Visio ER 図の情報を取得する

[Visio][VBA]

・http://msdn.microsoft.com/ja-jp/library/cc377254.aspx

モジュール

Visio の ER 図を CSV 出力するマクロ

メタ情報が取得しかたがわからない (そもそもできる？)。。。

```
' ログ出力の抑制
Private Const IS_LOGGING As Boolean = True

' 除外する Visio シート名 ( 一部一致をカンマ区切り )
Private Const EXCEPT_SHEET_KEYWORD As String = " 背景 "

' エンティティを識別するためのマスターシェイプ名
Private Const MASTER_SHAPE_ENTITY_NAME_U As String = "Entity"

'
' 主処理
'
Sub main()
    Dim docPages    As Pages
    Dim docPage     As Page
    Dim docPageCnt  As Integer
    Dim i           As Integer


    Dim entityDict  As Object   'Entity を保持する Map

    Set entityDict = CreateObject("Scripting.Dictionary")

    'Visio ドキュメントのページ単位で処理
    Set docPages = Application.ActiveDocument.Pages
    docPageCnt = docPages.Count

    For i = 1 To docPageCnt
        'Entity 辞書のクリア
        Call entityDict.RemoveAll

        Set docPage = docPages.Item(i)
        If Not IsExceptSheet(docPage.Name) Then
            Call ParsePage(docPage, entityDict)
        End If

        ' TODO
        '
        ' ここで、entityDict に 結果 { テーブル名 : テーブルオブジェクト }
        ' が格納されるので、なんらかの処理 ( ファイル出力など ) を行う
        '
        '


    Next
End Sub
'
' シートごとの処理
'
Private Sub ParsePage(ByRef docPage As Page, ByRef entityDict As Object)

    Dim docShapes       As Shapes
    Dim docShapeCnt     As Integer
    Dim i               As Integer
    Dim docShapeNameU   As String

    Set docShapes = docPage.Shapes
    docShapeCnt = docShapes.Count

    For i = 1 To docShapeCnt
        docShapeNameU = docShapes(i).NameU
```

```
        Select Case True
        Case InStr(1, docShapeNameU, MASTER_SHAPE_ENTITY_NAME_U) = 1
            ' テーブル、列、データ型
            Call ProcEntity(docShapes(i), entityDict)
        Case Else
            'Debug.Print docShapeNameU
        End Select

    Next
End Sub
'
' エンティティ処理
'
Private Sub ProcEntity(ByRef entityShape As Shape, ByRef entityDict As Object)
    Dim partsShapes     As Shapes
    Dim partsShape      As Shape
    Dim explainShape    As Shape
    Dim partsShapeCnt   As Integer
    Dim i               As Integer
    Dim j               As Integer
    Dim ent             As Entity
    Dim col             As Column
    Dim rareRow()       As String
    Dim splitedRow()    As String

    Set partsShapes = entityShape.Shapes
    partsShapeCnt = partsShapes.Count

    For i = 1 To partsShapeCnt

        Set partsShape = partsShapes(i)
        Select Case i
        Case 1
            ' テーブル名
            Set ent = New Entity
            ent.Name = partsShape.Text

            If entityDict.Exists(ent.Name) Then
                'TODO Duplicated
                Call MsgBox(" 重複 " + ent.Name)
            Else
                Call entityDict.Add(ent.Name, ent)
            End If

            Call Logging(vbCrLf & ent.Name)

        Case 2
            '2: 列情報
            rareRow = Split(partsShape.Text, vbLf)
            For j = LBound(rareRow) To UBound(rareRow)
                splitedRow = SplitRow(rareRow(j))

                Set col = New Column
                col.Name = splitedRow(0)
                col.DataType = splitedRow(1)

                If Trim(col.Name) <> "" Then
                    Call ent.AddColumn(col)
                End If
                Call Logging(vbTab & col.Name & " " & col.DataType)
            Next
        End Select
    Next

End Sub
'
' 列名とデータ型を分ける
'
Private Function SplitRow(ByRef row As String) As String()
    Const COL_SEP_CHAR As String = vbTab
    Dim sepPos          As Integer
    Dim i               As Integer
    Dim c               As String
    Dim ret(0 To 1)     As String
    Dim tmp()           As String

    ret(0) = ""
    ret(1) = ""

    If Trim(row) <> "" Then
        tmp = Split(row, COL_SEP_CHAR)
```

```
        If UBound(tmp) > 0 Then
            ret(0) = Trim(tmp(0))
            ret(1) = Trim(tmp(1))
        Else
            ret(0) = Trim(row)
        End If
    End If
    SplitRow = ret

End Function
'
' 除外シートのチェック
'
Private Function IsExceptSheet(sheetName As String) As Boolean
    Dim keywords()  As String
    Dim ret         As Boolean
    Dim i           As Integer

    keywords = Split(EXCEPT_SHEET_KEYWORD, ",")

    For i = LBound(keywords) To UBound(keywords)
        ret = InStr(sheetName, keywords(i)) > 0
        If ret Then
            Exit For
        End If
    Next

    IsExceptSheet = ret
End Function
'
' ログ
'
Private Sub Logging(log As String)
    If IS_LOGGING Then
        Debug.Print log
    End If
End Sub
```

クラス

■ Column

```
Private mName       As String
Private mDataType   As String
Private mExplain    As String


Public Property Get Name() As String
    Name = mName
End Property

Public Property Let Name(ByVal newName As String)
    mName = newName
End Property

Public Property Get DataType() As String
    DataType = mDataType
End Property

Public Property Let DataType(ByVal newDataType As String)
    mDataType = newDataType
End Property

Public Property Get Explain() As String
    Explain = mExplain
End Property

Public Property Let Explain(ByVal newExplain As String)
    mExplain = newExplain
End Property
```

■ Entity

```
Private mName       As String
Private mColumns()  As Column
```

```
Private mColumnCnt  As Integer


Public Property Get Name() As String
    Name = mName
End Property

Public Property Let Name(ByVal newName As String)
    mName = newName
End Property

Public Property Get Columns() As Column()
    Columns = mColumns
End Property
Public Property Let Columns(ByRef newColumns() As Column)
    Set mColumns = newColumns
End Property


Public Sub AddColumn(ByRef col As Column)


    ReDim Preserve mColumns(mColumnCnt)

    Set mColumns(mColumnCnt) = col

    mColumnCnt = mColumnCnt + 1
End Sub

Private Sub Class_Initialize()
    mColumnCnt = 0
End Sub
```