute tRecDelete

' then F

cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP ON ZZZ_KINREL_REDUCTION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " +
    "SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZZ_KINREL_REDUCTION].[c_kinrel_replacement], " +
    "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplified + ('+ZZZ_KINREL_REDUCTION.c_kinrel_target+>'" +
    "+[ZZZ_KINREL_REDUCTION].[c_kinrel_replacement]+')', " +
    "ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes]+'(' +ZZZ_KINREL_REDUCTION.c_kinrel_target+>'" +
    "[ZZZ_KINREL_REDUCTION].[c_kinrel_replacement]+')', " +

```

```

ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total]+[ZZZ_KINREL_REDUCTION].[c_up
_change], " + _
ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total]+[ZZZ_KINREL_REDUCTION].[
c_down_change], " + _
ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total]+[ZZZ_KINREL_REDUCTION].[c
col_change], " + _
ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total]+[ZZZ_KINREL_REDUCTION].[c
mar_change] " + _
"WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null) AND ((ZZZ_KINREL_REDUCTION.c_req
uired)=False) AND " + _
"((ZZZ_KINREL_REDUCTION.c_sex)='F') AND ((ZZ_KIN_LIST_TMP.c_kin_female)=True) AND ((ZZZ_K
INREL_REDUCTION.c_check_ego)=True) AND " + _
"((ZZ_KIN_LIST_TMP.c_personid)<>[ZZ_KIN_LIST_TMP].[c_kin_id]))"
cmdSQL.Execute tRecDelete

cmdSQL.CommandText = "UPDATE ZZZ_KINREL_REDUCTION RIGHT JOIN ZZ_KIN_LIST_TMP ON ZZZ_KINREL_REDUCT
ION.c_kinrel_target = ZZ_KIN_LIST_TMP.c_kinrel_test_text " + _
"SET ZZ_KIN_LIST_TMP.c_kinrel_total = [ZZ_KIN_LIST_TMP].[c_kinrel_root_text]+[ZZZ_KINREL_REDU
CTION].[c_kinrel_replacement], " + _
"ZZ_KIN_LIST_TMP.c_kinrel_total_simplified = ZZ_KIN_LIST_TMP.c_kinrel_root_text_simplifie
d+'(' +ZZZ_KINREL_REDUCTION.c_kinrel_target+'>' + _
"+[ZZZ_KINREL_REDUCTION].[c_kinrel_replacement]+')', " + _
"ZZ_KIN_LIST_TMP.c_notes = [ZZ_KIN_LIST_TMP].[c_notes]+'(' +ZZZ_KINREL_REDUCTION.c_kinrel
target+'>'+" + _
" [ZZZ_KINREL_REDUCTION].[c_kinrel_replacement]+')', " + _
"ZZ_KIN_LIST_TMP.c_up_total = [ZZ_KIN_LIST_TMP].[c_up_total]+[ZZZ_KINREL_REDUCTION].[c_up
_change], " + _
"ZZ_KIN_LIST_TMP.c_down_total = [ZZ_KIN_LIST_TMP].[c_down_total]+[ZZZ_KINREL_REDUCTION].[
c_down_change], " + _
"ZZ_KIN_LIST_TMP.c_col_total = [ZZ_KIN_LIST_TMP].[c_col_total]+[ZZZ_KINREL_REDUCTION].[c
col_change], " + _
"ZZ_KIN_LIST_TMP.c_mar_total = [ZZ_KIN_LIST_TMP].[c_mar_total]+[ZZZ_KINREL_REDUCTION].[c
mar_change] " + _
"WHERE (((ZZZ_KINREL_REDUCTION.c_kinrel_target) Is Not Null) AND ((ZZZ_KINREL_REDUCTION.c_req
uired)=False) AND " + _
"((ZZZ_KINREL_REDUCTION.c_sex)='M') AND ((ZZ_KIN_LIST_TMP.c_kin_female)=False) AND ((ZZZ_
KINREL_REDUCTION.c_check_ego)=True) AND " + _
"((ZZ_KIN_LIST_TMP.c_personid)<>[ZZ_KIN_LIST_TMP].[c_kin_id]))"
cmdSQL.Execute tRecDelete

End If
'
' now mark the records with bad metrics
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP SET ZZ_KIN_LIST_TMP.c_delete = 1 " + _
"WHERE (((ZZ_KIN_LIST_TMP.c_up_total)>" + Str(gMaxUp) + ")) OR " + _
"(((ZZ_KIN_LIST_TMP.c_down_total)>" + Str(gMaxDown) + ")) OR " + _
"(((ZZ_KIN_LIST_TMP.c_col_total)>" + Str(gMaxCol) + ")) OR " + _
"(((ZZ_KIN_LIST_TMP.c_mar_total)>" + Str(gMaxMarr) + "))"
cmdSQL.Execute tRecDelete
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDelete
'
' if the mourning filter needs to be applied, do so now
'
If ChkMourning.Value Then
    MsgBox "Filtering mourning circle"
    cmdSQL.CommandText = "UPDATE KIN_Mourning RIGHT JOIN ZZ_KIN_LIST_TMP " + _
    "ON KIN_Mourning.c_kinrel = ZZ_KIN_LIST_TMP.c_kinrel_total " + _
    "SET ZZ_KIN_LIST_TMP.c_delete = 1 " + _
    "WHERE (((KIN_Mourning.c_kinrel) Is Null))"
    cmdSQL.Execute tRecDelete
    cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
    cmdSQL.Execute tRecDelete
End If
'
' one final test: the only difference between ZZ_KIN_LIST_TEMP records is in PRIOR_FEMALE
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 INNER JOIN ZZ_KIN_LIST_TMP " + _
"ON (ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_kin_id) AND " + _
"(ZZ_KIN_LIST_TMP_1.c_personid = ZZ_KIN_LIST_TMP.c_personid) AND " + _
"(ZZ_KIN_LIST_TMP_1.c_kin_code = ZZ_KIN_LIST_TMP.c_kin_code) " + _
"SET ZZ_KIN_LIST_TMP.c_delete = 1 " + _
"WHERE (((ZZ_KIN_LIST_TMP.c_prior_female)=True) AND ((ZZ_KIN_LIST_TMP_1.c_prior_female)=False))"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete

```



```

' it turns out that getting rid of inverse records is tougher than I would like, so we try one last
time
'
cmdSQL.CommandText = tPruneTmpInversesQueryStr2
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' one last pair of clean-up routines is necessary. One can arrive at the same results through differ
ent paths
'
' first, take the shorter path
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 ON " + _
"(ZZ_KIN_LIST_TMP.c_personid_root = ZZ_KIN_LIST_TMP_1.c_personid_root) " + _
"AND (ZZ_KIN_LIST_TMP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id) AND (ZZ_KIN_LIST_TMP.c_personid = ZZ_KIN_LIST_TMP_1.c_personid) " + _
"AND (ZZ_KIN_LIST_TMP.c_kinrel = ZZ_KIN_LIST_TMP_1.c_kinrel) " + _
"SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " + _
"WHERE (((Len([ZZ_KIN_LIST_TMP].[c_notes])) < Len([ZZ_KIN_LIST_TMP_1].[c_notes])))"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' MsgBox "Last step"
'
' then take the string with the smaller value
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 ON " + _
"(ZZ_KIN_LIST_TMP.c_personid_root = ZZ_KIN_LIST_TMP_1.c_personid_root) " + _
"AND (ZZ_KIN_LIST_TMP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id) AND (ZZ_KIN_LIST_TMP.c_personid = ZZ_KIN_LIST_TMP_1.c_personid) " + _
"AND (ZZ_KIN_LIST_TMP.c_kinrel = ZZ_KIN_LIST_TMP_1.c_kinrel) " + _
"SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " + _
"WHERE ('X'+[ZZ_KIN_LIST_TMP].[c_notes] > 'X'+[ZZ_KIN_LIST_TMP_1].[c_notes])"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' now append the results: if no records are added, this should stop the looping
'
' MsgBox "Copying to ZZ_KIN_LIST"
cmdSQL.CommandText = tAppendQueryStr
cmdSQL.Execute tRecCount
'
' and clear ZZ_KIN_LIST_TMP
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP"
cmdSQL.Execute tRecDelete
'
End If

tLoopCount = tLoopCount + 1

If tLoopCount > tLoopMax Then
    MsgBox "Loop limit hit."
    tExitDo = True
    Exit Do
End If

Loop
'
' clean up the results
'
' MsgBox "Fixing ZZ_KIN_LIST inverses"
cmdSQL.CommandText = tAppendQueryStr
cmdSQL.Execute tRecDelete
'
' copy to the kinship table
'
' MsgBox "Copying to ZZ_SCRATCH_KINNET (not ego relative)"
' Set tAppendQuery = CurrentDb.QueryDefs("ZZ_KIN_LIST_APPEND kin Query")
' tAppendQuery.Execute
'
' split the long command into a mix + update person + update kin
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_KINNET ( c_person_id, c_name, c_name_chn, c_female, c_sex, c_index_year,
```

```

c_kin_id, " + _
    "c_kin_name, c_kin_chn, c_kin_index_year, c_kin_female, c_kin_sex, c_kin_code, c_kin_rel, c_up, c_down, c_
_collateral, " + _
    "c_marriage, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_addr_type, c_addr_desc, c_addr_desc_
chn, " + _
    "c_kin_addr_id, c_kin_addr_name, c_kin_addr_chn, kin_x_coord, kin_y_coord, c_notes, c_kin_addr_type, " + _
    "c_kin_addr_desc, c_kin_addr_desc_chn, c_upstep, c_dwnstep, c_marstep, c_colstep, c_distance, " + _
    "c_source, c_source_text_chn, c_source_text ) "
'tQueryStr = tQueryStr + _
    "SELECT DISTINCT ZZ_KIN_LIST.c_personid, ZZZ_KIN_BIOG_ADDR.c_person_name, " + _
    "ZZZ_KIN_BIOG_ADDR.c_person_name_chn, ZZZ_KIN_BIOG_ADDR.c_female, iif(ZZZ_KIN_BIOG_ADDR.c_female,'F','M'
), ZZZ_KIN_BIOG_ADDR.c_index_year, ZZZ_KIN_BIOG_ADDR.c_node_id, " + _
    "ZZZ_KIN_BIOG_ADDR.c_node_name, ZZZ_KIN_BIOG_ADDR.c_node_chn, ZZZ_KIN_BIOG_ADDR.c_node_index_year, " + _
    "ZZZ_KIN_BIOG_ADDR.c_node_female, iif(ZZZ_KIN_BIOG_ADDR.c_node_female,'F','M'), ZZZ_KIN_BIOG_ADDR.c_link_
code, ZZZ_KIN_BIOG_ADDR.c_link_desc, " + _
    "ZZ_KIN_LIST.c_up_total, ZZ_KIN_LIST.c_down_total, ZZ_KIN_LIST.c_col_total, ZZ_KIN_LIST.c_mar_total, " + _
    "ZZZ_KIN_BIOG_ADDR.c_addr_id, ZZZ_KIN_BIOG_ADDR.c_addr_name, ZZZ_KIN_BIOG_ADDR.c_addr_chn, " + _
    "ZZZ_KIN_BIOG_ADDR.x_coord, ZZZ_KIN_BIOG_ADDR.y_coord, ZZZ_KIN_BIOG_ADDR.c_addr_type, " + _
    "ZZZ_KIN_BIOG_ADDR.c_addr_desc, ZZZ_KIN_BIOG_ADDR.c_addr_desc_chn, ZZZ_KIN_BIOG_ADDR.c_node_addr_id, " + _
    "ZZZ_KIN_BIOG_ADDR.c_node_addr_name, ZZZ_KIN_BIOG_ADDR.c_node_addr_chn, ZZZ_KIN_BIOG_ADDR.node_xcoord, "
+ _
    "ZZZ_KIN_BIOG_ADDR.node_ycoord, ZZZ_KIN_BIOG_ADDR.c_notes, ZZZ_KIN_BIOG_ADDR.c_node_addr_type, " + _
    "ZZZ_KIN_BIOG_ADDR.c_node_addr_desc, ZZZ_KIN_BIOG_ADDR.c_node_addr_desc_chn, ZZZ_KIN_BIOG_ADDR.c_upstep,
" + _
    "ZZZ_KIN_BIOG_ADDR.c_dwnstep, ZZZ_KIN_BIOG_ADDR.c_marstep, ZZZ_KIN_BIOG_ADDR.c_colstep, " + _
    "ZZZ_KIN_BIOG_ADDR.c_distance, ZZ_KIN_LIST.c_source, ZZ_KIN_LIST.c_source_text_chn, ZZ_KIN_LIST.c_source_
text " + _
    "FROM ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_KIN_LIST ON (ZZZ_KIN_BIOG_ADDR.c_personid = ZZ_KIN_LIST.c_personid)
" + _
    "AND (ZZZ_KIN_BIOG_ADDR.c_link_code = ZZ_KIN_LIST.c_kin_code) AND " + _
    "(ZZZ_KIN_BIOG_ADDR.c_node_id = ZZ_KIN_LIST.c_kin_id)"

' insert

tQueryStr = "INSERT INTO ZZ_SCRATCH_KINNET ( c_person_id, c_kin_id, c_kin_code, c_kin_rel, c_source, c_source_
_text_chn, c_source_text) " + _
    "SELECT DISTINCT ZZ_KIN_LIST.c_personid, ZZ_KIN_LIST.c_kin_id, ZZ_KIN_LIST.c_kin_code, ZZ_KIN_LIS
T.c_kinrel, ZZ_KIN_LIST.c_source, " + _
    "ZZ_KIN_LIST.c_source_text_chn, ZZ_KIN_LIST.c_source_text " + _
    "FROM ZZ_KIN_LIST WHERE (((ZZ_KIN_LIST.c_personid)<>[ZZ_KIN_LIST].[c_kin_id]))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete
'
' update the person

tQueryStr = "UPDATE ZZ_SCRATCH_KINNET INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KINNET.c_person_id = ZZZ_BIOG_MA
IN.c_personid " + _
    "SET ZZ_SCRATCH_KINNET.c_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SCRATCH_KINNET.c_name_chn = [ZZZ_BIO
G_MAIN].[c_name_chn], " + _
    "ZZ_SCRATCH_KINNET.c_female = [ZZZ_BIOG_MAIN].[c_female], ZZ_SCRATCH_KINNET.c_sex = IIf([ZZZ_
BIOG_MAIN].[c_female], 'F', 'M'), " + _
    "ZZ_SCRATCH_KINNET.c_index_year = [ZZZ_BIOG_MAIN].[c_index_year], " + _
    "ZZ_SCRATCH_KINNET.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
    "ZZ_SCRATCH_KINNET.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
    "ZZ_SCRATCH_KINNET.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _
    "ZZ_SCRATCH_KINNET.x_coord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SCRATCH_KINNET.y_coord = [ZZZ_BIOG
_MAIN].[y_coord], " + _
    "ZZ_SCRATCH_KINNET.c_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " + _
    "ZZ_SCRATCH_KINNET.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " + _
    "ZZ_SCRATCH_KINNET.c_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn]"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete
'
' update the kin

tQueryStr = "UPDATE ZZ_SCRATCH_KINNET INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KINNET.c_kin_id = ZZZ_BIOG_MAIN.
c_personid " + _
    "SET ZZ_SCRATCH_KINNET.c_kin_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SCRATCH_KINNET.c_kin_chn = [ZZZ_
BIOG_MAIN].[c_name_chn], " + _
    "ZZ_SCRATCH_KINNET.c_kin_female = [ZZZ_BIOG_MAIN].[c_female], ZZ_SCRATCH_KINNET.c_kin_sex = I
f([ZZZ_BIOG_MAIN].[c_female], 'F', 'M'), " + _
    "ZZ_SCRATCH_KINNET.c_kin_index_year = [ZZZ_BIOG_MAIN].[c_index_year], " + _
    "ZZ_SCRATCH_KINNET.c_kin_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
    "ZZ_SCRATCH_KINNET.c_kin_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], " + _
    "ZZ_SCRATCH_KINNET.c_kin_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " + _

```

```

        "ZZ_SCRATCH_KINNET.kin_x_coord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SCRATCH_KINNET.kin_y_coord = [
ZZZ_BIOG_MAIN].[y_coord], " +
        "ZZ_SCRATCH_KINNET.c_kin_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " +
        "ZZ_SCRATCH_KINNET.c_kin_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], " +
        "ZZ_SCRATCH_KINNET.c_kin_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn]"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete
'
' update the relation
'
tQueryStr = "UPDATE ZZ_SCRATCH_KINNET INNER JOIN ZZZ_KIN_BIOG_ADDR ON (ZZ_SCRATCH_KINNET.c_kin_code = ZZZ_KIN
_BIOG_ADDR.c_link_code) AND " +
        "(ZZ_SCRATCH_KINNET.c_kin_id = ZZZ_KIN_BIOG_ADDR.c_node_id) AND (ZZ_SCRATCH_KINNET.c_person_id
d = ZZZ_KIN_BIOG_ADDR.c_personid) " +
        "SET ZZ_SCRATCH_KINNET.c_notes = ZZZ_KIN_BIOG_ADDR.c_notes, " +
        "ZZ_SCRATCH_KINNET.c_upstep = ZZZ_KIN_BIOG_ADDR.c_upstep, ZZ_SCRATCH_KINNET.c_dwnstep = ZZZ_K
IN_BIOG_ADDR.c_dwnstep, " +
        "ZZ_SCRATCH_KINNET.c_marstep = ZZZ_KIN_BIOG_ADDR.c_marstep, ZZ_SCRATCH_KINNET.c_colstep = ZZZ
_KIN_BIOG_ADDR.c_colstep, " +
        "ZZ_SCRATCH_KINNET.c_distance = ZZZ_KIN_BIOG_ADDR.c_distance "
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete
'
' the final step is to add the index year descriptive information
'
cmdSQL.CommandText = "UPDATE (ZZ_SCRATCH_KINNET INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KINNET.c_person_id = Z
ZZ_BIOG_MAIN.c_personid) " +
        "INNER JOIN ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 ON ZZ_SCRATCH_KINNET.c_kin_id = ZZZ_BIOG_MAIN_1.c_personid "
+
        "SET ZZ_SCRATCH_KINNET.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " +
        "ZZ_SCRATCH_KINNET.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " +
        "ZZ_SCRATCH_KINNET.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " +
        "ZZ_SCRATCH_KINNET.c_kin_index_year_type_code = [ZZZ_BIOG_MAIN_1].[c_index_year_type_code], " +
        "ZZ_SCRATCH_KINNET.c_kin_index_year_type_desc = [ZZZ_BIOG_MAIN_1].[c_index_year_type_desc], " +
        "ZZ_SCRATCH_KINNET.c_kin_index_year_type_hz = [ZZZ_BIOG_MAIN_1].[c_index_year_type_hz]"
cmdSQL.Execute tRecDelete
'
' copy to the ego-relative kinship table
'
' Before copying we need to clean up the data
'
' There is a bug in the algorithm that creates the occasional null value in c_kinrel_total. To debug, for t
he moment plug the hole
'
'MsgBox "Patching NULL bug"
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST SET ZZ_KIN_LIST.c_kinrel_total_simplified = '[Program Error]' WHERE
((ZZ_KIN_LIST.c_kinrel_total_simplified) Is Null)"
cmdSQL.Execute tRecDelete
'
'MsgBox "Inserting ego-relative"
tQueryStr = "INSERT INTO ZZ_KIN_LIST_TMP ( c_personid, c_kin_id, c_kinrel, c_kinrel_total, c_kinrel_total_sim
plified, " +
        "c_up, c_down, c_col, c_mar, c_notes, c_kin_code ) " +
        "SELECT DISTINCT ZZ_KIN_LIST.c_personid_root, ZZ_KIN_LIST.c_kin_id, ZZ_KIN_LIST.c_kinrel_total_raw, ZZ_KI
N_LIST.c_kinrel_total, " +
        "ZZ_KIN_LIST.c_kinrel_total_simplified, ZZ_KIN_LIST.c_up_total, ZZ_KIN_LIST.c_down_total, ZZ_KIN_LIST
.c_col_total, ZZ_KIN_LIST.c_mar_total, " +
        "ZZ_KIN_LIST.c_notes, 0 AS c_kin_code " +
        "FROM ZZ_KIN_LIST"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete

' first just get the string length

cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP SET ZZ_KIN_LIST_TMP.c_kinrel_len = Len([ZZ_KIN_LIST_TMP].[c_kinr
el_total_simplified])"
cmdSQL.Execute tRecDelete

' delete the longer strings (this may solve most of the problems)

cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 ON ZZ_KIN_LIST_T
MP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id " +
        "SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " +
        "WHERE ([ZZ_KIN_LIST_TMP_1].[c_kinrel_len]>[ZZ_KIN_LIST_TMP].[c_kinrel_len])"
cmdSQL.Execute tRecDelete

cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE ZZ_KIN_LIST_TMP.c_delete = 1"
cmdSQL.Execute tRecDelete

```

```

'
' the next version uses the string-compare function because sometimes the strings are of the same length
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 INNER JOIN " + _
    "ZZ_KIN_LIST_TMP ON (ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_kin_id) " + _
    "SET ZZ_KIN_LIST_TMP.c_delete = 1 " + _
    "WHERE ((StrComp([ZZ_KIN_LIST_TMP].[c_kinrel_total_simplified], [ZZ_KIN_LIST_TMP_1].[c_kinrel_total_simp" + _
    "lified])) > 0))"
cmdSQL.Execute tRecDelete

cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' the last version uses the total kinship path: take the shortest value
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 INNER JOIN " + _
    "ZZ_KIN_LIST_TMP ON (ZZ_KIN_LIST_TMP_1.c_kin_id = ZZ_KIN_LIST_TMP.c_kin_id) " + _
    "SET ZZ_KIN_LIST_TMP.c_delete = 1 " + _
    "WHERE (([ZZ_KIN_LIST_TMP].[c_down] + [ZZ_KIN_LIST_TMP].[c_col] + [ZZ_KIN_LIST_TMP].[c_mar] + [ZZ_KIN_LIS" + _
    "T_TMP].[c_up]>" + [ZZ_KIN_LIST_TMP_1].[c_down]+[ZZ_KIN_LIST_TMP_1].[c_col]+[ZZ_KIN_LIST_TMP_1].[c_mar]+[ZZ_KIN_LIS" + _
    "T_TMP_1].[c_up]))"
cmdSQL.Execute tRecDelete

cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
' one last clean-up: remove results that are the same but takes a longer path to get there
'
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 " + _
    "ON (ZZ_KIN_LIST_TMP.c_kinrel = ZZ_KIN_LIST_TMP_1.c_kinrel) AND (ZZ_KIN_LIST_TMP.c_personid = ZZ_KIN_LIST" + _
    "TMP_1.c_personid) AND " + _
    "(ZZ_KIN_LIST_TMP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id) " + _
    "SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " + _
    "WHERE ((Len([ZZ_KIN_LIST_TMP].[c_notes])<Len([ZZ_KIN_LIST_TMP_1].[c_notes])))"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
'MsgBox "Last step"
cmdSQL.CommandText = "UPDATE ZZ_KIN_LIST_TMP INNER JOIN ZZ_KIN_LIST_TMP AS ZZ_KIN_LIST_TMP_1 " + _
    "ON (ZZ_KIN_LIST_TMP.c_kinrel = ZZ_KIN_LIST_TMP_1.c_kinrel) AND (ZZ_KIN_LIST_TMP.c_personid = ZZ_KIN_LIST" + _
    "TMP_1.c_personid) AND " + _
    "(ZZ_KIN_LIST_TMP.c_kin_id = ZZ_KIN_LIST_TMP_1.c_kin_id) " + _
    "SET ZZ_KIN_LIST_TMP_1.c_delete = 1 " + _
    "WHERE ('X'+[ZZ_KIN_LIST_TMP].[c_notes] > 'X'+[ZZ_KIN_LIST_TMP_1].[c_notes]))"
cmdSQL.Execute tRecDelete
'
cmdSQL.CommandText = "DELETE * FROM ZZ_KIN_LIST_TMP WHERE c_delete = 1"
cmdSQL.Execute tRecDelete
'
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_KIN ( c_person_id, c_kin_id, c_kin_rel, c_kin_rel_total, c_kin_rel_0, c_u" + _
    "p, c_down, c_collateral, c_marriage, c_notes ) " + _
    "SELECT DISTINCT ZZ_KIN_LIST_TMP.c_personid, ZZ_KIN_LIST_TMP.c_kin_id, ZZ_KIN_LIST_TMP.c_kinrel, ZZ_KIN_L" + _
    "IST_TMP.c_kinrel_total, " + _
    "ZZ_KIN_LIST_TMP.c_kinrel_total_simplified, ZZ_KIN_LIST_TMP.c_up, ZZ_KIN_LIST_TMP.c_down, ZZ_KIN_LIST" + _
    "TMP.c_col, ZZ_KIN_LIST_TMP.c_mar, ZZ_KIN_LIST_TMP.c_notes " + _
    "FROM ZZ_KIN_LIST_TMP"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete

tQueryStr = "UPDATE (ZZZ_BIOG_MAIN INNER JOIN ZZ_SCRATCH_KIN ON ZZZ_BIOG_MAIN.c_personid = ZZ_SCRATCH_KIN.c_p" + _
    "erson_id) " + _
    "INNER JOIN ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 ON ZZ_SCRATCH_KIN.c_kin_id = ZZZ_BIOG_MAIN_1.c_personid " + _
    "SET ZZ_SCRATCH_KIN.c_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SCRATCH_KIN.c_name_chn = [ZZZ_BIOG_MAIN].[c_nam" + _
    "e_chn], " + _
    "ZZ_SCRATCH_KIN.c_index_year = [ZZZ_BIOG_MAIN].[c_index_year], ZZ_SCRATCH_KIN.c_female = [ZZZ_BIOG_MAIN]." + _
    "[c_female], " + _
    "ZZ_SCRATCH_KIN.c_sex = iif([ZZZ_BIOG_MAIN].[c_female], 'F', 'M'), " + _
    "ZZ_SCRATCH_KIN.c_kin_name = [ZZZ_BIOG_MAIN_1].[c_name], ZZ_SCRATCH_KIN.c_kin_chn = [ZZZ_BIOG_MAIN_1].[c" + _
    "name_chn], " + _
    "ZZ_SCRATCH_KIN.c_kin_index_year = [ZZZ_BIOG_MAIN_1].[c_index_year], ZZ_SCRATCH_KIN.c_kin_female = [ZZZ_B" + _
    "IOG_MAIN_1].[c_female], " + _
    "ZZ_SCRATCH_KIN.c_kin_sex = iif([ZZZ_BIOG_MAIN_1].[c_female], 'F', 'M'), ZZ_SCRATCH_KIN.c_kin_code = 0, ZZ" + _
    "SCRATCH_KIN.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " + _
    "ZZ_SCRATCH_KIN.c_addr_name = [ZZZ_BIOG_MAIN].[c_index_addr_name], ZZ_SCRATCH_KIN.c_addr_chn = [ZZZ_BIOG"

```

```

MAIN].[c_index_addr_chn], " + _
    "ZZ_SCRATCH_KIN.c_addr_type = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], ZZ_SCRATCH_KIN.c_addr_desc = [ZZZ
_BIOG_MAIN].[c_index_addr_type_desc], " + _
    "ZZ_SCRATCH_KIN.c_addr_desc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " + _
    "ZZ_SCRATCH_KIN.x_coord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SCRATCH_KIN.y_coord = [ZZZ_BIOG_MAIN].[y_coord],
" + _
    "ZZ_SCRATCH_KIN.c_kin_addr_id = [ZZZ_BIOG_MAIN_1].[c_index_addr_id], ZZ_SCRATCH_KIN.c_kin_addr_name = [ZZ
Z_BIOG_MAIN_1].[c_index_addr_name], " + _
    "ZZ_SCRATCH_KIN.c_kin_addr_chn = [ZZZ_BIOG_MAIN_1].[c_index_addr_chn], ZZ_SCRATCH_KIN.c_kin_addr_type = [
ZZZ_BIOG_MAIN_1].[c_index_addr_type_code], " + _
    "ZZ_SCRATCH_KIN.c_kin_addr_desc = [ZZZ_BIOG_MAIN_1].[c_index_addr_type_desc], ZZ_SCRATCH_KIN.c_kin_addr_d
esc_chn = [ZZZ_BIOG_MAIN_1].[c_index_addr_type_chn], " + _
    "ZZ_SCRATCH_KIN.kin_x_coord = [ZZZ_BIOG_MAIN_1].[x_coord], ZZ_SCRATCH_KIN.kin_y_coord = [ZZZ_BIOG_MAIN_1]
.[y_coord];"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDelete
'
' get the index year descriptive data
'
cmdSQL.CommandText = "UPDATE (ZZ_SCRATCH_KIN INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KIN.c_person_id = ZZZ_BIO
G_MAIN.c_personid) " + _
    "INNER JOIN ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 ON ZZ_SCRATCH_KIN.c_kin_id = ZZZ_BIOG_MAIN_1.c_personid " + _
    "SET ZZ_SCRATCH_KIN.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
    "ZZ_SCRATCH_KIN.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
    "ZZ_SCRATCH_KIN.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " + _
    "ZZ_SCRATCH_KIN.c_kin_index_year_type_code = [ZZZ_BIOG_MAIN_1].[c_index_year_type_code], " + _
    "ZZ_SCRATCH_KIN.c_kin_index_year_type_desc = [ZZZ_BIOG_MAIN_1].[c_index_year_type_desc], " + _
    "ZZ_SCRATCH_KIN.c_kin_index_year_type_hz = [ZZZ_BIOG_MAIN_1].[c_index_year_type_hz]"
cmdSQL.Execute tRecDelete
'
' update the dynasty information
'
cmdSQL.CommandText = "UPDATE (ZZZ_BIOG_MAIN INNER JOIN ZZ_SCRATCH_KINNET ON ZZZ_BIOG_MAIN.c_personid = ZZ_SCR
ATCH_KINNET.c_person_id) " + _
    "INNER JOIN ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 ON ZZ_SCRATCH_KINNET.c_kin_id = ZZZ_BIOG_MAIN_1.c_personid "
+ _
    "SET ZZ_SCRATCH_KINNET.c_dy = [ZZZ_BIOG_MAIN].[c_dy], " + _
    "ZZ_SCRATCH_KINNET.c_dynasty = [ZZZ_BIOG_MAIN].[c_dynasty], " + _
    "ZZ_SCRATCH_KINNET.c_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn], " + _
    "ZZ_SCRATCH_KINNET.c_kin_dy = [ZZZ_BIOG_MAIN_1].[c_dy], " + _
    "ZZ_SCRATCH_KINNET.c_kin_dynasty = [ZZZ_BIOG_MAIN_1].[c_dynasty], " + _
    "ZZ_SCRATCH_KINNET.c_kin_dynasty_chn = [ZZZ_BIOG_MAIN_1].[c_dynasty_chn]"
cmdSQL.Execute tRecDelete

cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_KIN INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KIN.c_kin_id = ZZZ_BIOG_MA
IN.c_personid " + _
    "SET ZZ_SCRATCH_KIN.c_kin_dy = [ZZZ_BIOG_MAIN].[c_dy], " + _
    "ZZ_SCRATCH_KIN.c_kin_dynasty = [ZZZ_BIOG_MAIN].[c_dynasty], " + _
    "ZZ_SCRATCH_KIN.c_kin_dynasty_chn = [ZZZ_BIOG_MAIN].[c_dynasty_chn]"
cmdSQL.Execute tRecDelete

' The ego-relative table needs to be cleaned up. The simplest approach is to take the shortest string
'
'
' calculate the xy_count
'
' use four SQL calls
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted
'
' get PersonID
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id ) " + _
    "SELECT DISTINCT ZZ_SCRATCH_KIN.c_person_id " + _
    "FROM ZZ_SCRATCH_KIN INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_KIN.c_person_id = ZZZ_BIOG_MAIN.c_personid"
cmdSQL.Execute tRecDeleted
'
' add KinID
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id ) " + _
    "SELECT DISTINCT ZZ_SCRATCH_KIN.c_kin_id " + _
    "FROM ZZ_SCRATCH_KIN LEFT JOIN ZZ_SCRATCH_PEOPLE ON ZZ_SCRATCH_KIN.c_kin_id = ZZ_SCRATCH_PEOPLE.c_person
id " + _
    "WHERE (((ZZ_SCRATCH_PEOPLE.c_person_id) Is Null))"
cmdSQL.Execute tRecDeleted
'
' get additional information from ZZZ_BIOG_MAIN
'

```

```

cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PEOPLE INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_PEOPLE.c_person_id = ZZ
Z_BIOG_MAIN.c_personid " +
"SET ZZ_SCRATCH_PEOPLE.c_name = [ZZZ_BIOG_MAIN].[c_name], ZZ_SCRATCH_PEOPLE.c_name_chn = [ZZZ_BIOG_MAIN].
[c_name_chn], " +
"ZZ_SCRATCH_PEOPLE.c_female = [ZZZ_BIOG_MAIN].[c_female], ZZ_SCRATCH_PEOPLE.c_index_year = [ZZZ_BIOG_
MAIN].[c_index_year], " +
"ZZ_SCRATCH_PEOPLE.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_addr_type_code], " +
"ZZ_SCRATCH_PEOPLE.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " +
"ZZ_SCRATCH_PEOPLE.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], ZZ_SCRATCH_PEOPLE.c
_dy = [ZZZ_BIOG_MAIN].[c_dy], " +
"ZZ_SCRATCH_PEOPLE.c_dynasty = [ZZZ_BIOG_MAIN].[c_dynasty], ZZ_SCRATCH_PEOPLE.c_dynasty_chn = [ZZZ_BI
OG_MAIN].[c_dynasty_chn], " +
"ZZ_SCRATCH_PEOPLE.c_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], ZZ_SCRATCH_PEOPLE.c_addr_name = [ZZ
Z_BIOG_MAIN].[c_index_addr_name], " +
"ZZ_SCRATCH_PEOPLE.c_addr_chn = [ZZZ_BIOG_MAIN].[c_index_addr_chn], ZZ_SCRATCH_PEOPLE.c_addr_type = [
ZZZ_BIOG_MAIN].[c_index_addr_type_code], " +
"ZZ_SCRATCH_PEOPLE.c_addr_desc = [ZZZ_BIOG_MAIN].[c_index_addr_type_desc], ZZ_SCRATCH_PEOPLE.c_addr_d
esc_chn = [ZZZ_BIOG_MAIN].[c_index_addr_type_chn], " +
"ZZ_SCRATCH_PEOPLE.x_coord = [ZZZ_BIOG_MAIN].[x_coord], ZZ_SCRATCH_PEOPLE.y_coord = [ZZZ_BIOG_MAIN].[
y_coord]"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from tmpXY"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " +
"SELECT ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord, Count(ZZ_SCRATCH_PEOPLE.x_coord
) AS CountOfx_coord, " +
"Count(ZZ_SCRATCH_PEOPLE.y_coord) AS CountOfy_coord " +
"FROM ZZ_SCRATCH_PEOPLE " +
"GROUP BY ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_KIN ON (tmpXY.y_coord = " +
"ZZ_SCRATCH_KIN.kin_y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_KIN.kin_x_coord) " +
"SET ZZ_SCRATCH_KIN.xy_count = [tmpXY].[CountOfx_coord];"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' record the XYCount in ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PEOPLE INNER JOIN tmpXY ON (ZZ_SCRATCH_PEOPLE.y_coord = tmpXY.y_coord
) " +
"AND (ZZ_SCRATCH_PEOPLE.x_coord = tmpXY.x_coord) " +
"SET ZZ_SCRATCH_PEOPLE.xy_count = [tmpXY].[CountOfx_coord]"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from tmpXY"
cmdSQL.Execute tRecDeleted
'
' calculate the distances for the ego-network
'
'DoCmd.RunSQL "ALTER TABLE ZZ_SCRATCH_KIN ADD COLUMN c_t_dist DOUBLE"

tSQLstr = "UPDATE ZZ_SCRATCH_KIN SET ZZ_SCRATCH_KIN.c_t_dist = " +
"Sqr((Sin(3.1415926536*(y_coord-kin_y_coord)/360))^2+Cos(3.1415926536*y_coord/180)*" +
"Cos(3.1415926536*kin_y_coord/180)*(Sin(3.1415926536*(x_coord-kin_x_coord)/360))^2) " +
"WHERE (((ZZ_SCRATCH_KIN.x_coord)>0) AND ((ZZ_SCRATCH_KIN.y_coord)>0) AND " +
"((ZZ_SCRATCH_KIN.kin_x_coord)>0) AND ((ZZ_SCRATCH_KIN.kin_y_coord)>0));"

cmdSQL.CommandText = tSQLstr
cmdSQL.Execute tRecDeleted

tSQLstr = "UPDATE ZZ_SCRATCH_KIN SET ZZ_SCRATCH_KIN.c_distance = " +
"25484*Atn(c_t_dist/(1+Sqr(1-c_t_dist*c_t_dist))) " +
"WHERE (((ZZ_SCRATCH_KIN.x_coord)>0) AND ((ZZ_SCRATCH_KIN.y_coord)>0) AND " +
"((ZZ_SCRATCH_KIN.kin_x_coord)>0) AND ((ZZ_SCRATCH_KIN.kin_y_coord)>0))"

cmdSQL.CommandText = tSQLstr
cmdSQL.Execute tRecDeleted

'DoCmd.RunSQL "ALTER TABLE ZZ_SCRATCH_KIN DROP COLUMN c_t_dist"
'
'GoTo Exit_CmdRun_Click
Exit_CmdRun_Click:

```

```

'
' reattach the tables
'
Set gRstPersonID = CurrentDb.OpenRecordset("ZZ_SCRATCH_KIN", dbOpenDynaset)
Set frmZZ_SCRATCH_KIN.Form.Recordset = gRstPersonID
Set frmZZ_SCRATCH_KINNET.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_KINNET", dbOpenDynaset)
Set frmZZ_SCRATCH_PEOPLE.Form.Recordset = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)

If gRstPersonID.RecordCount > 0 Then
    'cmdSQL.CommandText = tQueryStr
    'cmdSQL.Execute tRecDeleted

    Me.CmdGIS.Enabled = True
    Me.CmdGUESS.Enabled = True
    CmdStoreID.Enabled = True
    CmdUCINet.Enabled = True
    CmdUTF8Pajek.Enabled = True
    ChkIncludeID.Enabled = True
    CmdNeo4j.Enabled = True
Else
    Me.CmdGIS.Enabled = False
    Me.CmdGUESS.Enabled = False
    CmdStoreID.Enabled = False
    CmdUCINet.Enabled = False
    CmdUTF8Pajek.Enabled = False
    ChkIncludeID.Enabled = False
    CmdNeo4j.Enabled = False
End If
'
' close the tables
'

Set gRstPersonID = Nothing
'

frmZZ_SCRATCH_KIN.Form.OrderBy = "c_up,c_down,c_collateral,c_marriage"
frmZZ_SCRATCH_KIN.Form.OrderByOn = True

Exit Sub

Err_CmdRun_Click:
MsgBox Err.Description + tErrorStr
Resume Exit_CmdRun_Click

End Sub

Private Sub CmdPajek_Click()
On Error GoTo Err_CmdPajek_Click
'
' This program will dump the results of the search to a .net file
'
' for the moment I'll just describe the format of the .gdf file
'
' *Vertices NUM
' ID label "box" ic [color] bc [color]
' ID = str(c_person_id)
' label = c_name_chn
' color = red (1), orange (2), yellow (3), green (4), blue (5)
'
' *Edges
' node1 node2 1 1 "label"
' node1 = str(c_person_id) for node1
' node2 = str(c_node_id) for node2
' color = red (1), orange (2), yellow (3), green (4), blue (5)
' label = c_link_desc
'
' first see if there are any records to process
'
If frmZZ_SCRATCH_KIN.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
'

If frmZZ_SCRATCH_KINNET.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
'
' next get a file
'

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tMetricSum As Integer

```

```

Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstEdgeLookup As DAO.Recordset
' Dim tRstNodeList As DAO.Recordset
' Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tHitMax As Integer
Dim tColor(20) As String
Dim tFileSystem
' Dim tGDF
Dim tHitID(500) As Long, tHitCount As Integer, tFound As Boolean
Dim tUpStr As String, tDwnStr As String, tMarStr As String, tColStr As String
Dim tj As Integer, tSearchStr(9, 4, 2) As String, tUseList As Boolean
Dim tUpVal(9) As Integer, tDwnVal(9) As Integer, tMarVal(9) As Integer, tColVal(9) As Integer

```

```

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

```

```

tHitMax = 500

```

```

'Use a With...End With block to reference the FileDialog object.

```

```

With dlgSaveAs

```

```

    .InitialFileName = "kin.net"

```

```

    If .Show = -1 Then

```

```

        '

```

```

        tFileName = ""

```

```

        For Each tFN In .SelectedItems

```

```

            tFileName = tFN

```

```

            If Not tFileName = "" Then

```

```

                Exit For

```

```

            End If

```

```

        Next

```

```

        If tFileName = "" Then

```

```

            MsgBox "Bad file Name."

```

```

            GoTo Exit_CmdPajek_Click

```

```

        Else

```

```

            ' make sure the file name has a net extension

```

```

            If Len(tFileName) < 5 Then

```

```

                tFileName = tFileName + ".net"

```

```

            ElseIf Not (LCase(Right(tFileName, 4)) = ".net") Then

```

```

                tFileName = tFileName + ".net"

```

```

            End If

```

```

        End If

```

```

        '

```

```

        ' zap and open the scratch file

```

```

        '

```

```

        Dim cmdSQL As ADODB.Command

```

```

        Set cmdSQL = New ADODB.Command

```

```

        cmdSQL.ActiveConnection = CurrentProject.Connection

```

```

        cmdSQL.CommandType = adCmdText

```

```

        '

```

```

        cmdSQL.CommandText = "Delete from ZZ_SCRATCH_PAJEK where c_ID > -1"

```

```

        cmdSQL.Execute tRecDeleted

```

```

        '

```

```

        Set gRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)

```

```

        gRstNodeList.Index = "c_ID"

```

```

        '

```

```

        ' now process the file (second true removed to make ASCII)

```

```

        '

```

```

        Set tFileSystem = CreateObject("Scripting.FileSystemObject")

```

```

        Set gGdf = tFileSystem.CreateTextFile(tFileName, True)

```

```

        '

```

```

        ' define the colors for the nodes

```

```

        '

```

```

        tColor(1) = "White"

```

```

        tColor(2) = "Blue"

```

```

        tColor(3) = "Green"

```

```

        tColor(4) = "Yellow"

```

```

        tColor(5) = "Orange"

```

```

        For ti = 6 To 20

```

```

            tColor(ti) = "Red"

```

```

        Next

```

```

        '

```

```

        ' determine whether nodes come from a query (import list) or from the table

```

```

        '

```

```

        If Left(TxtName.Value, 1) = "[" Then

```

```

            tStr = "SELECT [ZZ_SCRATCH_KIN DISTINCT Query].*, ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id "

```

```

            tStr = tStr + "FROM [ZZ_SCRATCH_KIN DISTINCT Query] LEFT JOIN ZZ_SCRATCH_IMPORT_PEOPLE"

```

```

            tStr = tStr + " ON [ZZ_SCRATCH_KIN DISTINCT Query].c_kin_id = ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id"

```

```

            '

```

```

            Set tRstNode = CurrentDb.OpenRecordset(tStr, dbOpenDynaset)

```

```

            '

```

```

            ' to get an accurate record count

```



```

    '
    tRstNode.MoveLast
    tUseList = True
Else
    Set tRstNode = frmZZ_SCRATCH_KIN.Form.Recordset
    tUseList = False
End If
'
' process the two tables
'
Set gRstEdge = frmZZ_SCRATCH_KINNET.Form.Recordset
Set tRstEdgeLookup = gRstEdge.Clone

tC = Chr(44) ' the comma
'
' first the nodes: define the record structure
tStr = "*Vertices " + Trim(Str(tRstNode.RecordCount))
gGdf.WriteLine (tStr)
'
ti = 1
With tRstNode
    .MoveFirst
    Do While Not .EOF
        gGdf.Write (Trim(Str(ti)) + " ")
        ' name = the ID of the person
        If IsNull(!c_kin_name) Then
            gGdf.Write (" box ")
        Else
            gGdf.Write (Chr(34))
            gGdf.Write (!c_kin_name)
            gGdf.Write (Chr(34))
            gGdf.Write (" box ")
        End If
        ' label
        If tUseList Then
            '
            ' only people on the initial import list have a non-null c_person_id
            '
            If IsNull(!c_person_id) Then
                tMetricSum = 2
            Else
                tMetricSum = 1
            End If
        Else
            tMetricSum = !c_marriage + !c_up + !c_down + !c_collateral
            If tMetricSum = 0 Then
                tMetricSum = 1
            End If
            If tMetricSum > 20 Then
                tMetricSum = 20
            End If
        End If
        tStr = " ic " + tColor(tMetricSum)
        tStr = tStr + " bc " + tColor(tMetricSum)
        ' color = white (1), blue (2), green (3), yellow (4), orange (5)
        gGdf.WriteLine (tStr)
        '
        ' add the node to the list
        '
        gRstNodeList.AddNew
        gRstNodeList!c_v_num = Str(ti)
        gRstNodeList!c_ID = !c_kin_id
        gRstNodeList.Update
        '
        .MoveNext
        ti = ti + 1
    Loop
End With
'
' now the edges: define the record structure
tStr = "*Edges"
gGdf.WriteLine (tStr)

With gRstEdge
    '
    ' first process all the immediate relations
    '
    .MoveFirst
    Do While Not .EOF
        If !c_up = 1 And !c_down = 0 And !c_marriage = 0 And !c_collateral = 0 Then

```

```

        Call write_edge
    ElseIf !c_up = 0 And !c_down = 1 And !c_marriage = 0 And !c_collateral = 0 Then
        Call write_edge
    ElseIf !c_up = 0 And !c_down = 0 And !c_marriage = 1 And !c_collateral = 0 Then
        Call write_edge
    ElseIf !c_up = 0 And !c_down = 0 And !c_marriage = 0 And !c_collateral = 1 Then
        Call write_edge
    End If
    .MoveNext
Loop
'
' next look for all second degree connections
'
' the logic here is that there are four separate searched needed to confirm that there are no
' intermediate nodes between two nodes with a distance of 2, since the relevant nodes can
' appear at either end of the edge. There are nine possible configuration for a sum of 2
'
tUpVal(1) = 2
tDwnVal(1) = 0
tMarVal(1) = 0
tColVal(1) = 0
'
tUpVal(2) = 0
tDwnVal(2) = 2
tMarVal(2) = 0
tColVal(2) = 0
'
tUpVal(3) = 0
tDwnVal(3) = 0
tMarVal(3) = 2
tColVal(3) = 0
'
tUpVal(4) = 0
tDwnVal(4) = 0
tMarVal(4) = 0
tColVal(4) = 2
'
tUpVal(5) = 1
tDwnVal(5) = 0
tMarVal(5) = 1
tColVal(5) = 0
'
tUpVal(6) = 1
tDwnVal(6) = 0
tMarVal(6) = 0
tColVal(6) = 1
'
tUpVal(7) = 0
tDwnVal(7) = 0
tMarVal(7) = 1
tColVal(7) = 1
'
tUpVal(8) = 0
tDwnVal(8) = 1
tMarVal(8) = 1
tColVal(8) = 0
'
tUpVal(9) = 0
tDwnVal(9) = 1
tMarVal(9) = 0
tColVal(9) = 1
'
' the basic search conditions:
'
tUpStr = "c_up = 1 AND c_down = 0 AND c_marriage = 0 And c_collateral = 0"
tDwnStr = "c_up = 0 AND c_down = 1 AND c_marriage = 0 And c_collateral = 0"
tMarStr = "c_up = 0 AND c_down = 0 AND c_marriage = 1 And c_collateral = 0"
tColStr = "c_up = 0 AND c_down = 0 AND c_marriage = 0 And c_collateral = 1"

.MoveFirst
Do While Not .EOF
'
' the four search strings for each iteration
'
tSearchStr(1, 1, 1) = tUpStr & " AND c_person_id = " & Str(gRstEdge!c_person_id)
tSearchStr(1, 2, 1) = tDwnStr & " AND c_kin_id = " & Str(gRstEdge!c_person_id)
' the second two strings need to be appended at processing time
tSearchStr(1, 3, 1) = tUpStr & " AND c_kin_id = " & Str(gRstEdge!c_kin_id)
tSearchStr(1, 4, 1) = tDwnStr & " AND c_person_id = " & Str(gRstEdge!c_kin_id)
'

```



```
ateral = tColVal(tj) Then
```

```

tFound = False
'
tRstEdgeLookup.FindFirst tSearchStr(tj, 1, 1)
If tRstEdgeLookup.NoMatch Then
'
' look in the other direction
'
tRstEdgeLookup.FindFirst tSearchStr(tj, 2, 1)
If Not tRstEdgeLookup.NoMatch Then
'
tHitCount = 1
tHitID(1) = tRstEdgeLookup!c_person_id
'
' get the rest
'
Do While Not tRstEdgeLookup.NoMatch
tRstEdgeLookup.FindNext tSearchStr(tj, 2, 1)
If Not tRstEdgeLookup.NoMatch Then
tHitCount = tHitCount + 1
If tHitCount > tHitMax Then
MsgBox "At " & Str(tRstEdgeLookup!c_person_id) & " HitCount = " & _
+ Str(tHitCount)
End If
tHitID(tHitCount) = tRstEdgeLookup!c_person_id
End If
Loop
'
'
' now see if the person is in fact the same link
'
ti = 1
Do While ti <= tHitCount And Not tFound
tStr = "c_person_id = " & Str(tHitID(ti)) & " and " & tSearchStr(tj, 3, 1)

tRstEdgeLookup.FindFirst tStr
If tRstEdgeLookup.NoMatch Then
'
' look for the other edge
'
tStr = "c_kin_id = " & Str(tHitID(ti)) & " and " & tSearchStr(tj, 4, 1)

tRstEdgeLookup.FindFirst tStr
If Not tRstEdgeLookup.NoMatch Then
tFound = True
End If
Else
tFound = True
End If
ti = ti + 1
Loop
End If
Else
'
tHitCount = 1
tHitID(1) = tRstEdgeLookup!c_kin_id
'
' get the rest
'
Do While Not tRstEdgeLookup.NoMatch
tRstEdgeLookup.FindNext tSearchStr(tj, 1, 1)
If Not tRstEdgeLookup.NoMatch Then
tHitCount = tHitCount + 1
If tHitCount > tHitMax Then
MsgBox "(2) At " & Str(tRstEdgeLookup!c_person_id) & " HitCount = " & _
+ Str(tHitCount)
End If
tHitID(tHitCount) = tRstEdgeLookup!c_kin_id
End If
Loop
'
' now see if the person is in fact the same link
'
ti = 1
Do While ti <= tHitCount And Not tFound
tStr = "c_person_id = " & Str(tHitID(ti)) & " and " & tSearchStr(tj, 3, 1)

tRstEdgeLookup.FindFirst tStr
If tRstEdgeLookup.NoMatch Then

```

```

        ' look for the other edge
        ,
        tStr = "c_kin_id = " & Str(tHitID(ti)) & " and " & tSearchStr(tj, 4, 1)
        ,
        tRstEdgeLookup.FindFirst tStr
        If Not tRstEdgeLookup.NoMatch Then
            ' one could tentatively write an edge between the found node and the
            ' current kin node, but I need to wait for instruction
            ,
            tFound = True
        End If
    Else
        tFound = True
    End If
    ti = ti + 1
Loop
End If
If tj > 4 And Not tFound Then
    tRstEdgeLookup.FindFirst tSearchStr(tj, 1, 2)
    If tRstEdgeLookup.NoMatch Then
        ,
        ' look in the other direction
        ,
        tRstEdgeLookup.FindFirst tSearchStr(tj, 2, 2)
        If Not tRstEdgeLookup.NoMatch Then
            ,
            tHitCount = 1
            tHitID(1) = tRstEdgeLookup!c_person_id
            ,
            ' get the rest
            ,
            Do While Not tRstEdgeLookup.NoMatch
                tRstEdgeLookup.FindNext tSearchStr(tj, 2, 2)
                If Not tRstEdgeLookup.NoMatch Then
                    tHitCount = tHitCount + 1
                    If tHitCount > tHitMax Then
                        MsgBox "(3) At " & Str(tRstEdgeLookup!c_person_id) & " HitCou
                                + Str(tHitCount)
                    End If
                    tHitID(tHitCount) = tRstEdgeLookup!c_person_id
                End If
            Loop
            ,
            ' now see if the person is in fact the same link
            ,
            ti = 1
            Do While ti <= tHitCount And Not tFound
                tStr = "c_person_id = " & Str(tHitID(ti)) & " and " & tSearchStr(tj,
3, 2)
                ,
                tRstEdgeLookup.FindFirst tStr
                If tRstEdgeLookup.NoMatch Then
                    ' look for the other edge
                    ,
                    tStr = "c_kin_id = " & Str(tHitID(ti)) & " and " & tSearchStr(tj,
4, 2)
                    ,
                    tRstEdgeLookup.FindFirst tStr
                    If Not tRstEdgeLookup.NoMatch Then
                        ' one could tentatively write an edge between the found node
                        ,
                        ' current kin node, but I need to wait for instruction
                        ,
                        tFound = True
                    End If
                Else
                    tFound = True
                End If
                ti = ti + 1
            Loop
        End If
    Else
        ,
        tHitCount = 1
        tHitID(1) = tRstEdgeLookup!c_kin_id
        ,
        ' get the rest
        ,
        Do While Not tRstEdgeLookup.NoMatch

```

```

tRstEdgeLookup.FindNext tSearchStr(tj, 1, 2)
If Not tRstEdgeLookup.NoMatch Then
    tHitCount = tHitCount + 1
    If tHitCount > tHitMax Then
        MsgBox "(4) At " + Str(tRstEdgeLookup!c_person_id) + " HitCount = "
            + Str(tHitCount)
    End If
    tHitID(tHitCount) = tRstEdgeLookup!c_kin_id
End If
Loop
'
' now see if the person is in fact the same link
'
ti = 1
Do While ti <= tHitCount And Not tFound
    tStr = "c_person_id = " & Str(tHitID(ti)) & " and " & tSearchStr(tj, 3, 2)
    '
    tRstEdgeLookup.FindFirst tStr
    If tRstEdgeLookup.NoMatch Then
        ' look for the other edge
        '
        tStr = "c_kin_id = " & Str(tHitID(ti)) & " and " & tSearchStr(tj, 4,
            '
            tRstEdgeLookup.FindFirst tStr
            If Not tRstEdgeLookup.NoMatch Then
                '
                tFound = True
            End If
        Else
            tFound = True
        End If
        ti = ti + 1
    Loop
End If
'
If tFound Then
    .Edit
    !c_processed = True
    .Update
Else
    Call write_edge
End If
End If
Next
'
' this is where I stopped: must deal with multiple hits
'
.MoveNext
Loop
'
' next get all the rest (although I could do the 3rd degree, but that would require a more elega
'
' algorithm
'
.MoveNextFirst
Do While Not .EOF
    If Not !c_processed Then
        Call write_edge
    End If
    .MoveNext
Loop
End With
'
gGdf.Close
gRstNodeList.Close
'
'Set tRstNode = Nothing
'Set gRstEdge = Nothing
Set gGdf = Nothing
Set tFileSystem = Nothing
'Set gRstNodeList = Nothing
Else
    'The user pressed Cancel.
End If
End With

```

```
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing
```

```
Exit_CmdPajek_Click:
Exit Sub
```

```
Err_CmdPajek_Click:
MsgBox Err.Description
Resume Exit_CmdPajek_Click
```

```
End Sub
```

```
Private Sub write_edge()
Dim tStr As String, tMetricSum As Integer, tColor(20) As String
'
' define the colors for the nodes
'
tColor(1) = "White"
tColor(2) = "Blue"
tColor(3) = "Green"
tColor(4) = "Yellow"
tColor(5) = "Orange"
For ti = 6 To 20
    tColor(ti) = "Red"
Next
'
' find the vertex number of the first node
'
gRstNodeList.Seek "=", Str(gRstEdge!c_person_id)
If Not gRstNodeList.NoMatch Then
    tStr = gRstNodeList!c_v_num + " "
    ' find the vertex number of the second node
    gRstNodeList.Seek "=", Str(gRstEdge!c_kin_id)
    If Not gRstNodeList.NoMatch Then
        tStr = tStr + gRstNodeList!c_v_num + " 1 1 "
        gGdf.Write (tStr)
        gGdf.Write (Chr(34))
        gGdf.Write (gRstEdge!c_kin_rel)
        gGdf.Write (Chr(34))
        tMetricSum = gRstEdge!c_marriage + gRstEdge!c_up + gRstEdge!c_down + gRstEdge!c_collateral
        If tMetricSum = 0 Then
            tMetricSum = 1
        End If
        If tMetricSum > 20 Then
            tMetricSum = 20
        End If
        gGdf.WriteLine (" c " + tColor(tMetricSum))
        ' color = white (1), blue (2), green (3), yellow (4), orange (5)
        ' update the fact that the record has been processed
        gRstEdge.Edit
        gRstEdge!c_processed = True
        gRstEdge.Update
    End If
End If
End Sub
```

```
Private Sub write_edge_utf8()
Dim tStr As String, tMetricSum As Integer, tColor(20) As String
'
' define the colors for the nodes
'
tColor(1) = "White"
tColor(2) = "Blue"
tColor(3) = "Green"
tColor(4) = "Yellow"
tColor(5) = "Orange"
For ti = 6 To 20
    tColor(ti) = "Red"
Next
'
' find the vertex number of the first node
'
gRstNodeList.Seek "=", Str(gRstEdge!c_person_id)
If Not gRstNodeList.NoMatch Then
```

```

'
tStr = gRstNodeList!c_v_num + " "
'
' find the vertex number of the second node
'
gRstNodeList.Seek "=", Str(gRstEdge!c_kin_id)
If Not gRstNodeList.NoMatch Then
    tStr = tStr + gRstNodeList!c_v_num + " 1 1 "
    '
    gStream.WriteText tStr
    gStream.WriteText Chr(34)
    gStream.WriteText gRstEdge!c_kin_rel
    gStream.WriteText Chr(34)
    '
    tMetricSum = gRstEdge!c_marriage + gRstEdge!c_up + gRstEdge!c_down + gRstEdge!c_collateral
    If tMetricSum = 0 Then
        tMetricSum = 1
    End If
    If tMetricSum > 20 Then
        tMetricSum = 20
    End If
    gStream.WriteText " c " + tColor(tMetricSum), adWriteLine
    ' color = white (1), blue (2), green (3), yellow (4), orange (5)
    '
    ' update the fact that the record has been processed
    '
    gRstEdge.Edit
    gRstEdge!c_processed = True
    gRstEdge.Update
End If
End If
End Sub

Private Sub calculate_xy_count()
    Dim tX As Double, tY As Double, tXY As Integer, tBM As Variant, tWrite As Integer
    Dim tID As Long
    '
    ' the strategy is to first throw a bookmark at the first new value
    ' then count the number, then go back to the bookmark and update each record
    '
    ' in order to allow the use of the index, gRstPersonID must be reopened as a table
    '
    Set gRstPersonID = CurrentDb.OpenRecordset("ZZ_SCRATCH_KIN", dbOpenTable)
    With gRstPersonID
        .Index = "xy"
        .MoveFirst

        tX = -1#
        tY = -1#
        tXY = 0
        tWrite = 0
        tBM = .Bookmark

        Do While Not .EOF
            If tX <> !kin_x_coord Or tY <> !kin_y_coord Then
                If tWrite = 1 Then
                    ' go back to the first record with the value
                    .Bookmark = tBM
                    Do While tX = !kin_x_coord And tY = !kin_y_coord
                        .Edit
                        !xy_count = tXY
                        .Update
                        .MoveNext
                    Loop
                Else
                    tWrite = 1
                End If
                ' reset
                tXY = 0
                tBM = .Bookmark
                tX = !kin_x_coord
                tY = !kin_y_coord
                tID = -2
            End If
            ' increment the count and move to the next
            If tID <> !c_kin_id Then
                tXY = tXY + 1
                tID = !c_kin_id
            End If
            .MoveNext
        
```



```

        ti = ti + 2
    Else
        tStrRelTotal = tStrRelTotal + tStrTest
    End If
Case "K"
    If Mid(tStrRel, ti + 1, 1) = "+" Or Mid(tStrRel, ti + 1, 1) = "-" Then
        If Mid(tStrRel, ti + 2, 1) = "n" Or IsNumeric(Mid(tStrRel, ti + 2, 1)) Then
            tStrRelTotal = tStrRelTotal + Mid(tStrRel, ti, 3)
            ti = ti + 2
        Else
            tStrRelTotal = tStrRelTotal + Mid(tStrRel, ti, 2)
            ti = ti + 1
        End If
    Else
        tStrRelTotal = tStrRelTotal + tStrTest
    End If
Case "G"
    If Mid(tStrRel, ti + 1, 1) = "+" Or Mid(tStrRel, ti + 1, 1) = "-" Then
        If Mid(tStrRel, ti + 2, 1) = "n" Or IsNumeric(Mid(tStrRel, ti + 2, 1)) Then
            If IsNumeric(Mid(tStrRel, ti + 3, 1)) Then
                tStrRelTotal = tStrRelTotal + Mid(tStrRel, ti, 4)
                ti = ti + 3
            Else
                tStrRelTotal = tStrRelTotal + Mid(tStrRel, ti, 3)
                ti = ti + 2
            End If
        End If
    End If
Case Else
    ti = ti
End Select
End If
ti = ti + 1
Loop
tStrRelTotalFinal = tStrRelTotal
'MsgBox "tStrRel = " + tStrRel + ", tStrRelTotal = " + tStrRelTotal + _
'      ", and tStrRelTotalFinal = " + tStrRelTotalFinal
'
'    update the total kinrel string
'
.Edit
!c_kinrel_total = tStrRelTotalFinal
.Update
.MoveNext
Loop
End If
End With
gRst.Close
End Sub
Private Sub CmdSelectPerson_Click()
On Error GoTo Err_CmdSelectPerson_Click

Dim stDocName As String, tRstPeople As DAO.Recordset
Dim stLinkCriteria As String, cmdSQL As ADODB.Command
Dim strPERSON_ID As String, tStrQuery As String, tQuit As Boolean

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

,
tQuit = False
If Not tQuit Then
    TxtPersonID.Visible = True
    TxtPersonID.SetFocus
    strPERSON_ID = TxtPersonID.Text

    stDocName = "frmSelectPerson"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strPERSON_ID

    If CurrentProject.AllForms("frmSelectPerson").IsLoaded Then
        Dim lngPERSON_ID As Long
        Dim strPERSON_NM As String
        Dim strPERSON_NM_CHN As String

        Forms!frmSelectPerson!frmPersonSearch.Form!c_personid.SetFocus
        lngPERSON_ID = Forms!frmSelectPerson!frmPersonSearch.Form!c_personid.Value
        TxtPersonID.Value = lngPERSON_ID
    End If
End If

```

```
Forms!frmSelectPerson!frmPersonSearch.Form!c_name_chn.SetFocus
strPERSON_NM_CHN = Forms!frmSelectPerson!frmPersonSearch.Form!c_name_chn.Value
TxtNameChn.Value = strPERSON_NM_CHN
```

```
Forms!frmSelectPerson!frmPersonSearch.Form!c_name.SetFocus
strPERSON_NM = Forms!frmSelectPerson!frmPersonSearch.Form!c_name.Value
TxtName.Value = strPERSON_NM
```

```
DoCmd.Close acForm, stDocName
```

```
' now load the one ID into the ImportPeople table
```

```
' Clear the people table now that we are ready to go
```

```
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
cmdSQL.Execute tRecDeleted
```

```
' add the name
```

```
tStrQuery = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_
female, c_addr_id, c_addr_name, c_addr_chn, " +
"c_addr_type, c_addr_desc, c_addr_desc_chn, x_coord, y_coord ) " +
"SELECT ZZZ_BIOG_MAIN.c_personid, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MAIN.c_
_index_year, " +
"ZZZ_BIOG_MAIN.c_female, ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_
_MAIN.c_index_addr_chn, " +
"ZZZ_BIOG_MAIN.c_index_addr_type_code, ZZZ_BIOG_MAIN.c_index_addr_type_desc, ZZZ_BIOG_MAIN.c_inde
x_addr_type_chn, ZZZ_BIOG_MAIN.x_coord, " +
"ZZZ_BIOG_MAIN.y_coord " +
"FROM ZZZ_BIOG_MAIN WHERE ((ZZZ_BIOG_MAIN.c_personid)=" + Str(lngPERSON_ID) + ")")"
```

```
cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecDeleted
```

```
If tRecDeleted = 0 Then
```

```
MsgBox "The ID " + Trim(Str(lngPERSON_ID)) + " is not valid"
```

```
CmdRun.Enabled = False
```

```
Else
```

```
' now enable the Run button
```

```
CmdRun.Enabled = True
```

```
End If
```

```
Set cmdSQL = Nothing
```

```
End If
```

```
End If
```

```
gCurRecallSource = ""
```

```
CmdSelectPerson.SetFocus
```

```
TxtPersonID.Visible = False
```

```
Exit_CmdSelectPerson_Click:
```

```
Exit Sub
```

```
Err_CmdSelectPerson_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdSelectPerson_Click
```

```
End Sub
```

```
Private Sub CmdUCINet_Click()
```

```
On Error GoTo Err_CmdUCINet_Click
```

```
' This program will dump the results of the search to a .vna file
```

```
' for the moment I'll just describe the format of the .vna file
```

```
' *node data
```

```
' ID index_year sex x_coord y_coord nodedist
```

```
' ID = str(c_person_id)
```

```
' indexyear = c_index_year INT
```

```
' nodedist = c_node_dist INT
```

```
' sex = c_female > (F,M)
```

```
' *node properties
```

```

' ID color shape size shortlabel active
'   color = red (1), orange (2), yellow (3), green (4), blue (5)
'   shortlabel = c_name
'   shape = 2
'   active = TRUE
'
' *tie data
' from to edgetype nodedist
'   from = str(c_person_id)
'   to = str(c_node_id)
'   edgetype= c_link_type (K,N)
'
' *tie properties
' from to color size active
'   from = str(c_person_id)
'   to = str(c_node_id)
'   color = red (255), orange (26367), yellow (65535), green (32768), blue (16711680)
'   size = 1-5 (the weight)
'
' the central question is whether to do distance optimizations
'
' first see if there are any records to process
'

```

```

If frmZZ_SCRATCH_KINNET.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUCINet_Click
End If
'

```

```

If frmZZ_SCRATCH_KIN.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUCINet_Click
End If
'

```

```

' next get a file
'

```

```

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstAssocType As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tSearchStr As String
Dim tColor(20) As String, tQuote As String
Dim tFileSystem, tVNA

```

```

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

```

```

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "kin_network.vna"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdUCINet_Click
        Else
            ' make sure the file name has a vna extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".vna"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".vna") Then
                tFileName = tFileName + ".vna"
            End If
        End If
        '
        ' now process the file (second true removed to make ASCII)
        '
        Set tFileSystem = CreateObject("Scripting.FileSystemObject")
        Set tVNA = tFileSystem.CreateTextFile(tFileName, True)

        ' define the colors for the nodes
        '
        tColor(1) = "0 " ' black
        tColor(2) = "16711680 " ' blue
    End With

```

```

tColor(3) = "32768 "      ' green
tColor(4) = "65535 "      ' yellow
tColor(5) = "26367 "      ' orange
For ti = 6 To 20
    tColor(ti) = "255 "    ' red
Next
'
' process the two tables
'
Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SCRATCH_KINNET", dbOpenDynaset)
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_KIN", dbOpenDynaset)
tQuote = Chr(34) ' the quotation mark
'
' first the nodes: define the node data structure
tVNA.WriteLine ("*node data")
tVNA.WriteLine ("ID index_year dy_code dynasty sex x_coord y_coord kindist")
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' name = the ID of the person
        If IsNull(!c_kin_id) Then
            tStr = "[?] "
        Else
            tStr = Trim(Str(!c_kin_id)) + " "
        End If
        '
        ' indexyear = c_index_year INT
        If IsNull(!c_kin_index_year) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!c_kin_index_year)) + " "
        End If
        '
        ' dy = c_kin_dy INT
        If IsNull(!c_kin_dynasty) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!c_kin_dy)) + " "
        End If
        '
        ' dynasty = c_kin_dynasty
        If IsNull(!c_kin_dynasty) Then
            tStr = tStr + tQuote + "Unknown" + tQuote + " "
        Else
            tStr = tStr + tQuote + Trim(!c_kin_dynasty) + tQuote + " "
        End If
        '
        ' sex = c_female > (F,M)
        If !c_kin_female = -1 Then
            tStr = tStr + tQuote + "F" + tQuote + " "
        Else
            tStr = tStr + tQuote + "M" + tQuote + " "
        End If
        '
        ' x_coord
        If IsNull(!kin_x_coord) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!kin_x_coord)) + " "
        End If
        '
        ' y_coord
        If IsNull(!kin_y_coord) Then
            tStr = tStr + "0 "
        Else
            tStr = tStr + Trim(Str(!kin_y_coord)) + " "
        End If
        '
        ' node distance
        tStr = tStr + Trim(Str(!c_up + !c_down + !c_collateral + !c_marriage))
        '
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the node properties
'
' note: the ACTIVE parameter has been removed (MAF 2018/07/22)

```

```

'
tVNA.WriteLine ("*node properties")
tVNA.WriteLine ("ID color shape size shortlabel")
'
With tRstNode
  .MoveFirst
  Do While Not .EOF
    ' ID = the ID of the person
    If IsNull(!c_kin_id) Then
      tStr = "[?] "
    Else
      tStr = Trim(Str(!c_kin_id)) + " "
    End If
    ' color = black (1), blue (2), green (3), yellow (4), orange (5)
    tStr = tStr + tColor(!c_up + !c_down + !c_collateral + !c_marriage + 1)
    ' shape = 2? / size = 1?
    tStr = tStr + "2 1 "
    ' shortlabel (+Active = "TRUE" removed)
    If IsNull(!c_kin_name) Then
      tStr = tStr + "[Missing]"
    Else
      tStr = tStr + tQuote + !c_kin_name + tQuote
    End If
    tVNA.WriteLine (tStr)
    .MoveNext
  Loop
End With
'
' now the edges: define the record structure
'
tStr = "from to " + tQuote + "EdgeWeight" + tQuote + " " + tQuote + "edgetype"
tStr = tStr + tQuote + " " + tQuote + "edgelist" + tQuote
tVNA.WriteLine ("*tie data")
tVNA.WriteLine (tStr)
'
' For the moment, I am not combining parallel edges
'
With tRstEdge
  .MoveFirst
  Do While Not .EOF
    ' From = str(c_person_id) for node1
    tStr = Trim(Str(!c_person_id)) + " "
    ' to = str(c_node_id) for node2
    tStr = tStr + Trim(Str(!c_kin_id)) + " 1 "
    '
    If IsNull(!c_kin_rel) Then
      tStr = tStr + "[?] "
    Else
      tStr = tStr + tQuote + !c_kin_rel + tQuote + " "
    End If
    ' edgelist
    tStr = tStr + Trim(Str(!c_upstep + !c_dwnstep + !c_marstep + !c_colstep))
    tVNA.WriteLine (tStr)
    .MoveNext
  Loop
End With
'
' now the edges properties
'
tVNA.WriteLine ("*tie properties")
tVNA.WriteLine ("from to color size active")

'With tRstEdge
'  .MoveFirst
'  Do While Not .EOF
'    ' from = str(c_person_id) for node1
'    tStr = Trim(Str(!c_person_id)) + " "
'    ' to = str(c_node_id) for node2
'    tStr = tStr + Trim(Str(!c_node_id)) + " 1 "
'

```

```

        '    color = black (1), blue (2), green (3), yellow (4), orange (5)
        'tStr = tStr + tColor(!c_edge_dist)
        '
        '    size = 1? active = TRUE
        'tStr = tStr + "1 TRUE"
        '
        'tVNA.WriteLine (tStr)
        '.MoveNext
    'Loop
'End With
'
tVNA.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tVNA = Nothing
Set tFileSystem = Nothing
Set tRstAssocType = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

```

```

Exit_CmdUCINet_Click:
Exit Sub

```

```

Err_CmdUCINet_Click:
MsgBox Err.Description
Resume Exit_CmdUCINet_Click

```

```
End Sub
```

```

Private Sub CmdUTF8Pajek_Click()
On Error GoTo Err_CmdUTF8Pajek_Click

```

```

'
'    This program will dump the results of the search to a .net file
'
'    for the moment I'll just describe the format of the .gdf file
'
'    *Vertices NUM
'    ID label "box" ic [color] bc [color]
'        ID = str(c_person_id)
'        label = c_name_chn
'        color = red (1), orange (2), yellow (3), green (4), blue (5)
'
'    *Edges
'    node1 node2 1 1 "label"
'        node1 = str(c_person_id) for node1
'        node2 = str(c_node_id) for node2
'        color = red (1), orange (2), yellow (3), green (4), blue (5)
'        label = c_link_desc
'
'
'    first see if there are any records to process
'
If frmZZ_SCRATCH_KIN.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUTF8Pajek_Click
End If
'
If frmZZ_SCRATCH_KINNET.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUTF8Pajek_Click
End If
'
'    next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant, tMetricSum As Integer
Dim tRstNodeList As DAO.Recordset
' Dim tRstNodeList As DAO.Recordset
' Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tQueryStr As String
Dim tColor(20) As String
Dim tFileSystem

```

```
' Dim tGDF
```

```
Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)
```

```
' to write to a UTF-8 file, use the ADO stream object
```

```
Set gStream = New ADODB.Stream
```

```
If CodeFrame.Value = 1 Then
```

```
    gStream.Charset = "utf-8"
```

```
    tCodeStr = "UTF8.net"
```

```
ElseIf CodeFrame.Value = 2 Then
```

```
    gStream.Charset = "big5"
```

```
    tCodeStr = "BIG5.net"
```

```
ElseIf CodeFrame.Value = 3 Then
```

```
    gStream.Charset = "gb2312"
```

```
    tCodeStr = "GB2312.net"
```

```
Else
```

```
    gStream.Charset = "ascii"
```

```
    tCodeStr = "ascii.net"
```

```
End If
```

```
gStream.Mode = adModeReadWrite
```

```
gStream.Type = adTypeText
```

```
gStream.Open
```

```
'Use a With...End With block to reference the FileDialog object.
```

```
With dlgSaveAs
```

```
    .InitialFileName = "kin_" + tCodeStr
```

```
    If .Show = -1 Then
```

```
        '
```

```
        tFileName = ""
```

```
        For Each tFN In .SelectedItems
```

```
            tFileName = tFN
```

```
            If Not tFileName = "" Then
```

```
                Exit For
```

```
            End If
```

```
        Next
```

```
        If tFileName = "" Then
```

```
            MsgBox "Bad file Name."
```

```
            GoTo Exit_CmdUTF8Pajek_Click
```

```
        Else
```

```
            ' make sure the file name has a net extension
```

```
            If Len(tFileName) < 5 Then
```

```
                tFileName = tFileName + ".net"
```

```
            ElseIf Not (LCase(Right(tFileName, 4)) = ".net") Then
```

```
                tFileName = tFileName + ".net"
```

```
            End If
```

```
        End If
```

```
        '
```

```
        ' zap and open the scratch file
```

```
        '
```

```
        Dim cmdSQL As ADODB.Command
```

```
        Set cmdSQL = New ADODB.Command
```

```
        cmdSQL.ActiveConnection = CurrentProject.Connection
```

```
        cmdSQL.CommandType = adCmdText
```

```
        '
```

```
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
```

```
        cmdSQL.Execute tRecDeleted
```

```
        '
```

```
        Set gRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
```

```
        gRstNodeList.Index = "c_ID"
```

```
        '
```

```
        ' define the colors for the nodes
```

```
        '
```

```
        tColor(1) = "Black"
```

```
        tColor(2) = "Blue"
```

```
        tColor(3) = "Green"
```

```
        tColor(4) = "Yellow"
```

```
        tColor(5) = "Orange"
```

```
        For ti = 6 To 20
```

```
            tColor(ti) = "Red"
```

```
        Next
```

```
        '
```

```
        ' prepare the temp table for the edge data
```

```
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_KINNET_EDGE"
```

```
        cmdSQL.Execute tRecDeleted
```

```
        '
```

```
        ' copy the data
```



```

'
tQueryStr = "INSERT INTO ZZ_SCRATCH_KINNET_EDGE SELECT ZZ_SCRATCH_KINNET.* FROM ZZ_SCRATCH_KINNET"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now delete the unneeded edges (i.e. if we have e->F and F->FF, we don't need e->FF)
'
tQueryStr = "UPDATE (ZZ_SCRATCH_KINNET_EDGE INNER JOIN ZZ_SCRATCH_KINNET_EDGE AS " + _
"ZZ_SCRATCH_KINNET_EDGE_1 ON ZZ_SCRATCH_KINNET_EDGE.c_person_id = " + _
"ZZ_SCRATCH_KINNET_EDGE_1.c_person_id) INNER JOIN ZZ_SCRATCH_KINNET_EDGE AS " + _
"ZZ_SCRATCH_KINNET_EDGE_2 ON (ZZ_SCRATCH_KINNET_EDGE_1.c_kin_id = " + _
"ZZ_SCRATCH_KINNET_EDGE_2.c_person_id) AND (ZZ_SCRATCH_KINNET_EDGE.c_kin_id = " + _
"ZZ_SCRATCH_KINNET_EDGE_2.c_kin_id) SET ZZ_SCRATCH_KINNET_EDGE.c_delete = 1 " + _
"WHERE (((ZZ_SCRATCH_KINNET_EDGE.c_upstep)=[ZZ_SCRATCH_KINNET_EDGE_1].[c_upstep]+ " + _
"[ZZ_SCRATCH_KINNET_EDGE_2].[c_upstep]) AND ((ZZ_SCRATCH_KINNET_EDGE.c_dwnstep)=" + _
"[ZZ_SCRATCH_KINNET_EDGE_1].[c_dwnstep]+[ZZ_SCRATCH_KINNET_EDGE_2].[c_dwnstep]) AND " + _
"((ZZ_SCRATCH_KINNET_EDGE.c_marstep)=[ZZ_SCRATCH_KINNET_EDGE_1].[c_marstep]+ " + _
"[ZZ_SCRATCH_KINNET_EDGE_2].[c_marstep]) AND ((ZZ_SCRATCH_KINNET_EDGE.c_colstep)=" + _
"[ZZ_SCRATCH_KINNET_EDGE_1].[c_colstep]+[ZZ_SCRATCH_KINNET_EDGE_2].[c_colstep]))"
'
'MsgBox "About to prune"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_KINNET_EDGE WHERE c_delete = 1"
cmdSQL.Execute tRecDeleted
'MsgBox "Pruning complete"
'
' now set the count to 1 ;

cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_KINNET_EDGE SET ZZ_SCRATCH_KINNET_EDGE.c_kin_rel_count = 1"
cmdSQL.Execute tRecDeleted
'
' now get the nodes
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
cmdSQL.Execute tRecDeleted
'
' fill the node list (Imported lists processed differently from single-person)
'
If Left(TxtName.Value, 1) = "[" Then
    If CodeFrame.Value = 4 Then
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_KIN.c_kin_id, ZZ_SCRATCH_KIN.c_kin_name, " + _
        "1 as c_distance, " + _
        "str(ZZ_SCRATCH_KIN.c_kin_id) AS c_v_num, TRUE as c_delete FROM ZZ_SCRATCH_KIN"
    Else
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_KIN.c_kin_id, ZZ_SCRATCH_KIN.c_kin_chn, " + _
        "1 as c_distance, " + _
        "str(ZZ_SCRATCH_KIN.c_kin_id) AS c_v_num, TRUE as c_delete FROM ZZ_SCRATCH_KIN"
    End If
Else
    If CodeFrame.Value = 4 Then
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_KIN.c_kin_id, ZZ_SCRATCH_KIN.c_kin_name, " + _
        "(ZZ_SCRATCH_KIN.c_up + ZZ_SCRATCH_KIN.c_down + " + _
        "ZZ_SCRATCH_KIN.c_marriage + ZZ_SCRATCH_KIN.c_collateral) as c_distance, " + _
        "str(ZZ_SCRATCH_KIN.c_kin_id) AS c_v_num, TRUE as c_delete FROM ZZ_SCRATCH_KIN"
    Else
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_KIN.c_kin_id, ZZ_SCRATCH_KIN.c_kin_chn, " + _
        "(ZZ_SCRATCH_KIN.c_up + ZZ_SCRATCH_KIN.c_down + " + _
        "ZZ_SCRATCH_KIN.c_marriage + ZZ_SCRATCH_KIN.c_collateral) as c_distance, " + _
        "str(ZZ_SCRATCH_KIN.c_kin_id) AS c_v_num, TRUE as c_delete FROM ZZ_SCRATCH_KIN"
    End If
End If

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now reset the original people on the list to distance = 0
'
If Left(TxtName.Value, 1) = "[" Then
    tQueryStr = "UPDATE ZZ_SCRATCH_IMPORT_PEOPLE INNER JOIN ZZ_SCRATCH_PAJEK ON " + _

```

```

= 0"
        "ZZ_SCRATCH_IMPORT_PEOPLE.c_person_id = ZZ_SCRATCH_PAJEK.c_ID SET ZZ_SCRATCH_PAJEK.c_distance
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
    End If
    '
    ' if needed, find the 0-degree nodes, using the edge list to mark the node list
    '
    If ChkDegree.Value Then
        '
        cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PAJEK INNER JOIN ZZ_SCRATCH_KINNET ON ZZ_SCRATCH_PAJEK.c_
id = ZZ_SCRATCH_KINNET.c_person_id " + _
        "SET ZZ_SCRATCH_PAJEK.c_delete = False"
        cmdSQL.Execute tRecDeleted
        '
        cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PAJEK INNER JOIN ZZ_SCRATCH_KINNET ON ZZ_SCRATCH_PAJEK.c_
id = ZZ_SCRATCH_KINNET.c_kin_id " + _
        "SET ZZ_SCRATCH_PAJEK.c_delete = False"
        cmdSQL.Execute tRecDeleted
        '
        ' remove records where c_degree is NULL
        '
        'MsgBox "Got through update"
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK WHERE ((ZZ_SCRATCH_PAJEK.c_delete) = TRUE )"
        cmdSQL.Execute tRecDeleted
    End If

    Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
    tRstNodeList.Index = "c_ID"
    '
    ' there probably is an SQL way to do this, but...
    '
    ti = 1
    With tRstNodeList
        .MoveFirst
        Do While Not .EOF
            .Edit
            !c_v_num = Trim(Str(ti))
            .Update
            ti = ti + 1
            .MoveNext
        Loop
    End With
    tRstNodeList.Close
    '
    ' fill in the edge list
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK_EDGE"
    cmdSQL.Execute tRecDeleted
    '
    ' this commented-out approach allows parallel edges.: NOTE: c_edge_desc MUST BE ADDED to the PRIMA
    '
    'tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK_EDGE ( c_node_1, c_node_2, c_edge_desc, c_edge_dist ) " + _
    "SELECT Val(ZZ_SCRATCH_PAJEK.c_v_num) AS c_node_1, Val(ZZ_SCRATCH_PAJEK_1.c_v_num) " + _
    "AS c_node_2, ZZ_SCRATCH_KINNET_EDGE.c_kin_rel, (ZZ_SCRATCH_KINNET_EDGE.c_upstep + " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_dwnstep + ZZ_SCRATCH_KINNET_EDGE.c_marstep + " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_colstep) AS c_edge_dist " + _
    "FROM (ZZ_SCRATCH_PAJEK INNER JOIN ZZ_SCRATCH_KINNET_EDGE ON ZZ_SCRATCH_PAJEK.c_ID = " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_person_id) INNER JOIN ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 " + _
    "ON ZZ_SCRATCH_KINNET_EDGE.c_kin_id = ZZ_SCRATCH_PAJEK_1.c_ID"

    ' because it is possible to have parallel edges (more than one significant kinship relation between t
    ' we need to do as we have done in other forms
    '
    ' fill the edge list
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK_EDGE ( c_node_1, c_node_2, c_edge_count, c_edge_dist ) " + _
    "SELECT Val([ZZ_SCRATCH_PAJEK].[c_v_num]) AS c_node_1, " + _
    "Val([ZZ_SCRATCH_PAJEK_1].[c_v_num]) AS c_node_2, " + _
    "Sum(ZZ_SCRATCH_KINNET_EDGE.c_kin_rel_count) AS SumOfc_kin_rel_count, " + _
    "Min((ZZ_SCRATCH_KINNET_EDGE.c_upstep + " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_dwnstep + ZZ_SCRATCH_KINNET_EDGE.c_marstep + " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_colstep)) AS MinOfc_edge_dist " + _
    "FROM ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 INNER JOIN " + _

```

```
"(ZZ_SCRATCH_KINNET_EDGE INNER JOIN ZZ_SCRATCH_PAJEK ON ZZ_SCRATCH_KINNET_EDGE.c_person_id = " +
"ZZ_SCRATCH_PAJEK.c_ID) ON ZZ_SCRATCH_PAJEK_1.c_ID = ZZ_SCRATCH_KINNET_EDGE.c_kin_id " + _
"GROUP BY Val([ZZ_SCRATCH_PAJEK].[c_v_num]), Val([ZZ_SCRATCH_PAJEK_1].[c_v_num])"
```

```
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
```

```
' *****If we are allowing parallel edges, this section is no longer relevant
```

```
' now fill in the edge description. This requires three steps
```

```
'cmdSQL.CommandText = "DROP TABLE tmp_scratch_pajek"
'cmdSQL.Execute tRecDeleted
```

```
tQueryStr = "SELECT ZZ_SCRATCH_PAJEK.c_ID, Val(ZZ_SCRATCH_PAJEK.c_v_num) AS c_v_num INTO " + _
"TMP_SCRATCH_PAJEK FROM ZZ_SCRATCH_PAJEK"
```

```
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
```

```
tQueryStr = "UPDATE ((ZZ_SCRATCH_KINNET_EDGE INNER JOIN TMP_SCRATCH_PAJEK ON " + _
"ZZ_SCRATCH_KINNET_EDGE.c_person_id = TMP_SCRATCH_PAJEK.c_ID) " + _
" INNER JOIN ZZ_SCRATCH_PAJEK_EDGE ON " + _
"TMP_SCRATCH_PAJEK.c_v_num = ZZ_SCRATCH_PAJEK_EDGE.c_node_1) " + _
"INNER JOIN TMP_SCRATCH_PAJEK AS TMP_SCRATCH_PAJEK_1 ON (TMP_SCRATCH_PAJEK_1.c_v_num = " + _
"ZZ_SCRATCH_PAJEK_EDGE.c_node_2) AND (ZZ_SCRATCH_KINNET_EDGE.c_kin_id = TMP_SCRATCH_PAJEK_1.c_
_ID) " + _
"SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = [ZZ_SCRATCH_KINNET_EDGE].[c_kin_rel] " + _
"WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_count)=1))"
```

```
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
```

```
cmdSQL.CommandText = "DROP TABLE tmp_scratch_pajek"
cmdSQL.Execute tRecDeleted
```

```
tQueryStr = "UPDATE ZZ_SCRATCH_PAJEK_EDGE SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = " + _
"'Parallel Edges merged' WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_count)>1))"
```

```
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
```

```
Set gRstEdge = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK_EDGE", dbOpenDynaset)
Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenDynaset)
tRstNodeList.MoveLast
```

```
tC = Chr(44) ' the comma
```

```
' first the nodes: define the record structure
```

```
With tRstNodeList
.MoveFirst
```

```
tStr = "*Vertices " + Trim(Str(tRstNodeList.RecordCount))
gStream.WriteText tStr, adWriteLine
```

```
Do While Not .EOF
```

```
gStream.WriteText !c_v_num + " " + Chr(34)
' name = the ID of the person
```

```
If ChkIncludeID.Value Then
gStream.WriteText !c_lbl + ":" + Trim(Str(!c_ID)) + Chr(34)
```

```
Else
gStream.WriteText !c_lbl + Chr(34)
End If
```

```
gStream.WriteText " box "
' label
```

```
If !c_distance > 19 Then
tStr = " ic " + tColor(20) + " bc " + tColor(20)
```

```
Else
tStr = " ic " + tColor(!c_distance + 1) + " bc " + tColor(!c_distance + 1)
End If
```

```
' color = white (1), blue (2), green (3), yellow (4), orange (5)
gStream.WriteText tStr, adWriteLine
```

```

        .MoveNext
    Loop
End With
'
' now the arcs:  define the record structure as directional
tStr = "*Arcs"
gStream.WriteText tStr, adWriteLine

With gRstEdge
    .MoveFirst
    Do While Not .EOF
        gStream.WriteText Trim(Str(!c_node_1)) + " " + Trim(Str(!c_node_2)) + " 1 1 "
        gStream.WriteText Chr(34)
        gStream.WriteText !c_edge_desc
        gStream.WriteText Chr(34)
        If !c_edge_dist > 19 Then
            gStream.WriteText " c " + tColor(20), adWriteLine
        ElseIf !c_edge_dist = 0 Then
            gStream.WriteText " c " + tColor(1), adWriteLine
        Else
            gStream.WriteText " c " + tColor(!c_edge_dist), adWriteLine
        End If
        .MoveNext
    Loop
End With

gRstNodeList.Close

' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite

gStream.Close
Set gStream = Nothing

'Set tRstNode = Nothing
'Set gRstEdge = Nothing
'Set gRstNodeList = Nothing

Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdUTF8Pajek_Click:
Exit Sub

Err_CmdUTF8Pajek_Click:
MsgBox Err.Description
Resume Exit_CmdUTF8Pajek_Click

End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim tRstDummy As DAO.Recordset
    Dim cmdDel As ADODB.Command, tRecDeleted As Long
    '
    ' Clear the input and output tables
    '
    Set cmdDel = New ADODB.Command
    cmdDel.ActiveConnection = CurrentProject.Connection
    cmdDel.CommandType = adCmdText
    cmdDel.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
    cmdDel.Execute tRecDeleted
    '
    Set gRstPersonID = Forms!LookAtKinship!frmZZ_SCRATCH_KIN.Form.Recordset
    Set gRstKinList = Forms!LookAtKinship!frmZZ_SCRATCH_KINNET.Form.Recordset
    '
    If (gRstPersonID.RecordCount + gRstKinList.RecordCount) > 0 Then
        cmdDel.CommandText = "Delete * from ZZ_SCRATCH_KIN"
    End If

```

```

        cmdDel.Execute tRecDeleted
        cmdDel.CommandText = "Delete * from ZZ_SCRATCH_KINNET"
        cmdDel.Execute tRecDeleted
        Set cmdDel = Nothing
    '
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_KIN", dbOpenDynaset)
    Set Forms!LookAtKinship!frmZZ_SCRATCH_KIN.Form.Recordset = tRstDummy
    gRstPersonID.Close
    Set gRstPersonID = CurrentDb.OpenRecordset("ZZ_SCRATCH_KIN", dbOpenDynaset)
    Set Forms!LookAtKinship!frmZZ_SCRATCH_KIN.Form.Recordset = gRstPersonID
    '
    Set Forms!LookAtKinship!frmZZ_SCRATCH_KINNET.Form.Recordset = tRstDummy
    gRstKinList.Close
    Set gRstKinList = CurrentDb.OpenRecordset("ZZ_SCRATCH_KINNET", dbOpenDynaset)
    Set Forms!LookAtKinship!frmZZ_SCRATCH_KINNET.Form.Recordset = gRstKinList
    '
    Set tRstDummy = Nothing
End If
'
ChkMourning.Value = False
ChkIncludeID.Value = False

gCurRecallSource = ""
'
'open a comforting message window
'
' first determine the language
gLCID = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
If gLCID = 2052 Or gLCID = 4100 Then      ' 2052 = PRC, 4100 = Singapore
    gDisplayLanguage = "S"
ElseIf gLCID = 3076 Or gLCID = 1028 Then ' 3076 = Hong Kong, 1028 = Taiwan
    gDisplayLanguage = "T"
    Call changeDisplayLanguage
Else
    gDisplayLanguage = "E"
    Call changeDisplayLanguage
End If

If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
    CmdRecallID.Enabled = True
End If
End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

    If gDisplayLanguage = "T" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "T"
    End If

    Call changeDisplayLanguage

Exit_CmdFanti_Click:
Exit Sub

Err_CmdFanti_Click:
MsgBox Err.Description
Resume Exit_CmdFanti_Click

End Sub

Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

    If gDisplayLanguage = "S" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "S"
    End If

    Call changeDisplayLanguage

Exit_CmdJianti_Click:
Exit Sub

Err_CmdJianti_Click:
MsgBox Err.Description
Resume Exit_CmdJianti_Click

```

End Sub

```

Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 34) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 34 And Not .EOF
            If !c_form = "LAK" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then
            tLang = 2
        Else
            tLang = 3
        End If
        '
        ' now comes the basic routine
        '

        Me.CmdSelectPerson.Caption = tLabelLanguage(tLang, 1)
        Me.CmdImport.Caption = tLabelLanguage(tLang, 2)
        Me.LblMourning.Caption = tLabelLanguage(tLang, 3)
        Me.LblMaxAncestor.Caption = tLabelLanguage(tLang, 4)
        Me.LblMaxDescend.Caption = tLabelLanguage(tLang, 5)
        Me.LblMaxCol.Caption = tLabelLanguage(tLang, 6)
        Me.LblMaxMar.Caption = tLabelLanguage(tLang, 7)
        Me.LblMaxLoop.Caption = tLabelLanguage(tLang, 8)
        Me.CmdRun.Caption = tLabelLanguage(tLang, 9)
        Me.CmdClose.Caption = tLabelLanguage(tLang, 10)
        Me.CmdUCINet.Caption = tLabelLanguage(tLang, 11)

        Me.CmdUTF8Pajek.Caption = tLabelLanguage(tLang, 12)
        Me.LblIncludeID.Caption = tLabelLanguage(tLang, 13)
        Me.LblPajek_Pinyin.Caption = tLabelLanguage(tLang, 14)
        Me.LblPajek_GB.Caption = tLabelLanguage(tLang, 15)
        Me.LblPajek_Big5.Caption = tLabelLanguage(tLang, 16)
        Me.CmdGUESS.Caption = tLabelLanguage(tLang, 17)
        Me.CmdGIS.Caption = tLabelLanguage(tLang, 18)
        Me.LblGIS_GB.Caption = tLabelLanguage(tLang, 19)
        Me.CmdFanti.Caption = tLabelLanguage(tLang, 20)
        Me.CmdJianti.Caption = tLabelLanguage(tLang, 21)
        Me.PageKinshipNet.Caption = tLabelLanguage(tLang, 22)
        Me.PageEgoRel.Caption = tLabelLanguage(tLang, 23)
        Me.LblSaveClipboard.Caption = tLabelLanguage(tLang, 24)
        Me.LblDisplay.Caption = tLabelLanguage(tLang, 25)
        Me.CmdHelp.Caption = tLabelLanguage(tLang, 26)
        Me.CmdStoreID.Caption = tLabelLanguage(tLang, 27)
        Me.CmdRecallID.Caption = tLabelLanguage(tLang, 28)
        Me.LblChkDegree.Caption = tLabelLanguage(tLang, 29)
        Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 30)
        Me.PagePeople.Caption = tLabelLanguage(tLang, 31)
        Me.LblChkSimplify.Caption = tLabelLanguage(tLang, 32)
        Me.LblChkEgo.Caption = tLabelLanguage(tLang, 33)
    End If

```

```

End If

End Sub

Private Sub CmdHelp_Click()
On Error GoTo Err_CmdHelp_Click
Dim tStrPDF As String

tStrPDF = Application.CurrentProject.Path + "\HelpFiles\HelpFile_LookAtKinship.pdf"

'MsgBox tStrPDF

Application.FollowHyperlink tStrPDF, , True

Exit_CmdHelp_Click:
Exit Sub

Err_CmdHelp_Click:
MsgBox Err.Description
Resume Exit_CmdHelp_Click

End Sub

Private Sub writeKML()

Dim tStrKML As String
'
' This program will dump the results to a .kml file
'
If frmZZ_SCRATCH_KIN.Form.Recordset.RecordCount = 0 Then
MsgBox "There are no records to save."
GoTo Exit_writeKML
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

If GISFrame.Value = 1 Then
tStream.Charset = "utf-8"
tCodeStr = "UTF8"
Else
tStream.Charset = "gb18030"
tCodeStr = "GB18030"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "kin_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
'
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
tFileName = tFN
If Not tFileName = "" Then
Exit For
End If
Next
If tFileName = "" Then
MsgBox "Bad file Name."
GoTo Exit_writeKML
Else
' make sure the file name has a txt extension
If Len(tFileName) < 5 Then
tFileName = tFileName + ".kml"
ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
tFileName = tFileName + ".kml"
End If

```



```

tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs, even where there should not be any
        '
        ' write the point header
        '
        tStream.WriteText tC + "<Placemark>", adWriteLine

        If IsNull(!c_kin_name) Then
            tStr = "[Bad Data] "
        Else
            tStr = !c_kin_name
        End If
        tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

        tStream.WriteText tC + tC + "<styleUrl>#kin-balloon-template</styleUrl>", adWriteLine
        '
        ' First Year as time stamp
        '
        If IsNull(!c_kin_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_kin_index_year)
        End If
        tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
        '
        tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#KinGISId" + tDQ + ">", adWriteLine
        '
        ' person ID
        '
        tStr = Str(!c_kin_id)
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "KinID" + tDQ + ">" + tStr + "<
/SchemaData>", adWriteLine
        '
        ' Person Name Chn
        '
        If IsNull(!c_kin_chn) Then
            tStr = tStr + "[Bad Data]"
        Else
            If Trim(!c_kin_chn) = "" Then
                tStr = "[?]"
            Else
                tStr = !c_kin_chn
            End If
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "KinNameHZ" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
        '
        ' Index Year
        '
        If IsNull(!c_kin_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_kin_index_year)
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
        '
        ' Address Name
        '
        If IsNull(!c_kin_addr_name) Then
            tStr = "[?]"
        ElseIf Trim(!c_kin_addr_name) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_kin_addr_name
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "KinAddrName" + tDQ + ">" + tStr
r + "</SimpleData>", adWriteLine
        '
        ' Address Name Chinese
        '
        If IsNull(!c_kin_addr_chn) Then
            tStr = "[?]"

```

```

        ElseIf Trim(!c_kin_addr_chn) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_kin_addr_chn
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "KinAddrHZ" + tDQ + ">" + tStr +
+ "</SimpleData>", adWriteLine
        '
        ' XY Count
        '
        If IsNull(!xy_count) Then
            tStr = "0"
        Else
            tStr = Str(!xy_count)
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
        '
        tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
        tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + "<Point>", adWriteLine
        '
        ' coordinates
        If IsNull(!kin_x_coord) Then
            tStr = "0"
        Else
            tStr = Str(!kin_x_coord)
        End If

        If IsNull(!kin_y_coord) Then
            tStr = tStr + ",0"
        Else
            tStr = tStr + "," + Str(!kin_y_coord)
        End If
        tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
        '
        ' footer
        '
        tStream.WriteText tC + tC + "</Point>", adWriteLine
        tStream.WriteText tC + tC + "</Placemark>", adWriteLine
        .MoveNext
    Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_writeKML:
Exit Sub

Err_writeKML:
MsgBox Err.Description
Resume Exit_writeKML

End Sub

Private Sub CmdStoreID_Click()
    Dim cmdSQL As ADODB.Command, tRecCount As Variant

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '

```

```

If DCount("?", "ZZ_STORE_PERSON_ID") > 0 Then
    ' Display message.
    If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) =
vbNo Then
        Exit Sub
    Else
        cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
        cmdSQL.Execute tRecCount
    End If
End If

tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id FROM
ZZ_SCRATCH_PEOPLE"

cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecCount
MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."
' update storage source
cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='Kinship' WHERE PersonIDSource.LineNum =1"
cmdSQL.Execute tRecCount

End Sub

Private Sub CmdRecallID_Click()
On Error GoTo Err_CmdRecallID_Click
Dim tStrSQL As String, tRecCount As Long, tStrQuestion As String, tRst As DAO.Recordset, tID As Long

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

tRecCount = DCount("?", "ZZ_SCRATCH_IMPORT_PEOPLE")

If tRecCount > 0 Then
    If tRecCount = 1 Then
        tStrQuestion = "Do you wish to replace the current person?"
    Else
        tStrQuestion = "Do you wish to replace the current list of IDs?"
    End If
    ' Display message.
    If MsgBox(tStrQuestion, vbYesNo + vbQuestion + vbDefaultButton2) = vbNo Then
        Exit Sub
    Else
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_IMPORT_PEOPLE"
        cmdSQL.Execute tRecCount
    End If
End If

' Clear the error table now that we are ready to go
'
cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted
' copy the IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_IMPORT_PEOPLE ( c_person_id ) SELECT DISTINCT c_personid FROM ZZ_STORE_PERS
ON_ID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted = 0 Then
    TxtName.Value = "[Error]"
    TxtNameChn.Value = "[Error]"
' ;
    CmdRun.Enabled = False
Else
    If tRecDeleted = 1 Then
        Set tRst = CurrentDb.OpenRecordset("SELECT ZZ_STORE_PERSON_ID.c_personid FROM ZZ_STORE_PERSON_ID")
        tRst.MoveFirst
        tID = tRst!c_personid

        Set tRst = CurrentDb.OpenRecordset("SELECT BIOG_MAIN.c_name, BIOG_MAIN.c_name_chn FROM BIOG_MAIN WHER
E (((BIOG_MAIN.c_personid)=" + Str(tID) + ")))")
        tRst.MoveFirst

        TxtName.Value = tRst!c_name
        TxtNameChn.Value = tRst!c_name_chn
        tRst.Close
        Set tRst = Nothing
    End If
End If

```

```

Else
    TxtName.Value = "[Recalled List]"
    TxtNameChn.Value = "[" + ChrW(&H53EC) + ChrW(&H56DE) + ChrW(&H7684) + ChrW(&H4EBA) + ChrW(&H540D) + "
]"
End If

Set tRst = CurrentDb.OpenRecordset("SELECT PersonIDSource.SourceForm FROM PersonIDSource WHERE (((PersonIDSource.LineNum)=1))")
tRst.MoveFirst
gCurRecallSource = tRst!SourceForm
tRst.Close
Set tRst = Nothing

CmdRun.Enabled = True
End If

Set cmdSQL = Nothing

Exit_CmdRecallID_Click:
Exit Sub

Err_CmdRecallID_Click:
MsgBox Err.Description
Resume Exit_CmdRecallID_Click

End Sub

Private Sub CmdInfo_Click()
On Error GoTo Err_CmdInfo_Click

If Me.Dirty Then Me.Dirty = False
DoCmd.Close

Exit_CmdInfo_Click:
Exit Sub

Err_CmdInfo_Click:
MsgBox Err.Description
Resume Exit_CmdInfo_Click

End Sub

Private Sub saveNeo4jFiles()
On Error GoTo Err_CmdNeo4j_Click

'
' This program will dump the results of the search to four CSV files
'
' for the moment I'll just describe the format of the CSV file
' Note: Neo4j seems to treat all fields as strings, so there is no need to explicitly mark strings
'
' People.CSV
' nameID, NameHZ, NamePY, indexyear, sex
'     nameID = c_person_id
'     nameHZ = c_name_chn
'     namePY = c_name
'     indexyear = c_index_year
'     personDynasty = c_dynasty
'     sex = c_female > (F,M)
'
' Places.CSV
'     placeID = c_addr_id
'     placeHZ = c_addr_chn
'     placePY = c_addr_name
'     placeX = x_coord
'     placeY = y_coord
'
' PeoplePlaces.CSV
'     nameID
'     placeID
'     personPlaceRelation
'
' PeopleKinship.CSV
' node1_ID, node2_ID, kinshipRelation
'     node1 = str(c_person_id) for node1
'     node2 = str(c_node_id) for node2
'     kinshipRelation = c_link_desc
'
' first see if there are any records to process
'

```

```

If frmZZ_SCRATCH_KIN.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdNeo4j_Click
End If
'
' allocate the file variables
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
'
' next get the People file
'
Dim tRstNode As DAO.Recordset, tRstEdge As DAO.Recordset, tRstPlace As DAO.Recordset, tRstPeoplePlace As DAO.
Recordset
Dim tStr As String, tC As String, ti As Integer, tUseList As Boolean
Dim tColor(50) As String, tMetricSum As Integer, tQueryStr As String, tPersonID As Long
Dim gStream As ADODB.Stream, tCodeStr As String
'
' the optional recordsets
'
Dim tRstOffice As DAO.Recordset, tRstPostings As DAO.Recordset
'
'Dim tFileSystem, tGDF

' set up the stream to write to

Set gStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
    gStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf CodeFrame.Value = 2 Then
    gStream.Charset = "big5"
    tCodeStr = "BIG5"
ElseIf CodeFrame.Value = 3 Then
    gStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    gStream.Charset = "ascii"
    tCodeStr = "ascii"
End If

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
    'Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)
    '
    ' we have a file name: now open the stream for writing

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open
    '
    ' prepare the temp tables for the people, place, peoplePlace and kinship data

```

```

Dim cmdSQL As ADODB.Command
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_KINNET_EDGE"
cmdSQL.Execute tRecDeleted
'
'   copy the data for determining edges
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_KINNET_EDGE SELECT ZZ_SCRATCH_KINNET.* FROM ZZ_SCRATCH_KINNET"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'   now delete the unneeded edges
'
tQueryStr = "UPDATE (ZZ_SCRATCH_KINNET_EDGE INNER JOIN ZZ_SCRATCH_KINNET_EDGE AS " + _
    "ZZ_SCRATCH_KINNET_EDGE_1 ON ZZ_SCRATCH_KINNET_EDGE.c_person_id = ZZ_SCRATCH_KINNET_EDGE_1.c_pers
on_id) " + _
    "INNER JOIN ZZ_SCRATCH_KINNET_EDGE AS ZZ_SCRATCH_KINNET_EDGE_2 ON " + _
    "(ZZ_SCRATCH_KINNET_EDGE_1.c_kin_id = ZZ_SCRATCH_KINNET_EDGE_2.c_person_id) AND " + _
    "(ZZ_SCRATCH_KINNET_EDGE.c_kin_id = ZZ_SCRATCH_KINNET_EDGE_2.c_kin_id) " + _
    "SET ZZ_SCRATCH_KINNET_EDGE.c_delete = 1 " + _
    "WHERE (((ZZ_SCRATCH_KINNET_EDGE.c_upstep)=[ZZ_SCRATCH_KINNET_EDGE_1].[c_upstep])+ " + _
    "[ZZ_SCRATCH_KINNET_EDGE_2].[c_upstep]) AND ((ZZ_SCRATCH_KINNET_EDGE.c_dwnstep)=" + _
    "[ZZ_SCRATCH_KINNET_EDGE_1].[c_dwnstep]+[ZZ_SCRATCH_KINNET_EDGE_2].[c_dwnstep]) AND " + _
    "((ZZ_SCRATCH_KINNET_EDGE.c_marstep)=[ZZ_SCRATCH_KINNET_EDGE_1].[c_marstep])+ " + _
    "[ZZ_SCRATCH_KINNET_EDGE_2].[c_marstep]) AND ((ZZ_SCRATCH_KINNET_EDGE.c_colstep)=" + _
    "[ZZ_SCRATCH_KINNET_EDGE_1].[c_colstep]+[ZZ_SCRATCH_KINNET_EDGE_2].[c_colstep]))"
'
' MsgBox "About to prune"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_KINNET_EDGE WHERE c_delete = 1"
cmdSQL.Execute tRecDeleted
' MsgBox "Pruning complete"
'
'   now collect the node information
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_GEPHI_NODE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_GEPHI_NODE_DISTINCT"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_GEPHI_NODE ( c_person_id, c_name, c_name_chn, c_index_year, c_fem
ale, c_addr_id, c_addr_name, c_addr_chn, " + _
    "x_coord, y_coord, c_dy, c_dynasty, c_dynasty_chn) " + _
    "SELECT ZZ_SCRATCH_KINNET_EDGE.c_person_id, ZZ_SCRATCH_KINNET_EDGE.c_name, ZZ_SCRATCH_KINNET_EDGE
.c_name_chn, " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_index_year, ZZ_SCRATCH_KINNET_EDGE.c_female, ZZ_SCRATCH_KINNET_EDGE.c_a
ddr_id, ZZ_SCRATCH_KINNET_EDGE.c_addr_name, " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_addr_chn, ZZ_SCRATCH_KINNET_EDGE.x_coord, ZZ_SCRATCH_KINNET_EDGE.y_coord,
" + _
    "ZZ_SCRATCH_KINNET_EDGE.c_dy, ZZ_SCRATCH_KINNET_EDGE.c_dynasty, ZZ_SCRATCH_KINNET_EDGE.c_dynasty_
chn " + _
    "FROM ZZ_SCRATCH_KINNET_EDGE"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_GEPHI_NODE ( c_person_id, c_name, c_name_chn, c_index_year, c_fem
ale, c_addr_id, c_addr_name, c_addr_chn, " + _
    "x_coord, y_coord, c_dy, c_dynasty, c_dynasty_chn) " + _
    "SELECT ZZ_SCRATCH_KINNET_EDGE.c_kin_id, ZZ_SCRATCH_KINNET_EDGE.c_kin_name, ZZ_SCRATCH_KINNET_EDG
E.c_kin_chn, " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_kin_index_year, ZZ_SCRATCH_KINNET_EDGE.c_kin_female, ZZ_SCRATCH_KINNET_
EDGE.c_kin_addr_id, ZZ_SCRATCH_KINNET_EDGE.c_kin_addr_name, " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_kin_addr_chn, ZZ_SCRATCH_KINNET_EDGE.kin_x_coord, ZZ_SCRATCH_KINNET_EDG
E.kin_y_coord, " + _
    "ZZ_SCRATCH_KINNET_EDGE.c_kin_dy, ZZ_SCRATCH_KINNET_EDGE.c_kin_dynasty, ZZ_SCRATCH_KINNET_EDGE.c_
kin_dynasty_chn " + _
    "FROM ZZ_SCRATCH_KINNET_EDGE"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'

```

```

        ' append the results
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_GEPHI_NODE_DISTINCT ( c_person_id, c_name, c_name_chn, c_index_year, c_female, c_imported, c_addr_id, c_addr_name, c_addr_chn, " + _
        "x_coord, y_coord, c_dy, c_dynasty, c_dynasty_chn)" + _
        "SELECT DISTINCT ZZ_SCRATCH_GEPHI_NODE.c_person_id, ZZ_SCRATCH_GEPHI_NODE.c_name, ZZ_SCRATCH_GEPHI_NODE.c_name_chn, " + _
        "ZZ_SCRATCH_GEPHI_NODE.c_index_year, ZZ_SCRATCH_GEPHI_NODE.c_female, FALSE AS c_imported, " + _
        "ZZ_SCRATCH_GEPHI_NODE.c_addr_id, ZZ_SCRATCH_GEPHI_NODE.c_addr_name, ZZ_SCRATCH_GEPHI_NODE.c_addr_chn, ZZ_SCRATCH_GEPHI_NODE.x_coord, " + _
        "ZZ_SCRATCH_GEPHI_NODE.y_coord, ZZ_SCRATCH_GEPHI_NODE.c_dy, ZZ_SCRATCH_GEPHI_NODE.c_dynasty, ZZ_SCRATCH_GEPHI_NODE.c_dynasty_chn " + _
        "FROM ZZ_SCRATCH_GEPHI_NODE"

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SCRATCH_KINNET_EDGE", dbOpenDynaset)
    Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_GEPHI_NODE_DISTINCT", dbOpenDynaset)
    tRstNode.MoveLast
    '
    ' process the four tables
    '
    tC = Chr(44) ' the comma
    '
    ' first the nodes: define the record structure
    '
    ' if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a pinyin field
    If tCodeStr = "ascii" Then
        tStr = "nameID" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
    Else
        tStr = "nameID" + tC + "nameHZ" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
    End If
    gStream.WriteText tStr, adWriteLine
    'tGDF.WriteLine (tStr)
    '
    With tRstNode
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_person_id)) + tC
            '
            ' name
            '
            If tCodeStr = "ascii" Then
                If IsNull(!c_name) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
            Else
                If IsNull(!c_name_chn) Then
                    tStr = tStr + "Missing" + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If
            End If

            If IsNull(!c_name) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        End If
        '
        ' indexyear = c_index_year INT
        '
        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + tC
        End If

        ' dynasty information
        '
        If IsNull(!c_dynasty) Then
            tStr = tStr + "unknown" + tC
        Else
            If tCodeStr = "ascii" Then
                tStr = tStr + !c_dynasty + tC
            End If
        End If
    End With

```

```

Else
    tStr = tStr + !c_dynasty_chn + tC
End If
End If
'
sex = c_female > (F,M)
tStr = tStr + IIf(!c_female, "F", "M")
'
gStream.WriteText tStr, adWriteLine
'
.MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' data from source of PersonIDs
'
tQueryStr = ""
If Not (gCurRecallSource = "") Then
    ' clear the scratch table

    cmdSQL.CommandText = "DELETE * FROM zz_addresses"
    cmdSQL.Execute tRecDeleted
    '
    Select Case gCurRecallSource
        Case "Office"
            '
            If MsgBox("Do you wish to include office data?", vbYesNo + vbQuestion + vbDefaultButton2) = v
bYes Then
                '
                ' get all the addresses for the postings for the node personIDs
                '
                cmdSQL.CommandText = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord,
y_coord ) " + _
                    "SELECT DISTINCT ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_id, ZZZ_POSTED_TO_ADDR_DATA.c_
office_addr_name, " + _
                    "ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_chn, " + _
                    "ZZZ_POSTED_TO_ADDR_DATA.office_x_coord, ZZZ_POSTED_TO_ADDR_DATA.office_y_coord "
+ _
                    "FROM ZZ_SCRATCH_GEPHI_NODE_DISTINCT INNER JOIN ZZZ_POSTED_TO_ADDR_DATA ON " + _
                    "ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c_person_id = ZZZ_POSTED_TO_ADDR_DATA.c_personid
" + _
                    "WHERE (((ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_id) Is Not Null))"
                cmdSQL.Execute tRecCount
                '
                ' get all the address for the people
                '
                cmdSQL.CommandText = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord,
y_coord ) " + _
                    "SELECT DISTINCT ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c_addr_id, ZZ_SCRATCH_GEPHI_NODE_DIST
INCT.c_addr_name, " + _
                    "ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c_addr_chn, ZZ_SCRATCH_GEPHI_NODE_DISTINCT.x_coord, " + _
                    "ZZ_SCRATCH_GEPHI_NODE_DISTINCT.y_coord " + _
                    "FROM ZZ_SCRATCH_GEPHI_NODE_DISTINCT"
                cmdSQL.Execute tRecCount
                '
                ' now get all the postings
                '
                cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_OFFICE"
                cmdSQL.Execute tRecCount
                '
                cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_OFFICE ( c_personid, c_posting_id, c_office
id, c_firstyear, " + _
                    "c_lastyear, c_office_chn, c_office_trans, c_office_pinyin, c_office_addr_id ) "
+ _
                    "SELECT DISTINCT ZZZ_POSTED_TO_ADDR_DATA.c_personid, ZZZ_POSTED_TO_ADDR_DATA.c_postin
g_id, ZZZ_POSTED_TO_ADDR_DATA.c_office_id, " + _
                    "ZZZ_POSTED_TO_ADDR_DATA.c_firstyear, ZZZ_POSTED_TO_ADDR_DATA.c_lastyear, ZZZ_POS

```



```

TED_TO_ADDR_DATA.c_office_chn, " + _
    "ZZZ_POSTED_TO_ADDR_DATA.c_office_trans, ZZZ_POSTED_TO_ADDR_DATA.c_office_pinyin,
ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_id " + _
    "FROM ZZ_SCRATCH_GEPHI_NODE_DISTINCT INNER JOIN ZZZ_POSTED_TO_ADDR_DATA " + _
    "ON ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c_person_id = ZZZ_POSTED_TO_ADDR_DATA.c_person
id;"

cmdSQL.Execute tRecCount
'
' process the postings
'
' get a file name
dlgSaveAs.InitialFileName = "Postings_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_personid, ZZ_SCRATCH_OFFICE.c_postin
g_id, ZZ_SCRATCH_OFFICE.c_office_id, " + _
        "ZZ_SCRATCH_OFFICE.c_firstyear, ZZ_SCRATCH_OFFICE.c_lastyear, ZZ_SCRATCH_OFFICE.c
_office_addr_id FROM ZZ_SCRATCH_OFFICE"
    Set tRstPostings = CurrentDb.OpenRecordset(tQueryStr)

    tStr = "PersonID" + tC + "PostingID" + tC + "OfficeID" + tC + "PostingFirstYear" + tC
+ "PostingLastYear" + tC + "PostingAddrID"
    gStream.WriteText tStr, adWriteLine

    With tRstPostings
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_personid)) + tC
            tStr = tStr + Trim(Str(!c_posting_id)) + tC
            tStr = tStr + Trim(Str(!c_office_id)) + tC
            '
            If IsNull(!c_firstyear) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_firstyear)) + tC
            End If
            '
            If IsNull(!c_lastyear) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_lastyear)) + tC
            End If
            '
            If IsNull(!c_office_addr_id) Then
                tStr = tStr + "0"
            Else
                tStr = tStr + Trim(Str(!c_office_addr_id))
            End If
            '
            gStream.WriteText tStr, adWriteLine
            '
            .MoveNext
        Loop
    End With
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite

```

```

        ,
        gStream.Close
End If
'
' now process the offices from the postings
'
' get a file name
dlgSaveAs.InitialFileName = "Offices_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    ' get the offices
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_office_id, ZZ_SCRATCH_OFFICE.c_offic
e_chn, " + _
                "ZZ_SCRATCH_OFFICE.c_office_pinyin, ZZ_SCRATCH_OFFICE.c_office_trans FROM ZZ_SCRA
TCH_OFFICE"

    Set tRstOffice = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "OfficeID" + tC + "OfficePY" + tC + "OfficeTrans"
    Else
        tStr = "OfficeID" + tC + "OfficePY" + tC + "OfficeHZ" + tC + "OfficeTrans"
    End If
    gStream.WriteText tStr, adWriteLine

    With tRstOffice
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_office_id)) + tC
            ,
            ' name
            ,
            If IsNull(!c_office_pinyin) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_office_pinyin + tC
            End If

            If Not (tCodeStr = "ascii") Then
                If IsNull(!c_office_chn) Then
                    tStr = tStr + "Missing" + tC
                Else
                    tStr = tStr + !c_office_chn + tC
                End If
            End If

            If IsNull(!c_office_trans) Then
                tStr = tStr + "Missing"
            Else
                tStr = tStr + !c_office_trans
            End If
            ,
            gStream.WriteText tStr, adWriteLine
            ,
            .MoveNext
        Loop
    End With
    ' now make sure all the data is copied to tStream

```

```

        gStream.Flush
        ' and write the stream to the file
        gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
End If
'
' finally, redefine the Places query
tQueryStr = "SELECT DISTINCT ZZ_ADDRESSES.c_addr_id, ZZ_ADDRESSES.c_name AS c_addr_name,
" + _
        "ZZ_ADDRESSES.c_name_chn AS c_addr_chn, ZZ_ADDRESSES.x_coord, ZZ_ADDRESSES.y_coord FR
OM ZZ_ADDRESSES"
'
' clean up
Set tRstPostings = Nothing
Set tRstOffice = Nothing
End If
'
Case Else
End Select
End If
'
' now places
'
' get a file name
'
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    ' now process the file
    '
    ' One may need to add place IDs for postings, entry, social institutions, or texts, depending on lis
t source
    '
    ' default query string
    '
    If tQueryStr = "" Then
        tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c_addr_id, ZZ_SCRATCH_GEPHI_NODE_DIST
INCT.c_addr_name, " + _
            "ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c_addr_chn, ZZ_SCRATCH_GEPHI_NODE_DISTINCT.x_coord, ZZ_SCRATC
H_GEPHI_NODE_DISTINCT.y_coord " + _
            "FROM ZZ_SCRATCH_GEPHI_NODE_DISTINCT"
    End If
    '
    Set tRstPlace = CurrentDb.OpenRecordset(tQueryStr)
    '
    If tCodeStr = "ascii" Then
        tStr = "placeID" + tC + "placePY" + tC + "placeX" + tC + "placeY"
    Else
        tStr = "placeID" + tC + "placePY" + tC + "placeHZ" + tC + "placeX" + tC + "placeY"
    End If
    gStream.WriteText tStr, adWriteLine
    With tRstPlace
        .MoveFirst
        Do While Not .EOF
            ' the ID of the place
            If Not IsNull(!c_addr_id) Then
                tStr = Trim(Str(!c_addr_id)) + tC
            '

```

```

        ' address name

        If IsNull(!c_addr_name) Then
            tStr = tStr + "unknown" + tC
        Else
            tStr = tStr + !c_addr_name + tC
        End If
    '
    If Not (tCodeStr = "ascii") Then
        If IsNull(!c_addr_chn) Then
            tStr = tStr + "unknown" + tC
        Else
            tStr = tStr + !c_addr_chn + tC
        End If
    End If

    ' latitude = !y_coord
    If IsNull(!y_coord) Then
        tStr = tStr + "0.0" + tC
    Else
        tStr = tStr + Str(!y_coord) + tC
    End If

    ' longitude = !x_coord
    If IsNull(!x_coord) Then
        tStr = tStr + "0.0"
    Else
        tStr = tStr + Str(!x_coord)
    End If
    '
    gStream.WriteText tStr, adWriteLine
End If
.MoveNext
Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces
'
dlgSaveAs.InitialFileName = "PeoplePlaces_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c_person_id, ZZ_BIOG_MAIN.c_index_addr_i"
    d, ZZ_BIOG_MAIN.c_index_addr_type_desc, " + _
        "ZZ_BIOG_MAIN.c_index_addr_type_chn " + _
        "FROM ZZ_SCRATCH_GEPHI_NODE_DISTINCT INNER JOIN ZZ_BIOG_MAIN ON ZZ_SCRATCH_GEPHI_NODE_DISTINCT.c"
    _person_id = ZZ_BIOG_MAIN.c_personid"

    Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

```

```

tStr = "nameID" + tC + "placeID" + tC + "personPlaceTrans" + tC + "personPlaceHZ"

gStream.WriteText tStr, adWriteLine

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_index_addr_id) Then
            '
            tStr = Trim(Str(!c_person_id)) + tC
            '
            tStr = tStr + Trim(Str(!c_index_addr_id)) + tC
            '
            tStr = tStr + !c_index_addr_type_desc + tC + !c_index_addr_type_chn
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
dlgSaveAs.InitialFileName = "Kinship_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    ' now the edges: define the record structure
    tStr = "node1" + tC + "node2" + tC + "kinshipRelation"
    gStream.WriteText tStr, adWriteLine
    'tGDF.WriteLine (tStr)

    With tRstEdge
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_kin_rel) Then
                tStr = Trim(Str(!c_person_id)) + tC
                ' node1 = str(c_person_id) for node1
                tStr = tStr + Trim(Str(!c_kin_id)) + tC
                ' node2 = str(c_node_id) for node2
                tStr = tStr + !c_kin_rel
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    '
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file

```

```
        gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close
    Set gStream = Nothing
    'tGDF.Close
    ,
    Set tRstNode = Nothing
    Set tRstEdge = Nothing
    'Set tGDF = Nothing
    'Set tFileSystem = Nothing
Else
    'The user pressed Cancel.
End If
MsgBox "Finished saving to Neo4j"

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click

End Sub
```

frm_LookAtOffice - 1

Option Compare Database

```
Public gRstPeople As DAO.Recordset, gDisplayLanguage As String, gLabelsOK As Boolean,
    gImportPlacesPeople As Boolean, gImportPlacesOffice As Boolean, gUseOfficeADDRID As Boolean, gUsePeopleAD
DRID As Boolean, gUseOfficeID As Boolean
Public gFromDynasty As Integer, gToDynasty As Integer, gUseIndexYears As Boolean, gUseDynasties As Boolean,
    gFromDynastyBegin As Integer, gFromDynastyEnd As Integer, gToDynastyBegin As Integer, gToDynastyEnd As In
teger, gUseOfficeYears As Boolean
```

Private Sub CmdAllDynasties_Click()

```
    gFromDynasty = -2
    gToDynasty = -2
    TxtFromDynasty.Value = ""
    TxtFromDynastyPY.Value = "All"
    TxtToDynasty.Value = ""
    TxtToDynastyPY.Value = "All"
```

End Sub

Private Sub CmdFromDynasty_Click()

```
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strFromDynasty As String
```

```
    If gFromDynasty < 0 Then
        strFromDynasty = ""
    Else
        strFromDynasty = Str(gFromDynasty)
    End If
```

stDocName = "frmPickDynasty"

DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

```
    Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
    gFromDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value
```

```
    Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
    gFromDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value
```

```
    Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
    gFromDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
```

```
    ' check to see if we have a problem and reject selection
```

```
    If gToDynasty > -1 Then
```

```
        If gFromDynastyBegin > gToDynastyEnd Then
```

```
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' D
ynasty ends!", vbExclamation
```

```
            gFromDynasty = -1
            TxtFromDynasty.Value = ""
            TxtFromDynastyPY.Value = ""
```

```
        End If
```

```
    End If
```

```
    ' value is OK
```

```
    If gFromDynasty > -1 Then
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
        TxtFromDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
        TxtFromDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
```

```
    End If
```

DoCmd.Close acForm, stDocName

```
    ' reset ToDynasty if necessary (-2 = all dynasties)
```

```
    If gToDynasty = -2 Then
```

```
        gToDynasty = -1
        TxtToDynasty.Value = ""
        TxtToDynastyPY.Value = ""
```

```
    End If
```

End Sub

End Sub

```

Private Sub CmdGISPeople_Click()
On Error GoTo Err_CmdGISPeople_Click
'
' If it is a KML file, call the routine and exit
'
If ChkPeopleKML.Value Then
    Call writePersonKML
    Exit Sub
End If
'
' This program will dump the results to a .gis file
'
If ZZ_SCRATCH_P_OFFICE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGISPeople_Click
End If
'
If FrameGISPeople.Value = 1 Then
    tCodeStr = "GB18030"
Else
    tCodeStr = "UTF8"
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tC As String
Dim tRstGIS As DAO.Recordset
Dim tStr As String
Dim gStream As ADODB.Stream

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "office_people_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
'
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdGISPeople_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If
'
' write the file
'
'SELECT ZZ_SCRATCH_P_OFFICE.c_name AS Name, ZZ_SCRATCH_P_OFFICE.c_name_chn AS NameChn,
'ZZ_SCRATCH_P_OFFICE.c_sex AS Sex, ZZ_SCRATCH_P_OFFICE.c_index_year AS IndexYear,
'ZZ_SCRATCH_P_OFFICE.c_addr_id AS AddrID, ZZ_SCRATCH_P_OFFICE.c_addr_name AS AddrName,
'ZZ_SCRATCH_P_OFFICE.c_addr_chn AS AddrChn, Str(ZZ_SCRATCH_P_OFFICE.x_coord) AS X,
'Str(ZZ_SCRATCH_P_OFFICE.y_coord) AS Y, ZZ_SCRATCH_P_OFFICE.xy_count AS XYcount
'
' process the table
'
'DoCmd.TransferText acExportDelim, , "OFFICE_PEOPLE_GIS_QUERY", tFileName, True

' we have a file name: now open the stream for writing
'
Set gStream = New ADODB.Stream
gStream.Mode = adModeReadWrite
gStream.Type = adTypeText

tC = Chr(9) ' the tab
'
If FrameGISOffice.Value = 1 Then
    gStream.Charset = "GB18030"
Else
    gStream.Charset = "utf-8"

```



```

End If
'
gStream.Open
'
' process the table
'
Set tRstGIS = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_OFFICE", dbOpenDynaset)
'
' write the header
'
tStr = "Name" + tC + "NameChn" + tC + "IndexYear" + tC + "Sex" + tC + "AddrName" + tC + "AddrChn" + tC +
      "X" + tC + "Y" + tC + "xy_count"
gStream.WriteText tStr, adWriteLine
'
With tRstGIS
  .MoveFirst
  Do While Not .EOF
    tStr = ""
    '
    If IsNull(!c_name) Then
      tStr = "[Name Missing]"
    Else
      tStr = !c_name
    End If
    '
    If IsNull(!c_name_chn) Then
      tStr = tStr + tC + "[Name Missing]"
    Else
      tStr = tStr + tC + !c_name_chn
    End If
    '
    If IsNull(!c_index_year) Then
      tStr = tStr + tC + "[ ]"
    Else
      tStr = tStr + tC + Str(!c_index_year)
    End If
    '
    If IsNull(!c_sex) Then
      tStr = tStr + tC + "[ ]"
    Else
      tStr = tStr + tC + !c_sex
    End If
    '
    If IsNull(!c_addr_name) Then
      tStr = tStr + tC + "[Addr Name Missing]"
    Else
      tStr = tStr + tC + !c_addr_name
    End If
    '
    If IsNull(!c_addr_chn) Then
      tStr = tStr + tC + "[Addr Chn Missing]"
    Else
      tStr = tStr + tC + !c_addr_chn
    End If
    '
    '
    If IsNull(!x_coord) Then
      tStr = tStr + tC + "[ ]"
    Else
      tStr = tStr + tC + CStr(!x_coord)
    End If
    '
    If IsNull(!y_coord) Then
      tStr = tStr + tC + "[ ]"
    Else
      tStr = tStr + tC + CStr(!y_coord)
    End If
    '
    If IsNull(!xy_count) Then
      tStr = tStr + tC + "[ ]"
    Else
      tStr = tStr + tC + Str(!xy_count)
    End If
    '
    If Not (tStr = "") Then
      gStream.WriteText tStr, adWriteLine
    End If
    .MoveNext
  Loop

```

```

        End With
        ' now make sure all the data is copied to tStream
        gStream.Flush
        ' and write the stream to the file
        gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
End If

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGISPeople_Click:
Exit Sub

Err_CmdGISPeople_Click:
MsgBox Err.Description
Resume Exit_CmdGISPeople_Click

End Sub

Private Sub CmdHelp_Click()
    Dim tStrPDF As String

    tStrPDF = Application.CurrentProject.Path + "\HelpFiles\HelpFile_LookAtOffices.pdf"

    'MsgBox tStrPDF

    Application.FollowHyperlink tStrPDF, , True

End Sub

Private Sub CmdAllOffices_Click()
On Error GoTo Err_CmdAllOffices_Click

    TxtOfficeID.Value = -1

    TxtOfficeChn.Value = ""
    TxtOfficeDesc.Value = ""
    TxtTypeDesc.Value = ""
    TxtTypeChn.Value = ""
    gUseOfficeID = False
    CmdPickOffice.SetFocus
    CmdAllOffices.Enabled = False

    If Not gUseOfficeADDRID And Not gUsePeopleADDRID Then
        CmdQuery.Enabled = False
    End If

Exit_CmdAllOffices_Click:
Exit Sub

Err_CmdAllOffices_Click:
MsgBox Err.Description
Resume Exit_CmdAllOffices_Click

End Sub

Private Sub CmdAllPlacesOffices_Click()
On Error GoTo Err_CmdAllPlacesOffices_Click

    TxtOfficeAddrID.Value = -1

    TxtPlaceOfficeChn.Value = ""
    TxtPlaceOfficePY.Value = ""
    gUseOfficeADDRID = False
    If gUsePeopleADDRID = False Then
        ChkUseXY.Enabled = False
    End If
    ChkSubUnitsOffice.Enabled = False
    CmdPlaceOffice.SetFocus
    CmdAllPlacesOffices.Enabled = False

    If IsNull(TxtOfficeID.Value) Then
        CmdQuery.Enabled = False
    End If

Exit_CmdAllPlacesOffices_Click:
Exit Sub

```

```

Err_CmdAllPlacesOffices_Click:
    MsgBox Err.Description
    Resume Exit_CmdAllPlacesOffices_Click

End Sub

Private Sub CmdAllPlacesPeople_Click()
On Error GoTo Err_CmdAllPlacesPeople_Click

    TxtPersonAddrID.Value = -1

    TxtPlacePeopleChn.Value = ""
    TxtPlacePeoplePY.Value = ""
    gUsePeopleADDRID = False
    If gUseOfficeADDRID = False Then
        ChkUseXY.Enabled = False
    End If
    CmdPlacePeople.SetFocus
    CmdAllPlacesPeople.Enabled = False

    If IsNull(TxtOfficeID.Value) Then
        CmdQuery.Enabled = False
    End If

Exit_CmdAllPlacesPeople_Click:
    Exit Sub

Err_CmdAllPlacesPeople_Click:
    MsgBox Err.Description
    Resume Exit_CmdAllPlacesPeople_Click

End Sub

Private Sub CmdImportOffices_Click()
On Error GoTo Err_CmdImportOffices_Click

    Dim stDocName As String, tRstOffices As DAO.Recordset
    Dim stLinkCriteria As String, tRstImportOffices As DAO.Recordset
    Dim tString As String, tOfficeID As Long, ti As Integer, tStrID As String, tQuit As Boolean
    Dim tLen As Integer, cmdSQL As ADODB.Command

    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tFileSystem, tList

    ' first see if we already have a list

    tQuit = False
    '
    If Not tQuit Then
        ' open the list

        Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

        'Use a With...End With block to reference the FileDialog object.
        With dlgSaveAs
            .InitialFileName = ""
            If .Show = -1 Then
                '
                tFileName = ""
                For Each tFN In .SelectedItems
                    tFileName = tFN
                    If Not tFileName = "" Then
                        Exit For
                    End If
                Next
                If tFileName = "" Then
                    MsgBox "Bad file Name."
                    GoTo Exit_CmdImportOffices_Click
                End If
            End If
        End With
        '
        ' Clear the address table now that we are ready to go
        '

        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
    
```

```

cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
cmdSQL.Execute tRecDeleted
'

cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "OfficeListImport Specification", "TempImportList", tFileName, 0
'
'   TransferType=acImportDelim
'   SpecificationName = "TempImportList" (apparently it is saved in the database itself)
'   TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
'   HasFieldNames = False (0)
'
'   copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
"FROM OFFICE_CODES RIGHT JOIN TempImportList ON OFFICE_CODES.c_office_id = TempImportList.ImportID "
+ _
"WHERE (((OFFICE_CODES.c_office_id) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If
'
'   copy the good IDs
'

tStrSQL = "INSERT INTO ZZ_OFFICE_CODE ( c_office_id ) SELECT DISTINCT TempImportList.ImportID " + _
"FROM OFFICE_CODES INNER JOIN TempImportList ON OFFICE_CODES.c_office_id = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

Me.TxtTypeDesc.Value = ""
Me.TxtTypeChn.Value = ""
If tRecDeleted > 0 Then
    Me.TxtOfficeDesc.Value = "[Imported List]"
    Me.TxtOfficeChn.Value = "[Imported List]"
    Me.CmdAllOffices.Enabled = True
    Me.CmdQuery.Enabled = True
    Me.CmdSaveOffices.Enabled = True
Else
    Me.TxtOfficeDesc.Value = ""
    Me.TxtOfficeChn.Value = ""
    Me.CmdAllOffices.Enabled = False
    Me.CmdQuery.Enabled = False
    Me.CmdSaveOffices.Enabled = False
End If

Set cmdSQL = Nothing
End If

Exit_CmdImportOffices_Click:
Exit Sub

Err_CmdImportOffices_Click:
MsgBox Err.Description
Resume Exit_CmdImportOffices_Click

End Sub

Private Sub CmdImportPlaceOffice_Click()
    On Error GoTo Err_CmdImportPlaceOffice_Click

    Dim stDocName As String, tRstAddresses As DAO.Recordset
    Dim stLinkCriteria As String, tRstImportPlaces As DAO.Recordset
    Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String, tQuit As Boolean
    Dim tLen As Integer, cmdSQL As ADODB.Command

    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tFileSystem, tList

    ' first see if we already have a list

```

```

tQuit = False
'
If Not tQuit Then
    ' open the list

    Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

    'Use a With...End With block to reference the FileDialog object.
    With dlgSaveAs
        .InitialFileName = ""
        If .Show = -1 Then
            '
            tFileName = ""
            For Each tFN In .SelectedItems
                tFileName = tFN
                If Not tFileName = "" Then
                    Exit For
                End If
            Next
            If tFileName = "" Then
                MsgBox "Bad file Name."
                GoTo Exit_CmdImportPlaceOffice_Click
            End If
        End If
    End With

    ' Clear the address table now that we are ready to go
    '

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_OFFICE"
    cmdSQL.Execute tRecDeleted
    '
    cmdSQL.CommandText = "Delete * from InputErrorList"
    cmdSQL.Execute tRecDeleted

    cmdSQL.CommandText = "Delete * from TempImportList"
    cmdSQL.Execute tRecDeleted

    DoCmd.TransferText acImportDelim, "ImportPlaceList_Space", "TempImportList", tFileName, 0
    ' TransferType=acImportDelim
    ' SpecificationName = "TempImportList" (apparently it is saved in the database itself)
    ' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
    ' HasFieldNames = False (0)
    '
    ' copy the bad IDs
    '
    tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
        "FROM ADDR_CODES RIGHT JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID " + _
        "WHERE (((ADDR_CODES.c_addr_id) Is Null))"

    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted

    If tRecDeleted > 0 Then
        MsgBox "Some ID were not successfully imported: please look at InputErrorList."
    End If
    '
    ' copy the good IDs
    '
    tStrSQL = "INSERT INTO ZZ_SCRATCH_ADDR_OFFICE ( c_addr_id ) SELECT DISTINCT TempImportList.ImportID " + _
        "FROM ADDR_CODES INNER JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID"

    cmdSQL.CommandText = tStrSQL
    cmdSQL.Execute tRecDeleted

    If tRecDeleted > 0 Then
        Me.TxtPlaceOfficeChn.Value = "[Imported List]"
        Me.TxtPlaceOfficePY.Value = "[Imported List]"
        gUseOfficeADDRID = True
        ChkUseXY.Enabled = True
        ChkSubUnitsOffice.Enabled = True
    End If

    Set cmdSQL = Nothing
End If

Exit_CmdImportPlaceOffice_Click:

```

```

Exit Sub

Err_CmdImportPlaceOffice_Click:
MsgBox Err.Description
Resume Exit_CmdImportPlaceOffice_Click

End Sub

Private Sub CmdImportPlacePeople_Click()
On Error GoTo Err_CmdImportPlacePeople_Click

Dim stDocName As String, tRstAddresses As DAO.Recordset
Dim stLinkCriteria As String, tRstImportPlaces As DAO.Recordset
Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String, tQuit As Boolean
Dim tLen As Integer, cmdSQL As ADODB.Command

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant

' first see if we already have a list

tQuit = False
'
If Not tQuit Then
' open the list

Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
.InitialFileName = ""
If .Show = -1 Then
'
tFileName = ""
For Each tFN In .SelectedItems
tFileName = tFN
If Not tFileName = "" Then
Exit For
End If
Next
If tFileName = "" Then
MsgBox "Bad file Name."
GoTo Exit_CmdImportPlacePeople_Click
End If
End With
' Clear the address table now that we are ready to go
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_PEOPLE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "ImportPlaceList_Space", "TempImportList", tFileName, 0
' TransferType=acImportDelim
' SpecificationName = "TempImportList" (apparently it is saved in the database itself)
' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
' HasFieldNames = False (0)
'
' copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
"FROM ADDR_CODES RIGHT JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID " + _
"WHERE ((ADDR_CODES.c_addr_id) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If

```

```

'
'   copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_ADDR_PEOPLE ( c_addr_id ) SELECT DISTINCT TempImportList.ImportID " + _
"FROM ADDR_CODES INNER JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    Me.TxtPlacePeopleChn.Value = "[Imported List]"
    Me.TxtPlacePeoplePY.Value = "[Imported List]"
    gUsePeopleADDRID = True
    ChkUseXY.Enabled = True
    ChkSubUnitsPeople.Enabled = True
End If

Set cmdSQL = Nothing
End If

Exit_CmdImportPlacePeople_Click:
Exit Sub

Err_CmdImportPlacePeople_Click:
MsgBox Err.Description
Resume Exit_CmdImportPlacePeople_Click

End Sub

Private Sub CmdNeo4j_Click()
On Error GoTo Err_CmdNeo4j_Click
'
'   This program will dump the results of the search to five CSV files
'
'   for the moment I'll just describe the format of the CSV file
'   Note: Neo4j seems to treat all fields as strings, so there is no need to explicitly mark strings
'
'   1. People.CSV
'       nameID = c_person_id
'       nameHZ = c_name_chn
'       namePY = c_name
'       indexyear = c_index_year
'       personDynasty = c_dynasty
'       sex = c_female > (F,M)
'
'   2. Places.CSV
'       placeID = c_addr_id
'       placeHZ = c_addr_chn
'       placePY = c_addr_name
'       placeX = x_coord
'       placeY = y_coord
'
'   3. PeoplePlaces.CSV
'       nameID
'       placeID
'       personPlaceRelation
'
'   4. PeoplePlaceCodes
'
'   5. PeopleOffice.CSV
'       nameID = str(c_person_id)
'       officeID = str(c_node_id)
'       officePlaceID
'       kinID
'       kinRelID
'       AssocPersonID
'       AssocRelID
'       SocialInstID
'       SocialInstNameID
'       EntryYear
'       EntryDynasty
'
'   6. OfficeCodes.CSV
'       officeID = str(c_office_id)
'       officeDesc = c_office_desc
'
'   7. Institution codes
'
'   first see ifthere are any records to process
'

```

```

If Me.ZZ_SCRATCH_OFFICE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdNeo4j_Click
End If
'
' warn the user that a lot of files will be created
'
MsgBox "Neo4j requires that from 6 to 7 files be created."
'
' allocate the file variables
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
'
' next get the People file
'
Dim tRstPeople As DAO.Recordset, tRstOfficeCodes As DAO.Recordset, tRstPlace As DAO.Recordset
Dim tRstPostings As DAO.Recordset, tRstPeoplePlace As DAO.Recordset, tStr As String, tC As String, ti As Integer
Dim tQueryStr As String
Dim gStream As ADODB.Stream, tCodeStr As String
'
' the optional recordset
'
Dim tRstInstitutions As DAO.Recordset
'
'Dim tFileSystem, tGDF
'
' set up the stream to write to
Set gStream = New ADODB.Stream
'
' for the moment, set the character set to UTF-8
gStream.Charset = "utf-8"
tCodeStr = "UTF8"
'If CodeFrame.Value = 1 Then
'    gStream.Charset = "utf-8"
'    tCodeStr = "UTF8"
'ElseIf CodeFrame.Value = 2 Then
'    gStream.Charset = "big5"
'    tCodeStr = "BIG5"
'ElseIf CodeFrame.Value = 3 Then
'    gStream.Charset = "gb2312"
'    tCodeStr = "GB2312"
'Else
'    gStream.Charset = "ascii"
'    tCodeStr = "ascii"
'End If
tC = Chr(44) ' the comma
'
' prepare the temp tables for the people, place, peoplePlace and office data
Dim cmdSQL As ADODB.Command
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' start with people
'
' clear ZZ_SCRATCH_PEOPLE and copy the records
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_dynasty, c_dynasty_chn, c_female, c_addr_id, " + _
" c_addr_name, c_addr_chn, c_addr_type, c_addr_desc, c_addr_desc_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_personid, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_chn, ZZ_Z_BIOG_MAIN.c_index_year, " + _
"ZZZ_BIOG_MAIN.c_dynasty, ZZZ_BIOG_MAIN.c_dynasty_chn, ZZZ_BIOG_MAIN.c_female, ZZZ_BIOG_MAIN.c_index_addr_id, " + _
"ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_MAIN.c_index_addr_chn, ZZZ_BIOG_MAIN.c_index_addr_type_code, ZZZ_BIOG_MAIN.c_index_addr_type_desc, " + _
"ZZZ_BIOG_MAIN.c_index_addr_type_chn, ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord " + _
"FROM ZZ_SCRATCH_OFFICE INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_OFFICE.c_personid = ZZZ_BIOG_MAIN.c_personid"
'
cmdSQL.CommandText = tQueryStr

```



```

cmdSQL.Execute tRecDeleted
'
Set tRstPostings = CurrentDb.OpenRecordset("ZZ_SCRATCH_OFFICE", dbOpenDynaset)
Set tRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)

' Open the People file

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    ' we have a file name: now open the stream for writing

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    tRstPeople.MoveLast
    '
    ' process the four tables
    '
    ' first the nodes: define the record structure
    '
    ' if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a pin
    yin field
    If tCodeStr = "ascii" Then
        tStr = "nameID" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
    Else
        tStr = "nameID" + tC + "nameHZ" + tC + "namePY" + tC + "indexyear" + tC + "dynasty" + tC + "sex"
    End If
    gStream.WriteText tStr, adWriteLine
    '
    With tRstPeople
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_person_id)) + tC
            '
            ' name
            '
            If tCodeStr = "ascii" Then
                If IsNull(!c_name) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
            Else
                If IsNull(!c_name_chn) Then
                    tStr = tStr + "Missing" + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If

                If IsNull(!c_name) Then
                    tStr = tStr + "Missing" + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
            End If
        End While
    End With

```

```

        End If
        '
        '   indexyear = c_index_year INT
        '
        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + tC
        End If

        '   dynasty information
        '
        If IsNull(!c_dynasty) Then
            tStr = tStr + "unknown" + tC
        Else
            If tCodeStr = "ascii" Then
                tStr = tStr + !c_dynasty + tC
            Else
                tStr = tStr + !c_dynasty_chn + tC
            End If
        End If

        '
        '   sex = c_female > (F,M)
        tStr = tStr + IIf(!c_female, "F", "M")
        '
        gStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now the PeopleOffice file
'
dlgSaveAs.InitialFileName = "PeopleOffice_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        '   make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    tStr = "NameID" + tC + "OfficeCode" + tC + "OfficeAddrID" + tC + "SocialInstID" + tC + "PostingFirstYear"
+ tC + "
        "PostingLastYear" + tC + "PostingDynasty"
    gStream.WriteText tStr, adWriteLine
    '
    With tRstPostings
        .MoveFirst
        Do While Not .EOF
            '   the ID of the person
            tStr = Trim(Str(!c_personid)) + tC
            '

```

```

        ' office code
        '
        If IsNull(!c_office_id) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_office_id)) + tC
        End If
        '
        ' office addr id
        '
        If IsNull(!c_office_addr_id) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_office_addr_id)) + tC
        End If
        '
        ' social inst ID
        '
        If IsNull(!c_inst_code) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Right("000000" + Trim(Str(!c_inst_code)), 6) + Right("000000" + Trim(Str(!c_inst_
t_name_code)), 6) + tC
        End If
        '
        ' posting first year
        '
        If IsNull(!c_firstyear) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_firstyear)) + tC
        End If
        '
        ' posting last year
        '
        If IsNull(!c_lastyear) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_lastyear)) + tC
        End If
        '
        ' posting dynasty
        '
        If IsNull(!c_dy) Then
            tStr = tStr + "0"
        Else
            tStr = tStr + Trim(Str(!c_dy))
        End If
        '
        gStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now places
'
' get a file name
'
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then

```

```

        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    ' now process the file
    '
    ' there are three sources of places: the list of people, the posting locations, and the list of institut
ions
    ' since ZZ_SCRATCH_P_TEXT has the required fields, just reuse it before copying to ZZ_ADDRESSES
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_TEXT"
    cmdSQL.Execute tRecDeleted
    '
    ' get the people IDs
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_addr_id, ZZ_SCRATCH_PEOPLE.c_addr_name, ZZ_SCRATCH_PEOP
E.c_addr_chn, " + _
        "ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord " + _
        "FROM ZZ_SCRATCH_PEOPLE"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_office_addr_id, ZZ_SCRATCH_OFFICE.c_office_addr_name, ZZ
_SCRATCH_OFFICE.c_office_addr_chn, " + _
        "ZZ_SCRATCH_OFFICE.office_x_coord, ZZ_SCRATCH_OFFICE.office_y_coord " + _
        "FROM ZZ_SCRATCH_OFFICE " + _
        "WHERE (((ZZ_SCRATCH_OFFICE.c_office_addr_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT SOCIAL_INSTITUTION_ADDR.c_inst_addr_id, ADDR_CODES.c_name, ADDR_CODES.c_name
_chn, ADDR_CODES.x_coord, ADDR_CODES.y_coord " + _
        "FROM ADDR_CODES INNER JOIN (ZZ_SCRATCH_OFFICE INNER JOIN SOCIAL_INSTITUTION_ADDR " + _
        "ON (ZZ_SCRATCH_OFFICE.c_inst_name_code = SOCIAL_INSTITUTION_ADDR.c_inst_name_code) " + _
        "AND (ZZ_SCRATCH_OFFICE.c_inst_code = SOCIAL_INSTITUTION_ADDR.c_inst_code)) " + _
        "ON (ADDR_CODES.c_addr_id = SOCIAL_INSTITUTION_ADDR.c_inst_addr_id) AND (ADDR_CODES.c_add
r_id = SOCIAL_INSTITUTION_ADDR.c_inst_addr_id) " + _
        "WHERE (((ZZ_SCRATCH_OFFICE.c_inst_code)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' now copy the results
    cmdSQL.CommandText = "Delete * from ZZ_ADDRESSES"
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_addr_id, ZZ_SCRATCH_P_TEXT.c_addr_name, ZZ_SCRATCH_P_TEX
T.c_addr_chn, " + _
        "ZZ_SCRATCH_P_TEXT.x_coord, ZZ_SCRATCH_P_TEXT.y_coord " + _
        "FROM ZZ_SCRATCH_P_TEXT WHERE (((ZZ_SCRATCH_P_TEXT.c_addr_id)>0))"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    Set tRstPlace = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
    '
    If tCodeStr = "ascii" Then
        tStr = "placeID" + tC + "placePY" + tC + "placeX" + tC + "placeY"
    Else
        tStr = "placeID" + tC + "placePY" + tC + "placeHZ" + tC + "placeX" + tC + "placeY"
    End If
    gStream.WriteText tStr, adWriteLine
    With tRstPlace
        .MoveFirst

```

```

Do While Not .EOF
    ' the ID of the place
    If Not IsNull(!c_addr_id) Then
        tStr = Trim(Str(!c_addr_id)) + tC
        '
        ' address name
        If IsNull(!c_name) Then
            tStr = tStr + "unknown" + tC
        Else
            tStr = tStr + !c_name + tC
        End If
        '
        If Not (tCodeStr = "ascii") Then
            If IsNull(!c_name_chn) Then
                tStr = tStr + "unknown" + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If
        End If

        ' latitude = !y_coord
        If IsNull(!y_coord) Then
            tStr = tStr + "0.0" + tC
        Else
            tStr = tStr + Str(!y_coord) + tC
        End If

        ' longitude = !x_coord
        If IsNull(!x_coord) Then
            tStr = tStr + "0.0"
        Else
            tStr = tStr + Str(!x_coord)
        End If
        '
        gStream.WriteText tStr, adWriteLine
    End If
    .MoveNext
Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces: use ZZ_SCRATCH_PEOPLE
'
dlgSaveAs.InitialFileName = "PeoplePlaces_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_person_id, ZZ_SCRATCH_PEOPLE.c_addr_id, ZZ_SCRATCH_PEOPLE.c_addr_type " + _

```

```

        "FROM ZZ_SCRATCH_PEOPLE WHERE (((ZZ_SCRATCH_PEOPLE.c_addr_id) > 0))"

Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

tStr = "nameID" + tC + "placeID" + tC + "personPlaceCode"

gStream.WriteText tStr, adWriteLine

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_addr_id) Then
            ,
            tStr = Trim(Str(!c_person_id)) + tC
            ,
            tStr = tStr + Trim(Str(!c_addr_id)) + tC
            ,
            tStr = tStr + Trim(Str(!c_addr_type))
            ,
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
,
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
,
' now peoplePlaceCode: use ZZ_SCRATCH_PEOPLE
,
dlgSaveAs.InitialFileName = "PeoplePlacesCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_addr_type, ZZ_SCRATCH_PEOPLE.c_addr_desc, ZZ_SCRATCH_PEO
PLE.c_addr_desc_chn " + _
        "FROM ZZ_SCRATCH_PEOPLE WHERE (((ZZ_SCRATCH_PEOPLE.c_addr_type) > 0))"

Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "personPlaceCode" + tC + "personPlaceTrans"
Else
    tStr = "personPlaceCode" + tC + "personPlaceTrans" + tC + "personPlaceHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_addr_type) Then

```

```

        '
        tStr = Trim(Str(!c_addr_type)) + tC
        '
        tStr = tStr + !c_addr_desc
        '
        If Not (tCodeStr = "ascii") Then
            tStr = tStr + tC + !c_addr_desc_chn
        End If
        gStream.WriteText tStr, adWriteLine
    End If
    .MoveNext
Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' finally, get office codes and institution codes, if there are any
'
' now the EntryCode file
'
dlgSaveAs.InitialFileName = "OfficeCode_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    If tCodeStr = "ascii" Then
        tStr = "OfficeCode" + tC + "OfficeTrans" + tC + "OfficePinyin"
    Else
        tStr = "OfficeCode" + tC + "OfficeTrans" + tC + "OfficePinyin" + tC + "OfficeHZ"
    End If
    gStream.WriteText tStr, adWriteLine
    '
    ' get the codes
    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_office_id, ZZ_SCRATCH_OFFICE.c_office_trans, ZZ_SCRATCH_
OFFICE.c_office_pinyin, ZZ_SCRATCH_OFFICE.c_office_chn " + _
        "FROM ZZ_SCRATCH_OFFICE"
    Set tRstOfficeCode = CurrentDb.OpenRecordset(tQueryStr)
    With tRstOfficeCode
        .MoveFirst
        Do While Not .EOF
            '
            tStr = Trim(Str(!c_office_id)) + tC
            '
            ' office trans
            '
            If IsNull(!c_office_trans) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + Trim(!c_office_trans) + tC
            End If
        Loop
    End With
End With

```

```

        End If
        '
        ' office pinyin
        '
        If IsNull(!c_office_pinyin) Then
            tStr = tStr + "Missing"
        Else
            tStr = tStr + Trim(!c_office_pinyin)
        End If
        '
        ' office HZ
        '
        If Not (tCodeStr = "ascii") Then
            tStr = tStr + tC + Trim(!c_office_chn)
        End If
        '
        gStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted
'
' the final selection is for social institutions
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id ) " + _
            "SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_personid " + _
            "FROM ZZ_SCRATCH_OFFICE " + _
            "WHERE (((ZZ_SCRATCH_OFFICE.c_inst_code)>0))"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "SELECT ZZ_SCRATCH_OFFICE.c_inst_code, ZZ_SCRATCH_OFFICE.c_inst_name_code, ZZ_SCRATCH_OFFICE.c_in
st_name_hz, ZZ_SCRATCH OFFICE.c inst name py " + _
            "FROM ZZ_SCRATCH_OFFICE WHERE (((ZZ_SCRATCH_OFFICE.c_inst_code)>0))"

If tRecDeleted > 0 Then
    dlgSaveAs.InitialFileName = "InstitutionCodes_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
    End If
    '
    gStream.Open
    '
    Set tRstInstitutions = CurrentDb.OpenRecordset(tQueryStr)
    '
    If tCodeStr = "ascii" Then

```



```

        tStr = "InstitutionCode" + tC + "InstitutionNamePY"
    Else
        tStr = "InstitutionCode" + tC + "InstitutionNamePY" + tC + "InstitutionNameHZ"
    End If
    gStream.WriteText tStr, adWriteLine
    'tGDF.WriteLine (tStr)

    With tRstAssocCodes
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_inst_code) Then
                tStr = Right("000000" + Trim(Str(!c_inst_code)), 6) + Right("000000" + Trim(Str(!c_inst_n
ame_code)), 6) + tC
                '
                If IsNull(!c_inst_name_py) Then
                    tStr = tStr + "NameMissing"
                Else
                    tStr = tStr + Trim(!c_inst_name_py)
                End If
                '
                If Not (tCodeStr = "ascii") Then
                    If IsNull(!c_inst_name_hz) Then
                        tStr = tStr + tC + "NameMissing"
                    Else
                        tStr = tStr + tC + !c_inst_name_hz
                    End If
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    '
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
End If
End If

cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
cmdSQL.Execute tRecDeleted
'
MsgBox "Finished saving to Neo4j"
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click

End Sub

Private Sub CmdPlaceOffice_Click()
On Error GoTo Err_CmdPlaceOffice_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strADDR As String

TxtOfficeAddrID.Visible = True
TxtOfficeAddrID.SetFocus
strADDR = TxtOfficeAddrID.Text

stDocName = "frmPickAddresses_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR

If CurrentProject.AllForms("frmPickAddresses_multi").IsLoaded Then

    ' if the user selected a group of addresses, ZZ_ADDRESSES will have records

Dim tAddrID As Long, tRstAddr As DAO.Recordset
Dim strADDR_CHN As String, strADDR_PY As String

```

```

Dim cmdSQL As ADODB.Command

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

gUseOfficeADDRID = True
CmdAllPlacesOffices.Enabled = True
ChkUseXY.Enabled = True
ChkSubUnitsOffice.Enabled = True

'MsgBox "Checking zz_addresses"
' tRstAddresses.MoveFirst
Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Visible = True
Forms!frmPickAddresses_multi.Form!TxtAddrFilter.SetFocus
If Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Value Then
    '
    TxtOfficeAddrID.Value = 0
    strADDR_PY = Forms!frmPickAddresses_multi.Form!TxtFilterPY
    strADDR_CHN = Forms!frmPickAddresses_multi.Form!TxtFilterChn

    If strADDR_CHN = "" Then
        TxtPlaceOfficeChn.Value = "[[Filter]]"
        TxtPlaceOfficePY.Value = "[[" + strADDR_PY + "]"
    Else
        TxtPlaceOfficeChn.Value = "[[" + strADDR_CHN + "]"
        TxtPlaceOfficePY.Value = "[[Filter]]"
    End If
Else
    Forms!frmPickAddresses_multi.Form!TxtSelectCount.Visible = True
    Forms!frmPickAddresses_multi.Form!TxtSelectCount.SetFocus
    If Forms!frmPickAddresses_multi.Form!TxtSelectCount.Value > 1 Then
        TxtPlaceOfficeChn.Value = "[[" + ChrW(22810) + ChrW(36984) + "]"
        TxtPlaceOfficePY.Value = "[[Multi-Select]]"
        TxtOfficeAddrID.Value = 0
    Else
        ' only one record in ZZ_ADDRESSES: get its field values
        '
        Set tRstAddr = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
        tRstAddr.MoveFirst
        'MsgBox "Checking zz_addresses: no records"
        TxtOfficeAddrID.Value = tRstAddr!c_addr_id
        TxtPlaceOfficeChn.Value = tRstAddr!c_name_chn
        TxtPlaceOfficePY.Value = tRstAddr!c_name
        tRstAddr.Close
        Set tRstAddr = Nothing
    End If
End If

' now copy the records
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_OFFICE"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_OFFICE ( c_addr_id ) SELECT DISTINCT " + _
    "ZZ_ADDRESSES.c_addr_id FROM ZZ_ADDRESSES"
cmdSQL.Execute tRecDeleted
DoCmd.Close acForm, "frmPickAddresses_multi"

CmdQuery.Enabled = True
End If
CmdPlaceOffice.SetFocus
TxtOfficeAddrID.Visible = False

Exit_CmdPlaceOffice_Click:
Exit Sub

Err_CmdPlaceOffice_Click:
MsgBox Err.Description
Resume Exit_CmdPlaceOffice_Click

End Sub

Private Sub CmdPlacePeople_Click()
On Error GoTo Err_CmdPlacePeople_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strADDR As String

```

```

TxtPersonAddrID.Visible = True
TxtPersonAddrID.SetFocus
strADDR = TxtPersonAddrID.Text

```

```

stDocName = "frmPickAddresses_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR

```

```

If CurrentProject.AllForms("frmPickAddresses_multi").IsLoaded Then

```

```

    ' if the user selected a group of addresses, ZZ_ADDRESSES will have records

```

```

    Dim tAddrID As Long, tRstAddr As DAO.Recordset
    Dim strADDR_CHN As String, strADDR_PY As String
    Dim cmdSQL As ADODB.Command

```

```

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

```

```

    gUsePeopleADDRID = True
    CmdAllPlacesPeople.Enabled = True
    ChkUseXY.Enabled = True

```

```

    'MsgBox "Checking zz_addresses"

```

```

    ' tRstAddresses.MoveFirst

```

```

    Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Visible = True

```

```

    Forms!frmPickAddresses_multi.Form!TxtAddrFilter.SetFocus

```

```

    If Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Value Then

```

```

        TxtPersonAddrID.Value = 0

```

```

        strADDR_PY = Forms!frmPickAddresses_multi.Form!TxtFilterPY

```

```

        strADDR_CHN = Forms!frmPickAddresses_multi.Form!TxtFilterChn

```

```

        If strADDR_CHN = "" Then

```

```

            TxtPlacePeopleChn.Value = "[[Filter]]"

```

```

            TxtPlacePeoplePY.Value = "[" + strADDR_PY + "]"

```

```

        Else

```

```

            TxtPlacePeopleChn.Value = "[" + strADDR_CHN + "]"

```

```

            TxtPlacePeoplePY.Value = "[[Filter]]"

```

```

        End If

```

```

    Else

```

```

        Forms!frmPickAddresses_multi.Form!TxtSelectCount.Visible = True

```

```

        Forms!frmPickAddresses_multi.Form!TxtSelectCount.SetFocus

```

```

        If Forms!frmPickAddresses_multi.Form!TxtSelectCount.Value > 1 Then

```

```

            TxtPlacePeopleChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"

```

```

            TxtPlacePeoplePY.Value = "[[Multi-Select]]"

```

```

            TxtPersonAddrID.Value = 0

```

```

        Else

```

```

            ' only one record in ZZ_ADDRESSES: get its field values

```

```

            Set tRstAddr = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
            tRstAddr.MoveFirst

```

```

            'MsgBox "Checking zz_addresses: no records"

```

```

            TxtPersonAddrID.Value = tRstAddr!c_addr_id

```

```

            TxtPlacePeopleChn.Value = tRstAddr!c_name_chn

```

```

            TxtPlacePeoplePY.Value = tRstAddr!c_name

```

```

            tRstAddr.Close

```

```

            Set tRstAddr = Nothing

```

```

        End If

```

```

    End If

```

```

    '

```

```

    ' now copy the records

```

```

    '

```

```

    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_PEOPLE"

```

```

    cmdSQL.Execute tRecDeleted

```

```

    cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_PEOPLE ( c_addr_id ) SELECT DISTINCT " + _
        "ZZ_ADDRESSES.c_addr_id FROM ZZ_ADDRESSES"

```

```

    cmdSQL.Execute tRecDeleted

```

```

    DoCmd.Close acForm, "frmPickAddresses_multi"

```

```

    CmdQuery.Enabled = True

```

```

End If

```

```

CmdPlacePeople.SetFocus

```

```

TxtPersonAddrID.Visible = False

```

```

Exit_CmdPlacePeople_Click:

```

```

Exit Sub

```

```

Err_CmdPlacePeople_Click:

```

```

MsgBox Err.Description
Resume Exit_CmdPlacePeople_Click

End Sub

Private Sub CmdPickOffice_Click()
    On Error GoTo Err_CmdPickOffice_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strOffice As String

    TxtOfficeID.Visible = True
    TxtOfficeID.SetFocus
    strOffice = TxtOfficeID.Text

    stDocName = "frmPickOfficeTree_multi_2"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strOffice

    If CurrentProject.AllForms("frmPickOfficeTree_multi_2").IsLoaded Then
        Dim tOfficeID As Long
        Dim strOffice_DESC As String, strOffice_DESC_chn As String, tStrDynasty As String, tStrDynastyChn As String, _
            strOfficeType_DESC As String, strOfficeType_DESC_chn As String

        Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeCode.Visible = True
        Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeCode.SetFocus
        tOfficeID = Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeCode.Value
        ' MsgBox "Office code: " + Str(Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeCode.Value)
        Forms!frmPickOfficeTree_multi_2.Form!TxtSearch.SetFocus
        Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeCode.Visible = False
        TxtOfficeID.Value = tOfficeID

        'MsgBox "Office code: " + Str(tOfficeID)

        ' We need to get the office name first to get the dynasty, if needed

        Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDesc.Visible = True
        Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDesc.SetFocus

        If IsNull(Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDesc.Value) Then
            strOffice_DESC = ""
        Else
            strOffice_DESC = Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDesc.Value
        End If

        Forms!frmPickOfficeTree_multi_2.Form!subTreeView.SetFocus
        Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDesc.Visible = False

        TxtOfficeDesc.Value = strOffice_DESC
        tStrDynasty = strOffice_DESC

        Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDescChn.Visible = True
        Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDescChn.SetFocus
        If IsNull(Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDescChn.Value) Then
            strOffice_DESC_chn = ""
        Else
            strOffice_DESC_chn = Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDescChn.Value
        End If
        Forms!frmPickOfficeTree_multi_2.Form!subTreeView.SetFocus
        Forms!frmPickOfficeTree_multi_2.Form!TxtOfficeDescChn.Visible = False
        TxtOfficeChn.Value = strOffice_DESC_chn

        ' now we get the type descriptions, which is the dynasty for Office ID > 0

        Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDesc.Visible = True
        Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDesc.SetFocus
        If IsNull(Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDesc.Value) Then
            strOfficeType_DESC = ""
        Else
            strOfficeType_DESC = Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDesc.Value
        End If
        Forms!frmPickOfficeTree_multi_2.Form!TxtSearch.SetFocus
        Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDesc.Visible = False
        TxtTypeDesc.Value = strOfficeType_DESC

        Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDescChn.Visible = True
        Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDescChn.SetFocus
        If IsNull(Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDescChn.Value) Then
            strOfficeType_DESC_chn = ""
        Else

```

```

        strOfficeType_DESC_chn = Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDescChn.Value
    End If
    Forms!frmPickOfficeTree_multi_2.Form!subTreeView.SetFocus
    Forms!frmPickOfficeTree_multi_2.Form!TxtTypeDescChn.Visible = False
    TxtTypeChn.Value = strOfficeType_DESC_chn

    'MsgBox "Office ID = " + Str(tOfficeID)

    If TxtOfficeID.Value < 0 Then
        '
        ' Note: the query will use a join of ZZ_SCRATCH_OFFICE_CODES, the table used by frmPickOfficeTree t
o store office code values
        ' unless the type description is "N/A", which means that no ALL office codes are to be used
        '
        tStrDynasty = strOffice_DESC
        tStrDynastyChn = strOffice_DESC_chn
        If TxtOfficeID.Value = -1 Then
            If tStrDynasty = "" Then
                TxtOfficeDesc.Value = "[All]"
                TxtOfficeChn.Value = "[All]"
            Else
                TxtOfficeDesc.Value = "[" + tStrDynasty + "]"
                TxtOfficeChn.Value = "[" + tStrDynastyChn + "]"
            End If
        Else
            TxtOfficeDesc.Value = "[Multi-Select]"
            TxtOfficeChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
        End If

        If TxtTypeDesc.Value = "" Then
            If TxtTypeChn = "" Then
                TxtTypeDesc.Value = "[All]"
                TxtTypeChn.Value = ""
            End If
            If TxtOfficeID.Value = -1 Then
                gUseOfficeID = False
            Else
                gUseOfficeID = True
            End If
        Else
            gUseOfficeID = True
        End If
        CmdAllOffices.Enabled = True
    Else
        tStrDynasty = strOfficeType_DESC
        tStrDynastyChn = strOfficeType_DESC_chn

        TxtTypeDesc.Value = tStrDynasty
        TxtTypeChn.Value = tStrDynastyChn
        CmdAllOffices.Enabled = True
        gUseOfficeID = True
    End If

    DoCmd.Close acForm, stDocName
    '
    CmdQuery.Enabled = True
    CmdSaveOffices.Enabled = True
Else
    If IsNull(TxtOfficeID.Value) Then
        CmdQuery.Enabled = False
        CmdAllOffices.Enabled = False
        CmdSaveOffices.Enabled = False
        gUseOfficeID = False
    End If
End If

CmdPickOffice.SetFocus
TxtOfficeID.Visible = False

Exit_CmdPickOffice_Click:
Exit Sub

Err_CmdPickOffice_Click:
MsgBox Err.Description
Resume Exit_CmdPickOffice_Click

End Sub

```

```

Private Sub CmdQuery_Click()
    On Error GoTo Err_CmdQuery_Click

    Dim tRstOffice As DAO.Recordset, tRstDummy As DAO.Recordset
    Dim tStrQuery As String, tQueryInsertStr As String
    Dim tQuerySelectStr As String, tRecCount As Long, tUseYears As Boolean, tStrAndYears As String, tStrWhereYear
s As String
    Dim cmdSQL As ADODB.Command

    Set cmdSQL = New ADODB.Command
    '
    ' clear the tables
    '
    Set tRstOffice = ZZ_SCRATCH_OFFICE.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SO", dbOpenDynaset)
    Set ZZ_SCRATCH_OFFICE.Form.Recordset = tRstDummy
    tRstOffice.Close
    '

    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_OFFICE"
    cmdSQL.Execute tRecDeleted
    '
    ' now the people table
    '
    Set gRstPeople = ZZ_SCRATCH_P_OFFICE.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SOP", dbOpenDynaset)
    Set ZZ_SCRATCH_P_OFFICE.Form.Recordset = tRstDummy
    gRstPeople.Close
    '

    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_OFFICE"
    cmdSQL.Execute tRecDeleted
    '
    ' get the index year information
    '
    If gUseIndexYears Then
        If IsNull(Me.TxtFromYear.Value) And IsNull(TxtToYear.Value) Then
            gUseIndexYears = False
        ElseIf IsNull(Me.TxtFromYear.Value) Then
            tStrAndYears = " AND ((ZPAD.c_index_year)<= " + Str(TxtToYear.Value) + "))"
            tStrWhereYears = " Where ((ZPAD.c_index_year)<= " + Str(TxtToYear.Value) + ")"
        ElseIf IsNull(Me.TxtToYear.Value) Then
            tStrAndYears = " AND ((ZPAD.c_index_year)>= " + Str(TxtFromYear.Value) + "))"
            tStrWhereYears = " Where ((ZPAD.c_index_year)>= " + Str(TxtFromYear.Value) + ")"
        Else
            tStrAndYears = " AND ((ZPAD.c_index_year)<= " + Str(TxtToYear.Value) + ") AND ((ZPAD.c_index_year)>= " + Str(TxtFromYear.Value) + ")"
            tStrWhereYears = " Where ((ZPAD.c_index_year)<= " + Str(TxtToYear.Value) + ") AND ((ZPAD.c_index_year)>= " + Str(TxtFromYear.Value) + ")"
        End If
    ElseIf gUseDynasties Then
        If gFromDynasty = -2 Then
            tStrAndYears = " AND ((ZPAD.c_person_dy) > 0 )"
            tStrWhereYears = " Where ((ZPAD.c_person_dy) > 0 )"
        ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
            tStrAndYears = " AND ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + "))"
            tStrWhereYears = " WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ")"
        ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
            tStrAndYears = " AND ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + "))"
            tStrWhereYears = " WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ")"
        ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
            tStrAndYears = " AND ((ZPAD.c_person_dy) = " + Str(gToDynasty) + " )"
            tStrWhereYears = " Where ((ZPAD.c_person_dy) = " + Str(gToDynasty) + " )"
        ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
            tStrAndYears = " AND ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
                "((DYNASTIES.c_start)<=" + Str(gToDynastyEnd) + "))"
            tStrWhereYears = " WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
                "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ")"
        Else
            tStrAndYears = " AND ((ZPAD.c_person_dy) > 0 )"
            tStrWhereYears = " Where ((ZPAD.c_person_dy) > 0 )"
        End If
    ElseIf gUseOfficeYears Then
        If IsNull(Me.TxtOfficeFrom.Value) And IsNull(TxtOfficeTo.Value) Then
            gUseOfficeYears = False
        ElseIf IsNull(Me.TxtOfficeFrom.Value) Then
            tStrAndYears = " AND ((ZPAD.c_lastyear)<= " + Str(TxtOfficeTo.Value) + "))"
            tStrWhereYears = " Where ((ZPAD.c_lastyear)<= " + Str(TxtOfficeTo.Value) + ")"
        ElseIf IsNull(Me.TxtOfficeTo.Value) Then
            tStrAndYears = " AND ((ZPAD.c_lastyear)>= " + Str(TxtOfficeFrom.Value) + "))"
            tStrWhereYears = " Where ((ZPAD.c_lastyear)>= " + Str(TxtOfficeFrom.Value) + ")"
        Else
            tStrAndYears = " AND ((ZPAD.c_lastyear)<= " + Str(TxtOfficeTo.Value) + ") AND ((ZPAD.c_lastyear)>= " + Str(TxtOfficeFrom.Value) + ")"
            tStrWhereYears = " Where ((ZPAD.c_lastyear)<= " + Str(TxtOfficeTo.Value) + ") AND ((ZPAD.c_lastyear)>= " + Str(TxtOfficeFrom.Value) + ")"
        End If
    End If
End Sub

```

```

        tStrAndYears = " AND ((ZPAD.c_firstyear)>= " + Str(TxtOfficeFrom.Value) + "))"
        tStrWhereYears = " Where ((ZPAD.c_firstyear)>= " + Str(TxtOfficeFrom.Value) + "))"
    Else
        tStrAndYears = " AND ((ZPAD.c_lastyear)<= " + Str(TxtOfficeTo.Value) + ") AND ((ZPAD.c_firstyear)>= " + Str(TxtOfficeFrom.Value) + "))"
        tStrWhereYears = " Where ((ZPAD.c_lastyear)<= " + Str(TxtOfficeTo.Value) + ") AND ((ZPAD.c_firstyear)>= " + Str(TxtOfficeFrom.Value) + "))"
    End If

End If

'MsgBox "About to process address"
If gUseOfficeADDRID Then
    '
    ' ZZ_SCRATCH_ADDR_OFFICE has at least one record
    '
    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_ADDR_LIST"
    cmdSQL.Execute tRecDeleted

    If ChkSubUnitsOffice.Value Then
        cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) " + _
            "SELECT DISTINCT ZZZ_BELONGS_TO.c_addr_id " + _
            "FROM ZZ_SCRATCH_ADDR_OFFICE INNER JOIN ZZZ_BELONGS_TO ON " + _
            "ZZ_SCRATCH_ADDR_OFFICE.c_addr_id = ZZZ_BELONGS_TO.c_belongs_to"
    Else
        cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT c_addr_id " + _
            "FROM ZZ_SCRATCH_ADDR_OFFICE"
    End If
    cmdSQL.Execute tRecDeleted

    '
    ' see if we need to use the historical XY search
    '
    If ChkUseXY.Value Then
        '
        ' the strategy here is to dump the IDs to ZZ_ADDRESSES then copy to ZZ_SCRATCH_ADDR_LIST
        ' (I borrow ZZ_ADDRESSES from the Pick Addresses form in order to keep the initial selection
        ' of addresses for the query intact.)
        '
        ' zap the list
        '
        tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted

        ' run the query
        '
        tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT DISTINCT ADDR_CODES.c_addr_id " + _
            "FROM ADDR_CODES, ZZ_SCRATCH_ADDR_LIST INNER JOIN ADDR_CODES AS ADDR_CODES_1 ON " + _
            "ZZ_SCRATCH_ADDR_LIST.c_addr_id = ADDR_CODES_1.c_addr_id " + _
            "WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.03) And " + _
            "(ADDR_CODES.x_coord)<=([ADDR_CODES_1].[x_coord]+0.03)) AND " + _
            "((ADDR_CODES.y_coord)>=([ADDR_CODES_1].[y_coord]-0.03) And " + _
            "(ADDR_CODES.y_coord)<=([ADDR_CODES_1].[y_coord]+0.03)))"

        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted

        ' now get the address IDs from the initial list that have no xy coordinates
        '
        tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT ZZ_SCRATCH_ADDR_LIST.c_addr_id " + _
            "FROM ZZ_SCRATCH_ADDR_LIST INNER JOIN ADDR_CODES ON " + _
            "ZZ_SCRATCH_ADDR_LIST.c_addr_id = ADDR_CODES.c_addr_id " + _
            "WHERE (((ADDR_CODES.x_coord) Is Null)) OR (((ADDR_CODES.y_coord) Is Null))"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted

        ' zap ZZ_SCRATCH_ADDR
        '
        tQueryStr = "DELETE * FROM ZZ_SCRATCH_ADDR_LIST"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted

        ' copy the list
        '
        tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT ZZ_ADDRESSES.c_addr_id " + _
            "FROM ZZ_ADDRESSES"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted

```

```

        '
        ' zap the temporary list
        '
        tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
    End If
End If
'MsgBox "Finished processing address"
'
If gUsePeopleADDRID Then
    '
    ' ZZ_SCRATCH_ADDR_OFFICE has at least one record
    '
    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_ADDR_LIST_PEOPLE"
    cmdSQL.Execute tRecDeleted

    If ChkSubUnitsPeople.Value Then
        cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_LIST_PEOPLE ( c_addr_id ) " + _
            "SELECT DISTINCT ZZZ_BELONGS_TO.c_addr_id " + _
            "FROM ZZ_SCRATCH_ADDR_PEOPLE INNER JOIN ZZZ_BELONGS_TO ON " + _
            "ZZ_SCRATCH_ADDR_PEOPLE.c_addr_id = ZZZ_BELONGS_TO.c_belongs_to"
    Else
        cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_LIST_PEOPLE ( c_addr_id ) SELECT DISTINCT c_addr_id
" + _
        "FROM ZZ_SCRATCH_ADDR_PEOPLE"
    End If
    cmdSQL.Execute tRecDeleted

    '
    ' see if we need to use the historical XY search
    '
    If ChkUseXY.Value Then
        '
        ' the strategy here is to dump the IDs to ZZ_ADDRESSES then copy to ZZ_SCRATCH_ADDR_LIST
        ' (I borrow ZZ_ADDRESSES from the Pick Addresses form in order to keep the initial selection
        ' of addresses for the query intact.)
        '
        ' zap the list
        '
        tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
        ' run the query
        '
        tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT DISTINCT ADDR_CODES.c_addr_id " + _
            "FROM ADDR_CODES, ZZ_SCRATCH_ADDR_LIST_PEOPLE INNER JOIN ADDR_CODES AS ADDR_CODES_1 ON " + _
            "ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id = ADDR_CODES_1.c_addr_id " + _
            "WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.03) And " + _
            "(ADDR_CODES.x_coord)<=([ADDR_CODES_1].[x_coord]+0.03)) AND " + _
            "((ADDR_CODES.y_coord)>=([ADDR_CODES_1].[y_coord]-0.03) And " + _
            "(ADDR_CODES.y_coord)<=([ADDR_CODES_1].[y_coord]+0.03)))"

        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
        ' now get the address IDs from the initial list that have no xy coordinates
        '
        tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id " + _
            "FROM ZZ_SCRATCH_ADDR_LIST_PEOPLE INNER JOIN ADDR_CODES ON " + _
            "ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id = ADDR_CODES.c_addr_id " + _
            "WHERE (((ADDR_CODES.x_coord) Is Null)) OR (((ADDR_CODES.y_coord) Is Null)))"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
        ' zap ZZ_SCRATCH_ADDR
        '
        tQueryStr = "DELETE * FROM ZZ_SCRATCH_ADDR_LIST_PEOPLE"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
        ' copy the list
        '
        tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR_LIST_PEOPLE ( c_addr_id ) SELECT DISTINCT ZZ_ADDRESSES.c_addr
_id " + _
        "FROM ZZ_ADDRESSES"
        cmdSQL.CommandText = tQueryStr

```



```

        cmdSQL.Execute tRecDeleted
    '
    ' zap the temporary list
    '
    tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
End If
End If
'MsgBox "Finished processing address"
'
' Define the query
'
tQueryInsertStr = "INSERT INTO ZZ_SCRATCH_OFFICE ( c_person_name, c_person_name_chn, " + _
    "c_index_year, c_female , c_sex, c_person_dy, c_person_dynasty, c_person_dy_chn, c_personid, c_posting_id
, " + _
    "c_office_id, c_office_pinyin, c_office_chn, c_office_trans, c_sequence, c_firstyear, " + _
    "c_fy_nh_code, c_fy_nh_chn, c_fy_nh_py, c_fy_nh_year, c_fy_range, c_fy_range_desc, " + _
    "c_fy_range_chn, c_lastyear, c_ly_nh_code, c_ly_nh_chn, c_ly_nh_py, c_ly_nh_year, " + _
    "c_ly_range, c_ly_range_desc, c_ly_range_chn, c_appt_code, c_appt_desc_chn, " + _
    "c_appt_desc, c_assume_office_code, c_assume_office_desc_chn, " + _
    "c_assume_office_desc, c_inst_code, c_inst_name_code, c_inst_name_hz, c_inst_name_py, " + _
    "c_source, c_title_chn, c_title, c_pages, c_notes, c_fy_intercalary, c_fy_month, " + _
    "c_ly_intercalary, c_ly_month, c_fy_day, c_ly_day, c_fy_day_gz, c_fy_day_gz_chn, " + _
    "c_fy_day_gz_py, c_ly_day_gz, c_ly_day_gz_chn, c_ly_day_gz_py, c_dy, c_dynasty, " + _
    "c_dynasty_chn, c_office_category_id, c_category_desc, c_category_desc_chn, " + _
    "c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_admin_type, c_office_addr_id, " + _
    "c_office_addr_name, c_office_addr_chn, office_x_coord, office_y_coord, c_addr_type, " + _
    "c_addr_desc, c_addr_desc_chn ) "
tQuerySelectStr = "SELECT ZPAD.c_person_name, ZPAD.c_person_name_chn, ZPAD.c_index_year, " + _
    "ZPAD.c_female , iif(ZPAD.c_female,'F','M'),ZPAD.c_person_dy,ZPAD.c_person_dynasty, ZPAD.c_person_dy_chn
, ZPAD.c_personid, " + _
    "ZPAD.c_posting_id, ZPAD.c_office_id, ZPAD.c_office_pinyin, ZPAD.c_office_chn, " + _
    "ZPAD.c_office_trans, ZPAD.c_sequence, ZPAD.c_firstyear, ZPAD.c_fy_nh_code, " + _
    "ZPAD.c_fy_nh_chn, ZPAD.c_fy_nh_py, ZPAD.c_fy_nh_year, ZPAD.c_fy_range, " + _
    "ZPAD.c_fy_range_desc, ZPAD.c_fy_range_chn, ZPAD.c_lastyear, ZPAD.c_ly_nh_code, " + _
    "ZPAD.c_ly_nh_chn, ZPAD.c_ly_nh_py, ZPAD.c_ly_nh_year, ZPAD.c_ly_range, " + _
    "ZPAD.c_ly_range_desc, ZPAD.c_ly_range_chn, ZPAD.c_appt_code, " + _
    "ZPAD.c_appt_desc_chn, ZPAD.c_appt_desc, ZPAD.c_assume_office_code, " + _
    "ZPAD.c_assume_office_desc_chn, ZPAD.c_assume_office_desc, ZPAD.c_inst_code, " + _
    "ZPAD.c_inst_name_code, ZPAD.c_inst_name_hz, ZPAD.c_inst_name_py, ZPAD.c_source, " + _
    "ZPAD.c_title_chn, ZPAD.c_title, ZPAD.c_pages, ZPAD.c_notes, ZPAD.c_fy_intercalary, " + _
    "ZPAD.c_fy_month, ZPAD.c_ly_intercalary, ZPAD.c_ly_month, ZPAD.c_fy_day, ZPAD.c_ly_day, " + _
    "ZPAD.c_fy_day_gz, ZPAD.c_fy_day_gz_chn, ZPAD.c_fy_day_gz_py, ZPAD.c_ly_day_gz, " + _
    "ZPAD.c_ly_day_gz_chn, ZPAD.c_ly_day_gz_py, ZPAD.c_dy, ZPAD.c_dynasty, " + _
    "ZPAD.c_dynasty_chn, ZPAD.c_office_category_id, ZPAD.c_category_desc, " + _
    "ZPAD.c_category_desc_chn, ZPAD.c_addr_id, ZPAD.c_addr_name, ZPAD.c_addr_chn, " + _
    "ZPAD.x_coord, ZPAD.y_coord, ZPAD.c_admin_type, ZPAD.c_office_addr_id, " + _
    "ZPAD.c_office_addr_name, ZPAD.c_office_addr_chn, ZPAD.office_x_coord, " + _
    "ZPAD.office_y_coord, ZPAD.c_addr_type, ZPAD.c_addr_desc, ZPAD.c_addr_desc_chn "

' to handle addresses, there are 4 possibilities, but because one can run the query with addresses BUT NOT o
office selected,
' all of this is doubled yet again
'
' There also are two versions of each FROM statement, with/without DYNASTIES

If Not gUsePeopleADDRID And Not gUseOfficeADDRID Then
    If TxtTypeDesc.Value = "N/A" Then
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                "FROM DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS ZPAD ON DYNASTIES.c_dy = ZPAD.c_person_
dy WHERE ( " + _
                "((ZPAD.c_office_id)= " + Str(TxtOfficeID.Value) + ") "
            Else
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                    "FROM ZZZ_POSTED_TO_ADDR_DATA AS ZPAD WHERE ( " + _
                    "((ZPAD.c_office_id)= " + Str(TxtOfficeID.Value) + ") "
                'MsgBox "Simple query: " + Str(TxtOfficeID.Value)
            End If

            If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
                tStrQuery = tStrQuery + tStrAndYears
            Else
                tStrQuery = tStrQuery + ") "
            End If
        Else
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _

```

```

"FROM (DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS ZPAD ON DYNASTIES.c_dy = ZPAD.c_person_dy) " + _
    "INNER JOIN ZZ_OFFICE_CODE ON ZPAD.c_office_id = ZZ_OFFICE_CODE.c_office_id"
Else
    tStrQuery = tQueryInsertStr + tQuerySelectStr + _
        "FROM ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_OFFICE_CODE ON " + _
        "ZPAD.c_office_id = ZZ_OFFICE_CODE.c_office_id"
End If

If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
    tStrQuery = tStrQuery + tStrWhereYears
End If
End If

ElseIf Not gUsePeopleADDRID And gUseOfficeADDRID Then
    If gUseOfficeID Then
        If TxtTypeDesc.Value = "N/A" Then
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                    "FROM ZZ_SCRATCH_ADDR_LIST INNER JOIN (DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS ZPAD " + _
                    "ON DYNASTIES.c_dy = ZPAD.c_person_dy) ON ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZPAD.c_office_addr_id " + _
                    "WHERE (((ZPAD.c_office_id)= " + Str(TxtOfficeID.Value) + "))"
            Else
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                    "FROM ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST ON ZPAD.c_office_addr_id = ZZ_SCRATCH_ADDR_LIST.c_addr_id " + _
                    "WHERE (((ZPAD.c_office_id)= " + Str(TxtOfficeID.Value) + "))"
            End If

            If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
                tStrQuery = tStrQuery + tStrAndYears
            Else
                tStrQuery = tStrQuery + ")"
            End If
        Else
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                    "FROM ZZ_SCRATCH_ADDR_LIST INNER JOIN ((DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS ZPAD ON DYNASTIES.c_dy = ZPAD.c_person_dy) " + _
                    "INNER JOIN ZZ_OFFICE_CODE ON ZPAD.c_office_id = ZZ_OFFICE_CODE.c_office_id) " + _
                    "ON ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZPAD.c_office_addr_id"
            Else
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                    "FROM (ZZ POSTED TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST ON ZPAD.c_office_addr_id = ZZ_SCRATCH_ADDR_LIST.c_addr_id) " + _
                    "INNER JOIN ZZ_OFFICE_CODE ON ZPAD.c_office_id = ZZ_OFFICE_CODE.c_office_id"
            End If

            If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
                tStrQuery = tStrQuery + tStrWhereYears
            End If
        End If
    Else
        ' If we are NOT using Office IDs, then it does not matter whether TxtTypeDesc is "N/A" or not
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                "FROM ZZ_SCRATCH_ADDR_LIST INNER JOIN (DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS ZPAD " + _
                "ON DYNASTIES.c_dy = ZPAD.c_person_dy) ON ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZPAD.c_office_addr_id "
        Else
            tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                "FROM ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST ON ZPAD.c_office_addr_id = ZZ_SCRATCH_ADDR_LIST.c_addr_id "
        End If

        If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
            tStrQuery = tStrQuery + tStrWhereYears
        Else
            ' tStrQuery = tStrQuery + ")"
        End If
    End If

ElseIf gUsePeopleADDRID And Not gUseOfficeADDRID Then
    If gUseOfficeID Then
        If TxtTypeDesc.Value = "N/A" Then
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                    "FROM ZZ_SCRATCH_ADDR_LIST PEOPLE INNER JOIN (DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS ZPAD " + _
                    "ON DYNASTIES.c_dy = ZPAD.c_person_dy) ON ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZPAD.c_office_addr_id "
            Else
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                    "FROM ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST ON ZPAD.c_office_addr_id = ZZ_SCRATCH_ADDR_LIST.c_addr_id "
            End If

            If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
                tStrQuery = tStrQuery + tStrWhereYears
            Else
                ' tStrQuery = tStrQuery + ")"
            End If
        End If
    End If
End If

```

```

                "ON DYNASTIES.c_dy = ZPAD.c_person_dy) ON ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id = ZPA
D.c_addr_id " + _
                "WHERE (((ZPAD.c_office_id)= " + Str(TxtOfficeID.Value) + ") "
            Else
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                "FROM ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST_PEOPLE ON ZPAD.c_ad
dr_id = ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id " + _
                "WHERE (((ZPAD.c_office_id)= " + Str(TxtOfficeID.Value) + ") "
            End If

            If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
                tStrQuery = tStrQuery + tStrAndYears
            Else
                tStrQuery = tStrQuery + ") "
            End If
        Else
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                "FROM ZZ_SCRATCH_ADDR_LIST_PEOPLE INNER JOIN ((DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DA
TA AS ZPAD ON DYNASTIES.c_dy = ZPAD.c_person_dy) " + _
                "INNER JOIN ZZ_OFFICE_CODE ON ZPAD.c_office_id = ZZ_OFFICE_CODE.c_office_id) " + _
                "ON ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id = ZPAD.c_addr_id"
            Else
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                "FROM (ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST_PEOPLE ON ZPAD.c_a
ddr_id = ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id) " + _
                "INNER JOIN ZZ_OFFICE_CODE ON ZPAD.c_office_id = ZZ_OFFICE_CODE.c_office_id"
            End If

            If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
                tStrQuery = tStrQuery + tStrWhereYears
            End If
        End If
    Else
        If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
            tStrQuery = tQueryInsertStr + tQuerySelectStr + _
            "FROM ZZ_SCRATCH_ADDR_LIST_PEOPLE INNER JOIN (DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS
ZPAD " + _
            "ON DYNASTIES.c_dy = ZPAD.c_person_dy) ON ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id = ZPA
D.c_addr_id "
        Else
            tStrQuery = tQueryInsertStr + tQuerySelectStr + _
            "FROM ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST_PEOPLE ON ZPAD.c_addr_i
d = ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id "
        End If

        If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
            tStrQuery = tStrQuery + tStrWhereYears
        Else
            tStrQuery = tStrQuery + ") "
        End If

        MsgBox Right(tStrQuery, 250)
    End If
Else
    ' Using both People and Office Addresses
    If gUseOfficeID Then
        If TxtTypeDesc.Value = "N/A" Then
            If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                "FROM (ZZ_SCRATCH_ADDR_LIST INNER JOIN (DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS Z
PAD " + _
                "ON DYNASTIES.c_dy = ZPAD.c_person_dy) ON ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZPAD.c_off
ice_addr_id) " + _
                "INNER JOIN ZZ_SCRATCH_ADDR_LIST_PEOPLE ON ZPAD.c_addr_id = ZZ_SCRATCH_ADDR_LIST_PEOP
LE.c_addr_id " + _
                "WHERE (((ZPAD.c_office_id) = " + Str(TxtOfficeID.Value) + ") "
            Else
                tStrQuery = tQueryInsertStr + tQuerySelectStr + _
                "FROM (ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST ON ZPAD.c_office_a
ddr_id = ZZ_SCRATCH_ADDR_LIST.c_addr_id) " + _
                "INNER JOIN ZZ_SCRATCH_ADDR_LIST_PEOPLE ON ZPAD.c_addr_id = ZZ_SCRATCH_ADDR_LIST_PEOPLE.c
_addr_id " + _
                "WHERE (((ZPAD.c_office_id) = " + Str(TxtOfficeID.Value) + ") "
            End If

            If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
                tStrQuery = tStrQuery + tStrAndYears
            Else
                tStrQuery = tStrQuery + ") "
            End If

```

```

Else
    If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
        tStrQuery = tQueryInsertStr + tQuerySelectStr + _
            "FROM ((ZZ_SCRATCH_ADDR_LIST INNER JOIN (DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS ZPAD ON DYNASTIES.c_dy = ZPAD.c_person_dy) ON ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZPAD.c_office_addr_id) " + _
            "INNER JOIN ZZ_SCRATCH_ADDR_LIST_PEOPLE ON ZPAD.c_addr_id = ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id) " + _
            "INNER JOIN ZZ_OFFICE_CODE ON ZPAD.c_office_id = ZZ_OFFICE_CODE.c_office_id"
    Else
        tStrQuery = tQueryInsertStr + tQuerySelectStr + _
            "FROM ((ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST ON ZPAD.c_office_id = ZZ_SCRATCH_ADDR_LIST.c_addr_id) " + _
            "INNER JOIN ZZ_OFFICE_CODE ON ZPAD.c_office_id = ZZ_OFFICE_CODE.c_office_id) INNER JOIN ZZ_SCRATCH_ADDR_LIST_PEOPLE ON " + _
            "ZPAD.c_addr_id = ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id"
    End If

    If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
        tStrQuery = tStrQuery + tStrWhereYears
    End If
End If

Else
    If gUseDynasties And (gFromDynasty > -1 Or gToDynasty > -1) Then
        tStrQuery = tQueryInsertStr + tQuerySelectStr + _
            "FROM (ZZ_SCRATCH_ADDR_LIST INNER JOIN (DYNASTIES INNER JOIN ZZZ_POSTED_TO_ADDR_DATA AS ZPAD ON DYNASTIES.c_dy = ZPAD.c_person_dy) " + _
            "ON ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZPAD.c_office_addr_id) INNER JOIN ZZ_SCRATCH_ADDR_LIST_PEOPLE " + _
            "ON ZPAD.c_addr_id = ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id"
    Else
        tStrQuery = tQueryInsertStr + tQuerySelectStr + _
            "FROM (ZZZ_POSTED_TO_ADDR_DATA AS ZPAD INNER JOIN ZZ_SCRATCH_ADDR_LIST ON ZPAD.c_office_addr_id = ZZ_SCRATCH_ADDR_LIST.c_addr_id) " + _
            "INNER JOIN ZZ_SCRATCH_ADDR_LIST_PEOPLE ON ZPAD.c_addr_id = ZZ_SCRATCH_ADDR_LIST_PEOPLE.c_addr_id"
    End If

    If gUseIndexYears Or gUseDynasties Or gUseOfficeYears Then
        tStrQuery = tStrQuery + tStrWhereYears
    End If
End If

End If

cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecCount
'
' Next, a simple query to add the index year description data
'
cmdSQL.CommandText = "UPDATE ZZZ_BIOG_MAIN INNER JOIN ZZ_SCRATCH_OFFICE ON ZZZ_BIOG_MAIN.c_personid = ZZ_SCRATCH_OFFICE.c_personid " + _
    "SET ZZ_SCRATCH_OFFICE.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
    "ZZ_SCRATCH_OFFICE.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
    "ZZ_SCRATCH_OFFICE.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz]"
cmdSQL.Execute tRecDeleted
'
' Now the People
'
tStrQuery = "INSERT INTO ZZ_SCRATCH_P_OFFICE ( c_personid, c_name, c_name_chn, c_index_year, " + _
    "c_index_year_type_code, c_index_year_type_desc, c_index_year_type_hz, " + _
    "c_female, c_sex, c_dy, c_dynasty, c_dynasty_chn, c_addr_id, c_addr_name, c_addr_chn, c_addr_type, " + _
    "c_addr_desc, c_addr_desc_chn, x_coord, y_coord ) " + _
    "SELECT DISTINCT ZZ_SCRATCH_OFFICE.c_personid, ZZ_SCRATCH_OFFICE.c_person_name AS c_name, " + _
    "ZZ_SCRATCH_OFFICE.c_person_name_chn AS c_name_chn, ZZ_SCRATCH_OFFICE.c_index_year, " + _
    "ZZ_SCRATCH_OFFICE.c_index_year_type_code, ZZ_SCRATCH_OFFICE.c_index_year_type_desc, ZZ_SCRATCH_OFFICE.c_index_year_type_hz, " + _
    "ZZ_SCRATCH_OFFICE.c_female, ZZ_SCRATCH_OFFICE.c_sex, ZZ_SCRATCH_OFFICE.c_person_dy AS c_dy, ZZ_SCRATCH_OFFICE.c_person_dynasty AS c_dynasty, " + _
    "ZZ_SCRATCH_OFFICE.c_person_dy_chn AS c_dynasty_chn, ZZ_SCRATCH_OFFICE.c_addr_id, " + _
    "ZZ_SCRATCH_OFFICE.c_addr_name, ZZ_SCRATCH_OFFICE.c_addr_chn, ZZ_SCRATCH_OFFICE.c_addr_type, " + _
    "ZZ_SCRATCH_OFFICE.c_addr_desc, ZZ_SCRATCH_OFFICE.c_addr_desc_chn, ZZ_SCRATCH_OFFICE.x_coord, " + _
    "ZZ_SCRATCH_OFFICE.y_coord " + _
    "FROM ZZ_SCRATCH_OFFICE"
cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecDeleted
'
' the final step is to calculate the xy_count for people and for offices
'

```

```

If tRecCount = 0 Then
    CmdGIS.Enabled = False
    CmdGISPeople.Enabled = False
    CmdStoreID.Enabled = False
    CmdNeo4j.Enabled = False
Else
    CmdStoreID.Enabled = True
    CmdNeo4j.Enabled = True
    '
    ' first the people xy
    '
    ' use three SQL calls
    '
    cmdSQL.CommandText = "DELETE * FROM tmpXY"
    cmdSQL.Execute tRecDeleted
    '
    tStrQuery = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
        "SELECT ZZ_SCRATCH_P_OFFICE.x_coord, ZZ_SCRATCH_P_OFFICE.y_coord, Count(ZZ_SCRATCH_P_OFFICE.x_coord) " + _
        "AS CountOfx_coord, Count(ZZ_SCRATCH_P_OFFICE.y_coord) AS CountOfy_coord " + _
        "FROM ZZ_SCRATCH_P_OFFICE " + _
        "GROUP BY ZZ_SCRATCH_P_OFFICE.x_coord, ZZ_SCRATCH_P_OFFICE.y_coord;"
    '
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    '
    tStrQuery = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_P_OFFICE ON (tmpXY.y_coord = " + _
        "ZZ_SCRATCH_P_OFFICE.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_P_OFFICE.x_coord) SET " + _
        "ZZ_SCRATCH_P_OFFICE.xy_count = [tmpXY].[CountOfx_coord];"
    '
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    '
    ' then the offices
    '
    cmdSQL.CommandText = "DELETE * FROM tmpXY"
    cmdSQL.Execute tRecDeleted
    '
    tStrQuery = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
        "SELECT ZZ_SCRATCH_OFFICE.office_x_coord, ZZ_SCRATCH_OFFICE.office_y_coord, " + _
        "Count(ZZ_SCRATCH_OFFICE.office_x_coord) AS CountOfx_coord, " + _
        "Count(ZZ_SCRATCH_OFFICE.office_y_coord) AS CountOfy_coord " + _
        "FROM ZZ_SCRATCH_OFFICE " + _
        "GROUP BY ZZ_SCRATCH_OFFICE.office_x_coord, ZZ_SCRATCH_OFFICE.office_y_coord"
    '
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    '
    If tRecDeleted > 0 Then
        CmdGISPeople.Enabled = True
    Else
        CmdGISPeople.Enabled = False
    End If
    '
    tStrQuery = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_OFFICE ON (tmpXY.y_coord = " + _
        "ZZ_SCRATCH_OFFICE.office_y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_OFFICE.office_x_coord) " + _
        "SET ZZ_SCRATCH_OFFICE.office_xy_count = [tmpXY].[CountOfx_coord];"
    '
    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecDeleted
    '
    If tRecDeleted > 0 Then
        CmdGIS.Enabled = True
    Else
        CmdGIS.Enabled = False
    End If
End If

Set tRstOffice = CurrentDb.OpenRecordset("ZZ_SCRATCH_OFFICE", dbOpenDynaset)
Set ZZ_SCRATCH_OFFICE.Form.Recordset = tRstOffice
'
Set gRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_OFFICE", dbOpenDynaset)
Set ZZ_SCRATCH_P_OFFICE.Form.Recordset = gRstPeople

```

Exit_CmdQuery_Click:

```

'
' close everything
'

```

```
Set tRstDummy = Nothing
Set cmdSQL = Nothing
```

```
Exit Sub
```

```
Err_CmdQuery_Click:
```

```
MsgBox Err.Description
Resume Exit_CmdQuery_Click
```

```
End Sub
```

```
Private Sub calculate_xy_count()
```

```
Dim tX As Double, tY As Double, tXY As Integer
Dim tBM As Variant, tWrite As Integer
```

```
'
' the strategy is to first throw a bookmark at the first new value
' then count the number, then go back to the bookmark and update each record
'
```

```
With gRstPeople
.Index = "xy"
.MoveFirst
```

```
tX = -1#
tY = -1#
tXY = 0
tWrite = 0
tBM = .Bookmark
```

```
Do While Not .EOF
If tX <> !x_coord Or tY <> !y_coord Then
If tWrite = 1 Then
' go back to the first record with the value
.Bookmark = tBM
Do While tX = !x_coord And tY = !y_coord
.Edit
!xy_count = tXY
.Update
.MoveNext
```

```
Loop
```

```
Else
tWrite = 1
```

```
End If
' reset
tXY = 0
tBM = .Bookmark
tX = !x_coord
tY = !y_coord
```

```
End If
' increment the count and move to the next
tXY = tXY + 1
.MoveNext
```

```
Loop
```

```
'
' the last xy value still needs to be written
'
```

```
.Bookmark = tBM
```

```
Do While Not .EOF
```

```
.Edit
!xy_count = tXY
.Update
.MoveNext
```

```
Loop
```

```
.Index = "index_year"
```

```
End With
```

```
End Sub
```

```
Private Sub calculate_office_xy_count()
```

```
Dim tX As Double, tY As Double, tXY As Integer
Dim tBM As Variant, tWrite As Integer, tRstOffice As DAO.Recordset
```

```
' the strategy is to first throw a bookmark at the first new value
' then count the number, then go back to the bookmark and update each record
'
```

```
' in order to do this, I need to open the table as a table
```

```
Set tRstOffice = CurrentDb.OpenRecordset("ZZ_SCRATCH_OFFICE", dbOpenTable)
```

```
With tRstOffice
.Index = "xy"
.MoveFirst
```

```

tX = -1#
tY = -1#
tXY = 0
tWrite = 0
tBM = .Bookmark

Do While Not .EOF
    If tX <> !office_x_coord Or tY <> !office_y_coord Then
        If tWrite = 1 Then
            ' go back to the first record with the value
            .Bookmark = tBM
            Do While tX = !office_x_coord And tY = !office_y_coord
                .Edit
                !office_xy_count = tXY
                .Update
                .MoveNext
            Loop
        Else
            tWrite = 1
        End If
        ' reset
        tXY = 0
        tBM = .Bookmark
        tX = !office_x_coord
        tY = !office_y_coord
    End If
    ' increment the count and move to the next
    tXY = tXY + 1
    .MoveNext
Loop
'
' the last xy value still needs to be written
'
.Bookmark = tBM
Do While Not .EOF
    .Edit
    !office_xy_count = tXY
    .Update
    .MoveNext
Loop
.Index = "IndexYear"
End With
End Sub

Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
Dim gStream As ADODB.Stream, tCodeStr As String
'
' If it is a KML file, call the routine and exit
'
If ChkOfficeKML.Value Then
    Call writeOfficeKML
    Exit Sub
End If
'
' the optional recordset
'
Dim tRstGIS As DAO.Recordset
'
If FrameGISOffice.Value = 1 Then
    tCodeStr = "GB18030"
Else
    tCodeStr = "UTF8"
End If
'
tC = Chr(9) ' the tab
'
' This program will dump the results to a .gis file
'
If ZZ_SCRATCH_OFFICE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGIS_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer

```

```

Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "office_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdGIS_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If

    ' we have a file name: now open the stream for writing
    '
    'MsgBox "creating stream"

    Set gStream = New ADODB.Stream
    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText

    If FrameGISOffice.Value = 1 Then
        gStream.Charset = "GB18030"
    Else
        gStream.Charset = "utf-8"
    End If

    gStream.Open
    '
    ' write the file
    '
    'SELECT ZZ_SCRATCH_OFFICE.c_name AS Name, ZZ_SCRATCH_OFFICE.c_name_chn AS NameChn,
    'ZZ_SCRATCH_OFFICE.c_index_year AS IndexYear, ZZ_SCRATCH_OFFICE.c_sex AS Sex,
    'ZZ_SCRATCH_OFFICE.c_addr_name AS AddrName, ZZ_SCRATCH_OFFICE.c_addr_chn AS AddrChn,
    'Str(ZZ_SCRATCH_OFFICE.x_coord) AS PersonX, Str(ZZ_SCRATCH_OFFICE.y_coord) AS PersonY,
    'ZZ_SCRATCH_OFFICE.c_office_trans AS Office, ZZ_SCRATCH_OFFICE.c_office_chn AS OfficeChn,
    'ZZ_SCRATCH_OFFICE.c_firstyear AS FirstYear, ZZ_SCRATCH_OFFICE.c_lastyear AS LastYear,
    'ZZ_SCRATCH_OFFICE.c_dy_desc AS Dynasty,
    'ZZ_SCRATCH_OFFICE.c_office_addr_name AS OfficeAddr,
    'ZZ_SCRATCH_OFFICE.c_office_addr_chn AS OfficeAddrChn,
    'Str(ZZ_SCRATCH_OFFICE.office_x_coord) AS X, Str(ZZ_SCRATCH_OFFICE.office_y_coord) AS Y,
    'ZZ_SCRATCH_OFFICE.office_xy_count AS XY_count
    '
    ' process the table
    '
    'DoCmd.TransferText acExportDelim, , "OFFICE_GIS_QUERY", tFileName, True
    Set tRstGIS = CurrentDb.OpenRecordset("ZZ_SCRATCH_OFFICE", dbOpenDynaset)
    '
    ' write the header
    '
    tStr = "Name" + tC + "NameChn" + tC + "IndexYear" + tC + "Sex" + tC + _
        "AddrName" + tC + "AddrChn" + tC + "PersonX" + tC + "PersonY" + tC + _
        "Office" + tC + "OfficeChn" + tC + "FirstYear" + tC + "LastYear" + tC + _
        "Dynasty" + tC + "OfficeAddr" + tC + "OfficeAddrChn" + tC + "X" + tC + "Y" + tC + "xy_count"
    gStream.WriteText tStr, adWriteLine
    '
    'MsgBox "Beginning to process query"

    With tRstGIS
        .MoveFirst
        Do While Not .EOF
            tStr = ""
            '
            If !office_x_coord > 0 Then
                'MsgBox "Name"

                If IsNull(!c_person_name) Then

```



```

        tStr = "[Name Missing]"
    Else
        tStr = !c_person_name
    End If
    '
    'MsgBox "NameChn"

    If IsNull(!c_person_name_chn) Then
        tStr = tStr + tC + "[Name Missing]"
    Else
        tStr = tStr + tC + !c_person_name_chn
    End If
    '
    '
    'MsgBox "IndexYear"

    If IsNull(!c_index_year) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + Str(!c_index_year)
    End If
    '
    '
    'MsgBox "Sex"

    If IsNull(!c_sex) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + !c_sex
    End If
    '
    '
    'MsgBox "AddrName"

    If IsNull(!c_addr_name) Then
        tStr = tStr + tC + "[Addr Name Missing]"
    Else
        tStr = tStr + tC + !c_addr_name
    End If
    '
    'MsgBox "AddrChn"

    If IsNull(!c_addr_chn) Then
        tStr = tStr + tC + "[Addr Chn Missing]"
    Else
        tStr = tStr + tC + !c_addr_chn
    End If
    '
    'MsgBox "PersonX"

    If IsNull(!x_coord) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + CStr(!x_coord)
    End If
    '
    'MsgBox "PersonY"

    If IsNull(!y_coord) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + CStr(!y_coord)
    End If
    '
    'MsgBox "Office"

    If IsNull(!c_office_trans) Then
        tStr = tStr + tC + "[Office Missing]"
    Else
        tStr = tStr + tC + !c_office_trans
    End If
    '
    'MsgBox "OfficeChn"

    If IsNull(!c_office_chn) Then
        tStr = tStr + tC + "[Office Chn Missing]"
    Else
        tStr = tStr + tC + !c_office_chn
    End If
    '

```

```

        'MsgBox "FirstYear"

        If IsNull(!c_firstyear) Then
            tStr = tStr + tC + "[ ]"
        Else
            tStr = tStr + tC + Str(!c_firstyear)
        End If
    '
    'MsgBox "LastYear"

    If IsNull(!c_lastyear) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + Str(!c_lastyear)
    End If
    '
    'MsgBox "Dynasty"

    If IsNull(!c_dynasty_chn) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + !c_dynasty_chn
    End If
    '
    'MsgBox "OfficeAddr"

    If IsNull(!c_office_addr_name) Then
        tStr = tStr + tC + "[Office Addr Missing]"
    Else
        tStr = tStr + tC + !c_office_addr_name
    End If
    '
    'MsgBox "OfficeAddrChn"

    If IsNull(!c_office_addr_chn) Then
        tStr = tStr + tC + "[Office Addr Chn Missing]"
    Else
        tStr = tStr + tC + !c_office_addr_chn
    End If
    '
    'MsgBox "X"

    If IsNull(!office_x_coord) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + CStr(!office_x_coord)
    End If
    '
    'MsgBox "Y"

    If IsNull(!office_y_coord) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + CStr(!office_y_coord)
    End If
    '
    'MsgBox "xy_count"

    If IsNull(!office_xy_count) Then
        tStr = tStr + tC + "[ ]"
    Else
        tStr = tStr + tC + Str(!office_xy_count)
    End If
    '
    If Not (tStr = "") Then
        gStream.WriteText tStr, adWriteLine
    End If
End If
.MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
End If

```

```
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing
```

```
Exit_CmdGIS_Click:
Exit Sub
```

```
Err_CmdGIS_Click:
MsgBox Err.Description
Resume Exit_CmdGIS_Click
```

```
End Sub
```

```
Private Sub CmdSaveOffices_Click()
On Error GoTo Err_CmdSaveOffices_Click
```

```
' This program will store the current list of office IDs to a .txt file
```

```
Dim tStream As ADODB.Stream, tStreamNoBOM As ADODB.Stream
Set tStream = New ADODB.Stream
```

```
tStream.Charset = "utf-8"
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
```

```
Set tStreamNoBOM = New ADODB.Stream
tStreamNoBOM.Type = adTypeBinary
tStreamNoBOM.Open
```

```
' next get a file
```

```
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstIDs As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF
```

```
Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)
```

```
dlgSaveAs.InitialFileName = "office_id_list.txt"
If dlgSaveAs.Show = -1 Then
```

```
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
tFileName = tFN
If Not tFileName = "" Then
Exit For
End If
```

```
Next
```

```
If tFileName = "" Then
MsgBox "Bad file Name."
GoTo Exit_CmdSaveOffices_Click
Else
```

```
' make sure the file name has a txt extension
If Len(tFileName) < 5 Then
tFileName = tFileName + ".txt"
ElseIf Not (LCase(Right(tFileName, 4)) = ".txt") Then
tFileName = tFileName + ".txt"
End If
```

```
End If
```

```
' write the file
```

```
' process the table
```

```
tStr = "SELECT ZZ_OFFICE_CODE.c_office_id, OFFICE_CODES.c_office_chn, OFFICE_CODES.c_office_trans " + _
"FROM ZZ_OFFICE_CODE INNER JOIN OFFICE_CODES ON ZZ_OFFICE_CODE.c_office_id = OFFICE_CODES.c_office_id
```

```
Set tRstIDs = CurrentDb.OpenRecordset(tStr, dbOpenDynaset)
```

```
tTab = Chr(9)
```

```
With tRstIDs
```

```
.MoveFirst
' MsgBox "writing file"
Do While Not .EOF
tStr = Str(!c_office_id) + tTab
```

```

        If IsNull(!c_office_trans) Then
            tStr = tStr + "" + tTab
        Else
            tStr = tStr + !c_office_trans + tTab
        End If
        tStr = tStr + !c_office_chn
    '
    tStream.WriteText tStr, adWriteLine
    .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
tStream.Position = 3
tStream.CopyTo tStreamNoBOM
' and write the stream to the file
tStreamNoBOM.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstIDs = Nothing

tStream.Close
Set tStream = Nothing
tStreamNoBOM.Close
Set tStreamNoBOM = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdSaveOffices_Click:
Exit Sub

Err_CmdSaveOffices_Click:
MsgBox Err.Description
Resume Exit_CmdSaveOffices_Click

End Sub

Private Sub CmdStoreID_Click()
    Dim cmdSQL As ADODB.Command, tRecCount As Variant

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
        ' Display message.
        If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) =
vbNo Then
            Exit Sub
        Else
            cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
            cmdSQL.Execute tRecCount
        End If
    End If

    tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT DISTINCT ZZ_SCRATCH_P_OFFICE.c_personid FROM ZZ_SCRATCH_P_OFFICE"

    cmdSQL.CommandText = tStrQuery
    cmdSQL.Execute tRecCount
    MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."
    '
    ' update storage source
    cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='Office' WHERE PersonIDSource.LineNum =1"
    cmdSQL.Execute tRecCount

End Sub

Private Sub CmdToDynasty_Click()
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strToDynasty As String

    If gToDynasty = -1 Then
        strToDynasty = ""
    Else
        strToDynasty = Str(gToDynasty)
    End If

```

```

End If

stDocName = "frmPickDynasty"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

    Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
    gToDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
    gToDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
    gToDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
    '
    ' check to see if we have a problem and reject selection if needed
    '
    If gFromDynasty > -1 Then
        If gFromDynastyBegin > gToDynastyEnd Then
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' D
            ynasty ends!", vbExclamation
            gToDynasty = -1
            TxtToDynasty.Value = ""
            TxtToDynastyPY.Value = ""
        End If
    End If
    '
    ' value is OK
    '
    If gToDynasty > -1 Then
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
        TxtToDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
        TxtToDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
    End If

    DoCmd.Close acForm, stDocName
    '
    ' reset FromDynasty if necessary (-2 = all dynasties)
    '
    If gFromDynasty = -2 Then
        gFromDynasty = -1
        TxtFromDynasty.Value = ""
        TxtFromDynastyPY.Value = ""
    End If
    '
End If

```

End Sub

```

Private Sub Form_Open(Cancel As Integer)
    Dim cmdSQL As ADODB.Command, tRecDeleted As Variant
    Dim tRstOfficeCode As DAO.Recordset, tRstDummy As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' clear the list of offices
    '
    cmdSQL.CommandText = "Delete * from ZZ_OFFICE_CODE"
    cmdSQL.Execute tRecDeleted
    '
    ' to clear the tables, briefly close and then delete records
    '
    Set tRstOfficeCode = ZZ_SCRATCH_OFFICE.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SO", dbOpenDynaset)
    Set ZZ_SCRATCH_OFFICE.Form.Recordset = tRstDummy
    tRstOfficeCode.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_OFFICE"
    cmdSQL.Execute tRecDeleted
    '
    ' now reopen
    '
    Set tRstOfficeCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_OFFICE", dbOpenDynaset)
    Set ZZ_SCRATCH_OFFICE.Form.Recordset = tRstOfficeCode

```

```

'
Set tRstOfficeCode = ZZ_SCRATCH_P_OFFICE.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SOP", dbOpenDynaset)
Set ZZ_SCRATCH_P_OFFICE.Form.Recordset = tRstDummy
tRstOfficeCode.Close
'

cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_OFFICE"
cmdSQL.Execute tRecDeleted
'

' now reopen
'

Set tRstOfficeCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_Office", dbOpenDynaset)
Set ZZ_SCRATCH_P_OFFICE.Form.Recordset = tRstOfficeCode

'ChkIndexYears.Value = True
'

' initialize state variables
gUsePeopleADDRID = False
gUseOfficeADDRID = False
gUseOfficeID = False
gUseIndexYears = False
gUseOfficeYears = False
gUseDynasties = False
gFromDynasty = -1
gToDynasty = -1

' first determine the language
Dim gLCID As MsoAppLanguageID

gLCID = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
If gLCID = 2052 Or gLCID = 3076 Then ' 2052 = PRC, 3076 = Hong Kong
    gDisplayLanguage = "S"
ElseIf gLCID = 4100 Or gLCID = 1028 Then ' 4100 = Singapore, 1028 = Taiwan
    gDisplayLanguage = "T"
    Call changeDisplayLanguage
Else
    gDisplayLanguage = "E"
    Call changeDisplayLanguage
End If

End Sub

Private Sub CmdGUESS_Click()
On Error GoTo Err_CmdGUESS_Click
'
' This program will dump the results of the search to a .gdf file
'
' for the moment I'll just describe the format of the .gdf file
'
' nodedef> name, color, label, labelvisible, style, pinyin VARCHAR(50), nodedist INT
'     name = str(c_person_id)
'     color = red (1), orange (2), yellow (3), green (4), blue (5)
'     label = c_name_chn
'     style = 4 (text inside a rectangle)
'     pinyin = c_name
'     nodedist = c_node_dist INT
'     indexyear = c_index_year INT
'     sex = c_female > (F,M)
'
' edgedef> node1, node2, color, label, labelvisible, edge_desc VARCHAR(50)
'     node1 = str(c_person_id) for node1
'     node2 = str(c_node_id) for node2
'     color = red (1), orange (2), yellow (3), green (4), blue (5)
'     label = c_link_chn
'     edge_desc = c_link_desc
'
' the central question is whether to do distance optimizations
'
' first see if there are any records to process
'

If ZZ_SCRATCH_OFFICE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGUESS_Click
End If
'

' next get a file
'

```

```

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer
Dim tColor(50) As String, tMetricSum As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "default.gdf"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdGUESS_Click
        End If
        '
        ' now process the file (second true removed to make ASCII)
        '
        Set tFileSystem = CreateObject("Scripting.FileSystemObject")
        Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)

        ' define the colors for the nodes
        '
        tColor(1) = "white"
        tColor(2) = "blue"
        tColor(3) = "green"
        tColor(4) = "yellow"
        tColor(5) = "orange"
        For ti = 6 To 50
            tColor(ti) = "red"
        Next
        '
        ' process the two tables
        '
        Set tRstEdge = ZZ_SCRATCH_OFFICE.Form.Recordset
        Set tRstNode = ZZ_SCRATCH_P_OFFICE.Form.Recordset
        tC = Chr(44) ' the comma
        '
        ' first the nodes: define the record structure
        tStr = "nodedef> name" + tC + "color" + tC + "label" + tC + "labelvisible"
        tStr = tStr + tC + "style" + tC + "pinyin VARCHAR(50)"
        tStr = tStr + tC + "indexyear INT" + tC + "sex VARCHAR(1)"
        tGDF.WriteLine (tStr)
        '
        With tRstNode
            .MoveFirst
            Do While Not .EOF
                tStr = Trim(Str(!c_person_id)) + tC
                ' name = the ID of the person
                '
                tStr = tStr + tColor(1) + tC
                '
                If IsNull(!c_name_chn) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If
                ' label
                tStr = tStr + "true" + tC + "4" + tC
                ' labelvisible = true, style = 4 (text inside a rectangle)
                If IsNull(!c_name) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
                ' pinyin = c_name
                '
            Loop
        End With
    End If
End With

```

```

        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + tC
        End If
        ' indexyear = c_index_year INT
        If Not IsNull(!c_sex) Then
            tStr = tStr + !c_sex
        End If
        tGDF.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the edges: define the record structure
tStr = "edgedef> node1" + tC + "node2" + tC + "color" + tC + "label"
tGDF.WriteLine (tStr)

With tRstEdge
    .MoveFirst
    Do While Not .EOF
        tStr = Trim(Str(!c_person_id)) + tC
        ' node1 = str(c_person_id) for node1
        tStr = tStr + Trim(Str(!c_office_id)) + tC
        ' node2 = str(c_node_id) for node2
        ,
        tStr = tStr + tColor(1) + tC
        ' color = white (1), blue (2), green (3), yellow (4), orange (5)
        ,
        If IsNull(!c_office_desc) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_office_desc
        End If
        ' label = the Officeiation
        ,
        tGDF.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
tGDF.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tGDF = Nothing
Set tFileSystem = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGUESS_Click:
Exit Sub

Err_CmdGUESS_Click:
MsgBox Err.Description
Resume Exit_CmdGUESS_Click

End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

If gDisplayLanguage = "T" Then
    gDisplayLanguage = "E"
Else
    gDisplayLanguage = "T"
End If

Call changeDisplayLanguage

Exit_CmdFanti_Click:
Exit Sub

Err_CmdFanti_Click:

```



```

    MsgBox Err.Description
    Resume Exit_CmdFanti_Click
End Sub

Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

    If gDisplayLanguage = "S" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "S"
    End If

    Call changeDisplayLanguage

Exit_CmdJianti_Click:
    Exit Sub

Err_CmdJianti_Click:
    MsgBox Err.Description
    Resume Exit_CmdJianti_Click
End Sub

Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 40) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 40 And Not .EOF
            If !c_form = "LAO" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then
            tLang = 2
        Else
            tLang = 3
        End If
        '
        ' now comes the basic routine
        '
        Me.LblFrom.Caption = tLabelLanguage(tLang, 1)
        Me.LblTo.Caption = tLabelLanguage(tLang, 2)
        Me.LblType.Caption = tLabelLanguage(tLang, 3)
        Me.CmdPickOffice.Caption = tLabelLanguage(tLang, 4)
        Me.CmdQuery.Caption = tLabelLanguage(tLang, 5)
        Me.CmdGIS.Caption = tLabelLanguage(tLang, 6)
        Me.CmdGISPeople.Caption = tLabelLanguage(tLang, 7)
        Me.CmdFanti.Caption = tLabelLanguage(tLang, 8)
        Me.CmdJianti.Caption = tLabelLanguage(tLang, 9)
        Me.PageOffice.Caption = tLabelLanguage(tLang, 10)
        Me.PagePeople.Caption = tLabelLanguage(tLang, 11)
        'Me.LblIndexYears.Caption = tLabelLanguage(tLang, 12)
    End If

```

```

Me.LblDisplay.Caption = tLabelLanguage(tLang, 13)
Me.CmdHelp.Caption = tLabelLanguage(tLang, 14)
Me.CmdAllOffices.Caption = tLabelLanguage(tLang, 15)
Me.CmdAllPlacesOffices.Caption = tLabelLanguage(tLang, 16)
Me.CmdAllPlacesPeople.Caption = tLabelLanguage(tLang, 17)
Me.CmdImportPlaceOffice.Caption = tLabelLanguage(tLang, 18)
Me.CmdImportPlacePeople.Caption = tLabelLanguage(tLang, 19)
Me.CmdPlaceOffice.Caption = tLabelLanguage(tLang, 20)
Me.CmdPlacePeople.Caption = tLabelLanguage(tLang, 21)
Me.LblChkUseXY.Caption = tLabelLanguage(tLang, 22)
Me.LblPlaceOffice.Caption = tLabelLanguage(tLang, 23)
Me.LblPlacePeople.Caption = tLabelLanguage(tLang, 24)
Me.CmdStoreID.Caption = tLabelLanguage(tLang, 25)
Me.LblSubUnitsOffice.Caption = tLabelLanguage(tLang, 26)
Me.LblSubUnitsPeople.Caption = tLabelLanguage(tLang, 27)
Me.LblDynasties.Caption = tLabelLanguage(tLang, 28)
Me.CmdFromDynasty.Caption = tLabelLanguage(tLang, 29)
Me.CmdToDynasty.Caption = tLabelLanguage(tLang, 30)
Me.CmdAllDynasties.Caption = tLabelLanguage(tLang, 31)
Me.LblIndexYears.Caption = tLabelLanguage(tLang, 32)
Me.LblOptNoDates.Caption = tLabelLanguage(tLang, 33)
Me.LblOptIndexYears.Caption = tLabelLanguage(tLang, 34)
Me.LblOptDynasties.Caption = tLabelLanguage(tLang, 35)
Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 36)
Me.CmdImportOffices.Caption = tLabelLanguage(tLang, 37)
Me.CmdSaveOffices.Caption = tLabelLanguage(tLang, 38)
Me.LblOptOffice.Caption = tLabelLanguage(tLang, 39)
End If

```

End Sub

Private Sub writeOfficeKML()

```

Dim tStrKML As String
'
'   This program will dump the results to a .kml file
'
If ZZ_SCRATCH_OFFICE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_writeOfficeKML
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

If FrameGISOffice.Value = 1 Then
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
Else
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
'   next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "office_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_writeOfficeKML
    End If

```

```

Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".kml"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
        tFileName = tFileName + ".kml"
    End If
End If
'
' write the file
'
'SELECT ZZ_SCRATCH_OFFICE.c_name AS Name, ZZ_SCRATCH_OFFICE.c_name_chn AS NameChn,
'ZZ_SCRATCH_OFFICE.c_index_year AS IndexYear, ZZ_SCRATCH_OFFICE.c_sex AS Sex,
'ZZ_SCRATCH_OFFICE.c_addr_name AS AddrName, ZZ_SCRATCH_OFFICE.c_addr_chn AS AddrChn,
'Str(ZZ_SCRATCH_OFFICE.x_coord) AS PersonX, Str(ZZ_SCRATCH_OFFICE.y_coord) AS PersonY,
'ZZ_SCRATCH_OFFICE.c_office_trans AS Office, ZZ_SCRATCH_OFFICE.c_office_chn AS OfficeChn,
'ZZ_SCRATCH_OFFICE.c_firstyear AS FirstYear, ZZ_SCRATCH_OFFICE.c_lastyear AS LastYear,
'ZZ_SCRATCH_OFFICE.c_dy_desc AS Dynasty,
'ZZ_SCRATCH_OFFICE.c_office_addr_name AS OfficeAddr,
'ZZ_SCRATCH_OFFICE.c_office_addr_chn AS OfficeAddrChn,
'Str(ZZ_SCRATCH_OFFICE.office_x_coord) AS X, Str(ZZ_SCRATCH_OFFICE.office_y_coord) AS Y,
'ZZ_SCRATCH_OFFICE.office_xy_count AS XY_count

'
' tStr = "PostingID" (c_posting_id) + "Office" (c_name) + "OfficeChn" (c_name_chn) + _
' "FirstYear" (c_firstyear) + "LastYear" + (c_lastyear) + _
' "Dynasty" (c_dy) + "OfficeAddr" (c_office_addr_name) + "OfficeAddrHZ" (c_office_addr_chn) + _
' "X" (office_x_coord) + "Y" (office_y_coord) + "xy_count" (office_xy_count)
'
' process the table
'
Set tRstNode = ZZ_SCRATCH_OFFICE.Form.Recordset
tC = Chr(9) ' the tab
tDQ = Chr(34) ' the double quotation mark
'
' write the header
'
tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
tStream.WriteText "<Document>", adWriteLine
tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
tStream.WriteText tC + "<open>1</open>", adWriteLine
tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

tStream.WriteText tC + "<Style id=" + tDQ + "office-balloon-template" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
tStream.WriteText tC + tC + tC + "<text>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "<![CDATA["", adWriteLine
tStream.WriteText tC + tC + tC + tC + "ID: ${OfficePosting/PostingID} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Person Chn: ${OfficePosting/PersonNameHZ} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Begin Year: ${OfficePosting/BeginYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "End Year: ${OfficePosting/EndYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Office Desc: ${OfficePosting/OfficeName} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Office Chn: ${OfficePosting/OfficeNameHZ} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Office Dynasty: ${OfficePosting/OfficeDyn} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Address: ${OfficePosting/AddrName} ${OfficePosting/AddrNameHZ} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "XY Count: ${OfficePosting/XYCount} <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine
tStream.WriteText tC + "<!-- Declare the type " + tDQ + "OfficePosting" + tDQ + " with 6 fields -->", adWriteLine

tStream.WriteText tC + "<Schema name=" + tDQ + "OfficePosting" + tDQ + " id=" + tDQ + "OfficePostingId" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "PersonNameHZ" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person Chn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrNameHZ" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PostingID" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + tC + "<displayName><![CDATA[ID]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine

```

```

        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "FirstYear" + tDQ
+ ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Begin Year]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "LastYear" + tDQ
+ ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[End Year]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "OfficeName" +
tDQ + ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Office Name]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "OfficeNameHZ"
+ tDQ + ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Office Chn]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "OfficeDyn" + tDQ
+ ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Office Dyn]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + "</Schema>", adWriteLine

With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs, even where there should not be any
        '
        ' write the point header
        '
        tStream.WriteText tC + "<Placemark>", adWriteLine

        If IsNull(!c_person_name) Then
            tStr = "[Bad Data] " + Str(!c_posting_id)
        Else
            tStr = !c_person_name + " " + Str(!c_posting_id)
        End If
        tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

        tStream.WriteText tC + tC + "<styleUrl>#office-balloon-template</styleUrl>", adWriteLine
        '
        ' First Year as time stamp
        '
        If IsNull(!c_firstyear) Then
            tStr = "0"
        Else
            tStr = Str(!c_firstyear)
        End If
        tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
        '
        tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#OfficePostingID" + tDQ + ">",
adWriteLine
        '
        ' posting ID
        '
        tStr = Str(!c_posting_id)
        tStream.WriteText tC + tC + tC + "<SimpleData name=" + tDQ + "PostingID" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
        '
        ' Person Name Chn
        '
        If IsNull(!c_person_name_chn) Then
            tStr = tStr + "[Bad Data]"
        Else
            If Trim(!c_person_name_chn) = "" Then
                tStr = "[?]"
            Else
                tStr = !c_person_name_chn
            End If
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonNameHZ" + tDQ + ">" + tS
tr + "</SimpleData>", adWriteLine
        '
        ' Office Name
        '

```

```

        If IsNull(!c_office_trans) Then
            tStr = tStr + "[Bad Data]"
        Else
            If Trim(!c_office_trans) = "" Then
                tStr = "[?]"
            Else
                tStr = !c_office_trans
            End If
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "OfficeName" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
        '
        ' Office Chinese Name
        '
        If IsNull(!c_office_chn) Then
            tStr = tStr + "[Bad Data]"
        Else
            If Trim(!c_office_chn) = "" Then
                tStr = "[?]"
            Else
                tStr = !c_office_chn
            End If
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "OfficeNameHZ" + tDQ + ">" + tS
tr + "</SimpleData>", adWriteLine
        '
        ' First Year
        '
        If IsNull(!c_firstyear) Then
            tStr = "0"
        Else
            tStr = Str(!c_firstyear)
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "FirstYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
        '
        ' Last Year
        '
        If IsNull(!c_lastyear) Then
            tStr = "0"
        Else
            tStr = Str(!c_lastyear)
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "LastYear" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
        '
        ' Office Dynasty
        '
        If IsNull(!c_dy) Then
            tStr = "0"
        Else
            tStr = Str(!c_dy)
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "OfficeDyn" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
        '
        ' Address Name
        '
        If IsNull(!c_office_addr_name) Then
            tStr = "[?]"
        ElseIf Trim(!c_office_addr_name) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_office_addr_name
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
        '
        ' Address Name Chinese
        '
        If IsNull(!c_office_addr_chn) Then
            tStr = "[?]"
        ElseIf Trim(!c_office_addr_chn) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_office_addr_chn
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrNameHZ" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
        '

```

```

        ' XY Count
        '
        If IsNull(!office_xy_count) Then
            tStr = "0"
        Else
            tStr = Str(!office_xy_count)
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
        '
        tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
        tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + "<Point>", adWriteLine
        '
        ' coordinates
        If IsNull(!office_x_coord) Then
            tStr = "0"
        Else
            tStr = Str(!office_x_coord)
        End If

        If IsNull(!office_y_coord) Then
            tStr = tStr + ",0"
        Else
            tStr = tStr + "," + Str(!office_y_coord)
        End If
        tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
        '
        ' footer
        '
        tStream.WriteText tC + tC + "</Point>", adWriteLine
        tStream.WriteText tC + "</Placemark>", adWriteLine
        .MoveNext
    Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

```

```

Exit_writeOfficeKML:
Exit Sub

```

```

Err_writeOfficeKML:
MsgBox Err.Description
Resume Exit_writeOfficeKML

```

```
End Sub
```

```
Private Sub writePersonKML()
```

```

    Dim tStrKML As String
    '
    ' This program will dump the results to a .kml file
    '
    If ZZ_SCRATCH_P_OFFICE.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_writePersonKML
    End If
    '
    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream

```

```

If FrameGISOffice.Value = 1 Then
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
Else
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "office_people_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_writePersonKML
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".kml"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
            tFileName = tFileName + ".kml"
        End If
    End If
    '
    ' write the file
    '
    'SELECT ZZ_SCRATCH_OFFICE.c_name AS Name, ZZ_SCRATCH_OFFICE.c_name_chn AS NameChn,
    'ZZ_SCRATCH_OFFICE.c_index_year AS IndexYear, ZZ_SCRATCH_OFFICE.c_sex AS Sex,
    'ZZ_SCRATCH_OFFICE.c_addr_name AS AddrName, ZZ_SCRATCH_OFFICE.c_addr_chn AS AddrChn,
    'Str(ZZ_SCRATCH_OFFICE.x_coord) AS PersonX, Str(ZZ_SCRATCH_OFFICE.y_coord) AS PersonY,
    'ZZ_SCRATCH_OFFICE.c_office_trans AS Office, ZZ_SCRATCH_OFFICE.c_office_chn AS OfficeChn,
    'ZZ_SCRATCH_OFFICE.c_firstyear AS FirstYear, ZZ_SCRATCH_OFFICE.c_lastyear AS LastYear,
    'ZZ_SCRATCH_OFFICE.c_dy_desc AS Dynasty,
    'ZZ_SCRATCH_OFFICE.c_office_addr_name AS OfficeAddr,
    'ZZ_SCRATCH_OFFICE.c_office_addr_chn AS OfficeAddrChn,
    'Str(ZZ_SCRATCH_OFFICE.office_x_coord) AS X, Str(ZZ_SCRATCH_OFFICE.office_y_coord) AS Y,
    'ZZ_SCRATCH_OFFICE.office_xy_count AS XY_count

    '
    tStr = "PostingID" (c_posting_id) + "Office" (c_name) + "OfficeChn" (c_name_chn) + _
        "FirstYear" (c_firstyear) + "LastYear" + (c_lastyear) + _
        "Dynasty" (c_dy) + "OfficeAddr" (c_office_addr_name) + "OfficeAddrHZ" (c_office_addr_chn) + _
        "X" (office_x_coord) + "Y" (office_y_coord) + "xy_count" (office_xy_count)
    '
    ' process the table
    '
    Set tRstNode = ZZ_SCRATCH_P_OFFICE.Form.Recordset
    tC = Chr(9) ' the tab
    tDQ = Chr(34) ' the double quotation mark
    '
    ' write the header
    '
    tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
    tStream.WriteText "<Document>", adWriteLine
    tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
    tStream.WriteText tC + "<open>1</open>", adWriteLine
    tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLi
ne
    tStream.WriteText tC + "<Style id=" + tDQ + "office-balloon-template" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
    tStream.WriteText tC + tC + tC + "<text>", adWriteLine

```

```

tStream.WriteText tC + tC + tC + tC + "<![CDATA["", adWriteLine
tStream.WriteText tC + tC + tC + tC + "ID: ${OfficePosting/PersonID} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Name Chn: ${OfficePosting/NameHZ} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Index Year: ${OfficePosting/IndexYear} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Dynasty: ${OfficePosting/Dyn} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "Address: ${OfficePosting/AddrName} ${OfficePosting/AddrNameHZ} <br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "XY Count: ${OfficePosting/XYCount} <br/><br/>", adWriteLine
tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
tStream.WriteText tC + tC + tC + "</text>", adWriteLine
tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
tStream.WriteText tC + "</Style>", adWriteLine
tStream.WriteText tC + "<!-- Declare the type " + tDQ + "OfficePosting" + tDQ + " with 6 fields -->", adWriteLine
tStream.WriteText tC + "<Schema name=" + tDQ + "OfficePosting" + tDQ + " id=" + tDQ + "OfficePersonID" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<displayName><![CDATA[ID]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "NameHZ" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<displayName><![CDATA[Person Chn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrNameHZ" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "Dyn" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<displayName><![CDATA[Dyn]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ + ">", adWriteLine
tStream.WriteText tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
tStream.WriteText tC + "</Schema>", adWriteLine

```

With tRstNode

,

.MoveFirst

Do While Not .EOF

 ' must guard against NULLs, even where there should not be any

 ,

 ' write the point header

 ,

tStream.WriteText tC + "<Placemark>", adWriteLine

If IsNull(!c_name) Then

 tStr = "[Bad Data] " + Str(!c_personid)

Else

 tStr = !c_name + " " + Str(!c_personid)

End If

tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

tStream.WriteText tC + tC + "<styleUrl>#office-balloon-template</styleUrl>", adWriteLine

 ,

 ' First Year as time stamp

 ,

If IsNull(!c_index_year) Then

 tStr = "N/A"

Else

 tStr = Str(!c_index_year)

End If

tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine

 ,

tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine

tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#OfficePersonID" + tDQ + ">",

adWriteLine

 ,

 ' person ID

 ,


```

        tStr = Str(!c_personid)
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    '   Person Name Chn
    '
    If IsNull(!c_name_chn) Then
        tStr = tStr + "[Bad Data]"
    Else
        If Trim(!c_name_chn) = "" Then
            tStr = "[?]"
        Else
            tStr = !c_name_chn
        End If
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NameHZ" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
    '
    '   Index Year
    '
    If IsNull(!c_index_year) Then
        tStr = "N/A"
    Else
        tStr = Str(!c_index_year)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    '   Dynasty
    '
    If IsNull(!c_dy) Then
        tStr = "0"
    Else
        tStr = Str(!c_dy)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "Dyn" + tDQ + ">" + tStr + "</S
impleData>", adWriteLine
    '
    '   Address Name
    '
    If IsNull(!c_addr_name) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_name) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_name
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    '   Address Name Chinese
    '
    If IsNull(!c_addr_chn) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_chn
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrNameHZ" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    '   XY Count
    '
    If IsNull(!xy_count) Then
        tStr = "0"
    Else
        tStr = Str(!xy_count)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
    tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
    tStream.WriteText tC + tC + "<Point>", adWriteLine
    '
    '   coordinates
    '
    If IsNull(!x_coord) Then
        tStr = "0"
    Else

```

```

        tStr = Str(!x_coord)
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + ",0"
    Else
        tStr = tStr + "," + Str(!y_coord)
    End If
    tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
    ' footer
    '
    tStream.WriteText tC + tC + "</Point>", adWriteLine
    tStream.WriteText tC + "</Placemark>", adWriteLine
    .MoveNext
Loop
End With
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_writePersonKML:
Exit Sub

Err_writePersonKML:
MsgBox Err.Description
Resume Exit_writePersonKML

End Sub

Private Sub FrameFilterYears_Click()
    ' the simplest approach is to turn it all off and then turn on the appropriate objects

    ' disable all
    Me.CmdFromDynasty.Enabled = False
    Me.CmdToDynasty.Enabled = False
    Me.CmdAllDynasties.Enabled = False
    Me.TxtFromDynasty.Enabled = False
    Me.TxtFromDynastyPY.Enabled = False
    Me.TxtToDynasty.Enabled = False
    Me.TxtToDynastyPY.Enabled = False
    Me.TxtFromDynasty.Locked = False
    Me.TxtFromDynastyPY.Locked = False
    Me.TxtToDynasty.Locked = False
    Me.TxtToDynastyPY.Locked = False

    Me.TxtFromYear.Enabled = False
    Me.TxtToYear.Enabled = False
    Me.TxtOfficeFrom.Enabled = False
    Me.TxtOfficeTo.Enabled = False

    gUseIndexYears = False
    gUseOfficeYears = False
    gUseDynasties = False

    If FrameFilterYears.Value = 2 Then
        ' enable index years
        Me.TxtFromYear.Enabled = True
        Me.TxtToYear.Enabled = True
        gUseIndexYears = True
    
```

```
ElseIf FrameFilterYears.Value = 3 Then
```

```
    ' enable dynasties  
    Me.CmdFromDynasty.Enabled = True  
    Me.CmdToDynasty.Enabled = True  
    Me.CmdAllDynasties.Enabled = True  
    Me.TxtFromDynasty.Enabled = True  
    Me.TxtFromDynastyPY.Enabled = True  
    Me.TxtToDynasty.Enabled = True  
    Me.TxtToDynastyPY.Enabled = True  
    Me.TxtFromDynasty.Locked = True  
    Me.TxtFromDynastyPY.Locked = True  
    Me.TxtToDynasty.Locked = True  
    Me.TxtToDynastyPY.Locked = True  
    gUseDynasties = True
```

```
ElseIf FrameFilterYears.Value = 4 Then
```

```
    ' enable office years  
    Me.TxtOfficeFrom.Enabled = True  
    Me.TxtOfficeTo.Enabled = True  
    gUseOfficeYears = True
```

```
End If
```

```
End Sub
```

```
Option Compare Database
```

```
Public gRstPeople As DAO.Recordset, gDisplayLanguage As String, gLabelsOK As Boolean
```

```
Public gImportPlaces As Boolean, gUseADDRID As Boolean, gFilterBAC As Boolean
```

```
Public gFromDynasty As Integer, gToDynasty As Integer, gUseIndexYears As Boolean, gUseDynasties As Boolean, _  
gFromDynastyBegin As Integer, gFromDynastyEnd As Integer, gToDynastyBegin As Integer, gToDynastyEnd As In  
teger
```

```
Private Sub ChkAssocPerson_Click()
```

```
Call QueryOK
```

```
End Sub
```

```
Private Sub ChkAssocPlace_Click()
```

```
Call QueryOK
```

```
End Sub
```

```
Private Sub ChkEntry_Click()
```

```
Call QueryOK
```

```
End Sub
```

```
Private Sub ChkIndexYears_Click()
```

```
Me.TxtFromYear.Enabled = ChkIndexYears.Value
```

```
Me.TxtToYear.Enabled = ChkIndexYears.Value
```

```
End Sub
```

```
Private Sub ChkIndividual_Click()
```

```
Call QueryOK
```

```
If ChkIndividual.Value Then
```

```
CmdPickBAC.Enabled = True
```

```
Else
```

```
CmdPickBAC.Enabled = False
```

```
End If
```

```
End Sub
```

```
Private Sub ChkInstitution_Click()
```

```
Call QueryOK
```

```
End Sub
```

```
Private Sub ChkKin_Click()
```

```
Call QueryOK
```

```
End Sub
```

```
Private Sub ChkOffice_Click()
```

```
Call QueryOK
```

```
End Sub
```

```
Private Sub CmdAllDynasties_Click()
```

```
gFromDynasty = -2
```

```
gToDynasty = -2
```

```
TxtFromDynasty.Value = ""
```

```
TxtFromDynastyPY.Value = "All"
```

```
TxtToDynasty.Value = ""
```

```
TxtToDynastyPY.Value = "All"
```

```
End Sub
```

```
Private Sub CmdFromDynasty_Click()
```

```
Dim stDocName As String
```

```
Dim stLinkCriteria As String
```

```
Dim strFromDynasty As String
```

```
If gFromDynasty < 0 Then
```

```
strFromDynasty = ""
```

```
Else
```

```
strFromDynasty = Str(gFromDynasty)
```

```
End If
```

```
stDocName = "frmPickDynasty"
```

```
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty
```

```
If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then
```

```
Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
```

```
gFromDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value
```

```
Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
```

```
gFromDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value
```

```
Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
```

```
gFromDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
```

```
,
```

```

' check to see if we have a problem and reject selection
'
If gToDynasty > -1 Then
    If gFromDynastyBegin > gToDynastyEnd Then
        MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dyn
ynasty ends!", vbExclamation
        gFromDynasty = -1
        TxtFromDynasty.Value = ""
        TxtFromDynastyPY.Value = ""
    End If
End If
'
' value is OK
'
If gFromDynasty > -1 Then
    Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
    TxtFromDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
    TxtFromDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
End If

DoCmd.Close acForm, stDocName
'
' reset ToDynasty if necessary (-2 = all dynasties)
'
If gToDynasty = -2 Then
    gToDynasty = -1
    TxtToDynasty.Value = ""
    TxtToDynastyPY.Value = ""
End If
'
End If

End Sub

Private Sub CmdImportPlaces_Click()
    On Error GoTo Err_CmdImportPlaces_Click

    Dim stDocName As String, tRstAddresses As DAO.Recordset
    Dim stLinkCriteria As String, tRstImportPlaces As DAO.Recordset
    Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String
    Dim tLen As Integer, cmdSQL As ADODB.Command, tQuit As Boolean

    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tFileSystem, tList

    '
    tQuit = False
    If Not tQuit Then
        ' open the list

        Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

        'Use a With...End With block to reference the FileDialog object.
        With dlgSaveAs
            .InitialFileName = ""
            If .Show = -1 Then
                '
                tFileName = ""
                For Each tFN In .SelectedItems
                    tFileName = tFN
                    If Not tFileName = "" Then
                        Exit For
                    End If
                Next
                If tFileName = "" Then
                    MsgBox "Bad file Name."
                    GoTo Exit_CmdImportPlaces_Click
                End If
            End If
        End With
        '
        ' Clear the address table now that we are ready to go
        '
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText

```

```

'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "ImportPlaceList_Space", "TempImportList", tFileName, 0
'   TransferType=acImportDelim
'   SpecificationName = "TempImportList" (apparently it is saved in the database itself)
'   TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
'   HasFieldNames = False (0)
'
'   copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
"FROM ADDR_CODES RIGHT JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID " + _
"WHERE ((ADDR_CODES.c_addr_id) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported:  please look at InputErrorList."
End If
'
'   copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT TempImportList.ImportID " + _
"FROM ADDR_CODES INNER JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    Me.TxtPlace.Value = "[Imported List]"
    Me.TxtPlaceChn.Value = "[Imported List]"
    Me.CmdQuery.Enabled = True
    gUseADDRID = True
    ChkXYRef.Enabled = True
    ChkSubUnits.Enabled = True
End If

Set cmdSQL = Nothing
Set tFileSystem = Nothing
End If

Exit_CmdImportPlaces_Click:
Exit Sub

Err_CmdImportPlaces_Click:
MsgBox Err.Description
Resume Exit_CmdImportPlaces_Click

End Sub

Private Sub CmdNeo4j_Click()
On Error GoTo Err_CmdNeo4j_Click
'
'   This program will dump the results of the search to five CSV files
'
'   warn the user that a lot of files will be created
'
MsgBox "Neo4j requires that from 6 to 9 files be created."
'
'   allocate the file variables
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
'
'   next get the People file
'
Dim tRstPeople As DAO.Recordset, tRstPlace As DAO.Recordset
Dim tRstPeoplePlace As DAO.Recordset, tStr As String, tC As String
Dim tQueryStr As String, tPersonID As Long, tRecDeleted As Long
Dim gStream As ADODB.Stream, tCodeStr As String
'

```

```

' set up the stream to write to
'
Set gStream = New ADODB.Stream
'
If CodeFrame.Value = 1 Then
    gStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf CodeFrame.Value = 2 Then
    gStream.Charset = "big5"
    tCodeStr = "BIG5"
ElseIf CodeFrame.Value = 3 Then
    gStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    gStream.Charset = "ascii"
    tCodeStr = "ascii"
End If
'
tC = Chr(44) ' the comma
'
' prepare the temp tables for the people, place, peoplePlace and entry data

Dim cmdSQL As ADODB.Command
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
Set tRstPeopleStatus = CurrentDb.OpenRecordset("ZZ_SCRATCH_STATUS", dbOpenDynaset)

' Open the People file

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    ' we have a file name: now open the stream for writing

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open
    '
    ' get the list of people: there are two sources for people -- c_person_id and c_assoc_id
    '
    cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_P_TEXT"
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id, c_name, c_name_chn, c_index_year ) " + _
        "SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_name, ZZ_PLACE.c_name_chn, ZZ_PLACE.c_index_
year " + _
        "FROM ZZ_PLACE"
    '
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id, c_name, c_name_chn, c_index_year ) " + _
        "SELECT DISTINCT ZZ_PLACE.c_assoc_id, ZZ_PLACE.c_assoc_name, ZZ_PLACE.c_assoc_chn, ZZ_PLACE.c_
assoc_index_year " + _
        "FROM ZZ_PLACE " + _

```

```

        "WHERE (ZZ_PLACE.c_assoc_id > 0)"
    ,
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZ_SCRATCH_P_TEXT.c_name, ZZ_SCRATCH_P_TEXT.c_
_name_chn, " + _
        "ZZ_SCRATCH_P_TEXT.c_index_year, ZZZ_BIOG_MAIN.c_dynasty, ZZZ_BIOG_MAIN.c_dynasty_chn, ZZ
Z_BIOG_MAIN.c_female " + _
        "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_BIOG_
MAIN.c_personid"

    Set tRstPeople = CurrentDb.OpenRecordset(tQueryStr, dbOpenDynaset)
    tRstPeople.MoveLast
    ,
    ' process the four tables
    ,
    ' first the nodes:  define the record structure
    ,
    ' if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a pin
yin field
    If tCodeStr = "ascii" Then
        tStr = "NameID" + tC + "NamePY" + tC + "IndexYear" + tC + "Dynasty" + tC + "Sex"
    Else
        tStr = "NameID" + tC + "NameHZ" + tC + "NamePY" + tC + "IndexYear" + tC + "Dynasty" + tC + "Sex"
    End If
    gStream.WriteText tStr, adWriteLine
    ,
    With tRstPeople
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_person_id)) + tC
            ,
            ' name
            ,
            If tCodeStr = "ascii" Then
                If IsNull(!c_name) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
            Else
                If IsNull(!c_name_chn) Then
                    tStr = tStr + "Missing" + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If

                If IsNull(!c_name) Then
                    tStr = tStr + "Missing" + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
            End If
            ,
            ' indexyear = c_index_year INT
            ,
            If IsNull(!c_index_year) Then
                tStr = tStr + "-2000" + tC
            Else
                tStr = tStr + Trim(Str(!c_index_year)) + tC
            End If

            ' dynasty information
            ,
            If IsNull(!c_dynasty) Then
                tStr = tStr + "unknown" + tC
            Else
                If tCodeStr = "ascii" Then
                    tStr = tStr + !c_dynasty + tC
                Else
                    tStr = tStr + !c_dynasty_chn + tC
                End If
            End If
            ,
            If IsNull(!c_female) Then
                tStr = tStr + "Missing"
            Else
                tStr = tStr + IIf(!c_female, "F", "M")
            End If
        End While
    End With

```



```

        End If
    ,
    gStream.WriteText tStr, adWriteLine
    ,
    .MoveNext
Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
,
' now the PeopleIndexAddr file
,
dlgSaveAs.InitialFileName = "PeopleIndexAddr_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZZ_BIOG_MAIN.c_index_addr_id, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_code " + _
        "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_BIOG_
MAIN.c_personid " + _
        "WHERE (ZZZ_BIOG_MAIN.c_index_addr_type_code > 0)"

    Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

    tStr = "NameID" + tC + "PlaceID" + tC + "PersonPlaceCode"

    gStream.WriteText tStr, adWriteLine

    With tRstPeoplePlace
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_index_addr_id) Then
                ,
                tStr = Trim(Str(!c_person_id)) + tC
                ,
                tStr = tStr + Trim(Str(!c_index_addr_id)) + tC
                ,
                tStr = tStr + Trim(Str(!c_index_addr_type_code))
                ,
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    ,
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close
Else

```

```

        'The user pressed Cancel.
        GoTo Exit_CmdNeo4j_Click
End If
'
' now places
'
' get a file name
'
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    ' There are two sources of places: people's index addresses and the c_addr_id in ZZ_PLACE
    '
    cmdSQL.CommandText = "DELETE * FROM ZZ_ADDRESSES"
    cmdSQL.Execute tRecDeleted
    '
    'MsgBox "About to collect first addresses"
    '
    tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_BIOG_MAIN.c_index_addr_name, ZZZ_BIOG_MAI
N.c_index_addr_chn, " + _
        "ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord " + _
        "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_BIOG_
MAIN.c_personid " + _
        "WHERE (((ZZZ_BIOG_MAIN.c_index_addr_id)>0))"

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    'MsgBox "About to collect second addresses"
    '
    tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id, c_name, c_name_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT ZZ_PLACE.c_addr_id, ZZ_PLACE.c_addr_name, ZZ_PLACE.c_addr_chn, ZZ_PLACE.x_co
ord, ZZ_PLACE.y_coord " + _
        "FROM ZZ_PLACE " + _
        "WHERE (((ZZ_PLACE.c_addr_id)>0))"

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' now process the file
    '
    tQueryStr = "SELECT DISTINCT ZZ_ADDRESSES.c_addr_id, ZZ_ADDRESSES.c_name, ZZ_ADDRESSES.c_name_chn, ZZ_ADD
RESSES.x_coord, " + _
        "ZZ_ADDRESSES.y_coord " + _
        "FROM ZZ_ADDRESSES"

    '
    'MsgBox "Opening addresses"
    '
    Set tRstPlace = CurrentDb.OpenRecordset(tQueryStr)
    '
    If tCodeStr = "ascii" Then
        tStr = "PlaceID" + tC + "PlacePY" + tC + "PlaceX" + tC + "PlaceY"
    Else
        tStr = "PlaceID" + tC + "PlacePY" + tC + "PlaceHZ" + tC + "PlaceX" + tC + "PlaceY"
    End If
    gStream.WriteText tStr, adWriteLine

    'MsgBox "Writing addresses"

```

```

'
With tRstPlace
    .MoveFirst
    Do While Not .EOF
        ' the ID of the place
        If Not IsNull(!c_addr_id) Then
            tStr = Trim(Str(!c_addr_id)) + tC
            '
            ' address name

            If IsNull(!c_name) Then
                tStr = tStr + "unknown" + tC
            Else
                tStr = tStr + !c_name + tC
            End If
            '
            If Not (tCodeStr = "ascii") Then
                If IsNull(!c_name_chn) Then
                    tStr = tStr + "unknown" + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If
            End If

            If IsNull(!x_coord) Then
                tStr = tStr + "0.0" + tC
            Else
                tStr = tStr + Str(!x_coord) + tC
            End If

            If IsNull(!y_coord) Then
                tStr = tStr + "0.0"
            Else
                tStr = tStr + Str(!y_coord)
            End If
            '
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces
'
dlgSaveAs.InitialFileName = "PeoplePlaceRelations_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_addr_id, ZZ_PLACE.c_assoc_id, ZZ_PLACE.c_fir

```

Form_LookAtPlace - 9

```
styear, " + _
                "ZZ_PLACE.c_lastyear, ZZ_PLACE.c_rel_type, ZZ_PLACE.c_rel_code, ZZ_PLACE.c_rel_desc, ZZ_P
PLACE.c_rel_chn " + _
                "FROM ZZ_PLACE"

Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

If tCodeStr = "ascii" Then
    tStr = "PersonID" + tC + "PlaceID" + tC + "AssocID" + tC + "PersoPlaceRelFirstYear" + tC + "PersonPla
ceRelLastYear" + tC + _
        "PersonPlaceRelType" + tC + "PersonPlaceRelCode" + tC + "PersonPlaceRelDesc"
Else
    tStr = "PersonID" + tC + "PlaceID" + tC + "AssocID" + tC + "PersoPlaceRelFirstYear" + tC + "PersonPla
ceRelLastYear" + tC + _
        "PersonPlaceRelType" + tC + "PersonPlaceRelCode" + tC + "PersonPlaceRelDesc" + tC + "PersonPlaceR
elHZ"
End If

gStream.WriteText tStr, adWriteLine

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_personid) Then
            '
            tStr = Trim(Str(!c_personid)) + tC
            '
            tStr = tStr + Trim(Str(!c_addr_id)) + tC
            '
            If IsNull(!c_assoc_id) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_assoc_id)) + tC
            End If
            '
            If IsNull(!c_firstyear) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_firstyear)) + tC
            End If
            '
            If IsNull(!c_lastyear) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_lastyear)) + tC
            End If
            '
            If IsNull(!c_rel_type) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + Trim(!c_rel_type) + tC
            End If
            '
            If IsNull(!c_rel_code) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_rel_code)) + tC
            End If
            '
            If IsNull(!c_rel_desc) Then
                tStr = tStr + "Missing"
            Else
                tStr = tStr + Trim(!c_rel_desc)
            End If
            '
            If Not (tCodeStr = "ascii") Then
                If IsNull(!c_rel_chn) Then
                    tStr = tStr + tC + "Missing"
                Else
                    tStr = tStr + tC + Trim(!c_rel_chn)
                End If
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
```

```

        ' and write the stream to the file
        gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces
'

dlgSaveAs.InitialFileName = "PeoplePlaceRelationCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_PLACE.c_rel_code,ZZ_PLACE.c_rel_type, ZZ_PLACE.c_rel_desc, ZZ_PLACE.c_re
l_chn " + _
        "FROM ZZ_PLACE WHERE (ZZ_PLACE.c_rel_code > 0)"

    Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "PersonPlaceRelCode" + tC + "PersonPlaceRelType" + tC + "PersonPlaceRelDesc"
    Else
        tStr = "PersonPlaceRelCode" + tC + "PersonPlaceRelType" + tC + "PersonPlaceRelDesc" + tC + "PersonPla
ceRelHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstPeoplePlace
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_rel_code) Then
                '
                tStr = Trim(Str(!c_rel_code)) + tC
                '
                If IsNull(!c_rel_type) Then
                    tStr = tStr + "Missing" + tC
                Else
                    tStr = tStr + Trim(!c_rel_type) + tC
                End If
                '
                If IsNull(!c_rel_desc) Then
                    tStr = tStr + "Missing"
                Else
                    tStr = tStr + Trim(!c_rel_desc)
                End If
                '
                If Not (tCodeStr = "ascii") Then
                    If IsNull(!c_rel_chn) Then
                        tStr = tStr + tC + "Missing"
                    Else
                        tStr = tStr + tC + Trim(!c_rel_chn)
                    End If
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With

```

```

End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
'The user pressed Cancel.
GoTo Exit_CmdNeo4j_Click
End If
'
' finally, get status codes
'
dlgSaveAs.InitialFileName = "IndexAddrCode_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
'
tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
tFileName = tFN
If Not tFileName = "" Then
Exit For
End If
Next
If tFileName = "" Then
MsgBox "Bad file Name."
GoTo Exit_CmdNeo4j_Click
Else
' make sure the file name has a txt extension
If Len(tFileName) < 5 Then
tFileName = tFileName + ".csv"
ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
tFileName = tFileName + ".csv"
End If
End If
'
gStream.Mode = adModeReadWrite
gStream.Type = adTypeText
gStream.Open

If tCodeStr = "ascii" Then
tStr = "IndexAddrTypeCode" + tC + "IndexAddrTypeDesc"
Else
tStr = "IndexAddrTypeCode" + tC + "IndexAddrTypeDesc" + tC + "IndexAddrTypeDescHZ"
End If
gStream.WriteText tStr, adWriteLine
'
' get the codes
'
tQueryStr = "SELECT DISTINCT ZZZ_BIOG_MAIN.c_index_addr_type_code, ZZZ_BIOG_MAIN.c_index_addr_type_desc,
" + _
"ZZZ_BIOG_MAIN.c_index_addr_type_chn " + _
"FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZZ_BIOG_
MAIN.c_personid " + _
"WHERE (ZZZ_BIOG_MAIN.c_index_addr_type_code > 0)"

Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)
With tRstPeoplePlace
.MoveFirst
Do While Not .EOF
'
tStr = Trim(Str(!c_index_addr_type_code)) + tC
'
' entry desc
'
If IsNull(!c_index_addr_type_desc) Then
tStr = tStr + "Missing"
Else
tStr = tStr + Trim(!c_index_addr_type_desc)
End If
'
' kin ID
'
If Not (tCodeStr = "ascii") Then
If IsNull(!c_index_addr_type_chn) Then
tStr = tStr + tC + "Missing"
Else
tStr = tStr + tC + Trim(!c_index_addr_type_chn)
End If

```

```

        End If
    ,
    gStream.WriteText tStr, adWriteLine
    ,
    .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
,
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
,
'Set the object variable to Nothing.
MsgBox "Finished saving to Neo4j"
Set dlgSaveAs = Nothing

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click

End Sub

Private Sub CmdPickBAC_Click()
On Error GoTo Err_CmdPickBAC_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strBAC As String

    stDocName = "frmPickBAC_multi"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog

    If CurrentProject.AllForms("frmPickBAC_multi").IsLoaded Then
        ' if the user selected a group of biographical address codes, ZZ_BIOG_ADDR_CODES will have records

        Forms!frmPickBAC_multi.Form!TxtSelectAll.Visible = True
        Forms!frmPickBAC_multi.Form!TxtSelectAll.SetFocus
        If Forms!frmPickBAC_multi.Form!TxtSelectAll.Value Then
            ' All codes have been selected. This means there is no need to filter by biographical address code
            ,
            gFilterBAC = False
        Else
            gFilterBAC = True
        End If
        ,
        DoCmd.Close acForm, "frmPickBAC_multi"
    Else
        gFilterBAC = False
    End If
    CmdPickBAC.SetFocus

Exit_CmdPickBAC_Click:
Exit Sub

Err_CmdPickBAC_Click:
MsgBox Err.Description
Resume Exit_CmdPickBAC_Click

End Sub

Private Sub CmdSelectPlace_Click()
On Error GoTo Err_CmdSelectPlace_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strADDR As String

    TxtAddrID.Visible = True
    TxtAddrID.SetFocus

```

```

strADDR = TxtAddrID.Text

stDocName = "frmPickAddresses_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR

If CurrentProject.AllForms("frmPickAddresses_multi").IsLoaded Then

    ' if the user selected a group of addresses, ZZ_ADDRESSES will have records

    Dim tAddrID As Long, tRstAddr As DAO.Recordset
    Dim strADDR_CHN As String, strADDR_PY As String
    Dim cmdSQL As ADODB.Command

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

    gUseADDRID = True
    ChkXYRef.Enabled = True
    ChkSubUnits.Enabled = True

    'MsgBox "Checking zz_addresses"
    ' tRstAddresses.MoveFirst
    Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Visible = True
    Forms!frmPickAddresses_multi.Form!TxtAddrFilter.SetFocus
    If Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Value Then
        '
        TxtAddrID.Value = 0
        strADDR_PY = Forms!frmPickAddresses_multi.Form!TxtFilterPY
        strADDR_CHN = Forms!frmPickAddresses_multi.Form!TxtFilterChn

        If strADDR_CHN = "" Then
            TxtPlaceChn.Value = "[[Filter]]"
            TxtPlace.Value = "[" + strADDR_PY + "]"
        Else
            TxtPlaceChn.Value = "[" + strADDR_CHN + "]"
            TxtPlace.Value = "[[Filter]]"
        End If
        gImportPlaces = True
    Else
        Forms!frmPickAddresses_multi.Form!TxtSelectCount.Visible = True
        Forms!frmPickAddresses_multi.Form!TxtSelectCount.SetFocus
        If Forms!frmPickAddresses_multi.Form!TxtSelectCount.Value > 1 Then
            TxtPlaceChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
            TxtPlace.Value = "[[Multi-Select]]"
            TxtAddrID.Value = 0
        Else
            ' only one record in ZZ_ADDRESSES: get its field values
            '
            Set tRstAddr = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
            tRstAddr.MoveFirst
            'MsgBox "Checking zz_addresses: no records"
            TxtAddrID.Value = tRstAddr!c_addr_id
            TxtPlaceChn.Value = tRstAddr!c_name_chn
            TxtPlace.Value = tRstAddr!c_name
            tRstAddr.Close
            Set tRstAddr = Nothing
        End If
        gImportPlaces = False
    End If
    '
    ' now copy the records
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR"
    cmdSQL.Execute tRecDeleted

    cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT " + _
        "ZZ_ADDRESSES.c_addr_id FROM ZZ_ADDRESSES"
    cmdSQL.Execute tRecDeleted

    DoCmd.Close acForm, "frmPickAddresses_multi"

    Call QueryOK

End If
CmdSelectPlace.SetFocus
TxtAddrID.Visible = False

Exit_CmdSelectPlace_Click:

```


Exit Sub

```
Err_CmdSelectPlace_Click:
MsgBox Err.Description
Resume Exit_CmdSelectPlace_Click
```

End Sub

```
Private Sub CmdQuery_Click()
On Error GoTo Err_CmdQuery_Click
```

```
Dim tRstPlace As DAO.Recordset, tRstDummy As DAO.Recordset, tUseFirstYear As Boolean, tUseLastYear As Boolean
' -
tFirstYearStr As String, tLastYearStr As String, tSNA_count As Long
Dim cmdSQL As ADODB.Command, tRecDeleted As Long, tQueryStr As String, tQueryInsertStr As String, tQueryWhere
Str As String, tQueryFromStr As String

tSNA_count = 0

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' to clear the table, briefly close and then delete records
'
Set tRstPlace = frmZZZ_PLACE.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_PL", dbOpenDynaset)
Set frmZZZ_PLACE.Form.Recordset = tRstDummy
tRstPlace.Close
'
cmdSQL.CommandText = "Delete * from ZZ_PLACE"
cmdSQL.Execute tRecDeleted
'
' now reopen
'
Set tRstPlace = CurrentDb.OpenRecordset("ZZ_PLACE", dbOpenDynaset)
Set frmZZZ_PLACE.Form.Recordset = tRstPlace
'
' ZZ_SCRATCH_PLACE_AGG
'
Set tRstPlace = ZZ_SCRATCH_PLACE_AGG.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_PLACE_AGG", dbOpenDynaset)
Set ZZ_SCRATCH_PLACE_AGG.Form.Recordset = tRstDummy
tRstPlace.Close
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PLACE_AGG"
cmdSQL.Execute tRecDeleted
'
' now reopen
'
Set tRstPlace = CurrentDb.OpenRecordset("ZZ_SCRATCH_PLACE_AGG", dbOpenDynaset)
Set ZZ_SCRATCH_PLACE_AGG.Form.Recordset = tRstPlace
'
' ZZ_SCRATCH_PLACE_PEOPLE
'
Set tRstPlace = ZZ_SCRATCH_PLACE_PEOPLE.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_PLACE_PEOPLE", dbOpenDynaset)
Set ZZ_SCRATCH_PLACE_PEOPLE.Form.Recordset = tRstDummy
tRstPlace.Close
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PLACE_PEOPLE"
cmdSQL.Execute tRecDeleted
'
' now reopen
'
Set tRstPlace = CurrentDb.OpenRecordset("ZZ_SCRATCH_PLACE_PEOPLE", dbOpenDynaset)
Set ZZ_SCRATCH_PLACE_PEOPLE.Form.Recordset = tRstPlace
'
' start with BIOG_ADDR
'
' define the rel_type
' DoCmd.RunSQL "ALTER TABLE ZZ_PLACE ALTER COLUMN c_rel_type SET DEFAULT 'BIOGRAPHY'"
'
' get the index year strings
'
tUseFirstYear = False
tUseLastYear = False
tFirstYearStr = ""
tLastYearStr = ""
```

```

If gUseIndexYears Then
    If Not IsNull(TxtFromYear.Value) Then
        tFirstYearStr = Str(TxtFromYear.Value)
        tUseFirstYear = True
        'MsgBox "First year = " + tFirstYearStr
    End If
    If Not IsNull(TxtToYear.Value) Then
        tLastYearStr = Str(TxtToYear.Value)
        tUseLastYear = True
        'MsgBox "Last year = " + tLastYearStr
    End If
End If
'
' preserve the initial list
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_ADDR_LIST"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT c_addr_id FROM ZZ_SCRATCH_ADDR"
cmdSQL.Execute tRecDeleted
'
' get the subordinate units, if selected
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
If ChkSubUnits.Value Then
    tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) " + _
        "SELECT DISTINCT ZZZ_BELONGS_TO.c_addr_id " + _
        "FROM ZZ_SCRATCH_ADDR INNER JOIN ZZZ_BELONGS_TO ON " + _
        "ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_BELONGS_TO.c_belongs_to"
Else
    tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT DISTINCT c_addr_id FROM ZZ_SCRATCH_ADDR"
End If

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
' zap ZZ_SCRATCH_ADDR
'
tQueryStr = "DELETE * FROM ZZ_SCRATCH_ADDR"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' copy the list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT ZZ_ADDRESSES.c_addr_id FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' clean up by zapping the temporary list
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' need to deal with XY ref
'
If ChkXYRef.Value Then
    '
    ' the strategy here is to dump the IDs to ZZ ADDRESSES then copy to ZZ_SCRATCH_ADDR_LIST
    ' (I borrow ZZ_ADDRESSES from the Pick Addresses form in order to keep the initial selection
    ' of addresses for the query intact.)
    '
    ' zap the list
    '
    tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
    ' run the query
    '
    tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT DISTINCT ADDR_CODES.c_addr_id " + _
        "FROM ADDR_CODES, ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES AS ADDR_CODES_1 ON " + _
        "ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES_1.c_addr_id " + _
        "WHERE (((ADDR_CODES.x_coord)>=(ADDR_CODES_1.[x_coord]-0.03) And " + _

```

```
"(ADDR_CODES.x_coord)<=([ADDR_CODES_1].[x_coord]+0.03)) AND " + _
"((ADDR_CODES.y_coord)>=([ADDR_CODES_1].[y_coord]-0.03) And " + _
"(ADDR_CODES.y_coord)<=([ADDR_CODES_1].[y_coord]+0.03)))"
```

```
cmdSQL.CommandText = tQueryStr
```

```
cmdSQL.Execute tRecDeleted
```

```
' now get the address IDs from the initial list that have no xy coordinates
```

```
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT ZZ_SCRATCH_ADDR.c_addr_id " + _
"FROM ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES ON " + _
"ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES.c_addr_id " + _
"WHERE (((ADDR_CODES.x_coord) Is Null)) OR (((ADDR_CODES.y_coord) Is Null))"
```

```
cmdSQL.CommandText = tQueryStr
```

```
cmdSQL.Execute tRecDeleted
```

```
' zap ZZ_SCRATCH_ADDR
```

```
tQueryStr = "DELETE * FROM ZZ_SCRATCH_ADDR"
```

```
cmdSQL.CommandText = tQueryStr
```

```
cmdSQL.Execute tRecDeleted
```

```
' copy the list
```

```
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT ZZ_ADDRESSES.c_addr_id " + _
"FROM ZZ_ADDRESSES"
```

```
cmdSQL.CommandText = tQueryStr
```

```
cmdSQL.Execute tRecDeleted
```

```
' clean up by zapping the temporary list
```

```
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
```

```
cmdSQL.CommandText = tQueryStr
```

```
cmdSQL.Execute tRecDeleted
```

```
End If
```

```
' For the individual
```

```
If Me.ChkIndividual.Value Then
```

```
tQueryInsertStr = "INSERT INTO ZZ_PLACE ( c_personid, c_name, c_name_chn, c_index_year, c_female, c_dy, c_dynasty, c_dynasty_chn, " + _
"c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, " + _
"c_rel_code, c_rel_desc, c_rel_chn, c_firstyear, c_lastyear, c_assoc_name, c_assoc_chn, c_assoc_id, c_rel_type, c_source ) " + _
"SELECT ZZZ_BIOG_ADDR_DATA.c_personid, ZZZ_BIOG_MAIN.c_name, ZZZ_BIOG_MAIN.c_name_chn, ZZZ_BIOG_MAIN.c_index_year, ZZZ_BIOG_MAIN.c_female, " + _
"ZZZ_BIOG_MAIN.c_dy, ZZZ_BIOG_MAIN.c_dynasty, ZZZ_BIOG_MAIN.c_dynasty_chn, " + _
"ZZZ_BIOG_ADDR_DATA.c_addr_id, ZZZ_BIOG_ADDR_DATA.c_addr_name, ZZZ_BIOG_ADDR_DATA.c_addr_chn, ZZZ_BIOG_ADDR_DATA.x_coord, ZZZ_BIOG_ADDR_DATA.y_coord, " + _
"ZZZ_BIOG_ADDR_DATA.c_addr_type, ZZZ_BIOG_ADDR_DATA.c_addr_desc, ZZZ_BIOG_ADDR_DATA.c_addr_desc_chn, ZZZ_BIOG_ADDR_DATA.c_firstyear, " + _
"ZZZ_BIOG_ADDR_DATA.c_lastyear, '[N/A]' AS c_assoc_name, '[N/A]' AS c_assoc_chn, 0 AS c_assoc_id, 'Biography' as c_rel_type, " + _
"ZZZ_BIOG_ADDR_DATA.c_source "
```

```
If gFilterBAC Then
```

```
tQueryFromStr = "FROM ((ZZZ_BIOG_ADDR_DATA INNER JOIN ZZ_SCRATCH_ADDR ON ZZZ_BIOG_ADDR_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
"INNER JOIN ZZZ_BIOG_MAIN ON ZZZ_BIOG_ADDR_DATA.c_personid = ZZZ_BIOG_MAIN.c_personid) INNER JOIN ZZ_BIOG_ADDR_CODES ON " + _
"ZZZ_BIOG_ADDR_DATA.c_addr_type = ZZ_BIOG_ADDR_CODES.c_addr_type "
```

```
Else
```

```
tQueryFromStr = "FROM (ZZZ_BIOG_ADDR_DATA INNER JOIN ZZ_SCRATCH_ADDR ON ZZZ_BIOG_ADDR_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
"INNER JOIN ZZZ_BIOG_MAIN ON ZZZ_BIOG_ADDR_DATA.c_personid = ZZZ_BIOG_MAIN.c_personid "
```

```
End If
```

```
tQueryWhereStr = ""
```

```
If tUseFirstYear Or tUseLastYear Then
```

```
If tUseFirstYear And tUseLastYear Then
```

```
tQueryWhereStr = "WHERE (((ZZZ_BIOG_MAIN.c_index_year)>= " + tFirstYearStr + " And (ZZZ_BIOG_MAIN.c_index_year)<= " + tLastYearStr + "))"
```

```
ElseIf tUseFirstYear Then
```

```
tQueryWhereStr = "WHERE (((ZZZ_BIOG_MAIN.c_index_year)>= " + tFirstYearStr + "))"
```

```
ElseIf tUseLastYear Then
```

```
tQueryWhereStr = "WHERE (((ZZZ_BIOG_MAIN.c_index_year)<= " + tLastYearStr + "))"
```

```
End If
```

```
ElseIf gUseDynasties Then
```

```

        ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
    ,
    If gFromDynasty > -2 And gFromDynasty <> gToDynasty Then
        If gFilterBAC Then
            tQueryFromStr = "FROM ZZ_BIOG_ADDR_CODES INNER JOIN (DYNASTIES INNER JOIN ((ZZZ_BIOG_ADDR_DATA INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_BIOG_ADDR_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) INNER JOIN ZZZ_BIOG_MAIN " + _
                "ON ZZZ_BIOG_ADDR_DATA.c_personid = ZZZ_BIOG_MAIN.c_personid) ON DYNASTIES.c_dy = ZZZ_BIOG_MAIN.c_dy) " + _
                "ON ZZ_BIOG_ADDR_CODES.c_addr_type = ZZZ_BIOG_ADDR_DATA.c_addr_type "
        Else
            tQueryFromStr = "FROM DYNASTIES INNER JOIN ((ZZZ_BIOG_ADDR_DATA INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_BIOG_ADDR_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) INNER JOIN ZZZ_BIOG_MAIN " + _
                "ON ZZZ_BIOG_ADDR_DATA.c_personid = ZZZ_BIOG_MAIN.c_personid) " + _
                "ON DYNASTIES.c_dy = ZZZ_BIOG_MAIN.c_dy "
        End If
    End If

    If gFromDynasty = -2 Then
        tQueryWhereStr = "Where ((ZZZ_BIOG_MAIN.c_dy) > 0 ) "
    ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
        'tQueryFromStr = "FROM DYNASTIES INNER JOIN ((ZZZ_BIOG_ADDR_DATA INNER JOIN ZZ_SCRATCH_ADDR " + _
            "ON ZZZ_BIOG_ADDR_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) INNER JOIN ZZZ_BIOG_MAIN " + _
            "ON ZZZ_BIOG_ADDR_DATA.c_personid = ZZZ_BIOG_MAIN.c_personid) " + _
            "ON DYNASTIES.c_dy = ZZZ_BIOG_MAIN.c_dy "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
        'tQueryFromStr = "FROM DYNASTIES INNER JOIN ((ZZZ_BIOG_ADDR_DATA INNER JOIN ZZ_SCRATCH_ADDR " + _
            "ON ZZZ_BIOG_ADDR_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) INNER JOIN ZZZ_BIOG_MAIN " + _
            "ON ZZZ_BIOG_ADDR_DATA.c_personid = ZZZ_BIOG_MAIN.c_personid) " + _
            "ON DYNASTIES.c_dy = ZZZ_BIOG_MAIN.c_dy "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tQueryWhereStr = "WHERE ((ZZZ_BIOG_MAIN.c_dy) = " + Str(gFromDynasty) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
            "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
        'tQueryFromStr = "FROM DYNASTIES INNER JOIN ((ZZZ_BIOG_ADDR_DATA INNER JOIN ZZ_SCRATCH_ADDR " + _
            "ON ZZZ_BIOG_ADDR_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) INNER JOIN ZZZ_BIOG_MAIN " + _
            "ON ZZZ_BIOG_ADDR_DATA.c_personid = ZZZ_BIOG_MAIN.c_personid) " + _
            "ON DYNASTIES.c_dy = ZZZ_BIOG_MAIN.c_dy "
    Else
        tQueryWhereStr = ""
    End If

    End If

    cmdSQL.CommandText = tQueryInsertStr + tQueryFromStr + tQueryWhereStr
    cmdSQL.Execute tRecDeleted
End If

' for office postings

If Me.ChkOffice.Value Then
    tQueryInsertStr = "INSERT INTO ZZ_PLACE ( c_personid, c_name, c_name_chn, c_index_year, c_female, c_dy, c_dynasty, c_dynasty_chn, " + _
        "c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_rel_type, c_rel_code, c_rel_desc, c_rel_chn, c_firstyear, c_lastyear, c_source ) " + _
        "SELECT ZZZ_POSTED_TO_ADDR_DATA.c_personid, ZZZ_POSTED_TO_ADDR_DATA.c_person_name, ZZZ_POSTED_TO_ADDR_DATA.c_person_name_chn, " + _
        "ZZZ_POSTED_TO_ADDR_DATA.c_index_year, ZZZ_POSTED_TO_ADDR_DATA.c_female, ZZZ_POSTED_TO_ADDR_DATA.c_person_dy, ZZZ_POSTED_TO_ADDR_DATA.c_person_dynasty, " + _
        "ZZZ_POSTED_TO_ADDR_DATA.c_person_dy_chn, ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_id, ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_name, " + _
        "ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_chn, ZZZ_POSTED_TO_ADDR_DATA.office_x_coord, " + _
        "ZZZ_POSTED_TO_ADDR_DATA.office_y_coord, 'Office Place' AS c_rel_type, ZZZ_POSTED_TO_ADDR_DATA.c_office_id, " + _
        "ZZZ_POSTED_TO_ADDR_DATA.c_office_trans, ZZZ_POSTED_TO_ADDR_DATA.c_office_chn, ZZZ_POSTED_TO_ADDR_DATA.c_firstyear, " + _
        "ZZZ_POSTED_TO_ADDR_DATA.c_lastyear, ZZZ_POSTED_TO_ADDR_DATA.c_source "
    tQueryFromStr = "FROM ZZ_SCRATCH_ADDR INNER JOIN ZZZ_POSTED_TO_ADDR_DATA ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_id "

    tQueryWhereStr = ""

    If tUseFirstYear Or tUseLastYear Then
        If tUseFirstYear And tUseLastYear Then

```

```

tQueryWhereStr = "WHERE (((ZZZ_POSTED_TO_ADDR_DATA.c_index_year)>= " + tFirstYearStr + _
    " And (ZZZ_POSTED_TO_ADDR_DATA.c_index_year)<= " + tLastYearStr + "))"
ElseIf tUseFirstYear Then
    tQueryWhereStr = "WHERE (((ZZZ_POSTED_TO_ADDR_DATA.c_index_year)>= " + tFirstYearStr + "))"
ElseIf tUseLastYear Then
    tQueryWhereStr = "WHERE (((ZZZ_POSTED_TO_ADDR_DATA.c_index_year)<= " + tLastYearStr + "))"
End If
ElseIf gUseDynasties Then
    ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
    If gFromDynasty = -2 Then
        tQueryWhereStr = "Where ((ZZZ_POSTED_TO_ADDR_DATA.c_person_dy) > 0 ) "
    ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
        tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_POSTED_TO_ADDR_DATA "
            "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_id) " + _
            "ON DYNASTIES.c_dy = ZZZ_POSTED_TO_ADDR_DATA.c_person_dy "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
        tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_POSTED_TO_ADDR_DATA "
            "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_id) " + _
            "ON DYNASTIES.c_dy = ZZZ_POSTED_TO_ADDR_DATA.c_person_dy "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tQueryWhereStr = "WHERE ((ZZZ_POSTED_TO_ADDR_DATA.c_person_dy) = " + Str(gFromDynasty) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
            "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
        tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_POSTED_TO_ADDR_DATA "
            "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_POSTED_TO_ADDR_DATA.c_office_addr_id) " + _
            "ON DYNASTIES.c_dy = ZZZ_POSTED_TO_ADDR_DATA.c_person_dy "
    Else
        tQueryWhereStr = ""
    End If
End If

cmdSQL.CommandText = tQueryInsertStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
End If

' For entry

If Me.ChkEntry.Value Then
    tQueryInsertStr = "INSERT INTO ZZ_PLACE ( c_personid, c_name, c_name_chn, c_index_year, c_female, c_dy, c_dynasty, c_dynasty_chn, " + _
        "c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_firstyear, " + _
        "c_rel_type, c_rel_code, c_rel_desc, c_rel_chn, c_source ) " + _
        "SELECT ZZZ_ENTRY_DATA.c_personid, ZZZ_ENTRY_DATA.c_name, ZZZ_ENTRY_DATA.c_name_chn, ZZZ_ENTRY_DATA.c_index_year, ZZZ_ENTRY_DATA.c_female, ZZZ_ENTRY_DATA.c_dy, " + _
        "ZZZ_ENTRY_DATA.c_dynasty, ZZZ_ENTRY_DATA.c_dynasty_chn, ZZZ_ENTRY_DATA.c_entry_addr_id, ZZZ_ENTRY_DATA.c_entry_addr_name, " + _
        "ZZZ_ENTRY_DATA.c_entry_addr_chn, ZZZ_ENTRY_DATA.c_entry_xcoord, ZZZ_ENTRY_DATA.c_entry_ycoord, ZZZ_ENTRY_DATA.c_year, " + _
        "'Entry' AS c_rel_type, ZZZ_ENTRY_DATA.c_entry_code, ZZZ_ENTRY_DATA.c_entry_desc, ZZZ_ENTRY_DATA.c_entry_desc_chn, ZZZ_ENTRY_DATA.c_source "
    tQueryFromStr = "FROM ZZ_SCRATCH_ADDR INNER JOIN ZZZ_ENTRY_DATA ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_ENTRY_DATA.c_entry_addr_id "

    tQueryWhereStr = ""

    If tUseFirstYear Or tUseLastYear Then
        If tUseFirstYear And tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_ENTRY_DATA.c_index_year)>= " + tFirstYearStr + _
                " And (ZZZ_ENTRY_DATA.c_index_year)<= " + tLastYearStr + "))"
        ElseIf tUseFirstYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_ENTRY_DATA.c_index_year)>= " + tFirstYearStr + _
                " And (ZZZ_ENTRY_DATA.c_index_year)<= " + tLastYearStr + "))"
        ElseIf tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_ENTRY_DATA.c_index_year)<= " + tLastYearStr + _
                " And (ZZZ_ENTRY_DATA.c_index_year)>= " + tFirstYearStr + "))"
        End If
    ElseIf gUseDynasties Then
        ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
        If gFromDynasty = -2 Then
            tQueryWhereStr = "Where ((ZZZ_ENTRY_DATA.c_dy) > 0 ) "
        ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_ENTRY_DATA "
                "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_ENTRY_DATA.c_entry_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "
        ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_ENTRY_DATA "
                "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_ENTRY_DATA.c_entry_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "
        ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
            tQueryWhereStr = "WHERE ((ZZZ_ENTRY_DATA.c_dy) = " + Str(gFromDynasty) + ") "
        ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
                "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_ENTRY_DATA "
                "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_ENTRY_DATA.c_entry_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "
        Else
            tQueryWhereStr = ""
        End If
    End If
End If

```

```

ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
    tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_ENTRY_DATA " + _
        "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_ENTRY_DATA.c_entry_addr_id) " + _
        "ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "
ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
    tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
    tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_ENTRY_DATA " + _
        "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_ENTRY_DATA.c_entry_addr_id) " + _
        "ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "
ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
    tQueryWhereStr = "WHERE ((ZZZ_ENTRY_DATA.c_dy) = " + Str(gFromDynasty) + ") "
ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
    tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
        "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_ENTRY_DATA " + _
        "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_ENTRY_DATA.c_entry_addr_id) " + _
        "ON DYNASTIES.c_dy = ZZZ_ENTRY_DATA.c_dy "
Else
    tQueryWhereStr = ""
End If

cmdSQL.CommandText = tQueryInsertStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
End If

' For kinship

If Me.ChkKin.Value Then
    tQueryInsertStr = "INSERT INTO ZZ_PLACE ( c_personid, c_name, c_name_chn, c_index_year, c_female, c_dy, c_
_dynasty, c_dynasty_chn, " + _
        "c_addr_id, c_addr_name, c_addr_chn, c_assoc_id, c_assoc_name, " + _
        "c_assoc_chn, c_rel_type, c_rel_code, c_rel_desc, c_rel_chn, x_coord, y_coord, c_assoc_index_year
, assoc_x_coord, assoc_y_coord, c_source ) " + _
        "SELECT ZZZ_KIN_BIOG_ADDR.c_personid, ZZZ_KIN_BIOG_ADDR.c_person_name, ZZZ_KIN_BIOG_ADDR.c_person_nam
e_chn, ZZZ_KIN_BIOG_ADDR.c_index_year, " + _
        "ZZZ_KIN_BIOG_ADDR.c_female, ZZZ_KIN_BIOG_ADDR.c_dy, ZZZ_KIN_BIOG_ADDR.c_dynasty, ZZZ_KIN_BIOG_AD
DR.c_dynasty_chn, " + _
        "ZZZ_KIN_BIOG_ADDR.c_node_addr_id, ZZZ_KIN_BIOG_ADDR.c_node_addr_name, ZZZ_KIN_BIOG_ADDR.c_node_a
ddr_chn, ZZZ_KIN_BIOG_ADDR.c_node_id, " + _
        "ZZZ_KIN_BIOG_ADDR.c_node_name, ZZZ_KIN_BIOG_ADDR.c_node_chn, 'Kinship' AS c_rel_type, ZZZ_KIN_BI
OG_ADDR.c_link_code, ZZZ_KIN_BIOG_ADDR.c_link_desc, " + _
        "ZZZ_KIN_BIOG_ADDR.c_link_chn, ZZZ_KIN_BIOG_ADDR.person_x_coord, ZZZ_KIN_BIOG_ADDR.person_y_coord
, ZZZ_KIN_BIOG_ADDR.c_node_index_year, " + _
        "ZZZ_KIN_BIOG_ADDR.node_xcoord, ZZZ_KIN_BIOG_ADDR.node_ycoord, ZZZ_KIN_BIOG_ADDR.c_source "
    tQueryFromStr = "FROM ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_SCRATCH_ADDR ON ZZZ_KIN_BIOG_ADDR.c_node_addr_id =
ZZ_SCRATCH_ADDR.c_addr_id "

    tQueryWhereStr = ""

    If tUseFirstYear Or tUseLastYear Then
        If tUseFirstYear And tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_KIN_BIOG_ADDR.c_index_year)>= " + tFirstYearStr + "
-
            " And (ZZZ_KIN_BIOG_ADDR.c_index_year)<= " + tLastYearStr + ")))"
        ElseIf tUseFirstYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_KIN_BIOG_ADDR.c_index_year)>= " + tFirstYearStr + "
)))"
        ElseIf tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_KIN_BIOG_ADDR.c_index_year)<= " + tLastYearStr + "
)))"
        End If
    ElseIf gUseDynasties Then
        ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
        ,
        If gFromDynasty = -2 Then
            tQueryWhereStr = "Where ((ZZZ_KIN_BIOG_ADDR.c_dy) > 0 ) "
        ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_KIN_BIOG_ADDR.c_node_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_KIN_BIOG_ADDR.c_dy "
        ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_KIN_BIOG_ADDR.c_node_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_KIN_BIOG_ADDR.c_dy "
        ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then

```

```

        tQueryWhereStr = "WHERE ((ZZZ_KIN_BIOG_ADDR.c_dy) = " + Str(gFromDynasty) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
            "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
        tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_KIN_BIOG_ADDR INNER JOIN ZZ_SCRATCH_ADDR " + _
            "ON ZZZ_KIN_BIOG_ADDR.c_node_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
            "ON DYNASTIES.c_dy = ZZZ_KIN_BIOG_ADDR.c_dy "
    Else
        tQueryWhereStr = ""
    End If
End If

cmdSQL.CommandText = tQueryInsertStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted

tSNA_count = tRecDeleted

End If

' For the institutions

If Me.ChkInstitution.Value Then
    tQueryInsertStr = "INSERT INTO ZZ_PLACE ( c_personid, c_name, c_name_chn, c_index_year, c_female, c_dy, c_
_dynasty, c_dynasty_chn, " + _
        "c_addr_id, c_addr_name, c_assoc_chn, c_assoc_name, " + _
        "c_addr_chn, c_assoc_id, c_firstyear, c_lastyear, c_rel_type, c_rel_code, c_rel_desc, c_rel_chn,
x_coord, y_coord, c_source ) " + _
        "SELECT ZZZ_BIOG_INST_DATA.c_personid, ZZZ_BIOG_INST_DATA.c_name, ZZZ_BIOG_INST_DATA.c_name_chn, ZZZ_
BIOG_INST_DATA.c_index_year, " + _
        "ZZZ_BIOG_INST_DATA.c_female, ZZZ_BIOG_INST_DATA.c_dy, ZZZ_BIOG_INST_DATA.c_dynasty, ZZZ_BIOG_INST_
T_DATA.c_dynasty_chn, " + _
        "ZZ_SCRATCH_ADDR.c_addr_id, ZZZ_BIOG_INST_DATA.c_inst_addr_pinyin, ZZZ_BIOG_INST_DATA.c_inst_name
_hz, ZZZ_BIOG_INST_DATA.c_inst_name_py, " + _
        "ZZZ_BIOG_INST_DATA.c_inst_addr_chn, 0 AS c_assoc_id, ZZZ_BIOG_INST_DATA.c_bi_begin_year, ZZZ_BIO
G_INST_DATA.c_bi_end_year, " + _
        "'Institution' AS c_rel_type, ZZZ_BIOG_INST_DATA.c_bi_role_code, ZZZ_BIOG_INST_DATA.c_bi_role_des
c, " + _
        "ZZZ_BIOG_INST_DATA.c_bi_role_chn, ZZZ_BIOG_INST_DATA.inst_xcoord, ZZZ_BIOG_INST_DATA.inst_ycoord
, ZZZ_BIOG_INST_DATA.c_source "
    tQueryFromStr = "FROM ZZZ_BIOG_INST_DATA INNER JOIN ZZ_SCRATCH_ADDR ON ZZZ_BIOG_INST_DATA.c_inst_addr_id
= ZZ_SCRATCH_ADDR.c_addr_id "

    tQueryWhereStr = ""

    If tUseFirstYear Or tUseLastYear Then
        If tUseFirstYear And tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_BIOG_INST_DATA.c_index_year)>= " + tFirstYearStr
+ _
                " And (ZZZ_BIOG_INST_DATA.c_index_year)<= " + tLastYearStr + "))"
        ElseIf tUseFirstYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_BIOG_INST_DATA.c_index_year)>= " + tFirstYearStr
+ _
                ")))"
        ElseIf tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_BIOG_INST_DATA.c_index_year)<= " + tLastYearStr +
_
                ")))"
        End If
    End If
    ElseIf gUseDynasties Then
        ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
        If gFromDynasty = -2 Then
            tQueryWhereStr = "Where ((ZZZ_BIOG_INST_DATA.c_dy) > 0 ) "
        ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_BIOG_INST_DATA INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_BIOG_INST_DATA.c_inst_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_BIOG_INST_DATA.c_dy "
        ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_BIOG_INST_DATA INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_BIOG_INST_DATA.c_inst_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_BIOG_INST_DATA.c_dy "
        ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
            tQueryWhereStr = "WHERE ((ZZZ_BIOG_INST_DATA.c_dy) = " + Str(gFromDynasty) + ") "
        ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
                "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_BIOG_INST_DATA INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_BIOG_INST_DATA.c_inst_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_BIOG_INST_DATA.c_dy "

```

```

Else
    tQueryWhereStr = ""
End If
End If

cmdSQL.CommandText = tQueryInsertStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted
End If

' For associates

If Me.ChkAssocPerson.Value Then
    tQueryInsertStr = "INSERT INTO ZZ_PLACE ( c_personid, c_name, c_name_chn, c_index_year, c_female, c_dy, c_dynasty, c_dynasty_chn, " + _
        "c_addr_id, c_addr_name, c_addr_chn, c_assoc_id, c_assoc_name, " + _
        "c_assoc_chn, c_rel_type, c_rel_code, c_rel_desc, c_rel_chn, x_coord, y_coord, c_assoc_index_year, assoc_x_coord, assoc_y_coord, c_source ) " + _
        "SELECT ZZZ_NONKIN_BIOG_ADDR.c_personid, ZZZ_NONKIN_BIOG_ADDR.c_person_name, ZZZ_NONKIN_BIOG_ADDR.c_person_name_chn, ZZZ_NONKIN_BIOG_ADDR.c_index_year, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_female, ZZZ_NONKIN_BIOG_ADDR.c_dy, ZZZ_NONKIN_BIOG_ADDR.c_dynasty, ZZZ_NONKIN_BIOG_ADDR.c_dynasty_chn, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_name, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_chn, ZZZ_NONKIN_BIOG_ADDR.c_node_id, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_name, ZZZ_NONKIN_BIOG_ADDR.c_node_chn, 'Associate Place' AS c_rel_type, ZZZ_NONKIN_BIOG_ADDR.c_link_code, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_link_desc, ZZZ_NONKIN_BIOG_ADDR.c_link_chn, ZZZ_NONKIN_BIOG_ADDR.person_x_coord, ZZZ_NONKIN_BIOG_ADDR.person_y_coord, " + _
        "ZZZ_NONKIN_BIOG_ADDR.c_node_index_year, ZZZ_NONKIN_BIOG_ADDR.node_xcoord, ZZZ_NONKIN_BIOG_ADDR.node_ycoord, ZZZ_NONKIN_BIOG_ADDR.c_source "
    tQueryFromStr = "FROM ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZ_SCRATCH_ADDR ON ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id = ZZ_SCRATCH_ADDR.c_addr_id "

    tQueryWhereStr = ""

    If tUseFirstYear Or tUseLastYear Then
        If tUseFirstYear And tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)>= " + tFirstYearStr + _
                " And (ZZZ_NONKIN_BIOG_ADDR.c_index_year)<= " + tLastYearStr + "))"
        ElseIf tUseFirstYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)>= " + tFirstYearStr + _
                ")))"
        ElseIf tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)<= " + tLastYearStr + _
                ")))"
        End If
    ElseIf gUseDynasties Then
        ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
        If gFromDynasty = -2 Then
            tQueryWhereStr = "Where ((ZZZ_NONKIN_BIOG_ADDR.c_dy) > 0 ) "
        ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy "
        ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy "
        ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
            tQueryWhereStr = "WHERE ((ZZZ_NONKIN_BIOG_ADDR.c_dy) = " + Str(gFromDynasty) + ") "
        ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
                "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZZ_NONKIN_BIOG_ADDR INNER JOIN ZZ_SCRATCH_ADDR " + _
                "ON ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " + _
                "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy "
        Else
            tQueryWhereStr = ""
        End If
    End If

    cmdSQL.CommandText = tQueryInsertStr + tQueryFromStr + tQueryWhereStr
    cmdSQL.Execute tRecDeleted

    tSNA_count = tSNA_count + tRecDeleted

```



```

End If

' For the association

If Me.ChkAssocPlace.Value Then
    tQueryInsertStr = "INSERT INTO ZZ_PLACE ( c_personid, c_name, c_name_chn, c_index_year, c_female, c_dy, c_dynasty, c_dynasty_chn, " +
        "c_addr_id, c_addr_name, c_addr_chn, c_rel_addr_id, c_rel_addr_name, c_rel_addr_chn, c_assoc_id, c_assoc_name, " +
        "c_assoc_chn, c_rel_type, c_rel_code, c_rel_desc, c_rel_chn, x_coord, y_coord, c_source, c_firstyear, c_lastyear ) " +
        "SELECT ZZZ_NONKIN_BIOG_ADDR.c_personid, ZZZ_NONKIN_BIOG_ADDR.c_person_name, ZZZ_NONKIN_BIOG_ADDR.c_person_name_chn, ZZZ_NONKIN_BIOG_ADDR.c_index_year, " +
        "ZZZ_NONKIN_BIOG_ADDR.c_female, ZZZ_NONKIN_BIOG_ADDR.c_dy, ZZZ_NONKIN_BIOG_ADDR.c_dynasty, ZZZ_NONKIN_BIOG_ADDR.c_dynasty_chn, " +
        "ZZZ_NONKIN_BIOG_ADDR.c_node_addr_id, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_name, ZZZ_NONKIN_BIOG_ADDR.c_node_addr_chn, " +
        "ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_id, ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_name, " +
        "ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_chn, ZZZ_NONKIN_BIOG_ADDR.c_node_id, ZZZ_NONKIN_BIOG_ADDR.c_node_name, " +
        "ZZZ_NONKIN_BIOG_ADDR.c_node_chn, 'Place of Association' AS c_rel_type, ZZZ_NONKIN_BIOG_ADDR.c_link_code, ZZZ_NONKIN_BIOG_ADDR.c_link_desc, " +
        "ZZZ_NONKIN_BIOG_ADDR.c_link_chn, ADDR_CODES.x_coord, ADDR_CODES.y_coord, ZZZ_NONKIN_BIOG_ADDR.c_source, " +
        "ZZZ_NONKIN_BIOG_ADDR.c_assoc_first_year, ZZZ_NONKIN_BIOG_ADDR.c_assoc_last_year "
    tQueryFromStr = "FROM ADDR_CODES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR " +
        "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_id) ON ADDR_CODES.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id "
    tQueryWhereStr = ""

    If tUseFirstYear Or tUseLastYear Then
        If tUseFirstYear And tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_assoc_last_year)>= " + tFirstYearStr +
                " And (ZZZ_NONKIN_BIOG_ADDR.c_assoc_first_year)<= " + tLastYearStr + "))"
        ElseIf tUseFirstYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_assoc_last_year)>= " + tFirstYearStr +
                " OR (ZZZ_NONKIN_BIOG_ADDR.c_assoc_first_year)>= " + tFirstYearStr + "))"
        ElseIf tUseLastYear Then
            tQueryWhereStr = tQueryWhereStr + "WHERE (((ZZZ_NONKIN_BIOG_ADDR.c_index_year)<= " + tLastYearStr +
                " OR (ZZZ_NONKIN_BIOG_ADDR.c_assoc_last_year)<= " + tLastYearStr + "))"
        End If
    ElseIf gUseDynasties Then
        ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
        If gFromDynasty = -2 Then
            tQueryWhereStr = "Where ((ZZZ_NONKIN_BIOG_ADDR.c_dy) > 0 ) "
        ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ADDR_CODES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR " +
                "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_id) " +
                "ON ADDR_CODES.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " +
                "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy "
        ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ADDR_CODES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR " +
                "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_id) " +
                "ON ADDR_CODES.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " +
                "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy "
        ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
            tQueryWhereStr = "WHERE ((ZZZ_NONKIN_BIOG_ADDR.c_dy) = " + Str(gFromDynasty) + ") "
        ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
            tQueryWhereStr = "WHERE ((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " +
                "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
            tQueryFromStr = "FROM DYNASTIES INNER JOIN (ADDR_CODES INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_NONKIN_BIOG_ADDR " +
                "ON ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_NONKIN_BIOG_ADDR.c_assoc_addr_id) " +
                "ON ADDR_CODES.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) " +
                "ON DYNASTIES.c_dy = ZZZ_NONKIN_BIOG_ADDR.c_dy "
        Else
            tQueryWhereStr = ""
        End If
    End If
End If

```

```

cmdSQL.CommandText = tQueryInsertStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecDeleted

tSNA_count = tSNA_count + tRecDeleted

End If
'
' get the index year descriptive data for people
'
cmdSQL.CommandText = "UPDATE ZZZ_BIOG_MAIN AS ZZZ_BIOG_MAIN_1 INNER JOIN (ZZZ_BIOG_MAIN INNER JOIN ZZ_PLACE "
+ "ON ZZZ_BIOG_MAIN.c_personid = ZZ_PLACE.c_personid) ON ZZZ_BIOG_MAIN_1.c_personid = ZZ_PLACE.c_assoc_id "
+ "SET ZZ_PLACE.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
  "ZZ_PLACE.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
  "ZZ_PLACE.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz], " + _
  "ZZ_PLACE.c_assoc_index_year_type_code = [ZZZ_BIOG_MAIN_1].[c_index_year_type_code], " + _
  "ZZ_PLACE.c_assoc_index_year_type_desc = [ZZZ_BIOG_MAIN_1].[c_index_year_type_desc], " + _
  "ZZ_PLACE.c_assoc_index_year_type_hz = [ZZZ_BIOG_MAIN_1].[c_index_year_type_hz]"
cmdSQL.Execute tRecDeleted
'
' finally, get the source titles
'
cmdSQL.CommandText = "UPDATE ZZ_PLACE INNER JOIN TEXT_CODES ON ZZ_PLACE.c_source = TEXT_CODES.c_textid " + _
  "SET ZZ_PLACE.c_source_text = [TEXT_CODES].[c_title], " + _
  "ZZ_PLACE.c_source_text_chn = [TEXT_CODES].[c_title_chn]"
cmdSQL.Execute tRecDeleted
'
' now reopen
'
Set tRstPlace = CurrentDb.OpenRecordset("ZZ_PLACE", dbOpenDynaset)
Set frmZZ_PLACE.Form.Recordset = tRstPlace
'
' the final step is to calculate the xy_count
'
If tRstPlace.RecordCount > 0 Then
'
' get the aggregated records
'
Call getAggregatedRecords
'
' get the people
'
Call getPeopleRecords
'
' calculate_xy_count
'
cmdSQL.CommandText = "Delete * from tmpXY"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
  "SELECT ZZ_PLACE.x_coord, ZZ_PLACE.y_coord, Count(ZZ_PLACE.x_coord) AS CountOfx_coord, Count(ZZ_PLACE
.y_coord) AS CountOfy_coord " + _
  "FROM ZZ_PLACE " + _
  "GROUP BY ZZ_PLACE.x_coord, ZZ_PLACE.y_coord"
'
cmdSQL.Execute tRecCount
'
cmdSQL.CommandText = "UPDATE tmpXY INNER JOIN ZZ_PLACE ON (tmpXY.y_coord = ZZ_PLACE.y_coord) AND (tmpXY.x
_coord = ZZ_PLACE.x_coord) " + _
  "SET ZZ_PLACE.xy_count = [tmpXY].[CountOfx_coord]"
'
cmdSQL.Execute tRecCount

CmdStoreID.Enabled = True
CmdNeo4j.Enabled = True
Else
CmdStoreID.Enabled = False
CmdNeo4j.Enabled = False
End If

If tSNA_count > 0 Then
CmdGephi.Enabled = True
CmdPajek.Enabled = True
CmdUCINet.Enabled = True
Else
CmdGephi.Enabled = False
CmdPajek.Enabled = False
CmdUCINet.Enabled = False
End If

```

```

'
' restore the initial list
'
cmdSQL.CommandText = "DELETE * FROM ZZ_SCRATCH_ADDR"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) SELECT DISTINCT c_addr_id FROM ZZ_SCRATCH_ADD
R_LIST"
cmdSQL.Execute tRecDeleted

Exit_CmdQuery_Click:
'
' close everything
'
Set tRstPlace = Nothing
Set tRstDummy = Nothing
Set cmdSQL = Nothing
Exit Sub

Err_CmdQuery_Click:
MsgBox Err.Description
Resume Exit_CmdQuery_Click

End Sub

Private Sub calculate_xy_count()
Dim tX As Double, tY As Double, tXY As Integer
Dim tBM As Variant, tWrite As Integer, tRstPlace As DAO.Recordset
'
' the strategy is to first throw a bookmark at the first new value
' then count the number, then go back to the bookmark and update each record
'
' in order to do this, I need to open the table as a table
'
Set tRstPlace = CurrentDb.OpenRecordset("ZZ_PLACE", dbOpenTable)
With tRstPlace
.Index = "xy"
.MoveFirst

tX = -1#
tY = -1#
tXY = 0
tWrite = 0
tBM = .Bookmark

Do While Not .EOF
If tX <> !x_coord Or tY <> !y_coord Then
If tWrite = 1 Then
' go back to the first record with the value
.Bookmark = tBM
Do While tX = !x_coord And tY = !y_coord
.Edit
!xy_count = tXY
.Update
.MoveNext
Loop
Else
tWrite = 1
End If
' reset
tXY = 0
tBM = .Bookmark
tX = !x_coord
tY = !y_coord
End If
' increment the count and move to the next
tXY = tXY + 1
.MoveNext
Loop
'
' the last xy value still needs to be written
'
.Bookmark = tBM
Do While Not .EOF
.Edit
!xy_count = tXY
.Update
.MoveNext
Loop
.Index = "name"
End With

```

```

tRstPlace.Close
Set tRstPlace = Nothing

End Sub

Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
    ' This program will dump the results to a .gis file
    '
    If frmZZZ_PLACE.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_CmdGIS_Click
    End If
    '
    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream

    If GISFrame.Value = 1 Then
        tStream.Charset = "utf-8"
        tCodeStr = "UTF8"
    Else
        tStream.Charset = "gb18030"
        tCodeStr = "GB18030"
    End If
    tStream.Mode = adModeReadWrite
    tStream.Type = adTypeText
    tStream.Open
    '
    ' next get a file
    '
    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant, tFemale As String
    Dim tRstNode As DAO.Recordset
    Dim tStr As String, tC As String, ti As Integer

    Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".txt"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdGIS_Click
        End If
        '
        ' write the file
        '
        'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
        '
        ' process the table
        '
        Set tRstNode = frmZZZ_PLACE.Form.Recordset
        tC = Chr(44) ' the comma
        '
        With tRstNode
            ' write the header
            '
            tStr = "Name" + tC + "NameChn" + tC + "IndexYear" + tC
            tStr = tStr + "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC
            tStr = tStr + "xy_count"
            tStream.WriteText tStr, adWriteLine
            .MoveFirst
            Do While Not .EOF
                ' must guard against NULLs
                '
                If Trim(!c_name) = "" Then
                    tStr = "[?]" + tC
                Else
                    tStr = !c_name + tC
                End If
            End If
        End With
    End If

```

```

    If Trim(!c_name_chn) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_name_chn + tC
    End If

    If IsNull(!c_index_year) Then
        tStr = tStr + "-2000" + tC
    Else
        tStr = tStr + Str(!c_index_year) + tC
    End If

```

```

' here guard against blanks as well

```

```

    If IsNull(!c_addr_name) Then
        tStr = tStr + "[?]" + tC
    ElseIf Trim(!c_addr_name) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If

```

```

    If IsNull(!c_addr_chn) Then
        tStr = tStr + "[?]" + tC
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_addr_chn + tC
    End If

```

```

    If IsNull(!x_coord) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Str(!x_coord) + tC
    End If

```

```

    If IsNull(!y_coord) Then
        tStr = tStr + "0" + tC
    Else
        tStr = tStr + Str(!y_coord) + tC
    End If

```

```

    If IsNull(!xy_count) Then
        tStr = tStr + "0"
    Else
        tStr = tStr + Str(!xy_count)
    End If

```

```

    tStream.WriteText tStr, adWriteLine
    .MoveNext

```

```

    Loop
End With
Else
    'The user pressed Cancel.
End If

```

```

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

```

```

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

```

```

Exit_CmdGIS_Click:
Exit Sub

```

```

Err_CmdGIS_Click:
MsgBox Err.Description
Resume Exit_CmdGIS_Click

```

```

End Sub

```

```

Private Sub CmdToDynasty_Click()

```

```

Dim stDocName As String
Dim stLinkCriteria As String
Dim strToDynasty As String

If gToDynasty = -1 Then
    strToDynasty = ""
Else
    strToDynasty = Str(gToDynasty)
End If

stDocName = "frmPickDynasty"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

    Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
    gToDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
    gToDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
    gToDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
    '
    ' check to see if we have a problem and reject selection if needed
    '
    If gFromDynasty > -1 Then
        If gFromDynastyBegin > gToDynastyEnd Then
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
            gToDynasty = -1
            TxtToDynasty.Value = ""
            TxtToDynastyPY.Value = ""
        End If
    End If
    '
    ' value is OK
    '
    If gToDynasty > -1 Then
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
        TxtToDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
        TxtToDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
    End If

    DoCmd.Close acForm, stDocName
    '
    ' reset FromDynasty if necessary (-2 = all dynasties)
    '
    If gFromDynasty = -2 Then
        gFromDynasty = -1
        TxtFromDynasty.Value = ""
        TxtFromDynastyPY.Value = ""
    End If
End If

End Sub

Private Sub CmdUCINet_Click()
On Error GoTo Err_CmdUCINet_Click
    '
    ' This program will dump the results of the search to a .vna file
    '
    ' for the moment I'll just describe the format of the .vna file
    '
    ' *node data
    ' ID index_year sex x_coord y_coord nodedist
    ' ID = str(c_person_id)
    ' indexyear = c_index_year INT
    ' nodedist = c_node_dist INT
    ' sex = c_female > (F,M)
    ' *node properties
    ' ID color shape size shortlabel active
    ' color = red (1), orange (2), yellow (3), green (4), blue (5)
    ' shortlabel = c_name
    ' shape = 2
    ' active =TRUE
    '

```

```

' *tie data
' from to edgetype nodedist
'   from = str(c_person_id)
'   to = str(c_node_id)
'   edgetype= c_link_type (K,N)
'
' *tie properties
' from to color size active
'   from = str(c_person_id)
'   to = str(c_node_id)
'   color = red (255), orange (26367), yellow (65535), green (32768), blue (16711680)
'   size = 1-5 (the weight)
'
' the central question is whether to do distance optimizations
'
' first see if there are any records to process
'
If frmZZZ_PLACE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUCINet_Click
End If
'
' to write to a UTF-8 file, use the ADO stream object
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8.net"
ElseIf CodeFrame.Value = 2 Then
    tStream.Charset = "big5"
    tCodeStr = "BIG5.net"
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030.net"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstAssocType As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tSearchStr As String
Dim tColor(20) As String, tQuote As String
Dim tFileSystem, tVNA

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network.vna"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdUCINet_Click
        Else
            ' make sure the file name has a vna extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".vna"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".vna") Then
                tFileName = tFileName + ".vna"
            End If
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
    'Set tVNA = tFileSystem.CreateTextFile(tFileName, True)

```

```

' now prepare the node list by getting all the person ID and the assoc IDs
'
' the strategy is to dump both into ZZ_SOCIAL_NETWORK and then copy to ZZ_SCRATCH_PEOPLE
'
Dim cmdSQL As ADODB.Command
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted

tStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year ) " + _
"SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_name, ZZ_PLACE.c_name_chn, ZZ_PLACE.c_index_year " + _
"FROM ZZ_PLACE WHERE (((ZZ_PLACE.c_rel_type)='Kinship')) OR (((ZZ_PLACE.c_rel_type)='Associate Pl" + _
"ace')))"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
If tRecDeleted = 0 Then
    MsgBox "There are no networks associated with this place."
    Exit Sub
End If

tStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year ) " + _
"SELECT DISTINCT ZZ_PLACE.c_assoc_id, ZZ_PLACE.c_assoc_name, ZZ_PLACE.c_assoc_chn, ZZ_PLACE.c_ass" + _
"oc_index_year " + _
"FROM ZZ_PLACE WHERE (((ZZ_PLACE.c_rel_type)='Kinship')) OR (((ZZ_PLACE.c_rel_type)='Associate Pl" + _
"ace')))"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
' now copy to create the nodes table
'
tStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year ) " + _
"SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id, ZZ_SOCIAL_NETWORK.c_name, ZZ_SOCIAL_NETWORK.c_nam" + _
"e_chn, ZZ_SOCIAL_NETWORK.c_index_year " + _
"FROM ZZ_SOCIAL_NETWORK"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
' to get the edges, just copy the relevant records into ZZ_SOCIAL_NETWORK
'
cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
tStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_node_id, c_link_code, c_link_desc, c_link_chn" + _
" ) " + _
"SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_assoc_id, ZZ_PLACE.c_rel_code, ZZ_PLACE.c_rel_de" + _
"sc, ZZ_PLACE.c_rel_chn " + _
"FROM ZZ_PLACE WHERE (((ZZ_PLACE.c_rel_type)='Kinship')) OR (((ZZ_PLACE.c_rel_type)='Associate Pl" + _
"ace')))"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
MsgBox "Created tables"
'
' process the two tables
'
Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
tQuote = Chr(34) ' the quotation mark
'
' first the nodes: define the node data structure
tStr = "*node data"
tStream.WriteText tStr, adWriteLine
tStr = "ID index_year x_coord y_coord"
tStream.WriteText tStr, adWriteLine
'
With tRstNode
    .MoveFirst

```



```

Do While Not .EOF
    ' name = the ID of the person
    tStr = Trim(Str(!c_person_id)) + " "
    '
    ' indexyear = c_index_year INT
    If IsNull(!c_index_year) Then
        tStr = tStr + "0 "
    Else
        tStr = tStr + Trim(Str(!c_index_year)) + " "
    End If
    '
    ' x_coord
    If IsNull(!x_coord) Then
        tStr = tStr + "0 "
    Else
        tStr = tStr + Trim(Str(!x_coord)) + " "
    End If
    '
    ' y_coord
    If IsNull(!y_coord) Then
        tStr = tStr + "0 "
    Else
        tStr = tStr + Trim(Str(!y_coord)) + " "
    End If
    '
    tStream.WriteText tStr, adWriteLine
    .MoveNext
Loop
End With
'
' now the node properties
'
' Note: ACTIVE removed as a property (MAF 2018/07/22)
'
tStr = "*node properties"
tStream.WriteText tStr, adWriteLine
tStr = "ID shape size shortlabel"
tStream.WriteText tStr, adWriteLine
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' ID = the ID of the person
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' shape = 2? / size = 1?
        tStr = tStr + "2 1 "
        '
        ' shortlabel (+ Active = TRUE removed)
        If IsNull(!c_name) Then
            tStr = tStr + "[Missing]"
        Else
            tStr = tStr + tQuote + !c_name + tQuote
        End If
        tStream.WriteText tStr, adWriteLine
        .MoveNext
    Loop
End With
'
'MsgBox "wrote nodes"
' now the edges: define the record structure
'
tStr = "*tie data"
tStream.WriteText tStr, adWriteLine
tStr = "from to " + tQuote + "EdgeWeight" + tQuote + " " + tQuote + "edgedesc" + tQuote
tStream.WriteText tStr, adWriteLine
'
' For the moment, I am not combining parallel edges
'
With tRstEdge
    .MoveFirst
    Do While Not .EOF
        '
        ' From = str(c_person_id) for node1
        tStr = Trim(Str(!c_person_id)) + " "
        '
        ' to = str(c_assoc_id) for node2
        tStr = tStr + Trim(Str(!c_node_id)) + " 1 "
        '
        ' edgedesc

```

```

        '
        tStr = tStr + tQuote + Trim(!c_link_desc) + tQuote
        '
        tStream.WriteText tStr, adWriteLine
        .MoveNext
    Loop
End With
'
'MsgBox "wrote edges"
'
' now the edges properties
'
'tVNA.WriteLine ("*tie properties")
'tVNA.WriteLine ("from to color size active")

'With tRstEdge
    '.MoveFirst
    'Do While Not .EOF
        '
        '    from = str(c_person_id) for node1
        'tStr = Trim(Str(!c_person_id)) + " "
        '
        '    to = str(c_node_id) for node2
        'tStr = tStr + Trim(Str(!c_node_id)) + " 1 "
        '
        '    color = black (1), blue (2), green (3), yellow (4), orange (5)
        'tStr = tStr + tColor(!c_edge_dist)
        '
        '    size = 1? active = TRUE
        'tStr = tStr + "1 TRUE"
        '
        'tVNA.WriteLine (tStr)
        '.MoveNext
    'Loop
'End With
'
'tVNA.Close
'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tStream = Nothing
'Set tVNA = Nothing
'Set tFileSystem = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

```

```

Exit_CmdUCINet_Click:
Exit Sub

```

```

Err_CmdUCINet_Click:
MsgBox Err.Description
Resume Exit_CmdUCINet_Click

```

```
End Sub
```

```

Private Sub Form_Open(Cancel As Integer)
    Dim cmdSQL As ADODB.Command, tRecDeleted As Variant
    Dim tRstPlaceCode As DAO.Recordset, tRstDummy As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    '
    ' to clear the tables, briefly close and then delete records
    '
    Set tRstPlaceCode = frmZZZ_PLACE.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_PL", dbOpenDynaset)
    Set frmZZZ_PLACE.Form.Recordset = tRstDummy
    tRstPlaceCode.Close
    '
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText

```

```

    '
    cmdSQL.CommandText = "Delete from ZZ_PLACE where c_personid > -1"
    cmdSQL.Execute tRecDeleted
    '
    ' now reopen
    '
    Set tRstPlaceCode = CurrentDb.OpenRecordset("ZZ_PLACE", dbOpenDynaset)
    Set frmZZ_PLACE.Form.Recordset = tRstPlaceCode
    '
    ' ZZ_SCRATCH_PLACE_AGG
    '
    Set tRstPlaceCode = ZZ_SCRATCH_PLACE_AGG.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_PLACE_AGG", dbOpenDynaset)
    Set ZZ_SCRATCH_PLACE_AGG.Form.Recordset = tRstDummy
    tRstPlaceCode.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PLACE_AGG"
    cmdSQL.Execute tRecDeleted
    '
    ' now reopen
    '
    Set tRstPlaceCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PLACE_AGG", dbOpenDynaset)
    Set ZZ_SCRATCH_PLACE_AGG.Form.Recordset = tRstPlaceCode
    '
    ' ZZ_SCRATCH_PLACE_PEOPLE
    '
    Set tRstPlaceCode = ZZ_SCRATCH_PLACE_PEOPLE.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_PLACE_PEOPLE", dbOpenDynaset)
    Set ZZ_SCRATCH_PLACE_PEOPLE.Form.Recordset = tRstDummy
    tRstPlaceCode.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PLACE_PEOPLE"
    cmdSQL.Execute tRecDeleted
    '
    ' now reopen
    '
    Set tRstPlaceCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PLACE_PEOPLE", dbOpenDynaset)
    Set ZZ_SCRATCH_PLACE_PEOPLE.Form.Recordset = tRstPlaceCode
    '
    gUseADDRID = False
    gImportPlaces = False
    gFromDynasty = -1
    gToDynasty = -1
    gUseIndexYears = False
    gUseDynasties = False
End Sub

Private Sub CmdPajek_Click()
On Error GoTo Err_CmdPajek_Click
    '
    ' This program will dump the results of the search to a .net file
    '
    ' for the moment I'll just describe the format of the .gdf file
    '
    ' *Vertices NUM
    ' ID label "box" ic [color] bc [color]
    ' ID = str(c_person_id)
    ' label = c_name_chn
    ' color = red (1), orange (2), yellow (3), green (4), blue (5)
    '
    ' *Edges
    ' node1 node2 1 1 "label"
    ' node1 = str(c_person_id) for node1
    ' node2 = str(c_node_id) for node2
    ' color = red (1), orange (2), yellow (3), green (4), blue (5)
    ' label = c_link_desc
    '
    '
    ' first see if there are any records to process
    '
    If frmZZ_PLACE.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_CmdPajek_Click
    End If
    '
    ' next get a file
    '
    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant

```

```

Dim tRstNode As DAO.Recordset, tRstNodeList As DAO.Recordset
Dim tRstEdge As DAO.Recordset, tRstAssocType As DAO.Recordset
Dim tRstAssocCodeType As DAO.Recordset, tRstEdgeList As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tQuote As String, tFindStr As String
Dim tColor(20) As String, tStrNode1 As String, tStrNode2 As String, tCodeStr As String

```

```

' to write to a UTF-8 file, use the ADO stream object

```

```

Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8.net"
ElseIf CodeFrame.Value = 2 Then
    tStream.Charset = "big5"
    tCodeStr = "BIG5.net"
ElseIf CodeFrame.Value = 3 Then
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030.net"
Else
    tStream.Charset = "ascii"
    tCodeStr = "ASCII.net"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

```

```

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

```

```

'Use a With...End With block to reference the FileDialog object.

```

```

With dlgSaveAs
    .InitialFileName = "network_" + tCodeStr
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdPajek_Click
        Else
            ' make sure the file name has a net extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".net"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".net") Then
                tFileName = tFileName + ".net"
            End If
        End If
        '
        ' zap and open the scratch file
        '
        Dim cmdSQL As ADODB.Command
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
        cmdSQL.Execute tRecDeleted
        '
        ' fill the node list
        '
        ' first get the people
        '
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
            "SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_name_chn, " + _
            "1 AS c_distance, str(ZZ_PLACE.c_personid) AS c_v_num, TRUE as c_delete FROM ZZ_PLACE"

        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
        ' next get any node ID not among the people
        '
        tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK ( c_ID, c_lbl, c_distance, c_v_num, c_delete ) " + _
            "SELECT DISTINCT ZZ_PLACE.c_assoc_id, ZZ_PLACE.c_assoc_chn, 1 AS c_distance, Str([ZZ_PLACE].[c_as" & _
            "soc_id]) AS c_v_num, True AS c_delete " + _

```

```

"FROM ZZ_PLACE LEFT JOIN ZZ_SCRATCH_PAJEK ON ZZ_PLACE.c_assoc_id = ZZ_SCRATCH_PAJEK.c_ID " + _
"WHERE (((ZZ_SCRATCH_PAJEK.c_ID) Is Null))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
tRstNodeList.Index = "c_ID"
'
'   there probably is an SQL way to do this, but...
'
ti = 1
With tRstNodeList
    .MoveFirst
    Do While Not .EOF
        .Edit
        !c_v_num = Trim(Str(ti))
        .Update
        ti = ti + 1
        .MoveNext
    Loop
End With
tRstNodeList.Close
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK_EDGE"
cmdSQL.Execute tRecDeleted
'
'   fill the edge list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_PAJEK_EDGE ( c_node_1, c_node_2, c_edge_dist, c_edge_count ) " + _
"SELECT DISTINCT Val([ZZ_SCRATCH_PAJEK.c_v_num]) AS c_node_1, Val([ZZ_SCRATCH_PAJEK_1.c_v_num]) AS c_node_2, " + _
"1 AS c_edge_distance, Count(ZZ_SCRATCH_PAJEK_1.c_v_num) AS CountOfc_v_num " + _
"FROM ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 INNER JOIN (ZZ_SCRATCH_PAJEK INNER JOIN ZZ_PLACE " + _
"ON ZZ_SCRATCH_PAJEK.c_ID = ZZ_PLACE.c_personid) " + _
"ON ZZ_SCRATCH_PAJEK_1.c_ID = ZZ_PLACE.c_assoc_id " + _
"GROUP BY Val([ZZ_SCRATCH_PAJEK.c_v_num]), Val([ZZ_SCRATCH_PAJEK_1.c_v_num]), 1"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
'   *****If we are allowing parallel edges, this section is no longer relevant
'
'   now fill in the edge description.  This requires three steps
'
cmdSQL.CommandText = "DROP TABLE tmp_scratch_pajek"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "SELECT ZZ_SCRATCH_PAJEK.c_ID, Val(ZZ_SCRATCH_PAJEK.c_v_num) AS c_v_num INTO " + _
"TMP_SCRATCH_PAJEK FROM ZZ_SCRATCH_PAJEK"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
tQueryStr = "UPDATE ((ZZ_PLACE INNER JOIN TMP_SCRATCH_PAJEK ON " + _
"ZZ_PLACE.c_personid = TMP_SCRATCH_PAJEK.c_ID) " + _
"INNER JOIN ZZ_SCRATCH_PAJEK_EDGE ON " + _
"TMP_SCRATCH_PAJEK.c_v_num = ZZ_SCRATCH_PAJEK_EDGE.c_node_1) " + _
"INNER JOIN TMP_SCRATCH_PAJEK AS TMP_SCRATCH_PAJEK_1 ON (TMP_SCRATCH_PAJEK_1.c_v_num = " + _
"ZZ_SCRATCH_PAJEK_EDGE.c_node_2) AND (ZZ_PLACE.c_assoc_id = TMP_SCRATCH_PAJEK_1.c_ID) " + _
"SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = [ZZ_PLACE].[c_rel_type]+':[ZZ_PLACE].[c_rel_desc] " + _
"WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_count)=1))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "DROP TABLE tmp_scratch_pajek"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "UPDATE ZZ_SCRATCH_PAJEK_EDGE SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = " + _
"'Parallel Edges merged' WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_count)>1))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'

```

```

'MsgBox "Tables successfully built"
,
Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenDynaset)
Set tRstEdgeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK_EDGE", dbOpenDynaset)
,
'MsgBox "Tables opened"
' set the Quote delimiter
,
tQuote = Chr(34)
,
' define the colors for the nodes
,
tColor(1) = "Black"
tColor(2) = "Blue"
tColor(3) = "Green"
tColor(4) = "Yellow"
tColor(5) = "Orange"
For ti = 6 To 20
    tColor(ti) = "Red"
Next
,
tC = Chr(44) ' the comma
,
' first the nodes: define the record structure
,
tRstNodeList.MoveLast
tStr = "*Vertices " + Trim(Str(tRstNodeList.RecordCount))
tStream.WriteText tStr, adWriteLine
,
'MsgBox "header written"
,
With tRstNodeList
    .MoveFirst

    Do While Not .EOF
        tStream.WriteText !c_v_num + " "
        ,
        If IsNull(!c_lbl) Then
            tStream.WriteText Chr(34)
            tStream.WriteText "Error-" + Trim(Str(!c_ID))
            tStream.WriteText Chr(34)
            tStream.WriteText " box "
        Else
            If !c_lbl = "" Then
                tStream.WriteText Chr(34)
                tStream.WriteText "Error-" + Trim(Str(!c_ID))
                tStream.WriteText Chr(34)
                tStream.WriteText " box "
            Else
                tStream.WriteText Chr(34)
                tStream.WriteText !c_lbl
                tStream.WriteText ":" + Trim(Str(!c_ID))
                tStream.WriteText Chr(34)
                tStream.WriteText " box "
            End If
        End If
        ' label
        tStr = " ic " + tColor(!c_distance + 1)
        tStr = tStr + " bc " + tColor(!c_distance + 1)
        ' color = white (1), blue (2), green (3), yellow (4), orange (5)
        tStream.WriteText tStr, adWriteLine
        ,
        .MoveNext
    Loop
End With
,
'MsgBox "Nodes written to stream"
,
' now the edges: define the record structure
,
tStream.WriteText "*Edges", adWriteLine

If tRstEdgeList.RecordCount > 0 Then
    With tRstEdgeList
        .MoveFirst
        Do While Not .EOF
            tStr = Trim(Str(!c_node_1)) + " " + Trim(Str(!c_node_2))
            ,
            ' now get the weight
            ,

```

```

        If !c_edge_count < 6 Then
            tStr = tStr + " " + Trim(Str(!c_edge_count)) + " "
        Else
            tStr = tStr + " 5 "
        End If
        '
        ' now get the label
        '
        tStr = tStr + "l " + tQuote
        If !c_edge_count = 1 Then
            tStr = tStr + !c_edge_desc + tQuote + " "
        Else
            tStr = tStr + Trim(Str(!c_edge_count)) + " links" + tQuote + " "
        End If

        tStr = tStr + "c " + tColor(!c_edge_dist + 1)
        ' color = white (1), blue (2), green (3), yellow (4), orange (5)
        '
        tStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
End If
'
'MsgBox "Edges written to stream"
'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
'
'MsgBox "Writing to file"
'
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tRstNodeList.Close

tStream.Close
Set tStream = Nothing
'
'Set tGDF = Nothing
'Set tFileSystem = Nothing
Set tRstNodeList = Nothing
Set tRstEdgeList = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

```

```

Exit_CmdPajek_Click:
Exit Sub

```

```

Err_CmdPajek_Click:
MsgBox Err.Description
Resume Exit_CmdPajek_Click

```

```
End Sub
```

```
Private Sub CmdGephi_Click()
```

```

'
' This program will dump the results of the search to a .gdf file
'
' for the moment I'll just describe the format of the .gdf file
'
' nodedef> name, color, label, labelvisible, style, pinyin VARCHAR(50), nodedist INT
' name = str(c_person_id)
' label = c_name_chn
' style = 4 (text inside a rectangle)
' pinyin = c_name
' indexyear = c_index_year INT
'
' edgedef> node1, node2, color, label, labelvisible, edge_desc VARCHAR(50)
' node1 = str(c_person_id) for node1
' node2 = str(c_assoc_id) for node2
' label = c_rel_chn
' edge_desc = c_rel_desc
'

```

```

' the central question is whether to do distance optimizations
'
' first see if there are any records to process
'
If frmZZZ_PLACE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGUESS_Click
End If
'
' next get a file
'

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tQuote As String
Dim tMetricSum As Integer

' to write to a UTF-8 file, use the ADO stream object
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream
If GephiFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8.net"
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030.net"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "default.gdf"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdGUESS_Click
        End If
        '
        Dim cmdSQL As ADODB.Command
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        ' now prepare the node list by getting all the person ID and assoc IDs
        '
        ' the strategy is to dump both into ZZ_SOCIAL_NETWORK and then copy to ZZ_SCRATCH_PEOPLE
        '
        cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
        cmdSQL.Execute tRecDeleted
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
        cmdSQL.Execute tRecDeleted
        '
        tStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
            "SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_name, ZZ_PLACE.c_name_chn, ZZ_PLACE.c_index_year, " + _
            "ZZ_PLACE.c_addr_name, ZZ_PLACE.c_addr_chn, ZZ_PLACE.x_coord, ZZ_PLACE.y_coord " + _
            "FROM ZZ_PLACE WHERE (((ZZ_PLACE.c_rel_type)='Kinship')) OR (((ZZ_PLACE.c_rel_type)='Associate Pl" + _
            "ace')))"
        '
        cmdSQL.CommandText = tStr
        cmdSQL.Execute tRecDeleted
        '
        tStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _

```



```

"SELECT DISTINCT ZZ_PLACE.c_assoc_id, ZZ_PLACE.c_assoc_name, ZZ_PLACE.c_assoc_chn, ZZ_PLACE.c_ass
oc_index_year, " + _
"ZZ_PLACE.c_assoc_addr_name, ZZ_PLACE.c_assoc_addr_chn, ZZ_PLACE.assoc_x_coord, ZZ_PLACE.assoc_y_
coord " + _
"FROM ZZ_PLACE WHERE (((ZZ_PLACE.c_rel_type)='Kinship')) OR (((ZZ_PLACE.c_rel_type)='Associate Pl
ace')))"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
' now copy to create the nodes table
'
tStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_addr_name, c
_addr_chn, x_coord, y_coord ) " + _
"SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id, ZZ_SOCIAL_NETWORK.c_name, ZZ_SOCIAL_NETWORK.c_nam
e_chn, ZZ_SOCIAL_NETWORK.c_index_year, " + _
"ZZ_SOCIAL_NETWORK.c_addr_name, ZZ_SOCIAL_NETWORK.c_addr_chn, ZZ_SOCIAL_NETWORK.x_coord, ZZ_SOCIA
L_NETWORK.y_coord " + _
"FROM ZZ_SOCIAL_NETWORK"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
If tRecDeleted = 0 Then
    MsgBox "There are no networks associated with this place."
    Exit Sub
End If
'
' to get the edges, just copy the relevant records into ZZ_SOCIAL_NETWORK
'
cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
tStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_node_id, c_link_code, c_link_desc, c_link_chn
) " + _
"SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_assoc_id, ZZ_PLACE.c_rel_code, ZZ_PLACE.c_rel_de
sc, ZZ_PLACE.c_rel_chn " + _
"FROM ZZ_PLACE WHERE (((ZZ_PLACE.c_rel_type)='Kinship')) OR (((ZZ_PLACE.c_rel_type)='Associate Pl
ace')))"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
' process the two tables
'
Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
tC = Chr(44) ' the comma
tQuote = Chr(34) 'the Quote delimiter
'
' first the nodes: define the record structure
tStr = "nodedef> name VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible BOOLEAN" + tC + "style INT
" + tC + "pinyin VARCHAR(50)" + _
tC + "indexyear INT" + tC + "addr_name VARCHAR" + tC + "addr_chn VARCHAR" + tC + "latitude DOUBLE
" + tC + "longitude DOUBLE"
tStream.WriteText tStr, adWriteLine
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' name = the ID of the person
        tStr = Trim(Str(!c_person_id)) + tC
        '
        ' label
        If IsNull(!c_name_chn) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_name_chn + tC
        End If

        ' labelvisible = true, style = 4 (text inside a rectangle)
        tStr = tStr + "true" + tC + "4" + tC

        ' pinyin = c_name
        If IsNull(!c_name) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_name + tC
        End If
    End If
End With

```

```

        ' indexyear = c_index_year INT
    If IsNull(!c_index_year) Then
        tStr = tStr + "-2000" + tC
    Else
        tStr = tStr + Trim(Str(!c_index_year)) + tC
    End If

    ' addr_name
    If IsNull(!c_addr_name) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If

    ' addr_chn
    If IsNull(!c_addr_chn) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_chn + tC
    End If
    ' latitude = !y_coord
    If IsNull(!y_coord) Then
        tStr = tStr + "0.0" + tC
    Else
        tStr = tStr + Str(!y_coord) + tC
    End If

    ' longitude = !x_coord
    If IsNull(!x_coord) Then
        tStr = tStr + "0.0"
    Else
        tStr = tStr + Str(!x_coord)
    End If
    tStream.WriteText tStr, adWriteLine
    .MoveNext
Loop
End With
'
' now the edges: define the record structure
tStr = "edgedef> node1" + tC + "node2" + tC + "label"
tStream.WriteText tStr, adWriteLine

With tRstEdge
    .MoveFirst
    Do While Not .EOF
        tStr = Trim(Str(!c_person_id)) + tC
        ' node1 = str(c_person_id) for node1
        tStr = tStr + Trim(Str(!c_node_id)) + tC
        ' node2 = str(c_node_id) for node2
        ,
        If IsNull(!c_link_desc) Then
            tStr = tStr + tC
        Else
            tStr = tStr + tQuote + !c_link_desc + tQuote
        End If
        ' label = the association
        ,
        tStream.WriteText tStr, adWriteLine
    .MoveNext
Loop
End With
'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tStream = Nothing

cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted

Else
    'The user pressed Cancel.
End If

```

```

End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGUESS_Click:
Exit Sub

Err_CmdGUESS_Click:
MsgBox Err.Description
Resume Exit_CmdGUESS_Click

End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

If gDisplayLanguage = "T" Then
    gDisplayLanguage = "E"
Else
    gDisplayLanguage = "T"
End If

Call changeDisplayLanguage

Exit_CmdFanti_Click:
Exit Sub

Err_CmdFanti_Click:
MsgBox Err.Description
Resume Exit_CmdFanti_Click

End Sub

Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

If gDisplayLanguage = "S" Then
    gDisplayLanguage = "E"
Else
    gDisplayLanguage = "S"
End If

Call changeDisplayLanguage

Exit_CmdJianti_Click:
Exit Sub

Err_CmdJianti_Click:
MsgBox Err.Description
Resume Exit_CmdJianti_Click

End Sub

Public Sub changeDisplayLanguage()
Dim tLabelLanguage(3, 35) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False
With tRstLabelList
.MoveFirst
ti = 1

Do While ti < 35 And Not .EOF
    If !c_form = "LAP" Then
        gLabelsOK = True
        If ti <> !c_label_id Then
            MsgBox "Uh oh: mismatched label table"
            gLabelsOK = False
            Exit Do
        End If
        tLabelLanguage(1, ti) = !c_english
        tLabelLanguage(2, ti) = !c_fanti
        tLabelLanguage(3, ti) = !c_jianti
        ti = ti + 1
    End If
End With

```

```

        End If
        .MoveNext
    Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If gLabelsOK Then
    If gDisplayLanguage = "E" Then
        tLang = 1
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '
    Me.LblFrom.Caption = tLabelLanguage(tLang, 1)
    Me.LblTo.Caption = tLabelLanguage(tLang, 2)
    Me.CmdSelectPlace.Caption = tLabelLanguage(tLang, 3)
    Me.CmdImportPlaces.Caption = tLabelLanguage(tLang, 4)
    Me.CmdQuery.Caption = tLabelLanguage(tLang, 5)
    Me.CmdGephi.Caption = tLabelLanguage(tLang, 6)
    Me.CmdPajek.Caption = tLabelLanguage(tLang, 7)
    Me.CmdFanti.Caption = tLabelLanguage(tLang, 8)
    Me.CmdJianti.Caption = tLabelLanguage(tLang, 9)
    Me.LblChkXYRef.Caption = tLabelLanguage(tLang, 10)
    'Me.LblChkIndexYears.Caption = tLabelLanguage(tLang, 11)
    Me.LblChkIndividual.Caption = tLabelLanguage(tLang, 12)
    Me.LblChkInstitution.Caption = tLabelLanguage(tLang, 13)
    Me.LblChkEntry.Caption = tLabelLanguage(tLang, 14)
    Me.LblChkKin.Caption = tLabelLanguage(tLang, 15)
    Me.LblChkAssocPerson.Caption = tLabelLanguage(tLang, 16)
    Me.LblChkAssocPlace.Caption = tLabelLanguage(tLang, 17)
    Me.LblChkOffice.Caption = tLabelLanguage(tLang, 18)
    Me.CmdStoreID.Caption = tLabelLanguage(tLang, 19)
    Me.CmdUCINet.Caption = tLabelLanguage(tLang, 20)
    Me.LblDynasties.Caption = tLabelLanguage(tLang, 21)
    Me.CmdFromDynasty.Caption = tLabelLanguage(tLang, 22)
    Me.CmdToDynasty.Caption = tLabelLanguage(tLang, 23)
    Me.CmdAllDynasties.Caption = tLabelLanguage(tLang, 24)
    Me.LblIndexYears.Caption = tLabelLanguage(tLang, 25)
    Me.LblOptNoDates.Caption = tLabelLanguage(tLang, 26)
    Me.LblOptIndexYears.Caption = tLabelLanguage(tLang, 27)
    Me.LblOptDynasties.Caption = tLabelLanguage(tLang, 28)
    Me.LblChkSubUnits.Caption = tLabelLanguage(tLang, 29)
    Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 30)
    Me.PlacePage.Caption = tLabelLanguage(tLang, 31)
    Me.PagePeoplePlaceAgg.Caption = tLabelLanguage(tLang, 32)
    Me.PagePeople.Caption = tLabelLanguage(tLang, 33)
    Me.CmdPickBAC.Caption = tLabelLanguage(tLang, 34)
End If

```

End Sub

```

Private Sub QueryOK()
    If Not gUseADDRID Then
        CmdQuery.Enabled = False
        Exit Sub
    ElseIf Me.ChkAssocPerson.Value Then
        CmdQuery.Enabled = True
    ElseIf Me.ChkAssocPlace.Value Then
        CmdQuery.Enabled = True
    ElseIf Me.ChkEntry.Value Then
        CmdQuery.Enabled = True
    ElseIf Me.ChkIndividual.Value Then
        CmdQuery.Enabled = True
    ElseIf Me.ChkInstitution.Value Then
        CmdQuery.Enabled = True
    ElseIf Me.ChkKin.Value Then
        CmdQuery.Enabled = True
    ElseIf Me.ChkOffice.Value Then
        CmdQuery.Enabled = True
    Else
        CmdQuery.Enabled = False
    End If
End Sub

```

```

Private Sub CmdStoreID_Click()
    Dim cmdSQL As ADODB.Command, tRecCount As Variant

```

```

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

```

```

If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then

```

```

    ' Display message.

```

```

    If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) =
vbNo Then

```

```

        Exit Sub

```

```

    Else

```

```

        cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"

```

```

        cmdSQL.Execute tRecCount

```

```

    End If

```

```

End If

```

```

tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT DISTINCT ZZ_PLACE.c_personid FROM ZZ_PLACE"

```

```

cmdSQL.CommandText = tStrQuery

```

```

cmdSQL.Execute tRecCount

```

```

MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."

```

```

'
' update storage source

```

```

cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='Place' WHERE PersonIDSource.LineNum =1"

```

```

cmdSQL.Execute tRecCount

```

```

End Sub

```

```

Private Sub FrameFilterYears_Click()

```

```

    '

```

```

    ' the simplest approach is to turn it all off and then turn on the appropriate objects

```

```

    ' disable all

```

```

Me.CmdFromDynasty.Enabled = False

```

```

Me.CmdToDynasty.Enabled = False

```

```

Me.CmdAllDynasties.Enabled = False

```

```

Me.TxtFromDynasty.Enabled = False

```

```

Me.TxtFromDynastyPY.Enabled = False

```

```

Me.TxtToDynasty.Enabled = False

```

```

Me.TxtToDynastyPY.Enabled = False

```

```

Me.TxtFromDynasty.Locked = False

```

```

Me.TxtFromDynastyPY.Locked = False

```

```

Me.TxtToDynasty.Locked = False

```

```

Me.TxtToDynastyPY.Locked = False

```

```

Me.TxtFromYear.Enabled = False

```

```

Me.TxtToYear.Enabled = False

```

```

gUseIndexYears = False

```

```

gUseDynasties = False

```

```

If FrameFilterYears.Value = 2 Then

```

```

    ' enable index years

```

```

    Me.TxtFromYear.Enabled = True

```

```

    Me.TxtToYear.Enabled = True

```

```

    gUseIndexYears = True

```

```

ElseIf FrameFilterYears.Value = 3 Then

```

```

    ' enable dynasties

```

```

    Me.CmdFromDynasty.Enabled = True

```

```

    Me.CmdToDynasty.Enabled = True

```

```

    Me.CmdAllDynasties.Enabled = True

```

```

    Me.TxtFromDynasty.Enabled = True

```

```

    Me.TxtFromDynastyPY.Enabled = True

```

```

    Me.TxtToDynasty.Enabled = True

```

```

    Me.TxtToDynastyPY.Enabled = True

```

```

    Me.TxtFromDynasty.Locked = True

```

```

    Me.TxtFromDynastyPY.Locked = True

```

```

    Me.TxtToDynasty.Locked = True

```

```

    Me.TxtToDynastyPY.Locked = True

```

```

    gUseDynasties = True

```

```

End If

```

```

End Sub

```

```

Private Sub CmdPajek_Click_Old()

```

```

On Error GoTo Err_CmdPajek_Click_Old

```

```

    '

```

```

' This program will dump the results of the search to a .net file
'
' for the moment I'll just describe the format of the .gdf file
'
' *Vertices NUM
' ID label "box" ic [color] bc [color]
'     ID = str(c_person_id)
'     label = c_name_chn
'     color = red (1), orange (2), yellow (3), green (4), blue (5)
'
' *Edges
' nodel node2 1 1 "label"
'     nodel = str(c_person_id) for nodel
'     node2 = str(c_node_id) for node2
'     color = red (1), orange (2), yellow (3), green (4), blue (5)
'     label = c_link_desc
'
'
' first see if there are any records to process
'
If frmZZZ_PLACE.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click_Old
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstNodeList As DAO.Recordset
Dim tRstEdge As DAO.Recordset, tRstAssocType As DAO.Recordset
Dim tRstAssocCodeType As DAO.Recordset, tRstEdgeList As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tQuote As String, tFindStr As String
Dim tColor(20) As String, tStrNodel As String, tStrNode2 As String, tCodeStr As String

' to write to a UTF-8 file, use the ADO stream object
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream
If CodeFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8.net"
ElseIf CodeFrame.Value = 2 Then
    tStream.Charset = "big5"
    tCodeStr = "BIG5.net"
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030.net"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network_" + tCodeStr
    If .Show = -1 Then
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdPajek_Click_Old
        End If
        ' zap and open the scratch file
        '
        Dim cmdSQL As ADODB.Command
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
    
```

```

'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
cmdSQL.Execute tRecDeleted
'
Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
tRstNodeList.Index = "c_ID"
'
cmdSQL.CommandText = "Delete from ZZ_SCRATCH_PAJEK_EDGE where c_node_1 > -100"
cmdSQL.Execute tRecDeleted
'
Set tRstEdgeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK_EDGE", dbOpenDynaset)
'
' now prepare the node list by getting all the person ID and the assoc IDs
'
' the strategy is to dump both into ZZ_SOCIAL_NETWORK and then copy to ZZ_SCRATCH_PEOPLE
'
cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted

tStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year ) " + _
      "SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_name, ZZ_PLACE.c_name_chn, ZZ_PLACE.c_index_year " + _
      "FROM ZZ_PLACE WHERE (((ZZ_PLACE.c_rel_type)='Kinship')) OR (((ZZ_PLACE.c_rel_type)='Associate Pl" + _
      "ace')))"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
If tRecDeleted = 0 Then
    MsgBox "There are no networks associated with this place."
    Exit Sub
End If

tStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_name, c_name_chn, c_index_year ) " + _
      "SELECT DISTINCT ZZ_PLACE.c_assoc_id, ZZ_PLACE.c_assoc_name, ZZ_PLACE.c_assoc_chn, ZZ_PLACE.c_ass" + _
      "oc_index_year " + _
      "FROM ZZ_PLACE WHERE (((ZZ_PLACE.c_rel_type)='Kinship')) OR (((ZZ_PLACE.c_rel_type)='Associate Pl" + _
      "ace')))"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
' now copy to create the nodes table
'
tStr = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year ) " + _
      "SELECT DISTINCT ZZ_SOCIAL_NETWORK.c_person_id, ZZ_SOCIAL_NETWORK.c_name, ZZ_SOCIAL_NETWORK.c_nam" + _
      "e_chn, ZZ_SOCIAL_NETWORK.c_index_year " + _
      "FROM ZZ_SOCIAL_NETWORK"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
' to get the edges, just copy the relevant records into ZZ_SOCIAL_NETWORK
'
cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
tStr = "INSERT INTO ZZ_SOCIAL_NETWORK ( c_person_id, c_node_id, c_link_code, c_link_desc, c_link_chn" + _
      ") " + _
      "SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_assoc_id, ZZ_PLACE.c_rel_code, ZZ_PLACE.c_rel_de" + _
      "sc, ZZ_PLACE.c_rel_chn " + _
      "FROM ZZ_PLACE WHERE (((ZZ_PLACE.c_rel_type)='Kinship')) OR (((ZZ_PLACE.c_rel_type)='Associate Pl" + _
      "ace')))"

cmdSQL.CommandText = tStr
cmdSQL.Execute tRecDeleted
'
'
' set the Quote delimiter
'
tQuote = Chr(34)
'
' define the colors for the nodes
'
tColor(1) = "White"
tColor(2) = "Blue"
tColor(3) = "Green"

```

```

tColor(4) = "Yellow"
tColor(5) = "Orange"
For ti = 6 To 20
    tColor(ti) = "Red"
Next
'
' process the two tables
'
Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SOCIAL_NETWORK", dbOpenDynaset)
Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_PEOPLE", dbOpenDynaset)
tC = Chr(44) ' the comma
'
' first the nodes:  define the record structure
'
tStr = "*Vertices " + Trim(Str(tRstNode.RecordCount))
tStream.WriteText tStr, adWriteLine
'
ti = 1
With tRstNode
    .MoveFirst
    Do While Not .EOF
        tStream.WriteText Trim(Str(ti)) + " "
        '
        If IsNull(!c_name_chn) Then
            If !c_name = "" Or Left(!c_name, 12) = "***BAD DATA**" Then
                tStream.WriteText Chr(34)
                tStream.WriteText "Error-" + Trim(Str(!c_person_id))
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            Else
                tStream.WriteText Chr(34)
                tStream.WriteText !c_name
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            End If
        Else
            If !c_name_chn = "" Then
                If !c_name = "" Or Left(!c_name, 12) = "***BAD DATA**" Then
                    tStream.WriteText Chr(34)
                    tStream.WriteText "Error-" + Trim(Str(!c_person_id))
                    tStream.WriteText Chr(34)
                    tStream.WriteText " box ", adWriteLine
                Else
                    tStream.WriteText Chr(34)
                    tStream.WriteText !c_name
                    tStream.WriteText Chr(34)
                    tStream.WriteText " box ", adWriteLine
                End If
            Else
                tStream.WriteText Chr(34)
                tStream.WriteText !c_name_chn
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            End If
        End If
        '
        ' add the node to the list
        '
        tRstNodeList.AddNew
        tRstNodeList!c_v_num = Str(ti)
        tRstNodeList!c_ID = !c_person_id
        tRstNodeList.Update
        '
        .MoveNext
        ti = ti + 1
    Loop
End With
'
' now the edges:  define the record structure
'
tStream.WriteText "*Edges", adWriteLine

With tRstEdge
    .MoveFirst
    Do While Not .EOF
        ' the problem with the Network Workbench is that it cannot accept
        ' parallel edges, so I first accumulate all edges (adding up duplicates)
        ' and then write the scratch table to the file
        '
        ' find the vertex number of the first node

```



```

tRstNodeList.Seek "=", Str(!c_person_id)
If Not tRstNodeList.NoMatch Then
    '
    tStrNode1 = tRstNodeList!c_v_num
    '
    ' find the vertex number of the second node
    '
    tRstNodeList.Seek "=", Str(!c_node_id)
    If Not tRstNodeList.NoMatch Then
        '
        ' see if an edge already exists
        '
        tStrNode2 = tRstNodeList!c_v_num
        tStr = "c_node_1 = " + tStrNode1 + " and c_node_2 = " + tStrNode2
        tRstEdgeList.FindFirst tStr
        If tRstEdgeList.NoMatch Then
            ' look for the other way around
            tStr = "c_node_2 = " + tStrNode1 + " and c_node_1 = " + tStrNode2
            tRstEdgeList.FindFirst tStr
            If tRstEdgeList.NoMatch Then
                ' add the edge
                '
                tRstEdgeList.AddNew
                tRstEdgeList!c_node_1 = Val(tStrNode1)
                tRstEdgeList!c_node_2 = Val(tStrNode2)
                tRstEdgeList!c_edge_count = 1
                tRstEdgeList!c_edge_desc = !c_link_desc
                '
                ' process the label now
                '
                tRstEdgeList.Update
            Else
                ' update the count
                ti = tRstEdgeList!c_edge_count + 1
                '
                tRstEdgeList.Edit
                '
                tRstEdgeList!c_edge_count = ti
                tRstEdgeList.Update
            End If
        Else
            ' update the count
            ti = tRstEdgeList!c_edge_count + 1
            '
            tRstEdgeList.Edit
            '
            tRstEdgeList!c_edge_count = ti
            tRstEdgeList.Update
        End If
    End If
End If
.MoveNext
Loop
End With

If tRstEdgeList.RecordCount > 0 Then
    With tRstEdgeList
        .MoveFirst
        Do While Not .EOF
            tStr = Trim(Str(!c_node_1)) + " " + Trim(Str(!c_node_2))
            '
            ' now get the weight
            '
            If !c_edge_count < 6 Then
                tStr = tStr + " " + Trim(Str(!c_edge_count)) + " "
            Else
                tStr = tStr + " 5 "
            End If
            '
            ' now get the label
            '
            tStr = tStr + "1 " + tQuote
            If !c_edge_count = 1 Then
                tStr = tStr + !c_edge_desc + tQuote + " "
            Else
                tStr = tStr + Trim(Str(!c_edge_count)) + " links" + tQuote + " "
            End If
        End If
    End With

    tStream.WriteText tStr, adWriteLine

```

```

        .MoveNext
    Loop
End With
End If
'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tRstNodeList.Close

tStream.Close
Set tStream = Nothing
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tGDF = Nothing
Set tFileSystem = Nothing
Set tRstNodeList = Nothing
Set tRstEdgeList = Nothing

cmdSQL.CommandText = "Delete * from ZZ_SOCIAL_NETWORK"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecDeleted

Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdPajek_Click_Old:
Exit Sub

Err_CmdPajek_Click_Old:
MsgBox Err.Description
Resume Exit_CmdPajek_Click_Old

End Sub

Private Sub getAggregatedRecords()
Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tRecCount As Long

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' first, the aggregation
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PEOPLE"
cmdSQL.Execute tRecCount
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c_female
, c_dy, c_dynasty_chn, c_addr_id, c_addr_name, c_addr_chn, " + _
    "x_coord, y_coord, xy_count ) " + _
    "SELECT ZZ_PLACE.c_personid, ZZ_PLACE.c_name, ZZ_PLACE.c_name_chn, ZZ_PLACE.c_index_year, ZZ_PLACE.c_fema
le, ZZ_PLACE.c_dy, ZZ_PLACE.c_dynasty_chn, ZZ_PLACE.c_addr_id, " + _
    "ZZ_PLACE.c_addr_name, ZZ_PLACE.c_addr_chn, ZZ_PLACE.x_coord, ZZ_PLACE.y_coord, Count(ZZ_PLACE.c_p
ersonid) AS CountOfc_personid " + _
    "FROM ZZ_PLACE " + _
    "GROUP BY ZZ_PLACE.c_personid, ZZ_PLACE.c_name, ZZ_PLACE.c_name_chn, ZZ_PLACE.c_index_year, ZZ_PLACE.c_fe
male, ZZ_PLACE.c_dy, ZZ_PLACE.c_dynasty_chn, " + _
    "ZZ_PLACE.c_addr_id, ZZ_PLACE.c_addr_name, ZZ_PLACE.c_addr_chn, ZZ_PLACE.x_coord, ZZ_PLACE.y_coord
"
cmdSQL.Execute tRecCount
'
' now get the records where a person and an address appear just once
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PLACE_AGG ( c_personid, c_name, c_name_chn, c_index_year, c_fema
le, c_dy, c_dynasty, c_dynasty_chn, place_addr_id, " + _
    "place_addr_hz, place_addr_py, place_x_coord, place_y_coord, c_rel_desc, c_rel_type, c_rel_code,
c_rel_chn ) " + _
    "SELECT DISTINCT ZZ_PLACE.c_personid, ZZ_PLACE.c_name, ZZ_PLACE.c_name_chn, ZZ_PLACE.c_index_year, ZZ
_PLACE.c_female, ZZ_PLACE.c_dy, ZZ_PLACE.c_dynasty, " + _

```

```

        "ZZ_PLACE.c_dynasty_chn, ZZ_PLACE.c_addr_id, ZZ_PLACE.c_addr_chn, ZZ_PLACE.c_addr_name, ZZ_PLACE.
x_coord, ZZ_PLACE.y_coord, ZZ_PLACE.c_rel_desc, " +
        "ZZ_PLACE.c_rel_type, ZZ_PLACE.c_rel_code, ZZ_PLACE.c_rel_chn " +
        "FROM ZZ_PLACE INNER JOIN ZZ_SCRATCH_PEOPLE ON (ZZ_PLACE.c_addr_id = ZZ_SCRATCH_PEOPLE.c_addr_id) AND
(ZZ_PLACE.c_personid = ZZ_SCRATCH_PEOPLE.c_personid) " +
        "WHERE (((ZZ_SCRATCH_PEOPLE.xy_count)=1))"
cmdSQL.Execute tRecCount
'
' now get the rest
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PLACE_AGG ( c_personid, c_name, c_name_chn, c_index_year, c_fema
le, c_dy, c_dynasty, c_dynasty_chn, place_addr_id, " +
        "place_addr_hz, place_addr_py, place_x_coord, place_y_coord, c_rel_desc, c_rel_type, c_rel_code,
c_rel_chn ) " +
        "SELECT DISTINCT ZZ_SCRATCH_PEOPLE.c_personid, ZZ_SCRATCH_PEOPLE.c_name, ZZ_SCRATCH_PEOPLE.c_name_chn,
ZZ_SCRATCH_PEOPLE.c_index_year, ZZ_SCRATCH_PEOPLE.c_female, " +
        "ZZ_SCRATCH_PEOPLE.c_dy, ZZ_SCRATCH_PEOPLE.c_dynasty, ZZ_SCRATCH_PEOPLE.c_dynasty_chn, ZZ_SCRATCH
_PEOPLE.c_addr_id, ZZ_SCRATCH_PEOPLE.c_addr_chn, " +
        "ZZ_SCRATCH_PEOPLE.c_addr_name, ZZ_SCRATCH_PEOPLE.x_coord, ZZ_SCRATCH_PEOPLE.y_coord, 'Multiple R
elations' AS rel_desc, 'Multiple' AS rel_type, 0 AS rel_code, " +
        """" + ChrW(22810) + ChrW(31278) + ChrW(38365) + ChrW(20418) + "" AS c_rel_chn " +
        "FROM ZZ_SCRATCH_PEOPLE " +
        "WHERE (((ZZ_SCRATCH_PEOPLE.xy_count)>1))"
cmdSQL.Execute tRecCount
'
' get the xy count
'
If tRecCount > 0 Then
'
        cmdSQL.CommandText = "Delete * from tmpXY"
        cmdSQL.Execute tRecDeleted
'
        cmdSQL.CommandText = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " +
        "SELECT ZZ_SCRATCH_PLACE_AGG.place_x_coord, ZZ_SCRATCH_PLACE_AGG.place_y_coord, Count(ZZ_SCRATCH_PLAC
E_AGG.place_x_coord) AS CountOfx_coord, " +
        "Count(ZZ_SCRATCH_PLACE_AGG.place_y_coord) AS CountOfy_coord " +
        "FROM ZZ_SCRATCH_PLACE_AGG " +
        "GROUP BY ZZ_SCRATCH_PLACE_AGG.place_x_coord, ZZ_SCRATCH_PLACE_AGG.place_y_coord"
'
        cmdSQL.Execute tRecCount
'
        cmdSQL.CommandText = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_PLACE_AGG ON (tmpXY.y_coord = ZZ_SCRATCH_PLACE_A
GG.place_y_coord) " +
        "AND (tmpXY.x_coord = ZZ_SCRATCH_PLACE_AGG.place_x_coord) " +
        "SET ZZ_SCRATCH_PLACE_AGG.xy_count = [tmpXY].[CountOfx_coord]"
'
        cmdSQL.Execute tRecCount
End If
'
' now fill in the addition index address data
'
cmdSQL.CommandText = "UPDATE ZZ_SCRATCH_PLACE_AGG INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_PLACE_AGG.c_personid
= ZZZ_BIOG_MAIN.c_personid " +
        "SET ZZ_SCRATCH_PLACE_AGG.c_index_addr_id = [ZZZ_BIOG_MAIN].[c_index_addr_id], " +
        "ZZ_SCRATCH_PLACE_AGG.c_index_addr_py = [ZZZ_BIOG_MAIN].[c_index_addr_name], " +
        "ZZ_SCRATCH_PLACE_AGG.c_index_addr_hz = [ZZZ_BIOG_MAIN].[c_index_addr_chn], " +
        "ZZ_SCRATCH_PLACE_AGG.index_addr_x_coord = [ZZZ_BIOG_MAIN].[x_coord], " +
        "ZZ_SCRATCH_PLACE_AGG.index_addr_y_coord = [ZZZ_BIOG_MAIN].[y_coord]"
cmdSQL.Execute tRecCount

Set tRst = CurrentDb.OpenRecordset("ZZ_SCRATCH_PLACE_AGG", dbOpenDynaset)
Set ZZ_SCRATCH_PLACE_AGG.Form.Recordset = tRst

End Sub

Private Sub getPeopleRecords()
Dim cmdSQL As ADODB.Command, tRst As DAO.Recordset, tRecCount As Long

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' distinct people
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PLACE_PEOPLE ( c_person_id, c_name, c_name_chn, c_index_year, c
female, c_dy, c_dynasty, c_dynasty_chn, " +
        "c_index_addr_id, c_index_addr_py, c_index_addr_hz, x_coord, y_coord ) " +
        "SELECT DISTINCT ZZ_SCRATCH_PLACE_AGG.c_personid, ZZ_SCRATCH_PLACE_AGG.c_name, ZZ_SCRATCH_PLACE_AGG.c_nam
e_chn, ZZ_SCRATCH_PLACE_AGG.c_index_year, " +
        "ZZ_SCRATCH_PLACE_AGG.c_female, ZZ_SCRATCH_PLACE_AGG.c_dy, ZZ_SCRATCH_PLACE_AGG.c_dynasty, ZZ_SCRATCH
_PLACE_AGG.c_dynasty_chn, ZZ_SCRATCH_PLACE_AGG.c_index_addr_id, " +

```

```

        "ZZ_SCRATCH_PLACE_AGG.c_index_addr_py, ZZ_SCRATCH_PLACE_AGG.c_index_addr_hz, ZZ_SCRATCH_PLACE_AGG.index_addr_x_coord, ZZ_SCRATCH_PLACE_AGG.index_addr_y_coord " + _
        "FROM ZZ_SCRATCH_PLACE_AGG"
cmdSQL.Execute tRecCount

If tRecCount > 0 Then
    cmdSQL.CommandText = "Delete * from tmpXY"
    cmdSQL.Execute tRecDeleted
    cmdSQL.CommandText = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
        "SELECT ZZ_SCRATCH_PLACE_PEOPLE.x_coord, ZZ_SCRATCH_PLACE_PEOPLE.y_coord, Count(ZZ_SCRATCH_PLACE_PEOPLE.x_coord) AS CountOfx_coord, " + _
        "Count(ZZ_SCRATCH_PLACE_PEOPLE.y_coord) AS CountOfy_coord " + _
        "FROM ZZ_SCRATCH_PLACE_PEOPLE " + _
        "GROUP BY ZZ_SCRATCH_PLACE_PEOPLE.x_coord, ZZ_SCRATCH_PLACE_PEOPLE.y_coord"
    cmdSQL.Execute tRecCount
    cmdSQL.CommandText = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_PLACE_PEOPLE ON (tmpXY.y_coord = ZZ_SCRATCH_PLACE_PEOPLE.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_PLACE_PEOPLE.x_coord) " + _
        "SET ZZ_SCRATCH_PLACE_PEOPLE.xy_count = [tmpXY].[CountOfx_coord]"
    cmdSQL.Execute tRecCount
End If

Set tRst = CurrentDb.OpenRecordset("ZZ_SCRATCH_PLACE_PEOPLE", dbOpenDynaset)
Set ZZ_SCRATCH_PLACE_PEOPLE.Form.Recordset = tRst
End Sub

```

Option Compare Database

Public gRstPeople As DAO.Recordset, gDisplayLanguage As String, gLabelsOK As Boolean

Public gImportPlaces As Boolean, gUseADDRID As Boolean, gFromStr As String, gToStr As String

Public gStatusCodeStr As String, gStatusTypeStr As String, gFromDynasty As Integer, gToDynasty As Integer

Public gFromDynastyBegin As Integer, gFromDynastyEnd As Integer, gToDynastyBegin As Integer, gToDynastyEnd As Integer

Private Sub ChkIndexYears_Click()

If TxtFromYear.Enabled Then

TxtFromYear.Enabled = False

TxtToYear.Enabled = False

Else

TxtFromYear.Enabled = True

TxtToYear.Enabled = True

End If

End Sub

Private Sub CmdGephi_Click()

On Error GoTo Err_CmdGephi_Click

'

' This program will dump the results of the search to a .gdf file

'

' for the moment I'll just describe the format of the .gdf file

'

' nodedef> name, label, labelvisible, style, pinyin VARCHAR(50), nodedist INT

'

' name = str(c_person_id)

' label = c_name_chn

' style = 4 (text inside a rectangle)

' pinyin = c_name

' nodedist = c_node_dist INT

' indexyear = c_index_year INT

' sex = c_female > (F,M)

'

' edgedef> node1, node2, label, labelvisible, edge_desc VARCHAR(50)

' node1 = str(c_person_id) for node1

' node2 = str(c_node_id) for node2

' label = c_link_chn

' edge_desc = c_link_desc

' edgetype= c_link_type (K,N)

'

' the central question is whether to do distance optimizations

'

' first see if there are any records to process

'

If ZZ_SCRATCH_STATUS.Form.Recordset.RecordCount = 0 Then

MsgBox "There are no records to save."

GoTo Exit_CmdGephi_Click

End If

'

If ZZ_SCRATCH_P_STATUS.Form.Recordset.RecordCount = 0 Then

MsgBox "There are no records to save."

GoTo Exit_CmdGephi_Click

End If

'

' next get a file

'

Dim dlgSaveAs As FileDialog

Dim tFileNum As Integer

Dim tFileName As String, tFN As Variant

Dim tRstNode As DAO.Recordset

Dim tRstEdge As DAO.Recordset

Dim tStr As String, tC As String, ti As Integer, tCodeStr As String

'Dim tFileSystem, tGDF

' to write to a UTF-8 file, use the ADO stream object

Dim tStream As ADODB.Stream

Set tStream = New ADODB.Stream

tPinyin = False

If CodeFrame.Value = 1 Then

tStream.Charset = "utf-8"

tCodeStr = "UTF8"

ElseIf CodeFrame.Value = 2 Then

tStream.Charset = "big5"

tCodeStr = "BIG5"

ElseIf CodeFrame.Value = 3 Then

tStream.Charset = "gb18030"

tCodeStr = "GB18030"

Else

```

tStream.Charset = "ascii"
tCodeStr = "ASCII"
tPinyin = True
End If

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
.InitialFileName = "status_" + tCodeStr + ".gdf"
If .Show = -1 Then
    '
    tFileName = ""
    For Each tFN In .SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdGephi_Click
    Else
        ' make sure the file name has a gdf extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".gdf"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".gdf") Then
            tFileName = tFileName + ".gdf"
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    'Set tFileSystem = CreateObject("Scripting.FileSystemObject")
    'Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)
    '
    tStream.Mode = adModeReadWrite
    tStream.Type = adTypeText
    tStream.Open

    ' process the two tables
    '
    Set tRstEdge = ZZ_SCRATCH_STATUS.Form.Recordset
    Set tRstNode = ZZ_SCRATCH_P_STATUS.Form.Recordset
    tC = Chr(44) ' the comma
    tQuote = Chr(34) 'the Quote delimiter
    '
    ' first the nodes: define the record structure
    ' if ASCII, no pinyin field, no characters
    '
    If tCodeStr = "ASCII" Then
        tStr = "nodedef> name VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible BOOLEAN" + _
            tC + "style INT" + tC + "indexyear INT" + tC + "sex VARCHAR(1)" + _
            tC + "addr_name VARCHAR" + tC + "latitude DOUBLE" + tC + "longitude DOUBLE"
    Else
        tStr = "nodedef> name VARCHAR" + tC + "label VARCHAR" + tC + "labelvisible BOOLEAN" + _
            tC + "style INT" + tC + "pinyin VARCHAR(50)" + tC + "indexyear INT" + tC + "sex VARCHAR(1)" + _
            tC + "addr_chn VARCHAR" + tC + "addr_name VARCHAR" + tC + "latitude DOUBLE" + tC + "longitude
DOUBLE"
    End If
    tStream.WriteText tStr, adWriteLine
    'tGDF.WriteLine (tStr)
    '
    With tRstNode
        .MoveFirst
        Do While Not .EOF
            ' name = the ID of the person
            tStr = Trim(Str(!c_person_id)) + tC

            ' label
            If tCodeStr = "ASCII" Then
                If IsNull(!c_name) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
            Else
                If IsNull(!c_name_chn) Then

```

```

        tStr = tStr + tC
    Else
        tStr = tStr + !c_name_chn + tC
    End If
End If
' labelvisible = true, style = 4 (text inside a rectangle)
tStr = tStr + "true" + tC + "4" + tC

If Not (tCodeStr = "ASCII") Then
    ' pinyin = c_name
    tStr = tStr + !c_name + tC
End If

' indexyear = c_index_year INT
If IsNull(!c_index_year) Then
    tStr = tStr + "-2000" + tC
Else
    tStr = tStr + Trim(Str(!c_index_year)) + tC
End If

' sex = F,M
tStr = tStr + !c_sex + tC

' address name(s)
If tCodeStr = "ASCII" Then
    If IsNull(!c_addr_name) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If
Else
    If IsNull(!c_addr_chn) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_chn + tC
    End If
    If IsNull(!c_addr_name) Then
        tStr = tStr + tC
    Else
        tStr = tStr + !c_addr_name + tC
    End If
End If
'
' latitude = !y_coord
If IsNull(!y_coord) Then
    tStr = tStr + "0.0" + tC
Else
    tStr = tStr + Str(!y_coord) + tC
End If

' longitude = !x_coord
If IsNull(!x_coord) Then
    tStr = tStr + "0.0"
Else
    tStr = tStr + Str(!x_coord)
End If
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)
.MoveNext
Loop
End With
'
' now the edges: define the record structure
' if ASCII, the label is the status_desc and there is not edge_desc
'
If tCodeStr = "ASCII" Then
    tStr = "edgedef> node1 VARCHAR" + tC + "node2 VARCHAR" + tC + "label VARCHAR(50)"
Else
    tStr = "edgedef> node1 VARCHAR" + tC + "node2 VARCHAR" + tC + "label VARCHAR" + tC + "edge_desc V
VARCHAR(50)"
End If
tStream.WriteText tStr, adWriteLine
'tGDF.WriteLine (tStr)

With tRstEdge
.MoveNext
Do While Not .EOF
    ' node1 = str(c_person_id) for node1
    tStr = Trim(Str(!c_person_id)) + tC

```

```

        ' node2 = str(c_status_id) for node2
        tStr = tStr + Trim(Str(!c_status_id)) + tC

        ' label
        If tCodeStr = "ASCII" Then
            If IsNull(!c_status_desc) Then
                tStr = tStr + tQuote + "[none]" + tQuote
            Else
                tStr = tStr + tQuote + Trim(Left(!c_status_desc + Space(50), 50)) + tQuote
            End If
        Else
            If IsNull(!c_status_desc_chn) Then
                tStr = tStr + tC
            Else
                tStr = tStr + tQuote + !c_status_desc_chn + tQuote + tC
            End If
        End If

        If Not (tCodeStr = "ASCII") Then
            ' edge_desc = c_link_desc
            If IsNull(!c_status_desc) Then
                tStr = tStr + tQuote + "[none]" + tQuote
            Else
                tStr = tStr + tQuote + Trim(Left(!c_status_desc + Space(50), 50)) + tQuote
            End If
        End If

        tStream.WriteText tStr, adWriteLine
        'tGDF.WriteLine (tStr)
        .MoveNext
    Loop
End With

'
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tStream.Close
Set tStream = Nothing
'
'tGDF.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
'Set tGDF = Nothing
'Set tFileSystem = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGephi_Click:
Exit Sub

Err_CmdGephi_Click:
MsgBox Err.Description
Resume Exit_CmdGephi_Click

End Sub

Private Sub CmdAllDynasties_Click()
    gFromDynasty = -2
    gToDynasty = -2
    TxtFromDynasty.Value = ""
    TxtFromDynastyPY.Value = "All"
    TxtToDynasty.Value = ""
    TxtToDynastyPY.Value = "All"
End Sub

Private Sub CmdFromDynasty_Click()
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strFromDynasty As String

    If gFromDynasty < 0 Then
        strFromDynasty = ""
    
```


frm_LookAtStatus - 5

```
Else
    strFromDynasty = Str(gFromDynasty)
End If

stDocName = "frmPickDynasty"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

    Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
    gFromDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
    gFromDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
    gFromDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
    '
    ' check to see if we have a problem and reject selection
    '
    If gToDynasty > -1 Then
        If gFromDynastyBegin > gToDynastyEnd Then
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
            gFromDynasty = -1
            TxtFromDynasty.Value = ""
            TxtFromDynastyPY.Value = ""
        End If
    End If
    '
    ' value is OK
    '
    If gFromDynasty > -1 Then
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
        TxtFromDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
        TxtFromDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
    End If

    DoCmd.Close acForm, stDocName
    '
    ' reset ToDynasty if necessary (-2 = all dynasties)
    '
    If gToDynasty = -2 Then
        gToDynasty = -1
        TxtToDynasty.Value = ""
        TxtToDynastyPY.Value = ""
    End If
End If

End Sub

Private Sub CmdImportStatusCodes_Click()
On Error GoTo Err_CmdImportStatusCodes_Click

Dim stDocName As String, tRstStatus As DAO.Recordset
Dim stLinkCriteria As String, tRstImportStatusCodes As DAO.Recordset
Dim tString As String, tOfficeID As Long, ti As Integer, tStrID As String, tQuit As Boolean
Dim tLen As Integer, cmdSQL As ADODB.Command

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tFileSystem, tList

' first see if we already have a list

tQuit = False
'
If Not tQuit Then
    ' open the list

    Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

    'Use a With...End With block to reference the FileDialog object.
    With dlgSaveAs
        .InitialFileName = ""
        If .Show = -1 Then
```

```

    '
    tFileName = ""
    For Each tFN In .SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdImportStatusCodes_Click
    End If
End If
End With
'
' Clear the address table now that we are ready to go
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_STATUS_CODE"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "StatusCodeListImport Specification", "TempImportList", tFileName, 0
'   TransferType=acImportDelim
'   SpecificationName = "TempImportList" (apparently it is saved in the database itself)
'   TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
'   HasFieldNames = False (0)
'
'   copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
    "FROM STATUS_CODES RIGHT JOIN TempImportList ON STATUS_CODES.c_status_code = TempImportList.ImportID"
" + _
    "WHERE (((STATUS_CODES.c_status_code) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If
'
'   copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_STATUS_CODE ( c_status_code ) SELECT DISTINCT TempImportList.ImportID " + _
    "FROM STATUS_CODES INNER JOIN TempImportList ON STATUS_CODES.c_status_code = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

Me.TxtTypeDesc.Value = ""
Me.TxtTypeChn.Value = ""
If tRecDeleted > 0 Then
    Me.TxtStatusDesc.Value = "[Imported List]"
    Me.TxtStatusChn.Value = "[Imported List]"
    Me.CmdQuery.Enabled = True
    Me.CmdSaveStatusCodes.Enabled = True
Else
    Me.TxtStatusDesc.Value = ""
    Me.TxtStatusChn.Value = ""
    Me.CmdQuery.Enabled = False
    Me.CmdSaveStatusCodes.Enabled = False
End If

Set cmdSQL = Nothing
End If

Exit_CmdImportStatusCodes_Click:
Exit Sub

Err_CmdImportStatusCodes_Click:
MsgBox Err.Description

```

```
Resume Exit_CmdImportStatusCodes_Click
```

```
End Sub
```

```
Private Sub CmdNeo4j_Click()
On Error GoTo Err_CmdNeo4j_Click
'
' This program will dump the results of the search to five CSV files
'
' warn the user that a lot of files will be created
'
' MsgBox "Neo4j requires that from 6 to 9 files be created."
'
' allocate the file variables
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
'
' next get the People file
'
Dim tRstPeople As DAO.Recordset, tRstStatus As DAO.Recordset, tRstStatusCodes As DAO.Recordset, tRstPlace As
DAO.Recordset
Dim tRstPeopleStatus As DAO.Recordset, tRstPeoplePlace As DAO.Recordset, tStr As String, tC As String
Dim tQueryStr As String, tPersonID As Long
Dim gStream As ADODB.Stream, tCodeStr As String
'
' set up the stream to write to

Set gStream = New ADODB.Stream
'
If GISFrame.Value = 1 Then
    gStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    gStream.Charset = "big5"
    tCodeStr = "BIG5"
ElseIf GISFrame.Value = 3 Then
    gStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    gStream.Charset = "ascii"
    tCodeStr = "ascii"
End If
'
tC = Chr(44) ' the comma
'
' prepare the temp tables for the people, place, peoplePlace and entry data

Dim cmdSQL As ADODB.Command
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
Set tRstPeopleStatus = CurrentDb.OpenRecordset("ZZ_SCRATCH_STATUS", dbOpenDynaset)
Set tRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_STATUS", dbOpenDynaset)

' Open the People file

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
'
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
End If
```

```

End If
'
' now process the file (second true removed to make ASCII)
'
' we have a file name: now open the stream for writing

gStream.Mode = adModeReadWrite
gStream.Type = adTypeText
gStream.Open

tRstPeople.MoveLast
'
' process the four tables
'
' first the nodes: define the record structure
'
' if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a pin
y in field
If tCodeStr = "ascii" Then
    tStr = "NameID" + tC + "NamePY" + tC + "IndexYear" + tC + "Dynasty" + tC + "Sex"
Else
    tStr = "NameID" + tC + "NameHZ" + tC + "NamePY" + tC + "IndexYear" + tC + "Dynasty" + tC + "Sex"
End If
gStream.WriteText tStr, adWriteLine
'
With tRstPeople
    .MoveFirst
    Do While Not .EOF
        ' the ID of the person
        tStr = Trim(Str(!c_person_id)) + tC
        '
        ' name
        '
        If tCodeStr = "ascii" Then
            If IsNull(!c_name) Then
                tStr = tStr + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        Else
            If IsNull(!c_name_chn) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If

            If IsNull(!c_name) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + !c_name + tC
            End If
        End If

        ' indexyear = c_index_year INT
        '
        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Trim(Str(!c_index_year)) + tC
        End If

        ' dynasty information
        '
        If IsNull(!c_dynasty) Then
            tStr = tStr + "unknown" + tC
        Else
            If tCodeStr = "ascii" Then
                tStr = tStr + !c_dynasty + tC
            Else
                tStr = tStr + !c_dynasty_chn + tC
            End If
        End If

        '
        If IsNull(!c_sex) Then
            tStr = tStr + "Missing"
        Else
            tStr = tStr + !c_sex
        End If
        '
        gStream.WriteText tStr, adWriteLine
    End Do
End With

```

```

        '
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now the PeopleEntry file
'
dlgSaveAs.InitialFileName = "PeopleStatus_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    tStr = "NameID" + tC + "StatusCode" + tC + "FirstYear" + tC + "LastYear"
    gStream.WriteText tStr, adWriteLine
    '
    With tRstPeopleStatus
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_personid)) + tC
            '
            ' entry code
            '
            If IsNull(!c_status_code) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_status_code)) + tC
            End If
            '
            ' first year
            '
            If IsNull(!c_firstyear) Then
                tStr = tStr + "0" + tC
            Else
                tStr = tStr + Trim(Str(!c_firstyear)) + tC
            End If
            '
            ' last year
            '
            If IsNull(!c_lastyear) Then
                tStr = tStr + "0"
            Else
                tStr = tStr + Trim(Str(!c_lastyear))
            End If
            '
            gStream.WriteText tStr, adWriteLine
            '
            .MoveNext
        Loop
    End With

```

```

        End With
        ' now make sure all the data is copied to tStream
        gStream.Flush
        ' and write the stream to the file
        gStream.SaveToFile tFileName, adSaveCreateOverWrite
    ,
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
,
' now places
,
' get a file name
,
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    ,
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    ,
    gStream.Open
    ,
    ' now process the file
    ,
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_STATUS.c_addr_id, ZZ_SCRATCH_P_STATUS.c_addr_name, ZZ_SCRATCH_P_
_STATUS.c_addr_chn, " + _
        "ZZ_SCRATCH_P_STATUS.x_coord, ZZ_SCRATCH_P_STATUS.y_coord " + _
        "FROM ZZ_SCRATCH_P_STATUS " + _
        "WHERE (ZZ_SCRATCH_P_STATUS.c_addr_id > 0)"
    ,
    Set tRstPlace = CurrentDb.OpenRecordset(tQueryStr, dbOpenDynaset)
    ,
    If tCodeStr = "ascii" Then
        tStr = "PlaceID" + tC + "PlacePY" + tC + "PlaceX" + tC + "PlaceY"
    Else
        tStr = "PlaceID" + tC + "PlacePY" + tC + "PlaceHZ" + tC + "PlaceX" + tC + "PlaceY"
    End If
    gStream.WriteText tStr, adWriteLine
    With tRstPlace
        .MoveFirst
        Do While Not .EOF
            ' the ID of the place
            If Not IsNull(!c_addr_id) Then
                tStr = Trim(Str(!c_addr_id)) + tC
                ,
                ' address name

                If IsNull(!c_addr_name) Then
                    tStr = tStr + "unknown" + tC
                Else
                    tStr = tStr + !c_addr_name + tC
                End If
                ,
                If Not (tCodeStr = "ascii") Then
                    If IsNull(!c_addr_chn) Then
                        tStr = tStr + "unknown" + tC
                    Else
                        tStr = tStr + !c_addr_chn + tC
                    End If
                End If
            End If

            If IsNull(!x_coord) Then

```

```

        tStr = tStr + "0.0" + tC
    Else
        tStr = tStr + Str(!x_coord) + tC
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + "0.0"
    Else
        tStr = tStr + Str(!y_coord)
    End If
    '
    gStream.WriteText tStr, adWriteLine
End If
.MoveNext

Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces
'
dlgSaveAs.InitialFileName = "PeoplePlaces_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_STATUS.c_person_id, ZZZ_BIOG_MAIN.c_index_addr_id, " + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_code " + _
        "FROM ZZ_SCRATCH_P_STATUS INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_STATUS.c_person_id = ZZZ_B"
    IOG_MAIN.c_personid " + _
        "WHERE (ZZZ_BIOG_MAIN.c_index_addr_type_code > 0)"

    Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

    tStr = "NameID" + tC + "PlaceID" + tC + "PersonPlaceCode"

    gStream.WriteText tStr, adWriteLine

    With tRstPeoplePlace
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_index_addr_id) Then
                '
                tStr = Trim(Str(!c_person_id)) + tC
                '
                tStr = tStr + Trim(Str(!c_index_addr_id)) + tC
                '
                tStr = tStr + Trim(Str(!c_index_addr_type_code))
                '
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        End While
    End With

```

```

        Loop
    End With
    '
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaceCode: use ZZ_SCRATCH_PEOPLE
'
dlgSaveAs.InitialFileName = "PeoplePlacesCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZZ_BIOG_MAIN.c_index_addr_type_code, ZZZ_BIOG_MAIN.c_index_addr_type_desc,
" + _
        "ZZZ_BIOG_MAIN.c_index_addr_type_chn " + _
        "FROM ZZ_SCRATCH_P_STATUS INNER JOIN ZZZ_BIOG_MAIN ON ZZ_SCRATCH_P_STATUS.c_person_id = ZZZ_B
IOG_MAIN.c_personid " + _
        "WHERE (ZZZ_BIOG_MAIN.c_index_addr_type_code > 0)"

    Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "personPlaceCode" + tC + "personPlaceTrans"
    Else
        tStr = "personPlaceCode" + tC + "personPlaceTrans" + tC + "personPlaceHZ"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstPeoplePlace
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_index_addr_type_code) Then
                '
                tStr = Trim(Str(!c_index_addr_type_code)) + tC
                '
                tStr = tStr + !c_index_addr_type_desc
                '
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + !c_index_addr_type_chn
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    '
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '

```



```

        gStream.Close
    Else
        'The user pressed Cancel.
        GoTo Exit_CmdNeo4j_Click
    End If
    '
    ' finally, get status codes
    '
    dlgSaveAs.InitialFileName = "StatusCode_" + tCodeStr + ".csv"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdNeo4j_Click
        Else
            ' make sure the file name has a txt extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".csv"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
                tFileName = tFileName + ".csv"
            End If
        End If
        '
        gStream.Mode = adModeReadWrite
        gStream.Type = adTypeText
        gStream.Open

        If tCodeStr = "ascii" Then
            tStr = "StatusCode" + tC + "StatusDesc"
        Else
            tStr = "StatusCode" + tC + "StatusDesc" + tC + "StatusDescHZ"
        End If
        gStream.WriteText tStr, adWriteLine
        '
        ' get the codes
        '
        tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_STATUS.c_status_code, ZZ_SCRATCH_STATUS.c_status_desc, ZZ_SCRATCH_STATUS.c_status_desc_chn " + _
            "FROM ZZ_SCRATCH_STATUS " + _
            "WHERE (ZZ_SCRATCH_STATUS.c_status_code > 0)"

        Set tRstStatusCodes = CurrentDb.OpenRecordset(tQueryStr)
        With tRstStatusCodes
            .MoveFirst
            Do While Not .EOF
                '
                tStr = Trim(Str(!c_status_code)) + tC
                '
                ' entry desc
                '
                If IsNull(!c_status_desc) Then
                    tStr = tStr + "Missing"
                Else
                    tStr = tStr + Trim(!c_status_desc)
                End If
                '
                ' kin ID
                '
                If Not (tCodeStr = "ascii") Then
                    tStr = tStr + tC + Trim(!c_status_desc_chn)
                End If
                '
                gStream.WriteText tStr, adWriteLine
                '
                .MoveNext
            Loop
        End With
        ' now make sure all the data is copied to tStream
        gStream.Flush
        ' and write the stream to the file
        gStream.SaveToFile tFileName, adSaveCreateOverWrite
        '
        gStream.Close
    End If
End Sub

```

```

Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If

MsgBox "Finished saving to Neo4j"
,
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click

End Sub

Private Sub CmdPickStatus_Click()
    On Error GoTo Err_CmdPickStatus_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strStatus As String

    TxtStatusCode.Visible = True
    TxtStatusCode.SetFocus
    strStatus = TxtStatusCode.Text

    stDocName = "frmPickStatus_multi"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strStatus

    If CurrentProject.AllForms("frmPickStatus_multi").IsLoaded Then
        Dim intStatus As Integer
        Dim strStatus_DESC As String

        Forms!frmPickStatus_multi.Form!TxtStatusID.Visible = True
        Forms!frmPickStatus_multi.Form!TxtStatusID.SetFocus
        intStatus = Forms!frmPickStatus_multi.Form!TxtStatusID.Value
        Forms!frmPickStatus_multi.Form!subTreeView.SetFocus
        Forms!frmPickStatus_multi.Form!TxtStatusID.Visible = False
        TxtStatusCode.Value = intStatus
        gStatusCodeStr = Trim(Str(intStatus))

        If TxtStatusCode.Value < 0 Then
            If TxtStatusCode.Value = -1 Then
                TxtStatusDesc.Value = "[[All]]"
                TxtStatusChn.Value = "[[All]]"
            Else
                TxtStatusDesc.Value = "[[Multi-Select]]"
                TxtStatusChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
            End If

            Forms!frmPickStatus_multi.Form!TxtTypeID.Visible = True
            Forms!frmPickStatus_multi.Form!TxtTypeID.SetFocus
            strStatus_DESC = Forms!frmPickStatus_multi.Form!TxtTypeID.Value
            Forms!frmPickStatus_multi.Form!subTreeView.SetFocus
            Forms!frmPickStatus_multi.Form!TxtTypeID.Visible = False
            TxtTypeCode.Value = strStatus_DESC
            gStatusTypeStr = Trim(strStatus_DESC)

            If TxtTypeCode.Value = "000" Then
                TxtTypeDesc.Value = "[ALL]"
                TxtTypeChn.Value = "[ALL]"
            Else
                Forms!frmPickStatus_multi.Form!TxtTypeDesc.Visible = True
                Forms!frmPickStatus_multi.Form!TxtTypeDesc.SetFocus

                strStatus_DESC = Forms!frmPickStatus_multi.Form!TxtTypeDesc.Value

                Forms!frmPickStatus_multi.Form!subTreeView.SetFocus
                Forms!frmPickStatus_multi.Form!TxtTypeDesc.Visible = False

                TxtTypeDesc.Value = strStatus_DESC

                Forms!frmPickStatus_multi.Form!TxtTypeDescChn.Visible = True
                Forms!frmPickStatus_multi.Form!TxtTypeDescChn.SetFocus

                strStatus_DESC = Forms!frmPickStatus_multi.Form!TxtTypeDescChn.Value

```

```

Forms!frmPickStatus_multi.Form!subTreeView.SetFocus
Forms!frmPickStatus_multi.Form!TxtTypeDescChn.Visible = False

    TxtTypeChn.Value = strStatus_DESC
End If
Else
Forms!frmPickStatus_multi.Form!TxtStatusDesc.Visible = True
Forms!frmPickStatus_multi.Form!TxtStatusDesc.SetFocus
strStatus_DESC = Forms!frmPickStatus_multi.Form!TxtStatusDesc.Value
Forms!frmPickStatus_multi.Form!subTreeView.SetFocus
Forms!frmPickStatus_multi.Form!TxtStatusDesc.Visible = False
TxtStatusDesc.Value = strStatus_DESC

Forms!frmPickStatus_multi.Form!TxtStatusDescChn.Visible = True
Forms!frmPickStatus_multi.Form!TxtStatusDescChn.SetFocus
strStatus_DESC = Forms!frmPickStatus_multi.Form!TxtStatusDescChn.Value
Forms!frmPickStatus_multi.Form!subTreeView.SetFocus
Forms!frmPickStatus_multi.Form!TxtStatusDescChn.Visible = False
TxtStatusChn.Value = strStatus_DESC

    TxtTypeCode.Value = ""
    TxtTypeDesc.Value = "N/A"
    TxtTypeChn.Value = "N/A"
End If

DoCmd.Close acForm, stDocName
'
CmdQuery.Enabled = True
Me.CmdSaveStatusCodes.Enabled = True
Else
CmdQuery.Enabled = False
Me.CmdSaveStatusCodes.Enabled = False
End If

CmdPickStatus.SetFocus
TxtStatusCode.Visible = False

Exit_CmdPickStatus_Click:
Exit Sub

Err_CmdPickStatus_Click:
MsgBox Err.Description
Resume Exit_CmdPickStatus_Click

End Sub

Private Sub CmdQuery_Click()
On Error GoTo Err_Run_Query

Dim rst As DAO.Recordset, tContinue As Integer
Dim tRstStatus As DAO.Recordset, tRstAddrList As DAO.Recordset, tRstDummy As DAO.Recordset
Dim tQueryInsertStr As String, tQuerySelectStr As String, tQueryFromStr As String, tQueryWhereStr As String
Dim tQueryStr As String, tRecDrop As Long, tStrWhereSQL As String
Dim tUseAddr As Boolean, tUseIndexYears As Boolean, tUseDynasties As Boolean

Dim cmdSQL As ADODB.Command, tRecCount As Long, tRecIndexYearCount As Long

Set cmdSQL = New ADODB.Command
'
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
' to clear the table, close and then delete records
'
Set tRstStatus = ZZ_SCRATCH_STATUS.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SC", dbOpenDynaset)
Set ZZ_SCRATCH_STATUS.Form.Recordset = tRstDummy
tRstStatus.Close
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_STATUS"
cmdSQL.Execute tRecCount
'
' now the people table
'
Set gRstPeople = ZZ_SCRATCH_P_STATUS.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SP", dbOpenDynaset)
Set ZZ_SCRATCH_P_STATUS.Form.Recordset = tRstDummy
gRstPeople.Close
'

```

```

cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_STATUS"
cmdSQL.Execute tRecCount
'
' see whether index years or dynasties will be used
'
If Me.FrameFilterYears.Value = 1 Then
    tUseIndexYears = False
    tUseDynasties = False
ElseIf Me.FrameFilterYears.Value = 2 Then
    tUseIndexYears = True
    tUseDynasties = False
Else
    tUseIndexYears = False
    tUseDynasties = True
End If
'
' now see if address IDs will be used.  If so, zap the scratch file and repopulate
'
' MsgBox "About to process address"
If gUseADDRID Then
    ' the strategy here is to fill the scratch file with all the relevant addresses from ZZZ_BELONGS_TO
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR"
    cmdSQL.Execute tRecCount
    '
    If ChkSubUnits.Value Then
        tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) " + _
            "SELECT DISTINCT ZZZ_BELONGS_TO.c_addr_id " + _
            "FROM ZZ_SCRATCH_ADDR_LIST INNER JOIN ZZZ_BELONGS_TO ON " + _
            "ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZZZ_BELONGS_TO.c_belongs_to"
    Else
        tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) " + _
            "SELECT DISTINCT c_addr_id " + _
            "FROM ZZ_SCRATCH_ADDR_LIST"
    End If

    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecCount
    '
    ' see if we need to use the historical XY search
    '
    If ChkXYRef.Value Then
        ' the strategy here is to dump the IDs to ZZ_ADDRESSES then copy to ZZ_SCRATCH_ADDR_LIST
        ' (I borrow ZZ_ADDRESSES from the Pick Addresses form in order to keep the initial selection
        ' of addresses for the query intact.)
        '
        ' zap the list
        '
        tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
        ' run the query
        '
        ' FrameXY.Value = 2 :: Narrow, FrameXY.Value = 1 :: Broad
        '
        If FrameXY.Value = 2 Then
            tStrWhereSQL = "WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.03) And " + _
                "(ADDR_CODES.x_coord)<=([ADDR_CODES_1].[x_coord]+0.03)) AND " + _
                "((ADDR_CODES.y_coord)>=([ADDR_CODES_1].[y_coord]-0.03) And " + _
                "(ADDR_CODES.y_coord)<=([ADDR_CODES_1].[y_coord]+0.03)))"
        Else
            tStrWhereSQL = "WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.06) And " + _
                "(ADDR_CODES.x_coord)<=([ADDR_CODES_1].[x_coord]+0.06)) AND " + _
                "((ADDR_CODES.y_coord)>=([ADDR_CODES_1].[y_coord]-0.06) And " + _
                "(ADDR_CODES.y_coord)<=([ADDR_CODES_1].[y_coord]+0.06)))"
        End If
        '
        tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id )SELECT DISTINCT ADDR_CODES.c_addr_id " + _
            "FROM ADDR_CODES, ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES AS ADDR_CODES_1 ON " + _
            "ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES_1.c_addr_id " + tStrWhereSQL

        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecDeleted
        '
        ' now get the address IDs from the initial list that have no xy coordinates
        '
        tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT ZZ_SCRATCH_ADDR.c_addr_id " + _

```

```

        "FROM ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES ON " + _
        "ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES.c_addr_id " + _
        "WHERE (((ADDR_CODES.x_coord) Is Null)) OR (((ADDR_CODES.y_coord) Is Null))"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' zap ZZ_SCRATCH_ADDR
'
tQueryStr = "DELETE * FROM ZZ_SCRATCH_ADDR"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' copy the list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id )SELECT DISTINCT ZZ_ADDRESSES.c_addr_id " + _
        "FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' zap the temporary list
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
End If

tUseAddr = True
Else
    tUseAddr = False
End If

' next build the appropriate query string

tQueryInsertStr = "INSERT INTO ZZ_SCRATCH_STATUS ( c_personid, c_status_code, c_sequence, c_index_year, c_dy,
c_dynasty, c_dynasty_chn, " + _
        "c_status_desc_chn, c_status_desc, c_name, c_name_chn, c_sex, " + _
        "c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord, c_source, c_title, c_title_chn ) "
tQuerySelectStr = "SELECT DISTINCT ZZZ_STATUS_DATA.c_personid, ZZZ_STATUS_DATA.c_status_code, ZZZ_STATUS_DATA
.c_sequence, ZZZ_STATUS_DATA.c_index_year, " + _
        "ZZZ_STATUS_DATA.c_dy, ZZZ_STATUS_DATA.c_dynasty, ZZZ_STATUS_DATA.c_dynasty_chn, " + _
        "ZZZ_STATUS_DATA.c_status_desc_chn, ZZZ_STATUS_DATA.c_status_desc, ZZZ_STATUS_DATA.c_name, ZZZ_STATUS_DAT
A.c_name_chn, ZZZ_STATUS_DATA.c_sex, " + _
        "ZZZ_STATUS_DATA.c_addr_id, ZZZ_STATUS_DATA.c_addr_name, ZZZ_STATUS_DATA.c_addr_chn, ZZZ_STATUS_DATA.x_co
ord, ZZZ_STATUS_DATA.y_coord, " + _
        "ZZZ_STATUS_DATA.c_source, ZZZ_STATUS_DATA.c_title, ZZZ_STATUS_DATA.c_title_chn "

' set the from tables with two possible joins: ZZ_SCRATCH_ADDR for addresses and DYNASTIES for dynasties
' the actual options are a bit more complicated, but the DYNASTIES and STATUS_CODE_TYPE_REL files are small
enough that the joins should not be too costly to leave as is

If tUseDynasties Then
    If tUseAddr Then
        tQueryFromStr = "FROM (STATUS_CODE_TYPE_REL INNER JOIN (DYNASTIES INNER JOIN ZZZ_STATUS_DATA ON DYNA
STIES.c_dy = ZZZ_STATUS_DATA.c_dy) ON " + _
            "STATUS_CODE_TYPE_REL.c_status_code = ZZZ_STATUS_DATA.c_status_code) INNER JOIN ZZ_SCRATCH_ADDR ON
N ZZZ_STATUS_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id "
        tQueryFromStr = "FROM DYNASTIES INNER JOIN ((ZZ_STATUS_DATA INNER JOIN ZZ_SCRATCH_ADDR ON " + _
            "ZZZ_STATUS_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id) INNER JOIN ZZ_STATUS_CODE ON " + _
            "ZZZ_STATUS_DATA.c_status_code = ZZ_STATUS_CODE.c_status_code) ON DYNASTIES.c_dy = ZZZ_STATUS_DAT
A.c_dy "
    Else
        tQueryFromStr = "FROM STATUS_CODE_TYPE_REL INNER JOIN (DYNASTIES INNER JOIN ZZZ_STATUS_DATA ON DYNAS
TIES.c_dy = ZZZ_STATUS_DATA.c_dy) ON " + _
            "STATUS_CODE_TYPE_REL.c_status_code = ZZZ_STATUS_DATA.c_status_code "
        tQueryFromStr = "FROM DYNASTIES INNER JOIN (ZZ_STATUS_DATA INNER JOIN ZZ_STATUS_CODE ON " + _
            "ZZZ_STATUS_DATA.c_status_code = ZZ_STATUS_CODE.c_status_code) ON DYNASTIES.c_dy = ZZZ_STATUS_DAT
A.c_dy "
    End If
Else
    If tUseAddr Then
        tQueryFromStr = "FROM STATUS_CODE_TYPE_REL INNER JOIN (ZZ_SCRATCH_ADDR INNER JOIN ZZZ_STATUS_DATA ON
" + _
            "ZZ_SCRATCH_ADDR.c_addr_id = ZZZ_STATUS_DATA.c_addr_id) ON STATUS_CODE_TYPE_REL.c_status_code = Z
ZZ_STATUS_DATA.c_status_code "
        tQueryFromStr = "FROM (ZZ_STATUS_DATA INNER JOIN ZZ_STATUS_CODE ON " + _
            "ZZZ_STATUS_DATA.c_status_code = ZZ_STATUS_CODE.c_status_code) INNER JOIN ZZ_SCRATCH_ADDR ON " + _
            "ZZZ_STATUS_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id "
    Else

```

```

        'tQueryFromStr = "FROM STATUS_CODE_TYPE_REL INNER JOIN ZZZ_STATUS_DATA AS ZZZ_STATUS_DATA ON " + _
            "STATUS_CODE_TYPE_REL.c_status_code = ZZZ_STATUS_DATA.c_status_code "
        tQueryFromStr = "FROM ZZZ_STATUS_DATA INNER JOIN ZZ_STATUS_CODE ON ZZZ_STATUS_DATA.c_status_code = ZZ
STATUS_CODE.c_status_code "
    End If
End If

' set the where conditions
'
' Start with the index years
'
tQueryWhereStr = ""
If tUseIndexYears Then
    '
    ' four possibilities
    '
    If gFromStr = "" And gToStr = "" Then
        tQueryWhereStr = ""
    ElseIf gFromStr = "" Then
        tQueryWhereStr = "WHERE (((ZZZ_STATUS_DATA.c_index_year)<=" + gToStr + ") "
    ElseIf gToStr = "" Then
        tQueryWhereStr = "WHERE (((ZZZ_STATUS_DATA.c_index_year)>=" + gFromStr + ") "
    Else
        tQueryWhereStr = "WHERE (((ZZZ_STATUS_DATA.c_index_year)<=" + gToStr + ") AND " + _
            "((ZZZ_STATUS_DATA.c_index_year)>=" + gFromStr + ") "
    End If
ElseIf tUseDynasties Then
    '
    ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
    '
    If gFromDynasty = -2 Then
        tQueryWhereStr = "Where (((ZZZ_STATUS_DATA.c_dy) > 0 ) "
    ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_dy) = " + Str(gFromDynasty) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
            "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    Else
        tQueryWhereStr = ""
    End If
End If

If Not (tQueryWhereStr = "") Then
    tQueryWhereStr = tQueryWhereStr + ")"
End If

' This section has been made irrelevant since all the selected codes, from 1, to multi, to all, are in ZZ_ST
STATUS_CODE
' Because I no longer rely on the status type codes, STATUS_CODE_TYPE_REL disappears from the FROM statement

'If TxtTypeDesc.Value = "[ALL]" Then
'    If Not (tQueryWhereStr = "") Then
'        tQueryWhereStr = tQueryWhereStr + ")"
'    End If
'ElseIf TxtTypeDesc.Value = "N/A" Then
'    If tQueryWhereStr = "" Then
'        tQueryWhereStr = "WHERE ((ZZZ_STATUS_DATA.c_status_code)= " + gStatusCodeStr + ")"
'    Else
'        tQueryWhereStr = tQueryWhereStr + "AND ((ZZZ_STATUS_DATA.c_status_code)= " + gStatusCodeStr + "))"
'    End If
'ElseIf Len(TxtTypeCode.Value) = 2 Then
'    If tQueryWhereStr = "" Then
'        tQueryWhereStr = "WHERE ((Left(STATUS_CODE_TYPE_REL.c_status_type_code,2))= ' " + gStatusTypeStr + "'
)"
'    Else
'        tQueryWhereStr = tQueryWhereStr + "AND ((Left(STATUS_CODE_TYPE_REL.c_status_type_code,2))= ' " + gSta
tusTypeStr + "'))"
'    End If
'Else
'    If tQueryWhereStr = "" Then
'        tQueryWhereStr = "WHERE ((STATUS_CODE_TYPE_REL.c_status_type_code)= ' " + gStatusTypeStr + "' )"
'    Else
'        tQueryWhereStr = tQueryWhereStr + "AND ((STATUS_CODE_TYPE_REL.c_status_type_code)= ' " + gStatusTypeS
tr + "'))"
'    End If
'End If

```

```

' run the query

' MsgBox tQueryInsertStr + tQuerySelectStr + tQueryFromStr + tQueryWhereStr

cmdSQL.CommandText = tQueryInsertStr + tQuerySelectStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecCount
'
' now update the index year information
'
cmdSQL.CommandText = "UPDATE ZZZ_BIOG_MAIN INNER JOIN ZZ_SCRATCH_STATUS ON ZZZ_BIOG_MAIN.c_personid = ZZ_SCRATCH_STATUS.c_personid " + _
    "SET ZZ_SCRATCH_STATUS.c_index_year_type_code = [ZZZ_BIOG_MAIN].[c_index_year_type_code], " + _
    "ZZ_SCRATCH_STATUS.c_index_year_type_desc = [ZZZ_BIOG_MAIN].[c_index_year_type_desc], " + _
    "ZZ_SCRATCH_STATUS.c_index_year_type_hz = [ZZZ_BIOG_MAIN].[c_index_year_type_hz]"
cmdSQL.Execute tRecIndexYearCount
'
' the next step is to make the table of people from the statusiations
'
If tRecCount > 0 Then
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_STATUS ( c_person_id, c_name, c_name_chn, c_index_year, c_index_year_type_code, c_index_year_type_desc, " + _
        "c_index_year_type_hz, c_dy, c_dynasty, c_dynasty_chn, c_sex, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_STATUS.c_personid, ZZ_SCRATCH_STATUS.c_name, ZZ_SCRATCH_STATUS.c_name_chn, ZZ_SCRATCH_STATUS.c_index_year, " + _
        "ZZ_SCRATCH_STATUS.c_index_year_type_code, ZZ_SCRATCH_STATUS.c_index_year_type_desc, ZZ_SCRATCH_STATUS.c_index_year_type_hz, " + _
        "ZZ_SCRATCH_STATUS.c_dy, ZZ_SCRATCH_STATUS.c_dynasty, ZZ_SCRATCH_STATUS.c_dynasty_chn, " + _
        "ZZ_SCRATCH_STATUS.c_sex, ZZ_SCRATCH_STATUS.c_addr_id, ZZ_SCRATCH_STATUS.c_addr_name, " + _
        "ZZ_SCRATCH_STATUS.c_addr_chn, ZZ_SCRATCH_STATUS.x_coord, ZZ_SCRATCH_STATUS.y_coord " + _
        "FROM ZZ_SCRATCH_STATUS"
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecCount
    '
    ' the final step is to calculate the xy_count
    '
    If tRecCount > 0 Then
        '
        cmdSQL.CommandText = "Delete * from tmpXY"
        cmdSQL.Execute tRecDeleted
        '
        tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
            "SELECT ZZ_SCRATCH_P_STATUS.x_coord, ZZ_SCRATCH_P_STATUS.y_coord, Count(ZZ_SCRATCH_P_STATUS.x_coord) AS CountOfx_coord, Count(ZZ_SCRATCH_P_STATUS.y_coord) AS CountOfy_coord " + _
            "FROM ZZ_SCRATCH_P_STATUS " + _
            "GROUP BY ZZ_SCRATCH_P_STATUS.x_coord, ZZ_SCRATCH_P_STATUS.y_coord"
        '
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecCount
        '
        tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_P_STATUS ON (tmpXY.y_coord = " + _
            "ZZ_SCRATCH_P_STATUS.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_P_STATUS.x_coord) " + _
            "SET ZZ_SCRATCH_P_STATUS.xy_count = [tmpXY].[CountOfx_coord]"
        '
        cmdSQL.CommandText = tQueryStr
        cmdSQL.Execute tRecCount
        CmdGIS.Enabled = True
        CmdNeo4j.Enabled = True
        CmdStoreID.Enabled = True
    Else
        CmdGIS.Enabled = False
        CmdNeo4j.Enabled = False
        CmdStoreID.Enabled = False
    End If
End If

Exit_Run_Query:
'
' now reopen the tables
'
Set tRstStatus = CurrentDb.OpenRecordset("ZZ_SCRATCH_STATUS", dbOpenDynaset)
Set ZZ_SCRATCH_STATUS.Form.Recordset = tRstStatus
'
Set gRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_STATUS", dbOpenDynaset)
Set ZZ_SCRATCH_P_STATUS.Form.Recordset = gRstPeople

```

```

Form_LookAtStatus - 20
'
'   close everything
'
Set rst = Nothing
Set tRstDummy = Nothing
Set cmdSQL = Nothing
Exit Sub

Err_Run_Query:
MsgBox Err.Description
Resume Exit_Run_Query

End Sub

Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
'
'   If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
    Call writeKML
    Exit Sub
End If

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tPinyin As Boolean
Dim tFileSystem, tGDF
'
'   This program will dump the results to a .gis file
'
If ZZ_SCRATCH_P_STATUS.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGIS_Click
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False
If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
'   next get a file
'

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
'
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdGIS_Click
    Else
        '   make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then

```



```

        tFileName = tFileName + ".tab"
    End If
End If
'
' write the file
'
'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
'
' process the table
'
Set tRstNode = ZZ_SCRATCH_P_STATUS.Form.Recordset
tC = Chr(9) ' the tab
'
With tRstNode
    '
    ' write the header
    '
    If tPinyin Then
        tStr = "Name" + tC + "Sex" + tC + "IndexYear" + tC + _
            "AddrName" + tC + "X" + tC + "Y" + tC + "xy_count"
    Else
        tStr = "Name" + tC + "NameChn" + tC + "Sex" + tC + "IndexYear" + tC + _
            "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC + "xy_count"
    End If
    tStream.WriteText tStr, adWriteLine
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs
        '
        If Trim(!c_name) = "" Then
            tStr = "[?]" + tC
        Else
            tStr = !c_name + tC
        End If

        If Not tPinyin Then
            If Trim(!c_name_chn) = "" Then
                tStr = tStr + "[?]" + tC
            Else
                tStr = tStr + !c_name_chn + tC
            End If
        End If

        If IsNull(!c_sex) Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_sex + tC
        End If

        If IsNull(!c_index_year) Then
            tStr = tStr + "-2000" + tC
        Else
            tStr = tStr + Str(!c_index_year) + tC
        End If

        ' here guard against blanks as well

        If IsNull(!c_addr_name) Then
            tStr = tStr + "[?]" + tC
        ElseIf Trim(!c_addr_name) = "" Then
            tStr = tStr + "[?]" + tC
        Else
            tStr = tStr + !c_addr_name + tC
        End If

        If Not tPinyin Then
            If IsNull(!c_addr_chn) Then
                tStr = tStr + "[?]" + tC
            ElseIf Trim(!c_addr_chn) = "" Then
                tStr = tStr + "[?]" + tC
            Else
                tStr = tStr + !c_addr_chn + tC
            End If
        End If

        If IsNull(!x_coord) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Str(!x_coord) + tC
        End If
    End While
End With

```

```

        If IsNull(!y_coord) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Str(!y_coord) + tC
        End If

        If IsNull(!xy_count) Then
            tStr = tStr + "0"
        Else
            tStr = tStr + Str(!xy_count)
        End If

        tStream.WriteText tStr, adWriteLine
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGIS_Click:
Exit Sub

Err_CmdGIS_Click:
MsgBox Err.Description
Resume Exit_CmdGIS_Click

End Sub

Private Sub CmdUCINet_Click()
On Error GoTo Err_CmdUCINet_Click
'
' This program will dump the results of the search to a .vna file
'
' for the moment I'll just describe the format of the .vna file
'
' *node data
' ID index_year sex x_coord y_coord nodedist
'   ID = str(c_person_id)
'   indexyear = c_index_year INT
'   nodedist = c_node_dist INT
'   sex = c_female > (F,M)
' *node properties
' ID color shape size shortlabel active
'   color = red (1), orange (2), yellow (3), green (4), blue (5)
'   shortlabel = c_name
'   shape = 2
'   active = TRUE
'
' *tie data
' from to edgetype nodedist
'   from = str(c_person_id)
'   to = str(c_node_id)
'   edgetype= c_link_type (K,N)
'
' *tie properties
' from to color size active
'   from = str(c_person_id)
'   to = str(c_node_id)
'   color = red (255), orange (26367), yellow (65535), green (32768), blue (16711680)
'   size = 1-5 (the weight)
'
' the central question is whether to do distance optimizations
'
' first see ifthere are any records to process
'

```

```

If ZZ_SCRATCH_STATUS.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUCINet_Click
End If
'
If ZZ_SCRATCH_P_STATUS.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdUCINet_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstStatusType As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tSearchStr As String
Dim tColor(20) As String, tQuote As String
Dim tFileSystem, tVNA

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

' open the status type look-up table
Set tRstStatusType = CurrentDb.OpenRecordset("STATUS_CODE_TYPE_REL", dbOpenDynaset)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network.vna"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdUCINet_Click
        Else
            ' make sure the file name has a vna extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".vna"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".vna") Then
                tFileName = tFileName + ".vna"
            End If
        End If
        '
        ' now process the file (second true removed to make ASCII)
        '
        Set tFileSystem = CreateObject("Scripting.FileSystemObject")
        Set tVNA = tFileSystem.CreateTextFile(tFileName, True)

        ' process the two tables
        '
        Set tRstEdge = CurrentDb.OpenRecordset("ZZ_SCRATCH_STATUS", dbOpenDynaset)
        Set tRstNode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_STATUS", dbOpenDynaset)
        tQuote = Chr(34) ' the quotation mark
        '
        ' first the nodes: define the node data structure
        tVNA.WriteLine ("*node data")
        tVNA.WriteLine ("ID index_year sex x_coord y_coord")
        '
        With tRstNode
            .MoveFirst
            Do While Not .EOF
                ' name = the ID of the person
                tStr = Trim(Str(!c_person_id)) + " "
                '
                ' indexyear = c_index_year INT
                If IsNull(!c_index_year) Then
                    tStr = tStr + "0 "
                Else
                    tStr = tStr + Trim(Str(!c_index_year)) + " "
                End If
                '
                ' sex = c_female > (F,M)
                If !c_sex = "F" Then

```

```

        tStr = tStr + tQuote + "F" + tQuote + " "
    Else
        tStr = tStr + tQuote + "M" + tQuote + " "
    End If
    '
    '   x_coord
    If IsNull(!x_coord) Then
        tStr = tStr + "0 "
    Else
        tStr = tStr + Trim(Str(!x_coord)) + " "
    End If
    '
    '   y_coord
    If IsNull(!y_coord) Then
        tStr = tStr + "0 "
    Else
        tStr = tStr + Trim(Str(!y_coord)) + " "
    End If
    '
    tVNA.WriteLine (tStr)
    .MoveNext
Loop
End With
'
' now the node properties
'
' Note:  ACTIVE removed as a property (MAF 2018/07/22)
'
tVNA.WriteLine ("*node properties")
tVNA.WriteLine ("ID shape size shortlabel")
'
With tRstNode
    .MoveFirst
    Do While Not .EOF
        '   ID = the ID of the person
        tStr = Trim(Str(!c_person_id)) + " "
        '
        '   shape = 2? / size = 1?
        tStr = tStr + "2 1 "
        '
        '   shortlabel (+ Active = TRUE removed)
        If IsNull(!c_name) Then
            tStr = tStr + "[Missing]"
        Else
            tStr = tStr + tQuote + !c_name + tQuote
        End If
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the edges:  define the record structure
'
tStr = "from to " + tQuote + "EdgeWeight" + tQuote + " " + tQuote + "edgedesc" + tQuote
tVNA.WriteLine ("*tie data")
tVNA.WriteLine (tStr)
'
'   For the moment, I am not combining parallel edges
'
With tRstEdge
    .MoveFirst
    Do While Not .EOF
        '
        '   From = str(c_person_id) for node1
        tStr = Trim(Str(!c_person_id)) + " "
        '
        '   to = str(c_status_id) for node2
        tStr = tStr + Trim(Str(!c_status_id)) + " 1 "
        '
        '   edgedesc
        '
        tStr = tStr + tQuote + Trim(!c_status_desc) + tQuote
        '
        tVNA.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
' now the edges properties
'

```

```

'tVNA.WriteLine ("*tie properties")
'tVNA.WriteLine ("from to color size active")

'With tRstEdge
'  .MoveFirst
'  Do While Not .EOF
'    '
'    '    from = str(c_person_id) for node1
'tStr = Trim(Str(!c_person_id)) + " "
'    '
'    '    to = str(c_node_id) for node2
'tStr = tStr + Trim(Str(!c_node_id)) + " 1 "
'    '
'    '    color = black (1), blue (2), green (3), yellow (4), orange (5)
'tStr = tStr + tColor(!c_edge_dist)
'    '
'    '    size = 1? active = TRUE
'tStr = tStr + "1 TRUE"
'    '
'tVNA.WriteLine (tStr)
'  .MoveNext
'Loop
'End With
'
tVNA.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tVNA = Nothing
Set tFileSystem = Nothing
Set tRstStatusType = Nothing
Else
'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdUCINet_Click:
Exit Sub

Err_CmdUCINet_Click:
MsgBox Err.Description
Resume Exit_CmdUCINet_Click

End Sub

Private Sub CmdSaveStatusCodes_Click()
On Error GoTo Err_CmdSaveStatusCodes_Click
'
' This program will store the current list of office IDs to a .txt file
'
Dim tStream As ADODB.Stream, tStreamNoBOM As ADODB.Stream
Set tStream = New ADODB.Stream

tStream.Charset = "utf-8"
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set tStreamNoBOM = New ADODB.Stream
tStreamNoBOM.Type = adTypeBinary
tStreamNoBOM.Open

' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstIDs As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "status_code_list.txt"
If dlgSaveAs.Show = -1 Then
'
tFileName = ""

```

```

For Each tFN In dlgSaveAs.SelectedItems
    tFileName = tFN
    If Not tFileName = "" Then
        Exit For
    End If
Next
If tFileName = "" Then
    MsgBox "Bad file Name."
    GoTo Exit_CmdSaveStatusCodes_Click
Else
    ' make sure the file name has a txt extension
    If Len(tFileName) < 5 Then
        tFileName = tFileName + ".txt"
    ElseIf Not (LCase(Right(tFileName, 4)) = ".txt") Then
        tFileName = tFileName + ".txt"
    End If
End If
'
' write the file
'
' process the table
'
tStr = "SELECT ZZ_STATUS_CODE.c_status_code, STATUS_CODES.c_status_desc, STATUS_CODES.c_status_desc_chn "
+ _
"FROM ZZ_STATUS_CODE INNER JOIN STATUS_CODES ON ZZ_STATUS_CODE.c_status_code = STATUS_CODES.c_status_"
code"
Set tRstIDs = CurrentDb.OpenRecordset(tStr, dbOpenDynaset)
'
tTab = Chr(9)
With tRstIDs

    .MoveFirst
    ' MsgBox "writing file"
    Do While Not .EOF
        tStr = Str(!c_status_code) + tTab
        If IsNull(!c_status_desc) Then
            tStr = tStr + "" + tTab
        Else
            tStr = tStr + !c_status_desc + tTab
        End If
        tStr = tStr + !c_status_desc_chn
        '
        tStream.WriteText tStr, adWriteLine
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
tStream.Position = 3
tStream.CopyTo tStreamNoBOM
' and write the stream to the file
tStreamNoBOM.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstIDs = Nothing

tStream.Close
Set tStream = Nothing
tStreamNoBOM.Close
Set tStreamNoBOM = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdSaveStatusCodes_Click:
Exit Sub

Err_CmdSaveStatusCodes_Click:
MsgBox Err.Description
Resume Exit_CmdSaveStatusCodes_Click

End Sub

Private Sub CmdToDynasty_Click()
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strToDynasty As String

```

```

If gToDynasty = -1 Then
    strToDynasty = ""
Else
    strToDynasty = Str(gToDynasty)
End If

stDocName = "frmPickDynasty"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

    Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
    gToDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
    gToDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

    Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
    gToDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
    '
    ' check to see if we have a problem and reject selection if needed
    '
    If gFromDynasty > -1 Then
        If gFromDynastyBegin > gToDynastyEnd Then
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
            gToDynasty = -1
            TxtToDynasty.Value = ""
            TxtToDynastyPY.Value = ""
        End If
    End If
    '
    ' value is OK
    '
    If gToDynasty > -1 Then
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
        TxtToDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
        TxtToDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
    End If

    DoCmd.Close acForm, stDocName
    '
    ' reset FromDynasty if necessary (-2 = all dynasties)
    '
    If gFromDynasty = -2 Then
        gFromDynasty = -1
        TxtFromDynasty.Value = ""
        TxtFromDynastyPY.Value = ""
    End If
End If

End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim cmdSQL As ADODB.Command, tRecDeleted As Variant
    Dim tRstStatusCode As DAO.Recordset, tRstDummy As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' to clear the tables, briefly close and then delete records
    '
    Set tRstStatusCode = ZZ_SCRATCH_STATUS.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SC", dbOpenDynaset)
    Set ZZ_SCRATCH_STATUS.Form.Recordset = tRstDummy
    tRstStatusCode.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_STATUS"
    cmdSQL.Execute tRecDeleted
    '
    ' now reopen
    '
    Set tRstStatusCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_STATUS", dbOpenDynaset)
    Set ZZ_SCRATCH_STATUS.Form.Recordset = tRstStatusCode
    '

```

```

Set tRstStatusCode = ZZ_SCRATCH_P_STATUS.Form.Recordset
Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SP", dbOpenDynaset)
Set ZZ_SCRATCH_P_STATUS.Form.Recordset = tRstDummy
tRstStatusCode.Close
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_STATUS"
cmdSQL.Execute tRecDeleted
'
' now reopen
'
Set tRstStatusCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_STATUS", dbOpenDynaset)
Set ZZ_SCRATCH_P_STATUS.Form.Recordset = tRstStatusCode

' first determine the language
gLCID = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
If gLCID = 2052 Or gLCID = 3076 Then ' 2052 = PRC, 3076 = Hong Kong
    gDisplayLanguage = "S"
ElseIf gLCID = 4100 Or gLCID = 1028 Then ' 4100 = Singapore, 1028 = Taiwan
    gDisplayLanguage = "T"
    Call changeDisplayLanguage
Else
    gDisplayLanguage = "E"
    Call changeDisplayLanguage
End If

' set the index year default values

gFromDynasty = -1
gToDynasty = -1
gFromStr = "-200"
gToStr = "1911"
TxtFromYear.Value = -200
TxtToYear.Value = 1911
End Sub

Private Sub CmdPajek_Click()
On Error GoTo Err_CmdPajek_Click
'
' This program will dump the results of the search to a .net file
'
' for the moment I'll just describe the format of the .gdf file
'
' *Vertices NUM
' ID label "box" ic [color] bc [color]
' ID = str(c_person_id)
' label = c_name_chn
' color = red (1), orange (2), yellow (3), green (4), blue (5)
'
' *Edges
' nodel node2 1 1 "label"
' nodel = str(c_person_id) for nodel
' node2 = str(c_node_id) for node2
' color = red (1), orange (2), yellow (3), green (4), blue (5)
' label = c_link_desc
'
'
' first see if there are any records to process
'
If ZZ_SCRATCH_STATUS.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
'
If ZZ_SCRATCH_P_STATUS.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdPajek_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset, tRstNodeList As DAO.Recordset
Dim tRstEdge As DAO.Recordset, tRstEdgeList As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tQuote As String, tFindStr As String, tPinyin As Boolean
Dim tColor(20) As String, tStrNodel As String, tStrNode2 As String, tCodeStr As String, tRecDeleted As Long

' to write to a UTF-8 file, use the ADO stream object
Dim tStream As ADODB.Stream

```



```

Set tStream = New ADODB.Stream

tPinyin = False
If CodeFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8.net"
ElseIf CodeFrame.Value = 2 Then
    tStream.Charset = "big5"
    tCodeStr = "BIG5.net"
ElseIf CodeFrame.Value = 3 Then
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030.net"
Else
    tStream.Charset = "ascii"
    tCodeStr = "ascii.net"
    tPinyin = True
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.
With dlgSaveAs
    .InitialFileName = "network_" + tCodeStr
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdPajek_Click
        Else
            ' make sure the file name has a net extension
            If Len(tFileName) < 5 Then
                tFileName = tFileName + ".net"
            ElseIf Not (LCase(Right(tFileName, 4)) = ".net") Then
                tFileName = tFileName + ".net"
            End If
        End If
        '
        ' zap and open the scratch file
        '
        Dim cmdSQL As ADODB.Command
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK"
        cmdSQL.Execute tRecDeleted
        '
        Set tRstNodeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK", dbOpenTable)
        tRstNodeList.Index = "c_ID"
        '
        cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_PAJEK_EDGE"
        cmdSQL.Execute tRecDeleted
        '
        ' set the Quote delimiter
        '
        tQuote = Chr(34)
        '
        ' define the colors for the nodes
        '
        tColor(1) = "Black"
        tColor(2) = "Blue"
        tColor(3) = "Green"
        tColor(4) = "Yellow"
        tColor(5) = "Orange"
        For ti = 6 To 20
            tColor(ti) = "Red"
        Next
        '

```

```

' process the two tables
'
Set tRstEdge = ZZ_SCRATCH_STATUS.Form.Recordset
Set tRstNode = ZZ_SCRATCH_P_STATUS.Form.Recordset
tC = Chr(44) ' the comma
'
' first the nodes:  define the record structure
'
tRstNode.MoveFirst
tStr = "*Vertices " + Trim(Str(tRstNode.RecordCount))
tStream.WriteText tStr, adWriteLine
'
ti = 1
With tRstNode
    .MoveFirst
    Do While Not .EOF
        tStream.WriteText Trim(Str(ti)) + " "
        '
        If IsNull(!c_name_chn) Then
            If !c_name = "" Or Left(!c_name, 12) = "***BAD DATA**" Then
                tStream.WriteText Chr(34)
                tStream.WriteText "Error-" + Trim(Str(!c_person_id))
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            Else
                tStream.WriteText Chr(34)
                tStream.WriteText !c_name
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            End If
        Else
            If !c_name_chn = "" Then
                If !c_name = "" Or Left(!c_name, 12) = "***BAD DATA**" Then
                    tStream.WriteText Chr(34)
                    tStream.WriteText "Error-" + Trim(Str(!c_person_id))
                    tStream.WriteText Chr(34)
                    tStream.WriteText " box ", adWriteLine
                Else
                    tStream.WriteText Chr(34)
                    tStream.WriteText !c_name
                    tStream.WriteText Chr(34)
                    tStream.WriteText " box ", adWriteLine
                End If
            Else
                tStream.WriteText Chr(34)
                If tPinyin Then
                    tStream.WriteText !c_name
                Else
                    tStream.WriteText !c_name_chn
                End If
                If ChkIDs.Value Then
                    tStream.WriteText " (" + Trim(Str(!c_person_id)) + ")"
                End If
                tStream.WriteText Chr(34)
                tStream.WriteText " box ", adWriteLine
            End If
        End If
        '
        ' add the node to the list
        '
        tRstNodeList.AddNew
        tRstNodeList!c_v_num = Str(ti)
        tRstNodeList!c_ID = !c_person_id
        tRstNodeList.Update
        '
        .MoveNext
        ti = ti + 1
    Loop
End With
tRstNodeList.Close
'
' now the edges:  define the record structure
'
tStream.WriteText "*Edges", adWriteLine
'
' first aggregate the data to a temporary table (use the edge weight to count the number of records)
'
cmdSQL.CommandText = "SELECT ZZ_SCRATCH_STATUS.c_person_id, ZZ_SCRATCH_STATUS.c_status_id, " + _
    "Count(ZZ_SCRATCH_STATUS.c_status_code) AS CountOfc_status_code, " + _
    "Sum(ZZ_SCRATCH_STATUS.c_status_count) AS SumOfc_status_count INTO tmp_pajek_edge " + _

```

```

        "FROM ZZ_SCRATCH_STATUS GROUP BY ZZ_SCRATCH_STATUS.c_person_id, ZZ_SCRATCH_STATUS.c_status_id"
cmdSQL.Execute tRecDeleted
'
' now join to the node IDs and copy to the edge table
'
cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_PAJEK_EDGE ( c_node_1, c_node_2, c_edge_weight, c_edge_c
ount, " + _
        "c_node_1_str, c_node_2_str ) " + _
        "SELECT Val([ZZ_SCRATCH_PAJEK].[c_v_num]) AS c_node_1, Val(ZZ_SCRATCH_PAJEK_1.c_v_num) AS c_node_
2, " + _
        "tmp_pajek_edge.CountOfc_status_code, tmp_pajek_edge.SumOfc_status_count, [ZZ_SCRATCH_PAJEK].[c_v
_num], " + _
        "[ZZ_SCRATCH_PAJEK_1].[c_v_num] " + _
        "FROM ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 INNER JOIN (ZZ_SCRATCH_PAJEK INNER JOIN " + _
        "tmp_pajek_edge ON ZZ_SCRATCH_PAJEK.c_ID = tmp_pajek_edge.c_person_id) " + _
        "ON ZZ_SCRATCH_PAJEK_1.c_ID = tmp_pajek_edge.c_status_id"
cmdSQL.Execute tRecDeleted
'
cmdSQL.CommandText = "DROP TABLE tmp_pajek_edge"
cmdSQL.Execute tRecDeleted
'
' now fill in the edge description.
'
If tPinyin Then
    tQueryStr = "UPDATE ((ZZ_SCRATCH_PAJEK_EDGE INNER JOIN ZZ_SCRATCH_PAJEK " + _
        "ON ZZ_SCRATCH_PAJEK_EDGE.c_node_1_str = ZZ_SCRATCH_PAJEK.c_v_num) INNER JOIN " + _
        "ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 " + _
        "ON ZZ_SCRATCH_PAJEK_EDGE.c_node_2_str = ZZ_SCRATCH_PAJEK_1.c_v_num) INNER JOIN " + _
        "ZZ_SCRATCH_STATUS ON (ZZ_SCRATCH_STATUS.c_status_id = ZZ_SCRATCH_PAJEK_1.c_ID) " + _
        "AND (ZZ_SCRATCH_PAJEK.c_ID = ZZ_SCRATCH_STATUS.c_person_id) " + _
        "SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = [ZZ_SCRATCH_STATUS].[c_status_desc] " + _
        "WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_weight)=1))"
Else
    tQueryStr = "UPDATE ((ZZ_SCRATCH_PAJEK_EDGE INNER JOIN ZZ_SCRATCH_PAJEK " + _
        "ON ZZ_SCRATCH_PAJEK_EDGE.c_node_1_str = ZZ_SCRATCH_PAJEK.c_v_num) INNER JOIN " + _
        "ZZ_SCRATCH_PAJEK AS ZZ_SCRATCH_PAJEK_1 " + _
        "ON ZZ_SCRATCH_PAJEK_EDGE.c_node_2_str = ZZ_SCRATCH_PAJEK_1.c_v_num) INNER JOIN " + _
        "ZZ_SCRATCH_STATUS ON (ZZ_SCRATCH_STATUS.c_status_id = ZZ_SCRATCH_PAJEK_1.c_ID) " + _
        "AND (ZZ_SCRATCH_PAJEK.c_ID = ZZ_SCRATCH_STATUS.c_person_id) " + _
        "SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = [ZZ_SCRATCH_STATUS].[c_status_desc] + ' ' + " + _
        "[ZZ_SCRATCH_STATUS].[c_status_desc_chn] WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_weight)=1))"
End If

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted

tQueryStr = "UPDATE ZZ_SCRATCH_PAJEK_EDGE SET ZZ_SCRATCH_PAJEK_EDGE.c_edge_desc = " + _
        "'Parallel Edges merged' WHERE (((ZZ_SCRATCH_PAJEK_EDGE.c_edge_weight)>1))"

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' open the table
'
Set tRstEdgeList = CurrentDb.OpenRecordset("ZZ_SCRATCH_PAJEK_EDGE", dbOpenDynaset)

With tRstEdgeList
    .MoveFirst
    Do While Not .EOF
        tStr = !c_node_1_str + " " + !c_node_2_str
        '
        ' now get the weight
        '
        If !c_edge_count < 6 Then
            tStr = tStr + " " + Trim(Str(!c_edge_count)) + " "
        Else
            tStr = tStr + " 5 "
        End If
        '
        ' now get the label
        '
        tStr = tStr + "1 " + tQuote
        If !c_edge_weight = 1 Then
            tStr = tStr + !c_edge_desc + tQuote + " "
        Else
            tStr = tStr + Trim(Str(!c_edge_count)) + " relations merged" + tQuote + " "
        End If

        tStream.WriteText tStr, adWriteLine
    End While
End With

```

```

        .MoveNext
    Loop
End With

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
tStream.Close
Set tStream = Nothing
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tRstNodeList = Nothing
Set tRstEdgeList = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdPajek_Click:
Exit Sub

Err_CmdPajek_Click:
MsgBox Err.Description
Resume Exit_CmdPajek_Click

End Sub

Private Sub guess_write()
'
' This program will dump the results of the search to a .gdf file
'
' for the moment I'll just describe the format of the .gdf file
'
' nodedef> name, color, label, labelvisible, style, pinyin VARCHAR(50), nodedist INT
'     name = str(c_person_id)
'     color = red (1), orange (2), yellow (3), green (4), blue (5)
'     label = c_name_chn
'     style = 4 (text inside a rectangle)
'     pinyin = c_name
'     nodedist = c_node_dist INT
'     indexyear = c_index_year INT
'     sex = c_female > (F,M)
'
' edgedef> node1, node2, color, label, labelvisible, edge_desc VARCHAR(50)
'     node1 = str(c_person_id) for node1
'     node2 = str(c_node_id) for node2
'     color = red (1), orange (2), yellow (3), green (4), blue (5)
'     label = c_link_chn
'     edge_desc = c_link_desc
'
' the central question is whether to do distance optimizations
'
' first see if there are any records to process
'
If ZZ_SCRATCH_STATUS.Form.Recordset.RecordCount = 0 Then
    MsgBox "There are no records to save."
    GoTo Exit_CmdGUESS_Click
End If
'
' next get a file
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tRstNode As DAO.Recordset
Dim tRstEdge As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer
Dim tColor(50) As String, tMetricSum As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

'Use a With...End With block to reference the FileDialog object.

```

```

With dlgSaveAs
    .InitialFileName = "default.gdf"
    If .Show = -1 Then
        '
        tFileName = ""
        For Each tFN In .SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
        If tFileName = "" Then
            MsgBox "Bad file Name."
            GoTo Exit_CmdGUESS_Click
        End If
        '
        ' now process the file (second true removed to make ASCII)
        '
        Set tFileSystem = CreateObject("Scripting.FileSystemObject")
        Set tGDF = tFileSystem.CreateTextFile(tFileName, True, True)

        ' define the colors for the nodes
        '
        tColor(1) = "white"
        tColor(2) = "blue"
        tColor(3) = "green"
        tColor(4) = "yellow"
        tColor(5) = "orange"
        For ti = 6 To 50
            tColor(ti) = "red"
        Next
        '
        ' process the two tables
        '
        Set tRstEdge = ZZ_SCRATCH_STATUS.Form.Recordset
        Set tRstNode = ZZ_SCRATCH_P_STATUS.Form.Recordset
        tC = Chr(44) ' the comma
        '
        ' first the nodes: define the record structure
        tStr = "nodedef> name" + tC + "color" + tC + "label" + tC + "labelvisible"
        tStr = tStr + tC + "style" + tC + "pinyin VARCHAR(50)"
        tStr = tStr + tC + "indexyear INT" + tC + "sex VARCHAR(1)"
        tGDF.WriteLine (tStr)
        '
        With tRstNode
            .MoveFirst
            Do While Not .EOF
                tStr = Trim(Str(!c_person_id)) + tC
                ' name = the ID of the person
                '
                tStr = tStr + tColor(1) + tC
                '
                If IsNull(!c_name_chn) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If
                ' label
                tStr = tStr + "true" + tC + "4" + tC
                ' labelvisible = true, style = 4 (text inside a rectangle)
                If IsNull(!c_name) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
                ' pinyin = c_name
                '
                If IsNull(!c_index_year) Then
                    tStr = tStr + "-2000" + tC
                Else
                    tStr = tStr + Trim(Str(!c_index_year)) + tC
                End If
                ' indexyear = c_index_year INT
                If Not IsNull(!c_sex) Then
                    tStr = tStr + !c_sex
                End If
                tGDF.WriteLine (tStr)
                .MoveNext
            Loop
        End With
    End With

```

```

'
' now the edges:  define the record structure
tStr = "edgedef> node1" + tC + "node2" + tC + "color" + tC + "label"
tGDF.WriteLine (tStr)

With tRstEdge
    .MoveFirst
    Do While Not .EOF
        tStr = Trim(Str(!c_person_id)) + tC
        ' node1 = str(c_person_id) for node1
        tStr = tStr + Trim(Str(!c_status_id)) + tC
        ' node2 = str(c_node_id) for node2
        ,
        tStr = tStr + tColor(1) + tC
        ' color = white (1), blue (2), green (3), yellow (4), orange (5)
        ,
        If IsNull(!c_status_desc) Then
            tStr = tStr + tC
        Else
            tStr = tStr + !c_status_desc
        End If
        ' label = the statusiation
        ,
        tGDF.WriteLine (tStr)
        .MoveNext
    Loop
End With
'
tGDF.Close
'
Set tRstNode = Nothing
Set tRstEdge = Nothing
Set tGDF = Nothing
Set tFileSystem = Nothing
Else
    'The user pressed Cancel.
End If
End With

'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdGUESS_Click:
Exit Sub

Err_CmdGUESS_Click:
MsgBox Err.Description
Resume Exit_CmdGUESS_Click

End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

If gDisplayLanguage = "T" Then
    gDisplayLanguage = "E"
Else
    gDisplayLanguage = "T"
End If

Call changeDisplayLanguage

Exit_CmdFanti_Click:
Exit Sub

Err_CmdFanti_Click:
MsgBox Err.Description
Resume Exit_CmdFanti_Click

End Sub

Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

If gDisplayLanguage = "S" Then
    gDisplayLanguage = "E"
Else
    gDisplayLanguage = "S"
End If

```

Call changeDisplayLanguage

Exit_CmdJianti_Click:

Exit Sub

Err_CmdJianti_Click:

MsgBox Err.Description

Resume Exit_CmdJianti_Click

End Sub

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 32) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 32 And Not .EOF

If !c_form = "LAS" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

,

' now comes the basic routine

,

Me.LblFrom.Caption = tLabelLanguage(tLang, 1)

Me.LblTo.Caption = tLabelLanguage(tLang, 2)

Me.LblType.Caption = tLabelLanguage(tLang, 3)

Me.CmdPickStatus.Caption = tLabelLanguage(tLang, 4)

Me.CmdQuery.Caption = tLabelLanguage(tLang, 5)

Me.CmdGIS.Caption = tLabelLanguage(tLang, 6)

Me.CmdFanti.Caption = tLabelLanguage(tLang, 7)

Me.CmdJianti.Caption = tLabelLanguage(tLang, 8)

Me.PageStatus.Caption = tLabelLanguage(tLang, 9)

Me.PagePeople.Caption = tLabelLanguage(tLang, 10)

Me.CmdSelectPlace.Caption = tLabelLanguage(tLang, 11)

Me.CmdImportPlaces.Caption = tLabelLanguage(tLang, 12)

Me.CmdAllPlaces.Caption = tLabelLanguage(tLang, 13)

Me.LblDisplay.Caption = tLabelLanguage(tLang, 14)

Me.CmdHelp.Caption = tLabelLanguage(tLang, 15)

Me.LblXYRef.Caption = tLabelLanguage(tLang, 16)

Me.LblNarrow.Caption = tLabelLanguage(tLang, 17)

Me.LblBroad.Caption = tLabelLanguage(tLang, 18)

Me.CmdStoreID.Caption = tLabelLanguage(tLang, 19)

Me.LblChkSubUnits.Caption = tLabelLanguage(tLang, 20)

Me.LblDynasties.Caption = tLabelLanguage(tLang, 21)

Me.CmdFromDynasty.Caption = tLabelLanguage(tLang, 22)

Me.CmdToDynasty.Caption = tLabelLanguage(tLang, 23)

Me.CmdAllDynasties.Caption = tLabelLanguage(tLang, 24)

Me.LblIndexYears.Caption = tLabelLanguage(tLang, 25)

```

Me.LblNoYears.Caption = tLabelLanguage(tLang, 26)
Me.LblUseIndexYears.Caption = tLabelLanguage(tLang, 27)
Me.LblUseDynasty.Caption = tLabelLanguage(tLang, 28)

Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 29)
Me.CmdImportStatusCodes.Caption = tLabelLanguage(tLang, 30)
Me.CmdSaveStatusCodes.Caption = tLabelLanguage(tLang, 31)
End If

End Sub

Private Sub CmdSelectPlace_Click()
On Error GoTo Err_CmdSelectPlace_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strADDR As String
Dim cmdSQL As ADODB.Command

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

TxtAddrID.Visible = True
TxtAddrID.SetFocus
strADDR = TxtAddrID.Text

stDocName = "frmPickAddresses_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR

If CurrentProject.AllForms("frmPickAddresses_multi").IsLoaded Then
Dim tAddrID As Long, tRstAddr As DAO.Recordset
Dim strADDR_CHN As String, strADDR_PY As String

gUseADDRID = True
CmdAllPlaces.Enabled = True
ChkXYRef.Enabled = True
ChkSubUnits.Enabled = True
FrameXY.Enabled = True

Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Visible = True
Forms!frmPickAddresses_multi.Form!TxtAddrFilter.SetFocus
If Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Value Then
'
TxtAddrID.Value = 0
strADDR_PY = Forms!frmPickAddresses_multi.Form!TxtFilterPY
strADDR_CHN = Forms!frmPickAddresses_multi.Form!TxtFilterChn

If strADDR_CHN = "" Then
TxtPlaceChn.Value = "[[Filter]]"
TxtPlace.Value = "[" + strADDR_PY + "]"
Else
TxtPlaceChn.Value = "[" + strADDR_CHN + "]"
TxtPlace.Value = "[[Filter]]"
End If
Else
Forms!frmPickAddresses_multi.Form!TxtSelectCount.Visible = True
Forms!frmPickAddresses_multi.Form!TxtSelectCount.SetFocus
If Forms!frmPickAddresses_multi.Form!TxtSelectCount.Value > 1 Then
TxtPlaceChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
TxtPlace.Value = "[[Multi-Select]]"
TxtAddrID.Value = 0
Else
' only one record in ZZ_ADDRESSES: get its field values
'
Set tRstAddr = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
tRstAddr.MoveFirst
' MsgBox "Checking zz_addresses: no records"
TxtAddrID.Value = tRstAddr!c_addr_id
TxtPlaceChn.Value = tRstAddr!c_name_chn
TxtPlace.Value = tRstAddr!c_name
tRstAddr.Close
Set tRstAddr = Nothing
End If
End If
'
' now copy the records
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_LIST"
cmdSQL.Execute tRecDeleted

```



```

        cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT " + _
            "ZZ_ADDRESSES.c_addr_id FROM ZZ_ADDRESSES"
        cmdSQL.Execute tRecDeleted
    End If

    DoCmd.Close acForm, stDocName
    CmdSelectPlace.SetFocus
    TxtAddrID.Visible = False

Exit_CmdSelectPlace_Click:
    Exit Sub

Err_CmdSelectPlace_Click:
    MsgBox Err.Description
    Resume Exit_CmdSelectPlace_Click

End Sub

Private Sub CmdAllPlaces_Click()
    On Error GoTo Err_CmdAllPlaces_Click

    TxtAddrID.Value = -1

    TxtPlaceChn.Value = ""
    TxtPlace.Value = ""
    gUseADDRID = False
    ChkXYRef.Enabled = False
    ChkSubUnits.Enabled = False
    FrameXY.Enabled = False

Exit_CmdAllPlaces_Click:
    Exit Sub

Err_CmdAllPlaces_Click:
    MsgBox Err.Description
    Resume Exit_CmdAllPlaces_Click

End Sub

Private Sub CmdImportPlaces_Click()
    On Error GoTo Err_CmdImportPlaces_Click

    Dim stDocName As String, tRstAddresses As DAO.Recordset, tRstImportPlaces As DAO.Recordset
    Dim stLinkCriteria As String
    Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String, tLen As Integer, tQuit As Boolean

    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tFileSystem, tList

    tQuit = False
    '
    If Not tQuit Then
        ' open the list

        Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

        'Use a With...End With block to reference the FileDialog object.
        With dlgSaveAs
            .InitialFileName = ""
            If .Show = -1 Then
                '
                tFileName = ""
                For Each tFN In .SelectedItems
                    tFileName = tFN
                    If Not tFileName = "" Then
                        Exit For
                    End If
                Next
                If tFileName = "" Then
                    MsgBox "Bad file Name."
                    GoTo Exit_CmdImportPlaces_Click
                End If
            End If
        End With
        ' Clear the address table now that we are ready to go
        '
        Set cmdSQL = New ADODB.Command
        cmdSQL.ActiveConnection = CurrentProject.Connection
        cmdSQL.CommandType = adCmdText

```

```

'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_LIST"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "ImportPlaceList_Space", "TempImportList", tFileName, 0
'   TransferType=acImportDelim
'   SpecificationName = "TempImportList" (apparently it is saved in the database itself)
'   TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
'   HasFieldNames = False (0)
'
'   copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
"FROM ADDR_CODES RIGHT JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID " + _
"WHERE ((ADDR_CODES.c_addr_id) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported:  please look at InputErrorList."
End If
'
'   copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT TempImportList.ImportID " + _
"FROM ADDR_CODES INNER JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    Me.TxtPlace.Value = "[Imported List]"
    Me.TxtPlaceChn.Value = "[Imported List]"
    gUseADDRID = False
    ChkXYRef.Enabled = True
    ChkSubUnits.Enabled = True
    FrameXY.Enabled = True
End If

Set cmdSQL = Nothing
Set tFileSystem = Nothing
End If

Exit_CmdImportPlaces_Click:
Exit Sub

Err_CmdImportPlaces_Click:
MsgBox Err.Description
Resume Exit_CmdImportPlaces_Click

End Sub

Private Sub FrameFilterYears_Click()

'   disable all
Me.CmdFromDynasty.Enabled = False
Me.CmdToDynasty.Enabled = False
Me.CmdAllDynasties.Enabled = False
Me.TxtFromDynasty.Enabled = False
Me.TxtFromDynastyPY.Enabled = False
Me.TxtToDynasty.Enabled = False
Me.TxtToDynastyPY.Enabled = False
Me.TxtFromDynasty.Locked = False
Me.TxtFromDynastyPY.Locked = False
Me.TxtToDynasty.Locked = False
Me.TxtToDynastyPY.Locked = False

Me.TxtFromYear.Enabled = False
Me.TxtToYear.Enabled = False

If FrameFilterYears.Value = 2 Then

```

```

    ' enable index years
    Me.TxtFromYear.Enabled = True
    Me.TxtToYear.Enabled = True
ElseIf FrameFilterYears.Value = 3 Then
    ' enable dynasties
    Me.CmdFromDynasty.Enabled = True
    Me.CmdToDynasty.Enabled = True
    Me.CmdAllDynasties.Enabled = True
    Me.TxtFromDynasty.Enabled = True
    Me.TxtFromDynastyPY.Enabled = True
    Me.TxtToDynasty.Enabled = True
    Me.TxtToDynastyPY.Enabled = True
    Me.TxtFromDynasty.Locked = True
    Me.TxtFromDynastyPY.Locked = True
    Me.TxtToDynasty.Locked = True
    Me.TxtToDynastyPY.Locked = True
End If
End Sub

Private Sub TxtFromYear_LostFocus()
    gFromStr = Trim(TxtFromYear.Text)
End Sub

Private Sub TxtToYear_LostFocus()
    gToStr = Trim(TxtToYear.Text)
End Sub

Private Sub CmdHelp_Click()
    Dim tStrPDF As String

    tStrPDF = Application.CurrentProject.Path + "\HelpFiles\HelpFile_LookAtStatusiations.pdf"

    'MsgBox tStrPDF

    Application.FollowHyperlink tStrPDF, , True
End Sub

Private Sub writeKML()
    On Error GoTo Err_writeKML
    '
    ' This program will dump the results to a .gis file
    '
    If ZZ_SCRATCH_P_STATUS.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_writeKML
    End If
    '
    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream

    tPinyin = False
    If GISFrame.Value = 1 Then
        tStream.Charset = "utf-8"
        tCodeStr = "UTF8"
    ElseIf GISFrame.Value = 2 Then
        tStream.Charset = "ascii"
        tCodeStr = "ASCII"
        tPinyin = True
    Else
        tStream.Charset = "gb18030"
        tCodeStr = "GB18030"
    End If
    tStream.Mode = adModeReadWrite
    tStream.Type = adTypeText
    tStream.Open
    '
    ' next get a file
    '

    Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

    dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".kml"
    If dlgSaveAs.Show = -1 Then
        '
        tFileName = ""
        For Each tFN In dlgSaveAs.SelectedItems
            tFileName = tFN
            If Not tFileName = "" Then
                Exit For
            End If
        Next
    End If

```

```

        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_writeKML
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".kml"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
            tFileName = tFileName + ".kml"
        End If
    End If

    '
    ' write the file
    '
    'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count
    '
    ' process the table
    '
    Set tRstNode = ZZ_SCRATCH_P_STATUS.Form.Recordset
    tC = Chr(9) ' the tab
    tDQ = Chr(34) ' the double quotation mark
    '
    ' write the header
    '

    tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
    tStream.WriteText "<Document>", adWriteLine
    tStream.WriteText tC + "<name>ExtendedData+SchemaData</name>", adWriteLine
    tStream.WriteText tC + "<open>1</open>", adWriteLine '
    tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

    tStream.WriteText tC + "<Style id=" + tDQ + "status-balloon-template" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
    tStream.WriteText tC + tC + tC + "<text>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "<![CDATA[", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "ID: ${StatusGIS/PersonID} <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Name Chn: ${StatusGIS/NameHZ} <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Index Year: ${StatusGIS/IndexYear} <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Sex: ${StatusGIS/Sex} <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Address: ${StatusGIS/AddrName} ${StatusGIS/AddrHZ} <br/>", adWriteLine

    tStream.WriteText tC + tC + tC + tC + "XY Count: ${StatusGIS/XYCount} <br/><br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
    tStream.WriteText tC + tC + tC + "</text>", adWriteLine
    tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
    tStream.WriteText tC + "</Style>", adWriteLine
    tStream.WriteText tC + "<!-- Declare the type " + tDQ + "StatusGIS" + tDQ + " with 6 fields -->", adWriteLine

    tStream.WriteText tC + "<Schema name=" + tDQ + "StatusGIS" + tDQ + " id=" + tDQ + "StatusGISId" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person ID]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "NameHZ" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Name Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "Sex" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Sex]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrHZ" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + "</Schema>", adWriteLine

```

```

With tRstNode
,
.MoveFirst
Do While Not .EOF
' must guard against NULLs, even where there should not be any
'
' write the point header
'
tStream.WriteText tC + "<Placemark>", adWriteLine

If IsNull(!c_name) Then
tStr = "[Bad Data] "
Else
tStr = !c_name
End If
tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

tStream.WriteText tC + tC + "<styleUrl>#status-balloon-template</styleUrl>", adWriteLine
'
' First Year as time stamp
'
If IsNull(!c_index_year) Then
tStr = "N/A"
Else
tStr = Str(!c_index_year)
End If
tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
'
tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#StatusGISId" + tDQ + ">", adW
riteLine
'
' person ID
'
tStr = Str(!c_person_id)
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
' Person Name Chn
'
If IsNull(!c_name_chn) Then
tStr = tStr + "[Bad Data]"
Else
If Trim(!c_name_chn) = "" Then
tStr = "[?]"
Else
tStr = !c_name_chn
End If
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NameHZ" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
'
' Index Year
'
If IsNull(!c_index_year) Then
tStr = "N/A"
Else
tStr = Str(!c_index_year)
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
'
If IsNull(!c_sex) Then
tStr = "[?]"
Else
tStr = !c_sex
End If
tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "Sex" + tDQ + ">" + tStr + "</S
impleData>", adWriteLine
'
' Address Name
'
If IsNull(!c_addr_name) Then
tStr = "[?]"
ElseIf Trim(!c_addr_name) = "" Then
tStr = "[?]"
Else
tStr = !c_addr_name
End If

```

```

        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    '   Address Name Chinese
    '
    If IsNull(!c_addr_chn) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_chn
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrHZ" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    '   XY Count
    '
    If IsNull(!xy_count) Then
        tStr = "0"
    Else
        tStr = Str(!xy_count)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
    '
    tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
    tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
    tStream.WriteText tC + tC + "<Point>", adWriteLine
    '
    '   coordinates
    If IsNull(!x_coord) Then
        tStr = "0"
    Else
        tStr = Str(!x_coord)
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + ",0"
    Else
        tStr = tStr + "," + Str(!y_coord)
    End If
    tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
    '
    '   footer
    '
    tStream.WriteText tC + tC + "</Point>", adWriteLine
    tStream.WriteText tC + "</Placemark>", adWriteLine
    .MoveNext
    Loop
End With
'
'   footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_writeKML:
Exit Sub

Err_writeKML:
MsgBox Err.Description
Resume Exit_writeKML

End Sub
Private Sub CmdStoreID_Click()

```

```

Dim cmdSQL As ADODB.Command, tRecCount As Variant

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
    ' Display message.
    If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) =
vbNo Then
        Exit Sub
    Else
        cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
        cmdSQL.Execute tRecCount
    End If
End If

tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT DISTINCT ZZ_SCRATCH_P_STATUS.c_person_id FR
OM ZZ_SCRATCH_P_STATUS"

cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecCount
MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."
'
' update storage source
cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='Status' WHERE PersonIDSource.LineNum =1"
cmdSQL.Execute tRecCount

End Sub

```

```
Option Compare Database
```

```
Public gRstPeople As DAO.Recordset, gDisplayLanguage As String, gLabelsOK As Boolean
```

```
Public gImportPlaces As Boolean, gUseADDRID As Boolean, gFromStr As String, gToStr As String
```

```
Public gTextCatStr As String, gTextCatTypeStr As String, gFromDynasty As Integer, gToDynasty As Integer
```

```
Public gFromDynastyBegin As Integer, gFromDynastyEnd As Integer, gToDynastyBegin As Integer, gToDynastyEnd As Integer
```

```
Private Sub ChkIndexYears_Click()
```

```
    If TxtFromYear.Enabled Then
```

```
        TxtFromYear.Enabled = False
```

```
        TxtToYear.Enabled = False
```

```
    Else
```

```
        TxtFromYear.Enabled = True
```

```
        TxtToYear.Enabled = True
```

```
    End If
```

```
End Sub
```

```
Private Sub CmdAllDynasties_Click()
```

```
    gFromDynasty = -2
```

```
    gToDynasty = -2
```

```
    TxtFromDynasty.Value = ""
```

```
    TxtFromDynastyPY.Value = "All"
```

```
    TxtToDynasty.Value = ""
```

```
    TxtToDynastyPY.Value = "All"
```

```
End Sub
```

```
Private Sub CmdFromDynasty_Click()
```

```
    Dim stDocName As String
```

```
    Dim stLinkCriteria As String
```

```
    Dim strFromDynasty As String
```

```
    If gFromDynasty < 0 Then
```

```
        strFromDynasty = ""
```

```
    Else
```

```
        strFromDynasty = Str(gFromDynasty)
```

```
    End If
```

```
    stDocName = "frmPickDynasty"
```

```
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty
```

```
    If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
```

```
        gFromDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
```

```
        gFromDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
```

```
        gFromDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
```

```
    '
```

```
    ' check to see if we have a problem and reject selection
```

```
    '
```

```
    If gToDynasty > -1 Then
```

```
        If gFromDynastyBegin > gToDynastyEnd Then
```

```
            MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
```

```
            gFromDynasty = -1
```

```
            TxtFromDynasty.Value = ""
```

```
            TxtFromDynastyPY.Value = ""
```

```
        End If
```

```
    End If
```

```
    '
```

```
    ' value is OK
```

```
    '
```

```
    If gFromDynasty > -1 Then
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
```

```
        TxtFromDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value
```

```
        Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
```

```
        TxtFromDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
```

```
    End If
```

```
    DoCmd.Close acForm, stDocName
```

```
    '
```

```
    ' reset ToDynasty if necessary (-2 = all dynasties)
```

```
    '
```

```
    If gToDynasty = -2 Then
```


Prim_LookAtTexts - 2

```
        gToDynasty = -1
        TxtToDynasty.Value = ""
        TxtToDynastyPY.Value = ""
    End If
    '
End If
```

End Sub

Private Sub CmdImportTextCategories_Click()

On Error GoTo Err_CmdImportTextCategories_Click

```
Dim stDocName As String, tRstTextCategories As DAO.Recordset
Dim stLinkCriteria As String, tRstImportTextCategories As DAO.Recordset
Dim tString As String, tTextCatCode As Long, ti As Integer, tStrID As String, tQuit As Boolean
Dim tLen As Integer, cmdSQL As ADODB.Command
```

```
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant
Dim tFileSystem, tList
```

' first see if we already have a list

tQuit = False

'

If Not tQuit Then

' open the list

Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

'Use a With...End With block to reference the FileDialog object.

With dlgSaveAs

.InitialFileName = ""

If .Show = -1 Then

'

tFileName = ""

For Each tFN In .SelectedItems

tFileName = tFN

If Not tFileName = "" Then

Exit For

End If

Next

If tFileName = "" Then

MsgBox "Bad file Name."

GoTo Exit_CmdImportTextCategories_Click

End If

End If

End With

'

' Clear the address table now that we are ready to go

'

Set cmdSQL = New ADODB.Command

cmdSQL.ActiveConnection = CurrentProject.Connection

cmdSQL.CommandType = adCmdText

'

cmdSQL.CommandText = "Delete * from ZZ_TEXT_BIBLCAT_CODES"

cmdSQL.Execute tRecDeleted

'

cmdSQL.CommandText = "Delete * from InputErrorList"

cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"

cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "TextBiblcatsListImport Specification", "TempImportList", tFileName, 0

' TransferType=acImportDelim

' SpecificationName = "TempImportList" (apparently it is saved in the database itself)

' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)

' HasFieldNames = False (0)

'

' copy the bad IDs

'

tStrSQL = "INSERT INTO InputErrorList (c_ID) SELECT TempImportList.ImportID " +

List.ImportID " +

"FROM TEXT_BIBLCAT_CODES RIGHT JOIN TempImportList ON TEXT_BIBLCAT_CODES.c_text_cat_code = TempImport

cmdSQL.CommandText = tStrSQL

```

cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If
'
' copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_TEXT_BIBLCAT_CODES ( c_text_cat_code ) SELECT DISTINCT TempImportList.ImportID
" + "FROM TEXT_BIBLCAT_CODES INNER JOIN TempImportList ON TEXT_BIBLCAT_CODES.c_text_cat_code = TempImport
List.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

Me.TxtTypeDesc.Value = ""
Me.TxtTypeChn.Value = ""
If tRecDeleted > 0 Then
    Me.TxtTextCatDesc.Value = "[Imported List]"
    Me.TxtTextCatDescChn.Value = "[Imported List]"
    Me.CmdQuery.Enabled = True
    Me.CmdSaveTextCategories.Enabled = True
Else
    Me.TxtTextCatDesc.Value = ""
    Me.TxtTextCatDescChn.Value = ""
    Me.CmdQuery.Enabled = False
    Me.CmdSaveTextCategories.Enabled = False
End If

Set cmdSQL = Nothing
End If

Exit_CmdImportTextCategories_Click:
Exit Sub

Err_CmdImportTextCategories_Click:
MsgBox Err.Description
Resume Exit_CmdImportTextCategories_Click

End Sub

Private Sub CmdNeo4j_Click()
On Error GoTo Err_CmdNeo4j_Click
'
' This program will dump the results of the search to five CSV files
'
' warn the user that a lot of files will be created
'
MsgBox "Neo4j requires that from 6 to 9 files be created."
'
' allocate the file variables
'
Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer, tFileName As String, tFN As Variant
'
' next get the People file
'
Dim tRstPeople As DAO.Recordset, tRstTexts As DAO.Recordset, tRstTextCodes As DAO.Recordset, tRstPlace As DAO
.Recordset
Dim tRstPeopleTexts As DAO.Recordset, tRstPeoplePlace As DAO.Recordset, tStr As String, tC As String
Dim tQueryStr As String, tPersonID As Long
Dim gStream As ADODB.Stream, tCodeStr As String
'
' set up the stream to write to

Set gStream = New ADODB.Stream
'
If GISFrame.Value = 1 Then
    gStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    gStream.Charset = "big5"
    tCodeStr = "BIG5"
ElseIf GISFrame.Value = 3 Then
    gStream.Charset = "gb2312"
    tCodeStr = "GB2312"
Else
    gStream.Charset = "ascii"
    tCodeStr = "ascii"

```

```

End If
'
tC = Chr(44) ' the comma
'
' prepare the temp tables for the people, place, peoplePlace and entry data

Dim cmdSQL As ADODB.Command
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'

Set tRstPeopleTexts = CurrentDb.OpenRecordset("ZZ_SCRATCH_BIOG_TEXT_DATA", dbOpenDynaset)
Set tRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_TEXT", dbOpenDynaset)

' Open the People file

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "People_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    ' now process the file (second true removed to make ASCII)
    '
    ' we have a file name: now open the stream for writing

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

    tRstPeople.MoveLast
    '
    ' process the four tables
    '
    ' first the nodes: define the record structure
    '
    ' if the file is strictly ASCII, the label is the pinyin, but if there are characters, then we add a pin
    yin field
    If tCodeStr = "ascii" Then
        tStr = "NameID" + tC + "NamePY" + tC + "IndexYear" + tC + "Dynasty" + tC + "Sex"
    Else
        tStr = "NameID" + tC + "NameHZ" + tC + "NamePY" + tC + "IndexYear" + tC + "Dynasty" + tC + "Sex"
    End If
    gStream.WriteText tStr, adWriteLine
    '
    With tRstPeople
        .MoveFirst
        Do While Not .EOF
            ' the ID of the person
            tStr = Trim(Str(!c_person_id)) + tC
            '
            ' name
            '
            If tCodeStr = "ascii" Then
                If IsNull(!c_name) Then
                    tStr = tStr + tC
                Else
                    tStr = tStr + !c_name + tC
                End If
            Else
                If IsNull(!c_name_chn) Then
                    tStr = tStr + "Missing" + tC
                End If
            End If
        Loop
    End With

```

```

        Else
            tStr = tStr + !c_name_chn + tC
        End If

        If IsNull(!c_name) Then
            tStr = tStr + "Missing" + tC
        Else
            tStr = tStr + !c_name + tC
        End If
    End If
    '
    ' indexyear = c_index_year INT
    '
    If IsNull(!c_index_year) Then
        tStr = tStr + "-2000" + tC
    Else
        tStr = tStr + Trim(Str(!c_index_year)) + tC
    End If

    ' dynasty information
    '
    If IsNull(!c_dynasty) Then
        tStr = tStr + "unknown" + tC
    Else
        If tCodeStr = "ascii" Then
            tStr = tStr + !c_dynasty + tC
        Else
            tStr = tStr + !c_dynasty_chn + tC
        End If
    End If
    '
    If IsNull(!c_sex) Then
        tStr = tStr + "Missing"
    Else
        tStr = tStr + !c_sex
    End If
    '
    gStream.WriteText tStr, adWriteLine
    '
    .MoveNext
End With
Loop
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now the PeopleText file
'
dlgSaveAs.InitialFileName = "PeopleText_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

```

```

tStr = "NameID" + tC + "TextID" + tC + "TextRoleID"
gStream.WriteText tStr, adWriteLine
'
With tRstPeopleTexts
    .MoveFirst
    Do While Not .EOF
        ' the ID of the person
        tStr = Trim(Str(!c_personid)) + tC
        '
        ' text id
        '
        If IsNull(!c_textid) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_textid)) + tC
        End If
        '
        ' role id
        '
        If IsNull(!c_role_id) Then
            tStr = tStr + "0" + tC
        Else
            tStr = tStr + Trim(Str(!c_role_id))
        End If
        '
        gStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now places
'
' get a file name
'
dlgSaveAs.InitialFileName = "Places_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    ' now process the file
    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_addr_id, ZZ_SCRATCH_P_TEXT.c_addr_name, ZZ_SCRATCH_P_TEX
T.c_addr_chn, " + _
        "ZZ_SCRATCH_P_TEXT.x_coord, ZZ_SCRATCH_P_TEXT.y_coord " + _
        "FROM ZZ_SCRATCH_P_TEXT " + _
        "WHERE (ZZ_SCRATCH_P_TEXT.c_addr_id > 0)"
    '
    Set tRstPlace = CurrentDb.OpenRecordset(tQueryStr, dbOpenDynaset)
    '

```

```

If tCodeStr = "ascii" Then
    tStr = "PlaceID" + tC + "PlacePY" + tC + "PlaceX" + tC + "PlaceY"
Else
    tStr = "PlaceID" + tC + "PlacePY" + tC + "PlaceHZ" + tC + "PlaceX" + tC + "PlaceY"
End If
gStream.WriteText tStr, adWriteLine
With tRstPlace
    .MoveFirst
    Do While Not .EOF
        ' the ID of the place
        If Not IsNull(!c_addr_id) Then
            tStr = Trim(Str(!c_addr_id)) + tC
            ' address name

            If IsNull(!c_addr_name) Then
                tStr = tStr + "unknown" + tC
            Else
                tStr = tStr + !c_addr_name + tC
            End If
            '
            If Not (tCodeStr = "ascii") Then
                If IsNull(!c_addr_chn) Then
                    tStr = tStr + "unknown" + tC
                Else
                    tStr = tStr + !c_addr_chn + tC
                End If
            End If

            If IsNull(!x_coord) Then
                tStr = tStr + "0.0" + tC
            Else
                tStr = tStr + Str(!x_coord) + tC
            End If

            If IsNull(!y_coord) Then
                tStr = tStr + "0.0"
            Else
                tStr = tStr + Str(!y_coord)
            End If
            '
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaces
'
dlgSaveAs.InitialFileName = "PeoplePlaces_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
End If

```

```

End If
'
gStream.Open
'
tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id, ZZ_BIOG_MAIN.c_index_addr_id, " + _
            "ZZ_BIOG_MAIN.c_index_addr_type_code " + _
            "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZ_BIOG_
MAIN.c_personid " + _
            "WHERE (ZZ_BIOG_MAIN.c_index_addr_type_code > 0)"

Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

tStr = "NameID" + tC + "PlaceID" + tC + "PersonPlaceCode"

gStream.WriteText tStr, adWriteLine

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_index_addr_id) Then
            '
            tStr = Trim(Str(!c_person_id)) + tC
            '
            tStr = tStr + Trim(Str(!c_index_addr_id)) + tC
            '
            tStr = tStr + Trim(Str(!c_index_addr_type_code))
            '
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With
'
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now peoplePlaceCode: use ZZ_SCRATCH_PEOPLE
'

dlgSaveAs.InitialFileName = "PeoplePlacesCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_BIOG_MAIN.c_index_addr_type_code, ZZ_BIOG_MAIN.c_index_addr_type_desc,
" + _
            "ZZ_BIOG_MAIN.c_index_addr_type_chn " + _
            "FROM ZZ_SCRATCH_P_TEXT INNER JOIN ZZ_BIOG_MAIN ON ZZ_SCRATCH_P_TEXT.c_person_id = ZZ_BIOG_
MAIN.c_personid " + _
            "WHERE (ZZ_BIOG_MAIN.c_index_addr_type_code > 0 AND ZZ_BIOG_MAIN.c_index_addr_id > 0)"

    Set tRstPeoplePlace = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then

```

```

        tStr = "personPlaceCode" + tC + "personPlaceTrans"
Else
    tStr = "personPlaceCode" + tC + "personPlaceTrans" + tC + "personPlaceHZ"
End If

```

```

gStream.WriteText tStr, adWriteLine

```

```

With tRstPeoplePlace
    .MoveFirst
    Do While Not .EOF
        If Not IsNull(!c_index_addr_type_code) Then
            ,
            tStr = Trim(Str(!c_index_addr_type_code)) + tC
            ,
            tStr = tStr + !c_index_addr_type_desc
            ,
            If Not (tCodeStr = "ascii") Then
                tStr = tStr + tC + !c_index_addr_type_chn
            End If
            gStream.WriteText tStr, adWriteLine
        End If
        .MoveNext
    Loop
End With

```

```

' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close

```

```

Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If

```

```

'
' get text codes

```

```

dlgSaveAs.InitialFileName = "TextCode_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then

```

```

    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next

```

```

    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdNeo4j_Click
    Else

```

```

        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If

```

```

    gStream.Mode = adModeReadWrite
    gStream.Type = adTypeText
    gStream.Open

```

```

    If tCodeStr = "ascii" Then
        tStr = "TextID" + tC + "TextPY" + tC + "TextCategoryCode" + tC + "TextCategoryDesc"
    Else

```

```

        tStr = "TextID" + tC + "TextPY" + tC + "TextHZ" + tC + "TextCategoryCode" + tC + "TextCategoryDesc" +
tC + "TextCategoryHZ"
    End If
    gStream.WriteText tStr, adWriteLine

```

```

' get the codes

```

```

tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_BIOG_TEXT_DATA.c_textid, ZZ_SCRATCH_BIOG_TEXT_DATA.c_title, " + _
            "ZZ_SCRATCH_BIOG_TEXT_DATA.c_title_chn, ZZ_SCRATCH_BIOG_TEXT_DATA.c_text_cat_code, " + _
            "ZZ_SCRATCH_BIOG_TEXT_DATA.c_text_cat_desc, ZZ_SCRATCH_BIOG_TEXT_DATA.c_text_cat_desc_chn
" + _
            "FROM ZZ_SCRATCH_BIOG_TEXT_DATA"

```



```

Set tRstTextCodes = CurrentDb.OpenRecordset(tQueryStr)
With tRstTextCodes
    .MoveFirst
    Do While Not .EOF
        '
        tStr = Trim(Str(!c_textid)) + tC
        '
        ' title
        '
        If IsNull(!c_title) Then
            tStr = tStr + "Missing" + tC
        Else
            tStr = tStr + Trim(!c_title) + tC
        End If
        '
        ' title HZ
        '
        If Not (tCodeStr = "ascii") Then
            If IsNull(!c_title_chn) Then
                tStr = tStr + "Missing" + tC
            Else
                tStr = tStr + Trim(!c_title_chn) + tC
            End If
        End If
        '
        ' category code
        If IsNull(!c_text_cat_code) Then
            tStr = tStr + "Missing" + tC
        Else
            tStr = tStr + Trim(!c_text_cat_code) + tC
        End If
        '
        ' category desc
        '
        If IsNull(!c_text_cat_desc) Then
            tStr = tStr + "Missing"
        Else
            tStr = tStr + Trim(!c_text_cat_desc)
        End If
        '
        ' category HZ
        '
        If Not (tCodeStr = "ascii") Then
            If IsNull(!c_text_cat_desc_chn) Then
                tStr = tStr + tC + "Missing"
            Else
                tStr = tStr + tC + Trim(!c_text_cat_desc_chn)
            End If
        End If
        '
        gStream.WriteText tStr, adWriteLine
        '
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
gStream.Flush
' and write the stream to the file
gStream.SaveToFile tFileName, adSaveCreateOverWrite
'
gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If
'
' now Role Code:
'
dlgSaveAs.InitialFileName = "PersonTextRoleCodes_" + tCodeStr + ".csv"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
    End If
End If

```

```

        GoTo Exit_CmdNeo4j_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".csv"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".csv") Then
            tFileName = tFileName + ".csv"
        End If
    End If
    '
    gStream.Open
    '
    tQueryStr = "SELECT DISTINCT ZZ_SCRATCH_BIOG_TEXT_DATA.c_role_id, ZZ_SCRATCH_BIOG_TEXT_DATA.c_role_desc,
" + _
        "ZZ_SCRATCH_BIOG_TEXT_DATA.c_role_desc_chn " + _
        "FROM ZZ_SCRATCH_BIOG_TEXT_DATA " + _
        "WHERE (ZZ_SCRATCH_BIOG_TEXT_DATA.c_role_id > -1)"

    Set tRstTextCode = CurrentDb.OpenRecordset(tQueryStr)

    If tCodeStr = "ascii" Then
        tStr = "PersonTextRoleCode" + tC + "PersonTextRoleDesc"
    Else
        tStr = "PersonTextRoleCode" + tC + "PersonTextRoleDesc" + tC + "PersonTextRoleDesc"
    End If

    gStream.WriteText tStr, adWriteLine

    With tRstTextCode
        .MoveFirst
        Do While Not .EOF
            If Not IsNull(!c_role_id) Then
                '
                tStr = Trim(Str(!c_role_id)) + tC
                '
                If IsNull(!c_role_desc) Then
                    tStr = tStr + "Missing"
                Else
                    tStr = tStr + !c_role_desc
                End If
                '
                If Not (tCodeStr = "ascii") Then
                    If IsNull(!c_role_desc_chn) Then
                        tStr = tStr + tC + "Missing"
                    Else
                        tStr = tStr + tC + !c_role_desc_chn
                    End If
                End If
                gStream.WriteText tStr, adWriteLine
            End If
            .MoveNext
        Loop
    End With
    '
    ' now make sure all the data is copied to tStream
    gStream.Flush
    ' and write the stream to the file
    gStream.SaveToFile tFileName, adSaveCreateOverWrite
    '
    gStream.Close
Else
    'The user pressed Cancel.
    GoTo Exit_CmdNeo4j_Click
End If

MsgBox "Finished saving to Neo4j"
'
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdNeo4j_Click:
Exit Sub

Err_CmdNeo4j_Click:
MsgBox Err.Description
Resume Exit_CmdNeo4j_Click

End Sub

Private Sub CmdPickTextCat_Click()

```

```
On Error GoTo Err_CmdPickTextCat_Click
```

```
Dim stDocName As String
Dim stLinkCriteria As String
Dim strTextCat As String
```

```
TxtTextCatCode.Visible = True
TxtTextCatCode.SetFocus
strTextCat = TxtTextCatCode.Text
```

```
stDocName = "frmPickTextCat_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strTextCat
```

```
If CurrentProject.AllForms("frmPickTextCat_multi").IsLoaded Then
    Dim intTextCat As Integer
    Dim strTextCat_DESC As String
```

```
Forms!frmPickTextCat_multi.Form!TxtTextCatID.Visible = True
Forms!frmPickTextCat_multi.Form!TxtTextCatID.SetFocus
intTextCat = Forms!frmPickTextCat_multi.Form!TxtTextCatID.Value
Forms!frmPickTextCat_multi.Form!subTreeView.SetFocus
Forms!frmPickTextCat_multi.Form!TxtTextCatID.Visible = False
TxtTextCatCode.Value = intTextCat
gTextCatCodeStr = Trim(Str(intTextCat))
```

```
If TxtTextCatCode.Value < 0 Then
    If TxtTextCatCode.Value = -1 Then
        TxtTextCatDesc.Value = "[All]"
        TxtTextCatDescChn.Value = "[All]"
    Else
        TxtTextCatDesc.Value = "[Multi-Select]"
        TxtTextCatDescChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
    End If
```

```
Forms!frmPickTextCat_multi.Form!TxtTypeID.Visible = True
Forms!frmPickTextCat_multi.Form!TxtTypeID.SetFocus
strTextCat_DESC = Forms!frmPickTextCat_multi.Form!TxtTypeID.Value
Forms!frmPickTextCat_multi.Form!subTreeView.SetFocus
Forms!frmPickTextCat_multi.Form!TxtTypeID.Visible = False
TxtTypeCode.Value = strTextCat_DESC
gTextCatTypeStr = Trim(strTextCat_DESC)
```

```
If TxtTypeCode.Value = "000" Then
    TxtTypeDesc.Value = "[ALL]"
    TxtTypeChn.Value = "[ALL]"
Else
```

```
Forms!frmPickTextCat_multi.Form!TxtTypeDesc.Visible = True
Forms!frmPickTextCat_multi.Form!TxtTypeDesc.SetFocus

strTextCat_DESC = Forms!frmPickTextCat_multi.Form!TxtTypeDesc.Value

Forms!frmPickTextCat_multi.Form!subTreeView.SetFocus
Forms!frmPickTextCat_multi.Form!TxtTypeDesc.Visible = False

TxtTypeDesc.Value = strTextCat_DESC

Forms!frmPickTextCat_multi.Form!TxtTypeDescChn.Visible = True
Forms!frmPickTextCat_multi.Form!TxtTypeDescChn.SetFocus

strTextCat_DESC = Forms!frmPickTextCat_multi.Form!TxtTypeDescChn.Value

Forms!frmPickTextCat_multi.Form!subTreeView.SetFocus
Forms!frmPickTextCat_multi.Form!TxtTypeDescChn.Visible = False

TxtTypeChn.Value = strTextCat_DESC
End If
```

```
Else
    Forms!frmPickTextCat_multi.Form!TxtTextCatDesc.Visible = True
    Forms!frmPickTextCat_multi.Form!TxtTextCatDesc.SetFocus
    strTextCat_DESC = Forms!frmPickTextCat_multi.Form!TxtTextCatDesc.Value
    Forms!frmPickTextCat_multi.Form!subTreeView.SetFocus
    Forms!frmPickTextCat_multi.Form!TxtTextCatDesc.Visible = False
    TxtTextCatDesc.Value = strTextCat_DESC

    'MsgBox "Getting Chinese Description"
    Forms!frmPickTextCat_multi.Form!TxtTextCatDescChn.Visible = True
    Forms!frmPickTextCat_multi.Form!TxtTextCatDescChn.SetFocus
    strTextCat_DESC = Forms!frmPickTextCat_multi.Form!TxtTextCatDescChn.Value
    Forms!frmPickTextCat_multi.Form!subTreeView.SetFocus
    Forms!frmPickTextCat_multi.Form!TxtTextCatDescChn.Visible = False
```

```

        TxtTextCatDescChn.Value = strTextCat_DESC

        'MsgBox "Setting Type to Null"
        TxtTypeCode.Value = ""
        TxtTypeDesc.Value = "N/A"
        TxtTypeChn.Value = "N/A"
    End If

    DoCmd.Close acForm, stDocName
    '
    CmdQuery.Enabled = True
    Me.CmdSaveTextCategories.Enabled = True
Else
    CmdQuery.Enabled = False
    Me.CmdSaveTextCategories.Enabled = False
End If

CmdPickTextCat.SetFocus
TxtTextCatCode.Visible = False

Exit_CmdPickTextCat_Click:
Exit Sub

Err_CmdPickTextCat_Click:
MsgBox Err.Description
Resume Exit_CmdPickTextCat_Click

End Sub

Private Sub CmdQuery_Click()
    On Error GoTo Err_Run_Query

    Dim rst As DAO.Recordset, tContinue As Integer
    Dim tRstTextCat As DAO.Recordset, tRstAddrList As DAO.Recordset, tRstDummy As DAO.Recordset
    Dim tQueryInsertStr As String, tQuerySelectStr As String, tQueryFromStr As String, tQueryWhereStr As String
    Dim tQueryStr As String, tRecDrop As Long, tStrWhereSQL As String
    Dim tUseAddr As Boolean, tUseIndexYears As Boolean, tUseDynasties As Boolean

    Dim cmdSQL As ADODB.Command, tRecCount As Long, tRecIndexYearCount As Long

    Set cmdSQL = New ADODB.Command
    '
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' to clear the table, close and then delete records
    '
    Set tRstTextCat = ZZ_SCRATCH_TEXT.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_TR", dbOpenDynaset)
    Set ZZ_SCRATCH_TEXT.Form.Recordset = tRstDummy
    tRstTextCat.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_BIOG_TEXT_DATA"
    cmdSQL.Execute tRecCount
    '
    ' now the people table
    '
    Set gRstPeople = ZZ_SCRATCH_P_TEXT.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_SP", dbOpenDynaset)
    Set ZZ_SCRATCH_P_TEXT.Form.Recordset = tRstDummy
    gRstPeople.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_TEXT"
    cmdSQL.Execute tRecCount
    '
    ' see whether index years or dynasties will be used
    '
    If Me.FrameFilterYears.Value = 1 Then
        tUseIndexYears = False
        tUseDynasties = False
    ElseIf Me.FrameFilterYears.Value = 2 Then
        tUseIndexYears = True
        tUseDynasties = False
    Else
        tUseIndexYears = False
        tUseDynasties = True
    End If
    '
    ' now see if address IDs will be used. If so, zap the scratch file and repopulate

```

```

'
' MsgBox "About to process address"
If gUseADDRID Then
'
' the strategy here is to fill the scratch file with all the relevant addresses from ZZZ_BELONGS_TO
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR"
cmdSQL.Execute tRecCount
'
If ChkSubUnits.Value Then
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) " + _
"SELECT DISTINCT ZZZ_BELONGS_TO.c_addr_id " + _
"FROM ZZ_SCRATCH_ADDR_LIST INNER JOIN ZZZ_BELONGS_TO ON " + _
"ZZ_SCRATCH_ADDR_LIST.c_addr_id = ZZZ_BELONGS_TO.c_belongs_to"
Else
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id ) " + _
"SELECT DISTINCT c_addr_id " + _
"FROM ZZ_SCRATCH_ADDR_LIST"
End If

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
' see if we need to use the historical XY search
'
If ChkXYRef.Value Then
'
' the strategy here is to dump the IDs to ZZ_ADDRESSES then copy to ZZ_SCRATCH_ADDR_LIST
' (I borrow ZZ_ADDRESSES from the Pick Addresses form in order to keep the initial selection
' of addresses for the query intact.)
'
' zap the list
'
tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' run the query
'
' FrameXY.Value = 2 :: Narrow, FrameXY.Value = 1 :: Broad
'
If FrameXY.Value = 2 Then
tStrWhereSQL = "WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.03) And " + _
"([ADDR_CODES.x_coord]<=([ADDR_CODES_1].[x_coord]+0.03)) AND " + _
"([ADDR_CODES.y_coord]>=([ADDR_CODES_1].[y_coord]-0.03) And " + _
"([ADDR_CODES.y_coord]<=([ADDR_CODES_1].[y_coord]+0.03)))"
Else
tStrWhereSQL = "WHERE (((ADDR_CODES.x_coord)>=([ADDR_CODES_1].[x_coord]-0.06) And " + _
"([ADDR_CODES.x_coord]<=([ADDR_CODES_1].[x_coord]+0.06)) AND " + _
"([ADDR_CODES.y_coord]>=([ADDR_CODES_1].[y_coord]-0.06) And " + _
"([ADDR_CODES.y_coord]<=([ADDR_CODES_1].[y_coord]+0.06)))"
End If
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id )SELECT DISTINCT ADDR_CODES.c_addr_id " + _
"FROM ADDR_CODES, ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES AS ADDR_CODES_1 ON " + _
"ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES_1.c_addr_id " + tStrWhereSQL

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' now get the address IDs from the initial list that have no xy coordinates
'
tQueryStr = "INSERT INTO ZZ_ADDRESSES ( c_addr_id ) SELECT ZZ_SCRATCH_ADDR.c_addr_id " + _
"FROM ZZ_SCRATCH_ADDR INNER JOIN ADDR_CODES ON " + _
"ZZ_SCRATCH_ADDR.c_addr_id = ADDR_CODES.c_addr_id " + _
"WHERE (((ADDR_CODES.x_coord) Is Null)) OR (((ADDR_CODES.y_coord) Is Null)))"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' zap ZZ_SCRATCH_ADDR
'
tQueryStr = "DELETE * FROM ZZ_SCRATCH_ADDR"
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecDeleted
'
' copy the list
'
tQueryStr = "INSERT INTO ZZ_SCRATCH_ADDR ( c_addr_id )SELECT DISTINCT ZZ_ADDRESSES.c_addr_id " + _
"FROM ZZ_ADDRESSES"
cmdSQL.CommandText = tQueryStr

```

```

        cmdSQL.Execute tRecDeleted
    '
    ' zap the temporary list
    '
    tQueryStr = "DELETE * FROM ZZ_ADDRESSES"
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecDeleted
    '
End If

tUseAddr = True
Else
    tUseAddr = False
End If

' next build the appropriate query string
tQueryInsertStr = "INSERT INTO ZZ_SCRATCH_BIOG_TEXT_DATA ( c_personid, c_name_chn, c_name, c_sex, c_dy, c_dyn
asty, c_dynasty_chn, " + _
    "c_index_year, c_index_year_type_code, c_index_year_type_desc, c_index_year_type_hz, c_addr_id, c_addr_na
me, c_addr_chn, " + _
    "x_coord, y_coord, c_textid, c_title, c_title_chn, c_text_cat_code, c_text_cat_desc, c_text_cat_desc_chn,
c_role_id, c_role_desc, c_role_desc_chn, " + _
    "c_source, c_source_title, c_source_chn ) "
tQuerySelectStr = "SELECT ZZZ_BIOG_TEXT_DATA.c_personid, ZZZ_BIOG_TEXT_DATA.c_name_chn, ZZZ_BIOG_TEXT_DATA.c_
name, ZZZ_BIOG_TEXT_DATA.c_sex, " + _
    "ZZZ_BIOG_TEXT_DATA.c_dy, ZZZ_BIOG_TEXT_DATA.c_dynasty, ZZZ_BIOG_TEXT_DATA.c_dynasty_chn, ZZZ_BIOG_TEXT_D
ATA.c_index_year, " + _
    "ZZZ_BIOG_TEXT_DATA.c_index_year_type_code, ZZZ_BIOG_TEXT_DATA.c_index_year_type_desc, ZZZ_BIOG_TEXT_DATA
.c_index_year_type_hz, " + _
    "ZZZ_BIOG_TEXT_DATA.c_addr_id, ZZZ_BIOG_TEXT_DATA.c_addr_name, ZZZ_BIOG_TEXT_DATA.c_addr_name_chn, ZZZ_BI
OG_TEXT_DATA.x_coord, " + _
    "ZZZ_BIOG_TEXT_DATA.y_coord, ZZZ_BIOG_TEXT_DATA.c_textid, ZZZ_BIOG_TEXT_DATA.c_title, ZZZ_BIOG_TEXT_DATA.
c_title_chn, " + _
    "ZZZ_BIOG_TEXT_DATA.c_bibl_cat_code, ZZZ_BIOG_TEXT_DATA.c_bibl_cat_desc, ZZZ_BIOG_TEXT_DATA.c_bibl_cat_de
sc_chn, " + _
    "ZZZ_BIOG_TEXT_DATA.c_role_id, ZZZ_BIOG_TEXT_DATA.c_role_desc, ZZZ_BIOG_TEXT_DATA.c_role_desc_chn, " + _
    "ZZZ_BIOG_TEXT_DATA.c_source, ZZZ_BIOG_TEXT_DATA.c_source_title, ZZZ_BIOG_TEXT_DATA.c_source_chn "

tQueryInsertStr = "INSERT INTO ZZ_SCRATCH_BIOG_TEXT_DATA ( c_personid, c_name_chn, c_name, c_sex, c_dy, c_dyn
asty, c_dynasty_chn, c_index_year, " + _
    "c_index_year_type_desc, c_index_year_type_hz, c_textid, c_title, c_title_chn, c_text_cat_desc, c_text_ca
t_desc_chn, c_role_id, c_role_desc, " + _
    "c_role_desc_chn, c_addr_id, c_addr_name, c_addr_chn, x_coord, y_coord ) "
tQuerySelectStr = "SELECT ZZZ_BIOG_TEXT_DATA.c_personid, ZZZ_BIOG_TEXT_DATA.c_name_chn, ZZZ_BIOG_TEXT_DATA.c_
_name, IIf([c_female], 'F', 'M') AS c_sex, " + _
    "ZZZ_BIOG_TEXT_DATA.c_dy, ZZZ_BIOG_TEXT_DATA.c_dynasty, ZZZ_BIOG_TEXT_DATA.c_dynasty_chn, ZZZ_BIOG_TEXT_D
ATA.c_index_year, " + _
    "ZZZ_BIOG_TEXT_DATA.c_index_year_type_desc, " + _
    "ZZZ_BIOG_TEXT_DATA.c_index_year_type_hz, ZZZ_BIOG_TEXT_DATA.c_textid, ZZZ_BIOG_TEXT_DATA.c_title, ZZZ_BI
OG_TEXT_DATA.c_title_chn, " + _
    "ZZ_TEXT_BIBLCAT_CODES.c_text_cat_desc, ZZ_TEXT_BIBLCAT_CODES.c_text_cat_desc_chn, ZZZ_BIOG_TEXT_DATA.c_r
ole_id, " + _
    "ZZZ_BIOG_TEXT_DATA.c_role_desc, ZZZ_BIOG_TEXT_DATA.c_role_desc_chn, ZZZ_BIOG_MAIN.c_index_addr_id, ZZZ_B
IOG_MAIN.c_index_addr_name, " + _
    "ZZZ_BIOG_MAIN.c_index_addr_chn, ZZZ_BIOG_MAIN.x_coord, ZZZ_BIOG_MAIN.y_coord "

If tUseDynasties And gFromDynasty > -2 Then
    If tUseAddr Then
        tQueryFromStr = "FROM ((ZZZ_BIOG_TEXT_DATA INNER JOIN DYNASTIES ON ZZZ_BIOG_TEXT_DATA.c_dy = DYNASTIE
S.c_dy) " + _
            "INNER JOIN ZZ_TEXT_BIBLCAT_CODES ON ZZZ_BIOG_TEXT_DATA.c_bibl_cat_code = ZZ_TEXT_BIBLCAT_CODES.c_
_text_cat_code) " + _
            "INNER JOIN ZZ_SCRATCH_ADDR ON ZZZ_BIOG_TEXT_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id "
    Else
        tQueryFromStr = "FROM (ZZZ_BIOG_TEXT_DATA INNER JOIN DYNASTIES ON ZZZ_BIOG_TEXT_DATA.c_dy = DYNASTIES
.c_dy) " + _
            "INNER JOIN ZZ_TEXT_BIBLCAT_CODES ON ZZZ_BIOG_TEXT_DATA.c_bibl_cat_code = ZZ_TEXT_BIBLCAT_CODES.c_
_text_cat_code "
    End If
Else
    If tUseAddr Then
        tQueryFromStr = "FROM (ZZZ_BIOG_TEXT_DATA " + _
            "INNER JOIN ZZ_TEXT_BIBLCAT_CODES ON ZZZ_BIOG_TEXT_DATA.c_bibl_cat_code = ZZ_TEXT_BIBLCAT_CODES.c_
_text_cat_code) " + _
            "INNER JOIN ZZ_SCRATCH_ADDR ON ZZZ_BIOG_TEXT_DATA.c_addr_id = ZZ_SCRATCH_ADDR.c_addr_id "
    Else
        tQueryFromStr = "FROM ZZZ_BIOG_TEXT_DATA " + _
            "INNER JOIN ZZ_TEXT_BIBLCAT_CODES ON ZZZ_BIOG_TEXT_DATA.c_bibl_cat_code = ZZ_TEXT_BIBLCAT_CODES.c_
_text_cat_code "
    End If
End If

```

```

End If

' set the where conditions
'
' Start with the index years
'
tQueryWhereStr = ""
If tUseIndexYears Then
    '
    ' four possibilities
    '
    If gFromStr = "" And gToStr = "" Then
        tQueryWhereStr = ""
    ElseIf gFromStr = "" Then
        tQueryWhereStr = "WHERE (((ZZZ_BIOG_TEXT_DATA.c_index_year)<=" + gToStr + ") "
    ElseIf gToStr = "" Then
        tQueryWhereStr = "WHERE (((ZZZ_BIOG_TEXT_DATA.c_index_year)>=" + gFromStr + ") "
    Else
        tQueryWhereStr = "WHERE (((ZZZ_BIOG_TEXT_DATA.c_index_year)<=" + gToStr + ") AND " + _
            "((ZZZ_BIOG_TEXT_DATA.c_index_year)>=" + gFromStr + ") "
    End If
ElseIf tUseDynasties Then
    '
    ' five possibilities (all, just from, just to, both from and to, and a cluelessly unset parameter)
    '
    If gFromDynasty = -2 Then
        tQueryWhereStr = "Where (((ZZZ_BIOG_TEXT_DATA.c_dy) > 0 ) "
    ElseIf gFromDynasty = -1 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty = -1 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") "
    ElseIf gFromDynasty = gToDynasty And gFromDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_dy) = " + Str(gFromDynasty) + ") "
    ElseIf gFromDynasty > 0 And gToDynasty > 0 Then
        tQueryWhereStr = "WHERE (((DYNASTIES.c_end)>" + Str(gFromDynastyBegin) + ") AND " + _
            "((DYNASTIES.c_start)<" + Str(gToDynastyEnd) + ") "
    Else
        tQueryWhereStr = ""
    End If
End If

If Not (tQueryWhereStr = "") Then
    tQueryWhereStr = tQueryWhereStr + ")"
End If

' run the query

'MsgBox tQueryInsertStr + tQuerySelectStr + tQueryFromStr + tQueryWhereStr

cmdSQL.CommandText = tQueryInsertStr + tQuerySelectStr + tQueryFromStr + tQueryWhereStr
cmdSQL.Execute tRecCount
'
' the next step is to make the table of people from the TextCatiations
'
If tRecCount > 0 Then
    '
    tQueryStr = "INSERT INTO ZZ_SCRATCH_P_TEXT ( c_person_id, c_name_chn, c_name, c_sex, c_dy, c_dynasty, c_d
ynasty_chn, c_index_year, " + _
        "c_index_year_type_code, c_index_year_type_desc, c_index_year_type_hz, c_addr_id, c_addr_name, c_addr
_chn, x_coord, y_coord ) " + _
        "SELECT DISTINCT ZZ_SCRATCH_BIOG_TEXT_DATA.c_personid, ZZ_SCRATCH_BIOG_TEXT_DATA.c_name_chn, ZZ_SCRAT
CH_BIOG_TEXT_DATA.c_name, " + _
        "ZZ_SCRATCH_BIOG_TEXT_DATA.c_sex, ZZ_SCRATCH_BIOG_TEXT_DATA.c_dy, ZZ_SCRATCH_BIOG_TEXT_DATA.c_dyn
asty, " + _
        "ZZ_SCRATCH_BIOG_TEXT_DATA.c_dynasty_chn, ZZ_SCRATCH_BIOG_TEXT_DATA.c_index_year, ZZ_SCRATCH_BIOG
_TEXT_DATA.c_index_year_type_code, " + _
        "ZZ_SCRATCH_BIOG_TEXT_DATA.c_index_year_type_desc, ZZ_SCRATCH_BIOG_TEXT_DATA.c_index_year_type_hz
, " + _
        "ZZ_SCRATCH_BIOG_TEXT_DATA.c_addr_id, ZZ_SCRATCH_BIOG_TEXT_DATA.c_addr_name, ZZ_SCRATCH_BIOG_TEXT
_DATA.c_addr_chn, " + _
        "ZZ_SCRATCH_BIOG_TEXT_DATA.x_coord, ZZ_SCRATCH_BIOG_TEXT_DATA.y_coord " + _
        "FROM ZZ_SCRATCH_BIOG_TEXT_DATA"
    cmdSQL.CommandText = tQueryStr
    cmdSQL.Execute tRecCount
    '
    ' the final step is to calculate the xy_count
    '
    If tRecCount > 0 Then
        '
    
```

```

cmdSQL.CommandText = "Delete * from tmpXY"
cmdSQL.Execute tRecDeleted
'
tQueryStr = "INSERT INTO tmpXY ( x_coord, y_coord, CountOfx_coord, CountOfy_coord ) " + _
"SELECT ZZ_SCRATCH_P_TEXT.x_coord, ZZ_SCRATCH_P_TEXT.y_coord, Count(ZZ_SCRATCH_P_TEXT.x_coord) "
+ _
"AS CountOfx_coord, Count(ZZ_SCRATCH_P_TEXT.y_coord) AS CountOfy_coord " + _
"FROM ZZ_SCRATCH_P_TEXT " + _
"GROUP BY ZZ_SCRATCH_P_TEXT.x_coord, ZZ_SCRATCH_P_TEXT.y_coord"
'
cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
'
tQueryStr = "UPDATE tmpXY INNER JOIN ZZ_SCRATCH_P_TEXT ON (tmpXY.y_coord = " + _
"ZZ_SCRATCH_P_TEXT.y_coord) AND (tmpXY.x_coord = ZZ_SCRATCH_P_TEXT.x_coord) " + _
"SET ZZ_SCRATCH_P_TEXT.xy_count = [tmpXY].[CountOfx_coord]"
'

cmdSQL.CommandText = tQueryStr
cmdSQL.Execute tRecCount
CmdGIS.Enabled = True
CmdNeo4j.Enabled = True
CmdStoreID.Enabled = True
Else
CmdGIS.Enabled = False
CmdNeo4j.Enabled = False
CmdStoreID.Enabled = False
End If
End If

Exit_Run_Query:
'
' now reopen the tables
'
Set tRstTextCat = CurrentDb.OpenRecordset("ZZ_SCRATCH_BIOG_TEXT_DATA", dbOpenDynaset)
Set ZZ_SCRATCH_TEXT.Form.Recordset = tRstTextCat
'
Set gRstPeople = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_TEXT", dbOpenDynaset)
Set ZZ_SCRATCH_P_TEXT.Form.Recordset = gRstPeople
'
' close everything
'
Set rst = Nothing
Set tRstDummy = Nothing
Set cmdSQL = Nothing
Exit Sub

Err_Run_Query:
MsgBox Err.Description
Resume Exit_Run_Query

End Sub

Private Sub CmdGIS_Click()
On Error GoTo Err_CmdGIS_Click
'
' If it is a KML file, call the routine and exit
'
If ChkKML.Value Then
Call writeKML
Exit Sub
End If

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstNode As DAO.Recordset
Dim tStr As String, tC As String, ti As Integer, tPinyin As Boolean
Dim tFileSystem, tGDF
'
' This program will dump the results to a .gis file
'
If ZZ_SCRATCH_P_TEXT.Form.Recordset.RecordCount = 0 Then
MsgBox "There are no records to save."
GoTo Exit_CmdGIS_Click
End If
'
Dim tStream As ADODB.Stream
Set tStream = New ADODB.Stream

tPinyin = False

```



```

If GISFrame.Value = 1 Then
    tStream.Charset = "utf-8"
    tCodeStr = "UTF8"
ElseIf GISFrame.Value = 2 Then
    tStream.Charset = "ascii"
    tCodeStr = "ASCII"
    tPinyin = True
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".tab"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdGIS_Click
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".tab"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".tab") Then
            tFileName = tFileName + ".tab"
        End If
    End If
    '
    ' write the file
    '
    'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count,NodeDist
    '
    ' process the table
    '
    Set tRstNode = ZZ_SCRATCH_P_TEXT.Form.Recordset
    tC = Chr(9) ' the tab
    '
    With tRstNode
        '
        ' write the header
        '
        If tPinyin Then
            tStr = "Name" + tC + "Sex" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "X" + tC + "Y" + tC + "xy_count"
        Else
            tStr = "Name" + tC + "NameChn" + tC + "Sex" + tC + "IndexYear" + tC + _
                "AddrName" + tC + "AddrChn" + tC + "X" + tC + "Y" + tC + "xy_count"
        End If
        tStream.WriteText tStr, adWriteLine
        .MoveFirst
        Do While Not .EOF
            ' must guard against NULLs
            '
            If Trim(!c_name) = "" Then
                tStr = "[?]" + tC
            Else
                tStr = !c_name + tC
            End If

            If Not tPinyin Then
                If Trim(!c_name_chn) = "" Then
                    tStr = tStr + "[?]" + tC
                Else
                    tStr = tStr + !c_name_chn + tC
                End If
            End If
        End If
    End With

```

```

If IsNull(!c_sex) Then
    tStr = tStr + "[?]" + tC
Else
    tStr = tStr + !c_sex + tC
End If

If IsNull(!c_index_year) Then
    tStr = tStr + "-2000" + tC
Else
    tStr = tStr + Str(!c_index_year) + tC
End If

```

```

' here guard against blanks as well

```

```

If IsNull(!c_addr_name) Then
    tStr = tStr + "[?]" + tC
ElseIf Trim(!c_addr_name) = "" Then
    tStr = tStr + "[?]" + tC
Else
    tStr = tStr + !c_addr_name + tC
End If

If Not tPinyin Then
    If IsNull(!c_addr_chn) Then
        tStr = tStr + "[?]" + tC
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = tStr + "[?]" + tC
    Else
        tStr = tStr + !c_addr_chn + tC
    End If
End If

```

```

If IsNull(!x_coord) Then
    tStr = tStr + "0" + tC
Else
    tStr = tStr + Str(!x_coord) + tC
End If

```

```

If IsNull(!y_coord) Then
    tStr = tStr + "0" + tC
Else
    tStr = tStr + Str(!y_coord) + tC
End If

```

```

If IsNull(!xy_count) Then
    tStr = tStr + "0"
Else
    tStr = tStr + Str(!xy_count)
End If

```

```

tStream.WriteText tStr, adWriteLine
.MoveNext

```

```

Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite

```

```

Else
    'The user pressed Cancel.
End If

```

```

Set tRstNode = Nothing

```

```

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

```

```

Exit_CmdGIS_Click:
Exit Sub

```

```

Err_CmdGIS_Click:
MsgBox Err.Description
Resume Exit_CmdGIS_Click

```

```

End Sub

```

```

Private Sub CmdSaveTextCategories_Click()
On Error GoTo Err_CmdSaveTextCategories_Click
'
'   This program will store the current list of office IDs to a .txt file
'
Dim tStream As ADODB.Stream, tStreamNoBOM As ADODB.Stream
Set tStream = New ADODB.Stream

tStream.Charset = "utf-8"
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open

Set tStreamNoBOM = New ADODB.Stream
tStreamNoBOM.Type = adTypeBinary
tStreamNoBOM.Open
'
'   next get a file
'

Dim dlgSaveAs As FileDialog
Dim tFileNum As Integer
Dim tFileName As String, tFN As Variant, tFemale As String
Dim tRstIDs As DAO.Recordset
Dim tStr As String, tTab As String, ti As Integer
Dim tFileSystem, tGDF

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "text_bilblcat_list.txt"
If dlgSaveAs.Show = -1 Then
'
'   tFileName = ""
For Each tFN In dlgSaveAs.SelectedItems
tFileName = tFN
If Not tFileName = "" Then
Exit For
End If
Next
If tFileName = "" Then
MsgBox "Bad file Name."
GoTo Exit_CmdSaveTextCategories_Click
Else
'   make sure the file name has a txt extension
If Len(tFileName) < 5 Then
tFileName = tFileName + ".txt"
ElseIf Not (LCase(Right(tFileName, 4)) = ".txt") Then
tFileName = tFileName + ".txt"
End If
End If
'
'   write the file
'
'   process the table
'

tStr = "SELECT ZZ_TEXT_BIBLCAT_CODES.c_text_cat_code, TEXT_BIBLCAT_CODES.c_text_cat_desc, TEXT_BIBLCAT_CO
DES.c_text_cat_desc_chn " + _
"FROM ZZ_TEXT_BIBLCAT_CODES INNER JOIN TEXT_BIBLCAT_CODES ON " + _
"ZZ_TEXT_BIBLCAT_CODES.c_text_cat_code = TEXT_BIBLCAT_CODES.c_text_cat_code"
Set tRstIDs = CurrentDb.OpenRecordset(tStr, dbOpenDynaset)
'
tTab = Chr(9)
With tRstIDs

.MoveFirst
' MsgBox "writing file"
Do While Not .EOF
'
tStr = Str(!c_text_cat_code) + tTab
If IsNull(!c_text_cat_desc) Then
tStr = tStr + "" + tTab
Else
tStr = tStr + !c_text_cat_desc + tTab
End If
If IsNull(!c_text_cat_desc_chn) Then
tStr = tStr + ""
Else
tStr = tStr + !c_text_cat_desc_chn
End If

```

```

        tStream.WriteText tStr, adWriteLine
        .MoveNext
    Loop
End With
' now make sure all the data is copied to tStream
tStream.Flush
tStream.Position = 3
tStream.CopyTo tStreamNoBOM
' and write the stream to the file
tStreamNoBOM.SaveToFile tFileName, adSaveCreateOverWrite
'
Else
    'The user pressed Cancel.
End If

Set tRstIDs = Nothing

tStream.Close
Set tStream = Nothing
tStreamNoBOM.Close
Set tStreamNoBOM = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_CmdSaveTextCategories_Click:
Exit Sub

Err_CmdSaveTextCategories_Click:
MsgBox Err.Description
Resume Exit_CmdSaveTextCategories_Click

End Sub

Private Sub CmdToDynasty_Click()
    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strToDynasty As String

    If gToDynasty = -1 Then
        strToDynasty = ""
    Else
        strToDynasty = Str(gToDynasty)
    End If

    stDocName = "frmPickDynasty"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strFromDynasty

    If CurrentProject.AllForms("frmPickDynasty").IsLoaded Then

        Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.SetFocus
        gToDynasty = Forms!frmpickdynasty!frmDYNASTIES.Form!Dy_Code.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.SetFocus
        gToDynastyBegin = Forms!frmpickdynasty!frmDYNASTIES.Form!c_start.Value

        Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.SetFocus
        gToDynastyEnd = Forms!frmpickdynasty!frmDYNASTIES.Form!c_end.Value
        '
        ' check to see if we have a problem and reject selection if needed
        '
        If gFromDynasty > -1 Then
            If gFromDynastyBegin > gToDynastyEnd Then
                MsgBox "Warning: There is a problem with chronology: the 'From' Dynasty begins after the 'To' Dynasty ends!", vbExclamation
                gToDynasty = -1
                TxtToDynasty.Value = ""
                TxtToDynastyPY.Value = ""
            End If
        End If
        '
        ' value is OK
        '
        If gToDynasty > -1 Then
            Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.SetFocus
            TxtToDynastyPY.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty.Value

            Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.SetFocus
            TxtToDynasty.Value = Forms!frmpickdynasty!frmDYNASTIES.Form!c_dynasty_chn.Value
        End If
    End If

```

```

        DoCmd.Close acForm, stDocName
    '
    ' reset FromDynasty if necessary (-2 = all dynasties)
    '
    If gFromDynasty = -2 Then
        gFromDynasty = -1
        TxtFromDynasty.Value = ""
        TxtFromDynastyPY.Value = ""
    End If
    '
End If

End Sub

Private Sub Form_Open(Cancel As Integer)
    Dim cmdSQL As ADODB.Command, tRecDeleted As Variant
    Dim tRstTextCode As DAO.Recordset, tRstDummy As DAO.Recordset

    Set cmdSQL = New ADODB.Command
    cmdSQL.ActiveConnection = CurrentProject.Connection
    cmdSQL.CommandType = adCmdText
    '
    ' to clear the tables, briefly close and then delete records
    '
    Set tRstTextCode = ZZ_SCRATCH_TEXT.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_TR", dbOpenDynaset)
    Set ZZ_SCRATCH_TEXT.Form.Recordset = tRstDummy
    tRstTextCode.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_BIOG_TEXT_DATA"
    cmdSQL.Execute tRecDeleted
    '
    ' now reopen
    '
    Set tRstTextCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_BIOG_TEXT_DATA", dbOpenDynaset)
    Set ZZ_SCRATCH_TEXT.Form.Recordset = tRstTextCode
    '
    Set tRstTextCode = ZZ_SCRATCH_P_TEXT.Form.Recordset
    Set tRstDummy = CurrentDb.OpenRecordset("Z_SCRATCH_DUMMY_TP", dbOpenDynaset)
    Set ZZ_SCRATCH_P_TEXT.Form.Recordset = tRstDummy
    tRstTextCode.Close
    '
    cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_P_TEXT"
    cmdSQL.Execute tRecDeleted
    '
    ' now reopen
    '
    Set tRstTextCode = CurrentDb.OpenRecordset("ZZ_SCRATCH_P_TEXT", dbOpenDynaset)
    Set ZZ_SCRATCH_P_TEXT.Form.Recordset = tRstTextCode
    '
    ' first determine the language
    gLCID = Application.LanguageSettings.LanguageID(msoLanguageIDUI)
    If gLCID = 2052 Or gLCID = 3076 Then ' 2052 = PRC, 3076 = Hong Kong
        gDisplayLanguage = "S"
    ElseIf gLCID = 4100 Or gLCID = 1028 Then ' 4100 = Singapore, 1028 = Taiwan
        gDisplayLanguage = "T"
        Call changeDisplayLanguage
    Else
        gDisplayLanguage = "E"
        Call changeDisplayLanguage
    End If

    ' set the index year default values

    gFromDynasty = -1
    gToDynasty = -1
    gFromStr = "-200"
    gToStr = "1911"
    TxtFromYear.Value = -200
    TxtToYear.Value = 1911
End Sub

Private Sub CmdFanti_Click()
On Error GoTo Err_CmdFanti_Click

    If gDisplayLanguage = "T" Then
        gDisplayLanguage = "E"
    Else

```

```

        gDisplayLanguage = "T"
    End If

    Call changeDisplayLanguage

Exit_CmdFanti_Click:
    Exit Sub

Err_CmdFanti_Click:
    MsgBox Err.Description
    Resume Exit_CmdFanti_Click

End Sub

Private Sub CmdJianti_Click()
On Error GoTo Err_CmdJianti_Click

    If gDisplayLanguage = "S" Then
        gDisplayLanguage = "E"
    Else
        gDisplayLanguage = "S"
    End If

    Call changeDisplayLanguage

Exit_CmdJianti_Click:
    Exit Sub

Err_CmdJianti_Click:
    MsgBox Err.Description
    Resume Exit_CmdJianti_Click

End Sub

Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 32) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 32 And Not .EOF
            If !c_form = "LAT" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then
            tLang = 2
        Else
            tLang = 3
        End If
        '
        ' now comes the basic routine
        '
        Me.LblFrom.Caption = tLabelLanguage(tLang, 1)
        Me.LblTo.Caption = tLabelLanguage(tLang, 2)
        Me.LblType.Caption = tLabelLanguage(tLang, 3)
    End If

```

```

Me.CmdPickTextCat.Caption = tLabelLanguage(tLang, 4)
Me.CmdQuery.Caption = tLabelLanguage(tLang, 5)
Me.CmdGIS.Caption = tLabelLanguage(tLang, 6)
Me.CmdFanti.Caption = tLabelLanguage(tLang, 7)
Me.CmdJianti.Caption = tLabelLanguage(tLang, 8)
Me.PageTexts.Caption = tLabelLanguage(tLang, 9)
Me.PagePeople.Caption = tLabelLanguage(tLang, 10)
Me.CmdSelectPlace.Caption = tLabelLanguage(tLang, 11)
Me.CmdImportPlaces.Caption = tLabelLanguage(tLang, 12)
Me.CmdAllPlaces.Caption = tLabelLanguage(tLang, 13)
Me.LblDisplay.Caption = tLabelLanguage(tLang, 14)
Me.CmdHelp.Caption = tLabelLanguage(tLang, 15)
Me.LblXYRef.Caption = tLabelLanguage(tLang, 16)
Me.LblNarrow.Caption = tLabelLanguage(tLang, 17)
Me.LblBroad.Caption = tLabelLanguage(tLang, 18)
Me.CmdStoreID.Caption = tLabelLanguage(tLang, 19)
Me.LblChkSubUnits.Caption = tLabelLanguage(tLang, 20)
Me.LblDynasties.Caption = tLabelLanguage(tLang, 21)
Me.CmdFromDynasty.Caption = tLabelLanguage(tLang, 22)
Me.CmdToDynasty.Caption = tLabelLanguage(tLang, 23)
Me.CmdAllDynasties.Caption = tLabelLanguage(tLang, 24)
Me.LblIndexYears.Caption = tLabelLanguage(tLang, 25)
Me.LblNoYears.Caption = tLabelLanguage(tLang, 26)
Me.LblUseIndexYears.Caption = tLabelLanguage(tLang, 27)
Me.LblUseDynasty.Caption = tLabelLanguage(tLang, 28)

Me.CmdNeo4j.Caption = tLabelLanguage(tLang, 29)
Me.CmdImportTextCategories.Caption = tLabelLanguage(tLang, 30)
Me.CmdSaveTextCategories.Caption = tLabelLanguage(tLang, 31)

```

```
End If
```

```
End Sub
```

```
Private Sub CmdSelectPlace_Click()
```

```
On Error GoTo Err_CmdSelectPlace_Click
```

```

Dim stDocName As String
Dim stLinkCriteria As String
Dim strADDR As String
Dim cmdSQL As ADODB.Command

```

```

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText

```

```

TxtAddrID.Visible = True
TxtAddrID.SetFocus
strADDR = TxtAddrID.Text

```

```

stDocName = "frmPickAddresses_multi"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR

```

```

If CurrentProject.AllForms("frmPickAddresses_multi").IsLoaded Then
    Dim tAddrID As Long, tRstAddr As DAO.Recordset
    Dim strADDR_CHN As String, strADDR_PY As String

```

```

gUseADDRID = True
CmdAllPlaces.Enabled = True
ChkXYRef.Enabled = True
ChkSubUnits.Enabled = True
FrameXY.Enabled = True

```

```

Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Visible = True
Forms!frmPickAddresses_multi.Form!TxtAddrFilter.SetFocus
If Forms!frmPickAddresses_multi.Form!TxtAddrFilter.Value Then
    TxtAddrID.Value = 0
    strADDR_PY = Forms!frmPickAddresses_multi.Form!TxtFilterPY
    strADDR_CHN = Forms!frmPickAddresses_multi.Form!TxtFilterChn

```

```

    If strADDR_CHN = "" Then
        TxtPlaceChn.Value = "[[Filter]]"
        TxtPlace.Value = "[" + strADDR_PY + "]"
    Else
        TxtPlaceChn.Value = "[" + strADDR_CHN + "]"
        TxtPlace.Value = "[[Filter]]"
    End If

```

```

Else
    Forms!frmPickAddresses_multi.Form!TxtSelectCount.Visible = True
    Forms!frmPickAddresses_multi.Form!TxtSelectCount.SetFocus
    If Forms!frmPickAddresses_multi.Form!TxtSelectCount.Value > 1 Then

```

```

        TxtPlaceChn.Value = "[" + ChrW(22810) + ChrW(36984) + "]"
        TxtPlace.Value = "[[Multi-Select]]"
        TxtAddrID.Value = 0
    Else
        ' only one record in ZZ_ADDRESSES: get its field values
        '
        Set tRstAddr = CurrentDb.OpenRecordset("ZZ_ADDRESSES", dbOpenDynaset)
        tRstAddr.MoveFirst
        'MsgBox "Checking zz_addresses: no records"
        TxtAddrID.Value = tRstAddr!c_addr_id
        TxtPlaceChn.Value = tRstAddr!c_name_chn
        TxtPlace.Value = tRstAddr!c_name
        tRstAddr.Close
        Set tRstAddr = Nothing
    End If
End If
'
' now copy the records
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_LIST"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT " + _
    "ZZ_ADDRESSES.c_addr_id FROM ZZ_ADDRESSES"
cmdSQL.Execute tRecDeleted
End If

DoCmd.Close acForm, stDocName
CmdSelectPlace.SetFocus
TxtAddrID.Visible = False

Exit_CmdSelectPlace_Click:
Exit Sub

Err_CmdSelectPlace_Click:
MsgBox Err.Description
Resume Exit_CmdSelectPlace_Click

End Sub

Private Sub CmdAllPlaces_Click()
On Error GoTo Err_CmdAllPlaces_Click

    TxtAddrID.Value = -1

    TxtPlaceChn.Value = ""
    TxtPlace.Value = ""
    gUseADDRID = False
    ChkXYRef.Enabled = False
    ChkSubUnits.Enabled = False
    FrameXY.Enabled = False

Exit_CmdAllPlaces_Click:
Exit Sub

Err_CmdAllPlaces_Click:
MsgBox Err.Description
Resume Exit_CmdAllPlaces_Click

End Sub

Private Sub CmdImportPlaces_Click()
On Error GoTo Err_CmdImportPlaces_Click

    Dim stDocName As String, tRstAddresses As DAO.Recordset, tRstImportPlaces As DAO.Recordset
    Dim stLinkCriteria As String
    Dim tString As String, tAddrID As Long, ti As Integer, tStrID As String, tLen As Integer, tQuit As Boolean

    Dim dlgSaveAs As FileDialog
    Dim tFileNum As Integer
    Dim tFileName As String, tFN As Variant
    Dim tFileSystem, tList

    tQuit = False
    '
    If Not tQuit Then
        ' open the list

        Set dlgSaveAs = Application.FileDialog(msoFileDialogOpen)

        'Use a With...End With block to reference the FileDialog object.
        With dlgSaveAs

```



```

.InitialFileName = ""
If .Show = -1 Then
    '
    tFileName = ""
    For Each tFN In .SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_CmdImportPlaces_Click
    End If
End If
End With
'
' Clear the address table now that we are ready to go
'
Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
cmdSQL.CommandText = "Delete * from ZZ_SCRATCH_ADDR_LIST"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from InputErrorList"
cmdSQL.Execute tRecDeleted

cmdSQL.CommandText = "Delete * from TempImportList"
cmdSQL.Execute tRecDeleted

DoCmd.TransferText acImportDelim, "ImportPlaceList_Space", "TempImportList", tFileName, 0
' TransferType=acImportDelim
' SpecificationName = "TempImportList" (apparently it is saved in the database itself)
' TableName = "TempImportList" (probably requires that I drop the table first, but I can test)
' HasFieldNames = False (0)
'
' copy the bad IDs
'
tStrSQL = "INSERT INTO InputErrorList ( c_ID ) SELECT TempImportList.ImportID " + _
    "FROM ADDR_CODES RIGHT JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID " + _
    "WHERE ((ADDR_CODES.c_addr_id) Is Null))"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    MsgBox "Some ID were not successfully imported: please look at InputErrorList."
End If
'
' copy the good IDs
'
tStrSQL = "INSERT INTO ZZ_SCRATCH_ADDR_LIST ( c_addr_id ) SELECT DISTINCT TempImportList.ImportID " + _
    "FROM ADDR_CODES INNER JOIN TempImportList ON ADDR_CODES.c_addr_id = TempImportList.ImportID"

cmdSQL.CommandText = tStrSQL
cmdSQL.Execute tRecDeleted

If tRecDeleted > 0 Then
    Me.TxtPlace.Value = "[Imported List]"
    Me.TxtPlaceChn.Value = "[Imported List]"
    gUseADDRID = False
    ChkXYRef.Enabled = True
    ChkSubUnits.Enabled = True
    FrameXY.Enabled = True
End If

Set cmdSQL = Nothing
Set tFileSystem = Nothing
End If

Exit_CmdImportPlaces_Click:
Exit Sub

Err_CmdImportPlaces_Click:
MsgBox Err.Description
Resume Exit_CmdImportPlaces_Click

End Sub

```

```
Private Sub FrameFilterYears_Click()
```

```
    ' disable all
    Me.CmdFromDynasty.Enabled = False
    Me.CmdToDynasty.Enabled = False
    Me.CmdAllDynasties.Enabled = False
    Me.TxtFromDynasty.Enabled = False
    Me.TxtFromDynastyPY.Enabled = False
    Me.TxtToDynasty.Enabled = False
    Me.TxtToDynastyPY.Enabled = False
    Me.TxtFromDynasty.Locked = False
    Me.TxtFromDynastyPY.Locked = False
    Me.TxtToDynasty.Locked = False
    Me.TxtToDynastyPY.Locked = False
```

```
    Me.TxtFromYear.Enabled = False
    Me.TxtToYear.Enabled = False
```

```
    If FrameFilterYears.Value = 2 Then
        ' enable index years
        Me.TxtFromYear.Enabled = True
        Me.TxtToYear.Enabled = True
    ElseIf FrameFilterYears.Value = 3 Then
        ' enable dynasties
        Me.CmdFromDynasty.Enabled = True
        Me.CmdToDynasty.Enabled = True
        Me.CmdAllDynasties.Enabled = True
        Me.TxtFromDynasty.Enabled = True
        Me.TxtFromDynastyPY.Enabled = True
        Me.TxtToDynasty.Enabled = True
        Me.TxtToDynastyPY.Enabled = True
        Me.TxtFromDynasty.Locked = True
        Me.TxtFromDynastyPY.Locked = True
        Me.TxtToDynasty.Locked = True
        Me.TxtToDynastyPY.Locked = True
```

```
    End If
```

```
End Sub
```

```
Private Sub TxtFromYear_LostFocus()
    gFromStr = Trim(TxtFromYear.Text)
End Sub
```

```
Private Sub TxtToYear_LostFocus()
    gToStr = Trim(TxtToYear.Text)
End Sub
```

```
Private Sub CmdHelp_Click()
    Dim tStrPDF As String
```

```
    tStrPDF = Application.CurrentProject.Path + "\HelpFiles\HelpFile_LookAtTextCatiations.pdf"
```

```
    'MsgBox tStrPDF
```

```
    Application.FollowHyperlink tStrPDF, , True
```

```
End Sub
```

```
Private Sub writeKML()
    On Error GoTo Err_writeKML
```

```
    ' This program will dump the results to a .gis file
```

```
    If ZZ_SCRATCH_P_TEXT.Form.Recordset.RecordCount = 0 Then
        MsgBox "There are no records to save."
        GoTo Exit_writeKML
    End If
```

```
    Dim tStream As ADODB.Stream
    Set tStream = New ADODB.Stream
```

```
    tPinyin = False
```

```
    If GISFrame.Value = 1 Then
        tStream.Charset = "utf-8"
        tCodeStr = "UTF8"
```

```
    ElseIf GISFrame.Value = 2 Then
        tStream.Charset = "ascii"
        tCodeStr = "ASCII"
```

```

        tPinyin = True
Else
    tStream.Charset = "gb18030"
    tCodeStr = "GB18030"
End If
tStream.Mode = adModeReadWrite
tStream.Type = adTypeText
tStream.Open
'
' next get a file
'

Set dlgSaveAs = Application.FileDialog(msoFileDialogSaveAs)

dlgSaveAs.InitialFileName = "network_gis_" + tCodeStr + ".kml"
If dlgSaveAs.Show = -1 Then
    '
    tFileName = ""
    For Each tFN In dlgSaveAs.SelectedItems
        tFileName = tFN
        If Not tFileName = "" Then
            Exit For
        End If
    Next
    If tFileName = "" Then
        MsgBox "Bad file Name."
        GoTo Exit_writeKML
    Else
        ' make sure the file name has a txt extension
        If Len(tFileName) < 5 Then
            tFileName = tFileName + ".kml"
        ElseIf Not (LCase(Right(tFileName, 4)) = ".kml") Then
            tFileName = tFileName + ".kml"
        End If
    End If
    '
    ' write the file
    '
    'Name,NameChn,Female,IndexYear,AddrName,AddrChn,X,Y,xy_count
    '
    ' process the table
    '
    Set tRstNode = ZZ_SCRATCH_P_TEXT.Form.Recordset
    tC = Chr(9) ' the tab
    tDQ = Chr(34) ' the double quotation mark
    '
    ' write the header
    '
    tStream.WriteText "<kml xmlns=" + tDQ + "http://www.opengis.net/kml/2.2" + tDQ + ">", adWriteLine
    tStream.WriteText "<Document>", adWriteLine
    tStream.WriteText tC + "<name>ExtendedData</name>", adWriteLine
    tStream.WriteText tC + "<open>1</open>", adWriteLine
    tStream.WriteText tC + "<!-- Create a balloon template referring to the user-defined type -->", adWriteLine

    tStream.WriteText tC + "<Style id=" + tDQ + "TextCat-balloon-template" + tDQ + ">", adWriteLine
    tStream.WriteText tC + tC + "<BalloonStyle>", adWriteLine
    tStream.WriteText tC + tC + tC + "<text>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "<![CDATA[" + tC + "ID: $[TextCatGIS/PersonID] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Name Chn: $[TextCatGIS/NameHZ] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Index Year: $[TextCatGIS/IndexYear] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Sex: $[TextCatGIS/Sex] <br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "Address: $[TextCatGIS/AddrName] $[TextCatGIS/AddrHZ] <br/>", adWriteLine

    tStream.WriteText tC + tC + tC + tC + "XY Count: $[TextCatGIS/XYCount] <br/><br/>", adWriteLine
    tStream.WriteText tC + tC + tC + tC + "]]>", adWriteLine
    tStream.WriteText tC + tC + tC + "</text>", adWriteLine
    tStream.WriteText tC + tC + "</BalloonStyle>", adWriteLine
    tStream.WriteText tC + "</Style>", adWriteLine
    tStream.WriteText tC + "<!-- Declare the type " + tDQ + "TextCatGIS" + tDQ + " with 6 fields -->", adWriteLine

    tStream.WriteText tC + "<Schema name=" + tDQ + "TextCatGIS" + tDQ + " id=" + tDQ + "TextCatGISId" + tDQ +
">", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "uint" + tDQ + " name=" + tDQ + "PersonID" + tDQ +
">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Person ID]]></displayName>", adWriteLine
    tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
    tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "NameHZ" + tDQ +
">", adWriteLine
    tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Name Chn]]></displayName>", adWriteLine

```

```

        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "Sex" + tDQ +
">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Sex]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrName" + t
DQ + ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "AddrHZ" + tDQ
+ ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Address Chn]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "string" + tDQ + " name=" + tDQ + "IndexYear" +
tDQ + ">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[Index Year]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "<SimpleField type=" + tDQ + "int" + tDQ + " name=" + tDQ + "XYCount" + tDQ +
">", adWriteLine
        tStream.WriteText tC + tC + tC + "<displayName><![CDATA[XY Count]]></displayName>", adWriteLine
        tStream.WriteText tC + tC + "</SimpleField>", adWriteLine
        tStream.WriteText tC + tC + "</Schema>", adWriteLine

With tRstNode
    .MoveFirst
    Do While Not .EOF
        ' must guard against NULLs, even where there should not be any
        '
        ' write the point header
        '
        tStream.WriteText tC + "<Placemark>", adWriteLine

        If IsNull(!c_name) Then
            tStr = "[Bad Data] "
        Else
            tStr = !c_name
        End If
        tStream.WriteText tC + tC + "<name>" + tStr + "</name>", adWriteLine

        tStream.WriteText tC + tC + "<styleUrl>#TextCat-balloon-template</styleUrl>", adWriteLine
        '
        ' First Year as time stamp
        '
        If IsNull(!c_index_year) Then
            tStr = "N/A"
        Else
            tStr = Str(!c_index_year)
        End If
        tStream.WriteText tC + tC + "<TimeStamp>" + tStr + "</TimeStamp>", adWriteLine
        '
        tStream.WriteText tC + tC + "<ExtendedData>", adWriteLine
        tStream.WriteText tC + tC + tC + "<SchemaData schemaUrl=" + tDQ + "#TextCatGISId" + tDQ + ">", ad
WriteLine
        '
        ' person ID
        '
        tStr = Str(!c_person_id)
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "PersonID" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
        '
        ' Person Name Chn
        '
        If IsNull(!c_name_chn) Then
            tStr = tStr + "[Bad Data]"
        Else
            If Trim(!c_name_chn) = "" Then
                tStr = "[?]"
            Else
                tStr = !c_name_chn
            End If
        End If
        tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "NameHZ" + tDQ + ">" + tStr +
"</SimpleData>", adWriteLine
        '
        ' Index Year
        '
        If IsNull(!c_index_year) Then
            tStr = "N/A"
        Else

```

```

        tStr = Str(!c_index_year)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "IndexYear" + tDQ + ">" + tStr
+ "</SimpleData>", adWriteLine
    '
    If IsNull(!c_sex) Then
        tStr = "[?]"
    Else
        tStr = !c_sex
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "Sex" + tDQ + ">" + tStr + "</S
impleData>", adWriteLine
    '
    ' Address Name
    '
    If IsNull(!c_addr_name) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_name) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_name
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrName" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
    '
    ' Address Name Chinese
    '
    If IsNull(!c_addr_chn) Then
        tStr = "[?]"
    ElseIf Trim(!c_addr_chn) = "" Then
        tStr = "[?]"
    Else
        tStr = !c_addr_chn
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "AddrHZ" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
    '
    ' XY Count
    '
    If IsNull(!xy_count) Then
        tStr = "0"
    Else
        tStr = Str(!xy_count)
    End If
    tStream.WriteText tC + tC + tC + tC + "<SimpleData name=" + tDQ + "XYCount" + tDQ + ">" + tStr + "
</SimpleData>", adWriteLine
    '
    tStream.WriteText tC + tC + tC + "</SchemaData>", adWriteLine
    tStream.WriteText tC + tC + "</ExtendedData>", adWriteLine
    tStream.WriteText tC + tC + "<Point>", adWriteLine
    '
    ' coordinates
    If IsNull(!x_coord) Then
        tStr = "0"
    Else
        tStr = Str(!x_coord)
    End If

    If IsNull(!y_coord) Then
        tStr = tStr + ",0"
    Else
        tStr = tStr + "," + Str(!y_coord)
    End If
    tStream.WriteText tC + tC + tC + "<coordinates>" + tStr + "</coordinates>", adWriteLine
    '
    ' footer
    '
    tStream.WriteText tC + tC + "</Point>", adWriteLine
    tStream.WriteText tC + tC + "</Placemark>", adWriteLine
    .MoveNext
Loop
End With
'
' footer
'
tStream.WriteText "</Document>", adWriteLine
tStream.WriteText "</kml>", adWriteLine
Else
    'The user pressed Cancel.
End If

```

```

' now make sure all the data is copied to tStream
tStream.Flush
' and write the stream to the file
tStream.SaveToFile tFileName, adSaveCreateOverWrite
'
Set tRstNode = Nothing

tStream.Close
Set tStream = Nothing
'Set the object variable to Nothing.
Set dlgSaveAs = Nothing

Exit_writeKML:
Exit Sub

Err_writeKML:
MsgBox Err.Description
Resume Exit_writeKML

End Sub

Private Sub CmdStoreID_Click()
Dim cmdSQL As ADODB.Command, tRecCount As Variant

Set cmdSQL = New ADODB.Command
cmdSQL.ActiveConnection = CurrentProject.Connection
cmdSQL.CommandType = adCmdText
'
If DCount("*", "ZZ_STORE_PERSON_ID") > 0 Then
' Display message.
If MsgBox("Do you wish to replace the current stored values?", vbYesNo + vbQuestion + vbDefaultButton2) =
vbNo Then
Exit Sub
Else
cmdSQL.CommandText = "Delete * from ZZ_STORE_PERSON_ID"
cmdSQL.Execute tRecCount
End If
End If

tStrQuery = "INSERT INTO ZZ_STORE_PERSON_ID ( c_personid ) SELECT DISTINCT ZZ_SCRATCH_P_TEXT.c_person_id FROM
ZZ_SCRATCH_P_TEXT"

cmdSQL.CommandText = tStrQuery
cmdSQL.Execute tRecCount
MsgBox "Person IDs successfully stored. Click on 'Recall Person IDs' to reuse these IDs in other forms."
'
' update storage source
cmdSQL.CommandText = "UPDATE PersonIDSource SET SourceForm ='Texts' WHERE PersonIDSource.LineNum =1"
cmdSQL.Execute tRecCount

End Sub

```

```

Option Compare Database
Option Explicit
Public gPersonIDsaveSourceStr As String

Private Sub CmdInput_Click()
On Error GoTo Err_CmdInput_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "CBDB_Editor"
    DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdInput_Click:
Exit Sub

Err_CmdInput_Click:
MsgBox Err.Description
Resume Exit_CmdInput_Click

End Sub

Private Sub CmdEnterTxt_Click()
On Error GoTo Err_CmdEnterTxt_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "TEXTS EnterForm"
    DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdEnterTxt_Click:
Exit Sub

Err_CmdEnterTxt_Click:
MsgBox Err.Description
Resume Exit_CmdEnterTxt_Click

End Sub

Private Sub CmdEnterEvents_Click()
On Error GoTo Err_CmdEnterEvents_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "EVENT_CODES EnterForm"
    DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdEnterEvents_Click:
Exit Sub

Err_CmdEnterEvents_Click:
MsgBox Err.Description
Resume Exit_CmdEnterEvents_Click

End Sub

Private Sub CmdBrowse_Click()
On Error GoTo Err_CmdBrowse_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "CBDB_Browser_2"
    DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdBrowse_Click:
Exit Sub

Err_CmdBrowse_Click:
MsgBox Err.Description
Resume Exit_CmdBrowse_Click

End Sub

Private Sub CmdEnterPpl_Click()
On Error GoTo Err_CmdEnterPpl_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "CBDB_Editor"

```

```

DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdEnterPpl_Click:
Exit Sub

Err_CmdEnterPpl_Click:
MsgBox Err.Description
Resume Exit_CmdEnterPpl_Click

End Sub

Private Sub CmdIndexAddr_Click()
Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "FrmIndexAddr"
DoCmd.OpenForm stDocName, , , stLinkCriteria

End Sub

Private Sub CmdLinkTables_Click()
On Error GoTo Err_CmdLinkTables

Dim tRstLinkInit As DAO.Recordset
Dim tStrPath As String, tStrPathBase As String, tStrDataBaseVersion As String, tdf As TableDef, _
tRstLinkedTable As DAO.Recordset, tCurrentDB As Database, tStrUserType As String, tStrDataFile As String, _
tNameLen As Integer

' get the current dataset

Set tRstLinkInit = CurrentDb.OpenRecordset("LinkListInit", dbOpenDynaset)

tRstLinkInit.MoveFirst
tStrDataBaseVersion = tRstLinkInit!c_dataset

' Open the form to get the Data file number

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "frmGetDataVersion"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, tStrDataBaseVersion

If CurrentProject.AllForms("frmGetDataVersion").IsLoaded Then
Forms!frmGetDataVersion.Form!c_data_version.SetFocus
tStrDataBaseVersion = Trim(Forms!frmGetDataVersion.Form!c_data_version.Value)

DoCmd.Close acForm, stDocName
End If

If Len(tStrDataBaseVersion) = 8 Then
'MsgBox "Beginning"
tStrPath = CurrentProject.FullName

'MsgBox tStrPath

If InStr(UCase(tStrPath), "ADMIN") > 0 Then
tStrUserType = "ADMIN"
tNameLen = 13
Else
tStrUserType = "User"
tNameLen = 12
End If

tStrPathBase = Left(tStrPath, Len(tStrPath) - tNameLen) + "_" + Trim(tStrDataBaseVersion) + "_DATA"

'MsgBox tStrPathBase

Set tRstLinkedTable = CurrentDb.OpenRecordset("LinkedTables", dbOpenDynaset)
'MsgBox "Table opened"

' define the database
Set tCurrentDB = CurrentDb

With tRstLinkedTable
.MoveFirst
Do While Not .EOF
'MsgBox "Linking " + !c_table_name + " to " + tStrPathBase + Trim(!c_data_file) + ".mdb"
tStrDataFile = Trim(!c_data_file)

```



```

'
' test if this is an ADMIN (data file = 5 or 6) or User table (data file = 1, 2, or 3)
'
If InStr("12345", tStrDataFile) > 0 Then
    Set tdf = tCurrentDB.TableDefs(Trim(!c_table_name))
    tdf.Connect = "MS Access;DATABASE=" + tStrPathBase + tStrDataFile + ".mdb"
    tdf.RefreshLink
ElseIf tStrDataFile = "6" And tStrUserType = "ADMIN" Then
    Set tdf = tCurrentDB.TableDefs(Trim(!c_table_name))
    tdf.Connect = "MS Access;DATABASE=" + tStrPathBase + "1" + ".mdb"
    tdf.RefreshLink
ElseIf tStrDataFile = "7" And tStrUserType = "ADMIN" Then
    Set tdf = tCurrentDB.TableDefs(Trim(!c_table_name))
    tdf.Connect = "MS Access;DATABASE=" + tStrPathBase + "2" + ".mdb"
    tdf.RefreshLink
End If
'MsgBox "TDF connection set"
.MoveNext
Loop
End With

tRstLinkedTable.Close
Set tRstLinkedTable = Nothing
Set tCurrentDB = Nothing

' reset the link initialization data

tRstLinkInit.Edit
tRstLinkInit!c_path = CurrentProject.FullName
tRstLinkInit!c_dataset = tStrDataBaseVersion
tRstLinkInit.Update

Set tdf = Nothing
Else
    MsgBox "The dataset ID " + tStrDataBaseVersion + " does not have the correct format (YYYYMMDD)."
End If

tRstLinkInit.Close
Set tRstLinkInit = Nothing

Exit_CmdLinkTables:
Exit Sub

Err_CmdLinkTables:
MsgBox Err.Description
Resume Exit_CmdLinkTables

End Sub

Private Sub CmdQueryGroup_Click()
On Error GoTo Err_CmdQueryGroup_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "LookAtGroupData"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdQueryGroup_Click:
Exit Sub

Err_CmdQueryGroup_Click:
MsgBox Err.Description
Resume Exit_CmdQueryGroup_Click

End Sub

Private Sub CmdQueryStatus_Click()
On Error GoTo Err_CmdQueryStatus_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "LookAtStatus"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdQueryStatus_Click:
Exit Sub

Err_CmdQueryStatus_Click:

```

```

    MsgBox Err.Description
    Resume Exit_CmdQueryStatus_Click
End Sub

Private Sub CmdQueryTexts_Click()
On Error GoTo Err_CmdQueryTexts_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "LookAtTexts"
    DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdQueryTexts_Click:
    Exit Sub

Err_CmdQueryTexts_Click:
    MsgBox Err.Description
    Resume Exit_CmdQueryTexts_Click
End Sub

Private Sub Form_Open(Cancel As Integer)
On Error GoTo Err_SetLinkTables

    Dim tRstLinkInit As DAO.Recordset
    Dim tStrPath As String, tStrPathBase As String, tStrDataBaseVersion As String, tdf As New DAO.TableDef, _
        tRstLinkedTable As DAO.Recordset, tContinue As Boolean, tCurrentDB As Database, tStrUserType As String, t
    StrDataFile As String, _
        tNameLen As Integer

    Set tRstLinkInit = CurrentDb.OpenRecordset("LinkListInit", dbOpenDynaset)

    tRstLinkInit.MoveFirst

    ' get the current path: if it matches the stored string, there is nothing to do
    If IsNull(tRstLinkInit!c_path) Then
        tContinue = True
    Else
        tStrPath = tRstLinkInit!c_path
        If tStrPath = CurrentProject.FullName Then
            tRstLinkInit.Close
            Set tRstLinkInit = Nothing
            Exit Sub
        End If
    End If

    ' get the current dataset, which should be set at download
    Set tCurrentDB = CurrentDb

    tStrDataBaseVersion = tRstLinkInit!c_dataset

    If IsNull(tStrDataBaseVersion) Or IsNull(tRstLinkInit!c_path) Then
        Call CmdLinkTables_Click
    Else
        If Len(tStrDataBaseVersion) = 8 Then
            tStrPath = CurrentProject.FullName

            Set tRstLinkedTable = CurrentDb.OpenRecordset("LinkedTables", dbOpenDynaset)
            'MsgBox "Table opened"

            If InStr(UCase(tStrPath), "ADMIN") > 0 Then
                tStrUserType = "ADMIN"
                tNameLen = 13
            Else
                tStrUserType = "User"
                tNameLen = 12
            End If
            'MsgBox "User type = " + tStrUserType

            tStrPathBase = Left(tStrPath, Len(tStrPath) - tNameLen) + "_" + Trim(tStrDataBaseVersion) + "_DATA"

            'MsgBox tStrPathBase

            ' define the database
            Set tCurrentDB = CurrentDb
            'MsgBox "Database defined"

```

```

With tRstLinkedTable
    .MoveFirst
    Do While Not .EOF
        'MsgBox "Linking " + !c_table_name + " to " + tStrPathBase + Trim(!c_data_file) + ".mdb"
        tStrDataFile = Trim(!c_data_file)
        'MsgBox !c_table_name
        '
        ' test if this is an ADMIN (data file = 5 or 6) or User table (data file = 1, 2, or 3)
        '
        If InStr("123", tStrDataFile) > 0 Then
            Set tdf = tCurrentDB.TableDefs(Trim(!c_table_name))
            tdf.Connect = "MS Access;DATABASE=" + tStrPathBase + tStrDataFile + ".mdb"
            tdf.RefreshLink
        ElseIf tStrDataFile = "5" And tStrUserType = "ADMIN" Then
            Set tdf = tCurrentDB.TableDefs(Trim(!c_table_name))
            tdf.Connect = "MS Access;DATABASE=" + tStrPathBase + "1" + ".mdb"
            tdf.RefreshLink
        ElseIf tStrDataFile = "6" And tStrUserType = "ADMIN" Then
            Set tdf = tCurrentDB.TableDefs(Trim(!c_table_name))
            tdf.Connect = "MS Access;DATABASE=" + tStrPathBase + "2" + ".mdb"
            tdf.RefreshLink
        End If
        'MsgBox "TDF connection set"
        .MoveNext
    Loop
End With
tRstLinkedTable.Close
Set tRstLinkedTable = Nothing

' reset the link path

tRstLinkInit.Edit
tRstLinkInit!c_path = CurrentProject.FullName
tRstLinkInit.Update

Set tdf = Nothing
Else
    Call CmdLinkTables_Click
End If
End If

tRstLinkInit.Close
Set tRstLinkInit = Nothing
Set tCurrentDB = Nothing

' define the one global variable

gPersonIDsaveSourceStr = ""

Exit_SetLinkTables:
Exit Sub

Err_SetLinkTables:
MsgBox Err.Description
Resume Exit_SetLinkTables

End Sub

Private Sub CmdLookAtOffice_Click()
On Error GoTo Err_CmdLookAtOffice_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "LookAtOffice"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdLookAtOffice_Click:
Exit Sub

Err_CmdLookAtOffice_Click:
MsgBox Err.Description
Resume Exit_CmdLookAtOffice_Click

End Sub

Private Sub CmdPlace_Click()
On Error GoTo Err_CmdPlace_Click

```

```

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "LookAtPlace"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdPlace_Click:
Exit Sub

Err_CmdPlace_Click:
MsgBox Err.Description
Resume Exit_CmdPlace_Click

End Sub

Private Sub CmdQueryAssoc_Click()
On Error GoTo Err_CmdQueryAssoc_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "LookAtAssociations"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdQueryAssoc_Click:
Exit Sub

Err_CmdQueryAssoc_Click:
MsgBox Err.Description
Resume Exit_CmdQueryAssoc_Click

End Sub

Private Sub CmdQueryEntry_Click()
On Error GoTo Err_CmdQueryEntry_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "LookAtEntry"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdQueryEntry_Click:
Exit Sub

Err_CmdQueryEntry_Click:
MsgBox Err.Description
Resume Exit_CmdQueryEntry_Click

End Sub

Private Sub CmdQueryKinship_Click()
On Error GoTo Err_CmdQueryKinship_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "LookAtKinship"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdQueryKinship_Click:
Exit Sub

Err_CmdQueryKinship_Click:
MsgBox Err.Description
Resume Exit_CmdQueryKinship_Click

End Sub

Private Sub CmdQueryNetwork_Click()
On Error GoTo Err_CmdQueryNetwork_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "LookAtNetworks"
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdQueryNetwork_Click:
Exit Sub

```

```

Err_CmdQueryNetwork_Click:
    MsgBox Err.Description
    Resume Exit_CmdQueryNetwork_Click

End Sub

Private Sub CmdClose_Click()
On Error GoTo Err_CmdClose_Click

    DoCmd.Close

Exit_CmdClose_Click:
    Exit Sub

Err_CmdClose_Click:
    MsgBox Err.Description
    Resume Exit_CmdClose_Click

End Sub

Private Sub CmdEnterPplAlt_Click()
On Error GoTo Err_CmdEnterPplAlt_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "CBDB_Browser"
    DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdEnterPplAlt_Click:
    Exit Sub

Err_CmdEnterPplAlt_Click:
    MsgBox Err.Description
    Resume Exit_CmdEnterPplAlt_Click

End Sub

Private Sub CmdQueryPairAssoc_Click()
On Error GoTo Err_CmdQueryPairAssoc_Click

    Dim stDocName As String
    Dim stLinkCriteria As String

    stDocName = "LookAtAssociationPairs"
    DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_CmdQueryPairAssoc_Click:
    Exit Sub

Err_CmdQueryPairAssoc_Click:
    MsgBox Err.Description
    Resume Exit_CmdQueryPairAssoc_Click

End Sub

Private Sub CmdUsersGuideEng_Click()
On Error GoTo Err_CmdUsersGuideEng_Click

    Dim tStrPDF As String

    tStrPDF = Application.CurrentProject.Path + "\HelpFiles\CBDB Users Guide.pdf"

    'MsgBox tStrPDF

    Application.FollowHyperlink tStrPDF, , True

Exit_CmdUsersGuideEng_Click:
    Exit Sub

Err_CmdUsersGuideEng_Click:
    MsgBox Err.Description
    Resume Exit_CmdUsersGuideEng_Click

End Sub

Private Sub LinkTables()
On Error GoTo Err_LinkTables

```

```

Dim tPath As String, tPathBase As String, tDataBaseVersion As String, tdf As New DAO.TableDef, _
    tRstLinkedTable As DAO.Recordset

' Open the form to get the Data file number

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = "frmGetDataVersion"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog

If CurrentProject.AllForms("frmGetDataVersion").IsLoaded Then
    Forms!frmGetDataVersion.Form!c_data_version.SetFocus
    tDataBaseVersion = Forms!frmGetDataVersion.Form!c_data_version.Text

    DoCmd.Close acForm, stDocName
End If

tPath = CurrentProject.FullName
tPathBase = "MS Access;DATABASE=" + Left(tPath, Len(tPath) - 13) + "_" + Trim(tDataBaseVersion) + "_DATA"

Set tRstLinkedTable = CurrentDb.OpenRecordset("LinkedTables", dbOpenDynaset)

With tRstLinkedTable
    .MoveFirst
    Do While Not .EOF
        If Trim(!c_base_file) = "6" Then
            .MoveNext
        Else
            Set tdf = CurrentDb.TableDefs(!c_table_name)
            tdf.Connect = tPathBase + Trim(!c_base_file) + ".mdb"
            tdf.RefreshLink
            .MoveNext
        End If
    Loop
End With

tRstLinkedTable.Close
Set tRstLinkedTable = Nothing
Set tdf = Nothing

Exit_LinkTables:
Exit Sub

Err_LinkTables:
MsgBox Err.Description
Resume Exit_LinkTables

End Sub

```

frm_POSSESSION_DATA Subform 1

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Private Sub c_possession_desc_chn_AfterUpdate()

Dim intLastRecID As Long

Dim intRecID As Long

If IsNull(c_possession_record_id.Value) And Not IsNull(c_possession_desc_chn.Value) Then
intLastRecID = DMax("c_possession_record_id", "POSSESSION_DATA")
LastRecordID.Value = intLastRecID

LastRecordID.Visible = True

LastRecordID.SetFocus

intRecID = LastRecordID.Value

c_possession_record_id.Value = intRecID + 1

c_possession_record_id.SetFocus

LastRecordID.Visible = False

End If

End Sub

Private Sub CmdPickAddr_Click()

On Error GoTo Err_CmdPickAddr_Click

Dim stDocName As String

Dim stLinkCriteria As String

Dim strADDR As String

c_addr_id.Visible = True

c_addr_id.SetFocus

strADDR = c_addr_id.Text

stDocName = "frmPickADDRESSES"

DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR

If CurrentProject.AllForms("frmPickADDRESSES").IsLoaded Then

Dim intAddr As Long

Dim strADDR_CHN As String

Dim strADDR_PY As String

Forms!frmPickADDRESSES!frmADDRESSES.Form!c_addr_id.SetFocus

intAddr = Forms!frmPickADDRESSES!frmADDRESSES.Form!c_addr_id.Value

c_addr_id.Value = intAddr

Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name_chn.SetFocus

strADDR_CHN = Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name_chn.Value

TxtAddrCHN.Value = strADDR_CHN

Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name.SetFocus

strADDR_PY = Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name.Value

TxtAddrPY.Value = strADDR_PY

DoCmd.Close acForm, stDocName

End If

CmdPickAddr.SetFocus

c_addr_id.Visible = False

Exit_CmdPickAddr_Click:

Exit Sub

Err_CmdPickAddr_Click:

MsgBox Err.Description

Resume Exit_CmdPickAddr_Click

End Sub

Private Sub CmdPickNH_Click()

On Error GoTo Err_CmdPickNH_Click

Dim stDocName As String

Dim stLinkCriteria As String

Dim strNH As String

c_possession_nh_code.Visible = True

c_possession_nh_code.SetFocus

strNH = c_possession_nh_code.Text

stDocName = "frmPickNIAN_HAO"

DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

```

    If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
        Dim intNH As Integer
        Dim strNH_CHN As String

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
        intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
        c_possession_nh_code.Value = intNH

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
        strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
        TxtNH.Value = strNH_CHN

        DoCmd.Close acForm, stDocName
    End If

    CmdPickNH.SetFocus
    c_possession_nh_code.Visible = False

Exit_CmdPickNH_Click:
    Exit Sub

Err_CmdPickNH_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickNH_Click

End Sub

Private Sub CmdPickSource_Click()
On Error GoTo Err_CmdPickSource_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strSC As String

    c_source.Visible = True
    c_source.SetFocus
    strSC = c_source.Text

    stDocName = "frmPickTEXTS"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strSC

    If CurrentProject.AllForms("frmPickTEXTS").IsLoaded Then
        Dim intSC As Long
        Dim strSC_CHN As String

        Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.SetFocus
        intSC = Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.Value
        c_source.Value = intSC

        Forms!frmPickTEXTS!frmTEXTS.Form!c_title.SetFocus
        strSC_CHN = Forms!frmPickTEXTS!frmTEXTS.Form!c_title_chn.Value
        TxtTitle_CHN.Value = strSC_CHN

        DoCmd.Close acForm, stDocName
    End If

    CmdPickSource.SetFocus
    c_source.Visible = False

Exit_CmdPickSource_Click:
    Exit Sub

Err_CmdPickSource_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickSource_Click

End Sub

Private Sub Form_AfterDelConfirm(STATUS As Integer)
Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

If STATUS = acDeleteOK Then
    rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.Find "c_flag = " & 1
    rst!c_flag = 0

```



```

    rst.Update
    rst.Close
Else
    rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.Find "c_flag = " & 1
    rst.Delete
    rst.Update
    rst.Close
End If
End Sub

```

```

Private Sub Form_Current()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    If IsNull(c_possession_nh_code.Value) Then
        TxtNH.Value = ""
    Else
        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_nianhao_id = " & c_possession_nh_code.Value

        TxtNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If

    If IsNull(c_addr_id.Value) Then
        TxtAddrCHN.Value = ""
    Else
        rst.Open "ADDRESSES", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_addr_id = " & c_addr_id.Value

        TxtAddrCHN.Value = rst.Fields("c_name_chn")
        rst.Close
    End If

    If IsNull(c_addr_id.Value) Then
        TxtAddrPY.Value = ""
    Else
        rst.Open "ADDRESSES", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_addr_id = " & c_addr_id.Value

        TxtAddrPY.Value = rst.Fields("c_name")
        rst.Close
    End If

    If IsNull(c_source.Value) Then
        TxtTitle_CHN.Value = ""
    Else
        rst.Open "TEXT_CODES", CurrentProject.Connection, adOpenDynamic, _
            adLockOptimistic
        rst.Find "c_textid = " & c_source.Value

        TxtTitle_CHN.Value = rst.Fields("c_title_chn")
        rst.Close
    End If
End Sub

Private Sub CmdAddNew_Click()
    On Error GoTo Err_CmdAddNew_Click

    DoCmd.GoToRecord , , acNewRec

Exit_CmdAddNew_Click:
    Exit Sub

Err_CmdAddNew_Click:
    MsgBox Err.Description
    Resume Exit_CmdAddNew_Click
End Sub

Private Sub CmdDelete_Click()
    On Error GoTo Err_CmdDelete_Click

    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

    Dim blnRecordAdded As Boolean

```

```

If Not IsNull(c_personid) Then
    rst.Open "Del_Log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.AddNew
    rst!c_personid = c_personid
    rst!c_subform = "POSSESSIONS"
    rst!c_flag = 1
    rst!c_possession_act_code = c_possession_act_code
    rst!c_possession_desc_chn = c_possession_desc_chn
    rst!c_possession_desc = c_possession_desc
    rst!c_sequence = c_sequence
    rst!c_year = c_possession_yr
    rst!c_nh_code = c_possession_nh_code
    rst!c_nh_yr = c_possession_nh_yr
    rst!c_range = c_possession_yr_range
    rst!c_quantity = c_quantity
    rst!c_measure_code = c_measure_code
    rst!c_addr_id = c_addr_id
    rst!c_source = c_source
    rst!c_pages = c_pages
    rst!c_notes = c_notes
    rst.Update
    blnRecordAdded = True
    rst.Close
End If

```

```

DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70

```

```
Exit_CmdDelete_Click:
```

```
Exit Sub
```

```
Err_CmdDelete_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdDelete_Click
```

```
End Sub
```

```
Public Sub changeDisplayLanguage()
```

```
Dim tLabelLanguage(3, 18) As String, tLang As Integer
```

```
Dim tRstLabelList As DAO.Recordset, ti As Integer
```

```
Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)
```

```
tRstLabelList.Index = "label"
```

```
gLabelsOK = False
```

```
With tRstLabelList
```

```
.MoveFirst
```

```
ti = 1
```

```
Do While ti < 18 And Not .EOF
```

```
    If !c_form = "SF_POS" Then
```

```
        gLabelsOK = True
```

```
        If ti <> !c_label_id Then
```

```
            MsgBox "Uh oh: mismatched label table"
```

```
            gLabelsOK = False
```

```
            Exit Do
```

```
        End If
```

```
        tLabelLanguage(1, ti) = !c_english
```

```
        tLabelLanguage(2, ti) = !c_fanti
```

```
        tLabelLanguage(3, ti) = !c_jianti
```

```
        ti = ti + 1
```

```
    End If
```

```
.MoveNext
```

```
Loop
```

```
End With
```

```
' tRstLabelList.Close
```

```
Set tRstLabelList = Nothing
```

```
If gLabelsOK Then
```

```
    If gDisplayLanguage = "E" Then
```

```
        tLang = 1
```

```
    ElseIf gDisplayLanguage = "T" Then
```

```
        tLang = 2
```

```
    Else
```

```
        tLang = 3
```

```
    End If
```

```
,
```

```
' now comes the basic routine
'
Me.LblSequence.Caption = tLabelLanguage(tLang, 1)
Me.LblRecordID.Caption = tLabelLanguage(tLang, 2)
Me.LblAct.Caption = tLabelLanguage(tLang, 3)
Me.LblDescChn.Caption = tLabelLanguage(tLang, 4)
Me.LblDesc.Caption = tLabelLanguage(tLang, 5)
Me.LblQuantity.Caption = tLabelLanguage(tLang, 6)
Me.LblUnit.Caption = tLabelLanguage(tLang, 7)
Me.LblYear.Caption = tLabelLanguage(tLang, 8)
Me.LblNHYear.Caption = tLabelLanguage(tLang, 9)
Me.LblRange.Caption = tLabelLanguage(tLang, 10)
Me.LblPages.Caption = tLabelLanguage(tLang, 11)
Me.LblNotes.Caption = tLabelLanguage(tLang, 12)
Me.CmdPickNH.Caption = tLabelLanguage(tLang, 13)
Me.CmdPickAddr.Caption = tLabelLanguage(tLang, 14)
Me.CmdPickSource.Caption = tLabelLanguage(tLang, 15)
Me.CmdDelete.Caption = tLabelLanguage(tLang, 16)
Me.CmdAddNew.Caption = tLabelLanguage(tLang, 17)
End If

End Sub

Public Sub noEdits()

    Me.AllowAdditions = False
    Me.AllowDeletions = False
    Me.AllowEdits = False

End Sub
```

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 18) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 18 And Not .EOF

If !c_form = "SF_POS" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

Me.LblSequence.Caption = tLabelLanguage(tLang, 1)

Me.LblRecordID.Caption = tLabelLanguage(tLang, 2)

Me.LblAct.Caption = tLabelLanguage(tLang, 3)

Me.LblDescChn.Caption = tLabelLanguage(tLang, 4)

Me.LblDesc.Caption = tLabelLanguage(tLang, 5)

Me.LblQuantity.Caption = tLabelLanguage(tLang, 6)

Me.LblUnit.Caption = tLabelLanguage(tLang, 7)

Me.LblYear.Caption = tLabelLanguage(tLang, 8)

Me.LblNHYear.Caption = tLabelLanguage(tLang, 9)

Me.LblRange.Caption = tLabelLanguage(tLang, 10)

Me.LblPages.Caption = tLabelLanguage(tLang, 11)

Me.LblNotes.Caption = tLabelLanguage(tLang, 12)

Me.LblNH.Caption = tLabelLanguage(tLang, 13)

Me.LblAddr.Caption = tLabelLanguage(tLang, 14)

Me.LblSource.Caption = tLabelLanguage(tLang, 15)

' Me.CmdDelete.Caption = tLabelLanguage(tLang, 16)

' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 17)

End If

End Sub

Public Sub noEdits()

Me.AllowAdditions = False

Me.AllowDeletions = False

Me.AllowEdits = False

End Sub

frm_POST_ADDR Subform - 1

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Private Sub CmdPickAddr_Click()

On Error GoTo Err_CmdPickAddr_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strADDR As String

c_addr_id.Visible = True
c_addr_id.SetFocus
strADDR = c_addr_id.Text

stDocName = "frmPickADDRESSES"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strADDR

If CurrentProject.AllForms("frmPickADDRESSES").IsLoaded Then

Dim intAddr As Long
Dim strADDR_CHN As String
Dim strADDR_PY As String

Forms!frmPickADDRESSES!frmADDRESSES.Form!c_addr_id.SetFocus
intAddr = Forms!frmPickADDRESSES!frmADDRESSES.Form!c_addr_id.Value
c_addr_id.Value = intAddr

Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name_chn.SetFocus
strADDR_CHN = Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name_chn.Value
TxtAddrCHN.Value = strADDR_CHN

Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name.SetFocus
strADDR_PY = Forms!frmPickADDRESSES!frmADDRESSES.Form!c_name.Value
TxtAddrPY.Value = strADDR_PY

DoCmd.Close acForm, stDocName
End If

CmdPickAddr.SetFocus
c_addr_id.Visible = False

Exit_CmdPickAddr_Click:

Exit Sub

Err_CmdPickAddr_Click:

MsgBox Err.Description
Resume Exit_CmdPickAddr_Click

End Sub

Private Sub Form_AfterDelConfirm(STATUS As Integer)

Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

If STATUS = acDeleteOK Then

rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.Find "c_flag = " & 1
rst!c_flag = 0
rst.Update
rst.Close

Else

rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.Find "c_flag = " & 1
rst.Delete
rst.Update
rst.Close

End If

End Sub

Private Sub Form_Current()

Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

If IsNull(c_addr_id.Value) Then
TxtAddrCHN.Value = ""
TxtAddrPY.Value = ""

Else

rst.Open "ADDR_CODES", CurrentProject.Connection, adOpenDynamic, _
adLockOptimistic

```

        rst.Find "c_addr_id = " & c_addr_id.Value

        TxtAddrCHN.Value = rst.Fields("c_name_chn")
        TxtAddrPY.Value = rst.Fields("c_name")
        rst.Close
    End If

End Sub

Private Sub CmdDelete_Click()
On Error GoTo Err_CmdDelete_Click

Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

Dim blnRecordAdded As Boolean

If Not IsNull(c_addr_id) Then
    'This If condition is to make sure that the record one is attempting to delete is not an empty record. Thus, the
    'field used in the condition must be a required field for the record.
    rst.Open "Del_Log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.AddNew
    rst!c_personid = c_personid
    rst!c_subform = "POST_ADDR"
    rst!c_flag = 1
    rst!c_addr_id = c_addr_id
    rst!c_posting_id = c_posting_id
    rst!c_firstyear = c_firstyear
    rst.Update
    blnRecordAdded = True
    rst.Close
End If

    DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
    DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70

Exit_CmdDelete_Click:
    Exit Sub

Err_CmdDelete_Click:
    MsgBox Err.Description
    Resume Exit_CmdDelete_Click

End Sub

Private Sub CmdAddNew_Click()
On Error GoTo Err_CmdAddNew_Click

    DoCmd.GoToRecord , , acNewRec

Exit_CmdAddNew_Click:
    Exit Sub

Err_CmdAddNew_Click:
    MsgBox Err.Description
    Resume Exit_CmdAddNew_Click

End Sub

Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 4) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 4 And Not .EOF
            If !c_form = "SF_PADDR" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
            End If
            ti = ti + 1
        Loop
    End With

```

```
        tLabelLanguage(2, ti) = !c_fanti
        tLabelLanguage(3, ti) = !c_jianti
        ti = ti + 1
    End If
    .MoveNext
Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If gLabelsOK Then
    If gDisplayLanguage = "E" Then
        tLang = 1
    ElseIf gDisplayLanguage = "T" Then
        tLang = 2
    Else
        tLang = 3
    End If
    '
    ' now comes the basic routine
    '
    Me.CmdPickAddr.Caption = tLabelLanguage(tLang, 1)
    Me.CmdAddNew.Caption = tLabelLanguage(tLang, 2)
    Me.CmdDelete.Caption = tLabelLanguage(tLang, 3)
End If

End Sub

Public Sub noEdits()

    Me.AllowAdditions = False
    Me.AllowDeletions = False
    Me.AllowEdits = False

End Sub
```

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Private Sub c_office_id_AfterUpdate()

Dim intLastPostID As Long

Dim intPostID As Long

If IsNull(c_posting_id.Value) And Not IsNull(c_office_id) Then

intLastPostID = DMax("c_posting_id", "POST_DATA")

TxtLastPostID.Value = intLastPostID

TxtLastPostID.Visible = True

TxtLastPostID.SetFocus

intPostID = TxtLastPostID.Value

c_posting_id.Value = intPostID + 1

c_posting_id.SetFocus

TxtLastPostID.Visible = False

End If

End Sub

Private Sub CmbApptType_AfterUpdate()

Dim rst As ADODB.Recordset

Set rst = New ADODB.Recordset

If IsNull(CmbApptType.Value) Then

TxtApptType.Value = ""

Else

rst.Open "APPOINTMENT_CODES", CurrentProject.Connection, adOpenDynamic, _

adLockOptimistic

rst.Find "c_appt_code = " & CmbApptType.Value

TxtApptType.Value = rst.Fields("c_appt_desc_chn")

rst.Close

End If

End Sub

Private Sub CmdPickOffice_Click()

On Error GoTo Err_CmdPickOffice_Click

Dim stDocName As String

Dim stLinkCriteria As String

Dim strOffice As String

c_office_id.Visible = True

c_office_id.SetFocus

strOffice = c_office_id.Text

stDocName = "frmPickOfficeTree"

DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strOffice

If CurrentProject.AllForms("frmPickOfficeTree").IsLoaded Then

Dim intOF As Long

Dim strOF_CHN As String

Forms!frmPickOfficeTree!frmOFFICE_CODES.Form!c_office_id.SetFocus

intOffice = Forms!frmPickOfficeTree!frmOFFICE_CODES.Form!c_office_id.Value

c_office_id.Value = intOffice

Forms!frmPickOfficeTree!frmOFFICE_CODES.Form!c_office_chn.SetFocus

strOFFICE_CHN = Forms!frmPickOfficeTree!frmOFFICE_CODES.Form!c_office_chn.Value

c_office_chn.Value = strOFFICE_CHN

DoCmd.Close acForm, stDocName

End If

CmdPickOffice.SetFocus

c_office_id.Visible = False

Dim intLastPostID As Long

Dim intPostID As Long

If IsNull(c_posting_id.Value) And Not IsNull(c_office_id) Then

intLastPostID = DMax("c_posting_id", "POST_DATA")

TxtLastPostID.Value = intLastPostID

TxtLastPostID.Visible = True

TxtLastPostID.SetFocus

intPostID = TxtLastPostID.Value

c_posting_id.Value = intPostID + 1


```

        c_posting_id.SetFocus
        TxtLastPostID.Visible = False
    End If

Exit_CmdPickOffice_Click:
    Exit Sub

Err_CmdPickOffice_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickOffice_Click

End Sub

Private Sub CmdPickFY_NH_Click()
On Error GoTo Err_CmdPickFY_NH_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strNH As String

    c_fy_nh_code.Visible = True
    c_fy_nh_code.SetFocus
    strNH = c_fy_nh_code.Text

    stDocName = "frmPickNIAN_HAO"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

    If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
        Dim intNH As Integer
        Dim strNH_CHN As String

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
        intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
        c_fy_nh_code.Value = intNH

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
        strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
        TxtFYNH.Value = strNH_CHN

        DoCmd.Close acForm, stDocName
    End If

    CmdPickFY_NH.SetFocus
    c_fy_nh_code.Visible = False

Exit_CmdPickFY_NH_Click:
    Exit Sub

Err_CmdPickFY_NH_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickFY_NH_Click

End Sub

Private Sub CmdPickLY_NH_Click()
On Error GoTo Err_CmdPickLY_NH_Click

    Dim stDocName As String
    Dim stLinkCriteria As String
    Dim strNH As String

    c_ly_nh_code.Visible = True
    c_ly_nh_code.SetFocus
    strNH = c_ly_nh_code.Text

    stDocName = "frmPickNIAN_HAO"
    DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

    If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
        Dim intNH As Integer
        Dim strNH_CHN As String

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
        intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
        c_ly_nh_code.Value = intNH

        Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
        strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
        TxtLYNH.Value = strNH_CHN

```

```
DoCmd.Close acForm, stDocName
End If
```

```
CmdPickLY_NH.SetFocus
c_ly_nh_code.Visible = False
```

```
Exit_CmdPickLY_NH_Click:
Exit Sub
```

```
Err_CmdPickLY_NH_Click:
MsgBox Err.Description
Resume Exit_CmdPickLY_NH_Click
```

```
End Sub
```

```
Private Sub CmdPickSource_Click()
On Error GoTo Err_CmdPickSource_Click
```

```
Dim stDocName As String
Dim stLinkCriteria As String
Dim strSC As String
```

```
c_source.Visible = True
c_source.SetFocus
strSC = c_source.Text
```

```
stDocName = "frmPickTEXTS"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strSC
```

```
If CurrentProject.AllForms("frmPickTEXTS").IsLoaded Then
Dim intSC As Long
Dim strSC_CHN As String

Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.SetFocus
intSC = Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.Value
c_source.Value = intSC

Forms!frmPickTEXTS!frmTEXTS.Form!c_title.SetFocus
strSC_CHN = Forms!frmPickTEXTS!frmTEXTS.Form!c_title_chn.Value
TxtTitle_CHN.Value = strSC_CHN

DoCmd.Close acForm, stDocName
End If
```

```
CmdPickSource.SetFocus
c_source.Visible = False
```

```
Exit_CmdPickSource_Click:
Exit Sub
```

```
Err_CmdPickSource_Click:
MsgBox Err.Description
Resume Exit_CmdPickSource_Click
```

```
End Sub
```

```
Private Sub Form_AfterDelConfirm(STATUS As Integer)
```

```
Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset
```

```
Dim intPost As Long
Dim intPerson As Long
Dim intAddr As Long
Dim blnRecordAdded As Boolean
```

```
'Dim intLoopEnd As Integer
'intLoopEnd = 0
```

```
If STATUS = acDeleteOK Then
rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.Find "c_flag = " & 1
intPost = rst.Fields("c_posting_id")
rst!c_flag = 0
rst.Update
rst.Close
```

```
'Delete corresponding records in POST_ADDR table, using c_posting_id; and create Del_log entries for these co
```

rresponding records

Do

```

rst.Open "POST_ADDR", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.Find "c_posting_id = " & intPost
If rst.EOF Then
    rst.Close
    Exit Do
End If
intPerson = rst.Fields("c_personid")
intAddr = rst.Fields("c_addr_id")
rst.Delete
rst.Update
rst.Close

```

```

rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.AddNew
rst!c_personid = intPerson
rst!c_subform = "POST_ADDR"
rst!c_posting_id = intPost
rst!c_addr_id = intAddr
rst.Update
blnRecordAdded = True
rst.Close

```

Loop

Else

```

rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
rst.Find "c_flag = " & 1
rst.Delete
rst.Update
rst.Close

```

End If

End Sub

```

Private Sub CmdDelete_Click()
On Error GoTo Err_CmdDelete_Click

```

```

Dim rst As ADODB.Recordset
Set rst = New ADODB.Recordset

```

```

Dim blnRecordAdded As Boolean

```

```

If Not IsNull(c_personid) Then
    rst.Open "Del_Log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.AddNew
    rst!c_personid = c_personid
    rst!c_subform = "POSTINGS"
    rst!c_flag = 1
    rst!c_posting_id = c_posting_id
    rst!c_office_id = c_office_id
    rst!c_sequence = c_sequence
    rst!c_firstyear = c_firstyear
    rst!c_lastyear = c_lastyear
    rst!c_fy_nh_code = c_fy_nh_code
    rst!c_ly_nh_code = c_ly_nh_code
    rst!c_fy_nh_yr = c_fy_nh_year
    rst!c_ly_nh_yr = c_ly_nh_year
    rst!c_fy_range = c_fy_range
    rst!c_ly_range = c_ly_range
    rst!c_source = c_source
    rst!c_pages = c_pages
    rst!c_notes = c_notes
    rst.Update
    blnRecordAdded = True
    rst.Close

```

End If

```

DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70

```

Exit_CmdDelete_Click:

Exit Sub

Err_CmdDelete_Click:

```

MsgBox Err.Description
Resume Exit_CmdDelete_Click

```

End Sub

Form_POST_DATA Subform - 5

```
Private Sub CmdAddNew_Click()  
On Error GoTo Err_CmdAddNew_Click
```

```
DoCmd.GoToRecord , , acNewRec
```

```
Exit_CmdAddNew_Click:  
Exit Sub
```

```
Err_CmdAddNew_Click:  
MsgBox Err.Description  
Resume Exit_CmdAddNew_Click
```

```
End Sub
```

```
Public Sub changeDisplayLanguage()  
Dim tLabelLanguage(3, 3) As String, tLang As Integer  
  
Dim tRstLabelList As DAO.Recordset, ti As Integer  
  
Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)  
  
tRstLabelList.Index = "label"  
  
gLabelsOK = False  
With tRstLabelList  
    .MoveFirst  
    ti = 1  
  
    Do While ti < 3 And Not .EOF  
        If !c_form = "SF_POST" Then  
            gLabelsOK = True  
            If ti <> !c_label_id Then  
                MsgBox "Uh oh: mismatched label table"  
                gLabelsOK = False  
                Exit Do  
            End If  
            tLabelLanguage(1, ti) = !c_english  
            tLabelLanguage(2, ti) = !c_fanti  
            tLabelLanguage(3, ti) = !c_jianti  
            ti = ti + 1  
        End If  
        .MoveNext  
    Loop  
End With  
' tRstLabelList.Close  
Set tRstLabelList = Nothing  
  
If gLabelsOK Then  
    If gDisplayLanguage = "E" Then  
        tLang = 1  
    ElseIf gDisplayLanguage = "T" Then  
        tLang = 2  
    Else  
        tLang = 3  
    End If  
    '  
' now comes the basic routine  
'  
    Me.CmdAddNew.Caption = tLabelLanguage(tLang, 1)  
    Me.CmdDelete.Caption = tLabelLanguage(tLang, 2)  
  
    Me.POST_ADDR_Subform.Form.gDisplayLanguage = gDisplayLanguage  
    Me.POST_ADDR_Subform.Form.changeDisplayLanguage  
  
    Me.POSTING_OFFICE_DATA_Subform.Form.gDisplayLanguage = gDisplayLanguage  
    Me.POSTING_OFFICE_DATA_Subform.Form.changeDisplayLanguage  
End If  
  
End Sub
```

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean, gTest As Integer

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 23) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 24

'Do While tI < 23 And Not .EOF

' If !c_form = "SF_POFF" Then

' gLabelsOK = True

' If tI <> !c_label_id Then

' MsgBox "Uh oh: mismatched label table"

' gLabelsOK = False

' Exit Do

' End If

' tLabelLanguage(1, tI) = !c_english

' tLabelLanguage(2, tI) = !c_fanti

' tLabelLanguage(3, tI) = !c_jianti

' tI = tI + 1

' End If

' .MoveNext

'Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

'Me.LblSequence.Caption = tLabelLanguage(tLang, 1)

'Me.LblOfficeCat.Caption = tLabelLanguage(tLang, 2)

'Me.LblPostCat.Caption = tLabelLanguage(tLang, 3)

'Me.LblFirstyear.Caption = tLabelLanguage(tLang, 4)

'Me.LblFYRange.Caption = tLabelLanguage(tLang, 5)

'Me.LblAssume.Caption = tLabelLanguage(tLang, 6)

'Me.LblFYNHYear.Caption = tLabelLanguage(tLang, 7)

'Me.LblFYIintercalary.Caption = tLabelLanguage(tLang, 8)

'Me.LblFYGZ.Caption = tLabelLanguage(tLang, 9)

'Me.LblLastyear.Caption = tLabelLanguage(tLang, 10)

'Me.LblLYRange.Caption = tLabelLanguage(tLang, 11)

'Me.LblLYNHYear.Caption = tLabelLanguage(tLang, 12)

'Me.LblLYintercalary.Caption = tLabelLanguage(tLang, 13)

'Me.LblLYGZ.Caption = tLabelLanguage(tLang, 14)

'Me.LblPages.Caption = tLabelLanguage(tLang, 15)

'Me.LblNotes.Caption = tLabelLanguage(tLang, 16)

'Me.CmdPickOffice.Caption = tLabelLanguage(tLang, 17)

'Me.CmdPickFY_NH.Caption = tLabelLanguage(tLang, 18)

'Me.CmdPickLY_NH.Caption = tLabelLanguage(tLang, 19)

'Me.CmdPickSource.Caption = tLabelLanguage(tLang, 20)

'Me.CmdAddNew.Caption = tLabelLanguage(tLang, 21)

'Me.CmdDelete.Caption = tLabelLanguage(tLang, 22)

End If

End Sub

Private Sub CmdPlace_Click()

On Error GoTo Err_CmdPlace_Click

If Me.Dirty Then Me.Dirty = False

DoCmd.Close

```
Exit_CmdPlace_Click:
    Exit Sub
```

```
Err_CmdPlace_Click:
    MsgBox Err.Description
    Resume Exit_CmdPlace_Click

End Sub
```

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 23) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

' set the language

Dim tmli As MsoLanguageID

' get the labels

tmli = Application.LanguageSettings.LanguageID(msoLanguageIDUI)

If tmli = msoLanguageIDSimplifiedChinese Then

gDisplayLanguage = "S"

ElseIf tmli = msoLanguageIDTraditionalChinese Then

gDisplayLanguage = "T"

ElseIf tmli = msoLanguageIDEnglishUS Then

gDisplayLanguage = "E"

Else

gDisplayLanguage = "E"

End If

'MsgBox "Beginning Posted to Office Data labels"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 23 And Not .EOF

If !c_form = "SF_POFF" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

'MsgBox "Beginning to change Posted to Office Data labels"

Me.LblSequence.Caption = tLabelLanguage(tLang, 1)

Me.LblOfficeCat.Caption = tLabelLanguage(tLang, 2)

Me.LblPostCat.Caption = tLabelLanguage(tLang, 3)

Me.LblFirstyear.Caption = tLabelLanguage(tLang, 4)

Me.LblFYRange.Caption = tLabelLanguage(tLang, 5)

Me.LblAssume.Caption = tLabelLanguage(tLang, 6)

Me.LblFYNHYear.Caption = tLabelLanguage(tLang, 7)

Me.LblFYIntercalary.Caption = tLabelLanguage(tLang, 8)

Me.LblFYGZ.Caption = tLabelLanguage(tLang, 9)

Me.LblLastyear.Caption = tLabelLanguage(tLang, 10)

Me.LblLYRange.Caption = tLabelLanguage(tLang, 11)

Me.LblLYNHYear.Caption = tLabelLanguage(tLang, 12)

Me.LblLYIntercalary.Caption = tLabelLanguage(tLang, 13)

Me.LblLYGZ.Caption = tLabelLanguage(tLang, 14)

Me.LblPages.Caption = tLabelLanguage(tLang, 15)

```
Me.LblNotes.Caption = tLabelLanguage(tLang, 16)
Me.LblOfficeName.Caption = tLabelLanguage(tLang, 17)
Me.LblFYNH.Caption = tLabelLanguage(tLang, 18)
Me.LblLYNH.Caption = tLabelLanguage(tLang, 19)
Me.LblSource.Caption = tLabelLanguage(tLang, 20)
' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 21)
' Me.CmdDelete.Caption = tLabelLanguage(tLang, 22)

' Me.POSTED_TO_ADDR_DATA_2_Subform.Form.gTest = 1
'
' Me.POSTED_TO_ADDR_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage
' Me.POSTED_TO_ADDR_DATA_2_Subform.Form.changeDisplayLanguage
End If
```

End Sub

Public Sub noEdits()

```
Me.AllowAdditions = False
Me.AllowDeletions = False
Me.AllowEdits = False
```

End Sub

Option Compare Database

```
Private Sub CmdPlace_Click()  
On Error GoTo Err_CmdPlace_Click
```

```
    If Me.Dirty Then Me.Dirty = False  
    DoCmd.Close
```

```
Exit_CmdPlace_Click:  
Exit Sub
```

```
Err_CmdPlace_Click:  
MsgBox Err.Description  
Resume Exit_CmdPlace_Click
```

```
End Sub
```

Form_POSTING_DATA_2 Subform -1

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 3) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 3 And Not .EOF

If !c_form = "SF_POST" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 1)

' Me.CmdDelete.Caption = tLabelLanguage(tLang, 2)

' Me.POST_ADDR_Subform.Form.gDisplayLanguage = gDisplayLanguage

' Me.POST_ADDR_Subform.Form.changeDisplayLanguage

'MsgBox "Beginning Posted to Office_data_2 subform 1"

Me.POSTED_TO_OFFICE_DATA_2_Subform.Form.gDisplayLanguage = gDisplayLanguage

'MsgBox "Beginning Posted to Office_data_2 subform 2"

Me.POSTED_TO_OFFICE_DATA_2_Subform.Form.changeDisplayLanguage

End If

End Sub

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Private Sub Form_Current()

Dim rst As ADODB.Recordset

Set rst = New ADODB.Recordset

If IsNull(c_fy_nh_code.Value) Then

TxtFYNH.Value = ""

Else

rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
adLockOptimistic

rst.Find "c_nianhao_id = " & c_fy_nh_code.Value

TxtFYNH.Value = rst.Fields("c_nianhao_chn")

rst.Close

End If

If IsNull(c_ly_nh_code.Value) Then

TxtLYNH.Value = ""

Else

rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
adLockOptimistic

rst.Find "c_nianhao_id = " & c_ly_nh_code.Value

TxtLYNH.Value = rst.Fields("c_nianhao_chn")

rst.Close

End If

If IsNull(c_source.Value) Then

TxtTitle_CHN.Value = ""

Else

rst.Open "TEXT_CODES", CurrentProject.Connection, adOpenDynamic, _
adLockOptimistic

rst.Find "c_textid = " & c_source.Value

TxtTitle_CHN.Value = rst.Fields("c_title_chn")

rst.Close

End If

If IsNull(CmbApptType.Value) Then

TxtApptType.Value = ""

Else

rst.Open "APPOINTMENT_CODES", CurrentProject.Connection, adOpenDynamic, _
adLockOptimistic

rst.Find "c_appt_code = " & CmbApptType.Value

TxtApptType.Value = rst.Fields("c_appt_desc_chn")

rst.Close

End If

If IsNull(c_office_category_id.Value) Then

Me.TxtCatDesc.Value = ""

Me.TxtCatDescChn.Value = ""

Else

rst.Open "OFFICE_CATEGORIES", CurrentProject.Connection, adOpenDynamic, _
adLockOptimistic

rst.Find "c_office_category_id = " & c_office_category_id.Value

TxtCatDescChn.Value = rst.Fields("c_category_desc_chn")

TxtCatDesc.Value = rst.Fields("c_category_desc")

rst.Close

End If

End Sub

Private Sub CmdDelete_Click()

On Error GoTo Err_CmdDelete_Click

DoCmd.RunCommand acCmdSelectRecord

DoCmd.RunCommand acCmdDeleteRecord

Exit_CmdDelete_Click:

Exit Sub

Err_CmdDelete_Click:

MsgBox Err.Description

```

Resume Exit_CmdDelete_Click

End Sub

Private Sub CmdAddNew_Click()
On Error GoTo Err_CmdAddNew_Click

DoCmd.GoToRecord , , acNewRec

Exit_CmdAddNew_Click:
Exit Sub

Err_CmdAddNew_Click:
MsgBox Err.Description
Resume Exit_CmdAddNew_Click

End Sub

Public Sub changeDisplayLanguage()
Dim tLabelLanguage(3, 23) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False
With tRstLabelList
.MoveFirst
ti = 1

Do While ti < 23 And Not .EOF
If !c_form = "SF_POFF" Then
gLabelsOK = True
If ti <> !c_label_id Then
MsgBox "Uh oh: mismatched label table"
gLabelsOK = False
Exit Do
End If
tLabelLanguage(1, ti) = !c_english
tLabelLanguage(2, ti) = !c_fanti
tLabelLanguage(3, ti) = !c_jianti
ti = ti + 1
End If
.MoveNext
Loop
End With
' tRstLabelList.Close
Set tRstLabelList = Nothing

If gLabelsOK Then
If gDisplayLanguage = "E" Then
tLang = 1
ElseIf gDisplayLanguage = "T" Then
tLang = 2
Else
tLang = 3
End If
'
' now comes the basic routine
'
Me.LblSequence.Caption = tLabelLanguage(tLang, 1)
Me.LblOfficeCat.Caption = tLabelLanguage(tLang, 2)
Me.LblPostCat.Caption = tLabelLanguage(tLang, 3)
Me.LblFirstyear.Caption = tLabelLanguage(tLang, 4)
Me.LblFYRange.Caption = tLabelLanguage(tLang, 5)
Me.LblAssume.Caption = tLabelLanguage(tLang, 6)
Me.LblFYNHYear.Caption = tLabelLanguage(tLang, 7)
Me.LblFYIntercalary.Caption = tLabelLanguage(tLang, 8)
Me.LblFYGZ.Caption = tLabelLanguage(tLang, 9)
Me.LblLastyear.Caption = tLabelLanguage(tLang, 10)
Me.LblLYRange.Caption = tLabelLanguage(tLang, 11)
Me.LblLYNHyear.Caption = tLabelLanguage(tLang, 12)
Me.LblLYIntercalary.Caption = tLabelLanguage(tLang, 13)
Me.LblLYGZ.Caption = tLabelLanguage(tLang, 14)
Me.LblPages.Caption = tLabelLanguage(tLang, 15)
Me.LblNotes.Caption = tLabelLanguage(tLang, 16)
Me.CmdPickOffice.Caption = tLabelLanguage(tLang, 17)
Me.CmdPickFY_NH.Caption = tLabelLanguage(tLang, 18)

```

```
Me.CmdPickLY_NH.Caption = tLabelLanguage(tLang, 19)
Me.CmdPickSource.Caption = tLabelLanguage(tLang, 20)
Me.CmdAddNew.Caption = tLabelLanguage(tLang, 21)
Me.CmdDelete.Caption = tLabelLanguage(tLang, 22)
End If
```

End Sub

Public Sub noEdits()

```
Me.AllowAdditions = False
Me.AllowDeletions = False
Me.AllowEdits = False
```

End Sub

Option Compare Database
Public gDisplayLanguage As String, gLabelsOK As Boolean

Private Sub CmdPickFYNH_Click()
On Error GoTo Err_CmdPickFYNH_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strNH As String

c_fy_nh_code.Visible = True
c_fy_nh_code.SetFocus
strNH = c_fy_nh_code.Text

stDocName = "frmPickNIAN_HAO"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
Dim intNH As Integer
Dim strNH_CHN As String

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
c_fy_nh_code.Value = intNH

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
TxtFYNH.Value = strNH_CHN

DoCmd.Close acForm, stDocName
End If

CmdPickLYNH.SetFocus
c_fy_nh_code.Visible = False

Exit_CmdPickFYNH_Click:
Exit Sub

Err_CmdPickFYNH_Click:
MsgBox Err.Description
Resume Exit_CmdPickFYNH_Click

End Sub

Private Sub CmdPickLYNH_Click()
On Error GoTo Err_CmdPickLYNH_Click

Dim stDocName As String
Dim stLinkCriteria As String
Dim strNH As String

c_ly_nh_code.Visible = True
c_ly_nh_code.SetFocus
strNH = c_ly_nh_code.Text

stDocName = "frmPickNIAN_HAO"
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strNH

If CurrentProject.AllForms("frmPickNIAN_HAO").IsLoaded Then
Dim intNH As Integer
Dim strNH_CHN As String

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.SetFocus
intNH = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_id.Value
c_ly_nh_code.Value = intNH

Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.SetFocus
strNH_CHN = Forms!frmPickNIAN_HAO!frmNIAN_HAO.Form!c_nianhao_chn.Value
TxtLYNH.Value = strNH_CHN

DoCmd.Close acForm, stDocName
End If

CmdPickLYNH.SetFocus
c_ly_nh_code.Visible = False

```
Exit_CmdPickLYNH_Click:
```

```
Exit Sub
```

```
Err_CmdPickLYNH_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdPickLYNH_Click
```

```
End Sub
```

```
Private Sub CmdPickSocDtn_Click()
```

```
On Error GoTo Err_CmdPickSocDtn_Click
```

```
Dim stDocName As String
```

```
Dim stLinkCriteria As String
```

```
Dim strSD As String
```

```
c_status_code.Visible = True
```

```
c_status_code.SetFocus
```

```
strSD = c_status_code.Text
```

```
stDocName = "frmPickSTATUS_CODES"
```

```
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strSD
```

```
If CurrentProject.AllForms("frmPickSTATUS_CODES").IsLoaded Then
```

```
Dim intSD As Long
```

```
Dim strSD_CHN As String
```

```
Forms!frmPickSTATUS_CODES!frmSTATUS_CODES.Form!c_status_code.SetFocus
```

```
intSD = Forms!frmPickSTATUS_CODES!frmSTATUS_CODES.Form!c_status_code.Value
```

```
c_status_code.Value = intSD
```

```
Forms!frmPickSTATUS_CODES!frmSTATUS_CODES.Form!c_status_desc_chn.SetFocus
```

```
strSD_CHN = Forms!frmPickSTATUS_CODES!frmSTATUS_CODES.Form!c_status_desc_chn.Value
```

```
c_status_desc_chn.Value = strSD_CHN
```

```
DoCmd.Close acForm, stDocName
```

```
End If
```

```
CmdPickSocDtn.SetFocus
```

```
c_status_code.Visible = False
```

```
Exit_CmdPickSocDtn_Click:
```

```
Exit Sub
```

```
Err_CmdPickSocDtn_Click:
```

```
MsgBox Err.Description
```

```
Resume Exit_CmdPickSocDtn_Click
```

```
End Sub
```

```
Private Sub CmdPickSource_Click()
```

```
On Error GoTo Err_CmdPickSource_Click
```

```
Dim stDocName As String
```

```
Dim stLinkCriteria As String
```

```
Dim strSC As String
```

```
c_source.Visible = True
```

```
c_source.SetFocus
```

```
strSC = c_source.Text
```

```
stDocName = "frmPickTEXTS"
```

```
DoCmd.OpenForm stDocName, , , stLinkCriteria, , acDialog, strSC
```

```
If CurrentProject.AllForms("frmPickTEXTS").IsLoaded Then
```

```
Dim intSC As Long
```

```
Dim strSC_CHN As String, strSC_Eng
```

```
Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.SetFocus
```

```
intSC = Forms!frmPickTEXTS!frmTEXTS.Form!c_textid.Value
```

```
c_source.Value = intSC
```

```
Forms!frmPickTEXTS!frmTEXTS.Form!c_title.SetFocus
```

```
strSC_CHN = Forms!frmPickTEXTS!frmTEXTS.Form!c_title_chn.Value
```

```
strSC_Eng = Forms!frmPickTEXTS!frmTEXTS.Form!c_title.Value
```

```
TxtTitle_CHN.Value = strSC_CHN
```

```

        DoCmd.Close acForm, stDocName
    End If

```

```

CmdPickSource.SetFocus
c_source.Visible = False

```

```

Exit_CmdPickSource_Click:
    Exit Sub

```

```

Err_CmdPickSource_Click:
    MsgBox Err.Description
    Resume Exit_CmdPickSource_Click

```

```

End Sub

```

```

Private Sub Form_AfterDelConfirm(STATUS As Integer)
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

```

```

    If STATUS = acDeleteOK Then
        rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
        rst.Find "c_flag = " & 1
        rst!c_flag = 0
        rst.Update
        rst.Close
    Else

```

```

        rst.Open "Del_log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
        rst.Find "c_flag = " & 1
        rst.Delete
        rst.Update
        rst.Close
    End If

```

```

End Sub

```

```

End Sub

```

```

Private Sub Form_Current()
    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

```

```

    If IsNull(c_fy_nh_code.Value) Then
        TxtFYNH.Value = ""
    Else

```

```

        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
        adLockOptimistic
        rst.Find "c_nianhao_id = " & c_fy_nh_code.Value

        TxtFYNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If

```

```

End If

```

```

    If IsNull(c_ly_nh_code.Value) Then
        TxtLYNH.Value = ""
    Else

```

```

        rst.Open "nian_hao", CurrentProject.Connection, adOpenDynamic, _
        adLockOptimistic
        rst.Find "c_nianhao_id = " & c_ly_nh_code.Value

        TxtLYNH.Value = rst.Fields("c_nianhao_chn")
        rst.Close
    End If

```

```

End If

```

```

    If IsNull(c_source.Value) Then
        TxtTitle_CHN.Value = ""
    Else

```

```

        rst.Open "TEXT_CODES", CurrentProject.Connection, adOpenDynamic, _
        adLockOptimistic
        rst.Find "c_textid = " & c_source.Value

        TxtTitle_CHN.Value = rst.Fields("c_title_chn")
        rst.Close
    End If

```

```

End If

```

```

End Sub

```

```

Private Sub CmdDelete_Click()
    On Error GoTo Err_CmdDelete_Click

```

```

    Dim rst As ADODB.Recordset
    Set rst = New ADODB.Recordset

```



```
Dim blnRecordAdded As Boolean
```

```
If Not IsNull(c_personid) Then
    rst.Open "Del_Log", CurrentProject.Connection, adOpenDynamic, adLockOptimistic
    rst.AddNew
    rst!c_personid = c_personid
    rst!c_subform = "STATUS"
    rst!c_flag = 1
    rst!c_status_code = c_status_code
    rst!c_sequence = c_sequence
    rst!c_firstyear = c_firstyear
    rst!c_lastyear = c_lastyear
    rst!c_fy_nh_code = c_fy_nh_code
    rst!c_ly_nh_code = c_ly_nh_code
    rst!c_fy_nh_yr = c_fy_nh_year
    rst!c_ly_nh_yr = c_ly_nh_year
    rst!c_fy_range = c_ly_range
    rst!c_source = c_source
    rst!c_pages = c_pages
    rst!c_notes = c_notes
    rst.Update
    blnRecordAdded = True
    rst.Close
End If
```

```
DoCmd.DoMenuItem acFormBar, acEditMenu, 8, , acMenuVer70
DoCmd.DoMenuItem acFormBar, acEditMenu, 6, , acMenuVer70
```

```
Exit_CmdDelete_Click:
Exit Sub
```

```
Err_CmdDelete_Click:
MsgBox Err.Description
Resume Exit_CmdDelete_Click
```

```
End Sub
Private Sub CmdAddNew_Click()
On Error GoTo Err_CmdAddNew_Click
```

```
DoCmd.GoToRecord , , acNewRec
```

```
Exit_CmdAddNew_Click:
Exit Sub
```

```
Err_CmdAddNew_Click:
MsgBox Err.Description
Resume Exit_CmdAddNew_Click
```

```
End Sub
```

```
Public Sub changeDisplayLanguage()
    Dim tLabelLanguage(3, 16) As String, tLang As Integer

    Dim tRstLabelList As DAO.Recordset, ti As Integer

    Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

    tRstLabelList.Index = "label"

    gLabelsOK = False
    With tRstLabelList
        .MoveFirst
        ti = 1

        Do While ti < 16 And Not .EOF
            If !c_form = "SF_STATUS" Then
                gLabelsOK = True
                If ti <> !c_label_id Then
                    MsgBox "Uh oh: mismatched label table"
                    gLabelsOK = False
                    Exit Do
                End If
                tLabelLanguage(1, ti) = !c_english
                tLabelLanguage(2, ti) = !c_fanti
                tLabelLanguage(3, ti) = !c_jianti
                ti = ti + 1
            End If
            .MoveNext
        End While
    End With
End Sub
```

```
        Loop
    End With
    ' tRstLabelList.Close
    Set tRstLabelList = Nothing

    If gLabelsOK Then
        If gDisplayLanguage = "E" Then
            tLang = 1
        ElseIf gDisplayLanguage = "T" Then
            tLang = 2
        Else
            tLang = 3
        End If
        '
        ' now comes the basic routine
        '
        Me.LblSequence.Caption = tLabelLanguage(tLang, 1)
        Me.LblFirstyear.Caption = tLabelLanguage(tLang, 2)
        Me.LblFYNH.Caption = tLabelLanguage(tLang, 3)
        Me.LblFYRange.Caption = tLabelLanguage(tLang, 4)
        Me.LblLastyear.Caption = tLabelLanguage(tLang, 5)
        Me.LblLYNH.Caption = tLabelLanguage(tLang, 6)
        Me.LblLYRange.Caption = tLabelLanguage(tLang, 7)
        Me.LblPages.Caption = tLabelLanguage(tLang, 8)
        Me.LblNotes.Caption = tLabelLanguage(tLang, 9)
        Me.CmdPickSocDtn.Caption = tLabelLanguage(tLang, 10)
        Me.CmdPickFYNH.Caption = tLabelLanguage(tLang, 11)
        Me.CmdPickLYNH.Caption = tLabelLanguage(tLang, 12)
        Me.CmdPickSource.Caption = tLabelLanguage(tLang, 13)
        Me.CmdDelete.Caption = tLabelLanguage(tLang, 14)
        Me.CmdAddNew.Caption = tLabelLanguage(tLang, 15)
    End If

End Sub

Public Sub noEdits()

    Me.AllowAdditions = False
    Me.AllowDeletions = False
    Me.AllowEdits = False

End Sub
```

Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 16) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 16 And Not .EOF

If !c_form = "SF_STATUS" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

Me.LblSequence.Caption = tLabelLanguage(tLang, 1)

Me.LblFirstyear.Caption = tLabelLanguage(tLang, 2)

Me.LblFYNYHYear.Caption = tLabelLanguage(tLang, 3)

Me.LblFYRange.Caption = tLabelLanguage(tLang, 4)

Me.LblLastyear.Caption = tLabelLanguage(tLang, 5)

Me.LblLYNHyear.Caption = tLabelLanguage(tLang, 6)

Me.LblLYRange.Caption = tLabelLanguage(tLang, 7)

Me.LblPages.Caption = tLabelLanguage(tLang, 8)

Me.LblNotes.Caption = tLabelLanguage(tLang, 9)

Me.LblStatus.Caption = tLabelLanguage(tLang, 10)

Me.LblFYNH.Caption = tLabelLanguage(tLang, 11)

Me.LblLYNH.Caption = tLabelLanguage(tLang, 12)

Me.LblSource.Caption = tLabelLanguage(tLang, 13)

' Me.CmdDelete.Caption = tLabelLanguage(tLang, 14)

' Me.CmdAddNew.Caption = tLabelLanguage(tLang, 15)

End If

End Sub

Public Sub noEdits()

Me.AllowAdditions = False

Me.AllowDeletions = False

Me.AllowEdits = False

End Sub

form_subTreeView - 1

'Build 025

'BC Requires references to MS Forms 2.0 (C:\Windows\system32\FM20.DLL)
'ufTreeView UserForm object must be included in the VBA project
'modStartup module demonstrates how to create an userform programmatically
'and can be used in development to create a new ufTreeview. The modStartup
'code must be executed before trying to using subTreeview due to early-binding
'Late-binding the userform does not appear to work very well.

'You should ensure that ufTreeview userform is blank during development and ensure
'you don't save changes to the ufTreeview with controls added which may cause errors
'next it is used.

Option Compare Text

Option Explicit

Private Const GWL_EXSTYLE = (-20)
Private Const GWL_STYLE = (-16)
Private Const WS_EX_APPWINDOW As Long = &H40000
Private Const SWP_SHOWWINDOW As Long = &H40

Private Type RECT
pxLeft As Long
pxTop As Long
pxRight As Long
pxBottom As Long
End Type

#If VBA7 Then

Private Declare PtrSafe Function SetParent Lib "user32" (
ByVal hWndChild As LongPtr,
ByVal hWndNewParent As LongPtr) As LongPtr

Private Declare PtrSafe Function FindWindow Lib "user32" Alias "FindWindowA" (
ByVal lpClassName As String,
ByVal lpWindowName As String) As LongPtr

#If Win64 Then

Private Declare PtrSafe Function GetWindowLongPtr Lib "user32" Alias "GetWindowLongPtrA" (
ByVal hwnd As LongPtr,
ByVal nIndex As Long) As LongPtr

Private Declare PtrSafe Function SetWindowLongPtr Lib "user32" Alias "SetWindowLongPtrA" (
ByVal hwnd As LongPtr,
ByVal nIndex As Long,
ByVal dwNewLong As LongPtr) As LongPtr

#Else

Private Declare PtrSafe Function GetWindowLongPtr Lib "user32" Alias "GetWindowLongA" (
ByVal hwnd As LongPtr,
ByVal nIndex As Long) As LongPtr

Private Declare PtrSafe Function SetWindowLongPtr Lib "user32" Alias "SetWindowLongA" (
ByVal hwnd As LongPtr,
ByVal nIndex As Long,
ByVal dwNewLong As LongPtr) As LongPtr

#End If

Private Declare PtrSafe Function DrawMenuBar Lib "user32" (
ByVal hwnd As LongPtr) As Long

Private Declare PtrSafe Function GetWindowRect Lib "user32" (
ByVal hwnd As LongPtr,
lpRect As RECT) As Long

Private Declare PtrSafe Function SetWindowPos Lib "user32" (
ByVal hwnd As LongPtr,
ByVal hWndInsertAfter As LongPtr,
ByVal X As Long,
ByVal Y As Long,
ByVal cX As Long,
ByVal cY As Long,
ByVal uFlags As Long) As Long

Private Declare PtrSafe Function BringWindowToTop Lib "user32.dll" (
ByVal hwnd As LongPtr) As Long

```

        ByVal hwnd As LongPtr) As Long

Private mUFhWnd As LongPtr

#Else

Private Declare Function SetParent Lib "user32" ( _
    ByVal hWndChild As Long, _
    ByVal hWndNewParent As Long) As Long

Private Declare Function FindWindow Lib "user32" Alias "FindWindowA" ( _
    ByVal lpClassName As String, _
    ByVal lpWindowName As String) As Long

Private Declare Function GetWindowLongPtr Lib "user32" Alias "GetWindowLongA" ( _
    ByVal hwnd As Long, _
    ByVal nIndex As Long) As Long

Private Declare Function SetWindowLongPtr Lib "user32" Alias "SetWindowLongA" ( _
    ByVal hwnd As Long, _
    ByVal nIndex As Long, _
    ByVal dwNewLong As Long) As Long

Private Declare Function DrawMenuBar Lib "user32" ( _
    ByVal hwnd As Long) As Long

Private Declare Function GetWindowRect Lib "user32" ( _
    ByVal hwnd As Long, _
    lpRect As RECT) As Long

Private Declare Function SetWindowPos Lib "user32" ( _
    ByVal hwnd As Long, _
    ByVal hWndInsertAfter As Long, _
    ByVal X As Long, _
    ByVal Y As Long, _
    ByVal cX As Long, _
    ByVal cY As Long, _
    ByVal uFlags As Long) As Long

Private Declare Function BringWindowToTop Lib "user32.dll" ( _
    ByVal hwnd As Long) As Long

Private mUFhWnd As Long 'Ptr                'BC Pointer to the UserForm
#End If

Private mUF As ufTreeView
Private mcolTabControls As VBA.Collection    'BC A sorted list of controls that may be tabbed to.
Private mlMaxTabIndex As Long              'BC Identify the maximum value a containing form may have for
tab index

Private WithEvents moSubControl As Access.SubForm 'BC reference to the containing form's subform control.
Private WithEvents mfrTreeControl As MSForms.Frame 'BC reference to the treeview containing control
Private WithEvents mTreeview As clsTreeview      'BC reference to the treeview for easy access

Public Property Get pTreeview() As clsTreeview
' create a new treeclass and assign the TreeControl frame
Set mTreeview = New clsTreeview
Set mTreeview.TreeControl = mfrTreeControl
Set pTreeview = mTreeview
#If DebugMode = 1 Then
    If IsSubform Then
        Set mTreeview.Form = Me.Parent
    Else
        Set mTreeview.Form = Me
    End If
#End If
End Property

Public Property Get pTreeControl() As MSForms.Frame
' return a reference to the TreeControl frame,
' useful if want to change its default font properties which node labels inherit
Set pTreeControl = mfrTreeControl
End Property

Private Sub ApplyDefaultFont()
' Node labels will inherit font properties from the frame container
' Adapt as required -
'
' With pTreeControl.Font
'     .Name = "Calibri"

```

```

Form_SubTreeView - 3
'
' .Size = 9
' End With
' pTreeControl.ForeColor = 13995605 ' blue
'
End Sub

Private Function IsSubform() As Boolean
    On Error Resume Next
    IsSubform = (Not Me.Parent Is Nothing)
End Function

Private Sub Form_Close()
    If Not mTreeView Is Nothing Then
        mTreeView.TerminateTree
    End If
    Set mTreeView = Nothing
    Unload mUF
    Set mUF = Nothing
End Sub

Private Sub Form_Load()

#If Win64 Then
    Dim lStyle As LongPtr
    Dim res As LongPtr
#Else
    Dim lStyle As Long
    Dim res As Long
#End If

    Dim ctl As Access.Control
    Dim lngIndex As Long

    Set mUF = New ufTreeView

    'Determine if the form is loaded as a subform and if so,
    'find the containing subform control
    If IsSubform Then
        With Me.Parent
            For Each ctl In .Controls
                If ctl.ControlType = acSubform Then
                    If ctl.Form Is Me Then
                        Set moSubControl = ctl
                        Exit For
                    End If
                End If
            Next
        End With

        'When subform control has been found, enable it to raise events so that
        'the subTreeView can react to the Enter/Exit event of the subform control
        'Then, gather all controls within same section of the form where the subform
        'control resides and put them in a VBA collection using TabIndex as key,
        'so we have a sorted list of controls to tab through.
        If Not moSubControl Is Nothing Then
            moSubControl.OnEnter = "[Event Procedure]"
            moSubControl.OnExit = "[Event Procedure]"
            Set mcolTabControls = New VBA.Collection
            On Error Resume Next
            For Each ctl In moSubControl.Parent.Section(moSubControl.Section).Controls
                lngIndex = ctl.TabIndex
                If Err.Number = 0 Then
                    If ctl.TabIndex > mlMaxTabIndex Then
                        mlMaxTabIndex = ctl.TabIndex
                    End If
                    mcolTabControls.Add ctl.Name, CStr(ctl.TabIndex)
                Else
                    Err.Clear
                End If
            Next
            On Error GoTo 0
        End If
    End If

    ' load the userform
    Set mUF = New ufTreeView

    'For developer's convenience, the subTreeView's backcolor
    'can be set in design view to easily change the color of UF
    mUF.BackColor = Me.Section(acDetail).BackColor

```

```

'Add a frame as the treeview's container
'(the frame must be added to the userform at runtime, do not add any controls at design and save with the use
rform)
Set mfrTreeControl = mUF.Controls.Add("Forms.Frame.1", "frTreeControl", True)

ApplyDefaultFont ' apply any font properties to the frame for the treeview to inherit

'Assign Form's hWnd to mUF's caption, to ensure we don't find the wrong mUF
'and search for the UF's handle.
mUF.Caption = mUF.Caption & Me.hwnd
mUFhWnd = FindWindow("ThunderDFrame", mUF.Caption)

' make subTreeView the userform's parent window
res = SetParent(mUFhWnd, Me.hwnd)

' remove the userform's border and caption
lStyle = GetWindowLongPtr(mUFhWnd, GWL_STYLE)
lStyle = lStyle And Not &HC00000
SetWindowLongPtr mUFhWnd, GWL_STYLE, lStyle
SetWindowLongPtr mUFhWnd, GWL_EXSTYLE, WS_EX_APPWINDOW
DrawMenuBar mUFhWnd

ResizeUF
End Sub

Private Sub Form_Resize()
    ResizeUF
End Sub

Private Sub ResizeUF()
    Dim r As RECT

    'Ensure that the treeview will take up all space that
    'the subform control occupies. APIs are used so that
    'anchoring can be used on the subform container, which
    'allows for dynamic sizing of the treeview.
    If Not mfrTreeControl Is Nothing Then
        Me.Painting = False
        GetWindowRect Me.hwnd, r
        SetWindowPos mUFhWnd, &H0, 0, 0, r.pxRight - r.pxLeft, r.pxBottom - r.pxTop, SWP_SHOWWINDOW
        mfrTreeControl.Height = mUF.InsideHeight
        mfrTreeControl.Width = mUF.InsideWidth
        Me.Painting = True
    End If
End Sub

Private Sub mfrTreeControl_KeyDown(ByVal KeyCode As MSForms.ReturnInteger, ByVal Shift As Integer)
    Dim lngCurrentIndex As Long
    Dim lngStep As Long
    Dim ctl As Access.Control

    If KeyCode = vbKeyTab Then
        If IsSubform Then
            'Tab key was pressed so we need to find the
            'next control on the subform control's section
            'to set focus.
            lngCurrentIndex = moSubControl.TabIndex
            'If Shift was held down, we want to go backward
            If Shift Then
                lngStep = -1
            Else
                lngStep = 1
            End If
            Do
                lngCurrentIndex = lngCurrentIndex + lngStep
                Select Case True
                    Case lngCurrentIndex < 0
                        lngCurrentIndex = m1MaxTabIndex
                    Case lngCurrentIndex > m1MaxTabIndex
                        lngCurrentIndex = 0
                End Select
                Set ctl = moSubControl.Parent.Controls(mcolTabControls(CStr(lngCurrentIndex)))
                Select Case True
                    Case ctl.Enabled = False, ctl.Visible = False, ctl.TabStop = False
                        'Not eligible to receive focus, skip
                    Case Else
                        moSubControl.Parent.SetFocus
                        ctl.SetFocus
                        Exit Do
                End Select
            Loop
        End If
    End Sub

```

```
        End Select
    Loop Until lngCurrentIndex = moSubControl.TabIndex
End If
End If
End Sub

Private Sub moSubControl_Enter()
    'When the subform control gains focus, the focus
    'should be passed to the userform
    If Not mTreeview Is Nothing Then
        mTreeview.EnterExit False
        'mfrTreeControl.SetFocus
        BringWindowToTop mUFhWnd
    End If
End Sub

Private Sub moSubControl_Exit(Cancel As Integer)
    If Not mTreeview Is Nothing Then
        mTreeview.EnterExit bExit:=True
    End If
End Sub

Private Sub mTreeView_Click(cNode As clsNode)
    'We want to ensure that in case where user click
    'on the treeview directly, the containing form know
    'that the focus is now on the subform container
    moSubControl.SetFocus
End Sub
```


Option Compare Database

Public gDisplayLanguage As String, gLabelsOK As Boolean

Public Sub changeDisplayLanguage()

Dim tLabelLanguage(3, 6) As String, tLang As Integer

Dim tRstLabelList As DAO.Recordset, ti As Integer

Set tRstLabelList = CurrentDb.OpenRecordset("FormLabels", dbOpenTable)

tRstLabelList.Index = "label"

gLabelsOK = False

With tRstLabelList

.MoveFirst

ti = 1

Do While ti < 6 And Not .EOF

If !c_form = "SF_TEXTS" Then

gLabelsOK = True

If ti <> !c_label_id Then

MsgBox "Uh oh: mismatched label table"

gLabelsOK = False

Exit Do

End If

tLabelLanguage(1, ti) = !c_english

tLabelLanguage(2, ti) = !c_fanti

tLabelLanguage(3, ti) = !c_jianti

ti = ti + 1

End If

.MoveNext

Loop

End With

' tRstLabelList.Close

Set tRstLabelList = Nothing

If gLabelsOK Then

If gDisplayLanguage = "E" Then

tLang = 1

ElseIf gDisplayLanguage = "T" Then

tLang = 2

Else

tLang = 3

End If

'

' now comes the basic routine

'

Me.LblTitle.Caption = tLabelLanguage(tLang, 1)

Me.LblTitleChn.Caption = tLabelLanguage(tLang, 2)

Me.LblRole.Caption = tLabelLanguage(tLang, 3)

Me.LblWritings.Caption = tLabelLanguage(tLang, 4)

' Me.CmdDelete.Caption = tLabelLanguage(tLang, 5)

End If

End Sub

Public Sub noEdits()

Me.AllowAdditions = False

Me.AllowDeletions = False

Me.AllowEdits = False

End Sub

Form_ZZ_SCRATCH_GROUP_PLACE -1

Option Compare Database

Form_ZZ_SCRATCH_GROUP_STATUS--1

Option Compare Database

Form_ZZ_SCRATCH_GROUP_TEXT -1

Option Compare Database

Form_ZZ_SCRATCH_STATUS - 1

Option Compare Database

Form_ZZ_SCRATCH_TEXT - 1

Option Compare Database

Option Compare Database
Option Explicit

```
Private Sub cmdAddCol_Click()
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes ADD COLUMN type1 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes ADD COLUMN tdesc TEXT"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes ADD COLUMN subtype1 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes ADD COLUMN sdesc TEXT"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes ADD COLUMN type2 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes ADD COLUMN subtype2 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes ADD COLUMN type3 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes ADD COLUMN subtype3 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes ADD COLUMN id TEXT"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes ADD COLUMN category TEXT"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes ADD COLUMN comments TEXT"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes ADD COLUMN rank TEXT"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes ADD COLUMN kin TEXT"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes ADD COLUMN assoc TEXT"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes ADD COLUMN type1 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes ADD COLUMN subtype1 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes ADD COLUMN type2 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes ADD COLUMN subtype2 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes ADD COLUMN tcode TEXT"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes ADD COLUMN tdesc TEXT"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes ADD COLUMN tdescshn TEXT"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes ADD COLUMN scode TEXT"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes ADD COLUMN sdesc TEXT"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes ADD COLUMN sdescchn TEXT"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes ADD COLUMN songshichn TEXT"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes ADD COLUMN sameas TEXT"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes ADD COLUMN kincat TEXT"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes ADD COLUMN kintype TEXT"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes ADD COLUMN kindist TEXT"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes ADD COLUMN gen TEXT"
CurrentProject.Connection.Execute "ALTER TABLE office_codes ADD COLUMN c_category_1 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE office_codes ADD COLUMN c_category_2 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE office_codes ADD COLUMN c_category_3 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE office_codes ADD COLUMN c_category_4 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE office_codes ADD COLUMN c_office_type TEXT"
CurrentProject.Connection.Execute "ALTER TABLE office_codes ADD COLUMN c_office_subtype TEXT"
CurrentProject.Connection.Execute "ALTER TABLE office_codes ADD COLUMN c_level TEXT"
CurrentProject.Connection.Execute "ALTER TABLE office_codes ADD COLUMN c_fnc TEXT"
CurrentProject.Connection.Execute "ALTER TABLE office_codes ADD COLUMN c_rnk TEXT"
CurrentProject.Connection.Execute "ALTER TABLE status_codes ADD COLUMN type1 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE status_codes ADD COLUMN subtype1 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE status_codes ADD COLUMN type2 TEXT"
CurrentProject.Connection.Execute "ALTER TABLE status_codes ADD COLUMN subtype2 TEXT"
MsgBox ("Success!")
End Sub
```

```
Private Sub cmdClear_Click()
CurrentProject.Connection.Execute "DELETE FROM addresses"
CurrentProject.Connection.Execute "DELETE FROM appointment_type_codes"
CurrentProject.Connection.Execute "DELETE FROM assoc_codes"
CurrentProject.Connection.Execute "DELETE FROM assoc_data"
CurrentProject.Connection.Execute "DELETE FROM assume_office_codes"
CurrentProject.Connection.Execute "DELETE FROM bare_authors"
CurrentProject.Connection.Execute "DELETE FROM biog_addr_codes"
CurrentProject.Connection.Execute "DELETE FROM biog_addr_data"
CurrentProject.Connection.Execute "DELETE FROM biog_main"
CurrentProject.Connection.Execute "DELETE FROM choronym_codes"
CurrentProject.Connection.Execute "DELETE FROM country_codes"
CurrentProject.Connection.Execute "DELETE FROM dynasties"
CurrentProject.Connection.Execute "DELETE FROM entry_codes"
CurrentProject.Connection.Execute "DELETE FROM entry_data"
CurrentProject.Connection.Execute "DELETE FROM ethnicity_codes"
CurrentProject.Connection.Execute "DELETE FROM event_codes"
CurrentProject.Connection.Execute "DELETE FROM events_addr"
CurrentProject.Connection.Execute "DELETE FROM events_data"
CurrentProject.Connection.Execute "DELETE FROM extant_codes"
CurrentProject.Connection.Execute "DELETE FROM fix_altnames"
CurrentProject.Connection.Execute "DELETE FROM ganzhi_codes"
CurrentProject.Connection.Execute "DELETE FROM genre_codes"
CurrentProject.Connection.Execute "DELETE FROM kin_data"
CurrentProject.Connection.Execute "DELETE FROM kinship_codes"
CurrentProject.Connection.Execute "DELETE FROM literarygenre_codes"
CurrentProject.Connection.Execute "DELETE FROM measure_codes"
CurrentProject.Connection.Execute "DELETE FROM name_types"
CurrentProject.Connection.Execute "DELETE FROM nian_hao"
CurrentProject.Connection.Execute "DELETE FROM occasion_codes"
```

```

CurrentProject.Connection.Execute "DELETE FROM office_categories"
CurrentProject.Connection.Execute "DELETE FROM office_codes"
CurrentProject.Connection.Execute "DELETE FROM possession_act_codes"
CurrentProject.Connection.Execute "DELETE FROM possession_addr"
CurrentProject.Connection.Execute "DELETE FROM possession_data"
CurrentProject.Connection.Execute "DELETE FROM post_addr"
CurrentProject.Connection.Execute "DELETE FROM post_data"
CurrentProject.Connection.Execute "DELETE FROM scholarlytopic_codes"
CurrentProject.Connection.Execute "DELETE FROM school_codes"
CurrentProject.Connection.Execute "DELETE FROM status_codes"
CurrentProject.Connection.Execute "DELETE FROM status_data"
CurrentProject.Connection.Execute "DELETE FROM texts"
CurrentProject.Connection.Execute "DELETE FROM year_range_codes"
MsgBox ("Success!")
End Sub

Private Sub CmdDropCol_Click()
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes DROP COLUMN type1"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes DROP COLUMN tdesc"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes DROP COLUMN subtype1"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes DROP COLUMN sdesc"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes DROP COLUMN type2"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes DROP COLUMN subtype2"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes DROP COLUMN type3"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes DROP COLUMN subtype3"
CurrentProject.Connection.Execute "ALTER TABLE assoc_codes DROP COLUMN id"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes DROP COLUMN category"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes DROP COLUMN comments"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes DROP COLUMN rank"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes DROP COLUMN kin"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes DROP COLUMN assoc"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes DROP COLUMN type1"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes DROP COLUMN subtype1"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes DROP COLUMN type2"
CurrentProject.Connection.Execute "ALTER TABLE entry_codes DROP COLUMN subtype2"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes DROP COLUMN tcode"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes DROP COLUMN tdesc"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes DROP COLUMN tdescshn"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes DROP COLUMN scode"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes DROP COLUMN sdesc"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes DROP COLUMN sdescchn"
CurrentProject.Connection.Execute "ALTER TABLE genre_codes DROP COLUMN songshichn"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes DROP COLUMN sameas"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes DROP COLUMN kincat"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes DROP COLUMN kintype"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes DROP COLUMN kindist"
CurrentProject.Connection.Execute "ALTER TABLE kinship_codes DROP COLUMN gen"
CurrentProject.Connection.Execute "ALTER TABLE office_codes DROP COLUMN c_category_1"
CurrentProject.Connection.Execute "ALTER TABLE office_codes DROP COLUMN c_category_2"
CurrentProject.Connection.Execute "ALTER TABLE office_codes DROP COLUMN c_category_3"
CurrentProject.Connection.Execute "ALTER TABLE office_codes DROP COLUMN c_category_4"
CurrentProject.Connection.Execute "ALTER TABLE office_codes DROP COLUMN c_office_type"
CurrentProject.Connection.Execute "ALTER TABLE office_codes DROP COLUMN c_office_subtype"
CurrentProject.Connection.Execute "ALTER TABLE office_codes DROP COLUMN c_level"
CurrentProject.Connection.Execute "ALTER TABLE office_codes DROP COLUMN c_fnc"
CurrentProject.Connection.Execute "ALTER TABLE office_codes DROP COLUMN c_rnk"
CurrentProject.Connection.Execute "ALTER TABLE status_codes DROP COLUMN type1"
CurrentProject.Connection.Execute "ALTER TABLE status_codes DROP COLUMN subtype1"
CurrentProject.Connection.Execute "ALTER TABLE status_codes DROP COLUMN type2"
CurrentProject.Connection.Execute "ALTER TABLE status_codes DROP COLUMN subtype2"
MsgBox ("Success!")
End Sub

```

Form_ZZZ_DONT_PANIC - 1

Option Compare Database

```
Private Sub CommandClose_Click()  
On Error GoTo Err_CommandClose_Click
```

```
    DoCmd.Close
```

```
Exit_CommandClose_Click:  
Exit Sub
```

```
Err_CommandClose_Click:  
MsgBox Err.Description  
Resume Exit_CommandClose_Click
```

```
End Sub
```

Form_ZZZ_DONT_PANIC_PRC - 1

Option Compare Database

```
Private Sub CommandClose_Click()  
On Error GoTo Err_CommandClose_Click
```

```
    DoCmd.Close
```

```
Exit_CommandClose_Click:  
Exit Sub
```

```
Err_CommandClose_Click:  
MsgBox Err.Description  
Resume Exit_CommandClose_Click
```

```
End Sub
```

Form_ZZZ_DONT_PANIC_TW - 1

Option Compare Database

```
Private Sub CommandClose_Click()  
On Error GoTo Err_CommandClose_Click
```

```
    DoCmd.Close
```

```
Exit_CommandClose_Click:  
Exit Sub
```

```
Err_CommandClose_Click:  
MsgBox Err.Description  
Resume Exit_CommandClose_Click
```

```
End Sub
```


FixCBDB_extra_programs - 1

Public gDisplay As String
Option Compare Database

Sub add_kin_name()

Dim rstKIN As ADODB.Recordset
Dim rstBIOG As ADODB.Recordset
Dim tID As Long

Set rstKIN = New ADODB.Recordset
Set rstBIOG = New ADODB.Recordset

rstKIN.Open "kin_data", CurrentProject.Connection,
adLockOptimistic

rstBIOG.Open "BIOG_MAIN", CurrentProject.Connection,
adLockOptimistic

With rstKIN

.MoveFirst
.Find ("c_personid = 32534")

' .MoveNext

Do While Not .EOF

'

If .EOF Then

Exit Do

End If

'

tID = .Fields("c_kin_id")

rstBIOG.MoveFirst

rstBIOG.Find ("c_personid = " & S

If Not rstBIOG.EOF Then

```
FixCBDB_extra_programs - 2  
  
    Set rstAssoc = Nothing  
    Set rstBIOG = Nothing  
End Sub
```

ModStartup - 1

Option Compare Database
Option Explicit

'Demonstrate how to safely check if a userform is included in a VBA project and create it for the project's use

Public Function CheckUF() As Boolean

Dim bolRes As Boolean

'Determine if the userform is included and if not create it.

On Error Resume Next

Application.WizHook.Key = 51488399

With Application.WizHook.DbcVbProject

bolRes = Len(.VBComponents("ufTreeView").Name)

If Err.Number Then

Err.Clear

bolRes = False

If vbYes = MsgBox("Userform ufTreeView does not exist in this VBA Project. Create it?" & vbNewLine & vbNewLine & "Without ufTreeView, the treeview functionality will not be available.", vbYesNo, "Missing UserForm")

Then

With .VBComponents.Add(3) 'vbext_ct_MSForm

.Name = "ufTreeView"

End With

DoCmd.RunCommand acCmdCompileAndSaveAllModules

End If

bolRes = Len(.VBComponents("ufTreeView").Name)

If Err.Number Then

bolRes = False

End If

End If

End With

CheckUF = bolRes

On Error GoTo 0

End Function

Public Function StartUp() As Boolean

If CheckUF Then

DoCmd.OpenForm "frmDemo"

End If

End Function

Option Explicit
Option Compare Text

```
' Import this code into a new general code module in an empty workbook with a single worksheet and save as .xlsm
'
' Ensure the following object libraries are linked (Tools > References... option in VBE)
' Microsoft Office 16.0 Object Library
' Microsoft Forms 2.0 Object Library
' Microsoft Visual Basic for Applications Extensibility 5.3
' Microsoft Access 16.0 Object Library
' Microsoft Word 16.0 Object Library
'
' You can create a couple of buttons on the worksheet entitled "Export" and "Dedupe" pointing to macros
' ExportModuleCode() and RemoveDuplicates()
'
' Note that any open Access database or Word documents will be closed after their VBA has been exported
' Any open Excel documents will remain open after their VBA has been exported
```

```
Private Declare PtrSafe Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)
Private Declare PtrSafe Function Beep Lib "kernel32" (ByVal dwFreq As Long, ByVal dwDuration As Long) As Long
```

```
Public Const strSubfolder As String = "CodeStore"
Public strFileName As Variant
Public strFolderPath As String
Public VBProj As VBIDE.VBProject
```

```
Dim ws As Worksheet
Dim wkbk As Workbook
Dim iLastRow As Long
Dim iFiles As Integer
Dim iModules As Integer
Dim iTopRow As Long
```

```
Dim oWord As Word.Application
Dim oAccess As Access.Application
Dim oVBE As VBE
Dim oMod As VBComponent
Dim oProj As VBProject
Dim obj As VBComponent
Dim oFSO As Object
```

```
'=====+
' Main program code |
'=====+
```

```
Public Sub ExportModuleCode()
```

```
    Dim sFileArray As Variant
    Dim iPtr As Integer
    Dim dtStart As Date
    Dim iLineCount As Long
    Dim iFileFound As String
    Dim iDeleted As Long
    Dim dtTimeLimit As Date
    Dim sFileType As String
    Dim bWasOpen As Boolean

    Set ws = ThisWorkbook.Sheets(1) ' change this if you add extra worksheets

    ChDrive Left(ThisWorkbook.Path, 2)
    ChDir Mid(ThisWorkbook.Path & "\", 3)

    sFileArray = Application.GetOpenFilename(
        FileFilter:="All Macro-enabled Access/Excel/Word (*.mdb;*.accdb;*.xls;*.xlsm;*.doc;*.docm), *.mdb;*.accdb;*.xls;*.xlsm;*.doc;*.docm", _
        MultiSelect:=True)
    If Not IsArray(sFileArray) Then Exit Sub

    dtStart = Now()
    iModules = 0
    iFiles = 0
    Application.Cursor = xlWait

    ' set up some column headings
    With ws.Range("A1:E1")
        .Value = Array(vbCr & "Workbook File Name", "Module Name", "Export File Name", "Number" & vbCrLf & "Of Lines", "Date/Time")
        .Columns("A").ColumnWidth = 60
        .Columns("B").ColumnWidth = 30
```

```

Module1 - 2
.Columns("C").ColumnWidth = 80
.Columns("D").ColumnWidth = 12
.Columns("E").ColumnWidth = 24
.Font.Bold = True
.Interior.Pattern = xlSolid
.Interior.PatternColorIndex = xlAutomatic
.Interior.ThemeColor = xlThemeColorAccent1
.Interior.ThemeColor = xlThemeColorAccent1
.Interior.TintAndShade = 0.799981688894314
.Borders(xlDiagonalDown).LineStyle = xlNone
.Borders(xlDiagonalUp).LineStyle = xlNone
.Borders(xlEdgeLeft).LineStyle = xlContinuous
.Borders(xlEdgeTop).LineStyle = xlContinuous
.Borders(xlEdgeBottom).LineStyle = xlContinuous
.Borders(xlEdgeRight).LineStyle = xlContinuous
.Borders(xlInsideVertical).LineStyle = xlContinuous
.Borders(xlInsideHorizontal).LineStyle = xlContinuous
End With
' columns F:G not used here but they're used and cleared in RemoveDuplicates()

With ActiveWindow
    .SplitColumn = 0
    .SplitRow = 1
    .FreezePanels = True
End With

iLastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row
ActiveWindow.ScrollRow = 1
ActiveWindow.ScrollRow = IIf(iLastRow <= 12, 1, iLastRow - 12) ' sets the number of lines kept in view during p
rocessing

For Each strFileName In sFileArray
    DoEvents
    ' check file type up front - this simplifies any If...Then...ElseIf...EndIf or Select...Case coding
    If strFileName = ThisWorkbook.FullName Then
        sFileType = "This Excel"
    ElseIf Right(strFileName, 4) = ".xls" Or Right(strFileName, 5) = ".xlsm" Then
        sFileType = "Other Excel"
    ElseIf Right(strFileName, 4) = ".doc" Or Right(strFileName, 5) = ".docm" Then
        sFileType = "Word"
    ElseIf Right(strFileName, 4) = ".mdb" Or Right(strFileName, 6) = ".accdb" Then
        sFileType = "Access"
    End If
    '=====+
    ' Process this Excel workbook |
    '=====+
    If sFileType = "This Excel" Then
        iPtr = InStrRev(strFileName, "\")
        strFolderPath = Left(strFileName, iPtr)
        strFileName = Mid(strFileName, iPtr + 1)
        Set oFSO = CreateObject("Scripting.FileSystemObject")
        If Not oFSO.FolderExists(strFolderPath & strSubfolder) Then
            oFSO.CreateFolder (strFolderPath & strSubfolder)
        End If
        ' delete old export files
        iDeleted = 0
        iFileFound = Dir(strFolderPath & strSubfolder & "\" & strFileName & "_*.bas")
        Do Until iFileFound = ""
            iFileFound = Dir()
            iDeleted = iDeleted + 1
        Loop
        If iDeleted > 0 Then Kill strFolderPath & strSubfolder & "\" & strFileName & "_*.bas"
        Application.ScreenUpdating = True
        Application.EnableEvents = False
        Set wkbk = ThisWorkbook
        Application.EnableEvents = True
        Set VBProj = Application.Workbooks(strFileName).VBProject
        ' export each module type in turn: worksheet/workbook modules, type 100
        iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
        For Each obj In VBProj.VBComponents
            If obj.Type = 100 Then
                Call ExtractCode1(obj.Name)
            End If
        DoEvents
        Next obj
        If iLastRow > iTopRow Then Call SortLastSection
        ' export general code modules, type 1
        iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
        For Each obj In wkbk.VBProject.VBComponents
            If obj.Type = 1 Then

```

```

        Call ExtractCode1(obj.Name)
    End If
    DoEvents
Next obj
If iLastRow > iTopRow Then Call SortLastSection
If iLastRow > iTopRow Then Call SortLastSection
' export userform modules, type 3
iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
For Each obj In wkbk.VBProject.VBComponents
    If obj.Type = 3 Then
        Call ExtractCode1(obj.Name)
    End If
    DoEvents
Next obj
If iLastRow > iTopRow Then Call SortLastSection
' export class modules type 2
iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
For Each obj In wkbk.VBProject.VBComponents
    If obj.Type = 2 Then
        Call ExtractCode1(obj.Name)
    End If
    DoEvents
Next obj
If iLastRow > iTopRow Then Call SortLastSection
iFiles = iFiles + 1
End If
'=====+
' Process an external Excel workbook |
'=====+
If sFileType = "Other Excel" Then
    iPtr = InStrRev(strFileName, "\")
    strFolderPath = Left(strFileName, iPtr)
    strFileName = Mid(strFileName, iPtr + 1)
    Set oFSO = CreateObject("Scripting.FileSystemObject")
    If Not oFSO.FolderExists(strFolderPath & strSubfolder) Then
        oFSO.CreateFolder (strFolderPath & strSubfolder)
    End If
    ' delete old export files
    iDeleted = 0
    iFileFound = Dir(strFolderPath & strSubfolder & "\" & strFileName & "_*.bas")
    Do Until iFileFound = ""
        iFileFound = Dir()
        iDeleted = iDeleted + 1
    Loop
    If iDeleted > 0 Then Kill strFolderPath & strSubfolder & "\" & strFileName & "_*.bas"
    ' check whether it's open already
    If IsWorkBookOpen(strFileName) Then
        bWasOpen = True
        Set wkbk = Workbooks(strFileName)
    Else
        bWasOpen = False
        Application.EnableEvents = False
        Set wkbk = Workbooks.Open(strFolderPath & "\" & strFileName)
        Application.EnableEvents = True
    End If
    Windows(strFileName).Visible = False
    Set VBProj = Application.Workbooks(strFileName).VBProject
    ' export each module type in turn: worksheet/workbook modules, type 100
    iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
    For Each obj In VBProj.VBComponents
        If obj.Type = 100 Then
            Call ExtractCode1(obj.Name): Debug.Print obj.Name, obj.Type
        End If
        DoEvents
    Next obj
    If iLastRow > iTopRow Then Call SortLastSection
    ' export general code modules, type 1
    iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
    For Each obj In wkbk.VBProject.VBComponents
        If obj.Type = 1 Then
            Call ExtractCode1(obj.Name): Debug.Print obj.Name, obj.Type
        End If
        DoEvents
    Next obj
    If iLastRow > iTopRow Then Call SortLastSection
    ' export userform modules, type 3
    iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
    For Each obj In wkbk.VBProject.VBComponents
        If obj.Type = 3 Then
            Call ExtractCode1(obj.Name): Debug.Print obj.Name, obj.Type
        End If
    Next obj

```



```

        Next oMod
    Next oProj
    If iLastRow > iTopRow Then Call SortLastSection
    ' export class modules type 2
    iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
    For Each oProj In oVBE.VBProjects
        For Each oMod In oProj.VBComponents
            If oMod.Type = 2 Then
                Call ExtractCode2
            End If
            DoEvents
        Next oMod
    Next oProj
    If iLastRow > iTopRow Then Call SortLastSection
    Application.EnableEvents = False
    oWord.Documents.Open (strFolderPath & strFileName)
    Application.EnableEvents = True
    Windows(strFileName).Visible = True
    iFiles = iFiles + 1
    oWord.Quit
    Application.ScreenUpdating = True
    Set oVBE = Nothing
    Set oWord = Nothing
End If
'=====+
' Process an Access database                                     |
'=====+
If sFileType = "Access" Then
    iPtr = InStrRev(strFileName, "\")
    strFolderPath = Left(strFileName, iPtr)
    strFileName = Mid(strFileName, iPtr + 1)
    Set oFSO = CreateObject("Scripting.FileSystemObject")
    If Not oFSO.FolderExists(strFolderPath & strSubfolder) Then
        oFSO.CreateFolder (strFolderPath & strSubfolder)
    End If
    ' delete old export files
    iDeleted = 0
    iFileFound = Dir(strFolderPath & strSubfolder & "\" & strFileName & "_*.bas")
    Do Until iFileFound = ""
        iFileFound = Dir()
        iDeleted = iDeleted + 1
    Loop
    If iDeleted > 0 Then Kill strFolderPath & strSubfolder & "\" & strFileName & "_*.bas"
    Application.ScreenUpdating = False
    Set oAccess = CreateObject("Access.Application")
    Call oAccess.OpenCurrentDatabase(strFolderPath & strFileName)
    Set oVBE = oAccess.VBE
    ' export each module type in turn: database modules, type 100
    iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
    For Each oProj In oVBE.VBProjects
        For Each oMod In oProj.VBComponents
            If oMod.Type = 100 Then
                Call ExtractCode2
            End If
            DoEvents
        Next oMod
    Next oProj
    If iLastRow > iTopRow Then Call SortLastSection
    ' export general code modules, type 1
    iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
    For Each oProj In oVBE.VBProjects
        For Each oMod In oProj.VBComponents
            If oMod.Type = 1 Then
                Call ExtractCode2
            End If
            DoEvents
        Next oMod
    Next oProj
    If iLastRow > iTopRow Then Call SortLastSection
    ' export class modules type 2
    iTopRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
    For Each oProj In oVBE.VBProjects
        For Each oMod In oProj.VBComponents
            If oMod.Type = 2 Then
                Call ExtractCode2
            End If
            DoEvents
        Next oMod
    Next oProj
    If iLastRow > iTopRow Then Call SortLastSection

```



```

        iFiles = iFiles + 1
        oAccess.Quit
        Application.ScreenUpdating = True
        Set oVBE = Nothing
        Set oAccess = Nothing
    End If
    ActiveWindow.ScrollRow = IIf(iLastRow <= 12, 1, iLastRow - 12)
Next strFileName

Application.Cursor = xlDefault
Beep 1024, 30
Beep 768, 20
iLastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row
iLineCount = Application.Sum(Range("D2").Resize(iLastRow, 1))

MsgBox vbCrLf & "Done: "
    & Format(iFiles, "#,##0") & " file" & IIf(iFiles = 1, "", "s") & " read, "
    & Format(iModules, "#,##0") & " module" & IIf(iModules = 1, "", "s") & " written." & Space(10) & vbCrLf & vb
CrLf
    & Format(iLineCount, "#,##0") & " lines of code in library." & vbCrLf & vbCrLf
    & "Run time: " & Format(Now() - dtStart, "hh:nn:ss"), vbOKOnly + vbInformation, "Export Module Code v4"

End Sub

'=====+
' Export VBA code from Excel/Word |
'=====+

Private Sub ExtractCode1(ByVal argModuleName As String)

    Dim strExportFile As String
    Dim intFH As Integer
    Dim intLines As Long
    Dim strVBACode As String

    strExportFile = strFolderPath & strSubfolder & "\" & strFileName & "_" & argModuleName & ".bas"
    intLines = VBProj.VBComponents(argModuleName).CodeModule.CountOfLines
    If intLines > 0 Then
        strVBACode = VBProj.VBComponents(argModuleName).CodeModule.Lines(1, intLines)
    End If
    ' write a file even if the module was empty as this proves it exists
    Close
    intFH = FreeFile()
    Open strExportFile For Output As intFH
    Print #intFH, "Attribute VB_Name = "" & argModuleName & """"
    Print #intFH, strVBACode
    Close intFH
    iModules = iModules + 1
    iLastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
    Application.ScreenUpdating = True
    With ws
        .Cells(iLastRow, 1) = strFolderPath & strFileName
        .Cells(iLastRow, 2) = argModuleName & Replace(" (" & RealName(argModuleName) & ")", " ()", "")
        .Cells(iLastRow, 3) = strFolderPath & strSubfolder & "\" & strFileName & "_" & argModuleName & ".bas"
        .Cells(iLastRow, 4) = intLines
        .Cells(iLastRow, 5) = Now()
    End With
    Application.Wait Now() + TimeValue("00:00:01")
    Application.ScreenUpdating = False

End Sub

'=====+
' Export VBA code from Access |
'=====+

Private Sub ExtractCode2()

    Dim strExportFile As String
    Dim intFH As Integer
    Dim intLines As Long
    Dim strVBACode As String
    Dim strCleanName As String

    Application.ScreenUpdating = True
    strCleanName = Replace(oMod.Name, "/", "")
    strExportFile = strFolderPath & strSubfolder & "\" & strFileName & "_" & strCleanName & ".bas"
    intLines = oMod.CodeModule.CountOfLines
    If intLines > 0 Then
        strVBACode = oMod.CodeModule.Lines(1, intLines)
    End If

```

```

End If
' write a file even if the module was empty as this proves it exists
Close
intFH = FreeFile()
Open strExportFile For Output As intFH
Print #intFH, "Attribute VB_Name = "" & oMod.Name & """"
Print #intFH, strVBACode
Close intFH
iModules = iModules + 1
iLastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row + 1
With ws
    .Cells(iLastRow, 1) = strFolderPath & strFileName
    .Cells(iLastRow, 2) = oMod.Name
    .Cells(iLastRow, 3) = strFolderPath & strSubfolder & "\" & strFileName & "_" & oMod.Name & ".bas"
    .Cells(iLastRow, 4) = intLines
    .Cells(iLastRow, 5) = Now()
End With
Sleep 100
Application.ScreenUpdating = True

```

End Sub

```

'=====+
' For each module type within a project, sort the names into alphabetical order |
'=====+

```

Private Sub SortLastSection()

```

' for some reason ThisWorkbook modules are exported twice from Word, so delete the earlier one
If ws.Cells(iTopRow, "C") = ws.Cells(iLastRow, "C") Then
    ws.Rows(iTopRow).EntireRow.Delete
Else
    With ws.Sort
        .SortFields.Clear
        .SortFields.Add Key:=Range("B" & CStr(iTopRow) & ":B" & CStr(iLastRow)), SortOn:=xlSortOnValues, _
            Order:=xlAscending, DataOption:=xlSortNormal
        .SetRange Range("A" & CStr(iTopRow) & ":E" & CStr(iLastRow))
        .Header = xlNo
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With
End If

```

End Sub

Private Function IsWorkbookOpen(ByVal wbName As String) As Boolean

```

Dim oWB As Excel.Workbook

IsWorkbookOpen = False
For Each oWB In Application.Workbooks
    If oWB.Name = wbName Then
        IsWorkbookOpen = True
        Exit For
    End If
Next oWB
Set oWB = Nothing

```

End Function

Private Function RealName(ByVal rName As String) As String

```

Dim wks As Worksheet

For Each wks In wkbk.Sheets
    If obj.Type = 100 Then
        If LCase(rName) = LCase(wks.codename) Then RealName = wks.Name
    End If
Next wks

```

End Function

Public Sub RemoveDuplicates()

```

Dim ws As Worksheet
Dim iLastRow As Long
Dim iRow As Long
Dim iDuplicates As Long

```

```

Dim iDeleted As Long
Dim iInterval As Long
Dim iProgressBarWidth As Long
Dim iLineCount As Long
Dim dtStart As Date

Set ws = ThisWorkbook.Sheets(1) ' change this if you add extra worksheets dtStart = Now()
Application.ScreenUpdating = False
iLastRow = ws.Cells(ws.Rows.Count, 1).End(xlUp).Row

' preserve the original row numbers
Application.Calculation = xlCalculationManual
ws.Range("G2") = "=ROW()"
ws.Range("G2").AutoFill Destination:=ws.Range("G2:G" & CStr(iLastRow))
Application.Calculation = xlCalculationAutomatic
Application.CalculateFull
ws.Range("G2:G" & CStr(iLastRow)).Copy
ws.Range("G2:G" & CStr(iLastRow)).PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False

With ws.Sort ' sort by date exported to put the latest version of each file below any older versions
    With .SortFields
        .Clear
        .Add2 Key:=Range("E1:E" & CStr(iLastRow)), SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
        .Add2 Key:=Range("C1:C" & CStr(iLastRow)), SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    End With
    .SetRange Range("A1:G" & CStr(iLastRow))
    .Header = xlYes
    .MatchCase = False
    .Orientation = xlTopToBottom
    .SortMethod = xlPinYin
    .Apply
End With

Application.Calculation = xlCalculationManual
ws.Range("F2") = "=COUNTIF(C$2:C$" & CStr(iLastRow) & ",C2)-COUNTIF(C$2:C2,C2)"
ws.Range("F2").AutoFill Destination:=ws.Range("F2:F" & CStr(iLastRow))
Application.Calculation = xlCalculationAutomatic
Application.CalculateFull
ws.Range("F2:F" & CStr(iLastRow)).Copy
ws.Range("F2:F" & CStr(iLastRow)).PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks:=False, Transpose:=False

With ws.Sort ' sort by duplicate indicator where 0 = latest version of file, anything else is an older version
    With .SortFields
        .Clear
        .Add2 Key:=Range("F1:F" & CStr(iLastRow)), SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    End With
    .SetRange Range("A1:G" & CStr(iLastRow))
    .Header = xlYes
    .MatchCase = False
    .Orientation = xlTopToBottom
    .SortMethod = xlPinYin
    .Apply
End With

iDuplicates = Application.WorksheetFunction.CountIf(Range("F2:F" & CStr(iLastRow)), ">0")
For iRow = iLastRow To 2 Step -1
    DoEvents
    If ws.Cells(iRow, "F") = 0 Then ' no more duplicates
        Exit For
    Else ' delete this duplicate
        ws.Rows(iRow).ClearContents
        iDeleted = iDeleted + 1
    End If
Next iRow

iLastRow = ws.Cells(ws.Rows.Count, 1).End(xlUp).Row
With ws.Sort ' finally return the deduplicated entries back to their original positions in the worksheet
    With .SortFields
        .Clear
        .Add2 Key:=Range("G1:G" & CStr(iLastRow)), SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    End With
    .SetRange Range("A1:G" & CStr(iLastRow))
    .Header = xlYes
    .MatchCase = False

```

```

.Orientation = xlTopToBottom
.SortMethod = xlPinYin
.Apply
End With

```

```

Application.ScreenUpdating = True
ws.Columns("F:G").ClearContents
iLastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row
iLineCount = Application.Sum(Range("D2").Resize(iLastRow - 1, 1))
ActiveWindow.ScrollRow = IIf(iLastRow <= 12, 1, iLastRow - 12)

```

```

MsgBox vbCrLf & "Worksheet '" & ws.Name & "': "
      & IIf(iDeleted = 0, "no", Format(iDeleted, "#,##0")) & " duplicate record" & IIf(iDeleted = 1, " ", "s ")
-
      & IIf(iDeleted = 0, "found", "removed") & "." & Space(10) & vbCrLf & vbCrLf _
      & Space(4) & Format(iLastRow - 1, "#,##0") & " code modules currently in library." & Space(30) & vbCrLf &
vbCrLf
      & Space(4) & Format(iLineCount, "#,##0") & " lines of code in library." & vbCrLf & vbCrLf _
      & "Run time: " & Format(Now() - dtStart, "hh:nn:ss") & ".", _
      vbOKOnly + vbInformation, "Export Module Code v4"

```

```

End Sub

```



```

clsNode - 1
'Build 025
'*****
'
' Authors:   JKP Application Development Services, info@jkgp-ads.com, http://www.jkgp-ads.com
'           Peter Thornton, pmbthornton@gmail.com
'
' (c)2013, all rights reserved to the authors
'
' You are free to use and adapt the code in these modules for
' your own purposes and to distribute as part of your overall project.
' However all headers and copyright notices should remain intact
'
' You may not publish the code in these modules, for example on a web site,
' without the explicit consent of the authors
'*****
'-----
' Module      : clsNode
' Company     : JKP Application Development Services (c)
' Author      : Jan Karel Pieterse (www.jkgp-ads.com)
' Created     : 15-01-2013
' Purpose     : Holds all information of a node of the tree
'-----

Option Explicit

Private mbExpanded As Boolean

Private mcolChildNodes As Collection

Private moParentNode As clsNode
Private moLastActiveNode As clsNode
Private moTree As clsTreeview

Private msKey As String
Private mvCaption
Private msControlTipText As Variant

Private mlChecked As Long          ' PT checkbox tristate boolean 0/-1 or 1 for null
Private mbVisible As Boolean      ' PT determines if the node can be displayed
Private mnIndex As Long           ' PT order added to Treeview's mcolNodes, won't change
Private mlVisIndex As Long        ' PT the visible order in the current view, changes with expand/collapse
Private mvIconMainKey             ' PT string name or numeric index as icon Key for the Image collection
Private mvIconExpandedKey         ' PT ditto for expanded icon
Private mlIconCnt As Long         ' PT number of icons available for this node 0, 1 or 2
Private msngTextWidth As Single  ' PT autosized text width before the node is widened beyond the frame
Private mlBackColor As Long      ' PT
Private mbBold As Boolean         ' PT
Private mlForeColor As Long      ' PT
Private mvTag

Private WithEvents mctlControl As MSForms.Label
Private WithEvents mctlExpander As MSForms.Label
Private WithEvents moEditBox As MSForms.TextBox      ' PT editbox
Private WithEvents mctlCheckBox As MSForms.Label     ' PT checkbox

Private mctlExpanderBox As MSForms.Label
Private mctlVLine As MSForms.Label  ' PT vertical line, only the first child node with children will have a vertical line
Private mctlHLine As MSForms.Label  ' PT horizontal line
Private mctlIcon As MSForms.Image   ' PT separate icon image control

Public Enum ndSortOrder
    ndAscending = 1
    ndDescending = 2
End Enum

Public Enum ndCompareMethod
    ndBinaryCompare = 0
    ndTextCompare = 1
End Enum

Public Enum ndMouse
    ndDown = 1
    ndUp = 2
    ndMove = 3
    ndBeforeDragOver = 4
    ndBeforeDropOrPaste = 5
End Enum

#If Mac Then
    Const mcFullWidth As Long = 800
#Else

```

```

clsNode - 2
    Const mcFullWidth As Long = 600
#End If

'*****
'* Public Properties *
'*****

Public Property Get BackColor() As Long

    BackColor = mlBackColor ' if zero the treecaller will apply the frame container's backcolor

End Property

Public Property Let BackColor(lColor As Long)
'PT if lColor is written as 0/black, change it to 1 as 0 means default
    mlBackColor = lColor
    If mlBackColor = 0 Then mlBackColor = 1
    If Not mctlControl Is Nothing Then
        mctlControl.BackColor = lColor
    End If
End Property

Public Property Get Bold() As Boolean
    Bold = mbBold
End Property

Public Property Let Bold(bBold As Boolean)
    mbBold = bBold
    If Not mctlControl Is Nothing Then
        mctlControl.Font.Bold = mbBold
    End If
End Property

Public Property Get Caption()
    Caption = mvCaption
End Property

Public Property Let Caption(ByVal vCaption)
    mvCaption = vCaption
    If Not mctlControl Is Nothing Then
        mctlControl.Caption = CStr(vCaption)
    End If
End Property

Public Property Get Checked() ' PT
    ' Checked values are -1 true, 0 false, +1 mixed
    ' If TriState is enabled be careful not to return a potential +1 to a boolean or it'll coerce to True
    Checked = mlChecked
End Property

Public Property Let Checked(vChecked) ' PT
    Dim bFlag As Boolean, bTriState As Boolean
    Dim lChecked As Long
    Dim cChild As clsNode

    ' Checked values are -1 true, 0 false, +1 mixed
    ' if vChecked is a boolean Checked will coerce to -1 or 0
    ' if vChecked is Null Checked is set as +1

    If VarType(vChecked) = vbBoolean Then
        lChecked = vChecked
    ElseIf IsNull(vChecked) Then
        lChecked = 1
    ElseIf vChecked >= -1 And vChecked <= 1 Then
        lChecked = vChecked
    End If

    bFlag = lChecked <> mlChecked
    mlChecked = lChecked

    If Not mctlCheckBox Is Nothing And bFlag Then
        moTree.Changed = True
        UpdateCheckbox
    End If

    If Not moTree Is Nothing Then ' eg during clone
        bFlag = moTree.CheckBoxes(bTriState)
        If bTriState Then
            If ParentNode.Caption <> "RootHolder" Then
                ParentNode.CheckTriStateParent
            End If
        End If
    End If
End Property

```

clsNode - 3

```
        End If

        If Not ChildNodes Is Nothing Then
            For Each cChild In ChildNodes
                cChild.CheckTriStateChildren mlChecked
            Next
        End If
    End If
End If

End Property

Public Property Get Child() As clsNode
' PT Returns a reference to the first Child node, if any
    On Error Resume Next
    Set Child = mcolChildNodes(1)
End Property

Public Property Get ChildNodes() As Collection
    Set ChildNodes = mcolChildNodes
End Property

Public Property Set ChildNodes(colChildNodes As Collection)
    Set mcolChildNodes = colChildNodes
End Property

Public Property Get ControlTipText() As String
    ControlTipText = msControlTipText
End Property

Public Property Let ControlTipText(ByVal sControlTipText As String)
    msControlTipText = sControlTipText
    If Not mctlControl Is Nothing Then
        mctlControl.ControlTipText = msControlTipText
    End If
End Property

Public Property Get Expanded() As Boolean
    Expanded = mbExpanded
End Property

Public Property Let Expanded(ByVal bExpanded As Boolean)
    mbExpanded = bExpanded
    If Not Me.Expander Is Nothing Then
        UpdateExpanded bControlOnly:=False
    ElseIf Not Me.Control Is Nothing Then
        UpdateExpanded bControlOnly:=True
    End If
End Property

Public Property Get ForeColor() As Long
    ForeColor = mlForeColor
End Property

Public Property Let ForeColor(lColor As Long)
'PT if lColor is written as 0/black, change it to 1 as 0 means default
    mlForeColor = lColor
    If mlForeColor = 0 Then mlForeColor = 1
    If Not mctlControl Is Nothing Then
        mctlControl.ForeColor = lColor
    End If
End Property

Public Property Get FirstSibling() As clsNode
    If Not moParentNode Is Nothing Then ' PT Root has no parent
        Set FirstSibling = moParentNode.GetChild(1)
    End If
End Property

Public Property Get LastSibling() As clsNode
    If Not moParentNode Is Nothing Then ' PT Root has no parent
        Set LastSibling = moParentNode.GetChild(-1) ' -1 flags GetChild to return the last Child
    End If
End Property

Public Property Get ImageExpanded()
' PT string name or numeric index for the main icon key
    ImageExpanded = mvIconExpandedKey
End Property
```



```

Public Property Let ImageExpanded(vImageExpanded)
' PT string name or numeric index for an expanded icon key
On Error GoTo errExit
If Not IsMissing(vImageExpanded) Then
    If Not IsEmpty(vImageExpanded) Then
        If Len(mvIconMainKey) = 0 Then
            mvIconMainKey = vImageExpanded
        End If
        mvIconExpandedKey = vImageExpanded
        mIconCnt = 2
    End If
End If
errExit:
End Property

Public Property Get ImageMain()
' PT string name or numeric index for the main icon key
ImageMain = mvIconMainKey
End Property

Public Property Let ImageMain(vImageMain)
' PT string name or numeric index for the main icon key
On Error GoTo errExit
If Not IsMissing(vImageMain) Then
    If Not IsEmpty(vImageMain) Then
        mvIconMainKey = vImageMain
        If mIconCnt = 0 Then mIconCnt = 1
    End If
End If
errExit:
End Property

Public Property Get Key() As String
Key = msKey
End Property

Public Property Let Key(ByVal sKey As String)
Dim bIsInMainCol As Boolean
Dim i As Long
Dim cTmp As clsNode

On Error GoTo errH

If Tree Is Nothing Then
    msKey = sKey
    Exit Property
ElseIf msKey = sKey Or Len(sKey) = 0 Then
    Exit Property
End If

On Error Resume Next
Set cTmp = Tree.Nodes.Item(sKey)
On Error GoTo errH

If Not cTmp Is Nothing Then
    Err.Raise 457, , ' standard duplicate key error
End If

' to change the Key, remove Me and add Me back where it was with the new key
For Each cTmp In Tree.Nodes
    i = i + 1
    If cTmp Is Me Then
        bIsInMainCol = True
        Exit For
    End If
Next

If bIsInMainCol Then
    With Tree.Nodes
        .Remove i
        If .Count Then
            .Add Me, sKey, i
        Else
            .Add Me
        End If
    End With
Else
    ' Let Key called by via move/copy
End If

```

clsNode - 5

```
msKey = sKey

Exit Property

errH:
Err.Raise Err.Number, "Let Key", Err.Description
End Property

Public Property Get Level() As Long
Dim lLevel As Long
Dim cNode As clsNode

On Error GoTo errH
lLevel = -1
Set cNode = Me.ParentNode
While Not cNode Is Nothing
    lLevel = lLevel + 1
    Set cNode = cNode.ParentNode
Wend
Level = lLevel
Exit Property

errH:
#If DebugMode = 1 Then
    Stop
    Resume
#End If
End Property

Public Property Get NextNode() As clsNode ' can't name this proc 'Next' in VBA
' PT return the next sibling if there is one
Dim i As Long
Dim cNode As clsNode

With Me.ParentNode
    For Each cNode In .ChildNodes
        i = i + 1
        If cNode Is Me Then
            Exit For
        End If
    Next
    If .ChildNodes.Count > i Then
        Set NextNode = .ChildNodes(i + 1)
    End If
End With
End Property

Public Property Get ParentNode() As clsNode
Set ParentNode = moParentNode
End Property

Public Property Set ParentNode(oParentNode As clsNode)
Set moParentNode = oParentNode
End Property

Public Property Get Previous() As clsNode
' PT return the previous sibling if there is one
Dim i As Long
Dim cNode As clsNode

With Me.ParentNode
    For Each cNode In Me.ParentNode.ChildNodes
        i = i + 1
        If cNode Is Me Then
            Exit For
        End If
    Next
    If i > 1 Then
        Set NextNode = .ChildNodes(i - 1)
    End If
End With
End Property

Public Property Get Root() As clsNode
Dim cTmp As clsNode
Set cTmp = Me
Do While Not cTmp.ParentNode.ParentNode Is Nothing
    Set cTmp = cTmp.ParentNode
Loop
Set Root = cTmp
End Property
```

```

Public Property Get Tag()
    Tag = mvTag
End Property

Public Property Let Tag(vTag)
    mvTag = vTag
End Property

'*****
'* Public subs and functions *
'*****

Public Function Sort(Optional ByVal ndOrder As ndSortOrder = ndAscending, _
                    Optional ByVal ndCompare As ndCompareMethod = ndTextCompare) As Boolean
' PT Sorts the child nodes,
' returns True if the order has changed to flag Refresh should be called
Dim sCaptions() As String
Dim lStart As Long, lLast As Long, i As Long
Dim colNodes As New Collection
Dim bIsUnSorted As Boolean

On Error GoTo errExit
lStart = 1
lLast = ChildNodes.Count      ' error if no childnodes to sort

If lLast = 1 Then
    ' nothing to sort
    Exit Function
End If

ReDim idx(lStart To lLast) As Long
ReDim sCaptions(lStart To lLast) As String
For i = lStart To lLast
    idx(i) = i
    sCaptions(i) = ChildNodes.Item(i).Caption
Next

If ndOrder <> ndAscending Then ndOrder = -1      ' descending
If ndCompare <> ndTextCompare Then ndCompare = ndBinaryCompare

Call BinarySortIndexText(sCaptions(), lStart, lLast, idx, ndOrder, ndCompare)

For i = lStart To lLast - 1
    If idx(i) <> idx(i + 1) - 1 Then
        bIsUnSorted = True
        Exit For
    End If
Next

If bIsUnSorted Then
    For i = lStart To lLast
        colNodes.Add ChildNodes(idx(i))
    Next
    Set ChildNodes = colNodes
    Sort = True
End If

errExit:
' Probably(?) any error was because there were no childnodes, no need to raise an error
End Function

Public Function AddChild(Optional sKey As String, _
                        Optional vCaption, _
                        Optional vImageMain, _
                        Optional vImageExpanded) As clsNode

Dim cChild As clsNode

On Error GoTo errH
Set cChild = New clsNode

With moTree.Nodes

    If Len(sKey) Then
100         .Add cChild, sKey
101
        cChild.Key = sKey
    Else
        .Add cChild
    End If
End With

```

```

        End If

        cChild.Index = .Count
    End With

    If mcolChildNodes Is Nothing Then
        Set mcolChildNodes = New Collection
    End If

    mcolChildNodes.Add cChild

    With cChild
        If Not IsMissing(vImageMain) Then
            If Len(vImageMain) Then
                .ImageMain = vImageMain
            End If
        End If

        If Not IsMissing(vImageExpanded) Then
            If Len(vImageExpanded) Then
                .ImageExpanded = vImageExpanded
            End If
        End If

        .Caption = vCaption

        Set .Tree = moTree
        Set .ParentNode = Me
    End With

    Set AddChild = cChild

Exit Function
errH:
    #If DebugMode = 1 Then
        Stop
        Resume
    #End If

    If Err = 100 And Err.Number = 457 Then
        Err.Raise vbObjectError + 1, "clsNode.AddChild", "Duplicate key: '" & sKey & "'"
    Else
        Err.Raise Err.Number, "clsNode.AddChild", Err.Description
    End If
End Function

Public Function ChildIndex(sKey As String) As Long
'-----
' Procedure : ChildIndex
' Company   : JKP Application Development Services (c)
' Author    : Jan Karel Pieterse (www.jkp-ads.com)
' Created   : 15-01-2013
' Purpose    : Returns the index of a childnode using its key
'-----
    Dim cNode As clsNode
    Dim lCt As Long
    For Each cNode In mcolChildNodes
        lCt = lCt + 1
        If sKey = cNode.Key Then
            ChildIndex = lCt
            Set cNode = Nothing
            Exit Function
        End If
    Next
    Set cNode = Nothing
End Function

Public Function FullPath() As String
' PT, get all the grand/parent keys
' assumes use of key

    Dim s As String
    Dim cNode As clsNode

    On Error GoTo errDone
    s = Me.Key
    Set cNode = Me

    While Err.Number = 0
        Set cNode = cNode.ParentNode
    End While

```

```

        s = cNode.Key & "\" & s
    Wend

errDone:
    FullPath = s
End Function

Public Function GetChild(vKey As Variant) As clsNode
'-----
' Procedure : GetChild
' Company   : JKP Application Development Services (c)
' Author    : Jan Karel Pieterse (www.jkp-ads.com)
' Created   : 15-01-2013
' Purpose   : Returns a childnode using its key
'-----
    Dim cNode As clsNode
    Dim lIdx As Long

    If VarType(vKey) = vbString Then

        For Each cNode In mcolChildNodes
            If vKey = cNode.Key Then
                Set GetChild = cNode
                Set cNode = Nothing
                Exit Function
            End If
        Next

    ElseIf Not mcolChildNodes Is Nothing Then
        lIdx = vKey
        If lIdx = -1 Then
            lIdx = mcolChildNodes.Count
        End If
        If lIdx > 0 Then
            Set GetChild = mcolChildNodes(lIdx)
        Else: Set mcolChildNodes = Nothing
        End If
    End If

    Set cNode = Nothing
End Function

'*****
'*      Friend Properties, Subs & Funtions      *
'*      ** these procedures are visible throughout the project but should *
'*      ** only be used to communicate with the TreeView, ie clsTreeView *
'*****

Friend Property Get Control() As MSForms.Label
    Set Control = mctlControl
End Property

Friend Property Set Control(ctlControl As MSForms.Label)
    Set mctlControl = ctlControl
    If Not mctlControl Is Nothing Then
        If Not moTree Is Nothing Then
            Set mctlControl.Font = moTree.TreeControl.Font
        Else
            Stop
        End If
    End If
End Property

Friend Property Get Index() As Long      ' PT
    Index = mnIndex
End Property

Friend Property Let Index(idx As Long)
' PT Index: the order this node was added to Treeview's collection mcolNodes
' Index will never increase but may decrement if previously added nodes are removed
    mnIndex = idx
End Property

Friend Property Let VisIndex(lVisIndex As Long)
    mVisIndex = lVisIndex
End Property

Friend Property Get VisIndex() As Long      ' PT
    VisIndex = mVisIndex

```

```

End Property

Friend Property Get Tree() As clsTreeview
    Set Tree = moTree
End Property

Friend Property Set Tree(oTree As clsTreeview)
    Set moTree = oTree
End Property

Friend Property Get Checkbox() As MSForms.Control
    Set Checkbox = mctlCheckBox
End Property

Friend Property Set Checkbox(oCtl As MSForms.Control)
    Set mctlCheckBox = oCtl
End Property

Friend Property Get Expander() As MSForms.Label
    Set Expander = mctlExpander
End Property

Friend Property Set Expander(ctlExpander As MSForms.Label)
    Set mctlExpander = ctlExpander
End Property

Friend Property Get ExpanderBox() As MSForms.Label
    Set ExpanderBox = mctlExpanderBox
End Property

Friend Property Set ExpanderBox(ctlExpanderBox As MSForms.Label)
    Set mctlExpanderBox = ctlExpanderBox
End Property

Friend Property Set HLine(ctlHLine As MSForms.Label)
    Set mctlHLine = ctlHLine
End Property

Friend Property Get HLine() As MSForms.Label
    Set HLine = mctlHLine
End Property

Friend Property Set Icon(ctlIcon As MSForms.Image)
    Set mctlIcon = ctlIcon
End Property

Friend Property Get Icon() As MSForms.Image
    Set Icon = mctlIcon
End Property

Friend Property Get TextWidth() As Single
    TextWidth = msngTextWidth
End Property

Friend Property Let TextWidth(sngTextWidth As Single)
    msngTextWidth = sngTextWidth
End Property

Friend Property Get VLine() As MSForms.Label
    Set VLine = mctlVLine
End Property

Friend Property Set VLine(ctlVLine As MSForms.Label)
    Set mctlVLine = ctlVLine
End Property

Friend Sub CheckTriStateParent()
    ' PT set triState value of parent according to its childnodes' values
    Dim alChecked(-1 To 1) As Long
    Dim cChild As clsNode

    If Not ChildNodes Is Nothing Then
        For Each cChild In ChildNodes
            alChecked(cChild.Checked) = alChecked(cChild.Checked) + 1
        Next
        If alChecked(1) Then
            alChecked(1) = 1
        ElseIf alChecked(-1) = ChildNodes.Count Then
            alChecked(1) = -1
        ElseIf alChecked(0) = ChildNodes.Count Then

```

```

        alChecked(1) = 0
    Else
        alChecked(1) = 1
    End If

    If Checked <> alChecked(1) Then
        mlChecked = alChecked(1)
        UpdateCheckbox
    End If

End If

If Not Me.Caption = "RootHolder" Then
    If Not ParentNode.ParentNode Is Nothing Then
        ParentNode.CheckTriStateParent
    End If
End If

End Sub

Friend Sub CheckTriStateChildren(lChecked As Long)
' PT, make checked values of children same as parent's
' only called if triState is enabled
Dim cChild As clsNode

mlChecked = lChecked
UpdateCheckbox

If Not ChildNodes Is Nothing Then
    For Each cChild In ChildNodes
        cChild.CheckTriStateChildren lChecked
    Next
End If

End Sub

Friend Function hasIcon(vKey) As Boolean
' PT get the appropriate icon key/index, if any
If mlIconCnt = 2 And mbExpanded Then
    vKey = mvIconExpandedKey
    hasIcon = True 'Not IsEmpty(vKey) '(True
ElseIf mlIconCnt Then
    vKey = mvIconMainKey
    hasIcon = True 'Not IsEmpty(vKey)
End If

End Function

Friend Sub EditBox(bEnterEdit As Boolean) ' PT new in 006PT2 ,,move to clsTreeView?
'-----
' Procedure : moCtl_Click
' Author : Peter Thornton
' Created : 20-01-2013
' Purpose : Enter/exit Editmode, show/hide the edit textbox
'-----

On Error Resume Next
Set moEditBox = moTree.TreeControl.Controls("EditBox")
On Error GoTo 0

If bEnterEdit Then

    If moEditBox Is Nothing Then
        Set moEditBox = moTree.TreeControl.Controls.Add("forms.textbox.1", False)
        moEditBox.Name = "EditBox"
    End If

    With moEditBox
        .Left = Control.Left - 3
        .Top = Control.Top - 1.5
        .AutoSize = True
        .BorderStyle = fmBorderStyleSingle
        .Text = Caption
        Control.Visible = False ' hide the node label while editing
        .ZOrder 0
        .Visible = True
        .SelStart = 0
        .SelLength = Len(.Text)
        .SetFocus
    End With

ElseIf Not moEditBox Is Nothing Then
    ' exit editmode

```

clsNode - 11

```
        If Not moEditBox Is Nothing Then
            ' error if moEditBox has already been removed
            On Error Resume Next
            moEditBox.Visible = False
            moEditBox.Text = ""
            Set moEditBox = Nothing
        End If
        Control.Visible = True

    End If
End Sub

Friend Function RemoveChild(cNode As clsNode) As Boolean
'PT remove a node from the collection,
' note, this is only one part of the process of removing a node

    Dim lCt As Long
    Dim cTmp As clsNode
    On Error GoTo errH

    For Each cTmp In mcolChildNodes
        lCt = lCt + 1
        If cTmp Is cNode Then
            mcolChildNodes.Remove lCt
            RemoveChild = True
            Exit For
        End If
    Next

    If mcolChildNodes.Count = 0 Then
        Set mcolChildNodes = Nothing
        Me.Expanded = False
    End If

    Exit Function
errH:
    Err.Raise vbObjectError, "RemoveChild", Err.Description
End Function

Friend Sub RemoveNodeControls()
    Dim cChild As clsNode
    If Not ChildNodes Is Nothing Then
        For Each cChild In ChildNodes
            cChild.RemoveNodeControls
        Next
    End If
    DeleteNodeControls False
End Sub

Friend Sub TerminateNode(Optional bDeleteNodeControls As Boolean)
'-----
' Procedure : TerminateNode
' Company   : JKP Application Development Services (c)
' Author    : Jan Karel Pieterse (www.jkp-ads.com)
' Created   : 15-01-2013
' Purpose   : Terminates the class instance
'-----
    Dim cChild As clsNode
    'Instead of the Terminate event of the class we use this public
    'method so it can be explicitly called by parent classes.
    'This is done because to break the two way or circular references
    'between the parent child classes.

    'The most important call in this routine is to destroy the reference
    'between this node class and the parent treeview class -
    ' < Set moTree = Nothing >
    'Once all the moTree references to have been destroyed everything else will
    ' 'tear down' normally

    If Not ChildNodes Is Nothing Then
        For Each cChild In ChildNodes
            ' recursively drill down to all child nodes in this branch
            cChild.TerminateNode bDeleteNodeControls
        Next
    End If

    ' If deleting individual nodes while the treeview is running we also want to
    ' remove all associated controls as well as removing references

    If bDeleteNodeControls Then
```


clsNode - 12

```
        DeleteNodeControls True
        If bDeleteNodeControls Then
            Index = -1
        End If
    End If

    Set mcolChildNodes = Nothing
    Set moTree = Nothing
End Sub

'*****
'* Private subs and functions *
'*****

Private Sub BinarySortIndexText(sCaptions() As String, ByVal lStart As Long, ByVal lEnd As Long, ByRef idx() As Long, ndOrder As Long, ndCompare As ndCompareMethod)
' PT sorts the index array based on the string array
    Dim lSmall As Long, lLarge As Long, sMid As String, lTmp As Long

    lSmall = lStart
    lLarge = lEnd
    sMid = sCaptions(idx((lSmall + lLarge) / 2))

    Do While lSmall <= lLarge
        Do While (StrComp(sCaptions(idx(lSmall)), sMid, ndCompare) = -ndOrder And lSmall < lEnd)
            lSmall = lSmall + 1
        Loop
        Do While (StrComp(sCaptions(idx(lLarge)), sMid, ndCompare) = ndOrder And lLarge > lStart)
            lLarge = lLarge - 1
        Loop
        If lSmall <= lLarge Then
            lTmp = idx(lSmall)
            idx(lSmall) = idx(lLarge)
            idx(lLarge) = lTmp
            lSmall = lSmall + 1
            lLarge = lLarge - 1
        End If
    Loop

    If lStart <= lLarge Then
        Call BinarySortIndexText(sCaptions(), lStart, lLarge, idx, ndOrder, ndCompare)
    End If
    If lSmall <= lEnd Then
        Call BinarySortIndexText(sCaptions(), lSmall, lEnd, idx, ndOrder, ndCompare)
    End If
End Sub

Private Sub DeleteNodeControls(bClearIndex As Boolean)
'PT Delete all controls linked to this node

    On Error GoTo errH

    With moTree.TreeControl.Controls
        If Not mctlControl Is Nothing Then
            .Remove mctlControl.Name
            Set mctlControl = Nothing
            If Not mctlHLine Is Nothing Then
                .Remove mctlHLine.Name
                Set mctlHLine = Nothing
            End If
            If Not mctlIcon Is Nothing Then
                .Remove mctlIcon.Name
                Set mctlIcon = Nothing
            End If
            If Not mctlIcon Is Nothing Then
                .Remove mctlIcon.Name
                Set mctlIcon = Nothing
            End If
        End If

        If Not mctlExpander Is Nothing Then
            .Remove mctlExpander.Name
            Set mctlExpander = Nothing
        End If
        If Not mctlExpanderBox Is Nothing Then
            .Remove mctlExpanderBox.Name
            Set mctlExpanderBox = Nothing
        End If
        If Not mctlVLine Is Nothing Then
```

clsNode - 13

```
.Remove mctlVLine.Name
Set mctlVLine = Nothing
End If

If Not moEditBox Is Nothing Then
    .Remove moEditBox.Name
    Set moEditBox = Nothing
End If
If Not mctlCheckBox Is Nothing Then
    .Remove mctlCheckBox.Name
    Set mctlCheckBox = Nothing
End If

If Not Me.ParentNode Is Nothing Then
    ' if Me is the last child delete parent's expander and VLine (if it has one)
    If FirstSibling Is LastSibling Then

        If Not Me.ParentNode.VLine Is Nothing Then
            .Remove Me.ParentNode.VLine.Name
            Set Me.ParentNode.VLine = Nothing
        End If

        If Not Me.ParentNode.ExpanderBox Is Nothing Then
            .Remove Me.ParentNode.ExpanderBox.Name
            Set Me.ParentNode.ExpanderBox = Nothing
        End If

        If Not Me.ParentNode.Expander Is Nothing Then
            .Remove Me.ParentNode.Expander.Name
            Set Me.ParentNode.Expander = Nothing
        End If

        Me.ParentNode.Expanded = False

    End If

End If

End With

If bClearIndex Then
    Me.Index = -1 ' flag this node to be removed from mcolNodes in NodeRemove
End If

Exit Sub
errH:
    ' Stop
    Resume Next
End Sub

Private Function UpdateCheckbox()
Dim pic As StdPicture
    If Not mctlCheckBox Is Nothing Then
        With mctlCheckBox
            If moTree.GetCheckboxIcon(mlChecked, pic) Then
                .Picture = pic
            Else
                .Caption = IIf(mlChecked, "a", "")
                If (mlChecked = 1) <> (.ForeColor = RGB(180, 180, 180)) Then
                    .ForeColor = IIf(mlChecked = 1, RGB(180, 180, 180), vbWindowText)
                End If
            End If
        End With
    End If
End Function

Private Sub UpdateExpanded(bControlOnly As Boolean)
'-----
' Procedure : UpdateExpanded
' Author    : Peter Thornton
' Created   : 27-01-2013
' Purpose   : Called via an Expander click or arrow keys
'             Updates the Expanded property and changes +/- caption
'-----
Dim bFullWidth As Boolean
Dim vKey
Dim pic As StdPicture

If Not bControlOnly Then
```

```

        With Me.Expander
            If moTree.GetExpanderIcon(mbExpanded, pic) Then
                .Picture = pic
            Else
                If mbExpanded Then
                    .Caption = "-"
                Else
                    .Caption = "+"
                End If
            End If
        End With
    End If

    On Error GoTo errExit
    If Me.hasIcon(vKey) Then
        If moTree.GetNodeIcon(vKey, pic, bFullWidth) Then
            If bFullWidth Then
                Me.Icon.Picture = pic ' potential error if Icon is nothing, let error abort
            Else
                Me.Control.Picture = pic
            End If
        End If
    End If
errExit:
End Sub

```

```

*****
'*   Node Events       *
*****

```

```

Private Sub mctlCheckBox_Click() ' PT new in 006PT2
'-----
' Procedure : moCtl_Click
' Author    : Peter Thornton
' Created   : 20-01-2013
' Purpose   : Event fires when a Checkbox label is clicked
'-----

    If moTree.EditMode(Me) Then
        ' exit editmode if in editmode
        moTree.EditMode(Me) = False
    End If
    If mlChecked = 0 Then

        Checked = -1
    Else
        Checked = 0
    End If

    Set moTree.ActiveNode = Me
    moTree.NodeClick mctlCheckBox, Me ' share the checkbox click event
End Sub

```

```

Private Sub mctlControl_Click()
'-----
' Procedure : mctlControl_Click
' Company   : JKP Application Development Services (c)
' Author    : Jan Karel Pieterse (www.jkp-ads.com)
' Created   : 15-01-2013
' Purpose   : Event fires when a treebranch is clicked
'-----

```

```

' PT the call to NodeClick will raise the click event to the form
Dim bFlag As Boolean

```

```

    If Not moLastActiveNode Is Nothing Then
        moLastActiveNode.Control.BorderStyle = fmBorderStyleNone
        Set moLastActiveNode = Nothing
        bFlag = True
    End If

```

```

    If moTree.ActiveNode Is Nothing Then
        Set moTree.ActiveNode = Me
        bFlag = True
    ElseIf Not bFlag Then
        bFlag = mctlControl.BorderStyle <> fmBorderStyleNone
    End If

```

```

    If Not moTree.ActiveNode Is Me Or bFlag Then
        ' only raise the event the first time the node is activated
    End If

```

```

        moTree.NodeClick Control, Me

        ' if preferred the click event is always raised to the form (even if the
        ' node was previously active) simply comment or remove this If/EndIf check
    End If

End Sub

Private Sub mctlControl_DblClick(ByVal Cancel As MSForms.ReturnBoolean)
' PT a node label has been double-clicked, enter edit-mode if manual editing is enabled
    Dim bDummy As Boolean

    If moTree.EnableLabelEdit(bDummy) Then
        moTree.EditMode(Me) = True
        EditText bEnterEdit:=True
    End If

End Sub

Private Sub mctlControl_MouseDown(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
' PT temporarily activate and highlight the MouseDown node and a grey border to the previous activenode
' MouseUp and Click events will confirm the action or reset the previous active node
    Dim bFlag As Boolean

    If moTree.ActiveNode Is Me Then
        bFlag = Me.Control.BackColor = vbHighlight
        ' bFlag = bFlag Or Me.Control.BorderStyle = fmBorderStyleSingle ' in Access this should be uncommented
    End If

    If Not bFlag Then
        Set moLastActiveNode = moTree.ActiveNode
        Set moTree.ActiveNode = Me
        If Not moLastActiveNode Is Nothing Then
            moLastActiveNode.Control.BorderStyle = fmBorderStyleSingle
            moLastActiveNode.Control.BorderColor = RGB(200, 200, 200)
        End If
    End If

    If moTree.EditMode(Me) Then
        ' if any node is in edit mode exit edit mode
        moTree.EditMode(Me) = False
    End If

End Sub

Private Sub mctlControl_MouseUp(ByVal Button As Integer, ByVal Shift As Integer, ByVal X As Single, ByVal Y As Single)
' PT MouseUp fires before the Click event, at this point we don't know 100% if user
' definately wants to activate the MouseDown node. If user drags the mouse off the MouseDown node the
' Click event will not fire which means user wants to cancel and revert to the previous activenode.
' If MouseUp occurs with the cursor not over the node reset the original activenode

    Dim bFlag As Boolean
    Dim bMouseIsOver As Boolean
    Dim bMoveCopy As Boolean

    If Not moLastActiveNode Is Nothing Then
        With Me.Control
            ' is the mouse over the node or within a pixel of it
            bMouseIsOver = (X >= -1 And X <= .Width + 1) And (Y >= -1 And Y <= .Height + 1)
        End With

        If Not bMouseIsOver Then
            ' if the last-activenode was marked for MoveCopy we will need to reset it
            bFlag = moLastActiveNode Is moTree.MoveCopyNode(bMoveCopy)

            ' reset the original activenode
            moLastActiveNode.Control.BorderStyle = fmBorderStyleNone
            Set moTree.ActiveNode = moLastActiveNode

            If bFlag Then
                Set moTree.MoveCopyNode(bMoveCopy) = moLastActiveNode
            End If

            Set moLastActiveNode = Nothing
        End If

        ElseIf Button = 2 Then
            ' the click event doesn't fire with right click so explicitly call it

```

```

        mctlControl_Click
    End If
End If

End Sub

Private Sub mctlExpander_Click()
    Expanded = Not Expanded
    If moTree.EditMode(Me) Then
        ' if any node is in edit mode exit edit mode
        moTree.EditMode(Me) = False
    End If
    Tree.NodeClick Expander, Me
End Sub

Private Sub moEditBox_KeyDown(ByVal KeyCode As MSForms.ReturnInteger, ByVal Shift As Integer) 'PT
    ' PT Textbox key events to Enter or Esc the Editbox, 006PT2

    Dim bCancel As Boolean
    Dim bSort As Boolean
    Dim sNewText As String

    If KeyCode = vbKeyReturn Then
        sNewText = moEditBox.Value
        If sNewText = Caption Then
            KeyCode = vbKeyEscape
        Else
            bCancel = moTree.RaiseAfterLabelEdit(Me, sNewText)
            If Not bCancel Then
                Me.Caption = moEditBox.Value
                Control.Caption = sNewText
                Control.AutoSize = True
                TextWidth = Control.Width
                Control.AutoSize = False
                If TextWidth < mcFullWidth And moTree.FullWidth Then
                    Control.Width = mcFullWidth
                End If
                moTree.UpdateScrollLeft ' scroll to show all the label
                moTree.Changed = True
                moTree.NodeClick Control, Me
                bCancel = moTree.LabelEdit(bSort)
                If bSort Then
                    If Me.ParentNode.Sort Then
                        moTree.Refresh
                    End If
                End If
            End If
            EditBox False
        End If
    End If
    If KeyCode = vbKeyEscape Then
        moTree.EditMode(Me) = False
        EditBox False
    End If
End Sub

Private Sub Class_Initialize()
    ' default properties
    mbExpanded = True ' default

    #If DebugMode = 1 Then
        gClsNodeInit = gClsNodeInit + 1 ' PT, for testing only, remove, see ClassCounts() in the normal module
    #End If
End Sub

Private Sub Class_Terminate()
    #If DebugMode = 1 Then
        gClsNodeTerm = gClsNodeTerm + 1 ' PT, for testing,
    #End If
    Set moTree = Nothing
End Sub

```

```

'Build 025
'*****
'
' Authors:   JKP Application Development Services, info@jkgp-ads.com, http://www.jkgp-ads.com
'           Peter Thornton, pmbthornton@gmail.com
'
' (c)2013, all rights reserved to the authors
'
' You are free to use and adapt the code in these modules for
' your own purposes and to distribute as part of your overall project.
' However all headers and copyright notices should remain intact
'
' You may not publish the code in these modules, for example on a web site,
' without the explicit consent of the authors
'*****
'-----
' Module      : clsTreeView
' Company     : JKP Application Development Services (c)
' Author      : Jan Karel Pieterse (www.jkgp-ads.com)
' Created     : 15-01-2013
' Purpose     : Creates a VBA Treeview control in a frame on your UserForm
'-----

Option Explicit
#Const HostProject = "Access" ', or Excel or Word

Public WithEvents TreeControl As MSForms.Frame

Private mbInactive As Boolean          'PT the treeview is not in focus
Private mbAlwaysRedesign As Boolean    'PT temporary flag to force mbRedesign=true, see Move()
Private mbAutoSort As Boolean          'PT sort siblings after manual edit
Private mbChanged As Boolean           'PT "dirty", user has edited node(s)
Private mbCheckboxes As Boolean        'PT show checkboxes
Private mbLabelEdit As Boolean         'PT allow manual editing with F2 and double click
Private mbTriState As Boolean          'PT enable tripple state checkboxes
Private mbCheckboxImage As Boolean     'PT determins if icons are used for checkboxes
Private mbEditMode As Boolean          'PT flag if in editmode
Private mbFullWidth As Boolean         'PT use separate image controls for icons, can highlight nodes to full width
Private mbGotIcons As Boolean          'PT got a collection of images
Private mbExpanderImage As Boolean     'PT determines if icons will be used for collapse/expand controls
Private mbKeyDown As Boolean           'PT Enter-keyup in a Textbox occurs when next control gets focus
Private mbMove As Boolean              'PT flag intention of the MoveCopyNode
Private mbRedesign As Boolean          'PT flag to reset all dim's after changing NodeHeight or Indentation at runti
me
Private mbRootButton As Boolean        'PT Root has an expander button
Private mbShowExpanders As Boolean     'PT Show +/- buttons
Private mbShowLines As Boolean         'PT determines if lines will be created and shown
Private mlBackColor As Long           'PT frameholder's backcolor
Private mlForeColor As Long           'PT frameholder's ForeColor
Private mlLabelEdit As Long           'PT 0-Automatic, 1-Manual can't be edited
Private mlNodesCreated As Long        'PT in/de-cremented as clsNodes are added/deleted from mcolNodes
Private mlNodesDeleted As Long        'PT incremented as clsNode.controls are deleted, purpose to give unique id fo
r control names
Private mlVisCount As Long            'PT incremented from zero as each node is displayed
Private mlVisOrder() As Long          'PT an index array to identify displayed nodes in the order as displayed
Private msAppName As String           'JKP: Title of messageboxes
Private msngChkBoxPad As Single       'PT offset if using checkboxes
Private msngChkBoxSize As Single      'PT checkbox size
Private msngIndent As Single          'PT default 15
Private msngLineLeft As Single        'PT Left pos of Root H & V lines, 3 + alpha
Private msngNodeHeight As Single      'JKP: vertical distance between nodes
Private msngRootLine As Single        'PT if mbRootButton, same as msngIndent, else 0
Private msngTopChk As Single          'PT top checkbox (these "tops" are offsets down from the top a given node)
Private msngTopExpB As Single          'PT top expander button (a label)
Private msngTopExpT As Single         'PT top expander text (a label)
Private msngTopHV As Single           'PT top for Horiz' & Vert' lines (mid height of a node + top padding))
Private msngTopIcon As Single         'PT top icon
Private msngTopLabel As Single        'PT top node label, if font height less than NodeHeight
Private msngVisTop As Single          'PT activenode top relative to scroll-top
Private msngMaxWidths() As Single     'PT array, max width of text in each level, helps determine scroll-width
Private moActiveNode As clsNode       'JKP: refers to the selected node
Private moEditNode As clsNode         'PT the node in EditMode
Private moMoveNode As clsNode         'PT node waiting to be moved
Private moRootHolder As clsNode       'PT parent for the root node(s), although a clsNode it's not a real node
Private mcolIcons As Collection        'PT collection of stdPicture objects, their names as keys
Private mcolNodes As Collection       'JKP: global collection of all the nodes
Private moCheckboxImage(-1 To 1) As StdPicture 'PT checkbox true/false/triState icons
Private moExpanderImage(-1 To 0) As StdPicture 'PT collapse/expand icons
#If HostProject = "Access" Then
Private moForm As Access.Form         'PT the main form, eg to return debug stats to the caption

```

```

#Else
    Private moForm As MSForms.UserForm
#End If
'-----

'Public Enum tvMouse
'    tvDown = 1
'    tvUp = 2
'    tvMove = 3
'    tvBeforeDragOver = 4
'    tvBeforeDropOrPaste = 5
'End Enum

Public Enum tvTreeRelationship
    tvFirst = 0
    tvLast = 1
    tvNext = 2
    tvPrevious = 3
    tvChild = 4
End Enum

Event Click(cNode As clsNode)          'Node click event
Event NodeCheck(cNode As clsNode)      'Checkbox change event
Event AfterLabelEdit(ByRef Cancel As Boolean, NewString As String, cNode As clsNode)
Event KeyDown(cNode As clsNode, ByVal KeyCode As MSForms.ReturnInteger, ByVal Shift As Integer)

Private Type POINTAPI
    X As Long
    Y As Long
End Type

#If VBA7 And Not Mac Then
    Private Declare PtrSafe Function GetCursorPos Lib "user32.dll" ( _
        ByRef lpPoint As POINTAPI) As Long
    Private Declare PtrSafe Function SetCursorPos Lib "user32.dll" ( _
        ByVal X As Long, _
        ByVal Y As Long) As Long
    Private Declare PtrSafe Function getTickCount Lib "kernel32.dll" Alias "GetTickCount" () As Long
#Else
    Private Declare Function GetCursorPos Lib "user32.dll" ( _
        ByRef lpPoint As POINTAPI) As Long
    Private Declare Function SetCursorPos Lib "user32.dll" ( _
        ByVal X As Long, _
        ByVal Y As Long) As Long
    Private Declare Function getTickCount Lib "kernel32.dll" Alias "GetTickCount" () As Long
#End If

' Mac displays at 72 pixels per 72 points vs (typically) 96/72 in Windows
' The respective constants help size and position node controls appropriately in the different OS
' Search the project for instances of the Mac constant

#If Mac Then
    Const mcCheckboxFont As Long = 13
    Const mcCheckboxPad As Single = 19
    Const mcCheckboxPadImg As Single = 15
    Const mcChkBoxSize As Single = 13
    Const mcExpanderFont As Long = 13
    Const mcExpButSize As Single = 15
    Const mcExpBoxSize As Long = 12
    Const mcFullWidth As Long = 800
    Const mcIconPad As Single = 17
    Const mcIconSize As Long = 16
    Const mcTLpad As Long = 4
    Const mcLineLeft As Single = mcTLpad + 10
    Const mcPtPx1 As Single = 1
#Else
    Const mcCheckboxFont As Long = 10
    Const mcCheckboxPad As Single = 15
    Const mcCheckboxPadImg As Single = 11.25
    Const mcChkBoxSize As Single = 10.5
    Const mcExpanderFont As Long = 10
    Const mcExpButSize As Single = 11.25
    Const mcExpBoxSize As Long = 9
    Const mcFullWidth As Long = 600
    Const mcIconPad As Single = 14.25
    Const mcIconSize As Long = 12
    Const mcTLpad As Long = 3
    Const mcLineLeft As Single = mcTLpad + 7.5
    Const mcPtPx1 As Single = 0.75
#End If

```

```
Private Const mcSource As String = "clsTreeView"
```

```
*****
'*      Public Properties      *
*****
```

```
Public Property Get ActiveNode() As clsNode
    Set ActiveNode = moActiveNode
End Property
```

```
Public Property Set ActiveNode(oActiveNode As clsNode)
```

```
'-----
' Procedure : ActiveNode
' Company   : JKP Application Development Services (c)
' Author    : Jan Karel Pieterse (www.jkp-ads.com)
' Created   : 17-01-2013
' Purpose   : Setting the activenode also updates the node colors
'             and ensures the node is scrolled into view
'-----
```

```
Dim cTmp As clsNode
If oActiveNode Is MoveCopyNode(False) Then
    Set MoveCopyNode(False) = Nothing
End If
```

```
If moActiveNode Is oActiveNode Then
    SetActiveNodeColor
    Exit Property
End If
```

```
ResetActiveNodeColor ActiveNode
```

```
If oActiveNode.Control Is Nothing Then
    Set cTmp = oActiveNode.ParentNode
    While Not cTmp.Caption = "RootHolder"
        cTmp.Expanded = True
        Set cTmp = cTmp.ParentNode
    Wend
```

```
    If mlNodesCreated Then
        BuildRoot False
    End If
```

```
End If
```

```
Set moActiveNode = oActiveNode
SetActiveNodeColor
```

```
End Property
```

```
Public Sub ExpandNode(cNode As clsNode)
    Dim cTmp As clsNode
```

```
    Set cTmp = cNode.ParentNode
    While Not cTmp.Caption = "RootHolder"
        cTmp.Expanded = True
    Wend
```

```
End Sub
```

```
Public Property Get AppName() As String
    AppName = msAppName
End Property
```

```
Public Property Let AppName(ByVal sAppName As String)
    msAppName = sAppName
End Property
```

```
Public Property Get Changed() As Boolean
'PT user has edited node(s) and/or changed Checked value(s)
    Changed = mbChanged
End Property
```

```
Public Property Let Changed(ByVal bChanged As Boolean)
' called after manual node edit and Checked change
    mbChanged = bChanged
End Property
```

```
Public Property Get CheckBoxes(Optional bTriState As Boolean) As Boolean ' PT
```



```

    CheckBoxes = mbCheckboxes
    bTriState = mbTriState
End Property

Public Property Let CheckBoxes(Optional bTriState As Boolean, ByVal bCheckboxes As Boolean) ' PT
    Dim bOrig As Boolean
    Dim bOrigTriState As Boolean

    bOrig = mbCheckboxes
    mbCheckboxes = bCheckboxes

    bOrigTriState = mbTriState
    mbTriState = bTriState
    If bCheckboxes Then
        msngChkBoxPad = mcCheckboxPad
        If msngNodeHeight < mcExpButSize Then msngNodeHeight = mcExpButSize
    Else
        msngChkBoxPad = 0
    End If

    If Not TreeControl Is Nothing Then
        If TreeControl.Controls.Count And (bOrig <> mbCheckboxes Or bOrigTriState <> mbTriState) Then
            ' Checkboxes added changed after start-up so update the treeview
            mbRedesign = True
            Refresh
        End If
    End If
End Property

#If HostProject = "Access" Then
    Public Property Set Form(frm As Access.Form)
        Set moForm = frm
    End Property
#Else
    Public Property Set Form(frm As MSForms.UserForm)
        Set moForm = frm
    End Property
#End If

Public Property Get FullWidth() As Boolean
    FullWidth = mbFullWidth
End Property

Public Property Let FullWidth(bFullWidth As Boolean)
    mbFullWidth = bFullWidth
End Property

Public Property Set Images(objImages As Object)
    Dim sDesc As String
    Dim pic As stdole.StdPicture
    Dim obj As Object
    ' PT objImages can be a collection of StdPicture objects
    ' a Frame containing only Image controls (or controls with an image handle)
    ' stdole.IPictureDisp or stdole.StdPicture objects

    On Error GoTo errH
    If TypeName(objImages) = "Collection" Then
        Set mcolIcons = objImages
        For Each pic In mcolIcons
            ' if not a valid picture let the error abort
        Next
    Else
        Set mcolIcons = New Collection

        '#If HostProject = "Access" Then
            ' if the frame is on an Access form include .Object
            'For Each obj In objImages.Object.Controls

200         For Each obj In objImages.Controls
            mcolIcons.Add obj.Picture, obj.Name
        Next
    End If

    ' Flag we have a valid collection of images
    mbGotIcons = mcolIcons.Count >= 1

    Exit Property
errH:

```

```

Set mcolIcons = Nothing
If Erl = 100 Then
    sDesc = "The obImages collection includes an invaluse StdPicture object"
ElseIf Erl = 200 Then
    sDesc = "A control in objImages does not contain a valid Picture object"
End If
sDesc = sDesc & vbCrLf & Err.Description

Err.Raise Err.Number, "Images", sDesc

End Property

Public Property Get Indentation() As Single
    Indentation = msngIndent
End Property

Public Property Let Indentation(sngIndent As Single)
    Dim cNode As clsNode
    Dim sngOld As Single

    sngOld = msngIndent

    #If Mac Then
        If sngIndent < 16 Then
            msngIndent = 16      ' min indent ?
        ElseIf sngIndent > 80 Then
            msngIndent = 80      ' max indent
        Else
            msngIndent = Int(sngIndent)
        End If
    #Else
        If sngIndent < 12 Then
            msngIndent = 12      ' min indent ?
        ElseIf sngIndent > 60 Then
            msngIndent = 60      ' max indent
        Else
            msngIndent = Int((sngIndent * 2 + mcPtPxl) / 3 * 2) * mcPtPxl
        End If
    #End If

    If mbRootButton Then msngRootLine = msngIndent

    If Not TreeControl Is Nothing And Not (sngOld = msngIndent) Then
        ' changed after start-up so update the treview
        If TreeControl.Controls.Count Then
            Set cNode = Me.ActiveNode
            Refresh
            If Not cNode Is Nothing Then
                Set ActiveNode = cNode
            End If
        End If
    End If
End Property

Public Property Get EnableLabelEdit(Optional bAutoSort As Boolean) As Boolean
    EnableLabelEdit = mbLabelEdit
    bAutoSort = mbAutoSort
End Property

Public Property Let EnableLabelEdit(Optional bAutoSort As Boolean, ByVal bLabelEdit As Boolean)    ' PT
    ' optional bAutoSort: automatically resort siblings after a manual edit
    mbLabelEdit = bLabelEdit
    mbAutoSort = bAutoSort
End Property

Public Property Get LabelEdit(Optional bAutoSort As Boolean) As Long    ' PT
    ' PT, equivalent to Treeview.LabelEdit
    ' 0/tvwAutomatic nodes can be manually edited
    ' optional bAutoSort: automatically resort siblings after a manual edit

    LabelEdit = mlLabelEdit
    bAutoSort = mbAutoSort
End Property

Public Property Let LabelEdit(Optional bAutoSort As Boolean, ByVal nLabelEdit As Long)    ' PT
    mlLabelEdit = nLabelEdit
    mbLabelEdit = (nLabelEdit = 0)
    mbAutoSort = bAutoSort
End Property

Public Property Get MoveCopyNode(Optional bMove As Boolean, Optional lColor As Long) As clsNode

```

```

    bMove = mbMove
    Set MoveCopyNode = moMoveNode
End Property
Public Property Set MoveCopyNode(Optional bMove As Boolean, Optional lColor As Long, cNode As clsNode)
    Static lOrigBackColor As Long

    mbMove = bMove
    If lColor = 0 Then
        If bMove Then
            lColor = RGB(255, 231, 162)
        Else: lColor = RGB(159, 249, 174)
        End If
    End If

    If Not moMoveNode Is Nothing Then
        moMoveNode.BackColor = lOrigBackColor
        moMoveNode.Control.BackColor = lOrigBackColor
        Set moMoveNode = Nothing
    Else
        End If

    If Not cNode Is Nothing Then
        lOrigBackColor = cNode.BackColor
        If lOrigBackColor = 0 Then lOrigBackColor = mlBackColor
        cNode.BackColor = lColor

        cNode.Control.BackColor = cNode.BackColor
        cNode.Control.ForeColor = cNode.ForeColor
        Set moMoveNode = cNode
    Else
        End If
End Property

'Public Property Get MultiSelect() As Boolean
'    MultiSelect = mbMultiSelect
'End Property
'Public Property Let MultiSelect(mbMultiSelect As Boolean)
'    mbMultiSelect = MultiSelect
'End Property

Public Property Get NodeHeight() As Single
    If msngNodeHeight = 0 Then msngNodeHeight = 12
    NodeHeight = msngNodeHeight
End Property

Public Property Let NodeHeight(ByVal sngNodeHeight As Single)
    Dim cNode As clsNode
    Dim sngOld As Single

    sngOld = msngNodeHeight

    #If Mac Then
        If sngNodeHeight < 12 Then ' height of expander-box is 9
            msngNodeHeight = 12
        ElseIf sngNodeHeight > 60 Then
            msngNodeHeight = 60
        Else
            msngNodeHeight = Int(sngNodeHeight)
        End If
    #Else
        If sngNodeHeight < 9 Then ' height of expander-box is 9
            msngNodeHeight = 9
        ElseIf sngNodeHeight > 45 Then
            msngNodeHeight = 45
        Else
            msngNodeHeight = Int((sngNodeHeight * 2 + mcPtPx1) / 3 * 2) * mcPtPx1
        End If
    #End If

    If mbRootButton Then msngRootLine = msngIndent
    If Not TreeControl Is Nothing And Not (sngOld = msngNodeHeight) Then
        If TreeControl.Controls.Count Then
            Set cNode = Me.ActiveNode
            Refresh
            If Not cNode Is Nothing Then
                Set ActiveNode = cNode
            End If
        End If
    End If

```

```

clsTreeView - 7

    End If
End Property

Public Property Get Nodes() As Collection
' Global collection of the nodes
' *DO NOT USE* its Nodes.Add and Nodes.Remove methods
' To add & remove nodes use clsNode.AddChild() or clsTreeView.NodeAdd and clsTeevView.NodeRemove()
    If mcolNodes Is Nothing Then Set mcolNodes = New Collection
    Set Nodes = mcolNodes
End Property

Public Property Get RootButton() As Boolean
    If mbRootButton Then RootButton = 1
End Property

Public Property Let RootButton(lRootLeader As Boolean)
' PT The Root nodes have expanders and lines (if mbShowlines)

    mbRootButton = lRootLeader
    If mbRootButton Then
        msngRootLine = msngIndent
    Else
        msngRootLine = 0
    End If

    If Not Me.TreeControl Is Nothing Then
        If Not moRootHolder Is Nothing Then
            If Not moRootHolder.ChildNodes Is Nothing Then
                Refresh
            End If
        End If
    End If
End Property

Public Property Get RootNodes() As Collection
'PT returns the collection of Root-nodes
' **should be treated as read only. Use AddRoot and NodeRemove to add/remove a root node**
    Set RootNodes = moRootHolder.ChildNodes
End Property

Public Property Get ShowExpanders() As Boolean
    ShowExpanders = mbShowExpanders
End Property

Public Property Let ShowExpanders(bShowExpanders As Boolean)

    mbShowExpanders = bShowExpanders

    If Not TreeControl Is Nothing Then
        If TreeControl.Controls.Count Then
            Refresh
        End If
    End If
End Property

Public Property Get ShowLines() As Boolean
    ShowLines = mbShowLines
End Property

Public Property Let ShowLines(bShowLines As Boolean)
' PT Show horizontal & vertical lines
Dim bOrig As Boolean
Dim cNode As clsNode

    bOrig = mbShowLines
    mbShowLines = bShowLines

    If Not TreeControl Is Nothing Then
        If TreeControl.Controls.Count Then
            If bOrig <> mbShowLines Then
                ' ShowLines added after start-up so update the treeview
                Refresh
            End If
        End If
    End If
End Property

End Property

*****
'*      Public functions and subs      *

```

```
*****
```

```
Public Function AddRoot(Optional sKey As String, Optional vCaption, Optional vImageMain, _
    Optional vImageExpanded) As clsNode
```

```
On Error GoTo errH
```

```
If moRootHolder Is Nothing Then
    Set moRootHolder = New clsNode
    Set moRootHolder.ChildNodes = New Collection
    Set moRootHolder.Tree = Me
    moRootHolder.Caption = "RootHolder"
    If mcolNodes Is Nothing Then
        Set mcolNodes = New Collection
    End If
End If
```

```
Set AddRoot = moRootHolder.AddChild(sKey, vCaption, vImageMain, vImageExpanded)
```

```
Exit Function
```

```
errH:
    #If DebugMode = 1 Then
        Stop
        Resume
    #End If
    Err.Raise Err.Number, "AddRoot", Err.Description
```

```
End Function
```

```
Public Sub CheckboxImage(picFalse As StdPicture, picTrue As StdPicture, Optional picTriState As StdPicture)
```

```
On Error GoTo errExit:
Set moCheckboxImage(0) = picFalse
Set moCheckboxImage(-1) = picTrue
If Not IsMissing(picTriState) Then
    Set moCheckboxImage(1) = picTriState
End If
```

```
mbCheckboxImage = True
```

```
errExit:
```

```
End Sub
```

```
Public Sub EnterExit(bExit As Boolean)
```

```
'PT WithEvents can't trap Enter/Exit events, if we need them here they can be
' called from the TreeControl's Enter/Exit events in the form
mbInactive = bExit
SetActiveNodeColor bExit ' apply appropriate vbInactiveCaptionText / vbHighlight
```

```
End Sub
```

```
Public Sub ExpanderImage(picMinus As StdPicture, picPlus As StdPicture)
```

```
On Error GoTo errExit:
Set moExpanderImage(0) = picPlus
Set moExpanderImage(-1) = picMinus
mbExpanderImage = True
```

```
errExit:
```

```
End Sub
```

```
Public Sub ExpandToLevel(lExpansionLevel As Long, Optional bReActivate As Boolean = True)
```

```
' PT call SetTreeExpansionLevel and reactivates the closest expanded parent if necessary
' eg, if activeNode.level = 4 and lExpansionLevel = 2, the activenode's grandparent will be activated
Dim cTmp As clsNode
```

```
Call SetTreeExpansionLevel(lExpansionLevel - 1)
```

```
If bReActivate Then
    If ActiveNode.Level > lExpansionLevel Then
        Set cTmp = ActiveNode.ParentNode
        While cTmp.Level > lExpansionLevel
            Set cTmp = cTmp.ParentNode
        Wend
        Set ActiveNode = cTmp
    End If
End If
```

```
End Sub
```

```
Public Sub Copy(cSource As clsNode, cDest As clsNode, _
    Optional vBefore, Optional ByVal vAfter, _
    Optional ByVal bShowError As Boolean)
```

```

Set MoveCopyNode(False) = Nothing
Clone cDest, cSource, vBefore, vAfter
SetActiveNodeColor

```

```
End Sub
```

```

Public Sub Move(cSource As clsNode, cDest As clsNode, _
    Optional vBefore, Optional ByVal vAfter, _
    Optional ByVal bShowError As Boolean)
' PT Move source node + children to destination node
' cannot move the Root and cannot move to a descendant
' vBefore/vAfter work as for normal collection; error if invalid, eg a new collection, after the last item, etc
'

Dim sErrDesc As String
Dim bIsParent As Boolean
Dim cNode As clsNode
Dim cSourceParent As clsNode

Set MoveCopyNode(False) = Nothing
On Error GoTo errH

If cSource Is Nothing Or cDest Is Nothing Or cSource Is cDest Then ' Or cSource Is Root
Exit Sub
End If

Set cNode = cDest
bIsParent = False
Do
Set cNode = cNode.ParentNode
bIsParent = cNode Is cSource
Loop Until cNode Is Nothing Or bIsParent

If bIsParent Then
Err.Raise vbObjectError + 110
End If

If cDest.ChildNodes Is Nothing Then
' the child becomes a parent for the first time
Set cDest.ChildNodes = New Collection
' expander & VLine will get created automatically if necessary
End If

AddNodeToCol cDest.ChildNodes, cSource, False, vBefore, vAfter

Set cSourceParent = cSource.ParentNode
With cSourceParent
.RemoveChild cSource '
' if the old parent has no more children remove its expander & VLine

If .ChildNodes Is Nothing Then

If Not .Expander Is Nothing Then
Me.TreeControl.Controls.Remove .Expander.Name
Set .Expander = Nothing
End If

If Not .ExpanderBox Is Nothing Then
Me.TreeControl.Controls.Remove .ExpanderBox.Name
Set .ExpanderBox = Nothing
End If

If Not .VLine Is Nothing Then
Me.TreeControl.Controls.Remove .VLine.Name
Set .VLine = Nothing
End If

.Expanded = False

End If
End With

Set cSource.ParentNode = cDest
cDest.Expanded = True

If mbTriState Then
cDest.CheckTriStateParent
cSourceParent.CheckTriStateParent
End If

SetActiveNodeColor

```

```

    mbAlwaysRedesign = True      ' ensure Left's get recalc'd during future refresh

Exit Sub
errH:

Select Case Err.Number
Case vbObjectError + 110
    sErrDesc = "Cannot cut and move a Node to a descendant node"
Case Else
    sErrDesc = "Move: " & Err.Description
End Select

If bShowError Then
    MsgBox sErrDesc, , AppName
Else
    Err.Raise Err.Number, mcSource, "Move: " & sErrDesc
End If

End Sub

Public Function NodeAdd(Optional vRelative, _
                        Optional vRelationship, _
                        Optional sKey As String, _
                        Optional vCaption, _
                        Optional vImageMain, _
                        Optional vImageExpanded) As clsNode      ' As tvTreevRelationship

'PT, similar to the old tv's nodes.add method
'  main difference is vRelative can be a Node object as well as a key or index
'  see also clsNode.AddChild

Dim i As Long
Dim cNode As clsNode
Dim cRelative As clsNode
Dim cParent As clsNode
Dim cTmp As clsNode
'    tvFirst = 0  tvlast = 1 tvNext = 2 tvprevious = 3  tvChild = 4

If IsMissing(vRelative) Then

    Set NodeAdd = Me.AddRoot(sKey, vCaption, vImageMain, vImageExpanded)
    Exit Function
Else

    On Error Resume Next
    Set cRelative = vRelative
    If cRelative Is Nothing Then
        Set cRelative = mcolNodes(vRelative)
    End If

    On Error GoTo errH
    If cRelative Is Nothing Then
        Err.Raise vbObjectError + 100, "NodeAdd", "vRelative is not a valid node or a node.key"
    End If
End If

If IsMissing(vRelationship) Then
    vRelationship = tvTreeRelationship.tvNext      ' default
End If

If vRelationship = tvChild Or cRelative Is cRelative.Root Then
    Set cParent = cRelative
Else
    Set cParent = cRelative.ParentNode
End If

Set cNode = New clsNode

If Len(sKey) Then
100     mcolNodes.Add cNode, sKey
101
Else
    mcolNodes.Add cNode
End If

If cParent.ChildNodes Is Nothing Then
    Set cParent.ChildNodes = New Collection
End If

With cParent.ChildNodes

```

```

        If .Count = 0 Then
            .Add cNode
        Else
            i = 0
            If vRelationship = tvNext Or vRelationship = tvPrevious Then
                For Each cTmp In cParent.ChildNodes
                    i = i + 1
                    If cTmp Is cRelative Then
                        Exit For
                    End If
                Next
            End If
            Select Case vRelationship
            Case tvFirst: .Add cNode, , 1
            Case tvLast: .Add cNode, after:=.Count
            Case tvNext: .Add cNode, after:=i
            Case tvPrevious: .Add cNode, before:=i
            Case tvChild: .Add cNode
            End Select
        End If
    End With

    With cNode
        .Key = sKey
        .Caption = CStr(vCaption)
        .ImageMain = vImageMain
        .ImageExpanded = vImageExpanded
        .Index = mcolNodes.Count

        Set .ParentNode = cParent
        Set .Tree = Me
    End With

    Set cNode.Tree = Me      ' do this after let key = skey
    Set NodeAdd = cNode

    Exit Function
errH:
    If mcolNodes Is Nothing Then
        Set mcolNodes = New Collection
        Resume
    End If
    If Err = 100 And Err.Number = 457 Then
        Err.Raise vbObjectError + 1, "clsNode.AddChild", "Duplicate key: '" & sKey & "'"
    Else
        #If DebugMode = 1 Then
            Stop
            Resume
        #End If
        Err.Raise Err.Number, "clsNode.AddChild", Err.Description
    End If
End Function

Public Sub NodeRemove(cNode As clsNode)
' PT Remove a Node, its children and grandchildrem
'   remove all associated controls and tear down class objects
'   Call Refresh() when done removing nodes

    Dim lIdx As Long
    Dim lNodeCtrlsOrig As Long
    Dim cParent As clsNode
    Dim cNodeAbove As clsNode, cNd As clsNode

    On Error GoTo errH

    Set MoveCopyNode = Nothing

    Set cNodeAbove = NextVisibleNodeInTree(cNode, bUp:=True)
    Set cParent = cNode.ParentNode

    cNode.TerminateNode True

    cParent.RemoveChild cNode

    cNode.Index = -1      ' flag to get removed from mcolNodes in the loop
    If ActiveNode Is cNode Then
        Set moActiveNode = Nothing
    End If
    Set moEditNode = Nothing

```


clsTreeview - 12

```
lIdx = 0
lNodeCntlsOrig = mlNodesCreated
mlNodesCreated = 0

For Each cNd In mcolNodes
    lIdx = lIdx + 1
    If cNd.Index = -1 Then
        mcolNodes.Remove lIdx
        lIdx = lIdx - 1
    Else
        mlNodesCreated = mlNodesCreated - CLng(Not cNd.Control Is Nothing)
        cNd.Index = lIdx
    End If
Next

mlNodesDeleted = mlNodesDeleted + lNodeCntlsOrig - mlNodesCreated

Set cNode = Nothing ' should terminate the class

If mlNodesCreated Then
    If Not cNodeAbove Is Nothing Then
        Set Me.ActiveNode = cNodeAbove
    ElseIf mcolNodes.Count Then
        Set Me.ActiveNode = mcolNodes(1)
    End If
Else
    'all nodes deleted
    Erase mlVisOrder
    Erase msngMaxWidths
    mlVisCount = 0
    mlNodesCreated = 0
    mlNodesDeleted = 0
End If

Exit Sub

errH:
    #If DebugMode = 1 Then
        Debug.Print Err.Description, Err.Number
        Stop
        Resume
    #End If
End Sub

Public Sub NodesClear()
' PT, similar to Treeview.Nodes.Clear
Dim i As Long
On Error GoTo errH

If Not TreeControl Is Nothing Then
    With TreeControl
        For i = TreeControl.Controls.Count - 1 To 0 Step -1
            TreeControl.Controls.Remove i
        Next
        .ScrollBars = fmScrollBarsNone
    End With
End If

Erase mlVisOrder
Erase msngMaxWidths
mlVisCount = 0
mlNodesCreated = 0
mlNodesDeleted = 0

TerminateTree

mbChanged = False

Exit Sub

errH:
    #If DebugMode = 1 Then
        Stop
        Resume
    #End If
End Sub

Public Sub PopulateTree()
' PT add and displays all the controls for the Treeview for the first time

MsgBox "In beta-023 PopulateTree() was deprecated and merged with Refresh()" & vbNewLine & vbNewLine & _
    "Please replace ''PopulateTree'' with ''Refresh'' in your code", , AppName
```

Refresh

End Sub

```
Public Sub Refresh()
' Create node controls as required the first time respective parent's Expanded property = true
' hide or show and (re)position node controls as required
' Call Refresh after changing any Treeview properties or after adding/removing/moving any nodes
' or making any change that will alter placement of nodes in the treeview
    Dim bInit As Boolean

    If Me.TreeControl Is Nothing Then
        TerminateTree
        ' a Frame (container for the treeview) should have been referenced to me.TreeControl
        Err.Raise vbObjectError + 10, mcSource, "Refresh: 'TreeControl' frame is not referenced"

    ElseIf moRootHolder Is Nothing Then
        '
        Err.Raise vbObjectError + 11, mcSource, "Refresh: No Root nodes have been created"
    ElseIf moRootHolder.ChildNodes Is Nothing Then
        ' nothing to do
        mlVisCount = 0
        mlNodesCreated = 0
        mlNodesDeleted = 0
        Erase mlVisOrder
        Erase msgMaxWidths
        Exit Sub

    ElseIf Me.TreeControl.Controls.Count = 0 Then
        ' display the treeview for first time
        bInit = True
    Else
        ' ensure all node properties are checked, eg after changing indentation or nodeheight during runtime
        mbRedesign = True
    End If

    On Error GoTo errExit

    BuildRoot bInit

    Exit Sub
```

```
errExit:
    Err.Raise Err.Number, mcSource, "Error in BuildRoot: " & Err.Description
End Sub
```

```
Public Sub ScrollToView(Optional cNode As clsNode, _
                        Optional Top1Bottom2 As Long, _
                        Optional bCollapseOthers As Boolean)
' PT scrolls the treeview to position the node in view
' Top1Bottom2= 0 roughly 1/3 from the top
' Top1Bottom2= 1 or -1 at the top
' Top1Bottom2= 2 or -2 at the bottom
```

```
    Dim bIsVisible As Boolean
    Dim bWasCollapsed As Boolean
    Dim lVisIndex As Long
    Dim sngTop As Single
    Dim sngBot As Single
    Dim sngVisHt As Single
    Dim sngScrollTop As Single
    Dim cTmp As clsNode
```

```
    If cNode Is Nothing Then
        Set cNode = ActiveNode
    End If
```

```
    If bCollapseOthers Then
        SetTreeExpansionLevel 0
    End If
```

```
    Set cTmp = cNode.ParentNode
    While Not cTmp.Caption = "RootHolder"
        If Not cTmp.Expanded Then
            bWasCollapsed = True
            cTmp.Expanded = True
        End If
        Set cTmp = cTmp.ParentNode
    Wend
```

```

If bWasCollapsed Then
    BuildRoot False
End If

lVisIndex = cNode.VisIndex
sngBot = mcTLpad + lVisIndex * NodeHeight
sngTop = sngBot - NodeHeight

With TreeControl
    sngVisHt = .InsideHeight
    If .ScrollBars = fmScrollBarsBoth Or .ScrollBars = fmScrollBarsHorizontal Then
        sngVisHt = sngVisHt - 15 ' roughly(?) width of a scrollbar
    End If

    bIsVisible = sngTop > .ScrollTop And _
        sngBot < .ScrollTop + sngVisHt

    If Not bIsVisible Or Top1Bottom2 > 0 Then

        If Top1Bottom2 < 0 Then Top1Bottom2 = Top1Bottom2 * -1

        If Top1Bottom2 = 0 Then ' place about 1/3 from top
            sngScrollTop = lVisIndex * NodeHeight - .InsideHeight / 3

        ElseIf Top1Bottom2 = 1 Then ' scroll to top
            sngScrollTop = sngTop - mcTLpad
        Else
            sngScrollTop = sngBot - sngVisHt + mcTLpad ' scroll to bottom
        End If

        If sngScrollTop < 0 Then
            sngScrollTop = 0
        End If

        .ScrollTop = sngScrollTop
    End If
End With
End Sub

Public Sub TerminateTree()
'-----
' Procedure : TerminateTree
' Company   : JKP Application Development Services (c)
' Author    : Jan Karel Pieterse (www.jkp-ads.com)
' Created   : 15-01-2013
' Purpose   : Terminates this class' instance
'-----

Dim cNode As clsNode
'Instead of the terminate event of the class
'we use this public method so it can be
'explicitly called by parent classes
'this is done because we'll end up having multiple circular references
'between parent and child classes, which may cause the terminate events to be ignored.

If Not moRootHolder Is Nothing Then
    If Not moRootHolder.ChildNodes Is Nothing Then
        For Each cNode In moRootHolder.ChildNodes

            cNode.TerminateNode
        Next
    End If
    moRootHolder.TerminateNode
End If

Set moMoveNode = Nothing
Set moEditNode = Nothing
Set moActiveNode = Nothing
Set moRootHolder = Nothing
Set mcolNodes = Nothing

'** by design TerminateTree does NOT reset treeview properties or remove
'** the reference TreeControl reference to the treeview's Frame control
'
' If the form is being unloaded it's enough to call TerminateTree in it's close event, node controls will a
'utomatically unload with the form.
' However the treeview is to be cleared or moved but the main form is not being unloaded
' call the NodesRemove method which will remove all node controls then call TerminateTree
End Sub

```

```

'*****
'*      Friend properties, functions and subs
'*      although visible throughout the project these are only intended to be called by clsNodes
'*****

```

```

Friend Property Get EditMode(cNode As clsNode) As Boolean ' PT
    EditMode = mbEditMode
End Property

```

```

Friend Property Let EditMode(cNode As clsNode, ByVal bEditMode As Boolean) ' PT

```

```

    Set MoveCopyNode(False) = Nothing
    mbEditMode = bEditMode

```

```

    If Not moEditNode Is Nothing Then
        moEditNode.EditBox False
    End If

```

```

    If bEditMode Then
        Set moEditNode = cNode
    Else
        Set moEditNode = Nothing
    End If
End Property

```

```

Friend Function GetExpanderIcon(bExpanded As Boolean, pic As StdPicture) As Boolean

```

```

    If mbExpanderImage Then
        Set pic = moExpanderImage(bExpanded)
        GetExpanderIcon = True
    End If
End Function

```

```

Friend Function GetCheckboxIcon(lChecked As Long, pic As StdPicture) As Boolean

```

```

    If mbCheckboxImage Then
        Set pic = moCheckboxImage(lChecked)
        GetCheckboxIcon = True
    End If
End Function

```

```

Friend Function GetNodeIcon(vKey, pic As StdPicture, bFullWidth As Boolean) As Boolean

```

```

    On Error GoTo errExit
    Set pic = mcolIcons(vKey)
    bFullWidth = mbFullWidth
    GetNodeIcon = True

```

```

errExit:

```

```

End Function

```

```

Friend Function RaiseAfterLabelEdit(cNode As clsNode, sNewText As String) As Boolean

```

```

' PT called from moEditBox_KeyDown after vbKeyEnter
'

```

```

    Dim Cancel As Boolean
    RaiseEvent AfterLabelEdit(Cancel, sNewText, cNode)
    RaiseAfterLabelEdit = Cancel
End Function

```

```

Friend Sub NodeClick(ByRef oCtl As MSForms.Control, ByRef cNode As clsNode)

```

```

'-----
' Procedure : NodeClick
' Company   : JKP Application Development Services (c)
' Author    : Jan Karel Pieterse (www.jkp-ads.com)
' Created   : 15-01-2013
' Purpose   : Handles clicks on the treeview. Called from clsNode
'-----

```

```

' PT also called from checkbox (label) click event in clsNode

```

```

    Dim bFlag As Boolean
    Dim lngViewable As Long
    Dim cLastChild As clsNode

```

```

    If oCtl.Name Like "Exp*" Then
        bFlag = Not ActiveNode Is cNode
        If bFlag Then
            Set ActiveNode = cNode
        End If
    End If

```

```

    BuildRoot False

```

```

    If cNode.Expanded Then
        If Not cNode.ChildNodes Is Nothing Then
            Set cLastChild = cNode.ChildNodes(cNode.ChildNodes.Count)

```

```

        If Not NodeIsVisible(cLastChild, lngViewable) Then
            If lngViewable > cNode.ChildNodes.Count Then
                ScrollToView cLastChild, Top1Bottom2:=2
            Else
                ScrollToView cNode, Top1Bottom2:=1
            End If
        End If
    End If
End If
If bFlag Then
    RaiseEvent Click(cNode)
End If

ElseIf oCtl.Name Like "CheckBox*" Then    ' PT
    ' RaiseEvent for the checkbox moved to clsNode
    RaiseEvent NodeCheck(cNode)

ElseIf oCtl.Name Like "Node*" Then
    If Not ActiveNode Is cNode Then
        Set ActiveNode = cNode
    Else
        SetActiveNodeColor
    End If
    RaiseEvent Click(cNode)
End If

End Sub

Friend Function UniqueKey(sKey As String) As String
    Dim cNode As clsNode
    For Each cNode In Nodes
        If cNode.Key = sKey Then
            Err.Raise vbObjectError + 1, "clsTreeView", "Duplicate key: '" & sKey & "'"
        End If
    Next
    UniqueKey = sKey
End Function

Friend Sub UpdateScrollLeft()
' PT, moved all this from Let-Changed() in v025,
' called after manual node edit, update scrollLeft if/as necessary to show end of the new text
    Dim sngChangedRight As Single
    Dim sngIconPad As Single
    Dim pic As StdPicture
    Dim v

    If Not ActiveNode Is Nothing Then

        sngChangedRight = ActiveNode.Control.Left + ActiveNode.TextWidth + 15

        If mbFullWidth Then
            If ActiveNode.hasIcon(v) Then
                sngIconPad = mcIconPad
            End If
        End If

        If ActiveNode.TextWidth + sngIconPad > msngMaxWidths(ActiveNode.Level) Then
            msngMaxWidths(ActiveNode.Level) = ActiveNode.TextWidth + sngIconPad
        End If

        With Me.TreeControl

            If MaxNodeWidth > .InsideWidth Then

                If .ScrollBars > fmScrollBarsHorizontal Then
                    .ScrollBars = fmScrollBarsBoth
                Else
                    .ScrollBars = fmScrollBarsHorizontal
                End If

                .ScrollWidth = MaxNodeWidth + mcTLpad

                If .ScrollLeft + .InsideWidth < sngChangedRight Then
                    .ScrollLeft = sngChangedRight - .InsideWidth + mcTLpad
                End If

            End If
        End With
    End If
End Sub

```

End Sub

```

*****
'*      Private events      *
*****

```

```

Private Sub TreeControl_Click()
' PT exit editmode if an empty part of the treeview is clicked
    EditMode(ActiveNode) = False
End Sub

```

```

*****
'*      Private functions and subs      *
*****

```

```

Private Sub Class_Initialize()
' Set Root = New clsNode
' Set moRoot = New clsNode ' maybe(?) but keep Root() as read only

' set some defaults
    mbRootButton = True
    mbShowExpanders = True
    mbShowLines = True
    #If Mac Then
        msngIndent = 20
        msngNodeHeight = 16
    #Else
        msngIndent = 15
        msngNodeHeight = 12
    #End If
    msngRootLine = msngIndent
    msAppName = "TreeView"

    #If DebugMode = 1 Then
        gClsTreeViewInit = gClsTreeViewInit + 1      'for testing only
    #End If

```

End Sub

```

Private Sub Class_Terminate()
    #If DebugMode = 1 Then
        gClsTreeViewTerm = gClsTreeViewTerm + 1
    #End If
End Sub

```

```

Private Function AddNodeToCol(colNodes As Collection, cAddNode As clsNode, bTreeCol As Boolean, Optional vBefore,
Optional vAfter)
    Dim i As Long
    Dim sKey As String
    Dim cTmp As clsNode
    Dim pos As Long

    If bTreeCol Then sKey = cAddNode.Key

    If Len(sKey) Then
        On Error Resume Next
        i = 0
        Set cTmp = colNodes(sKey)
        If Not cTmp Is Nothing Then
            pos = InStr(1, sKey, "_copy:")
            If pos Then
                sKey = Left$(sKey, pos - 1)
            End If
            sKey = sKey & "_copy:"
            While Not cTmp Is Nothing
                Set cTmp = Nothing
                i = i + 1
                Set cTmp = colNodes(sKey & i)
            Wend
            sKey = sKey & i

            If bTreeCol Then
                cAddNode.Key = sKey
            End If
        End If

        On Error GoTo 0      ' error returns to caller

        If IsMissing(vBefore) And IsMissing(vAfter) Then

```

```

        colNodes.Add cAddNode, sKey
    ElseIf IsMissing(vAfter) Then
        colNodes.Add cAddNode, sKey, vBefore
    Else
        colNodes.Add cAddNode, sKey, , vAfter
    End If

Else ' no key
    If IsMissing(vBefore) And IsMissing(vAfter) Then
        colNodes.Add cAddNode
    ElseIf IsMissing(vAfter) Then
        colNodes.Add cAddNode, , vBefore
    Else
        colNodes.Add cAddNode, , , vAfter
    End If
End If
End Function

Private Sub BuildRoot(bInit As Boolean)
    Dim bCursorWait As Boolean
    Dim bTriStateOrig As Boolean
    Dim lLastRootVisIndex As Long
    Dim sngActiveNodeScrollTop As Single ' PT distance activenode was from scrolltop top before refresh, if vi
sible
    Dim sngChkBoxPad As Single
    Dim sngHeightAllNodes As Single
    Dim sngIconPad As Single
    Dim sngMaxWidth As Single
    Dim cRoot As clsNode
    Dim objCtrl As MSForms.Control
    Dim pt As POINTAPI
    Dim vIconKey

    Dim sCap As String
    Dim sngTickCnt As Single

    On Error GoTo locErr

    #If DebugMode Then
        #If Win32 Or Win64 Then
            sngTickCnt = getTickCount
        #Else ' Mac
            sngTickCnt = Timer
        #End If
    #End If

    bInit = TreeControl.Count = 0

    'TODO find equivalent for cancel key in Access & Word
    #If HostProject = "Access" Then
    #ElseIf HostProject = "Word" Then
    #Else
        Application.EnableCancelKey = xlErrorHandler
    #End If

    If mbAlwaysRedesign Then mbRedesign = True

    '    mcChkBoxSize = 10.5    ' 11.25
    '    mcLineLeft = 3 + 7.5    'msngIndent / 2

    ' PT if these arrays aren't large enough Redim Preserve is done in error handler
    ReDim mlVisOrder(1 To mlNodesCreated + 100)
    If bInit Or mbRedesign Then
        ReDim msngMaxWidths(0 To 7)
    End If

    If mcolNodes.Count - mlNodesCreated > 400 Then
        ' creating many controls might take a while
        #If HostProject = "Access" Then
            Application.DoCmd.Hourglass True
        #ElseIf HostProject = "Word" Then
            System.Cursor = wdCursorWait
        #Else
            Application.Cursor = xlWait
        #End If
        bCursorWait = True
    End If
    If Not bInit Then
        If NodeIsVisible Then
            sngActiveNodeScrollTop = (ActiveNode.VisIndex - 1) * NodeHeight - Me.TreeControl.ScrollTop

```

```

        End If
    End If

    mlVisCount = 0
    bTriStateOrig = mbTriState
    mbTriState = False

    If CheckBoxes Then
        If mbCheckboxImage Then
            sngChkBoxPad = mcCheckboxPadImg
        Else
            sngChkBoxPad = mcCheckboxPad
        End If
        If mcChkBoxSize > msngNodeHeight Then
            msngNodeHeight = mcChkBoxSize
        End If
    End If

    ' work out respective offsets to various node controls from node tops
    msngTopExpB = mcTLpad + (msngNodeHeight - mcExpButSize) / 2 + 1.5
    If mbExpanderImage Then
        msngTopExpT = mcTLpad + (msngNodeHeight - (mcExpButSize - 4)) / 2
    Else
        msngTopExpT = mcTLpad + (msngNodeHeight - mcExpButSize) / 2
    End If

    msngTopChk = mcTLpad + (msngNodeHeight - mcChkBoxSize) / 2
    msngTopIcon = mcTLpad + (msngNodeHeight - mcIconSize) / 2
    msngTopHV = mcTLpad + msngNodeHeight / 2
    Call Round75

    With TreeControl
        mlBackColor = .BackColor      ' default colours for node labels
        mlForeColor = .ForeColor

        If bInit Then
            .SpecialEffect = 2        ' fmSpecialEffectSunken
        Else
            ' PT, refresh, start by hiding all the controls
            For Each objCtrl In .Controls
                objCtrl.Visible = False
            Next
        End If

        For Each cRoot In moRootHolder.ChildNodes
            sngIconPad = 0
            If mbFullWidth Then
                If mbGotIcons And cRoot.hasIcon(vIconKey) Then
                    sngIconPad = mcIconPad
                End If
            End If

            If cRoot.Control Is Nothing Then
                mlNodesCreated = mlNodesCreated + 1
                'Add the rootnode to the tree
                Set cRoot.Control = TreeControl.Controls.Add("Forms.label.1", "Node" & mlNodesDeleted + mlNodesCr
eated, False)

                With cRoot.Control

                    If Not mbFullWidth And mbGotIcons Then
                        If cRoot.hasIcon(vIconKey) Then
                            .PicturePosition = fmPicturePositionLeftCenter
                            .Picture = mcolIcons(vIconKey)
                        End If
                    End If

                    .Top = mcTLpad + mlVisCount * msngNodeHeight
                    .Left = mcTLpad + msngRootLine + sngIconPad + msngChkBoxPad

                    If cRoot.BackColor Then
                        .BackColor = cRoot.BackColor
                    End If
                    If cRoot.ForeColor Then
                        .ForeColor = cRoot.ForeColor
                    End If

                    If cRoot.Bold Then .Font.Bold = True
                    .Caption = cRoot.Caption
                End With
            End If
        Next
    End With

```



```

.AutoSize = True
.WordWrap = False

cRoot.TextWidth = .Width

If .Width + sngIconPad > msngMaxWidths(0) Then
    msngMaxWidths(0) = .Width + sngIconPad
End If

' calc msngTopLabel to align node label to mid NodeHeight
' first calc min NodeHeight if not set higher by user
If .Height > msngNodeHeight Then
    ' optimal NodeHeight for the current font
    msngNodeHeight = .Height ' 'don't use the Property method or Refresh will be called
ElseIf .Height < msngNodeHeight Then
    #If Mac Then
        msngTopLabel = Int(msngNodeHeight - .Height) / 2
    #Else
        msngTopLabel = Int((msngNodeHeight - .Height + mcPtPx1) / 3 * 2) * mcPtPx1
    #End If
    .Top = mcTlpad + msngTopLabel + mlVisCount * msngNodeHeight
End If

If mbFullWidth Then
    If msngTopLabel < mcFullWidth Then
        .Width = mcFullWidth
        .AutoSize = False
    End If
End If

If Len(cRoot.ControlTipText) Then
    .ControlTipText = cRoot.ControlTipText
End If

.WordWrap = False
.ZOrder 0
.Visible = True

End With
Else

    With cRoot.Control

        If mbRedesign Then
            .Left = mcTlpad + msngRootLine + sngIconPad + msngChkBoxPad

            If cRoot.TextWidth + sngIconPad > msngMaxWidths(0) Then
                msngMaxWidths(0) = cRoot.TextWidth + sngIconPad
            End If
        End If

        If .Height > msngNodeHeight Then
            msngNodeHeight = .Height
        ElseIf .Height < msngNodeHeight Then
            #If Mac Then
                msngTopLabel = Int(msngNodeHeight - .Height) / 2
            #Else
                msngTopLabel = Int((msngNodeHeight - .Height + mcPtPx1) / 3 * 2) * mcPtPx1
            #End If
        End If

        .Top = mcTlpad + msngTopLabel + mlVisCount * msngNodeHeight

        .Visible = True

    End With
End If

' horizontal line
If mbRootButton And mbShowLines Then
    If cRoot.HLine Is Nothing Then
        Set cRoot.HLine = TreeControl.Controls.Add("Forms.label.1", "HLine" & cRoot.Control.Name, Fal
se)

        With cRoot.HLine
            .Top = msngTopHV + mlVisCount * msngNodeHeight
            .Left = mcLineLeft
            .Caption = ""
            .BorderStyle = fmBorderStyleSingle
            .BorderColor = vbScrollBars
            .Width = msngIndent

```

```

        .Height = mcPtPx1
        .TextAlign = fmTextAlignCenter
        .BackStyle = fmBackStyleTransparent
        .ZOrder 1
        .Visible = True
    End With
Else
    With cRoot.HLine
        .Width = msngIndent
        .Top = msngTopHV + mlVisCount * msngNodeHeight ' 3 + NodeHeight/2 (to nearest 0.75)
        .Visible = True
    End With
End If
End If

' Checkbox
If Checkboxes Then
    If cRoot.Checkbox Is Nothing Then
        Set cRoot.Checkbox = TreeControl.Controls.Add("Forms.label.1", "CheckBox" & cRoot.Control.Nam

    With cRoot.Checkbox
        .Left = mcTLpad + msngRootLine
        .Top = msngTopChk + mlVisCount * msngNodeHeight

        If mbCheckboxImage Then
            'Use an image
            .BorderStyle = fmBorderStyleNone
            .Picture = mcCheckboxImage(cRoot.Checked)
            .PicturePosition = fmPicturePositionLeftTop
            .AutoSize = True
            '.Width = 7.5
            '.Height = 7.5
        Else
            .Width = mcChkBoxSize
            .Height = mcChkBoxSize
            .Font.Name = "Marlett" ' "a" is a tick
            .FontSize = mcCheckboxFont '9
            .BorderStyle = fmBorderStyleSingle
            .BackColor = vbWindowBackground
            .ForeColor = vbWindowText
        End If
    End With
End If

' If cRoot.Checked Then cRoot.Checked = True
' .Visible = True
End With
Else
    With cRoot.Checkbox
        .Left = mcTLpad + msngRootLine
        .Top = msngTopChk + mlVisCount * msngNodeHeight
        .Visible = True
    End With
End If
End If

' Icon
If mbFullWidth And mbGotIcons Then
    If cRoot.hasIcon(vIconKey) Then
        If cRoot.Icon Is Nothing Then
            Set cRoot.Icon = TreeControl.Controls.Add("Forms.Image.1", "Icon" & cRoot.Control.Name, F

        With cRoot.Icon
            .BackStyle = fmBackStyleTransparent
            .BorderStyle = fmBorderStyleNone
            '.AutoSize
            .Width = mcIconSize
            .Height = mcIconSize
            .Left = mcTLpad + msngRootLine + msngChkBoxPad
            .Top = msngTopIcon + mlVisCount * msngNodeHeight
            .Picture = mcolIcons(vIconKey)
            .BackStyle = fmBackStyleTransparent
            .Visible = True
        End With
    End If
End If

```

```

        End With
    Else
        With cRoot.Icon
            .Left = mcTLpad + msngRootLine + msngChkBoxPad
            .Top = msngTopIcon + mlVisCount * msngNodeHeight
            .Visible = True
        End With
    End If
Else
    sngIconPad = 0
End If
End If

mlVisCount = mlVisCount + 1
mlVisOrder(mlVisCount) = cRoot.Index
cRoot.VisIndex = mlVisCount

lLastRootVisIndex = mlVisCount

'Now add this root's children
If Not cRoot.ChildNodes Is Nothing Then
    BuildTree cRoot, 1, True
End If

Next

'Vertical line for multiple roots
If mbRootButton And mbShowLines Then
    If moRootHolder.ChildNodes.Count > 1 Then

        If moRootHolder.VLine Is Nothing Then
            Set moRootHolder.VLine = TreeControl.Controls.Add("forms.label.1", "VLine_Roots")
            With moRootHolder.VLine
                .ZOrder 1
                .Width = mcPtPx1
                .Caption = ""
                .BorderColor = vbScrollBars
                .BorderStyle = fmBorderStyleSingle
                .Top = msngTopHV
                .Left = mcLineLeft
                .Height = (lLastRootVisIndex - 1) * msngNodeHeight
            End With

        Else

            With moRootHolder.VLine
                .Top = msngTopHV
                .Height = (lLastRootVisIndex - 1) * msngNodeHeight
                .Visible = True
            End With
        End If

    End If
End If

sngHeightAllNodes = mlVisCount * NodeHeight + (mcTLpad * 2)    ' mcTLpad for top/bottom padding
If bInit Then
    .ScrollHeight = 0
    .ScrollLeft = 0
End If

sngMaxWidth = MaxNodeWidth

If sngHeightAllNodes > .InsideHeight Then
    If sngMaxWidth + 15 > .InsideWidth Then
        .ScrollBars = fmScrollBarsBoth
        .ScrollWidth = sngMaxWidth + mcTLpad
    Else
        .ScrollBars = fmScrollBarsVertical
        .ScrollLeft = 0
        .ScrollWidth = 0
    End If
    .ScrollHeight = sngHeightAllNodes
Else
    If sngMaxWidth > .InsideWidth + IIf(.ScrollBars > 1, 15, 0) Then
        .ScrollBars = fmScrollBarsHorizontal
        .ScrollWidth = sngMaxWidth + mcTLpad
    Else
        .ScrollBars = fmScrollBarsNone
        .ScrollLeft = 0
    End If
End If

```

```

        .ScrollWidth = 0
    End If

    .ScrollTop = 0
    .ScrollHeight = 0
End If

If bInit Then ' startup
    ' make the first root node active but don't highlight it
    Set moActiveNode = moRootHolder.ChildNodes(1)
    ' or if preferred highlighted at startup
    'Set ActiveNode = moRootHolder.ChildNodes(1)
ElseIf Not ActiveNode Is Nothing Then
    If Not NodeIsVisible Then
        .ScrollTop = (ActiveNode.VisIndex - 1) * NodeHeight - sngActiveNodeScrollTop
    End If
End If

End With

#If DebugMode Then
#If Win32 Or Win64 Then
    sngTickCnt = (getTickCount - sngTickCnt) / 1000
#Else ' if Mac
    sngTickCnt = Timer - sngTickCnt
#End If

sCap = "Seconds: " & Format(sngTickCnt, "0.00") & _
    " Nodes: " & mcolNodes.Count & _
    " created: " & mlNodesCreated & _
    " visible: " & mlVisCount & _
    " Total controls: " & TreeControl.Controls.Count

#If HostProject = "Access" Then
    If Not moForm Is Nothing Then
        moForm.Caption = sCap
    End If
#Else
    Me.TreeControl.Parent.Caption = sCap
#End If
#End If

mbRedesign = False
mbTriState = bTriStateOrig
done:

If bCursorWait Then

    #If HostProject = "Access" Then
        Application.DoCmd.Hourglass False
    #ElseIf HostProject = "Word" Then
        System.Cursor = wdCursorNormal
    #Else
        Application.Cursor = xlDefault
    #End If

    #If Win32 Or Win64 Then
        ' in some systems the cursor fails to reset to default, this forces it
        GetCursorPos pt
        SetCursorPos pt.X, pt.Y
    #End If
End If

'TODO: implement API equivalent for cancel key in Access & Word
#If HostProject = "Access" Then
#ElseIf HostProject = "Word" Then
#Else
    Application.EnableCancelKey = xlInterrupt
#End If

Exit Sub

locErr:
mbRedesign = False
mbTriState = bTriStateOrig

If Err.Number = 9 And (mlVisCount = UBound(mlVisOrder) + 1) Then
    ' most likely an array needs enlarging
    If mlVisCount = UBound(mlVisOrder) + 1 Then
        ReDim Preserve mlVisOrder(LBound(mlVisOrder) To mlVisCount + 100)
    End If
End If

```

```

        Resume
    End If
ElseIf Err.Number = 18 Then
    ' user pressed ctrl-break
    MsgBox "Loading/refreshing Treeview aborted", , AppName
    NodesClear
    Resume done
End If

#If DebugMode = 1 Then
    Debug.Print Err.Number, Err.Description
    Stop
    Resume
#End If

Err.Raise Err.Number, "BuildRoot", Err.Description
End Sub

Private Sub BuildTree(cNode As clsNode, ByVal lLevel As Long, Optional lMaxLevel As Long = -1)
    Dim cChild As clsNode
    Dim lVLineTopIdx As Long

    ' On Error GoTo locErr

    If (lLevel > 1 Or mbRootButton) And mbShowExpanders Then

        'Expand/collapse button box (not needed if we use icons are used for expanders)
        If Not mbExpanderImage Then
            If cNode.ExpanderBox Is Nothing Then
                Set cNode.ExpanderBox = TreeControl.Controls.Add("Forms.label.1", "ExpBox" & cNode.Control.Name,
False)

                With cNode.ExpanderBox
                    .Top = (mlVisCount - 1) * NodeHeight + msngTopExpB
                    .Left = mcTLpad * 2 + (lLevel - 2) * msngIndent + msngRootLine
                    .Width = mcExpBoxSize
                    .Height = mcExpBoxSize
                    .BorderStyle = fmBorderStyleSingle
                    .BorderColor = vbScrollBars
                    .BackStyle = fmBackStyleOpaque
                    .Visible = True
                End With
            Else
                With cNode.ExpanderBox
                    If mbRedesign Then .Left = mcTLpad * 2 + (lLevel - 2) * msngIndent + msngRootLine
                    .Top = (mlVisCount - 1) * NodeHeight + msngTopExpB
                    .Visible = True
                End With
            End If
        End If

        'Expand/collapse button text (or icon)
        If cNode.Expander Is Nothing Then
            Set cNode.Expander = TreeControl.Controls.Add("Forms.label.1", "ExpText" & cNode.Control.Name, False)
            With cNode.Expander
                .Left = mcTLpad * 2 + (lLevel - 2) * msngIndent + msngRootLine
                .Top = (mlVisCount - 1) * NodeHeight + msngTopExpT

                If mbExpanderImage Then
                    'Use an image
                    .AutoSize = True
                    .Width = 7.5
                    .Height = 7.5
                    .BorderStyle = fmBorderStyleNone
                    .PicturePosition = fmPicturePositionLeftTop
                    .Picture = moExpanderImage(cNode.Expanded)
                    #If Mac Then
                        .BackStyle = fmBackStyleTransparent
                    #End If
                Else
                    'use +/- text
                    .Width = mcExpButSize
                    .Height = mcExpButSize

                    If cNode.Expanded = True Then
                        .Caption = "-"
                        .Font.Bold = True
                    Else
                        .Caption = "+"
                        .Font.Bold = False
                    End If
                End If
            End With
        End If
    End If
End Sub

```

```

        .Font.Size = mcExpanderFont
        .TextAlign = fmTextAlignCenter
        .BackStyle = fmBackStyleTransparent
    End If
    .Visible = True
End With
Else
    With cNode.Expander
        If mbRedesign Then .Left = mcTLpad * 2 + (lLevel - 2) * msngIndent + msngRootLine
        .Top = (mlVisCount - 1) * NodeHeight + msngTopExpT
        .Visible = True
    End With
End If

End If ' lLevel > 1 Or mbRootButton) And mbShowExpanders

If cNode.Expanded And (lMaxLevel < lLevel Or lMaxLevel = -1) Then

    'Vertical line
    If mbShowLines Then
        If cNode.VLine Is Nothing Then
            Set cNode.VLine = TreeControl.Controls.Add("Forms.label.1", "VLine" & cNode.Control.Name, False)
            lVLineTopIdx = mlVisCount
            With cNode.VLine
                .ZOrder 1
                .Top = msngTopHV + (lVLineTopIdx - 1) * NodeHeight
                .Left = mcLineLeft + msngRootLine + msngIndent * (lLevel - 1)
                .Width = mcPtPx1
                .Height = NodeHeight
                .Caption = ""
                .BorderColor = vbScrollBars
                .BorderStyle = fmBorderStyleSingle
                .Visible = True
            End With

        Else
            lVLineTopIdx = mlVisCount
            With cNode.VLine
                .Top = msngTopHV + (lVLineTopIdx - 1) * NodeHeight
                If mbRedesign Then
                    .Left = mcLineLeft + msngRootLine + msngIndent * (lLevel - 1)
                    .Visible = True
                End If
            End With
        End If
    End If

    For Each cChild In cNode.ChildNodes

        ' extend the vertical line
        If mbShowLines Then
            With cNode.VLine
                .Height = (mlVisCount - lVLineTopIdx + 1) * msngNodeHeight
                .Visible = True
            End With
        End If

        BuildNodeControls cChild, lLevel

        If Not cChild.ChildNodes Is Nothing Then
            BuildTree cChild, lLevel + 1
        End If

    Next

End If ' cNode.Expanded And (lMaxLevel < lLevel Or lMaxLevel = -1)

Exit Sub

'locErr:
'    #If DebugMode = 1 Then
'        Stop
'        Resume
'    #End If
End Sub

Private Sub BuildNodeControls(cNode As clsNode, ByVal lLevel As Long)
' PT, create or (un)hide the controls, size & position to suit
' all created nodes have a caption, and optionally a horizontal line, checkbox and separate icon

```

```

Dim sngIconPad As Single
Dim sName As String
Dim vKey

```

```

On Error GoTo locErr

```

```

' Application.EnableCancelKey = xlErrorHandler

```

```

If cNode.Control Is Nothing Then
    mlNodesCreated = mlNodesCreated + 1
    sName = "Node" & mlNodesDeleted + mlNodesCreated
ElseIf mbRedesign Then
    sName = cNode.Control.Name
End If

```

```

'Horizontal line

```

```

If mbShowLines Then
    If cNode.HLine Is Nothing Then
        Set cNode.HLine = TreeControl.Controls.Add("Forms.label.1", "HLine" & sName, False)
        With cNode.HLine
            .Left = mcLineLeft + msngRootLine + msngIndent * (lLevel - 1)
            .Top = msngTopHV + mlVisCount * NodeHeight
            .Width = msngIndent
            .Height = mcPtPxl
            .Caption = ""
            .BorderStyle = fmBorderStyleSingle
            .BorderColor = vbScrollBars
            If mbRedesign Then
                .ZOrder 1
            End If
            .Visible = True
        End With
    Else
        With cNode.HLine
            If mbRedesign Then
                .Left = mcLineLeft + msngRootLine + msngIndent * (lLevel - 1)
                .Width = msngIndent
            End If
            .Top = msngTopHV + mlVisCount * NodeHeight
            .Visible = True
        End With
    End If
End If

```

```

' Checkbox

```

```

If CheckBoxes Then
    If cNode.Checkbox Is Nothing Then
        Set cNode.Checkbox = TreeControl.Controls.Add("Forms.label.1", "CheckBox" & sName, False)
        With cNode.Checkbox
            .Left = mcTLpad + msngRootLine + msngIndent * lLevel
            .Top = mlVisCount * NodeHeight + msngTopChk

            If mbCheckboxImage Then
                'Use an image
                .BorderStyle = fmBorderStyleNone
                .Picture = moCheckboxImage(cNode.Checked)
                .PicturePosition = fmPicturePositionLeftBottom
                .AutoSize = True
            Else

                .Width = mcChkBoxSize
                .Height = mcChkBoxSize
                .Font.Name = "Marlett"
                .Font.Size = 10
                .TextAlign = fmTextAlignCenter
                .BorderStyle = fmBorderStyleSingle
                If cNode.Checked Then
                    .Caption = "a"
                    If cNode.Checked = 1 Then
                        .ForeColor = RGB(180, 180, 180)
                    End If
                End If
            End If

            .Visible = True
        End With
    Else
        With cNode.Checkbox
            If mbRedesign Then .Left = mcTLpad + msngRootLine + msngIndent * lLevel

```

```

        .Top = mlVisCount * NodeHeight + msngTopChk
        .Visible = True
    End With
End If
End If

' Icon, in its own image control if using FullWidth, otherwise it goes in the label
If mbFullWidth And mbGotIcons Then
    If cNode.hasIcon(vKey) Then
        sngIconPad = mcIconPad
        If cNode.Icon Is Nothing Then
            Set cNode.Icon = TreeControl.Controls.Add("Forms.Image.1", "Icon" & sName, False)
            With cNode.Icon
                .BorderStyle = fmBorderStyleNone
                .Left = mcTLpad + msngRootLine + msngIndent * lLevel + msngChkBoxPad
                .Top = mlVisCount * NodeHeight + msngTopIcon
                .AutoSize
                .Width = mcIconSize
                .Height = mcIconSize
                .BackStyle = fmBackStyleTransparent
                .Picture = mcolIcons(vKey)
                .BackStyle = fmBackStyleTransparent
                .Visible = True
            End With
        Else
            With cNode.Icon
                If mbRedesign Then
                    .Left = mcTLpad + msngRootLine + msngIndent * lLevel + msngChkBoxPad
                End If
                .Top = mlVisCount * NodeHeight + msngTopIcon
                .Visible = True
            End With
        End If
    Else
        sngIconPad = 0
    End If
End If

'The node itself
If cNode.Control Is Nothing Then

    Set cNode.Control = TreeControl.Controls.Add("Forms.label.1", sName, False)
    With cNode.Control
        .WordWrap = False
        .AutoSize = True
        .Left = mcTLpad + msngRootLine + msngIndent * lLevel + msngChkBoxPad + sngIconPad
        .Top = mcTLpad + msngTopLabel + mlVisCount * NodeHeight

        If Not mbFullWidth And mbGotIcons Then
            If cNode.hasIcon(vKey) Then
                .PicturePosition = fmPicturePositionLeftCenter
                .Picture = mcolIcons(vKey)
            End If
        End If

        If cNode.Bold Then .Font.Bold = True
        .WordWrap = False
        .AutoSize = True
        .Caption = cNode.Caption
        cNode.TextWidth = .Width

        If cNode.TextWidth + sngIconPad > msngMaxWidths(lLevel) Then
            msngMaxWidths(lLevel) = cNode.TextWidth + sngIconPad
        End If

        If mbFullWidth Then
            .AutoSize = False
            If .Width <= mcFullWidth Then .Width = mcFullWidth
        End If
        If cNode.BackColor Then
            .BackColor = cNode.BackColor
        End If
        If cNode.ForeColor Then
            .ForeColor = cNode.ForeColor
        End If

        If Len(cNode.ControlTipText) Then
            .ControlTipText = cNode.ControlTipText
        End If
    End With
End If

```



```

        .Visible = True
    End With

Else
    With cNode.Control
        If mbRedesign Then
            .Left = mcTLpad + msngRootLine + msngIndent * lLevel + sngIconPad + msngChkBoxPad

            If cNode.TextWidth + sngIconPad > msngMaxWidths(lLevel) Then
                msngMaxWidths(lLevel) = cNode.TextWidth + sngIconPad
            End If
        End If

        .Top = mlVisCount * NodeHeight + mcTLpad + msngTopLabel
        .Visible = True
    End With

End If

mlVisCount = mlVisCount + 1
mlVisOrder(mlVisCount) = cNode.Index
cNode.VisIndex = mlVisCount

Exit Sub

locErr:
If Err.Number = 9 Then
    ' most likely an array needs enlarging
    If mlVisCount = UBound(mlVisOrder) + 1 Then
        ReDim Preserve mlVisOrder(LBound(mlVisOrder) To mlVisCount + 100)
        Resume
    ElseIf lLevel > UBound(msngMaxWidths) Then
        ReDim Preserve msngMaxWidths(LBound(msngMaxWidths) To lLevel + 5)
        Resume
    End If
ElseIf Err.Number = 18 Then
    Err.Raise 18 ' user pressed ctrl-break, pass to BuildRoot
Else
    #If DebugMode = 1 Then
        Stop
        Resume
    #End If
    Err.Raise Err.Number, "BuildNodeControls", Err.Description
End If

End Sub

Private Sub Clone(cParent As clsNode, cNode As clsNode, Optional vBefore, Optional ByVal vAfter)
' PT clone a node and add the 4-way references
    Dim bTriStateOrig As Boolean
    Dim cClone As clsNode
    Dim cChild As clsNode

    On Error GoTo errH

    If cParent Is Nothing Or cNode Is Nothing Then
        Exit Sub
    End If

    bTriStateOrig = mbTriState
    mbTriState = False

    Set cClone = New clsNode

    With cNode
        If .BackColor = 0 Then .BackColor = mlBackColor
        cClone.BackColor = .BackColor
        cClone.Caption = .Caption
        cClone.Checked = .Checked
        cClone.Expanded = .Expanded
        If .ForeColor = 0 Then .ForeColor = mlForeColor
        cClone.ImageExpanded = .ImageExpanded
        cClone.ImageMain = .ImageMain
        cClone.ForeColor = .ForeColor
        cClone.Key = .Key
    End With

    If cParent.ChildNodes Is Nothing Then
        Set cParent.ChildNodes = New Collection
    End If

```

```

Set cClone.ParentNode = cParent

If Not cNode.ChildNodes Is Nothing Then
    For Each cChild In cNode.ChildNodes
        Clone cClone, cChild      ' don't pass vBefore/vAfter
    Next
End If

AddNodeToCol cParent.ChildNodes, cClone, False, vBefore, vAfter

Set cClone.Tree = Me

AddNodeToCol mcolNodes, cClone, bTreeCol:=True

cClone.Index = Nodes.Count

mbTriState = bTriStateOrig
If mbTriState Then
    cClone.ParentNode.CheckTriStateParent
End If

Exit Sub

errH:
    #If DebugMode = 1 Then
        Debug.Print Err.Description
        Stop
        Resume
    #End If
    mbTriState = bTriStateOrig
End Sub

Private Function MaxNodeWidth() As Single
'-----
' Procedure : MaxNodeWidth
' Author    : Peter Thornton
' Created   : 27-01-2013
' Purpose   : Get the max right for horizontal scroll
'-----

    Dim lLevel As Long
    Dim sngMax As Single

    ''' msngMaxWidths(), contains maximum text-width + additional icon width (if any) in each level
    ' tot-width = 3 + msngRootLine + msngIndent * lLevel + msngChkBoxPad + [ msngIconPad + text-width]

    For lLevel = 0 To UBound(msngMaxWidths)
        If msngMaxWidths(lLevel) Then
            If mcTLpad + msngRootLine + msngIndent * lLevel + msngChkBoxPad + msngMaxWidths(lLevel) > sngMax Then
                sngMax = mcTLpad + msngRootLine + msngIndent * lLevel + msngChkBoxPad + msngMaxWidths(lLevel)
            End If
        End If
    Next
    MaxNodeWidth = sngMax
End Function

Private Function NextVisibleNodeInTree(ByRef cStartNode As clsNode, Optional bUp As Boolean = True) As clsNode
'-----
' Procedure : NextVisibleNodeInTree
' Company   : JKP Application Development Services (c)
' Author    : Jan Karel Pieterse (www.jkp-ads.com)
' Created   : 16-01-2013
' Purpose   : Function that returns either the next or the previous node adjacent to the active node
'-----

    Dim lStep As Long
    Dim lNextVis As Long      'PT

    On Error GoTo errH
    If bUp Then lStep = -1 Else lStep = 1

    If cStartNode Is Nothing Then
        Set NextVisibleNodeInTree = mcolNodes(1)
    Else

        lNextVis = cStartNode.VisIndex
        lNextVis = lNextVis + lStep
        If lNextVis >= 1 And lNextVis <= mlVisCount Then
            lNextVis = mlVisOrder(lNextVis)
        End If
    End If
End Function

```

```

        Set NextVisibleNodeInTree = mcolNodes(lNextVis)
    End If
End If
Exit Function
errH:
#If DebugMode = 1 Then
    Stop
    Debug.Print Err.Description
    Resume
#End If
End Function

Private Function NodeIsVisible(Optional cNode As clsNode, Optional lngCntVisible As Long) As Boolean
Dim idxFirstVis As Long
Dim idxLastVis As Long

    If TreeControl Is Nothing Then
        Exit Function
    End If

    With TreeControl
        idxFirstVis = .ScrollTop / NodeHeight + 1
        lngCntVisible = (.InsideHeight - mcTLpad) / NodeHeight
        idxLastVis = lngCntVisible + idxFirstVis - 1
    End With

    If cNode Is Nothing Then
        If Not ActiveNode Is Nothing Then

            Set cNode = ActiveNode
        Else
            Exit Function
        End If
    End If

    If idxLastVis > mlVisCount Then idxLastVis = mlVisCount

    If Not cNode Is Nothing Then
        NodeIsVisible = cNode.VisIndex >= idxFirstVis And cNode.VisIndex <= idxLastVis
    End If

End Function

Private Sub ResetActiveNodeColor(cNode As clsNode)
Dim lBColor As Long
Dim lFColor As Long
If Not cNode Is Nothing Then
    lBColor = cNode.BackColor
    lFColor = cNode.ForeColor
    With cNode.Control
        .BorderStyle = fmBorderStyleNone
        .BackColor = IIf(lBColor, lBColor, mlBackColor)
        .ForeColor = IIf(lFColor, lFColor, mlForeColor)
    End With
End If
End Sub

Private Sub Round75()
'-----
' Procedure : Round75
' Author    : Peter Thornton
' Created   : 29-01-2013
' Purpose    : Make size & position dims a factor of 0.75 points (units of 1 pixel)
'-----
#If Mac Then
    msgngTopExpB = Int(msgngTopExpB)
    msgngTopExpT = Int(msgngTopExpT)
    msgngTopHV = Int(msgngTopHV)
    msgngTopIcon = Int(msgngTopIcon)
    msgngTopChk = Int(msgngTopChk)
    msgngTopLabel = Int(msgngTopLabel)
#Else
    msgngTopExpB = Int((msgngTopExpB * 2 + mcPtPx1) / 3 * 2) * mcPtPx1
    msgngTopExpT = Int((msgngTopExpT * 2 + mcPtPx1) / 3 * 2) * mcPtPx1
    msgngTopHV = Int((msgngTopHV * 2 + mcPtPx1) / 3 * 2) * mcPtPx1
    msgngTopIcon = Int((msgngTopIcon * 2 + mcPtPx1) / 3 * 2) * mcPtPx1
    msgngTopChk = Int((msgngTopChk * 2 + mcPtPx1) / 3 * 2) * mcPtPx1
    msgngTopLabel = Int((msgngTopLabel * 2 + mcPtPx1) / 3 * 2) * mcPtPx1
#End If
End Sub

```

```
Private Sub SetActiveNodeColor(Optional bInactive)
```

```
    If Not ActiveNode Is Nothing Then
```

```
        If IsMissing(bInactive) Then
```

```
            On Error Resume Next
```

```
            #If HostProject = "Access" Then
```

```
                bInactive = mbInactive
```

```
            #Else
```

```
                bInactive = Not Me.TreeControl Is Me.TreeControl.Parent.ActiveControl
```

```
            #End If
```

```
            On Error GoTo 0
```

```
        End If
```

```
        ' system highlight colours, bInactive set and called from EnterExit event
```

```
        With ActiveNode.Control
```

```
            If bInactive Then
```

```
                ''' when treeview not in focus
```

```
                    ResetActiveNodeColor moActiveNode
```

```
                    ''' just a grey border
```

```
                    .BorderStyle = fmBorderStyleSingle
```

```
                    .BorderColor = RGB(190, 190, 190)
```

```
                    ''' inactive colours
```

```
                    .BackColor = vbInactiveTitleBar
```

```
                    .ForeColor = vbWindowText
```

```
            Else
```

```
                ' in focus
```

```
                .BorderStyle = fmBorderStyleNone
```

```
                .BackColor = vbHighlight
```

```
                .ForeColor = vbHighlightText
```

```
            End If
```

```
        End With
```

```
    End If
```

```
End Sub
```

```
Private Sub SetTreeExpansionLevel(lLevel As Long, Optional lCurLevel As Long, _  
                                Optional cNode As clsNode, Optional bExit As Boolean = False)
```

```
-----  
' Procedure : SetTreeExpansionLevel  
' Company   : JKP Application Development Services (c)  
' Author    : Jan Karel Pieterse (www.jkp-ads.com)  
' Created   : 17-01-2013  
' Purpose   : Updates the expanded properties according to lLevel  
'           : Called recursively.  
-----
```

```
    Dim cChild As clsNode
```

```
    If bExit Then Exit Sub
```

```
    If cNode Is Nothing Then
```

```
        For Each cNode In moRootHolder.ChildNodes
```

```
            If lLevel > -1 Then
```

```
                cNode.Expanded = True
```

```
            Else
```

```
                cNode.Expanded = False
```

```
            End If
```

```
            If Not cNode.ChildNodes Is Nothing Then
```

```
                For Each cChild In cNode.ChildNodes
```

```
                    cChild.Expanded = (lLevel > lCurLevel)
```

```
                    SetTreeExpansionLevel lLevel, lCurLevel + 1, cChild, False
```

```
                Next
```

```
            End If
```

```
        Next
```

```
    ElseIf Not cNode.ChildNodes Is Nothing Then
```

```
        For Each cChild In cNode.ChildNodes
```

```
            cChild.Expanded = (lLevel > lCurLevel)
```

```
            SetTreeExpansionLevel lLevel, lCurLevel + 1, cChild, False
```

```
        Next
```

```
    End If
```

```
End Sub
```

```
Private Sub TreeControl_KeyDown(ByVal KeyCode As MSForms.ReturnInteger, ByVal Shift As Integer)
```

```
    Dim sngVisTop As Single
```

```

Dim cNode As clsNode

' PT toggle expand/collapse with key Enter
If KeyCode = vbKeyReturn Then
    If ActiveNode.Expanded Then
        KeyCode = vbKeyLeft
    Else
        KeyCode = vbKeyRight
    End If
End If

If Not ActiveNode Is Nothing Then
    Select Case KeyCode

    Case vbKeyLeft
        If ActiveNode.Level = 0 And Not mbRootButton Then
            ' don't attempt to collapse the Root if it doesn't have a button

        ElseIf Not ActiveNode.ChildNodes Is Nothing Then
            If ActiveNode.Expanded Then
                ActiveNode.Expanded = False
                BuildRoot False
            Else
                If Not ActiveNode.ParentNode Is Nothing Then
                    If ActiveNode.ParentNode.Expanded Then
                        'If Not ActiveNode.ParentNode.Level = 0 And mbRootButton Then
                        If ActiveNode.ParentNode.Level >= 0 Then
                            Set ActiveNode = ActiveNode.ParentNode
                            ScrollToView , -1
                            NodeClick ActiveNode.Control, ActiveNode 'AVDV
                        End If
                    End If
                End If
            End If
        Else
            If Not ActiveNode.ParentNode Is Nothing Then
                If ActiveNode.ParentNode.Level = 0 And Not mbRootButton Then
                    ' don't attempt to collapse the Root if it doesn't have a button
                    ' redundant ?
                ElseIf ActiveNode.ParentNode.Expanded Then
                    If ActiveNode.ParentNode.Caption <> "RootHolder" Then
                        Set ActiveNode = ActiveNode.ParentNode
                        ScrollToView ActiveNode, -1
                        NodeClick ActiveNode.Control, ActiveNode 'AVDV
                    End If
                End If
            End If
        End If
    End If

    Case vbKeyRight
        If Not ActiveNode.ChildNodes Is Nothing Then
            If ActiveNode.Expanded = False Then
                ActiveNode.Expanded = True
                If Not ActiveNode.Expander Is Nothing Then
                    NodeClick ActiveNode.Expander, ActiveNode 'AVDV
                    ' BuildRoot False will be called in NodeClick
                Else
                    ' a Root node and mbRootButton = False
                    BuildRoot False
                End If
            Else
                Set ActiveNode = ActiveNode.ChildNodes(1)
                NodeClick ActiveNode.Control, ActiveNode 'AVDV
            End If
        End If

    Case vbKeyUp, vbKeyDown
        If ActiveNode.VisIndex = mlVisCount And KeyCode = vbKeyDown Then
            ' if the activenode is the last node and collapsed, expand it and activate the 1st childnode
            If Not ActiveNode.ChildNodes Is Nothing Then
                If ActiveNode.Expanded = False Then
                    ActiveNode.Expanded = True
                    BuildRoot False
                End If
            End If
        End If

        Set cNode = NextVisibleNodeInTree(ActiveNode, (KeyCode = vbKeyUp))
    End If
End If

```

```

        If Not cNode Is Nothing Then
            Set ActiveNode = cNode
            ScrollToView ActiveNode, IIf(KeyCode = vbKeyUp, -1, -2) ' the -ve means will scroll won't change
        if the node is visible
            NodeClick ActiveNode.Control, ActiveNode
        End If

    Case vbKeyPageUp, vbKeyPageDown
        'store the activenode's vertical position to reset a similar in the keyup
        If Not mbKeyDown Then
            sngVisTop = (ActiveNode.VisIndex - 1) * NodeHeight - TreeControl.ScrollTop
            If sngVisTop > 0 And sngVisTop < TreeControl.InsideHeight Then
                msngVisTop = sngVisTop
            Else
                msngVisTop = 0
            End If
        End If

    Case vbKeyEscape
        Set MoveCopyNode(False) = Nothing

    Case vbKeySpace ' PT toggle checkbox with space
        If CheckBoxes Then
            ActiveNode.Checked = Not ActiveNode.Checked

            RaiseEvent NodeCheck(ActiveNode)
        End If
    End Select

    mbKeyDown = True ' PT

    RaiseEvent KeyDown(ActiveNode, KeyCode, Shift)
Else
    If Not mcolNodes Is Nothing Then
        If mcolNodes.Count Then
            Set ActiveNode = mcolNodes(1)
        End If
    End If
End If

End Sub

Private Sub TreeControl_KeyUp(ByVal KeyCode As MSForms.ReturnInteger, ByVal Shift As Integer)
'-----
' Procedure : TreeControl_KeyUp
' Company   : JKP Application Development Services (c)
' Author    : Jan Karel Pieterse (www.jkp-ads.com)
' Created   : 17-01-2013
' Purpose   : Handles collapsing and expanding of the tree using left and right arrow keys
              : and moving up/down the tree using up/down arrow keys
              : Also handles folding of the tree when you use the numeric keys.
'-----
    Dim lIdx As Long
    Dim sngNewScrollTop As Single

    If Not mbKeyDown Then 'PT
        ' PT KeyDown was initiated in some other control,
        ' eg Key Enter in the Editbox or tabbing to the treecontrol (enter event)
        Exit Sub
    Else

        mbKeyDown = False
    End If

    If Not ActiveNode Is Nothing Then

        Select Case KeyCode

            ' PT look into moving more key events into KeyDown

        Case 48 To 57, 96 To 105
            If KeyCode >= 96 Then KeyCode = KeyCode - 48
            If (KeyCode > vbKey0 Or mbRootButton) And Shift = 0 Then
                'SetTreeExpansionLevel (KeyCode - 49)
                ExpandToLevel (KeyCode - 48)
                BuildRoot False
            End If

        Case vbKeyF2, 93 ' F2 & key right/context menu (?) PT
            If mbLabelEdit Then

```

```
        If Not ActiveNode Is Nothing Then
            EditMode(ActiveNode) = True
            ActiveNode.Editable = True
        End If
    End If
    Case vbKeyPageUp, vbKeyPageDown
        ' PT activate node in the same position as previous activenode when scrolling
        With Me.TreeControl
            sngNewScrollTop = .ScrollTop
            lIdx = (sngNewScrollTop + msngVisTop) / NodeHeight + 1

            If (lIdx - 1) * NodeHeight < .ScrollTop Then
                lIdx = lIdx + 1

                ElseIf lIdx * NodeHeight > .InsideHeight + .ScrollTop Then
                    lIdx = lIdx - 1
                End If
        End With

        If lIdx > 1 And lIdx <= mlVisCount Then
            lIdx = mlVisOrder(lIdx)
            Set ActiveNode = mcolNodes(lIdx)
        End If

    Case vbKeyHome, vbKeyEnd
        If KeyCode = vbKeyHome Then lIdx = 1 Else lIdx = mlVisCount
        lIdx = mlVisOrder(lIdx)
        If ActiveNode.Index <> lIdx Then
            Set ActiveNode = mcolNodes(lIdx)
        End If
    Case Else

    End Select
Else
    If Not mcolNodes Is Nothing Then
        If mcolNodes.Count Then
            Set ActiveNode = mcolNodes(1)
        End If
    End If
End If
End Sub
```