

# ACCESS

**BASICS**

DAS ACCESS-MAGAZIN FÜR ALLE,  
DIE VON 0 AUF 100 WOLLEN



**In diesem Heft:**

**ABFRAGEN [BASICS]: AKTIONSABFRAGEN (S. 3)**

**FORMULARE [BASICS]: EINFACHE DATENSÄTZE IM DETAIL (S. 16)**

**FORMULARE [BASICS]: DETAILFORMULARE EFFIZIENT NUTZEN (S. 24)**

**FORMULARE [BASICS]: SCHALTFLÄCHEN PROGRAMMIEREN (S. 30)**

**FORMULARE [BASICS]: OK- UND ABBRECHEN-BUTTONS (S. 36)**



André Minhorst Verlag

### Das neue Access [basics] Kompendium 2022 ist da!

Das neue Access [basics] Kompendium 2022 mit allen Ausgaben von 2010 bis 2022 ist erschienen.

Als aktueller Abonnent erhalten Sie das Kompendium derzeit zum Preis von 79 EUR statt 299 EUR. Sie müssen nur bei der Bestellung den Gutscheincode **komp100** eingeben.

Sie erhalten mit dem Kompendium alle Ausgaben in einem praktischen PDF mit über 3.000 Seiten. Außerdem können Sie passend dazu

ein Archiv mit allen Beispieldatenbanken der Magazine herunterladen und haben somit alles im Überblick – auch, wenn Sie mal offline sind.

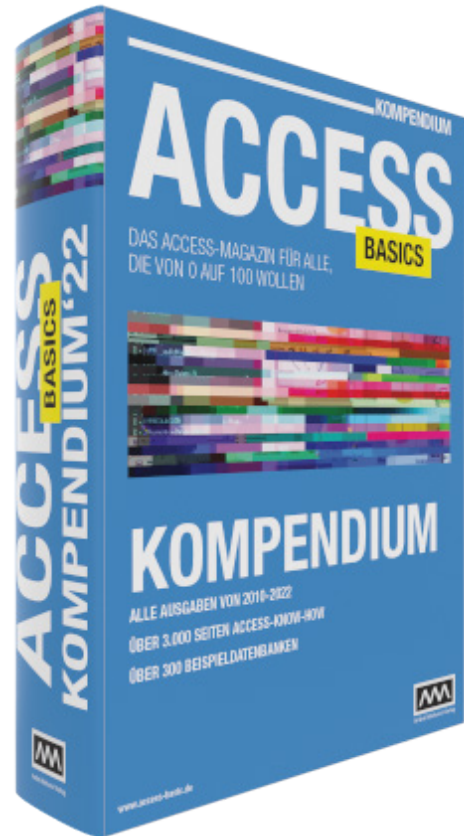
Hier ist der Link zur Bestellung des neuen Kompendiums:

<https://shop.minhorst.com/detail/index/sArticle/376/sCategory/76>

Für die Schweizer Kunden:

<https://elopage.com/s/amvs-hop/access-basics-kompendium-2022>

Viel Spaß beim Lesen!



## IMPRESSUM

### ACCESS [BASICS] WIRD HERAUSGEGEBEN VON:

Minhorst und Minhorst GbR - André Minhorst Verlag  
Borkhofer Straße 17  
47137 Duisburg

Die hier veröffentlichten Texte sind urheberrechtlich geschützt. Übersetzung und Vervielfältigung bedürfen der ausdrücklichen schriftlichen Genehmigung des Verlages. Sämtliche Veröffentlichungen in Access [basics] erfolgen ohne Berücksichtigung eines eventuellen Patentschutzes, auch werden Warennamen ohne Gewährleistung einer freien Verwendung benutzt. André Minhorst Fachverlag für Softwareentwicklung übernimmt für beschriebene oder zum Download bereitstehende Programme weder Gewähr noch Haftung, außer für Vorsatz

oder grobe Fahrlässigkeit. Bezugspreise erfahren Sie auf [www.access-basics.de](http://www.access-basics.de).

### REDAKTION:

André Minhorst (V.i.S.d.P)  
Telefon: 0203/4495577

E-Mail: [info@access-basics.de](mailto:info@access-basics.de)  
Internet: [www.access-basics.de](http://www.access-basics.de)

Geschäftsführung, Herstellung, Text- und Schlussredaktion,  
Layout von Magazin und Webseite: André Minhorst  
Autor: André Minhorst  
Fach- und Sprachlektorat: Carsten Gromberg

ISSN: 2190-8761

## Abfragen [basics]: Aktionsabfragen

Unter Access gibt es verschiedene Abfragetypen. Neben den verschiedenen Auswahlabfragen sind die Aktionsabfragen ebenfalls wichtiger Bestandteil. Auch Aktionsabfragen kann man prima über den Abfrageentwurf programmieren. Dieser Artikel stellt die vier verschiedenen Aktionsabfragen vor, mit denen Du Datensätze zu einer Tabelle hinzufügen, die Daten bestehender Datensätze ändern, bestehende Datensätze löschen oder auch neue Tabellen auf Basis einer Auswahlabfragen anlegen kannst.

### Beispieldatenbank

Die Beispiele dieses Artikels findest Du in der Datenbank **AbfragenBasics\_Aktionsabfragen.accdb**.

### Arten von Aktionsabfragen

Es gibt vier verschiedene Typen von Aktionsabfragen:

- **Anfügeabfragen:** Fügen neue Datensätze zu einer vorhandenen Tabelle hinzu.
- **Aktualisierungsanfragen:** Aktualisieren bestehende Datensätze einer Tabelle.
- **Löschabfragen:** Löschen Datensätze aus einer Tabelle.
- **Tabellenerstellungsabfragen:** Erstellen eine neue Tabelle auf Basis der angegebenen Auswahlabfrage.

### Erstellen einer Anfügeabfrage

Wenn wir die Ribbonbefehle im Bereich **Erstellen|Abfragen** betrachten, suchen wir vergeblich nach Einträgen zum Erstellen der oben aufgeführten Abfragetypen (siehe Bild 1). Hier ist allein der Befehl

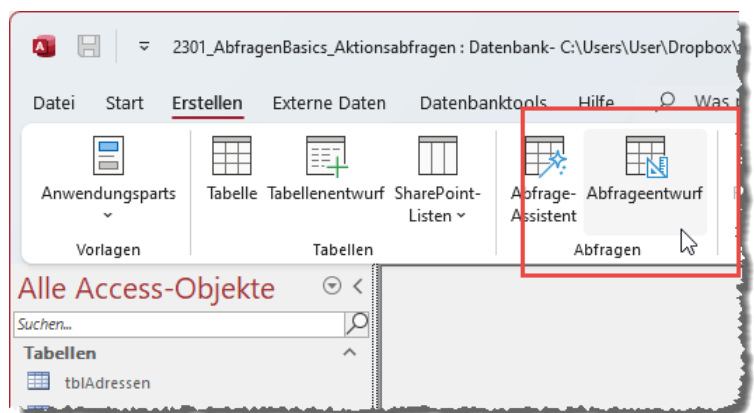


Bild 1: Anlegen einer neuen Abfrage als Basis für eine Aktionsabfrage

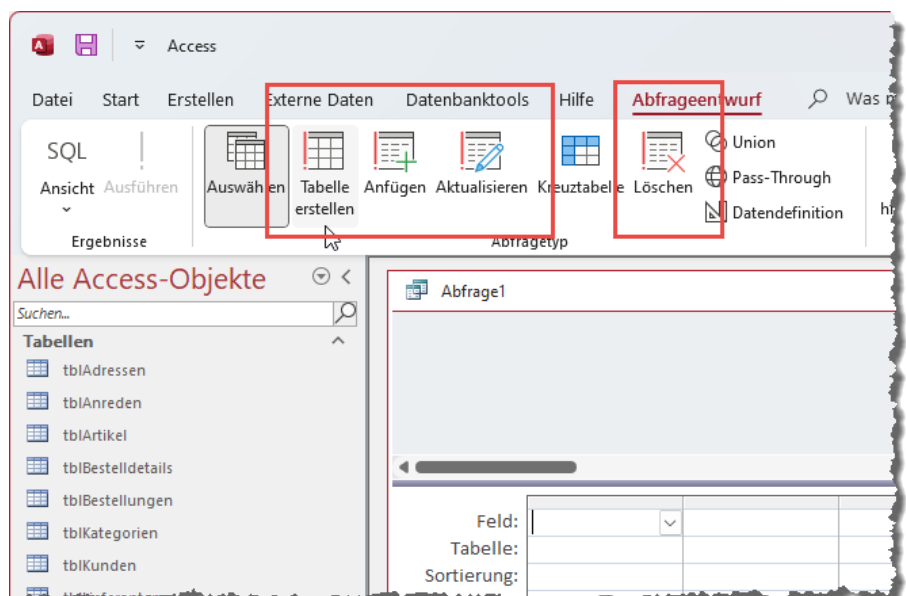


Bild 2: Die Befehle tauchen erst im Entwurf einer neuen Auswahlabfrage auf.

Abfrageentwurf zu finden, und auch ein Klick auf die Schaltfläche **Abfrage-Assistent** eröffnet keine offensichtlichen Möglichkeiten zum Erstellen von Aktionsabfragen.

Wenn wir allerdings einmal auf **Abfrageentwurf** klicken und somit eine neue, leere Abfrage in der Entwurfsansicht anzeigen, erscheinen im Ribbon unter **Abfrageentwurf** die gewünschten Einträge (siehe Bild 2).

### Datensätze löschen per Löschartfrage

Die Löschartfrage ist am einfachsten aufgebaut, daher schauen wir uns diese als Erstes an. Ausgehend von einer frisch angelegten Abfrage ziehen wir als Erstes die Tabelle in den Abfrageentwurf, aus der wir Datensätze löschen wollen – zum Beispiel die Tabelle **tblKunden**.

Nachdem wir die Tabelle gespeichert haben, fügen wir das Feld **KundeID** zum Entwurfsraster hinzu und klicken im Ribbon auf die Schaltfläche **Abfrageentwurf|Abfragetyp|Löschen** (siehe Bild 3).

Dies ändert einige Zeilen im Entwurfsraster:

- die Zeilen **Sortierung** und **Anzeigen** fallen weg und
- die Zeile **Löschen** wird eingblendet.

Ohne weitere Änderungen wie beispielsweise das Setzen eines Kriteriums können wir nun mit dem Ribbonbefehl **Abfrageentwurf|Ergebnisse|Ansicht|Datenblattansicht** die voraussichtlich von der Aktionsabfrage betroffenen Datensätze anzeigen (siehe Bild 4).

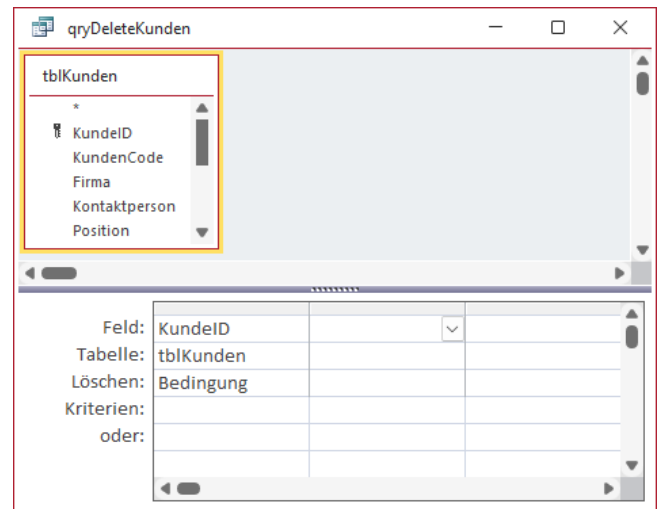


Bild 3: Erster Entwurf einer Löschartfrage

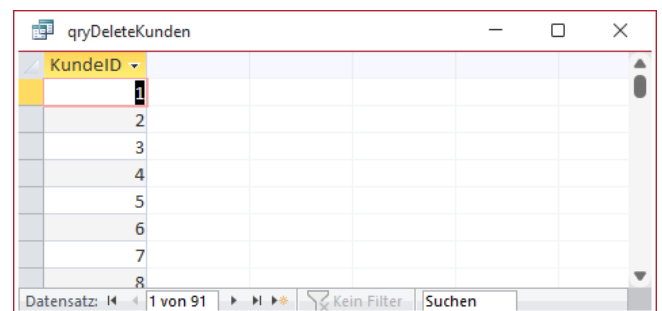


Bild 4: Voraussichtlich von der Aktion betroffene Datensätze

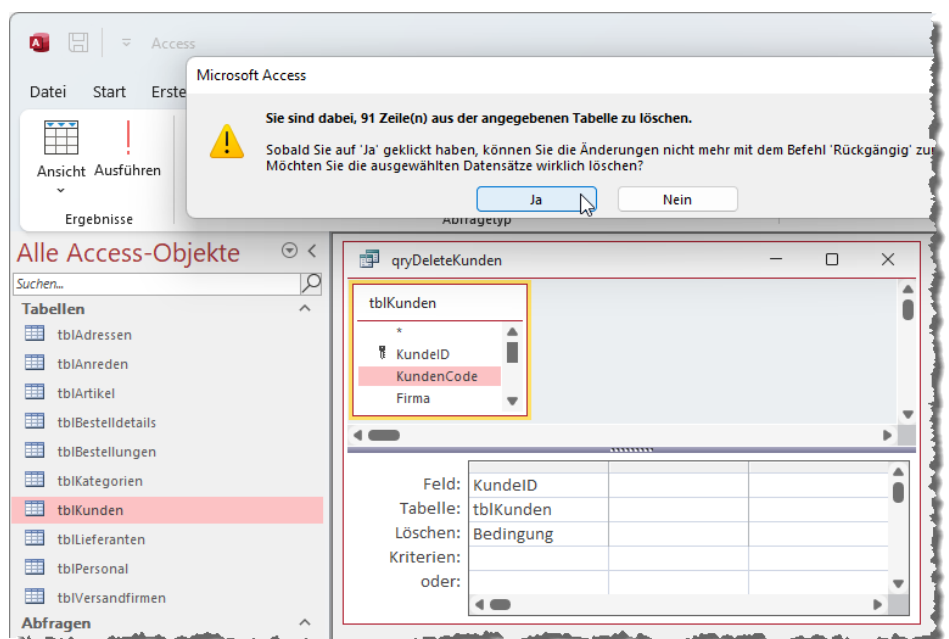


Bild 5: Meldung vor dem Löschen von Datensätzen

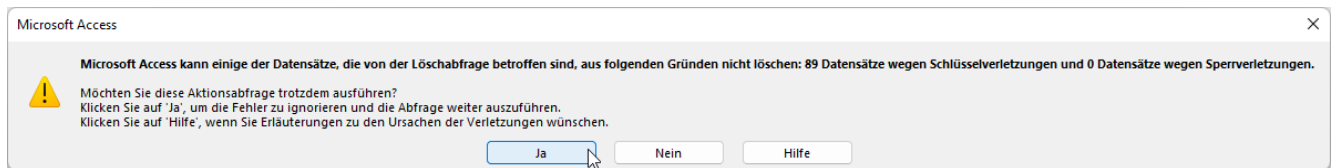


Bild 6: Meldung wegen Schlüsselverletzungen

Dies sind allerdings nur die Datensätze, die durch die aktuellen Kriterien ausgewählt werden würden. Welche tatsächlich gelöscht werden, hängt noch von anderen Faktoren ab – zum Beispiel davon, ob die Daten dieser Tabelle gegebenenfalls noch mit anderen Tabellen verknüpft sind und eine Löschartbeschränkung vorliegt. Es könnte zum Beispiel sein, dass ein Kunde bereits mit einem Datensatz der Tabelle `tblBestellungen` verknüpft ist, was ein Löschart verhindern würde.

Die tatsächliche Aktion der Abfrage führen wir dann mit einem Klick auf die Schaltfläche **Ausführen** im Ribbon aus. Dies zeigt zunächst noch eine Meldung an, die der Benutzer bestätigen muss (siehe Bild 5). Erst wenn das geschehen ist, wird die Aktionsabfrage ausgeführt.

In unserem Beispiel geschieht das, was wir bereits beschrieben haben: Die meisten Datensätze sind bereits mit Bestellungen verknüpft und werden nicht gelöscht (siehe Bild 6).

### Kriterien zur Löschartfrage hinzufügen

Eigentlich kommt eine Löschartfrage mit einem Feld aus. Erst wenn wir Kriterien für die zu löschenden Datensätze definieren wollen, müssen wir die Felder hinzufügen, die für die Festlegung der Kriterien benötigt werden. Wenn wir also beispielsweise alle Kunden löschen wollen, die noch keiner

Bestellung zugeordnet sind, und somit die Meldung von oben umgehen wollen, können wir eine Unterabfrage als Kriterium des Feldes `KundeID` hinterlegen:

```
Nicht In (SELECT KundeID FROM tblBestellungen
WHERE tblBestellungen.KundeID = tblKunden.KundeID)
```

Nach dem Hinzufügen dieser Unterabfrage (siehe Bild 7) löscht die Löschartfrage nur noch die beiden Datensätze der Tabelle `tblKunden`, die noch keiner Bestellung zugeordnet sind und so verhindern wir auch die Anzeige der zweiten Meldung.

### Bestätigungen beim Ausführen von Aktionsabfragen unterbinden

Wenn Du nicht möchtest, dass die Meldungen angezeigt werden, mit denen der Benutzer angeben kann, ob er die Abfrage tatsächlich durchführen will, kannst Du diese deaktivieren. Dazu öffnest Du mit **Datei|Optionen** die Access-Optionen und wechselst dort zum Bereich **Clienteneinstellungen**. Hier findest Du unter **Bearbeiten** relativ weit unten die Option **Bestätigen**. Wenn Du hier den Haken bei Aktionsabfragen

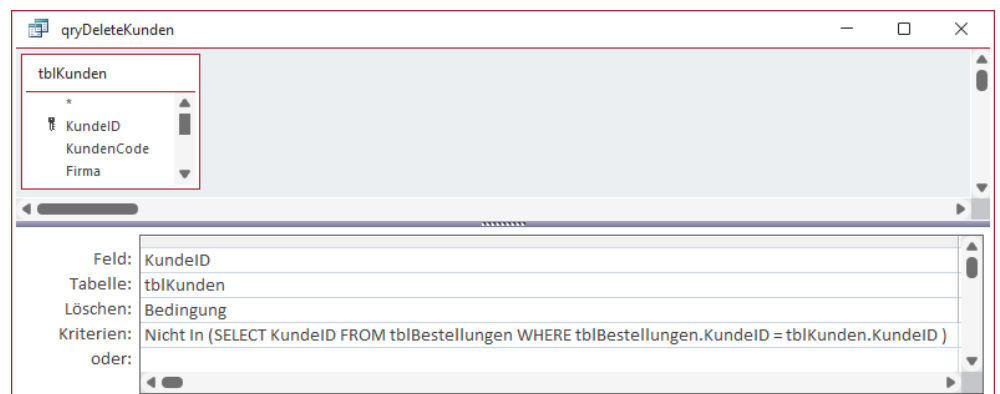


Bild 7: Unterabfrage als Kriterium vor dem Löschart von Datensätzen

entfernt, werden solche Meldungen in Zukunft nicht mehr angezeigt (siehe Bild 8).

Der Bereich **Clienteneinstellungen**, in dem sich diese Option befindet, deutet darauf hin, dass die Einstellung nur für den aktuellen Client gilt und beim Weitergeben der Datenbank nicht übernommen wird – dazu müsste man andere Schritte unternehmen, deren Beschreibung an dieser Stelle den Rahmen sprengen würde.

### Datensätze ändern per Aktualisierungsabfrage

Das Ändern der Daten einer Tabelle kann verschiedene Gründe haben. Vielleicht möchtest Du leere Felder mit Inhalten füllen, vorhandene Inhalte ändern oder vielleicht sollen auch vorhandene Daten aus Feldern entfernt werden. Die Vorgehensweise ist immer ähnlich, nur wenn die Änderung eines Feldes auf Basis der vorhandenen Daten geschehen soll, dann gibt es einen kleinen Unterschied.

### Feldinhalte ersetzen per Aktualisierungsabfrage

Im ersten Beispiel wollen wir die Inhalte eines Feldes ersetzen, wenn dieses einen bestimmten Wert aufweist. Wir wollen die Daten der Tabelle **tblKunden** international machen und die Bezeichnungen der Länder auf Englisch umstellen – zunächst nur für Deutschland.

Dazu erstellen wir zuerst wieder eine neue, leere Abfrage in der Entwurfsansicht. Dieser fügen wir die Tabelle **tblKunden** hinzu. Dann ziehen wir die beiden Felder **KundeID** und **Land** in das Entwurfsraster.

Um die Auswahlabfrage in eine Aktualisierungsabfrage umzuwandeln, klicken wir im Ribbon auf die Schaltfläche **Abfrageentwurf|Abfragetyp|Aktualisieren**.

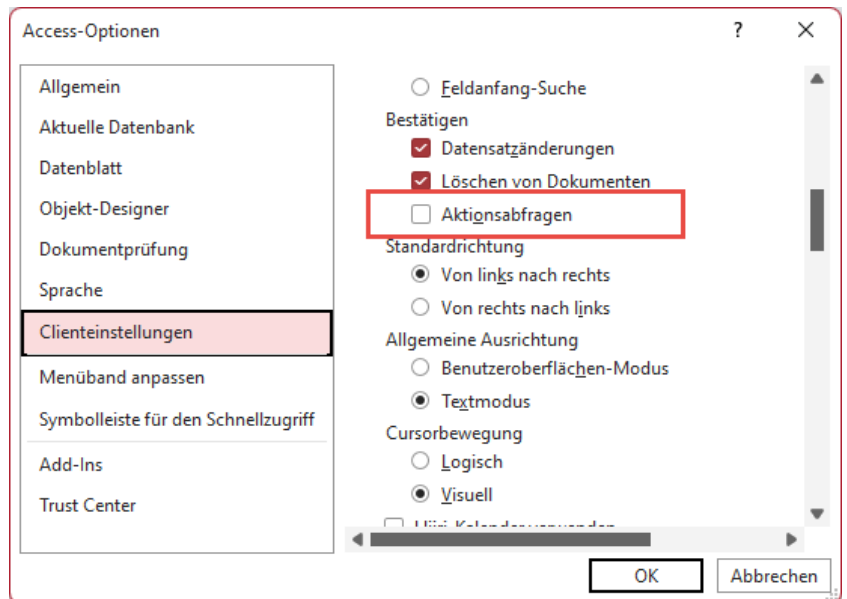


Bild 8: Einstellen der Optionen für Aktionsabfragen

Dies fügt die Zeile **Aktualisieren** hinzu, der wir im Feld **Land** den Wert **"Germany"** hinzufügen (siehe Bild 9).

Entscheidend ist nun, dass wir für die gleiche Spalte noch das Kriterium **"Deutschland"** angeben, denn sonst würde die Abfrage für alle Datensätze den Wert **Germany** eintragen – auch für solche, die im Feld **Land** einen anderen Wert als **Deutschland** aufweisen.

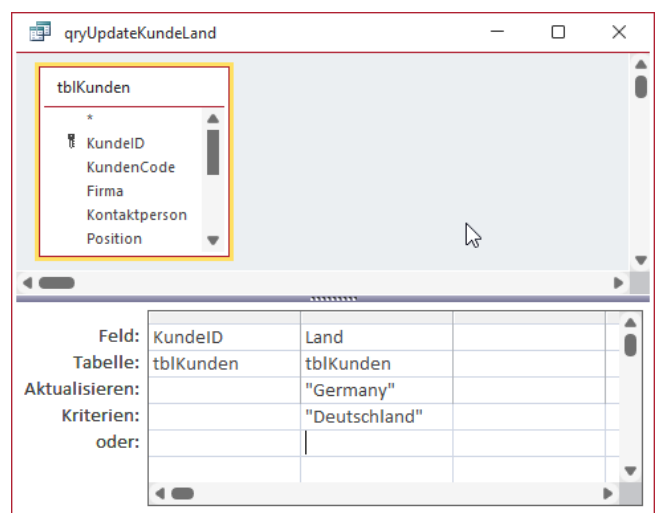


Bild 9: Abfrage zum Ändern des Landes von Deutschland in Germany

Anschließend können wir die Abfrage wie bereits bei der Löschabfrage noch testen, das heißt, wir können anzeigen, auf welche Datensätze sich die Änderungen auswirken. Das Ergebnis aus Bild 10 zeigt, dass wir wohl die richtigen Datensätze erwischen werden.

Wir wollen die Aktionsabfrage diesmal nicht mit dem Ribbonbefehl **Abfrageentwurf|Ergebnisse|Ausführen** starten, sondern schließen diese und klicken diesmal doppelt auf den Namen der Abfrage im Navigationsbereich von Access. Je nachdem, ob Du die Anzeige von Bestätigungsmeldungen wie oben beschrieben deaktiviert hast, erscheint nun noch eine Rückfrage oder Access ändert die betroffenen Datensätze direkt.

### Feldinhalte ändern per Aktualisierungsabfrage

Der Unterschied zu der zuvor beschriebenen Abfrage ist, dass wir diesmal den Feldwert auf Basis des zuvor enthaltenen Werts anpassen wollen. Dazu gibt es verschiedene Anlässe:

- Datumsangaben um eine bestimmte Menge von Tagen nach vorn oder hinten verändern
- Preis um x Prozent erhöhen oder senken
- Bestimmten Texten Informationen hinzufügen oder Teile der Texte ersetzen

In allen Fällen wollen wir den neuen Inhalt des Feldes aufbauend auf dem alten Inhalt gestalten und müssen daher auf den Feldinhalt zugreifen.

Im folgenden Beispiel wollen wir die Preise aller Artikel der Kategorie **Gewürze** um 10 Prozent erhöhen. Dazu erstellen wir eine neue Abfrage, der wir die Tabelle **tblArtikel** hinzufügen. Aus dieser Tabelle ziehen wir die Felder **KategorieID** und **Einzelpreis** in das Entwurfsraster. Für das Feld **KategorieID** legen wir den Wert **2** als Kriterium fest. Diesen Wert müssen wir zuvor aus der Tabelle **tblKategorien** für die Kategorie **Gewürze** ermitteln.

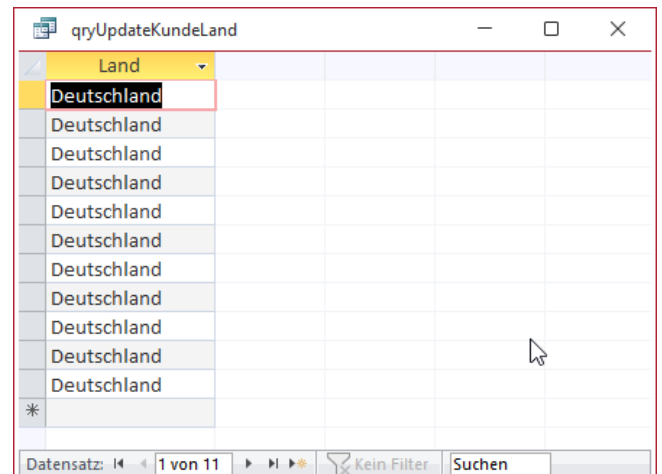


Bild 10: Testen der Abfrage zum Ändern des Landes

Danach wandeln wir die Abfrage wieder in eine Aktualisierungsabfrage um. In der Zeile Aktualisieren tragen wir nun für die Spalte Einzelpreis den folgenden Wert ein:

[Einzelpreis]\*1,1

Den Feldnamen müssen wir in eckige Klammern einfassen, da der Ausdruck sonst als Zeichenkette

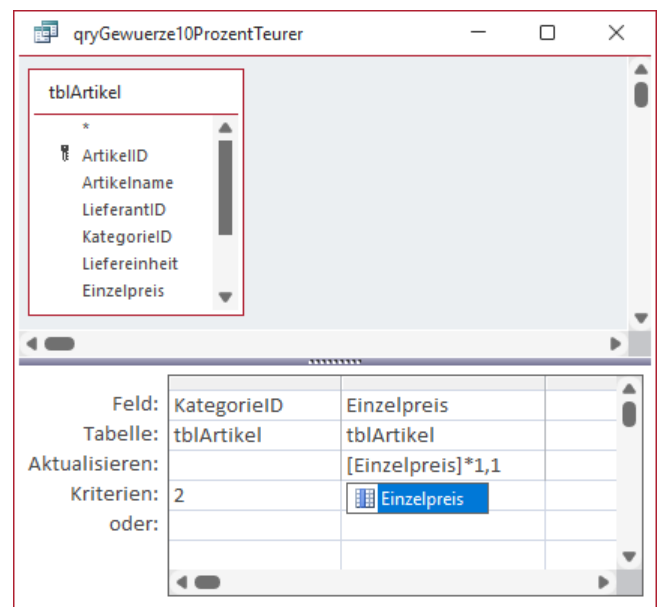


Bild 11: Aktionsabfrage zum Erhöhen des Preises für die Artikel einer Kategorie um zehn Prozent

## Formulare [basics]: Einfache Datensätze im Detail

Eine der wichtigsten Aufgaben für Formulare ist die Anzeige der Daten eines einzelnen Datensatzes – und zwar in einer Form, in welcher der Benutzer die Daten komfortabel eingeben kann. Dazu gehört, dass man zunächst die Tabelle oder Abfrage, deren Daten im Formular erscheinen sollen, als Datensatzquelle angibt. Dann nutzt man die praktische Feldliste, aus der wie alle Felder der Datenquelle einfach in das Formular ziehen kann. Schließlich passen wir die Elemente noch an und erhalten so ein Formular zur Anzeige der Details eines Datensatzes.

### Beispieldatenbank

Die Beispiele dieses Artikels findest Du in der Datenbank **FormulareBasics\_Grundlagen.accdb**.

### Anlegen des Formulars

Das neue Formular legen wir über den Menübefehl **Erstellen|Formulare|Formularentwurf** an. Wir erhalten so ein neues, leeres Formular in der Entwurfsansicht (siehe Bild 1).

### Formular speichern

Als erste Aktion speichern wir das Formular nun. Dazu betätigen wir die Tastenkombination **Strg + S**. Sofern das Formular zuvor noch nicht gespeichert wurde, erscheint der Dialog **Speichern unter**, wo wir den Formularnamen **frmLeserDetails** eingeben (siehe Bild 2).

Direkt danach ändert sich der Name im Titel des Formulars auf den angegebenen Wert.

Wechseln wir danach in die Formularansicht (zum Beispiel mit **For-**

**mularentwurf|Ansichten|Formularansicht**), sehen wir ein komplett leeres Formular (siehe Bild 3). Zu beachten ist der untere Bereich mit den Navigationsschaltflächen: Hier werden **1 von 1** Datensätze angezeigt, aber man kann keinen neuen Datensatz anlegen. Logisch: Wir haben ja auch noch keine Datensatzquelle hinterlegt, in der ein neuer Datensatz hinterlegt werden könnte.

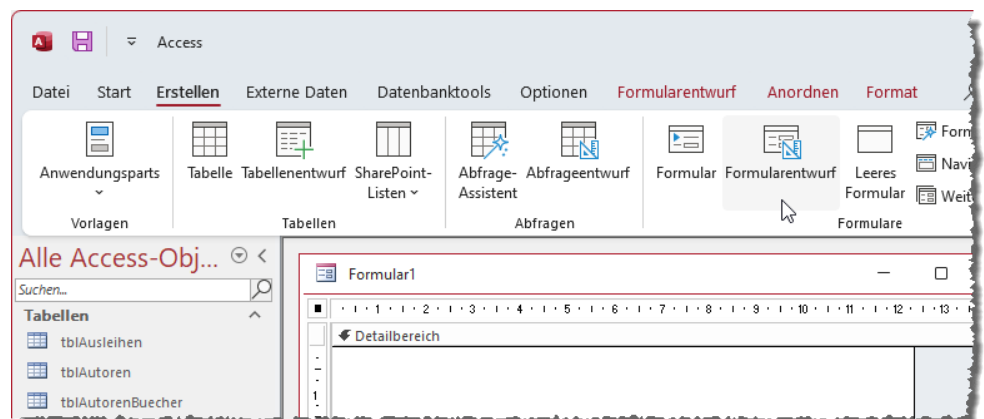


Bild 1: Anlegen des neuen Formulars

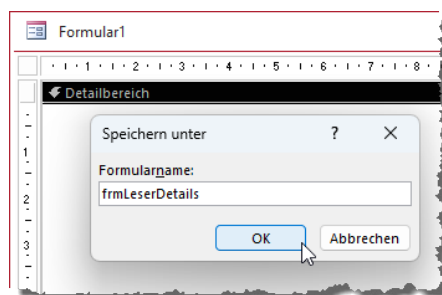


Bild 2: Speichern des Formulars

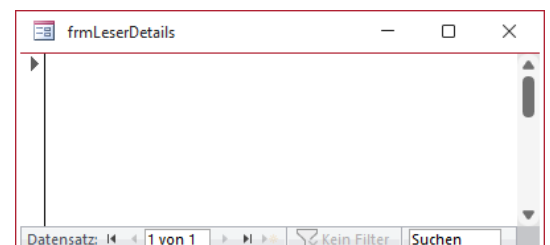


Bild 3: Das Formular in der Formularansicht – hier noch ohne Daten



### Formular erneut entwerfen

Wenn das Formular einmal geschlossen ist und Du willst es erneut bearbeiten, gelingt das am schnellsten mit einem Rechtsklick auf den Formularnamen im Navigationsbereich von Access und Auswahl des Kontextmenü-Eintrags **Entwurfsansicht**.

### Schritte zum Anzeigen von Daten im Formular

Unser neues Formular soll die Daten eines Datensatzes der Tabelle **tblLeser** anzeigen. Dazu sind folgende Schritte nötig:

- Zuweisen dieser Tabelle als Datensatzquelle des Formulars
- Hinzufügen von Steuerelementen, die an die Felder dieser Tabelle gebunden sind

### Zuweisen der Datensatzquelle

Wird das Formular in der Entwurfsansicht angezeigt, erscheint üblicherweise auch das **Eigenschaftenblatt**. Ist dies nicht der Fall, blenden wir dieses nun mit der Taste **F4** ein.

Hier wechseln wir zur Registerseite **Daten**. Diese hält die Eigenschaft **Datensatzquelle** bereit, mit der wir alle Tabellen oder Abfragen der Datenbank auswählen können. Wir wählen hier den Eintrag **tblLeser** aus (siehe Bild 4).

Welche Änderung erreichen wir damit? Wechseln wir erneut in die **Formularansicht**, sehen wir im unteren Bereich zwischen den Navigationsschaltflächen die Anzeige **1 von 10** (siehe Bild 5).

Und auch wenn wir noch keine Steuerelemente zur Anzeige von Daten hinzugefügt haben, sehen wir hier schon, dass das Formular grundsätzlich Daten enthält. Wir können sogar schon darin navigieren.

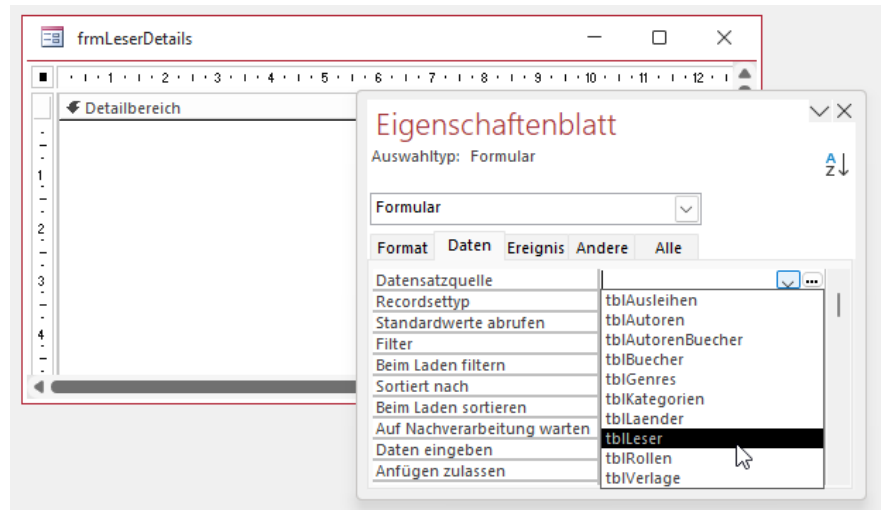


Bild 4: Zuweisen der Datensatzquelle

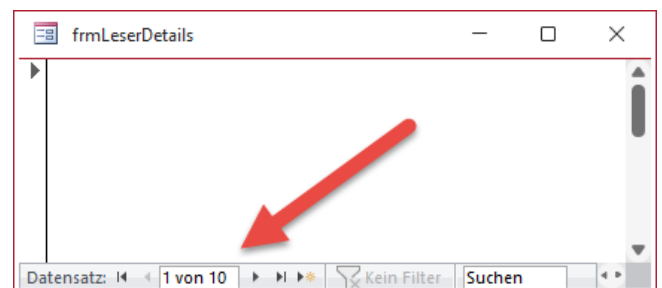


Bild 5: Das Formular enthält bereits Daten.

### Steuerelemente mit Daten hinzufügen

Access ist nicht umsonst als die Datenbankanwendung unter Windows bekannt, mit der sich am schnellsten nutzbare Anwendungen erstellen lassen. Einer der Gründe dafür ist, dass wir nach dem Auswählen einer Datensatzquelle für ein Formular direkt gebundene Steuerelemente auf Basis der Felder dieser Tabelle erstellen können.

Dazu blenden wir mit dem Menübefehl **Formularentwurf|Tools|Vorhandene Felder hinzufügen** die sogenannte **Feldliste** ein.

Diese zeigt alle Felder der Tabelle oder Abfrage an, die wir als Datensatzquelle angegeben haben. Wir markieren durch Aufziehen eines Rahmens mit der Maus alle Felder und ziehen diese in den Detailbereich des **Formularentwurfs** (siehe Bild 8).

### Was automatisch passiert

Durch das Hineinziehen der Steuerelemente sind verschiedene Dinge geschehen:

- Es wurde jeweils ein Textfeld für jedes der Felder angelegt.
- Diesem wurde ein Bezeichnungsfeld hinzugefügt. Dass das Textfeld und das Bezeichnungsfeld verbunden sind, erkennen wir erstens daran, dass wenn wir eines der beiden Elemente markieren, auch das kleine graue Kästchen am anderen Element erscheint. Außerdem können wir in der Eigenschaft **Bezeichnungname** des Textfeldes den Namen des verknüpften Bezeichnungsfeldes entnehmen (siehe Bild 6).
- Als Name für das Textfeld wurde der Name des Feldes eingetragen. Dieser Name ist wichtig, wenn Du später einmal per VBA auf das Feld zugreifen möchtest – oder den Inhalt des Textfeldes als Teil eines berechneten Ausdrucks eines anderen Steuerelements zu nutzen.

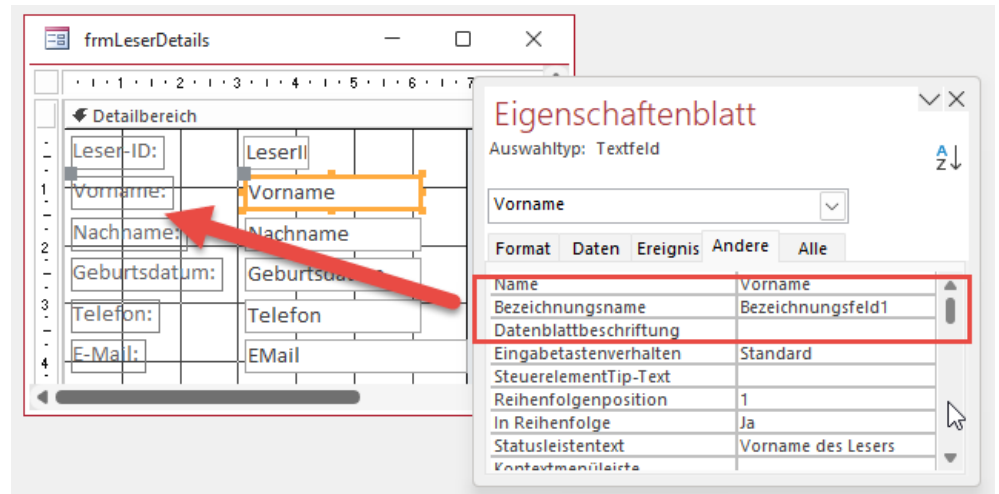


Bild 6: Angabe des Bezeichnungsfeldes eines Textfeldes

- Für das Textfeld wurde außerdem die Eigenschaft **Steuerelementinhalt** auf den Namen des Feldes eingestellt, das wir aus der Feldliste in den Formularentwurf gezogen haben. Dies gibt an, aus welchem Feld der Datensatzquelle dieses Steuerelement seinen Wert für den aktuellen Datensatz beziehen soll.

### Resultierende Bindungen

Wir haben nun also auf zwei Ebenen Bindungen zwischen dem Formular und der Tabelle **tblLeser** erstellt:

- Wir haben das Formular **frmLeserDetails** über die Eigenschaft **Datensatzquelle** an die Tabelle **tblLeser** gebunden.
- Wir haben Steuerelemente hinzugefügt, die automatisch an die Felder dieser Datensatzquelle gebunden, damit die Daten des aktuellen Datensatzes in den Steuerelementen angezeigt werden.



Bild 7: Ändern des Namens eines Textfeldes

### Optimierungsidee Steuerelementname

Wie Du siehst, lautet der Name der Textfelder nun jeweils wie der Name des Feldes, an welches das Feld gebunden ist. Das kann mitunter zu Verwirrung führen. Daher schlage ich vor, dass wir die Namen von gebundenen Steuerelementen so ändern, dass diese ein Präfix erhalten. Dieses Präfix soll gleichzeitig andeuten, um was für eine Art von Steuerelement

## Formulare [basics]: Detailformulare effizient nutzen

Im Artikel [Formulare \[basics\]: Einfache Datensätze anzeigen \(www.access-basics.de/621\)](http://www.access-basics.de/621) haben wir gezeigt, wie man die Daten einer einfachen Tabelle in einem Formular anzeigt. Damit kannst Du ein Formular über den Navigationsbereich aufrufen, aber Du weißt noch nicht, wie Du es beispielsweise aufrufst, um einen bestimmten Datensatz anzuzeigen, wie Du dem Benutzer eine intuitive Möglichkeit zum Speichern oder Verwerfen der bearbeiteten Daten bietest oder wie Du verschiedene Elemente ausblendest, die Du vielleicht gerade gar nicht benötigst. In diesem Artikel liefern wir einige Grundlagen dazu und verweisen auf weitere Artikel, die spezielle Themen vertiefen.

### Beispieldatenbank

Die Beispiele dieses Artikels findest Du in der Datenbank **FormulareBasics\_Detailformulare.accdb**.

### Blättern oder nicht?

Die erste Frage, die wir uns stellen, wenn wir ein Formular zur Anzeige einzelner Datensätze anbieten, ist die nach der Navigation. Standardmäßig blendet Access Navigationsschaltflächen unten im Formular ein (siehe Bild 1). Diese erlauben das Blättern zum ersten, letzten, vorherigen und letzten Datensatz sowie zu einem neuen, leeren Datensatz. Außerdem können wir in das Textfeld zwischen den Schaltflächen noch den Index eines Datensatzes angeben, zu dem wir springen wollen.

den. In diesem Fall sind die Navigationsschaltflächen nicht nur unnötig, sondern sie stören vielleicht sogar. Wir können diese Elemente allerdings ausblenden.

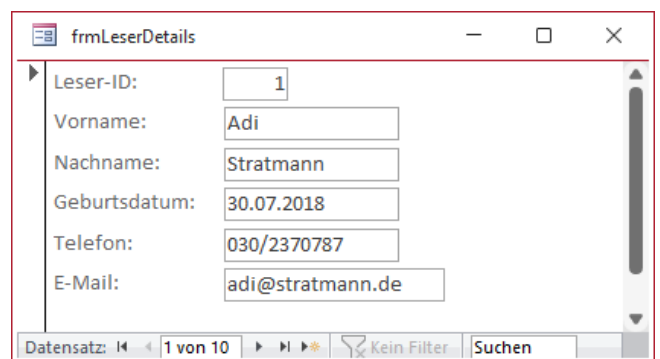


Bild 1: Formular mit Navigationsschaltflächen

### Navigationsschaltflächen ausblenden

Das ist alles recht praktisch, wenn man dem Benutzer die Möglichkeit bieten möchte, im Formular zwischen den Datensätzen der Datenquelle zu navigieren. In vielen Fällen soll ein solches Formular aber gar nicht zum Navigieren, sondern nur zum Anlegen eines neuen Datensatzes oder zum Anzeigen eines bestehenden Datensatzes verwendet werden.

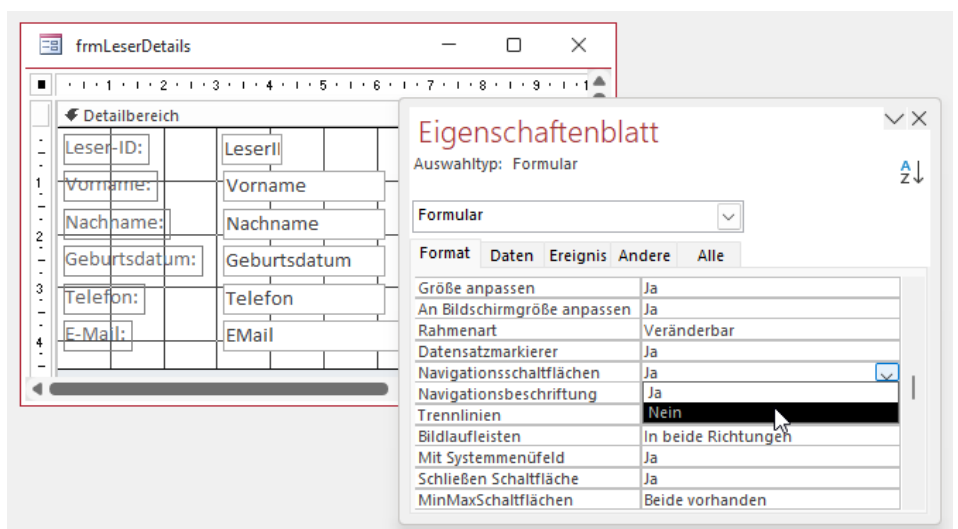


Bild 2: Deaktivieren der Navigationsschaltflächen

Dazu wechseln wir in die Entwurfsansicht des Formulars und aktivieren das Eigenschaftenblatt mit **F4**, sofern dieses noch nicht sichtbar ist. Dort finden wir auf der Registerseite **Format** die Eigenschaft **Navigationsschaltflächen**, mit der wir diese ausblenden können. Dazu stellen wir die Eigenschaft auf den Wert **Nein** ein (siehe Bild 2).

Wechseln wir wieder in die Formularansicht, sind die Navigationsschaltflächen ausgeblendet (siehe Bild 3). Damit sieht das Formular fast ein wenig kahl aus.

### Datensatzmarkierer anzeigen oder nicht?

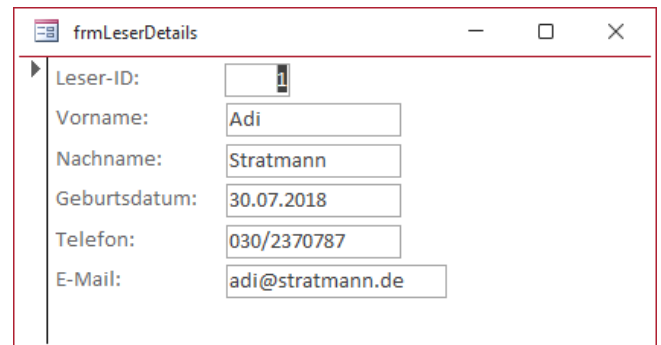
Der Datensatzmarkierer, also der Bereich links im Formular, hat folgende Aufgaben:

- Anzeigen eines Symbols, das den aktuellen Status des Formulars anzeigt.
- Markieren des kompletten Datensatzes, um diesen beispielsweise zu löschen, zu kopieren oder auszuschneiden.
- Speichern eines in Bearbeitung befindlichen Datensatzes

Der Datensatzmarkierer kann drei verschiedene Symbole anzeigen:

- Ein nach rechts zeigendes Dreieck für einen unbearbeiteten Datensatz,
- einen Stift für einen in Bearbeitung befindlichen Datensatz oder
- ein Symbol, das anzeigt, dass der Datensatz gesperrt ist und nicht bearbeitet werden kann.

Benötigen wir den Datensatzmarkierer nun? Das kommt darauf an, ob und welche Informationen wir dem Benutzer über den Bearbeitungszustand des Datensatzes geben wollen. Wenn wir diesen Bereich weglassen, können wir dessen Aufgaben teilweise durch eine OK-Schaltfläche abbilden. Diese könn-



The screenshot shows a window titled 'frmLeserDetails' with a standard Windows title bar. The form contains several text boxes with the following values: 'Leser-ID:' (empty), 'Vorname:' (Adi), 'Nachname:' (Stratmann), 'Geburtsdatum:' (30.07.2018), 'Telefon:' (030/2370787), and 'E-Mail:' (adi@stratmann.de). The left side of the form, where the data set marker would typically be, is empty.

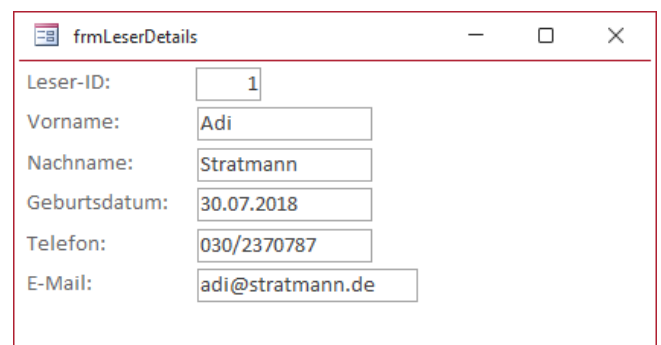
Bild 3: Formular ohne Navigationsschaltflächen

ten wir aktivieren, wenn sich der aktuell angezeigte Datensatz in Bearbeitung befindet, also wenn sich seit dem Anzeigen oder dem letzten Speichern Änderungen ergeben haben, und deaktivieren, wenn der aktuelle Zustand dem gespeicherten Zustand entspricht. Wie wir das abbilden können, zeigen wir im Artikel **Formulare [basics]: OK- und Abbrechen-Buttons** ([www.access-basics.de/627](http://www.access-basics.de/627)).

Zusätzlich würden wir noch eine **Abbrechen**-Schaltfläche hinzufügen, die der Benutzer statt der **Esc**-Taste zum Verwerfen ungespeicherter Änderungen nutzen kann.

### Datensatzmarkierer ausblenden

Um den Datensatzmarkierer auszublenden, wechseln wir wieder in die Entwurfsansicht und stellen die Eigenschaft **Datensatzmarkierer** auf **Nein** ein. In der Formularansicht sieht das Formular nun noch aufgeräumter aus (siehe Bild 4).



The screenshot shows the same 'frmLeserDetails' window. The form fields are identical to Bild 3. The 'Leser-ID' field now contains the number '1'. The left side of the form remains empty, indicating the data set marker is hidden.

Bild 4: Formular ohne Datensatzmarkierer

### Titelleiste des Formulars anpassen

Damit können wir uns um die Titelleiste des Formulars kümmern, die standardmäßig ein Formular-Icon und den Namen des Formulars anzeigt. Außerdem finden wir dort drei Schaltflächen zum Minimieren, Maximieren und Schließen des Formulars.

Weitere Elemente finden wir vor, wenn wir links oben auf das Icon in der Titelleiste klicken (siehe Bild 5).

Die erste Anpassung ist die des Textes in der Titelleiste. Dazu stellen wir die Eigenschaft **Beschriftung** des Formulars ein, beispielsweise auf **Leser - Detailansicht** (siehe Bild 6).

### Systemfeldmenü

Das oben erwähnte Menü und die Schaltflächen können wir komplett ausblenden, indem wir die Eigenschaft **Systemfeldmenü** auf **Nein** einstellen. Damit verschwindet allerdings auch das Formular-Icon (siehe Bild 7). Dafür erscheint, wenn wir auch die Navigationsschaltflächen ausgeblendet haben, unten ein unschöner grauer Bereich.

### Schließen-Schaltfläche deaktivieren

Wenn wir hingegen die Eigenschaft **Schließen Schaltfläche** auf **Nein** einstellen, verschwindet der **Schließen**-Befehl nicht aus dem Systemfeldmenü und auch die **Schließen**-Schaltfläche wird nicht ausgeblendet. Beide werden einfach nur deaktiviert (siehe Bild 8).

### Minimieren und Maximieren ausblenden

Durch Einstellen der Eigenschaft **MinMaxSchaltflächen** können wir diese beiden Schaltflächen rechts oben ausblenden. Im Systemfeldmenü werden diese allerdings noch angezeigt – wenn auch deaktiviert (siehe Bild 9).

### Bildlaufleisten

Je nachdem, wie groß ein Formular ist, kann es sinnvoll sein, Bildlaufleisten zu unterstützen. Das ist standardmäßig der Fall. Wenn wir diese Option nicht wünschen, können wir diese mit der Eigenschaft Bild-

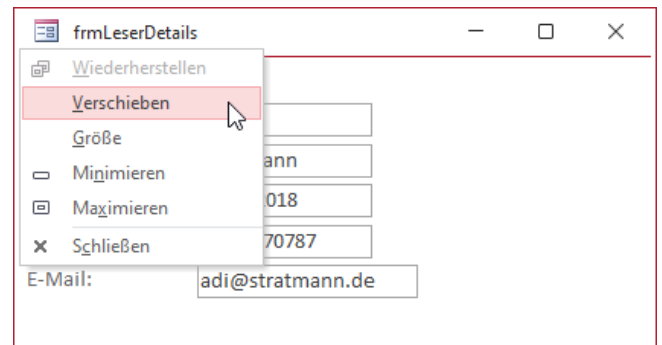


Bild 5: Formularmenü

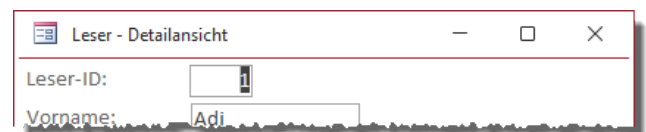


Bild 6: Individuelle Beschriftung



Bild 7: Ausgeblendetes Systemfeldmenü

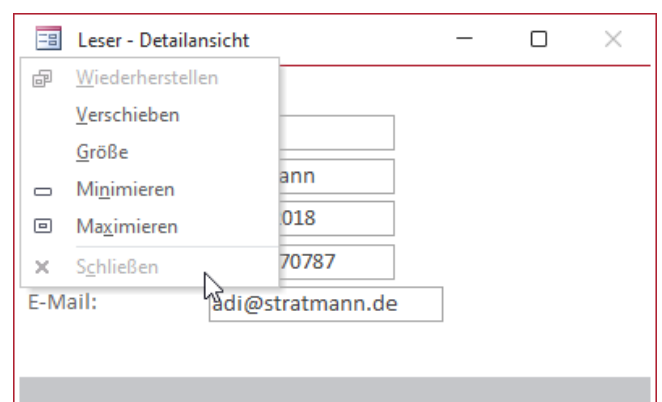


Bild 8: Deaktivierte Schließen-Schaltfläche

## Formulare [basics]: Schaltflächen programmieren

Wenn wir von der Programmiersprache VBA in Zusammenhang mit Access-Formularen reden, geht es sehr oft um den Code, der durch das Anklicken einer Schaltfläche ausgelöst wird. Damit können wir einem Formular die Funktionen hinzufügen, die für das Verarbeiten der enthaltenen Daten benötigt werden. In manchen Fällen handelt es sich um das einfache Schließen des Formulars, in anderen soll ein Klick auf eine Schaltfläche vielleicht ein anderes Formular öffnen, das die Details zu einem aus einer Liste ausgewählten Datensatz liefert. Dieser Artikel zeigt die Grundlagen zur Programmierung von Schaltflächen per VBA.

### Schaltfläche hinzufügen

Um eine Schaltfläche zu einem Formular hinzuzufügen, zeigen wir dieses in der Entwurfsansicht an. Im Bereich **Formularentwurf** des Ribbons finden wir dann alle in Access eingebauten Steuerelemente, unter anderem auch die Schaltfläche. Klicken wir diese an, haben wir zwei Möglichkeiten:

- Wir können im Formular entweder an die Stelle klicken, an welcher die linke, obere Ecke der Schaltfläche landen soll. Dann wird die Schaltfläche in der Standardgröße angelegt.
- Oder wir ziehen direkt einen Rahmen in der gewünschten Größe auf, wenn die Schaltfläche eine andere als die Standardgröße erhalten soll.

In beiden Fällen landet eine Schaltfläche im Formular, die eine Beschriftung wie **Befehl0** und einen gleichlautenden Namen erhält (siehe Bild 1).

### Name und Beschriftung anpassen

Bevor wir die Schaltfläche mit einer Funktion füllen, wollen wir diese beiden Eigenschaften ändern. Den angezeigten Text, damit der Benutzer weiß, was ge-

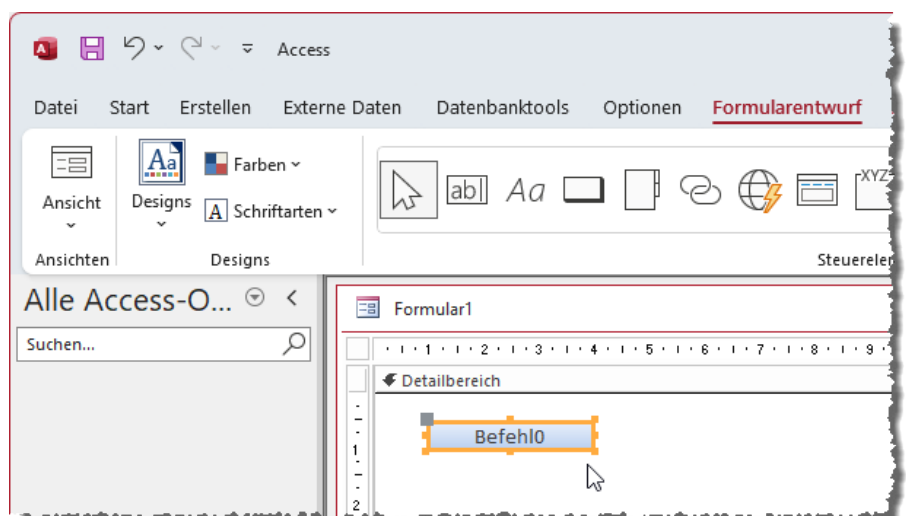


Bild 1: Hinzufügen einer Schaltfläche zur Entwurfsansicht

schieht, wenn er auf die Schaltfläche klickt, und den Namen, damit wir die Schaltfläche später einfach per VBA-Code referenzieren können. Und da die durch die Schaltfläche ausgelöste Prozedur einen Namen erhält, der zum Teil aus dem Namen der Schaltfläche besteht, sollte dieser recht aussagekräftig sein.

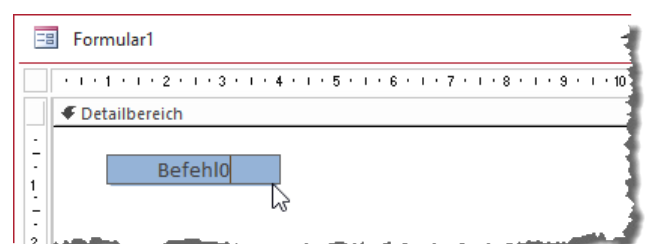


Bild 2: Ändern der Beschriftung

Die Beschriftung ändern wir, indem wir die Schaltfläche einmal anklicken und dann noch einmal (nicht zu schnell hintereinander) und die dann im Änderungsmodus angezeigte Beschriftung anpassen (siehe Bild 2).

Wir können die Beschriftung auch direkt im Eigenschaftenblatt der Schaltfläche ändern. Dazu muss die Schaltfläche markiert sein. Die Beschriftung ändern wir wie im Beispiel aus Bild 3 in OK.

Die Anpassung des Namens erfolgt über die Eigenschaft **Name** auf der Registerseite **Andere** des Eigenschaftenblatts. Hier geben wir den Namen `cmdOK` ein. Mit dem Präfix `cmd` geben wir an, dass es sich bei dem Steuerelement um einen **CommandButton** handelt, also eine Schaltfläche. Damit können wir dieses später, wenn wir einmal per VBA auf solche Steuerelemente zugreifen wollen, leichter per IntelliSense auffinden.

Nachdem dies erledigt ist, finden wir auch in der Auswahlliste oben im Eigenschaftenblatt bereits den neuen Namen der Schaltfläche vor. Sollte dieses aus irgendwelchen Gründen einmal nicht direkt im Formular auszuwählen sein, beispielsweise, weil es durch andere Steuerelemente verdeckt ist, können wir es über die Auswahl in dieser Liste selektieren (siehe Bild 4).

### Ereignisprozedur anlegen

Damit kommen wir auch schon zum magischen Teil dieses Artikels: Dem Anlegen einer Ereignisprozedur, die dann die für diese Schaltfläche vorgesehenen Codezeilen ausführt.

Das erledigen wir über die Registerseite **Ereignis** des Eigenschaftenblatts, während die Schaltfläche `cmdOK` markiert ist. Hier finden wir verschiedene Ereignisse vor, von denen uns aktuell nur das Ereignis

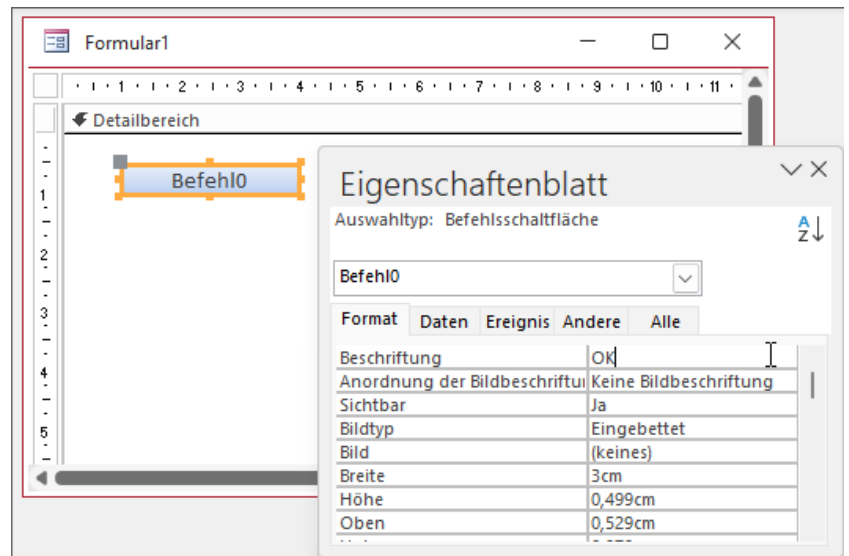


Bild 3: Ändern der Beschriftung per Eigenschaftenblatt

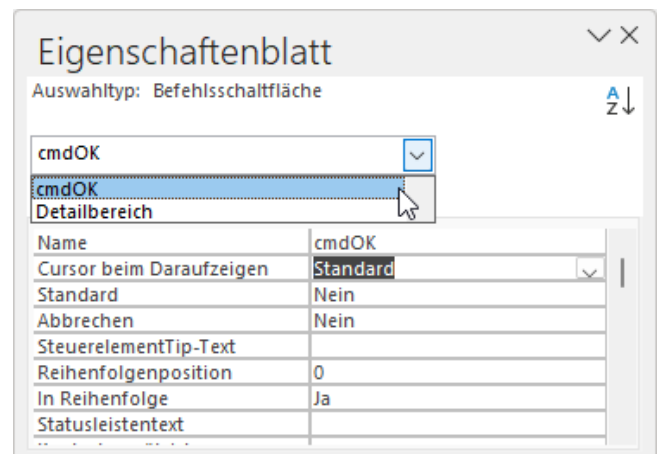


Bild 4: Ändern des Namens der Schaltfläche

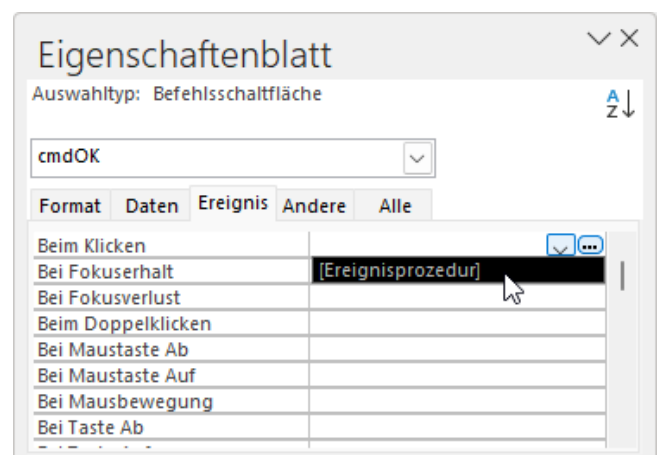


Bild 5: Auswählen des Eintrags [Ereignisprozedur]

Beim Klicken interessiert. Mit der Schaltfläche mit dem nach unten zeigenden Pfeil öffnen wir eine Liste, die aktuell nur den Eintrag [Ereignisprozedur] enthält (siehe Bild 5).

### Anzeigen, dass es eine Ereignisprozedur für ein Ereignis gibt

Genau diesen benötigen wir und wählen ihn aus. Damit legen wir jedoch noch nicht die Prozedur an, die beim Anklicken der Schaltfläche ausgelöst werden soll. Vielmehr geben wir nur an, dass Access beim Anklicken der Schaltfläche im Klassenmodul des Formulars nachschauen soll, ob dort ein passendes Ereignis notiert ist. Um dieses anzulegen, klicken wir nach dem Auswählen des Eintrags [Ereignisprozedur] auf die rechts davon befindliche Schaltfläche mit den drei Punkten.

Dies öffnet den VBA-Editor, der eine Ansicht wie in Bild 6 bietet. Wichtig für das Verständnis ist: Access hat nun ein Klassenmodul namens **Form\_frmSchaltflaechen** angelegt. Ein solches Klassenmodul wird für jedes Formular und jeden Bericht mit der ersten Ereignisprozedur angelegt und über die Eigenschaft **Enthält Modul** können wir es auch ohne Ereignisprozedur hinzufügen. Das Klassenmodul hat bei Formularen immer einen Namen, der aus **Form\_** und dem Formularnamen zusammengesetzt ist. Dieser wird links im Projektextplorer angezeigt und in der Titelleiste, sofern dieses Klassenmodul gerade aktiviert ist.

In diesem Klassenmodul hat Access bereits die Ereignisprozedur angelegt. Ihre Benennung folgt genau wie die des Klassenmoduls einem bestimmten Schema. Bei Ereignisprozeduren lautet dieses: **Steuer-elementname plus Unterstrich plus Ereignisname**.

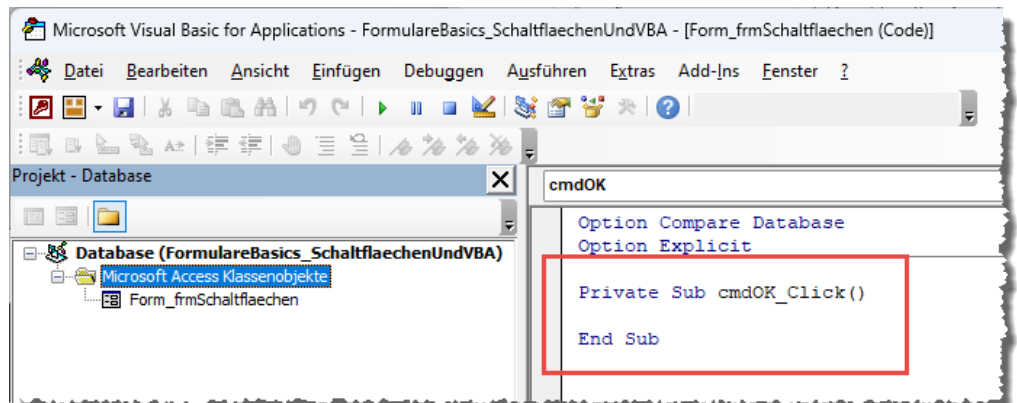


Bild 6: Die neu angelegte Ereignisprozedur

In diesem Fall für die Schaltfläche **cmdOK** und das Beim Klicken-Ereignis (Click) also **cmdOK\_Click**.

### Prozedur mit Funktion füllen

Die einzige Aufgabe, die im ersten Schritt noch bleibt, ist das Füllen der Ereignisprozedur mit den auszuführenden Anweisungen. In diesem Fall wollen wir, dass die Schaltfläche das Formular schließt.

Das erreichen wir mit der **Close**-Methode der **DoCmd**-Klasse. Dieser übergeben wir den Typ des zu schließenden Elements sowie den Namen:

```
Private Sub cmdOK_Click()  
    DoCmd.Close acForm, "frmSchaltflaechen"  
End Sub
```

### Ereignisprozedur ausprobieren

Wenn wir die Prozedur testen wollen, wechseln wir zuerst in die Formularansicht des Formulars. Ein Klick auf die Schaltfläche schließt das Formular nun.

### Name zur Laufzeit ermitteln

Wir optimieren den Code gleich. Aktuell hätten wir nämlich zusätzlichen Aufwand, wenn wir den Namen des Formulars einmal ändern. Angenommen, wir ändern diesen von **frmSchaltflaechen** auf **frmSchaltflaeche**. Wenn wir die Schaltfläche betätigen, versucht die Prozedur immer noch, ein Formular namens **frmSchaltflaechen** zu schließen. Dies führt zwar nicht



## Formulare [basics]: OK- und Abbrechen-Buttons

In Detailformularen zur Anzeige von einzelnen Datensätzen gibt es keine für den Benutzer ersichtliche Möglichkeit, was beim Schließen des Formulars mit einem neu angelegten oder einem geänderten Datensatz geschieht. Werden die Änderungen gespeichert oder nicht? Und wenn ich Änderungen durchgeführt habe und diese verwerfen möchte, wie gehe ich dann vor? In diesem Artikel gehen wir auf dieses Thema ein und zeigen sowohl das Standardverhalten von Access-Formularen bezüglich der verschiedenen Aktionen auf als auch die von uns bevorzugte Möglichkeit, dem Benutzer das Speichern oder das Verwerfen der durchgeführten Änderungen zu realisieren.

### Beispieldatenbank

Die Beispiele dieses Artikels findest Du in der Datenbank **FormulareBasics\_OKUndAbbrechen.accdb**.

### Ausgangssituation

Wenn ein Access-Einsteiger ein Detailformular baut, das keine Schaltflächen oder VBA enthält, und diese Anwendung einem Benutzer bereitstellt, sieht dieser zwar die Daten in den entsprechenden Feldern im Formular. Er weiß jedoch nicht, was genau geschieht, wenn er beispielsweise die Daten eines vorhandenen Datensatzes ändert oder Daten zu einem neuen, leeren Datensatz hinzufügt. Was er sieht, wenn der Benutzer den Datensatzmarkierer aktiviert hat, ob sich der Datensatz gerade in Bearbeitung befindet. Das kann er an dem Symbol oben im Datensatzmarkierer erkennen (siehe Bild 1).

Auch beim Bearbeiten eines neu angelegten Datensatzes taucht dieses Icon auf. Dadurch weiß der Benutzer aber noch nicht, was geschieht, wenn er nun das Formular beispielsweise durch einen Mausklick auf die **Schließen**-Schaltfläche rechts oben in der Titelleiste schließt. Wird der Datensatz gespeichert? Bleibt er im Bearbeiten-Zustand? Werden die Änderungen verworfen?

### Datensatz speichern

Erst durch Ausprobieren findet er heraus, dass die Änderungen beim Schließen des Formulars gespeichert werden – außer, er hat die Daten so geändert,

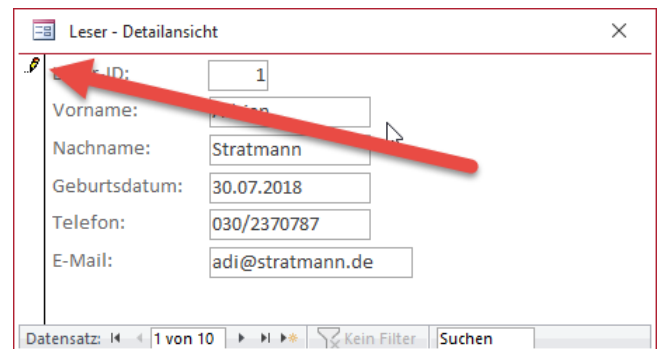


Bild 1: Anzeige des aktuellen Zustands des Datensatzes

dass diese gegen eine Restriktion verstoßen und der Datensatz nicht gespeichert werden kann. Dann würde jedoch eine entsprechende Meldung erscheinen. Das Gleiche geschieht, wenn der Benutzer einen Datensatz bearbeitet und dann über die Navigationsschaltflächen zu einem anderen Datensatz wechselt oder einen neuen Datensatz anlegt.

Und es gibt noch weitere Methoden, um den aktuellen Datensatz zu speichern, wenn seit dem letzten Speichern Änderungen vorgenommen wurden:

- Mit der Tastenkombination **Strg + S**
- Durch Anklicken des Datensatzmarkierers
- Programmieren einer eigenen Funktion, die beispielsweise durch eine Schaltfläche ausgelöst werden kann

Auch das sind jedoch keine für den unerfahrenen Benutzer offensichtlichen Optionen zum Speichern.

### Datensatz verwerfen

Das Stift-Symbol im Datensatzmarkierer zeigt an, dass es seit dem letzten Speichern des Datensatzes Änderungen gegeben hat. In diesem Fall können wir nicht nur die durchgeführten Änderungen speichern, sondern diese auch verwerfen und den Datensatz in den Zustand versetzen, den er unmittelbar nach dem letzten Speichern hatte. Dazu gibt es die folgenden Möglichkeiten:

- Betätigen der Esc-Taste
- Programmieren einer eigenen Funktion, die beispielsweise durch eine Schaltfläche ausgelöst werden kann

Wir sehen: Die Menge der eingebauten Möglichkeiten ist begrenzt. Außerdem ist die eine Möglichkeit für den Benutzer nicht offen ersichtlich.

### Umsetzen der verschiedenen Funktionen mit Schaltflächen und VBA

Die Lösung, die wir für diesen Zweck vorschlagen, basiert auf dem Einsatz von Schaltflächen, bei denen der Benutzer anhand der Beschriftung erkennen kann, ob der Datensatz gespeichert oder ob die Änderungen verworfen werden sollen.

Wir können Schaltflächen mit Makros ausstatten, aber diese sind relativ unflexibel. Mit der Programmiersprache VBA stehen uns umfangreiche Möglichkeiten zur Programmierung der gewünschten Funktionen zur Verfügung. Wir wollen verschiedene Schaltflächen vorstellen, von denen Du Dir die gewünschten aussuchen kannst. Wir verwenden standardmäßig zwei Schaltflächen:

- **OK:** Speichert die Änderungen seit dem letzten Speichern und schließt das Formular.
- **Abbrechen:** Verwirft die Änderungen seit dem letzten Speichern und schließt das Formular.

Man könnte hier auch noch eine Schaltfläche zum Zwischenspeichern der bisherigen Änderungen hinzufügen.

Es gibt jedoch auch andere Varianten. Denkbar wären auch die beiden folgenden Schaltflächen:

- **Datensatz speichern:** Speichert nur die Änderungen, ohne das Formular zu schließen
- **Änderungen verwerfen:** Verwirft die Änderungen und schließt das Formular ebenfalls nicht, sodass der Benutzer wieder die zuletzt gespeicherte Fassung vor sich sieht.
- Hier könnte man noch eine Schaltfläche zum Schließen des Formulars hinzufügen, wenn man beispielsweise die **Schließen**-Schaltfläche oben rechts in der Titelleiste deaktiviert oder ausgeblendet hat.

Wir schauen uns in den folgenden Abschnitten die verschiedenen Varianten und den jeweils benötigten VBA-Code an. Dabei verweisen wir schon einmal auf den Artikel **Formulare [basics]: Schaltflächen programmieren** ([www.access-basics.de/645](http://www.access-basics.de/645)), in dem wir genauer auf die Eigenschaften von Schaltflächen und das Anlegen von Ereignisprozeduren für das **Beim Klicken**-Ereignis eingehen.

### Speichern eines Datensatzes per VBA

Die erste Frage, die sich stellt, ist: Wie speichern wir eigentlich per VBA den Datensatz, der gerade in einem Formular angezeigt wird? Dazu gibt es verschiedene Möglichkeiten, die alle voraussetzen, dass der Benutzer alle benötigten Daten eingegeben hat und diese keine Restriktionen verletzen:

- Wir schließen das Formular einfach. Dadurch wird der aktuelle Datensatz automatisch gespeichert.
- Wir wechseln den Datensatz. Dadurch werden offene Änderungen am aktuellen Datensatz gespeichert.