SQL - deel 2 (joins)

We gaan door met de student pagina uit deze les.

ERD

Deze database heeft drie tabellen.

image-1676378321011.png

We gaan kijken naar de koppel tabel. In de database heet deze tabel cijfer.

image-1676378544439.png

Tabel cijfer

We gaan nu alle gegevens selecteren uit de cijfertabel waarbij de student een 10 gehaald heeft.

Let op dat de cijfers in de database staan als getal tussen 0 en 100. 50 betekent een 5, 85 betekent een 8.5 en een 100 betekent dus een 10.

Voer de volgende query uit om alle 10's uit de cijfers tabel te selecteren.

```
SELECT *
FROM cijfer
WHERE cijfer = 100
```

Als het goed is krijg je 10 regels terug.

Op één van deze regels staat het volgende:

image-1676395284323.png

We zien hier dus dat student met id=10 een cijfer 100 (dat is dus een 10) heeft gehaald voor vak met id=2.

Laten we eens kijken welke student dit is.

```
SELECT *
FROM student
WHERE id = 10
```

En voor welk vak is dit?

```
SELECT *
FROM vak
WHERE id = 2
```

INNER JOIN

Wat we nu met deze drie queries hebben gedaan is het samenvoegen van de informatie van de drie tabellen.

Het *student_id* en *vak_id* uit de *cijfer* tabel zijn FK's (foreign keys) en die verwijzen naar de PK's (primary keys) van de gekoppelde tabellen *student* en *vak*.

Laten we de tabellen automatisch gaan combineren. Latne we eerst de cijfer tabel met de student tabel combineren.

```
SELECT *
FROM cijfer
INNER JOIN student ON student.id = cijfer.student_id
WHERE cijfer = 100
```

We hebben verschillende soorten JOIN maar standaard gebruiken we een INNER JOIN. Later wordt nog uitgelegd welke andere joins er zijn.

We maken dus een verbinding met de tabel student door het commando INNER JOIN. Dan moeten we nog wel vertellen hoe de tabellen zijn gekoppeld, wat is de FK en welke PK hoort daarbij? Dat zetten we achter de ON.

Zie je dat er *student.id* staat, dat is de kolom *id* uit de tabel *student* en dat is de PK van deze tabel. Deze koppelen we aan de FK uit de *cijfer* tabel, *cijfer.student_id*.

Je ziet nu dus ook dat als er meer tabellen zijn het gebruikelijk is om de tabelnaam voor de kolomnaam te zetten. Dit voorkomt verwarring (achter de WHERE zou je ook *cijfer.cijfer* mogen schrijven).

We gaan nu de tabel *vak* ook koppelen aan *cijfer* tabel.

```
SELECT *
FROM cijfer
INNER JOIN student on student.id = cijfer.student_id
INNER JOIN vak on vak.id = cijfer.vak_id
WHERE cijfer = 100
```

We zien nu alle velden uit alle tabellen, dat is wat onoverzichtelijk. Laten we alleen de naam, vak en cijfer afdrukken en laten we gelijk nette aliassen gebruiken. Die aliassen zijn in het vorige deel uitgelegd.

```
SELECT student.voornaam 'Voornaam',
  student.achternaam 'Achternaam',
        vak.naam 'naam',
        round(cijfer.cijfer/10,1) 'cijfer'
FROM cijfer
INNER JOIN student on student.id = cijfer.student_id
INNER JOIN vak on vak.id = cijfer.vak_id
WHERE cijfer = 100
```

Op regel 4 delen we het cijfer door 10 (cijfer.cijfer/10) en ronden af op één decimaal met de *round* functie. Dit werkt ongeveer hetzelfde als in PHP.

En bij welk vak zijn de meeste 10-en gevallen?

Soorten JOINS

Twee tabellen

Student				**Cijfer**		
ID (**PK**)	voornaam	achternaam		ID (PK)	student_ID (**FK**)	cijfer
1	Paddie	Durban		1	1	5
2	Rosemonde	Surr		2	2	6
4	Kenyon	Russilll		3	4	9
6	Shaylyn	Bilbrey		4	12	7
7	Terrijo	Anear		5	13	8
				6	14	7

(INNER) JOIN

Resultaat is alleen de rijen die een match hebben, dus waarbij de FK en PK hetzelfde is.

Paddie	Durban	5
Rosemonde	Surr	6

Kenyon	Russillll	9
Shaylyn	Bilbrey	7

LEFT (OUTER) JOIN

Resultaat is alle rijen uit de 'linker' tabel. De linker table is de tabel die direct na de WHERE staat.

Paddie	Durban	5
Rosemonde	Surr	6
Kenyon	Russillll	*NULL*
Shaylyn	Bilbrey	7
Terrijo	Anear	*NULL*

RIGHT (OUTER) JOIN

Resultaat is alle rijen uit de 'rechter' tabel. De rechter table is de tabel die in de JOIN staat.

Paddie	Durban	5
Rosemonde	Surr	6
Kenyon	Russillll	9
NULL	*NULL*	7
NULL	*NULL*	8
NULL	*NULL*	7

Revision #13
Created 2023-02-14 16:46:50 UTC by Max
Updated 2023-02-18 13:54:27 UTC by Max