Access [basics] 9/2011
www.access-basics.de
Das Access-Magazin für alle, die schnell von 0 auf 100 wollen

Editorial

Wollen Sie Daten wie Postleitzahlen, Orte, Länder und so weiter immer wieder komplett eintippen müssen? Nein! Sollen solche Daten deshalb in einer Lookup-Tabelle landen, damit Sie diese per Kombinationsfeld auswählen können? Nein! Also zeigen wir Ihnen im ersten Artikel der neuen Ausgabe, wie Sie ganz einfach auf die bereits einmal eingegebenen Werte eines Feldes zugreifen, indem Sie die Datensatzherkunft eines Kombinationsfeldes mit den entsprechenden Daten füllen. Sie geben dann nur noch die ersten paar Zeichen ein und betätigen die Tabulator-Taste, sobald die Autovervollständigen-Funktion den gesuchten Eintrag anzeigt! Wie dies genau funktioniert, zeigt unser Beitrag **Kombinationsfeld als Eingabehilfe für Textfelder** ab Seite 2.

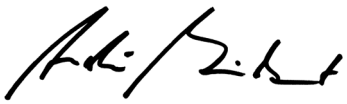
In einem weiteren Artikel stellen wir Ihnen die Optionsgruppe als Eingabesteuerelement für gebundene und ungebundene Daten vor. Damit erlauben Sie dem Benutzer, schnell per Mausklick eine von mehreren Auswahlmöglichkeiten zu selektieren, statt dafür erst ein Kombinationsfeld zu öffnen oder den Wert sogar von Hand eintippen zu müssen. Alles zu den beiden Steuerelementen Optionsgruppe und Optionsfeld lesen Sie im Artikel **Optionsgruppen und Optionsfelder** ab Seite 5.

Wenn Sie noch nicht so viele Berichte erstellt haben, sollten Sie unbedingt einen Blick in den Artikel **Berichtsbereiche** ab Seite 11 werfen. Dort lernen Sie die einzelnen Bereiche von Berichten und deren Funktion kennen.

Auch im Artikel **Sortieren in Berichten** ab Seite 14 dreht es sich um Berichte. Hier erfahren Sie, dass in Tabellen und Abfragen festgelegte Sortierungen nur sporadisch in Berichten funktionieren. Dort gibt es nämlich ein eigenes Werkzeug zum Festlegen der Sortierung.

Und schließlich liefern wir im Artikel **Tipps und Tricks** ab Seite 17 eine Reihe interessanter Informationen. Dort erfahren Sie etwa, wie Sie das Standardverzeichnis einer Datenbank einstellen, Objekte wie Tabellen, Abfragen, Formulare und Berichte im Datenbankfenster beziehungsweise Navigationsbereich ausblenden und somit vor den Blicken der Benutzer schützen und wie Sie diese temporär wieder sichtbar machen. Außerdem zeigen wir Ihnen, wie Sie Datum und Uhrzeit schnell per Tastenkombination einfügen und wie Sie den Wert eines Feldes aus dem vorherigen Datensatz auch im aktuellen Datensatz einfügen.

Und nun: Viel Spaß beim Lesen!



Ihr André Minhorst

Impressum

Access [basics] wird monatlich herausgegeben von:
André Minhorst | Fachverlag für Softwareentwicklung | Borkhofer Straße 17 | 47137 Duisburg

Redaktion:
André Minhorst (V.i.S.d.P) | Telefon: 0203/4495577 | E-Mail: info@access-basics.de | Internet: www.access-basics.de

Geschäftsführung, Herstellung, Text- und Schlussredaktion, Layout von Magazin und Webseite: André Minhorst

Autor: André Minhorst

ISSN: 2190-8761

ISSN: 2190-8761

Kombinationsfeld als Eingabehilfe für Textfelder

Kombinationsfelder haben Sie in Access [basics] bisher als Möglichkeit kennengelernt, Verknüpfungen zu Daten aus anderen Tabellen herzustellen. Das ist auch der meistgenutzte Anwendungszweck von Kombinationsfeldern. Wenn Sie den Benutzern Ihrer Anwendung jedoch das Leben erleichtern möchten, können Sie es auch an anderer Stelle einsetzen – beispielsweise, um bereits eingegebene Werte eines Feldes zur Auswahl anzubieten. Das ist vor allem hilfreich, wenn das Feld Werte enthält, die mehrfach vorkommen, aber dennoch nicht in eine Lookup-Tabelle ausgegliedert wurden.

Beispieldatenbank

Die Beispiele dieses Artikels finden Sie in der Datenbank **1109_KombinationsfeldAlsEingabehilfe.mdb**.

Hilfe bei PLZ und Ort

Ein Beispiel für diesen Fall sind PLZ und Ort bei Adressdaten. Normalerweise gibt der Benutzer Postleitzahlen und Orte immer komplett ein – ohne jegliche Eingabehilfe. Was aber, wenn die einzugebenden Adressen, beispielsweise von Kunden, immer wieder gleich sind, weil die Kunden alle aus dem direkten Umkreis stammen? Da wäre es doch sinnvoll, dem Benutzer die eine oder andere Erleichterung zu bieten. In diesem Fall verwenden Sie einfach ein Kombinationsfeld statt eines Textfeldes, dass sowohl die Eingabe neuer Werte als auch die Auswahl bereits vorhandener Werte ermöglicht – und somit auch die automatische Ergänzung beim Eingeben der ersten paar Zeichen einer PLZ oder eines Ortes.

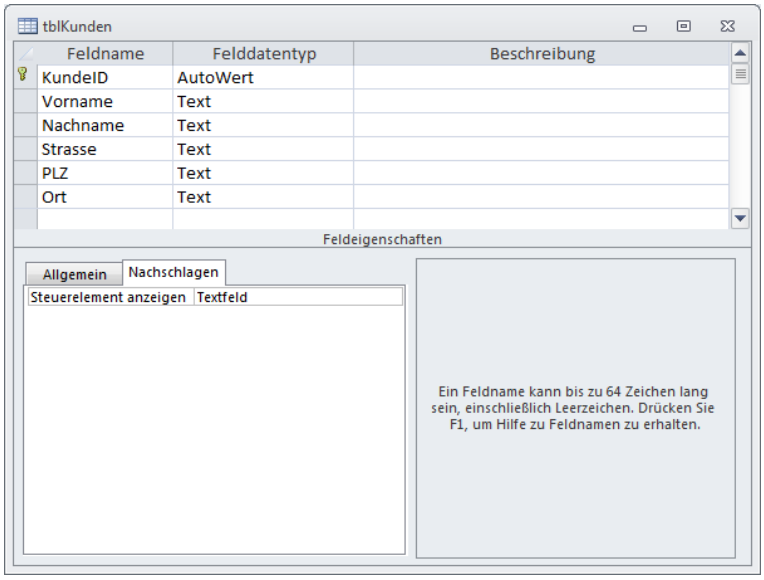


Bild 1: Beispieltabelle **tblKunden**

Als Beispiel dienen die Felder **PLZ** und **Ort** der Tabelle **tblKunden** der Beispieldatenbank (siehe Bild 1).

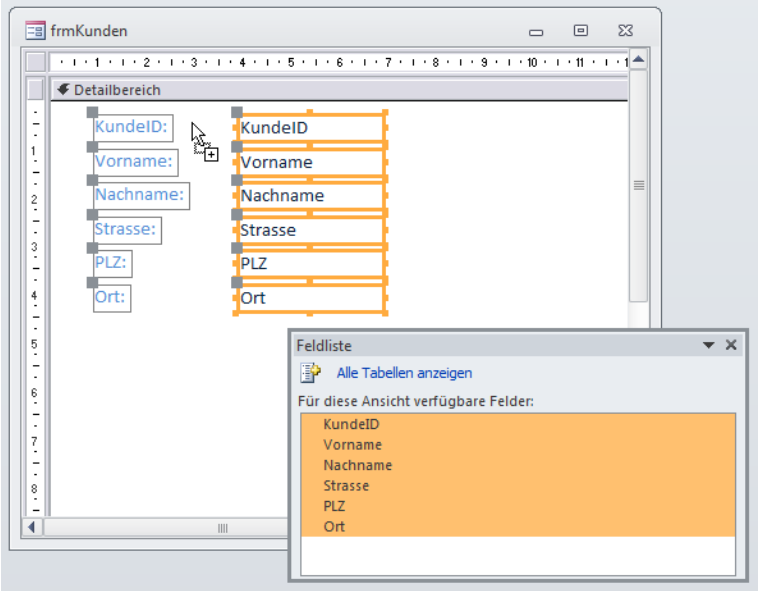


Bild 2: Hinzufügen der Felder der Tabelle **tblKunden** zum Formular **frmKunden**

Das Formular **frmKunden** verwendet diese Tabelle als Datenherkunft. Ziehen Sie in der Entwurfsansicht des Formulars einfach alle Felder aus der Feldliste in den Detailbereich. Sie erhalten dort zunächst eine Reihe Textfelder, auch für die beiden Felder **PLZ** und **Ort** (siehe Bild 2). Das ist kein Wunder, denn diese Felder sind ja im Entwurf der Tabelle nicht anders definiert worden. Damit sich dies ändert, klicken Sie zunächst mit der rechten Maustaste auf das Textfeld **PLZ**. Wählen Sie aus dem Kontextmenü den Eintrag **Ändern zu|Kombinationsfeld** aus. Führen Sie den gleichen Schritt für das Textfeld **Ort** aus. Benennen Sie die beiden Steuerelemente dann in **cboPLZ** und **cboOrt** um.

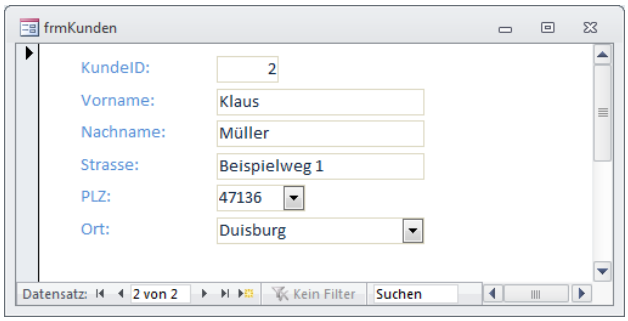


Bild 3: Eintragen von Daten in Textfelder, die in Kombinationsfelder umgewandelt wurden

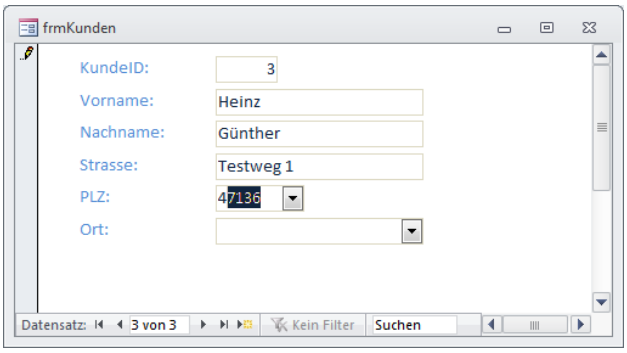


Bild 5: Kombinationsfeld mit automatischer Ergänzung

Nach dem Wechsel in die Formularansicht tut sich nicht viel: Sie können ganz gewöhnlich Daten in die Kombinationsfelder eintragen (siehe Bild 3). Das Aufklappen der Kombinationsfelder bietet keine weiteren Auswahlmöglichkeiten an.

Das ändern wir jetzt, und zwar zunächst für die Postleitzahl. Wechseln Sie zurück in die Entwurfsansicht des Formulars und klicken Sie auf das Steuerelement **cboPLZ**. Klicken Sie in die Eigenschaft **Datensatzherkunft** und dann auf die nun erscheinende Schaltfläche mit den drei Punkten (...). Sie öffnen damit die Abfrage-Entwurfsansicht und können nun eine Abfrage als Datensatzherkunft für das Kombinationsfeld einstellen.

Wählen Sie im Dialog **Tabelle anzeigen** den Eintrag **tblKunden** aus und schließen Sie diesen Dialog. Ziehen Sie das Feld **PLZ** aus der Tabelle **tblKunden** in das Entwurfsraster des Abfrageentwurfs und stellen Sie die **Sortierung** auf **Aufsteigend** ein. Wenn Sie nun in die Datenblattansicht wechseln, zeigt diese bereits alle Postleitzahlen der Tabelle an – allerdings auch doppelte. Damit dies nicht geschieht, stellen Sie die Eigenschaft **Keine Duplikate** der Tabelle auf **Ja** ein (siehe Bild 4). Sollte das Eigenschaftsfenster gerade nur Feldeigenschaften anzeigen, ist aktuell ein Feld markiert – vermutlich, weil Sie dieses gerade bearbeitet haben. Klicken Sie dann einfach in den grau hinterlegten Bereich oben im Abfrageentwurf, um die Abfrage-Eigenschaften einzublenden.

Schließen Sie den Abfrageentwurf und kehren Sie zum Formular zurück. Wechseln Sie zur Formularansicht und geben Sie für einen neuen Kunden die Postleitzahl ein, die testweise genauso beginnt wie eine bereits vorhandene Postleitzahl.

Und das Ergebnis überzeugt: Das Kombinationsfeld zeigt gleich den ersten Treffer der bisher eingegebenen PLZs an, das mit den bereits eingegebenen Zeichen übereinstimmt (siehe Bild 5). Der Benutzer könnte nun direkt zum nächsten Feld springen und den aktuellen Wert so übernehmen. Sie haben aber noch mehr Möglichkeiten: Sie können zum Beispiel das Kombinationsfeld aufklappen und einen der bisher vergebenen Werte auswählen. Für Tastatur-Fans: Ein Kombinationsfeld öffnen Sie mit **F4**.

Automatische Ergänzung aktiv?

Das die automatische Ergänzung so funktioniert (wenn sie funktioniert), liegt an der Einstellung der Eigenschaft **Automatische Ergänzung** des Kombinationsfeldes. Wenn bei Ihnen nicht automatisch der nächste verfügbare Eintrag angezeigt wird, ist diese Eigenschaft möglicherweise auf den Wert **Nein** eingestellt. Ändern Sie den Wert in diesem Fall einfach auf **Ja**.

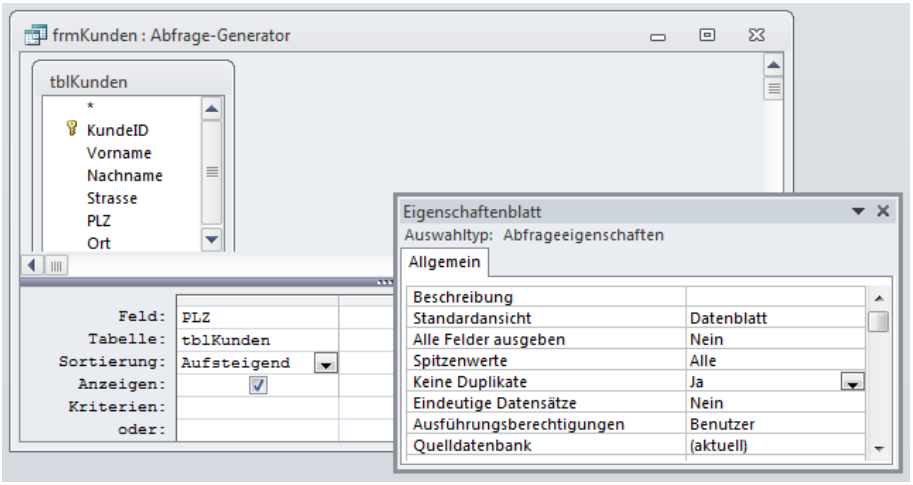


Bild 4: Einstellen der Datensatzherkunft des Kombinationsfeldes

Automatisch aufklappen?

Wenn Sie möchten, können Sie das Kombinationsfeld so programmieren, dass es beim Fokuserhalt direkt aufgeklappt wird. Dazu legen Sie in der Entwurfsansicht des Formulars eine Prozedur an, die durch das Ereignis **Bei Fokuserhalt** des Kombinationsfeldes **cboPLZ** ausgelöst wird. Markieren Sie dieses Steuerelement, wählen Sie im Eigenschaftsfeld auf die Eigenschaft **Bei Fokuserhalt** den Eintrag **[Ereignisprozedur]** aus und klicken Sie dann auf die Schaltfläche mit den drei Punkten. Die öffnet den VBA-Editor, der gleich den passenden Prozedurrumpf anzeigt. Diesen ergänzen Sie wie folgt:

```
Private Sub cboPLZ_GotFocus()
    Me!cboPLZ.Dropdown
End Sub
```

Wechseln Sie nun zurück in die Formularansicht des Formulars, wird das Kombinationsfeld beim Anklicken oder auch beim Wechseln per Tabulator-Taste direkt aufgeklappt. Dies mag hinderlich sein, wenn Sie gerade die bereits gefüllten Felder eines Datensatzes durchlaufen. Also fügen wir eine Bedingung hinzu, die das Kombinationsfeld nur aufklappt, wenn dieses noch leer ist:

```
Private Sub cboPLZ_GotFocus()
    If IsNull(Me!cboPLZ) Then
        Me!cboPLZ.Dropdown
    End If
End Sub
```

Kombinationsfeld aktualisieren

Wenn Sie auf diese Weise ein paar Datensätze anlegen, fällt Ihnen schnell auf, dass die neu eingegebenen Postleitzahlen gar nicht im Kombinationsfeld **cboPLZ** auftauchen. Dies sollte aber zumindest beim Wechsel zu einem neuen Datensatz der Fall sein. Das Ereignis, das beim Wechsel des Datensatzes im Formular ausgelöst wird, heißt **Beim Anzeigen**.

Legen Sie auch für dieses Ereignis eine Ereignisprozedur an. Diese enthält nur eine einzige Anweisung:

```
Private Sub Form_Current()
    Me!cboPLZ.Requery
End Sub
```

Diese Anweisung sorgt für das erneute Abfragen der Datensatzherkunft des Kombinationsfeldes. Allerdings stellt sich die Frage, ob dies wirklich bei jedem Datensatzwechsel nötig und sinnvoll ist. Immerhin wird die als Datensatzherkunft angegebene Abfrage jedes Mal neu ausgeführt!

Wie wäre es denn, wenn der Inhalt des Kombinationsfeldes nur nach Eingabe eins neuen Wertes aktualisiert würde? Eine passende Ereigniseigenschaft wäre **Nach Aktualisierung** des Steuerelements **cboPLZ**. Verwerfen wir doch die vorherige Prozedur und verwenden stattdessen diese hier:

```
Private Sub cboPLZ_AfterUpdate()
    Me!cboPLZ.Requery
End Sub
```

Nun wird die Datensatzherkunft nach jeder Änderung des Wertes des Feldes **PLZ** eines der Datensätze aktualisiert. Dummerweise macht sich dies beim Aufklappen des Kombinationsfeldes gar nicht bemerkbar: Frisch eingegebene Werte tauchen dort nämlich nicht auf.

Der Grund ist ganz einfach: Das Kombinationsfeld zeigt nur die in der Tabelle gespeicherten Postleitzahlen an. Durch das Aktualisieren des Kombinationsfeldes wird der Datensatz aber noch gar nicht gespeichert, sodass die Änderung noch nicht bis zur Tabelle durchgeschlagen ist!

Also wählen wir noch eine andere Methode, nämlich das Ereignis **Vor Aktualisierung** des Formulars. Dieses wird, so sollte man annehmen, vor dem Speichern der Daten in der Tabelle durchgeführt.

Dies ist jedoch falsch: Der Aufruf der **Requery**-Methode des Kombinationsfeldes führt hier zielsicher zur Anzeige auch eines eventuell neu eingegebenen Eintrags.

Damit das Kombinationsfeld nur aktualisiert wird, wenn auch die PLZ im aktuellen Datensatz geändert wurde, vergleicht die folgende Prozedur auch noch den alten und den aktuellen Wert des Kombinationsfeldes. Den aktuellen Wert liefert dabei die Eigenschaft **Value**, den alten Wert erhalten Sie mit der Eigenschaft **OldValue**.

```
Private Sub Form_BeforeUpdate(Cancel As Integer)
    If Not (Nz(Me!cboPLZ.Value) = _
            Nz(Me!cboPLZ.OldValue)) Then
        Me!cboPLZ.Requery
    End If
End Sub
```

Optionsgruppen und Optionsfelder

Wenn sich für ein Feld einer Tabelle mehrere Werte einstellen lassen, liegen diese Werte meist in einer Lookup-Tabelle und werden per Nachschlagefeld beziehungsweise per Kombinationsfeld ausgewählt. Manchmal ist die Anzahl der verfügbaren Werte jedoch begrenzt und wird vor allem nicht erweitert. Und wenn die Werte dann auch noch übersichtlich im Formular dargestellt werden sollen, ist eine Optionsgruppe mit entsprechenden Optionsfeldern die richtige Wahl.

Beispieldatenbank

Die Beispiele dieses Artikels finden Sie in der Datenbank **1109_Optionsgruppen.mdb**.

Wert auswählen

Kombinationsfelder, Listenfelder, Kontrollkästchen und Optionsgruppen sind allesamt Steuerelemente, mit denen Sie mindestens einen von mehreren Werten auswählen können – unabhängig davon, ob das Steuerelement an ein Feld einer Datenherkunft gebunden ist oder nicht. Während Kombinationsfelder meist zur Auswahl eines Wertes aus einer verknüpften Tabellen dienen, Sie mit Listenfeldern einen oder mehrere Werte markieren können und Kontrollkästchen meist Ja/Nein-Werte markieren, können Sie mit einer Optionsgruppe und den enthaltenen Optionsfeldern einen Wert aus einer limitierten Anzahl von Werten auslesen.

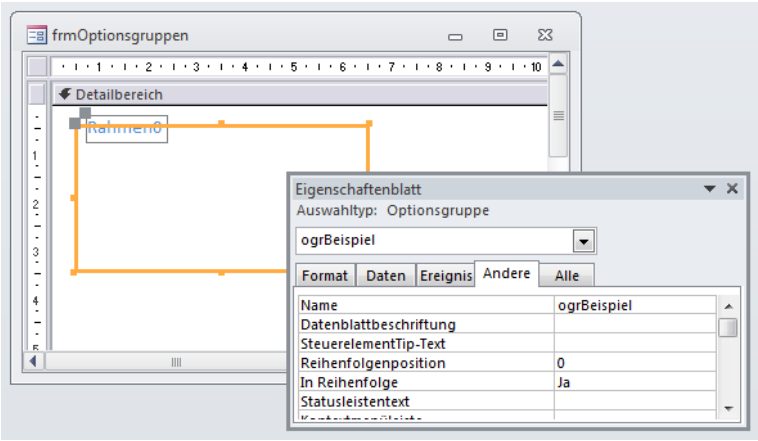


Bild 1: Einstellen des Namens einer frisch angelegten Optionsgruppe

Da die Optionsfelder einer Optionsgruppe statisch im Entwurf eines Formulars angelegt werden, kommen Sie nicht für die Auswahl von Daten infrage, die sich im Laufe der Zeit ändern können, weil Daten hinzukommen oder wegfallen. Sie dienen eher der Festlegung von Werten wie Klein/Mittel/Groß, von Farben, alternativ zum Kontrollkästchen für Ja/Nein-Werte und Artverwandte oder auch für andere Eigenschaften, für die zwei oder mehr Werte zur Auswahl stehen. Eine Optionsgruppe kann an ein Feld einer Tabelle gebunden sein oder auch für sich allein stehen – etwa, um ein zu öffnendes Formular oder einen Bericht auszuwählen, dass danach per Schaltfläche geöffnet wird.

Ungebundene Optionsgruppen

Im ersten Beispiel legen wir einfach eine Optionsgruppe mit drei Optionsfeldern an, um die wichtigsten Eigenschaften dieser Steuerelemente kennenzulernen. Erstellen Sie dazu ein neues Formular in der Entwurfsansicht und fügen Sie diesem zunächst eine Optionsgruppe hinzu. Dazu klicken Sie in der Liste der Steuerelemente auf das Optionsgruppe-Steuerelement und ziehen dann im Entwurf des Formulars einen Rahmen mit der gewünschten Größe auf.

Die Optionsgruppe hat zunächst eine wichtige Eigenschaft: den Namen (siehe Bild 1). Diesen stellen Sie auf einen aussagekräftigen Wert ein (in diesem Fall **ogrBeispiel**). Der Name ist nicht zu verwechseln mit der Beschriftung: Diese kommt mit einem angehefteten Bezeichnungsfeld. Auch diese sollten Sie direkt anpassen, indem Sie zuerst auf das Bezeichnungsfeld klicken und dann seine Eigenschaft **Beschriftung** ändern. Den Namen des Bezeichnungsfeldes müssen Sie indes nicht anpassen, Sie werden im Normalfall nicht per VBA darauf zugreifen.

Optionsfelder hinzufügen

Wenn Sie nun in der Liste der Steuerelemente auf ein Optionsfeld klicken und dann mit der Maus über die Optionsgruppe fahren, wissen Sie, wann Sie loslassen müssen – nämlich dann, wenn der Hintergrund der Optionsgruppe schwarz wird (siehe Bild 2). Achtung: Wenn Sie das Optionsfeld irgendwo außerhalb der Optionsgruppe platzieren und es später in die Optionsgruppe ziehen, wird es nicht mit der Optionsgruppe verknüpft, sondern steht für sich allein! Die

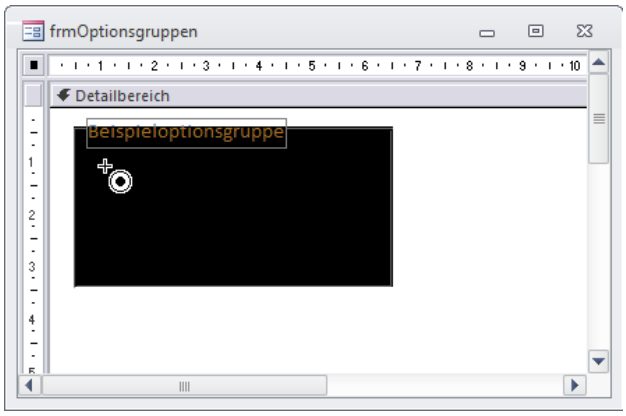


Bild 2: Hinzufügen eines Optionsfeldes zu einer Optionsgruppe

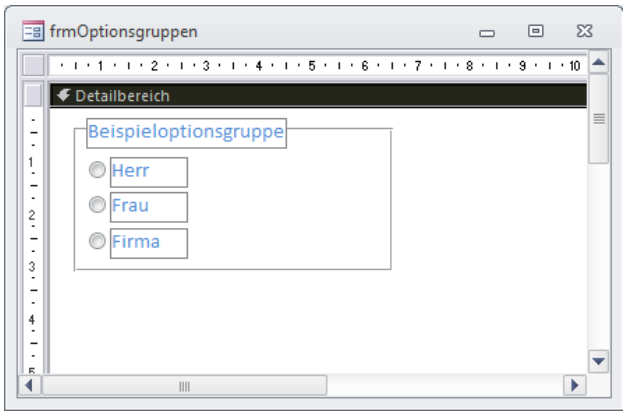


Bild 3: Hinzufügen weiterer Optionsfelder

einzige Möglichkeit, ein solches Optionsfeld noch zur Optionsgruppe hinzuzufügen, ist das Ausschneiden (zum Beispiel mit **Strg + X**), Markieren der Optionsgruppe und Einfügen (**Strg + V**). Fügen Sie auf diese Weise drei Optionsfelder zur Optionsgruppe hinzu. Die Optionsfelder bestehen wiederum aus dem eigentlichen Steuerelement und einem daran gebundenen Bezeichnungsfeld.

Stellen Sie die Beschriftung der drei Optionsfelder auf **Herr**, **Frau** und **Firma** ein (siehe Bild 3). Die Beschriftungen sind, bezogen auf die Funktion der angelegten Steuerelemente, Schall und Rauch. Sie dienen lediglich dazu, dem Benutzer zu zeigen, welche Bedeutung die einzelnen Optionen haben. Tatsächlich wichtig ist der Wert, den Sie für die Eigenschaft **Wert** der drei Optionsfelder vergeben. Wenn Sie die Optionsfelder alle einzeln über die Steuerelement-Liste angelegt haben, hat Access die Eigenschaft **Wert** automatisch vorbelegt, und zwar mit den Werten **1**, **2** und **3**. Bild 4 zeigt die Vorbelegung dieser Eigenschaft mit dem Wert **1**. Sie können die Optionsfelder auch hinzufügen, indem Sie das erste auf herkömmlichem Wege über die Steuerelement-Liste beziehen und dieses dann entsprechend der benötigten Anzahl von Optionsfeldern kopieren. Diese Vorgehensweise zaubert zwar schnell einige Optionsfelder in die Optionsgruppe, übernimmt aber leider auch den Wert des Originaloptionsfeldes. Das heißt, dass Sie die Werte der übrigen Felder später manuell anpassen müssen. Das ist natürlich kein Problem, wenn Sie ohnehin andere Werte als **1**, **2**, **3** und so weiter benötigen.

Standardwert festlegen

Beim Öffnen des Formulars soll üblicherweise gleich eine Option ausgewählt sein. Dies erreichen Sie, indem Sie für die Eigenschaft **Standardwert** des Optionsgruppen-Steuerelements einen der für die Optionsfelder vergebenen Werte eintragen. Im Beispiel aus Bild 5 handelt es sich um das oberste Optionsfeld mit dem Wert **1**. Wenn Sie keinen Standardwert auswählen und der Benutzer keine Option auswählt, hat die Optionsgruppe den Wert **Null**. Da dies in den wenigsten Fällen erwünscht, sollten Sie dem mit einer entsprechenden Validierung oder der Vorgabe eines Standardwerts vorbeugen.

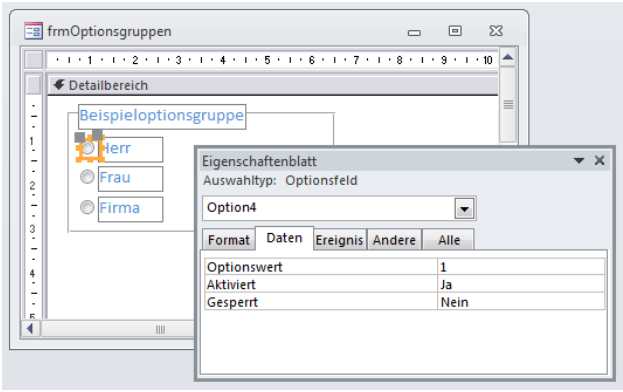


Bild 4: Wert eines der Optionsfelder

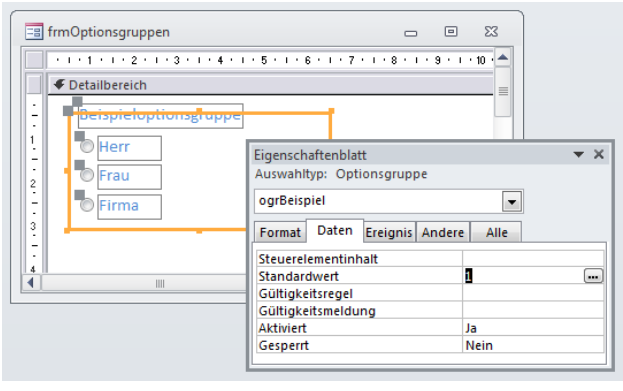


Bild 5: Standardwert der Optionsgruppe festlegen

Wert auslesen

Auf den Wert einer Optionsgruppe greifen Sie über die **Wert**-Eigenschaft der Optionsgruppe zu. Wenn Sie testweise den aktuellen Wert der Optionsgruppe sehen möchten, fügen Sie ein Textfeld zum Formular hinzu und stellen seine Eigenschaft **Steuerelementinhalt** auf den folgenden Wert ein:

```
=[ogrBeispiel].[Wert]
```

Da die Eigenschaft **Wert** die Standardeigenschaft dieses Steuerelements ist, reicht auch der folgende Ausdruck:

```
=[ogrBeispiel]
```

Wenn Sie nun in die Formularansicht wechseln, zeigt das Textfeld direkt nach dem Auswählen jeweils den Wert des aktuell ausgewählten Optionsfeldes an (siehe Bild 6).

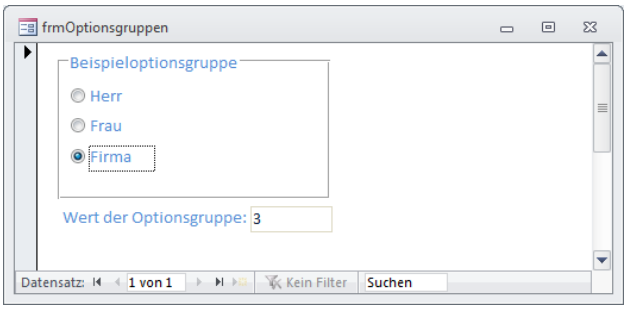


Bild 6: Ausgabe des aktuellen Werts einer Optionsgruppe

Per VBA auf eine Änderung reagieren

Wenn Sie beim Ändern der ausgewählten Option eine VBA-Prozedur starten möchten, die eine Aktion auf Basis der ausgewählten Option durchführt, ist das auch kein Problem. Markieren Sie die Optionsgruppe in der Entwurfsansicht des Formulars und tragen Sie für die Eigenschaft **Nach Aktualisierung** den Wert **[Ereignisprozedur]** ein. Klicken Sie dann auf die Schaltfläche mit den drei Punkten rechts neben der Eigenschaft und ergänzen Sie die nun im VBA-Editor erscheinende Prozedur wie folgt:

```
Private Sub ogrBeispiel_AfterUpdate()
    MsgBox "Der aktuelle Wert der Optionsgruppe ⤷
                lautet " & Me!ogrBeispiel & "."
End Sub
```

Der Wert der Optionsgruppe wird hier einfach durch das Referenzieren des entsprechenden Steuerelements über seinen Namen realisiert (**Me!ogrBeispiel**), der Wert in einem Meldungsfenster ausgeben. Wenn Sie lieber die für das Optionsfeld angegebene Beschriftung ausgeben möchten, kommen Sie um ein wenig Schreibarbeit nicht herum. Das Ergebnis sieht so aus:

```
Private Sub ogrBeispiel_AfterUpdate()
    Select Case ogrBeispiel
        Case 1
            MsgBox "Gewählter Wert: Herr"
        Case 2
            MsgBox "Gewählter Wert: Frau"
        Case 3
            MsgBox "Gewählter Wert: Firma"
    End Select
End Sub
```

Gebundene Optionsgruppen

Sie können Optionsgruppen auch an ein Feld der Datenherkunft eines Formulars binden. Die einzelnen Optionsfelder der Optionsgruppe repräsentieren dann die Werte, die das zugrunde liegende Feld annehmen kann. Im Beispiel wollen wir die Anrede eines Kunden auswählen. Das Formular binden Sie an die Tabelle **tblKunden** der Beispieldatenbank. Die Beispieldatenbank besitzt eine weitere Tabelle namens **tblAnreden**. Diese enthält die Lookupdaten für das Feld **AnredeID** der Tabelle **tblKunden**, die wir mit der Optionsgruppe abbilden wollen (siehe Bild 7). Die Tabelle **tblAnreden** enthält die drei Werte **Herr**, **Frau** und **Firma**, denen die Primärschlüsselwert **1**, **2** und **3** zugewiesen wurden. Diese werden gleich noch benötigt! Erstens soll natürlich in das Fremdschlüsselfeld **AnredeID** der Tabelle **tblKunden** einer der drei Werte **1**, **2** oder **3** eingetragen werden, um dem Kunden die entsprechende Anrede zuzuweisen. Zweitens müssen wir die Optionsfelder der Optionsgruppe mit genau diesen Werten ausstatten. Legen Sie also ein neues Formular an und legen Sie die Tabelle **tblKunden** als Datenherkunft fest. Ziehen Sie die drei Felder

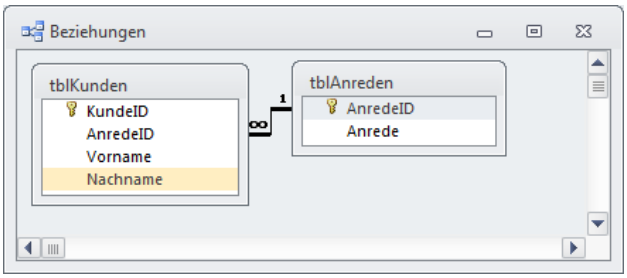


Bild 7: Beispieltabellen zur Demonstration einer gebundenen Optionsgruppe

KundeID, **Vorname** und **Nachname** in den Detailbereich des Formularentwurfs.

Gebundene Optionsgruppe hinzufügen

Das Anlegen der gebundenen Optionsgruppe führen Sie auf eine der folgenden Arten durch. Bei der ersten Methode fügen Sie zunächst eine Optionsgruppe zum Formular hinzu und stellen die Beschriftung auf einen Text wie etwa **Anrede** ein. Wählen Sie für die Eigenschaft **Steuerelementinhalt** der Optionsgruppe das Feld **AnredeID** aus und vergeben Sie den Namen **ogrAnredeID** für die Optionsgruppe. Ziehen Sie dann drei Optionsfelder in die Optionsgruppe und beschriften Sie diese mit **Herr**, **Frau** und **Firma**. Weisen Sie dann der Eigenschaft **Wert** die drei Zahlen **1**, **2** und **3** zu.

Nach dem Wechsel in die Formularansicht und dem Eingeben eines Datensatzes sieht die Optionsgruppe wie in Bild 8 aus. Wenn Sie die Tabelle tblKunden öffnen, finden Sie die per Optionsgruppe ausgewählten Zahlenwerte im Feld **AnredeID** wieder (siehe Bild 9). Wenn Sie durch die Datensätze blättern, stellen Sie außerdem fest, dass die Optionsgruppe jeweils das richtige Optionsfeld markiert.

Mit einer alternativen Vorgehensweise sparen Sie sich beim Anlegen einen Schritt. Dazu klicken Sie zunächst auf das Optionsgruppen-Steuerelement in der Steuerelementliste und ziehen dann das Feld, an welches die Optionsgruppe gebunden werden soll, in den Formularentwurf – in diesem Fall das Feld **AnredeID**. Das Steuerelement heißt dann gleich **AnredeID** und die Beschriftung erhält ebenfalls diesen Wert. Außerdem wird die Optionsgruppe dann direkt an das gleichnamige Feld gebunden. Die Optionsfelder müssen Sie allerdings weiterhin von Hand hinzufügen.

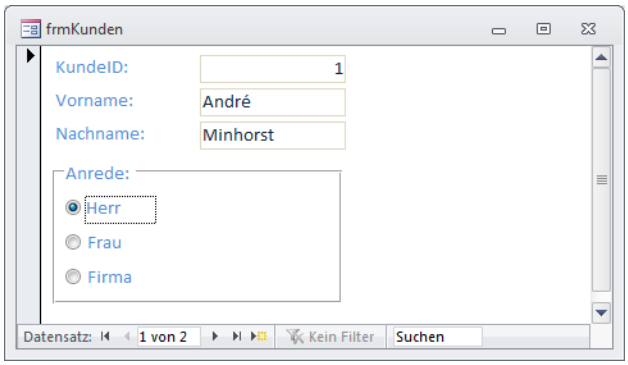


Bild 8: Optionsgruppe zum Auswählen der Anrede

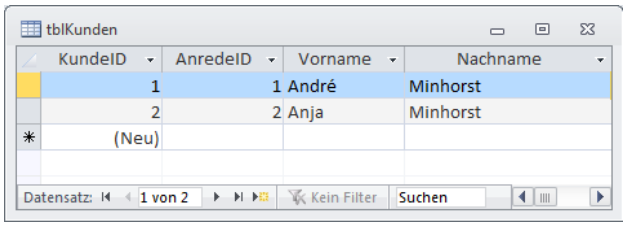


Bild 9: Die mit der Optionsgruppe ausgewählten Werte landen zuverlässig in der Tabelle.

Abhängige Optionsgruppen

Als Praxisbeispiel schauen wir uns noch die Funktionsweise zweier abhängiger Optionsgruppen an. Dazu fügen wir der Tabelle **tblKunden** zwei Zahlenfelder namens **KundenartID** und **ZahlungsartID** hinzu (siehe Bild 10). Da wir die Werte (**Privat** und **Geschäftlich** als Kundenart sowie **Banküberweisung**, **Bankeinzug**, **Per Nachname** und **Paypal**) sowieso nur in Form der Beschriftung der Optionsfelder sehen, legen wir der Einfachheit halber keine Lookup-Tabellen für diese beiden Felder an.

Fügen Sie für das Feld **KundenartID** eine Optionsgruppe namens **ogrKundenartID** hinzu (nicht vergessen, **KundenartID** als **Steuerelementinhalt** festzulegen) und statten Sie dieses mit zwei Optionsfeldern aus. Das erste erhält die Beschriftung **Privat** und den Wert **1**, die zweite die Beschriftung **Geschäftlich** und den Wert **2**.

Das Feld **ZahlungsartID** wird durch die Optionsgruppe **ogrZahlungsartID** repräsentiert. Es enthält vier Optionsfelder, die wie in Bild 11 angeordnet und von **1** bis **4** durchnummeriert sind.

Da die Optionsfelder der Optionsgruppe **ogrZahlungsartID** in Abhängigkeit des Wertes der Optionsgruppe ogrKundenartID aktiviert oder deaktiviert werden sollen, müssen Sie diese später per Code ansprechen. Damit der Code übersichtlich bleibt, benennen Sie daher die vier Optionsgruppen mit **optUeberweisung**, **optBankeinzug**, **optNachnahme** und **optPaypal**.

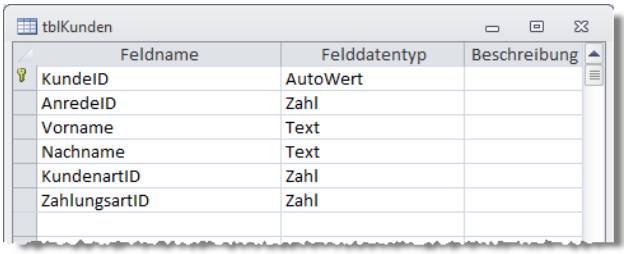


Bild 10: Zwei weitere Felder dienen als Beispiel für abhängige Optionsgruppen.

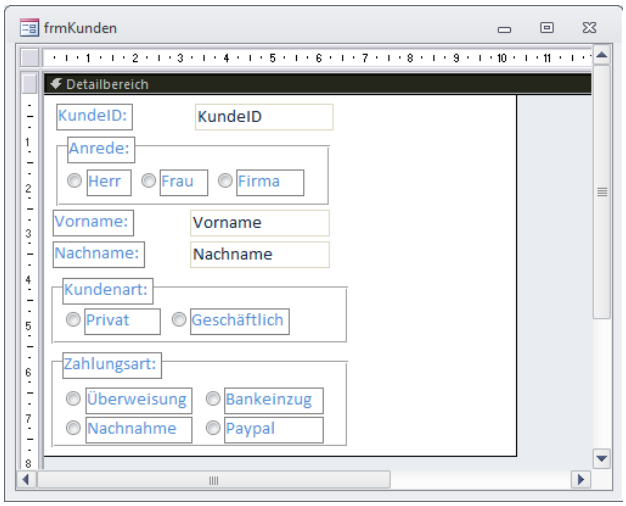


Bild 11: Formular mit abhängigen Optionsgruppen in der Entwurfsansicht

Sie können den Wert der beiden Felder nun schon prima mit den beiden Optionsgruppen einstellen.

Allerdings wollen wir erreichen, dass für eine Auswahl der Kundenart nur bestimmte Zahlungsarten freigeschaltet werden – beispielsweise soll ein Privatkunde nur die beiden Zahlungsarten **Nachnahme** und **Paypal** verwenden dürfen und ein geschäftlicher Kunde nur **Bankeinzug**, **Paypal** und **Überweisung**.

Wir müssen also das Ereignis **Nach Aktualisierung** der Optionsgruppe **ogrKundenartID** abgreifen und abhängig vom gewählten Wert die entsprechenden Zahlungsarten aktivieren beziehungsweise deaktivieren.

Dazu legen Sie die folgende Ereignisprozedur an:

```
Private Sub ogrKundenart_AfterUpdate()
    Select Case Me!ogrKundenart
        Case 1 'Privat
            Me!optBankeinzug.Enabled = False
            Me!optNachnahme.Enabled = True
            Me!optPaypal.Enabled = True
            Me!optUeberweisung.Enabled = False
        Case 2 'Geschäftlich
            Me!optBankeinzug.Enabled = True
            Me!optNachnahme.Enabled = False
            Me!optPaypal.Enabled = True
            Me!optUeberweisung.Enabled = True
    End Select
End Sub
```

Diese Prozedur ermittelt den aktuellen Wert der Optionsgruppe **ogrKundenart** und aktiviert oder deaktiviert die Optionsfelder der Optionsgruppe **ogrZahlungsart**, indem es deren Eigenschaft **Enabled** auf einen der beiden Werte **True** oder **False** einstellt.

Sie können nun bereits eine der beiden Kundenarten auswählen und betrachten, wie die Felder der Optionsgruppe **ogrZahlungsarten** aktiviert und deaktiviert werden. Die bisherige Vorgehensweise ist jedoch verbesserungswürdig: Beim Wechsel des Datensatzes ändert sich der Aktiviert-Zustand der Optionsfelder nicht – auch nicht, wenn für die Kunden-Datensätze verschiedene Kundenarten festgelegt wurden.

Die Lösung ist: Die Aktivierung/Deaktivierung muss auch beim Wechsel des Datensatzes erfolgen. Also quartieren wir die notwendigen Anweisungen in eine eigene Prozedur aus, die wie folgt aussieht:

```
Private Sub ZahlungsartenAktivieren()
    Select Case Me!ogrKundenart
        Case 1 'Privat
            Me!optBankeinzug.Enabled = False
            Me!optNachnahme.Enabled = True
            Me!optPaypal.Enabled = True
            Me!optUeberweisung.Enabled = False
        Case 2 'Geschäftlich
            Me!optBankeinzug.Enabled = True
            Me!optNachnahme.Enabled = False
            Me!optPaypal.Enabled = True
            Me!optUeberweisung.Enabled = True
    End Select
End Sub
```

Die Aktivierungs-Prozedur wird nun nicht nur durch die Ereignisprozedur aufgerufen, die beim Aktualisieren der Kundenart ausgelöst wird, sondern auch noch durch die Prozedur, die beim Ereignis **Beim Anzeigen** des Formulars feuert:

```
Private Sub Form_Current()
    ZahlungsartenAktivieren
End Sub

Private Sub ogrKundenart_AfterUpdate()
    ZahlungsartenAktivieren
End Sub
```

Fehlt noch ein wichtiger Schritt: Wenn Sie nämlich erst etwa die Kombination Privatkunde/Nachnahme ausgewählt haben und dann die Kundenart auf Geschäftskunde einstellen, wird zwar die Option Nachnahme deaktiviert – die Option bleibt jedoch ausgewählt.

In diesem Fall soll die Auswahl in der Optionsgruppe ogrZahlungsart einfach gelöscht werden. Die erreichen Sie, indem Sie die folgenden Zeilen zur Prozedur ZahlungsartenAktivieren hinzufügen:

```
If Me!optUeberweisung.Enabled = False And _
                        Me!ogrZahlungsart = 1 Then
    Me!ogrZahlungsart = Null
End If
If Me!optBankeinzug.Enabled = False And _
                        Me!ogrZahlungsart = 2 Then
    Me!ogrZahlungsart = Null
End If
If Me!optNachnahme.Enabled = False And _
                        Me!ogrZahlungsart = 3 Then
    Me!ogrZahlungsart = Null
End If
If Me!optPaypal.Enabled = False And _
                        Me!ogrZahlungsart = 4 Then
    Me!ogrZahlungsart = Null
End If
```

Die **If**-Bedingung prüft für jede der vier Optionen, ob das Optionsfeld deaktiviert ist und die Option dennoch ausgewählt ist. Ist dies der Fall, setzt die Prozedur den Wert der Optionsgruppe **ogrZahlungsart** auf **Null**.

Dies lässt sich übrigens auch eleganter lösen – Sie brauchen nicht für jede Option eine eigene **If...Then**-Bedingung aufzubauen. Dafür ist jedoch eine kurze Vorarbeit nötig, die Sie im Formular **frmKunden_II** der Beispieldatenbank nachvollziehen können.

Dazu benennen Sie die vier Optionsfelder der Optionsgruppe ogrZahlungsart zunächst in **opt1**, **opt2**, **opt3** und **opt4** um.

Dann deklarieren Sie in der Prozedur **ZahlungsartenAktivieren** eine neue Variable namens **i**:

```
Dim i As Integer
```

Das eventuell notwendige Leeren des Steuerelements **ogrZahlungsart** führen Sie dann in einer Schleife über alle vier Optionsfelder durch:

```
For i = 1 To 4
    If Me("opt" & i).Enabled = False And _
                        Me!ogrZahlungsart = i Then
        Me!ogrZahlungsart = Null
    End If
Next i
```

Die **If**-Bedingung prüft nun nicht mehr ein Steuerelement mit einem festen Namen, sondern alle Steuerelemente, deren Name mit opt beginnt und mit einer der vier Zahlen **1**, **2**, **3** oder **4** endet.

Wenn Sie den Steuerelementnamen dabei nicht mit der Ausrufezeichen-Syntax angeben (**Me!opt1**), sondern mit der Auflistungsform **Me("opt1")**, können Sie die Zahl aus dem String mit dem Steuerelementnamen herausziehen und durch eine Variable ersetzen: **Me("opt" & i)**.

Und auch im zweiten Ausdruck der **If...Then**-Bedingung ersetzen Sie die explizite Angabe des Wertes von **ogrZahlungsart** durch die Variable **i**.

Natürlich müssen Sie auch noch die übrigen in der Prozedur **ZahlungsartenAktivieren** angegebenen Optionsfelder anpassen:

```
Private Sub ZahlungsartenAktivieren()
    Dim i As Integer
    Select Case Me!ogrKundenart
        Case 1 'Privat
            Me!opt1.Enabled = False
            Me!opt2.Enabled = False
            Me!opt3.Enabled = True
            Me!opt4.Enabled = True
        Case 2 'Geschäftlich
            Me!opt1.Enabled = True
            Me!opt2.Enabled = True
            Me!opt3.Enabled = False
            Me!opt4.Enabled = True
    End Select
    For i = 1 To 4
        If Me("opt" & i).Enabled = False And _
                        Me!ogrZahlungsart = i Then
            Me!ogrZahlungsart = Null
        End If
    Next i
End Sub
```

Zusammenfassung und Ausblick

Das Optionsgruppen-Steuerelement und das Optionsfeld-Steuerelement erlauben die Auswahl von numerischen Werten für gebundene und ungebundene Einsatzzwecke.

Sie eignen sich speziell für solche Fälle, bei denen die Anzahl der zur Auswahl stehenden Werte erstens begrenzt ist und sich zweitens im Laufe der Zeit voraussichtlich nicht geändert wird.

Berichtsbereiche

Ein Bericht bietet einige Bereiche mehr als ein Formular. Damit können Sie auch komplexere Daten strukturiert in einem Bericht darstellen. Neben den Berichtsbereichen, die nur einmal pro Bericht angezeigt werden (Berichtskopf und -fuß) sowie den seitenweise erscheinenden Bereichen (Seitenkopf und -fuß) gibt es noch die Kopf- und Fußbereiche der einzelnen Gruppierungen - um diese kümmert sich ein weiterer Artikel. Dieser Artikel stellt die einzelnen Bereiche vor und erklärt, was Sie bei deren Einsatz beachten müssen.

Beispieldatenbank

Die Beispiele dieses Artikels finden Sie in der Datenbank **1109_Berichtsbereiche.mdb**.

Standardbericht

Wenn Sie einen neuen Bericht in der Entwurfsansicht öffnen, zeigt dieser standardmäßig drei Bereiche an: den Seitenkopf, den Detailbereich und den Seitenfuß (siehe Bild 1). Wenn Sie diesen Bereichen einen Text hinzufügen, um diese später identifizieren zu können, sieht die Seitenansicht wie in Bild 2 aus. Die wichtigste Erkenntnis, die wir hier gewinnen ist die folgende: Der Seitenfuß wird immer ganz unten angeordnet. Es spielt keine Rolle, wieviele Detaildatensätze sich auf der Seite befinden, er rutscht niemals nach oben.

Damit Sie die einzelnen Berichtsbereiche unterscheiden können, haben wir Bezeichnungsfelder mit dem jeweiligen Bereichsnamen integriert. Wenn Sie später Überschriften et cetera zum Berichts- und/oder Seitenkopf hinzufügen, verwenden Sie am besten ebenfalls Bezeichnungsfelder.

Bereiche ein- und ausblenden

Vielleicht brauchen Sie Seitenkopf und -fuß nicht, oder Sie benötigen zusätzlich einen Berichtskopf und -fuß: Dann blenden Sie diese Bereiche am einfachsten über das Kontextmenü eines der Bereichsköpfe ein oder aus (siehe Bild 3). Diese Bereiche können Sie auch über entsprechende Einträge der Menüleiste (Access 2003 und älter) beziehungsweise des Ribbons (Access 2007 und jünger) beein-

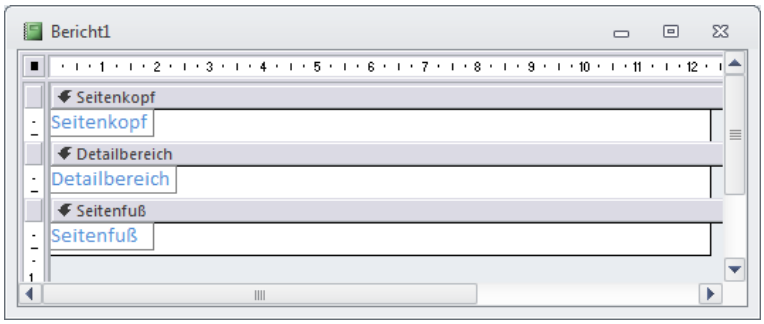


Bild 1: Standardmäßig angezeigte Berichtsbereiche

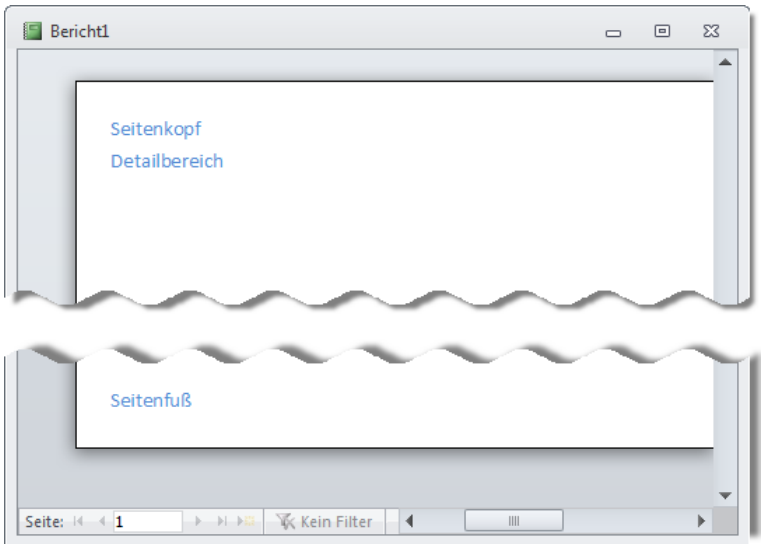


Bild 2: Berichtsbereiche in der Seitenansicht

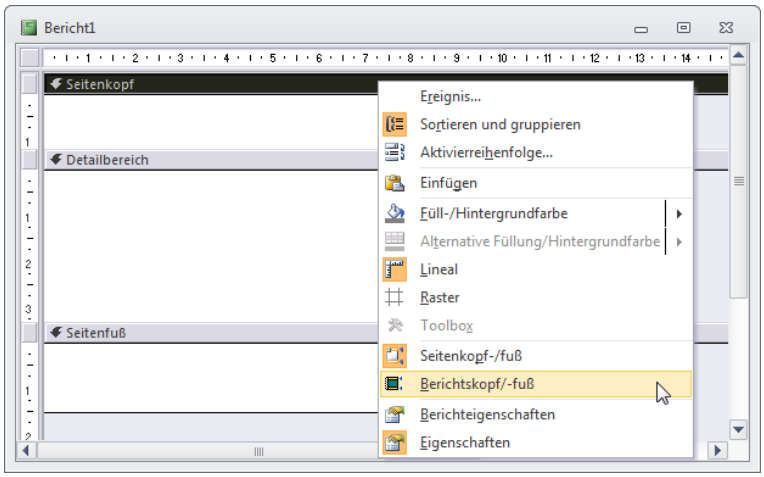


Bild 3: Ein- und Ausblenden von Berichtsbereichen

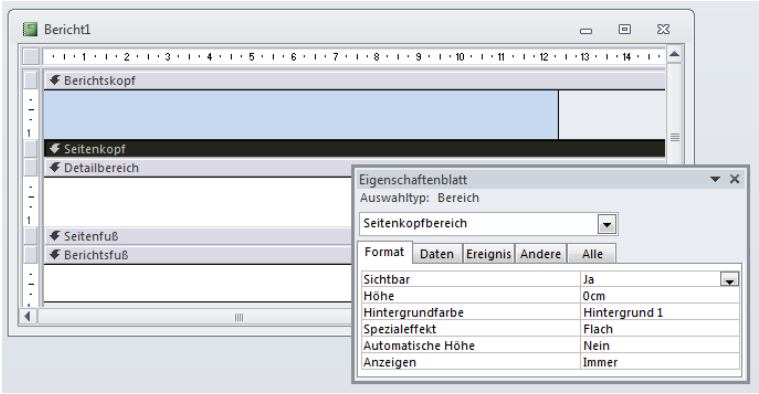


Bild 4: Bericht mit ausgeblendeten Berichtsbereichen

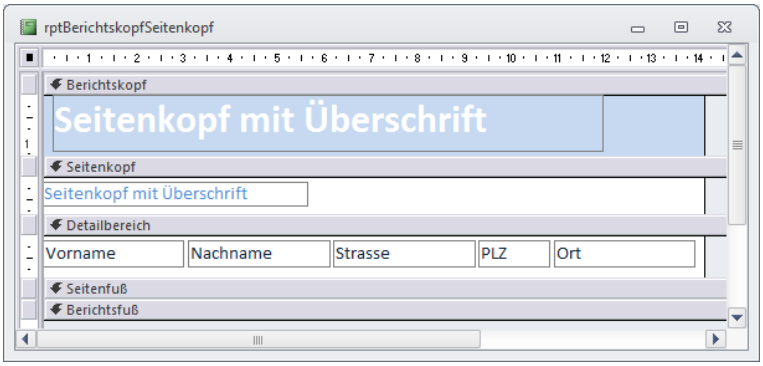


Bild 5: Bericht mit Berichtskopf und Seitenkopf

Bild 6: Berichts- und Seitenkopf erscheinen beide auf der ersten Seite.

Bild 7: Seitenkopf ab Seite 2? Kein Problem mit dieser Eigenschaft!

flussen. Die Kontextmenü-Variante funktioniert jedoch versionsübergreifend.

Wenn Sie nur den Seitenkopf oder -fuß beziehungsweise den Berichtskopf oder -fuß benötigen, aktivieren Sie jeweils beide Bereiche und stellen die Eigenschaft Höhe des unerwünschten Bereichs im Eigenschaftsfenster auf **0** ein. Alternativ verschieben Sie einfach den darunter liegenden Bereich soweit nach oben, bis der nicht erwünschte Bereich nicht mehr sichtbar ist (wenn der Bereich sich ganz unten befindet, ziehen Sie den unteren Rand einfach mit der Maus nach oben, bis die Höhe **0** ist). Bild 4 zeigt einen Bericht mit Berichts- und Seitenbereichen, von denen jeweils einer durch Einstellen der Eigenschaft **Höhe** auf den Wert **0cm** ausgeblendet wird.

Überschriften in Berichts- und Seitenkopf

Wenn Sie einen Bericht mit Berichtskopf und Seitenkopf ausstatten, hat das meist folgenden Grund: Sie möchten auf der ersten Seite eine große Überschrift abbilden, die auf den folgenden Seiten wiederkehrt – allerdings möglichst nicht mehr allzu groß. Dann legen Sie wie in einen entsprechenden Berichtskopfbereich und einen Seitenkopfbereich an (siehe Bild 5).

Ein Wechsel in die Seitenansicht sorgt allerdings für Ernüchterung: Das der Seitenkopf bereits auf der ersten Seite angezeigt wird, war so nicht gedacht (siehe Bild 6). Aber woher soll Access auch wissen, dass der Seitenkopf erst ab der zweiten Seite erscheinen soll? Ganz einfach: Sie teilen es mit einer entsprechenden Eigenschaft mit.

Dazu wechseln Sie erneut in die Entwurfsansicht des Berichts. Die Eigenschaft **Seitenkopf** ist nicht etwa eine Eigenschaft des entsprechenden Bereichsbereichs, sondern des Berichts selbst. Deshalb markieren Sie zunächst den Bericht und finden dann unter dem Registerreiter **Format** die gewünschte Eigenschaft (siehe Bild 7).

Für unseren Fall ist die Einstellung Außer Berichtskopf die Richtige. Sie sorgt dafür, dass der Seitenkopf auf allen Seiten außer auf der mit dem Berichtskopf erscheint. In der Regel bedeutet dies, dass der Seitenkopf erstmalig auf der zweiten Seite erscheint. Das gilt auch, wenn gar kein Berichtskopf vorhanden ist. Sollte der Berichtskopf mehr als eine Seite umfassen, verschiebt sich das erstmalige Erscheinen des Seitenkopfes entsprechend nach hinten.

Sie können den Seitenkopf auch für die Seite mit dem Berichtsfuß ausblenden. Den Seitenfuß blenden Sie auf die gleiche Weise für den Berichtskopf, den Berichtsfuß oder beide aus.

Tipp: Bericht markieren

Wenn Sie wie oben beschrieben nicht einen Berichtsbereich oder ein Steuerelements markieren wollen, sondern den Bericht selbst, haben Sie mehrere Möglichkeiten:

- Klicken Sie in das kleine Kästchen links oben in der Entwurfsansicht (nur sichtbar, wenn Lineal eingeblendet – einzublenden über den Kontextmenüeintrag **Lineal** eines der Bereichsköpfe),
- klicken Sie auf den grauen Bereich rechts neben oder unter den Berichtsbereichen oder
- wählen Sie im Kombinationsfeld oben im Eigenschaftsfenster den Eintrag **Bericht** aus.

Dokumentinformationen in Berichts- und Seitenfuß

Dokumentinformationen wie Seitenzahl, Datum et cetera können Sie in allen Berichtsbereichen unterbringen, aber meistens landen diese Daten im Seitenfuß des Berichts. Dieser erscheint standardmäßig auf jeder Seite – außer, Sie stellen die Eigenschaft **Seitenfuß** des Berichts auf einen anderen Wert als **Alle Seiten** ein.

Wenn Sie den Seitenfußbereich eingeblendet haben, fügen Sie diesem für jede gewünschte Information ein Textfeld hinzu, dessen Eigenschaft **Steuerelementinhalt** mit den benötigten Formeln gefüllt wird.

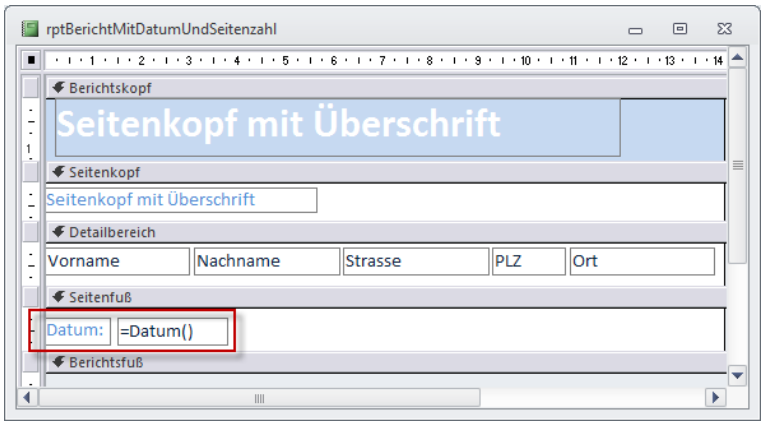


Bild 8: Datum im Seitenfuß

Bild 9: Datum und Seitenzahl im Seitenfuß

Wenn Sie das Datum im Seitenfuß anzeigen möchten, fügen Sie dort ein Textfeld hinzu. Ändern Sie den Text des Bezeichnungsfeldes auf Datum und den Namen des Textfeldes auf **txtDatum**. Stellen Sie außerdem den Wert der Eigenschaft **Steuerelementinhalt** auf **=Datum()** ein (siehe Bild 8). Die Funktion **Datum()** liefert das aktuelle Datum entsprechend den aktuellen Ländereinstellung für das Datum im Kurzformat.

Das es sich bei dem angezeigten Wert um ein Datum handelt, wird jeder gleich erkennen können – daher ist das Bezeichnungsfeld eigentlich überflüssig und kann gelöscht werden.

Die Seitenzahl fügen Sie auf ähnliche Weise zum Bericht hinzu. Das Steuerelement soll **txtSeiten** heißen, der Inhalt lautet **=[Seite] & "/" & [Seiten]**. Die Funktion **Seite** liefert die aktuelle Seite, **Seiten** die Gesamtzahl der Seiten des Dokuments.

Die unterschiedliche Schreibweise der Funktionen für das Datum (**Datum()**) und die Seitenzahlen (**[Seite]** und **[Seiten]**) rührt wohl daher, dass **Datum()** eine allgemeine VBA-Funktion ist, **Seite** und **Seiten** jedoch berichtsspezifische Funktionen liefern.

Das Ergebnis sieht wie in Bild 9 aus.

Sortieren in Berichten

In Tabellen stellen Sie die Sortierung in der Datenblattansicht ein, in Abfragen legen Sie diese im Entwurfsraster fest und in Formularen und den enthaltenen Steuerelementen wie Kombinationsfeldern und Listenfeldern gibt die für die Datenherkunft festgelegte Reihenfolge den Takt vor. Wie aber sieht es in Berichten aus? Um es vorweg zu nehmen: ganz anders. Dieser Artikel erläutert, wie Sie die in Berichten anzuzeigenden Daten sortieren.

Beispieldatenbank

Die Beispiele dieses Artikels finden Sie in der Datenbank **1109_SortierenInBerichten.mdb**.

Sortieren und ...

Es ist sicher einer der weitverbreitetsten Anfängerfehler: Sie stellen die in einem Bericht anzuzeigenden Daten mithilfe einer entsprechenden Abfrage zusammen und weisen diese dem Bericht als Datenherkunft zu. Sie füllen den Bericht mit Steuerelementen, wechseln in die Seitenansicht und ... was ist das? Warum sortiert der Bericht die Daten nicht wie in der zugrunde liegenden Abfrage vorgegeben?

Der Grund für dieses Verhalten ist: Berichte haben einen eigenen Dialog, mit dem Sie die Reihenfolge der anzuzeigenden Daten festlegen. Die Sortierreihenfolge der als Datenherkunft verwendeten Tabellen oder Abfragen wird unter Umständen schlicht ignoriert, auch wenn Sie im Bericht gar keine explizite Sortierreihenfolge angeben. Das bedeutet: Es kommt vor, dass die in der Abfrage angegebene Sortierreihenfolge berücksichtigt wird, aber dies geschieht nicht zuverlässig. Das ist aber auch kein Problem, denn Berichte bringen ihre eigene Funktion zum Festlegen der Sortierung fest, ja sogar einen eigenen Dialog.

Diesen öffnen Sie am einfachsten, indem Sie in der Entwurfsansicht des Berichts mit der rechten Maustaste auf einen der Bereichsköpfe klicken und den Eintrag **Gruppieren und sortieren** aus dem Kontextmenü auswählen (siehe Bild 1).

Es erscheint der Dialog aus Bild 2, mit dem Sie die Sortierreihenfolge einstellen

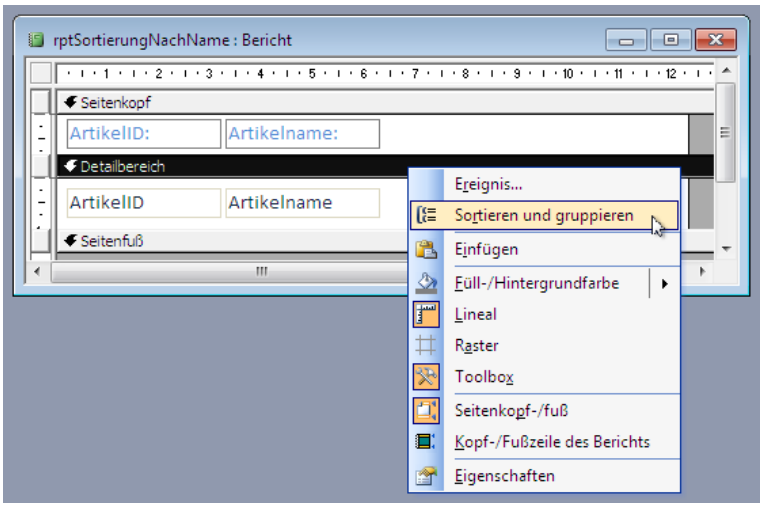


Bild 1: Aufruf des Dialogs zum Angeben einer Sortierung für die Daten des Berichts

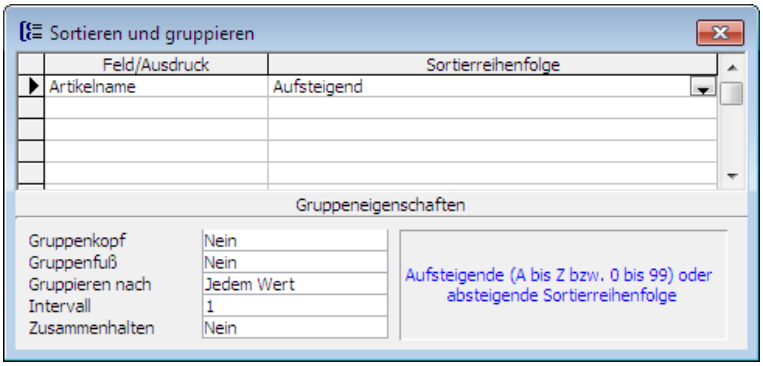


Bild 2: Angeben der Sortierreihenfolge eines Berichts

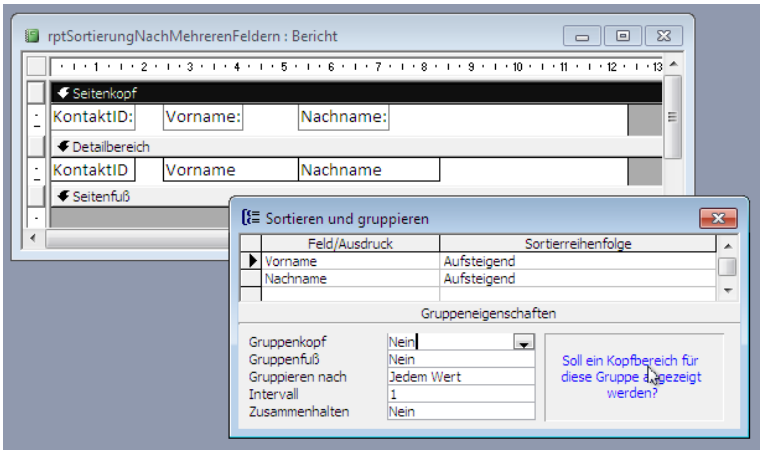


Bild 3: Bericht mit zweifacher Sortierung

können. Wenn Sie beispielsweise eine Artikelliste nach dem Inhalt des Feldes **Artikelname** sortieren möchten, wählen Sie unter **Feld/Ausdruck** den Eintrag **Artikelname** aus und behalten rechts unter Sortierreihenfolge den Wert **Aufsteigend** bei.

Wenn Sie wie in Bild 3 gleich nach mehreren Felder sortieren möchten, berücksichtigen Sie, dass die Sortierung von oben nach unten abgearbeitet wird. Das bedeutet in diesem Beispiel, dass erst nach dem Vornamen und dann nach dem Nachnamen sortiert wird (siehe Bild 4).

Sortieren nach Nachschlagefeldern

Access macht es dem Entwickler an vielen Stellen sehr einfach. Sie können beispielsweise Datensätze der Tabelle **tblArtikel** mitsamt Kategorie- und Lieferantennamen in einem Bericht anzeigen, ohne dass die Tabellen **tblKategorien** und **tblLieferanten** explizit zur Datenherkunft hinzugefügt werden. Die anzuzeigenden Werte liegen lediglich in Form der Datensatzherkunft der in der Tabelle **tblArtikel** definierten Nachschlagefelder vor. Wenn Sie diese Felder zum Berichtsentwurf hinzufügen, werden diese zwar noch als Kombinationsfeld angezeigt (Access übernimmt hier die im Tabellenentwurf festgelegten Einstellungen, aus einem Nachschlagefeld wird also ein Kombinationsfeld). Beim Wechsel in die Seitenansicht und auch beim Drucken ersetzt Access die Kombinationsfelder jedoch durch einfache Textfelder und zeigt schlicht die Texte an, die auch die Kombinationsfelder angezeigt hätten.

Was aber geschieht, wenn Sie etwa nach dem Feld **LieferantID** der Tabelle **tblArtikel** sortieren (siehe Bild 5)? Der Bericht zeigt zwar die Lieferantennamen statt der ID an, aber die Sortierung berücksichtigt die im Fremdschlüsselfeld gespeicherten Werte (siehe Bild 6).

Wenn Sie in einem Bericht also tatsächlich nach Feldinhalten sortieren wollen, die in Kombinationsfeldern angezeigt werden, müssen Sie die entsprechenden Felder der verknüpften Tabellen mit in die Datenherkunft aufnehmen und diese im Dialog **Sortieren und gruppieren** angeben. Die Abfrage mit den Feldern **Lieferant** und **Kategorie** sähe dann etwa wie in Bild 7 aus.

Nachdem Sie diese Abfrage als Datenherkunft des Berichts eingestellt haben, können Sie im Dialog **Sortieren und gruppieren** das Feld **Firma** als Sortierkriterium auswählen (siehe Bild 8). Danach gelingt auch die Sortierung nach Feldern wie **Firma** oder **Kategoriename** (siehe Bericht **rptArtikelNachLieferant**).

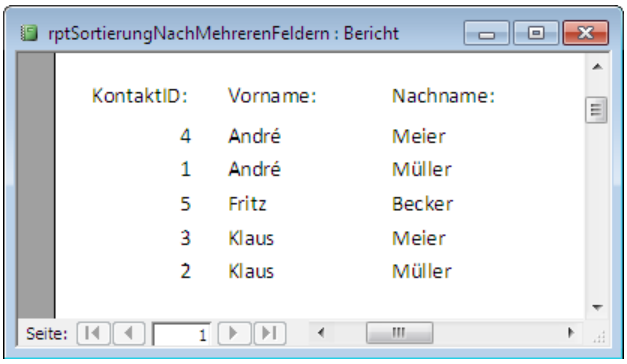


Bild 4: Sortierung nach Vorname und Nachname

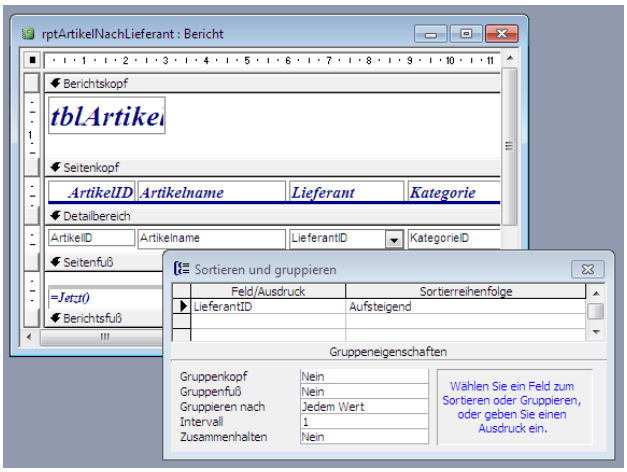


Bild 5: Einrichten der Sortierung nach einem Fremdschlüssel-/Nachschlagefeld

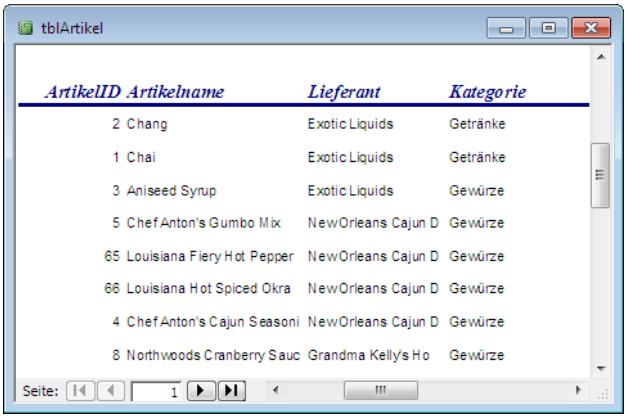


Bild 6: Das Sortieren nach Werten aus Nachschlagefeldern funktioniert nicht wie erwartet.

Dynamisch sortieren

Gegebenenfalls möchten Sie dem Benutzer die Möglichkeit bieten, selbst die Sortierreihenfolge im Be-

richt festzulegen. Im Beispielbericht **rptArtikelSortiert** haben wir dies realisiert. Allerdings benötigen Sie zusätzlich ein Formular, mit dem Sie den Bericht öffnen und das gewünschte Sortierfeld festlegen.

Das Formular heißt **frmArtikelbericht** und sieht wie in Bild 9 aus. Es enthält eine Optionsgruppe namens **ogrSortierung** mit drei Optionen, die den Werten **1**, **2** und **3** entsprechen. Die Schaltfläche **cmdBerichtOeffnen** enthält nur eine einzige Anweisung, die den Bericht **rptArtikelSortiert** in der Seitenansicht öffnet:

```
Private Sub cmdBerichtOeffnen_Click()
    DoCmd.OpenReport _
        "rptArtikelSortiert", _
        acViewPreview
End Sub
```

Wie erfährt der Bericht aber nun, welche Sortierung verwendet werden soll und wie stellen wir die Sortierung schließlich im Bericht ein? Dazu verwenden wir eine Ereignisprozedur, die durch das Ereignis **Beim Öffnen** des Berichts ausgelöst wird.

Die Prozedur greift auf das Steuerelement ogrSortierung des noch geöffneten Formulars zu und liest dessen Wert aus. Lautet dieser **1**, soll beispielsweise nach dem Feld **Artikelname** sortiert werden. Dazu stellen die folgenden Zeilen den Wert, der im Dialog **Sortieren und gruppieren** in der Spalte **Feld/Ausdruck** zu finden ist, auf das zu sortierende Feld ein, also etwa **Artikelname**:

```
Private Sub Report_Open(Cancel As Integer)
  Dim intSortierung As Integer
  intSortierung = _
    Forms!frmArtikelbericht!ogrSortierung
  Select Case intSortierung
    Case 1
      Me.GroupLevel(0).ControlSource = "Artikelname"
    Case 2
      Me.GroupLevel(0).ControlSource = "Firma"
    Case 3
      Me.GroupLevel(0).ControlSource = "Kategoriename"
  End Select
End Sub
```

Dies funktioniert übrigens nur, wenn Sie für den Bericht mindestens eine Sortierung im Dialog **Sortieren und gruppieren** eingerichtet haben. Falls nicht, ist

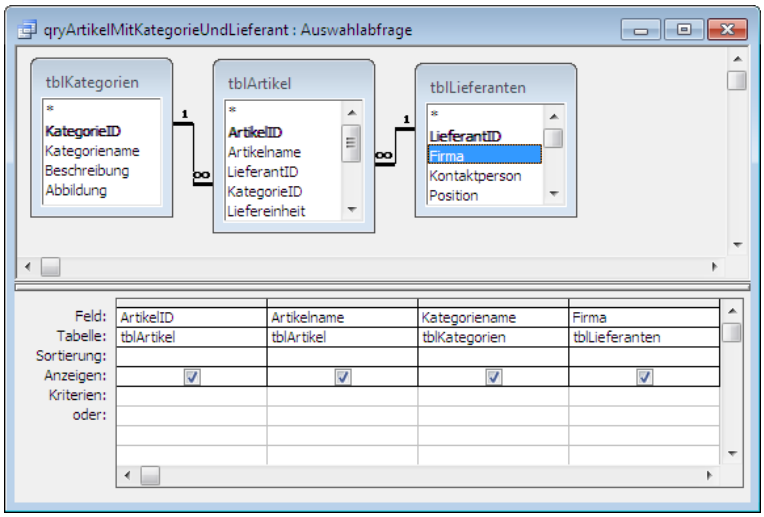


Bild 7: Um im Bericht nach dem Inhalt von Nachschlagefeldern sortieren zu können, müssen die Felder mit den Werten in der Datenherkunft enthalten sein.

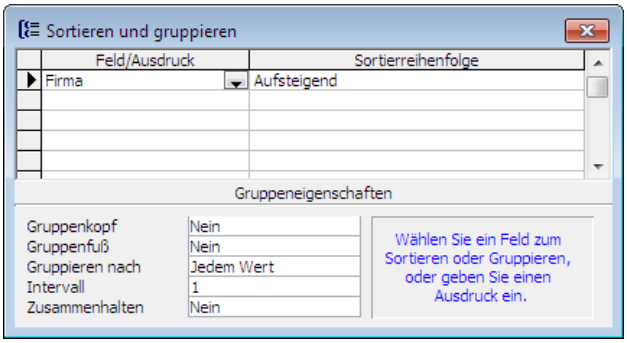


Bild 8: Das Feld **Firma** der Tabelle **tblHersteller** wird nun als Sortierkriterium festgelegt.

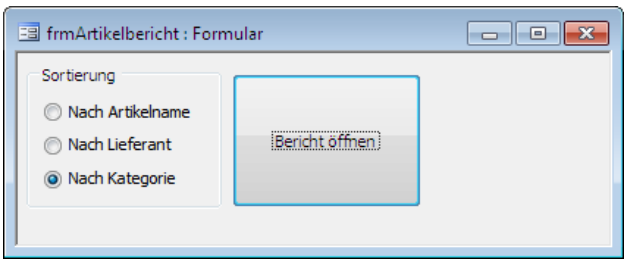


Bild 9: Öffnen eines Berichts mit dynamischer Sortierung

Me.GroupLevel(0), also die ganz oben angegebene Sortierung, nicht vorhanden und es wird ein Fehler ausgelöst. Wenn Sie per VBA nach weiteren Ebenen sortieren möchten, legen Sie weitere Sortierungen im Dialog Sortieren und gruppieren an und stellen diese über die Eigenschaft **Me.GroupLevel(1).ControlSource** und so weiter ein.

Und wenn Sie auch noch die Sortierreihenfolge festlegen möchten, stellen Sie **Me.GroupLevel(0).SortOrder** auf den Wert **True** (für absteigende Sortierung) oder **False** (für aufsteigende Sortierung) ein.

Tipps und Tricks

In dieser Ausgabe zeigen wir Ihnen, wie Sie das Standardverzeichnis beim Öffnen und Erstellen von Datenbanken anpassen, wie Sie Datenbankobjekte ausblenden, damit Otto Normalverbraucher diese nicht mehr im Datenbankfenster beziehungsweise im Navigationsbereich vorfindet, wie Sie schnell die aktuellen Werte für Zeit und Datum in Textfelder eintragen, wie Sie den Wert aus dem gleichen Feld des vorherigen Datensatzes übernehmen und vieles mehr!

Beispieldatenbank

Die Beispiele dieses Artikels finden Sie in der Datenbank **1109_TippsUndTricks.mdb**.

Standardverzeichnis einstellen

Wenn Sie unter Access eine neue Datenbank anlegen oder eine Datenbank öffnen möchten, zeigt Access immer den entsprechenden Dateidialog an. Standardmäßig wird dort je nach Access-Version ein Verzeichnis wie **Eigene Dokumente** oder ähnlich angezeigt.

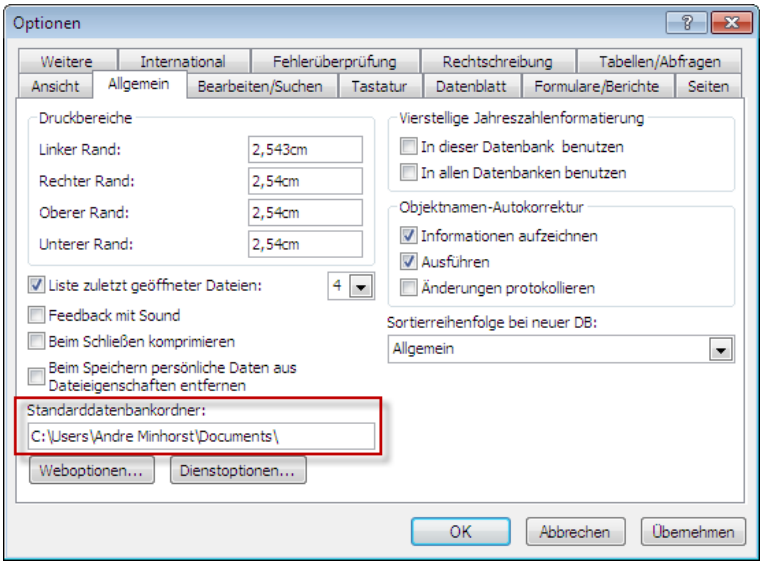


Bild 1: Einstellen des Standardverzeichnisses von Access

Normalerweise verwenden Sie aber ein anderes Verzeichnis, um Ihre Access-Datenbanken zu speichern – oder zumindest ein gemeinsames übergeordnetes Verzeichnis, von dem aus Sie mit ein paar Mausklicks zum Zielverzeichnis gelangen.

Da ist es doch praktisch, dass Sie das Verzeichnis, das beim Öffnen oder Anlegen von Datenbankdateien angezeigt wird, selbst festlegen können. Dazu gehen Sie wie folgt vor:

- Unter Access 2003 und älter wählen Sie den Menüeintrag **Extras|Optionen** (dieser Eintrag ist nur bei geöffneter Datenbank aktiviert) und wechseln im nun erscheinenden Dialog zur Registerseite **Allemein**. Dort finden Sie die Option **Standarddatenbankordner**, mit der Sie das gewünschte Verzeichnis festlegen können (siehe Bild 1).

- Unter Access 2007 klicken Sie auf den Office-Button und dann auf die Schaltfläche **Access-Optionen**. Die Option **Standarddatenbankordner** befindet sich im Bereich **Häufig verwendet**.

- Unter Access 2010 wähle Sie den Ribbon-Eintrag **Datei|Optionen** aus, wechseln zum Bereich **Allgemein** und stellen das Standardverzeichnis unter **Datenbanken erstellen|Standarddatenbankordner** ein.

Objekte ausblenden

Normalerweise sollte eine professionelle Anwendung eine eigene Benutzeroberfläche bestehend aus einer Menü- oder Ribbon-Steuerung, Formularen und Berichten mitbringen. Das Datenbankfenster (bis Access 2003) beziehungsweise der Navigationsbereich (ab Access 2007) sollten für den Anwender Ihrer Datenbanken tabu sein.

Wenn die Anwendung jedoch nicht ganz so professionellen Ansprüchen genügen muss, kann man den Benutzer die benötigten Access-Objekte natürlich über Datenbankfenster/Navigationsbereich öffnen lassen.

Dann sollten Sie aber zumindest nur die für die Benutzung der Datenbank wichtigen Objekte anzeigen. Die anderen blenden Sie einfach aus, und zwar so:

- Unter Access 2003 klicken Sie im Datenbankfenster mit der rechten Maustaste auf das fragliche Objekt und wählen den Kontextmenüeintrag **Eigenschaften** aus (siehe Bild 2). Im nun erschein-

den Eigenschaftsfenster für dieses Objekt aktivieren Sie die Option **Ausgeblendet** (siehe Bild 3).

- Unter Access 2007 und Access 2010 gehen Sie genauso vor – mit dem Unterschied, dass Sie nicht auf das Objekt im Datenbankfenster, sondern im Navigationsbereich klicken und dass der Eintrag im Kontextmenü je nach Objekt **Tabelleneigenschaften**, **Objekteigenschaften** et cetera heißt.

Nach dem Schließen des jeweiligen Eigenschaftsfensters blendet Access den Eintrag des betroffenen Elements aus.

Ausgeblendete Objekte anzeigen

Was dem Benutzer die Arbeit mit Ihrer Anwendung erleichtert, ist für den Entwickler hinderlich: Wie soll man nun auf die Schnelle auf die ausgeblendeten Objekte zugreifen? Auch das ist kein Problem, denn Access hält eine Option bereit, mit der Sie auch ausgeblendete Objekte im Datenbankfenster beziehungsweise im Navigationsbereich anzeigen können.

- Unter Access 2003 und älter betätigen Sie dazu zunächst den Menüeintrag **Extras|Optionen** und wechseln im Optionen-Dialog zur Registerseite **Ansicht**. Dort finden Sie die Option **Ausblendete Objekte**, die Sie aktivieren müssen (siehe Bild 4). Wenn Sie den Optionen-Dialog nun schließen, zeigt Access die ausgeblendeten Objekte mit abgeblendetem Symbol an (siehe Bild 5).

- Unter Access 2007 und jünger finden Sie diese Option, wenn Sie mit der rechten Maustaste auf die Titelzeile des Navigationsbereichs klicken und den Eintrag **Navigationsoptionen** auswählen. Der nun erscheinenden Dialog **Navigationsoptionen** bietet ebenfalls die Option **Ausgeblendete Objekte** an. Hier werden die wieder sichtbar gemachten ausgeblendeten Objekte zusätzlich in grauer Schrift angezeigt.

Aktuelles Datum und aktuelle Zeit blitzschnell

Datum und Zeit muss man immer wieder mal in Textfelder eingeben, manchmal sogar beides. Dummerweise muss man beim Datum meist erstmal überlegen (okay, manch einer hat auch immer das aktuelle Datum parat), und die Uhrzeit kann man unten rechts in der Taskleiste ablesen. Aber diese Informationen

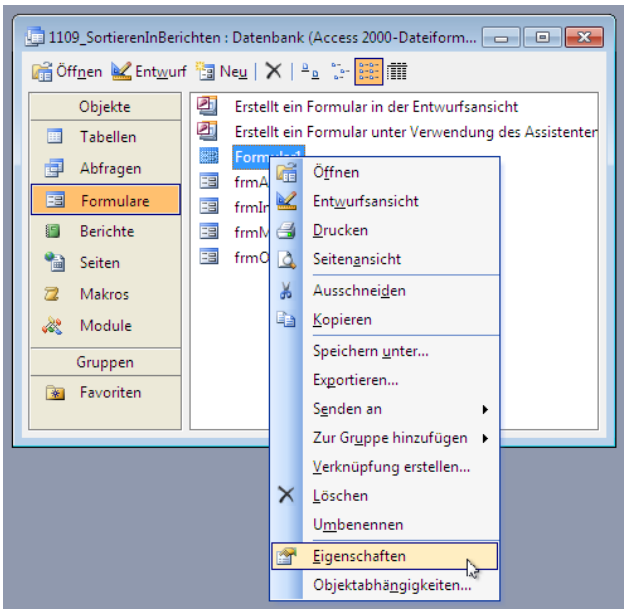


Bild 2: Anzeigen der Eigenschaften eines Datenbank-Objekts

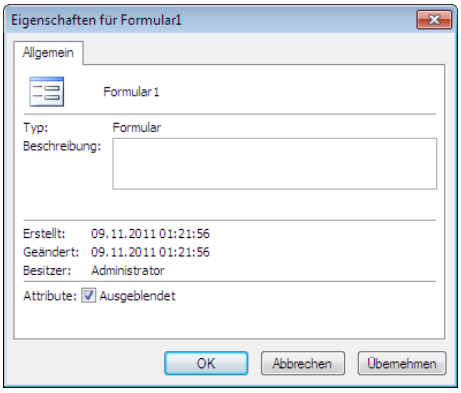


Bild 3: Ausblenden eines Objekts

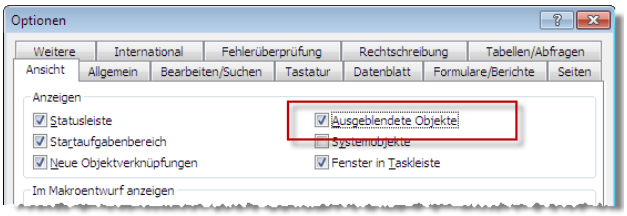


Bild 4: Einblenden ausgeblendeter Objekte

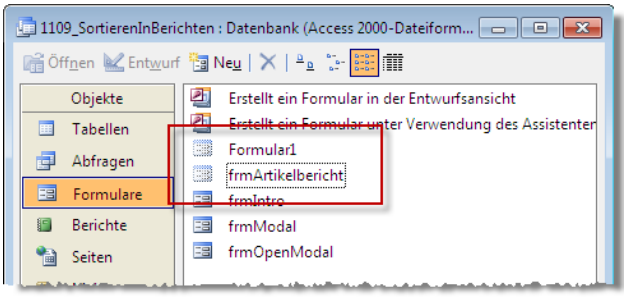


Bild 5: Ausgeblendete Objekte wieder eingeblendet

dann auch noch mühevoll in ein Textfeld übertragen? Das ist erstens fehleranfällig und zweitens viel zu viel Arbeit. Was halten Sie von je einer Tastenkombination für das aktuelle Datum und die aktuelle Uhrzeit? Hier sind sie:

- Aktuelles Datum: **Strg + ;** (also **Strg + Umschalt + ,**)
- Aktuelle Zeit: **Strg + :** (also **Strg + Umschalt + .**)

Wert aus vorherigem Datensatz übernehmen

Manchmal geben Sie Daten ein, die in bestimmten Feldern immer wieder die gleichen Werte haben, teilweise sogar den Wert, den Sie erst im vorherigen Datensatz vergeben haben.

Hier bietet Access eine wirklich stiefmütterlich behandelte Tastenkombination an (auch der Autor dieser Zeilen entdeckte diesen Shortcut erst bei der Recherche zu diesem Artikel): Betätigen Sie einfach die Tastenkombination **Strg + #**, um im aktuellen Datensatz den Wert des gleichen Feldes des vorherigen Datensatzes einzufügen.

Faszinierend einfach – das einzige Problem ist, dass Sie sich diese Tastenkombination merken müssen (siehe Bild 6).

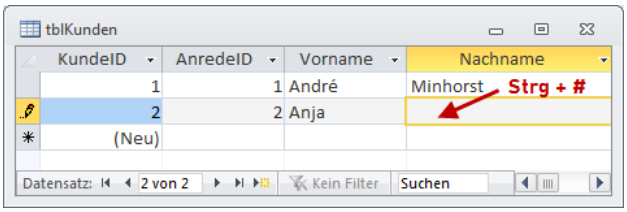


Bild 6: Daten aus dem vorherigen Datensatz übernehmen Sie mit der Tastenkombination **Strg + #**.

Datum und Zeit im Formulartitel

Wenn eine Anwendung es erfordert, dass der Benutzer immer die aktuelle Zeit und gegebenenfalls auch das Datum im Blick hat, können Sie diese Informationen leicht im Titel des Formulars anzeigen (vorausgesetzt, Sie verwenden nicht die Registerkartenansicht von Access 2007 und jünger).

Das scheint ganz einfach zu sein: Sie legen einfach eine Prozedur an, die durch das Ereignis Beim Laden des Formulars ausgelöst wird. Dazu wählen Sie in der Entwurfsansicht des Formulars den Wert [Ereignisprozedur] für die Eigenschaft Beim Laden aus und ergänzen die nach einem Klick auf die Schaltfläche mit den drei Punkten (...) erscheinende Prozedur wie folgt:

```
Private Sub Form_Load()
    Me.Caption = Now
End Sub
```

Damit schreiben Sie den Wert der Funktion **Now**, die das Datum und den Tag zurückliefert, in die Eigenschaft **Caption** des Formulars – und dies entspricht der Beschriftung der Titelleiste des Formulars. Das Problem ist: Es wird nur die beim Laden des Formulars aktuelle Zeit eingetragen und es erfolgt keine Aktualisierung.

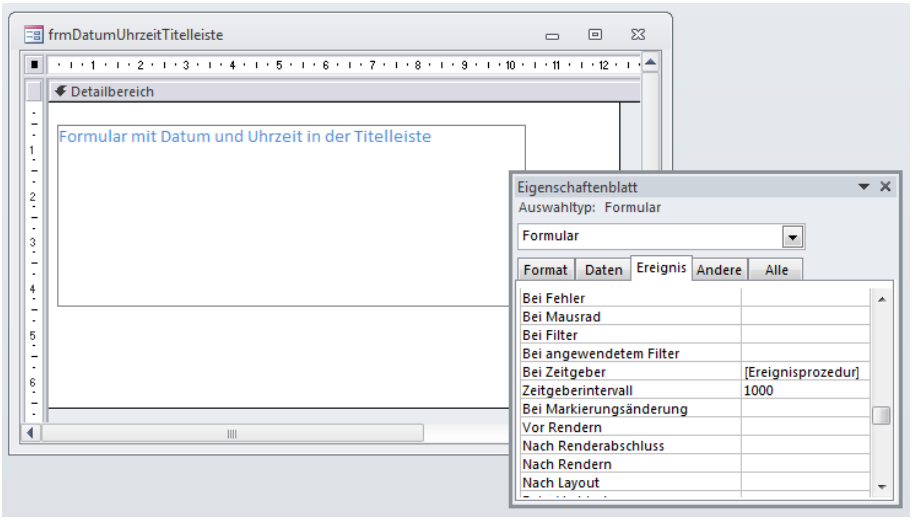


Bild 7: Für die sekündliche Aktualisierung der Uhrzeit in der Titelleiste verwenden Sie die Zeitgeber-Ereignisprozedur eines Formulars.

Das ist ein Fall für das Ereignis **Bei Zeitgeber**. Legen Sie auch hier eine Ereignisprozedur an, die ähnlich wie die vorherige Prozedur aussieht:

```
Private Sub Form_Timer()
    Me.Caption = Now
End Sub
```

Aber wann wird diese Prozedur ausgelöst? Dies legen Sie durch Vergabe eines Zahlenwertes für die Eigenschaft **Zeitgeberintervall** fest. Sie erwartet die Angabe eines Intervalls in Millisekunden. Wenn Sie also eine

Aktualisierung der Zeit in der Titelleiste wünschen, geben Sie den Wert **1.000** an (siehe Bild 7). Dies zeigt die aktuelle Zeit jedoch erst eine Sekunde nach dem Öffnen des Formulars an, weshalb Sie die Prozedur für das Ereignis **Beim Laden** gleich behalten sollten. Das Ergebnis sieht wie in Bild 8 aus.

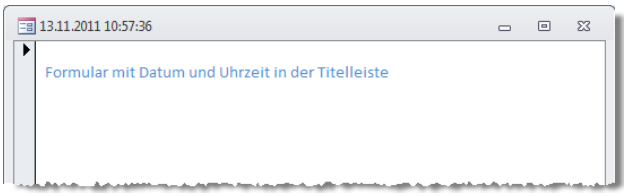


Bild 8: Formular mit Datum und Zeit in der Titellleiste

Insgesamt sind mit dem **Timer**-Ereignis verbundene Bildschirmaktualisierungen jedoch mit Vorsicht zu genießen: Befindet sich das betroffene Formular einmal nicht im Vordergrund, kann es sein, dass die darüber liegenden Fenster Flackern et cetera.

Memo-Feld als Sortierungs- und Gruppierungsfeld

Memo-Felder haben den große Vorteil gegenüber Textfeldern, dass Sie Text mit viel mehr Zeichen speichern können als herkömmliche Textfelder. Allerdings haben sie auch Nachteile: So lassen sich in Berichten keine Sortierungen oder Gruppierungen auf Basis dieser Felder durchführen – Memofelder tauchen im Dialog **Sortieren und Gruppieren** eines Berichts einfach nicht auf.

Mit einem kleinen Trick ist jedoch auch dies möglich. Dazu erstellen Sie eine Abfrage, die das Memo-Feld und die übrigen benötigten Felder enthält. Fügen Sie der Abfrage ein weiteres Feld hinzu, das folgenden Ausdruck enthält (ausgehend davon, dass das Memo-Feld beispielsweise **Bemerkungen** heißt):

```
BemerkungenSort: Links([Bemerkungen];255)
```

Die Zeichenkettenfunktion **Links** ermittelt die ersten 255 Zeichen des Inhalts des Memo-Felds und gibt diese in einem neuen Feld der Abfrage aus. Da dieses Feld die für ein Textfeld zulässigen 255 Zeichen nicht überschreitet, wird es als Textfeld interpretiert und darf somit auch als Feld für eine Sortierung oder Gruppierung im Bericht verwendet werden (siehe Bild 9).

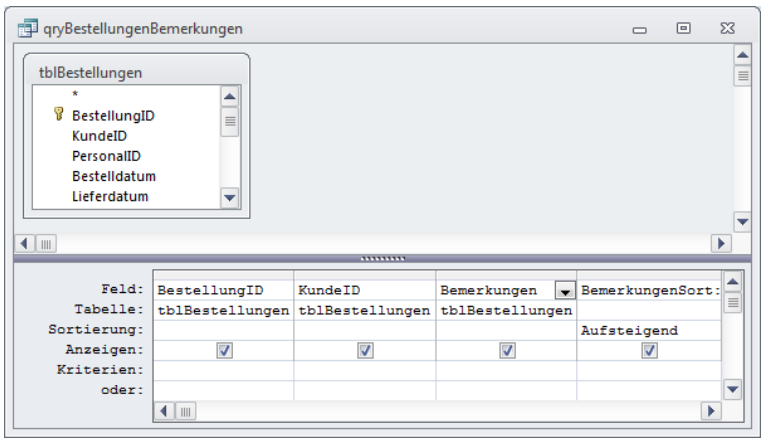


Bild 9: Abfrage mit Memo-Feld als Sortierkriterium

Datenbankkennwort vergeben

Auch wenn es Werkzeuge gibt, mit denen man das Datenbankkennwort zumindest älterer Access-Versionen knacken kann: Otto Normalverbraucher lässt in der Regel doch die Finger von solchen Methoden.

Sollten Sie also nur solchen Benutzern den Zugriff auf eine Datenbank gewähllten wollen, denen Sie ein dafür vorgesehenes Kennwort übermittelt haben, lässt sich dies unter Access leicht einrichten:

- Unter Access 2003 und älter rufen Sie dazu den Menübefehl **Extras|Sicherheit|Datenbankkennwort festlegen...** auf. Geben Sie ein Kennwort an und öffnen Sie die Datenbank erneut, um den Dialog zur Eingabe des Kennworts vorzufinden.

- Unter Access 2007 finden Sie den notwendigen Befehl im Ribbon unter **Datenbanktools|Datenbanktools|Datenbankkennwort festlegen**.

- Unter Access 2010 wurde der Befehl komplett verschoben, und zwar in den Backstage-Bereich. Also: Klick auf den Ribbon-Reiter **Datei**, dann zum Bereich **Informationen** wechseln und dort auf **Datenbankkennwort festlegen** klicken (siehe Bild 10).

Allen Vorgehensweisen ist gemein, dass Sie die Datenbank im exklusiven Modus öffnen müssen.

Dies gelingt am einfachsten, indem Sie zunächst Access öffnen, dann den Dialog zum Öffnen einer Datenbank aktivieren und dort nicht direkt auf die **Öffnen**-Schaltfläche klicken, sondern auf den Teil der Schaltfläche, der einen Pfeil nach unten anzeigt. Es erscheinen dann einige

weitere Befehle, unter anderem auch **Exklusiv öffnen** (variiert mit den verschiedenen Versionen).

Nach dem Schließen und erneuten Öffnen der Datenbank zeigt diese eine Inputbox zur Eingabe des Kennwort an.

Die Funktionen zum Ändern oder Entfernen des Kennworts finden Sie an gleicher oder ähnlicher Stelle wie die zum Anlegen des Kennworts.

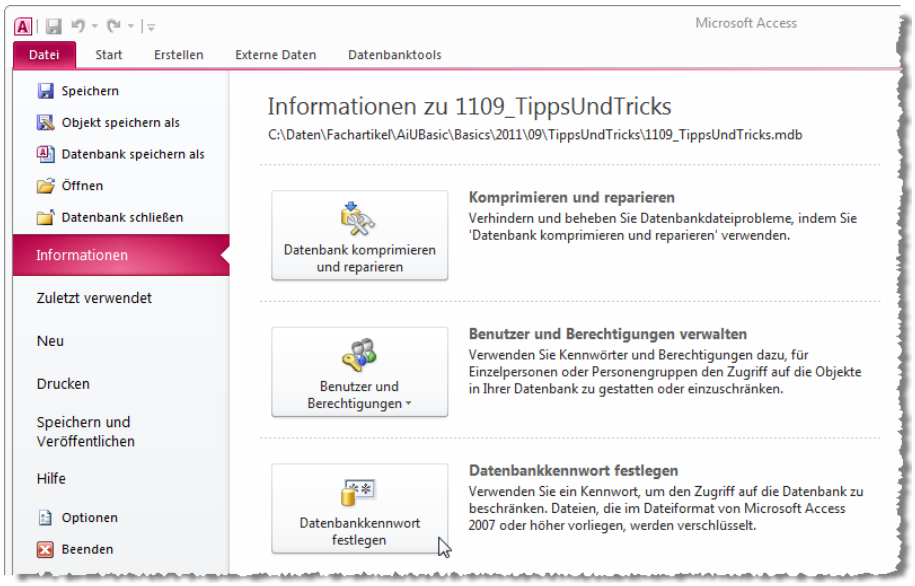


Bild 10: Festlegen des Datenbank-Kennworts unter Access 2010

Textbausteine verwenden

Wenn Sie zum Beispiel Anschreiben oder ähnliche Textdokumente mit Access als Bericht zusammenfügen, werden Sie diese Texte gelegentlich auch manuell eintippen. Dabei wäre es hilfreich, Standardfloskeln wie **Mit freundlichen Grüßen** et cetera durch Abkürzungen wie **mfg** in das entsprechende Textfeld eingeben zu können.

Das notwendige Werkzeug heißt **Autokorrektur** und wird von allen Office-Anwendungen eingesetzt, auch in Access. Das bedeutet, dass Eingaben in Textfeldern von der Autokorrektur geprüft und gegebenenfalls korrigiert werden.

Wenn Sie also etwa einen Punkt im Anschluss an ein Wort eintippen und das nächste Wort klein beginnen, wird die Autokorrektur aktiv. Sie korrigiert den vermeintlich fehlerhaften Text automatisch und teilt dies durch ein kleines Symbol mit. Mit einem Klick auf die beim Überfahren mit dem Mauszeiger erscheinende Schaltfläche zeigen Sie die Optionen der soeben durchgeführten Korrektur an (siehe Bild 11).

Mit einem Klick auf den Eintrag **Autokorrektur-Optionen steuern** öffnen Sie den Dialog aus Bild 12. Wenn Sie die Autokorrektur schlicht nicht mögen, deaktivieren Sie einfach alle Optionen dieses Dialogs.

Wenn Sie die Autokorrektur hingegen nutzen möchten, um Kürzel wie **mfg** automatisch in Texte wie **Mit freundlichen Grüßen** umzuwandeln, können Sie dies leicht bewerkstelligen. Geben Sie einfach unter **Ersetzen:** den Text **mfg** ein und unter **Durch:** den Text

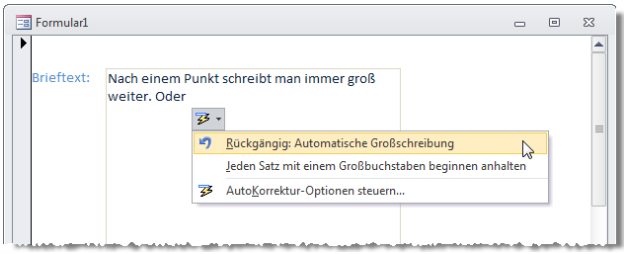


Bild 11: Die Autokorrektur in Aktion

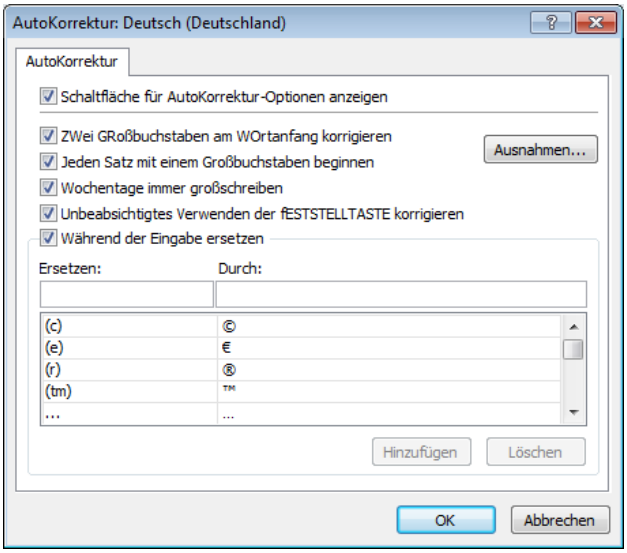


Bild 12: Optionen der Autokorrektur

Mit freundlichen Grüßen und klicken Sie auf **Hinzufügen** – schon ist die neue Floskel gespeichert.

Geben Sie diesen Text nun in ein Textfeld eines Access-Formulars ein, ersetzt Access diesen Text automatisch.