

An abstract graphic featuring three blue circles of varying sizes, each composed of concentric rings. Two thin blue lines intersect at a point between the top and middle circles. A large blue circle is partially visible at the bottom right corner.

Databases Ontwerpen en Normaliseren

Ontwerpen van een database volgens de regels
van Boyce/Codd

Lic. André JONCKERS

2012-2013

1 Introductie

In deze cursus leer je een database ontwerpen. Databases zijn programma's die gegevens opslaan en beschikbaar maken. Veel programma's maken gebruik van een database. Je maakt zelf waarschijnlijk dagelijks gebruik van databases.

Als je geld opneemt bij de pinautomaat raadpleeg je de database van de bank, als je websites op internet bezoekt raadpleeg je een database waar (o.a.) paginateksten in opgeslagen zijn, als je een nieuw paspoort aanvraagt wordt ongetwijfeld een database met persoonsgegevens geraadpleegd, etc.

In deze cursus relationele database ontwerpen leer je zelf een database ontwerpen volgens het relationeel model.

2 Databasegeschiedenis

Vroeger, toen informatici nog stoffige jasjes en grote brillen droegen, werden gegevens opgeslagen in 'platte bestanden', in het Engels 'flat files'. Flat files zijn tekstbestanden waarin gegevens gescheiden door komma's of tabs zijn opgenomen. Met behulp van een programma kunnen deze gegevens uit het bestand gelezen worden. Het programma herkent de komma's of tabs (scheidingstekens) en kan zo gegevens uit het bestand selecteren.

```
'cursus_id', 'titel', 'categorie'  
'1', 'Cursus Access', 'Software'  
'2', 'Cursus Excel', 'Software'  
'3', 'Cursus Relationele Database Ontwerpen', 'Software'  
'4', 'Cursus Oracle DBA ', 'Software'  
'5', 'Cursus Raid Storage ', 'Hardware'  
'6', 'Een netwerk aanleggen', 'Netwerken'
```

Deze flat files doen hun werk, maar erg efficiënt zijn ze niet als ontwerp voor een database.

Om bijvoorbeeld de 'Cursus Relationele Database Ontwerpen' te selecteren in bovenstaande flat file moet het programma alle regels stuk voor stuk afzoeken totdat de cursus gevonden is.

cursus_id	titel	categorie
1	Cursus Access	Software
2	Cursus Excel	Software
3	Cursus Relationele Database Ontwerpen	Software
4	Cursus Oracle DBA	Software
5	Cursus Raid Storage	Hardware
6	Een netwerk aanleggen	Netwerken

Door in deze tabel naar een cursus_id te zoeken in de cursus_id-kolom kunnen snel de bijhorende titel en categorie opgevraagd worden. Dat gaat veel sneller dan door een tekstbestand heen wandelen totdat je de gewenste informatie gevonden hebt. Bij een relationele database kunnen gegevens uit specifieke rijen, kolommen en zelfs uit verschillende tabellen gecombineerd opgevraagd worden.

Je kunt het cursus_id (de 'primaire sleutel') gebruiken om de cursus te koppelen aan gegevens in andere tabellen. Hierover later meer.

Het concept van de relationele database waarin gegevens zijn geordend in tabellen is bedacht in de jaren 70 door **Ted Codd**. De relationele database is een zeer krachtige methode voor het opslaan van gegevens gebleken.

Tegenwoordig gebruiken heel veel programma's een relationele database om gegevens in op te slaan.

Tegenwoordig zijn er tal van verschillende **Relationele Database Management Systemen (RDBMS)** in gebruik.

Tot de toppers van de database-industrie behoren:

- [Oracle](#). Oracle wordt in de regel gebruikt voor grotere professionele applicaties
- [Microsoft SQL server](#). Het professioneel RDBMS van Microsoft wordt gebruikt voor alle soorten applicaties, van studentenapplicaties tot professionele applicaties met veel gebruikers. MSSQL is alleen beschikbaar voor Windows.
- [Mysql](#) community en onder zowel hobbyisten en beginners als professionelen wereldwijd. Mysql is gratis (of beter gezegd 'vrij') verkrijgbaar.
- [IBM](#) doet ook een grote duit in het zakje met verschillende databasesystemen, waarvan DB2 de populairste is.
- [Microsoft Access](#) is een uitgetekend RDBMS. Het mist sommige functies van een professioneel RDBMS en is bedoeld voor kantoor- en thuisgebruik.

Een databaseontwerp maak je niet voor een bepaald RDBMS. Het ontwerp van de database is onafhankelijk van het databasesysteem, zolang je een relationeel databasesysteem (RDBMS) gebruikt.

Je zou de database die we in deze cursus ontwerpen op elk van bovengenoemde databasesystemen kunnen maken.

3 Voordelen en mogelijkheden van relationele databases

Naast het snel opzoeken van informatie heeft een relationele database nog een aantal voordelen

- Een relationele database kan gegevens in verschillende tabellen relateren door het gebruik van sleutels. Daardoor kunnen gerelateerde gegevens uit verschillende tabellen tegelijk opgevraagd worden.

- Efficiënte opslag van gegevens. Gegevens worden maar één keer en op één plek zijn opgeslagen. Dit zorgt ervoor dat een wijziging of verwijdering van informatie altijd maar op één plek hoeft te gebeuren.
- Met een relationele database kan je regels opstellen voor het soort gegevens dat in een veld opgeslagen kan worden. Zo kun je onder andere datumvelden, tekstvelden en numerieke velden aanmaken.
- Integriteit van gegevens. Door het formuleren van de juiste relaties van tabellen kun je de integriteit (correctheid, betrouwbaarheid) van de gegevens die zijn opgeslagen beter garanderen.
- De meeste relationele databasesystemen kennen een rechtenstructuur, waarmee aan verschillende gebruikers verschillende rechten toegekend kunnen worden.
- Zo kun je het recht hebben om gegevens uit de database te op te vragen, maar niet om nieuwe gegevens in te voeren.
- Relationele databases zijn geavanceerde programma's die geoptimaliseerd zijn voor bepaalde taken, zoals bijvoorbeeld het zoeken door numerieke velden, data sorteren, etc. Deze technologie maakt het mogelijk om zeer snel gegevens te zoeken en te presenteren.
- Voor het bevragen van een relationele database is een krachtige 'query taal' beschikbaar met de naam '**Structured Query Language**' (SQL). Deze taal maakt het onder andere mogelijk om zeer geavanceerde gegevensselecties te maken uit de database.
- Het relationeel is een standaard, net als SQL, de taal om relationele database te bevragen. Standaardisering maakt het mogelijk om gegevens uit de ene database vaak relatief eenvoudig naar de andere database over te zetten.

3.1 Tabellen en de primaire sleutel

Een relationele database bestaat uit tabellen. In deze tabellen zijn gegevens van dezelfde soort in rijen of in vaktaal records opgeslagen.

Om bijvoorbeeld klanten uniek te kunnen identificeren moet elke klant gekoppeld zijn aan een uniek stukje informatie, bijvoorbeeld een klantnummer.

Deze unieke informatie wordt de primaire sleutel (**primary key**) genoemd.

Een van de belangrijkste regels van het relationeel model is dat alle rijen in een tabel uniek te identificeren zijn door middel van de primaire sleutel.

Er zijn tal van voorbeelden te bedenken van nummers en codes uit het dagelijks leven die waarschijnlijk dienen als primaire sleutel in een database.

- Een bestelnummer
- Een rekeningnummer
- Een sofinummer

- Een factuurnummer
- Een klantnummer
- Een productnummer

Wat hebben al deze codes gemeen?

- Ze zijn allemaal **uniek**. Jouw rekeningnummer bestaat altijd maar één keer, net als een factuurnummer, je sofinummer, etc.
- Ze werken allemaal als toegangsweg naar meer informatie.
- Aan een factuurnummer is een datum, een bedrag, etc. gekoppeld.
- Aan een productnummer kan een productbeschrijving, een plaatje, etc. gekoppeld zijn.

De primaire sleutel wordt dus gebruikt om rijen in tabellen uniek te identificeren en om gegevens in verschillende tabellen aan elkaar te koppelen.

Primaire sleutels zijn zoals gezegd altijd uniek en mogen daarom maar 1 keer voorkomen in de kolom. Het is in onderstaande tabel dus niet mogelijk om een nieuwe rij met klantnummer 1 toe te voegen. Geef je aan dat het klantnummer de primaire sleutel is, dan zal het databasesysteem er automatisch voor zorgen dat klantnummers in die kolom uniek zijn.

klantnummer	voornaam	tussenvoegsel	achternaam	e-mailadres	telefoonnummer
1	Jan		Jansen	janjansen@hotmail.nl	0612345678
2	Klaas	van	Baalen	kvb@kvbmail.nl	0201234567

3.2 Gegevens koppelen

contactmoment_id	klantnummer	datum	tijd	beschrijving	inkomend_uit	com_type
1	1	10-10-2006	9:30:00	Klant geeft aan problemen	inkomend	telefoon
2	1	14-10-2006	14:21:00	Doorgeven wijziging	inkomend	telefoon
3	2	16-10-2006	17:59:00	Geachte meneer Ha	uitgaand	e-mail
4	2	17-10-2006	17:55:00	Hallo, kunt u mij ver	inkomend	e-mail
5	1	17-10-2006	12:40:00	Brief met gegevens	inkomend	brief
6	2	18-10-2006	17:50:00	Geachte meneer Ha	uitgaand	brief
7	2	18-10-2006	8:45:00	Klant gebeld om te v	uitgaand	telefoon

Op elke rij van de contactmomententabel staat informatie over een uniek contactmoment. Elk contactmoment is uniek identificeerbaar door de primaire sleutel, het **contactmoment_id** (eerste kolom). Bovendien is elk contactmoment gekoppeld aan een klant uit de klanttabel door middel van het **klantnummer** in de tweede kolom.

Met behulp van het klantnummer kunnen bij elke klant in de klanttabel de contactmomenten opgezocht worden en andersom bij elk contactmoment de betreffende klant uit de klanttabel. Er is hier sprake van een '1 op veel' relatie tussen de klant en zijn of haar contactmomenten.

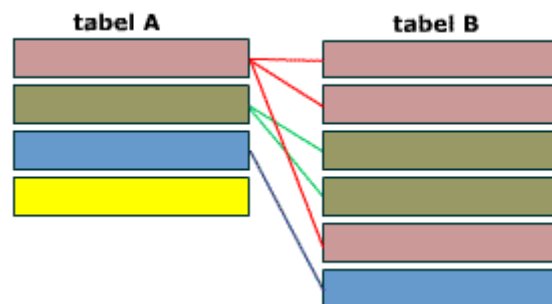
Nu je een voorbeeld gezien hebt van een database is het tijd om in te gaan op de relaties die kunnen bestaan tussen tabellen. Het voorbeeld van de klanttabel en de

contactmomententabel is een 'één op veel relatie'. Voor elke klant (1) kunnen meerdere (veel) contactmomenten opgeslagen zijn. Bij het ontwerpen van relationele databases onderscheiden we 3 soorten relaties tussen tabellen.

- De **één op veel** relatie
- De **veel op veel** relatie
- De **één op één** relatie

3.3 De een op veel relatie

Bij de **één op veel relatie** is 1 ding gekoppeld aan 0, 1 of meer andere dingen. Je zag al een voorbeeld met klanten en contactmomenten. 1 klant kan 0, 1 of meerdere contactmomenten hebben en een contactmoment is altijd gekoppeld aan 1 klant.



Hetzelfde geldt voor de relatie tussen moeders en hun kinderen. Elk kind heeft maar 1 moeder, maar een moeder kan geen, één of meerdere kinderen hebben. De één op veel relatie modelleer je in een relationele database als twee tabellen. Elke rij in tabel A correspondeert met 0, 1, of meerdere rijen uit tabel B. (zie afbeelding).

De klanttabel speelt de rol van tabel A en de contactmomententabel die van tabel B. Elke rij in tabel A (ofwel, elke klant) is gekoppeld aan 0,1 of meerdere rijen (ofwel, contactmomenten) in tabel B.

Het klantnummer is de primaire sleutel in tabel A, het klantnummer in tabel B wordt de *vreemde sleutel* genoemd. Een vreemde sleutel is een veld in een tabel dat verwijst naar de primaire sleutel van een andere tabel. Het klantnummer in onze database is de primaire sleutel in de klanttabel en in andere tabellen is het klantnummer een vreemde sleutel.

tabel klant

ps

klantnummer	voornaam	tussenvoegsel	achternaam	e-mailadres	telefoonnummer
1	Jan		Jansen	janjansen@hotr	0612345678
2	Klaas	van	Baalen	kvb@kvbmail.nl	0201234567

	ps	vs					
contactmoment	klant		datum	tijd	beschrijving	inkomend_uur	com_type
1	1		10-10-2006	9:30:00	Klant geeft aan probleem	inkomend	telefoon
2	1		14-10-2006	14:21:00	Doorgeven wijziging	inkomend	telefoon
3	2		16-10-2006	17:59:00	Geachte meneer Ha	uitgaand	e-mail
4	2		17-10-2006	17:55:00	Hallo, kunt u mij ver	inkomend	e-mail
5	1		17-10-2006	12:40:00	Brief met gegevens	inkomend	brief
6	2		18-10-2006	17:50:00	Geachte meneer Ha	uitgaand	brief
7	2		18-10-2006	8:45:00	Klant gebeld om te	uitgaand	telefoon

Klantnummer is de primaire sleutel (ps) in de klanttabel

tabel contactmoment Klantnummer is de vreemde sleutel (vs) in de contactmomententabel. Hij verwijst naar het klantnummer in de klanttabel. Het contactmoment_id is de primaire sleutel (ps) in de contactmomententabel.

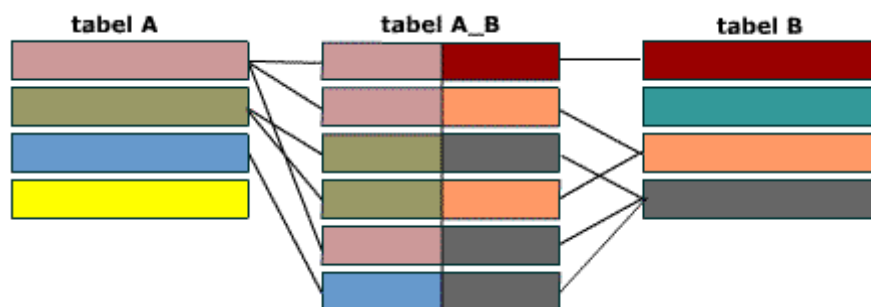
3.4 De veel op veel relatie

De veel op veel relatie is een relatie waarbij meerdere rijen uit tabel A gekoppeld kunnen zijn aan meerdere rijen uit tabel B.

Een voorbeeld is een school, waarbij docenten les geven aan studenten. Elke docent geeft les aan 0, 1 of meer studenten. Andersom kan elke student les krijgen van 0, 1 of meerdere docenten.

Een ander voorbeeld is de relatie tussen biermerken en leveranciers. Elke leverancier levert meerdere biermerken, maar elk biermerk kan ook door meerdere leveranciers geleverd worden.

De veel op veel relatie modelleren we met drie tabellen. Twee brontabellen en één koppeltabel.



Tabel A en tabel B zijn beide tabellen met een primaire sleutel. Tabel A_B verbindt records uit tabel A en B en heet een 'koppeltabel'. Een koppeltabel bestaat uit maar twee kolommen die allebei een vreemde sleutel bevatten. De vreemde sleutel in de linkerkolom verwijst naar de primaire sleutel uit tabel A en die in de rechterkolom naar de primaire sleutel uit tabel B. De primaire sleutel in koppeltabel A_B is *samengesteld* uit de twee vreemde sleutels. Je ziet in het voorbeeld dan ook dat de *combinatie* van de twee velden uniek moet zijn in koppeltabel A_B. In het

klantcontactstelsysteem komt (vooralsnog) geen veel op veel relatie voor, daarom staat hieronder een voorbeeld met bieren en hun leveranciers.

Tabel bier		Tabel bier_leverancier		Tabel leverancier	
<u>bier_id</u>	naam	<u>bier_id</u>	<u>leverancier_id</u>	<u>leverancier_id</u>	naam
157	Gentse Tripel	157	AC001	AB999	De vrolijke drinker
158	Uilenspiegel	157	AB899	AC001	Horeca Import NL
146	Duvel	157	AC009	AC002	van Gent bierimport
123	Leffe	158	AC009	AB899	Jansen Horeca
222	Brugse Tripel	163	AC009	AC008	De bierleverancier
160	Sint Bernardus Pater	160	AB999	AC009	Petersen Drankenhandel
163	Jupiler	163	AB999		
		222	AB999		
		123	AB999		
		146	AB999		
		158	AB999		
		157	AB999		

Bovenstaande tabellen koppelen bieren en leveranciers in een veel op veel relatie. Eén bier kan door 0, 1 of meerdere leveranciers geleverd worden. Gentse Tripel (157) wordt bijvoorbeeld geleverd door Horeca Import NL (157, AC001), Jansen Horeca (157, AB899) en Petersen Drankenhandel (157, AC009). Andersom levert Petersen Drankenhandel 3 bieren uit de bierentabel, te weten Gentse Tripel (157, AC009), Uilenspiegel (158, AC009) en Jupiler (163, AC009).

Merk op dat in bovenstaande tabellen de primaire sleutelvelden blauw en onderstreept zijn weergegeven. In modellen van relationele databases worden primaire sleutels vaak onderstreept.

Zoals je ziet is de primaire sleutel van de koppeltabel 'bier_leverancier' samengesteld uit twee velden. Elke rij in de koppeltabel bestaat uit de combinatie van een bier_id met een leverancier_id.

Deze tabel bestaat zögezegd uit een **samengestelde primaire sleutel**. Primaire sleutels moeten uniek zijn, dus de combinaties moeten uniek zijn in de tabel.

Een ander goed gebruik is om in databasemodellen relaties te benoemen. Je zou bij de hierboven afgebeelde relatie in een model de opmerkingen "levert" kunnen zetten. Elke rij in de tabel zegt namelijk "bier X wordt geleverd door leverancier Y", of "Leverancier Y levert bier X".

Het benoemen van relaties maakt een databasemodel leesbaarder. Deze benoeming komt niet terug in de eigenlijke database of in de software die gebruik maakt van de database.

3.5 De een op een relatie

De één op één relatie tussen tabellen komt niet heel vaak voor. Bij de één op één relatie heeft elke rij in tabel A 0 of 1 corresponderende rij in tabel B. Deze relatie wordt weleens gebruikt om de tekortkomingen van een databasesysteem te omzeilen door bijvoorbeeld een tabel op te delen om prestatiewinst te behalen.

Merk op dat elke *individuele tabel* vol zit met 1 op 1 relaties: de relatie tussen de primaire sleutel en de rest van de gegevens op die rij is een 1 op 1 relatie, of laat ik zeggen *zou dat moeten zijn* als je het relationeel model helemaal volgt.

Een voorbeeldje: in de biertabel op de vorige pagina is Gentse Tripel in een 1 op 1 relatie gekoppeld aan de primaire sleutel 157. Dit is precies de reden dat 1 op 1 relaties *tussen tabellen* niet vaak voorkomen.

Een relationeel databaseontwerp is zoals je hebt gelezen een verzameling van tabeldefinities, waarin gegevens in verschillende tabellen gekoppeld zijn in relaties. Voor het kiezen van de juiste relaties tussen gegevens formuleert het relationeel model een aantal 'normaaltvormen'. In het volgende hoofdstuk ga ik in op het doel van normaliseren en de taken die hierbij horen.

4 Database normaliseren

De regels voor goed relationeel databaseontwerp zijn samengevat in 5 'normaaltvormen', waarbij de eerste normaalvorm de laagste en de vijfde de hoogste (meest genormaliseerd) is. Deze normaalvormen zijn richtlijnen voor het juist ontwerpen van een relationele database.

Normaliseren heeft een aantal doelen.

- **Flexibiliteit.** De genormaliseerde structuur van de database zorgt ervoor dat gegevens op veel verschillende manieren opgevraagd en bijgewerkt kunnen worden.
- **Integriteit.** In een genormaliseerde database ben je gegevens zeer betrouwbaar opslaan.
- In een genormaliseerde database worden gegevens maar op 1 plek opgeslagen. Als je data wil invoeren, aanpassen of verwijderen hoef je dat dus maar op 1 plek te doen.

Het normaliseren van een database schijnt voor veel mensen taaie materie te zijn.

Het normaliseren van een database komt eigenlijk neer op het nastreven van de volgende zaken en die zijn met een beetje oefening en puzzelen vaak redelijk gemakkelijk te realiseren.

- Het verdelen van gegevens in logische samenhangende groepen.

- Het minimaliseren van de hoeveelheid data die dubbel opgeslagen is, ofwel het voorkomen van 'redundancy'.
- De gegevens zo organiseren dat het aanpassen of verwijderen van een gegeven altijd maar op één plek hoeft te gebeuren.
- Gegevens zo organiseren dat ze snel en efficiënt op te vragen zijn.

De meeste applicaties gebruiken databases die zijn genormaliseerd tot de 1ste, de 2de of de 3de normaalvorm.

De 4de en 5de normaalvorm zie je zelden. In deze cursus databases wordt daarom alleen de eerste, tweede en derde normaalvorm besproken.

5 Uitgewerkte methodiek met voorbeeld

Als je een database ontwikkelt is het belangrijk dat het aan een aantal eisen voldoet. Eén daarvan is de eis dat het bestand optimaal ontworpen is. Dit betekent dat er geen overbodige redundantie mag zijn.

Er zijn meerdere stappen om naar deze goede database te komen. Deze stappen zijn Normaalvormen.

1. 0 NV (Nulde Normaalvorm - Inventarisatie)
2. 1 NV (Eerste Normaalvorm)
3. 2 NV (Tweede Normaalvorm)
4. 3 NV (Derde Normaalvorm)

Verder zijn er ook nog de BCNV (Boyce Codd Normaalvorm), 4NV en 5NV.

Op deze laatste Normaalvormen gaan we niet in omdat deze bijna nooit gebruikt worden.

Uitgangspunt voor het normaliseren is steeds de **informatiebehoefte** van de toekomstige gebruiker van de database.

De indeling en inhoud van de tabellen wordt bepaald door de informatie die de gebruiker wenst te zien.

5.1 Nulde Normaalvorm

Om tot de Nulde Normaalvorm te komen moeten we de informatiebehoefte gaan **inventariseren**. Voor het gemak heb ik even een afleverbon gemaakt:

AFLEVERBON

ORDERNR.: 3405
KLANTNR.: 100658

ORDERDATUM: 15-06-2004

C1000 Oud-Beijerland
Graaf van Egmondstraat 67B
3261 AK Oud-Beijerland

Artnr.	Art.omschrijving	Aantal	Prijs	Totaal
15201	Biologisch ei 6st per 20	1	12,50	12,50
21987	Biologische kaas Campina 10*0.5 kg Aanbiedingpr.	6	8,99	53,94
12365	Rolcontainer	1	125,00	125,00
12366	CBL Krat Zwart 24, emballage	7	3,60	25,20

Totaal € 216,64

Nu gaan we inventariseren. Dit betekent dat we alle gegevens op de afleverbonnetjes onder elkaar gaan zetten.

We krijgen dan het volgende:

0 NV

ORDERS

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

artnr

artomschrijving

aantal

prijs

regeltotaal

eindtotaal

Zoals je ziet heb ik de inventarisatie ook een naam gegeven. Deze staat in hoofdletters.

5.2 Eerste Normaalvorm 1 NV

Na het inventariseren is het de bedoeling dat we naar de Eerste Normaalvorm moeten gaan. Dit doen we altijd met de volgende stappen:

1. Verwijder alle procesgegevens.

2. Geef de sleutel van de groep aan.
 3. Geef de deelverzameling aan die een herhaald aantal keren voorkomt t.o.v. de primaire sleutel.
 4. Herhaal de sleutelgegevens van de oorspronkelijke groep samen met de gegevens van de zich herhalende deelverzameling als een nieuwe groep.
 5. Verwijder de zich herhalende deelverzameling uit de oorspronkelijke groep.
- Oké, dan nu verder met ons voorbeeld:

AFLEVERBON				
ORDERNR.: 3405		ORDERDATUM: 15-06-2004		
KLANTNR.: 100658				
C1000 Oud-Beijerland Graaf van Egmondstraat 67B 3261 AK Oud-Beijerland				
Artnr.	Art.omschrijving	Aantal	Prijs	Totaal
15201	Biologisch ei 6st per 20	1	12,50	12,50
21987	Biologische kaas Campina 10*0.5 kg Aanbiedingpr.	6	8,99	53,94
12365	Rolcontainer	1	125,00	125,00
12366	CBL Krat Zwart 24, emballage	7	3,60	25,20
				Totaal € 216,64

1. Verwijder alle procesgegevens.

In dit voorbeeld is het natuurlijk overduidelijk wat de procesgegevens zijn. Het **regeltotaal** wordt berekend uit **aantal * prijs** en het **eindtotaal** wordt berekend als **som van alle regeltotalen**. Deze twee strepen we dus weg.

2. Geef de sleutel van de groep aan.

Een sleutel is altijd uniek. Je ziet ze op het Internet vaak als 'id'. Het kan natuurlijk anders heten, maar het heeft wel dezelfde functie: het uniek identificeren van een tupel.

In ons voorbeeld is de sleutel ordernr, immers een klant kan meerdere keren een order plaatsen, maar de orders kunnen nooit hetzelfde nummer hebben.

De sleutel geven we aan door het te onderstrepen.

3. Geef de deelverzameling aan die een herhaald aantal keren voorkomt t.o.v. de primaire sleutel. Deze deelverzameling noemt men ook wel eens de Repeterende Groep (RG). Dit zijn de gegevens die vaker voorkomen. In ons voorbeeld is dit het tabelletje met de bestelde goederen. Geef deze gegevens aan in je inventarisatie. We hebben nu dus het volgende:

4. ONV

ORDERS

ordernr

orderdatum

klantnr
klantnaam
adres
postcode
plaats

RG artnr

RG artomschrijving

RG aantal

RG prijs

~~X regeltotaal (procesgegeven)~~

~~X eindtotaal (procesgegeven)~~

Herhaal de sleutelgegevens van de oorspronkelijke groep samen met de gegevens van de zich herhalende deelverzameling als een nieuwe groep.

0 NV

ORDERS

ordernr
orderdatum
klantnr
klantnaam
adres
postcode
plaats

RG artnr

RG artomschrijving

RG aantal

RG prijs

BESTELDE_ARTIKELEN

ordernr
artnr
artomschrijving
aantal
prijs

Zoals je ziet heeft de tweede groep nog geen sleutel. De sleutel moet je zodanig kiezen dat er zo min mogelijk herhaalde groepen voorkomen t.o.v. deze sleutel. Het liefst natuurlijk geen herhaalde groepen meer, anders moet je de vorige stappen nog een keer herhalen.

Meestal kun je een combinatie nemen van de sleutel van de oorspronkelijke groep en het gegeven dat in de Repeterende Groep de sleutelrol vervult.

De sleutel wordt in dit geval een combinatie van ordernr en artnr.

5. Verwijder de zich herhalende deelverzameling uit de oorspronkelijke groep.
Hier moeten we de RG dus weer opruimen en dan hebben we de Eerste Normaalvorm:

1NV

ORDERS

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

BESTELDE_ARTIKELEN

ordernr

artnr

artomschrijving

aantal

prijs

5.3 Tweede Normaalvorm

Nu is het tijd om naar de Tweede Normaalvorm te gaan.

Dit doen we ook weer met een aantal vaste stappen:

1. Geef de attributen aan die niet functioneel afhankelijk zijn van de volledige sleutel.
2. Formeer een aparte groep voor ieder deel van de sleutel waarvan de attributen functioneel afhankelijk zijn.
3. Neem in iedere groep de attributen met het bijbehorende sleuteldeel op en wijs de primaire sleutel aan.
4. Verwijder deze attributen uit de oorspronkelijke groep.

Het voorbeeld:

AFLEVERBON

ORDERNR.: 3405
KLANTNR.: 100658

ORDERDATUM: 15-06-2004

C1000 Oud-Beijerland
Graaf van Egmondstraat 67B
3261 AK Oud-Beijerland

Artnr.	Art.omschrijving	Aantal	Prijs	Totaal
15201	Biologisch ei 6st per 20	1	12,50	12,50
21987	Biologische kaas Campina 10*0.5 kg Aanbiedingpr.	6	8,99	53,94
12365	Rolcontainer	1	125,00	125,00
12366	CBL Krat Zwart 24, emballage	7	3,60	25,20

Totaal € 216,64

Geef de attributen aan die niet functioneel afhankelijk zijn van de volledige sleutel.
We herhalen nog even de Eerste Normaalvorm:

1NV**ORDERS**

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

BESTELDE_ARTIKELEN

ordernr

artnr

artomschrijving

aantal

prijs

De eerste groep komt niet in aanmerking voor deze stap omdat het niet beschikt over een samengestelde sleutel.

Binnen de andere groep zijn er wel gegevens die functioneel afhankelijk zijn van een deel van de sleutel.

Kijk maar eens naar artomschrijving en prijs.

Blijkbaar zijn deze afhankelijk van artnr en niet van ordernr. Je kunt dit nagaan door te kijken wat er veranderd als het artnr gewijzigd wordt.

We geven dit zo aan in het voorbeeld:

1NV

ORDERS

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

BESTELDE_ARTIKELEN

ordernr

> artnr

> artomschrijving

aantal

> prijs

Let erop dat aantal niet alleen afhankelijk is van artnr. Het aantal per artikel kan verschillen per order.

2. Formeer een aparte groep voor ieder deel van de sleutel waarvan de attributen functioneel afhankelijk zijn.

Het kan gebeuren dat een samengestelde sleutel in meerdere delen gesplitst kan worden en dat van ieder deel afzonderlijk attributen functioneel afhankelijk zijn. Er moeten dan meerdere groepen gevormd worden. In het voorbeeld ontstaat slechts één nieuwe groep: ARTIKELEN.

3. Neem in iedere groep de attributen met het bijbehorende sleuteldeel op en wijs de primaire sleutel aan.

1NV

ORDERS

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

BESTELDE_ARTIKELEN

ordernr

> artnr

> artomschrijving

aantal

> prijs

ARTIKELEN

artnr

artomschrijving

prijs

Verwijder deze attributen uit de oorspronkelijke groep.

Na deze stap hebben we onze Tweede Normaalvorm en begint het al lekker op te schieten:

2 NV

ORDERS

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

BESTELDE_ARTIKELEN

ordernr

artnr

aantal

ARTIKELEN

artnr

artomschrijving

prijs

5.4 Derde Normaalvorm

De laatste Normaalvorm die ik behandel in deze tutorial is de Derde Normaalvorm.

Hiervoor hebben we de volgende stappen:

1. Geef de niet-sleutel attributen aan die functioneel afhankelijk zijn van andere niet-sleutel attributen.
2. Formeer een aparte groep voor ieder attribuut of combinatie van attributen, waar andere attributen functioneel van afhankelijk zijn.
3. Neem in iedere groep de attributen met bijbehorende sleutel op en wijs de primaire sleutel aan.
4. Verwijder de attributen van de nieuwe groep(en) uit de oorspronkelijke groep.

Ik begin nog maar eens met de uitkomst van de Tweede Normaalvorm:

2NV

ORDERS

ordernr
orderdatum
klatnr
klatnaam
adres
postcode
plaats

BESTELDE_ARTIKELEN

ordernr
artnr
aantal

ARTIKELEN

artnr
artomschrijving
prijs

1. Geef de niet-sleutel attributen aan die functioneel afhankelijk zijn van andere niet-sleutel attributen.

Als we kijken in de tabel ORDERS zien we dat er een aantal gegevens zijn die niet afhankelijk zijn van ordernr. Deze zijn klatnaam, adres, postcode en plaats. Deze zijn afhankelijk van het klatnr.

Klatnr is in deze tabel een niet-sleutelattribuut, dus deze stap is makkelijk uit te voeren:

(de afhankelijkheid is aangegeven met een A)

2NV

ORDERS

ordernr
orderdatum
A klatnr
A klatnaam
A adres
A postcode
A plaats

BESTELDE_ARTIKELEN

ordernr
artnr

aantal

ARTIKELEN

artnr

artomschrijving

prijs

2. Formeer een aparte groep voor ieder attribuut of combinatie van attributen, waar andere attributen functioneel van afhankelijk zijn.

Vrij vertaald moeten we voor de groep die we net aangegeven hebben een nieuwe groep maken:

2NV

ORDERS

ordernr

orderdatum

A klantnr

A klantnaam

A adres

A postcode

A plaats

KLANTEN

klantnaam

adres

postcode

plaats

BESTELDE_ARTIKELEN

ordernr

artnr

aantal

ARTIKELEN

artnr

artomschrijving

prijs

3. Neem in iedere groep de attributen met bijbehorende sleutel op en wijs de primaire sleutel aan.

We moeten dus de sleutel waarvan de nieuwe groep afhankelijk is in de nieuwe

groep plaatsen, daarna moeten we de sleutel aangeven. De nieuwe groep ziet er dan zo uit:

KLANTEN

klantnr

klantnaam

adres

postcode

plaats

Let er wel op dat klantnr niet zomaar uit ORDERS gehaald kan worden, dan zouden de orders namelijk nooit aan een klant gekoppeld kunnen worden.

4. Verwijder de attributen van de nieuwe groep(en) uit de oorspronkelijke groep.

Als we deze stap hebben gedaan zijn we klaar met het normaliseren. Dit ziet er dan zo uit:

3NV

ORDERS

ordernr

klantnr

orderdatum

KLANTEN

klantnr

klantnaam

adres

postcode

plaats

BESTELDE_ARTIKELEN

ordernr

artnr

aantal

ARTIKELEN

artnr

artomschrijving

prijs

5.5 Afsluiting

In deze tutorial heb ik geprobeerd om je een klein beetje de beginselen van het Normaliseren onder de knie te brengen. Natuurlijk heb ik hier een makkelijk voorbeeld gebruikt. Maar met te complexe voorbeelden kom je er met een begin niet uit.

Ik hoop dat mensen gaan inzien dat normaliseren een geweldige manier is om tot een goede structuur van de database te komen.

5.6 Begrippen

Begrippen

- Redundantie: Het dubbel opslaan van gegevens.
- Gegevens: Feiten of gebeurtenissen die op een bepaalde manier vastgelegd zijn.
- Informatie: De betekenis die aan gegevens ontleend wordt.
- Procesgegevens: Gegeven wat uit een berekening van andere gegevens wordt afgeleid.
- Tupel: Rij in een tabel
- Attribuut: Kolom in een tabel
- Graad: Het aantal attributen in een tabel
- Kardinaliteit: Het aantal tupels in een tabel
- Functionele afhankelijkheid: We zeggen dat een gegeven B functioneel afhankelijk is van een ander gegeven A als er bij een waarde A één (en niet meer dan één) waarde van B kan voorkomen (naam van een artikel is functioneel afhankelijk van het artnr).

6 Oefeningen:

6.1 De sportclub

U wenst de administratie van de sportvereniging **“Fitter door sport vzw”** bij te houden. Zoals uit onderstaande gegevensverzameling kan afgeleid worden, bestaat deze sportclub uit een aantal leden. Deze leden kunnen zich voor één of meerdere sportdisciplines inschrijven. Er zijn wel een aantal voorschriften nodig om de tabellen te ontwerpen die leiden tot een efficiënte relationele database.

Nog even de meest gebruikte normalisatiestappen van Boyce/Codd opsommen:

1. NV 0 Inventariseer alle relevante gegevens en verwijder de procesgegevens.
2. NV 1 Verwijder alle “herhalende groepen”.
3. NV 2 Verwijder alle gegevens die niet of slechts gedeeltelijk afhankelijk zijn van de primaire sleutel.

4. NV 3Verwijder verborgen afhankelijkheden.

Met verwijderen verstaan we: het plaatsen van de verwijderde gegevens in een andere tabel met een verwijzing (referentie) naar de oorspronkelijke tabel.

6.1.1 Een overzicht:

naam	adres	post	gemeente	m/v	geboren	sporttak	lidgeld	bet	inschrijving	lft
Maas Jan	Poststraat 33	2500	Lier	m	10/01/1974	voetbal	10	ja	2003-01-01	29
						tennis	25	nee	2003-09-25	
Pijl Luc	Vrijgeweide 12	3545	Halen	m	12/11/1982	volleybal	18	ja	2003-02-10	20
						surfen	30	ja	2003-03-15	
De Cock Fanny	Dorpsstraat 28	3390	Tielt-Winge	v	10/10/1985	fitness	30	ja	2003-03-16	18
Verheyden Inge	Veldweg 33	3290	Diest	v	5/08/1987	fitness	30	ja	2003-04-17	16
						aerobic	20	nee	2003-10-18	
Pelsmakers An	Demerstraat 13	3290	Diest	v	10/10/1988	squash	15	ja	2003-08-19	15

- 6.1.2 Pas de normalisatiestappen toe op het hierboven geschetste onderwerp.**
- 6.1.3 Ontwerp de tabellen, rekening houdend met een aantal voorschriften.**
- 6.1.4 Definieer de primaire sleutels.**
- 6.1.5 Leg de passende relaties tussen de tabellen.**
- 6.1.6 Maak de nodige formulieren om de gegevens toe te voegen.**
- 6.1.7 Welke gegevens zijn absoluut noodzakelijk om "operationeel" te kunnen zijn? Verklaar.**
- 6.1.8 Maak een formulier om leden daadwerkelijk te kunnen inschrijven.**
- 6.1.9 Nulde normaalvorm inventariseren van de gegevens**

	Veldnaam	Gegevenstype	
	naam	Tekst	
	voornaam	Tekst	
	straat	Tekst	
	nr	Tekst	
	postcode	Tekst	
	gemeente	Tekst	
	geslacht	Tekst	
	geboortedatum	Tekst	
	sporttak1	Tekst	
	sporttak2	Tekst	
	sporttak3	Tekst	
	sporttak4	Tekst	
	lidgeld1	Tekst	
	lidgeld2	Tekst	
	lidgeld3	Tekst	
	lidgeld4	Tekst	
	betaald1	Tekst	
	betaald2	Tekst	
	betaald3	Tekst	
	betaald4	Tekst	
	inschrijf1	Tekst	
	inschrijf2	Tekst	
	inschrijf3	Tekst	
	inschrijf4	Tekst	
	▶ leeftijd	Tekst	▼

6.1.10 Eerste normaalvorm:

Verwijder overvloedige niet relevante (belangrijke) gegevens

Zet herhaalde groepen in een aparte tabel met verwijzing naar de oorspronkelijke tabel, want anders verlies je gegevens.

Verwijder ook berekende velden (procesgegevens)

Geef elke tabel een primaire sleutel/

Resultaten

Microsoft Access - [tblLeden : Tabel]			
Bestand Bewerken Beeld Invoegen Extra Venster Help			
	Veldnaam	Gegevenstype	
PK	lidId	AutoNummering	
	naam	Tekst	
	voornaam	Tekst	
	straat	Tekst	
	nr	Tekst	
	postcode	Tekst	
	gemeente	Tekst	
	geslacht	Tekst	
	geboortedatum	Tekst	
	sporttaknr	Tekst	verwijst naar de tabel sporttakken

Microsoft Access - [tblSporttakken : Tabel]			
Bestand Bewerken Beeld Invoegen Extra V			
	Veldnaam	Gegevenstype	
PK	sporttakID	AutoNummering	
	sporttak	Tekst	
	lidgeld	Tekst	
	inschrijdatum	Tekst	
	betaald	Tekst	

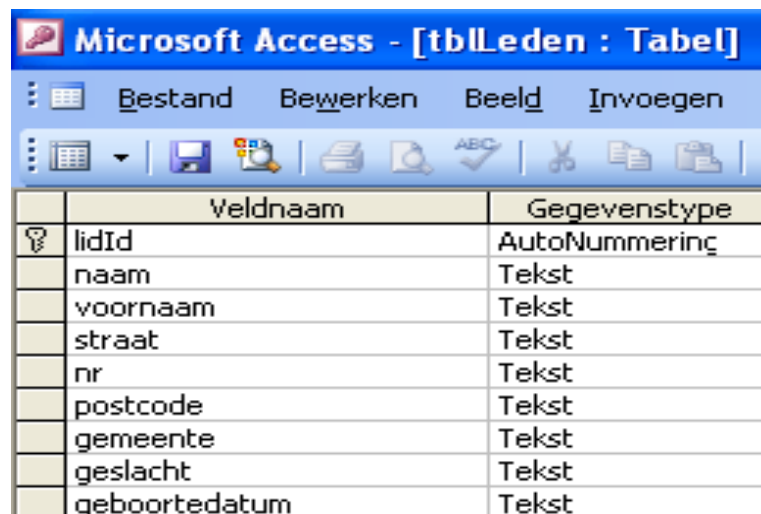
6.1.11 Tweede normaalvorm:

Zet alle attributen (velden of gegevens) die NIET of slechts gedeeltelijk afhankelijk zijn van de primaire sleutel in een aparte tabel met behoud van de gegevens (wie wat waar door relatie PK → FK (foreign key))

tblInschrijvingen : Tabel		
	Veldnaam	Gegevenstype
PK	lidnr	Tekst
FK	sporttaknr	Tekst
	inschrijfdatum	Tekst
	betaald	Tekst

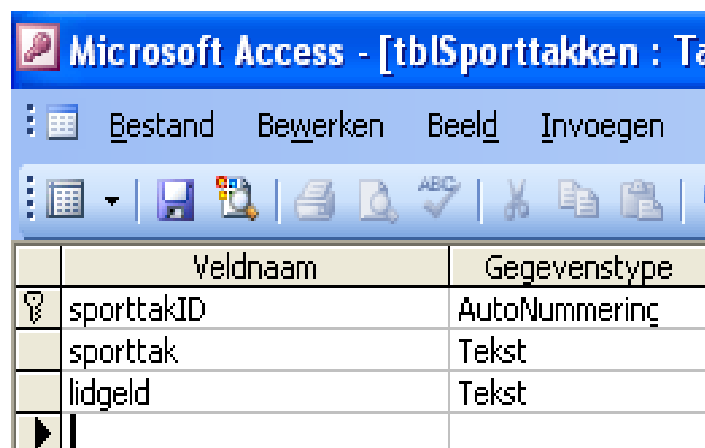
De sleutel is samengesteld zodanig dat één lid slechts voor één sporttak persoonlijk kan inschrijven.

Tabel leden



	Veldnaam	Gegevenstype
	lidId	AutoNummering
	naam	Tekst
	voornaam	Tekst
	straat	Tekst
	nr	Tekst
	postcode	Tekst
	gemeente	Tekst
	geslacht	Tekst
	geboortedatum	Tekst

Tabel sporttakken



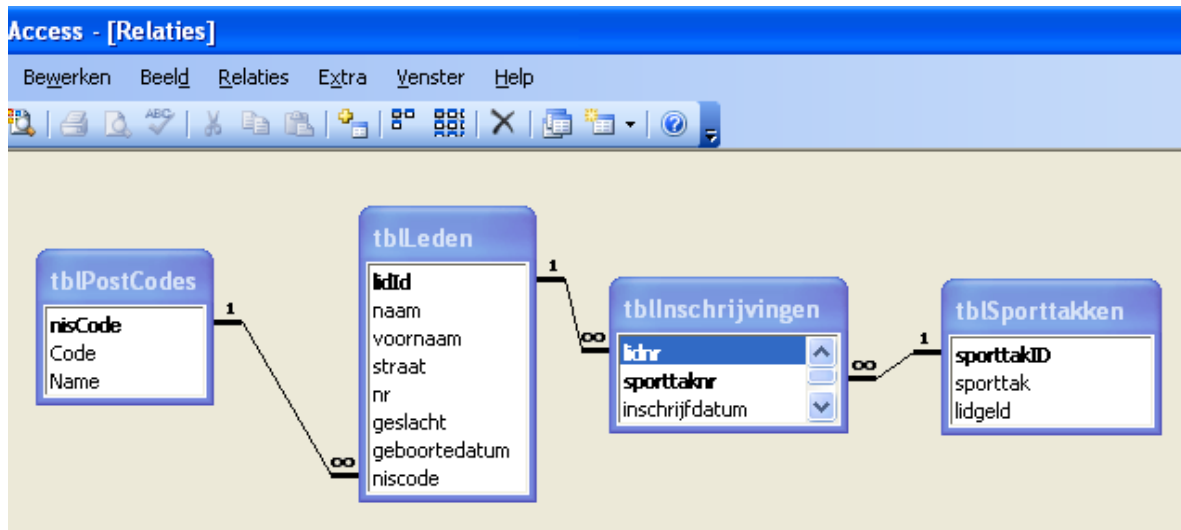
	Veldnaam	Gegevenstype
	sporttakID	AutoNummering
	sporttak	Tekst
	lidgeld	Tekst

6.1.12 Derde normaalvorm:

Verwijder verborgen afhankelijkheden zoals hier de postcode en de gemeente door een aparte al beschikbare tabel te importeren.

De verwijzing naar de niscode plaatsen in de tabel leden.

6.1.13 Na normalisatie en relaties



6.1.14 Implementeren van de database

Data toevoegen via formulieren:

Sporttakken

6.2 Situatie 1: Abonnementen

De abonneeadministratie van een uitgeverij, die meerdere weekbladen onder haar hoede heeft, verstuurt regelmatig aanmaningen naar abonnees die niet op tijd betaald hebben.

Hieronder vindt u zo'n voorbeeld.

Uitgeverij Knick Knack – Demerstraat 35 3290 DIEST

Abonnementsnummer: 47392

Abonneenummer 32845

De heer Paul DECOCK

Kruisstraat 122

3390 TIELT-WINGE

Geachte abonnee

Uit onze administratie is gebleken dat u uw abonnement voor het blad Voetbalmagazine nog niet betaald heeft.

Wij verzoeken u het verschuldigde bedrag van 63,96 EUR met het hierbij gevoegde overschrijvingsformulier zo vlug mogelijk te willen regelen.

Met de meeste hoogachting

Jos Den Inner


Debiteurenadministratie

6.3 Opdracht 2: Puntenboek leerkracht

Een leerkracht gebruikt o.a. het overzicht zoals dat in figuur 1 staat afbeeld.

Hierop houdt hij de cijfers van een klas bij, tezamen met het vak waarvoor deze cijfers zijn behaald.

De leerkracht geeft namelijk meerdere vakken, aan meerdere klassen.



Vak	Informatica		
Klas	3 BA1		
Toetsdatum	Leerlingnummer	Naam	Cijfer
12-05-2001	75564	Piet Overloon	7,3
	74421	Klaas Dam	4,2
	69642	Gerrit Stram	9,0

18-05-2001	75564	Piet Overloon	8,7
	80032	Gerda Roever	5,4
	74421	Klaas Dam	3,8
	69642	Gerrit Stram	8,4

Figuur 1

6.4 Opdracht 3: de garage

We gaan uit van een overzicht zoals dat bij een autodealer wordt gebruikt, zie figuur 1.

Automobielbedrijf Mandarijntje Spijkenisse			
Werkplaats factuur			
Merk	Citroen	Factuurnummer : 0200278	
Type	Xantia		
Kenteken	LT-SN-73		
Km. Stand	47320		
Datum	2-1-2002		
Aantal	Nummer	Omschrijving	Bedrag
1	93110000	Onderhoud 45000 km	59,63
1	56	4-gas meting uitvoeren	14,79
		Totaal werkzaamheden	74,42
1	52	Afmeldkosten APK	5,84
1	997918LOS	Koelvloeistof los	1,81
1	621632	Steeklamp 5 Watt	0,65
4	BNL51240198	Bollamp Amber	15,80
1	031327	Pakking	1,06
1	6426E0	Wisserblad LV	20,00
1	1109N3	Oliefilter	15,98
1	BNL9945916	Ruitspr. Vl. 1 ltr	2,06
4,75	00	Ultraoil 10W40	34,74
		Totaal onderdelen	97,94
		BTW 19,0 %	32,75
		Totaal te voldoen	€ 130,69

Gebruik de 4 stappen om te komen tot een genormaliseerde database.

6.5 Opdracht 4: de artsenadministratie

U wenst de administratie van een huisarts te automatiseren met een tool zoals MS Access of SQL Server 2008.

Zoals uit onderstaande patiëntenoverzicht blijkt, worden de gegevens om dit overzicht te maken uit verschillende genormaliseerde tabellen gehaald.

Dat is ook uw taak rekening houdend met onderstaande bijkomende informatie.

Patiëntenoverzicht

Patiënt	Op consultatie	Aandoening
Janssens Luc	2003-07-01	Oogontsteking
	2003-07-08	Oogontsteking
	2003-09-01	Sinusitis
Peeters Anouk	2003-08-10	Oorontsteking
	2003-10-15	Bronchitis
	2003-11-12	Bronchitis
	2003-11-25	Hypertensie

Bijkomende informatie:

Patiënt:

Van de patiënt worden volgende gegevens bijgehouden:

Patiëntnr – naam en voornaam – straat en huisnr – postcode en gemeente – bloedgroep – geboortedatum –

Geslacht – leeftijd en telefoonnummer

Ontwerp de tabellen, rekening houdend met een aantal normalisatieregels.

Veldstructuur:

Houdt eveneens rekening met de veldlengte en het type veld bij het ontwerp van de tabellen.

Leg de passende relaties tussen de ontworpen tabellen.

Postcodes:

Hiervoor hebt u ergens een tabel ter beschikking.