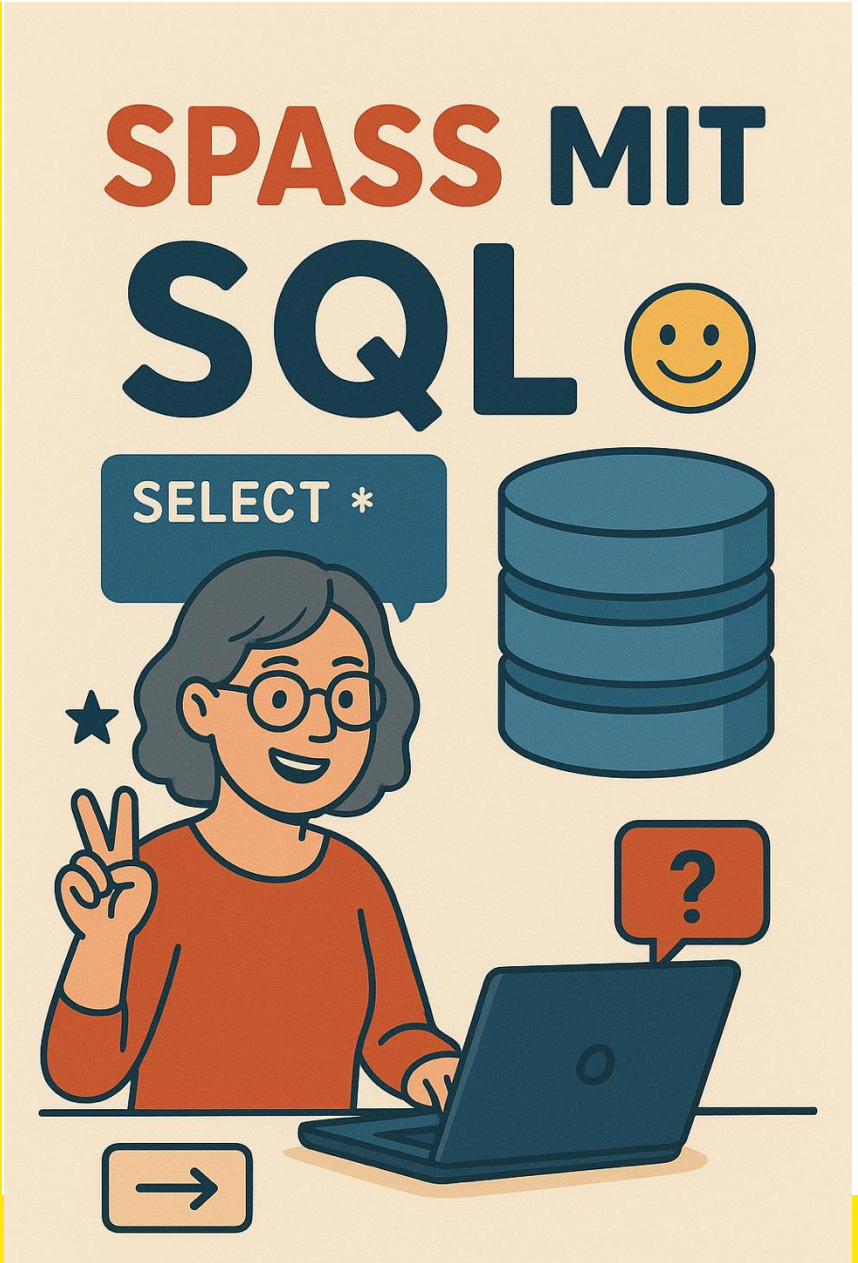
Fachhochschule Nordwestschweiz

Spass mit SQL

Wie Studis heute SQL lernen – und was du davon mitnehmen kannst

Dr. Andrea Kennel, fhnw

20. November 2025



Dr. Andrea Kennel

Consultant

Dozentin für Datenbanken

Coach für Project Management

Fachhochschule Nordwestschweiz

Brugg/Windisch, Schweiz

andrea.kennel@fhnw.ch
andrea@infokennel.ch
www.infokennel.ch

Gruppenaufgabe

Fülle Deinen Datensatz aus:

Name und **Vorname** müssen korrekt gefüllt werden

Kinder (Anzahl der Kinder), **Wohnort** und **Geburtsdatum** dürfen erfunden sein.

Gruppenaufgabe

```
SELECT * FROM person;
```

Gruppenaufgabe

```
SELECT * FROM person ORDER BY vorname;
```

Gruppenaufgabe

```
SELECT DISTINCT vorname FROM person;
```

Gruppenaufgabe

```
SELECT * FROM person WHERE kinder > 0;

SELECT vorname FROM person
WHERE kinder > 0;
```

Gruppenaufgabe

```
SELECT sum (kinder) FROM person
WHERE anzahl_kinder > 0;

SELECT sum (kinder) FROM person;
```

Gruppenaufgabe

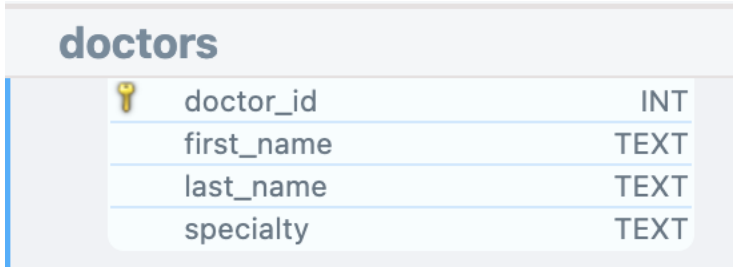
```
SELECT count(*) FROM person
  WHERE geburtstag >= TO_DATE('1995-01-01');

SELECT count(name) FROM person
  WHERE geburtstag >= TO_DATE('1995-01-01');

SELECT count(1) FROM person
  WHERE geburtstag >= TO_DATE('1995-01-01');
```

Ein paar Wochen später

Prüfungsfrage 1 SELECT



Formuliere eine SQL-Abfrage, so dass Vorname, Nachname und Spezialgebiet der Ärztinnen und Ärzte (Tabelle doctors) ausgegeben werden. Dabei sollen alle Ärztinnen und Ärzte der Spezialgebiete (specialty) 'Oncologist' und 'Cardiologist' aufgelistet werden, sortiert nach Spezialgebiet und Nachname.
Die Anweisung soll auch für eine andere Datenlage korrekt sein.

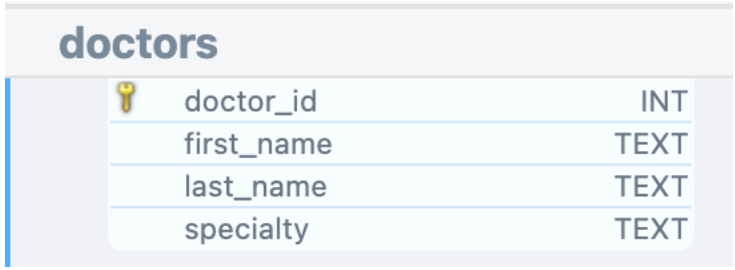
```
SELECT first_name, last_name, specialty
FROM patients
ORDER BY speciality ASC;
```

```
SELECT first_name, last_name, specialty FROM doctors
Where specialty = 'Oncologist' or specialty = 'Cardiologist'
order by specialty, last_name asc;
```

```
SELECT first_name, last_name, specialty
FROM doctors
where specialty = 'Cardiologist' And speciality =
'Oncologist’
```

```
select first_name, last_name,specialty from doctors
where specialty=('Oncologist' or 'Cardiologist')
order by specialty, last_name;
```

Prüfungsfrage 1 SELECT



Formuliere eine SQL-Abfrage, so dass Vorname, Nachname und Spezialgebiet der Ärztinnen und Ärzte (Tabelle doctors) ausgegeben werden. Dabei sollen alle Ärztinnen und Ärzte der Spezialgebiete (specialty) 'Oncologist' und 'Cardiologist' aufgelistet werden, sortiert nach Spezialgebiet und Nachname.
Die Anweisung soll auch für eine andere Datenlage korrekt sein.

```
SELECT first_name, last_name, specialty
FROM patients
ORDER BY speciality ASC;
```

```
SELECT first_name, last_name, specialty FROM doctors
Where specialty = 'Oncologist' or specialty = 'Cardiologist'
order by specialty, last_name asc;
```

```
SELECT first_name, last_name, specialty
FROM doctors
where specialty = 'Cardiologist' And speciality =
'Oncologist’
```

```
select first_name, last_name,specialty from doctors
where specialty=('Oncologist' or 'Cardiologist')
order by specialty, last_name;
```

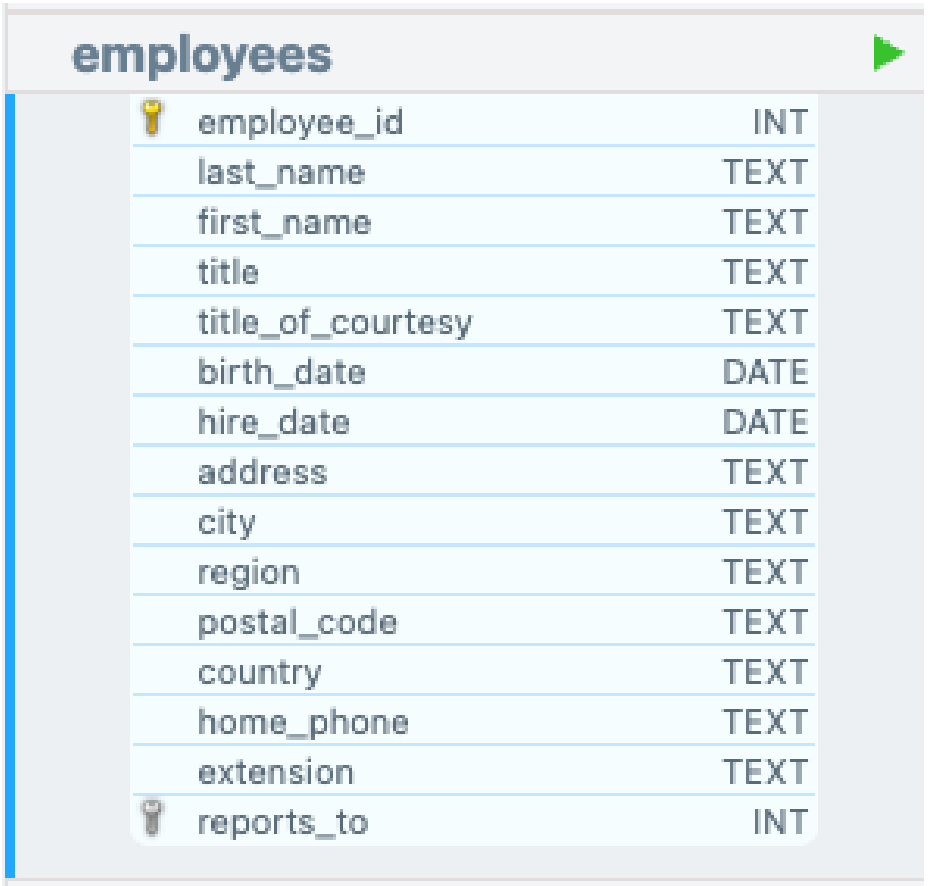
Prüfungsfrage 6 SELF_JOIN

In der Tabelle **employees** gibt das Attribut **reports_to** an, wer Chef/in ist.

Erstelle eine SQL-Anweisung, die Mitarbeitende auflistet, die in der gleichen Stadt leben wie ihr Chef/ihre Chefin. Dabei soll von beiden (Mitarbeitende/r und Chef/in) je der Vorname (first_name) sowie einmalig die Stadt (city) ausgegeben werden. Die Ausgabereihenfolge richtet sich nach dem Geburtsdatum (birth_date) des/r Mitarbeitenden (aufsteigend).
Die Anweisung soll auch bei anderer Datenlage korrekt funktionieren.

Es wird das folgende Ergebnis erwartet:

FNAME_EMP	FNAME_MGR	city
Robert	Steven	London
Michael	Steven	London
Anne	Steven	London



Prüfungsfrage 6 SELF_JOIN

In der Tabelle **employees** gibt das Attribut **reports_to** an, wer Chef/in ist.

Erstelle eine SQL-Anweisung, die Mitarbeitende auflistet, die in der gleichen Stadt leben wie ihr Chef/ihre Chefin. Dabei soll von beiden (Mitarbeitende/r und Chef/in) je der Vorname (first_name) sowie einmalig die Stadt (city) ausgegeben werden. Die Ausgabereihenfolge richtet sich nach dem Geburtsdatum (birth_date) des/r Mitarbeitenden (aufsteigend).
Die Anweisung soll auch bei anderer Datenlage korrekt funktionieren.

Es wird das folgende Ergebnis erwartet:

FNAME_EMP	FNAME_MGR	city
Robert	Steven	London
Michael	Steven	London
Anne	Steven	London

```
SELECT FNAME_EMP, FNAME_MGR, city
FROM employees;

SELECT first_name AS FNAME_EMP,
  (SELECT first_name
   WHERE employee_id = reports_to) AS
  FNAME_MGR, city
FROM employees;

SELECT e.first_name as FNAME_EMP,
  mgr.first_name as FNAME_MGR, e.city
FROM employees e JOIN employees mgr
  ON e.employee_id = mgr.reports_to
WHERE e.city = mgr.city;
```

Prüfungsfrage 6 SELF_JOIN

In der Tabelle **employees** gibt das Attribut **reports_to** an, wer Chef/in ist.

Erstelle eine SQL-Anweisung, die Mitarbeitende auflistet, die in der gleichen Stadt leben wie ihr Chef/ihre Chefin. Dabei soll von beiden (Mitarbeitende/r und Chef/in) je der Vorname (first_name) sowie einmalig die Stadt (city) ausgegeben werden. Die Ausgabereihenfolge richtet sich nach dem Geburtsdatum (birth_date) des/r Mitarbeitenden (aufsteigend).
Die Anweisung soll auch bei anderer Datenlage korrekt funktionieren.

Es wird das folgende Ergebnis erwartet:

FNAME_EMP	FNAME_MGR	city
Robert	Steven	London
Michael	Steven	London
Anne	Steven	London

```
SELECT emp1.first_name as FNAME_EMP,
  emp2.first_name as FNAME_MGR, emp1.city
FROM employees emp1 LEFT JOIN employees emp2
  ON emp1.reports_to = emp2.employee_id
WHERE emp1.city = 'London' AND
  emp1.title not like '%Manager%'
ORDER BY emp1.birth_date ASC;

SELECT e.first_name as FNAME_EMP,
  m.first_name as FNAME_MGR, e.city
FROM employees e INNER JOIN employees m
WHERE e.reports_to = m.employee_id
  AND e.city = m.city order by e.birth_date;
```

Prüfungsfrage 6 SELF_JOIN

In der Tabelle **employees** gibt das Attribut **reports_to** an, wer Chef/in ist.

Erstelle eine SQL-Anweisung, die Mitarbeitende auflistet, die in der gleichen Stadt leben wie ihr Chef/ihre Chefin. Dabei soll von beiden (Mitarbeitende/r und Chef/in) je der Vorname (first_name) sowie einmalig die Stadt (city) ausgegeben werden. Die Ausgabereihenfolge richtet sich nach dem Geburtsdatum (birth_date) des/r Mitarbeitenden (aufsteigend).
Die Anweisung soll auch bei anderer Datenlage korrekt funktionieren.

Es wird das folgende Ergebnis erwartet:

FNAME_EMP	FNAME_MGR	city
Robert	Steven	London
Michael	Steven	London
Anne	Steven	London

```
SELECT e.first_name as FNAME_EMP,
  m.first_name as FNAME_MGR, e.city
FROM employees e INNER JOIN employees m
  ON e.reports_to = m.employee_id
WHERE e.city = m.city order by e.birth_date
ORDER BY e.birth_date;
```

Eine kleine Übung zu GROUP BY

Eine kleine Übung

Sie erhalten Legos in diversen Farben und Formen

Beantworten Sie folgende Fragen:

- Wie viele Legos haben Sie je Farbe?
- Wie viele Legos haben Sie je Form?
- Wie viele Legos haben Sie je Form und je Farbe?

Wie sind Sie vorgegangen?

Wie viele Legos haben Sie je …?

... Farbe:

```
SELECT farbe, count(*)
FROM lego
GROUP BY farbe;
```

... Form:

```
SELECT form, count(*)
FROM lego
GROUP BY form;
```

Wie viele Legos haben Sie je Form und je Farbe?

```
SELECT form, farbe, count(*)
FROM lego
GROUP BY form, farbe;
```

Von welchen Legos gibt es weniger als 10 je Form und Farbe?

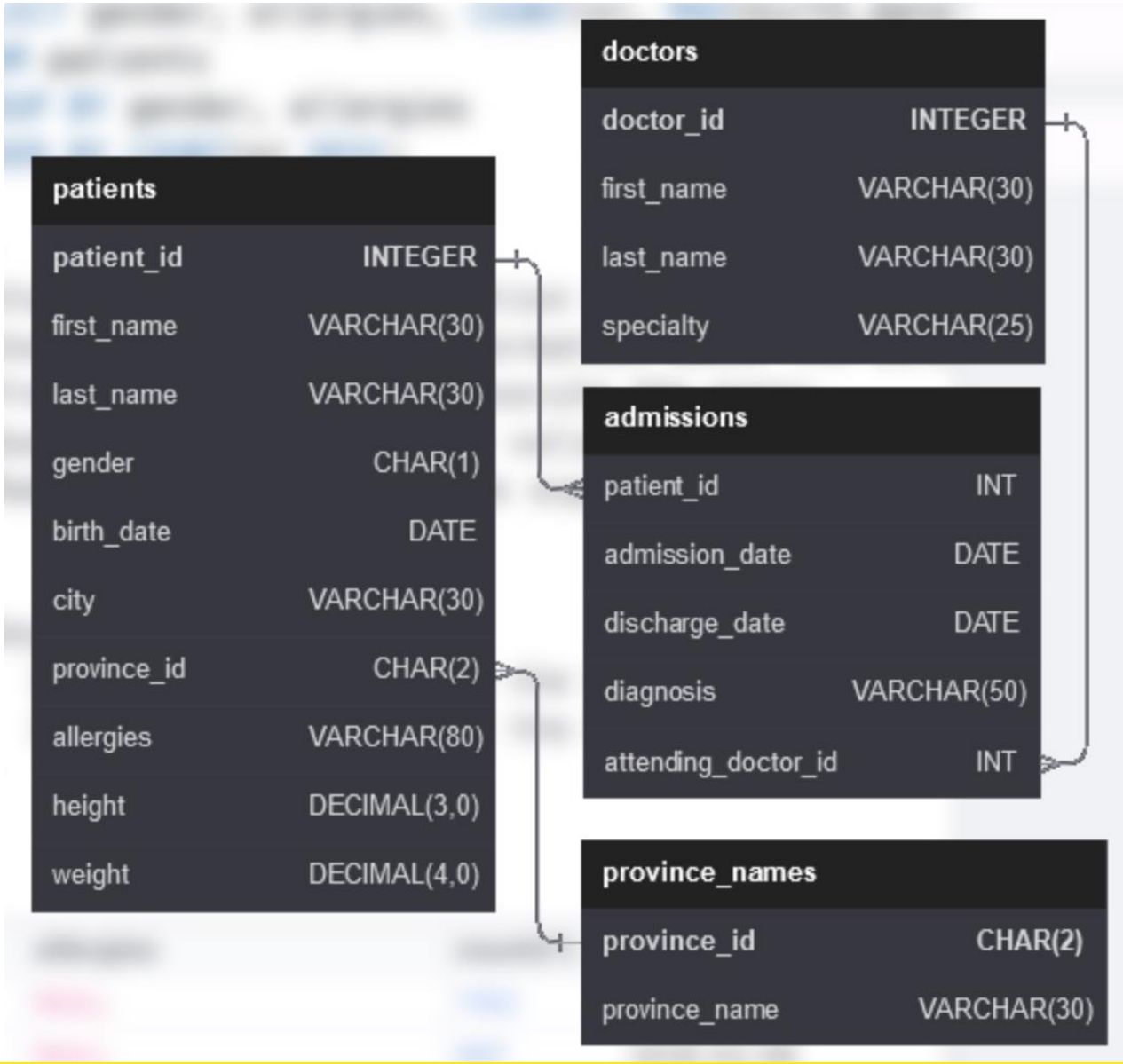
```
SELECT form, farbe, count(*)
FROM lego
GROUP BY form, farbe
HAVING count(*) < 10;
```

Prüfungsfrage 2 SELECT

Formuliere eine SQL-Abfrage, die folgende Anforderung erfüllt. Die Daten zu Patienten sollen so ausgewertet werden, dass man je Geschlecht (gender) sieht, welche Allergie (allergies) wie oft vorkommt und wie jung die jüngste betroffene Person ist. Die Liste soll nach Häufigkeit absteigend sortiert werden.
Die Anweisung soll auch für eine andere Datenlage korrekt sein.

Es wird das folgende Ergebnis erwartet (nur Ausschnitt):

gender	allergies	count(*)	max(birth_date)
M	NULL	1162	2018-07-21
F	NULL	897	2018-02-06
M	Penicillin	555	2018-07-09



Prüfungsfrage 2 SELECT

Formuliere eine SQL-Abfrage, die folgende Anforderung erfüllt. Die Daten zu Patienten sollen so ausgewertet werden, dass man je Geschlecht (gender) sieht, welche Allergie (allergies) wie oft vorkommt und wie jung die jüngste betroffene Person ist. Die Liste soll nach Häufigkeit absteigend sortiert werden.
Die Anweisung soll auch für eine andere Datenlage korrekt sein.

Es wird das folgende Ergebnis erwartet (nur Ausschnitt):

gender	allergies	count(*)	max(birth_date)
M	NULL	1162	2018-07-21
F	NULL	897	2018-02-06
M	Penicillin	555	2018-07-09

```
SELECT gender, allergies, COUNT(*) AS 'count(*)',
birth_date AS 'max(birth_date)'
FROM patients
ORDER By birth_date desc;

SELECT gender, allergies, birth_date
FROM patients
order by birth_date desc

SELECT p.gender, p.allergies, count(*),
max(p.birth_date)
FROM admissions ad
JOIN patients p ON p.patient_id = ad.patient_id
GROUP BY p.patient_id
ODER BY count(*) DESC;

SELECT gender, allergies, count(*), max(birth_date)
from patients
GROUP BY gender, allergies
ORDER BY count(*) DESC;
```

Prüfungsfrage 2 SELECT

Formuliere eine SQL-Abfrage, die folgende Anforderung erfüllt. Die Daten zu Patienten sollen so ausgewertet werden, dass man je Geschlecht (gender) sieht, welche Allergie (allergies) wie oft vorkommt und wie jung die jüngste betroffene Person ist. Die Liste soll nach Häufigkeit absteigend sortiert werden.
Die Anweisung soll auch für eine andere Datenlage korrekt sein.

Es wird das folgende Ergebnis erwartet (nur Ausschnitt):

gender	allergies	count(*)	max(birth_date)
M	NULL	1162	2018-07-21
F	NULL	897	2018-02-06
M	Penicillin	555	2018-07-09

```
SELECT gender, allergies, COUNT(*) AS 'count(*)',
birth_date AS 'max(birth_date)'
FROM patients
ORDER By birth_date desc;
```

```
SELECT gender, allergies, birth_date
FROM patients
order by birth_date desc
```

```
SELECT p.gender, p.allergies, count(*),
max(p.birth_date)
FROM admissions ad
JOIN patients p ON p.patient_id = ad.patient_id
GROUP BY p.patient_id
ODER BY count(*) DESC;
```

```
SELECT gender, allergies, count(*), max(birth_date)
from patients
GROUP BY gender, allergies
ORDER BY count(*) DESC;
```

Das Spiel SCROLLS

https://github.com/fhnw-sql/FHNW-SQLScrolls

Dr. Andrea Kennel

Consultant

Dozentin für Datenbanken

Coach für Project Management

Fachhochschule Nordwestschweiz

Brugg/Windisch, Schweiz

andrea.kennel@fhnw.ch
andrea@infokennel.ch
www.infokennel.ch