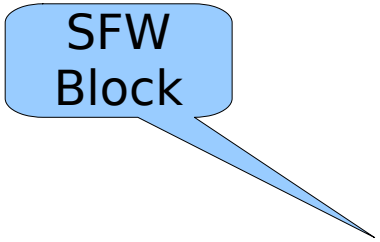
Tag 3
Inhaltsverzeichnis



SQL Select-Anweisung (Grundbild)

```
SELECT [DISTINCT] Auswahlliste [AS Neuer Name]+
FROM Quelle
[WHERE Where-Klausel]
[GROUP BY (Group-by-Attribut)+
 [HAVING Having-Klausel]]
[ORDER BY (Sortierungsattribut [ASC|DESC])+];
```

- **S:**
  - Projektionsliste (--> *Spalten* selektieren) ("*" für alle)
  - Arithmetische Operationen, Aggregatfunktionen
  - Resultat: eine neue Relation
- **F:**
  - Zu verwendende Relationen
- **W:**
  - Bedingung(en) (--> *Zeilen* selektieren)
  - Schachtelung möglich

Daten filtern
Gleichheitsbedingung (1)

```
mysql> select * from autor where PersNr='12';
+--------+---------+--------+
| persnr | vorname | name   |
+--------+---------+--------+
|     12 | Alfons  | Kemper |
+--------+---------+--------+
```

Die SQL-Schlüsselworte schreibe ich klein

```
mysql> select * from autor a where a.PersNr='12';
+--------+---------+--------+
| persnr | vorname | name   |
+--------+---------+--------+
|     12 | Alfons  | Kemper |
+--------+---------+--------+
```

Verwendung einer *Tabellen-Variable*

```
mysql> select persnr as keypers from autor a where a.persnr='12';
+---------+
| keypers |
+---------+
|      12 |
+---------+
```

Spalten-Alias

Daten filtern
Gleichheitsbedingung (2)

```
mysql> select distinct vorname from autor;
+---------+
| vorname |
+---------+
| Alfons  |
| Michael |
| Gilles  |
+---------+
```

Entferne Duplicate

```
mysql> select vorname, name from autor
       order by vorname, name asc;
+---------+--------+
| vorname | name   |
+---------+--------+
| Alfons  | Kemper |
| Gilles  | Maitre |
| Michael | Kofler |
+---------+--------+
```

Sortierung

Daten filtern
Wertebereichsbedingung

```
mysql> select * from autor
       where persnr >= 11 and persnr <= 35;
+--------+---------+--------+
| persnr | vorname | name   |
+--------+---------+--------+
|     12 | Alfons  | Kemper |
|     34 | Michael | Kofler |
+--------+---------+--------+
```

```
mysql> select * from autor where persnr
       between 11 and 35;
+--------+---------+--------+
| persnr | vorname | name   |
+--------+---------+--------+
|     12 | Alfons  | Kemper |
|     34 | Michael | Kofler |
+--------+---------+--------+
```

Daten filtern
Mitgliedschaftbedingung

```
mysql> select * from autor where persnr in ('11', '35');
Empty set (0.00 sec)


mysql> select * from autor where persnr in ('12', '34');
+--------+---------+--------+
| persnr | vorname | name   |
+--------+---------+--------+
|     12 | Alfons  | Kemper |
|     34 | Michael | Kofler |
+--------+---------+--------+
```

Daten filtern
Wildcards

"_": genau ein Zeichen
"%": beliebig viele Zeichen

```
mysql> select * from autor where name = "K%";
Empty set (0.00 sec)
```

```
mysql> select * from autor where name like "K%";
+--------+---------+--------+
| persnr | vorname | name   |
+--------+---------+--------+
|     12 | Alfons  | Kemper |
|     34 | Michael | Kofler |
+--------+---------+--------+
```

```
mysql> select * from autor where vorname regexp "^[AM]";
+--------+---------+--------+
| persnr | vorname | name   |
+--------+---------+--------+
|     12 | Alfons  | Kemper |
|     34 | Michael | Kofler |
+--------+---------+--------+
```

Daten filtern
Null ist nicht null

```
mysql> alter table autor add column bemerkung varchar(20) after name;

mysql> select * from autor where bemerkung = null;
Empty set (0.00 sec)


mysql> select * from autor where bemerkung is null;
+--------+---------+--------+-----------+
| persnr | vorname | name   | bemerkung |
+--------+---------+--------+-----------+
|     12 | Alfons  | Kemper | NULL      |
|     34 | Michael | Kofler | NULL      |
|     56 | Gilles  | Maitre | NULL      |
+--------+---------+--------+-----------+


mysql> alter table autor drop column bemerkung;
```

Komplexe Abfragen strukturieren
Common Table Expression (CTE)

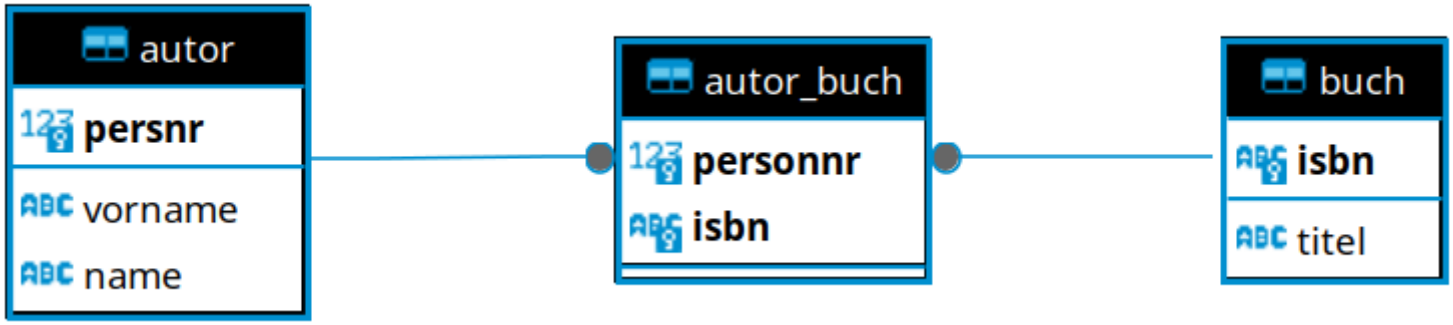
```
WITH query_name1 AS (
      SELECT ...
      )
    , query_name2 AS (
      SELECT ...
        FROM query_name1
          ...
      )
SELECT ...

-- Beispiel aus http://www.mysqltutorial.org/mysql-cte/

WITH customers_in_usa AS (
     SELECT customerName, state FROM customers WHERE country =
'USA'
)
SELECT customerName FROM customers_in_usa WHERE state = 'CA';
```


Quelle: https://modern-sql.com/de/feature/with

Mehrere Tabellen abfragen (Joins)
Erster Versuch

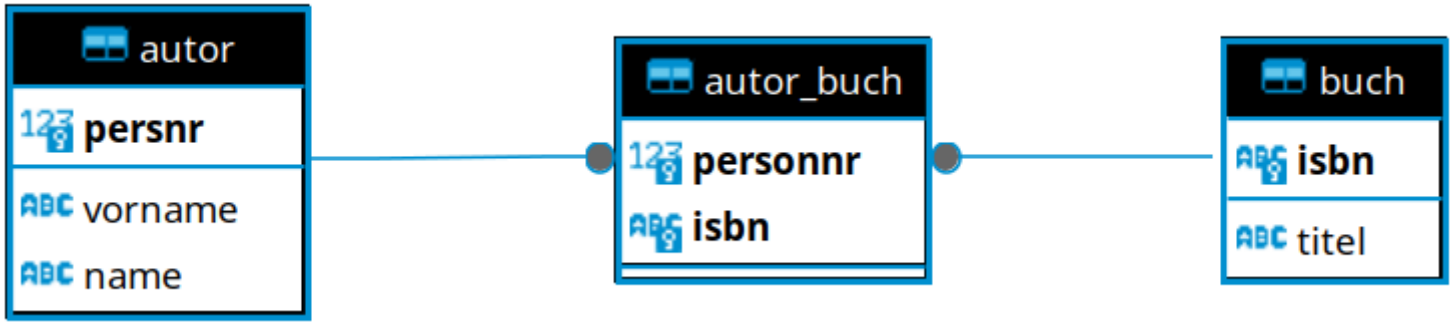



```
mysql> select autor.name, buch.titel from autor join buch ;
+--------+--------------------+
| name   | titel              |
+--------+--------------------+
| Kemper | VisualBasic 2008   |
| Kofler | VisualBasic 2008   |
| Maitre | VisualBasic 2008   |
| Wall   | VisualBasic 2008   |
| Kemper | Datenbanksysteme   |
| Kofler | Datenbanksysteme   |
| Maitre | Datenbanksysteme   |
| Wall   | Datenbanksysteme   |
...
```

Auf die Frage: welche Autoren haben welche Bücher geschrieben?

*Das kartesisches Produkt gibt die Antwort **nicht**!*

Mehrere Tabellen abfragen (Joins)
Lösung "von Hand"

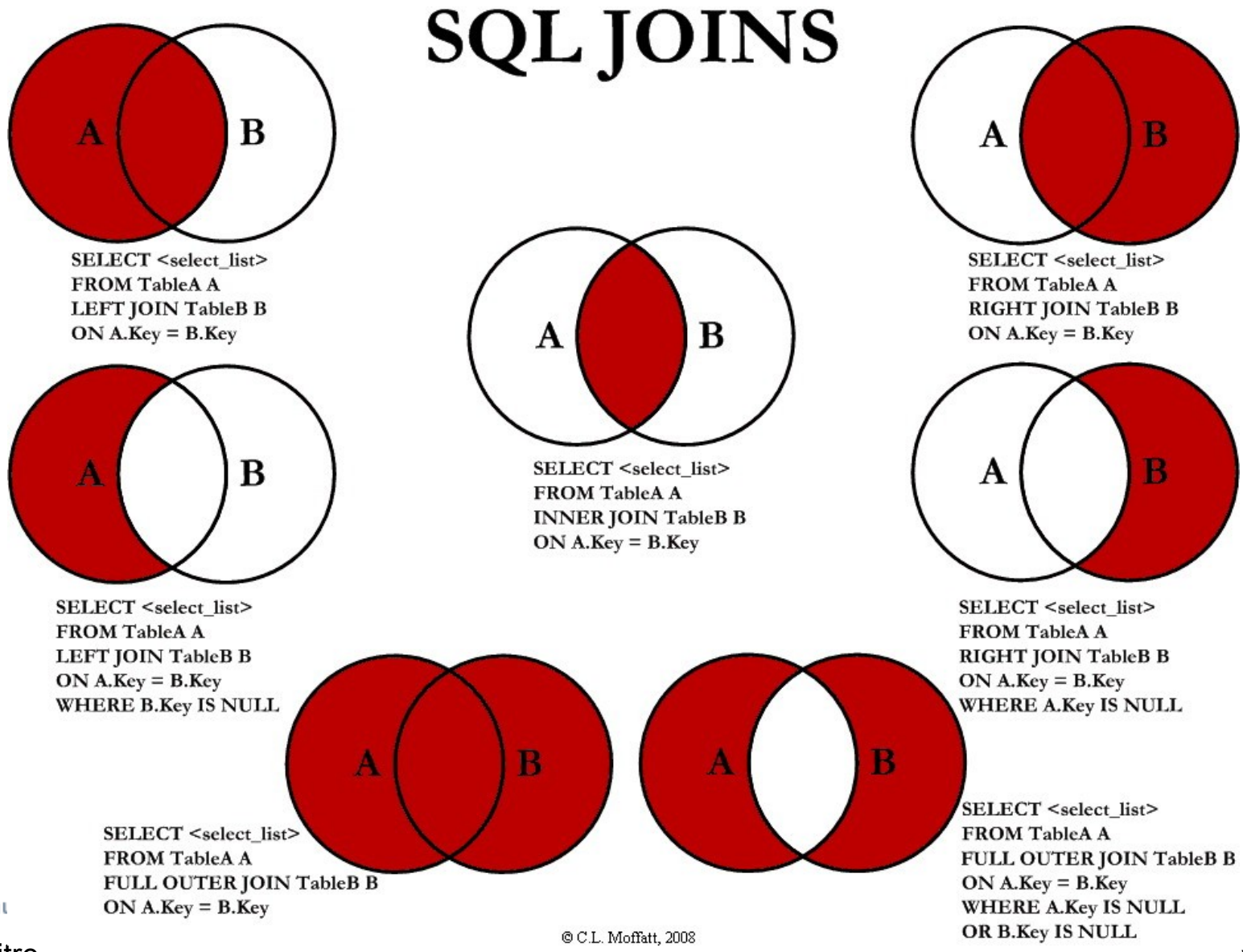



```
mysql> select autor.name, buch.titel
       from autor, autor_buch, buch
       where autor.persnr = autor_buch.personnr
       and buch.isbn = autor_buch.isbn;
+--------+------------------+
| name   | titel            |
+--------+------------------+
| Kemper | Datenbanksysteme |
| Kofler | VisualBasic 2008 |
| Kofler | Mathematica      |
| Kofler | MySQL 5          |
| Kofler | Linux            |
+--------+------------------+
```

Frage: welche Autoren haben welche Bücher geschrieben?

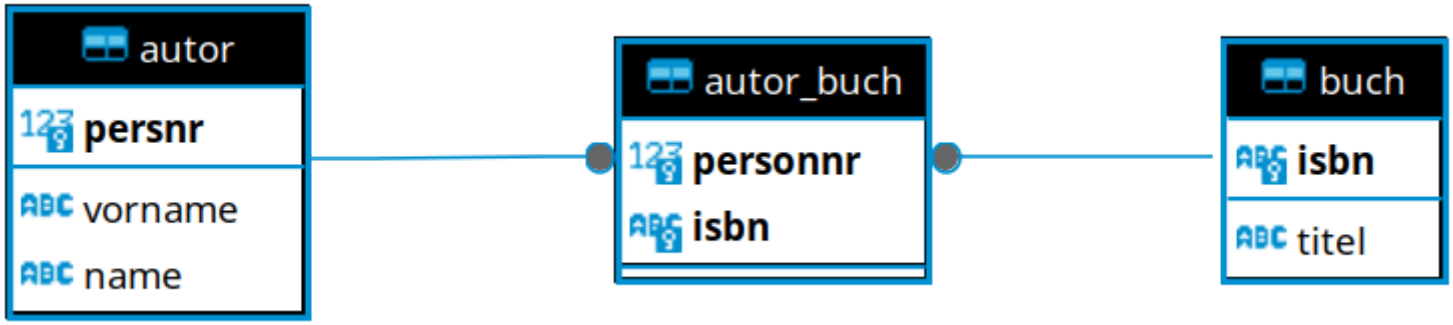
Kartesisches Produkt ***PLUS*** *Selektion*

Mehrere Tabellen abfragen (Joins)
Join-Typen



Quelle: http://www.codeproject.com/Articles/33052/Visual-Representation-of-SQL-Joins

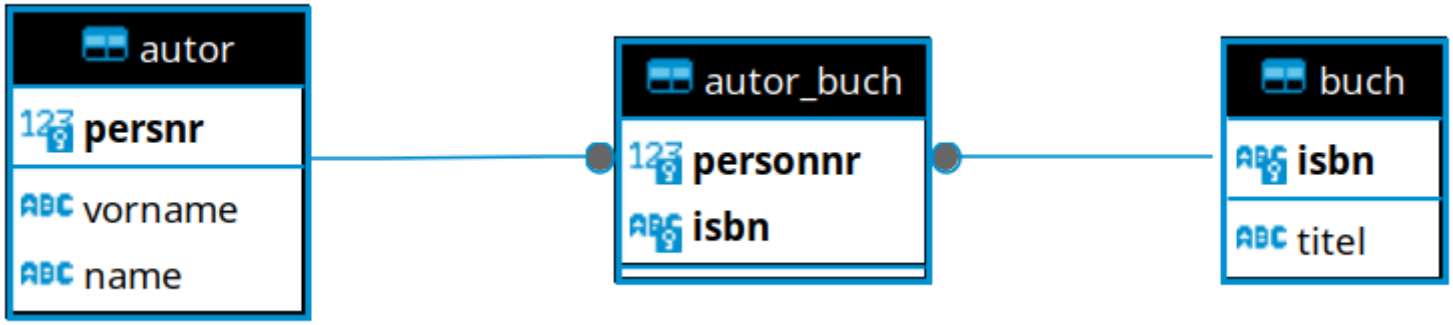
Mehrere Tabellen abfragen (Joins)
Lösung "mit INNER JOIN" (1)




```
mysql> select a.name, b.titel from autor a
       inner join autor_buch ab on a.persnr=ab.personnr
       inner join buch b on b.isbn=ab.isbn;
+--------+-------------------+
| name   | titel             |
+--------+-------------------+
| Kemper | Datenbanksysteme  |
| Kofler | VisualBasic 2008  |
| Kofler | Mathematica       |
| Kofler | MySQL 5           |
| Kofler | Linux             |
+--------+-------------------+
```

"INNER JOIN" ist der Default-JOIN

Mehrere Tabellen abfragen (Joins)
Left Join




```
mysql> select a.name, b.titel from autor a
       left join autor_buch ab on a.persnr = ab.personnr
       left join buch b using (ISBN);
+--------+------------------+
| name   | titel            |
+--------+------------------+
| Kemper | Datenbanksysteme |
| Kofler | VisualBasic 2008 |
| Kofler | Mathematica      |
| Kofler | MySQL 5          |
| Kofler | Linux            |
| Maitre | NULL             |
| Wall   | NULL             |
+--------+------------------+
```

Nur möglich wenn beide Tabellen 'isbn' als Attribut haben

Arbeiten mit Strings
Datentypen

- **CHAR(n)**
  - Festlänge, MySQL bis 255 Zeichen, Oracle 2000
- **varchar(n)**
  - Variable Länge, MySQL bis 65K Zeichen, Oracle 4K
- **text(n)**
  - Für Dokumente, MySQL bis 4 GB, Oracle 128 TB
  - (CLOB in Oracle)

Für alle: (n) == maximale Länge
insert(Data > n) => ERROR 1406 (22001): Data too long for column '...'

Arbeiten mit Datum/Zeit
Datentypen

- Date (YYYY-MM-DD)
  (Bereich: '1000-01-01' bis '9999-12-31')
  3 Bytes
- Datetime (YYYY-MM-DD HH:MI:SS)
  (Bereich: '1000-01-01 00:00:00' bis
  '9999-12-31 23:59:59')
  8 Bytes
- Timestamp (YYYY-MM-DD HH:MI:SS)
  (Bereich: '1970-01-01 00:00:01' UTC bis '2038-01-09 03:14:07' UTC)
  4 Bytes, #Sek. seit 1970-01-01 00:00:00
- Time (HH:MI:SS)
  3 Bytes
- Mögliche Eingabe-Formate:
  - 2009/07/17 14:29:00
  - 2009,07,17,14,29,00
  - 20090717142900

Arbeiten mit Datum/Zeit
Beispiele

```
mysql> select str_to_date('17 July 09', '%d %M %y');
+---------------------------------------+
| str_to_date('17 July 09', '%d %M %y') |
+---------------------------------------+
| 2009-07-17                            |
+---------------------------------------+

mysql> select current_date(), current_time(), current_timestamp();
+----------------+----------------+---------------------+
| current_date() | current_time() | current_timestamp() |
+----------------+----------------+---------------------+
| 2009-07-17     | 14:34:38       | 2009-07-17 14:34:38 |
+----------------+----------------+---------------------+

mysql> select now();
+---------------------+
| now()               |
+---------------------+
| 2009-07-17 14:35:19 |
+---------------------+

mysql> select date_add(now(), interval 3 day);
+-----------------------------------+
| date_add(now(), interval 3 day)   |
+-----------------------------------+
| 2009-07-20 14:38:20               |
+-----------------------------------+
```

MySQL
build-in Funktionen

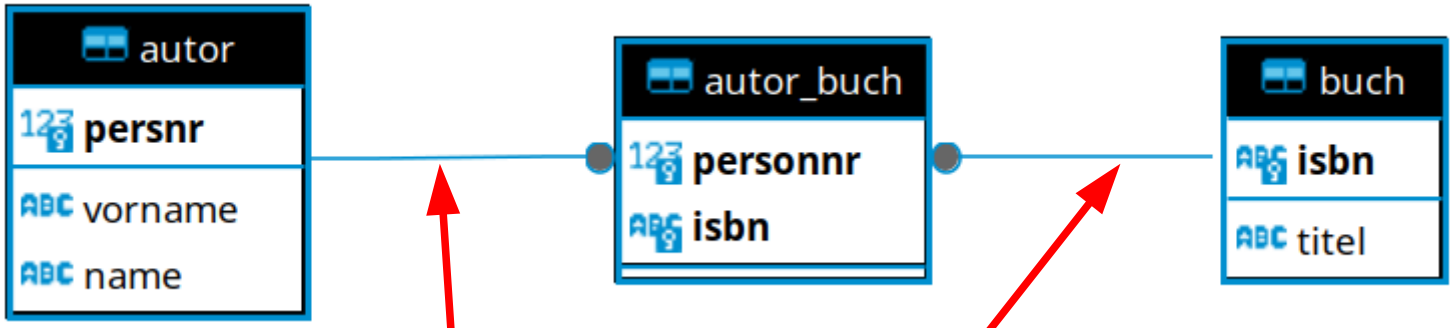
Chapter 11. Functions and Operators

Table of Contents [+/-]

Referenzielle Integrität
Problematik der Daten-Integrität

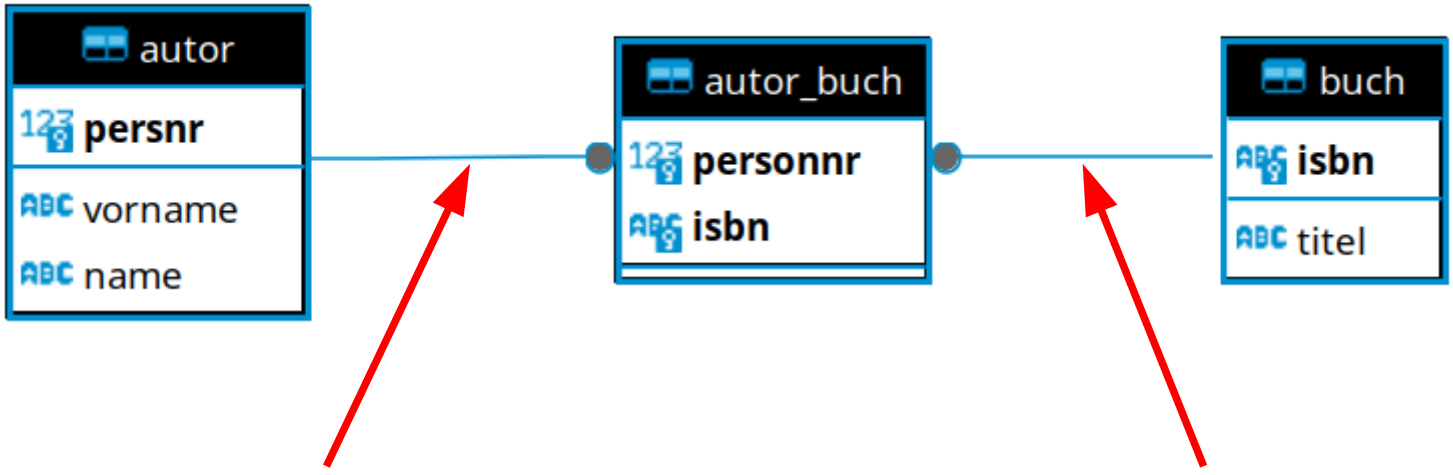
- **Integrität der Primärschlüssel**
  - Unique, not null
- **Integrität der Fremdschlüssel**
  - Referenzierte Primärschlüssel existieren
- **Integrität der anderen Daten (nach Typ)**
  - Wertbereich
  - Nicht NULL

Referenzielle Integrität
Konsistenz der Fremdschlüssel



- Kohärenz der Fremdschlüssel
  - "Dangling Pointer" Problem vermeiden
- => **Integritätsregel** setzen

Referenzielle Integrität
FK Namenskonvention



Meine FK Namenskonvention:
autor_buch_fk_personnr

Meine FK Namenskonvention:
autor_buch_fk_isbn

Meine FK Namenskonvention:
'fromTable'_FK_'fromField'

Referenzielle Integrität
Regel erstellen

- **In MySQL, nur für InnoDB-Tabellen**

Namenskonvention:
'fromTable'_FK_'fromField'

- **Beispiel:**

```
ALTER TABLE autor_buch
    ADD CONSTRAINT autor_buch_fk_personnr
        FOREIGN KEY (personnr) REFERENCES autor (`persnr`);
```

=> In "autor" wird kein Löschen von einem referenzierten Autor möglich.

- **Syntax:**

```
FOREIGN KEY [Name] (Kolonne 1) REFERENCES Tabelle2 (Kolonne
2)
  [ON DELETE {CASCADE | SET NULL | NO ACTION | RESTRICT}]
  [ON UDPATE {CASCADE | SET NULL | NO ACTION | RESTRICT}]
```

- **"RESTRICT" ist der Default**

Referenzielle Integrität
Regel anzeigen

- **In DBeaver**
  - "Table Information" anzeigen / Tab "Foreign Keys"

- **In MySQL Command Line Interpreter**
  - show create table <table name>

```
CREATE TABLE autor_buch (
  personnr int unsigned NOT NULL,
  isbn varchar(10) NOT NULL,
  PRIMARY KEY (personnr,isbn),
  CONSTRAINT autor_buch_fk_isbn FOREIGN KEY (isbn) REFERENCES buch (isbn),
  CONSTRAINT autor_buch_fk_personnr FOREIGN KEY (personnr) REFERENCES autor (persnr)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Referenzielle Integrität
Regel prüfen

```
mysql> delete from autor where persnr='12';

ERROR 1451 (23000):
Cannot delete or update a parent row: a foreign key constraint fails
(`autorbuch/autor_buch`,
CONSTRAINT `autor_buch_fk_personnr`
FOREIGN KEY (`personnr`) REFERENCES `autor` (`persnr`))
```

Referenzielle Integrität
Prinzipien

- Erzeugen, ändern, löschen: ALTER TABLE...
- Löschen: ALTER TABLE 'table' DROP FOREIGN KEY 'name';
- Temporär Regel ausschalten:
  - SET foreign_key_checks = {0, 1};
- Regel muss von Anfang an stimmen!
  - Problem mit existierenden Daten...
- Wieviel Integrität ist erwünscht?
  - Kompromiss *Integrität / Flexibilität* suchen

Referenzielle Integrität
Leichen mit Unterabfrage suchen

```
-- Gibt es Autoren in autor_buch, die aber in autor
-- gelöscht wurden?

mysql> select * from autor_buch where personnr not in
          (select distinct persnr from autor);

Empty set
```

Daten gruppieren

```
mysql> select * from autor_buch;
+----------+------+
| personnr | isbn |
+----------+------+
|       34 | 111  |
|       12 | 123  |
|       34 | 222  |
|       34 | 456  |
|       34 | 789  |
+----------+------+

mysql> select personnr from autor_buch group by personnr;
+----------+
| personnr |
+----------+
|       12 |
|       34 |
+----------+
-- Welche Personen haben mehr als 2 Bücher geschrieben?
mysql> select personnr, count(personnr) from autor_buch group by
personnr having count(personnr) > 2;
+----------+-----------------+
| personnr | count(personnr) |
+----------+-----------------+
|       34 |               4 |
+----------+-----------------+
```

Daten gruppieren
Referenzielle Integrität, PK-Prüfung

```
-- Frage: Ist mein Primary Key wirklich unique?

mysql> select persnr, count(persnr) from autor
       group by persnr having count(persnr) > 1;

Empty set

-- Eigentlich kann man damit nach Duplikaten suchen!
```

Daten aggregieren

- Funktionen: max(), min(), avg(), sum(), count()

```
mysql> select count(personnr) from autor_buch;
+-----------------+
| count(personnr) |
+-----------------+
|               5 |
+-----------------+

mysql> select count(distinct personnr) from autor_buch;
+--------------------------+
| count(distinct personnr) |
+--------------------------+
|                        2 |
+--------------------------+
```

Daten aggregieren
count(*) != count('Spalte')

- Achtung:
  - count(*) => Anzahl *aller Zeilen der Tabelle*
  - count('Spalte') => Anzahl aller *nicht-NULL Zeilen dieser Spalte*

```
-- Dies könnte unterschiedliche Resultate liefern!

select sum(column) / count(*) from table

select sum(column) / count(column) from table
```

Daten gruppieren UND aggregieren

```
mysql> select personNr, count(isbn) from autor_buch
       group by personnr;
+----------+-------------+
| personnr | count(isbn) |
+----------+-------------+
|       12 |           1 |
|       34 |           4 |
+----------+-------------+

mysql> select ab.personnr, a.name, count(isbn)
       from autor_buch ab, autor a where ab.personnr=a.persnr
       group by personnr, name;
+----------+--------+-------------+
| personnr | name   | count(isbn) |
+----------+--------+-------------+
|       12 | Kemper |           1 |
|       34 | Kofler |           4 |
+----------+--------+-------------+
```

Übungen

Meine CD-DB steht Ihnen zur Verfügung:
Die Datei aus https://web.mtg1.bfh.science/SD-RDB/gmcd.sql
herunterladen, DBeaver starten, Localhost Connection eröffnen,
unter dem Menüpunkt SQL-Editor / SQL Script laden und ausführen.

1) Wann wurde die CD "Yellow Submarine" publiziert?

2) Alle Titel und deren Dauer auflisten, die sich auf der CD "The Koeln Concert" befinden (ohne und mit JOIN).

3) Wie heissen die Musiker, die das Instrument "Gitarre" spielen?

4) Wie heissen alle Musiker, die auf der CD "Yellow Submarine" spielen? (mit JOIN, und die Spalte soll "Musiker" heissen).

5) Welches sind die Stücke (Titel auflisten), die länger dauern als die Durchschnittsdauer (Funktion "AVG" verwenden)?

6) Gleiche Frage wie bei 5) aber mit dem WITH-Statement.

7) Setzen Sie referentielle Integritätsregel in die CD DB (mit MySQL CommandLine und/oder Workbench). Und prüfen Sie sie.