



Datenbanken abfragen mit SQL

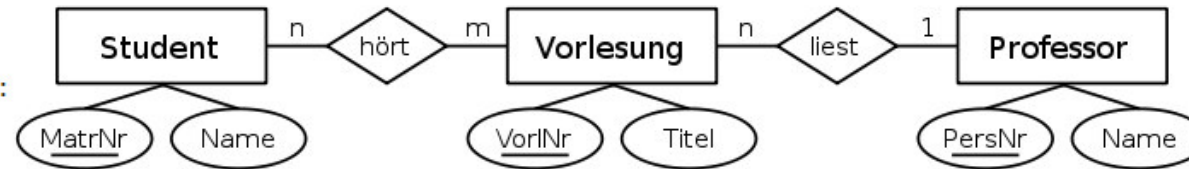
S. Baldes

SQL Überblick

- SQL (Structured Query Language)
- **1970er Jahre Vorgänger SEQUEL (IBM), erste Standardisierung 1987 (SQL-1), 1992 (SQL-2), 1999 (SQL-3), ...**
- SQL besteht aus Data Definition Language (DDL)
 - **CREATE, DROP, ALTER DB: Tabellen anlegen, löschen, ändern**
- Data Manipulation Language (DML)
 - **INSERT, DELETE, UPDATE: Daten einfügen, löschen, ändern**
- **SELECT: Daten anzeigen**
- Wichtigsten Befehle
 - **CREATE TABLE**
 - **INSERT INTO ... VALUES ...**
 - **SELECT ... FROM ... WHERE**

Einfache Abfragen

ER-Diagramm:



Relationen:

Student	
<u>MatrNr</u>	Name
26120	Fichte
25403	Jonas
27103	Fauler

hört	
<u>MatrNr</u>	<u>VorlNr</u>
25403	5001
26120	5001
26120	5045

Vorlesung		
<u>VorlNr</u>	Titel	PersNr
5001	ET	15
5022	IT	12
5045	DB	12

Professor	
<u>PersNr</u>	Name
12	Wirth
15	Tesla
20	Urlauber

Abfrage
gesamte Tabelle

```
SELECT *
FROM Student;
```

MatrNr	Name
26120	Fichte
25403	Jonas
27103	Fauler

Abfrage
mit Spaltenauswahl

```
SELECT VorlNr, Titel
FROM Vorlesung;
```

VorlNr	Titel
5001	ET
5022	IT
5045	DB

Abfrage
mit eindeutigen Werten

```
SELECT DISTINCT
MatrNr FROM hört;
```

MatrNr
25403
26120

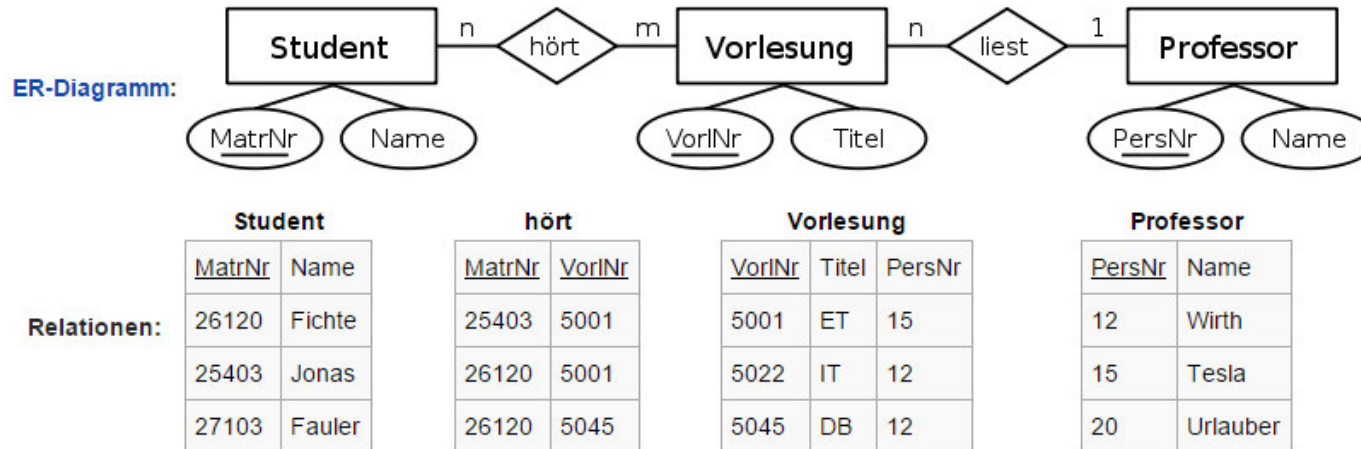
Abfrage
mit Umbenennungen

```
SELECT MatrNr AS
Matrikelnummer, Name
FROM Student;
```

Matrikelnummer	Name
26120	Fichte
25403	Jonas
27103	Fauler

Abfragen mit Auswahl

SELECT ... FROM ... WHERE



Abfrage
mit Filter

```
SELECT VorlNr, Titel
FROM Vorlesung
WHERE Titel = 'ET';
```

VorlNr	Titel
5001	ET

Abfrage
mit Sortierung

```
SELECT * FROM Student
ORDER BY Name DESC;
```

MatrNr	Name
27103	Fauler
26120	Fichte
25403	Jonas

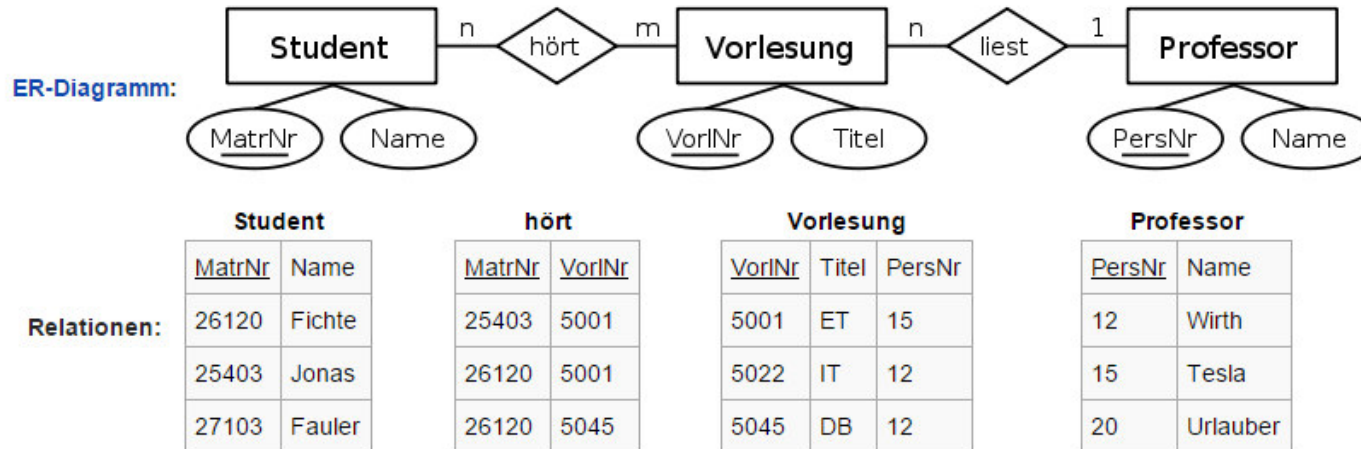
Abfrage
mit verknüpften Tabellen

```
SELECT Vorlesung.Titel, Professor.Name
FROM Professor, Vorlesung
WHERE Vorlesung.PersNr = Professor.PersNr;
```

Titel	Name
ET	Tesla
IT	Wirth
DB	Wirth

Abfragen mit Auswahl

SELECT ... FROM ... WHERE



Abfrage
mit Filter

```
SELECT VorlNr, Titel
FROM Vorlesung
WHERE Titel = 'ET';
```

VorlNr	Titel
5001	ET

Abfrage
mit Sortierung

```
SELECT * FROM Student
ORDER BY Name DESC;
```

MatrNr	Name
27103	Fauler
26120	Fichte
25403	Jonas

Abfrage
mit verknüpften Tabellen

```
SELECT Vorlesung.Titel, Professor.Name
FROM Professor, Vorlesung
WHERE Vorlesung.PersNr = Professor.PersNr;
```

Titel	Name
ET	Tesla
IT	Wirth
DB	Wirth

Weitere Abfragen mit Auswahl

SELECT ... FROM ... WHERE

- Vergleichsoperatoren =, <, >, <=, >=, !=
- ***SELECT Name FROM City WHERE Population > 100000;***
- Matching **[NOT] LIKE % _**
- ***SELECT Name FROM Country WHERE Name LIKE '__e%';***
- % entspricht * bei regulären Ausdrücken, Platzhalter für null oder beliebige Zeichen
- _ entspricht ? bei regulären Ausdrücken, Platzhalter für genau ein beliebiges Zeichen
- Zwischen **[NOT] BETWEEN**
- ***SELECT Name FROM City WHERE Latitude BETWEEN -5 AND 5;***
- Enthalten **[NOT] IN**
- ***SELECT Name FROM City WHERE Name NOT in ('Lahr', 'Offenburg', 'Ulm');***
- Leerheit **IS [NOT] NULL**
- ***SELECT Name FROM City WHERE River IS NOT NULL;***

Weitere Abfragen mit Auswahl

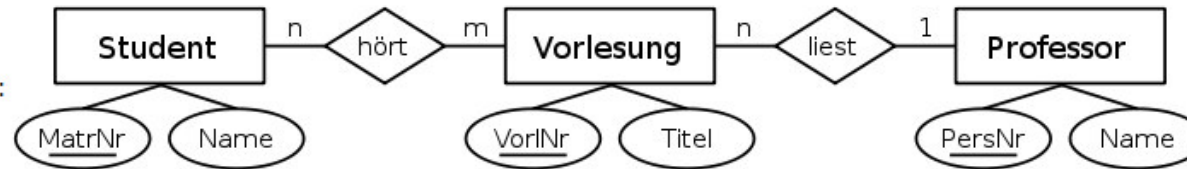
SELECT ... FROM ... WHERE

- Logische Operatoren: **AND, OR, NOT**
- **SELECT Name FROM City**
WHERE (Country = 'de') OR (Country = 'fr' AND Population > 400000);
- **NOT A OR B AND C** entspricht **(NOT A) OR (B AND C)** //Bindung NOT AND OR
- Aufsteigend/Absteigend **sortieren**
- **SELECT * FROM Customers ORDER BY Country ASC, Name DESC;**
- Anzahl Datensätze begrenzen **LIMIT**
- **SELECT * FROM City LIMIT 5;**

Aggregats-Funktionen

count, max/min, sum, avg

ER-Diagramm:



Relationen:

Student	
<u>MatrNr</u>	Name
26120	Fichte
25403	Jonas
27103	Fauler

hört	
<u>MatrNr</u>	<u>VorlNr</u>
25403	5001
26120	5001
26120	5045

Vorlesung		
<u>VorlNr</u>	Titel	PersNr
5001	ET	15
5022	IT	12
5045	DB	12

Professor	
<u>PersNr</u>	Name
12	Wirth
15	Tesla
20	Urlauber

count()

Anzahl Zeilen
zählen

```
SELECT count(*)
FROM Vorlesung;
```

count(*)
3

max() / min()

größte/kleinst
Wert

```
SELECT max(PersNr)
FROM Vorlesung;
```

max(PersNr)
15

sum()

Summe der
Werte

```
SELECT sum(PersNr)
FROM Vorlesung;
```

sum(PersNr)
39

avg()

Mittelwert der
Werte

```
SELECT avg(PersNr)
FROM Vorlesung;
```

avg(PersNr)
13

```
SELECT
count(DISTINCT PersNr)
FROM Vorlesung;
```

count(DISTINCT PersNr)
2

Beachte: **Keine weiteren Attribute im SELECT erlaubt!**
Ausnahme: GROUP BY (s. nächste Folie)

Gruppierungen

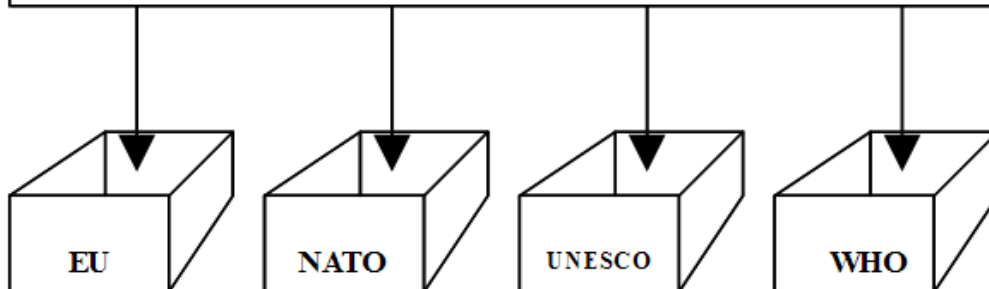
GROUP BY

<u>Organization</u>	<u>Country</u>	Type
EU	de	member
WHO	de	member
UNESCO	at	member
UNESCO	af	member
NATO	us	member
WHO	af	member
UNSECO	de	member
UNESCO	fr	member

**GROUP BY
Organization**

<u>Organization</u>	<u>Country</u>	Type
EU	de	member
NATO	us	member
UNESCO	af	member
UNSECO	de	member
UNESCO	fr	member
UNESCO	at	member
WHO	af	Member
WHO	de	member

D A T E N S Ä T Z E gruppiert nach
der Spalte **Organization** in der Tabelle *is_member*



SELECT Organization, **count(*)**
FROM is_member
GROUP BY Organization;

Organization	count(*)
EU	1
NATO	1
UNESCO	4
WHO	2

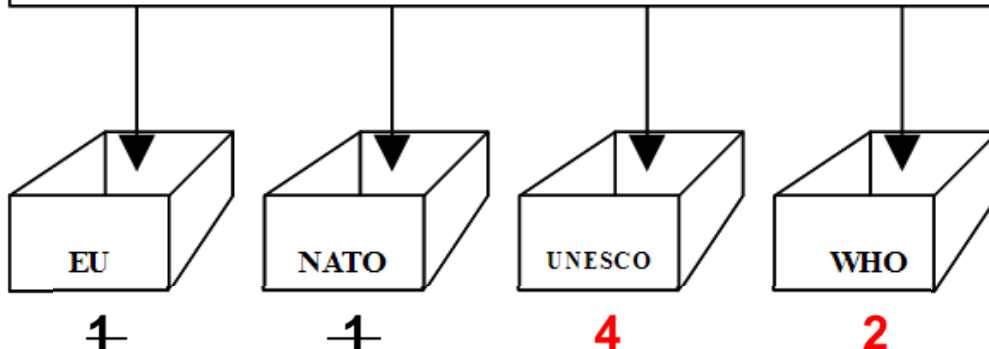
Auswahl der einzelnen Gruppen HAVING

Organization	Country	Type
EU	de	member
WHO	de	member
UNESCO	at	member
UNESCO	af	member
NATO	us	member
WHO	af	member
UNSECO	de	member
UNESCO	fr	member

**GROUP BY
Organization**

Organization	Country	Type
EU	de	member
NATO	us	member
UNESCO	af	member
UNSECO	de	member
UNESCO	fr	member
UNESCO	at	member
WHO	af	Member
WHO	de	member

D A T E N S Ä T Z E gruppiert nach
der Spalte **Organization** in der Tabelle *is_member*



SELECT Organization, **count(*)**
FROM is_member
GROUP BY Organization
HAVING count(*)>=2;

Organization	count(*)
UNESCO	4
WHO	2

Zusammenfassung SQL-Select-Statement

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

Besonderheiten SQL

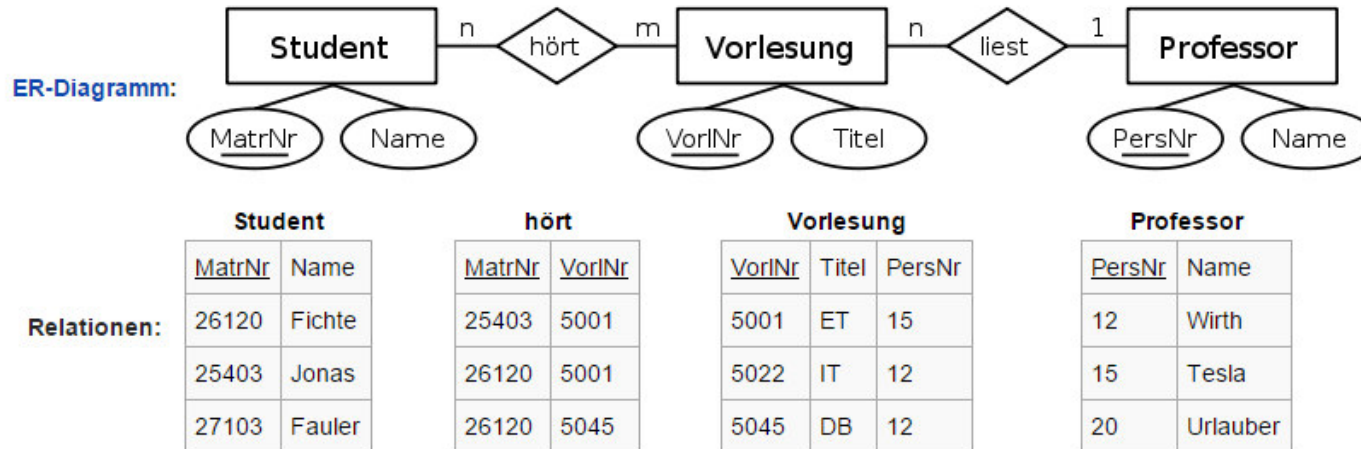
- **Rechnen** im SELECT-Ausdruck

```
SELECT anzahl * preis AS "Gesamtkosten"  
...
```

- Mehrere Anfragen mit **Union** verknüpfen (bei identischen SELECT-Ausdruck)

```
SELECT sum(Kosten) FROM Halbjahr1  
UNION  
SELECT sum(Kosten) FROM Halbjahr2;
```

Abfragen mit verknüpften Tabellen WHERE / JOIN



Abfrage
mit WHERE

```
SELECT Vorlesung.Titel, Professor.Name
FROM Vorlesung, Professor
WHERE Vorlesung.PersNr = Professor.PersNr;
```

Titel	Name
ET	Tesla
IT	Wirth
DB	Wirth

Abfrage
mit JOIN

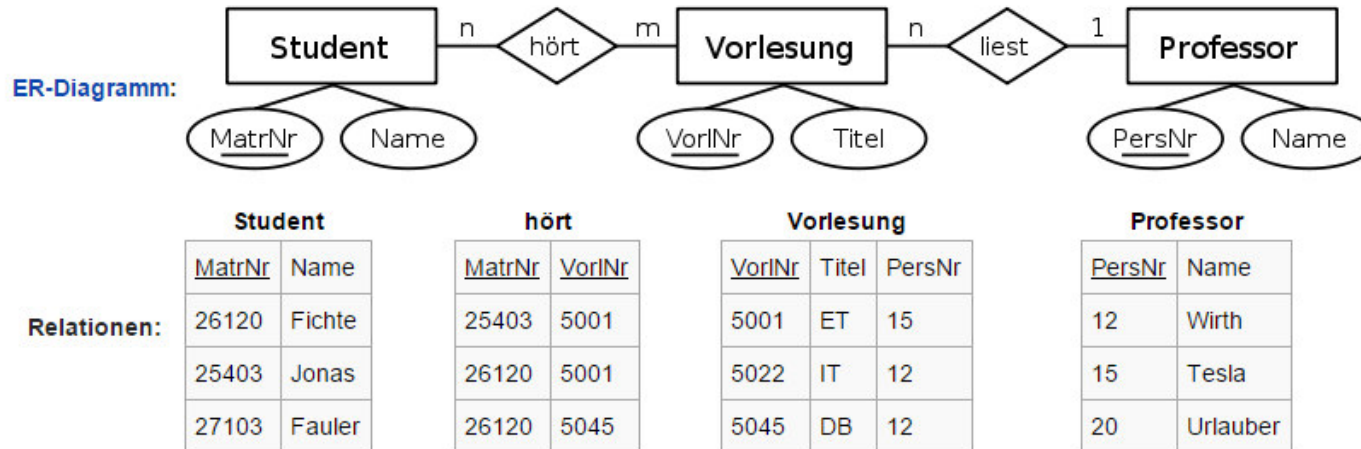
```
SELECT Vorlesung.Titel, Professor.Name
FROM Vorlesung JOIN Professor
ON Vorlesung.PersNr = Professor.PersNr;
```

VorlNr	Titel	PersNr	PersNr	Name
5001	ET	15	12	Wirth
5001	ET	15	15	Tesla
5001	ET	15	20	Urlauber
5022	IT	12	12	Wirth
...				



Titel	Name
ET	Tesla
IT	Wirth
DB	Wirth

Abfragen mit verknüpften Tabellen und Auswahl (WHERE / JOIN)



Abfrage
mit WHERE

```
SELECT Vorlesung.Titel, Professor.Name
FROM Vorlesung, Professor
WHERE Vorlesung.PersNr = Professor.PersNr
AND Professor.Name='Wirth';
```

Abfrage
mit JOIN

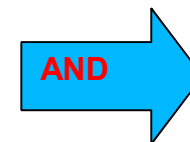
```
SELECT Vorlesung.Titel, Professor.Name
FROM Vorlesung JOIN Professor
ON Vorlesung.PersNr = Professor.PersNr
AND Professor.Name='Wirth';
```

Vorlesung JOIN Professor

VorlNr	Titel	PersNr	PersNr	Name
5001	ET	15	12	Wirth
5001	ET	15	15	Tesla
5001	ET	15	20	Urlauber
5022	IT	12	12	Wirth
...				



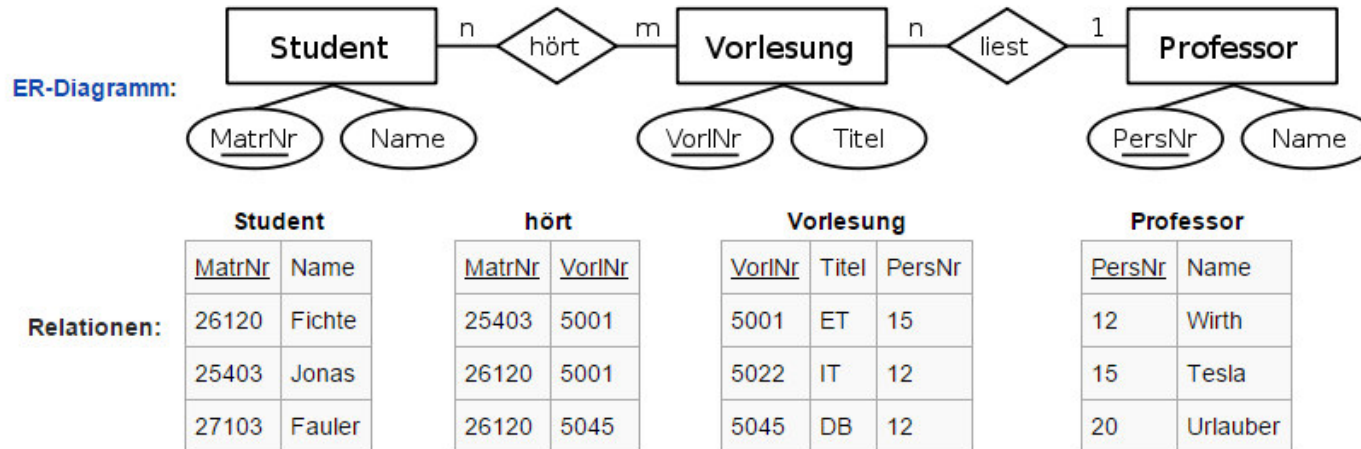
Titel	Name
ET	Tesla
IT	Wirth
DB	Wirth



Titel	Name
IT	Wirth
DB	Wirth

Left / Right / Full Join

Anzeigen auf bei leeren Felder



Abfrage mit WHERE

```
SELECT Professor.Name, Vorlesung.Titel
FROM Vorlesung, Professor
WHERE Vorlesung.PersNr = Professor.PersNr;
```

Name	Titel
Wirth	IT
Wirth	DB
Tesla	ET

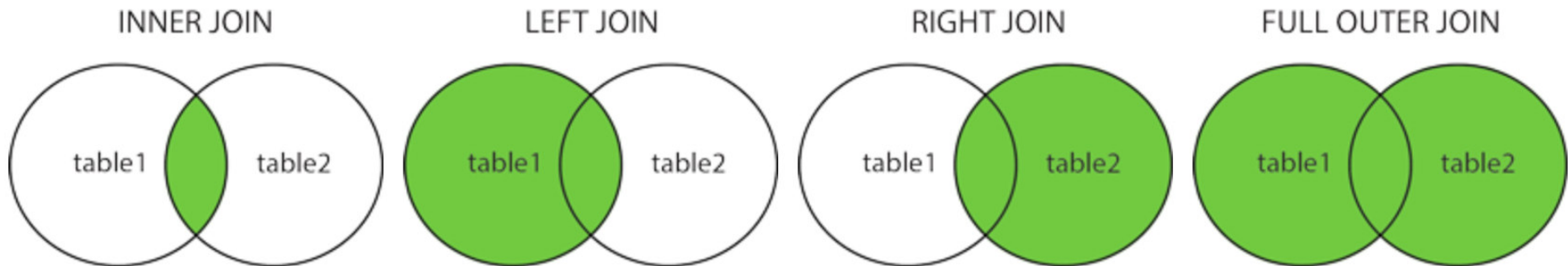
Abfrage mit LEFT JOIN

Auch Professoren ohne Vorlesungen

```
SELECT Vorlesung.Titel, Professor.Name
FROM Vorlesung LEFT JOIN Professor
ON Vorlesung.PersNr = Professor.PersNr;
```

Name	Titel
Wirth	IT
Wirth	DB
Tesla	ET
Urlauber	null

Left / Right / Full Join Funktion



- **(INNER) JOIN:** Es werden nur Zeilen ausgegeben, deren Felder in **beiden** Tabellen **nicht leer** sind (On-Bedingung).
- **LEFT (OUTER) JOIN:** Es werden alle Zeilen ausgegeben, deren Felder der **linken** Tabelle **nicht leer** sind (Im Bsp: Auch Professoren, die keine Vorlesung haben).
- **RIGHT(OUTER) JOIN:** Es werden alle Zeilen ausgegeben, deren Felder der **rechten** Tabelle **nicht leer** sind (Im Bsp: Auch Vorlesungen, deren Professor noch nicht bekannt ist).
- **FULL (OUTER) JOIN:** Es werden alle Zeilen ausgegeben, auch wenn deren Felder **leer** sind (Im Bsp: Auch Professoren ohne Vorlesung und Vorlesungen ohne Professoren).

Bemerkung: Die Wörter in Klammern können weggelassen werden.

Self-Join

Tabellen mit sich selbst verknüpfen

Professor	
<u>PersNr</u>	Name
12	Wirth
15	Tesla
20	Urlauber

Die Professoren spielen gegeneinander Schach. Erstelle eine Turnier-Tabelle dazu.

Abfrage
mit WHERE

```
SELECT p1.Name As "P1", p2.Name AS "P2"  
FROM Professor p1, Professor p2  
WHERE p1.Name <> p2.Name;
```

Abfrage
mit JOIN

```
SELECT p1.Name As "P1", p2.Name AS "P2"  
FROM Professor p1 join Professor p2  
ON p1.Name <> p2.Name;
```

P1	P2
Wirth	Wirth
Tesla	Wirth
Urlauber	Wirth
Wirth	Tesla
Tesla	Tesla
Urlauber	Tesla
Wirth	Urlauber
Tesla	Urlauber
Urlauber	Urlauber



P1	P2
Tesla	Wirth
Urlauber	Wirth
Wirth	Tesla
Urlauber	Tesla
Wirth	Urlauber
Tesla	Urlauber

Unterabfrage (Subquery)

schuelerin

name	note
Michaela	3,4
Kati	1,2
Conny	1,8
Linda	4,1
Andrea	2,1
Susanne	2,1

Die Schülerin mit der besten Note

```
SELECT name, note FROM schuelerin  
WHERE note =  
    (SELECT MIN(note) FROM schuelerin);
```

Unterabfrage,
Liefert einen
eigenen Wert
oder Tabelle;

hier: 1,2

Ergebnistabelle	
name	note
Kati	1,2

Unterabfragen mit IN, ALL, ANY, SOME, EXISTS

schuelerin

name	note
Michaela	3,4
Kati	1,2
Conny	1,8
Linda	4,1
Andrea	2,1
Susanne	2,1

schueler

name	note
Fabian	3,7
Gerd	2,1
Matthias	3,5
Werner	2,6
Leo	4,1
Alex	2,7

Die Schülerinnen, deren Noten besser ist
als die aller männlichen Schüler.

```
SELECT name, note FROM schuelerin  
WHERE note <  
    ALL (SELECT note FROM schueler);
```

Ergebnistabelle

name	note
Kati	1,2
Conny	1,8

Unterabfrage,
Liefert einen
eigenen Wert
oder Tabelle;

hier: Tabelle
mit Noten aller
Schüler

note
3,7
2,1
3,5
2,6
4,1
2,7

Quellen

- <http://www.w3schools.com/sql/default.asp>
- <http://de.wikipedia.org/wiki/SQL>
- Ahmad Nessar Nazar: Unterrichtsunterlagen
- Michael Dienert: Unterrichtsunterlagen