Abfragen aus mehreren Tabellen (mit join)

Bislang wurden nur einzelne Tabellen eingesetzt. Das reicht oft auch aus, doch sobald man etwas komplexere Webprojekte realisieren möchten, wird man kaum umhinkommen, mit mehreren Tabellen zu arbeiten, deren Inhalte untereinander in Beziehung (Relation) stehen und die für die Ausgabe im Browser verknüpft werden.

Arbeiten in der Datenbank „garten“

Übung:
Es solle jede Pflanze angezeigt werden und daneben der Firmenname des Lieferanten. Dafür muss man die Tabelle „pflanzen“ und die Tabelle „lieferanten“ verknüpfen.

Bei FROM kann man beide Tabellen angeben, bei WHERE muss man festlegen, über welches Feld die beiden Tabellen verknüpft werden sollen:

SELECT name, firma
FROM pflanzen, lieferanten
WHERE pflanzen.liefer_id = lieferanten.liefer_id;

```
1 SELECT name, firma
2 FROM pflanzen, lieferanten
3 WHERE pflanzen.liefer_id = lieferanten.liefer_id;
4
```

Da die Spalte liefer_id in beiden Tabellen existiert, muss man in der WHERE-Klausel zusätzlich vor dem Spaltennamen **mit einem Punkt den Namen der Tabelle** angeben.

Ergebnis:

+ Optionen

name	firma
Feldahorn	Gärtnerbedarf Müller
Warzenbirke	Grünes Allerlei
Pfeifenwinde	Grünes Allerlei
Filigranfarn	Grüner Finger
Kahle Apfelbeere	Grüner Finger
Essigrose	Gärtnerbedarf Müller
Büffelbeere	Gärtnerbedarf Müller
Hopfenbuche	Grüner Finger
Baumwürger	Grüner Finger

- **INNER JOIN - liefert dasselbe Ergebnis**

Dieses SELECT lässt sich auch über eine alternative Syntax schreiben, über den INNER JOIN:

SELECT name, firma
FROM pflanzen
INNER JOIN lieferanten
ON pflanzen.liefer_id = lieferanten.liefer_id;

```
1 SELECT name, firma
2 FROM pflanzen
3 INNER JOIN lieferanten
4 ON pflanzen.liefer_id = lieferanten.liefer_id;
5 |
```

- **GROUP BY**

Damit erreicht man, das gleiche Lieferanten-IDs zusammengefasst werden. Man kann ermitteln, wie viele Pflanzen die einzelnen Lieferanten liefern.

SELECT liefer_id, COUNT(*)
FROM pflanzen
GROUP BY liefer_id;

Ergebnis;

+ Optionen

liefer_id	COUNT(*)
1	3
2	2
3	4

Soll nun der Firmenname vorne angezeigt werden, anstelle der liefer_id, so müssen die Tabellen verknüpft werden:

SELECT firma, COUNT(*) AS Anzahl
FROM pflanzen, lieferanten
WHERE pflanzen.liefer_id = lieferanten.liefer_id
GROUP BY pflanzen.liefer_id;

Ergebnis: Name des Lieferanten, mit der Anzahl an Pflanzen, die er liefert.

+ Optionen

firma	Anzahl
Gärtnerbedarf Müller	3
Grünes Allerlei	2
Grüner Finger	4

- **HAVING**

Mit HAVING kann man eine Bedingung formulieren, dass z.B. nur Lieferanten angezeigt werden, die mehr als 2 Pflanzen liefern

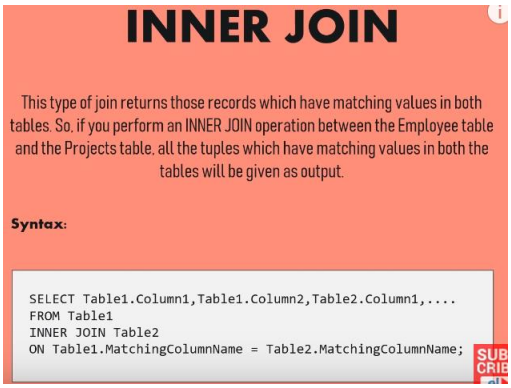
```
SELECT firma, COUNT(*) AS Anzahl
FROM pflanzen, lieferanten
WHERE pflanzen.liefer_id = lieferanten.liefer_id
GROUP BY pflanzen.liefer_id
HAVING Anzahl > 2;
```

Ergebnis:

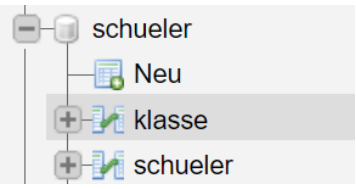
+ Optionen

firma	Anzahl
Gärtnerbedarf Müller	3
Grüner Finger	4

Arten von JOINS:



Übung mit der Datenbank „schueler“



Die Datenbank „schueler“ soll angelegt werden:

1)Erstelle eine Datenbank für Schüler

a)Datenbank „schueler“ mit Tabelle „schueler“. Füge vier Datensätze ein. Bild muss nicht sein, war bei den beiden oberen schon da.

s_id	vorname	nachname	geschlecht	klasse	geb	bild	benutzername	email	pass
15	Melvin	Berthold	m	3CK	2002-11-11	[BLOB - 28,1 KiB]	melvin-r	melvin@gmx.at	$2y$10$JQCh0k8sKZaa1i565MPR4uDoneHHs6R.paizpvnetAg...
17	Kevin	Eckelhart	m	3CK	2002-02-02	[BLOB - 19,0 KiB]	kevin	kevin@gmx.at	$2y$10$Biu7.ObXBshMnSwe6vjWPe8VuZqIfQp46oGuoRbZEfW...
23	Alex	Müller	m	4CK	2002-08-12	*NULL*	alex	alexmueller@gmail.com	hallo
24	Sandra	Schöller	w	3AK	2003-06-10	*NULL*	sandra	sandrasch@protonmail.ch	1234

b)Daneben die Tabelle „klasse“. Füge vier Datensätze auch hier ein:

Beachte:

- die „klasse_id“ **ohne** „auto-inkrement“ und als Typ soll sein „varchar“ aber auch mit den Stellen „3“, wie auch in der Tabelle „schueler“ – muss gleich sein
- beamer – als „text“ mit 4 Stellen (für ja oder nein)
- digbiz - genau so

Tabellenstruktur | Beziehungsansicht

	#	Name	Typ	Kollation	Attribute	Null	Standard	Kommentare	Extra	Aktion
☐	1	**klasse_id**	varchar(3)	utf8_general_ci		Nein	*kein(e)*			Bearbeiten
☐	2	**flaeche**	decimal(9,2)			Nein	*kein(e)*			Bearbeiten
☐	3	**beamer**	text	utf8_general_ci		Nein	*kein(e)*			Bearbeiten
☐	4	**digbiz**	text	utf8_general_ci		Nein	*kein(e)*			Bearbeiten

Füge 4 Datensätze ein.

	klasse_id	flaeche	beamer	digbiz
Bearbeiten Kopieren Löschen	3AK	70.00	nein	nein
Bearbeiten Kopieren Löschen	3CK	80.00	ja	ja
Bearbeiten Kopieren Löschen	4AK	65.00	ja	nein
Bearbeiten Kopieren Löschen	4CK	90.00	ja	ja

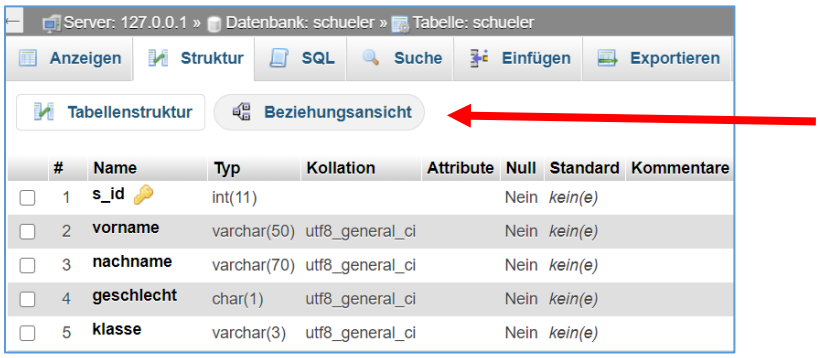
Beziehungen erstellen

Nun soll die Beziehung zwischen den folgenden Spalten hergestellt werden.

Beachte: Nur wenn in beiden Tabellen auch Daten eingegeben sind, kann man eine Beziehung herstellen.

- klasse in der Tabelle „schueler“
- klassen_id in der Tabelle „klasse“

Öffne die „Beziehungsansicht“ in der Tabelle „schueler“.



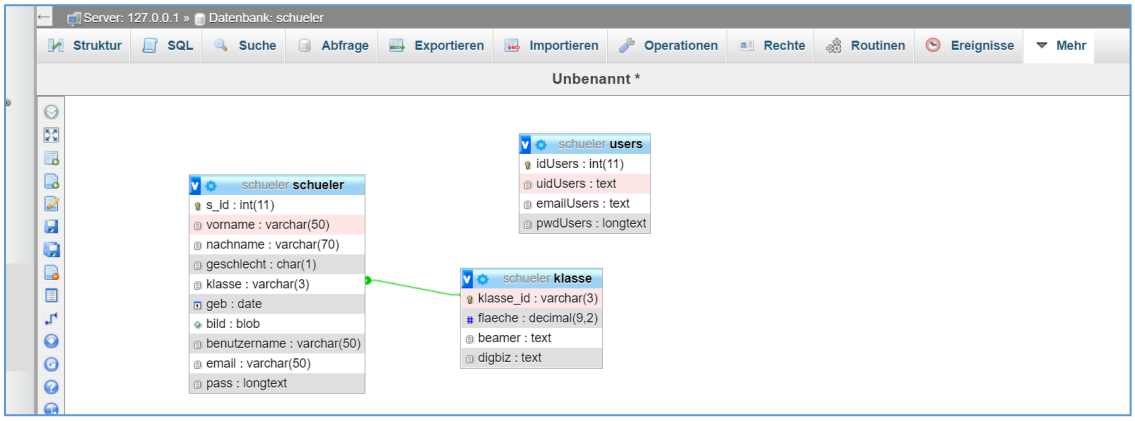
Fremdschlüssel „klasse“ zur „schueler“-Tabelle einrichten:

1. Man muss sich in der Tabelle befinden, in der der Fremdschlüssel ist, hier also „schueler“.
2. Dort dann in der „Struktur“-Ansicht klicke
3. auf „Beziehungsansicht“ und

- gib einen Text an, der beide Felder betrifft – FS (für Fremdschlüssel) FS_klasse_zu_schueler
- Spalte „klasse“
- wähle die Datenbank „schueler“ aus
- dann die Referenztabelle „klasse“ – wo auch der Primärschlüssel ist
- und in der Spalte die ID zur Tabelle.
- für „delete“ – RESTRICT (wenn eine Klasse beim Schüler gelöscht wird, soll was was passieren, weil ohne Klasse soll keiner sein)
- für „update“ auch RESTRICT.

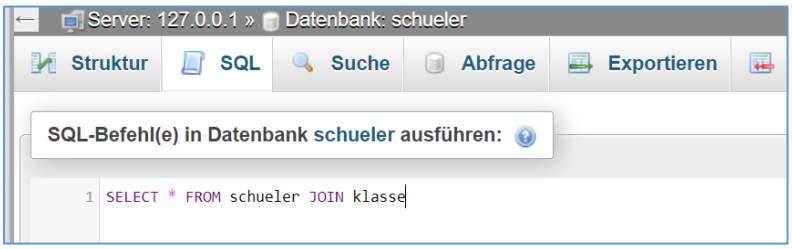
Zur Kontrolle kann man die erfolgreiche Setzung der Beziehung auch in der „Designer“-Ansicht finden. Dazu darf man nicht in einer Tabelle sich befinden, sondern in der Datenbank ansich. Dort

unter dem Reiter ganz rechts „Mehr“ auf „Designer“ klicken:



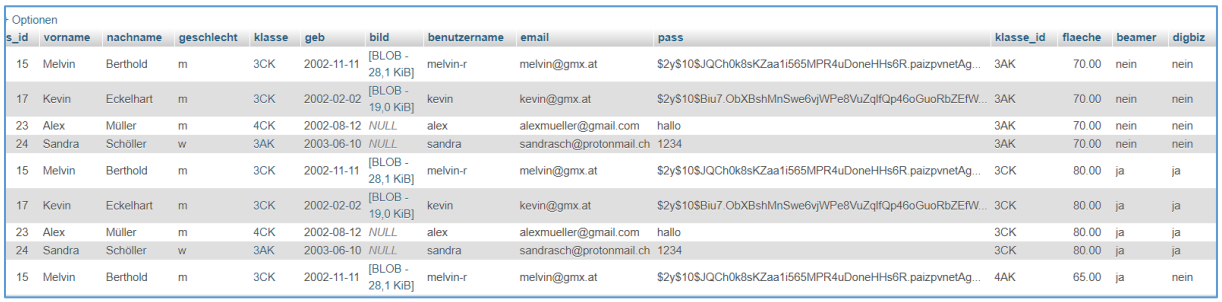
Nun folgend die Abfragen aus diesen beiden Tabellen:

1. Allgemeine Abfrage:
   SELECT * FROM schueler JOIN klasse



Dabei werden alle Datensätze der einen Datei mit allen Datensätzen der anderen kombiniert. Das ist nicht sonderlich sinnvoll. https://www.youtube.com/watch?v=L3J8abbGPCQ 2:45

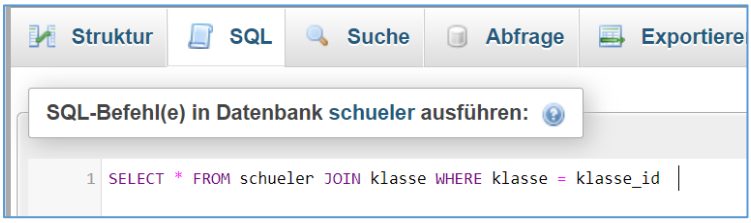
Ergebnis:



+ Optionen

s_id	vorname	nachname	geschlecht	klasse	geb	bild	benutzername	email	pass	klasse_id	flaeche	beamer	digbiz
15	Melvin	Berthold	m	3CK	2002-11-11	[BLOB - 28,1 KiB]	melvin-r	melvin@gmx.at	$2y$10$JQCh0k8sKZaa1i565MPR4uDoneHHs6R.paizpvnetAg...	3AK	70.00	nein	nein
17	Kevin	Eckelhart	m	3CK	2002-02-02	[BLOB - 19,0 KiB]	kevin	kevin@gmx.at	$2y$10$Biu7.ObXBshMnSwe6vjWPe8VuZqIfQp46oGuoRbZEfW...	3AK	70.00	nein	nein
23	Alex	Müller	m	4CK	2002-08-12	NULL	alex	alexmueller@gmail.com	hallo	3AK	70.00	nein	nein
24	Sandra	Schöller	w	3AK	2003-06-10	NULL	sandra	sandrasch@protonmail.ch	1234	3AK	70.00	nein	nein
15	Melvin	Berthold	m	3CK	2002-11-11	[BLOB - 28,1 KiB]	melvin-r	melvin@gmx.at	$2y$10$JQCh0k8sKZaa1i565MPR4uDoneHHs6R.paizpvnetAg...	3CK	80.00	ja	ja
17	Kevin	Eckelhart	m	3CK	2002-02-02	[BLOB - 19,0 KiB]	kevin	kevin@gmx.at	$2y$10$Biu7.ObXBshMnSwe6vjWPe8VuZqIfQp46oGuoRbZEfW...	3CK	80.00	ja	ja
23	Alex	Müller	m	4CK	2002-08-12	NULL	alex	alexmueller@gmail.com	hallo	3CK	80.00	ja	ja
24	Sandra	Schöller	w	3AK	2003-06-10	NULL	sandra	sandrasch@protonmail.ch	1234	3CK	80.00	ja	ja
15	Melvin	Berthold	m	3CK	2002-11-11	[BLOB - 28,1 KiB]	melvin-r	melvin@gmx.at	$2y$10$JQCh0k8sKZaa1i565MPR4uDoneHHs6R.paizpvnetAg...	4AK	65.00	ja	nein

2. Daher muss man eine Einschränkung vornehmen, wo der Fremdschlüssel mit dem Primärschlüssel übereinstimmt.
   SELECT * FROM schueler JOIN klasse WHERE klasse (Fremdschlüssel beim schueler) = klasse_id (Primärschlüssel bei Klasse)



Ergebnis: schon viel besser

Alles anzeigen | Anzahl der Datensätze: 25 Zeilen filtern: Diese Tabelle durchsuchen Nach Schlüssel sortieren: keine

+ Optionen

s_id	vorname	nachname	geschlecht	klasse	geb	bild	benutzername	email	pass	klasse_id	flaeche	beamer	digbiz
24	Sandra	Schöller	w	3AK	2003-06-10	NULL	sandra	sandrasch@protonmail.ch	1234	3AK	70.00	nein	nein
15	Melvin	Berthold	m	3CK	2002-11-11	[BLOB - 28,1 KiB]	melvin-r	melvin@gmx.at	$2y$10$JQCh0k8sKZaa1i565MPR4uDoneHHs6R.paizpvnetAg...	3CK	80.00	ja	ja
17	Kevin	Eckelhart	m	3CK	2002-02-02	[BLOB - 19,0 KiB]	kevin	kevin@gmx.at	$2y$10$Biu7.ObXBshMnSwe6vjWPe8VuZqlfQp46oGuoRbZEfW...	3CK	80.00	ja	ja
23	Alex	Müller	m	4CK	2002-08-12	NULL	alex	alexmueller@gmail.com	hallo	4CK	90.00	ja	ja

3. alle Vornamen und Nachnamen, die einen Beamer in der Klasse haben

```
SELECT vorname, nachname FROM schueler JOIN klasse ON klasse = klasse_id WHERE beamer = "ja"
```

SELECT vorname, nachname FROM schueler JOIN klasse ON klasse = klasse_id WHERE beamer = "ja"

Ergebnis:

+ Optionen

vorname	nachname
Melvin	Berthold
Kevin	Eckelhart
Alex	Müller

BEACHTE:

- ON
  steht zwischen dem „JOIN“ und „WHERE“ und vergleicht die beiden Felder (Fremdschlüssel und ID)
- Ist ein Name einer Spalte in einer Tabelle auch in einer anderen Tabelle gleich, dann muss man zur eindeutigen Identifizierung den Tabellennamen davor stellen mit Punkt dazwischen
  Beispiel:

```
SELECT vorname, nachname FROM schueler JOIN klasse ON schueler.klasse = klasse.klasse_id  WHERE beamer = "ja"
```

Weitere Infos:

SELECT * FROM kunden JOIN ort

Dabei werden alle Datensätze der einen Datei mit allen Datensätzen der anderen kombiniert. Das ist nicht sonderlich sinnvoll. https://www.youtube.com/watch?v=L3J8abbGPCQ 2:45

Daher muss man eine Einschränkung vornehmen, wo der Fremdschlüssel mit dem Primärschlüssel übereinstimmt.
SELECT * FROM kunden JOIN ort WHERE ort_plz (Fremdschlüssel beim Kunden) = plz (Primärschlüssel bei Ort

```
SELECT * FROM `products` JOIN korb ON korb.product_id = products.id WHERE
korb.id = 6
```

+ Optionen

id	titel	beschreibung	preis	id	product_id	user_id	anzahl	created
2	10 kg Orangen	süße 10 Kilo Orangen	8.00	6	2	274580567	0	0000-00-00 00:00:00

```
SELECT products.id,titel,beschreibung,preis FROM products JOIN korb ON
korb.product_id = products.id WHERE korb.user_id = '274580567'
```

+ Optionen

id	titel	beschreibung	preis
1	5kg Orangen	ein kleiner Kübel mit Orangen	5.00
2	10 kg Orangen	süße 10 Kilo Orangen	8.00
3	Ernte des ganzen Baumes	je nach Wachstum kann das sehr viel sein	99.00
4	drei Bäume gesamt	die Ernte von 3 Bäumen	249.00

https://www.youtube.com/watch?v=bLL5NbBEg2I