INHOUDSOPGAVE

1 Introductie 4
2 Overzicht theorie 5
2.1 Inleiding en tabellen 5
2.2 Conceptueel datamodel 5
2.3 Relationele modellen 5
2.4 Normaliseren 5
2.5 Werken met Workbench 5
2.6 ERD Verdieping 5
2.7 Subtypering 5
3 Inleiding 6
3.1 Introductie 6
3.2 Data versus informatie 6
3.3 Hoe ontstaan databases? 7
3.4 Voorbeeld Pizzabezorging 9
3.5 Primaire sleutels 11
3.6 Primaire sleutels bij Pizzabezorging 11
3.7 Samengestelde sleutels, kandidaatsleutels en surrogaatsleutels 13
3.8 Opdrachten 13
4 Conceptueel Datamodel (ERD) 14
4.1 Weergave van de tabellen in een ERD 15
4.2 Weergave van relaties in een ERD 16
4.3 Voorbeeld Pizzabezorging 16
4.4 Voorbeelden van relaties 17
4.5 Stappenplan voor het maken van een ERD 20
4.6 Voorbeeld bibliotheek 20
4.7 Opdrachten 23
5 Relationele modellen 24
5.1 Vreemde sleutels 24
5.2 Voorbeeld Pizzabezorging 25
5.3 Voorbeeld Docent en student 25
5.4 Strokendiagrammen 26
5.5 Oefening 27

5.6 Veel-op-veel relaties omzetten 27
5.7 Tussentabellen 28
5.8 Voorbeeld Pizzabezorging 28
5.9 Oefening Pizzabezorging 29
5.10 Oefening 31
5.11 1-op-1 relatie omzetten 32
5.12 Oefening bibliotheek 33
5.13 Eisen aan primaire en vreemde sleutels. 34
5.14 Oefening Afdelingen 35
5.15 Opdracht 36
5.16 Relaties met eigen attributen 37
5.17 Oefening Meubels 38
5.18 Opdrachten 39
6 Normaliseren 40
6.1 Introductie normaliseren 40
6.2 (Samengestelde) meerwaardige attributen 40
6.3 Redundantie 42
6.4 Afleidbare gegevens 42
6.5 Het normalisatieproces 43
6.6 Voorbeeld sportwinkel 43
6.7 Oefening docentinzet 48
6.8 Opdracht 49
6.9 Voorbeeld Open Universiteit 49
6.10 Oefening Woningbezoeken 53
6.11 Intuïtief of gestructureerd? 54
6.12 Opdrachten 54
7 Werken met Workbench 55
7.1 Van conceptueel model naar implementatie 55
7.2 Symbolen in Workbench 55
7.3 Opdracht 57
7.4 Oefening bibliotheek in Workbench 57
7.5 Uitbreiding van de bibliotheek 57
7.6 Dependent en independent entiteiten 59
7.7 Oefening Projectinzet 60
7.8 Herhaling dependent en independent entiteiten 61

7.9 Opdrachten 62
8 ERD Verdieping 63
8.1 Recursieve relaties 63
8.2 Oefening recursie in Workbench 64
8.3 Identifying relatie (reprise) 65
8.4 Opdracht Hotel 67
8.5 Loops 67
8.6 Oefening docent, klas en vak 70
8.7 Opdrachten 70
9 Subtypering 71
9.1 Subtypering: Medewerker en Manager 71
9.2 Subtypering: Artsen 73
9.3 Subtypering School 74
9.4 Samenvatting subtypering 74
9.5 Opdracht 75
9.6 Implementatie van subtypering 75
9.7 Opdrachten 78
10 Oefenopdrachten 79

1 INTRODUCTIE

We gebruiken een zelfstudie document om je op weg te helpen. Neem de stof door en doe de opdrachten. We gebruiken geen boek voor deze module. In plaats daarvan verwijzen we naar korte video-instructies op *LinkedIn*.

In de tekst staat af en toe een vraag of een oefening. Onze uitwerking volgt daar direct op. Probeer dus zoveel mogelijk de vraag te beantwoorden of de oefening te doen zonder naar onze uitwerking te kijken. In sommige gevallen hebben we de uitwerking bewust op de volgende pagina geplaatst.

De bijbehorende opdrachten en hun uitwerkingen staan per hoofdstuk/onderwerp op DLO.

2 OVERZICHT THEORIE

2.1 Inleiding en tabellen

- Context van de cursus
- Inleiding databases
- Database Management Systemen

2.2 Conceptueel datamodel

- Tabellen
- Primaire sleutel
- Entiteit Relatie Diagram (ERD)

2.3 Relationele modellen

- Van ERD naar relationeel model
- Vreemde sleutel
- Strokendiagram
- Tussentabel

2.4 Normaliseren

- Normaliseren
- Verdieping normaliseren

2.5 Werken met Workbench

- Werken met Workbench

2.6 ERD Verdieping

- ERD recursie
- Verdieping ERD
- Loops

2.7 Subtypering

- ERD subtypering

3 INLEIDING

3.1 Introductie

1. Net zoals in ***Programming*** verwijzen we naar instructievideo's op *LinkedIn Learning*. Je kunt inloggen met je HvA e-mailadres.
2. We gebruiken het materiaal in de cursus ***Programming Foundations: Database***. In de volgende vier video's krijg je een kort overzicht van het hoe en waarom van databases. Later in de cursus gaan we dieper in op de verschillende onderwerpen.
3. Bekijk de video *Why use a database* uit de les ***Introduction***.
4. Bekijk de drie daaropvolgende video's (dit gaat automatisch als je niets doet):
   What you should know
   Understanding databases: Benefits of spreadsheets
   Understanding databases: Benefits of structured data

3.2 Data versus informatie

1. Een database is een verzameling opgeslagen *gegevens* (*data*). Het gaat hierbij om 'ruwe' data, zoals feiten en cijfers. De gegevens kunnen worden verwerkt tot *informatie*, dat wil zeggen, kennis die afgeleid is uit gegevens of gegevens, die in een betekenisvolle context worden gebruikt. De gegevens worden op een logische wijze gegroepeerd in tabellen, waar je gegevens van hetzelfde soort opslaat.
2. Neem bijvoorbeeld de onderstaande tabel.

500149969	Amrane		S	8	8	8	8	8	8	8
500104970	Badulkhan		MI	3	3	6				
500141978	Biyik		A	3	4	6	7	8	8	8
500517068	Boon		JP	2	3	3	4	4	5	6
500601077	Bruinsma		RJ	3	4	5				
500621378	Koerts		P	3	3	4	6	6	7	7
500623070	Kücük		A	3	4	8		8		
500626729	Loo	van der	R	7	12	12	12	12	12	12
500627672	Noppe		OL	3	4	4	6	7	8	7
500628610	Reding		IC	7	6	8	8	8	8	
500628817	Rooij	van	M	2	3	4	5	4	6	7
500628820	San José	de	JA	3	4	4	7	6	8	8

3. Wat zou de betekenis van deze *gegevens* kunnen zijn? Welke *informatie* staat hier?
4. Zonder context zijn de bovenstaande vragen niet te beantwoorden. Hierboven staan ruwe gegevens, die zo niets zeggen.
5. Pas wanneer de context meegegeven wordt, dan krijgt de tabel betekenis. De tabel zegt dat de student met nummer ***500149969*** alle opdrachten in week ***8*** van dit blok heeft afgetekend en dat de student met nummer ***500104970*** nog vier opdrachten moet aftekenen.

3.3 Hoe ontstaan databases?

1. Hieronder volgt een kort overzicht van de stappen, die men in het algemeen neemt om tot een geïmplementeerde database te komen. Later in de cursus gaan we dieper in op alle stappen.

   a. Het begin is dat een gebruiker/organisatie de wens heeft om gegevens op een gestructureerde manier vast te leggen.

   b. Dan volgt een inventarisatie van de informatiebehoefte.

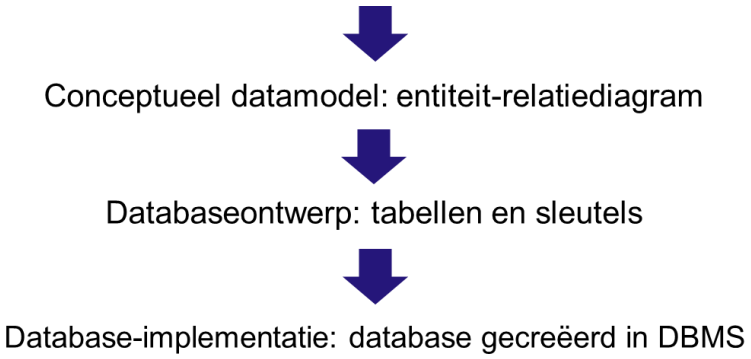
   c. Daar wordt uit afgeleid welke gegevens van belang zijn om aan de informatiebehoefte te voldoen.

   d. De gegevens worden op een logische wijze geordend en gegroepeerd in zogeheten *entiteittypen*. Ook de *relaties* tussen de diverse entiteittypen worden bepaald. Dit is het *conceptuele datamodel* en dit wordt meestal weergegeven in een *Entiteit Relatie Diagram* (*ERD*).

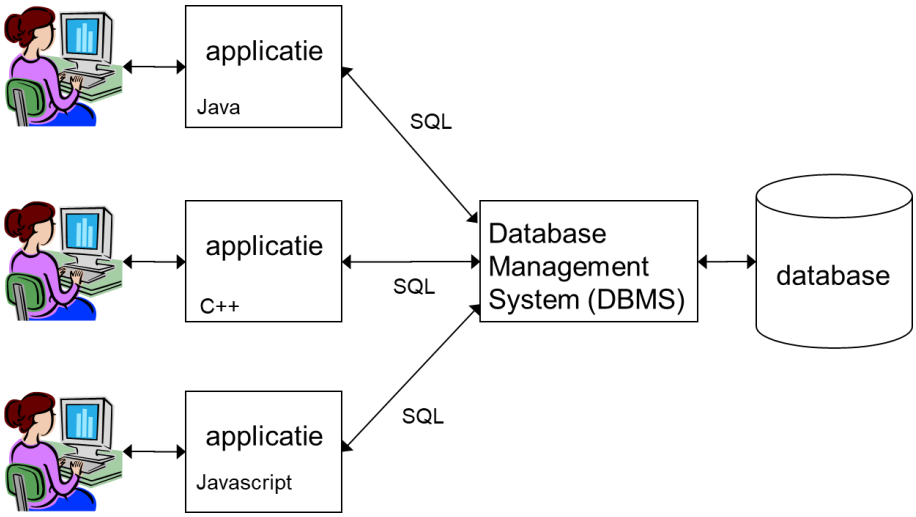
   e. **Let op**: dit is een model van de werkelijkheid, gezien vanuit de gebruiker en diens informatiebehoefte.

   f. De entiteittypen worden geconcretiseerd in *relaties* (*tabellen*) en een *primaire sleutel*/*identifier* wordt bepaald. Eventuele noodzakelijke tussentabellen worden nu bepaald. Er is sprake van een *logisch database ontwerp*.

   g. Tot slot wordt de database geïmplementeerd in een *databasemanagementsysteem* (*DBMS*). Hierbij worden de *datatypes* van de attributen bepaald. We hebben het dan over een *relationele database*.



2. Een *DBMS* is een schil om de database heen. Een *DBMS* regelt heel veel om, onder andere, de kwaliteit en de integriteit van de database te bewaken. Een *DBMS* wordt meestal via een applicatie benaderd:



3. De volgende terminologie zullen we de komende lessen veel gebruiken. Hier geven we de Engelse en Nederlandse termen in een tabel. Later in de cursus gaan we dieper op hun betekenis in.

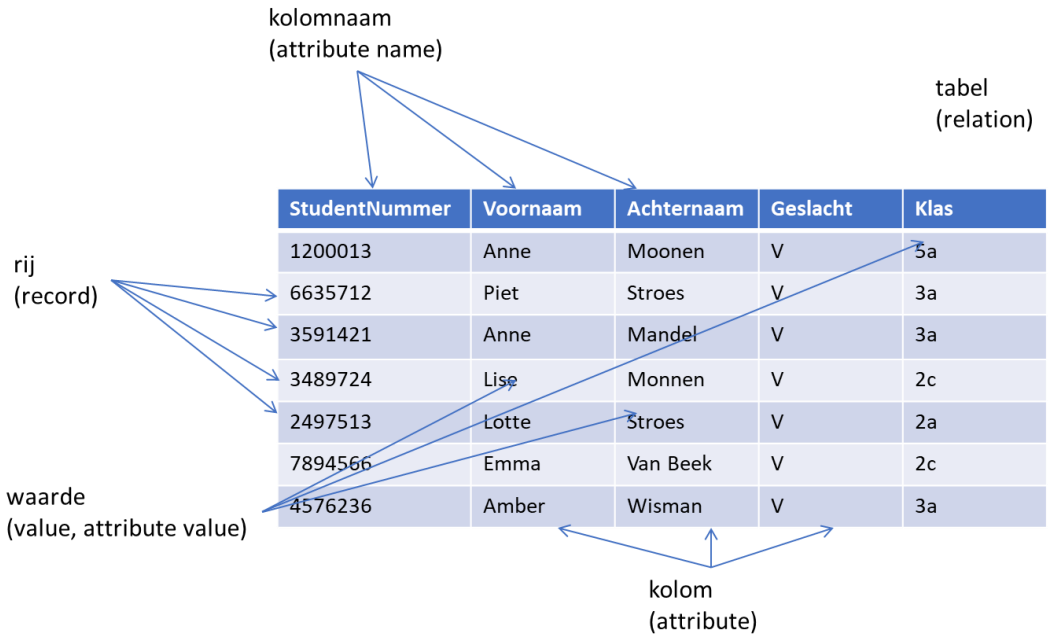
Engels	Nederlands
Relation	Relatie
Primary Key	Primaire sleutel
Candidate Key	Kandidaatsleutel
Composite Key	Samengestelde sleutel
Surrogate Key	Surrogaatsleutel
Foreign Key	Vreemde sleutel
Referential integrity	Referentiële integriteit
Functional Dependency	Functionele afhankelijkheid
Determinant	Determinant
Normal Form	Normaalvorm

4. Een *relatie* is een verzameling rijen. De volgorde van de rijen is niet van belang en er mogen geen identieke rijen voorkomen. De term *tabel* wordt bij databases gebruikt als synoniem voor *relatie*.

5. Bekijk de video behorende bij onderdeel *Relational databases* uit de les ***1. Database Foundations***.

6. In een tabel gebruiken we de volgende terminologie:

A relation is a collection of rows. The order of the rows does not matter, and duplicate rows are not allowed. In databases, the term "table" is used as a synonym for "relation."



StudentNummer	Voornaam	Achternaam	Geslacht	Klas
1200013	Anne	Moonen	V	5a
6635712	Piet	Stroes	V	3a
3591421	Anne	Mandel	V	3a
3489724	Lise	Monnen	V	2c
2497513	Lotte	Stroes	V	2a
7894566	Emma	Van Beek	V	2c
4576236	Amber	Wisman	V	3a

7. Een *rij* bevat gegevens van één object/entiteit/ding. In het bovenstaande bevat één rij gegevens van een student.

8. Een *kolom* bevat gegevens van dezelfde soort. De kolom met naam ***Voornaam*** bevat dus allemaal voornamen.

3.4 Voorbeeld Pizzabezorging

1. Bekijk het volgende voorbeeld over de bezorging van pizza's.

code vestiging	plaats	code bezorger	naam bezorger	nummer voertuig	soort voertuig	brandstof
Al1	Almere	FHE TDI	F. Heerlen T. Dieters	1064 1065 1066	Scooter Fiets Scooter	Elektrisch Benzine
Al2	Almere	KAB RBE	K. Abma R. Berendse	1029	Scooter	Benzine
H1	Huizen	HCO TDI	H. L. Cornelisse T. Dieters	1034 1035	Bestelbus Scooter	Benzine Elektrisch
N1	Naarden	KER JEE RAH	K. Erdman J. van Eerde R. Ahmammout	1027 1028	Scooter Scooter	Benzine Elektrisch
B1	Bussum	KAB	K. Abma	1087	Bestelbus	Diesel

2. Dit is een typisch voorbeeld van hoe men – meestal met behulp van een werkblad/spreadsheet – gegevens vastlegt. Er zijn echter flink wat nadelen.

   a. De tabel suggereert dat ***F. Heerlen*** de naam van een bezorger is, die altijd hoort bij code vestiging ***Al1***. Hoe leggen we vast dat ***F. Heerlen*** niet altijd bij code vestiging ***Al1*** hoort, maar – bijvoorbeeld – als invalkracht ook ingezet kan worden op andere plekken?

b. In de kolom ***naam bezorger*** komen sommige namen meer dan één keer voor. Wat als bijvoorbeeld ***Dieters*** twee keer verschillend is gespeld? Wat is dan de correcte schrijfwijze?

c. Kijk naar de laatste twee kolommen in de rij, die begint met ***H1***. De tabel suggereert dat de ***bestelbus*** op ***benzine*** rijdt en dat de ***scooter elektrisch*** is. Hiervoor is de volgorde van de rijen van belang en deze volgorde mag dus niet veranderen.

d. Conclusie: deze manier van vastleggen is ongestructureerd en het onderhoud is lastig.

3. Welke tabellen zou je hier onderscheiden (oftewel, hoeveel logische groepen – ook wel *entiteiten* genoemd – zie je hier)? Denk hier even over na voordat je verder gaat.

4. In de bovenstaande tabel zeggen de eerste twee kolommen iets over een ***vestiging*** (***code*** en ***naam***), dan volgen er twee kolommen met gegevens over ***bezorgers*** (ook weer ***code*** en ***naam***) en de laatste drie bevatten gegevens over ***voertuigen*** (***nummer***, ***soort*** en ***brandstof***). Daar kun je de volgende tabellen van maken:

Vestiging

code vestiging	plaats
Al1	Almere
Al2	Almere
H1	Huizen
N1	Naarden
B1	Bussum

Bezorger

code bezorger	naam bezorger
FHE	F. Heerlen
TDI	T. Dieters
KAB	K. Abma
RBE	R. Berendse
HCO	H. L. Cornelisse
TDI	T. Dieters
KER	K. Erdman
JEE	J. van Eerde
RAH	R. Ahmammout
KAB	K. Abma

Voertuig

nummer voertuig	soort voertuig	brandstof
1064	Scooter	Elektrisch
1065	Fiets	
1066	Scooter	Benzine
1029	Scooter	Elektrisch
1034	Bestelbus	Benzine
1035	Scooter	Elektrisch
1027	Scooter	Benzine
1028	Scooter	Elektrisch
1087	Bestelbus	Diesel

5. **Let op**: de conventie in deze cursus is dat de tabelnaam begint met een hoofdletter en in het algemeen een zelfstandig naamwoord in het enkelvoud is. Kolomnamen (attribuutnamen) zijn ook enkelvoud en met kleine letters.

6. Merk op dat er nu enkele rijen dubbel voorkomen.

Vestiging

code vestiging	plaats
Al1	Almere
Al2	Almere
H1	Huizen
N1	Naarden
B1	Bussum

Bezorger

code bezorger	naam bezorger
FHE	F. Heerlen
TDI	T. Dieters
KAB	K. Abma
RBE	R. Berendse
HCO	H. L. Cornelisse
TDI	T. Dieters
KER	K. Erdman
JEE	J. van Eerde
RAH	R. Ahmammout
KAB	K. Abma

Voertuig

nummer voertuig	soort voertuig	brandstof
1064	Scooter	Elektrisch
1065	Fiets	
1066	Scooter	Benzine
1029	Scooter	Elektrisch
1034	Bestelbus	Benzine
1035	Scooter	Elektrisch
1027	Scooter	Benzine
1028	Scooter	Elektrisch
1087	Bestelbus	Diesel

7. Dat mag niet in een tabel en daarom is dit het eindresultaat:

Vestiging

code vestiging	plaats
Al1	Almere
Al2	Almere
H1	Huizen
N1	Naarden
B1	Bussum

Bezorger

code bezorger	naam bezorger
FHE	F. Heerlen
TDI	T. Dieters
KAB	K. Abma
RBE	R. Berendse
HCO	H. L. Cornelisse
KER	K. Erdman
JEE	J. van Eerde
RAH	R. Ahmammout

Voertuig

nummer voertuig	soort voertuig	brandstof
1064	Scooter	Elektrisch
1065	Fiets	
1066	Scooter	Benzine
1029	Scooter	Elektrisch
1034	Bestelbus	Benzine
1035	Scooter	Elektrisch
1027	Scooter	Benzine
1028	Scooter	Elektrisch
1087	Bestelbus	Diesel

8. Wat nu echter ontbreekt is dat we niet meer weten welke bezorger bij welke vestiging hoort en welk voertuig bij welke vestiging hoort. Dat was (impliciet) wel uit de oorspronkelijke tabel te halen, maar die relaties zijn nu weggevallen.

9. Bekijk de video *Modeling and planning a database* in de les ***2. Tables***.

3.5 Primaire sleutels

1. Bekijk de twee volgende twee video's: *Keys and unique values* in de les ***1. Database Foundations*** en *Primary and foreign keys* in de les ***2.Tables***. We zullen het later hebben over de genoemde *vreemde sleutels* (*foreign keys*).

2. Een (*primaire*) *sleutel* wordt gebruikt om een rij uniek te definiëren, zodat iedereen weet waar je het over hebt. Dat is een kolom, waarvan je weet dat iedere waarde maar één keer voorkomt (*uniciteit*) en dat er altijd een waarde is (*totaliteit*). In de praktijk blijkt dat er heel vaak gekozen wordt voor een eigen gemaakte code (dat kan een nummer zijn, zoals je burgerservicenummer (BSN), klantnummer, factuurnummer, registratienummer, boeknummer (ISBN), enzovoorts, of een code (denk hierbij aan je *HvA-ID*).

3.6 Primaire sleutels bij Pizzabezorging

1. Bedenk eerst zelf hoe je in iedere tabel met behulp van de waarde in een kolom een hele rij zou kunnen identificeren. Kijk dan naar de uitwerking hieronder.

2. Het zal je niet verbazen dat in de gegeven tabellen de kolommen met een code of nummer identificerend zijn voor een rij.

Vestiging

code vestiging	plaats
Al1	Almere
Al2	Almere
H1	Huizen
N1	Naarden
B1	Bussum

Bezorger

code bezorger	naam bezorger
FHE	F. Heerlen
TDI	T. Dieters
KAB	K. Abma
RBE	R. Berendse
HCO	H. L. Cornelisse
KER	K. Erdman
JEE	J. van Eerde
RAH	R. Ahmammout

Voertuig

nummer voertuig	soort voertuig	brandstof
1064	Scooter	Elektrisch
1065	Fiets	
1066	Scooter	Benzine
1029	Scooter	Elektrisch
1034	Bestelbus	Benzine
1035	Scooter	Elektrisch
1027	Scooter	Benzine
1028	Scooter	Elektrisch
1087	Bestelbus	Diesel

3. Een *primaire sleutel* wil dus zeggen dat we een rij uniek kunnen identificeren (bepalen) op basis van de gegeven waarde in de sleutelkolom. Zo is het meteen duidelijk waar we het over hebben als we verwijzen naar de vestiging met code ***H1***, de bezorger met code ***RBE*** en het voertuig met nummer ***1034***. Al deze verwijzingen zijn uniek en dus kan er geen misverstand ontstaan.

4. Daarentegen is het onduidelijk als we het hebben over de vestiging in ***Almere***. Hebben we het dan over ***Al1*** of ***Al2***? Hetzelfde geldt voor het voertuig van de soort ***Scooter***. Over welke van de zes scooters in de tabel hebben we het dan?

5. **Let op**: de naam van de bezorger is (nu) wel uniek. Er kan dan ook geen misverstand over ontstaan. Toch is het geen goed idee om deze kolom als primaire sleutel te gebruiken, omdat het denkbaar is dat er ooit twee bezorgers met dezelfde naam **kunnen** zijn. Bij het ontwerp van je database moet je ook denken aan situaties, die later kunnen ontstaan.

6. Bedenk dat de belangrijkste eisen aan een *primaire sleutel* zijn dat de waarde uniek moet zijn (*uniciteitsregel*) en dat er altijd een waarde is (*totaliteitsregel*). Bekijk het volgende voorbeeld.

Totaliteitsregel

Uniciteitsregel

nummer voertuig	soort voertuig	brandstof
1064	Scooter	Elektrisch
1065	Fiets	
1066	Scooter	Benzine
1029	Scooter	Elektrisch
	Bestelbus	Benzine
1035	Scooter	Elektrisch
1087	Scooter	Benzine
1028	Scooter	Elektrisch
1087	Bestelbus	Diesel

7. In dit voorbeeld zie je het belang van de twee eisen. Als het nummer niet uniek is (***1087*** komt twee keer voor), dan weten we nog steeds niet exact over welk voertuig we het hebben: de scooter of de bestelbus. Als er geen nummer is, dan wordt het lastig om ernaar te verwijzen.

3.7 Samengestelde sleutels, kandidaatsleutels en surrogaatsleutels

1. Soms is het niet mogelijk om een rij uniek te definiëren op basis van één kolom. Bijvoorbeeld in het geval van een adres. Er zijn twee kolommen nodig. De postcode is niet uniek, het huisnummer is niet uniek, maar de combinatie van de postcode en het huisnummer is wel uniek. De *primaire sleutel* (enkelvoud!) bestaat dan uit twee kolommen. We zeggen dat dit een *samengestelde sleutel* is (*composite key*).

straat	huisnr	postcode	plaats
Duivendrechtsekade	36	1096 AH	Amsterdam
Duivendrechtsekade	38	1096 AH	Amsterdam
Ruychrocklaan	18	2597 EN	Den Haag
Rijnstraat	36	1079 DG	Amsterdam
Dorpsstraat	3	7844 RF	Ons Dorp
Mauritskade	431	2529 WK	Den Haag
Dorpsstraat	3	5428 XV	Appelscha

2. Een *kandidaatsleutel* (*candidate key*) is een kolom of kolomcombinatie die een rij uniek identificeert.
   a. Het kan zijn dat er in een tabel meer dan één *kandidaatsleutel* is.
   b. Eén van de *kandidaatsleutels* wordt geselecteerd als primaire sleutel.
   c. Elke tabel heeft precies één *primaire sleutel*, die een *samengestelde sleutel* mag zijn.
   d. De ideale *primaire sleutel* is kort, numeriek en wijzigt niet.
3. In de praktijk kiest men in bijna alle gevallen voor een *surrogaatsleutel* (*surrogate key*) zoals een klantnummer, BSN, registratienummer, enzovoorts. Een *surrogaatsleutel* is een kunstmatige kolom die aan een tabel wordt toegevoegd om als *primaire sleutel* te fungeren. Een *surrogaatsleutel*:
   a. wordt gebruikt als *kandidaatsleutels* groot of onhandig zijn;
   b. is kort, numeriek en wijzigt niet (de ideale *primaire sleutel*!);
   c. wordt veelal automatisch toegekend door het *DBMS*;
   d. is vaak een kunstmatige waarde zonder betekenis voor de gebruikers;
   e. wordt meestal verborgen in applicaties en overzichten.

3.8 Opdrachten

1. Doe opdracht **Inleiding.1 Bands** (niveau: basis).
2. Doe opdracht **Inleiding.2 Landen** (niveau: basis).

4 CONCEPTUEEL DATAMODEL (ERD)

1. Hieronder geven we de oorspronkelijke databehoefte voor de pizzabezorging weer.

code vestiging	plaats	code bezorger	naam bezorger	nummer voertuig	soort voertuig	brandstof
Al1	Almere	FHE TDI	F. Heerlen T. Dieters	1064 1065 1066	Scooter Fiets Scooter	Elektrisch Benzine
Al2	Almere	KAB RBE	K. Abma R. Berendse	1029	Scooter	Benzine
H1	Huizen	HCO TDI	H. L. Cornelisse T. Dieters	1034 1035	Bestelbus Scooter	Benzine Elektrisch
N1	Naarden	KER JEE RAH	K. Erdman J. van Eerde R. Ahmammout	1027 1028	Scooter Scooter	Benzine Elektrisch
B1	Bussum	KAB	K. Abma	1087	Bestelbus	Diesel

2. Eerder in deze les hebben we de databehoefte opgesplitst in drie tabellen en voor iedere tabel een primaire sleutel bepaald.

Vestiging

code vestiging	plaats
Al1	Almere
Al2	Almere
H1	Huizen
N1	Naarden
B1	Bussum

Bezorger

code bezorger	naam bezorger
FHE	F. Heerlen
TDI	T. Dieters
KAB	K. Abma
RBE	R. Berendse
HCO	H. L. Cornelisse
KER	K. Erdman
JEE	J. van Eerde
RAH	R. Ahmammout

Voertuig

nummer voertuig	soort voertuig	brandstof
1064	Scooter	Elektrisch
1065	Fiets	
1066	Scooter	Benzine
1029	Scooter	Elektrisch
1034	Bestelbus	Benzine
1035	Scooter	Elektrisch
1027	Scooter	Benzine
1028	Scooter	Elektrisch
1087	Bestelbus	Diesel

3. We hebben ook al geconstateerd dat we nu niet meer weten welke bezorger bij welke vestiging hoort en welk voertuig bij welke vestiging hoort!

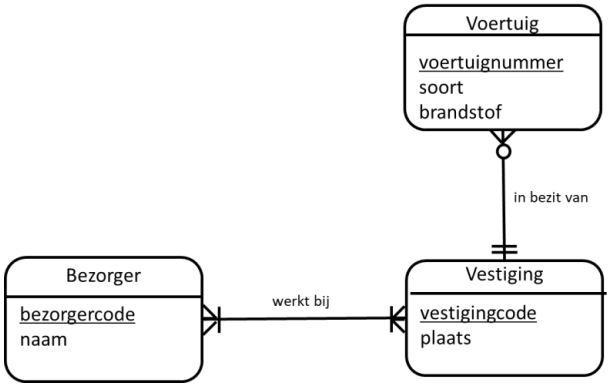
4. Een *ERD* – *Entiteit Relatie Diagram* of *Entity Relationship Diagram* geeft behalve de entiteiten (tabellen) ook de relaties (relationships) tussen de entiteiten weer. Bedenk ook dat een ERD een model van de werkelijkheid is, gezien vanuit de gebruiker. Het wordt gebruikt als bouwplan of blauwdruk voor een database ontwerp. De belangrijkste eisen aan een *ERD* zijn:

   a. *Integriteit* (het model klopt met werkelijkheid, het is conform de informatiebehoefte en het komt overeen met bedrijfsregels).

   b. *Geen redundantie* (elk gegeven wordt maar één keer weergeven).

   c. *Consistentie* (het model bevat geen tegenstrijdigheden).

5. Een relatie tussen twee entiteiten kan *1-op-veel*, *veel-op-veel* en *1-op-1* zijn.
   a. Voorbeeld van een *1-op-veel* relatie. De relatie tussen ***Vestiging*** en ***Voertuig*** is *1-op-veel* omdat een specifieke vestiging veel voertuigen kan bezitten en een specifiek voertuig altijd bij precies één vestiging hoort.
   b. Voorbeeld van een *veel-op-veel* relatie. De relatie tussen ***Bezorger*** en ***Vestiging*** is *veel-op-veel* omdat een specifieke bezorger bij meer (veel) vestigingen kan werken en bij een specifieke vestiging meer (veel) bezorgers kunnen werken.
   c. Voorbeeld van een *1-op-1* relatie (niet gebaseerd op ***Pizzabezorging***). Een medewerker heeft maximaal één leasefiets en een leasefiets hoort bij precies één medewerker.
6. Daarnaast kan de relatie *optioneel* of *verplicht* zijn.
   a. Voorbeelden van *optionele* relaties zijn: een voertuig hoeft niet in bezit van een vestiging te zijn of een vestiging hoeft geen voertuigen te hebben.
   b. Voorbeelden van *verplichte* relaties zijn: een voertuig moet altijd bij een vestiging horen en een vestiging moet altijd een (of meer) voertuigen hebben.
7. Hieronder volgt een voorbeeld van een *ERD* voor ***Pizzabezorging***. De toelichting volgt daaronder.



4.1 Weergave van de tabellen in een ERD

1. *Tabellen* worden in een *ERD* weergegeven met een (afgeronde) rechthoek.
2. Een *tabel* bevat informatie over het *entiteittype*. Dat is een representatie van een fundamenteel belang voor een organisatie en iets waarover de gebruiker gegevens wil vastleggen. Dit kan zijn een
   a. Object (bijvoorbeeld Artikel, Vak, Boek).
   b. Plaats (bijvoorbeeld: Vestiging, Locatie, Land).
   c. Persoon (bijvoorbeeld: Klant, Student, Werknemer).
   d. Gebeurtenis (bijvoorbeeld: Inschrijving, Bestelling, Deelname).
2. Bovenin de rechthoek (de eerste rij) staat de naam van de tabel. Dit is een entiteittype (enkelvoud en beginnend met een hoofdletter)

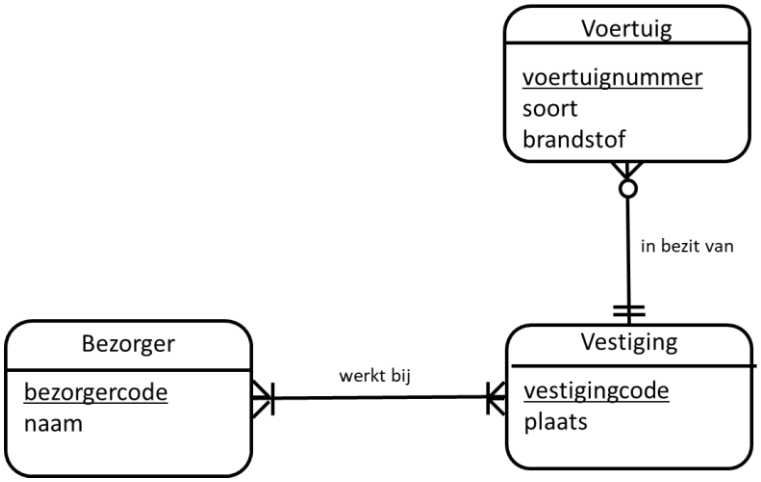
3. Een tabel bevat ook informatie over de *attributen*. Dat zijn de kenmerken van een entiteittype, die de eigenschappen van een entiteit beschrijven. Een eigenschap kan op te splitsen zijn in betekenisvolle onderdelen (bijvoorbeeld: een naam is op te splitsen in voorletters, tussenvoegsel, achternaam).
4. In de tweede rij van een rechthoek zie je de attributen (enkelvoud en met kleine letters). Het onderstreepte attribuut is de *primaire sleutel*. Als de *primaire sleutel samengesteld* is, dan zijn alle *sleutelattributen* onderstreept.

4.2 Weergave van relaties in een ERD

1. *Relaties* geven een verband tussen twee entiteiten weer.
2. Maak per relatie twee zinnen:
   a. Formuleer iedere zin gezien vanuit één entiteit naar de gerelateerde entiteit.
   b. Geef in elke zin een *minimum kardinaliteit* en een *maximum kardinaliteit* aan door gebruik te maken van *nul of meer*, *één of meer*, *nul of één*, *precies één*.
   c. De *minimum kardinaliteit* is altijd ***0*** (weergegeven met een cirkel) of ***1*** (weergegeven met een streep), de *maximum kardinaliteit* is altijd ***1*** of ***meer*** (weergegeven met de kraaienpoot).
   d. De kardinaliteiten worden weergegeven aan de 'passieve' kant. We lichten dit toe aan de hand van enkele voorbeelden.

4.3 Voorbeeld Pizzabezorging

1. Formuleer twee zinnen, die de relatie tussen ***Bezorger*** en ***Vestiging*** beschrijven.
   a. Een bezorger werkt op *1 of meer* vestigingen.
   b. Op een vestiging werken *1 of meer* bezorgers.
2. Formuleer twee zinnen, die de relatie tussen ***Voertuig*** en ***Vestiging*** beschrijven.
   a. Een vestiging bezit *0 of meer* voertuigen.
   b. Een voertuig hoort bij *precies 1* vestiging.
3. **Let op**: tussen twee entiteiten (bijvoorbeeld ***Voertuig*** en ***Vestiging***) kan het zo zijn, dat de ene relatie wel verplicht is (een voertuig moet bij een vestiging horen), terwijl de omgekeerde relatie optioneel kan zijn (een vestiging hoeft geen voertuig te hebben).
4. Geef elke relatie in het *ERD* een zinvolle beschrijving. In het algemeen bevat de relatie een stuk tekst dat één van de twee richtingen toelicht.
   a. Een bezorger **werkt bij** een vestiging.
   b. Een voertuig is **in bezit van** een vestiging.
5. Hierbij nogmaals het *ERD* van ***Pizzabezorging***.



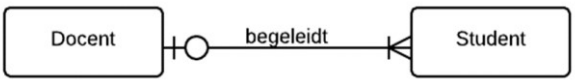
6. De interpretatie is dan:
   a. Een vestiging bezit minimaal ***0*** en maximaal ***meer*** voertuigen.
   b. En de andere kant op: een voertuig hoort bij minimaal ***1*** en maximaal ***1*** (dus is het ***precies 1***) vestiging.
   c. De relatie tussen ***Vestiging*** en ***Voertuig*** is dan *1-op-veel* (*1 : n*) aangezien er aan de ene kant maximaal meer (*veel*) voertuigen en aan de andere kant maximaal *één* vestiging is.
   d. Bij een vestiging werkt minimaal ***1*** en maximaal ***meer*** bezorgers.
   e. En de andere kant op: een bezorger werkt bij minimaal ***1*** en maximaal ***meer*** vestigingen.
   f. De relatie tussen ***Vestiging*** en ***Bezorger*** is dan *veel-op-veel* (*n : m*) aangezien er aan de ene kant maximaal meer (*veel*) bezorgers zijn en aan de andere kant maximaal meer (*veel*) vestigingen zijn.
7. Meer voorbeelden komen hieronder!

4.4 Voorbeelden van relaties

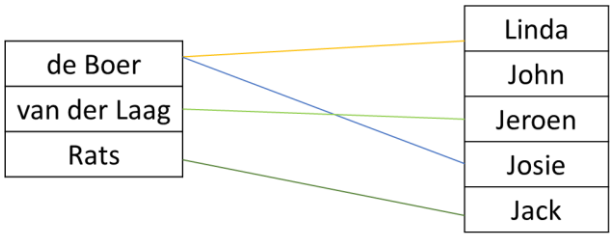
1. **Voorbeeld 1**

2. Bedenk nu zelf twee zinnen waarmee je de relatie tussen ***Docent*** en ***Student*** beschrijft.
3. **Uitwerking**:
   a. Een docent begeleidt 0 of meer studenten (optioneel).
   b. Een student wordt door precies één docent begeleid (verplicht).
   c. De relatie is *1-op-veel*.

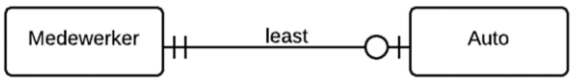
4. **Voorbeeld 2**



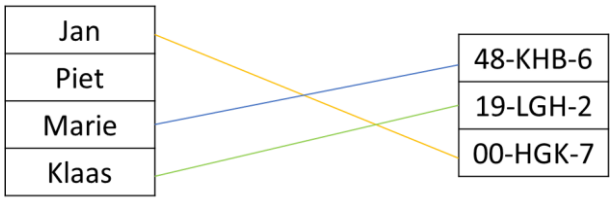
5. Bedenk nu zelf twee zinnen waarmee je de relatie tussen ***Docent*** en ***Student*** beschrijft.
6. **Uitwerking**:
   a. Een docent begeleidt 1 of meer studenten (verplicht).
   b. Een student wordt begeleid door 0 of 1 docent (optioneel).
   c. De relatie is *1-op-veel*.
7. Hieronder volgt een voorbeeld van de manier waarop de tabellen gevuld kunnen zijn.



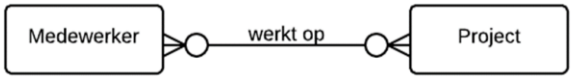
8. **Toelichting**:
   a. Iedere docent (links) begeleidt minimaal 1 student (rechts). Bijvoorbeeld: docent ***de Boer*** begeleidt studenten ***Linda*** en ***Josie***.
   b. Niet iedere student heeft een begeleider. Bijvoorbeeld: ***John*** heeft geen begeleider. Studenten hebben nooit meer dan 1 begeleider.
9. **Voorbeeld 3**



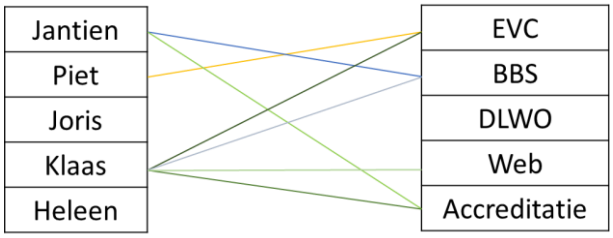
10. Bedenk nu zelf twee zinnen waarmee je de relatie tussen ***Medewerker*** en ***Auto*** beschrijft.
11. **Uitwerking**:
    a. Een medewerker leaset 0 of 1 auto (optioneel).
    b. Een auto wordt geleaset door precies één medewerker (verplicht).
    c. De relatie is *1-op-1*.
12. Hieronder volgt een voorbeeld van de manier waarop de tabellen gevuld kunnen zijn.



13. **Toelichting**:
    a. Niet iedere medewerker (links) heeft een lease-auto. Bijvoorbeeld: ***Piet*** heeft geen lease-auto. Medewerkers hebben nooit meer dan 1 lease-auto.
    b. Iedere lease-auto wordt geleaset door precies 1 medewerker.
14. **Voorbeeld 4**



15. Bedenk nu zelf twee zinnen waarmee je de relatie tussen ***Medewerker*** en ***Project*** beschrijft.
16. **Uitwerking**:
    a. Een medewerker werkt op 0 of meer projecten (optioneel).
    b. Op een project werken 0 of meer medewerkers (optioneel).
    c. De relatie is *veel-op-veel*.
17. Hieronder volgt een voorbeeld van de manier waarop de tabellen gevuld kunnen zijn.



18. **Toelichting**:
    a. Niet iedere medewerker (links) werkt op een project (rechts). Bijvoorbeeld: ***Joris*** werkt niet op een project.
    b. Medewerkers kunnen op 1 of meer projecten werken. Bijvoorbeeld: ***Piet*** werkt alleen op het ***EVC***, terwijl ***Klaas*** op de projecten ***EVC***, ***BBS***, ***Web*** en ***Accreditatie*** werkt.
    c. Op niet ieder project werkt een medewerker. Bijvoorbeeld: op het project ***DLWO*** werken (nog) geen medewerkers.
    d. Op projecten kunnen 1 of meer medewerkers werken. Op de projecten ***EVC***, ***BBS*** en ***Accreditatie*** werken 2 medewerkers, terwijl er op het project ***Web*** maar 1 medewerker werkt.

4.5 Stappenplan voor het maken van een ERD

1. De volgende stappen zet je om te komen tot een *ERD*.
    a. Stap 1: selecteer voorbeelden uit de werkelijkheid.
    b. Stap 2: leg uit in natuurlijke taal.
    c. Stap 3: bepaal *entiteittypes*.
    d. Stap 4: bepaal *attributen* en *identifiers*.
    e. Stap 5: bepaal *relaties*.
    f. Stap 6: check, recheck, dubbelcheck.

4.6 Voorbeeld bibliotheek

1. We tonen de stappen aan de hand van een voorbeeld uit een bibliotheek.
2. **Stap 1: selecteer voorbeelden uit de werkelijkheid**. Het doel is om te komen tot een zo volledig mogelijk beeld van de informatiebehoefte en de afbakening daarvan. Na overleg met de gebruiker blijkt dat men de gegevens omtrent boeken uit de bibliotheek wil vastleggen. De informatiebehoefte beperkt zich tot de gegevens, die getoond worden op de kaart.

Auteur(s): G.F. de Boer & N.Y. de Vries	
Titel:	**Het gegevensmodellenboek**
Uitgever:	**Cpress, Rotterdam**
ISBN:	**1111-0**
Geschikt voor:	**Jeugd**
Exemplaren:	**1111-0.1 (1998)** **1111-0.2 (2001)**

3. **Stap 2: leg uit in natuurlijke taa**l. Het doel is om de informatiebehoefte te kunnen analyseren en te concretiseren. Dit gebeurt altijd door middel van interviews of workshops met de eindgebruikers. Sommige aspecten kunnen rechtstreeks van de kaart worden afgeleid, andere aspecten worden pas duidelijk na een overleg. Neem echter nooit iets aan en verifieer het altijd met de eindgebruiker.
4. Bedenk eerst zelf enkele zinnen waarmee je de gegevens op de kaart kunt omschrijven.
5. De volgende aspecten zijn naar voren gekomen na een overleg:
    a. Van elk boek worden de titel en ISBN vastgelegd.
    b. Van elk boek worden de voorletters en achternaam van de auteur(s) vastgelegd.
    c. Van elk boek worden de naam en vestigingsplaats van de uitgever vastgelegd.

d. Ook uitgevers, waarvan de bibliotheek geen boeken heeft, kunnen worden vastgelegd.

e. Van elk boek wordt de doelgroep vastgelegd.

f. Er wordt altijd per boek maar één doelgroep vastgelegd.

g. Ieder exemplaar van een boek krijgt een exemplaarnummer en jaar van uitgifte.

h. Het eerste deel van het exemplaarnummer is altijd gelijk aan het ISBN.

i. In een exemplaarnummer wordt het ISBN gevolgd door een punt en een uniek volgnummer.

j. Van een boek kunnen meerdere exemplaren in de bibliotheek aanwezig zijn.

6. **Stap 3: bepaal entiteittypes**. Het doel is om 'objecten' te bepalen waarover de gebruiker gegevens wil vastleggen.

7. Bedenk eerst zelf welke objecten (dingen) je hier ziet.

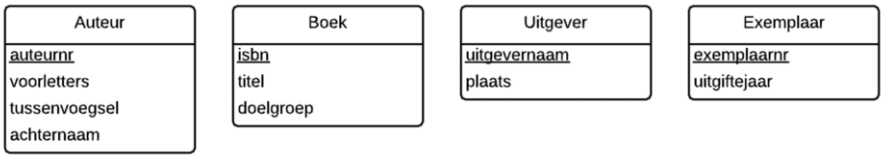
8. Aan de hand van de bovenstaande zinnen komen wij tot de volgende objecten: ***Boek***, ***Auteur***, ***Uitgever*** en ***Exemplaar***.

9. **Stap 4: bepaal attributen en identifiers**. Het doel is om vast te leggen welke gegevens er per object vastgelegd moeten worden en hoe we ieder object op een unieke wijze kunnen identificeren.

10. Bedenk eerst zelf welke *attributen* je onderscheidt en welke *identifier* je per object kunt gebruiken.

11. **Let op**: het is nu nog niet de bedoeling dat je al relaties gaat leggen (bijvoorbeeld welke auteur welk boek heeft geschreven). Het gaat om de 'kale' gegevens.

12. Hieronder volgt onze uitwerking:



13. **Merk op:**

a. We gebruiken voor de tabel ***Auteur*** een *surrogaatsleutel* als identifier. Namen zijn in het algemeen niet uniek en dus niet geschikt als identifier.

b. Van uitgevers verwachten we dat de naam wel uniek is, dat lijkt ons voor de uitgevers zelf wel zo handig.

14. **Stap 5: bepaal relaties**. Het doel is om de directe relaties tussen de verschillende objecten te leggen.

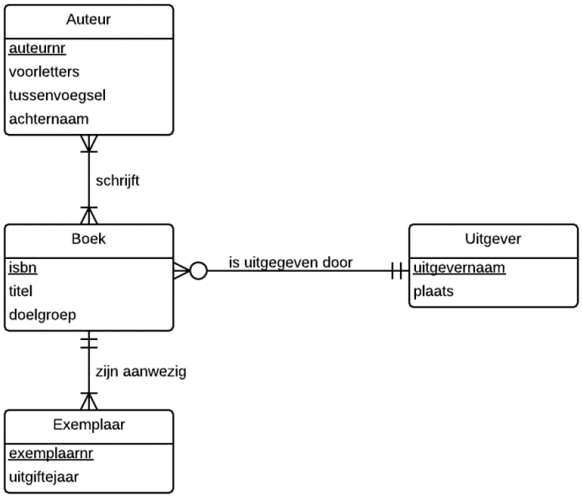
15. Bedenk eerst zelf welke relaties je ziet.

16. Het boek is het centrale object in dit voorbeeld. Een boek heeft een relatie met de andere drie tabellen. Relaties tussen bijvoorbeeld auteur en uitgever lopen via het boek. Wij komen tot de volgende conclusies:

a. Relatie ***Auteur*** en ***Boek***:
   i. Een ***Auteur*** heeft één of meer ***Boeken*** geschreven.
   ii. Een ***Boek*** wordt geschreven door één of meer ***Auteurs***.
   iii. De relatie is *veel-op-veel*.
b. Relatie ***Boek*** en ***Uitgever***:
   i. Een ***Boek*** wordt uitgegeven door precies één ***Uitgever.***
   ii. Een ***Uitgever*** geeft nul of meer ***Boeken*** uit.
   iii. De relatie is *1-op-veel*.
c. Relatie ***Boek*** en ***Exemplaar***
   i. Van een ***Boek*** zijn één of meer ***Exemplaren*** aanwezig.
   ii. Een ***Exemplaar*** hoort bij precies één ***Boek***.
   iii. De relatie is *1-op-veel*.

17. Stel nu eerst zelf het ERD op.

18. Dit is onze uitwerking.



19. **Stap 6: check, recheck, dubbelcheck**. Een *ERD* is zelden in één keer goed.
    a. Hebben alle gegevens in de voorbeelden een plek gekregen in het *ERD*?
    b. Heeft elke *entiteit* een *identifier*?
    c. Is elk *attribuut* precies één keer weergegeven?
    d. Heeft elke relatie een beschrijving?
    e. Is bij elke relatie aan beide kanten *minimumkardinaliteit* en *maximumkardinaliteit* correct weergegeven?
    f. Is de naamgeving consequent?
        i. Entiteiten: zelfstandig naamwoord, enkelvoud, startend met hoofdletter
        ii. Attributen: zelfstandig naamwoord, enkelvoud, kleine letters

4.7 Opdrachten

1. Doe opdracht **Conceptueel datamodel.1 Bands ERD** (niveau: basis);
2. Doe opdracht **Conceptueel datamodel.2 Landen ERD** (niveau: basis);
3. Doe opdracht **Conceptueel datamodel.3 Turnwedstrijden ERD** (niveau: basis).

5 RELATIONELE MODELLEN

5.1 Vreemde sleutels

1. We gebruiken de ***Pizzabezorging*** als voorbeeld. We hebben de volgende tabellen gemaakt op basis van de informatiebehoefte.

Vestiging

code vestiging	plaats
AI1	Almere
AI2	Almere
H1	Huizen
N1	Naarden
B1	Bussum

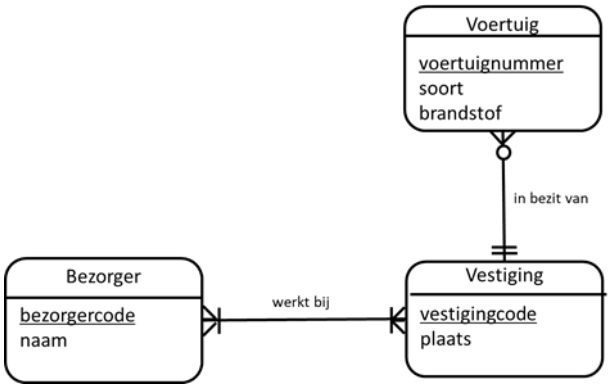
Bezorger

code bezorger	naam bezorger
FHE	F. Heerlen
TDI	T. Dieters
KAB	K. Abma
RBE	R. Berendse
HCO	H. L. Cornelisse
KER	K. Erdman
JEE	J. van Eerde
RAH	R. Ahmammout

Voertuig

nummer voertuig	soort voertuig	brandstof
1064	Scooter	Elektrisch
1065	Fiets	
1066	Scooter	Benzine
1029	Scooter	Elektrisch
1034	Bestelbus	Benzine
1035	Scooter	Elektrisch
1027	Scooter	Benzine
1028	Scooter	Elektrisch
1087	Bestelbus	Diesel

2. Vervolgens legden we de relaties tussen de verschillende objecten vast in een *ERD*.



3. We weten dat er een relatie is tussen ***Voertuig*** en ***Vestiging***, maar hoe leggen we die vast in de tabellen? Hoe leggen we vast dat het voertuig met nummer ***1064*** bij de vestiging ***AI1*** hoort? Of, omgekeerd, dat bij vestiging ***AI1*** de voertuigen ***1064***, ***1065*** en ***1066*** horen?

4. We leggen de relatie vast door een (of meer) extra kolom(men) toe te voegen aan een tabel. Dit heet dan een *vreemde sleutel*. Deze kolom of kolommen verwijzen naar de *primaire sleutel* in een andere tabel.

5. Bekijk de vier video's in de links hieronder.
   Creating relationships
   One to many relationships
   Many to many relationships
   One to one relationships

6. Het is van belang goed na te denken over de positie van de vreemde sleutel. De vreemde sleutel komt altijd in de tabel waarvan je zeker weet dat er maar naar één entiteit verwezen kan worden.

7. Bijvoorbeeld: een vestiging kan meer voertuigen bezitten, een voertuig kan maar bij één vestiging horen en daarom zetten we de vreemde sleutel in de tabel ***Voertuig***.

8. Bedenk zelf hoe de tabel ***Vestiging*** eruit moet zien als je daar een verwijzing naar ***Voertuig*** zou opnemen. Hoe weet je zeker dat je alle voertuigen bij een vestiging goed hebt vastgelegd? Hoeveel voertuigen kunnen er maximaal bij een vestiging horen?

5.2 Voorbeeld Pizzabezorging

1. De beste oplossing is dus door de tabel ***Voertuig*** uit te breiden met een extra kolom.

Vestiging

code vestiging	plaats
Al1	Almere
Al2	Almere
H1	Huizen
N1	Naarden
B1	Bussum

Voertuig

nummer voertuig	soort voertuig	brandstof	code vestiging
1064	Scooter	Elektrisch	Al1
1065	Fiets		Al1
1066	Scooter	Benzine	Al1
1029	Scooter	Elektrisch	Al2
1034	Bestelbus	Benzine	H1
1035	Scooter	Elektrisch	H1
1027	Scooter	Benzine	N1
1028	Scooter	Elektrisch	N1
1087	Bestelbus	Diesel	B1

2. De kolom ***code vestiging*** in de tabel ***Voertuig*** noemen we de *vreemde sleutel* (*foreign key*) en deze kolom verwijst naar de *primaire sleutel* in de tabel ***Vestiging***.

3. We kunnen nu uit de tabel ***Voertuig*** afleiden dat het voertuig met het nummer ***1064*** bij de vestiging ***Al1*** hoort.

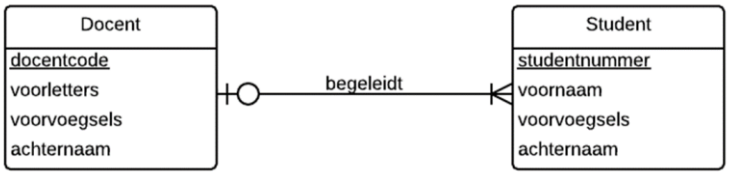
4. We kunnen hier ook uit afleiden, hoewel dat iets meer moeite kost, welke voertuigen er bij vestiging ***Al1*** horen. Hoe zou je dat kunnen doen?

5. De tabel met de *vreemde sleutel* heet de *kindtabel* (*child table*). In dit voorbeeld is de tabel ***Voertuig*** dus de *kindtabel*.

6. De tabel waarheen wordt verwezen heet de *oudertabel* (*parent table*). In dit voorbeeld is de tabel ***Vestiging*** dus de *oudertabel*.

5.3 Voorbeeld Docent en student

1. Kijk naar de onderstaande relatie tussen ***Docent*** en ***Student***.



2. Een docent kan veel (10, 20, 30?) studenten begeleiden. Het is ondoenlijk om in de tabel ***Docent*** verwijzingen op te nemen naar alle studenten, die een docent kan begeleiden. Je zou dan kolommen ***student1***, ***student2***, ***student3***, enzovoorts moeten opnemen. Maar waar eindig je, als er geen maximum bekend is?

3. Het is dus veel beter om bij de tabel ***Student*** een verwijzing op te nemen naar de ene docent die (optioneel) de student begeleidt.

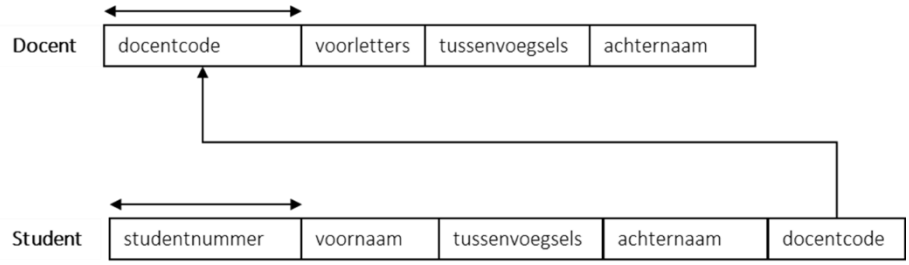
4. **Conclusie**: de *primaire sleutel* van de *oudertabel* (de één-kant van de *1-op-veel* relatie) wordt opgenomen als *vreemde sleutel* in de *kindtabel* (de veel-kant van de 1-*op-veel* relatie). In dit voorbeeld betekent dit dat er in de tabel ***Student*** een kolom ***docentcode*** bij komt. Dit is in de tabel ***Student*** een *vreemde sleutel*, die verwijst naar de *primaire sleutel* in de tabel ***Docent***.

5. **Let op**: je kunt zelf de naam van de kolom met de vreemde sleutel kiezen. Je kunt de oorspronkelijke naam (***docentcode***) nemen, maar je kunt ook de naam aanpassen en er bijvoorbeeld ***begeleider*** of ***begeleidercode*** voor nemen. Het wordt dan concreter wat de relatie inhoudt.

5.4 Strokendiagrammen

1. Een *ERD* is vooral bedoeld als conceptueel model. Je legt ermee vast dat er een relatie is tussen twee objecten, maar je legt niet vast hoe de relatie wordt geïmplementeerd. Om dit inzichtelijk te maken gebruiken we een *relationeel model*, in deze opleiding gebruiken we een *strokendiagram*.

2. Het *strokendiagram* van het voorbeeld hierboven ziet er als volgt uit:



3. **Toelichting**:

   a. Een ‘strook’ is een weergave van een tabel. Het bevat de tabelnaam en de attributen.

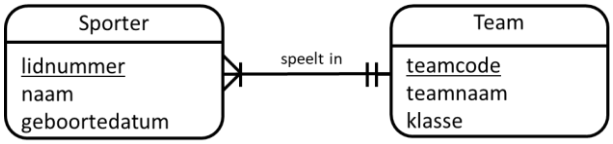
   b. Een tweezijdige pijl wordt boven de *primaire sleutel* getekend. Bij een *samengestelde sleutel* zal de pijl boven alle *sleutelattributen* weergegeven worden.

   c. Een eenzijdige pijl wijst van de *vreemde sleutel* in de verwijzende tabel naar de *primaire sleutel* van de tabel waarnaar verwezen wordt.

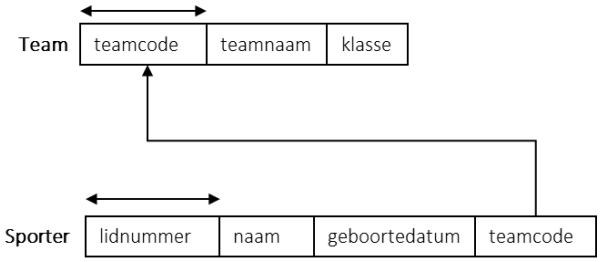
4. **Merk op** dat we in een *strokendiagram* niet meer kunnen zien of het gaat om een *1-op-1* relatie of een *1-op-veel* relatie. Ook zijn de *minimumkardinaliteiten* (0 of 1) en *maximumkardinaliteiten* (1 of meer) niet meer af te leiden.

5.5 Oefening

1. Maak zelf een strokendiagram behorende bij onderstaand *ERD*.



2. **Uitwerking**:



3. **Toelichting**: de vreemde sleutel komt bij de 'veel-kant' (***Sporter***) en verwijst naar de primaire sleutel aan de '1-kant' (***Team***).

5.6 Veel-op-veel relaties omzetten

1. Bij een *veel-op-veel* relatie is het niet mogelijk om een vreemde sleutel te plaatsen zodat er maar één mogelijkheid is. Per definitie zijn er aan allebei de kanten meer mogelijkheden.
2. Neem bijvoorbeeld de relatie tussen ***Vestiging*** en ***Bezorger*** van ***Pizzabezorging***.

Vestiging

code vestiging	plaats
Al1	Almere
Al2	Almere
H1	Huizen
N1	Naarden
B1	Bussum

Bezorger

code bezorger	naam bezorger
FHE	F. Heerlen
TDI	T. Dieters
KAB	K. Abma
RBE	R. Berendse
HCO	H. L. Cornelisse
KER	K. Erdman
JEE	J. van Eerde
RAH	R. Ahmammout

3. Een bezorger kan op meerdere vestigingen werken en op een vestiging werken meerdere bezorgers. Waar komt dan de vreemde sleutel? Het toevoegen van een vreemde sleutel in één van de tabellen leidt tot hetzelfde dilemma als bij ***Docent*** en ***Student*** hierboven.
4. Als we de tabel ***Vestiging*** uitbreiden met kolommen ***bezorger1***, ***bezorger*** 2, enzovoorts, hoe weten we dat we de tabel genoeg hebben uitgebreid?
5. En omgekeerd kunnen we de tabel ***Bezorger*** uitbreiden met alle mogelijke vestigingen. Maar wat als er een vestiging bijkomt?
6. **Let op**: het aanpassen van de structuur van een tabel (oftewel het toevoegen of verwijderen van attributen) is vele malen lastiger dan het toevoegen of verwijderen

van een rij binnen een tabel. Het is dus niet wenselijk om je database ontwerp steeds te moeten aanpassen.

5.7 Tussentabellen

1. De oplossing van het geschetste probleem is om te werken met een *tussentabel* of *koppeltabel*. Zo'n *tussentabel* wordt visueel tussen de twee bestaande tabellen geplaatst en koppelt het ene object met het andere door middel van de *primaire sleutels* van de beide objecten.
2. De *tussentabel* tussen ***Vestiging*** en ***Bezorger*** ziet er dan zo uit.

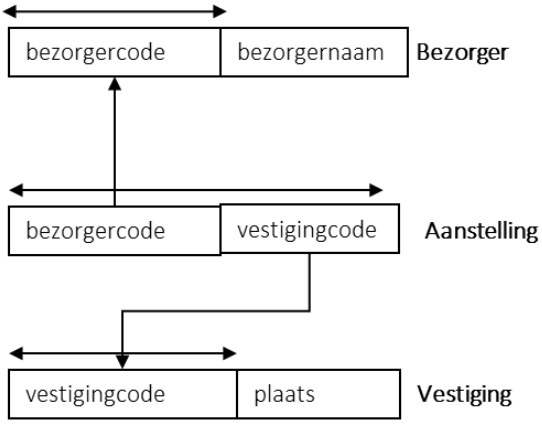
Aanstelling	
code vestiging	code bezorger
Al1	FHE
Al1	TDI
Al2	KAB
Al2	RBE
H1	HCO
H1	TDI
N1	KER
N1	JEE
N1	RAH
B1	KAB

3. Deze tabel koppelt vestiging ***Al1*** aan bezorgers ***FHE*** en ***TDI***. Omgekeerd wordt bezorger ***KAB*** gekoppeld aan vestigingen ***Al2*** en ***B1***.
4. Wat is nu de *primaire sleutel* van deze tabel? De kolom ***code vestiging*** is niet uniek. Ook de kolom ***code bezorger*** is niet uniek. De combinatie van de twee is dat wel. Het heeft namelijk geen zin om meer dan één keer vast te leggen dat een bezorger bij een vestiging hoort. Kortom, de *primaire sleutel* van de *koppeltabel* is een *samengestelde sleutel*.
5. In deze tabel zijn beide kolommen de *primaire sleutels* van de twee gekoppelde tabellen. Daarom zijn deze kolommen afzonderlijk een *vreemde sleutel* naar hun oorspronkelijke tabel.

5.8 Voorbeeld Pizzabezorging

1. Een deel van de *ERD* van ***Pizzabezorging*** ziet er zo uit:

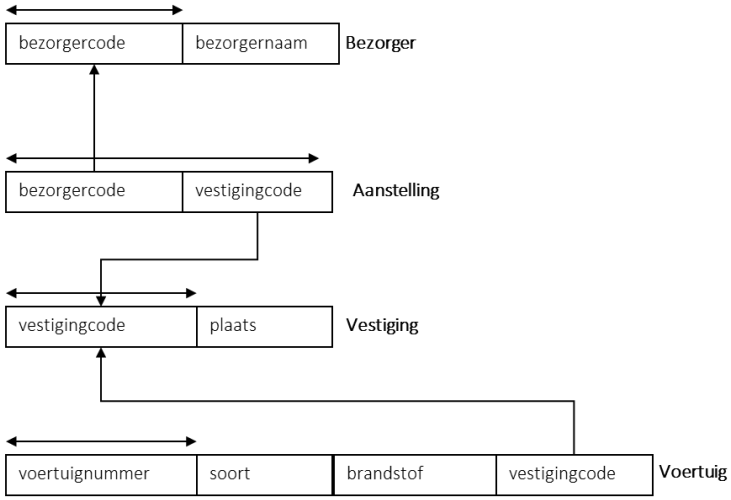
2. De implementatie van de *veel-op-veel* relatie ziet er dan zo uit in een strokenschema:



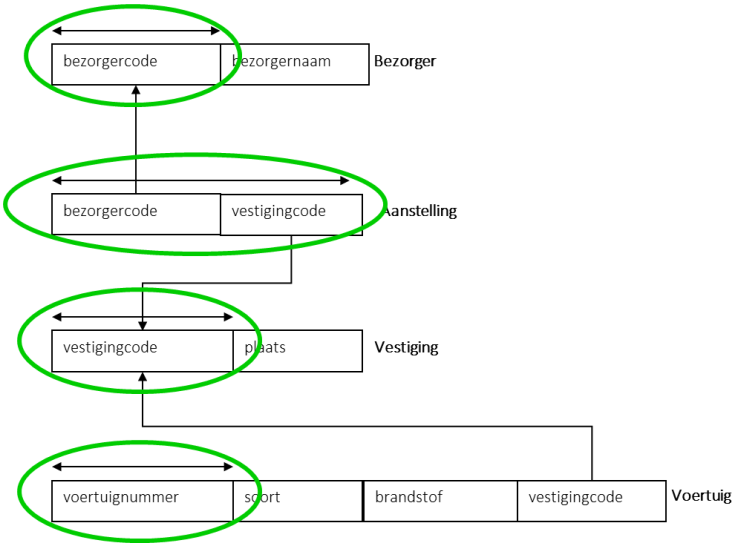
5.9 Oefening Pizzabezorging

1. Breid het bovenstaande strokenschema uit tot de hele casus van ***Pizzabezorging***.

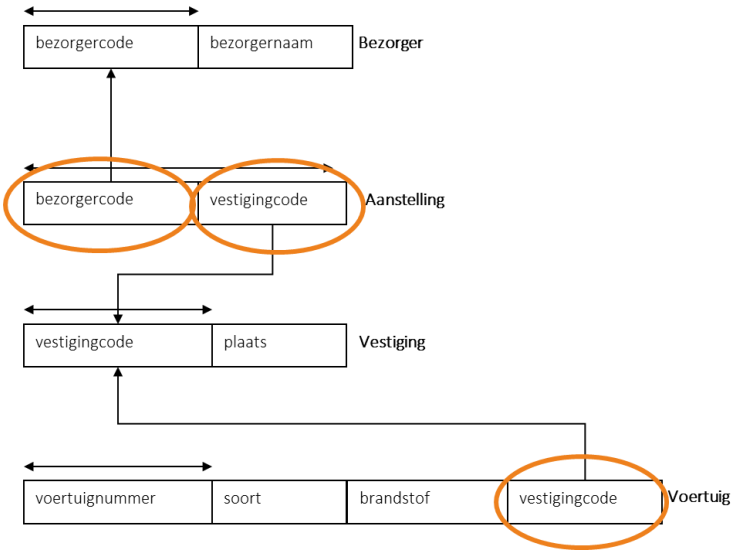
2. Onze uitwerking:



3. In de bovenstaande uitwerking staan de volgende *primaire sleutels*.



4. In de bovenstaande uitwerking staan de volgende *vreemde sleutels*.



5. **Let op**: in een strokendiagram loopt er altijd een pijl van een *vreemde sleutel* naar een *primaire sleutel*.

5.10 Oefening

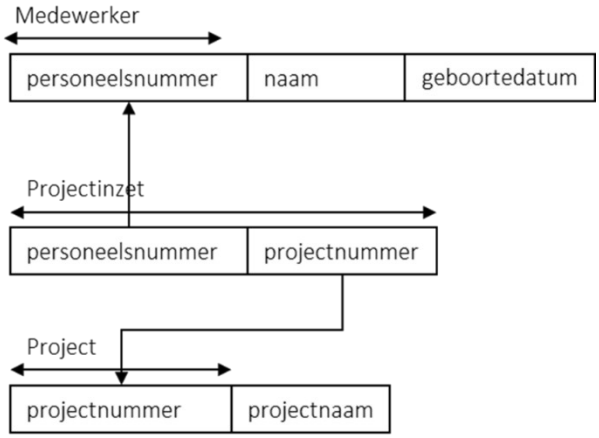
1. Maak een *ERD* en een *strokendiagram* op basis van de onderstaande tabel. Denk daarbij aan het stappenplan:
   a. Stap 1: selecteer voorbeelden uit de werkelijkheid.
   b. Stap 2: leg uit in natuurlijke taal.
   c. Stap 3: bepaal entiteittypes.
   d. Stap 4: bepaal attributen en identifiers.
   e. Stap 5: bepaal relaties.
   f. Stap 6: check, recheck, dubbelcheck.
2. **Stap 1: selecteer voorbeelden uit de praktijk**.
3. De onderstaande tabel geeft de informatiebehoefte weer.

perso-neelsnr	naam	geboorte-datum	project-nr	pnaam
130	Cupido	16-06-1980	10 11	BTS De Zwart
131	Doeksen	26-11-1984	10	BTS
145	Spanjer	15-07-1983	11 12	De Zwart D&D
146	Wiersma	20-03-1982	10 11 12	BTS De Zwart D&D
147	Smit	21-05-1995		
156	Meijer	23-031986	12	D&D
			13	E&E

4. Op basis van deze tabel volgt onze uitwerking hieronder.
5. **Stap 2: Leg uit in natuurlijke taal**.
   a. Van een ***medewerker*** leggen we het ***personeelsnummer***, de ***naam*** en de ***geboortedatum*** vast.
   b. Van een ***project*** leggen we het ***nummer*** en de ***naam*** vast.
   c. ***Medewerkers*** worden toegewezen aan 0 of meer ***projecten***.
   d. ***Projecten*** hebben 0 of meer ***medewerkers*** toegewezen.
6. **Stap 3: bepaal entiteittypes**.
   a. Dat zijn ***Medewerker*** en ***Project***.
7. **Stap 4: bepaal attributen en identifiers**.
   a. Medewerker: <u>personeelsnummer</u>, naam, geboortedatum.
   b. Project: <u>projectnummer</u>, projectnaam.

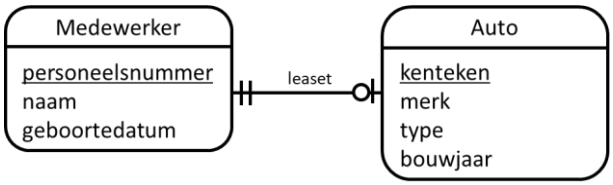
8. **Stap 5: bepaal relaties.**
   a. Zie de opmerking hierboven. Er is een *veel-op-veel* relatie tussen ***Medewerker*** en ***Project***.
   b. Dit leidt tot het volgende *ERD*.

9. **Stap 6: check, recheck, dubbelcheck**.
   a. Het ziet er goed uit ;-).
10. Omdat het gaat om een *veel-op-veel* relatie, komt er in de implementatie een *tussentabel* bij met daarin de *primaire sleutels* van beide tabellen.
11. Het *strokendiagram* ziet er dan als volgt uit.



5.11 1-op-1 relatie omzetten

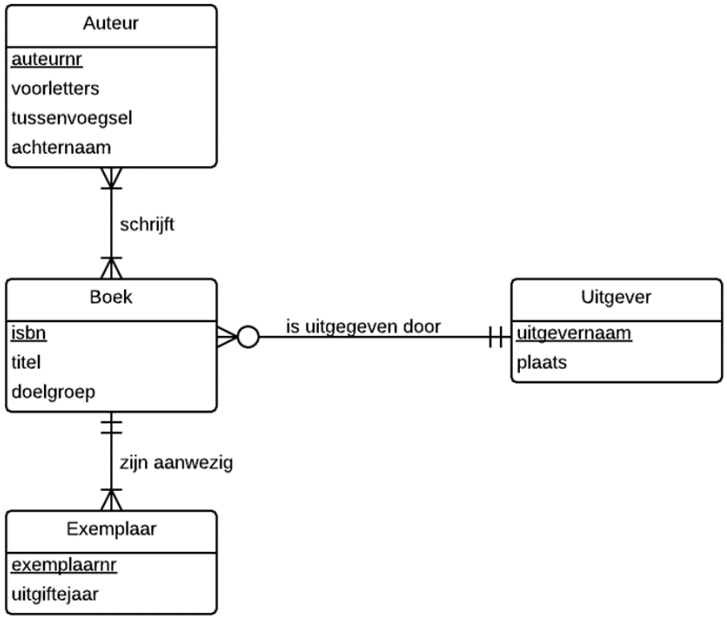
1. Bij een *1-op-1* relatie kun je de *vreemde sleutel* aan beide kanten zetten. Bekijk het volgende *ERD*.



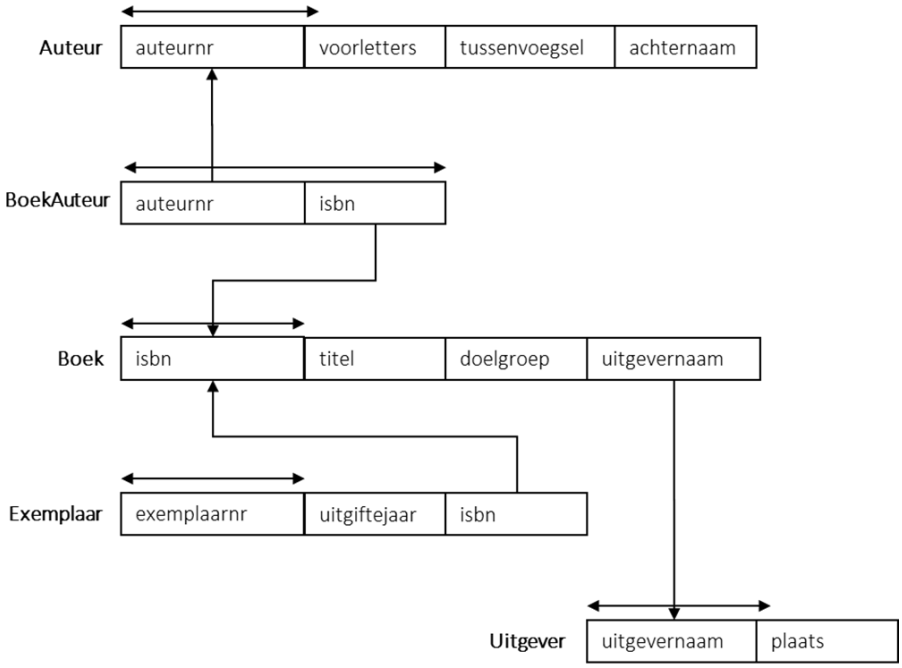
2. Je kunt kiezen waar je de *vreemde sleutel* toevoegt. Je kunt het ***personeelsnummer*** opnemen in de tabel ***Auto***, óf het ***kenteken*** opnemen in de tabel ***Medewerker***.
3. **Let op**: zet de vreemde sleutel bij voorkeur aan een kant waar deze verplicht is.

5.12 Oefening bibliotheek

1. Je hebt het volgende *ERD* van de bibliotheek al gezien. Bedenk welke vreemde sleutels en tussentabellen nodig zijn om het *ERD* om te zetten naar een *relationeel model*. Maak vervolgens een *strokendiagram*.



2. **Uitwerking**:
   a. De relatie tussen ***Auteur*** en ***Boek*** is *veel-op-veel*. Hier komt een *tussentabel*.
   b. De relatie tussen ***Boek*** en ***Uitgever*** is *1-op-veel*. Er komt een *vreemde sleutel* bij de tabel ***Boek***, die verwijst naar de *primaire sleutel* in de tabel ***Uitgever***.
   c. De relatie tussen ***Boek*** en ***Exemplaar*** is *1-op-veel*. Er komt een *vreemde sleutel* bij de tabel ***Exemplaar***, die verwijst naar de *primaire sleutel* in de tabel ***Boek***.
3. Dit leidt tot het volgende strokendiagram.



5.13 Eisen aan primaire en vreemde sleutels.

1. Bekijk de volgende video: relationship rules and referential integrity.
2. Voor een *primaire sleutel* geldt:
   a. deze is verplicht (*totaliteitsregel*).
   b. deze is uniek (*uniciteitsregel*).
   c. deze is zo 'smal' mogelijk gekozen (*minimaliteitseis*).
3. Voor een *vreemde sleutel* geldt:
   a. deze verwijst naar een bestaande *primaire sleutel* in een andere tabel (*referentiële integriteitsregel*).

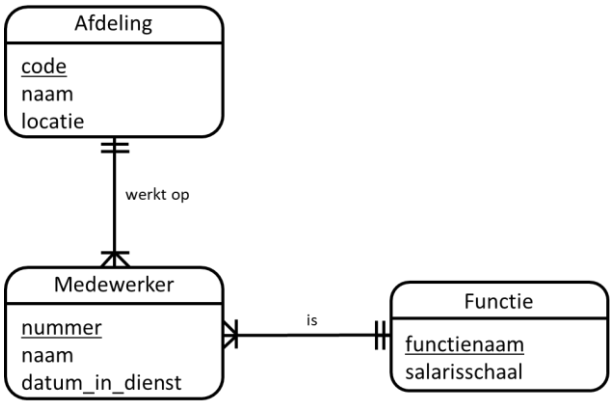
5.14 Oefening Afdelingen

1. Hieronder vind je een tabel met de informatiebehoefte.

Afdeling (code, naam)	Locatie	Medewerkernaam	Datum in dienst	Functie	Salaris-schaal
PZ Personeelszaken	Hoorn	Cees van Boven	15-12-2010	Afdelingshoofd	13
		Jannetje Wildeman	01-05-1999	Personeels-functionaris	11
		Willemijn de Boer	01-09-2013	Personeels-functionaris	11
		Wim Dijkman	01-01-2007	Administrateur	9
FIN Financiën	Amsterdam	Marieke van Boven	01-01-2007	Controller	11
		Arend Johansen	01-03-2000	Administrateur	9
		Wilma de Vries	15-12-2010	Administrateur	9
		John Diepenbroek	01-02-2009	Administrateur	9
MKT Marketing	Amsterdam	Marieke Vreeswijk	01-04-2014	Afdelingshoofd	13

2. **Toelichting**:
   a. Elke medewerker werkt op één afdeling.
   b. Elke medewerker heeft een functie.
   c. Bij elke functie hoort een vaste salarisschaal. Zo zitten afdelingshoofden bijvoorbeeld altijd in schaal 13.
3. Maak een *ERD* en een *strokendiagram* op basis van de bovenstaande informatie.

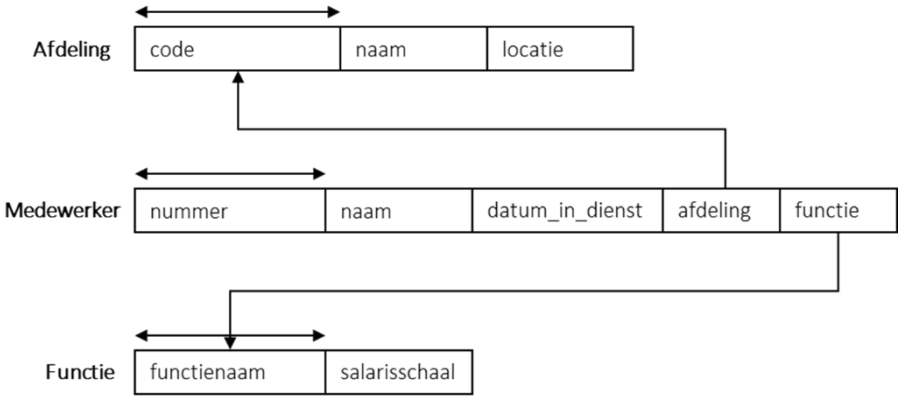
4. Onze uitwerking van het *ERD*.



5. **Toelichting**:

   a. We hebben al eerder geconstateerd dat de naam van een persoon in het algemeen niet uniek is en dus niet geschikt als *primaire sleutel*. Vandaar dat er gekozen wordt voor een *surrogaatsleutel*.

6. Onze uitwerking van het *strokendiagram*:



5.15 Opdracht

1. Doe opdracht **Relationele modellen.1 Bands strokendiagram** (niveau: basis).

5.16 Relaties met eigen attributen

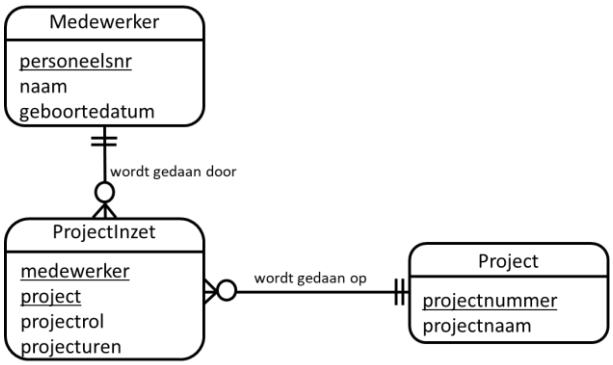
1. Bekijk de volgende informatiebehoefte.

perso-neelsnr	naam	geboorte-datum	project nr	project naam	projectrol	projecturen
130	Cupido	16-06-1980	10 11	BTS De Zwart	Projectleider Analist	200 500
131	Doeksen	26-11-1984	10	BTS	Programmeur	500
145	Spanjer	15-07-1983	11 12	De Zwart D&D	Programmeur Programmeur	300 250
146	Wiersma	20-03-1982	10 11 12	BTS De Zwart D&D	Projectleider Analist Projectleider	200 180 100
156	Meijer	23-03-1986	12	D&D	DataArchitect	600

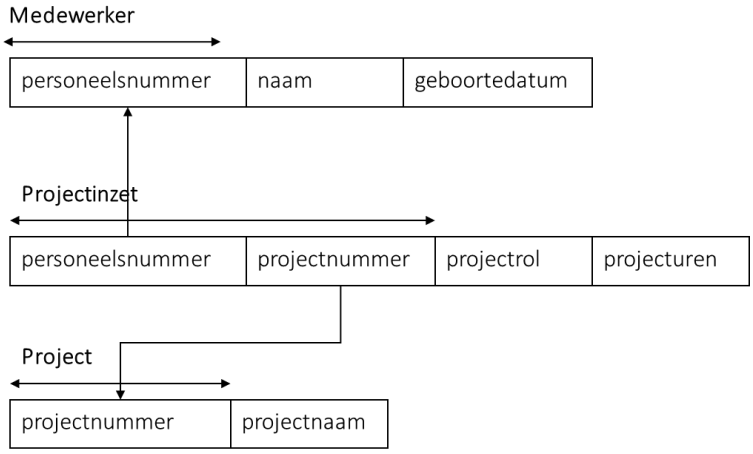
2. Toelichting:
   a. Medewerker met personeelsnummer ***130*** werkt op project met nummer ***10*** in de rol van ***projectleider***.
   b. Medewerker met personeelsnummer ***130*** werkt op project met nummer ***11*** in de rol van ***analist***.
   c. Medewerker met personeelsnummer ***130*** werkt op project met nummer ***10*** op basis van ***200*** projecturen.
   d. Medewerker met personeelsnummer ***130*** werkt op project met nummer ***11*** op basis van ***500*** projecturen.
3. Bij het bepalen van de entiteittypes komen we tot een ***Medewerker*** met een ***personeelsnummer***, een ***naam*** en een ***geboortedatum*** en een ***Project*** met een ***nummer*** en een ***naam***.
4. Er is een *veel-op-veel* relatie tussen ***Medewerker*** en ***Project*** aangezien een medewerker op meerdere projecten kan werken en een project meerdere medewerkers kent.
5. Het volgende *ERD* ontstaat dan:

6. In de implementatie ontstaat hier een *koppeltabel*.
7. Waar horen de attributen projectrol en projecturen thuis? Het zijn geen eigenschappen van een medewerker, noch van een project. Deze attributen zeggen iets over de relatie zelf. Bij de inzet in project ***10*** heeft de medewerker ***130*** de rol van ***projectleider*** voor ***200*** uur. Deze waarden gelden heel specifiek voor deze inzet!
8. De conclusie is dat deze attributen worden geplaatst in de *koppeltabel*. Hiermee wordt de *koppeltabel* meer dan een implementatie van een logisch model en moet de relatie

als entiteit worden opgenomen in het logische model (de *ERD*). De relatie heeft zelf eigenschappen en is daarmee een object geworden. Het *ERD* ziet er dan zo uit:



9. We krijgen dan het volgende strokendiagram.

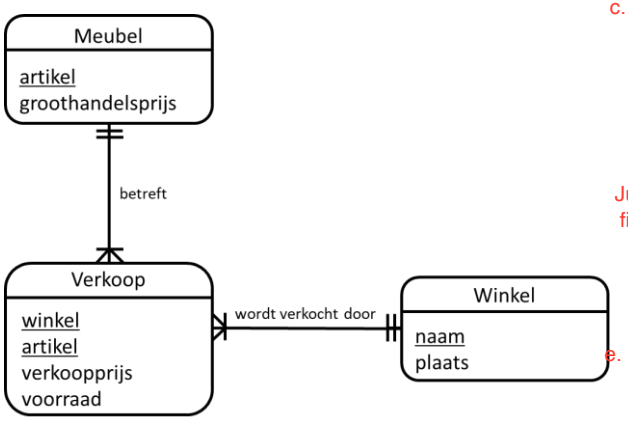


5.17 Oefening Meubels

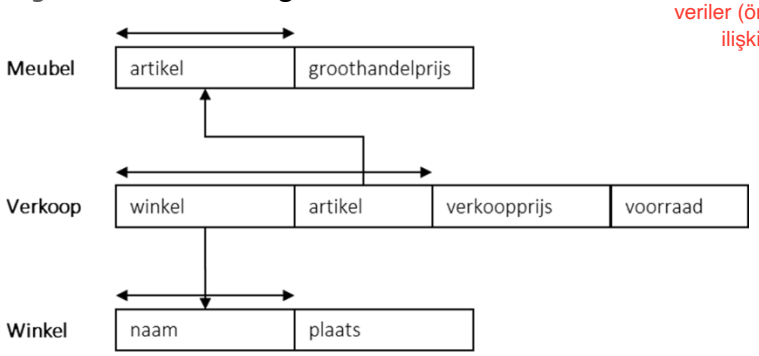
1. Maak een *ERD* en een *strokendiagram* op basis van de volgende informatiebehoefte.

artikel	groothandel prijs	winkel	plaats	verkoopprijs	voorraad
Jultorp	190,95	IDEA	Amsterdam	248,00	35
		Holfer	Utrecht	230,50	-
		Habdite	Groningen	250,00	10
Bomsund	159,95	IDEA	Amsterdam	189,00	44
		Hoptrendy	Rotterdam	199,00	23
Helting	174,95	Brundis	Zwolle	239,95	42
		Nehkamp	Eindhoven	214,50	21
		Hoptrendy	Rotterdam	224,25	14
Memphis	310,50	Holfer	Utrecht	414,95	41
		Brundis	Zwolle	350,00	24
		Habdite	Groningen	385,75	9
Veerallee	285,95	Holfer	Utrecht	335,00	-
		IDEA	Amsterdam	353,95	11
		Nehkamp	Eindhoven	315,00	2

2. Onze uitwerking:
   a. Van een ***Meubel*** worden de ***artikel*** en de ***groothandelsprijs*** vastgelegd.
   b. Van een ***Winkel*** worden de ***naam*** en de ***plaats*** vastgelegd.
   c. Er is een *veel-op-veel* relatie tussen ***Meubel*** en ***Winkel***.
   d. De attributen ***verkoopprijs*** en ***voorraad*** geven een nadere invulling van deze relatie.
      i. Zo is de verkoopprijs van ***Jultorp*** bij ***IDEA €248,00*** en zijn er ***35*** op voorraad.
      ii. Daarentegen is de verkoopprijs van ***Jultorp*** bij ***Holfer €230,50*** en is de voorraad ***onbekend***.
   e. Deze attributen worden toegevoegd aan de *koppeltabel*.
3. Het *ERD* ziet er dan als volgt uit.



4. Het *strokendiagram* ziet er als volgt uit.



a. Bir Mobilya (Meubel) için ürün numarası (artikel) ve toptan satış fiyatı (groothandelsprijs) kaydedilir.
b. Bir Mağaza (Winkel) için isim (naam) ve konum (plaats) kaydedilir.
c. Mobilya ile Mağaza arasında çoktan-çoğa (veel-op-veel) bir ilişki vardır.
d. Satış fiyatı (verkoopprijs) ve stok miktarı (voorraad), bu ilişkinin detaylarını belirtir.

Örneğin:

Jultorp adlı mobilya, IDEA mağazasında €248,00 fiyatla satılmakta ve 35 adet stok bulunmaktadır.

Aynı mobilya Holfer mağazasında ise €230,50 fiyattadır ve stok durumu bilinmemektedir.

e. Bu tür bilgiler, ilişkiye ait özel bilgiler olduğundan, koppeltabel (bağlantı tablosu)'na eklenir.

🔄 Sonuç:
Bu tür çoktan-çoğa ilişkilerde, ilişkiye özel olan veriler (örneğin satış fiyatı, stok miktarı), doğrudan ilişkiyi tanımlayan koppeltabel'de tutulur.

5.18 Opdrachten

1. Doe opdracht **Relationele modellen.2 Landen strokendiagram** (niveau: basis).
2. Doe opdracht **Relationele modellen.3 Hondenuitlaat ERD** (niveau: gevorderd).
3. Doe opdracht **Relationele modellen.4 Hondenuitlaat strokendiagram** (niveau: gevorderd).

6 NORMALISEREN

6.1 Introductie normaliseren

1. Tot nu toe hebben we een relationeel model (*strokendiagram*) gemaakt op basis van logisch denken en gedeeltelijk op basis van intuïtie. Er is ook een gestructureerde manier om te komen tot een relationeel model. Dit proces heet *normaliseren*.

2. Bekijk de volgende vier video's over *normaliseren*:
   Normalization
   First normal form
   Second normal form
   Third normal form

3. Wat is *normaliseren*? Volgens Wikipedia is het:
   Database normalization is the process of organizing the fields and tables of a relational database to minimize redundancy and dependency. Normalization usually involves dividing large tables into smaller (and less redundant) tables and defining relationships between them. The objective is to isolate data so that additions, deletions, and modifications of a field can be made in just one table and then propagated through the rest of the database via the defined relationships.

4. Kort samengevat: ieder 'ding' of 'object' krijgt zijn eigen tabel en er mag geen redundantie zijn, dus geen gegevens die op meer dan één plek worden vastgelegd.

5. **Merk op** dat er heel veel *normaalvormen* (*normal forms*) zijn. In het algemeen geeft de *derde normaalvorm* een goede balans tussen database optimalisatie en query optimalisatie (het opzoeken van data in je database). In deze opleiding gaan we daarom niet verder dan de *derde normaalvorm*.

6.2 (Samengestelde) meerwaardige attributen

1. Voordat we beginnen met het proces van *normaliseren* lichten we enkele situaties toe die met behulp van *normalisatie* opgelost kunnen worden. Eén van die problemen betreft *(samengestelde) meerwaardige attributen*.

2. Bekijk de volgende tabel.

klantnr	voor letters	tussen voegsels	achter naam	auto
1922	J.A.E.	van	Dee	48-KHB-4
9277	J.J.		Luingen	67-KFX-1
7731	B.M.	El	Amrani	76-PV-HG, 74-PRT-2
4442	S.	van	Vianen	29-PZL-7
1970	W.H.		Vogelzang	12-TR-ZX, 15-FZ-DG, GF-VL-18
7884	R.S.	dos	Santos	76-JD-PD

3. In de kolom (attribuut) ***auto*** zie je op sommige plaatsen meer dan één kenteken staan. Dit noemen we een *meerwaardig attribuut*.

4. Dit vertaalt zich erg lastig naar een tabel in de database. Het uitgangspunt in een tabel is dat elk attribuut *enkelwaardig* is (dus maar één waarde bevat). Dit maakt het opslaan, toevoegen, verifiëren en opzoeken van een ***auto*** een stuk makkelijker voor het *DBMS*.

5. Bekijk de volgende tabel.

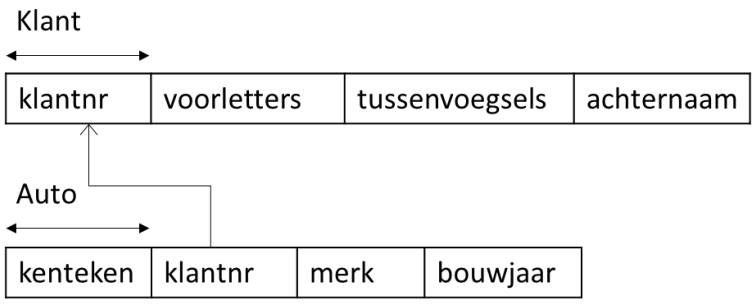
klantnr	voor letters	tussen voegsels	achter naam	kenteken	merk	bouw jaar
1922	J.A.E.	van	Dee	48-KHB-4	Peugeot	2008
9277	J.J.		Luingen	67-KFX-1	BMW	2008
7731	B.M.	El	Amrani	76-PV-HG 74-PRT-2	Honda Mercedes	1999 2011
4442	S.	van	Vianen	29-PZL-7	Toyota	2012
1970	W.H.		Vogelzang	12-TR-ZX 15-FZ-DG GF-VL-18	Renault Renault Renault	2001 1998 1990
7884	R.S.	dos	Santos	76-JD-PD	Audi	1999

6. De tabel suggereert dat de klant met nummer ***7731*** een relatie heeft met twee auto's, net zoals in het vorige voorbeeld. Alleen leggen we nu niet alleen het ***kenteken*** vast van de auto, maar ook het ***merk*** en het ***bouwjaar***. Hierbij is de auto met kenteken ***76-PV-HG*** een ***Honda*** uit ***1999***. De drie attributen ***kenteken***, ***merk*** en ***bouwjaar*** horen bij elkaar en in sommige gevallen bevatten ze meer dan één waarde. Dit is een voorbeeld van *samengestelde meerwaardige attributen*.

7. Net zoals de gewone meerwaardige attributen is dit niet handig om het op deze manier vast te leggen in een database.

8. De oplossing is in feite al bekend: elk ding (object) krijgt een eigen tabel. Zie het *ERD* hieronder.

9. In de implementatie krijgt het *kindtabel* (***Auto***) een extra kolom, die verwijst naar de *primaire sleutel* van de *oudertabel* (***Klant***). Het bijbehorende *strokendiagram* ziet er dan als volgt uit.



6.3 Redundantie

1. Een ander probleem dat we met normalisatie oplossen is *redundantie*.
2. De onderstaande tabel is een andere weergave van het voorbeeld hierboven. Merk op dat er geen sprake is van *(samengestelde) meerwaardige attributen*.

kenteken	merk	bouw jaar	klantnr	voor letters	tussen voegsels	achter naam
48-KHB-4	Peugeot	2008	1922	J.A.E.	van	Dee
67-KFX-1	BMW	2008	9277	J.J.		Luingen
76-PV-HG	Honda	1999	7731	B.M.	El	Amrani
29-PZL-7	Toyota	2012	4442	S.	van	Vianen
74-PRT-2	Mercedes	2011	7731	B.M.	El	Amrani
12-TR-ZX	Renault	2001	1970	W.H.		Vogelzang
15-FZ-DG	Renault	1998	1970	W.H.		Vogelzang
76-JD-PD	Audi	1999	7884	R.S.	dos	Santos
GF-VL-18	Renault	1990	1970	W.H.		Vogelzang

Redundans, verilerin birden fazla yerde tekrar edilmesi anlamına gelir ve bu, veritabanı tasarımında istenmeyen bir durumdur. Normalizasyon süreci, bu tür tekrarları ortadan kaldırmak amacıyla verileri uygun şekilde ayrıştırır.

Örneğin, önceki tabloda bir müşteri birden fazla arabaya sahip olduğunda, her arabayla ilişkili veriler (örneğin, marka, model) her müşteri için tekrar edebilir. Ancak, normalizasyon işlemiyle bu tekrarı önlemek için her varlık (müşteri, araba) kendi tablosunda tutulur ve aralarındaki ilişki yabancı anahtarlarla sağlanır.

3. Er is echter sprake van een ander probleem. We zien dat de gegevens van klant ***7731*** op twee plaatsen wordt vastgelegd en de gegevens van klant ***1970*** zelfs op drie plaatsen.
4. Het grootste nadeel van het vastleggen van dezelfde gegevens op verschillende plekken is dat er tegenstrijdigheden kunnen ontstaan. Wat is de waarheid als de achternaam de ene keer als ***Vogelzang*** wordt gegeven en op een andere plek als ***Vogelenzang***?
5. Ook nu is de oplossing om elk ding/object een eigen tabel te geven en daarmee de gegevens maar op één plaats vast te leggen.

Bu durum, veri tutarsızlıklarına yol açar. Eğer aynı müşteri farklı tablolarda farklı şekillerde saklanırsa, bu çelişkili bilgilere yol açabilir (örneğin, aynı soyadının iki farklı şekilde yazılması gibi). Bu nedenle, normalizasyon sürecinde her varlık (örneğin müşteri) kendi tablosunda tutulur ve ilişkiler yabancı anahtarlarla sağlanır. Bu sayede veri sadece bir kez tutulur ve çelişkilerin önüne geçilir.

6.4 Afleidbare gegevens

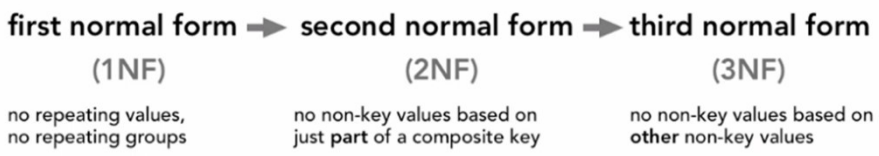
1. Een speciale vorm van redundantie ontstaat wanneer een attribuut af te leiden is uit andere attributen.
2. Bekijk het onderstaande voorbeeld.

startdatum	einddatum	duur
12-08-2014	26-08-2014	15 dagen

3. De ***duur*** in dagen (berekend inclusief de ***start***- en ***einddatum***) is te berekenen op basis van de twee data.
4. Het nadeel van het vastleggen van de ***duur*** is weer dat er inconsistentie kan ontstaan. Als iemand de ***duur*** verandert in ***16 dagen***, betekent dit dan dat de ***startdatum*** ook aangepast moet worden, of de ***einddatum***, of misschien wel allebei?
5. De oplossing is om afleidbare gegevens niet op te nemen in een tabel.

6.5 Het normalisatieproces

1. Het normalisatieproces (tot en met de *derde normaalvorm*) doorloopt altijd de volgende stappen.



2. De *eerste normaalvorm* zegt dat er geen *meerwaardige attributen* mogen zijn en geen *samengestelde meerwaardige attributen* mogen zijn.
3. De *tweede normaalvorm* zegt dat er geen niet-sleutel attributen mogen zijn, die functioneel afhankelijk zijn van een deel van de samengestelde sleutel. We lichten dit hieronder toe aan de hand van een voorbeeld.
4. De *derde normaalvorm* zegt dat er geen niet-sleutel attributen mogen zijn, die functioneel afhankelijk zijn van een ander niet-sleutel attribuut. Het gaat hier ook afleidbare attributen. We lichten dit hieronder toe aan de hand van een voorbeeld.

6.6 Voorbeeld sportwinkel

1. Hieronder staat een tabel met de informatiebehoefte van een sportwinkel om de bestelgegevens vast te leggen. We lichten de normalisatie stappen toe aan de hand van dit voorbeeld.

order nr	datum	klant nr	naam	artikel nr	artikel naam	aantal	artikel prijs	totaal prijs
178	12-07-14	331	Hamoudi	376	Halter 6 kg	2	12,99	25,98
				431	Springtouw	1	5,99	5,99
179	12-07-14	201	van Dam	299	Hometrainer	1	129,00	129,00
181	12-07-14	491	De Vries	300	Gewichtmanchet	4	4,29	17,16
				376	Halter 6 kg	1	12,99	12,99
				379	Halter 10 kg	2	14,99	29,98
182	15-07-14	201	van Dam	473	Fitnessbal	1	14,99	14,99
				431	Springtouw	5	5,99	29,95
185	16-07-14	175	Yilmaz	144	Abtrainer	1	27,99	27,99

2. Een ***klant*** met een ***nummer*** en een ***naam*** bestelt een of meer ***artikelen***. De ***bestelling*** krijgt een ***nummer*** en een ***datum*** van bestelling. Op de ***bestelling*** wordt per ***artikel*** vastgelegd om welk ***artikel*** het gaat, ***hoeveel*** er besteld zijn, wat de ***prijs*** van het artikel is en wat de ***totaalprijs*** is voor dit artikel.
3. **Eerste normaalvorm**: Er mogen geen *(samengestelde) meerwaardige attributen* zijn. Deze worden in een eigen tabel geplaatst.
4. In de tabel zijn de laatste vijf attributen *samengesteld meerwaardig*. Die worden geplaatst in een eigen tabel. De relatie wordt vastgelegd door in de nieuwe tabel een

kolom op te nemen die verwijst naar de *primaire sleutel* van de andere (eerste) vier kolommen. Dat is het ***ordernummer***. In de nieuwe tabel is ordernummer dan een *vreemde sleutel* (foreign key).

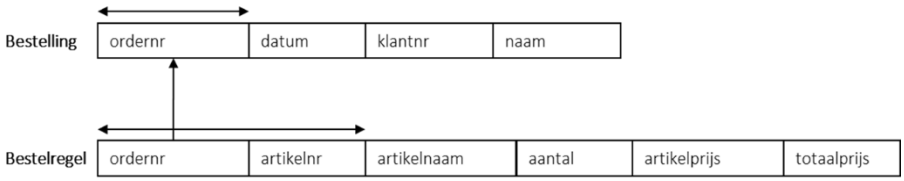
5. Het resultaat in tabelvorm is dan.

order nr	datum	klant nr	naam
178	12-07-14	331	Hamoudi
179	12-07-14	201	van Dam
181	12-07-14	491	De Vries
182	15-07-14	201	van Dam
185	16-07-14	175	Yilmaz

order nr	artikel nr	artikel naam	aantal	artikel prijs	totaal prijs
178	376	Halter 6 kg	2	12,99	25,98
178	431	Springtouw	1	5,99	5,99
179	299	Hometrainer	1	129,00	129,00
181	300	Gewichtmanchet	4	4,29	17,16
181	376	Halter 6 kg	1	12,99	12,99
181	379	Halter 10 kg	2	14,99	29,98
182	473	Fitnessbal	1	14,99	14,99
182	431	Springtouw	5	5,99	29,95
185	144	Abtrainer	1	27,99	27,99

6. Hoe identificeer je nu een rij in de rechtertabel? Merk op dat de combinatie van ***ordernummer*** en ***artikelnummer*** uniek is. Dat wordt dus de *primaire sleutel* van de rechter tabel.

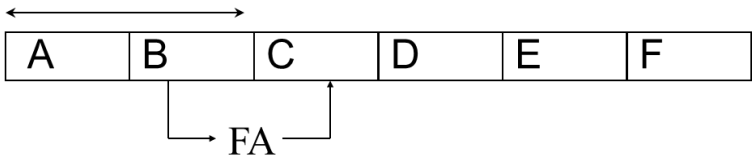
7. En in een *strokendiagram*.



8. **Tweede normaalvorm**: Geen enkel niet-sleutel attribuut wordt bepaald door (is functioneel afhankelijk van) een gedeelte van de primaire sleutel.

9. Omdat er sprake is van "een gedeelte van de primaire sleutel" is deze eis alleen betekenisvol voor tabellen met een *samengestelde sleutel*.

10. In een schema (FA staat voor functioneel afhankelijk):



11. De tabel ***Bestelling*** heeft een enkelvoudige *primaire sleutel* en dus hoeven we deze tabel niet te bekijken.

12. De tabel ***Bestelregel*** heeft wel een *samengestelde sleutel*.

    a. De kolommen ***artikelnaam*** en ***artikelprijs*** worden alleen bepaald door het ***artikelnummer***. Ongeacht op welke bestelregel het staat zal de naam van het artikel met nummer ***376*** altijd ***Halter 6 kg*** zijn en zal de prijs altijd ***12,99*** zijn).

    b. De andere twee kolommen zijn wel afhankelijk van de hele sleutel. Het ***aantal*** kan verschillen per ***bestelregel*** en dus ook de ***totaalprijs***.

order nr	artikel nr	artikel naam	aantal	artikel prijs	totaal prijs
178	376	Halter 6 kg	2	12,99	25,98
178	431	Springtouw	1	5,99	5,99
179	299	Hometrainer	1	129,00	129,00
181	300	Gewichtmanchet	4	4,29	17,16
181	376	Halter 6 kg	1	12,99	12,99
181	379	Halter 10 kg	2	14,99	29,98
182	473	Fitnessbal	1	14,99	14,99
182	431	Springtouw	5	5,99	29,95
185	144	Abtrainer	1	27,99	27,99

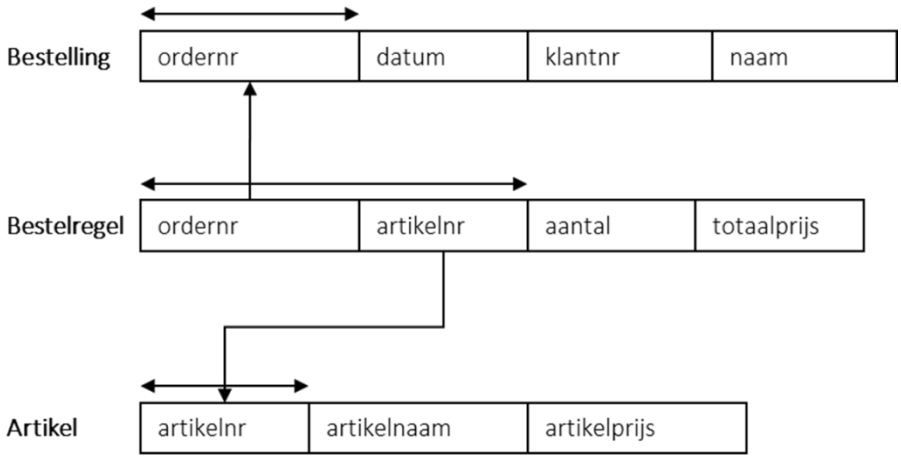
13. De gebruikte terminologie is:
    a. De ***artikelnaam*** en de ***artikelprijs*** zijn *functioneel afhankelijk* van het ***artikelnr***.
    b. Het ***artikelnr*** is de *determinant* voor de ***artikelnaam*** en de ***artikelprijs***.
    c. Het ***artikelnr*** *bepaalt* de ***artikelnaam*** en de ***artikelprijs***.
14. In de *tweede normaalvorm* geldt:
    a. De *determinant* en de *functioneel afhankelijke* velden krijgen een aparte tabel.
    b. De *determinant* is *primaire sleutel* van de nieuwe tabel.
    c. De *determinant* blijft staan in de oorspronkelijke tabel en verwijst dan naar de primaire sleutel van de nieuwe tabel.
    d. In de oorspronkelijk tabel moet de determinant blijven om de relatie te leggen en dus is de determinant in die tabel een *vreemde sleutel*.
15. Het resultaat in tabelvorm is dan:

order nr	datum	klant nr	naam
178	12-07-14	331	Hamoudi
179	12-07-14	201	van Dam
181	12-07-14	491	De Vries
182	15-07-14	201	van Dam
185	16-07-14	175	Yilmaz

order nr	artikel nr	aantal	totaal prijs
178	376	2	25,98
178	431	1	5,99
179	299	1	129,00
181	300	4	17,16
181	376	1	12,99
181	379	2	29,98
182	473	1	14,99
182	431	5	29,95
185	144	1	27,99

artikel nr	artikel naam	artikel prijs
376	Halter 6 kg	12,99
431	Springtouw	5,99
299	Hometrainer	129,00
300	Gewichtmanchet	4,29
379	Halter 10 kg	14,99
473	Fitnessbal	14,99
144	Abtrainer	27,99

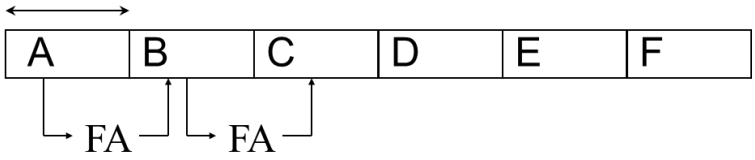
16. En in een *strokendiagram*:



17. **Derde normaalvorm**: Geen enkel niet-sleutel attribuut wordt bepaald door een ander niet-sleutel attribuut.

18. Het gaat hier om directe functionele afhankelijkheid en om afleidbare gegevens.

19. In een schema: ***B*** is een niet-sleutel attribuut en *functioneel afhankelijk* van ***A***, een sleutelattribuut. ***C*** is een ander niet-sleutel attribuut en *functioneel afhankelijk* van ***B***.



20. In de tabel ***Bestelling*** zien we dat het niet-sleutel attribuut ***naam*** (van de klant) bepaald wordt door het niet-sleutel attribuut ***klantnr***.

order nr	datum	klant nr	naam
178	12-07-14	331	Hamoudi
179	12-07-14	201	van Dam
181	12-07-14	491	De Vries
182	15-07-14	201	van Dam
185	16-07-14	175	Yilmaz

21. In de *derde normaalvorm* geldt dat (net zoals in de *tweede normaalvorm*):
    a. De *determinant* en de functioneel afhankelijke velden krijgen een aparte tabel.
    b. De *determinant* is *primaire sleutel* van de nieuwe tabel.
    c. De *determinant* blijft staan in de oorspronkelijke tabel en verwijst dan naar de primaire sleutel van de nieuwe tabel
    d. In de oorspronkelijk tabel moet de determinant blijven om de relatie te leggen en dus is de determinant in die tabel een *vreemde sleutel*.

22. In de tabel ***Bestelregel*** is het niet-sleutel attribuut ***totaalprijs*** af te leiden van het niet-sleutel attribuut ***aantal*** in combinatie met de ***prijs*** van het artikel. Dit attribuut laten we daarom weg.

order nr	artikel nr	aantal	~~totaal prijs~~
178	376	2	25,98
178	431	1	5,99
179	299	1	129,00
181	300	4	17,16
181	376	1	12,99
181	379	2	29,98
182	473	1	14,99
182	431	5	29,95
185	144	1	27,99

23. In de tabel ***Artikel*** zijn er geen functionele afhankelijkheden tussen niet-sleutel attributen.

24. Het resultaat in tabelvorm is dan:

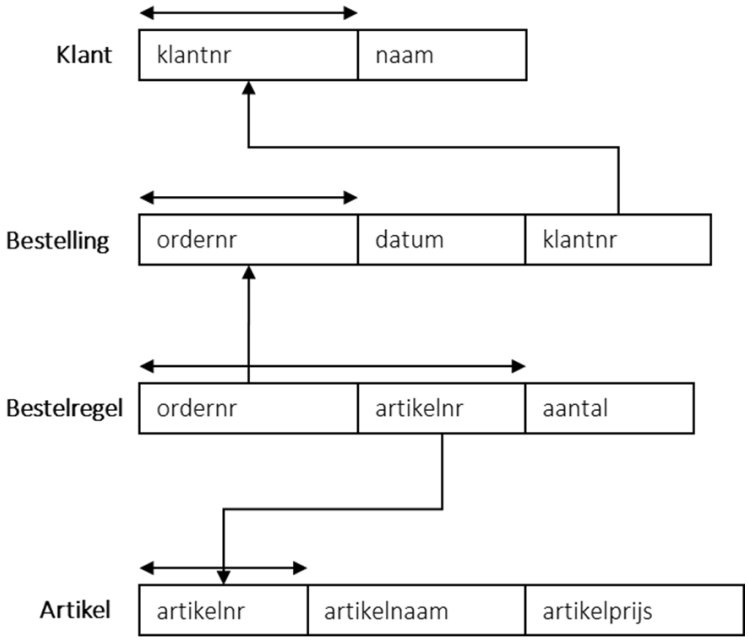
klant nr	naam
331	Hamoudi
201	van Dam
491	De Vries
175	Yilmaz

order nr	datum	klant nr
178	12-07-14	331
179	12-07-14	201
181	12-07-14	491
182	15-07-14	201
185	16-07-14	175

order nr	artikel nr	aantal
178	376	2
178	431	1
179	299	1
181	300	4
181	376	1
181	379	2
182	473	1
182	431	5
185	144	1

artikel nr	artikel naam	artikel prijs
376	Halter 6 kg	12,99
431	Springtouw	5,99
299	Hometrainer	129,00
300	Gewichtmanchet	4,29
379	Halter 10 kg	14,99
473	Fitnessbal	14,99
144	Abtrainer	27,99

25. En in een *strokendiagram*:



6.7 Oefening docentinzet

26. Normaliseer de data in deze tabel tot en met de *derde normaalvorm*. Geef je antwoord in een *strokenschema*.

docent code	docent naam	vakcode	studie-punten	leer-jaar	aantal groepen
KAB	K. Abma	DW1 DW2 RD1	3 3 4	1 1 1	2 2 4
HCO	H. L. Cornelisse	RD1 RD2	4 6	1 2	2 2
KER	K. Erdman	RD1	4	1	4
HIP	H. Ipperra	DW1 DW2 NT IP	3 3 5 6	1 1 2 3	2 2 2 1

27. Onze uitwerking (zeer beknopt):

28. *Eerste normaalvorm* (er mogen geen (samengestelde) meerwaardige attributen zijn).

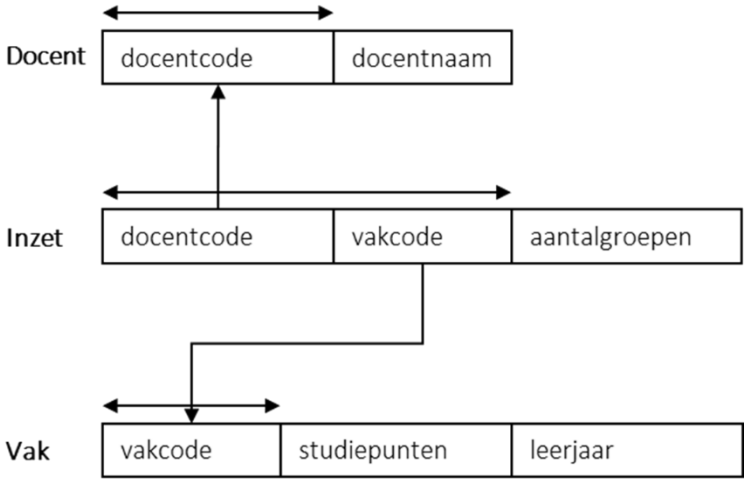
    a. De kolommen ***vakcode***, ***studiepunten***, ***leerjaar*** en ***aantalgroepen*** zijn *samengestelde meerwaardige attributen*.

    b. Deze krijgen een eigen tabel en de *identifier* van de oorspronkelijke tabel (***docentcode***) wordt toegevoegd.

    c. De combinatie ***docentcode*** en ***vakcode*** wordt dan de ***identifier*** voor de tweede tabel.

    d. Het resultaat is dan ***Docent*** (docentcode, docentnaam) en ***Inzet*** (docentcode, vakcode, studiepunten, leerjaar, aantalgroepen)

29. *Tweede normaalvorm* (Geen enkel niet-sleutel attribuut wordt bepaald dooreen gedeelte van de primaire sleutel).
    a. In de tabel ***Inzet*** zijn de kolommen ***studiepunten*** en ***leerjaar*** alleen afhankelijk van ***vakcode***. Deze worden in een aparte tabel gezet. De *identifier* blijft in de oorspronkelijke tabel.
    b. Het resultaat is dan ***Docent*** (docentcode, docentnaam) en ***Inzet*** (docentcode, vakcode, aantalgroepen) en ***Vak*** (vakcode, studiepunten, leerjaar).
30. *Derde normaalvorm* (Geen enkel niet-sleutel attribuut wordt bepaald door een ander niet-sleutel attribuut).
    a. Merk op dat alle tabellen al in de *derde normaalvorm* zijn. Er hoeft dus niets meer te worden aangepast.
    b. Het resultaat in een *strokendiagram* is dan:



6.8 Opdracht

1. Doe de opdracht **Normaliseren.1 Apotheek** (niveau: basis).

6.9 Voorbeeld Open Universiteit

1. Bekijk de volgende tabel.

student nr	studentnaam	cursus code	cursustitel	inschrijf datum	code studiecentrum	plaats studiecentrum
830025146	Yvette Nobel	E06112	Kunstmatige Intelligentie	15 apr 2010	17	Venlo
		N05112	Logica	9 sept 2012	17	Venlo
		T02112	Programmeren in Java	10 okt 2012	18	Sittard
		T03111	Systemen	12 dec 2013	17	Venlo
830019766	Mike van der Wel	T02112	Programmeren in Java	4 okt 2012	13	Rotterdam
830018259	Gijs Slot	B04112	Marketing	2 apr 2012	07	Middelburg
		E01112	Macro-economie	6 mei 2013	07	Middelburg
		T02112	Programmeren in Java	2 sep 2013	13	Rotterdam
830015613	Michael Jonker	E01112	Macro-economie	9 mei 2010	13	Rotterdam
		E06112	Kunstmatige Intelligentie	6 juni 2012	11	Leiden

2. In de tabel staan inschrijfgegevens van de Open Universiteit. Een student schrijft zich per cursus in en kiest hierbij voor een studiecentrum. In het studiecentrum kan de student terecht voor begeleiding, afname van tentamens etc.
3. Bedenk eerst zelf hoe de informatiebehoefte er in de *derde normaalvorm* uit kan zien. Onze uitwerking volgt hieronder.

4. **Uitwerking**: De laatste vijf attributen zijn *samengesteld meerwaardig* en worden in een eigen tabel geplaatst waaraan de *primaire sleutel* van de oorspronkelijke tabel wordt toegevoegd.

studentnr	studnaam	cursus-code	cursustitel	inschrijf datum	studie-centrum code	studie-centrum naam:
830025146	Y.Nobel	E06112 N05112 T02112 T03111	Statistiek Wiskunde Informatica Systemen	15 apr 85 9 sept 85 10 okt 85 12 dec 85	17 17 18 17	Venlo Venlo Sittard Venlo
830019766	M.v.d.Wel	T02112	Informatica	4 okt 85	13	Rotterdam
830018259	G.Slot	B04112 E01112 T02112	Marketing Economie Informatica	2 apr 85 6 mei 85 2 sep 85	07 07 13	Middelburg Middelburg Rotterdam
830015613	M.Jonker	E01112 E06112	Economie Statistiek	9 mei 85 6 juni 85	01 01	Leeuwarden Leeuwarden

5. De nieuwe tabel ***Inschrijving*** ziet er dan als volgt uit: ***Inschrijving*** (studentnr, cursuscode, cursustitel, inschrijfdatum, studiecentrumcode, studiecentrumnaam).

6. Wat is de *primaire sleutel* voor deze tabel? Waarschijnlijk heb je gekozen voor de combinatie ***studentnr*** en ***cursuscode***.

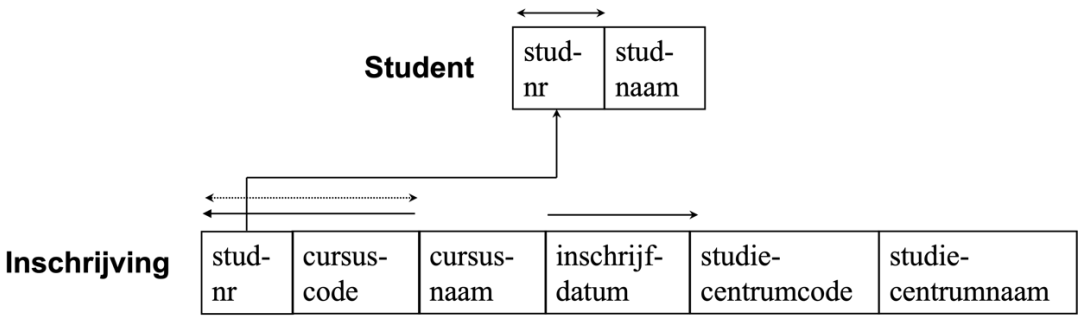
7. De consequentie is dat de combinatie ***studentnr*** en ***cursuscode*** maar één keer kan voorkomen in de tabel. Met andere woorden, een cursist kan zich niet twee keer voor dezelfde cursus inschrijven.

8. Het kan wenselijk zijn (dit wordt door de gebruiker bepaald) om een student de kans te geven om zich op een later moment nog eens in te schrijven voor dezelfde cursus. Wat moet er dan veranderd worden om dit mogelijk te maken?

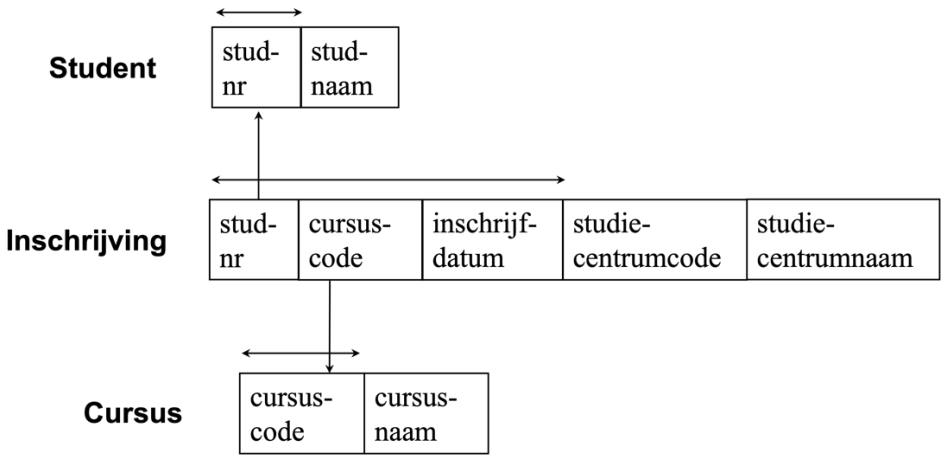
9. De meest waarschijnlijke optie is om de sleutel uit te breiden met de ***inschrijfdatum***. Dit betekent dat de combinatie ***studentnr***, ***cursuscode*** en ***inschrijfdatum*** uniek moet zijn. Dit maakt het mogelijk voor studenten om zich op een ander moment nog eens in te schrijven voor dezelfde cursus.

10. **Conclusie**: realiseer je dat de keuze van de *primaire sleutel* consequenties kan hebben voor de mogelijkheden die de gebruiker heeft. Stem dit dus altijd af met de gebruiker en laat deze bepalen wat wenselijk is.

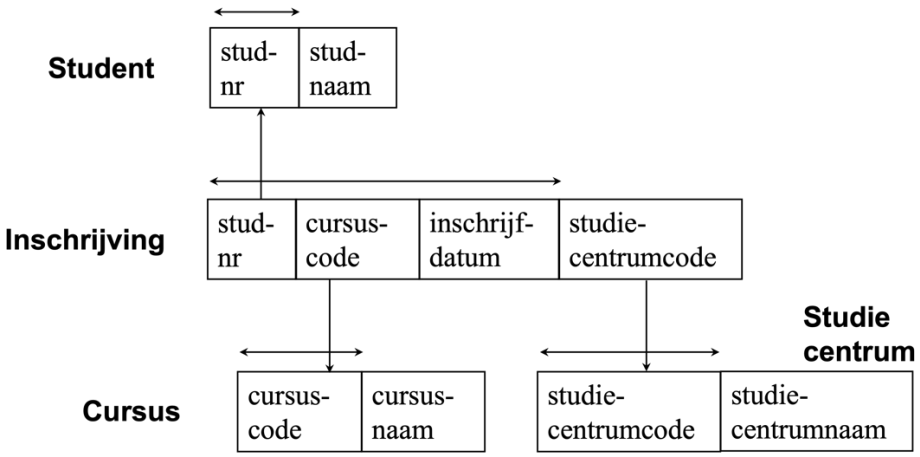
11. Het strokendiagram voor de *eerste normaalvorm* ziet er dan zo uit (met de twee opties voor ***Inschrijving*** aangegeven).



12. Voor de *tweede normaalvorm* kijken we weer alleen naar de tabel(len) met een samengestelde sleutel. Merk op dat ***cursusnaam*** afhankelijk is van ***cursuscode*** en die maakt deel uit van de samengestelde sleutel. We moeten een tabel afsplitsen.
13. Het strokendiagram voor de *tweede normaalvorm* ziet er dan zo uit. We kiezen voor de optie waarbij ***Inschrijving*** een *primaire sleutel* heeft bestaande uit de combinatie ***studentnr***, ***cursuscode*** en ***inschrijfdatum.***



14. Merk op dat ***studiecentrumnaam*** afhankelijk is van ***studiecentrumcode***. En ***studiecentrumcode*** is geen sleutel attribuut. Dit betekent dat we hier nog een tabel moeten afsplitsen.
15. Het strokendiagram voor de *derde normaalvorm* ziet er dan zo uit.

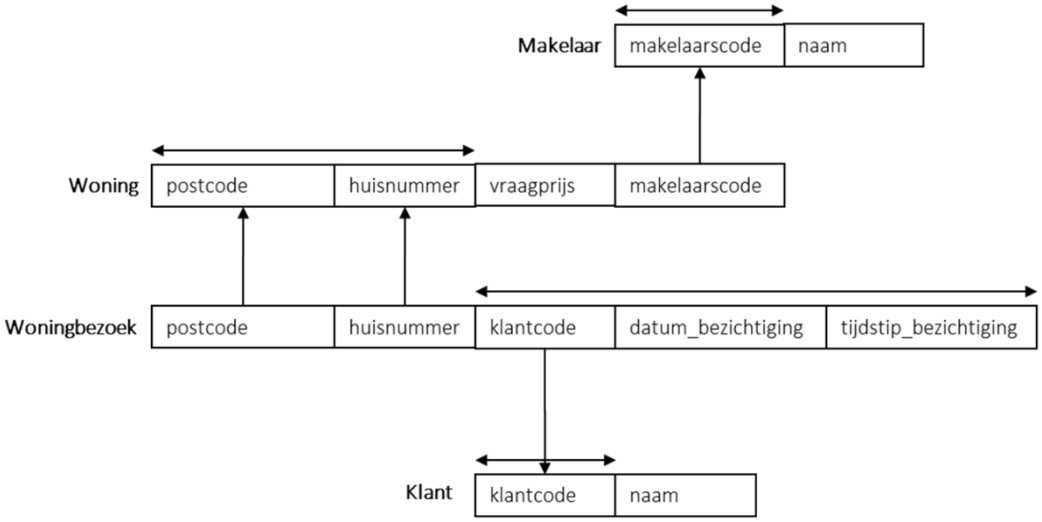


6.10 Oefening Woningbezoeken

1. Bekijk de volgende tabel.

woning (postcode, huisnr)	vraag-prijs	datum bezichtiging	tijd bezichti-ging	klant	make-laars code	makelaars naam
2309KV, 34	215.000	23-10-2010 23-10-2010 24-10-2010 25-10-2010	10:00u 11:00u 10:00u 15:00u	2314, Smit 2317, Donk 4256, Hooft 2314, Smit	HHU	Hans Huizen
1093VD, 65	380.000	23-10-2010 23-10-2010	10:00u 16:00u	4256, Hooft 2317, Donk	JHU	Jan Huizen
2308KY, 12	590.000	23-10-2010	14:00u	6723, Abel	GBA	Gerard Baas
1200DZ, 145	249.000	23-10-2010 24-10-2010	15:00u 11:00u	2314, Smit 2317, Donk	HHU	Hans Huizen

2. Normaliseer de informatiebehoefte tot en met de *derde normaalvorm*.
3. Onze **uitwerking** (beknopt):
4. *Eerste normaalvorm*: ***Woning*** (postcode, huisnr, vraagprijs, makelaarscode, makelaarsnaam); ***Woningbezoek*** (postcode, huisnr, datum_bezichtiging, tijd_bezichtiging, klantcode, klantnaam).
5. Hoe identificeer je een woningbezoek?
6. Het ligt voor de hand om de oorspronkelijke sleutel te combineren met een sleutel van de nieuwe tabel. Je kunt dan komen tot ***postcode***, ***huisnr***, ***datum_bezichtiging*** en ***tijd_bezichtiging***. Dit zijn vier attributen. Voor een primaire sleutel geldt een *minimaliteitseis* en aangezien een klant nooit op hetzelfde moment op twee verschillende plekken een woning kan bezoeken is de combinatie ***datum_bezichtiging***, ***tijd_bezichtiging*** en ***klantcode*** ook uniek.
7. De *eerste normaalvorm* ziet er dan als volgt uit. ***Woning*** (postcode, huisnr, vraagprijs, makelaarscode, makelaarsnaam); ***Woningbezoek*** (postcode, huisnr, datum_bezichtiging, tijd_bezichtiging, klantcode, klantnaam).
8. Tweede *normaalvorm*: het niet-sleutel attribuut ***klantnaam*** in de tabel ***Woningbezoek*** is functioneel afhankelijk van een gedeelte van de sleutel, namelijk van ***klantcode***. Dit wordt een aparte tabel.
9. De *tweede normaalvorm* ziet er dan als volgt uit. ***Woning*** (postcode, huisnr, vraagprijs, makelaarscode, makelaarsnaam); ***Woningbezoek*** (postcode, huisnr, datum_bezichtiging, tijd_bezichtiging, klantcode); ***Klant*** (klantcode, klantnaam).
10. *Derde normaalvorm*: het niet-sleutel attribuut ***makelaarsnaam*** in de tabel ***Woning*** is functioneel afhankelijk van het niet-sleutel attribuut ***makelaarscode***. Dit wordt ook een aparte tabel.
11. Het *strokendiagram* van de *derde normaalvorm* is als volgt.



6.11 Intuïtief of gestructureerd?

1. Bij het ontwerpen van een database hebben we inmiddels twee manieren laten zien om te komen tot een relationeel model.
   a. We zijn begonnen met een min of meer intuïtieve manier om te komen tot een logisch, conceptueel model, het *ERD*. Daarna keken we naar een implementatie van het *ERD*, waarbij we vreemde sleutels en eventuele tussentabellen aangaven. Dit werd vastgelegd in een *strokendiagram*.
   b. De tweede manier was via het normalisatie proces, waarbij we op een gestructureerde wijze tot een *ERD* en een *strokendiagram* kwamen.
2. Voor deze opleiding maakt het in het vervolg niet uit op welke wijze je tot je database ontwerp komt. Het gaat ons om het eindresultaat. Dat moet voldoen aan de *derde normaalvorm*. Je kunt de stappen van het normalisatieproces doorlopen óf je begint met de “intuïtieve” manier en dan controleer je aan het einde of je ontwerp in de *derde normaalvorm* is.

6.12 Opdrachten

1. Doe opdracht **Normaliseren.2 Normaliseren Gekke data** (niveau: gevorderd).
2. Doe opdracht **Normaliseren.3 Normaliseren Ziekenhuis** (niveau: basis).

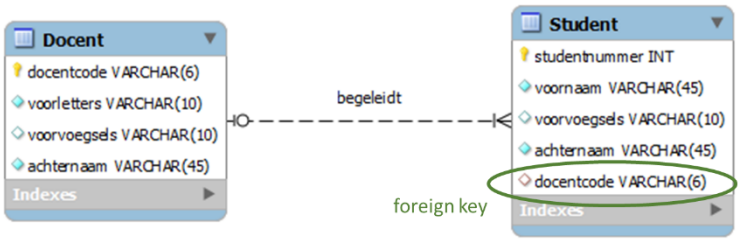
7 WERKEN MET WORKBENCH

7.1 Van conceptueel model naar implementatie

1. In een relationeel model ontbreekt nog een stukje implementatie in je database ontwerp. Je hebt alle tabellen met sleutels, attributen en eventuele vreemde sleutels. Wat je nog niet hebt zijn de *datatypes* van je attributen. Oftewel, wat voor soort attribuut is het? Is het een stuk tekst, is het een geheel getal, een kommagetal, een boolean (true/false), een datum, enzovoorts?
2. In een *database management systeem* zijn veel datatypes beschikbaar.
3. Bekijk de volgende video's:
   Columns and data types
   Numbers and other types
4. We gebruiken *Workbench* als tool om het conceptuele model te implementeren. Eén van de belangrijke aspecten van een implementatie is dat je de correcte datatypen gebruikt. Daarnaast moet je bij sommige datatypen ook de 'lengte' aangeven. De meest gebruikte datatypen zijn:
   a. *INT* voor gehele getallen.
   b. *DECIMAL*(totale lengte, aantal decimalen) voor kommagetallen. *DECIMAL*(6,2) wil zeggen dat je getallen tot 9999,99 kwijt kunt (vier cijfers voor de komma en twee cijfers achter de komma). De komma telt niet mee als karakter.
   c. *VARCHAR* voor tekst, standaard gebruikt *Workbench VARCHAR*(45).
   d. *DATE* voor data (in de vorm jjjj-mm-dd).
   e. *BOOLEAN* voor boolean (true/false). *Workbench* vertaalt dit standaard naar *TINYINT()*.

7.2 Symbolen in Workbench

1. Voordat we beginnen met *Workbench* lichten we een aantal symbolen toe, die *Workbench* gebruikt.
2. **Voorbeeld 1**.



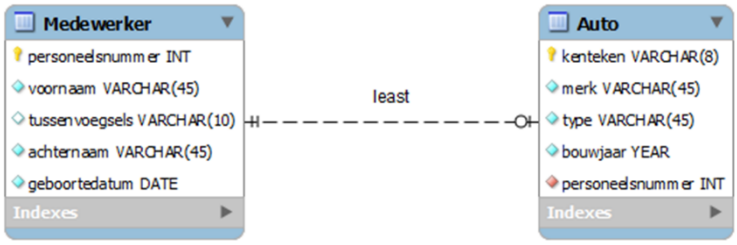
3. De gebruikte symbolen zijn:
   a. *Gele sleutel*: primaire sleutel attribuut. Bijvoorbeeld bij ***docentcode*** in de tabel ***Docent***.

b. *Blauwe gevulde ruit*: verplicht niet-sleutel attribuut. Bijvoorbeeld bij ***voorletters*** in de tabel ***Docent***.

c. *Blauwe open ruit*: optionele niet-sleutel attribuut. Bijvoorbeeld bij ***voorvoegsels*** in de tabel ***Docent***.

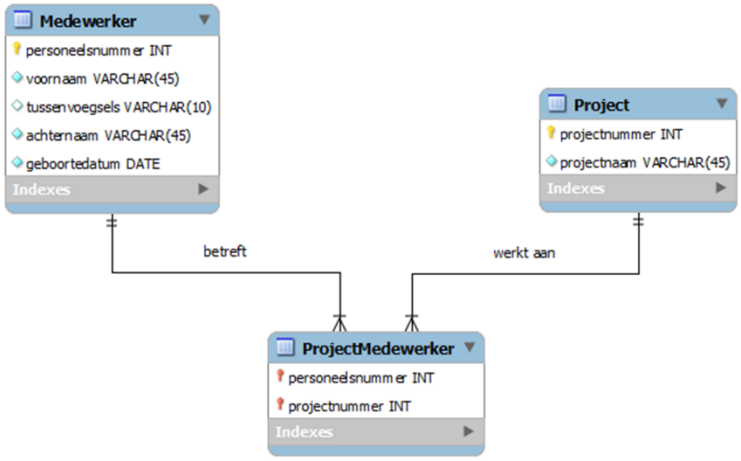
d. *Rode open ruit*: optionele vreemde sleutel attribuut. Bijvoorbeeld bij ***docentcode*** in de tabel ***Student***. Dit betekent dat in de tabel ***Student*** het attribuut ***docentcode*** niet gevuld hoeft te zijn (*optionaliteit*). Als het gevuld is dan moet het een waarde hebben, die verwijst naar de primaire sleutel van de tabel ***Docent*** en daar ook moet bestaan (*referentiële integriteitsregel*).

4. **Voorbeeld 2**.



a. *Rode gevulde ruit*: verplichte vreemde sleutel attribuut. Bijvoorbeeld ***personeelsnummer*** in de tabel ***Auto***. Deze kolom moet altijd een waarde hebben.

5. **Voorbeeld 3.**



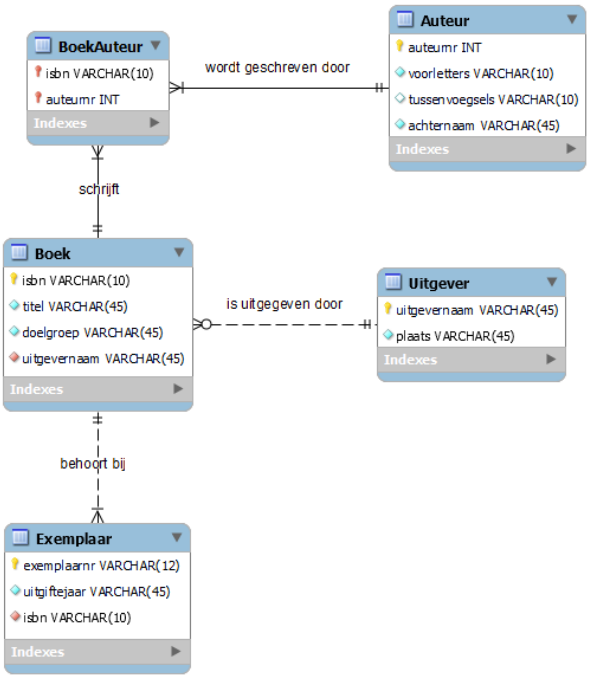
a. *Rode sleutel*: vreemde sleutel, die ook (onderdeel van een) primaire sleutel is. Bijvoorbeeld ***personeelsnummer*** en ***projectnummer*** in de tabel ***ProjectMedewerker***.

7.3 Opdracht

1. Als het goed is heb je *Workbench* al geïnstalleerd en heeft de docent al gecontroleerd dat dit goed gegaan is.
2. Doe de opdracht: **Werken met Workbench.1 ERD Pizzabezorging in Workbench** (niveau: basis).
   Deze opdracht leidt je stap voor stap door *Workbench* heen.

7.4 Oefening bibliotheek in Workbench

1. Maak het gegeven *ERD* in *Workbench*.



7.5 Uitbreiding van de bibliotheek

1. Een bibliotheek gaat ook om het uitlenen van boeken aan leden. De informatiebehoefte wordt nu uitgebreid met twee objecten op basis van de volgende voorbeelden.

Lidnummer:1281	
Naam :	René Vogels
Adres:	Willemstraat 3
Woonplaats:	Landsmeer
Geb. datum:	11 nov. 2012
Soort lid: Jeugd	

Lidnummer:1281
Lidnaam :René Vogels

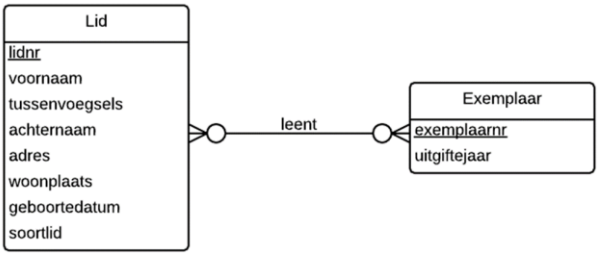
Exemplaarnummer:1111.1

Uitleendatum:02-03-2022
Terugdatum:23-03-2022

2. Op de voorbeeldkaarten zie je dat een ***Lid*** met ***lidnummer 1281*** een ***Exemplaar*** met ***exemplaarnummer 1111.1*** leent.

3. Wat is de relatie tussen ***Lid*** en ***Exemplaar***? *1-op-1*, *1-op-veel* of *veel-op-veel*?

4. Een ***Lid*** kan één of meer ***Exemplaren*** lenen en een ***Exemplaar*** kan uitgeleend worden aan meerdere ***Leden***. Er is dus een *veel-op-veel* relatie tussen ***Lid*** en ***Exemplaar***.



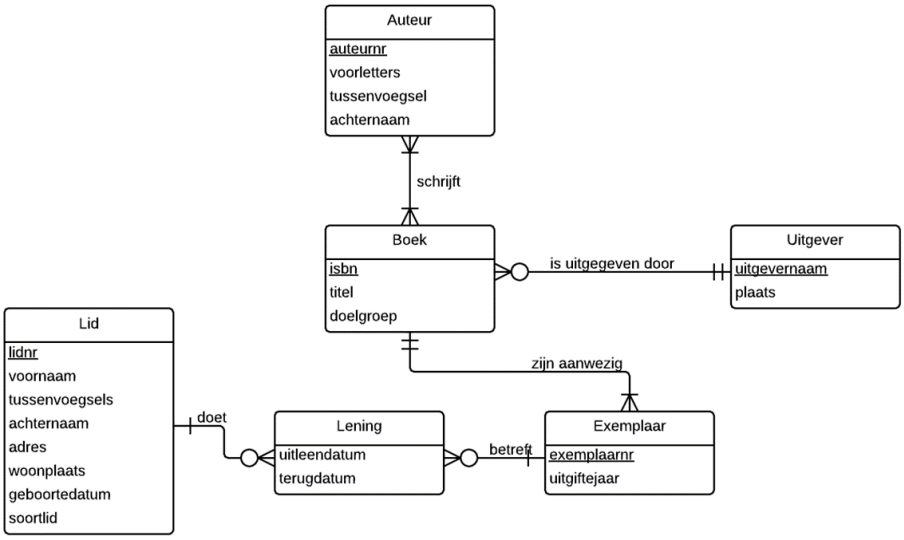
5. Waar horen de attributen ***uitleendatum*** en ***terugdatum***?

6. De attributen ***uitleendatum*** en ***terugdatum*** horen bij de relatie tussen ***Lid*** en ***Exemplaar***. Ze geven betekenis aan het feit dat een ***Lid*** een ***Exemplaar*** op een bepaalde datum heeft geleend en op een bepaalde datum heeft teruggebracht.

7. De attributen ***uitleendatum*** en ***terugdatum*** horen bij de ***Lening*** van een ***Exemplaar*** door een ***Lid***.

8. Bij de relatie horen dus attributen en dus wordt ***Lening*** een *entiteittype* (met een eigen tabel) in het logische model.

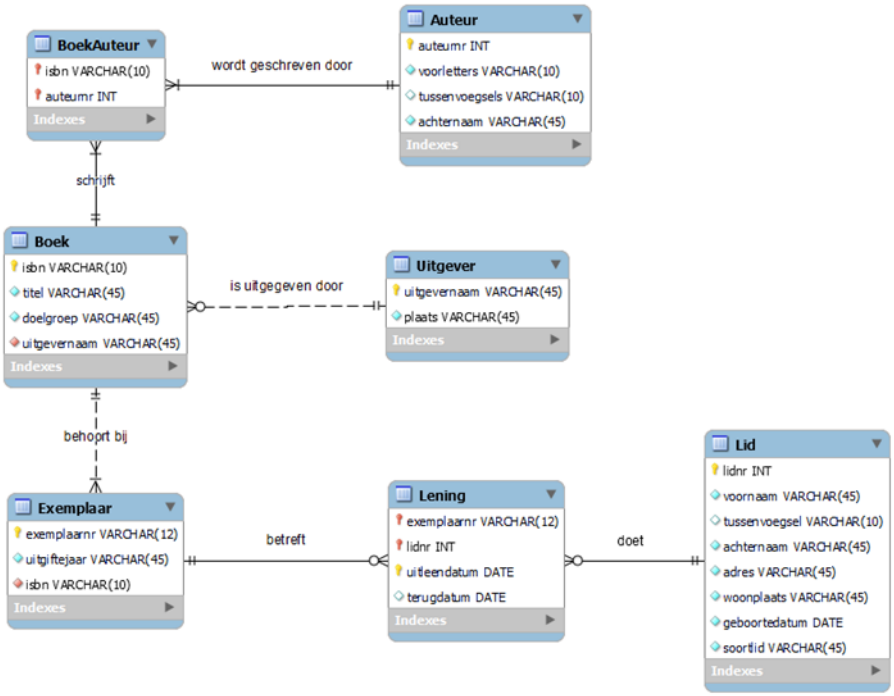
9. Het ERD ziet er dan als volgt uit.



10. In het *ERD* hebben we nog niet aangegeven hoe een entiteit van ***Lening*** geïdentificeerd kan worden. Wat zou de *identifier* kunnen zijn? Bedenk dat het in de praktijk mogelijk moet zijn dat een ***Lid*** hetzelfde ***Exemplaar*** meer dan één keer leent, dus de combinatie ***lidnr*** en ***exemplaarnr*** is niet voldoende.

11. Je hebt dus drie attributen nodig voor de *primaire sleutel*: ***lidnr***, ***exemplaarnr*** en ***uitleendatum***.

12. Het *ERD* in *Workbench* ziet er dan als volgt uit.



7.6 Dependent en independent entiteiten

1. Als de identificatie van een entiteit – zoals ***Lening*** – (mede) bepaald wordt door een andere entiteit – zoals ***Lid*** en ***Exemplaar*** – dan noemen we deze entiteit een *dependent entiteit*.
2. Dit is zichtbaar doordat een vreemde sleutel ook (een deel van de) primaire sleutel is en dus in *Workbench* wordt aangegeven met een rode sleutel.
3. De relatie noemen we een *identificerende relatie* of *identifying relation*.
4. In Workbench wordt dit aangegeven met een **doorgetrokken lijn**.
5. Als de identificatie van een entiteit niet afhankelijk is van een andere entiteit (in het voorbeeld van de bibliotheek is de identificatie van een ***Boek*** niet afhankelijk van de ***Uitgever***), dan noemen we deze entiteit een *independent entiteit*.
6. Dit is in *Workbench* het geval als alle eventuele verwijzingen (vreemde sleutels) aangegeven worden met een – al dan niet gevulde – rode ruit.
7. De relatie heet dan *niet-identificerend* of *non-identifying*.
8. Dit wordt in *Workbench* aangegeven met een **gestippelde lijn**.
9. Het blijkt dat een ***Exemplaar*** maar één keer per dag uitgeleend kan worden, omdat alle ingeleverde boeken pas de volgende ochtend teruggezet worden op hun plek in de bibliotheek.
10. Wat heeft dit voor consequentie voor de identificatie van een ***Lening***?

Örneğin, Lening varlığı (ödünç alma) kimliği açısından Lid (üye) ve Exemplaar (kitap nüshası) varlıklarına bağlıdır. Bu yüzden, Lening varlığının kimliği, Lid ve Exemplaar varlıklarının kimliklerine bağlıdır. Bu tür bir ilişki identificerende relatie (kimlik belirleyici ilişki) olarak adlandırılır ve Workbench'te bu ilişkiyi gösterirken koyu çizgi kullanılır.

Örneğin, bir Boek (kitap) varlığının kimliği, Uitgever (yayıncı) varlığına bağlı değildir. Boek varlığının kimliği kendi başına bağımsızdır.

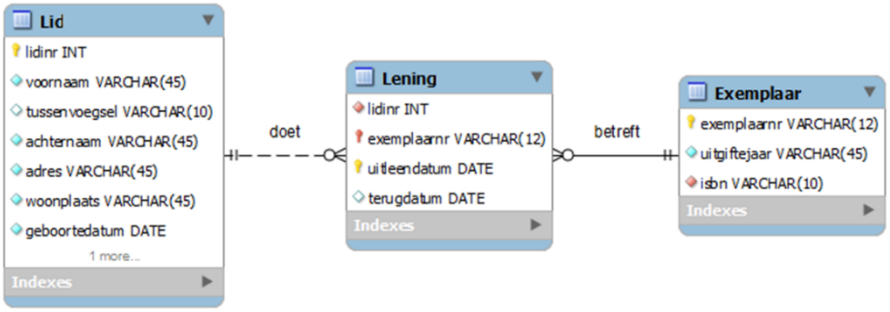
Bağımsız varlıkların ilişkileri non-identifying relation (kimlik belirleyici olmayan ilişki) olarak adlandırılır ve Workbench'te bu tür ilişkiler kesikli çizgi ile gösterilir.

lidnr	exemplaarnr	uitleendatum	terugdatum
1281	1111.0.1	02-03-2012	23-03-2012
1356	6824.0.1	14-02-2012	04-03-2012
[illegible]	6824.0.1	06-03-2012	06-03-2012
1267	6824.0.1	06-03-2012	09-03-2012
1356	6824.0.1	12-03-2012	21-03-2012

Verboden

11. De combinatie ***exemplaarnr*** en ***uitleendatum*** kan nu maar één keer voorkomen en kan daarom gebruikt worden als *identifier* voor ***Lening***.

12. Het relevante gedeelte van het *ERD* komt er dan zo uit te zien.



13. De *primaire sleutel* van ***Lening*** is nu de combinatie van ***exemplaarnr*** en ***uitleendatum***, dit levert een strengere *uniciteitsregel* op.

14. Merk op dat het attribuut ***lidnr*** in de tabel ***Lening*** niet langer een *sleutelattribuut* is. Het blijft een verwijzing (*vreemde sleutel*) naar de tabel ***Lid*** en wordt dus weergegeven met een rode ruit.

15. De relatie van ***Lid*** naar ***Lening*** is niet meer *identifying*. We hoeven het ***lidnr*** niet meer te weten om toch een rij in de tabel ***Lening*** te kunnen identificeren, het ***exemplaarnr*** en ***uitleendatum*** zijn voldoende.

7.7 Oefening Projectinzet

1. We hebben dit voorbeeld al in paragraaf 5.16 uitgewerkt in een *ERD* en *strokendiagram*.

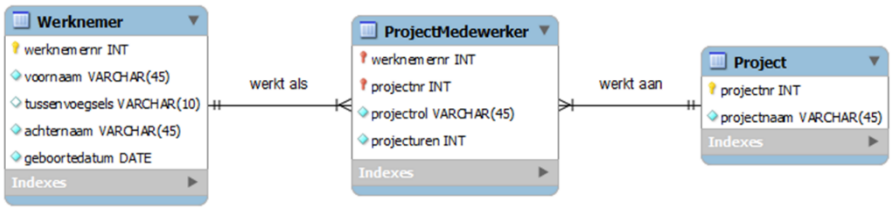
werk-nemernr	wnaam	wgebdat	project nr	pnaam	projectrol	projecturen
130	Cupido	16-06-1980	10 11	BTS De Zwart	Projectleider Analist	200 500
131	Doeksen	26-11-1984	10	BTS	Programmeur	500
145	Spanjer	15-07-1983	11 12	De Zwart D&D	Programmeur Programmeur	300 250
146	Wiersma	20-03-1982	10 11 12	BTS De Zwart D&D	Projectleider Analist Projectleider	200 180 100
156	Meijer	23-03-1986	12	D&D	DataArchitect	600

2. Maak het *ERD* in *Workbench*.

7.8 Herhaling dependent en independent entiteiten

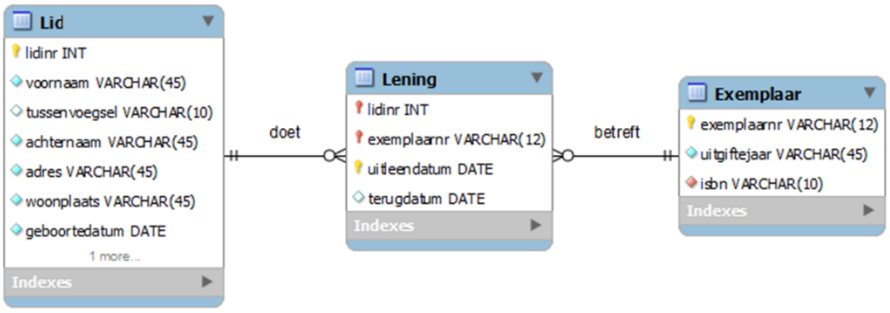
1. Omdat het concept van *dependent* en *independent entiteiten* en *identifying* en *non-identifying relations* belangrijk is, geven we nog enkele voorbeelden.

2. **Voorbeeld 1**



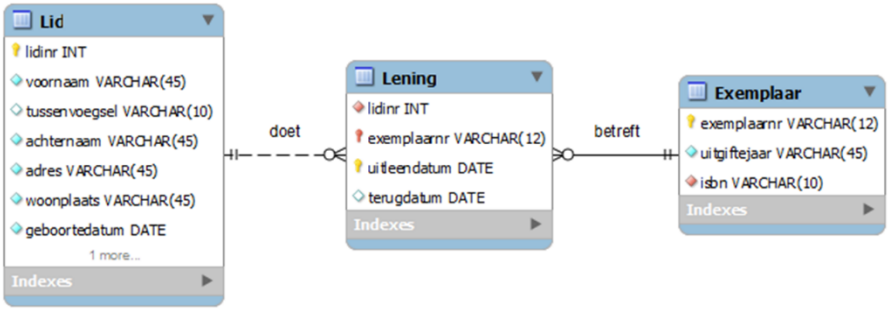
3. Hierboven wordt de entiteit ***ProjectMedewerker* volledig** bepaald door (de identificatie is volledig afhankelijk van) de entiteiten ***Project*** en ***Medewerker***. Beide relaties zijn daarom *identifying* en worden weergegeven met een doorgetrokken lijn.

4. **Voorbeeld 2**



5. Hierboven wordt de entiteit ***Lening* mede** bepaald door de entiteiten ***Lid*** en ***Exemplaar***. Omdat deze combinatie niet uniek hoeft te zijn (een ***Lid*** kan hetzelfde ***Exemplaar*** vaker lenen) komt er een eigen *identifier* ***uitleendatum*** bij. We hebben al eerder geconstateerd dat het nu mogelijk is om hetzelfde ***Exemplaar*** meer dan één keer per dag uit te lenen (aan een ander ***Lid***). Beide relaties zijn nog steeds *identifying* en worden weergegeven met een doorgetrokken lijn.

6. **Voorbeeld 3**



7. In het bovenstaande geldt de extra eis, dat een exemplaar maar één keer per dag uitgeleend mag worden. De entiteit ***Lening*** wordt **mede** geïdentificeerd door **één** gerelateerde entiteit. De identificatie van de entiteit ***Lening*** is niet meer afhankelijk

van de entiteit ***Lid***. De relatie tussen ***Lid*** en ***Lening*** wordt dan *non-identifying*, wat met een stippellijn wordt weergegeven.

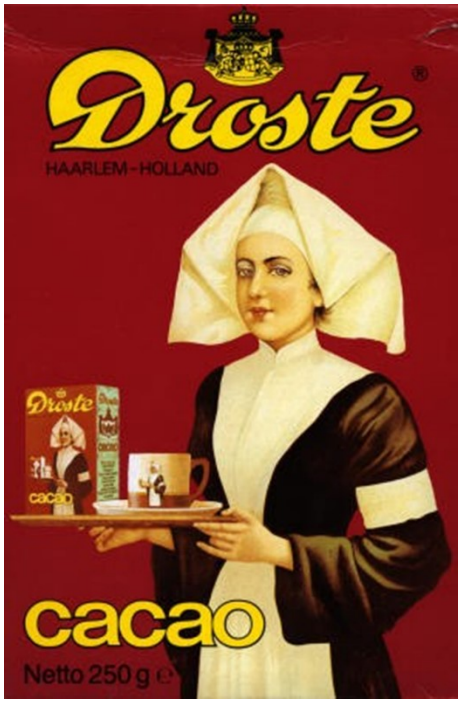
7.9 Opdrachten

1. Doe de opdracht **Werken met Workbench.2 ERD Landen in Workbench** (niveau: basis).
2. Doe de opdracht **Werken met Workbench.3 ERD Speelautomaten** (niveau: basis);

8 ERD VERDIEPING

8.1 Recursieve relaties

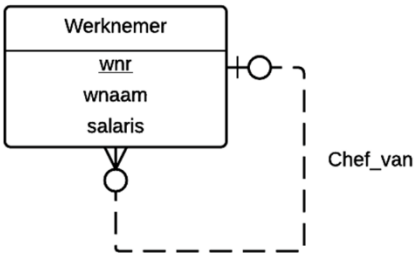
1. *Recursie* is het optreden van een constructie als onderdeel van zichzelf of verwijzing naar zichzelf. Het bekendste voorbeeld is ongetwijfeld de *Droste* verpakking, waar een afbeelding van de *Droste* verpakking opstaat, die weer een afbeelding van de *Droste* verpakking bevat.



2. In een database kan er een recursieve relatie zijn. Dat wil zeggen, een object heeft een relatie met een object van hetzelfde type.
3. Veel voorkomende recursieve relaties zijn:
   a. ***Werknemer 1372*** *is chef van* ***werknemer 1367***. Een werknemer heeft een chef, die zelf ook een werknemer is.
   b. ***Afdeling A34*** *is onderafdeling van* ***afdeling A12***. Een afdeling kan bestaan uit meerdere onderafdelingen, die zelf ook afdelingen zijn.
   c. ***Project GAT*** *is deelproject van* ***project POINT***. Een project kan bestaan uit meerdere deelprojecten, die zelf ook projecten zijn.
4. De tabel ***Werknemer*** in het bovenstaande voorbeeld zou er dan zo uit kunnen zien:

wnr	wnaam	salaris	chef
1367	J. de Groot	2346,00	1372
1368	C. Pekdemir	2067,00	1369
1369	N. Joosten	2258,00	1372
1370	A.D. Zonneveld	1947,00	1368
1371	M.N. El Kharimi	3348,00	
1372	G.F. de Smit	2815,00	1371

5. Uit de tabel is af te leiden dat ***N. Joosten*** de naam is van de chef van ***C. Pekdemir***.
6. In een *ERD* ziet een recursieve relatie er zo uit:

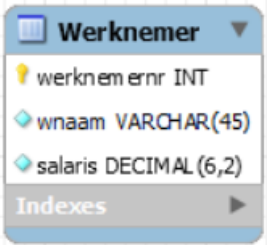


7. Waar komt de *vreemde sleutel*? Waarom zijn beide kanten niet verplicht?
8. Je kunt het al in het bovenstaande voorbeeld zien. De *vreemde sleutel* wordt toegevoegd aan de tabel zelf (als een kolom met de naam ***chef***) en verwijst daarmee naar de primaire sleutel van de eigen tabel!
9. Niet iedere werknemer is een chef (dit zijn de werknemers onderaan het organogram) en niet iedere werknemer heeft een chef (dat is de werknemer bovenaan het organogram).
10. Recursie in *Workbench*:
    a. De verwijzende sleutel die naar primaire sleutel van dezelfde tabel verwijst, moet een andere naam krijgen.
    b. *Workbench* doet dit automatisch, door de tabelnaam ervoor te zetten of door er een getal achter te zetten. Dit is afhankelijk van je instellingen.

    c. Hieronder laten we zien hoe je dat doet in *Workbench*.

8.2 Oefening recursie in Workbench

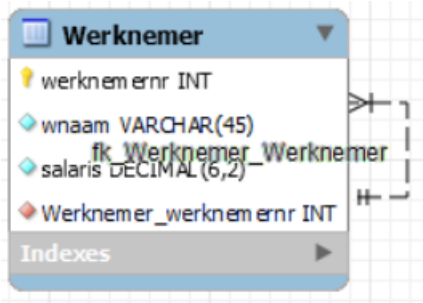
1. Start *Workbench* en maak een nieuw diagram.
2. Maak een tabel ***Werknemer*** zoals hieronder:



3. Klik op het 1 : n non-identifying symbool:

4. Klik vervolgens twee keer op de tabel ***Werknemer***.

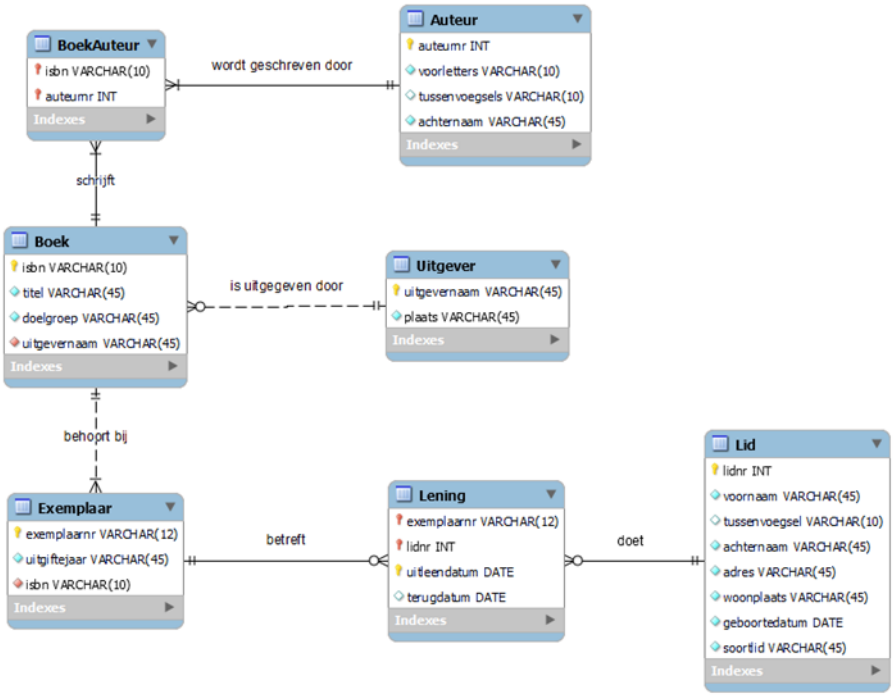
5. *Workbench* maakt de relatie en voegt een *caption* toe:



6. Je kunt nu het diagram aanpassen door de caption te veranderen en optionaliteiten toe te voegen aan beide kanten.

8.3 Identifying relatie (reprise)

1. In het voorbeeld van de bibliotheek zijn we gekomen tot het volgende diagram:



2. Kan een exemplaar volgens dit model meerdere keren per dag worden uitgeleend? Waaraan zie je dit?

3. Merk op dat de identificatie van een lening *afhankelijk* (*dependent*) is van zowel het ***exemplaar*** als het ***lid***. Het zijn allebei *identificerende relaties*.

4. Het voorbeeld van de bibliotheek kan uitgebreid worden, omdat er nog steeds redundantie is in het nummer van een exemplaar.

Auteur(s):	G.F. de Boer & N.Y. de Vries
Titel:	Het gegevensmodellenboek
Uitgever:	Cpress, Rotterdam
ISBN:	1111-0
Geschikt voor:	Jeugd
Exemplaren:	1111-0-1 (1998) 1111-0-2 (2001)

5. De kaart suggereert dat de regel bij de bibliotheek is dat een ***exemplaarnummer*** altijd bestaat uit ***ISBN*** + ***volgnummer***. Dus als het ***ISBN*** wijzigt, dan moet dit op meer dan één plek aangepast worden, namelijk in de *vreemde sleutel* ***ISBN*** en in de *primaire sleutel* ***exemplaarnr***. Kortom, ons model is nog niet correct.

6. **Merk op** dat we in het huidige diagram een *non-identifying relatie* hebben tussen ***Exemplaar*** en ***Boek***. We hebben voor de identificatie van een exemplaar geen *vreemde sleutel* ***ISBN*** nodig. Dit komt natuurlijk omdat in de primaire sleutel van ***Exemplaar*** de verwijzing naar het ***Boek*** is opgenomen.

7. Als we de redundantie verwijderen, dan krijgen we het volgende diagram:

8. ***Exemplaar*** is nu *dependent* van ***Boek*** en de relatie tussen ***Exemplaar*** en ***Boek*** is een *identificerende relatie*.

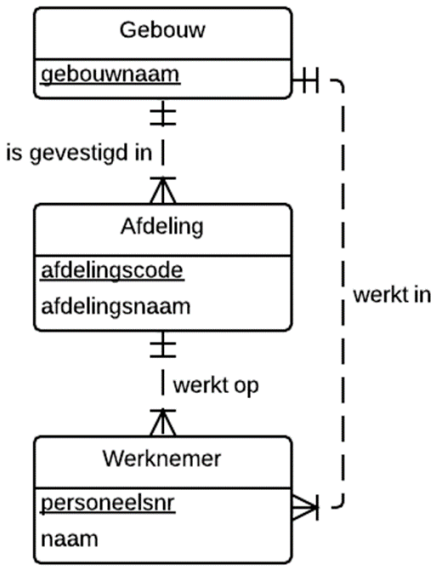
9. De *vreemde sleutel* in ***Lening*** die naar ***Exemplaar*** verwijst is nu *samengesteld*, omdat de *primaire sleutel* in ***Exemplaar*** *samengesteld* is. De *primaire sleutel* van ***Lening*** bestaat nu dus uit vier kolommen. Dit lijkt lastig, maar het heeft ook een voordeel: in ***Lening*** kun je direct zien om welk ***Boek*** het gaat, omdat ***ISBN*** onderdeel uitmaakt van ***Lening***.

8.4 Opdracht Hotel

1. Doe opdracht **ERD Verdieping.1 ERD Hotel** (niveau: basis).
2. Denk goed na over hoe je een entiteit kunt identificeren en wat dat dan zegt over de relaties (*identificerend* of *niet-identificerend*).

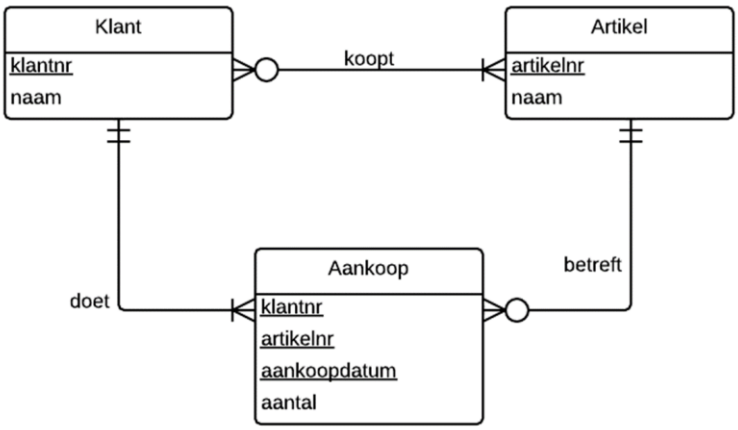
8.5 Loops

1. Als er een *loop* (*lus*) zit in je *ERD*, onderzoek dan of een van de relaties overbodig is.
2. **Voorbeeld 1**



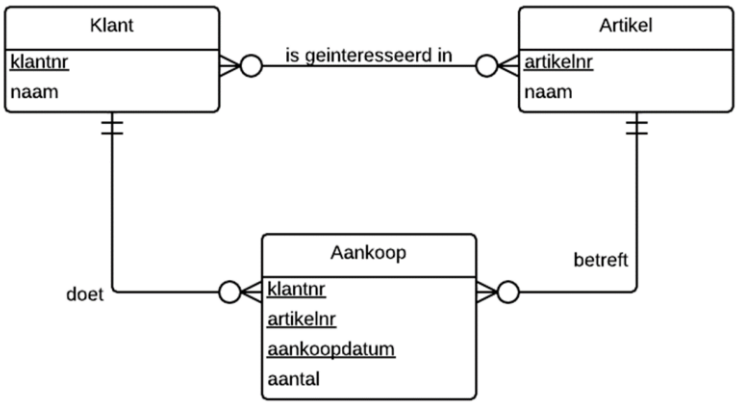
3. Hier geldt dat als een ***Werknemer*** op één ***Afdeling*** werkt en een ***Afdeling*** is in één ***Gebouw*** gehuisvest, dan is de relatie ***Werknemer*** – ***Gebouw*** overbodig. Je kunt de relatie immers in beide richtingen afleiden.

4. **Voorbeeld 2**



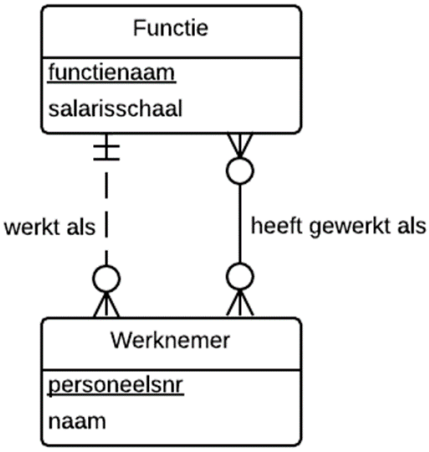
5. In dit voorbeeld is hetzelfde te zeggen over de relatie tussen ***Klant*** en ***Artikel***. Via ***Aankoop*** is af te leiden welke ***Artikelen*** een ***Klant*** heeft gekocht en welke ***Klanten*** een ***Artikel*** hebben gekocht.

6. **Voorbeeld 3**

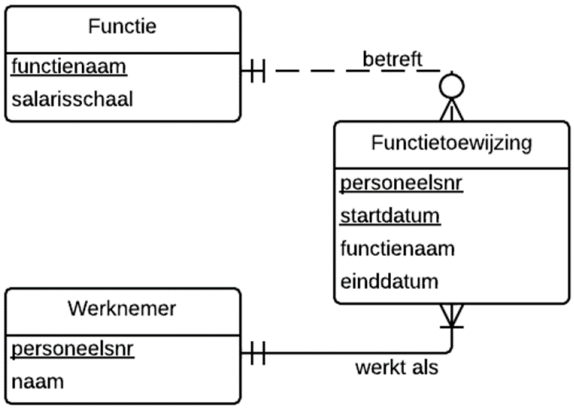


7. Hier krijgt de relatie tussen ***Klant*** en ***Artikel*** een andere betekenis: de klant toont interesse. Dan is de relatie wel nodig.

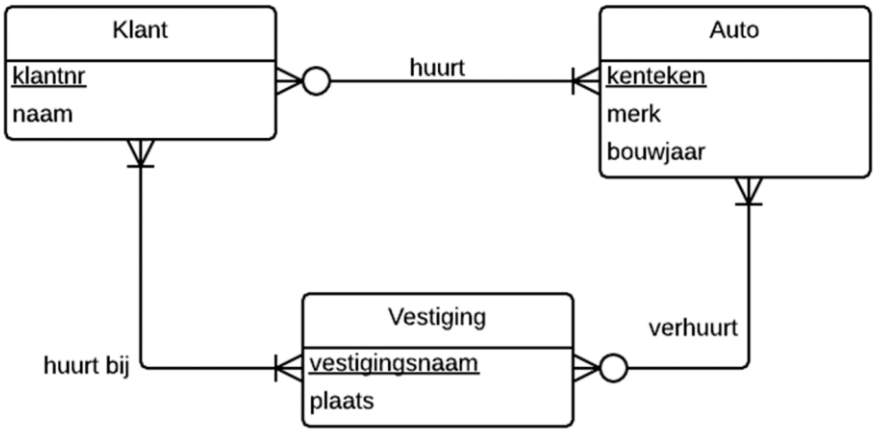
8. **Voorbeeld 4**



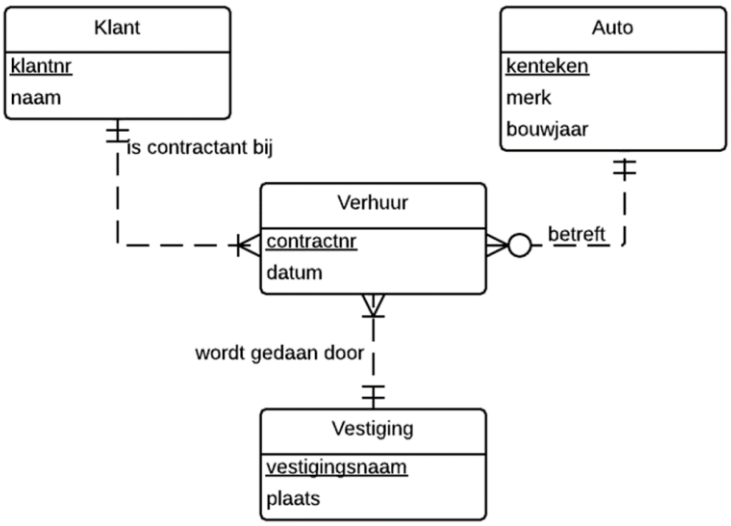
9. In dit voorbeeld willen we niet alleen weten welke functie een werknemer nu heeft, maar ook welke functies zij in het verleden hebben vervuld. In het model is dit getekend als twee verschillende relaties tussen ***Functie*** en ***Werknemer***.
10. Dit is ook een soort loop en is dus alleen goed als de betekenis van beide relaties verschillend is. Maar hier hebben beide relaties eigenlijk dezelfde betekenis, alleen is de één de huidige situatie en de andere de situatie in het verleden.
11. De gebruikelijkste oplossing hiervoor is het onderstaande model. Er is dan maar één relatie nodig. Deze situatie komt heel veel voor. Weliswaar heeft een werknemer op elk moment maar één functie, maar als je ook de functies in het verleden wil weten, is de relatie tussen ***Functie*** en ***Werknemer*** toch een *veel-op-veel* relatie:



12. Als de einddatum gevuld is dan betreft het een functietoewijzing uit het verleden. Als de einddatum leeg is, dan betreft het de huidige functie.
13. **Voorbeeld 5**

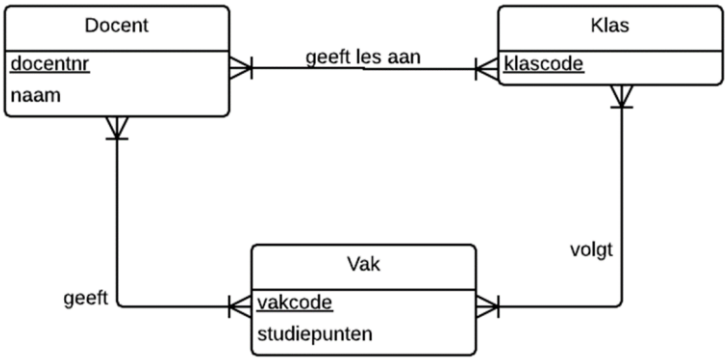


14. Het model gaat over een autoverhuurbedrijf met verschillende vestigingen. Weet je bij dit model, als een klant een auto huurt, bij welke vestiging hij/zij dit doet? Een loop met drie veel-op-veel relaties tussen drie entiteiten wijst vrijwel altijd op ‘een entiteit in het midden’, die de drie entiteiten met elkaar verbindt.



8.6 Oefening docent, klas en vak

1. Gegeven het volgende schema. Welke entiteit vervangt de drie veel-op-veel loops?



2. En hoe wordt zo'n entiteit geïdentificeerd?

8.7 Opdrachten

1. Doe de opdracht **ERD Verdieping.2 ERD Personeelsafdeling** (niveau: gevorderd).
2. Doe de opdracht **ERD Verdieping.3 ERD Vakantiepark** (niveau: basis);

9 SUBTYPERING

9.1 Subtypering: Medewerker en Manager

1. Stel, we hebben een tabel waarin we de gegevens van medewerkers vastleggen. Sommige medewerkers zijn manager en van de managers willen we extra attributen vastleggen.

Medewerker
medewerkernr
medewerkernaam
indienst_datum
salaris
mg_titel
mg_niveau
mgmtBonusVerdiend
mgmtBonusUitbetaald

2. De eerste vier attributen worden voor alle medewerkers ingevuld, de laatste vier attributen worden alleen ingevuld voor medewerkers, die manager zijn. Voor managers moet verplicht de ***mg_titel*** en ***mg_niveau*** worden ingevuld.
3. In een tabel ziet dit er dan zo uit.

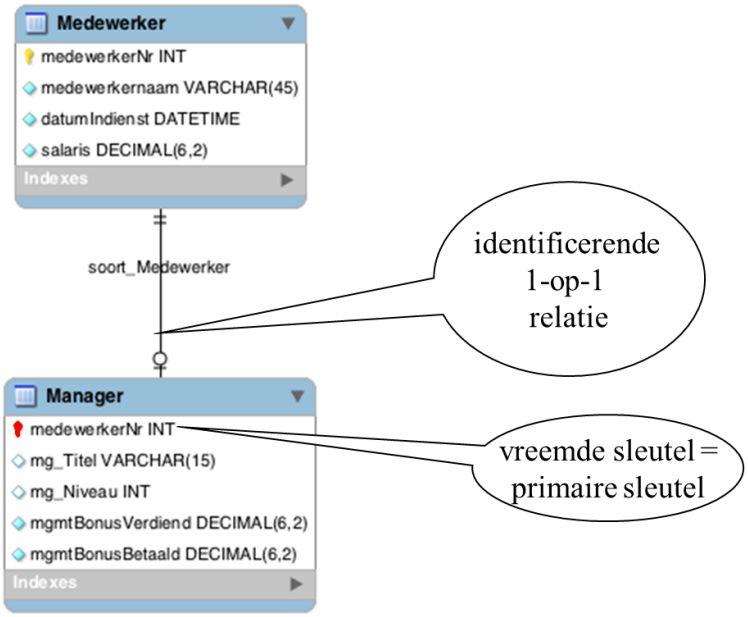
mwnr	mnaam	indienst	.....	mg_titel	mg_niveau	.....	.....
	Medewerker records				null waarden		
	Manager records			Manager informatie			

4. Dit is onmogelijk om te doen met één tabel. Voor de meeste medewerkers zullen de laatste vier kolommen een *NULL* waarde (dat betekent dat het veld helemaal leeg is, dus ook geen 0 of een lege tekenreeks) moeten hebben, terwijl sommige attributen voor sommige medewerkers verplicht ingevuld moeten worden. Voor een *DBMS* is een verplicht attribuut verplicht voor alle rijen en niet voor een selectie.
5. Managers vormen een deel(-verzameling) van alle medewerkers.
6. De oplossing voor dit probleem is om er twee tabellen van te maken. Maak een aparte tabel ***Manager*** met de manager specifieke attributen en houdt de oorspronkelijke tabel met de attributen, die voor iedereen gelden.

Medewerker
<u>medewerkernr</u>
medewerkernaam
indienst_datum
salaris

Manager
mg_titel
mg_niveau
mgmtBonusVerdiend
mgmtBonusUitbetaald

7. In de tabel ***Medewerker*** worden enkele zaken voor **alle** medewerkers (dus ook de managers) vastgelegd. In de tabel ***Manager*** worden de relevante zaken vastgelegd voor de ***Medewerkers*** die ***Manager*** zijn. Nu kunnen we de attributen ***mg_titel*** en ***mg_niveau*** verplicht maken.
8. De ***Manager*** is dus een ***Medewerker*** met eigen attributen (en/of relaties zoals we later zullen zien).
9. Merk op dat er geen *primaire sleutel* is voor de tabel ***Manager***. Wat zou de sleutel moeten zijn?
10. De ***Manager*** is ook een ***Medewerker*** en heeft dus een ***medewerkernr***, waarmee deze uniek te identificeren is.
11. **Oplossing**: de relatie tussen ***Medewerker*** en ***Manager*** is een *1-op-1* relatie, waarbij de *primaire sleutel* voor de tabel ***Manager*** gelijk is aan de *primaire sleutel* voor de tabel ***Medewerker***.
12. In *Workbench* ziet dit er dan zo uit:



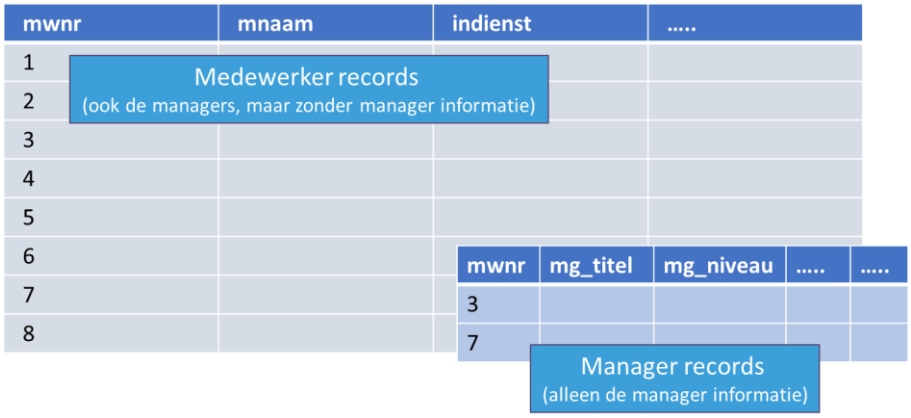
13. **Toelichting**:
    a. Deze oplossing wordt *subtypering* genoemd.
    b. De entiteit ***Medewerker*** wordt het *supertype* genoemd.
    c. De entiteit ***Manager*** wordt het *subtype* genoemd.

d. De *primaire sleutel* van het *subtype* is altijd identiek aan de *primaire sleutel* van het *supertype*.

e. De ***Manager*** is een bijzonder soort ***Medewerker*** en dat wordt ook zodanig aangegeven. Dit noemen we *overerving* (*inheritance*).

f. De relatie is altijd *identificerend* en *1-op-1* met een *optionaliteit* aan de subtype kant (niet iedere medewerker is een manager, maar iedere manager is wel een medewerker).

14. De tabellen zien er dan zo uit:



9.2 Subtypering: Artsen

1. De gedachte achter subtypering is ook toe te passen met meerdere subtyperingen, waarbij ieder subtype eigen attributen heeft, zoals in het volgende voorbeeld.

2. Een polikliniek heeft diverse artsen in hun artsenbestand. Elke ***Arts*** heeft een unieke code en naam.

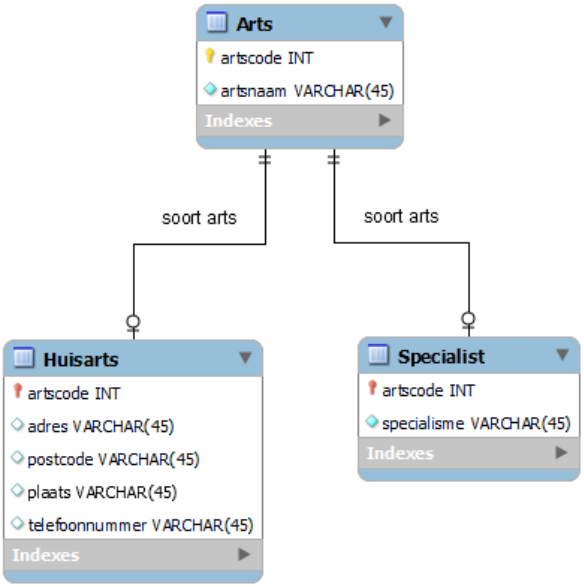
3. Er zijn twee soorten artsen:

   a. Een ***Huisarts*** heeft een adres en een telefoonnummer.

   b. Een ***Specialist*** heeft een verplicht in te vullen specialisme.

   c. Een ***Arts*** is óf ***Huisarts*** óf ***Specialist***

4. Ook nu passen we subtypering toe en in *Workbench* ziet het er dan zo uit:



9.3 Subtypering School

1. Een andere reden voor het toepassen van subtypering is wanneer een deel van de populatie een eigen relatie heeft, zoals in het volgende voorbeeld.
2. Op een school werken ***Personeelsleden***, van wie men een personeelsnummer, naam, adres, datum in dienst en salaris bijhoudt.
3. Sommige personeelsleden zijn ***Onderwijzende personeelsleden*** en daarvan houdt men de academische graad en het aantal lesuren bij.
4. Andere personeelsleden zijn ***Ondersteunende personeelsleden*** en daarvan houdt men bij welke ***Taken*** (denk aan roosters maken, cijferadministratie, personeelsadministratie, lokaalbeheer, enzovoorts) er in het takenpakket zitten. Omdat een ondersteunend personeelslid meerdere taken kan uitvoeren en taken door meerdere personeelsleden kunnen worden uitgevoerd is er een aparte tabel met ***Taak*** en een *tussentabel* ***Taaktoewijzing***. In *Workbench* is dit het resultaat.

9.4 Samenvatting subtypering

1. De redenen om over te gaan op *subtypering* zijn:
   a. Als voor sommige objecten geldt dat ze meer attributen hebben (de medewerker en de manager-medewerker).

b. Als de ene soort andere attributen heeft dan de andere soort van dezelfde entiteit (de huis-arts en de specialist-arts).

c. Als de ene soort andere relatie(s) heeft dan de andere soort van dezelfde entiteit (het onderwijs personeelslid en het ondersteunend personeelslid).

2. **Enkele tips**

a. Gebruik de naam van de *supertype* in de naam van het *subtype*. Bijvoorbeeld ***Huisarts*** en ***Onderwijspersoneelslid***.

b. Gebruik in de naam van de subtypering relatie ***soort_<naam_entiteit>***. Zie de voorbeelden: soort medewerker, soort arts, soort personeelslid

c. Bij het subtype is altijd de formulering ***<Subtype> is een soort <supertype>*** mogelijk: 'Huisarts is een soort arts' of 'Manager is een soort medewerker'.

3. **Let op**: in veel gevallen zullen de subtypes elkaar uitsluiten (je bent óf onderwijzend óf ondersteunend personeelslid. Je bent óf Huisarts óf Specialist. Het is denkbaar dat een deel van de populatie in beide subtypes voorkomt. Klanten en Medewerkers zijn Personen, maar een Medewerker kan ook een Klant zijn

9.5 Opdracht

1. Doe onderdelen a. en b. van opdracht **Subtypering.1 ERD Ebay Trader** (niveau: basis).

9.6 Implementatie van subtypering

1. Er zijn vier veel varianten om een subtyperingsrelatie te implementeren. De vier mogelijkheden werken we hieronder uit:

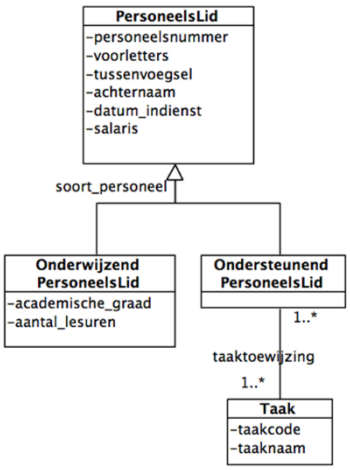
a. Alleen *generate parent*

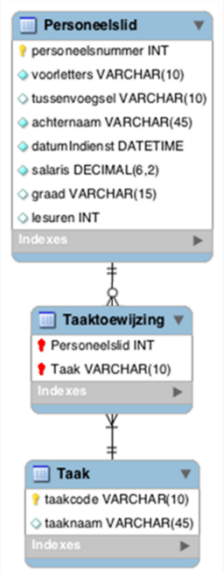
b. Generate parent, generate children, inherit all

c. Generate parent, generate children, inherit identifier

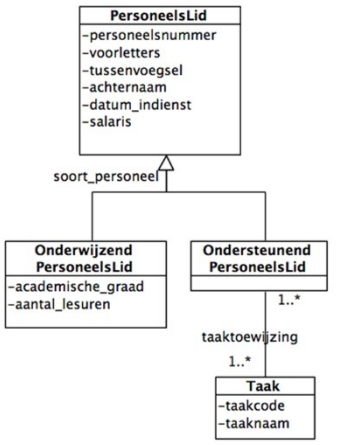
d. Alleen *generate children*

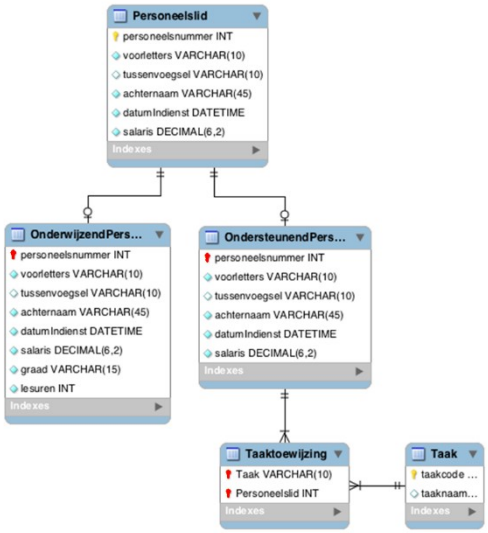
2. *Generate parent* houdt in dat je geen subtypering toepast en alleen de *parent* met alle mogelijke attributen houdt. Verplichte attributen van subtypes zijn dan niet af te leiden.



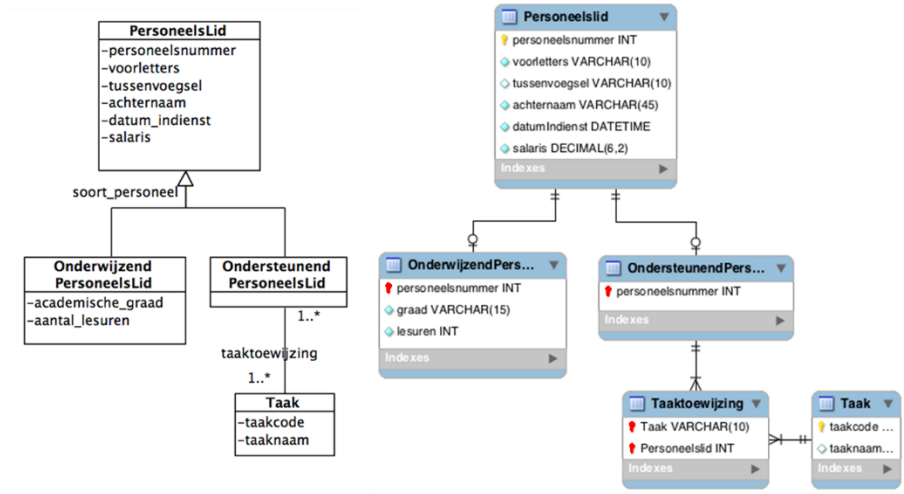


3. *Generate parent, children, inherit all* houdt in dat je alle attributen van de *parent* nog eens vastlegt in alle *children* (*subtypes*). Dit leidt tot heel veel redundantie.

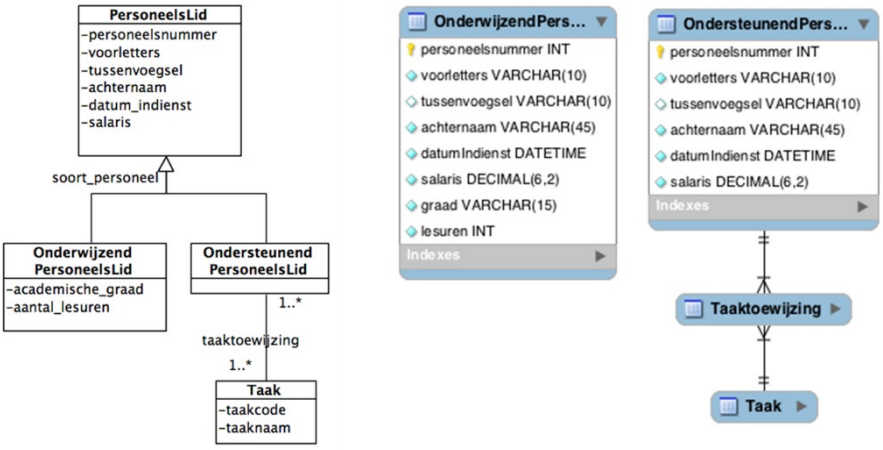




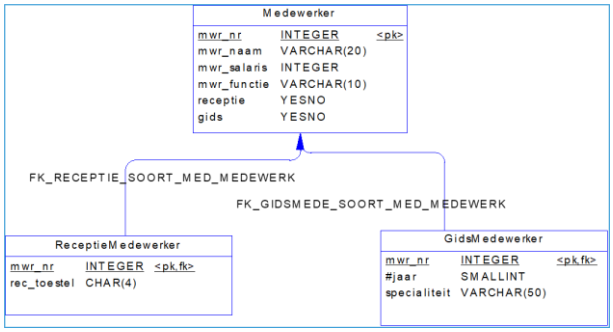
4. *Generate parent, children, inherit id* is de situatie zoals we die in het voorgaande hebben beschreven en is vaak de meest gewenste.



5. *Generate children* houdt in dat je losse tabellen maakt voor alle subtypes. Je moet dan in twee of meer tabellen gaan zoeken naar gegevens omtrent de medewerkers.



6. Je kunt er voor kiezen om in het *supertype* extra kolommen toe te voegen waarbij je per *subtype* bijhoudt of het bij een *subtype* hoort.



7. Een nadeel is dat je bij het toevoegen of verwijderen van een *subtype* de structuur van de tabel moet aanpassen.

9.7 Opdrachten

1. Doe de opdracht: **Subtypering.1 ERD Ebay Trader** (onderdeel c.) (niveau: basis).
2. Doe de opdracht: **Subtypering.2 ERD Luchtvaartmaatschappij** (niveau: basis).
3. En als je nog tijd en zin hebt kun je ook nog de volgende opdracht doen: **Subtypering.3 ERD Personeelsafdeling met subtypering** (niveau: gevorderd).

10 OEFENOPDRACHTEN

Tot slot hebben we nog drie oefenopdrachten:

1. **Oefenopdrachten.1 ERD Testvlucht** (niveau: basis).
2. **Oefenopdrachten.2 ERD Museum** (niveau: basis).
3. **Oefenopdrachten.3 ERD Extra** (niveau: basis).