

HOOFDSTUK 9: MEERDERE TABELLEN

Leerdoelen

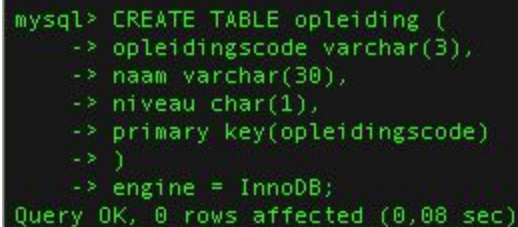
Na het bestuderen van dit hoofdstuk wordt van je verwacht dat je:

- weet wat een primaire sleutel (primary key) is
- weet wat een vreemde sleutel (foreign key) is
- weet wat een relatie is

Een tweede tabel

In de database is nog niet opgenomen aan welke opleidingen de studenten deelnemen. Om dat te kunnen doen moeten we eerst de opleidingen in de database opnemen. Dat doen we door een tweede tabel te maken met daarin alle opleidingen:

```
CREATE TABLE opleiding (  
    opleidingscode varchar(3),  
    naam varchar(30),  
    niveau char(1),  
    primary key(opleidingscode)  
)  
engine = InnoDB;
```



```
mysql> CREATE TABLE opleiding (  
-> opleidingscode varchar(3),  
-> naam varchar(30),  
-> niveau char(1),  
-> primary key(opleidingscode)  
-> )  
-> engine = InnoDB;  
Query OK, 0 rows affected (0,08 sec)
```

Opdracht 9.1

Voer de query hierboven uit zodat je een tabel `opleiding` in je database hebt.

Primaire sleutels

De meeste regels hierboven zijn duidelijk (anders: zie hoofdstuk 2). Alleen de regel met `primary key` is nog niet behandeld.

Wat je in het Engels de primary key noemt, heet in het Nederlands de primaire sleutel. In dit geval is de kolom `opleidingscode` de primaire sleutel van de tabel. Met zo'n sleutel kan de

database een record (rij) in de tabel uniek identificeren, net zoals je een boek kunt herkennen aan het ISBN of een persoon kunt identificeren aan de hand van zijn of haar BSN. Waarom is het belangrijk om een primary key te hebben? Bijvoorbeeld als je één specifieke record wilt verwijderen. Dan is het fijn dat er minstens één kolom is waarvan je weet dat elke record een unieke waarde bevat, zodat je specifiek die record kan aanwijzen die je bedoelt.

Let op: een tweede reden om altijd gebruik te maken van sleutels is voor het leggen van relaties tussen tabellen, maar daarover lees je straks meer.

Opdracht 9.1

Voer de volgende query in:

```
INSERT INTO opleiding (opleidingscode, naam, niveau)
VALUES ("A0", "ApplicatieOntwikkeling", "4");
```

Deze query gaat goed. Voer nu de volgende query in:

```
INSERT INTO opleiding (opleidingscode, naam, niveau)
VALUES ("A0", "ApothekersOpleiding", "4");
```

We krijgen dan de volgende foutmelding:

```
ERROR 1062 (23000): Duplicate entry 'A0' for key 'PRIMARY'
```

Met andere woorden: de database accepteert geen tweede record met dezelfde sleutel. Wel accepteert de database de volgende query, omdat de sleutel een andere waarde heeft.

Opdracht 9.2

Voer de volgende query in:

```
INSERT INTO opleiding (opleidingscode, naam, niveau)
VALUES ("MBI", "Medewerker Beheer ICT", "3");
```

Opdracht 9.3

Vul de tabel aan met de volgende data:

- MV, MediaVormgeving, 4
- DT3, DeskTopPublising, 3
- DT2, DeskTopPublising, 2
- IB, ICT Beheerder, 4

Vreemde sleutels

We moeten een manier vinden om in de database vast te leggen welke student welke opleiding of opleidingen volgt. Daarvoor kunnen we het veld **opleidingscode** toevoegen aan de tabel **student**. In dat veld zetten we vervolgens de primaire sleutel van de opleiding die de student volgt. Zo'n sleutel die verwijst naar gegevens in een andere tabel noemen we een vreemde sleutel, of in het Engels: foreign key.

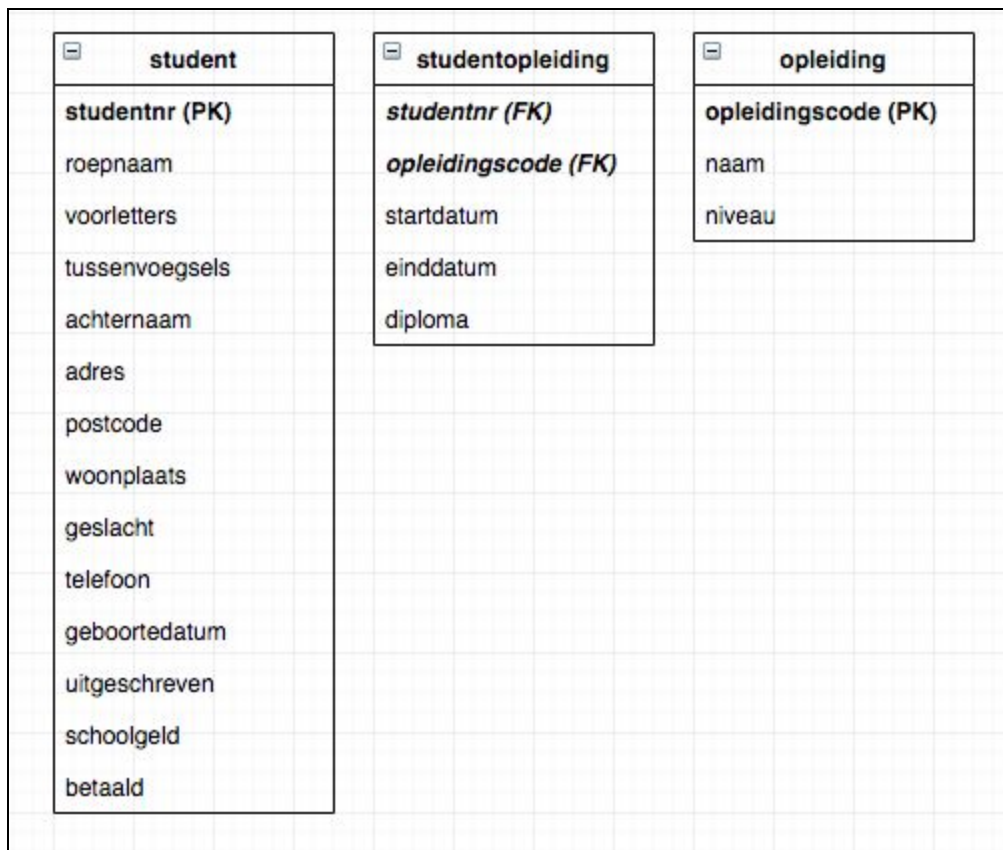
student	opleiding
studentnr (PK)	opleidingscode (PK)
roepnaam	naam
voorletters	niveau
tussenvoegsels	
achternaam	
adres	
postcode	
woonplaats	
geslacht	
telefoon	
geboortedatum	
uitgeschreven	
schoolgeld	
betaald	
opleidingscode (FK)	

In deze simpele opzet komen we wel een paar praktische problemen tegen. Voorbeelden:

- Als de student van opleiding verandert, kunnen we dat niet vastleggen zonder te verliezen wat hij eerder studeerde, omdat het veld slechts één opleiding kan bevatten.
- Omdat er maar één veld beschikbaar is, kan een student ook altijd maar één opleiding volgen.
- We kunnen ook geen gegevens vastleggen die zowel met de student als met de opleiding te maken hebben, bijvoorbeeld: wanneer is een student met een opleiding begonnen?

Tussentabellen

Voor bovenstaande problemen hanteren we de volgende oplossing: we maken *nog* een tabel en die noemen we **studentopleiding**. In die tabel gebruiken we twee velden. Het eerste veld is voor de sleutel van de student en het tweede veld is voor de sleutel van de opleiding. Deze sleutels zijn weer vreemde sleutels (foreign keys) omdat deze sleutels afkomstig zijn uit andere tabellen waar ze als primaire sleutels (primary keys) functioneren. Verder maken we nog twee velden aan met de namen **startdatum** en **einddatum** (wanneer is de student begonnen met de opleiding en wanneer is hij gestopt met de opleiding). Tot slot maken we een veld aan met de naam **diploma**, waarin staat of de opleiding is afgesloten met een diploma of niet.



Een dergelijke tabel, die bedoeld is om twee andere tabellen aan elkaar te koppelen, noemen we een tussentabel of koppeltabel. Je weet dat je een koppeltabel nodig hebt als er een zogenaamde veel-op-veel relatie bestaat. In dit geval:

- Een student kan meerdere opleidingen volgen
- Een opleiding kan door meerdere studenten gevolgd worden

Opdracht 9.4

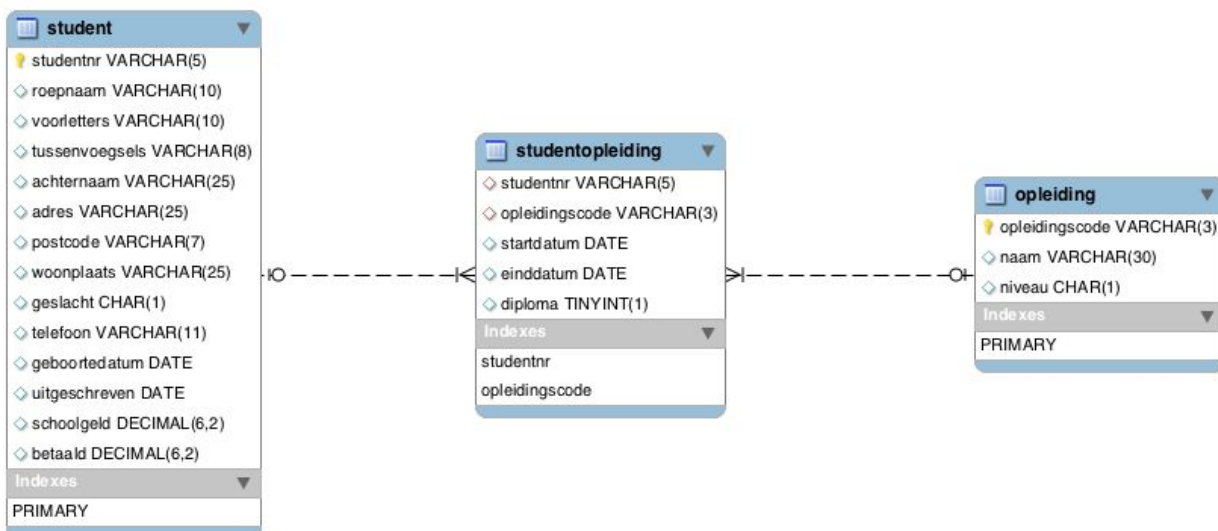
Maak de tussentabel aan met de volgende query:

```
CREATE TABLE studentopleiding (  
    studentnr varchar(5),  
    opleidingscode varchar(3),  
    startdatum date,  
    einddatum date,  
    diploma tinyint(1) default 0,  
    CONSTRAINT FOREIGN KEY(studentnr) REFERENCES student(studentnr)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
    CONSTRAINT FOREIGN KEY(opleidingscode) REFERENCES  
opleiding(opleidingscode)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION  
)  
engine = InnoDB;
```

Toelichting op bovenstaande query

- `tinyint(1)` is de database-variant van een boolean zoals je die bijvoorbeeld kent uit Java. Dat betekent dat het veld diploma alleen de waarde 0 (false) of 1 (true) kan hebben.
- `default 0` betekent dat de beginwaarde altijd 0 (false) is, omdat we ervan uit gaan dat een student nog geen diploma van de opleiding heeft als hij aan de opleiding begint.

De structuur van de database ziet er nu als volgt uit:



Samengestelde sleutels

Je ziet hierboven drie tabellen: `student`, `studentopleiding` en `opleiding`. Daarin staan de velden die in de tabel gebruikt worden. Aan het sleuteltje voor een veld kun je de primary key herkennen. Bij `studentopleiding` worden de twee vreemde sleutels onder indexes genoemd. Deze twee vreemde sleutels worden ook wel een samengestelde sleutel genoemd, omdat je ze moet combineren om tot een unieke combinatie te komen.

Relaties

De relaties tussen de tabellen worden in de afbeelding weergegeven door stippellijnen met aan het einde verschillende symbooltjes die aangeven hoeveel van de ene tabel horen bij hoeveel van de andere tabel. Voorbeeld:

De relatie tussen `student` en `studentopleiding` is een één-op-veel relatie, omdat één student meerdere records kan vullen in `studentopleiding` (hij of zij kan meerdere opleidingen volgen), maar andersom kan een record in `studentopleiding` altijd maar gaan over één student.

Hetzelfde geldt voor de relatie tussen `opleiding` en `studentopleiding`. Eén opleiding kan voorkomen in meerdere records van `studentopleiding` (de opleiding wordt dan gedaan door meerdere studenten), maar andersom kan één record van `studentopleiding` altijd maar gaan over één opleiding.

De mogelijke symbooltjes aan het eind van de stippellijnen staan hieronder in een tabel:

<u>is assigned</u>	is toegekend aan
	komt nul, een of meerdere keren voor
	komt een of meerdere keren voor
	komt precies een keer voor
	komt nul of een keer voor

Opdracht 9.5

Zorg ervoor dat de tabel studentopleiding de volgende inhoud heeft (de einddatum blijft leeg):

- 1111, AO, 1-8-2015
- 2222, MV, 1-8-2015
- 3333, DT3, 1-8-2015
- 4444, DT2, 1-8-2015
- 5555, MBI, 1-8-2015
- 6666, IB, 1-8-2015

INNER JOIN

Als je nu wilt opzoeken wie welke opleiding volgt, dan doe je dat als volgt:

```
SELECT roepnaam, tussenvoegsels, achternaam, opleiding.naam
FROM student
INNER JOIN studentopleiding
ON studentopleiding.studentnr = student.studentnr
INNER JOIN opleiding
ON opleiding.opleidingscode = studentopleiding.opleidingscode;
```

- Achter **SELECT** staan de namen van velden die je wilt zien
- Achter **FROM** staat de naam van de tabel waarvan de meeste velden achter **SELECT** staan
- Daarachter wordt met **INNER JOIN** de twee relaties aangegeven.
- Eerst staat er de naam van een tabel die nog niet is genoemd en achter **ON** staat vervolgens de relatie. Welke primaire sleutel uit één tabel komt overeen met welke vreemde sleutel uit de andere tabel.

Opdracht 9.6

Voer de bovenstaande query uit en zie wat het resultaat is.

OUTER JOIN

Aan de naam **INNER JOIN** kun je afleiden dat er ook een **OUTER JOIN** bestaat. Die bestaat inderdaad, maar heet in SQL een **LEFT JOIN** of een **RIGHT JOIN**. Bij een **INNER JOIN** zien we alleen de studenten die ook daadwerkelijk een opleiding volgen. Als we alle studenten willen zien, dus ook degenen die geen opleiding volgen, dan gebruiken we de **OUTER JOIN**:

```
SELECT studentnr, roepnaam, tussenvoegsels, achternaam, opleiding.naam
FROM student
LEFT JOIN studentopleiding
USING(studentnr)
LEFT JOIN opleiding
USING(opleidingscode);
```

Opdracht 9.7

Voer de bovenstaande query uit en zie wat het resultaat is. Zijn er ook studenten die nog geen opleiding volgen?

Opdracht 9.8

Welke opleiding volgt Berend van der Tol? Geef de query waarmee je specifiek dat gegeven opvraagt en niet de query waarmee je het complete overzicht krijgt (hint: gebruik [WHERE](#)).

Opdracht 9.9

Wie volgt/volgen de opleiding ApplicatieOntwikkeling? Geef de query waarmee je specifiek dat gegeven opvraagt en niet de query waarmee je het complete overzicht krijgt.

Opdracht 9.10

Welke studenten volgen een opleiding op niveau 4? Geef de query waarmee je specifiek dat gegeven opvraagt en niet de query waarmee je het complete overzicht krijgt.

Opdracht 9.11

Maak een lijst van de niveaus met daarachter het aantal studenten die een opleiding op dat niveau volgen.

Vaktaal

- **Primaire sleutel (primary key):** het veld waaraan een record in de tabel uniek wordt herkend.
- **Vreemde sleutel (foreign key):** een veld waarin een sleutel afkomstig uit een andere tabel staat, ter verwijzing naar de inhoud van het record in die andere tabel.
- **Samengestelde sleutel:** twee of meerdere velden in een tabel die in combinatie uniek zijn.
- **Relatie:** De verwijzing van de ene naar de andere tabel.