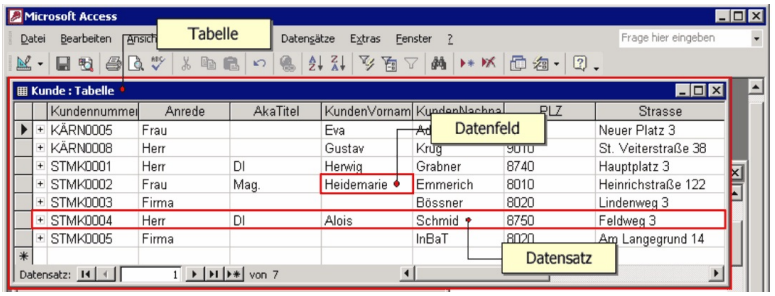
Definition: "Eine **Datenbank** ist eine **Sammlung von Informationen** zu einem bestimmten Thema oder Zweck."

Man sich eine Datenbank als einzelne Datei oder Gruppe von Dateien vorstellen. Datenbankdateien enthalten Datensätze zu einem bestimmten Thema sowie Objekte, die zur Datenein- und -ausgabe benötigt werden. Verschiedene Werkzeuge wie Formulare oder Abfragen ermöglichen ein effizientes Erfassen und ein rasches wieder Auffinden von Daten.

Zu den Vorläufern eines modernen Datenbankprogramms zählt der **Karteikasten** (Behälter oder Schrank zum geordneten Ablegen von Karteikarten). Die Weiterentwicklung derartiger Ablagesysteme und deren Integration in die elektronische Datenverarbeitung hat das Verwalten von Informationen enorm verbessert. In der heutigen Welt der Computer schreibt man jedoch kaum noch Informationen auf Karteikarten, sondern man verwendet dafür **Datenbankmanagementsysteme** (DBMS).

Tabelle

Die wichtigste Entität (Objekt) einer relationalen Datenbank ist die Tabelle. Sie beschreibt Objekte der realen Welt, die Tabellenspalten nehmen die einzelnen Eigenschaften (Attribute) dieses Objekts auf. Aus der Sicht eines DBMS entspricht eine Tabelle einem ganzen Karteikasten. Sie zeigt also nicht nur die Informationen einer einzelnen Karteikarte, sondern alle zum Karteikasten gehörenden Karten.



Diese Tabellen bestehen aus Zeilen und Spalten, wobei eine Zeile als **Datensatz** und der Schnittpunkt einer Zeile und einer Spalte als **Datenfeld** (kleinste Informationseinheit innerhalb eines Datensatzes) bezeichnet wird.

Datentypen

Für jedes Datenfeld muss ein bestimmter Speicherplatz auf dem Datenträger vorgesehen werden. Da die Speicherung von unterschiedlichen Daten (z.B. Text und Zahl) auf gänzlich verschiedenen Methoden beruht, muss der Datentyp bereits bei der Planung festgelegt werden. Außerdem sollte die maximale Größe des Datenfelds bestimmt werden.

Die wichtigsten Felddatentypen in Access:

Text	Text oder Kombinationen aus Text und Zahlen.	Bis zu 255 Zeichen
Memo	Langer Text oder Kombinationen aus Text und Zahlen..	Bis zu 65,535 Zeichen
Datum/ Uhrzeit	Datums- und Zeitwerte für die Jahre 100 bis 9999.	8 Bytes
AutoWert	Eine eindeutige, automatisch zugewiesene, fortlaufende Zahl (die jeweils um 1 hochgezählt wird) oder eine Zufallszahl	4 Bytes
Ja/Nein	Ja/Nein (Häkchen oder leer)	1 Bit
Zahl	Byte Zahlen von 0 bis 255 Integer Zahlen von -32.768 bis 32.767 Long Integer Zahlen von -2.147.483.648 bis 2.147.483.647 Single –3.4E38 bis –1.4E–45 für negative Werte 1.4E–45 bis 3.4E38 für positive Werte (auf 7 Stellen genau) Double –1.79E308 bis –4.94E–324 für negative Werte 4.94E–324 bis 1.79E308 für positive Werte (auf 15 Stellen) Dezimal –10^38–1 bis 10^38–1 (auf 28 Stellen genau)	1 Byte 2 Bytes 4 Bytes 4 Bytes 8 Bytes 12 Bytes
Währung	Währungswerte auf bis zu 15 Stellen links, und bis zu 4 Stellen rechts vom Dezimaltrennzeichen genau	8 Bytes

Datenbankentwurf

Die Realisierung einer Datenbank ist eine relativ komplexe und arbeitsintensive Aufgabe. Deshalb sollten vor der Realisierung am Computer die Struktur und der Aufbau der Datenbank genau festgelegt werden.

Normalisierung der Daten

Die gesammelten, meist unstrukturiert vorliegenden Daten werden in eine Form gebracht, mit der sie in der Datenbank abgebildet werden können. Ziel ist es, Redundanzen zu vermeiden.

Redundanzen

Kommen ein- und dieselben Informationen mehrfach in einer Datenbank vor, so spricht man von redundanten Informationen. Redundanzen sind meist unerwünscht, da diese zusätzlichen Speicherbedarf verursachen. Darüber hinaus gestaltet sich die Wartung von redundanten Informationen meist schwierig. In der Datenbankpraxis werden Redundanzen dadurch beseitigt, dass die mehrfach vorkommenden Informationen auf zwei oder mehrere Tabellen verteilt werden (Normalisierung).

Beispiel:

Betreut Thomas Maier im Jahr etwa 30 Kurse, dann müssen seine Daten 30-mal eingegeben werden. Diese Redundanzen sollten vermieden werden. Die Daten werden auf zwei Tabellen aufgeteilt.

Trainer-Nr.	Vorname	Nachname	Kurs-Nr.	Bezeichnung	Ort
1001	Thomas	Maier	TA0231	Tennis Anfänger	Halle 1
1001	Thomas	Maier	TP0712	Tennis für Profis	Halle 2
1002	Doris	Wegerer	SF0462	Surfen Fortgeschrittene	See
1002	Doris	Wegerer	SA0126	Surfen Anfänger	See
1003	Klaus	Müller	GA0564	Golf Anfänger	Green

Trainer

Trainer-Nr.	Vorname	Nachname
1001	Thomas	Maier
1002	Doris	Wegerer
1003	Klaus	Müller

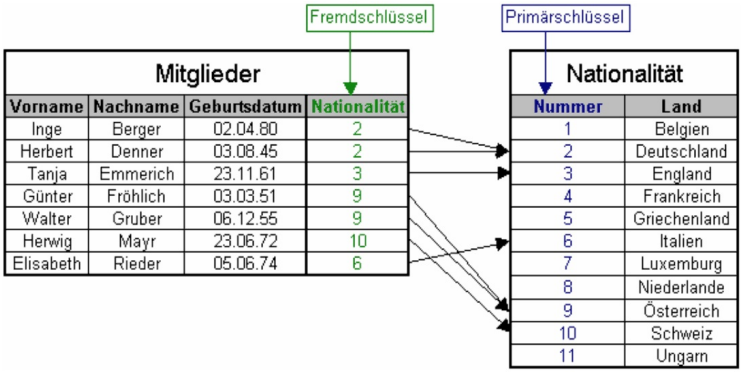
Kurse

Kurs-Nr.	Bezeichnung	Ort	Trainer-Nr.
TA0231	Tennis Anfänger	Halle 1	1001
TP0712	Tennis für Profis	Halle 2	1001
SF0462	Surfen Fortgeschrittene	See	1002
SA0126	Surfen Anfänger	See	1002
GA0564	Golf Anfänger Green	1003	1003

Primär- und Fremdschlüssel

Relationen (Beziehungen zwischen Tabellen) werden in **relationalen Datenbanken** über so genannte Schlüssel hergestellt. Möchte man so eine Beziehung erstellen, müssen alle Datensätze eindeutig identifizierbar sein. Diese eindeutige Kennzeichnung ist mit dem so genannten Primärschlüssel möglich.
Man unterscheidet zwischen zwei Schlüssel-Typen:

- Der **Primärschlüssel** besteht aus einem oder mehreren Feldern, die jeden Datensatz eindeutig kennzeichnen (Bsp: SVNr., Kundennr., Autokennzeichen, ...).
- Der **Fremdschlüssel** ist ein Tabellenfeld, das auf den Primärschlüssel einer anderen Tabelle verweist (muss nicht eindeutig sein).



Indizes

Normalerweise sind die Datensätze einer Tabelle ungeordnet, d. h., neue Datensätze werden jeweils am Ende der betreffenden Tabelle angehängt. Möchte man nun nach einem bestimmten Kriterium sortieren oder einen bestimmten Eintrag suchen, müssen alle vorhanden Datensätze durchlaufen werden. Diese Vorgangsweise ist recht langsam.
Indizes beschleunigen das Sortieren von Datensätzen. Der Einsatz von Indizes lohnt sich vor allem bei Tabellen mit vielen Datensätzen.
Nachteile:

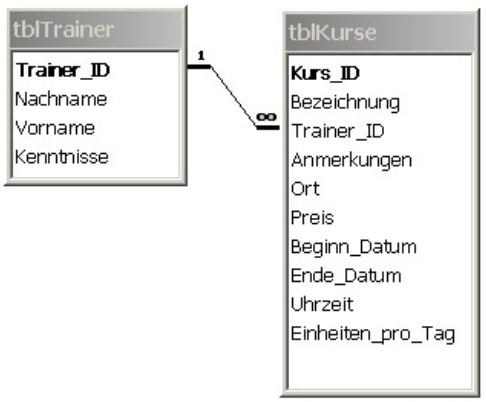
- Das Einfügen von Datensätzen erfordert nicht nur eine Änderung der Tabelle, sondern auch die Änderung aller vorhandenen Indizes (Bearbeitung kann so länger dauern).
- Jeder Index stellt eine redundante (überflüssige) Information dar, d.h. die Datenbank benötigt mehr Speicher.

Relationen (Beziehungen)

Im Schritt der Normalisierung wurden die Daten auf mehrere Tabellen aufgeteilt. Damit zusammengehörige Daten logisch wieder zusammengefügt werden können, müssen zwischen den Tabellen Beziehungen definiert werden. Bei der Verknüpfung zweier Tabellen muss ein Datenfeld in beiden Tabellen vorhanden sein. Voraussetzung ist allerdings, dass die Datentypen beider Verknüpfungsfelder übereinstimmen.

Die 1:N-Relation

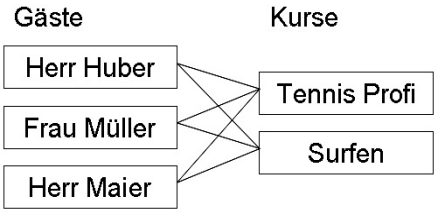
Die 1:N-Relation kommt in relationalen Datenbanken am häufigsten vor. Bei dieser Beziehungsart kann ein Datensatz aus der Haupttabelle mit beliebig vielen Datensätzen einer Detailtabelle verknüpft werden.
Bei der 1:N-Relation muss das Verbindungsfeld in der Haupttabelle als Primärschlüssel definiert sein. In der Detailtabelle darf das Feld nicht als Primärschlüssel festgelegt werden, es handelt sich hier um den Fremdschlüssel.



Bsp.: In der Beziehung zwischen der Tabelle "Trainer" (Haupttabelle) und der Tabelle "Kurse" (Detailtabelle) können einem Trainer mehrere Kurse zugeordnet werden.
Ein (1) Trainer kann mehrere (N) Kurse halten, aber ein Kurs wird immer nur von einem Trainer gehalten.

Die M:N-Relation

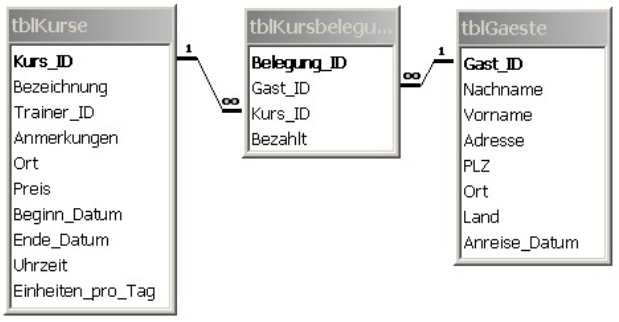
Bei der M:N-Relation können beliebig viele Datensätze einer Tabelle mit beliebig vielen Datensätzen aus einer anderen Tabelle in Beziehung stehen.



Ein Gast kann mehrere Kurse belegen und an einem Kurs werden meistens mehrere Gäste teilnehmen.

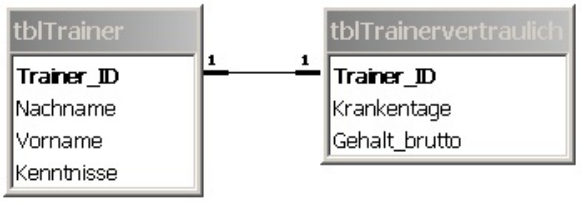
Damit M:N-Relationen abgebildet werden können, muss in den meisten Datenbankprogrammen eine weitere Tabelle eingefügt werden. Damit wird die M:N-Relation in zwei 1:N-Relationen aufgeteilt.

Bei obigem Beispiel ist es etwa sinnvoll, eine Tabelle mit den Kursbelegungen einzufügen. In dieser Tabelle können zudem weitere Daten erfasst werden.

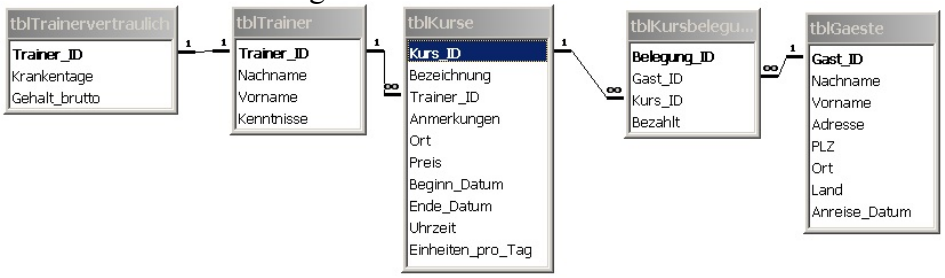


Die 1:1-Relation

Wird genau ein Datensatz einer Tabelle mit einem Datensatz einer anderen Tabelle verknüpft, dann spricht man von einer 1:1-Relation. In der Praxis ist diese Beziehungsart eher selten anzutreffen, da alle Daten meist ohnehin in einer einzigen Tabelle enthalten sein können. 1:1-Relationen machen etwa dann Sinn, wenn vertrauliche Informationen in einer separaten Tabelle gespeichert werden sollen.



Ein "fertiges" Datenmodell könnte etwa folgendermaßen aussehen:



Referentielle Integrität bei Beziehungen
Möchte man zusätzlich bestimmen, dass im Fremdschlüsselfeld einer Tabelle nur Werte eingetragen werden dürfen, die bereits im Primärschlüsselfeld der anderen Tabelle enthalten sind, so aktiviert man in der Beziehung die Option "Referentielle Integrität". Referentielle Integrität ist ein Regelsystem in relationalen Datenbanken, das die Gültigkeit von Beziehungen sicherstellt und das versehentliche Löschen von verknüpften Daten verhindert. Sie kann unter den folgenden Voraussetzungen festgelegt werden:

- Das übereinstimmende Feld aus der Mastertabelle ist ein Primärschlüssel oder hat einen eindeutigen Index.
- Die Detailfelder haben denselben Datentyp. Ausnahme: AutoWert - kann nur mit einem Feld des Datentyps Zahl verknüpft werden, dessen Eigenschaft Feldgröße auf Long Integer eingestellt ist.

Wenn das Kontrollkästchen der **Aktualisierungs**- oder **Löschweitergabe** aktiviert wird, sind Lösch- und Aktualisierungsoperationen zulässig, die normalerweise durch die Regeln der referentiellen Integrität verhindert werden.

Abfragen

Abfragen sind ein sehr mächtiges Werkzeug, um Datensätze aus einer Tabelle herauszufiltern. Um die Bedingungen zu definieren, welche Datensätze der Tabelle angezeigt werden sollen, können unterschiedliche Operatoren verwendet werden.

Abfragen können auch verwendet werden, um dynamisch Daten aus verschiedenen Tabellen zu verknüpfen. Es werden automatisch die Beziehungen zwischen mehreren Tabellen übernommen.

"Wien"	Alle Zeichenfolgen, die gleich "Wien" sind.
Wie "A*"	Alle Zeichenfolgen, die mit "A" beginnen.
Wie "?e*"	Alle Zeichenfolgen, die als zweiten Buchstaben ein "e" beinhalten.
"Wien" Oder "Graz"	Alle Zeichenfolgen gleich "Wien" oder "Graz".
> 100	Alle Zahlen, die größer als 100 sind.
>=10 Und <=20	Alle Zahlen zwischen 10 und 20.
Zwischen 10 Und 20	Alle Zahlen zwischen 10 und 20.
< > 0	Alle Einträge, die ungleich der Zahl 0 sind.
<#1.1.2000#	Jedes Datum vor dem 1. Jänner 2000
Ist Null	Leere Felder
Ist Nicht Null	Nicht-leere Felder
Nein	Häkchen nicht angehakt (bei Ja/Nein Datentyp)

Berechnete Felder in Abfragen
In Abfragen können auch berechnete Felder definiert werden. Durch das Ausführen der Abfrage werden sämtliche berechnete Werte neu ermittelt.
Zuerst wird der Name des neuen Feldes mit Doppelpunkt eingegeben. Im Anschluss kann der berechnete Ausdruck eingegeben werden. Bestehende Feldnamen werden von Access automatisch in eckige Klammern eingeschlossen.

Beispiele: Bruttowert: [Nettowert]*1,2
AnzahlTage: [AbreiseDatum]-[AnreiseDatum]+1
GanzerName: [Vorname]&" "&[Nachname]
Statistik: Wenn([Ort]="Amstetten";"Einheimischer";"Hinterweltler")

Parameterabfragen
Abfragen können mit Parametern flexibel gestaltet werden. Ändern sich bei einer Abfrage immer wieder bestimmte Kriterien, dann können sie als Parameter beim Ausführen der Abfrage eingegeben werden.

Ende_Datum	Bezahlt
tblKurse	tblKursbelegung
☑	☐
<[Kursende eingeben]	Nein

Um Eingaben in Abfragen als Parameter zu deklarieren, wird statt dem Kriterium ein Ausdruck in eckigen Klammern eingegeben.

Beim Öffnen der Abfrage wird zur Eingabe des Parameterwerts ein Dialogfenster eingeblendet.

Gruppierungen, Summen und andere Statistiken

- **Gruppierung**: Mit dieser Einstellung werden alle gleich lautenden Feldinhalte zusammengefasst und gruppiert.
- **Bedingung**: Ein Feld wird zur Selektion verwendet. (Dieses Feld wird in diesem Fall nicht angezeigt)
- **Statistische Werte**: Summe, Mittelwert, Anzahl, Min, Max, ...

Feld:	Ort	Anzahl von Gast_ID:	Land
Tabelle:	tblGaeste	tblGaeste	tblGaeste
Funktion:	Gruppierung	Anzahl	Bedingung
Sortierung:			
Anzeigen:	☑	☑	☐
Kriterien:			"A"

Bsp: Anzahl der Gäste aus Österreich, gruppiert nach dem Ort.

Übungen (Kursverwaltung.mdb)

Einfache Abfragen

1. Erstellen Sie eine Abfrage mit allen Gästen aus Deutschland. Die Auswertung soll die Felder "Vornamen", "Nachnamen" und "Land" enthalten und nach dem Nachnamen sortiert sein. Speichern Sie die Abfrage unter dem Namen *qryGaeste_aus_Deutschland*.
2. Erstellen Sie eine Abfrage aller Kurse, die in der Segelschule abgehalten werden. Die Abfrage soll die Felder "Kurs_ID", "Bezeichnung", "Beginn_Datum" und "Ende_Datum" enthalten und nach dem "Beginn_Datum" sortiert sein. (*qrySegelschule*)
3. Erstellen Sie eine Abfrage aller Gäste aus Linz mit den Feldern "Vorname", "Nachname" und "PLZ". Die Abfrage soll nach dem Nachnamen sortiert sein. (*qryGaeste_aus_Linz*)
4. Erstellen Sie eine Abfrage mit den Tenniskursen in der Vorsaison (also bis zum 30. 6. des Jahres. Die Abfrage soll die Felder "Kurs_ID", "Bezeichnung", "Trainer_ID", "Beginn_Datum" und Uhrzeit enthalten und nach dem "Beginn_Datum" aufsteigend sortiert sein. (*qryTennis_Vorsaison*)
5. Erweitern Sie diese Abfrage um die Golfkurse in diesem Zeitraum. (*qryTennis_und_Golf_Vorsaison*)
6. Erstellen Sie eine Abfrage aller Gäste aus Österreich oder Deutschland, die vor dem Juni dieses Jahres angereist sind. Die Abfrage soll die Felder "Nachname", "Vorname" und "Postleitzahl" enthalten und nach dem "Nachnamen" sortiert sein. (*qryGaeste_vor_Juni*)
7. Erweitern Sie diese Abfrage so, dass die Daten primär nach der "Postleitzahl" und sekundär nach dem "Nachnamen" sortiert sind. Die Anordnung der Felder soll jedoch unverändert bleiben. (*qryGaeste_nach_PLZ*)
8. Erstellen Sie eine Abfrage aller ausländischen Gäste (also nicht aus Österreich), die zwischen Anfang April und Ende Mai angereist sind. Die Abfrage soll die Felder "Nachname", "Vorname" und "Land" enthalten und primär nach dem "Land" und sekundär nach dem "Nachnamen" sortiert sein. (*qryAuslaendische_Gaeste*)
9. Erstellen Sie eine Abfrage aller Gäste mit den Feldern "Kurs_ID" und "Gast_ID", die Tenniskurse für Anfänger belegt haben. (*qryTennis_fuer_Anfaenger*)

Abfragen mit mehreren Tabellen

10. Erstellen Sie eine Abfrage aller Tenniskurse. Die Liste soll die Kursbezeichnung, den Vor- und Nachnamen sowie die Kenntnisse des Trainers enthalten. Die Abfrage soll nach der Kursbezeichnung sortiert sein und darf auf keinen Fall die "Kurs_ID" enthalten (*qryKurse_mit_Trainer*)
11. Erstellen Sie eine Abfrage aller Gäste, die Golfkurse belegt haben, mit den Feldern "Kurs_ID", "Vorname", "Nachname" und "Land". (*qryGolfkurse*)
12. Erstellen Sie eine Abfrage aller Tenniskurse und Wassersportkurse, die noch nicht bezahlt wurden, mit den Feldern "Gast_ID", "Kursbezeichnung", "Preis" und "Beginndatum". Schränken Sie die Abfrage zudem auf den Zeitraum Mai bis Juni 2005 ein. Die Abfrage soll nach dem "Beginndatum" und der "Kursbezeichnung" sortiert sein. (*qryNicht_bezahlt*)

Berechnete Felder in Abfragen

13. Erweitern Sie die Tabelle "tblTrainer" um die Felder "Jahresgehalt" und Solleinheiten":

Feldname	Datentyp	Feldgröße/Format	Beschreibung
Jahresgehalt	Währung	Standardzahl	Gehalt inkl. Dienstgeberanteil
Solleinheiten	Zahl	Long Integer	Kurseinheiten pro Jahr

14. Ergänzen Sie die Tabelle "tblTrainer" mit folgenden Informationen:

Trainer_ID	Jahresgehalt	Solleinheiten
HUBER1	€ 40.000,00	1200
MAIER1	€ 25.000,00	1200
MAIER2	€ 20.000,00	1000

15. Erstellen Sie eine Abfrage mit folgenden Feldern: "Nachname", "Vorname", "Jahresgehalt", "Solleinheiten", "WochenSoll" und "Kosten_pro_Einheit". Das Feld "WochenSoll" berechnet die Kurseinheiten pro Woche bei 46 Jahreswochen. Das Feld "Kosten_pro_Einheit" berechnet die Kosten pro Einheit gemessen am Jahresgehalt und den Solleinheiten. (*qryTrainerkosten*)
16. In der Tabelle "tblKurse" sind lediglich die Kurseinheiten pro Tag enthalten. Für statistische Zwecke wird eine Auswertung benötigt, in der die Kursdauer in Tagen sowie die gesamte Anzahl der Kurseinheiten enthalten sind. Benötigte Felder: "Bezeichnung", "Trainer", "Beginn_Datum", "Ende_Datum", Einheiten_pro_Tag", "AnzahlTage", "AnzahlEinheiten". Das Feld "Trainer" beinhaltet "Vorname" und "Nachname". Die beiden letzten Felder sind berechnet. Die Abfrage ist nach dem Beginndatum sortiert und enthält lediglich Tenniskurse. (*qryKursdetail*)
17. Für die monatlichen Meldungen an das regionale Fremdenverkehrsbüro wird folgende Gästestatistik benötigt: Vorname und Nachname des Gastes sowie die Anmerkung "Inländer" bei österreichischen Gästen sowie "Ausländer" bei allen anderen Gästen. (*qryFremdenverkehrsstatistik*)

Parameterabfragen

18. Erstellen Sie eine Parameterabfrage, die alle Kursbelegungen auflistet, die sieben Tage nach dem Kursende noch nicht bezahlt wurden. Das Kursende soll bei Aufruf der Abfrage eingegeben werden können. Folgende Felder sollen angezeigt werden: "Gast_ID", "Kurs_ID", "Bezeichnung", Beginn_Datum" und "Ende_Datum". (*qryNicht_bezahlte_Kurse*)
19. Erweitern Sie die Parameterabfrage dahingehend, dass außerdem die Kursart (Tennis, Surfen, etc.) beim Aufruf angegeben werden kann. (*qryNicht_bezahlte_Kurse_erweitert*)
20. Erstellen Sie eine Parameterabfrage, mit der eine Gästeliste aller Gäste eines Landes mit den Feldern "Vorname", "Nachname" und "Anreise_Datum" abgerufen werden kann. (*qryGaeste_nach_Laendern*)

Gruppierungen, Summen und andere Statistiken

21. Am Ende der Hauptsaison möchte das Sporthotel eine Liste mit allen Kursen und deren Umsatz bilden. Erstellen Sie eine Umsatzauswertung, in der für jeden Kurs die Summe über den Preis errechnet wird. (qryUmsaetze)
22. Erweitern Sie die Umsatzauswertung um eine Parametereingabe bei der "Kurs_ID". Die Abfrage soll mit entsprechenden Eingaben alle Kurse, nur die Tenniskurse, nur die Tennisanfängerkurse etc. anzeigen. (qryUmsaetze_nach_Kurs)
23. Erstellen Sie eine Auswertung über die Anzahl der Gäste aus Österreich, gruppiert nach dem Ort. (qryGaestestatistik)
24. Erstellen Sie eine Auswertung über den Gesamtwert der nicht bezahlten Kurse. (qry_nicht_bezahlte_Kurse)
25. Erstellen Sie eine Auswertung, in der die Umsätze nach Ländern gruppiert sind. (qry_Umsatz_nach_Laender)

Aktualisierungsabfragen

Änderungen von Daten können problemlos manuell durchgeführt werden. Solange in der Datenbank nur wenige Daten enthalten sind, hält sich der Änderungsaufwand in Grenzen.
Mittels Aktualisierungsabfragen können die Änderungen vieler Daten automatisiert werden.

Feld:	Preis	Beginn_Datum
Tabelle:	tblKurse	tblKurse
Aktualisieren:	[Preis]*1,05	
Kriterien:		>=#01.05.2002#
oder:		

26. Erstellen Sie eine Abfrage, mit der die Preise aller Kurse, die ab 1. Mai des Jahres beginnen um 5 % erhöht werden. (qryPreiserhoehung)
27. Erstellen Sie eine Abfrage, in der Sie den Kursort aller Golfkurse in "Golfclub" ändern. (qryAenderung_Kursort)
28. Aufgrund von Umbauarbeiten müssen die Kurse vom 14. 4. des Jahres um einen Tag verschoben werden. (qryTerminverschiebung)

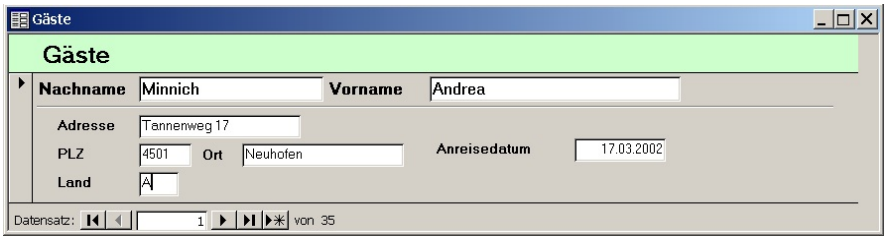
Löschabfragen

Mit verschiedenen Kriterien können Datensätze ausgewählt werden, die automatisch gelöscht werden.

29. Erstellen Sie eine Abfrage, mit der all jene Kurse definitiv gelöscht werden, die vor einem bestimmten Datum zu Ende gegangen sind. Das Datum soll über einen Parameter eingegeben werden können. (qryKurse_loeschen)

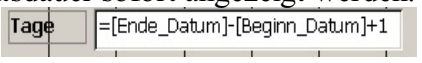
Formulare

Formulare dienen zum Eingeben, Betrachten, Ändern und eingeschränkt auch zum Ausdruck der Daten. Natürlich können alle Daten auch in der Datenblattansicht von Access eingegeben werden. Enthält eine Tabelle viele Datenfelder, dann wird die Dateneingabe in der Datenblattansicht aber schnell unübersichtlich.



Formular für die Eingabe der Gästedaten

Formulare bieten auch die Möglichkeit - ähnlich den berechneten Feldern in Abfragen - dem Abfrageentwurf selbst definierte Felder hinzuzufügen. Mittels Textfelder kann beispielsweise die Aufenthaltsdauer sofort angezeigt werden. Der berechnete Inhalt muss nur mit einer Formel ins Textfeld eingegeben werden:



Berichte

Mit Berichten können Datenlisten in beliebigem Layout erstellt werden. Im Prinzip sind Berichte Formularen sehr ähnlich, dienen aber hauptsächlich dem Ausdruck der Daten auf Papier. Berichte können entweder auf Tabellen oder auch auf Abfragen basieren. Wurden die Beziehungen zwischen den Tabellen festgelegt, dann können auch Datenfelder aus mehreren Tabellen verwendet werden.

Im Register Berichte des Datenbankfensters wird der **Berichts-Assistent** gestartet. Die Datenfelder werden mit den Pfeilsymbolen ausgewählt. Anschließend können **Gruppierungsebenen** festgelegt werden, wodurch alle gleichen Inhalte eines Felds in einer Gruppe zusammengefasst werden (beispielsweise nach dem Land oder dem ersten Buchstaben der PLZ)
Meist entspricht der automatisch erzeugte Bericht nur zum Teil den eigenen Vorstellungen. Das Aussehen kann sehr individuell verändert werden.
Berichte verfügen über folgende Bereiche:

- **Berichtskopf und Berichtsfuß**: Diese werden am Anfang (erste Seite) bzw. am Ende (letzte Seite) gedruckt.
- **Seitenkopf und Seitenfuß**: Diese Bereiche werden im Kopf- bzw. im Fußbereich jeder Seite gedruckt.
- **Gruppenkopf und Gruppenfuß**: Wurde im Berichts-Assistenten eine Gruppierung definiert, dann werden für dieses Feld ein Kopf- und Fußbereich eingerichtet.
- **Detailbereich**: Hier werden die eigentlichen Datensätze ausgedruckt.