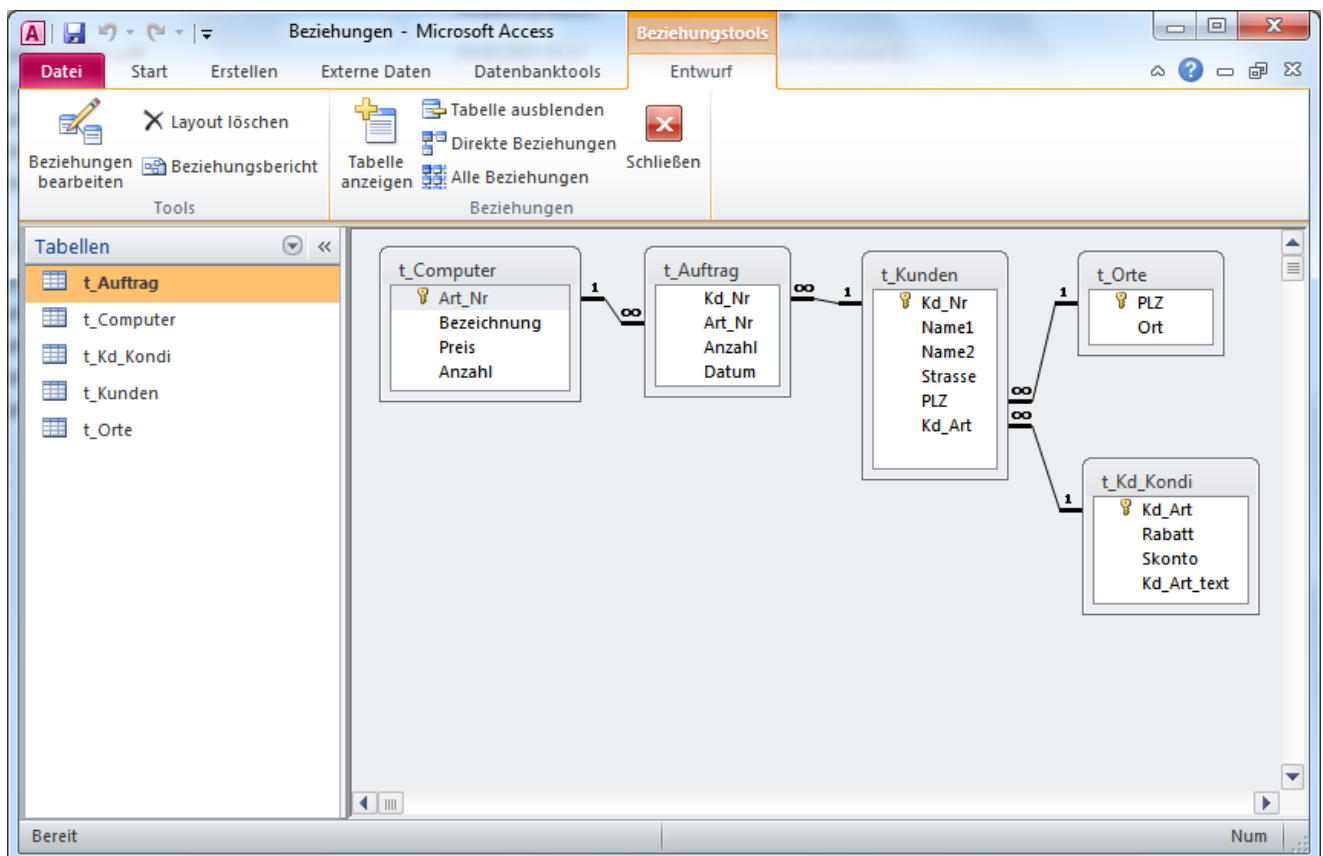


Datenbank-Anwendungen mit Access



Inhaltsverzeichnis

1. Einführung in Datenbanken	1
1.1 Einführungsübung	1
1.2 Definition und Aufgaben einer Datenbank	3
1.3 manuelle vs. EDV-gestützte Datenbanken	3
1.4 Relationale vs. Nicht-relationale Datenbanken	4
1.5 Datenbankbegriffe	7
2. Erstellung einer Datenbank	11
2.1 Vorabinformation	11
2.2 Anlegen der Datenbank	11
2.3 Erstellung der Tabelle t_Kundenkonditionen	12
2.3.1 Anlage des Datenfelds „Kondition“	14
2.3.2 Anlage des Datenfelds „Rabatt“	16
2.3.3 Anlage des Datenfelds „Skonto“	17
2.3.4 Anlage des Datenfelds „Konditions-Text“	17
2.3.5 Speicherung des Tabellenentwurfs & Datenerfassung	18
2.4 Datenfeldeigenschaften	20
2.4.1 Feldname	20
2.4.2 Felddatentyp bzw. Datenfeldtyp	20
2.4.3 Beschreibung	22
2.4.4 Feldgröße	22
2.4.5 Format	22
2.4.6 Dezimalstellen	23
2.4.7 Eingabeformat	23
2.4.8 Beschriftung	24
2.4.9 Standardwert	24
2.4.10 Gültigkeitsregel	24
2.4.11 Gültigkeitsmeldung	25
2.4.12 Eingabe erforderlich	25
2.4.13 Leere Zeichenfolge	25
2.4.14 Indiziert	26
2.5 Erstellung der Tabelle t_Orte	26
2.6 Import der übrigen Tabellen	32

2.6.1 Import der Tabelle t_Artikel	32
2.6.2 Import der Tabelle t_Aufträge.....	33
2.6.3 Import der Tabelle t_Kunden	36
2.7 Beziehungen einrichten - Verbindung der Tabellen	40
2.7.1 Angleichung der Fremdschlüssel an die Primärschlüssel	40
2.7.2 Beziehungen einrichten.....	42
2.7.3 Nachschlagefunktionen einrichten.....	45
3. Das Entity-Relationship Modell (ERM).....	49
3.1 Bausteine und Regeln des ERM	49
3.2 Beziehungstypen (Kardinalität).....	50
3.2.1 Die 1 : n - Beziehung	50
3.2.2 Die n : m - Beziehung	51
3.2.3 Die 1 : 1 - Beziehung	52
3.3 Aufgaben	53
4. Datenbanken-Abfragen (Queries).....	55
4.1 Sinn und Zweck von Datenbankabfragen	55
4.2 Einfache Abfragen erstellen	56
4.3 Abfragen mit Kriterien	62
4.3.1 Vergleichsoperatoren für Kriterien	63
4.3.2 Mustervergleich mit dem Wie-Operator	65
4.3.3 Kriterien mit Parameterübergabe	66
4.4 Abfragen mit Verknüpfungsoperatoren	68
4.4.1 UND - Verknüpfung	68
4.4.2 ODER - Verknüpfung.....	69
4.4.3 Der ZWISCHEN - Operator	72
4.4.4 Der IN - Operator.....	73
4.5 Abfragen über mehrere Tabellen.....	74
4.6 Abfragen mit Berechnungen.....	76
4.6.1 Mathematische Berechnungen	76
4.6.2 Verkettung von Texten	81
4.7 Abfragen mit Funktionen	83
4.7.1 Einrichtung einer Funktion per Drop-Down Menü.....	83
4.7.2 Manuelle Formulierung der Syntax einer Funktion.....	85
4.7.3 Verwendung von Funktionen mit weiteren Datenfeldern	86

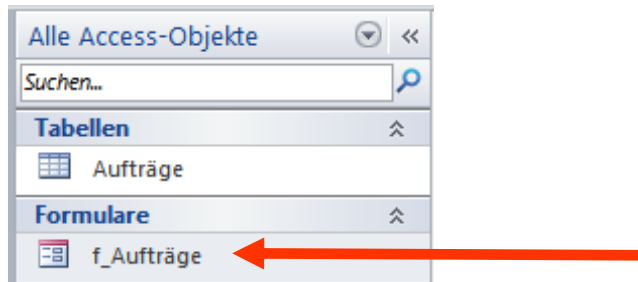
4.8 Abfrageergebnisse gruppieren	89
4.9 Abschlussübung	92

1. Einführung in Datenbanken

1.1 Einführungsübung

Situation: In einem Computershop müssen einige Aufträge in das Computersystem eingegeben werden. Hierfür steht eine vorstrukturierte Datenbank zur Verfügung:

Rufen Sie die Übung [Ü01_Auftragserfassung.accdB](#) auf und öffnen Sie in dieser das Formular **f_Aufträge**:

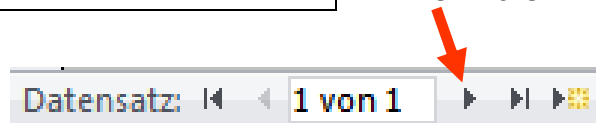


Tragen Sie in diesem Formular der Reihe nach die folgenden Bestelldaten diverser Kunden ein:

Auftrag 1:

Auftrags-Nr	
Kunden-Code	29
Name der Firma	Sportstudio Zabel
Straße	Kastanienweg 29
Postleitzahl	55743
Stadt	Idar-Oberstein
Artikel-Nr	155
Artikelname	Notebook 2,0 GHz TFT
Bestellte Einheiten	1
Preis pro Stück	1.599,00

nach der Eingabe bitte
hier klicken



Auftrag 2:

Auftrags-Nr	
Kunden-Code	29
Name der Firma	Sportstudio Zabel
Straße	Kastanienweg 29
Postleitzahl	55743
Stadt	Idar-Oberstein
Artikel-Nr	6001
Artikelname	Drucker HP 630 C
Bestellte Einheiten	1
Preis pro Stück	99,00

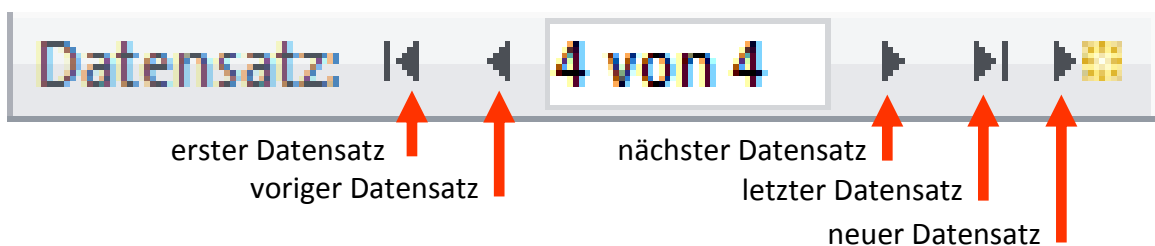
Auftrag 3:

Auftrags-Nr	
Kunden-Code	98
Name der Firma	Steuerberater Ternieden
Straße	Marienburger Str. 47
Postleitzahl	55606
Stadt	Kirn
Artikel-Nr	155
Artikelname	Notebook 2,0 GHz TFT
Bestellte Einheiten	1
Preis pro Stück	1.599,00

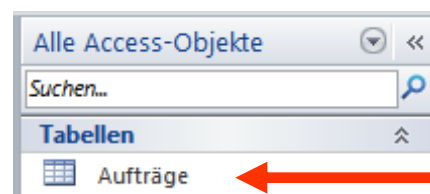
Auftrag 4:

Auftrags-Nr	
Kunden-Code	98
Name der Firma	Steuerberater Ternieden
Straße	Marienburger Str. 47
Postleitzahl	55606
Stadt	Kirn
Artikel-Nr	6001
Artikelname	Drucker HP 630 C
Bestellte Einheiten	1
Preis pro Stück	99,00

Kontrollieren Sie die von Ihnen erfassten Aufträge mittels der folgenden Schaltflächen:



Durch Aufruf der Tabelle **Aufträge** können Sie sich alle erfassten Aufträge gleichzeitig in Tabellenform ansehen. In den Tabellen werden die erfassten Daten letztendlich auch gespeichert.



Aufgabe:

Überlegen Sie kritisch, ob das hier gewählte Verfahren zur Eingabe der Daten in das Computersystem sinnvoll ist! Welche Kritikpunkte fallen Ihnen dazu ein?

Berechtigte Kritikpunkte an dem zuvor gewählten Verfahren zur Eingabe der Daten sind:

- die Daten der Kunden wie Kunden-Nr, Name, Anschrift müssen mehrfach eingegeben werden
- die Daten, die den Artikel beschreiben wie Artikel-Nr, Bezeichnung, Preis müssen mehrfach eingegeben werden

Hierdurch entsteht einerseits ein **erhöhter Arbeitsaufwand** und zum anderen eine **höhere Gefahr für Eingabefehler** bzw. **widersprüchliche Daten**.

1.2 Definition und Aufgaben einer Datenbank

Eine **Datenbank** ist eine **Ansammlung von Informationen, die sich auf ein bestimmtes Thema oder einen bestimmten Zweck beziehen und verwaltet werden müssen**. (z. B.: Bücher in einer Bibliothek, Waren im Warenlager, Gehaltsabrechnungen im Betrieb, etc.)

Unter einem **Datenbanksystem** verstehen wir eine *nach einer bestimmten Struktur aufgebaute Datensammlung **plus** ein entsprechendes Verwaltungssystem, welches die Zugriffe auf die Datenbank und deren Bestand organisiert*.

Mit der Verwendung von Datenbanken verfolgen wir grundsätzlich folgende **Ziele**:

- **Wirtschaftlichkeit** Vereinfachung und Beschleunigung der Arbeit
- **Redundanzvermeidung** Daten über ein Objekt (Kunde, Artikel, etc.) sollen nur einmal gespeichert werden (Redundanz = doppelt oder mehrfach vorhandene Informationen)
- **Datenintegrität** Daten über ein Objekt müssen fehlerfrei und vollständig sein (Integrität = Unversehrtheit)
- **Datenkonsistenz** Daten müssen eindeutig sein (Konsistenz = Übereinstimmung, Schlüssigkeit)
- **Selektivität** Der jeweilige Nutzer braucht nur die Daten zu sehen, die er für einen bestimmten Zweck benötigt (Selektivität = Auswählbarkeit, benutzerorientierte Sicht der Daten)

1.3 manuelle vs. EDV-gestützte Datenbanken

Ein klassisches Beispiel für **manuelle Datenbanken** sind die guten alten **Karteikastensysteme**, bei welchen das Auffinden der Daten durch Register erleichtert wird. Ein Karteikasten beinhaltet dabei stets zusammengehörende und nach bestimmten Kriterien geordnete Daten. Jede Karte enthält i.d.R. die gleiche, festgelegte Anzahl von Eintragungen (z. Bsp. Kundenarteikasten, alphabetisch nach Nachname geordnet, Register erleichtern das alphabetische Auffinden bestimmter Kunden)

Eine EDV-gestützte Datenbank hingegen besteht aus einer **Datensammlung** **plus** einem entsprechenden **Verwaltungsprogramm**.

Die Vorteile von EDV-gestützten Datenbanken gegenüber dem guten alten Karteikastensystem liegen dabei auf der Hand:

- bequemere Erfassung der Daten
 - schnelleres Finden von Daten
 - einfachere, z. T. automatische Aktualisierung der Daten
 - flexible, datenbankübergreifende Anordnung und Zusammenstellung von Daten
- => Aufdecken bisher unbekannter Informationszusammenhänge
=> effektivere Auswertung der Daten.

Die grundlegende Struktur der Datenorganisation bei einem Karteikastensystem und einer EDV-gestützten Datenbank ist jedoch miteinander vergleichbar:

Karteikastensystem



Datenbanksystem

Karteikasten



Tabelle

Karteikarte



Datensatz/Tabellenzeile

Eintragungsfeld



Datenfeld/Tabellenzelle

The screenshot shows an Access form titled 'auftrag'. It contains several text boxes for data entry:

- Auftrags-Nr: 1
- Kunden-Code: 29
- Name der Firma: Sportstudio Zabel
- Straße: Kastanienweg 29
- Postleitzahl: 55743
- Stadt: Idar-Oberstein
- Artikel-Nr: 155
- Artikelname: Notebook 2,0 GHz TFT
- Bestellte Einheiten: 1
- Preis pro Stück: 1.599,00 €

At the bottom, there is a status bar showing 'Datensatz: 1 von 4' and 'Kein Filter'.

The screenshot shows an Access table titled 'Aufträge'. It contains the following data:

Auftrags-Nr	Kunden-Code	Name der Firma	Straße
1	29	Sportstudio Zabel	Kastanienweg
2	29	Sportstudio Zabel	Kastanienweg
3	98	Steuerberater Ternieden	Marienburger
4	98	Steuerberater Ternieden	Marienburger

1.4 Relationale vs. Nicht-relationale Datenbanken

Die Datenbank zur Auftragserfassung aus unserem bisherigen Beispiel ist eine so genannte nicht-relationale Datenbank. Eine **nicht-relationale Datenbank besteht** in der Regel **aus nur einer einzigen Tabelle**, in welcher sämtliche Informationen undifferenziert voneinander gesammelt

werden. Den damit verbundenen **Nachteil** haben Sie bereits erkannt: Die Datenbank beinhaltet **redundante** (sich wiederholende) **Informationen**, die **mehrfach** in die Tabelle **eingetragen** werden und bei Änderungen bestimmter Daten (Kundenadresse, Artikelpreis, etc.) **mehrfach überarbeitet** werden müssen:

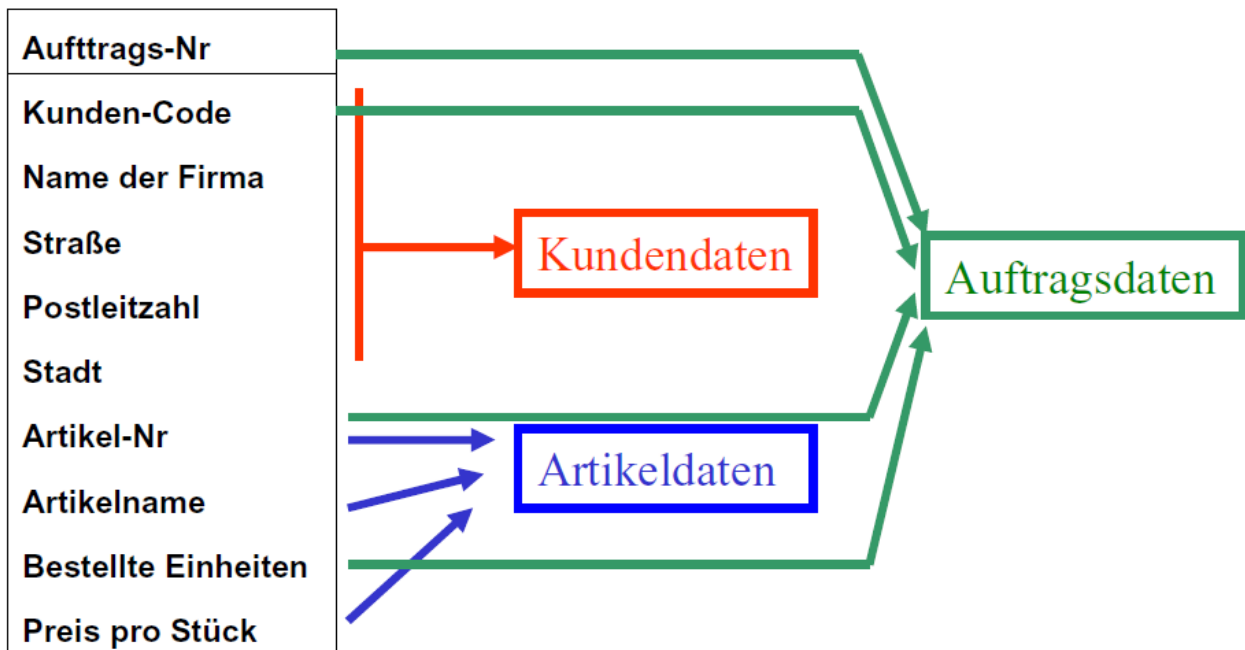
Auftrags-Nr	Kunden-Code	Name der Firma	Straße	Postleitzahl	Stadt	Artikel-Nr	Artikelname	Bestellte Einheiten	Preis pro Stück
1	29	Sportstudio Zabel	Kastanienweg 29	55743	Idar-Oberstein	155	Notebook 2,0 GHz TFT	1	1599,00
2	29	Sportstudio Zabel	Kastanienweg 29	55743	Idar-Oberstein	6001	Drucker HP 630 C	1	99,00
3	98	Steuerberater Ternieden	Marienburger Str. 47	55606	Kirn	155	Notebook 2,0 GHz TFT	1	1599,00
4	98	Steuerberater Ternieden	Marienburger Str. 47	55606	Kirn	6001	Drucker HP 630 C	1	99,00

Um diese Probleme zu vermeiden, ist es sinnvoll, die Daten in **mehrere Tabellen** aufzuteilen, die jeweils nur die Daten zu spezifischen „Objekten“ (z. Bsp. Kunden, Artikel, Bestellungen) enthalten und **miteinander in Beziehung** stehen. Eine solche Datenbank bezeichnet man sodann als **relationale Datenbank** (Relation = Beziehung).

Relationale Datenbanken beinhalten somit **mehrere Tabellen + Informationen, wie die Tabellen miteinander zusammenhängen**.

Der **Vorteil** liegt auf der Hand: Es gibt **keine redundanten Informationen** mehr, so dass **Eintragungen oder Überarbeitungen** einzelner Daten **jeweils nur einmal** vorgenommen werden müssen.

Schauen wir uns hierzu die **Datenstruktur** unserer bisher verwendeten Tabelle einmal etwas näher an: Die Tabelle beinhaltet einerseits **Kundendaten**, andererseits aber auch **Artikeldaten** und **Auftragsdaten**:



Es ist somit sinnvoll, die **Datenstruktur** dahingehend zu **ändern**, dass wir die **Daten in drei** entsprechende **Tabellen aufteilen**. Dieser Vorgang wird auch als **Normalisierung** bezeichnet:

Kundendaten	Auftragsdaten	Artikeldaten
Kunden-Code Name der Firma Straße Postleitzahl Stadt	Auftrags-Nr Kunden-Code Artikel-Nr Bestellte Einheiten	 Artikel-Nr Artikelname Preis pro Stück

Zur eindeutigen Identifikation eines jeden Eintrags in einer jeweiligen Tabelle legen wir für jede Tabelle weiterhin entsprechende **Schlüsselfelder** fest (es kann z. Bsp. zwei Kunden namens „Hans Müller“ in unserer Kundentabelle geben, doch ein jeder davon hat einen andren Kunden-Code, über welchen die beiden Kunden eindeutig voneinander unterschieden und identifiziert werden können):

t_Kunde	t_Auftrag	t_Artikel
Kunden-Code Name der Firma Straße Postleitzahl Stadt	Auftrags-Nr Kunden-Code Artikel-Nr Bestellte Einheiten	Artikel-Nr Artikelname Preis pro Stück

Schließlich müssen wir die drei Tabellen noch miteinander **in Beziehung bringen** bzw. **verbinden**, um zusammenhängende Informationen bezüglich einer Bestellung auch miteinander in Zusammenhang bringen zu können. So verweist der Kunden-Code in der Tabelle t_Auftrag auf den jeweiligen Kunden, der diesen Auftrag erteilt hat ... und die Artikel-Nr. auf den Artikel, der mit dem Auftrag bestellt wurde. Über diese **Querverweise** ist der Zugriff auf die übrigen Informationen zum jeweiligen Kunden sowie zum bestellten Artikel gewährleistet:

t_Kunde	t_Auftrag	t_Artikel
Kunden-Code Name der Firma Straße Postleitzahl Stadt	Auftrags-Nr Kunden-Code Artikel-Nr Bestellte Einheiten	Artikel-Nr Artikelname Preis pro Stück

Unsere neue Datenbankstruktur sieht in der **Datenblattansicht** (mit Daten befüllte Tabellen) nun wie folgt aus. Wie auffällt, kommt die Datenbank auf diese Weise mit deutlich weniger Einträgen aus, da die einzelnen Kundendaten und Artikeldaten jeweils nur einmal eingetragen werden

mussten und Redundanzen somit vermieden wurden und zusammenhängende Daten über die entsprechenden Schlüsselfelder miteinander verbunden und zuordenbar sind:

t_Kunde

Kunden-Code	Name der Firma	Straße	Postleitzahl	Stadt
29	Sportstudio Zabel	Kastanienweg 29	55743	Idar-Oberstein
98	Steuerberater Ternieden	Marienburger Str. 47	55606	Kirn

t_Auftrag

Auftrags-Nr	Kunden-Code	Artikel-Nr	Bestellte Einheiten
1	29	155	1
2	29	6001	1
3	98	155	1
4	98	6001	1

t_Artikel

Artikel-Nr	Artikelname	Preis pro Stück
155	Notebook 2,0 GHz TFT	1599,00
6001	Drucker HP 630 C	99,00

Aufgabe:

Öffnen Sie die Übung [Ü02 Verbindungen.xlsx](#) und zeichnen Sie in dieser die Verbindungen zwischen den Tabellen ein.

1.5 Datenbankbegriffe

Relationale Datenbank

Dies ist eine Menge von Tabellen, die so aufeinander abgestimmt sind, dass Verknüpfungen zwischen ihnen bestehen.

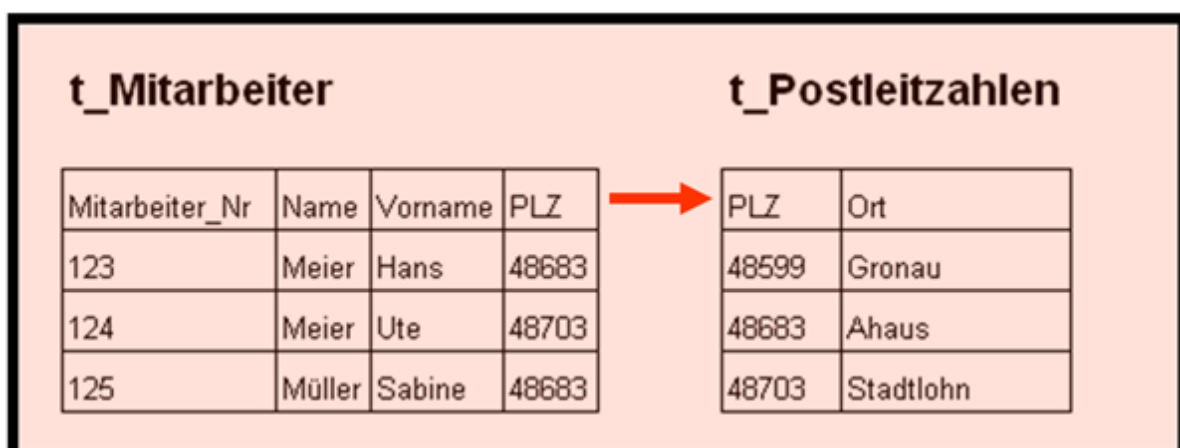


Tabelle (auch: „Datei“)

Hier sind alle Datensätze mit einheitlichen Merkmalen für ein Objekt gespeichert (z. Bsp. Mitarbeiter).

t_Mitarbeiter				t_Postleitzahlen	
Mitarbeiter_Nr	Name	Vorname	PLZ	PLZ	Ort
123	Meier	Hans	48683	48599	Gronau
124	Meier	Ute	48703	48683	Ahaus
125	Müller	Sabine	48683	48703	Stadtlohn

Datensatz (auch: „Tupel“ oder „Tabellenzeile“)

Hierunter verstehen wir alle Informationen / Attribute zu einem einzelnen Objekt (z. Bsp. für die Mitarbeiterin Ute Meier).

t_Mitarbeiter				t_Postleitzahlen	
Mitarbeiter_Nr	Name	Vorname	PLZ	PLZ	Ort
123	Meier	Hans	48683	48599	Gronau
124	Meier	Ute	48703	48683	Ahaus
125	Müller	Sabine	48683	48703	Stadtlohn

Datenfeld (auch: „Tabellenspalte“, „Attribut“ oder „Feldname“)

Ein Datenfeld / Attribut entspricht einem Merkmal eines Tupels (Datensatz) und beschreibt eine spezifische Eigenschaft eines Objekts (z. Bsp.: Name).

t_Mitarbeiter				t_Postleitzahlen	
Mitarbeiter_Nr	Name	Vorname	PLZ	PLZ	Ort
123	Meier	Hans	48683	48599	Gronau
124	Meier	Ute	48703	48683	Ahaus
125	Müller	Sabine	48683	48703	Stadtlohn

Für jedes einzelne **Datenfeld** müssen grundsätzlich die folgenden **Merkmale** definiert werden:

- **Datenfeld-Name** (sollte auf den Verwendungszweck des Feldes hinweisen, z. Bsp. Kundenr, Name, ect.)
- **Datenfeld-Typ** (richtet sich nach dem geplanten Verwendungszweck, z. Bsp. Textfeld oder Zahlenfeld. Textfelder enthalten „starre“ Informationen, Zahlenfelder hingegen können für Berechnungen herangezogen werden)
- **Datenfeld-Länge** (Anzahl der reservierten Zeichen für das Datenfeld, z. Bsp. 5 Zeichen für PLZ)

Datenfeldinhalt (auch: „Attributwert“, „Wert“ oder „Datum“)

Dies ist die jeweilige Information, welche das zugehörige Attribut eines Tupels beschreibt. Sie besteht aus einzelnen numerischen oder alphanumerischen Zeichen (z. Bsp. „Müller“).

t_Mitarbeiter

Mitarbeiter_Nr	Name	Vorname	PLZ
123	Meier	Hans	48683
124	Meier	Ute	48703
125	Müller	Sabine	48683

t_Postleitzahlen

PLZ	Ort
48599	Gronau
48683	Ahaus
48703	Stadtlohn

Primärschlüssel (auch: „Identifikationsschlüssel“ oder „ID-Schlüssel“)

Jeder Datensatz einer Tabelle muss mittels eines Schlüssels eindeutig identifizierbar sein. In der Tabelle Mitarbeiter ist die Mitarbeiter_Nr der Primärschlüssel, da jeder Mitarbeiter hierdurch **eindeutig identifiziert** werden kann. In der Spalte, die den Primärschlüssel einer Tabelle bildet, darf jeder Wert nur einmal vorkommen. Duplikate sind nicht erlaubt!

t_Mitarbeiter

Mitarbeiter_Nr	Name	Vorname	PLZ
123	Meier	Hans	48683
124	Meier	Ute	48703
125	Müller	Sabine	48683

t_Postleitzahlen

PLZ	Ort
48599	Gronau
48683	Ahaus
48703	Stadtlohn

Fremdschlüssel

Um eine **Verbindung zwischen zwei Tabellen herstellen** zu können, muss ein Datenfeld in einer der beiden Tabellen auf den Primärschlüssel in der anderen Tabelle querverweisen. Dasjenige Datenfeld, welches den **Querverweis** auf den Primärschlüssel in der anderen Tabelle darstellt, bezeichnet man als **Fremdschlüssel**, da es auf den **Primärschlüssel in einer anderen bzw. fremden Tabelle** verweist. Zum Beispiel ist das Datenfeld PLZ in t_Postleitzahlen der Primärschlüssel, da es hier zur eindeutigen Identifizierung der Orte dient. In der Tabelle t_Mitarbeiter hingegen ist das Datenfeld PLZ ein Fremdschlüssel, da es hier als Querverweis auf das entsprechende Primärschlüsselfeld der Tabelle t_Postleitzahlen dient und somit die Verbindung zur Tabelle t_Postleitzahlen herstellt. Im Gegensatz zu einem Primärschlüsselfeld kann in einem Fremdschlüsselfeld ein bestimmter Wert auch mehrmals vorkommen (z. Bsp. können mehrere Mitarbeiter aus dem gleichen Ort kommen, so dass die entsprechende PLZ in der Tabelle t_Mitarbeiter häufiger vorkommt – in der Tabelle t_Postleitzahlen hingegen nur einmal).

t_Mitarbeiter

Mitarbeiter_Nr	Name	Vorname	PLZ
123	Meier	Hans	48683
124	Meier	Ute	48703
125	Müller	Sabine	48683

Fremdschlüssel ↓

t_Postleitzahlen

PLZ	Ort
48599	Gronau
48683	Ahaus
48703	Stadtlohn

Primärschlüssel ↓



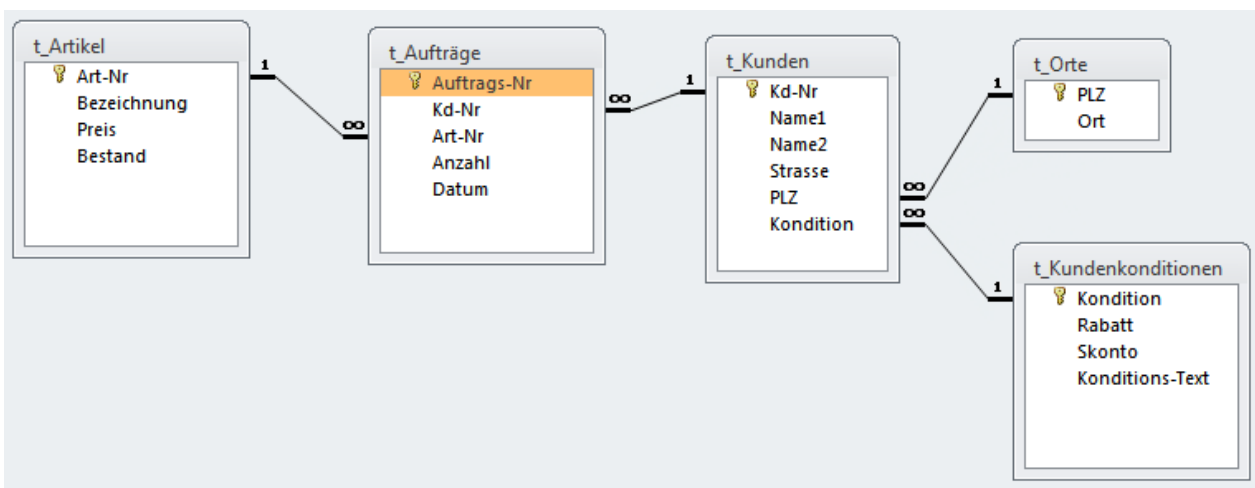
2. Erstellung einer Datenbank

2.1 Vorabinformation

Situation: Zur Erfassung der Aufträge in einem Computershop soll eine relationale Datenbank erstellt werden. Die Datenbank soll aus folgenden 5 Tabellen bestehen:

- t_Kunden dient zur Speicherung der Kundendaten
- t_Orte dient zur Speicherung der Postleitzahlen und Orte
- t_Kundenkonditionen dient zur Speicherung der Kundenkonditionen
- t_Artikel dient zur Speicherung der Artikeldaten
- t_Aufträge dient zur Speicherung der Auftragsdaten

Die Tabellen sollen dabei wie folgt miteinander in Beziehung stehen:



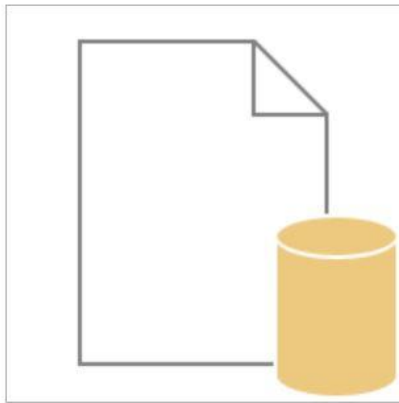
Die Tabellen t_Orte und t_Kundenkonditionen legen wir dabei manuell an. Die Tabellen t_Kunden, t_Artikel und t_Aufträge hingegen importieren wir aus bereits vorhandenen Dateien, in welchen die entsprechenden Daten schon gespeichert sind.

2.2 Anlegen der Datenbank

Starten Sie Access durch Klick auf das entsprechende Icon auf Ihrem Desktop und legen Sie durch Klick auf das Symbol „Leere Desktopdatenbank“ zunächst eine neue Datenbank an:



Geben Sie in dem daraufhin erscheinenden Dialogfenster ein, dass Ihre Datenbank unter dem Dateinamen „Computerladen.accdb“ gespeichert werden soll und wählen Sie über die recht daneben befindliche Schaltfläche einen Speicherort für Ihre Datenbank aus (z. Bsp. Ihren USB-Stick oder Ihr Home-Verzeichnis). Legen Sie die Datenbankdatei sodann über die Schaltfläche „Erstellen“ an:



Leere Desktopdatenbank

Sollte ich eine Access 2013-App oder eine Access-Desktopdatenbank erstellen?

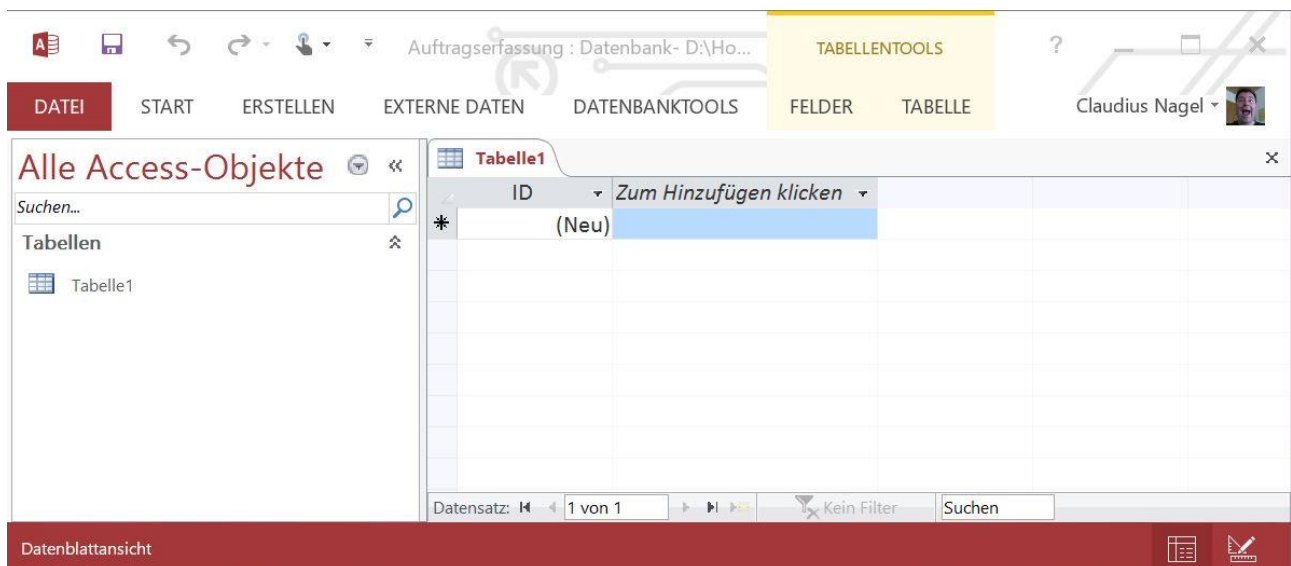
Dateiname

Computerladen.accdb

D:\Home\Datenbanken\

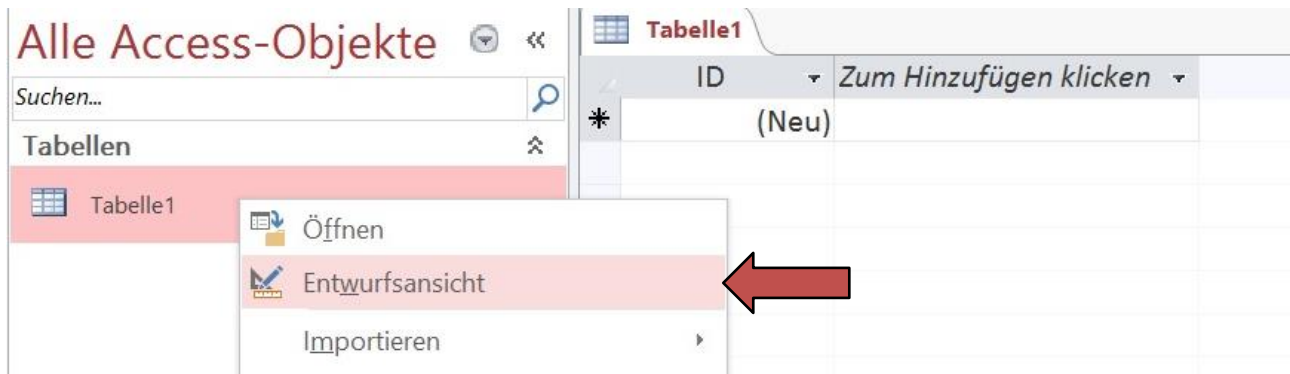


Access legt die Datenbankdatei sodann entsprechend an und öffnet diese zur Bearbeitung. Natürlich ist die Datenbank zunächst noch leer, wobei jedoch bereits eine leere Tabelle namens „Tabelle1“ vorangelegt ist:

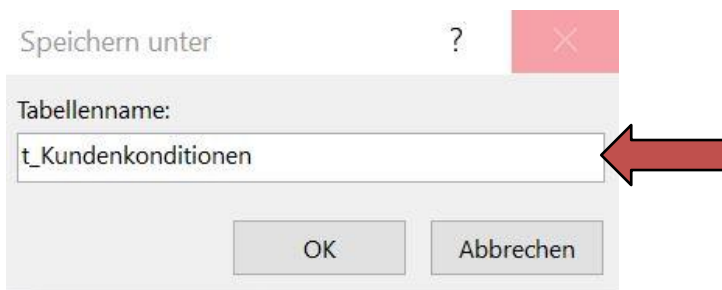


2.3 Erstellung der Tabelle t_Kundenkonditionen

Die bereits vorangelegte Tabelle wollen wir nun zur Erstellung der ersten Tabelle unserer Datenbank verwenden, welches die Tabelle „t_Kundenkonditionen“ sein soll. Klicken Sie die Tabelle hierzu mit der rechten Maustaste an und wählen Sie aus dem Kontextmenü den Menüpunkt „Entwurfsansicht“ aus:



Access verlangt daraufhin die Eingabe eines Namens für die Tabelle. Geben Sie für die Tabelle den Namen „t_Kundenkonditionen“ ein. Das führende „t“ ist dabei ein Bezeichner, der aussagt, dass es sich bei diesem Access-Objekt um eine Tabelle handelt, was es uns später erleichtert, die verschiedenen Objekttypen in Access voneinander zu unterscheiden:



Die Tabelle öffnet sich daraufhin in der **Entwurfsansicht**:



In der **Entwurfsansicht** sehen wir nicht den eigentlichen Dateninhalt der Tabelle, sondern wie der Name schon sagt deren Entwurf bzw. **Struktur**. In der Entwurfsansicht legen wir fest, welche **Datenfelder** (= **Spalten**) unsere Tabelle umfasst. **Für ein jedes Datenfeld** müssen wir dabei als **Mindestangaben** definieren, welchen **Feldnamen**, welchen **Felddatentyp** und welche **Feldgröße** dieses haben soll. Zudem können noch weitere Feldeigenschaften definiert werden.

Die Tabelle t_Kundenkonditionen soll die folgenden Datenfelder enthalten:

- Kondition (Datenfeldtyp: Zahl, Feldgröße: Integer)
- Rabatt (Datenfeldtyp: Zahl, Feldgröße: Single)
- Skonto (Datenfeldtyp: Zahl, Feldgröße: Single)
- Konditions-Text (Datenfeldtyp: Text, Feldgröße: 30 Zeichen)

Auf die Bedeutung der einzelnen Eigenschaften gehen wir später ein – legen wir zunächst einmal die Struktur unserer Tabelle an:

2.3.1 Anlage des Datenfelds „Kondition“

Legen Sie als erstes den **Feldnamen**, den **Felddatentyp** und die **Feldgröße** für das Datenfeld „Kondition“ fest. Den Felddatentyp und die Feldgröße können Sie jeweils über das entsprechende Dropdownmenü auswählen. Den Feldnamen hingegen müssen Sie von Hand eingeben. Optional können Sie in der Spalte Beschreibung auch noch eine Erläuterung oder Bemerkung zu dem Datenfeld angeben:

Feldname	Felddatentyp	Beschreibung (optional)
Kondition	Zahl	Schlüsselfeld

Feldeigenschaften

Allgemein

Nachschlagen

Feldgröße

Integer

Da dieses Feld die einzelnen Kundenkonditionen angibt bzw. identifiziert, welchen jeweils ein spezifischer Rabatt- und Skontosatz zugeordnet sein wird, ist dieses Feld als **Primärschlüsselfeld** zu bestimmen.

Wie anhand des Schlüsselsymbols in der ersten Spalte zu sehen ist, war dieses Datenfeld in der von Access vorab angelegten Tabelle bereits als **Primärschlüsselfeld** eingerichtet:

Feldname	Felddatentyp
Kondition	Zahl

Wäre dies nicht der Fall, so müssten wir dies noch nachholen, indem wir entweder

- a) das Datenfeld mit der rechten Maustaste anklicken und aus dem Kontextmenü sodann den Menüpunkt „Primärschlüssel“ auswählen:



oder

- b) das Datenfeld mit der linken Maustaste anklicken (so dass es ausgewählt ist und der Cursor darin blinkt) und im Register „Entwurf“ sodann die Schaltfläche „Primärschlüssel“ aktivieren:



Auf die gleiche Weise lässt sich für ein Datenfeld die Eigenschaft als Primärschlüssel auch wieder deaktivieren.

Nun wollen wir noch einige weitere Eigenschaften für das Datenfeld „Kondition“ festlegen. Nehmen Sie hierzu im Bereich „Feldeigenschaften“ die folgenden Eintragungen vor:

Feldeigenschaften

Allgemein		Nachschlagen
Feldgröße	Integer	
Format		
Dezimalstellenanzeige	Automatisch	
Eingabeformat		
Beschriftung	Konditionenart	
Standardwert	0	
Gültigkeitsregel	>0	
Gültigkeitsmeldung	Bitte keine Werte unter 1 eingeben!	
Eingabe erforderlich	Ja	
Indiziert	Ja (Ohne Duplikate)	
Textausrichtung	Standard	

Die Beschriftung ändert die Spaltenbeschriftung in der Datenblattansicht für unsere Tabelle von „Kondition“ in „Konditionenart“, d.h. der Name des Datenfelds wird später in der Datenblattansicht durch den im Eigenschaftsfeld „Beschriftung“ eingegebenen Wert als Spaltenüberschrift ersetzt.

Durch die Gültigkeitsregel „>0“ wird später bei der Eingabe der Daten in unsere Tabelle überprüft, ob die Daten, die in dieses Datenfeld eingegebenen werden, auch tatsächlich größer 0 sind. Sofern bei der Dateneingabe gegen die Gültigkeitsregel verstoßen wird, wird die Dateneingabe mit der Gültigkeitsmeldung „Bitte keine Werte unter 1 eingeben!“ von Access zurückgewiesen.

2.3.2 Anlage des Datenfelds „Rabatt“

Legen Sie nun als nächstes die Eigenschaften für das Datenfeld „Rabatt“ fest:

Feldname	Felddatentyp	Beschreibung
Kondition	Zahl	Schlüsselfeld
Rabatt	Zahl	Rabattsatz

Feldeigenschaften

Allgemein	Nachschlagen
Feldgröße	Single
Format	Prozentzahl
Dezimalstellenanzeige	2
Eingabeformat	
Beschriftung	Rabattsatz
Standardwert	0
Gültigkeitsregel	>=0 Und <=0,3
Gültigkeitsmeldung	Bitte nur Werte eingeben, die größer sind als 0 und gleichzeitig nicht größer als 30%!
Eingabe erforderlich	Nein
Indiziert	Nein
Textausrichtung	Standard

Der Rabattsatz soll als Prozentzahl mit zwei Nachkommastellen dargestellt werden, weshalb hier die entsprechenden Eigenschaften für „Format“ und „Dezimalstellenanzeige“ einzurichten sind. Hinsichtlich der Gültigkeitsregel beachten Sie bitte, dass 30% rechnerisch genau genommen dem Dezimalwert 0,3 entsprechen. Höhere Rabattsätze als 30% sollen nicht erlaubt sein, weshalb die Gültigkeitsregel demgemäß die Eingabe von Werten fordert, die kleiner oder gleich dem Wert 0,3 sind.

2.3.3 Anlage des Datenfelds „Skonto“

Legen Sie nun als nächstes die Eigenschaften für das Datenfeld „Skonto“ fest. Die Eigenschaften für dieses Feld sind analog zu dem des Datenfelds „Rabatt“:

Feldname	Felddatentyp	Beschreibung
Kondition	Zahl	Schlüsselfeld
Rabatt	Zahl	Rabattsatz
Skonto	Zahl	Skontosatz

Feldeigenschaften

Allgemein	Nachschlagen
Feldgröße	Single
Format	Prozentzahl
Dezimalstellenanzeige	2
Eingabeformat	
Beschriftung	Skontosatz
Standardwert	0
Gültigkeitsregel	>=0 Und <=0,03
Gültigkeitsmeldung	Bitte nur Werte eingeben, die größer sind als 0 und gleichzeitig nicht größer als 3%!
Eingabe erforderlich	Nein
Indiziert	Nein
Textausrichtung	Standard

2.3.4 Anlage des Datenfelds „Konditions-Text“

Als letztes muss nun noch das Datenfeld „Konditions-Text“ angelegt werden. Richten Sie dieses mit den folgenden Eigenschaften ein:

Feldname	Felddatentyp	Beschreibung
Kondition	Zahl	Schlüsselfeld
Rabatt	Zahl	Rabattsatz
Skonto	Zahl	Skontosatz
Konditions-Text	Kurzer Text	Beschreibt Skonto und Zahlungsziel

Feldeigenschaften

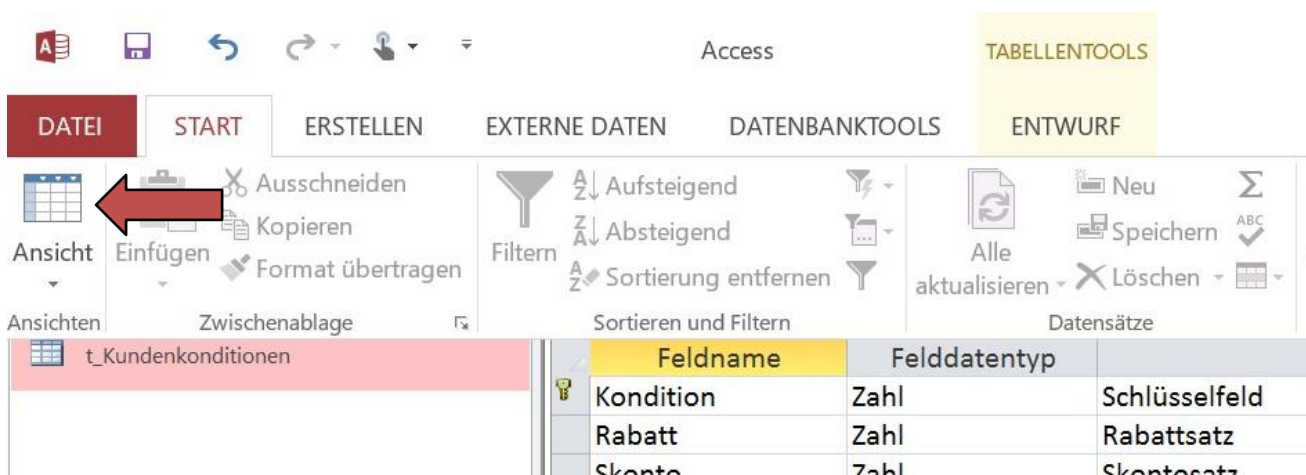
Allgemein	Nachschlagen
Feldgröße	30
Format	
Eingabeformat	
Beschriftung	Gibt das Zahlungsziel an
Standardwert	"10 Tage 2%, 30 Tage netto"

Achten Sie auf die Anführungszeichen beim Texteintrag für die Eigenschaft „Standardwert“!

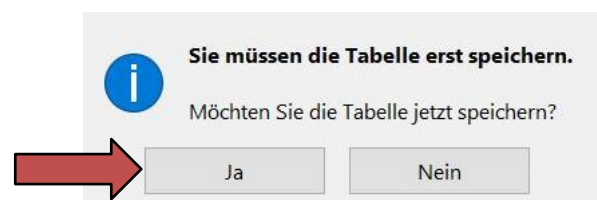
Die Eigenschaft „Standardwert“ definiert einen vordefinierten Wert, der für das betreffende Datenfeld bei der Eingabe automatisch von Access voreingetragen wird. Dies macht z. Bsp. dann Sinn, wenn für die meisten Datensätze der Tabelle in diesem Feld ein bestimmter Wert besonders häufig vorkommt. Auf diese Weise muss der Anwender das Feld nicht jedes Mal von Hand mit dem betreffenden Wert befüllen. Soll das Feld jedoch für manche Datensätze doch mal einen anderen Wert haben, dann ist dieser einfach entsprechend anzupassen.

2.3.5 Speicherung des Tabellenentwurfs & Datenerfassung

Damit haben wir den Entwurf für unsere erste Tabelle der Datenbank abgeschlossen. Um den Tabellenentwurf zu speichern und zur **Datenblattansicht** der Tabelle zu wechseln, klicken Sie nun einfach auf das Symbol „Ansicht“ im Register „Start“ oder „Entwurf“:



Sie werden daraufhin gefragt, ob Sie die Tabelle vor dem Wechsel in die Datenblattansicht speichern möchten, damit die vorgenommenen Änderungen am Tabellenentwurf wirksam werden. Bestätigen Sie dies mit „Ja“:



Anmerkung: Mit dem Wechsel in die Datenblattansicht ändert sich automatisch auch das entsprechende Ansichtssymbol. Mit den beiden Symbolen können Sie **zwischen Entwurfsansicht und Datenblattansicht** hin und her **wechseln**:



Nach dem Wechsel in die Datenblattansicht wird Ihnen nun nicht mehr der Tabellenentwurf, sondern die eigentliche Tabelle mit deren Inhalten angezeigt. Da noch keine Daten eingetragen wurden, ist die Tabelle natürlich noch leer. Befüllen Sie diese daher nun mit den folgenden Daten:

t_Kundenkonditionen				
Konditionenart ▾	Rabattsatz ▾	Skontosatz ▾	Gibt das Zahlungsziel an ▾	
1	0,00%	0,00%	30 Tage netto	
2	0,00%	2,00%	10 Tage 2%, 30 Tage netto	
3	5,00%	2,00%	10 Tage 2%, 30 Tage netto	
4	10,00%	2,00%	10 Tage 2%, 30 Tage netto	
*	0	0,00%	10 Tage 2%, 30 Tage netto	

Die letzte Zeile der Tabelle, welche mit dem * -Symbol markiert ist, ist faktisch kein Datensatz, sondern eine bereits vorangelegte Zeile für den nächsten (neuen) Datensatz. Der Stern repräsentiert als „Geburtssymbol“ quasi die Anlage eines neuen, weiteren Datensatzes.

Übung:

Abschließend wollen wir noch eine kleine Übung vornehmen, um diverse Eigenschaften der verschiedenen Datenfelder zu testen. Legen Sie einen weiteren, neuen Datensatz an und vergeben Sie diesem folgende Werte:

- Konditionenart = 4
- Rabattsatz = 35%
- Skontosatz = 5%
- Konditions-Text = „Der Kunde darf innerhalb von 10 Tagen 5% Skonto abziehen“

Was stellen Sie hinsichtlich des Eingabeversuchs für die verschiedenen Werte fest? Woran könnte dies liegen und wie können Sie dies im Einzelnen lösen?


Löschen Sie anschließend den 5. Datensatz durch **Rechtsklick auf den Datensatz > Datensatz löschen**, so dass die Tabelle schließlich nur die vier ursprünglichen Datensätze enthält. Schließen Sie daraufhin die Tabelle und bestätigen Sie dabei, dass die vorgenommenen Änderungen gespeichert werden sollen.


2.4 Datenfeldeigenschaften

Bei der Erstellung unserer ersten Tabelle haben wir für deren verschiedene Datenfelder eine Reihe von Datenfeldeigenschaften festgelegt, die wir nun noch näher betrachten und erläutern wollen:

2.4.1 Feldname

Bei der Anlage eines Datenfeldes ist in der ersten Spalte die **Bezeichnung für das jeweilige Datenfeld** einzugeben, welche idealerweise bereits auf den Verwendungszweck des Datenfeldes hindeuten sollte:





	Feldname	Felddatentyp	Beschreibung
	Kondition	Zahl	Schlüsselfeld

Ein Feldname kann wie der Name jedes anderen Objektes bis zu 64 Zeichen umfassen und aus einer beliebigen Folge von Ziffern, Buchstaben, Leerstellen und den meisten Sonderzeichen bestehen. Nicht erlaubte Sonderzeichen sind die eckigen Klammern, der Punkt, das Ausrufungszeichen und der Accent grave. Aus arbeitsökonomischen Gründen sollten Sie jedoch keine zu langen Namen verwenden - und aus Gründen der Übersicht nicht aussagekräftige Namen wie N1 oder N2 vermeiden.

2.4.2 Felddatentyp bzw. Datenfeldtyp

In der Spalte „Felddatentyp“ wird festgelegt, welche **Art von Informationen** in diesem Feld gespeichert werden sollen und wieviel Speicherplatz ACCESS dafür reservieren soll. Sie können zu dieser Spalte entweder per Mausklick oder durch Drücken der Tasten <TAB> bzw. <EINGABE> wechseln:



	Feldname	Felddatentyp	Beschreibung
	Kondition	Zahl	Schlüsselfeld

Folgende Felddatentypen stehen dabei über das entsprechende DropDown-Menü zur Auswahl:

Felddatentyp	Beschreibung
Kurzer Text	ACCFSS vergibt als Standard den Felddatentyp „Kurzer Text“ mit einer Länge von 255 Zeichen. In Feldern dieses Typs können Buchstaben, Ziffern oder Sonderzeichen gespeichert werden. Zur Speicherung von Ziffernfolgen, die eine führende Null aufweisen können (so wie dies beispielsweise bei einer Postleitzahl der Fall ist), müssen ebenfalls Datenfelder vom Typ Text verwendet werden. Die Feldgröße eines Datenfeldes vom Typ Text kann von 0 bis maximal 255 Zeichen betragen.
Langer Text	Erlaubt die Speicherung eines Textes mit maximal 64.000 Zeichen und wird beispielsweise für Kommentare und längere Bemerkungen verwendet.
Zahl	Ein Feld dieses Typs kann einen numerischen Wert aufnehmen, mit dem gerechnet werden kann. Welche Zahlen in einem Datenfeld vom Typ „Zahl“ konkret gespeichert werden können, hängt von der Feldgröße ab.
Datum/Uhrzeit	In Feldern dieses Typs können zulässige Datums- und Uhrzeitangaben für die Jahre 100 bis 9999 gespeichert werden. Nicht zulässige Werte werden bei der späteren Datenerfassung abgewiesen:
Währung	Dieser Datentyp ist für Felder vorgesehen, in denen Geldbeträge gespeichert werden sollen. Sie werden standardmäßig mit Tausendertrennzeichen und zwei Dezimalstellen dargestellt.
Autowert	Enthält eine fortlaufende, mit 1 beginnende Zahl, die von ACCESS automatisch um den Wert 1 erhöht wird, wenn der Tabelle ein neuer Datensatz hinzugefügt wird. Dies wird beispielsweise für die fortlaufende Vergabe von Rechnungsnummern oder Kundennummern benötigt.
Ja/Nein	Erlaubt die Speicherung der Wahrheitswerte „Ja/Nein“ oder „Wahr/Falsch“ oder „Ein/Aus“. Dies kann beispielsweise verwendet werden für Felder wie „Lieferbar“ (Ist der Artikel lieferbar?) oder „Gemahnt“ (Wurde der Kunde gemahnt?). Intern speichert ACCESS „Ja“ als 1 und „Nein“ als 0.
OLE-Objekt	Dieser Datentyp ermöglicht die Speicherung von Objekten aus anderen Anwendungen (Diagramme, Bilder, ActiveX-Objekte), die in die Tabelle eingebunden werden.
Link	Enthält einen Hyperlinkverweis auf eine Adresse im Internet oder auf eine externe Datei.
Anlage	Dieser Datentyp ermöglicht das Einfügen von Anlagen bzw. Anhängen wie Bilder, Dokumente, Kalkulationstabellen oder Diagrammen. Jedes Anlage-Feld kann eine unbegrenzte Anzahl von Anlagen pro Datensatz enthalten, bis zum Speichergrenzwert von 2 GB.
Berechnet	Dieser Datentyp ermöglicht das Erstellen eines Rechenausdrucks, der Daten aus mindestens einem Feld verwendet.
Nachschlage-Assistent	Der Nachschlage-Assistent ist nicht wirklich ein Datentyp, sondern er hilft Ihnen dabei, ein Nachschlagefeld zu erstellen, mittels dessen Sie z. Bsp. einen Wert aus einer anderen Tabelle oder aus einer vordefinierten Werteliste auswählen können.

2.4.3 Beschreibung

In der Spalte „Beschreibung“ können Sie eine kurze Erläuterung für jedes Feld angeben.

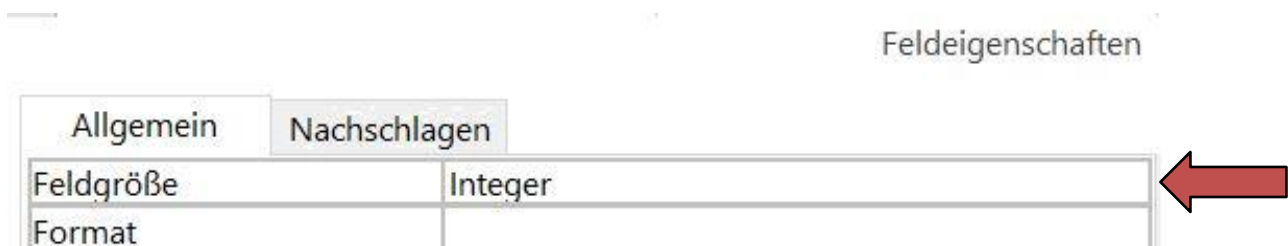


	Feldname	Felddatentyp	Beschreibung
	Kondition	Zahl	Schlüsselfeld

Diese Erläuterung wird später in der Datenblattansicht bei der Erfassung von Daten in der Statusleiste von ACCESS angezeigt, welche sich am unteren Fensterrand befindet. Wenn Sie keine Beschreibung eingeben, so enthält die Statusleiste nur den Vermerk „Datenblattansicht.“

2.4.4 Feldgröße

Für Felder des Datentyps „Kurzer Text“ oder „Zahl“ kann eine Feldgröße definiert werden.



Der Vorgabewert 255 für Textfelder kann in einen Wert zwischen 0 und 255 Zeichen geändert werden.

Wird ein Feld des Datentyps „Zahl“ eingerichtet, so kann im Bereich Feldeigenschaften durch Klicken des Listenfeldpfeils eine geeignete Feldgröße gewählt werden:

Feldgröße	Wertebereich	Dezimalstellen	Speicherbedarf
Byte	0 bis 255	Keine	1 Byte
Integer	-32768 bis +32767	Keine	2 Byte
Long Integer	-2.147.483.648 bis +2.147.483.647	Keine	4 Byte
Single	$-3,4 \times 10^{38}$ bis $+3,4 \times 10^{38}$	7	4 Byte
Double	$-1,797 \times 10^{308}$ bis $+1,797 \times 10^{308}$	15	8 Byte
Dezimal	$-9,999 \times 10^{27}$ bis $+9,999 \times 10^{27}$	28	12 Byte

2.4.5 Format

Definiert das Anzeigeformat für Zahlen, Kalenderdaten oder Uhrzeit. Ist als Datentyp „Zahl“ eingestellt, kann aus dem Listefeld eines der folgenden Formate gewählt werden:

- „Allgemeine Zahl“,
- „Währung“,
- „Euro“ (abhängig von der Einstellung von Windows),
- „Festkommazahl“,

- „Standardzahl“,
- „Prozentzahl“,
- „Exponentialzahl“.

Wird kein Zahlenformat gewählt, wird die Zahl als „Allgemeine Zahl“ dargestellt. Zur Anzeige des Kalenderdatums oder der Uhrzeit stehen sieben Formate zur Verfügung.

2.4.6 Dezimalstellen

Wenn die Anzahl der darzustellenden Dezimalstellen nicht direkt vom gewählten Zahlenformat abhängt (beispielsweise zwei Dezimalstellen beim Währungsformat), können Sie für Zahlenfelder über diese Eigenschaft eine Anzahl zwischen 0 bis 15 Dezimalstellen festlegen.

2.4.7 Eingabeformat

Zur vereinfachten Erfassung von Daten und zur gleichzeitigen Kontrolle der eingegebenen Daten auf formale Richtigkeit kann bei Feldern des Datentyps „Text“ oder „Datum/Zeit“ eine Eingabemaske vorgegeben werden. Die Gestaltung dieser Eingabemaske erfolgt am einfachsten mit Hilfe des Eingabeformatassistenten. Dieser Assistent wird durch die Editor-Schaltfläche rechts neben der aktivierten Eigenschaft „Eingabeformat“ gestartet.

Zur Gestaltung der Eingabemaske stehen insgesamt folgende Zeichen zur Verfügung:

Zeichen	Beschreibung
0	Eingabe einer Ziffer 0-9, Plus- (+) und Minuszeichen (-) sind nicht erlaubt, eine Eingabe ist erforderlich.
9	Eingabe einer Ziffer oder eines Leerzeichens, Plus- und Minuszeichen sind nicht erlaubt, keine Eingabe vorgeschrieben.
#	Eingabe einer Ziffer oder eines Leerzeichens, Plus- und Minuszeichen sind erlaubt, keine Eingabe vorgeschrieben.
L	Eingabe eines Buchstabens A-Z, eine Eingabe ist erforderlich.
?	Eingabe eines Buchstabens, keine Eingabe vorgeschrieben.
A	Eingabe eines Buchstabens oder einer Ziffer, eine Eingabe ist erforderlich.
a	Eingabe eines Buchstabens oder einer Ziffer, keine Eingabe vorgeschrieben.
&	Eingabe eines beliebigen Zeichens oder eines Leerzeichens, eine Eingabe ist erforderlich.
C	Eingabe eines beliebigen Zeichens oder eines Leerzeichens, keine Eingabe vorgeschrieben.
.,:;/	Platzhalter für Dezimal-, Tausender-, Datums- und Zeit-Trennzeichen.
>	Alle nachfolgenden Zeichen werden in Großbuchstaben umgewandelt.
<	Alle nachfolgenden Zeichen werden in Kleinbuchstaben umgewandelt.
!	Die Eingabe erfolgt nicht von links nach rechts, sondern von rechts nach links. Diese Einstellung wirkt sich nur dann aus, wenn keine Eingabe vorgeschrieben ist (dies ist bei Verwendung der Zeichen 9 # ? a C der Fall).
"ab"	Ein beliebiger Text, der unverändert in der Eingabemaske dargestellt wird und nicht überschrieben werden kann.
\	Das nachfolgende Zeichen wird als literales Zeichen dargestellt (beispielsweise wird \A als A angezeigt).

2.4.8 Beschriftung

In der Datenblattansicht zeigt ACCESS als Spaltenüberschrift den Feldnamen an. Wurde als Feldname beispielsweise die verkürzte Bezeichnung „LiefNr“ gewählt, kann in der Eigenschaft Beschriftung die Bezeichnung „Lieferernummer“ eingetragen werden, was sodann in der Datenblattansicht ersatzweise anstatt des Feldnamens als Spaltenüberschrift angezeigt wird.

2.4.9 Standardwert

Diese Eigenschaft kann einen Text, eine Zahl oder einen Wahrheitswert enthalten, den jeder neu angelegte Datensatz in dem betreffenden Feld automatisch erhält, beispielsweise in einer Artikeltabelle den Wert 19% als allgemeinen Umsatzsteuersatz für den betreffenden Artikel. Wenn ein Standardwert für ein Datenfeld vom Datentyp „Kurzer Text“ festgelegt wird, so ist dieser in Anführungszeichen einzugeben, da es sich hierbei ja um ein Textfeld handelt (siehe Datenfeld Konditions-Text in der Tabelle t_Kundenkonditionen).

2.4.10 Gültigkeitsregel

In dieser Zeile können Bedingungen für die Dateneingabe in das jeweilige Datenfeld definiert werden. ACCESS prüft bei der Dateneingabe in diesem Feld, ob diese Bedingungen erfüllt sind. In einem Feld „Rabatt“ kann beispielsweise durch die Bedingung $\leq 0,2$ verhindert werden, dass ein höherer Rabattsatz eingegeben wird. Die zu prüfenden Bedingungen werden als Ausdrücke eingegeben. Sie können Feldbezeichnungen, Datumsangaben, Text, arithmetische Operatoren, Vergleichsoperatoren oder logische Operatoren enthalten. Feldbezeichnungen müssen davor und danach eine eckige Klammer [] aufweisen. Datumsangaben müssen innerhalb des Nummernzeichenpaars ## stehen (z. Bsp. $>\#03.03.1916\#$), Texte innerhalb der Anführungszeichen "" (z. Bsp. $<>\text{„Meier“}$).

Als **arithmetische Operatoren** kommen vor allem die Rechenoperatoren +, -, * und / in Betracht.

Als **Vergleichsoperatoren** sind zulässig:

Operator	Beispiel	Bedeutung
<	<50	kleiner als 50
<=	<=50	kleiner als oder gleich 50
>	>50	größer als 50
>=	>=50	größer als oder gleich 50
=	=50	gleich 50
<>	<>50	ungleich 50

Komplexe Ausdrücke können mit folgenden **logischen Operatoren** erstellt werden:

Operator	Beispiel	Bedeutung
Und	>20 Und <30	Wert muss größer als 20 und kleiner als 30 sein
Oder	0 Oder >30	Wert muss entweder 0 oder größer als 30 sein
Nicht	Nicht #01.01.2016#	Das Datum 01.01.2016 wird bei der Eingabe zurückgewiesen

ACCESS kennt außerdem die vier weiteren Operatoren

Beispiel	Bedeutung
Zwischen 30 Und 50	Der Wert muss innerhalb dieser Grenzen liegen
In(„USA“;„Kanada“)	Nur einer der beiden Texte kann eingegeben werden
Ist Nicht Null	Durch das Schlüsselwort „Null“ wird in Verbindung mit dem logischen Operator „Nicht“ festgelegt, dass ein Leerwert ungültig ist
Wie „AC*“	Der eingegebene Text muss mit den Buchstaben AC beginnen
Wie „AC????“	Wie zuvor, der Text muss jedoch sechs Zeichen umfassen
Wie „A[a-d]##“	Der Text muss mit dem Buchstaben A beginnen, gefolgt von einem Buchstaben im Bereich von einschließlich „a“ bis einschließlich „d“ und zwei folgenden Ziffern

2.4.11 Gültigkeitsmeldung

Sie wird anstelle der ACCESS-Standardmeldung angezeigt, wenn eine Verletzung der Gültigkeitsregel vorliegt. Geben Sie in dieses Feld einen entsprechenden Meldetext, beispielsweise „Unzulässige Lieferernummer“ ein.

2.4.12 Eingabe erforderlich

Durch diese Eigenschaft können Sie festlegen, dass in dieses Feld zwingend ein Wert eingegeben werden muss. Ein Datensatz wird somit erst dann gespeichert, wenn in diesem Feld ein Wert erfasst wurde. Ändern Sie dazu die Standardeinstellung „Nein“ auf „Ja“ ab.

2.4.13 Leere Zeichenfolge

Durch diese nur bei den Datentypen „Kurzer Text“ und „Langer Text“ definierbare Eigenschaft mit der Standardeinstellung „Nein“ legen Sie durch die Einstellung „Ja“ fest, dass bei der Eingabe von zwei Anführungszeichen ohne dazwischen liegende Zeichen eine leere Zeichenfolge gespeichert wird. ACCESS unterscheidet bei der Auswertung des Feldes in einer Abfrage, ob dieses Feld leer ist (einen Nullwert enthält) oder nichts (eine leere Zeichenfolge) enthält.

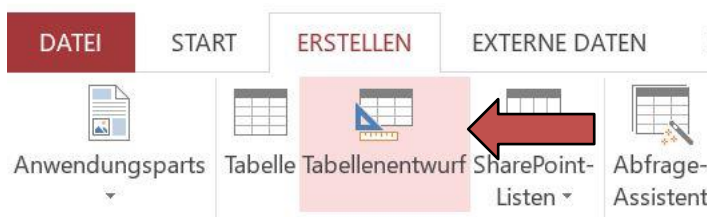
2.4.14 Indiziert

Suchoperationen in einem Feld können schneller ablaufen, wenn das betreffende Feld indiziert wurde. Beim Indizieren richtet ACCESS im Hintergrund eine für den Anwender unsichtbare Index-Tabelle ein, bei welcher die in diesem Datenfeld für die verschiedenen Datensätze gespeicherten Datenfeldwerte aufsteigend sortiert sind, wodurch das Auffinden eines bestimmten Datenfeldwertes wesentlich schneller abläuft, da aufgrund der aufsteigenden Sortierung nicht die gesamte Tabelle bzw. jeder Datensatz nach dem betreffenden Wert durchsucht werden muss, sondern aufgrund der aufsteigenden Sortierung direkt zu dem gesuchten Wert bzw. zu den entsprechenden Datensätzen gesprungen werden kann. Ein solcher Index ist vergleichbar zu dem alphabetischen Index im hinteren Teil eines Buches, wo für bestimmte Fachbegriffe die entsprechenden Seitenzahlen des Buches angegeben sind, auf welchen zu dem jeweiligen Begriff etwas geschrieben steht. Dadurch können die entsprechenden Seiten bzw. Informationen zu dem Begriff direkt nachgeschlagen werden, anstatt das gesamte Buch nach dem betreffenden Begriff durchsuchen zu müssen.

ACCESS benötigt zudem ein eindeutiges Unterscheidungskriterium für die Datensätze in einer Tabelle. Ein solches Unterscheidungskriterium wird als **Primärschlüssel** bezeichnet. Felder, die Primärschlüssel sind, sind grundsätzlich indiziert und erhalten von ACCESS automatisch die Eigenschaft „Ja (Ohne Duplikate)“.

2.5 Erstellung der Tabelle t_Orte

Fahren Sie nun mit der Erstellung der Datenbank fort. Sie können hierzu über den folgenden Link den [Zwischenstand](#) von der letzten Übung zur Erstellung der Tabelle „t_Kundenkonditionen“ aufrufen. Als nächstes ist die Tabelle t_Orte zu erstellen. Rufen Sie hierzu in Access das Register „Erstellen“ auf und wählen Sie aus diesem den Befehl „Tabellenentwurf“ aus:



Daraufhin wird eine neue Tabelle erstellt und in der Entwurfsansicht angezeigt. Legen Sie hier als erstes Datenfeld das Datenfeld „PLZ“ mit dem Felddatentyp „Kurzer Text“ an. Achten Sie zudem darauf, dass Sie dieses Feld als Primärschlüsselfeld einrichten, da über die PLZ die Identifikation der verschiedenen Orte in unserer Tabelle erfolgt:



Anmerkung: Da die Postleitzahlen nur aus Ziffern bestehen, könnte man unterstellen, dass für das Feld „PLZ“ der Datentyp „Zahl“ angemessen sei – da man mit Postleitzahlen jedoch nicht rechnen muss und es zudem auch Postleitzahlen mit führender Null gibt, welche von einem Zahlenfeld nicht angezeigt würde, ist für das Datenfeld „PLZ“ der Datentyp „Kurzer Text“ die geschicktere Wahl.

Geben Sie als Feldgröße 5 Zeichen an (mehr benötigen wir für deutsche Postleitzahlen nicht) und klicken Sie danach auf die DropDown-Schaltfläche hinter der Feldeigenschaft „Eingabeformat“:

Allgemein		Nachschlagen
Feldgröße	5	
Format		
Eingabeformat		

Es erscheint nun die Meldung, dass die Tabelle zuerst gespeichert werden muss, bevor das Dialogfenster für das Eingabeformat aufgerufen werden kann. Bestätigen Sie die Frage „Jetzt speichern?“ mit Ja und benennen Sie die Tabelle „t_Orte“:

Speichern unter ? X

Tabellenname:

t_Orte

OK Abbrechen

Nach dem Speichern der Tabelle erscheint das Dialogfenster zur Festlegung des Eingabeformats. Wählen Sie hier das Eingabeformat „PLZ“ und klicken Sie dann auf „Weiter“:

Eingabeformat:	Darstellung:
Telefonnummer, intl.	+49 55454 23245656
PLZ, intl.	D-44534
Telefonnummer	(555)43216753
Telefonnummer	(55454)23245656
PLZ	44534
ISBN	ISBN 1-55615-877-7

Es erscheint ein weiteres Fenster, in welchem Sie das Eingabeformat manuell ändern und zudem ein Platzhalterzeichen auswählen können, welches später bei der Eingabe der Postleitzahlen für die einzelnen Ziffern angezeigt wird. Ändern Sie das Platzhalterzeichen in ein #-Zeichen ab:

Eingabeformatname: Benutzerdefiniert

Eingabeformat: 00000

Welches Platzhalterzeichen soll im Feld angezeigt werden?

Während Sie Daten in das Feld eingeben, werden die Platzhalter ersetzt.

Platzhalterzeichen: #

Die fünf Nullen im Eingabeformat stehen für die 5 Stellen der Postleitzahlen und bedeuten, dass bei der Eingabe jeweils nur Ziffern für die 5 PLZ-Stellen zulässig sind (siehe Kapitel 2.4.7). Klicken Sie nochmals auf „Weiter“.

Im nächsten Feld erscheint die Frage, wie die Daten gespeichert werden sollen.

Wie sollen die Daten gespeichert werden?

☒ Mit den Symbolen im folgenden Eingabeformat:
83331

☐ Ohne die Symbole im folgenden Eingabeformat:
52538

Ändern Sie die Voreinstellung auf „Mit den Symbolen ...“ und klicken Sie erneut auf „Weiter“ und im nächsten Fenster sodann auf „Fertigstellen“. Der Eingabeformat-Assistent legt daraufhin auf Basis Ihrer vorgenommenen Einstellungen folgenden Eintrag im Eigenschaftsfeld „Eingabeformat“ an:

Allgemein	Nachschlagen
Feldgröße	5
Format	
Eingabeformat	00000;0;#

Sie erinnern sich: die 5 Nullen stehen für das zulässige Eingabeformat der 5 PLZ-Stellen (nur Ziffern), das #-Zeichen ist das festgelegte Platzhalterzeichen und die einzelne Null zwischen dem Eingabeformat und dem Platzhalterzeichen bedeutet, dass die Daten „mit den Symbolen“ gespeichert werden. Die drei Parameter werden dabei jeweils durch Semikola voneinander abgegrenzt.

Stellen Sie für das Datenfeld PLZ nun noch die folgenden Feldeigenschaften ein:

Beschriftung	Postleitzahl
Standardwert	
Gültigkeitsregel	
Gültigkeitsmeldung	
Eingabe erforderlich	Ja
Leere Zeichenfolge	Nein
Indiziert	Ja (Ohne Duplikate)

Die Eigenschaft „Eingabe Erforderlich = Ja“ bedeutet, dass das Datenfeld bei der Eingabe zwingend befüllt werden muss. Die Eigenschaft „Leere Zeichenfolge = Nein“ hingegen bedeutet, dass es zudem auch nicht zulässig ist, das Datenfeld mit einer leeren Zeichenfolge (zwei aufeinander folgende Anführungszeichen ohne dazwischen liegenden Inhalt) zu befüllen (siehe Kapitel 2.4.13).

Legen Sie nun als weiteres Datenfeld noch das Feld „Ort“ mit dem Felddatentyp „Kurzer Text“ und einer Feldgröße von 30 Zeichen an. Legen Sie zudem auch die übrigen Feldeigenschaften gemäß der folgenden Abbildung fest:

Feldname	Felddatentyp
PLZ	Kurzer Text
Ort	Kurzer Text

Feldeigenschaften

Allgemein	Nachschlagen
Feldgröße	30
Format	
Eingabeformat	
Beschriftung	Ort
Standardwert	
Gültigkeitsregel	
Gültigkeitsmeldung	
Eingabe erforderlich	Ja
Leere Zeichenfolge	Nein

Unser Tabellenentwurf für die Tabelle „t_Orte“ wäre damit abgeschlossen. Wechseln Sie nun in die Datenblattansicht und bestätigen Sie dabei die Frage, dass Sie die Tabelle speichern möchten.

Die Tabelle „t_Orte“ wird daraufhin in der Datenblattansicht angezeigt. Natürlich sind noch keine Daten darin enthalten:

t_Orte		
Postleitzahl	Ort	Zum Hinzufügen klicken
*		

Im Grunde genommen könnten wir nun damit beginnen, die Tabelle mit Daten zu befüllen. Für diesen Zweck wollen wir jedoch zunächst noch ein Formular erstellen, über welches wir die Daten sodann eingeben werden. Rufen sie hierzu das Register „Erstellen“ auf und wählen Sie in diesem den Befehl „Formular“ aus:

DATEI	START	ERSTELLEN	EXTERNE DATEN	DATENBANKTOOLS	FELDER	TABELLE
Anwendungsparts	Tabelle	Tabellenentwurf	SharePoint-Listen	Abfrage-Assistent	Abfrageentwurf	Formular

Formularentwurf Leeres Formular

Access legt daraufhin automatisch ein Formular an, welches für jedes Datenfeld unserer Tabelle ein entsprechendes Eingabefeld enthält. Das Formular ist dabei gleichsam automatisch mit unserer Tabelle verknüpft worden – d.h. alle Eingaben über dieses Formular werden automatisch

in der Tabelle „t_Orte“ gespeichert. Das Formular wird uns jedoch zunächst nur in der Entwurfsansicht angezeigt, in welcher es nicht möglich ist, Daten darin einzugeben. Wechseln Sie daher über den entsprechenden Befehl im Register „Start“ oder im Register „Entwurf“ in die Formularansicht:



Geben Sie **in das Formular** nun die folgenden Postleitzahlen mit den zugehörigen Orten ein:

PLZ	Ort
48683	Ahaus
48599	Gronau
46325	Borken
48653	Coesfeld
48734	Reken

Hinweis: Wie Ihnen bei der Eingabe der Postleitzahlen gewiss auffällt, wird in dem Datenfeld PLZ für jede einzugebende PLZ-Stelle das Platzhalterzeichen angezeigt, das wir zuvor bei der Definition des Eingabeformats für dieses Datenfeld festgelegt hatten. Durch die Platzhalterzeichen sieht der Anwender sofort, wie viele Zeichen er in das Datenfeld eingeben muss:



Schließen Sie das Formular, nachdem Sie alle 5 Postleitzahlen mit den zugehörigen Orten eingegeben haben. Beim Schließen des Formulars werden Sie gefragt, ob Sie die vorgenommenen Änderungen speichern wollen. Bestätigen Sie dies mit „Ja“ und speichern Sie das Formular unter dem Namen „f_Orte“ (das voran gestellt Kürzel „f“ gibt uns dabei den Hinweis, dass es sich bei diesem Objekt um ein Formular handelt):

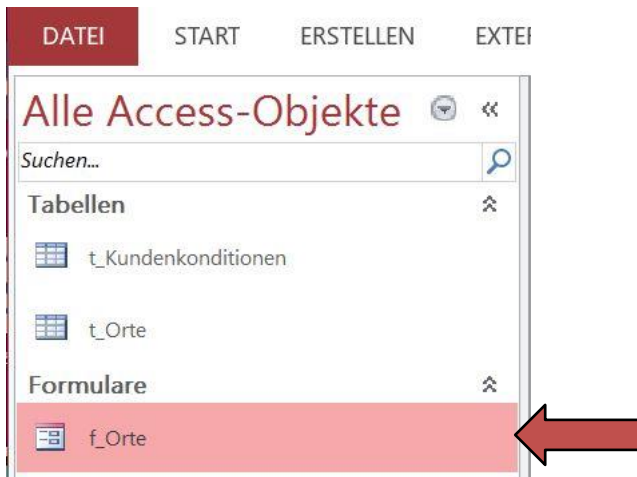
Speichern unter ? X

Formularname:

f_Orte

OK Abbrechen

Nach dem Speichern des Formulars wird Ihnen dieses nun in der am linken Fensterrand befindlichen Objektliste Ihrer Datenbank auch als entsprechendes Access-Objekt angezeigt:



Rufen Sie nun Ihre **Tabelle** „t_Orte“ in der Datenblattansicht auf. (Falls Sie diese noch geöffnet hatten, müssen Sie diese ggf. zuerst schließen und sodann nochmals erneut öffnen, damit die Ansicht aktualisiert wird). Wie Sie sehen können, wurden die Eingaben, die Sie im Formular vorgenommen hatten automatisch in die Tabelle übernommen:

Postleitzahl	Ort	Zum Hinzufügen klicken
46325	Borken	
48599	Gronau	
48653	Coesfeld	
48683	Ahaus	
48734	Reken	
*		

2.6 Import der übrigen Tabellen

Nun müssen wir unsere Datenbank noch durch die 3 übrigen Tabellen (t_Artikel, t_Kunden und t_Aufträge) vervollständigen. Diese wollen wir wie bereits vorab erwähnt aus bereits bestehenden Dateien importieren, welche Sie im Unterordner **Übungen\Importdateien** finden. Zum Fortsetzen Ihrer Arbeit können über den folgenden Link den [Zwischenstand](#) von der letzten Übung zur Erstellung der Tabelle „t_Orte“ aufrufen.

2.6.1 Import der Tabelle t_Artikel

Die Tabelle t_Artikel liegt bereits als Tabelle in einer anderen Access-Datenbank vor, nämlich in der Datenbank-Datei **Artikeldatenbank.accdb**. Um die Tabelle aus dieser Datenbank-Datei in unsere Datenbank zu importieren, rufen Sie bitte den Befehl „Access“ aus dem Register „EXTERNE DATEN“ auf:



Wählen Sie in dem daraufhin erscheinenden Dialogfenster die Datei **Artikeldatenbank.accdb** aus dem Unterordner **Übungen\Importdateien** als Import-Quelle aus und aktivieren Sie die Option den Punkt „Importieren Sie Tabellen ... in die aktuelle Datenbank“:

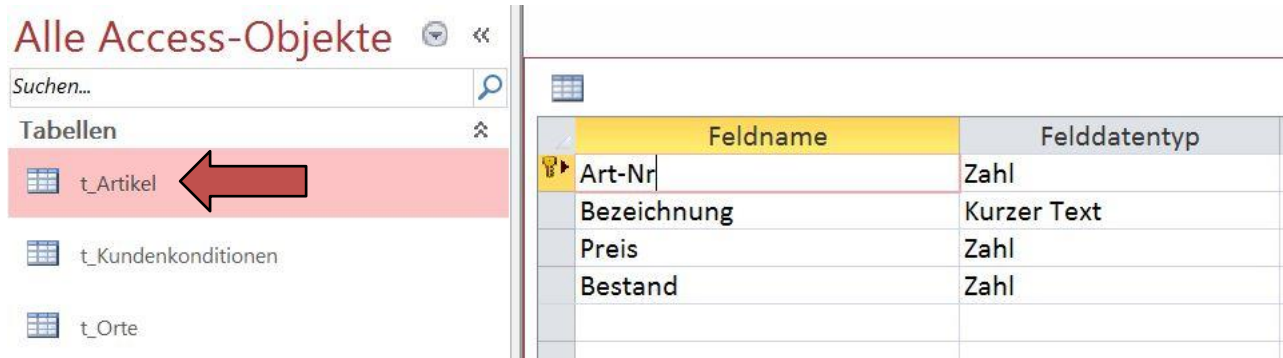


Im nächsten Dialogfenster wählen Sie aus dem Register „Tabellen“ bitte die Tabelle „t_Artikel“ aus (da die Datei **Artikeldatenbank.accdb** nur diese eine Tabelle enthält, besteht keine andere Auswahlmöglichkeit):



Die daraufhin erscheinende Frage, ob die Importschritte gespeichert werden sollen, können Sie verneinen bzw. ignorieren und den Import-Assistenten schließen.

Wie Sie daraufhin feststellen können, wurde die Tabelle „t_Artikel“ in unsere Datenbank übernommen:



Überprüfen Sie in der Entwurfsansicht die Struktur der Tabelle sowie die Eigenschaften der Datenfelder und schauen Sie sich anschließend in der Datenblattansicht die Artikeldaten an.

2.6.2 Import der Tabelle t_Aufträge

Die Tabelle t_Aufträge liegt bereits in Form einer Excel-Tabelle vor, nämlich in der Datei **Aufträge.xlsx**. Um die Daten aus dieser Excel-Datei in unsere Datenbank zu importieren, rufen Sie bitte den Befehl „Excel“ aus dem Register „EXTERNE DATEN“ auf:



Wählen Sie in dem daraufhin erscheinenden Dialogfenster die Datei **Aufträge.xlsx** aus dem Unterordner **Übungen\Importdateien** als Import-Quelle aus und aktivieren Sie die Option den Punkt „Importieren Sie Quelldaten in eine neue Tabelle in der aktuellen Datenbank“:



Wählen Sie in dem daraufhin erscheinenden Importassistenten die Option „Arbeitsblätter anzeigen“ und sodann das Arbeitsblatt „AUFTRÄGE“ aus der ausgewählten Excel-Datei aus (da die Excel-Datei *Aufträge.xlsx* nur dieses eine Arbeitsblatt enthält, besteht keine andere Auswahlmöglichkeit). Im unteren Bereich des Dialogfensters wird Ihnen sodann eine entsprechende Ansicht vom Inhalt des ausgewählten Excel-Arbeitsblattes angezeigt:

Import-Assistent für Kalkulationstabellen

Ihre Tabellenkalkulationsdatei enthält mehrere Arbeitsblätter oder Bereiche. Welches Blatt bzw. welchen Bereich möchten Sie anzeigen?

☒ **Arbeitsblätter anzeigen** ☐ Benannte Bereiche anzeigen

AUFTRÄGE

Beispieldaten für Arbeitsblatt 'AUFTRÄGE'.

	Auftrags-Nr	Kd-Nr	Art-Nr	Anzahl	Datum
1	1	591	0110	3	17.08.2016
2	2	591	0304	1	17.08.2016
3	3	591	6010	1	17.08.2016
4	4	361	3152	2	21.08.2016

Da die erste Zeile des Excel-Arbeitsblattes die Spaltenüberschriften für die einzelnen Tabellenspalten enthält, markieren Sie im nächsten Fenster des Import-Assistenten bitte entsprechend die Option „Erste Zeile enthält Spaltenüberschriften“:

Import-Assistent für Kalkulationstabellen

Microsoft Access kann Ihre Spaltenüberschriften als Feldnamen verwenden. Enthält die erste angegebene Zeile Spaltenüberschriften?

☒ **Erste Zeile enthält Spaltenüberschriften**

	Auftrags-Nr	Kd-Nr	Art-Nr	Anzahl	Datum
1	1	591	0110	3	17.08.2016
2	2	591	0304	1	17.08.2016
3	3	591	6010	1	17.08.2016
4	4	361	3152	2	21.08.2016

Bevor Sie auf „Weiter“ klicken, testen Sie einfach mal, was passiert, wenn Sie diese Option deaktivieren. In der Tabellenblattansicht darunter können Sie den Unterschied beobachten. Was würde beim Import geschehen, wenn Sie die Option in diesem Fall deaktiviert belassen würden?

Denken Sie daran, das Häkchen wieder zu setzen, bevor Sie auf „Weiter“ klicken!

Im nächsten Schritt des Import-Assistenten haben Sie nun die Möglichkeit, für eine jede Spalte der Tabelle bestimmte Eigenschaften festzulegen. Da die Spalten der Excel-Tabelle den Datenfeldern unserer späteren Access-Tabelle entsprechen, sind dies dementsprechend die wesentlichen **Datenfeldeigenschaften**, nämlich der **Feldname**, der **Datentyp** und **Feldgröße** sowie ob das Datenfeld **indiziert** werden soll oder nicht. Außerdem können Sie für bestimmte Spalten auch auswählen, dass diese nicht als Datenfelder importiert werden sollen. Um die Eigenschaften für ein Datenfeld festzulegen, ist die entsprechende Spalte in der Ansicht zunächst zu markieren.


Legen Sie für das Datenfeld „Auftrags-Nr“ folgende Eigenschaften fest (da dies später unser Primärschlüsselfeld für diese Tabelle sein wird, legen wir bereits beim Import fest, dass es ein indiziertes Feld ohne Duplikate sein wird):

Import-Assistent für Kalkulationstabellen

Sie können Informationen zu jedem zu importierenden Feld angeben. Wählen Sie dazu Felder aus dem unten stehenden Bereich aus, und bearbeiten Sie dann die Feldinformationen im Bereich 'Feldoptionen'.

Feldoptionen

Feldname: Datentyp:
 Indiziert: ☐ Feld nicht importieren (Überspringen)




	Auftrags-Nr	Kd-Nr	Art-Nr	Anzahl	Datum
1	1	591	0110	3	17.08.2016
2	2	591	0304	1	17.08.2016
3	3	591	6010	1	17.08.2016

Für das Datenfeld „Kd-Nr“ legen Sie bitte folgende Eigenschaften fest:

Feldoptionen


Feldname: Datentyp:
 Indiziert: ☐ Feld nicht importieren (Überspringen)



	Auftrags-Nr	Kd-Nr	Art-Nr	Anzahl	Datum
1	1	591	0110	3	17.08.2016
2	2	591	0304	1	17.08.2016

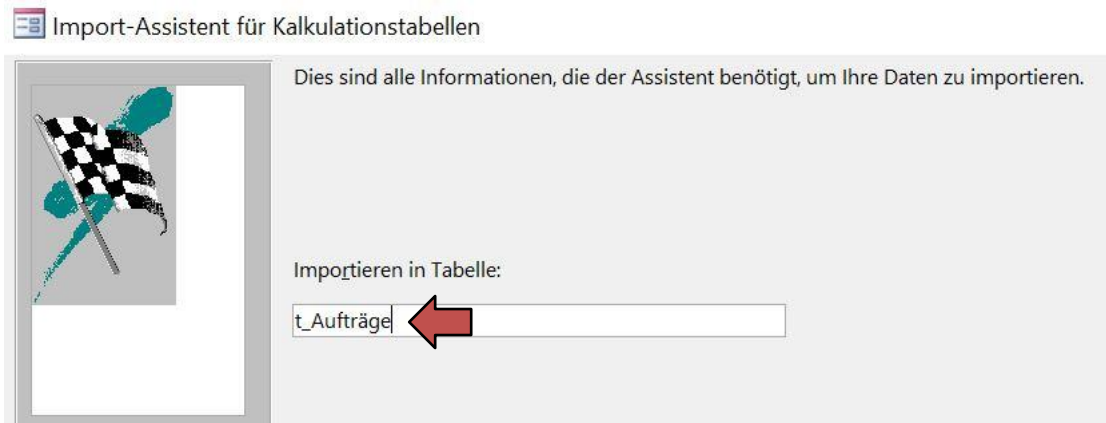
Alle anderen Datenfelder belassen Sie bitte wie voreingestellt und klicken sodann auf „Weiter“.

Im nächsten Schritt des Importassistenten können Sie nun noch festlegen, welches Datenfeld den Primärschlüssel der Tabelle bilden soll. Wir möchten in diesem Fall das Datenfeld „Auftrags-Nr“ als Primärschlüsselfeld verwenden. Aktivieren Sie daher die Option „Eigenen Primärschlüssel auswählen“ und wählen Sie im DropDown-Feld dahinter das Datenfeld „Auftrags-Nr“ aus:

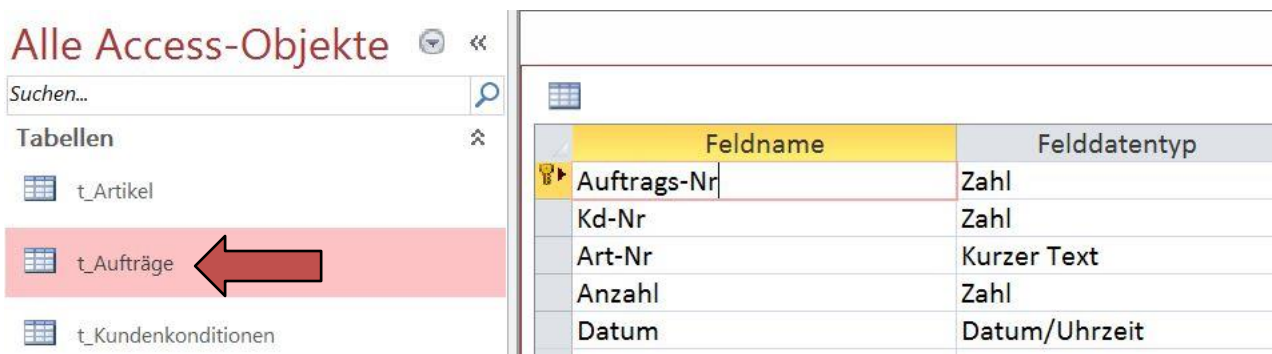


☐ Primärschlüssel soll von Access hinzugefügt werden
☒ **Eigenen Primärschlüssel auswählen**
☐ Kein Primärschlüssel

Zu guter Letzt müssen wir der importierten Tabelle in unserer Datenbank auch noch einen Namen geben. Access schlägt hier standardmäßig den Arbeitsblattnamen des entsprechenden Excel-Arbeitsblattes vor. Ändern Sie diesen in „t_Aufträge“ und schließen Sie den Import durch Klick auf „Fertig stellen“ ab:



Wie Sie daraufhin feststellen können, wurde die Tabelle „t_Aufträge“ in unsere Datenbank übernommen:



Überprüfen Sie in der Entwurfsansicht die Struktur der Tabelle sowie die Eigenschaften der Datenfelder und schauen Sie sich anschließend in der Datenblattansicht die Auftragsdaten an.

2.6.3 Import der Tabelle t_Kunden

Die Tabelle t_Kunden liegt bereits in Form einer Text-Datei vor, nämlich in der Datei **t_Kunden.txt**. Um die Daten aus dieser Datei in unsere Datenbank zu importieren, rufen Sie bitte den Befehl „Textdatei“ aus dem Register „EXTERNE DATEN“ auf:



Wählen Sie in dem daraufhin erscheinenden Dialogfenster die Datei **t_Kunden.txt** aus dem Unterordner **Übungen\Importdateien** als Import-Quelle aus und aktivieren Sie die Option den Punkt „Importieren Sie Quelldaten in eine neue Tabelle in der aktuellen Datenbank“:

Wählen Sie Quelle und Ziel der Daten aus

Geben Sie die Quelle für die Definition der Objekte an.

Dateiname: Durchsuchen...

Geben Sie an, wie und wo Sie die Daten in der aktuellen Datenbank speichern möchten.

☒ **Importieren Sie die Quelldaten in eine neue Tabelle in der aktuellen Datenbank.**

Wenn die angegebene Tabelle nicht vorhanden ist, wird sie durch Access erstellt. Wenn die angegebene Tabelle bereits vorhanden ist, werden die Inhalte durch Access möglicherweise mit den importierten Daten überschrieben. Änderungen an den Quelldaten werden nicht in die Datenbank übernommen.

Da die Daten der einzelnen Datenfelder in der Text-Datei durch Tabulator-Trennzeichen voneinander getrennt sind, wählen Sie in dem daraufhin erscheinenden Import-Assistenten bitte die Option „Mit Trennzeichen“ aus:

Textimport-Assistent

Es scheint, dass Ihre Daten im Format 'Mit Trennzeichen' vorliegen. Sollte das nicht der Fall sein, wählen Sie das Format aus, das Ihre Daten genauer beschreibt.

☒ **Mit Trennzeichen - Felder sind durch Zeichen, wie z.B. Komma oder Tabulator, voneinander getrennt.**

☐ Feste Breite - Felder sind in Spalten angeordnet, die durch Leerzeichen unterteilt sind.

Im nächsten Fenster des Import-Assistenten geben Sie als Trennzeichen demgemäß bitte das Tabulatorzeichen an. Da die erste Zeile der Textdatei die Feldnamen für die einzelnen Datenfelder enthält, markieren Sie zudem bitte entsprechend die Option „Erste Zeile enthält Feldnamen“. Im unteren Bereich des Dialogfensters wird Ihnen im Übrigen eine entsprechende Voransicht vom Inhalt der späteren Tabelle angezeigt:

Textimport-Assistent

Welches Trennzeichen unterteilt Ihre Felder? Wählen Sie das passende Trennzeichen aus, und beobachten Sie unten in der Vorschau, wie sich der Text dadurch ändert.

Wählen Sie das Trennzeichen aus, das Ihre Felder unterteilt:

☒ **Tabulatorzeichen** ☐ Semikolon ☐ Komma ☐ Leerzeichen ☐ Anderes:

☒ **Erste Zeile enthält Feldnamen** Textqualifizierer: v

Kd-Nr	Name1	Name2	Strasse	PLZ	Kondition
482	Baubetriebe	Zabel	Bachweg 6	48683	4
294	Restaurant	Zur Wilden Sau	August-Schmidt-Ring 23	48683	3

Im nächsten Schritt des Import-Assistenten haben Sie nun – analog zum bereits zuvor erfolgten Import der Excel-Tabelle – die Möglichkeit, für eine jede Spalte der Tabelle bestimmte Eigenschaften festzulegen.

Legen Sie für das Datenfeld „Kd-Nr“ folgende Eigenschaften fest (da dies später unser Primärschlüsselfeld für diese Tabelle sein wird, legen wir bereits beim Import fest, dass es ein indiziertes Feld ohne Duplikate sein wird):

Textimport-Assistent

Sie können Informationen zu jedem zu importierenden Feld angeben. Wählen Sie dazu Felder aus dem unten stehenden Bereich aus, und bearbeiten Sie dann die Feldinformationen im Bereich 'Feldoptionen'.

Feldoptionen

Feldname: Datentyp:

Indiziert: ☐ Feld nicht importieren (Überspringen)

Kd-Nr	Name1	Name2	Strasse	PLZ	Kondition
482	Baubetriebe	Zabel	Bachweg 6	48683	4
294	Restaurant	Zur Wilden Sau	August-Schmidt-Ring 23	48683	3

Für das Datenfeld „PLZ“ legen Sie bitte folgende Eigenschaften fest:

Textimport-Assistent

Sie können Informationen zu jedem zu importierenden Feld angeben. Wählen Sie dazu Felder aus dem unten stehenden Bereich aus, und bearbeiten Sie dann die Feldinformationen im Bereich 'Feldoptionen'.

Feldoptionen

Feldname: Datentyp:

Indiziert: ☐ Feld nicht importieren (Überspringen)

Kd-Nr	Name1	Name2	Strasse	PLZ	Kondition
482	Baubetriebe	Zabel	Bachweg 6	48683	4
294	Restaurant	Zur Wilden Sau	August-Schmidt-Ring 23	48683	3

Alle anderen Datenfelder belassen Sie bitte wie voreingestellt und klicken sodann auf „Weiter“.

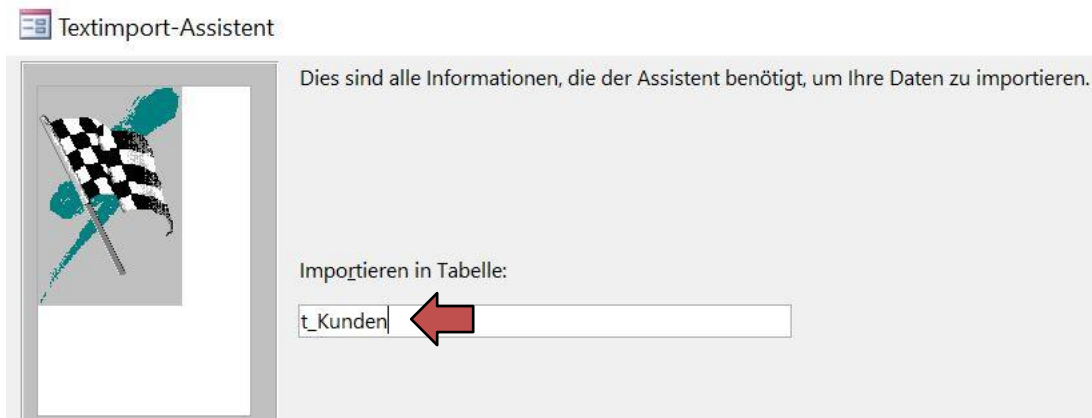
Im nächsten Schritt des Importassistenten können Sie nun wieder festlegen, welches Datenfeld den Primärschlüssel der Tabelle bilden soll. Wir möchten in diesem Fall das Datenfeld „Kd-Nr“ als Primärschlüsselfeld verwenden. Aktivieren Sie daher die Option „Eigenen Primärschlüssel auswählen“ und wählen Sie im DropDown-Feld dahinter das Datenfeld „Kd-Nr“ aus:

☐ Primärschlüssel soll von Access hinzugefügt werden

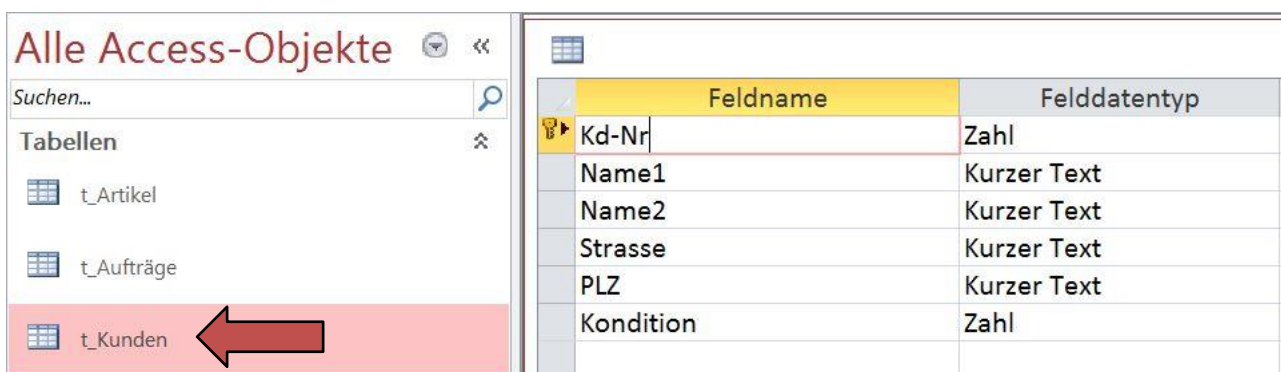
☒ **Eigenen Primärschlüssel auswählen**

☐ Kein Primärschlüssel

Zu guter Letzt müssen wir der importierten Tabelle in unserer Datenbank auch wieder einen Namen geben. Benennen Sie diese Tabelle „t_Kunden“ und schließen Sie den Import durch Klick auf „Fertig stellen“ ab:



Wie Sie daraufhin feststellen können, wurde die Tabelle „t_Kunden“ in unsere Datenbank übernommen:

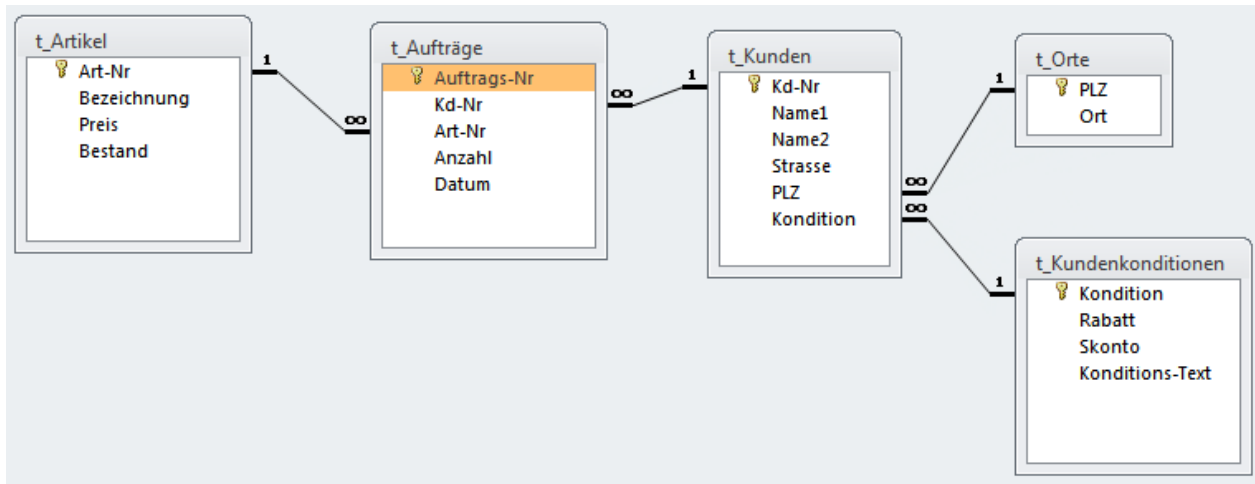


Überprüfen Sie in der Entwurfsansicht die Struktur der Tabelle sowie die Eigenschaften der Datenfelder und schauen Sie sich anschließend in der Datenblattansicht die Auftragsdaten an.

Unsere Datenbank zur Auftragserfassung im Computerladen beinhaltet nun sämtliche erforderlichen Tabellen mitsamt allen in diesen enthaltenen Daten. Im nächsten Schritt müssen wir nun noch die Verbindungen zwischen den verschiedenen Tabellen einrichten.

2.7 Beziehungen einrichten - Verbindung der Tabellen

In diesem Kapitel richten wir die diversen Beziehungen zwischen den Tabellen unserer Datenbank ein. Zur Erinnerung: Die Tabellen bzw. die für die Verbindung erforderlichen Primär- und Fremdschlüsselfelder sollen wie folgt miteinander verbunden werden:



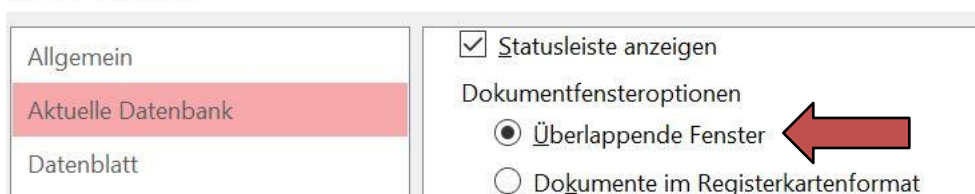
Zum Fortsetzen Ihrer Arbeit können über den folgenden Link den [Zwischenstand](#) von der letzten Übung aufrufen.

2.7.1 Angleichung der Fremdschlüssel an die Primärschlüssel

Da jedes Fremdschlüsselfeld stets auf das zugehörige Primärschlüsselfeld verweist, müssen **beide Datenfelder hinsichtlich Datentyp und Feldgröße** miteinander **übereinstimmen**, bevor wir diese miteinander verbinden können! Grundsätzlich ist dabei das **Fremdschlüsselfeld an das Primärschlüsselfeld anzugleichen**!

Widmen wir uns diesbezüglich zunächst den Tabellen „t_Aufträge“ und „t_Artikel“. Um das Fremdschlüsselfeld in der Tabelle „t_Aufträge“ an das zugehörige Primärschlüsselfeld in der Tabelle „t_Artikel“ anzugleichen, müssen wir zunächst beide Tabellen aufrufen und darin die Eigenschaften des jeweils entsprechenden Datenfelds aufrufen. Um beide Tabellen als frei positionierbare Fenster nebeneinander anzeigen zu können (anstatt als fest positionierte Register), rufen Sie bitte im **Register Datei** den Menüpunkt **Optionen > Aktuelle Datenbank** auf und stellen Sie dort unter **Dokumentfensteroptionen** die Option **Überlappende Fenster** ein und bestätigen Sie dies mit OK:

Access-Optionen



Öffnen Sie sodann die Tabellen „t_Aufträge“ sowie „t_Artikel“ jeweils in der Entwurfsansicht und positionieren Sie die beiden Tabellen nebeneinander. Wählen Sie in einer jeden der beiden Tabellen das Datenfeld „Art-Nr“ aus, so dass dessen Eigenschaften in beiden Tabellen angezeigt werden:

Feldname	Felddatentyp
Auftrags-Nr	Zahl
Kd-Nr	Zahl
Art-Nr	Kurzer Text
Anzahl	Zahl
Datum	Datum/Uhrzeit

Feldeigenschaften

Allgemein Nachschlagen

Feldgröße: 255

Feldname	Felddatentyp
Art-Nr	Zahl
Bezeichnung	Kurzer Text
Preis	Zahl
Bestand	Zahl

Feldeigenschaften

Allgemein Nachschlagen

Feldgröße: Long Integer

Da das Datenfeld „Art-Nr“ in der Tabelle „t_Artikel“ Primärschlüsselfeld ist und als Felddatentyp „Zahl“ mit der Feldgröße „Long Integer“ hat, müssen diese Eigenschaften auch für das entsprechende Fremdschlüsselfeld „Art-Nr“ in der Tabelle „t_Aufträge“ eingestellt werden. Nehmen Sie diese Einstellungen vor, um das Fremdschlüsselfeld hinsichtlich Felddatentyp und Feldgröße an das Primärschlüsselfeld anzugleichen:

Feldname	Felddatentyp
Auftrags-Nr	Zahl
Kd-Nr	Zahl
Art-Nr	Zahl
Anzahl	Zahl
Datum	Datum/Uhrzeit

Feldeigenschaften

Allgemein Nachschlagen

Feldgröße: Long Integer

Schließen Sie daraufhin die beiden Tabellen wieder und bestätigen Sie, dass die von Ihnen vorgenommenen Änderungen an der Tabelle „t_Aufträge“ gespeichert werden sollen. Sie werden dabei die folgende Meldung erhalten: **„Es kann sein, dass einige Daten verloren gehen.“** Setzen Sie den Vorgang trotzdem fort, indem Sie die Meldung mit JA bestätigen (es werden keine Daten verloren gehen, da wir lediglich den Datentyp und die Feldgröße des Fremdschlüsselfeldes an das des Primärschlüsselfeldes angeglichen haben – beide Datenfelder beinhalten jedoch die gleiche Art von Daten, so dass folglich keine Daten verloren gehen können).

Damit sind die Voraussetzungen geschaffen, um diese beiden Datenfelder miteinander verbinden zu können. Bevor wir die eigentliche Verbindung einrichten, sollten wir jedoch zuerst auch noch die Fremdschlüsselfelder an die Primärschlüsselfelder für alle übrigen vorgesehenen Beziehungen in unserer Datenbank angleichen! Überprüfen Sie daher zunächst auf die gleiche Weise systematisch die entsprechenden **Datenfeldtypen** und **Datenfeldgrößen** der Fremdschlüsselfelder

und Primärschlüsselfelder für die Beziehungen zwischen den folgenden Tabellen und nehmen Sie jeweils die ggf. erforderliche Anpassung vor (manche Fremdschlüssel stimmen bereits mit den zugehörigen Primärschlüsseln überein):

- **Kd-Nr** in Tabelle „t_Aufträge“ (Fremdschlüssel) und „t_Kunden“ (Primärschlüssel)
- **PLZ** in Tabelle „t_Kunden“ (Fremdschlüssel) und „t_Orte“ (Primärschlüssel)
- **Kondition** in Tabelle „t_Kunden“ (Fremdschlüssel) und „t_Orte“ (Primärschlüssel)

2.7.2 Beziehungen einrichten

Nach diesen Vorbereitungen stimmen für sämtliche vorgesehenen Beziehungen die jeweiligen Fremdschlüssel mit den zugehörigen Primärschlüsseln überein, so dass wir die eigentlichen Verbindungen nun herstellen können. Dies erfolgt über die Beziehungsansicht von Access. Rufen Sie diese bitte über das Register **Datenbanktools** den Befehl **Beziehungen** auf:



Daraufhin öffnet sich folgendes Fenster:



Wählen Sie hier nacheinander eine jede Tabelle aus und fügen Sie diese mittels der Schaltfläche „Hinzufügen“ der Beziehungsansicht zu. Ordnen Sie die Tabellen daraufhin wie folgt in der Beziehungsansicht an:



Um die Tabellen nun miteinander zu verbinden, müssen wir für eine jede vorgesehene Beziehung lediglich das jeweilige Fremdschlüsselfeld per Drag&Drop auf das zugehörige Primärschlüsselfeld ziehen (oder umgekehrt). Beginnen wir mit der Verbindung zwischen der Tabelle „t_Aufträge“ und der Tabelle „t_Artikel“. Ziehen Sie das Fremdschlüsselfeld „Art-Nr“ von der Tabelle „t_Aufträge“ auf das Primärschlüsselfeld „Art-Nr“ in der Tabelle „t_Artikel“ und lassen Sie die Maustaste sodann los. Daraufhin öffnet sich das folgende Fenster zur Bearbeitung der Beziehung:

Beziehungen bearbeiten

Tabelle/Abfrage: t_Artikel Verwandte Tabelle/Abfrage: t_Aufträge

Art-Nr Art-Nr

☒ Mit referentieller Integrität

☐ Aktualisierungsweitergabe an verwandte Felder

☐ Löschesweitergabe an verwandte Datensätze

Beziehungstyp: 1:n

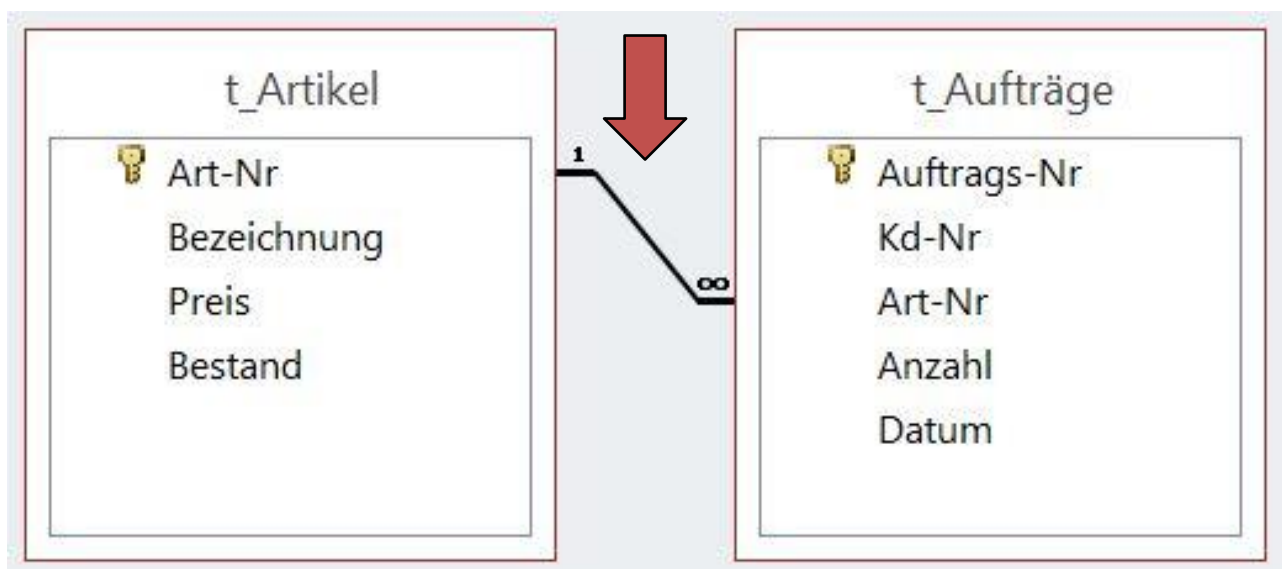
Erstellen

Abbrechen

Verknüpfungstyp...

Neue erstellen...

Wie ersichtlich ist, werden zur Bestätigung nochmals aus beiden Tabellen die jeweiligen Datenfelder angezeigt, die wir miteinander verbinden wollen. Um eine funktionelle Verbindung herzustellen, ist es jedoch wichtig, die Option „Mit referentieller Integrität“ zu aktivieren (siehe obige Abbildung). Bestätigen Sie die Einstellungen durch Klick auf die Schaltfläche „Erstellen“. Die Verbindung zwischen den beiden Datenfeldern wird daraufhin eingerichtet:



Beachten Sie die beiden Bezeichner an einem jeweiligen Ende der Beziehungslinie. Das Linienende am Primärschlüsselfeld der Tabelle „t_Artikel“ trägt als Bezeichner eine 1, was bedeutet, dass ein jeder Datenfeldinhalt (also eine jede Artikel-Nummer) in dieser Tabelle jeweils nur einmal

vorkommen darf. Duplikate sind hier nicht erlaubt/möglich, da es ein Primärschlüsselfeld ist. Das Linienfeld am Fremdschlüsselfeld aus der Tabelle „t_Aufträge“ hingegen, trägt als Bezeichner ein Unendlichkeits-Zeichen, was bedeutet, dass ein bestimmter Datenfeldinhalt hier beliebig oft vorkommen kann. Da ein bestimmter Artikel im Rahmen verschiedener Aufträge ja des Öfteren bestellt werden kann und die jeweilige Artikel-Nr somit in mehreren Aufträgen vorkommen kann.

Die beiden Bezeichner „1“ zu „Unendlich“ geben dabei gleichsam zu erkennen, dass es sich bei dieser Beziehung um eine so genannte **1 : n Beziehung** handelt (siehe Abbildung unten) – sprich: In einer der beiden Tabellen (dort wo die Art-Nr. Primärschlüssel ist) kann eine bestimmte Artikel-Nr. nur einmal vorkommen – in der anderen Tabelle hingegen (dort wo die Art-Nr. Fremdschlüssel ist) beliebig oft.

Gehen wir nun noch auf die Bedeutung der „**referentiellen Integrität**“ ein: Diese Option ist dafür zuständig, dass Datensätze, auf dessen Primärschlüssel von einer verknüpften Tabelle aus verwiesen wird, nicht einfach gelöscht oder hinsichtlich des Primärschlüsselinhalts geändert werden können. Es wird durch die referentielle Integrität also sichergestellt, dass ein entsprechender **Querverweis** als solcher auch **gültig bleibt**. Hierzu ein Beispiel: Gehen wir davon aus, dass in einem Auftrag z. Bsp. die Artikel-Nr. 1201 bestellt wurde und der entsprechende Auftragsdatensatz auf diesen Artikel verweist. Würde man den Artikel später aus der Artikeldatenbank löschen (da der Artikel ggf. aus dem Sortiment genommen wurde), so würde der entsprechende Querverweis aus dem Auftragsdatensatz ins Leere laufen, da der Artikel nicht mehr existiert. Gleiches würde geschehen, wenn man die entsprechende Artikel-Nr. in der Tabelle „t_Artikel“ einfach in eine andere Nummer abändern würde, so dass die Artikel-Nr. aus dem Auftragsdatensatz nicht mehr gefunden würde. Die referenzielle Integrität stellt sicher, dass dies nicht geschehen kann, da Löschungen oder Änderungen von Primärschlüsselfeldinhalten durch diese unterbunden werden.

Falls man die Option vorbehalten möchte, dass Primärschlüsselfeldinhalte trotzdem geändert werden können – oder entsprechende Datensätze gelöscht werden können, so kann man dies durch die Zusatzoptionen „Aktualisierungsweitergabe“ bzw. „Löschweitergabe“ einrichten:

Neue erstellen...

☒ Mit referentieller Integrität

☐ Aktualisierungsweitergabe an verwandte Felder

☐ Löschweitergabe an verwandte Datensätze

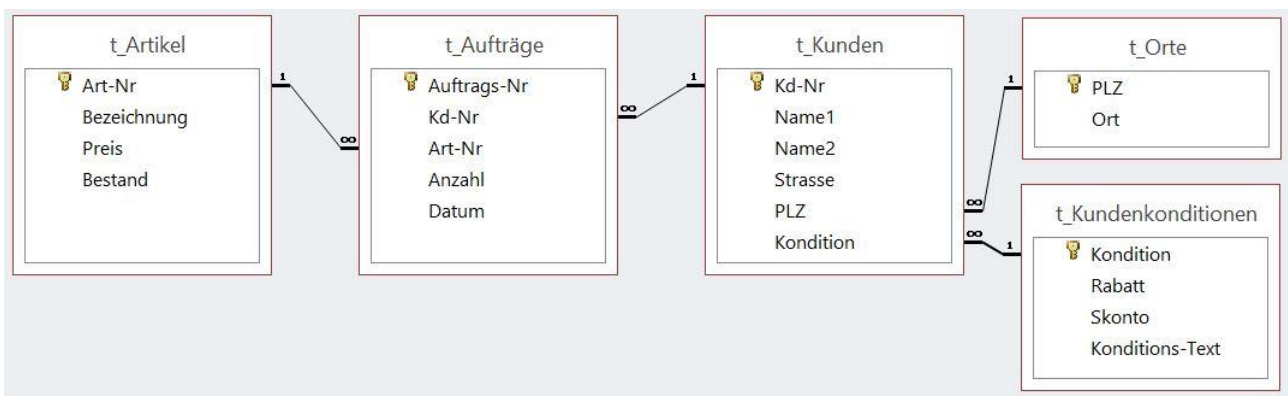
Beziehungstyp: 1:n

Die Option **Aktualisierungsweitergabe** sorgt dafür, dass bei **Änderung eines Primärschlüsselfeldinhalts** der entsprechende Wert auch in **allen Fremdschlüsselfeldern** der darauf verweisenden Datensätze **mitgeändert** wird, so dass die entsprechenden Querverweise gültig bleiben.

Die Option **Löschweitergabe** hingegen sorgt dafür, dass bei **Löschung eines Datensatzes mit einem Primärschlüsselinhalt**, auf welchen durch einen Fremdschlüssel im Datensatz einer anderen Tabelle verwiesen wird, **auch der darauf verweisende Datensatz** in der betreffenden Tabelle **gelöscht** wird. Diese Option sollte mit äußerster Vorsicht verwendet werden, da die Löschweitergabe zum unbeabsichtigten Verlust von ggf. noch benötigten Daten führen kann!

Nach Erläuterung der verschiedenen Beziehungsoptionen, richten Sie nun bitte noch die folgenden Verbindungen zwischen den übrigen Tabellen ein, so dass Ihre Datenbank letztendlich wie folgt in der Beziehungsansicht dargestellt wird:

- **Kd-Nr** von Tabellen „t_Aufträge“ (Fremdschlüssel) und „t_Kunden“ (Primärschlüssel)
- **PLZ** von Tabellen „t_Kunden“ (Fremdschlüssel) und „t_Orte“ (Primärschlüssel)
- **Kondition** von Tabelle „t_Kunden“ (Fremdschlüssel) und „t_Kundenkonditionen“ (Primärschlüssel)



Damit wären nun sämtliche Beziehungen eingerichtet, womit unsere Datenbank nun vollständig funktionell ist und für Informationsabfragen verwendet werden kann. Sie können das Fenster mit der Beziehungsansicht nun schließen und die Speicherung der darin vorgenommenen Änderungen bestätigen.

2.7.3 Nachschlagefunktionen einrichten

Rufen Sie nun die Tabelle „t_Aufträge“ in der Datenblattansicht auf. Wir wollen hier nun einen neuen Auftrag eingeben. Der Rechtsanwalt Klocke hat am 24.11.2016 ein Tower-Gehäuse bestellt, was als entsprechender Auftrag in der Tabelle „t_Aufträge“ zu erfassen ist:

Auftrags-Nr	Kd-Nr	Art-Nr	Anzahl	Datum	Zum H
14	122	5011	1	06.09.2016	
15	591	8001	1	10.09.2016	
16		0	1	24.11.2016	

Das Problem: Wir kennen zwar den Namen des Kunden und die Bezeichnung des Artikels – jedoch nicht die zugehörige Kd-Nr und Art-Nr, welche in der Auftragsstabelle als Querverweise in die entsprechenden Fremdschlüsselfelder einzutragen sind. Wir müssten faktisch die Tabellen

„t_Kunden“ und „t_Artikel“ aufrufen, um die betreffenden Nummern zu dem Kunden und dem bestellten Artikel aus diesen heraus zu suchen. Im Sinne eines höheren Anwendungskomforts ist es daher sinnvoll, für die betreffenden Fremdschlüsselfelder in der Tabelle „t_Aufträge“ jeweils eine entsprechende Nachschlagefunktion einzurichten, die uns aus der Tabelle „t_Aufträge“ heraus einen direkten Zugriff auf die Inhalte der beiden Tabellen „t_Kunden“ und „t_Artikel“ ermöglicht.

Fangen wir mit dem Datenfeld „Kd-Nr“ an: Wechseln Sie hierzu in die Entwurfsansicht der Tabelle „t_Aufträge“ und klicken Sie dort das Datenfeld „Kd-Nr“ an. Rufen Sie unten im Bereich der Feldeigenschaften das Register „Nachschlagen“ auf und wählen Sie aus dem DropDown-Menü die Option „Kombinationsfeld“ aus:



Daraufhin können Sie eine Reihe von Einstellungsoptionen für das Kombinationsfeld eingeben. Befüllen Sie diese wie folgt:

Allgemein	Nachschlagen
Steuerelement anzeigen	Kombinationsfeld
Herkunftstyp	Tabelle/Abfrage
Datensatzherkunft	t_Kunden
Gebundene Spalte	1
Spaltenanzahl	3
Spaltenüberschriften	Ja
Spaltenbreiten	1cm;4cm;4cm
Zeilenanzahl	16
Listenbreite	9cm

Zur Erläuterung:

Der **Herkunftstyp** „**Tabelle/Abfrage**“ sagt aus, dass die Nachschlagefunktion auf eine Tabelle oder Abfrage zugreift, um den Wert für die gewünschte Kd-Nr aus dieser zu übernehmen.

Die **Datensatzherkunft** „**t_Kunden**“ legt fest, dass dabei auf die Tabelle „t_Kunden“ zugegriffen wird.

Die **Gebundene Spalte** gibt an, **aus welcher Spalte der Tabelle der Wert übernommen wird**. Die Kd-Nr befindet sich in der Tabelle „t_Kunden“ in der 1. Spalte – daher ist dies auch die gebundene Spalte – sprich: Wir wollen den Wert aus der **Spalte Nr. 1** der Tabelle „t_Kunden“ übernehmen.

Die **Spaltenanzahl** legt fest, wie viele Spalten der Tabelle „t_Kunden“ von der Nachschlagefunktion angezeigt werden sollen. Name 1 und Name 2 der jeweiligen Kunden befinden sich in der 2. und 3. Spalte der Tabelle „t_Kunden“. Um diese beiden Informationen mit angezeigt zu bekommen, reichen daher die ersten **3** Spalten aus.

Da es auch beim Nachschlagen immer sinnvoll ist, dass auch die **Spaltenüberschriften** angezeigt werden, ist diese Option mit **Ja** versehen.

Der Parameter **Spaltenbreiten** gibt an, **wie breit die einzelnen Spalten** in der Nachschlagefunktion **dargestellt werden sollen**. Da in der **ersten Spalte** nur die **Kd-Nr** angezeigt wird, reicht für diese eine Breite von **1cm** aus. Für die Darstellung von **Name 1** und **Name 2** hingegen wird jeweils eine größere Breite erforderlich sein, weshalb für die **zweite und dritte Spalte** eine Breite von **jeweils 4cm** sinnvoll ist. Die Angaben der einzelnen Spaltenbreiten sind jeweils durch Semikola voneinander zu trennen.

Da wir mit den **Breiten der einzelnen Spalten in der Summe** auf **9 cm** kommen, sollte auch die **Listenbreite** auf diesen Wert eingestellt werden, so dass in der Nachschlagefunktion auch alle drei Spalten vollständig sichtbar sind und durch die Listenbreite der Nachschlagefunktion abgedeckt werden.

Die **Zeilenanzahl** gibt an, **wie viele Datenzeilen aus der verknüpften Tabelle** durch die Nachschlagefunktion angezeigt werden sollen. In unserem Beispiel sind es 16, wobei in der Nachschlagefunktion natürlich bei Bedarf auch zu tiefer gelegenen Zeilen gescrollt werden kann, sofern sich der gesuchte Datensatz nicht unter den ersten 16 Zeilen befindet.

Wechseln Sie nach diesen Einstellungen wieder in die Datenblattansicht der Tabelle „t_Aufträge“ und bestätigen Sie, dass Ihre Änderungen am Tabellenentwurf gespeichert werden sollen.

Klicken Sie dort nun im Datenfeld „Kd-Nr“ in der Zeile zur Eingabe eines neuen Datensatzes auf das DropDown-Feld am rechten Rand des betreffenden Datenfeldes. Wie Sie sehen, werden Ihnen daraufhin die ersten drei Spalten der Kunden-Tabelle (Kd-Nr, Name 1 und Name 2) angezeigt, so dass Sie aus der betreffenden Listen direkt den Rechtsanwalt Klocke auswählen können:

t_Aufträge					
Auftrags-Nr	Kd-Nr	Art-Nr	Anzahl	Datum	
14	122	5011	1	01.01.2023	
15	591	8001	1	15.02.2023	
*			0		
	Kd-Nr	Name1	Name2		
	121	Heilpraktiker	Anton van t'Hof		
	122	Rechtsanwalt	Klocke		
	169	Tanzschule	Pellinger		

Bevor Sie den Rechtsanwalt Klocke jedoch anklicken, um den Eintrag vorzunehmen, wechseln Sie zuvor nochmals in die Entwurfsansicht der Tabelle „t_Aufträge“ und richten Sie in dieser weiterhin noch auf die gleiche Weise eine Nachschlagefunktion für das Fremdschlüsselfeld „Art-Nr“ auf die Tabelle „t_Artikel“ ein. Hier genügt die Anzeige der ersten zwei Spalten – nämlich der Art-Nr und der Artikelbezeichnung. Für die Spalte Art-Nr genügt eine Breite von 1cm – für die Artikelbezeichnung sollten es 5cm sein.

Danach können Sie wieder in die Datenblattansicht wechseln und den Auftrag von Rechtsanwalt Glocke mit komfortabler Nachschlagefunktion auf die Kundenliste und Artikelliste eingeben:

t_Aufträge

Auftrags-Nr ▾	Kd-Nr ▾	Art-Nr ▾	Anzahl ▾	Datum ▾	Zum l
15	591	8001	1	10.09.2016	
16	122	▾	1	24.11.2016	
*		<div> <div>Artike</div> <div>Bezeichnung</div> <div>2002 Midi-Tower-Gehäuse</div> <div>2003 Tower-Gehäuse</div> <div>2004 Luxus-Tower-Gehäuse</div> </div>			

Sofern die Zeit noch reicht, können Sie nun auch noch eine Nachschlagefunktion für das Fremdschlüsselfeld „PLZ“ in der Tabelle „t_Kunden“ auf die Tabelle „t_Orte“ einrichten – sowie eine Nachschlagefunktion für das Fremdschlüsselfeld „Kondition“ in der Tabelle „t_Kunden“ auf die Tabelle „t_Kundenkonditionen“. Damit wäre unsere Datenbank sodann vollendet.

3. Das Entity-Relationship Modell (ERM)

Die Datenbank zur Erfassung von Kundenaufträgen in einem Computerladen haben Sie soweit nach entsprechenden Vorgaben in Access erstellt. Bevor man jedoch eine Datenbank in einer entsprechenden Anwendungssoftware wie Access erstellt, ist es im Allgemeinen erforderlich, für die jeweilige Datenbank zuvor ein diesbezügliches **Modell** bzw. einen Konstruktionsplan entwickelt zu haben – so wie für den Bau eines Hauses zuvor auch ein entsprechender Bauplan entworfen wird. Für den Entwurf bzw. die **Modellierung von Datenbanken** verwendet man dabei das so genannte **Entity-Relationship-Modell**, welches auch kurz als **ERM** bezeichnet wird.

Arbeitsauftrag: Erstellen Sie für die Datenbank zur Auftragserfassung in einem Computerladen das zugehörige Entity-Relationship-Modell, indem Sie

1. sich zunächst mit den folgenden Informationen zum ERM auseinander setzen und sodann
2. die Aufgabe 1 auf Seite 53 bearbeiten.

3.1 Bausteine und Regeln des ERM

Das **Entity-Relationship-Modell (ERM)** ist eine **grafische Darstellung der Beziehungen zwischen verschiedenen Objekten der Wirklichkeit** bzw. **zwischen den entsprechenden Tabellen einer Datenbank** (siehe Beispiel auf der folgenden Seite).

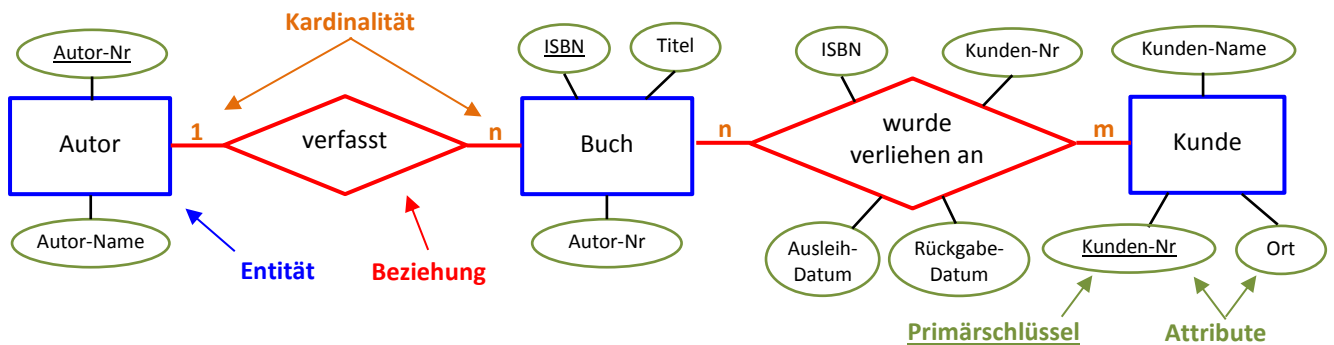
Eine **Entity** bzw. **Entität** stellt einen **Repräsentant für die Dinge** (Objekte, Gegenstände) **der Wirklichkeit** dar (z. B.: Buch, Autor, Verlag). In Datenbanken werden Entitäten grundsätzlich durch **Tabellen** dargestellt – im ERM hingegen als **Rechteck**.

Attribute beschreiben jeweils **bestimmte Eigenschaften einer Entität** (z. Bsp. Name des Autors, Titel eines Buches, ISBN). In einer Datenbank werden Attribute durch die verschiedenen **Datenfelder** einer Tabelle repräsentiert – im ERM hingegen als **Oval**, welches per Verbindungslinie an die jeweilige Entität angehängt wird. Dasjenige Datenfeld/Attribut, welches den **Primärschlüssel** bildet, wird im ERM zudem **unterstrichen** dargestellt.

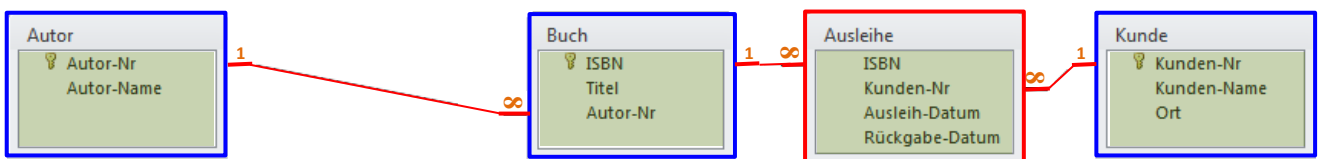
Unter einer **Relationship** (= Beziehung) verstehen wir den **inhaltlichen Zusammenhang zwischen zwei Objekten** bzw. Tabellen (z. B.: „ist der Verfasser“ als Beziehung zwischen Autor und Buch). In der Beziehungsansicht von Access werden Beziehungen durch entsprechende **Verbindungen zwischen den Tabellen** dargestellt – im ERM hingegen als **Raute**, welche die betreffenden Entitäten über entsprechende Linien miteinander verbindet.

Unter der **Kardinalität** (auch: Konnektivität) versteht man den **Beziehungstyp zwischen zwei Objekten** bzw. Tabellen (z. B.: ein Autor kann mehrere Bücher verfassen, doch ein bestimmtes Buch kann stets nur von einem Autor verfasst worden sein). Die Kardinalität wird in der Beziehungsansicht von Access wie auch im ERM durch eine entsprechende **Informationsangabe an der Beziehung** notiert (z. Bsp. 1:n oder n:m).

Die nachfolgende Darstellung zeigt das **Beispiel eines Entity-Relationship-Modells** für die Datenbank einer Bücherei:



In der **Beziehungsansicht von Access** würde die entsprechende Datenbank wie folgt aussehen:

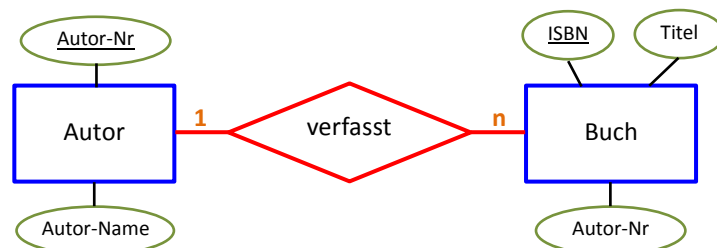


Diese Darstellung bedarf noch ergänzender Erläuterungen zu den verschiedenen Beziehungstypen:

3.2 Beziehungstypen (Kardinalität)

3.2.1 Die 1 : n - Beziehung

Die **1 : n - Beziehung** ist der klassische und gleichsam am häufigsten vorkommende Beziehungstyp:



1 : n bedeutet in diesem Fall:

- Ein **Buch** kann immer nur von **einem Autor (1)** verfasst worden sein
- Ein **Autor** kann jedoch **mehrere Bücher (n)** verfasst haben

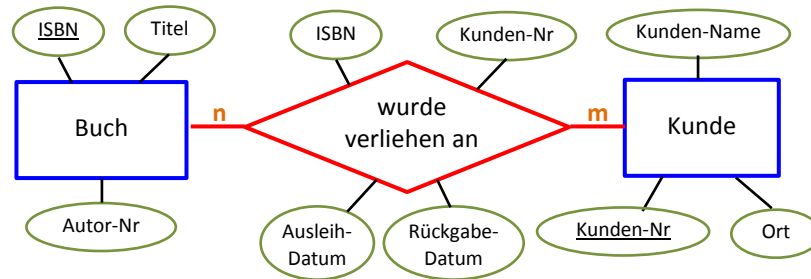
Die betreffende Kardinalität (1 bzw. n) wird jeweils an der gegenüber liegenden Entität notiert.

In der Tabellenansicht wird gleichsam deutlich, dass z. Bsp. der Autor M. Goldratt zwei Bücher verfasst hat, ein jedes Buch aber stets nur einem Autor zugeordnet werden kann:

Tabelle „Autor“		Tabelle „Buch“		
Autor-Nr	Autor-Name	ISBN	Titel	Autor-Nr
1	M. Goldratt	978-3593367019	Das Ziel	1
2	Dee Jacob	978-3593369136	Das Ergebnis	1
3	Gene Kim	978-3593391038	Velocity	2
		978-3958751750	Projekt Phoenix	3

3.2.2 Die **n : m** - Beziehung

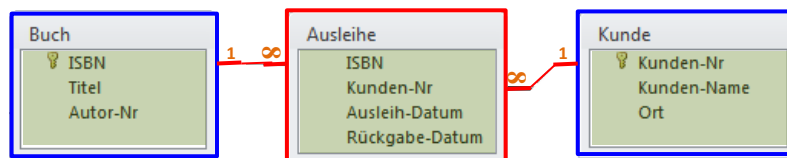
Die **n : m** - **Beziehung** ist schon etwas spezieller und stellt sich im **ERM** wie folgt dar:



n : m bedeutet in diesem Fall:

- Ein **Kunde** kann **mehrere Bücher (n)** ausgeliehen haben
- Ein **Buch** kann im Laufe der Zeit auch an **mehrere Kunden (m)** verliehen worden sein

In Access kann eine **n : m** - **Beziehung** nur **indirekt** über eine **dritte Tabelle** eingerichtet werden, welche jeweils in einer **1 : n** - **Beziehung** mit den beiden zugehörigen Tabellen steht:



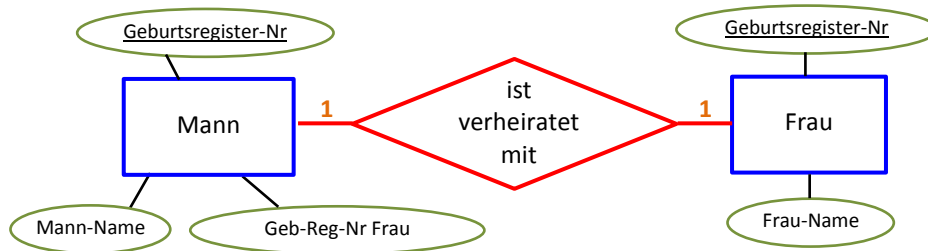
In der Tabellenansicht wird gleichsam deutlich, dass z. Bsp. das Buch „Das Ziel“ von zwei verschiedenen Kunden ausgeliehen wurde, gleichsam die Kundin „Sabine Fröhlich“ auch zwei verschiedene Bücher ausgeliehen hat. Die Tabelle „Buch“ und die Tabelle „Kunden“ stehen dabei jeweils in einer 1 : n - Beziehung mit der Tabelle „Ausleihe“:

Tabelle „Buch“			Tabelle „Ausleihe“			Tabelle „Kunde“		
ISBN	Titel	Au	ISBN	Kunden-Nr	Ausleih-Dat	Kunden-Nr	Kunden-Name	C
978-3593367019	Das Ziel		978-3958751750	1	28.09	1	Manfred Fuchs	Her
978-3593369136	Das Ergebnis		978-3593367019	2	22.09	2	Sabine Fröhlich	Kirn
978-3593391038	Velocity		978-3593369136	2	05.10	3	Hannes Kuhn	Her
978-3958751750	Projekt Phoenix		978-3593367019	3	03.10	4	Franz Maier	Fisc

Da die **n : m** - **Beziehung** nur über eine dritte **Zuordnungstabelle** eingerichtet werden kann, welche ihrerseits wiederum verschiedene **Datenfelder** beinhaltet, sind diese **im ERM als entsprechende Attribute an der n : m - Beziehung anzuhängen** (siehe obiges Beispiel).

3.2.3 Die 1 : 1 - Beziehung

Die **1 : 1 - Beziehung** ist ein relativ selten vorkommender Beziehungstyp. Ein entsprechendes Beispiel wäre z. Bsp. in einer Datenbank über geschlossene Ehestände gegeben:

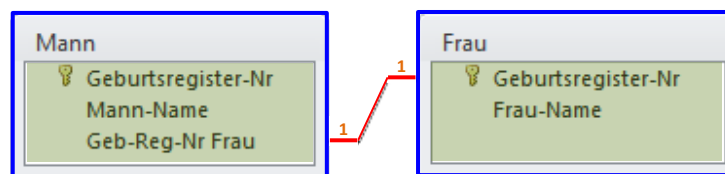


1 : 1 bedeutet in diesem Fall:

- Ein **Mann** ist grundsätzlich mit **einer Frau (1)** verheiratet
- Eine **Frau** ist gleichsam immer mit **einem Mann (1)** verheiratet

Einem Datensatz einer Tabelle ist dabei jeweils genau ein Datensatz einer anderen Tabelle zugeordnet.

In Access würde sich diese **1 : 1 - Beziehung** wie folgt darstellen:



In der Tabellenansicht wird gleichsam deutlich, dass jedem Mann exakt eine Frau (und umgekehrt) zugeordnet ist:

Tabelle „Mann“			Tabelle „Frau“	
Geburtsregister-Nr	Mann-Name	Geb-Reg-Nr Frau	Geburtsregister-Nr	Frau-Name
1245/1984	Friedrich Klein	253/1987	253/1987	Julia Fischer
1246/1984	Stefan Koch	251/1987	251/1987	Sabine Kurz
1248/1984	Achim Groß	1421/1990	1421/1990	Rosi Fröhlich

Anmerkung: Um eine **1 : 1 - Beziehung** in Access einrichten zu können, muss das entsprechende **Fremdschlüsselfeld**, über welches die Beziehung zwischen den beiden Tabellen hergestellt wird (in obigem Beispiel das Datenfeld „Geb-Reg-Nr. Frau“ aus der Tabelle „Mann“) **indiziert** werden, wobei **keine Duplikate** zulässig sind, da für eine 1 : 1 - Beziehung auch in dem querverweisenden Fremdschlüsselfeld ein jeder Datenfeldinhalt nur einmal vorkommen darf:

Indiziert ☒ Ja (Ohne Duplikate) ☒

Beim Einrichten der Beziehung in der Beziehungsansicht von Access, wird im entsprechenden Dialogfenster als Beziehungstyp sodann der Typ 1 : 1 angezeigt:

Beziehungstyp: 1:1

3.3 Aufgaben

Aufgabe 1: Erstellen Sie auf Basis der erfolgten Informationsdarstellung nun das entsprechende Entity-Relationship-Modell für die Datenbank zur Auftragserfassung in einem Computerladen, indem Sie

1. in Access die Datenbank [Computerladen.accdb](#) in der Beziehungsansicht öffnen und sodann
2. die Datenbank **in Form des Entity-Relationship-Modells** auf einem separaten Blatt **skizzieren**.

Aufgabe 2: Geben Sie zu den folgenden Sachverhalten den jeweiligen **Beziehungstyp** an:

- Mitarbeiter „arbeitet in“ Projekt.
- Mitarbeiter „ist Abteilungsleiter von“ Abteilung
- Mitarbeiter „gehört zu“ Abteilungen.

Aufgabe 3: Die Firma „Bugs“ fertigt verschiedene Geräte. Für die betriebliche Organisation dieser Firma soll eine relationale Datenbank eingesetzt werden. Dabei gilt folgendes:

Jedes Bauteil, das verwendet wird, hat eine eindeutige Nummer und eine Bezeichnung, die allerdings für mehrere verschiedene Bauteile gleich sein kann. Von jedem Teil werden außerdem der Name des Herstellers, der Einkaufspreis pro Stück und der am Lager vorhandene Vorrat gespeichert. Jedes herzustellende Gerät hat eine eindeutige Bezeichnung. Auch von jedem schon gefertigten Gerätetyp soll der aktuelle Lagerbestand gespeichert werden, ebenso wie der Verkaufspreis des Gerätes. In unserem fiktiven Betrieb gilt die Regelung, dass Maschinen, die mehr als 1000,- EUR kosten, versandkostenfrei an die Kunden ausgeliefert werden; für Geräte, die weniger kosten, wird zusätzlich zum Preis eine gerätespezifische Anliefergebühr berechnet. In der Datenbank ist ebenfalls zu speichern, welche Bauteile für welche Geräte benötigt werden. Es gibt Bauteile, die für mehrere Geräte verwendet werden. Von jedem Kunden werden der Name, die Adresse und die Branche gespeichert. Es kann verschiedene Kunden mit demselben Namen oder derselben Adresse geben, weshalb jeder Kunde auch eine eindeutige Kundennummer erhält. Außerdem ist zu jedem Kunden vermerkt, wer aus unserer Firma für die entsprechende Kundenbetreuung zuständig ist. Weiterhin ist zu speichern, welche Kunden mit welchen Geräten beliefert wurden. Es kann sein, dass gewissen Kunden für bestimmte Geräte Sonderkonditionen eingeräumt worden sind, dies soll ebenfalls in der Datenbank vermerkt werden.

- a) Entwickeln Sie für diese Anforderungen das entsprechende **Entity-Relationship-Modell**.
- b) Erstellen Sie auf Basis des ERM daraufhin die entsprechende **Datenbank in Access**.

Aufgabe 4: Für die Verwaltung einer Schule soll eine relationale Datenbank entwickelt werden:

In der Schule arbeiten verschiedene Lehrer, wobei jeder Lehrer eine eindeutige Personalnummer hat. Zudem soll in der Datenbank sein Nachname, sein Vorname, seine Adresse und sein Wohnort gespeichert werden.

Die Schule wird von vielen Schülern besucht. Zur eindeutigen Identifikation wird jedem Schüler eine Schülernummer zugeteilt. Außerdem soll zu jedem Schüler der Nachname, der Vorname und der Wohnort erfasst werden.

Die Schüler der Schule sind in Klassen eingeteilt, wobei jede Klasse eine eindeutige Klassenbezeichnung hat. Ein Schüler kann immer nur einer Klasse zugehörig sein. Eine jede Klasse besteht aber aus mehreren Schülern.

Die Schule verfügt über verschiedene Räume mit einer jeweils eindeutigen Raumnummer. Jeder Klasse wird exakt ein Raum als Klassenraum zugeteilt. Umgekehrt ist auch ein jeder Klassenraum stets nur einer bestimmten Klasse zugeteilt.

Jeder Lehrer unterrichtet i.d.R. mehrere Klassen, wobei eine Klasse in den verschiedenen Fächern auch jeweils von mehreren Lehrern unterrichtet wird. In der Datenbank soll dabei gleichsam erfasst werden, welcher Lehrer in welcher Klasse welches Fach unterrichtet.

- a) Entwickeln Sie für diese Anforderungen das entsprechende Entity-Relationship-Modell.
- b) Erstellen Sie auf Basis des ERM daraufhin die entsprechende **Datenbank in Access**.

4. Datenbanken-Abfragen (Queries)

4.1 Sinn und Zweck von Datenbankabfragen

Datenbankabfragen – so genannte **Queries** – ermöglichen es uns, **zielgerichtet bestimmte Informationen aus unserer Datenbank heraus zu lesen**. Eine Datenbank enthält üblicherweise eine Fülle von Daten. Mit Hilfe einer Abfrage stellen Sie Fragen zu den gespeicherten Daten. Dies geschieht am häufigsten mit Hilfe einer **Auswahlabfrage**. Damit können Sie **Daten** aus einer oder mehreren Tabellen **anzeigen, analysieren** und auch **ändern**. Die Daten, die Access Ihnen dann anzeigt, werden in einem **Dynaset** gesammelt. Dieser Dynaset hat zwar das **Aussehen einer Tabelle**, besteht aber **nur** aus einer **dynamischen Ansicht von Daten**. Ein Dynaset wird immer auf der Basis des aktuellen Datenbestandes erstellt. Nehmen Sie innerhalb einer Abfrage Veränderungen an den Dateninhalten vor, so wirken sich diese Änderungen auch auf die der Abfrage zugrundeliegenden Tabellen aus.

Abfragen dienen in Access folgenden **Zwecken**:

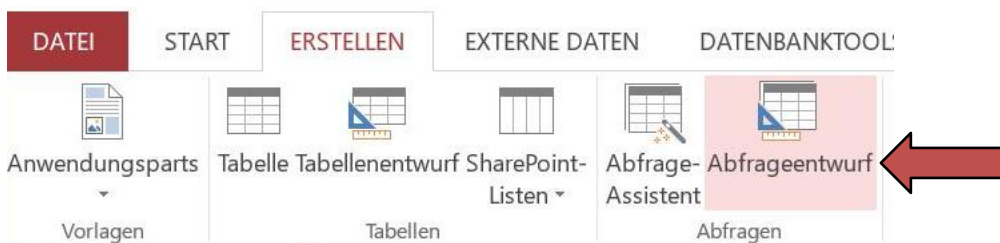
- **Felder auswählen**
Sie können sich diejenigen Felder ansehen, deren Inhalt Für Sie gerade interessant ist. Die restlichen Felder werden ausgeblendet.
- **Datensätze wählen**
Sie geben die Kriterien an, denen die Datensätze entsprechen sollen, und ACCESS erstellt ein Dynaset dazu.
- **Datensätze sortieren**
Sie können Datensätze in einer bestimmten Reihenfolge nach verschiedenen Feldern sortieren. Wenn Datensätze in einem Formular sortiert erscheinen sollen, wird zunächst eine Abfrage erstellt, die dann als Datenquelle für das Formular dient.
- **Fragen zu Daten in mehreren Tabellen stellen**
Anhand einer Abfrage können Daten aus mehreren Tabellen abgerufen und auf einem einzelnen Datenblatt dargestellt werden.
- **Berechnungen durchführen**
Auf der Basis der vorhandenen Daten können Sie so genannte berechnete Felder erstellen.
- **Abfragen als Datenquelle für Formulare, Berichte und andere Abfragen verwenden**
Sollen in einem Formular oder einem Bericht bestimmte Informationen erscheinen, so erstellen Sie zunächst eine Abfrage. Diese Abfrage dient dann wiederum als Datenquelle für das Formular oder den Bericht.
- **Daten in Tabellen ändern / löschen**
Mittels der Aktualisierungsabfrage sind Sie in der Lage, Gruppen von Datensätzen gleichzeitig zu aktualisieren, zu löschen oder anzufügen. Sie können aber auch eine vollständig neue Tabelle auf der Basis der Abfrage erstellen.

4.2 Einfache Abfragen erstellen

Im Folgenden wollen wir zu unserer bereits bestehenden Datenbank eine paar einfache Auswahlabfragen stellen, die zunächst einmal zwecks besseren Überblicks lediglich eine reduzierte Sicht auf die darin enthaltenen Daten bietet – d.h. wir wollen aus bestimmten Tabellen lediglich bestimmte Datenfelder (Spalten) auswählen, die gerade interessant für uns sind, während die übrigen Datenfelder der betreffenden Tabellen in der Abfrage ausgeblendet werden. Öffnen Sie hierzu in Access den letzten Stand der Datenbankdatei [Computerladen.accdb](#).

Beispielaufgabe: Es soll zunächst eine **nach dem Namen alphabetisch geordnete Liste der Kunden** erstellt werden. In dieser Liste sollen **nur** die Felder **KD-Nr, Name1, Name2** sichtbar sein.

1. Rufen Sie hierzu in Access über das Register **Erstellen** den Befehl **Abfrageentwurf** auf:



2. Rufen Sie diese bitte über das Register **Datenbanktools** den Befehl **Beziehungen** auf

In Access erscheint daraufhin ein noch leerer **Abfrageentwurf** sowie in dessen Vordergrund ein Dialogfenster namens „Tabelle Anzeigen“, mit dessen Hilfe Sie festlegen können, aus welchen Tabellen die Daten der Abfrage stammen sollen. In dem Fenster werden alle zur Datenbank gehörigen Tabellen aufgelistet:

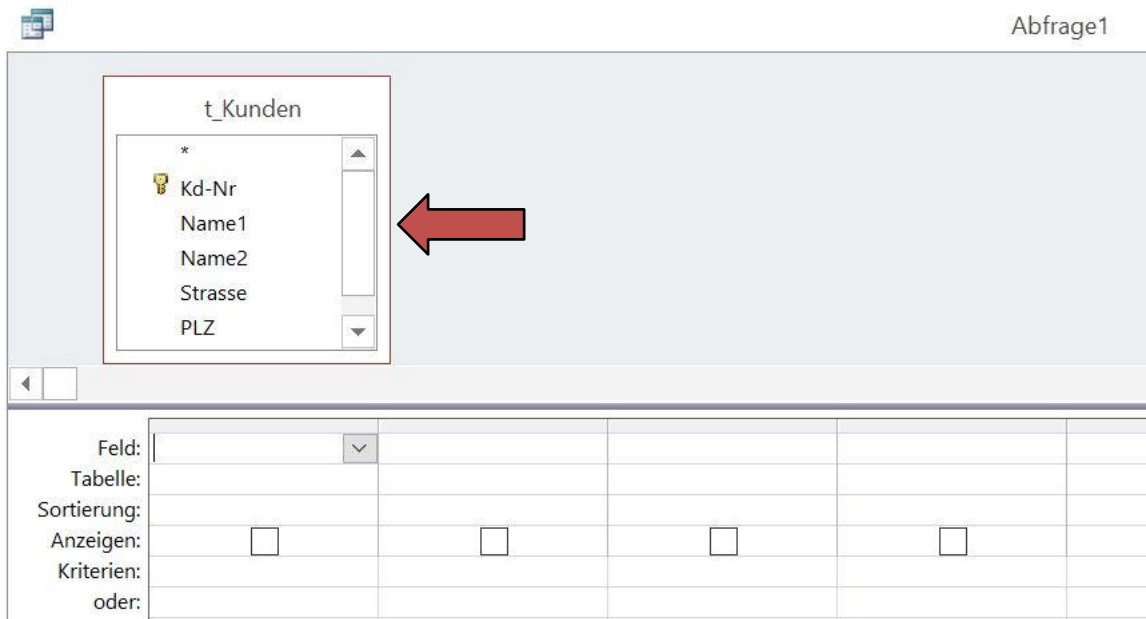


Um der Abfrage eine Tabelle hinzuzufügen, müssen Sie lediglich einen Doppelklick auf den betreffenden Tabellennamen ausführen oder alternativ den Tabellennamen markieren und sodann auf „Hinzufügen“ klicken. Sollen mehrere Tabellen als Datenbasis für die Abfrage dienen, so ist dieser Schritt für alle benötigten Tabellen zu wiederholen.

Gemäß obiger Aufgabenstellung wollen wir eine sortierte Kundenliste erstellen, in welcher **nur** die Felder **KD-Nr, Name1, Name2** sichtbar sein sollen. Da die entsprechenden Daten aus der Tabelle **t_Kunden** stammen, muss diese Tabelle somit der Abfrage hinzugefügt werden:

3. **Doppelklicken** Sie also auf die Tabelle **t_Kunden** und schließen Sie danach das Dialogfenster.

In der oberen Hälfte des Abfrageentwurfs wird daraufhin die ausgewählte Tabelle mitsamt einer Liste aller darin enthaltenen Datenfelder angezeigt:



In der unteren Hälfte des Abfrageentwurfs ist der so genannte **QBE-Bereich** (Query by Example = Abfrage nach Beispiel) sichtbar. Mit Hilfe dieses Tools können Abfragen auf die im oberen Bereich angezeigten Tabellen erstellt werden, indem die für die Abfragen erforderlichen Datenfelder in den QBE-Entwurfsbereich gezogen werden. Dazu ist der Mauszeiger auf einen Feldnamen in der Feldliste zu setzen und bei gedrückter linker Maustaste in die Zeile „Feld“ einer Spalte des QBE-Bereiches zu ziehen. Alternativ können die gewünschten Datenfelder auch einfach durch Doppelklick auf den entsprechenden Datenfeldnamen in den QBE-Bereich gebracht werden oder mittels des Drop-Down Menüs in der Zeile „Feld“ für die betreffende Spalte des QBE-Bereichs ausgewählt werden.

Die Felder erscheinen später in der Datenblattansicht in der gleichen Reihenfolge wie der des QBE-Bereiches.

Fügen wir nun also die drei Datenfelder **Kd-Nr**, **Name1** und **Name2** aus der Tabelle t_Kunden in den QBE-Bereich des Abfrageentwurfs ein. Gehen Sie hierzu wie folgt vor, um alle 3 Möglichkeiten exerziert zu haben:

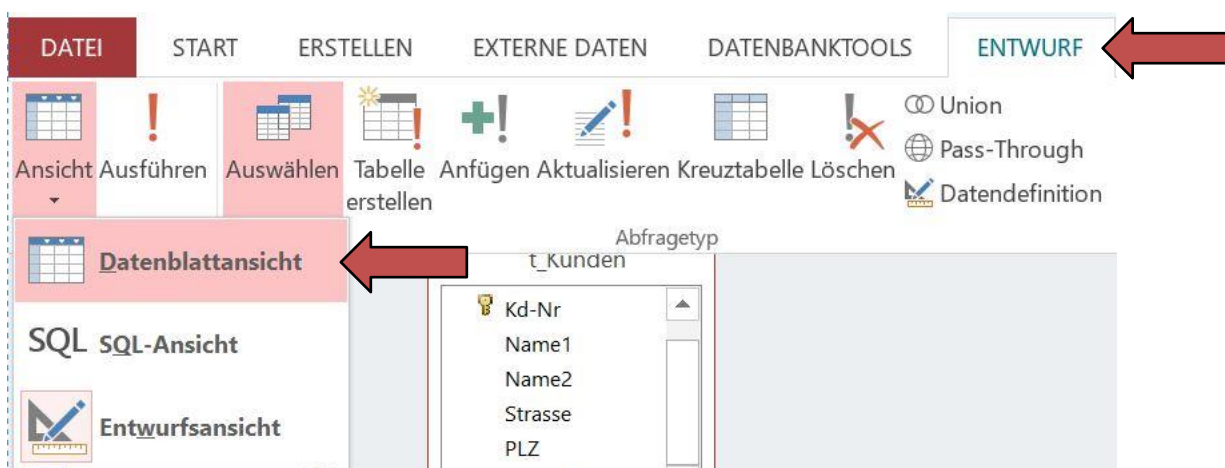
- 4a: Ziehen Sie das Datenfeld **Kd-Nr** aus der Datenfeldliste der Tabelle t_Kunden auf die Zeile „Feld“ der ersten Spalte des QBE-Bereichs.
- 4b: Doppelklicken Sie in der Datenfeldliste der Tabelle t_Kunden auf das Datenfeld Name1.
- 4c: Wählen Sie in der dritten Spalte des QBE-Bereichs über das Drop-Down Menü in der Zeile „Feld“ den Datenfeldnamen Name2 aus. Hinweis: Sie müssen mit der Maus zunächst in die Zeile hinein klicken, damit das Drop-Down Menü sichtbar wird.

Wenn Sie alles richtig gemacht haben, sollten die drei Datenfelder nun in den ersten drei Spalten des QBE-Bereichs jeweils in der Zeile „Feld“ angezeigt werden. In der zweiten Zeile „Tabelle“ wird gleichsam jeweils die Tabelle angezeigt, aus welcher die betreffenden Datenfelder stammen:

Feld:	Kd-Nr	Name1	Name2
Tabelle:	t_Kunden	t_Kunden	t_Kunden
Sortierung:			
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:			
oder:			

Die Kontrollhäkchen in der Zeile „Anzeigen“ ermöglichen es, ein bestimmtes Datenfeld in der Datenblattansicht je nach Bedarf ein- oder auszublenden. Um ein Datenfeld in der Datenblattansicht auszublenden bzw. nicht anzuzeigen, muss das betreffende Kontrollhäkchen deaktiviert werden.

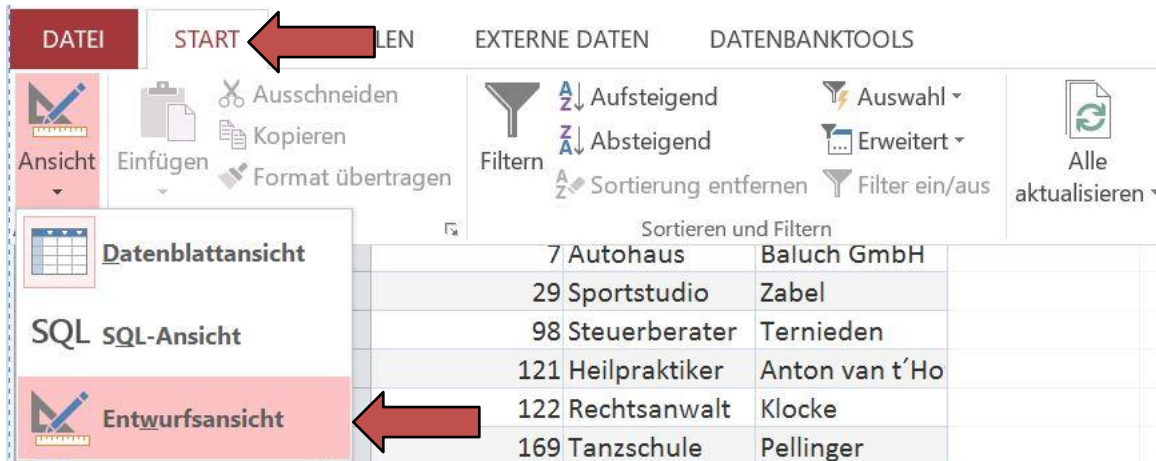
Die bisherige Darstellung unserer Abfrage ist lediglich der Abfrageentwurf. Letzten Endes wollen wir jedoch auch das **Abfrageergebnis** sehen – also die **Datenblattansicht** bzw. das so genannte **Dynaset**. Um von der Entwurfsansicht auf die Datenblattansicht der Abfrage umzuschalten, rufen Sie im Register **Entwurf** bitte den Befehl **Ansicht > Datenblattansicht** auf:



Daraufhin wird Ihnen das **Abfrageergebnis** bzw. das so genannte **Dynaset** der Abfrage in Form einer - wie auch schon von den Datenbanktabellen her bekannten - Datenblattansicht angezeigt. Wie Sie sehen können, werden die einzelnen Spalten bzw. Datenfelder dabei in der gleichen Reihenfolge wie der des Abfrageentwurfs angezeigt:

Abfrage1		
Kd-Nr	Name1	Name2
7	Autohaus	Baluch GmbH
29	Sportstudio	Zabel
98	Steuerberater	Ternieden
121	Heilpraktiker	Anton van t'Ho
122	Rechtsanwalt	Klocke

Damit wäre unsere Aufgabe auch fast schon erledigt – fast! Laut unserer Aufgabenstellung, soll die Kundenliste alphabetisch nach dem Namen sortiert sein, was jedoch noch nicht der Fall ist. Um dies zu tun, müssen wir noch eine entsprechende Anpassung an unserem Abfrageentwurf vornehmen. Um vom Dynaset zum Abfrageentwurf zurück zu kehren, rufen Sie nun im Register **Start** bitte den Befehl **Ansicht > Entwurfsansicht** auf:



Um das Abfrageergebnis nun noch alphabetisch nach Name1 zu sortieren, klicken Sie in der Entwurfsansicht in die Zeile „Sortierung“ der betreffenden Spalte, woraufhin ein Drop-Down Menü erscheint, über welches Sie auswählen können, ob das Abfrageergebnis nach dieser Spalte **aufsteigend sortiert**, **absteigend sortiert** oder **nicht sortiert** werden soll:

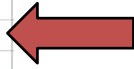
Feld:	Kd-Nr	Name1	Name2
Tabelle:	t_Kunden	t_Kunden	t_Kunden
Sortierung:			
Anzeigen:	<input checked="" type="checkbox"/>	Aufsteigend	<input checked="" type="checkbox"/>
Kriterien:		Absteigend	
oder:		(nicht sortiert)	

Bei einer aufsteigenden Sortierung wird von A nach Z oder bei Zahlen von klein nach groß sortiert. Bei einer absteigenden Sortierung hingegen wird von Z nach A oder bei Zahlen von groß nach klein sortiert. Da eine alphabetische Sortierung im Regelfall von A nach Z verläuft, wählen Sie bitte „Aufsteigend“ aus.

Nun soll die Sortierung jedoch nicht nur nach **Name1**, sondern in **2. Ordnung** auch nach **Name2** erfolgen. Beispiel: Angenommen, das Datenfeld **Name1** beinhaltet den Nachnamen des Kunden und das Datenfeld **Name2** dessen Vornamen. Wenn wir nun mehrere Kunden mit dem Nachnamen „Müller“ (Name1) in der Datenbank hätten, so würden wir wollen, dass die Sortierung der Müllers weiterhin in 2. Ordnung nach deren Vornamen (Name2) erfolgt – also dass z. Bsp. Alfons Müller in der Liste vor Peter Müller käme.

Um das Abfrageergebnis also nun in 2. Ordnung weiterhin alphabetisch nach dem Datenfeld **Name2** zu sortieren, klicken Sie in der Entwurfsansicht nun noch in die Zeile „Sortierung“ der Spalte **Name2** und wählen Sie über das Drop-Down Menü auch hier „Aufsteigend“ aus. Ihre Entwurfsansicht sollte daraufhin wie folgt aussehen:

Feld:	Kd-Nr	Name1	Name2
Tabelle:	t_Kunden	t_Kunden	t_Kunden
Sortierung:		Aufsteigend	Aufsteigend
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:			
oder:			



Wenn wie in diesem Fall nach mehreren Feldern sortiert wird, so ist die **Reihenfolge, in der die Felder im QBE-Bereich erscheinen** auch **maßgeblich für die Reihenfolge bzw. die Priorität der Sortierung**. Zunächst wird nach dem Feld links sortiert und dann nach den rechts liegenden Feldern. In obigem Beispiel wird somit in **1. Ordnung** nach dem Feld **Name1** sortiert und sodann in **2. Ordnung** nach dem Feld **Name2**.

Rufen Sie nun nochmals die Datenblattansicht zu Ihrer Abfrage auf und betrachten Sie das Abfrageergebnis. Wie Ihnen auffallen wird, ist das Abfrageergebnis in erster Ordnung nach **Name1** alphabetisch sortiert (von Autohaus bis Tanzschule). Für gleiche Einträge unter **Name1** (wie z. Bsp. bei „Restaurant“) erfolgt die Sortierung jedoch weiterhin in 2. Ordnung nach Name2 (Restaurant „Bosniak“ kommt in der Liste vor Restaurant „Zur Wilden Sau“).

Speichern Sie Ihre Abfrage nun ab, indem Sie das Abfragefenster wahlweise in der Entwurfsansicht oder in der Datenblattansicht schließen. Daraufhin erscheint eine Abfrage, die es Ihnen ermöglicht, Ihre Abfrage unter einem bestimmten Namen zu speichern. Speichern Sie diese unter der Bezeichnung q_Abfrage01, wobei das q für „Query“steht:

Speichern unter
?
✕

Abfragename:

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

1. Erstellen Sie die Abfrage wie zuvor beschrieben und speichern Sie diese als „q_Abfrage01“.
2. Erstellen Sie eine Artikelliste per Abfrage auf die Tabelle t_Artikel, aus welcher nur die Bezeichnung und der Preis der einzelnen Artikel hervor geht. Die Daten sollen aufsteigend nach dem Preis sortiert sein. Speichern Sie die Abfrage als „q_Abfrage02“.
3. Erstellen Sie eine alphabetisch sortierte Liste aller in der Datenbank enthaltenen Orte. Die Liste soll nur die Ortsbezeichnungen ohne Postleitzahlen enthalten. Speichern Sie die Abfrage als „q_Abfrage03“.

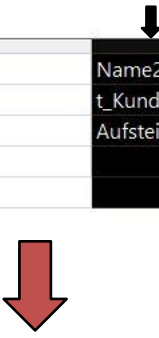
4. Kopfnuss:

Erstellen Sie im Rahmen einer neuen Abfrage nochmals eine Kundenliste, welche lediglich die Datenfelder **Kd-Nr**, **Name1** und **Name2** enthält. Die Liste soll in 1. Ordnung nach dem Datenfeld **Name1** und in 2. Ordnung nach dem Datenfeld **Name2** sortiert sein.

Die Abfrage soll nun jedoch dahingehend modifiziert werden, dass die Liste in **1. Ordnung** nach **Name2** und erst in **2. Ordnung** nach **Name1** sortiert wird. Die Spalte **Name1** soll in der Datenblattansicht jedoch weiterhin **vor** der Spalte **Name2** stehen!

Erläuterung der Hintergrundproblematik:

Um die Sortierreihenfolge zu ändern, so dass zunächst in 1. Ordnung nach **Name2** und sodann erst in 2. Ordnung nach **Name1** sortiert wird, ist das Datenfeld **Name2** entsprechend links vor **Name1** zu verschieben:



Feld:	Kd-Nr	Name1	Name2
Tabelle:	t_Kunden	t_Kunden	t_Kunden
Sortierung:		Aufsteigend	Aufsteigend
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:			
oder:			

Feld:	Kd-Nr	Name2	Name1
Tabelle:	t_Kunden	t_Kunden	t_Kunden
Sortierung:		Aufsteigend	Aufsteigend
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:			
oder:			

Dies bewirkt allerdings gleichsam, dass **Name2** sodann auch in der Datenblattansicht vor **Name1** erscheint. D.h. die **Reihenfolge der Datenfelder im QBE-Bereich** wirkt sich sowohl auf deren **Reihenfolge in der Datenblattansicht** UND auch hinsichtlich deren **Sortierreihenfolge** aus!

Überlegen Sie sich, wie Sie das Dilemma auflösen können und modifizieren Sie Ihre Abfrage entsprechend. **Tipp:** Einen diesbezüglichen Lösungsansatz bietet die Zeile „Anzeigen“ mit den darin befindlichen Kontrollhäkchen. Speichern Sie Ihre Abfrage unter dem Namen „q_Abfrage04“.

4.3 Abfragen mit Kriterien

Kriterien ermöglichen uns die Auswahl bzw. Anzeige all derjenigen Datensätze bzw. Zeilen einer Tabelle, auf welche bestimmte Eigenschaften zutreffen. So können wir auf diese Weise z. Bsp. in Erfahrung bringen, welche Kunden ein bestimmtes Produkt gekauft oder welche Kunden weiblichen Geschlechts sind, um diese z. Bsp. in einer Werbeaktion für Damenwäsche ganz gezielt ansprechen zu können. Öffnen Sie zur Erläuterung anhand eines konkreten Beispiels in Access den letzten Stand der Datenbankdatei [Computerladen.accdb](#).

Beispielaufgabe: Wir wollen eine Liste derjenigen Artikel abfragen, die derzeit nicht an Lager sind.

1. Erstellen Sie hierzu über das Register **ERSTELLEN > Abfrageentwurf** eine neue Abfrage in der Entwurfsansicht. Da die erforderlichen Daten für unsere Abfrage aus der Tabelle **t_Artikel** stammen, wählen Sie bitte diese Tabelle für die Abfrage aus.
2. Als **Datenfelder** übernehmen Sie bitte die Felder **Art-Nr**, **Bezeichnung** und **Bestand** in den QBE-Bereich:

Feld:	Art-Nr	Bezeichnung	Bestand
Tabelle:	t_Artikel	t_Artikel	t_Artikel
Sortierung:			
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:			
oder:			

Bevor wir weitermachen, rufen Sie zunächst das Dynaset der Abfrage auf, indem Sie von der Entwurfsansicht in die Datenblattansicht wechseln:

Artikel-Numr	Bezeichnung	Bestand
101	PC 3,0 GHZ	2
102	Bundle PC 2,2 GHZ	1
103	PC 2,3 GHZ	0
110	PC 2,6 GHz	5
120	PC 2,4 GHZ+Monitor 1	1

Wie Sie sehen können, beinhaltet das Abfrageergebnis sämtliche Artikel aus der Tabelle t_Artikel – ganz unbenommen dessen, wie hoch der jeweilige Lagerbestand ist. Dies wollen wir nun im Weiteren ändern, da wir im Abfrageergebnis nur diejenigen Artikel sehen wollen, die momentan **nicht an Lager** sind und somit einen **Bestand von 0** haben. Wechseln Sie daher nun nochmals zurück in die Entwurfsansicht der Abfrage.

3. Damit im Dynaset unserer Abfrage nur diejenigen Artikel erscheinen, deren **Bestand = 0** ist, müssen wir dies als **Kriterium** in der Spalte Bestand angeben. Der entsprechende Eintrag muss dabei mittels eines **Vergleichsoperators** sowie eines **Vergleichswertes** erfolgen – in unserem Fall also **=0**

Feld:	Art-Nr	Bezeichnung	Bestand
Tabelle:	t_Artikel	t_Artikel	t_Artikel
Sortierung:			
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:			=0
oder:			

4. Rufen Sie nun erneut das Abfrageergebnis auf. Wie Sie sehen können, werden Im **Abfrageergebnis** jetzt **nur noch diejenigen Datensätze** bzw. Artikel angezeigt, **auf welche der Vergleich zutrifft** – d.h. deren Bestand = 0 ist.

Artikel-Numr	Bezeichnung	Bestand
223	Upgradesatz	0
1221	Graphik-Karte TIGRA	0
5001	Monitor 17" TFT	0
103	PC 2,3 GHZ	0
6012	Laser HP xxxx	0
*	0	

4.3.1 Vergleichsoperatoren für Kriterien

Als **Vergleichswerte** von Kriterien können **Zahlen**, **Texte** oder auch **Kalenderdaten** dienen, wobei Texte und Kalenderdaten mittels entsprechender Sonderzeichen als solche zu **kennzeichnen** sind. Zur Kennzeichnung von **Texten** werden **Anführungszeichen** verwendet, für Datumsangaben hingegen **Nummernzeichen** (#).

Hinsichtlich der **Vergleichsoperatoren** sind die bereits aus Excel bekannten Vergleichsoperatoren zu verwenden:

Vergleichsoperatoren:		Kennzeichnung:	
=	gleich	Zahlen	ohne Kennzeichnung z. Bsp.: < 345
<	kleiner		
>	größer	Texte	Anführungszeichen z. Bsp.: >= "Meier"
<=	kleiner oder gleich		
>=	größer oder gleich	Kalenderdaten	Nummernzeichen z. Bsp.: = #01.02.2017#
<>	ungleich		

Nachfolgend einige **Beispiele verschiedener Vergleichskriterien**:

Vergleichskriterien	zeigt	zeigt nicht
= 123 oder nur*) 123	123	122, 124
> 123	124, 125,, 122, 123
< 123	..., 121, 122	123, 124, ...
>= 123	123, 124, 125,, 121, 122
<= 123	..., 122, 123	124, 125, ...
<> 123	..., 122, 124, ...	123
#16.09.2016#	16.09.2016	..., 15.09.2016, 17.09.2016, ...
< #17.09.2016#	..., 15.09.2016, 16.09.2016	17.09.2016, 18.09.2016, ...
"Borken"	Borken	Ahaus, Reken
< "Borken"	Ahaus	Borken, Reken
> "Ahaus"	Borken, Reken	Ahaus

*) Wird **kein Vergleichsoperator** verwendet, unterstellt Access automatisch, dass ein „**ist gleich**“ Vergleich durchgeführt werden soll. D.h. in diesem Fall verwendet Access automatisch das Gleichheitszeichen als Vergleichsoperator. Das Gleichheitszeichen kann daher als einziger Vergleichsoperator auch weg gelassen werden.

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

1. Erstellen Sie die Abfrage wie zuvor beschrieben (es sind die Artikelnummer, die Bezeichnung und die Anzahl derjenigen Artikel anzuzeigen, die derzeit nicht am Lager sind) und speichern Sie diese als „q_Abfrage05“.
2. Es sind alle Aufträge des Kunden 591 anzuzeigen. Speichern Sie die Abfrage als „q_Abfrage06“.
3. Es sind alle Aufträge anzuzeigen, die vor dem 01.09.2016 erteilt wurden. Speichern Sie die Abfrage als „q_Abfrage07“.
4. Es sind alle Kunden anzuzeigen, die im Ort mit der Postleitzahl 48683 wohnen. Speichern Sie die Abfrage als „q_Abfrage08“.
5. Welche Postleitzahl hat Ahaus? Speichern Sie die Abfrage als „q_Abfrage09“.

4.3.2 Mustervergleich mit dem Wie-Operator

Der **Wie-Operator** ermöglicht es in **Verbindung mit** bestimmten **Platzhalterzeichen**, Mustervergleiche anzustellen – also **nach bestimmten Mustern** in einer **Zeichenkette zu suchen**.

Folgende **Platzhalterzeichen** stehen dabei zur Verfügung:

Platzhalterzeichen	Bedeutung / Erläuterung
*	Jokerzeichen – steht für beliebig viele Zeichen
?	Jokerzeichen – steht für genau ein Zeichen
#	Jokerzeichen – steht für genau eine Ziffer
[...]	Zeichenmuster – alle zwischen den eckigen Klammern stehenden Zeichen
!	Nicht – Ausschließung des/der folgenden Zeichen(s)

Die folgende Tabelle zeigt einige **Beispiele** zum Mustervergleich:

Wie-Beispiel	zeigt	Zeigt nicht	Anmerkung
Wie "m*"	Madrid, Mannheim	Orte, die nicht mit m beginnen	* = Jokerzeichen
Wie "A*n"	Aachen	Ahaus	
Wie "*t *"	Portland	Alles, was kein tl hat	
Wie "*en"	Reken, Borken	Alles, was nicht auf en endet	
Wie "a????"	Ahaus	Aachen	? = genau ein Zeichen
Wie "[m-r]*"	Madrid, Paris, Rom	Ahaus, Steinfurt	Zeichenmuster
Wie "[!A]*"	Borken, Reken	Ahaus	! = Nicht
Wie "[!m-r]*"	Leverkusen, Soest	Madrid, Rom	
Wie "##68#"	48683	48653	# = eine Ziffer

Der Wie-Operator ist dabei ebenfalls in die Zeile Kriterien der betreffenden Spalte einzutragen und darf nicht vergessen werden! :

Feld:	Kd-Nr	Name1	Name2	Strasse	PLZ
Tabelle:	t_Kunden	t_Kunden	t_Kunden	t_Kunden	t_Kunden
Sortierung:					
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:					Wie "##68#"
oder:					

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

1. Zeigen Sie die Kunden an, deren Name1 mit „K“ anfängt. Speichern Sie diese als „q_Abfrage10“.
2. Welche Kunden wohnen im Postleitzahlenbereich „46“? Speichern Sie die Abfrage als „q_Abfrage11“.
3. Ermitteln Sie, welche Festplatten in der Tabelle t_Artikel enthalten sind. Speichern Sie die Abfrage als „q_Abfrage12“.
4. Bei welchen Kundenkonditionen ist der Text „10 Tage 2%“ enthalten? Speichern Sie die Abfrage als „q_Abfrage13“.

4.3.3 Kriterien mit Parameterübergabe

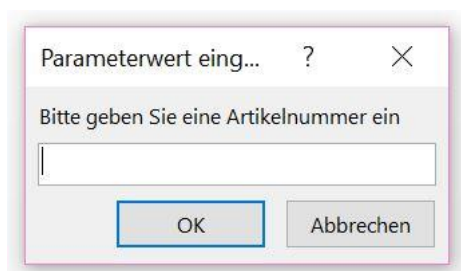
Abfragen sind so flexibel, dass man anstatt eines festgelegten Wertes für ein Kriterium auch einen offenen **Parameter** einrichten kann, dessen **Wert** sodann **bei der Durchführung der Abfrage vom jeweiligen Benutzer** je nach aktuellem Bedarf über ein Dialogfeld **eingegeben** werden kann.

Angenommen, wir wollen eine Abfrage zur Anzeige eines bestimmten Artikels erstellen, wobei jedoch der Nutzer zum Zeitpunkt der Abfrage selbst entscheiden kann, zu welcher Artikelnummer er die zugehörigen Artikeldaten sehen möchte.

Hierzu tragen wir in der Zeile Kriterien der Spalte Artikelnummer **hinter dem erforderlichen Vergleichsoperator in eckigen Klammern** eine entsprechenden **Dialoghinweis** ein, anhand dessen der Anwender bei Aufruf der Abfrage weiß, welche Information von ihm benötigt wird:

Feld:	Art-Nr	Bezeichnung	Preis	Bestand
Tabelle:	t_Artikel	t_Artikel	t_Artikel	t_Artikel
Sortierung:				
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:	= Bitte geben Sie eine Artikelnummer ein			
oder:				

Bei Aufruf der Abfrage erscheint dem Benutzer sodann ein Dialogfenster mit dem Hinweis, der in die eckigen Klammern eingetragen wurde:



Der **Parameterwert**, den der Benutzer in das Dialogfeld eingibt, wird sodann als entsprechender **Vergleichswert** herangezogen und bildet **zusammen mit dem vor der eckigen Klammer stehenden Vergleichsoperator** das **Vergleichskriterium**.

Wollen wir z. Bsp. eine Liste aller Artikel anzeigen lassen, deren Artikelnummer kleiner oder gleich der Artikelnummer ist, die der Benutzer in das Dialogfeld eingibt, so muss vor der eckigen Klammer der entsprechende Vergleichsoperator (<=) angegeben werden:

Feld:	Art-Nr	Bezeichnung	Preis	Bestand
Tabelle:	t_Artikel	t_Artikel	t_Artikel	t_Artikel
Sortierung:				
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
oder:	<= [Bitte geben Sie eine Artikelnummer ein]			

Nach Eingabe einer bestimmten Artikelnummer (z. Bsp. 150) in das Dialogfenster erscheinen im Abfrageergebnis sodann alle Artikel mit einer Artikelnummer, die kleiner oder gleich der vom Benutzer eingegebenen Artikelnummer ist:

Artikel-Nr	Bezeichnung	Preis	Bestand
101	PC 3,0 GHZ	399,00	2
102	Bundle PC 2,2 GHZ	899,00	1
103	PC 2,3 GHZ	499,00	0
110	PC 2,6 GHz	790,00	5
120	PC 2,4 GHZ+Monitor 1	699,00	1
150	Notebook 900 Celeron	1200,00	1
*	0		

Parameter können im Übrigen auch **mit Platzhalterzeichen kombiniert** werden – z. Bsp.:

Wie [Bitte einen Buchstaben eingeben] & "*"

Dies würde alle Datensätze anzeigen, die mit dem vom Benutzer eingegebenen Buchstaben beginnen. Das **&**-Zeichen muss zur **Verkettung** zwischen Parameter und Platzhalterzeichen eingegeben werden. Das **Platzhalterzeichen** muss weiterhin von **Anführungszeichen** umgeben werden, da es stellvertretend für **Text** steht.

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

1. Es soll eine Liste aller Artikel angezeigt werden, deren Artikelnummer kleiner oder gleich der vom Benutzer eingegebenen Artikelnummer ist. Speichern Sie diese als „q_Abfrage14“.
2. Es sollen alle Kunden angezeigt werden, deren Name1 eine vom Benutzer einzugebende Zeichenfolge enthält. Speichern Sie die Abfrage als „q_Abfrage15“.

4.4 Abfragen mit Verknüpfungsoperatoren

Verknüpfungsoperatoren sind so genannte **Bool'sche Operatoren**, die eine **logische Verknüpfung mehrerer Kriterien** herstellen und uns somit die Auswahl bzw. Anzeige all derjenigen Datensätze bzw. Zeilen einer Tabelle ermöglichen, auf welche eine **Kombination verschiedener Eigenschaften** zutreffen.

Grundsätzlich unterscheiden wir verschiedene Arten **logischer Verknüpfungsoperatoren**:

4.4.1 UND - Verknüpfung

Die **UND**-Verknüpfung wird benötigt, wenn in einer Abfrage **zwei (oder mehrere) Vergleiche** durchgeführt werden und dabei **beide (oder alle) Kriterien zutreffen** müssen.

Im Falle der **UND**-Verknüpfung gibt es also eine **Gesamtbedingung**, die **aus mehreren Teilbedingungen** besteht, wobei **jede einzelne Teilbedingung erfüllt sein muss**, damit die Gesamtbedingung als erfüllt gilt. Trifft auch nur eine einzige Teilbedingung nicht zu, so gilt gleichermaßen auch die Gesamtbedingung als nicht erfüllt und der jeweilige Datensatz wird nicht angezeigt:

Teilbedingung 1	Teilbedingung 2	Gesamtbedingung	Anzeige des Satzes
Wahr	Wahr	Wahr	Ja
Wahr	Falsch	Falsch	Nein
Falsch	Wahr	Falsch	Nein
Falsch	Falsch	Falsch	Nein

Hinsichtlich der **UND**-Verknüpfung bestehen grundsätzlich folgende zwei Möglichkeiten:

- a) Es müssen zwei (oder mehrere) **Kriterien in unterschiedlichen Datenfeldern** zutreffen

In diesem Fall müssen die verschiedenen Kriterien **in der jeweils gleichen Kriterien-Zeile** eingetragen werden, um diese mittels eines logischen **UND** miteinander zu verknüpfen:

The screenshot shows the Microsoft Access Query Design View for a query named 't_Artikel'. The design grid is as follows:

Feld:	Art-Nr	Bezeichnung	Preis	Bestand
Tabelle:	t_Artikel	t_Artikel	t_Artikel	t_Artikel
Sortierung:				
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:			> 100	> 0
oder:				

A red arrow points to the 'oder' row, indicating the logical operator used to connect the criteria.

- b) Es müssen zwei (oder mehrere) **Kriterien im gleichen Datenfeld** zutreffen

In diesem Fall müssen die verschiedenen Bedingungen **in der Zeile „Kriterien“ des betreffenden Datenfeldes** durch das Wort „**Und**“ miteinander verknüpft werden. Das Wort „Und“ ist dabei jeweils zwischen den verschiedenen Kriterien einzutragen:

Feld:	Art-Nr	Bezeichnung	Preis	Bestand
Tabelle:	t_Artikel	t_Artikel	t_Artikel	t_Artikel
Sortierung:				
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:				>0 Und <10
oder:				

In obiger Darstellung würde die Abfrage z. Bsp. lauten: „Zeige alle Artikel an, deren Bestand größer Null **und** gleichzeitig kleiner 10 ist.“ Die Abfrage würde dabei für die folgenden Werte jeweils ergeben:

Bestands-Wert	>0	<10	Anzeige
7	ja	ja	ja
11	ja	nein	nein
0	nein	ja	nein

4.4.2 ODER - Verknüpfung

Die **ODER**-Verknüpfung wird benötigt, wenn in einer Abfrage **zwei (oder mehrere) Vergleiche** durchgeführt werden und dabei **wenigstens eines der verschiedenen Kriterien** zutreffen muss.

Im Falle der **ODER**-Verknüpfung gibt es also eine **Gesamtbedingung**, die **aus mehreren Teilbedingungen** besteht, wobei es **bereits ausreichend** ist, **wenn** nur **eine Teilbedingung erfüllt ist**, damit die Gesamtbedingung als erfüllt gilt:

Teilbedingung 1	Teilbedingung 2	Gesamtbedingung	Anzeige des Satzes
Wahr	Wahr	Wahr	Ja
Wahr	Falsch	Wahr	Ja
Falsch	Wahr	Wahr	Ja
Falsch	Falsch	Falsch	Nein

Auch hinsichtlich der **ODER**-Verknüpfung bestehen grundsätzlich folgende zwei Möglichkeiten:

- a) Es muss eines von zwei (oder mehreren) **Kriterien in unterschiedlichen Datenfeldern** zutreffen

In diesem Fall müssen die verschiedenen Kriterien **in jeweils unterschiedlichen Kriterien-Zeilen** der betreffenden Datenfelder eingetragen werden, um diese mittels eines logischen **ODER** miteinander zu verknüpfen – so z. Bsp.:

- das erste Kriterium in der **Zeile „Kriterien“** des betreffenden Datenfeldes,
- das zweite Kriterium in der **Zeile „oder“** der betreffenden Datenfeldes,
- und das dritte Kriterium in der **unterhalb der der Zeile „oder“** befindlichen Zeile des betreffenden Datenfeldes, etc.

Die folgende Abfrage würde z. Bsp. alle Artikel anzeigen, deren *Art-Nr. kleiner als 300* **ODER** deren *erster Buchstabe in der Bezeichnung ein H* **ODER** deren *Preis größer als 100 €* **ODER** deren *Bestand kleiner als 10* ist.

Feld:	Art-Nr	Bezeichnung	Preis	Bestand
Tabelle:	t_Artikel	t_Artikel	t_Artikel	t_Artikel
Sortierung:				
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:	<300	Wie "H**"	> 100	<10
oder:				

Achten Sie bei entsprechenden Aufgaben **auf die Feinheiten in der sprachlichen Formulierung**, denn die Aufgabenstellung für diese Abfrage würde z. Bsp. **sprachlich formuliert** wie folgt lauten: „Es sollen alle Artikel mit einer Artikelnummer kleiner als 300 angezeigt werden, alle Artikel deren Bezeichnung mit einem H beginnt, alle Artikel deren Preis über 100 € liegt **und** auch alle Artikel, deren Bestand geringer als 10 ist.“

Das **und** in der obigen Formulierung ist also **kein logisches UND**, sondern ein **sprachliches und**, welches in diesem Falle die Bedeutung eines **logischen ODERs** hat!

Achten Sie für eine **ODER-Verknüpfung** zudem darauf, die betreffenden Kriterien stets in **unterschiedliche Zeilen** einzutragen, da die Kriterien bei Eintrag in die gleiche Zeile durch ein logisches UND miteinander verknüpft würden und sodann alle zutreffen müssten!

b) Es muss eines von zwei (oder mehreren) **Kriterien im gleichen Datenfeld** zutreffen

In diesem Fall müssen die verschiedenen Kriterien **in jeweils unterschiedlichen Kriterien-Zeilen** des betreffenden Datenfeldes eingetragen werden, um diese mittels eines logischen **ODER** miteinander zu verknüpfen – so z. Bsp.:

- das erste Kriterium in der **Zeile „Kriterien“** des betreffenden Datenfeldes,
- das zweite Kriterium in der **Zeile „oder“** der betreffenden Datenfeldes,
- und das dritte Kriterium in der **unterhalb der der Zeile „oder“** befindlichen Zeile des betreffenden Datenfeldes, etc.

Die folgende Abfrage würde z. Bsp. alle Artikel anzeigen, deren *erster Buchstabe in der Bezeichnung ein A* **ODER** *H* **ODER** deren *letzter Buchstabe in der Bezeichnung ein p* **ODER** *z* ist.

Feld:	Art-Nr	Bezeichnung	Preis	Bestand
Tabelle:	t_Artikel	t_Artikel	t_Artikel	t_Artikel
Sortierung:				
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:		Wie "A*"		
oder:		Wie "H*"		
		Wie "*p"		
		Wie "*z"		

Alternativ zum Eintrag der verschiedenen Kriterien in unterschiedliche Zeilen, können diese jedoch auch alle in **einer Kriterien-Zeile des betreffenden Datenfeldes** eingetragen und dabei durch das Wort **„Oder“** miteinander verknüpft werden:

Feld:	Art-Nr	Bezeichnung	Preis	Bestand
Tabelle:	t_Artikel	t_Artikel	t_Artikel	t_Artikel
Sortierung:				
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:				=0 Oder >=100
oder:				

In obiger Darstellung würde die Abfrage z. Bsp. lauten: „Zeige alle Artikel an, deren Bestand gleich Null **oder** größer bzw. gleich 100 ist.“ Die Abfrage würde dabei für die folgenden Werte jeweils ergeben:

Bestands-Wert	=0	>=100	Anzeige
7	nein	nein	nein
0	ja	nein	ja
120	nein	ja	ja

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

1. Zeigen Sie alle Kunden aus Ahaus (PLZ 48683) und Coesfeld (PLZ 48653) an. Speichern Sie diese als „q_Abfrage16“.
2. Welche Artikel liegen im Preisbereich von einschließlich 100€ bis einschließlich 1000€ ? Speichern Sie die Abfrage als „q_Abfrage17“.
3. Welche Artikel hat der Kunde 591 am 17.08.2016 bestellt? Speichern Sie die Abfrage als „q_Abfrage18“.
4. Welche Artikel liegen im Preisbereich von einschließlich 100€ bis einschließlich 1000€ und sind derzeit nicht am Lager? Speichern Sie die Abfrage als „q_Abfrage19“.
5. Ermitteln Sie die Kunden, die im Namen1 oder im Namen2 mit "K" beginnen! Speichern Sie die Abfrage als „q_Abfrage20“.

4.4.3 Der ZWISCHEN - Operator

Mit Hilfe des **ZWISCHEN**-Operators können **Werte** abgefragt werden, die **innerhalb zweier Grenzwerte** liegen, wofür ansonsten zwei Kriterien mit Vergleichsoperatoren erforderlich wären, welche mittels UND verknüpft werden müssten. Die **Werte, die als Grenze dienen**, werden dabei in die Auswahl **mit einbezogen**. Der ZWISCHEN-Operator kann dabei sowohl für Zahlenwerte, Datumsangaben wie auch für Textwerte (alphabetische Ordnung) verwendet werden.

Die Formulierung muss dabei mit **ZWISCHEN** beginnen, wobei **zwischen den beiden Grenzwerten** weiterhin das Wort **UND** stehen muss:

Formulierung mit ZWISCHEN	Formulierung mit Vergleichsoperatoren
Zwischen 100 und 1000	>= 100 und <= 1000

Beispiele:

Ausdruck	Zeigt	Zeigt nicht
Zwischen #01.01.96# UND #31.12.96#	alle Datumsangaben im Jahr 1996	Datumsangaben vor oder nach 1996
Zwischen 100 und 1000	100, ..., 1000	..., 99, 1001, ...
Zwischen "a*" und "i*"	Aachen, ... , Izmir	Jachenau, ...

4.4.4 Der IN - Operator

Mit Hilfe des **IN**-Operators können Werte aus **einer definierten Menge** abgefragt werden, wofür ansonsten mehrere "=" Vergleichskriterien erforderlich wären, die mittels ODER verknüpft werden müssten. Alle **zur definierten Menge zugehörigen Werte**, werden dabei in die Auswahl **mit einbezogen**.

Die Mengendefinition erfolgt dabei durch den Operator **IN** – gefolgt von einer **Auflistung der zugehörigen Werte**, welche **von Klammern umschlossen** und **durch Semikola voneinander zu trennen** sind:

IN ("Wert A"; "Wert B"; "Wert C")

Beispiele:

Ausdruck	Zeigt	Zeigt nicht	Hinweis
In ("Ahaus"; "Coesfeld")	Ahaus, Coesfeld	alle anderen Orte	; verwenden !
In ("A*"; "Coe*")	nichts		* ist nicht verwendbar

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

1. Welche Orte liegen im PLZ-Bereich von einschließlich 48599 bis einschließlich 48683? Lösen Sie die Abfrage mit Hilfe des Zwischen-Operators und speichern Sie diese als „q_Abfrage21“.
2. Stellen Sie fest, wie viele Exemplare der Software „WIN 8“ und „WIN 10“ am Lager sind. Lösen Sie die Abfrage mit Hilfe des IN-Operators und speichern Sie die Abfrage als „q_Abfrage22“.

4.5 Abfragen über mehrere Tabellen

Oftmals beinhalten Datenbankabfragen spezifische **Informationen aus verschiedenen Tabellen**. In einem solchen Fall sind **sämtliche Tabellen in den Abfrageentwurf zu übernehmen, die für die Abfrage relevant sind!** Dabei sind folgende **Regeln** zu beachten:

- Es muss **jede Tabelle** in den Abfrageentwurf übernommen werden, **welche spezifische Datenfelder enthält, die im Abfrageergebnis erscheinen sollen**.
- Weiterhin muss auch eine **jede Tabelle** übernommen werden, **über welche die Tabellen aus a) durch entsprechende Beziehungen miteinander verbunden sind**. D.h. im Abfrageentwurf müssen die enthaltenen Tabellen durchgängig miteinander verbunden sein. **Keine Tabelle darf isoliert** im Abfrageentwurf stehen.
- Darüber hinaus dürfen **keine** weiteren bzw. **überflüssigen Tabellen** im Abfrageentwurf vorhanden sein, **die für die Abfrage nicht zwingend relevant sind**.
- Es darf **keine Tabelle doppelt** aufgeführt sein.

Tipp: Um zu ermitteln, welche Tabellen in den Abfrageentwurf übernommen werden müssen, ist es hilfreich, sich die Datenbank zuvor nochmals in der **Beziehungsansicht** anzusehen, um einen entsprechenden Überblick zu erhalten – insbesondere dann, wenn einem die Datenbank und deren Struktur noch unbekannt ist.

Hierzu folgende **Beispielaufgabe**:

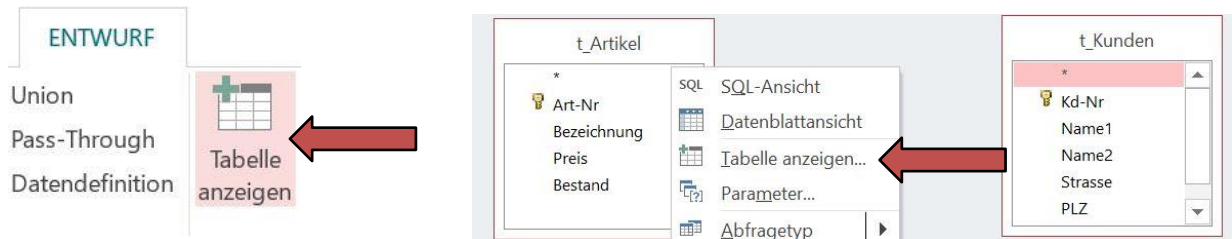
Es sollen alle Kunden (Name1, Name2) angezeigt werden, die den Artikel „EXCEL 2013“ bestellt haben.

- Die spezifischen Datenfelder für diese Abfrage befinden sich in der Tabelle **t_Kunden** (Name1 und Name2) sowie in der Tabelle **t_Artikel**. Also müssen diese beiden Tabellen zwingend in den Abfrageentwurf übernommen werden und die entsprechenden Datenfelder dem QBE-Bereich hinzugefügt werden. Als **Kriterium** ist in der Spalte (Artikel-) **Bezeichnung** zudem das Vergleichskriterium **=“EXCEL 2013“** einzutragen:

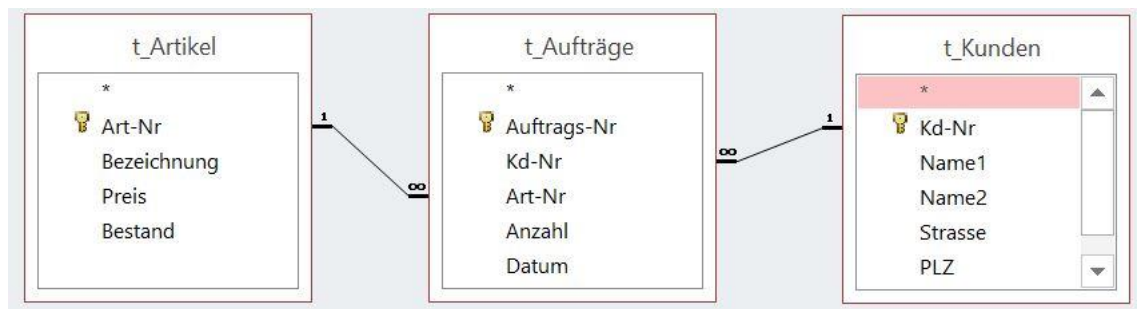
Feld:	Name1	Name2	Bezeichnung		
Tabelle:	t_Kunden	t_Kunden	t_Artikel		
Sortierung:					
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Kriterien:			= "EXCEL 2013"		
oder:					

- b) Nun wird diese Abfrage jedoch kein korrektes Abfrageergebnis liefern, da die Regel b) nicht eingehalten wurde, gemäß derer die im Abfrageentwurf enthaltenen Tabellen durchgängig miteinander verbunden sein müssen. Die Tabellen **t_Kunden** und **t_Artikel** sind in der Datenbank über die Tabelle **t_Aufträge** miteinander verbunden. Also muss auch die Tabelle **t_Auftrag** unserem Abfrageentwurf mit hinzugefügt werden.

Tipp: Falls Sie es bei der ersten Tabellenauswahl versäumt haben, eine erforderliche Tabelle mit in den Abfrageentwurf zu übernehmen, so können Sie das Tabellenauswahlfenster über das Register **Entwurf** und den Menüpunkt **Tabelle anzeigen** nochmals aufrufen, um die Tabelle nachträglich einzufügen. **Alternativ** können Sie auch einen **Rechtsklick in den Tabellenbereich** vornehmen und aus dem daraufhin erscheinenden Kontextmenü den Menüpunkt **Tabelle anzeigen** auswählen.



Nachdem die Tabelle **t_Aufträge** der Abfrage hinzugefügt wurde, erscheinen nun alle im Abfrageentwurf enthaltenen **Tabellen** über die entsprechenden Beziehungen als **durchgängig miteinander verbunden**:



- c) Die Tabellen **t_Orte** und **t_Kundenkonditionen** enthalten keinerlei Datenfelder, die für diese Abfrage relevant sind und dürfen daher auch nicht in den Abfrageentwurf übernommen werden.
- d) Zudem darf keine der enthaltenen Tabellen doppelt vorkommen.

Tipp: Falls Sie versehentlich eine überflüssige Tabelle in den Abfrageentwurf übernommen haben oder eine Tabelle doppelt enthalten ist, können Sie diese wieder entfernen, indem Sie die **Tabelle mit der rechten Maustaste anklicken** und aus dem daraufhin erscheinenden **Kontextmenü** den Menüpunkt **Tabelle entfernen** auswählen.

Unsere Abfrage ist somit nun fertig, woraufhin das Abfrageergebnis in der Datenblattansicht aufgerufen werden kann.

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

1. Führen Sie zunächst die zuvor erläuterte Beispielaufgabe durch: Es sollen alle Kunden (Name1, Name2) angezeigt werden, die den Artikel „EXCEL 2013“ bestellt haben.
Führen Sie die Abfrage zunächst fehlerhaft aus, indem Sie
 - a) die Tabelle t_Aufträge auslassen
 - b) eine überflüssige Tabelle (t_Orte oder t_Kundenkonditionen) mit einbinden
 - c) eine Tabelle doppelt einbindenBetrachten Sie jeweils das Abfrageergebnis.
Führen Sie die Abfrage sodann korrekt nach den vorgegebenen Regeln durch. Betrachten Sie das Abfrageergebnis und speichern Sie die Abfrage als „q_Abfrage23“.
2. Stellen Sie fest, welche Kunden (Name1 und Name2) in Gronau wohnen. Speichern Sie die Abfrage als „q_Abfrage24“.
3. Stellen Sie fest, welche Artikel (Artikel-Nr., Bezeichnung und Preis) in welcher Anzahl im August 2016 bestellt wurden. Speichern Sie die Abfrage als „q_Abfrage25“.
4. Stellen Sie fest, welche Artikel (Artikel-Nr., Bezeichnung und Preis) an Kunden aus Coesfeld verkauft wurden. Speichern Sie die Abfrage als „q_Abfrage26“.
5. Stellen Sie fest, welche Artikel (Artikel-Nr., Bezeichnung und Preis) mit 10% Rabatt und 2% Skonto verkauft wurden. Speichern Sie die Abfrage als „q_Abfrage27“.
6. Stellen Sie fest, welche Kunden (Name1 und Name2) aus Ahaus im September 2016 Artikel mit einem Preis von über 300,00 € oder in einer Menge von mehr als 2 Stück bestellt haben. Speichern Sie die Abfrage als „q_Abfrage28“.

4.6 Abfragen mit Berechnungen

4.6.1 Mathematische Berechnungen

Häufig ist es notwendig, **aus den vorhandenen Daten weitere Daten** (so genannte Prozessdaten) zu **ermitteln, wofür** entsprechende **Berechnungen** in den Abfragen **durchgeführt werden müssen**. Beispielsweise ist es im Rahmen einer Rechnung notwendig, aus der Anzahl der Artikel und dem jeweiligen Einzelpreis den Gesamtpreis für diese Waren zu ermitteln.

In einem solchen Fall muss quasi eine **neue Spalte** (hier Gesamtpreis) **in den QBE-Bereich der Abfrage** integriert werden, deren Wert sich aus der Multiplikation

$$\text{Gesamtpreis} = \text{Artikelanzahl} * \text{Einzelpreis}$$

ergibt.

Um das **Ergebnis der Berechnung** als **neue Spalte in der Abfrage** zu implementieren, ist die **Bezeichnung des entsprechenden Datenfelds bzw. der Spalte mitsamt der** entsprechenden **Rechenanweisung** in einer **vorgeschriebenen Form zu definieren** und als entsprechender **Ausdruck in die gewünschte Spalte des QBE-Bereichs einzutragen**.

Beispielaufgabe: Es soll für alle Aufträge der jeweilige Gesamtpreis aus der Artikelanzahl und dem jeweiligen Einzelpreis errechnet und angezeigt werden.

Hierfür benötigen wir die Datenfelder **Auftrags-Nr.** und **Anzahl** aus der Tabelle **t_Aufträge** sowie (Artikel-) **Bezeichnung** und **Preis** aus der Tabelle **t_Artikel**. Um Aus der jeweils bestellten Anzahl und dem Preis der einzelnen Artikel den Gesamtpreis zu berechnen, müssen wir nun weiterhin noch eine **neue Spalte** mit der Bezeichnung **Gesamtpreis** einrichten und für diese gleichsam die entsprechende **Rechenanweisung definieren**:

The screenshot shows the Access interface. At the top, a relationship diagram connects the **t_Artikel** table (fields: Art-Nr, Bezeichnung, Preis, Bestand) and the **t_Aufträge** table (fields: Auftrags-Nr, Kd-Nr, Art-Nr, Anzahl, Datum). A line connects the **Preis** field in **t_Artikel** to the **Anzahl** field in **t_Aufträge**. Below this, a QBE query grid is displayed with the following data:

Feld:	Auftrags-Nr	Bezeichnung	Preis	Anzahl	Gesamtpreis: [Anzahl]*[Preis]
Tabelle:	t_Aufträge	t_Artikel	t_Artikel	t_Aufträge	
Sortierung:					
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:					
oder:					

A red arrow points to the formula **Gesamtpreis: [Anzahl]*[Preis]** in the 'Feld' row of the QBE grid.

Wie Sie aus der obigen Darstellung ersehen können, sind hinsichtlich der **Definition des berechneten Datenfeldes** bestimmte **Regeln** zu beachten:

1. Die **Definition** des neuen Datenfeldes erfolgt in der Zeile **Feld** der betreffenden Spalte.
2. Der **Name des Datenfeldes** ist **als erstes** in die Zeile **Feld einzutragen**.
3. **Dem Namen** des Datenfeldes **folgt** nicht etwa ein Gleichheitszeichen, sondern **ein Doppelpunkt!**
4. Nach dem Doppelpunkt ist sodann die Berechnungsformel anzugeben, wobei **ein jedes verwendete Datenfeld** von **eckigen Klammern** zu umschließen ist.

Der daraus resultierende Ausdruck ergibt sich somit wie folgt:

2. Name des neuen Datenfelds → **Gesamtpreis:[Anzahl]*[Preis]** ← 4. Berechnungsformel mit verwendeten Datenfeldern in eckigen Klammern

3. Doppelpunkt anstelle eines Gleichheitszeichens

In einem solchen Ausdruck können **die üblichen Rechenzeichen** verwendet werden:

Rechenzeichen	Bedeutung
+	Addition
–	Subtraktion
*	Multiplikation
/	Division
^	Potenzieren
\	Ganzzahlige Division
MOD	Restwertermittlung (Gegenstück zur ganzzahligen Division)

Sofern die übliche Rechenreihenfolge **Punktrechnung vor Strichrechnung** aus Berechnungsgründen zu **ändern** ist, kann dies wie gewohnt durch runde Klammern vorgenommen werden:

Beispiel: **Gesamtpreis:[Anzahl]*([Preis]-5)**

wenn der Kunde z. Bsp. jeden Artikel 5,00 € günstiger als zum Listenpreis bekäme.

Beachten Sie außerdem hinsichtlich der Angabe der Datenfelder in der Berechnungsformel:

Genau genommen, müsste **zu jedem** in der Berechnungsformel verwendeten **Datenfeld** gleichsam **auch die Tabelle angegeben** werden, aus welcher es stammt. Dabei ist der **Tabellenname** stets **vor dem Datenfeldnamen** anzugeben **und** von diesem **durch einen Punkt zu trennen**. Der Ausdruck zur Definition des berechneten Datenfelds Gesamtpreis müsste genau genommen also wie folgt aussehen:

Gesamtpreis:[t_Aufträge.Anzahl]*[t_Artikel.Preis]

↑
Punkt zur Abgrenzung von Tabellenname und Datenfeldname

Glücklicherweise ist Access diesbezüglich jedoch ziemlich **tolerant**, **solange in der Berechnung kein Datenfeld** verwendet wird, welches **den gleichen Datenfeldnamen** hat **wie ein anderes Datenfeld** aus einer anderen Tabelle, die ebenfalls im Abfrageentwurf enthalten ist.

Würde z. Bsp. das Datenfeld **Bestand** aus der Tabelle **t_Artikel** ebenfalls **Anzahl** heißen (anstatt **Bestand**), dann müsste für das Datenfeld **Anzahl** aus der Tabelle **t_Aufträge** zwingend die zugehörige Tabelle mit angegeben werden, aus welcher es entstammt (so wie im obigen Beispiel). Die alleinige Angabe des Datenfeldnamens **Anzahl** wäre ansonsten nicht eindeutig, weil unklar bliebe, welches der beiden Datenfelder namens **Anzahl** für die Berechnung konkret verwendet werden soll.

Da in unserer Abfrage jedoch ein jedes in der Berechnung verwendetes Datenfeld einen eindeutigen Datenfeldnamen hat, welcher in keiner anderen Tabelle der Abfrage ebenfalls vorkommt, können wir in diesem Fall auf die Angabe der zugehörigen Tabellennamen in der Berechnungsformel verzichten.

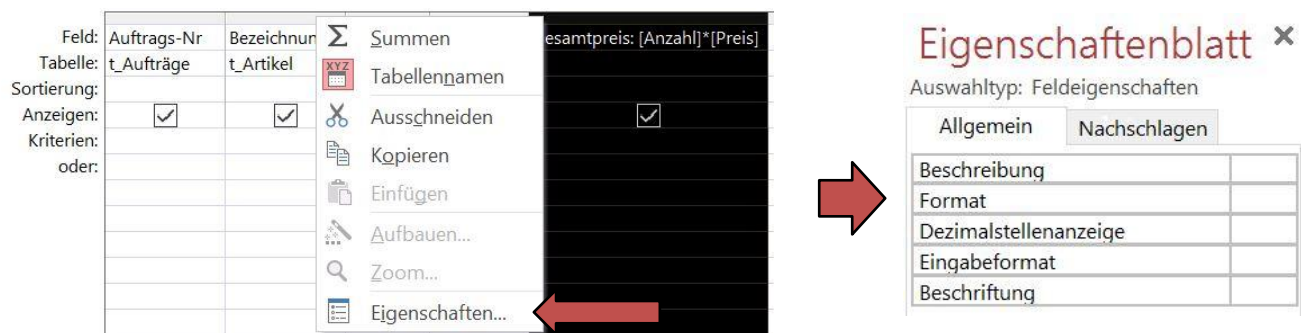
Schauen wir uns nun also das Abfrageergebnis an:

Auftrags-Nr	Bezeichnung	Preis	Anzahl	Gesamtpreis
1	PC 2,6 GHz	790,00	3	4740
2	DDR-RAM 5 GB	198,00	1	792
3	Canon BJX xxx	159,00	1	636
4	CD ROM Laufwerk 32 fach	169,00	2	845
5	EXCEL 2013	229,00	3	1374
6	Maus Noname	9,00	1	36
7	Monitor 21" TFT	299,00	2	1195

Was nun noch etwas unschön ist, ist die Darstellung der Zahlen in der Spalte Gesamtpreis. Diese wollen wir nun noch mit 2 Nachkommastellen und einem €-Symbol formatieren:

Exkurs: Formatierung von Datenfeld-Werten:

Um die Werte einer bestimmten Spalte auf spezifische Weise zu **formatieren**, müssen sie im **Abfrageentwurf** lediglich den entsprechenden **Spaltenkopf** mit der **rechten Maustaste anklicken** und aus dem daraufhin erscheinenden Kontextmenü sodann den Menüpunkt **Eigenschaften** auswählen. Daraufhin erscheint im **rechten Bildschirmbereich** das **Eigenschaftsblatt** für das betreffende Datenfeld:



In der **rechten Spalte des Eigenschaftsblattes** können nun (teilweise durch Drop-Down Menüs) verschiedene **Eigenschaften** für das Datenfeld festgelegt werden:

Beschreibung: Hier können Sie eine Beschreibung für das Datenfeld eingeben.

Beschreibung	
Format	Euro
Dezimalstellenanzeige	2
Eingabeformat	
Beschriftung	Gesamtbetrag

Format: Ermöglicht die Einrichtung verschiedener Formate (z. Bsp. Allgemeine Zahl, Währung, Euro, Prozentzahl, Datumsformate, Zeitformate, etc.) für das Datenfeld.

Dezimalstellenanzeige: Hier können Sie festlegen, wie viele Dezimalstellen angezeigt werden sollen.

Beschriftung: Ermöglicht die Festlegung einer alternativen Bezeichnung für das Datenfeld, welche sodann als Spaltenüberschrift im Abfrageergebnis erscheint.

Nach entsprechender Festlegung der Datenfeldeigenschaften gemäß der obigen Abbildung, sieht unser Abfrageergebnis nun wie folgt aus:

Auftrags-Nr	Bezeichnung	Preis	Anzahl	Gesamtbetrag
1	PC 2,6 GHz	790,00	3	2.370,00 €
2	DDR-RAM 5 GB	198,00	1	198,00 €
3	Canon BJX xxx	159,00	1	159,00 €
4	CD ROM Laufwerk 32 fach	169,00	2	338,00 €
5	EXCEL 2013	229,00	3	687,00 €

Beachten Sie, dass die **Überschrift der Spalte Gesamtpreis** nun **Gesamtbetrag** heißt (Eigenschaft **Beschriftung**) und sämtliche Werte mit **2 Nachkommastellen** und dem **€-Symbol** (Eigenschaften **Format** und **Dezimalstellenanzeige**).

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

- Es soll eine Abfrage erstellt werden, aus der Sie für alle Aufträge die Auftrags-Nummer, die Artikel-Nummer, die Bezeichnung, die Anzahl, den Einzelpreis und den **Gesamtpreis** der bestellten Artikel ansehen können.
Formatieren Sie die berechnete Spalte Gesamtpreis im Euro-Format mit 2 Nachkommastellen und verleihen Sie ihr die Beschriftung **Gesamtbetrag**. Betrachten Sie das Abfrageergebnis und speichern Sie die Abfrage als „q_Abfrage29“.
- Erweitern Sie die Abfrage29 dahingehend, dass in deren Ergebnis auch der jeweilige **Rabattbetrag** (im Euro-Format) für einen jeden Auftrag angezeigt wird.
(**Lösungshinweise:** Es ist möglich, ein errechnetes Datenfeld für weitere Berechnungen in anderen Spalten zu verwenden. Zudem muss ein Datenfeld, welches in einer Berechnung verwendet wird, auch nicht zwingend in den QBE-Bereich übernommen werden – es genügt, wenn es in einer der Tabellen in der Abfrage vorhanden ist.)
Speichern Sie die erweiterte Abfrage als „q_Abfrage30“.
- Erweitern Sie die Abfrage30 dahingehend, dass in deren Ergebnis auch der jeweilige **Skontobetrag** (im Euro-Format) sowie der sich ergebende **Endpreis** (im Euro-Format) für einen jeden Auftrag angezeigt werden. Speichern Sie die erweiterte Abfrage als „q_Abfrage31“.

4.6.2 Verkettung von Texten

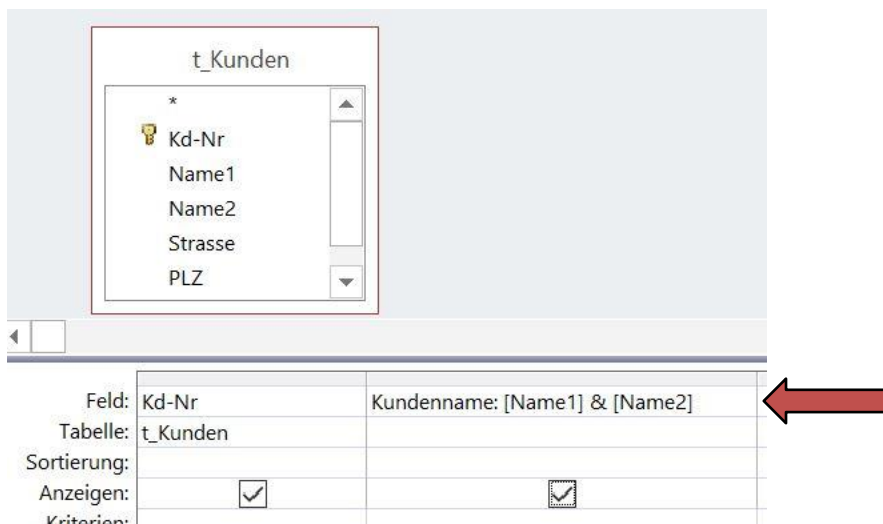
Eine andere Form von „Berechnung“ besteht in der **Verkettung von Texten**. Hierbei werden die Inhalte verschiedener Textfelder in einem neuen Datenfeld zusammen geführt und/oder durch weitere Textbestandteile ergänzt. Um Texte miteinander zu verketteten, dient der **&-Operator** als entsprechendes Verkettungszeichen.

Beispiel: Es soll eine Kundenliste erstellt werden, in welcher der **Name1** und **Name2** des jeweiligen Kunden nicht separat in zwei Spalten, **sondern zusammen in einer Spalte** erscheinen soll.

Um dies zu tun, müssen wir in der Abfrage eine neue Spalte erstellen, in welcher wir wie bei einer mathematischen Berechnung die **Bezeichnung des neuen Datenfeldes festlegen** und sodann **gefolgt von einem Doppelpunkt** weiterhin die **Verkettungsformel** definieren, welche angibt, aus welchen Textbestandteile das neue Datenfeld besteht.

Kundenname: [Name1] & [Name2]

&-Operator zur Verkettung der Textinhalte



Wie im Abfrageergebnis zu sehen ist, werden die Textinhalte der Felder **Name1** und **Name2** im neuen Feld **Kundenname** zusammengeführt:

Kd-Nr	Kundenname
7	AutohausBaluch GmbH
29	SportstudioZabel
98	SteuerberaterTernieden
121	HeilpraktikerAnton van t'Hof

Etwas unschön ist noch der Umstand, dass die beiden Textfelder ohne Leerzeichen direkt hintereinander verkettet werden. Daher fügen wir ein solches noch dazwischen:

Kundenname: [Name1] & " " & [Name2]

Leerzeichen als zusätzliches Textelement in Anführungszeichen zwischen Name1 und Name2

Natürlich bedarf es zur Verkettung eines zusätzlichen Textelementes auch eines weiteren **&**-Operators:

Feld:	Kd-Nr	Kundenname: [Name1] & " " & [Name2]
Tabelle:	t_Kunden	
Sortierung:		
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:		

Im Abfrageergebnis werden **Name1** und **Name2** nun durch ein **Leerzeichen** getrennt dargestellt:

Kd-Nr	Kundenname
7	Autohaus Baluch GmbH
29	Sportstudio Zabel
98	Steuerberater Ternieden
121	Heilpraktiker Anton van t'Hof

Auf diese Weise kann man alle möglichen Textelemente oder Textfelder in beliebiger Kombination miteinander verketteten.

Aufgabe:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgende Abfrage:

- Es soll eine Kundenliste erstellt werden, in welcher jeder Kunde mit seiner **Kd-Nr**, seinem **vollständigen Namen** (bestehend aus Name1 und Name 2 – getrennt durch ein Leerzeichen), seiner **Straßenanschrift** sowie seinem **Wohnort** (bestehend aus PLZ und Ortsbezeichnung – getrennt durch ein Leerzeichen) angezeigt wird. Der Postleitzahl soll zudem das Länderkennzeichen **D** gefolgt von einem **Bindestrich** vorangestellt werden. (Beispiel: D-55743 Idar-Oberstein). Betrachten Sie das Abfrageergebnis und speichern Sie die Abfrage als „q_Abfrage32“.

4.7 Abfragen mit Funktionen

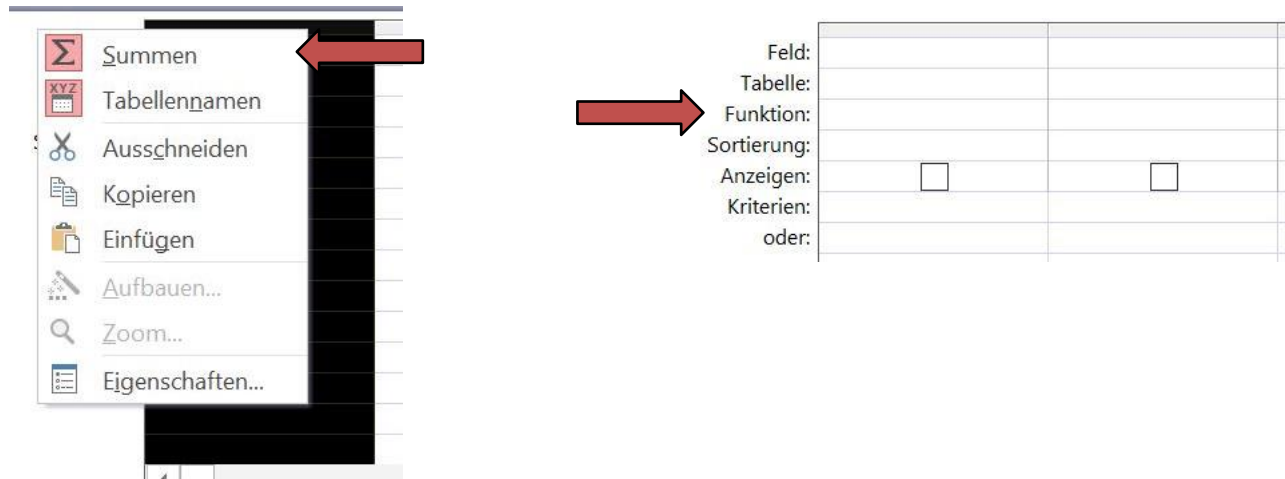
Wie auch in Excel kann man sich in Access verschiedener **Funktionen zur Durchführung von Standardberechnungen** bedienen - wie z. Bsp. zur Ermittlung einer **Summe**, einer bestimmten **Anzahl**, oder des **Mittelwertes** über eine bestimmte Spalte bzw. über ein bestimmtes Datenfeld. Diesbezüglich stehen zwei verschiedene Möglichkeiten offen, um dies zu bewerkstelligen:

1. Die **Einrichtung einer Funktion per Drop-Down Menü** oder
2. Die **manuelle Formulierung der Syntax einer Funktion**.

Beide Möglichkeiten wollen wir nachfolgend anhand einer Beispielaufgabe erörtern, bei welcher die **Anzahl der in der Kundenliste befindlichen Kunden** zu **ermitteln** ist. Hierfür müssen wir zunächst eine neue Abfrage erstellen, für welche wir die Tabelle **t_Kunden** auswählen müssen.

4.7.1 Einrichtung einer Funktion per Drop-Down Menü

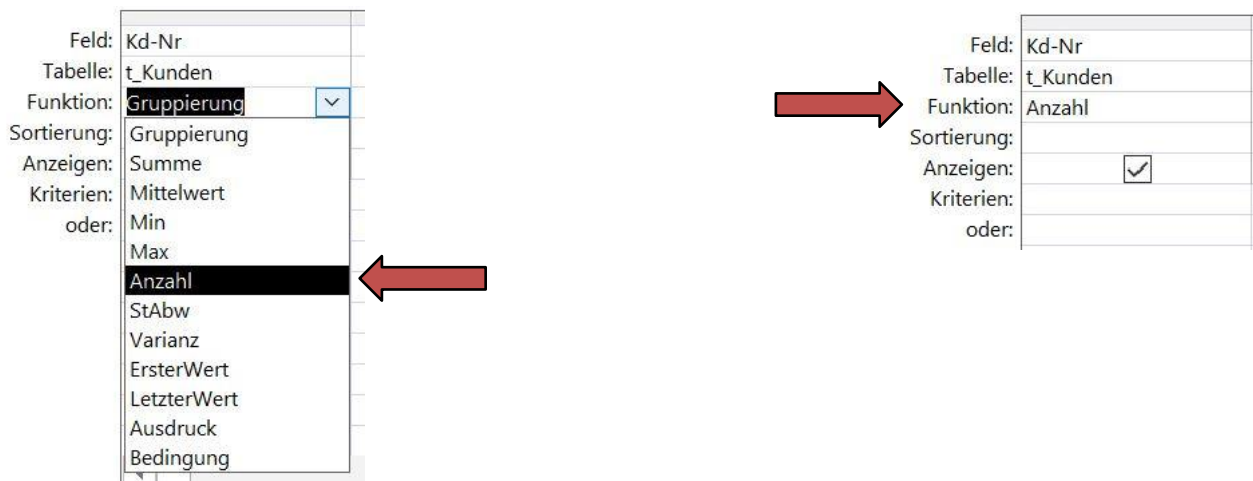
Um die Möglichkeit der Funktionsauswahl per Drop-Down Menü einzurichten, muss im QBE-Bereich der Abfrage zunächst die Zeile **Funktion** eingeblendet werden. Klicken Sie hierzu mit der **rechten Maustaste** auf einen beliebigen **Spaltenkopf** und aktivieren Sie im daraufhin erscheinenden Kontextmenü den Menüpunkt **Summen**, woraufhin im QBE-Bereich eine neue Zeile namens **Funktion** eingeblendet wird:



Zur Ermittlung der Anzahl an Kunden, genügt uns irgendein Datenfeld aus der Tabelle **t_Kunden**. So können wir hierfür z. Bsp. das Datenfeld **Kd-Nr** in den QBE-Bereich übernehmen. Wie Sie sehen, wird nun in der Zeile **Funktion** zunächst die Funktion **Gruppierung** angezeigt, mit welcher wir uns aber erst im nächsten Kapitel näher auseinander setzen werden:



Klicken Sie nun mit der Maustaste in der Zeile **Funktion** den **rechten Spaltenrand** des Feldes **Kd-Nr** an, um das Drop-Down Menü zur Auswahl der gewünschten Funktion aufzurufen und wählen Sie aus dem Drop-Down Menü die Funktion **Anzahl** aus:



Die Funktion **Anzahl** ermittelt die **Anzahl aller Datenzeilen** des regulären Abfrageergebnisses, welches ohne die Verwendung der Funktion zustande käme. Unsere Kundentabelle beinhaltet insgesamt 20 Kunden, welche ohne die Verwendung der Funktion Anzahl allesamt angezeigt würden. Die Funktion **Anzahl** hingegen **zählt diese zusammen** und gibt das **Funktionsergebnis** als **Abfrageergebnis** aus:



Wie Sie anhand des obigen Drop-Down Menüs ansehen können, stehen in Access einige Funktionen zur Auswahl, wobei die wichtigsten sind:

Funktion	Funktionsweise
Summe	Gibt die Summe der Werte zurück, die in einem bestimmten Feld einer Abfrage enthalten sind
Mittelwert	Berechnet den arithmetischen Durchschnitt von Werten, die in einem bestimmten Feld einer Abfrage enthalten sind
Min	Gibt den kleinsten Wert derjenigen Werte zurück, die in einem bestimmten Feld einer Abfrage enthalten sind
Max	Gibt den größten Wert derjenigen Werte zurück, die in einem bestimmten Feld einer Abfrage enthalten sind
Anzahl	Zählt die Anzahl von Elementen, die von einer Abfrage zurück gegeben werden
ErsterWert	Gibt den ersten Wert bzw. den Wert des ersten Datensatzes derjenigen Werte zurück, die in einem bestimmten Feld einer Abfrage enthalten sind
LetzterWert	Gibt den letzten Wert bzw. den Wert des letzten Datensatzes derjenigen Werte zurück, die in einem bestimmten Feld einer Abfrage enthalten sind

4.7.2 Manuelle Formulierung der Syntax einer Funktion

Eine weitere Möglichkeit zur Verwendung von Funktionen besteht in der **manuellen Formulierung der jeweiligen Funktionssyntax**. Wie bei einer Berechnung ist in diesem Fall ein **neues Datenfeld** bzw. eine **neue Spalte** im QBE-Bereich der Abfrage einzurichten und die **Bezeichnung des entsprechenden Datenfelds bzw. der Spalte mitsamt der** entsprechenden **Funktionssyntax** in einer **vorgeschriebenen Form** zu definieren.

Hinsichtlich der **Definition des entsprechenden Datenfeldes** sowie der Funktionssyntax sind dabei folgende **Regeln** zu beachten:

1. Die **Definition** des neuen Datenfeldes erfolgt in der Zeile Feld der betreffenden Spalte.
2. Der **Name des Datenfeldes** ist **als erstes** in die Zeile Feld **einzutragen**.
3. **Dem Namen** des Datenfeldes **folgt** nicht etwa ein Gleichheitszeichen, sondern **ein Doppelpunkt!**
4. Nach dem Doppelpunkt ist sodann der **Funktionsname** anzugeben.
5. Dem Funktionsnamen folgt wie bei Excel eine **runde Klammer**, innerhalb derer die **Funktionsargumente** anzugeben sind (i.d.R ist dies entweder ein Datenfeld oder eine Berechnung aus mehreren Datenfeldern, wobei **ein jedes verwendete Datenfeld** von **eckigen Klammern** zu umschließen ist).

Der daraus resultierende Ausdruck ergibt sich somit z. Bsp. wie folgt:

4. Funktionsname
↓
2. Name des neuen Datenfelds → **Kundenanzahl:Anzahl([Kd-Nr])** ← 5. Funktionsargumente in runden Klammern – darin verwendete Datenfelder in eckigen Klammern.
↑
3. Doppelpunkt anstelle eines Gleichheitszeichens

Für unsere aus dem vorherigen Unterkapitel bereits bekannte Beispielaufgabe würde die entsprechende Funktionsdefinition wie folgt aussehen:

Feld:	Kundenanzahl: Anzahl([Kd-Nr])
Tabelle:	
Sortierung:	
Anzeigen:	<input checked="" type="checkbox"/>
Kriterien:	
oder:	

... und hier das Abfrageergebnis:

Kundenanzahl
20

4.7.3 Verwendung von Funktionen mit weiteren Datenfeldern

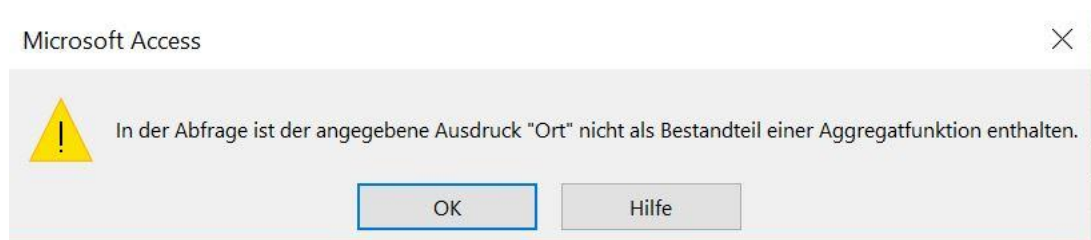
Funktionen können natürlich auch **in Verbindung mit weiteren Datenfeldern** (z. Bsp. zur Einbeziehung von Kriterien) verwendet werden, wobei sich in diesem Fall eine besondere **Problematik** ergibt:

Beispielaufgabe: Es soll die Anzahl an Kunden aus Gronau und Coesfeld ermittelt werden.

Hierfür benötigen wir ergänzend zu unserer bisher erstellten Abfrage noch die Tabelle **t_Orte**, wobei wir aus der Tabelle **t_Orte** das Datenfeld **Ort** benötigen, um das Abfrageergebnis nach den Kunden aus **Gronau oder Coesfeld** zu filtern:

Feld:	Kundenanzahl: Anzahl([Kd-Nr])	Ort
Tabelle:		t_Orte
Sortierung:		
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:		= "Gronau" Oder "Coesfeld"
oder:		

Wenn wir nun jedoch das Abfrageergebnis aufrufen, dann erhalten wir folgende Fehlermeldung:




Um den Konflikt aufzulösen, müssen wir die **Anzeige** des Datenfelds **Ort** daher aus der Abfrage **ausblenden** – d.h. das Datenfeld ist zwar für die Angabe der Kriterien bzw. für die entsprechende Filterung des Abfrageergebnisses erforderlich, es darf jedoch nicht angezeigt werden:

Feld:	Kundenanzahl: Anzahl([Kd-Nr])	Ort
Tabelle:		t_Orte
Sortierung:		
Anzeigen:	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Kriterien:		= "Gronau" Oder "Coesfeld"
oder:		

Ein vergleichbares Problem ergibt sich auch dann, wenn die verwendete Funktion per Drop-Down Menü ausgewählt wird. Beachten Sie, dass hier nun auch für das Datenfeld **Ort** eine Funktion angezeigt wird – in diesem Fall die Funktion „Gruppierung“, welche jedoch in Konflikt mit der Funktion „Anzahl“ über das Datenfeld **Kd-Nr** steht:

Feld:	Kd-Nr	Ort
Tabelle:	t_Kunden	t_Orte
Funktion:	Anzahl	Gruppierung
Sortierung:		
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:		= "Gronau" Oder "Coesfeld"
oder:		



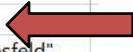
Das Abfrageergebnis ist dementsprechend fehlerhaft. Genau genommen wird es in diesem Fall nach den Orten Gronau und Coesfeld gruppiert:

Anzahl von Kd-Nr	Ort
3	Coesfeld
4	Gronau

Dies ist jedoch eigentlich nicht erwünscht, da wir einfach nur die Gesamtanzahl der Kunden aus Gronau und Coesfeld als **eine Zahl** ausgegeben haben möchten!

In diesem Fall ist es jedoch **noch nicht ausreichend**, das Datenfeld **Ort** zwar für die Angabe der Kriterien zu verwenden, es jedoch aus der Anzeige des Abfrageergebnisses einfach **auszublenden**. Dies alleine würde immer noch ein gruppiertes Abfrageergebnis zur Folge haben, da die Funktion „Gruppierung“ auch dann wirksam wird, wenn das Datenfeld **Ort** in der Ergebnisanzeige ausgeblendet ist. Darüber hinaus muss außerdem auch die für das Datenfeld **Ort** ausgewählte **Funktion** in entweder „**Ausdruck**“ oder „**Bedingung**“ geändert werden:

Feld:	Kd-Nr	Ort
Tabelle:	t_Kunden	t_Orte
Funktion:	Anzahl	Ausdruck
Sortierung:		
Anzeigen:	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Kriterien:		= "Gronau" Oder "Coesfeld"
oder:		

Das Resultat ist sodann ein nicht-gruppiertes Abfrageergebnis:

Anzahl von Kd-Nr
7

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

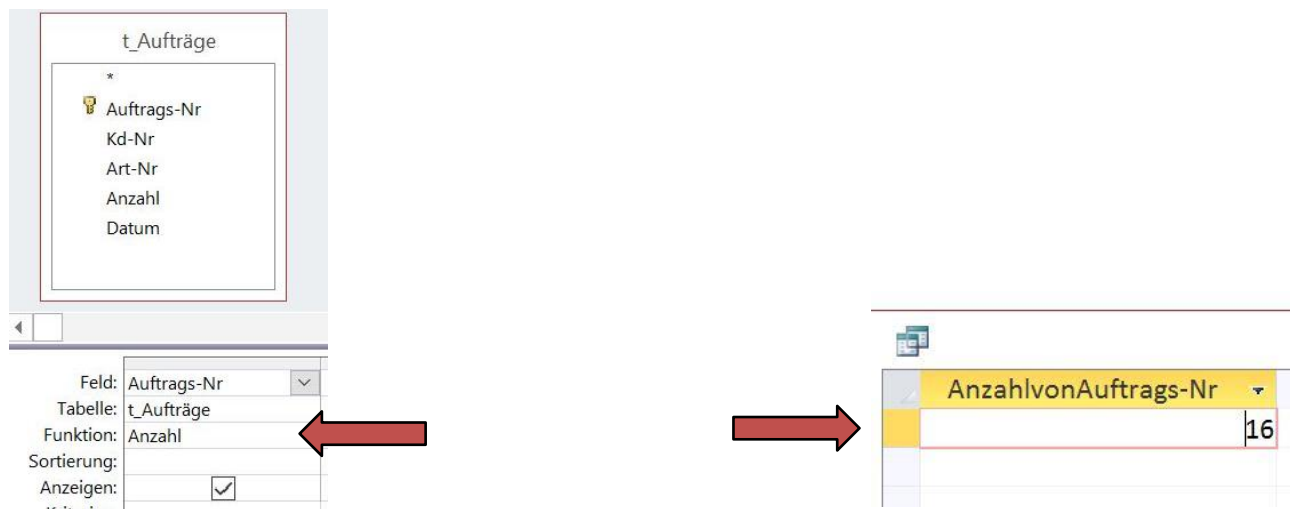
1. Wie hoch ist der durchschnittliche Artikelpreis der im Sortiment befindlichen Artikel. Formatieren Sie die Ergebnisse in Euro und speichern Sie die Abfrage als „q_Abfrage33“.
2. Ermitteln Sie die Anzahl der Kunden, die nicht in Ahaus wohnen. Speichern Sie die Abfrage als „q_Abfrage34“.
3. Wie viele Artikel befinden sich derzeit nicht an Lager? Speichern Sie die Abfrage als „q_Abfrage35“.
4. Ermitteln Sie den Gesamtumsatz (aus Bestellmenge mal Artikelpreis) aller bisherigen Bestellungen. Formatieren Sie das Ergebnis in Euro und speichern Sie die Abfrage als „q_Abfrage36“.
5. Ermitteln Sie aus den bisherigen Bestellungen den Höchstumsatz, der bei einer Bestellung erzielt wurde. Formatieren Sie das Ergebnis in Euro und speichern Sie die Abfrage als „q_Abfrage37“.
6. Welchen Wert hat der aktuelle Lagerbestand? Formatieren Sie das Ergebnis in Euro und speichern Sie die Abfrage als „q_Abfrage38“.
7. Wie viele Bestellungen haben wir von Kunden aus Ahaus erhalten? Speichern Sie die Abfrage als „q_Abfrage39“.
8. Welchen Umsatz haben wir an den Bestellungen von Kunden aus Coesfeld durchschnittlich erzielt? Formatieren Sie das Ergebnis in Euro und speichern Sie die Abfrage als „q_Abfrage40“.

4.8 Abfrageergebnisse gruppieren

Zuweilen kommt es vor, dass man **Abfrageergebnisse nach bestimmten Merkmalen bzw. nach bestimmten Datensätzen gruppiert** angezeigt bekommen möchte – so z. Bsp. die Frage, von welchen Kunden wir wie viele Bestellungen erhalten haben.

Eine solche Aufgabe lässt sich unter Verwendung einer speziellen Funktion lösen, die Sie im vorherigen Kapitel bereits ansatzweise kennen gelernt haben – nämlich mit Hilfe der Funktion **Gruppierung**.

Um einfach nur die Anzahl aller Bestellungen abzufragen, würden wir lediglich eine Abfrage auf ein beliebiges Datenfeld aus der Tabelle **t_Aufträge** benötigen. Nehmen wir hierfür z. Bsp. das Datenfeld **Auftrags-Nr.** Bei einer einfachen Abfrage würden wir allerdings lediglich eine Auflistung aller Auftragsnummern erhalten. Um stattdessen jedoch die Anzahl aller Aufträge angezeigt zu bekommen, muss auf das Datenfeld noch die Funktion **Anzahl** angewendet werden:



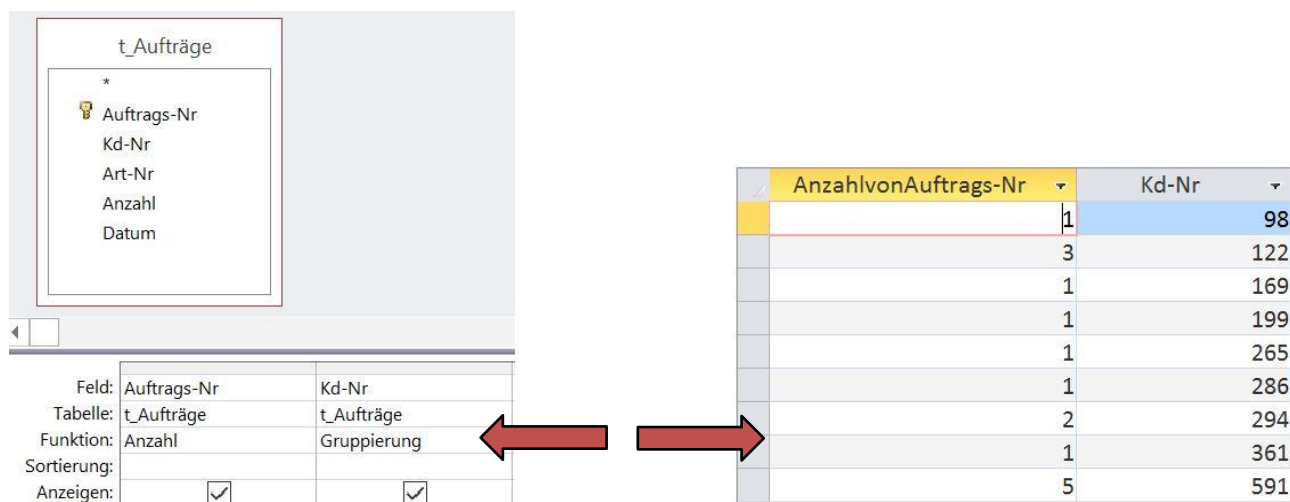
The screenshot shows the 't_Aufträge' table with fields: Auftrags-Nr, Kd-Nr, Art-Nr, Anzahl, and Datum. Below the table, the query design grid is shown with the following settings:

Feld:	Auftrags-Nr
Tabelle:	t_Aufträge
Funktion:	Anzahl
Sortierung:	
Anzeigen:	<input checked="" type="checkbox"/>

Red arrows point from the 'Feld:' and 'Funktion:' rows to the resulting query result table on the right.

Anzahl von Auftrags-Nr
16

Nun wollen wir jedoch nicht einfach nur die Gesamtanzahl aller Aufträge sehen, sondern wir wollen diese **nach Kunden gruppiert** angezeigt bekommen, so dass wir sehen können, welche Kunden jeweils wie viele Aufträge erteilt haben. Hierzu müssen wir zusätzlich noch das Datenfeld **Kd-Nr** in die Abfrage einfügen und auf dieses die Funktion **Gruppierung** anwenden:



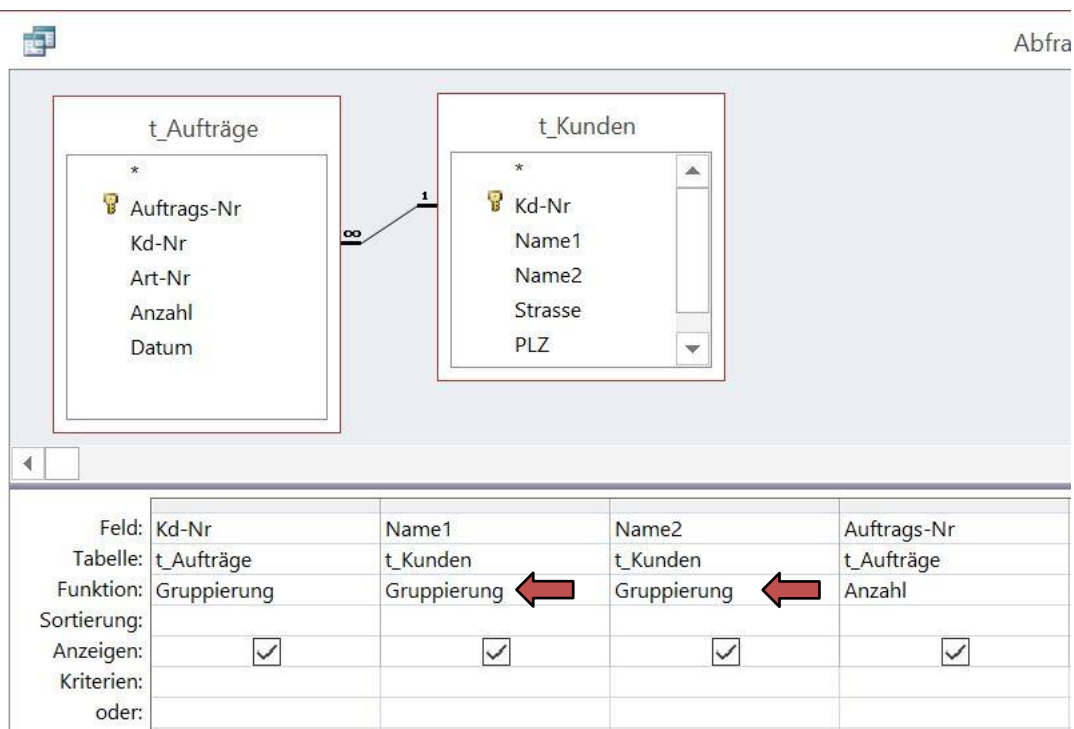
The screenshot shows the 't_Aufträge' table with fields: Auftrags-Nr, Kd-Nr, Art-Nr, Anzahl, and Datum. Below the table, the query design grid is shown with the following settings:

Feld:	Auftrags-Nr	Kd-Nr
Tabelle:	t_Aufträge	t_Aufträge
Funktion:	Anzahl	Gruppierung
Sortierung:		
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Red arrows point from the 'Feld:' and 'Funktion:' rows to the resulting query result table on the right.

Anzahl von Auftrags-Nr	Kd-Nr
1	98
3	122
1	169
1	199
1	265
1	286
2	294
1	361
5	591

Das war im Grunde genommen schon alles – jedoch bleibt das Abfrageergebnis natürlich ziemlich abstrakt und wenig aussagekräftig, wenn die einzelnen Kunden lediglich mit ihrer Kundennummer angezeigt werden. Optimieren wir unsere Abfrage daher noch ein wenig, indem wir die Abfrage noch um die Datenfelder **Name1** und **Name2** aus der Tabelle **t_Kunden** ergänzen. Um die Gruppierungsfunktion nicht zu verletzen, muss natürlich **auch auf diese Datenfelder** die Funktion **Gruppierung** angewendet werden, was jedoch kein Problem darstellt, da die Datenfelder **Kd-Nr**, **Name1** und **Name2** für einen jeden Kunden im entsprechenden Kundendatensatz zueinander zugehörig sind und somit jeweils ein gemeinsames **Tupel** bilden – d.h. die Anzahl der erteilten Aufträge wird dabei weiterhin nach unseren Kunden gruppiert:



Abfrage1

Kd-Nr	Name1	Name2	Anzahl von Auftrags-Nr
98	Steuerberater	Ternieden	1
122	Rechtsanwalt	Klocke	3
169	Tanzschule	Pellinger	1
199	Kunstschmiede	Wegener	1
265	Malermeister	Minarzik	1
286	Partnervermittlung	Netta	1
294	Restaurant	Zur Wilden Sau	2
361	Fitness-Center	Geisler	1
591	Kaufmännische Schule	Ahaus	5

Aufgaben:

Öffnen Sie die Datei [Computerladen.accdb](#) und erstellen Sie in dieser die folgenden Abfragen:

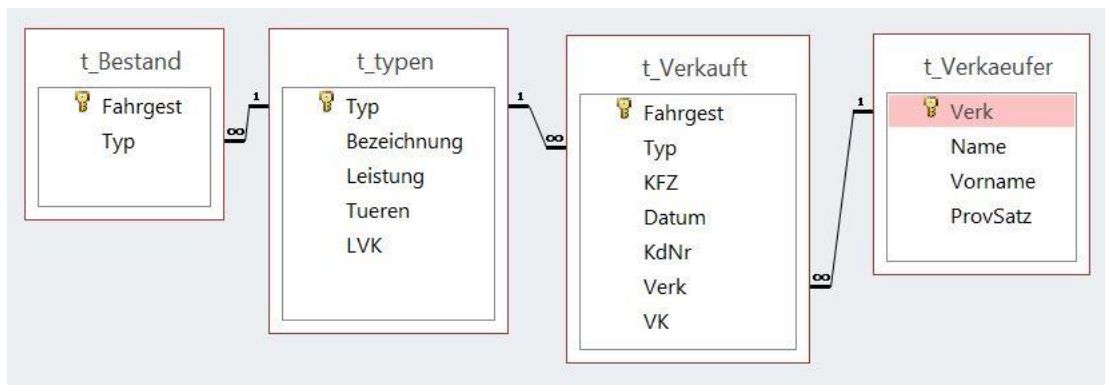
1. Erstellen Sie die Abfrage aus dem vorherigen Beispiel: Welche Kunden haben jeweils wie viele Aufträge erteilt. Angezeigt werden soll die Kd-Nr, Name1 und Name2 des jeweiligen Kunden sowie die Anzahl der vom Kunden erteilten Aufträge. Speichern Sie die Abfrage als „q_Abfrage41“.
2. Erstellen Sie eine Abfrage, aus der hervorgeht, wie viele Aufträge wir aus welchem Ort jeweils erhalten haben. Speichern Sie die Abfrage als „q_Abfrage42“.
3. Erstellen Sie eine Abfrage, aus der hervorgeht, wie viele Artikel aus welchem Ort jeweils bestellt wurden. Tipp: Diese Aufgabe ist ähnlich zur vorherigen, allerdings müssen Sie aus der Tabelle t_Aufträge hier ein anderes Datenfeld abfragen und auf dieses auch eine andere Funktion anwenden. Beschriften Sie das Datenfeld, aus welcher die jeweilige Artikelanzahl für jeden Ort hervorgeht mit „Summe Artikel“ und speichern Sie die Abfrage als „q_Abfrage43“.
4. Wie hoch ist der Auftragswert je Kunde? Tipp: Bei dieser Abfrage müssen Sie außerdem auch eine Berechnung durchführen, um zunächst erst mal den Auftragswert für einen jeden Auftrag zu berechnen. Formatieren Sie das Ergebnis in Euro und speichern Sie die Abfrage als „q_Abfrage44“.
5. Wie viele Kunden wohnen in den einzelnen Orten? Speichern Sie die Abfrage als „q_Abfrage45“.

4.9 Abschlussübung

Um Ihr Wissen und Können zu festigen, können Sie nun noch eine Abschlussübung machen. Öffnen Sie hierzu die Datenbank [Ü03 Autohaus.accdb](#)

Wenn Sie Abfragen auf eine Ihnen bislang noch unbekannte Datenbank durchführen müssen, ist es sinnvoll, die betreffende **Datenbank zunächst** erst mal hinsichtlich ihrer Datenstruktur und ihrer Dateninhalte zu **analysieren**. Am besten ruft man hierzu als erstes die Beziehungsansicht der Datenbank auf:

Die Autohaus GmbH verwaltet in ihrer Datenbank die Daten der im Angebot befindlichen PKWs (t_typen und t_Bestand), der erfolgten Verkaufsvorgänge (t_Verkauft) sowie der Verkäufer (t_Verkaeuer). Folgende Relationen liegen dabei zwischen den verschiedenen Tabellen vor:



Zudem macht es weiterhin auch Sinn, mal einen Blick in die einzelnen Tabellen hinein zu werfen, um zu sehen, welche Daten sich in diesen **konkret** befinden und mit welcher Art von Daten die einzelnen Datenfelder jeweils befüllt sind. Rufen Sie die **Tabellen** daher nicht nur in der **Datenblattansicht** auf, sondern schauen Sie sich zudem auch jeweils die **Entwurfsansicht** einer jeden Tabelle an, um ersehen zu können, von welchem Datentyp ein jedes Datenfeld ist.

Bei der Erstellung der einzelnen Abfragen, ist es sodann hilfreich, die folgenden Leitfragen in der angegebenen Reihenfolge heran zu ziehen:

1. Welche Tabellen werden zwingend gebraucht bzw. in welchen Tabellen finde ich die benötigten Daten?
2. Müssen zudem auch noch weitere Tabellen in die Abfrage eingefügt werden, über welche die Tabellen aus (1) miteinander verknüpft sind?
3. Welche Datenfelder sollen in der Abfrage konkret angezeigt werden?
4. Soll das Abfrageergebnis nach irgendwelchen Datenfeldern sortiert werden? Wenn ja, nach welchen Datenfeldern in welcher Reihenfolge?
5. Sollen im Abfrageergebnis nur bestimmte Datensätze angezeigt werden? Wenn ja, nach welchen Kriterien?
6. Müssen bestimmte Datenfelder erst durch entsprechende Berechnungen erzeugt werden? Wenn ja, welche Berechnungen sind hierzu erforderlich?
7. Müssen bestimmte Funktionen verwendet werden? Wenn ja, welche?

Aufgaben:

q_Abfrage1:

Ermitteln Sie den aktuellen Lagerbestand je Autotyp!

anzuzeigende Datenfelder: **Typ** **Bestand**

q_Abfrage2:

Ermitteln Sie den Lagerwert zum Listenverkaufspreis (LVK)!

anzuzeigende Datenfelder: **Wert**

q_Abfrage3:

Stellen Sie fest, welche PKW im November 2002 verkauft wurden.

anzuzeigende Datenfelder: **Typ** **Bezeichnung** **Datum** **LVK** **VK**

q_Abfrage4:

Welche PKW sind derzeit auf Lager, deren Listenverkaufspreis (LVK) 30.000 unterschreitet.

anzuzeigende Datenfelder: **Typ** **Bezeichnung** **LVK**

q_Abfrage5:

Stellen Sie die Abweichungen zwischen den erzielten Verkaufspreisen (VK) und den Listenverkaufspreisen (LVK) je Autotyp fest.

Datenfelder: **Typ** **Bezeichnung** **Summe von VK** **Summe von LVK** **Summe Abweichung**

q_Abfrage6:

Stellen Sie fest, welche der verkauften PKW im Kreis Coesfeld zugelassen wurden!

anzuzeigende Datenfelder: **Kfz** **Datum** **KDNR**

q_Abfrage7:

Stellen Sie fest, welche Einzelumsätze die Verkäufer zu LVK und VK erzielt haben sowie an Provisionen erhalten werden. Die Darstellung soll nach dem Namen des Verkäufers geordnet sein.

anzuzeigende Datenfelder: **Name** **Summe von VK** **Summe von LVK** **Provision**

q_Abfrage8:

Lassen Sie sich für jeden Verkäufer die Summe seiner Provision anzeigen. Die Darstellung soll nach der Höhe der Provision (höchster Wert zuerst) sortiert sein.

anzuzeigende Datenfelder: **Name** **Provision**