Kapitel 3

Datenmakros

In diesem Kapitel:

W. Doberenz, Th. Gewinnus: Microsoft Access 2010 - Programmierung;
 ISBN 978-3-86645-459-0

Eines der bedeutendsten neuen Features in Access 2010 ist die Fähigkeit, Datenmakros zu nativen Access Tabellen und auch zu Tabellen in Web-Datenbanken hinzuzufügen. Sogar Tabellen, die mit Access-Daten in anderen Datenbanken verlinkt sind, unterstützen Datenmakros.

HINWEIS Die herkömmlichen Makros, wie sie hauptsächlich zum Steuern der Benutzerschnittstelle verwendet werden (Formular oder Bericht öffnen, Reagieren auf Button-Klicks usw.) sind vereinbarungsgemäß kein Thema dieses Buchs, da hier der Einsatz von VBA-Code in der Regel die weitaus leistungsfähigeren Alternativen bietet.

Obwohl unser Buch sich schwerpunktmäßig der VBA-Programmierung widmet und Datenmakros kein VBA kennen, können wir es uns leider nicht leisten, die Datenmakros links liegen zu lassen, da diese über Features verfügen, die weit über die Möglichkeiten von VBA hinausgehen bzw. den VBA-Code drastisch vereinfachen. Der Access-Anwendungsprogrammierer ist deshalb gut beraten, sowohl VBA als auch Datenmakros sinnvoll miteinander zu kombinieren.

HINWEIS Einen ersten Eindruck vermittelt Ihnen das Beispiel »Programmieren mit Datenmakros« in Kapitel 1.

Im Folgenden gehen wir davon aus, dass Sie bereits über Grundkenntnisse der Programmierung von UI-Makros verfügen, d.h., dass der Umgang mit Makro-Editor und Ausdrucks-Generator für Sie kein allzu großes Problem ist.

Grundlagen von Datenmakros

Ein Datenmakro ist Logik, die Sie an eine Tabelle »anheften« um datenorientierte Aktivitäten auf Tabellenebene durchzuführen. Weil Datenmakros auf Tabellenebene eingesetzt werden, wird exakt dieselbe Aktion immer dann ausgeführt, wenn Tabellendaten aktualisiert werden.

Einsatzmöglichkeitem

Es gibt viele unterschiedliche Möglichkeiten für die Verwendung von Datenmakros in Access 2010, beispielsweise:

- Überprüfen, ob ein Kunde bezahlt hat bevor weitere Bestellungen entgegengenommen werden
- Versandkosten automatisch als Teil der Verkaufstabelle berechnen
- Absichern, dass der Wert eines Felds innerhalb eines bestimmten Bereichs liegt, bevor der Datensatz abgespeichert wird
- Änderungen an einer Tabelle protokollieren

Natürlich wären die gleichen Regeln auch leicht mit den herkömmlichen UI-Makros umsetzbar. Die entsprechende Logik müsste dann aber für jeden einzelnen Fall programmiert werden, in welchem Daten durch die Anwendung geändert werden. Wenn aber diese relativ einfachen Aktionen auf Datenebene implementiert werden, wird die UI-Logik entlastet und kann sich komplexeren Operationen widmen (mittels VBA Code und herkömmlichen Makros).

Funktionsprinzip

Datenmakros können Sie zu folgenden Tabellen-Ereignissen hinzufügen:

- *Vor Änderung (BeforeChange)*
- *Vor Löschung (BeforeDelete)*
- *Nach Einfügung (AfterInsert)*
- *Nach Aktualisierung (AfterUpdate)*
- *Nach Löschung (AfterDelete)*

Wie Sie leicht erkennen, teilen sich die auswertbaren Ereignisse in zwei Gruppen auf, die *Vorabereignisse* und die *Nachfolgeereignisse*.

Datenmakros sind vergleichbar mit den Triggern des Microsoft SQL Server. Gewissermaßen ähnelt ein Datenmakro einer Validierungsregel, nur dass eine Validierungsregel recht »dumm« ist, denn sie kann keine Daten ändern oder bestimmen, welche Korrekturen erforderlich sind. Alles was sie kann ist die Anzeige einer Meldung an den User.

Oft verwendet man Datenmakros zur Durchsetzung von Geschäftsregeln – beispielsweise darf ein Betrag eine bestimmte Größe nicht überschreiten – oder für Datenkonvertierungen während der Eingabe. Obwohl sich dies auch leicht mit VBA-Code realisieren lässt, ist der große Vorteil von Datenmakros, dass sie immer und überall wirksam werden, wo man die Daten der Tabelle nutzt. Wenn Sie ein Datenmakro für ein bestimmtes Tabellenereignis definieren, wird es immer und zuverlässig ausgeführt, unabhängig davon wie auf die Daten zugegriffen wird (Makros, VBA, SQL, DAO, ...).

BEISPIEL

Im letzten Einführungsbeispiel von Kapitel 1 wird ein mit dem *Vor Änderung*-Ereignis verbundenes Datenmakro in eine *Personal*-Tabelle eingebettet, um sicherzustellen, dass das Monatsgehalt einer Person einen bestimmten Höchstbetrag nicht übersteigt. Jedes Mal, wenn das Monatsgehalt neu eingegeben oder aktualisiert werden soll, tritt das Datenmakro in Aktion. Egal ob direkt in die Tabelle geschrieben wird oder die Eingabe in einem oder mehreren angeschlossenen Formularen oder Berichten erfolgt – das Datenmakro beobachtet die Änderungen und überwacht die Tabellendaten.

Datenmakros teilen einige der fortgeschrittenen Konstrukte von UI-Makros, z.B. die Verzweigung mittels *Wenn (If)*-Aktion oder das Iterieren durch Datensätze mit der *Für jeden Datensatz (ForEachRecord)*-Aktion. Zusammen mit den Aktionen der Datenmakros ergibt dies ein leistungsfähiges Werkzeug für den Access-Entwickler.

HINWEIS Datenmakros funktionieren sofort, sodass Sie einfach damit arbeiten und die Wirkungen beobachten können, ohne dass Sie kompilieren oder zwischen Entwurfs- und Datenblattansicht der Tabellen umschalten müssten.

W. Doberenz, Th. Gewinnus: Microsoft Access 2010 - Programmierung;

Datenmakros und VBA

In den Becher der Freude fällt für uns VBA-Programmierer leider ein dicker Wermutstropfen:

HINWEIS Datenmakros verstehen kein VBA, die Arbeit mit dem Makro-Editor ist dem VBA-Programmierer ein Graus[1]!

Hinzu kommt, dass der Informationsaustausch zwischen VBA und Datenmakros sehr umständlich ist. Aus unerfindlichen Gründen wird beispielsweise die *TempVars*-Auflistung nur von UI-Makros, nicht aber von Datenmakros unterstützt[2].

Doch Kopf hoch! Ob wir wollen oder nicht, wir müssen uns im Folgenden mit der ungeliebten aber durchaus nützlichen Spezies der Datenmakros nach dem Prinzip »soviel wie nötig« auseinandersetzen.

Erzeugen von Datenmakros

Im Unterschied zu den UI-Makros werden Datenmakros nicht im Navigationsbereich unter Makros angezeigt, sondern für eine in der Datenblattansicht gezeigte Tabelle in der Registerkarte *Tabelle* verwaltet.

Es gibt zwei Haupttypen von Datenmakros:

- ereignisgesteuerte Datenmakros (siehe »Ein ereignisgesteuertes Datenmakro erstellen» auf Seite 148)
- benannte Datenmakros (siehe »Arbeiten mit einem benannten Datenmakro» auf Seite 153)

Erstere sind unmittelbar mit einem bestimmten Tabellenereignis verbunden, letztere können von verschiedenen Stellen aus aufgerufen werden (von VBA über die *RunDataMacro*-Methode).

Datenmakros nutzen denselben Makro-Editor, wie er auch für eingebettete und für UI-Makros eingesetzt wird. Der wesentliche Unterschied besteht darin, dass der Aktionskatalog unterschiedliche Aktionen in Abhängigkeit vom Kontext anbietet.

Im Allgemeinen werden Datenmakros erzeugt, indem mehrere Makro-Aktionen miteinander verknüpft werden, von denen jede eine einfache Operation ausführt, beispielsweise das Zuweisen eines Feldwertes in einem Datensatz.

Das Hinzufügen eines Datenmakros zu einer Tabelle ist recht einfach. Tatsächlich muss die Access-Tabelle nicht einmal in der Entwurfsansicht angezeigt werden (siehe Abbildung 3.1) – sie kann auch in der Datenblattansicht eingeblendet sein.

Abbildung 3.1 Teilansicht der Registerkarte *Tabellentools*

[1] Die Pein beginnt bereits bei der auf »Teufel komm raus» krampfhaft eingedeutschten Syntax ...

[2] Einen Workaround bietet die *RunDataMacro*-Methode bzw. das Zwischenspeichern von Übergabeparameter in einer Hilfstabelle der Datenbank.

Datenmakros umbenennen, löschen und ändern

Über den Menübefehl *Makro umbenennen/löschen* (siehe Abbildung 3.1) lässt sich der Datenmakro-Manager öffnen. Dieser bietet eine Übersicht über alle in der Datenbank enthaltenen ereignisgesteuerten und benannten Datenmakros (siehe Abbildung 3.2).

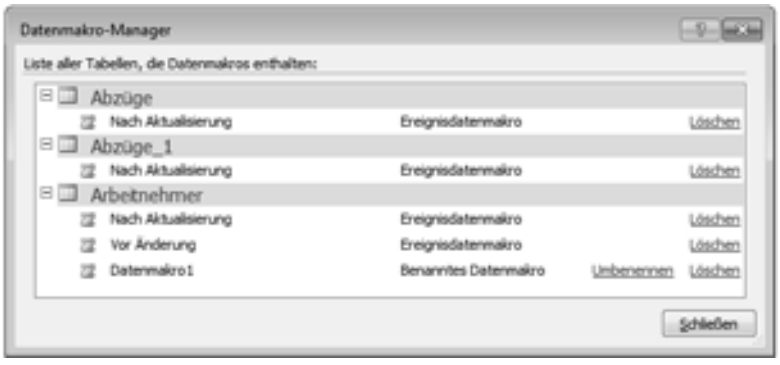

Abbildung 3.2 Der Datenmakro-Manager zeigt alle in der Datenbank enthaltenen Datenmakros (siehe Praxisbeispiele am Ende des Kapitels)

Leider lassen sich mit dem Datenmakro-Manager nur Umbenennungen und Löschungen vornehmen. Wollen Sie ein vorhandenes Datenmakro editieren, so müssen Sie wieder den Menübefehl *Datenmakros erstellen* verwenden und den Makro-Editor für das betreffende Tabellenereignis öffnen.

USysApplicationLog

Die *USysApplicationLog*-Tabelle ist eine Systemtabelle, die Datenmakro- und Anwendungsfehler protokolliert (siehe Abbildung 3.3).

Die aufrufenden Makros können Parameterwerte übergeben und eine Collection von Rückgabewerten oder einen Fehler zurückbekommen. Die *USysApplicationLog* Tabelle kann leicht in der Backstage-Ansicht betrachten werden, die sowohl in Web- als auch Nicht-Web-Datenbanken zur Verfügung steht.

Vorher aber sollten Sie über das Kontextmenü *Navigationsoptionen...* (Klick mit der rechten Maustaste auf den Navigationsbereich) die Anzeigeoption *Systemobjekte anzeigen* einschalten.

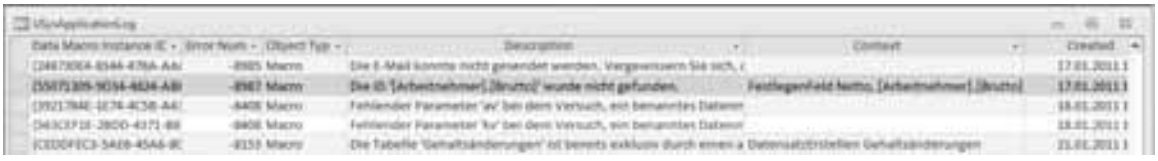

Abbildung 3.3 Beispiel für Einträge in der Tabelle *USysApplicationLog*

HINWEIS Ein Eintrag in die Tabelle *USysApplicationLog* kann auch durch die Aktion *ProtokollierenEreignis* eines Nachfolgeereignisses erfolgen (siehe Tabelle 3.1).

W. Doberenz, Th. Gewinnus: Microsoft Access 2010 - Programmierung;

Aktionen in Datenmakros

Wie jedes andere Makro besteht auch ein Datenmakro aus einer Folge von Aktionen, die mit dem Makro-Editor bzw. dem Ausdrucks-Generator zugewiesen werden. Der erfahrene Makro-Programmierer wird allerdings feststellen, dass die Anzahl der pro Ereignis zur Verfügung stehenden Aktionen deutlich geringer ist als bei den herkömmlichen UI-Makros.

Aktionen für alle Tabellenereignisse

Die Tabelle 3.1 zeigt die für alle Vorab- und Nachfolgeereignisse verfügbaren Aktionen.

Aktion	Beschreibung
Gruppieren	... ermöglicht es, Aktionen und Programmfluss in einem benannten Block zu gruppieren, der reduziert werden kann und nicht ausgeführt wird
Kommentar	... zeigt Informationen an, wenn das Makro ausgeführt wird
Wenn	... führt einen logischen Block aus, wenn die Bedingung wahr ist
AuslösenFehler	... teilt der Anwendung mit, dass ein Fehler ausgelöst wurde
BeiFehler	... legt die bei Auftreten des Fehlers auszuführende Aktion fest
FestlegenFeld	... weist das Ergebnis eines Ausdrucks dem Wert eines Felds zu
FestlegenLokaleVar	... erstellt oder ändert eine lokale Variable
LöschenMakroFehler	... löscht den Inhalt des *MacroError*-Objekts
NachschlagenDatensatz	... führt Aktionen mit dem Datensatz aus, der über das Abfrage-Argument nachgeschlagen wurde
StoppMakro	... beendet das aktuelle Makro sofort

Tabelle 3.1 Gemeinsame Aktionen aller Tabellenereignisse

Zusätzliche Aktionen der Nachfolgeereignisse

Die Nachfolgeereignisse (*Nach Einfügung*, *Nach Aktualisierung* und *Nach Löschung*) ermöglichen zusätzliche Aktionen, die von den Vorabereignissen (*Vor Änderung*, *Vor Löschung*) nicht unterstützt werden (siehe folgende Tabelle 3.2).

Aktion	Beschreibung
StoppAlleMakros	... beendet sofort alle gerade ausgeführten Makros
AusführenDatenmakro	... führt ein benanntes Makro in der Datenbank aus
DatensatzBearbeiten	... leitet einen Block ein, mit dessen Aktionen der Datensatz bearbeitet wird
DatensatzErstellen	... erzeugt einen neuen Datensatz
DatensatzLöschen	... löscht einen Datensatz, der durch einen Ausdruck spezifiziert wurde

Tabelle 3.2 Zusätzliche Aktionen der Tabellenereignisse *Nach Einfügung*, *Nach Aktualisierung*, *Nach Löschung*

Aktion	Beschreibung
AbbrechenDatensatzänderung	... beendet den *DatenSatzErstellen*- oder *DatensatzBearbeiten*-Datenblock, ohne den aktuellen Datensatz zu speichern
FürJedenDatensatz	... leitet einen Block mit Aktionen ein, die für jeden Datensatz ausgeführt werden, der einer Bedingung entspricht
ProtokollierenEreignis	... protokolliert einen Datensatz in der *USysApplicationLog*-Tabelle (siehe Seite 145)
SendenEMail	... sendet eine E-Mail

Tabelle 3.2 Zusätzliche Aktionen der Tabellenereignisse *Nach Einfügung*, *Nach Aktualisierung*, *Nach Löschung (Fortsetzung)*

HINWEIS Einige Aktionen (z.B. *FestlegenFeld*) sind nicht sofort, sondern erst innerhalb eines anderen untergeordneten Blocks verfügbar!

Auswahl des richtigen Tabellenereignisses

Wann soll ich welches Tabellenereignis verwenden? Im Folgenden wollen wir versuchen, diese Frage kurz zu beantworten sowie auf die Beschränkungen von Datenmakros einzugehen.

Vorabereignisse

Die Ereignisse *Vor Änderung* und *Vor Löschung* sind für schnelle, leichtgewichtige Operationen gedacht. Datenmakros, die mit diesen Ereignissen verknüpft sind, können die alten und neuen Werte des aktuellen Datensatzes auswerten und diese mit einem Rekord der aktuellen oder einer anderen Tabelle vergleichen, indem sie die Aktion *NachschlagenDatensatz* verwenden.

Sie können *FestlegenFeld* verwenden, um Daten in der geänderten Datenzeile zuzuweisen oder die Änderung unterdrücken. Um tatsächlich eine leichtgewichtige Operation vorzunehmen sollte man es aber vermeiden über eine Auflistung von Datensätzen zu iterieren. Auch ein Aufruf benannter Datenmakros ist aus gutem Grund nicht möglich.

HINWEIS Das *Vor Änderung*-Ereignis feuert sowohl beim Einfügen als auch beim Aktualisieren, man kann aber den *IsInsert*-Ausdruck verwenden, um den Typ der Operation zu ermitteln.

Nachfolgeereignisse

Die Ereignisse *Nach Aktualisierung, Nach Einfügung* und *Nach Löschung* unterstützen auch länger andauernde Operationen, die beispielsweise Iterationen durchführen. Verfügbar sind die originalen und die geänderten Werte.

Die durch diese Ereignisse ausgelösten Makros können andere Datensätze innerhalb der Tabelle und in anderen Tabellen auswerten und modifizieren. Typischerweise sollten Sie diese Ereignisse nicht verwenden um den aktuellen Rekord zu modifizieren, dafür sind die Vorabereignisse *Vor Änderung* und *Vor Löschung* besser geeignet.

W. Doberenz, Th. Gewinnus: Microsoft Access 2010 - Programmierung;

Gelegentlich brauchen Sie ein Datenmakro um den aktuellen Rekord zu modifizieren, wobei das Makro wiederholt rekursiv aufgerufen wird.

HINWEIS Datenmakros sind auf 10 Rekursionsebenen beschränkt!

Man kann aber die *Aktualisiert ("FeldName")*-Funktion aufrufen, welche *Wahr* oder *Falsch* zurückgibt, um die Spalte(n) zu bestimmen, die von der aktuellen Änderung betroffen sind. Dieses Feature kann helfen, zyklische Rekursionen zu vermeiden.

Wichtige unterstützte Aktionen sind *DatensatzErstellen*, *DatensatzBearbeiten* und *DatensatzLöschen* (siehe Tabelle 3.2). Die Nachfolgeereignisse erlauben unter anderem auch das Erstellen einer Log-Datei (siehe »Änderungen von Tabelleninhalten protokollieren« ab Seite 158).

Einschränkungen

Auf folgende Einschränkungen der Datenmakros sei besonders hingewiesen:

- Im Unterschied zu den Triggern eines SQL-Servers funktionieren Datenmakros nicht innerhalb einer Transaktion. Access 2010 stellt keinerlei Transaktionsmechanismen für irgendwelche seriellen Datenoperationen zur Verfügung, diese sind alle atomar.
- Datenmakros können keine Daten von mehrwertigen (multi-value) oder angefügten (attachment) Spalten verarbeiten.
- Access 2007 SP1 kann Daten in verlinkten Access 2010 Tabellen mit Datenmakros nur lesen und nicht schreiben, weil die Access 2007-Datenbankengine diese nicht ausführen kann.
- Wenn SharePoint-Listen in Access-Applikationen offline sind, wird die Ausführung der Makros so lange verzögert, bis sich der User wieder mit dem Server verbindet. Alle Änderungen im abgetrennten Client werden automatisch an den Server geschickt sobald die Verbindung verfügbar ist und die Datenmakros auf dem Server laufen.

Praxisbeispiele

Die folgenden Beispiele bauen auf einer gemeinsamen Datenbasis auf und sollten am besten von vorn nach hinten abgearbeitet werden[1].

Ein ereignisgesteuertes Datenmakro erstellen

Wenn-, *NachschlagenDatensatz*-, *FestlegenLokaleVar*-Aktion; *Aktualisiert*-, *Updated*-Ausdruck;

Am Beispiel einer Gehaltsabrechnung für Arbeitnehmer wollen wir das Erstellen und die Funktionsweise eines mit dem Ereignis *Vor Änderung* verbundenen Datenmakros demonstrieren. Unser Beispiel ist etwas anspruchsvoller als das Einführungsbeispiel »Programmieren mit Datenmakro« von Kapitel 1, denn in

[1] Die hier gezeigte Brutto-Netto-Lohnrechnung ist nur für Lernzwecke, keinesfalls aber für den kommerziellen Gebrauch geeignet!

unserem Datenmakro ist auch so genannte *LookUpRecord*-Funktionalität zu implementieren, da Werte in einer Fremdtabelle nachgeschlagen werden müssen.

Datenbank

Ausgangspunkt ist eine Tabelle *Arbeitnehmer*, deren Datenblattansicht die Abbildung 3.4 zeigt[1].

Arbeitnehmer

ID	Nachname	Vorname	Geburtstag	Brutto	Netto
1	Müller	Paul	14.04.1976	1.900,00 €	1.508,13 €
2	Krause	Hans	11.11.1965	2.200,00 €	1.746,25 €
3	Schulze	Wolfgang	15.08.1949	1.350,00 €	1.071,56 €
4	Kowalski	Ede	10.05.1962	1.500,00 €	1.190,63 €
5	Wank	Friedhelm	30.12.1962	1.100,00 €	873,13 €
6	Kieninger	Harald	14.01.1945	3.200,00 €	2.540,00 €
7	Hoffmann	Ute	15.09.1954	810,00 €	642,94 €
(Neu)					

Datensatz: 1 von 7 | Kein Filter | Suchen

Abbildung 3.4 Datenblattansicht der Tabelle *Arbeitnehmer*

Neben den allgemeinen Personaldaten wird in die Tabelle nur das *Brutto*-Gehalt eingegeben, das *Netto*-Gehalt hingegen soll mit einem Datenmakro berechnet und automatisch eingetragen werden.

Der Arbeitnehmeranteil der gesetzlichen Abzüge ist in der Tabelle *Abzüge* gespeichert. In unserem Beispiel besteht diese Tabelle einfachheitshalber aus nur zwei Datensätzen (Werte vor dem 31.12.2010 und Werte ab 1.1.2011, siehe Abbildung 3.5)[2].

Abzüge

ID	KV	RV	PV	AV	Info
1	7,90%	9,95%	0,98%	1,40%	gültig bis 31.12.2010
2	8,20%	9,95%	0,98%	1,50%	gültig ab 1.1.2011
(Neu)					

Datensatz: 1 von 2 | Kein Filter | Suchen

Abbildung 3.5 Datenblattansicht der Tabelle *Abzüge*

Sobald in der Tabelle *Arbeitnehmer* der *Brutto*-Wert verändert werden soll, müssen in der Tabelle *Abzüge* die Prozentwerte für Kranken-, Renten-, Pflege- und Arbeitslosenversicherung nachgeschlagen werden.

Erstellen des Datenmakros

Öffnen Sie die Tabelle *Arbeitnehmer* in der Entwurfsansicht und klicken Sie auf die Registerkarte *Entwurf*. Hier klappen Sie den Menüeintrag *Datenmakros erstellen* auf (siehe Abbildung 3.6). Es werden die fünf Tabellen-Ereignisse angeboten (siehe Abbildung 3.6).

[1] Dabei sind hier die *Netto*-Beträge bereits durch das Datenmakro eingetragen, im Ausgangszustand ist die *Netto*-Spalte leer!

[2] Für die Richtigkeit können die Autoren keine Haftung übernehmen, Erweiterungen bzw. Spezifizierungen für weiter zurückliegende Jahre dürften aber kein Problem darstellen.

Abbildung 3.6 Erstellen eines Datennmakros

Zum gleichen Ziel können Sie aber auch über die Datenblattansicht der Tabelle gelangen: Klicken Sie auf die Registerkarte *Tabelle* und auch hier werden Ihnen alle fünf Tabellenereignisse angeboten (siehe Abbildung 3.7).

Abbildung 3.7 Datenmakro in der Datenblattansicht hinzufügen

Wir entscheiden uns in beiden Fällen für das Ereignis *Vor Änderung* und es öffnet sich der Makro-Editor (siehe Abbildung 3.8).

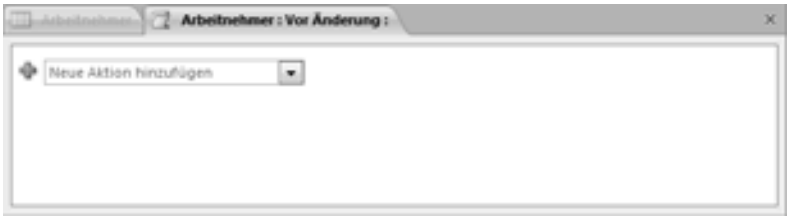

Abbildung 3.8 Makroeditor

Warum haben wir uns für das Ereignis *Vor Änderung* entschieden?

Es handelt sich bei der Berechnung des Nettowerts um eine schnelle, leichtgewichtige Operation, da nur ein einzelner Datensatz geändert werden muss und eine Iteration über mehrere Datensätze nicht erforderlich ist. Das Abspeichern des geänderten Datensatzes erfolgt erst nach Abschluss der Bearbeitung (siehe dazu Seite 147).

Arbeiten mit dem Makro-Editor

Ihre »Programmiertätigkeit« besteht im Wesentlichen aus dem Festlegen einer Abfolge einzelner Aktionen[1], die im Kombinationsfeld *Neue Aktion hinzufügen* angeboten werden. Alternativ können Sie auch doppelt auf den entsprechenden Eintrag im Aktionskatalog klicken (siehe Abbildung 3.9).

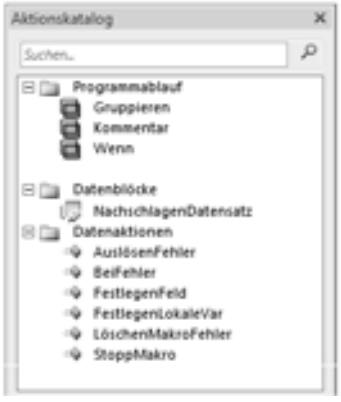

Abbildung 3.9 Aktionskatalog

Das Eintragen des Codes in die sich bei Bedarf öffnenden Eingabefelder ist größtenteils selbsterklärend und wird vom Makro-Editor nach besten Kräften durch Intellisense unterstützt.

Das vorweggenommene Endergebnis zeigt die Abbildung 3.10

Abbildung 3.10 Das fertige Datenmakro im Makro-Editor

[1] Für den VBA-Programmierer wird das Verständnis der blockweise organisierten Struktur kein Problem darstellen, das dürften eher die furchtbaren deutschen Bezeichner sein.

Im Folgenden wollen wir die Aktionen im Einzelnen kommentieren:

Wenn Aktualisiert

Das komplette Datenmakro ist in einem *Wenn*-Block eingeschlossen, der nur dann ausgeführt wird, wenn der Wert des *Brutto*-Felds geändert wurde:

```
Wenn Aktualisiert("Brutto") Dann
   ...
Ende Wenn
```

erfüllt ist[1].

HINWEIS Achten Sie darauf, dass der Feldnamen in doppelte Anführungszeichen einzuschließen ist, anderenfalls wird nur der Wert des Feldes genommen!

Datensatz nachschlagen

Bei der Aktion *NachschlagenDatensatz* werden Ihnen alle in der Datenbank verfügbaren Tabellen angeboten. Nach Auswahl der Tabelle *Abzüge* entscheiden wir uns für den zweiten Datensatz (gültig ab 1.1.2011) und tragen deshalb die folgende Bedingung ein:

```
Bedingung = [Abzüge].[ID]=2
Alias   Abz
```

Um später vereinfacht auf die einzelnen Felder zugreifen zu können, haben wir für diesen Datensatz den Alias *Abz* eingetragen (siehe auch Abbildung 3.11).

Abbildung 3.11 Programmieren der Aktion *NachschlagenDatensatz* im Makro-Editor

FestlegenLokaleVar

Mit dieser Aktion führen wir die eigentliche Berechnung des Nettogehalts durch, d.h., wir erzeugen eine lokale Variable *nett* und weisen ihr einen Ausdruck zu, in welchem Netto aus Brutto minus Abzüge ermittelt wird:

```
Name     nett
Ausdruck = [Arbeitnehmer].[Brutto]*(1-[Abz].[AV]-[Abz].[KV]-[Abz].[PV]-[Abz].[RV]
```

[1] Sie können auch *Updated("Brutto")* eintragen, der Makro-Editor korrigiert das später automatisch in *Aktualisiert("Brutto")*.

FestlegenFeld

Die letzte Aktion. Hier weisen wir dem Feld *Netto* der *Arbeitnehmer*-Tabelle den Wert der lokalen Variablen *nett* zu:

```
Name    Arbeitnehmer.Netto
Wert    [nett]
```

Schließen Sie den Makro-Editor und speichern Sie die Tabelle.

Test des Datenmakros

Zum Testen der Funktionsfähigkeit unseres Datenmakros brauchen wir nicht unbedingt ein Formular. Es genügt, wenn wir die Datenblattansicht der Tabelle *Arbeitnehmer* öffnen und dort an einzelnen Bruttogehältern herumdoktern. Wie von Geisterhand werden die entsprechenden Nettogehälter eingetragen bzw. korrigiert (siehe Abbildung 3.4).

Bemerkungen

- Für das Umbenennen und Löschen eines Datenmakros nutzen Sie den Menübefehl *Makro umbenennen/löschen*.
- Für das nachträgliche Bearbeiten eines vorhandenen Datenmakros gibt es keinen extra Menübefehl. Gehen Sie deshalb so vor, als wollten Sie ein neues Datenmakro zum Tabellenereignis *Vor Änderung* anlegen. Es öffnet sich der Makro-Editor mit dem bereits vorhandenen Datenmakro.
- Erwartungsgemäß werden Sie beim Experimentieren mit unserem Beispiel feststellen, dass Änderungen der Beitragssätze in der Tabelle *Abzüge* keinerlei unmittelbare Auswirkungen auf die Nettogehälter der Tabelle *Arbeitnehmer* haben, es sei denn, Sie ändern nacheinander alle Bruttogehälter und lassen sich jeweils den entsprechenden Nettowert berechnen und eintragen. Dies wäre, insbesondere bei vielen Datensätzen, eine »schildbürgerhafte« Vorgehensweise und wir müssen uns deshalb nach einer praktikableren Lösung umschauen (siehe folgendes Beispiel).

Arbeiten mit einem benannten Datenmakro

AusführenDatenmakro-, *FürJedenDatensatz-*, *DatensatzBearbeiten*-Aktion*; DoCmd*-Objekt: *SetParameter-*, *RunDataMacro*-Methode;

Ein benanntes oder »eigenständiges« Datenmakro ist zwar auch einer bestimmten Tabelle zugeordnet, im Unterschied zum ereignisgesteuerten Datenmakro aber keinem bestimmten Ereignis. Sie können ein benanntes Datenmakro in anderen Datenmakros, eigenständigen Makros oder auch per VBA-Code aufrufen.

Im vorliegenden Beispiel wollen wir zur Tabelle *Arbeitnehmer* (siehe Vorgängerbeispiel) ein benanntes Datenmakro hinzufügen, welches quasi »in einem Ritt« alle Nettogehälter korrigiert. Übergabeparameter sind die einzelnen Beitragssätze, die der Tabelle *Abzüge* entnommen werden sollen.

Vorbereitungen

Öffnen Sie die Tabelle *Arbeitnehmer* in der Datenblattansicht und klicken Sie auf der Registerkarte *Tabelle* auf *Benanntes Makro* und dann auf *Benanntes Makro erstellen* (siehe Abbildung 3.12).

Es wird nun der Makro-Generator geöffnet, in welchem Sie mit dem Hinzufügen von Aktionen beginnen können.

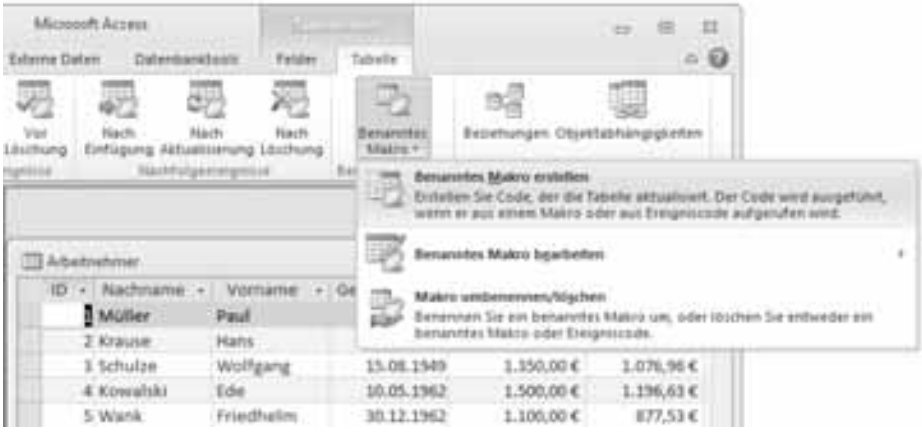

Abbildung 3.12 Erstellen des benannten Datenmakros ausgehend von der Datenblattansicht der Tabelle *Arbeitnehmer*

Das fertige benannte Datenmakro zeigt die Abbildung 3.13. Wie Sie sehen, müssen zu Beginn die vier Übergabeparameter spezifiziert werden. Durch Klick auf den Link *Parameter erstellen* wird jeweils ein neuer Parameter hinzugefügt (gelöscht werden kann durch Klick auf das schwarze Kreuzchen rechts).

Abbildung 3.13 Programmierung des benannten Datenmakros im Makro-Editor

Es folgen einige Erklärungen zu den einzelnen Aktionen:

FürJedenDatensatz

Diese Aktion durchläuft ausnahmslos alle Datensätze der *Arbeitnehmer*-Tabelle, eine Bedingung gibt es deshalb nicht (leerer Eintrag). Der Zugriff auf jeden einzelnen Datensatz wird durch den Alias *AN* abgekürzt.

```
Für jeden Datensatz in Arbeitnehmer
        Bedingung
           Alias AN
```

DatensatzBearbeiten

Bei dieser Aktion kann der Alias entfallen. Eingebettet ist die *FestlegenFeld*-Aktion, in welcher dem *Netto*-Feld der aus dem *Brutto*-Feld und den Übergabeparametern *av*, *kv*, *rv* und *pv* errechnete Wert zugewiesen wird:

```
    FestlegenFeld
       Name Netto
      Wert = [AN].[Brutto]*(1-[av]-[kv]-[rv]-[pv])
```

Wir belassen es beim standardmäßig vorgegebenen Namen *Datenmakro1*, können aber bei Bedarf den entsprechenden Menübefehl verwenden, um es aufgabenbezogen umzubenennen.

Schließen Sie den Makro-Editor und speichern Sie die Tabelle.

Der Aufruf erfolgt von einem weiteren Datenmakro, welches wir im Folgenden erstellen werden.

Aufruf des benannten Datenmakros

Wechseln Sie zur Tabelle *Abzüge* (Datenblatt- oder Entwurfsansicht ist egal) und fügen Sie ein ereignisgesteuertes Datenmakro hinzu. Die Vorgehensweise wurde bereits ausführlich im Vorgängerbeispiel beschrieben, diesmal wählen wir aber das Tabellenereignis *Nach Aktualisierung*.

Die Namen aller in der Datenbank vorhandenen benannten Datenmakros werden aufgeklappt. Wie Sie sehen setzt sich der Namen (hier *Arbeitnehmer.Datenmakro1*)aus zwei Teilen zusammen, dem Namen der Tabelle, in welcher das Datenmakro gespeichert ist, und dem eigentlichen Namen.

AusführenDatenmakro

Wir benötigen nur eine einzige Aktion, nämlich *AusführenDatenmakro* (siehe Abbildung 3.14)[1]. Den vier zu übergebenen Parametern müssen die Werte der Felder *AV*, *KV*, *RV* und *PV* der Tabelle *Abzüge* zugewiesen werden.

[1] Die Aktion *AusführenDatenmakro* steht nur für Nachfolgeereignisse (*Nach Aktualisierung*, *Nach Einfügung*, *Nach Löschung*) zur Verfügung.

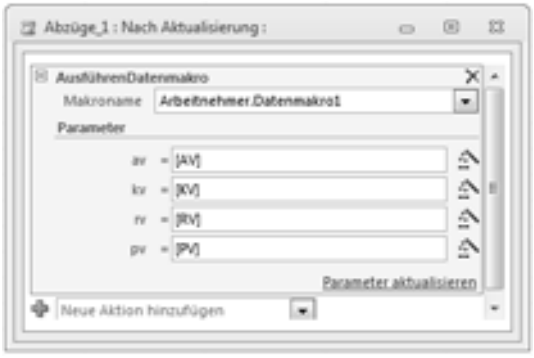

Abbildung 3.14 Programmierung der Aktion *AusführenDatenmakro* im Makro-Editor

Schließen Sie den Makro-Editor und speichern Sie die Tabelle.

Test

Um das Zusammenspiel beider Datenmakros zu überprüfen, öffnen Sie am besten gleichzeitig die Tabellen *Arbeitnehmer* und *Abzüge* in der Datenblattansicht, sodass Sie beide Tabelleninhalte im Blick haben.

Führen Sie jetzt Änderungen der Beitragssätze in der Tabelle *Abzüge* aus, so werden nach dem Abspeichern der Änderungen schlagartig alle Nettogehälter in der Tabelle *Arbeitnehmer* korrigiert (siehe Abbildung 3.15).

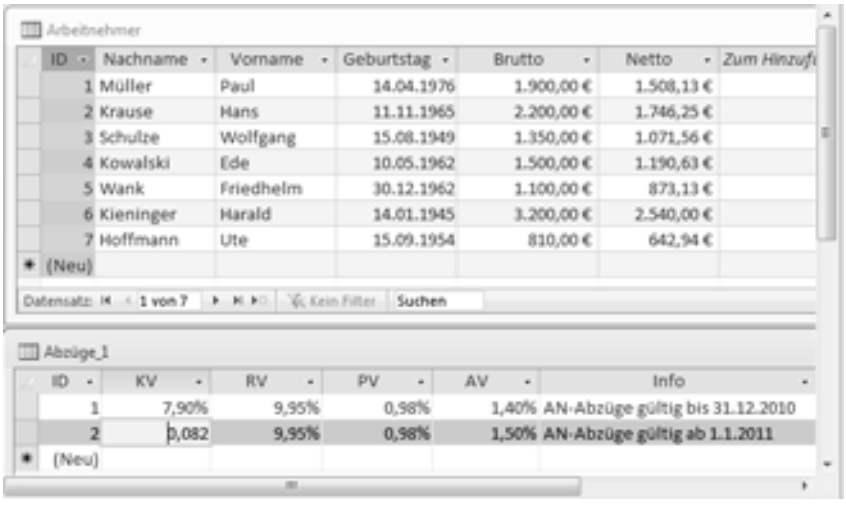

Abbildung 3.15 Änderungen in der Tabelle *Abzüge* wirken sich auf die *Netto*-Spalte in der Tabelle *Arbeitnehmer* aus.

Bemerkungen

- Falls die Änderungen nicht sofort sichtbar werden, müssen Sie zunächst der Tabelle *Arbeitnehmer* den Fokus geben
- Das folgende Beispiel zeigt, wie Sie das benannte Datenmakro auch durch Klick auf eine Befehlsschaltfläche starten können

Per VBA auf ein benanntes Datenmakro zugreifen

DoCmd-Objekt: *RunDataMacro*-, *SetParameter*-Methode; *Replace*-Methode

Die für den Informationsaustausch zwischen VBA-Code und Makros äußerst hilfreiche *TempVars*-Auflistung steht bislang leider nur für UI-Makros und nicht für Datenmakros zur Verfügung. Lediglich die *RunDataMacro*-Methode des *DoCmd*-Objekts bietet uns eine Möglichkeit, per VBA ein benanntes Datenmakro zu starten.

Ziel des vorliegenden Beispiel soll es sein, das im Vorgängerbeispiel entwickelte Datenmakro per Klick auf eine Schaltfläche zu starten.

Oberfläche

Diesmal werden wir etwas mehr Aufwand als in den beiden Vorgängerbeispielen betreiben und den Tabellen *Arbeitnehmer* und *Abzüge* eigene Benutzerschnittstellen spendieren (siehe Abbildung 3.16).

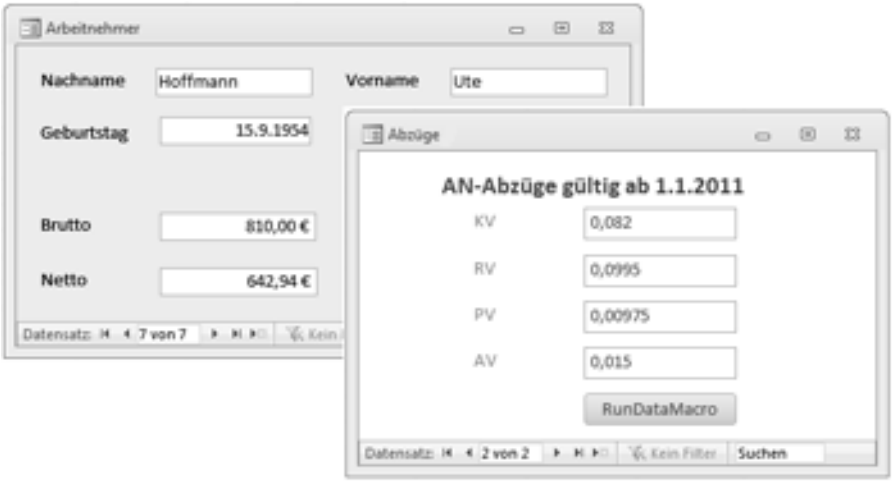

Abbildung 3.16 Bedienoberflächen für die Tabellen *Arbeitnehmer* und *Abzüge*

Quelltext

Das Ereignis *Beim Klicken* der Befehlsschaltfläche *RunDateMacro* hinterlegen wir mit folgendem VBA-Code:

```
Private Sub Befehl17_Click()
   DoCmd.SetParameter "av", Replace(txtAV, ",", ".")
   DoCmd.SetParameter "kv", Replace(txtKV, ",", ".")
   DoCmd.SetParameter "rv", Replace(txtRV, ",", ".")
   DoCmd.SetParameter "pv", Replace(txtPV, ",", ".")

   DoCmd.RunDataMacro "Arbeitnehmer.Datenmakro1"

End Sub
```

W. Doberenz, Th. Gewinnus: Microsoft Access 2010 - Programmierung;

Dazu einige Erklärungen:

- Bevor das Datenmakro aufgerufen werden kann, muss es mittels *SetParameter*-Methode mit den vier erforderlichen Parametern versorgt werden, welche hier direkt den Textfeldern des *Abzüge*-Formulars entnommen werden
- Leider können die Parameterwerte mit dem Komma als Dezimaltrennzeichen nichts anfangen (Fehlermeldung), sodass mit einem Workaround (hier mittels *Replace*-Methode) die Umwandlung in einen Dezimalpunkt erfolgen muss

Test

Geben Sie in die Tabelle *Abzüge* einen neuen Datensatz ein oder ändern Sie einen vorhandenen. Im Unterschied zum Vorgängerrezept erfolgt die Aktualisierung der Nettogehälter in der Tabelle *Arbeitnehmer* nicht sofort, sondern erst nach dem Klick auf die Schaltfläche »RunDataMacro«.

HINWEIS Falls die Änderungen nicht sofort sichtbar werden, sollten Sie zunächst das *Arbeitnehmer*-Formular aktivieren.

Bemerkungen

- Vor dem Test müssen Sie das aus dem Vorgängerbeispiel in der *Abzüge*-Tabelle eventuell noch vorhandene ereignisgesteuerte Datenmakro entfernen
- Für den Informationsaustausch zwischen VBA-Code und Datenmakros eignen sich zur Not auch die Werte einer extra angelegten Hilfstabelle (Zugriff z.B. per DAO)

Änderungen von Tabelleninhalten protokollieren

Nach Aktualisierung-Ereignis; *DatensatzErstellen*-, *FestlegenFeld*-, *Wenn*-Aktion; *Alt*-Objekt; *Jetzt*-Funktion;

In diesem Beispiel wollen wir alle Änderungen der Bruttogehälter der *Arbeitnehmer*-Tabelle in einer extra Tabelle protokollieren. Dafür eignet sich das Nachfolgeereignisse *Nach Aktualisierung*, welches uns (ebenso wie die Tabellenereignisse *Nach Löschung* und *Nach Einfügung*) die Möglichkeit gibt, Datenänderungen innerhalb einer Tabelle zur verfolgen.

Entwurf des Logbuchs

Bevor wir mit der Programmierung beginnen können, müssen wir zunächst eine neue Tabelle *Gehaltsänderungen* erstellen, welche uns quasi als »Logbuch« für die vorgenommenen Änderungen dienen soll (siehe Abbildung).


Gehaltsänderungen

Feldname	Felddatentyp	Beschreibung
ID	AutoWert	
Nachname	Text	
PersID	Zahl	ID aus Arbeitnehmer-Tabelle
Datum	Datum/Uhrzeit	Aktualisierungsdatum
AltesGehalt	Währung	Brutto vor Aktualisierung
NeuesGehalt	Währung	Brutto nach Aktualisierung

Abbildung 3.17 Entwurfsansicht der Tabelle *Gehaltsänderungen*

W. Doberenz, Th. Gewinnus: Microsoft Access 2010 - Programmierung;

Erstellen des Datenmakros

Wir fügen der Tabelle *Arbeitnehmer* ein ereignisgesteuertes Datenmakro hinzu, das mit dem Ereignis *Nach Aktualisierung* verknüpft ist.

Hier der fertige Code, wie Sie ihn bequem im Makro-Editor erstellen:

```
Wenn  Aktualisiert("Brutto") Dann
       Datensatz erstellen in  Gehaltsänderungen
                     Alias
   FestlegenFeld
                       Name   Nachname
                       Wert   = [Arbeitnehmer].[Nachname]
   FestlegenFeld
                       Name   PersID
                       Wert   = [Arbeitnehmer].[ID]
   FestlegenFeld
                       Name   NeuesGehalt
                       Wert   = [Arbeitnehmer].[Brutto]
   FestlegenFeld
                       Name   AltesGehalt
                       Wert   = [Alt].[Brutto]
   FestlegenFeld
                       Name   Datum
                       Wert   = Jetzt()
Ende Wenn
```

Nun einige Erklärungen zu den einzelnen Aktionen:

Wenn

Diese Bedingung gewährleistet, dass das Datenmakro nur dann gestartet wird, wenn Änderungen am *Brutto*-Gehalt eines Arbeitnehmers vorgenommen und abgespeichert wurden.

DatensatzErstellen

Diese Aktion gibt es nur für Nachfolgeereignisse. Es wird ein neuer Datensatz in die Tabelle *Gehaltsänderungen* eingefügt. Die Werte der einzelnen Felder werden anschließend mit den diversen *FestlegenFeld*-Aktionen zugewiesen.

- Etwas aus der Rolle fällt der Feldname *Alt*, denn ein Feld mit dieser Bezeichnung findet sich nicht in der Tabelle *Gehaltsänderungen*, sondern wird vom Makro-Editor als temporäres Objekt automatisch zur Verfügung gestellt
- Den Wert *Jetzt()* für das aktuelle Datum können Sie mit Hilfe des Ausdrucks-Generators zuweisen

Test

Öffnen Sie die Tabelle *Arbeitnehmer* in der Datenblattansicht und verändern Sie einige Bruttogehälter. Nach dem Abspeichern erscheint jede Änderung als neuer Datensatz in der Tabelle *Gehaltsänderungen* (siehe Abbildung 3.18).

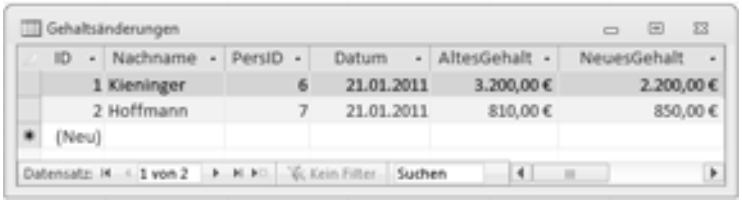

Abbildung 3.18 Datenblattansicht der Tabelle *Gehaltsänderungen*