

# IV. Datenbankmanagement

## Kapitel 2: Datenmanipulationssprache SQL

- 1. Einleitung und Grundelemente (Select, From, Where)
- 2. Funktionen
- 3. JOINS
- 4. Weitere Arten von SQL-Befehlen
  - (Create, Insert, Update, Delete, Drop)

- SQL - Structured Query Language
  - Mitte der 70er Jahre entwickelt
  - Zurzeit Standard für relationale Datenbanken:
    - o ANSI (American National Standards Institute)
    - o ISO (International Standardization Organization),
    - o Norm für SQL 2 liegt vor, Weiterentwicklung SQL 3
  - Nicht-prozedurale, deskriptive Datenbanksprache
    - o Eine SQL-Anfrage drückt aus, wie das gewünschte Ergebnis aussehen soll, und nicht, wie das Resultat ermittelt werden soll.

- Struktur der Grundelemente
  - Struktur:
    - o SELECT Attribut(e)
    - o FROM Relation(en)
    - o [ WHERE Bedingung ]
    - o [ GROUP BY Attribut(e) ]
    - o [ ORDER BY Attribut(e) ]
  - Bedeutung:
    - o Suche Attributwerte
    - o in Relation(en)
    - o [ wobei gilt: Bedingung ]
    - o [ Aggregation nach ... ]
    - o [ Sortierung nach ... ]

**SELECT** \* ← *Alle Spalten*  
**FROM** Produkte ← *Tabelle „Produkte“*  
**ORDER BY** ID ← *Sortierung über Spalte „ID“*

ID	Produktname	Farbe	Artikelnummer	Verkaufspreis	Einkaufspreis	Lagerbestand	Verkaufte Einheiten	Ort
1	Monitor 17"	weiß	1297812542	399,00	249,99	50	134	Frankfurt
2	Monitor 19"	schwarz	2457897145	499,00	379,00	12	289	Berlin
3	Monitor 17"	schwarz	1297467815	405,00	249,99	25	124	Frankfurt
4	Monitor 19"	weiß	2459871327	509,00	389,99	150	12	Frankfurt
5	Monitor 20"	schwarz	2789441512	799,00	599,00	520	1052	Berlin
6	Monitor 20"	weiß	2799151424	829,00	549,99	100	26	Berlin
7	Monitor 20"	anthrazit	2764657527	819,00	589,99	50	127	Nürnberg
8	Monitor 21"	anthrazit	2845161215	999,00	799,99	100	279	Hamburg
9	Monitor 24"	weiß	2945712415	1299,00	945,00	25	124	Berlin
10	Monitor 24"	schwarz	2955745742	1350,00	956,00	450	1024	Hamburg
...								

```
SELECT ID, Ort, Lagerbestand  
FROM Produkte  
ORDER BY ID
```

ID	Ort	Lagerbestand
1	Frankfurt	50
2	Berlin	12
3	Frankfurt	25
4	Frankfurt	150
5	Berlin	520
6	Berlin	100
7	Nürnberg	50
8	Hamburg	100
9	Berlin	25
10	Hamburg	450
...	...	...

**SELECT** \*  
**FROM** Produkte  
**WHERE** Einkaufspreis > 550 AND Ort = 'Berlin'

ID	Produktname	Farbe	Artikelnummer	Verkaufspreis	Einkaufspreis	Lagerbestand	Verkaufte Einheiten	Ort
5	Monitor 20"	schwarz	2789441512	799,00	599,00	520	1052	Berlin
6	Monitor 20"	weiß	2799151424	829,00	549,99	100	26	Berlin
9	Monitor 24"	weiß	2945712415	1299,00	945,00	25	124	Berlin

**SELECT** Ort, **SUM**(Lagerbestand)  
**FROM** Produkte  
**GROUP BY** Ort

Ort	SUM(Lagerbestand)
Frankfurt	225
Berlin	657
Nürnberg	50
Hamburg	550
...	

$$225 = 50 + 25 + 150$$

$$657 = 12 + 520 + 100 + 25$$

$$50$$

$$550 = 100 + 450$$

**SELECT**      Benutzername  
**FROM**        Kundenstammdaten  
**WHERE**        Geburtsdatum > '1974-12-31' AND  
                  Geburtsdatum < '1976-01-01'

Benutzername
wonne
nightmoon
schubby
sommergeflüster

Das Datum wird in einfachen Hochkommata angegeben.

Syntax: 'JJJJ-MM-TT'

JJJJ = Jahr = 2007

MM = Monat = 12

TT = Tag = 24

Strings werden ebenfalls in Hochkommata angegeben.

Beispiel: ... Where Hobby = 'Schwimmen'

Zahlen werden ohne Hochkommata angegeben.

Beispiel: ... Where Alter = 23

- 1. Einleitung und Grundelemente (Select, From, Where)
- 2. Funktionen
- 3. JOINS
- 4. Weitere Arten von SQL-Befehlen
  - (Create, Insert, Update, Delete, Drop)

Weitere Funktionen, die im SELECT-Statement angegeben werden können:

<b>Funktion</b>	<b>Beschreibung</b>
AVG(Spalte)	Liefert den Durchschnittswert einer Spalte zurück
COUNT(Spalte)	Liefert die Anzahl der Zeilen in einer Spalte zurück
MAX(Spalte)	Liefert den höchsten Wert einer Spalte zurück
MIN(Spalte)	Liefert den niedrigsten Wert einer Spalte zurück
...	

**SELECT**      **AVG**(Lagerbestand)  
**FROM**        Produkt

Lagerbestand
153,2
...

$$((50 + 12 + 25 + 150 + 520 + 100 + 50 + 100 + 25 + 450) / 10)$$

**SELECT**      **MAX**(Lagerbestand)  
**FROM**        Produkt

Lagerbestand
520
...

- 1. Einleitung und Grundelemente (Select, From, Where)
- 2. Funktionen
- 3. JOINS
- 4. Weitere Arten von SQL-Befehlen
  - (Create, Insert, Update, Delete, Drop)

- JOINS werden verwendet, um Verknüpfungen zwischen zwei oder mehreren Tabellen zu realisieren.
- Tabellen werden durch die Verwendung von Schlüsseln miteinander in Bezug gesetzt.
- Ein Primärschlüssel ist eine Spalte mit einem eindeutigen Wert für jede Zeile.
- Folgende Formen können u.a. unterschieden werden:
  - **INNER JOIN**
  - LEFT JOIN
  - RIGHT JOIN

## Tabelle „Produkt“

ID	Produktname	Farbe	Artikelnummer	Verkaufspreis	Einkaufspreis	Lagerbestand	Verkaufte Einheiten	Ort
1	Monitor 17"	weiß	1297812542	399,00	249,99	50	134	Frankfurt
2	Monitor 19"	schwarz	2457897145	499,00	379,00	12	289	Berlin
3	Monitor 17"	schwarz	1297467815	405,00	249,99	25	124	Frankfurt
4	Monitor 19"	weiß	2459871327	509,00	389,99	150	12	Frankfurt
5	Monitor 20"	schwarz	2789441512	799,00	599,00	520	1052	Berlin
..								

## Tabelle „Details“

ID	Artikelnummer	Gewicht	Auflösung	Stromverbrauch
1	1297812542	4	1280X1024	26
2	2457897145	5	1280X1024	29
3	1297467815	4	1280X1024	27
4	2459871327	5,5	1280X1024	34
5	2789441512	8	1600x1280	53
..				

**SELECT** Produkt.Produktname, Details.Gewicht  
**FROM** Produkt **INNER JOIN** Details **ON**  
Produkt.Artikelnummer = Details.Artikelnummer

Produktname	Gewicht
Monitor 17"	4
Monitor 19"	5
Monitor 17"	4
Monitor 19"	5,5
Monitor 20"	8

Der INNER JOIN liefert nur diejenigen Zeilen zurück, bei denen eine Übereinstimmung der Spalten innerhalb der Join-Bedingung besteht.

- 1. Einleitung und Grundelemente (Select, From, Where)
- 2. Funktionen
- 3. JOINS
- 4. Weitere Arten von SQL-Befehlen
  - (Create, Insert, Update, Delete, Drop)

## INSERT INTO VALUES

Details  
(2689875627,6,1280x1024,55)

### Tabelle „Details“

ID	Artikelnummer	Gewicht	Auflösung	Stromverbrauch
1	1297812542	4	1280X1024	26
2	2457897145	5	1280X1024	29
3	1297467815	4	1280X1024	27
4	2459871327	5,5	1280X1024	34
5	2789441512	8	1600x1280	53
6	2689875627	6	1280X1024	55
..				

**INSERT INTO**

Details (Artikelnummer, Gewicht,  
Auflösung, Stromverbrauch)  
(2689875627,6,1280x1024,55)

**VALUES**

Tabelle „Details“

ID	Artikelnummer	Gewicht	Auflösung	Stromverbrauch
1	1297812542	4	1280X1024	26
2	2457897145	5	1280X1024	29
3	1297467815	4	1280X1024	27
4	2459871327	5,5	1280X1024	34
5	2789441512	8	1600x1280	53
6	2689875627	6	1280X1024	55
..				

**UPDATE**      Details  
**SET**          Gewicht = 12  
**WHERE**        Artikelnummer = 2689875627

Tabelle „Details“

ID	Artikelnummer	Gewicht	Auflösung	Stromverbrauch
1	1297812542	4	1280X1024	26
2	2457897145	5	1280X1024	29
3	1297467815	4	1280X1024	27
4	2459871327	5,5	1280X1024	34
5	2789441512	8	1600x1280	53
6	2689875627	12	1280X1024	55
..				

**UPDATE**      Details  
**SET**            Gewicht = 12, Auflösung = '1800x1400'  
**WHERE**        Artikelnummer = 2689875627

Tabelle „Details“

ID	Artikelnummer	Gewicht	Auflösung	Stromverbrauch
1	1297812542	4	1280X1024	26
2	2457897145	5	1280X1024	29
3	1297467815	4	1280X1024	27
4	2459871327	5,5	1280X1024	34
5	2789441512	8	1600x1280	53
6	2689875627	12	1800X1400	55
..				

```
CREATE TABLE table_name
(
  column_name1 data_type,
  column_name2 data_type,
  .....
)
```

## Datentypen

Datentyp	Bedeutung
integer(size)	Integer, „size“ gibt die Anzahl der Stellen an
decimal(size,d)	Dezimal, „size“ gibt die Anzahl der Stellen und „d“ die Anzahl der Nachkommastellen an
char(size)	Anzahl von Zeichen einer fixen Länge „size“
varchar(size)	Anzahl von Zeichen bis zu einer Länge „size“
date(yyyymmdd)	Annahme eines Datums z.B. 20070115
...	

```
CREATE TABLE Kunde  
(  
  Name varchar(30),  
  Vorname varchar(15),  
  .....  
)
```

Tabelle „Kunde“

ID	Name	Vorname

**DELETE FROM  
WHERE**

Details  
Artikelnummer = 2689875627

Tabelle „Details“

ID	Artikelnummer	Gewicht	Auflösung	Stromverbrauch
1	1297812542	4	1280X1024	26
2	2457897145	5	1280X1024	29
3	1297467815	4	1280X1024	27
4	2459871327	5,5	1280X1024	34
5	2789441512	8	1600x1280	53
..				

gelöscht

6	2689875627	12	1280X1024	55
---	------------	----	-----------	----

**DROP Table** Tabellenname

Löscht eine Tabelle in der Datenbank.

**DROP Database** Datenbankname

Löscht die gesamte Datenbank auf dem Datenbankserver.

Zu Übungszwecken kann die E-Learning-Plattform des ISE für SQL verwendet werden:

SQL Spielwiese · Professur für Information Systems Engineering

```

/* SELECT: Ausgabe aller Konten und Guthaben pro Kunde */
SELECT C.customer_name, D.account_number, A.balance
FROM customer AS C
JOIN deposit AS D ON C.customer_name=D.customer_name
JOIN account AS A ON D.account_number=A.account_number
ORDER BY C.customer_name
    
```

**Customer\_name**    **account\_number**    **balance**

Hayes	A-101	500.00
Hayes	A-102	400.00
Hayes	A-217	750.00
Johnson	A-101	500.00
Jones	A-201	900.00
Lindsay	A-217	750.00
Majors	A-333	700.00
Smith	A-215	850.00
Smith	A-444	700.00
Turner	A-305	350.00

[www.ise.wiwi.uni-frankfurt.de/spielwiesen](http://www.ise.wiwi.uni-frankfurt.de/spielwiesen)