

Handout zur Unit
Datenbanken und SQL



Web-Technologien
SQL

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

1



Prof. Dr. rer. nat.
Nane Kratzke

*Praktische Informatik und
betriebliche Informationssysteme*

- **Raum:** 17-0-10
- **Tel.:** 0451 300 5549
- **Email:** nane.kratzke@fh-luebeck.de

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

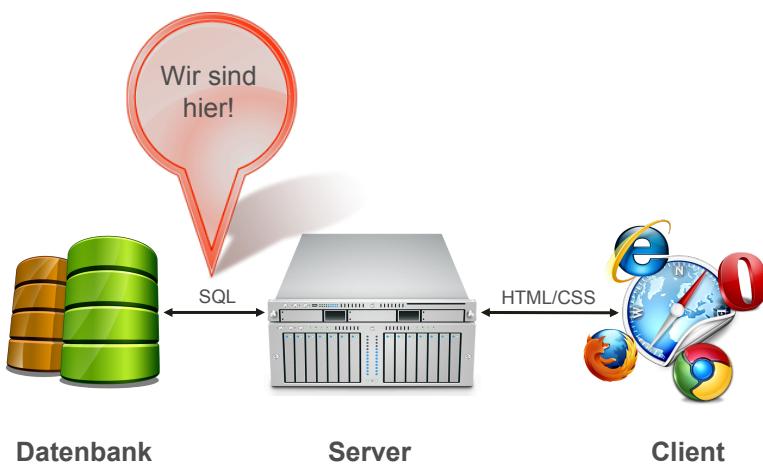
2

Datenbank – Server – Client

Wo waren wir nochmal?



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

3

Zum Nachlesen ...



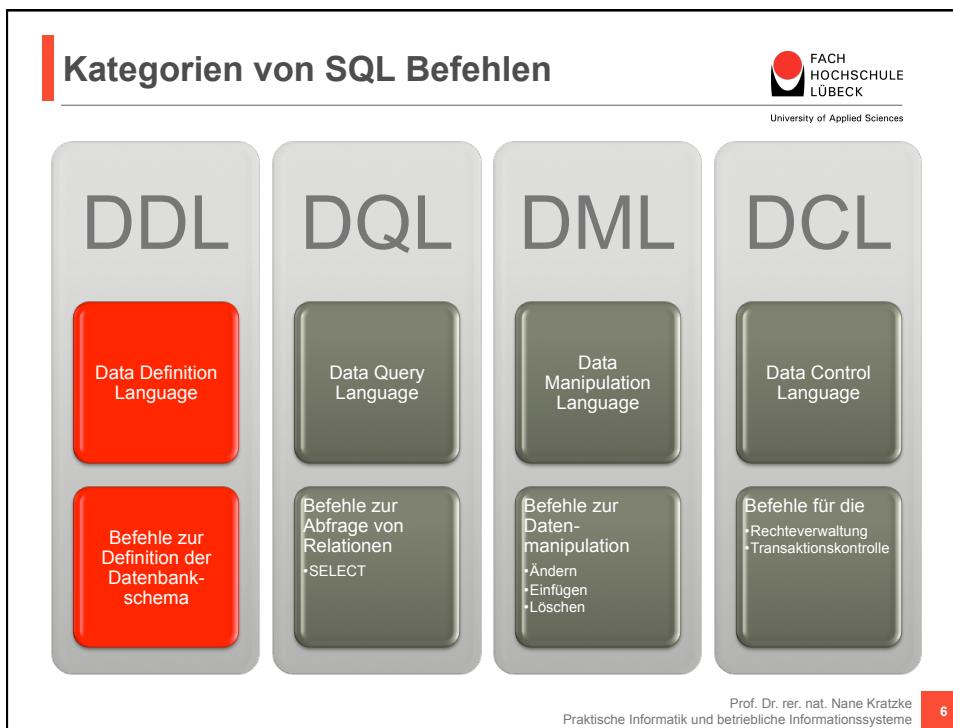
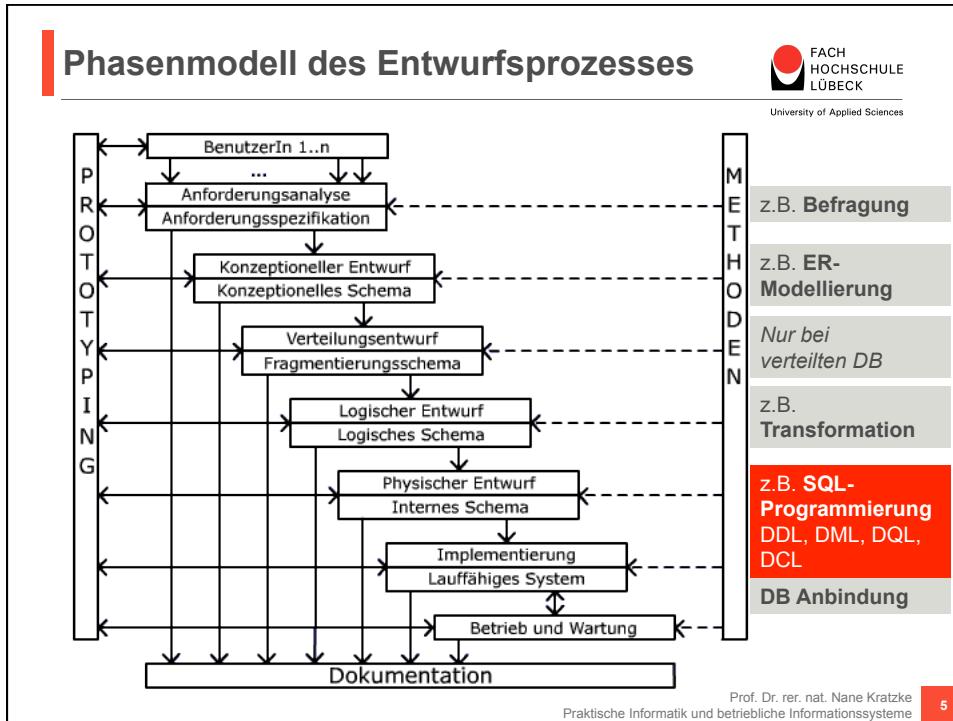
University of Applied Sciences



Kapitel 6
SQL – Der relationale Datenbankstandard

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

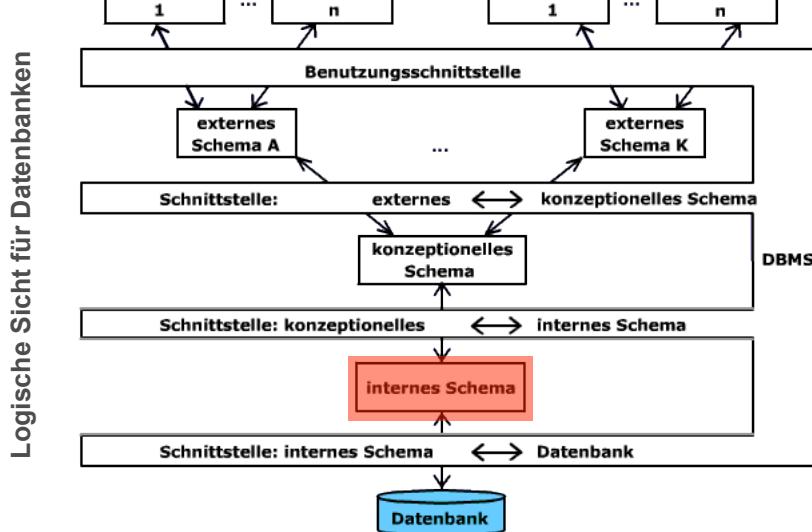
4



Wo sind wir in der Drei-Ebenen-Schemaarchitektur?



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke

7

Data Definition Language create



University of Applied Sciences

Anlegen einer Datenbank

hier Datenbank mit Namen *Bibliothek*

```
CREATE DATABASE Bibliothek;
```

Anlegen einer Tabelle

Hier Tabelle über alle Bücher mit Attributen *ISBN*, *Titel*, *Exemplare*, *Leihfrist*.

```
CREATE TABLE Buch (
    ISBN VARCHAR(20) NOT NULL,
    Titel VARCHAR(100),
    Exemplare NUMBER(3,0) NOT NULL,
    Leihfrist Frist);
```

Anlegen eines Datentyps

hier zweistellige Frist mit Standardwert 30 und zulässigen Werten von 15, 30 und 60.

```
CREATE DOMAIN Frist AS NUMBER(2,0)
DEFAULT 30
CONSTRAINT UngueltigeFrist CHECK
    (VALUE IN (15, 30, 60));
```

Prof. Dr. rer. nat. Nane Kratzke

8

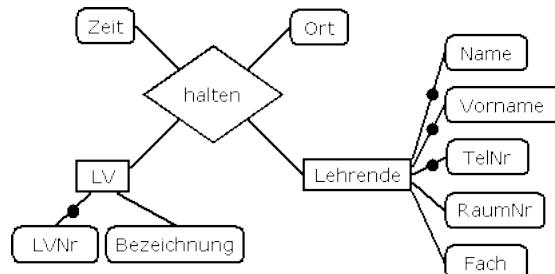
Data Definition Language Erweiterte `create table` Anweisung



University of Applied Sciences

```
CREATE TABLE [Database.]Relname (
Attributdefinition, ...,
[PRIMARY KEY (Keys...),]
[FOREIGN KEY (Attr...) REFERENCES Relname(Keys...),]
[CONSTRAINT Regelname (CHECK|UNIQUE)-Bedingung]);
```

Veranschaulichung
an folgendem
Beispiel



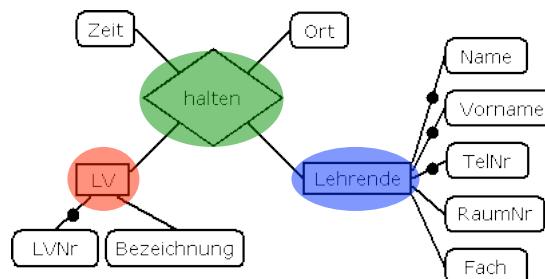
Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

9

Transformation des ER-Modells in Relationenschema



University of Applied Sciences



LV(LVNr, Bezeichnung)

Lehrende(Name, Vorname, TelNr,
RaumNr, Fach)

halten(LVNr -> LV.LVNr,
Name -> Lehrende.Name,
Vorname -> Lehrende.Vorname,
TelNr -> Lehrende.TelNr,
Zeit, Ort)

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

10

Übersetzen der Relationenschema in **create table** Statements



University of Applied Sciences

LV(LVNr, Bezeichnung)

```
CREATE TABLE LV (
  LVNr NUMBER(4,0),
  Bezeichnung VARCHAR(32),
  PRIMARY KEY LVNr);
```

Lehrende(Name, Vorname, TelNr, RaumNr, Fach)

```
CREATE TABLE Lehrende (
  Name VARCHAR(32),
  Vorname VARCHAR(32),
  TelNr VARCHAR(32),
  RaumNr VARCHAR(16),
  Fach VARCHAR(32),
  PRIMARY KEY (Name, Vorname, TelNr);
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

11

Übersetzen der Relationenschema in **create table** Statements



University of Applied Sciences

*halten(LVNr -> LV.LVNr, Name -> Lehrende.Name,
Vorname -> Lehrende.Vorname, TelNr -> Lehrende.TelNr,
Zeit, Ort)*

```
CREATE TABLE halten
  LVNr NUMBER(4,0),
  Name VARCHAR(32),
  Vorname VARCHAR(32),
  TelNr VARCHAR(32),
  Zeit TIME,
  Ort VARCHAR(32),
  PRIMARY KEY (LVNr, Name, Vorname, TelNr),
  FOREIGN KEY (LVNr REFERENCES LV(LVNr)),
  FOREIGN KEY (Name REFERENCES Lehrende(Name)),
  FOREIGN KEY (Vorname REFERENCES Lehrende(Vorname)),
  FOREIGN KEY (TelNr REFERENCES Lehrende(TelNr))
);
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

12

Weitere DDL Befehle und Klauseln



University of Applied Sciences

• Erzeugen von Indizes
• Zur Beschleunigung von Queries

CREATE INDEX

• Änderungen von Relationen
• Löschen und Ergänzen von Attributen

ALTER

• Prüfung von Konsistenzbedingungen
• Sicherstellen das Fremdschlüssel nur auf ein Element zur Zeit verweisen

CHECK/ UNIQUE

• Löschen von Tabellen

DROP

Hinweis: Eine Erläuterung aller SQL-DDL-Befehle und – Klauseln finden Sie in Ihrem Online Skript.



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

13

Kategorien von SQL Befehlen



University of Applied Sciences

DDL

Data Definition Language

Befehle zur Definition der Datenbankschema

DQL

Data Query Language

Befehle zur Abfrage von Relationen
•SELECT

DML

Data Manipulation Language

Befehle zur Datenmanipulation
•Ändern
•Einfügen
•Löschen

DCL

Data Control Language

Befehle für die
•Rechteverwaltung
•Transaktionskontrolle

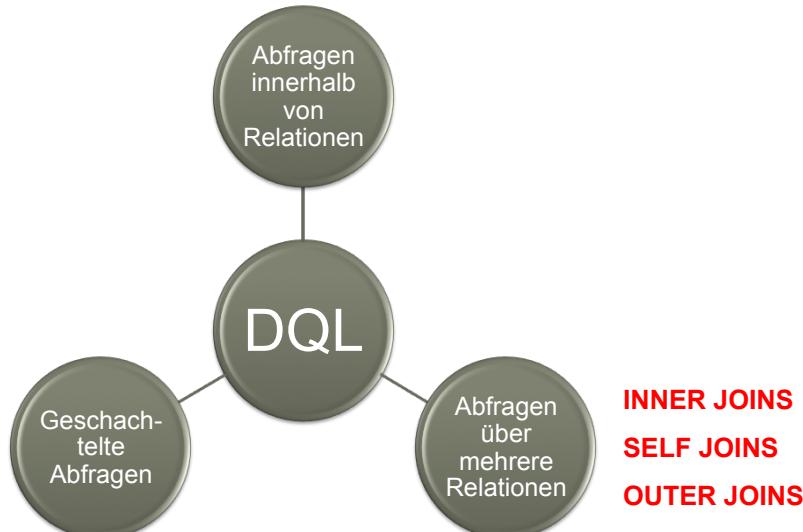
Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

14

Data Query Language SQL SELECT Klausel



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

15

Abfragen innerhalb von Relationen



University of Applied Sciences

Grundmuster einer Query

```
SELECT Attr, Attr, ...      // Selektion der Spalten
FROM Relation                // Angabe der Relation
WHERE Bedingung              // Selektion der Zeilen
[ORDER BY Attr [ASC|DESC]]; // Sortierung
```

Hinweis: In Ihrem Skript „SQL Programmierung“ im Kapitel 2.3 und 4.3 finden Sie eine kurze und knappe Übersicht wie WHERE Klauseln zu formulieren sind.

Die WHERE Klausel ist vermutlich die wichtigste Klausel der DQL.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

16

Abfragen innerhalb von Relationen Beispiel



University of Applied Sciences

Buch	Autor	Titel	Seitenzahl
	A. Diavolo	Die Pest	21
	M. Engels	Der Himmel	19
	F. Marx	Das Kapital	1005
	M. Muster	Die Hölle	235

```
SELECT Titel, Autor
FROM Buch
WHERE Seitenzahl > 20 AND
      NOT Titel = 'Die Pest';
```

Autor	Titel
F. Marx	Das Kapital
M. Muster	Die Hölle

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

17

Abfragen über mehrere Relationen INNER JOINS



University of Applied Sciences

Im Allgemeinen werden bei einer Datenbankanfrage Informationen aus mehreren Relationen benötigt und zusammengestellt.

Hier sehen sie das Grundmuster, wie dies in SQL mittels einer SELECT Klausel ausgedrückt werden kann.

Grundmuster eines INNER JOINS

```
SELECT r1.Attrs, ..., r2.Attrs
FROM Rel1 AS r1, Rel2 AS r2, ...
WHERE Bedingung;
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

18

**Handout zur Unit
Datenbanken und SQL**

Abfragen über mehrere Relationen INNER JOINS - Beispiel

University of Applied Sciences

Buch	Autor	Titel
	A. Diavolo	Die Pest
	M. Engels	Der Himmel
	F. Marx	Das Kapital
	M. Muster	Die Hölle

liest	Titel	MatrNr
	Die Pest	12345
	Der Himmel	12345
	Die Pest	43215
	Die Pest	53421

Studenten	MatrNr	Name
	12345	A. Meier
	43215	B. Müller
	53421	C. Kunze
	111335	D. Huber

```
SELECT S.Name, B.Autor
FROM Buch as B, liest as l,
      Studenten as S
WHERE S.MatrNr = l.MatrNr AND
      l.Titel = B.Titel;
```

Name	Autor
A. Meier	A. Diavolo
A. Meier	M. Engels
B. Müller	A. Diavolo
C. Kunze	A. Diavolo

Prof. Dr. rer. nat. Nane Kratzke | 19

Praktische Informatik und betriebliche Informationssysteme

Abfragen über ein und dieselbe Relation SELF JOINS

University of Applied Sciences

Studierende	Name	GebDat
	Meier	07.10.85
	Schulze	03.05.80
	König	07.10.85
	Baum	12.11.82
	Dreier	25.02.84
	Hesse	07.10.85

Grundmuster eines SELF JOINS

```
SELECT a.Attrs, ...
FROM Rel AS a, Rel AS b
WHERE Bedingung;
```

```
SELECT DISTINCT a.Name, a.GebDat
FROM Studierende AS a,
      Studierende AS b
WHERE a.GebDat = b.GebDat AND
      a.Name != b.Name
ORDER BY GebDat;
```

Name	GebDat
Meier	07.10.85
König	07.10.85
Hesse	07.10.85

Prof. Dr. rer. nat. Nane Kratzke | 20

Praktische Informatik und betriebliche Informationssysteme

Abfragen über mehrere Relationen **OUTER JOINS**



University of Applied Sciences

Neben den Inner-Joins gibt es auch noch die Outer-Joins. Ein Outer-Join übernimmt im Gegensatz zum Inner-Join auch die Datensätze aus den Ausgangstabellen, die nicht in beiden Tabellen Entsprechungen besitzen.

Grundmuster eines OUTER JOINS

```
SELECT LR.Attrs, ..., RR.Attrs
FROM Leftrelation AS LR
{LEFT|RIGHT|FULL} OUTER JOIN Rightrelation AS RR
ON Bedingung;
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

21

OUTER JOINS **Beispiel RIGHT OUTER JOIN**



University of Applied Sciences

liest

Titel	MatrNr
Die Pest	12345
Der Himmel	12345
Die Pest	43215
Die Pest	53421

Studenten

MatrNr	Name
12345	A. Meier
43215	B. Müller
53421	C. Kunze
111335	D. Huber

```
SELECT S.Name, l.Titel
FROM liest AS l
RIGHT OUTER JOIN Studenten AS S
ON S.MatrNr = l.MatrNr;
```

Name	Titel
A. Meier	Die Pest
A. Meier	Der Himmel
B. Müller	Die Pest
C. Kunze	Die Pest
D. Huber	

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

22

Geschachtelte Abfragen Bsp.: Gleicher Geburtstag mit einer Person



University of Applied Sciences

Studierende

Name	GebDat
Meier	07.10.85
Schulze	03.05.80
König	07.10.85
Baum	12.11.82
Dreier	25.02.84
Hesse	07.10.85

```
SELECT Name, GebDat
FROM Studierende
WHERE GebDat IN
(SELECT GebDat
FROM Studierende
WHERE Name = 'Hesse')
ORDER BY GebDat;
```

GebDat
07.10.85

Hinweis: Unterabfragen und Joins können oft dasselbe Ergebnis zurückliefern und sind in der Anwendung ähnlich komplex, so dass es häufig dem persönlichen Geschmack überlassen ist, welche Variante man nutzt.

Name	GebDat
Meier	07.10.85
König	07.10.85
Hesse	07.10.85

Prof. Dr. rer. nat. Nane Kratzke | 23
Praktische Informatik und betriebliche Informationssysteme

Weitere SELECT-Klauseln



University of Applied Sciences

DISTINCT

- Keine Doppelausgabe identischer Tupel

ORDER BY

- Sortieren nach bestimmten Kriterien

Hinweis: Eine Erläuterung aller SELECT-Klauseln finden Sie in Ihrem Online Skript.

GROUP BY und HAVING

- Zusammenfassen und ggf. Filtern von Datensätzen mit gleichen Wertevorkommen in einem Attribut

Aggregationsfunktionen

- COUNT, SUM, MAX, MIN, AVG



Prof. Dr. rer. nat. Nane Kratzke | 24
Praktische Informatik und betriebliche Informationssysteme

Kategorien von SQL Befehlen



University of Applied Sciences

DDL

Data Definition Language

Befehle zur
Definition der
Datenbankschema

DQL

Data Query Language

Befehle zur
Abfrage von
Relationen
•SELECT

DML

Data Manipulation Language

Befehle zur
Daten-
manipulation
•Ändern
•Einfügen
•Löschen

DCL

Data Control Language

Befehle für die
•Rechteverwaltung
•Transaktionskontrolle

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

25

Data Manipulation Language



University of Applied Sciences

Insert

DML

Delete

Update

Der DML-Anteil von
SQL dient dazu Daten
einer Daten
• hinzuzufügen
• zu ändern
• und zu löschen

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

26

INSERT INTO



University of Applied Sciences

Grundmuster einer INSERT Klausel:

```
INSERT INTO Relation(Attrs, ...)
VALUES (Vals, ...);
```

```
INSERT INTO Kunde(KNr, Name)
VALUES (3, 'C. Meier');
```

Zweite Variante: Kopieren aus einer anderen Tabelle mittels **SELECT**

NeuKNr	Name
3	C. Meier
...	...

```
INSERT INTO Kunde(KNr, Name)
SELECT NeuKNr AS KNr, Name
FROM Neukunden;
```

KNr	Name
1	A. Müller
2	B. Kunze
3	C. Meier

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

27

UPDATE



University of Applied Sciences

Grundmuster einer UPDATE Klausel:

```
UPDATE Relation
SET Attr = Wert
WHERE Bedingung;
```

Der **UPDATE** Befehl ist dazu gedacht, in der Datenbank befindliche Datensätze zu verändern. Die zu ändernden Datensätze werden durch eine **WHERE** Klausel ausgewählt.

KNr	Name
1	Müller
2	Kunze
3	Meier

```
UPDATE Kunde
SET Name = 'Mayer'
WHERE Name LIKE 'M%';
```

KNr	Name
1	Mayer
2	Kunze
3	Mayer

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

28

DELETE



University of Applied Sciences

Grundmuster einer DELETE Klausel:

```
DELETE FROM Relation
WHERE Bedingung;
```

Mit dem **DELETE** Befehl können vorhandene Datensätze aus einer Relation gelöscht werden. Es werden immer komplette Datensätze (Zeilen) gelöscht. Die zu löschen Datensätze werden durch eine **WHERE** Klausel ausgewählt.

KNr	Name
1	Müller
2	Kunze
3	Meier



```
DELETE FROM Kunde
WHERE Name LIKE ,M%';
```

KNr	Name
4	Müller
2	Kunze
3	Meier

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

29

Kategorien von SQL Befehlen



University of Applied Sciences

DDL

Data Definition Language

Befehle zur Definition der Datenbankschema

DQL

Data Query Language

Befehle zur Abfrage von Relationen
•SELECT

DML

Data Manipulation Language

Befehle zur Datenmanipulation
•Ändern
•Einfügen
•Löschen

DCL

Data Control Language

Befehle für die
•Rechteverwaltung
•Transaktionskontrolle

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

30

Data Control Language (DCL) Transaction Control Language (TCL)



University of Applied Sciences

DCL

- **GRANT** zur Erteilung von Zugriffsrechten
- **REVOKE** zur Entziehung von Zugriffsrechten

TCL

- **COMMIT** zum abschließen einer Transaktion
- **ROLLBACK** zum wiederherstellen eines validen Zustands der Datenbank vor Transaktionsstart
- Die TCL dient der Integrität von Datenbanken in Mehrbenutzerszenarien mit konkurrierenden Zugriffen.

Hinweis: DCL und TCL werden in dieser Veranstaltung nicht weiter angesprochen und bleiben dem Selbststudium im Rahmen der Projektarbeit überlassen.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

31

Kategorien von SQL Befehlen



University of Applied Sciences

DDL

Data Definition Language

Befehle zur
Definition
Datenbank-
schema

DQL

Data Quer-
zungs-
language

SELECT

DML

Data Mani-
pulation
language

Befehle zur
Daten-
manipula-
tion
• Ändern
• Einfügen
• Löschen

DCL

Data Control
Language

Befehle für die
• Rechteverwaltung
• Transaktionskontrolle

Sichten

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

32

Was ist der Unterschied zwischen einer Relation und einer Sicht?



University of Applied Sciences

Relation

- Speicherstruktur in Datenbank
- Speichert physisch Daten

Sicht

- Virtuelle Relation
- In einer Sicht sind keine Daten gespeichert
- eine Art „gespeicherter“ SELECT Befehl
- Zusammenstellung der Daten für spez. Zwecke
- Schutz von Daten (z.B. Ausblenden von Gehaltszahlen)

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

33

Beispiel für Sichten mittels CREATE VIEW



University of Applied Sciences

liest

<u>Titel</u>	<u>MatrNr</u>
Die Pest	12345
Der Himmel	12345
Die Pest	43215
Die Pest	53421

Studenten

<u>MatrNr</u>	<u>Name</u>
12345	A. Meier
43215	B. Müller
53421	C. Kunze
111335	D. Huber

<u>Name</u>	<u>Titel</u>
A. Meier	Die Pest
A. Meier	Der Himmel
B. Müller	Die Pest
C. Kunze	Die Pest

```
CREATE VIEW Studenten_leSEN_Titel
    (Name, Titel)
SELECT S.Name, l.Titel
FROM    liest AS l, Studenten AS S
WHERE   S.MatrNr = l.MatrNr;
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

34

Zusammenfassung



University of Applied Sciences

- Data Definition Language (DDL)
- Data Query Language (DQL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)
- Views

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

35