

DATENBANKEN SS 2022

SQL – DATA MANIPULATION

Prof. Dr. Eva-Maria Iwer
Juni 2022

CREATE TABLE

```
CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    (create_definition,...) [table_options    ]... [partition_options]
CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    [(create_definition,...)] [table_options    ]... [partition_options]
    select_statement
CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    { LIKE old_table_name | (LIKE old_table_name) }

select_statement:
    [IGNORE | REPLACE] [AS] SELECT ...    (Some legal select statement)
```

```
create_definition:
  { col_name column_definition | index_definition | period_definition | CHECK (expr) }

column_definition:
  data_type
  [NOT NULL | NULL] [DEFAULT default_value | (expression)]
  [ON UPDATE [NOW | CURRENT_TIMESTAMP] [(precision)]]
  [AUTO_INCREMENT] [ZEROFILL] [UNIQUE [KEY] | [PRIMARY] KEY]
  [INVISIBLE] [{WITH|WITHOUT} SYSTEM VERSIONING]
  [COMMENT 'string'] [REF_SYSTEM_ID = value]
  [reference_definition]
  | data_type [GENERATED ALWAYS]
  AS { { ROW {START|END} } | { (expression) [VIRTUAL | PERSISTENT | STORED] } }
  [UNIQUE [KEY]] [COMMENT 'string']

constraint_definition:
  CONSTRAINT [constraint_name] CHECK (expression)
```

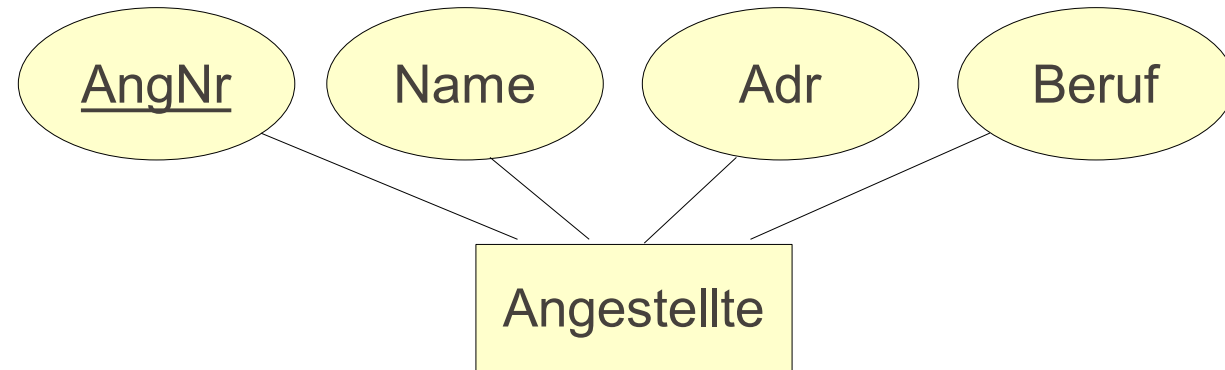
Datatype – wichtigste Datentypen in SQL

INTEGER	Ganzzahl
INT	Kurzform von INTEGER
SMALLINT	Ganzzahl
NUMERIC(x,y)	x stellige Zahl mit y Nachkommastellen
DECIMAL(x,y)	x stellige Zahl mit y Nachkommastellen
FLOAT(x)	Gleitpunktzahl mit x Nachkommastellen
CHARACTER(n)	Zeichenkette der festen Länge n
CHAR(n)	Kurzform von CHARACTER (n)
CHARACTER VARYING(n)	Variable Zeichenkette mit bis zu n Zeichen
VARCHAR(n)	Kurzform von CHARACTER VARYING (n)
BIT(n)	Bitleiste der festen Länge n
DATE	Datum (Jahr, Monat, Tag)
TIME	Uhrzeit (Stunde, Minute, Sekunde)
DATETIME	Kombination aus DATE und TIME

DATENBANK-DEFINITION MIT SQL INTEGRITÄTSBEDINGUNGEN

Beispiel

```
CREATE TABLE
  angestellte
  (angnr INT(5) NOT NULL,
   name VARCHAR(50),
   adr VARCHAR(50),
   beruf VARCHAR(20),
   PRIMARY KEY (angnr)
  );
```



DATENBANK-DEFINITION MIT SQL INTEGRITÄTSBEDINGUNGEN

Fremdschlüssel

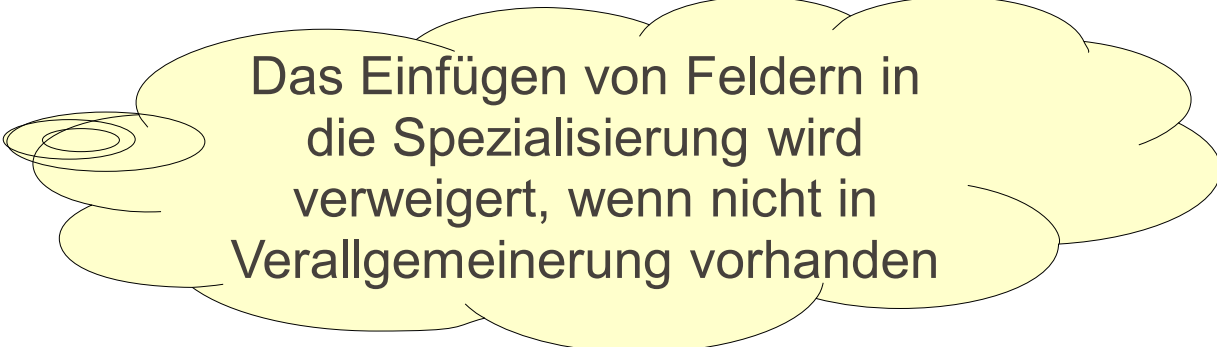
- Ist ein Attribut (Attributmenge) in einer Relation gleichzeitig Schlüssel einer anderen Relation, ist das Attribut (Attributmenge) ein Fremdschlüssel. Es ist ein Schlüssel einer fremden Relation.

```
CREATE TABLE tbl_name ( ...  
  [CONSTRAINT [cstr_name]]  
  FOREIGN KEY (list_of_column_names)  
  REFERENCES table_name (list_of_column_names)  
  [ON DELETE  
    { NO ACTION | CASCADE | SET NULL }]  
  [ON UPDATE  
    { NO ACTION | CASCADE | SET NULL }],  
  ... )
```

DATENBANK-DEFINITION MIT SQL INTEGRITÄTSBEDINGUNGEN

Fremdschlüssel

```
CREATE TABLE tbl_name ( ...  
  [CONSTRAINT [cstr_name]]  
  FOREIGN KEY (list_of_column_names)  
  REFERENCES table_name (list_of_column_names)  
  [ON DELETE  
    { NO ACTION | CASCADE | SET NULL }]  
  [ON UPDATE  
    { NO ACTION | CASCADE | SET NULL }],  
  ... )
```



Das Einfügen von Feldern in
die Spezialisierung wird
verweigert, wenn nicht in
Verallgemeinerung vorhanden

DATENBANK-DEFINITION MIT SQL INTEGRITÄTSBEDINGUNGEN

Fremdschlüssel

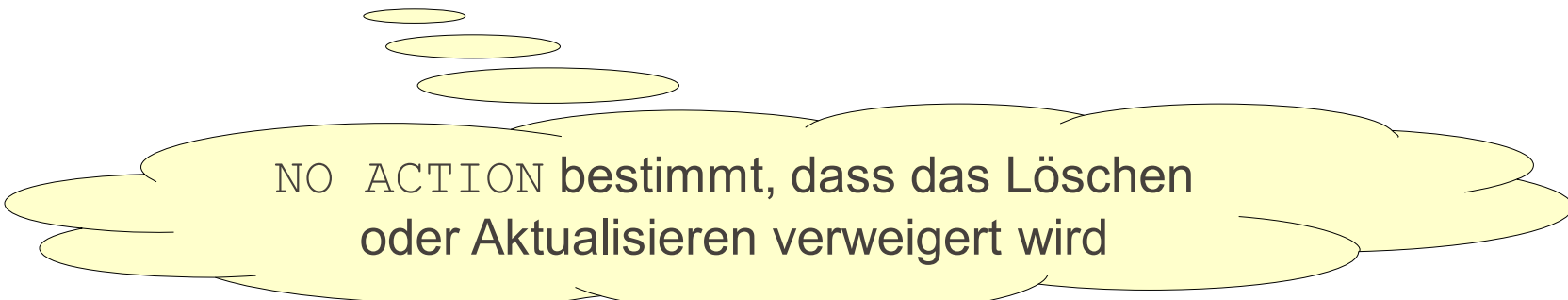
```
CREATE TABLE tbl_name ( ...  
  [CONSTRAINT [cstr_name]]  
  FOREIGN KEY (list_of_column_names)  
  REFERENCES table_name (list_of_column_names)  
  [ON DELETE  
    { NO ACTION | CASCADE | SET NULL }]  
  [ON UPDATE  
    { NO ACTION | CASCADE | SET NULL }],  
  ... )
```

Die Aktion, die ausgeführt wird, wenn ein Feld
in der Verallgemeinerung gelöscht oder
aktualisiert wird, kann mit ON DELETE bzw.
ON UPDATE bestimmt werden

DATENBANK-DEFINITION MIT SQL INTEGRITÄTSBEDINGUNGEN

Fremdschlüssel

```
CREATE TABLE tbl_name ( ...  
  [CONSTRAINT [cstr_name]]  
  FOREIGN KEY (list_of_column_names)  
  REFERENCES table_name (list_of_column_names)  
  [ON DELETE  
    { NO ACTION | CASCADE | SET NULL }]  
  [ON UPDATE  
    { NO ACTION | CASCADE | SET NULL }],  
  ... )
```

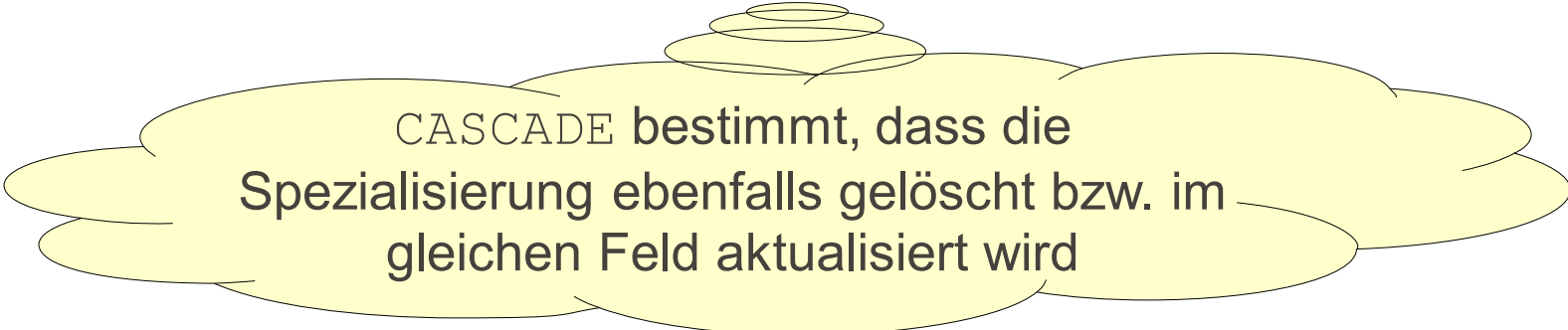


NO ACTION bestimmt, dass das Löschen
oder Aktualisieren verweigert wird

DATENBANK-DEFINITION MIT SQL INTEGRITÄTSBEDINGUNGEN

Fremdschlüssel

```
CREATE TABLE tbl_name ( ...  
  [CONSTRAINT [cstr_name]]  
  FOREIGN KEY (list_of_column_names)  
  REFERENCES table_name (list_of_column_names)  
  [ON DELETE  
    { NO ACTION | CASCADE | SET NULL }]  
  [ON UPDATE  
    { NO ACTION | CASCADE | SET NULL }],  
  ... )
```



CASCADE bestimmt, dass die
Spezialisierung ebenfalls gelöscht bzw. im
gleichen Feld aktualisiert wird

DATENBANK-DEFINITION MIT SQL INTEGRITÄTSBEDINGUNGEN

Fremdschlüssel

```
CREATE TABLE tbl_name ( ...  
  [CONSTRAINT [cstr_name]]  
  FOREIGN KEY (list_of_column_names)  
  REFERENCES table_name (list_of_column_names)  
  [ON DELETE  
    { NO ACTION | CASCADE | SET NULL }]  
  [ON UPDATE  
    { NO ACTION | CASCADE | SET NULL }],  
  ... )
```

SET NULL bestimmt, dass das Feld in der
Spezialisierung auf NULL gesetzt wird (Feld
darf nicht auf NOT NULL gesetzt sein)

DATENBANK-DEFINITION MIT SQL INTEGRITÄTSBEDINGUNGEN

Beispiel

```
CREATE TABLE
```

```
  pilot
```

```
  (angnr INT(5) NOT NULL,
```

```
   std INT(5),
```

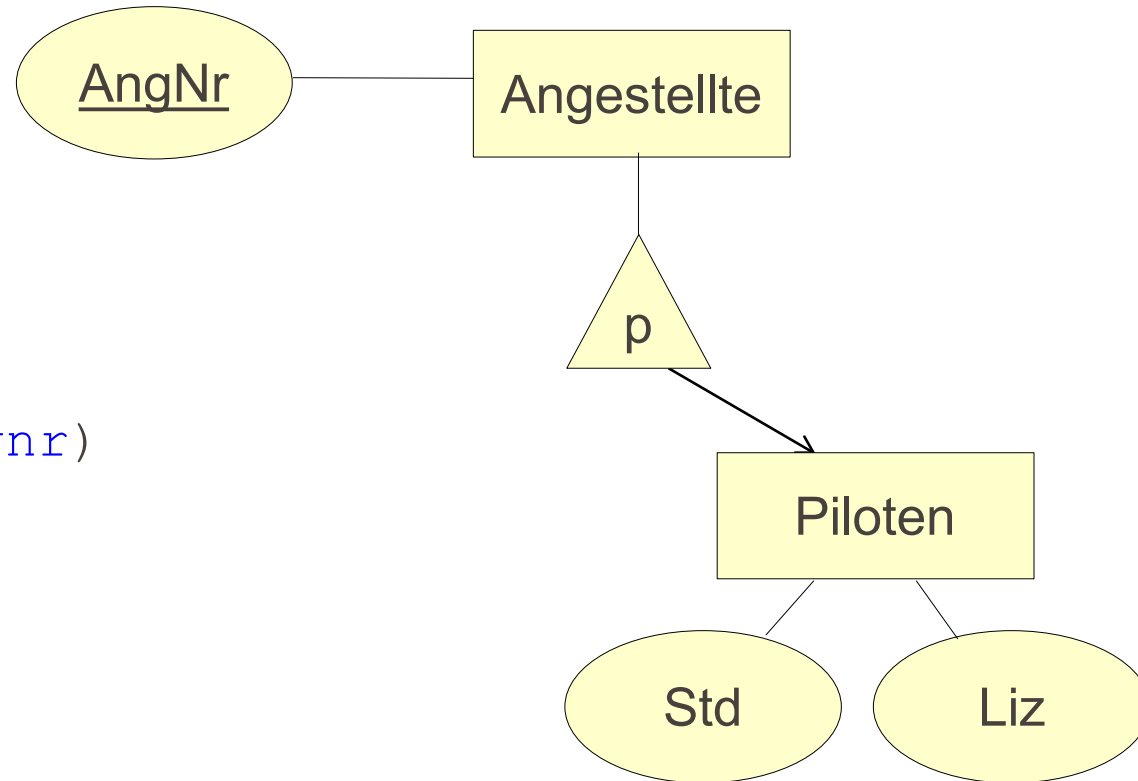
```
   liz VARCHAR(50),
```

```
   PRIMARY KEY (angnr),
```

```
   FOREIGN KEY (angnr)
```

```
     REFERENCES angestellte(angnr)
```

```
);
```



DATENBANK-DEFINITION MIT SQL

ZUGRIFFSRECHTE

Zugriffsrechte für Tabellen

```
GRANT { SELECT | INSERT | DELETE |  
      UPDATE | ALL }  
ON table_name  
TO user_name [, ...]  
[ WITH GRANT OPTION ]
```

Mit `GRANT OPTION` können eigene Privilegien an Dritte weiter gegeben werden

Erzeuger einer Tabelle besitzt automatisch alle Privilegien mit Weitergaberecht

DATENBANK-DEFINITION MIT SQL SCHEMAMODIFIKATIONEN

Nachträgliche Änderungen zur Laufzeit

Bestehende Definition modifizieren oder löschen

ALTER-Befehl mit Kombination von ADD, CHANGE oder DROP

DATENBANK-DEFINITION MIT SQL SCHEMAMODIFIKATIONEN

```
ALTER TABLE table_name
{ ADD [COLUMN] col_name definition
| CHANGE [COLUMN] old_col_name
    new_col_name definition
| DROP [COLUMN] col_name
| ADD [CONSTRAINT [cstr_name]]
    { PRIMARY KEY (list_of_column_names)
    | UNIQUE (list_of_column_names)
    | FOREIGN KEY (list_of_column_names)
      REFERENCES ...
    }
| DROP PRIMARY KEY
| DROP FOREIGN KEY cstr_name
}, ...
```

ALTER

ALTER BEFEHL

Ausschnitt

```
ALTER [ONLINE] [IGNORE] TABLE [IF EXISTS] tbl_name
  [WAIT n | NOWAIT]
  alter_specification [, alter_specification] ...

alter_specification:
  table_option ...
| ADD [COLUMN] [IF NOT EXISTS] col_name column_definition
  [FIRST | AFTER col_name ]
| ADD [COLUMN] [IF NOT EXISTS] (col_name column_definition,...)
| ADD {INDEX|KEY} [IF NOT EXISTS] [index_name]
  [index_type] (index_col_name,...) [index_option] ...
| ADD [CONSTRAINT [symbol]] PRIMARY KEY
  [index_type] (index_col_name,...) [index_option] ...
| ADD [CONSTRAINT [symbol]]
  UNIQUE [INDEX|KEY] [index_name]
  [index_type] (index_col_name,...) [index_option] ...
| ADD FULLTEXT [INDEX|KEY] [index_name]
  (index_col_name,...) [index_option] ...
| ADD SPATIAL [INDEX|KEY] [index_name]
  (index_col_name,...) [index_option] ...
| ADD [CONSTRAINT [symbol]]
  FOREIGN KEY [IF NOT EXISTS] [index_name] (index_col_name,...)
  reference_definition
| ADD PERIOD FOR SYSTEM_TIME (start_column_name, end_column_name)
| ALTER [COLUMN] col_name SET DEFAULT literal | (expression)
| ALTER [COLUMN] col_name DROP DEFAULT
| ALTER {INDEX|KEY} index_name [NOT] INVISIBLE
| CHANGE [COLUMN] [IF EXISTS] old_col_name new_col_name column_definition
  [FIRST|AFTER col_name]
```

DATENBANK-DEFINITION MIT SQL SCHEMAMODIFIKATIONEN

Beispiel:

```
ALTER TABLE angestellte  
  CHANGE angnr  
    angnr INT(5) NOT NULL AUTO_INCREMENT
```

DROP

DROP BEFEHL

```
DROP [TEMPORARY] TABLE [IF EXISTS] [/*COMMENT TO SAVE*/]  
tbl_name [, tbl_name] ...  
[WAIT n|NOWAIT]  
[RESTRICT | CASCADE]
```

DATENBANK-DEFINITION MIT SQL SCHEMAMODIFIKATIONEN

Eine Tabelle löschen

```
DROP TABLE tbl_name [, tbl_name]
```

Beispiel:

```
DROP TABLE angestellte
```

INSERT

ERINNERUNG: BIER BEISPIEL

- Unterstrichen sind die Schlüsselattribute:
Beers(name, manf)
Bars(name, addr, license)
Drinkers(name, addr, phone)
Likes(drinker, beer)
Sells(bar, beer, price)
Frequents(drinker, bar)

INSERT BEFEHL

```
INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
[INTO] tbl_name [PARTITION (partition_list)] [(col,...)]
{VALUES | VALUE} ({expr | DEFAULT},...),(...),...
[ ON DUPLICATE KEY UPDATE
  col=expr
  [, col=expr] ... ] [RETURNING select_expr
  [, select_expr ...]]
```


INHALTE HINZUFÜGEN

- Um ein einzelnes Tupel hinzuzufügen:

```
INSERT INTO <relation>  
VALUES ( <list of values> );
```

- **Beispiel:** hinzufügen zu **Likes(drinker, beer)**, dass Sally Bud mag.

```
INSERT INTO Likes  
VALUES ('Sally', 'Bud');
```

SPEZIFIZIERUNGSATTRIBUTE IN INSERT

- Sie können die Attributnamen der Relation hinzufügen
- Dieses hätte den Vorteil:
 - Keine Verwechslung bei der Standardsortierung
 - Angabe von NULL oder default-Werten

```
INSERT INTO Likes (beer, drinker)
VALUES ('Bud', 'Sally');
```

HINZUFÜGEN VON MEHREREN TUPELN

```
INSERT INTO <relation>
    ( <subquery> );
INSERT INTO PotBuddies
(SELECT d2.drinker
 FROM Frequent d1, Frequent d2
 WHERE d1.drinker = 'Sally' AND
       d2.drinker <> 'Sally' AND
       d1.bar = d2.bar
);
```

DELETE

DELETE

BEFEHL

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
FROM tbl_name [PARTITION (partition_list)]
[WHERE where_condition]
[ORDER BY ...]
[LIMIT row_count]
[RETURNING select_expr
 [, select_expr ...]]
```

TUPELENTFERNUNG BZW. DELETION

```
DELETE FROM <relation>  
    WHERE <condition>;
```

```
DELETE FROM Likes  
    WHERE drinker = 'Sally' AND  
        beer = 'Bud';
```

ALLE TUPEL ENTFERNEN

- `DELETE FROM Likes;`

EINIGE TUPEL ENTFERNEN

```
DELETE FROM Beers b
WHERE EXISTS (
  SELECT name FROM Beers
  WHERE manf = b.manf AND
        name <> b.name);
```


UPDATE

DELETE BEFEHL

```
UPDATE [LOW_PRIORITY] [IGNORE] table_reference  
  [PARTITION (partition_list)]  
  SET col1={expr1|DEFAULT} [,col2={expr2|DEFAULT}] ...  
  [WHERE where_condition]  
  [ORDER BY ...]  
  [LIMIT row_count]
```

UPDATE VON TUPELN

```
UPDATE <relation>  
    SET <list of attribute assignments>  
    WHERE <condition on tuples>;
```

```
UPDATE Drinkers  
    SET phone = '555-1212'  
    WHERE name = 'Fred';
```

UPDATE VON MEHREREN TUPELN

```
UPDATE Sells  
    SET price = 4.00  
    WHERE price > 4.00;
```

VIEWS

VIEWS

Spezielle Benutzersicht

„virtuelle“ Tabelle

Werden erst zur Laufzeit ausgewertet

In einem View können auch mit `UPDATE` Daten aktualisiert werden

```
CREATE VIEW view_name
    [ (column_list) ]
AS select_statement
    [WITH CHECK OPTION]
```

VIEWS

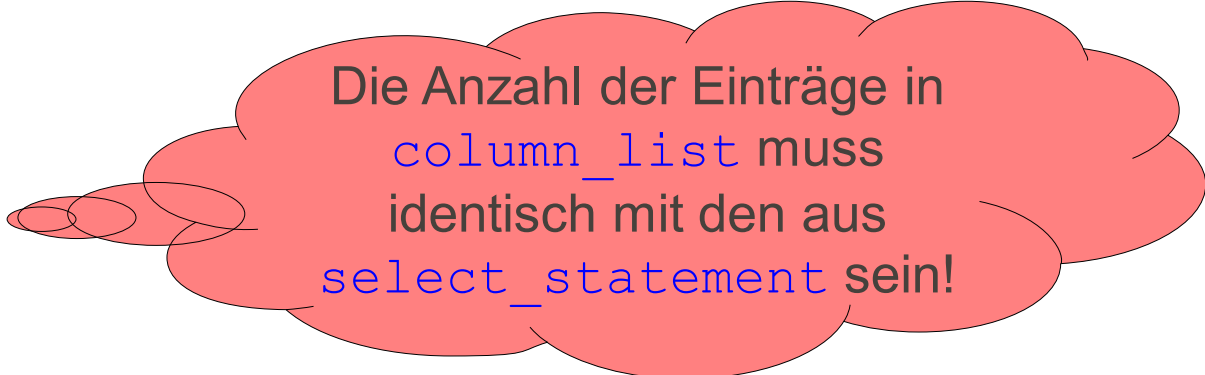
Spezielle Benutzersicht

„virtuelle“ Tabelle

Werden erst zur Laufzeit ausgewertet

In einem View können auch mit `UPDATE` Daten aktualisiert werden

```
CREATE VIEW view_name  
    [(column_list)]  
AS select_statement  
[WITH CHECK OPTION]
```



Die Anzahl der Einträge in
`column_list` muss
identisch mit den aus
`select_statement` sein!

VIEWS

Spezielle Benutzersicht

„virtuelle“ Tabelle

Werden erst zur Laufzeit ausgewertet

In einem View können auch mit UPDATE Daten aktualisiert werden

```
CREATE VIEW view_name  
    [(column_list)]  
AS select_statement  
[WITH CHECK OPTION]
```



WITH CHECK OPTION lässt
das UPDATE nur zu, wenn die
WHERE Bedingung auf
select_statement passt!

VIEW

Beispiel:

```
CREATE VIEW v_pilot AS
  SELECT a.*, p.std, p.liz
  FROM angestellte AS a
  RIGHT OUTER JOIN pilot AS p
  ON a.angnr = p.angnr
```

angnr	name	adr	beruf	std	liz
1	Hans Fliege	Wiesbaden	Pilot	150	Airbus 380

LITERATUR

Pröll, S., Zangerle, E. Und Gassler, W.: MySQL – Das umfassende Handbuch, 2. Auflage, Galileo Press, 2013

Vossen, Gottfried: Datenmodelle, Datenbank-sprachen und Datenbankmanagementsysteme, 5. Auflage, Oldenburg Wissenschaftsverlag, 2008

Schwinn, Hans: Relationale Datenbanksysteme, Hanser, 1992

Sun Microsystems: MySQL 5.1 Reference Manual,
<http://dev.mysql.com/doc/refman/5.1/en/>