Les S-01: De basisbeginselen van SQL

1.0 Relationele databases en SQL

Een **database** is een bestand waarin gegevens worden opgeslagen in de vorm van **tabellen**. Zo kan een huisarts met behulp van een database een administratie bijhouden met daarin zijn patiënten, hun bezoeken, hun medicatie en hun betalingen. Op een vergelijkbare manier kunnen een webwinkel de klantenadministratie en een vereniging de ledenadministratie bijhouden.
In de tabel met patiënten is elke **rij** een verzameling gegevens van een patiënt (**record**) en elke **kolom** een eigenschap van de patiënten (**veld**). Hieronder is ter illustratie (een deel van) de tabel met patiënten weergeven.

patiëntnr	voorletters	tussen	achternaam	adres	postcode	plaats	telefoonnr
1	J.P.	van	Dijk	Hoofdweg 23	3067 GH	Rotterdam	010-2345678
2	E.G.		Eland	Nystadstraat 5	3067 TX	Rotterdam	010-8765432
3	G.		Ferdinandus	Evenaar 13	3067 FJ	Rotterdam	010-2002002

Een database bestaat dus uit tabellen.
Tabellen bestaan uit records.
Records bestaan uit velden.

Elk record moet op een unieke manier geïdentificeerd kunnen worden. Persoonlijke gegevens zijn vaak niet uniek. Er kunnen meerdere mensen dezelfde achternaam hebben, op hetzelfde adres wonen of dezelfde geboortedatum hebben. Vandaar dat je in tabellen vaak een **sleutelveld** aantreft: in het voorbeeld hierboven is dat het patiëntnummer. In de tabel met patiënten wordt iedere patiënt uniek aangegeven met behulp van het sleutelveld patiëntnummer. Met behulp van dit veld kan een patiënt dus altijd worden gevonden. We noemen het veld patiëntnummer een **primaire sleutel** omdat een patiënt primair wordt bepaald door zijn/haar nummer.

In de tabel met behandelingen wordt bijgehouden op welk moment een bepaalde patiënt wordt behandeld:

bnr	Datum	tijd	Patiëntnr	klacht	diagnose	behandeling/medicatie
1	12-9-2008	9:00	1	oorpijn	ontsteking	antibiotica
2	12-9-2008	9:15	156	maagpijn	maagzweer	rust + controle
3	12-9-2008	9:30	27	keelpijn	onstoken amandelen	antibiotica
4	12-9-2008	9:45	47	controle	goed herstel beenbreuk	geen
5	13-9-2008	9:00	27	keelpijn	onstoken amandelen	verwijzing naar ziekenhuis

Ook in deze tabel komt het sleutelveld patiëntnummer voor, alleen is in deze tabel het voorkomen van een bepaalde patiënt niet uniek. Een patiënt kan immers meerdere keren behandeld worden. Een dergelijk sleutelveld wordt ook wel een **vreemde sleutel** genoemd.
De primaire sleutel van de tabel met behandelingen is het behandelingsnummer bnr.

Tenslotte wordt in de tabel facturen bijgehouden of de patiënten hun behandelingen ook betaald hebben:

bnr	bedrag	betaald	datum betaling
1	€ 45	nee	
2	€ 50	nee	
3	€ 40	nee	
4	€ 60	nee	

Via de sleutelvelden zijn de verschillende tabellen aan elkaar gekoppeld ofwel gerelateerd. We noemen een database waarvan de tabellen aan elkaar zijn gerelateerd ook wel een relationele database.

Het **relationele model** van deze database ziet er als volgt uit:

patiënten (patiëntnr, voorletters, tussen, achternaam, adres, postcode, plaats, telefoonnummer, geboortedatum)
behandelingen (bnr, datum, tijd, <patiëntnr>, klacht, behandeling)
betalingen (<bnr>, bedrag, betaald, datumbetaling)

In dit model van de relationele database worden primaire sleutels onderstreept weergegeven bnr en vreemde sleutels onderstreept tussen haken weergegeven <bnr>.

Waarom worden niet gewoon alle gegevens in één tabel gezet ? Dat zou tot gevolg hebben dat één patiënt meerdere keren in de tabel voor zou komen. Vooral bij wijzigingen in de database (bijvoorbeeld bij een adreswijziging) zou dat tot gevolg hebben dat gegevens op meerdere plekken moeten worden aangepast. Er zit dan namelijk heel wat op meerdere plaatsen opgeslagen, overtollige informatie in de database. We spreken in dit geval van redundantie.

In deze les leer je hoe je met behulp van **SQL** (**Structured Query Language**) bepaalde gegevens uit databases kunt halen of een bepaalde bewerking kunt uitvoeren.

1.1 SQL

De taal SQL, Structured Query Language, bestaat uit instructies waarmee je gegevens uit een database kunt selecteren, toevoegen of verwijderen. In deze les beperken we ons tot SQL instructies waarmee je gegevens uit een database selecteert.
In het algemeen levert een SQL instructie een tabel met geselecteerde gegevens op. We zeggen ook wel dat we met het uitvoeren van een SQL instructie de database bevragen. Een SQL instructie wordt ook wel een **query** genoemd: een vraag die je stelt aan een database.

De basisvorm van een SQL query ziet er als volgt uit:

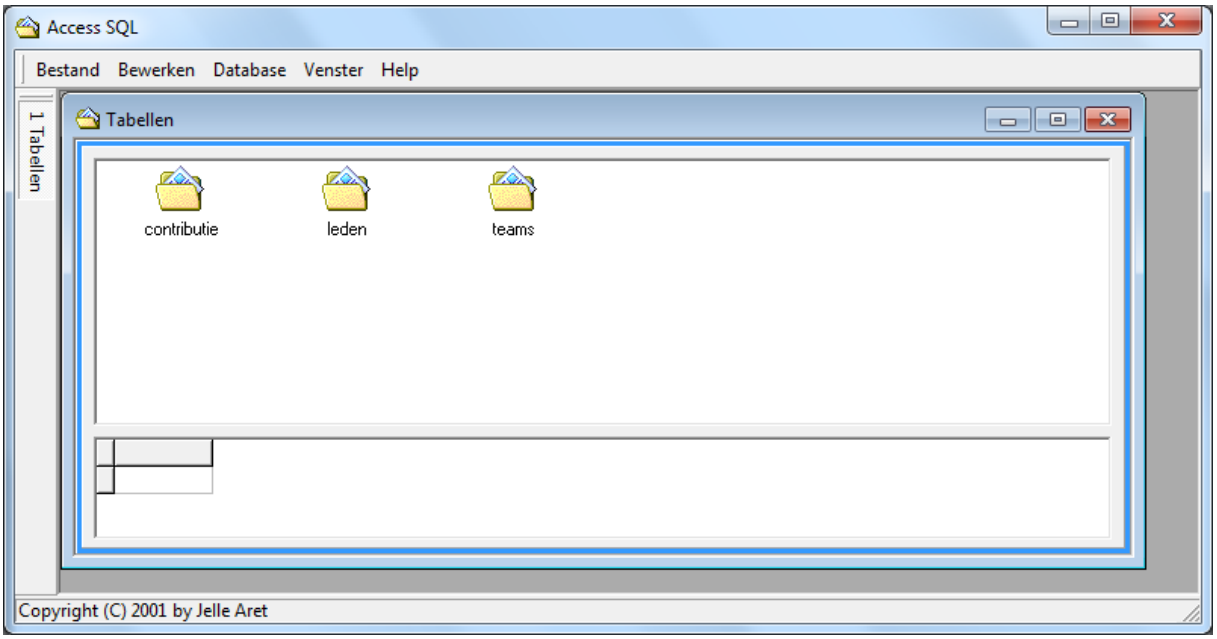
```
SELECT    selecteer de velden
FROM      uit de tabel
WHERE     waar de volgende voorwaarde geldt ;
```

Let op: een query eindigt altijd met een puntkomma ;

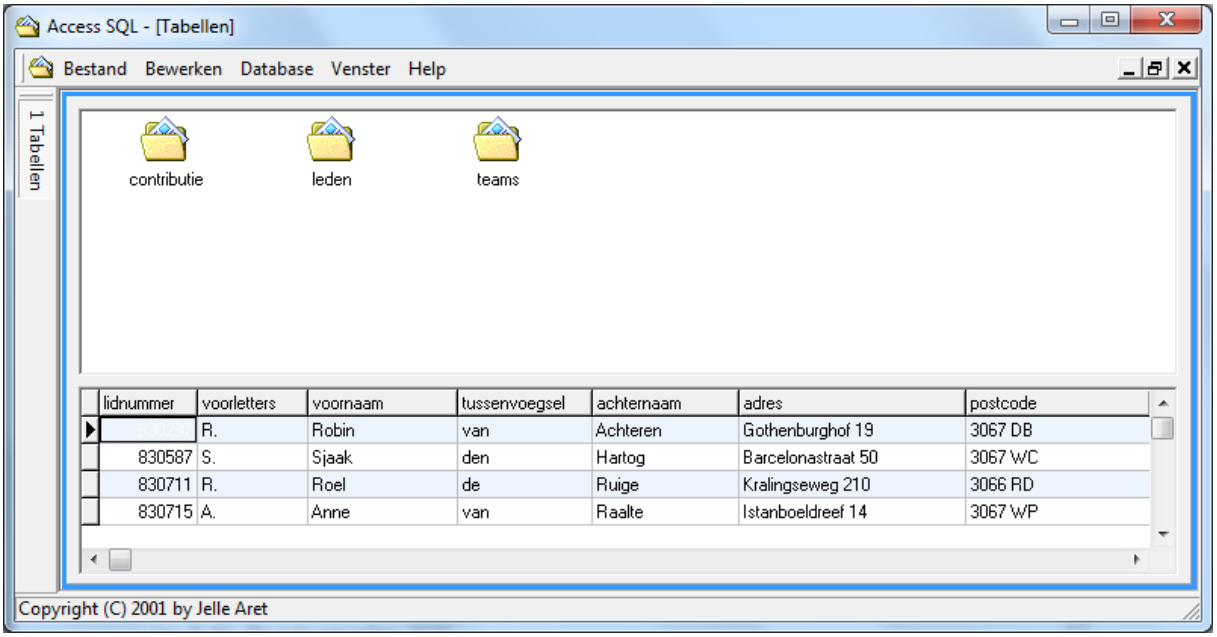
In deze lesbrief gebruiken we de database van de denkbeeldige volleybalvereniging “SET UP”. Deze relationele database bestaat uit drie tabellen.

Als je zelf niet over MS Access versie 2003 of 2010 beschikt kan je het programma AccSQL gebruiken. Dit programma bevat alleen de query editor van Access en is in die zin ook een stuk overzichtelijker in gebruik.

Open het programma AccSQL.
Kies voor **“Bestand”** en **“Open”** en open de database **“volleybalvereniging”.**
Kies voor **“Database”** en **“Tabellen ..”**



We zien dat de database van de volleybalvereniging uit de tabellen met leden, teams en contributies bestaat. Als je op de tabel **“Leden”** klikt komen daaronder de lidgegevens beschikbaar. Dat kan je natuurlijk ook met de andere tabellen doen.



Het relationele model bij de verenigingsdatabase is:

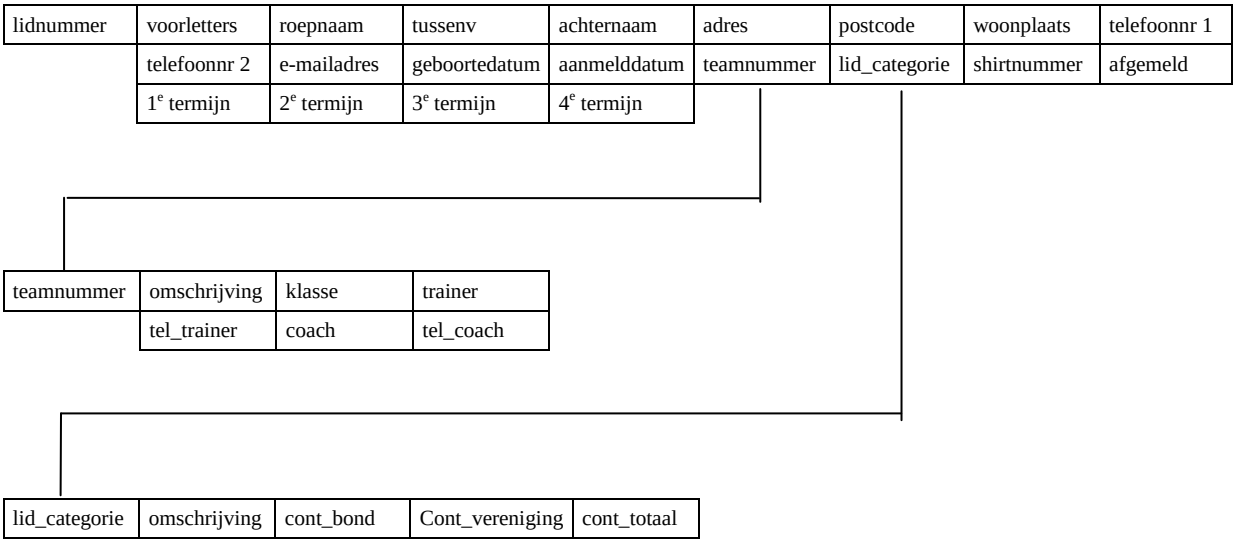
leden (lidnummer, voorletters, voornaam, tussenvoegsel, achternaam, adres, postcode, woonplaats, telefoonnummer, geboortedatum, aanvanglidmaatschap, <lidmaatschapscategorie>, <teamnummer>, shirtnummer, afgemeld, betaald1, betaald2, betaald3, betaald4)

teams (teamnummer, omschrijving, klasse, trainer, teltrainer, coach, telcoach)

contributie (lidmaatschapscategorie, omschrijving, bondscontributie, verenigingscontributie, totalecontributie)

Aan het relationele model kan je zien welke tabellen (**entiteiten**) worden onderscheiden en welke eigenschappen (**attributen**) in elke tabel worden bijgehouden. Ook kan je zien via welke eigenschappen de tabellen aan elkaar gekoppeld zijn (**relaties**). Bij een lid hoort een bepaald teamnummer, bv teamnummer H3, en bij dat teamnummer hoort een bepaalde klasse, trainer, etc. De primaire sleutels van de tabellen zijn het lidnummer, het teamnummer en de lidmaatschapscategorie.

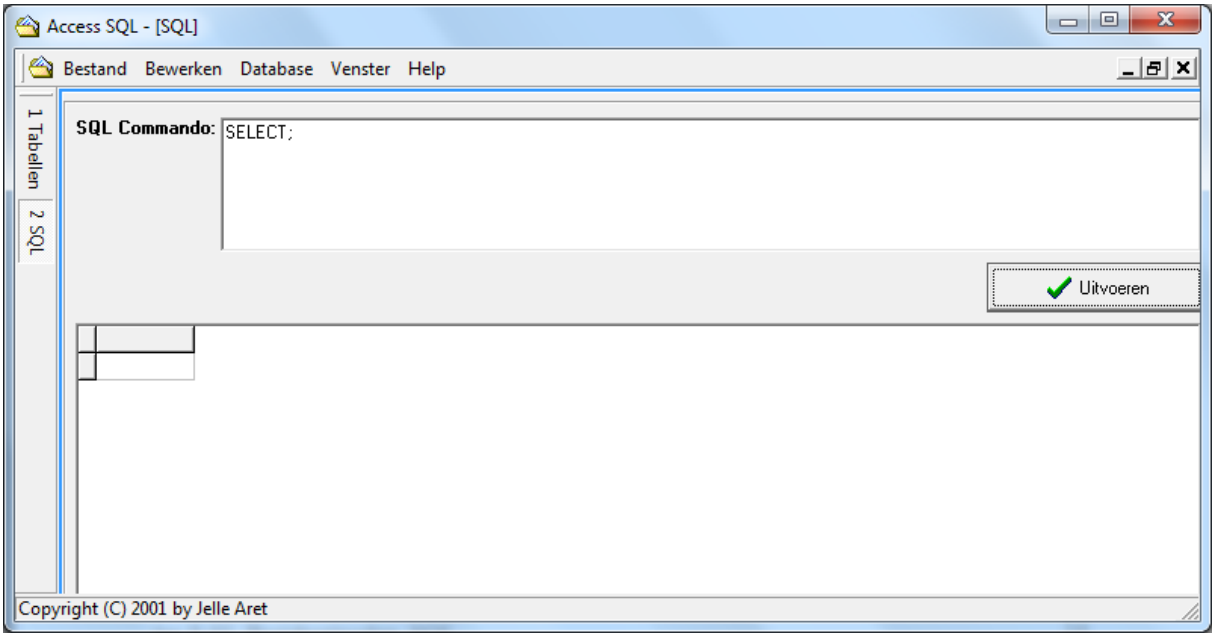
Dit model kan ook worden weergegeven als **strokendiagram**:



Dit strokendiagram geeft een iets duidelijker beeld hoe de tabellen aan elkaar zijn gekoppeld. Let op: in dit diagram zijn de veldnamen niet helemaal correct weergegeven.

Voorbeeld 1
We maken allereerst een query waarmee we alle namen van de spelers uit het eerste herenteam selecteren.

Kies nu voor **"Database"** en **"SQL"**.



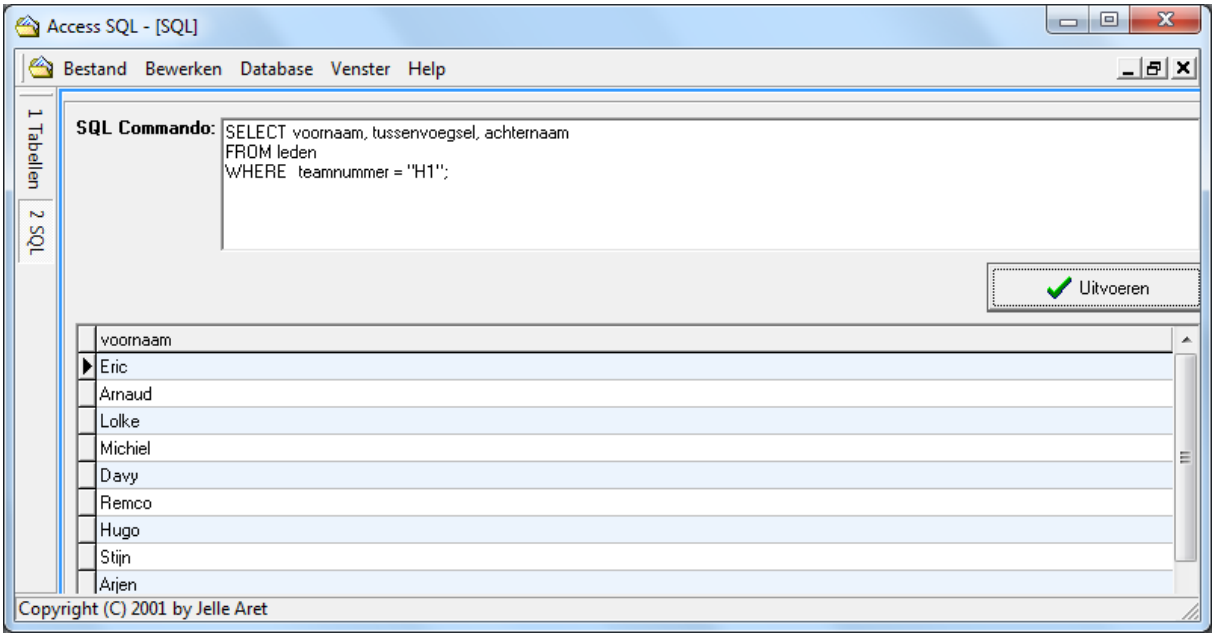
De basisvorm van een **SQL-opdracht** of **query** is:

SELECT	*selecteer de velden*
FROM	*uit de tabel*
WHERE	*waar de volgende voorwaarde geldt* ;

Met de query:

SELECT	voornaam, tussenvoegsel, achternaam
FROM	leden
WHERE	teamnummer = "H1";

selecteer je alle namen van de spelers van het eerste herenteam.



Voorbeeld 2
Met de query:

```
SELECT    *
FROM      leden
WHERE     teamnummer = “H1”;
```

selecteer je **alle gegevens** van de spelers van het eerste herenteam.

Voorbeeld 3
Met de query:

```
SELECT    lidmaatschapscategorie
FROM      contributie
WHERE     totalecontributie > 150;
```

selecteer je alle lidmaatschapscategorieën die meer dan 150 euro contributie moeten betalen.

Voorbeeld 4
Met de query:

```
SELECT    voornaam, tussenvoegsel, achternaam
FROM      leden
WHERE     geboortedatum < #01-01-1970#;
```

selecteer je alle leden die voor 1 januari 1970 zijn geboren.

Voorbeeld 5
Met de query:

```
SELECT    DISTINCT teamnummer
FROM      leden
```

selecteer je alle verschillende teamnummers die in de tabel leden worden onderscheiden.

Naar aanleiding van de voorbeelden 1 tot en met 4 merken we het volgende op:
De velden waarop je selecteert kunnen van een verschillend type zijn (tekst, getal, datum).
Afhankelijk van het gegevenstype dat bij een bepaald veld hoort gebruik je dus:

"……"	bij tekst
# ..... #	bij datum/tijd
niets	bij getallen

Daarnaast zijn er verschillende **operatoren** die je bij de selectie van gegevens kunt gebruiken:

=	is gelijk aan
<	is kleiner dan
>	is groter dan
<=	is kleiner of gelijk aan
>=	is groter of gelijk aan
<>	is ongelijk aan

Ook kan je in SQL op basis van meerdere voorwaarden informatie selecteren.
In SQL gebruik je daarvoor de **logische operatoren**:

AND	als aan een voorwaarde EN een andere voorwaarde voldaan moet worden
OR	als aan een voorwaarde OF een andere voorwaarde voldaan moet worden
NOT	als aan een voorwaarde NIET voldaan moet worden

Voorbeeld 6
Met de query:

```
SELECT    voornaam, tussenvoegsel, achternaam
FROM      leden
WHERE     teamnummer =”H1” OR teamnummer =”H2”;
```

selecteer je de heren die in heren 1 of heren 2 zitten.

Voorbeeld 7
Met de query:

```
SELECT    voornaam, tussenvoegsel, achternaam
FROM      leden
WHERE     teamnummer =”H1” AND NOT plaats = ”Rotterdam”;
```

selecteer je de heren die in heren 1 zitten en niet in Rotterdam wonen.

OPDRACHT

Opdracht 1.1
Wat is er fout aan de query:

```
SELECT    voornaam, tussenvoegsel, achternaam
FROM      leden
WHERE     teamnummer =”H1” OR “H2”;
```

Opdracht 1.2
Hoe kan de query:

```
SELECT    voornaam, tussenvoegsel, achternaam
FROM      leden
WHERE     teamnummer = ”H1” AND NOT plaats = ”Rotterdam”
```

ook zonder de logische operator NOT worden geschreven?

Opdracht 1.3
Schrijf een query die de naamgegevens en telefoonnummers van de speelsters van het tweede damesteam selecteert.

Opdracht 1.4
Schrijf een query die selecteert welke teams getraind worden door E. Berends.

Opdracht 1.5
Schrijf een query die selecteert welke leden in het jaar 1980 geboren zijn.

1.2 Functies

In SQL kan je ook een aantal **functies** gebruiken waarmee je berekeningen kunt uitvoeren. De belangrijkste functies zijn:

COUNT(veldnaam)	telt het aantal rijen (records) waarin een bepaald veld verschillende waarden aanneemt
COUNT(*)	telt het aantal rijen (records) in een tabel
SUM(veldnaam)	telt de waarden van een bepaald veld in een tabel bij elkaar op
AVG(veldnaam)	geeft de gemiddelde waarde van een bepaald veld in een tabel
MAX(veldnaam)	geeft de maximale waarde van een bepaald veld in een tabel
MIN(veldnaam)	geeft de minimale waarde van een bepaald veld in een tabel

Voorbeeld 8
Met de query:

```
SELECT    COUNT(*)
FROM      leden
WHERE     teamnummer=”H1” OR teamnummer=”H2”;
```

selecteer je hoeveel heren er in heren 1 of heren 2 zitten.

Voorbeeld 9
Met de query:

```
SELECT    COUNT(totalecontributie)
FROM      contributie
WHERE     totalecontributie=120;
```

tel je hoeveel lidmaatschapscategorieën er zijn met een totale contributie van 120 euro.

Groeperen met GROUP BY
Met query’s wil men nogal eens overzichten maken. Door gegevens te groeperen met de instructie GROUP BY kan je per groep een berekening uitvoeren.

Voorbeeld 10
Met de query:

```
SELECT    teamnummer, COUNT(*)
FROM      leden
GROUP BY  teamnummer;
```

krijg je een tabel waarin per team het aantal spellers wordt weergegeven.

Voorwaarden binnen een groep stellen met HAVING
In sommige gevallen wil je aan groepen ook nog een voorwaarde toevoegen.
Dat kan met de instructie HAVING.

Voorbeeld 11
Met de query:

```
SELECT    teamnummer, COUNT(*)
FROM      leden
GROUP BY  teamnummer
HAVING    COUNT(*) < 8;
```

krijg je een tabel waarin het aantal spelers staat van de teams met minder dan 8 spelers.

OPDRACHTEN

Opdracht 1.6
Schrijf een query die telt hoeveel leden er in Capelle a/d ijssel wonen.

Opdracht 1.7
Schrijf een query die per woonplaats aangeeft hoeveel leden er wonen.

ANTWOORDEN

Opdracht 1.1
Wat is er fout aan de query:

```
SELECT    voornaam, tussenvoegsel, achternaam
FROM      leden
WHERE     teamnummer=”H1” OR teamnummer=“H2”;
```

Opdracht 1.2
De query kan op de volgende manier:

```
SELECT    voornaam, tussenvoegsel, achternaam
FROM      leden
WHERE     teamnummer=”H1” AND woonplaats<>”Rotterdam”;
```

ook zonder de logische operator NOT worden

Opdracht 1.3
Schrijf een query die de naamgegevens en telefoonnummers van de speelsters van het tweede damesteam selecteert.

```
SELECT voornaam, tussenvoegsel, achternaam, telefoonnummer
FROM leden
WHERE teamnummer="D2";
```

Opdracht 1.4
Schrijf een query die selecteert welke teams getraind worden door E. Berends.

```
SELECT teamnummer
FROM teams
WHERE trainer=”E. Berends”;
```

Opdracht 1.5
Schrijf een query die selecteert welke leden in het jaar 1980 geboren zijn.

```
SELECT voornaam, tussenvoegsel, achternaam
FROM leden
WHERE geboortedatum>#31/12\1979# AND geboortedatum<#01/01/1981#;
```

Opdracht 1.6
Schrijf een query die telt hoeveel leden er in Capelle a/d ijssel wonen.

```
SELECT plaats,  COUNT(plaats)
FROM leden
WHERE plaats="Capelle a/d ijssel"
GROUP BY plaats;
```

Opdracht 1.7
Schrijf een query die per woonplaats aangeeft hoeveel leden er wonen.

```
SELECT plaats,  COUNT(plaats)
FROM leden
GROUP BY plaats;
```