

e-Book

Microsoft

Access 2003

VBA, SQL und Makros
Anleitung + Beispiele

Edi Bauer

Stundenermittlung		Summe	Überstunden	02:00
		1.209,75 €	Maximal	
KW	Tag	Vormittag	Nachmittag	Plus/Minus
16	Mi			
17	Do			
18	Fr			
19	Sa			
20	So			
21	38 Mo			
22	Di			
23	Mi			
24	Do			
25	Fr			
26	Sa			
27	So			
28	39 Mo			
29	Di			
30	Mi			
31	Do			
32	Fr			

Ertragsübersicht		2002		% v. Kosten		2001		% v. Kosten	
3	1 Erlöse	1.545.600	100%			105,2%	1.468.720	100%	
4	4 Wareneinsatz	687.512	44,5%			88,3%	778.215	50,4%	
5	5 Rohertrag	858.088	55,5%			124,3%	690.505	44,7%	
7	7 Personalkosten	324.530	21,0%	58,9%	94,0%	345.120	22,3%	59,6%	
8	8 Raumkosten	82.000	5,3%	14,9%	98,1%	83.560	5,4%	14,4%	
9	9 Beiträge	5.621	0,4%	1,0%	86,7%	6.487	0,4%	1,1%	
10	10 KFZ-Kosten	25.300	1,6%	4,6%	88,3%	28.658	1,9%	4,9%	
11	11 Werbekosten	5.600	0,4%	1,0%	114,3%	4.900	0,3%	0,8%	
12	12 Beratungskosten	12.580	0,8%	2,3%	92,0%	13.680	0,9%	2,4%	
13	13 Abschreibungen	15.800	1,0%	2,9%	89,4%	17.678	1,1%	3,1%	
14	14 Sonstige Kosten	79.222	5,1%	14,4%	100,1%	79.124	5,1%	13,7%	
15	15 Kosten gesamt	550.653	35,6%	100%	95,1%	579.207	37,5%	100%	
17	Reingewinn	307.435	19,9%		278,2%	111.209	7,0%		



Inhaltsverzeichnis

Inhaltsverzeichnis.....	2
1 SQL.....	4
1.1 Einführung.....	4
1.2 Der SQL-Editor.....	4
1.3 Anwendungsmöglichkeiten.....	5
1.4 Auswahlabfragen mit SQL.....	6
1.5 Berechnungen mit SQL durchführen.....	11
1.6 Verknüpfungen (Joins).....	12
1.7 Unterabfragen.....	14
1.8 UNION-Abfragen.....	15
1.9 Aktionsabfragen mit SQL.....	16
2 Makros.....	18
2.1 Einführung.....	18
2.2 Makros erstellen.....	19
2.3 Makros starten.....	21
2.4 Makros in Formularen und Berichten.....	22
2.5 Makrogruppen.....	25
2.6 Makros in VBA konvertieren.....	25
2.7 Makros mit Bedingungen.....	26
2.8 Das Tastenbelegungsmakro Autokeys.....	28
2.9 Fehlermeldungen in Makros.....	29
2.10 Das Startmakro AutoExec.....	29
3 Visual Basic für Anwendungen.....	30
3.1 Der VBA-Editor.....	30
3.2 Prozeduren.....	32
3.3 Variablen und Konstanten.....	34
3.4 Datenfelder (Arrays).....	37
3.5 Formeln und Operatoren.....	38
3.6 Funktionen.....	39
3.7 Programmierhilfen.....	41
3.8 Meldungsfenster mit MsgBox.....	42
3.9 Eingabedialog mit Inputbox.....	43
3.10 Bedingte Anweisungen.....	44



3.11	Schleifen.....	46
4	Access-VBA.....	48
4.1	Grundbegriffe.....	48
4.2	Steuerelemente.....	50
4.3	Formulare.....	56
4.4	Berichte.....	61
5	Datenzugriff.....	62
5.1	Domänenfunktionen.....	62
5.2	Data Access Objects (DAO).....	63
5.3	Datenzugriff mit SQL.....	68
5.4	DAO und SQL in Formularen.....	68
6	Wichtiges.....	70
6.1	Code drucken.....	70
6.2	Die VBA-Hilfe.....	70
6.3	Programme testen und Fehler suchen.....	71
6.4	Fehlerbehandlung.....	73
7	Beispiele.....	74
	Autovermietung: Abfragen mit SQL.....	74
	Lagerverwaltung: Abfragen mit SQL.....	77
	Sportverein: Benutzeroberfläche mit Makros.....	78
	Lagerverwaltung: Import und Aktualisierung mit Makros.....	82
	Datenzugriff mit VBA und DAO.....	84
	Feldinhalte aufteilen.....	86
	Datenauswertung mit den Data Access Objects.....	87
	Benutzeroberfläche und Gültigkeitsprüfungen.....	89
	Index.....	98



1 SQL

1.1 Einführung



Die **Structured Query Language** (strukturierte Abfragesprache, abgekürzt SQL) bildet in Access die Basis aller Abfragen. Für den fortgeschrittenen Access-Anwender stellt sie ein leistungsstarkes Werkzeug dar, ohne dessen Kenntnis nur ein Bruchteil der Möglichkeiten der Datenbankprogrammierung realisiert werden kann.

Im vorliegenden Kapitel werden Sie mit den SQL-Befehlen aus den Gruppen **DQL** (Data Query Language) für Auswahlabfragen und **DML** (Data Manipulation Language) für Aktionsabfragen vertraut gemacht.

1.2 Der SQL-Editor

The screenshot illustrates the workflow in Microsoft Access for creating and executing an SQL query. It is divided into six numbered steps:

- 1**: Selecting fields in the QBE grid.
- 2**: Viewing the QBE grid with criteria.
- 3**: Switching to SQL view from the View menu.
- 4**: The SQL Editor window showing the generated SQL code:


```
SELECT AUTO.[Marke und Typ], AUTO.Art, AUTO.Hubraum, AUTO.PS, AUTO.Tagespreis, AUTO.[km-Preis]
FROM AUTO
WHERE (((AUTO.Art)="kombi"))
ORDER BY AUTO.[Marke und Typ];
```
- 5**: Clicking the 'Ausführen' (Execute) button.
- 6**: The resulting data table:

Marke und Typ	Art	Hubraum	PS	Tagespreis	km-Preis
Audi A6 2.5 TDI	Kombi	2,5 Liter	170	124,00 €	0,20 €
Mercedes C200 T-Modell	Kombi	2,2 Liter	143	124,00 €	0,20 €
Opel Vectra Caravan	Kombi	1,8 Liter	122	90,00 €	0,10 €
Skoda Octavia	Kombi	1,9 Liter	110	74,00 €	0,05 €
*		0,0 Liter		0,00 €	0,00 €

① Um SQL-Code für eine Auswahl- oder Aktionsabfrage einzugeben, öffnen Sie zunächst die **Entwurfsansicht** einer neuen oder bestehenden Abfrage. Dazu wechseln Sie im Datenbankfenster in die Objektkategorie **Abfragen** und klicken auf die Schaltfläche **Neu** für eine neue oder **Entwurf** für eine bestehende Abfrage.

② Nun haben Sie entweder die Möglichkeit im **QbE-Bereich** (QbE = Query by Example) die gewünschte Abfrage zu gestalten, oder

③ Sie können über den Menüpunkt **SQL-Ansicht** in der Auswahlliste der Schaltfläche **Ansicht** (alternativ über den Menübefehl **Ansicht / SQL-Ansicht**) in den

④ **SQL-Editor** wechseln. Hier können Sie die Abfrage in SQL formulieren. Alles, was Sie im QbE-Bereich bereits definiert haben, wird automatisch in SQL übersetzt. Über die Schaltfläche

⑤ **Ausführen** können Sie die Abfrage starten und erhalten im Fall einer Auswahlabfrage das

⑥ **Abfrageergebnis**.



1.3 Anwendungsmöglichkeiten

Viele Abfragen werden Sie weiterhin mit dem QbE-Editor gestalten können. Um aber alle Möglichkeiten auszuschöpfen, brauchen Sie SQL. Auf dieser Seite bekommen Sie einen kurzen Überblick, wo Sie SQL-Code antreffen und verwenden können.



Abfragen

Bei Abfragen, die als Kriterium **Unterabfragen** verwenden (siehe Seite 14), muss die Unterabfrage in SQL geschrieben werden:


Der Abfrageentwurf zeigt als Ergebnis alle Fahrzeuge mit überdurchschnittlichem Hubraum.

Feld:	Marke und Typ	Hubraum
Tabelle:	AUTO	AUTO
Sortierung:		
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:		>(SELECT AVG(Hubraum) FROM Auto)

Ein weiteres Beispiel sind die

UNION-Abfragen, die Feldinhalte mehrerer nicht verbundener Tabellen kombinieren (Seite 15). UNION-Abfragen lassen sich über den QbE-Editor **nicht** definieren.

Listen- und Kombinationsfelder

In Formularen kann der Inhalt von **Listen-** oder **Kombinationsfeldern** durch einen SQL-Ausdruck in der Eigenschaft **Datensatzherkunft** definiert werden. Alternativ besteht hier die Möglichkeit, über die Schaltfläche  den QbE-Editor aufzurufen und die Datenherkunft dort zu gestalten.

The screenshot shows the 'Kunden+Preise' form with a list field 'Liste18'. The 'Liste18' dialog box is open, showing the 'Datensatzherkunft' property set to 'SELECT KUNDE.KNR, KUNDE.Nachname, KUNDE.Vorname FROM KUNDE;'. A red circle highlights the 'More options' button (three dots) in the bottom right corner of the dialog box.

Das Gleiche gilt für **Formulare**, die mit einer gefilterten Datenquelle arbeiten.

SQL und VBA

Beim Programmieren mit Access-VBA spielt SQL beim Erzeugen von Datenmengen (Recordsets, s. Seite 68) eine wichtige Rolle. Auf ein Recordset kann dann über die Datenzugriffsobjekte (DAO) zugegriffen werden:

```
dim db as DAO.Database
dim rs as DAO.Recordset
Set db = CurrentDb
Set rs = db.OpenRecordset("SELECT englisch, deutsch FROM Wörter")
rs.MoveLast
```

Das Beispiel definiert ein Recordset, das die Felder *englisch* und *deutsch* aller Datensätze aus der Tabelle **Wörter** enthält. Der Befehl *rs.MoveLast* bewirkt den Sprung zum letzten Datensatz des Recordset. Analog können Sie über SQL Datensätze hinzufügen, ändern oder löschen. Mehr über das Programmieren mit DAO erfahren Sie in Kap. 5.2.



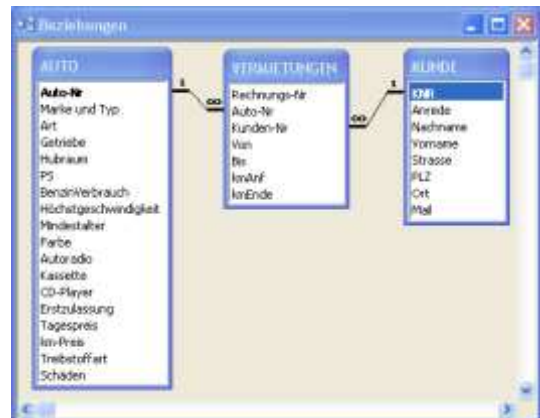
1.4 Auswahlabfragen mit SQL

Beispieldatenbank



Alle Beispiele in diesem Kapitel können Sie mit der Datenbank **Beispiele_Kapitel_01.mdb** nachvollziehen. Sie können diese Beispieldatenbank von www.redmonds.de herunterladen.

Die Abbildung zeigt den prinzipiellen Aufbau der Datenbank, mit der der Fuhrpark, die Kunden und die Mietvorgänge einer virtuellen Autovermietung erfasst und verwaltet werden.



SELECT ... FROM

Aufbau

Mit der Anweisung **SELECT** definieren Sie die Felder, die im Abfrageergebnis erscheinen sollen. Die Feldnamen werden durch Kommata getrennt. Der **FROM**-Klausel folgt der Name der Tabelle, die die Datenquelle der Abfrage darstellt. Mehrere Tabellennamen werden ebenfalls durch Kommata getrennt (s. Seite 12):

```
SELECT [Marke und Typ], Art, Hubraum, PS, Tagespreis, [km-Preis]
FROM Auto
```

Feldnamen

Feldnamen, die **Leerzeichen** oder **Sonderzeichen** enthalten (z. B. *Marke und Typ* oder *km-Preis*) müssen in eckigen Klammern geschrieben werden.

Bei einer Abfrage über mehrere Tabellen sollten die Tabellennamen, getrennt durch einen Punkt, vor den Feldnamen geschrieben werden. Diese Schreibweise ist zwingend, wenn in den verwendeten Tabellen gleiche Feldnamen vorkommen (Beispiel: *Auto.[Auto-Nr]* und *Vermietungen.[Auto-Nr]*, s. Seite 12).

Alle Felder einer Tabelle werden über den Stern angesprochen:

```
SELECT * FROM Auto
```

Die Prädikate ALL, DISTINCT und DISTINCTROW

Über diese Prädikate kann die **SELECT**-Anweisung eingeschränkt werden:

Prädikat	Beispiel	Erklärung
ALL	SELECT ALL Nachname, Vorname FROM Kunde	Zeigt alle Datensätze an (entspricht SELECT)
DISTINCT	SELECT DISTINCT Nachname, Vorname FROM Kunde	Von mehreren Datensätzen, bei denen Nachname und Vorname gleich sind, wird nur einer angezeigt.
DISTINCTROW	SELECT DISTINCTROW Nachname, Vorname FROM Kunde	Zeigt nur einen von mehreren Datensätzen an, bei denen alle Felder gleich sind.

Mit dem Prädikat **TOP** lassen sich weitere Einschränkungen vornehmen (s. Seite 10).

Groß- und Kleinschreibung

Bei Access-SQL spielt die Groß-/Kleinschreibung keine Rolle. Im Buch werden wegen der Übersichtlichkeit dennoch alle Schlüsselwörter groß geschrieben.

Übungen:
Autovermietung 74
Lagerverwaltung..... 77

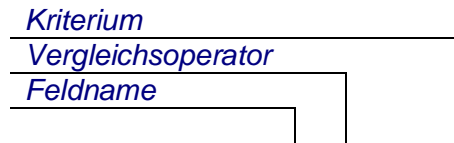




Kriterien verwenden

Aufbau der WHERE-Klausel

Mit dem Schlüsselwort *WHERE* können Sie in SQL Datensätze auswählen. Der Ausdruck hinter WHERE ergibt den logischen Wert *True* oder *False*. Datensätze mit dem Ergebnis *True* werden angezeigt. Das Abfrageergebnis des Beispiels zeigt nur Kunden an, die in Augsburg wohnen:



```
SELECT Nachname, Vorname, Ort FROM Kunde WHERE Ort = "Augsburg"
```

Vergleichsoperatoren

Zum Einsatz kommen die bekannten Operatoren:

Vergleichsoperator	Bedeutung	Vergleichsoperator	Bedeutung
=	gleich	<>	ungleich
<	kleiner	<=	kleiner oder gleich
>	größer	>=	größer oder gleich

Übungen:
Autovermietung74
Lagerverwaltung77

Kriterien

Die exakte Formulierung von Kriterien hängt vom Datentyp des zugrunde liegenden Feldes ab:

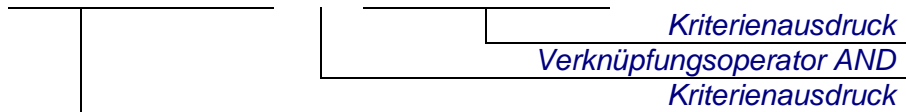
Datentyp	Beispiel	Wirkung
Text	SELECT Nachname, Vorname, Ort FROM Kunde WHERE Anrede = "Frau"	Zeigt alle weiblichen Kunden an.
	SELECT [Marke und Typ], Hubraum, PS, Mindestalter FROM Auto WHERE Mindestalter >="21"	Zeigt Fahrzeuge an, bei denen der Mieter mindestens 21 Jahre alt sein muss.
Kriterien des Datentyps Text werden in Anführungszeichen gesetzt. Das Gleiche gilt für Zahlen, wenn sie in einem Feld dieses Datentyps stehen.		
Zahl Währung	SELECT [Marke und Typ], Hubraum FROM Auto WHERE Hubraum >= 2.5	Zeigt Fahrzeuge an, deren Hubraum mindestens 2,5 cm ³ beträgt.
In SQL-Ausdrücken gilt für Zahlen die angelsächsische Darstellung. Dezimalzahlen müssen deswegen mit einem Punkt anstelle des Kommas geschrieben werden. Weitere Formatierungen (z. B. Maßeinheiten) oder Tausendertrennzeichen müssen weggelassen werden.		
Datum	SELECT [Rechnungs-Nr], [Auto-Nr], Von, Bis FROM Vermietungen WHERE Bis >= #11/15/2004#	Alle Fahrzeuge mit Rückgabedatum ab dem 15. November 2004.
Die allgemeine Form für ein Kriterium des Datentyps Datum lautet #Monat/Tag/Jahr#		
Ja/Nein	SELECT [Marke und Typ], Hubraum, Art, [CD-Player] FROM Auto WHERE [CD-Player]= yes	Alle Fahrzeuge mit CD-Spieler werden angezeigt.
Ja/Nein-Felder werden mit <i>yes</i> oder <i>no</i> abgefragt. Alternativ können Sie das Kriterium <i>-1</i> (für Ja) oder <i>0</i> (für Nein) verwenden. Beispiel: <i>[CD-Player] = -1</i>		



Kriterien verknüpfen

Sie können mehrere Kriterienausdrücke über Verknüpfungsoperatoren (Boolesche Operatoren) verbinden:

```
SELECT Nachname, Vorname, Ort FROM Kunde
WHERE Ort = "Bayreuth" AND Anrede = "Frau"
```



Das Beispiel zeigt alle Kundinnen mit Wohnort Bayreuth an.

Übung:
Autovermietung 74

Verknüpfungsoperatoren

Operator	Beispiel	Wirkung
<i>OR</i>	SELECT Nachname, Vorname, Ort FROM Kunde WHERE Ort = "Bayreuth" OR Ort = "Augsburg"	Anzeige aller Kunden aus Bayreuth und Augsburg
Entweder das eine oder das andere Kriterium (oder beide) müssen erfüllt sein, damit der Datensatz angezeigt wird. Der Operator <i>OR</i> steht oft im Gegensatz zur umgangssprachlichen Beschreibung des Abfrageergebnisses (... aus Bayreuth und Augsburg).		
<i>AND</i>	SELECT [Marke und Typ], Hubraum FROM Auto WHERE Hubraum >= 1.5 AND Hubraum <= 2.0	Alle Fahrzeuge mit einem Hubraum von mindestens 1,5 cm ³ und höchstens 2,0 cm ³ werden angezeigt.
Beide Bedingungen müssen gleichzeitig erfüllt sein, damit der Datensatz angezeigt wird. Die Verknüpfung mit <i>AND</i> dient häufig zur Definition von Bereichen . Alternativ können Sie hier die bequemere Kombination mit <i>BETWEEN ... AND</i> verwenden (siehe unten).		
<i>XOR</i>	SELECT Nachname, Vorname FROM Kunde WHERE Ort = "Bayreuth" XOR Anrede = "Herr"	Entweder ist der Kunde weiblich und kommt aus Bayreuth oder er ist männlich, darf dann aber nicht in Bayreuth wohnen.
Bei Verwendung des exklusiven Oder wird ein Datensatz nur angezeigt, wenn genau eine der aufgeführten Bedingungen erfüllt ist.		



Weitere nützliche Operatoren

Operator	Beispiel	Wirkung
<i>BETWEEN ... AND</i>	SELECT [Marke und Typ], Hubraum FROM Auto WHERE Hubraum BETWEEN 1.5 AND 2.0	Alle Fahrzeuge mit Hubraum zwischen 1,5 und 2,0 cm ³ werden angezeigt.
Die bequemere Variante einen Bereich zu definieren. Die Grenzen (hier: 1,5 und 2,0) gehören zum Definitionsbereich!		
<i>IN</i>	SELECT Nachname, Vorname FROM Kunde WHERE Ort IN ("Augsburg", "Bayreuth", "Regensburg")	Das Abfrageergebnis zeigt Kunden aus Augsburg, Bayreuth und Regensburg.
Durch die <i>IN</i> -Klausel können Sie sich Schreibarbeit sparen. Hier ersetzt sie die sperrigere (aber nichtsdestotrotz genauso funktionierende) Anweisung <i>WHERE Ort = "Augsburg" OR Ort = "Bayreuth" OR Ort = "Regensburg"</i>		
<i>NOT</i>	SELECT [Marke und Typ], Art FROM Auto WHERE NOT (Art = "Kombi")	Fahrzeuge des Typs Kombi werden nicht angezeigt.
Dieser Operator verneint die nachfolgende Bedingung. Alternative: <i>WHERE Art <> "Kombi"</i>		



Leere Felder

Datensätze, die leere Felder enthalten, finden Sie über das Schlüsselwort **NULL**. Als Vergleichsoperator müssen Sie in diesem Fall **IS** verwenden. Das Beispiel zeigt Kunden an, bei denen keine Mail-Adresse eingetragen ist:

```
SELECT Nachname, Vorname, Mail FROM KUNDE WHERE Mail IS NULL
```

Felder, die nicht leer sind, finden Sie durch Verneinung mit **NOT**:

```
SELECT Nachname, Vorname, Mail FROM Kunde WHERE Mail IS NOT NULL
```



Kriterien mit Platzhaltern

Für unscharfes Suchen verwenden Sie den Operator **LIKE** zusammen mit geeigneten Platzhaltern. Beispiel:

```
SELECT Nachname, Vorname FROM KUNDE WHERE Nachname LIKE "M*"
```

	Nachname	Vorname
	Mutschler	Ruth
	Maier	Horst
	Müller	Monika
▶	Metzger	Rolf

Übung:
Autovermietung74

Mögliche Platzhalter

Platzhalter	Beispiel	Wirkung	Beschreibung																	
Stern	WHERE Nachname LIKE "ha*"	<table border="1"> <thead> <tr><th>Nachname</th><th>Vorname</th></tr> </thead> <tbody> <tr><td>Hartmann</td><td>Josef</td></tr> <tr><td>Hartl</td><td>Andrea</td></tr> <tr><td>Hansen</td><td>Lutz</td></tr> </tbody> </table>	Nachname	Vorname	Hartmann	Josef	Hartl	Andrea	Hansen	Lutz	Alle Nachnamen, die mit ha beginnen									
	Nachname	Vorname																		
	Hartmann	Josef																		
Hartl	Andrea																			
Hansen	Lutz																			
WHERE Nachname LIKE "*er"	<table border="1"> <thead> <tr><th>Nachname</th><th>Vorname</th></tr> </thead> <tbody> <tr><td>Seelhar</td><td>Thea</td></tr> <tr><td>Mutschler</td><td>Ruth</td></tr> <tr><td>Maier</td><td>Horst</td></tr> <tr><td>Müller</td><td>Monika</td></tr> <tr><td>Koetner</td><td>Paula</td></tr> <tr><td>Zeller</td><td>Doris</td></tr> <tr><td>Hunger</td><td>Franz</td></tr> <tr><td>Metzger</td><td>Rolf</td></tr> </tbody> </table>	Nachname	Vorname	Seelhar	Thea	Mutschler	Ruth	Maier	Horst	Müller	Monika	Koetner	Paula	Zeller	Doris	Hunger	Franz	Metzger	Rolf	Alle Nachnamen, die mit er enden.
Nachname	Vorname																			
Seelhar	Thea																			
Mutschler	Ruth																			
Maier	Horst																			
Müller	Monika																			
Koetner	Paula																			
Zeller	Doris																			
Hunger	Franz																			
Metzger	Rolf																			
WHERE Nachname LIKE "*sch*"	<table border="1"> <thead> <tr><th>Nachname</th><th>Vorname</th></tr> </thead> <tbody> <tr><td>Mutschler</td><td>Ruth</td></tr> <tr><td>Scheuvelin</td><td>Uwe</td></tr> </tbody> </table>	Nachname	Vorname	Mutschler	Ruth	Scheuvelin	Uwe	Alle Nachnamen mit sch , egal an welcher Stelle.												
Nachname	Vorname																			
Mutschler	Ruth																			
Scheuvelin	Uwe																			
Der Stern steht als Platzhalter für beliebig viele Zeichen .																				
Fragezeichen	WHERE Nachname LIKE "ma?er"	<table border="1"> <thead> <tr><th>Nachname</th><th>Vorname</th></tr> </thead> <tbody> <tr><td>Maier</td><td>Horst</td></tr> <tr><td>Mayer</td><td>Jonathan</td></tr> <tr><td>Maler</td><td>Gerlinde</td></tr> </tbody> </table>	Nachname	Vorname	Maier	Horst	Mayer	Jonathan	Maler	Gerlinde	Alle Nachnamen, die mit ma beginnen, mit er enden und fünf Buchstaben haben.									
	Nachname	Vorname																		
Maier	Horst																			
Mayer	Jonathan																			
Maler	Gerlinde																			
WHERE Nachname LIKE "m?er"	<table border="1"> <thead> <tr><th>Nachname</th><th>Vorname</th></tr> </thead> <tbody> <tr><td>Maier</td><td>Horst</td></tr> <tr><td>Mayer</td><td>Jonathan</td></tr> <tr><td>Maler</td><td>Gerlinde</td></tr> <tr><td>Meier</td><td>Sabrina</td></tr> </tbody> </table>	Nachname	Vorname	Maier	Horst	Mayer	Jonathan	Maler	Gerlinde	Meier	Sabrina	Alle Nachnamen, die mit m beginnen, mit er enden und fünf Buchstaben haben.								
Nachname	Vorname																			
Maier	Horst																			
Mayer	Jonathan																			
Maler	Gerlinde																			
Meier	Sabrina																			
Das Fragezeichen ist Platzhalter für ein Zeichen																				
Raute	WHERE PLZ LIKE "86###"	<table border="1"> <thead> <tr><th>Nachname</th><th>PLZ</th></tr> </thead> <tbody> <tr><td>Helm</td><td>86153</td></tr> <tr><td>Mayer</td><td>86198</td></tr> <tr><td>Maler</td><td>86198</td></tr> <tr><td>Meier</td><td>86179</td></tr> </tbody> </table>	Nachname	PLZ	Helm	86153	Mayer	86198	Maler	86198	Meier	86179	Alle Postleitzahlen, die mit 86 beginnen, restliche Ziffern beliebig.							
Nachname	PLZ																			
Helm	86153																			
Mayer	86198																			
Maler	86198																			
Meier	86179																			
Als Platzhalter für Zahlen zwischen 0 und 9 verwenden Sie die Raute.																				
Klammer	WHERE Nachname LIKE "ma[iy]er"	<table border="1"> <thead> <tr><th>Nachname</th><th>Vorname</th></tr> </thead> <tbody> <tr><td>Maier</td><td>Horst</td></tr> <tr><td>Mayer</td><td>Jonathan</td></tr> </tbody> </table>	Nachname	Vorname	Maier	Horst	Mayer	Jonathan	Die angezeigten Nachnamen beginnen mit ma , enthalten dann entweder i oder y und enden auf er .											
	Nachname	Vorname																		
Maier	Horst																			
Mayer	Jonathan																			
WHERE Nachname LIKE "ma[!iy]er"	<table border="1"> <thead> <tr><th>Nachname</th><th>Vorname</th></tr> </thead> <tbody> <tr><td>Maler</td><td>Gerlinde</td></tr> </tbody> </table>	Nachname	Vorname	Maler	Gerlinde	Jetzt dürfen die Nachnamen an der entsprechenden Stelle i oder y nicht enthalten.														
Nachname	Vorname																			
Maler	Gerlinde																			
Die Mengenklammer wirkt wie eine Liste der Zeichen , nach denen gesucht werden soll. Ein vor die Liste gestelltes Ausrufezeichen schließt die Buchstaben aus.																				

Wichtig!
Den **LIKE**-Operator können Sie nur bei Feldern vom Datentyp **Text** oder **Memo** einsetzen!



Sortieren

Das Abfrageergebnis können Sie mit der **ORDER BY**-Klausel sortieren. Im Beispiel erfolgt die Sortierung nach dem Nachnamen:

```
SELECT * FROM Kunde WHERE Anrede = "Herr" ORDER BY Nachname
```

Um nach mehreren Feldern zu sortieren verwenden Sie eine Liste der durch Kommata getrennten Feldnamen:

```
SELECT * FROM Kunde ORDER BY Nachname, Vorname
```

Nachname	Vorname
Müller	Monika
Mutschler	Matthias
Mutschler	Ruth
Österle	Karin
Patena	Matthias

Sortierreihenfolge

Mit Hilfe der Schlüsselwörter **ASC** und **DESC** können Sie die Sortierreihenfolge einstellen. Der Beispielausdruck mit **DESC** liefert ein **absteigend** nach Nachname sortiertes Ergebnis (Nachnamen, die mit Z beginnen, kommen zuerst):

```
SELECT * FROM Kunde WHERE Anrede = "Herr" ORDER BY Nachname DESC
```

Die **aufsteigende Sortierung** ist voreingestellt, so dass Sie das Schlüsselwort **ASC** auch weglassen können.

TOP

Im Zusammenhang mit der Sortierung können Sie das Prädikat **TOP** verwenden, das die Ergebnismenge einschränkt. Das Beispiel ermittelt die beiden Fahrzeuge mit dem größten Hubraum:

```
SELECT TOP 2 [Marke und Typ], Hubraum
FROM Auto ORDER BY Hubraum DESC
```

Marke und Typ	Hubraum
BMW 740d	3,9 Liter
Audi A6 2.5 TDI	2,5 Liter

Mit dem Schlüsselwort **PERCENT** können Sie die Fahrzeuge ermitteln, deren Benzinverbrauch im Bereich der unteren 20 Prozent liegt:

```
SELECT TOP 20 PERCENT [Marke und Typ], Benzinverbrauch
FROM Auto ORDER BY Benzinverbrauch
```

Formatieren von Abfrageergebnissen

Sie können die Anzeige von Feldbezeichnern und Feldinhalten im Abfrageergebnis durch folgende Schlüsselwörter und Funktionen beeinflussen:

AS `SELECT [Marke und Typ] AS Fahrzeug,`
 `PS AS Leistung FROM Auto`

Fahrzeug	Leistung
Audi A6 2.5 TDI	170
Audi A4 Avant	115

Mit der **AS**-Klausel verändern Sie die **Anzeige des Feldnamens** zur Laufzeit der Abfrage.

Verkettung `SELECT Nachname + ", " + Vorname`
 `AS Name FROM Kunde`

Name
Ludwig, Lotte
Dietrich, Gunter

Über die Operatoren **+** oder **&** können **Feldinhalte** in einem Feld ausgegeben werden.

Funktion `SELECT Nachname, NZ(Mail, "Keine`
NZ `Mail-Adresse") AS [Mail-Adresse]`
 `FROM Kunde`

Nachname	Mail-Adresse
Ludwig	#mailto:ludwig@web.de#
Dietrich	#mailto:gunter@web.de#
▶ Fees	Keine Mail Adresse

Mit der Funktion **NZ(Feldname, Textausdruck)** definieren Sie den **Feldinhalt** für leere Felder.

Diese Funktion können Sie auch in **VBA** verwenden.

Funktion `SELECT Format(von, "yyyy-mm-dd") AS`
FORMAT `Anfang FROM Vermietungen`

Anfang
2004-11-27
2004-11-13

Die Funktion **FORMAT(Feldname, Formatausdruck)** bietet vielfältige Möglichkeiten zur Formatierung von Feldinhalten. Mehr zu den möglichen Formatausdrücken lesen Sie auf Seite 40.

Funktion `SELECT Anrede, IIF(Anrede = "Herr",`
IIF `"Sehr geehrter Herr", "Sehr geehrte`
 `Frau") AS Briefanrede FROM Kunde`

Anrede	Briefanrede
Frau	Sehr geehrte Frau
Herr	Sehr geehrter Herr

Die Funktion **IIF(Prüfung, Dann_Wert, Sonst_Wert)** ergibt einen Feldinhalt, der vom Ergebnis des Prüfausdrucks abhängt. Diese Funktion wird Ihnen beim Thema VBA wieder begegnen (Seite 45).

Übung:
Autover-
mietung 74





1.5 Berechnungen mit SQL durchführen

Berechnete Felder

In SQL bauen Sie die Formel für ein berechnetes Feld folgendermaßen auf:

```
SELECT [Marke und Typ], Tagespreis*1.16 AS [Tagespreis brutto]
FROM AUTO
```

Formel

AS-Klausel

Name des berechneten Feldes



Übung:
Autover-
mietung74

Zahlenwerte in Formeln erhalten den Punkt als **Dezimaltrennzeichen**. Für die Schreibweise der Feldnamen gilt das schon auf Seite 6 Gesagte.

Marke und Typ	Tagespreis brutto
Audi A6 2.5 TDI	143,84 €
Audi A4 Avant	102,08 €
BMW 320i	104,40 €

Aggregatfunktionen

Um in einem Feld Berechnungen über mehrere Datensätze durchzuführen, verwenden Sie **Aggregatfunktionen**:

```
SELECT COUNT(*) AS [Anzahl Vermietungen]
      AVG(kmEnde-kmAnf) AS Durchschnitt,
      SUM(kmEnde-kmAnf) AS Gesamt,
      MAX(kmEnde-kmAnf) AS [Längste Strecke],
      MIN(kmEnde-kmAnf) AS [Kürzeste Strecke]
FROM Vermietungen
```

Das Beispiel berechnet für jede Vermietung eines Fahrzeuges über den Aus-

Anzahl Vermietungen	Durchschnitt	Gesamt	Längste Strecke	Kürzeste Strecke
12	1348,3333333	16180	3800	200

druck *kmEnde - kmAnf* die gefahrene Strecke und ermittelt den **Durchschnitt (AVG)**, die **Summe (SUM)**, das **Maximum (MAX)** und das **Minimum (MIN)** der Strecken. Über **COUNT** wird die **Anzahl** der gefahrenen Strecken ermittelt. Da es keine Rolle spielt, in welchem Feld die Datensätze gezählt werden, erhält COUNT als Argument den Stern.

Datensätze gruppieren

Um die Anzahl der Vermietungen gesondert für jedes Fahrzeug anzeigen zu lassen, verwenden Sie die **GROUP BY-Klausel**:

```
SELECT [Auto-Nr],
      Count (*) AS [Anzahl Vermietungen]
FROM Vermietungen
GROUP BY [Auto-Nr]
```

Auto-Nr	Anzahl Vermietungen
AU001	4
AU002	1
BM002	2
BM003	1
FI001	2
FO001	1
OP001	1

Die Ergebnisse der COUNT-Funktion werden auf diese Weise für jede *Auto-Nr* zusammengefasst (gruppiert).

Kriterien in gruppierten Abfragen

Möchten Sie in gruppierten Abfragen Datensätze selektieren, verwenden Sie dazu die **HAVING-Klausel**:

```
SELECT [Auto-Nr],
      Count (*) AS [Anzahl Vermietungen]
FROM Vermietungen
GROUP BY [Auto-Nr]
HAVING Count (*) >=2
```

Auto-Nr	Anzahl Vermietungen
AU001	4
BM002	2
FI001	2

Es werden nur Fahrzeuge angezeigt, die mindestens zwei Mal vermietet wurden.





1.6 Verknüpfungen (Joins)

Grundlagen

Über **Joins** können Daten aus zwei oder mehr Tabellen zu einem Abfrageergebnis zusammengefasst werden. Die Tabellen werden jeweils paarweise verknüpft. Voraussetzung ist ein Schlüsselfeld, das in beiden Tabellen vorkommt und den gleichen Datentyp sowie die gleiche Feldgröße aufweist.



Übungen:
Autovermietung 74
Lagerverwaltung..... 77

Im Beispiel werden die Inhalte der Felder *Auto-Nr* und *Marke und Typ* aus der Tabelle **AUTO** und die Inhalte der Felder *kmAnf* und *kmEnde* (Kilometerstand bei Aushändigung und Rückgabe des Fahrzeugs) aus der Tabelle **VERMIETUNGEN** zusammengeführt. Die Tabellen werden über das Schlüsselfeld *Auto-Nr* verknüpft.

Tabelle AUTO		Tabelle VERMIETUNGEN		
Auto-Nr	Marke und Typ	Auto-Nr	kmAnf	kmEnde
AU001	Audi A6 2.5 TDI	AU001	17.000	17.200
AU002	Audi A4 Avant	AU001	14.000	14.500
BM001	BMW 320i	AU001	14.500	17.000
BM002	BMW 320i Cabrio	AU001	17.200	21.000
BM003	BMW 740i	AU002	30.000	30.450
FI001	Fiat Cinquecento 0.9 i.e.	BM002	73.000	73.500
		BM002	73.500	75.000
		BM003	85.000	85.500
		FI001	96.000	96.800
		FI001	96.800	100.000
		FI001	75.000	77.000

```
INNER JOIN : Auswahlabfrage
SELECT Auto.[Auto-Nr], Auto.[Marke und Typ], Vermietungen.kmAnf, Vermietungen.kmEnde
FROM Auto INNER JOIN Vermietungen
ON Vermietungen.[Auto-Nr]=Auto.[Auto-Nr]
```

Auto-Nr	Marke und Typ	kmAnf	kmEnde
AU001	Audi A6 2.5 TDI	17.000	17.200
AU001	Audi A6 2.5 TDI	14.000	14.500
AU001	Audi A6 2.5 TDI	14.500	17.000
AU001	Audi A6 2.5 TDI	17.200	21.000
AU002	Audi A4 Avant	30.000	30.450
BM002	BMW 320i Cabrio	73.000	73.500
BM002	BMW 320i Cabrio	73.500	75.000
BM003	BMW 740i	85.000	85.500
FI001	Fiat Cinquecento 0.9 i.e.	96.000	96.800
FI001	Fiat Cinquecento 0.9 i.e.	96.800	100.000
FI001	Fiat Cinquecento 0.9 i.e.	75.000	77.000

Inner-Join

Beim **Inner-Join** oder **Equi-Join** liefert das Abfrageergebnis nur solche Datensätze, bei denen der Inhalt des Schlüsselfeldes in beiden Tabellen vorkommt. Im Beispiel werden also nur Fahrzeuge angezeigt, die vermietet wurden.

Erste Möglichkeit: Erweiterung der WHERE-Klausel

Die Schlüsselfelder werden über die **WHERE**-Klausel verknüpft:

```
SELECT [Auto-Nr], [Marke und Typ], kmAnf, kmEnde
FROM Auto, Vermietungen WHERE Vermietungen.[Auto-Nr]=Auto.[Auto-Nr]
```

Weitere Kriterienausdrücke können über Verknüpfungsoperatoren (Seite 7) angehängt werden. In beiden Tabellen vorkommende Feldnamen müssen durch den vorangestellten Tabellennamen unterschieden werden (siehe Seite 6).

Zweite Möglichkeit: Verwendung der JOIN-Klausel

Die verbundenen Tabellen werden durch die Klausel **INNER JOIN** deklariert. Hinter dem Schlüsselwort **ON** geben Sie die Schlüsselfelder an, über die die Tabellen verknüpft werden.

```
SELECT [Auto-Nr], [Marke und Typ], kmAnf, kmEnde
FROM Auto INNER JOIN Vermietungen
```

Deklaration der Tabellen

ON Vermietungen.[Auto-Nr]=Auto.[Auto-Nr]

Deklaration der Schlüsselfelder

Dynaset

Von einem Dynaset (= dynamic set = dynamische Menge) spricht man, wenn das Abfrageergebnis veränderbar ist. In einem Dynaset können Sie Datensätze wie in einer Tabelle abändern (Ausnahme: Berechnete Felder, siehe Seite 11). Beachten Sie, dass nur die zweite Möglichkeit (Join) ein Dynaset liefert.



Outer-Join (Left- oder Right-Join)

Nicht zu jedem Fahrzeug gibt es auch eine Vermietung. Um auch die Fahrzeuge zu sehen, die noch nicht vermietet wurden, verknüpfen Sie die Tabellen über einen **Outer-Join** (Im Beispiel **LEFT JOIN**):

```
SELECT [Auto-Nr],
[Marke und Typ], kmAnf, kmEnde
FROM Auto LEFT JOIN Vermietungen
ON Vermietungen.[Auto-Nr] =
Auto.[Auto-Nr]
```

Alle Datensätze aus der Tabelle **AUTO** werden angezeigt. Die Felder *kmAnf* und *kmEnde* aus **VERMIETUNGEN** werden bei nicht gemieteten Fahrzeugen mit dem Wert *Null* aufgefüllt.

Auto-Nr	Marke und Typ	kmAnf	kmEnde
AU001	Audi A6 2.5 TDI	17.000	17.200
AU001	Audi A6 2.5 TDI	14.000	14.500
AU001	Audi A6 2.5 TDI	14.500	17.000
AU001	Audi A6 2.5 TDI	17.200	21.000
AU002	Audi A4 Avant	30.000	30.450
BM001	BMW 320i		
BM002	BMW 320i Cabrio	73.000	73.500
BM002	BMW 320i Cabrio	73.500	75.000
BM003	BMW 740d	65.000	65.500
FI001	Fiat Cinquecento 0.9 i.e.	96.000	96.600
FI001	Fiat Cinquecento 0.9 i.e.	96.600	100.000
FO001	Ford Fiesta 1.8 D Fun	75.000	77.000
FO002	Ford Mondeo 1.8i CLX		
MB001	Mercedes C200 T-Modell		
MB002	Mercedes A140		
MB003	Mercedes C220 CDI		
OP001	Opel Corsa	56.000	56.230
OP002	Opel Vectra		
OP003	Opel Vectra Caravan		
SK001	Skoda Octavia		



Left-Join oder Right-Join?

Über die **FROM**-Klausel

```
...FROM Auto LEFT JOIN Vermietungen
```

werden alle Datensätze der links stehenden Tabelle (**AUTO**) angezeigt. Der Ausdruck

```
...FROM Vermietungen RIGHT JOIN Auto
```

liefert demzufolge das gleiche Ergebnis. Jetzt sollte es für Sie kein Problem mehr sein den folgenden SQL-Ausdruck zu lesen:

```
SELECT Nachname, Vorname, [Rechnungs-Nr]
FROM Kunde LEFT JOIN Vermietungen
ON Kunde.KNR = Vermietungen.[Kunden-Nr]
```

Er liefert alle Datensätze aus der Tabelle **KUNDE** und, soweit vorhanden, die Rechnungsnummern aus der Tabelle **VERMIETUNGEN**.

Nachname	Vorname	Rechnungs-Nr
Ludwig	Lotte	00007
Ludwig	Lotte	00010
Ludwig	Lotte	00011
Dietrich	Gunter	00002
Dietrich	Gunter	00006
Dietrich	Gunter	00009
Dietrich	Gunter	00012
Fees	Barbara	
Hartmann	Josef	

Übung:
Autovermietung74

Mehr als zwei Tabellen verknüpfen

Bei der Verknüpfung von drei Tabellen wird der SQL-Ausdruck komplexer:

```
SELECT Nachname, Vorname, [Marke und Typ], Von, Bis
FROM Kunde INNER JOIN (Vermietungen INNER JOIN Auto ON
Vermietungen.[Auto-Nr] = Auto.[Auto-Nr])
ON Kunde.KNR = Vermietungen.[Kunden-Nr]
```

Nachname	Vorname	Marke und Typ	Von	Bis
Ganthen	Lilli	BMW 320i Cabrio	01.07.2004	02.07.2004
Dietrich	Gunter	BMW 740d	11.11.2004	12.11.2004
Zeller	Doris	Ford Fiesta 1.8 D Fun	12.11.2004	30.11.2004
Ganthen	Lilli	BMW 320i Cabrio	10.09.2004	12.09.2004
Zimmermann	Otto	Audi A6 2.5 TDI	27.11.2004	27.11.2004
Dietrich	Gunter	Audi A6 2.5 TDI	13.11.2004	13.11.2004
Ludwig	Lotte	Fiat Cinquecento 0.9 i.e.	25.06.2004	30.06.2004
Harti	Andrea	Audi A6 2.5 TDI	14.11.2004	25.11.2004
Dietrich	Gunter	Audi A6 2.5 TDI	01.12.2004	10.12.2004
Ludwig	Lotte	Fiat Cinquecento 0.9 i.e.	12.09.2004	15.09.2004
Ludwig	Lotte	Opel Corsa	01.07.2004	01.07.2004
Dietrich	Gunter	Audi A4 Avant	11.12.2004	11.12.2004

In der Klammer wird die Verknüpfung zwischen den Tabellen **VERMIETUNGEN** und **AUTO** definiert. Sie liefert ein Dynaset, das wiederum über das Schlüsselfeld *Kunden-Nr* mit der Tabelle **KUNDE** verknüpft wird (vergleichen Sie hierzu auch nochmals

das Beziehungsfenster auf Seite 6).

Übersetzungshilfe

Entwerfen Sie komplexere Abfragen zunächst im **QbE-Editor** (siehe Seite 4) und ändern Sie den von Access automatisch erzeugten SQL-Ausdruck dann nach Ihren Wünschen ab.



1.7 Unterabfragen

Sollen Kriterien erst zur Laufzeit einer Abfrage ermittelt werden, müssen Sie Unterabfragen verwenden. Wenn Sie die Abfrage im QbE-Editor entwerfen, können Sie die Unterabfrage in SQL schreiben oder mit dem QbE-Editor gestalten und in die Zeile **Kriterium** kopieren.

Feld:	Marke und Typ	Hubraum
Tabelle:	Auto	Auto
Sortierung:		
Anzeigen:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Kriterien:	>All (SELECT Hubraum FROM Auto WHERE [Auto-Nr] LIKE "AU??")	

Die Unterabfrage liefert genau einen Wert zurück

Sie können die Frage **Welche Kunden wohnen im gleichen Ort wie Herr Erich Helm?** lösen, indem Sie zunächst die Abfrage

```
SELECT Ort FROM Kunde WHERE Nachname = "Helm" AND Vorname = "Erich"
```

ausführen und dann das Ergebnis **Augsburg** in die Abfrage

```
SELECT Nachname, Vorname FROM Kunde WHERE Ort = "Augsburg"
```

einsetzen. Alternativ erledigen Sie das Problem mit einer Unterabfrage und ersparen sich einen Arbeitsgang:

```
SELECT Nachname, Vorname
FROM Kunde
WHERE Ort = (
    SELECT Ort FROM Kunde WHERE Nachname = "Helm"
    AND Vorname = "Erich")
```

Oder Sie möchten alle Kunden ermitteln, die ein Fahrzeug überdurchschnittlich lange gemietet haben:

```
SELECT [Kunden-Nr], (Bis-Von) AS Mietdauer
FROM Vermietungen
WHERE (Bis-Von)>(SELECT AVG(Bis-Von) FROM Vermietungen)
```

Nur ein Feld

Beachten Sie, dass Unterabfragen nur über ein Feld ausgeführt werden dürfen. Eine Ausnahme hiervon bilden Unterabfragen mit dem Operator **EXISTS** (Seite 15)

Die Unterabfrage liefert eine Menge von Werten

In diesem Fall müssen Sie mit Operatoren einen dieser Werte auswählen.

Operator	SQL-Ausdruck	Ergebnis
Unterabfrage	SELECT Hubraum FROM Auto WHERE [Auto-Nr] LIKE "AU???"	Hubraum 2,5 Liter 1,9 Liter

Ermittelt wird der Hubraum aller Fahrzeuge der Marke Audi

```
ANY SELECT [Marke und Typ], Hubraum
FROM Auto
WHERE Hubraum >
ANY (SELECT Hubraum FROM Auto
WHERE [Auto-Nr] LIKE "AU??")
```

Marke und Typ	Hubraum
Audi A6 2.5 TDI	2,5 Liter
BMW 320i	2,2 Liter
BMW 320i Cabrio	2,2 Liter
BMW 740d	3,9 Liter
Mercedes C200 T-Modell	2,2 Liter
Mercedes C220 CDI	2,2 Liter

Es werden alle Fahrzeuge angezeigt, deren Hubraum größer ist als **irgendein** Hubraum (in Verbindung mit dem Vergleichsoperator **>** entspricht dies dem kleinsten Hubraum) eines Fahrzeugs der Marke Audi. Die Kombination **= ANY** entspricht dem Operator **IN**.

```
ALL SELECT [Marke und Typ], Hubraum
FROM Auto
WHERE Hubraum >
ALL (SELECT Hubraum FROM Auto
WHERE [Auto-Nr] LIKE "AU??")
```

Marke und Typ	Hubraum
BMW 740d	3,9 Liter

Es werden alle Fahrzeuge angezeigt, deren Hubraum größer ist als **jeder** Hubraum (= als der größte Hubraum) eines Fahrzeugs der Marke Audi. **<> ALL** entspricht **NOT IN**.

Übungen:
Autover-
mietung 74



Operator	SQL-Ausdruck	Ergebnis												
<i>IN</i>	SELECT [Marke und Typ], Hubraum	<table border="1"> <thead> <tr> <th>Marke und Typ</th> <th>Hubraum</th> </tr> </thead> <tbody> <tr> <td>Audi A6 2.5 TDI</td> <td>2,5 Liter</td> </tr> <tr> <td>Audi A4 Avant</td> <td>1,9 Liter</td> </tr> <tr> <td>Skoda Octavia</td> <td>1,9 Liter</td> </tr> <tr> <td>VW 1.9 TDI</td> <td>1,9 Liter</td> </tr> <tr> <td>VW Golf 1.9 TDI</td> <td>1,9 Liter</td> </tr> </tbody> </table>	Marke und Typ	Hubraum	Audi A6 2.5 TDI	2,5 Liter	Audi A4 Avant	1,9 Liter	Skoda Octavia	1,9 Liter	VW 1.9 TDI	1,9 Liter	VW Golf 1.9 TDI	1,9 Liter
Marke und Typ	Hubraum													
Audi A6 2.5 TDI	2,5 Liter													
Audi A4 Avant	1,9 Liter													
Skoda Octavia	1,9 Liter													
VW 1.9 TDI	1,9 Liter													
VW Golf 1.9 TDI	1,9 Liter													
<i>NOT IN</i>	FROM Auto WHERE Hubraum IN (SELECT Hubraum FROM Auto WHERE [Auto-Nr] LIKE "AU????")													

Es werden alle Fahrzeuge angezeigt, deren Hubraum **gleich** dem Hubraum eines der Fahrzeuge der Marke Audi ist. *IN* können Sie durch *NOT* verneinen.

EXISTS SELECT [Marke und Typ], [km-Preis], Tagespreis FROM Auto
WHERE NOT EXISTS (
SELECT [Auto-Nr] FROM Vermietungen
WHERE Auto.[Auto-Nr] = Vermietungen.[Auto-Nr])

Eine mit dem Operator *EXISTS* formulierte Unterabfrage ergibt *True*, wenn die Unterabfrage Werte liefert. Im Beispiel prüft die Unterabfrage, ob zu einer *Auto-Nr* ein Eintrag in der Tabelle **VERMIETUNGEN** existiert (erste Möglichkeit eines Inner-Joins, Seite 12). Für Fahrzeuge, die noch nicht vermietet wurden, liefert die Unterabfrage *False*. Über *NOT EXISTS* wird daraus *True*. Dadurch werden alle Fahrzeuge, die noch nicht vermietet wurden, angezeigt.

1.8 UNION-Abfragen

Mit dem Operator *UNION* vereinigen Sie die Daten von zwei oder mehr Tabellen oder Abfragen in **einem** Abfrageergebnis. Diesen Abfragetyp können Sie nicht im QbE-Editor entwerfen.

Übungen:
Autover-
mietung74

Aufbau

In der Datenbank **Autovermietung** werden Vermietungen, die länger als ein Jahr zurückliegen, in eine Tabelle **ARCHIV** verschoben (siehe Seite 17). Um die Vermietungen aus beiden Tabellen zu sehen, verwenden Sie den Operator *UNION*:

```
SELECT [Auto-Nr], Von, Bis, kmAnf, kmEnde FROM Vermietungen
UNION
SELECT [Auto-Nr], Von, Bis, kmAnf, kmEnde FROM Archiv
```

Wichtig!

Die einzelnen Abfragen einer *UNION*-Operation müssen die gleiche Anzahl von Feldern definieren. Feldgröße oder Datentyp müssen nicht zwingend übereinstimmen.

Das Schlüsselwort TABLE

Über TABLE lassen sich alle Felder einer Tabelle einbinden:

```
TABLE Vermietungen
UNION SELECT * FROM Archiv WHERE Von > #6/1/2005#
```

Die Tabelle zeigt alle Datensätze aus **VERMIETUNGEN** an sowie diejenigen aus **ARCHIV** mit Mietbeginn nach dem 1. Juni 2005. Alle Felder werden angezeigt.

Das Prädikat ALL

Der Operator *UNION* zeigt nur einen von mehrfach vorhandenen Datensätzen an. Sollen auch Duplikate angezeigt werden, verwenden Sie *UNION ALL*. Die Abfrage

```
SELECT Nachname FROM Kunde
UNION ALL SELECT Nachname FROM Kundenarchiv
```

zeigt Nachnamen auch dann an, wenn Sie mehrfach vorkommen.

Filtern, Gruppieren und Sortieren in UNION-Abfragen

Kriterien (*WHERE*) und Gruppierungen (*GROUP BY, HAVING*) müssen in jeder *SELECT*-Anweisung wiederholt werden. Die Sortierung (*ORDER BY*) der Datensätze erfolgt am Ende der letzten Abfrage.



1.9 Aktionsabfragen mit SQL

Im Gegensatz zu Auswahlabfragen, die lediglich eine bestimmte Sicht auf die Daten definieren, werden durch Aktionsabfragen Datensätze verändert.

Übersicht

Typ	Wirkung	SQL-Befehl
Aktualisierungsabfrage	Ändert die Feldinhalte von Datensätzen nach der im SQL-Ausdruck angegebenen Formel	<i>UPDATE</i>
Anfügeabfrage	Fügt Datensätze, die dem Kriterienausdruck entsprechen, an eine bestehende Tabelle an	<i>INSERT INTO</i>
Tabellenerstellungsabfrage	Fügt Datensätze, die dem Kriterienausdruck entsprechen, in eine neue Tabelle ein. Besteht die Tabelle bereits, werden die vorhandenen Datensätze überschrieben	<i>SELECT INTO</i>
Löschabfrage	Löscht alle Datensätze, die den angegebenen Kriterien entsprechen	<i>DELETE</i>

Aktualisierungsabfragen mit UPDATE

Aufbau

Der SQL-Ausdruck für eine Aktualisierungsabfrage wird vom *UPDATE*-Befehl eingeleitet. Über den Befehl *SET* wird mit einer Formel der neue Wert festgelegt. Wenn Sie nur bestimmte Felder ändern möchten, können Sie die Abfrage mit der *WHERE*-Klausel einschränken.

UPDATE Auto

UPDATE definiert die zu ändernde Tabelle | SET Tagespreis = Tagespreis*1.05
SET definiert neuen Wert | WHERE Art = "Cabrio"
WHERE setzt Kriterien

Das Beispiel setzt die Tagespreise aller Cabriolets um 5% nach oben.

Mehrere Felder ändern

Die Änderungsanweisungen für mehrere Felder setzen Sie durch Kommata getrennt hintereinander:

UPDATE AUTO

SET [km-Preis]=[km-Preis]*1.02, Tagespreis = Tagespreis*1.05
 WHERE Art = "Cabrio"

Der Kilometerpreis aller Cabrios wird um 2%, der Tagespreis um 5% erhöht.

Löschabfragen mit DELETE

Für Löschabfragen verwenden Sie in SQL-Ausdrücken die *DELETE*-Anweisung:

DELETE FROM Auto

DELETE FROM definiert die Tabelle | WHERE Art = "Cabrio"
WHERE setzt Kriterien

Die Abfrage löscht alle Cabriolets aus der Tabelle *AUTO*.

Alle Datensätze löschen

Wenn Sie die *DELETE*-Anweisung ohne Kriterien verwenden, werden alle Datensätze aus der angegebenen Tabelle gelöscht. Beispiel: *DELETE FROM Auto*

Übungen:
 Autover-
 mietung 74
 Lager-
 verwaltung.... 77



Tabellenerstellungsabfragen mit SELECT INTO

Der folgende SQL-Ausdruck kopiert die Nachnamen und Vornamen aller Kunden, deren Postleitzahl mit **8** beginnt, in eine neue Tabelle **Postleitzahl**:



```

SELECT Nachname, Vorname
  INTO Postleitzahl8
  FROM Kunde
 WHERE PLZ LIKE "8*"

```

Feldauswahl | *Zieltabelle* | *Quelltabelle* | *Kriterienausdruck*



Ist die Zieltabelle bereits vorhanden, wird sie nach dem Bestätigen der Warnmeldung mit den neuen Daten überschrieben.

Anfügeabfragen mit INSERT INTO

Mehrere Datensätze einfügen

Das Beispiel kopiert in eine bestehende Tabelle **ARCHIV** die Datensätze aus der Tabelle **VERMIETUNGEN**, bei denen das Rückgabedatum **Bis** vor dem 1. Juli 2006 liegt:



```

INSERT INTO Archiv
  SELECT *
  FROM Vermietungen
 WHERE Bis <= #6/30/2006#

```

Zieltabelle | *Feldauswahl* | *Quelltabelle* | *Kriterienausdruck*

Übungen:
Autovermietung74
Lagerverwaltung77

Wenn die Zieltabelle nicht existiert, wird eine Fehlermeldung ausgegeben.



Über die **SELECT**-Anweisung kann eine Feldauswahl vorgenommen werden. Der Ausdruck

```

INSERT INTO Postleitzahl8
SELECT Nachname, Vorname FROM Kunde WHERE PLZ LIKE "86*"

```

fügt nur den Nachnamen und Vornamen der Kunden in die bereits bestehende Tabelle **Postleitzahl8** ein.

Einen Datensatz einfügen mit VALUES

Über das Schlüsselwort **VALUES** können Sie einen einzelnen Datensatz mit Werten füllen:

```

INSERT INTO Vermietungen VALUES
(13, "AU002", "00012", #8/4/2006#, #8/12/2006#, 30450, 30800)

```

Rechnungs-Nr	Auto-Nr	Kunden-Nr	Von	Bis	kmAnf	kmEnde
00013	AU002	00012	04.08.2006	12.08.2006	30.450	30.800

Bei dieser Variante müssen alle Felder belegt werden. Die Feldinhalte werden in der gleichen Weise geschrieben wie die Kriterien bei der WHERE-Klausel (Seite 7).

Alternativ kann eine Liste der zu belegenden Felder definiert werden:

```

INSERT INTO Vermietungen
([Auto-Nr], [Kunden-Nr], Von, Bis, kmAnf, kmEnde)
VALUES ("AU002", "00012", #8/13/2006#, #8/15/2006#, 30800, 31200)

```

In diesem Beispiel füllt Access das Feld **Rechnungs-Nr** automatisch aus, da es vom Datentyp **Autowert** ist.



Index

- &**
- &-Operator 38
- A**
- Abfragen mit SQL 4
 - Access-Konstante 35
 - Aggregatfunktionen
 - SQL 11
 - VBA 62
 - Aktion 19
 - Aktionsargument 19
 - Aktualisierungsabfrage 16
 - ALL 6, 14, 15
 - AND 8
 - Anfügeabfragen 17
 - Ansichtenauswahl 31
 - Anweisung 38
 - ANY 14
 - Application 48
 - Argument 40, 49
 - Array 37
 - AS 10, 11
 - ASC 10
 - Ausdrucksgenerator 24
 - AutoExec-Makro 29
 - AutoKeys-Makro 28
 - AVG 11
- B**
- Bedingte Anweisung 44
 - Benannter Parameter... 40, 49
 - Benutzeroberfläche... ..78, 89
 - Bericht
 - VBA 61
 - Bericht öffnen 58
 - Bericht schließen 58
 - BETWEEN ... AND 8
 - Bookmark 69
- C**
- Call 33
 - CDbl 54
 - Choose 45
 - Close 58
 - Codefenster 31
 - COUNT 11
 - CurrentProject 56
- D**
- DAO 63
 - Data Access Objects 63
 - Data Manipulation Language ...*Siehe* DML
 - Data Query Language... ..*Siehe* DQL
 - Date 40
 - Datenauswertung (DAO).. 87
 - Datenfeld 37
 - Datenimport 82
 - Datensatz
 - Anzahl ermitteln 65
 - editieren 65
 - hinzufügen 65
 - löschen 59, 66
 - Navigation 59, 66
 - rückgängig 59
 - speichern 59
 - suchen 67
 - Datentyp 35
 - Datenzugriff 63
 - DCount 62
 - Debuggen 71
 - Deklaration 33, 34
 - erzwingen 34
 - DELETE 16
 - DESC 10
 - Dim 36
 - Direktbereich 72
 - Direktfenster 31
 - Dirty 68
 - DISTINCT 6
 - DISTINCTROW 6
 - DLookup 62
 - DML 4
 - Do ... Loop-Schleife 46
 - DoCmd 58, 59
 - Domänenfunktionen 62
 - DoMenuItem 59
 - DQL 4
 - Drucken 70
 - DSum 62
 - Dynaset 12
- E**
- Editor
 - SQL 4
 - VBA 30
 - Eigenschaften anzeigen 41
 - Eigenschaften...48, 51, 56, 64
 - Eigenschaftenfenster... 31, 51
 - Eingabedialog 43
 - Einzelschritt 71
 - Erase 37
 - Ereignis 23
 - BeforeUpdate 53
 - Beim Klicken 24
 - Click 54
 - Current 60
 - KeyUp 53
 - Load 57
 - Nach Aktualisierung 23
 - Resize 57
 - Vor Aktualisierung 26
 - Ereignisbehandlungsrouti ne 32, 49
 - Ereignisliste 31
 - Ereignisorientiertes Programmieren 49
 - Ereignisse
 - Bericht 61
 - Formular 56
 - Steuerelement 52
 - Eventhandler 32, 49
 - EXISTS 15
 - Exit Do 47
 - Exit For 47
 - Explizite Deklaration.... 34
- F**
- Fallauswahl 45
 - Farbwerte 51
 - Fehlerbehandlung 73
 - Fehlersuche 71
 - Feldinhalt aufteilen 86
 - Feldnamen in SQL 6
 - Filtern
 - DAO 67
 - Formular 69
 - VBA 60
 - FindFirst 67
 - FindNext 67
 - For ... Next-Schleife 46
 - FORMAT
 - SQL 10
 - VBA 40
 - Formel 38
 - Formular
 - DAO und SQL 68
 - Makro 22
 - VBA 56
 - Formular öffnen 58
 - Formular schließen 58



- Function39
 Funktion 30, 39, 40
 Fußgesteuerte Schleife .46
- G**
- Generator.....22
 GoToRecord.....59
 GROUP BY.....11
 Gruppieren11
 Gültigkeitsbereich ..33, 36
 Gültigkeitsdauer36
 Gültigkeitsprüfung89
- H**
- Haltepunkt31, 71
 HAVING11
 Hilfe
 Eigenschaften 41, 51
 Ereignisse 23
 Konstanten..... 41
 Makros 20
 Methoden..... 41
 Parameter..... 41
 Syntax 41
 VBA..... 70
 VBA-Editor 41
- I**
- If ... Then.....44
 IIF
 SQL..... 10
 VBA..... 45
 Import automatisieren...82
 IN..... 8, 15, 27
 InputBox43
 INSERT ... INTO17
 InStr40
 IstNull27
- J**
- JOIN
 Inner Join 12
 Left Join..... 13
 Outer Join 13
 Right Join 13
- K**
- Klassenmodul..... 30, 32
 Kombinationsfeld ...55, 69
 Konstante34
 Deklaration 33, 34
 Kontrollkästchen55
 Konvertierungsfunktion 35
 Kopfgesteuerte Schleife 46
- Kriterien
 Datentypen..... 7, 27
 in gruppierten Abfragen... 11
 verknüpfen..... 8
- L**
- Laufzeitfehler73
 LCase40
 Leere Felder9
 Left.....40
 Lesezeichen 31, 71
 LIKE9
 Listenfeld23, 55, 69
 durchsuchen..... 91
 sortieren 90
 Logische Fehler73
 Lokalfenster.....31, 72
 Löschabfrage16
 LTrim40
- M**
- Makro
 Aktion 19
 Aktionsargument 19
 an Ereignis binden..... 26
 ausführen 21, 23
 AutoExec-Makro 29
 AutoKeys-Makro..... 28
 bearbeiten 20
 Bedingungen..... 26
 Drag & Drop..... 20
 erstellen..... 19, 22
 Fehlermeldungen 29
 in Formularen/Berichten .. 22
 konvertieren in VBA 25
 speichern 20
 starten..... 21, 23
 starten aus Menü..... 21
 starten über Schaltfläche .. 21
 starten über
 Tastenkombination 21
 Startmakro 29
 Steuerelementzugriff .. 23, 24
 Tastenbelegung..... 28
 verwenden..... 23
 Verzweigungen..... 26
 Makroentwurf.....19
 Makrogruppe25
 MAX11
 Me50, 56
 Meldungsfenster42
 Rückgabewert..... 43
 Menü erstellen21
 Methoden 48, 50, 64
 Methoden anzeigen.....41
 Mid.....40
 MIN11
 Modul30
 Klassenmodul 30
 Standardmodul..... 30
 MsgBox 42
 Rückgabewert 43
- N**
- Name
 Eventhandler 49
 Konstante 34
 Prozedur 33
 Steuerelement 50
 Variable..... 34
 Netto_zu_Brutto
 Prozedur 32
 NewRecord..... 68
 NICHT..... 27
 NoMatch..... 67
 NOT8, 15
 Now 40
 NULL 9
 NZ
 SQL..... 10
 VBA 86, 93, 96
- O**
- Objekt
 Einzelobjekt 48
 Objekt in Auflistung 48
 Objektkatalog 49
 Objektliste 31
 Objektvariable 63
 OpenForm..... 58
 OpenReport 58
 Operator
 arithmetisch 38
 Boolesch 38
 Logisch 38
 Vergleich..... 38
 Zeichenverkettung 38
 OPTION EXPLICIT 34
 Optional..... 39
 Optionaler Parameter...39,
 40, 49
 Optionsfeld 54
 Optionsgruppe 54
 Optionswert 54
 OR 8
 ORDER BY 10
- P**
- Parameter
 benannt..... 40, 49
 optional 39, 40, 49
 Parameterinfo 41
 PERCENT 10
 Platzhalter..... 9
 Preserve 37



- Private 33, 36
 Programm testen 71
 Programmierhilfen 41
 Projekt-Explorer..... 30
 Prozedur 30, 32, 33
 Aufbau 33
 aufrufen..... 33, 39
 Public 33, 36
- Q**
- QbE-Bereich 4
 QuickInfo 41
- R**
- Recordset 63, 65
 RecordsetClone 69
 RecordSource..... 69
 Redim..... 37
 RGB-Funktion 51
 Right 40
 RowSource..... 69
 RTrim..... 40
 RunCommand 59
- S**
- Schleife
 Bedingungsschleife 46
 rückwärts zählen..... 46
 Schleife verlassen..... 47
 Schrittweite 46
 Until 47
 While..... 47
 Zählschleifen 46
 Select ... Case 45
 SELECT ... FROM..... 6
 SELECT ... INTO 17
 Sicherheitswarnung 18
 Sort 66
 Sortieren
 DAO..... 66
 Formular 69
 SQL..... 10
 VBA..... 60
 SQL..... 4
 Abfrage ausführen 4
 Abfrage erstellen 4
 Aggregatfunktionen..... 11
 Aktionsabfragen 16
 Anwendungen..... 5
- Berechnete Felder 11
 Datensätze gruppieren 11
 Editor..... 4
 Ergebnis formatieren 10
 Feldnamen..... 6
 in VBA verwenden..... 68
 Kriterien verknüpfen 8
 Selektieren 7
 Sortieren..... 10
 Sortierreihenfolge 10
 Suche mit Platzhaltern..... 9
 Tabellen verknüpfen..... 12
 Top-Ten Filter..... 10
 UNION-Abfrage..... 15
 Unterabfrage 14
 Vergleichsoperatoren..... 7
 Verknüpfungsoperator..... 8
- Standardmodul 30, 32
 Startformular 80
 Startmakro 29
 Static 36
 Steuerelement
 Ausdrucksgenerator..... 24
 Name..... 23, 24
 Zugriff 23, 24, 50
- Structured Query
 Language *Siehe SQL*
 Suche in Listenfeld..... 91
- Suchen
 DAO..... 67
 Formular..... 69
- SUM..... 11
 Switch 45
 Symbolleiste erstellen .. 21
 Syntaxfehler 73
 Syntaxüberprüfung 41
- T**
- Tabellenerstellungsabfrage
 e 17
 Tastenbelegungsmakro. 28
 Tastenkombinationen über
 Makro 28
 Textfeld..... 53
 TOP 10
 Trim 40
- U**
- Überwachungsfenster...
 31, 72
- UCase..... 40
 Umwandlungsfunktion .35
 Undo..... 59
 UNION-Abfrage 15
 Unterabfrage 14
 Unterformular..... 80
 Until 47
 UPDATE..... 16
- V**
- VALUES 17
 Variable 34
 Deklaration..... 33, 34
 zurücksetzen 36
 Variant 34
 VBA-Editor..... 30
 VBA-Hilfe..... 70
 Vergleichsoperator ... 7, 38
 Verknüpfungsoperator... 8,
 27
 Verzweigung
 einseitige Auswahl..... 44
 Fallauswahl 45
 mehrseitige Auswahl 45
 verschachteln 44
 zweiseitige Auswahl..... 44
 Verzweigungsfunktion .45
 Visual-Basic-Konstante 35
- W**
- WHERE 7
 WhereCondition 58
 While..... 47
 With-Anweisung 55
 Wortvervollständigung .41
- X**
- XOR 8
- Z**
- Zählschleifen 46
 Zeichenverkettung 38
 Zeilenumbruch (Code) .33
 ZWISCHEN ... UND.... 27