Access

Datumsfunktionen

eine Anleitung für Datumsfunktionen in Abfragen

Besonderheit beim Export von Datumsfeldern in csv-Dateien

Intern werden Datumsfelder mit Datum und Uhrzeit abgespeichert. Will man ein Datumsfeld in ein csv-File exportieren. wird auch die Zeit mit „00:00:00“ mitexportiert. Will man das unterbinden, hilft die Anlage eines Hilfsfeldes, das dann statt dem ursprünglichen Feld ausgegeben wird.

```
NurDatum: NFormat(ZDate([Datum]);"jjjj-mm-tt")
```

- NurDatum ist das neu formatierte Datumsfeld
- Datum ist das ursprüngliche Datumsfeld, das nicht ausgegeben werden soll

Datumsbereich einer Kalenderwoche ermitteln

Bekommt man z.B. von einem Onlineversand die Information, dass ein Artikel in der KW 50 geliefert wird, kann man mit Hilfe dieser Funktion den Datumsbereich ermitteln.

Übergabeparameter: Kalenderwoche, Jahr

```
Function KWStartDatum(intKW As Integer, _
                      intJahr As Integer) As Date
  Dim dtStart As Date
  Dim dtJahr As Date

  KWStartDatum = DateSerial(1980, 1, 1) 'Default: 1.1.1980
  If Not IsDate(DateSerial(intJahr, 1, 1)) Then
    intJahr = Year(Now) 'Aktuelles Jahr
  End If
  If intKW <1 Or intKW> 53 Then
    dtJahr = DateSerial(intJahr, 1, 1) 'ab 1.1.JJJJ
    intKW = KW(dtJahr)
    dtStart = dtJahr
  Else
    dtStart = DateSerial(intJahr, 1, 1) 'ab 1.1.JJJJ
  End If
  Do Until KW(dtStart) = intKW
    DoEvents
    dtStart = dtStart + 1
  Loop

  KWStartDatum = dtStart
```

```
End Function
```

Formularfelder nach Auswahl aus Kombinationsfeld füllen

In einem Formular soll ein Kombinationsfeld eingesetzt werden, das Kundenadressen anzeigt und nach einer Auswahl Name und Adresse in die Felder des Formulars einträgt.Ein Tipp von Smart Tools Publishing.

Leere Datenbank per VBA anlegen

Versionen: Access 2013/2010 (32- und 64-Bit), 2007, 2003, 2002/XP und Office 365 (Access)

Ein Tipp von Smart Tools Publishing

Um per VBA eine leere neue Datenbank anzulegen, setzen Sie die Funktion „CreateDatabase()“ wie folgt ein:

- Wechseln Sie mit **Alt+F11** in die VBA-Entwicklungsumgebung.

1. Wählen Sie das Menü *Extras-Verweise* an.
2. Setzen Sie eine Referenz auf „Microsoft DAO Object Library 3.6x“.
3. Nehmen Sie in Ihrer VBA-Routine die folgende Deklaration bzw. die folgenden Anweisungen auf:

Dim db As DAO.Database

Set db = DAO.CreateDatabase(„X:\Testdatenbank.mdb“, _
dbLangGeneral)
'.....
'Hier Ihre Anweisungen...
'.....
db.Close

Der erste Parameter definiert Laufwerk, Pfad und Dateiname der anzulegenden Datenbank. Über den zweiten Parameter legen Sie das Gebietsschema bzw. die Sprache fest: „dbLangGeneral“ steht hier für den Einsatz im europäischen Bereich. Weitere mögliche Konstanten für den Einsatz in anderen Regionen können Sie in Erfahrung bringen, indem Sie den Cursor auf die Anweisung „CreateDatabase“ setzen und **F1** drücken.

Ein dritter, optionaler Parameter erlaubt die Angabe einer Datenbank-/Jet Engine-Version. Wenn Sie beispielsweise aus Access 2002/XP eine Datenbank anlegen möchten, die auch mit Access 97 genutzt werden kann, geben Sie als dritten Parameter „dbVersion30“ an. Dadurch wird dann eine zur Jet Engine 3.x kompatible Datenbank angelegt. Fehlt dieser Parameter, wird die neue Datenbank in der Version der aktuellen Datenbank erstellt.

Links zu Access VBAs

Microsoft Access Database Development with VBA

Recordsets zum Bearbeiten von Datensätzen

entnommen aus den SmartTools Access Weekly 33-35

Um programmatisch auf die Inhalte von Tabellen zuzugreifen, werden „Recordsets“ eingesetzt. Ein Recordset ist eine Sammlung von Datensätzen, die in Schleifen durchlaufen und auf verschiedener Art verarbeitet werden kann.

Um mit DAO arbeiten zu können, muss zunächst ein entsprechender Verweis gesetzt sein. Access erledigt dies automatisch beim Anlegen einer neuen Datenbank. Access setzt dabei einen Verweis auf die Access Database Engine, die DAO enthält.

1. Wechseln Sie mit [Alt] + [F11] in die VBA-Entwicklungsumgebung.
2. Wählen Sie das Menü *Extras-Verweise* an.
3. Setzen Sie das Kontrollkästchen vor dem Eintrag „Microsoft ActiveX Data Objects x.x Library“. Für „x.x“ wählen Sie die höchste verfügbare Version.
4. Übernehmen Sie die Änderungen mit einem Klick auf *Ok*.

Beispiel für die Initialisierung eines Recordsets:

```
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim fld As DAO.Field
Set db = CurrentDb()
Set rs = db.OpenRecordset("Kunden", dbOpenDynaset)
For Each fld In rs.Fields
  Debug.Print fld.Value & ";";
Next
rs.Close
Set rs = Nothing
Set db = Nothing
End Sub
```

Um sich durch Recordsets zu bewegen oder den Status von Recordsets abzufragen, werden die folgenden Methoden und Eigenschaften verwendet:

- `rs.RecordCount`: Anzahl Datensätze im Recordset ermitteln
- `rs.MoveFirst`: Auf den ersten Datensatz springen
- `rs.MoveLast`: Auf den letzten Datensatz springen
- `rs.MoveNext`: Zum nächsten Datensatz gehen
- `rs.MovePrevious`: Zum vorhergehenden Datensatz gehen

ein Beispiel:

```
Sub Test3()
```

```
  Dim db As DAO.Database
  Dim rs As DAO.Recordset
  Dim Anz As Long, I As Long

  Set db = CurrentDb()
  Set rs = _
    db.OpenRecordset("SELECT * FROM Kunden " & _
                     "WHERE Land = 'Deutschland'")
  Anz = rs.RecordCount
  For I = 1 To Anz
    Debug.Print rs("Firma")
    rs.MoveNext
  Next
  rs.Close

  Set rs = Nothing
  Set db = Nothing

End Sub
```

Um einen **Feldinhalt mit DAO zu ändern,** sind grundsätzlich folgende Schritte notwendig:

```
rs.Edit:                                  Datensatz für Bearbeitung
vorbereiten
rs("Feld")= "NeuerInhalt":                Neuen Inhalt zuweisen
rs.Update:                                Datensatz speichern
```

Beispiel:

```
Sub Test6a()
  Dim db As DAO.Database
  Dim rs As DAO.Recordset
  Dim Anz As Long, I As Long

  Set db = CurrentDb()
  Set rs = _
    db.OpenRecordset("Kunden", dbOpenDynaset)
  rs.MoveLast
  Anz = rs.RecordCount
  rs.MoveFirst
  For I = 1 To Anz
    rs.Edit
    rs("UmsatzAktJahr") = 0
    rs.Update
    rs.MoveNext
  Next
  rs.Close

  Set rs = Nothing
  Set db = Nothing
```

```
End Sub
```

Um neue Datensätze zu einem Recordset hinzuzufügen, verwenden Sie mit DAO die Methode „AddNew“. Danach können die einzelnen Felder wie bei einer Änderung mit „rs(<Feldname>) = <NeuerInhalt>“ gefüllt werden. Anschließend ist bei DAO ein „Update“ notwendig, damit der neue Datensatz gespeichert wird.

Beispiel:

```
Sub Test7a()
  Dim db As DAO.Database
  Dim rs As DAO.Recordset
  Dim I As Long

  Set db = CurrentDb()
  Set rs = _
    db.OpenRecordset("Kunden", dbOpenDynaset)
  rs.AddNew
  rs("Kunden-Code") = "ABCDE"
  rs("Firma") = "Testfirma"
  .....
  rs("Telefon") = "030/123 456"
  rs.Update
  rs.Close

  Set rs = Nothing
  Set db = Nothing

End Sub
```

Das Löschen eines Datensatzes erfolgt bei DAO mit der Methode „rs.Delete“.

Beispiel:

```
Sub Test8()
  Dim db As DAO.Database
  Dim rs As DAO.Recordset

  Set db = CurrentDb()
  Set rs = _
    db.OpenRecordset("Artikel")
  While Not rs.EOF
    If rs("Auslaufartikel") = -1 And _
       rs("Lagerbestand") = 0 Then
      If MsgBox("Artikel '" & _
                rs("Artikelname") & _
                "' löschen?", _
                vbYesNo + vbQuestion) = vbYes Then
        rs.Delete
      End If
    End If
```

```
    rs.MoveNext
  Wend
  rs.Close

  Set rs = Nothing
  Set db = Nothing

End Sub
```