

R | R | Z | N |

Regionales Rechenzentrum für Niedersachsen



# Access 2010 – Programmierung Datenzugriff mit Hilfe von VBA

## DAO (Data Access Object) ...

- ist eine Möglichkeit, auf Daten in einer Access-Datenbank zuzugreifen.
- ist seit Access 2007 der Standardzugriff.
- wird in der Bibliothek Microsoft Office 14.0 Access database engine Object Library definiert.
- wird im Verzeichnis C:\Program Files\Common Files\Microsoft Shared\OFFICE14\ACEDAO.DLL gespeichert.
- wird in der Hilfe unter *Access 2010-Entwicklerreferenz – Microsoft-Datenzugriffsobjekte (DAO)-Referenz* beschrieben.

## Zugriff auf die aktuelle Datenbank

```
Sub datenbankZugriff()  
  Dim dbs As DAO.Database  
  
  Set dbs = CurrentDb  
  
  dbs.Close  
  Set dbs = Nothing  
End Sub
```

## Aktuelle Datenbank « CurrentDb » ...

- ist eine Methode des Objekts « Application». Der Rückgabewert der Methode kann einer Objektvariablen zugewiesen werden.
- liefert einen Verweis auf die aktuell, geöffnete Datenbank.
- wird zusammen mit DAO (Data Access Object) genutzt.

## Objektvariablen ...

- verweisen auf ein bestimmtes Objekt in Access.
- enthalten eine Referenz auf die Datenbank selbst oder auf Elemente in der Datenbank.
- werden für Objekte definiert, die häufig in einem Programm genutzt werden.

## ... deklarieren

« Dim dbs As DAO.Database »

« Zugriff objektvariablename As Objekttyp »

- Objektvariablen werden genauso wie Variablen vom Datentyp Integer etc. deklariert.
- Objektvariablen sind von einem bestimmten Typ „Objekt“.
- In diesem Beispiel wird eine Variable deklariert, die ein Verweis auf eine beliebige DAO-Datenbank ( zum Beispiel Access) speichern kann.

## Verweis auf ein Objekt speichern

« Set dbs = CurrentDb »

« Set objektvariablename = Objektverweis »

- Die Zuweisung muss mit dem Schlüsselwort « Set » eingeleitet werden.
- Mit Hilfe des Gleichheitszeichens wird der Variablen ein Verweis auf ein bestimmtes Objekt zugewiesen.

## Verweis zurücksetzen

« Set dbs = Nothing »

« If dbs Is Nothing Then »

- Mit Hilfe der Zuweisung « Nothing » wird der in, der Variablen gespeicherte Verweis überschrieben.
- Die Variable ist nicht initialisiert. Die Variable ist leer.
- Durch die Zuweisung wird ein Zugriff auf einen nicht gültigen Verweis vermieden.

## Datenbank schließen

- « `db.Close` » schließt die Datenbank in VBA, auf die in der Objektvariablen verwiesen wird.
- Access wird nicht geschlossen!

## Zugriff auf ein „Recordset“

```
Dim dbs As DAO.Database
Dim rs As DAO.Recordset

Set dbs = CurrentDb
Set rs = dbs.OpenRecordset("tblKunde", dbOpenTable)

rs.Close
dbs.Close

Set rs = Nothing
Set dbs = Nothing
```

## Recordset ...

- ist ein Verweis auf eine Tabelle oder Abfrage einer Datenbank.
- kann mit Hilfe einer SQL-Anweisung gebildet werden.
- liefert eine bestimmte Anzahl von Datensätzen oder keine zurück.
- ermöglicht den Zugriff auf einzelne Datenfelder.

## Ablauf

Objektvariablen deklarieren

Datenbank öffnen

Tabelle / Abfrage öffnen

Tabelle / Abfrage schließen

Datenbank schließen

Verweise zerstören

```
Dim dbs As DAO.Database  
Dim rs As DAO.Recordset
```

```
Set dbs = CurrentDB
```

```
Set rs = dbs.OpenRecordset()
```

```
rs.Close
```

```
dbs.close
```

```
Set rs = Nothing  
Set dbs = Nothing
```

## Recordset-Typen

- « .OpenRecordset("tabelle", dbOpenTable) » öffnet eine Tabelle. Die Daten werden, wie in der Tabelle gespeichert, gelesen. Wenn keine Angaben gemacht werden, wird immer versucht, eine Tabelle zu öffnen.
- « .OpenRecordset("tabelle", dbOpenDynaset) » verweist auf eine verknüpfte Tabelle oder Abfrage. Die Daten können gefiltert und sortiert werden.
- « .OpenRecordset("tabelle", dbOpenSnapshot) » verweist auf eine Momentaufnahme der Daten in einer Tabelle oder Abfrage. Die Daten können nur gelesen, aber nicht verändert werden.

## Zugriff auf eine Abfrage

```
Sub abfragenZugriff()
```

```
Dim dbs As DAO.Database
```

```
Dim rs As DAO.Recordset
```

```
Set dbs = CurrentDb
```

```
Set rs = dbs.OpenRecordset("qryGerichtKategorie", dbOpenDynaset)
```

```
rs.Close
```

```
dbs.Close
```

```
Set rs = Nothing
```

```
Set dbs = Nothing
```

```
End Sub
```

## Anzahl der Datensätze

```
Sub abfragenZugriff()
```

```
Dim dbs As DAO.Database
```

```
Dim rs As DAO.Recordset
```

```
Set dbs = CurrentDb
```

```
Set rs = dbs.OpenRecordset("qryGerichtKategorie", dbOpenDynaset)
```

```
Debug.Print "Anzahl der Datensätze: " & rs.RecordCount
```

```
rs.MoveLast
```

```
Debug.Print "Anzahl der Datensätze nach .MoveLast" & rs.RecordCount
```

```
rs.Close
```

```
dbs.Close
```

```
End Sub
```

## Hinweise

- Die Eigenschaft « .RecordCount » liefert die ungefähre Anzahl der Datensätze in einem Recordset.
- Falls als Quelle eine Tabelle genutzt wird, ist die Anzahl der Datensätze korrekt.
- Falls als Quelle eine Abfrage genutzt wird, hat « .RecordCount » immer den Wert eins. Nach dem das Recordset vollständig durchlaufen ist, wird die korrekte Anzahl von Datensätzen angezeigt.

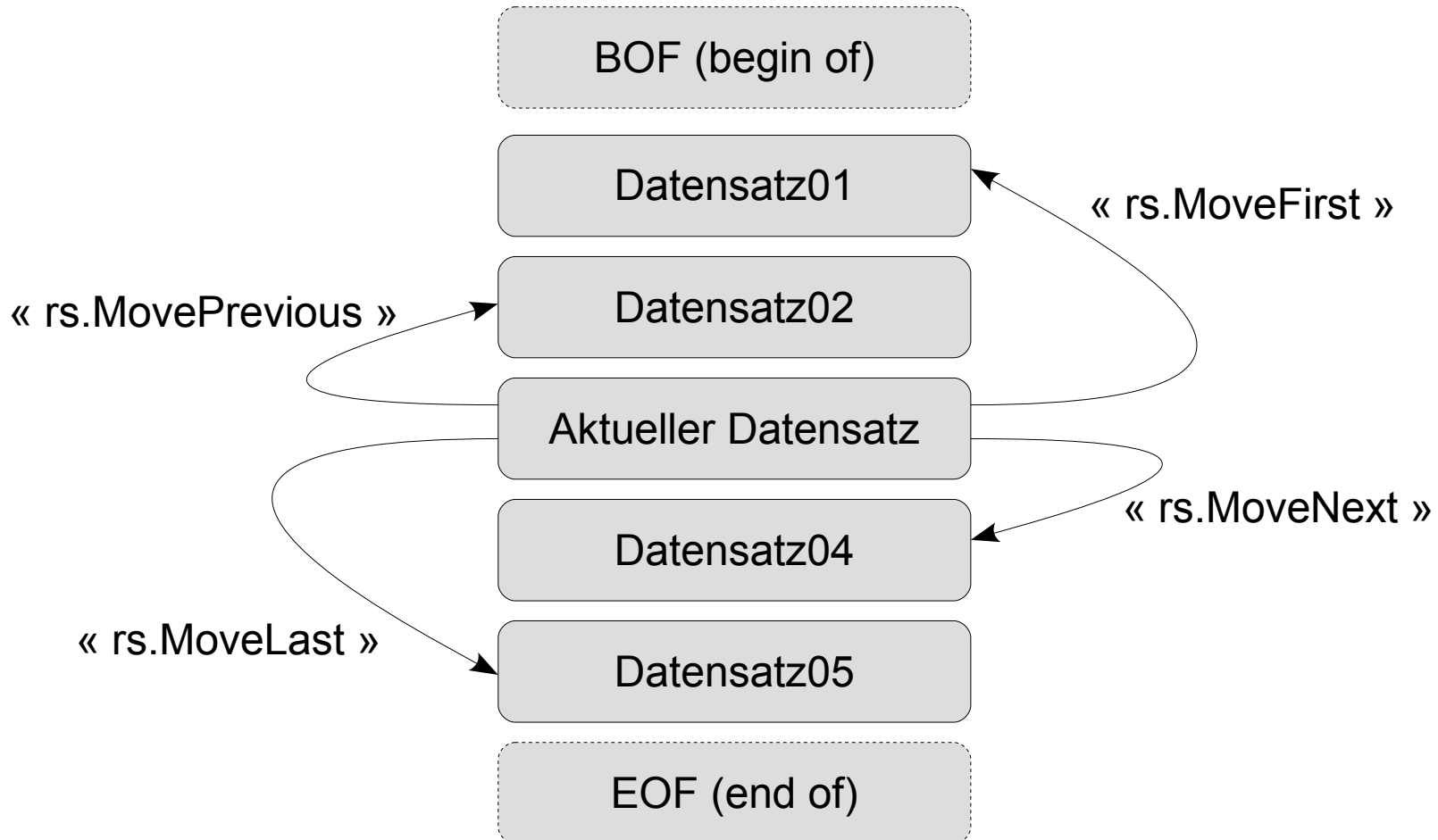
## Navigation in einem Recordset

```
With rs
  If (.BOF And .EOF) Then
    MsgBox "Das Recordset ist leer."
  Else
    .MoveFirst

    ' Alle Datensätze werden vom ersten bis zum letzten durchlaufen
    Do While Not .EOF
      Debug.Print .Fields("txtGericht")
      .MoveNext
    Loop

  End If
End With
```

# Erläuterung



## Datenfelder lesen

```
With rs
    .MoveFirst

    ' Alle Datensätze werden vom ersten bis zum letzten durchlaufen
    Do While Not .EOF
        If .Fields("txtKategorie") = "Vorspeise" Then

            End If
        .MoveNext
    Loop

End With
```

## Erläuterung

- Die Auflistung « `rs.Fields()` » enthält alle Datenfelder des gewählten Recordset.
- In den runden Klammern wird ein Index, welches ein Feld eindeutig identifiziert übergeben. In dem Beispiel wird der Name des Feldes zur Identifizierung genutzt.
- Der Name des Feldes in der Datenquelle und der Index für die Auflistung müssen exakt übereinstimmen. Andernfalls wird ein Fehler gemeldet.

## Daten ändern

```
Do While Not rs.EOF
  If ( rs.Fields("txtKategorie") = "Vorspeise") Then
    rs.Edit
    rs.Fields("txtKategorie") = "Antipasta"
    rs.Update
  End If

  rs.MoveNext
Loop
```

## Erläuterung

- « rs.Edit » kopiert den aktuellen Datensatz in einen Puffer zur späteren Bearbeitung.
- « rs.Update » speichert die Änderungen in der Datenquelle. Der Datensatzzeiger zeigt anschließend auf den geänderten Datensatz.
- Bei einem Wechsel des Datensatzes werden Änderungen nicht automatisch gespeichert. Falls die Anweisung « rs.Update » nicht ausgeführt wird, gehen die Änderungen verloren.

## Daten hinzufügen

```
Dim dbs As DAO.Database
Dim rs As DAO.Recordset

Set dbs = CurrentDb
Set rs = dbs.OpenRecordset("tblKategorie", dbOpenDynaset)

With rs
    .AddNew
    .Fields("txtKategorie") = "Salate"
    .Update
End With
```

## Erläuterung

- « rs.AddNew » erstellt einen neuen Datensatz und fügt diesen dem Recordset hinzu. Standardwerte für die einzelnen Datenfelder werden bei der Neuanlage beachtet.
- « rs.Update » speichert den neuen Datensatz in der Datenquelle. Falls diese Anweisung nicht ausgeführt wird, geht der neue Datensatz ohne Warnung verloren.
- Der neue Datensatz muss eindeutig mit Hilfe eines Schlüsselwertes identifiziert werden.

## Datenquelle eines Listenfeldes klonen

```
Private Sub cmdGetAll_Click()  
    Dim rs As DAO.Recordset  
  
    Me.IstAuswahl.ColumnCount = Me.IstOptionen.ColumnCount  
    Me.IstAuswahl.ColumnWidths = Me.IstOptionen.ColumnWidths  
  
    Set rs = Me.IstOptionen.Recordset ' Datenquelle  
  
    Me.IstAuswahl.RowSourceType = "Table/Query"  
    Set Me.IstAuswahl.Recordset = rs  
  
    rs.Close  
    Set rs = Nothing  
End Sub
```

## Daten löschen

With rs

.MoveFirst

Do While Not .EOF

    If (.Fields("IDGericht") = index) Then

        .Delete

    Exit Do

    End If

    .MoveNext

Loop

End With

## Erläuterung

- « rs.Delete » löscht einen Datensatz aus der Datenquelle ohne Nachfrage.
- Der Datensatzzeiger zeigt anschließend auf den gelöschten Datensatz und muss vom Entwickler neu gesetzt werden.
- Formulare / Berichte, die diesen Datensatz anzeigen, müssen anschließend mit Hilfe der Methode « .Refresh » die Datenquelle neu abfragen.

## Transaktionen rückgängig machen

With rsQuelle

Do While Not .EOF

ws.BeginTrans

rsZiel.AddNew

...

rsZiel.Update

button = MsgBox(ausgabe, vbYesNo, "Warnhinweis")

If button = vbYes Then

ws.CommitTrans

Else

ws.Rollback

End If

## Voraussetzung

- Ein Workspace (Arbeitsbereich) ist definiert.
- In Access kann immer nur ein Arbeitsbereich geöffnet werden.
- Beispiel:
  - « Dim ws As DAO.Workspace » definiert eine Variable vom Typ „Arbeitsbereich“.
  - « ws = DAO.DBEngine.Workspaces(0) » erstellt eine Objektvariable, die auf den Standardarbeitsbereich verweist.

## Transaktion beginnen

« ws.BeginTrans »

- Eine Transaktion beginnt. Alle Operationen werden gepuffert.
- Transaktionen sind ...
  - Änderungen an Datensätzen,
  - neue Datensätze hinzufügen oder
  - Löschen von Datensätzen.
- Einzelne Transaktionen können gebündelt werden.

## Transaktion übernehmen

« ws.CommitTrans »

- Die Änderungen an den Datensätzen werden aus dem Puffer in die Tabelle übernommen.

## Transaktion rückgängig machen

« ws.Rollback »

- Die Transaktionen wird beendet.
- Änderungen an den Datensätzen werden nicht übernommen.  
Die Datenbank wird in den Ursprungszustand versetzt.